

Specifying a forest harvest scheduling system which includes wood properties

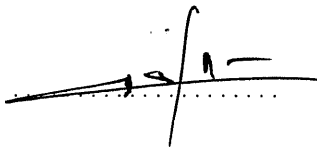
by

C. S. Price


Submitted in fulfillment of the academic
requirements for the degree of
Master of Science in the
School of Computer Science,
University of KwaZulu-Natal

June 2009

As the candidate's supervisor I have/have not approved this dissertation for submission

Signed:  Name: J. R. P. P. P. P. P. Date: 01 JUNE 2009

As the candidate's co-supervisor I have/have not approved this dissertation for
submission

Signed:  Name: F. C. BLAKEWAY Date: 01/06/2009

Abstract

This dissertation aims to specify a forest harvest scheduling system which includes wood properties in the harvesting decisions for the long-term (strategic) planning horizon. This system will be used by plantation forestry companies which supply wood to pulp manufacturers, who desire a more uniform raw material entering pulp mills so that a more uniform product results. Vertically integrated forestry companies would benefit particularly, as the allocation of timber to mills, as well as the timber transport costs, are included in the system.

It has been found from literature that only one forest harvest scheduling system exists which includes wood properties in the harvesting decision; however, this system was a short-term (operational) system. To our knowledge, no other system which includes wood properties in the harvesting decision has been reported.

As the forest harvest scheduling system is affected by the forest, transport, mill and forest planning domains, their procedures and constraints, these domains were described first, and the forest harvest scheduling system described next. The system and the environments (or domains) were specified with two techniques: semi-formal and formal methods. The semi-formal method used the Zachman framework to structure the specification. The Business owner's view of the system was used. This framework uses complementary models such as entity-relationship diagrams, business process diagrams and state charts to describe aspects of the same thing. The formal method specification used the Z notation which is based on set theory and predicate logic.

The semi-formal and formal specifications together form a complementary specification. The semi-formal specification is more understandable by clients, but could contain inconsistencies. The formal specification is more precise, but because it uses mathematical notation, is not as well understood. The semi-formal specification describes more features, while the formal specification describes the features in depth.

The forest harvest scheduling system specified uses wood properties in the harvesting and timber allocation decisions over the strategic planning horizon. When the system is implemented, wood having more uniform properties will be delivered to the mill, ensuring a more uniform pulp product.

Keywords:

plantation forestry, forest harvest scheduling systems, timber allocation, wood properties, semi-formal specification, Zachman framework (Business owner's view), formal specification, Z notation

Preface


The experimental work described in this dissertation was carried out in the School of Computer Science, University of KwaZulu-Natal, Durban, from September 2006 to May 2009, under the supervision of Prof. J.-R. Tapamo and Miss F. Blakeway.

These studies represent original work by the author and have not otherwise been submitted in any form for any degree or diploma to any tertiary institution. Where use has been made of the work of others it is duly acknowledged in the text.

Declaration 1 – Plagiarism

I, Catherine Susan Price, declare that

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain another person's data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain another person's writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
 - (a) Their words have been re-written but the general information attributed to them has been referenced
 - (b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References section.

Signed: 

Declaration 2 – Publications

DETAILS OF CONTRIBUTION TO PUBLICATIONS that form part and/or include research presented in this dissertation (including publications in preparation, submitted, in press and published, giving details of the contributions of each author to the experimental work and writing of each publication).

Publication 1

"Plantation forest harvest scheduling systems which include wood properties applicable to pulp mills: a review" by C.S. Price, F. Blakeway, F. Ahmed and J.-R. Tapamo, submitted to South African Journal of Science (Price *et al.*, 2008a). Contributions: C.S. Price: research and writing the paper; F. Blakeway: outline, editing; F. Ahmed: outline, editing; J.-R. Tapamo: outline, editing.

Publication 2

"Describing the plantation forestry domain" by C.S. Price, J.-R. Tapamo, F. Blakeway and F. Ahmed, abstract submitted to and accepted for V International Symposium of Sustainable Management of the Forestry Resources (SIMFOR 2008), Pinar Del Rio, Cuba, April 2008. Symposium not attended. Contributions: C.S. Price: writing the abstract, proposing outline; J.-R. Tapamo: advice on outline; editing; F. Blakeway: advice on outline, editing; F. Ahmed: advice on outline, editing.

Publication 3

"Plantation forestry: an analysis of the domain" by C.S. Price, J.-R. Tapamo, F. Blakeway and F. Ahmed, submitted to Annual Conference of the South African Institute of Computer Scientists and Information Technologists (SAICSIT 2008), Wilderness, South Africa, October 2008. Paper not accepted. Contributions: C.S. Price: developing specifications and diagrams, proposing outline, writing the paper; J.-R. Tapamo: specification review, suggestion of inclusion of Figure 1, advice on outline, editing; F. Blakeway: editing; F. Ahmed: editing.


Publication 4

"Plantation forestry: an analysis of the domain" by C.S. Price, J.-R. Tapamo, F. Blakeway and F. Ahmed, accepted for Domain Engineering Workshop to be held at the 21st International Conference on Advanced Information Systems Engineering (CAiSE'09),

Amsterdam, June 2009 (Price *et al.*, 2009). Contributions: C.S. Price: developing specifications and diagrams, proposing outline, writing the paper; J.-R. Tapamo: specification review, suggestion of inclusion of Figure 1, advice on outline, editing; F. Blake-way: editing; F. Ahmed: editing.

Publication 5

“Plantation forestry: an analysis of the domain” by C.S. Price, J.-R. Tapamo, F. Blake-way and F. Ahmed, incomplete (Price *et al.*, 2008b). Contributions: C.S. Price: devel-oping specifications and diagrams, proposing outline, writing the paper; J.-R. Tapamo: advice on outline, editing; F. Blakeway: advice on outline, editing; F. Ahmed: advice on outline, editing.

Signed: 

Contents

Abstract	ii
Preface	iv
Declaration 1 – Plagiarism	v
Declaration 2 – Publications	vi
List of Tables	xvi
List of Figures	xix
Glossary of forestry, wood and pulping property terms	xxiv
Glossary of Z notation used	xxxiv
Acknowledgements	xxxviii
1 Introduction and purpose of the study	1
2 Literature review	7
2.1 Literature included	7
2.2 Introduction	8
2.3 Forest-to-mill supply chain	10
2.3.1 Plantation forestry and planning	11
2.3.2 Transport	13
2.3.3 Mills	14
2.4 Wood properties	14
2.4.1 Wood properties	14

2.4.2	Wood property variation and its effect on end-product variation	15
2.4.3	Important wood properties for processes	17
2.4.4	Log sorting schemes aimed at reducing end-product variation .	20
2.5	Forest harvest scheduling and timber allocation systems	21
2.5.1	Background	21
2.5.2	Strategic planning	24
2.5.3	Strategic/tactical planning	26
2.5.4	Tactical planning	26
2.5.5	Operational planning	28
2.5.6	Linking plans in the planning hierarchy	30
2.6	Specifying a forest harvest scheduling system	34
2.6.1	Systems development	34
2.6.2	Methods for specifying systems	36
2.6.3	What is included in a specification?	42
2.6.4	Specifications of forest harvest scheduling systems in the literature	43
2.7	Conclusion	44
3	Methods and techniques	46
3.1	Introduction	46
3.2	Semi-formal analysis	48
3.3	Formal analysis	53
3.3.1	Choice of formal method	53
3.3.2	Introducing the Z notation	54
3.3.3	Notation conventions used in this dissertation	60
3.3.4	Method for developing Z specifications	60
3.3.5	Level of abstraction chosen for the specification	61
3.3.6	Tools used	62
3.4	Conclusion	63
4	Semi-formal specification	64
4.1	Introduction	64
4.2	Forest-to-mill domain	65
4.3	Plantation forestry domain	68

4.3.1	Summary of actions and constraints for the forestry domain . . .	69
4.3.2	Spatial layout of plantation forests and mills	70
4.3.3	Genetic material (“species”)	73
4.3.4	Regimes	74
4.3.5	Stand forestry	77
4.3.6	Forestry costs and income	87
4.3.7	Forestry and logistics staff	89
4.4	Transport domain	92
4.4.1	Summary of actions and constraints for the transport domain . . .	92
4.4.2	Transport in the forest-to-mill domain	93
4.4.3	Transport costs	96
4.5	Mill domain	97
4.5.1	Summary of actions and constraints for the mill domain	97
4.5.2	Mills and their processes	98
4.5.3	Mill costs and income	100
4.6	Forest planning domain	101
4.6.1	Summary of actions and constraints for the forest planning domain	103
4.6.2	Stand volume prediction	103
4.6.3	Annual forest volume prediction	106
4.6.4	Planning horizons and plan contents	107
4.6.5	Developing the plans	109
4.7	Forest harvest scheduling system	129
4.7.1	Aim of the system	129
4.7.2	System context	133
4.7.3	System inputs and outputs	136
4.7.3.1	System inputs	136
4.7.3.2	System outputs	140
4.7.4	Typical run of the system	141
4.7.5	Wood property prediction and measurement	146
4.7.6	Impact of implementing system on business processes	148
4.8	Conclusion	149

5	Formal specification	151
5.1	First abstraction: forest, transport, mill	153
5.1.1	Scope of the domain	153
5.1.2	Plantation forest	154
5.1.3	Transport	161
5.1.4	Mill	169
5.2	Second abstraction: forest, transport, mill	173
5.2.1	'Species' or genetic material	174
5.2.2	Regimes	174
5.2.3	Forestry company hierarchy	175
5.2.4	Volume or mass of logs produced when harvesting the stand . .	176
5.2.5	Logistics chain	178
5.2.6	Transporting the timber: roads	179
5.3	Forest harvest scheduling system	181
5.3.1	Planning horizon	182
5.3.2	Wood property prediction	183
5.3.3	Mill's requirements (constraints)	184
5.3.4	Cost inputs	186
5.3.5	Enumerating all the stand options	188
5.3.6	Enumerating all the possible solutions	192
5.3.7	Determining if a possible solution is feasible	193
5.3.8	Determining the objective function for all feasible solutions . .	197
5.3.9	Calculating the optimal solution	201
5.3.10	The forest harvest scheduling system	202
5.4	Conclusion	204
6	Discussion and conclusions	206
6.1	Introduction	206
6.2	Identification of domains, and their actions and constraints	207
6.3	Semi-formal specifications	207
6.4	Formal specifications	209
6.5	Combining both semi-formal and formal specifications	215
6.6	Forest harvest scheduling system specification	217

6.7	Conclusions	220
6.7.1	Contributions	222
6.7.2	Benefits of the contributions	223
6.8	Future work	225
6.9	Concluding remarks	227
A	Forest harvest scheduling systems/models in literature	228
B	Mathematical formulation of forest harvest scheduling systems/models in literature	241
C	Impact of the system on the domains	254
C.1	Forestry domain: stand lifecycle	254
C.2	Transport domain: separation of logs at depots and mills	257
C.3	Mill domain: criteria for accepting logs	258
C.4	Planning domain: contents of the plans	258
C.5	Planning domain: plan acceptance criteria	260
C.6	Planning domain: planning process	261
D	Formal specification of the forest-to-mill domain	272
D.1	Overview of the forest-to-mill supply chain	272
D.2	Forest management	274
D.2.1	Genetic material of trees	274
D.2.1.1	Preliminary definitions	275
D.2.1.2	Genetic material names	277
D.2.1.3	Genus	278
D.2.1.4	Species	278
D.2.1.5	Wood's genetic material	281
D.2.2	Regimes	281
D.2.2.1	Preliminary definitions	281
D.2.2.2	Date definition	282
D.2.2.3	Regime's genetic material	283
D.2.2.4	Regime's planting details	283
D.2.2.5	Regime's felling details	285

D.2.2.6	Tracking which is the current actual regime	286
D.2.2.7	Planned and actual regimes	287
D.2.2.8	Storing the regimes	289
D.3	Forest-to-mill logistics chain	290
D.3.1	Preliminary definitions	290
D.3.2	Logistics chain	290
D.3.3	Log mass in the logistics chain	292
D.3.4	At the stand's roadside	292
D.3.5	Depot	294
D.3.6	Mill	295
D.4	Plantation forestry	297
D.4.1	Forest management hierarchy	298
D.4.2	Stand's land	300
D.4.3	Stand of trees	302
D.4.4	Planting the stand	307
D.4.5	Harvesting the stand	310
D.4.6	Plantation actions	318
D.5	Transport	319
D.5.1	Preliminary definitions	319
D.5.2	Roads	320
D.5.3	Trucks	321
D.5.4	Short-haul transport	322
D.5.4.1	Short-haul trip	322
D.5.4.2	Load logs at stand (roadside)	323
D.5.4.3	Transport and unload logs at depot	325
D.5.4.4	Short-haul transport	326
D.5.5	Long-haul transport	326
D.5.5.1	Long-haul trip	327
D.5.5.2	Load logs at depot	327
D.5.5.3	Transport and unload logs at mill	329
D.5.5.4	Long-haul transport	330
D.5.6	Transport	330

D.5.7	Accepting logs from other suppliers	331
D.6	Processing	331
D.6.1	Removing logs for processing	331
D.7	Forest-to-mill supply chain	333
D.8	Log file generated by ZTC for the forest-to-mill domain	333
E	Formal specification of the forest harvest scheduling system	336
E.1	Planning horizon	336
E.2	Wood properties	337
E.3	Mill requirements	338
E.4	Costs and income	339
E.4.1	Preliminary definition	339
E.4.2	Forestry costs	340
E.4.2.1	Company-level forestry costs	340
E.4.2.2	Estate-level forestry costs	340
E.4.2.3	Regime-incurred (stand-level) forestry costs	341
E.4.3	Transport costs	343
E.4.4	Forestry income from mills	343
E.5	Stand options	344
E.5.1	Preliminary definitions	344
E.5.2	Defining the number of stand options	346
E.5.3	Defining the stand options	347
E.5.4	Populating the stand options	349
E.6	Possible and feasible solutions	357
E.6.1	Calculating the possible solutions	357
E.6.2	Calculating if the possible solutions are feasible	360
E.6.3	Determining if one or more feasible solutions exist	364
E.7	Calculating the costs of each stand option	366
E.7.1	Determining the first regime action per stand option	366
E.7.2	Calculating regime costs for each stand option	367
E.7.3	Calculating transport costs for each stand option	372
E.7.4	Calculating total costs for each stand option	377
E.8	Objective function	379

E.8.1	Preparatory calculations	379
E.8.1.1	Calculating the mills' delivered log cost	380
E.8.1.2	Calculating the cost for each possible solution	383
E.8.1.3	Calculating the total non-stand cost	385
E.8.2	Calculating the objective function for each feasible solution	388
E.9	Optimal solution	391
E.9.1	Calculating the optimal solution	391
E.9.2	Updating the schema <i>MillForStandsLogs</i> with the optimal solution	392
E.10	Forest harvest scheduling system	393
E.11	Log file generated by ZTC for the forest harvest scheduling system specification	395
F	Literature concept matrix	401
	References	407
	Index of schema names	431

List of Tables

2.1	Area of plantations by region and planting purpose in 2000 (from Carle <i>et al.</i> (2002))	9
2.2	Levels of planning in the forest supply chain (adapted from Rönqvist (2003))	11
2.3	Effects of biological and chemical wood properties on pulp and paper properties (from Gartner (2005))	18
2.4	Summary of sorting studies aimed at reducing wood property variation	21
3.1	Zachman framework (from Hay (2003, p.3))	50
4.1	Business owner's row in the Zachman framework (from Hay (2003, p.3))	65
4.2	Analyses undertaken from the Business owner point of view of the Zachman framework	66
4.3	Actions and constraints for the forester (grower)	69
4.4	Typical planned regime for growing trees for pulp	75
4.5	Typical costs per annum for a pulp regime	87
4.6	Actions and constraints for the transporter	92
4.7	Example of a stratified cost-per-tonne-per-km table	96
4.8	Actions and constraints for the pulp miller (processor)	97
4.9	Actions and constraints for the integrated forestry company staff when planning	102
4.10	Pseudocode for working out possible harvesting options for each stand	145
A.1	Product aims of strategic forest harvest scheduling systems & models found in literature	229
A.2	Product aims of strategic/tactical forest harvest scheduling systems & models found in literature	230
A.3	Product aims of tactical & tactical/operational forest harvest scheduling systems & models found in literature	231

A.4	Product aims of operational forest harvest scheduling systems & models found in literature	232
A.5	Background details of strategic forest harvest scheduling systems & models found in literature	233
A.6	Background details of strategic/tactical forest harvest scheduling systems & models found in literature	234
A.7	Background details of tactical & tactical/operational forest harvest scheduling systems & models found in literature	235
A.8	Background details of operational forest harvest scheduling systems & models found in literature	236
A.9	Included features of strategic forest harvest scheduling systems & models found in literature	237
A.10	Included features of strategic/tactical forest harvest scheduling systems & models found in literature	238
A.11	Included features of tactical & tactical/operational forest harvest scheduling systems & models found in literature	239
A.12	Included features of operational forest harvest scheduling systems & models found in literature	240
B.1	Mathematical formulation of strategic forest harvest scheduling systems & models found in literature	242
B.2	Mathematical formulation of strategic/tactical forest harvest scheduling systems & models found in literature	243
B.3	Mathematical formulation of strategic/tactical forest harvest scheduling systems & models found in literature (cont.)	244
B.4	Mathematical formulation of tactical forest harvest scheduling systems & models found in literature	245
B.5	Mathematical formulation of tactical/operational forest harvest scheduling systems & models found in literature	246
B.6	Mathematical formulation of operational forest harvest scheduling systems & models found in literature	247
B.7	Mathematical formulation of operational forest harvest scheduling systems & models found in literature (cont.)	248
B.8	Problem size & solution details of strategic forest harvest scheduling systems & models found in literature	249
B.9	Problem size & solution details of strategic/tactical forest harvest scheduling systems & models found in literature	250
B.10	Problem size & solution details of tactical & tactical/operational forest harvest scheduling systems & models found in literature	251

B.11 Problem size & solution details of operational forest harvest scheduling systems & models found in literature 252

B.12 Problem size & solution details of operational forest harvest scheduling systems & models found in literature (cont.) 253

F.1 Literature concept matrix 401

List of Figures

3.1	Outline of the method followed in generating the semi-formal and formal specifications	49
3.2	Example of how an entity-relationship diagram (ERD) should be read	51
3.3	Example of a business rule represented by an entity-relationship diagram	53
3.4	Relation between elements of two sets	55
3.5	Two ways of representing the same function (from Meyer (1985))	56
4.1	ERD giving an overview of the forest-to-mill supply chain	66
4.2	ERD giving more detail of the forest-to-mill supply chain	67
4.3	State chart giving an overview of the forest-to-mill supply chain	68
4.4	Spatial layout of a plantation forest, showing hierarchy of management levels	71
4.5	Hierarchy of forestry company management levels	72
4.6	ERD giving forest land ownership options	72
4.7	Hierarchy of genetic material (loosely called “species”)	73
4.8	Diagram showing the relationship between pure species and hybrids, seedlings, cuttings and clones	74
4.9	ERD showing the factors on which a tree’s growth on a particular stand depends	78
4.10	ERD of the characteristics of a monoculture plantation forestry stand	79
4.11	State transition chart showing the possible states of a stand	79
4.12	Business process diagram of the stand’s lifecycle (part 1 of 4)	80
4.13	Business process diagram of the stand’s lifecycle (part 2 of 4)	82
4.14	Business process diagram of the stand’s lifecycle (part 3 of 4)	83
4.15	Business process diagram of the stand’s lifecycle (part 4 of 4)	84
4.16	Graph showing the typical volume growth of a stand of trees over time	85
4.17	ERD showing costs involved in the forestry domain	88

4.18 ERD showing income options for the forestry domain	88
4.19 Organogram of forestry staff, showing area of responsibility	89
4.20 ERD showing responsibilities of forestry company staff	90
4.21 ERD showing responsibilities of the Timber logistics manager	91
4.22 ERD showing the different transport type and method options	93
4.23 ERD showing the transport of logs to the mill	94
4.24 ERD showing the loading point for different long-haul transport methods	94
4.25 ERD showing the relationship between timber volume and its mass . .	95
4.26 ERD of the costs involved in the transport domain	96
4.27 ERD showing the entities of the mill	98
4.28 ERD giving a mill's criteria for accepting logs	98
4.29 ERD showing features of mills' processes	100
4.30 ERD showing the costs involved in the mill domain	101
4.31 ERD giving inputs to the Growth & yield simulator before enumeration	104
4.32 ERD giving inputs to the Growth & yield simulator after enumeration .	105
4.33 Age-class distribution which is nearly normalised	106
4.34 Relationship between the strategic, tactical and operational plans and their planning horizons	107
4.35 ERD showing the contents of the strategic, tactical and operational plans	108
4.36 ERD showing the development of the strategic, tactical and operational plans	109
4.37 State chart showing the development of the strategic, tactical and op- erational plans	110
4.38 ERD showing the criteria on which the strategic plan is assessed by the Planning forester	111
4.39 ERD showing the criteria on which the tactical and operational plans are assessed by the Co-ordinating and Managing foresters	111
4.40 ERD showing the criteria on which the tactical and operational plans are assessed by the Timber logistics manager	113
4.41 Business process diagram of the planning process (part 1 of 18)	114
4.42 Business process diagram of the planning process (part 2 of 18)	115
4.43 Business process diagram of the planning process (part 3 of 18)	116
4.44 Business process diagram of the planning process (part 4 of 18)	117
4.45 Business process diagram of the planning process (part 5 of 18)	118

4.46	Business process diagram of the planning process (part 6 of 18)	119
4.47	Business process diagram of the planning process (part 7 of 18)	120
4.48	Business process diagram of the planning process (part 8 of 18)	121
4.49	Business process diagram of the planning process (part 9 of 18)	121
4.50	Business process diagram of the planning process (part 10 of 18) . . .	122
4.51	Business process diagram of the planning process (part 11 of 18) . . .	124
4.52	Business process diagram of the planning process (part 12 of 18) . . .	125
4.53	Business process diagram of the planning process (part 13 of 18) . . .	125
4.54	Business process diagram of the planning process (part 14 of 18) . . .	126
4.55	Business process diagram of the planning process (part 15 of 18) . . .	126
4.56	Business process diagram of the planning process (part 16 of 18) . . .	127
4.57	Business process diagram of the planning process (part 17 of 18) . . .	127
4.58	Business process diagram of the planning process (part 18 of 18) . . .	128
4.59	Context diagram giving an overview of the forest harvest scheduling system (FHSS)	134
4.60	Business process diagram showing typical use of the FHSS (part 1 of 2)	142
4.61	Business process diagram showing typical use of the FHSS (part 2 of 2)	143
4.62	Diagram illustrating the combinatorial problem of the FHSS	144
4.63	ERD showing the inputs to a wood property model	146
4.64	Graph showing the spread of wood property values entering a mill . . .	148
5.1	Development of the contents of this chapter, showing how the formal specification for the FHSS was developed	152
5.2	Main entities and actors of the domain	153
5.3	State chart of the forest-to-mill domain's scope modelled in the Z spec- ification	154
5.4	Hierarchy of genetic material (loosely called "species") modelled in the Z specification	174
5.5	Diagram showing the relationship between pure species and hybrids mod- elled in the Z specification	175
5.6	Hierarchy of forestry company management levels modelled in the Z specification	176
5.7	Stand options for an unplanted stand, where there are three possible destination mills	190

5.8	Stand options for a planted stand, where there are three possible destination mills	191
5.9	A possible solution for the forest harvest scheduling system	192
5.10	Calculation of the annual mass of logs destined for each possible mill, for a particular possible solution	195
5.11	Calculation of the objective function, for a particular possible solution .	200
C.1	Updated business process diagram of the stand's lifecycle (part 1 of 4)	255
C.2	Updated business process diagram of the stand's lifecycle (part 2 of 4)	255
C.3	Updated business process diagram of the stand's lifecycle (part 3 of 4)	256
C.4	Updated business process diagram of the stand's lifecycle (part 4 of 4)	256
C.5	ERD showing the transport of logs to the mill, including sorting by mill process	257
C.6	ERD giving a process' criteria for accepting logs	258
C.7	ERD showing the updated contents of the strategic, tactical and operational plans	259
C.8	ERD showing the updated criteria on which the strategic plan is assessed by the planning forester	260
C.9	ERD showing the updated criteria on which the tactical and operational plans are assessed by the Co-ordinating and Managing foresters	260
C.10	ERD showing the updated criteria on which the tactical and operational plans are assessed by the Timber logistics manager	261
C.11	Updated business process diagram of the planning process (part 1 of 18)	262
C.12	Updated business process diagram of the planning process (part 2 of 18)	263
C.13	Updated business process diagram of the planning process (part 3 of 18)	263
C.14	Updated business process diagram of the planning process (part 4 of 18)	264
C.15	Updated business process diagram of the planning process (part 5 of 18)	264
C.16	Updated business process diagram of the planning process (part 6 of 18)	265
C.17	Updated business process diagram of the planning process (part 7 of 18)	265
C.18	Updated business process diagram of the planning process (part 8 of 18)	266
C.19	Updated business process diagram of the planning process (part 9 of 18)	266
C.20	Updated business process diagram of the planning process (part 10 of 18)	267
C.21	Updated business process diagram of the planning process (part 11 of 18)	267
C.22	Updated business process diagram of the planning process (part 12 of 18)	268
C.23	Updated business process diagram of the planning process (part 13 of 18)	268

C.24	Updated business process diagram of the planning process (part 14 of 18)	269
C.25	Updated business process diagram of the planning process (part 15 of 18)	269
C.26	Updated business process diagram of the planning process (part 16 of 18)	270
C.27	Updated business process diagram of the planning process (part 17 of 18)	270
C.28	Updated business process diagram of the planning process (part 18 of 18)	271
D.1	ER diagram giving an overview of the forest-to-mill supply chain	272
D.2	State chart giving an overview of the forest-to-mill supply chain	273
D.3	State chart giving an overview of the forest-to-mill supply chain modelled in the Z specifications	274
D.4	Hierarchy of genetic material (loosely called "species")	275
D.5	Diagram showing the relationship between pure species and hybrids, seedlings, cuttings and clones	276
D.6	Hierarchy of genetic material (loosely called "species") modelled in the Z specification	276
D.7	Diagram showing the relationship between pure species and hybrids mod- elled in the Z specification	277
D.8	Hierarchy of forestry company management levels	298
D.9	Hierarchy of forestry company management levels modelled in the Z specification	299
E.1	Stand options for an unplanted stand, where there are three possible destination mills	350
E.2	Stand options for a planted stand, where there are three possible desti- nation mills	352
E.3	A possible solution for the forest harvest scheduling system	359
E.4	Calculation of the annual mass of logs destined for each possible mill, for a particular possible solution	362
E.5	Calculation of the objective function, for a particular possible solution .	390

Glossary of forestry, wood and pulping property terms

The following are definitions of forestry and wood properties terms which are found in this dissertation. A more complete list of forestry and forest products terms can be found in the work which Ford-Robertson edited 1971 and its update, edited by Winters (1977). Another source of forestry definitions is the glossary of the South African Forestry Handbook (Owen, 2000a). Some wood properties and pulping terminology is given in the dissertation of Naidu (2003, pp.xxviii–xxxv), while a more complete list of paper and pulp and paper property definitions can be found in the APPA Dictionary of Paper (American Paper and Pulp Association, 1965).

Term	Definition
Afforestation:	Converting land into, or planting with, a forest (Sykes, 1982).
Age-class distribution:	Graph of area versus age for a working circle which shows how much timber will be available for harvesting over time.
Age of trees in a stand:	The number of years since establishment (planting or coppicing) has occurred. Measured in years.
Altitude:	Height above sea level (in m).
APO (annual plan of operations):	Short-term plan for managing the forest. As its name implies, covers a planning horizon of one year.
Aspect, stand ~ :	Which cardinal direction (North, South, East, West) the stand faces.

Term	Definition
Available hours p.a.:	The number of hours the process of a mill can work per annum (in hours).
Blanking:	Replacing the dead cuttings or seedlings (of same genetic material) in a newly planted stand. Undertaken shortly after planting.
Brightness:	“The reflectance or brilliance of the paper when measured under a specially calibrated blue light. Defined as the amount of bluewhite reflectance compared to magnesium oxide, which is considered 100% bright” (Naidu, 2003, p.xxxi). Expressed as a percentage.
Bucking:	The act of cutting up a tree into logs. Done using the guiding instructions of a bucking pattern.
Bucking pattern:	Rules of how to cut up a tree into logs. Consists of at least one log type specification, with minimum and maximum dimensions.
Canopy closure:	When the branches of the trees planted in a stand form a “roof”, thus receiving most of the sunlight falling on the stand.
Certification status:	A record of whether or not a particular management level (e.g. estate, stand) has been managed according to the guidelines given in the certification system.
Certification system:	An audited (certified) set of rules and guidelines for growing trees and managing the land on which they are grown in an environmentally sound way. An example of a certification system is Forestry Stewardship Council (FSC) (Forest Stewardship Council, 2003a).
Chemical pulping:	Using chemical means to form pulp by dissolving the lignin which binds the wood fibres. See also Kraft pulping.
Clearfelling:	see Harvesting.
Clone:	“Plants produced vegetatively from one original seedling or stock” (Sykes, 1982).

Term	Definition
Collapsibility, fibre ~ :	A measure of the ability of each wood fibre to be flattened in the paper forming process. The flatter the fibre, the more surface area it will have to bond with other fibres, thus making stronger paper.
Compartment:	see Stand.
Co-ordinating forester:	Forester who oversees Managing foresters and co-ordinates forestry activities at a high management level (e.g. at a regional level).
Coppice:	Shoots which grow from a live tree stump after harvesting has occurred. Some species do not coppice.
Coppice reduction:	The removal of the smaller or weaker shoots which have grown from a tree stump after harvesting.
Cross-cutting:	see Bucking.
Cutting:	“Plant material which was cut from another plant for propagation” (Sykes, 1982).
Day length:	The amount of time per day when there is sunlight.
DBH:	see Diameter at breast height.
Delignification:	The extent to which the lignin has been removed from the wood when pulped.
Density, wood ~ :	Relative mass of the wood, measured in kg/m ² .
Depot:	Place in or near the estate where logs are stored briefly before they are transported long-haul to the mill.
Diameter at breast height (DBH):	Diameter of a tree 1.3m from the ground (measured in cm).
Dissolving pulping:	A chemical method of pulping which uses a acid sulphite or acid bisulphite solution to dissolve the lignin. See also Chemical pulping.
Distance:	Measured in km.
End use:	Purpose for which plantation is primarily planted (e.g. pulp, saw timber, poles, etc.).

Term	Definition
Enumeration:	Measuring the dimensions (diameter, height, stems per hectare, etc.) of a stand of trees.
Establishment:	Activities involved with either planting or growing from coppiced shoots. Includes Blanking.
Estate:	A group of stands which is managed together as a unit. It is overseen by a Managing forester.
Exotic:	Plant “introduced from abroad” (Sykes, 1982). Opposite of Indigenous.
Extractive:	Any compound in the wood which is soluble in a neutral organic solvent or water. Extractives can build up on pulping equipment, leading to blockages, lower production time and higher manufacturing costs. High extractive content in wood can also lead to reduced pulp quality.
Felling:	see Harvesting.
Forest Certification:	see Certification system.
FSC:	Forest Stewardship Council: certification system for growing trees and managing the land on which they are grown in an environmentally sound way.
Fibre, wood ~ :	A wood cell, made up largely of cellulose. The pulping process aims to disassociate them from each other so that they can be formed into flat paper.
Genera:	Plural of genus – see Genus.
Genetic material:	Generic term used to denote the genus, species, hybrid, clone, seedling origin of a tree. See also Genotype.
Genotype:	“Genetic constitution of an individual” (Sykes, 1982).
Genus:	“Group of plants having common structural characteristics distinct from all other groups, usually having several species” (Sykes, 1982).
Harvesting:	The act of cutting down all the trees in a stand. Same as Felling and Clearfelling.

Term	Definition
Harvest production rate:	The rate at which certain harvesting methods and equipment can fell a hectare of trees.
Harvest scheduling system:	A decision support system which decides which stands of trees to fell, when.
Hybrid:	“Offspring of two plants of different species or varieties” (Sykes, 1982).
Incidence of frost:	Whether frost occurs or not in a stand, estate, or higher management level.
Indigenous:	Plant which belongs “naturally” to a region (Sykes, 1982). Opposite of Exotic.
Intensively managed plantation forestry:	An approach to plantation forestry where tree breeding (tree improvement) and silviculture is used to produce crops of trees.
K number:	A measure of the residual lignin in dissolving pulp.
Kappa number:	A measure of the residual lignin in chemical pulp.
Kraft pulping:	A chemical method of pulping which uses an alkaline solution to dissolve the lignin. Also called Sulphate pulping. See also Chemical pulping.
Lignin:	The substance in wood which holds fibres of trees together.
Log type:	Type of log which could be made (e.g. pulp log, saw log). Related to end use, but could include more detail than only the end use definition.
Logyard:	Storage area for logs delivered to a mill.
Long-haul transport:	Transport where long distances are involved (e.g. from the Depot or Siding to the Mill). Could be by road or rail.
MAI (Mean Annual Increment):	A measure of the growth rate of the stand.

Term	Definition
Maintenance:	Activities in a stand or estate, like building firebreaks, checking for disease or pests. Maintenance activities occur annually.
Managing forester:	Manages an estate, or group of estates.
Management level:	Level in the forestry company's landholding hierarchy. There could be many levels, depending on the company. The lowest level is the stand and the second lowest is the estate. All levels have a parent level, except for the forestry company, which is the top node in the hierarchy.
Mass of logs:	Measured in tonnes.
Mean annual precipitation:	Average rainfall.
Mean annual temperature:	Average temperature.
Mechanical pulping:	Using mechanical means to form pulp by grinding the wood fibres from each other.
Merchandising:	see Bucking.
Merchantable volume:	Volume of a tree or stand of trees which, when harvested and cut into logs, can be sold and used (by a mill).
Mill logyard:	see Logyard.
Natural regeneration:	Regrowth of trees from the seeds which fell from the trees.
Normal age-class distribution:	"Occurs when the forest has sufficient area in each age class, adjusted for differences in site productivity, to allow equal annual harvests" (Leuschner, 1990, p.4).
Normalised forest:	The forest will produce the same volume of timber annually over the planning horizon.
Operational plan:	Short-term plan for managing the forest. See APO.
Planning forester:	Forester whose job it is to ensure a long-term supply of wood for mills.
Planning horizon:	Time period for which the plan is valid.

Term	Definition
Planting:	Putting a tree cutting or seedling into the ground so that it can grow.
Plantation forest:	Planted forest - sometimes called a tree farm or a plantation.
Planting density:	Number of stems (trees) per hectare which have been or are to be planted in a stand.
Process capacity:	The number of tonnes of logs a process can accept per hour (in tonnes/hour).
Process throughput:	The number of tonnes p.a. of logs that the process can accept (in tonnes).
Process type:	Type of process a mill has, typified by they kind of product it manufactures, e.g. kraft pulping, thermomechanical pulping (TMP), dissolving pulping.
Provenance:	“Place of origin” (Sykes, 1982).
Pulping:	The process of disassociating wood fibres from each other so that they can be reassembled in flat sheets of paper.
Pulp product:	The product manufactured in the pulping process.
Recovery:	The volume of output (e.g. pulp or sawn timber) divided by the input volume of logs. It is sometimes used as a measure of efficiency in the forest products industries. Expressed as a percentage.
Regime:	List of activities which are to be followed (like a recipe) for growing trees in a stand.
Regime action:	One of the activities in the regime that are to be applied to all the trees in a stand.
Rotation:	The ideal, theoretical, time between planting and harvesting a stand of trees (von Gadow and Bredenkamp, 1992, p.84). Sometimes used in the sense of “crop”.
Rotation age:	The age at which the stand of trees should theoretically be harvested.
Seedling:	Young tree (or plant) grown from seed.

Term	Definition
Short-haul transport:	Transport where short distances are involved (e.g. from the Stand to the Depot or Siding). Transport is undertaken by road.
SI:	see Site index.
Siding:	Place in or near estate where logs are stored briefly before they are transported by rail to the mill.
Silvicultural activities:	Include weeding, thinning, pruning.
Silviculture:	“The art and science of growing forest crops for all their diverse ranges of uses. It is concerned with the growing of trees just as agriculture is concerned with the growing of food crops in the fields” (Savill <i>et al.</i> , 1997, p.6).
Site:	“A spatially uniform complex of climate, geomorphology and soil conditions” (Zwolinski and Hinze, 2000).
Site-species matching:	Sometimes also called Soil-species matching. Matching the tree species to the features of the stand’s site. Poor site-species matching could lead to poor growth and disease.
Site index:	An indication of growth rate of a stand of trees. It is calculated by averaging the top 20% of stand of trees’ heights. It is indexed by age for comparison purposes (i.e. SI at age x is denoted SI_x) and given in metres.
Slope, stand ~ :	Amount by which stand slopes above the horizontal. Measured in degrees.
Species (plural species):	“Group of plants subordinate in classification to genus, and having members that can interbreed and that differ only in minor details” (Sykes, 1982); the term is used loosely in forestry circles to mean the genetic material of a tree (as in “site-species matching”).
Soil-species matching:	see Site-species matching.
Solar radiation:	Energy emanating from the sun.

Term	Definition
Stand:	Smallest management unit of land on which trees are grown homogeneously in a plantation forest. In farming parlance, a stand could be considered to be a field. A stand of trees is planted at the same time, with the same genetic material, and is managed according to the same regime. This means that all the regime actions (including felling) are applied to the stand of trees at the same time.
Stand area:	The area of the stand (in hectares).
Stand state:	Description of the stand's current condition: planting has commenced, establishment completed (this is the same as "planted"), felling has started, temporarily unplanted.
Stems per hectare:	Also called Trees per hectare. Number of trees planted per hectare. See also Planting density and Stocking density.
Stocking density:	The number of trees per hectare in a stand.
Stone groundwood pulping:	A type of mechanical pulping whereby the pulp logs are broken up by rubbing them against stones. See also Mechanical pulping.
Strategic plan:	Long-term plan for managing the forest.
Sulphate pulping:	see Kraft pulping.
Tactical plan:	Medium-term plan for managing the forest.
Temporarily unplanted:	A stand whose trees have been felled, but which has not yet been replanted.
Timber:	Generic term for wood from trees which have been cut down. Could be in tree-lengths, or logs or sawn boards (planks).
Timber allocation system:	A decision support system which ensures that logs from the forest are transported to the correct mill so that the mill has enough of the right timber at the right time.
Timber logistics manager:	Person whose job it is to ensure that the right logs are delivered to the right mill at the right time.

Term	Definition
Thermo-mechanical pulping:	Using steam while grinding wood chips against metal plates to form pulp. See also Mechanical pulp.
Transport method:	The method of transporting timber, e.g. by road or by rail.
Transport type:	The type of transport undertaken: either long-haul or short-haul.
Trees per hectare:	Number of trees planted per hectare. See also Stems per hectare and Stocking density.
User-enforced date:	A date on which a certain regime action must be undertaken. This date could be put into a regime because of lease obligations, or because a weather-event or disaster (fire, insect attack) has caused the planned regime to be changed. See Regime and Regime action.
Working circle:	The intended end use for a stand of trees, usually determined before planting.
Yield:	see Recovery.

Glossary of Z notation used

The following are definitions of the Z notation used in this dissertation. For a complete definition of Z notation, refer to Spivey (1998) or Woodcock and Davies (1996).

Notation	Meaning
\emptyset	Empty set. Same as $\{\}$.
$x \in S$	The element x is in the set S .
$x \notin S$	The element x is not in the set S .
$\mathbb{P}S$	Power set of S - i.e. the set of all the subsets of the set S (could have an infinite number of entries).
$\mathbb{F}S$	Finite power set of S - i.e. the finite set of all the subsets of the set S .
$\#S$	The number of elements in the set S .
$S \subseteq T$	Set S is a subset of, or is equal to, set T .
$S \cup T$	Set S union set T (i.e. all of set S 's elements are collected together with all of set T 's elements in the new set).
$S \cap T$	Set S intersects set T (i.e. the elements of set S which also belong to set T).
\forall	For all ...
\exists	There exists at least one...

Notation	Meaning
\exists_1	There exists a single ...
$\bullet \dots$	'such that', or, 'it is always true that' ...
$\exists sID : STANDID \bullet \dots$	There exists at least one variable <i>sID</i> of type <i>STANDID</i> such that ...
$p \Rightarrow q$	Conditional logical connective; is read ' <i>p</i> implies <i>q</i> '. "The expression $p \Rightarrow q$ is always true, except when <i>p</i> is true and <i>q</i> is false" (Dean, 1997, p.186).
$p \wedge q$	Logical conjunction of the two propositions <i>p</i> and <i>q</i> ; is read ' <i>p</i> and <i>q</i> '. "The expression $p \wedge q$ is only true when both <i>p</i> and <i>q</i> are true" (Dean, 1997, p.186).
$p \vee q$	Logical disjunction of the two propositions <i>p</i> and <i>q</i> ; is read ' <i>p</i> or <i>q</i> '. "The expression $p \vee q$ is only false when both <i>p</i> and <i>q</i> are false" (Dean, 1997, p.187).
\mathbb{Z}	Set of integers, i.e. $\{\dots, -2, -1, 0, 1, 2, \dots\}$
\mathbb{N}	Set of natural numbers, i.e. $\{0, 1, 2, 3, \dots\}$
\mathbb{N}_1	Set of non-zero natural numbers, i.e. $\{1, 2, 3, \dots\}$
$m \bmod n$	The modulus of <i>m</i> with <i>n</i> (i.e. the integer remainder obtained when <i>m</i> is divided by <i>n</i>).
$m \operatorname{div} n$	The integer number of times that <i>m</i> can be divided by <i>n</i> .
$A \times B$	The Cartesian product of two sets <i>A</i> and <i>B</i> , which is a set of couples (2-uplets). The first element of the couple belongs to the set <i>A</i> and the the second belongs to the set <i>B</i> : i.e. $a \in A$ and $b \in B$ means that $(a, b) \in A \times B$
$first(a, b)$	The first element of the tuple, i.e. <i>a</i> .
$second(a, b)$	The second element of the tuple, i.e. <i>b</i> .
dom	Domain (i.e. input) of a function.
ran	Range (i.e. output) of a function. Also called the image.

Notation	Meaning
$f : A \leftrightarrow B$	Partial function f which maps elements of type A to elements of type B . For $a \in A$, the partial function f means that sometimes $f(a)$ exists and sometimes it does not, i.e. $\exists a \in A$ such that $f(a) = \emptyset$.
$g : A \rightarrow B$	Function g which maps elements of type A to elements of type B . For $a \in A$, the function g guarantees that there is always an element $b \in B$ such that $g(a) = b$.
\oplus	Override. Used to replace a particular value of the range (relating to a particular value in the domain), without changing the other values in the range.
\mapsto	Maplet. Used to relate a value of the domain to a value in its range.
$myFunction \sim$	Inverse of the function called $myFunction$.
$PLANTINGSTATE ::= unplanted \mid planted$	Enumerated type $PLANTINGSTATE$ which has two values: $unplanted$ or $planted$.
$seq A$	A sequence of type A .
$aString \hat{\ } anotherString$	Concatenation of string $aString$ with string $anotherString$ (i.e. string "addition").
$plantingStatus sID$	The function $plantingStatus$ evaluated at sID . Could also be written $plantingStatus (sID)$.
$input?$	An input variable, denoted as input by the $?$ at the end.
$output!$	An output variable, denoted as output by the $!$ at the end.
$variable$	A variable (before an action has taken place).
$variable'$	A variable after an action has taken place – denoted as after an action because it is decorated with a dash. Functions, schemas and sets can also be decorated with a dash.

Notation	Meaning
$\Xi SchemaA$	$\Xi SchemaA$. Means that the functions and variables of $SchemaA$ are not changed by the schema into which they are included.
$\Delta SchemaB$	Delta $SchemaB$. Means that the functions and variables of $SchemaB$ could be changed by the schema into which they are included.
$SchemaA \hat{=} SchemaB \wedge SchemaC$	$SchemaA$ is made up of the conjoined (ANDed) schemas $SchemaB$ and $SchemaC$. This means that the declarations of the two schemas $SchemaB$ and $SchemaC$ are merged into the declaration part of $SchemaA$, while the predicate parts of the two schemas are conjoined (ANDed) in $SchemaA$'s predicate part.
$S.x$	Selection of field x from S .

Acknowledgements

I would like to thank:

My supervisors, Jules-Raymond Tapamo and Flic Blakeway, and Fethi Ahmed (technical committee member), for their guidance, comments and inputs during this study.

Anabel Fossey, for her excellent seminar on writing up a dissertation, and for reviewing an earlier version of the literature review.

Rosemary Quilling, for her encouragement, and advice about dissertation-writing.

Supporting family members, and friends, and particularly my parents, who tolerated the major topic of conversation being this dissertation for the past 2+ years. Thank you for your prayer and encouragement.

The UKZN Westville campus Inter-library loan staff, for helping me source so many books and articles that were not otherwise available.

Bill Esler and Hans Peters, for their comments on the semi-formal analysis, and for being available to clarify certain details.

The Mondi staff, with whom we interacted during the requirements-gathering phase of the forest harvest scheduling system project we undertook. Thank you for sharing your knowledge with us.

The CSIR, for the opportunity to learn about sawmilling, forestry and pulp and paper making over the 19+ years of my employment there.

The team at the Forestry and Forest Products Research Centre (FFP), Durban, for the opportunity to learn about wood properties in a multi-disciplinary context.

Flic Blakeway and Tammy Bush, for providing me with infrastructural support at FFP from July 2007.

Manoj Maharaj and the staff at the School of Information Systems & Technology, UKZN, for their support in my completing this dissertation.

Caroline Goodier, for her support, advice and encouragement.

God, the Father, for His encouragement and for the courage and endurance He gives
... "*Dieu, tu es fidèle.*"

Chapter 1

Introduction and purpose of the study

This study came about from a confluence of ideas, experiences and facts which occurred in the mid to late 1990s and early 2000s.

- The author was employed at a Forestry and Forest Products research organisation, and in the 1990s was involved in developing, testing and implementing computer systems which would simulate and optimise the recovery of the sawmilling industry. This experience led to an understanding that a global optimum (e.g. a solution which would satisfy all the goals of the business) is much preferred to the addition of several local optima (e.g. different parts of the business seeing to their internal needs, rather than taking into account how their actions would affect other parts of the business).
- The second aspect was work which was being done by colleagues in the wood anatomy and pulp and paper properties field in the late 1990s and early 2000s: several studies which characterised wood and pulping properties were undertaken on plantation-grown eucalypts and pines of different ages, species and site indices (growth rates). Before these studies had been undertaken, it was believed that trees of the same species and age had the same wood and pulp properties. The

studies showed, however, that older trees which had grown more slowly might be better grouped with younger trees which had been faster growing. Selecting the harvesting date of a stand of trees based solely on its age would therefore cause the trees entering the mill (or a process in the mill) to have wider variation than it would have if the wood and/or pulping properties were taken into account when making the harvesting decision.

- The third aspect was a study undertaken by Turner *et al.* (2000) for a pulp manufacturer at the turn of the century: their manufacturing process was producing pulp of high variability which they wanted to minimise. In this study, the fibre's collapsibility was chosen as the anatomical property of importance. Stands of trees which were to be felled in the next time frame were categorised into those with high collapsibility and those with low collapsibility and fed into two different processes. A field trial showed that doing this reduced the variability of the pulp in the mill.
- The fourth aspect was a follow up to this study, which estimated the average collapsibility of trees destined for the two different processes for the next six years. The harvesting decision for these stands of trees had already been made based on the trees' age. This study showed that while separating the timber into piles of high and low collapsibility would be an effective strategy for each of the six years, the high collapsibility and the low collapsibility values changed between years. The wood entering the mill would therefore have a varying collapsibility over time, which is undesirable; mills would prefer to have wood having similar properties entering each process over time.

As the result of these factors, a project was undertaken to design a forest harvest scheduling system. This system would have to take into account the mill's requirement for timber. It would have to take a long-term view to ensure that the mass of logs could be supplied to the mill sustainably over time; at the same time, it would have to ensure that the properties of the wood entering the mill remained within certain predefined limits. A brief review of the literature showed that several forest harvest scheduling

systems had been developed, but they had different time frames and different aims. Some systems concentrated on ensuring a steady supply of logs to the mill over a long period; others were more operational in nature, supporting weekly and daily decisions; yet others were in between and aided decision-making for several months or a few years. No system resolved all three time frames at once because the mathematical technique used to obtain an optimal solution would not be able to find a solution in an acceptable time. In the initial review also, no system was found which included wood properties in the harvesting decision.

A more thorough literature review was then undertaken to determine:

- if a long-term forest harvest scheduling system had been developed before which takes wood properties into account when making the harvesting decision;
- what the features of this and other forest harvest scheduling systems are; and
- what techniques were used to specify such systems.

As a result of the literature review, it was decided that an opportunity existed to develop a system which would fill the gap in the market. The forest harvest scheduling system to be developed would be aimed at the plantation forestry industry, and particularly that part of the industry which provides the raw material for pulp and paper manufacture. The decision was taken not only because it is the most simple case, but of all the industrial uses of timber, pulp and paper is the largest. It was understood that the system could be expanded later to accommodate sawtimber production; it could also be expanded to be applicable to natural forests.

A project was started in which the requirements for the system were gathered. An integrated plantation forestry and pulp and paper company helped to provide the business rules and context. The output of this was a report (Easton *et al.*, 2003) in which the requirements were described. The project was later terminated due to insufficient

funds. In this project, it was found that the business rules relating to the forestry aspects of the system were complex, and a technique which would enable the description of the system and its environment to be captured in a precise and unambiguous way was needed. Natural language can be very ambiguous; semi-formal techniques (such as entity-relationship diagrams, business process diagrams and state charts) are less ambiguous, but they cannot be checked for consistency mathematically. Formal methods have a mathematical basis, which allows them to be checked automatically for consistency and correctness. A disadvantage of formal methods is that they are not readily understood by clients and some developers, so a combination of semi-formal methods and formal methods, combined with descriptive text, can be used to describe and analyse the system and its environment.

The aim of this study is therefore to use semi-formal and formal specification techniques to specify a strategic plantation forest harvest scheduling decision support system which includes wood properties in the harvesting decision. This can be broken down into sub-goals:

1. Specifying the plantation forestry domain using semi-formal and formal methods. This aims to capture the “agricultural” (silvicultural) aspects of plantation forestry – how trees are grown as a crop.
2. Specifying the forest-to-mill domain using semi-formal and formal methods. This aims to capture the business aspects of plantation forestry, i.e. the logistical, financial, management and environmental aspects.
3. Giving a computer science/information systems specification of the strategic (long-term) plantation forestry harvesting scheduling plan to ensure that there is always a sufficient mass of wood for the mill. This includes appropriate aspects of forest management, including logistical, financial and environmental constraints. It is not an Operations Research description (i.e. mathematical formulation) of the problem.

4. Adding wood properties to the strategic plantation forest harvesting plan to ensure that the mill receives wood of a consistent quality.
5. Incorporating logistical constraints (which are usually included in the tactical plan) to refine the strategic plantation forest harvesting plan and make it more realistic.

The semi-formal and formal analyses of the forestry and forest-to-mill domains and the specification of the forest harvest scheduling system were undertaken based on the interviews held with the integrated plantation forestry company. Semi-formal analyses were undertaken first. A formal description of the domain and the system were then developed, first as a very abstract version, with more detail being added with subsequent refinement. The semi-formal specifications were reviewed by one of the main interviewees, as well as other forestry experts. The narrative text of the formal specifications were reviewed by a forestry expert. The formal specification was reviewed by two experts in the formal notation used: one who had a knowledge of forestry and the other who had no prior forestry experience.

The literature review of forest harvest scheduling systems and their specification (as outlined above) is given next (Chapter 2). As part of the work for this chapter, systems and models in the literature were reviewed to determine if a forest harvest scheduling system which includes wood properties in the harvesting decision had ever been developed. The features and use of the systems and models, and the mathematical techniques and formulations used to solve them, were also recorded in tabular form. These tables can be found in Appendices A and B. Appendix A gives summaries of the forest harvest scheduling systems and models found in the literature, their use and features. Appendix B gives a summary of the mathematical formulation of these systems and models and the solution technique used.

Chapter 3 gives an outline of the methods and techniques used in describing the forest harvest scheduling system and its environment (i.e. the domain) using a semi-formal and formal approach. The semi-formal approach to specifying the domains and the

system is presented in Chapter 4, while the formal approach is presented in Chapter 5. Chapter 4 starts with a semi-formal description of the forest-to-mill supply chain, plantation forestry, transport, mill and forest planning domains, and finally describes the forest harvest scheduling system. Aspects of the domains described in Chapter 4 which would be impacted by implementing the forest harvest scheduling system are described in more detail in Appendix C. Chapter 5 starts with the presentation of an abstract version of the forest-to-mill domain using the formal notation Z , and then goes on to show how it was expanded to include more detail in preparation for the forest harvest scheduling system specification. (The formal specification of the forestry, transport and processing domains can be found in Appendix D.) Chapter 5 also introduces highlights of the forest harvest scheduling system specification. (The formal forest harvest scheduling system specification can be found in Appendix E.) Appendices D and E each conclude with the output of the type checker which was used to ensure that there were no syntactical errors in the formal specification. An index of all the schema names specified in these two appendices can be found on page 431.

The system described in Chapters 4 and 5 has been patented (Turner and Price, 2005). The dissertation ends with discussion and conclusions in Chapter 6, together with an outline of future work.

As this is a multi-disciplinary study, a glossary of forestry and wood and pulp property terms used in this dissertation can be found in the frontice pages (page xxiv). Also in the frontice pages is a list of Z notation symbols used in this dissertation (page xxxiv). The references (given on page 407) have been categorised into the five main areas of research described in this dissertation. This concept matrix can be found in Appendix F.

Chapter 2

Literature review

2.1 Literature included

Five major topics are covered in this review: forestry (i.e. the agricultural aspects of growing a crop of trees); forest management (which includes planning, and incorporates logistical, environmental and financial constraints); wood and pulping properties; Operations Research (which involves algorithms and formulae to optimise a system in a quantitative manner); and the specification of information systems.

The forestry and forestry management literature covered in this review concentrates on that which applies to plantation forests. However, literature concentrating on natural forestry was considered where it could be applied to a plantation forestry environment. For example, in natural forests, there are constraints to harvesting (clearfelling) a stand, and then the stand directly next to it, as there would not be habitat and food for the animals which live in the forest. In this case, the first stand felled would have to be replanted and the trees be a certain age or have reached a certain height before the adjacent stand could be harvested (the so-called “green-up constraints”). In plantation forestry, by comparison, one would probably want to harvest stands in close proximity to each other, before moving the harvesting equipment and teams to another location. In cases such as these, the literature on adjacency and green-up constraints

were ignored. However, when there was literature about upgrading roads prior to harvesting, this was included, as this is a constraint for both natural and plantation forests. Literature about forestry and forest management which is aimed at producing pulp timber was emphasised in the study.

The literature about wood and pulp properties which was included in this study was that which was relevant to pulp and paper production. The Operations Research literature considered concentrated on optimisation techniques and heuristics which could provide a 'best' solution for the forest-to-mill chain. The specification literature included in this study concentrated on semi-formal (i.e. diagrammatic) and formal (i.e. mathematics-based) methods of specification.

The literature review presented in sections 2.2 to 2.5 is based on Price *et al.* (2008a).

2.2 Introduction

It is estimated that by 2050, more than half the world's requirements for industrial timber could be supplied from plantations (Kirilenko and Sedjo, 2007). While natural forests have been used for centuries for their timber, in the last few decades, with increasing pressure to conserve forests and forest species habitats, timber is increasingly being sourced from managed or plantation forests (Sedjo, 1999; Dyck, 2003). Intensively managed plantations are able to produce increasing volumes of timber due to gains from tree breeding and continually improving management, thus enabling the natural forests to be retained for other purposes such as maintaining biodiversity and recreation (Dyck, 2003). Table 2.1 shows the area worldwide which was under plantation forestry in 2000 (from Carle *et al.*, 2002). 'Industrial purpose' means trees which were planted to provide a resource for commercial processing, such as building, making panel products, sawn timber and pulp and paper. 'Non-industrial' timber plantations were planted for fodder, firewood, windbreaks, to prevent desertification,

rehabilitate land and promote conservation. Sometimes the planting purpose was unknown or unspecified at the time of planting; this falls into the 'Unspecified' category. The area under plantation forestry had increased from 17.8 million hectares worldwide in 1980 to 187 million hectares in 2000 (Carle *et al.*, 2002). By 2005, this area had increased to 272 million hectares (Fins, 2008).

Table 2.1: Area of plantations by region and planting purpose in 2000 (from Carle *et al.* (2002))

Region	Plantation area (in 1 000 Ha) by purpose			
	Total	Industrial	Non-industrial	Unspecified
Africa	8 036	3 392	3 273	1 371
Asia	115 847	58 803	43 662	13 381
Oceania	3 201	189	24	2 987
Europe	32 015	569	15	31 431
North & Central America	17 533	16 775	471	287
South America	10 455	9 446	1 004	6
WORLD TOTAL	187 087	89 175	48 449	49 463
Percentage	100%	47.7%	25.9%	26.4%

Foresters need to ensure that enough wood is available for mills to use in the long term, and that the appropriate volume of wood is delivered to mills in the short term (Leuschner, 1990; Gordon *et al.*, 2006). Because the industries supplied by the forests are often very capital-intensive (Gordon *et al.*, 2006; Philpott and Everett, 2001; Jones and Ohlmann, 2008), they need assurance of a constant supply of wood to guarantee a return on the capital invested (Evans and Turnbull, 2004). In plantations, this wood comes from stands which contain trees which typically have the same species and same age. The stand is the smallest homogeneous area of trees (Barros and Weintraub, 1982; Louw, 2000b). The same activities will be applied to all the trees in the stand at the same time, thereby creating a crop which is silviculturally uniform (Evans and Turnbull, 2004).

Managing a large number of forest stands over vast tracts of land, and deciding when to harvest each stand, is a complex task. The simplest case would be if a single species of tree were grown, and each stand had a similar soil type and climate. If the trees were

harvestable after n years, the harvesting decision would then be to cut $1/n$ th of the plantation area each year to keep the mill(s) supplied (Leuschner, 1990). Unfortunately, the situation is more complex than this. Often, the afforestable land covers a range of altitudes, soil types, and is subject to different weather patterns (Evans and Turnbull, 2004). Trees are chosen to suit the sites in which they are grown (Evans and Turnbull, 2004) (for example, over 20 different species are used commercially in South Africa, excluding hybrids and clones (Herbert, 2000)) and this variety means that the trees grow at different rates and become mature (harvestable) at different ages (Smith and Cunningham, 2002). Decisions need to be taken about which stands of trees to harvest and when, how logs should be cut from these trees, to which mill these logs should be sent and what mode of transport to use (if more than one method of transport is available), over a variety of time frames. There are many other constraints involved in forest management (Weintraub and Davis, 1996), including environmental, financial and logistical concerns. Looking at this problem as a supply chain, from source (forest) to sink (mill), would ensure that optimal results for the whole chain are achieved and that customers (mills and their clients) are satisfied (Pulkki, 2001; Stadtler, 2005). Forest harvest scheduling models and systems have been used to help with various aspects of this decision-making process since the 1960s (Weintraub and Bare, 1996).

2.3 Forest-to-mill supply chain

While many authors look only at forestry when developing a forest harvest scheduling system, the whole chain from forest to mill is important. In this chain, the forest is the source. Transport and the mill (the sink) are also an integral part. Considering the system as a supply chain ensures that the correct resource (in the form of logs) can be delivered to the mill at the overall lowest cost (Gordon *et al.*, 2006; Pulkki, 2001; Karlsson *et al.*, 2003). This is particularly important, given that log costs are around one third of a pulp mill's input costs (Burger and Jamnick, 1995). Systems which include timber allocation also enable decision makers to assess the effects of sourcing

logs from different plantations (Andalaft *et al.*, 2003).

Table 2.2: Levels of planning in the forest supply chain (adapted from Rönqvist (2003))

	Forest management and harvesting	Transportation and routing	Processing operation
Strategic planning (> 5 years)	Planting, evaluation, long term harvesting	Road building, road upgrading, fleet management	Investment planning
Tactical planning (6 months to 5 years)	Annual harvest plans	Road upgrading, equipment utilization	Annual production planning
Operational planning (1 day to 6 months)	Crew scheduling, harvest sequencing	Catchment areas, back-haulage planning, scheduling	Lot sizing ^a , scheduling
Online planning (< 1 day)	Bucking ^b	Truck dispatching	Process control, paper roll cutting, cross-cutting

^a How much to process with a particular set of process parameters.

^b Cutting trees into logs.

The supply chain uses a temporal planning hierarchy (Rönqvist, 2003), as can be seen in Table 2.2. In this table, the kinds of activities that are likely to be undertaken by forest managers, transport or logistics managers and production manager at each mill are shown for long-term (strategic), medium-term (tactical) and short-term (operational) planning. Another category is also given (online planning) which entails plans made for a planning horizon of less than one day.

2.3.1 Plantation forestry and planning

Plantation forests typically contain many stands of an even-aged monoculture of trees (Shepherd, 1986, p.133) where species or hybrids of a small number of genera are planted (Kanowski, 1997). A stand is the smallest homogeneous area of trees (Barros and Weintraub, 1982; Louw, 2000b). During the trees' lifetime, the same activities will be applied to all the trees in the stand, thus creating a crop of uniform age, species and silviculture (Evans, 1997).

Plantation forestry companies often use a hierarchy to manage their forest land (Louw, 2000b; Weintraub and Cholaky, 1991), with stands being at the lowest level. Depending on the plantation forest company, there are differing numbers of groupings in the hierarchy. The largest grouping of afforested land is often called a region (Louw, 2000b).

Plantation forests are usually established to produce a sustained yield of wood (Kanowski, 1997; von Gadow and Bredenkamp, 1992, pp.84–122; Leuschner, 1990, pp.1–8). This means that in the long-term, planning activities aim to promote a “normalised” forest (i.e. the amount of wood which is being harvested annually in the forest is growing to replace it). This long-term, or strategic, planning is undertaken over a long time horizon – usually covering two rotations of tree growth (Barros and Weintraub, 1982; Martell *et al.*, 1998; Epstein *et al.*, 1999a; Weintraub and Murray, 2006).

Once an acceptable strategic plan has been developed, the plan for the medium-term is studied in more detail by regional foresters (Weintraub and Cholaky, 1991; Ribeiro *et al.*, 2005). This tactical plan can have a duration of up to half a rotation (Nelson *et al.*, 1991; McNaughton *et al.*, 2000). In South Africa, however, it is typically three to five years (Brink and Kellogg, 2000). The part of the strategic plan which falls into this tactical time frame is examined for each region in terms of the planned harvesting activities to balance the terrain with the harvesting equipment (Brink and Kellogg, 2000) and harvesting crews. The overall rate of harvesting is then assessed annually per region to ensure that an even flow of timber is available for delivery to the mills. If the harvesting of the stands proposed by the strategic plan is not feasible, changes are proposed, and the strategic plan is re-run (Ribeiro *et al.*, 2005). Once the harvesting of stands has been scheduled, other activities can be planned. These include upgrading the roads leading to the stands, measuring (enumerating) the stands one or two years before harvesting to get a better estimate of the resultant log volume, and ordering

the seedlings or cuttings from the nursery which will be replanted in each stand. In the latter case, care is taken to match the appropriate species or genotype to the stand's site.

The short-term, or operational, plan is drawn from the first one or two years of the tactical plan (Beaudoin *et al.*, 2007). In the operational planning phase, the required work is planned and costed (Louw, 2000a). Decisions at the operational level include harvesting techniques, road alignments, scheduling of specific stands to be harvested, and assigning harvesting teams and equipment to these stands (Church *et al.*, 1994). The bucking pattern for cutting trees into logs is also assigned. Activities are allocated to the months of the year. Work that is seasonally or weather-dependent (e.g. planting, making fire breaks, some felling, and pruning) is allocated to the appropriate months (Louw, 2000a). Operational planning could be regarded as a more detailed version of tactical planning. Should it not be possible to match the harvesting terrain with the equipment and teams available, the planners may exchange stands due to be felled in the operational planning horizon with those due to be felled later (in the tactical planning horizon) (G. Fick, 2002, pers. comm.¹).

2.3.2 Transport

Trees are transported to the mills as logs (Andalaft *et al.*, 2003; Carlsson and Rönnqvist, 2005) or as tree-lengths (W. Esler, 2007, pers. comm.²). Timber transport is one of the main contributors to the delivered cost of logs (Andalaft *et al.*, 2003; McGuigan and Scott, 1995). There may be limitations on the choice of which mode of transport is used: a forestry estate or a mill may not have a railway siding, or there may be limited capacity to unload trucks or railway cars at a particular mill (D. Alborough, 2002, pers. comm.³). Logs may have to be sourced from one or more plantation in

¹Mr G. Fick, Systems Analyst, Mondi Forests, P.O. Box 37, 2000 Johannesburg, South Africa

²Mr W. Esler, Planning Forester, Mondi Forests, P. O. Box 39, 3200 Pietermaritzburg, South Africa

³Mr D. Alborough, Logistics Manager, Mondi Forests, P. O. Box 39, 3200 Pietermaritzburg, South Africa

order to make up the volume and/or quality required by the mill.

2.3.3 Mills

Mills contain at least one process. Mills (and individual processes) have requirements for logs: sometimes they do not accept certain genera or species. Apart from their volume requirements, they have log dimension requirements (Andalافت *et al.*, 2003; Williams, 1994). They may also have wood property requirements so that products of consistent quality can be manufactured (Epstein *et al.*, 1999a).

2.4 Wood properties

2.4.1 Wood properties

Wood properties can be characterised as chemical and physical properties. The main chemical constituents of wood are cellulose, hemicellulose, lignin and other materials (including extractives) (Wilson and White, 1986, p.144; Clarke, 1995, p.2). Cellulose molecules make up the longitudinal structural strength of wood, while the lignin adds rigidity to the wood's cell wall (Raven *et al.*, 1992, pp.33–37). The hemicellulose provides an interface between the cellulose and lignin molecules (Weigel, 2005, p.2). Wood's physical properties can be categorised as anatomical and densitometric. Anatomical properties are measurements of the fibres (for example, the thickness of the cell wall, and the length of the fibre). Wood's density is a measurement of the amount of wood per unit volume (Naidu, 2003, p.14).

2.4.2 Wood property variation and its effect on end-product variation

Wood is a variable resource. This means that there is a range in the chemical and physical properties of wood of a particular species, and also of wood within a particular tree (Wilson and White, 1986, p.198). Wood properties of trees of a particular species vary with harvesting age and growing environment (Malan, 2003). Variation in wood properties affects the products produced (McGuigan, 1984) and leads to inefficient processing (Zobel *et al.*, 1983).

Although trees from plantations have less variability than those from natural forests (Baillères *et al.*, 1997), trees' wood properties differ from pith to bark, from base to tip and amongst trees of the same genetic material, planted at the same time, and having received the same silvicultural treatment (Spångberg, 1999). Wood properties also vary between species (Zobel *et al.*, 1983). In addition to these factors, the sites and geographical areas (especially altitude and latitude where the trees are planted) also play a role in wood variation (Malan, 2003). The greatest variation is between the pith and the bark, in a radial direction (Larson, 1967), although the wood properties within a species or provenance can also vary greatly (depending on where the trees are grown and/or where the seed came from) (Zobel and van Buijtenen, 1989, p.298).

Work done at the CSIR and at the Forestry and Forest Products Research Centre (*ffp*) confirms that South African plantation-grown trees have different properties, depending on where they are grown, and how fast they grow (Dyer *et al.*, 1997; Retief *et al.*, 1997; Crafford *et al.*, 1998; Turner *et al.*, 1999, 2000). Their genetic makeup (genus, species) also plays a role (Turner and Retief, 1998; Grzeskowiak and Turner, 2000). In addition, Zbonak (2005) showed that altitude made a difference in being able to predict the wood properties of *Pinus patula* in plantations of KwaZulu-Natal, South Africa. This means that trees from one compartment may be well-suited to a particular process and poorly-matched with another (Sefara *et al.*, 2001).

Processing a variable wood resource produces a variable product (McGuigan, 1984; Spångberg, 1999), and mill processes are negatively affected by wood whose properties are not homogeneous (Turner *et al.*, 2000). The variation in wood properties leads to inefficient processing (Zobel *et al.*, 1983) which in turn reduces the profitability of the processor (Gordon *et al.*, 2006). For example, when pulping a variable wood resource, some wood could be underpulped and some wood overpulped, giving poor yields and quality (Zobel and van Buijtenen, 1989, p.72).

Several studies have been undertaken to quantify the effect of wood variability on the pulp process. When pulping different aged *Eucalyptus grandis* which were grown on a variety site qualities using the kraft method, the cooking time to Kappa 20 was found to vary between 21 and 105 minutes. This is a variation of 500% (Sefara *et al.*, 2001). In an American study on the pulping and fibre characteristics of hybrid poplar trees, the pulp yield was found to vary between 49 and 57% (Goyal *et al.*, 1999). In a Canadian study on the pulping efficiencies of Trembling Aspen (*Populus tremuloides* Michx), there was a six percentage point difference in the pulp yield (Mansfield and Weineisen, 2007). Another study found that for comparable conditions, the pulp yield of several different species varied between 35 to 56% (Higgins, 1970). In a dissolving pulp study, the properties of some *Eucalyptus* clones were analysed (Clarke, 2001). Although the variability of pulping properties was narrower per clone, between clones there was a wide variation in properties. For two clones in particular, their K number and viscosity were so different that different cooking conditions were recommended. If trees from two different clones, which had very different pulping properties, were pulped at the same time, there would be a wide range of variability, making the processing more difficult and expensive. It is therefore important to know the wood and pulping properties of trees grown in the plantations so that a strategy of processing can be followed (campaigns or blending). It would therefore be better for both the pulping process and the resultant pulp to classify logs according to their wood properties

(Spångberg, 1999; Turner *et al.*, 2000). In a Brazilian study, it was found that the type of *Eucalyptus* wood (13 samples of *E. grandis* x *E. urophylla* clones) had a significant effect on the brightness stability of fully bleached pulp irrespective of the pulp and bleaching processes used (Colodette *et al.*, 2004).

The aim for a processor is recognise the inherent variability in wood and use the “right wood in the right process” (McGuigan, 1984) and to use wood whose properties are uniform and predictable to ensure the manufacture of an end product with known properties (Zobel and van Buijtenen, 1989, p.303). This echoes the customer-oriented philosophy of supply chain management, whose aim is to ensure that the right type of raw material (wood) needs to be delivered to the customer (the mill) at the right time, in the right quantities (Andalaft *et al.*, 2003).

2.4.3 Important wood properties for processes

Different pulp and paper products require different properties: for example, fine paper needs good surface strength, dimensional stability, bending stiffness and smoothness, while tissue needs softness, smoothness, water absorption ability, as well as wet and dry strength (Karlsson, 2006, pp.85–87). Each of these properties is affected by different biological and chemical wood properties. According to Gartner (2005), for pulp manufacture, the most important properties are strength, smoothness, softness and yield (see Table 2.3). The properties which are important depend to some extent on the process used. Other authors' opinions are summarised in the following text.

Many authors cite higher wood density as improving pulp quality and strength (Spångberg, 1999; Muneri, 1994, p.3; McDonough, 1998; Miranda and Pereira, 2002; Lundqvist, 2003; Turner *et al.*, 2005; FAO, 2006). This would correlate with findings that pulps made from juvenile wood (which has a lower density than mature wood) have a decreased yield of up to three percentage points (Jackson and Megraw, 1986;

Table 2.3: Effects of biological and chemical wood properties on pulp and paper properties (from Gartner (2005))

	Product property	Required biological/chemical property
Pulp and paper properties	Paper strength	Fibre length and coarseness
	Pulp smoothness	Fibre length
	Pulp softness	Fibre length and coarseness
	Pulp yield	Extractive content, density and knots

Kennedy, 1995).

Pulp yield is also an important factor for pulp mills (Miranda and Pereira, 2002; Turner *et al.*, 2005). In 2005, Turner *et al.* showed that increasing the recovery by one percent translates to approximately R23 million p.a. for a mill whose intake is 1.6 million tonnes p.a.. Wood density and pulp yield together affect the production of pulp per unit area of grown trees (Miranda and Pereira, 2002). The cellulose content of wood is also highly positively correlated with pulp yield (du Plooy, 1980; Collins *et al.*, 1990; Wallis *et al.*, 1996; Kube and Raymond, 2002).

Another important factor is the lower lignin content of the wood (Jackson and Megraw, 1986) and the time it takes for a chemical process to remove the lignin to acceptable levels. Decreasing the cooking time by 1% in a kraft process means a similar increase in income for a mill whose intake is 1.6 million tonnes per year (Turner *et al.*, 2005). Lignin content affects process efficiency (Miranda and Pereira, 2002), and pulping woods with higher lignin contents in a chemical process will need more chemicals than those with lower lignin contents (FAO, 2006; Muneri, 1994, p.64). Extractives and lignin content affect process efficiency (Miranda and Pereira, 2002).

Fibre characteristics have also been found to affect pulp and paper quality (Spångberg, 1999). Long fibres generally give good paper strength properties (Karlsson, 2006, p.22) including tear strength (Clarke, 1995, p.8; Wimmer *et al.*, 2002). Fibre length is also positively related to pulp yield (Wimmer *et al.*, 2002). Thin walled fibres improve paper strength (Karlsson, 2006, p.22). A minimum length of fibre is needed so that the fibres

will bond with each other (Ivkovich, 2000, p.17). Cell wall thickness correlates well with pulp strength properties (Xu *et al.*, 1997). Thin walled fibres are more likely to collapse with refining, giving a greater surface area to bond with other fibres (Paavilainen, 1994).

In spite of the way in which wood properties within and between trees vary, wood can be matched to different processes according to the wood, pulp and fibre properties (Naidu, 2003, p.154). Softwoods with a high density make kraft pulps with increased pulp yields with increased tear strength, resistance to beating and increased bulk, while softwoods with a lower density are more suited to writing and tissue paper, as they have a low tear strength and a low yield, but a high tensile strength and good folding properties (FAO, 2006). High density hardwood is well suited to kraft pulp and the pulp quality is further improved by mixing it with a pulp made from a high density softwood. Low density hardwood is suited to making tissue, writing and printing paper. Harvesting eucalypts when they are younger (before the age of six to eight) would reduce the content of polyphenolic extractives which causes problems in the chemical pulping process (FAO, 2006). This approach of matching the wood properties to the end-product is confirmed by Kennedy (1995), who proposes that if paper with a high sheet density, good burst and tensile strength is required, one should use juvenile wood. If paper with a higher tear strength is required, mature softwood should be used.

For the thermo-mechanical and stone groundwood pulping processes, when aiming to increase the fibre collapsibility (a function of cell diameter and cell wall thickness), the bulk density, tensile and tear for stone groundwood pulp increased while the tensile and burst increased significantly and the reject rate decreased for the thermo-mechanical pulp process (Turner *et al.*, 2000). A Finnish study found that when making good grades of thermo-mechanical pulp, the most important factor was the freshness of the logs (Mikkonen, 1996). A recent European study summarised important properties for the thermo-mechanical pulp mills and their end-users: tensile and tear strength, stiffness, stretch, porosity and drainage, paper formation, the smoothness and strength

of the surface, the pulp's opacity, brightness and bulk (Lundqvist, 2003).

2.4.4 Log sorting schemes aimed at reducing end-product variation

Over the past decade, a number of scientists have recognised the need to reduce the variation of wood entering the mill, and have investigated log sorting schemes as a remedy. Table 2.4 gives an overview of sorting studies which have been reported in the literature. The outcome of Megown *et al.*'s report (2001) needs to be highlighted: they analysed and categorised the wood annually earmarked by a harvest scheduling system to be felled for a six year period into piles of high and low collapsibility. While applying this sorting would reduce the variation in the mill for each year, the collapsibility values for the high and low collapsibility piles respectively varied between the years. It was concluded that the harvesting decision needed to be made after the assessment of the trees' wood properties (not before), and that the trees' wood properties must be assessed over a long period to ensure that the properties entering a mill's process remains stable over time.

Only two of the sorting studies reviewed described a decision support system which implemented a sorting scheme. The first system (Fibre Prediction System) (Megown *et al.*, 2000) implements the sorting scheme described by Turner *et al.* (2000). It predicts the wood properties (e.g. collapsibility) of stands of trees once the harvesting decision has been made and allocates wood with lower collapsibility to one process and that having higher collapsibility to another process. The other model (Weigel, 2005, pp.22–33) describes a pulp mill supply chain and allows for log sorting to take place, based on species, age, growth site and density or other wood properties. This model also does not include forestry or harvesting decisions.

Table 2.4: Summary of sorting studies aimed at reducing wood property variation

Reference	Country	Species	Sorting criterion	Comments
Williams (1994)	New Zealand	<i>Pinus radiata</i> (and other)	Species and density ^a	Sorting logs helped to match the wood's fibre characteristics to the pulp and paper products to be manufactured in the mill.
Duchesne <i>et al.</i> (1997)	Sweden	<i>Picea abies</i> and <i>Pinus sylvestris</i>	Species, tree size ^b , log type ^c	Sorting logs in the forest based on wood and fibre properties increased the ability to control the pulping process.
Spångberg (1999)	Sweden	<i>Picea abies</i>	Mean annual growth ring width or harvest class type ^d	Sorting by mean annual growth ring width gave the best way of classifying logs.
Turner <i>et al.</i> (2000)	South Africa	<i>Pinus patula</i>	Average fibre collapsibility ^e	Variation of the pulp made by thermo-mechanical and stone groundwood processes was significantly reduced by sorting the logs.
Megown <i>et al.</i> (2001)	South Africa	<i>Pinus patula</i>	Average fibre collapsibility	Harvesting decisions need to be made after assessing wood properties to supply wood of consistent properties to mill.
EUAgrinet (Lundqvist <i>et al.</i> , 2003; Lundqvist, 2003; EU-AgriNet, 2000)	Europe	<i>Picea abies</i> L. Karst.	Wood, fibre and pulping properties	Prediction of wood properties in-field helped to sort logs for more uniform thermo-mechanical pulp.
Weigel (2005, pp.22–33)	Canada	Several species ^f	Species, age, growth site, density	Comprehensive pulp mill value chain model from roadside to mill product; maximises chain's profit; decides which sorting options should be implemented to make certain pulp products.

^a *P. radiata* was categorised into low, medium and high density classes; other species were grouped into one class.

^b Dominant and suppressed trees.

^c Butt and middle logs in one class and top logs in the other class; all logs of the suppressed trees, of both species, were grouped together.

^d First thinning, second thinning and clearfelling.

^e A function of cell diameter and cell wall thickness.

^f Douglas fir, Western hemlock, Western red cedar, Amabilis fir, Logepole pine, Jack pine, Black spruce, Hybrid poplar, White birch are amongst the common Canadian species.

2.5 Forest harvest scheduling and timber allocation systems

2.5.1 Background

The problem that forest harvest scheduling systems are trying to solve is a complex one (Weintraub and Davis, 1996; Martell *et al.*, 1998). At a strategic level, the aim of such systems is to ensure that a sustainable yield comes from the forest, over a long time.

In some instances, this period could be as long as 150 years (Nelson *et al.*, 1991); it is typically the length of two rotations (Barros and Weintraub, 1982; Martell *et al.*, 1998; Epstein *et al.*, 1999a; Weintraub and Murray, 2006). In the medium term, foresters need to know which stands to harvest (Murray and Church, 1995). They also need to plan for roads to be built or upgraded prior to the commencement of harvesting (Weintraub *et al.*, 1995; Guignard *et al.*, 1998). In the short term, forest harvesting crews and equipment need to be allocated to specific stands and decisions need to be made about how to cut trees into logs to meet specific mills' demands (Epstein *et al.*, 1999b; Mitchell, 2004, pp.1–2). These systems often fall into the “cut-plant-grow” category (Davis and Martell, 1993) and do not consider how to allocate the logs to the appropriate mill (McGuigan, 1984), which is also important, especially because transport costs play a big role in the cost of the delivered log (Barros and Weintraub, 1982; McGuigan and Scott, 1995). Systems which include timber allocation enable one to assess the effects of sourcing logs from different plantations, especially if one plantation is significantly closer to the mill than others (Andalaf *et al.*, 2003). The problem therefore has a wide range of issues that it needs to address, including time scale and level of detail of decisions to be made (Weintraub and Davis, 1996; Cea and Jofré, 2000).

Since the late 1950s, models and systems have been developed to address aspects of the problem (Martell *et al.*, 1998). Optimisation techniques such as linear programming (LP), integer programming (IP) and mixed integer programming (MIP) have been favoured, as they are able to find the optimal solution for the problem (Martell *et al.*, 1998). Optimisation approaches are favoured because they offer precise solutions while addressing the problem's objectives (Mathey *et al.*, 2008). In California, it is law to use a LP approach to produce long-term sustained yield forest plans (Martell *et al.*, 1998). Other problem solving paradigms, like heuristics, have also been applied to these problems (Nelson *et al.*, 1991; Weintraub *et al.*, 1994, 2000), and although they give feasible solutions, they cannot guarantee an optimal solution.

LP, IP and MIP techniques have limitations in terms of the problem size they can effectively solve (Martell *et al.*, 1998; Cea and Jofré, 2000; Weintraub, 2007). The LP technique can solve large problems, giving real numbers in the solution, whereas the IP technique will ensure that all the variables in the solution are integers, but the solvable problem size is smaller (Lundqvist *et al.*, 2003; Weintraub and Bare, 1996). For forest harvest scheduling systems, where decisions need to be made about which stand to harvest, and when, binary (also called “0-1”) programming (a specific case of IP) needs to be used (Weintraub and Cholak, 1991) (where 0 indicates that one should not harvest a particular stand, and 1 indicates that one should). In MIP formulations it is the number of binary variables that causes such problems to be difficult to solve (Murray and Church, 1995; Cea and Jofré, 2000), and as a result, large MIP problems are difficult to solve (Nelson *et al.*, 1991; Weintraub and Cholak, 1991; Yoshimoto *et al.*, 1994; McNaughton *et al.*, 2000; Turner *et al.*, 2002).

It is impossible to solve the harvest scheduling problem with all its detail in one attempt, as this problem is too large (Kent *et al.*, 1991; Beaudoin *et al.*, 2008). The most common approach to resolve this problem is to view the large problem as a hierarchy of three smaller problems, each one addressing a different time frame: the strategic problem addresses the long term, the tactical problem addresses the medium term and the operational addresses the short term. These problems also tend to apply to different regions of the forest (large areas are considered for strategic problems, diminishing to smaller areas for operational problems). The aim and focus of the problems at these different levels is also different.

Many models and systems have been developed over the last forty years to help forest-based companies manage their operations. Each has its own particular aims, focus and intended planning horizon. The following sections describe models and systems that either apply to plantation forestry, or could do so, categorised by the planning horizon. A tabular summary of systems and models reported on in the

literature can be found in Appendices A and B. It should be noted that different authors are not consistent with their use of terminology when describing the planning horizon (e.g. “tactical” could mean 30 years to one and six months to another).

2.5.2 Strategic planning

Strategic planning models and systems are usually formulated as LP problems (Martell *et al.*, 1998), where the aim is to maximize net present worth of the harvested timber subject to linear constraints, such as harvest flow, resource availability and forest management. The decision variables indicate what harvesting, planting etc. needs to take place for each time period in the forest. In order to simplify the formulation, individual stands are grouped according to similar features (Weintraub and Cholaky, 1991; Church *et al.*, 1994), such as land, climate, site type, species planted, silvicultural regime and growth rates (Garcia, 1984; Gunn, 1991; Laroze and Greber, 1991; Morales *et al.*, 1994; McGuigan and Scott, 1995; USDA Forest Service, 1996; Martell *et al.*, 1998; Cea and Jofré, 2000; McNaughton *et al.*, 2000; Andersson and Eriksson, 2007). Institutional or ownership considerations could also play a part in how these groups are formed (Garcia, 1984; Gunn, 1991; Weintraub and Cholaky, 1991). These groups should respond in the same or similar way to a specific set of management practices (e.g. silviculture and harvesting) (Garcia, 1984; Kent *et al.*, 1991). Statistical cluster analysis techniques can be used to assist with the classification of these groupings (McGuigan and Scott, 1995). The LP therefore gives the area of a particular forest group type that should be harvested, or planted in each time period.

In terms of time scale, a strategic plan would typically cover two rotations (Barros and Weintraub, 1982; Martell *et al.*, 1998; Epstein *et al.*, 1999a; Weintraub and Murray, 2006), although some authors suggest that covering one rotation is sufficient (McNaughton *et al.*, 2000; Andersson and Eriksson, 2007). The LP problem description may also aggregate periods in the long time horizon so as to decrease the problem size.

This is sometimes done by allocating a decade, or a number of years to a single time period (Gunn and Rai, 1987; Barber, 1983; Manley and Threadgill, 1991; Nelson *et al.*, 1991; Eid and Hobbelstad, 2000). In other cases, models make allowances for a time period to be variable: it can represent a single year, two years, or any other number of years (Barros and Weintraub, 1982; Morales and Weintraub, 1991), providing that the accompanying data is also aggregated according to these time frames (McGuigan and Scott, 1995).

The early forest harvest scheduling systems of the 1970s and early 1980s (e.g. Timber RAM (Navon, 1971; Chappelle *et al.*, 1976; Tedder *et al.*, 1978), ECHO (Walker, 1974; Tedder *et al.*, 1978) and HARVEST (Barber, 1983)) concentrated on ensuring a sustained timber supply. FORPLAN (Kent *et al.*, 1991; Johnson *et al.*, 1980, 1986; Garcia, 1988) was developed in the 1980s to respond to the demands of the USA National Forest Management Act (1976) which not only required that forest management plans would ensure sustained yield and multiple use of the forests, but also maximise the public benefits of the forests in “an environmentally sound manner” (Federal register, 1982). FORPLAN includes decisions on road making (Bare, 1996), specifies when activities such as harvesting should occur (Field, 1984; Church *et al.*, 2000) and schedules treatments and the resultant product flows (Field, 1984). FORPLAN’s detractors include the difficulties in addressing spatial aspects of the problem, and the fact that the formulation was trying to include too many issues, making the problem too large and therefore difficult to solve (Weintraub and Cholaky, 1991). Another strategic planning system, FOLPI (Garcia, 1984, 1990; Manley and Threadgill, 1991; Papps and Manley, 1992; Manley *et al.*, 1996), which was developed in the 1980s supports decisions about when to harvest forests and when apply silvicultural regimes while ensuring required levels of harvest production. In the 1990s, MEDFOR was developed in Chile (Morales and Weintraub, 1991; Epstein *et al.*, 1999a). Its aim was to allocate silvicultural regime to stands, and decide what land to acquire (Morales and Weintraub, 1991) while ensuring that timber production is maintained (Morales and Weintraub, 1991; Epstein *et al.*, 1999a).

FORPLAN, a later version of FOLPI (Manley and Threadgill, 1991) and MEDFOR all allocate timber to mills according to the mill's volume demand. None of the strategic planning systems reviewed included log bucking decisions, transport to the mill or wood properties.

2.5.3 Strategic/tactical planning

Some models and systems are classified by their authors as being either for strategic or tactical purposes, depending on the data that is input. Examples of these are Barros and Weintraub (Barros and Weintraub, 1982), FOLPI (Garcia, 1984, 1990; Manley and Threadgill, 1991; Papps and Manley, 1992; Manley *et al.*, 1996), RegRAM-1 (McGuigan and Scott, 1995) and Microforest Harvest Scheduling System (HSS) (Syndicate Database Solutions, 2006). Of these, all except the last uses linear programming; HSS uses simulation. The model described by Barros and Weintraub's planning horizon spans two rotations of 25 years. For the first few years of the planning horizon, the planning period is yearly, changing to two years per period for the intermediate part of the horizon and 5-10 years per period for the last part of the horizon. This model and RegRAM-1 both include log allocation and transport to the pulp mill, and they both include the possibility of transporting chips from local sawmills to the pulp mill. HSS includes bucking decisions. None includes wood properties in the decision-making process.

2.5.4 Tactical planning

The harvest scheduling problem at the tactical level includes more forestry detail, and the planning horizon is shorter. Tactical problems are also resolved at a smaller area scale than the strategic level problem (Weintraub and Cholaky, 1991; Church *et al.*, 1994). Since strategic-level systems aggregate the forestry area into similar groupings

to aid ease of solution, tactical level planning software's first job is to allocate specific stands which are due to have forestry activities performed on them from the strategic level's groupings (Nelson *et al.*, 1991; Papps and Manley, 1992; Church *et al.*, 1994; Murray and Church, 1995; McNaughton *et al.*, 2000). Some systems also include road making and maintenance in their formulations (Nelson *et al.*, 1991; Murray and Church, 1995; Weintraub *et al.*, 1995; Guignard *et al.*, 1998; Cea and Jofré, 2000; McNaughton *et al.*, 2000) as one needs an appropriate road network to access the harvested trees. Allocation of harvesting crews and equipment is also considered at this stage (Rönnqvist, 2003). The time horizon for the tactical problem ranges from three years (Cea and Jofré, 2000) to 15 years (Nelson *et al.*, 1991). This translates to between six and 20 percent of the length of the strategic planning horizon.

In native forests, environmental concerns cause additional spatial constraints to be imposed: in order to ensure that there is enough natural habitat for wildlife, the area that can be clearfelled at one time is restricted (Murray and Church, 1995; Weintraub and Bare, 1996; Martell *et al.*, 1998; McDill *et al.*, 2002; Weintraub and Murray, 2006; Baskent and Keles, 2005; Vielma *et al.*, 2007). There are further restrictions in that stands adjacent to the clearfelled stand(s) may not be felled until the clearfelled stand(s) have been replanted and those trees have reached a certain height, or until a certain time has elapsed. These clearfelling restrictions and so-called "green-up" constraints mean that spatial (or adjacency) constraints need to be imposed in the model (Brumelle *et al.*, 1998; Baskent and Keles, 2005; Weintraub and Murray, 2006; Vielma *et al.*, 2007). In plantation forests, these spatial constraints do not apply if the plantations are grown according to environmentally sustainable standards. For example in South Africa, only 65% of the land area owned by plantation companies is planted to trees; the rest is allocated to roads and conservation areas (Everard, 2000).

A type of spatial constraint is, however, applicable to plantation forestry: as it is expensive to move harvesting equipment from one location to another, it makes sense

to harvest other nearby stands with that equipment, if possible. Also, if the road leading to a particular group of stands has been upgraded, it would be beneficial to use it to transport as many logs harvested from local stands as possible (McNaughton, 1998; Mitchell, 2004, pp.1–2).

Because the tactical problem relates to cutting specific stands or building specific roads, a MIP formulation with 0-1 variables is used (Weintraub *et al.*, 1994, 1995; Bettinger and Chung, 2004). Heuristic techniques have also been used to solve these problems (Weintraub *et al.*, 1994, 1995; Bettinger and Chung, 2004). The objective function could either be to maximise profit or to minimise costs (Weintraub *et al.*, 1994).

Two models or systems having a tactical planning horizon of a few years were reviewed: Guignard *et al.*'s model (1998) and OPTIMED (Epstein *et al.*, 1999a; Weintraub *et al.*, 2000; Andalaft *et al.*, 2003). The aim of both of these is to plan the timber harvest including road making or upgrading. Guignard *et al.*'s model uses LP with constraint strengthening through lifting as well as branch-and-bound with double contracting to solve the problem. It works with individual stands. OPTIMED uses LP to solve the MIP problem, then uses heuristic rules to round the 0-1 variables. It works with aggregate stands. Years are divided into summer and winter seasons, as road-building and log transport cannot occur in the wet winter. It determines how much timber to cut, to store (in summer), to allocate to each mill and to transport (by road) to that mill. The harvesting machinery and trucks to be used is also determined by the system. Neither of the two includes log bucking capabilities or wood properties in the decision.

2.5.5 Operational planning

Operational-level harvest scheduling problems have a time frame of one year or less (Laroze and Greber, 1991; Karlsson *et al.*, 2004) or even as short as a number of weeks (Karlsson *et al.*, 2003; Mitchell, 2004, pp.1–2; Gordon *et al.*, 2006). Operational

harvest scheduling systems cover what trees to harvest (given that they are mature and can be accessed by an upgraded road) (Murray and Church, 1995; Martell *et al.*, 1998; Epstein *et al.*, 1999b; Karlsson *et al.*, 2003; Karlsson *et al.*, 2004; Mitchell, 2004, pp.53-73), what harvesting machinery to use (Martell *et al.*, 1998; Epstein *et al.*, 1999b; Karlsson *et al.*, 2003; Andersson and Eriksson, 2007), which harvesting crews to use (Gunn, 1991; Karlsson *et al.*, 2003; Karlsson *et al.*, 2004; Mitchell, 2004, pp.53-73; Gordon *et al.*, 2006), what weekly volume to cut (Epstein *et al.*, 1999b; Karlsson *et al.*, 2003; Mitchell, 2004, p.53-73; Gordon *et al.*, 2006), what bucking pattern to use (Martell *et al.*, 1998; Epstein *et al.*, 1999b; Mitchell, 2004, pp.53-73; Gordon *et al.*, 2006), and how to ensure that the right log products are delivered to satisfy demand (Gunn, 1991; Martell *et al.*, 1998; Epstein *et al.*, 1999b; Karlsson *et al.*, 2003; Karlsson *et al.*, 2004; Mitchell, 2004, pp.53-73; Gordon *et al.*, 2006). A mixture of operational research methods are used to solve these operational problems: MIP (Karlsson *et al.*, 2003; Karlsson *et al.*, 2004; Mitchell, 2004, pp.53-73; Gordon *et al.*, 2006) and LP (Epstein *et al.*, 1999b) (optimisation) as well as heuristic methods (Karlsson *et al.*, 2003) are common.

Bucking patterns play an important role when making logs for sawtimber, as sawmills require logs of different diameters, lengths and grades. The silviculture which has been applied to the stand also plays a role (e.g. pruning trees causes clear wood to form) (Turner and Price, 1996). Pulp logs, however, are generally of a standard length, which simplifies the bucking pattern. Some systems apply a heuristic algorithm to determine the yield from a bucked tree, while others go a step further and optimise how logs are bucked, so that the products made can be allocated to the market (Mitchell, 2004, pp.53-73).

Five operational harvest scheduling models or systems which use optimisation were reviewed: OPTICORT (Epstein *et al.*, 1999a,b; Weintraub *et al.*, 2000), Karlsson *et al.*'s short-term crew allocation model (2003), Karlsson *et al.*'s annual harvest

planning model (2004), Mitchell's operational forest harvest optimisation model (2004, pp.53-73) and the Atlas Market Supply prototype (Gordon *et al.*, 2006). Of these, OPTICORT, Karlsson *et al.* (2003), Karlsson *et al.* (2004) and Mitchell include harvesting decisions; OPTICORT, Mitchell and Atlas Market Supply prototype include bucking decisions; all include log allocation and transportation to the mill, and the Atlas Market Supply prototype was the only system which uses a wood property (density) – as well as stem characteristics – in the decision-making process for making saw logs. Both Karlsson *et al.* (2003, 2004) include age-related log storage costs.

A further model has been developed for a one year planning horizon (Beaudoin *et al.*, 2007): this model concentrates on the timber allocation part of the supply chain for an integrated company. It includes harvesting decisions and transport to the mill but does not include wood properties, although it does include wood freshness as a surrogate for wood properties.

Two systems which do not involve optimisation but which include wood properties are also reported on: Fibre Prediction System (FPS) (Megown *et al.*, 2000) and ARTLIS (Meynink and Borough, 2005). After the harvesting decisions have been made, FPS allocates logs to pulp mill processes depending on the processes' wood property requirements; ARTLIS is an inventory system which keeps track of logs so as to minimise log age (which affects pulp brightness).

2.5.6 Linking plans in the planning hierarchy

Because of the time frames and detail involved, it is difficult to create one formulation which will solve the all aspects of the forest harvest scheduling system with a single model (Martell *et al.*, 1998). Breaking the problem up in a hierarchical way has the advantages that it can achieve a solution (because the problem size is more manageable) (Gunn, 1991; Laroze and Greber, 1991; Bare, 1996) and the problems match the

decision-making hierarchy (Laroze and Greber, 1991; Weintraub and Cholaky, 1991; Church *et al.*, 1994; Bare, 1996; Weintraub and Davis, 1996). However, this hierarchical approach has the additional challenge that the different levels of the hierarchy must talk to each other (Bare, 1996), and care must be taken not to lose accuracy when aggregating and disaggregating data (Weintraub and Davis, 1996; Cea and Jofré, 2000).

A common approach with hierarchical systems is to run the strategic-level problem first (Rönqvist, 2003). The outcome of this is used as a constraint to the tactical plan, which is run later. Because the strategic-level problem generally uses aggregated stand groupings and does not include detail about stand area or spatial features (like the stand's relationships to other stands and roads, the land topology), some consider the outcome of the strategic plan to be the upper bound of solutions at lower levels of the hierarchy (Bettinger and Chung, 2004). However, because lower-level requirements are not being taken into account when solving the higher-level problem, solving the tactical problem often causes any optimality gained in solving the strategic problem to be lost (Guignard *et al.*, 1998; McNaughton *et al.*, 2000). Solving the strategic and tactical problems separately could also result in different answers – i.e. different volumes of timber to be harvested, or different costs to be incurred (Cea and Jofré, 2000). The aim of tactical planning should be to maintain the direction which has been set by the strategic plan, while incorporating the additional constraints (Carlsson *et al.*, 2006).

To overcome these problems, some semi-integrated models have been proposed. Yoshimoto *et al.* (1994) and Öhman and Eriksson (2002) added spatial considerations (which are typical of tactical level problems) to long-term plans. Smith (1978) proposed first developing a 20 year (tactical) wood flow plan, and then including this in a one or two rotation (strategic) plan to check that the short term plan does not cut more timber than the goals set out in the long term plan. Nelson *et al.* (1991) had a three-step approach to integrating short-term area-based logging plans with long-term harvest schedules. Firstly, a long-term plan was run. Stands were aggregated, and the

150 year time horizon was broken up into 15 ten-year intervals. Next, a model was run for the first three time periods (i.e. 30 years) on individual stands, using a random search technique called Monte Carlo Integer Programming. This model aimed at harvesting the volumes from individual stands which the strategic plan had estimated would be cut during its first three time periods. In this model, adjacency constraints were also taken into account. In the final step, a long-term model was run which incorporated the results of the second model. Road upgrading costs were included.

Weintraub and Cholaky (1991) proposed running the strategic plan by aggregating forest zone or region into areas of similar geography, silviculture or economic criteria. Tactical plans for each forest zone were then run, including outputs of the strategic plan for that zone as constraints. Some leeway was allowed with these constraints as errors of aggregation may have been incurred in the strategic plan. The sum of all the forest zones' results were then compared with the strategic plan's to ensure consistency; if they were not consistent, the aggregations of the strategic plan were changed, and the two models re-run.

Davis and Martell (1993, 1996) proposed incorporating both the strategic and the tactical models in a single model. The strategic model is run with one time period covering ten years, and a time period of one year in the tactical model. The tenth period of the tactical model is linked to the first period of the strategic model to ensure synergy between the two planning horizons.

McNaughton *et al.* (2000) proposed a large model where aspects of the tactical plan were linked with the strategic plan, and used column generation and constraint branching techniques to overcome the size of the problem. In this problem, a 30-year strategic plan was solved with aggregated stands, and linked to a tactical plan of six years. The tactical problem allowed for roads to be constructed prior to harvesting. It also ensured that there was a continuous route from the harvested site to the mill.

Adjacency constraints could also be imposed (e.g. if one wanted to harvest two stands that are close to each other within a short time).

Cea and Jofré (2000) proposed a top-down strategic model and a bottom-up tactical model, and an iterative procedure for reducing the gap between the answers. First, a 45-year strategic plan was run with aggregated stand groups and time periods of three years. The result of the first, three-year, period was then disaggregated and the three-year tactical problem solved. This model calculated which roads should be built or improved between a stand due to be harvested and a mill. An algorithm which minimises the difference between both level plans aggregated the tactical plan's stands by clustering, and these were fed into the first period of the strategic plan, which was then re-run. This procedure was repeated until the difference between both plans was sufficiently reduced.

Andersson (2005, p.24) proposed a two-level top-down integrated approach. A strategic plan which contains stands, but which excludes any spatial aspects, was run: one time period has a duration of 10 years. The results were then put into two categories: those which have harvesting activities in the first ten (tactical) years, and those that do not. (Stands due to be felled in the tactical time frame may not be moved to the non-tactical time frame.) With the tactical set of stands, another model was run which included both the tactical and strategic forestry management decisions. This model also included access to stands. The ten years of the second model were broken up into ten time periods, with three seasons per year.

2.6 Specifying a forest harvest scheduling system

2.6.1 Systems development

After deciding that a computer system is to be developed⁴, the first step is to interview experts in the domain and potential users to find out what the existing system (be it automated or not) does, and what they would like the new system (which is to be automated by software) to do. The features and functions of the existing and new systems are captured, usually in a natural language (like English). Further analysis is then undertaken by modelling certain aspects of the system or requirements. It is in undertaking this analysis that one becomes aware of details which were thought to have been understood, but whose detail was missing. The natural language description of the system's environment and features, and sometimes the models, make up the system requirements specification document (Sommerville, 2007, p.220), which is given to the domain experts and users to review and check that what they require, and the system's environment, have been captured correctly. There may be many cycles of elicitation and review to validate that the right system is being built (Zave, 1984) before the domain experts and users are satisfied with the system requirements specification.

Once there is agreement on the contents of the system requirements specification, the system is specified. Whereas the requirements state what the system *must* do, the specification says what it *will* do (Hall, 1998). As with the requirements specification, there is an iterative process of writing and review to ensure that the specification is correct.

After the specification has been captured, system design activities start. The difference between the requirements and specification phase of development and the design phase is that the requirements and specification state *what* the system must do, while the

⁴i.e. the necessary feasibility assessments have been undertaken

design states *how* it should do it (Yeh and Zave, 1980; Heitmeyer and McLean, 1983; Hall, 1990; Faulk, 1996). During the design phase, different hardware and software options are assessed and decisions made on the way to proceed. The design document can be seen as the blueprint from which programmers can code (program) the parts of the system. Once the coding of modules has been accomplished, they are tested, and gradually integrated with other modules (and tested together). The software is then installed on the hardware specified in the design, and retested, not only to ensure that the software works, but also to verify that the system to be delivered is the one described in the specification (Zave, 1984). After this, the system is handed over to the customer. If bugs⁵ are found in the program, they are fixed by the development team before customer acceptance, and by a maintenance team after acceptance.

It should be noted that there are several approaches to implementing the systems development lifecycle: the Waterfall method (Royce, 1970) of gathering and analysing *all* the requirements before proceeding to design, completing the design before any coding is started etc. is not as practical as the Spiral model (Boehm, 1988) and similar methods, which aim to elicit some requirements and build software around those, before adding other requirements. However, in each of these methods, one must have a statement of requirements before that part of the system can be designed and coded.

The first stage of systems development is not an exact science, and problems occur in it, or as a result of it. Sometimes requirements are missed (either forgotten by the domain experts, or thought to be so obvious that they are not mentioned); sometimes they are misunderstood; some requirements are stated or captured in an ambiguous way (although thought to be precisely-understood at the time of writing them); some requirements contradict others, or the domain experts cannot agree on them. It is difficult for analysts to learn enough about the system to be built, and users sometimes have unrealistic expectations about what systems can do for them (Gane and Sarson,

⁵i.e. features or functions of the software which do not work properly

1979, p.2). Many are of the opinion that the first stages of systems development (i.e. the requirements gathering, analysis and specification) are the most difficult of the systems development lifecycle (Gane and Sarson, 1979, p.1; Brooks (Jr.), 1986; Bowen and Hinchey, 1995b). As the System Engineer said: *“Writing the code isn’t the problem. Understanding the problem is the problem.”* (Curtis et al., 1988).

Requirements problems are hard and costly to fix the further along the development cycle the system has progressed (Boehm, 1985; Faulk, 1996), as rework in the later systems development phases would be needed to rectify them. It has been estimated that it could cost as much as 100 times more to fix such errors once the system has been installed and accepted by the customer (Boehm, 1985). Different methods are therefore used to analyse, uncover and fully understand the relationships and dependencies in the requirements of a system and its environment.

2.6.2 Methods for specifying systems

It is generally accepted that specifying using natural language alone is not precise enough, due to its ambiguous nature (Meyer, 1985; Ince, 1990; Morgan and Sufrin, 1993; IEEE Standards Board, 1994). Common practice is to develop diagrammatic models of the entities and data to enhance the natural language description (Yeh and Zave, 1980; Polack, 2001) (e.g. using the Structured Systems Analysis and Design Method (SSADM) (Ashworth, 1988), or Yourdon’s Structured Analysis (Yourdon, 1989) method). These models are called semi-formal (France and Docker, 1988; Pohl, 1993), because they are more precise than natural language, but they cannot be cross-checked automatically for inconsistencies.

There are many semi-formal models and modelling methodologies. Some examples of semi-formal models are entity-relationship diagrams (Chen, 1976), structured analysis diagrams (Ross, 1977), JSD (Jackson, 1983) and OOA (Coad and Yourdon, 1991).

These models are sometimes called essential models (McMenamin and Palmer, 1984), conceptual models (Mylopoulos, 1992; Faulk, 1996; Wieringa, 1998) or semantic models (Mylopoulos, 1992). Semi-formal methods have the benefit that they can be understood by non-specialists (Bowen and Hinchey, 1995a), they give a good overview of the subject matter, and are widely-used in industry (Plat *et al.*, 1991; Semmens and Allen, 1991). They rely on readers to review and cross-check them for ambiguities, but this is often inadequate (Bryant, 1995).

To overcome the problem of describing large, complex systems with many component interfaces, Zachman proposed a framework of models to describe the system at various stages of development (Zachman, 1987). This framework not only proposes a different type of model to answer each of the questions, 'what?' (data), 'how?' (function), 'where?' (network), 'who' (people), 'when?' (time) and 'why?' (motivation), it also proposes that a different model be drawn up to match the perspective of the different people who are involved in conceptualising and developing the system: the planner, the business owner, the system's designer, the system's builder and the system's coder (Hay, 2003, p.342). Each model is an important representation or view of the model, which highlights different aspects of the system, and communicates different features of the model. These models can be cross-checked to ensure better consistency and a more complete description.

Even if semi-formal methods are used, there is still the possibility of ambiguity and inconsistency in the specification (Bryant, 1995). A solution to this problem is to use formal methods, so-called because they are based on mathematics, and can be checked for consistency automatically. Formal methods can be used in several steps of the software development cycle: they can be used to help clarify the requirements (Wing, 1990) and to add precision to them (Craig *et al.*, 1993; Barden *et al.*, 1994, p.5). They can be used to prove that a more detailed version of a specification conforms to the more abstract one and also to prove properties

of the specification (Wing, 1990; Gaudel, 1994; Barden *et al.*, 1994, p.7). The specification can be animated using animation tools (this is akin to “running” the specification) (Barden *et al.*, 1994, p.7). Test cases can also be generated as a result of the formal language specification (Wing, 1990; Barden *et al.*, 1994, p.7). However, even if one does not perform proofs or check the system for consistency and integrity, it is still beneficial to write the specification in a formal notation without doing proofs (Wing, 1990; Bowen and Hinchey, 1995b), because of the value of getting the requirements right at the beginning of the development (Bowen and Hinchey, 2006).

The specifications developed using formal methods are more precise (Hall, 1990, 1996; Bowen and Hinchey, 1995b); they enable errors to be found and removed earlier in the development process (Hall, 1990; Bowen and Hinchey, 1995b, 2006), which helps to improve the quality of the system (Tretmans *et al.*, 2001; Bowen and Hinchey, 2006). Users of formal methods agree that they helped to reveal assumptions which were not explicitly stated, things which were inconsistent and unintentionally incomplete (Wing, 1990). Using formal methods helps to eliminate errors in the specification (Meyer, 1985; Hall, 1990; Bowen and Hinchey, 1995b, 2006), thus helping the developers to create a higher-quality product (Ledru, 1996; Bowen and Hinchey, 2006). It is cheaper to spend the extra time on using formal methods to specify a system (thus eliminating errors) than to spend the time debugging in the testing or maintenance phases (Bowen and Hinchey, 2006). Because of using formal methods, the English narrative in specifications was found to be more precise (Nix and Collins, 1988; Barden *et al.*, 1992), which confirms Meyer’s prediction that more precise natural language would be a byproduct of using formal methods (Meyer, 1985). In another development, customers found that the specification was comprehensive, and it could easily be used to develop test cases which meant that they could see the system’s level of testedness of the system constantly (Hall, 1996). In this development, the system documentation had to be revised very little, because of the use of formal methods. The use of formal methods in system development can give customers the assurance that their system will behave as expected (Hall, 1990).

Those not in favour of using formal methods argue that because they use a mathematical notation, specifications produced using formal notations are not understood by customers and often also developers (Bryant, 1995; Craigen *et al.*, 1995; France and Larrondo-Petrie, 1995; Luqi and Goguen, 1997; George and Vaughn, 2003). Many non-users believe that they need a high level of mathematics to implement formal methods (Hall, 1990). Some cite the lack of supporting tools as a reason not to use them (Ince, 1990; Luqi and Goguen, 1997), or the amount of time they take to produce (Bryant, 1995; George and Vaughn, 2003).

It is true that mathematical notation may not be immediately understood by those who have not had formal methods training; however it is proposed that specifications be interspersed with natural language descriptive text to aid understanding (Meyer, 1985; Gravell, 1991; Barden *et al.*, 1992; IEEE Standards Board, 1994; Bowen and Hinchey, 1995b). In a review of users of the Z formal notation, many specifiers were aiming to write as many lines of descriptive text as there were lines of formal notation (Barden *et al.*, 1992).

It has been proposed that semi-formal methods (diagrams) be included in formal specifications, as this would aid understanding and help those who do not understand the formal notation (Semmens and Allen, 1991; Semmens *et al.*, 1992; Polack *et al.*, 1993; Barden *et al.*, 1994, p.17; Bowen and Hinchey, 1995a; France and Larrondo-Petrie, 1995; Mander and Polack, 1995; Bruel *et al.*, 1998; Hinchey, 2002). Many have proposed a method for translating or converting their semi-formal methods into formal notations (e.g. France and Docker, 1988; Plat *et al.*, 1991; France and Larrondo-Petrie, 1995). Much work has been done on this, but if the semi-formal analysis or description is vague, the correct interpretation of the diagram may not be achieved (Bruel *et al.*, 1998). Another approach is to develop formal descriptions of the domain and system in a way which is informed by the semi-formal analysis, but not tightly coupled to

it. Doing this gives a complementary set of specifications which enables designers to concentrate on the aspects of the system relevant to them (Bowen and Hinchey, 1995a).

The mathematics on which formal methods are based is quite simple; often training in set theory and formal logic overcomes this objection (Hall, 1990). In fact, Tretmans *et al.* (2001) reported that they were better off without a highly trained mathematician in their development team, as the maths experts were seeking the most concise way to state things: how they expressed their statements was correct but it was not always readable by everyone on the team.

Regarding the lack of supporting tools, much has been done since the early 1990s to address this, and more work is underway (Bowen and Hinchey, 2006). Tools are needed for checking the syntax and the types of the formal specification (Utting, 1995; Saaltink, 1997), for schema expansion (Saaltink, 1997), for checking the proofs related to the specification (Utting, 1995; Bowen and Hinchey, 1995b; Saaltink, 1997) and for animating the specification (Barden *et al.*, 1994, p.14; Robinson, 2002).

It is true that writing formal specifications takes time, but the benefit is in having been made to think about the specification (Bryant, 1990; Hall, 1990). One has to think about what the system is about (Barden *et al.*, 1994, p.327), and one is forced to ask (and answer) questions which may have been unasked when using a more informal approach (Meyer, 1985). This very often translates into a more consistent and precise system description.

Another aspect to take into account when using formal methods in the development lifecycle is that the time allocated to specification and design activities will be lengthened (Ince, 1990; Tretmans *et al.*, 2001; George and Vaughn, 2003). However, the time taken to test the system will be shorter (1994, Barden *et al.*, p.12; Tretmans *et al.*, 2001), as many more errors are removed in the specification stage (Bowen

and Hinchey, 1995b). This shift in the expenditure of time (and therefore money) in development may be construed negatively, especially by managers who have a budget to develop systems but not to maintain them (Hall, 1990). Managers should be made aware of this shift, and the added benefit of a higher quality system with fewer errors.

There are many different formal method notations available for the specification of systems. Clarke *et al.* (1996) give three categories: those which are used to specify the behaviour of sequential systems (like Z (pronounced 'zed') (Spivey, 1998), VDM (the Vienna Development Method) (Jones, 1990) and Larch (Guttag and Horning, 1993)); those which are used to specify the behaviour of concurrent systems (like CSP (Communicating Sequential Processes) (Hoare, 1985), CCS (a Calculus of Communicating Systems) (Milner, 1980), Statecharts (Harel, 1987)); and those which combine these two (e.g. RAISE (a Rigorous Approach to Industrial Software Engineering) (The RAISE Language Group, 1992) and LOTOS (Language Of Temporal Ordering Specification) (van Eijk *et al.*, 1989)). The forest harvest scheduling system belongs to the first category. Formal methods for sequential systems model system states with sets, relations and functions. System behaviour is modelled by using pre- and post-conditions to assert what must be true before and after the operation (Sommerville, 1996, p.169; Clarke *et al.*, 1996).

There are several criteria for choice of a formal method. Firstly, it must be an appropriate notation for the task at hand (i.e. not using a concurrent notation for a sequential system and vice versa) (Bowen and Hinchey, 1995b). It helps to have a notation which enables the specification to be built up in a modular fashion (Nix and Collins, 1988; Ince, 1990). It is advantageous to use an established notation with many users (Bowen and Hinchey, 1995b). It is necessary to have the support of tools (Tretmans *et al.*, 2001). It is beneficial to have access to literature which teaches the notation and gives examples of specifications using the notation (Craigien *et al.*, 1995) and to have a guru available from whom help can

be requested (Nix and Collins, 1988; Bowen and Hinchey, 1995b; Tretmans *et al.*, 2001).

2.6.3 What is included in a specification?

The aim of a specification is to describe the system to be built in terms of its functions and behaviour, constraints and interfaces (IEEE Standards Board, 1994). It should also give the inputs and outputs of the system (IEEE Standards Board, 1994; DeMarco, 1997, pp.212–213), and contain a description of any system complexity (Alagar and Periyasamy, 1998). It should give the response to all expected types of input data and all types of situations (IEEE Standards Board, 1994). The specification should contain the outputs of any semi-formal and/or formal analyses undertaken (IEEE Standards Board, 1994).

Specifiers must take care to choose the right level of abstraction for the specification (Wing, 1990). A very abstract specification is not helpful, as it would be incomplete. On the other extreme, a specification with too much detail may lean towards a certain implementation bias (i.e. design) (Morgan and Sufrin, 1993; Bowen and Hinchey, 1995b), which would leave the designer with less choice for implementation (Wing, 1990). The specification must have enough detail for the reader to understand what the system does, without adding unnecessary detail (Hall, 1990).

Capturing the environment, or domain, in which the proposed system is to be situated is an important part of a specification (Jackson, 1995; Bjørner *et al.*, 1997; Jackson, 2001; Bjørner, 2006a,b, 2007). This step has been advocated by many authors over nearly 20 years (e.g. DeMarco, 1979, pp.27–28; Yourdon, 1989, pp.333–357; Dorfman, 1990; Holland *et al.*, 1994; Sommerville and Sawyer, 1997b, pp.169–172; Luqi and Goguen, 1997; Weaver *et al.*, 1998, pp.39–41; Robertson and Robertson, 2006, pp.44–45), but is often neglected (Jackson, 2005; Cox *et al.*, 2005), often with costly and detrimental results (Jackson, 2001, p.2). Domain descriptions are important,

because often the complex rules which impact the system are buried in the domain (Luqi and Goguen, 1997).

The domain (or system environment) should be described as it is at present (Jackson, 1995, p.166; Bjørner, 2006b, p.8), without references to requirements or future systems (Bjørner, 1999). They aim to capture all the aspects of the domain (Bjørner, 1999) in a set of precise, abstract models which domain experts can agree upon (Bjørner, 1998; Siau, 2004; Bjørner, 2006b, pp.343–349). These models help the analysts to understand what the entities of the domain are, how they interrelate with each other, what actions (functions) are applied to them (and when, by whom/what), and how they are changed by these activities. They help requirements gatherers to develop developing complete, consistent and accurate requirements (Sutcliffe and Maiden, 1998). Customers of future systems can check the description to ensure that their 'reality' is adequately captured by the models (Kang *et al.*, 1990). They help system designers to design efficiently, since the ability to design future systems for the domain efficiently depends on the degree to which knowledge about the domain has been captured (Greenspan *et al.*, 1982). They also help the developers (Barden *et al.*, 1994, p.9) and maintainers of systems in that domain to understand the context better (Burton-Jones and Meso, 2008).

2.6.4 Specifications of forest harvest scheduling systems in the literature

While many authors have described mathematical models which can be used to aid forest harvesting decision-making and forest management decisions in general, very few have reported on forest planning system specifications from a computer science point of view. Böhlen *et al.* (1998) describe the requirements for a spatio-temporal relational database system, using data for a simplified forest harvest scheduling system in their examples. Baskent *et al.* (2001) give a conceptual framework for the design

of forestry management problems using object-oriented techniques. This work covers natural forestry management, but is the most thorough analysis of the forestry domain. Nobre and Rodriguez (2005) describe the data modelling aspect of large forest harvest scheduling problems, and Ribeiro *et al.* (2005) use the Zachman framework (Zachman, 1987) to design the enterprise architecture for an integrated forest planning system. The latter two papers refer to the plantation forestry industry.

2.7 Conclusion

A growing number of authors agree that there is a need to deliver less variable wood to processors (Duchesne *et al.*, 1997; Spångberg, 1999; Lundqvist, 2003; Carlsson and Rönnqvist, 2005). Amongst the many models and systems which aim to address some aspect of the forestry supply chain, only one optimising system (Gordon *et al.*, 2006) made harvesting decisions based on wood properties (wood density) (see Tables A.9 to A.12 on pages 237 to 240), and this system had a short planning horizon of eight weeks. In this system, the wood properties were assessed three to five years before harvesting. Megown *et al.*'s system (2000) did include wood properties, but it allocated wood to the mill only after the harvesting decision was made. This system does not optimise, and cannot deliver wood of similar properties to a mill's process over a number of years. Weigel's model (2005, pp.22–33) also has the capabilities to include wood properties in the log sorting classes, but its aim is to maximise the pulp manufacturing value chain, and does not include forestry decisions. Three sources (Mikkonen, 1996; Meynink and Borough, 2005; and Beaudoin *et al.*, 2007) used wood freshness as a surrogate of wood quality.

Megown *et al.* (2001) showed that in order to ensure that a long-term supply of wood with similar properties is available for the mill, it is necessary to be able to predict the wood properties for longer than six years. This implies that a strategic model

must be used. However, as stated by many authors (Smith, 1978; Weintraub and Cholak, 1991; Nelson *et al.*, 1991; Davis and Martell, 1993, 1996; Cea and Jofré, 2000; McNaughton *et al.*, 2000; Öhman and Eriksson, 2002; Andersson, 2005), there is a need to make the outputs of strategic models more realistic by including tactical model criteria. The model also needs to include transport costs and distance so that the right volume of logs, having the required properties, can be delivered to the mill over time, at the lowest cost to the supply chain.

As a strategic-level forest harvest scheduling system which aims at delivering wood with more uniform properties to a pulp mill does not exist, one needs to be specified. This specification needs to include aspects of the tactical plan as outlined in section 2.5.6. Because of the complexity of the system, techniques such as formal methods (which reduce ambiguities in the specification) should be used, but for readability, semi-formal methods should be included to create a complementary specification. A description of the forestry and forest-to-mill domain is necessary to record the complexity of the environment in which the forest harvest scheduling system will be operational. It is proposed these be specified from the Business owner's point of view (Zachman, 1987). The next chapter describes the methods used to specify the forest harvest scheduling system and the domains in which it is embedded, using both semi-formal and formal techniques.

Chapter 3

Methods and techniques

3.1 Introduction

As described in the previous chapter (section 2.6.1), systems development starts with a series of interviews with domain experts and potential users. The interviews used for this study took place as part of a project, the aim of which was to develop a forest harvest scheduling system which would incorporate wood properties in the harvesting decisions. The domain experts and potential users interviewed worked for an integrated plantation forestry company in South Africa. These included the planning forester, regional foresters, estate foresters, the systems analyst, the database administrator, the IT specialist and the logistics manager. During the requirements gathering phase, every attempt was made to understand plantation forestry as it is undertaken world-wide, rather than concentrating on the particular company's implementation. This was achieved by comparing the outcomes of the interviews with books on plantation forestry (for example, Shepherd, 1986; Leuschner, 1990; von Gadow and Bredenkamp, 1992; Savill *et al.*, 1997; Evans and Turnbull, 2004) as well as other literature.

The system requirements for this project were captured in Easton *et al.* (2003), but neither semi-formal or formal analyses of the system or its domain were included in that document. The work undertaken in this dissertation is based on these interviews. Al-

though every attempt was made to make the analyses applicable to plantation forestry world-wide, they may represent a South African reality.

As mentioned in the literature review (section 2.6.3), capturing the environment, or domain, in which the proposed system is to be situated is an important part of a specification (Jackson, 1995; Bjørner *et al.*, 1997; Jackson, 2001; Bjørner, 2006a,b, 2007). The domains in which the forest harvest scheduling system is embedded were thus identified. For each of these domains, a list of applicable actions and constraints was developed. These were useful for both the semi-formal and formal analyses.

For each domain, semi-formal models were drawn up, accompanied by English descriptions. The Zachman framework was used to structure them (see section 3.2 for more details). These models were verified by the main interviewee (the planning forester) and two other forestry experts. Thereafter, formal specifications were developed using a formal notation (see section 3.3). The formal specification underwent two types of reviews at various stages of completion of the specification. The first was a series of internal reviews (by the author), where the specification was compared to desirable features of Z specifications found in literature, (for example, Barden *et al.*, 1992; Barden *et al.*, 1994, pp.24–26). The second series of reviews was undertaken by three people: a forestry expert (not familiar with the formal notation used) reviewed the narrative text of the formal specifications. Two experts in the formal notation (one who had a knowledge of forestry and on who did not) reviewed the entire specification. The result of the reviews were incorporated in the specification. The outcome of the reviews is discussed in Chapter 6.

During both the semi-formal and formal analyses, it was realised that certain aspects of the forestry and transport domains needed further clarity. This is expected during such analysis exercises (Meyer, 1985; Bryant, 1990; Hall, 1990). Domain experts were consulted to gain the necessary clarification.

Because of the difficulties which are experienced when attempting to develop formal descriptions which have a one-to-one correspondence with the semi-formal description (Bruel *et al.*, 1998), the approach used was to create complementary formal descriptions of the domain and system, as advocated by Bowen and Hinchey (1995a). The formal models were not developed independently of the semi-formal models: based on the entity-relationship diagrams, lists of sets, functions and constraints which applied to the domains and system were drawn up. The semi-formal models were also used as an '*aide-memoire*' when developing the formal models, so that important details would not be omitted. France and Larrondo-Petrie (1995) give two categories of processes in developing formal specifications from semi-formal specifications: one is based on generating the specification from the semi-formal specification using rules, while the other is based on cognitive skill. In this study the latter was used. The steps followed in developing both semi-formal and formal specifications is shown in Figure 3.1.

The next section (section 3.2) describes the methods and techniques used to write the semi-formal specification and section 3.3 describes the formal specification methods and techniques used.

3.2 Semi-formal analysis

Because of the benefits of model cross-checking to obtain a more complete analysis of the domain and the system, the Zachman framework (Zachman, 1987) was used to structure the semi-formal analysis. The version of the framework adapted by Hay was used (Hay, 2003, p.3). As described in the literature review, the domain and system were described from the Business or Enterprise Owner's point of view (i.e. the second row of Table 3.1).

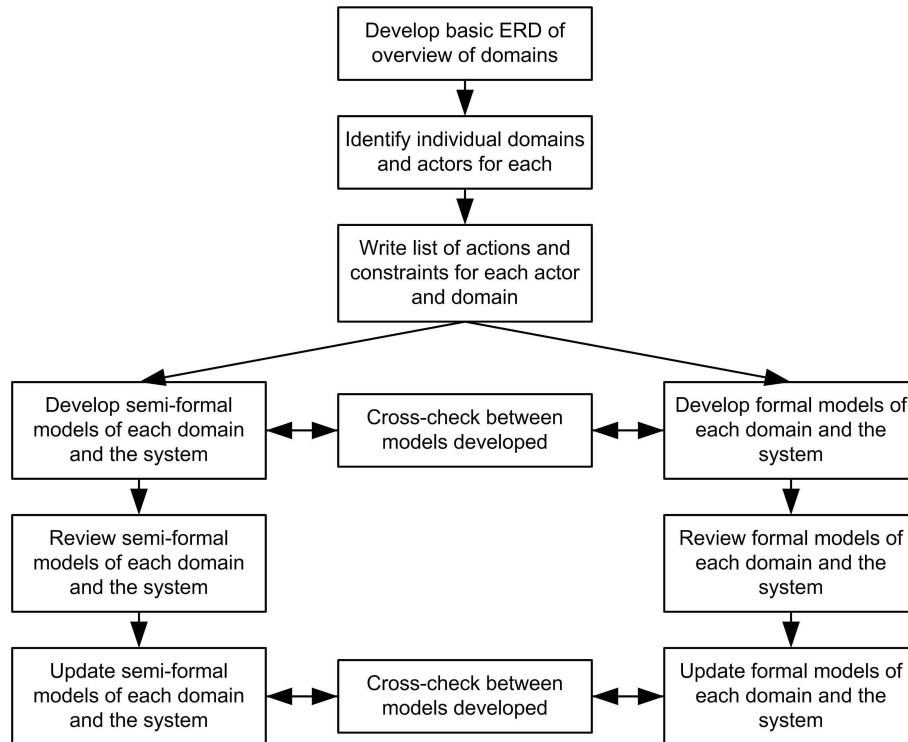


Figure 3.1: Outline of the method followed in generating the semi-formal and formal specifications

The 'Data' model of the Business owner's row has two parts. One is the language of the business owner, which includes the terminology and their meanings; the other is a divergent data model – so-called because it includes many data items which are not necessarily related¹ (Hay, 2003, p.7). This data model is a basic entity-relationship model, with nouns presented by entities and the lines representing the relationships. The business terminology can be found in the glossary on page xxiv. This list is particularly useful when words could have specific meanings, or nuances of meanings, to different people (Sommerville and Sawyer, 1997a, p.51). It is also important, as not everyone involved will have a similar understanding of all the terms (Morgan, 2002, p.18).

To develop the basic entity-relationship diagrams (ERDs), the system or domain was

¹By the time the architect has described the system in the third row, the data will be convergent, i.e. the important data will be included in an entity-relationship model.

