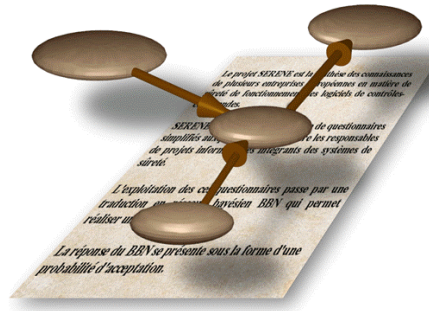


# SafEty and Risk Evaluation using bayesian NEts : SERENE



## Task 5.3 Report:

### The SERENE Method Manual      Version 1.0 (F)

The SERENE Partners: CSR, EdF, ERA, OT, TUV

EC Project No. 22187

Project Doc Number SERENE/5.3/CSR/3053/R/1

Internal Doc Ref

D:\NormanFiles\projects\serene\deliverables\R133fmth.doc

Commercial-in-confidence

This document is the property of the SERENE Partners and is provided in confidence on the basis that no unauthorised disclosure is permitted.

Approved by:

---

William Marsh  
Project Manager  
ERA Technology Ltd

14 May 1999

Copyright 1999 SERENE Partners

No part of this document may be photocopied or otherwise reproduced without the prior permission in writing of the SERENE partners. Such written permission must also be obtained before any part of this document is stored in a retrieval system of whatever nature.

## Summary

This report has been prepared by the partners in EC Project No. 22187, entitled ‘Safety and Risk Evaluation Using Bayesian Nets: SERENE’. The overall objective of the SERENE Project was to develop a means of using Bayesian Belief Nets (BBNs) to reason about the safety of Programmable Electronic Systems (PES). This objective has been addressed by developing a method, the SERENE Method, for representing PES safety arguments using a BBN and by enhancing an existing BBN tool with features needed to support the method.

The report is addressed to engineers involved in developing or assessing safety-related Programmable Electronic Systems. The reader should have some familiarity with the processes of safety engineering and management, for example as described in the draft IEC 61508 ‘Functional Safety: Safety Related Systems’ (IEC, 1995). However, the SERENE Method is not dependent on any particular standard. No previous experience in the use of Bayesian statistics is assumed; appendices provide an introduction to Bayesian statistics and Bayesian nets, requiring only basic mathematical knowledge.

The SERENE Method is concerned with the functional safety of complex systems: functional safety concerns ‘the ability of a system to carry out the actions necessary to achieve or maintain a safe state’ (IEC 1995, adapted). Standards applicable to developing complex safety-related systems are based on consensus views of the ‘best practice’ in each aspect of the system development. The requirements of these standards typically cover the overall safety processes, the use of hazard analysis to determine safety requirements, techniques for the design of hardware and software for use in safety-related systems and general requirements for quality systems and configuration control.

The activities carried out during the development of a safety-related system provide evidence of safety. The evidence is used to construct a justification that a system is sufficiently safe to be operated in a specified environment. Existing standards and guidelines, appropriate to the requirements of different countries and industry sectors, cover many aspects of the safety justification, including the safety life-cycle and the acceptable forms of safety evidence. However, the standards do not explain how to combine the many different types of evidence into an overall safety justification. This is the problem addressed by the SERENE Method.

The approach to combining safety evidence used in the SERENE method is to model, using a BBN, the beliefs connecting each step of the development process to the overall safety of the system. The benefits of this approach are expected to include:

1. improved communication of the argument for safety
2. greater focus on the properties which lead to safety
3. a basis for empirical validation of the beliefs of safety experts and of the rationale for existing standards.

The benefits of the use of a BBN representation include:

1. the uncertainty associated with the causes of safety can be included in the model and the uncertainty about the overall system safety is explicit
2. the safety achieved can be quantified, allowing alternative safety strategies to be compared.

Following an introduction and overview of the method, this document describes how to build BBNs for combining safety evidence. The approach is based on two levels of BBN structure that can be reused. The first level consists of a small number of BBN ‘idioms’ that are introduced in Chapter 3. The idioms are commonly occurring generic BBN fragments that have been found to be the basis for most BBNs used in safety assessment. To build a BBN, the appropriate idiom is chosen and adapted to model each phenomenon that forms part of the safety justification. For example, each stage of the software development process can be described using a ‘process-product’ idiom, giving a model of the influences on the attributes of each product of the process.

At a higher level of BBN reuse, the SERENE partners have also developed a number of BBN templates. These serve both as examples of the way BBNs are constructed and could also be incorporated into a BBN prepared by a user of the SERENE Method. Chapter 4 describes some example templates. Chapter 5 demonstrates how the SERENE Tool is used to construct a BBN, using the techniques described in the earlier chapters.

The appendices provide background and tutorial material, including a reference manual for the SERENE tool.

# Contents

	Page No.
<b>1. INTRODUCTION AND BACKGROUND .....</b>	<b>15</b>
1.1 INTENDED AUDIENCE .....	15
1.2 FUNCTIONAL SAFETY OF COMPLEX SYSTEMS .....	15
1.3 OBJECTIVES OF SERENE .....	16
1.4 STRUCTURE OF THE METHOD MANUAL .....	16
<b>2. OVERVIEW OF THE SERENE METHOD .....</b>	<b>19</b>
2.1 SAFETY EVIDENCE AND ARGUMENT .....	19
2.2 USING A BBN TO REPRESENT A SAFETY ARGUMENT .....	20
2.2.1 <i>What is a BBN?</i> .....	20
2.2.2 <i>BBNs and SERENE</i> .....	22
2.2.3 <i>Help with defining the BBN topology</i> .....	23
2.2.4 <i>Help with defining the Node Probability Tables</i> .....	24
2.3 ACTIVITIES IN THE SERENE METHOD .....	25
2.3.1 <i>Argument Preparation</i> .....	25
2.3.2 <i>Argument Construction</i> .....	26
2.3.3 <i>Argument Use</i> .....	26
2.4 AUTOMATION OF THE SERENE METHOD .....	27
2.4.1 <i>Tool Features</i> .....	28
2.4.2 <i>Use of the SERENE Tool</i> .....	28
2.5 THE SERENE PARTNER SAFETY ARGUMENTS IN THE SERENE TOOL .....	29
2.5.1 <i>Hazard-based PES Safety Argument (ERA)</i> .....	29
2.5.2 <i>The life-cycle based argument (EDF)</i> .....	30
2.6 BENEFITS OF THE SERENE METHOD .....	30
<b>3. CONSTRUCTING SERENE SAFETY ARGUMENTS USING IDIOMS .....</b>	<b>33</b>
3.1 THE RATIONALE FOR USING IDIOMS .....	33
3.2 SOME IMPORTANT DEFINITIONS .....	33
3.3 IDIOM SPECIFICATIONS .....	34
3.3.1 <i>The Definitional/Synthesis Idiom</i> .....	35
3.3.2 <i>The Process-Product Idiom</i> .....	38
3.3.3 <i>Measurement Idiom</i> .....	40
3.3.4 <i>Induction idiom</i> .....	41
3.3.5 <i>Prediction/Reconciliation Idiom</i> .....	42
3.4 CHOOSING THE RIGHT IDIOM IN THE SERENE METHOD .....	43
3.5 JOINING IDIOMS TO MAKE ARGUMENTS .....	44
3.5.1 <i>The problem</i> .....	45
3.5.2 <i>The formal definition of the join operation</i> .....	47
3.6 THE SERENE METHOD: A COMPLETE EXAMPLE .....	49
3.6.1 <i>Task 1: List the key entities (products, process and resources)</i> .....	50
3.6.2 <i>Task 2: Determine the key attributes of the entities that are relevant for the safety argument</i> ..	50
3.6.3 <i>Task 3: Group together related attributes</i> .....	50
3.6.4 <i>Task 4: Determine the appropriate idioms that relate the attributes</i> .....	51
3.6.5 <i>Task 5: Define the NPTs for each node in each idiom</i> .....	52
3.6.6 <i>Task 6: Apply the join operation to build the complete safety argument</i> .....	53
3.6.7 <i>Task 7: Enter observations and make predictions with the safety argument template</i> .....	54
<b>4. EXAMPLES OF USING THE SERENE METHOD TO CONSTRUCT SAFETY ARGUMENTS</b>	<b>59</b>
4.1 THE DEFECT-DENSITY ARGUMENT .....	59
4.1.1 <i>Defect Density: Idiom Instances</i> .....	59
4.1.2 <i>Creating the Defect Density Argument from Templates</i> .....	60
4.1.3 <i>Defect Density: Execution Scenarios</i> .....	63
4.2 RELIABILITY ARGUMENT .....	64

4.2.1	<i>Reliability: Idiom Instances</i> .....	64
4.2.2	<i>Building the Reliability Argument</i> .....	67
4.2.3	<i>Reliability: Execution Scenarios</i> .....	68
4.3	<b>THE SAFETY RISK ARGUMENT</b> .....	70
4.3.1	<i>Idiom Instances in the Safety Risk Template</i> .....	71
4.3.2	<i>The Safety Risk Template</i> .....	72
4.3.3	<i>The Satellite Templates in Safety Risk Argument</i> .....	72
4.3.4	<i>Creating the Safety Risk Argument from Templates</i> .....	75
4.3.5	<i>Safety Risk: Execution Scenario</i> .....	76
<b>5.</b>	<b>USING THE SERENE TOOL TO BUILD A BBN FROM IDIOMS</b> .....	<b>80</b>
5.1	<b>BASIC FUNCTIONALITY OF THE TOOL</b> .....	80
5.2	<b>DEFECTS EXAMPLE</b> .....	84
5.2.1	<i>Step 1: Create instantiation of the definition idiom</i> .....	84
5.2.2	<i>Step 2: Create an instantiation of the measurement idiom</i> .....	93
5.2.3	<i>Step 3: Create instantiation of definition idiom: testing accuracy</i> .....	97
5.2.4	<i>Step 4: Combining the templates</i> .....	98
<b>6.</b>	<b>REFERENCES</b> .....	<b>104</b>
<b>7.</b>	<b>APPENDIX A: CONTEXT OF THE SERENE METHOD</b> .....	<b>110</b>
7.1	<b>SAFETY DEVELOPMENT AND ASSESSMENT</b> .....	110
7.2	<b>SAFETY REGULATIONS AND STANDARDS</b> .....	112
7.3	<b>RISK ASSESSMENT</b> .....	114
7.4	<b>SAFETY PRINCIPLES</b> .....	115
7.4.1	<i>Principles to determine acceptability of risk</i> .....	115
7.4.2	<i>The ALARP Principle</i> .....	115
7.5	<b>SAFETY LIFECYCLES</b> .....	116
7.6	<b>RELATIONSHIP TO SAFETY ANALYSIS TECHNIQUES</b> .....	117
7.6.1	<i>Failure Analysis Techniques</i> .....	117
7.6.2	<i>Process Quality</i> .....	118
7.6.3	<i>Metrics</i> .....	118
<b>8.</b>	<b>APPENDIX B: FUNDAMENTALS OF PROBABILITY THEORY</b> .....	<b>120</b>
8.1	<b>DIFFERENT APPROACHES TO PROBABILITY</b> .....	120
8.1.1	<i>Frequentist approach to probability</i> .....	120
8.1.2	<i>Bayesian approach to probability</i> .....	120
8.2	<b>PROBABILITY AXIOMS</b> .....	121
8.3	<b>VARIABLES AND PROBABILITY DISTRIBUTIONS</b> .....	121
8.4	<b>JOINT EVENTS AND MARGINALISATION</b> .....	122
8.5	<b>CONDITIONAL PROBABILITY</b> .....	123
8.6	<b>BAYES THEOREM</b> .....	123
8.6.1	<i>Bayes Theorem Example</i> .....	124
8.6.2	<i>Likelihood Ratio</i> .....	125
8.7	<b>CHAIN RULE</b> .....	125
8.8	<b>INDEPENDENCE AND CONDITIONAL INDEPENDENCE</b> .....	126
<b>9.</b>	<b>APPENDIX C: BAYESIAN BELIEF NETS TUTORIAL</b> .....	<b>128</b>
9.1	<b>DEFINITION OF BBNs: GRAPHS AND PROBABILITY TABLES</b> .....	128
9.2	<b>ANALYSING A BBN: ENTERING EVIDENCE AND PROPAGATION</b> .....	129
9.3	<b>THE NOTION OF 'EXPLAINING AWAY' EVIDENCE</b> .....	130
9.4	<b>WHY DO WE NEED A BBN FOR THE PROBABILITY COMPUTATIONS?</b> .....	131
9.5	<b>GENERAL CASE OF JOINT PROBABILITY DISTRIBUTION IN BBN</b> .....	131
9.6	<b>DEALING WITH THE INCREASES IN THE NUMBER OF VARIABLES</b> .....	132
9.7	<b>WHY WE SHOULD USE BBNs</b> .....	132
9.8	<b>HOW BBNs DEAL WITH EVIDENCE</b> .....	132
9.8.1	<i>Serial Connection</i> .....	133
9.8.2	<i>Diverging connection</i> .....	133

9.8.3	<i>Converging connection</i> .....	135
9.8.4	<i>The notion of d-separation</i> .....	135
9.9	<b>TYPES OF REASONING PERMITTED IN BBN'S</b> .....	136
9.9.1	<i>Causal determination</i> .....	136
9.9.2	<i>Statistical Determination</i> .....	139
9.9.3	<i>Structural Determination</i> .....	140
9.9.4	<i>Idioms and types of determination</i> .....	142
<b>10.</b>	<b>APPENDIX D: BBNS AND DECISION MAKING</b> .....	<b>144</b>
10.1	INTRODUCTION .....	144
10.2	THE EXAMPLE PROBLEMS.....	144
10.3	IDENTIFYING OBJECTIVE AND PERSPECTIVE .....	146
10.4	THE DECISION PROBLEM AND ITS CONSTITUENT PARTS .....	147
10.5	DEFINING CRITERIA.....	149
10.6	UNCERTAIN CRITERIA AND INFERENCE .....	150
10.6.1	<i>Criteria that require uncertain inference</i> .....	150
10.6.2	<i>External factors</i> .....	151
10.6.3	<i>Internal factors</i> .....	153
10.7	ANALOGY WITH GQM .....	153
10.8	SERENE AND DECISION MAKING: CONCLUDING REMARKS .....	154
<b>11.</b>	<b>APPENDIX E: BIASES AND FALLACIES IN REASONING ABOUT PROBABILITY</b> .....	<b>156</b>
11.1	REPRESENTATIVENESS .....	156
11.1.1	<i>The problem of base-rate neglect</i> .....	156
11.1.2	<i>Insensitivity to prior probability of outcomes</i> .....	157
11.1.3	<i>Insensitivity to sample size</i> .....	158
11.1.4	<i>Misperception of chance and randomness</i> .....	158
11.2	DENIAL OF UNCERTAINTY .....	158
11.3	AVAILABILITY .....	159
11.3.1	<i>Retrievability of instances</i> .....	159
11.3.2	<i>Illusory Correlation</i> .....	159
11.3.3	<i>Biases due to the effectiveness of a search set</i> .....	159
11.3.4	<i>Biases of imaginability</i> .....	159
11.4	ADJUSTMENT OF UNCERTAINTY .....	160
11.5	CONJUNCTION FALLACY .....	160
11.6	HINDSIGHT BIAS.....	161
11.7	DIFFICULTIES IN ASSESSING VARIANCE, COVARIANCE, AND CORRELATION .....	161
11.8	CONSERVATISM .....	162
11.9	OVERCONFIDENCE .....	162
11.10	FALLACIES ASSOCIATED WITH CAUSAL AND DIAGNOSTIC REASONING .....	163
<b>12.</b>	<b>APPENDIX F: BBNS AND BAYESIAN PROBABILITY RESOURCES</b> .....	<b>166</b>
12.1	PROBABILITY REFERENCES .....	166
12.2	REFERENCES ON ELICITING PROBABILITIES .....	166
12.3	BOOKS ABOUT BBNS.....	166
12.4	PAPERS ABOUT BBNS .....	167
12.5	BBN RELATED WEB SITES .....	167
12.6	BBN TOOLS .....	168
12.7	BBN PROJECTS .....	168
12.8	RELEVANT JOURNALS FOR BBNS .....	169
12.9	LECTURES/TEACHING MATERIAL FOR BBNS .....	169
<b>13.</b>	<b>APPENDIX G: SERENE TOOL REFERENCE SECTION</b> .....	<b>170</b>
13.1	ABOUT .....	171
13.2	ADD LINK .....	171
13.3	ADD NODE.....	171
13.4	CASE FILE.....	171
13.5	DELETE.....	172

13.6	DOMAIN .....	172
13.7	DOMAIN MENU .....	172
13.8	DOMAIN WINDOW .....	172
13.9	EDIT MENU .....	173
13.10	EVIDENCE .....	173
13.11	EXIT .....	173
13.12	EXPORT DOMAIN AS HKB .....	174
13.13	EXPORT DOMAIN AS NET .....	174
13.14	EXPORT TEMPLATE AS NET .....	174
13.15	EXPRESSION BUILDING .....	174
13.15.1	Constant Values.....	174
13.15.2	Discrete Distribution Functions.....	175
13.15.3	Continuous Distribution Functions.....	176
13.15.4	Arithmetic Functions.....	177
13.15.5	Boolean Functions.....	177
13.15.6	Comparison Operators.....	178
13.15.7	The if-then-else Function.....	178
13.16	EXPRESSION EDITOR .....	179
13.17	FILE MENU.....	180
13.18	GENERATE DOMAIN DOCUMENTATION.....	180
13.19	GENERATE TEMPLATE DOCUMENTATION .....	181
13.20	HELP .....	181
13.21	HELP MENU .....	181
13.22	IMPORT TEMPLATE FROM NET.....	182
13.23	JOINT PROBABILITY .....	182
13.24	LINKS.....	182
13.24.1	Causal Links.....	183
13.24.2	Information Links .....	183
13.24.3	Join Links .....	183
13.24.4	Utility Links.....	183
13.25	LOAD CASE FILE .....	184
13.26	MAKE DOMAIN .....	184
13.27	MONITOR WINDOW.....	184
13.28	NEW PROJECT .....	185
13.29	NEW TEMPLATE.....	185
13.30	NODE CATEGORIES .....	186
13.30.1	Chance Nodes.....	186
13.30.2	Decision Nodes.....	186
13.30.3	Utility Nodes.....	186
13.30.4	Abstract Nodes .....	186
13.31	NODE KINDS .....	187
13.31.1	Discrete Labelled .....	187
13.31.2	Boolean .....	187
13.31.3	Discrete Numbered.....	187
13.31.4	Interval.....	187
13.31.5	Continuous .....	187
13.32	NODE PROPERTIES .....	188
13.32.1	General tab.....	188
13.32.2	States tab.....	188
13.32.3	Probabilities tab.....	188
13.33	NODE TABLE.....	189
13.34	NODES .....	189
13.35	OPEN PROJECT .....	189
13.36	OPEN TEMPLATE.....	189
13.37	PROPAGATE .....	189
13.38	REMOVE TEMPLATE.....	190
13.39	RETRACT ALL EVIDENCE.....	190
13.40	SAVE CASE FILE .....	190



13.41	SAVE PROJECT .....	190
13.42	SAVE PROJECT AS.....	190
13.43	SAVE TEMPLATE .....	190
13.44	SAVE TEMPLATE AS.....	190
13.45	SENSITIVITY ANALYSIS.....	191
13.46	TEMPLATE .....	192
13.47	TEMPLATE DOCUMENTATION .....	192
13.48	TEMPLATE MENU.....	193
13.49	TEMPLATE WINDOW .....	193
13.50	TEMPLATES WINDOW .....	194
13.51	TOOLS MENU .....	194
13.52	WINDOW MENU .....	194
<b>14.</b>	<b>INDEX.....</b>	<b>195</b>

## Figures List

FIGURE 2-1: A SAFETY ARGUMENT WITHIN A SAFETY CASE .....	20
FIGURE 2-2: SYSTEM SAFETY ASSESSMENT BBN EXAMPLE .....	21
FIGURE 2-3: PROPAGATING EVIDENCE IN THE SYSTEM SAFETY ASSESSMENT BBN EXAMPLE.....	22
FIGURE 2-4: USING A BBN TO REPRESENT A SAFETY ARGUMENT.....	23
FIGURE 2-5: DISCOVERED DEFECTS.....	24
FIGURE 2-6: NUMBER OF RESIDUAL DEFECTS .....	25
FIGURE 2-7: SERENE TOOLSET ARCHITECTURE.....	28
FIGURE 2-8: USE OF THE SERENE TOOL.....	29
FIGURE 3-1: EXAMPLE OF PROCESS, PRODUCT AND RESOURCE ENTITY RELATIONS. ....	34
FIGURE 3-2: DEFINITIONAL/SYNTHESIS IDIOM.....	35
FIGURE 3-3: INSTANTIATION OF DEFINITIONAL/SYNTHESIS IDIOM (VELOCITY).....	36
FIGURE 3-4: INSTANTIATION OF DEFINITIONAL/SYNTHESIS IDIOM (RELIABILITY).....	36
FIGURE 3-5: INSTANTIATION OF DEFINITIONAL/SYNTHESIS IDIOM (SAFETY).....	36
FIGURE 3-6: INSTANTIATION OF DEFINITIONAL/SYNTHESIS IDIOM (TESTING QUALITY).....	37
FIGURE 3-7: IDIOM INSTANTIATION FOR A DEFINITIONAL HIERARCHY .....	38
FIGURE 3-8: THE PROCESS-PRODUCT IDIOM .....	39
FIGURE 3-9: PROCESS-PRODUCT IDIOM INSTANTIATION (SOFTWARE FAILURES).....	39
FIGURE 3-10: PROCESS-PRODUCT IDIOM INSTANTIATION (CODING PROCESS).....	40
FIGURE 3-11: MEASUREMENT IDIOM.....	40
FIGURE 3-12: MEASUREMENT IDIOM INSTANTIATION (TESTING).....	41
FIGURE 3-13: INDUCTION IDIOM.....	41
FIGURE 3-14: INDUCTION IDIOM INSTANTIATION (TESTING COMPETENCE) .....	42
FIGURE 3-15: PREDICTION/RECONCILIATION IDIOM .....	42
FIGURE 3-16: A PROCESS-PRODUCT IDIOM INSTANTIATION FOR TEST QUALITY.....	43
FIGURE 3-17: A DEFINITIONAL/SYNTHESIS IDIOM INSTANTIATION FOR TEST QUALITY .....	43
FIGURE 3-18: PREDICTION/RECONCILIATION IDIOM INSTANTIATION FOR TEST QUALITY .....	43
FIGURE 3-19: CHOOSING THE RIGHT IDIOM .....	44
FIGURE 3-20: TWO BBNs WITH COMMON NODE “# FAULTS” .....	45
FIGURE 3-21: THE BBNs OF FIGURE 3-20 JOINED AT NODE “# FAULTS” .....	46
FIGURE 3-22: ANOTHER BBN TO JOIN .....	46
FIGURE 3-23: THE BBNs OF FIGURE 3-21 AND FIGURE 3-22 JOINED AT NODE “TEST COVERAGE” .....	47
FIGURE 3-24: TWO BBNs READY FOR JOINING ON NODE $X_1, \dots, X_N$ .....	48
FIGURE 3-25: SCHEMATIC OF JOIN OPERATION (THE BBN B THAT RESULTS FROM JOINING THE BBNs IN FIGURE 3-24.....	48
FIGURE 3-26: JOINING BBNs IN THE SERENE TOOL.....	49
FIGURE 3-27: NPT FOR SYSTEM SAFETY NODE.....	53
FIGURE 3-28: COMPLETE SAEFTY ARGUMENT TEMPLATE .....	54
FIGURE 3-29: FULL NET IN INITIALISED STATE .....	55
FIGURE 3-30: LOW FAULTS OBSERVATION ENTERED .....	55
FIGURE 3-31: LOW FAULTS, HIGH TESTING ACCURACY .....	56
FIGURE 3-32: WORST EVIDENCE .....	56
FIGURE 3-33: 'BEST' EVIDENCE .....	57
FIGURE 4-1: IDIOM INSTANCES FOR DEFECT DENSITY TEMPLATE.....	60
FIGURE 4-2: ONE_PHASE_REWORK TEMPLATE.....	61
FIGURE 4-3: DENSITY TEMPLATE .....	61
FIGURE 4-4: TWO_PHASE_REWORK TEMPLATE .....	62
FIGURE 4-5: TWO_PHASE_REWORK TEMPLATE WITH DENSITY TEMPLATE ADDED.....	62
FIGURE 4-6: ONE_PHASE_REWORK TEMPLATE WITH EVIDENCE ENTERED.....	63
FIGURE 4-7: DENSITY TEMPLATE WITH EVIDENCE ENTERED .....	64
FIGURE 4-8: IDIOM INSTANCES FOR RELIABILITY TEMPLATE.....	66
FIGURE 4-9: RELIABILITY TEMPLATE .....	67
FIGURE 4-10: RELIABILITY TEMPLATE INITIAL STATES .....	68
FIGURE 4-11: RELIABILITY TEMPLATE BEST CASE SCENARIO .....	69
FIGURE 4-12: RELIABILITY TEMPLATE WORST CASE SCENARIO.....	70
FIGURE 4-13 IDIOM INSTANTIATIONS IN THE SAFETY TEMPLATE .....	71
FIGURE 4-14 SAFETY TEMPLATE BBN TOPOLOGY .....	72
FIGURE 4-15 DEVELOPER QUALITY TEMPLATE.....	73

FIGURE 4-16 SEVERITY TEMPLATE.....	74
FIGURE 4-17 TEST QUALITY TEMPLATE.....	74
FIGURE 4-18 TESTING GROUP QUALITY TEMPLATE.....	75
FIGURE 4-19 SAFETY_ARGUMENT TEMPLATE .....	76
FIGURE 4-20 TEST QUALITY TEMPLATE EVIDENCE ENTERED .....	77
FIGURE 4-21 SAFETY ARGUMENT TEMPLATE WITH EVIDENCE ENTERED.....	78
FIGURE 4-22 SAFETY TEMPLATE WITH POSITIVE TEST RESULTS.....	78
FIGURE 7-1: OVERALL SAFETY PROCESS .....	111
FIGURE 7-2: HIERARCHICAL DEVELOPMENT OF SAFETY SYSTEMS .....	112
FIGURE 7-3: THE INFLUENCE OF STANDARDS AND REGULATION ON THE SERENE METHOD.....	114
FIGURE 7-4: RISK REDUCTION.....	115
FIGURE 7-5: THE ALARP TRIANGLE .....	116
FIGURE 7-6: RELATIONSHIP OF SAFETY LIFECYCLE AND ARGUMENT .....	116
FIGURE 7-7: SERENE AND SAFETY ANALYSIS TECHNIQUES.....	118
FIGURE 10-1: DEFINITION OF SYSTEM DEPENDABILITY (CAN BE VIEWED AS AN INSTANTIATION OF THE DEFINITIONAL/SYNTHESIS IDIOM.....	149
FIGURE 10-2: BBN FOR PREDICTING THE UNCERTAIN CRITERIA IN THE TRAVEL EXAMPLE.....	151
FIGURE 10-3: CALCULATING VALUES OF THE UNCERTAIN CRITERIA .....	152
FIGURE 10-4: BBN TO PREDICT SOFTWARE FAILURE RATE .....	152
FIGURE 10-5: HOW THE SERENE APPROACH FITS IN WITH GQM AND MCDA .....	154

*This page intentionally left blank*

## Glossary

**AHP:** *Analytical Hierarchy Process*. A method for performing multi-criteria decision analysis

**Attribute:** a property of an entity. Attributes can be represented as nodes of a BBN.

**Abstract node:** a set of nodes or sub-net treated as a single abstract concept. Abstract nodes have input and output nodes from which they can be joined to any other node.

**BBN:** *Bayesian Belief Net*. A BBN is a special type of diagram, or graph, together with an associated set of probability tables (see NPT). The graph consists of nodes and associated arcs. In the SERENE method, nodes of the BBN represent entities and their attributes; the arcs describe the relationships between these. BBNs are sometimes known as Graphical networks, graphical probability tables, causal nets, or influence diagrams.

**E/E/PE:** *Electrical/Electronic/Programmable Electronic Device*. A device based on electrical and/or electronic and/or programmable electronic technology.

**Entity:** thing or event of relevance to the safety objective. In SERENE the entities we are interested in are:

- Processes which are activities carried out by human or machine.
- Resources which are the miscellany of items (including people and machines) considered as inputs to the process and
- Products which are artefacts that are outputs from processes.

**EUC:** *Equipment under control*. Equipment/machinery/apparatus/plant used for manufacturing, process, transportation, medical or other activities for which designated safety-related systems could be used to prevent hazardous events associated with the EUC from taking place, or mitigate the effects of the hazardous event. (IEC61508)

**Functional safety:** The ability of a safety-related system to carry out the actions necessary to achieve or maintain a safe state for the equipment under control. (IEC61508)

**GQM:** *Goal-Question-Metric*: A top-down approach to measurement

**IEC:** *International Electrotechnical Commission* Standards body.

**Idiom:** Generic atomic component of safety argument that has been found to recur in many different safety arguments and which therefore forms a reusable pattern. Idioms are abstractions of general methods of reasoning with BBNs.

**Idiom instantiation:** Particular implementation of idiom in which the nodes have names of real entities/attributes. These model real world relations between real entities, however they do not necessarily contain real world NPTs.

**MCDA:** *Multi-Criteria Decision Aid*. The name given to the collection of methods used for solving decision problems when there are multiple criteria.

**NPT:** *Node probability table*. Every node A in a BBN has an associated Node Probability Table, the size of which is determined by the number of parent nodes of A. For example, if node A has parent nodes B and C then the NPT for node A expresses the probability  $p(A|B,C)$  for all possible combinations of A, B and C. If A has x states, B has y states and C has z states then the NPT has xyz cells. If node A is a root node (i.e. it has no parents), then the NPT simply lists the x prior probabilities for each state of node A.

**PES:** *Programmable Electronic System* A system based on one or more E/E/PE(s), connected to (and including) input devices and/or output devices for the purpose of control, protection or monitoring. (IEC61508)

**Root node:** A node of a BBN that has no incoming arcs.

**Safety:** Freedom from unacceptable risk of harm (IEC61508)

**Safety analysis techniques:** Techniques which generate safety evidence, eg. HAZOP, FTA, FMECA, and others. The safety evidence generated is then used by the SERENE method in order to represent a safety argument.

**Safety argument:** A link between the safety evidence and the safety objective that allows predictions to be made concerning the sufficiency of the safety evidence in demonstrating that the safety objective has been achieved. A safety argument is represented as a BBN.

**Safety case:** This term is not used in the SERENE method. The SERENE method is only concerned with the safety argument.

**Safety claim:** This term is not used in the SERENE method.

**Safety evidence:** Facts about a system or its development history, assessed or measured during or after the development of the system, with a positive or negative influence on safety.

**Safety integrity:** The probability of a safety-related system satisfactorily performing the required safety functions under all the stated conditions within the stated period of time. (IEC61508)

**Safety life-cycle:** The necessary activities involved in the implementation of safety-related systems, occurring during a period of time that starts at the concept phase of a project and finishes when none of the safety-related systems are any longer available for use. (IEC61508)

**Safety objective:** A safety-related goal eg. that a system meets its functional requirements, or that a particular safety requirement can be met.

**Safety-related system:** A system that implements the required safety functions necessary to achieve or maintain a safe state for the equipment under control, OR is intended to achieve, on its own or with other safety-related systems, the necessary level of safety integrity for the implementation of the required safety functions. (IEC 61508).

In the SERENE method, the safety-related system is also defined according to its system boundaries which *must* include an E/E/PE.

**Safety sub-argument:** Safety arguments can be composed of safety sub-arguments. Safety sub-arguments are sub-networks of the whole BBN. Example: A safety argument might contain two safety sub-arguments - one leading to a prediction of system reliability and another leading to predictions about failure severity.

**SERENE:** *SafEty and Risk Evaluation using bayesian Nets*. ESPRIT Framework IV project 22187. Partners CSR, EdF, ERA, OT, TUV.

**Template:** A safety argument or safety sub-argument pre-prepared from another or the same organisation which could possibly be reused elsewhere. Two forms of reuse are available: reuse of the graph topology and reuse of the NPTs. Templates can be generic or specialised for a particular system type or line of reasoning. Templates are BBNs. The simplest templates are instantiations of idioms.

**Team Expertise:** The expertise of the project team, which results from a combination of training and experience, and may be considered to have an influence on the safety objective.

# 1 Introduction and background

## 1.1 Intended Audience

This report has been prepared by the partners in EC Project No. 22187, entitled 'Safety and Risk Evaluation Using Bayesian Nets: SERENE'. It is addressed to engineers involved in developing or assessing safety-related Programmable Electronic Systems. The reader should have some familiarity with the processes of safety engineering and management, for example as described in the draft IEC 61508 'Functional Safety: Safety Related Systems' (IEC, 1995). However, the SERENE Method is not dependent on any particular standard.

No previous experience in the use of Bayesian statistics is assumed; appendices provide tutorial material on Bayesian statistics and Bayesian nets, requiring only basic mathematical knowledge.

## 1.2 Functional Safety of Complex Systems

The SERENE Method is concerned with the functional safety of complex systems, particularly programmable electronic systems which fall within the scope of draft IEC 61508 and similar standards. Functional safety concerns 'the ability of a system to carry out the actions necessary to achieve or maintain a safe state' (IEC 1995, adapted). In a complex system the demonstration of functional safety must take account of both random and 'systematic' failures. Systematic failures include those that result from design errors. All complex systems are potentially subject to systematic failures, but this difficulty applies most of all to software, for which systematic failures are the only form of failure.

Traditional reliability techniques account for randomly occurring failures but do not take account of systematic failures. This arises because reliability analysis depends on knowledge of the failure modes that result from wear-out and other random processes. In contrast, systematic failure modes are unknown since any known errors are removed during development. If the specification of the properties required for safety - the safety requirements - is wrong, the system can even behave unsafely without a failure occurring.

Standards applicable to developing complex safety-related systems are based on consensus views of the 'best practice' in each aspect of the system development. The requirements of these standards typically cover the overall safety processes, the use of hazard analysis to determine safety requirements, techniques for the design of hardware and software for use in safety-related systems and general requirements for quality systems and configuration control. A number of limitations of the existing 'best practice' approach to safety have been suggested (Fenton, 1997), including:

1. lack of a rationale for the requirements of standards, based on the contribution to safety
2. no quantification of the resulting safety.

It is to be assumed that each requirement of a standard such as IEC 61508 is intended to contribute to safety. However, there is normally no rationale given to explain how the required measures increase safety. For example, IEC 61508 requires that the 'design method chosen shall possess features that 'facilitate ... modularity'. It is generally agreed that modular organisation of software is better than a monolithic organisation; however, what is the connection between modularity and safety? It could be that modularity makes testing easier, or that it makes a program easier to understand, or a combination of these and other factors. With no understanding of the rationale, there is a possibility that the benefits will not be obtained. For example, some design tools or languages provide automatic checking of module interfaces, others do not. Will modularity be beneficial in both cases?

The 'best practice' approach does not attempt to quantify the safety that results from meeting the requirements of standards. Instead, the best practice approach is based on *compliance*: if the system meets all the requirements of the standard, the system is assumed to be sufficiently safe. If any requirements are not met, the safety is insufficient. A quantitative approach - if it were possible - would allow different safety processes to be compared, or the cost of an increase in the safety margin to be investigated. No system can be perfectly safe, even one that complies fully with the requirements of a safety standard; in some critical applications the lack of a method to quantify failure rates has prevented the use of programmable systems.

### 1.3 Objectives of SERENE

The SERENE method addresses the lack of quantification of safety in the current standards-based approach. The overall objective of the project was to use Bayesian Belief Nets (BBNs) to reason about the safety of Programmable Electronic Systems (PES). This objective has been addressed by developing a method, the SERENE Method, for representing a PES safety arguments using BBNs and by enhancing an existing BBN tool with features needed to support the method.

In the SERENE Method, a safety argument is a prediction of one or more properties of a system relevant to safety. The prediction is based on all the relevant evidence. In addition to the overall purpose of the safety argument the SERENE method satisfies a number of secondary objectives:

1. rationally combine different sources and types of evidence in a single model
2. identify weaknesses in the argument such that it can be improved
3. identify weaknesses in products and processes to aid process improvement
4. specify degrees of confidence associated with predictions
5. provide a sound basis for rational discussion and negotiation about the systems development and deployment.

### 1.4 Structure of the Method Manual

This manual is organised as follows:

Chapter 2 introduces the notion of a BBN and provides an overview of how the SERENE method is used to build a safety argument and then make predictions from it. It describes the activities of the three phases of the method: argument preparation, argument construction, and argument use. It includes a brief introduction to the SERENE tool, and a summary of the safety arguments BBNs built by the partners using the method and tool. This chapter is essential reading for all SERENE users.

In Chapter 3 the SERENE method is described in detail. The important concept of a *BBN idiom* is introduced. An idiom is a fragment of a BBN that captures an often-occurring pattern of reasoning. A collection of idioms has been prepared by the SERENE project to ease the task of building BBN models into safety arguments. The Chapter includes a full example of the SERENE method from start to finish.

Chapter 4 gives a number of examples of the use of the SERENE Method and Tool. The safety arguments included in this chapter are generic. They differ in both the objective of the prediction and the factors taken into account. This illustrates the range of safety arguments to which the SERENE Method can be applied. SERENE users building safety arguments for their own systems can use the example safety arguments as 'templates'.

Chapter 5 provides a tutorial on how to use the SERENE tool to build a BBN argument from idioms.

Background and foundation chapters are found in the appendices. These are intended for readers who are not already familiar with one or more of the technical foundations of the SERENE Method. Also



the appendices provide additional guidance to SERENE users during specific stages of applying the method.

Appendix A describes the context of the SERENE method, including the relationship between SERENE and other methods used for ensuring the safety of electronic and programmable systems.

Appendix B contains a basic introduction to Bayesian probability. Readers uncertain of the differences between the Bayesian and traditional ‘frequentist’ approaches to probability, or confused by terms such as ‘conditional probability’ or ‘conditional independence’ are likely to find this appendix useful.

Appendix C provides a primer on the theory of BBNs and the types of reasoning that can be represented by them. This chapter is essential reading for users without previous experience of BBNs.

Appendix D explains the broader decision making context in which SERENE lies. While the SERENE method can provide a prediction about the safety or reliability of a specific system, it cannot of course make *decisions* such as whether or not to deploy the system. This appendix explains how BBNs, as used on SERENE, can integrate with higher level decision analysis methods.

Appendix E explains the common fallacies in probability elicitation and reasoning and how to avoid them.

Appendix F provides a list of resources for BBNs and probability theory.

Appendix G is a reference section for the SERENE tool.

*This page is intentionally left blank*

## 2 Overview of the SERENE Method

In this chapter we provide an overview of the SERENE method. It is structured as follows:

- Section 2.1 defines the central notions of safety evidence and safety arguments
- Section 2.2 provides an introduction to BBNs and explains how they are used to model safety arguments
- Section 2.3 describes the three activities of the SERENE method - argument preparation, argument construction, and argument use
- Section 2.4 provides an overview of the SERENE tool
- Section 2.5 provides a summary of the SERENE partner safety argument BBNs as they were built using the SERENE tool (the full BBN descriptions appear in appendices)
- Section 2.6 summarises the benefits of the SERENE method over other approaches to safety argumentation.

### 2.1 Safety Evidence and Argument

The specifications, analyses, reviews and testing carried out during the development of a safety-related system provide evidence of safety. The evidence is used to construct a justification that a system is sufficiently safe to be operated in a specified environment. The justification of safety, sometimes called a safety case, may be submitted to safety regulators for approval. Safety legislation may place detailed requirements on the scope and contents of a safety case. Since these requirements differ in different countries and industry sectors, we do not wish to depend upon a particular detailed prescription for a safety case. Instead, the SERENE Method focuses on the *safety argument*.

A *safety argument* provides a 'link between the safety evidence and a safety claim, showing that the safety evidence is sufficient to support the claim'. The term 'safety argument' is sometimes used as a synonym for safety case. Here, we use safety argument to mean that part of the safety case that combines the safety evidence, showing that the evidence is sufficient to demonstrate that the system is acceptably safe. The use of the term safety argument for a specific part of the safety case is illustrated in Figure 2-1. A typical safety case also references applicable standards and regulations, derives safety targets, gives an overview of the system and its operation, and contains or references safety evidence.

Standards such as draft IEC 61508 and assessment frameworks such as the GAM framework (developed by the CASCADE ESPRIT project) address the collection of safety evidence, related to both the development process and to the final product. However, methods for combining diverse evidence into an overall safety justification have not been addressed in any previous safety standards. This is the problem addressed by the SERENE Method.