Table 3.1: Zachman framework (from Hay (2003, p.3))

	Data (what?)	Activities (how?)	Locations (where?)	People (who?)	Time (when?)	Motivation (why?)
Objectives/ Scope (Planner's view)	List of things important to the enterprise	List of processes the enterprise performs	List of enterprise locations	Organization approaches	Business master schedule	Business vision and mission
Enterprise model (Business owner's view)	Language, divergent data model	Business process model	Logistics network model	Organization chart	State/ transition diagram	Business strategies, tactics, policies, rules
Model of fundamental concepts (Architect's view)	Convergent E-R model	Essential data flow diagram	Location of roles	The viable system, use cases	Entity life history	Business rule model
Technology model (Designer's view)	Database design	System design, program structure	Hardware, software distribution	User interface, security design	Control structure	Business rule design
Detailed representation (Builder's view)	Physical storage design	Detailed program design	Network architecture, protocols	Screens, security coding	Timing definitions	Rule specification program logic
Functioning system	Converted data	Executable programs	Communications facilities	Trained people	Business events	Enforced rules

described in a few sentences, and the 'subject-verb-object' phrases extracted. These subjects and objects (nouns) usually became the entities (but abstract nouns could become the attributes of the entity). Entities are represented in rectangles. The verb was put into a rounded rectangle with a line joining the two relevant entities. This is similar to the approach used by Chen (1983), except that in his notation, verbs are shown in diamonds. The ERDs presented here should be read in the direction of the arrows, from noun (or noun phrase) to verb to noun (or noun phrase). If the second noun (or noun phrase) has an arrow exiting from it, a new sentence must be started with that noun (or noun phrase) (see Figure 3.2). The most simple, abstract form of the domain was captured in an ERD, and was later expanded to capture more of the intricacies present. These resulting diagrams were grouped according to their subject matter so that one diagram relates to a subject in the domain (Wieringa, 1998).

Although an ERD could be used to describe the make-up of the genetic material of the trees planted in the stand, it was thought that they are better represented in a hierarchical diagram. The taxonomic structure of the genus and species lends itself to

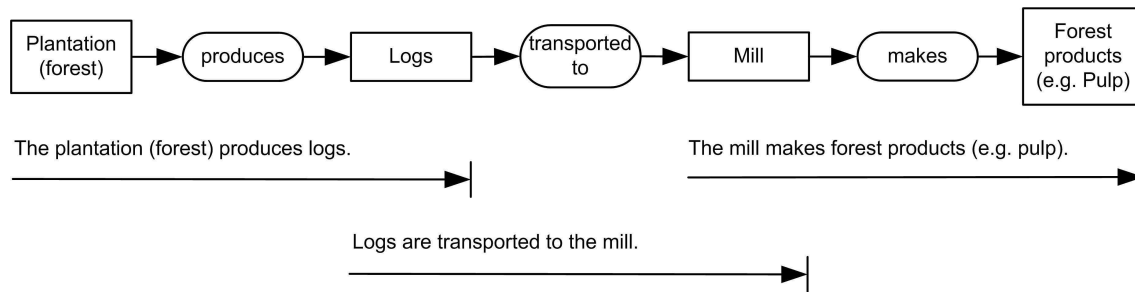


Figure 3.2: Example of how an entity-relationship diagram (ERD) should be read

this treatment.

For the 'Activities' column, a business process diagram was produced, using the Business Process Modelling Notation (White, 2004). These diagrams are also called Swimlane diagrams, because in these diagrams, there is a row (or swimming lane) for each person or department's responsibility (Hay, 2003, p.186; White, 2004). The diagram shows what the processes are, and their sequence of occurrence and dependencies, and who is responsible for doing each. Each diagram is annotated with a legend to aid interpretation.

For forestry companies, the 'Locations' and the 'People' models are related. This is because the land overseen by the foresters is managed in a hierarchical manner (Weintraub and Bare, 1996; Martell *et al.*, 1998), and the foresters in charge of these levels match the hierarchical structure. Both these models are represented as organisational hierarchy diagrams (Bachman, 1969).

To model the 'Time' aspect, a state chart or transition diagram is used to show how different states respond to events (Wieringa, 1998; Hay, 2003, p.259; Sommerville, 2007, p.176). These diagrams have been used to describe the required workings many systems, including user interfaces (for example, Parnas, 1969; Wasserman, 1985), computer security (Ilgun *et al.*, 1995) and discrete-event systems (Harel, 1987). In these diagrams, circles describe the states and labelled arrows show the events which will lead

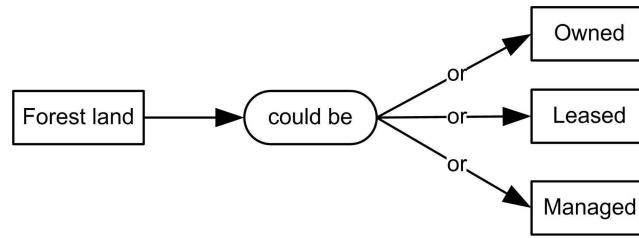
from one state to another (Davis, 1988; Wieringa, 1998). The initial state is denoted by small arrow (Wieringa, 1998).

The 'Motivation' of the system is modelled as business rules. These are statements which define or constrain a particular part of the business (Business Rules Group, 2000), or the way in which the business works (Rosca *et al.*, 1997; Kardasis and Loucopoulos, 2004). Their name derives from the fact that these rules were made explicit when developing business-oriented systems; these rules or constraints can, however, be applied to systems in general, and do not have to be confined to a business or organization. Business rules can be categorised into four types: terms or definitions, facts, derivations and constraints (Business Rules Group, 2000; Hay, 2004). The terms or definitions are usually put into glossaries (Business Rules Group, 2000). The facts cover the relationship between terms (von Halle, 2002, pp.31–37; Business Rules Group, 2000). Derivations transform knowledge from one form into another (Business Rules Group, 2000). Constraints could be mandatory rules or guidelines, or rules which cause an event to happen (Morgan, 2002, p.108; von Halle, 2002, pp.31–37). Many of these rules are generated when performing the analysis for the Architectural view of the system (Morgan, 2002, p.21), so only the rules constraining or causing the business activities are described.

In the semi-formal specification, a modified form of ERD (an example of which is shown in Figure 3.3) was used to describe the business rules. Other constraints of the business activities are listed in Tables 4.3, 4.6, 4.8 and 4.9 on pages 69, 92, 97 and 102 respectively.

Having developed the individual models, they were cross-checked to test for inconsistencies.

Microsoft® Office Visio® Professional 2003 was used as the diagramming tool.



The forest land could be owned or leased or managed

Figure 3.3: Example of a business rule represented by an entity-relationship diagram

The semi-formal specification of the domains involved and the forest harvest scheduling system are presented in Chapter 4.

3.3 Formal analysis

This section covers the methods and techniques used when using formal analysis to write a formal specification. First, the choice of the formal notation Z is discussed in section 3.3.1. This notation is described further in section 3.3.2. The notational conventions employed when writing the Z specifications is discussed next (section 3.3.3). The 'established strategy' (or method) for developing Z specifications is described in section 3.3.4. The approach to finding an appropriate level of abstraction when specifying a system is explained in section 3.3.5, and section 3.3.6 covers the tools used when developing the specification.

3.3.1 Choice of formal method

In the literature review (section 2.6.2), criteria for choosing a formal method were given. The notation must be suitable (i.e. not using a concurrent notation for a sequential system and vice versa) (Bowen and Hinchey, 1995b). A notation which supports the

modular development of a system is preferred (Nix and Collins, 1988; Ince, 1990; Hall, 1996). It is an advantage to use a notation which is established and has many users (Bowen and Hinchey, 1995b). Similarly, it is an advantage to use a notation which has a wide body of supporting literature (Craigien *et al.*, 1995). The support of tools is necessary (Tretmans *et al.*, 2001), as is a guru from whom help can be requested (Nix and Collins, 1988; Bowen and Hinchey, 1995b; Tretmans *et al.*, 2001).

The Z (pronounced 'zed') notation was found to be fit these requirements. There are several factors in favour of using Z for this study. Z can be used to create modular specifications (Ince, 1990), and it is a suitable notation. It has a large support base in the U.K. (Barden *et al.*, 1994, p.327), and in Europe (Dean and Hinchey, 1995). There is a wide body of literature available about Z: for example, there are textbooks (Wordsworth, 1992; Barden *et al.*, 1994; Diller, 1994; Bottaci and Jones, 1995; Woodcock and Davies, 1996; Jacky, 1997; Spivey, 1998; Lightfoot, 2001), sample specifications (Hayes, 1987, 1993; Bowen, 1996; Abrial *et al.*, 1996; Hinchey and Bowen, 1999), numerous conference proceedings, such as (Hutter *et al.*, 1999; Valentine, 1995; Bert *et al.*, 2003), and journal articles (Woodcock, 1989; Cooke *et al.*, 1996; Zave and Jackson, 1996), amongst many others. Bowen gives a bibliography of Z-related literature (Bowen, 1998). There is also an online archive (Z Archive, 1996). Since formal methods are not taught in undergraduate Computer Science courses in South Africa, having a "guru" available is important, as well as having access to sufficient literature. A type checker for Z is freely available on the internet (see section 3.3.6).

3.3.2 Introducing the Z notation

Z (pronounced 'zed') is more of a formal notation than a formal method (Ince, 1990), and was developed in the late 1970s (Bowen, 1996, p.xii). It is based on set-theory and predicate logic (Alagar and Periyasamy, 1998, p.281). Types in the specification are represented by sets, and different types are distinctly different (disjoint) sets (Craigien

et al., 1993). A set contains an unordered number of elements of the same type (Craig *et al.*, 1993). An element of a set could be simple (i.e. of a single type) or it could be made up of many types, in which case it is called a tuple. Tuples are ordered collections of elements, which are not necessarily of the same type (Craig *et al.*, 1993). An example of a tuple is a $(name, phone_number, birth_date)$ combination.

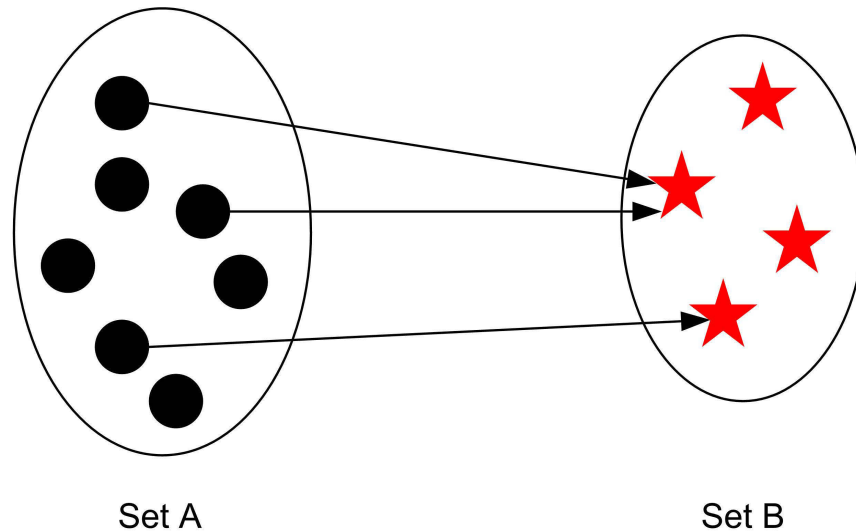


Figure 3.4: Relation between elements of two sets. Set A is called the domain of the function and set B the range.

Functions and relations can be defined to relate one set's elements with another set's elements. Figure 3.4 shows this; elements of either set could be excluded from the function. The elements of either set could be of a simple (single) or complex (tuple) type. The starting set for the function is called the domain and the ending set the range. They could also be called the input and output of the function. An example of a function taking a simple type and giving a tuple would be the function whose input was someone's ID number and the output was their name and birth date, i.e. $f(ID_number) = (name, birth_date)$.

Z uses schemas to present small chunks of the model (Somerville, 2007, p.230). Be-

cause of schemas, models can be built up in a modular way.

<p><i>Schema name</i> _____</p> <p><i>Declarations (Signature) : options are</i></p> <ul style="list-style-type: none"> – <i>include other schemas as part of this schema</i> – <i>declare functions</i> – <i>include inputs?</i> – <i>include outputs!</i> <hr/> <p><i>Predicates : options are</i></p> <ul style="list-style-type: none"> – <i>include constraints</i> – <i>include pre – conditions of actions</i> – <i>include post – conditions of actions</i>
--

The schema is referred to by its name. When it is included in another schema, it is as if the 'code' for the included schema is typed into the including schema. Options for including schemas are including it 'as is', or including it but not allowing the including schema to change any values in the included schema (\exists SchemaB – read as 'Xi SchemaB'), or including it but allowing the including schema to change values in it (Δ SchemaC – read as 'Delta SchemaC').

<p><i>SchemaName</i> _____</p> <p>SchemaA</p> <p>\existsSchemaB</p> <p>ΔSchemaC</p> <hr/> <p>...</p>

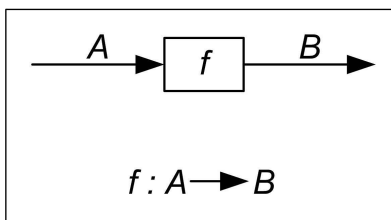


Figure 3.5: Two ways of representing the same function (from Meyer (1985)). The name of the function is f ; the input (domain) is A and the output (range) is B .

When declaring a function, it needs to be named, and the type (or types) of the domain (input) and the range (output) of the function need to be specified (see Figure 3.5). Before declaring the function, the types used by the function must be declared: here the *IDNUMBER* is declared as a finite set of natural numbers, and *NAME* and *DATE* are two distinct types. In the schema that follows (called *GetNameAndBirthdate*), the function *getNameAndBirthdate* is declared. Given a person's ID number, the function will give that person's name and birth date (shown by the complex type $(NAME \times DATE)$).

<i>IDNUMBER</i> : FN		
[<i>NAME</i> , <i>DATE</i>]		
<table border="1" style="border-collapse: collapse; width: 100%;"> <tr> <td style="padding: 5px;"> <i>GetNameAndBirthdate</i> _____ <i>getNameAndBirthdate</i> : <i>IDNUMBER</i> \rightarrow (<i>NAME</i> \times <i>DATE</i>) </td> </tr> <tr> <td style="padding: 5px;">...</td> </tr> </table>	<i>GetNameAndBirthdate</i> _____ <i>getNameAndBirthdate</i> : <i>IDNUMBER</i> \rightarrow (<i>NAME</i> \times <i>DATE</i>)	...
<i>GetNameAndBirthdate</i> _____ <i>getNameAndBirthdate</i> : <i>IDNUMBER</i> \rightarrow (<i>NAME</i> \times <i>DATE</i>)		
...		

In the first schema shown, two of the declaration options were inputs and outputs. In *Z*, a variable which is an input has a question mark after it (as in *identityNumber?*) and outputs have exclamation marks after them (as in *birthDate!*).

In the predicate (second) part of the schema, constraints on the schema's actions are listed. In the example below, the identity number which is an input to the schema *GetNameAndBirthdate* must be in the domain of the function and the couple $(name!, birthDate!)$ must be in this function's range. When evaluated at *identityNumber?*, the function *getNameAndBirthdate* gives the output $(name!, birthDate!)$. Each of the lines in the predicate part of the schema should be read as if there is an AND between them – i.e. they all have to be true (unless there is a disjunction (\vee) between the statements) (Sommerville, 2007, p.230).

GetNameAndBirthdate

$getNameAndBirthdate : IDNUMBER \rightarrow (NAME \times DATE)$

$identityNumber? : IDNUMBER$

$name! : NAME$

$birthDate! : DATE$

$identityNumber? \in \text{dom } getNameAndBirthdate$

$(name!, birthDate!) \in \text{ran } getNameAndBirthdate$

$getNameAndBirthdate(identityNumber?) = (name!, birthDate!)$

When describing an action in Z, there are conditions which must hold true before the action has taken place and those which must hold true afterwards. These pre- and post-conditions are also placed in the predicate part of the schema. In Z, the post-action state of a variable or function is denoted with a dash ('), as in *standStatus'*. In the simplified schema excerpt below, before a stand of trees is harvested, it must be in the *planted* state. After it is harvested, it returns to the *unplanted* state.

$PLANTINGSTATE ::= unplanted \mid planted$

HarvestStand

...

$standStatus : PLANTINGSTATE$

...

...

$standStatus = planted$

$standStatus' = unplanted$

...

In addition to including schemas in other schemas to build up the model, schemas can be created by conjoining (using \wedge) or disjoining (using \vee) schemas. In the following example, a schema called *BigSchema* is made up by combining *Schema1* and either *Schema2* or *Schema3*.

$BigSchema \hat{=} Schema1 \wedge (Schema2 \vee Schema3)$

Although including schemas in others to expand the model using the Ξ and Δ notations is the most common, this is not the only way of using Z schemas (Stepney *et al.*,

2003). Schemas can be used to define ‘records’ of data which can then be defined as types. The example below shows that the type *DATE* is a finite set of the type defined in schema *DateDefinition*. This means that all the constraints of the values of *year*, *month* and *day* are included in the type *DATE*. The individual ‘fields’ of *DATE* (*year*, *month* and *day*) can be accessed via the ‘dot’ notation. An example of this is shown in the (very simplified) schema *AgeGivenTwoDates*, where the years of the two input dates (*date1?* and *date2?*) are compared and the age calculated and output (*age!*).

<i>DateDefinition</i> $year, month, day : \mathbb{N}$... <i>Constraints on values of year, month and day</i> ...

$DATE : \mathbb{F} \textit{DateDefinition}$ $AGE : \mathbb{F} \mathbb{N}$
--

<i>AgeGivenTwoDates</i> $date1?, date2? : DATE$ $age! : AGE$ <hr/> $age! \geq 0$ $(date1?.year \leq date2?.year) \Rightarrow$ $age! = date2?.year - date1?.year$ $(date1?.year > date2?.year) \Rightarrow$ $age! = date1?.year - date2?.year$...

A list of Z notations used in this dissertation can be found in the Glossary (page xxxiv). More information about the Z notation and its use can be found in textbooks (for example, Wordsworth, 1992; Barden *et al.*, 1994; Diller, 1994; Bottaci and Jones, 1995; Bowen, 1996; Woodcock and Davies, 1996; Jacky, 1997; Spivey, 1998) and papers (for example, Woodcock, 1989; Valentine, 1995; Cooke *et al.*, 1996).

3.3.3 Notation conventions used in this dissertation

Layout and naming conventions are helpful when developing Z specifications (Gravell, 1991; Tretmans *et al.*, 2001). In an attempt to help the readability of the specification, schemas used in the specification were given names starting with capital letters and the rest of the name in camel case (i.e. words run together, with beginnings of new words having a capital letter) – for example, *RegimePlantAge*. Functions were given names starting with lower case letters, with the rest of the name in camel case (if necessary), for example *plantAge*. Functions with similar names to schema names were declared in the schema of the same or similar name (for example, the function *plantAge* was declared in schema *RegimePlantAge*).

In the declaration part of each schema, the schemas to be included were declared first, followed by functions and inputs and outputs. In the predicate part of the schema, the constraints on the included functions (declared or included) were specified before any changes to the state were made. As Gravell (1991) suggests, when variables were declared in the schemas, an attempt was made to give them names which corresponded to their type, for example,

$$\forall sID : STANDID \exists mID : MILLID \bullet \dots$$

and to use the same variable name for variables of that type throughout the specification.

As specifications should be accompanied by narrative text (Meyer, 1985; Gravell, 1991; Barden *et al.*, 1992, 1994), the narrative of each schema or section of the specification precedes the Z notation it is describing.

3.3.4 Method for developing Z specifications

The 'established strategy' for developing Z specifications is firstly to describe the background definitions. The system's normal state is then described. The system is then

initialised. Once this is done, actions which change all or part of one state to another state are defined (as if the system is working as expected). Next, schemas which describe the system if it is not working as expected (i.e. exceptions) are written. Schemas which combine the operations and the exceptions are written. All of these descriptions are combined to form a full specification. The Z notation should be accompanied by natural language descriptions, and an index of schema names (Barden *et al.*, 1994, pp.20–21).

3.3.5 Level of abstraction chosen for the specification

In the literature review (section 2.6.3), it was noted that the level of abstraction used in the specification is important. It should not be too abstract, as this will obscure the system's purpose (Hall, 1990). On the other hand, it should not contain too much detail, as this may lean towards a particular implementation bias (Wing, 1990; Morgan and Sufrin, 1993; Bowen and Hinchey, 1995b).

The approach taken was to write a very abstract version of the specifications for the forest-to-mill domain. This was based on an ERD which described the forest-to-mill domain. The actors for these entities were then identified, and tables of actions and constraints applying to each were captured. Basic schemas of each entity were then developed, and the actions which would change the entity's state (for example, planting a stand, harvesting a stand, logs arriving at the mill, etc.). This specification is presented in Chapter 5 (section 5.1).

This initial specification was written at the most simple level of abstraction possible. Items not yet in the specification, but needed for the Forest Harvest Scheduling System specification were then identified. These were added to the specification and incorporated into appropriate schemas (Chapter 5, section 5.2). Based on this domain specification, the specification for the Forest Harvest Scheduling System was developed. This

can also be found in Chapter 5, section 5.3. The full specification for the forest-to-mill domain can be found in Appendix D. The specification of the forest harvest scheduling system is given in Appendix E.

3.3.6 Tools used

The MiKTeX environment used to develop the Z specifications (and this dissertation) was MiKTeX 2.5. The editing tool used to write the \LaTeX files was TeXnicCenter version 1 Beta 7.01. This tool offers spell checking and parenthesis checking (for simple brackets like (), [] and { }). It also shows commented text, maths mode text and verbatim mode text in different colours. The typechecker used was Z Type Checker (ZTC) Jia (2002). The Z specifications were typed into a \LaTeX file, which included the `ztc.sty` and the `oz.sty` (object Z) style files using the command

```
\documentstyle[11pt,oz,ztc]{article}
```

The ZTC style file `ztc.sty` was downloaded from the internet (MSU, 2004) and was stored amongst the TeX files in the path `MiKTeX\tex\latex\ztc` (the object Z style file was already included in the MiKTeX installation). The Z type checker checks that function inputs and outputs are of the defined types; it also checks for syntactical errors. The outputs of the Z type checker can be seen in Appendix D, section D.8 and Appendix E, section E.11. The command used to obtain these files was

```
ztc -It filename.tex
```

where *filename.tex* is the name of the file containing the Z specs.

3.4 Conclusion

The methods and techniques for specifying the forest harvest scheduling system (and its domains) using semi-formal methods was described in section 3.2. The Zachman framework of describing different aspects of the system was used to structure the semi-formal specification, which can be found in Chapter 4. The methods and techniques for specifying the forest harvest scheduling system (and its domains) using formal methods was described in section 3.3. The formal specification which was created using these techniques and methods, using the Z notation, is introduced in Chapter 5 and is given in Appendices D and E. An index of schema names which have been defined in these two appendices can be found on page 431.

Chapter 4

Semi-formal specification

4.1 Introduction

The aim of this dissertation is to present a specification of a forest harvest scheduling system which includes wood properties in the harvesting decision, using semi-formal and formal methods. In this chapter, the results of the semi-formal analyses are presented.

When specifying a system, it is necessary to describe the systems and procedures that currently exist as well as the system which is to be built (Jackson and Zave, 1993). What currently exists is captured in sections 4.2 to 4.6. These are the domains (or, environments) of the forest harvest scheduling system. The forest-to-mill domain is described first in section 4.2. This is an overview description of the forestry, transport and mill domains, which are described in more detail in sections 4.3, 4.4 and 4.5 respectively. Finally, the forest planning domain is described in section 4.6.

The system, which is embedded in these domains, is specified in section 4.7. This section includes a description of the system's aims (section 4.7.1), the system's context, which shows all the systems, databases and people who interact with the system (section 4.7.2), and the system's inputs and outputs (section 4.7.3). A typical run of the

system is then described (section 4.7.4). Central to the system are the wood properties which influence the harvesting decisions. Modelling, predicting and measuring them is discussed next (section 4.7.5). Finally, implementing the system would impact on the domains described in sections 4.2 to 4.6. These impacts are discussed briefly in section 4.7.6 and described in more detail using semi-formal notation in Appendix C.

The Zachman framework at the Business owner’s level was used to structure the analyses presented in this chapter. Table 4.1 gives a recapitulation of the Business owner’s row of the framework. (Table 3.1 on page 50 gives the full Zachman framework table.) Table 4.2 summarises the aspects of the domain and the system which are modelled in this chapter. Empty cells in this table do not mean that a particular aspect has not been modelled; rather, it means that it is not necessary to repeat a model which has been presented elsewhere in this chapter, as there are many overlaps.

The analyses presented in this chapter and in Chapter 5 describe the activities of an integrated plantation forestry company, where the plantation and mill are owned by the same organisation. The tables of actions and constraints presented in sections 4.3, 4.4, 4.5 and 4.6 are based on Price *et al.* (2009).

Table 4.1: Business owner’s row in the Zachman framework (from Hay (2003, p.3))

	Data (what?)	Activities (how?)	Locations (where?)	People (who?)	Time (when?)	Motivation (why?)
Enterprise model (Business owner’s view)	Language, divergent data model	Business process model	Logistics network model	Organization chart	State/ transition diagram	Business strategies, tactics, policies, rules

4.2 Forest-to-mill domain

The forest-to-mill domain (or forest-to-mill supply chain) is characterised as a source (forest) which eventually leads to a sink (mill, or forest products processor). Figure 4.1

Table 4.2: Analyses undertaken from the Business owner point of view of the Zachman framework

	Data (what?)	Activities (how?)	Locations (where?)	People (who?)	Time (when?)	Motivation (why?)
Forest-to-mill domain	✓			(✓)	✓	✓
Plantation forestry domain	✓	✓	✓	✓	✓	✓
Transport domain	✓				(✓)	✓
Mill domain	✓				(✓)	✓
Forest planning domain	✓	✓		(✓)	✓	✓
Forest harvest scheduling system	(✓)	✓		(✓)	(✓)	✓

shows this in the most abstract form. Figure 4.2 expands on this: forest stands are grouped into estates. The stands are planted with trees, which are eventually felled (or harvested) and extracted to the stand's roadside. At roadside, the tree-lengths are sometimes cut into logs and put into piles awaiting short-haul transport to a depot or siding. There is a depot and/or siding in or near each forestry estate. It should be noted that tree-lengths are not always cut into logs at roadside; logs are sometimes also made at the depot or siding, or at the mill. The timber is taken to the depot if it is to be transported long-haul to the mill by road; it will be taken to the siding if it is to be transported by rail. (There could be other methods of long-haul transport, not mentioned explicitly here; it is normal for the short-haul trip to be made by road.) At the depot or siding, the logs are put into piles according to their type¹ and the destination mill.

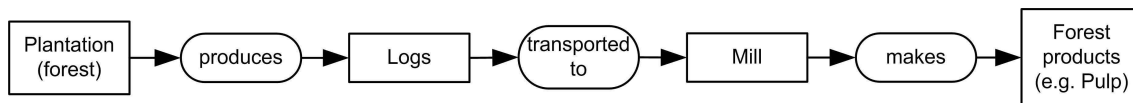


Figure 4.1: ERD giving an overview of the forest-to-mill supply chain

Timber is transported to the mill long-haul via road or rail. Timber could also be sent to the mill from another timber grower. At the mill, the timber is added to the mill's logyard. From the logyard, the logs are fed into a process and a pulp product is made.

¹In this description which concentrates on pulp manufacture, there is only one log type (pulp log), but stands which were grown for sawtimber or veneer have many log type categories per stand.

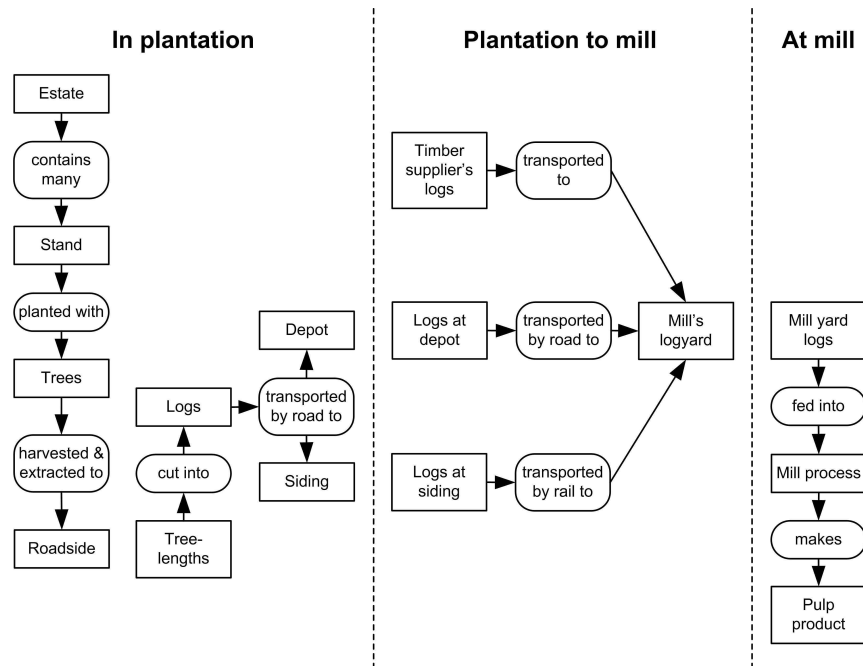


Figure 4.2: ERD giving more detail of the forest-to-mill supply chain

(The transport domain is described in more detail in section 4.4. The mill, and its processes, is described in more detail in section 4.5.)

Figure 4.3 uses a state chart to describe the forest-to-mill supply chain. This shows the action which triggers a change from one state to another in the supply chain. In the forestry part of the chain, planting changes an *unplanted* stand to a *planted* one. Harvesting returns the stand's state to *unplanted*. Harvesting is also the action which causes there to be logs (or timber) at the stand's roadside. Loading the short-haul transport truck and then transporting and unloading them at the depot will change the logs' state from being *at roadside* to *on the truck*, and then from *on the truck* to *at the depot*. There is a similar situation for long-haul transport. Logs can arrive at the mill and be added to the logyard via long-haul transport, or from a timber supplier (external to the integrated forestry company). From the logyard, the logs are processed.

Each part of the forest-to-mill supply chain incurs costs. The financial flows for the

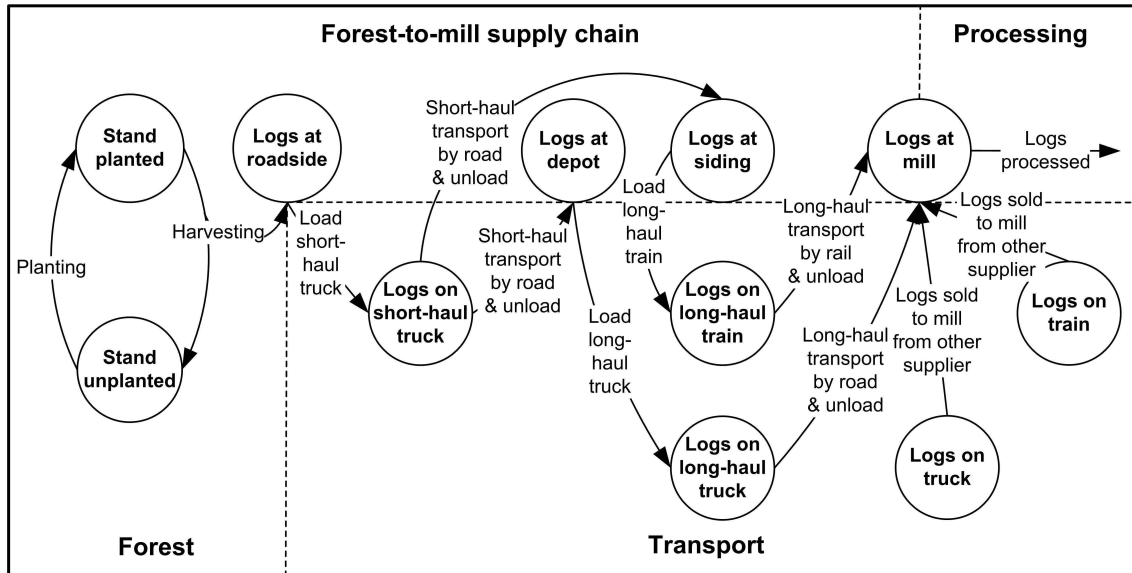


Figure 4.3: State chart giving an overview of the forest-to-mill supply chain

forestry, transport and mill parts of the supply chain are described in their respective domain descriptions (see sections 4.3.6, 4.4.3 and 4.5.3). These costs will be taken into account in the forest harvest scheduling system, as the objective function maximises profit (see section 4.7.1 for more detail).

4.3 Plantation forestry domain

As activities which occur in plantation forestry affect the forest harvest scheduling system, the plantation forestry domain needs to be described. This will make the constraints and procedures which need to be followed by the system more explicit. This section covers several aspects of the plantation forestry domain. Firstly, the actions and constraints for the plantation forestry domain are summarised (section 4.3.1). Next, the spatial nature of plantation forestry is described in section 4.3.2: this covers how the company's plantations are spread over vast areas, and how these are broken up to aid management (from regions to stands). It also describes the ownership of the

land on which the trees are planted. The third aspect is a description of the genetic makeup (genus, species etc.) of the trees to be planted in the stands (section 4.3.3). The fourth aspect, covered in section 4.3.4, is a description of the regimes used to govern what happens in the various stages of a stand's growth. Stand forestry and the stand's lifecycle are described in section 4.3.5. The financial costs and income of the forestry part of the integrated forestry company are discussed in section 4.3.6. Lastly, the roles and responsibilities of the forestry and logistics staff who are involved in the managing and planning activities in the plantation forest are given in section 4.3.7.

4.3.1 Summary of actions and constraints for the forestry domain

Table 4.3: Actions and constraints for the forester (grower)

Actions	<ul style="list-style-type: none"> • Order seedlings/cuttings from the nursery • Plant • Maintain stands/estates • Perform other silvicultural activities • Ensure that roads are upgraded prior to harvesting stands • Harvest trees in stand & extract to roadside • (Make logs from tree-lengths) • (Debark logs) • Record actions taken in plantation database
Constraints	<ul style="list-style-type: none"> • Afforestable/afforested area is widely spread • Many different species of trees could be planted • Use regime to guide forestry actions for a stand • Trees' growth rates differ, depending on where planted and species <p>Planting:</p> <ul style="list-style-type: none"> • Should plant as soon after harvesting as possible • May have to wait for rainy season to commence planting • Not all species can be planted everywhere (e.g. some are frost sensitive) • Should only plant a species suitable for the stand • Should plant tree species which are acceptable to mills • Sufficient stock of suitable seedlings or cuttings may not be available <p>Harvesting:</p> <ul style="list-style-type: none"> • Aim to harvest at rotation age, but could harvest earlier or later (within limits) • Harvesting rate varies, depending on the slope of the land and the terrain • Different harvesting equipment is needed, depending on the slope of the land, the terrain and prevailing weather • Cannot harvest stands whose soils are compaction-sensitive in wet weather • Roads leading from stand to be harvested must be upgraded before harvesting commences

Table 4.3 shows the actions and constraints which apply to the plantation forestry domain. The actions taken by the foresters, and the constraints placed on these actions, are described to a large extent in section 4.3.5, but various aspects mentioned in this table are also covered in sections 4.3.2, 4.3.3 and 4.3.4. Actions which may or may not occur are shown in parentheses. For example, logs may be made by the foresters, or tree-lengths may be cut up at the depot/siding, at at the mill's logyard. Some logs of genera such as eucalypts and wattle have their bark stripped in the field before the logs are transported to the depot or siding.

4.3.2 Spatial layout of plantation forests and mills

When establishing a pulp mill, much wood is needed to produce the large quantities of pulp which are required to ensure that the unit production costs are acceptable (Evans and Turnbull, 2004, p.74). This means that large tracts of land are needed for planting the trees. Ideally, trees would be planted on land which is close to the mill (to reduce transport costs) and not too far apart from each other (Evans and Turnbull, 2004, p.60) (to reduce management and harvesting costs). The land accessible to the plantation forestry company for afforestation is subdivided into smaller area units for ease of management. There may be several levels in this management hierarchy (Weintraub and Davis, 1996; Louw, 2000b). The smallest unit of management is the stand (or compartment). A group of stands forms an estate (or 'Forest block' (Evans and Turnbull, 2004, p.45)).

Stands are bounded by roads (Herbert, 2000, p.32), tracks or fire-breaks (Evans and Turnbull, 2004, p.45). Although roads are sometimes only developed just before harvesting (Savill *et al.*, 1997, p.45; Epstein *et al.*, 1999a), roads of a basic quality are often built at the time of planting, to enable access for workers and materials (Savill *et al.*, 1997, p.45). If the layout of an estate needs to change (e.g. roads built elsewhere, stands merged or split, or stands becoming conservation areas) this will most

likely happen after the stand's trees have been harvested (Savill *et al.*, 1997, p.45).

Within the boundaries of the estate, natural forest is often conserved (Evans and Turnbull, 2004, p.10). In South Africa, it is estimated that only 60–65% of land owned by plantation forestry companies is used to grow trees (Everard, 2000), the remaining land being retained as conservation areas.

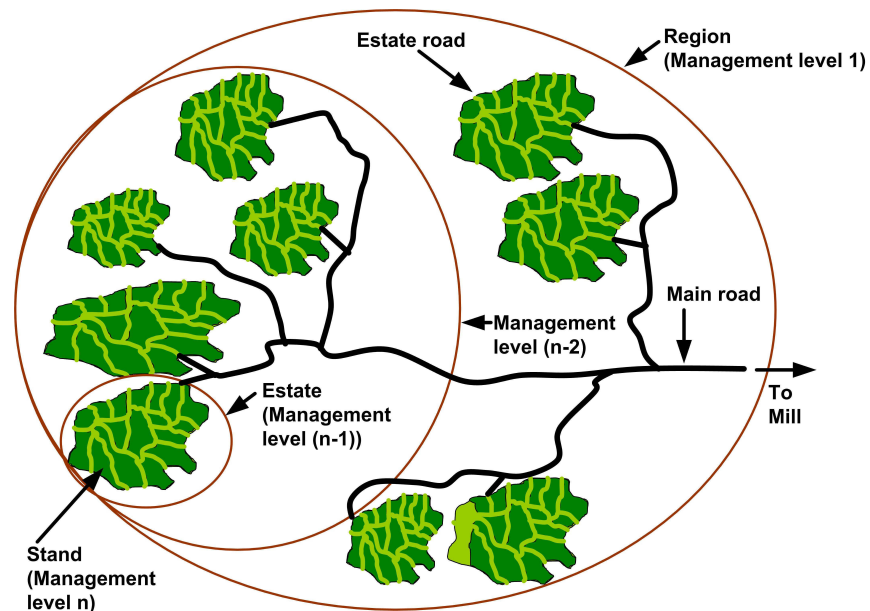


Figure 4.4: Spatial layout of a plantation forest, showing hierarchy of management levels

Figure 4.4 shows a conceptual representation of the spatial layout of a plantation forest. There are many estates in the forest, and a grouping of estates makes up a higher management level (the names for these management levels vary from company to company). This figure also shows the roads in the estates (for short-haul transport), and the main road leading from the estates to the mill (for long-haul transport by road).

Figure 4.5 shows the plantation forestry management levels in organogram representation. The forestry company is subdivided into regions, which are in turn subdivided

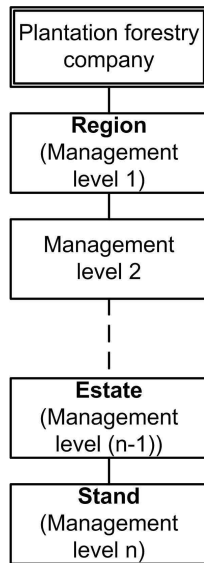


Figure 4.5: Hierarchy of forestry company management levels

further and managed separately. The number of management levels in this hierarchy depends on forestry company. The second last management level (i.e. the $(n - 1)$ st level) is the estate, which contains many stands (the n th management level). The lines between Management level 2 and the estate (Management level $(n - 1)$) is dotted so that each forestry company could add management levels at this point of the diagram if necessary. The organisational structure of the company, and the names of the management levels of the plantation forest may change from company to company.

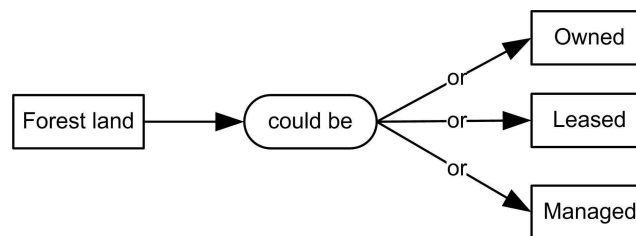


Figure 4.6: ERD giving forest land ownership options

Land does not have to be owned by the integrated forestry company in order for trees to be grown on it. The land may be leased from another owner or managed by the

plantation forestry company on behalf of the owner (see Figure 4.6). This has the implication that the stand's land may not be available for afforestation for the whole strategic planning horizon.

4.3.3 Genetic material (“species”)

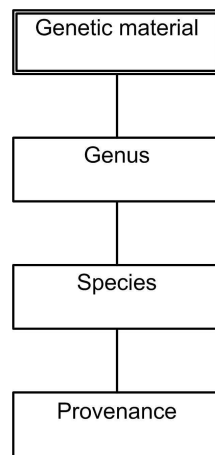


Figure 4.7: Hierarchy of genetic material (loosely called “species”)

The trees which are planted in the stand can be identified by their genetic material, which can be described as a hierarchy. A genus comprises of one or more species, and the genus and species are mentioned together (e.g. *Eucalyptus grandis*, *Pinus patula*, *Acacia mearnsii*). In some instances, a species is further subdivided into a provenance (e.g. *E. nitens* (Ebor) and *E. nitens* (Tallaganda)). Figure 4.7 shows the hierarchy of the tree's genetic material.

In forestry nurseries, the trees are propagated either by seed (in which case they are called seedlings) or by cutting (in which case they are called clones). When pollinating trees (to make seeds), the pollen of two different species could be used, in which case the resultant seed will be a hybrid of the two species (e.g. *Eucalyptus grandis* x *nitens*), or the pollen of the same species could be used, in which case the resultant seed will

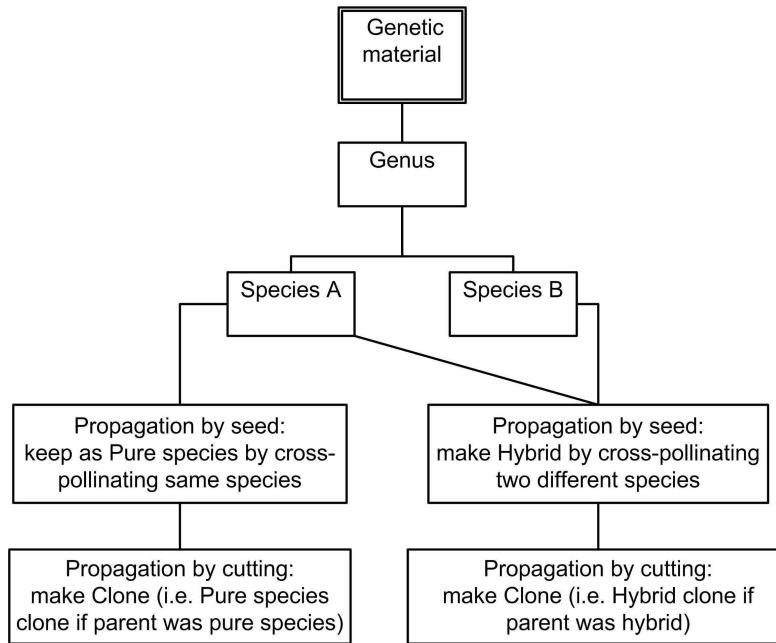


Figure 4.8: Diagram showing the relationship between pure species and hybrids, seedlings, cuttings and clones

be pure species. Cuttings will produce clones of the tree from which the cuttings were made, i.e. a hybrid clone if the tree was a hybrid of two different species, or a pure species clone. Figure 4.8 shows the relationships between pure species and hybrids, seedlings, cuttings and clones.

4.3.4 Regimes

Stands of trees are managed using regimes (Shepherd, 1986, p.265). A regime is planned before the stand is planted. It specifies the actions which should be carried out by year. In Table 4.4, a regime which could be applied to any stand of *Eucalyptus* trees in Region 4 is outlined. (Regimes could be applied to any management level, and to any level in the genetic material hierarchy.)

In year 0, the trees are planted to the specified planting density. Shortly after planting,

Table 4.4: Typical planned regime for growing trees for pulp

Year	Regime action	Details	Stage of growth of stand
0	Planting	Planting density: 2.7m x 2.7m	Establishment
		Species: <i>Eucalyptus</i>	
		Applied to: Region 4	
	Blanking		
	Weeding (Fertilizing?) Maintenance		
1	Weeding ^a (Fertilizing?) ^a Maintenance ^a		
	...		
3	Maintenance		Canopy closure
4	Maintenance ^b		
	...		
6	Enumerate stand		Just prior to felling
7	Fell stand		Rotation age

^a These three actions are repeated annually until canopy closure has been achieved.

^b This action is repeated annually until the trees have been harvested.

the stand is checked to see if there were mortalities; if so, the dead seedlings or cuttings are replaced Zwolinski and Hinze (2000). This is called blanking. In spite of the months' age difference, these trees are still considered to be the same age (Evans and Turnbull, 2004, p.41). Planting and blanking are categorised as establishment activities.

Depending on the soil type, fertilizer may be applied. While the trees are young, weeding occurs to remove plants that may compete for the trees' nutrients, water and sunlight. As the trees grow and their branches form a canopy, it is harder for weeds to survive in the understory, so weeding is no longer necessary. The stand and estate is maintained annually. Maintenance activities include checking for diseases, pests and weather event damage (e.g. wind or hail storms) and building fire breaks. About a year before the stand is due to be felled, the stand is enumerated to obtain a more accurate picture of the growth of the stand, so that the stand's volume can be predicted more accurately. The trees' height and diameter at breast height (DBH), as well as the stand density are measured. The stand is to be felled at rotation age. This is the "ideal" age for such a stand of trees to be felled, but they could be felled a few years before (minimum

harvestable age) or a few years after this age (maximum harvestable age). This is discussed in more detail in section 4.6.2; Figure 4.16 on page 85 shows the minimum and maximum harvesting ages, as well as the rotation age.

A variation on the regime presented in Table 4.4 is one where the trees in the stand are allowed to coppice after clearfelling (Evans and Turnbull, 2004, p.292). The practice of coppicing only applies to genera and species which sprout shoots from the tree's stump after harvesting, and stands for which coppicing is compatible with the intended end use (e.g. pulpwood, poles, firewood). It is practiced because it saves establishment costs. Instead of planting and blanking, coppice reduction occurs over the first few years of the stand's regrowth: this is when the less vigorous shoots are removed, leaving up to three stronger shoots to grow to maturity.

When regimes are implemented, the action taken, and the date it was taken are recorded in the plantation database by the Managing forester. Recording activities undertaken in the plantation is important, because things do not always go according to plan, the Managing forester who was in charge may no longer be there (or remember exactly what transpired), and the record needs to be available for planning the future of the stand and the forest. In the record of the actual regime, the date that the action was completed is recorded. The planned, and if available, the actual regime information is used when predicting the volume of the stand's timber (for more on stand volume prediction, see section 4.6.2).

Some regime actions have a date attribute attached which means that that action needs to be carried out on that date. This "user-enforced" date is a useful attribute to invoke when things do not go according to plan (e.g. a fire swept through the stand, and the trees need to be harvested in the next two months or they will be subject to beetle attack and susceptible to strong winds). It is also useful when the land on which the tree stand is grown is not owned by the forestry company, and certain conditions need

to be met before returning the land to its owners (e.g. the leased land must be returned with a normalised set of stands on it).

Section 4.3.5 contains diagrams which elaborate on the use of regimes in stand forestry: Figure 4.11 (page 79) gives a state transition chart which shows the various stages of a regime for a stand. Figures 4.12 to 4.15 (pages 80 to 4.15) present a more detailed insight into the use of regimes in managing a stand of trees using Business Process Modelling Notation.

4.3.5 Stand forestry

The smallest unit of management in a plantation forest is the stand (Louw, 2000b). Before a stand of trees is planted, the end use for the trees (e.g. pulpwood, saw timber, poles, etc.) has usually been determined (Evans and Turnbull, 2004, p.107). This choice narrows down the list of possible species which can be grown (some may not be suitable for all end uses, or some species may not be acceptable to some mills). As some species may grow well on one particular stand and not on another, the stand's properties must be inspected to determine the most suitable species to match that stand.

A tree of a particular species would grow well on a particular stand, depending on the tree's genetic material, the stand's soil type, position (altitude, latitude, longitude, slope, aspect) and prevailing weather conditions (mean annual temperature, mean annual precipitation, incidence of frost, day length and solar radiation). Figure 4.9 outlines these dependencies.

Once the end purpose for the trees and the tree species have been chosen, an appropriate regime needs to be chosen. The regime's actions could be adjusted, according to the end purpose for which the trees are to grown, the species chosen and the soil,

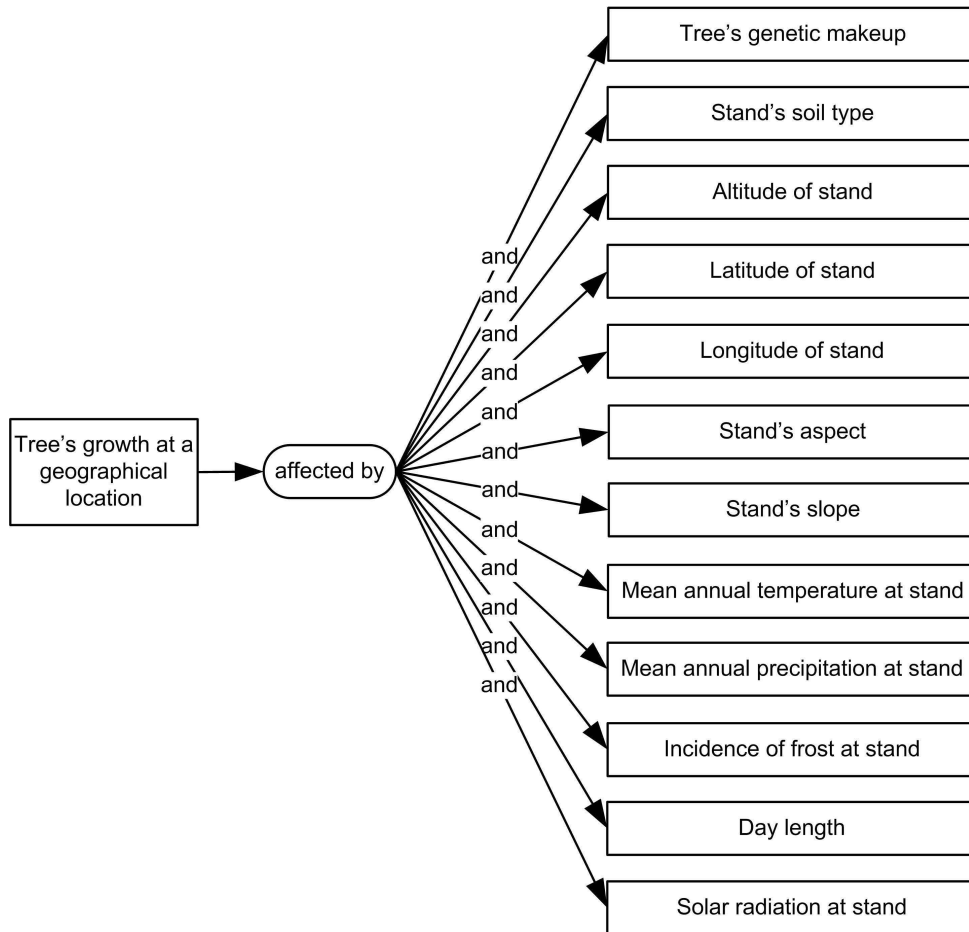


Figure 4.9: ERD showing the factors on which a tree's growth on a particular stand depends

position and weather conditions of the stand. (A regime for an end use and species may be applied to a particular management level of the plantation). Figure 4.10 shows the characteristics of a monoculture plantation forestry stand.

Activities for a new stand of trees begin either after the previous crop has been harvested (clearfelled), or (in the case of a first afforestation) as soon as the land has been prepared. In both instances, soil preparation occurs (see Figure 4.11). Soil preparation and planting may be delayed until the rainy season, if there are distinct rainy and dry seasons (Shepherd, 1986, p.212; Evans and Turnbull, 2004, p.213); however, the aim is

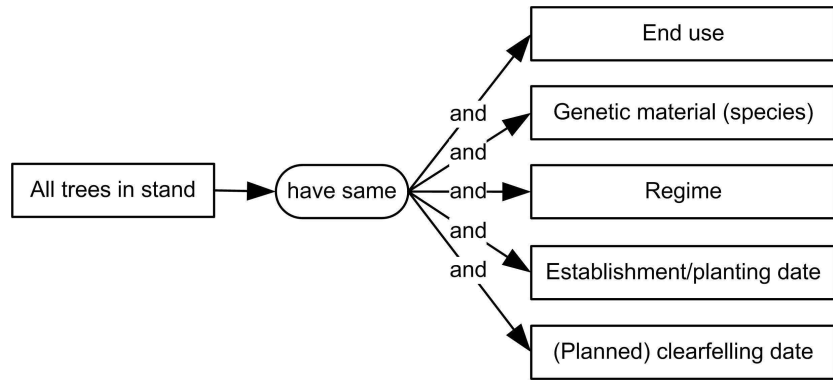


Figure 4.10: ERD of the characteristics of a monoculture plantation forestry stand

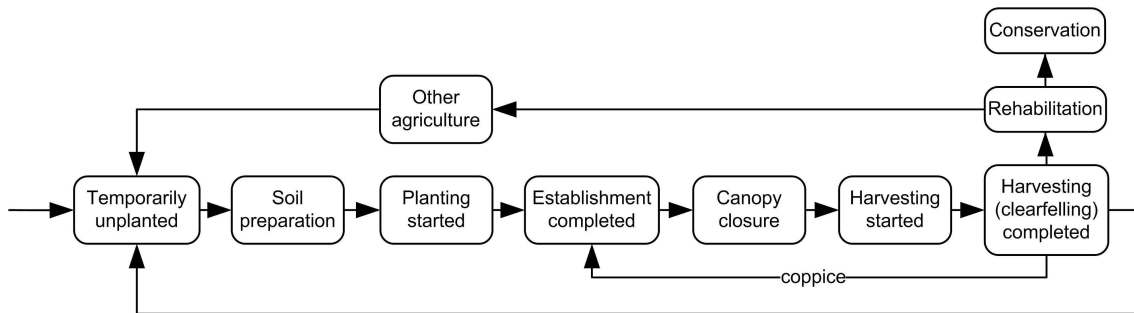


Figure 4.11: State transition chart showing the possible states of a stand

to replant as soon after clearfelling as possible. Once planting has started, the aim is to complete it as soon as is possible. Before the stand is due to be felled, the stand's trees are measured and the roads leading from the stand are upgraded to be able to withstand the heavy logging trucks (Evans and Turnbull, 2004, p.204). As with planting, harvesting is completed as soon after it is started as possible. The action triggers in Figure 4.11 have not been put in this diagram explicitly. The trigger which causes the stand to change state is the fact that the time has come to implement the next regime action.

Once all the trees have been harvested, the land could become rehabilitated and used for conservation or other agricultural uses; if it is to be replanted with more trees, it becomes temporarily unplanted. An exception is if the regime dictates that the tree stumps should be left to coppice. In this case, the establishment phase of the regime is

skipped, but the other regime actions (e.g. weeding, maintenance) are still undertaken. Usually, foresters wait until the trees are harvested until they change the land-use or boundaries of a stand.

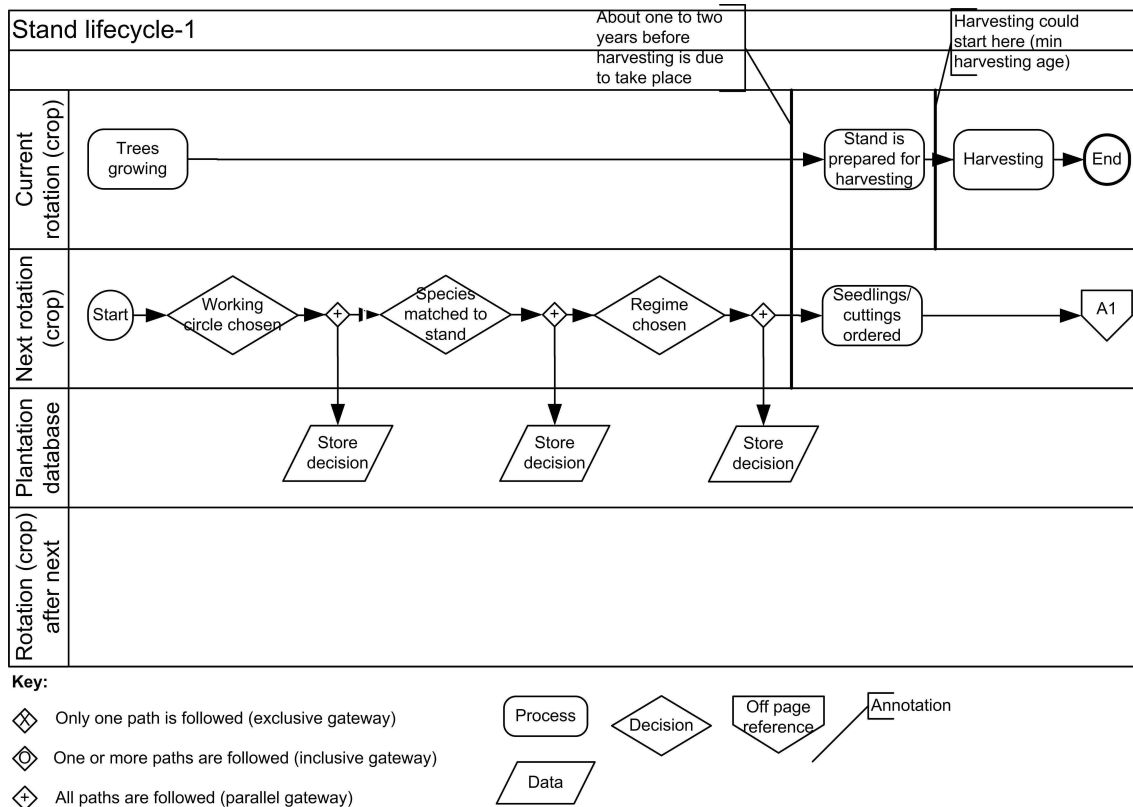


Figure 4.12: Business process diagram of the stand's lifecycle (part 1 of 4)

The stand's lifecycle is presented using Business Process Modelling Notation in Figures 4.12 to 4.15. Different swimming lanes are shown for the current rotation (or crop of trees), the plantation database (in which the details of the stand's history and plans are stored), and the next rotation. Although the stand's lifecycle starts physically with soil preparation and planting (see top row of Figure 4.13 on page 82), the decision about what to plant is taken long before that, (see second row of Figure 4.12), while the previous crop is still in the ground. First, the working circle (end use of timber grown) for the stand is decided upon. This decision is made bearing in mind the forestry company's strategy and the mills for which it is intending to grow trees. Working circles

determine the types of logs that will be produced (e.g. pulp logs, saw logs). (A saw timber working circle has saw logs as its main product, and pulp logs a side product.) The working circle decision for the next rotation is stored in the plantation database. Next, the best genus and species for the stand are determined. Three possible species are chosen for the stand using the following rules:

- rule out any genus/species which is not suited to the stand's geographic area, altitude, rainfall, aspect or slope
- rule out a particular species for this working circle if:
 - (nearby) mills accepting logs from this working circle do not want that species (Shepherd, 1986, p.201; Evans and Turnbull, 2004, p.107; Savill *et al.*, 1997, p.77)
 - (nearby) mills will accept logs of that working circle and species, but only in limited amounts, and a sufficient supply of that age-class has already been planted, or is planned to be planted
- choose the best remaining species for the stand according to the stand's soil type, altitude, rainfall, etc.

The best three possible species are stored in the plantation database. Next, the regime for the stand's next rotation is chosen and recorded in the plantation database. Between 18 to 24 months before the stand is to be planted, seedlings or cuttings for the best genus and species are ordered from the nursery. Note that in the first diagram (Figure 4.12), the detail of the "current rotation" is not given. It is described in detail in Figure 4.14 (page 83).

The stand's physical life cycle starts with soil preparation (when the stand is in its temporarily unplanted state) (see top row of Figure 4.13). This may entail removing old tree stumps, harrowing and fertilizing. The stand is planted as soon as possible after

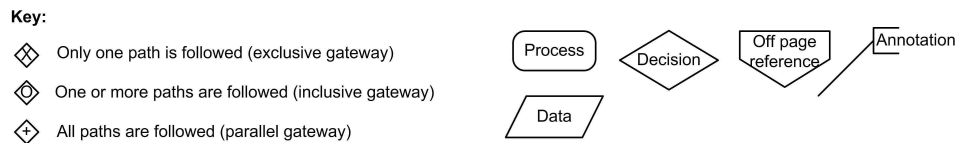
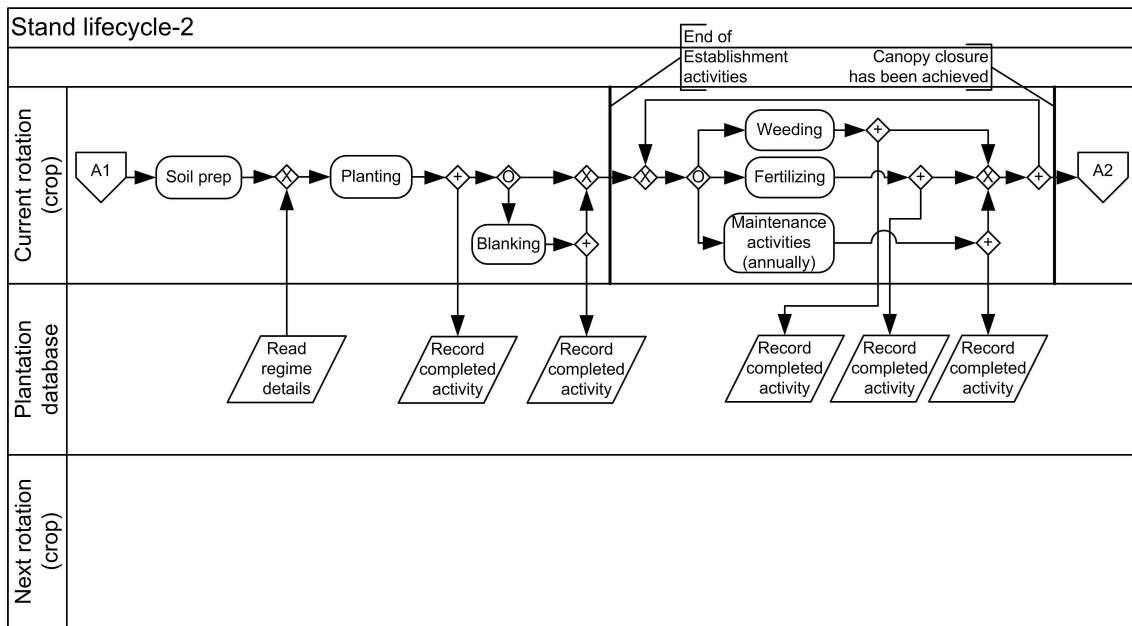


Figure 4.13: Business process diagram of the stand's lifecycle (part 2 of 4)

harvesting because the faster the next crop is planted, the faster it can be harvested. Also, the sooner the trees are established, the sooner canopy closure will occur, which means that weeding within the stand will no longer be required. However, planting may need to be delayed until the start of the rainy season. Planting will occur with the genus and species planned and recorded in the plantation database. The growing stock is planted according to the planting density (stems per hectare) prescribed in the regime which is stored in the plantation database. An exception is that at the time of planting, if there was a shortage of planting stock of the most suitable species, the second or third best choice may be planted.

A short while after planting, the stand will be blanked. Note that if the stand is under a coppice regime, after harvesting a few (one, two or three) times, planting and blanking will not occur, but the stumps will be allowed to produce shoots. A coppicing regime

has not been shown in these business process diagrams.

Figure 4.13 also shows the part of the regime until canopy closure is achieved. During this time, weeding and fertilizing may occur, and maintenance activities (such as making fire breaks, checking for pests and disease, etc.) will occur annually. Activities undertaken in the stand are recorded in the plantation database.

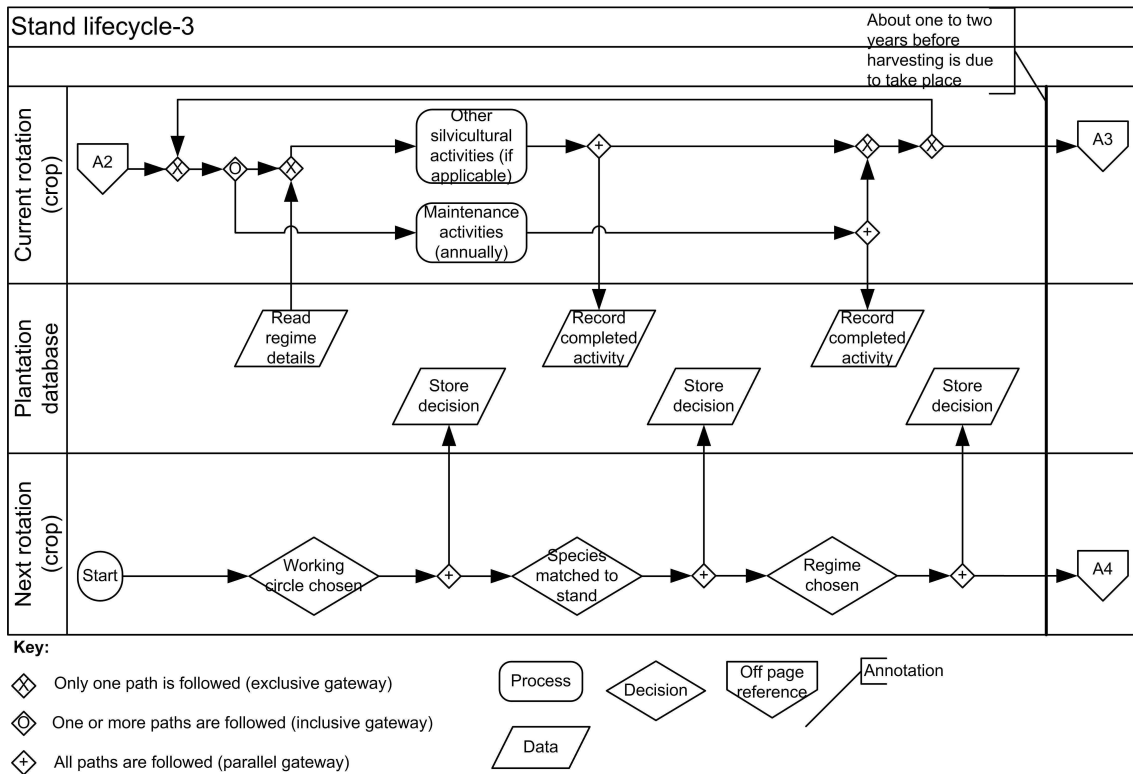


Figure 4.14: Business process diagram of the stand's lifecycle (part 3 of 4)

After canopy closure has been achieved, activities in the stand decrease (see Figure 4.14). In the case of sawtimber regimes, further silvicultural activities (like thinning and pruning) will occur. Maintenance activities will continue annually. The time after canopy closure and before preparations to harvest is normally the longest period of the stand's life.

Between one and two years before the stand is due to be harvested, preparations are made for harvesting and replanting (see Figure 4.15 on page 84). The stand is enumerated: the diameter at breast height (DBH) and tree height of the trees in the stand, and the stocking density are measured. The measurements are recorded in the plantation database. Using this data, the stand's expected timber volume will be able to be predicted more accurately. Roads leading from the stand are upgraded, so that the heavy logging trucks and harvesting equipment can access the stand (Shepherd, 1986, p.124). Seedlings or cuttings for the next rotation are ordered from the nursery.

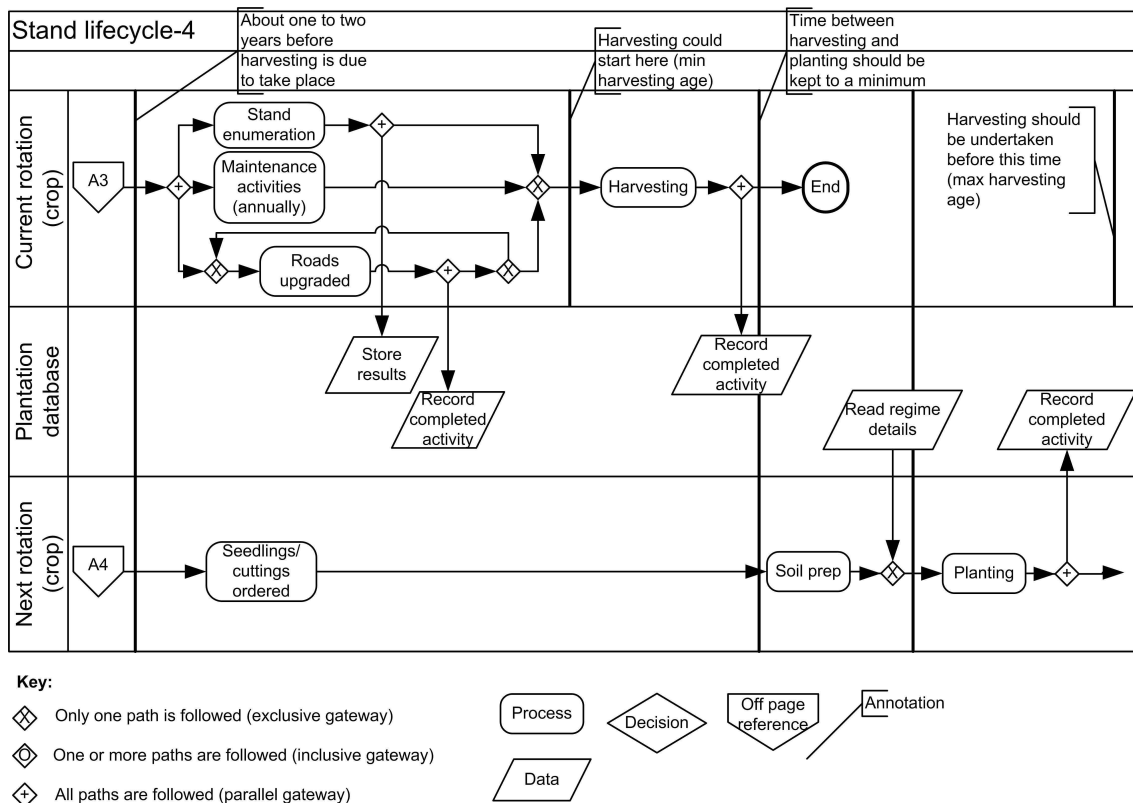


Figure 4.15: Business process diagram of the stand's lifecycle (part 4 of 4)

Although the regime specifies a rotation age (most desirable age at which to harvest the trees), there is in fact a minimum and a maximum age between which the trees could be harvested. Harvesting them before this age would not be desirable because the trees would be too small and the return on investment would not be great enough.

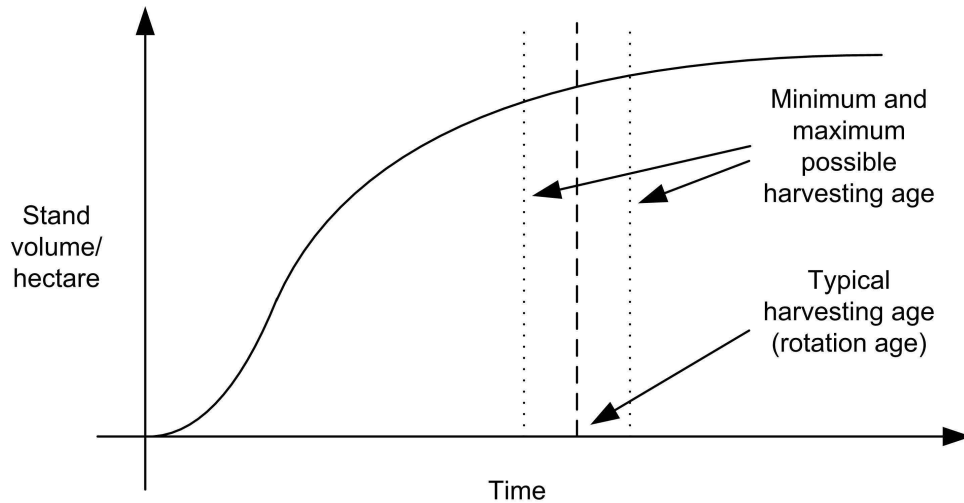


Figure 4.16: Graph showing the typical volume growth of a stand of trees over time. The growth rate declines as time increases.

Harvesting the trees after this age may mean that the trees are too large for the process for which they are destined; they may start becoming stressed due to competition with each other, and the return on investment starts to decline as the trees' growth rate declines (von Gadow and Bredenkamp, 1992, p.92) (see Figure 4.16 on page 85). (For more on predicting the volume of the trees in the stand, see section 4.6.2.) Although trees would normally be harvested between the minimum and maximum harvesting ages, circumstances could cause a stand to be older than the maximum harvesting age. In this case, these old stands would be scheduled to be harvested first.

Harvesting is a complex and costly activity. Once trees have been felled, they are extracted to roadside, where they are either cut up into logs and transported to the depot or siding, or transported to the depot/siding or the mill as tree-lengths. (A rare exception to this would be that if there had been severe damage in the stand (for example, caused by fire or disease) the trees may be felled and left in the stand to rot.) The type of equipment used to harvest and extract the trees depends on the stand's soil type, the weather, the slope and the terrain. If the stand's soils are susceptible to compaction, harvesting cannot take place in wet weather, as this will negatively impact

on the growth of the stand's next crop. For some species, harvesting is preferable in the rainy season as it is easier to remove the bark (e.g. for eucalypt or wattle pulp logs). Pulp logs are cut into 2.4m lengths, but other end uses (like sawmilling) require a more complicated set of rules for cutting the tree into logs (i.e. bucking or merchandising the tree) in order to get the most value out of it. Factors affecting harvesting, and the decision about which stand to harvest, are discussed in more detail in section 4.6.

The four business process diagrams (Figures 4.12 to 4.15 on pages 80 to 84) present the normal order of events for a stand's lifecycle. If damage (caused by fire, hail, drought, pests, etc.) occurs, the stand would be assessed to see if the trees were still alive, and if they would reach rotation age. If not, the stand would be harvested and replanted as soon as possible. If every tree in a section of the stand has been damaged beyond recovery, these trees would be harvested, and the remaining trees would be grown until maturity. In this case, the plantation database would be annotated with a smaller stand area for the current rotation. If a stand's trees were all partially damaged, an estimation of the damage (as a percentage of the stand's volume) will be entered into the plantation database so that the final volume of logs produced by the stand can be reduced appropriately. Trees will be made into the highest possible valued log type and sold to a mill, if possible. Otherwise, the trees will be made into firewood.

In order to ensure good environmental and sustainable management, many forest plantation companies have adopted certifications schemes, such as Forest Stewardship Council (2003a), which aims to certify that "the forest resources and associated lands should be managed to meet the social, economic, ecological, cultural and spiritual needs of present and future generations" (Forest Stewardship Council, 2003b). Each activity described in the business process diagrams would have to be undertaken in such a way that the certification was maintained.

4.3.6 Forestry costs and income

In the plantation forest, costs can be categorised as forestry overhead costs, establishment costs, silvicultural costs, maintenance costs and harvesting costs (see Figure 4.17). Forestry overhead costs are made up of two components: the costs to support company-level structures and overhead costs at an estate level. The latter include costs for leasing land, buying land, upgrading roads prior to harvesting, and land conversion (e.g. if land which was afforested is to become a conservation area, or have another land-use). Establishment costs cover the stand preparation prior to planting and the planting and blanking processes. Silvicultural costs cover weeding, fertilizing and other silvicultural activities. The costs of making and maintaining fire breaks, and checking for pests and disease are covered by the maintenance costs. Finally, harvesting costs cover all the harvesting activities.

Establishment, silvicultural, maintenance and harvesting costs may be available as an average for a typical stand at a particular management level, or in a certain region, for a particular genus and working circle (end use). An example of such costs is given in Table 4.5 (page 87), which shows the average costs per annum which apply to a regime. (Note that the costs shown in this table are fictitious).

Table 4.5: Typical costs per annum for a pulp regime. These costs are fictitious.

Regime activity	Cost (R/ha)	Annual cost (R/ha)
Establishment	R13.00	
Silviculture (before canopy closure)		R 7.00
Silviculture (after canopy closure)		R 5.00
Maintenance		R 3.00
Harvesting	R30.00	

The income which the forestry part of the integrated forestry company could be derived in two ways. One way would be to decide a log “transfer” cost (in R/tonne), so that the forestry part of the company would have as income the tonnes delivered multiplied by that transfer cost. The other method would be to charge the mills proportionally

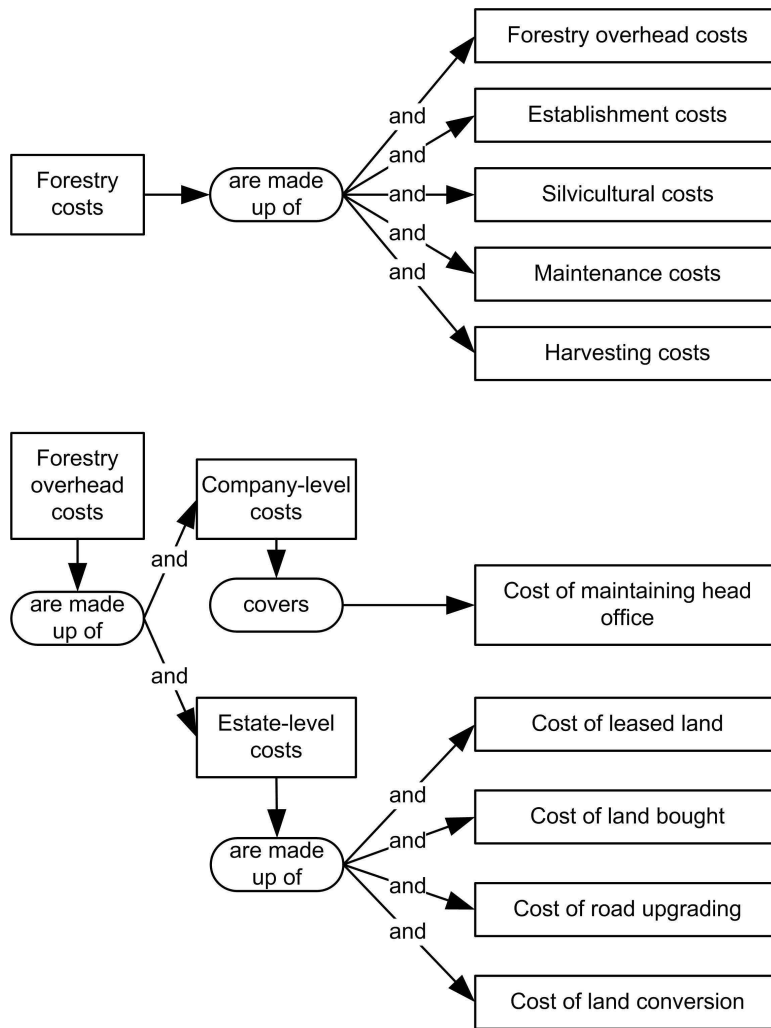


Figure 4.17: ERD showing costs involved in the forestry domain

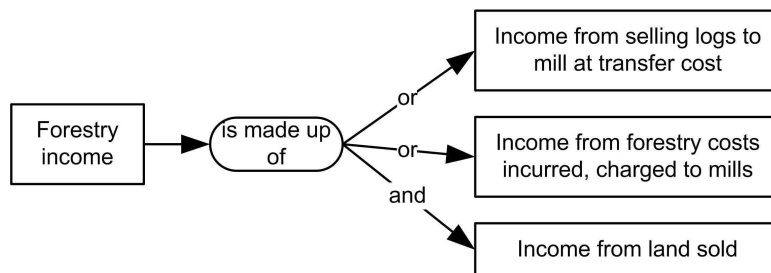


Figure 4.18: ERD showing income options for the forestry domain

the sum of actual forestry costs (see Figure 4.18). The first option is most common. In addition to this income is the value of any forestry land sold.

4.3.7 Forestry and logistics staff

Figure 4.19 shows the forestry organisation hierarchy, and which staff are responsible at different levels of this management hierarchy. As with the names of the forestry organisation management levels presented here and in Figure 4.5, the titles of the forestry company staff may change from company to company.

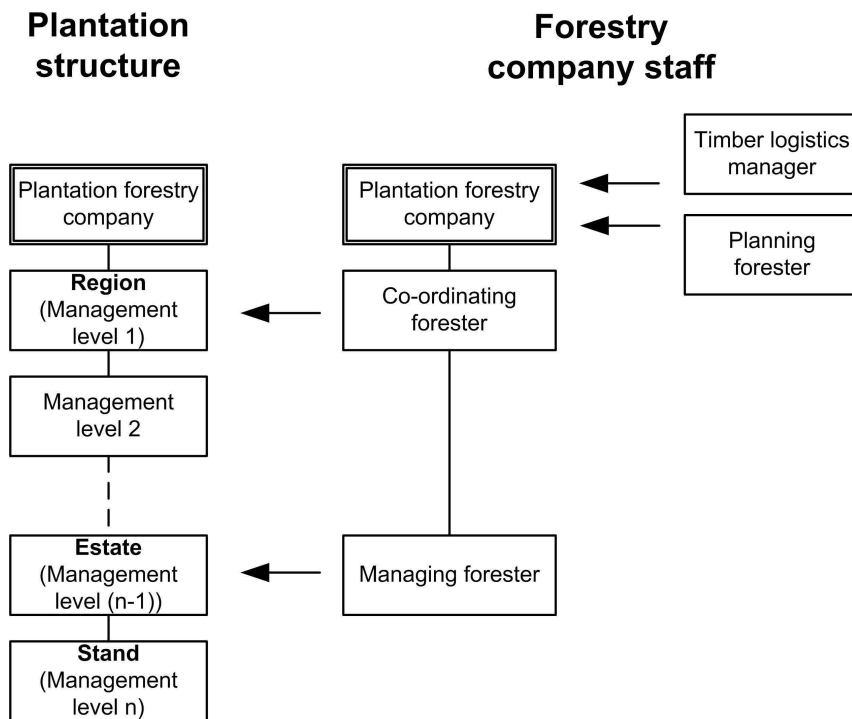


Figure 4.19: Organogram of forestry staff, showing area of responsibility

Figure 4.20 shows the responsibilities of the forestry company staff. The Managing forester is in charge of one or more estates, and has to implement the operational plan

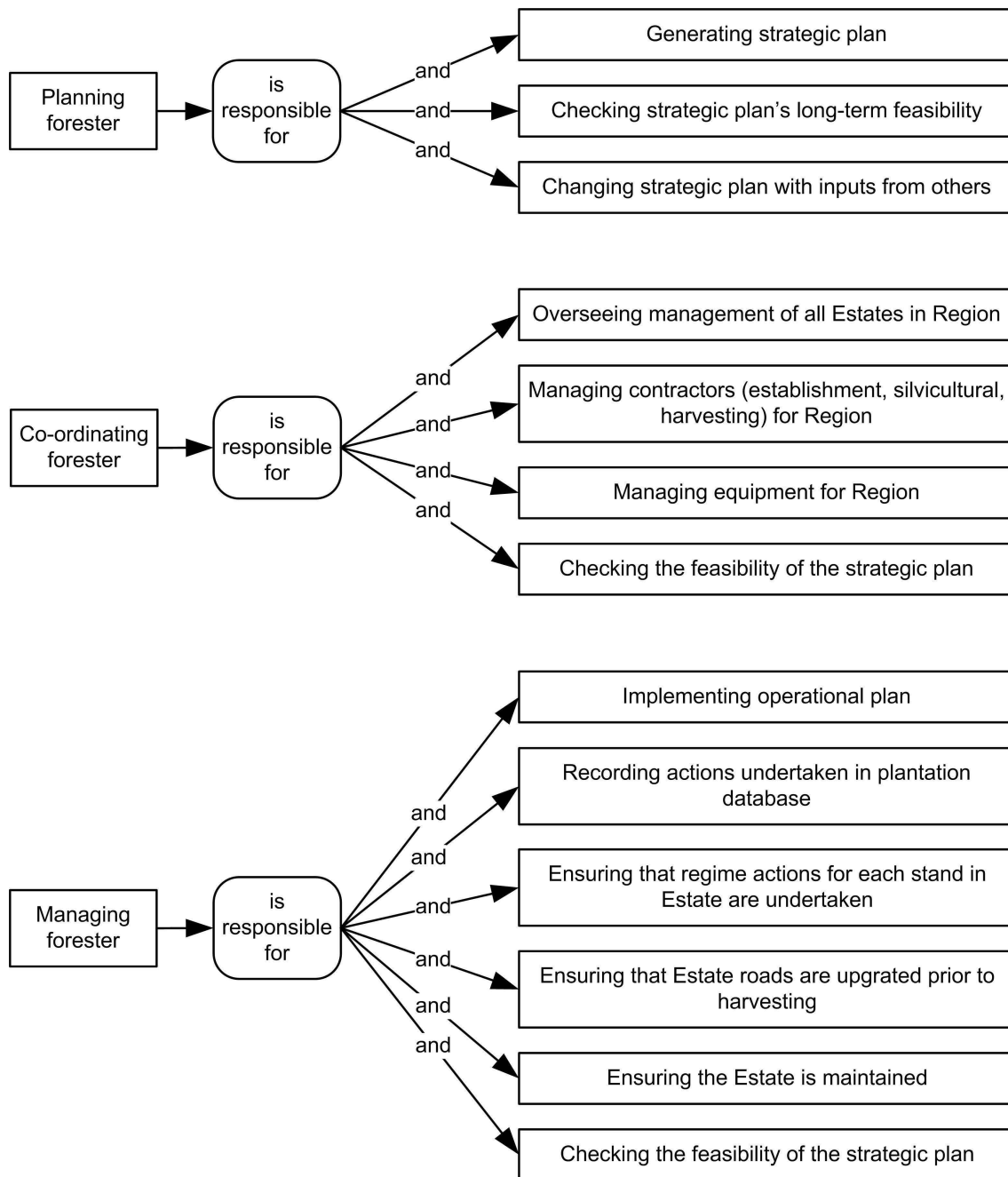


Figure 4.20: ERD showing responsibilities of forestry company staff

(i.e. every task and regime action for the estate(s) and stands in his² care). If the estates are managed according to certification standards, the Managing forester has to ensure that compliance with the certification standards is maintained. The Managing forester reports to the Co-ordinating forester, who is responsible for overseeing all the estates in the region. During the planning cycle, the Managing and Co-ordinating forester work together to assess the viability of the proposed strategic plan.

The Planning forester works at a company level, and is responsible for ensuring that the mills have a long-term supply of timber (i.e. trees are growing at about the same rate as they are being harvested). (Section 4.6 describes the planning process in more detail.)

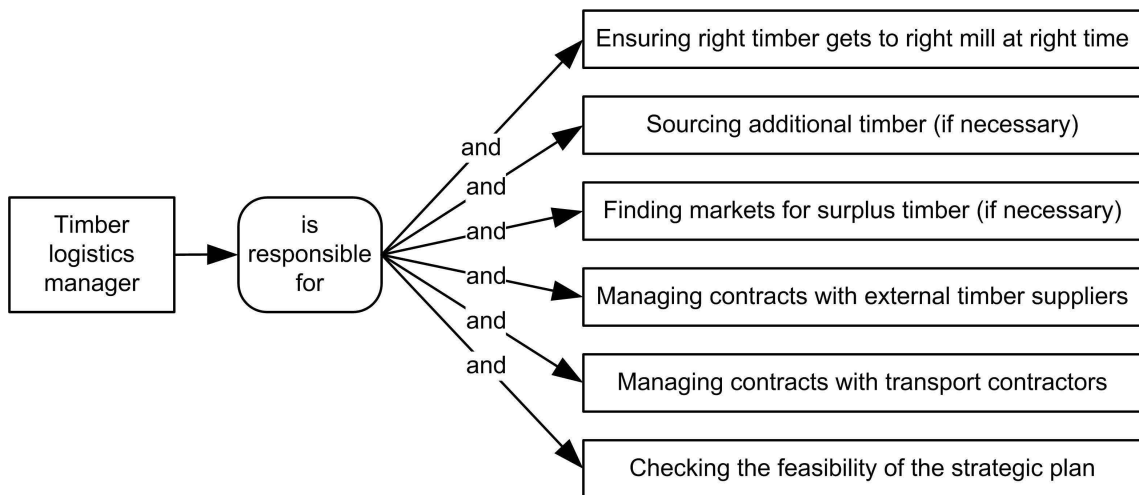


Figure 4.21: ERD showing responsibilities of the Timber logistics manager

The Timber logistics manager also works at a company level, and has to make sure that the mills have the timber they require. He is involved in logistics planning at the strategic, tactical and operational levels, and checking that the proposed strategic plan is feasible from the logistics point of view. At the operational level, he is responsible for ensuring that each mill gets the mass of logs, species mix, and log type required, and that there are enough logs with the correct certification status entering each mill.

²For 'his' please read 'his/her'. For 'he' please read 'he/she'.

If there is a timber shortage, timber has to be sourced from elsewhere. If there is an overproduction of timber from the plantations, the Timber logistics manager has to find alternative markets for the surplus timber. The Timber logistics manager is also responsible for managing contracts with timber suppliers who are external to the integrated forestry organisation, for managing contracts with transport contractors, and ensuring that the strategic plan is feasible from a logistics point of view (see Figure 4.21).

4.4 Transport domain

Transportation is the important link between the forests and the mills. The transport domain's actions and constraints are summarised in tabular form (see section 4.4.1). Section 4.4.2 outlines transport in the forest-to-mill domain. The costs incurred and income generated by the transport part of the supply chain are described in section 4.4.3.

4.4.1 Summary of actions and constraints for the transport domain

Table 4.6: Actions and constraints for the transporter

Actions	<ul style="list-style-type: none"> ● Load timber (logs/tree-lengths) at stand's roadside ● Transport timber to depot (short-haul) ● Unload timber at depot ● (Make logs from tree-lengths) ● Load timber at depot ● Transport timber to mill (long-haul) ● Unload timber at mill
Constraints	<ul style="list-style-type: none"> ● Want equipment available when ready to load/unload ● Paid for number of tonnes hauled over the haulage distance ● Each truck has a maximum tonnage it may carry ● Part-loads of timber left at roadside ● May decide to drive around the clock to maximise vehicle R.O.I. ● Number of hours a driver may work per day or per week is limited ● Sufficient transport of each transport type (long-haul or short-haul) and transport method (road, rail, ...) is available

The actions and constraints for the transportation part of the supply chain are summarised in Table 4.6. The transportation of timber makes up the largest part of the delivered timber cost. Timber is collected at the stand's roadside and taken to a depot (if long-haul transport is to be by road) or to a siding (for rail transport). At the depot or siding, the timber is stacked into piles according to the timber's destination mill. If there is a part-load at the stand's roadside, this will probably not be transported, but left to rot, as it is not economically worthwhile to transport this timber. If logs are not made at the stand's roadside or at the mill's logyard, they will be made from tree-lengths at the depot or siding.

4.4.2 Transport in the forest-to-mill domain

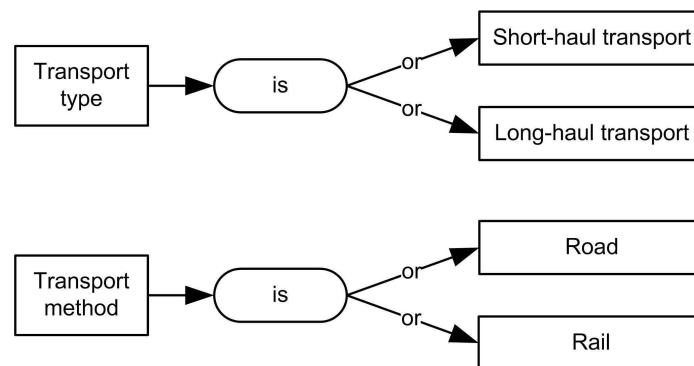


Figure 4.22: ERD showing the different transport type and method options

This domain was introduced in section 4.2. Figures 4.2 and 4.3 (pages 67 and 68) give an overview of how the transport activities fit into the supply chain. There are two types of transport: short-haul and long-haul (see Figure 4.22). In this description, there are also two transport methods for the long-haul transport: by road or by rail.

Figure 4.23 shows the flow of logs (or timber) from the stand to the mill. The logs are taken by road to the depot or siding where they are put into piles according to their

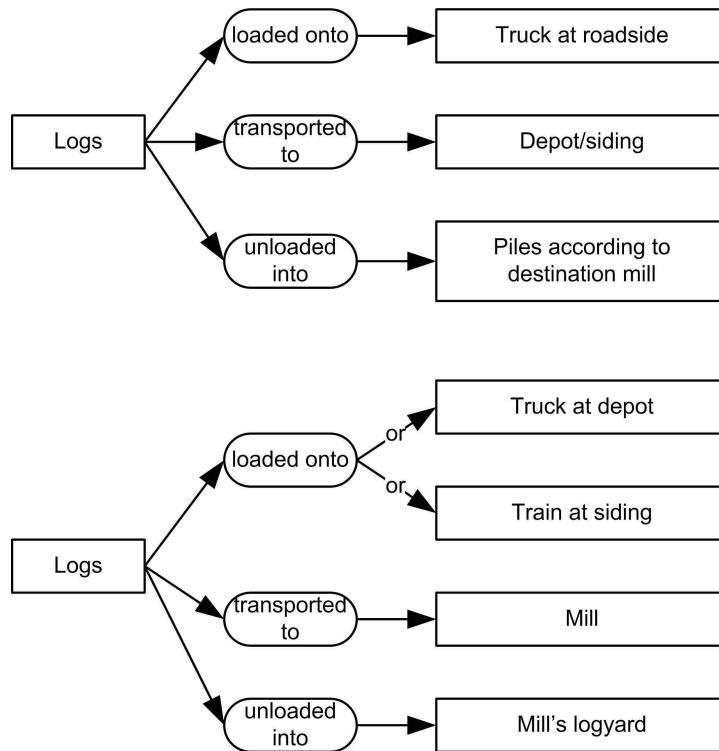


Figure 4.23: ERD showing the transport of logs to the mill

destination mill. From there, they are taken to the mill: by road on a truck, if they are at a depot, and by rail on a train if they are at a siding (see Figure 4.24). When they arrive at the mill, they are unloaded into the mill's logyard.

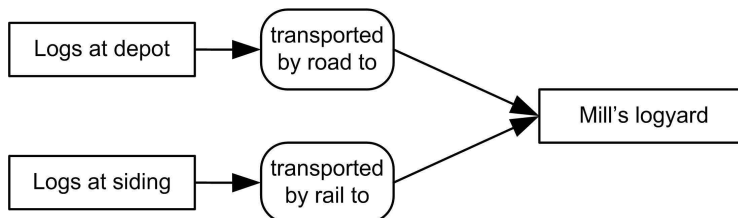


Figure 4.24: ERD showing the loading point for different long-haul transport methods

Timber is transported short-haul from the stand's roadside to a depot or siding. From there, it will be transported long-haul to the mill. The long-haul transport method is road or rail. Short-haul transport is by road only (see Figure 4.22 on page 93). If the

long-haul transport method is road, the timber will be picked up from a depot. If it is by rail, it will be loaded at a siding (see Figure 4.24).

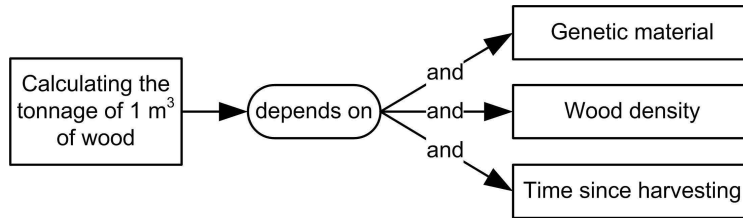


Figure 4.25: ERD showing the relationship between timber volume and its mass

In South Africa, the amount of timber transported is measured by mass (per tonne), whereas in the plantation, its volume is estimated in cubic metres. Elsewhere in the world, the amount of timber transported is measured by cubic metre. There is an almost 1:1 ratio between the timber's volume and mass, but this depends on its genetic material, wood density, and the amount of time since it was harvested (see Figure 4.25). In South Africa, the transportation of the logs is often undertaken by contractors.

4.4.3 Transport costs

Transport costs charged to the integrated forestry company are made up by adding the short-haul and long-haul costs. Both these types of costs can be calculated by multiplying the distance travelled by the cost-per-tonne-per-km rate (see Figure 4.26). The cost-per-tonne-per-km rate could be given in a stratified table, as in Table 4.7.

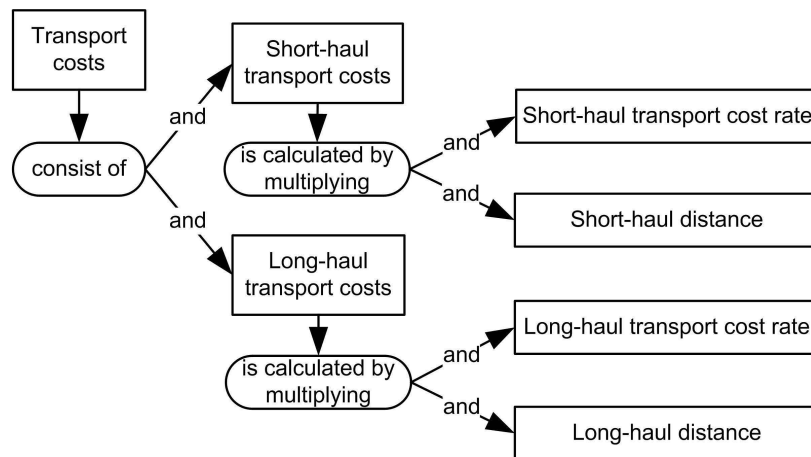


Figure 4.26: ERD of the costs involved in the transport domain

Table 4.7: Example of a stratified cost-per-tonne-per-km table. The costs are fictitious.

Distance (km)		Cost (R/tonne/km)
Minimum	Maximum	
0.0	– 10	25.5
10.1	– 20	22.5
20.1	– 50	20.0
50.1	– 100	19.0
etc.		

4.5 Mill domain

Pulp and pulp products are made in the mill. This section describes the mills as they are at the moment, as well as changes which would need to be made to stratify the wood entering each process, according to wood properties.

The mill domain was introduced in section 4.2. Figures 4.2 and 4.3 (pages 67 and 68) give an overview of how the mill activities fit into the forest-to-mill supply chain. The actions and constraints applicable to the pulp mill are outlined in section 4.5.1, followed by a description of the mill and its processes (section 4.5.2) and the costs and income (section 4.5.3).

4.5.1 Summary of actions and constraints for the mill domain

Table 4.8: Actions and constraints for the pulp miller (processor)

Actions	<ul style="list-style-type: none"> • Accept timber from own forests and/or other suppliers • (Make logs from tree-lengths, if not already done) • (Debark logs) • Remove timber from logyard & feed into pulping process • Make pulp • Find source of additional timber (if not enough produced by own plantation)
Constraints	<ul style="list-style-type: none"> • Need constant timber supply so process can work 24/7 • Need buffer timber stock in logyard in case transport fails or stands cannot be accessed • Some species of timber are not acceptable • Want certain timber species, or timber species mix • Want timber's collection point (depot/siding) to be near mill, because transport costs make up a large proportion of delivered timber costs • Want timber with similar properties to be processed together

The actions and constraints for the processing part of the supply chain are summarised in Table 4.8. The mill accepts timber mostly from its own plantation forests, but could also accept timber from other growers. If the timber arrives at the mill in tree-lengths, logs will be made. If needed, the logs will be debarked (some genera such as eucalypts and wattle are debarked in the field, after harvesting). The logs are fed into the mill's

processes and made into pulp. The last constraint, viz. being able to process timber of similar properties together, is addressed in the forest harvest scheduling system described in section 4.7.

4.5.2 Mills and their processes

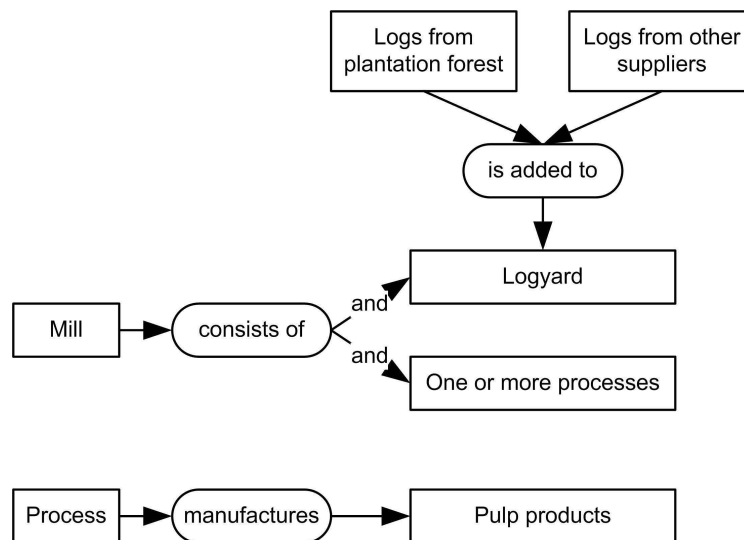


Figure 4.27: ERD showing the entities of the mill

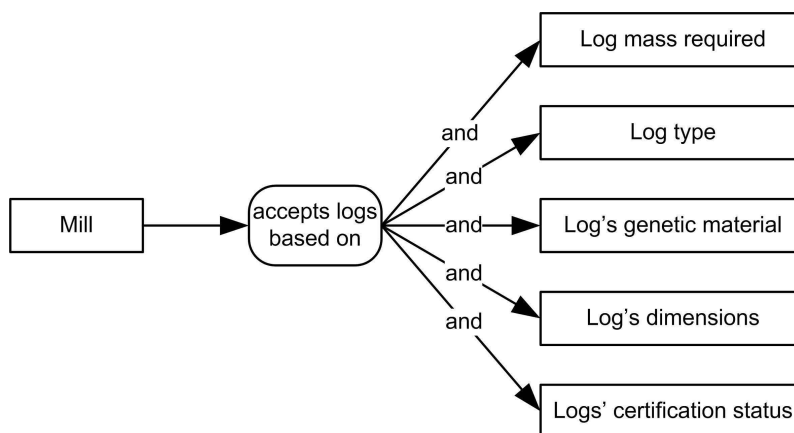


Figure 4.28: ERD giving a mill's criteria for accepting logs

Pulp mills have at least one process and a logyard (see Figure 4.27). Logs from the plantation forestry company's forests and from other timber suppliers' forests are unloaded into the logyard. The process manufactures pulp products.

The mills need a minimum mass of logs per annum to operate continuously. It is unusual for mills to accept logs which are not of the required type (e.g. for a sawmill to accept pulp logs for processing). However, it could happen that a pulp mill would accept higher quality sawlogs or veneer logs, but this would waste the value of those logs. Mills accept logs into the logyard which are of a certain genetic material (i.e. genus and/or species and/or clone/seedling); they sometimes also have log size limits (which, in the case of pulp mills, could be due to limitations of mill equipment, e.g. debarkers). If the forest products are to be sold in markets which require that the forests were responsibly and certifiably managed (e.g. Forest Stewardship Council, 2003a), the mill will require a minimum percentage of the incoming logs to have been grown under those conditions. The system proposed in this dissertation requires that mills also accept logs based on their wood properties. These aspects are shown in Figure 4.28.

Figure 4.29 show the features of a process. A process is typified by the process type (e.g. kraft pulping, thermomechanical pulping (TMP), dissolving pulping), the throughput (the number of tonnes of logs that the process can accept annually) and the recovery rate. The process throughput is determined by the product of the number of hours the process can work annually and the process capacity (the number of tonnes of logs a process can accept per hour). The process recovery rate is the yield from the process (output mass divided by input mass, expressed as a percentage), and is affected by the logs' site index, age, genetic makeup, the altitude at which they were grown, and the type of process.

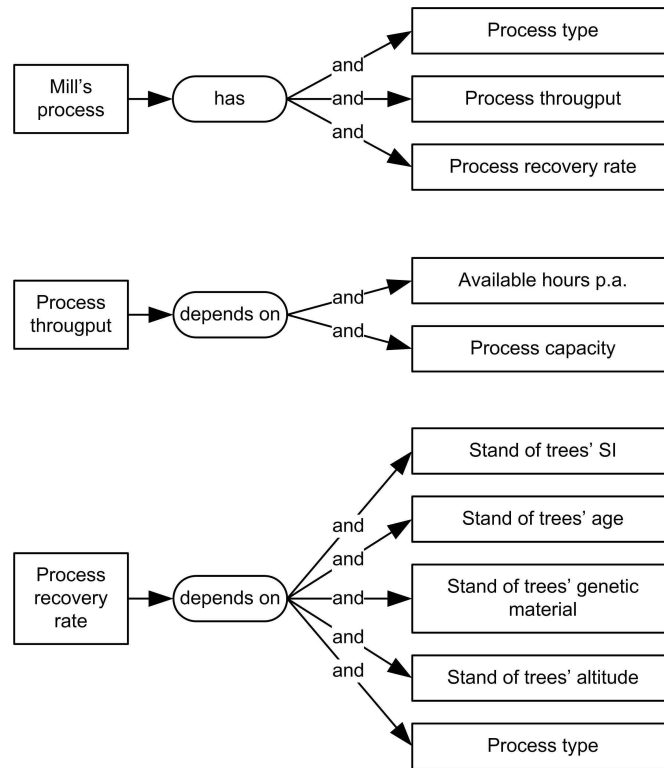


Figure 4.29: ERD showing features of mills' processes

4.5.3 Mill costs and income

The mill costs are made up of two main categories: variable costs and fixed costs. The variable costs are those relating to the logs which are to be processed. If the logs come from the company's plantations, this cost is made up of the log cost and the cost of transporting them to the mill. In the case of logs from other timber suppliers, this cost is the delivered log cost. The fixed costs are the mill costs which will be paid annually, independently of the mass of logs which are processed. They include the cost of energy, chemicals, salaries, maintenance, etc. These costs are outlined in Figure 4.30.

The mill's income is made up of the the product of the sale price of pulp product and the tonnage sold.

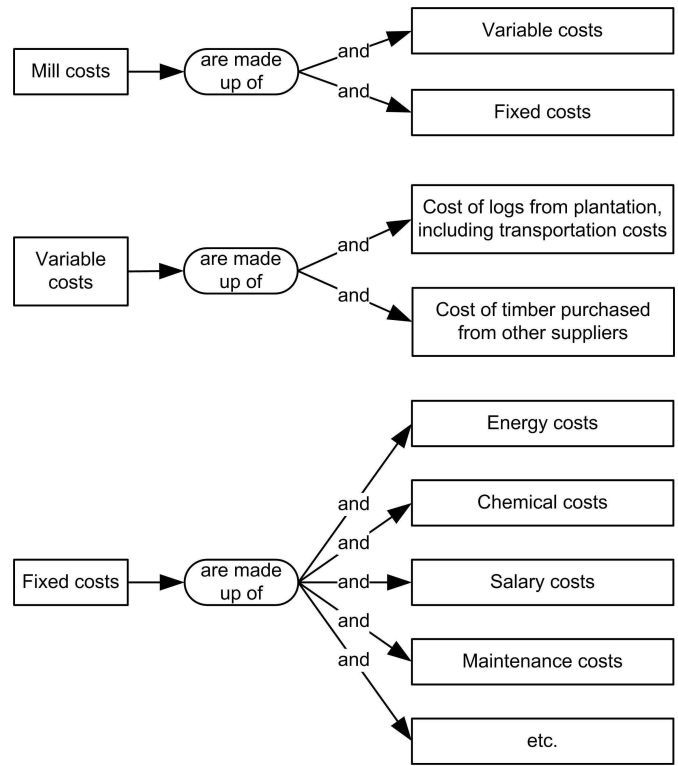


Figure 4.30: ERD showing the costs involved in the mill domain

In South Africa, logs and pulp are sold by mass (i.e. per tonne), whereas in other countries, they are sold by volume.

4.6 Forest planning domain

As the forest harvest scheduling system to be specified is affected by forestry planning, this domain is described next. This section starts with a summary of the actions and constraints which affect the forest planning domain (see section 4.6.1). Since the forestry company being described is vertically integrated, the planning applies to aspects of the forests, the transport and the mill.

Table 4.9: Actions and constraints for the integrated forestry company staff when planning

Actions	<ul style="list-style-type: none"> • Develop long-term, medium-term and short-term plans • Check feasibility of the plans from a long-term, medium-term and short-term perspective • Check feasibility of the plans from a forestry, transport and mill perspective • Swap stands to be harvested between the years of the medium-term plan, if necessary • Swap stands to be harvested between the months of the short-term plan, if necessary • Find additional timber if not enough trees are growing in the plantation • Find markets for excess timber (if more trees need harvesting than mills require)
Constraints	<ul style="list-style-type: none"> • Forest must produce wood at the same rate that it is being harvested • There are many stands, each with trees of differing levels of maturity and growth rates • Mills must receive required mass and species mix of logs • Sufficient proportion of the logs entering each mill has been grown according to certification standards • Logs entering mill have the required wood properties • There is a transport link between the stand and its possible mill destinations • Logs are allocated according to capacity of transport method (road, rail, ...) • There should be (but may not be) enough storage capacity at the depot/siding and mill • There should be (but may not be) enough transport capacity to move the timber to the depots/sidings and from there to the mills • There should be (but may not be) enough loading and unloading capacity at the depots/sidings and unloading capacity at the mill • There may be limited resources to undertake establishment, silvicultural, maintenance, harvesting and transportation activities • Make sure that roads in the estates are upgraded prior to harvesting • Cannot harvest stands with soils sensitive to compaction in wet weather • Make sure that there are enough stands scheduled to be harvested each year which do not have soils sensitive to compaction in wet weather • Stands to be harvested in a region/management level p.a. can be harvested at a fast enough rate (i.e. not all stands to be harvested in a particular year are on steep slopes or difficult terrain) • Stands to be harvested in one year should not be spread across the forest landscape. Preference is for stands to be in close proximity to one another. This impacts on upgrading roads prior to harvesting and moving harvest equipment and teams around at harvesting. It should be borne in mind that a stand scheduled to be harvested in November/December should be close to those scheduled to be harvested in the following January. • Nursery should have sufficient stock of trees (seedlings or cuttings) to be planted (but may not) • Establishment, silvicultural and harvesting contractors have enough work within a management level (e.g. a group of estates) • Need to ensure that all plans (short-, medium- and long-term) agree with each other

When managing a stand of trees, one needs to know their volume at harvesting age. Estimation of a stand's volume is described in section 4.6.2. The assessment of the volume of the whole company's trees at harvesting age is then discussed in section 4.6.3. Section 4.6.4 gives a description of the planning horizons involved in developing a plan for the plantation and the contents of the plans for each planning horizon. Finally, the process of developing a long-term plan which is feasible at the tactical and operational levels is described in section 4.6.5.

4.6.1 Summary of actions and constraints for the forest planning domain

Table 4.9 gives a summary of actions which the integrated plantation forestry staff undertake when generating a strategic plan, and the constraints which act on these actions. The long-term, medium-term and short-term plans form one plan with varying time horizons and details. The plans must agree with each other. (The plans and their contents are described more in section 4.6.4). There are constraints from each of the three domains (forestry, transport and mill).

The fifth constraint, viz. that logs entering mill have the required wood properties, is addressed in the forest harvest scheduling system described in section 4.7.

4.6.2 Stand volume prediction

An integral part of forest planning is being able to predict the volume of the forest stand. This is done using a growth and yield modelling system. The system uses a series of equations to predict the stand's average height, diameter at breast height (DBH), and utilizable volume (von Gadow and Bredenkamp, 1992, pp.50–72). The system predicts the volume of a stand of trees, using as input the genetic makeup of the trees planted

at the stand, the stand's regime (from which can be derived the planting purpose or working circle). The stand's site index (or growth rate), age, stem density and area are also inputs. The co-efficients of the growth and yield models to be used must also be input, as well as the bucking pattern (how the growth and yield modelling system should cut the tree-lengths up) (see Figure 4.31). When calculating the stand's volume, the actual regime data is used where available; for actions yet to happen in the stand, the planned regime actions are used.

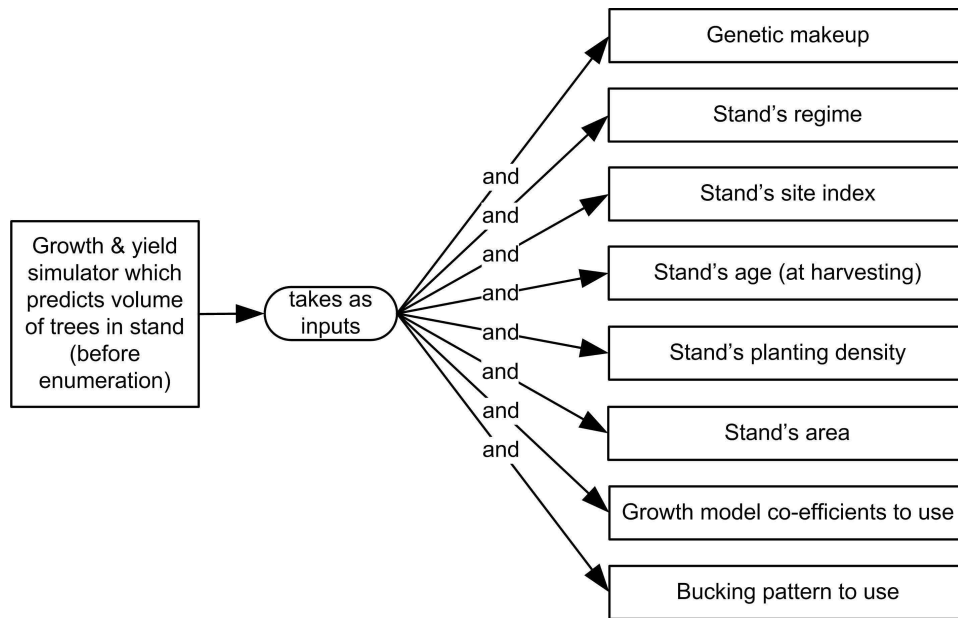


Figure 4.31: ERD giving inputs to the Growth & yield simulator before enumeration

After enumeration, the equations can be calibrated by using more accurate stand information (shown in grey in Figure 4.32). The information gathered during the enumeration is the stand's site index (obtained from measuring the trees' heights), the trees' diameter at breast height (DBH) and the stocking density of the stand (how many stems per hectare there are).

The growth and yield models used to predict the stand's volume are developed by measuring the heights and growths of trees planted in many regions and then running

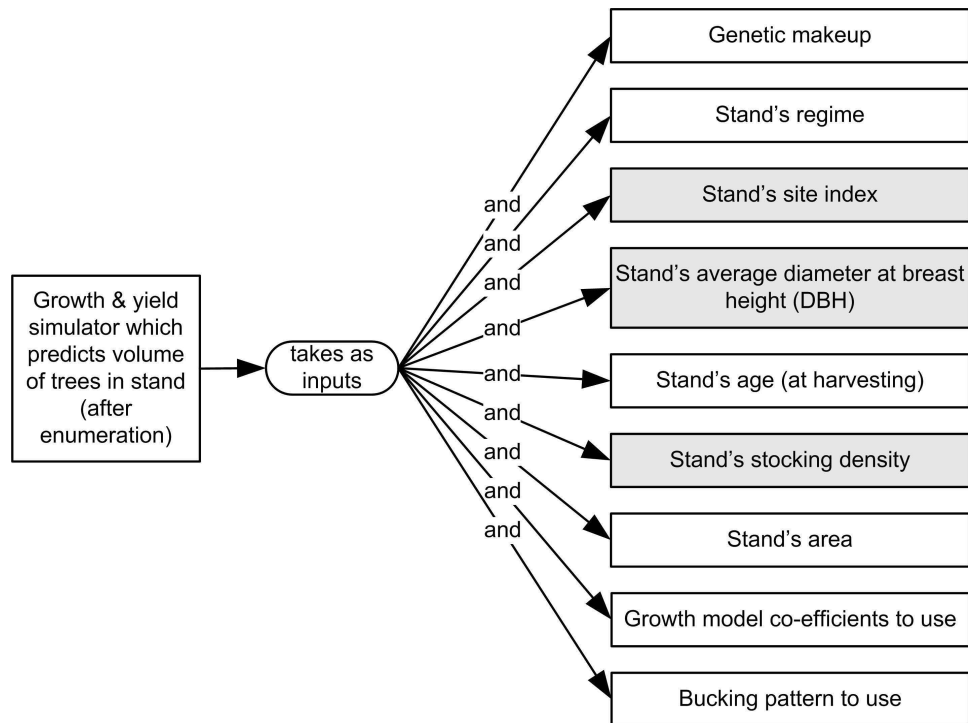


Figure 4.32: Inputs to the Growth & Yield Simulator after enumeration. The greyed inputs are added or updated during the stand enumeration.

regressions to obtain the model co-efficients for a particular planting purpose (e.g. pulpwood), genetic material, and area (Barros and Weintraub, 1982). Process-based models which predict the volume of a stand of trees based on the factors shown in Figure 4.9 (page 78) (e.g. rainfall, solar radiation, day length) are starting to be developed by the scientific community (Landsberg and Waring, 1997; Dye, 2001; Landsberg *et al.*, 2001; Sands and Landsberg, 2002; Landsberg *et al.*, 2003; Almeida *et al.*, 2004; Nightingale *et al.*, 2008), but are not common; much work still needs to be done before there are process-based models for all the genera and species which are planted by the plantation companies.

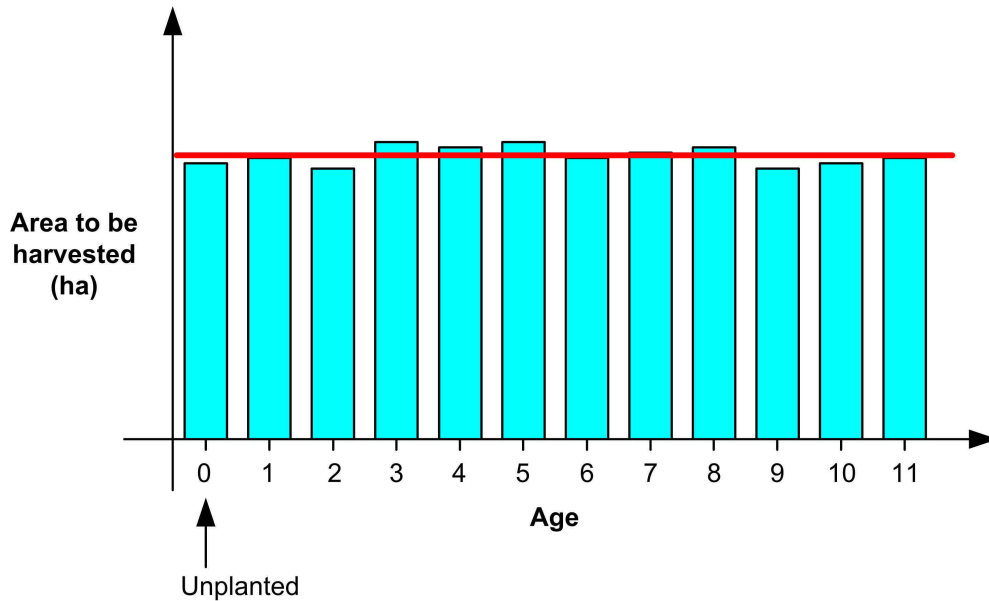


Figure 4.33: Age-class distribution which is nearly normalised

4.6.3 Annual forest volume prediction

When developing a long-term plan for the forest, the Planning forester needs to ensure that the volume of wood that is cut from the plantation in any particular year is being replaced by growing trees which will be ready for harvest in subsequent years. For an even annual mill intake, the forestry company would not want to harvest faster than the rate which the forest is growing (as it would diminish future stocks); nor would it (generally) want to harvest at a slower rate. Figure 4.33 shows the graph of an age-class distribution which is used to help foresters assess whether their forest is “normal” (i.e. producing as much as is being consumed). The horizontal line shows the area that should be harvested each year. Areas which fall under the line mean there is not enough timber available for that year; areas above the line means that there is a surplus. In this figure, there is a slight timber surplus in three, four, five and eight years’ time, and a deficit in two, nine and ten years’ time. For plantations with a wide range of growth rates, the areas in the age-class distribution must be adjusted for growth rate (von Gadov and Bredenkamp, 1992, pp.91–101). Age-class distributions can be drawn for the whole company’s plantations, or for a particular level in the

management hierarchy. They can be drawn for different end uses (or working circles, e.g. pulpwood, sawtimber), and also for different genera or genus-species combinations.

4.6.4 Planning horizons and plan contents

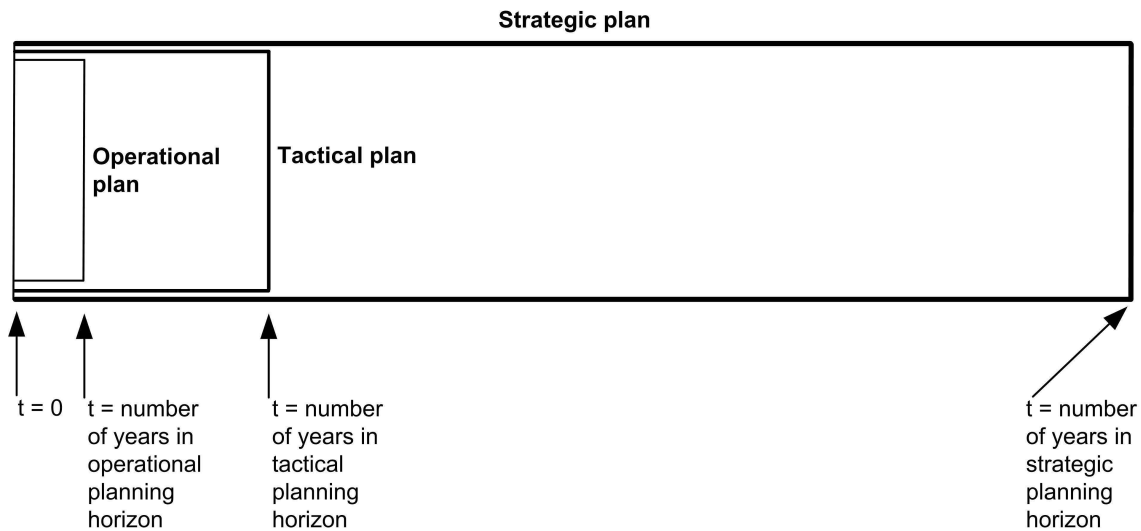


Figure 4.34: Relationship between the strategic, tactical and operational plans and their planning horizons

When planning the plantation, there are three planning horizons which need to be considered: the long-term, or strategic horizon, the medium-term or tactical horizon and the short-term or operational horizon. Even though it sounds like there are three different plans for the three planning horizons, they are in fact one plan (see Figure 4.34), with the operational plan having more detail than the tactical plan, which in turn has more detail than the strategic plan. A typical time horizon for the strategic plan is the length of two rotations; the tactical plan is typically three to five years long and the operational plan typically one year long. The operational plan's information is detailed at a monthly level.

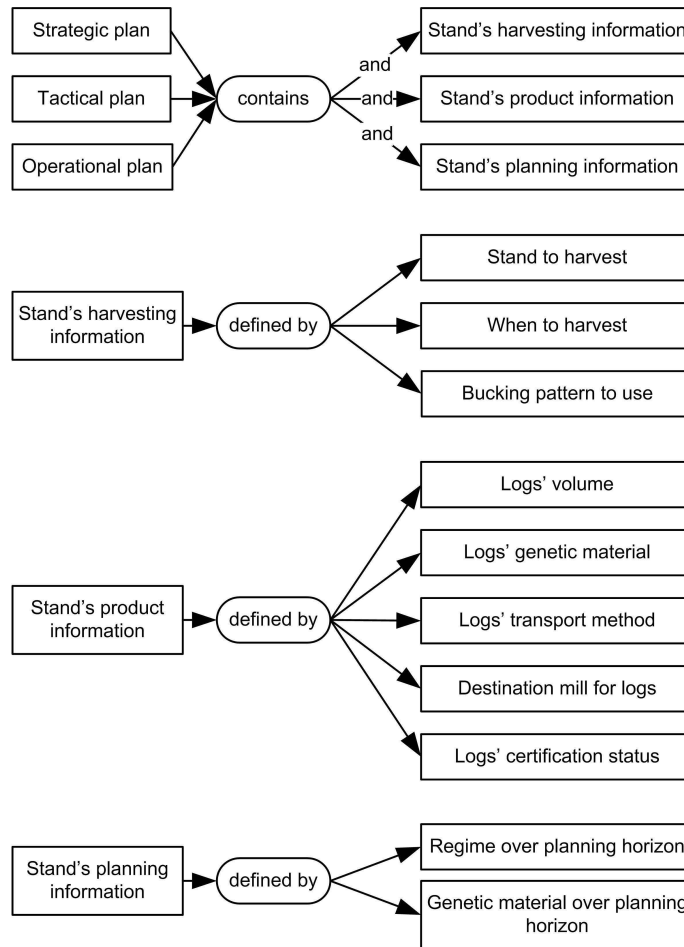


Figure 4.35: ERD showing the contents of the strategic, tactical and operational plans

Figure 4.35 gives more detail of the contents of the plans. All of the plans contain the same types of information, i.e. the stand's harvesting information, the stand's product information and the stand's planning information. The operational plan has the information by month, whereas the tactical and strategic plan has it by year. The stand's harvesting information includes which stand to harvest and when. It also includes the bucking pattern to be used (for pulp regimes this is simple, as the trees are cut into 2.4m long logs). The stand's product information includes the expected log volume from the stand (by log type category), the genetic material (genus, species, ...) of the logs, the mill which has been allocated the logs and the method of transport for getting them there, and the logs' certification status (i.e. was the stand in which the trees

grew managed in a certifiable manner). The stand’s planning information includes the regime information for each stand for the planning horizon, and the genetic material which is to be planted.

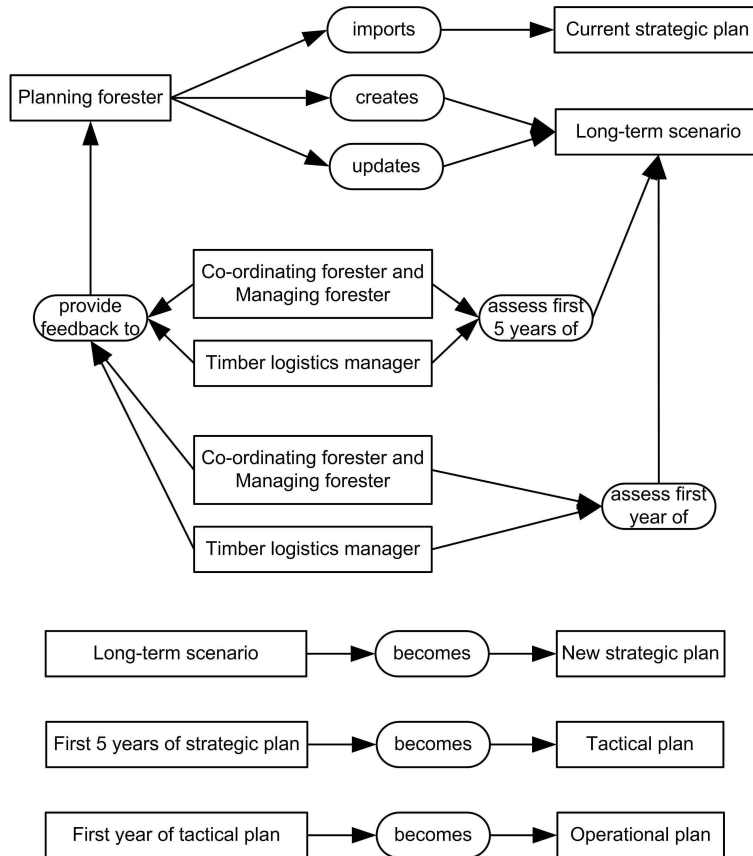


Figure 4.36: ERD showing the development of the strategic, tactical and operational plans, and the integrated forestry company staff’s involvement in its development

4.6.5 Developing the plans

This section describes the planning process. It is presented in a number of ways: firstly using ERDs and a state chart. The criteria for judging the feasibility of the plan are also shown using ERDs. The planning process is then described in more detail using business process diagrams.