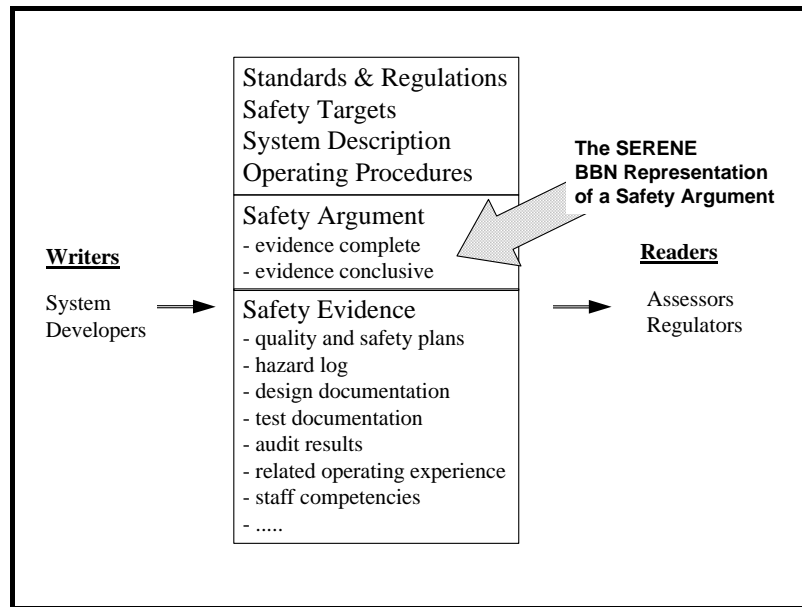


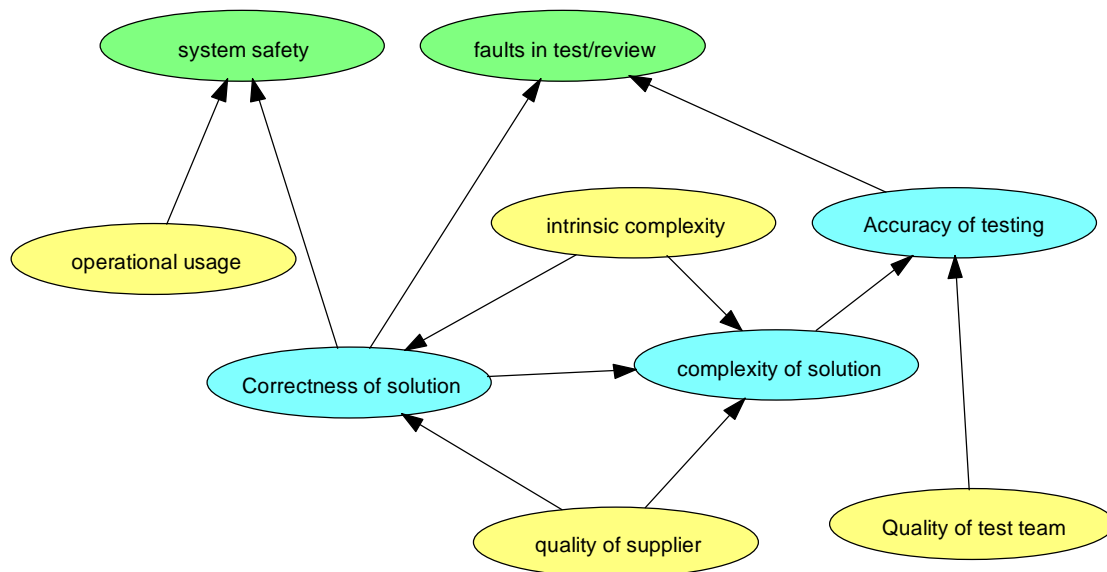
Figure 2-1: A Safety Argument within a Safety Case

## 2.2 Using a BBN to Represent a Safety Argument

The kind of 'safety' evidence that arises when developing or assessing a critical system is characterised both by its diversity and its *uncertainty*. For example, the number of defects discovered during testing such a system will be dependent on uncertain factors like *the number of inserted defects* and the *accuracy of the testing process*. The individual factors are diverse in the sense that some (like number of defects) may be objectively measurable whereas others (like accuracy of testing) are more subjective. It is crucial to be able to combine such diverse types of evidence. For example, in assessing system reliability it is well known that it is not possible to assure reliability at very high levels using product failure data alone. However, it seems reasonable to believe that we could assure reliability at higher levels if we could incorporate not just the results from testing but also other evidence about the process and product. Bayesian Belief Nets (BBNs) provide the best and most mature quantitative formalism for combining diverse, uncertain information.

### 2.2.1 What is a BBN?

A BBN is a directed graph, together with an associated set of probability tables. The graph consists of nodes and arcs as shown in Figure 2-2. The nodes represent variables, which can be discrete or continuous. For example, the node 'faults in test/review' is discrete having values 0,1,2,..., whereas the node 'system safety' might be continuous (such as the probability of failure on demand). The arcs represent causal/influential relationships between variables. For example, the number of faults in test/review is influenced by the correctness of the solution and the accuracy of testing, hence we model this relationship by drawing appropriate arcs as shown.



**Figure 2-2: System safety assessment BBN example**

The key feature of BBNs is that they enable us to model and reason about *uncertainty*. The BBN forces the assessor to expose all assumptions about the impact of different forms of evidence and hence provides a visible and auditable dependability or safety argument.

Consider the example of Figure 2-2. Suppose we knew that the correctness of the solution was ‘low’ and the accuracy of testing is ‘high’. Then the probability of finding a high number of faults in test would be more than if we knew the correctness of the solution was ‘high’ and the accuracy of testing is ‘low’. In the BBN we model this kind of dependence between uncertain variables by filling in a *node probability table* (NPT). Thus, for the node ‘faults in test/review’ the NPT might look like that shown in Table 2-1. The NPTs capture the conditional probabilities of a node given the state of its parent nodes.

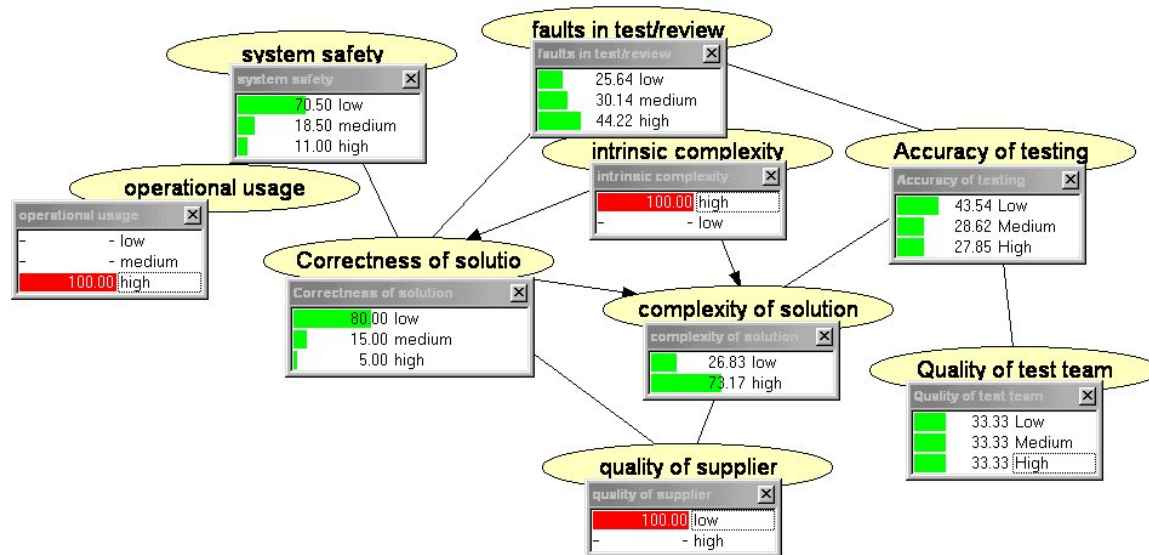
**Table 2-1: Faults found in inspection/testing**

Accuracy of testing	Low			Medium			High		
	Low	Medium	High	Low	Medium	High	Low	Medium	High
Correctness of solution									
Low	.33	.33	.3	.2	.33	.5	.1	.25	.7
Medium	.33	.33	.3	.3	.33	.3	.2	.5	.2
High	.33	.33	.4	.5	.33	.2	.7	.25	.1

BBNs are a way of describing complex probabilistic reasoning. The advantage of describing a probabilistic argument via a BBN, compared to describing it via mathematical formulas and prose, is that the BBN represents the structure of the argument in an intuitive, graphical format. The main use of BBNs is in situations that require statistical inference — in addition to statements about the probabilities of events, the user knows some evidence, that is, some events that have actually been observed, and wishes to infer the probabilities of other events, which have not as yet been observed. Using probability calculus and Bayes theorem (Appendix B) it is then possible to update the values of

all the other probabilities in the BBN. This is called *propagation*. Bayesian analysis can be used for both ‘forward’ and ‘backward’ inference.

For example, suppose we know that the quality of the supplier is ‘low’ and the intrinsic complexity of the problem is ‘high’. Then if we enter these facts and propagate their effects through the BBN it will update all of the other probabilities, giving for example a revised value for the probability of system safety. Figure 2-3 shows the actual results of entering different types of evidence (evidence is shown as a red bar).



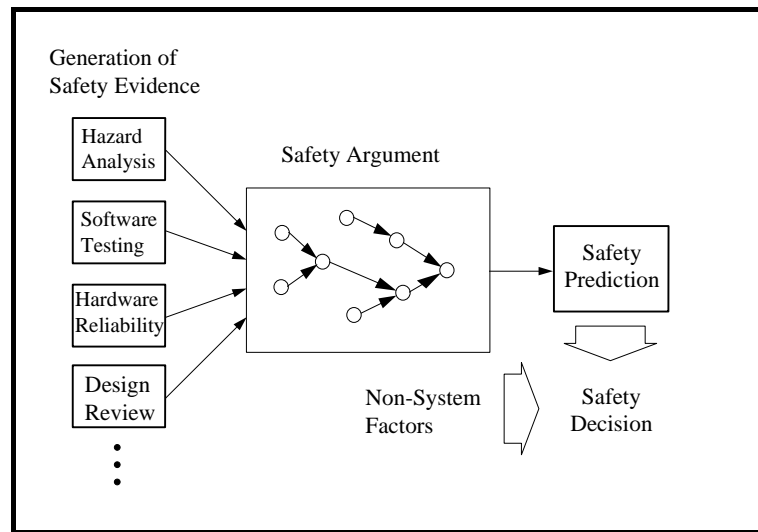
**Figure 2-3: Propagating evidence in the system safety assessment BBN example**

Although Bayesian probability has been around for a long time it is only in the last few years that efficient algorithms (and tools to implement them) have been developed to enable propagation in networks with a reasonable number of variables. The Bayesian propagation computations shown in Figure 2-3, even for an example as small as this, are very complex and cannot be calculate manually. Here we have used the Hugin tool, that implements the breakthrough propagation algorithm of Lauritzen and Spiegelhalter. With this tool it is possible to perform fast propagation in large BBNs (with hundreds of nodes and millions of state combinations). The recent explosion of interest in BBNs is due to these developments which mean that for the first time realistic size problems can be solved. These recent developments, in our view, make BBNs the best method for reasoning about uncertainty. To date BBNs have proven useful in applications such as medical diagnosis and diagnosis of mechanical failures. Their most celebrated use has been by Microsoft - BBNs underlie the help wizards in Microsoft Office and also underlie the interactive printer fault diagnostic system on the Microsoft web site.

## 2.2.2 BBNs and SERENE

Figure 2-4 illustrates the use of a BBN to represent a safety argument. Established system safety and software engineering techniques, such as hazard analysis inspections and software testing, are used to generate safety evidence. The safety evidence determines the inputs to a BBN, which predicts one or more properties of the system relevant to safety. Figure 2-4 also shows that to reach a decision about safety, other ‘non-system factors’ need to be taken into account, in addition to the safety prediction. The factors include legislation, regulation and the determination of safety targets. The SERENE

Method is not a decision making method and does not address these topics. However, Appendix D explains how BBNs as used in SERENE fit into a broader decision making framework.



**Figure 2-4: Using a BBN to Represent a Safety Argument**

Although the benefits of using BBNs to reason about uncertainty are widely recognised, it is not easy to build a ‘good’ BBN to model a complex reasoning process. As with any modelling method, the accuracy of the results is constrained by the accuracy of the model. The SERENE method, therefore, is explicitly concerned with helping to build good BBNs for the particular application domain of safety arguments. Thus, in SERENE we have provided novel solutions to help users tackle the two problems that constitute the major challenges in building a good BBN, namely:

- defining the BBN topology (getting the ‘right’ collection of nodes and arcs)
- defining the NPTs (the probabilities for each node).

We deal with these in turn next.

### 2.2.3 Help with defining the BBN topology

In Phase 1 of SERENE the partners built various BBNs that modelled different perspectives of a safety argument. Despite the different perspectives, and the different variables that appeared as nodes in these BBNs, we identified a small number of generic BBN fragments (called *idioms*) that occurred frequently. It appears that most BBNs in this application domain can be built from instantiations of this handful of idioms. The idioms are described in detail in Chapter 3. Instantiations of idioms are called *templates*. Commonly used templates are available to users of the SERENE method to reuse in building their own safety argument BBN. The SERENE method provides guidance in selecting appropriate idioms, and also provides a mechanism for joining templates with common nodes to build complete BBNs. The notion of joining templates is critical in that it provides a means of building large BBNs from small components. To manage the complexity of building large BBNs the SERENE method and tool also provides the notion of ‘abstract’ nodes. These are templates in which the only nodes shown are those which can be joined to other templates. In this way the SERENE method and tool provides an object oriented approach to developing BBNs. This in itself represents a breakthrough in BBN technology. Chapter 4 provides examples of how to build complete BBN arguments using idioms and templates, while Chapter 5 shows how this approach is automated within the SERENE tool.

In addition to the idioms and templates, the SERENE method also provides a number of safety argument templates. These are BBNs that represent complete or partial safety arguments that have been built by the partners to represent different perspectives. These templates can be reused ‘as is’ or modified in various ways, such as by changing the NPTs, changing variables names or incorporating them with other templates or idiom instantiations.

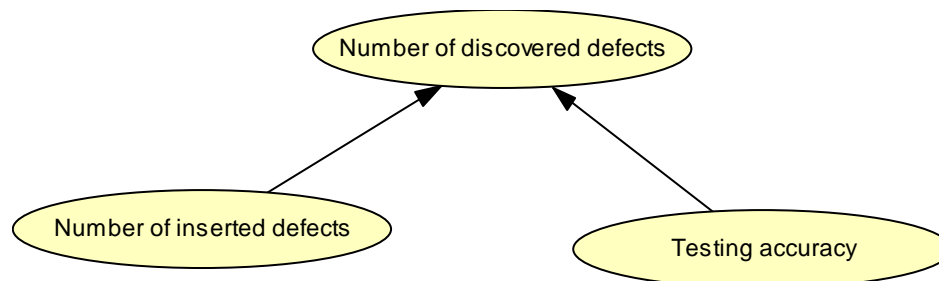
### 2.2.4 Help with defining the Node Probability Tables.

The task of defining NPTs is normally the most difficult and time-consuming aspect of BBN construction. To appreciate the complexity, consider the relatively trivial example NPT of Table 2-1. This is for a node that can take on 3 values and that has two parents each of which can take on 3 values. This results in an NPT with 27 different state combinations. Each of these 27 separate probability values normally has to be elicited from an expert. In more realistic examples, we would expect variables to take on many states (possibly even an infinite number such as a variable representing ‘number of faults’). In such circumstances it is impossible to elicit all the relevant probability values. Over the years people building BBNs have attempted to ‘solve’ the problem in very crude ways such as:

- restructuring the BBN topology so that no node has more than two parents
- restricting severely the number of values that a node can assume (such as in our own example above where ‘number faults found in test’ is restricted to the 3 values ‘low’, ‘medium’, ‘high’ rather than 0,1,2,3,..).

Solutions such as these often result in models that are over-simplistic, inaccurate, or simply not useful because they are too ‘coarse’. It turns out, however, that in many situations it is not necessary to extract probabilities for all possible state combinations. Moreover, in some situations it is not necessary to extract any probabilities at all. Consider the following examples:

**Example 1:** Figure 2-5 shows a simple BBN in which the ‘number of discovered defects’ has two parents: the ‘number of inserted defects’ and the ‘testing accuracy’.

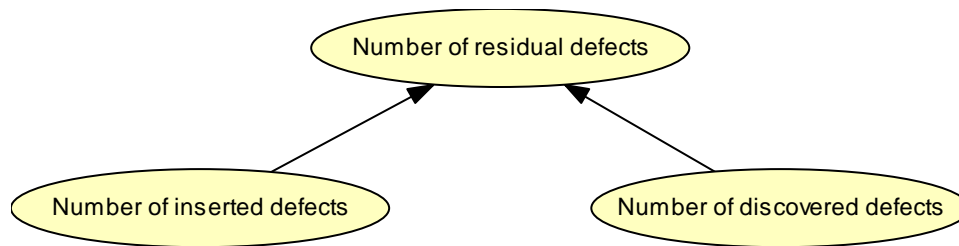


**Figure 2-5: Discovered defects**

Suppose that both of the ‘defects’ nodes have values  $\{0,1,2,3,\dots\}$ . Suppose that testing accuracy takes on the values  $\{0, 0.1, 0.2, \dots, 0.9, 1\}$  where 1 represents perfect accuracy and 0 represents the opposite extreme. Although the NPT for ‘number of discovered defects’ is infinite and therefore impossible to complete, there are intuitively simple ways to describe the probabilities. Specifically ‘number of discovered defects’ can be defined by a *probability distribution function*. There are a number of standard probability distribution functions, such as the Binomial, Poisson, Uniform, Normal, Gamma, and Beta. The shape of any of these functions is determined by a small number of parameters. The task of defining the complete distribution therefore reduces simply to defining the parameters. For example the binomial distribution models the number of successful ‘trials’ out of a sample of  $n$  trials. It has two parameters  $n$ , the total number of trials, and  $p$  the probability that any single trial will be successful. In the example above it would be reasonable to assume that ‘number of discovered defects’ is a Binomial distribution where  $n$  is the *number of inserted defects* and  $p$  is the *accuracy of testing*.

**Example 2:** Figure 2-6 shows a simple BBN in which the ‘number of residual defects’ has two parents: ‘number of inserted defects’ and ‘number of discovered defects’.





**Figure 2-6: Number of residual defects**

In this case the relationship between the child node and its parents is completely deterministic. The ‘number of residual defects’ is exactly equal to ‘number of inserted defects’ minus ‘number of discovered defects’.

In both of the above examples we ought to be able to define the NPT in terms of the appropriate function (either probabilistic as in example 1 or deterministic as in example 2) rather than having to fill in every individual probability cell. The SERENE tool achieved a major breakthrough in BBN technology by incorporating precisely this facility. Specifically, for any node in a BBN users can select one of the following options:

- *Define the NPT manually*
- *Define the NPT in terms of a function:* In this case the user calls on the tool’s *expression editor* either to define a deterministic function using normal arithmetic or use pre-defined distribution functions. They can also combine functions in any way that is appropriate.

While the expression editor provides an extremely powerful way for defining NPTs there are still inevitably going to be occasions when NPTs will need to be entered manually. To do this there are two choices. If there is real data from past experience, this can be used to generate the probabilities. If not, we have to enter so-called *subjective* probability assessments. If the constructor of the BBN is an expert in safety, he or she may input their own assessments of the probabilities. If not, they will need to *elicit* such probabilities from an expert. In either case the user must be aware of the common biases that are known to occur when such subjective probabilities are made. Appendix B provides a comprehensive description of these biases and how to avoid them, and hence forms an important background component of the SERENE method.

## 2.3 Activities in the SERENE Method

The SERENE Method is made up of three activities:

1. argument preparation
2. argument construction
3. argument use.

Each of these activities is described in more detail, below.

### 2.3.1 Argument Preparation

Argument preparation is primarily about identifying all the relevant evidence necessary for building the safety argument. This evidence is descriptions of relevant variables i.e. *entities*, such as processes, products or resources (see Section 3.2 for further discussion of these terms) and their *attributes* or properties. Each item of evidence will be a node of the eventual BBN that represents the safety argument.

It is important firstly, to identify the objective or goal of the argument. For example, in Chapter 4, the first example is a small net entitled the Defect Density Template. The goal of this net is to make predictions about the attribute, defect density, of the entity source code. Defect density is defined

(Section 4.1) as the *number of residual defects* divided by the *estimated size of design*. These then are two more attributes of relevance to this Defect Density argument. The attributes and entities that affect these, are also listed. For example, *testing effort*, *design size*, *problem complexity*, *introduced defects* and so on.

At this stage, there is no *need* to consider the relationships between these entities and attributes. The important thing is simply to get the list down. Relevant standards and life-cycles must also be considered. In summary, for Argument Preparation we should:

1. identify the goal, i.e. what it is you want to make predictions about
2. identify those entities and attributes whose effects on your goal you know you will want to vary and test
3. identify other sub-goals, entities and attributes involved in the prediction, including those relating to development processes and product features etc.
4. examine the development life-cycle and specify key processes
5. discuss informal arguments that you might deploy if arguing for or against a safe product
6. examine relevant standards/procedures/guidelines.

### 2.3.2 Argument Construction

Argument construction is the process of building up the BBN. In summary this consists of the following steps:

1. *Identify the nodes of the BBN*: The candidate nodes of the BBN are those variables identified in the argument preparation phase
2. *Define the 'type' of each node*. Allowable types in SERENE are:
  - *Discrete labelled*
  - *Boolean*
  - *Discrete Numbered*
  - *Interval*
  - *Continuous Gaussian*

For all types except the Boolean and Continuous Gaussian it is also necessary to define the range of values.

3. *Build the BBN topology*: The recommended SERENE approach is to think in terms of idioms as a means of determining relationships between nodes. The method provides guidelines in how to select the most appropriate idiom, and how to combine them to build a complete argument or 'safety template'. Although this approach is 'bottom-up', sometimes it is easier to do it as a 'top-down' process by looking for similar patterns in previous examples of nets.
4. *Define the node probability tables (NPT's) for each node*: As discussed above, for each node this is either done manually (using expert elicitation or probabilities based on real data) or by using function definitions through the SERENE tool's expression editor (which enables us to define functions in terms of pre-defined probability distributions or deterministic functions).

### 2.3.3 Argument Use

Once the BBN has been constructed, it is ready for use. This means that you can enter data for any of the variables or nodes in your safety argument and consider the combined effect that this data has on

all the other connected nodes and the overall outcome. In other words, you can generate any number of execution scenarios and achieve a predicted outcome in probabilistic terms.

Of course, it is important to check first that the argument construction stage has been completed satisfactorily, i.e. that the reasoning on which the BBN is built is sound and that the NPT's are realistic. In order to do this, some hypothetical scenarios can be generated specifically to validate the argument, as opposed to for their predictive value. In practice, it is most likely that the whole process will be an iterative one of testing and improving.

One aspect of argument use using the SERENE tool that can be particularly useful, is the ability to test out which particular items of evidence are actually the most important influences on the overall predicted outcome. It may even be the case, that certain items of evidence previously deemed essential, prove to have little or no effect, once alternative scenarios are tested out with varying values of the item in question.

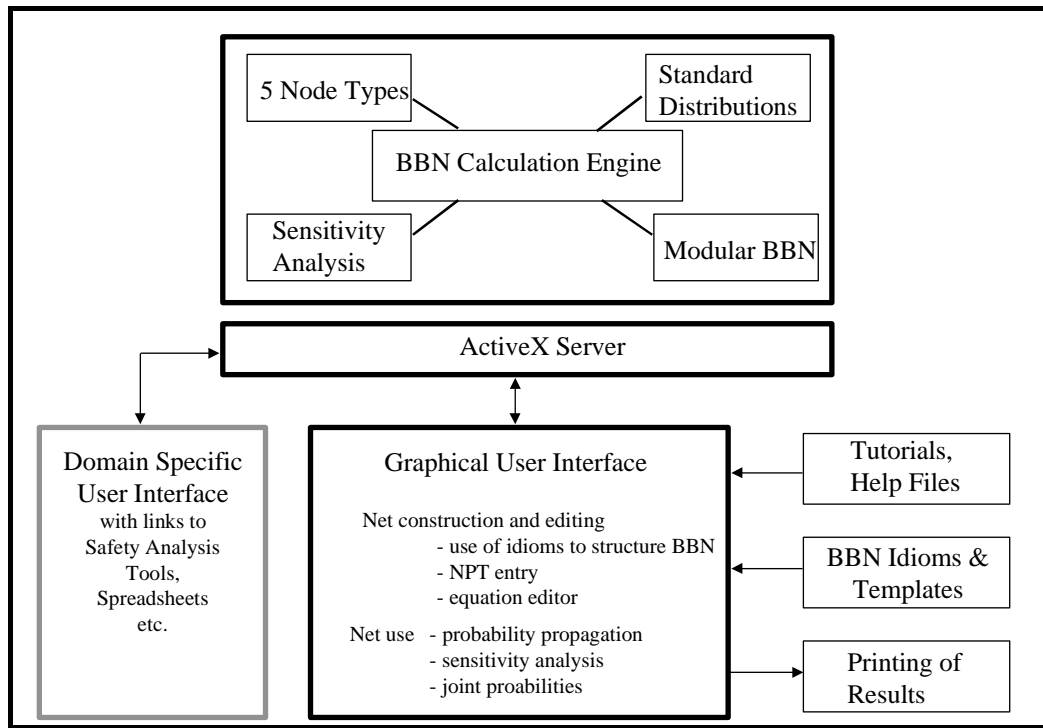
In summary, argument use consists of the following steps:

1. test the argument with hypothetical data and examine whether the reasoning makes sense
2. collect data/evidence/judgements from system to be assessed as soon as possible in the product's development life-cycle
3. input this data into the argument and predict argument and sub-argument goals
4. if predictions are not acceptable then examine argument for improvements in process and/or product
5. perform these steps repeatedly during development making predictions each time and guiding the development process if possible
6. finally make prediction of safety based on all available, current, evidence.

We can also see that process improvement and quality assurance life-cycles are interwoven into the argument use process. In some situations the assessor/regulator may not be able to specify improvements because of legal impediments. Clearly the whole process will require to be tailored to each situation at hand.

## **2.4 Automation of the SERENE Method**

The SERENE Method is supported by a BBN tool, which automates the Bayesian inference calculations. In this section, the main features of the tool are introduced and an overview given of its use within the SERENE Method.



**Figure 2-7: SERENE Toolset Architecture**

### 2.4.1 Tool Features

The use of a BBN tool is indispensable, since for complex nets, manual calculation of the probabilities would be impossible. However, the SERENE Tool (Figure 2-7) is not just a BBN package: overall the SERENE Tool includes:

1. the BBN calculation engine
2. a graphical user interface for the preparing, querying and printing BBNs
3. help files for the tool and the SERENE Method
4. templates, which are sample BBNs which can be adapted or incorporated into a BBN prepared by a user.

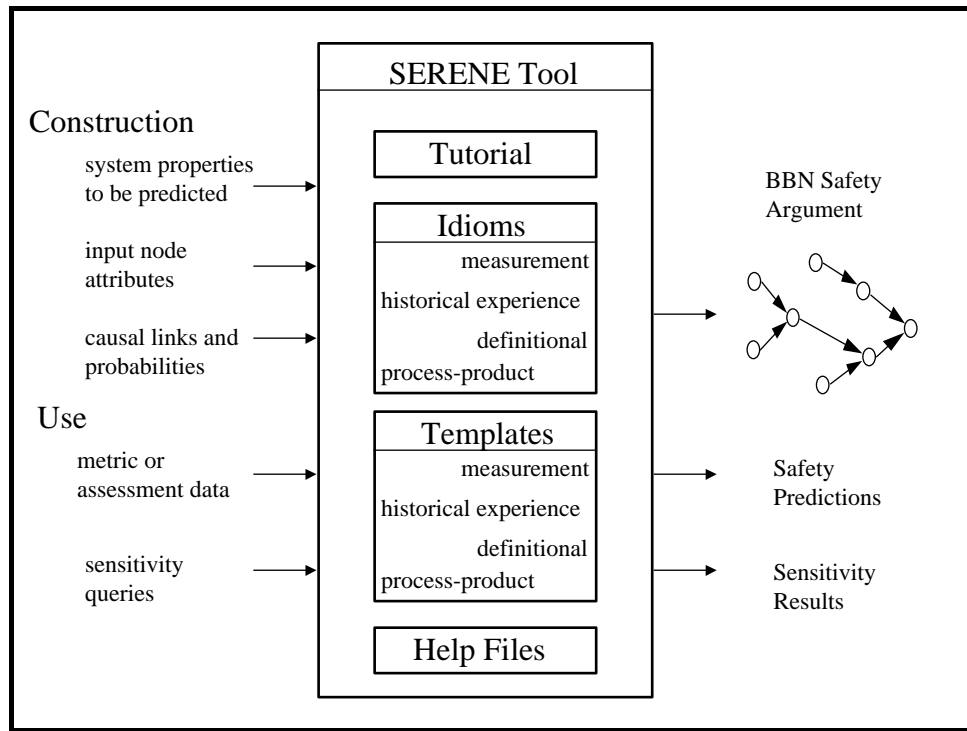
The SERENE Tool provides a number of features intended to be applicable to the SERENE Method and the field of safety engineering. These include:

1. hierarchical structuring of BBNs allowing BBNs to be constructed using the SERENE BBN idioms and the BBN join operation
2. standard distributions for use with both discrete and continuous variables
3. sensitivity analysis.

The capabilities of the tool are exposed using an ActiveX server. This architecture allows domain-specific user interfaces to be created with the maximum reuse of the capabilities already implemented.

### 2.4.2 Use of the SERENE Tool

The SERENE Tool supports the argument construction and argument use activities of the SERENE Method, as shown in Figure 2-8.



**Figure 2-8: Use of the SERENE Tool**

For argument construction, the tool supports the creation of nodes, manipulation of the net structure and entry of NPTs. By using 'abstract nodes' the net can be constructed in a modular way. Each abstract node is expanded into another BBN, with one or more nodes joined at the higher level.

The use of the tool is interactive: data is entered and the probabilities are calculated through the graphical user interface. Probability distributions can be displayed for selected nodes. Sensitivity analysis is also carried out using the graphical user interface. Results can be printed using HTML.

## 2.5 The SERENE partner safety arguments in the SERENE tool

In addition to a number of generic safety arguments that have been built using the SERENE method (and which are described in Chapter 4) the SERENE tool comes with the BBNs that represent the safety argument approach of each of the partners who are involved in safety assessment. In this section we provide a summary of the safety argument BBNs of ERA, EdF and TUV. Full details of these BBNs are provided in the appendices.

### 2.5.1 Hazard-based PES Safety Argument (ERA)

ERA has used the SERENE Method to prepare a safety argument for a programmable electronic system, which is safety-related. The example is based on an assessment carried out by ERA of a system providing a vehicle function: the precise details of the application are commercially confidential. The argument has the form of a prediction of the risk of deploying the system. The main steps leading to this prediction are as follows.

The risk arising from the hazards and possible accidents that have been identified

- The possibility that hazard may not have been identified is also contributes to this risk: this part of the risk is predicted to be small if the rigour of the hazard analysis process is high, taking account of factors likely to affect hazard identification

- The prediction of the risk arising from known hazards is based on the probability of the hazard leading to an accident, the severity of the accident and the frequency with which the hazard arises
- The hazard frequencies are predicted from the failure rates of critical functions within the system. Critical components of the system may include the operator, whose failure is predicted from factors such as the nature of the task, the training and the design process for the user interface.
- Where critical components of the system are redundant, the predicted failure rate takes account of the possibility of common cause failure. The frequency of this is predicted from factors such as the level of integration in the system architecture, the common cause analysis and the level of experience with existing architectures.
- Failure rates for software subsystems are predicted using a error introduction and removal idiom, at each stage of the software life-cycle.

The example illustrates the use of SERENE templates to structure the net into parts of manageable size and complexity. The overall structure of the argument is visible in the net structure at the top-level while the prediction of intermediate properties can be examined independently of their context.

### 2.5.2 The life-cycle based argument (EDF)

This BBN was built by EDF, and corresponds to a situation often encountered in EDF: the assessment of a system for which EDF writes the requirements specification, then monitors the development (which is conducted by an external supplier), and finally validates the system.

The BBN was built by a group of experts in assessment of such systems, who took as an example the assessment of a part of the instrumentation and control of a nuclear power plant. The BBN is relevant for systems belonging to an intermediate safety integrity level.

The BBN presents an overview of products and processes assessment at various stages of the system life-cycle, in order to ensure / assess the functional safety of the system delivered by the supplier, and how one can integrate the view points and opinions of various technical experts. It focuses only on functional safety. In particular, it does not address:

- hardware issues (random failures, resistance to aggressive environment: radiation, high temperature, vibrations...),
- human factor issues,
- global issues encompassing other systems & equipment and the overall design of the power plant and of its process,
- modification of systems already in operation,
- how technical features of the I&C system are assessed.

These limitations are justified by the need to limit the size and the complexity of the BBN. Even with these simplifications, the whole model includes more than a hundred nodes, and has a rather complex structure. Its complete description is given in the appendix.

## 2.6 Benefits of the SERENE Method

Existing safety standards are little more than a collection of process-based techniques that are assumed by a small group of experts to be 'best practice'. The expectation is that system safety will be improved by adopting each of the recommended process activities. Although many of these beliefs have not yet been confirmed empirically, we argue that there are still practical benefits to be gained from making the beliefs explicit.

The approach to safety arguments in SERENE project is to create a model of the beliefs connecting each step of the safety development process to the overall safety of the system. The benefits of this approach include:

1. improved communication of the safety argument, with a clearer rationale for each safety measure
2. greater focus on the properties which lead to safety; this is possible since the properties and the contribution of each to safety are made explicit
3. a basis for empirical validation of the beliefs of safety experts and of the rationale for existing standards.

The benefits of the use of a BBN representation of a safety argument include:

1. the uncertainty associated with the causes of safety can be included in the model and the uncertainty about the overall system safety is explicit
2. the safety achieved can be quantified, allowing alternative safety strategies to be compared.

*This page is intentionally left blank*



### 3 Constructing SERENE Safety Arguments Using Idioms

The purpose of this chapter is to show how BBNs can be used to construct a safety argument for a complex system. The focus is on identifying, reusing and combining BBN graph structures called *idioms*, to help build the BBN topology. The chapter is structured as follows:

- Section 3.1 provides a rationale for using idioms
- Section 3.2 provides some important definitions
- Section 3.3 specifies the five idioms along with examples of their instantiation
- Section 3.4 provides guidelines for choosing the right idiom
- Section 3.5 explains how to join idioms to build complex safety arguments
- Section 3.6 describes the steps involved in building and using a safety argument. We provide a complete example from start to finish.

#### 3.1 The rationale for using idioms

*Idioms* are commonly occurring patterns within networks that can be reused by other safety case developers. Idioms act as building blocks that can be joined together to form a safety argument. Safety argument developers can choose which idioms best fit the line of argument and prediction they want to make. This offers a number of advantages.

1. Emphasises reuse. Effort is thereby reduced.
2. Reuse in this way also maintains the integrity of the underlying causal reasoning required by the BBN formalism.
3. By reusing the idioms we gain the advantage of being able to identify objects that should be more cohesive and self-contained than objects that have been created without some underlying method.
4. Makes building the safety argument easier. You don't have to be an expert in BBNs.
5. Enhances abstraction. Readability is improved.
6. Promotes standardisation.

Simply put, it should help prevent safety engineers and others making mistakes in their arguments.

There is no 'proven' set of evidence that everyone would agree should be represented in a safety argument nor is there agreement over what power that evidence should play in any prediction of system safety or reliability. Therefore the view of the SERENE project with respect to what *content* should be represented in a BBN is pluralistic. We cannot prescribe that specific activities take place, like the use of formal methods or complexity analysis. Nor can we say that the consequences of particular development activities should lead to higher or lower levels of safety than would occur with others. What we can do however is prescribe useful causal structures for how evidence should be combined which reflects how the real world seems to operate.

The examples used in this chapter are all taken from one example net. The emphasis is on topologies and not on the probability tables. The examples used in the following chapter, however, come from different, industrially inspired nets, and show how probability numbers make the idioms useful in practice.

#### 3.2 Some important definitions

In order to characterise and discuss the idioms, we need to use terms such as product, process, resource, entity, and attribute.

In accordance with Fenton (1996) we describe:

1. *Entities* as the things or events of interest and
2. *Attributes* as the properties of entities.

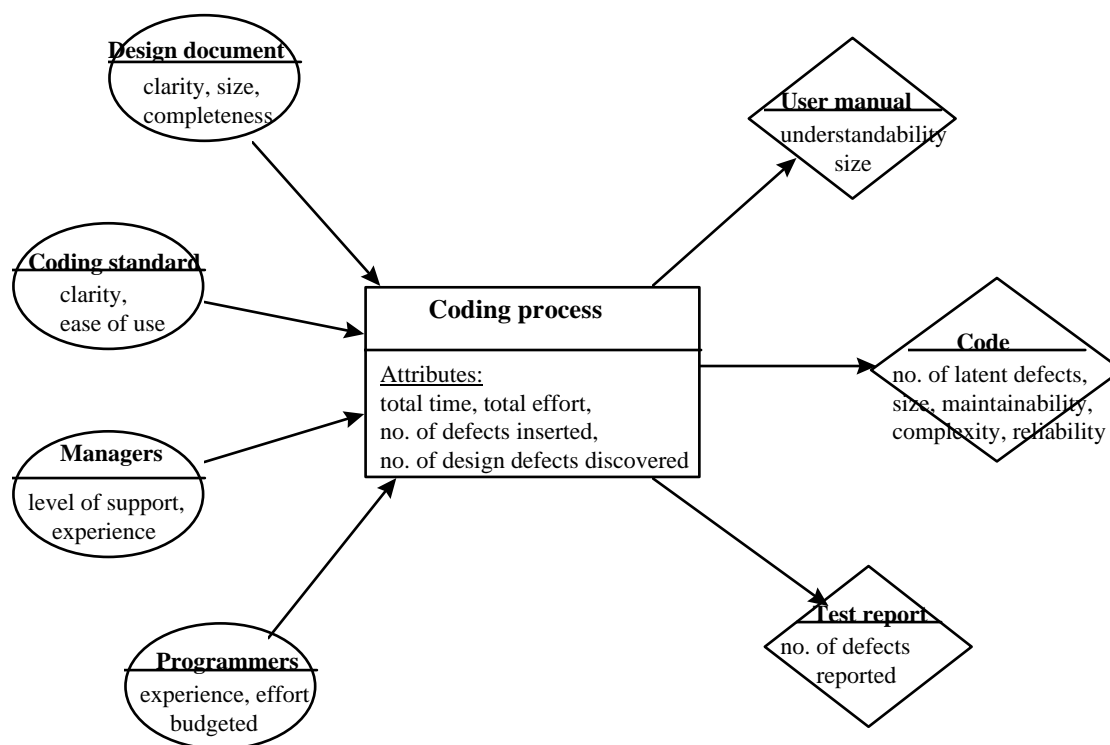
Generally we are interested in observing/measuring attributes of entities. In SERENE the entities we are interested in are:

1. *Processes*, which are activities carried out by human or machine
2. *Resources*, which are the miscellany of items (including people and machines) considered as inputs to the process and
3. *Products*, which are artefacts that are outputs from processes.

Note: When a product is an input to a process we regard it as a resource.

Figure 3-1 provides an example of this classification and the relationships between entities. The circles represent resources. The box represents the programmer coding process and the diamonds represent output products. Underneath each entity we have listed some attributes that we might be interested in observing/measuring.

Note: some attributes are *internal* (such as size of code, effort budgeted for programmers) in the sense that we can measure them without reference to any other entity, while some attributes are *external* (such as reliability of code) in the sense that we cannot measure them without reference to some other entity.



**Figure 3-1: Example of Process, Product and Resource Entity Relations.**

### 3.3 Idiom Specifications

The following five generic idioms have been identified during the SERENE project:

1. **Definitional/Synthesis idiom** – models the deterministic or uncertain definitions between variables; also models the synthesis or combination of many nodes into one node for the purpose of organising the BBN

2. **Process-product idiom** – models the uncertainty of a process with observable consequences
3. **Measurement idiom** – models the uncertainty about the accuracy of a measurement process or instrument
4. **Induction idiom** – models the uncertainty related to inductive reasoning (including historical experience) based on populations of similar or exchangeable members
5. **Prediction/Reconciliation idiom** – models the reconciliation of results from competing measurement or prediction systems.

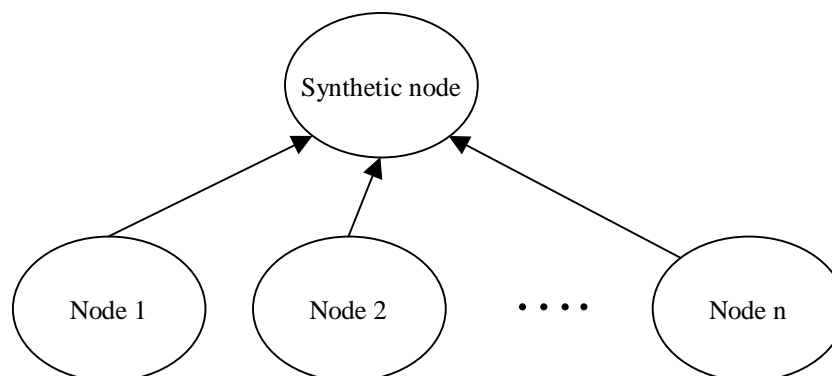
Each of these is explained in detail below.

The generic idioms are not BBNs as such since the nodes have only generic labels and there are no probability tables associated with the nodes. *Idiom instantiations* are idioms made concrete for a particular problem, with meaningful labels. Once probability values have been assigned then they can be included in a BBN. The SERENE tool comes supplied with a number of pre-defined idiom instantiations. Like any reusable BBN fragment within the SERENE tool these idiom instantiations are referred to as *templates*.

We do not claim that the idioms identified here form an exhaustive list of all of the types of reasoning that can be applied in all domains. However, all reasonable BBNs that have been built in the safety assessment domain, as well as systems and software engineering, do appear to be built up from this small set of idioms.

### 3.3.1 The Definitional/Synthesis Idiom

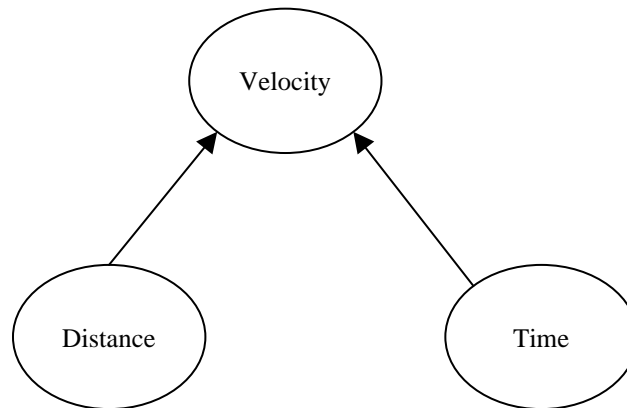
Although BBNs are used primarily to model causal relationships between variables, the most commonly occurring class of BBN fragments is not causal at all.



**Figure 3-2: Definitional/synthesis idiom**

The definitional/synthesis idiom, shown in Figure 3-2, models this class of BBN fragments and covers each of the following cases where the synthetic node is determined by the values of its parent using some combination rule.

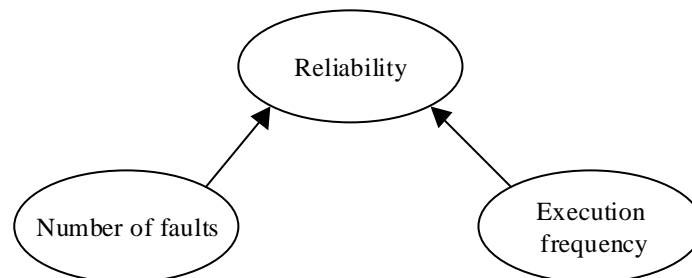
*Case 1. Definitional relationship between variables.* In this case the synthetic node is *defined* in terms of the nodes: node1, node2, ..., node *n* (where *n* can be any integer). This does not involve uncertain inference about one thing based on knowledge of another. For example, velocity *V* of a moving object is defined in terms of distance travelled *D* and time *T* by the relationship,  $V = D/T$ .