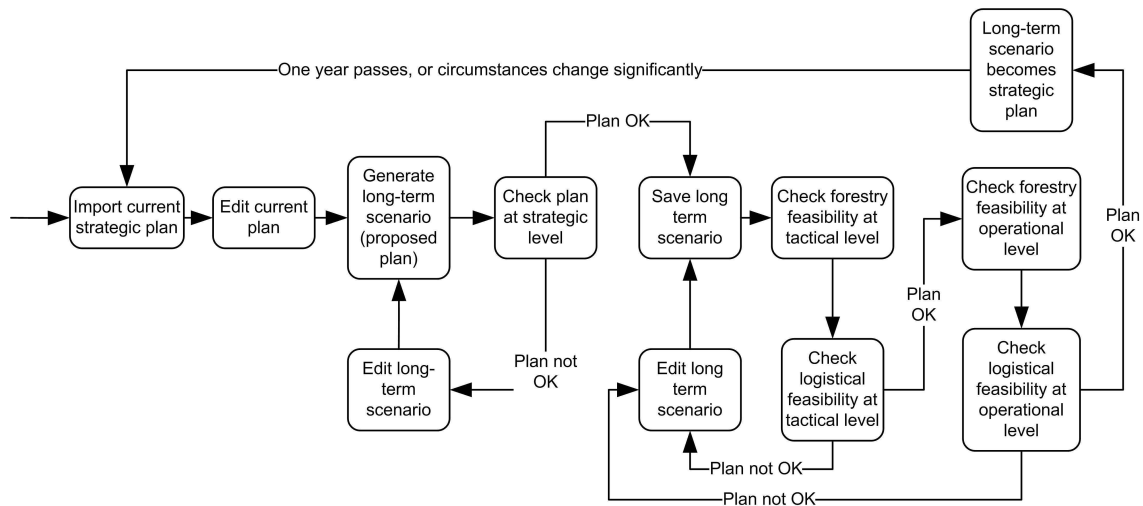


Figure 4.37: State chart showing the development of the strategic, tactical and operational plans

The planning process is very iterative as can be seen in Figures 4.36 and 4.37. It is undertaken annually (or more often, if the forest's ability to produce logs is hampered or the mill's demand for logs changes significantly). It starts with the planning forester reading in data from the plantation database. This data includes the actions for each stand which have already taken place, as well as planned actions (from the current strategic plan). Using these inputs, volume prediction models are run to determine the volume of the stand at felling. A long-term scenario, or proposed plan is developed.

The Planning forester assesses whether the long-term scenario is feasible by checking if the plantation has a normalised age-class distribution over the planning horizon (see Figure 4.33 on page 106), and if the mill's requirements for logs are met (see Figure 4.38). The mill needs the correct mass of logs annually over the planning horizon. The logs must be of an acceptable genetic material, and a sufficient mass of logs with certification status must be allocated to the mill.

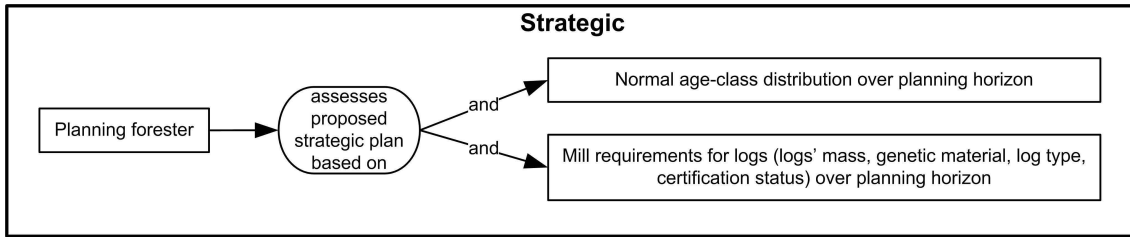


Figure 4.38: ERD showing the criteria on which the strategic plan is assessed by the Planning forester

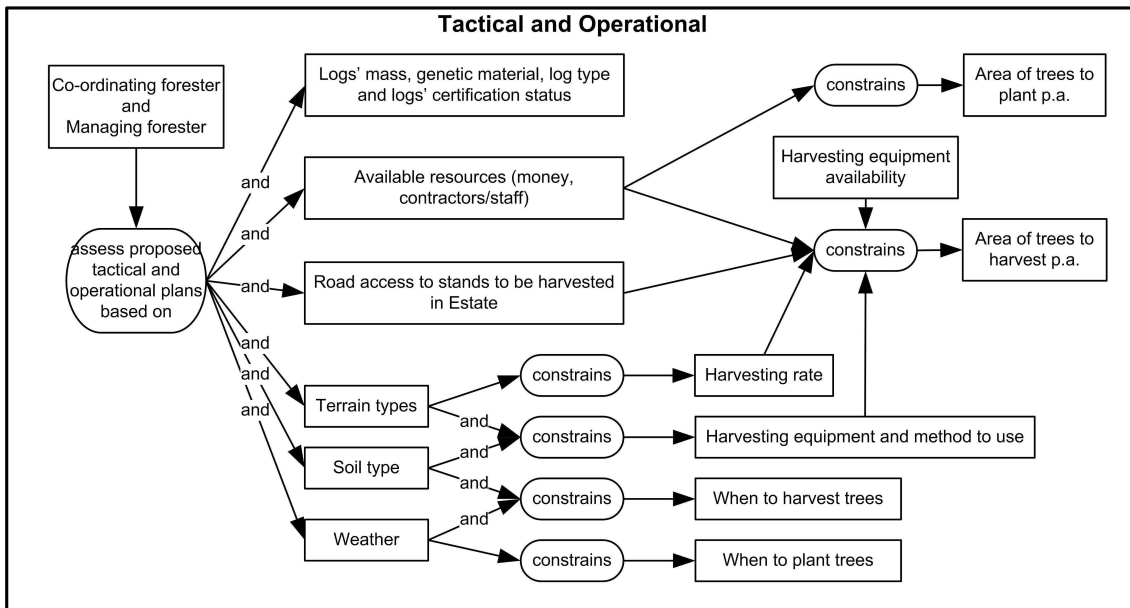


Figure 4.39: ERD showing the criteria on which the tactical and operational plans are assessed by the Co-ordinating and Managing foresters

When the strategic-level scenario is deemed feasible, it passed on to the Co-ordinating and Managing foresters. They assess the first five years of the plan at a tactical level (see Figure 4.39). In addition to the mills receiving the correct mass of logs, with acceptable genetic material, log types, certification status and they have to assess that the plan is feasible from a forestry point of view. There has to be enough resources (money in the budget, equipment, staff or contractors to carry out the plan. In the case of contractors, they need to be assigned enough work in a particular region (or lower

management level) to meet their contract obligations.) If there is a lack of resources, this may restrict the area which can be planted each year. The availability of resources, as well as road access to stands which need to be harvested, the harvest equipment and the harvesting method which need to be used, the harvesting equipment availability, and the estimated harvesting rate for the stand will affect the area of trees which can be harvested each year. For example, if all the stands to be harvested in a particular year happen to be on steep slopes, it will take longer to harvest these stands per hectare than if all the stands were all on flat terrain. The terrain type and the soil type constrain the harvesting equipment and method which can be used, as certain soils are more susceptible to soil compaction than others. Compacted soil has a reduced growth rate for the next crop. The soil type and the weather (especially wet weather) constrain when some stands of trees may be harvested, as the wet weather would increase the chances of soil compaction. At a tactical (and operational) level, the foresters need to make sure that there is a mix of stands with sensitive and not sensitive soils allocated to be harvested over the year, so that in the wet season, all harvesting will not have to cease. Weather constrains when trees can be planted. Trees cannot be planted when frost may occur, and when there is a distinct rainy and dry season, planting activities usually wait until the rainy seasons has started. One hectare of trees can usually be planted faster than it can be felled.

If the tactical plan is not feasible, it is sent back to the planning forester for adjustments. If it is deemed feasible, the Logistics manager reviews it for logistical feasibility (see Figure 4.40). This involves checking that the mill requirements for logs are met, that a transport link between the depot or siding and the mill exists, that there will be sufficient storage capacity at the mill and at the depot and siding. There needs to be sufficient unloading capacity at the mill and loading capacity at the depot or siding. There need to be sufficient vehicles and/or rail trucks available to do the transporting.

If the tactical plan is not feasible, it is again sent back to the planning forester for ad-

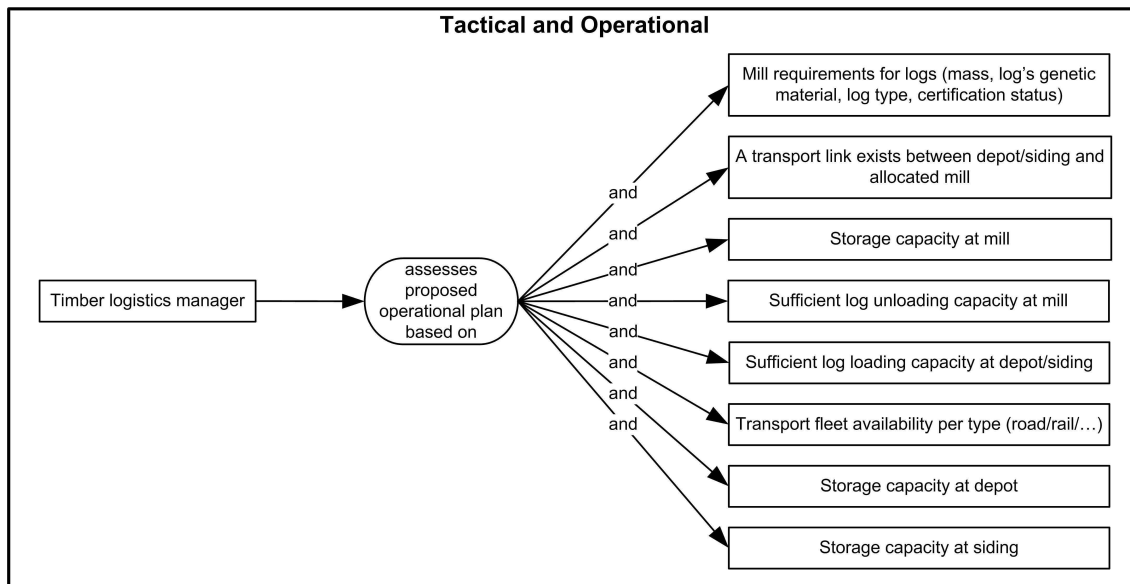


Figure 4.40: ERD showing the criteria on which the tactical and operational plans are assessed by the Timber logistics manager

justments. If it is feasible at a tactical level, the first year of the plan is studied by the Co-ordinating and Managing foresters to assess forestry feasibility and by the Timber logistics manager to assess logistical feasibility. In the operational plan, the activities are allocated to specific months. Similar criteria are used to assess the tactical and operational plans. If the plan is not feasible, it is referred back to the Planning forester for adjustment. If it is feasible at all these levels, the long-term scenario becomes the new strategic plan (with its encapsulated tactical and operational plans). The strategic plan is only regenerated after a year, during the planning cycle, or else if circumstances change significantly (e.g. a fire burns a large proportion of the stands in one area, or the market demand for logs changes drastically).

The planning cycle described above is described in more detail using Business Process Modelling Notation (see Figures 4.41 to 4.58).

Strategic planning starts with the Planning forester creating long term scenarios (see

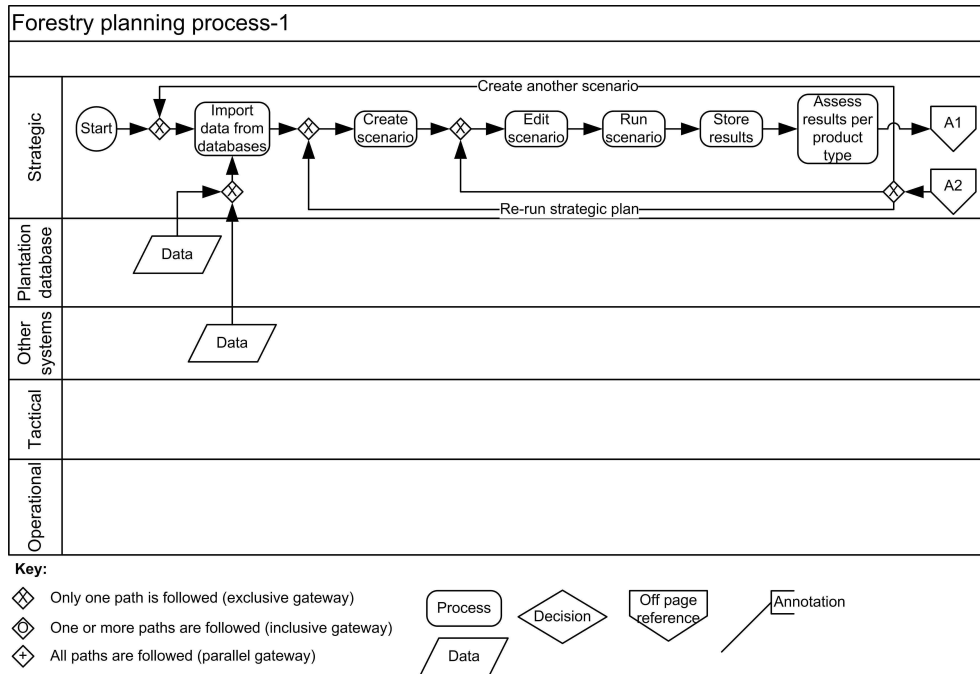


Figure 4.41: Business process diagram of the planning process (part 1 of 18)

Figure 4.41). To do this, he imports data from the plantation database and from other relevant systems or databases into a forest harvest scheduling system. He may need to edit the scenario's parameters, or scenario data. (The latter would occur if he were running "what-if" scenarios, for example, "what if the forestry company stopped growing wattle?" or, "what if the forestry company took on an additional lease in two years' time?") The scenario is then run using the forest harvest scheduling system and the results stored and analysed (see Figure 4.42). The two main assessments, by forest product type (e.g. pulpwood, sawtimber), are whether the plantation is normalised over the planning horizon; and whether the mills will receive a sufficient mass of logs, of acceptable genetic material, and which has the required certification status over the planning horizon. If the normalisation requirements or the mill requirements are not met, a new scenario is created, or the scenario's details are edited and the forest harvest scheduling system re-run.

If, after many scenarios have been run, there is a shortage of wood for a particular year,

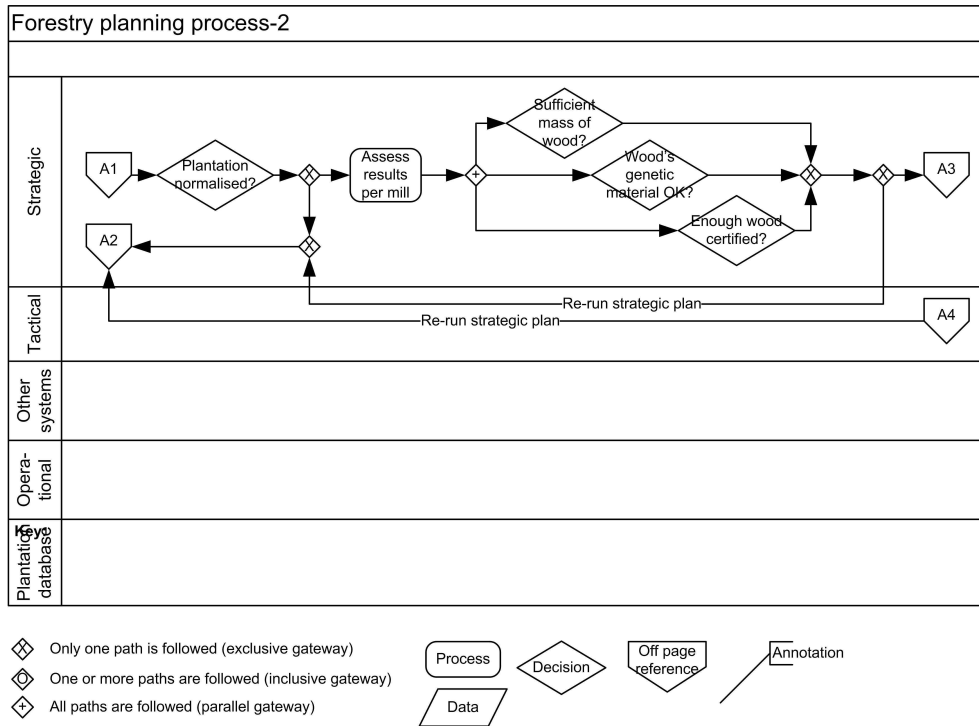


Figure 4.42: Business process diagram of the planning process (part 2 of 18)

the Timber logistics manager will be notified and requested to find alternative sources of timber (see Figure 4.43). The Timber logistics manager will also be notified if it is estimated that logs of unwanted genetic material will be produced by the plantation (these logs could be sold to other mills). Once all avenues for improving the scenario have been exhausted, the best scenario is chosen. This becomes the proposed strategic plan, and it is stored in the plantation database and other systems. The Planning forester then advises the Co-ordinating foresters that they can commence the tactical planning cycle for the land that they oversee.

The Co-ordinating forester imports the first five years of the proposed strategic plan, for the land that he oversees from the plantation database and other systems (see Figure 4.44 on page 117). Together with each Managing forester that reports to him, he assesses the terrain types and the soil types of all the stands that are due to be felled. If the terrain of too many of the stands due to be felled over the five year period is

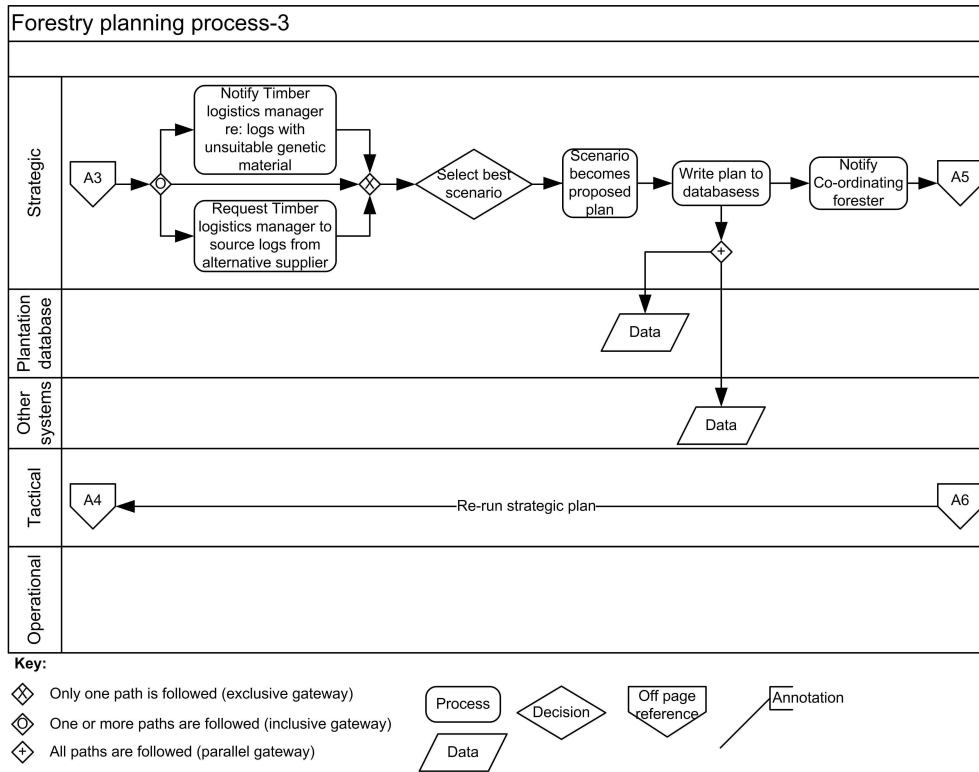


Figure 4.43: Business process diagram of the planning process (part 3 of 18)

step, the harvesting rate will not be sufficient to meet the log mass demand at the mill. Similarly, if all the stands due to be felled during the five year planning horizon have soils that are sensitive to compaction during wet weather, harvesting will have to stop during rainy weather and the required harvesting rate will not be achieved. If either of these conditions is not met, the strategic scenario may need to be re-run.

Harvesting equipment (and if applicable, harvesting contractors) are then allocated to the stands which are to be felled (see Figure 4.45 on page 118). (Harvesting contractors own their own harvesting equipment, so allocating a harvesting contractor implies also allocating harvesting equipment). The harvesting rate of the stands due to be felled is then assessed. If the harvesting rate is not fast enough, stands due to be felled are swapped between the years of the tactical planning horizon to even out the harvesting rate. Alternatively, an additional contractor could be allocated, thus increasing the

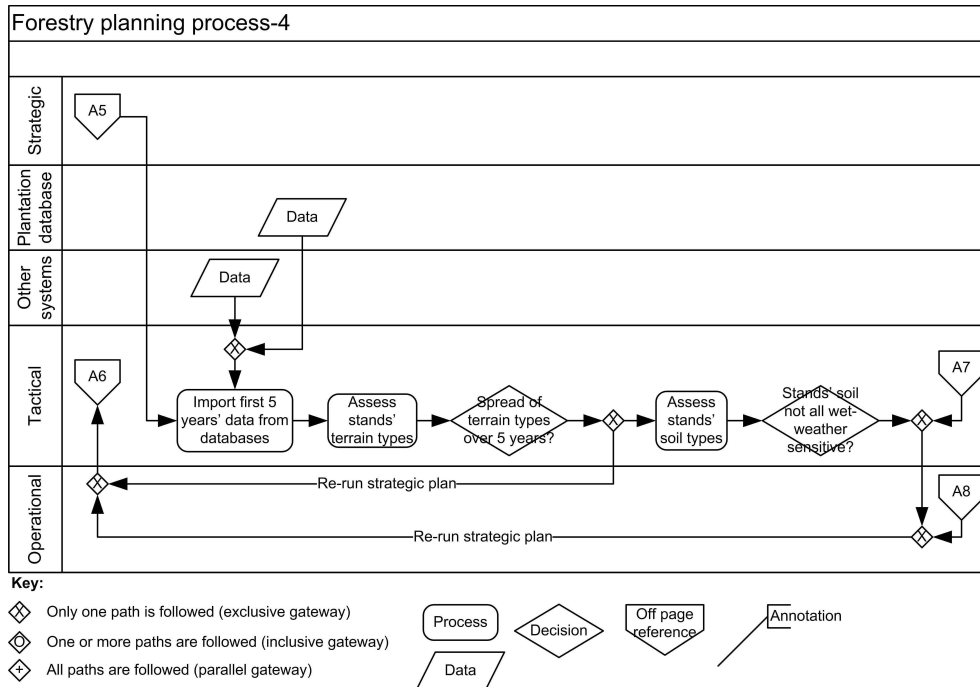


Figure 4.44: Business process diagram of the planning process (part 4 of 18)

harvesting rate.

Because moving the harvesting equipment is costly and unproductive (in terms of harvesting), it is better to harvest all the required stands in one area before moving the equipment to another area. If stands to be felled are widely spread in the plantation, stands can be swapped between the years of the tactical plan to try to amalgamate harvesting areas. The mill's requirements also need to be taken into account (see Figure 4.46 on page 119). The area for which the Co-ordinating forester is responsible needs to produce a sufficient mass of logs, which have an acceptable genetic material; sufficient wood needs to have been grown using certified methods. If this is not the case, stands need to be swapped between years and/or the strategic plan needs to be re-run.

The roads that lead from stands (which are due to be felled) to the depot or siding need to be assessed to ensure that they will be upgraded before harvesting commences (see

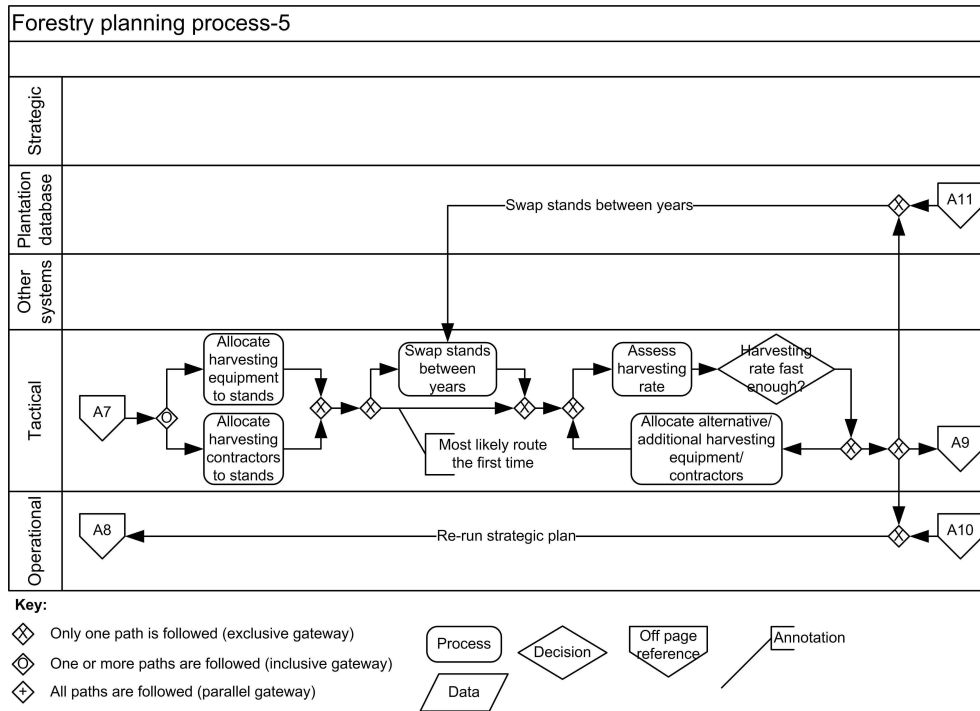


Figure 4.45: Business process diagram of the planning process (part 5 of 18)

Figure 4.47 on page 120). If they cannot be upgraded in time, stand swapping could be considered (provided that the requirements outlined in the previous paragraph are met, viz. fast enough harvesting rate, stands to be harvested in close enough proximity and logs meet mill requirements). An alternative is that the strategic plan be re-run.

The availability of planting stock having an appropriate genetic material is assessed next to ensure that the stand can be re-planted on time (see Figure 4.47 on page 120). If it seems that the right quantity will not be available, a second-best genetic material (genus or species or hybrid or clone) could be chosen (although it would be more common for this to occur at the operational planning phase).

Finally, a check needs to be made that the proposed plan will fall within budgetary constraints for the five-year period (see Figure 4.48 on page 121). If enough resources are not available, stand swapping could be considered, or the strategic plan re-run. After

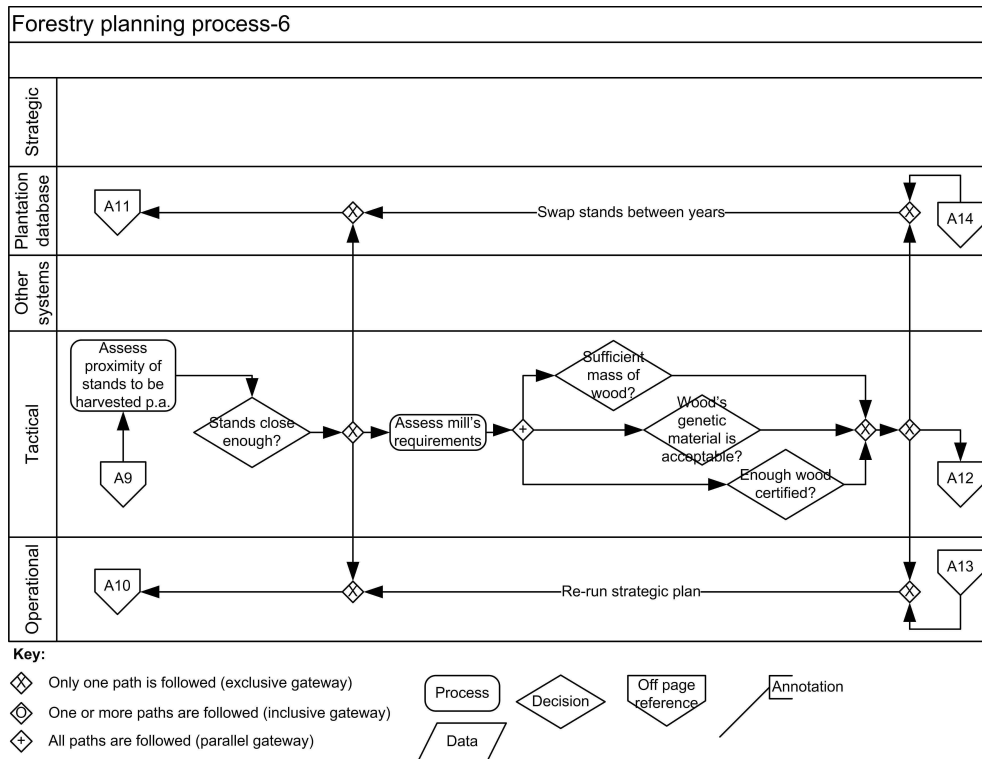


Figure 4.46: Business process diagram of the planning process (part 6 of 18)

these iterations have been completed, the tactical plan is deemed to be acceptable and is written back to the databases. The Timber logistics manager is then notified that he can start reviewing the tactical plan from a logistics point of view.

The Timber logistics manager reads the tactical plan from the databases, and assesses the log loading capacity in the estate and unloading capacity at the mill (see Figure 4.49 on page 121). If there is an imbalance of capacity he can request the Managing forester to swap stands to be felled between months. He also has to assess whether there will be enough transport (per transport type, e.g. road or rail) available and organise more if necessary (see Figure 4.50 on page 122). Finally, he assesses the mills' needs in terms of the mass of wood required, its genetic material and its certification status (see Figure 4.51 on page 124). If there are problems, he could ask the Coordinating forester to swap stands between the years of the tactical plan. From the

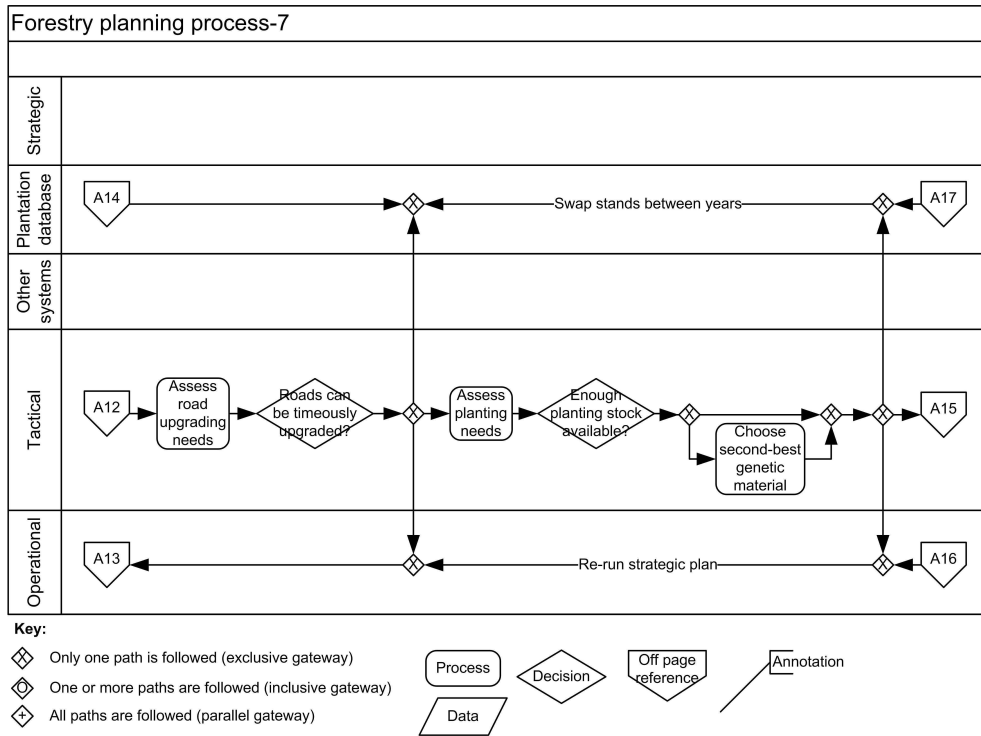


Figure 4.47: Business process diagram of the planning process (part 7 of 18)

analysis undertaken, he would know if trees would be harvested having genetic material unwanted by the mills, or if there were a shortage of wood of particular genetic material. If this were the case, he could arrange to sell the excess timber or try to source wood of suitable genetic material. The Timber logistics manager has the option of requesting that the stands to be harvested are swapped, between the years of the tactical plan, or that a new strategic plan be run. If this is not necessary, the tactical plan is complete.

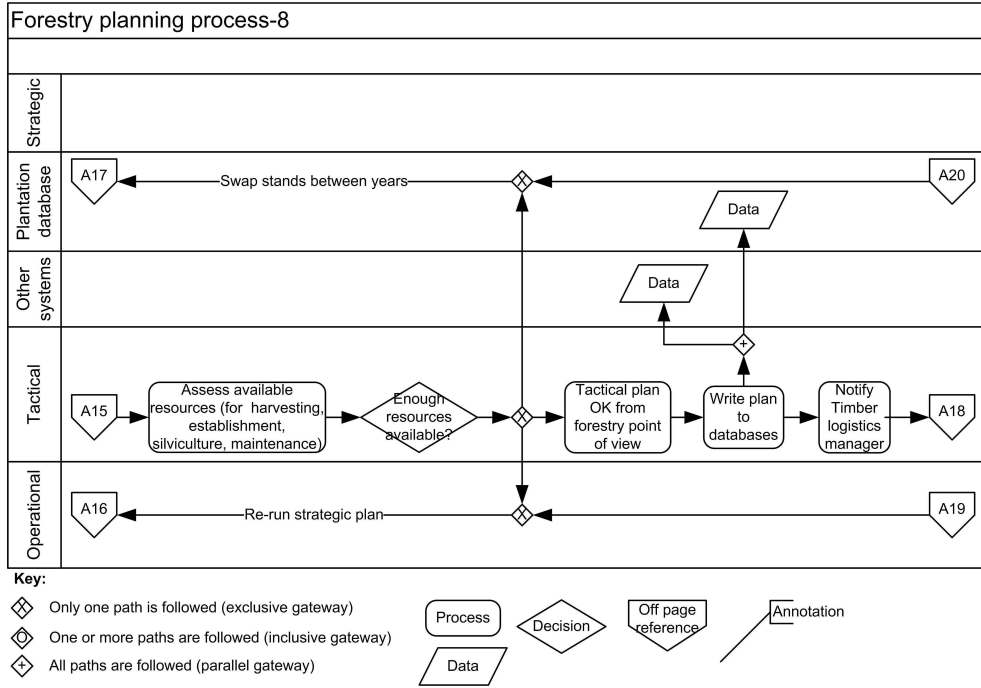


Figure 4.48: Business process diagram of the planning process (part 8 of 18)

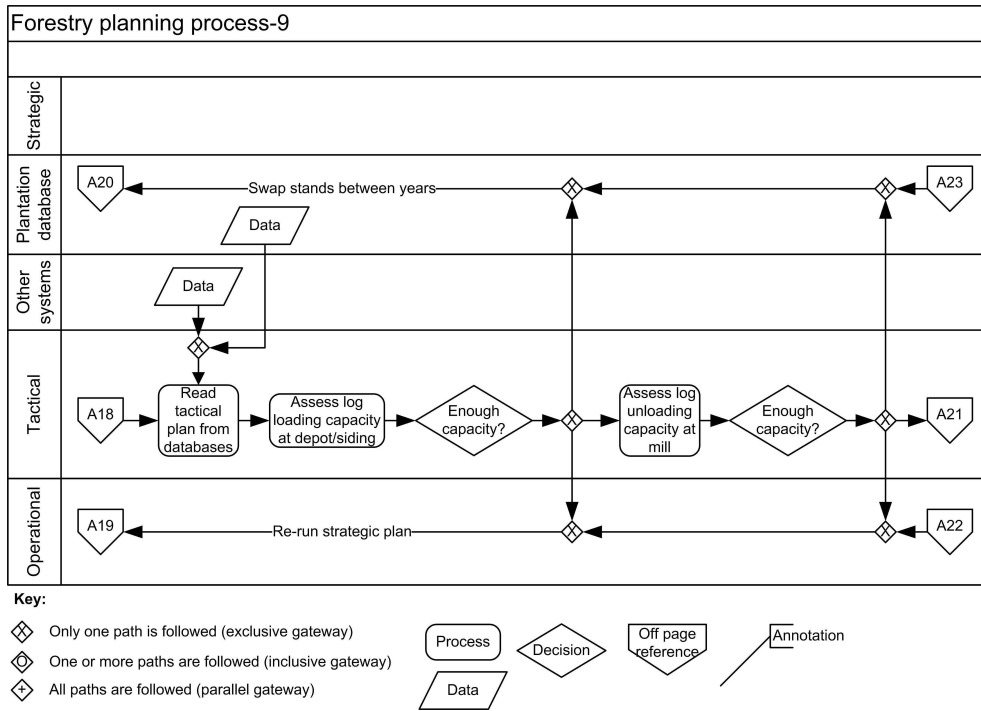


Figure 4.49: Business process diagram of the planning process (part 9 of 18)

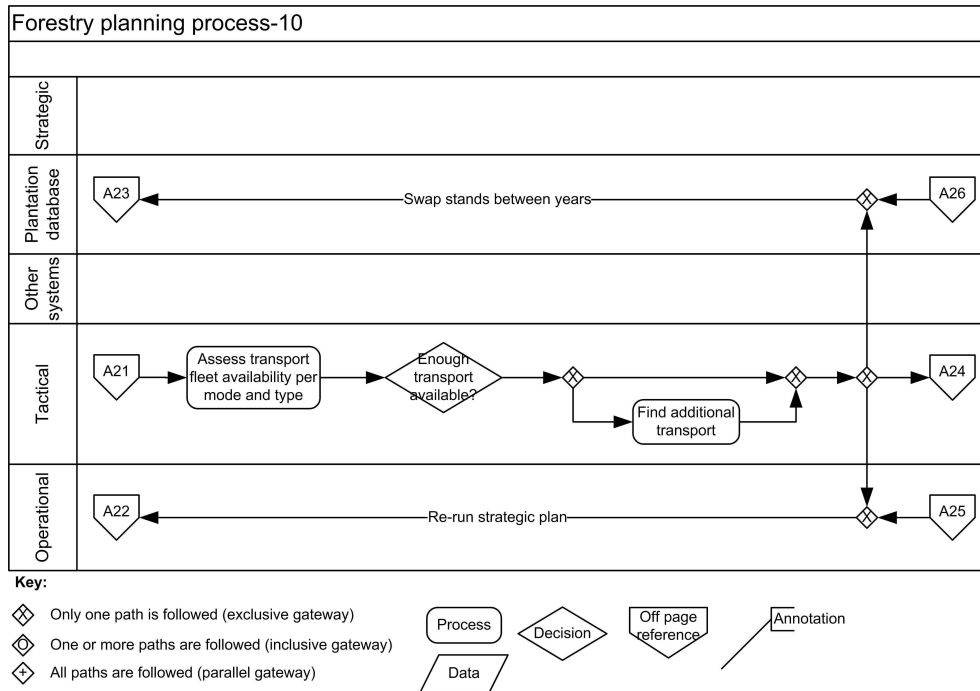


Figure 4.50: Business process diagram of the planning process (part 10 of 18)

The Managing forester is then notified that he can start working on the operational plan for the estates for which he is responsible (see Figure 4.52 on page 125). The Managing forester extracts the data for first year of the tactical (and strategic) plan for the estates under his control. The soil types of stands due to be harvested are assessed and the month during which they will be harvested is allocated. If there are not enough stands to fell in wet weather, the forester can consider swapping stands due to be felled in the tactical planning horizon so that the harvesting rate is not jeopardised.

The terrain of the stands due to be harvested is assessed and harvesting equipment and/or contractors are allocated to them. The harvesting rate is then assessed (see Figure 4.53 on page 125). If the harvesting rate is not fast enough, alternative or additional harvesting equipment or contractors could be allocated. Otherwise, stands can be swapped between the months of the year to even out the harvesting rate. Next, proximity of the stands (which are due to be felled) to each other is examined. If they are not close enough to each other (which would necessitate the costly moving of

harvesting equipment), stands due to be felled could be swapped between the months of the operational plan. The mills' requirements are assessed next to ensure that the estates produce a sufficient mass of logs, which have an acceptable genetic material; sufficient wood needs to have been grown using certified methods (see Figure 4.54 on page 126). Stands to be felled can be swapped between the months of the operational plan to ensure that the flow to each mill is suitable. If swapping between the months of the operational plan is not improving the situation, he could look at swapping stands to be felled between the years of the tactical plan (particularly those years close to the year of the operational plan).

Roads leading from the stands to be harvested are assessed to ensure that they can be upgraded in time to withstand the heaving harvesting equipment and logging trucks. Once again, if roads cannot be upgraded in time, options are to swap stands between the months of the operational plan, or between years of the tactical plan.

The re-planting of the stands, once they are felled, is assessed next to ensure that there is enough planting stock available at the nursery. If not, availability of the second-best genetic material suitable for that stand would be assessed (see Figure 4.55 on page 126). The month of replanting the stands is allocated according to the weather. Finally, the resources available for all the forestry activities (harvesting, establishment, silviculture and maintenance) are checked. If the resources are not sufficient, replanning of activities must be undertaken, either within the operational planning horizon, or the tactical planning horizon.

If there are enough resources available and the plan will ensure the production of a sufficient mass of timber, which has an acceptable genetic material, and that sufficient wood has been grown using certified methods, the plan is saved to the databases (see Figure 4.56 on page 127). Once all the operational plans have been assessed from a forestry point of view, the Timber logistics manager is notified.

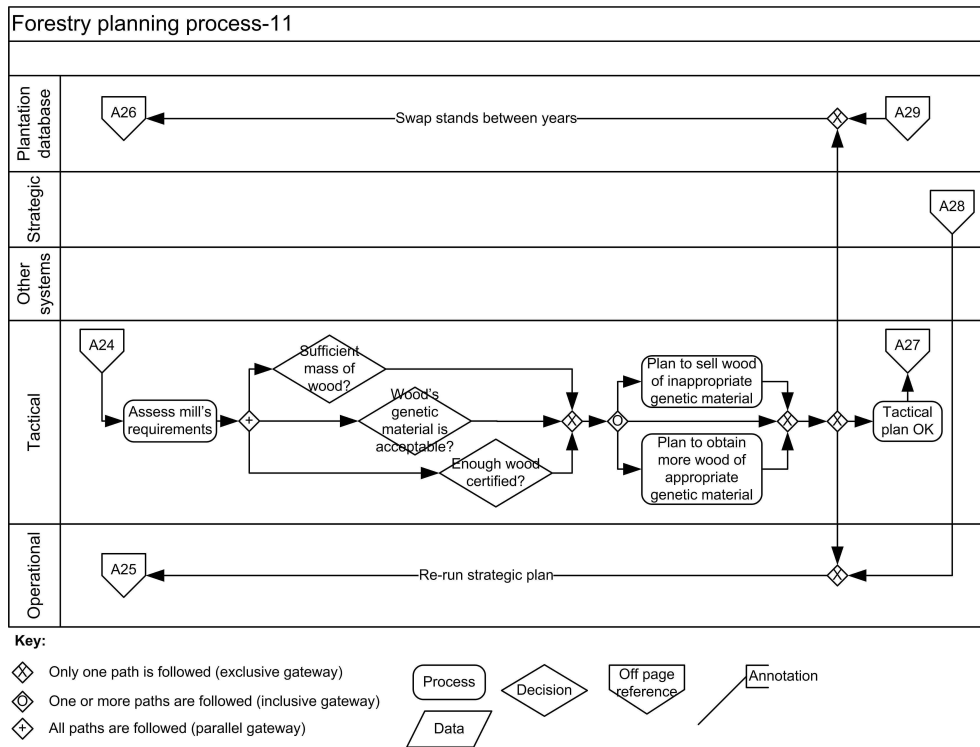


Figure 4.51: Business process diagram of the planning process (part 11 of 18)

The Timber logistics manager reads the operational plan from the databases, and assesses the log loading capacity in the estate and unloading capacity at the mill (see Figure 4.57 on page 127). If there is an imbalance of capacity he can request the Managing forester to swap stands to be felled between months. He also has to assess whether there will be enough transport (per transport type (e.g. road, rail)) available and organise more if necessary. Finally, he assesses the mills' needs in terms of the mass of wood, its genetic material and its certification status (see Figure 4.58 on page 128). If there are problems, he could ask for the stands to be swapped between the months of the operational plan. From this analysis, he would also know if trees would be harvested with genetic material which is unwanted by the mills, or if there were a shortage of wood of particular genetic material. If this were the case, he could arrange to sell the excess timber or try to obtain wood of suitable genetic material. The Timber logistics manager has the option of requesting that the stands to be harvested are

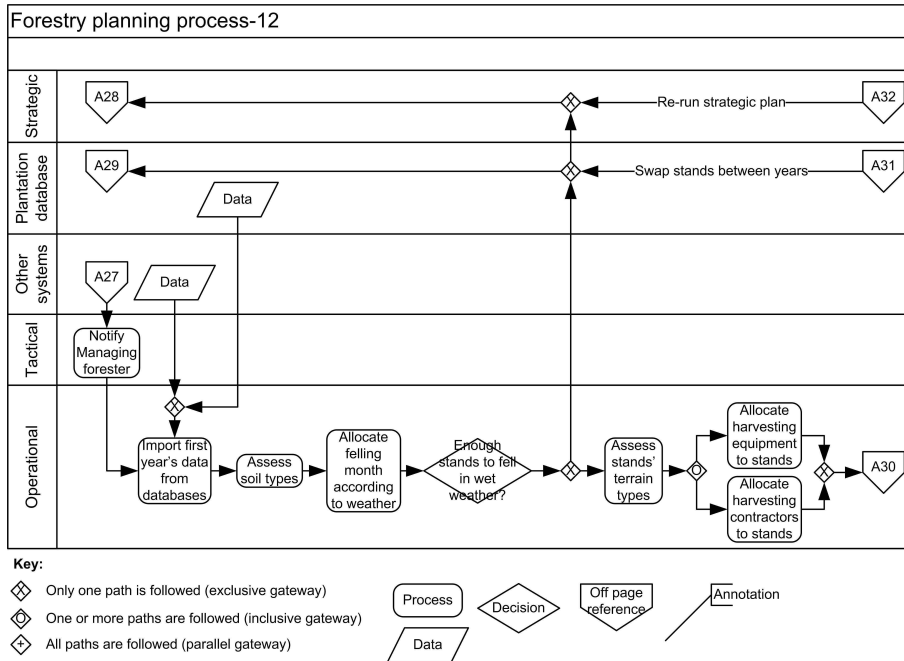


Figure 4.52: Business process diagram of the planning process (part 12 of 18)

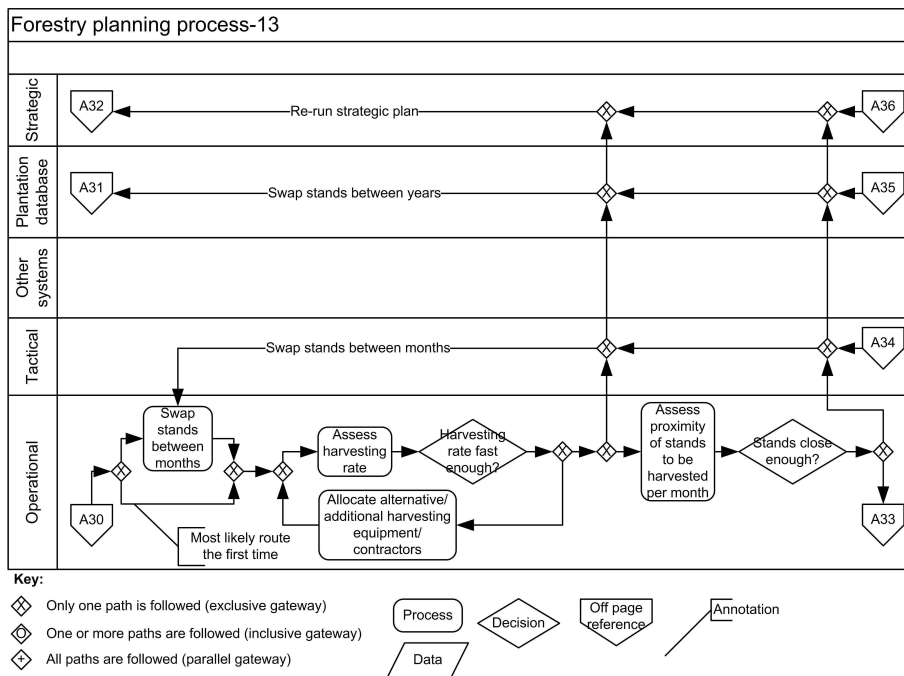


Figure 4.53: Business process diagram of the planning process (part 13 of 18)

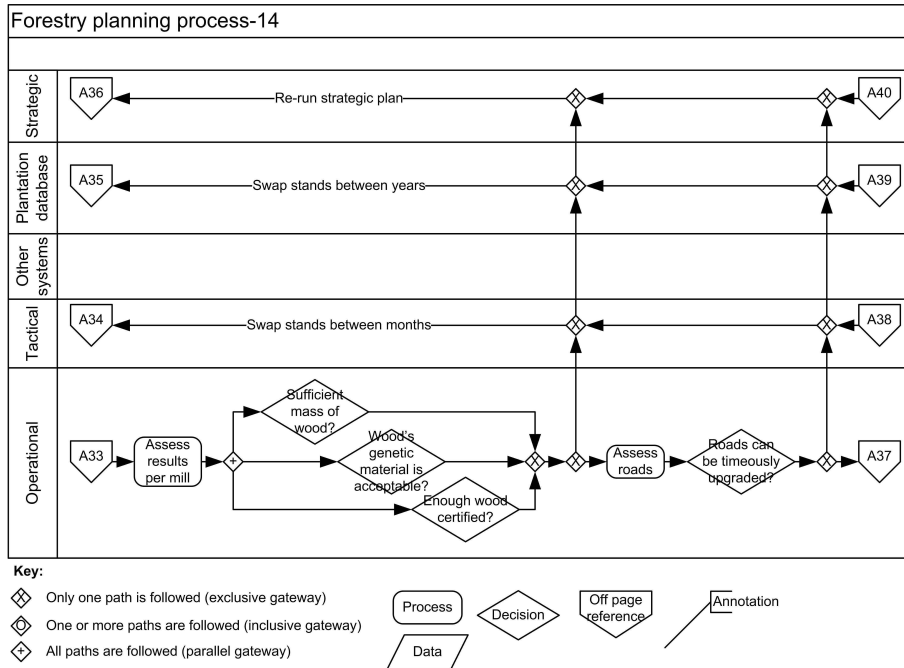


Figure 4.54: Business process diagram of the planning process (part 14 of 18)

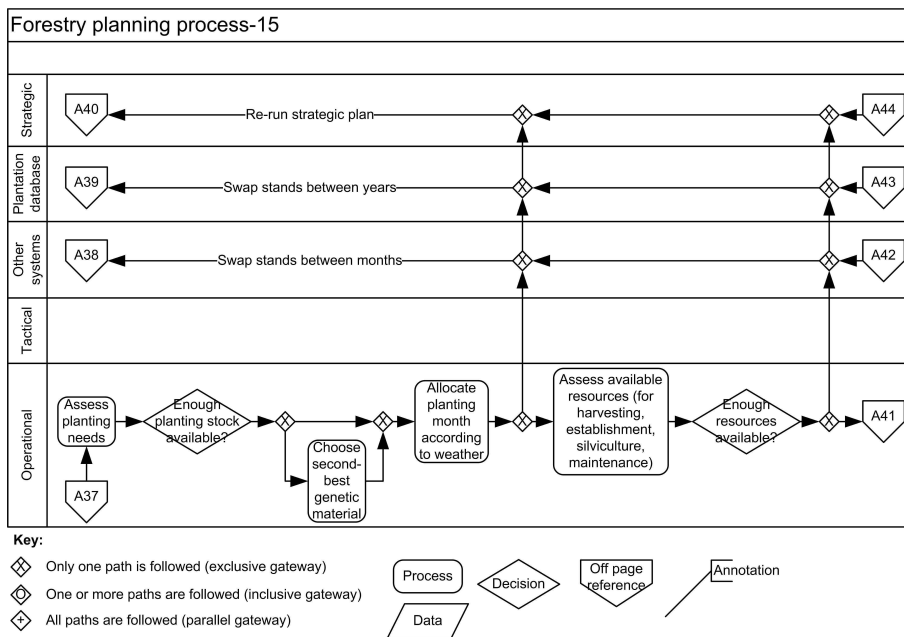


Figure 4.55: Business process diagram of the planning process (part 15 of 18)

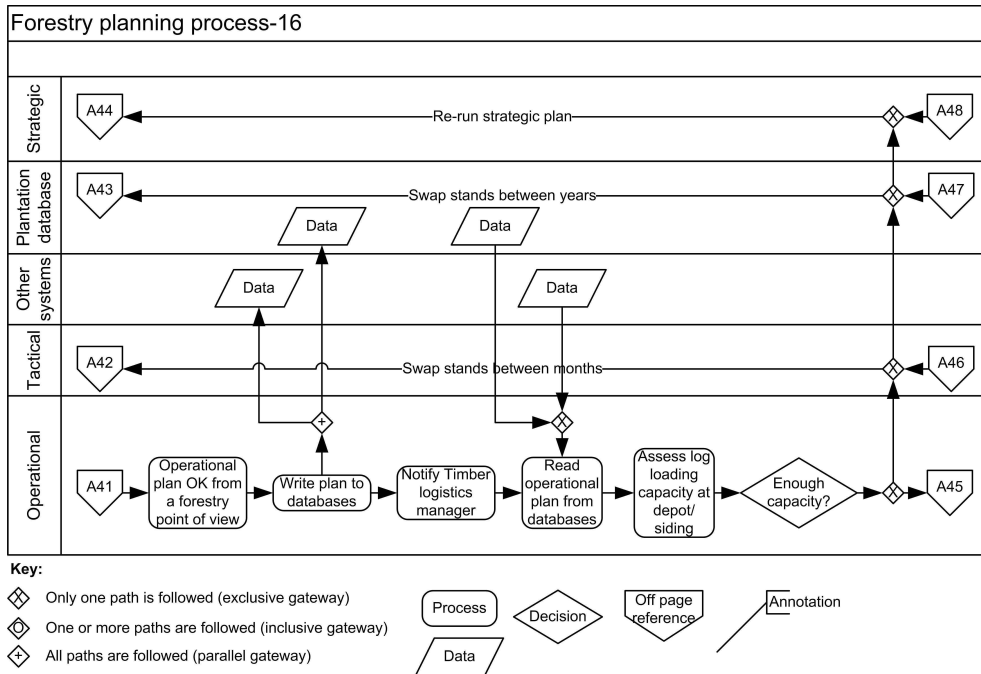


Figure 4.56: Business process diagram of the planning process (part 16 of 18)

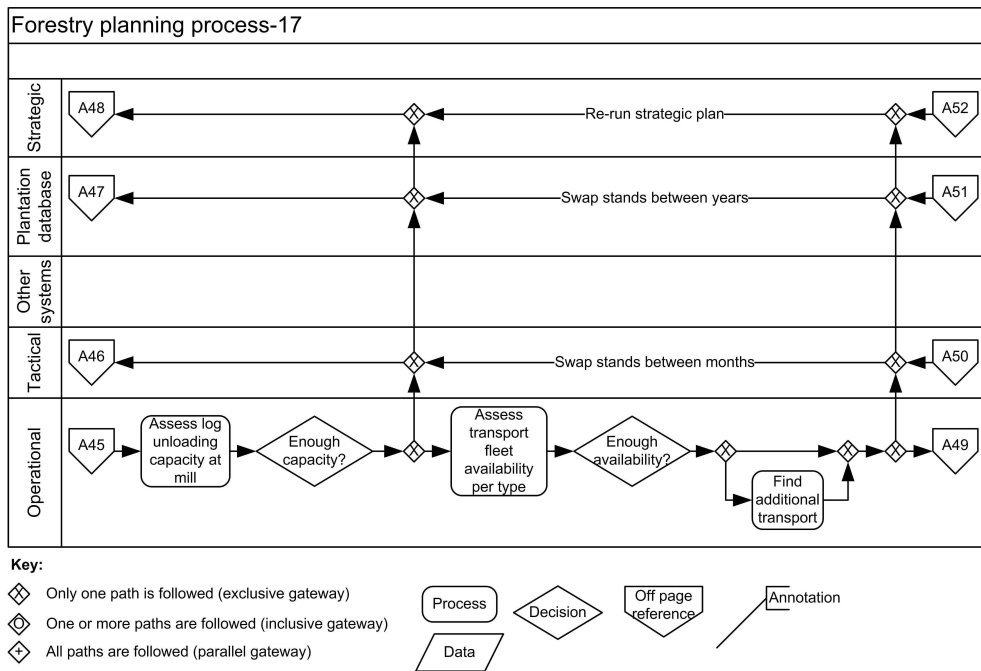


Figure 4.57: Business process diagram of the planning process (part 17 of 18)

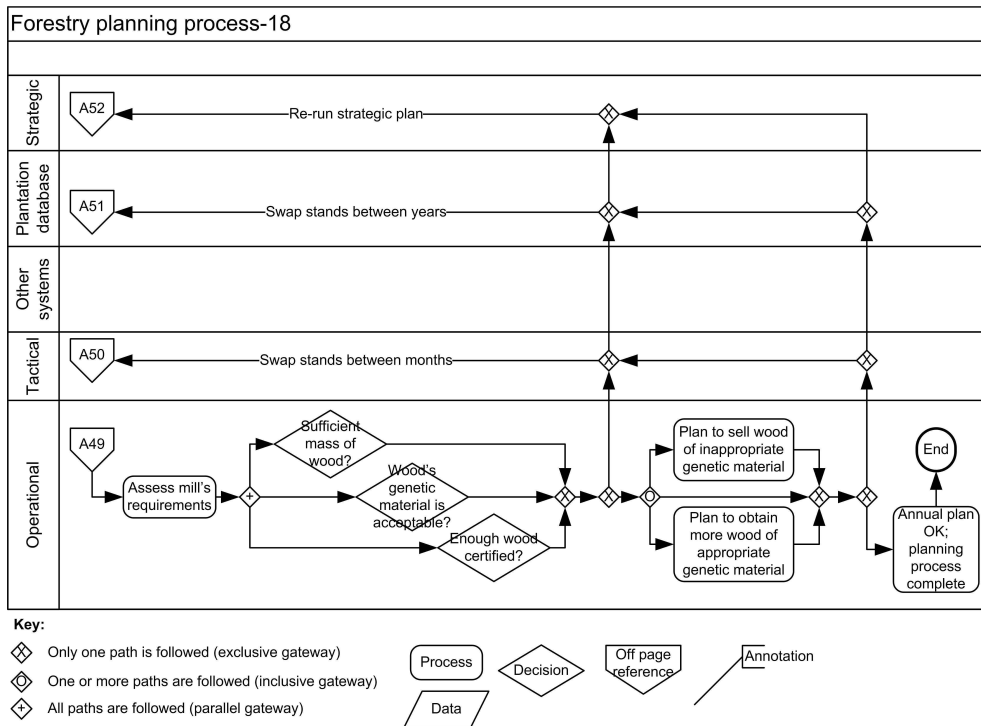


Figure 4.58: Business process diagram of the planning process (part 18 of 18)

swapped, between months, or years, or even that a new strategic plan be run. If this is not necessary, the planning phase is complete, and the proposed plan becomes the new plan.

4.7 Forest harvest scheduling system

In this section, the proposed forest harvest scheduling system is described. Firstly, the problem it is trying to solve is outlined in section 4.7.1. This includes the statement of the optimisation's objective function, the constraints and the decision variables. The system's context is then outlined in section 4.7.2. This shows the different databases, systems and users of the system, and at a summary level, what data flows between each. The inputs and outputs of the system (a necessary feature of a system's specification) are listed in section 4.7.3. Section 4.7.4 describes a typical run of the system, using a business process diagram. Pseudocode is used to clarify certain aspects. As wood properties, and their modelling, prediction and estimation are central to the system, they are described next (section 4.7.5). This covers how wood property models are generated and how they could be used to determine which logs enter the mill. Finally, section 4.7.6 covers the way in which business processes and decision-making would have to change when the proposed system is implemented. The diagrams which have been changed because of the inclusion of wood properties can be found in Appendix C. The system being described here has been patented (Turner and Price, 2005).

4.7.1 Aim of the system

The aim of the optimising forest harvest scheduling system is to maximise the profit of the forestry supply chain, while deciding which stands to harvest per annum, over the strategic planning horizon. It must also allocate the timber from each stand to the most suitable mill's process. If there is more than one long-haul transport option, the mode of transport which must be used to get the timber to the mill must also be allocated. This decision must be made so that each mill gets the required mass and species of timber (which has appropriate wood properties and certification status) annually, over the long-term planning horizon. The strategic plan developed must take into account tactical concerns, so that the plan is also feasible at a tactical level.

The decision variables for this system are therefore:

Stands to harvest p.a.

Mill and process to which timber from each harvested stand is to be sent

Long – haul transport method for each harvested stand

The forest harvest scheduling system decides each stand's harvesting age (between the minimum and maximum harvesting age) so that it will contribute to a mill's intake in the appropriate year. This is discussed more in section 4.7.4. It is assumed that stands will be planted as soon as possible after harvesting. The species to be planted and the regime to use are stored in the plantation database. It is also assumed that there is sufficient transport available for transporting the timber to the mills.

There are two options for the definition of the profit for the system. The first option is for the forest harvest scheduling system to optimise the profit for the forestry part of the supply chain (i.e. from the forest to the mill's logyard). In this case, the income is that which is accrued to the forestry part of the integrated forestry company. The second option is for the forest harvest scheduling system to optimise the profit for the whole supply chain. This means that the income is the value of sales generated by the mill for the pulp or pulp products sold. Both options would be useful to the integrated forestry company. The first would give the forestry part of the supply chain an indication of their costs, while the second would ensure that the whole supply chain's constraints were taken into account when making decisions.

Because the system optimises, there is a possibility that no feasible solution will exist. This is likely to stem from the fact that a mill needs logs which are not currently available. A way of overcoming this is to allow the system to "buy in" logs: these logs may not exist, but their mass, genetic material, wood properties and certification status will be listed by the system; this will help those using the system to adjust mill's process

constraints, or the Timber logistics manager to source such logs from other timber suppliers. Their cost could be entered into the system at the same price as the other logs cost, but it would be better to put in a higher price (i.e. it would be a disincentive or penalty for the system to choose these “bought in” logs over those produced by the plantation).

The objective function of the first option (forest to mill’s logyard) is:

$$\begin{aligned} \text{Maximise profit} &= \text{Income from selling logs to mills (own and others)} \\ &\quad - \text{log cost} - \text{cost of transporting logs to mill} \\ &\quad - \text{cost of timber to “buy in”} \end{aligned}$$

The log cost can be the internal transfer cost, or the sum of the forestry cost incurred to grow and make those logs, as discussed in section 4.3.6. If the integrated forestry company is selling logs to mills other than its own, the forestry cost to its own mills should be a proportional value based on the tonnes of timber delivered to its own mills.

The objective function for the second option (forest to pulp product) is:

$$\begin{aligned} \text{Maximise profit} &= \text{Income from selling pulp products} \\ &\quad - \text{log costs from own plantations} \\ &\quad - \text{cost of transporting own plantation's logs to mill} \\ &\quad - \text{cost of timber bought from other suppliers} \\ &\quad - \text{mill's fixed costs} - \text{cost of timber to “buy in”} \end{aligned}$$

The mill’s fixed costs are described in section 4.5.3. The benefit of using the second option, apart from addressing the whole forest-to-mill supply chain, is that one can sort the logs by wood properties which would increase their processing time (like rate of delignification), which could in turn increase the throughput of that process or mill.

The system's constraints are:

For each year in the strategic planning horizon,

the mill's process must receive the mass of logs required – between a user–specified lower and upper limit

the mill's process must receive the genetic material mix of logs required : a user–specified minimum and maximum mass of each genus / species / ...

the mass by genetic material cannot be greater than the amount specified without genetic material

the mill's process must receive logs having the required wood properties – between a user – specified lower and upper limit

(for optimisation option 2) : process capacity used is \leq the maximum process capacity available

for each mill, the percentage mass of logs entering the mill which were grown in certified stands \geq user – defined minimum percentage

for each transport method, there is a user – defined minimum and maximum percentage mass specified which should be transported with that method

the sum of each year's transport method's percentage = 100

for each chosen management level, there is a user–defined minimum area of stands due to be felled which do not have soils sensitive to compaction in wet weather

for each chosen management level, there is user–defined minimum area of stands due to be felled with slope \leq some constant (which indicates rapid harvesting conditions – e.g. 15 degrees)

all harvested trees are delivered to a mill (even if it is a “dummy” mill³, and the transport distance is 0)

each stand harvested has a transport link to a suitable mill

The problem which the optimising forest harvest scheduling system has to solve is

³This would be the case if a stand were burned in a fire.

large: a typical number of stands for an integrated plantation forestry company to own is 30 000; there are two transport methods used by South African companies. There are between five and 20 mills to consider. A typical strategic planning horizon is 30 years for fast-growing species.

4.7.2 System context

In this section, the proposed forest harvest scheduling system's context diagram is presented. This summarises the major databases, systems and people which will interact with the system, and what data will flow between them.

Figure 4.59 shows a context diagram of the proposed forest harvest scheduling system. There are three information sources: information about the plantation, the logistics and the mill. Information from these three sources is read into the system in order for the optimal solution to be calculated. The solution (the optimal strategic plan which maximises profits and meets the given constraints, or an optimal solution which has been edited by users to give more practicable results) is stored in the plantation database. The forest harvest scheduling system predicts the volumes of each stand at harvesting age by using a growth and yield system; similarly, wood property models are used to predict the average wood property of each stand at harvesting age. The optimal solution is calculated by an optimising solver.

While it is not the function of the forest harvest scheduling system to populate and maintain the three databases mentioned in the context diagram, it is assumed that the data in each of the databases is up-to-date and accurate. The Managing forester updates the plantation database with information about each activity which has taken place in the stands under his management. This means that there is a planned regime for each stand, as well as an actual regime, for stands which have been planted. The Co-ordinating and Managing foresters and the Timber logistics manager update the

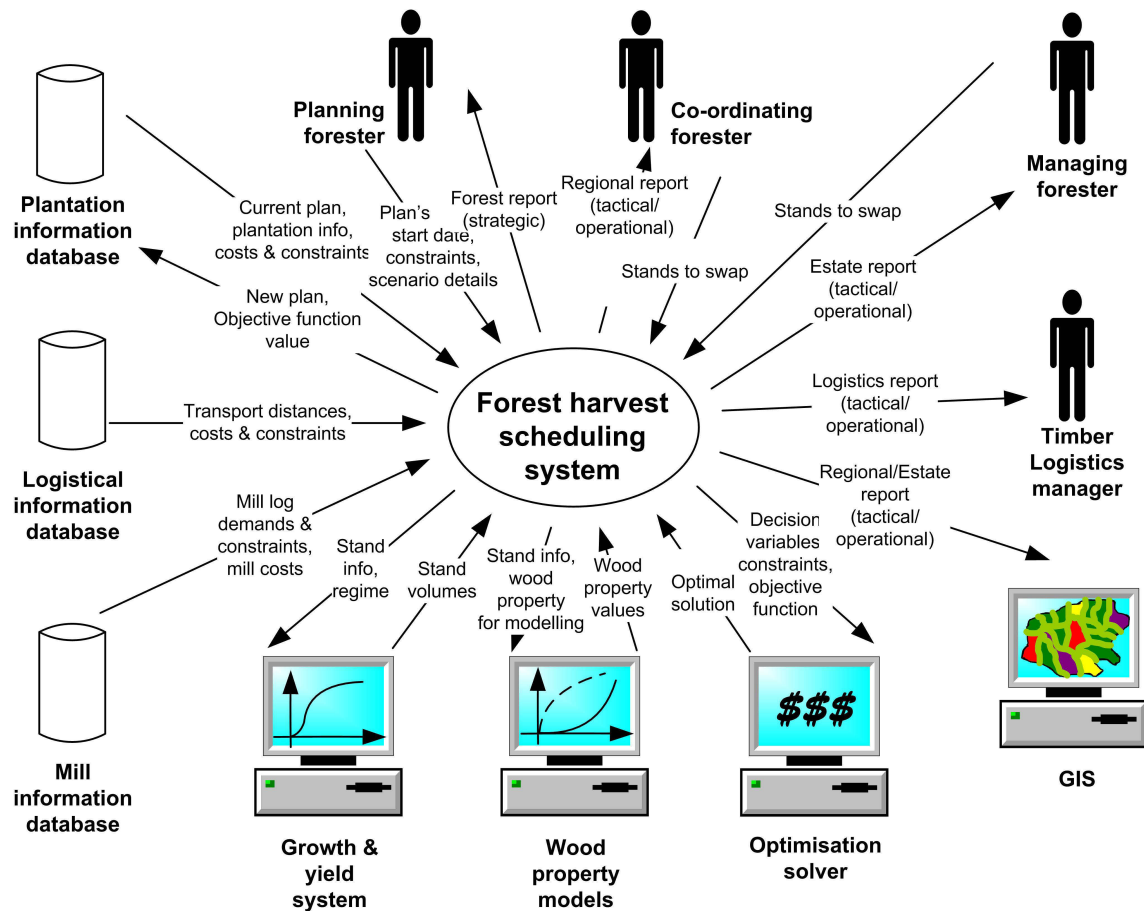


Figure 4.59: Context diagram giving an overview of the forest harvest scheduling system (FHSS)

databases with the constraints needed by the system.

The criteria used to accept plans at a strategic and tactical level are shown in section 4.6.4 in Figures 4.38, 4.39 and 4.40 (pages 111 to 113). To make the plan more practicable, the Co-ordinating and Managing foresters and the Timber logistics manager may change the optimal solution by swapping stands within the tactical planning horizon during the assessment of the plans (before the strategic plan is finalised), as well as during the implementation of the plans (after it has been finalised). In the latter case, if some circumstances have occurred which cause the plan not to be implementable, stands having similar wood properties and log mass may be swapped without the plan

having to be re-run.

Stands due for felling may be swapped if they are both in the tactical planning horizon. A limitation to this is that if the trees in the stand are at their maximum harvesting age, their felling may not be delayed. Similarly, if the trees in the stand are at their minimum felling age, their felling may not be bought forward. If a stand is due to be swapped, it must be replaced with a stand (or a number of stands) which have been allocated to the same mill and process, and which have a wood property value which is in the acceptable range for the mill's process.

Each kind of user needs to be able to draw a report at their level of responsibility. The Planning forester needs to obtain forestry-wide reports which will help him to confirm whether or not the plan is feasible at a strategic level. Age class distributions (see section 4.6.3 and Figure 4.33 on page 106) will be among the tools used by the Planning forester to determine whether the company's forests are normalised or not. The Co-ordinating and Managing foresters need to obtain region-wide and estate-wide reports detailing the actions to be undertaken in the tactical and operational planning horizons. Because the harvesting constraints at a tactical and operational level are mostly spatial in nature, the Co-ordinating and Managing foresters may want to review the output of the proposed plan using a Geographical Information System (GIS). The Timber logistics manager needs to obtain reports detailing the movement of timber and also what timber will be delivered to which mill, annually (for the strategic and tactical plans) and monthly (for the operational plan).

Because the forest harvest scheduling system uses optimization techniques, it may not be able to find a feasible optimal solution with the constraints entered by the user. A way of always obtaining an optimal solution is to allow the system to "buy in" timber (i.e. the system assumes that there is a sufficient supply of suitable timber available to be purchased). This "bought in" timber's cost is raised so that the system will not

choose to purchase this timber rather than using the timber grown in the plantation (i.e. this “buy-in” timber cost can be regarded as a penalty). Timber which the system “buys in” must be studied in detail by the Planning forester and the Timber logistics manager, as it gives a list of logs and log characteristics which is missing in the plantations. If such timber is not available on the open market, some constraints must be relaxed and the optimisation run again.

4.7.3 System inputs and outputs

In the literature review (in the section on the contents of a specification – section 2.6.3), it was noted that a system’s specification needs to give a list of the system’s inputs and outputs (IEEE Standards Board, 1994; DeMarco, 1997, pp.212–213). These inputs and outputs of the forest harvest scheduling system are listed in this section. This list expands on the inputs and outputs of the system which were drawn in the context diagram (Figure 4.59 on page 134).

4.7.3.1 System inputs

The following is a list of inputs for the forest harvest scheduling system, categorised according to types of inputs. It is assumed that the data in the relevant databases is up-to-date and accurate.

- Planning information
 - Plan start date
 - Strategic, tactical planning horizons (years)
- Forestry details
 - Stand’s current details:

- * Planned regime
 - * Actual regime
 - * Default or actual SI
 - * Stand's area for this rotation (is \leq stand's area, and is used when a part of the stand was affected by a disaster, e.g. fire, which effectively reduced the stand's area for the rotation)
 - * Percentage volume reduction for this rotation (is used when the trees in the stand are affected by a disaster, e.g. fire, which effectively reduced the stand's area for the rotation)
- Stand's planned details:
- * Planned regime for stand in subsequent rotations
 - * Planned genetic material for stand in subsequent rotations
 - * Default SI for that genetic material
- Stand's land
- * Stand's area
 - * Stand's average altitude
 - * Stand's slope
 - * Stand's aspect
 - * Stand's soil compact-sensitiveness
 - * Stand's terrain
 - * List of suitable "species" to grow in that stand
- Forestry management hierarchy
- * Ownership of a management level in the hierarchy (owned, leased, managed)
 - * Begin and end date of a management level in the hierarchy (e.g. lease starts on start date, and ends on end date, so stand/estate/... is available for planting trees on between those two dates)

- * Typical months when weather is wet for a management level
- * Management level which is managed according to certification principles (e.g. FSC)
- * Growth and yield model co-efficients to use for a particular end use (working circle), genetic material and management level
- * Costs
 - Average costs of regime actions per management level, end use (working circle) and genetic material
 - Overhead costs per estate, and at company level
- Wood properties
 - Wood property values to be obtained, per stand to be harvested
 - For each wood property to be estimated, the model co-efficients to use for a particular wood property and genetic material, OR
 - For each wood property of interest, the value measured in-field at time of enumeration
- Transport
 - Short-haul transport
 - * Distance from stand to depot/siding
 - * Cost per tonne per km for short-haul transport
 - Long-haul transport
 - * Transport methods available
 - * Distance from depot/siding to mill, for each transport method
 - * Cost per tonne per km for long-haul transport
- Mill
 - Number of processes per mill

– Process information

- * Products manufactured by process
- * Product sale price p.a.
- * Throughput, per process, p.a.
- * Recovery, per process, p.a. (can be calculated from a yield prediction model, which has as inputs: SI, age and genetic material of wood entering process)
- * For each process, the wood property which governs which logs are allocated to this process
- * For each process, whether or not logs whose wood properties are unknown (e.g. from an external timber supplier) can be allocated to this process

– Costs

- * Mill overhead costs, p.a.
- * Cost to purchase logs from plantation, p.a.
- * Cost to purchase logs from other suppliers, p.a.
- * Cost to transport timber from plantation, p.a.

● Constraints

– Which version of the system to run (forestry (option 1), or forest-to-mill (option 2))

– Mill constraints

- * The minimum and maximum mass of logs required, per process, p.a.
- * The minimum and maximum genetic material by mass required, per process, p.a.
- * The wood property, and minimum and maximum wood property value required, per process, p.a.
- * The minimum FSC percentage for the mill, p.a.

- * The maximum process capacity, p.a. (optimisation option 2)
- Transport constraints
 - * min and max percentage split of timber to be transport for each transport method, p.a.
- Forestry constraints
 - * For each year, at a user-chosen management level, there is a minimum area of stands to be harvested which do not have soils which are sensitive to compaction in wet weather
 - * For each year, at a user-chosen management level, there is a minimum area of stands to be harvested which has an average slope of x or less (where x denotes the upper limit of a “flat” stand)
 - * All wood from stands is sent to a mill (even if it’s a “dummy” mill)
- Stands to be swapped
 - IDs of the two (or more) stands to be swapped, together with each stand’s estimated volume and mass at harvesting, the wood property, the wood property value (measured or estimated), and the destination mill and process.

4.7.3.2 System outputs

- Harvesting decisions
 - Stands to harvest, p.a.
 - Expected volume (and mass) of timber per log type which will be produced from each stand
 - Genetic material of each stand
 - Wood property of importance, and estimated or measured wood property value
- Forestry decisions

- Regime actions for stand for each year in planning horizon
- Mill allocation decisions
 - Mill and process to which to send the timber from the stand
 - Mass of logs, their genetic material and average wood property value for those logs allocated from plantation's stands, per harvested stand, p.a.
 - Mass of timber allocated from other timber suppliers, and, if known, genetic material of logs and average wood property value of logs
- Transport
 - Mass of plantation forestry timber transported p.a., per transport type and transport method, together with origin and destination of trip
- Costs
 - Profit over planning horizon
 - Forestry costs per stand, per regime action, p.a.
 - Forestry income (optimisation option 1)
 - Transport costs for transporting plantation forestry timber p.a., per transport type and transport method, together with origin and destination of trip
 - Mill costs p.a.
 - Mill income (optimisation option 2)

4.7.4 Typical run of the system

A typical run of the forest harvest scheduling system (FHSS) is shown in Figures 4.60 and 4.61. At the start of the program, a choice is taken to create a new plan or edit an existing plan (see Figure 4.60). If a new plan is to be created, all the necessary data is read in. The user can edit constraints, if need be, as well as entering the starting date of the new plan, and the optimisation option to use. The FHSS then allocates

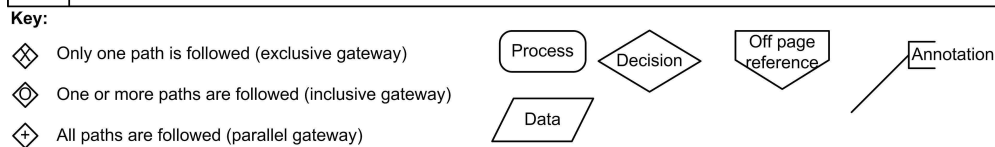
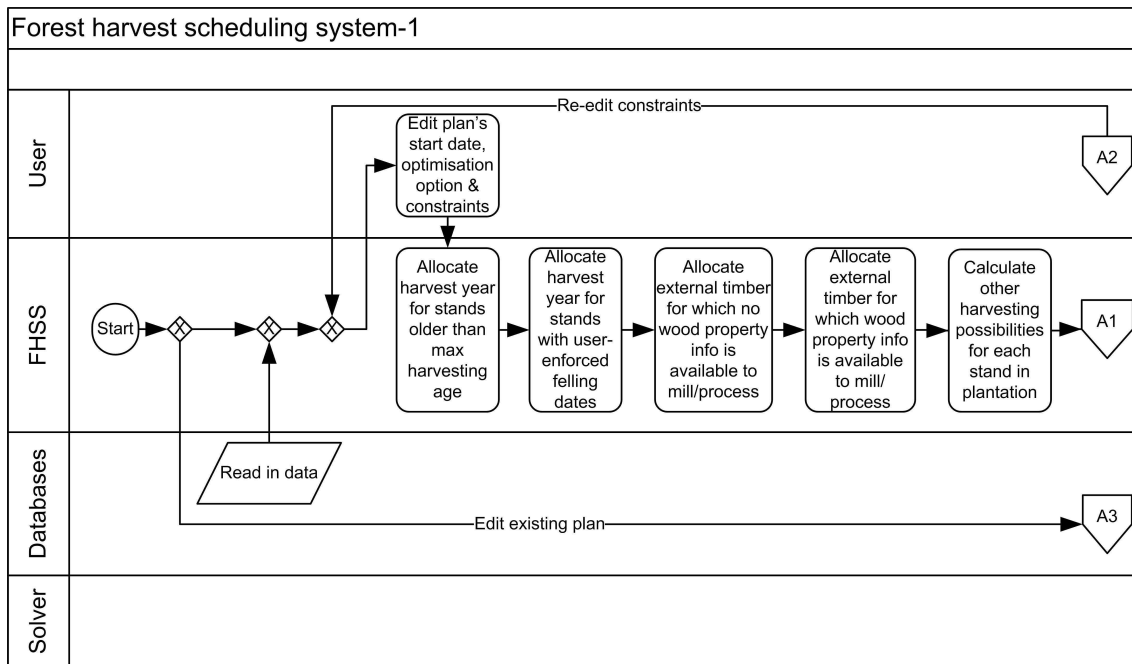


Figure 4.60: Business process diagram showing typical use of the FHSS (part 1 of 2)

harvesting dates to all stands of trees which are older than their maximum harvestable age (for more about regimes, and minimum and maximum harvesting ages, see section 4.3.4). It allocates harvesting (and other) dates which have been enforced by the user. It also allocates timber which is sold to the mill by an external supplier to a particular mill and process, and reduces that processes' demand by the amount allocated. If the site index, genetic material and age information⁴ of the logs' original stand are known, their wood properties can be calculated and the logs can be allocated to an appropriate mill's process. Otherwise, such logs need to be allocated to a process which is not very sensitive to variations in wood properties.

Next, other harvesting options are calculated for each stand, for the current regime

⁴Some models also require the stand's altitude to be known.

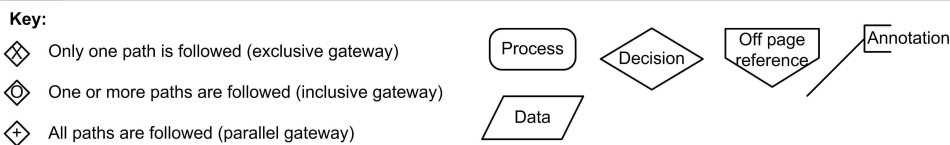
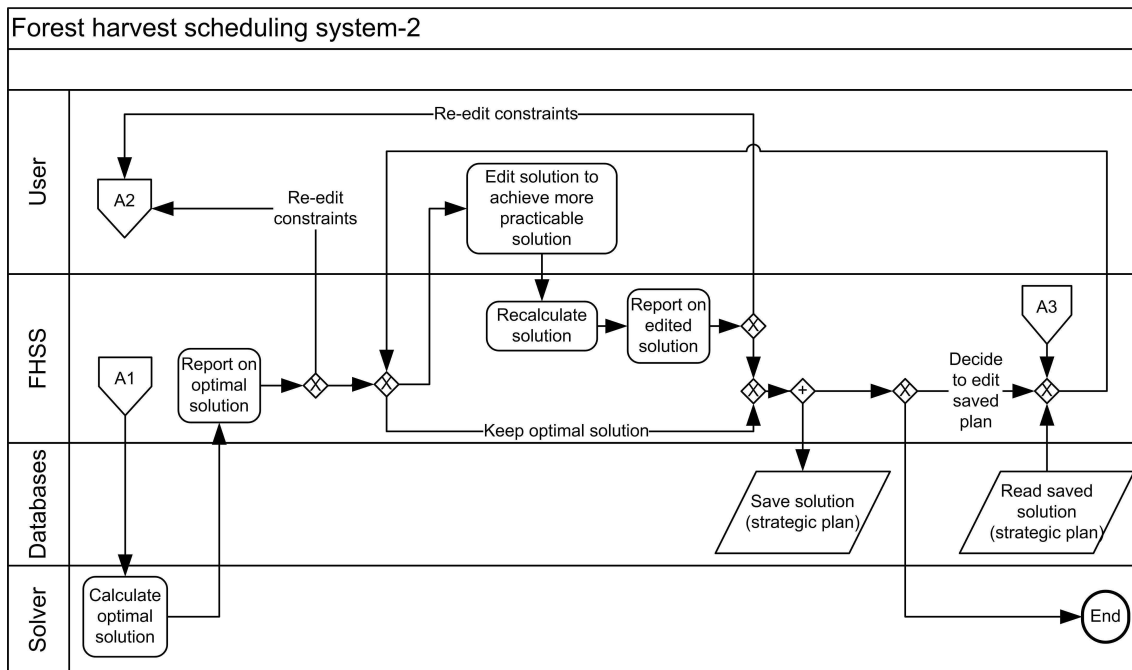


Figure 4.61: Business process diagram showing typical use of the FHSS (part 2 of 2)

(or, if the stand is currently temporarily unplanted, for the next rotation). This means that for each stand, the system must choose a harvesting date between the minimum harvesting age and the maximum harvesting age. The pseudocode excerpt in Table 4.10 (page 145) shows the logic of generating the harvesting options for each stand. Because of the combinatorial problem involved (see Figure 4.62), this will be done only for the current or first rotation. The subsequent rotations will be harvested by the system at the regime's rotation age. Unless specified otherwise in the plantation database, the subsequent rotations will be repeated until the strategic planning horizon is reached.

The activities which the FHSS undertakes in Figure 4.60 will populate the matrix for the optimisation solver prior to an optimisation run.

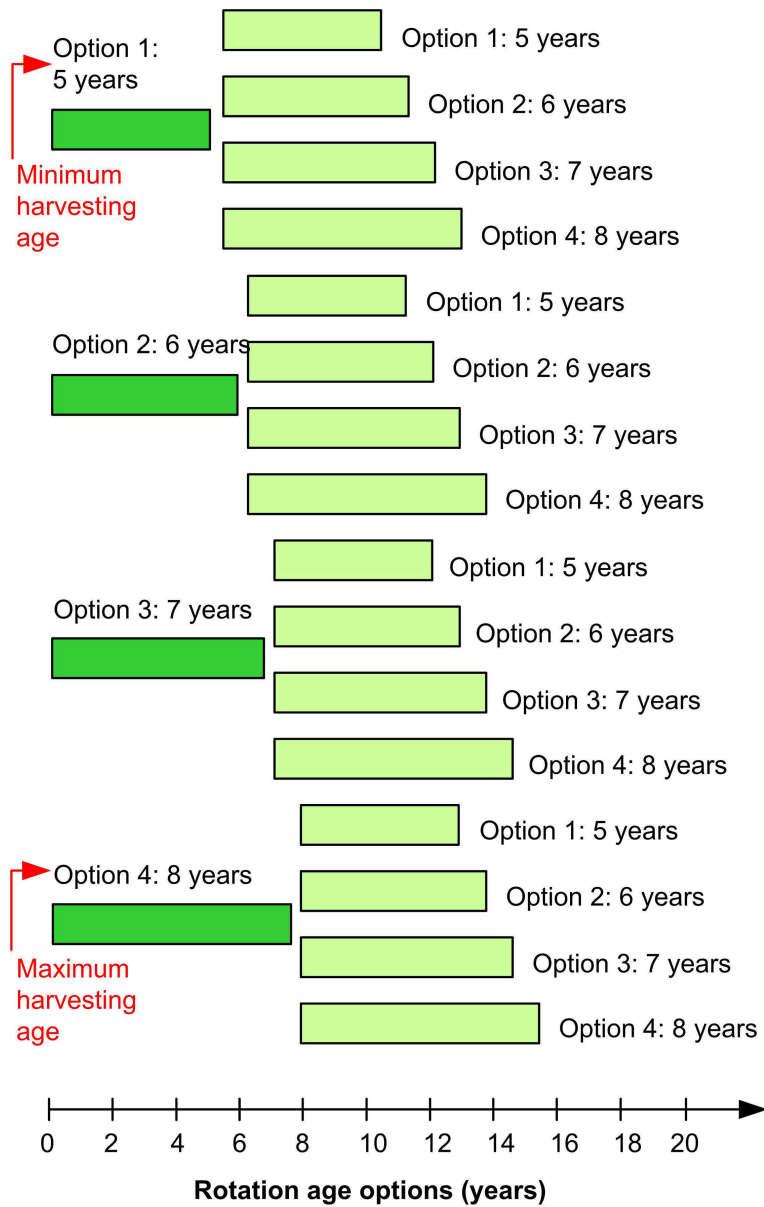


Figure 4.62: Diagram illustrating the combinatorial problem which solvers of the FHSS face: for one stand, four harvesting age options are shown, with only two rotations. For a strategic plan, another two rotations would be needed to fill the planning horizon.

Table 4.10: Pseudocode for working out possible harvesting options for each stand

```
if stand age  $\geq$  max harvesting age then  
    harvest stand immediately  
else  
    if stand has user – enforced harvesting date then  
        harvest stand in user – defined year  
    else  
        if (stand age < max harvesting age) and (stand age > min harvesting age) then  
            for i = current age to max harvesting age do  
                generate option to harvest stand at age i  
        else (if stand age  $\leq$  min harvesting age) or (stand temporarily unplanted) then  
            for i = min harvesting age to max harvesting age do  
                generate option to harvest stand at age i
```

Once all the harvesting alternatives are found, the solver is called to find the optimal solution (see Figure 4.61 on page 143). Once a solution is found, the optimal solution and the decision variables are reported on. The user may decide to save the plan as it is, or edit it to improve its practicability. The profit value and decision variables will be reported on. Note that if a user edits the optimal solution, it will no longer be optimal; in some way, the user has caused one or more constraints to be breached. The user may decide to go back and re-edit the constraints and re-run the optimisation, or he may save the resultant plan.

If the user decides to edit an existing plan (for example swap stands which have similar mass and wood properties, and which were destined for the same mill's process), the saved plan will be read in, and the program will allow the user to edit the plan (see A3 in Figure 4.61).

4.7.5 Wood property prediction and measurement

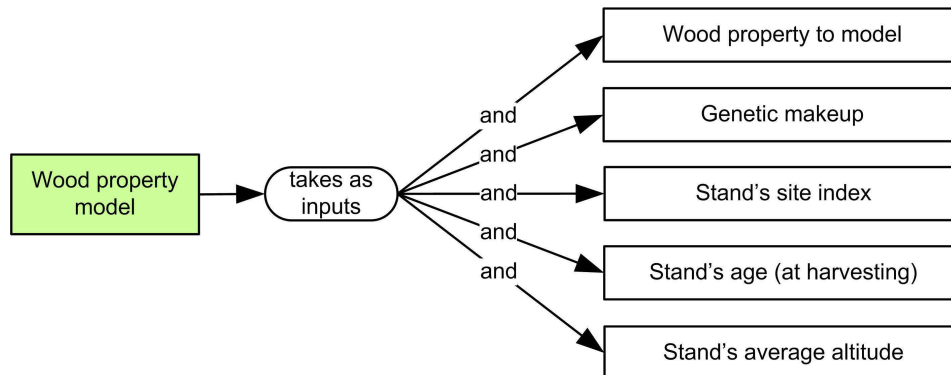


Figure 4.63: ERD showing the inputs to a wood property model

The term 'wood properties' is a generic term which applies to physical and chemical wood and fibre properties, as well as pulp and paper properties. Such models already developed by the Forestry and Forest Products Centre in Durban⁵ have as their inputs the wood property to be modelled, the genetic makeup (genus, species, ...) of the trees for which this model is applicable, the stand of trees' site index and the age at which the wood properties are to be predicted (usually harvesting age). Some models also have as input the stand's average altitude (see Figure 4.63). It is conceivable that as more research into the modelling of wood properties continues, other variables which are independent variables for predicting tree growth (such as those shown in Figure 4.9 on page 78) may also become input variables to wood property models.

A wood property model for a stand of trees can be developed by sampling a range of trees of a particular genus or species, over a wide range of site indices, ages and altitudes. The wood property or properties of these samples are measured and regression models obtained which can be used to predict the wood properties of the stand's trees. Models may be able to be developed at a genus level, in which case individual models

⁵A joint venture between CSIR and University of KwaZulu-Natal, P.O. Box 17001, 4013 Congella, South Africa

for species would not have to be developed.

An alternative method of assessing the wood properties of the stand would be to take core samples (or destructive samples) of the stand's trees just prior to harvesting (e.g. when the enumeration is being undertaken), and then measuring or inferring their wood properties. (Figure C.4 on page 256 shows when this would take place.) This approach would give a more accurate picture of the stand's wood property value.

Whichever method for measuring or predicting wood properties is used, it should be recognised that the wood property value obtained for the stand represents the average of the wide variation which exists between the trees in the stand and also the variation which exists within the tree. However, because the stand of trees has had a uniform silvicultural treatment, and all the trees have the same genetic source, assigning this wood property number for the stand of trees is a good way of allocating stands with similar properties to the appropriate processes, thus reducing the variation of the process' intake.

When implementing this system, the mill's management needs to decide which is the wood property, and its acceptable ranges, which will be used as the criterion for accepting logs into each mill's process. For example, if a mill had two processes, one could review the timber which is due to enter the mill for both processes over a certain period of time. Choosing a wood property which was deemed to be important for the mill's processes (e.g. wood density, tear, fibre length or pulp yield), one could determine the spread (variability) of that wood property's values. It may look like the wide distribution of wood property values in Figure 4.64). Depending on the mass requirements for both processes, one could separate the wood entering the mill and allocate it to the two processes. The variability of the wood entering each of the two processes would be reduced, as shown by the two narrower distributions in Figure 4.64).

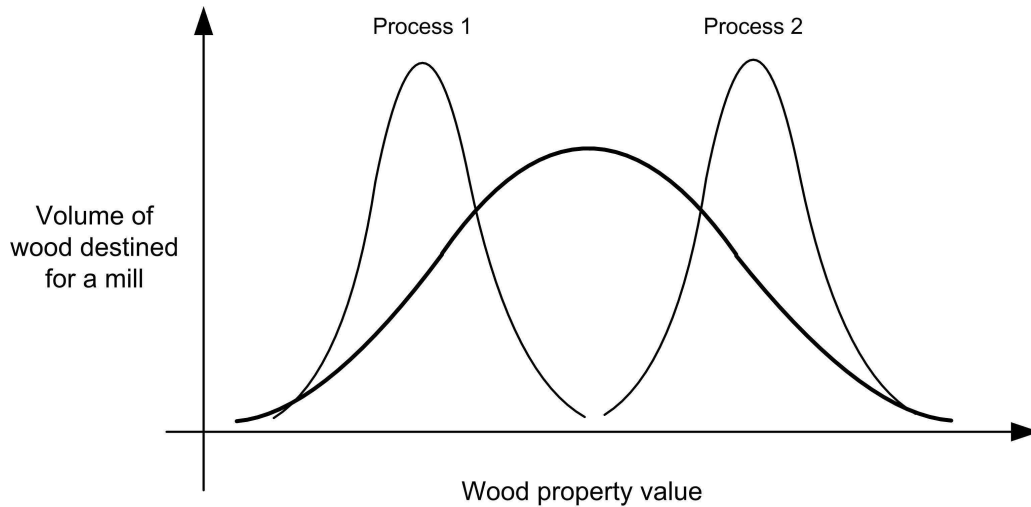


Figure 4.64: Graph showing the spread of wood property values entering a mill (wide distribution), and the effect of separating the wood according to wood property values for two different processes.

4.7.6 Impact of implementing system on business processes

In sections 4.2 to 4.6, the domains as they are at the moment were described. After the system has been implemented, however, certain aspects of the domain will change. The changes to these domains are recorded in detail in Appendix C. Section C.1 highlights when stands could be sampled for wood property measurement prior to harvesting (if wood properties were to be measured rather than modelled). Sections C.2 and C.3 show how the transport and mill domains would change as a result of implementing the system, respectively. Notable is the fact that in the depot or siding, timber would have to be stored (sorted) by mill's process and not just by the mill for which it was destined. At the mill, the logyard would have to have different piles for the different processes.

Implementing the system would also impact the planning process. Section C.4 shows how the strategic plan's contents would change, with the inclusion of wood property information. This wood property information is an additional acceptance criterion for the plan (see section C.5). Finally, the change in the planning process is shown in

section C.6 with business process diagrams. These show how wood produced by the plantation which has undesirable properties must be sold, and/or additional wood with suitable properties must be bought, and how the wood entering a particular process (rather than mill) must have suitable properties, in addition to the other criteria (log mass, genetic material and certification status).

4.8 Conclusion

This chapter used semi-formal methods to specify the forest harvest scheduling system. Because the system is embedded in many domains which affect and constrain it, the domains had to be identified and modelled before the system could be specified.

Four domains were identified, viz. the plantation forestry, transport, mill and forest planning domains. Actions and constraints were drawn up for each. The Zachman framework was used to structure the domain models. Entity-relationship diagrams (ERDs) and a glossary were used to describe the data (or things) in the domain. Business process models were used to describe the processes of the domain. These were developed to describe the stand's lifecycle and also the planning process. Hierarchical charts were used to describe the how the forest land is managed, and the reporting hierarchy of the forestry company. State charts were used to describe the forest-to-mill supply chain. ERDs were also used to describe the business rules. Using the Zachman framework was beneficial because one can cross-check the models and determine if aspects which needed to have been modelled were inadvertently left out.

The forest harvest scheduling system was then specified. As it is an optimising system, the system's objective function and constraints were defined. Two optional objective functions were specified, both aiming at maximising profit. The first optimises the forestry part of the company's profit, while the second objective function optimises the

profit of the forest-to-mill supply chain. Both use the stand's average wood properties in the harvesting and timber allocation decisions. The system specification includes a context diagram and a list of inputs and outputs.

The impact of implementing the forest harvest scheduling system on the domains was summarised at the end of this chapter. The main impact on current activities is that if a mill has more than one process, the timber would have to be sorted according to the mill's process (and transported separately). At the moment, the timber is being sorted according to its destination mill.

The next chapter develops specifications for the forest-to-mill domains and the forest harvest scheduling system using the formal notation Z.

Chapter 5

Formal specification

The aim of this dissertation is to present a specification of a forest harvest scheduling system which includes wood properties in the harvesting decision, using semi-formal and formal methods. The semi-formal specification of the system was presented in the previous chapter, while this chapter presents the results of the formal analysis of the forest-to-mill domain and the forest harvest scheduling system specification. The formal analysis is presented using the formal notation Z which is based on set theory and predicate logic.

As mentioned in the literature review (Chapter 2, section 2.6.3), the choice of abstraction level is important. Too little detail means that important information is left out. Too much detail may mean that the specification contains an implementation bias, which is undesirable (Morgan and Sufrin, 1993; Bowen and Hinchey, 1995b). As a result, an abstract version of the forest-to-mill and plantation forestry domains is presented first in Z notation (section 5.1). However, this specification was missing certain aspects of the forest-to-mill domain which were needed for the specification of the forest harvest scheduling system. A second, more detailed, abstraction of the forest-to-mill domain was then developed. Additions were necessary in preparation of the specification of the forest harvest scheduling system. The main changes between the two abstractions are highlighted in section 5.2; the full specification of the domain is given in Appendix D.

This chapter concludes with a description of a forest harvest scheduling system which includes wood properties in the harvesting decisions (section 5.3). For space reasons, important issues are highlighted in this chapter, while Appendix E contains the formal specification of the system. This specification relies on several aspects of the domain which were described in Appendix D. These aspects are referred to where necessary. Figure 5.1 shows the outline of this chapter and details the development of the forest harvest scheduling system specification, from its most abstract form to the final version.

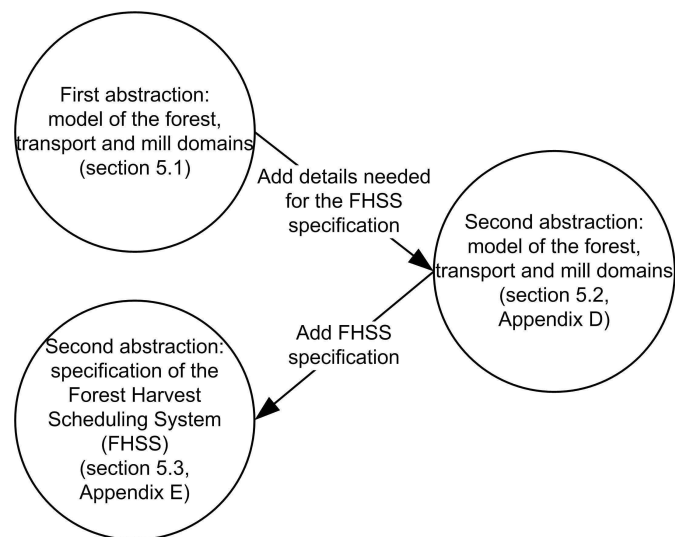


Figure 5.1: Development of the contents of this chapter, showing how the formal specification for the FHSS was developed

The Z specifications presented as the first abstraction of the forest-to-mill and plantation forestry domain (section 5.1) are based on the specification prepared for Price *et al.* (2009).

5.1 First abstraction: forest, transport, mill

5.1.1 Scope of the domain

This analysis starts with plantation forests and ends with timber arriving at a pulp mill. Papermaking is not considered. Figure 5.2 shows the main entities involved in this domain: the plantation forests, the transport and the pulp mill. The main actors are foresters (growers) (including planning, silvicultural and harvesting foresters), transporters (anyone involved with moving timber from one place to another, or the planning thereof), and millers/processors (anyone involved with receiving timber and processing it). Tables 4.3, 4.6 and 4.8 (on pages 69, 92 and 97, respectively) show the actions and constraints relevant to each of the main areas.

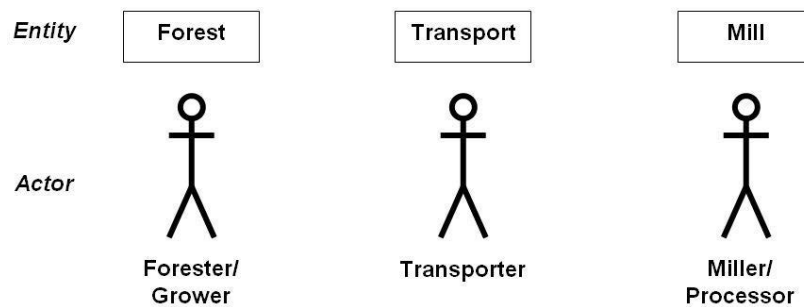


Figure 5.2: Main entities and actors of the domain

Figure 5.3 shows a state chart of the domain. This is a simplified version of the state chart shown in Figure 4.3 (page 68), as it only has one long-haul transport method (viz. road). Logs or tree-lengths are produced from the plantation on harvesting (felling) and are stacked at the stand's roadside; they are transported to the mill (via a depot) where they are processed to make pulp.

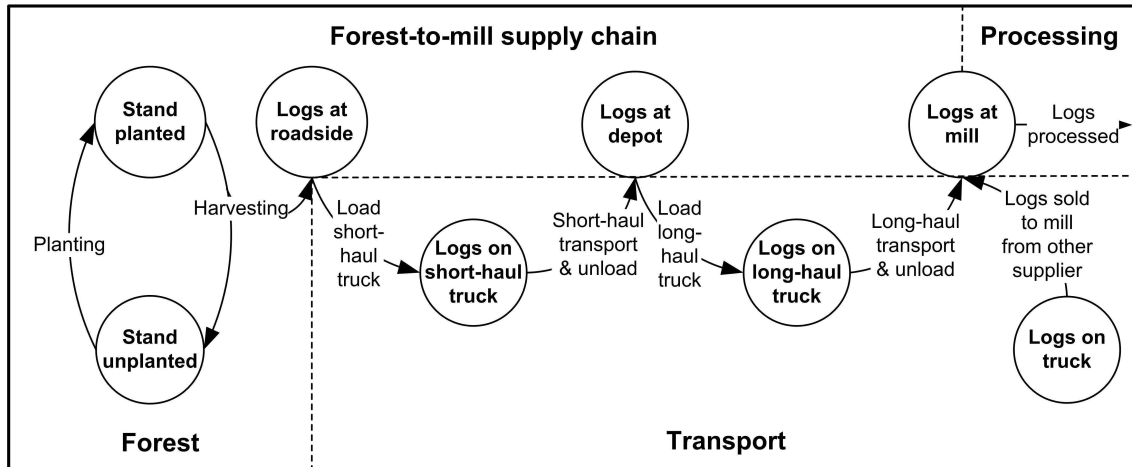


Figure 5.3: State chart of the forest-to-mill domain's scope modelled in the Z specification

5.1.2 Plantation forest

The plantation forest is divided into smaller management units, the smallest of which is the stand. A group of stands forms an estate. Stands are usually bounded by roads (Herbert, 2000). Although roads are sometimes only developed just prior to harvesting, basic quality roads are often built before planting (Evans and Turnbull, 2004). There is a depot located in or near each estate, to which harvested timber is transported. From there, it is transported to the mill.

The Z specification begins with the definitions needed. Each stand, mill and species is uniquely identified. The stand's planting state is either unplanted or planted. The type *AGE* describes the age of the trees planted in a stand; *MASS* describes the mass of timber harvested from the stand. *AGE* and *MASS* are defined here as a finite set of natural numbers (although they are actually real numbers; they are so defined because of a restriction in Z).

The type *MESSAGE* is defined to be able to give outputs in the exception schemas, should a stand need to be planted and it is already planted, should the stand be due

for harvesting and it is not planted, or should there not be sufficient logs in the mill's logyard to remove a load for processing.

$[STANDID, MILLID, SPECIESID]$

$PLANTINGSTATE ::= unplanted \mid planted$

$\mid AGE, MASS : \mathbb{FN}$

$MESSAGE ::= StandAlreadyPlanted \mid StandNotPlanted \mid NotEnoughLogs$

Two schemas follow: *StandSuitableSpecies* contains a function (*suitableSpecies*) which gives a list of suitable species which could be grown on each stand (determined by site-species matching). Each stand must have at least one suitable species. Since this specification describes an integrated forestry company, tree species would not be planted which were not acceptable to a mill. *MillAcceptableSpecies* contains the list of species (*acceptableSpecies*) acceptable to each mill. Each mill must have at least one acceptable species.

StandSuitableSpecies

$suitableSpecies : STANDID \rightarrow SPECIESID$

$\forall sID : STANDID \bullet$
 $sID \in \text{dom } suitableSpecies \wedge$
 $\#\{(suitableSpecies \ sID)\} \geq 1$

MillAcceptableSpecies

$acceptableSpecies : MILLID \rightarrow SPECIESID$

$\forall mID : MILLID \bullet$
 $mID \in \text{dom } acceptableSpecies \wedge$
 $\#\{(acceptableSpecies \ mID)\} \geq 1$

Schema *DefineTreeAge* defines the function *treeAge* which has as input the stand's ID and gives the trees' age as output. The stand's tree age is then initialised to the empty set in schema *InitTreeAge*. The two schemas are then combined to form

schema *TreeAge*. The function *treeAge* is assigned a value for the first time in schema *PlantStand*, when it is set to zero.

$$\frac{\text{DefineTreeAge}}{\text{treeAge} : \text{STANDID} \leftrightarrow \text{AGE}}$$

$$\frac{\text{InitTreeAge} \quad \Delta \text{DefineTreeAge}}{\text{treeAge}' = \emptyset}$$

$$\text{TreeAge} \hat{=} \text{DefineTreeAge} \wedge \text{InitTreeAge}$$

Schema *StandOfTrees* governs the relationships the stand's land and the trees that are planted on it.

$$\frac{\begin{array}{l} \text{StandOfTrees} \\ \exists \text{StandSuitableSpecies} \\ \exists \text{MillAcceptableSpecies} \\ \exists \text{TreeAge} \\ \text{plantingStatus} : \text{STANDID} \leftrightarrow \text{PLANTINGSTATE} \\ \text{plantedSpecies} : \text{STANDID} \leftrightarrow \text{SPECIESID} \end{array}}{\begin{array}{l} \forall sID : \text{STANDID} \bullet \exists mID : \text{MILLID} \bullet \\ sID \in \text{dom suitableSpecies} \wedge \\ mID \in \text{dom acceptableSpecies} \wedge \\ sID \in \text{dom plantingStatus} \wedge \\ sID \in \text{dom plantedSpecies} \wedge \\ sID \in \text{dom treeAge} \wedge \\ ((\text{plantingStatus } sID) = \text{planted} \Rightarrow \\ (\text{plantedSpecies } sID) \in \\ \{(suitableSpecies } sID)\}) \wedge \\ ((\text{plantedSpecies } sID) \in \\ \{(acceptableSpecies } mID)\}) \wedge \\ ((\text{plantingStatus } sID) = \text{planted} \Rightarrow \\ (\text{treeAge } sID) \geq 0) \wedge \\ ((\text{plantingStatus } sID) = \text{unplanted} \Rightarrow \\ \{(treeAge } sID)\} = \emptyset) \wedge \\ ((\text{plantingStatus } sID) = \text{unplanted} \Rightarrow \\ \{(plantedSpecies } sID)\} = \emptyset) \end{array}}$$

The schemas *StandSuitableSpecies*, *MillAcceptableSpecies* and *TreeAge* are included in schema *StandOfTrees*, but cannot be changed by it. Three functions are included: *plantingStatus* records whether the stand is unplanted or planted; and *plantedSpecies* records the species planted. If the stand is planted, the planted trees' species will be one of the stand's suitable species as well as one of the mills' acceptable species, and the trees' age will always be zero or above. If the stand is unplanted, the tree age and planted species will be undefined.

Prior to harvesting, the mill to which the stand's timber will be sent is determined and stored in function *millForStandsTimber* in schema *MillForStandsTimber*. Each stand has only one mill to which its timber will be sent.

<p><i>MillForStandsTimber</i></p> <p>$millForStandsTimber : STANDID \leftrightarrow MILLID$</p> <p>$\forall sID : STANDID \bullet \exists_1 mID : MILLID \bullet$ $sID \in \text{dom } millForStandsTimber \wedge$ $mID \in \text{ran } millForStandsTimber$</p>
--

Schema *MassOfFelledTrees* gives the mass of the stand's trees, when felled. The function *massOfFelledTrees*, which takes as inputs the stand's ID and that stand's harvesting age, gives the utilizable mass of the trees (i.e. the mass of the part of the trees which will eventually become logs). This mass is greater than or equal to zero.

<p><i>MassOfFelledTrees</i></p> <p>$massOfFelledTrees : (STANDID \times AGE) \leftrightarrow MASS$ $ageToFell? : AGE$</p> <p>$\forall sID : STANDID \bullet$ $(sID, ageToFell?) \in \text{dom } massOfFelledTrees \wedge$ $massOfFelledTrees (sID, ageToFell?) \geq 0$</p>
--

Once harvested, the stand's logs or tree-lengths are piled at roadside ready to be transported to the depot and then to the mill. Schema *DefineTimberAtRoadside*

contains information about the timber which is piled at roadside. It includes the unchangeable schema *MillForStandsTimber*, and a function, *timberAtRoadside*. This has as inputs the stand's ID and the mill's ID, and output the mass of timber. The mill's ID is that of the mill to which the stand's timber has been allocated. The mass of the timber at roadside is greater than or equal to zero.

In schema *InitTimberAtRoadside*, the function *timberAtRoadside* is initialised to the empty set. The schema *TimberAtRoadside* combines the two schemas *DefineTimberAtRoadside* and *InitTimberAtRoadside*. The contents of the function *timberAtRoadside* are changed in schema *HarvestStand*, where the stand's trees are harvested and extracted to roadside.

$$\begin{array}{l}
 \hline
 \textit{DefineTimberAtRoadside} \\
 \exists \textit{MillForStandsTimber} \\
 \textit{timberAtRoadside} : (\textit{STANDID} \times \textit{MILLID}) \mapsto \textit{MASS} \\
 \hline
 \forall sID : \textit{STANDID} \bullet \exists_1 mID : \textit{MILLID} \bullet \\
 \quad sID \in \text{dom } \textit{millForStandsTimber} \wedge \\
 \quad mID \in \text{ran } \textit{millForStandsTimber} \wedge \\
 \quad (sID, mID) \in \text{dom } \textit{timberAtRoadside} \wedge \\
 \quad \textit{millForStandsTimber } sID = mID \wedge \\
 \quad \textit{timberAtRoadside } (sID, (\textit{millForStandsTimber } sID)) \geq 0 \\
 \hline
 \end{array}$$

$$\begin{array}{l}
 \hline
 \textit{InitTimberAtRoadside} \\
 \Delta \textit{DefineTimberAtRoadside} \\
 \textit{timberAtRoadside}' = \emptyset \\
 \hline
 \end{array}$$

$$\textit{TimberAtRoadside} \hat{=} \textit{DefineTimberAtRoadside} \wedge \textit{InitTimberAtRoadside}$$

The action schema *PlantStandOK* describes the planting activities. This schema includes unchangeable schemas, *StandSuitableSpecies* and *MillAcceptableSpecies*, and changeable schema *StandOfTrees*. Inputs to this schema are *whichStand?* (the ID of the stand to be planted) and *speciesToPlant?* (the species to be planted). The species to be planted must be in the list of suitable species, and a mill must exist for which the species to be planted is in the list of acceptable species. The stand's status

prior to planting must be unplanted. After planting, the planting status becomes planted, the trees' age changes to zero and the planted species is assigned the value of *speciesToPlant?*.

<i>PlantStandOK</i>
\exists <i>StandSuitableSpecies</i> \exists <i>MillAcceptableSpecies</i> Δ <i>StandOfTrees</i> <i>whichStand?</i> : <i>STANDID</i> <i>speciesToPlant?</i> : <i>SPECIESID</i>
<i>whichStand?</i> \in dom <i>suitableSpecies</i> <i>whichStand?</i> \in dom <i>plantingStatus</i> <i>whichStand?</i> \in dom <i>plantedSpecies</i> <i>whichStand?</i> \in dom <i>treeAge</i> <i>speciesToPlant?</i> \in ran <i>suitableSpecies</i> <i>speciesToPlant?</i> \in ran <i>plantedSpecies</i> <i>speciesToPlant?</i> \in { <i>suitableSpecies whichStand?</i> } \exists <i>mID</i> : <i>MILLID</i> • <i>mID</i> \in dom <i>acceptableSpecies</i> \wedge <i>speciesToPlant?</i> \in { <i>acceptableSpecies mID</i> } (<i>plantingStatus whichStand?</i>) = <i>unplanted</i> <i>plantingStatus'</i> = <i>plantingStatus</i> \oplus { <i>whichStand?</i> \mapsto <i>planted</i> } <i>treeAge'</i> = <i>treeAge</i> \oplus { <i>whichStand?</i> \mapsto 0 } <i>plantedSpecies'</i> = <i>plantedSpecies</i> \oplus { <i>whichStand?</i> \mapsto <i>speciesToPlant?</i> }

The schema *ExceptionPlantStand* defines the exception which will occur if the stand which is to be planted is already planted. The schema includes the unchangeable schema *StandOfTrees*, the input *whichStand?* and the output *message!*. If the stand *whichStand?* which is due to be planted, already contains trees, an error message is output to say that the stand is already planted.

<i>ExceptionPlantStand</i>
\exists <i>StandOfTrees</i> <i>whichStand?</i> : <i>STANDID</i> <i>message!</i> : <i>MESSAGE</i>
<i>whichStand?</i> \in dom <i>plantingStatus</i> <i>plantingStatus whichStand?</i> = <i>planted</i> \Rightarrow <i>message!</i> = <i>StandAlreadyPlanted</i>

The schema *PlantStand* is made up of either *PlantStandOK* or *ExceptionPlantStand*.

$$PlantStand \hat{=} PlantStandOK \vee ExceptionPlantStand$$

The action schema for harvesting (felling) the stand (*HarvestStandOK*) includes changeable schemas *StandOfTrees* and *TimberAtRoadside*, and unchangeable schema *MassOfFelledTrees*. It has two inputs, *whichStand?* (the stand to fell) and *fellingAge?* (the age at which to fell the stand). Before harvesting, the planting status must be planted and the stand's age must be the same as *fellingAge?*. After felling, the logs/tree-lengths are piled at roadside: their mass takes the value of the function *massOfFelledTrees* (evaluated at *whichStand?* and *fellingAge?*). The stand's planting status becomes unplanted, and the planted species and tree age become undefined.

<i>HarvestStandOK</i>
$\Delta StandOfTrees$
$\Delta TimberAtRoadside$
$\Xi MassOfFelledTrees$
<i>whichStand?</i> : <i>STANDID</i>
<i>fellingAge?</i> : <i>AGE</i>
$whichStand? \in \text{dom } suitableSpecies$
$whichStand? \in \text{dom } plantingStatus$
$whichStand? \in \text{dom } plantedSpecies$
$whichStand? \in \text{dom } treeAge$
$fellingAge? \in \text{ran } treeAge$
$(whichStand?, fellingAge?) \in \text{dom } massOfFelledTrees$
$plantingStatus \text{ whichStand?} = planted$
$fellingAge? = (treeAge \text{ whichStand?})$
$timberAtRoadside' = timberAtRoadside \oplus$
$\{(whichStand?, millForStandsTimber \text{ whichStand?}) \mapsto$
$massOfFelledTrees (whichStand?, fellingAge?)\}$
$plantingStatus' = plantingStatus \oplus$
$\{whichStand? \mapsto unplanted\}$
$\{(plantedSpecies' \text{ whichStand?})\} = \emptyset$
$\{(treeAge' \text{ whichStand?})\} = \emptyset$

The schema *ExceptionHarvestStand* defines the exception which will occur if a stand which is due to be harvested contains no planted trees. The schema includes the unchangeable schema *StandOfTrees*, the input *whichStand?* and the output *message!*.

If the stand *whichStand?* which is due to be harvested, contains no trees, an error message is output to say that the stand is unplanted.

$\begin{array}{l} \textit{ExceptionHarvestStand} \\ \exists \textit{StandOfTrees} \\ \textit{whichStand?} : \textit{STANDID} \\ \textit{message!} : \textit{MESSAGE} \end{array}$
$\begin{array}{l} \textit{whichStand?} \in \text{dom } \textit{plantingStatus} \\ \textit{plantingStatus } \textit{whichStand?} = \textit{unplanted} \Rightarrow \\ \textit{message!} = \textit{StandNotPlanted} \end{array}$

The schema *HarvestStand* is made up of either *HarvestStandOK* or *ExceptionHarvestStand*.

$$\textit{HarvestStand} \hat{=} \textit{HarvestStandOK} \vee \textit{ExceptionHarvestStand}$$

ForestSchema combines the two forestry action schemas *PlantStand* and *HarvestStand*.

$$\textit{ForestSchema} \hat{=} \textit{PlantStand} \wedge \textit{HarvestStand}$$

5.1.3 Transport

The distances from the depot to the mill are sometimes long, as the forest stands extend over vast areas, and the forestry areas are not necessarily near the mills. Long-haul lorries take the timber to the mills; they may not carry more than their maximum load, as overloading would damage the road's surface.

Table 4.6 (page 92) shows the actions and constraints which apply to the long-haul transporter. Main concerns are being able to transport a load for as much of each 24 hour period as possible, and not wanting to wait for people/equipment to be able to

load and unload.

After harvesting, the timber is transported (short-haul) from the stand's roadside to the depot. From there, it is transported (long-haul) to the mill.

The transport section of this description starts by identifying each depot, truck and truck's trip.

$[DEPOTID, TRUCKID, TRIPID]$

Schema *DefineDepot* describes the depot which is in or near each estate. The schema includes a function, *timberAtDepot*, which has as input the depot's ID and the mill's ID (for which the timber is destined), and outputs the mass of timber in that pile. Each timber pile has a mass greater than or equal to zero.

$\begin{array}{l} \textit{DefineDepot} \\ \textit{timberAtDepot} : DEPOTID \times MILLID \leftrightarrow MASS \\ \forall dID : DEPOTID \bullet \exists mID : MILLID \bullet \\ \quad (dID, mID) \in \text{dom } \textit{timberAtDepot} \wedge \\ \quad \textit{timberAtDepot} (dID, mID) \geq 0 \end{array}$
--

In schema *InitDepot*, the function *timberAtDepot* is initialised to the empty set. In schema *NoLogsAtDepot* the mass of logs at the depot destined for each mill is set to zero. The schema *Depot* combines these three schemas (*DefineDepot*, *InitDepot* and *NoLogsAtDepot*).

$\begin{array}{l} \textit{InitDepot} \\ \Delta \textit{DefineDepot} \\ \textit{timberAtDepot}' = \emptyset \end{array}$

$\frac{\text{NoLogsAtDepot} \quad \Delta \text{InitDepot}}{\forall dID : \text{DEPOTID} \bullet \exists mID : \text{MILLID} \bullet \\ (dID, mID) \in \text{dom } \text{timberAtDepot} \wedge \\ \text{timberAtDepot}' = \text{timberAtDepot} \cup \{((dID, mID), 0)\}}$
--

$$\text{Depot} \hat{=} \text{DefineDepot} \wedge \text{InitDepot} \wedge \text{NoLogsAtDepot}$$

Schema *DepotForStandsTimber* records the depot to which the timber from a particular stand is to be taken. It includes two unchangeable schemas, *MillForStandsTimber* and *Depot*, and a function, *depotForStandsTimber*, which has as input the stand's ID and outputs the destination depot's ID. Each stand has only one destination depot to which its timber will be sent.

$\frac{\text{DepotForStandsTimber} \quad \exists \text{MillForStandsTimber} \quad \exists \text{Depot} \quad \text{depotForStandsTimber} : \text{STANDID} \leftrightarrow \text{DEPOTID}}{\forall sID : \text{STANDID} \bullet \exists_1 dID : \text{DEPOTID} \bullet \\ \exists_1 mID : \text{MILLID} \bullet \\ sID \in \text{dom } \text{millForStandsTimber} \wedge \\ mID \in \text{ran } \text{millForStandsTimber} \wedge \\ (dID, mID) \in \text{dom } \text{timberAtDepot} \wedge \\ sID \in \text{dom } \text{depotForStandsTimber} \wedge \\ dID \in \text{ran } \text{depotForStandsTimber} \wedge \\ \#\{\text{depotForStandsTimber } sID\} = 1}$
--

Schema *Truck* includes two functions, *thisLoad* and *maxLoad*. *thisLoad* gives the mass of the timber in each truck's load (for each trip); *maxLoad* gives the maximum mass that the truck is allowed to carry. Any truck's load may never exceed the maximum allowable load, as this would damage the road's surface. The truck's load is greater than or equal to zero.

<i>Truck</i>
$ \begin{aligned} & \text{thisLoad} : (\text{TRUCKID} \times \text{TRIPID}) \rightarrow \text{MASS} \\ & \text{maxLoad} : \text{TRUCKID} \rightarrow \text{MASS} \end{aligned} $
$ \begin{aligned} & \forall \text{trID} : \text{TRUCKID} \bullet \\ & \quad \exists \text{tripID} : \text{TRIPID} \bullet \\ & \quad (\text{trID}, \text{tripID}) \in \text{dom thisLoad} \wedge \\ & \quad \text{trID} \in \text{dom maxLoad} \wedge \\ & \quad (\text{thisLoad} (\text{trID}, \text{tripID})) \leq (\text{maxLoad trID}) \wedge \\ & \quad (\text{thisLoad} (\text{trID}, \text{tripID})) \geq 0 \end{aligned} $

A set containing trip IDs, *tripIDs*, is defined next in schema *DefineTripIDs* to ensure that each trip is unique. The set of trip IDs is initialised in schema *InitTripIDs* to the empty set. Schema *TripIDs* then combines the two schemas *DefineTripIDs* and *InitTripIDs*.

<i>DefineTripIDs</i>
$\text{tripIDs} : \mathbb{F} \text{TRIPID}$

<i>InitTripIDs</i>
$\Delta \text{DefineTripIDs}$
$\text{tripIDs}' = \emptyset$

$$\text{TripIDs} \hat{=} \text{DefineTripIDs} \wedge \text{InitTripIDs}$$

The truck's short-haul trips are defined in schema *ShortHaulTrip*. This schema includes the unchanged schema *TripIDs*, and defines a function, *shortHaulTrip*, which has as inputs the truck's ID and the trip's ID, and outputs the ID of the stand where the trip starts and the ID of the depot where the trip ends. For each truck and trip combination, there is only one short-haul trip.

<i>ShortHaulTrip</i>
$ \begin{aligned} & \exists \text{TripIDs} \\ & \text{shortHaulTrip} : (\text{TRUCKID} \times \text{TRIPID}) \rightarrow (\text{STANDID} \times \text{DEPOTID}) \end{aligned} $
$ \begin{aligned} & \forall \text{trID} : \text{TRUCKID} \bullet \exists_1 \text{tripID} : \text{TRIPID} \bullet \\ & \quad \text{tripID} \in \text{tripIDs} \wedge \\ & \quad (\text{trID}, \text{tripID}) \in \text{dom shortHaulTrip} \wedge \\ & \quad \#\{(\text{shortHaulTrip}(\text{trID}, \text{tripID}))\} = 1 \end{aligned} $

The schema *LoadTruckAtStand* governs the addition of a load of timber to particular truck at the stand's roadside.

$$\begin{array}{l}
\text{--- } \underline{LoadTruckAtStand} \text{ ---} \\
\exists MillForStandsTimber \\
\exists DepotForStandsTimber \\
\Delta TimberAtRoadside \\
\Delta Truck \\
\Delta TripIDs \\
\Delta ShortHaulTrip \\
\hline
\forall sID : STANDID \bullet \\
\quad \exists_1 dID : DEPOTID \bullet \\
\quad \exists_1 mID : MILLID \bullet \\
\quad \exists trID : TRUCKID \bullet \\
\quad \exists tripID : TRIPID \bullet \\
\quad sID \in \text{dom } millForStandsTimber \wedge \\
\quad mID \in \text{ran } millForStandsTimber \wedge \\
\quad sID \in \text{dom } depotForStandsTimber \wedge \\
\quad dID \in \text{ran } depotForStandsTimber \wedge \\
\quad (sID, mID) \in \text{dom } timberAtRoadside \wedge \\
\quad (trID, tripID) \in \text{dom } thisLoad \wedge \\
\quad trID \in \text{dom } maxLoad \wedge \\
\quad (trID, tripID) \in \text{dom } shortHaulTrip \wedge \\
\quad (sID, dID) \in \text{ran } shortHaulTrip \wedge \\
\quad tripID \notin tripIDs \wedge \\
\quad (thisLoad (trID, tripID)) = 0 \wedge \\
\quad (timberAtRoadside (sID, mID)) \geq (maxLoad trID) \wedge \\
\quad tripIDs' = tripIDs \cup \{tripID\} \wedge \\
\quad shortHaulTrip' = shortHaulTrip \cup \{(trID, tripID) \mapsto (sID, dID)\} \wedge \\
\quad thisLoad' = thisLoad \oplus \{(trID, tripID) \mapsto (maxLoad trID)\} \wedge \\
\quad timberAtRoadside' = timberAtRoadside \oplus \{(sID, mID) \mapsto \\
\quad \quad ((timberAtRoadside (sID, mID)) - (maxLoad trID))\} \wedge \\
\quad maxLoad' = maxLoad
\end{array}$$

Schema *LoadTruckAtStand* includes the unchangeable schemas *MillForStandsTimber* and *DepotForStandsTimber*, and the changeable schemas *TimberAtRoadside*, *Truck*, *TripIDs* and *ShortHaulTrip*. Before the trip, the truck is empty and there is more than a truck load of timber at roadside. The ID of the trip to the depot does not exist before the trip is made, and it is added to the set *tripIDs* (as a valid trip) once the trip is begun. The function *shortHaulTrip* is also updated with this trip's details. If the mass of timber at roadside is greater than or equal to the truck's maximum load,

the truck will be loaded with the maximum load it can carry and the mass of timber at roadside is diminished by the mass loaded. The function *maxLoad* is unchanged by this schema.