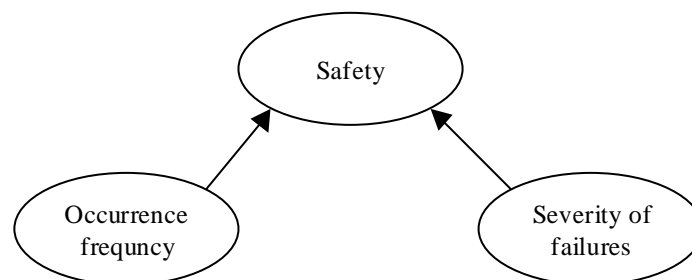
**Figure 3-3: Instantiation of definitional/synthesis idiom (velocity)**

Although  $D$  and  $T$  alone could be represented in a BBN (and would give us all of the information we might need about  $V$ ), it is useful to represent  $V$  in the BBN along with  $D$  and  $T$  (as shown in Figure 3-3). For example, we might be interested in other nodes conditioned on  $V$ . Clearly synthetic nodes, representing definitional relations, could be specified as deterministic functions.

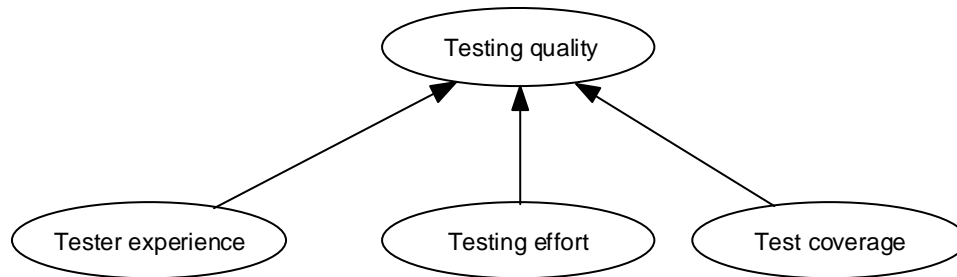
Instantiations of this idiom that arose in safety arguments are shown in Figure 3-4, Figure 3-5 and Figure 3-6. In Figure 3-4 system *reliability* defined as a function of the number of faults and the execution frequency. In Figure 3-5 *safety* is defined in terms of occurrence frequency of failures and the severity of failures. In Figure 3-6 *testing accuracy* is defined in terms of tester experience, testing effort, and test coverage.



**Figure 3-4: Instantiation of definitional/synthesis idiom (reliability)**



**Figure 3-5: Instantiation of definitional/synthesis idiom (safety)**

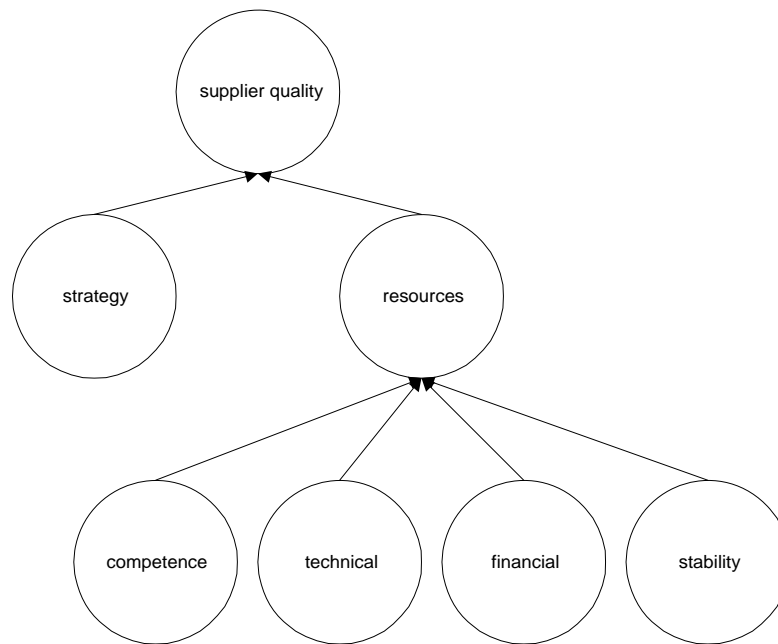


**Figure 3-6: Instantiation of definitional/synthesis idiom (testing quality)**

*Case 2: Combining different nodes together to reduce effects of combinatorial explosion (divorcing):* We can condition some node of interest on the synthetic node, rather than on the parents of the synthetic node itself, in order to ease probability elicitation. If the synthetic node is a deterministic function of its parents it then acts as a parameter on its child node, thus reducing the overhead of knowledge elicitation. For instance if we were interested in  $p(A | B, C, D)$  where  $A, B, C$  each have 5 states we will have to produce 54 separate probability numbers for  $A$ 's conditional probability table. If however we create the synthetic node,  $S$ , where  $p(A | S)$  and  $p(S | B, C, D) = B+C+D$  we require only 52 for the conditional probability table  $p(A | S)$ . This technique of cutting down the combinatorial space using synthetic nodes has been called divorcing by [Jensen]. Here the synthetic node,  $S$ , divorces the parents  $B, C$  and  $D$  from  $A$ .

*Case 3: Follow organisational principles used by expert to organise nodes using a hierarchy:* One of the first issues we face when building BBNs is whether we can combine variables using some hierarchical structure using a valid organising principle. There are a number of practical reasons for wanting to do this. Firstly we might view some nodes as being of the same type or having the same sort and degree of influence on some other node and might then wish to combine these. Also, to cut down on the demands of eliciting large volumes of probability numbers we might simply combine many nodes into one node and condition the node of interest on this new synthetic node.

We can specify a hierarchy of synthetic nodes to model the sorts of informal organising principles experts often use to organise variables. These hierarchies might model the attributes and sub-attributes of a complex variable. For example the variable 'testing quality' in Figure 3-4 is composed from {tester experience, testing effort, test coverage}, but tester experience might itself be complex and composed of {tester qualifications, years testing}. Another example is shown in Figure 3-7. Here the variable *supplier quality* is defined by an expert wishing to evaluate the quality of suppliers providing commercial software products to meet a specification. The expert defines supplier quality as being composed of two sub-attributes – the strategy used by the supplier to build the product and the resources available for this process. Resources is a complex attribute and is then decomposed into competence, technical, financial and stability sub-attributes.



**Figure 3-7: Idiom instantiation for a definitional hierarchy**

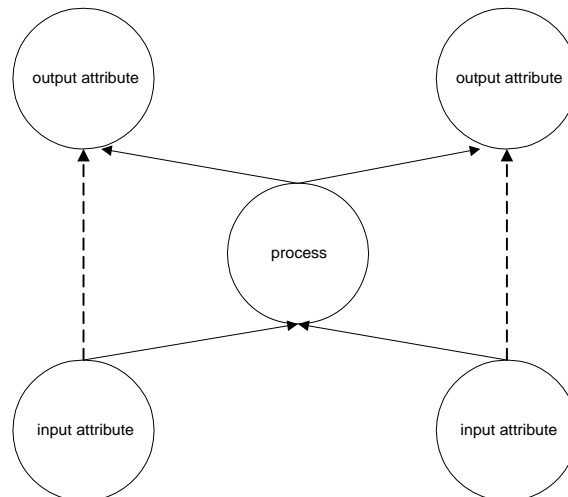
The arc directions in the synthesis idiom do not indicate causality. This would not make sense. Rather the direction indicates the direction in which a sub-attribute defines an attribute, in combination with other sub-attributes.

### 3.3.2 The Process-Product Idiom

The process-product idiom is used to model any case where a process is transforming inputs into outputs. The process itself can involve transformation of an existing product into a changed product or taking ‘raw’ input resources to produce a new output product. We use the process-product idiom to model situations where we wish to predict the outputs produced by some product from knowledge of the inputs that went into that process.

A process can be mechanical or intellectual in nature. A production line producing cars from parts, according to some production plan, is a process. Producing software from a specification using a team of programmers is a process that produces an output in the form of a software program. In both cases we might wish to evaluate some attribute of the inputs and the outputs in order to predict one from the other. For example, the number of faults in the software program will be dependent on the quality of the specification document and the quality of the programmers.

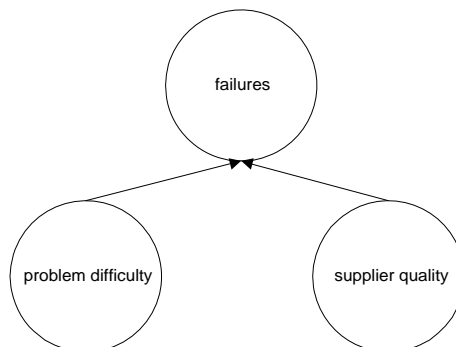
Figure 3-8 shows the process-product idiom. The plain arrows model causality whereby the inputs cause some change in the outputs via the process. The dotted arrow denotes alternative specifications for the input-output relations, bypassing the process node. This direct connection might be justified when it is easier to infer one from the other. Hence the process node can be viewed as a synthetic node used to organise the configuration of input and output nodes.



**Figure 3-8: The process-product idiom**

The directions for the arcs in Figure 3-8 are straightforward because they follow the causal direction implied by production or transformation.

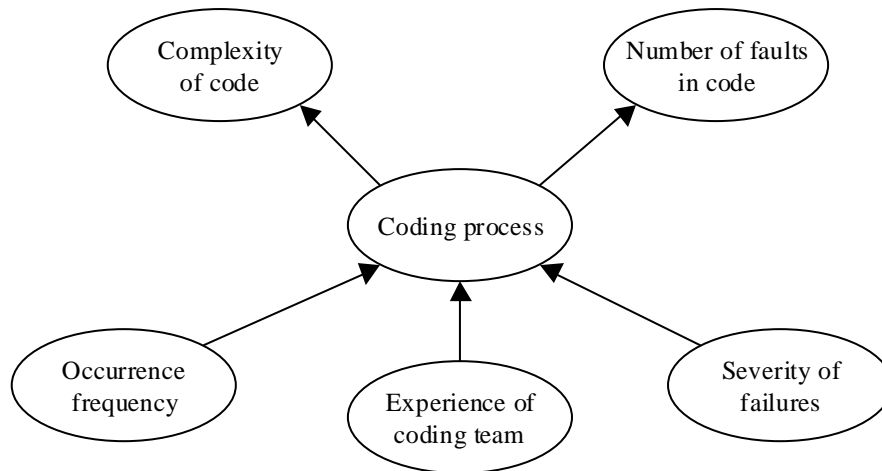
A simple instantiation of the process-product idiom is shown in Figure 3-9. Here we are predicting the frequency of software failures based on knowledge about problem difficulty and supplier quality (where supplier quality was defined in Section 3.3.1).



**Figure 3-9: Process-product idiom instantiation (software failures)**

Here the process involves the supplier producing the product (in this case the process node itself is omitted). A good quality supplier will be more likely to produce a failure-free piece of software than a poor quality supplier. However the more difficult the problem to be solved the more likely that faults will be introduced and the software may fail.

An instantiation of the process-product idiom in which the process node is included is shown in Figure 3-10

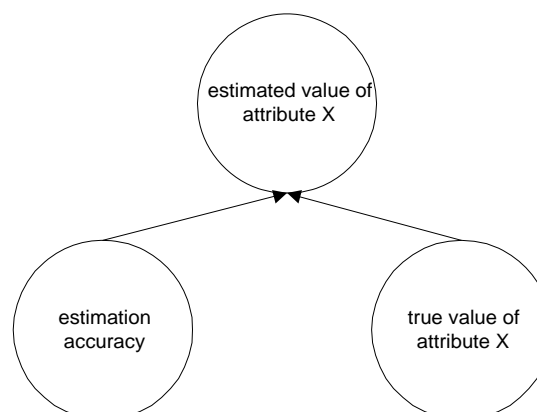


**Figure 3-10: Process-product idiom instantiation (Coding Process)**

### 3.3.3 Measurement Idiom

We can use BBNs to reason about the uncertainty we may have about our own judgements, those of others, or the accuracy of the instruments we use to make measurements. The measurement idiom represents uncertainties we have about the process of observation. By observation we mean the act of determining the true attribute, state or characteristic of some entity. The difference between this idiom and the process-product idiom is that here one node is an estimate of the other rather than the representing attributes of two different entities.

Figure 3-11 shows the measurement idiom. The arc directions here can be interpreted in a straightforward way. The true value must exist before the estimate in order for the act of measurement to take place. Next the measurement instrument interacts (physically or functionally) with the entity under evaluation and produces some result. This result can be more or less accurate depending on intervening circumstances and biases.

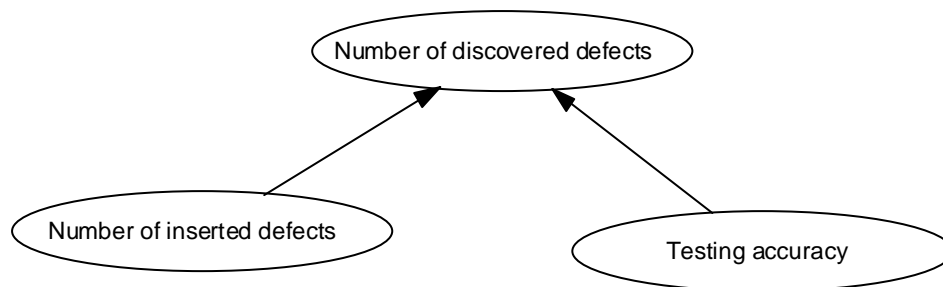


**Figure 3-11: Measurement Idiom**

The true value of attribute X is measured by a measurement instrument (person or machine) with a known estimation accuracy, the result of which is an estimated value of attribute X. Within the node estimation accuracy we could model different types of inaccuracies: expectation biases and over- and under-confidence biases.

A classic instantiation of the measurement idiom is the testing example that we already saw in Figure 2-5. We show this again in Figure 3-12.





**Figure 3-12: Measurement idiom instantiation (testing)**

When we are testing a product to find defects, we use the number of discovered defects as a *surrogate* for the *true* measure that we want, namely the number of inserted defects. In fact the measured number is dependent on the testing accuracy. Positive and encouraging test results could be explained by a combination of two things:

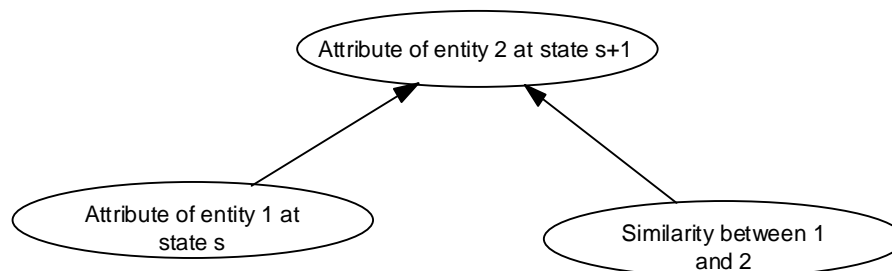
- Low number of inserted defects resulting in low number of discovered defects, or
- Very poor quality testing resulting in low number of defects detected during testing.

By using the measurement idiom we can explain away false positive results.

The measurement idiom is not intended to model a sequence of repeated experiments in order to infer the true state. Neither should it be used to model the inferences one might make from other, perhaps similar, entities. The induction idiom below is more appropriate in these two cases.

### 3.3.4 Induction idiom

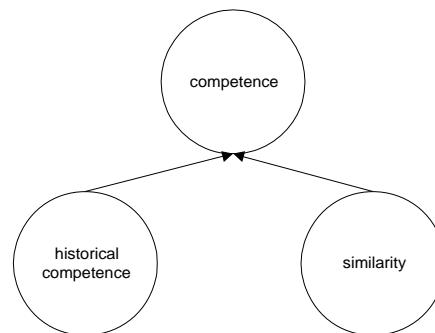
The induction idiom (Figure 3-13) involves modelling the process of statistical inference from a series of similar entities to infer something about a future entity with a similar attribute. The assumption here is that the entities are homogeneous in some way and can be considered as being drawn from a population where the members are exchangeable.



**Figure 3-13: Induction idiom**

None of the arcs in the induction idiom are causal.

An instantiation of the induction idiom is shown in Figure 3-14. When performing system testing we might wish to evaluate the competence of the testing organisation in order to assess the quality of the testing that is likely to be performed. The historical track record of the organisation might form a useful database of results to infer the true competence of the organisation. This can be summarised in the node historical competence, which can be used to infer the current level of competence.

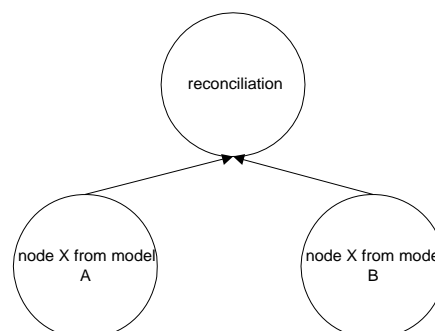


**Figure 3-14: Induction idiom instantiation (testing competence)**

However, we might feel that the historical track record has been performed on only a sub-set of systems of interest to us in the future. For example, the previous testing exercises were done for non-critical applications in the commercial sector rather than safety critical applications. Thus, the assessor might wish to adjust the track record according to the similarity of the track record to the current case.

### 3.3.5 Prediction/Reconciliation Idiom

The final idiom is the prediction/reconciliation idiom. The objective of this idiom is to reconcile independent sources of evidence about a single attribute X of a single entity, where these sources of evidence have been produced by different measurement or prediction methods. The idiom is shown in Figure 3-15.



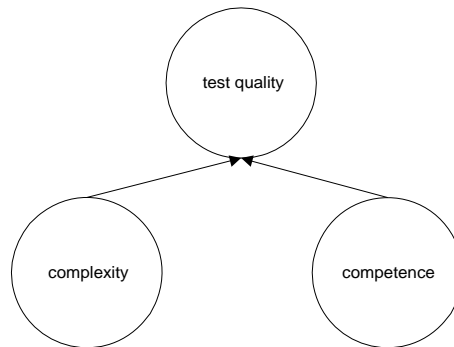
**Figure 3-15: Prediction/Reconciliation idiom**

The node of interest, node X, is estimated by two independent procedures, model A and model B. The reconciliation node is a Boolean node. When reconciliation is 'true' the value of X from model A is equal to the value of X from model B. Thus, by setting the reconciliation node to true we allow the flow of evidence from model B to model A. There is, however, one proviso - should both sets of evidence prove contradictory then the inferences cannot be reconciled.

The following example of a prediction/reconciliation idiom is typical of many we have come across in safety/reliability assessment. We have two models to estimate the quality of the testing performed on a piece of software:

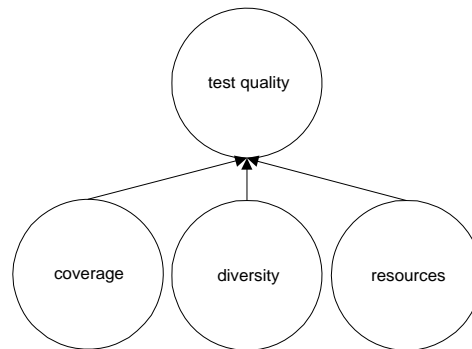
1. Prediction from known causal factors (represented by a process-product idiom instantiation);
2. Inference from sub-attributes of testing quality which when observed give a partial observation of testing quality (represented by a definitional/synthesis idiom instantiation).

The relevant process product idiom instantiation here is shown in Figure 3-16 (a complex product will be less easy to test and incompetent testers will be less likely to perform good testing).



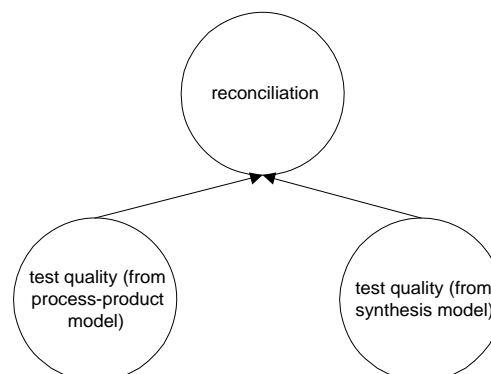
**Figure 3-16: A process-product idiom instantiation for test quality**

The definitional/synthesis idiom instantiation in Figure 3-17 shows how test quality is defined from three sub-attributes *coverage*, *diversity* and *resources*.



**Figure 3-17: A definitional/synthesis idiom instantiation for test quality**

We now have two models for inferring the state of test quality; one based on cause effect reasoning about the testing process and one based on attributes about the testing itself. The test quality from the definitional/synthesis model is conditionally dependent on the test quality process-product model, as shown in the instantiation of the prediction/reconciliation idiom in Figure 3-18.

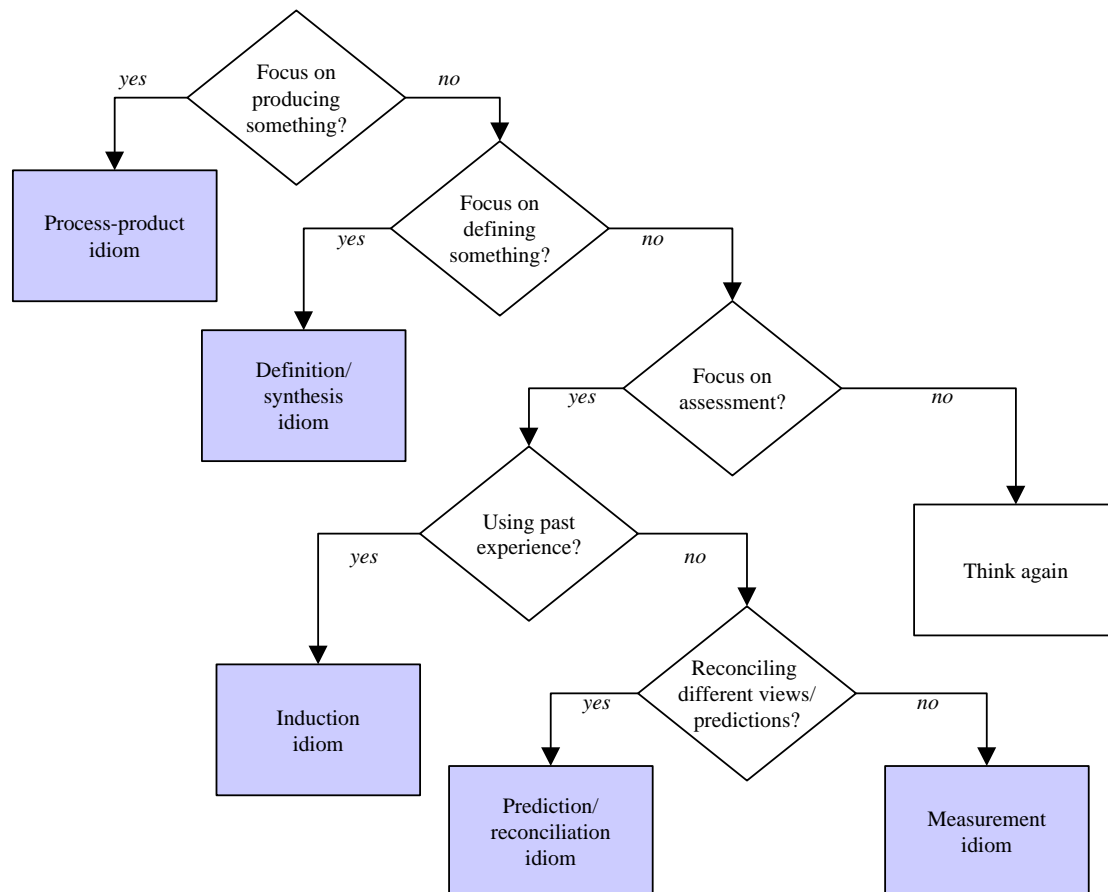


**Figure 3-18: Prediction/Reconciliation idiom instantiation for test quality**

### 3.4 Choosing the right idiom in the SERENE method,

Building safety arguments from idioms lies at the heart of the SERENE method. In the previous section we explained the individual idioms in some detail. Here we summarise a sequence of actions that should help users identify the ‘right’ set of idioms if they are building a safety argument from scratch:

1. Make a list of the entities and their attributes, which you believe to be of relevance to your safety argument.
2. Consider how the entities and attributes relate to one another. This should lead to subsets of entities and attributes grouped together.
3. Examine these subsets in terms of the flowchart (Figure 3-19) checklist in order to determine which idiom is possibly being represented:



**Figure 3-19: Choosing the right idiom**

Note: Some nodes and relations in the idioms may not be relevant to you. The method encourages you to consider every node in an idiom but if it really is not required then it may be left out (for example, the process node in the process-product idiom). However, always consider the effects of removing nodes on the probabilities.

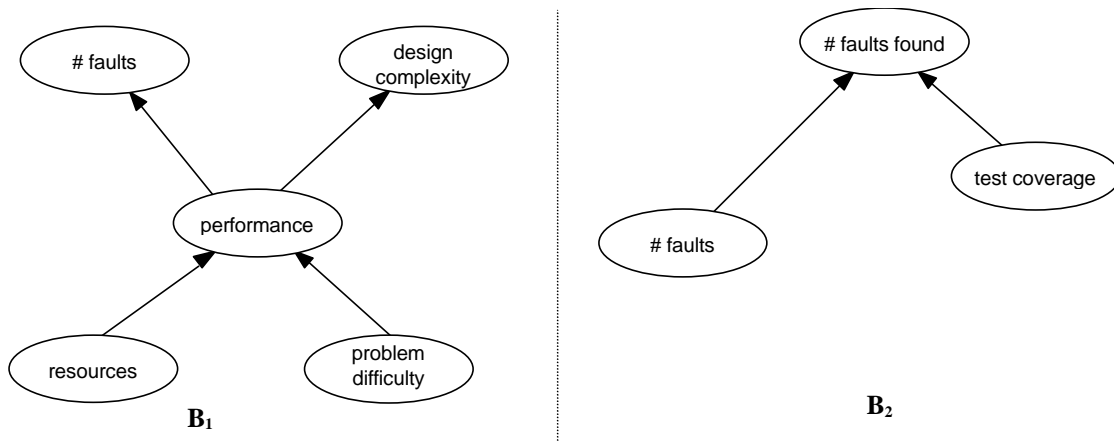
A complete example of this process is provided in Section 3.6.4.

### 3.5 Joining Idioms to make Arguments

If we are going to build complex BBNs from simpler structures and arguments then we need to define a formal, but intuitively reasonable BBN join operation. The join operation defined in this section, which is implemented in the SERENE tool, enables us to build complex arguments from idioms.

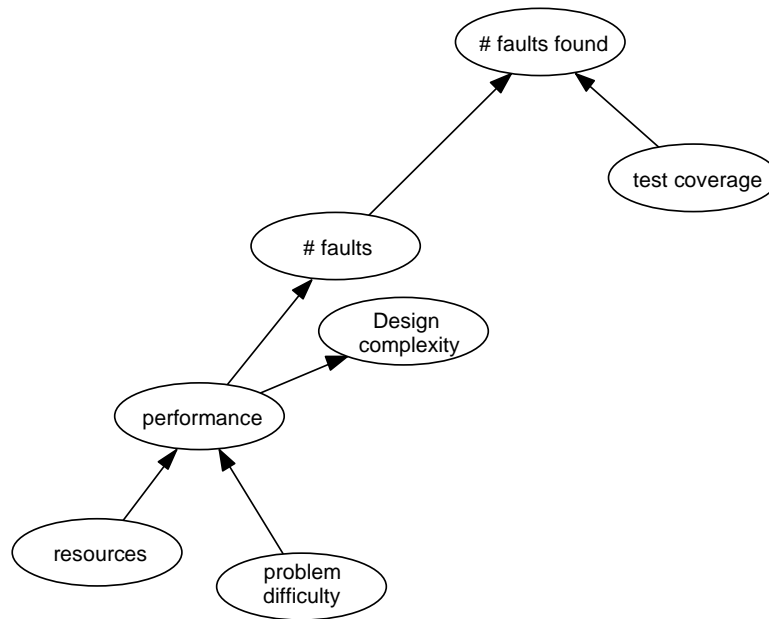
### 3.5.1 The problem

The general problem we have is that we have two BBNs, say  $B_1$  and  $B_2$  which have one or more common nodes. We want to be able to construct a new BBN, say  $B$ , which is somehow built by joining  $B_1$  and  $B_2$  at the common nodes. Consider, for example, the BBNs shown in Figure 3-20 (these correspond respectively to examples of process-product and measurement idioms as described in the previous section.) The common node is “# faults”.



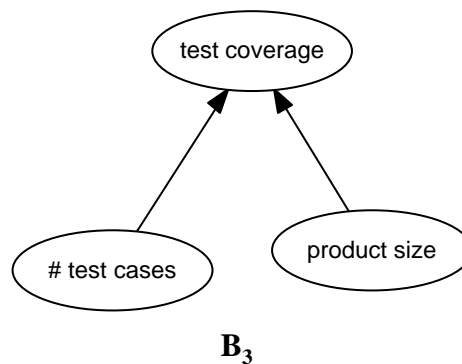
**Figure 3-20: Two BBNs with Common Node “# faults”**

It seems perfectly reasonable that we should be able to construct a new, joined, BBN as shown below in Figure 3-21:



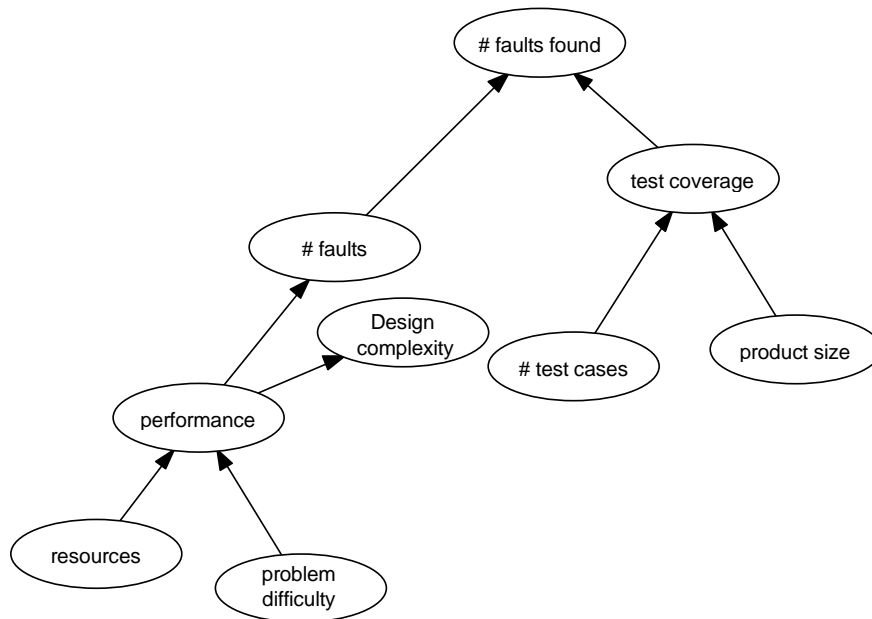
**Figure 3-21: The BBNs of Figure 3-20 Joined at Node “# faults”.**

Similarly, Figure 3-22 is an instantiation of the definitional/synthesis idiom with a node ‘test coverage’ in common with  $B_2$ .



**Figure 3-22: Another BBN to join**

It seems reasonable to be able to join this BBN to the one in Figure 3-21 to get the BBN shown in Figure 3-23.



**Figure 3-23: The BBNs of Figure 3-21 and Figure 3-22 joined at Node “test coverage”.**

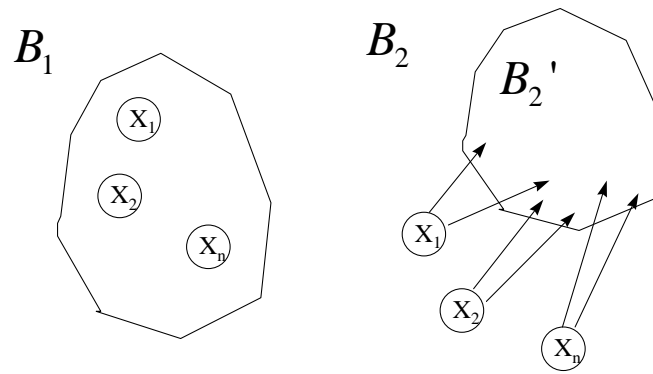
Of course a BBN consists not just of a graph topology but also an underlying set of NPTs. Thus our definition of the new ‘joined’ BBN should also specify the new NPTs. In this particular example, it seems reasonable that, with the exception of the common node, all nodes simply inherit their NPTs from their respective BBNs. For the common node we choose the NPT from  $B_1$  where, unlike  $B_2$  it is not a root node - we lose nothing in ‘throwing away’ the prior probabilities assigned to the common node when it is a root node.

What made the example straightforward was that the common node was parentless in  $B_2$ . If, for example, there was an arc from “# test case” to “# faults” in  $B_2$  (that is, in  $B_2$  “# faults” was not parentless) then the problem would come when we tried to define the NPT for “# faults” in the joined net. We cannot simply ‘inherit’ the NPT from  $B_1$  since this assumes a single parent (“performance”) rather than the two that we would now have.

### 3.5.2 The formal definition of the join operation

Motivated by the example we now define formally the join operation.

Suppose  $B_1$  and  $B_2$  are BBNs which have common nodes  $X_1, \dots, X_n$ . Suppose further that  $X_1, \dots, X_n$  are parentless in  $B_2$ , so that schematically  $B_1$  and  $B_2$  may be represented as shown in Figure 3-24.



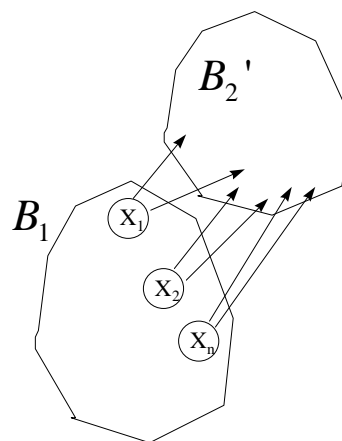
**Figure 3-24: Two BBNs Ready for Joining on Node  $X_1, \dots, X_n$**

Then we define the join of  $B_1$  and  $B_2$  on nodes  $X_1, \dots, X_n$  as the BBN  $B$  as follows (see the figure below):

- The graph of  $B$  is defined as:
  - $\text{Nodes}(B) = \{\text{Nodes}(B_1)\} \cup \{\text{Nodes}(B_2')\}$
  - $\text{Arcs}(B) = \{\text{Arcs}(B_1)\} \cup \{\text{Arcs}(B_2)\}$
- For each node  $Y$  of  $B$ , the NPTs are defined as follows (by convention, let us denote the NPT of a node  $Y$  in a BBN  $Z$  as  $\text{NPT}_Z(Y)$ ):

$$\text{NPT}_B(Y) = \begin{cases} \text{NPT}_{B_1}(Y) & \text{if } Y \in B_1 \\ \text{NPT}_{B_2'}(Y) & \text{if } Y \in B_2' \end{cases}$$

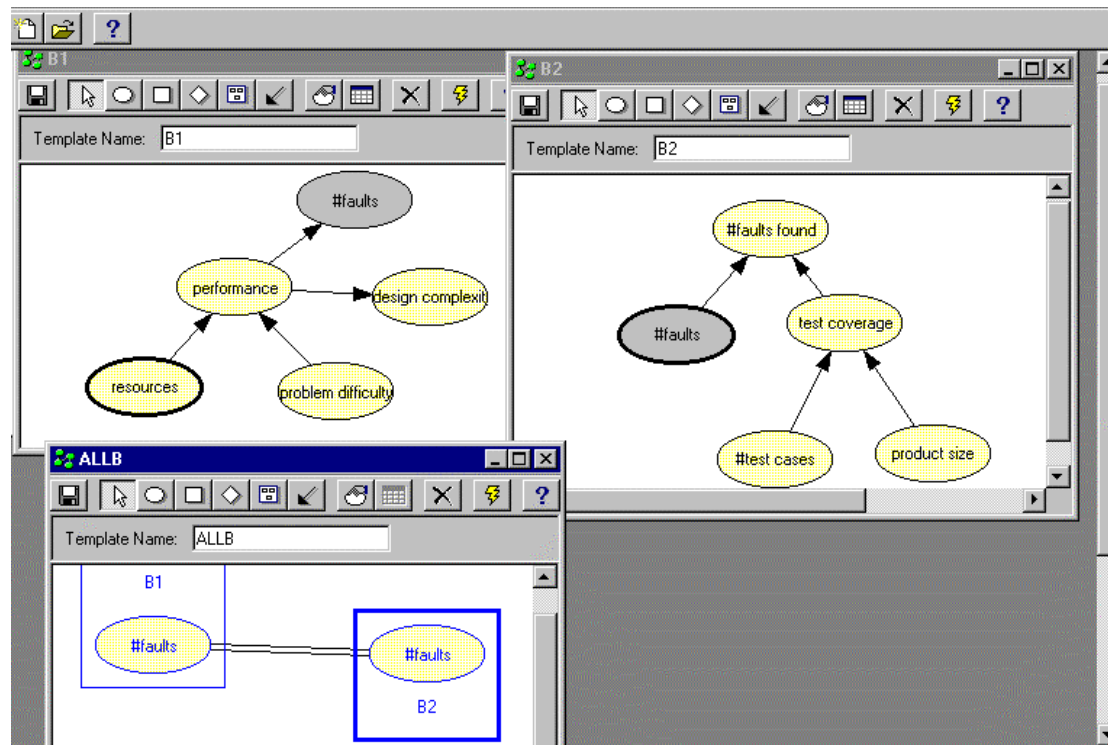
Schematically, the join operation is shown in Figure 3-25.



**Figure 3-25: Schematic of Join Operation (the BBN  $B$  that results from joining the BBNs in Figure 3-24)**

Figure 3-26 shows this join operation as it looks in the SERENE tool. Nets  $B_1$  and  $B_2$  are shown as separate templates  $B1$  and  $B2$ . Then another template has been created called ALLB which shows the nets  $B1$  and  $B2$  as *abstract nodes*, joined by the common node #faults.





**Figure 3-26: Joining BBNs in the SERENE Tool**

In Chapters 4 and 5 we will present more detailed examples of using the SERENE tool in this way.

### 3.6 The SERENE method: a complete example

To build and use a safety argument using the SERENE method consists of the following sequence of tasks:

- Task 1: List the key entities (products, process and resources) that are relevant for the problem
- Task 2: Determine the key attributes of the entities that are relevant for the safety argument
- Task 3: Group together related attributes
- Task 4: Determine the appropriate idioms that relates the attributes
- Task 5: Define the NPTs for each node in each idiom
- Task 6: Apply the join operation to build the complete safety argument
- Task 7: Use the safety argument template to enter observations and make predictions

In the example below we are in the situation of a typical regulator. We have to decide whether a system is sufficiently safe to be given a certificate. Although ultimately the decision itself is outside the scope of SERENE (see Appendix E) what we can do is make a prediction of system safety based on the information available. In this case the information comes in the form of some independent testing results and our knowledge of:

- the organisation who built the system
- the organisation who tested it
- the system requirements.

### 3.6.1 Task 1: List the key entities (products, process and resources)

Main product: the system to be assessed

Main processes: the independent testing process and the development process

Main resources: the independent testing team and the system developer.

### 3.6.2 Task 2: Determine the key attributes of the entities that are relevant for the safety argument.

This is shown in Table 3-1.

**Table 3-1: List of relevant entities and their attributes**

Entity type	Entities	Attributes
Product	System to be assessed	<ol style="list-style-type: none"> <li><i>Safety</i> (this is the attribute we really want to predict): In this context ‘higher’ safety levels, mean lower probability of the system failing.</li> <li><i>Operational usage</i> (this is clearly an important attribute because we know that the higher the use of the system the more likely it is to fail)</li> <li><i>Number of latent faults</i> (it is these that could cause failures in operation)</li> <li><i>Complexity of design solution</i> (this will impact on the ability to test it properly)</li> <li><i>Intrinsic complexity of system requirements</i></li> </ol>
Process	The independent testing process	<ol style="list-style-type: none"> <li><i>Number of faults found in test</i></li> <li><i>Accuracy</i></li> </ol>
Resource	The independent testing team	<i>Quality</i>
Process	The development process	<i>Quality</i>
Resource	System developer	<i>Quality</i>

### 3.6.3 Task 3: Group together related attributes

We have done this in Table 3-2. We have also provided a rationale for the choice.

**Table 3-2: Grouping related attributes**

Group	Attributes in group	Rationale for grouping
1	system safety operational usage number of latent faults	We believe that safety level is determined by the number of latent faults and the extent of the operational usage
2	faults found in test accuracy of testing testing process	Number of faults found in testing will be determined by the number of latent faults and the accuracy of the testing
3	accuracy of testing process quality of testing team complexity of design solution	Accuracy will be dependent on the quality of the team and the complexity of the design
4	intrinsic complexity of requirements quality of supplier development process complexity of design solution number of latent faults	The quality of the development process will be inhibited by the quality of the supplier and the intrinsic complexity of the requirements. The complexity of the design solution and number of latent faults are the attributes that result from the development process

### 3.6.4 Task 4: Determine the appropriate idioms that relate the attributes.

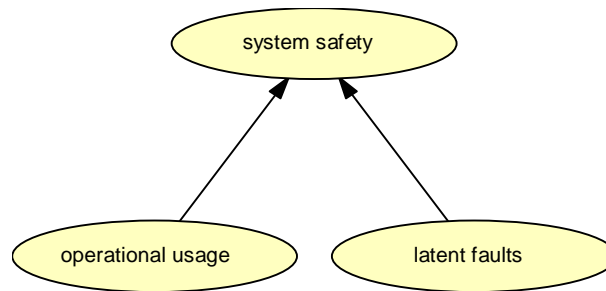
For each group we use the flowchart of Figure 3-19

Group 1

Is the focus here on producing something? No

Is the focus here on defining something? Yes. System safety is really defined in terms of the number of latent faults and the operational usage.

Hence choose **definitional/synthesis idiom**:



Group 2:

Is the focus here on producing something? No

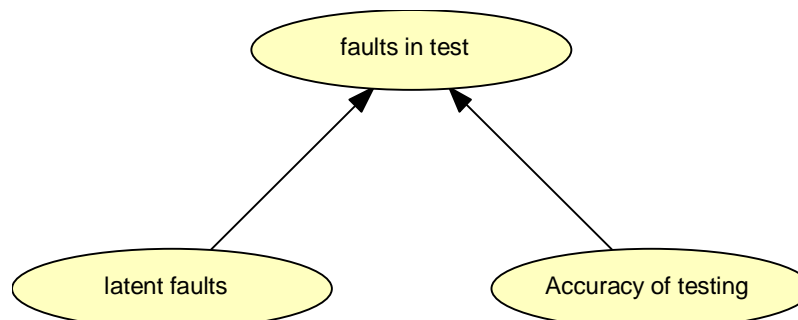
Is the focus here on defining something? No

Is the focus here on assessment? Yes.