The schema *TransportAndUnloadTruckAtDepot* describes the transportation and unloading of a load of timber from a particular truck at the depot.

$$\begin{array}{l}
 \text{--- } \underline{\text{TransportAndUnloadTruckAtDepot}} \text{ ---} \\
 \exists \text{MillForStandsTimber} \\
 \exists \text{DepotForStandsTimber} \\
 \exists \text{TripIDs} \\
 \exists \text{ShortHaulTrip} \\
 \Delta \text{Truck} \\
 \Delta \text{Depot} \\
 \hline
 \forall sID : \text{STANDID} \bullet \\
 \quad \exists_1 dID : \text{DEPOTID} \bullet \\
 \quad \exists_1 mID : \text{MILLID} \bullet \\
 \quad \exists trID : \text{TRUCKID} \bullet \\
 \quad \exists tripID : \text{TRIPID} \bullet \\
 \quad sID \in \text{dom millForStandsTimber} \wedge \\
 \quad mID \in \text{ran millForStandsTimber} \wedge \\
 \quad sID \in \text{dom depotForStandsTimber} \wedge \\
 \quad dID \in \text{ran depotForStandsTimber} \wedge \\
 \quad (dID, mID) \in \text{dom timberAtDepot} \wedge \\
 \quad (trID, tripID) \in \text{dom thisLoad} \wedge \\
 \quad tripID \in \text{tripIDs} \wedge \\
 \quad (trID, tripID) \in \text{dom shortHaulTrip} \wedge \\
 \quad (sID, dID) \in \text{ran shortHaulTrip} \wedge \\
 \quad (\text{depotForStandsTimber } sID) = dID \wedge \\
 \quad (\text{millForStandsTimber } sID) = mID \wedge \\
 \quad \text{shortHaulTrip } (trID, tripID) = (sID, dID) \wedge \\
 \quad \text{thisLoad } (trID, tripID) > 0 \wedge \\
 \quad \text{timberAtDepot}' = \text{timberAtDepot} \oplus \{(dID, mID) \mapsto \\
 \quad \quad (\text{timberAtDepot } (dID, mID) + \text{thisLoad } (trID, tripID))\} \wedge \\
 \quad \text{thisLoad}' = \text{thisLoad} \oplus \{(trID, tripID) \mapsto 0\} \wedge \\
 \quad \text{maxLoad}' = \text{maxLoad}
 \end{array}$$

In this schema, the timber is unloaded at the depot into a pile which is destined for a particular mill. The schema includes the unchangeable schemas *MillForStandsTimber*, *DepotForStandsTimber*, *TripIDs* and *ShortHaulTrip*, and the changeable schemas

Truck and *Depot*. The function *shortHaulTrip* determines that the trip started at the stand is the same one which ends at the depot. Before unloading at the depot, the truck must be loaded. The load is added to the pile for the destination mill. After unloading, the truck's load becomes zero. The function *maxLoad* is unchanged by this schema.

ShortHaulTransportSchema combines the two action schemas *LoadTruckAtStand* and *TransportAndUnloadTruckAtDepot*.

$$\text{ShortHaulTransportSchema} \hat{=} \text{LoadTruckAtStand} \wedge \text{TransportAndUnloadTruckAtDepot}$$

The truck's long-haul trips are defined in schema *LongHaulTrip*. This schema includes the unchanged schema *TripIDs*, and defines a function, *longHaulTrip*, which has as inputs the truck's ID and the trip's ID, and outputs the ID of the depot where the trip starts and the ID of the mill where the trip ends. For each truck and trip combination, there is only one long-haul trip.

$\begin{array}{l} \text{LongHaulTrip} \\ \exists \text{TripIDs} \\ \text{longHaulTrip} : (\text{TRUCKID} \times \text{TRIPID}) \rightarrow \\ \quad (\text{DEPOTID} \times \text{MILLID}) \end{array}$
$\begin{array}{l} \forall \text{trID} : \text{TRUCKID} \bullet \exists_1 \text{tripID} : \text{TRIPID} \bullet \\ \quad \text{tripID} \in \text{tripIDs} \wedge \\ \quad (\text{trID}, \text{tripID}) \in \text{dom longHaulTrip} \wedge \\ \quad \#\{(\text{longHaulTrip}(\text{trID}, \text{tripID}))\} = 1 \end{array}$

The schema *LoadTruckAtDepot* governs the addition of a load of timber to particular truck at the depot. It includes the changeable schemas *Depot*, *Truck*, *TripIDs* and *LongHaulTrip*. Before the trip, the truck is empty and there is more than a truck load of timber in the pile at the depot, destined for the mill. The ID of the trip to the depot does not exist before the trip is made, and it is added to the set *tripIDs* (as a valid trip) once the trip has begun. The function *longHaulTrip* is also updated with this

trip's details. If the mass of timber at the depot is greater than or equal to the truck's maximum load, the truck is loaded with the maximum load it can carry and the mass of timber at the depot is diminished by the mass loaded. The function *maxLoad* is unchanged by this schema.

$$\begin{array}{l}
\text{LoadTruckAtDepot} \\
\hline
\Delta \text{Depot} \\
\Delta \text{Truck} \\
\Delta \text{TripIDs} \\
\Delta \text{LongHaulTrip} \\
\hline
\forall dID : \text{DEPOTID} \bullet \\
\quad \exists_1 mID : \text{MILLID} \bullet \\
\quad \exists trID : \text{TRUCKID} \bullet \\
\quad \exists tripID : \text{TRIPID} \bullet \\
\quad (dID, mID) \in \text{dom } \text{timberAtDepot} \wedge \\
\quad (trID, tripID) \in \text{dom } \text{thisLoad} \wedge \\
\quad trID \in \text{dom } \text{maxLoad} \wedge \\
\quad (trID, tripID) \in \text{dom } \text{longHaulTrip} \wedge \\
\quad (dID, mID) \in \text{ran } \text{longHaulTrip} \wedge \\
\quad tripID \notin \text{tripIDs} \wedge \\
\quad (\text{thisLoad } (trID, tripID)) = 0 \wedge \\
\quad (\text{timberAtDepot } (dID, mID)) \geq (\text{maxLoad } trID) \wedge \\
\quad \text{tripIDs}' = \text{tripIDs} \cup \{tripID\} \wedge \\
\quad \text{longHaulTrip}' = \text{longHaulTrip} \cup \{(trID, tripID) \mapsto (dID, mID)\} \wedge \\
\quad \text{thisLoad}' = \text{thisLoad} \oplus \{(trID, tripID) \mapsto (\text{maxLoad } trID)\} \wedge \\
\quad \text{timberAtDepot}' = \text{timberAtDepot} \oplus \{(dID, mID) \mapsto \\
\quad \quad ((\text{timberAtDepot } (dID, mID)) - (\text{maxLoad } trID))\} \wedge \\
\quad \text{maxLoad}' = \text{maxLoad}
\end{array}$$

The schema *TransportAndUnloadTruckAtMill* describes the transportation and unloading at the mill of a load of timber from a particular truck. The schema includes the unchangeable schemas *TripIDs* and *longHaulTrip*, and the changeable schemas *Truck* and *Mill* (defined in section 5.1.4). The function *longHaulTrip* determines that the trip started at the depot is the same one which ends at the mill. Before unloading at the mill, the truck must be loaded. The load is added to the mill's logyard. After unloading, the truck's load becomes zero. The function *maxLoad* is unchanged by this schema.

TransportAndUnloadTruckAtMill

\exists *TripIDs*

\exists *LongHaulTrip*

Δ *Truck*

Δ *Mill*

\forall *dID* : *DEPOTID* •

\exists_1 *mID* : *MILLID* •

\exists *trID* : *TRUCKID* •

\exists *tripID* : *TRIPID* •

mID \in *dom logyard* \wedge

$(trID, tripID) \in \text{dom } thisLoad \wedge$

tripID \in *tripIDs* \wedge

$(trID, tripID) \in \text{dom } longHaulTrip \wedge$

$(dID, mID) \in \text{ran } longHaulTrip \wedge$

longHaulTrip (*trID*, *tripID*) = (*dID*, *mID*) \wedge

thisLoad (*trID*, *tripID*) > 0 \wedge

logyard' = *logyard* \oplus {*mID* \mapsto (*logyard* *mID* + *thisLoad* (*trID*, *tripID*))} \wedge

thisLoad' = *thisLoad* \oplus {(*trID*, *tripID*) \mapsto 0} \wedge

maxLoad' = *maxLoad*

LongHaulTransportSchema combines the two action schemas *LoadTruckAtDepot* and *TransportAndUnloadTruckAtMill*.

$$LongHaulTransportSchema \hat{=} LoadTruckAtDepot \wedge TransportAndUnloadTruckAtMill$$

5.1.4 Mill

Pulp mills are very capital intensive to build, so the millers try to keep it running all year round. This means that a raw material source needs to be available – either from the mill company's own forests, or from other timber suppliers.

Table 4.8 (page 97) shows the constraints and actions which apply to the miller (processor). If bark is stripped in the field, debarking of logs would not occur at the mill. If tree-lengths were made into logs earlier (e.g. at the stand's roadside or at the depot), logmaking would not occur at the mill. The main concerns are having enough logs

to keep the mill running continuously, and having the right species mix available to pulp.

The mill is described in the three schemas *DefineMill*, *InitMill* and *FillMillLogyardWithMinLogs*. Schema *DefineMill* defines two functions, *logyardMin* and *logyard*. The first specifies the minimum mass of logs each mill's logyard should contain; the second keeps track of the mass of logs in the logyard. The logyard must never be empty and the mass of logs in it must be greater than the minimum specified by the function *logyardMin*. This is to ensure that there are always enough logs in the logyard to cover the eventualities of the transport system failing, or stands not being accessible in adverse weather.

$\begin{array}{l} \textit{DefineMill} \\ \textit{logyardMin} : \textit{MILLID} \rightarrow \textit{MASS} \\ \textit{logyard} : \textit{MILLID} \rightarrow \textit{MASS} \end{array}$
$\begin{array}{l} \forall mID : \textit{MILLID} \bullet \\ mID \in \text{dom } \textit{logyardMin} \wedge \\ mID \in \text{dom } \textit{logyard} \wedge \\ (\textit{logyardMin } mID) > 0 \wedge \\ (\textit{logyard } mID) \geq (\textit{logyardMin } mID) \end{array}$

The mill's logyard is initialised in schema *InitMill* to the empty set. The function *logyardMin* is unchanged by this schema.

$\begin{array}{l} \textit{InitMill} \\ \Delta \textit{DefineMill} \end{array}$
$\begin{array}{l} \textit{logyard}' = \emptyset \\ \textit{logyardMin}' = \textit{logyardMin} \end{array}$

Schema *FillMillLogyardWithMinLogs* fills up the logyard with the minimum mass of logs (as required by that mill, and specified in the function *logyardMin*). It includes the changeable schema *InitMill*. In this schema, the logs are added to the logyard so that it reaches the minimum mass of logs required for that mill. The three schemas are then combined to form schema *Mill*. The contents of the function *logyard* are changed in

schemas *TransportAndUnloadTruckAtMill* and *AcceptOtherSuppliersTimberAtMill* when logs are delivered to the mill, and in schema *RemoveLogsForProcessing* when logs are removed for processing.

$\begin{array}{l} \text{---} \textit{FillMillLogyardWithMinLogs} \text{---} \\ \Delta \textit{InitMill} \\ \hline \forall mID : \textit{MILLID} \bullet \\ \quad mID \in \text{dom } \textit{logyardMin} \wedge \\ \quad mID \in \text{dom } \textit{logyard} \wedge \\ \quad \textit{logyard}' = \textit{logyard} \cup \{(mID, \textit{logyardMin } mID)\} \wedge \\ \quad \textit{logyardMin}' = \textit{logyardMin} \end{array}$

$$\textit{Mill} \hat{=} \textit{DefineMill} \wedge \textit{InitMill} \wedge \textit{FillMillLogyardWithMinLogs}$$

Timber from other timber suppliers are added to the mill's logyard in schema *AcceptOtherSuppliersTimberAtMill*. This schema includes the unchangeable schema *MillAcceptableSpecies* and the changeable schema *Mill*. It also includes two inputs, *load?*, which represents a load of timber brought to the mill and its species, and *whichMill?*, which identifies the mill to which the timber is being brought. In order for the timber to be accepted at the mill, it has to be of an acceptable species. After the transaction, the logyard contains whatever timber were there as well as the timber in the load.

$\begin{array}{l} \text{---} \textit{AcceptOtherSuppliersTimberAtMill} \text{---} \\ \exists \textit{MillAcceptableSpecies} \\ \Delta \textit{Mill} \\ \textit{load}? : (\textit{MASS} \times \textit{SPECIESID}) \\ \textit{whichMill}? : \textit{MILLID} \\ \hline \textit{whichMill}? \in \text{dom } \textit{logyard} \\ \textit{whichMill}? \in \text{dom } \textit{acceptableSpecies} \\ \textit{first}(\textit{load}?) \in \text{ran } \textit{logyard} \\ \textit{second}(\textit{load}?) \in \text{ran } \textit{acceptableSpecies} \\ \textit{logyard}' = \textit{logyard} \oplus \{\textit{whichMill}? \mapsto ((\textit{logyard } \textit{whichMill}?) + \textit{first}(\textit{load}?)\})\} \end{array}$

Schema *RemoveLogsForProcessing* also includes the changeable schema *Mill*, and has as inputs *load?* (the load of logs removed from the logyard) and *whichMill?* (to

identify the mill at which the logs are being removed). Before a load can be removed, the logyard must have at least a load of logs to remove. After the removal of the load, the logyard's mass is reduced by the mass of the load.

$\text{RemoveLogsForProcessing}$ <hr/> ΔMill $\text{load?} : \text{MASS}$ $\text{whichMill?} : \text{MILLID}$ <hr/> $\text{load?} \in \text{ran logyard}$ $\text{whichMill?} \in \text{dom logyard}$ $(\text{logyard } \text{whichMill?}) \geq \text{load?}$ $\text{logyard}' = \text{logyard} \oplus \{\text{whichMill?} \mapsto ((\text{logyard } \text{whichMill?}) - \text{load?})\}$
--

The schema *ExceptionProcessLogs* describes what will occur if, on removing a load of logs for processing, the mass in the logyard falls under the minimum allowable mass for that mill. The schema includes the unchangeable schema *Mill*, the input *whichMill?* and the output *message!*. When trying to process a load of logs (*load?*), if the mill *whichMill?* does not have enough logs to keep the minimum amount of logs in the logyard, an exception output message is given to say that there are not enough logs in the logyard.

The schema *RemoveLogsForProcessing* is made up of either *ProcessLogs* or *ExceptionProcessLogs*.

$\text{ExceptionProcessLogs}$ <hr/> $\exists \text{Mill}$ $\text{load?} : \text{MASS}$ $\text{whichMill?} : \text{MILLID}$ $\text{message!} : \text{MESSAGE}$ <hr/> $\text{load?} \in \text{ran logyard}$ $\text{whichMill?} \in \text{dom logyard}$ $\text{whichMill?} \in \text{dom logyardMin}$ $\text{logyard } \text{whichMill?} - \text{logyardMin } \text{whichMill?} < \text{load?} \Rightarrow$ $\text{message!} = \text{NotEnoughLogs}$

$$\text{RemoveLogsForProcessing} \hat{=} \text{ProcessLogs} \vee \text{ExceptionProcessLogs}$$

5.2 Second abstraction: forest, transport, mill

In preparation for the development of specifications for the forest harvest scheduling system, the specification of the forest-to-mill domain given in the previous section was inspected. Some aspects were not described in enough detail to support the functionality required by the system. The forest-to-mill domain specification was then refined to include the required detail. In particular, more detail about forestry aspect was required to describe the trees' genetic material, the regimes which govern the trees' growth and the forestry company hierarchy. The prediction of a stand of trees' log volume (or mass) at harvesting needed expansion, as this is the output of the stand which will be used to determine to which mill the logs should be sent. The logistics chain, which links the stand to the mill needed to be defined, to ensure that there was a link between every stand and the mill to which its timber should be sent. If this link does not exist, a stand may be harvested, but the timber would never get to the appropriate mill. Related to the logistics chain are roads along which the timber is to be transported, and the distance a truck will travel to get to the mill. This is important to be able to work out the most appropriate and cost-effective mill to which the stands' timber is to be sent.

The differences between (or additions to) the Z specifications are highlighted below, with the exception of the first three sections. These three sections, which cover the trees' 'species' or genetic material (section 5.2.1), regimes (section 5.2.2) and forest management hierarchy (section 5.2.3), are too lengthy to describe here. The detailed specifications can be found in sections D.2.1, D.2.2 and D.4.1 respectively.

The formal specification of the forest, transport and mill domains can be found in Appendix D; an index of schema names can be found on page 431.

5.2.1 ‘Species’ or genetic material

In the first abstraction, the genetic material of the trees grown in the stand is defined in terms of a *SPECIESID*. However, there are several business rules in forestry and the forest harvest scheduling system which require more detail. The genetic material hierarchy was defined in terms of genus and species (pure species and hybrids). The genetic material hierarchy is shown in Figures 4.7 and 4.8 (on pages 73 and 74 respectively). In the specification given in Appendix D, section D.2.1, the hierarchy as shown in Figures 5.4 and 5.5 were modelled.

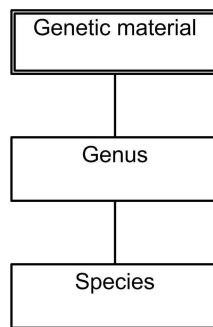


Figure 5.4: Hierarchy of genetic material (loosely called “species”) modelled in the Z specification

5.2.2 Regimes

In the first abstraction, regimes were not mentioned at all. Regimes were described in section 4.3.4. The planned regime describes the activities which are to be applied to all the trees in the stand, and when they are to take place. The actual regime records what has actually happened in the stand, and when the activities took place. In the forest harvest scheduling system, the planned regime is used to determine when a stand of trees should be harvested. The regime details are also used to predict the volume of harvested trees in the stand; this is important for being able to determine

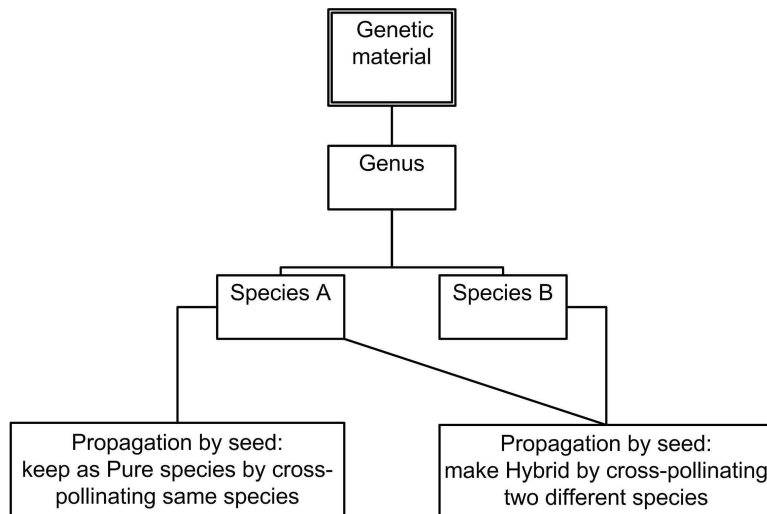


Figure 5.5: Diagram showing the relationship between pure species and hybrids modelled in the Z specification

how much of the mill's log demand will be met by harvesting this stand.

Because this specification describes forestry intended for pulp, the regime is relatively simple (i.e. it has few actions). The Z specification of the plantation forests' regimes can be found in section D.2.2.

5.2.3 Forestry company hierarchy

In the first abstraction, the only part of the forestry company hierarchy which was defined was the stand. In the second abstraction, the company, estates and stands were linked. There is at least one management level missing between the estate and the company¹ (see Figure 4.5 on page 72). Figure 5.6 shows the forestry company hierarchy defined in the Z specification (see section D.4.1).

¹The number of levels between the company and the estate depends on the forestry company.

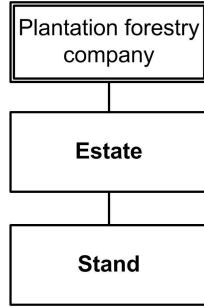


Figure 5.6: Hierarchy of forestry company management levels modelled in the Z specification

5.2.4 Volume or mass of logs produced when harvesting the stand

In the first abstraction, schema *MassOfFelledTrees* contained a function, *massOfFelledTrees*, which recorded the mass of logs at harvesting age. As discussed in section 4.6.3, the volume of the stand's logs is estimated with a growth and yield simulator. This volume is then converted into mass.

DefineEstimateStandsVolume

$$\text{standVolume} : (STANDID \times REGIMEID \times GENMATERIALID \times HEIGHT \times AGE \times SPH \times AREA) \rightarrow VOLUME$$

$$\forall sID : STANDID \bullet$$

$$\quad \forall regID : REGIMEID \bullet$$

$$\quad \forall genMatID : GENMATERIALID \bullet$$

$$\quad \forall SI : HEIGHT \bullet$$

$$\quad \forall age : AGE \bullet$$

$$\quad \forall stemsPerHa : SPH \bullet$$

$$\quad \forall standArea : AREA \bullet$$

$$SI > 0 \wedge$$

$$age \geq 0 \wedge$$

$$stemsPerHa > 0 \wedge$$

$$standArea > 0 \wedge$$

$$(sID, regID, genMatID, SI, age, stemsPerHa, standArea) \in \text{dom standVolume} \wedge$$

$$\text{standVolume}(sID, regID, genMatID, SI, age, stemsPerHa, standArea) \geq 0$$

Schema *DefineEstimateStandsVolume* shows the dependencies of calculating the stand's log volume: inputs are the stand's ID, the regime ID, the identifier of the genetic material grown, the site index of the stand (a measure of growth rate), the stand's age, number of stems (trees) planted in it, and the area of the stand.

DefineConvertStandVolumeToMass

\exists *EstimateStandsVolume*

$standMass : (STANDID \times GENMATERIALID \times VOLUME) \rightarrow MASS$

$\forall sID : STANDID \bullet$

$\forall regID : REGIMEID \bullet$

$\forall genMatID : GENMATERIALID \bullet$

$\forall SI : HEIGHT \bullet$

$\forall age : AGE \bullet$

$\forall stemsPerHa : SPH \bullet$

$\forall standArea : AREA \bullet$

$(sID, regID, genMatID, SI, age, stemsPerHa, standArea) \in \text{dom } standVolume \wedge$

$(sID, genMatID, standVolume(sID, regID, genMatID, SI, age, stemsPerHa, standArea)) \in \text{dom } standMass \wedge$

$standMass(sID, genMatID, standVolume(sID, regID, genMatID, SI, age, stemsPerHa, standArea)) \geq 0 \wedge$

$standVolume(sID, regID, genMatID, SI, age, stemsPerHa, standArea) > 0 \Rightarrow$

$standMass(sID, genMatID, standVolume(sID, regID, genMatID, SI, age, stemsPerHa, standArea)) > 0$

The stand's log volume is then converted to mass. The function which does this (*standMass*) is defined in schema *DefineConvertStandVolumeToMass*. In this model, the stand's log mass can be determined by giving the genetic material of the logs and the volume. As discussed in section 4.4.2, the mass also depends on the wood's density and the number of days since the tree was felled (not included in this version of the function *standMass*).

These two schemas are used in schema *StandsVolumeAndMassAtHarvesting*. These schemas can be found in section D.4.5.

5.2.5 Logistics chain

In the first abstraction, a schema (*MillForStandsTimber*) describes how there is a single mill to which the stand's timber will be sent. The specification further shows how timber is sent to a depot and then to that mill's logyard. However, this does not reflect reality adequately. It could be possible for a stand of trees to be sent to several different mills. The allocation of a particular mill (destination) to the stand will occur close to, or at, harvesting.

When planting a stand of trees, one needs to make sure that the genetic material of the trees planted is suited to the stand, and there is a mill which requires trees having that genetic material. In addition, there needs to be a road which links the stand to a depot, and another transport link which would enable the timber to be transported to the mill.

Schema *LogisticsChain* defines this logistics link:

<p><i>LogisticsChain</i></p> <p>$sendLogsToDepot : STANDID \leftrightarrow DEPOTID$ $sendLogsToMill : DEPOTID \leftrightarrow MILLID$</p> <p>$\forall sID : STANDID \bullet$ $\quad \exists dID : DEPOTID \bullet$ $\quad \exists mID : MILLID \bullet$ $sID \in \text{dom } sendLogsToDepot \wedge$ $dID \in \text{ran } sendLogsToDepot \wedge$ $dID \in \text{dom } sendLogsToMill \wedge$ $mID \in \text{ran } sendLogsToMill \wedge$ $(sendLogsToDepot \ sID) = dID \wedge$ $(sendLogsToMill \ dID) = mID \wedge$ $(sendLogsToMill (sendLogsToDepot \ sID)) = mID \wedge$ $\#\{(sendLogsToDepot \ sID)\} \geq 1 \wedge$ $\#\{(sendLogsToMill \ dID)\} \geq 1 \wedge$ $\#\{(sendLogsToMill (sendLogsToDepot \ sID))\} \geq 1$</p>

This schema shows that there is a route from the stand to the depot ($(sendLogsToDepot \ sID) = dID$), and there is also a route from the depot to

the mill $((sendLogsToMill\ dID) = mID)$. There could be more than one depots to which a stands' timber could be sent, and there could be more than one mill to which the timber could be sent. The schema which records which mill the stand's timber has been allocated to (schema *MillForStandsLogs*) records one of these mills.

These schemas can be found in section D.3.2.

5.2.6 Transporting the timber: roads

In the first abstraction, roads were not included, and the distance of a trip along a particular road between the stand and depot, or depot and mill was not calculated (or recorded). In addition, although each trip's load was calculated, it was not recorded. Both distance and mass transported are required for calculating the transport costs in the forest harvest scheduling system.

ShortHaulRoad

$\exists LogisticsChain$

$roadStandToDepot : ROADID \rightarrow (STANDID \times DEPOTID)$

$shortHaulDistance : (STANDID \times DEPOTID \times ROADID) \rightarrow DISTANCE$

$\forall sID : STANDID \bullet$

$\forall dID : DEPOTID \bullet$

$\exists rID : ROADID \bullet$

$sID \in \text{dom } sendLogsToDepot \wedge$

$dID \in \text{ran } sendLogsToDepot \wedge$

$dID \in \text{dom } sendLogsToMill \wedge$

$rID \in \text{dom } roadStandToDepot \wedge$

$sID \in \{first(roadStandToDepot\ rID)\} \wedge$

$dID \in \{second(roadStandToDepot\ rID)\} \wedge$

$(sendLogsToDepot\ sID) = dID \wedge$

$(roadStandToDepot\ rID) = (sID, (sendLogsToDepot\ sID)) \wedge$

$\#\{roadStandToDepot\} \geq 1 \wedge$

$shortHaulDistance\ (sID, dID, rID) \geq 0$

Schema *ShortHaulRoad* shows how the road and distances are included for short-haul transport. This schema defines a function, *roadStandToDepot*, in which road(s)

are recorded which link stands to depots. The stands and depots must be in the logistics chain, and there may be more than one road linking the two. In addition, the distance from a stand to a depot along a particular road is defined with function *shortHaulDistance*. A similar schema has been defined for long-haul transport (see section D.5.2).

The schema *ShortHaulTrip* in the first abstraction did not record the mass of timber transported by the trucks (although at the time that the load was being transported, the mass was known, as it was stored in the function *currentLoad*). As the cost of transport needs to be included in the forest harvest scheduling system, it is necessary to record the mass, which is done in function *shortHaulTripMass*. This schema can be seen in section D.5.4.1. A similar schema exists for long-haul trips (see section D.5.5.1).

$$\begin{array}{l}
 \text{ShortHaulTrip} \\
 \exists \text{TripIDs} \\
 \text{shortHaulTrip} : (\text{TRUCKID} \times \text{TRIPID}) \rightarrow (\text{STANDID} \times \text{DEPOTID} \times \text{ROADID}) \\
 \text{shortHaulTripMass} : (\text{TRUCKID} \times \text{TRIPID}) \rightarrow \text{MASS} \\
 \forall \text{trID} : \text{TRUCKID} \bullet \\
 \quad \exists_1 \text{tripID} : \text{TRIPID} \bullet \\
 \quad \text{tripID} \in \text{tripIDs} \wedge \\
 \quad (\text{trID}, \text{tripID}) \in \text{dom shortHaulTrip} \wedge \\
 \quad (\text{trID}, \text{tripID}) \in \text{dom shortHaulTripMass} \wedge \\
 \quad \#\{(\text{shortHaulTrip}(\text{trID}, \text{tripID}))\} = 1 \wedge \\
 \quad \#\{(\text{shortHaulTripMass}(\text{trID}, \text{tripID}))\} = 1 \wedge \\
 \quad \text{shortHaulTripMass}(\text{trID}, \text{tripID}) > 0
 \end{array}$$

5.3 Forest harvest scheduling system

This section gives highlights of the formal forest harvest scheduling system which is described in Appendix E. This specification describes a forest harvest scheduling system in which stands are harvested at rotation age, and the wood is allocated to a mill, depending on whether that mill needs the stand's wood, and whether the stand's wood has a wood property value which is acceptable to the mill. The objective function is to maximise the profit of the forestry part of the integrated forestry and pulp company over the strategic planning horizon, i.e. cost of delivered logs less the cost of log production and the cost of transporting logs to the mill. The optimal solution is the one which satisfies all the constraints and has the greatest profit.

As previously mentioned, the forest harvest scheduling system is embedded in the forestry, transport and mill domains. The forest harvest scheduling system specification makes use of many of the domain descriptions given in Appendix D. These descriptions are not repeated in Appendix E, but are referred to in the text.

There are several steps involved in getting to the optimal solution (or even determining if an optimal solution exists). The following shows the list of items which need to be defined or calculated, in logical order. The number given next to the item shows the section in which the item is discussed.

Inputs

- 5.3.1 The planning horizon length for the strategic plan needs to be defined.
- 5.3.2 The function for predicting the stand's average wood property needs to be defined.
- 5.3.3 Each mill's constraints (mass of wood required and wood properties of that wood) need to be defined.

Inputs, continued

- 5.3.4 The cost inputs for the forest, transport and mill domains need to be defined.

Calculations

- 5.3.5 The different options for each stand need to be enumerated for the duration of the strategic planning horizon to be determined.
- 5.3.6 Stand options need to be combined to form possible solutions. A stand may only be represented once in each possible solution.
- 5.3.7 Each possible solution needs to be checked to see if it meets the mill's requirements. If it does, it is tagged as a feasible solution.
- 5.3.8 For all the possible solutions which are also feasible, the objective function needs to be determined.
- 5.3.9 The optimal solution needs to be determined. This is the feasible solution which gives the highest value for the objective function.

FHSS system and outputs

- 5.3.10 The schemas of the forest harvest scheduling system need to be put together to form a single schema.

5.3.1 Planning horizon

Forestry is a long-term activity. Compared to other crops, a stand of trees takes a long time to mature, so recording information about the stands and planning what should happen to them is important. In this section, the type *YEARNO* is defined to cater for all the years in the planning horizon. In addition, the three forestry planning horizons (the long-term/strategic, medium-term/tactical and the short-term/operational planning horizons) and the date on which the plan is to start are defined.

The specification starts by defining the type *YEARNO* (a finite set of natural numbers) so that the planning horizons can be defined, as well as the activities and

constraints in each year of the planning horizon.

| *YEARNO* : FN

The planning horizon is defined in schema *PlanningHorizonDefinition* in terms of its three components: the operational, tactical and strategic planning horizons, which are given as three inputs to the system. The operational planning horizon (which is between one and two years in length) is always shorter than or equal to the tactical planning horizon (defined here to be between three and five years long). The tactical planning horizon, in turn, is shorter than that strategic planning horizon, which is between 20 and 30 years in length. Schema *PlanningHorizonDefinition* also takes in a date (*dateToStartPlan?*) which is the date on which the plan is to be started.

PlanningHorizonDefinition

strategicHorizon?, *tacticalHorizon?*, *operationalHorizon?* : *YEARNO*

dateToStartPlan? : *DATE*

$operationalHorizon? \leq tacticalHorizon? \leq strategicHorizon?$

$1 \leq operationalHorizon? \leq 2$

$3 \leq tacticalHorizon? \leq 5$

$20 \leq strategicHorizon? \leq 30$

This section mirrors that found in section E.1.

5.3.2 Wood property prediction

This section describes wood properties: how to identify them, and how to evaluate them at a stand level. In Appendix E, wood property prediction is defined in section E.2.

WOODPROPID (the identifier of the wood property), *WOODPROPNAME* (the name of the wood property, e.g. density, fibre length) and *WOODPROPVAL* (the value of the wood property measurement) are defined so that they can be used later

on in the specification.

[*WOODPROPID*, *WOODPROPNAME*]

| *WOODPROPVAL* : \mathbb{FN}

Schema *EstimateStandWoodProp* calculates the average wood property value for a stand of trees and stores the result in the function *standWoodProp*. As described in section 4.7.5, the prediction of wood properties depends on the wood property being predicted, the genetic material ('species') of the trees in the stand and the site index, age and altitude of the stand. In this schema, it is assumed that the site index is always greater than zero, the age of the stand of trees and the stand altitude are greater or equal to zero. The resultant predicted wood property value is greater than or equal to zero.

EstimateStandWoodProp
 $standWoodProp : (STANDID \times WOODPROPID \times GENMATERIALID \times HEIGHT \times AGE \times HEIGHT) \rightarrow WOODPROPVAL$

$\forall sID : STANDID \bullet$
 $\forall woodPropID : WOODPROPID \bullet$
 $\forall genMatID : GENMATERIALID \bullet$
 $\forall SI : HEIGHT \bullet$
 $\forall age : AGE \bullet$
 $\forall altitude : HEIGHT \bullet$
 $SI > 0 \wedge$
 $age \geq 0 \wedge$
 $altitude \geq 0 \wedge$
 $(sID, woodPropID, genMatID, SI, age, altitude) \in \text{dom } standWoodProp \wedge$
 $standWoodProp (sID, woodPropID, genMatID, SI, age, altitude) \geq 0$

5.3.3 Mill's requirements (constraints)

Each mill requires a certain mass of wood to process annually. The aim is for enough wood to be provided so that the mill does not need to stop working. This section describes the minimum and maximum mass requirements for the mill for each year in

the planning horizon, as well as the minimum and maximum acceptable wood property values for each year. In Appendix E, the mill requirements (or constraints) are defined in section E.3.

Schema *MillRequirements* defines the mill's timber requirements. It includes the unchangeable schema *PlanningHorizonDefinition*, and has five functions: *minMass*, *maxMass*, *woodPropForMill*, *minWoodPropValue* and *maxWoodPropValue*. The first two (*minMass* and *maxMass*) both take the millID and year number as inputs, and have mass of timber as an output. They record the minimum and maximum required mass of wood at that mill per annum. *woodPropForMill* records the wood property whose value variability the mill is trying to reduce. The next two functions, *minWoodPropValue* and *maxWoodPropValue*, store the minimum and maximum value for this wood property, per annum.

MillRequirements

\exists *PlanningHorizonDefinition*

minMass : (*MILLID* × *YEARNO*) → *MASS*

maxMass : (*MILLID* × *YEARNO*) → *MASS*

woodPropForMill : *MILLID* → *WOODPROPID*

minWoodPropValue : (*MILLID* × *WOODPROPID* × *YEARNO*) → *WOODPROPVAL*

maxWoodPropValue : (*MILLID* × *WOODPROPID* × *YEARNO*) → *WOODPROPVAL*

\forall *millID* : *MILLID* •

\forall *yearNo* : *YEARNO* •

$1 \leq \text{yearNo} \leq \text{strategicHorizon?} \wedge$

$(\text{millID}, \text{yearNo}) \in \text{dom } \text{minMass} \wedge$

$(\text{millID}, \text{yearNo}) \in \text{dom } \text{maxMass} \wedge$

$(\text{millID}) \in \text{dom } \text{woodPropForMill} \wedge$

$(\text{millID}, \text{woodPropForMill}(\text{millID}), \text{yearNo}) \in \text{dom } \text{minWoodPropValue} \wedge$

$(\text{millID}, \text{woodPropForMill}(\text{millID}), \text{yearNo}) \in \text{dom } \text{maxWoodPropValue} \wedge$

$\text{minMass}(\text{millID}, \text{yearNo}) \leq \text{maxMass}(\text{millID}, \text{yearNo}) \wedge$

$\#\{\text{woodPropForMill}(\text{millID})\} = 1 \wedge$

$\text{minWoodPropValue}(\text{millID}, \text{woodPropForMill}(\text{millID}), \text{yearNo}) \leq$

$\text{maxWoodPropValue}(\text{millID}, \text{woodPropForMill}(\text{millID}), \text{yearNo})$

For each mill, and for each year in the strategic planning horizon, the minimum mass required by the mill must be less than or equal to the maximum mass, and the

minimum wood property value must be less than or equal to the maximum wood property value. In addition, there is only one wood property for which the mill is managing its incoming resource.

5.3.4 Cost inputs

The cost inputs to the forest harvest scheduling system are divided into three sections: forestry costs, transport costs, and the cost of buying delivered logs. Each is described briefly below. The full specification can be found in section E.4.

The type *COST* is defined to be able to specify the costs involved in the system over the strategic planning horizon. These costs are specified in present day terms. This approach is appropriate because the aim of the system is to determine the overall profit of the supply chain and not to model expected interest rate fluctuations over the strategic planning horizon. *COST* is defined as a finite set of integers.

| *COST* : \mathbb{FZ}

Forestry costs are defined in three categories: company-level costs (e.g. the cost of having a head office, paying planning foresters and administrative staff etc.), estate-level costs (e.g. the cost of upgrading roads prior to harvesting, and implementing fire- and pest-prevention) and regime-incurred costs (the cost of planting, tending and growing the trees). The first two are defined in schemas *ForestryCompanyLevelCosts* and *EstateLevelCosts* in sections E.4.2.1 and E.4.2.2.

In order to implement the regime costs, an addition needs to be made to the regime: first of all, the different 'landmark' events in a regime's growth ('plant', 'canopy closure has been reached', and 'fell') need to be defined. This is captured in type

REGIMEACTION.

REGIMEACTION ::= *plant* | *cClosure* | *fell*

Next, a schema (*RegimeCanopyClosure*) needs to be added to the specification which denotes the age at which canopy closure will occur for a particular regime. This is important to know, as weeding activities can be reduced or cease after this time in the regime, so the stand's maintenance cost will be less.

<p style="text-align: center;"><i>RegimeCanopyClosure</i></p> <hr/> <p>\exists <i>PlannedRegimes</i> <i>regimeCanopyClosure</i> : <i>REGIMEID</i> \times <i>REGIMETYPE</i> \times <i>GENMATERIALID</i> \rightarrow <i>AGE</i></p> <hr/> <p>\forall <i>regID</i> : <i>REGIMEID</i> • \forall <i>genMatID</i> : <i>GENMATERIALID</i> • (<i>regID</i>, <i>planned</i>, <i>genMatID</i>) \in <i>dom regimeCanopyClosure</i> \wedge (<i>regID</i>, <i>planned</i>) \in <i>dom plantAge</i> \wedge (<i>regID</i>, <i>planned</i>) \in <i>dom fellAge</i> \wedge (<i>regimeCanopyClosure</i> (<i>regID</i>, <i>planned</i>, <i>genMatID</i>)) $>$ <i>plantAge</i> (<i>regID</i>, <i>planned</i>) \wedge (<i>regimeCanopyClosure</i> (<i>regID</i>, <i>planned</i>, <i>genMatID</i>)) $<$ <i>fellAge</i> (<i>regID</i>, <i>planned</i>) \wedge {<i>regimeCanopyClosure</i> (<i>regID</i>, <i>actual</i>, <i>genMatID</i>)} = \emptyset</p>

Lastly, the schema *RegimeCosts* defines the costs incurred by implementing a regime in a stand. It contains four functions, *regimePlantingCosts*, *annualRegimeMaintCostBeforeCClosure*, *annualRegimeMaintCostAfterCClosure* and *regimeHarvestingCosts*, all of which take the regimeID, regime type and genetic materialID as inputs and give the cost as an output. The regime and the regime type could be planned or actual. The first function (*regimePlantingCosts*) stores the cost of planting or establishing the stand. This is a once-off cost (i.e. it is incurred at planting). The second and third functions (*annualRegimeMaintCostBeforeCClosure* and *annualRegimeMaintCostAfterCClosure*) record the annual costs incurred when managing the stand: before canopy closure and after canopy closure. The latter costs are less than the former. The last function (*regimeHarvestingCosts*) stores the costs incurred at harvesting. All the costs are greater than, or equal to, zero.

Regime Costs

$regimePlantingCosts : REGIMEID \times REGIMETYPE \times GENMATERIALID \rightarrow COST$
 $annualRegimeMaintCostBeforeCClosure :$
 $REGIMEID \times REGIMETYPE \times GENMATERIALID \rightarrow COST$
 $annualRegimeMaintCostAfterCClosure :$
 $REGIMEID \times REGIMETYPE \times GENMATERIALID \rightarrow COST$
 $regimeHarvestingCosts : REGIMEID \times REGIMETYPE \times GENMATERIALID \rightarrow COST$

$\forall pRegimeID : REGIMEID \bullet \forall aRegimeID : REGIMEID \bullet$
 $\forall genMatID : GENMATERIALID \bullet$
 $(pRegimeID, planned, genMatID) \in \text{dom } regimePlantingCosts \wedge$
 $(aRegimeID, actual, genMatID) \in \text{dom } regimePlantingCosts \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom } annualRegimeMaintCostBeforeCClosure \wedge$
 $(aRegimeID, actual, genMatID) \in \text{dom } annualRegimeMaintCostBeforeCClosure \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom } annualRegimeMaintCostAfterCClosure \wedge$
 $(aRegimeID, actual, genMatID) \in \text{dom } annualRegimeMaintCostAfterCClosure \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom } regimeHarvestingCosts \wedge$
 $(aRegimeID, actual, genMatID) \in \text{dom } regimeHarvestingCosts \wedge$
 $(regimePlantingCosts (pRegimeID, planned, genMatID)) \geq 0 \wedge$
 $(regimePlantingCosts (aRegimeID, actual, genMatID)) \geq 0 \wedge$
 $(annualRegimeMaintCostBeforeCClosure (pRegimeID, planned, genMatID)) \geq 0 \wedge$
 $(annualRegimeMaintCostBeforeCClosure (aRegimeID, actual, genMatID)) \geq 0 \wedge$
 $(annualRegimeMaintCostAfterCClosure (pRegimeID, planned, genMatID)) \geq 0 \wedge$
 $(annualRegimeMaintCostAfterCClosure (aRegimeID, actual, genMatID)) \geq 0 \wedge$
 $(regimeHarvestingCosts (pRegimeID, planned, genMatID)) \geq 0 \wedge$
 $(regimeHarvestingCosts (aRegimeID, actual, genMatID)) \geq 0 \wedge$
 $(annualRegimeMaintCostBeforeCClosure (pRegimeID, planned, genMatID)) \geq$
 $(annualRegimeMaintCostAfterCClosure (pRegimeID, planned, genMatID)) \wedge$
 $(annualRegimeMaintCostBeforeCClosure (aRegimeID, actual, genMatID)) \geq$
 $(annualRegimeMaintCostAfterCClosure (aRegimeID, actual, genMatID))$

The transport costs and the delivered log costs are covered in sections E.4.3 and E.4.4 respectively.

5.3.5 Enumerating all the stand options

In this first of the calculations steps, the list of all the regime action options which could occur in a stand need to be enumerated. If the stand is not planted, it will be planted in the first year of the planning horizon. The planned regime will be followed repeatedly until the end of the strategic planning horizon has been reached. It is

assumed that planting will occur in the same year as felling. If the stand is planted when the plan starts, the options need to be filled in from the actual regime until the stand is felled, from which time the planned regime will be used until the end of the strategic planning horizon has been reached.

As the whole supply chain is being considered, if a stand's timber could be sent to more than one mill, all the combinations of stand options need to be enumerated. For example, if there are two felling events in the strategic planning horizon, and two mills to which the stand's timber could be sent, there would be four stand options generated altogether to capture all the options for that stand over the planning horizon:

Stand option	Felling event 1	Felling event 2
1	Mill A	Mill A
2	Mill A	Mill B
3	Mill B	Mill A
4	Mill B	Mill B

The total number of stand options per stand is determined by

total number of mills to which the stand's timber can be sent^{number of felling events}.

The full specification is given in section E.5. Figure 5.7 shows an example of stand options where the stand is not planted at the time of planning. Figure 5.8 (on page 191) shows an example of stand options where a stand is planted, but canopy closure has not been reached at the time of planning.

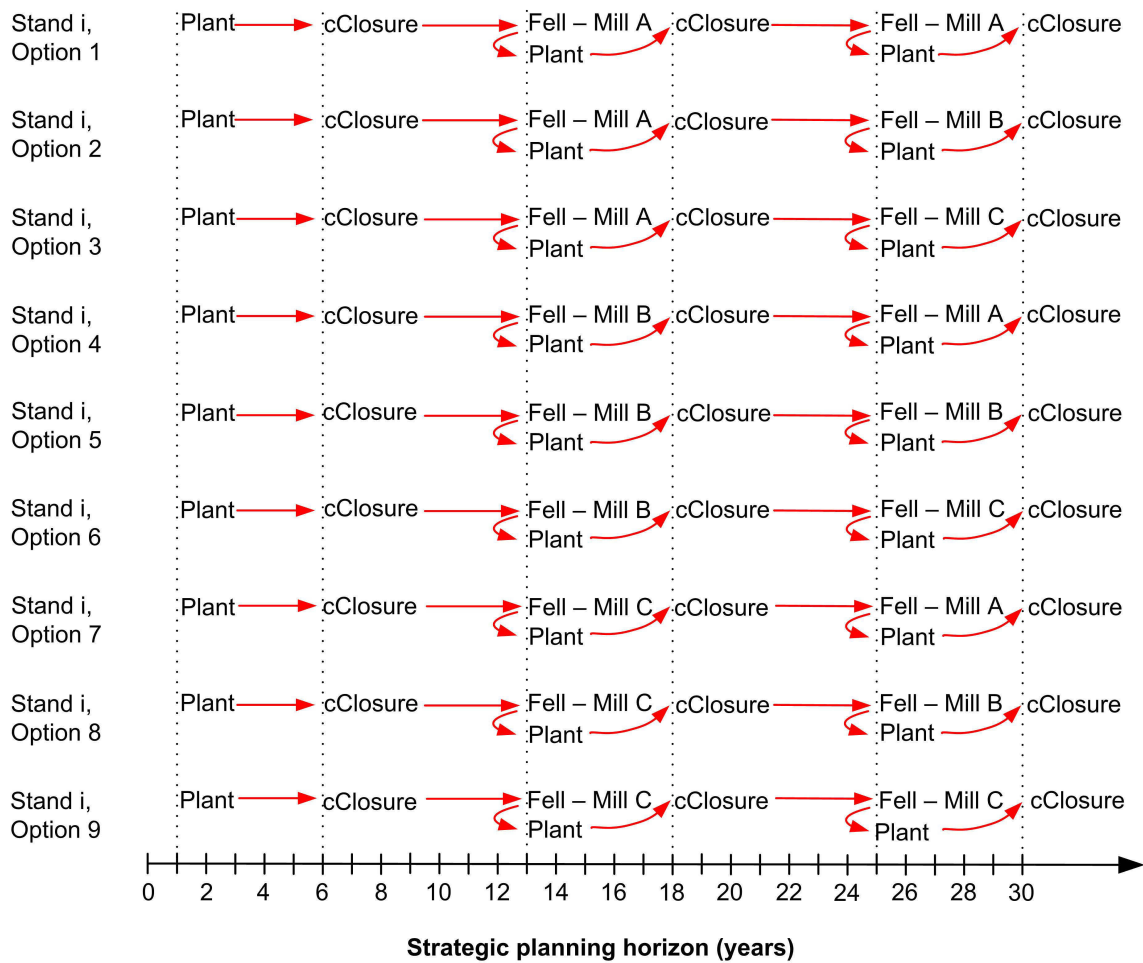


Figure 5.7: Stand options for an unplanted stand, where there are three possible destination mills

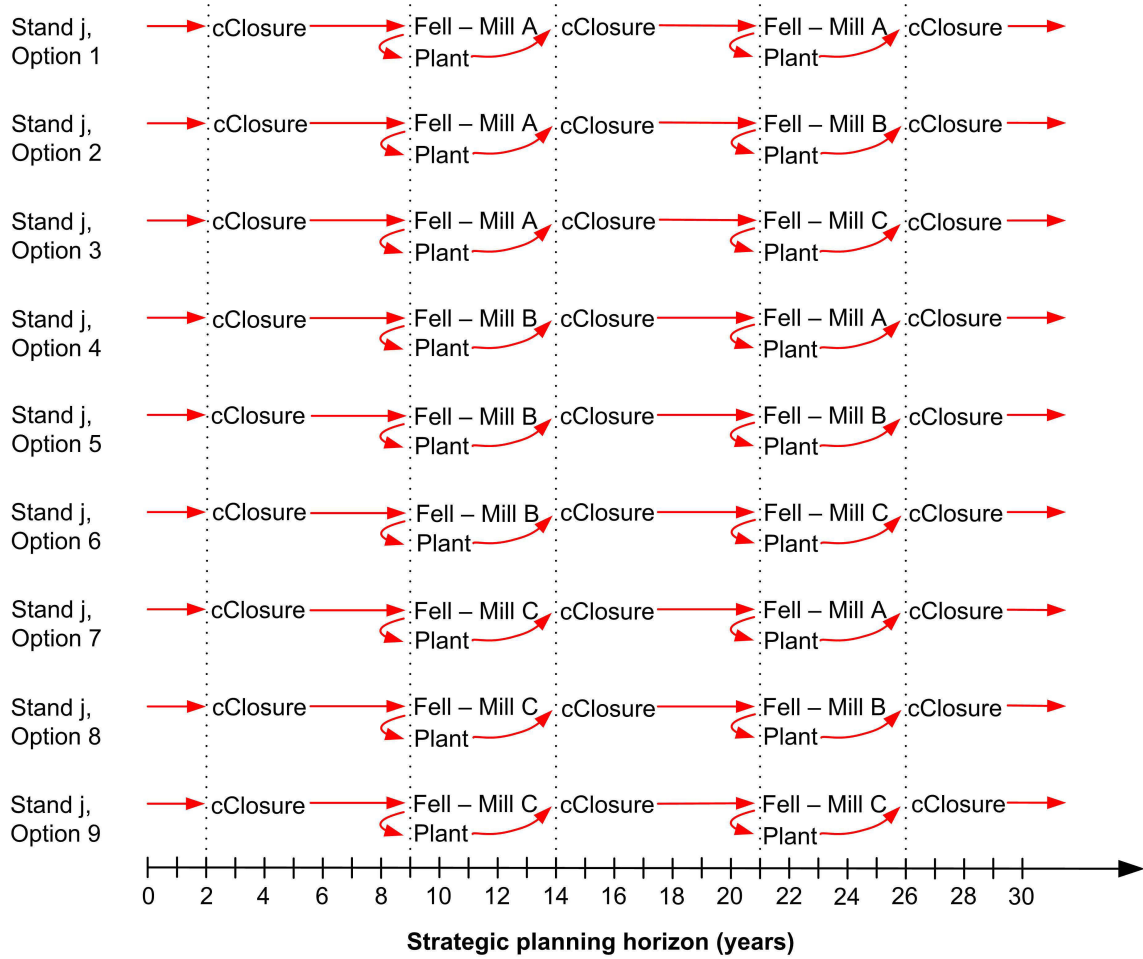


Figure 5.8: Stand options for a planted stand, where there are three possible destination mills

5.3.6 Enumerating all the possible solutions

In the previous step, all of the options for each stand over the strategic planning horizon were enumerated. In this step, all of the combinations of stand options need to be put together in possible solutions from which a single solution (the optimal solution) can be chosen. Each possible solution is defined so that a stand is represented in each possible solution only once. However, each possible combination of stand options must be enumerated to ensure that the solver has every possible option from which to choose a solution. The number of possible solutions will be the product of the total number of each stand's stand options.

Figure 5.9 gives an example of a single possible solution. The full specification of possible solution generation is given in section E.6.1.

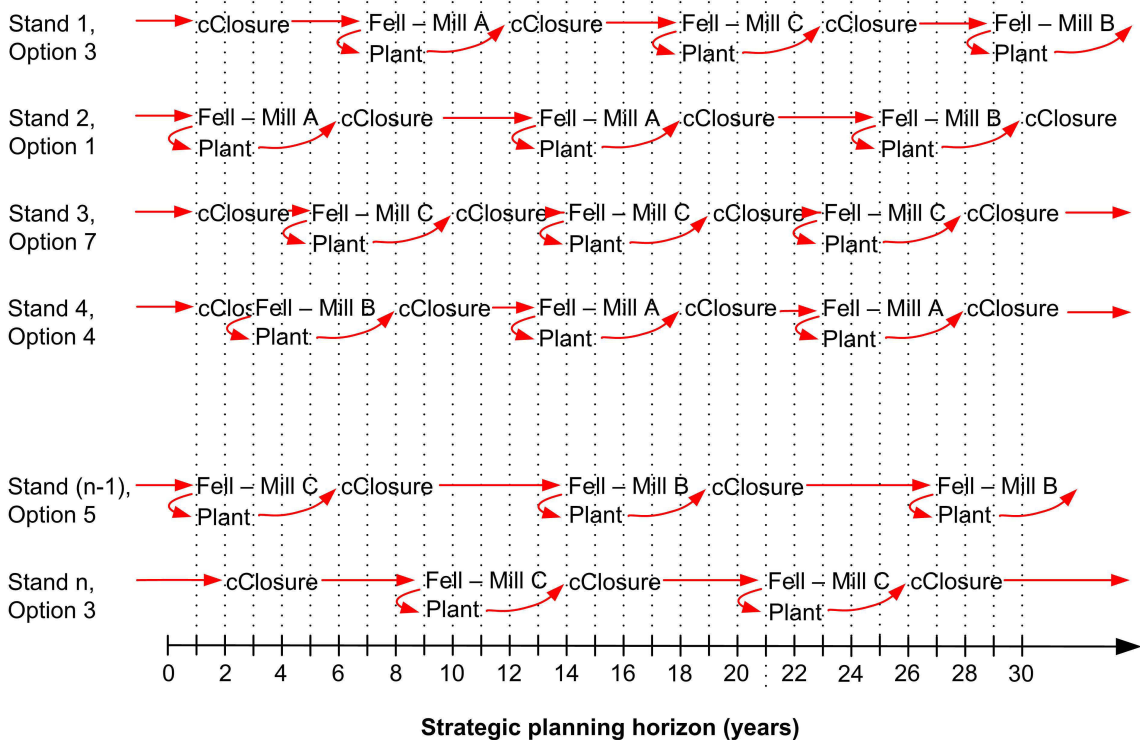


Figure 5.9: A possible solution for the forest harvest scheduling system. Each stand is represented once in each possible solution.

5.3.7 Determining if a possible solution is feasible

There is now a large number of possible solutions, which are made up of every combination of stand options. The task is now to determine if each of these possible solutions is feasible – i.e., if the possible solution meets the constraints (which are the mill requirements).

The first step in determining each possible solution's feasibility is determining the mass of logs which would be delivered to each mill by each possible solution. The schema *AnnualLogMassForMill* is defined and initialised in section E.6.2: it contains a function, *annualLogMassForMill*, defined as

$$\textit{annualLogMassForMill} : (\textit{POSSIBLESOLNID} \times \textit{MILLID} \times \textit{YEARNO}) \rightarrow \textit{MASS}.$$

Schema *CalculateAnnualLogMassForMill* given below shows how this function is updated with values.

Schema *CalculateAnnualLogMassForMill* updates the contents of the function *annualLogMassForMill*, which is the total mass of logs delivered to a particular mill in a particular year by a particular possible solution. This schema includes the unchangeable schemas *PlanningHorizonDefinition*, *StandHarvestingInfo* and *PossibleSolution* and the changeable schema *AnnualLogMassForMill*. For each mill and each year in the strategic planning horizon, the function *annualLogMassForMill* is updated with the mass of logs (*ThirdOf5* (*standHarvestingInfo* (*sID*, *oID*, *yearNo*))) destined for the mill in that year, for the possible solution.

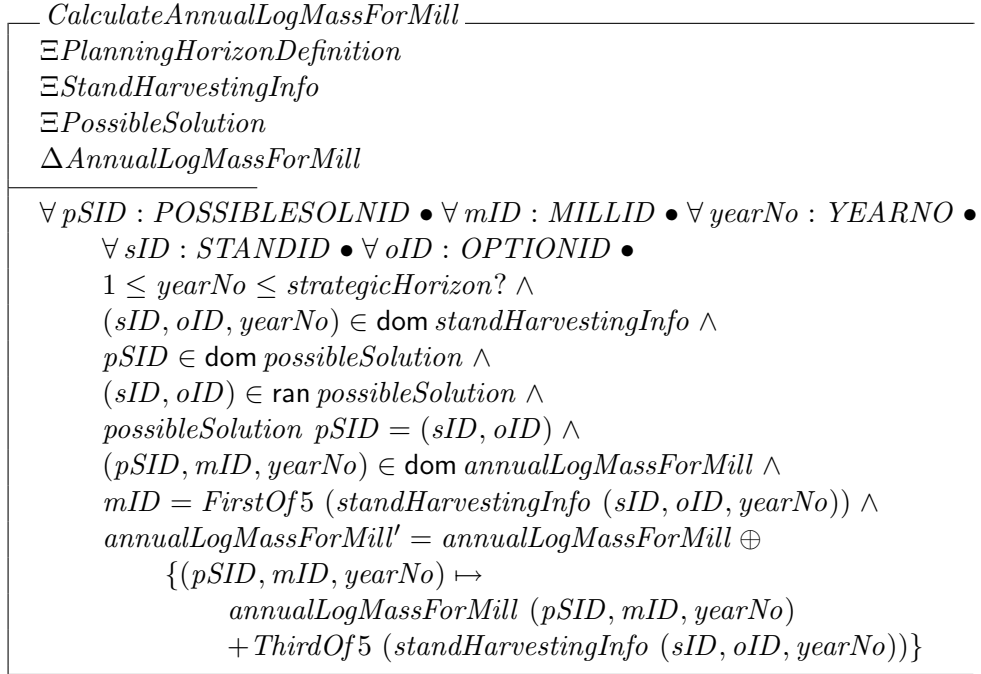


Figure 5.10 gives an example of a single possible solution and calculating the annual mass of logs which is destined for each mill for that possible solution. For each felling event, the mass of logs (which is stored in the third element of *standHarvestingInfo*) for that year and that mill are added up and stored in the function *annualLogMassForMill*. The dotted arrows downwards show when a felling event occurred. For each of these arrows, the mass of logs destined for each mill will be calculated. An example is given for the first year, where there are logs destined for Mills A and C, but none for Mill B.

The feasibility of each possible solution can now be calculated. Type *FEASIBILITY* is declared to tag whether each possible solution is *feasible* or *infeasible*. The outcome is stored in function *feasibleSolution* which is defined and initialised in schema *FeasibleSolution* and updated in schema *DetermineFeasibleSolutions*.

Type *FEASIBILITY* which can take on the values of *feasible* or *infeasible* is defined so that each possible solution can be assessed for feasibility.

$FEASIBILITY ::= \text{feasible} \mid \text{infeasible}$

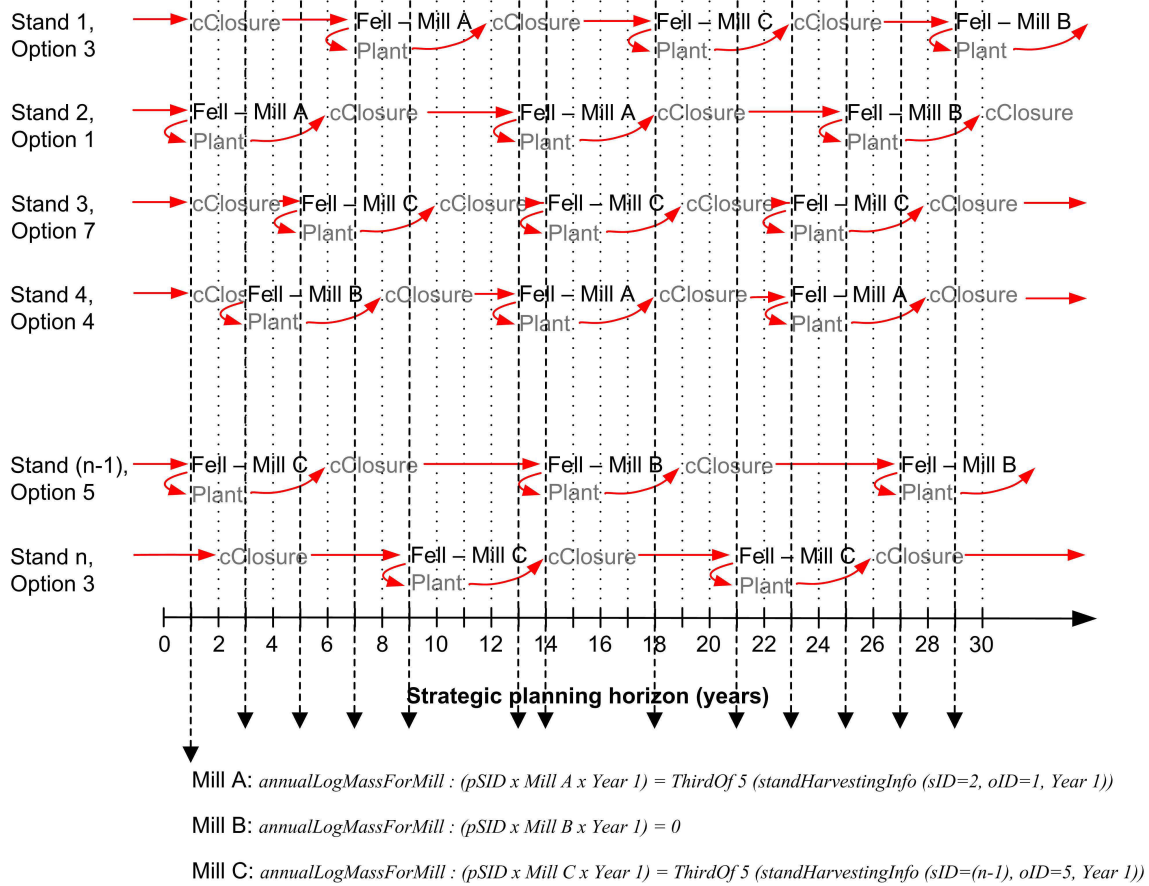


Figure 5.10: Calculation of the annual mass of logs destined for each possible mill, for a particular possible solution. The mass of logs is calculated for each year when there is a felling event in one or more stands. The calculations for the first year are shown as an example

Schema *DetermineFeasibleSolutions* determines which of the possible solutions are feasible solutions. A feasible solution is one which meets all the constraints (in this case, it meets the mill requirement constraints). Schema *DetermineFeasibleSolutions* includes five unchangeable schemas (*PlanningHorizonDefinition*, *StandHarvestingInfo*, *AnnualLogMassForMill*, *MillRequirements* and *PossibleSolution*) and one changeable schema (*FeasibleSolution*). For each possible solution, the schema checks whether the annual mill's requirements are met: for each year, the mass of logs delivered to the mill should be between the minimum and maximum required mass;

the wood property value of each stand's timber should be between the minimum and maximum allowable values. If these constraints are all met, then the possible solution is deemed feasible, and the function *feasibleSolution*'s range for that possible solutionID is updated to become *feasible*. Possible solutions not meeting the constraints are deemed *infeasible*; there is no need to update this function with this information, as the function *feasibleSolution* was initialised as being *infeasible* (see section E.6.2).

DetermineFeasibleSolutions

\exists *PlanningHorizonDefinition*

\exists *StandHarvestingInfo*

\exists *AnnualLogMassForMill*

\exists *MillRequirements*

\exists *PossibleSolution*

Δ *FeasibleSolution*

$\forall pSID : POSSIBLESOLNID \bullet \forall yearNo : YEARNO \bullet \forall mID : MILLID \bullet$
 $\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \exists woodPropID : WOODPROPID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$
 $(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$
 $pSID \in \text{dom } possibleSolution \wedge$
 $(sID, oID) \in \text{ran } possibleSolution \wedge$
 $pSID \in \text{dom } feasibleSolution \wedge$
 $(mID, yearNo) \in \text{dom } minMass \wedge (mID, yearNo) \in \text{dom } maxMass \wedge$
 $mID \in \text{dom } woodPropForMill \wedge woodPropID \in \text{ran } woodPropForMill \wedge$
 $woodPropForMill mID = woodPropID$
 $= \text{FourthOf5 } (standHarvestingInfo (sID, oID, yearNo)) \wedge$
 $(mID, woodPropID, yearNo) \in \text{dom } minWoodPropValue \wedge$
 $(mID, woodPropID, yearNo) \in \text{dom } maxWoodPropValue \wedge$
 $(minMass (mID, yearNo) \leq annualLogMassForMill (pSID, mID, yearNo)$
 $\leq maxMass (mID, yearNo) \wedge$
 $(minWoodPropValue (mID, woodPropID, yearNo)$
 $\leq \text{FifthOf5 } (standHarvestingInfo (sID, oID, yearNo))$
 $\leq maxWoodPropValue (mID, woodPropID, yearNo))) \Rightarrow$
 $(feasibleSolution' = feasibleSolution \oplus \{pSID \mapsto feasible\})$

5.3.8 Determining the objective function for all feasible solutions

The objective function can now be determined for all possible solutions which were feasible. From there, the optimal solution can be chosen: this is the feasible possible solution which has the largest objective function.

The cost of growing the trees and transporting the timber to the mill specified by the stand option is calculated in sections E.7. This entails adding up the regime costs, the short-haul and long-haul transport costs for all the stand's activities over the strategic planning horizon.

The delivered log costs are determined in section E.8.1.1. They form the income part of the profit calculation in the determination of the objective function.

The schema *CalculateAnnualMillLogPurchases* calculates the cost of logs delivered to each mill when implementing a particular feasible solution. The schema includes the unchangeable schemas *PlanningHorizonDefinition*, *PossibleSolution*, *FeasibleSolution* and *AnnualLogMassForMill* and the changeable schema *MillLogPurchases*. For each possible solution which is also feasible, and for each year in the strategic planning horizon, the cost of the mill's log purchases is calculated by multiplying the mass of logs which would be delivered to the mill (should this possible solution become the optimal solution) by the delivered log cost rate (*logCostRate* which was defined in section E.4.4).

CalculateAnnualMillLogPurchases

\exists *PlanningHorizonDefinition*

\exists *PossibleSolution*

\exists *FeasibleSolution*

\exists *AnnualLogMassForMill*

Δ *MillLogPurchases*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNO \bullet$

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$pSID \in \text{dom } possibleSolution \wedge$

$(sID, oID) \in \text{ran } possibleSolution \wedge$

$pSID \in \text{dom } feasibleSolution \wedge$

$feasibleSolution pSID = feasible \wedge$

$(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$

$(pSID, mID, yearNo) \in \text{dom } millLogPurchases \wedge$

$millLogPurchases' = millLogPurchases \oplus \{(pSID, mID, yearNo) \mapsto$

$(millLogPurchases (pSID, mID, yearNo)$

$+ annualLogMassForMill (pSID, mID, yearNo) * logCostRate)\}$

These delivered log costs are then summed to give an amount for each possible solution. This is done in schema *CalculateDeliveredLogMillCosts* (section E.8.1.1), and the total cost incurred by a possible solution is determined in schema *CalculatePossibleSolutionStandOptionCosts* (section E.8.1.2).

The next step is to calculate the total forestry costs which are incurred at company- and estate-level. These costs are incurred independently of the timber production costs. These are determined in section E.8.1.3.

Finally, the objective function can be determined. This is described in schema *CalculateObjectiveFunction* in section E.8.2, repeated below.

Schema *CalculateObjectiveFunction* updates the function *objectiveFunction* for each possible solution. This schema contains the five unchangeable schemas *PossibleSolution*, *FeasibleSolution*, *DeliveredLogMillCosts*, *PossibleSolutionStandOptionCosts* and *TotalNonStandForestryCosts* and the changeable schema *ObjectiveFunction*. For each possible solution which is also a

feasible solution, the objective function is calculated: the amount the mills pay for delivered logs less (the cost of producing the logs and delivering them to the mill, the estate-level costs and the company-level costs).

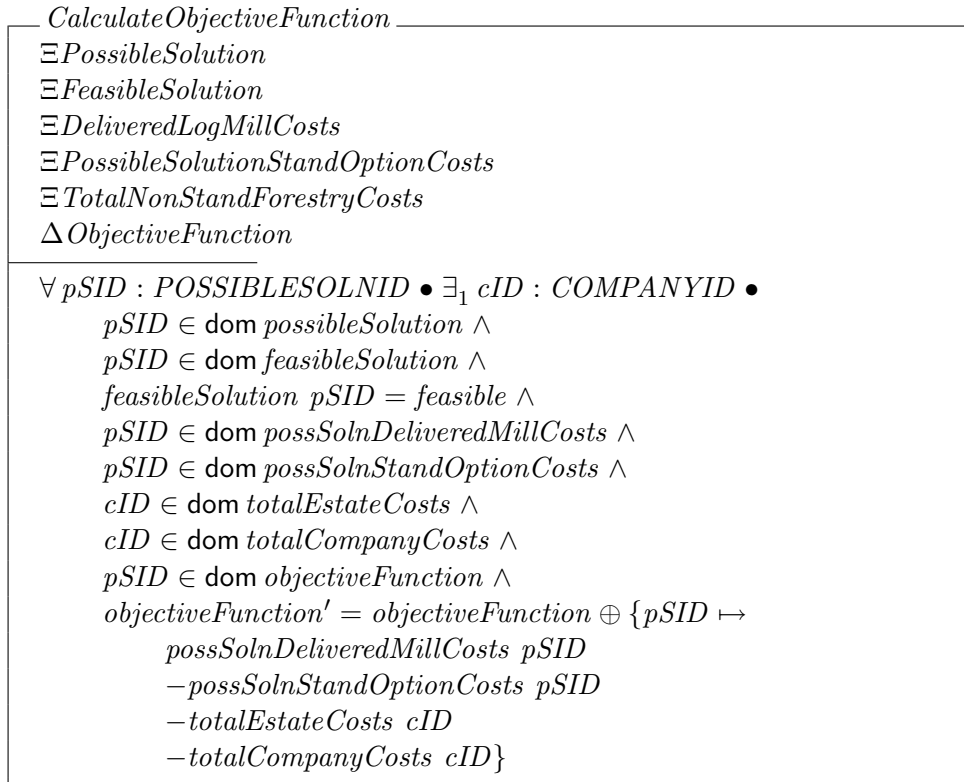


Figure 5.11 outlines how the objective function is calculated for a possible solution which is feasible. For each stand option, the costs of planting, maintaining the stand (before and after canopy closure), harvesting the stand's timber, and the costs of transporting the harvested timber to the appropriate mill are calculated and stored in *standOptionCosts*. These are shown to the right of the figure. These stand option costs are costs in the profit calculation of the objective function, so are shown as $- \text{standOptionCosts}$. These stand option costs are summed and stored in *possSolnStandOptionCosts*.

For each year in which there is a harvesting event, the delivered cost of logs to each mill is calculated and stored in *millLogPurchases*. This is the income in the profit

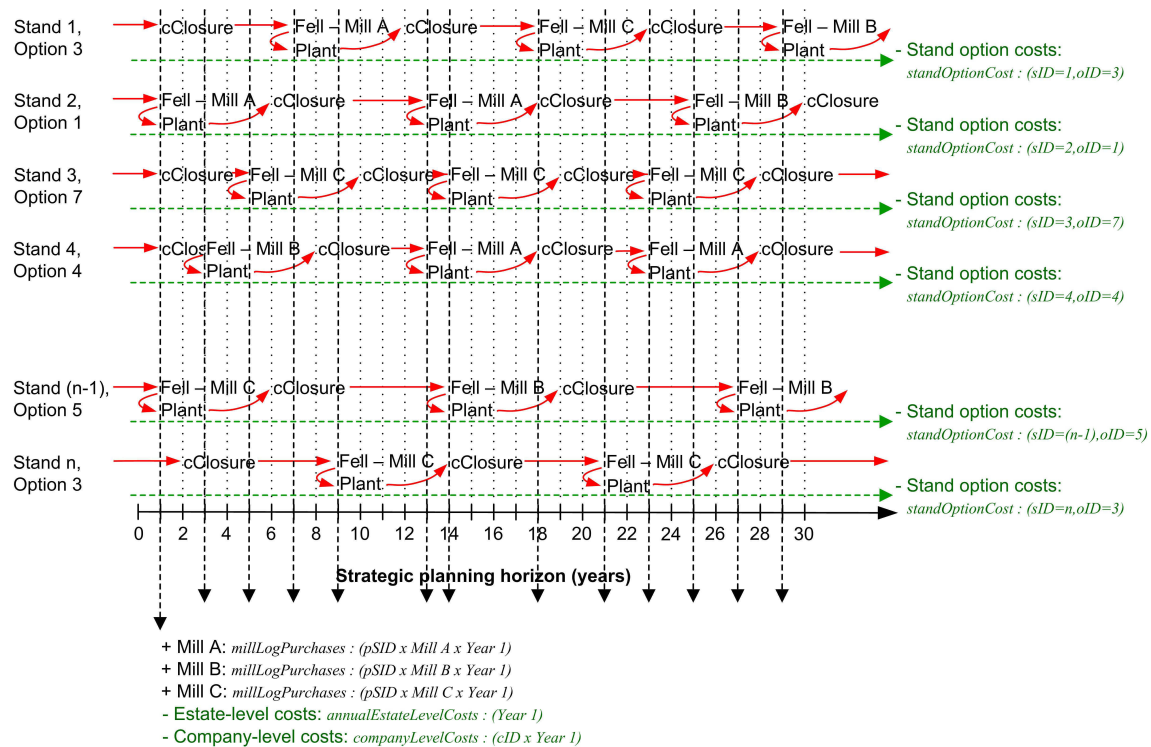


Figure 5.11: Calculation of the objective function, for a particular possible solution.

The objective function calculates the profit of a possible solution. The income is from the forests selling the logs to the mill. The costs are those of growing the trees and delivering them to the mill, as well as estate-level and company-level costs

calculation of the objective function, thus is shown as + *millLogPurchases* in the figure.

Not included in the stand option costs are the estate-level and company-level costs. These costs are defined for each year of the strategic planning horizon, and they are subtracted from the objective function.

5.3.9 Calculating the optimal solution

The optimal solution is defined to be the possible solution which is feasible, for which the objective function is at its maximum. Schema *DetermineOptimalSolution* below calculation of the optimal solution. The full specification can be found in section E.9. That section also shows how the optimal solution can be fed back into the domain description to update each stand's destination mill in function *millForStandsLogs*.

<p><i>DetermineOptimalSolution</i></p> <p>\exists <i>PossibleSolution</i></p> <p>\exists <i>FeasibleSolution</i></p> <p>\exists <i>ObjectiveFunction</i></p> <p>Δ <i>OptimalSolution</i></p>
<p>$\forall pSID : POSSIBLESOLNID \bullet \exists_1 cID : COMPANYID \bullet$</p> <p>$\exists_1 pSID1 : POSSIBLESOLNID \bullet$</p> <p>$pSID \in \text{dom possibleSolution} \wedge$</p> <p>$pSID \in \text{dom feasibleSolution} \wedge$</p> <p>$\text{feasibleSolution } pSID = \text{feasible} \wedge$</p> <p>$pSID \in \text{dom objectiveFunction} \wedge$</p> <p>$cID \in \text{dom optimalSolution} \wedge$</p> <p>$pSID \in \text{ran optimalSolution} \wedge$</p> <p>$pSID1 \in \text{dom possibleSolution} \wedge$</p> <p>$pSID1 \in \text{dom feasibleSolution} \wedge$</p> <p>$\text{feasibleSolution } pSID1 = \text{feasible} \wedge$</p> <p>$pSID1 \in \text{dom objectiveFunction} \wedge$</p> <p>$pSID1 \in \text{ran optimalSolution} \wedge$</p> <p>$\text{objectiveFunction } pSID1 = \max\{\text{objectiveFunction } pSID\} \Rightarrow$</p> <p>$\text{optimalSolution}' = \text{optimalSolution} \oplus \{cID \mapsto pSID1\}$</p>

Schema *DetermineOptimalSolution* determines which of the feasible solutions is optimal. This schema contains three unchangeable schemas (*PossibleSolution*, *FeasibleSolution* and *ObjectiveFunction*) and the changeable schema *OptimalSolution*. For all the possible solutions, there exists a single possible solution (*pSID1*) for which the objective function has the optimum value. This is the optimal solution.

Section E.9 also shows how the optimal solution can be fed back into the domain description to update each stand's destination mill in function *millForStandsLogs*, thus bringing the specification full circle.

5.3.10 The forest harvest scheduling system

The schemas of the forest harvest scheduling system which includes wood properties in harvesting decision can now be put together. There are three stages in putting them together: the first is to accept inputs, enumerate the stand options and possible solutions, and determine which possible solution is feasible. This is shown in schema *FHSSPart1* below.

FHSSPart1
PlanningHorizonDefinition
GenerateStandRegimeActionForStands
UpdateHarvestingOutcomes
CalculatePossibleSolutions
CalculateAnnualLogMassForMill
DetermineFeasibleSolutions
FeasibleSolutionsExist

Schema *FHSSPart1* includes five schema which will be called at the beginning of the forest harvest scheduling system run. The system first accepts the user's input of the strategic, tactical and operational planning horizon length, and the date on which the plan is to start. It then generates the regime actions for each stand and works out the harvesting outcome for those regime actions. The first regime in each stand option is determined so that the costs of growing the trees and transporting them to the mill can be calculated. The possible solutions are enumerated, and each possible solution is tested to see if it is also a feasible solution. The list of possible solutions is inspected to see if one or more is feasible; if so, the second part of the forest harvest scheduling system can be called.

The next step is to calculate the objective function for all the feasible solutions, and then determine the optimal solution. This is done in schema *FHSSPart2* below.

Schema *FHSSPart2* comprises of schemas which will be called if there is a feasible solution. They determine the cost of growing the trees and transporting them to the mills. They also determine how much the mills pay for their delivered logs, and what the total estate- and forestry company-level costs are over the strategic planning horizon. The objective function is determined for all feasible solutions, and the optimal solution determined.

FHSSPart2

DetermineFirstRegimeActionInStandOption
CalculateRegimeStandOptionCosts
CalculateSHaulTransportStandOptionCosts
CalculateLHaulTransportStandOptionCosts
CalculateTransportStandOptionCosts
CalculateStandOptionCosts
CalculateAnnualMillLogPurchases
CalculatePossibleSolutionStandOptionCosts
CalculateDeliveredLogMillCosts
CalculateTotalNonStandForestryCosts
CalculateObjectiveFunction
DetermineOptimalSolution

The final step is to put these together, giving the optimal solution if one or more feasible solutions exist, or giving a message saying that no optimal solution could be found if there were no feasible solutions. This can be seen in schema *FHSS*. The schema *FeasibleSolutionsExist* was defined in section E.6.3 to determine if any feasible solutions existed among the possible solutions.

Schema *FHSS* (which represents the forest harvest scheduling system) combines the two schemas *FHSSPart1* and *FHSSPart2*. It also includes the unchangeable schema *FeasibleSolutionsExist*, and two outputs, *optimalSolution!* (of type *COST*) and *message!* (of type *FEASIBLESOLN*). This schema states that *FHSSPart1* will be

run. If there are feasible solutions, the optimal solution will be calculated and the optimal objective function output in *optimalSolution!*. (Other reports, such as the description of the optimal solution, the list of logs to be transported per annum, the list of logs which will arrive at each mill's logyard per annum, etc. can also be output, but these have not been described here.) If there are no feasible solutions, a message will be output to this effect.

$ \begin{array}{l} FHSS \\ FHSSPart1 \\ FHSSPart2 \\ \exists FeasibleSolutionsExist \\ optimalSolution! : COST \\ message! : FEASIBLESOLN \end{array} $
$ \begin{array}{l} \exists_1 cID : COMPANYID \bullet \exists_1 pSID1 : POSSIBLESOLNID \bullet \\ FHSSPart1 \wedge \\ cID \in \text{dom } feasibleSolutionsExist \wedge \\ feasibleSolutionsExist \ cID = feasibleSolnsExist \Rightarrow \\ (FHSSPart2 \wedge \\ cID \in \text{dom } optimalSolution \wedge \\ pSID1 \in \text{ran } optimalSolution \wedge \\ pSID1 \in \text{dom } objectiveFunction \wedge \\ optimalSolution! = objectiveFunction (optimalSolution (cID))) \wedge \\ feasibleSolutionsExist \ cID = allSolnsInfeasible \Rightarrow \\ message! = allSolnsInfeasible \end{array} $

This part of the specification is found in section E.10.

5.4 Conclusion

This chapter used the formal notation Z to specify the forest-to-mill domains and the forest harvest scheduling system. The forestry, transport and mill domains identified in Chapter 4 were modelled in the most abstract way first. Features of the domain which would be needed by the forest harvest scheduling system were identified and added to the domain specification. The forest planning domain was not modelled explicitly in Z;

rather, relevant aspects which were identified in the semi-formal domain description were incorporated into the forest harvest scheduling system's formal specification.

The forest harvest scheduling system described in this chapter is an abstract version of the one described in Chapter 4. It harvests the stands at rotation age and chooses the mill to which it is to be sent, based on the mass of wood which would be harvested, and the mill's wood property requirements. The system specified shows how the stands' options over the strategic planning horizon are generated, and how these stand options are combined to become possible solutions (a possible solution being defined as a combination of stand options, where stands are only represented once per possible solution). Each possible solution is then tested for its feasibility: i.e. if it produces the required mass of logs for each mill, and it meets the mill's wood property requirements. Objective functions are calculated for the possible solutions which are feasible, and an optimal solution is thus found. This specification can be expanded to include other features in a future refinement. Possible features to include are given in the next chapter.

In both the modelling of the domain and the system, fewer features could be specified in Z compared to using semi-formal methods, for the same effort. However, the preciseness of the formal specification is greater than that of the semi-formal methods. In addition, it is easier to specify business rules using the formal notation, and the semi-formal models do not model exceptions.

The next chapter discusses the dissertation's results and lists areas which could be researched further.

Chapter 6

Discussion and conclusions

6.1 Introduction

In this final chapter of the dissertation, the results of the semi-formal and formal analyses (presented in Chapters 4 and 5, respectively) are discussed in five sections. The first (section 6.2) is the identification of the domains in which the forest harvest scheduling system is embedded, and the actions and constraints which apply to each. Next, the semi-formal and formal specifications of the domains and the forest harvest scheduling system (which were developed, sequentially, with reference to each other) are reviewed in sections 6.3 and 6.4 respectively. The fourth section (6.5) discusses the practice and benefits of including both semi-formal and formal methods in a specification. The specification of the forest harvest scheduling system is discussed in section 6.6. Sections 6.7 and 6.8 conclude the dissertation with a summary of its contributions, their relevance and areas which could be researched in future.

6.2 Identification of domains, and their actions and constraints

As outlined in Figure 3.1 (page 49), the domains in which the forest harvest scheduling system is embedded were identified, and lists of actions and constraints were developed for each domain. The domains are what exist at the moment (Jackson and Zave, 1993). In Chapter 4, the domains identified were the forestry, transport, mill and planning domains. The actions and constraints developed for these domains can be found in that chapter. When developing the lists of constraints, particularly for the planning domain, it was useful to be able to cross-check the lists of constraints of various systems and models, recorded in Appendix B, Tables B.1 to B.7 (pages 242 to 248).

The mechanisms used to explore these domains were semi-formal and formal analyses, the results of which are discussed in sections 6.3 and 6.4.

6.3 Semi-formal specifications

The semi-formal specifications which were presented in Chapter 4 used Hay's version (2003) of the Zachman framework (Zachman, 1987). The full framework is given in Table 3.1, page 50. The semi-formal specifications presented in Chapter 4 used the Business owner's view of the system. The models developed for each of the identified domains are summarised in Table 4.2 (page 66).

Using the Zachman framework means that many models are developed to describe different aspects of the same subject matter in a complementary manner. A different model is created to answer each of the questions "what?", "how?", "where?", "who?", "when?" and "why?", which means that there is more opportunity to discover omissions in the analyses through cross-checking and comparison with the other models.

During the development of the semi-formal models, this cross-checking brought omissions to light. For example, when developing the business process diagrams for the stand lifecycle (see Figures 4.12 to 4.15 on pages 80 to 4.15), it was realised that one of the Managing forester's obligations (previously forgotten) was to order enough stock (seedlings or cuttings) from the nursery (now in Table 4.3 on page 69).

During the development of the semi-formal models, it became necessary to ask a domain specialist about certain aspects of the domain. As stated in Chapter 3, this is expected (Meyer, 1985; Bryant, 1990; Hall, 1990), and is the purpose of doing the analysis. For example, it was thought that the Timber logistics manager only assessed the forestry plan at an operational level. After consulting a domain specialist, it was discovered that the Timber logistics manager had inputs at all three planning horizons (strategic, tactical and operational). Adjustments to the planning business process diagrams (Figures 4.41 to 4.58 on pages 114 to 128) were made accordingly.

As part of the iterative process of developing semi-formal specifications, the specifications were reviewed by domain experts. Very few errors were found. The Planning forester highlighted the fact that it is not only logs which are transported from the plantation forest to the mill; sometimes the log-making activity is delayed, and takes place at the depot or siding, or at the mill's logyard. One of the other forestry domain experts highlighted spelling errors and entities or processes which were not clear; these were changed.

When developing a forest planning system for use in a Portuguese forestry company, Ribeiro *et al.* (2005) also used the Zachman framework in their analysis. Their paper discusses their company's strategic planning process, and their proposed system. The planning process they describe is iterative, like the process which is described in section 4.6.5. Beaudoin *et al.* (2008) comment that the planning process is made up of many people giving inputs at many different hierarchical levels. This was also

found and reported in the analysis of the planning hierarchy of this study (section 4.6.5).

Baskent *et al.* (2001) present an object-oriented analysis of natural forest management. Because it is aimed at natural forests, spatial aspects are particularly important. However, certain aspects modelled in this work have similarities or counterparts in plantation forestry management. For example: a group of stands contributes to an age-class of the forest (as described in section 4.6.3); ‘Treatments’ are similar to regime actions, as described in section 4.3.4; and ‘Management units’ are similar to the management levels described in section 4.3.2.

6.4 Formal specifications

The formal specifications were presented in Chapter 5 and in Appendices D and E. As discussed in section 6.2, at the beginning of the formal and semi-formal analyses, domains were identified, and a list of actions and constraints for each domain developed. The actions for each of the domains helped to identify the Z action schemas and the constraints helped to identify the predicates in the schemas.

The “established strategy” of capturing the system’s normal state, initialising it, writing schemas which would change the state, and writing exceptions (Barden *et al.*, 1994, pp.20–21) was followed for the second abstraction of the forest-to-mill domain (section 5.2 and Appendix D) and for the specification of the forest harvest scheduling system (section 5.3 and Appendix E). Care was taken to ensure that the initialisations did not contradict the system’s normal state. If it was found that a contradiction had occurred, either the initialisation or the system’s state was changed. As can be seen in the system’s context diagram (Figure 4.59 on page 134), and from the list of system inputs given in section 4.7.3, there are many inputs from databases. Future refinements of the specification could include database data integrity checks.

As discussed in Chapter 2 (section 2.6.3) and Chapter 3 (section 3.3.5), the level of abstraction used in the specification is important. It is also difficult to gauge the correct level of abstraction to use (Nash, 1990; Tretmans *et al.*, 2001). The specification must be abstract enough to ignore unnecessary detail, or detail which implies an implementation bias. However, it must also be explicit enough to describe the system sufficiently (Hall, 1990; Wing, 1990; Barden *et al.*, 1994, pp.164–165; Bowen and Hinchey, 1995b).

The approach taken in this dissertation was to use an ERD which described the forest-to-mill domain very succinctly. This was then described in Z in the most abstract way possible (see section 5.1). The next step was to think ahead to the forest harvest scheduling system. Anything that would be required by that system would then need to be included in a more detailed specification. Highlights of this detailed specification are given in section 5.2. The full specification of the forest-to-mill domains is given in Appendix D. The specification of the forest harvest scheduling system is given in Appendix E, and highlights are given in section 5.3.

An example of the abstraction of the specification is the issue of the description of the genetic material of the trees planted in the stand. In the first (abstract) specification, it is denoted as *SPECIESID*, and this is not expanded upon. This use of the term “species” actually mirrors common parlance amongst foresters: they use it to denote a genus, or a genus and species, or a clone or hybrid, or any kind of term used to describe genetic material. In the second abstraction, genera, species and hybrids were described. As clonal forestry is becoming increasingly important for intensively-managed plantations (Kanowski, 1997), the distinction between seedlings and clones could be included in future versions of the specification. Another aspect which could also be included is the species’ provenance, which is currently recorded in the plantation database (W. Esler, 2002, pers. comm.¹). Other issues which could be included in the Z specifications

¹Mr W. Esler, Planning Forester, Mondi Forests, P. O. Box 39, 3200 Pietermaritzburg, South Africa

are listed in section 6.8.

During the development of the formal specification, aspects which were unclear came to light. These were clarified with domain experts, and the specification changed. An example is that it was unknown before writing the Z specifications that part-loads of logs at roadside would be left there to rot, as it is not worth the short-haul trip. Another example is that it was not appreciated that logs would be stored at the depot in piles according to the mill for which they were destined. These aspects can be seen in schemas *LoadTruckatStand* and *TransportAndUnloadTruckAtDepot* on pages 165 and 167 respectively.

Two types of reviews of the Z specification were undertaken during and after the development of the specification, by the specification developer and by Z experts. The internal review undertaken by the specification developer relied on literature to point to good practices in specification development.

Towards the end of the development of the second abstraction Z specifications of the domain, a review of the specification was undertaken by the specification developer. At the time of this review, Barden *et al.* (1992) (the results of a survey of Z users in the U.K.) was read, which helped highlight the features of a good Z specification. In this survey, it was reported that the median schema length was 6–10 lines (with a range of 1 to 20), and the proportion of Z “code” to English narrative aimed at was 1:1. As a result, long schemas were inspected and divided where possible; the amount of narrative description of the Z schemas was also increased. As a result of the review, the average schema length for the forest-to-mill domain specification (Appendix D) dropped from 25.5 to 11.4 lines (with a range of 1 to 48). The ratio of the Z lines of code to lines of English narrative improved from 59:41 to 53:47. After the forest harvest scheduling system specification had been developed (Appendix E), a similar analysis found that the average Z schema length was 9.5 lines (with a range 1 to 51), and the ratio of the

Z lines of code to lines of English narrative remained at 53:47.

An issue raised in the internal review was the use of the power set (\mathbb{P}). Reading Barden *et al.* (1994, p.167) highlighted the fact that power sets could have an infinite number of elements which would not be countable, whereas the elements of a finite powerset (\mathbb{F}) can be counted. As a result, instances of power sets in the specification were changed to finite power sets.

During the development of the forest-to-mill domain specification, the specification was reviewed by Z experts. In one of these reviews, the use of function types was questioned. In this review, too, a request was made for shorter schemas and more narrative text. As a result of the review, the code was inspected and found that in fact there were no functions specified which should have been an injection (1-to-1 relationship) or partial injection. The existing predominance of functions (\rightarrow) and partial functions (\twoheadrightarrow) is in line with other specifications reviewed where these two function types were found to be the most commonly used type of functions (Barden *et al.*, 1992).

In a subsequent review of the forest harvest scheduling system specification, the way in which functions were initialised in the system (to \emptyset), and then given an initial value of zero was changed from being defined in one step, e.g.

$$annualLogMassForMill' = annualLogMassForMill \oplus \{pSID, 0\}$$

was initialised first as

$$annualLogMassForMill' = \emptyset$$

and then initialised to zero:

$$annualLogMassForMill' = annualLogMassForMill \cup \{pSID, 0\}.$$

The second way represents reality better.

During the reviews, the logic of the schemas were reviewed, and where necessary, adjustments made. These reviews confirm how errors or improvements can be identified

through code walkthroughs (Boehm, 1985).

No proofs were undertaken when developing these specifications, although when they were developed, care was taken to keep them consistent with (i.e. not contradicting) each other. For example, when initialising the normal state, Barden *et al.* (1994, p.22) suggest that one should prove that this initial state is a state of the system. Although this proof was not explicitly written down in the specification, when developing each initialisation schema, the initialisation and normal states were checked that they did not contradict each other.

Gaudel (1994) calls this approach of using formal methods but not doing the proofs a formalised approach (as opposed to a formal approach), while others call it “level 0 formal methods” (Bowen and Hinchey, 2006), “formal methods light” (Jones, 1996; Hinchey, 2002) or “lightweight formal methods” (George and Vaughn, 2003). In spite of the fact that no proofs were undertaken, there were still benefits of undertaking the formal analysis. Much more intense thought had to go into the development of the formal specifications than for the semi-formal analysis; it forced questions to be asked that otherwise would have remained unasked (Meyer, 1985; Hall, 1990; Hinchey, 2002). For example, the fact that logs are put into piles according to the mill for which they are destined at the depot would have remained unknown if the formal description had not been generated.

The use of a tool is important when using formal methods (Ince, 1990; Luqi and Goguen, 1997). The type checker used (ZTC) was sufficient for the task. However, there were limitations with this tool. When writing lengthy statements in the schema’s body, for example

```

SchemaName
...
∇... • ∃... •
... condition 1... ∧
... condition 2... ∧
... condition 3... ∧
... condition 4

```

one cannot include comments next to one of the lines of the conditions, as this causes the type checker to think that there are errors in the code. Comments have to be added to the end of such statements, which may be later than one would want in order to have code which has good readability and maintainability.

Another constraint was that ZTC required all of the \forall and \exists statements to be clustered together, so one could not write code of the form:

```

SchemaName
...
∇... •
... condition 1... ∧
... condition 2... ∧
∃... •
... condition 3... ∧
... condition 4

```

Being able to break up the cluster of \forall and \exists statements would have made the Z more readable.

In the schemas checked by ZTC, one could not specify functions and give them a value in the same schema. This meant that all functions had to be declared in one schema and then initialised or updated in another. A final limitation found was the fact that one could not specify real numbers; instead they had to be specified as integer, and the narrative had to contain annotations to the effect that a real number was required.

6.5 Combining both semi-formal and formal specifications

When including semi-formal and formal methods in a specification, there are two approaches. The first is to attempt to generate or transform the semi-formal method output into a formal specification. The other is to develop complementary specifications Bowen and Hinchey (1995a). As mentioned in Chapter 3 (section 3.1), the latter approach was taken. In Mander and Polack's method (1995) of transforming entity-relationship diagrams (ERDs) into Z schemas, the entity would become a state schema. In the Z specifications presented in Chapter 5 and Appendices D and E, the state schemas sometimes correspond with entities in the ERDs, and sometimes not. However there is a closer correspondence between Z and the state charts. For example, in the state chart of the forest-to-mill domain (see Figure 5.3 on page 154), the state represented by a circle or state is similar to the state schema, while the activity which causes the state to change can be compared to a Z action schema. A benefit of using Z specifications in addition to semi-formal specifications is that Z's established strategy encourages the development of exceptions to the action schemas. These are useful as a starting point from which to develop test criteria. Another benefit of using the formal notation is that the business rules of the domain and the system are captured more precisely with the predicates.

Just as the Zachman framework describes different aspects of the same domain or system, using semi-formal and formal methods helps to describe the same thing but in a different way, thus giving a more rounded view of the subject. Hall (1998) suggests that different tools are needed for different tasks. In the semi-formal specifications which were presented, there were aspects described which were not modelled in Z. These include clones and seedlings, and provenances (genetic material), working circles (end-uses) which are decided before the stand is planted initially, the hierarchies of land management (and decisions which could be made for or rules applied to a partic-

ular management level), land ownership, regimes which cater for the trees to coppice, bucking patterns (for sawtimber and other end-uses), and other transport methods (in addition to road).

Using the formal notation in addition to the semi-formal description was beneficial. The semi-formal analyses are more accessible to clients and domain experts (for example, the forestry expert who did not understand Z only read the English narrative as reading the Z itself was too daunting). The formal notation, however, captures a precision which is lacking in the semi-formal approach. As one needs to be precise when describing domains and systems with Z, issues were uncovered which would have remained obscure if the formal approach had not been taken (for example the fact that at depots, logs are put into piles according to their destination mill). This confirms the statement which Meyer (1985), Bryant (1990), Hall (1990), and Barden *et al.* (1994, p.327) have made in favour of using formal methods: they make you think. Using both approaches helped get the requirements right before further development is undertaken (Bowen and Hinchey, 2006).

When comparing the semi-formal and formal descriptions of the forest-to-mill domains, and the forest harvest scheduling system, the semi-formal descriptions covered more features. However, the formal description gave more detail of how the different aspects of the domain fitted together. This detail would have had to be inferred from the semi-formal description or interpreted from the accompanying English text. Using the “established strategy” of capturing the system’s normal state, initialising it, writing action schemas and then writing exception schemas has the advantage over the semi-formal approach in that initialisations and exceptions are not covered in any of the modelling techniques used.

As described in Chapter 2 (section 2.6.2), it is proposed that semi-formal methods (diagrams) be included in formal specifications, as this would aid understanding and help

those who do not understand the formal notation (Semmens and Allen, 1991; Semmens *et al.*, 1992; Polack *et al.*, 1993; Barden *et al.*, 1994, p.17; Bowen and Hinchey, 1995a; France and Larrondo-Petrie, 1995; Mander and Polack, 1995; Bruel *et al.*, 1998; Hinchey, 2002). The semi-formal diagrams of Chapter 4 which described aspects of the forest-to-mill domain generally showed more detail than was implemented in the Z specifications. In Chapter 5, diagrams showing what was actually implemented in the forest-to-mill domain were included, together with references to the original diagrams in Chapter 4. In the Appendix containing the domain specification (Appendix D), both the more detailed diagrams (from Chapter 4), and those depicting what was actually implemented in Z, were included in the text to aid understanding. In the text describing the forest harvest scheduling system (section 5.3 and Appendix E), diagrams were added to aid the understanding of the enumeration of the stand options and possible solutions.

6.6 Forest harvest scheduling system specification

The forest harvest scheduling system specified in this dissertation aims at optimising the whole supply chain, rather than allowing some parts of the chain to be optimised at the cost of the whole chain to the detriment of the organisation or supply chain (Pearlson and Saunders, 2006, pp.109–132). With this system, a long-term plan is generated, which incorporates some tactical constraints (to make the long-term plan more realistic). The system gives opportunities for foresters who have more on-the-ground knowledge to change aspects of the plans: during the planning phase, as well as during the implementation of the plans. This feature gives the Managing foresters the support to make and implement decisions which they need to take, without impacting on the plan for the company. In this way, all participants can contribute to a common goal, without feeling that the system is imposing decisions on them. The system includes timber allocation, which means that timber can be sent to the closest appropriate mill. The system also includes wood properties as an acceptance criterion for each mill's process, which means that the stands' properties must be evaluated for each of the possible

harvesting years (as wood properties change with age). The solver then chooses the most appropriate year to harvest the stand, and the most appropriate mill's process to send it to.

The two versions of the optimisation (objective function) specified in the semi-formal version of the specification (section 4.7.1) mean that the forestry part of the integrated forestry and pulp and paper company can check their costs, and the whole supply chain can be optimised. Optimisation option two (see page 131) includes the mill's fixed costs, thus causing the model to feed the mill, rather than selling the timber to another (external) mill. The inclusion of energy and chemical costs, salaries and maintenance costs in this cost may seem to some (especially cost accountants) to be too "general". However, this approach is based on the fact that whether 1 million tonnes of logs are being processed, or 1.1 million tonnes, the energy, chemical, maintenance and human resource costs will probably remain more or less constant. This approach has been used with good effect in the sawmilling industry in a sawmill production planning system (Wessels and Price, 1999; Wessels *et al.*, 2006).

The feature which allows the possibility of "buying in" timber is beneficial, because it will help the optimisation solver to produce a feasible solution. The list of timber to be "bought in" gives the users an idea of what kind of timber (in terms of its genetic material, wood properties and timber mass) is still required by the mills. Together with the timber which the plantation produces and which has been allocated to a mill, this forms the requirement of the mills over the strategic planning horizon. The buying in of timber was also used in the sawmilling production planning system mentioned before (Wessels and Price, 1999; Wessels *et al.*, 2006).

In the forest harvest scheduling system described by Ribeiro *et al.* (2005), scenarios were stored, one of which would become a long-term plan. Their system took account of constraints (restrictions), information from a plantation database and regimes (man-

agement prescriptions), and had inputs from a Growth and yield models database. This is similar to the system specified in section 4.7.2 (see Figure 4.59 on page 134). In addition, the tactical plan of Ribeiro *et al.* is created from the first three years of the strategic plan, which is in agreement with the planning process described in Figure 4.34 on page 107.

The forest harvest scheduling system specified using Z (in Appendix E) describes an abstract form of the first option specified in section 4.7.1. In the formal specification, wood from the plantation's forests is accepted at mills if the mill needs that mass of logs to process, and if the wood properties of the timber is within the limits specified by the mill. The system specified the profit to the forestry part of the integrated forestry and pulp company is calculated. If there is at least one feasible solution, the optimal solution can be found (i.e. the feasible solution which generates the most profit over the strategic planning horizon).

In this first abstraction of the formal specification, the timber is harvested at rotation age (as opposed to choosing to harvest in one of the years just prior to or just after the rotation age), and there is an assumption that each mill has only one process. In addition, the timber 'buying in' feature has not been specified, and constraints which would ensure the generation of a more realistic tactical plan have not been added. These features could be added to future refinements.

Defining the forestry, transport and mill domains was beneficial for the development of the forest harvest scheduling system specification. The semi-formal specification of the domain helped to highlight features, issues and constraints which needed to be taken into account when specifying the system. In addition, although the formal description of the forest-to-mill domains could stand alone as a specification in its own right, sixty percent of the domain specification was incorporated into the forest harvest scheduling system specification.

The size of problem which the forest harvest scheduling system is attempting to solve is large, as can be confirmed from literature (Kent *et al.*, 1991; Beaudoin *et al.*, 2008) as well as the computations of numbers of stand options and possible solutions (see sections E.5.4 and E.6.1). The problem would be significantly larger with the addition of varying harvesting dates for each stand. In the forest harvest scheduling system, a logical way of determining the feasible and optimal solutions (if any exist) was described. This specification was not meant to replace the use of Operations Research techniques which find optimal solutions. The Z description was intended merely to demonstrate how an optimal solution for the system could be reached. In the Operations Research formulation of this system, the combinatorial problem size will have to be taken into account.

6.7 Conclusions

The aim of this dissertation was to use semi-formal and formal specification techniques to specify a strategic plantation forest harvest scheduling decision support system which includes wood properties in the harvesting decision. The inclusion of wood properties is important because wood has variable properties; mills would prefer a more uniform wood intake so that their processes can be more efficient and their product more uniform.

This goal can be broken down into sub-goals:

1. Specifying the plantation forestry domain using semi-formal and formal methods. This aims to capture the “agricultural” (silvicultural) aspects of plantation forestry – how trees are grown as a crop.
2. Specifying the forest-to-mill domain using semi-formal and formal methods. This aims to capture the business aspects of plantation forestry, i.e. the logistical, financial, management and environmental aspects.

3. Giving a computer science/information systems specification of the strategic (long-term) plantation forestry harvesting scheduling plan to ensure that there is always a sufficient mass of wood for the mill. This includes appropriate aspects of forest management, including logistical, financial and environmental constraints. It is not an Operations Research description (i.e. mathematical formulation) of the problem.
4. Adding wood properties to the strategic plantation forest harvesting plan to ensure that the mill receives wood of a consistent quality.
5. Incorporating logistical constraints (which are usually included in the tactical plan) to refine the strategic plantation forest harvesting plan and make it more realistic.

The first four goals were achieved using both semi-formal and formal methods to capture the specification. The fifth goal was achieved using only semi-formal techniques. In all cases, the formal specifications described fewer features (i.e. were more abstract) than the semi-formal specifications.

It was found the using semi-formal methods meant that more features could be described, but not with as much precision as the Z description. There was no way of checking the consistency (or, of checking that there were no contradictions) in the semi-formal method. Cross checking of these specifications had to be done by humans (the specification writer, and reviewers). Even so, it is possible that issues could be missed because of interpretation. Using the Zachman framework helped to counteract this lack in semi-formal methods, because the problem was described from many points of view.

Because of the preciseness required by the Z specification, fewer features could be incorporated with the same amount of effort, compared to the semi-formal specification. However, the system's business rules are better captured with the formal notation. The formal notation also promotes the development of exception cases, which are not cap-

tured at all in the semi-formal method.

The formal specification had the advantage that automated tools (like the type checker and development environment) could be used to check the consistency of the description. Human thought was still needed to make sure that one schema's logic did not contradict another's. The use of additional tools, such as animators and provers, would have helped in this regard, and would have given the assurance that what had been developed was robust.

Using both the semi-formal and formal methods to describe the system allowed for the all features of the system as well as the detail of the system to be captured. If readers do not understand the formal description, they can read the semi-formal description.

6.7.1 Contributions

The contributions of this dissertation are:

1. A specification of the forest harvest scheduling system which includes wood properties in the harvesting decisions using semi-formal methods (Chapter 4, section 4.7).
2. A specification of the forest harvest scheduling system which includes wood properties in the harvesting decisions using formal methods (Chapter 5, section 5.3 and Appendix E).
3. An analysis of how including wood properties in the harvesting decision would impact on the business processes and decisions taken, as described in the various domains (Chapter 4, section 4.7.6, Appendix C).
4. An analysis of the forest-to-mill domain using semi-formal methods, from the Business owner's point of view, using the Zachman framework (Chapter 4).

5. An analysis of the forest-to-mill domain using formal methods (Chapter 5 and Appendix D).
6. An analysis and tabular summary of log sorting schemes (reported in the literature) which use wood properties as their sorting criterion (Chapter 2, section 2.4.4).
7. An analysis and tabular summary of the aims, features, use, mathematical formulations and solution times and approaches of the forest harvest scheduling systems which appear in the literature (Chapter 2, section 2.5 and Appendices A and B).

6.7.2 Benefits of the contributions

The contributions made in the dissertation have the following benefits:

Forest harvest scheduling system which ensures delivery of more uniform logs to mills The forest harvest scheduling system described in this dissertation includes wood properties in the harvesting decision, which means when using it, mills will be processing a much more uniform mass of logs together. This has the added benefit that the processes can be adjusted to process the wood more efficiently. In addition, wood will be allocated to the most appropriate process, which may mean that less wood can be used to produce the same amount of pulp.

Forest harvest scheduling system which ensures most appropriate allocation of logs to mills The incorporation of transport distances and costs in the forest harvest scheduling specification means that the timber will be allocated to the closest appropriate mill, thus saving costs. Since the forest harvest scheduling specification includes the ability to “buy in” timber, it is less likely that no optimal solution will be obtained on running the solver. The timber to be “bought in” gives the users a good idea of what type of timber is lacking in the plantation. The timber produced by the plantation which could not be allocated to any mill’s process gives foresters a better idea of what

to plant in order to meet the mills' demands more closely.

Research into forest harvest scheduling systems The analysis and tabular summary of the forest harvest scheduling systems can be used by those who wish to investigate the field of forest harvest scheduling systems further. It gives a chronological overview of the systems and models reported on, by planning horizon. The aims and mathematical approach to solving each would benefit those wanting to make a contribution to this area.

Systems development/process improvement in the forest-to-mill supply chain

The domain of the forest-to-mill supply chain which has been captured using both semi-formal and formal methods can be used as a starting point by those who wish to develop software or improve the processes in these domains. It would help those developing user requirements statements (Kang *et al.*, 1990; Sutcliffe and Maiden, 1998), in that they could give the semi-formal analysis and the text of the formal analysis to their clients for them to compare with their current practices; the requirements analyst would not have to work from first principles. Because the Zachman framework was used to structure the semi-formal analysis, there is more detail about each domain. The analysis is also more likely to be complete, because of the cross-checking involved. The domains in which the forest harvest scheduling system is embedded which was captured using semi-formal and formal methods can be referred to by those developing the system further, e.g. by designers, developers, testers and maintainers of the system (Barden *et al.*, 1994, p.9; Burton-Jones and Meso, 2008). The documented domains would help them understand the domain better (Greenspan *et al.*, 1982), resolve issues instead of guessing, and it could also be of use in developing test cases and a user manual.

'Real life' specification to help others learn Z The use of a formal method to describe the domains in which the forest harvest scheduling system is embedded, and to specify the system, means that a more precise description of both now exists. This

could be regarded as a non-trivial example to help others learn Z. Luqi and Goguen (1997) comment that often only “toy” examples are used to teach formal methods or are used in papers. The specifications contained in Appendices D and E could contribute to the body of Z specifications to which others could refer.

6.8 Future work

There is much scope for work emanating from this dissertation. This work falls into two categories: expanding on what has already been presented, and using it to develop the system further. The work which has been done can be expanded in the following ways:

Specification development The specification could be improved by including proofs of the properties of the system, for example, that the system can reach its normal state from the initial conditions, or conformance between two abstractions of the specification (Hall, 1990). This could be undertaken by hand proofs, or by using a theorem prover or checker (Bowen and Hinchey, 2006). Animating the specification and generating test cases would be beneficial in the further development of the specification.

Expanding the domain specification Several aspects could be included in the domain specifications to refine them, such as adding clones and seedlings, and provenances to the specification of genetic material; including regimes which allow coppicing; including user-enforced dates to be specified in regimes (e.g. planting *will* occur on this date); describing the hierarchy of forest land management, and allowing decisions made at a user-defined higher-level to be cascaded down to all lower elements of the hierarchy; including land ownership in the management hierarchy; implementing weather-related constraints for planting or harvesting; including the fact that stands were managed according to environmentally sound principles (like FSC); including the stand’s planting

purpose (sometimes called working circle), which is useful for other end-uses such as sawmilling; including bucking patterns (also useful for sawmilling); including an alternative long-haul transport method (e.g. rail); adding capacity constraints to depot log piles and mill logyards. The specifications could also be expanded to cover areas such as plantation forestry aimed at sawtimber production (which is more complex than pulpwood production), or natural (uneven-aged) forestry.

Expanding the forest harvest scheduling system specification The forest harvest scheduling system could be refined by incorporating features such as: allowing a stand to be harvested just before, just after or at rotation age; including more than one process per mill; allowing timber to be 'bought in' to ensure that a feasible solution will always exist; including different long-haul transport options; allowing varying (distance-based) costs for short- and long-haul timber; including both the *option one* and *option two* calculations for the objective function (i.e. optimising forestry profit only, or the whole supply chain); allowing foresters to swap similar stands after the strategic plan has been made; incorporating tactical-level constraints; and developing reports for all the different types of users who may interact with the system.

Designing the system This would include: developing Architects' and Designers' views of the semi-formal analysis (see Table 3.1 on page 50), in preparation of the coding of the system; developing a formulation of the optimisation problem (an approach, such as outlined by McNaughton *et al.* (2000), Cea and Jofré (2000) or Andersson (2005) is proposed, as they incorporate aspects of the tactical plan into the strategic solution); coding, testing and deploying the system.

6.9 Concluding remarks

In an industry which produced 192 million tonnes of pulp in 2007 (FAO, 2007), an opportunity exists to ensure that pulp mills process the most appropriate wood (wood which has similar wood properties) and to improve the business processes in the forest-to-mill supply chain. Obtaining a global optimum for the supply chain is better than the individual parts of the chain creating optima in their silos or cost areas (Pearlson and Saunders, 2006, pp.109–132). The system specified in this dissertation fills this need.

Appendix A

Summary of forest harvest scheduling systems or models described in the literature

Many systems and models have been developed over the last forty years to help forest-based companies manage their operations. Each system or model has its own particular aims, focus and intended planning horizon. However, it should be noted that different authors are not consistent with their use of terminology when describing the planning horizon (e.g. tactical could mean 30 years to 6 months).

The tables in this appendix give an overview of some forest harvest scheduling systems/models that have been described in the literature. Systems/models were included in the table if they apply to plantation forests, or could be used for plantation forests. Systems/models applying to the forest-to-mill supply chain were also included in this assessment. A series of tables are given: for the strategic, the strategic/tactical, the tactical and tactical/operational and the operational planning horizons. The rows of these tables are ordered by the year in which the literature referring to the system/model was published. Tables A.1 – A.4 give the literature reference, the product name, the planning horizon that the system/model is intended for, the aim for which the product was developed and the technique used to calculate a solution. Tables A.5 – A.8 give background details, such as where the system/model was developed/has been used, how often it is used, whether it is intended for plantation or natural forest, and the number of species for which it can be used (this is important for multiple species/genera situations). Tables A.9 – A.12 include feature information about the system/model, such as whether or not harvesting, bucking, timber allocation, transportation and wood properties were included. It also includes comments about the system or model. If a table cell is empty, it means that the literature was not clear or explicit about this feature or aspect.

Table A.1: Product aims of strategic forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Planning horizon	Product aim	Technique
STRATEGIC					
1	Navon (1971); Chapelle <i>et al.</i> (1976); Tedder <i>et al.</i> (1978)	Timber RAM	Strategic (up to 350 years); tactical (10 years)	Concentrates on timber production. Road making and other spatial concerns ignored.	
2	Walker (1974); Tedder <i>et al.</i> (1978)	ECHO - economic harvest optimization model	Strategic	Simulator. Harvest scheduling options: area control, volume control, 3 types of even flow, economic harvest optimization (ECHO) based on net present worth or present net benefit maximization. Has ability to change from natural to managed forest, or to change management intensity once. Cannot simulate thinnings or multiple productivity classes.	
3	Barber (1983)	HARVEST	Strategic	Development aim: respond to the National Forest Management Act (1976) which required all plans to "provide for multiple use and sustained yield of goods from the National Forest System in such a way that maximizes long-term net public benefits in an environmentally sound manner" (Federal register, 1982). Aims to provide a framework for viewing the multiple use planning problem on a national forest (Kent <i>et al.</i> , 1991). Includes road making decisions; specifies when to perform activities, e.g. harvesting (Field, 1984; Church <i>et al.</i> , 2000); allocates forest land to general management objectives and schedules the treatments and resulting product flows (Field, 1984).	LP; cardinally-weighted goal programming available with Version 1 (Field, 1984). MIP solved as an LP (Kent <i>et al.</i> , 1991).
4	Johnson <i>et al.</i> (1980); Johnson <i>et al.</i> (1986); Garcia (1988); Kent <i>et al.</i> (1991); Weintraub <i>et al.</i> (2000)	FORPLAN	Strategic	To model strategic decisions such as which silvicultural regime to use and what land to acquire while ensuring that timber volume production is maintained. Considers the data and decisions in a "very aggregate" form. Aggregates stands of similar site characteristics, tree age, planting density and geographical location to form a macro-stand. Trees become "export", "sawmill" and "pulp" logs. Estimates volumes with inventory-growth simulations.	LP
5	Epstein <i>et al.</i> (1999a); Morales and Weintraub (1991)	MEDFOR	Strategic (50 years)	Deterministic simulation model for long-term forest management analysis. Allows analysis of different silvicultural and harvesting options, and calculates the associated cash flow, profitability and forest state. Regimes are mainly oriented towards timber production, although ecological, scenic and conservation-oriented regimes can also be implemented.	Simulation

Table A.2: Product aims of strategic/tactical forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Planning horizon	Product aim	Technique
STRATEGIC/TACTICAL				
7 Barros and Weintraub (1982)		Tactical (50 years); supports some strategic planning	To ensure that the sawmill, pulp mill and port are allocated the required amount of timber from the plantation for each time period. There are 3 types of wood sinks: export, sawmill and pulp mill. Chips from sawmill can be used in pulp mill. The road network may need construction and maintenance, but that is not included here.	LP
8 Garcia (1984); Garcia (1990); Manley and Threadgill (1991); Papps and Manley (1992); Manley et al. (1996)	FOLPI (Forestry-Oriented Linear Programming Inter-preter)	Medium to long range (tactical and strategic – has been used for planning horizon of 90 years).	To support decisions on harvest scheduling, log allocation, forest management planning, sustainable levels of harvest production and investment of processing plants.	LP
9 McGuigan and Scott (1995)	RegRAM-1 (Regional Resource Allocation Model)	Long to short-term (depends on user inputs)	Multi-period version of LOGRAM which includes harvest scheduling and resource allocation. Decides how much of a particular croptype to fell each year, how much resource (logs, chips) to buy in, where to send them, how to transport them there.	LP, simulation
10 McNaughton et al. (2000); McNaughton et al. (1998)		Strategic (30 years) and Tactical (6 years)	To solve tactical planning problem (which specific stands to fell, which roads to construct, which stands to fell in close time intervals to each other) at the same time as solving the strategic planning problem in order to produce a feasible, working plan, and avoid conflicts between the two.	LP with column generation and constraint branching
11 Cea and Jofré (2000)	(Strategic part) (Tactical part)	Strategic (45 years) Tactical (3 years)	Two-level model to integrate strategic and tactical forestry planning so as to reduce differences between strategic and tactical model outputs.	MIP; strategic solves it with LP using branch and bound. Iterate through road and tactical problems by using Simulated annealing and LP solver.
12 Syndicate Database Solutions (2006)	Microforest Scheduling System (HSS)	Strategic and tactical	To determine the outcome of scheduling harvests (and replanting) subject to user-defined constraints.	Simulation

Table A.3: Product aims of tactical & tactical/operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Planning horizon	Product aim	Technique
TACTICAL					
13	Guignard <i>et al.</i> (1998)		Tactical (3 time periods)	To incorporate transportation planning (road building) with timber harvest planning.	LP with constraint strengthening through lifting; also using Branch-and-Bound with double contracting.
14	Epstein <i>et al.</i> (1999a); Andalaft <i>et al.</i> (2003); Weintraub <i>et al.</i> (2000)	OPTIMED	Medium-term (2-5 years)	To match supply of standing timber with demands for logs having particular grades, lengths and diameters. Supports decisions on aggregate timber volumes (classified as "export", "sawlogs" and "pulplogs") to harvest to satisfy demand; what roads are to be built (dirt/gravel)/upgraded (from dirt to gravel) to access harvested areas; what volume of logs to be transported to the destinations; what volume of logs to store from summer to winter; choice of harvesting machinery and trucks. Is multi-period and season-based.	0-1 Mixed Integer LP (solved as LP, then heuristic rules round the 0-1 variables up or down).
TACTICAL/OPERATIONAL					
15	Hultqvist and Olsson (2005); Hultqvist and Olsson (2006)		Tactical (six months)	To integrate various aspects of the raw material supply chain from harvesting to processing, with emphasis on procuring raw material from the forest which is owned by the mill.	Convex mixed integer quadratic model with stochastic and deterministic programming.

Table A.4: Product aims of operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Planning horizon	Product aim	Technique
OPERATIONAL					
16	McGuigan (1984)	LOGRAM-1	Operational (1 financial year)	To enable user to send wood resource to best end use from the "best" location using the "best" transport method. Allocates logs (and chips from a processor) to an appropriate processor. One export location, two other processing locations. Transport of logs (and chips) available via road (on- and off-highway) and rail. Inputs to processes are limited to specific log types.	LP
17	Burger and Jamnick (1995)		Operational	To help a forestry firm that is organised into profit centres to make decisions on wood procurement and distribution.	LP
18	Epstein <i>et al.</i> (1999a); Epstein <i>et al.</i> (1999b); Weintraub <i>et al.</i> (2000)	OPTICORT	Short-term	To support decisions on which stands to harvest (given harvest readiness and accessibility by road), what type of machinery to use, what volumes of logs to cut weekly and what bucking patterns to use so that demand for logs of specific lengths and diameters is satisfied.	LP with branch and bound and column generation.
19	Megown <i>et al.</i> (2000)	Fibre Prediction System	Operational	To minimise the variability of the properties of wood entering a pulping process.	Allocation of wood to processes according to wood properties.
20	Karlsson <i>et al.</i> (2003)		Short-term (4-6 weeks)	To decide, at a district level, which stands to harvest, with which crew, and when to harvest; what type of machinery to use, what volumes of logs to obtain so that demand for logs of specific lengths and diameters is satisfied. Takes into account age-related storage costs.	MIP; likely schedules (having short travel distances for crew and equipment) are generated. Heuristic approach used to find feasible solutions fast.
21	Karlsson <i>et al.</i> (2004)		Annual	To decide, at a district level, what areas to harvest in the period of a year (choose from a list of available harvestable areas generated for next 1.5–2 years) so that each wood processor receives correct assortments; which harvesting team to use for each area; transportation and storage decisions are included.	MIP with linear relaxation.
22	Mitchell (2004, pp.89–133,191–212)		Operational (4-8 weeks)	To decide, week by week, how forest will be harvested. Gives location of harvest crews for each week of planning period. Gives log types to be made by the harvesting crews. Calculates any production shortfall in order to meet customers (mills)' demands.	MIP using relaxed integer solutions, yield prediction generation, column generation, constraint branching and integer allocation.
23	Meynink and Borough (2005)	ARTLIS	Short-term	To keep track of wood stocks in the forest and at the mill so as to minimise log age.	
24	Gordon <i>et al.</i> (2006)	Atlas Market Supply (prototype)	Operational (8 weeks)	To ensure that logs of the appropriate quality are supplied to meet the weekly demand, in an optimal way.	MIP/LP

Table A.5: Background details of strategic forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Where used	Frequency of use	Plantation/natural forest?	No. of species
STRATEGIC					
1 Navon (1971); Chappelle <i>et al.</i> (1976); Tedder <i>et al.</i> (1978)	Timber RAM			Natural	
2 Walker (1974); Tedder <i>et al.</i> (1978)	ECHO - economic harvest optimization model			Natural	
3 Barber (1983)	HARVEST			Natural	To be used for single species, even-aged forests.
4 Johnson <i>et al.</i> (1980); Johnson <i>et al.</i> (1986); Garcia (1988); Kent <i>et al.</i> (1991); Weintraub <i>et al.</i> (2000)	FORPLAN	USA (154 national forests)	At least every 10 years according to NFMA law (Kent <i>et al.</i> , 1991)	Natural	
5 Epstein <i>et al.</i> (1999a); Morales and Weintraub (1991)	MEDFOR	Used by Florestal Bio-Bio, Cholguan (Chile) and partially by Masisa. A similar model is being implemented at Aracruz (Brazil). Norway	Run several times a year.	Plantation	1 (<i>Pinus radiata</i>)
6 Eid and Hobbelstad (2000)	AVVIRK-2000				

Table A.6: Background details of strategic/tactical forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Where used	Frequency of use	Plantation/natural forest?	No. of species
STRATEGIC/TACTICAL					
7 Barros and Weintraub (1982)			Model to be run yearly.	Mainly natural	Only one species is grown.
8 Garcia (1984); Garcia (1990); Manley and Threadgill (1991); Papps and Manley (1992); Manley <i>et al.</i> (1996)	FOLPI (Forestry-Oriented Linear Programming Inter-preter)	Large forestry firms in New Zealand		Plantation	
9 McGuigan and Scott (1995)	RegRAM-1 (Regional Resource Allocation Model)	Tasman Forestry Ltd, New Zealand		Plantation	
10 McNaughton <i>et al.</i> (2000); McNaughton <i>et al.</i> (1998)		New Zealand		Plantation	
11 Cea and Jofré (2000)		Chile			1 genus is mentioned (pine)
12 Syndicate Database Solutions (2006)	Microforest Harvest Scheduling System (HSS)	South Africa		Plantation	

Table A.7: Background details of tactical & tactical/operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Where used	Frequency of use	Plantation/natural forest?	No. of species
TACTICAL						
13	Guignard <i>et al.</i> (1998)		USDA Forest Service		Natural	
14	Epstein <i>et al.</i> (1999a); Andalaft <i>et al.</i> (2003); Weintraub <i>et al.</i> (2000)	OPTIMED	Used by Florestal Mil-lalemu (Chile) since 1994. A similar planning tool is being developed for Aracruz (Brazil).	Run every few months	Plantation	1 genus (pine)
TACTICAL/OPERATIONAL						
15	Hultqvist and Olsson (2005); Hultqvist and Olsson (2006)		Sweden			

Table A.8: Background details of operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Where used	Frequency of use	Plantation/natural forest?	No. of species
OPERATIONAL						
16	McGuigan (1984)	LOGRAM-1	Tasman Pulp and Paper, New Zealand Canada		Plantation	
17	Burger and Jammnick (1995)					
18	Epstein <i>et al.</i> (1999a); Epstein <i>et al.</i> (1999b); Weintraub <i>et al.</i> (2000)	OPTICORT	In use at Bosques Arauco, Florestal Celco, Florestal Mininco & Florestal Bio-Bio since 1991. A similar planning tool is being developed for Aracruz (Brazil).	Run every two weeks with a rolling horizon of three months.	Plantation	1 (Pine)
19	Megown <i>et al.</i> (2000)	Fibre Prediction System	South Africa		Plantation	
20	Karlsson <i>et al.</i> (2003)		Sweden	Every 3-4 weeks		Many (3 genera are mentioned)
21	Karlsson <i>et al.</i> (2004)		Sweden			1
22	Mitchell (2004, pp.89–133,191–212)		New Zealand		Plantation	
23	Meynink and Borough (2005)	ARTLIS	Norske Skog (Australia)	Daily		1 (<i>P. radiata</i>)
24	Gordon <i>et al.</i> (2006)	Atlas Market Supply (prototype)	New Zealand		Plantation	1 (<i>P. radiata</i>)

Table A.9: Included features of strategic forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Includes harvesting decisions?	Includes bucking decisions?	Includes timber allocation?	Includes transport to mill?	Includes wood properties?	Comments
STRATEGIC							
1 Navon (1971); Chappelle <i>et al.</i> (1976); Tedder <i>et al.</i> (1978)	Timber RAM						
2 Walker (1974); Tedder <i>et al.</i> (1978)	ECHO - economic harvest optimization model						
3 Barber (1983)	HARVEST	Yes		Yes (Field, 1984)			Difficult to address spatial problems due to computational difficulties (no easy way to represent them in FORPLAN). FORPLAN won't run unless there is timber to harvest (Garcia, 1988). Using FORPLAN requires forest area to be grouped into areas that respond similarly to a set of management practices. The solver allocates prescriptions to area. This is problematic when the solution is implemented, because it may not work for specific stands (Kent <i>et al.</i> , 1991).
4 Johnson <i>et al.</i> (1980); Johnson <i>et al.</i> (1986); Garcia (1988); Kent <i>et al.</i> (1991); Weintraub <i>et al.</i> (2000)	FORPLAN						
5 Epstein <i>et al.</i> (1999a); Morales and Weintraub (1991)	MEDFOR	Yes (on macro stands)	No	Yes	No	No	
6 Eid and Hobbelstad (2000)	AVVIRK-2000						

Table A.10: Included features of strategic/tactical forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Includes harvesting decisions?	Includes bucking decisions?	Includes timber allocation?	Includes transport to mill?	Includes wood properties?	Comments
STRATEGIC/TACTICAL								
7	Barros and Weintraub (1982)				Yes	Yes		
8	Garcia (1984); Garcia (1990); Manley and Threadgill (1991); Papps and Manley (1992); Manley et al. (1996)	FOLPI (Forestry-Oriented Linear Programming Interpreter)	Yes - of a stand or aggregate stand	No	Yes (Manley and Threadgill (1991) version)	No	No	Assumption: trees reaching maximum clearfelling age are automatically felled.
9	McGiugan and Scott (1995)	RegRAM-1 (Regional Resource Allocation Model)	Yes		Yes	Yes		
10	McNaughton et al. (2000); McNaughton et al. (1998)		Yes					Includes road-making.
11	Cea and Jofré (2000)	(Strategic part)	Yes	No		Yes	No	
		(Tactical part)	Yes	No	Yes	Yes	No	Solved once a feasible road structure has been found.
12	Syndicate Database Solutions (2006)	Microforest Harvest Scheduling System (HSS)	Yes	Yes		No	No	

Table A.11: Included features of tactical & tactical/operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Includes harvesting decisions?	Includes bucking decisions?	Includes timber allocation?	Includes transport to mill?	Includes wood properties?	Comments
TACTICAL								
13	Guignard <i>et al.</i> (1998)		Yes					Includes road-making. Paper aims at exploring the techniques of solving the problem.
14	Epstein <i>et al.</i> (1999a); Andalaft <i>et al.</i> (2003); Weintraub <i>et al.</i> (2000)	OPTIMED	Yes – how many Ha of a “homogeneous” stand to harvest in a time period.		Yes	Yes, by road only.		
TACTICAL/OPERATIONAL								
15	Hultqvist and Olsson (2005); Hultqvist and Olsson (2006)		Yes	No	Yes	Yes	No	Stochastic variables included because of uncertainty of ability of roads and harvest areas to bear loads all year round. Problem solved is only for one of five districts of Sweden; would have to find other solution approaches if using a desktop computer for whole problem.