Are we using past experience? No

Are we reconciling different views/predictions? No

Hence choose **measurement idiom**:

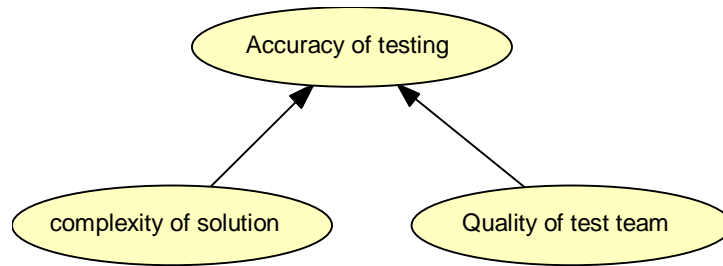


Group 3.

Is the focus here on producing something? No

Is the focus here on defining something? Yes. Accuracy of the testing process is actually defined in terms of quality of testing team and complexity of the design solution.

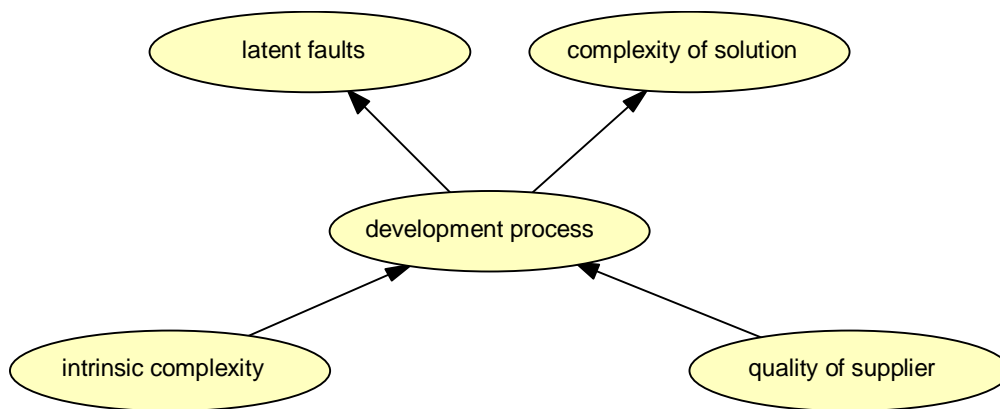
Hence choose **definitional/synthesis idiom**



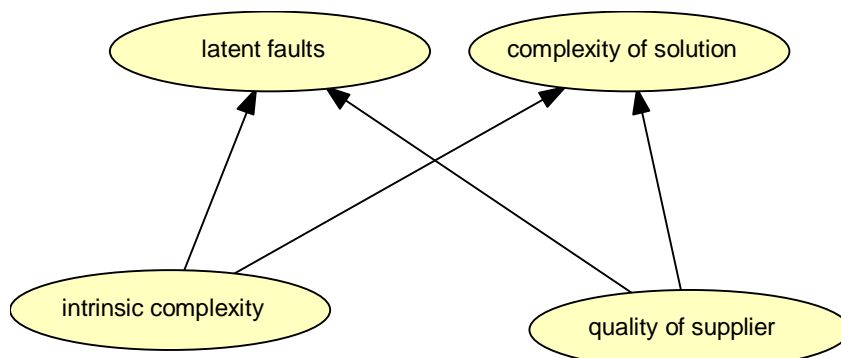
Group 4:

Is the focus here on producing something? Yes – the focus is on producing the system.

Hence choose **process-product idiom**:



However, in this case there is nothing that we can observe about the development process, and so we remove the development process node completely to arrive at the following instantiation of the idiom:



### 3.6.5 Task 5: Define the NPTs for each node in each idiom

Now that we have decided on the idioms we need to define the NPT for each one. To keep this example very simple we have given simple discrete value sets to each of the nodes (e.g. ‘low’, ‘medium’, ‘high’) and we enter the probability values directly, rather than use any functions or distributions (examples of those can be found in Chapters 4 and 5) . Since this particular example


appears as a template in the SERENE tool (*safety\_operational\_usage.snt*) we show just one example table produced as it appears in the tool.

operational us	low			medium			
latent faults	high	medium	low	high	medium	low	high
low	.5	.333333	.1	.65	.333333	.2	.8
medium	.3	.333333	.2	.25	.333333	.3	.15
high	.2	.333333	.7	.1	.333333	.5	.05

**Figure 3-27: NPT for system safety node**

The particular values entered reflect our understanding that the higher the operational use and number of latent faults, the lower will be the system safety level. The left hand column values are the values for system safety.

The other NPTs can be viewed as follows:

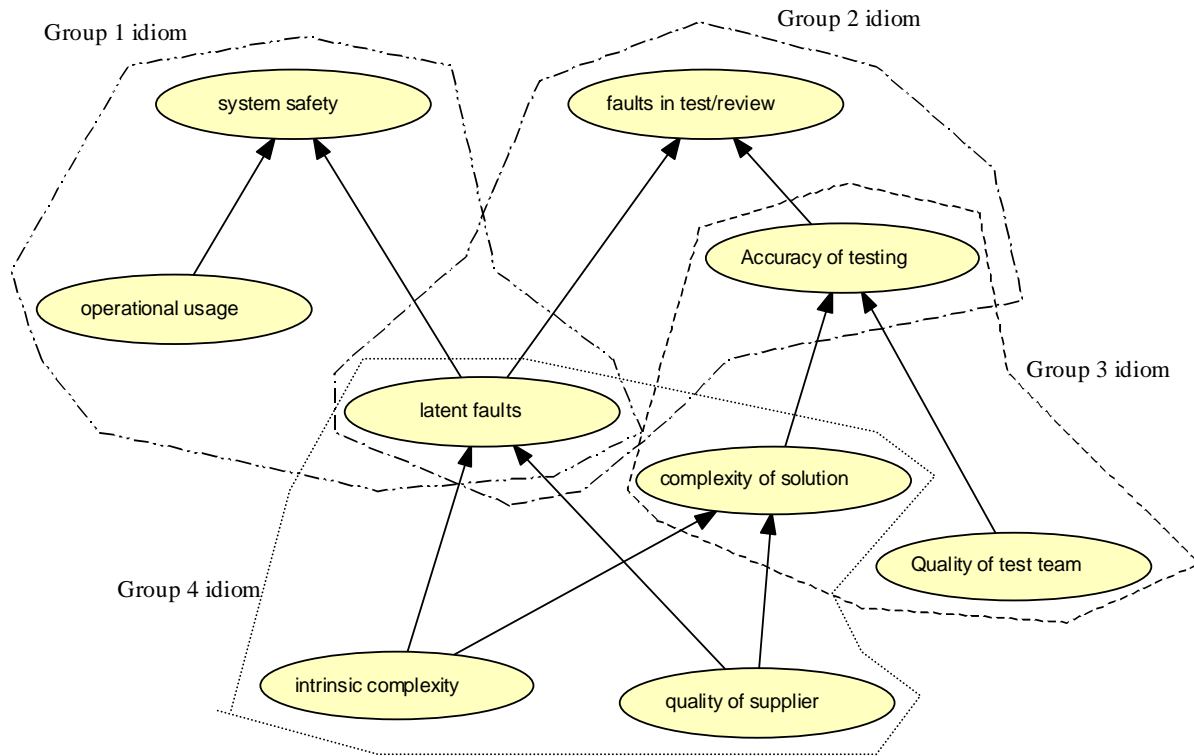
1. open the template *safety\_operational\_usage.snt* using the SERENE tool
2. select (by clicking once) any node whose NPT you wish to inspect. The node outline will become bold
3. click on the NPT icon  in the menu bar.

### 3.6.6 Task 6: Apply the join operation to build the complete safety argument.

This is done as follows:


1. join the idioms from Groups 1 and 2 on the node “latent faults”
2. join the resulting BBN with the idiom from group 3 on the node “testing accuracy”
3. join the resulting BBN with the idiom from group 4 on the nodes “latent faults and complexity of solution”.

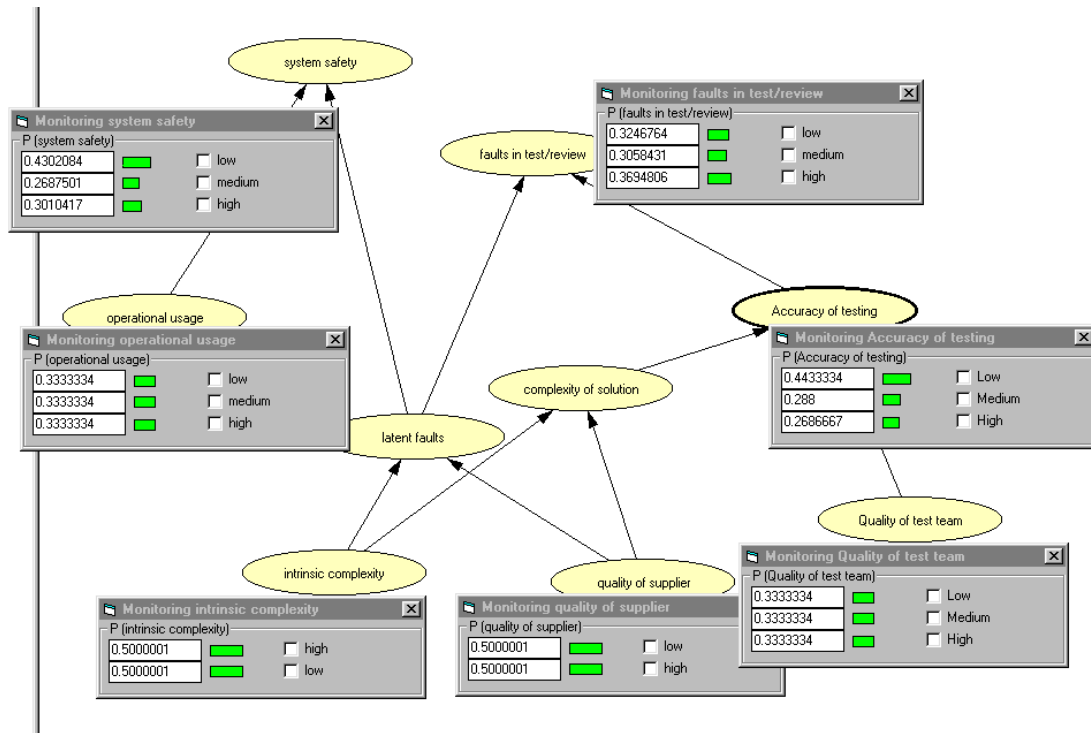
The resulting BBN template is shown in Figure 3-28 (we have shown schematically the underlying idioms and how they are joined).




**Figure 3-28: Complete safety argument template**

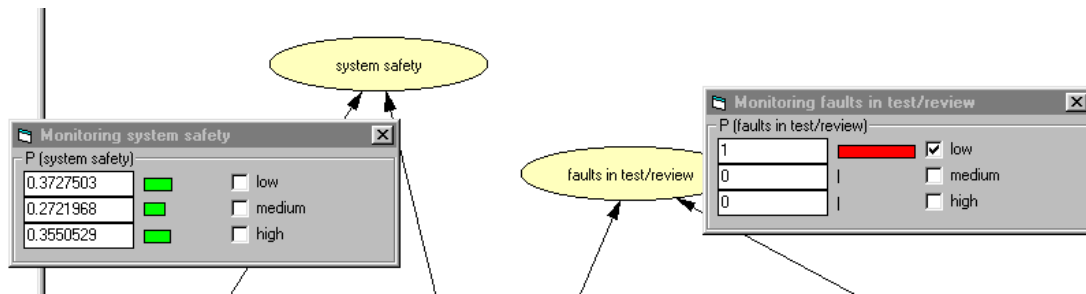
### 3.6.7 Task 7: Enter observations and make predictions with the safety argument template

With the template open in the SERENE tool click on the compile icon  and then double click on the nodes to bring up their probability monitors as shown in Figure 3-29.



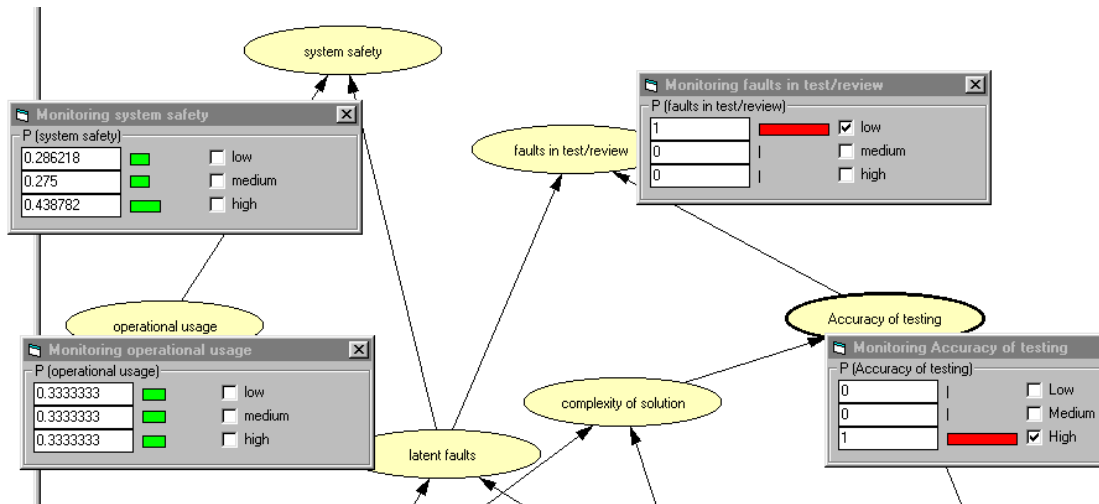
**Figure 3-29: Full net in initialised state**

In this initialised state we are more or less completely indifferent about the safety level of the system. Suppose that, from testing results alone we know that a low number of faults were found. We can enter this observation and click on the execute icon  to get the result shown in Figure 3-30.



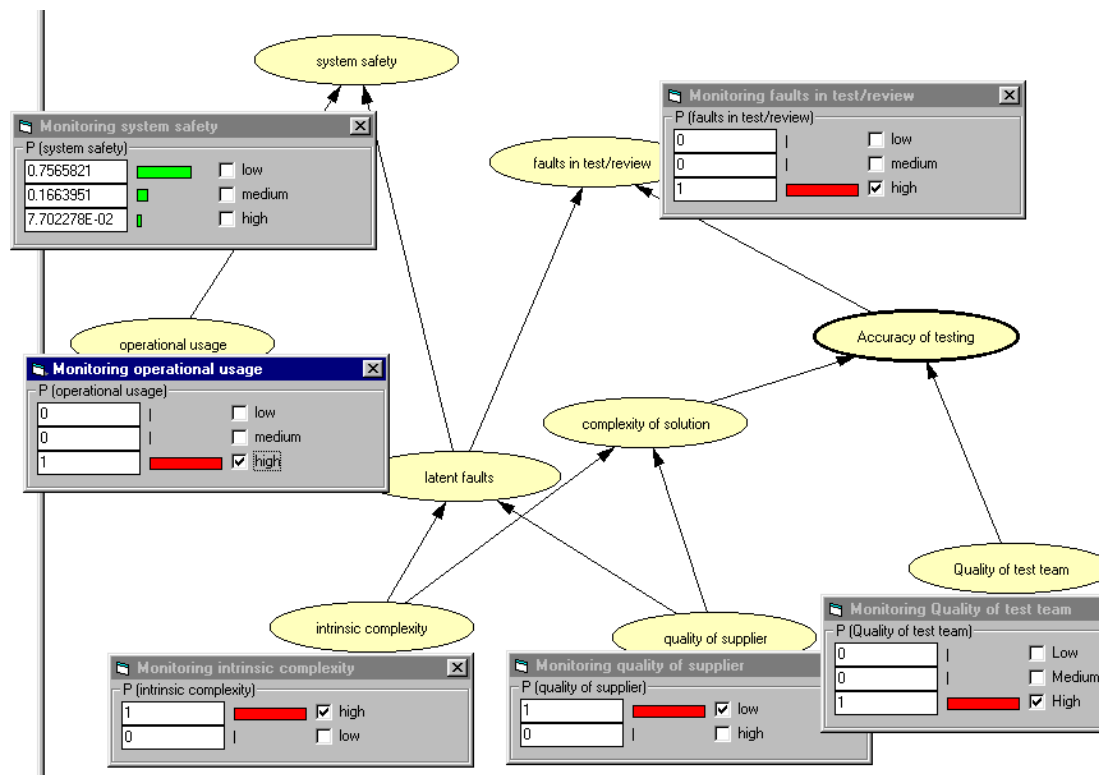
**Figure 3-30: Low faults observation entered**

Notice that with this observation our belief about safety has very slightly moved toward the higher end. The reason there is no great change is that we know nothing about the testing process. Suppose that we know that the testing accuracy is high. Then, as shown in Figure 3-31, we now have a much stronger belief that the system has high safety (if you enter a low accuracy observation then this ‘explains away’ the low number of faults and reverts the probabilities for system safety almost back to their initialised values).



**Figure 3-31: Low faults, high testing accuracy**

In Figure 3-32 we have entered observations which represent the most negative information we could have about the system. Specifically, the number of faults found in testing is high (and we also know the quality of the test team is high), the operational usage is high, the intrinsic complexity is high, and the quality of the supplier is low.



**Figure 3-32: 'Worst' evidence**

Notice how in this case there is now a very high probability that the system safety is low.



Figure 3-33 shows the opposite extreme, with the most optimistic evidence. In this case there is a very high probability that the system safety is high.

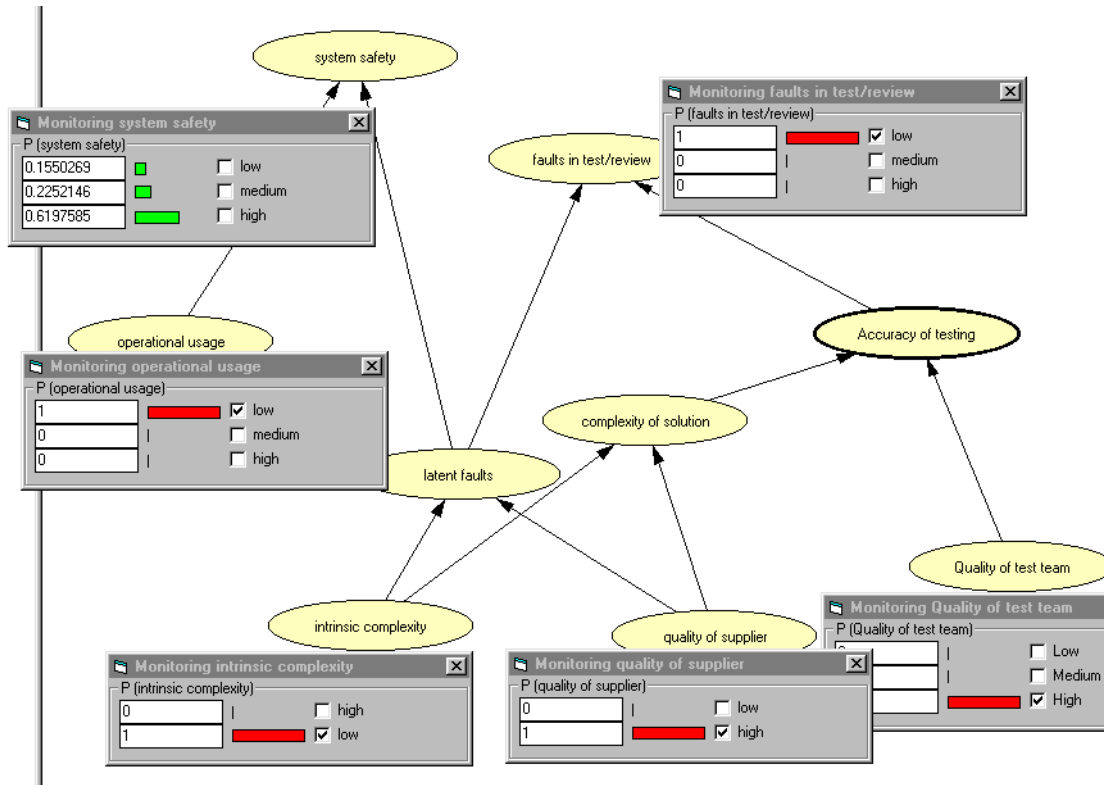


Figure 3-33: ‘Best’ evidence

*This page intentionally left blank*

## 4 Examples of using the SERENE Method to Construct Safety Arguments

The purpose of this chapter is to provide examples of templates derived from various different arguments, and from various different perspectives. The templates are described in terms of the idioms of which they are comprised, with descriptions and diagrams to show how they are joined. Various different execution scenarios are worked through to illustrate their use.