Table A.12: Included features of operational forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Includes harvesting decisions?	Includes bucking decisions?	Includes timber allocation?	Includes transport to mill?	Includes wood properties?	Comments
OPERATIONAL							
16 McGuigan (1984)	LOGRAM-1	No		Yes	Yes		Author recognises the variability in basic wood properties in the wood resource. 35 types of logs come from 40 different forests.
17 Burger and Jamnick (1995)		Yes	Yes	Yes	Yes		Product considers: land ownership (incl. min and max harvesting levels); sawmill chip volumes; harvest block volumes; harvest methods; transport distances; transport costs; min and max mill requirements.
18 Epstein <i>et al.</i> (1999a); Epstein <i>et al.</i> (1999b); Weintraub <i>et al.</i> (2000)	OPTICORT	Yes - how many Ha of a "homogeneous" stand to harvest in a time period.	Yes	Yes	Yes		Makes "macro-stands" of stands to be harvested to aid transporting decisions.
19 Megown <i>et al.</i> (2000)	Fibre Prediction System	No				Yes	Allocates wood to processes according to wood properties after a harvest scheduling system has made the harvesting decision.
20 Karlsson <i>et al.</i> (2003)		Yes	No	Yes	Yes	No	Age-related storage costs are included, as well as road-ownership hierarchy.
21 Karlsson <i>et al.</i> (2004)		Yes	No	Yes	Yes	No	Assumption: harvesting an area will be complete in no longer than two months. Age related storage costs are included, as well as road-ownership hierarchy.
22 Mitchell (2004, pp.89-133,191-212)		Yes	Yes	Yes	Yes	No	
23 Meynink and Borough (2005)	ARTLIS	No	No			No	Does not optimise. Uses wood freshness as a surrogate for wood quality (e.g. brightness).
24 Gordon <i>et al.</i> (2006)	Atlas Market Supply (prototype)	No	Yes	Yes	Yes	Yes (for saw timber); assesses wood properties (density) and stem characteristics 3-5 years before harvest date.	

Appendix B

Mathematical formulation of forest harvest scheduling systems or models described in the literature

This appendix gives details of the mathematical formulation of the systems and models outlined in Appendix A. Tables B.1 – B.7 give the model or system's the objective function, decision variables and constraints (if the system/model uses an optimization technique). Tables B.8 – B.12 give the technique used, the problem size, details of the solution approach and solution time, and notes about programming language and solver used are given where available. The problem size is important, along with the technique and details of the computer used, as this gives insight into the ease and speed of solution.

Table B.1: Mathematical formulation of strategic forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Objective function	Decision variables	Constraints
STRATEGIC				
1 Navon (1971); Chapelle <i>et al.</i> (1976); Tedder <i>et al.</i> (1978)	Timber RAM	Either silvicultural (max. harvested volume), or economic (max. present total worth of income from sale of harvested trees or min. management costs over time). Can use this to maximize the discounted cash flow from the forest while maintaining an even flow of volume having a prespecified rotation age.		Recreation and wildlife concerns treated as constraints.
2 Walker (1974); Tedder <i>et al.</i> (1978)	ECHO - economic harvest optimization model	Discounted cash flow over the planning horizon.		
3 Barber (1983)	HARVEST	Does not optimise.		
4 Johnson <i>et al.</i> (1980); Johnson <i>et al.</i> (1986); Garcia (1988); Kent <i>et al.</i> (1991); Weintraub <i>et al.</i> (2000)	FORPLAN	Maximises the net present value of the forest.		Wildlife habitat area constraints, non-declining timber harvest constraints (Field, 1984; Kent <i>et al.</i> , 1991).
5 Epstein <i>et al.</i> (1999a); Morales and Weintraub (1991)	MEDFOR	Max net discounted income.	Area of macro stand to which silvicultural prescription will be applied; area of macro stand to be established in the future, to which a particular prescription will be applied; volume of log product to be harvested or thinned in time t and sent to a mill; area of land having a particular site quality to be acquired in time t ; volume of standing timber remaining at $t =$ planning horizon.	No more than the maximum area of the macro stand may be considered (for current and future macro stands); an upper limit of land per site quality is available for acquisition in time t ; standing timber value and age constraints at the planning horizon; all timber harvested is sent to destination mills; product demands are met.
6 Eid and Hobbelstad (2000)	AVVIRK-2000	Does not optimise.		

Table B.2: Mathematical formulation of strategic/tactical forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Objective function	Decision variables	Constraints
STRATEGIC/TACTICAL				
7 Barros and Weintraub (1982)		Several, e.g. max net present revenue over time; max total timber obtained in next few time periods.	Decision variables correspond to no. of acres of a stand receiving fixed silvicultural treatments in given periods.	Mills have log dimension and volume constraints (min and max). Certain stands have access constraints (seasonal). There are upper limits on the amount of timber that can be purchased from outside the system in summer and winter. (It rains in winter, making access to compartments difficult). Continuity: there are conditions on the composition of timber classes at the horizon.
8 Garcia (1984); Garcia (1990); Weintraub et al. (2000); Manley and Threadgill (1991); Papps and Manley (1992); Manley et al. (1996)	FOLPI (Forestry-Oriented Linear Programming Inter-preter)	In most cases: max net present cash flow (Garcia, 1990).	Which stand (individual stand or aggregated stand of croptype) to harvest in each time period; logs to produce to meet demand at mills; to which mills to send logs.	Min and max age for clearfelling. Min and max volume of logs to make.
9 McGuigan and Scott (1995)	RegRAM-1 (Regional Resource Allocation Model)	Max NPV of future cash flows before tax (could be forestry only or whole value chain, depending on the problem the user specifies).	Best thinning and clearfelling regime, logs that are produced with these regimes, how much external resource (logs, chips) to buy from external sources, how to move these resources to the processor, which process should use them, what further product should be produced, which markets should they be sold in.	Min and max clearfelling age for each croptype. Max delay in age for scheduled thinning. Total harvest volumes per forest. Min MAI (to prevent forest from being over harvested). Prevent year-to-year volume harvested from declining or stumpage income from declining. Limitations on when a forest may be harvested. Limit amount of external resources bought in. Min and max sales for any resource (logs, chips). Min and max capacity for any process. Constrain movement of certain resources. For longer-term models: ensure that the end age-class distribution is similar to that of half the planning horizon (hasnt been needed). Limit total cash flow to max level of expenditure, or force the system to produce a non-declining total cash flow.

Table B.3: Mathematical formulation of strategic/tactical forest harvest scheduling systems & models found in literature (cont.)

	Literature reference	Name of product	Objective function	Decision variables	Constraints
STRATEGIC/TACTICAL (cont.)					
10	McNaughton et al. (2000); McNaughton et al. (1998)	(Strategic part) (Tactical part)	Max present net worth of forest (takes account of revenue and costs of future harvests).	Area (hectares) of a particular type of crop (croptype) established in a particular year, and harvested in year t . Road harvest plan (all the stands on a particular road that will be harvested during the tactical plan's duration); Overlap variables: area of a particular croptype established in a particular year and harvested in year $t+1$ which is located in stands where harvesting will start in year t .	Minimum area of a particular croptype to be felled in a year; non-declining yield; no more of a particular croptype is felled than is available. Only road harvest plan can be selected for a particular road; harvesting is allowed using a road if it has been constructed before harvesting begins; there must be a continuous access route from the harvested stand to the mill. Linking constraints (to link the strategic and tactical plan's variables). Adjacency constraints: given the harvesting of a certain stand in year t , ensuring that other stands needing similar harvesting methods on the same road will also be harvested within a limited period.
11	Cea and Jofré (2000)	(Strategic part)	Max (NPV)(income less harvesting costs, silvicultural regime costs, admin costs, product transport costs)	Existing and future macro-stand areas; volume harvested per zone, log type, destination and time	Silvicultural regimes for existing and future macro-stands; harvested volume is made up of current and future macro-stands' trees; forest supply is greater than or equal to demand; the building or transformation of processes.
		(Tactical part)	Max income from product sales – transport costs – cost of harvesting stands Does not optimize.	Volume of a particular product supplied by a node to the network per time period; area harvested in a certain stand in time t .	Conservation of wood flow; demand for wood is met at mills.
12	Syndicate Database Solutions (2006)	Microforest Scheduling System (HSS)			User-defined (e.g. fell a compartment or plantation on, before or after a certain age).

Table B.4: Mathematical formulation of tactical forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Objective function	Decision variables	Constraints
TACTICAL					
13	Guignard <i>et al.</i> (1998)			Which stands to cut at time t with a particular cutting pattern; whether road links are built at time t to standard k ; amount of traffic from A to B on a road link with standard k at time t ; the amount of traffic from B to final destination on an existing road at time t .	A stand may only be cut once, and only with one cutting pattern; a road can only be built once to a specific road quality; conservation of flow of volume of logs; road capacity constraint; if a link exists during time period t it will also exist in $t + 1$. May also be budget constraints, sustained yield constraints, adjacency constraints etc. (not described here). A stand should not be cut at time t if an access road does not yet exist. Isolated road links should also not be built.
14	Epstein <i>et al.</i> (1999a); Andalaft <i>et al.</i> (2003); Weintraub <i>et al.</i> (2000)	OPTIMED	Max NPV (income-costs).	Area of aggregated stand to be harvested; the amount of timber to harvest to satisfy demand; the roads to be built or upgraded (these are the 0-1 variables); the amount of timber to transport to the destinations; the amount of timber stocked from summer to winter, the choice of harvesting machinery and trucks.	"Higher value" logs (e.g. export quality) can be sent to "lower value" processes (e.g. pulpmills), but not the other way around. Dust roads can only be travelled on in summer; roads can only be built or upgraded in summer. Logs can be stored in summer for use in the next winter, but they have to be used up before the next summer. Each destination (pulpmill, sawmill, export, storage area) has a min and max volume requirement. If a stand is to be harvested, either 10 Ha or more can be harvested in a time period, or 0 Ha.

Table B.5: Mathematical formulation of tactical/operational forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Objective function	Decision variables	Constraints
15 TACTICAL/OPERATIONAL Hultqvist and Olsson (2005); Hultqvist and Olsson (2006); Hultqvist and Olsson (2004)		Minimise the cost of supplying the volumes of timber demanded. Costs include: harvesting costs, road maintenance and transport costs (road and rail), loading, unloading and storage costs, log purchasing costs, cost of having more than one log assortment per mill, costs of buying and transporting chips from sawmills to pulp mills (Hultqvist and Olsson, 2004).	Continuous variables: volume of log type to be used in a particular mill in a certain time t ; volume of log type to be transported to a mill not owned by company, in time t ; volume of log type bought on domestic market by a mill for a particular scenario in time t ; volume of log type bought on international market by a mill for a particular scenario in time t ; number of hours used to harvest, using scenario, in time t ; log type stored at particular place for a certain scenario from time u till t , that was initially stored during period k ; volume of log type stored at the end of a time period associated with a certain scenario; volume of log type transported on a particular road for a certain scenario in time t ; volume of chips transported from a sawmill to a pulp mill according to a certain scenario, in time t ; volume of log type transported by rail for a certain scenario during time t . Binary variables include: whether or not area is harvested in time t with a particular scenario; whether or not the log type is used at the sawmill; whether or not anything is harvested in time t ; whether or not anything is stored at a particular storage depot during time t , associated with a particular scenario; whether or not anything is transported on the railroad (Hultqvist and Olsson, 2004).	Harvesting: Can't harvest more timber than is available; harvesting of an area must be complete in up to two periods; the number of areas harvested in two consecutive periods is constrained by the number of harvesting teams available; harvesting must be completed and transported before thawing if there is demand for timber during that period; harvesting hours are restricted; only wood harvested can be transported. Transportation and roads: conservation of flow; restriction of volume transported; restriction of number of hours available to transport wood. Storage: conservation of flow; storage capacity; minimum safety levels. Mills: Min and max volume needed to be delivered (including wood bought on open market). Wood chips: chips from sawmills make up part of the demand at a pulp mill; the mill's demand for chips must be met. Markets: limitation of amount that can be bought on domestic/international markets. Rail: amount of wood transported by rail may be restricted.

Table B.6: Mathematical formulation of operational forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Objective function	Decision variables	Constraints
OPERATIONAL				
16 McGuigan (1984)	LOGRAM-1	Max ((sales of products)-(cost of making sales)).	Volume of each log type available from each forest to be allocated to each manufacturing plant; volume of each residue type to be allocated to a residue-using plant.	Min and max requirement of logs at the mill. Min and max processing capacity at the plant. Log type constraints (e.g. limiting the species to enter a process). Constraints on max amount of residues a pulp process will accept. Market demand constraints.
17 Burger and Jamnick (1995)		Min pulpmill's cost, or min forest's cost, or max forest's profit.	Proportion of a block (group of stands with same species) to harvest with a particular harvesting option; volume of product harvested with a particular option to be sent to a particular mill; volume of chips to be sent from a particular sawmill to the pulpmill.	Proportion of forest block to be harvested; every product that it cut down must be sent to a mill; total harvested volume using a particular harvesting method is less than that method's production capacity; minimum volume of chips to be purchased from each sawmill, and total volume of chips available from each sawmill; min and max product requirements for each mill; Max harvest level for each land ownership; (optional) pulp-mill's log cost is less than an upper limit; (optional) minimum total profit for forests.
18 Epstein <i>et al.</i> (1999a); Epstein <i>et al.</i> (1999b); Weintraub <i>et al.</i> (2000)	OPTICORT	Max (income after delivering logs to destination) – (cost of harvesting logs).	Which stands to harvest among those with mature trees on them; what machinery to use for each operation; what volume to cut, with which bucking pattern; what logs to deliver to different destinations to satisfy demand.	Harvested volume of trees is less than or equal to volume available; harvesting uses less than or equal to total harvest machine capacity; can transport only what is harvested; min and max demand for sale (of many types of logs) and for individual logs; average diameter of logs for each sale of logs needs to be met.
19 Megown <i>et al.</i> (2000)	Fibre Prediction System	Does not optimise.		
20 Karlsson <i>et al.</i> (2003)		Min cost (of schedule, moving equipment, road opening & maintenance, transportation costs, storage costs for logs).	Harvesting sequence to choose; road to choose; log flows (from area to industry/terminal, or from terminal to industry); amount of a particular log assortment having a particular age is to be used in industry during period t ; storage of logs of a particular age in sortment, having a particular age in area/terminal/industry during period t .	An area can only be harvested once; each team gets exactly one schedule; industry demand for logs is satisfied; at least one road to an area must be open for logs to be transported from it.

Table B.7: Mathematical formulation of operational forest harvest scheduling systems & models found in literature (cont.)

	Literature reference	Name of product	Objective function	Decision variables	Constraints
OPERATIONAL (cont.)					
21	Karlsson <i>et al.</i> (2004)		Min total cost = (harvesting cost + road-opening cost + cost of purchased logs + cost of transporting logs to mills + storage costs in forest, at mills and at terminals + cost for not satisfying demand + cost of not being able to access areas or roads + penalty for harvesting an area in two adjacent time periods).	What area to harvest in a period by a particular crew and what proportion to harvest in that period with that crew; what proportion of area to harvest in next period, for which harvesting commenced in previous period; what roads to maintain in a period; flow of assortment from area to mill in a particular period; storage of wood in forest (area) and at mill in particular period, unsatisfied demand at mill and purchase volume of assortment for a mill in a time period, number of extra working days assigned to a crew in the time period.	An area can only be harvested once; the whole harvest area must be harvested; logs are stored either in the forest, at the terminal or at the mill; volume of purchased logs is restricted; more logs cannot be stored at the terminal or mill than its storage capacity allows; at least one road to an area must be open for logs to be transported from it; the crew has a limited number of days to work each month.
22	Mitchell (2004, pp.89–133,191–212)		Min (crew scheduling cost – log transport costs – log downgrade costs + log demand + cost of not meeting log volume demands + cost of not meeting log diameter constraints + cost of not adhering to specified product fractions + cost of remaining trees per unit + inventory costs).	Allocation of crew to a schedule, volume of a particular log type in a certain harvest unit which is to be sent to a customer, logs stocks of a particular log type in a harvest unit at the end of a time period, volume of logs of a particular log type which is downgraded to another log type for a customer in a particular time period.	Harvest crews allocated to one schedule; a limited number of crews may work on a harvest unit; crews may cut a maximum of the total area available, customer's log volume, log diameter and product fraction demands are met, inventory of a particular log type in a harvest unit is limited, as is downgrade of logs.
23	Meynink and Borough (2005)	ARTLIS			
24	Gordon <i>et al.</i> (2006)	Atlas Market Supply (prototype)	Max (revenue from harvested stands and external supply) – (transport costs for delivery + harvesting costs + cost of purchasing logs externally).	Log volume, log price, transportation cost, harvesting cost, crew location and allocation (integer), external log supply cost, residual value of stocked logs, residual forest value, volume of stocked logs, residual area.	No more than one harvest crew is allocated to a stand per week; the harvesting crew must be allocated to one stand (unless they are 'idle'); supply constraints, amount of in-forest stocks; availability of harvest stands, harvest crews.

Table B.8: Problem size & solution details of strategic forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Technique	Time period	Problem size	Solution time & machine specs	Programming language/Solver
STRATEGIC						
1 Navon (1971); Chapelle et al. (1976); Tedder et al. (1978)	Timber RAM		1 period = 10 years. Can provide info at a 5 year level too for first 10 years).			
2 Walker (1974); Tedder et al. (1978)	ECHO - economic harvest optimization model		1 period = 1 year.			
3 Barber (Barber (1983))	HARVEST		Planning horizon: maximum 30 periods (periods could be 5, 10 or 20 years).			
4 Johnson et al. 1980; Johnson et al. (1986); Garcia (1988); Kent et al. (1991); Weintraub et al. (2000)	FORPLAN	LP; cardinally-weighted goal programming available with Version 1 (Field, 1984). MIP solved as an LP (Kent et al., 1991).	Inputs and outputs are given 10 years at a time; is usually run for 15 time periods.	1000-5000 rows, 15000-12000 columns.	4-100 mins, typically 30-65 mins (on UNIVAC 1100/92 mainframe).	Programming language: FORTRAN (Kent et al., 1991)
5 Epstein et al. 1999a; Morales and Weintraub (1991)	MEDFOR	LP	1 period = 3 years except for first three years which are 1 year each.	1200 variables, 600 constraints	About 2 hours on an IBM 386 PC.	Solver: HyperLindo; system written in dBase IV.
6 Eid and Hobbelstad 2000	AVVIRK-2000	Simulation	Planning horizon broken into 10 year intervals; forest is analysed at mid-point of each 10-year period.			

Table B.9: Problem size & solution details of strategic/tactical forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Technique	Time period	Problem size	Solution time & machine specs	Programming language/Solver
STRATEGIC/TACTICAL						
7 Barros and Weintraub 1982		LP	Planning horizon covers 2 rotations of 25 years. Periods in planning horizon are 1 yearly for first few years, 2 yearly for intermediate time periods and 5–10 yearly for last years.	Model considers 100–300 stands, 3–5 management alternatives per stand, 10–15 periods, 2 access points (depots). LP has 400–800 constraints, 1000–2500 variables.		
8 Garcia (1984); Garcia (1990); Weintraub et al. (2000); Manley and Threadgill (1991); Papps and Manley (1992); Manley et al. (1996)	FOLPI (Forestry-Oriented Linear Programming Inter-preter)	LP	Time intervals equal to number of years in each age class (e.g. 1 period = 2 years) (Manley and Threadgill, 1991).	1542 rows, 11460 columns, 40 periods, 60 age classes and 11 crop types (Garcia, 1984).	50 mins (Garcia, 1984).	LP2900 (ICL 1980) or MINOS, SCICONIC, XA, C-WHIZ (Manley et al., 1996)
9 McGuigan and Scott (1995)	RegRAM-1 (Regional Resource Allocation Model)	LP, simulation	Data entered annually; data preprocessor can aggregate data so that it represents more than a year – according to user's requirements.			Uses spreadsheets for initial data inputs.
10 McNaughton et al. (2000); McNaughton et al. (1998)	(Strategic part)	LP with column generation and constraint branching.	1 time period equals one year.	145 mature forest stands to be harvested over 6 years having 30 crop types, and 38 road segments. About 1820 continuous variables and about 1120 constraints.	6–8 mins.	
	(Tactical part)			About 1560 binary integer variables and about 1100 constraints.		
11 Cea and Jofré (2000)	(Strategic part)	MIP; strategic solves it with LP using branch and bound.	One period = 3 years.			
	(Tactical part)	Iterate through road and tactical problems by using Simulated annealing and LP solver.	One period = 1 year			
12 Syndicate Database Solutions (2006)	Microforest Scheduling System (HSS)	Simulation				

Table B.10: Problem size & solution details of tactical & tactical/operational forest harvest scheduling systems & models found in literature

	Literature reference	Name of product	Technique	Time period	Problem size	Solution time & machine specs	Programming language/Solver
TACTICAL							
13	Guignard <i>et al.</i> (1998)		LP with constraint strengthening through lifting; also using Branch-and-Bound with double contracting.		4 problems presented: 28–350 stands; 24–148 road segments; 30–548 adjacency constraints; 156–1803 0-1 variables.	15–30 mins	
14	Epstein <i>et al.</i> 1999a; Andalaft <i>et al.</i> (2003); Weintraub <i>et al.</i> (2000)	OPTIMED	0-1 Mixed Integer LP (solved as LP, then heuristic rules round the 0-1 variables up or down).	Planning years are divided into summer and winter seasons.	7000 constraints; 1000–1700 0-1 variables; 8000–12000 variables altogether.	30 mins.	
TACTICAL/OPERATIONAL							
15	Hultqvist and Olsson (2005); Hultqvist and Olsson (2006); Hultqvist and Olsson (2004)		Convex mixed integer quadratic model with stochastic and deterministic programming.		Deterministic: 10353 variables, 2569 integer variables, 3 quadratic variables. Stochastic: 39363 variables, 5779 integer variables, 13 quadratic variables, 48883 constraints (Hultqvist and Olsson, 2005).	Deterministic: 0.5–2mins to near-optimum solution; 0.25–5 hours to optimum solution; Stochastic: 1 min to near-optimum solution. PC: 2 GHz Pentium 4 with 512 MB RAM (Hultqvist and Olsson, 2005).	Solver: Lingo 8.0 from LINDO Systems (Hultqvist and Olsson, 2005)

Table B.11: Problem size & solution details of operational forest harvest scheduling systems & models found in literature

Literature reference	Name of product	Technique	Time period	Problem size	Solution time & machine specs	Programming language/Solver
OPERATIONAL						
16 McGuigan (1984)	LOGRAM-1	LP		Test case: 9 sawmills (logs supplied to 5; chips purchased from all 9), 1 pulp mill, 3 land-ownership types; 7 harvesting methods; 41 harvesting blocks = 779 variables and 467 constraints.	7 mins on an IBM compatible 386-25MHz machine.	Uses LINDO LP software; written in Turbo Pascal; data files stored in dBase format.
17 Burger and Jamnick (1995)		LP				
18 Epstein <i>et al.</i> (1999a); Epstein <i>et al.</i> (1999b); Weintraub <i>et al.</i> (2000)	OPTICORT	LP with branch and bound and column generation.		132 stands, 35 types of logs, 10 possible bucking patterns per stand. 3348 constraints, 7584 variables (Epstein <i>et al.</i> , 1999b).	23 mins (to optimality) on a 486 PC.	Uses CPLEX.
19 Megown <i>et al.</i> (2000)	Fibre Prediction System	Allocation of wood to processes according to wood properties.				
20 Karlsson <i>et al.</i> (2003)		MIP; likely schedules (having short travel distances for crew and equipment) are generated. Heuristic approach used to find feasible solutions fast.	Number of days (e.g. 5).	A case study: 6000 sequences, 36750 continuous variables, 5470 constraints.	6.3min on a 1.7 GHz Intel workstation with 1GB RAM; double the number of sequences: 120 mins.	Uses CPLEX 7.0 to solve the model. AMPL used to implement mathematical model. Harvesting sequences generated with a C++ program.
21 Karlsson <i>et al.</i> (2004)		MIP with linear relaxation.	One period = 1 month except for storage where it's 2-3 weeks.	238100 continuous variables, 27740 binary variables, 32400 constraints, 14975 branch-and-bound nodes, 0.4% gap.	15 hours on a 2.66 GHz PC with 2 GB RAM; More than a day for 0.01% tolerance of optimality.	Uses CPLEX 8.1 to solve the model. AMPL used to implement mathematical model.

Table B.12: Problem size & solution details of operational forest harvest scheduling systems & models found in literature (cont.)

	Literature reference	Name of product	Technique	Time period	Problem size	Solution time & machine specs	Programming language/Solver
OPERATIONAL (cont.)							
22	Mitchell (2004, pp.89–133,191–212)		MIP using relaxed integer solutions, yield prediction generation, column generation, constraint branching and integer allocation.		7 crews, 26 clearfell harvest units, 10 thinning harvest units, 81 log types, 10 customers, 1 species.	154.2 secs	ZIP environment (coded in FORTRAN)
23	Meynink and Borough (2005)	ARTLIS					
24	Gordon <i>et al.</i> (2006)	Atlas Market Supply (prototype)	MIP/LP	8 1-week time intervals	About 50000 rows and columns, of which about 20% are integer. This applies to 90000 Ha of plantation that produces 30000 m ³ logs per week, supplies 8–13 mills per week, and is harvested by 11 harvesting crews = 170 stands, 20 bucking patterns per harvest unit, 10 crews, 30 log products and 10 customers.	30–40 mins on a 3.0 GHz Pentium 4 with 2GB memory.	System written in C# and has SQL database to store data and reports. Uses third party solver (Dash Xpress-MP solver) for the optimisation.

Appendix C

Impact of the system on the domains

In sections 4.2 to 4.6 of Chapter 4, the domains as they are at the moment were described. In this appendix, the entities, business processes, rules and decisions which would change because of the implementation of the forest harvest system which includes of wood properties are described. Aspects of the domains which would change after implementing the proposed system are highlighted in the diagrams with pale green. Section C.1 highlights when stands could be sampled for wood property measurement prior to harvesting. Sections C.2 and C.3 show how the transport and mill domains would change as a result of implementing the system, respectively. At the depot or siding, timber would have to be stored (sorted) by mill's process and not just by the mill for which it was destined. At the mill, the logyard would have to have different piles for the different processes.

Implementing the system would also impact the planning processes. Section C.4 shows how the strategic plan's contents would change, while the additional acceptance criteria for the plan are highlighted in section C.5. Finally, the change in the planning process is shown in section C.6 with business process diagrams.

C.1 Forestry domain: stand lifecycle

In the lifecycle of a stand of trees, a few years before the stand is felled, an enumeration takes place. This is to update the plantation database with measurements about the stand so that its volume can be predicted. As referred to in section 4.7.5, if the stand's average wood properties were to be measured and input into the forest harvest scheduling system, it would also be undertaken when the stand is enumerated. (This is not current practice.) This is highlighted in Figure C.4 on page 256. The stand's lifecycle is described in section 4.3.5, Figures 4.12 to 4.15 (pages 80 to 84).

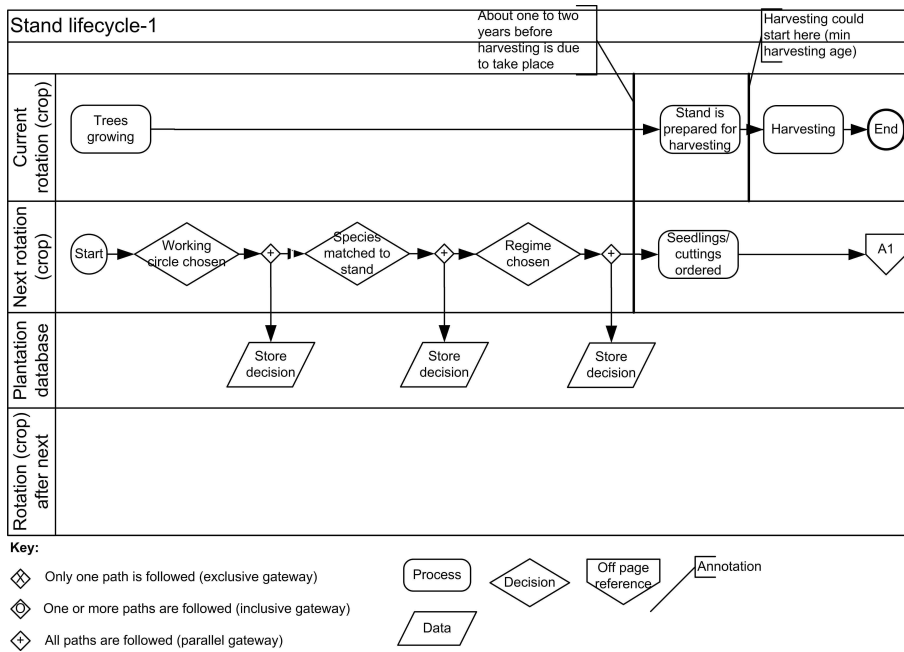


Figure C.1: Updated business process diagram of the stand's lifecycle (part 1 of 4)

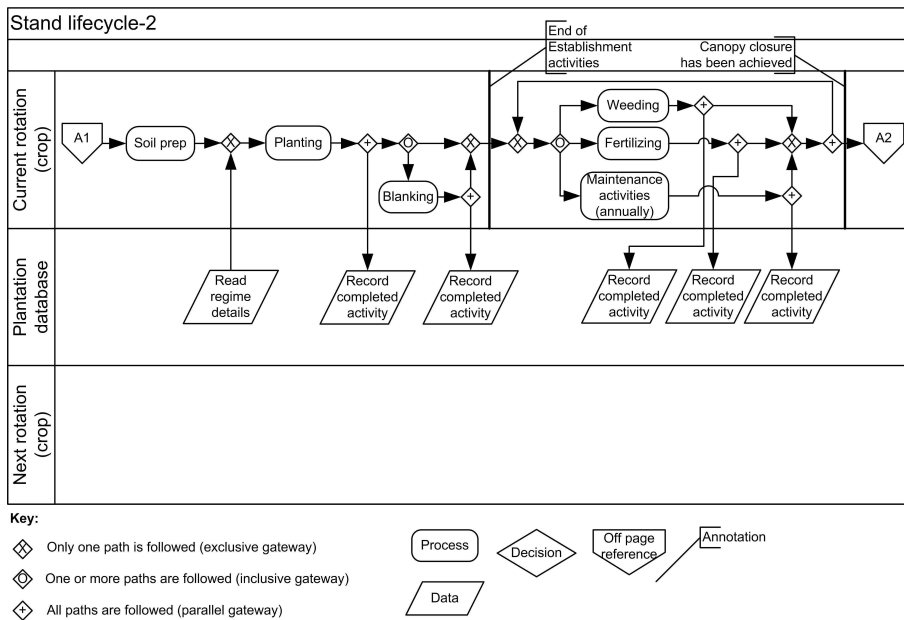


Figure C.2: Updated business process diagram of the stand's lifecycle (part 2 of 4)

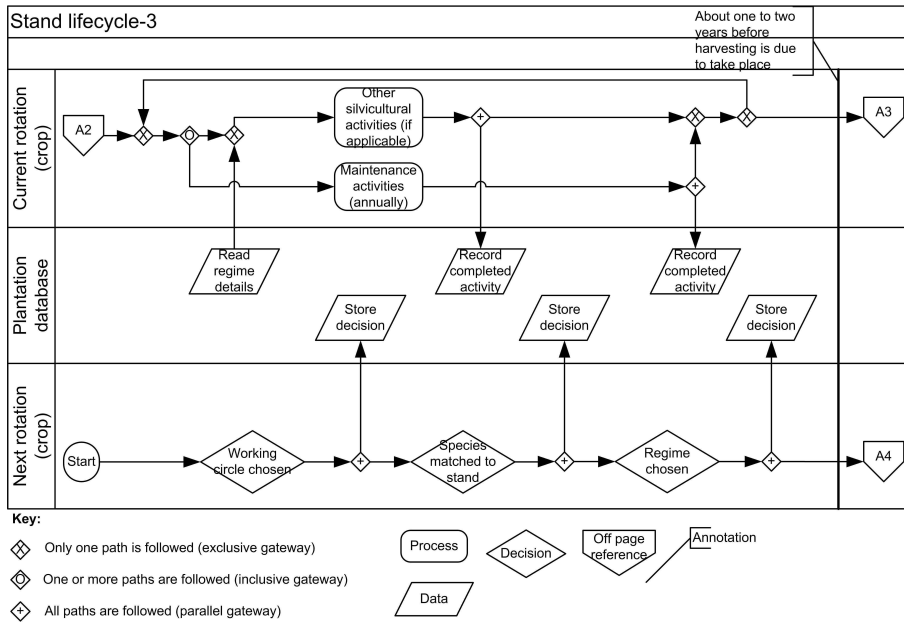


Figure C.3: Updated business process diagram of the stand's lifecycle (part 3 of 4)

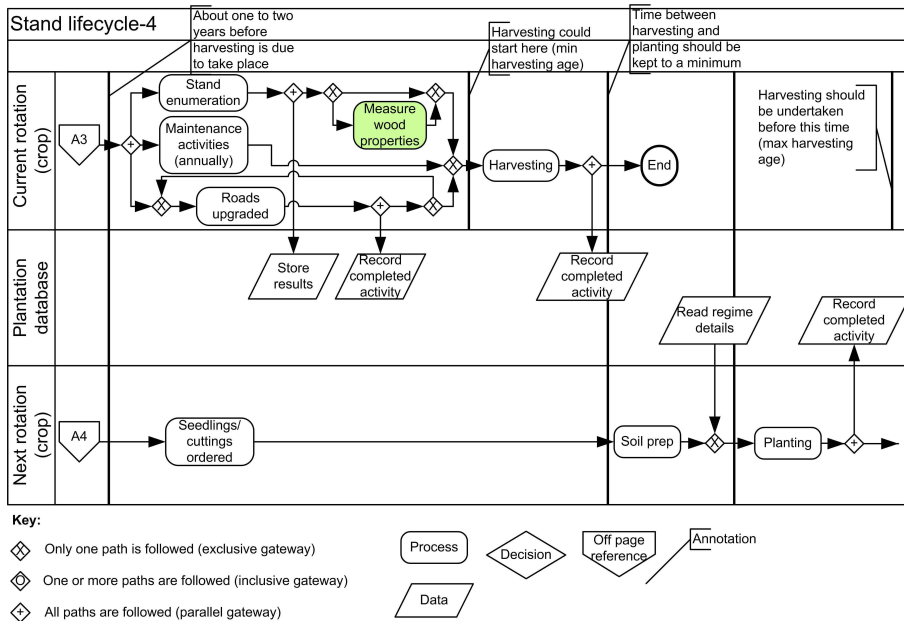


Figure C.4: Updated business process diagram of the stand's lifecycle (part 4 of 4)

C.2 Transport domain: separation of logs at depots and mills

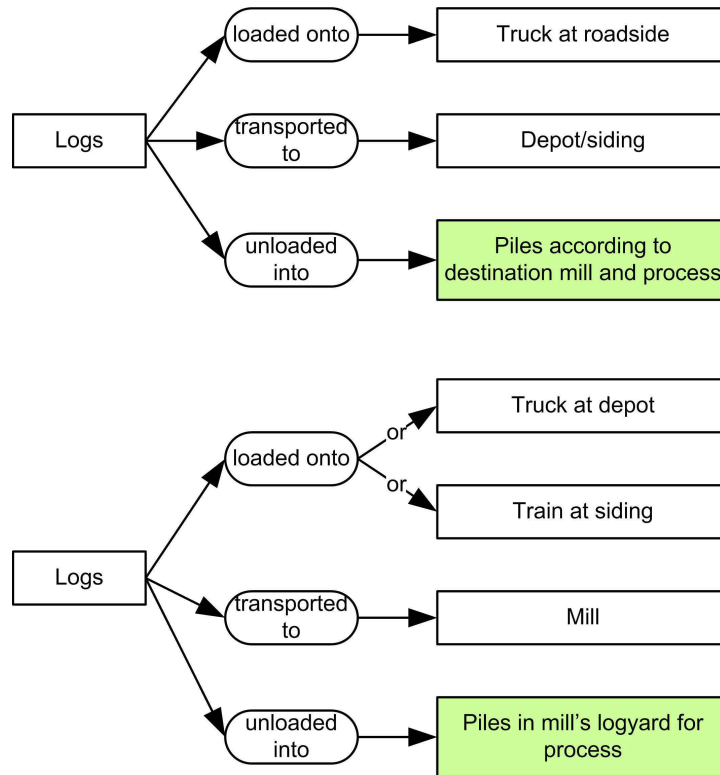


Figure C.5: ERD showing the transport of logs to the mill, including sorting by mill process

After implementing the forest harvest scheduling system, when transporting timber from the stand's roadside, a change would have to be made at the depot or siding and at the mills' logyards. The current practice is to put timber in piles at the depot according to the mill for which the timber is destined (see section 4.4.2, and Figure 4.23 on page 94). After the implementation of the forest harvest scheduling system, timber would have to be piled according to the mill's process – i.e. if a mill had more than one process, there would have to be as many piles at the depot or siding. At the mill, instead of the logs being unloaded into the logyard, they would have to be unloaded into the log pile for the appropriate process. This is shown in Figure C.5.

C.3 Mill domain: criteria for accepting logs

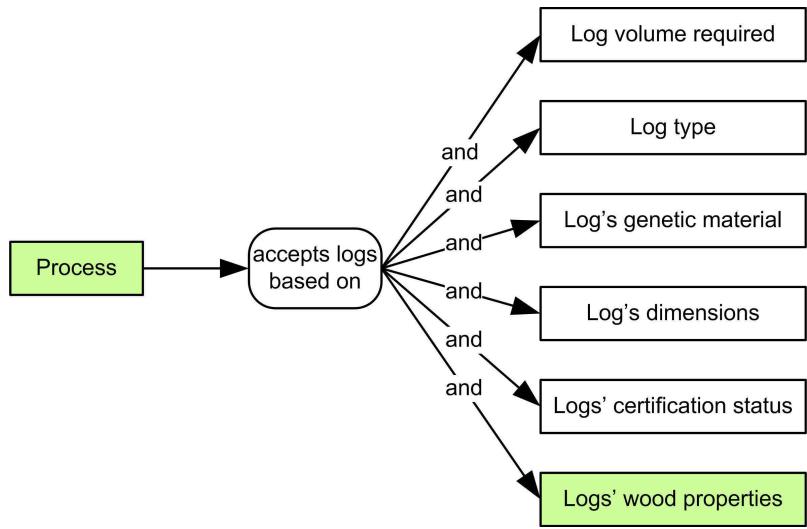


Figure C.6: ERD giving a process' criteria for accepting logs

At present, mills accept logs because they need additional volume to process. There may be acceptance criteria based on the log's type (e.g. generally sawmills do not accept pulplogs). The logs must be of certain specified genera or species (or, not of other specified genera or species). The logs' dimensions may be specified as acceptance criteria. Finally, the mill needs a minimum volume percentage intake of timber which has been grown according to certified management methods. This is outlined in section 4.5.2, and in Figure 4.28 on page 98. After the harvest scheduling system has been implemented, an additional criterion will be added: that the logs destined for the mill have specified wood properties. This is actually specified by process, as shown in Figure C.6.

C.4 Planning domain: contents of the plans

The strategic, tactical and operational plans contain stand harvesting information, stand product information and stand planning information (see 4.35 on page 108). This is described in full in section 4.6. After the harvest scheduling system has been implemented, the stand product information will be augmented by the logs' wood property information (Figure C.7 on page 259). This is the average wood property for the stand, at harvesting age.

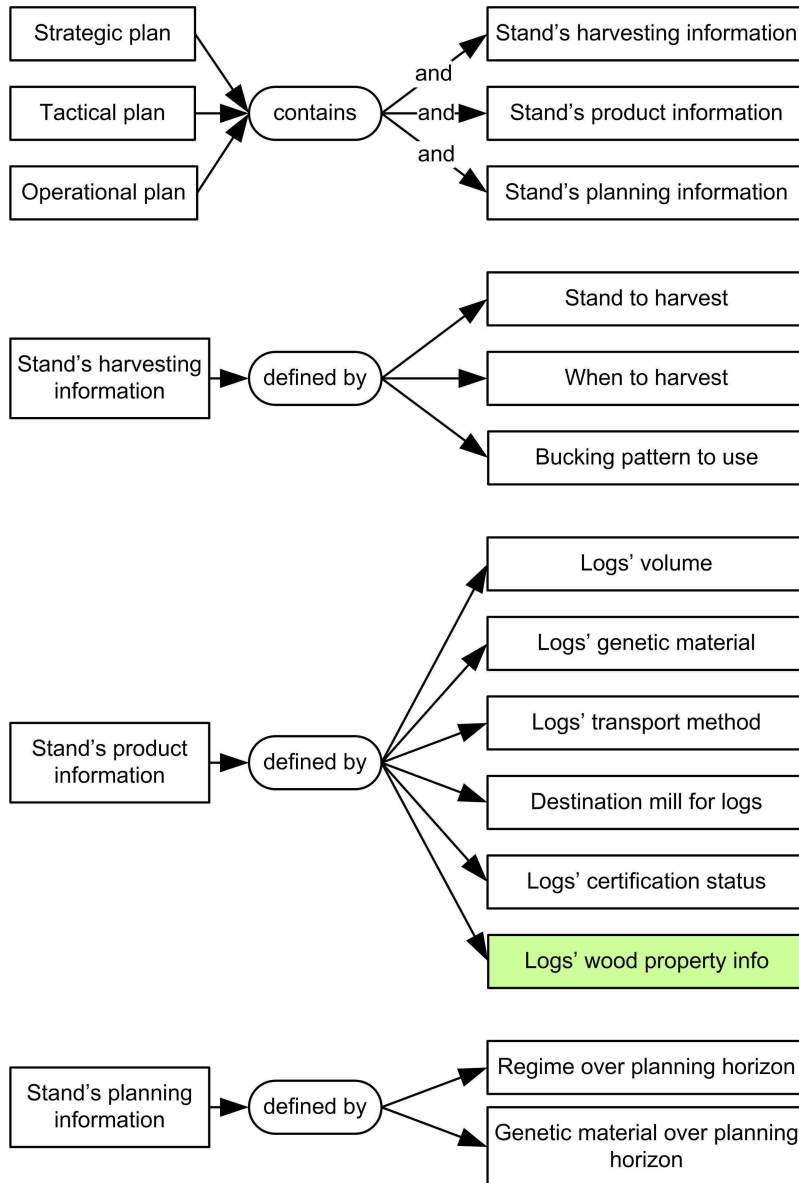


Figure C.7: ERD showing the updated contents of the strategic, tactical and operational plans

C.5 Planning domain: plan acceptance criteria

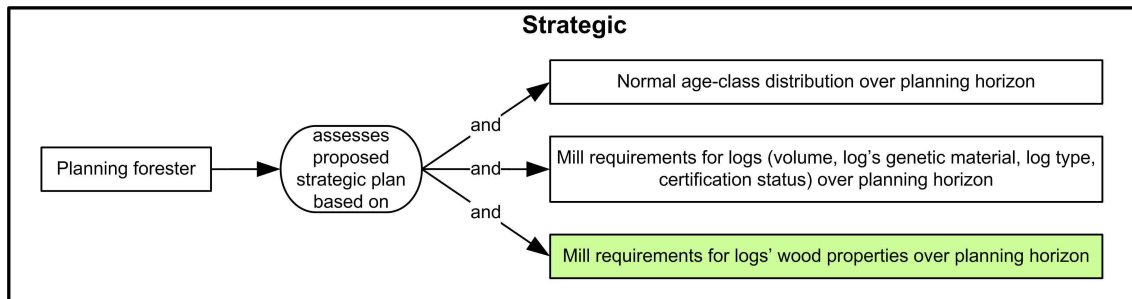


Figure C.8: ERD showing the updated criteria on which the strategic plan is assessed by the planning forester

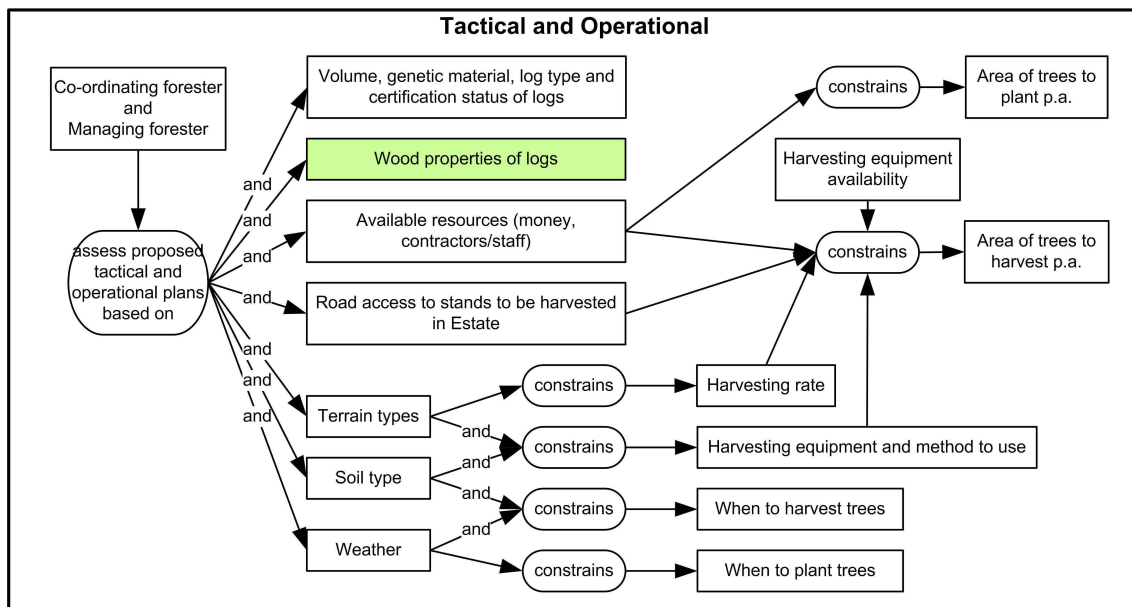


Figure C.9: ERD showing the updated criteria on which the tactical and operational plans are assessed by the Co-ordinating and Managing foresters

The development of the strategic plan is an iterative process. The Planning forester, Co-ordinating forester, Managing forester and Timber logistics manager all assess whether the plan is feasible according to criteria described in section 4.6, and depicted in Figures 4.38, 4.39 and 4.40 on pages 111, 111 and 113 respectively. After the harvest scheduling system has been implemented, the wood properties of the logs made will also be included in the assessment of the plans. This is illustrated in Figures C.8, C.9 and C.10 (pages 260, 260 and 261).

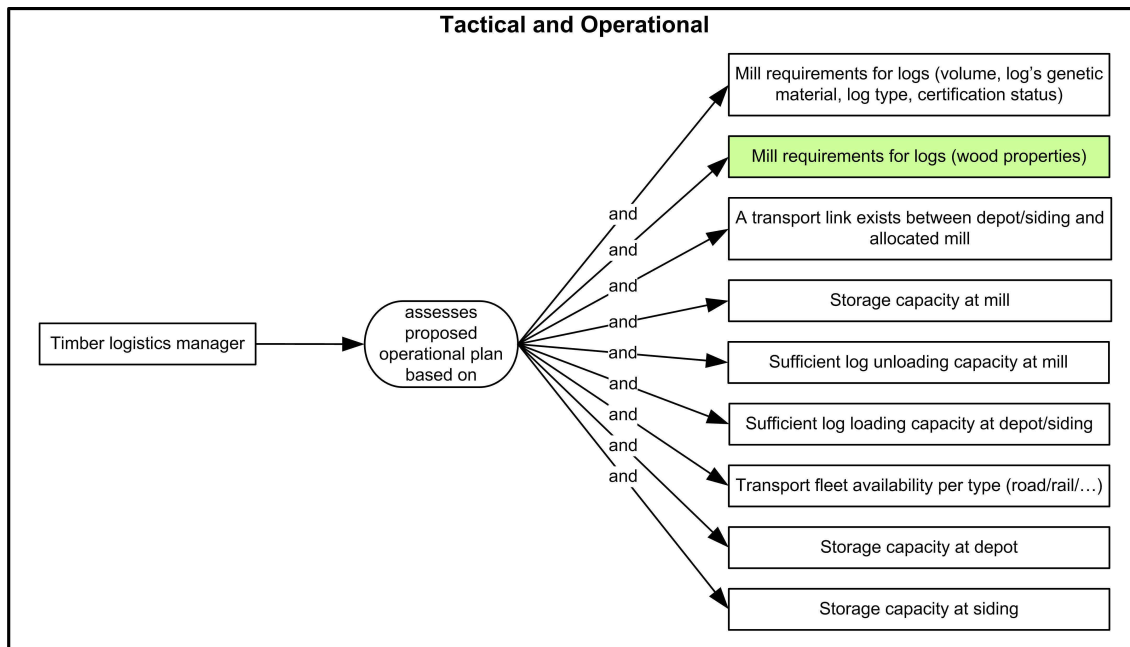


Figure C.10: ERD showing the updated criteria on which the tactical and operational plans are assessed by the Timber logistics manager

C.6 Planning domain: planning process

The planning process, which was described in section 4.6, was illustrated using business process diagrams (Figures 4.41 to 4.58 on pages 114 to 128). This section shows how the planning process would be changed after the implementation of the forest harvest scheduling system. The business process diagrams are shown in Figures C.11 to C.28. In the planning at the strategic level, the output of the plan is assessed per mill process, and the wood properties of the logs allocated to the process are assessed (see Figure C.12). If there are logs produced by the forest which have unsuitable genetic material or wood properties, the Timber logistics manager is notified so that he can attempt to find a market for them (Figure C.13 on page 263). When developing the tactical plan, the Co-ordinating and Managing foresters assess whether the logs' wood properties are appropriate for each mill's process to which they are allocated (see Figure C.16 on page 265). Similarly, when the Timber logistics manager assesses the tactical plan, he also makes sure that the wood properties are appropriate for each mill's process (see Figure C.21 on page 267). In this case, if the plantation has produced timber with unacceptable genetic material or properties the Timber logistics manager makes plans to sell it. If there is a shortfall of timber with appropriate genetic material or properties, the Timber logistics manager tries to source additional timber.

When developing the operational plan, the Co-ordinating and Managing foresters assess whether the logs' wood properties are appropriate for each mill's process to which

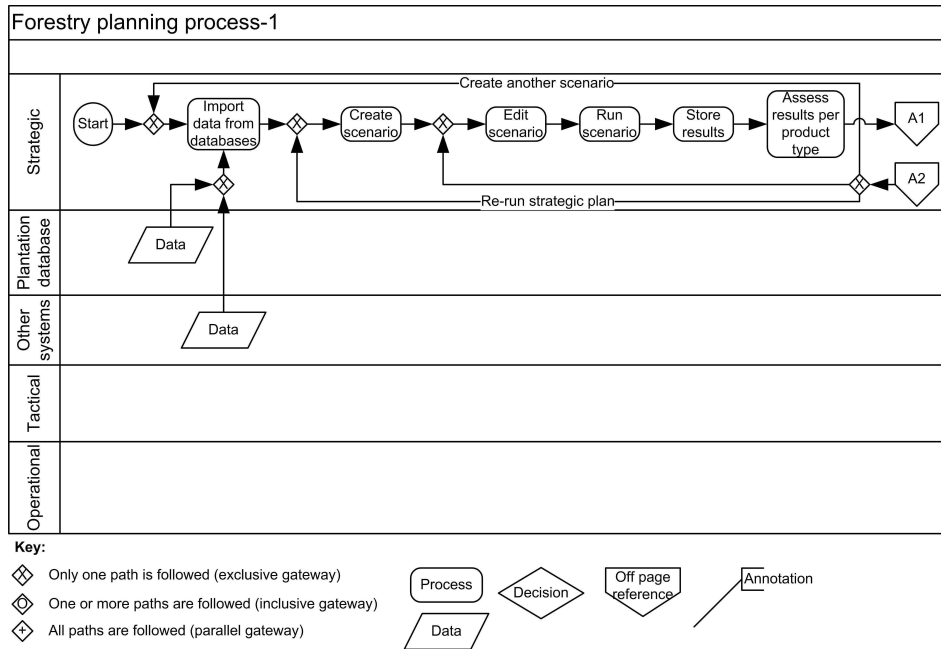


Figure C.11: Updated business process diagram of the planning process (part 1 of 18)

they are allocated (see Figure C.24 on page 269). Similarly, when the Timber logistics manager assesses the operational plan, he also checks that the wood properties are appropriate for each mill's process (see Figure C.28 on page 271). In this case, if the plantation has produced timber with unacceptable genetic material or properties the Timber logistics manager plans to sell it. If there is a shortfall of timber with appropriate genetic material or properties, the Timber logistics manager tries to obtain additional timber with the appropriate genetic material or properties.

The diagrams mentioned in this section which were changed can be compared to their respective pre-change counterparts: Figure C.12 can be compared to Figure 4.42 on page 263; Figure C.13 on page 263 can be contrasted with Figure 4.43 on page 116; Figure C.16 on page 265 can be compared to Figure 4.46 on page 119; C.21 on page 267 is the changed version of 4.51 on page 124; Figure C.24 on page 269 is an updated version of Figure 4.54 on page 126; and Figure C.28 on page 271 can be compared to Figure 4.58 on page 128.

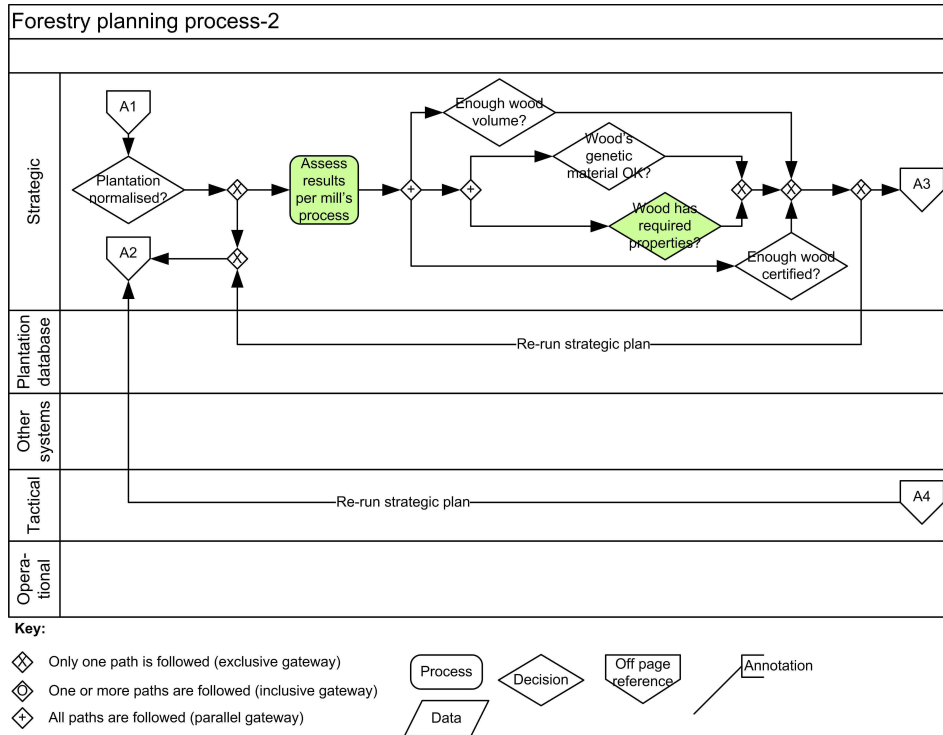


Figure C.12: Updated business process diagram of the planning process (part 2 of 18)

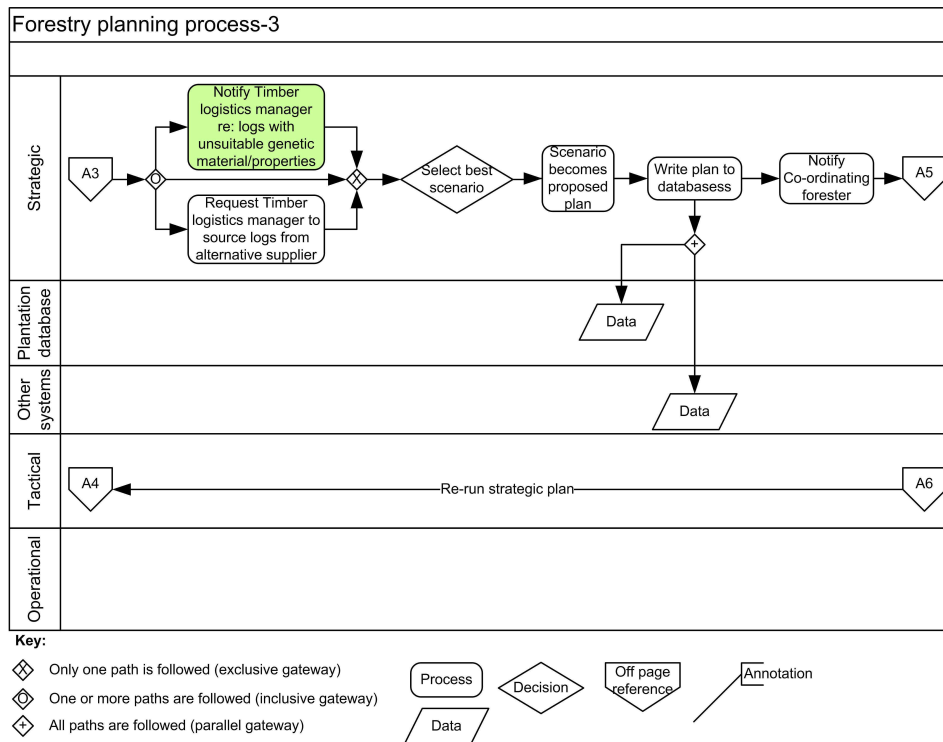


Figure C.13: Updated business process diagram of the planning process (part 3 of 18)

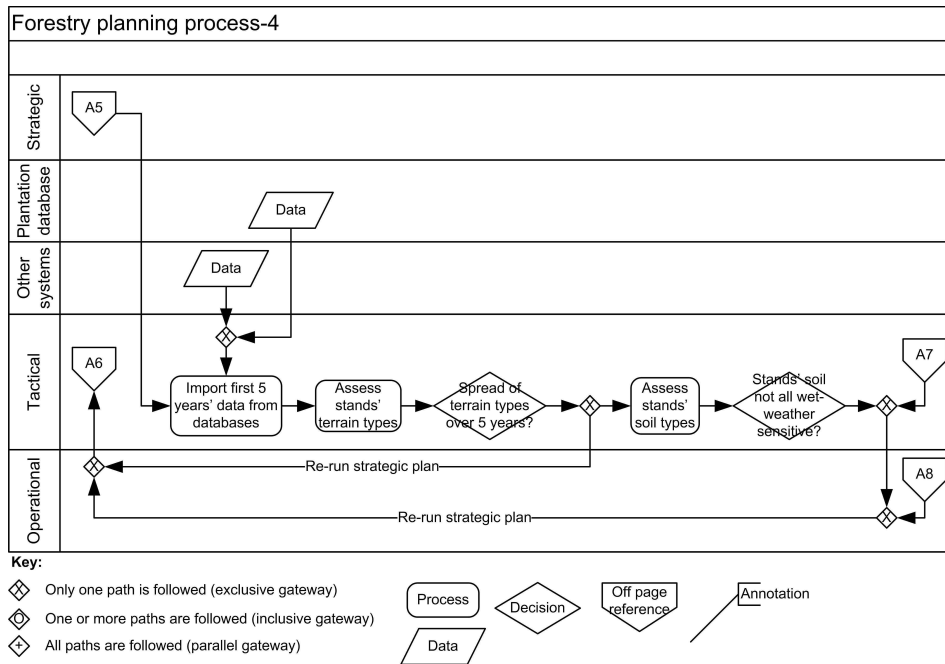


Figure C.14: Updated business process diagram of the planning process (part 4 of 18)

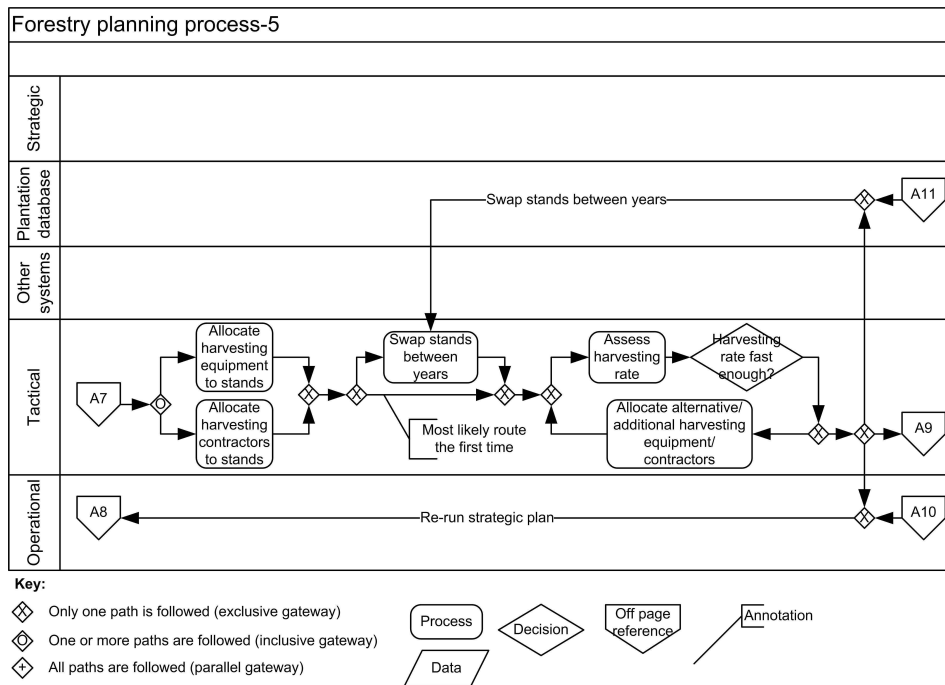


Figure C.15: Updated business process diagram of the planning process (part 5 of 18)

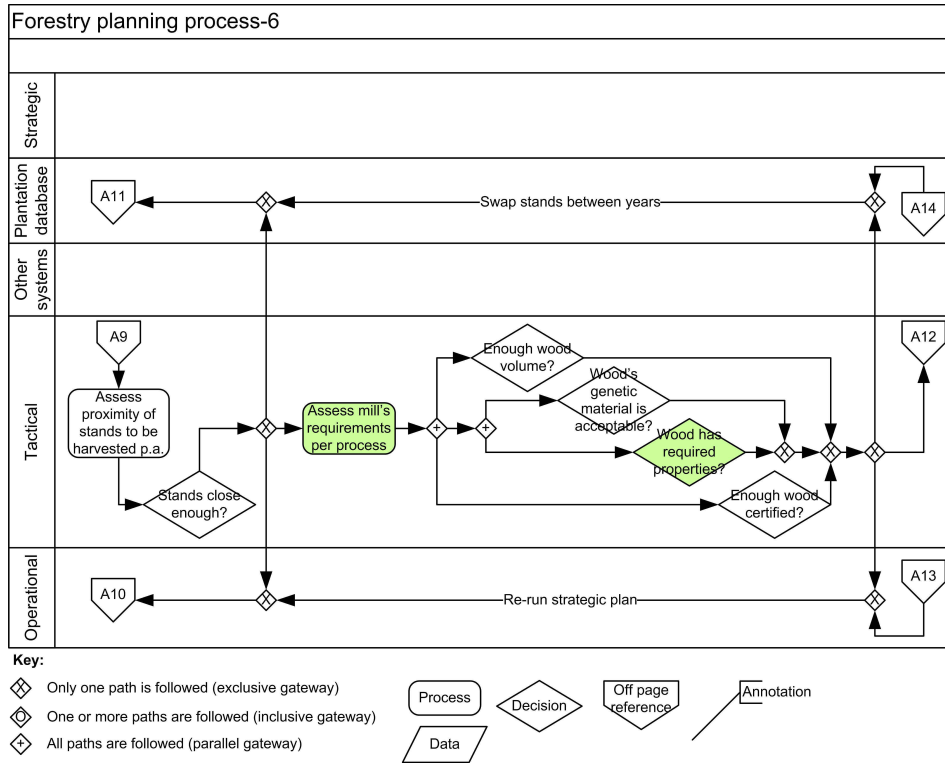


Figure C.16: Updated business process diagram of the planning process (part 6 of 18)

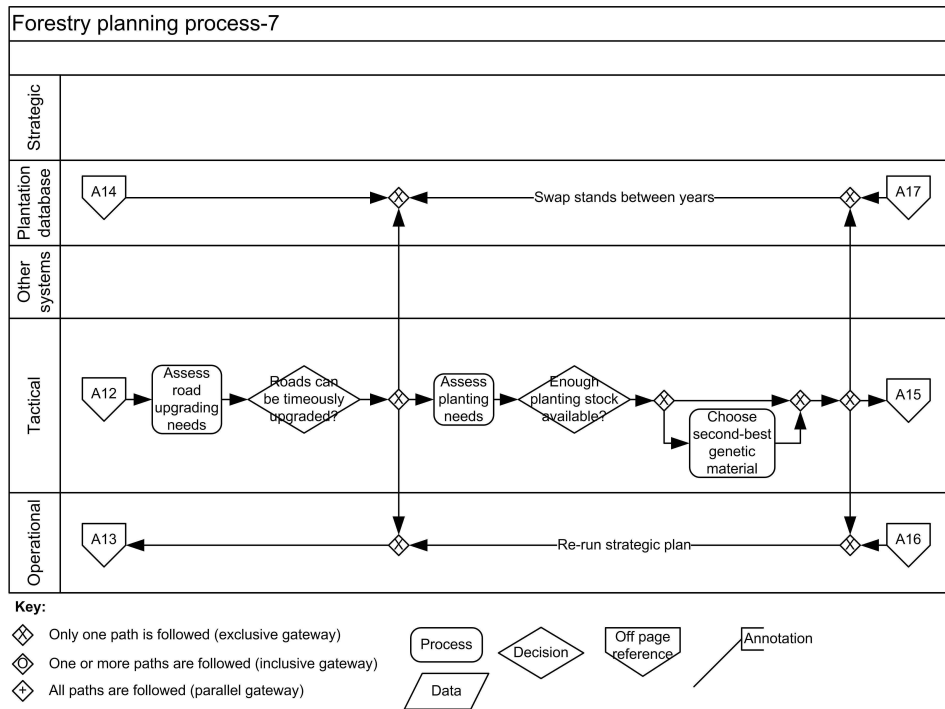


Figure C.17: Updated business process diagram of the planning process (part 7 of 18)

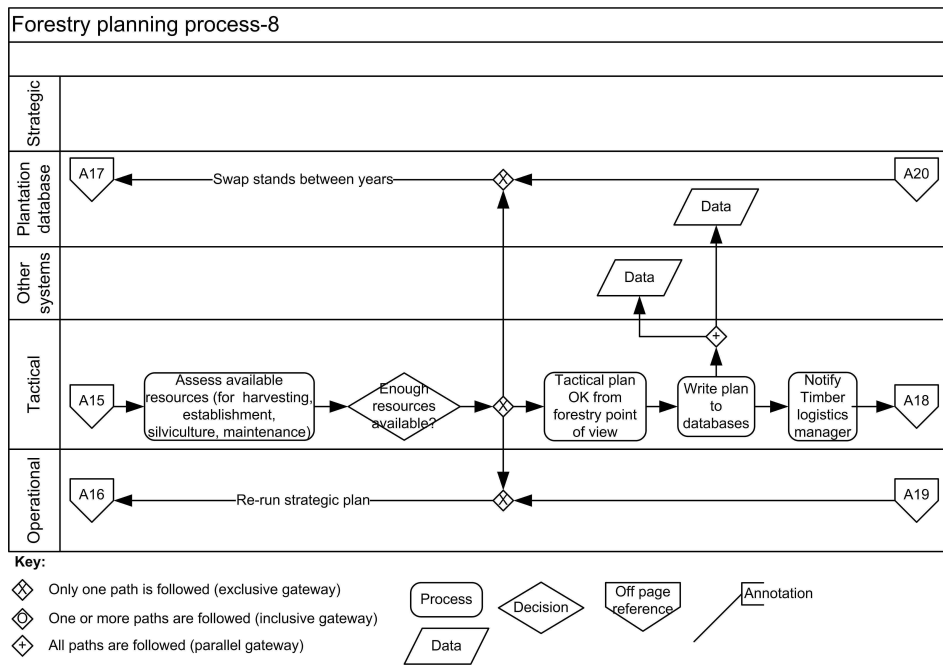


Figure C.18: Updated business process diagram of the planning process (part 8 of 18)

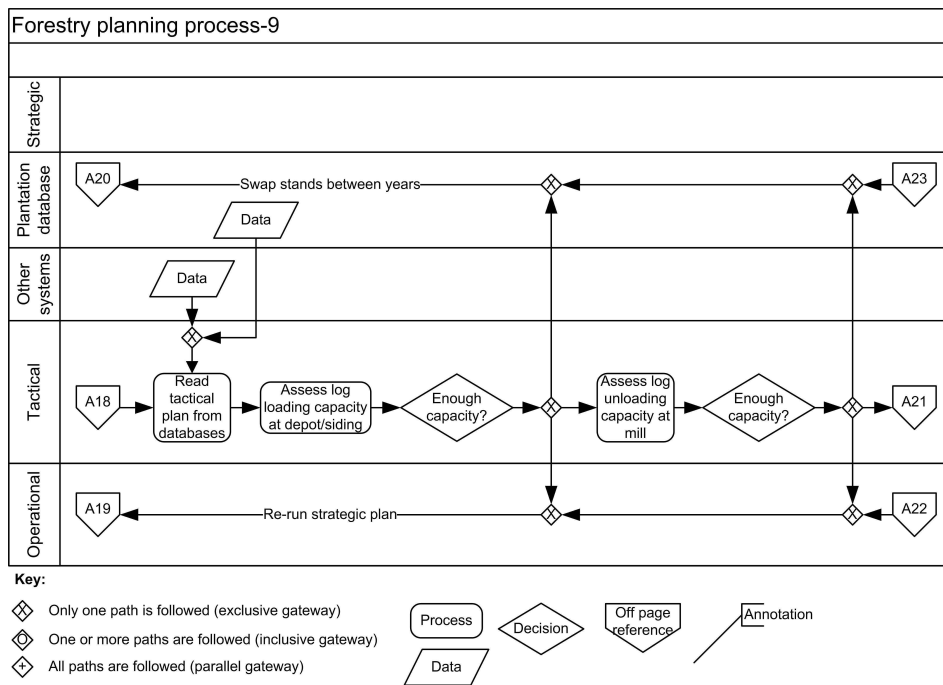


Figure C.19: Updated business process diagram of the planning process (part 9 of 18)

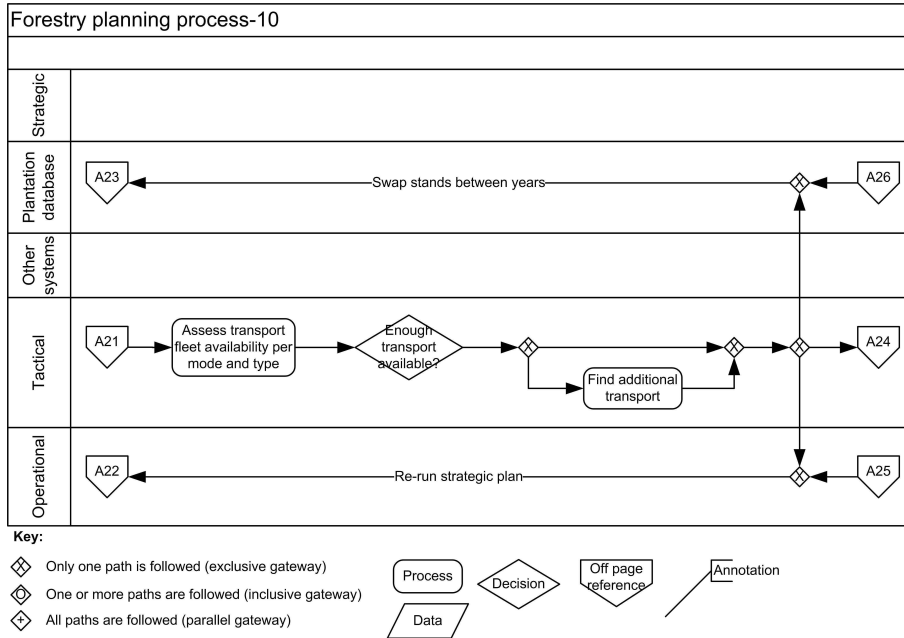


Figure C.20: Updated business process diagram of the planning process (part 10 of 18)

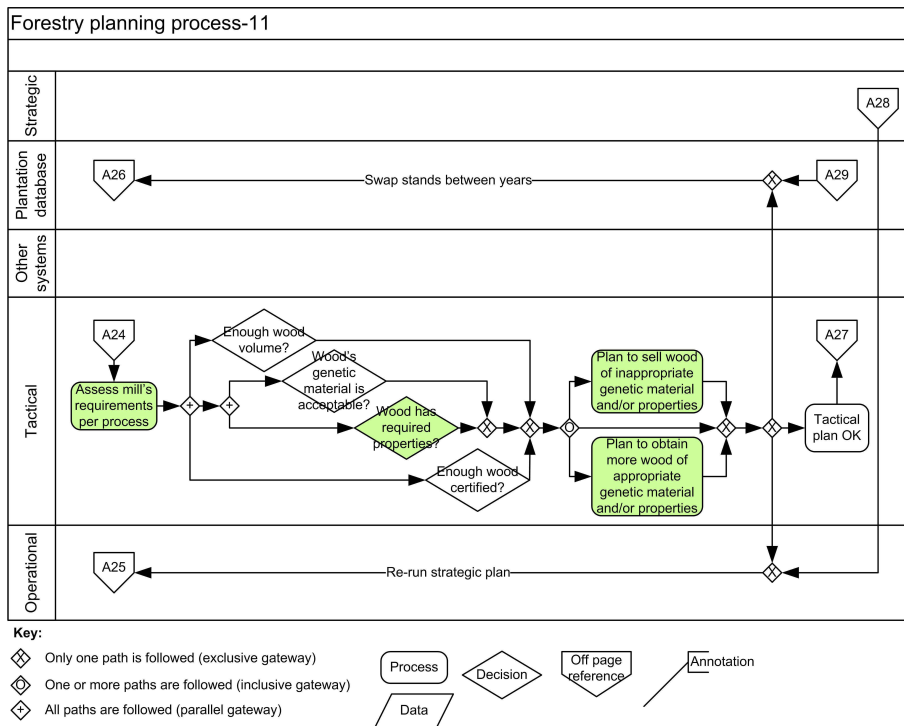


Figure C.21: Updated business process diagram of the planning process (part 11 of 18)

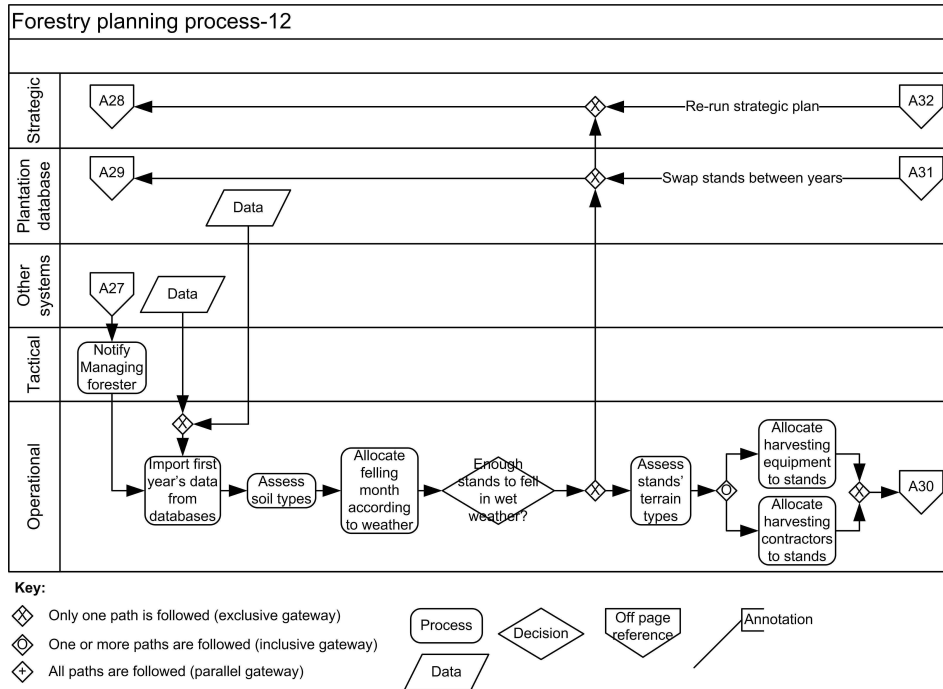


Figure C.22: Updated business process diagram of the planning process (part 12 of 18)

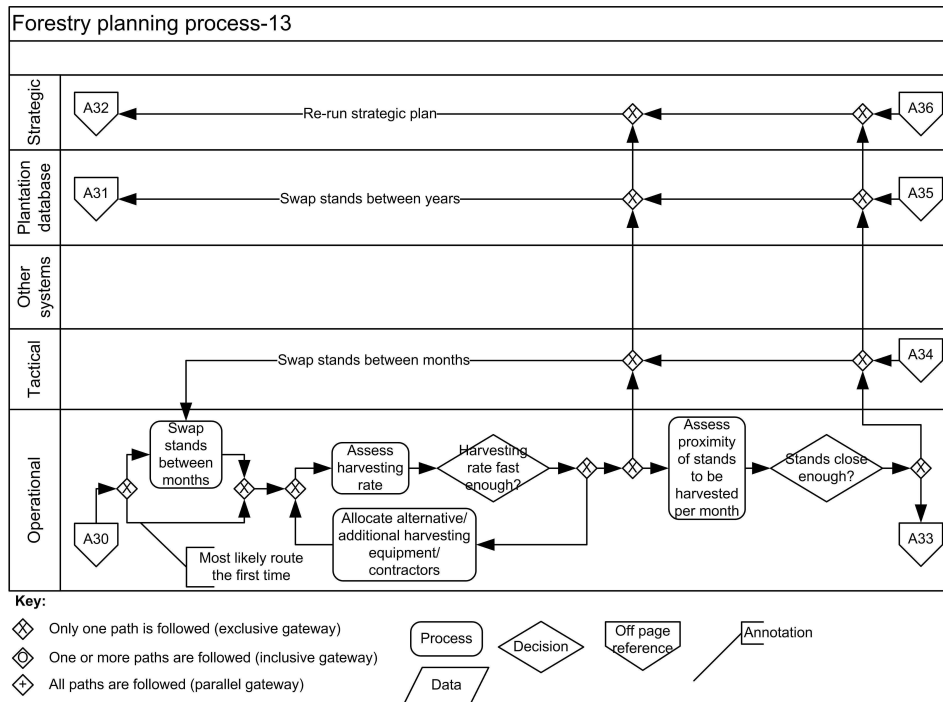


Figure C.23: Updated business process diagram of the planning process (part 13 of 18)

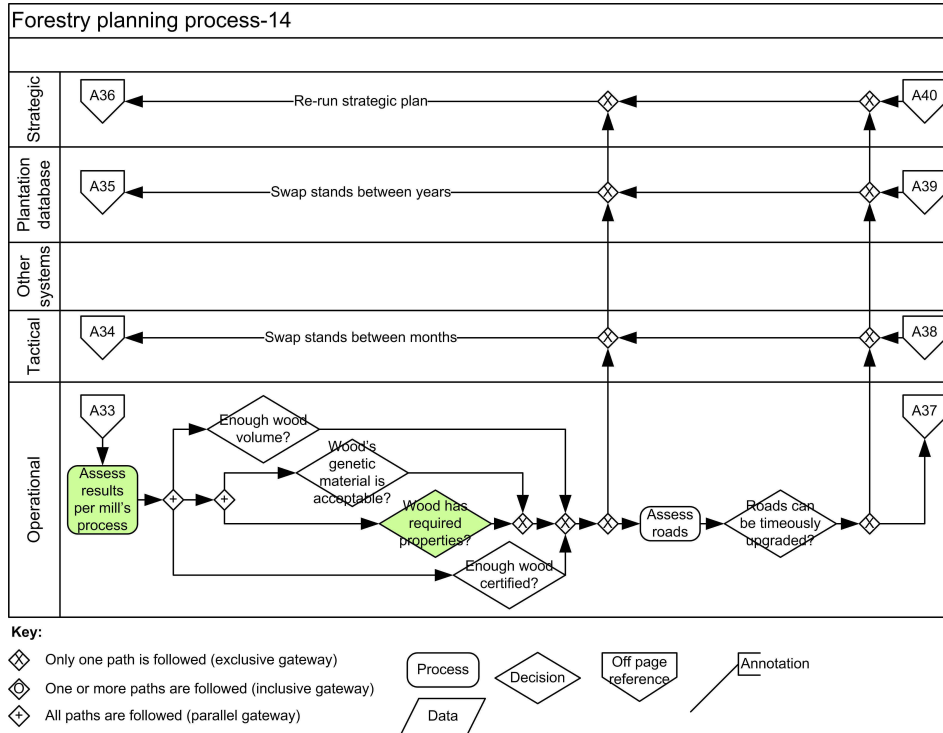


Figure C.24: Updated business process diagram of the planning process (part 14 of 18)

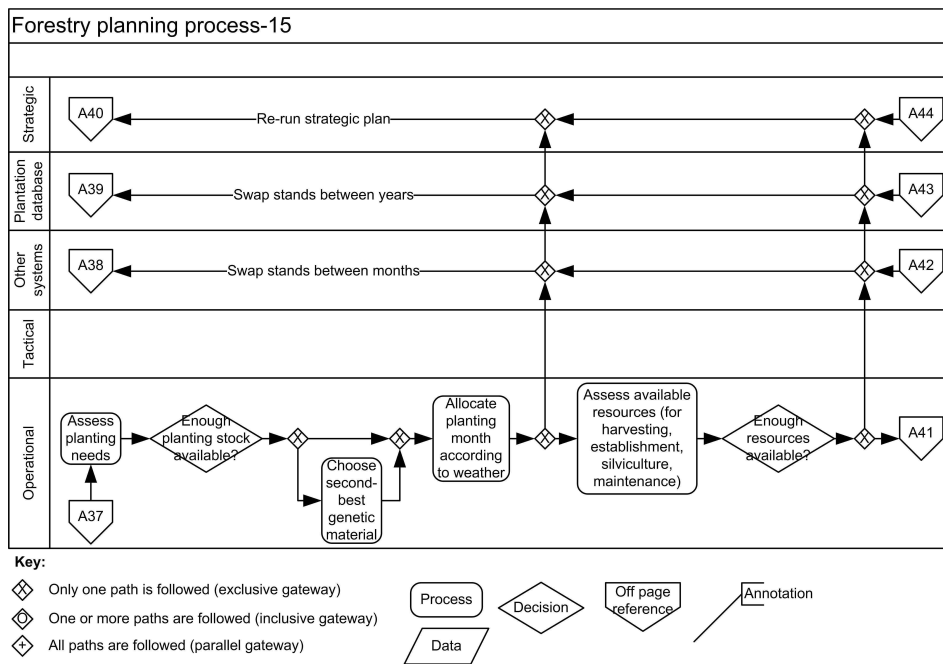


Figure C.25: Updated business process diagram of the planning process (part 15 of 18)

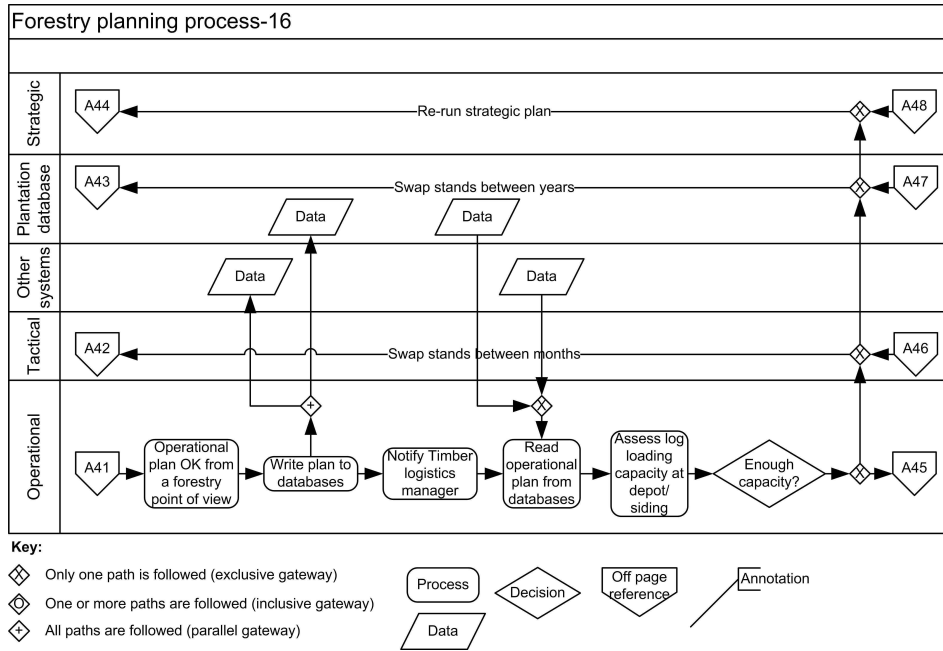


Figure C.26: Updated business process diagram of the planning process (part 16 of 18)

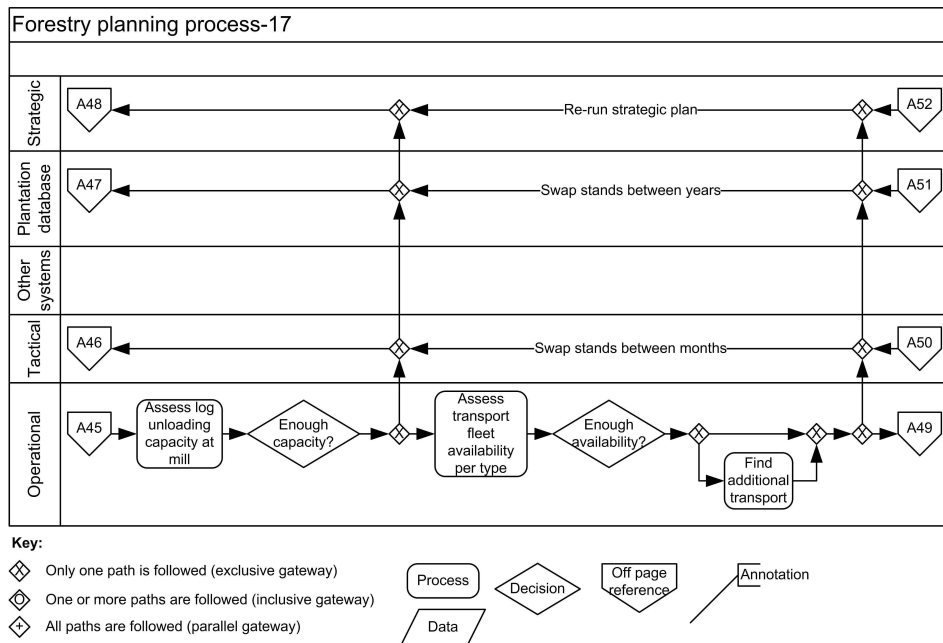


Figure C.27: Updated business process diagram of the planning process (part 17 of 18)

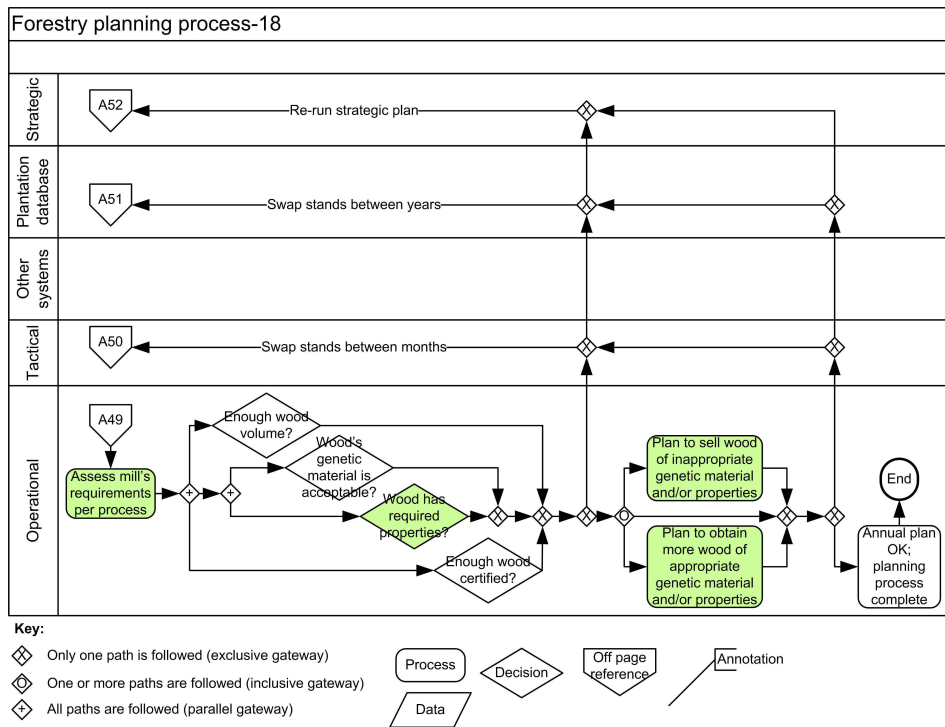


Figure C.28: Updated business process diagram of the planning process (part 18 of 18)

Appendix D

Formal specification of the forest-to-mill domain

This appendix gives the full and more detailed description of the forest-to-mill domain using the Z notation. The more abstract version of these specifications can be found in Chapter 5 in section 5.1. An index of schema names which are found in this appendix and in Appendix E can be found on page 431.

D.1 Overview of the forest-to-mill supply chain

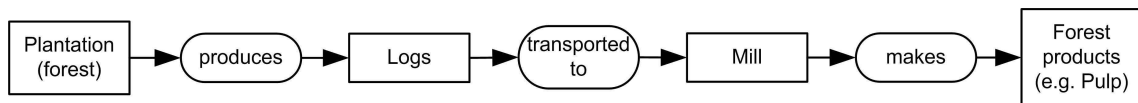


Figure D.1: ER diagram giving an overview of the forest-to-mill supply chain

The key entities in the plantation forest-to-mill supply chain are the forest, logs, the pulp mill and the pulp product. The forest grows trees, which are cut into logs. These logs are transported to the mill, where they are chipped and processed to become forest products (e.g. pulp and paper). Figure D.1 shows an entity-relationship diagram of the overview of the main entities involved. This overview diagram is expanded further in Figure D.2, which gives a state chart of the forest-to-mill supply chain. Figure D.3 shows the parts of this state chart which were modelled in the Z specification.

This state chart shows the states represented by circles, and the events (denoted by annotated arrows) which cause the states to change. Figure D.3 can be used as an overview of the forest-to-mill domain described in this specification, because in Z, schemas are used first of all to describe the state of a system, and then to describe the

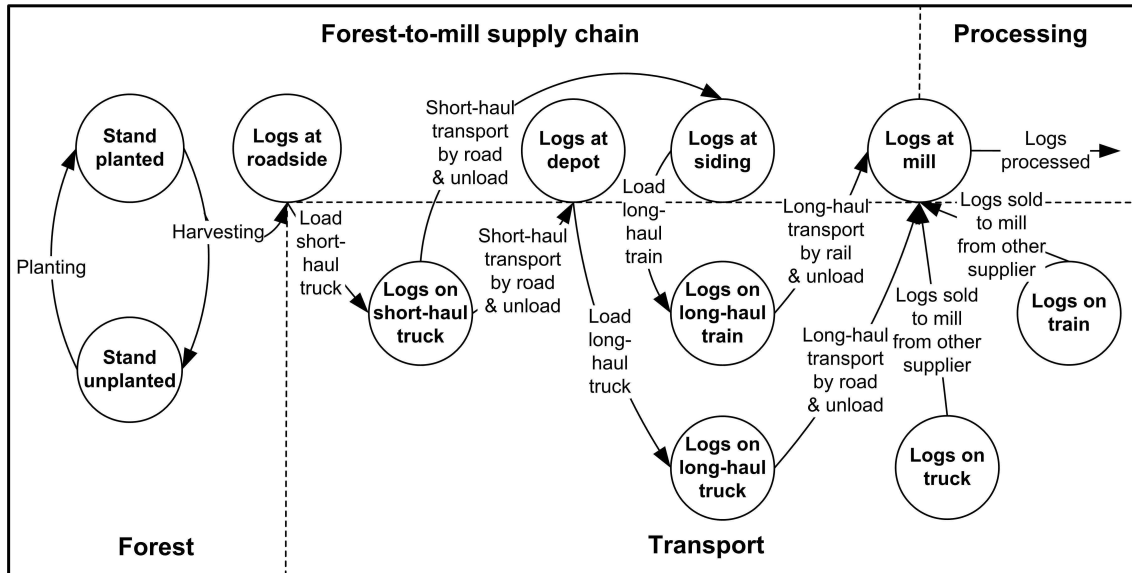


Figure D.2: State chart giving an overview of the forest-to-mill supply chain

actions which would cause those states (or parts of them) to change.

The left-hand side of Figure D.3 shows the forest, with stands being planted and harvested. The plantation forestry aspects of this specification are described in section D.4. The action schemas for planting the stand are described in section D.4.4. The action schemas for harvesting the stand, extracting the trees to roadside and making logs is described in section D.4.5. The logs at roadside are described in section D.3.4. The forest-to-mill logistics chain is described in section D.3. This chain ensures that there is a mill for every stand of trees planted, and that the mill is constantly supplied with wood from the stands. The transport of the logs is described in section D.5. Roads and trucks are described in sections D.5.2 and D.5.3 respectively. The short-haul action schemas for loading the logs at the stand and transporting them to the depot are described in section D.5.4. The long-haul action schemas for loading the logs at the depot and transporting them to the mill are described in section D.5.5. Should logs arrive at the mill from another grower (i.e. a grower other than the integrated plantation company), this is described in section D.5.7. The schema which describes how the logs are removed from the mill's logyard for processing is described in section D.6. The whole forest-to-mill supply chain is described in section D.7 by combining the plantation, transport and processing schemas.

Before the plantation forestry aspects can be specified, certain definitions and specifications are needed. These are given in sections D.2 and D.3, where the trees' genetic material is described (see section D.2.1), the regimes according to which the trees are grown are defined (see section D.2.2) and the logistics chain from the forest to the depot to the mill is defined (see section D.3).

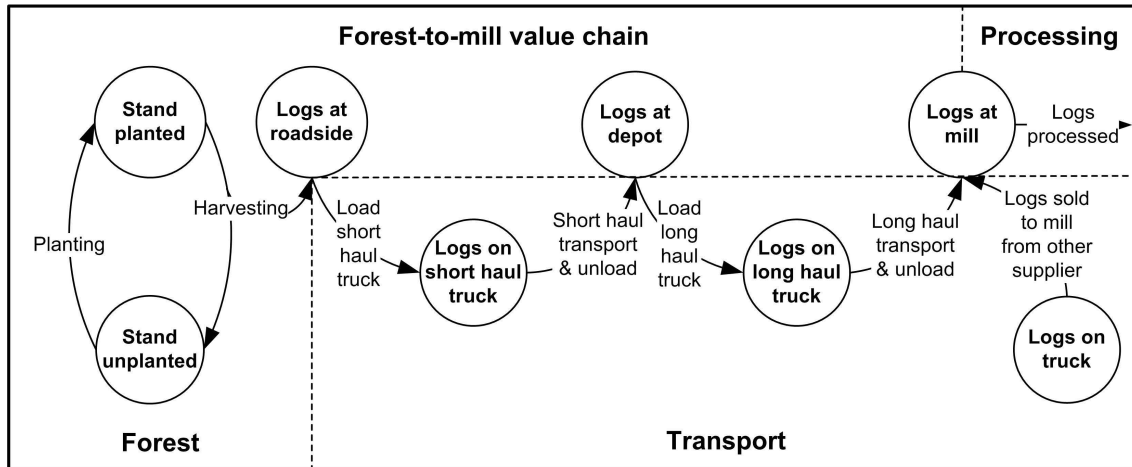


Figure D.3: State chart giving an overview of the forest-to-mill supply chain modelled in the Z specifications

D.2 Forest management

In specifications, since one must declare something before using it (Stepney *et al.*, 2003), some aspects of forest management need to be defined and specified before they can be used in later sections. These aspects include the composition of the genetic material of the trees planted in the forests, and the regimes (or rules) according to which the trees are planted, managed and harvested in the stand.

D.2.1 Genetic material of trees

Genetic material is a term used to describe the genetic material of biological matter, and in particular, trees. Trees are defined in terms of genus (plural genera) and species. A tree has both a genus and a species. Trees which have the same genus can be crossed with each other (i.e. cross-pollinated with each other) to make a hybrid tree. Foresters refer loosely to 'species' when they mean genus and species, or genus and species x species, (as in 'site-species matching' and 'species suitability' for a mill).

The specification begins with definitions of types which will be used in this section, and also later in the specification. The names of the genetic material which are used in the forest are defined first (in schema *GeneticMaterialNames*). Then the genus of the trees is specified in schema *Genus*. The species of the trees are described next: trees may be of pure species, in which case both parents have the same genus and

species. This is described in schema *PureSpecies*. Alternatively, the trees may be hybrids, which means that their parents have the same genus, but different species. This is described in schema *Hybrid*.

In Chapter 4, section 4.3.3, the trees' genetic material was described. Figures D.4 and D.5 show the hierarchy of genetic material and the relationship between pure species and hybrids, seedlings, cuttings and clones respectively. In this specification, the genetic material hierarchy of genus and species was modelled (see Figure D.6), and only pure species and hybrids were modelled (see Figure D.7).

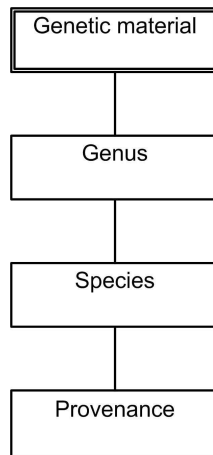


Figure D.4: Hierarchy of genetic material (loosely called “species”)

D.2.1.1 Preliminary definitions

GENUSID, *SPECIESID* and *GENMATERIALID* are unique identifiers of genera, species and genetic material descriptions (respectively). *GENMATERIALID* describes any of the genetic material constituents (genera, species, hybrids, etc.). The names of the genera and species (*GENMATNAME*) are specified with a sequence of characters (*CHAR*).

```
[GENUSID, SPECIESID, GENMATERIALID]  
[CHAR]  
GENMATNAME == seq CHAR
```

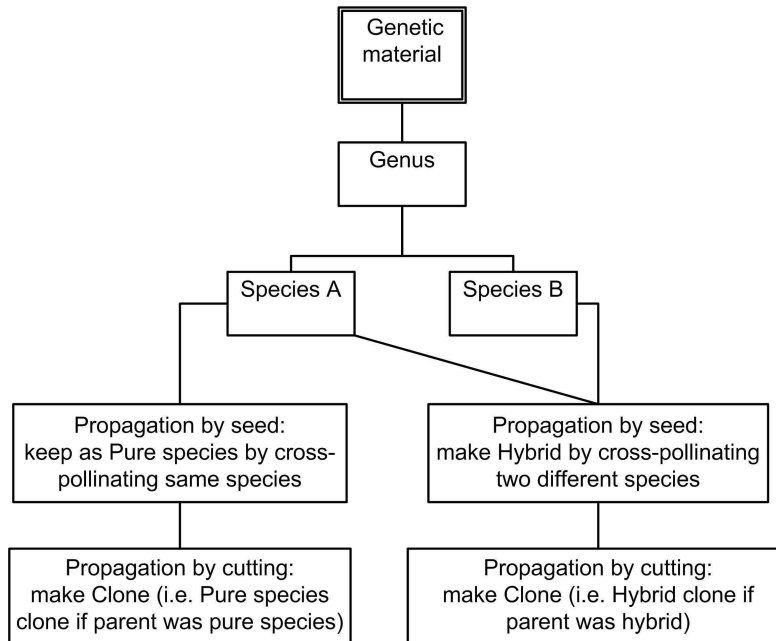


Figure D.5: Diagram showing the relationship between pure species and hybrids, seedlings, cuttings and clones

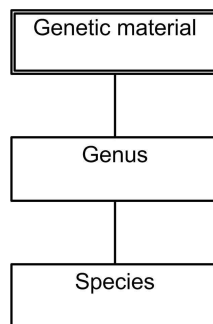


Figure D.6: Hierarchy of genetic material (loosely called “species”) modelled in the Z specification

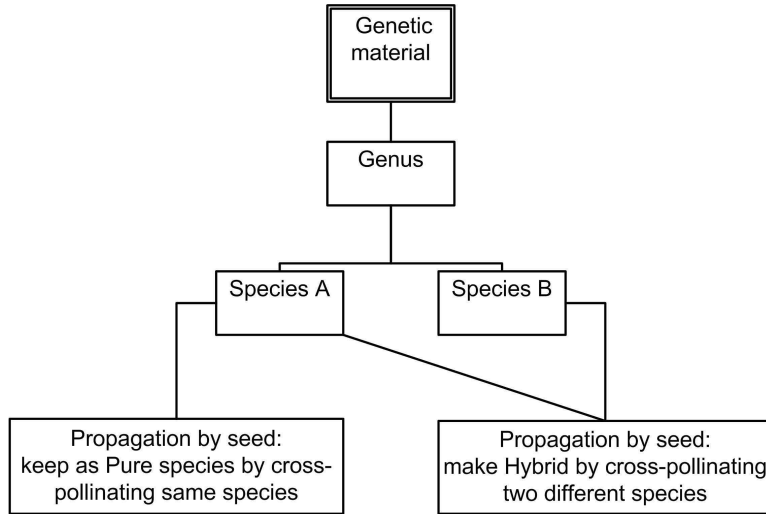
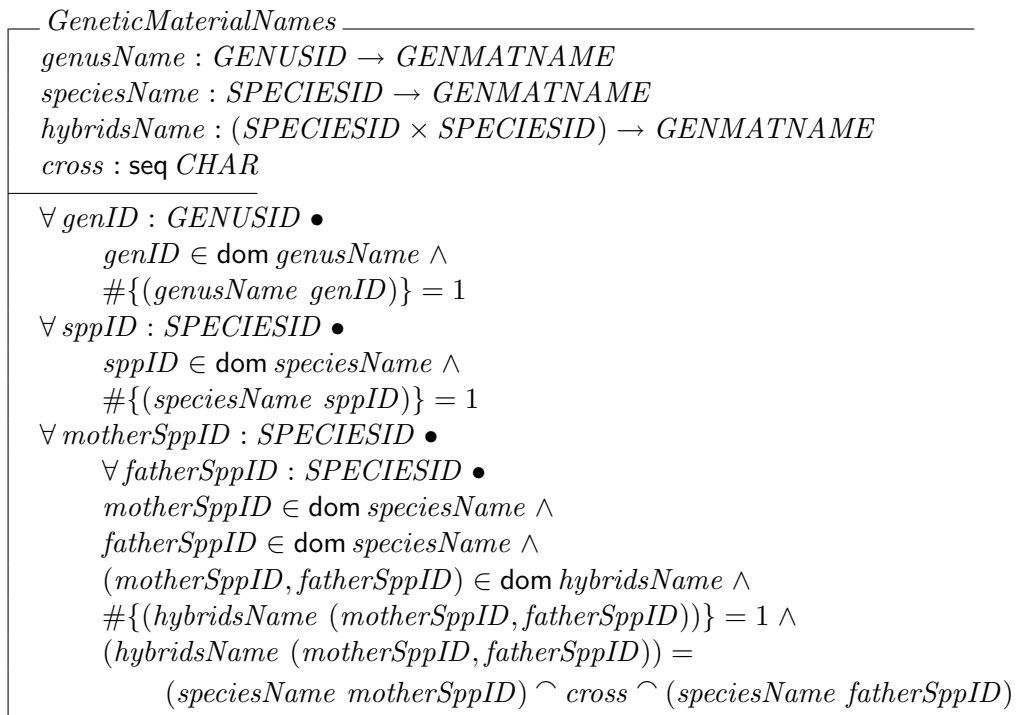


Figure D.7: Diagram showing the relationship between pure species and hybrids modelled in the Z specification

D.2.1.2 Genetic material names

The botanical names of the genetic material is described in the schema *GeneticMaterialNames*.



The schema *GeneticMaterialNames* describes the sets of names of genera (the plural of genus), of species and of hybrids. It contains four functions. *genusName* has as input a genus' ID and outputs the genus' botanical name; *speciesName* has as input a species' ID and outputs the species' botanical name; *hybridsName* has as input two species' IDs and outputs the the hybrid's botanical name; *cross* is a sequence (string) of characters (*CHAR*), and will contain the string ' x ' or ' cross ' in the final implementation. Each unique genus, species and hybrid has a single botanical name. The hybrid's name is made by concatenating the mother-species' name with the father-species' name (in that order), with a cross in between (e.g. *grandis* x *camaldulensis*, *patula* x *elliottii*).

D.2.1.3 Genus

The genus of the trees in the forest is described in the schema *Genus*.

The schema *Genus* describes the genera of trees which can be planted in the plantation. It contains the unchanged schema *GeneticMaterialNames* and two functions, *hasSpecies* and *genusWood*. *hasSpecies* relates the genus' ID to all the species of that genus which could be planted. *genusWood* relates a genetic material ID to the tree or log or chips or pulp's genus. Each genus has a related species and genetic material ID; they are related as follows: each genus has one botanical name (e.g. *Eucalyptus*, *Pinus*, *Acacia*), and has one or more species related to it.

<i>Genus</i>
$\exists GeneticMaterialNames$ $hasSpecies : GENUSID \rightarrow SPECIESID$ $genusWood : GENMATERIALID \rightarrow GENUSID$
$\forall genID : GENUSID \bullet$ $\exists sppID : SPECIESID \bullet$ $\exists genMatID : GENMATERIALID \bullet$ $genID \in \text{dom } genusName \wedge$ $genID \in \text{dom } hasSpecies \wedge$ $sppID \in \text{ran } hasSpecies \wedge$ $genMatID \in \text{dom } genusWood \wedge$ $genID \in \text{ran } genusWood \wedge$ $\#\{(genusName \ genID)\} = 1 \wedge$ $(hasSpecies \ genID) = sppID \wedge$ $\#\{(hasSpecies \ genID)\} \geq 1$

D.2.1.4 Species

The species of the trees in the forest is described next. There are two options: the tree is of pure species or is a hybrid. These two options are described in the schemas

PureSpecies and *Hybrid*.

The term ‘pure species’ means trees whose parents have the same genus and species. The schema *PureSpecies* describes the species of trees of pure species which can be planted in the plantation. It includes the unchanged schemas *Genus* and *GeneticMaterialNames* and contains three functions, *pureSpeciesName*, *pureSpeciesWood* and *genusOfPureSpeciesWood*. *pureSpeciesName* relates a pure species tree’s genus and species to its name (e.g. *Pinus patula*, *Eucalyptus grandis*), while *pureSpeciesWood* relates the genetic material’s ID to the genus and species of the tree. This function can equally be used to specify the genus and species of logs, chips or pulp. *genusOfPureSpeciesWood* relates the genetic material ID of a pure species tree to its genus. If there are trees of pure species present in the plantation¹, each genus-species combination has a single name, and the full name of the pure species’ tree (or log or chips or pulp) is made by adding the species’ name to the end of the genus’ name (i.e. concatenating the genus name with the species name).

<p><i>PureSpecies</i></p> <p>$\exists Genus$ $\exists GeneticMaterialNames$ <i>pureSpeciesName</i> : $(GENUSID \times SPECIESID) \rightarrow GENMATNAME$ <i>pureSpeciesWood</i> : $GENMATERIALID \rightarrow (GENUSID \times SPECIESID)$ <i>genusOfPureSpeciesWood</i> : $GENMATERIALID \rightarrow GENUSID$</p> <hr/> <p>$\forall genID : GENUSID \bullet$ $\exists sppID : SPECIESID \bullet$ $\exists genMatID : GENMATERIALID \bullet$ $genID \in \text{dom } genusName \wedge$ $genID \in \text{dom } hasSpecies \wedge$ $sppID \in \text{ran } hasSpecies \wedge$ $sppID \in \text{dom } speciesName \wedge$ $(genID, sppID) \in \text{dom } pureSpeciesName \wedge$ $genMatID \in \text{dom } pureSpeciesWood \wedge$ $genMatID \in \text{dom } genusOfPureSpeciesWood \wedge$ $(genID, sppID) \in \text{ran } pureSpeciesWood \wedge$ $genID \in \text{ran } genusOfPureSpeciesWood \wedge$ $\#\{\text{ran } pureSpeciesWood\} \geq 1 \Rightarrow$ $(\#\{pureSpeciesName (pureSpeciesWood genMatID)\} = 1 \wedge$ $pureSpeciesName (genID, sppID) =$ $(genusName genID) \wedge (speciesName sppID)) \wedge$ $(genusOfPureSpeciesWood genMatID) = genID$</p>

The term ‘hybrid’ means that trees have parents with the same genus but different species. The order of the species is important when describing hybrids: the mother is

¹There might only be hybrids - see schema *Hybrid*.

mentioned first and the father second; they are also called ‘crosses’.

The schema *Hybrid* describes the species of hybrid trees which can be planted in the plantation. It includes the unchanged schemas *Genus* and *GeneticMaterialNames* and contains three functions, *hybridName*, *hybridWood* and *genusOfHybridWood*. *hybridName* relates a hybrid tree’s genus and two species to its name (e.g. *Pinus elliottii* × *caribaea*, *Eucalyptus grandis* × *nitens*), while *hybridWood* relates the genetic material’s ID to the genus and two species of the tree. This function can equally be used to specify the genus and two species of logs, chips or pulp. *genusOfHybridWood* relates the genetic material ID of a hybrid tree to its genus. If there are hybrid trees present in the plantation², each genus-species-species combination has a single name, and the full hybrid name is made by adding the two species names to the end of the genus name (i.e. concatenating the genus name with the two species’ names).

<p><i>Hybrid</i></p> <p>$\exists Genus$</p> <p>$\exists GeneticMaterialNames$</p> <p>$hybridName : (GENUSID \times SPECIESID \times SPECIESID) \rightarrow GENMATNAME$</p> <p>$hybridWood : GENMATRIALID \rightarrow (GENUSID \times SPECIESID \times SPECIESID)$</p> <p>$genusOfHybridWood : GENMATRIALID \rightarrow GENUSID$</p> <hr/> <p>$\forall genID : GENUSID \bullet$</p> <p>$\exists motherSppID : SPECIESID \bullet$</p> <p>$\exists fatherSppID : SPECIESID \bullet$</p> <p>$\exists genMatID : GENMATRIALID \bullet$</p> <p>$genID \in \text{dom } genusName \wedge$</p> <p>$genID \in \text{dom } hasSpecies \wedge$</p> <p>$motherSppID \in \text{ran } hasSpecies \wedge$</p> <p>$fatherSppID \in \text{ran } hasSpecies \wedge$</p> <p>$motherSppID \in \text{dom } speciesName \wedge$</p> <p>$fatherSppID \in \text{dom } speciesName \wedge$</p> <p>$(genID, motherSppID, fatherSppID) \in \text{dom } hybridName \wedge$</p> <p>$genMatID \in \text{dom } hybridWood \wedge$</p> <p>$genMatID \in \text{dom } genusOfHybridWood \wedge$</p> <p>$(genID, motherSppID, fatherSppID) \in \text{ran } hybridWood \wedge$</p> <p>$genID \in \text{ran } genusOfHybridWood \wedge$</p> <p>$\#\{\text{ran } hybridWood\} \geq 1 \Rightarrow$</p> <p>$\#\{hybridName (hybridWood genMatID)\} = 1 \wedge$</p> <p>$hybridName (genID, motherSppID, fatherSppID) =$</p> <p>$(genusName genID) \wedge (hybridsName (motherSppID, fatherSppID)) \wedge$</p> <p>$(genusOfHybridWood genMatID) = genID$</p>
--

²There might only be trees of pure species - see schema *PureSpecies*.

D.2.1.5 Wood's genetic material

The genetic material of the wood (of the trees, tree-lengths, logs, chips or pulp) in the forest is described next by combining the previous specifications.

The schema *WoodGeneticMaterial* combines the schemas *Genus*, *PureSpecies* and *Hybrid* in the following way: *Genus* and either *PureSpecies* or *Hybrid* are applicable. This reflects the reality that trees (or logs, chips or pulp) all have a genus, and either are of pure species or they are hybrids.

$$\text{WoodGeneticMaterial} \hat{=} \text{Genus} \wedge (\text{PureSpecies} \vee \text{Hybrid})$$

D.2.2 Regimes

A regime is an ordered sequence of steps which must be followed when planting and harvesting trees, and performing other silvicultural activities. Regimes may apply to different genetic material, e.g. there is a regime for all *Pinus* trees grown in a certain area, or there is a regime for a particular hybrid. Regimes are planned in advance. A stand could have many planned regimes, depending on the ID of the genetic material (which is informed by the results of site-species matching). At any one time, a stand can only have one actual regime, which is the record of what happened in the stand to date.

The specification of regimes begins with definitions which are used from this part of the specification onwards. The date is also defined. This is important for being able to capture what has happened in the stand (in the actual regime). The specification continues by defining the three building blocks of the regime: its genetic material, its planting details, and its felling (harvesting) details. The next part of the specification concerns recording which regimes are being used for each stand. Firstly, the actual regime (a record of what's been done in the stand so far) is tracked, so that the current actual regime for a stand can be identified. Then the stands' planned and actual regimes are defined. The planned regime governs what could happen in a stand, while the actual regime records what has actually happened in the regime. The storage of the regimes is defined next, in the schema *StoreRegimes* by combining previously defined schemas.

D.2.2.1 Preliminary definitions

The definitions needed for specifying the regime are given first. Each regime is uniquely identified with a regime ID (*REGIMEID*), and each stand with a stand ID (*STANDID*). *AGE* is defined to record the age of the trees planted in a stand; *DISTANCE* is defined to represent the distance between two locations; *SPH*

(stems per hectare) defines how many trees there are per hectare. Although these three numbers would normally be real numbers, they are defined here as natural numbers (because of a restriction in the Z type checker used). The regime type (*REGIMETYPE*) is specified as being either planned or actual.

[*REGIMEID*, *STANDID*]

| *AGE* : \mathbb{N}
 | *DISTANCE* : \mathbb{N}
 | *SPH* : \mathbb{N}

REGIMETYPE ::= *planned* | *actual*

D.2.2.2 Date definition

The schema *DateDefinition* defines three natural numbers: *day*, *month* and *year*. The year is defined to be in the range 1950 to 2550. The month is defined to be in the range 1 to 12. When the months are numbered 1, 3, 5, 7, 8, 10 or 12, the day is defined to be in the range 1 to 31. When the months are numbered 4, 6, 9 or 11, the day is defined to be in the range 1 to 30. For month 2, when the year divided by 400 leaves no remainder, or the year divided by 100 leaves a remainder and the year divided by 4 leaves no remainder, it's a leap year, which means that the day will be in the range 1 to 29. For month 2, when the above rules do not hold, it's not a leap year, which means that the day will be in the range 1 to 28.

DateDefinition

year, *month*, *day* : \mathbb{N}

$1950 \leq \textit{year} \leq 2550$

$1 \leq \textit{month} \leq 12$

$((\textit{month} = 1) \vee (\textit{month} = 3) \vee (\textit{month} = 5) \vee (\textit{month} = 7) \vee$
 $(\textit{month} = 8) \vee (\textit{month} = 10) \vee (\textit{month} = 12)) \Rightarrow$

$1 \leq \textit{day} \leq 31$

$((\textit{month} = 4) \vee (\textit{month} = 6) \vee (\textit{month} = 9) \vee (\textit{month} = 11)) \Rightarrow$
 $(1 \leq \textit{day} \leq 30)$

$((\textit{month} = 2) \wedge ((\textit{year} \bmod 400) = 0 \vee$
 $((\textit{year} \bmod 100 \neq 0) \vee (\textit{year} \bmod 4 = 0)))) \Rightarrow$

$1 \leq \textit{day} \leq 29$

$((\textit{month} = 2) \wedge \neg ((\textit{year} \bmod 400) = 0 \vee$
 $((\textit{year} \bmod 100 \neq 0) \vee (\textit{year} \bmod 4 = 0)))) \Rightarrow$

$1 \leq \textit{day} \leq 28$

The type *DATE* is defined as a finite set of the type defined by schema *DateDefinition*.

| *DATE* : \mathbb{F} *DateDefinition*

D.2.2.3 Regime's genetic material

The genetic material of the trees to be planted in the stand is described in the schema *RegimeGeneticMaterial*. For the planned regime, this could be defined at a genus level, or at a genus and species level (pure species and/or hybrids). For the actual regime, the genetic material is defined at the species level.

The schema *RegimeGeneticMaterial* defines the genetic material which is specified in the regime. It includes the unchanged schema *WoodGeneticMaterial* and the function, *regimeGeneticMaterial*. This function has as input the regime's ID and regime type (planned or actual) and output the genetic material's ID (which has type *GENMATERIALID*). If the regime is a planned regime, the genetic material ID could relate to a genus, a genus-species, or a genus-species-species combination. This means that the planned regime could be applied, for example, to all *Pinus* trees, or all *Pinus elliotii* trees, or all *Pinus elliotii* × *caribaea* trees (depending what was specified). However, if the regime is an actual regime, there must be a specific genus-species (pure species) or genus-species-species (hybrid) combination to which is it applied. Each regime and regime type combination applies to only one genetic material ID.

<p><i>RegimeGeneticMaterial</i></p> <p>\exists <i>WoodGeneticMaterial</i></p> <p><i>regimeGeneticMaterial</i> : (REGIMEID × REGIMETYPE) → GENMATERIALID</p> <p>\forall <i>regID</i> : REGIMEID •</p> <p> \forall <i>genMatID</i> : GENMATERIALID •</p> <p> \forall <i>regType</i> : REGIMETYPE •</p> <p> (<i>regID</i>, <i>regType</i>) ∈ dom <i>regimeGeneticMaterial</i> ∧</p> <p> <i>genMatID</i> ∈ ran <i>regimeGeneticMaterial</i> ∧</p> <p> (<i>regType</i> = <i>planned</i> ⇒</p> <p> (<i>genMatID</i> ∈ dom <i>genusWood</i> ∨</p> <p> <i>genMatID</i> ∈ dom <i>pureSpeciesWood</i> ∨</p> <p> <i>genMatID</i> ∈ dom <i>hybridWood</i>)) ∧</p> <p> (<i>regType</i> = <i>actual</i> ⇒</p> <p> (<i>genMatID</i> ∈ dom <i>pureSpeciesWood</i> ∨</p> <p> <i>genMatID</i> ∈ dom <i>hybridWood</i>)) ∧</p> <p> #\{(<i>regimeGeneticMaterial</i> (<i>regID</i>, <i>regType</i>))\} = 1</p>
--

D.2.2.4 Regime's planting details

The regime's planting details are described in the next section. For the planned regime, the planting details are described in terms of the age of the trees in the stand; for the actual regime, the date is recorded. The schema *RegimePlanting* includes the two schemas *RegimePlantAge* and *RegimePlantDate*.

The schema *RegimePlantAge* specifies the function *plantAge*, which has the regime's ID and regime type as its inputs. The function *plantAge* gives the age of the stand at planting (this is always zero). This function is used for the planned regime (and is undefined for the actual regime).

<i>RegimePlantAge</i>
$plantAge : (REGIMEID \times REGIMETYPE) \rightarrow AGE$
$\forall regID : REGIMEID \bullet$ $\quad \forall regType : REGIMETYPE \bullet$ $\quad (regID, regType) \in \text{dom } plantAge \wedge$ $\quad (plantAge (regID, planned)) = 0 \wedge$ $\quad \{(plantAge (regID, actual))\} = \emptyset$

The schema *RegimePlantDate* specifies the function *plantDate*, which has the regime's ID and regime type as its inputs. The function *plantDate* is used for the actual regime, and has as output the date (day, month and year) of planting. It is undefined for the planned regime.

<i>RegimePlantDate</i>
$plantDate : (REGIMEID \times REGIMETYPE) \rightarrow DATE$
$\forall regID : REGIMEID \bullet$ $\quad \forall regType : REGIMETYPE \bullet$ $\quad (regID, regType) \in \text{dom } plantDate \wedge$ $\quad \{(plantDate (regID, planned))\} = \emptyset$

The schema *RegimePlanting* includes two schemas, *RegimePlantAge* and *RegimePlantDate*, and two functions, *plantDensity* and *plantedSPH*, which have the regime's ID and regime type as their input value. The function *plantDensity* gives the distances between the trees in the row and between the rows in the stand. These distances are always greater than zero (e.g. 2.7m x 2.7m, or 3m x 2m). (Planting density is sometimes also called stocking density.) The function *plantedSPH* gives the planted number of trees (or stems) per hectare, which is calculated by dividing 10 000 (the number of square metres in a hectare) by the multiplication of the two distances (output of the function *plantDensity*).

RegimePlanting

RegimePlantAge

RegimePlantDate

$plantDensity : (REGIMEID \times REGIMETYPE) \rightarrow (DISTANCE \times DISTANCE)$

$plantedSPH : (REGIMEID \times REGIMETYPE) \rightarrow SPH$

$\forall regID : REGIMEID \bullet$

$\forall regType : REGIMETYPE \bullet$

$(regID, regType) \in \text{dom } plantDensity \wedge$

$(regID, regType) \in \text{dom } plantedSPH \wedge$

$first(plantDensity (regID, regType)) > 0 \wedge$

$second(plantDensity (regID, regType)) > 0 \wedge$

$(plantedSPH (regID, regType)) > 0 \wedge$

$(plantedSPH (regID, regType)) = 10000 \text{div}$

$(first(plantDensity (regID, regType)) *$

$second(plantDensity (regID, regType)))$

D.2.2.5 Regime's felling details

As with planting the stand, the felling (or harvesting) of the stand is described in terms of the age of the trees in the stand for the planned regime, while for the actual regime, the date is recorded. The schema *RegimeFelling* combines the two schemas *RegimeFellAge* and *RegimeFellDate*.

The schema *RegimeFellAge* specifies the function *fellAge*. *fellAge* maps the regime's ID and regime type to the trees' felling age. This function is used for the planned regime (and is undefined for the actual regime). This age is always greater than zero.

RegimeFellAge

$fellAge : (REGIMEID \times REGIMETYPE) \rightarrow AGE$

$\forall regID : REGIMEID \bullet$

$\forall regType : REGIMETYPE \bullet$

$(regID, regType) \in \text{dom } fellAge \wedge$

$(fellAge (regID, planned)) > 0 \wedge$

$\{(fellAge (regID, actual))\} = \emptyset$

The schema *RegimeFellDate* specifies the function *fellDate*. This function maps the regime's ID and regime type to the trees' felling date (day, month and year). This function is used for the actual regime (and is undefined for the planned regime).

$\text{fellDate} : (\text{REGIMEID} \times \text{REGIMETYPE}) \rightarrow \text{DATE}$
$\begin{aligned} &\forall \text{regID} : \text{REGIMEID} \bullet \\ &\quad \forall \text{regType} : \text{REGIMETYPE} \bullet \\ &\quad (\text{regID}, \text{regType}) \in \text{dom fellDate} \wedge \\ &\quad \{(\text{fellDate}(\text{regID}, \text{planned}))\} = \emptyset \end{aligned}$

The schema *RegimeFelling* is made up of combination of the schemas *RegimeFellAge* and *RegimeFellDate*.

$$\text{RegimeFelling} \hat{=} \text{RegimeFellAge} \wedge \text{RegimeFellDate}$$

D.2.2.6 Tracking which is the current actual regime

After many rotations (crops) on a particular stand, many actual regimes would exist. The most recent one (i.e. the current one for any particular stand) could be determined by inspecting the planting dates of all the actual regimes; the schema *TrackActualRegimes* keeps a pointer to the most recent actual regime.

The schema *DefineTrackActualRegimes* contains two functions: *currentRegime*, which maps the stand's ID to the current actual regime's ID for the stand, and *prevRegime*, which maps the stand's ID to the previous actual regime's ID for the stand. These two functions are undefined if the regime type is planned. *currentRegime* is updated when the stand is planted, while *prevRegime* is updated when the stand is harvested.

The schema *InitTrackActualRegimes* initialises both the functions *currentRegime* and *prevRegime* to be undefined. Schema *TrackActualRegimes* combines the two schemas *DefineTrackActualRegimes* and *InitTrackActualRegimes*.

$\begin{aligned} &\text{currentRegime} : \text{STANDID} \leftrightarrow \text{REGIMEID} \\ &\text{prevRegime} : \text{STANDID} \leftrightarrow \text{REGIMEID} \end{aligned}$
$\begin{aligned} &\forall sID : \text{STANDID} \bullet \\ &\quad \forall \text{regID} : \text{REGIMEID} \bullet \\ &\quad \forall \text{regType} : \text{REGIMETYPE} \bullet \\ &\quad sID \in \text{dom currentRegime} \wedge sID \in \text{dom prevRegime} \wedge \\ &\quad (\text{regType} = \text{planned}) \Rightarrow \\ &\quad \quad \{(currentRegime sID)\} = \emptyset \wedge \\ &\quad \quad \{(prevRegime sID)\} = \emptyset \wedge \\ &\quad (\text{regType} = \text{actual}) \Rightarrow \\ &\quad \quad (\text{regID} \in \text{ran currentRegime} \wedge \\ &\quad \quad \text{regID} \in \text{ran prevRegime}) \end{aligned}$

$\text{InitTrackActualRegimes}$ $\Delta \text{DefineTrackActualRegimes}$
$\text{currentRegime}' = \emptyset$ $\text{prevRegime}' = \emptyset$

$$\text{TrackActualRegimes} \hat{=} \text{DefineTrackActualRegimes} \wedge \text{InitTrackActualRegimes}$$

D.2.2.7 Planned and actual regimes

The planned and actual regimes are defined next. In the schema *RegimeIDs*, the set of regimeIDs is defined and initialised. In the schema *RegimeFunctions*, each regime is defined according to whether it is a planned or an actual regime. The regime couple (planned and actual) for a particular stand is also recorded. The planned and actual regimes are then defined in the schemas *PlannedRegimes* and *ActualRegimes* respectively.

The schema *DefineRegimeIDs* a finite set called *regimeIDs*: it is a set of *REGIMEID* and *REGIMETYPE* pairs. This is used to ensure that on planting, a unique actual regimeID is assigned.

DefineRegimeIDs $\text{regimeIDs} : \mathbb{F} (\text{REGIMEID} \times \text{REGIMETYPE})$
--

InitRegimeIDs $\Delta \text{DefineRegimeIDs}$
$\text{regimeIDs}' = \emptyset$

$$\text{RegimeIDs} \hat{=} \text{DefineRegimeIDs} \wedge \text{InitRegimeIDs}$$

The schema *RegimeFunctions* defines two functions: *standRegimeType* and *standsRegimes*. The function *standRegimeType* maps the stand's ID and regime type (planned or actual) to a regime's ID. The function *standsRegimes* maps the stand's ID to its planned and actual regimes' IDs. In addition to these two functions, it includes schema *RegimeIDs*, which contains the initialised finite set *regimeIDs*.

RegimeFunctions $\text{standRegimeType} : (\text{STANDID} \times \text{REGIMETYPE}) \rightarrow \text{REGIMEID}$ $\text{standsRegimes} : \text{STANDID} \leftrightarrow (\text{REGIMEID} \times \text{REGIMEID})$ RegimeIDs
--

The schema *PlannedRegimes* includes four schemas: *RegimeGeneticMaterial*, *RegimePlanting*, *RegimeFellAgeMinMax* and *RegimeFunctions*. For *planned* regimes, there will be at least one regime for the stand. Since a planned regime is made up of five facets (functions) (the genetic material to be used, the planting age, the planting density, the planted SPH and the felling age), for each planned regime, all five of these must be present.

<p><i>PlannedRegimes</i></p> <p><i>RegimeGeneticMaterial</i></p> <p><i>RegimePlanting</i></p> <p><i>RegimeFellAge</i></p> <p><i>RegimeFunctions</i></p>
$ \begin{aligned} &\forall sID : STANDID \bullet \\ &\quad \exists regID : REGIMEID \bullet \\ &\quad \exists regType : REGIMETYPE \bullet \\ &\quad (regID, regType) \in regimeIDs \wedge \\ &\quad (regID, regType) \in \text{dom } regimeGeneticMaterial \wedge \\ &\quad (regID, regType) \in \text{dom } plantAge \wedge \\ &\quad (regID, regType) \in \text{dom } plantDensity \wedge \\ &\quad (regID, regType) \in \text{dom } plantedSPH \wedge \\ &\quad (regID, regType) \in \text{dom } fellAge \wedge \\ &\quad (sID, regType) \in \text{dom } standRegimeType \wedge \\ &\quad regID \in \text{ran } standRegimeType \wedge \\ &\quad (standRegimeType (sID, regType)) = regID \wedge \\ &\quad sID \in \text{dom } standsRegimes \wedge \\ &\quad regID \in \{first(standsRegimes sID)\} \wedge \\ &\quad (regType = planned) \Rightarrow \\ &\quad \quad (\#\{(standRegimeType (sID, planned))\} \geq 1 \wedge \\ &\quad \quad \#\{(regimeGeneticMaterial ((standRegimeType (sID, planned)), planned))\} = \\ &\quad \quad \#\{(plantAge ((standRegimeType (sID, planned)), planned))\} = \\ &\quad \quad \#\{(plantDensity ((standRegimeType (sID, planned)), planned))\} = \\ &\quad \quad \#\{(plantedSPH ((standRegimeType (sID, planned)), planned))\} = \\ &\quad \quad \#\{(fellAge ((standRegimeType (sID, planned)), planned))\}) \end{aligned} $

The schema *ActualRegimes* includes five schemas: *RegimeGeneticMaterial*, *RegimePlanting*, *RegimeFelling*, *TrackActualRegimes* and *RegimeFunctions*. For *actual* regimes, the two functions *plantAge* and *fellAge* are not used or defined. There is only one current actual regime per stand. In this regime, the planted trees' genetic material ID, planting date, planting density and planted stems per hectare are recorded; the fell date may also be recorded (if the stand has been felled and it's now temporarily unplanted). A new *currentRegime* is assigned on planting the stand (see schema *PlantStand* in section D.4.4).

ActualRegimes

RegimeGeneticMaterial

RegimePlanting

RegimeFellDate

TrackActualRegimes

RegimeFunctions

$$\begin{aligned} & \forall sID : STANDID \bullet \\ & \quad \exists regID : REGIMEID \bullet \\ & \quad \exists regType : REGIMETYPE \bullet \\ & \quad (regID, regType) \in regimeIDs \wedge \\ & \quad (regID, regType) \in \text{dom } regimeGeneticMaterial \wedge \\ & \quad (regID, regType) \in \text{dom } plantDate \wedge \\ & \quad (regID, regType) \in \text{dom } plantDensity \wedge \\ & \quad (regID, regType) \in \text{dom } plantedSPH \wedge \\ & \quad (regID, regType) \in \text{dom } fellDate \wedge \\ & \quad (sID, regType) \in \text{dom } standRegimeType \wedge \\ & \quad regID \in \text{ran } standRegimeType \wedge \\ & \quad (standRegimeType (sID, regType)) = regID \wedge \\ & \quad sID \in \text{dom } standsRegimes \wedge \\ & \quad sID \in \text{dom } currentRegime \wedge \\ & \quad (regType = actual) \Rightarrow \\ & \quad \quad ((currentRegime sID) = (standRegimeType (sID, actual)) \wedge \\ & \quad \quad \#\{(currentRegime sID)\} = 1 \wedge \\ & \quad \quad (\#\{(regimeGeneticMaterial ((standRegimeType (sID, actual)), actual))\} = \\ & \quad \quad \quad \#\{(plantDate ((standRegimeType (sID, actual)), actual))\} = \\ & \quad \quad \quad \#\{(plantDensity ((standRegimeType (sID, actual)), actual))\} = \\ & \quad \quad \quad \#\{(plantedSPH ((standRegimeType (sID, actual)), actual))\} = 1 \wedge \\ & \quad \quad \quad \#\{(fellDate ((standRegimeType (sID, actual)), actual))\} = \emptyset \vee \\ & \quad \quad \quad \#\{(fellDate ((standRegimeType (sID, actual)), actual))\} = 1) \wedge \\ & \quad \quad regID = \text{second}(standsRegimes sID) \wedge \\ & \quad \quad \#\{\text{first}(standsRegimes sID)\} = \#\{\text{second}(standsRegimes sID)\}) \end{aligned}$$

D.2.2.8 Storing the regimes

Information about storing the regimes is given in *StoreRegimes*, which combines the planned and actual regime schemas.

The schema *StoreRegimes* is made up of two schemas: *PlannedRegimes*, which stores the details of the planned regimes for each stand, and *ActualRegimes*, which stores the stand's actual regimes.

$$StoreRegimes \hat{=} PlannedRegimes \wedge ActualRegimes$$

D.3 Forest-to-mill logistics chain

As with the forest management specifications, which have to be declared before they are used (Stepney *et al.*, 2003), the forest-to-mill logistics chain also needs to be defined at the beginning of the specification. Trees are planted in plantations to provide a constant supply of wood to a pulp mill. However, it is counter productive if some stands have trees planted in them which are not desired by any mill.

This part of the specification starts with definitions necessary for this and later sections, and continues with the logistics chain – i.e. how the stands, depots and mills are linked. After the mass of logs (or tree-lengths) has been defined, each of element of the logistics chain is then specified individually.

D.3.1 Preliminary definitions

The definitions needed for specifying the forest-to-mill logistics chain are given first. Each depot and mill is uniquely identified. (Stands were identified in the section on regimes (section D.2.2), as regimes are applied to stands).

[*DEPOTID*, *MILLID*]

The type *MESSAGE* is defined to be able to give outputs in the exception schemas, should a stand need to be planted and it is already planted (see section D.4.4); should the stand be due for harvesting and it is not planted (see section D.4.5); or should there not be sufficient logs in the mill's logyard to remove a load for processing (see section D.6.1).

MESSAGE ::= *StandAlreadyPlanted* | *StandNotPlanted* | *NotEnoughLogs*

D.3.2 Logistics chain

The logistics chain is defined in two parts. First, the stands, depots and mills which are linked by roads are specified in the schema *LogisticsChain*. This outlines all the possible links between a stand and a mill. (This is important, because if a stand is not linked to a mill for which its logs are destined, the logs cannot get to the mill.) Second, the specific mill for which a stand's logs is destined is described in schema *MillForStandsLogs*. Before the stand is harvested, a harvesting decision will be taken and stands' logs allocated to a particular mill.

The schema *LogisticsChain* ensures that each stand's trees has at least one depot to which the logs (or tree-lengths) are to be sent, and each depot has at least one mill to which the logs (or tree-lengths) are to be sent³. Trees are grown in the stand in order to supply a mill. If there is no depot or mill to which that timber can be sent, the trees would not be planted in this plantation (i.e. we are not describing the situation of a farmer who plants a stand of trees in the hope that it could be sold to some wood processor sometime in the future). (In the case of planting up stands in advance of a mill being built, although the mill does not yet exist, the trees are still destined for that mill.) This schema includes two functions, *sendLogsToDepot*, which keeps track of the depot or depots to which the timber from each stand can be sent, and *sendLogsToMill*, which keeps track of the mill or mills to which the timber from the depot can be sent. Each stand must have at least one depot to which its timber could be sent; and each depot must have at least one mill to which its timber could be sent. In addition, there must be at least one link from stand to depot to mill – i.e. each stand's timber must have at least one destination (mill).

LogisticsChain

$sendLogsToDepot : STANDID \leftrightarrow DEPOTID$

$sendLogsToMill : DEPOTID \leftrightarrow MILLID$

$\forall sID : STANDID \bullet$

$\exists dID : DEPOTID \bullet$

$\exists mID : MILLID \bullet$

$sID \in \text{dom } sendLogsToDepot \wedge$

$dID \in \text{ran } sendLogsToDepot \wedge$

$dID \in \text{dom } sendLogsToMill \wedge$

$mID \in \text{ran } sendLogsToMill \wedge$

$(sendLogsToDepot sID) = dID \wedge$

$(sendLogsToMill dID) = mID \wedge$

$(sendLogsToMill (sendLogsToDepot sID)) = mID \wedge$

$\#\{(sendLogsToDepot sID)\} \geq 1 \wedge$

$\#\{(sendLogsToMill dID)\} \geq 1 \wedge$

$\#\{(sendLogsToMill (sendLogsToDepot sID))\} \geq 1$

The schema *MillForStandsLogs* defines the mill to which each stand's timber will be sent. This schema includes the unchangeable schema *LogisticsChain*, and defines a function, *millForStandsLogs*, which has as its input the stand's ID and output the mill's ID. For each stand, a single depot and a single mill exists that are linked logistically via the functions *sendLogsToDepot* and *sendLogsToMill* from schema *LogisticsChain*. The function *millForStandsLogs*, each stand only has only mill to which the timber from the stand can be sent, in which case the function is defined. The function may also be undefined (for example, when the stand is not planted).

³In this specification, the tree-lengths could be cut into logs at roadside, or at the depot or at the mill. Where (and when) the logs are made do not affect the specifications.

MillForStandsLogs

\exists *LogisticsChain*

millForStandsLogs : *STANDID* \leftrightarrow *MILLID*

$\forall sID : STANDID \bullet$

$\exists_1 dID : DEPOTID \bullet$

$\exists_1 mID : MILLID \bullet$

$sID \in \text{dom } \textit{sendLogsToDepot} \wedge$

$dID \in \text{ran } \textit{sendLogsToDepot} \wedge$

$dID \in \text{dom } \textit{sendLogsToMill} \wedge$

$mID \in \text{ran } \textit{sendLogsToMill} \wedge$

$sID \in \text{dom } \textit{millForStandsLogs} \wedge$

$mID \in \text{ran } \textit{millForStandsLogs} \wedge$

$(\textit{sendLogsToDepot } sID) = dID \wedge$

$(\textit{sendLogsToMill } dID) = mID \wedge$

$(\textit{sendLogsToMill } (\textit{sendLogsToDepot } sID)) = mID \wedge$

$(\textit{millForStandsLogs } sID) = mID \wedge$

$\#\{(\textit{sendLogsToDepot } sID)\} \geq 1 \wedge$

$\#\{(\textit{sendLogsToMill } dID)\} \geq 1 \wedge$

$\#\{(\textit{sendLogsToMill } (\textit{sendLogsToDepot } sID))\} \geq 1 \wedge$

$(\{\textit{millForStandsLogs } sID\} = \emptyset \vee \#\{\textit{millForStandsLogs } sID\} = 1)$

D.3.3 Log mass in the logistics chain

Logs (or tree-lengths) are transported along the logistics chain between the forest stand and the mill's process. In order to model this flow, their mass must be defined.

MASS is defined to determine the mass (tonnage) of the logs. Although this number would normally be a real number, it is defined here as a natural number (because of a restriction in the Z type checker used).

| *MASS* : \mathbb{FN}

D.3.4 At the stand's roadside

When a stand is harvested, the tree-lengths are extracted to the road which is adjacent to the stand. Often the tree-lengths are cut into logs at roadside and are piled up, waiting for the short-haul trucks to take them to the depot. However, this is not always the case. The tree-lengths could also be cut into logs at the depot, or at the mill; this specification is valid for any of the three. If logs are made later in the process, for 'log' read 'tree-length'. The schema *LogsAtRoadside* contains a function, *logsMassAtRoadside*, which keeps track of the mass of logs at the stand's roadside. Logs are added to the roadside pile in schema *HarvestStandAndUpdateRegime* in

section D.4.5. Logs are removed from the pile in schema *LoadLogsAtStand* in section D.5.4.2.

The schema *DefineLogsAtRoadside* describes the logs which have been cut from the trees in the stand and have been extracted to roadside and piled there. It includes two unchanged schemas: *LogisticsChain* and *MillForStandsLogs*, and one function: *logsMassAtRoadside*, which maps the stand's ID and the ID of the mill for which this timber is intended to the mass of the timber.

The mass of logs at roadside will be greater than or equal to zero. The stand is linked to the destination mill logistically via the two functions from the schema *LogisticsChain*, *sendLogsToDepot* and *sendLogsToMill*. It is assumed that the logs will be transported from a stand's roadside pile before the stand produces another crop of trees.

<p><i>DefineLogsAtRoadside</i></p> <p>\exists<i>LogisticsChain</i></p> <p>\exists<i>MillForStandsLogs</i></p> <p><i>logsMassAtRoadside</i> : (<i>STANDID</i> \times <i>MILLID</i>) \leftrightarrow <i>MASS</i></p> <hr/> <p>$\forall sID : STANDID \bullet \exists dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$ $(sID, mID) \in \text{dom } logsMassAtRoadside \wedge$ $sID \in \text{dom } millForStandsLogs \wedge$ $mID \in \text{ran } millForStandsLogs \wedge$ $sID \in \text{dom } sendLogsToDepot \wedge$ $dID \in \text{ran } sendLogsToDepot \wedge$ $dID \in \text{dom } sendLogsToMill \wedge$ $mID \in \text{ran } sendLogsToMill \wedge$ $(sendLogsToMill (sendLogsToDepot sID)) = mID \wedge$ $logsMassAtRoadside (sID, mID) \geq 0$</p>

The schema *InitLogsAtRoadside* includes changeable schema *DefineLogsAtRoadside* and initialises all values of *logsMassAtRoadside* to the empty set. Schema *NoLogsAtRoadside* then sets all values of *logsMassAtRoadside* to zero. Schema *LogsAtRoadside* combines the three schemas *DefineLogsAtRoadside*, *InitLogsAtRoadside* and *NoLogsAtRoadside*.

<p><i>InitLogsAtRoadside</i></p> <p>Δ<i>DefineLogsAtRoadside</i></p> <p><i>logsMassAtRoadside'</i> = \emptyset</p>

$\frac{\text{NoLogsAtRoadside}}{\Delta \text{InitLogsAtRoadside}}$
$\forall sID : STANDID \bullet \exists_1 mID : MILLID \bullet$ $(sID, mID) \in \text{dom } \text{logsMassAtRoadside} \wedge$ $\text{logsMassAtRoadside}' = \text{logsMassAtRoadside} \cup \{(sID, mID), 0\}$

$$\text{LogsAtRoadside} \hat{=} \text{DefineLogsAtRoadside} \wedge \text{InitLogsAtRoadside} \wedge \text{NoLogsAtRoadside}$$

D.3.5 Depot

Depots are situated near the forest stands – usually there is one depot per estate (a group of forest stands). When the stands from that estate are harvested, the logs (or tree-lengths) will be taken to the depot (via short-haul transport), where they are put into piles according to the mill which is their final destination. From there, the logs (or tree-lengths) are taken to the mill via long-haul transport. The schema *Depot* includes the function *logsMassAtDepot*, which governs the mass of logs at the depot. Logs are added to the depot piles in schema *TransportAndUnloadLogsAtDepot* (section D.5.4.3), and are removed from the piles in schema *LoadLogsAtDepot* (section D.5.5.2).

$\frac{\text{DefineDepot}}{\exists \text{LogisticsChain}}$ $\exists \text{MillForStandsLogs}$ $\text{logsMassAtDepot} : (\text{DEPOTID} \times \text{MILLID}) \leftrightarrow \text{MASS}$
$\forall sID : STANDID \bullet$ $\quad \exists dID : DEPOTID \bullet$ $\quad \exists mID : MILLID \bullet$ $\quad sID \in \text{dom } \text{millForStandsLogs} \wedge$ $\quad mID \in \text{ran } \text{millForStandsLogs} \wedge$ $\quad sID \in \text{dom } \text{sendLogsToDepot} \wedge$ $\quad dID \in \text{ran } \text{sendLogsToDepot} \wedge$ $\quad (dID, mID) \in \text{dom } \text{logsMassAtDepot} \wedge$ $\quad dID \in \text{dom } \text{sendLogsToMill} \wedge$ $\quad mID \in \text{ran } \text{sendLogsToMill} \wedge$ $\quad (\text{sendLogsToMill} (\text{sendLogsToDepot } sID)) = mID \wedge$ $\quad \text{logsMassAtDepot} (dID, mID) \geq 0$

The schema *DefineDepot* describes the depot to which the logs (or tree-lengths) will be transported from roadside. It includes the unchanged schemas *LogisticsChain* and *MillForStandsLogs*, and the function *logsMassAtDepot*. This function records the logs which are at the depot, which are piled according to the mill to which the logs will be sent. The output of the function is the mass of logs in each depot's mill pile, which is always greater than, or equal to, zero. The depot is linked

logistically to the stands that send it timber, and the mill(s) to which it sends the timber.

The schema *InitDepot* initialises the mass of the contents of the depot pile destined for a particular mill to the empty set. Schema *NoLogsAtDepot* the initialises the mass of logs at the depot destined for a mill to zero. The schema *Depot* combines the three schemas *DefineDepot*, *InitDepot* and *NoLogsAtDepot*.

$\frac{\textit{InitDepot}}{\Delta \textit{DefineDepot}}$
$\textit{logsMassAtDepot}' = \emptyset$

$\frac{\textit{NoLogsAtDepot}}{\Delta \textit{InitDepot}}$
$\forall dID : \textit{DEPOTID} \bullet \exists mID : \textit{MILLID} \bullet$ $(dID, mID) \in \textit{dom logsMassAtDepot} \wedge$ $\textit{logsMassAtDepot}' = \textit{logsMassAtDepot} \cup \{((dID, mID), 0)\}$

$$\textit{Depot} \hat{=} \textit{DefineDepot} \wedge \textit{InitDepot} \wedge \textit{NoLogsAtDepot}$$

D.3.6 Mill

The specification of the mill involves two aspects: the genetic material acceptable at the mill, and the mass of logs in the logyard. Mills accept some trees' genetic material, while other tree species are not desired. These relationships are captured in schema *MillAcceptableSpecies*. The mass of the logs at the mill is specified in the function *logyard*, which is contained in the schema *Mill*. Logs are added to the logyard in two independent schemas: *TransportAndUnloadLogsAtMill* in section D.5.5.3 and *AcceptLogsFromOtherSuppliers* in section D.5.7. Logs are removed from the logyard by processing (see section D.6.1).

The 'species' (genetic material) acceptable to the mill is described in schema *MillAcceptableSpecies*. This schema includes the unchanged schema *WoodGeneticMaterial*, and a function, *acceptableSpecies*. It has as input the mill's ID, and gives the set of species (i.e. the genus, or the genus and species (pure or hybrid) of the wood entering the logyard) which are acceptable to the mill. Acceptable 'species' are specified as either genera, genus-species combinations, or genus-species-species combinations. For example, the mill could accept any species of *Eucalyptus*, in which case the genus *Eucalyptus* will be specified. Alternatively, specific genus-species (pure species - e.g. *Eucalyptus grandis*) or genus-species-species (hybrid, e.g. *Eucalyptus grandis* x *nitens*) combinations could be specified.

MillAcceptableSpecies

\exists *WoodGeneticMaterial*

acceptableSpecies : *MILLID* \rightarrow *GENMATERIALID*

\forall *mID* : *MILLID* •

\exists *genMatID1* : *GENMATERIALID* •

\exists *genMatID2* : *GENMATERIALID* •

mID \in *dom acceptableSpecies* \wedge

genMatID1 \in *ran acceptableSpecies* \wedge

(*genMatID1* \in *dom genusWood* \vee

genMatID1 \in *dom pureSpeciesWood* \vee

genMatID1 \in *dom hybridWood*) \wedge

genMatID2 \in *ran acceptableSpecies* \wedge

(*genMatID2* \in *dom pureSpeciesWood* \vee

genMatID2 \in *dom hybridWood*) \wedge

(*genMatID1* = *genMatID2*) \Rightarrow

((*pureSpeciesWood genMatID1*) = (*pureSpeciesWood genMatID2*) \vee

(*hybridWood genMatID1*) = (*hybridWood genMatID2*)) \wedge

(*genMatID1* \neq *genMatID2*) \Rightarrow

(\exists_1 *genID* : *GENUSID* •

\exists *sppID1* : *SPECIESID* •

\exists *sppID2* : *SPECIESID* •

genID \in *ran genusWood* \wedge

((*genID*, *sppID1*) \in *ran pureSpeciesWood* \wedge

(*genID*, *sppID1*, *sppID2*) \in *ran hybridWood*) \wedge

(*genusWood genMatID2*) = *genID* \wedge

(*first(pureSpeciesWood genMatID1)* = *genID* \vee

(*hybridWood genMatID1*) = (*genID*, *sppID1*, *sppID2*)))

The mill is described in schema *DefineMill*. This schema includes the unchanged schemas *LogisticsChain*, *MillForStandsLogs* and *MillAcceptableSpecies*. It specifies a function, *logyard*, which takes as input the mill's ID, and maps this to the mass of logs in the logyard. The mass of logs in the logyard is greater than or equal to zero. (One could specify that the mass must exceed a certain mass to ensure that there are always enough logs in the logyard to cover the eventualities of the transport system failing, or stands not being accessible in adverse weather). The mill is linked logistically (via the functions in schema *LogisticsChain*) to at least one depot, which is in turn linked logistically to at least one stand, which grows the timber to be sent to the mill.

The schema *InitMill* initialises the function *logyard* to empty set, and schema *NoLogsAtMill* sets the logyard's mass to zero. The schema *Mill* combines the three schemas *DefineMill*, *InitMill* and *NoLogsAtMill*.

DefineMill

\exists LogisticsChain
 \exists MillForStandsLogs
 \exists MillAcceptableSpecies
 $logyard : MILLID \rightarrow MASS$

$\forall mID : MILLID \bullet$
 $\exists sID : STANDID \bullet$
 $\exists dID : DEPOTID \bullet$
 $\exists genMatID : GENMATERIALID \bullet$
 $sID \in \text{dom } sendLogsToDepot \wedge$
 $dID \in \text{ran } sendLogsToDepot \wedge$
 $dID \in \text{dom } sendLogsToMill \wedge$
 $mID \in \text{ran } sendLogsToMill \wedge$
 $sID \in \text{dom } millForStandsLogs \wedge$
 $mID \in \text{ran } millForStandsLogs \wedge$
 $mID \in \text{dom } logyard \wedge$
 $mID \in \text{dom } acceptableSpecies \wedge$
 $genMatID \in \text{ran } acceptableSpecies \wedge$
 $(sendLogsToMill (sendLogsToDepot sID)) = mID \wedge$
 $(millForStandsLogs sID) = mID \wedge$
 $(logyard mID) \geq 0$

InitMill

Δ DefineMill

$logyard' = \emptyset$

NoLogsAtMill

Δ InitMill

$\forall mID : MILLID \bullet$
 $mID \in \text{dom } logyard \wedge$
 $logyard' = logyard \cup \{(mID, 0)\}$

$Mill \hat{=} DefineMill \wedge InitMill \wedge NoLogsAtMill$

D.4 Plantation forestry

In plantation forestry, the plantations are managed in a hierarchy. The top level is the forestry company. As discussed in section 4.3.2, there are several levels under this (the number depending on the company). The second lowest level is the estate, which is a group of stands which are managed together, and at the lowest level is the stand, which is the smallest homogeneous growing unit is the stand (or compartment). In this

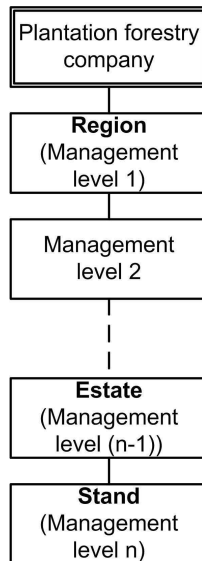


Figure D.8: Hierarchy of forestry company management levels

specification, the forest management hierarchy is described first (see section D.4.1). Thereafter, aspects relating to the stand are broken up into the following sections: section D.4.2 describes features and characteristics of the stand's land. Section D.4.3 describes the features of the stand, together with the trees planted on it. Section D.4.4 describes the planting of the stand, and section D.4.5 describes the harvesting of the stand.

D.4.1 Forest management hierarchy

The hierarchy described in this specification is a simplified version of that described in section 4.3.2: although there could be several management levels between the company level and the estate (see Figure D.8), only the company, estate and stand are modelled (see Figure D.9).

The stand's ID (*STANDID*) has already been defined (see section D.2.2.1). Here the *COMPANYID* and *ESTATEID* are defined.

[*COMPANYID*, *ESTATEID*]

The schema *CompanyHasEstates* contains a function, *companyHasEstates*, which maps the companyID to the estates which are included in the company. There is only one plantation forestry company specified, and this company has one or more estates.

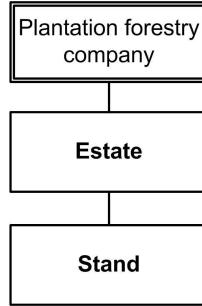
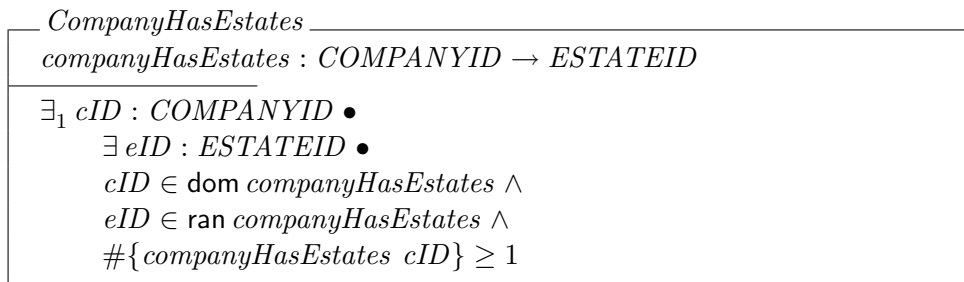
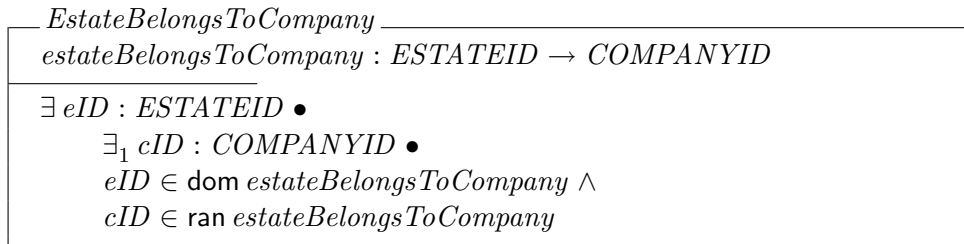


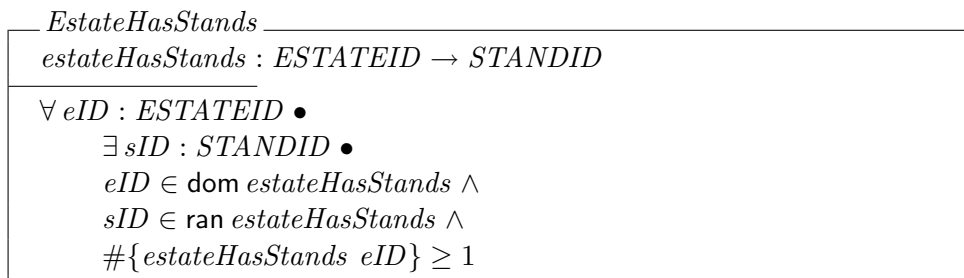
Figure D.9: Hierarchy of forestry company management levels modelled in the Z specification



The schema *EstateBelongsToCompany* includes a function, *estateBelongsToCompany*, which maps the estateID to the companyID.



The schema *EstateHasStands* contains a function, *estateHasStands*, which relates an estateID to the IDs of the stands it contains. An estate is made up of one or more stands.



The schema *StandBelongsToEstate* contains a function, *standBelongsToEstate*, which maps the stand's ID to the estate's ID. Each stand may only belong to one estate.

<i>StandBelongsToEstate</i>
<i>standBelongsToEstate</i> : <i>STANDID</i> → <i>ESTATEID</i>
$\forall sID : STANDID \bullet$ $\quad \exists_1 eID : ESTATEID \bullet$ $\quad sID \in \text{dom } standBelongsToEstate \wedge$ $\quad eID \in \text{ran } standBelongsToEstate$

D.4.2 Stand's land

This section describes the aspects of the stand which are independent of the trees which are planted on them. Certain species are suitable for planting on the stand, because of the soil type, the prevailing weather conditions, the stand's altitude etc. 'Site-species' matching will be undertaken to determine the most suitable species for planting, and the schema *StandSuitableSpecies* stores this list and the associated constraints. Other things which are independent of the trees to be grown on the stand are characteristics like the stand's area and altitude; these are stored in the schema *StandCharacteristics*. The schema *StandLand* is made up of these two schemas.

AREA is defined to store the area of each stand, and *HEIGHT* is defined to store the the height of the stand above sea level (altitude), or the site index (which is the average dominant height of the trees in the stand). Although these numbers would normally be a real number, they is defined here as a natural number (because of a restriction in the Z type checker used).

| *AREA, HEIGHT* : \mathbb{N}

The species suitable for a particular stand's land is described in the *StandSuitableSpecies* schema. This schema includes the unchanged schemas *WoodGeneticMaterial*, *StoreRegimes*, *LogisticsChain* and *MillAcceptableSpecies*. Each stand has a list of suitable 'species' which could be grown there (determined by site-species matching). This is denoted here by the function *suitableSpecies*, which has as its output a genetic material ID. This is either mapped to a genus-species combination (i.e. describing a pure species tree) or a genus-species-species combination (i.e. describing a hybrid tree). There must be at least one species in the list of suitable species for each stand. The schema captures the fact that if a stand has a suitable species, then it should also have a planned regime for all of those suitable species. These species could be genus-species combinations (pure species) or genus-species-species combinations (hybrids). It is also possible to have a planned regime for trees at the genus level (e.g. the regime would apply to all

Pinus trees) – which would cover the case where trees of a suitable ‘species’ have that genus. In addition, the ‘species’ (genetic material) for which the planned regime exists should be acceptable to a mill, and there should be a logistical link between the stand the the mill via the depot. Each stand must have at least one planned regime.

StandSuitableSpecies

\exists *WoodGeneticMaterial*

\exists *StoreRegimes*

\exists *LogisticsChain*

\exists *MillAcceptableSpecies*

suitableSpecies : *STANDID* \rightarrow *GENMATERIALID*

\forall *sID* : *STANDID* •

\forall *genMatID1* : *GENMATERIALID* • \forall *genMatID2* : *GENMATERIALID* •

\exists *regID* : *REGIMEID* •

\exists *regType* : *REGIMETYPE* •

sID \in *dom suitableSpecies* \wedge

genMatID1 \in *ran suitableSpecies* \wedge

(*genMatID1* \in *dom pureSpeciesWood* \vee *genMatID1* \in *dom hybridWood*) \wedge

$\#\{(suitableSpecies\ sID)\} \geq 1$ \wedge

(*sID*, *regType*) \in *dom standRegimeType* \wedge

regID \in *ran standRegimeType* \wedge

regType = *planned* \wedge

(*regID*, *regType*) \in *regimeIDs* \wedge

genMatID2 \in *ran suitableSpecies* \wedge

genMatID2 \in *ran acceptableSpecies* \wedge

(*genMatID2* \in *dom genusWood* \vee

genMatID2 \in *dom pureSpeciesWood* \vee

genMatID2 \in *dom hybridWood*) \wedge

(*genMatID1* = *genMatID2*) \Rightarrow

((*pureSpeciesWood* *genMatID1*) = (*pureSpeciesWood* *genMatID2*) \vee

(*hybridWood* *genMatID1*) = (*hybridWood* *genMatID2*)) \wedge

(*genMatID1* \neq *genMatID2*) \Rightarrow

(\exists_1 *genID* : *GENUSID* •

\exists *sppID1* : *SPECIESID* • \exists *sppID2* : *SPECIESID* •

genID \in *ran genusWood* \wedge

((*genID*, *sppID1*) \in *ran pureSpeciesWood* \wedge

(*genID*, *sppID1*, *sppID2*) \in *ran hybridWood*) \wedge

(*genusWood* *genMatID2*) = *genID* \wedge

(*first*(*pureSpeciesWood* *genMatID1*) = *genID* \vee

(*hybridWood* *genMatID1*) = (*genID*, *sppID1*, *sppID2*))) \wedge

standRegimeType (*sID*, *regType*) = *regID* \wedge

(*regID*, *regType*) \in *dom regimeGeneticMaterial* \wedge

(*regimeGeneticMaterial* (*regID*, *regType*)) = *genMatID2* \wedge

genMatID2 \in {(*acceptableSpecies* (*sendLogsToMill* (*sendLogsToDepot* *sID*)))} \wedge

$\#\{(standRegimeType\ (sID, regType))\} \geq 1$

Further characteristics of the stand's land are described in the schema *StandCharacteristics*. The schema contains two functions, *standArea*, which records the area of each stand, and *standAltitude*, which records the stand's average height above sea level. The area is always greater than zero and the altitude is always greater than or equal to zero.

<p><i>StandCharacteristics</i></p> <p>$standArea : STANDID \rightarrow AREA$ $standAltitude : STANDID \rightarrow HEIGHT$</p> <hr/> <p>$\forall sID : STANDID \bullet$ $sID \in \text{dom } standArea \wedge$ $sID \in \text{dom } standAltitude \wedge$ $standArea\ sID > 0 \wedge$ $standAltitude\ sID \geq 0$</p>

The stand's land (i.e. features of the stand which are not dependent on having trees planted on it) is described in the *StandLand* schema, which combines the two schemas *StandSuitableSpecies* and *StandCharacteristics*.

$$StandLand \hat{=} StandSuitableSpecies \wedge StandCharacteristics$$

D.4.3 Stand of trees

This section covers the aspects of the stand which describes the interaction of the trees on the stand's land. The stand is either planted or unplanted; this is recorded in schema *StandPlantingStatus*. The schema *StandOfTrees* describes the trees' genetic material and regimes applied to the stand.

The stand's planting state (defined by type *PLANTINGSTATE*) is either *unplanted* or *planted*.

$$PLANTINGSTATE ::= unplanted \mid planted$$

The schema *DefineStandPlantingStatus* contains a function, *plantingStatus*, which maps the stand's ID to the stand's planting status (planted or unplanted). This function is always defined with a single value.

The schema *InitStandPlantingStatus* initialises the function *plantingStatus* to be the empty set. Schema *StandPlantingStatusUnplanted* sets the value of the function *plantingStatus* to *unplanted* for each stand. The schema *StandPlantingStatus* combines the schemas *DefineStandPlantingStatus*, *InitStandPlantingStatus* and

StandPlantingStatusUnplanted.

<i>DefineStandPlantingStatus</i>
$plantingStatus : STANDID \rightarrow PLANTINGSTATE$
$\forall sID : STANDID \bullet$ $sID \in \text{dom } plantingStatus \wedge$ $\#\{(plantingStatus \ sID)\} = 1$

<i>InitStandPlantingStatus</i>
$\Delta DefineStandPlantingStatus$
$plantingStatus' = \emptyset$

<i>StandPlantingStatusUnplanted</i>
$\Delta InitStandPlantingStatus$
$\forall sID : STANDID \bullet$ $sID \in \text{dom } plantingStatus \wedge$ $plantingStatus' = plantingStatus \cup \{(sID, unplanted)\}$

$$StandPlantingStatus \hat{=} DefineStandPlantingStatus \wedge \\ InitStandPlantingStatus \wedge StandPlantingStatusUnplanted$$

The type *VOLUME* is defined so that the stand's log volume can be predicted. Although it is declared as a natural number, it should actually be declared as a real number.

$$| \quad VOLUME : \mathbb{FN}$$

In order to define the site index (a measure of the growth rate) for a stand of trees, the site index's evaluation age must be defined. This is achieved in schema *SiteIndexEvalAge*. This schema includes a function, *evalAge*, which maps the stand's genusID to an age. In other words, for pulp timber stands, there is a different site index evaluation age for different genera (e.g. *Eucalyptus*, *Pinus*, *Acacia*, ...). The evaluation age for the site index (written *SI_EvalAge*) is always greater than 0.

<i>SiteIndexEvalAge</i>
$evalAge : GENUSID \rightarrow AGE$
$\forall genID : GENUSID \bullet$ $\exists age : AGE \bullet$ $genID \in \text{dom } evalAge \wedge$ $age \in \text{ran } evalAge \wedge$ $age > 0$

The schema *StandSiteIndex* includes the unchangeable schemas *SiteIndexEvalAge* and *StandPlantingStatus*, and contains a function, *siteIndex*, which maps the stand's ID and the SI's evaluation (base) age to the site index of the trees planted in the stand. This Site Index is actually the average dominant height of the stand at the evaluation age. If the stand of trees is planted, the SI will be greater than zero. If the stand is unplanted, the SI will not be defined.

<p><i>StandSiteIndex</i></p> <p>$\exists SiteIndexEvalAge$</p> <p>$\exists StandPlantingStatus$</p> <p>$siteIndex : (STANDID \times AGE) \leftrightarrow HEIGHT$</p> <hr/> <p>$\forall sID : STANDID \bullet$</p> <p> $\forall genID : GENUSID \bullet$</p> <p> $\exists age : AGE \bullet$</p> <p> $genID \in \text{dom } evalAge \wedge$</p> <p> $age \in \text{ran } evalAge \wedge$</p> <p> $(evalAge \ genID) = age \wedge$</p> <p> $(sID, age) \in \text{dom } siteIndex \wedge$</p> <p> $(plantingStatus \ sID = \text{planted}) \Rightarrow$</p> <p> $(siteIndex \ (sID, age)) > 0 \wedge$</p> <p> $(plantingStatus \ sID = \text{unplanted}) \Rightarrow$</p> <p> $\{(siteIndex \ (sID, age))\} = \emptyset$</p>
--

Schema *DefineStandVolumeAndMass* calculates the stand's volume and mass at harvesting. It includes the functions *standVolumeAtHarvesting* and *standMassAtHarvesting*, which both have as inputs the stand's ID and the date of harvesting, and outputs the stand's volume and mass respectively. The volume and mass are always greater than or equal to zero, and if the stand's volume is greater than zero, this means that the mass of logs from that stand is also greater than zero.

Schema *InitStandVolumeAndMass* initialises the two functions *standVolumeAtHarvesting* and *standMassAtHarvesting* to empty set and schema *SetStandVolumeAndMassToZero* sets the value of these two functions to zero. Schema *StandVolumeAndMass* combines the three schemas *DefineStandVolumeAndMass*, *InitStandVolumeAndMass* and *SetStandVolumeAndMassToZero*.

DefineStandVolumeAndMass

$standVolumeAtHarvesting : STANDID \times DATE \rightarrow VOLUME$
 $standMassAtHarvesting : STANDID \times DATE \rightarrow MASS$

$\forall sID : STANDID \bullet$
 $\exists date : DATE \bullet$
 $(sID, date) \in \text{dom } standVolumeAtHarvesting \wedge$
 $(sID, date) \in \text{dom } standMassAtHarvesting \wedge$
 $standVolumeAtHarvesting(sID, date) \geq 0 \wedge$
 $standMassAtHarvesting(sID, date) \geq 0 \wedge$
 $standVolumeAtHarvesting(sID, date) > 0 \Rightarrow$
 $standMassAtHarvesting(sID, date) > 0$

InitStandVolumeAndMass

$\Delta DefineStandVolumeAndMass$

$standVolumeAtHarvesting' = \emptyset$
 $standMassAtHarvesting' = \emptyset$

SetStandVolumeAndMassToZero

$\Delta InitStandVolumeAndMass$

$\forall sID : STANDID \bullet$
 $\exists date : DATE \bullet$
 $(sID, date) \in \text{dom } standVolumeAtHarvesting \wedge$
 $(sID, date) \in \text{dom } standMassAtHarvesting \wedge$
 $standVolumeAtHarvesting' = standVolumeAtHarvesting \cup \{(sID, date), 0\} \wedge$
 $standMassAtHarvesting' = standMassAtHarvesting \cup \{(sID, date), 0\}$

$StandVolumeAndMass \hat{=} DefineStandVolumeAndMass \wedge$
 $InitStandVolumeAndMass \wedge SetStandVolumeAndMassToZero$

The schema *StandOfTrees* governs the relationships the stand's land and the trees that are planted on it. It includes the five unchangeable schemas *WoodGeneticMaterial*, *StoreRegimes*, *StandLand*, *StandPlantingStatus* and *MillForStandsLogs*.

If the stand is planted, there is a single current actual regime and there may be a single previous actual regime. There is a single planned regime which is being used to guide the activities taking place in the stand. The function *standRegimeType* stores the regime's ID for a given stand and regime type (planned/actual). The function *standsRegimes* outputs both the planned and actual regime's IDs given the input of the stand's ID. The genetic material ID for the planned and actual regimes either matches, or (in the case that the planned regime is defined at the genus-level only) the genus of the planned regime is the same as the genus for the actual regime.

StandOfTrees

\exists *WoodGeneticMaterial*
 \exists *StoreRegimes*
 \exists *StandLand*
 \exists *StandPlantingStatus*
 \exists *MillForStandsLogs*

\forall *sID* : *STANDID* •

\forall *genMatID* : *GENMATERIALID* •

\exists_1 *pRegimeID* : *REGIMEID* • \exists_1 *aRegimeID* : *REGIMEID* •

\exists_1 *genID* : *GENUSID* •

\exists *sppID1* : *SPECIESID* • \exists *sppID2* : *SPECIESID* •

$(sID, planned) \in \text{dom } standRegimeType \wedge (sID, actual) \in \text{dom } standRegimeType \wedge$

$sID \in \text{dom } currentRegime \wedge sID \in \text{dom } prevRegime \wedge$

$aRegimeID \in \text{ran } currentRegime \wedge$

$sID \in \text{dom } suitableSpecies \wedge$

$sID \in \text{dom } plantingStatus \wedge$

$sID \in \text{dom } millForStandsLogs \wedge$

$(pRegimeID, planned) \in \text{dom } regimeGeneticMaterial \wedge$

$(pRegimeID, planned) \in \text{regimeIDs} \wedge$

$(aRegimeID, actual) \in \text{dom } regimeGeneticMaterial$

$\wedge (aRegimeID, actual) \in \text{regimeIDs} \wedge$

$genMatID \in \text{ran } regimeGeneticMaterial \wedge$

$genID \in \text{ran } genusWood \wedge (genID, sppID1, sppID2) \in \text{ran } hybridWood \wedge$

$pRegimeID \in \text{ran } standRegimeType \wedge aRegimeID \in \text{ran } standRegimeType \wedge$

$((plantingStatus\ sID) = planted) \Rightarrow$

$\#\{(currentRegime\ sID)\} = 1 \wedge$

$(currentRegime\ sID) = aRegimeID = (standRegimeType\ (sID, actual)) \wedge$

$\{(\text{prevRegime}\ sID)\} = \emptyset \vee \#\{(\text{prevRegime}\ sID)\} = 1 \wedge$

$(standRegimeType\ (sID, planned)) = pRegimeID = \text{first}(\text{standsRegimes}\ sID) \wedge$

$(standRegimeType\ (sID, actual)) = aRegimeID = \text{second}(\text{standsRegimes}\ sID) \wedge$

$((regimeGeneticMaterial\ (pRegimeID, planned)) =$

$regimeGeneticMaterial\ (aRegimeID, actual)) \vee$

$(genusWood\ (regimeGeneticMaterial\ (pRegimeID, planned))) =$

$\text{first}(\text{pureSpeciesWood}\ (regimeGeneticMaterial\ (aRegimeID, actual)))) \vee$

$(genID = genusWood\ (regimeGeneticMaterial\ (pRegimeID, planned))) \wedge$

$(genID, sppID1, sppID2) =$

$hybridWood\ (regimeGeneticMaterial\ (aRegimeID, actual)))) \wedge$

$((genMatID \in \text{dom } pureSpeciesWood \vee genMatID \in \text{dom } hybridWood)) \wedge$

$\#\{(regimeGeneticMaterial\ (aRegimeID, actual))\} = 1 \wedge$

$\#\{\text{first}(\text{standsRegimes}\ sID)\} = \#\{\text{second}(\text{standsRegimes}\ sID)\} \wedge$

$\#\{(\text{sendLogsToDepot}\ sID)\} \geq 1 \wedge$

$\#\{(\text{millForStandsLogs}\ sID)\} = 1 \wedge$

$((plantingStatus\ sID) = unplanted) \Rightarrow$

$(\#\{(\text{prevRegime}\ sID)\} = 1 \vee \{(\text{prevRegime}\ sID)\} = \emptyset) \wedge$

$\#\{(\text{sendLogsToDepot}\ sID)\} \geq 0 \wedge$

$\{(\text{millForStandsLogs}\ sID)\} = \emptyset$

Trees of only one genetic material ('species') are planted in the stand. There are an equal number of planned and actual regimes for the stand (current and past). There must be at least one depot to which the logs can be sent. There is only one mill to which timber from this stand can be sent.

If the stand is unplanted, there will either be a single previous regime (which is assigned at harvesting – see schema *HarvestStandAndUpdateRegime* in section D.4.5), or, in the case of a greenfields plantation project, no previous regime will exist. If the stand is unplanted, the depot to which its logs will eventually be sent (once planted) may be unassigned (Evans and Turnbull, 2004) – i.e. there are zero or more depots to which the logs from this stand could be sent. Likewise, the mill to which timber from this stand should be sent may be unassigned.

D.4.4 Planting the stand

The planting of the stand has been broken up into two schemas. In the first, *DeterminePlantingRegime*, given the inputs of the stand's ID and the ID of the genetic material to be planted, the planned regime's ID is determined. In the second part, the trees are planted, and the actual regime is updated. This is undertaken in schema *PlantStandAndRecordRegime*.

The schema *DeterminePlantingRegime* describes the first part of planting a stand. It takes as input the stand's ID (*whichStand?*) and the 'species' (genetic material) ID (*whichSpecies?*) to be planted. It includes the changeable schemas *RegimeGeneticMaterial*, *TrackActualRegimes* and *StoreRegimes*, and the unchangeable schema *StandLand*. The species of tree to be planted must be in the list of suitable species for that stand (function *suitableSpecies*). Since each stand has a planned regime for the suitable species, this schema finds out which planned regime will be used for the stand, and then assigns the appropriate genetic material ID to the actual regime's genetic material (function *regimeGeneticMaterial*). Finding the appropriate planned regime may be straightforward, if *whichSpecies?* is the ID of a pure species' or hybrid's wood. If, however, the appropriate planned regime only specifies a genus, this has to be found by using the functions *genusOfPureSpeciesWood* or *genusOfHybridWood*.

Before an actual regime's ID is assigned, the regime's ID cannot be in the set of existing regime's IDs (*regimeIDs*). Once the actual regime's ID (*aRegimeID*) has been assigned, it is added to the set of existing regimeIDs. The current actual regime for the stand is updated to contain *aRegimeID*, while the function *standRegimeType* for the stand is updated with the planned and actual regime's IDs respectively. Likewise, the function *standsRegimes* for the stand is updated with the value (*pRegimeID*, *aRegimeID*). The actual regime's genetic material is updated with the ID of the genetic material to be planted (*whichSpecies?*). The function *prevRegime*

is unchanged by this schema.

<p><i>DeterminePlantingRegime</i></p> <p>Δ <i>RegimeGeneticMaterial</i></p> <p>Δ <i>TrackActualRegimes</i></p> <p>Δ <i>StoreRegimes</i></p> <p>Ξ <i>StandLand</i></p> <p><i>whichStand?</i> : <i>STANDID</i></p> <p><i>whichSpecies?</i> : <i>GENMATERIALID</i></p> <hr/> <p><i>whichStand?</i> \in dom <i>currentRegime</i></p> <p><i>whichStand?</i> \in dom <i>standsRegimes</i></p> <p><i>whichStand?</i> \in dom <i>suitableSpecies</i></p> <p><i>whichSpecies?</i> \in ran <i>suitableSpecies</i></p> <p>(<i>whichStand?</i>, <i>planned</i>) \in dom <i>standRegimeType</i></p> <p>(<i>whichStand?</i>, <i>actual</i>) \in dom <i>standRegimeType</i></p> <p>\exists <i>pRegimeID</i> : <i>REGIMEID</i> • \exists <i>aRegimeID</i> : <i>REGIMEID</i> •</p> <p> \exists <i>genID</i> : <i>GENUSID</i> •</p> <p> <i>aRegimeID</i> \in ran <i>currentRegime</i> \wedge</p> <p> <i>pRegimeID</i> \in ran <i>standRegimeType</i> \wedge</p> <p> <i>aRegimeID</i> \in ran <i>standRegimeType</i> \wedge</p> <p> (<i>pRegimeID</i>, <i>aRegimeID</i>) \in ran <i>standsRegimes</i> \wedge</p> <p> (<i>aRegimeID</i>, <i>actual</i>) \notin <i>regimeIDs</i> \wedge</p> <p> (<i>pRegimeID</i>, <i>planned</i>) \in dom <i>regimeGeneticMaterial</i> \wedge</p> <p> (<i>aRegimeID</i>, <i>actual</i>) \in dom <i>regimeGeneticMaterial</i> \wedge</p> <p> <i>whichSpecies?</i> \in ran <i>regimeGeneticMaterial</i> \wedge</p> <p> (<i>whichSpecies?</i> \in dom <i>genusOfPureSpeciesWood</i> \vee</p> <p> <i>whichSpecies?</i> \in dom <i>genusOfHybridWood</i>) \wedge</p> <p> ((<i>whichSpecies?</i> \in dom <i>pureSpeciesWood</i> \vee</p> <p> <i>whichSpecies?</i> \in dom <i>hybridWood</i>) \Rightarrow</p> <p> <i>regimeGeneticMaterial</i> (<i>pRegimeID</i>, <i>planned</i>) = <i>whichSpecies?</i>) \vee</p> <p> (<i>genID</i> \in ran <i>genusWood</i> \wedge</p> <p> <i>regimeGeneticMaterial</i> (<i>pRegimeID</i>, <i>planned</i>) = <i>genusWood</i>~ <i>genID</i> \wedge</p> <p> ((<i>genusOfPureSpeciesWood</i> <i>whichSpecies?</i>) = <i>genID</i> \vee</p> <p> (<i>genusOfHybridWood</i> <i>whichSpecies?</i>) = <i>genID</i>)) \wedge</p> <p> <i>regimeIDs</i>' = <i>regimeIDs</i> \cup {(<i>aRegimeID</i>, <i>actual</i>)} \wedge</p> <p> <i>currentRegime</i>' = <i>currentRegime</i> \oplus {<i>whichStand?</i> \mapsto <i>aRegimeID</i>} \wedge</p> <p> <i>standRegimeType</i>' = <i>standRegimeType</i> \oplus {(<i>whichStand?</i>, <i>planned</i>) \mapsto <i>pRegimeID</i>} \wedge</p> <p> <i>standRegimeType</i>' = <i>standRegimeType</i> \oplus {(<i>whichStand?</i>, <i>actual</i>) \mapsto <i>aRegimeID</i>} \wedge</p> <p> <i>standsRegimes</i>' = <i>standsRegimes</i> \oplus</p> <p> {<i>whichStand?</i> \mapsto (<i>pRegimeID</i>, <i>aRegimeID</i>)} \wedge</p> <p> <i>regimeGeneticMaterial</i>' = <i>regimeGeneticMaterial</i> \oplus</p> <p> {(<i>aRegimeID</i>, <i>actual</i>) \mapsto <i>whichSpecies?</i>} \wedge</p> <p> <i>prevRegime</i>' = <i>prevRegime</i></p>
--

The schema *PlantStandAndRecordRegimeOK* records that the stand is planted and updates the details of the actual regime. This schema includes the changeable schemas

RegimePlantDate, *RegimePlanting* and *StandPlantingStatus*. It also includes the unchangeable schemas *StoreRegimes* and *StandOfTrees*. It also includes two input values, *standToPlant?* (the stand to be planted) and *plantingCompletionDate?* (the date of completion of planting, stored as day, month and year)⁴. The actual regime of the stand is accessed via the function *currentRegime*. From this, the planned regime can be determined from interrogating the function *standsRegimes*. Before the stand is planted, it must have a planting status of *unplanted*. Once it is planted, this status changes to *planted*. The actual regime's planting date is updated with *plantingCompletionDate?*; the actual regime's planting density and planted stems per hectare are updated with the planned regime's planting density and planted stems per hectare, respectively.

PlantStandAndRecordRegimeOK

Δ *RegimePlantDate*

Δ *RegimePlanting*

Ξ *StoreRegimes*

Δ *StandPlantingStatus*

Ξ *StandOfTrees*

standToPlant? : *STANDID*

plantingCompletionDate? : *DATE*

$standToPlant? \in \text{dom } currentRegime$

$standToPlant? \in \text{dom } standsRegimes$

$standToPlant? \in \text{dom } plantingStatus$

$\exists pRegimeID : REGIMEID \bullet \exists aRegimeID : REGIMEID \bullet$

$aRegimeID \in \text{ran } currentRegime \wedge$

$(pRegimeID, aRegimeID) \in \text{ran } standsRegimes \wedge$

$(aRegimeID, actual) \in \text{dom } plantDate \wedge$

$plantingCompletionDate? \in \text{ran } plantDate \wedge$

$(aRegimeID, actual) \in \text{dom } plantDensity \wedge$

$(aRegimeID, actual) \in \text{dom } plantedSPH \wedge$

$(currentRegime \text{ } standToPlant?) = aRegimeID \wedge$

$(standsRegimes \text{ } standToPlant?) = (pRegimeID, aRegimeID) \wedge$

$plantingStatus \text{ } standToPlant? = unplanted \wedge$

$plantingStatus' = plantingStatus \oplus \{standToPlant? \mapsto planted\} \wedge$

$plantDate' = plantDate \oplus \{(aRegimeID, actual) \mapsto plantingCompletionDate?\} \wedge$

$plantDensity' = plantDensity \oplus$

$\{(aRegimeID, actual) \mapsto plantDensity (pRegimeID, planned)\} \wedge$

$plantedSPH' = plantedSPH \oplus$

$\{(aRegimeID, actual) \mapsto plantedSPH (pRegimeID, planned)\}$

The schema *ExceptionPlantStandAndRecordRegime* governs the eventuality that a stand is already planted when the forestry staff come to plant it (i.e. the records in the

⁴The way the schema is written makes it sound like the planting will be completed 'instantaneously'; in practice it may take a few days, depending on the size of the stand, and the date recorded as the planting date is the date of the completion of planting.

plantation database were not kept up to date). This schema includes the unchangeable schema *StandPlantingStatus* and states that if the stand which is to be planted (*standToPlant?*) is already planted, an error message will be given saying that the stand is already planted.

The schema *PlantStandAndRecordRegime* then implements schema *PlantStandAndRecordRegimeOK* or implements schema *ExceptionPlantStandAndRecordRegime*.

\exists <i>StandPlantingStatus</i> <i>standToPlant?</i> : <i>STANDID</i> <i>message!</i> : <i>MESSAGE</i>
<i>standToPlant?</i> \in <i>dom plantingStatus</i> <i>plantingStatus standToPlant? = planted</i> \Rightarrow <i>message! = StandAlreadyPlanted</i>

$$PlantStandAndRecordRegime \hat{=} PlantStandAndRecordRegimeOK \vee ExceptionPlantStandAndRecordRegime$$

The schema *PlantStand* combines the two schemas *DeterminePlantingRegime* and *PlantStandAndRecordRegime*. The two input standIDs (*whichStand?* and *standToPlant?*) are equal – i.e. they refer to the same stand.

<i>DeterminePlantingRegime</i> <i>PlantStandAndRecordRegime</i>
<i>whichStand? = standToPlant?</i>

D.4.5 Harvesting the stand

In preparation for determining the volume and mass of the logs which will be made when harvesting the stand, certain schemas are defined. First of all, the age of the trees in the stand is calculated (schema *CalculateAgeOfTreesInStand*). Next, schema *EstimateStandVolume* defines the estimated volume from the stand. Following that, schema *ConvertStandVolumeToMass* converts the estimated volume into mass, as timber is transported by mass. Finally, the stand is harvested.

The harvesting of the stand has been broken up into two schemas. In the first, *StandsVolumeAndMassAtHarvesting*, given the inputs of the stand's ID and the date

when harvesting must begin, the planned regime's ID is determined. In addition, the volume and mass of the logs which would be harvested from the stand at that date are calculated. In the second part, the trees are harvested, and the actual regime is updated. The logs are added to the pile at roadside. This is undertaken in schema *HarvestStandAndUpdateRegime*. These two schemas are combined to form the schema *HarvestStand*.

The schema *DefineAgeOfTreesInStand* includes a function *treeAge*, which takes as input two dates, and gives as output an age. This schema states that the age, which is the function's output, must always be greater than or equal to zero.

The schema *InitAgeOfTreesInStand* includes the changeable schema *DefineAgeOfTreesInStand*. This schema initialises the function *treeAge* to be undefined. The schema *AgeOfTreesInStand* then combines the two schemas *DefineAgeOfTreesInStand* and *InitAgeOfTreesInStand*.

<i>DefineAgeOfTreesInStand</i>
$treeAge : (DATE \times DATE) \rightarrow AGE$
$\forall date1 : DATE \bullet$ $\quad \forall date2 : DATE \bullet$ $\quad (date1, date2) \in \text{dom } treeAge \wedge$ $\quad treeAge (date1, date2) \geq 0$

<i>InitAgeOfTreesInStand</i>
$\Delta DefineAgeOfTreesInStand$
$treeAge' = \emptyset$

$$AgeOfTreesInStand \hat{=} DefineAgeOfTreesInStand \wedge InitAgeOfTreesInStand$$

The schema *CalculateAgeOfTreesInStand* is used to calculate the age of the trees in a stand (or the difference between two dates). It includes the changeable schema, *AgeOfTreesInStand*. In order to calculate the stand's age, the (planting date \times harvesting date) should be input into *treeAge*. If the first date is before the second date, the age is the difference in years and months. If the dates are the same, then the trees age is zero. If the first date is after the second date, the function is undefined.

This function works out the difference between two dates to the nearest month; this is the granularity with which tree ages are calculated in the forestry industry.

CalculateAgeOfTreesInStand
 Δ *AgeOfTreesInStand*

$\forall date1 : DATE \bullet$
 $\quad \forall date2 : DATE \bullet$
 $\quad \exists age : AGE \bullet$
 $\quad (date1, date2) \in \text{dom } treeAge \wedge$
 $\quad age \in \text{ran } treeAge \wedge$
 $\quad ((date1.year * 12 + date1.month) < (date2.year * 12 + date2.month)) \Rightarrow$
 $\quad \quad treeAge' = treeAge \oplus \{(date1, date2) \mapsto ((date2.year * 12 + date2.month) \text{ div } 12$
 $\quad \quad \quad - (date1.year * 12 + date1.month) \text{ div } 12)\} \wedge$
 $\quad (date1 = date2) \Rightarrow$
 $\quad \quad treeAge' = treeAge \oplus \{(date1, date2) \mapsto 0\}$

The schema *EstimateStandsVolume* is used to calculate the estimated stand volume at a certain age. Schema *DefineEstimateStandsVolume* contains a function, *standVolume*, which takes as input the standID, the regimeID, the ID of the genetic material planted, the stand's site index and age, the stand's stems per hectare and area. The output of this function is the estimated volume of logs resulting from harvesting the stand. The volume is greater than or equal to zero. Normally, the growth and yield models would calculate the volume per hectare, then multiply that value by the stand's area (see section 4.6.2).

The schema *InitEstimateStandsVolume* includes the changeable schema *DefineEstimateStandsVolume* and initialises the output of function *standVolume* to the empty set. The schema *SetEstimateStandsVolumeToZero* sets the values of the function *standVolume* to zero. The schema *EstimateStandsVolume* combines the three schemas *DefineEstimateStandsVolume*, *InitEstimateStandsVolume* and *SetEstimateStandsVolumeToZero*.

DefineEstimateStandsVolume
 $standVolume : (STANDID \times REGIMEID \times GENMATERIALID \times HEIGHT$
 $\quad \times AGE \times SPH \times AREA) \rightarrow VOLUME$

$\forall sID : STANDID \bullet$
 $\quad \forall regID : REGIMEID \bullet$
 $\quad \forall genMatID : GENMATERIALID \bullet$
 $\quad \forall SI : HEIGHT \bullet$
 $\quad \forall age : AGE \bullet$
 $\quad \forall stemsPerHa : SPH \bullet$
 $\quad \forall standArea : AREA \bullet$
 $\quad SI > 0 \wedge$
 $\quad age \geq 0 \wedge$
 $\quad stemsPerHa > 0 \wedge$
 $\quad standArea > 0 \wedge$
 $\quad (sID, regID, genMatID, SI, age, stemsPerHa, standArea) \in \text{dom } standVolume \wedge$
 $\quad standVolume (sID, regID, genMatID, SI, age, stemsPerHa, standArea) \geq 0$

<i>InitEstimateStandsVolume</i>
Δ <i>DefineEstimateStandsVolume</i>
$standVolume' = \emptyset$

<i>SetEstimateStandsVolumeToZero</i>
Δ <i>InitEstimateStandsVolume</i>
$\forall sID : STANDID \bullet$ $\quad \forall regID : REGIMEID \bullet$ $\quad \quad \forall genMatID : GENMATERIALID \bullet$ $\quad \quad \quad \forall SI : HEIGHT \bullet$ $\quad \quad \quad \quad \forall age : AGE \bullet$ $\quad \quad \quad \quad \quad \forall stemsPerHa : SPH \bullet$ $\quad \quad \quad \quad \quad \quad \forall standArea : AREA \bullet$ $\quad \quad \quad \quad \quad \quad \quad SI > 0 \wedge$ $\quad \quad \quad \quad \quad \quad \quad age \geq 0 \wedge$ $\quad \quad \quad \quad \quad \quad \quad stemsPerHa > 0 \wedge$ $\quad \quad \quad \quad \quad \quad \quad standArea > 0 \wedge$ $\quad \quad \quad (sID, regID, genMatID, SI, age, stemsPerHa, standArea) \in \text{dom } standVolume \wedge$ $\quad \quad \quad standVolume' = standVolume \cup$ $\quad \quad \quad \quad \{((sID, regID, genMatID, SI, age, stemsPerHa, standArea), 0)\}$

$$EstimateStandsVolume \hat{=} DefineEstimateStandsVolume \wedge$$

$$InitEstimateStandsVolume \wedge SetEstimateStandsVolumeToZero$$

The schema *ConvertStandVolumeToMass* converts the estimated volume of logs (produced by harvesting the stand) to a mass. Although there is a close (nearly 1:1) relationship between the volume and mass of logs, the conversion is necessary because the logs are transported and sold by mass, and not by volume. As described in section 4.4.2, the mass of the logs from the stand depends on the genetic material of the logs, and the volume. It also depends on the wood's density and the length of time since harvesting (not modelled here).

Schema *DefineConvertStandVolumeToMass* contains the unchangeable schema *EstimateStandsVolume*, and the function *standMass*. This function takes as inputs the standID, the ID of the genetic material of the logs and the volume, and gives the logs' mass. The output of the function *standMass* is always greater than or equal to zero. If the stand's volume is greater than zero, the mass will also be greater than zero.

Schema *InitConvertStandVolumeToMass* includes the changeable schema *DefineConvertStandVolumeToMass*. This schema initialises the function *standMass* to the sptyset, while schema *SetMassToZero* sets the function's

value to zero. Schema *ConvertStandVolumeToMass* combines the three schemas *DefineConvertStandVolumeToMass*, *InitConvertStandVolumeToMass* and *SetMassToZero*.

$\overline{\text{DefineConvertStandVolumeToMass}}$ $\exists \text{EstimateStandsVolume}$ $\text{standMass} : (\text{STANDID} \times \text{GENMATERIALID} \times \text{VOLUME}) \rightarrow \text{MASS}$
$\forall sID : \text{STANDID} \bullet$ $\quad \forall \text{regID} : \text{REGIMEID} \bullet$ $\quad \forall \text{genMatID} : \text{GENMATERIALID} \bullet$ $\quad \forall SI : \text{HEIGHT} \bullet$ $\quad \forall \text{age} : \text{AGE} \bullet$ $\quad \forall \text{stemsPerHa} : \text{SPH} \bullet$ $\quad \forall \text{standArea} : \text{AREA} \bullet$ $\quad (\text{sID}, \text{regID}, \text{genMatID}, SI, \text{age}, \text{stemsPerHa}, \text{standArea}) \in \text{dom standVolume} \wedge$ $\quad (\text{sID}, \text{genMatID}, \text{standVolume} (\text{sID}, \text{regID}, \text{genMatID}, SI, \text{age}, \text{stemsPerHa}, \text{standArea}))$ $\quad \in \text{dom standMass} \wedge$ $\quad \text{standMass} (\text{sID}, \text{genMatID}, \text{standVolume} (\text{sID}, \text{regID}, \text{genMatID}, SI, \text{age},$ $\quad \text{stemsPerHa}, \text{standArea})) \geq 0 \wedge$ $\quad \text{standVolume} (\text{sID}, \text{regID}, \text{genMatID}, SI, \text{age}, \text{stemsPerHa}, \text{standArea}) > 0 \Rightarrow$ $\quad \text{standMass} (\text{sID}, \text{genMatID}, \text{standVolume} (\text{sID}, \text{regID}, \text{genMatID}, SI, \text{age},$ $\quad \text{stemsPerHa}, \text{standArea})) > 0$

$\overline{\text{InitConvertStandVolumeToMass}}$ $\Delta \text{DefineConvertStandVolumeToMass}$ $\text{standMass}' = \emptyset$

$\overline{\text{SetMassToZero}}$ $\Delta \text{InitConvertStandVolumeToMass}$ $\forall sID : \text{STANDID} \bullet$ $\quad \forall \text{genMatID} : \text{GENMATERIALID} \bullet$ $\quad \exists \text{vol} : \text{VOLUME} \bullet$ $\quad (\text{sID}, \text{genMatID}, \text{vol}) \in \text{dom standMass} \wedge$ $\quad \text{standMass}' = \text{standMass} \cup \{((\text{sID}, \text{genMatID}, \text{vol}), 0)\}$
--

$$\text{ConvertStandVolumeToMass} \hat{=} \text{DefineConvertStandVolumeToMass} \wedge$$

$$\text{InitConvertStandVolumeToMass} \wedge \text{SetMassToZero}$$

StandsVolumeAndMassAtHarvesting

Ξ *TrackActualRegimes*

Ξ *RegimeFunctions*

Ξ *ActualRegimes*

Ξ *StandCharacteristics*

Ξ *StandSiteIndex*

Δ *StandVolumeAndMass*

Δ *ConvertStandVolumeToMass*

CalculateAgeOfTreesInStand

whichStand? : *STANDID*

dateToStartHarvesting? : *DATE*

$whichStand? \in \text{dom } currentRegime \wedge whichStand? \in \text{dom } standsRegimes$

$whichStand? \in \text{dom } standArea$

$(whichStand?, dateToStartHarvesting?) \in \text{dom } standVolumeAtHarvesting$

$(whichStand?, dateToStartHarvesting?) \in \text{dom } standMassAtHarvesting$

$\exists_1 pRegimeID : REGIMEID \bullet \exists_1 aRegimeID : REGIMEID \bullet \exists_1 genID : GENUSID \bullet$

$currentRegime \text{ whichStand?} = aRegimeID \wedge$

$standsRegimes \text{ whichStand?} = (pRegimeID, aRegimeID) \wedge$

$(aRegimeID, actual) \in \text{dom } fellDate \wedge$

$\{fellDate (aRegimeID, actual)\} = \emptyset \wedge$

$(aRegimeID, actual) \in \text{dom } plantDate \wedge$

$(aRegimeID, actual) \in \text{dom } regimeGeneticMaterial \wedge$

$(aRegimeID, actual) \in \text{dom } plantedSPH \wedge$

$(plantDate (aRegimeID, actual), dateToStartHarvesting?) \in \text{dom } treeAge \wedge$

$(\{genusOfPureSpeciesWood (regimeGeneticMaterial (aRegimeID, actual))\} \neq \emptyset \Rightarrow$

$genID = genusOfPureSpeciesWood (regimeGeneticMaterial (aRegimeID, actual)) \vee$

$\{genusOfHybridWood (regimeGeneticMaterial (aRegimeID, actual))\} \neq \emptyset \Rightarrow$

$genID = genusOfHybridWood (regimeGeneticMaterial (aRegimeID, actual)) \wedge$

$(whichStand?, evalAge (genID)) \in \text{dom } siteIndex \wedge$

$genID \in \text{dom } evalAge \wedge$

$(whichStand?, aRegimeID, regimeGeneticMaterial (aRegimeID, actual),$

$siteIndex (whichStand?, evalAge (genID)),$

$treeAge (plantDate (aRegimeID, actual), dateToStartHarvesting?),$

$plantedSPH (aRegimeID, actual), standArea \text{ whichStand?})$

$\in \text{dom } standVolume \wedge$

$standVolumeAtHarvesting' = standVolumeAtHarvesting \oplus$

$\{(whichStand?, dateToStartHarvesting?) \mapsto$

$standVolume (whichStand?, aRegimeID, regimeGeneticMaterial (aRegimeID, actual),$

$siteIndex (whichStand?, evalAge (genID)),$

$treeAge (plantDate (aRegimeID, actual), dateToStartHarvesting?),$

$plantedSPH (aRegimeID, actual), standArea \text{ whichStand?})\} \wedge$

$(whichStand?, regimeGeneticMaterial (aRegimeID, actual),$

$(standVolumeAtHarvesting' (whichStand?, dateToStartHarvesting?)))$

$\in \text{dom } standMass \wedge$

$standMassAtHarvesting' = standMassAtHarvesting \oplus$

$\{(whichStand?, dateToStartHarvesting?) \mapsto$

$standMass (whichStand?, regimeGeneticMaterial (aRegimeID, actual),$

$(standVolumeAtHarvesting' (whichStand?, dateToStartHarvesting?)))\}$

The schema *StandsVolumeAndMassAtHarvesting* estimates the stand's volume and mass at the time of harvesting. It includes the unchangeable schemas *TrackActualRegimes*, *RegimeFunctions*, *RegimeFunctions*,

StandCharacteristics and *StandSiteIndex*, and the changeable schemas *StandVolumeAndMass* and *ConvertStandVolumeToMass*. It also includes the schema *CalculateAgeOfTreesInStand*. The schema has two inputs: *whichStand?* (the stand which is due to be harvested) and *dateToStartHarvesting?* (the date on which harvesting is due to start).

The volume of the stand's logs needs to be calculated at the time of harvesting. In order to do this, the function *standVolume* needs to be called (this would be akin to calling a growth and yield simulator). This function takes as inputs the stand's ID, actual regime ID, genetic material ID, the site index (evaluated at the evaluation age for that genetic material's genus), the trees age, planting density (stems per hectare), and the stand's area. Before this function can be evaluated, the genus of the stand's genetic material needs to be determined, so as to be able to access the site index's evaluation age. The function *standVolumeAtHarvesting* is updated with the stand's volume. This volume is then converted into mass by using the function *standMass*, and the resultant mass is stored in *standMassAtHarvesting*.

The schema *HarvestStandAndUpdateRegimeOK* includes four unchangeable schemas, *TrackActualRegimes*, *RegimeFunctions*, *PlannedRegimes*, *ActualRegimes* and *StandVolumeAndMass*. It includes the schema *CalculateAgeOfTreesInStand*, which calculates the age of the trees in the stand, and the changeable schemas, *RegimeFellDate*, *LogsAtRoadside* and *StandPlantingStatus*. There are three inputs to this schema: the standID to be harvested (*standToHarvest?*), the date when harvesting should start (*harvestingStartDate?*), and the date when the harvesting is complete (*harvestingCompletionDate?*)⁵ – both recorded as day, month and year.

Via the function *standsRegimes*, evaluated for *whichStand?*, one can determine the stand's planned and actual regimes (*pRegimeID* and *aRegimeID*). *aRegimeID* is the stand's current actual regime (the output of the function *currentRegime standToHarvest?*). The actual regime has a genetic material ID, a planting date, a planting density and a record of the planted stems per hectare. It does not have a felling date.

Before harvesting can commence, the stand's planting status must be *planted* and the age of the trees in the stand must equal the planned rotation age. After the stand has been harvested, the actual regime's fell date is updated with the date when harvesting was completed, the current regime becomes the stand's previous regime, the current regime for the stand is no longer defined, and the mass of logs at roadside is updated with the mass of logs generated from the stand (stored in the function *standMassAtHarvesting* in schema *StandVolumeAndMassAtHarvesting*). Finally, the stand's planting status changes to unplanted.

⁵Felling a stand may take weeks or months, depending on its area and the terrain.

HarvestStandAndUpdateRegimeOK

\exists *TrackActualRegimes*

\exists *RegimeFunctions*

\exists *PlannedRegimes*

\exists *ActualRegimes*

Δ *RegimeFellDate*

CalculateAgeOfTreesInStand

\exists *StandVolumeAndMass*

Δ *LogsAtRoadside*

Δ *StandPlantingStatus*

standToHarvest? : *STANDID*

harvestingStartDate? : *DATE*

harvestingCompletionDate? : *DATE*

$standToHarvest? \in \text{dom } currentRegime \wedge standToHarvest? \in \text{dom } standsRegimes$

$(standToHarvest?, harvestingStartDate?) \in \text{dom } standMassAtHarvesting$

$\exists_1 pRegimeID : REGIMEID \bullet \exists_1 aRegimeID : REGIMEID \bullet$

$currentRegime \ standToHarvest? = aRegimeID \wedge$

$standsRegimes \ standToHarvest? = (pRegimeID, aRegimeID) \wedge$

$(aRegimeID, actual) \in \text{dom } fellDate \wedge$

$\{fellDate (aRegimeID, actual)\} = \emptyset \wedge$

$harvestingCompletionDate? \in \text{ran } fellDate \wedge$

$(aRegimeID, actual) \in \text{dom } plantDate \wedge$

$(aRegimeID, actual) \in \text{dom } regimeGeneticMaterial \wedge$

$(aRegimeID, actual) \in \text{dom } plantedSPH \wedge$

$\{fellDate (aRegimeID, actual)\} = \emptyset \wedge$

$(plantDate (aRegimeID, actual), harvestingStartDate?) \in \text{dom } treeAge \wedge$

$(standToHarvest?, millForStandsLogs \ standToHarvest?) \in \text{dom } logsMassAtRoadside \wedge$

$standToHarvest? \in \text{dom } plantingStatus \wedge$

$plantingStatus \ standToHarvest? = planted \wedge$

$treeAge (plantDate (aRegimeID, actual), harvestingStartDate?) =$

$fellAge (pRegimeID, planned) \wedge$

$fellDate' = fellDate \oplus \{(aRegimeID, actual) \mapsto harvestingCompletionDate?\} \wedge$

$prevRegime' = prevRegime \oplus \{standToHarvest? \mapsto (currentRegime \ standToHarvest?)\} \wedge$

$\{(currentRegime' \ standToHarvest?)\} = \emptyset \wedge$

$logsMassAtRoadside' = logsMassAtRoadside \oplus$

$\{(standToHarvest?, millForStandsLogs \ standToHarvest?) \mapsto$

$standMassAtHarvesting (standToHarvest?, harvestingStartDate?)\} \wedge$

$plantingStatus' = plantingStatus \oplus \{standToHarvest? \mapsto unplanted\}$

The schema *ExceptionHarvestStandAndUpdateRegime* governs the eventuality that a stand is due to be harvested and it is not planted when the forestry staff come to harvest it (i.e. the records in the plantation database were not kept up to date). This schema includes the unchangeable schema *StandPlantingStatus* and states that if the stand which is to be harvested (*standToHarvest?*) is unplanted, an error message will be given saying that the stand is unplanted.

The schema $HarvestStandAndUpdateRegime$ then implements schema $HarvestStandAndUpdateRegimeOK$ or schema $ExceptionHarvestStandAndUpdateRegime$.

$ExceptionHarvestStandAndUpdateRegime$ $\exists StandPlantingStatus$ $standToHarvest? : STANDID$ $message! : MESSAGE$
$standToHarvest? \in \text{dom } plantingStatus$ $plantingStatus \text{ } standToHarvest? = unplanted \Rightarrow$ $message! = StandNotPlanted$

$$HarvestStandAndUpdateRegime \hat{=} HarvestStandAndUpdateRegimeOK \vee ExceptionHarvestStandAndUpdateRegime$$

The schema $HarvestStand$ combines the two schemas $StandsVolumeAndMassAtHarvesting$ and $HarvestStandAndUpdateRegime$. The two schemas' input standIDs ($whichStand?$ and $standToHarvest?$) are equal – i.e. they refer to the same stand; the two schemas' input harvesting start dates ($dateToStartHarvesting?$ and $harvestingStartDate?$) are also equal.

$HarvestStand$ $StandsVolumeAndMassAtHarvesting$ $HarvestStandAndUpdateRegime$
$whichStand? = standToHarvest?$ $dateToStartHarvesting? = harvestingStartDate?$

D.4.6 Plantation actions

The two forestry schemas describing forestry actions are put together in one overarching schema called $PlantationActions$.

$$PlantationActions \hat{=} PlantStand \wedge HarvestStand$$

D.5 Transport

The transport section covers the roads, the trucks which will travel along them, the short-haul transport (taking logs from the stand to the depot) and the long-haul transport (taking the logs from the depot to the mill). Logs not grown by this integrated plantation forestry company may also arrive at the mill for processing: they are accepted at the mill in the schema *AcceptLogsFromOtherSuppliers*. Finally, and overarching schema governing all transportation is presented.

D.5.1 Preliminary definitions

Each road, truck (lorry) and trip are identified with IDs: *ROADID*, *TRUCKID* and *TRIPID*.

$$[ROADID, TRUCKID, TRIPID]$$

The schema *DefineTripIDs* defines the finite set of IDs for each trip which a truck makes. Each trip has a unique identifier (*tripID*). The schema *InitTripIDs* initialises this set to the empty set. Schema *TripIDs* combines schemas *DefineTripIDs* and *InitTripIDs*.

$\begin{array}{l} \textit{DefineTripIDs} \\ \textit{tripIDs} : \mathbb{F} \textit{TRIPID} \end{array}$
--

$\begin{array}{l} \textit{InitTripIDs} \\ \Delta \textit{DefineTripIDs} \\ \textit{tripIDs}' = \emptyset \end{array}$

$$\textit{TripIDs} \hat{=} \textit{DefineTripIDs} \wedge \textit{InitTripIDs}$$

An axiom is defined below in which a *smallDistance* (greater than zero) is defined to be less than 500 metres. (Ideally this would be defined as a real number, with the upper limit of 0.5km.) *smallDistance* is used in the schemas *LoadLogsAtStand* and *TransportAndUnloadLogsAtDepot*: if the distance between the stand's roadside and the depot is smaller than *smallDistance*, trucks will not be used to take the timber from roadside to the depot. The forwarders used in the extraction of the timber to roadside will be used to do the short-haul transport instead.

$\begin{array}{l} \textit{smallDistance} : \mathbb{N}_1 \\ \textit{smallDistance} \leq 500 \end{array}$

D.5.2 Roads

The roads in this specification are broken up into two parts: the short-haul roads are specified in schema *ShortHaulRoad* and the long-haul roads are specified in schema *LongHaulRoad*. They are merged in the schema *Road*.

The schema *ShortHaulRoad* describes the road which links the stand to the depot. It includes the unchanged schema *LogisticsChain*. Two functions are specified: *roadStandToDepot* defines which road (or roads) links (or link) the stand and the depot; and *shortHaulDistance* defines the distance between a stand and a depot on a particular road. For each stand and for each depot to which timber from that stand could be sent, there is at least one road linking them. The short-haul distance may be zero (when the depot is at roadside) but it may also be (usually is) greater than zero.

ShortHaulRoad

\exists *LogisticsChain*

roadStandToDepot : *ROADID* \leftrightarrow (*STANDID* \times *DEPOTID*)

shortHaulDistance : (*STANDID* \times *DEPOTID* \times *ROADID*) \rightarrow *DISTANCE*

\forall *sID* : *STANDID* •

\forall *dID* : *DEPOTID* •

\exists *rID* : *ROADID* •

sID \in dom *sendLogsToDepot* \wedge

dID \in ran *sendLogsToDepot* \wedge

dID \in dom *sendLogsToMill* \wedge

rID \in dom *roadStandToDepot* \wedge

sID \in {*first*(*roadStandToDepot* *rID*)}

\wedge *dID* \in {*second*(*roadStandToDepot* *rID*)}

\wedge (*sendLogsToDepot* *sID*) = *dID* \wedge

(*roadStandToDepot* *rID*) = (*sID*, (*sendLogsToDepot* *sID*)) \wedge

$\#\{\textit{roadStandToDepot}^{\sim}\} \geq 1 \wedge$

shortHaulDistance (*sID*, *dID*, *rID*) ≥ 0

The schema *LongHaulRoad* describes the road which links the depot to the mill. It includes the unchanged schema *LogisticsChain*. Two functions are specified: *roadDepotToMill* defines which road or roads links or link the depot to the mill; and *longHaulDistance* defines the distance between a depot and a mill on a particular road. For each depot and for each mill to which timber from that depot could be sent, there is always at least one road linking them. The long-haul distance is greater than zero.

LongHaulRoad <hr/> $\exists \text{LogisticsChain}$ $\text{roadDepotToMill} : \text{ROADID} \rightarrow (\text{DEPOTID} \times \text{MILLID})$ $\text{longHaulDistance} : (\text{DEPOTID} \times \text{MILLID} \times \text{ROADID}) \rightarrow \text{DISTANCE}$ <hr/> $\forall dID : \text{DEPOTID} \bullet$ $\quad \forall mID : \text{MILLID} \bullet$ $\quad \exists rID : \text{ROADID} \bullet$ $\quad dID \in \text{dom } \text{sendLogsToMill} \wedge$ $\quad mID \in \text{ran } \text{sendLogsToMill} \wedge$ $\quad rID \in \text{dom } \text{roadDepotToMill} \wedge$ $\quad dID \in \{\text{first}(\text{roadDepotToMill } rID)\} \wedge$ $\quad mID \in \{\text{second}(\text{roadDepotToMill } rID)\} \wedge$ $\quad (\text{sendLogsToMill } dID) = mID \wedge$ $\quad (\text{roadDepotToMill } rID) = (dID, (\text{sendLogsToMill } dID)) \wedge$ $\quad \#\{\text{roadDepotToMill} \sim\} \geq 1 \wedge$ $\quad \text{longHaulDistance } (dID, mID, rID) > 0$

The schema *Road* is made up of the two schemas, *ShortHaulRoad* and *LongHaulRoad*.

$$\text{Road} \hat{=} \text{ShortHaulRoad} \wedge \text{LongHaulRoad}$$

D.5.3 Trucks

In this specification, trucks are defined in terms of the maximum load they can carry (in schema *TruckMaxLoad*) and the load they are currently carrying (in schema *Truck*). The maximum possible load per truck is important because regulation exists which aims to prevent trucks from being overloaded, as this would damage the roads on which they travel.

The schema *TruckMaxLoad* defines each truck's maximum load. It specifies a function, *maxLoad*, which gives the maximum mass that the truck is allowed to carry.

TruckMaxLoad <hr/> $\text{maxLoad} : \text{TRUCKID} \rightarrow \text{MASS}$ <hr/> $\forall trID : \text{TRUCKID} \bullet$ $\quad trID \in \text{dom } \text{maxLoad} \wedge$ $\quad (\text{maxLoad } trID) > 0$

The schema *Truck* includes the unchanging schemas *TruckMaxLoad* and *TripIDs*, and specifies a function, *currentLoadsMass*. *currentLoadsMass* gives the mass of the logs in the load for the truck for the current trip. The trip's ID must be in

the set of trip's IDs (as each trip ID is unique). Any truck's load may never exceed the maximum allowable load, and the load's mass is greater than (or equal to) zero.

$$\begin{array}{l}
 \textit{Truck} \\
 \hline
 \exists \textit{TruckMaxLoad} \\
 \exists \textit{TripIDs} \\
 \textit{currentLoadsMass} : (\textit{TRUCKID} \times \textit{TRIPID}) \rightarrow \textit{MASS} \\
 \hline
 \forall \textit{trID} : \textit{TRUCKID} \bullet \\
 \quad \exists_1 \textit{tripID} : \textit{TRIPID} \bullet \\
 \quad (\textit{trID}, \textit{tripID}) \in \textit{dom currentLoadsMass} \wedge \\
 \quad \textit{trID} \in \textit{dom maxLoad} \wedge \\
 \quad \textit{tripID} \in \textit{tripIDs} \wedge \\
 \quad \textit{currentLoadsMass} (\textit{trID}, \textit{tripID}) \leq (\textit{maxLoad trID}) \wedge \\
 \quad \textit{currentLoadsMass} (\textit{trID}, \textit{tripID}) \geq 0
 \end{array}$$

D.5.4 Short-haul transport

The short-haul trip from the stand's roadside log pile to the depot involves defining the short-haul trip, loading the logs at the roadside, and transporting them and unloading them at the depot (in piles, according to the mill for which they are destined). These three aspects are undertaken in schemas *ShortHaulTrip*, *LoadLogsAtStand* and *TransportAndUnloadLogsAtDepot*. The schema *ShortHaulTransport* combines these last two schemas.

D.5.4.1 Short-haul trip

The short-haul trip schema given below records the details of each truck's trip between the stand (roadside) and the depot. The same truck may make multiple trips, but the trip's ID (which is unique) will be different for each trip.

The schema *ShortHaulTrip* includes the unchangeable schema *TripIDs* and contains the functions *shortHaulTrip* and *shortHaulTripMass*. Both have as inputs the truckID and tripID. *shortHaulTrip* has as output the origin (a stand) and a destination (a depot) of the short-haul journey, as well as the road on which the journey must be made. *shortHaulTripMass* has as output the mass of logs carried on that trip.

The trip's ID must be in the set *tripIDs*, as these IDs must be unique. For each truck ID and tripID, there will be only one output (a stand ID, depot ID and road ID tuple for the function *shortHaulTrip* and a mass for *shortHaulTripMass*). For *shortHaulTrip*, this means that by the time the truck takes the trip, there is only one stand which is the point of departure, there is only one destination depot, and only one road along

which the truck will travel for that trip. In addition, the mass of logs carried by the truck on the trip must be greater than zero.

$\begin{array}{l} \textit{ShortHaulTrip} \\ \exists \textit{TripIDs} \\ \textit{shortHaulTrip} : (\textit{TRUCKID} \times \textit{TRIPID}) \rightarrow (\textit{STANDID} \times \textit{DEPOTID} \times \textit{ROADID}) \\ \textit{shortHaulTripMass} : (\textit{TRUCKID} \times \textit{TRIPID}) \rightarrow \textit{MASS} \\ \hline \forall \textit{trID} : \textit{TRUCKID} \bullet \\ \quad \exists_1 \textit{tripID} : \textit{TRIPID} \bullet \\ \quad \textit{tripID} \in \textit{tripIDs} \wedge \\ \quad (\textit{trID}, \textit{tripID}) \in \textit{dom shortHaulTrip} \wedge \\ \quad (\textit{trID}, \textit{tripID}) \in \textit{dom shortHaulTripMass} \wedge \\ \quad \#\{(\textit{shortHaulTrip} (\textit{trID}, \textit{tripID}))\} = 1 \wedge \\ \quad \#\{(\textit{shortHaulTripMass} (\textit{trID}, \textit{tripID}))\} = 1 \wedge \\ \quad \textit{shortHaulTripMass} (\textit{trID}, \textit{tripID}) > 0 \end{array}$

D.5.4.2 Load logs at stand (roadside)

The first step in short-haul transport is to load the logs onto the truck at the stand's roadside. This is described in schema *LoadLogsAtStand*.

The schema *LoadLogsAtStand* governs the addition of a load of logs (or timber) to a particular truck at the stand's roadside. It includes the unchanging schemas, *LogisticsChain*, *MillForStandsLogs* and *ShortHaulRoad*, and the changeable schemas *LogsAtRoadside*, *TripIDs*, *Truck* and *ShortHaulTrip*. The schema states each stand has a single depot destination, and a single mill destination, for its logs. This depot and mill are linked logistically via the functions *sendLogsToDepot* and *sendLogsToMill*.

If the distance between the stand and the depot is between zero and the upper limit defined in the constant *smallDistance*, the loading of the logs and the short-haul transportation will not occur. If the distance between the stand and the depot is greater than *smallDistance* metres, then for every stand where there are logs at roadside, the schema states that there is always a depot to which those logs should be taken, there is always a road linking the two, and there is always a truck available⁶ which can transport them⁷, and at least one trip which that truck can take from the origin to the destination on the chosen road. Before any logs are loaded onto the truck, the truck must be empty. If the mass of logs at roadside is greater than the truck's maximum capacity (*maxLoad*), the current load (*currentLoadsMass*) for the truck and the trip will be the maximum load.

⁶i.e. we are not describing a logistics scheduling problem here.

⁷It is assumed that the logs will be transported to the depot sometime soon after felling and before the time the next rotation (crop) of trees is harvested.

LoadLogsAtStand

\exists *LogisticsChain*

\exists *MillForStandsLogs*

\exists *ShortHaulRoad*

Δ *LogsAtRoadside*

Δ *TripIDs*

Δ *Truck*

Δ *ShortHaulTrip*

$\forall sID : STANDID \bullet$

$\exists_1 dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$

$\exists_1 rID : ROADID \bullet$

$\exists trID : TRUCKID \bullet$

$\exists tripID : TRIPID \bullet$

$sID \in \text{dom } \text{sendLogsToDepot} \wedge$

$dID \in \text{ran } \text{sendLogsToDepot} \wedge$

$dID \in \text{dom } \text{sendLogsToMill} \wedge$

$mID \in \text{ran } \text{sendLogsToMill} \wedge$

$sID \in \text{dom } \text{millForStandsLogs} \wedge$

$mID \in \text{ran } \text{millForStandsLogs} \wedge$

$(\text{millForStandsLogs } sID) = mID \wedge$

$(sID, mID) \in \text{dom } \text{logsMassAtRoadside} \wedge$

$rID \in \text{dom } \text{roadStandToDepot} \wedge$

$(sID, dID) \in \text{ran } \text{roadStandToDepot} \wedge$

$(trID, tripID) \in \text{dom } \text{currentLoadsMass} \wedge$

$trID \in \text{dom } \text{maxLoad} \wedge$

$(trID, tripID) \in \text{dom } \text{shortHaulTrip} \wedge$

$(sID, dID, rID) \in \text{ran } \text{shortHaulTrip} \wedge$

$(trID, tripID) \in \text{dom } \text{shortHaulTripMass} \wedge$

$(sID, dID, rID) \in \text{dom } \text{shortHaulDistance} \wedge$

$(\text{sendLogsToMill } (\text{sendLogsToDepot } sID)) = mID \wedge$

$(\text{sendLogsToDepot } sID) = dID \wedge$

$\text{shortHaulDistance } (sID, dID, rID) > \text{smallDistance} \Rightarrow$

$tripID \notin \text{tripIDs} \wedge$

$(\text{currentLoadsMass } (trID, tripID)) = 0 \wedge$

$(\text{logsMassAtRoadside } (sID, mID)) \geq (\text{maxLoad } trID) \Rightarrow$

$(\text{currentLoadsMass}' = \text{currentLoadsMass} \oplus \{(trID, tripID) \mapsto$
 $(\text{maxLoad } (trID))\}) \wedge$

$\text{logsMassAtRoadside}' = \text{logsMassAtRoadside} \oplus \{(sID, mID) \mapsto$
 $(\text{logsMassAtRoadside } (sID, mID)$

$- \text{currentLoadsMass}' (trID, tripID))\} \wedge$

$\text{tripIDs}' = \text{tripIDs} \cup \{tripID\} \wedge$

$\text{shortHaulTrip}' = \text{shortHaulTrip} \oplus \{(trID, tripID) \mapsto (sID, dID, rID)\} \wedge$

$\text{shortHaulTripMass}' = \text{shortHaulTripMass} \oplus \{(trID, tripID) \mapsto$
 $\text{maxLoad } (trID)\})$

If there are fewer logs at roadside than the maximum possible load, these logs will be left at roadside to rot, as it is not financially worthwhile to make a trip with

a partial load. The mass of logs at roadside is reduced by the mass of the logs which were loaded onto the truck. Before the trip is undertaken, the trip's ID is not in the set of existing trip's IDs; after the trip is started, the trip's ID is added to the set *tripIDs*. The function *shortHaulTrip* for the truck and trip is assigned the values of the stand where the truck was loaded (origin) and the depot to which the truck will travel (destination), and the road along which it will travel between these two. The function *shortHaulTripMass* for the truck and trip is assigned the value of the truck's load. (This is needed for calculating the short-haul cost for the trip.)

D.5.4.3 Transport and unload logs at depot

The second step in short-haul transport is to transport the logs to the depot and unload them. The logs will be unloaded into piles according to the mill for which they are destined. This is described in schema *TransportAndUnloadLogsAtDepot*.

The schema *TransportAndUnloadLogsAtDepot* governs the short-haul transport of the loaded truck from the stand to the depot along the road, and also the unloading of the truck at the depot, for a particular trip. It includes five schemas which cannot be changed (*LogisticsChain*, *MillForStandsLogs*, *ShortHaulRoad*, *TripIDs* and *LoadLogsAtStand*) and two which can be changed (*Depot* and *Truck*). As with the loading in schema *LoadLogsAtStand*, this transportation and unloading only occurs if the short-haul distance (between the stand and the depot) is greater than *smallDistance* metres. In order to be unloaded, the the truck must be at the correct depot, and truck's load (*currentLoadsMass*) must be greater than zero. The logs on the truck are unloaded and added to the pile at the depot which is destined for the mill⁸, as specified by the function *MillForStandsLogs*. At the end of the unloading process, the truck is empty.

⁸As the specification stands at present, there are no space constraints at the depot.

TransportAndUnloadLogsAtDepot

\exists *LogisticsChain*

\exists *MillForStandsLogs*

\exists *ShortHaulRoad*

\exists *TripIDs*

\exists *LoadLogsAtStand*

Δ *Depot*

Δ *Truck*

$\forall sID : STANDID \bullet \exists_1 dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$

$\exists_1 rID : ROADID \bullet \exists trID : TRUCKID \bullet$

$\exists tripID : TRIPID \bullet$

$sID \in \text{dom } millForStandsLogs \wedge$

$mID \in \text{ran } millForStandsLogs \wedge$

$sID \in \text{dom } sendLogsToDepot \wedge$

$dID \in \text{ran } sendLogsToDepot \wedge$

$(dID, mID) \in \text{dom } logsMassAtDepot \wedge$

$rID \in \text{dom } roadStandToDepot \wedge$

$(sID, dID) \in \text{ran } roadStandToDepot \wedge$

$(trID, tripID) \in \text{dom } currentLoadsMass \wedge$

$(trID, tripID) \in \text{dom } shortHaulTrip \wedge$

$(sID, dID, rID) \in \text{ran } shortHaulTrip \wedge$

$(sID, dID, rID) \in \text{dom } shortHaulDistance \wedge$

$tripID \in tripIDs \wedge$

$shortHaulDistance (sID, dID, rID) > smallDistance \Rightarrow$

$((sendLogsToDepot sID) = dID \wedge$

$(shortHaulTrip (trID, tripID)) = (sID, dID, rID) \wedge$

$(currentLoadsMass (trID, tripID)) > 0 \wedge$

$logsMassAtDepot' = logsMassAtDepot \oplus \{(dID, mID) \mapsto$

$((logsMassAtDepot (dID, mID)) + currentLoadsMass (trID, tripID))\} \wedge$

$currentLoadsMass' = currentLoadsMass \oplus \{(trID, tripID) \mapsto 0\}$

D.5.4.4 Short-haul transport

The two schemas *LoadLogsAtStand* and *TransportAndUnloadLogsAtDepot* are combined in an overarching schema, *ShortHaulTransport*.

$$ShortHaulTransport \hat{=} LoadLogsAtStand \wedge TransportAndUnloadLogsAtDepot$$

D.5.5 Long-haul transport

The long-haul trip from the depot's log pile (sorted according to the mill for which the logs are destined) to the mill involves defining the long-haul trip, loading the logs at the depot, and transporting them and unloading them at the mill. These three aspects are undertaken in schemas *LongHaulTrip*, *LoadLogsAtDepot* and

TransportAndUnloadLogsAtMill. The schema *LongHaulTransport* combines these last two schemas.

D.5.5.1 Long-haul trip

The long-haul trip schema given below records the details of each truck's trip between the depot and the mill. The same truck may make multiple trips, but the trip's ID (which is unique) will be different for each trip.

The schema *LongHaulTrip* includes the unchangeable schema *TripIDs* and contains the functions *longHaulTrip* and *longHaulTripMass*. Both have as inputs the truckID and tripID. *longHaulTrip* has as output the origin (a stand) and a destination (a depot) of the long-haul journey, as well as the road on which the journey must be made. *longHaulTripMass* has as output the mass of logs carried on that trip.

The trip's ID must be in the set *tripIDs*, as these IDs must be unique. For each truck ID and tripID, there will be only one output (a stand ID, depot ID and road ID tuple for the function *longHaulTrip* and a mass for *longHaulTripMass*). For *longHaulTrip*, this means that by the time the truck takes the trip, there is only one stand which is the point of departure, there is only one destination depot, and only one road along which the truck will travel for that trip. In addition, the mass of logs carried by the truck on the trip must be greater than zero.

<p><i>LongHaulTrip</i></p> <p>$\exists TripIDs$</p> <p>$longHaulTrip : (TRUCKID \times TRIPID) \leftrightarrow (DEPOTID \times MILLID \times ROADID)$</p> <p>$longHaulTripMass : (TRUCKID \times TRIPID) \rightarrow MASS$</p> <hr/> <p>$\forall trID : TRUCKID \bullet$</p> <p>$\exists_1 tripID : TRIPID \bullet$</p> <p>$tripID \in tripIDs \wedge$</p> <p>$(trID, tripID) \in \text{dom } longHaulTrip \wedge$</p> <p>$(trID, tripID) \in \text{dom } longHaulTripMass \wedge$</p> <p>$\#\{(longHaulTrip (trID, tripID))\} = 1 \wedge$</p> <p>$\#\{(longHaulTripMass (trID, tripID))\} = 1 \wedge$</p> <p>$longHaulTripMass (trID, tripID) > 0$</p>

D.5.5.2 Load logs at depot

The first step in long-haul transport is to load the logs onto the truck at the depot. This is described in schema *LoadLogsAtDepot*.

LoadLogsAtDepot

\exists *LogisticsChain*

\exists *MillForStandsLogs*

\exists *LongHaulRoad*

Δ *Depot*

Δ *TripIDs*

Δ *Truck*

Δ *LongHaulTrip*

$$\begin{aligned} & \forall dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet \\ & \quad \exists_1 rID : ROADID \bullet \exists trID : TRUCKID \bullet \\ & \quad \exists tripID : TRIPID \bullet \\ & \quad (dID, mID) \in \text{dom } logsMassAtDepot \wedge \\ & \quad dID \in \text{dom } sendLogsToMill \wedge \\ & \quad mID \in \text{ran } sendLogsToMill \wedge \\ & \quad mID \in \text{ran } millForStandsLogs \wedge \\ & \quad \text{ran } millForStandsLogs \subseteq \text{ran } sendLogsToMill \wedge \\ & \quad rID \in \text{dom } roadDepotToMill \wedge \\ & \quad (dID, mID) \in \text{ran } roadDepotToMill \wedge \\ & \quad trID \in \text{dom } maxLoad \wedge \\ & \quad (trID, tripID) \in \text{dom } currentLoadsMass \wedge \\ & \quad (trID, tripID) \in \text{dom } longHaulTrip \wedge \\ & \quad (dID, mID, rID) \in \text{ran } longHaulTrip \wedge \\ & \quad (trID, tripID) \in \text{dom } longHaulTripMass \wedge \\ & \quad (dID, mID, rID) \in \text{dom } longHaulDistance \wedge \\ & \quad (sendLogsToMill dID) = mID \wedge \\ & \quad (logsMassAtDepot (dID, mID)) \geq (maxLoad trID) \Rightarrow \\ & \quad \quad tripID \notin tripIDs \wedge \\ & \quad \quad (currentLoadsMass (trID, tripID) = 0 \wedge \\ & \quad \quad \quad currentLoadsMass' = currentLoadsMass \oplus \\ & \quad \quad \quad \{(trID, tripID) \mapsto (maxLoad trID)\}) \wedge \\ & \quad \quad logsMassAtDepot' = logsMassAtDepot \oplus \{(dID, mID) \mapsto \\ & \quad \quad \quad (logsMassAtDepot (dID, mID) - currentLoadsMass (trID, tripID))\}) \wedge \\ & \quad \quad tripIDs' = tripIDs \cup \{tripID\} \wedge \\ & \quad \quad longHaulTrip' = longHaulTrip \oplus \{(trID, tripID) \mapsto (dID, mID, rID)\} \wedge \\ & \quad \quad \quad longHaulTripMass' = longHaulTripMass \oplus \{(trID, tripID) \mapsto \\ & \quad \quad \quad \quad maxLoad (trID)\}) \end{aligned}$$

The schema *LoadLogsAtDepot* governs the addition of a load of logs (or timber) to a particular truck at the depot. It includes the unchangeable schemas, *LogisticsChain*, *MillForStandsLogs* and *LongHaulRoad*, and the changeable schemas *Depot*, *TripIDs*, *Truck* and *LongHaulTrip*. For every depot which has timber, the schema states that there is always a mill to which those logs should be taken, there is always a road linking the two, and there is always a truck available⁹ which can transport them, and at least one trip which that truck can take from the origin to the destination on the chosen road.

⁹i.e. we are not describing a logistics scheduling problem here.

If the mass of logs at the depot, in the pile destined for mill *mID*, is greater than the truck's maximum capacity (*maxLoad*), some of the logs can be transported. Before loading, the truck must be empty. The current load of the truck (*currentLoadsMass*) will be the maximum load. If the mass of logs at the depot is less than the mass of the truck's maximum load, those logs will not be transported to the mill; they will be left at the depot, as it is not financially worthwhile to make a trip with a partial load. The mass of logs at the depot is reduced by the mass of the logs which were loaded onto the truck. Before the trip is undertaken, the trip's ID is not in the set of existing trip's IDs; after the trip is started, the trip's ID is added to the set *tripIDs*. The function *longHaulTrip* for the truck is assigned the values of the depot where the truck was loaded (origin) and the mill to which the truck will travel (destination). The function *longHaulTripMass* for the truck and trip is assigned the value of the truck's load. (This is needed for calculating the short-haul cost for the trip.)

D.5.5.3 Transport and unload logs at mill

The second step in long-haul transport is to transport the logs to the mill and unload them at the mill's logyard. This is described in the schema *TransportAndUnloadLogsAtMill*.

The schema *TransportAndUnloadLogsAtMill* governs the long-haul transport of the loaded truck from the depot to the mill along the road, and also the unloading of the truck at the mill, for a particular trip. It includes five schemas which cannot be changed (*LogisticsChain*, *MillForStandsLogs*, *LongHaulRoad*, *LoadLogsAtDepot* and *TripIDs*) and two which can be changed (*Mill* and *Truck*). In order to be unloaded, the truck must be at the correct mill, and the truck's load (*currentLoadsMass*) must be greater than zero. The logs on the truck are unloaded and added to the logs in the mill's logyard¹⁰, and at the end of the unloading process, the truck is empty.

¹⁰As the specifications stand at present, there are no space constraints at the mill's logyard.

TransportAndUnloadLogsAtMill

\exists *LogisticsChain*
 \exists *MillForStandsLogs*
 \exists *LongHaulRoad*
 \exists *LoadLogsAtDepot*
 \exists *TripIDs*
 Δ *Mill*
 Δ *Truck*

$\forall dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$
 $\exists_1 rID : ROADID \bullet \exists trID : TRUCKID \bullet$
 $\exists tripID : TRIPID \bullet$
 $dID \in \text{dom } sendLogsToMill \wedge$
 $mID \in \text{ran } sendLogsToMill \wedge$
 $mID \in \text{ran } millForStandsLogs \wedge$
 $\text{ran } millForStandsLogs \subseteq \text{ran } sendLogsToMill \wedge$
 $(dID, mID) \in \text{dom } logsMassAtDepot \wedge$
 $rID \in \text{dom } roadDepotToMill \wedge$
 $(dID, mID) \in \text{ran } roadDepotToMill \wedge$
 $(trID, tripID) \in \text{dom } currentLoadsMass \wedge$
 $(trID, tripID) \in \text{dom } longHaulTrip \wedge$
 $(dID, mID, rID) \in \text{ran } longHaulTrip \wedge$
 $(dID, mID, rID) \in \text{dom } longHaulDistance \wedge$
 $tripID \in tripIDs \wedge$
 $(sendLogsToMill dID) = mID \wedge$
 $(longHaulTrip (trID, tripID)) = (dID, mID, rID) \wedge$
 $currentLoadsMass (trID, tripID) > 0 \wedge$
 $logyard' = logyard \oplus \{mID \mapsto$
 $\quad ((logyard mID) + (currentLoadsMass (trID, tripID)))\} \wedge$
 $currentLoadsMass' = currentLoadsMass \oplus \{(trID, tripID) \mapsto 0\}$

D.5.5.4 Long-haul transport

The two schemas *LoadLogsAtDepot* and *TransportAndUnloadLogsAtMill* are combined in an overarching schema, *LongHaulTransport*.

$$LongHaulTransport \hat{=} LoadLogsAtDepot \wedge TransportAndUnloadLogsAtMill$$

D.5.6 Transport

The schema *Transport* made by combining the two schemas *ShortHaulTransport* and *LongHaulTransport*.

$$Transport \hat{=} ShortHaulTransport \wedge LongHaulTransport$$

D.5.7 Accepting logs from other suppliers

Not all logs which arrive at the mill will come from the plantation forestry company's own stands. Logs could be grown by other timber growers and delivered to the mill. The schema *AcceptLogsFromOtherSuppliers* describes this.

In the schema *AcceptLogsFromOtherSuppliers*, the schema *MillAcceptableSpecies* is included, and its contents can be changed. It also includes two inputs, *loadOfLogs?* (of type $(MASS \times GENMATERIALID)$), which represents a load of logs (or tree-lengths) brought to the mill, and *whichMill?*, which identifies the mill to which the logs (or tree-lengths) are being brought. In order for the load to be accepted at the mill, they have to be of an acceptable species. The genetic material can be described as either a genus, or a genus-species or genus-species-species combination¹¹. The logs are unloaded and added to the the logs in the mill's logyard.

<p><i>AcceptLogsFromOtherSuppliers</i></p> <p>$\exists MillAcceptableSpecies$</p> <p>$\Delta Mill$</p> <p><i>loadOfLogs?</i> : $(MASS \times GENMATERIALID)$</p> <p><i>whichMill?</i> : <i>MILLID</i></p> <hr/> <p><i>first(loadOfLogs?)</i> \in <i>ran logyard</i></p> <p><i>whichMill?</i> \in <i>dom logyard</i></p> <p><i>whichMill?</i> \in <i>dom acceptableSpecies</i></p> <p><i>second(loadOfLogs?)</i> \in $\{(acceptableSpecies\ whichMill?)\}$</p> <p>$\forall genMatID : GENMATERIALID \bullet$</p> <p>$genMatID \in \{second(loadOfLogs?)\} \wedge$</p> <p>$genMatID \in \{(acceptableSpecies\ whichMill?)\} \wedge$</p> <p>$(genMatID \in \text{dom } genusWood \vee$</p> <p>$genMatID \in \text{dom } pureSpeciesWood \vee$</p> <p>$genMatID \in \text{dom } hybridWood) \wedge$</p> <p>$logyard' = logyard \oplus \{whichMill? \mapsto ((logyard\ whichMill?) + first(loadOfLogs?))\}$</p>

D.6 Processing

D.6.1 Removing logs for processing

At the mill, the logs are removed from the logyard for processing. This is shown in schema *RemoveLogsForProcessing*.

¹¹Ideally, one would know both the genus and species of the logs or tree-lengths delivered, but the species information may not be available.

The schema *ProcessLogs* includes the changeable schema *Mill*, also has as inputs *load?* (of type *MASS*), the load of logs removed from the logyard, and *whichMill?*, to identify the mill at which the logs are being removed. Before a load of logs can be removed, one has to ensure that the logyard has at least a load of logs to remove. After the removal of the load, the mass of logs in the logyard is diminished by the mass of the load.

The schema *ExceptionProcessLogs* includes the unchangeable schema *Mill*, and also has as inputs *load?* and *whichMill?*. In addition, it has as an output *message!*. If there are not enough logs in the logyard to remove a load (*load?*) for processing, an output message is generated stating that there are not enough logs to remove a whole load.

Schema *RemoveLogsForProcessing* then chooses either schema *ProcessLogs* or schema *ExceptionProcessLogs*.

<i>ProcessLogs</i>
$\Delta Mill$ $load? : MASS$ $whichMill? : MILLID$
$load? \in \text{ran } logyard$ $whichMill? \in \text{dom } logyard$ $logyard \text{ whichMill?} \geq load? \Rightarrow$ $logyard' = logyard \oplus \{whichMill? \mapsto ((logyard \text{ whichMill?}) - load?)\}$

<i>ExceptionProcessLogs</i>
$\exists Mill$ $load? : MASS$ $whichMill? : MILLID$ $message! : MESSAGE$
$load? \in \text{ran } logyard$ $whichMill? \in \text{dom } logyard$ $logyard \text{ whichMill?} < load? \Rightarrow$ $message! = NotEnoughLogs$

$$RemoveLogsForProcessing \hat{=} ProcessLogs \vee ExceptionProcessLogs$$

D.7 Forest-to-mill supply chain

In this section, all of the aspects of the forest-to-mill supply chain are brought together into one schema called *ForestToMillSupplyChain*.

The overarching schema *ForestToMillSupplyChain* is made by combining the four schemas *PlantationActions*, *Transport*, *AcceptLogsFromOtherSuppliers* and *RemoveLogsForProcessing*. These schemas were defined in sections D.4.6, D.5.6, D.5.7 and D.6.1 respectively.

$$\text{ForestToMillSupplyChain} \hat{=} \text{PlantationActions} \wedge \text{Transport} \wedge \\ \text{AcceptLogsFromOtherSuppliers} \wedge \text{RemoveLogsForProcessing}$$

D.8 Log file generated by ZTC for the forest-to-mill domain

This section gives the Z Type Checker output file (.log) obtained when the formal specification given above was typechecked.

Log opened at: Mon Jun 01 06:45:03 2009

```
... Initializing.
... Loading Z mathematical tools library: math0.zed
Parsing main file: ..\latex\MSc_specs_31-05-09.tex
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 221
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 224
... Type checking Equivalence definition: GENMATNAME. "..\latex\MSc_specs_31-05-09.tex" Line 226
... Type checking Schema box: GeneticMaterialNames. "..\latex\MSc_specs_31-05-09.tex" Lines 254-273
... Type checking Schema box: Genus. "..\latex\MSc_specs_31-05-09.tex" Lines 295-314
... Type checking Schema box: PureSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 337-371
... Type checking Schema box: Hybrid. "..\latex\MSc_specs_31-05-09.tex" Lines 388-425
... Type checking Schema definition: WoodGeneticMaterial. "..\latex\MSc_specs_31-05-09.tex" Line 447
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 475
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 479-481
... Type checking Free type definition: REGIMETYPE. "..\latex\MSc_specs_31-05-09.tex" Line 486
... Type checking Schema box: DateDefinition. "..\latex\MSc_specs_31-05-09.tex" Lines 503-521
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 533
... Type checking Schema box: RegimeGeneticMaterial. "..\latex\MSc_specs_31-05-09.tex" Lines 555-572
... Type checking Schema box: RegimePlantAge. "..\latex\MSc_specs_31-05-09.tex" Lines 593-600
... Type checking Schema box: RegimePlantDate. "..\latex\MSc_specs_31-05-09.tex" Lines 612-618
... Type checking Schema box: RegimePlanting. "..\latex\MSc_specs_31-05-09.tex" Lines 630-652
... Type checking Schema box: RegimeFellAge. "..\latex\MSc_specs_31-05-09.tex" Lines 674-683
... Type checking Schema box: RegimeFellDate. "..\latex\MSc_specs_31-05-09.tex" Lines 726-734
... Type checking Schema definition: RegimeFelling. "..\latex\MSc_specs_31-05-09.tex" Line 751
... Type checking Schema box: DefineTrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 774-787
... Type checking Schema box: InitTrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 791-797
... Type checking Schema definition: TrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Line 802
... Type checking Schema box: DefineRegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 824-825
... Type checking Schema box: InitRegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 829-835
... Type checking Schema definition: RegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Line 840
... Type checking Schema box: RegimeFunctions. "..\latex\MSc_specs_31-05-09.tex" Lines 851-854
```

```

... Type checking Schema box: PlannedRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 867-900
... Type checking Schema box: ActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 912-949
... Type checking Schema definition: StoreRegimes. "..\latex\MSc_specs_31-05-09.tex" Line 971
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 1004
... Type checking Free type definition: MESSAGE. "..\latex\MSc_specs_31-05-09.tex" Line 1017
... Type checking Schema box: LogisticsChain. "..\latex\MSc_specs_31-05-09.tex" Lines 1036-1052
... Type checking Schema box: MillForStandsLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 1064-1084
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1106
... Type checking Schema box: DefineLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1132-1146
... Type checking Schema box: InitLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1150-1153
... Type checking Schema box: NoLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1157-1162
... Type checking Schema definition: LogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Line 1167
... Type checking Schema box: DefineDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1189-1214
... Type checking Schema box: InitDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1221-1224
... Type checking Schema box: NoLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1228-1233
... Type checking Schema definition: Depot. "..\latex\MSc_specs_31-05-09.tex" Line 1238
... Type checking Schema box: MillAcceptableSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 1260-1288
... Type checking Schema box: DefineMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1303-1330
... Type checking Schema box: InitMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1334-1337
... Type checking Schema box: NoLogsAtMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1341-1346
... Type checking Schema definition: Mill. "..\latex\MSc_specs_31-05-09.tex" Line 1351
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 1400
... Type checking Schema box: CompanyHasEstates. "..\latex\MSc_specs_31-05-09.tex" Lines 1411-1418
... Type checking Schema box: EstateBelongsToCompany. "..\latex\MSc_specs_31-05-09.tex" Lines 1429-1435
... Type checking Schema box: EstateHasStands. "..\latex\MSc_specs_31-05-09.tex" Lines 1446-1453
... Type checking Schema box: StandBelongsToEstate. "..\latex\MSc_specs_31-05-09.tex" Lines 1464-1470
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1490
... Type checking Schema box: StandSuitableSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 1502-1545
... Type checking Schema box: StandCharacteristics. "..\latex\MSc_specs_31-05-09.tex" Lines 1557-1565
... Type checking Schema definition: StandLand. "..\latex\MSc_specs_31-05-09.tex" Line 1578
... Type checking Free type definition: PLANTINGSTATE. "..\latex\MSc_specs_31-05-09.tex" Line 1598
... Type checking Schema box: DefineStandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1613-1618
... Type checking Schema box: InitStandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1622-1626
... Type checking Schema box: StandPlantingStatusUnplanted. "..\latex\MSc_specs_31-05-09.tex" Lines 1630-1636
... Type checking Schema definition: StandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1641-1642
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1679
... Type checking Schema box: SiteIndexEvalAge. "..\latex\MSc_specs_31-05-09.tex" Lines 1691-1698
... Type checking Schema box: StandSiteIndex. "..\latex\MSc_specs_31-05-09.tex" Lines 1709-1724
... Type checking Schema box: DefineStandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1738-1749
... Type checking Schema box: InitStandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1753-1757
... Type checking Schema box: SetStandVolumeAndMassToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 1761-1769
... Type checking Schema definition: StandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1774-1775
... Type checking Schema box: StandOfTrees. "..\latex\MSc_specs_31-05-09.tex" Lines 1792-1841
... Type checking Schema box: DeterminePlantingRegime. "..\latex\MSc_specs_31-05-09.tex" Lines 1870-1915
... Type checking Schema box: PlantStandAndRecordRegimeOK. "..\latex\MSc_specs_31-05-09.tex" Lines 1929-1966
... Type checking Schema box: ExceptionPlantStandAndRecordRegime. "..\latex\MSc_specs_31-05-09.tex"
    Lines 1983-1993
... Type checking Schema definition: PlantStandAndRecordRegime. "..\latex\MSc_specs_31-05-09.tex"
    Lines 1998-1999
... Type checking Schema box: PlantStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2011-2015
... Type checking Schema box: DefineAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2041-2047
... Type checking Schema box: InitAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2051-2054
... Type checking Schema definition: AgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2059-2060
... Type checking Schema box: CalculateAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2074-2086
... Type checking Schema box: DefineEstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2104-2120
... Type checking Schema box: InitEstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2124-2127
... Type checking Schema box: SetEstimateStandsVolumeToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 2131-2147
... Type checking Schema definition: EstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2152-2153
... Type checking Schema box: DefineConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex"
    Lines 2171-2189
... Type checking Schema box: InitConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex" Lines 2193-2197
... Type checking Schema box: SetMassToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 2201-2209
... Type checking Schema definition: ConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex"
    Lines 2214-2215
... Type checking Schema box: StandsVolumeAndMassAtHarvesting. "..\latex\MSc_specs_31-05-09.tex"
    Lines 2247-2300
... Type checking Schema box: HarvestStandAndUpdateRegimeOK. "..\latex\MSc_specs_31-05-09.tex" Lines 2320-2365
... Type checking Schema box: ExceptionHarvestStandAndUpdateRegime. "..\latex\MSc_specs_31-05-09.tex"

```

```
Lines 2379-2386
... Type checking Schema definition: HarvestStandAndUpdateRegime. "..\latex\MSc_specs_31-05-09.tex"
Lines 2391-2392
... Type checking Schema box: HarvestStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2403-2408
... Type checking Schema definition: PlantationActions. "..\latex\MSc_specs_31-05-09.tex" Line 2427
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 2455
... Type checking Schema box: DefineTripIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 2465-2466
... Type checking Schema box: InitTripIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 2470-2473
... Type checking Schema definition: TripIDs. "..\latex\MSc_specs_31-05-09.tex" Line 2478
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 2489-2491
... Type checking Schema box: ShortHaulRoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2511-2534
... Type checking Schema box: LongHaulRoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2546-2568
... Type checking Schema definition: Road. "..\latex\MSc_specs_31-05-09.tex" Line 2581
... Type checking Schema box: TruckMaxLoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2602-2610
... Type checking Schema box: Truck. "..\latex\MSc_specs_31-05-09.tex" Lines 2622-2636
... Type checking Schema box: ShortHaulTrip. "..\latex\MSc_specs_31-05-09.tex" Lines 2668-2682
... Type checking Schema box: LoadLogsAtStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2706-2749
... Type checking Schema box: TransportAndUnloadLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 2771-2802
... Type checking Schema definition: ShortHaulTransport. "..\latex\MSc_specs_31-05-09.tex" Line 2862
... Type checking Schema box: LongHaulTrip. "..\latex\MSc_specs_31-05-09.tex" Lines 2895-2909
... Type checking Schema box: LoadLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 2933-2968
... Type checking Schema box: TransportAndUnloadLogsAtMill. "..\latex\MSc_specs_31-05-09.tex" Lines 2993-3025
... Type checking Schema definition: LongHaulTransport. "..\latex\MSc_specs_31-05-09.tex" Line 3087
... Type checking Schema definition: Transport. "..\latex\MSc_specs_31-05-09.tex" Line 3105
... Type checking Schema box: AcceptLogsFromOtherSuppliers. "..\latex\MSc_specs_31-05-09.tex" Lines 3126-3142
... Type checking Schema box: ProcessLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 3172-3181
... Type checking Schema box: ExceptionProcessLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 3185-3194
... Type checking Schema definition: RemoveLogsForProcessing. "..\latex\MSc_specs_31-05-09.tex" Line 3199
... Type checking Schema definition: ForestToMillSupplyChain. "..\latex\MSc_specs_31-05-09.tex"
Lines 3219-3220
End of main file: ..\latex\MSc_specs_31-05-09.tex
Type report written in ".typ"
Log written in ".log"
```

Log closed at: Mon Jun 01 06:45:04 2009

Appendix E

Formal specification of the forest harvest scheduling system which includes wood properties in the harvesting decision

This appendix gives a formal description of the forest harvest scheduling system which includes wood properties in the harvesting decision. The system described is embedded in the forest-to-mill domain. For brevity, the aspects of the domain which were described in Appendix D are not repeated, but if a schema which was defined in the forest-to-mill domain was used in the forest harvest scheduling system specification, reference is made to the section of Appendix D where that schema is defined. An index of schema names which are found in this appendix and in Appendix D can be found on page 431.

The system described in this appendix is a simplified version of the 'Option 1' version described in section 4.7.1 on page 131. In this strategic-level specification, harvesting occurs at the stand's rotation age; in addition, an assumption has been made that each mill only has one process, which accepts wood according to the wood's mass and the stand's average wood property.

E.1 Planning horizon

Forestry is a long-term activity. Compared to other crops, a stand of trees takes a long time to mature, so recording information about the stands and planning what should happen to them is important. In this section, the type *YEARNO* is defined to cater for all the years in the planning horizon. In addition, the three forestry planning horizons (the long-term/strategic, medium-term/tactical and the

short-term/operational planning horizons) and the date on which the plan is to start are defined.

The specification starts by defining the type *YEARNO* (a finite set of natural numbers) so that the planning horizons can be defined, as well as the activities and constraints in each year of the planning horizon.

| *YEARNO* : \mathbb{N}

The planning horizon is defined in schema *PlanningHorizonDefinition* in terms of its three components: the operational, tactical and strategic planning horizons, which are given as three inputs to the system. The operational planning horizon (which is between one and two years in length) is always shorter than or equal to the tactical planning horizon (defined here to be between three and five years long). The tactical planning horizon, in turn, is shorter than that strategic planning horizon, which is between 20 and 30 years in length. Schema *PlanningHorizonDefinition* also takes in a date (*dateToStartPlan?*) which is the date on which the plan is to be started.

PlanningHorizonDefinition

strategicHorizon?, *tacticalHorizon?*, *operationalHorizon?* : *YEARNO*
dateToStartPlan? : *DATE*

operationalHorizon? \leq *tacticalHorizon?* \leq *strategicHorizon?*

$1 \leq$ *operationalHorizon?* ≤ 2

$3 \leq$ *tacticalHorizon?* ≤ 5

$20 \leq$ *strategicHorizon?* ≤ 30

E.2 Wood properties

This section describes wood properties: how to identify them, and how to evaluate them at a stand level.

WOODPROPID (the identifier of the wood property), *WOODPROPNAME* (the name of the wood property, e.g. density, fibre length) and *WOODPROPVAL* (the value of the wood property measurement) are defined so that they can be used later on in the specification.

[*WOODPROPID*, *WOODPROPNAME*]

| *WOODPROPVAL* : \mathbb{N}

Schema *EstimateStandWoodProp* calculates the average wood property value for a stand of trees and stores the result in the function *standWoodProp*. As described in

section 4.7.5, the prediction of wood properties depends on the wood property being predicted, the genetic material ('species') of the trees in the stand and the site index, age and altitude of the stand. In this schema, it is assumed that the site index is always greater than zero, the age of the stand of trees and the stand altitude are greater or equal to zero. The resultant predicted wood property value is greater than or equal to zero.

EstimateStandWoodProp

$$\text{standWoodProp} : (\text{STANDID} \times \text{WOODPROPID} \times \text{GENMATERIALID} \times \text{HEIGHT} \times \text{AGE} \times \text{HEIGHT}) \rightarrow \text{WOODPROPVAL}$$

$\forall sID : \text{STANDID} \bullet$

$\forall \text{woodPropID} : \text{WOODPROPID} \bullet$

$\forall \text{genMatID} : \text{GENMATERIALID} \bullet$

$\forall SI : \text{HEIGHT} \bullet$

$\forall \text{age} : \text{AGE} \bullet$

$\forall \text{altitude} : \text{HEIGHT} \bullet$

$SI > 0 \wedge$

$\text{age} \geq 0 \wedge$

$\text{altitude} \geq 0 \wedge$

$(sID, \text{woodPropID}, \text{genMatID}, SI, \text{age}, \text{altitude}) \in \text{dom standWoodProp} \wedge$

$\text{standWoodProp}(sID, \text{woodPropID}, \text{genMatID}, SI, \text{age}, \text{altitude}) \geq 0$

E.3 Mill requirements

Each mill requires a certain mass of wood to process annually. The aim is for enough wood to be provided so that the mill does not need to stop working. This section describes the minimum and maximum mass requirements for the mill for each year in the planning horizon, as well as the minimum and maximum acceptable wood property values for each year.

Schema *MillRequirements* defines the mill's timber requirements. It includes the unchangeable schema *PlanningHorizonDefinition*, and has five functions: *minMass*, *maxMass*, *woodPropForMill*, *minWoodPropValue* and *maxWoodPropValue*. The first two (*minMass* and *maxMass*) both take the millID and year number as inputs, and have mass of timber as an output. They record the minimum and maximum required mass of wood at that mill per annum. *woodPropForMill* records the wood property whose value variability the mill is trying to reduce. The next two functions, *minWoodPropValue* and *maxWoodPropValue*, store the minimum and maximum value for this wood property, per annum.

For each mill, and for each year in the strategic planning horizon, the minimum mass required by the mill must be less than or equal to the maximum mass, and the

minimum wood property value must be less than or equal to the maximum wood property value. In addition, there is only one wood property for which the mill is managing its incoming resource.

MillRequirements

\exists *PlanningHorizonDefinition*

$minMass : (MILLID \times YEARNNO) \rightarrow MASS$

$maxMass : (MILLID \times YEARNNO) \rightarrow MASS$

$woodPropForMill : MILLID \rightarrow WOODPROPID$

$minWoodPropValue : (MILLID \times WOODPROPID \times YEARNNO) \rightarrow WOODPROPVAL$

$maxWoodPropValue : (MILLID \times WOODPROPID \times YEARNNO) \rightarrow WOODPROPVAL$

$\forall millID : MILLID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(millID, yearNo) \in \text{dom } minMass \wedge$

$(millID, yearNo) \in \text{dom } maxMass \wedge$

$(millID) \in \text{dom } woodPropForMill \wedge$

$(millID, woodPropForMill(millID), yearNo) \in \text{dom } minWoodPropValue \wedge$

$(millID, woodPropForMill(millID), yearNo) \in \text{dom } maxWoodPropValue \wedge$

$minMass(millID, yearNo) \leq maxMass(millID, yearNo) \wedge$

$\#\{woodPropForMill(millID)\} = 1 \wedge$

$minWoodPropValue(millID, woodPropForMill(millID), yearNo) \leq$

$maxWoodPropValue(millID, woodPropForMill(millID), yearNo)$

E.4 Costs and income

In this section, the costs and income involved in the specification are defined. There are two types of costs: those generated by the forestry company in the growing of trees (see section E.4.2), and those generated when transporting the logs made from those trees to the mills (see section E.4.3). Finally the income generated by the forestry part of the integrated forestry company is described in section E.4.4.

E.4.1 Preliminary definition

COST is defined to be able to specify the costs involved in the system over the strategic planning horizon. These costs are specified in present day terms. This approach is appropriate because the aim of the system is to determine the overall profit of the supply chain and not to model expected interest rate fluctuations over the strategic planning horizon. *COST* is defined as a finite set of integers.

| $COST : \mathbb{FZ}$

E.4.2 Forestry costs

The forestry costs considered in this specification are broken up into three main parts: those incurred at the forestry company level (see section E.4.2.1), those incurred at the estate level (see section E.4.2.2), and those incurred due to implementing a regime (see section E.4.2.3). In this section, schemas are given describing these three types of cost. For the regime costs, the concept of canopy closure needs to be defined, so that annual maintenance costs before and after canopy closure can be specified.

E.4.2.1 Company-level forestry costs

Schema *ForestryCompanyLevelCosts* stores the costs incurred by the forestry company at a company-wide level, e.g. the cost of having a head office, the cost of employing a planning forester and a timber logistics manager, etc. The schema *ForestryCompanyLevelCosts* contains the unchangeable schema *PlanningHorizonDefinition*, and a function, *companyLevelCosts*, which maps the company ID and the year number to the cost for that year. For each year number in the strategic planning horizon, the cost is greater than or equal to zero.

$\begin{array}{l} \text{ForestryCompanyLevelCosts} \\ \exists \text{PlanningHorizonDefinition} \\ \text{companyLevelCosts} : \text{COMPANYID} \times \text{YEARNO} \rightarrow \text{COST} \\ \exists_1 cID : \text{COMPANYID} \bullet \\ \quad \forall yearNo : \text{YEARNO} \bullet \\ \quad 1 \leq yearNo \leq \text{strategicHorizon?} \wedge \\ \quad (cID, yearNo) \in \text{dom } \text{companyLevelCosts} \wedge \\ \quad (\text{companyLevelCosts } (cID, yearNo)) \geq 0 \end{array}$
--

E.4.2.2 Estate-level forestry costs

Schema *EstateLevelCosts* stores the costs of upgrading the estate roads prior to harvesting, the costs of employing foresters for that estate, ensuring that fire-prevention measures are undertaken for the estate, etc. Schema *EstateLevelCosts* contains the unchangeable schema *PlanningHorizonDefinition*, and a function, *estateLevelCosts*, which maps the estateID and the year number to the cost for that year. For each year number, the cost is greater than or equal to zero.

EstateLevelCosts

\exists *PlanningHorizonDefinition*

$estateLevelCosts : ESTATEID \times YEARNNO \rightarrow COST$

$\forall eID : ESTATEID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(eID, yearNo) \in \text{dom } estateLevelCosts \wedge$

$(estateLevelCosts (eID, yearNo)) \geq 0$

E.4.2.3 Regime-incurred (stand-level) forestry costs

As trees are planted, tended and harvested, costs are incurred. This section describes these stand-level costs which are incurred as a result of implementing a regime for a stand of trees. Regimes were described in section D.2.2, but canopy closure was not defined for these regimes. Canopy closure is important, because before it, the annual maintenance costs are greater due to activities such as weeding. After canopy closure, weeding can usually be stopped.

The types of regime actions are defined first, followed by a definition of canopy closure for each regime and genetic material. Regime costs for each part of the regime are then defined.

The type *REGIMEACTION* defines the various regime actions (e.g. 'plant'¹ and 'fell'). It also records when canopy closure is expected to occur (*cClosure*).

$REGIMEACTION ::= plant \mid cClosure \mid fell$

RegimeCanopyClosure

\exists *PlannedRegimes*

$regimeCanopyClosure : REGIMEID \times REGIMETYPE \times GENMATERIALID \rightarrow AGE$

$\forall regID : REGIMEID \bullet$

$\forall genMatID : GENMATERIALID \bullet$

$(regID, planned, genMatID) \in \text{dom } regimeCanopyClosure \wedge$

$(regID, planned) \in \text{dom } plantAge \wedge$

$(regID, planned) \in \text{dom } fellAge \wedge$

$(regimeCanopyClosure (regID, planned, genMatID)) > plantAge (regID, planned) \wedge$

$(regimeCanopyClosure (regID, planned, genMatID)) < fellAge (regID, planned) \wedge$

$\{regimeCanopyClosure (regID, actual, genMatID)\} = \emptyset$

¹In a later version of this specification, when coppicing regimes are considered, 'plant' should be changed to 'establish'.

Schema *RegimeCanopyClosure* is conceptually an addition to the planned regimes schemas, as it contains a function which stores when canopy closure is most likely to occur for a particular regime. This is important to know, as weeding activities can be reduced or cease after this time in the regime, so the stand's maintenance cost will be less.

$\text{regimePlantingCosts} : \text{REGIMEID} \times \text{REGIMETYPE} \times \text{GENMATERIALID} \rightarrow \text{COST}$ $\text{annualRegimeMaintCostBeforeCClosure} :$ $\text{REGIMEID} \times \text{REGIMETYPE} \times \text{GENMATERIALID} \rightarrow \text{COST}$ $\text{annualRegimeMaintCostAfterCClosure} :$ $\text{REGIMEID} \times \text{REGIMETYPE} \times \text{GENMATERIALID} \rightarrow \text{COST}$ $\text{regimeHarvestingCosts} : \text{REGIMEID} \times \text{REGIMETYPE} \times \text{GENMATERIALID} \rightarrow \text{COST}$
$\forall p\text{RegimeID} : \text{REGIMEID} \bullet \forall a\text{RegimeID} : \text{REGIMEID} \bullet$ $\forall \text{genMatID} : \text{GENMATERIALID} \bullet$ $(p\text{RegimeID}, \text{planned}, \text{genMatID}) \in \text{dom regimePlantingCosts} \wedge$ $(a\text{RegimeID}, \text{actual}, \text{genMatID}) \in \text{dom regimePlantingCosts} \wedge$ $(p\text{RegimeID}, \text{planned}, \text{genMatID}) \in \text{dom annualRegimeMaintCostBeforeCClosure} \wedge$ $(a\text{RegimeID}, \text{actual}, \text{genMatID}) \in \text{dom annualRegimeMaintCostBeforeCClosure} \wedge$ $(p\text{RegimeID}, \text{planned}, \text{genMatID}) \in \text{dom annualRegimeMaintCostAfterCClosure} \wedge$ $(a\text{RegimeID}, \text{actual}, \text{genMatID}) \in \text{dom annualRegimeMaintCostAfterCClosure} \wedge$ $(p\text{RegimeID}, \text{planned}, \text{genMatID}) \in \text{dom regimeHarvestingCosts} \wedge$ $(a\text{RegimeID}, \text{actual}, \text{genMatID}) \in \text{dom regimeHarvestingCosts} \wedge$ $(\text{regimePlantingCosts } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \geq 0 \wedge$ $(\text{regimePlantingCosts } (a\text{RegimeID}, \text{actual}, \text{genMatID})) \geq 0 \wedge$ $(\text{annualRegimeMaintCostBeforeCClosure } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \geq 0 \wedge$ $(\text{annualRegimeMaintCostBeforeCClosure } (a\text{RegimeID}, \text{actual}, \text{genMatID})) \geq 0 \wedge$ $(\text{annualRegimeMaintCostAfterCClosure } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \geq 0 \wedge$ $(\text{annualRegimeMaintCostAfterCClosure } (a\text{RegimeID}, \text{actual}, \text{genMatID})) \geq 0 \wedge$ $(\text{regimeHarvestingCosts } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \geq 0 \wedge$ $(\text{regimeHarvestingCosts } (a\text{RegimeID}, \text{actual}, \text{genMatID})) \geq 0 \wedge$ $(\text{annualRegimeMaintCostBeforeCClosure } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \geq$ $(\text{annualRegimeMaintCostAfterCClosure } (p\text{RegimeID}, \text{planned}, \text{genMatID})) \wedge$ $(\text{annualRegimeMaintCostBeforeCClosure } (a\text{RegimeID}, \text{actual}, \text{genMatID})) \geq$ $(\text{annualRegimeMaintCostAfterCClosure } (a\text{RegimeID}, \text{actual}, \text{genMatID}))$

Schema *RegimeCosts* contains the costs incurred by implementing a regime in a stand. It contains four functions, *regimePlantingCosts*, *annualRegimeMaintCostBeforeCClosure*, *annualRegimeMaintCostAfterCClosure* and *regimeHarvestingCosts*, all of which take the regimeID, regime type and genetic materialID as inputs and give the cost as an output. The regime and the regime type could be planned or actual. The first function (*regimePlantingCosts*) stores the cost of planting or establishing the stand. This is a once-off cost (i.e. it is incurred at planting). The second and third functions (*annualRegimeMaintCostBeforeCClosure* and *annualRegimeMaintCostAfterCClosure*) record the annual costs incurred when managing the stand: before canopy closure and after canopy closure. The latter costs

are less than the former. The last function (*regimeHarvestingCosts*) stores the costs incurred at harvesting. All the costs are greater than, or equal to, zero.

E.4.3 Transport costs

The cost of transporting a tonne of timber short-haul (from the stand to the depot) and long-haul (from the depot to the mill) is described in this section. The costs of transporting timber are calculated in section E.7.3.

COSTPERTONNEPERKM is defined to be able to specify the cost rate (cost per tonne per km) for short- and long-haul transport.

$$\left| \begin{array}{l} \text{COSTPERTONNEPERKM} : \mathbb{FN} \end{array} \right.$$

shortHaulTripRate is defined as the cost rate per tonne per km for short-haul trips. This rate is always greater than, or equal to zero. (A stratified version of the short-haul trip rate, as described in Table 4.7 on page 96, can be implemented in a later version.)

$$\left| \begin{array}{l} \text{shortHaulTripRate} : \text{COSTPERTONNEPERKM} \\ \text{shortHaulTripRate} \geq 0 \end{array} \right.$$

longHaulTripRate is defined as the cost rate per tonne per km for long-haul trips. This rate is always greater than, or equal to zero. (A stratified version of the long-haul trip rate, as described in Table 4.7 on page 96, can be implemented in a later version.)

$$\left| \begin{array}{l} \text{longHaulTripRate} : \text{COSTPERTONNEPERKM} \\ \text{longHaulTripRate} \geq 0 \end{array} \right.$$

E.4.4 Forestry income from mills

The cost of purchasing a tonne of timber is described in this section. The income derived from selling timber to the mills is calculated in section E.8.1.1.

COSTPERTONNE is defined to be able to specify the delivered cost rate (cost per tonne) of selling or buying logs.

$$\left| \begin{array}{l} \text{COSTPERTONNE} : \mathbb{FN} \end{array} \right.$$

logCostRate is defined as the delivered cost rate per tonne for logs entering the mill. This rate is always greater than, or equal to zero.

$$\left| \begin{array}{l} \logCostRate : COSTPERTONNE \\ \logCostRate \geq 0 \end{array} \right.$$

E.5 Stand options

In the forest harvest scheduling system, there are many different options which could be chosen for each stand over the strategic planning horizon. During the planning horizon, there are a number of felling (and replanting) events, and each time the stand is felled, a decision must be taken about the mill to which the timber should be sent. The list of possible options for each stand needs to be determined. From these options, a set of possible solutions is generated (see section E.6). This is the combinatorial combination of all the stand options (where each stand has a single stand option in the possible solution). From this, each possible solution can be assessed to determine if it meets the mill's requirements, and the cost of each possible solution can be calculated. The possible solution which meets the constraints and gives the highest profit is the optimal solution.

In this specification, a stand option is defined as the list of possible regime actions (e.g. plant, canopy closure is reached, fell) for a stand over the strategic planning horizon. In addition, as the wood which is harvested from a stand could be sent to many mills, the stand option includes harvesting information, for example, the mill that the timber could be sent to, the wood property that the mill uses to manage its inputs, the mass of timber which would be harvested, etc. In the Z specification which follows, two schemas are used to store this information: *StandRegimeAction* and *StandHarvestingInfo*. Figures E.1 and E.2 (on pages 350 and 352) show examples of stand options (discussed in more detail in section E.5.4).

This section starts with preliminary definitions (section E.5.1), and by defining the number of stand options which will be generated by the system (section E.5.2). The stand options (schemas *StandRegimeAction* and *StandHarvestingInfo*) are then defined and populated (see sections E.5.3 and E.5.4, respectively).

E.5.1 Preliminary definitions

In this section, the type *OPTIONID* is defined, as is the function *Exponent* (which is needed in the calculation of the number of stand options). Two generic definitions are also given to extract an element of a tuple having three or five elements. This is

needed later in the specification.

The type *OPTIONID*, together with the *STANDID*, uniquely defines a stand's option.

[*OPTIONID*]

Schema *Exponent* is defined to calculate a non-zero natural number to the exponent (or power) of another non-zero natural number. It contains a function *exponent* which has two numbers as an input, and outputs the first number to the power of the second number (sometimes written m^n). The output will also be a non-zero natural number.

$\begin{array}{l} \textit{Exponent} \\ \textit{exponent} : (\mathbb{N}_1 \times \mathbb{N}) \rightarrow \mathbb{N}_1 \\ \hline \forall m : \mathbb{N}_1; n : \mathbb{N} \bullet \\ \quad (m, n) \in \text{dom } \textit{exponent} \wedge \\ \quad \textit{exponent} (m, n) \geq 1 \end{array}$
--

A generic definition is defined so that elements from a tuple with three elements may be extracted individually. The elements could be of any types. (ZTC only caters for extracting the *first* and *second* of a couple.) In this generic definition, the function *FirstOf3* extracts the first element, *SecondOf3* extracts the second element and *ThirdOf3* extracts the third element.

$\begin{array}{l} [A, B, C] \\ \textit{FirstOf3} : (A \times B \times C) \rightarrow A \\ \textit{SecondOf3} : (A \times B \times C) \rightarrow B \\ \textit{ThirdOf3} : (A \times B \times C) \rightarrow C \\ \hline \forall a : A; b : B; c : C \bullet \\ \quad \textit{FirstOf3}(a, b, c) = a \wedge \\ \quad \textit{SecondOf3}(a, b, c) = b \wedge \\ \quad \textit{ThirdOf3}(a, b, c) = c \end{array}$

A generic definition is defined so that elements from a tuple with five elements may be extracted individually. The elements could be of any types. In this generic definition, the function *FirstOf5* extracts the first element, *SecondOf5* extracts the second element, *ThirdOf5* extracts the third element, *FourthOf5* extracts the fourth element and *FifthOf5* the fifth element.

$[A, B, C, D, E]$	<hr/> <hr/>
	$FirstOf5 : (A \times B \times C \times D \times E) \rightarrow A$ $SecondOf5 : (A \times B \times C \times D \times E) \rightarrow B$ $ThirdOf5 : (A \times B \times C \times D \times E) \rightarrow C$ $FourthOf5 : (A \times B \times C \times D \times E) \rightarrow D$ $FifthOf5 : (A \times B \times C \times D \times E) \rightarrow E$
	<hr/>
	$\forall a : A; b : B; c : C; d : D; e : E \bullet$ $FirstOf5(a, b, c, d, e) = a \wedge$ $SecondOf5(a, b, c, d, e) = b \wedge$ $ThirdOf5(a, b, c, d, e) = c \wedge$ $FourthOf5(a, b, c, d, e) = d \wedge$ $FifthOf5(a, b, c, d, e) = e$

E.5.2 Defining the number of stand options

The total number of stand options needs to be stored once it is calculated. In order to do this, schemas are given in the following text to define and initialise the total number of stand options.

Schema *DefineNumberOfStandOptions* defines the function *numberOfStandOptions*, which takes the standID as an input and has a natural number greater than zero as an output. For each stand, there is only one value in the range of the function *numberOfStandOptions*, which stores the maximum number of options for that stand.

$DefineNumberOfStandOptions$	<hr/>
	$numberOfStandOptions : STANDID \rightarrow \mathbb{N}_1$
	<hr/>
	$\forall sID : STANDID \bullet$ $sID \in \text{dom } numberOfStandOptions \wedge$ $\#\{numberOfStandOptions \ sID\} = 1$

Schema *InitNumberOfStandOptions* includes the schema *DefineNumberOfStandOptions* and initialises the value of each function to the empty set, while schema *SetNumberOfStandOptionsToOne* initialises the value of each function to one: i.e. there one stand will have at minimum one stand option. Schema *NumberOfStandOptions* combines *DefineNumberOfStandOptions*, *InitNumberOfStandOptions* and *SetNumberOfStandOptionsToOne*.

$InitNumberOfStandOptions$	<hr/>
	$\Delta DefineNumberOfStandOptions$
	<hr/>
	$numberOfStandOptions' = \emptyset$

$$\frac{\text{SetNumberOfStandOptionsToOne}}{\Delta \text{InitNumberOfStandOptions}}$$
$$\forall sID : \text{STANDID} \bullet$$
$$sID \in \text{dom } \text{numberOfStandOptions} \wedge$$
$$\text{numberOfStandOptions}' = \text{numberOfStandOptions} \cup \{(sID, 1)\}$$
$$\text{NumberOfStandOptions} \hat{=} \text{DefineNumberOfStandOptions} \wedge$$
$$\text{InitNumberOfStandOptions} \wedge \text{SetNumberOfStandOptionsToOne}$$

E.5.3 Defining the stand options

The stand options are defined in two different sets of schemas in this section. First, the schemas which make up the *StandRegimeAction* schema are defined and initialised to store the regime actions for each stand, for the strategic planning horizon. Next, schemas which make up the *StandHarvestingInfo* schema are defined and initialised to store the harvesting outcome for each *fell* event in the stand's regime action. This is done for each stand, for the duration of the strategic planning horizon.

Schema *DefineStandRegimeAction* stores the list of possible stand options for the strategic planning horizon. These options are stored in function *standRegimeAction*, which has as inputs the standID, the optionID and the strategic plan year number, and outputs the regime action. Because the possible regime actions are planting, achieving canopy closure and harvesting, there could be no regime actions for a particular year number. If canopy closure is achieved, there will be a single regime action stored. If the stand is to be harvested, there will be two regime actions recorded: harvesting and planting. (It is assumed that the stand will be planted in the same year that it is harvested.)

$$\frac{\text{DefineStandRegimeAction}}{\Xi \text{PlanningHorizonDefinition}}$$
$$\text{standRegimeAction} : \text{STANDID} \times \text{OPTIONID} \times \text{YEARNO} \leftrightarrow \text{REGIMEACTION}$$
$$\forall sID : \text{STANDID} \bullet$$
$$\forall oID : \text{OPTIONID} \bullet$$
$$\forall yearNo : \text{YEARNO} \bullet$$
$$1 \leq yearNo \leq \text{strategicHorizon}? \wedge$$
$$(sID, oID, yearNo) \in \text{dom } \text{standRegimeAction} \wedge$$
$$0 \leq \#\{\text{standRegimeAction}(sID, oID, yearNo)\} \leq 2$$

Schema *InitStandRegimeAction* initialises the function *standRegimeAction*. This schemas includes changeable schema *DefineStandRegimeAction*. The function *standRegimeAction* is initialised to be undefined.

<i>InitStandRegimeAction</i>
Δ <i>DefineStandRegimeAction</i>
<i>standRegimeAction'</i> = \emptyset

Schema *StandRegimeAction* is made up of the two schemas *DefineStandRegimeAction* and *InitStandRegimeAction*.

$$\text{StandRegimeAction} \hat{=} \text{DefineStandRegimeAction} \wedge \text{InitStandRegimeAction}$$

For each harvesting event, there may be several mills to which the stand's timber could be sent. Each of these options needs to be enumerated, and the outcomes of the harvesting event evaluated so that an option for a stand can be chosen by the optimising solver.

Schema *DefineStandHarvestingInfo* records the harvesting information for each of the stand's possible harvesting events. It includes two unchangeable schemas: *PlanningHorizonDefinition* and *StandRegimeAction*. It also includes a function, *standHarvestingInfo*, which takes as input the standID, optionID and year number and has as output the mill to which the timber could be sent, the genetic material of the trees planted in the stand, the estimated mass of logs which will be obtained on felling the stand, the wood propertyID which the mill uses as an entry criterion for logs, and the average wood property value for the stand of logs. Each time there is a *fell* regime action stored in function *standRegimeAction*, there will be an associated entry in the function *standHarvestingInfo*. In addition, for each stand, option and year number, there is a single harvesting outcome which is stored.

<i>DefineStandHarvestingInfo</i>
\exists <i>PlanningHorizonDefinition</i>
\exists <i>StandRegimeAction</i>
<i>standHarvestingInfo</i> : (<i>STANDID</i> \times <i>OPTIONID</i> \times <i>YEARNO</i>) \rightarrow (<i>MILLID</i> \times <i>GENMATERIALID</i> \times <i>MASS</i> \times <i>WOODPROPID</i> \times <i>WOODPROPVAL</i>)
\forall <i>sID</i> : <i>STANDID</i> • \exists <i>oID</i> : <i>OPTIONID</i> • \exists <i>yearNo</i> : <i>YEARNO</i> • $1 \leq \text{yearNo} \leq \text{strategicHorizon?} \wedge$ $(sID, oID, yearNo) \in \text{dom } \text{standRegimeAction} \wedge$ $(sID, oID, yearNo) \in \text{dom } \text{standHarvestingInfo} \wedge$ $\text{fell} \in \{ \text{standRegimeAction } (sID, oID, yearNo) \} \wedge$ $\# \{ \text{standHarvestingInfo } (sID, oID, yearNo) \} = 1$

Schema *InitStandHarvestingInfo* initialises the function *standHarvestingInfo*. The changeable schema *DefineStandHarvestingInfo* is included. The function

standHarvestingInfo is initialised to be undefined.

<i>InitStandHarvestingInfo</i>
Δ <i>DefineStandHarvestingInfo</i>
<i>standHarvestingInfo'</i> = \emptyset

Schema *StandHarvestingInfo* is made up of the two schemas *DefineStandHarvestingInfo* and *InitStandHarvestingInfo*.

$$\text{StandHarvestingInfo} \hat{=} \text{DefineStandHarvestingInfo} \wedge \text{InitStandHarvestingInfo}$$

E.5.4 Populating the stand options

The two schemas *StandRegimeAction* and *StandHarvestingInfo* are populated next in schemas *GenerateStandRegimeActionForUnplantedStands* and *GenerateStandRegimeActionForPlantedStands* (which are combined to form schema *GenerateStandRegimeActionForStands*) and *UpdateHarvestingOutcomesUnplantedStands* and *UpdateHarvestingOutcomesPlantedStands* (which are combined in schema *UpdateHarvestingOutcomes*).

Schema *GenerateStandRegimeActionForUnplantedStands* calculates the stand options for the strategic planning horizon for stands which are unplanted at the start of the planning cycle. It includes unchangeable schemas *LogisticsChain* (defined in section D.3.2), *StandPlantingStatus* (defined in section D.4.3), *StoreRegimes* (defined in section D.2.2.8), *RegimeCanopyClosure* and *PlanningHorizonDefinition*, the changeable schemas *StandRegimeAction* and *NumberOfStandOptions*, and the calculation schema *Exponent*. It plants the stand in the first year of the planning horizon, and then for each year in the strategic planning horizon, it calculates when canopy closure would occur for that rotation (and subsequent rotations). It also calculates when each successive rotation will end (which is the same year that the stand will be replanted). The result is stored in the function *standRegimeAction*. In addition, the function *numberOfStandOptions* is updated to contain the number of options for the stand, which is the number of mills to which the stand's timber can be sent, to the power of the number of harvesting events which the stand will undergo in the strategic planning horizon, i.e. there would be $(\text{number of possible mills})^{(\text{strategicHorizon?}-1) \text{ div } \text{rotation age}}$ stand options. The new contents of the function *numberOfStandOptions* for that stand is the number of elements in the set of stand options.

Figure E.1 shows the stand options where there are two harvesting events (at 13 years and 25 years), and three possible mills to which the timber can be sent at harvesting.

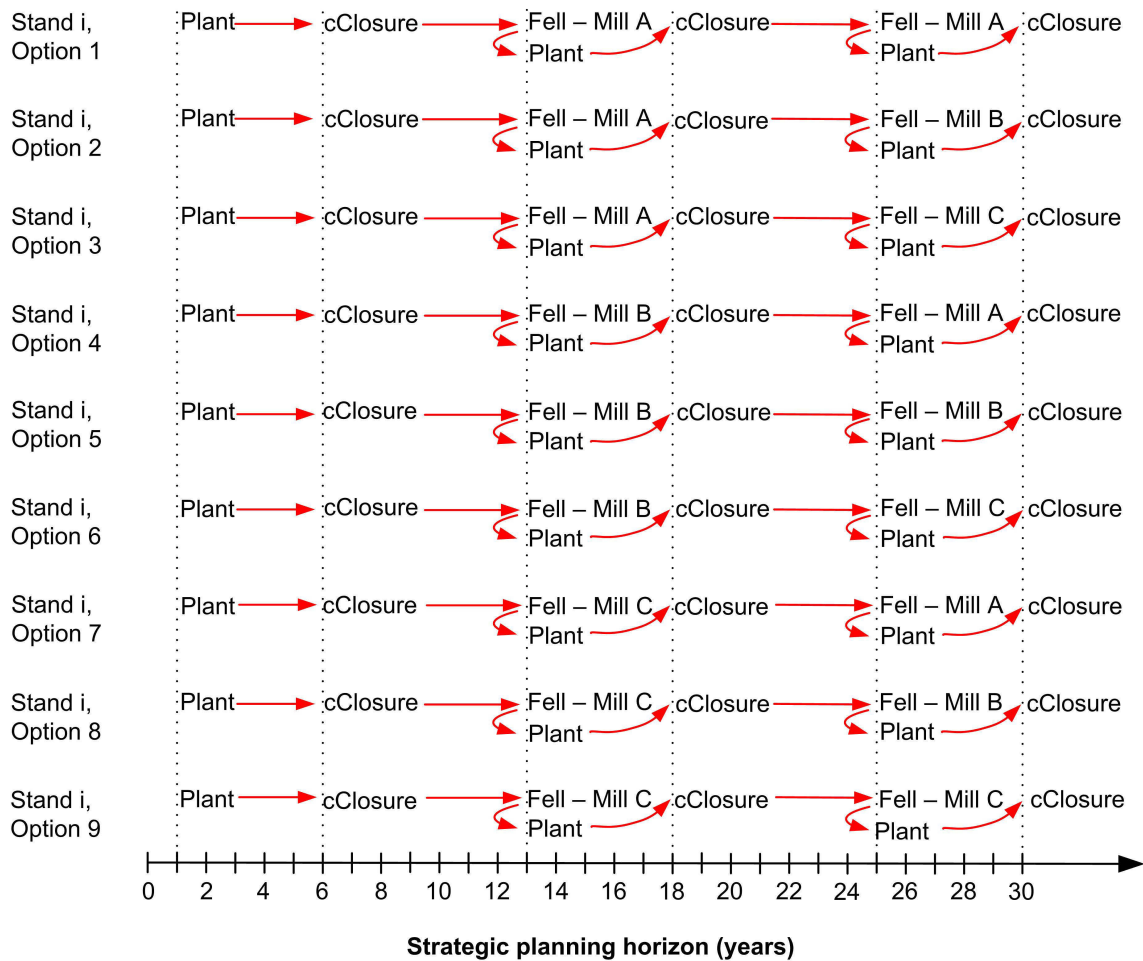


Figure E.1: Stand options for an unplanted stand, where there are three possible destination mills

Canopy closure occurs in the rotation's fifth year, i.e. in year 6 and year 20 of the strategic plan.

GenerateStandRegimeActionForUnplantedStands

\exists *LogisticsChain*
 \exists *StandPlantingStatus*
 \exists *StoreRegimes*
 \exists *RegimeCanopyClosure*
 \exists *PlanningHorizonDefinition*
 Δ *StandRegimeAction*
 Δ *NumberOfStandOptions*
Exponent

$\forall sID : STANDID \bullet$
 $\exists oID : OPTIONID \bullet$
 $\forall yearNo : YEARNO \bullet$
 $\exists pRegimeID : REGIMEID \bullet$
 $\exists genMatID : GENMATERIALID \bullet$
 $\exists mID : MILLID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $sID \in \text{dom } plantingStatus \wedge$
 $(sID, oID, yearNo) \in \text{dom } standRegimeAction \wedge$
 $sID \in \text{dom } standsRegimes \wedge$
 $pRegimeID = \text{first}(standsRegimes\ sID) \wedge$
 $(pRegimeID, planned) \in \text{dom } regimeGeneticMaterial \wedge$
 $regimeGeneticMaterial\ (pRegimeID, planned) = genMatID \wedge$
 $(pRegimeID, planned) \in \text{dom } fellAge \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom } regimeCanopyClosure \wedge$
 $sID \in \text{dom } sendLogsToDepot \wedge mID \in \text{ran } sendLogsToMill \wedge$
 $sID \in \text{dom } numberOfStandOptions \wedge$
 $(plantingStatus\ sID = unplanted) \Rightarrow$
 $(yearNo = 1 \Rightarrow$
 $(standRegimeAction' = standRegimeAction \cup \{(sID, oID, yearNo), plant\}) \wedge$
 $((yearNo - 1) \bmod fellAge\ (pRegimeID, planned)) =$
 $regimeCanopyClosure\ (pRegimeID, planned, genMatID) \Rightarrow$
 $(standRegimeAction' = standRegimeAction \cup$
 $\{(sID, oID, yearNo), cClosure\}) \wedge$
 $((yearNo - 1) \bmod fellAge\ (pRegimeID, planned)) = 0 \Rightarrow$
 $((standRegimeAction' = standRegimeAction \cup \{(sID, oID, yearNo), fell\}) \wedge$
 $(standRegimeAction' = standRegimeAction \cup \{(sID, oID, yearNo), plant\})) \wedge$
 $numberOfStandOptions' = numberOfStandOptions \oplus \{sID \mapsto$
 $exponent(\#\{sendLogsToMill\ (sendLogsToDepot\ sID)\},$
 $((strategicHorizon? - 1) \text{div } fellAge\ (pRegimeID, planned)))\} \wedge$
 $\#\{(sID, oID)\} = numberOfStandOptions'\ sID)$

Schema *GenerateStandRegimeActionForPlantedStands* calculates the stand options for the strategic planning horizon for stands which are planted at the start of the planning cycle. It includes unchangeable schemas *LogisticsChain*, (defined in section D.3.2), *StandPlantingStatus* (defined in section D.4.3), *StoreRegimes* (defined in section D.2.2.8), *RegimeCanopyClosure* and *PlanningHorizonDefinition*, the change-

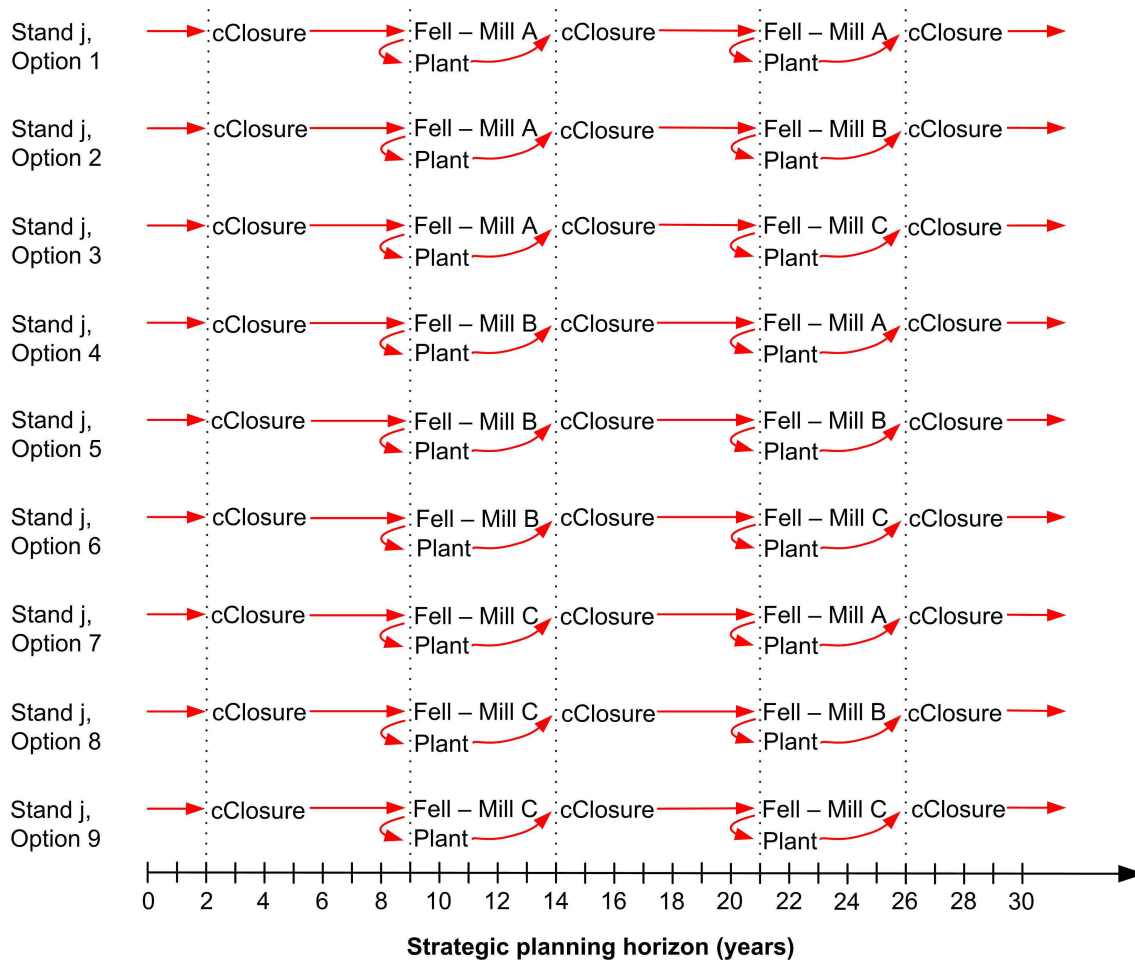


Figure E.2: Stand options for a planted stand, where there are three possible destination mills

able schemas *StandRegimeAction* and *NumberOfStandOptions*, and the two calculation schemas *CalculateAgeOfTreesInStand* and *Exponent*. As the stand is already planted, it determines if canopy closure has already occurred. If this is the case, it records canopy closure as having happened in year 1 of the planning horizon (this is needed to be able to calculate the costs of the regime’s maintenance, after canopy closure). Otherwise, the canopy closure is recorded in the year in which it will occur, as will the stand’s felling and planting. All of these regime actions are stored in function *standRegimeAction*. In addition, the function *numberOfStandOptions* is updated to contain the number of options for the stand, which is the number of mills to which the stand’s timber can be sent, to the power of the number of harvesting events which the stand will undergo in the strategic planning horizon, i.e. there would be $(\text{number of possible mills})^{(\text{strategicHorizon?}-1 + \text{tree age at start of plan}) \div \text{rotation age}}$ stand options.

GenerateStandRegimeActionForPlantedStands

\exists LogisticsChain
 \exists StandPlantingStatus
 \exists StoreRegimes
 \exists RegimeCanopyClosure
 \exists PlanningHorizonDefinition
 Δ StandRegimeAction
 Δ NumberOfStandOptions
 CalculateAgeOfTreesInStand
 Exponent

$\forall sID : STANDID \bullet$
 $\exists oID : OPTIONID \bullet$
 $\forall yearNo : YEARNO \bullet$
 $\exists pRegimeID : REGIMEID \bullet \exists aRegimeID : REGIMEID \bullet$
 $\exists genMatID : GENMATERIALID \bullet$
 $\exists mID : MILLID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $sID \in \text{dom } plantingStatus \wedge$
 $(sID, oID, yearNo) \in \text{dom } standRegimeAction \wedge$
 $sID \in \text{dom } standsRegimes \wedge$
 $pRegimeID = \text{first}(standsRegimes sID) \wedge aRegimeID = \text{second}(standsRegimes sID) \wedge$
 $(pRegimeID, planned) \in \text{dom } regimeGeneticMaterial \wedge$
 $regimeGeneticMaterial (pRegimeID, planned) = genMatID \wedge$
 $(pRegimeID, planned) \in \text{dom } fellAge \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom } regimeCanopyClosure \wedge$
 $sID \in \text{dom } sendLogsToDepot \wedge mID \in \text{ran } sendLogsToMill \wedge$
 $sID \in \text{dom } numberOfStandOptions \wedge$
 $(plantingStatus sID = planted) \Rightarrow$
 $((yearNo = 1) \wedge (treeAge(plantDate (aRegimeID, actual), dateToStartPlan?)) >$
 $regimeCanopyClosure (pRegimeID, planned, genMatID) \Rightarrow$
 $(standRegimeAction' = standRegimeAction \cup \{((sID, oID, 1), cClosure)\}) \wedge$
 $((yearNo - 1 + treeAge(plantDate (aRegimeID, actual), dateToStartPlan?))$
 $\text{mod } fellAge (pRegimeID, planned))$
 $= regimeCanopyClosure (pRegimeID, planned, genMatID) \Rightarrow$
 $(standRegimeAction' = standRegimeAction \cup$
 $\{((sID, oID, yearNo), cClosure)\}) \wedge$
 $((yearNo - 1 + treeAge(plantDate (aRegimeID, actual), dateToStartPlan?))$
 $\text{mod } fellAge (pRegimeID, planned)) = 0 \Rightarrow$
 $((standRegimeAction' = standRegimeAction \cup$
 $\{((sID, oID, yearNo), fell)\}) \wedge$
 $(standRegimeAction' = standRegimeAction \cup$
 $\{((sID, oID, yearNo), plant)\})) \wedge$
 $numberOfStandOptions' = numberOfStandOptions \oplus \{sID \mapsto$
 $exponent(\#\{sendLogsToMill (sendLogsToDepot sID)\},$
 $((strategicHorizon? - 1 +$
 $treeAge(plantDate (aRegimeID, actual), dateToStartPlan?))$
 $\text{div } fellAge (pRegimeID, planned)))\} \wedge$
 $\#\{(sID, oID, yearNo)\} = numberOfStandOptions' sID)$

The new contents of the function *numberOfStandOptions* for that stand is the number of elements in the set of stand options. Figure E.2 shows an example of a stand which has three possible destination mills. It was planted before the strategic planning horizon commenced.

The two schemas *GenerateStandRegimeActionForUnplantedStands* and *GenerateStandRegimeActionForPlantedStands* are combined in an overarching schema, *GenerateStandRegimeActionForStands*.

$$\begin{aligned} \text{GenerateStandRegimeActionForStands} &\hat{=} \\ &\text{GenerateStandRegimeActionForUnplantedStands} \wedge \\ &\text{GenerateStandRegimeActionForPlantedStands} \end{aligned}$$

Schema *UpdateHarvestingOutcomesUnplantedStands* updates the function *standHarvestingInfo* with information about the harvesting outcomes: for each stand and stand option, this function stores each of the possible mills the stand's timber could be sent to, the timber's genetic materialID and mass at harvesting, the ID of the wood property on which the mill evaluates incoming timber, and the stand's average wood property value.

The schema includes eight unchangeable schemas (*LogisticsChain* (defined in section D.3.2), *StoreRegimes* (defined in section D.2.2.8), *StandSiteIndex* (defined in section D.4.3), *StandCharacteristics* (defined in section D.4.2), *PlanningHorizonDefinition*, *EstimateStandWoodProp*, *MillRequirements* and *StandRegimeAction*), two calculation schemas (*EstimateStandsVolume* and *ConvertStandVolumeToMass*, both defined in section D.4.5) and one changeable schema (*StandHarvestingInfo*). The temporary variables *pRegimeID*, *genMatID* and *genID* are defined to store the contents of functions (planned regimeID, genetic materialID and genusID, respectively), which would have been more lengthy to write as function statements later in the schema. The genus of the stand's planned regime is found first of all, so as to obtain the site index's correct evaluation age in later calculations.

Each time the *standRegimeAction* function contains the regime action *fell*, an entry is added into the *standHarvestingInfo* function. The output of the function contains the details of the felling event, depending on when it occurs (i.e. rotation age), and to which mill the timber could be destined. This function is updated for all instances of felling events: whether the stand is currently planted or not. The exception is handled in the following schema, where the stand is currently planted and the first instance of the felling event needs to be changed to reflect the details of what is actually planted in the stand (which may differ from the plan). Subsequent rotations, though, are all planned, so this schema (*UpdateHarvestingOutcomesUnplantedStands*) caters for these felling events.

UpdateHarvestingOutcomesUnplantedStands

\exists *LogisticsChain*
 \exists *StoreRegimes*
 \exists *StandSiteIndex*
 \exists *StandCharacteristics*
EstimateStandsVolume
ConvertStandVolumeToMass
 \exists *PlanningHorizonDefinition*
 \exists *EstimateStandWoodProp*
 \exists *MillRequirements*
 \exists *StandRegimeAction*
 Δ *StandHarvestingInfo*

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \forall yearNo : YEARN0 \bullet$

$\forall mID : MILLID \bullet$

$\exists genMatID : GENMATERIALID \bullet$

$\exists_1 pRegimeID : REGIMEID \bullet$

$\exists_1 genID : GENUSID \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$sID \in \text{dom } sendLogsToDepot \wedge mID \in \text{ran } sendLogsToMill \wedge$

$mID = sendLogsToMill (sendLogsToDepot sID) \wedge$

$sID \in \text{dom } standsRegimes \wedge$

$first(standsRegimes sID) = pRegimeID \wedge$

$genMatID \in \text{ran } regimeGeneticMaterial \wedge$

$regimeGeneticMaterial (pRegimeID, planned) = genMatID \wedge$

$genID \in \text{dom } evalAge \wedge$

$(\{genusOfPureSpeciesWood (regimeGeneticMaterial (pRegimeID, planned))\} \neq \emptyset \Rightarrow$

$genID = genusOfPureSpeciesWood (regimeGeneticMaterial (pRegimeID, planned)) \vee$

$\{genusOfHybridWood (regimeGeneticMaterial (pRegimeID, planned))\} \neq \emptyset \Rightarrow$

$genID = genusOfHybridWood (regimeGeneticMaterial (pRegimeID, planned)) \wedge$

$sID \in \text{dom } standArea \wedge sID \in \text{dom } standAltitude \wedge$

$(sID, oID, yearNo) \in \text{dom } standRegimeAction \wedge$

$(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$

$(standRegimeAction (sID, oID, yearNo) = fell \Rightarrow$

$(standHarvestingInfo' = standHarvestingInfo \cup$

$\{(sID, oID, yearNo),$

$(mID, genMatID,$

$standMass (sID, genMatID, standVolume (sID, pRegimeID, genMatID,$

$siteIndex (sID, evalAge (genID)), fellAge (pRegimeID, planned),$

$plantedSPH (pRegimeID, planned), standArea sID)),$

$woodPropForMill mID,$

$standWoodProp (sID, woodPropForMill mID, genMatID,$

$siteIndex (sID, evalAge (genID)), fellAge (pRegimeID, planned),$

$standAltitude sID))))))$

UpdateHarvestingOutcomesPlantedStands

\exists *LogisticsChain*
 \exists *StandPlantingStatus*
 \exists *StoreRegimes*
 \exists *StandSiteIndex*
 \exists *StandCharacteristics*
EstimateStandsVolume
ConvertStandVolumeToMass
 \exists *PlanningHorizonDefinition*
 \exists *EstimateStandWoodProp*
 \exists *MillRequirements*
 \exists *StandRegimeAction*
 Δ *StandHarvestingInfo*

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \exists_1 yearNo1 : YEARNO \bullet \forall yearNo : YEARNO \bullet$
 $\forall mID : MILLID \bullet$
 $\exists genMatID : GENMATERIALID \bullet$
 $\exists_1 pRegimeID : REGIMEID \bullet \exists_1 aRegimeID : REGIMEID \bullet$
 $\exists_1 genID : GENUSID \bullet$
 $1 \leq yearNo1 \leq strategicHorizon? \wedge 1 \leq yearNo \leq strategicHorizon? \wedge$
 $sID \in \text{dom } sendLogsToDepot \wedge mID \in \text{ran } sendLogsToMill \wedge$
 $mID = sendLogsToMill (sendLogsToDepot sID) \wedge$
 $sID \in \text{dom } standsRegimes \wedge$
 $first(standsRegimes sID) = pRegimeID \wedge second(standsRegimes sID) = aRegimeID \wedge$
 $genMatID \in \text{ran } regimeGeneticMaterial \wedge$
 $regimeGeneticMaterial (aRegimeID, actual) = genMatID \wedge$
 $genID \in \text{dom } evalAge \wedge$
 $(\{genusOfPureSpeciesWood (regimeGeneticMaterial (aRegimeID, actual))\} \neq \emptyset \Rightarrow$
 $genID = genusOfPureSpeciesWood (regimeGeneticMaterial (aRegimeID, actual)) \vee$
 $\{genusOfHybridWood (regimeGeneticMaterial (aRegimeID, actual))\} \neq \emptyset \Rightarrow$
 $genID = genusOfHybridWood (regimeGeneticMaterial (aRegimeID, actual))) \wedge$
 $sID \in \text{dom } standArea \wedge sID \in \text{dom } standAltitude \wedge$
 $sID \in \text{dom } plantingStatus \wedge$
 $(sID, oID, yearNo1) \in \text{dom } standRegimeAction \wedge$
 $(sID, oID, yearNo) \in \text{dom } standRegimeAction \wedge$
 $yearNo1 = \min \{yearNo\} \wedge$
 $(sID, oID, yearNo1) \in \text{dom } standHarvestingInfo \wedge$
 $(plantingStatus sID = planted \wedge standRegimeAction (sID, oID, yearNo1) = fell) \Rightarrow$
 $(standHarvestingInfo' = standHarvestingInfo \oplus$
 $\{((sID, oID, yearNo1),$
 $(mID, genMatID,$
 $standMass (sID, genMatID, standVolume (sID, aRegimeID, genMatID,$
 $siteIndex (sID, evalAge (genID)), fellAge (pRegimeID, planned),$
 $plantedSPH (aRegimeID, actual), standArea sID)),$
 $woodPropForMill mID,$
 $standWoodProp (sID, woodPropForMill mID, genMatID,$
 $siteIndex (sID, evalAge (genID)),$
 $fellAge (pRegimeID, planned), standAltitude sID))))\}$

Schema *UpdateHarvestingOutcomesPlantedStands* is an exception to the schema *UpdateHarvestingOutcomesUnplantedStands*. If the stand is planted at the time of producing the strategic plan, the information for the first stand's harvest is overwritten (in function *standHarvestingInfo*) with the contents of the stand's actual details (actual planted density, actual genetic material planted). The fell age used for that rotation is still the planned one, as felling has not occurred yet.

The two schemas *UpdateHarvestingOutcomesUnplantedStands* and *UpdateHarvestingOutcomesPlantedStands* are combined in an overarching schema, *UpdateHarvestingOutcomes*.

$$UpdateHarvestingOutcomes \hat{=} UpdateHarvestingOutcomesUnplantedStands \wedge UpdateHarvestingOutcomesPlantedStands$$

E.6 Possible and feasible solutions

In the previous section, every stand options for each stand was defined. Only one of these stand options can be chosen for each stand, i.e. one cannot choose to send all the timber from a particular stand after a single felling event to both Mill A **and** Mill B. A possible solution is therefore defined to be the combination of all stand options so that each possible solution contains one stand option per stand, and all the stands have to be represented in each possible solution. Each possible solution can then be assessed to determine if it is feasible, i.e. determining if implementing that set of stand options would deliver the required mass of wood to each mill, and meet each mill's wood property constraints. Possible solutions which meet the mill constraints are feasible solutions. From the feasible solutions, the solution which gives the best profit can be determined: this is the optimal solution.

In this section, the possible solutions are calculated (section E.6.1), and each possible solution is examined to determine if it is feasible (section E.6.2). The profit (i.e. objective function) is calculated in section E.8, and the optimal solution is determined in section E.9.

E.6.1 Calculating the possible solutions

Before the possible solutions can be calculated, an identifier for each possible solution needs to be defined. Thereafter, the function which stores the possible solutions is defined, initialised and populated.

Type *POSSIBLESOLNID* defines the identifier of a possible solution (i.e. the set containing a set of stand options with one stand option per possible solution set).

[*POSSIBLESOLNID*]

Schema *DefinePossibleSolution* defines the function *possibleSolution* which has as input the possible solutionID and output a set of stand options. There are as many elements in the range of *possibleSolution* as there are stands in the plantation forest. In addition, a standID may not be present twice in one possible solution. (This means that one of the stand options has to be chosen for each possible solution, and therefore each optimal solution.)

Schema *InitPossibleSolution* initialises the function *possibleSolution* to be undefined. Schema *PossibleSolution* combines the two schemas *DefinePossibleSolution* and *InitPossibleSolution*.

DefinePossibleSolution

$possibleSolution : POSSIBLESOLNID \rightarrow (STANDID \times OPTIONID)$

$\forall pSID : POSSIBLESOLNID \bullet \forall sID : STANDID \bullet \forall oID : OPTIONID \bullet$
 $\quad \forall sID1 : STANDID \bullet \forall sID2 : STANDID \bullet$
 $\quad \forall pSID1 : POSSIBLESOLNID \bullet \forall pSID2 : POSSIBLESOLNID \bullet$
 $pSID \in \text{dom } possibleSolution \wedge$
 $(sID, oID) \in \text{ran } possibleSolution \wedge$
 $\#\{possibleSolution \ pSID\} = \#\{sID\} \wedge$
 $(pSID1 \in \text{dom } possibleSolution \wedge$
 $pSID2 \in \text{dom } possibleSolution \wedge$
 $(sID1, oID) \in \text{ran } possibleSolution \wedge$
 $(sID2, oID) \in \text{ran } possibleSolution \wedge$
 $sID1 = sID2 \iff pSID1 = pSID2)$

InitPossibleSolution

$\Delta DefinePossibleSolution$

$possibleSolution' = \emptyset$

$PossibleSolution \hat{=} DefinePossibleSolution \wedge InitPossibleSolution$

Schema *CalculatePossibleSolutions* updates the function *possibleSolution* with possible solutions. It includes the changeable schema *PossibleSolution*. If a standID is not already in a possible solution, it is added (with the optionID) to the range of it is added (with the optionID) to the range of *possibleSolution*, but only if the range of *possibleSolution* evaluated at any two distinct possible solutionIDs must be different.

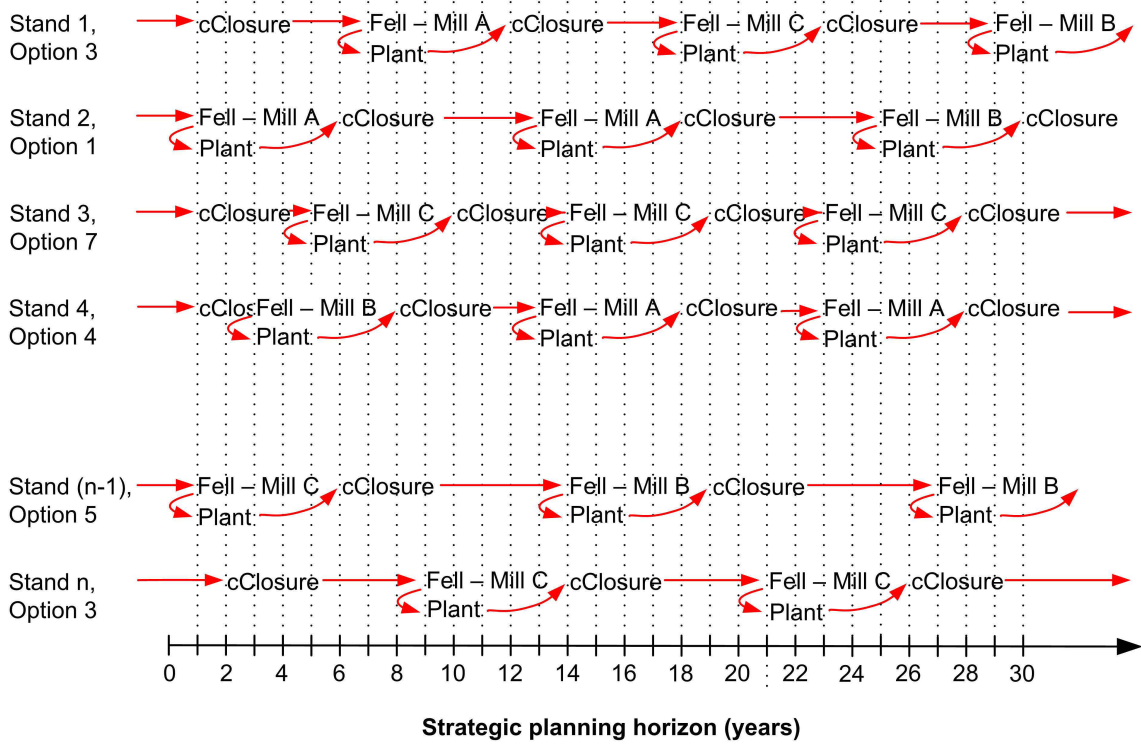


Figure E.3: A possible solution for the forest harvest scheduling system. Each stand is represented once in each possible solution.

Figure E.3 shows one possible solution for the forest harvest scheduling system. A stand may only appear in a possible solution once (i.e. it is not possible for a stand to have two different plans made for it).

The total number of possible solutions is calculated as the product of the number of stand options for each stand.

CalculatePossibleSolutions

Δ *PossibleSolution*

$\forall pSID : POSSIBLESOLNID \bullet$

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet$

$\forall pSID1 : POSSIBLESOLNID \bullet \forall pSID2 : POSSIBLESOLNID \bullet$

$pSID \in \text{dom possibleSolution} \wedge$

$pSID1 \in \text{dom possibleSolution} \wedge$

$pSID2 \in \text{dom possibleSolution} \wedge$

$\text{possibleSolution}' = \text{possibleSolution} \cup \{(pSID, (sID, oID))\} \wedge$

$sID \neq \text{first}(\text{possibleSolution } pSID) \Rightarrow$

$(\text{possibleSolution}' = \text{possibleSolution} \cup \{(pSID, (sID, oID))\}) \wedge$

$pSID1 \neq pSID2 \Rightarrow \text{possibleSolution } pSID1 \neq \text{possibleSolution } pSID2$

E.6.2 Calculating if the possible solutions are feasible

Having determined all the possible solutions in the previous section, the next step is to determine if each possible solution is feasible or infeasible. This means that each possible solution must be checked to see if it meets all the mill constraints (of required mass of logs, and wood properties) for each year in the strategic planning horizon. However, before this can be done, the mass of logs which each possible solution will deliver to each mill annually must be determined. After this has been calculated, the feasibility of each possible solution is determined.

Schema *DefineAnnualLogMassForMill* includes unchangeable schema *PlanningHorizonDefinition* and the function *annualLogMassForMill* whose inputs are the possible solutionID, the millID and year number, and has as output the mass of logs which will be delivered to a particular mill in a certain year, for a particular possible solution. The contents of the function are always greater than or equal to zero.

Schema *InitAnnualLogMassForMill* initialises the function *annualLogMassForMill* to be the empty set, while schema *SetAnnualLogMassForMillToZero* sets the value of the function *annualLogMassForMill* to zero. Schema *AnnualLogMassForMill* combines the schemas *DefineAnnualLogMassForMill*, *InitAnnualLogMassForMill* and *SetAnnualLogMassForMillToZero*.

DefineAnnualLogMassForMill

\exists *PlanningHorizonDefinition*

annualLogMassForMill : (*POSSIBLESOLNID* \times *MILLID* \times *YEARNO*) \rightarrow *MASS*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$
 $annualLogMassForMill (pSID, mID, yearNo) \geq 0$

InitAnnualLogMassForMill

Δ *DefineAnnualLogMassForMill*

annualLogMassForMill' = \emptyset

SetAnnualLogMassForMillToZero

\exists *PlanningHorizonDefinition*

Δ *InitAnnualLogMassForMill*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$
 $annualLogMassForMill' = annualLogMassForMill \cup \{((pSID, mID, yearNo), 0)\}$

$$\text{AnnualLogMassForMill} \hat{=} \text{DefineAnnualLogMassForMill} \wedge \\ \text{InitAnnualLogMassForMill} \wedge \text{SetAnnualLogMassForMillToZero}$$

Schema *CalculateAnnualLogMassForMill* updates the contents of the function *annualLogMassForMill*, which is the total mass of logs delivered to a particular mill in a particular year by a particular possible solution. This schema includes the unchangeable schemas *PlanningHorizonDefinition*, *StandHarvestingInfo* and *PossibleSolution* and the changeable schema *AnnualLogMassForMill*. For each mill and each year in the strategic planning horizon, the function *annualLogMassForMill* is updated with the mass of logs (*ThirdOf5* (*standHarvestingInfo* (*sID*, *oID*, *yearNo*))) destined for the mill in that year, for the possible solution.

$$\begin{array}{l} \text{CalculateAnnualLogMassForMill} \\ \exists \text{PlanningHorizonDefinition} \\ \exists \text{StandHarvestingInfo} \\ \exists \text{PossibleSolution} \\ \Delta \text{AnnualLogMassForMill} \\ \forall pSID : \text{POSSIBLESOLNID} \bullet \forall mID : \text{MILLID} \bullet \forall yearNo : \text{YEARNO} \bullet \\ \quad \forall sID : \text{STANDID} \bullet \forall oID : \text{OPTIONID} \bullet \\ \quad 1 \leq yearNo \leq \text{strategicHorizon?} \wedge \\ \quad (sID, oID, yearNo) \in \text{dom standHarvestingInfo} \wedge \\ \quad pSID \in \text{dom possibleSolution} \wedge \\ \quad (sID, oID) \in \text{ran possibleSolution} \wedge \\ \quad \text{possibleSolution } pSID = (sID, oID) \wedge \\ \quad (pSID, mID, yearNo) \in \text{dom annualLogMassForMill} \wedge \\ \quad mID = \text{FirstOf5} (\text{standHarvestingInfo} (sID, oID, yearNo)) \wedge \\ \quad \text{annualLogMassForMill}' = \text{annualLogMassForMill} \oplus \\ \quad \{(pSID, mID, yearNo) \mapsto \\ \quad \quad \text{annualLogMassForMill} (pSID, mID, yearNo) \\ \quad \quad + \text{ThirdOf5} (\text{standHarvestingInfo} (sID, oID, yearNo))\} \end{array}$$

Figure E.4 gives an example of a single possible solution and calculating the annual mass of logs which is destined for each mill for that possible solution. For each felling event, the mass of logs (which is stored in the third element of *standHarvestingInfo*) for that year and that mill are added up and stored in the function *annualLogMassForMill*. The dotted arrows downwards show when a felling event occurred. For each of these arrows, the mass of logs destined for each mill will be calculated. An example is given for the first year, where there are logs destined for Mills A and C, but none for Mill B.

The feasibility of each possible solution can now be calculated. Type *FEASIBILITY* is declared to tag whether each possible solution is *feasible* or *infeasible*. The outcome is stored in function *feasibleSolution* which is defined and initialised in schema *FeasibleSolution* and updated in schema *DetermineFeasibleSolutions*.

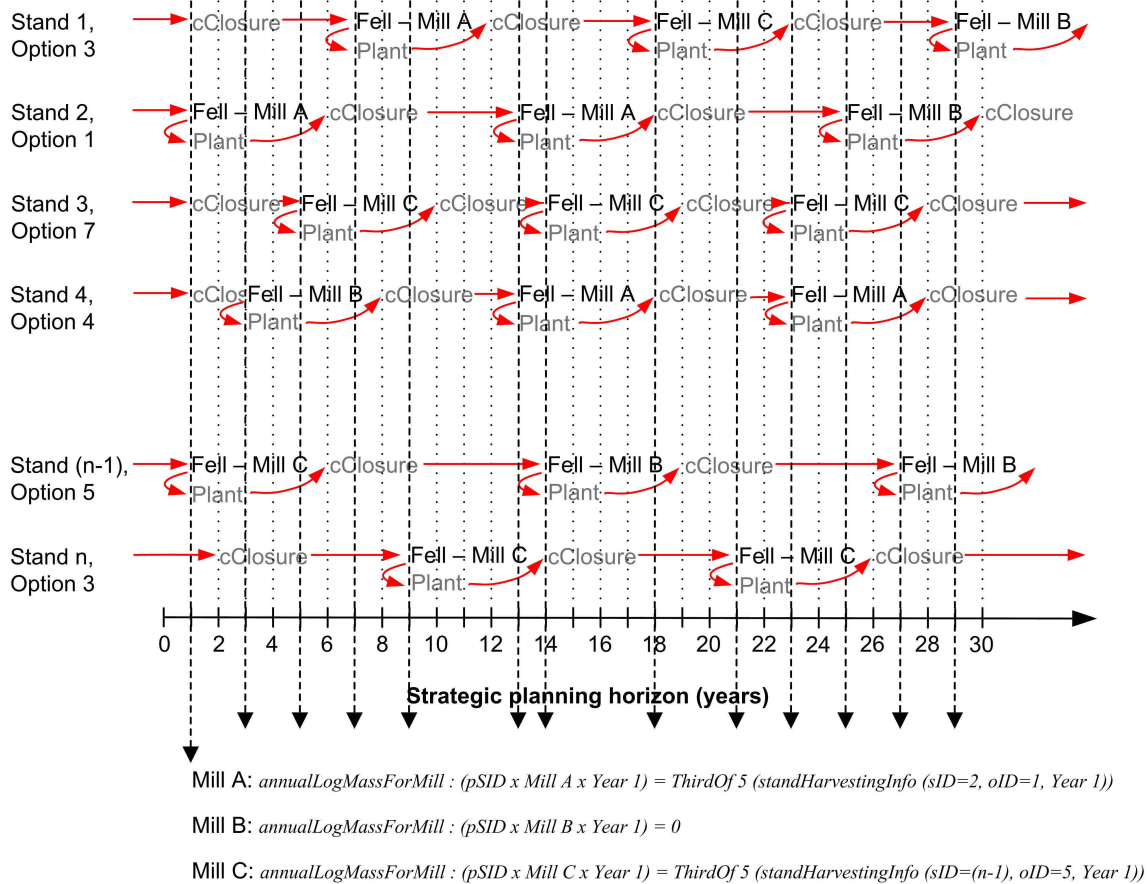


Figure E.4: Calculation of the annual mass of logs destined for each possible mill, for a particular possible solution. The mass of logs is calculated for each year when there is a felling event in one or more stands. The calculations for the first year are shown as an example

Type *FEASIBILITY* which can take on the values of *feasible* or *infeasible* is defined so that each possible solution can be assessed for feasibility.

$FEASIBILITY ::= feasible \mid infeasible$

Schema *DefineFeasibleSolution* includes the schema *PossibleSolution* and defines the function *feasibleSolution*, which has as input a possible solutionID, and output the type *FEASIBILITY*. Each possible solution is either feasible (i.e. it meets the constraints) or infeasible. The number of feasible solutions is a subset or equal to the number of possible solutions.

Schema *InitFeasibleSolution* sets the function *feasibleSolution* to the empty set. Schema *SetFeasibleSolutionToInfeasible* initialises the value of the function *feasibleSolution* to be infeasible. Schema *FeasibleSolution* combines the schemas

Define *FeasibleSolution*, *InitFeasibleSolution* and *SetFeasibleSolutionToInfeasible*.

DefineFeasibleSolution

Ξ *PossibleSolution*

feasibleSolution : *POSSIBLESOLNID* \rightarrow *FEASIBILITY*

$\forall pSID : POSSIBLESOLNID \bullet \exists_1 fS : FEASIBILITY \bullet$

$pSID \in \text{dom } possibleSolution \wedge$

$pSID \in \text{dom } feasibleSolution \wedge$

$fS \in \text{ran } feasibleSolution \wedge$

$feasibleSolution \ pSID = fS \wedge$

$\text{dom } feasibleSolution \subseteq \text{dom } possibleSolution$

InitFeasibleSolution

Δ *DefineFeasibleSolution*

feasibleSolution' = \emptyset

SetFeasibleSolutionToInfeasible

Δ *InitFeasibleSolution*

$\forall pSID : POSSIBLESOLNID \bullet$

$pSID \in \text{dom } feasibleSolution \wedge$

$feasibleSolution' = feasibleSolution \cup \{(pSID, infeasible)\}$

$FeasibleSolution \hat{=} DefineFeasibleSolution \wedge InitFeasibleSolution \wedge$
 $SetFeasibleSolutionToInfeasible$

Schema *DetermineFeasibleSolutions* determines which of the possible solutions are feasible solutions. A feasible solution is one which meets all the constraints (in this case, it meets the mill requirement constraints). Schema *DetermineFeasibleSolutions* includes five unchangeable schemas (*PlanningHorizonDefinition*, *StandHarvestingInfo*, *AnnualLogMassForMill*, *MillRequirements* and *PossibleSolution*) and one changeable schema (*FeasibleSolution*). For each possible solution, the schema checks whether the annual mill's requirements are met: for each year, the mass of logs delivered to the mill should be between the minimum and maximum required mass; the wood property value of each stand's timber should be between the minimum and maximum allowable values. If these constraints are all met, then the possible solution is deemed feasible, and the function *feasibleSolution*'s range for that possible solutionID is updated to become *feasible*. Possible solutions not meeting the constraints are deemed *infeasible*; there is no need to update this function with this information, as the function *feasibleSolution* was initialised as being *infeasible*.

DetermineFeasibleSolutions

\exists *PlanningHorizonDefinition*

\exists *StandHarvestingInfo*

\exists *AnnualLogMassForMill*

\exists *MillRequirements*

\exists *PossibleSolution*

Δ *FeasibleSolution*

$\forall pSID : POSSIBLESOLNID \bullet \forall yearNo : YEARNNO \bullet \forall mID : MILLID \bullet$
 $\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \exists woodPropID : WOODPROPID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$
 $(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$
 $pSID \in \text{dom } possibleSolution \wedge$
 $(sID, oID) \in \text{ran } possibleSolution \wedge$
 $pSID \in \text{dom } feasibleSolution \wedge$
 $(mID, yearNo) \in \text{dom } minMass \wedge (mID, yearNo) \in \text{dom } maxMass \wedge$
 $mID \in \text{dom } woodPropForMill \wedge woodPropID \in \text{ran } woodPropForMill \wedge$
 $woodPropForMill mID = woodPropID$
 $= FourthOf5 (standHarvestingInfo (sID, oID, yearNo)) \wedge$
 $(mID, woodPropID, yearNo) \in \text{dom } minWoodPropValue \wedge$
 $(mID, woodPropID, yearNo) \in \text{dom } maxWoodPropValue \wedge$
 $(minMass (mID, yearNo) \leq annualLogMassForMill (pSID, mID, yearNo)$
 $\leq maxMass (mID, yearNo) \wedge$
 $(minWoodPropValue (mID, woodPropID, yearNo)$
 $\leq FifthOf5 (standHarvestingInfo (sID, oID, yearNo))$
 $\leq maxWoodPropValue (mID, woodPropID, yearNo))) \Rightarrow$
 $(feasibleSolution' = feasibleSolution \oplus \{pSID \mapsto feasible\})$

E.6.3 Determining if one or more feasible solutions exist

Each possible solution has now been tagged as *feasible* or *infeasible*. To facilitate the running of the forest harvest scheduling system later (section E.10), a function (*feasibleSolutionsExist*) is updated with *feasibleSolnsExist* if one or more feasible solution exists, and *allSolnsInfeasible* if no possible solution is feasible.

The type *FEASIBLESOLN* has two options: *feasibleSolnsExist* or *allSolnsInfeasible*.

FEASIBLESOLN ::= feasibleSolnsExist | allSolnsInfeasible

Schema *DefineFeasibleSolutionsExist* defines a function *feasibleSolutionsExist* which is used to determine if there are any feasible solutions amongst the possible solutions. It is defined at a company-wide level (as the forest harvest scheduling system is solving a company-wide problem), and outputs the values *feasibleSolnsExist* or

allSolnsInfeasible.

Schema *InitFeasibleSolutionsExist* sets the function to be the empty set, while schema *DefineAllSolutionsInfeasible* initialises the output of this function to be *allSolnsInfeasible*. Schema *FeasibleSolutionsExist* combines the schemas *DefineFeasibleSolutionsExist*, *InitFeasibleSolutionsExist* and *DefineAllSolutionsInfeasible*.

$\begin{array}{l} \text{DefineFeasibleSolutionsExist} \\ \text{feasibleSolutionsExist} : \text{COMPANYID} \rightarrow \text{FEASIBLESOLN} \end{array}$
--

$\begin{array}{l} \text{InitFeasibleSolutionsExist} \\ \Delta \text{DefineFeasibleSolutionsExist} \\ \text{feasibleSolutionsExist}' = \emptyset \end{array}$
--

$\begin{array}{l} \text{DefineAllSolutionsInfeasible} \\ \Delta \text{InitFeasibleSolutionsExist} \\ \exists_1 cID : \text{COMPANYID} \bullet \\ \quad cID \in \text{dom feasibleSolutionsExist} \wedge \\ \quad \text{feasibleSolutionsExist}' = \text{feasibleSolutionsExist} \cup \{(cID, \text{allSolnsInfeasible})\} \end{array}$
--

$$\begin{aligned} \text{FeasibleSolutionsExist} &\hat{=} \text{DefineFeasibleSolutionsExist} \wedge \\ &\quad \text{InitFeasibleSolutionsExist} \wedge \text{DefineAllSolutionsInfeasible} \end{aligned}$$

Schema *DetermineIfFeasibleSolutionsExist* updates the function *feasibleSolutionsExist*. It includes the unchangeable schemas *PossibleSolution* and *FeasibleSolution* and the changeable schema *FeasibleSolutionsExist*. If any of the possible solutions are also feasible, then the value of the function *feasibleSolutionsExist* is changed to *feasibleSolnsExist*.

$\begin{array}{l} \text{DetermineIfFeasibleSolutionsExist} \\ \exists \text{PossibleSolution} \\ \exists \text{FeasibleSolution} \\ \Delta \text{FeasibleSolutionsExist} \\ \forall pSID : \text{POSSIBLESOLNID} \bullet \exists_1 cID : \text{COMPANYID} \bullet \\ \quad pSID \in \text{dom possibleSolution} \wedge \\ \quad pSID \in \text{dom feasibleSolution} \wedge \\ \quad cID \in \text{dom feasibleSolutionsExist} \wedge \\ \quad \text{feasibleSolution } pSID = \text{feasible} \Rightarrow \\ \quad \text{feasibleSolutionsExist}' = \text{feasibleSolutionsExist} \oplus \{cID \mapsto \text{feasibleSolnsExist}\} \end{array}$
--

E.7 Calculating the costs of each stand option

Each stand option contains a list of regime actions (plant, canopy closure is achieved, fell, plant, ...) for the planning horizon. This section calculates the cost of implementing a stand option. This means calculating the regime costs and the cost of transporting the logs to the mill in that stand option.

In order to calculate the regime costs for a stand option, the first regime option for that stand option needs to be determined. This will help to determine whether the stand is planted or not, or if the stand's canopy closure has been achieved or not. Based on this information, the regime's costs can be calculated (section E.7.2). The cost of transporting the timber to the mill in the stand option can then be calculated (see section E.7.3) and the total costs for each stand option calculated (see section E.7.4).

E.7.1 Determining the first regime action per stand option

In this section, the first regime action in the planning horizon for each stand option is determined. This is needed for determining the regime costs in section E.7.2. Before the first regime action can be determined, though, the function containing the result (*firstRegimeActionInStandOption*) needs to be defined and initialised.

Schema *DefineFirstRegimeActionInStandOption* includes the unchangeable schema *PlanningHorizonDefinition*, and defines a function, *firstRegimeActionInStandOption*, which takes as input the standID and optionID, and has as output the year in which the first regime action of the stand option will occur. For each stand option, there is only one output year.

In schema *InitFirstRegimeActionInStandOption*, the value of the function *firstRegimeActionInStandOption* is initialised to be undefined. Schema *FirstRegimeActionInStandOption* combines schemas *DefineFirstRegimeActionInStandOption* and *InitFirstRegimeActionInStandOption*.

<p><i>DefineFirstRegimeActionInStandOption</i> _____</p> <p>\exists <i>PlanningHorizonDefinition</i></p> <p><i>firstRegimeActionInStandOption</i> : (<i>STANDID</i> × <i>OPTIONID</i>) → <i>YEARNO</i></p> <hr/> <p>$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \exists_1 yearNo : YEARNO \bullet$ $1 \leq yearNo \leq strategicHorizon? \wedge$ $(sID, oID) \in \text{dom } firstRegimeActionInStandOption \wedge$ $yearNo \in \text{ran } firstRegimeActionInStandOption$</p>

$\frac{\text{InitFirstRegimeActionInStandOption}}{\exists \text{StandRegimeAction} \quad \Delta \text{DefineFirstRegimeActionInStandOption}}$
$\forall sID : \text{STANDID} \bullet \forall oID : \text{OPTIONID} \bullet$ $(sID, oID) \in \text{dom firstRegimeActionInStandOption} \wedge$ $\text{firstRegimeActionInStandOption}' = \emptyset$

$$\text{FirstRegimeActionInStandOption} \hat{=} \text{DefineFirstRegimeActionInStandOption} \wedge \text{InitFirstRegimeActionInStandOption}$$

Schema *DetermineFirstRegimeActionInStandOption* fills the function *firstRegimeActionInStandOption* with the appropriate contents. This schema includes the unchanged schemas *PlanningHorizonDefinition* and *StandRegimeAction*, and the changeable schema *FirstRegimeActionInStandOption*. For every stand option, there is a single year (*yearNo1*) which is the first year of the series of years when regime actions occur. This year then updated in the function *firstRegimeActionInStandOption*.

$\frac{\text{DetermineFirstRegimeActionInStandOption}}{\exists \text{PlanningHorizonDefinition} \quad \exists \text{StandRegimeAction} \quad \Delta \text{FirstRegimeActionInStandOption}}$
$\forall sID : \text{STANDID} \bullet \forall oID : \text{OPTIONID} \bullet \forall yearNo : \text{YEARNO} \bullet \exists_1 yearNo1 : \text{YEARNO} \bullet$ $1 \leq yearNo \leq \text{strategicHorizon?} \wedge$ $1 \leq yearNo1 \leq \text{strategicHorizon?} \wedge$ $(sID, oID, yearNo) \in \text{dom standRegimeAction} \wedge$ $(sID, oID) \in \text{dom firstRegimeActionInStandOption} \wedge$ $yearNo1 \in \text{ran firstRegimeActionInStandOption} \wedge$ $\text{firstRegimeActionInStandOption} (sID, oID) = yearNo1 \wedge$ $yearNo1 = \min\{\text{ThirdOf3}(sID, oID, yearNo)\} \wedge$ $\text{firstRegimeActionInStandOption}' = \text{firstRegimeActionInStandOption} \oplus$ $\{(sID, oID) \mapsto yearNo1\}$

E.7.2 Calculating regime costs for each stand option

The costs of felling and planting, and the maintenance costs incurred before and after canopy closure are calculated in this section. From the previous section, the first regime action for each stand option is known. This can be used to determine if the stand is planted or not, and whether canopy closure has already occurred.

In the determination of the regime costs for each stand option, the contents of the function *standRegimeAction* will be traversed for all years in the strategic planning horizon to calculate the regime costs. Certain variables (*yearOfPlanting*,

yearOfCClosure, *yearOfFelling*) are need to keep track of the year of planting, canopy closure and felling for a particular regime. Another variable (*firstRotation*) is needed to keep track of whether the rotation is the first one or not. If it is the first, then the genetic material of the trees (defined and stored in the variable *genMatID*) will be determined from the actual regime; for all other cases, it will be determined from the planned regime. Finally, the function which will contain the regime costs for the stand options (*regimeStandOptionCosts*) is declared and initialised (in schema *RegimeStandOptionCosts*). The regime costs are calculated and assigned to the function *regimeStandOptionCosts* in schema *CalculateRegimeStandOptionCosts*.

Schema *DefineYearVariables* contains three variable declarations: *yearOfPlanting*, *yearOfCClosure* and *yearOfFelling*. These variables (*yearOfPlanting*, *yearOfCClosure*, *yearOfFelling*) are need to keep track of the year of planting, canopy closure and felling for a particular regime. Schema *InitYearVariables* initialises these variables to the value 1, and schema *YearVariables* combines the two schemas *DefineYearVariables* and *InitYearVariables*.

<i>DefineYearVariables</i> <i>yearOfPlanting</i> , <i>yearOfCClosure</i> , <i>yearOfFelling</i> : YEARNO

<i>InitYearVariables</i> Δ <i>DefineYearVariables</i> <i>yearOfPlanting'</i> = 1 <i>yearOfCClosure'</i> = 1 <i>yearOfFelling'</i> = 1
--

$YearVariables \hat{=} DefineYearVariables \wedge InitYearVariables$

A boolean type, *FIRSTROTATION*, is declared to detect whether a stand's first rotation was underway when the planning cycle started. (If so, the actual regime's information is used to calculate the regime costs.)

FIRSTROTATION ::= *yes* | *no*

Schema *DefineFirstRotation* contains the declaration of the variable *firstRotation*. This variable (*firstRotation*) is needed to keep track of whether the rotation is the first one or not. Schema *InitFirstRotation* initialises this variable to *no*, and schema *FirstRotation* combines the two schemas *DefineFirstRotation* and *InitFirstRotation*.

<i>DefineFirstRotation</i> <i>firstRotation</i> : <i>FIRSTROTATION</i>

<i>InitFirstRotation</i>
$\Delta DefineFirstRotation$
$firstRotation' = no$

$$FirstRotation \hat{=} DefineFirstRotation \wedge InitFirstRotation$$

Schema *DefineGenMatID* contains the declaration of the variable *genMatID*. This variable is used to store the genetic material of each rotation while the contents of function *standRegimeAction* are being traversed. If the variable *firstRotation* (defined above) is *yes*, *genMatID* will contain the actual regime's genetic materialID, if that rotation is already planted. Otherwise, it will contain the planned regime's genetic materialID. Schema *InitGenMatID* initialises variable *genMatID* to be undefined, and schema *GenMatID* combines the two schemas *DefineGenMatID* and *InitGenMatID*.

<i>DefineGenMatID</i>
$genMatID : GENMATERIALID$

<i>InitGenMatID</i>
$\Delta DefineGenMatID$
$\{genMatID'\} = \emptyset$

$$GenMatID \hat{=} DefineGenMatID \wedge InitGenMatID$$

The function which will contain the regime costs for the stand options (*regimeStandOptionCosts*) is declared and initialised below. The function's contents will be updated in schema *CalculateRegimeStandOptionCosts*.

Schema *DefineRegimeStandOptionCosts* includes the unchangeable schema *PlanningHorizonDefinition* and a function, *regimeStandOptionCosts*, which stores the regime costs incurred should a particular stand option be undertaken. The function contents are always greater than, or equal to, zero.

<i>DefineRegimeStandOptionCosts</i>
$regimeStandOptionCosts : (STANDID \times OPTIONID) \rightarrow COST$
$\forall sID : STANDID \bullet$
$\forall oID : OPTIONID \bullet$
$(sID, oID) \in \text{dom } regimeStandOptionCosts \wedge$
$regimeStandOptionCosts (sID, oID) \geq 0$

Schema *InitRegimeStandOptionCosts* includes the unchangeable schema *DefineRegimeStandOptionCosts*. Every regime stand option set to the empty set. In schema *SetRegimeStandOptionCostsToZero*, the contents of the function *SetRegimeStandOptionCostsToZero* are set to zero.

<i>InitRegimeStandOptionCosts</i>
Δ <i>DefineRegimeStandOptionCosts</i>
$regimeStandOptionCosts' = \emptyset$

<i>SetRegimeStandOptionCostsToZero</i>
Δ <i>InitRegimeStandOptionCosts</i>
$\forall sID : STANDID \bullet$ $\quad \forall oID : OPTIONID \bullet$ $\quad (sID, oID) \in \text{dom } regimeStandOptionCosts \wedge$ $\quad regimeStandOptionCosts' = regimeStandOptionCosts \cup \{(sID, oID), 0\}$

Schema *RegimeStandOptionCosts* is composed by combining the schemas *DefineRegimeStandOptionCosts*, *InitRegimeStandOptionCosts* and *SetRegimeStandOptionCostsToZero*.

$$RegimeStandOptionCosts \cong DefineRegimeStandOptionCosts \wedge \\ InitRegimeStandOptionCosts \wedge SetRegimeStandOptionCostsToZero$$

Schema *CalculateRegimeStandOptionCosts* calculates the cost of implementing each regime for each stand option. This schema includes five unchanged schemas (*StoreRegimes* (defined in section D.2.2.8), *PlanningHorizonDefinition*, *RegimeCosts*, *StandRegimeAction* and *StandHarvestingInfo*), and four changeable schemas (*RegimeStandOptionCosts*, *YearVariables*, *FirstRotation* and *GenMatID*).

For each stand option, the first rotation action is determined through the function *firstRegimeActionInStandOption*, which contains the year of the first regime action for that stand option. If planting occurs in year 1, this means that all of the rotations for that stand option will be planted according to planned regimes (and no actual regime will exist; thus the regime's planned genetic materialID is used). However, if there is a regime action which takes place earlier than planting, (e.g. canopy closure, or felling), it means that an actual regime exists for the first rotation, and the second rotation can use the planned regime after the existing trees have been felled. In this case, the genetic materialID used for the first rotation is that of the actual regime, while that used for the second rotation is the planned regime's genetic materialID.

CalculateRegimeStandOptionCosts

\exists StoreRegimes
 \exists PlanningHorizonDefinition
 \exists RegimeCosts
 \exists StandRegimeAction
 \exists FirstRegimeActionInStandOption
 Δ RegimeStandOptionCosts
 Δ YearVariables
 Δ FirstRotation
 Δ GenMatID

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \forall yearNo : YEARNO \bullet \exists_1 yearNo1 : YEARNO \bullet$
 $\exists_1 pRegimeID : REGIMEID \bullet \exists_1 aRegimeID : REGIMEID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge 1 \leq yearNo1 \leq strategicHorizon? \wedge$
 $sID \in \text{dom standsRegimes} \wedge$
 $first(standsRegimes\ sID) = pRegimeID \wedge second(standsRegimes\ sID) = aRegimeID \wedge$
 $(pRegimeID, planned) \in \text{dom regimeGeneticMaterial} \wedge$
 $(aRegimeID, actual) \in \text{dom regimeGeneticMaterial} \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom regimePlantingCosts} \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom annualRegimeMaintCostBeforeCClosure} \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom annualRegimeMaintCostAfterCClosure} \wedge$
 $(pRegimeID, planned, genMatID) \in \text{dom regimeHarvestingCosts} \wedge$
 $(sID, oID, yearNo) \in \text{dom standRegimeAction} \wedge$
 $(sID, oID) \in \text{dom firstRegimeActionInStandOption} \wedge$
 $yearNo1 \in \text{ran firstRegimeActionInStandOption} \wedge$
 $firstRegimeActionInStandOption\ (sID, oID) = yearNo1 \wedge$
 $(sID, oID) \in \text{dom regimeStandOptionCosts} \wedge$
 $standRegimeAction\ (sID, oID, yearNo1) \neq plant \Rightarrow$
 $(firstRotation' = yes \wedge genMatID' = regimeGeneticMaterial\ (aRegimeID, actual)) \wedge$
 $standRegimeAction\ (sID, oID, yearNo1) = plant \Rightarrow$
 $(firstRotation' = no \wedge genMatID' = regimeGeneticMaterial\ (pRegimeID, planned)) \wedge$
 $standRegimeAction\ (sID, oID, yearNo) = plant \Rightarrow$
 $(yearOfPlanting' = yearNo \wedge$
 $regimeStandOptionCosts' = regimeStandOptionCosts \oplus \{(sID, oID) \mapsto$
 $regimeStandOptionCosts\ (sID, oID)$
 $+ regimePlantingCosts\ (pRegimeID, planned, genMatID)\}) \wedge$
 $standRegimeAction\ (sID, oID, yearNo) = cClosure \Rightarrow$
 $(yearOfCClosure' = yearNo \wedge$
 $yearOfPlanting \leq yearNo < yearOfCClosure \Rightarrow$
 $regimeStandOptionCosts' = regimeStandOptionCosts \oplus \{(sID, oID) \mapsto$
 $regimeStandOptionCosts\ (sID, oID)$
 $+ annualRegimeMaintCostBeforeCClosure\ (pRegimeID, planned, genMatID)\}) \wedge$
 $standRegimeAction\ (sID, oID, yearNo) = fell \Rightarrow$
 $(yearOfFelling' = yearNo \wedge$
 $regimeStandOptionCosts' = regimeStandOptionCosts \oplus \{(sID, oID) \mapsto$
 $regimeStandOptionCosts\ (sID, oID)$
 $+ regimeHarvestingCosts\ (pRegimeID, planned, genMatID)\}) \wedge$
 $yearOfCClosure \leq yearNo \leq yearOfFelling \Rightarrow$
 $regimeStandOptionCosts' = regimeStandOptionCosts \oplus \{(sID, oID) \mapsto$
 $regimeStandOptionCosts\ (sID, oID)$
 $+ annualRegimeMaintCostAfterCClosure\ (pRegimeID, planned, genMatID)\}) \wedge$
 $genMatID' = regimeGeneticMaterial\ (pRegimeID, planned)$

If the stand's regime action is *plant*, then the cost of planting will be added to the

function *regimeStandOptionCosts*. If the stand's regime action is *cClosure* (canopy closure has been reached), then for each year from the planting year to the year in which canopy closure is reached, the annual maintenance cost is added to the function *regimeStandOptionCosts*. If the stand's regime option is *fell*, then the cost of harvesting the stand is added to the function *regimeStandOptionCosts*, and for each year from the year that the canopy closure occurred to the year of harvesting, the maintenance cost is also added to the function *regimeStandOptionCosts*.

E.7.3 Calculating transport costs for each stand option

In the previous section, the costs of growing the trees for a particular stand option (i.e. over the strategic planning horizon) were calculated. This section calculates the cost of transporting the timber from the stand to the mill in that stand option (stored in function *StandHarvestingInfo*).

This calculation takes place in three parts: first the short-haul transport cost is defined, initialised and calculated. Next, the long-haul transport cost is defined, initialised and calculated. Finally, the total transport costs for each stand option are calculated by summing the short- and long-haul transport costs.

Schema *DefineSHaulTransportStandOptionCosts* defines the function *sHaulTransportStandOptionCosts* which stores the short-haul transport costs for a particular harvesting event, which will take place at a particular stand in a particular year in the strategic planning horizon.

Schema *InitSHaulTransportStandOptionCosts* initialises each value of the function *sHaulTransportStandOptionCosts* to the empty set, and schema *SetSHaulTransportStandOptionCostsToZero* initialises the function to zero.

Schema *SHaulTransportStandOptionCosts* combines the schemas *DefineSHaulTransportStandOptionCosts*, *InitSHaulTransportStandOptionCosts* and *SetSHaulTransportStandOptionCostsToZero* to form one schema.

DefineSHaulTransportStandOptionCosts _____

Ξ *PlanningHorizonDefinition*

$sHaulTransportStandOptionCosts : (STANDID \times OPTIONID \times YEARNNO) \rightarrow COST$

$\forall sID : STANDID \bullet$

$\forall oID : OPTIONID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(sID, oID, yearNo) \in \text{dom } sHaulTransportStandOptionCosts \wedge$

$sHaulTransportStandOptionCosts (sID, oID, yearNo) \geq 0$

InitSHaulTransportStandOptionCosts _____

Δ *DefineSHaulTransportStandOptionCosts*

$sHaulTransportStandOptionCosts' = \emptyset$

SetSHaulTransportStandOptionCostsToZero _____

Ξ *PlanningHorizonDefinition*

Δ *InitSHaulTransportStandOptionCosts*

$\forall sID : STANDID \bullet$

$\forall oID : OPTIONID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(sID, oID, yearNo) \in \text{dom } sHaulTransportStandOptionCosts \wedge$

$sHaulTransportStandOptionCosts' = sHaulTransportStandOptionCosts \cup \{((sID, oID, yearNo), 0)\}$

$sHaulTransportStandOptionCosts \hat{=} \text{DefineSHaulTransportStandOptionCosts} \wedge \text{InitSHaulTransportStandOptionCosts} \wedge \text{SetSHaulTransportStandOptionCostsToZero}$

Schema *CalculateSHaulTransportStandOptionCosts* calculates the cost of the short-haul transport for each harvesting event, for each stand option. It includes the unchangeable schemas *LogisticsChain* (defined in section D.3.2), *ShortHaulRoad* (defined in section D.5.2), *PlanningHorizonDefinition* and *StandHarvestingInfo* and the changeable schema *SHaulTransportStandOptionCosts*. For every stand option, there is a depot for which the stand's logs are destined (to be sent to the mill, whose ID is stored in the first element of the output tuple of function *standHarvestingInfo*). The short-haul transport costs (stored in *sHaulTransportStandOptionCosts*) are calculated as being the products of the mass of logs harvested at the stand, the distance between the stand and the depot, and the short-haul transport cost rate.

CalculateSHaulTransportStandOptionCosts

\exists *LogisticsChain*

\exists *ShortHaulRoad*

\exists *PlanningHorizonDefinition*

\exists *StandHarvestingInfo*

Δ *SHaulTransportStandOptionCosts*

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \forall yearNo : YEARN0 \bullet$

$\exists_1 dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$

$\exists_1 sHaulRoadID : ROADID \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$sID \in \text{dom } sendLogsToDepot \wedge$

$dID \in \text{ran } sendLogsToDepot \wedge$

$dID \in \text{dom } sendLogsToMill \wedge$

$mID \in \text{ran } sendLogsToMill \wedge$

$(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$

$mID = FirstOf5 (standHarvestingInfo (sID, oID, yearNo)) \wedge$

$(sID, dID, sHaulRoadID) \in \text{dom } shortHaulDistance \wedge$

$(sID, oID, yearNo) \in \text{dom } sHaulTransportStandOptionCosts \wedge$

$sHaulTransportStandOptionCosts' = sHaulTransportStandOptionCosts \oplus$

$\{(sID, oID, yearNo) \mapsto sHaulTransportStandOptionCosts (sID, oID, yearNo)$

$+ ThirdOf5 (standHarvestingInfo (sID, oID, yearNo))$

$* shortHaulDistance (sID, dID, sHaulRoadID) * shortHaulTripRate\}$

Schema *DefineLHaulTransportStandOptionCosts* defines the function *lHaulTransportStandOptionCosts* which stores the long-haul transport costs for a particular harvesting event, which will take place at a particular stand in a particular year in the strategic planning horizon.

Schema *InitLHaulTransportStandOptionCosts* initialises each value of the function *lHaulTransportStandOptionCosts* to the empty set and schema *SetLHaulTransportStandOptionCostsToZero* sets the contents of this function to zero.

Schema *LHaulTransportStandOptionCosts* combines the schemas *DefineLHaulTransportStandOptionCosts*, *InitLHaulTransportStandOptionCosts* and *SetLHaulTransportStandOptionCostsToZero* to form one schema.

DefineLHaulTransportStandOptionCosts _____

\exists *PlanningHorizonDefinition*

$lHaulTransportStandOptionCosts : (STANDID \times OPTIONID \times YEARNNO) \rightarrow COST$

$\forall sID : STANDID \bullet$

$\forall oID : OPTIONID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(sID, oID, yearNo) \in \text{dom } lHaulTransportStandOptionCosts \wedge$

$lHaulTransportStandOptionCosts (sID, oID, yearNo) \geq 0$

InitLHaulTransportStandOptionCosts _____

Δ *DefineLHaulTransportStandOptionCosts*

$lHaulTransportStandOptionCosts' = \emptyset$

SetLHaulTransportStandOptionCostsToZero _____

\exists *PlanningHorizonDefinition*

Δ *InitLHaulTransportStandOptionCosts*

$\forall sID : STANDID \bullet$

$\forall oID : OPTIONID \bullet$

$\forall yearNo : YEARNNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(sID, oID, yearNo) \in \text{dom } lHaulTransportStandOptionCosts \wedge$

$lHaulTransportStandOptionCosts' = lHaulTransportStandOptionCosts \cup \{((sID, oID, yearNo), 0)\}$

$LHaulTransportStandOptionCosts \hat{=} DefineLHaulTransportStandOptionCosts \wedge$
 $InitLHaulTransportStandOptionCosts \wedge SetLHaulTransportStandOptionCostsToZero$

Schema *CalculateLHaulTransportStandOptionCosts* calculates the cost of the long-haul transport for each harvesting event, for each stand option. It includes the unchangeable schemas *LogisticsChain* (defined in section D.3.2), *LongHaulRoad* (defined in section D.5.2), *PlanningHorizonDefinition* and *StandHarvestingInfo* and the changeable schema *LHaulTransportStandOptionCosts*. For every stand option, there is a depot for which the stand's logs are destined (to be sent to the mill, whose ID is stored in the first element of the output tuple of function *standHarvestingInfo*). The long-haul transport costs (stored in *lHaulTransportStandOptionCosts*) are calculated as being the products of the mass of logs harvested at the stand, the distance between the depot and the mill, and the long-haul transport cost rate.

CalculateLHaulTransportStandOptionCosts _____

\exists *LogisticsChain*

\exists *LongHaulRoad*

\exists *PlanningHorizonDefinition*

\exists *StandHarvestingInfo*

Δ *LHaulTransportStandOptionCosts*

$\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet \forall yearNo : YEARN0 \bullet$

$\exists_1 dID : DEPOTID \bullet \exists_1 mID : MILLID \bullet$

$\exists_1 lHaulRoadID : ROADID \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$sID \in \text{dom } sendLogsToDepot \wedge$

$dID \in \text{ran } sendLogsToDepot \wedge$

$dID \in \text{dom } sendLogsToMill \wedge$

$mID \in \text{ran } sendLogsToMill \wedge$

$(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$

$mID = FirstOf5 (standHarvestingInfo (sID, oID, yearNo)) \wedge$

$(dID, mID, lHaulRoadID) \in \text{dom } longHaulDistance \wedge$

$(sID, oID, yearNo) \in \text{dom } lHaulTransportStandOptionCosts \wedge$

$lHaulTransportStandOptionCosts' = lHaulTransportStandOptionCosts \oplus$

$\{(sID, oID, yearNo) \mapsto lHaulTransportStandOptionCosts (sID, oID, yearNo)$

$+ ThirdOf5 (standHarvestingInfo (sID, oID, yearNo))$

$* longHaulDistance (dID, mID, lHaulRoadID) * longHaulTripRate\}$

Schema *DefineTransportStandOptionCosts* includes the unchangeable schema *PlanningHorizonDefinition* and a function, *transportStandOptionCosts*, which stores the transport costs incurred should a particular stand option be undertaken. The function contents are always greater than, or equal to, zero.

DefineTransportStandOptionCosts _____

transportStandOptionCosts : $(STANDID \times OPTIONID) \rightarrow COST$

$\forall sID : STANDID \bullet$

$\forall oID : OPTIONID \bullet$

$(sID, oID) \in \text{dom } transportStandOptionCosts \wedge$

$transportStandOptionCosts (sID, oID) \geq 0$

Schema *InitTransportStandOptionCosts* includes the unchangeable schema *DefineTransportStandOptionCosts*. In this schema, the function *transportStandOptionCosts* is set to the empty set. In schema *SetTransportStandOptionCostsToZero*, every transport stand option cost is initialised to zero.

InitTransportStandOptionCosts _____

Δ *DefineTransportStandOptionCosts*

transportStandOptionCosts' = \emptyset

$$\begin{array}{l}
\text{SetTransportStandOptionCostsToZero} \\
\Delta \text{InitTransportStandOptionCosts} \\
\hline
\forall sID : \text{STANDID} \bullet \\
\quad \forall oID : \text{OPTIONID} \bullet \\
\quad (sID, oID) \in \text{dom transportStandOptionCosts} \wedge \\
\quad \text{transportStandOptionCosts}' = \text{transportStandOptionCosts} \cup \{(sID, oID), 0\}
\end{array}$$

Schema *TransportStandOptionCosts* is composed by combining the schemas *DefineTransportStandOptionCosts*, *InitTransportStandOptionCosts* and *SetTransportStandOptionCostsToZero*.

$$\text{TransportStandOptionCosts} \hat{=} \text{DefineTransportStandOptionCosts} \wedge \\
\text{InitTransportStandOptionCosts} \wedge \text{SetTransportStandOptionCostsToZero}$$

Schema *CalculateTransportStandOptionCosts* calculates the transport costs for a stand option from the short-haul and log-haul transport costs stored in functions *sHaulTransportStandOptionCosts* and *lHaulTransportStandOptionCosts*. This schema includes three unchangeable schemas (*PlanningHorizonDefinition*, *SHaulTransportStandOptionCosts* and *HaulTransportStandOptionCosts*). For each year in the strategic planning horizon, each stand option's transport costs are calculated by adding them for each harvesting event.

$$\begin{array}{l}
\text{CalculateTransportStandOptionCosts} \\
\exists \text{PlanningHorizonDefinition} \\
\exists \text{SHaulTransportStandOptionCosts} \\
\exists \text{LHaulTransportStandOptionCosts} \\
\Delta \text{TransportStandOptionCosts} \\
\hline
\forall sID : \text{STANDID} \bullet \forall oID : \text{OPTIONID} \bullet \forall yearNo : \text{YEARNO} \bullet \\
\quad 1 \leq yearNo \leq \text{strategicHorizon?} \wedge \\
\quad (sID, oID, yearNo) \in \text{dom sHaulTransportStandOptionCosts} \wedge \\
\quad (sID, oID, yearNo) \in \text{dom lHaulTransportStandOptionCosts} \wedge \\
\quad (sID, oID) \in \text{dom transportStandOptionCosts} \wedge \\
\quad \text{transportStandOptionCosts}' = \text{transportStandOptionCosts} \oplus \\
\quad \quad \{(sID, oID) \mapsto \text{transportStandOptionCosts} (sID, oID) \\
\quad \quad + \text{sHaulTransportStandOptionCosts} (sID, oID, yearNo) \\
\quad \quad + \text{lHaulTransportStandOptionCosts} (sID, oID, yearNo)\}
\end{array}$$

E.7.4 Calculating total costs for each stand option

In the previous two sections, the costs of growing the trees and transporting them to the appropriate mill were calculated. In this section, these two values are summed to achieve a cost per stand option (stored in function *standOptionCosts*). These

stand option costs are used in section E.8.1.2 to calculate the cost for a possible solution.

Schema *DefineStandOptionCosts* includes the unchangeable schema *PlanningHorizonDefinition* and a function, *standOptionCosts*, which stores the costs incurred should a particular stand option be undertaken. This includes the cost of growing the trees (regime costs) and the cost of delivering the logs grown at that stand to the depot, and then to the mill. The function contents are always greater than, or equal to, zero.

$\begin{array}{l} \textit{DefineStandOptionCosts} \\ \textit{standOptionCosts} : (\textit{STANDID} \times \textit{OPTIONID}) \rightarrow \textit{COST} \end{array}$
$\begin{array}{l} \forall sID : \textit{STANDID} \bullet \\ \quad \forall oID : \textit{OPTIONID} \bullet \\ \quad (sID, oID) \in \text{dom } \textit{standOptionCosts} \wedge \\ \quad \textit{standOptionCosts} (sID, oID) \geq 0 \end{array}$

Schema *InitStandOptionCosts* includes the unchangeable schema *DefineStandOptionCosts*. In this schema, the function *standOptionCosts* is set to the empty set. In schema *SetStandOptionCostsToZero*, all the stand option costs are initialised to zero.

$\begin{array}{l} \textit{InitStandOptionCosts} \\ \Delta \textit{DefineStandOptionCosts} \end{array}$
$\textit{standOptionCosts}' = \emptyset$

$\begin{array}{l} \textit{SetStandOptionCostsToZero} \\ \Delta \textit{InitStandOptionCosts} \end{array}$
$\begin{array}{l} \forall sID : \textit{STANDID} \bullet \\ \quad \forall oID : \textit{OPTIONID} \bullet \\ \quad (sID, oID) \in \text{dom } \textit{standOptionCosts} \wedge \\ \quad \textit{standOptionCosts}' = \textit{standOptionCosts} \cup \{(sID, oID), 0\} \end{array}$

Schema *StandOptionCosts* is composed by combining the schemas *DefineStandOptionCosts*, *InitStandOptionCosts* and *SetStandOptionCostsToZero*.

$$\textit{StandOptionCosts} \hat{=} \textit{DefineStandOptionCosts} \wedge \textit{InitStandOptionCosts} \wedge \textit{SetStandOptionCostsToZero}$$

Schema *CalculateStandOptionCosts* calculates the regime and transport cost for a single stand option. The schema includes the unchangeable schemas

RegimeStandOptionCosts and *TransportStandOptionCosts*, and the changeable schema *StandOptionCosts*. For every stand option, the function stand option costs are calculated by adding the regime stand option costs to the transport stand option costs.

$$\begin{array}{l}
 \text{CalculateStandOptionCosts} \\
 \exists \text{RegimeStandOptionCosts} \\
 \exists \text{TransportStandOptionCosts} \\
 \Delta \text{StandOptionCosts} \\
 \hline
 \forall sID : \text{STANDID} \bullet \\
 \quad \forall oID : \text{OPTIONID} \bullet \\
 \quad (sID, oID) \in \text{dom regimeStandOptionCosts} \wedge \\
 \quad (sID, oID) \in \text{dom transportStandOptionCosts} \wedge \\
 \quad (sID, oID) \in \text{dom standOptionCosts} \wedge \\
 \quad \text{standOptionCosts}' = \text{standOptionCosts} \oplus \{(sID, oID) \mapsto \\
 \quad \text{regimeStandOptionCosts}(sID, oID) + \text{transportStandOptionCosts}(sID, oID)\}
 \end{array}$$

E.8 Objective function

In the previous sections, all the possible stand options have been determined for the strategic planning horizon, the possible solutions to the forest harvest scheduling system have been listed, and the feasibility of each has been determined. The next step is to calculate the objective function (profit made over the strategic planning horizon) for each feasible solution.

The objective function is made up of several parts which need to be calculated before calculating the objective function. After these preparatory calculations have been made, the objective function can be calculated.

E.8.1 Preparatory calculations

Calculating the objective function necessitates combining several aspects. The first aspect to calculate is the income which the forestry part of the integrated pulp and paper company will receive as a result of delivering the logs to each mill over the strategic planning horizon. The next aspect to calculate is the cost of each possible solution. This uses the calculated cost of each stand option (which was determined previously in section E.7). Finally, the overhead costs which were incurred at a company and estate level are calculated. These costs are independent of the regime chosen.

E.8.1.1 Calculating the mills' delivered log cost

The objective function, which is defined and calculated in section E.8.2, calculates the profit, which is the income less the costs for the forest harvest scheduling system. In this section, the income is calculated for each feasible solution. The income is the money received by the forestry part of the company for selling logs to the mills.

The income is calculated in two steps: first, the annual cost of delivering logs to each mill is determined and stored in function *millLogPurchases*. Next, this annual delivered log cost is combined into a single cost for all mills, for all years, and is stored in function *possSolnDeliveredMillCosts*.

The schema *DefineMillLogPurchases* contains a function *millLogPurchases* which stores the cost of purchasing logs for each possible solution for each mill on a year by year basis. It also includes the unchangeable schema *PlanningHorizonDefinition*. For each year in the strategic planning horizon, the value of logs purchased is greater than or equal to zero.

Schema *InitMillLogPurchases* initialises the range of the function *millLogPurchases* to empty set, while schema *SetMillLogPurchasesToZero* sets the value of the function *millLogPurchases* to zero for each mill and year in the strategic planning horizon. The schema *MillLogPurchases* is then defined by combining *DefineMillLogPurchases*, *InitMillLogPurchases* and *SetMillLogPurchasesToZero*.

DefineMillLogPurchases

\exists *PlanningHorizonDefinition*

millLogPurchases : $POSSIBLESOLNID \times MILLID \times YEARNO \rightarrow COST$

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$(pSID, mID, yearNo) \in \text{dom } millLogPurchases \wedge$

$millLogPurchases(pSID, mID, yearNo) \geq 0$

InitMillLogPurchases

Δ *DefineMillLogPurchases*

$millLogPurchases' = \emptyset$

SetMillLogPurchasesToZero

\exists *PlanningHorizonDefinition*

Δ *InitMillLogPurchases*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(pSID, mID, yearNo) \in \text{dom } millLogPurchases \wedge$
 $millLogPurchases' = millLogPurchases \cup \{(pSID, mID, yearNo), 0\}$

$MillLogPurchases \hat{=} DefineMillLogPurchases \wedge InitMillLogPurchases \wedge$
 $SetMillLogPurchasesToZero$

The schema *CalculateAnnualMillLogPurchases* calculates the cost of logs delivered to each mill when implementing a particular feasible solution. The schema includes the unchangeable schemas *PlanningHorizonDefinition*, *PossibleSolution*, *FeasibleSolution* and *AnnualLogMassForMill* and the changeable schema *MillLogPurchases*. For each possible solution which is also feasible, and for each year in the strategic planning horizon, the cost of the mill's log purchases is calculated by multiplying the mass of logs which would be delivered to the mill (should this possible solution become the optimal solution) by the delivered log cost rate (*logCostRate* which was defined in section E.4.4).

CalculateAnnualMillLogPurchases

\exists *PlanningHorizonDefinition*

\exists *PossibleSolution*

\exists *FeasibleSolution*

\exists *AnnualLogMassForMill*

Δ *MillLogPurchases*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNNO \bullet$
 $\forall sID : STANDID \bullet \forall oID : OPTIONID \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $pSID \in \text{dom } possibleSolution \wedge$
 $(sID, oID) \in \text{ran } possibleSolution \wedge$
 $pSID \in \text{dom } feasibleSolution \wedge$
 $feasibleSolution pSID = feasible \wedge$
 $(pSID, mID, yearNo) \in \text{dom } annualLogMassForMill \wedge$
 $(pSID, mID, yearNo) \in \text{dom } millLogPurchases \wedge$
 $millLogPurchases' = millLogPurchases \oplus \{(pSID, mID, yearNo) \mapsto$
 $(millLogPurchases (pSID, mID, yearNo)$
 $+ annualLogMassForMill (pSID, mID, yearNo) * logCostRate)\}$

The annual cost of delivering logs to each mill has now been determined and stored in function *millLogPurchases*. To facilitate ease of calculating the objective function in section E.8.2, this annual delivered log cost is now combined into a single cost for all mills, for all years, and is stored in function *possSolnDeliveredMillCosts*, which is

defined, initialised and populated next.

Schema *DefineDeliveredLogMillCosts* defines a function, *possSolnDeliveredMillCosts*, which stores the total delivered mill costs for a particular possible solution. (This is the amount which the mills will pay for the delivered logs over the duration of the strategic planning horizon.) Schema *DefineDeliveredLogMillCosts* states that for every possible solution, the function *possSolnDeliveredMillCosts* must be greater than or equal to zero.

Schema *InitDeliveredLogMillCosts* initialises the contents of the function *possSolnDeliveredMillCosts* to be zero. Schema *DeliveredLogMillCosts* combines both schemas *DefineDeliveredLogMillCosts* and *InitDeliveredLogMillCosts*.

Schema *DefineDeliveredLogMillCosts* defines a function, *possSolnDeliveredMillCosts*, which stores the total delivered mill costs for a particular possible solution. (This is the amount which the mills will pay for the delivered logs over the duration of the strategic planning horizon.) Schema *DefineDeliveredLogMillCosts* states that for every possible solution, the function *possSolnDeliveredMillCosts* must be greater than or equal to zero.

Schema *InitDeliveredLogMillCosts* initialises the contents of the function *possSolnDeliveredMillCosts* to the empty set, while schema *SetDeliveredLogMillCostsToZero* sets the value of this function to zero. Schema *DeliveredLogMillCosts* combines the schemas *DefineDeliveredLogMillCosts*, *InitDeliveredLogMillCosts* and *SetDeliveredLogMillCostsToZero*.

$$\begin{array}{l}
 \text{--- } \underline{\textit{DefineDeliveredLogMillCosts}} \text{ ---} \\
 \textit{possSolnDeliveredMillCosts} : \textit{POSSIBLESOLNID} \rightarrow \textit{COST} \\
 \forall pSID : \textit{POSSIBLESOLNID} \bullet \\
 \quad pSID \in \text{dom } \textit{possSolnDeliveredMillCosts} \wedge \\
 \quad \textit{possSolnDeliveredMillCosts } pSID \geq 0
 \end{array}$$

$$\begin{array}{l}
 \text{--- } \underline{\textit{InitDeliveredLogMillCosts}} \text{ ---} \\
 \Delta \textit{DefineDeliveredLogMillCosts} \\
 \textit{possSolnDeliveredMillCosts}' = \emptyset
 \end{array}$$

$$\begin{array}{l}
 \text{--- } \underline{\textit{SetDeliveredLogMillCostsToZero}} \text{ ---} \\
 \Delta \textit{InitDeliveredLogMillCosts} \\
 \forall pSID : \textit{POSSIBLESOLNID} \bullet \\
 \quad pSID \in \text{dom } \textit{possSolnDeliveredMillCosts} \wedge \\
 \quad \textit{possSolnDeliveredMillCosts}' = \textit{possSolnDeliveredMillCosts} \cup \{(pSID, 0)\}
 \end{array}$$

$$\text{DeliveredLogMillCosts} \hat{=} \text{DefineDeliveredLogMillCosts} \wedge \\ \text{InitDeliveredLogMillCosts} \wedge \text{SetDeliveredLogMillCostsToZero}$$

Schema *CalculateDeliveredLogMillCosts* updates the function *possSolnDeliveredMillCosts* with the total delivered log costs to all mills over the strategic planning horizon, for each feasible solution which could be implemented. It includes the unchangeable schemas *PlanningHorizonDefinition*, *PossibleSolution*, *FeasibleSolution* and *MillLogPurchases* and the changeable schema *DeliveredLogMillCosts*. For all possible solutions which are also feasible solutions, the contents of the function *possSolnDeliveredMillCosts* are updated with the addition of the log purchases for each mill and each year in the strategic planning horizon.

CalculateDeliveredLogMillCosts

\exists *PlanningHorizonDefinition*

\exists *PossibleSolution*

\exists *FeasibleSolution*

\exists *MillLogPurchases*

Δ *DeliveredLogMillCosts*

$\forall pSID : POSSIBLESOLNID \bullet \forall mID : MILLID \bullet \forall yearNo : YEARNO \bullet$

$1 \leq yearNo \leq strategicHorizon? \wedge$

$pSID \in \text{dom } possibleSolution \wedge$

$pSID \in \text{dom } feasibleSolution \wedge$

$feasibleSolution \ pSID = feasible \wedge$

$pSID \in \text{dom } possSolnDeliveredMillCosts \wedge$

$(pSID, mID, yearNo) \in \text{dom } millLogPurchases \wedge$

$possSolnDeliveredMillCosts' = possSolnDeliveredMillCosts \oplus$

$\{pSID \mapsto possSolnDeliveredMillCosts \ pSID$

$+ millLogPurchases \ (pSID, mID, yearNo)\}$

E.8.1.2 Calculating the cost for each possible solution

The cost incurred by each possible solution is the cost of growing the trees, and then delivering them to the mill. This cost has been calculated for each stand option (in section E.7), but the cost for each possible solution (which is a combination of the stand options) now needs to be calculated.

The schema *DefinePossibleSolutionStandOptionCosts* defines a function, *possSolnStandOptionCosts*, which contains the sum of the forestry and transport costs for a particular possible solutionID. This schema includes the unchangeable schema *PlanningHorizonDefinition*. For each year in the strategic planning horizon, the range of the function *possSolnStandOptionCosts* is greater than or equal to zero.

Schema *InitPossibleSolutionStandOptionCosts* initialises the range of the function *possSolnStandOptionCosts* to the empty set, and schema *SetPSolutionStandOptionCostsToZero* sets the value of the function to zero. Schema *PossibleSolutionStandOptionCosts* combines the schemas *DefinePossibleSolutionStandOptionCosts*, *InitPossibleSolutionStandOptionCosts* and *SetPSolutionStandOptionCostsToZero*.

$\begin{array}{l} \text{---} \textit{DefinePossibleSolutionStandOptionCosts} \text{---} \\ \exists \textit{PlanningHorizonDefinition} \\ \textit{possSolnStandOptionCosts} : \textit{POSSIBLESOLNID} \rightarrow \textit{COST} \\ \forall pSID : \textit{POSSIBLESOLNID} \bullet \forall yearNo : \textit{YEARNO} \bullet \\ \quad 1 \leq yearNo \leq \textit{strategicHorizon}? \wedge \\ \quad pSID \in \text{dom } \textit{possSolnStandOptionCosts} \wedge \\ \quad \textit{possSolnStandOptionCosts } pSID \geq 0 \end{array}$

$\begin{array}{l} \text{---} \textit{InitPossibleSolutionStandOptionCosts} \text{---} \\ \Delta \textit{DefinePossibleSolutionStandOptionCosts} \\ \textit{possSolnStandOptionCosts}' = \emptyset \end{array}$
--

$\begin{array}{l} \text{---} \textit{SetPSolutionStandOptionCostsToZero} \text{---} \\ \exists \textit{PlanningHorizonDefinition} \\ \Delta \textit{InitPossibleSolutionStandOptionCosts} \\ \forall pSID : \textit{POSSIBLESOLNID} \bullet \forall yearNo : \textit{YEARNO} \bullet \\ \quad 1 \leq yearNo \leq \textit{strategicHorizon}? \wedge \\ \quad pSID \in \text{dom } \textit{possSolnStandOptionCosts} \wedge \\ \quad \textit{possSolnStandOptionCosts}' = \textit{possSolnStandOptionCosts} \cup \{(pSID, 0)\} \end{array}$

$$\textit{PossibleSolutionStandOptionCosts} \hat{=} \textit{DefinePossibleSolutionStandOptionCosts} \wedge \textit{InitPossibleSolutionStandOptionCosts} \wedge \textit{SetPSolutionStandOptionCostsToZero}$$

Schema *CalculatePossibleSolutionStandOptionCosts* updates the function *possSolnStandOptionCosts*, which stores the total costs of growing the trees for the various mills, and delivering them to the mills, for a possible solution. This schema includes three unchangeable schemas (*PossibleSolution*, *FeasibleSolution* and *StandOptionCosts*) and one changeable schema (*PossibleSolutionStandOptionCosts*). For each stand in the possible solution, the cost of each stand option is added to the function *possSolnStandOptionCosts*.

CalculatePossibleSolutionStandOptionCosts

\exists *PossibleSolution*

\exists *FeasibleSolution*

\exists *StandOptionCosts*

Δ *PossibleSolutionStandOptionCosts*

$\forall pSID : POSSIBLESOLNID \bullet \forall sID : STANDID \bullet \forall oID : OPTIONID \bullet$

$pSID \in \text{dom } possibleSolution \wedge$

$(sID, oID) \in \text{ran } possibleSolution \wedge$

$pSID \in \text{dom } feasibleSolution \wedge$

$possibleSolution \ pSID = (sID, oID) \wedge$

$feasibleSolution \ pSID = feasible \wedge$

$(sID, oID) \in \text{dom } standOptionCosts \wedge$

$pSID \in \text{dom } possSolnStandOptionCosts \wedge$

$possSolnStandOptionCosts' = possSolnStandOptionCosts \oplus$

$\{pSID \mapsto possSolnStandOptionCosts \ pSID + standOptionCosts \ (sID, oID)\}$

E.8.1.3 Calculating the total non-stand cost

The forestry company income has been calculated, and the cost of producing and delivering the logs has been determined for each possible solution. The cost of producing and delivering logs does not include the forestry company overhead costs which are incurred independently of the regime implemented or the transport costs incurred. These overhead costs are described and calculated below. The estate-level costs were defined annually per estate in section E.4.2.2. These are summed so that an annual estate-level cost is obtained. The costs for both the estate-level and company-level costs over the planning horizon is determined next. The company-level costs were defined in section E.4.2.1.

Schema *DefineAnnualEstateLevelCosts* is used to store the function *annualEstateLevelCosts*, which contains the sum of the estate-level costs, on a year-by-year basis. This schema includes the unchangeable schema *PlanningHorizonDefinition* and the function *annualEstateLevelCosts*, which takes as input the year number in the strategic plan, and has as output the annual costs incurred at an estate level. For each year, the function's value is greater than or equal to zero.

Schema *InitAnnualEstateLevelCosts* initialises the contents of the function *annualEstateLevelCosts* to the empty set, and schema *SetAnnualEstateLevelCostsToZero* sets all its values to zero. Schema *AnnualEstateLevelCosts* combines the schemas *DefineAnnualEstateLevelCosts*, *InitAnnualEstateLevelCosts* and *SetAnnualEstateLevelCostsToZero*.

DefineAnnualEstateLevelCosts _____
 Ξ *PlanningHorizonDefinition*
 $annualEstateLevelCosts : YEARNO \rightarrow COST$

$\forall yearNo : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $yearNo \in \text{dom } annualEstateLevelCosts \wedge$
 $(annualEstateLevelCosts \ yearNo) \geq 0$

InitAnnualEstateLevelCosts _____
 Δ *DefineAnnualEstateLevelCosts*

$annualEstateLevelCosts' = \emptyset$

SetAnnualEstateLevelCostsToZero _____

Ξ *PlanningHorizonDefinition*
 Δ *InitAnnualEstateLevelCosts*

$\forall yearNo : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $yearNo \in \text{dom } annualEstateLevelCosts \wedge$
 $annualEstateLevelCosts' = annualEstateLevelCosts \cup \{(yearNo, 0)\}$

$AnnualEstateLevelCosts \hat{=} DefineAnnualEstateLevelCosts \wedge$
 $InitAnnualEstateLevelCosts \wedge SetAnnualEstateLevelCostsToZero$

Schema *CalculateAnnualEstateLevelCosts* calculates the estate-level costs on an annual basis. This schema includes the two unchangeable schemas *PlanningHorizonDefinition* and *EstateLevelCosts*, and the changeable schema *AnnualEstateLevelCosts*. For each year in the strategic planning horizon, the sum of each estate's costs for that year is added to the function *annualEstateLevelCosts*.

CalculateAnnualEstateLevelCosts _____

Ξ *PlanningHorizonDefinition*
 Ξ *EstateLevelCosts*
 Δ *AnnualEstateLevelCosts*

$\forall eID : ESTATEID \bullet$
 $\forall yearNo : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge$
 $(eID, yearNo) \in \text{dom } estateLevelCosts \wedge$
 $yearNo \in \text{dom } annualEstateLevelCosts \wedge$
 $annualEstateLevelCosts' = annualEstateLevelCosts \oplus \{yearNo \mapsto$
 $annualEstateLevelCosts \ yearNo + estateLevelCosts (eID, yearNo)\}$

The annual estate-level costs have now been determined. The estate-level and company-level costs over the planning horizon can now be determined.

Schema *DefineTotalNonStandForestryCosts* defines two functions, *totalEstateCosts* and *totalCompanyCosts*, which will contain the total estate-level costs and company-level costs for the strategic planning horizon. This schema states that the total estate-level costs and the total company-level costs must be greater than or equal to zero.

Schema *InitTotalNonStandForestryCosts* initialises the contents of the two functions *totalEstateCosts* and *totalCompanyCosts* to the empty set, while schema *SetTotalNonStandForestryCostsToZero* sets the contents of the function *totalCompanyCosts* to zero. Schema *TotalNonStandForestryCosts* combines both these schemas *DefineTotalNonStandForestryCosts*, *InitTotalNonStandForestryCosts* and *SetTotalNonStandForestryCostsToZero*.

DefineTotalNonStandForestryCosts

$totalEstateCosts : COMPANYID \rightarrow COST$
 $totalCompanyCosts : COMPANYID \rightarrow COST$

$\exists_1 cID : COMPANYID \bullet$
 $cID \in \text{dom } totalEstateCosts \wedge$
 $cID \in \text{dom } totalCompanyCosts \wedge$
 $totalEstateCosts \ cID \geq 0 \wedge$
 $totalCompanyCosts \ cID \geq 0$

InitTotalNonStandForestryCosts

$\Delta DefineTotalNonStandForestryCosts$

$totalEstateCosts' = \emptyset$
 $totalCompanyCosts' = \emptyset$

SetTotalNonStandForestryCostsToZero

$\Delta InitTotalNonStandForestryCosts$

$\exists_1 cID : COMPANYID \bullet$
 $cID \in \text{dom } totalEstateCosts \wedge$
 $cID \in \text{dom } totalCompanyCosts \wedge$
 $totalEstateCosts' = totalEstateCosts \cup \{(cID, 0)\} \wedge$
 $totalCompanyCosts' = totalCompanyCosts \cup \{(cID, 0)\}$

$TotalNonStandForestryCosts \hat{=} DefineTotalNonStandForestryCosts \wedge$
 $InitTotalNonStandForestryCosts \wedge SetTotalNonStandForestryCostsToZero$

Schema *CalculateDeliveredLogMillCosts* updates the functions *totalEstateCosts* and *totalCompanyCosts* so that these functions contain a single cost, respectively. This schema contains the unchangeable schemas *PlanningHorizonDefinition*, *AnnualEstateLevelCosts* and *ForestryCompanyLevelCosts*, and the changeable schema *TotalNonStandForestryCosts*. For each year in the strategic planning horizon, the annual estate-level costs and the annual company-level costs are summed and the results placed in functions *totalEstateCosts* and *totalCompanyCosts* respectively.

<p style="text-align: center;"><i>CalculateTotalNonStandForestryCosts</i></p> <hr/> <p>\exists <i>PlanningHorizonDefinition</i> \exists <i>AnnualEstateLevelCosts</i> \exists <i>ForestryCompanyLevelCosts</i> Δ <i>TotalNonStandForestryCosts</i></p> <hr/> <p>$\exists_1 cID : COMPANYID \bullet \forall eID : ESTATEID \bullet \forall yearNo : YEARN0 \bullet$ $1 \leq yearNo \leq strategicHorizon? \wedge$ $yearNo \in \text{dom } annualEstateLevelCosts \wedge$ $(cID, yearNo) \in \text{dom } companyLevelCosts \wedge$ $cID \in \text{dom } totalEstateCosts \wedge$ $cID \in \text{dom } totalCompanyCosts \wedge$ $totalEstateCosts' = totalEstateCosts \oplus \{cID \mapsto$ $totalEstateCosts \ cID + annualEstateLevelCosts \ yearNo\} \wedge$ $totalCompanyCosts' = totalCompanyCosts \oplus \{cID \mapsto$ $totalCompanyCosts \ cID + companyLevelCosts \ (cID, yearNo)\}$</p>
--

E.8.2 Calculating the objective function for each feasible solution

The components which make up the objective function have now been determined. The objective function can now be defined, initialised and populated. The objective function is the income from selling logs to the mill less the cost of producing them and transporting them to the mill less the forestry overhead costs.

Schema *DefineObjectiveFunction* defines a function *objectiveFunction* which takes as input a possible solution ID, and gives a cost as output. Schema *InitObjectiveFunction* initialises all the values of the objective functions to the empty set, and schema *SetObjectiveFunctionToZero* sets the values to zero. Schema *ObjectiveFunction* combines the schemas *DefineObjectiveFunction*, *InitObjectiveFunction* and *SetObjectiveFunctionToZero*.

<p style="text-align: center;"><i>DefineObjectiveFunction</i></p> <hr/> <p><i>objectiveFunction</i> : POSSIBLESOLNID \rightarrow COST</p>
--

<i>InitObjectiveFunction</i>
Δ <i>DefineObjectiveFunction</i>
$objectiveFunction' = \emptyset$

<i>SetObjectiveFunctionToZero</i>
Δ <i>InitObjectiveFunction</i>
$\forall pSID : POSSIBLESOLNID \bullet$ $pSID \in \text{dom } objectiveFunction \wedge$ $objectiveFunction' = objectiveFunction \cup \{(pSID, 0)\}$

$$ObjectiveFunction \hat{=} DefineObjectiveFunction \wedge$$

$$InitObjectiveFunction \wedge SetObjectiveFunctionToZero$$

Schema *CalculateObjectiveFunction* updates the function *objectiveFunction* for each possible solution. This schema contains the five unchangeable schemas *PossibleSolution*, *FeasibleSolution*, *DeliveredLogMillCosts*, *PossibleSolutionStandOptionCosts* and *TotalNonStandForestryCosts* and the changeable schema *ObjectiveFunction*. For each possible solution which is also a feasible solution, the objective function is calculated: the amount the mills pay for delivered logs less (the cost of producing the logs and delivering them to the mill, the estate-level costs and the company-level costs).

<i>CalculateObjectiveFunction</i>
\exists <i>PossibleSolution</i>
\exists <i>FeasibleSolution</i>
\exists <i>DeliveredLogMillCosts</i>
\exists <i>PossibleSolutionStandOptionCosts</i>
\exists <i>TotalNonStandForestryCosts</i>
Δ <i>ObjectiveFunction</i>
$\forall pSID : POSSIBLESOLNID \bullet \exists_1 cID : COMPANYID \bullet$ $pSID \in \text{dom } possibleSolution \wedge$ $pSID \in \text{dom } feasibleSolution \wedge$ $feasibleSolution pSID = feasible \wedge$ $pSID \in \text{dom } possSolnDeliveredMillCosts \wedge$ $pSID \in \text{dom } possSolnStandOptionCosts \wedge$ $cID \in \text{dom } totalEstateCosts \wedge$ $cID \in \text{dom } totalCompanyCosts \wedge$ $pSID \in \text{dom } objectiveFunction \wedge$ $objectiveFunction' = objectiveFunction \oplus \{pSID \mapsto$ $\quad possSolnDeliveredMillCosts pSID$ $\quad - possSolnStandOptionCosts pSID$ $\quad - totalEstateCosts cID$ $\quad - totalCompanyCosts cID\}$

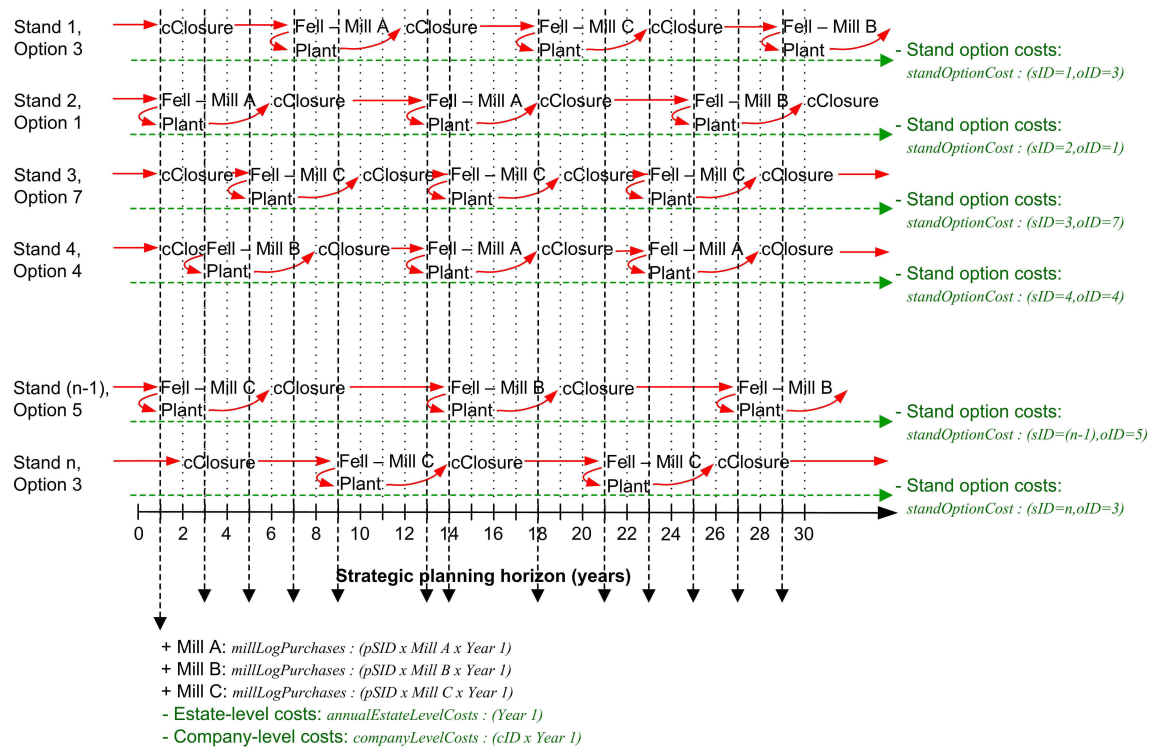


Figure E.5: Calculation of the objective function, for a particular possible solution. The objective function calculates the profit of a possible solution. The income is from the forests selling the logs to the mill. The costs are those of growing the trees and delivering them to the mill, as well as estate-level and company-level costs

Figure E.5 outlines how the objective function is calculated for a possible solution which is feasible. For each stand option, the costs of planting, maintaining the stand (before and after canopy closure), harvesting the stand's timber, and the costs of transporting the harvested timber to the appropriate mill are calculated and stored in *standOptionCosts*. These are shown to the right of the figure. These stand option costs are costs in the profit calculation of the objective function, so are shown as $- standOptionCosts$. These stand option costs are summed and stored in *possSolnStandOptionCosts*.

For each year in which there is a harvesting event, the delivered cost of logs to each mill is calculated and stored in *millLogPurchases*. This is the income in the profit calculation of the objective function, thus is shown as $+ millLogPurchases$ in the figure.

Not included in the stand option costs are the estate-level and company-level costs. These costs are defined for each year of the strategic planning horizon, and they are subtracted from the objective function.

E.9 Optimal solution

The optimal solution is defined to be the possible solution which is feasible, for which the objective function is at its maximum. In this section, the optimal solution is defined, initialised and calculated. The optimal solution is then fed back into the domain description in that the destination mill for each stand's first felling event is updated in schema *MillForStandsLogs* (defined in section D.3.2).

E.9.1 Calculating the optimal solution

In this section, the optimal solution is defined, initialised and calculated.

Schema *DefineOptimalSolution* defines a function *optimalSolution*, which takes the companyID as input and the possible solutionID as the output. Schema *InitOptimalSolution* initialises this function's range to be undefined, and schema *OptimalSolution* combines the two schemas *DefineOptimalSolution* and *InitOptimalSolution*.

<i>DefineOptimalSolution</i> $optimalSolution : COMPANYID \leftrightarrow POSSIBLESOLNID$
--

<i>InitOptimalSolution</i> $\Delta DefineOptimalSolution$
$\exists_1 cID : COMPANYID \bullet$ $cID \in \text{dom } optimalSolution \wedge$ $optimalSolution' = \emptyset$

$$OptimalSolution \hat{=} DefineOptimalSolution \wedge InitOptimalSolution$$

Schema *DetermineOptimalSolution* determines which of the feasible solutions is optimal. This schema contains three unchangeable schemas (*PossibleSolution*, *FeasibleSolution* and *ObjectiveFunction*) and the changeable schema *OptimalSolution*. For all the possible solutions, there exists a single possible solution (*pSID1*) for which the objective function has the optimum value. This is the optimal solution.

DetermineOptimalSolution

\exists *PossibleSolution*

\exists *FeasibleSolution*

\exists *ObjectiveFunction*

Δ *OptimalSolution*

$\forall pSID : POSSIBLESOLNID \bullet \exists_1 cID : COMPANYID \bullet$

$\exists_1 pSID1 : POSSIBLESOLNID \bullet$

$pSID \in \text{dom possibleSolution} \wedge$

$pSID \in \text{dom feasibleSolution} \wedge$

$\text{feasibleSolution } pSID = \text{feasible} \wedge$

$pSID \in \text{dom objectiveFunction} \wedge$

$cID \in \text{dom optimalSolution} \wedge$

$pSID \in \text{ran optimalSolution} \wedge$

$pSID1 \in \text{dom possibleSolution} \wedge$

$pSID1 \in \text{dom feasibleSolution} \wedge$

$\text{feasibleSolution } pSID1 = \text{feasible} \wedge$

$pSID1 \in \text{dom objectiveFunction} \wedge$

$pSID1 \in \text{ran optimalSolution} \wedge$

$\text{objectiveFunction } pSID1 = \max\{\text{objectiveFunction } pSID\} \Rightarrow$

$\text{optimalSolution}' = \text{optimalSolution} \oplus \{cID \mapsto pSID1\}$

E.9.2 Updating the schema *MillForStandsLogs* with the optimal solution

The optimal solution for the forest harvest scheduling system has been found in the previous section. The optimal solution is the possible solution whose profit had the maximum value. This possible solution is made up of a number of stand options (one stand option per stand). The mill to which the first felling event's timber can be sent can now be updated. This shows how the harvesting decision made by the forest harvest scheduling system affects what occurs in the forestry domain.

Schema *UpdateMillForStandsLogs* updates the function *millForStandsLogs* with the mill to which the first felling event of logs should be sent from the stand. This function was defined in the domain description (see section D.3.2).

Schema *UpdateMillForStandsLogs* includes the unchangeable schemas *PlanningHorizonDefinition*, *StandRegimeAction*, *StandHarvestingInfo*, *PossibleSolution* and *OptimalSolution*, and the changeable schema *MillForStandsLogs*. For every stand which is in the optimal solution, the first harvesting event is found, and the mill allocation decision for that stand is updated in the function *millForStandsLogs*.

UpdateMillForStandsLogs

\exists *PlanningHorizonDefinition*

\exists *StandRegimeAction*

\exists *StandHarvestingInfo*

\exists *PossibleSolution*

\exists *OptimalSolution*

Δ *MillForStandsLogs*

$\forall sID : STANDID \bullet \exists oID : OPTIONID \bullet \exists yearNo : YEARNO \bullet$
 $\exists_1 pSID1 : POSSIBLESOLNID \bullet \exists_1 mID : MILLID \bullet \exists_1 yearNo1 : YEARNO \bullet$
 $1 \leq yearNo \leq strategicHorizon? \wedge 1 \leq yearNo1 \leq strategicHorizon? \wedge$
 $(sID, oID, yearNo) \in \text{dom } standRegimeAction \wedge$
 $(sID, oID, yearNo) \in \text{dom } standHarvestingInfo \wedge$
 $(sID, oID, yearNo1) \in \text{dom } standRegimeAction \wedge$
 $(sID, oID, yearNo1) \in \text{dom } standHarvestingInfo \wedge$
 $pSID1 \in \text{dom } possibleSolution \wedge$
 $(sID, oID) \in \text{ran } possibleSolution \wedge$
 $possibleSolution\ pSID1 = (sID, oID) \wedge$
 $sID \in \text{dom } millForStandsLogs \wedge$
 $mID \in \text{ran } millForStandsLogs \wedge$
 $standRegimeAction\ (sID, oID, yearNo1) = fell \wedge$
 $yearNo1 = \min\{ThirdOf3\ (sID, oID, yearNo)\} \wedge$
 $mID = FirstOf5\ (standHarvestingInfo\ (sID, oID, yearNo1)) \wedge$
 $millForStandsLogs' = millForStandsLogs \oplus \{sID \mapsto mID\}$

E.10 Forest harvest scheduling system

The different components of the forest harvest scheduling system have been defined in sections E.1 to E.9.1. In this section, they are combined to form the forest harvest scheduling system which takes wood properties into account when making the harvesting decision.

Schema *FHSSPart1* includes five schema which will be called at the beginning of the forest harvest scheduling system run. The system first accepts the user's input of the strategic, tactical and operational planning horizon length, and the date on which the plan is to start. It then generates the regime actions for each stand and works out the harvesting outcome for those regime actions. The first regime in each stand option is determined so that the costs of growing the trees and transporting them to the mill can be calculated. The possible solutions are enumerated, and each possible solution is tested to see if it is also a feasible solution. The list of possible solutions is inspected to see if one or more is feasible; if so, the second part of the forest harvest scheduling system can be called.

FHSSPart1

PlanningHorizonDefinition
GenerateStandRegimeActionForStands
UpdateHarvestingOutcomes
CalculateStandOptionCosts
CalculatePossibleSolutions
CalculateAnnualLogMassForMill
DetermineFeasibleSolutions
FeasibleSolutionsExist

Schema *FHSSPart2* comprises of schemas which will be called if there is a feasible solution. They determine the cost of growing the trees and transporting them to the mills. They also determine how much the mills pay for their delivered logs, and what the total estate- and forestry company-level costs are over the strategic planning horizon. The objective function is determined for all feasible solutions, and the optimal solution determined.

FHSSPart2

DetermineFirstRegimeActionInStandOption
CalculateRegimeStandOptionCosts
CalculateSHaulTransportStandOptionCosts
CalculateLHaulTransportStandOptionCosts
CalculateTransportStandOptionCosts
CalculateAnnualMillLogPurchases
CalculatePossibleSolutionStandOptionCosts
CalculateDeliveredLogMillCosts
CalculateTotalNonStandForestryCosts
CalculateObjectiveFunction
DetermineOptimalSolution

Schema *FHSS* (which represents the forest harvest scheduling system) combines the two schemas *FHSSPart1* and *FHSSPart2*. It also includes the unchangeable schema *FeasibleSolutionsExist*, and two outputs, *optimalSolution!* (of type *COST*) and *message!* (of type *FEASIBLESOLN*). This schema states that *FHSSPart1* will be run. If there are feasible solutions, the optimal solution will be calculated and the optimal objective function output in *optimalSolution!*. (Other reports, such as the description of the optimal solution, the list of logs to be transported per annum, the list of logs which will arrive at each mill's logyard per annum, etc. can also be output, but these have not been described here.) If there are no feasible solutions, a message will be output to this effect.

FHSS

FHSSPart1

FHSSPart2

\exists *FeasibleSolutionsExist*

optimalSolution! : *COST*

message! : *FEASIBLESOLN*

\exists_1 *cID* : *COMPANYID* • \exists_1 *pSID1* : *POSSIBLESOLNID* •

FHSSPart1 \wedge

cID \in *dom feasibleSolutionsExist* \wedge

feasibleSolutionsExist cID = feasibleSolnsExist \Rightarrow

(*FHSSPart2* \wedge

cID \in *dom optimalSolution* \wedge

pSID1 \in *ran optimalSolution* \wedge

pSID1 \in *dom objectiveFunction* \wedge

optimalSolution! = *objectiveFunction (optimalSolution (cID))*) \wedge

feasibleSolutionsExist cID = allSolnsInfeasible \Rightarrow

message! = *allSolnsInfeasible*

E.11 Log file generated by ZTC for the forest harvest scheduling system specification

This section gives the Z Type Checker output file (.log) obtained when the formal specification of the forest harvest scheduling system given above was typechecked.

Log opened at: Mon Jun 01 06:45:03 2009

```
... Initializing.
... Loading Z mathematical tools library: math0.zed
Parsing main file: ..\latex\MSc_specs_31-05-09.tex
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 221
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 224
... Type checking Equivalence definition: GENMATNAME. "..\latex\MSc_specs_31-05-09.tex" Line 226
... Type checking Schema box: GeneticMaterialNames. "..\latex\MSc_specs_31-05-09.tex" Lines 254-273
... Type checking Schema box: Genus. "..\latex\MSc_specs_31-05-09.tex" Lines 295-314
... Type checking Schema box: PureSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 337-371
... Type checking Schema box: Hybrid. "..\latex\MSc_specs_31-05-09.tex" Lines 388-425
... Type checking Schema definition: WoodGeneticMaterial. "..\latex\MSc_specs_31-05-09.tex" Line 447
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 475
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 479-481
... Type checking Free type definition: REGIMETYPE. "..\latex\MSc_specs_31-05-09.tex" Line 486
... Type checking Schema box: DateDefinition. "..\latex\MSc_specs_31-05-09.tex" Lines 503-521
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 533
... Type checking Schema box: RegimeGeneticMaterial. "..\latex\MSc_specs_31-05-09.tex" Lines 555-572
... Type checking Schema box: RegimePlantAge. "..\latex\MSc_specs_31-05-09.tex" Lines 593-600
... Type checking Schema box: RegimePlantDate. "..\latex\MSc_specs_31-05-09.tex" Lines 612-618
... Type checking Schema box: RegimePlanting. "..\latex\MSc_specs_31-05-09.tex" Lines 630-652
... Type checking Schema box: RegimeFellAge. "..\latex\MSc_specs_31-05-09.tex" Lines 674-683
... Type checking Schema box: RegimeFellDate. "..\latex\MSc_specs_31-05-09.tex" Lines 726-734
... Type checking Schema definition: RegimeFelling. "..\latex\MSc_specs_31-05-09.tex" Line 751
... Type checking Schema box: DefineTrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 774-787
... Type checking Schema box: InitTrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 791-797
```

```

... Type checking Schema definition: TrackActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Line 802
... Type checking Schema box: DefineRegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 824-825
... Type checking Schema box: InitRegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 829-835
... Type checking Schema definition: RegimeIDs. "..\latex\MSc_specs_31-05-09.tex" Line 840
... Type checking Schema box: RegimeFunctions. "..\latex\MSc_specs_31-05-09.tex" Lines 851-854
... Type checking Schema box: PlannedRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 867-900
... Type checking Schema box: ActualRegimes. "..\latex\MSc_specs_31-05-09.tex" Lines 912-949
... Type checking Schema definition: StoreRegimes. "..\latex\MSc_specs_31-05-09.tex" Line 971
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 1004
... Type checking Free type definition: MESSAGE. "..\latex\MSc_specs_31-05-09.tex" Line 1017
... Type checking Schema box: LogisticsChain. "..\latex\MSc_specs_31-05-09.tex" Lines 1036-1052
... Type checking Schema box: MillForStandsLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 1064-1084
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1106
... Type checking Schema box: DefineLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1132-1146
... Type checking Schema box: InitLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1150-1153
... Type checking Schema box: NoLogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Lines 1157-1162
... Type checking Schema definition: LogsAtRoadside. "..\latex\MSc_specs_31-05-09.tex" Line 1167
... Type checking Schema box: DefineDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1189-1214
... Type checking Schema box: InitDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1221-1224
... Type checking Schema box: NoLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 1228-1233
... Type checking Schema definition: Depot. "..\latex\MSc_specs_31-05-09.tex" Line 1238
... Type checking Schema box: MillAcceptableSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 1260-1288
... Type checking Schema box: DefineMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1303-1330
... Type checking Schema box: InitMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1334-1337
... Type checking Schema box: NoLogsAtMill. "..\latex\MSc_specs_31-05-09.tex" Lines 1341-1346
... Type checking Schema definition: Mill. "..\latex\MSc_specs_31-05-09.tex" Line 1351
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 1400
... Type checking Schema box: CompanyHasEstates. "..\latex\MSc_specs_31-05-09.tex" Lines 1411-1418
... Type checking Schema box: EstateBelongsToCompany. "..\latex\MSc_specs_31-05-09.tex" Lines 1429-1435
... Type checking Schema box: EstateHasStands. "..\latex\MSc_specs_31-05-09.tex" Lines 1446-1453
... Type checking Schema box: StandBelongsToEstate. "..\latex\MSc_specs_31-05-09.tex" Lines 1464-1470
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1490
... Type checking Schema box: StandSuitableSpecies. "..\latex\MSc_specs_31-05-09.tex" Lines 1502-1545
... Type checking Schema box: StandCharacteristics. "..\latex\MSc_specs_31-05-09.tex" Lines 1557-1565
... Type checking Schema definition: StandLand. "..\latex\MSc_specs_31-05-09.tex" Line 1578
... Type checking Free type definition: PLANTINGSTATE. "..\latex\MSc_specs_31-05-09.tex" Line 1598
... Type checking Schema box: DefineStandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1613-1618
... Type checking Schema box: InitStandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1622-1626
... Type checking Schema box: StandPlantingStatusUnplanted. "..\latex\MSc_specs_31-05-09.tex" Lines 1630-1636
... Type checking Schema definition: StandPlantingStatus. "..\latex\MSc_specs_31-05-09.tex" Lines 1641-1642
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 1679
... Type checking Schema box: SiteIndexEvalAge. "..\latex\MSc_specs_31-05-09.tex" Lines 1691-1698
... Type checking Schema box: StandSiteIndex. "..\latex\MSc_specs_31-05-09.tex" Lines 1709-1724
... Type checking Schema box: DefineStandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1738-1749
... Type checking Schema box: InitStandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1753-1757
... Type checking Schema box: SetStandVolumeAndMassToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 1761-1769
... Type checking Schema definition: StandVolumeAndMass. "..\latex\MSc_specs_31-05-09.tex" Lines 1774-1775
... Type checking Schema box: StandOfTrees. "..\latex\MSc_specs_31-05-09.tex" Lines 1792-1841
... Type checking Schema box: DeterminePlantingRegime. "..\latex\MSc_specs_31-05-09.tex" Lines 1870-1915
... Type checking Schema box: PlantStandAndRecordRegimeOK. "..\latex\MSc_specs_31-05-09.tex" Lines 1929-1966
... Type checking Schema box: ExceptionPlantStandAndRecordRegime. "..\latex\MSc_specs_31-05-09.tex"
    Lines 1983-1993
... Type checking Schema definition: PlantStandAndRecordRegime. "..\latex\MSc_specs_31-05-09.tex"
    Lines 1998-1999
... Type checking Schema box: PlantStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2011-2015
... Type checking Schema box: DefineAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2041-2047
... Type checking Schema box: InitAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2051-2054
... Type checking Schema definition: AgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2059-2060
... Type checking Schema box: CalculateAgeOfTreesInStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2074-2086
... Type checking Schema box: DefineEstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2104-2120
... Type checking Schema box: InitEstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2124-2127
... Type checking Schema box: SetEstimateStandsVolumeToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 2131-2147
... Type checking Schema definition: EstimateStandsVolume. "..\latex\MSc_specs_31-05-09.tex" Lines 2152-2153
... Type checking Schema box: DefineConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex"
    Lines 2171-2189
... Type checking Schema box: InitConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex" Lines 2193-2197
... Type checking Schema box: SetMassToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 2201-2209
... Type checking Schema definition: ConvertStandVolumeToMass. "..\latex\MSc_specs_31-05-09.tex"

```

```

Lines 2214-2215
... Type checking Schema box: StandsVolumeAndMassAtHarvesting. "..\latex\MSc_specs_31-05-09.tex"
Lines 2247-2300
... Type checking Schema box: HarvestStandAndUpdateRegimeOK. "..\latex\MSc_specs_31-05-09.tex" Lines 2320-2365
... Type checking Schema box: ExceptionHarvestStandAndUpdateRegime. "..\latex\MSc_specs_31-05-09.tex"
Lines 2379-2386
... Type checking Schema definition: HarvestStandAndUpdateRegime. "..\latex\MSc_specs_31-05-09.tex"
Lines 2391-2392
... Type checking Schema box: HarvestStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2403-2408
... Type checking Schema definition: PlantationActions. "..\latex\MSc_specs_31-05-09.tex" Line 2427
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 2455
... Type checking Schema box: DefineTripIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 2465-2466
... Type checking Schema box: InitTripIDs. "..\latex\MSc_specs_31-05-09.tex" Lines 2470-2473
... Type checking Schema definition: TripIDs. "..\latex\MSc_specs_31-05-09.tex" Line 2478
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 2489-2491
... Type checking Schema box: ShortHaulRoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2511-2534
... Type checking Schema box: LongHaulRoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2546-2568
... Type checking Schema definition: Road. "..\latex\MSc_specs_31-05-09.tex" Line 2581
... Type checking Schema box: TruckMaxLoad. "..\latex\MSc_specs_31-05-09.tex" Lines 2602-2610
... Type checking Schema box: Truck. "..\latex\MSc_specs_31-05-09.tex" Lines 2622-2636
... Type checking Schema box: ShortHaulTrip. "..\latex\MSc_specs_31-05-09.tex" Lines 2668-2682
... Type checking Schema box: LoadLogsAtStand. "..\latex\MSc_specs_31-05-09.tex" Lines 2706-2749
... Type checking Schema box: TransportAndUnloadLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 2771-2802
... Type checking Schema definition: ShortHaulTransport. "..\latex\MSc_specs_31-05-09.tex" Line 2862
... Type checking Schema box: LongHaulTrip. "..\latex\MSc_specs_31-05-09.tex" Lines 2895-2909
... Type checking Schema box: LoadLogsAtDepot. "..\latex\MSc_specs_31-05-09.tex" Lines 2933-2968
... Type checking Schema box: TransportAndUnloadLogsAtMill. "..\latex\MSc_specs_31-05-09.tex" Lines 2993-3025
... Type checking Schema definition: LongHaulTransport. "..\latex\MSc_specs_31-05-09.tex" Line 3087
... Type checking Schema definition: Transport. "..\latex\MSc_specs_31-05-09.tex" Line 3105
... Type checking Schema box: AcceptLogsFromOtherSuppliers. "..\latex\MSc_specs_31-05-09.tex" Lines 3126-3142
... Type checking Schema box: ProcessLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 3172-3181
... Type checking Schema box: ExceptionProcessLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 3185-3194
... Type checking Schema definition: RemoveLogsForProcessing. "..\latex\MSc_specs_31-05-09.tex" Line 3199
... Type checking Schema definition: ForestToMillSupplyChain. "..\latex\MSc_specs_31-05-09.tex"
Lines 3219-3220
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 3253
... Type checking Schema box: PlanningHorizonDefinition. "..\latex\MSc_specs_31-05-09.tex" Lines 3264-3271
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 3289
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 3293
... Type checking Schema box: EstimateStandWoodProp. "..\latex\MSc_specs_31-05-09.tex" Lines 3304-3318
... Type checking Schema box: MillRequirements. "..\latex\MSc_specs_31-05-09.tex" Lines 3339-3361
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 3423
... Type checking Schema box: ForestryCompanyLevelCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 3445-3453
... Type checking Schema box: EstateLevelCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 3467-3475
... Type checking Free type definition: REGIMEACTION. "..\latex\MSc_specs_31-05-09.tex" Line 3497
... Type checking Schema box: RegimeCanopyClosure. "..\latex\MSc_specs_31-05-09.tex" Lines 3508-3519
... Type checking Schema box: RegimeCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 3530-3559
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 3578
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 3589-3591
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 3640-3642
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Line 3747
... Type checking Axiom box. "..\latex\MSc_specs_31-05-09.tex" Lines 3758-3760
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 3900
... Type checking Schema box: Exponent. "..\latex\MSc_specs_31-05-09.tex" Lines 3911-3916
... Type checking Generic box: "..\latex\MSc_specs_31-05-09.tex" Lines 3926-3936
... Type checking Generic box: "..\latex\MSc_specs_31-05-09.tex" Lines 3946-3962
... Type checking Schema box: DefineNumberOfStandOptions. "..\latex\MSc_specs_31-05-09.tex" Lines 3980-3985
... Type checking Schema box: InitNumberOfStandOptions. "..\latex\MSc_specs_31-05-09.tex" Lines 3996-3999
... Type checking Schema box: SetNumberOfStandOptionsToOne. "..\latex\MSc_specs_31-05-09.tex" Lines 4003-4008
... Type checking Schema definition: NumberOfStandOptions. "..\latex\MSc_specs_31-05-09.tex" Lines 4013-4014
... Type checking Schema box: DefineStandRegimeAction. "..\latex\MSc_specs_31-05-09.tex" Lines 4033-4042
... Type checking Schema box: InitStandRegimeAction. "..\latex\MSc_specs_31-05-09.tex" Lines 4053-4056
... Type checking Schema definition: StandRegimeAction. "..\latex\MSc_specs_31-05-09.tex" Lines 4068-4069
... Type checking Schema box: DefineStandHarvestingInfo. "..\latex\MSc_specs_31-05-09.tex" Lines 4083-4096
... Type checking Schema box: InitStandHarvestingInfo. "..\latex\MSc_specs_31-05-09.tex" Lines 4107-4110
... Type checking Schema definition: StandHarvestingInfo. "..\latex\MSc_specs_31-05-09.tex" Lines 4122-4123
... Type checking Schema box: GenerateStandRegimeActionForUnplantedStands. "..\latex\MSc_specs_31-05-09.tex"
Lines 4154-4198

```

```

... Type checking Schema box: GenerateStandRegimeActionForPlantedStands. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4222-4274
... Type checking Schema definition: GenerateStandRegimeActionForStands. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4290-4292
... Type checking Schema box: UpdateHarvestingOutcomesUnplantedStands. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4311-4361
... Type checking Schema box: UpdateHarvestingOutcomesPlantedStands. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4378-4427
... Type checking Schema definition: UpdateHarvestingOutcomes. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4443-4444
... Type checking Given set. "..\latex\MSc_specs_31-05-09.tex" Line 4472
... Type checking Schema box: DefinePossibleSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 4486-4499
... Type checking Schema box: InitPossibleSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 4503-4506
... Type checking Schema definition: PossibleSolution. "..\latex\MSc_specs_31-05-09.tex" Line 4511
... Type checking Schema box: CalculatePossibleSolutions. "..\latex\MSc_specs_31-05-09.tex" Lines 4535-4552
... Type checking Schema box: DefineAnnualLogMassForMill. "..\latex\MSc_specs_31-05-09.tex" Lines 4580-4587
... Type checking Schema box: InitAnnualLogMassForMill. "..\latex\MSc_specs_31-05-09.tex" Lines 4591-4594
... Type checking Schema box: SetAnnualLogMassForMillToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 4598-4605
... Type checking Schema definition: AnnualLogMassForMill. "..\latex\MSc_specs_31-05-09.tex" Lines 4610-4611
... Type checking Schema box: CalculateAnnualLogMassForMill. "..\latex\MSc_specs_31-05-09.tex" Lines 4622-4641
... Type checking Free type definition: FEASIBILITY. "..\latex\MSc_specs_31-05-09.tex" Line 4670
... Type checking Schema box: DefineFeasibleSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 4684-4693
... Type checking Schema box: InitFeasibleSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 4697-4700
... Type checking Schema box: SetFeasibleSolutionToInfeasible. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4704-4709
... Type checking Schema definition: FeasibleSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 4714-4715
... Type checking Schema box: DetermineFeasibleSolutions. "..\latex\MSc_specs_31-05-09.tex" Lines 4726-4756
... Type checking Free type definition: FEASIBLESOLN. "..\latex\MSc_specs_31-05-09.tex" Line 4777
... Type checking Schema box: DefineFeasibleSolutionsExist. "..\latex\MSc_specs_31-05-09.tex" Lines 4790-4792
... Type checking Schema box: InitFeasibleSolutionsExist. "..\latex\MSc_specs_31-05-09.tex" Lines 4796-4800
... Type checking Schema box: DefineAllSolutionsInfeasible. "..\latex\MSc_specs_31-05-09.tex" Lines 4804-4810
... Type checking Schema definition: FeasibleSolutionsExist. "..\latex\MSc_specs_31-05-09.tex" Lines 4815-4816
... Type checking Schema box: DetermineIfFeasibleSolutionsExist. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4827-4837
... Type checking Schema box: DefineFirstRegimeActionInStandOption. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4867-4874
... Type checking Schema box: InitFirstRegimeActionInStandOption. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4878-4882
... Type checking Schema definition: FirstRegimeActionInStandOption. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4887-4888
... Type checking Schema box: DetermineFirstRegimeActionInStandOption. "..\latex\MSc_specs_31-05-09.tex"
    Lines 4900-4915
... Type checking Schema box: DefineYearVariables. "..\latex\MSc_specs_31-05-09.tex" Lines 4938-4939
... Type checking Schema box: InitYearVariables. "..\latex\MSc_specs_31-05-09.tex" Lines 4943-4948
... Type checking Schema definition: YearVariables. "..\latex\MSc_specs_31-05-09.tex" Line 4953
... Type checking Free type definition: FIRSTROTATION. "..\latex\MSc_specs_31-05-09.tex" Line 4964
... Type checking Schema box: DefineFirstRotation. "..\latex\MSc_specs_31-05-09.tex" Lines 4975-4976
... Type checking Schema box: InitFirstRotation. "..\latex\MSc_specs_31-05-09.tex" Lines 4980-4983
... Type checking Schema definition: FirstRotation. "..\latex\MSc_specs_31-05-09.tex" Line 4988
... Type checking Schema box: DefineGenMatID. "..\latex\MSc_specs_31-05-09.tex" Lines 5001-5002
... Type checking Schema box: InitGenMatID. "..\latex\MSc_specs_31-05-09.tex" Lines 5006-5009
... Type checking Schema definition: GenMatID. "..\latex\MSc_specs_31-05-09.tex" Line 5014
... Type checking Schema box: DefineRegimeStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5028-5034
... Type checking Schema box: InitRegimeStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5045-5048
... Type checking Schema box: SetRegimeStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5052-5058
... Type checking Schema definition: RegimeStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5070-5071
... Type checking Schema box: CalculateRegimeStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5089-5144
... Type checking Schema box: DefineSHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5175-5184
... Type checking Schema box: InitSHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5188-5191
... Type checking Schema box: SetSHaulTransportStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5195-5205
... Type checking Schema definition: SHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5210-5211
... Type checking Schema box: CalculateSHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"

```

```

    Lines 5224-5249
... Type checking Schema box: DefineLHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5269-5278
... Type checking Schema box: InitLHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5282-5285
... Type checking Schema box: SetLHaulTransportStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5289-5299
... Type checking Schema definition: LHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5304-5305
... Type checking Schema box: CalculateLHaulTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5316-5341
... Type checking Schema box: DefineTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5355-5361
... Type checking Schema box: InitTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5372-5375
... Type checking Schema box: SetTransportStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5379-5385
... Type checking Schema definition: TransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5397-5398
... Type checking Schema box: CalculateTransportStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5410-5431
... Type checking Schema box: DefineStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5457-5463
... Type checking Schema box: InitStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5474-5477
... Type checking Schema box: SetStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 5481-5487
... Type checking Schema definition: StandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5499-5500
... Type checking Schema box: CalculateStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5512-5523
... Type checking Schema box: DefineMillLogPurchases. "..\latex\MSc_specs_31-05-09.tex" Lines 5562-5569
... Type checking Schema box: InitMillLogPurchases. "..\latex\MSc_specs_31-05-09.tex" Lines 5573-5576
... Type checking Schema box: SetMillLogPurchasesToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 5580-5587
... Type checking Schema definition: MillLogPurchases. "..\latex\MSc_specs_31-05-09.tex" Lines 5592-5593
... Type checking Schema box: CalculateAnnualMillLogPurchases. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5605-5625
... Type checking Schema box: DefineDeliveredLogMillCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5647-5652
... Type checking Schema box: InitDeliveredLogMillCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5656-5659
... Type checking Schema box: SetDeliveredLogMillCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5663-5668
... Type checking Schema definition: DeliveredLogMillCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5673-5674
... Type checking Schema box: CalculateDeliveredLogMillCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5685-5701
... Type checking Schema box: DefinePossibleSolutionStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5723-5730
... Type checking Schema box: InitPossibleSolutionStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5734-5737
... Type checking Schema box: SetPSolutionStandOptionCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5741-5748
... Type checking Schema definition: PossibleSolutionStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5753-5754
... Type checking Schema box: CalculatePossibleSolutionStandOptionCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5765-5781
... Type checking Schema box: DefineAnnualEstateLevelCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5806-5813
... Type checking Schema box: InitAnnualEstateLevelCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5817-5820
... Type checking Schema box: SetAnnualEstateLevelCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5824-5831
... Type checking Schema definition: AnnualEstateLevelCosts. "..\latex\MSc_specs_31-05-09.tex" Lines 5836-5837
... Type checking Schema box: CalculateAnnualEstateLevelCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5848-5859
... Type checking Schema box: DefineTotalNonStandForestryCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5876-5884
... Type checking Schema box: InitTotalNonStandForestryCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5888-5892
... Type checking Schema box: SetTotalNonStandForestryCostsToZero. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5896-5903
... Type checking Schema definition: TotalNonStandForestryCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5908-5909
... Type checking Schema box: CalculateTotalNonStandForestryCosts. "..\latex\MSc_specs_31-05-09.tex"
    Lines 5920-5936
... Type checking Schema box: DefineObjectiveFunction. "..\latex\MSc_specs_31-05-09.tex" Lines 5954-5955
... Type checking Schema box: InitObjectiveFunction. "..\latex\MSc_specs_31-05-09.tex" Lines 5959-5962
... Type checking Schema box: SetObjectiveFunctionToZero. "..\latex\MSc_specs_31-05-09.tex" Lines 5966-5971

```

```
... Type checking Schema definition: ObjectiveFunction. "..\latex\MSc_specs_31-05-09.tex" Lines 5976-5977
... Type checking Schema box: CalculateObjectiveFunction. "..\latex\MSc_specs_31-05-09.tex" Lines 5988-6010
... Type checking Schema box: DefineOptimalSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 6066-6067
... Type checking Schema box: InitOptimalSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 6071-6074
... Type checking Schema definition: OptimalSolution. "..\latex\MSc_specs_31-05-09.tex" Line 6079
... Type checking Schema box: DetermineOptimalSolution. "..\latex\MSc_specs_31-05-09.tex" Lines 6090-6110
... Type checking Schema box: UpdateMillForStandsLogs. "..\latex\MSc_specs_31-05-09.tex" Lines 6131-6154
... Type checking Schema box: FHSSPart1. "..\latex\MSc_specs_31-05-09.tex" Lines 6173-6181
... Type checking Schema box: FHSSPart2. "..\latex\MSc_specs_31-05-09.tex" Lines 6198-6211
... Type checking Schema box: FHSS. "..\latex\MSc_specs_31-05-09.tex" Lines 6222-6239
End of main file: ..\latex\MSc_specs_31-05-09.tex
Type report written in ".typ"
Log written in ".log"
```

Log closed at: Mon Jun 01 06:45:04 2009

Appendix F

Literature concept matrix

The following table categorises the literature cited in this dissertation in the five main categories: forestry (agricultural aspects), forest management, wood and pulp properties, Operations Research and specifying information systems. The list of references can be found on page 407.

Table F.1: Literature concept matrix

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
Abrial <i>et al.</i> (1996)					✓
Alagar and Periyasamy (1998)					✓
Almeida <i>et al.</i> (2004)		✓			
Andalaft <i>et al.</i> (2003)		✓		✓	
Andersson (2005)		✓		✓	
Andersson and Eriksson (2007)		✓		✓	
American Paper and Pulp Association (1965)			✓		
Ashworth (1988)					✓
Bachman (1969)					✓
Baillères <i>et al.</i> (1997)			✓		
Barber (1983)		✓		✓	
Barden <i>et al.</i> (1992)					✓
Barden <i>et al.</i> (1994)					✓
Bare (1996)		✓			
Barros and Weintraub (1982)	✓	✓		✓	
Baskent and Keles (2005)		✓		✓	
Baskent <i>et al.</i> (2001)		✓			✓
Beaudoin <i>et al.</i> (2008)		✓		✓	
Beaudoin <i>et al.</i> (2007)		✓		✓	
Bert <i>et al.</i> (2003)					✓
Bettinger and Chung (2004)		✓		✓	
Bevens and Barrett (2005)		✓		✓	
Bjørner (1998)					✓
Bjørner (1999)					✓
Bjørner (2006a)					✓
Bjørner (2006b)					✓
Bjørner (2007)					✓
Bjørner <i>et al.</i> (1997)					✓

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
Boehm (1985)					✓
Boehm (1988)					✓
Böhlen <i>et al.</i> (1998)		✓			✓
Bottaci and Jones (1995)					✓
Bowen (1996)					✓
Bowen (1998)					✓
Bowen and Hinchey (1995a)					✓
Bowen and Hinchey (1995b)					✓
Bowen and Hinchey (2006)					✓
Brink and Kellogg (2000)		✓			
Brooks (Jr.) (1986)					✓
Bruel <i>et al.</i> (1998)					✓
Brumelle <i>et al.</i> (1998)		✓		✓	
Bryant (1995)					✓
Bryant (1990)					✓
Buford (1991)		✓		✓	
Burger and Jamnick (1995)		✓		✓	
Burton-Jones and Meso (2008)					✓
Business Rules Group (2000)					✓
Carle <i>et al.</i> (2002)		✓			
Carlsson <i>et al.</i> (2006)		✓		✓	
Carlsson and Rönnqvist (2005)		✓		✓	
Cea and Jofré (2000)		✓		✓	
Chappelle <i>et al.</i> (1976)		✓		✓	
Chen (1976)					✓
Chen (1983)					✓
Church <i>et al.</i> (2000)		✓		✓	
Church <i>et al.</i> (1994)		✓		✓	
Clarke (1995)			✓		
Clarke (2001)		✓	✓		
Clarke <i>et al.</i> (1996)					✓
Coad and Yourdon (1991)					✓
Collins <i>et al.</i> (1990)			✓		
Colodette <i>et al.</i> (2004)			✓		
Cooke <i>et al.</i> (1996)					✓
Cox <i>et al.</i> (2005)					✓
Crafford <i>et al.</i> (1998)			✓		
Craigen <i>et al.</i> (1993)					✓
Craigen <i>et al.</i> (1995)					✓
Curtis <i>et al.</i> (1988)					✓
Davis (1988)					✓
Davis and Martell (1993)		✓		✓	
Davis and Martell (1996)		✓		✓	
Dean (1997)					✓
Dean and Hinchey (1995)					✓
DeMarco (1979)					✓
DeMarco (1997)					✓
Diller (1994)					✓
Dorfman (1990)					✓
du Plooy (1980)			✓		
Duchesne <i>et al.</i> (1997)		✓	✓		
Dyck (2003)		✓			
Dye (2001)		✓			
Dyer <i>et al.</i> (1997)			✓		
Easton <i>et al.</i> (2003)	✓	✓	✓	✓	✓
Eid and Hobbestad (2000)		✓		✓	
Epstein <i>et al.</i> (1999a)		✓		✓	
Epstein <i>et al.</i> (1999b)		✓		✓	
EU-AgriNet (2000)			✓		
Evans (1997)	✓	✓			
Evans and Turnbull (2004)	✓	✓			
Everard (2000)	✓				

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
FAO (2006)		✓			
FAO (2007)		✓			
Faulk (1996)					
Federal register (1982)		✓			
Field (1984)		✓		✓	
Fins (2008)	✓				
Ford-Robertson (1971)	✓	✓			
Forest Stewardship Council (2003a)	✓	✓			
Forest Stewardship Council (2003b)	✓	✓			
France and Docker (1988)					✓
France and Larrondo-Petrie (1995)					✓
Gane and Sarson (1979)					✓
Garcia (1988)		✓		✓	
Garcia (1984)		✓		✓	
Garcia (1990)		✓		✓	
Gartner (2005)			✓		
Gaudel (1994)					✓
George and Vaughn (2003)					✓
Gordon <i>et al.</i> (2006)		✓	✓	✓	
Goyal <i>et al.</i> (1999)			✓		
Gravell (1991)					✓
Greenspan <i>et al.</i> (1982)					✓
Grzeskowiak and Turner (2000)			✓		
Guignard <i>et al.</i> (1998)		✓		✓	
Gunn (1991)		✓		✓	
Gunn and Rai (1987)		✓		✓	
Gutttag and Horning (1993)					✓
Hall (1996)					✓
Hall (1990)					✓
Harel (1987)					✓
Hay (2003)					✓
Hay (2004)					✓
Hayes (1987)					✓
Hayes (1993)					✓
Heitmeyer and McLean (1983)					✓
Herbert (2000)	✓				
Higgins (1970)			✓		
Hinchey (2002)					✓
Hinchey and Bowen (1999)					✓
Hoare (1985)					✓
Holland <i>et al.</i> (1994)					✓
Hultqvist and Olsson (2004)		✓		✓	
Hultqvist and Olsson (2005)		✓		✓	
Hultqvist and Olsson (2006)		✓		✓	
Hutter <i>et al.</i> (1999)					✓
IEEE Standards Board (1994)					✓
Ilgun <i>et al.</i> (1995)					✓
Ince (1990)					✓
Ivkovich (2000)			✓		
Jackson (1983)					✓
Jackson (1995)					✓
Jackson (2001)					✓
Jackson (2005)					✓
Jackson and Megraw (1986)			✓		
Jackson and Zave (1993)					✓
Jacky (1997)					✓
Jia (2002)					✓
Johnson <i>et al.</i> (1980)		✓		✓	
Johnson <i>et al.</i> (1986)		✓		✓	
Jones (1990)					✓
Jones (1996)					✓
Jones and Ohlmann (2008)		✓		✓	

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
Kang <i>et al.</i> (1990)					✓
Kanowski (1997)	✓	✓			
Kardasis and Loucopoulos (2004)					✓
Karlsson (2006)			✓		
Karlsson <i>et al.</i> (2003)		✓		✓	
Karlsson <i>et al.</i> (2004)		✓		✓	
Kennedy (1995)			✓		
Kent <i>et al.</i> (1991)		✓		✓	
Kirilenko and Sedjo (2007)		✓			
Kube and Raymond (2002)			✓		
Landsberg <i>et al.</i> (2001)		✓			
Landsberg and Waring (1997)		✓			
Landsberg <i>et al.</i> (2003)		✓			
Laroze and Greber (1991)		✓		✓	
Larson (1967)	✓		✓		
Ledru (1996)					✓
Leuschner (1990)	✓	✓			
Lightfoot (2001)					✓
Louw (2000a)	✓	✓			
Louw (2000b)	✓	✓			
Lundqvist (2003)			✓		
Lundqvist <i>et al.</i> (2003)			✓		
Luqi and Goguen (1997)					✓
Malan (2003)			✓		
Mander and Polack (1995)					✓
Manley <i>et al.</i> (1996)		✓		✓	
Manley and Threadgill (1991)		✓		✓	
Mansfield and Weineisen (2007)			✓		
Martell <i>et al.</i> (1996)		✓		✓	
Martell <i>et al.</i> (1998)		✓		✓	
Mathey <i>et al.</i> (2008)		✓			
McDill <i>et al.</i> (2002)		✓		✓	
McDonough (1998)			✓		
McGuigan and Scott (1995)		✓		✓	
McGuigan (1984)		✓	✓	✓	
McMenamin and Palmer (1984)					✓
McNaughton (1998)		✓		✓	
McNaughton <i>et al.</i> (1998)		✓		✓	
McNaughton <i>et al.</i> (2000)		✓		✓	
Megown <i>et al.</i> (2001)		✓	✓		
Megown <i>et al.</i> (2000)		✓	✓		
Meyer (1985)					✓
Meynink and Borough (2005)		✓	✓		
Mikkonen (1996)		✓	✓		
Milner (1980)					✓
Miranda and Pereira (2002)			✓		
Mitchell (2004)		✓		✓	
Morales <i>et al.</i> (1994)		✓		✓	
Morales and Weintraub (1991)		✓		✓	
Morgan and Sufirin (1993)					✓
Morgan (2002)					✓
MSU (2004)					✓
Muneri (1994)			✓		
Murray and Church (1995)		✓		✓	
Mylopoulos (1992)					✓
Naidu (2003)			✓		
Nash (1990)					✓
Navon (1971)		✓		✓	
Nelson <i>et al.</i> (1991)		✓		✓	
Nightingale <i>et al.</i> (2008)		✓			
Nix and Collins (1988)					✓
Nobre and Rodriguez (2005)	✓	✓			✓

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
Öhman and Eriksson (2002)		✓		✓	
Owen (2000a)	✓				
Owen (2000b)	✓	✓			
Paavilainen (1994)			✓		
Papps and Manley (1992)		✓		✓	
Parnas (1969)					✓
Pearlson and Saunders (2006)					✓
Philpott and Everett (2001)		✓		✓	
Plat <i>et al.</i> (1991)					✓
Pohl (1993)					✓
Polack (2001)					✓
Polack <i>et al.</i> (1993)					✓
Price <i>et al.</i> (2008a)	✓	✓	✓	✓	
Price <i>et al.</i> (2008b)	✓				✓
Price <i>et al.</i> (2009)	✓				✓
Pulkki (2001)		✓			
Raven <i>et al.</i> (1992)			✓		
Retief <i>et al.</i> (1997)			✓		
Ribeiro <i>et al.</i> (2005)	✓	✓			✓
Robertson and Robertson (2006)					✓
Robinson (2002)					✓
Rönnqvist (2003)	✓	✓		✓	
Rosca <i>et al.</i> (1997)					✓
Ross (1977)					✓
Royce (1970)					✓
Saaltink (1997)					✓
Sands and Landsberg (2002)		✓			
Savill <i>et al.</i> (1997)	✓				
Sedjo (1999)		✓			
Sefara <i>et al.</i> (2001)			✓		
Semmens and Allen (1991)					✓
Semmens <i>et al.</i> (1992)					✓
Shepherd (1986)	✓	✓			
Siau (2004)					✓
Smith and Cunningham (2002)	✓				
Smith (1978)		✓		✓	
Sommerville (1996)					✓
Sommerville (2007)					✓
Sommerville and Sawyer (1997a)					✓
Sommerville and Sawyer (1997b)					✓
Spångberg (1999)		✓	✓		
Spivey (1998)					✓
Stadtler (2005)		✓			
Stepney <i>et al.</i> (2003)					✓
Sutcliffe and Maiden (1998)					✓
Sykes (1982)					
Syndicate Database Solutions (2006)		✓			
Tedder <i>et al.</i> (1978)		✓		✓	
Thayer and Dorfman (1990)					✓
The RAISE Language Group (1992)					✓
Tretmans <i>et al.</i> (2001)					✓
Turner <i>et al.</i> (2002)		✓		✓	
Turner <i>et al.</i> (2005)			✓		
Turner <i>et al.</i> (1999)			✓		
Turner <i>et al.</i> (2000)		✓	✓		
Turner and Price (1996)		✓	✓		
Turner and Price (2005)	✓	✓	✓	✓	
Turner and Retief (1998)			✓		
USDA Forest Service (1996)		✓		✓	
Utting (1995)					✓
Valentine (1995)					✓
van Eijk <i>et al.</i> (1989)					✓

Literature reference	Forestry	Forest management	Wood and pulp properties	Operations Research	Specifying information systems
Vielma <i>et al.</i> (2007)		✓		✓	
von Gadow and Bredenkamp (1992)	✓	✓		✓	
von Halle (2002)					✓
Walker (1974)		✓		✓	
Wallis <i>et al.</i> (1996)			✓		
Wasserman (1985)					✓
Weaver <i>et al.</i> (1998)					✓
Weigel (2005)		✓	✓	✓	
Weintraub (2007)		✓		✓	
Weintraub and Bare (1996)		✓		✓	
Weintraub and Cholaky (1991)		✓		✓	
Weintraub <i>et al.</i> (2000)		✓		✓	
Weintraub and Davis (1996)		✓		✓	
Weintraub <i>et al.</i> (1994)		✓		✓	
Weintraub <i>et al.</i> (1995)		✓		✓	
Weintraub and Murray (2006)		✓		✓	
Wessels and Price (1999)				✓	
Wessels <i>et al.</i> (2006)				✓	
White (2004)					✓
Wieringa (1998)					✓
Williams (1994)			✓		
Wilson and White (1986)			✓		
Wimmer <i>et al.</i> (2002)			✓		
Wing (1990)					✓
Winters (1977)	✓	✓			
Woodcock and Davies (1996)					✓
Woodcock (1989)					✓
Wordsworth (1992)					✓
Xu <i>et al.</i> (1997)			✓		
Yeh and Zave (1980)					✓
Yoshimoto <i>et al.</i> (1994)		✓		✓	
Yourdon (1989)					✓
Z Archive (1996)					✓
Zachman (1987)					✓
Zave (1984)					✓
Zave and Jackson (1996)					✓
Zbonak (2005)			✓		
Zobel <i>et al.</i> (1983)		✓	✓		
Zobel and van Buijtenen (1989)		✓	✓		
Zwolinski and Hinze (2000)	✓				

References

- Abrial, J.R., Börger, E. and Langmaack, H., eds. (1996). *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*. LNCS 1165. Springer-Verlag, Berlin Heidelberg, Germany. 54, 401
- Alagar, V.S. and Periyasamy, K. (1998). *Specification of Software Systems*. Springer-Verlag New York Inc., USA. 42, 54, 401
- Almeida, A.C., Landsberg, J.J. and Sands, P.J. (2004). Parameterisation of 3-PG model for fast-growing *Eucalyptus grandis* plantations. *Forest Ecology and Management*, 193(1–2):179–195. 105, 401
- American Paper and Pulp Association (1965). *The Dictionary of Paper – including Pulp, Paperboard, Paper Properties and Related Papermaking Terms*. American Paper and Pulp Association, 3rd edn. xxiv, 401
- Andalaft, N., Andalaft, P., Guignard, M., Magendzo, A., Wainer, A. and Weintraub, A. (2003). A problem of forest harvesting and road building solved through model strengthening and Lagrangean relaxation. *Operations Research*, 51(4):613–628. 11, 13, 14, 17, 22, 28, 231, 235, 239, 245, 251, 401
- Andersson, D. (2005). *Approaches to integrated strategic/tactical forest planning*. Licentiate thesis, Department of Forest Resource Management and Geomatics, Faculty of Forest Science, Swedish University of Agricultural Sciences, Umeå, Sweden. 33, 45, 226, 401
- Andersson, D. and Eriksson, L.O. (2007). Effects of temporal aggregation in integrated strategic/tactical and strategic forest planning. *Forest Policy and Economics*, 9(8):965–981. 24, 29, 401
- Ashworth, C. (1988). Structured systems analysis and design method (SSADM). *Information and Software Technology*, 30(3):153–163. 36, 401
- Bachman, C.W. (1969). Data structure diagrams. *ACM SIGMIS Database*, 1(2):4–10. 51, 401
- Baillères, H., Chanson, B. and Fournier-Djimbi, M. (1997). Man-made stands of quick-growing species and forest product quality in tropical countries. In: *Afforestation and Plantation Forestry for the 21st Century: Proceedings of the FAO XI World Forestry Congress, 13–22 October 1997*, pp. 47–54. FAO. 15, 401

- Barber, R.L. (1983). HARVEST, an interactive harvest scheduling simulator for even-aged forests. *Journal of Forestry*, 81(8):532–533. 25, 229, 233, 237, 242, 249, 401
- Barden, R., Stepney, S. and Cooper, D. (1992). The use of Z. In: J.E. Nicholls, ed., *Z User Workshop: Proceedings of the Sixth Annual Z User Meeting, York, 16–17 December 1991*, pp. 99–124. Springer-Verlag, London, UK. 38, 39, 47, 60, 211, 212, 401
- Barden, R., Stepney, S. and Cooper, D. (1994). *Z in practice*. Prentice Hall International (UK) Limited, Hemel Hempstead, Hertfordshire, UK. 37, 38, 39, 40, 43, 47, 54, 59, 60, 61, 209, 210, 212, 213, 216, 217, 224, 401
- Bare, B.B. (1996). *Hierarchical forest planning: some general observations*, pp. 164–165. In: Martell *et al.* (1996). 25, 30, 31, 401
- Barros, O. and Weintraub, A. (1982). Planning for a vertically integrated forest industry. *Operations Research*, 30(6):1168–1182. 9, 11, 12, 22, 24, 25, 26, 105, 230, 234, 238, 243, 250, 401
- Baskent, E.Z. and Keles, S. (2005). Spatial forest planning: A review. *Ecological Modelling*, 188(2–4):145–173. 27, 401
- Baskent, E.Z., Wightman, R.A., Jordan, G.A. and Zhai, Y. (2001). Object-oriented abstraction of contemporary forest management design. *Ecological Modelling*, 143(3):147–164. 43, 209, 401
- Beaudoin, D., Frayret, J.M. and LeBel, L. (2008). Hierarchical forest management with anticipation: an application to tactical-operational planning integration. *Canadian Journal of Forest Research*, 38(8):2198–2211. 23, 208, 220, 401
- Beaudoin, D., LeBel, L. and Frayret, J.M. (2007). Tactical supply chain planning in the forest products industry through optimization and scenario-based analysis. *Canadian Journal of Forest Research*, 37(1):128–140. 13, 30, 44, 401
- Bert, D., Bowen, J.P., King, S. and Waldén, M., eds. (2003). *ZB 2003: Formal Specification and Development in Z and B, Third International Conference of B and Z Users, Turku, Finland, 4–6 June 2003*. LNCS 2651. Springer-Verlag Berlin Heidelberg, Germany. 54, 401
- Bettinger, P. and Chung, W. (2004). The key literature of, and trends in, forest-level management planning in North America, 1950–2001. *The International Forestry Review*, 6(1):40–50. 28, 31, 401
- Bevens, M. and Barrett, T.M., eds. (2005). *System Analysis in Forest Resources: Proceedings of the 2003 Symposium*. General Technical Report PNW-GTR-656, USDA Forest Service, Pacific Northwest Research Station, Portland OR, USA. 401, 417, 423, 424

- Bjørner, D. (1998). Domains as a prerequisite for requirements and software domain perspectives and facets, requirements aspects and software views. In: M. Broy and B. Rumpe, eds., *Proceedings of the International Workshop on Requirements Targeting Software and Systems Engineering (RTSE '97), Bernried, Germany, 12–14 October 1997*, LNCS 1526, pp. 1–41. Springer-Verlag Berlin Heidelberg, Germany. 43, 401
- Bjørner, D. (1999). A triptych software development paradigm: domain, requirements and software towards a model development of a decision support system for sustainable development. In: E.R. Olderog and B. Steffen, eds., *Correct System Design: Recent Insights and Advances*, LNCS 1710, pp. 29–60. Springer-Verlag, Berlin Heidelberg, Germany. 43, 401
- Bjørner, D. (2006a). *Software Engineering 1: Abstraction and Modelling*. Springer-Verlag, Berlin, Germany. 42, 47, 401
- Bjørner, D. (2006b). *Software Engineering 3: Domains, Requirements, and Software Design*. Springer-Verlag, Berlin, Germany. 42, 43, 47, 401
- Bjørner, D. (2007). Domain engineering. <<http://www2.imm.dtu.dk/~db/5lectures/>> accessed on 17 January 2008. 42, 47, 401
- Bjørner, D., Koussobe, S., Noussi, R. and Satchok, G. (1997). Michael Jackson's problem frames: towards methodological principles of selecting and applying formal software development techniques and tools. In: L. ShaoQi and M. Hinchley, eds., *Proceedings of the International Conference on Formal Engineering Methods (ICFEM 97), 12–14 November 1997*, pp. 263–270. IEEE Computer Society Press. 42, 47, 401
- Boehm, B. (1985). Industrial metrics top 10 list. *IEEE Software*, 4(5). 36, 213, 402
- Boehm, B.W. (1988). A spiral model of software development and enhancement. *IEEE Computer*, 21(5):61–72. 35, 402
- Böhlen, M., Jensen, C.S. and Skjellaug, B. (1998). Spatio-temporal database support for legacy applications. In: *Proceedings of the 1998 ACM Symposium on Applied Computing*, pp. 226–234. ACM. 43, 402
- Bottaci, L. and Jones, J. (1995). *Formal Specification Using Z – a Modelling Approach*. International Thomson Publishing, London, UK. 54, 59, 402
- Bowen, J. (1996). *Formal Specification and Documentation Using Z: A Case Study Approach*. International Thompson Computer Press, London, UK. 54, 59, 402
- Bowen, J.P. (1998). Select Z bibliography. In: J.P. Bowen, A. Fett and M.G. Hinchey, eds., *The Z formal specification notation (ZUM '98), Proceedings of the 11th International Conference of Z Users, Berlin, Germany, 24–26 September 1998*, LNCS 1783, pp. 367–411. Springer-Verlag, Berlin Heidelberg, Germany. 54, 402

- Bowen, J.P. and Hinchey, M.G. (1995a). Seven more myths of formal methods. *IEEE Software*, 12(4):34–40. 37, 39, 40, 48, 215, 217, 402
- Bowen, J.P. and Hinchey, M.G. (1995b). Ten commandments of formal methods. *IEEE Computer*, 28(4):56–63. 36, 38, 39, 40, 41, 42, 53, 54, 61, 151, 210, 402
- Bowen, J.P. and Hinchey, M.G. (2006). Ten commandments of formal methods . . . ten years later. *IEEE Computer*, 39(1):40–48. 38, 40, 213, 216, 225, 402
- Brink, M. and Kellogg, L.D. (2000). *Planning the forest engineering value chain – a systems approach*, pp. 275–284. Vol. 1 of Owen (2000b). 12, 402
- Brooks (Jr.), F.P. (1986). No silver bullet – essence and accidents of software engineering. In: H.K. Kugler, ed., *Information Processing 86 (IFIP 86)*, pp. 1069–1076. Elsevier Science Publishers B.V. (North-Holland). 36, 402
- Bruel, J.M., Cheng, B., Easterbrook, S., France, R. and Rumpe, B. (1998). Integrating formal and informal specification techniques. Why? How? In: *2nd IEEE Workshop on Industrial Strength Formal Specification Techniques, 22 October 1998*, pp. 50–57. IEEE Computer Society, Washington DC, USA. 39, 48, 217, 402
- Brumelle, S., Granot, D., Halme, M. and Vertinsky, I. (1998). A tabu search algorithm for finding good forest harvest schedules satisfying green-up constraints. *European Journal of Operational Research*, 106(2–3):408–424. 27, 402
- Bryant, A. (1995). A pragmatic approach to harnessing formal specification. In: *IEE Colloquium on Practical Applications of Formal Methods, 19 May 1995*, pp. 1/1–1/7. IEEE. 37, 39, 402
- Bryant, T. (1990). Structured methodologies and formal notations: developing a framework for synthesis and investigation. In: J.E. Nichols, ed., *Z User Workshop: Proceedings of the Fourth Annual Z User Meeting, 15 December 1989*, pp. 229–241. Springer-Verlag, London Berlin Heidelberg New York. 40, 47, 208, 216, 402
- Buford, M.A., ed. (1991). *Proceedings of the 1991 Symposium on Systems Analysis in Forest Resources*. General Technical Report SE-74, USDA Forest Service Southeastern Forest Experiment Station, Asheville NC, USA. 402, 416, 419, 422
- Burger, D.H. and Jamnick, M.S. (1995). Using linear programming to make wood procurement and distribution decisions. *The Forestry Chronicle*, 71(1):89–96. 10, 232, 236, 240, 247, 252, 402
- Burton-Jones, A. and Meso, P. (2008). The effects of decomposition quality and multiple forms of information on novices understanding of a domain from a conceptual model. *Journal of the Association for Information Systems*, 9(12):748–802. 43, 224, 402

- Business Rules Group (2000). Defining business rules – what are they really? <http://www.businessrulesgroup.org/first_paper/BRG-whatBR_3ed.pdf> accessed on 7 February 2008. 52, 402
- Carle, J., Vuorinen, P. and Del Lungo, A. (2002). Status and trends in global forest plantation development. *Forest Products Journal*, 52(7–8):12–23. xvi, 8, 9, 402
- Carlsson, D., D'Amours, S., Martel, A. and Rönnqvist, M. (2006). *Supply chain management in the pulp and paper industry*. Working paper DT-2006-AM-3, Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT), Université Laval, Québec, Canada. 31, 402
- Carlsson, D. and Rönnqvist, M. (2005). Supply chain management in forestry – case studies at Sodra Cell AB. *European Journal of Operational Research*, 163(3):589–616. 13, 44, 402
- Cea, C. and Jofré, A. (2000). Linking strategic and tactical forestry planning decisions. *Annals of Operations Research*, 95(1–4):131–158. 22, 23, 24, 27, 31, 33, 45, 226, 230, 234, 238, 244, 250, 402
- Chappelle, D.E., Mang, M. and Miley, R.C. (1976). Evaluation of Timber RAM as a forest management planning model. *Journal of Forestry*, 74(5):288–293. 25, 229, 233, 237, 242, 249, 402
- Chen, P.P.S. (1976). The entity-relationship model – toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36. 36, 402
- Chen, P.P.S. (1983). English sentence structure and entity-relationship diagrams. *Information Sciences*, 29(2–3):127–149. 50, 402
- Church, R.L., Murray, A.T. and Barber, K.H. (2000). Forest planning at the tactical level. *Annals of Operations Research*, 95(1–4):3–18. 25, 229, 402
- Church, R.L., Murray, R.L. and Barber, K. (1994). Designing a hierarchical planning model for USDA Forest Service planning. In: J. Sessions and J.D. Brodie, eds., *Proceedings of the 1994 Symposium on Systems Analysis and Forest Resources*, pp. 401–409. Society of American Foresters, Bethesda MD, USA. 13, 24, 26, 27, 31, 402
- Clarke, C.R.E. (1995). *Variation in growth, wood, pulp and paper properties of nine eucalypt species with commercial potential in South Africa*. Ph.D thesis, Forestry, University College of North Wales, Bangor, UK. 14, 18, 402
- Clarke, C.R.E. (2001). Are *Eucalyptus* clones advantageous for the pulp mill? *Southern African Forestry Journal*, 190:61–65. 16, 402
- Clarke, E.M., Wing, J.M., Alur, R., Cleaveland, R., Dill, D., Emerson, A., Garland, S., German, S., Guttag, J., Hall, A., Henzinger, T., Holzmann, G., Jones, C., Kurshan, R., Leveson, N., McMillan, K., Moore, J., Peled, D., Pnueli, A., Rushby, J., Shankar,

- N., Sifakis, J., Sistla, P., Steffen, B., Wolper, P., Woodcock, J. and Zave, P. (1996). Formal methods: state of the art and future directions. *ACM Computing Surveys*, 28(4):626–643. 41, 402
- Coad, P. and Yourdon, E. (1991). *Object Oriented Analysis*. Yourdon Press, Englewood Cliffs NJ, USA. 36, 402
- Collins, D.J., Pilotti, C.A. and Wallis, A.F.A. (1990). Correlation of chemical composition and kraft pulping properties of some Papua New Guinea reforestation woods. *Appita Journal*, 43(3):193–198. 18, 402
- Colodette, J.L., Eiras, K.M.M., Oliveira, E. and Ventorim, G. (2004). Influence of eucalypt wood supply on pulp brightness stability. *Appita Journal*, 57(6):481–487. 17, 402
- Cooke, D., Gates, A., Demirörs, E., Demirörs, O., Tanik, M.M. and Krämer, B. (1996). Languages for the specification of software. *Journal of Systems and Software*, 32(3):269–308. 54, 59, 402
- Cox, K., Hall, J.G. and Rapanotti, L. (2005). A roadmap of problem frames research (editorial). *Information and Software Technology*, 47(14):891–902. 42, 402
- Crafford, J.G., Grzeskowiak, V., Turner, P. and Megown, R. (1998). *Second report on the image analysis evaluation of the anatomical properties of the Eucalyptus grandis TAG 5 clone and their influence on pulp and paper properties*. Confidential Research Report ENV-P-I 98159, CSIR ENVIRONMENTEK, Pretoria, South Africa. 15, 402
- Craigen, D., Gerhart, S. and Ralston, T. (1993). *An International Survey of Industrial Applications of Formal Methods*. Technical report NIST GCR-626, Department of Commerce, U.S. National Institute of Standards and Technology. 37, 54, 55, 402
- Craigen, D., Gerhart, S. and Ralston, T. (1995). Formal methods reality check: industrial usage. *IEEE Transactions on Software Engineering*, 21(2):90–98. 39, 41, 54, 402
- Curtis, B., Krasner, H. and Iscoe, N. (1988). A field study of the software design process for large systems. *Communications of the ACM*, 31(11):1268–1287. 36, 402
- Davis, A.M. (1988). A comparison of techniques for the specification of external system behaviour. *Communications of the ACM*, 31(9):1098–1115. 52, 402
- Davis, R.G. and Martell, D.L. (1993). A decision support system that links short-term silvicultural operating plans with long-term forest-level strategic plans. *Canadian Journal of Forest Research*, 23(6):1078–1095. 22, 32, 45, 402
- Davis, R.G. and Martell, D.L. (1996). *A decision support system to help forest managers evaluate silvicultural strategies and tactics*, pp. 148–154. In: Martell et al. (1996). 32, 45, 402

- Dean, N. (1997). *The Essence of Discrete Mathematics*. Pearson Education Limited, Harlow, Essex, UK. xxxv, 402
- Dean, N. and Hinchey, M.G. (1995). Introducing formal methods through role-playing. In: *Proceedings of the twenty-sixth SIGCSE Technical Symposium on Computer Science Education (SIGCSE '95), Nashville TN, USA, 2–4 March 1995*, pp. 302–306. ACM, New York NY, USA. 54, 402
- DeMarco, T. (1979). *Structured Analysis and System Specification*. Prentice-Hall, Inc., Englewood Cliffs NJ, USA. 42, 402
- DeMarco, T. (1997). *The Deadline – a Novel about Project Management*. Dorset House Publishing, New York NY, USA. 42, 136, 402
- Diller, A. (1994). *Z: an Introduction to Formal Methods*. John Wiley and Sons Ltd., Chichester, UK, 2nd edn. 54, 59, 402
- Dorfman, M. (1990). *System and software requirements engineering*, pp. 4–16. In: Thayer and Dorfman (1990). 42, 402
- du Plooy, A.B.J. (1980). The relationship between wood and pulp properties of *E. grandis* (Hill ex-Maiden) grown in South Africa. *Appita Journal*, 33(4):257–264. 18, 402
- Duchesne, I., Wilhelmsson, L. and Spångberg, K. (1997). Effects of in-forest sorting of Norway spruce (*Picea abies*) and Scots pine (*Pinus sylvestris*) on wood and fibre properties. *Canadian Journal of Forest Research*, 27(5):790–795. 21, 44, 402
- Dyck, B. (2003). Benefits of planted forests: social, ecological and economic. In: *UNFF Intersessional Experts Meeting on the Role of Planted Forests in Sustainable Forest Management, 24–30 March 2003*, pp. 1–10. 8, 402
- Dye, P.J. (2001). Modelling growth and water use in four *Pinus patula* stands with the 3-PG model. *Southern Forests: a Journal of Forest Science*, 191:53–64. 105, 402
- Dyer, S.T., Crafford, J. and Turner, P. (1997). The impact of the environment on the anatomical properties of a *Eucalyptus grandis* clone. In: *Proceedings of the 27th Microscopy Society of Southern Africa*, p. 71. MSSA. 15, 402
- Easton, H., Maunga, I., Tucker, K.C., Dell, M.P., Price, C.S. and Turner, P. (2003). *Harvest Optimisation Tool – System Requirement Specification*. Confidential Research Report ENV-D-C 2004-037, CSIR iCOMTEK and CSIR ENVIRONMENTEK, Durban, South Africa. 3, 46, 402
- Eid, T. and Hobbelstad, K. (2000). AVVIRK-2000: A large-scale forestry scenario model for long-term investment, income and harvest analyses. *Scandinavian Journal of Forest Research*, 15(4):472–482. 25, 229, 233, 237, 242, 249, 402

- Epstein, R., Morales, R., Serón, J. and Weintraub, A. (1999a). Use of OR systems in the Chilean forest industries. *Interfaces*, 29(1):7–29. 12, 14, 22, 24, 25, 28, 29, 70, 229, 231, 232, 233, 235, 236, 237, 239, 240, 242, 245, 247, 249, 251, 252, 402
- Epstein, R., Nieto, E., Weintraub, A., Chevalier, P. and Gabarró, J. (1999b). A system for the design of short term harvesting strategy. *European Journal of Operational Research*, 119(2):427–439. 22, 29, 232, 236, 240, 247, 252, 402
- EU-AgriNet (2000). EUROFIBER – fiber variability of European spruce and wood assortments for improved TMP production. European Commission, EU-funded Agricultural Portal. <http://ec.europa.eu/research/agriculture/projects/glrt_1999_01520_en.htm> accessed on 1 February 2007. 21, 402
- Evans, J. (1997). The sustainability of wood production in plantation forestry. In: *Afforestation and Plantation Forestry for the 21st Century: Proceedings of the FAO XI World Forestry Congress, 13–22 October 1997*, vol. 3, pp. 23–33. FAO. 11, 402
- Evans, J. and Turnbull, J.W. (2004). *Plantation Forestry in the Tropics: the Role, Silviculture, and Use of Planted Forests for Industrial, Social, Environmental, and Agroforestry Purposes*. Oxford University Press, Oxford, UK, 3rd edn. 9, 10, 46, 70, 71, 75, 76, 77, 78, 79, 81, 154, 307, 402
- Everard, D. (2000). *Biodiversity*, pp. 563–655. Vol. 2 of Owen (2000b). 27, 71, 402
- FAO (2006). *Management of wood properties in planted forests. A paradigm for global forest production*. Working paper FP/36/E, Forest Resources Development Services, Forest Resources Division, Forestry Department, FAO, Rome, Italy. 17, 18, 19, 403
- FAO (2007). FAOSTAT: ForesSTAT. Accessed <<http://faostat.fao.org/site/626/DesktopDefault.aspx?PageID=626#anchor>> on 10 April 2009. 227, 403
- Faulk, S.R. (1996). *Software requirements: a tutorial*, pp. 82–103. Wiley-IEEE Computer Society Press, USA. 35, 36, 37, 403
- Federal register (1982). National forest system land and resource management planning. 47(190):43026–43052. 25, 229, 403
- Field, R.C. (1984). National forest planning is promoting US Forest Service acceptance of Operations Research. *Interfaces*, 14(5):67–76. 25, 229, 237, 242, 249, 403
- Fins, L. (2008). Sustainable forestry: are planted forests the answer? <<http://www.cnr.uidaho.edu/for235/Pdf/Fall%202008/SustForsts-F08.pdf>> accessed on 29 March 2009. 9, 403
- Ford-Robertson, F.C., ed. (1971). *Terminology of Forest Science, Technology Practice and Products – English-language Version*, vol. 1 of *The Multilingual forestry terminology series*. Society of American Foresters, Washington DC, USA. xxiv, 403

- Forest Stewardship Council (2003a). About FSC. Accessed <<http://www.fsc.org/en/about>> on 7 February 2008. xxv, 86, 99, 403
- Forest Stewardship Council (2003b). Introduction to FSC's principles & criteria. Accessed <http://www.fsc.org/en/about/policy_standards/princ_criteria/1> on 29 August 2006. 86, 403
- France, R.B. and Docker, T.W.G. (1988). A formal basis for structured analysis. In: *Second IEE/BCS Conference on Software Engineering (Software Engineering 88), 11–15 July 1988*, pp. 191–195. IEEE. 36, 39, 403
- France, R.B. and Larrondo-Petrie, M.M. (1995). A two-dimensional view of integrated formal and informal specification techniques. In: J.P. Bowen and M.G. Hinchey, eds., *ZUM '95: The Z formal specification notation, Proceedings of the 9th International Conference of Z Users, Limerick, Ireland, 7–9 September 1995*, LNCS 967, pp. 434–448. Springer-Verlag Berlin Heidelberg, Germany. 39, 48, 217, 403
- Gane, C. and Sarson, T. (1979). *Structured Systems Analysis: Tools and Techniques*. Prentice-Hall Inc., Englewood Cliffs NJ, USA. 35, 36, 403
- Garcia, M.W. (1988). Using a forest management allocation and scheduling model for integrated resource planning. *Computers, Environment and Urban Systems*, 12(1):25–35. 25, 229, 233, 237, 242, 249, 403
- Garcia, O. (1984). FOLPI, a Forestry-Oriented Linear Programming Interpreter. In: H.N. et al., ed., *Proceedings of the IUFRO Symposium on Forest Management Planning and Managerial Economics*, pp. 293–305. 24, 25, 26, 230, 234, 238, 243, 250, 403
- Garcia, O. (1990). Linear programming and related approaches in forest planning. *New Zealand Journal of Forestry Science*, 20(3):307–331. 25, 26, 230, 234, 238, 243, 250, 403
- Gartner, B.L. (2005). Assessing wood characteristics and wood quality in intensively managed plantations. *Journal of Forestry*, 103(2):75–77. xvi, 17, 18, 403
- Gaudel, M.C. (1994). Formal specification techniques. In: *Proceedings of the 16th International Conference on Software Engineering (ICSE), 16–21 May 1994*, pp. 223–227. IEEE. 38, 213, 403
- George, V. and Vaughn, R. (2003). Application of lightweight formal methods in requirement engineering. *CrossTalk – The Journal of Defense Software Engineering*, 16(1):30. Accessed from <<http://www.stsc.hill.af.mil/crosstalk/2003/01/George.html>> on 27 January 2009. 39, 40, 213, 403

- Gordon, A.D., Wakelin, S.J. and Threadgill, J.A. (2006). Using measured and modelled wood quality information to optimise harvest scheduling and log allocation decisions. *New Zealand Journal of Forestry Science (Southern Journal of Biomaterials Research)*, 36(2–3):198–215. 9, 10, 16, 28, 29, 30, 44, 232, 236, 240, 248, 253, 403
- Goyal, G.C., Fisher, J.J., Krohn, M.J., Packood, R.E. and Olson, J.R. (1999). Variability in pulping and fiber characteristics of hybrid poplar trees due to their genetic makeup, environmental factors, and tree age. *Tappi Journal*, 82(5):141–147. 16, 403
- Gravell, A. (1991). What is a good formal specification? In: J.E. Nicholls, ed., *Z User Workshop: Proceedings of the Fifth Annual Z User Meeting, Oxford, 17–18 December 1990*, pp. 137–150. Springer-Verlag, London, UK. 39, 60, 403
- Greenspan, S.J., Mylopoulos, J. and Borgida, A. (1982). Capturing more world knowledge in the requirements specification. In: *Proceedings of the 6th International Conference on Software Engineering (ICSE '82), September 1982*, pp. 225–234. IEEE. 43, 224, 403
- Grzeskowiak, V. and Turner, P. (2000). *Anatomical characteristics of Eucalyptus and their impact on pulp strength properties*. Confidential Research Report ENV-D-C 2000-025, CSIR ENVIRONMENTEK, Durban, South Africa. 15, 403
- Guignard, M., Ryu, C. and Spielberg, K. (1998). Model tightening for integrated timber harvest and transportation planning. *European Journal of Operational Research*, 111(3):448–460. 22, 27, 28, 31, 231, 235, 239, 245, 251, 403
- Gunn, E.A. (1991). *Some aspects of hierarchical production planning in forest management*, pp. 54–62. In: Buford (1991). 24, 29, 30, 403
- Gunn, E.A. and Rai, A.K. (1987). Modelling and decomposition for planning long term forest harvesting in an integrated industry structure. *Canadian Journal of Forest Research*, 17(12):1507–1518. 25, 403
- Guttag, J.V. and Horning, J.J. (1993). *Larch: Languages and Tools for Formal Specification*. Springer-Verlag, New York NY, USA. 41, 403
- Hall, A. (1996). Using formal methods to develop an ATC information system. *IEEE Software*, 13(2):66–76. 38, 53, 403
- Hall, A. (1998). What does industry need from formal specification techniques? In: *Proceedings of the 2nd IEEE Workshop on Industrial Strength Formal Specification Techniques, 21–23 October 1998*, pp. 2–7. IEEE. 34, 215
- Hall, J.A. (1990). Seven myths of formal methods. *IEEE Software*, 7(5):11–19. 35, 38, 39, 40, 41, 42, 47, 61, 208, 210, 213, 216, 225, 403
- Harel, D. (1987). Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274. 41, 51, 403

- Hay, D. (2003). *Requirements Analysis: from Business View to Architecture*. Prentice-Hall PTR, Upper Saddle River NJ, USA. xvi, 37, 48, 49, 50, 51, 65, 207, 403
- Hay, D.C. (2004). Modelling business rules: what data models do. <<http://www.tdan.com/i027fe03.htm>> accessed on 2 March 2006. 52, 403
- Hayes, I., ed. (1987). *Specification Case Studies*. Prentice-Hall International (UK) Limited, UK. 54, 403
- Hayes, I., ed. (1993). *Specification Case Studies*. Prentice Hall International (UK) Limited, UK, 2nd edn. 54, 403, 422
- Heitmeyer, C.L. and McLean, J.D. (1983). Abstract requirements specification: a new approach and its application. *IEEE Transactions on Software Engineering*, SE-9(5):580–589. 35, 403
- Herbert, M. (2000). Site classification and forest management – prepared for University of Natal School of Agricultural Sciences and Agribusiness. Fractal Forest AFRICA, P.O. Box 1497, 3650 Hillcrest, South Africa. 10, 70, 154, 403
- Higgins, H.G. (1970). Technical assessment of eucalypt pulps in the papermaking economy. *Appita Journal*, 23(6):417–426. 16, 403
- Hinchey, M.G. (2002). Confessions of a formal methodist. In: P. Lindsay, ed., *Proceedings of the 7th Australian Workshop on Safety-Critical Systems and Software (SCS '02)*, vol. 15 of *Conferences in Research and Practice in Information Technology*, pp. 17–20. Australian Computer Society. 39, 213, 217, 403
- Hinchey, M.G. and Bowen, J.P., eds. (1999). *Industrial-strength Formal Methods in Practice*. Springer-Verlag London Limited, UK. 54, 403
- Hoare, C.A.R. (1985). *Communicating Sequential Processes*. Prentice-Hall International. 41, 403
- Holland, J., Sonksen, P., Carson, E. and Cohen, B. (1994). The directorate information system at St Thomas' hospital: a study in domain analysis. In: *Proceedings of the 1st International Conference on Requirements Engineering, 18–22 April 1994*, pp. 102–109. IEEE. 42, 403
- Hultqvist, D. and Olsson, L. (2004). *Demand-based tactical planning of the roundwood supply chain with respect to stochastic disturbances*. FSCN report R-03-44, Mid-Sweden University, Sundsvall, Sweden. 246, 251, 403
- Hultqvist, D. and Olsson, L. (2005). *A demand-based scenario optimization model for supply of raw material to the forest industry*, pp. 89–96. In: Bevers and Barrett (2005). 231, 235, 239, 246, 251, 403
- Hultqvist, D. and Olsson, L. (2006). Optimization of raw material procurement at pulp or paper – the influence of weather-related risks. *International Journal of Systems Science*, 37(4):253–269. 231, 235, 239, 246, 251, 403

- Hutter, D., Stephan, W., Traverso, P. and Ullmann, M., eds. (1999). *Applied Formal Methods – FM-Trends 98, International Workshop on Current Trends in Applied Formal Methods, Boppard, Germany, 7–9 October 1998*. LNCS 1641. Springer-Verlag, Berlin Heidelberg, Germany. 54, 403
- IEEE Standards Board (1994). *IEEE recommended practice for software requirements specifications*. Tech. Rep. IEEE Std 830-1993, IEEE Computer Society. 36, 39, 42, 136, 403
- Ilgun, K., Kemmerer, R.A. and Porras, P.A. (1995). State transition analysis: a rule-based intrusion detection approach. *IEEE Transactions on Software Engineering*, 21(3):181–199. 51, 403
- Ince, D. (1990). *Set piece*, pp. 370–372. In: Thayer and Dorfman (1990). 36, 39, 40, 41, 53, 54, 213, 403
- Ivkovich, M. (2000). *Breeding for improvement of tracheid characteristics in interior spruce*. D. Phil thesis, Department of Forest Science, Faculty of Forestry, University of British Columbia, Canada. 19, 403
- Jackson, M. (1983). *System Development*. Prentice-Hall International, Inc., London, UK. 36, 403
- Jackson, M. (1995). *Software Requirements and Specifications: a Lexicon of Practice, Principles and Prejudices*. Addison-Wesley, UK. 42, 43, 47, 403
- Jackson, M. (2001). *Problem Frames: Analyzing and Structuring Software Development Problems*. Addison-Wesley, Harlow, UK. 42, 47, 403
- Jackson, M. (2005). Problem frames and software engineering. *Information and Software Technology*, 47(14):903–912. 42, 403
- Jackson, M. and Megraw, R.A. (1986). Impact of juvenile wood on pulp and paper products. In: D. Robertson, ed., *Proceedings of a Technical Workshop: Juvenile Wood – What Does it Mean to Forest Management and Forest Products?*, pp. 75–81. 17, 18, 403
- Jackson, M. and Zave, P. (1993). Domain descriptions. In: *Proceedings of IEEE International Symposium on Requirements Engineering, San Diego CA, USA, 4–6 January 1993*, pp. 56–64. IEEE. 64, 207, 403
- Jacky, J. (1997). *The Way of Z: Practical Programming with Formal Methods*. Cambridge University Press, Cambridge, UK. 54, 59, 403
- Jia, X. (2002). *ZTC: A Type Checker for Z Notation – version 2.2*. School of Computer Science, Telecommunication, and Information Systems, DePaul University, Chicago IL, USA. 62, 403

- Johnson, K.N., Stuart, T. and Crimm, S.A. (1986). *FORPLAN, version 2: an overview*. USDA Forest Service, Land Management Planning Systems Section, Washington DC, USA. 25, 229, 233, 237, 242, 249, 403
- Johnson, K., Jones, D. and Kent, B. (1980). *Forest Planning Model (FORPLAN): users guide and operations manual (draft)*. USDA Forest Service, Land Management Planning. 25, 229, 233, 237, 242, 249, 403
- Jones, C.B. (1990). *Systematic Software Development Using VDM*. Prentice-Hall International, 2nd edn. 41, 403
- Jones, C.B. (1996). Formal methods light. *ACM Computing Surveys*, 28(4):121. 213, 403
- Jones, P.C. and Ohlmann, J.W. (2008). Long-range timber supply planning for a vertically integrated paper mill. *European Journal of Operational Research*, 191(2):558–571. 9, 403
- Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E. and Peterson, A.S. (1990). *Feature-oriented Domain Analysis (FODA) Feasibility Study*. Technical Report CMU/SEI-90-TR-21, Carnegie Mellon University, Software Engineering Institute, Pittsburgh PA, USA. 43, 224, 404
- Kanowski, P.J. (1997). Afforestation and plantation forestry. In: *Afforestation and Plantation Forestry for the 21st Century: Proceedings of the FAO XI World Forestry Congress, 13–22 October 1997*, vol. 3, pp. 23–33. FAO. 11, 12, 210, 404
- Kardasis, P. and Loucopoulos, P. (2004). Expressing and organising business rules. *Information and Software Technology*, 46(11):701–718. 52, 404
- Karlsson, H. (2006). *Fibre Guide – Fibre Analysis and Process Applications in the Pulp and Paper Industry*. AB Lorentzen & Wettre, Kista, Sweden. 17, 18, 404
- Karlsson, J., Rönnqvist, M. and Bergström, J. (2003). Short-term harvest planning including scheduling of harvest crews. *International Transactions of Operations Research*, 10(5):413–431. 10, 28, 29, 30, 232, 236, 240, 247, 252, 404
- Karlsson, J., Rönnqvist, M. and Bergström, J. (2004). An optimization model for annual harvest planning. *Canadian Journal of Forest Research*, 34(8):1747–1754. 28, 29, 30, 232, 236, 240, 248, 252, 404
- Kennedy, R.W. (1995). Coniferous wood quality in the future: concerns and strategies. *Wood Science and Technology*, 29(5):321–338. 18, 19, 404
- Kent, B., Bare, B.B., Field, R.C. and Bradley, G.A. (1991). Natural resource land management planning using large-scale linear programs: the USDA Forest Service experience with FORPLAN. *Operations Research*, 39(1):13–27. 23, 24, 25, 220, 229, 233, 237, 242, 249, 404

- Kirilenko, A.P. and Sedjo, R.A. (2007). Climate change impacts on forestry. *Proceedings of the National Academy of Sciences*, 104(50):19697–19702. 8, 404
- Kube, P.D. and Raymond, C.A. (2002). Prediction of whole tree basic density and pulp yield using wood core samples in *Eucalyptus nitens*. *Appita Journal*, 55(1):43–48. 18, 404
- Landsberg, J.J., Johnsen, K.H., Albaugh, T.J., Allen, H.L. and McKeand, S.E. (2001). Applying 3-PG, a simple process-based model designed to produce practical results, to data from loblolly pine experiments. *Forest Science*, 47(1):43–51. 105, 404
- Landsberg, J.J. and Waring, R.H. (1997). A generalised model of forest productivity using simplified concepts of radiation-use efficiency, carbon balance and partitioning. *Forest Ecology and Management*, 95(3):209–228. 105, 404
- Landsberg, J.J., Waring, R.H. and Coops, N.C. (2003). Performance of the forest productivity model 3-PG applied to a wide range of forest types. *Forest Ecology and Management*, 172(2–3):199–214. 105, 404
- Laroze, A. and Greber, B. (1991). *Multi-level harvest planning and log merchandising using goal-programming*, pp. 24–30. In: Buford (1991). 24, 28, 30, 31, 404
- Larson, P.R. (1967). Silvicultural control of the characteristics of wood used for furnish. In: *Proceedings of the 4th TAPPI Forest Biology Conference*, pp. 143–150. TAPPI. 15, 404
- Ledru, Y. (1996). Complementing semi-formal specifications with Z. In: *11th Knowledge-Based Software Engineering Conference (KBSE '96), 25–28 September 1996*, pp. 52–61. IEEE Computer Society. 38, 404
- Leuschner, W.A. (1990). *Forest Regulation, Harvest Scheduling and Planning Techniques*. John Wiley and Sons, New York, USA. xxix, 9, 10, 12, 46, 404
- Lightfoot, D. (2001). *Formal Specification Using Z*. Palgrave, Houndmills, Basingstoke, Hampshire, UK, 2nd edn. 54, 404
- Louw, W.J.A. (2000a). *Annual planning of operations*, pp. 199–204. Vol. 1 of Owen (2000b). 13, 404
- Louw, W.J.A. (2000b). *Subdivision and mapping of forestry land*, pp. 155–159. Vol. 1 of Owen (2000b). 9, 11, 12, 70, 77, 404
- Lundqvist, S.O. (2003). The EuroFiber project – objectives, layout and general results. EuroFiber Seminar, STFI, 6 May 2003. <http://www.stfipackforsk.se/upload/3439/d2-3_stfi-lundqvist1.pdf> accessed on 1 February 2007. 17, 20, 21, 44, 404

- Lundqvist, S.O., Grahn, T., Hedenberg, O. and Hansen, P. (2003). Visions and tools for forest and mill integration, Part 2. EuroFiber Seminar, STFI, 6 May 2003. <http://www.stfi-packforsk.se/upload/3439/02_d14-3_stfi-lundqvist2_Part%202x.pdf> accessed on 1 February 2007. 21, 23, 404
- Luqi and Goguen, J.A. (1997). Formal methods: promises and problems. *IEEE Software*, 14(1):73–85. 39, 42, 43, 213, 225, 404
- Malan, F.S. (2003). The wood quality of the South African timber resource for high-value solid wood products and its role in sustainable forestry. *Southern African Forestry Journal*, 198:53–62. 15, 404
- Mander, K.C. and Polack, F.A.C. (1995). Rigorous specification using structured systems analysis and Z. *Information and Software Technology*, 37(5–6):285–291. 39, 215, 217, 404
- Manley, B., Papps, S., Threadgill, J. and Wakelin, S. (1996). *Application of hierarchical forest planning in New Zealand*, pp. 107–116. In: Martell et al. (1996). 25, 26, 230, 234, 238, 243, 250, 404
- Manley, B.R. and Threadgill, J.A. (1991). LP used for valuation and planning of New Zealand plantation forests. *Interfaces*, 21(6):66–79. 25, 26, 230, 234, 238, 243, 250, 404
- Mansfield, S.D. and Weineisen, H. (2007). Wood fiber quality and kraft plping efficiencies of Trembling Aspen (*Populus tremuloides* Michx) clones. *Journal of Wood Chemistry and Technology*, 27(3–4):135–151. 16, 404
- Martell, D.L., Davis, L.S. and Weintraub, A., eds. (1996). *Proceedings of the Workshop on Hierarchical Approaches to Forest Management in Public and Private Organizations, 25–29 May 1992*. Information report PI-X-124, Petawawa National Forestry Institute, Chalk River ON, Canada. 404, 408, 412, 420, 429
- Martell, D.L., Gunn, E.A. and Weintraub, A. (1998). Forest management challenges for operational researchers. *European Journal of Operational Research*, 104(1):1–17. 12, 21, 22, 23, 24, 27, 29, 30, 51, 404
- Mathey, A.H., Krcmar, E., Dragicevic, S. and Vertinsky, I. (2008). An object-oriented cellular automata model for forest planning problems. *Ecological modelling*, 112(3–4):359–371. 22, 404
- McDill, M.E., Rebain, S.A. and Braze, J. (2002). Harvest scheduling with area-based adjacency constraints. *Forest Science*, 48(4):779–789. 27, 404
- McDonough, T.J. (1998). Kraft pulp yield basics. In: *Proceedings of the Breaking the Pulp Yield Barrier Symposium*, pp. 1–9. 17, 404

- McGuigan, B. and Scott, J. (1995). REGRAM-I – a flexible forest harvest scheduling and industrial processing, global optimization model. *New Zealand Journal of Forestry*, 40(2):17–20. 13, 22, 24, 25, 26, 230, 234, 238, 243, 250, 404
- McGuigan, B.N. (1984). A log resource allocation model to assist forest industry managers in process selection and location, wood allocation and transportation and production planning. *Appita Journal*, 37(4):289–296. 15, 16, 17, 22, 232, 236, 240, 247, 252, 404
- McMenamin, S.M. and Palmer, J.F. (1984). *Essential Systems Analysis*. Yourdon Press, New York, USA. 37, 404
- McNaughton, A. (1998). *Long-term scheduling of harvesting with adjacency and trigger constraints*. Ph.D thesis, Department of Engineering Science, School of Engineering, University of Auckland, New Zealand. 28, 404
- McNaughton, A., Rönnqvist, M. and Ryan, D. (1998). The scheduling of forest harvesting with adjacency constraints. In: *Proceedings of the 33rd Annual Conference of the Operational Research Society of New Zealand, 30 August–1 September 1998*, pp. 28–33. 230, 234, 238, 244, 250, 404
- McNaughton, A., Rönnqvist, M. and Ryan, D. (2000). A model which integrates strategic and tactical aspects of forest harvesting. In: M.J.D. Powell and S. Scholtes, eds., *19th IFIP TC7 Conference on System Modelling and Optimization: Methods, Theory and Applications, 12–16 July 1999*, pp. 189–208. Kluwer. 12, 23, 24, 27, 31, 32, 45, 226, 230, 234, 238, 244, 250, 404
- Megown, K.A., Turner, P. and Price, C.S. (2001). *Mill studies to evaluate the prototype Fibre Prediction System and recommendations for further development*. Confidential Research Report ENV-D-I 2001-07, CSIR ENVIRONMENTEK, Durban, South Africa. 20, 21, 44, 404
- Megown, R.A., Turner, P., Grzeskowiak, V. and Megown, K.A. (2000). The development of a GIS based wood quality prediction system for the pulp and paper industry. In: *Proceedings, TAPPSA Conference "African paper wood '2000 and beyond' "*, 17–20 October 2000, pp. 228–235. 20, 30, 44, 232, 236, 240, 247, 252, 404
- Meyer, B. (1985). On formalism in specifications. *IEEE Software*, 2(1):6–26. xix, 36, 38, 39, 40, 47, 56, 60, 208, 213, 216, 404
- Meynink, R. and Borough, C. (2005). Meeting fluctuating demand for different wood quality and specifications. In: *Institute of Foresters of Australia Conference, 2005*. Institute of Foresters of Australia Conference, <http://www.forestry.org.au/ifa/i/download-2005-conference-presentations.asp?filename=ChrisBorough.pdf> accessed on 10 April 2006. 30, 44, 232, 236, 240, 248, 253, 404

- Mikkonen, E. (1996). Quality-based wood procurement planning end-use of abstract wood. In: C.R. Blinn and M.A. Thompson, eds., *Proceedings of Papers Presented at the Joint Meeting of the Council on Forest Engineering and International Organization of Forest Research Organizations, "Planning and Implementing Forest Operations to Achieve Sustainable Forests"*, pp. 256–259. 19, 44, 404
- Milner, A. (1980). *A Calculus of Communicating Systems*. LNCS 92. Springer-Verlag, Berlin Heidelberg, Germany. 41, 404
- Miranda, I. and Pereira, H. (2002). Variation of pulpwood quality with provenances and site in *Eucalyptus globulus*. *Annals of Forest Science*, 59(3):283–291. 17, 18, 404
- Mitchell, S.A. (2004). *Operational forest harvest scheduling optimisation: a mathematical model and solution strategy*. D. Phil thesis, Department of Engineering Science, School of Engineering, University of Auckland, New Zealand. 22, 28, 29, 30, 232, 236, 240, 248, 253, 404
- Morales, R., Jofré, A. and Chevalier, P. (1994). Simultaneous planning of industrial structure and forest plantation management in wood based companies. In: J. Sessions and J.D. Brodie, eds., *Proceedings of the 1994 Symposium on Systems Analysis and Forest Resources*, pp. 44–50. Society of American Foresters, Bethesda MD, USA. 24, 404
- Morales, R. and Weintraub, A. (1991). *Model for strategic planning in the management of pine industrial plantations*, pp. 114–117. In: Buford (1991). 25, 229, 233, 237, 242, 249, 404
- Morgan, C.C. and Sufrin, B.A. (1993). *Specification of the Unix filing system*, pp. 45–78. In: Hayes (1993), 2nd edn. 36, 42, 61, 151, 404
- Morgan, T. (2002). *Business Rules and Information Systems: Aligning IT with Business Goals*. Addison-Wesley, Boston MA, USA. 49, 52, 404
- MSU (2004). ZTC-2.0 – Provers for “Formal methods in software development (CSE814)” course, Department of Computer Science and Engineering, Michigan State University, USA. <<http://www.cse.msu.edu/~cse814/Public/provers/Zed/ZTC-2.0/>> accessed on 23 July 2007. 62, 404
- Muneri, A. (1994). *Pulpwood testing and evaluation: a case study for a proposed kraft pulp mill and associated plantations in Zimbabwe*. D. Phil thesis, School of Forestry and Resource Conservation, University of Melbourne, Australia. 17, 18, 404
- Murray, A.T. and Church, R.L. (1995). Heuristic solution approaches to operational forest planning problems. *OR Spektrum*, 17(2–3):193–203. 22, 23, 27, 29, 404
- Mylopoulos, J. (1992). *Conceptual modelling and Telos*, pp. 49–68. John Wiley and Sons, New York, USA. 37, 404

- Naidu, R.D. (2003). *The impact of physical characteristics of Pinus patula on kraft pulp properties*. Master's thesis, Department of Life and Environmental Sciences, University of Natal, Durban, South Africa. xxiv, xxv, 14, 19, 404
- Nash, T.C. (1990). Using Z to describe large systems. In: J.E. Nicholls, ed., *Z User Workshop: Proceedings of the Fourth Annual Z User Meeting, 15 December 1989*, pp. 150–178. Springer-Verlag, London, UK. 210, 404
- Navon, D.I. (1971). *Timber RAM ... a long term planning method for commercial timber lands under multiple use management*. Research Paper PSW-70, USDA Forest Service, Berkley CA, USA. 25, 229, 233, 237, 242, 249, 404
- Nelson, J., Brodie, J.D. and Sessions, J. (1991). Integrating short-term area-based logging plans with long-term harvesting schedules. *Forest Science*, 37(1):101–122. 12, 22, 23, 25, 27, 31, 45, 404
- Nightingale, J.M., Hill, M.J., Phinn, S.R., Davies, I.D., Held, A.A. and Erskine, P.D. (2008). Use of 3-PG and 3-PGS to simulate forest growth dynamics of Australian tropical rainforests: I. Parameterisation and calibration for old-growth, regenerating and plantation forests. *Forest Ecology and Management*, 254(2):107–121. 105, 404
- Nix, C.J. and Collins, B.P. (1988). The use of software engineering, including the Z notation, in the development of CICS. *Quality Assurance*, 14(3):103–110. 38, 41, 42, 53, 54, 404
- Nobre, S.R. and Rodriguez, L.C.E. (2005). *A relational data model for the generation of large-scale forest scheduling problems*, pp. 299–306. In: Bevers and Barrett (2005). 44, 404
- Öhman, K. and Eriksson, L.O. (2002). Allowing for spatial consideration in long-term forest planning by linking linear programming with simulated annealing. *Forest Ecology and Management*, 161(1–3):221–230. 31, 45, 405
- Owen, D.L., ed. (2000a). *Glossary*, pp. 724–734. Vol. 2 of Owen (2000b). xxiv, 405
- Owen, D.L., ed. (2000b). *South African Forestry Handbook*. Southern African Institute of Forestry, Postnet Suite 329, Private Bag X4, 0102 Menlo Park, Pretoria, South Africa. 405, 410, 414, 420, 423, 430
- Paavilainen, L. (1994). Bonding potential of softwood sulphate pulp fibres. *Paperi ja Puu*, 76(3):162–173. 19, 405
- Papps, S.R. and Manley, B.R. (1992). Integrating short-term planning with long-term forest estate modeling using FOLPI. In: *Proceedings of the IUFRO conference on "Integrating Forest Information over Space and Time"*, pp. 188–198. IUFRO. 25, 26, 27, 230, 234, 238, 243, 250, 405

- Parnas, D.L. (1969). On the use of transition diagrams in the design of a user interface for an interactive computer system. In: *Proceedings of the 24th ACM Conference, 26–28 August 1969*, pp. 379–385. ACM. 51, 405
- Pearlson, K.E. and Saunders, C.S. (2006). *Managing and Using Information Systems: a Strategic Approach*. John Wiley and Sons Inc., Hoboken NJ, USA, 3rd edn. 217, 227, 405
- Philpott, A. and Everett, G. (2001). Supply chain optimisation in the paper industry. *Annals of Operations Research*, 108(1–4):225–237. 9, 405
- Plat, N., van Katwijk, J. and Pronk, K. (1991). A case for structured analysis/formal design. In: S. Prehn and W.J. Toetenel, eds., *VDM '91: Formal Software Development Methods, Noordwijkerhout, The Netherlands, 21–25 October 1991*, LNCS 551, pp. 81–105. Springer-Verlag, Berlin Heidelberg, Germany. 37, 39, 405
- Pohl, K. (1993). The three dimensions of requirements engineering. In: C. Rolland, F. Bodart and C. Cauvet, eds., *Advanced Information Systems Engineering: Proceedings of the Fifth International Conference (CAiSE '93), Paris, France, 8–11 June 1993*, LNCS 685, pp. 275–292. Springer-Verlag, Berlin, Germany. 36, 405
- Polack, F. (2001). *SAZ: SSADM version 4 and Z*, pp. 21–38. Springer-Verlag London Limited, London, UK. 36, 405
- Polack, F., Whiston, M. and Mander, K. (1993). The SAZ project: integrating SSADM and Z. In: J. Woodcock and P. Larsen, eds., *FME '93: Industrial Strength Formal Methods, Proceedings of the First International Symposium of Formal Methods Europe, Odense, Denmark, 19–23 April 1993*, LNCS 670. Springer-Verlag, Berlin Heidelberg, Germany. 39, 217, 405
- Price, C.S., Blakeway, F., Ahmed, F. and Tapamo, J.R. (2008a). Plantation forest harvest scheduling systems which include wood properties applicable to pulp mills: a review. In preparation; submitted to *South African Journal of Science*. vi, 8, 405
- Price, C.S., Tapamo, J.R., Blakeway, F. and Ahmed, F. (2008b). Plantation forestry: an analysis of the domain. *In preparation*. vii, 405
- Price, C.S., Tapamo, J.R., Blakeway, F. and Ahmed, F. (2009). Plantation forestry: an analysis of the domain. *Accepted for Domain Engineering Workshop to be held at the 21st International Conference on Advanced Information Systems Engineering (CAiSE 09)*. vii, 65, 152, 405
- Pulkki, R. (2001). Role of supply chain management in the wise use of wood resources. *Southern African Forestry Journal*, 191:89–95. 10, 405
- Raven, P.H., Evert, R.F. and Eichorn, S.E. (1992). *Biology of Plants*. Worth Publishers, New York, USA. 14, 405

- Retief, R.J., Male, J.R., Malan, F.S., Dyer, S.T., Conradie, D. and Turner, P. (1997). *First report on the effect of site quality on wood properties of Eucalyptus grandis clonal material*. Confidential Research Report ENV-P-I 97045, CSIR ENVIRONMENTEK, Pretoria, South Africa. 15, 405
- Ribeiro, R.P., Borges, J.G., Pereira, C.M., Sousa, P.M. and Lé, J.P. (2005). *Designing an integrated forest planning system for the forest industry: an application in Portugal*, pp. 89–96. In: Bevers and Barrett (2005). 12, 44, 208, 218, 219, 405
- Robertson, S. and Robertson, J. (2006). *Mastering the Requirements Process*. Addison-Wesley, Upper Saddle River NJ, USA, 2nd edn. 42, 405
- Robinson, N.J. (2002). Checking Z data refinements using an animation tool. In: D. Bert, J.P. Bowen, M.C. Henson and K. Robinson, eds., *ZB 2002: Formal Specification and Development in Z and B, 2nd International Conference of B and Z Users, Grenoble, France, 23–25 January 2002*, LNCS 2272, pp. 111–120. Springer-Verlag, Berlin Heidelberg, Germany. 40, 405
- Rönqvist, M. (2003). Optimization in forestry. *Mathematical programming: a publication of the Mathematical Programming Society, Series B*, 97(1–2):267–284. xvi, 11, 27, 31, 405
- Rosca, D., Greenspan, S., Febowitz, M. and Wild, C. (1997). A decision making methodology in support of the business rules lifecycle. In: *Proceedings of the Third IEEE International Symposium on Requirements Engineering (RE '97), 6–10 January 1997*, pp. 236–246. IEEE. 52, 405
- Ross, D.T. (1977). Structured Analysis (SA): a language for communicating ideas. *IEEE Transactions on Software Engineering*, SE-3(1):16–34. 36, 405
- Royce, W.W. (1970). Managing the development of large software systems. In: *IEEE WESTCON'70, August 1970*, pp. 1–9. TRW. 35, 405
- Saaltink, M. (1997). The Z/EVES system. In: J.P. Bowen, M.G. Hinchey and D. Till, eds., *ZUM '97: The Z Formal Specification Notation, Proceedings of the 10th International Conference of Z Users, Reading, UK, 3–4 April 1997*, LNCS 1212, pp. 81–105. Springer-Verlag, Berlin Heidelberg, Germany. 40, 405
- Sands, P.J. and Landsberg, J.J. (2002). Parameterisation of 3-PG for plantation grown *Eucalyptus globulus*. *Forest Ecology and Management*, 163(1–3):273–292. 105, 405
- Savill, P., Evans, J., Auclair, D. and Falck, J. (1997). *Plantation Silviculture in Europe*. Oxford University Press, Oxford, UK. xxxi, 46, 70, 71, 81, 405
- Sedjo, R.A. (1999). The potential of high-yield plantation forestry for meeting timber needs. *New Forests*, 17(1–3):339–360. 8, 405

- Sefara, N.L., Jacobs, S.M. and Turner, P. (2001). *The influence of site and age on the rate of delignification of Eucalyptus grandis seedling material*. Confidential Research Report ENV-D-C 2000-019, CSIR ENVIRONMENTEK, Durban, South Africa. 15, 16, 405
- Semmens, L. and Allen, P. (1991). Using Yourdon and Z: an approach to formal specification. In: J.E. Nicholls, ed., *Z User Workshop: Proceedings of the Fifth Annual Z User Meeting, Oxford, 17–18 December 1990*, pp. 228–253. Springer-Verlag, Berlin Heidelberg New York. 37, 39, 217, 405
- Semmens, L.T., France, R.B. and Docker, T.W.G. (1992). Integrated structured analysis and formal specification techniques. *The Computer Journal*, 35(6):600–610. 39, 217, 405
- Shepherd, K.R. (1986). *Plantation Silviculture*. Martinus Nijhof Publishers, Dordrecht, The Netherlands. 11, 46, 74, 78, 81, 84, 405
- Siau, K. (2004). Informational and computational equivalence in comparing information modeling methods. *Journal of Database Management*, 15(1):73–86. 43, 405
- Smith, C.W. and Cunningham, L. (2002). The effect of stand density on the growth of *Eucalyptus grandis* hybrids at 5 to 7 years of age. Institute for Commercial Forestry Research Bulletin series 03/2002, Pietermaritzburg, South Africa. 10, 405
- Smith, S. (1978). *A two-phase method for timber supply analysis*, pp. 76–81. General Technical Report PSW-GTR-32, Pacific Southwest Forest and Range Experiment Station, USDA Forest Service, Berkley CA, USA. 31, 45, 405
- Sommerville, I. (1996). *Software Engineering*. Addison-Wesley, Wokingham, UK, 5th edn. 41, 405
- Sommerville, I. (2007). *Software Engineering*. Addison-Wesley, Harlow, UK, 8th edn. 34, 51, 55, 57, 405
- Sommerville, I. and Sawyer, P. (1997a). *Requirements Engineering: a Good Practice Guide*. John Wiley and Sons Ltd., Chichester, West Sussex, UK. 49, 405
- Sommerville, I. and Sawyer, P. (1997b). Viewpoints: principles, problems and a practical approach to requirements engineering. *Annals of Software Engineering*, 3(0):101–130. 42, 405
- Spångberg, K. (1999). Classification of *Picea abies* pulpwood according to wood and stand properties. *Scandinavian Journal of Forestry Research*, 14(3):276–281. 15, 16, 17, 18, 21, 44, 405
- Spivey, J.M. (1998). *The Z Notation: a Reference Manual*. J. M. Spivey, Oriel College, Oxford, UK, 2nd edn. xxxiv, 41, 54, 59, 405

- Stadtler, H. (2005). Supply chain management and advanced planning – basics, overview and challenges. *European Journal of Operational Research*, 163(3):575–588. 10, 405
- Stepney, S., Polack, F. and Toyn, I. (2003). An outline pattern language for Z: five illustrations and two tables. In: D. Bert, J.P. Bowen, S. King and M. Waldén, eds., *ZB 2003: Formal Specification and Development in Z and B, Third International Conference of B and Z Users, Turku, Finland, 4–6 June 2003*, LNCS 2651. Springer-Verlag, Berlin Heidelberg, Germany. 58, 274, 290, 405
- Sutcliffe, A. and Maiden, N. (1998). The domain theory for requirements engineering. *IEEE Transactions on Software Engineering*, 24(3):174–196. 43, 224, 405
- Sykes, J.B., ed. (1982). *The Concise Oxford Dictionary of Current English*. Clarendon, Oxford, UK, 6th edn. xxiv, xxv, xxvi, xxvii, xxviii, xxx, xxxi, 405
- Syndicate Database Solutions (2006). Microforest Harvest Scheduling System (HSS). <<http://www.syndicate.co.za/files/hss.html>> accessed on 7 September 2007. 26, 230, 234, 238, 244, 250, 405
- Tedder, P.L., Weaver, G.H. and Kletke, D.D. (1978). Timber RAM and ECHO: alternative harvest planning models for a southern forest. *Southern Journal of Applied Forestry*, 2(3):98–103. 25, 229, 233, 237, 242, 249, 405
- Thayer, R.H. and Dorfman, M., eds. (1990). *System and Software Requirements Engineering*. IEEE Computer Society Press, Los Alamitos CA, USA. 405, 413, 417
- The RAISE Language Group (1992). *The RAISE Specification Language*. BCS Practitioner Series. Prentice Hall International (UK) Ltd, UK. 41, 405
- Tretmans, J., Wijbrans, K. and Chaudron, M. (2001). Software engineering with formal methods: the development of a storm surge barrier control system – revisiting seven myths of formal methods. *Formal Methods in System Design*, 19(2):195–215. 38, 40, 41, 42, 54, 60, 210, 405
- Turner, B.J., Chikumbo, O. and Davey, S.M. (2002). Optimisation modelling of sustainable forest management at the regional level: an Australian example. *Ecological Modelling*, 153(1–2):157–179. 23, 405
- Turner, P., Drew, D.M., Wesley-Smith, J. and Zbonak, A. (2005). *The impact of variation of wood properties in the profitability of a pulp mill*. Confidential Research Report ENV-D-C 2005-005, CSIR ENVIRONMENTEK, Durban, South Africa. 17, 18, 405
- Turner, P., Megown, K.A., Grzeskowiak, V., Male, J.R., Retief, R.J., Sefara, N.L., Conradie, D., Crafford, J.G. and Havenga, A. (1999). *A report on research activities of the Mondi-Sappi-CSIR Wood Quality Research Consortium for the 1998–1999 financial year*. Confidential Research Report ENV-P-I 98165, CSIR ENVIRONMENTEK, Pretoria, South Africa. 15, 405

- Turner, P., Megown, K.A., Grzeskowiak, V., Megown, R.A., Pauck, J. and Alborough, D. (2000). An industrial evaluation of a stand level grading system for *Pinus patula*. In: *Proceedings of the IUFRO Working Party 2.08.01 Forest Genetics for the Next Millennium, 8–13 October 2000*, pp. 228–235. 2, 15, 16, 17, 19, 20, 21, 405
- Turner, P. and Price, C.S. (1996). Maximising value of the plantation resource: Part 1 – development of a log grading proposal for pruned softwood logs. *South African Forestry Journal*, 176:29–34. 29, 405
- Turner, P. and Price, C.S. (2005). Timber resource management. PCT Patent Application No. PCT/IB 2005/001707. 6, 129, 405
- Turner, P. and Retief, R.J. (1998). *Wood quality studies to evaluate the impact of site quality, genotype and management regime on the basic wood, pulp and pulping properties of different eucalypt clones*. Confidential Research Report ENV-P-I 98117, CSIR ENVIRONMENTEK, Pretoria, South Africa. 15, 405
- USDA Forest Service (1996). Spectrum overview, USDA Forest Service, Inventory and Monitoring Institute, Fort Collins CO, USA. <<http://www.fs.fed.us/institute/PAG/SPECTRUM.shtml>> accessed on 29 August 2006. 24, 405
- Utting, M. (1995). Animating Z: Interactivity, transparency and equivalence. In: *Asia-Pacific Software Engineering Conference (APSEC '95), Brisbane, Australia, 6–9 December 1995*, pp. 294–303. IEEE Computer Society, Washington DC, USA. 40, 405
- Valentine, S.H. (1995). Equal rights for schemas in Z. In: J.P. Bowen and M.G. Hinchey, eds., *ZUM '95: The Z Formal Specification Notation, Proceedings of the 9th International Conference of Z Users, Limerick, Ireland, 7–9 September 1995*, LNCS 967, pp. 183–202. Springer-Verlag Berlin Heidelberg, Germany. 54, 59, 405
- van Eijk, P.H.J., Vissers, C.A. and Diaz, M., eds. (1989). *The Formal Description Technique LOTOS*. North-Holland. 41, 405
- Vielma, J.P., Murray, A.T., Ryan, D.M. and Weintraub, A. (2007). Improving computational capabilities for addressing volume constraints in forest harvest scheduling problems. *European Journal of Operational Research*, 176(2):1246–1264. 27, 406
- von Gadow, K. and Bredenkamp, B. (1992). *Forest Management*. Academica, Pretoria, South Africa. xxx, 12, 46, 85, 103, 106, 406
- von Halle, B. (2002). *Business Rules Applied: Building Better Systems using the Business Rules Approach*. John Wiley and Sons, New York, USA. 52, 406
- Walker, J.L. (1974). ECHO – an economic harvest optimization model. In: *Proceedings of the 65th Western Forestry Conference*. 25, 229, 233, 237, 242, 249, 406

- Wallis, A.F.A., Wearne, R.H. and Wright, P.J. (1996). Analytical characteristics of plantation woods relating to kraft pulp yields. *Appita Journal*, 49(6):427–432. 18, 406
- Wasserman, A.I. (1985). Extending state transition diagrams for the specification of human-computer interaction. *IEEE Transactions on Software Engineering*, SE-11(8):699–713. 51, 406
- Weaver, P.L., Lambrou, N. and Walkley, M. (1998). *Practical SSADM Version 4+: a Complete Tutorial Guide*. Pearson Education Limited, Harlow, UK, 2nd edn. 42, 406
- Weigel, G. (2005). *A strategic planning model for maximizing value in pulp and paper mills*. Master's thesis, Mechanical Engineering, Université Laval, Québec, Canada. 14, 20, 21, 44, 406
- Weintraub, A. (2007). Integer programming in forestry. *Annals of Operations Research*, 149(1):209–216. 23, 406
- Weintraub, A. and Bare, B.B. (1996). New issues in forest land management from an Operations Research perspective. *Interfaces*, 26(5):9–25. 10, 23, 27, 51, 406
- Weintraub, A. and Cholaky, A. (1991). A hierarchical approach to forest planning. *Forest Science*, 37(2):439–460. 12, 23, 24, 25, 26, 31, 32, 45, 406
- Weintraub, A., Church, R.L., Murray, A.T. and Guignard, M. (2000). Forest management models and combinatorial algorithms: analysis of state of the art. *Annals of Operations Research*, 96(1–4):271–285. 22, 28, 29, 229, 231, 232, 233, 235, 236, 237, 239, 240, 242, 243, 245, 247, 249, 250, 251, 252, 406
- Weintraub, A. and Davis, L. (1996). *Hierarchical planning in forest resource management: defining the dimensions of the subject area*, pp. 2–14. In: Martell et al. (1996). 10, 21, 22, 31, 70, 406
- Weintraub, A., Jones, G., Magendzo, A., Meacham, M. and Kirby, M. (1994). A heuristic system to solve mixed integer forest planning models. *Operations Research*, 42(6):1010–1024. 22, 28, 406
- Weintraub, A., Jones, J.P., Meacham, M., Magendzo, A., Magendzo, A. and Malchuck, D. (1995). Heuristic procedures for solving mixed-integer harvest scheduling-transportation planning models. *Canadian Journal of Forest Research*, 25(10):1618–1626. 22, 27, 28, 406
- Weintraub, A. and Murray, A.T. (2006). Review of combinatorial problems induced by spatial forest harvesting planning. *Discrete Applied Mathematics*, 154(5):867–879. 12, 22, 24, 27, 406
- Wessels, C.B. and Price, C.S. (1999). *Sawmill production planning system (SPPS) Manual – Version 1.1*. Confidential Research Report ENV-P-I 99002, CSIR ENVIRONMENTEK, Pretoria, South Africa. 218, 406

- Wessels, C.B., Price, C.S., Turner, P. and Dell, M.P. (2006). Integrating harvesting and sawmill operations using an optimized sawmill production planning system. In: *Proceedings of the IUFRO Precision Forestry Symposium, Division 3: Forest Operation, 5–10 March 2006*, pp. 341–347. IUFRO. 218, 406
- White, S.A. (2004). Introduction to BPMN. <<http://www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf>> accessed on 6 February 2008. 51, 406
- Wieringa, R. (1998). A survey of structured and object-oriented software specification methods and techniques. *ACM Computing Surveys*, 30(4):459–527. 37, 50, 51, 52, 406
- Williams, M.F. (1994). Matching wood fiber characteristics to pulp and paper processes and products. *Tappi Journal*, 77(3):227–233. 14, 21, 406
- Wilson, K. and White, D.J.B. (1986). *The Anatomy of Wood: its Diversity and Variability*. Stobart and Son Ltd., London, UK. 14, 15, 406
- Wimmer, R., Downes, G.M., Evans, R., Rasmussen, G. and French, J. (2002). Direct effects of wood characteristics on pulp and handsheet properties of *Eucalyptus globulus*. *Holzforschung*, 56(3):244–252. 18, 406
- Wing, J.M. (1990). A specifier's introduction to formal methods. *IEEE Computer*, 23(4):8 & 10–22 & 24. 37, 38, 42, 61, 210, 406
- Winters, R.K., ed. (1977). *Terminology of Forest Science, Technology Practice and Products – English-language Version, Addendum Number One*. The Multilingual forestry terminology series. Society of American Foresters, Washington DC, USA. xxiv, 406
- Woodcock, J. and Davies, J. (1996). *Using Z – Specification, Refinement, and Proof*. Prentice-Hall, Inc., Upper Saddle River NJ, USA. xxxiv, 54, 59, 406
- Woodcock, J.C.P. (1989). Structuring specifications in Z. *Software Engineering Journal*, 4(1):51–65. 54, 59, 406
- Wordsworth, J.B. (1992). *Software Development with Z*. Addison-Wesley Publishers Ltd., Wokingham, UK. 54, 59, 406
- Xu, L., Filonenko, Y. and Parker, I. (1997). Measurement of wall thickness of fully collapsed fibres by confocal microscopy and image analysis. *Appita Journal*, 50(6):501–504. 19, 406
- Yeh, R.T. and Zave, P. (1980). Specifying software requirements. *Proceedings of the IEEE*, 68(9):1077–1085. 35, 36, 406
- Yoshimoto, A., Brodie, J.D. and Sessions, J. (1994). A new heuristic to solve spatially constrained long-term harvest scheduling problems. *Forest Science*, 40(3):365–396. 23, 31, 406

- Yourdon, E. (1989). *Modern Structured Analysis*. Prentice-Hall, Inc., Englewood Cliffs NJ, USA. 36, 42, 406
- Z Archive (1996). Z archive. <<ftp://ftp.comlab.ox.ac.uk/pub/Zforum/>>. 54, 406
- Zachman, J.A. (1987). A framework for information systems architecture. *IBM Systems Journal*, 26(3):276–292. 37, 44, 45, 48, 207, 406
- Zave, P. (1984). The operational versus the conventional approach to software development. *Communications of the ACM*, 27(2):104–118. 34, 35, 406
- Zave, P. and Jackson, M. (1996). Where do operations come from? A multiparadigm specification technique. *IEEE Transactions on Software Engineering*, 22(7):508–528. 54, 406
- Zbonak, A. (2005). *The effect of elevation, site index and age on wood density and anatomical characteristics of Pinus patula*. Confidential Research Report ENV-D-C 2005-049, CSIR ENVIRONMENTEK, Durban, South Africa. 15, 406
- Zobel, B.J., Jr., E.C. and Ikemori, Y.K. (1983). Selecting and breeding for desirable wood. *Tappi Journal*, 66(1):70–74. 15, 16, 406
- Zobel, B.J. and van Buijtenen, J.P. (1989). *Wood Variation: its Causes and Control*. Springer, Berlin, Germany. 15, 16, 17, 406
- Zwolinski, J.B. and Hinze, W.H.F. (2000). *Pines*, pp. 116–120. Vol. 1 of Owen (2000b). xxxi, 75, 406

Index of schema names found in Appendices D and E

AcceptLogsFromOtherSuppliers, 326
ActualRegimes, 284
AgeOfTreesInStand, 306
AnnualEstateLevelCosts, 381
AnnualLogMassForMill, 356
CalculateAgeOfTreesInStand, 307
CalculateAnnualEstateLevelCosts, 381
CalculateAnnualLogMassForMill, 189, 356
CalculateAnnualMillLogPurchases, 193, 376
CalculateDeliveredLogMillCosts, 378
CalculateLHaulTransportStandOptionCosts, 371
CalculateObjectiveFunction, 194, 384
CalculatePossibleSolutionStandOptionCosts, 380
CalculatePossibleSolutions, 354
CalculateRegimeStandOptionCosts, 366
CalculateSHaulTransportStandOptionCosts, 369
CalculateStandOptionCosts, 374
CalculateTotalNonStandForestryCosts, 383
CalculateTransportStandOptionCosts, 372
CompanyHasEstates, 294
ConvertStandVolumeToMass, 309
DateDefinition, 277
DefineAgeOfTreesInStand, 306
DefineAllSolutionsInfeasible, 360
DefineAnnualEstateLevelCosts, 381
DefineAnnualLogMassForMill, 355
DefineConvertStandVolumeToMass, 172, 309
DefineDeliveredLogMillCosts, 377
DefineDepot, 289
DefineEstimateStandsVolume, 171, 307
DefineFeasibleSolution, 358
DefineFeasibleSolutionsExist, 360
DefineFirstRegimeActionInStandOption, 361
DefineFirstRotation, 363
DefineGenMatID, 364
DefineLHaulTransportStandOptionCosts, 370
DefineLogsAtRoadside, 288
DefineMill, 292
DefineMillLogPurchases, 375
DefineNumberOfStandOptions, 341
DefineObjectiveFunction, 383
DefineOptimalSolution, 386
DefinePossibleSolution, 353
DefinePossibleSolutionStandOptionCosts, 379
DefineRegimeIDs, 282
DefineRegimeStandOptionCosts, 364
DefineSHaulTransportStandOptionCosts, 368
DefineStandHarvestingInfo, 343
DefineStandOptionCosts, 373
DefineStandPlantingStatus, 298
DefineStandRegimeAction, 342
DefineStandVolumeAndMass, 300
DefineTotalNonStandForestryCosts, 382
DefineTrackActualRegimes, 281
DefineTransportStandOptionCosts, 371
DefineTripIDs, 314
DefineYearVariables, 363
DeliveredLogMillCosts, 378
Depot, 290
DetermineFeasibleSolutions, 191, 359
DetermineFirstRegimeActionInStandOption, 362
DetermineIfFeasibleSolutionsExist, 360
DetermineOptimalSolution, 196, 387
DeterminePlantingRegime, 303
EstateBelongsToCompany, 294
EstateHasStands, 294
EstateLevelCosts, 336
EstimateStandWoodProp, 179, 333
EstimateStandsVolume, 308
ExceptionHarvestStandAndUpdateRegime, 313
ExceptionPlantStandAndRecordRegime, 305
ExceptionProcessLogs, 327
Exponent, 340
FHSS, 199, 390
FHSSPart1, 197, 389
FHSSPart2, 198, 389
FeasibleSolution, 358
FeasibleSolutionsExist, 360
FirstRegimeActionInStandOption, 362
FirstRotation, 364
ForestToMillSupplyChain, 328
ForestryCompanyLevelCosts, 335
GenMatID, 364
GenerateStandRegimeActionForPlantedStands, 348

GenerateStandRegimeActionForStands, 349
GenerateStandRegimeActionForUnplantedStands, 290
 346
GeneticMaterialNames, 272
Genus, 273
HarvestStand, 313
HarvestStandAndUpdateRegime, 313
HarvestStandAndUpdateRegimeOK, 312
Hybrid, 275
InitAgeOfTreesInStand, 306
InitAnnualEstateLevelCosts, 381
InitAnnualLogMassForMill, 355
InitConvertStandVolumeToMass, 309
InitDeliveredLogMillCosts, 377
InitDepot, 290
InitEstimateStandsVolume, 308
InitFeasibleSolution, 358
InitFeasibleSolutionsExist, 360
InitFirstRegimeActionInStandOption, 362
InitFirstRotation, 364
InitGenMatID, 364
InitLHaulTransportStandOptionCosts, 370
InitLogsAtRoadside, 288
InitMill, 292
InitMillLogPurchases, 375
InitNumberOfStandOptions, 341
InitObjectiveFunction, 384
InitOptimalSolution, 386
InitPossibleSolution, 353
InitPossibleSolutionStandOptionCosts, 379
InitRegimeIDs, 282
InitRegimeStandOptionCosts, 365
InitSHaulTransportStandOptionCosts, 368
InitStandHarvestingInfo, 344
InitStandOptionCosts, 373
InitStandPlantingStatus, 298
InitStandRegimeAction, 343
InitStandVolumeAndMass, 300
InitTotalNonStandForestryCosts, 382
InitTrackActualRegimes, 282
InitTransportStandOptionCosts, 371
InitYearVariables, 363
LHaulTransportStandOptionCosts, 370
LoadLogsAtDepot, 323
LoadLogsAtStand, 319
LogisticsChain, 173, 286
LogsAtRoadside, 289
LongHaulRoad, 316
LongHaulTransport, 325
LongHaulTrip, 322
Mill, 292
MillAcceptableSpecies, 291
MillForStandsLogs, 287
MillLogPurchases, 376
MillRequirements, 180, 334
NoLogsAtDepot, 290
NoLogsAtMill, 292
NoLogsAtRoadside, 289
NumberOfStandOptions, 342
ObjectiveFunction, 384
OptimalSolution, 386
PlannedRegimes, 283
PlanningHorizonDefinition, 178, 332
PlantStand, 305
PlantStandAndRecordRegime, 305
PlantStandAndRecordRegimeOK, 304
PlantationActions, 313
PossibleSolution, 353
PossibleSolutionStandOptionCosts, 379
ProcessLogs, 327
PureSpecies, 274
RegimeCanopyClosure, 182, 336
RegimeCosts, 183, 337
RegimeFellAge, 280
RegimeFellDate, 281
RegimeFelling, 281
RegimeFunctions, 282
RegimeGeneticMaterial, 278
RegimeIDs, 282
RegimePlantAge, 279
RegimePlantDate, 279
RegimePlanting, 280
RegimeStandOptionCosts, 365
RemoveLogsForProcessing, 327
Road, 316
SHaulTransportStandOptionCosts, 368
SetAnnualEstateLevelCostsToZero, 381
SetAnnualLogMassForMillToZero, 355
SetDeliveredLogMillCostsToZero, 377
SetEstimateStandsVolumeToZero, 308
SetFeasibleSolutionToInfeasible, 358
SetLHaulTransportStandOptionCostsToZero, 370
SetMassToZero, 309
SetMillLogPurchasesToZero, 376
SetNumberOfStandOptionsToOne, 342
SetObjectiveFunctionToZero, 384
SetPSolutionStandOptionCostsToZero, 379
SetRegimeStandOptionCostsToZero, 365
SetSHaulTransportStandOptionCostsToZero, 368
SetStandOptionCostsToZero, 373
SetStandVolumeAndMassToZero, 300
SetTotalNonStandForestryCostsToZero, 382
SetTransportStandOptionCostsToZero, 372
ShortHaulRoad, 174, 315
ShortHaulTransport, 321
ShortHaulTrip, 175, 318
SiteIndexEvalAge, 298
StandBelongsToEstate, 295

StandCharacteristics, 297
StandHarvestingInfo, 344
StandLand, 297
StandOfTrees, 301
StandOptionCosts, 373
StandPlantingStatus, 298
StandPlantingStatusUnplanted, 298
StandRegimeAction, 343
StandSiteIndex, 299
StandSuitableSpecies, 296
StandVolumeAndMass, 300
StandsVolumeAndMassAtHarvesting, 310
StoreRegimes, 284
TotalNonStandForestryCosts, 382
TrackActualRegimes, 282
Transport, 325
TransportAndUnloadLogsAtDepot, 321
TransportAndUnloadLogsAtMill, 325
TransportStandOptionCosts, 372
TripIDs, 314
Truck, 317
TruckMaxLoad, 316
UpdateHarvestingOutcomes, 352
UpdateHarvestingOutcomesPlantedStands, 351
UpdateHarvestingOutcomesUnplantedStands, 350
UpdateMillForStandsLogs, 388
WoodGeneticMaterial, 276
YearVariables, 363