The arguments covered are:

- Defect density
- Reliability
- Safety risk

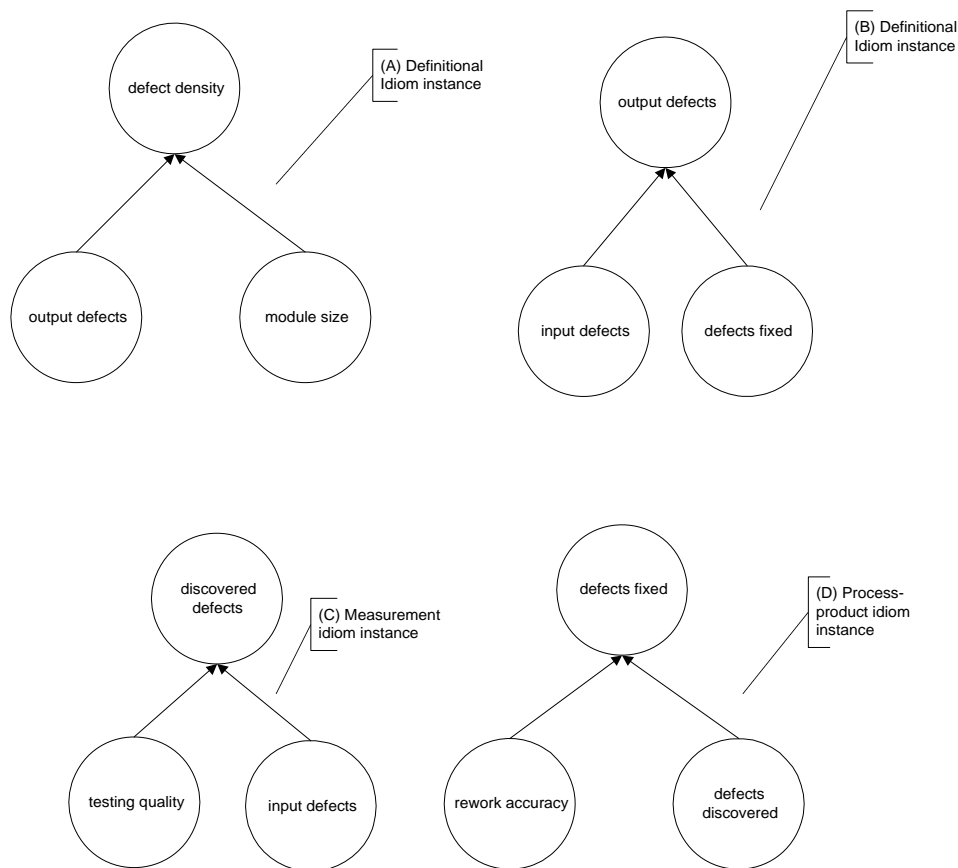
### 4.1 The Defect-Density Argument

The goal of this argument is to predict the defect density of a piece of software. Defect density is defined as the number of residual defects divided by the estimated size of the design. For example, a typical defect density may be expressed as defects per KLOC (Kilo lines of code) or per FP (Function Point).

#### 4.1.1 Defect Density: Idiom Instances

The definition of defect density suggests the use of a definitional/synthesis idiom. Idiom instance (A) in Figure 4-1 shows the use of this idiom to model defect density. Next we might wish to model the process by which defects are introduced, fixed and removed. Idiom instance (B) shows another instance of the definitional/synthesis idiom where output defects are defined as the difference between input defects and those defects fixed during the process. The number of defects left in the software after a number of process stages will simply be the number of defects introduced accidentally minus the number detected and fixed (assuming a perfect fixing process).

The number of detected defects will depend on how much effort is made to find them during each testing phase and how many defects are indeed there to be discovered. Idiom instance (C) shows how the discovery of defects is limited by the number of input defects and testing quality.



**Figure 4-1: Idiom Instances for Defect Density Template**

Process-product idiom instance (D) shows how the rework process fixes input defects size. The accuracy of the rework process determines the number of defects fixed.

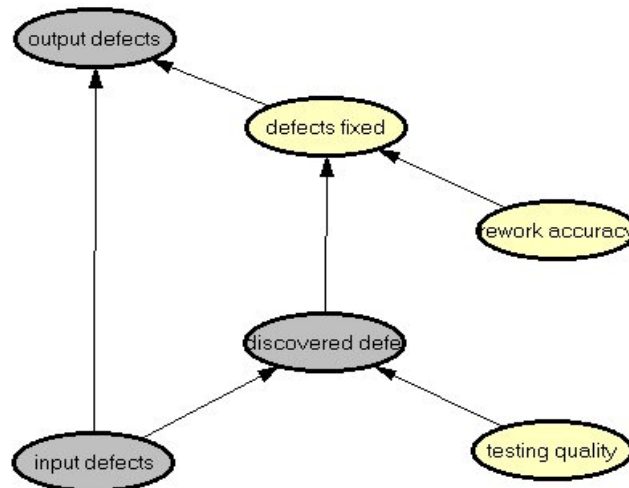
#### 4.1.2 Creating the Defect Density Argument from Templates

The idioms are general enough to lend themselves to the specification of any linear process where input defects are transformed, by testing and rework processes, into output defects. The output defects from one stage,  $i$ , are the input defects for stage  $i+1$ .

The defect density argument is formed from the following templates:

- one\_phase\_rework\_template
- density\_template

The one\_phase\_rework template is shown in Figure 4-2.



**Figure 4-2: One\_phase\_rework template**

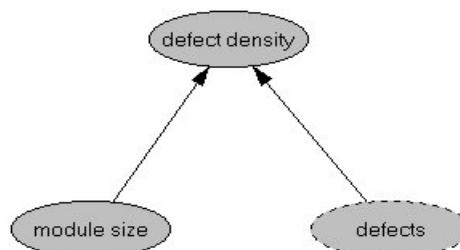
The `one_phase_rework` template is constructed from the idiom instances discussed in Section 4.1.1. The number of input defects discovered during testing is determined by the testing quality. The number of these discovered defects actually fixed is a function of the rework accuracy. Defects removed by successful rework decreases the number of input defects to the number of output defects.

The NPTs for this template are defined as follows:

- $p(\text{discovered defects} \mid \text{testing quality}, \text{input defects}) = \text{Beta}((50 - 1) / 5 * \text{test\_quality} + (50 - (50 - 1)), (1 - 50 / 5) * \text{test\_quality} + (1 - (1 - 50)), 0, \text{input\_defects})$
- $p(\text{defects fixed} \mid \text{discovered defects}, \text{rework accuracy}) = \text{Beta}((50 - 1) / 5 * \text{rework\_accuracy} + (50 - (50 - 1)), (1 - 50 / 5) * \text{rework\_accuracy} + (1 - (1 - 50)), 0, \text{disc\_defects})$
- $\max(\text{input\_defects} - \text{fixed\_defects}, 0)$

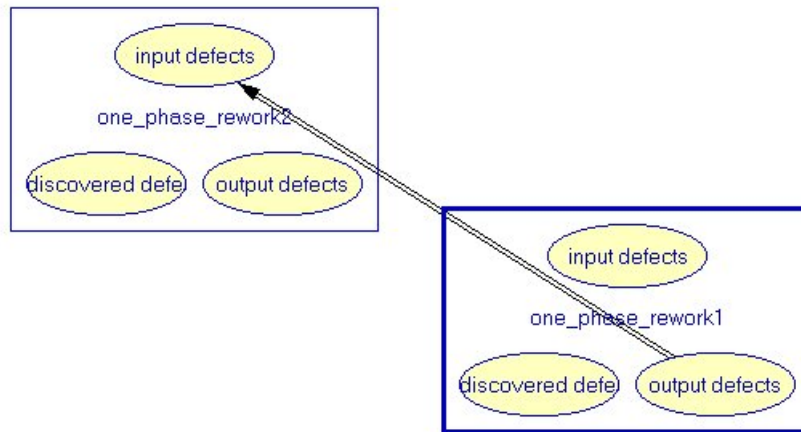
The complex Beta functions have been used to approximate expert judgements instead of the expert completing the large NPT. A procedure has been created to automatically generate this function by interpolating best and worst cases using an appropriate interpolation indice such as, in this case, testing quality or rework accuracy. The specification of output defects is obviously a simple deterministic function, with a max function to prevent negative numbers arising.

The density template is shown in Figure 4-3. Here defect density is simply the number of defects divided by the module size. The defect density node can be used to model input, output, fixed or discovered defects from the `one_phase_rework` template.



**Figure 4-3: Density template**

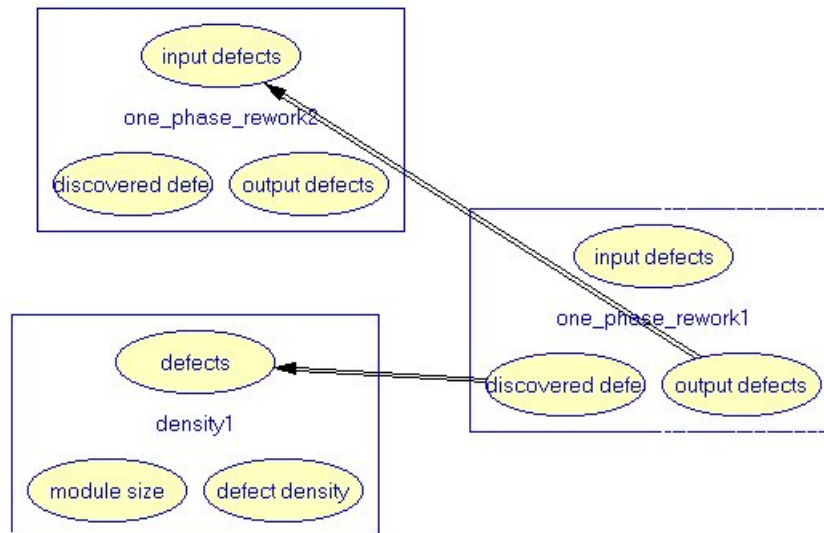
To create any argument based on defects detection and removal we can join these two, very generic, templates together as shown in the `two_phase_rework` template in Figure 4-4.



**Figure 4-4: Two\_phase\_rework template**

The two\_phase\_rework template results from simply joining two instances of the one\_phase\_rework template together. This operation is performed in the serene tool by copying the one\_phase\_rework template twice. The output node from the first template and the input node from the second template are joined.

By adding the density template, to evaluate the defect density of the discovered defects in one\_phase\_rework, we get the network shown in Figure 4-5.



**Figure 4-5: Two\_phase\_rework template with density template added**

### 4.1.3 Defect Density: Execution Scenarios

To illustrate how the defect density argument might be used we can execute a number of scenarios using evidence. Each scenario shows the effects that entering evidence has on the predictions made by the BBN.

Firstly we assume that there are between 18 and 21 defects in the system. If we apply a better than average quality of testing (3-4) and a very high rework accuracy (4-5) our forecast of the number of defects left in the system (output defects) is a distribution with a mean of 9 defects. Entering and propagating this evidence results in a forecast from the one\_phase\_rework template are shown in Figure 4.6.

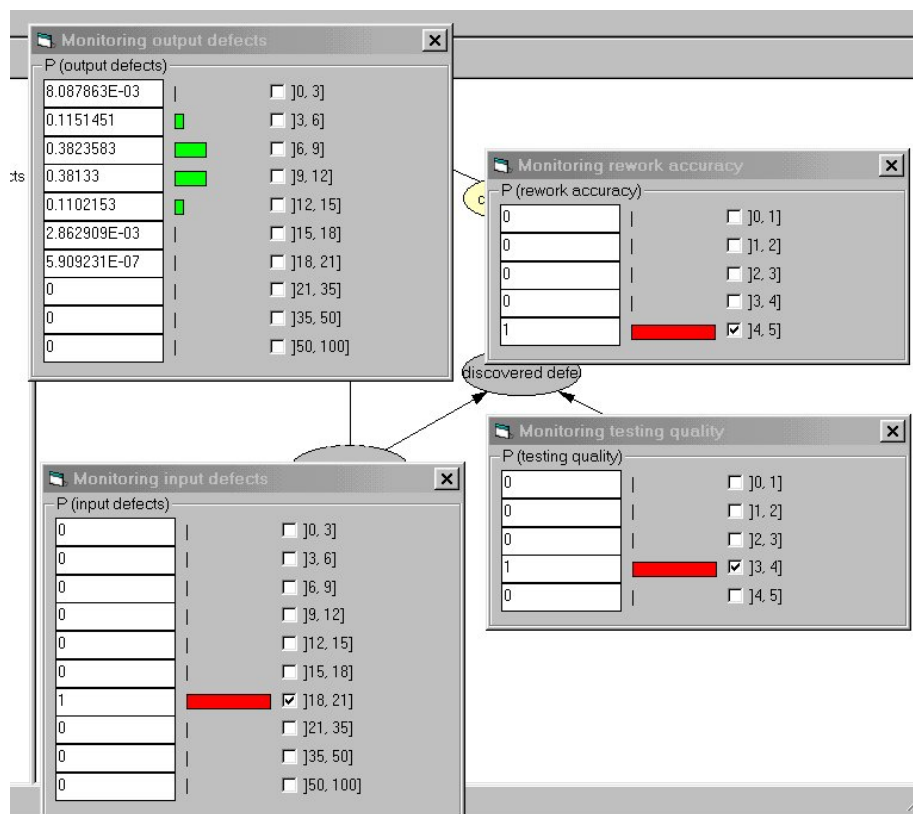


Figure 4-6: One\_phase\_rework template with evidence entered

The defect density of the system, given a module size between 4,000 and 5,000 lines of code in length is shown by the distribution in Figure 4-7. The defect density is most likely between 2 and 3 defects per KLOC.

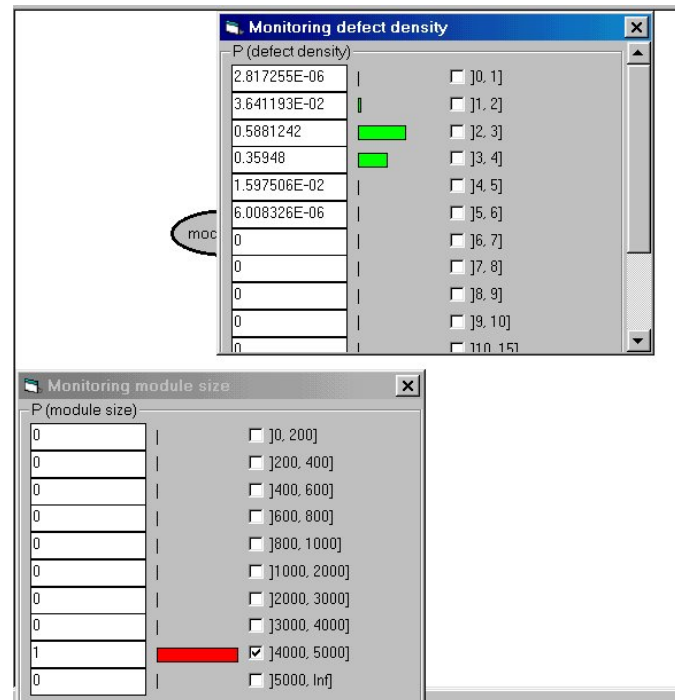


Figure 4-7: Density template with evidence entered

## 4.2 Reliability Argument

The goal of the reliability template is to predict the reliability of a piece of software. For discrete use systems reliability is defined as the probability of failure on demand. Therefore, we wish to predict the distribution of failure probability given evidence gained from direct testing of the software and the processes applied in its development.

### 4.2.1 Reliability: Idiom Instances

The definition of reliability suggests the use of the measurement idiom. Idiom instance (A) in Figure 4-8 shows the use of this idiom to estimate probability of failure on demand,  $p$ , from the failures observed,  $m$ , in a number of demands,  $n$ .

$$p(m/n, p) = \binom{n}{m} p^m (1-p)^{n-m}$$

However we might be unsure about how much trust to play in the testing process and by implication the failure count observed. Idiom instance (B) shows the measurement idiom used to model how the accuracy of the testing oracle influences the estimated failure count.

The probability of failure on demand ( $p$ ) can be defined by appealing to the notion of software testability. Here  $p$  as a function of the number of faults and the probability that these faults are revealed or triggered during a demand. Complex pieces of software may contain many and varied execution trajectories (paths) any of which may be executed by a demand on the system. Furthermore the inputs required to test these may be difficult to devise in advance of real use. If a fault lies on an execution trajectory which is executed rarely then the fault will only be revealed with low probability. The larger the number of execution trajectories the lower will be the probability of a demand revealing a fault if indeed one exists. We can conclude that, despite a large number of faults in the system, the majority may only cause failures relatively rarely. Idiom instantiation (C) shows the



definitional idiom instance for this, where the function connecting probability of failure on demand, number of faults,  $f$ , and probability that a demand reveals a fault,  $p'$ , is:

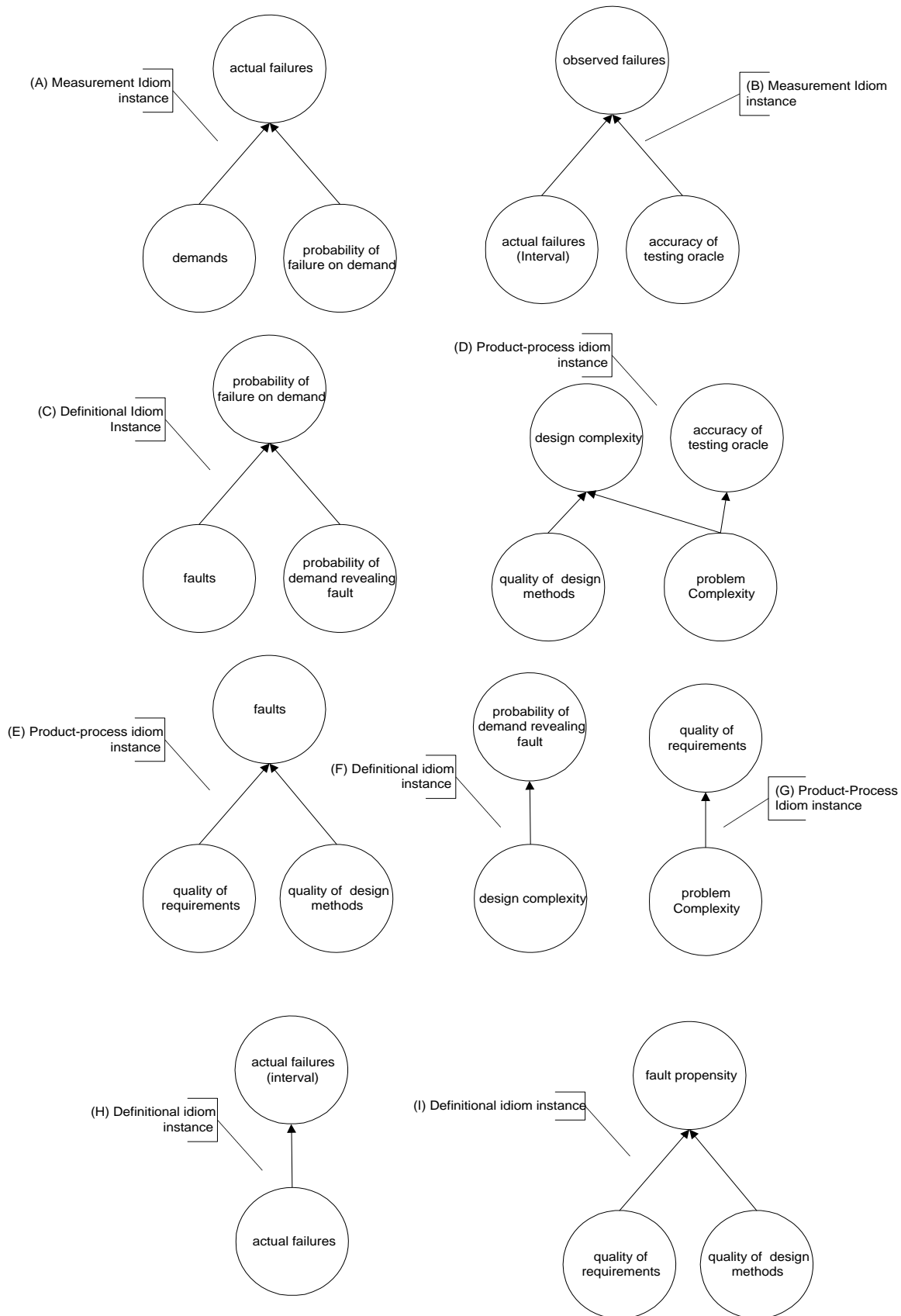
$$p = p(\text{trigger at least one fault}) = 1 - (1 - p')^f$$

The quality of requirements for the software, the accuracy of the test oracle and the design complexity will all be influenced by the underlying problem complexity. We might also wish to consider the specific effects of design method on the design complexity and requirements quality. Idiom instance (D) illustrates this product-process idiom. Similarly the number of faults introduced into the software may be caused by the interaction between requirements quality and design method quality. Poor design methods and poor quality requirements will likely jointly cause many faults. This is shown by idiom instance (E) another product-process idiom.

Idiom instantiation (F) shows how the definitional idiom can be used to characterise the attribute design complexity and its sub-attribute testability, defined here as probability of demand revealing fault. There may be many sub-attributes of design complexity but we will concentrate on only one here. Finally idiom instance (G) shows the process-product relationship between problem complexity and quality of requirements.

Two synthetic nodes have been created to aid calculation in the network. Firstly, we use a Beta function to model the relationship  $p(\text{observed failures} \mid \text{actual failures, accuracy of testing oracle})$ . The Beta function requires a continuous specification rather than a discrete one for the node, therefore the actual failures node was transformed into the actual failures (interval) node by simply allowing the interval child node take values equal to its parent. We then conditioned observed failures on actual failures (interval) rather than the observed failures node. This is shown by idiom instantiation (H).

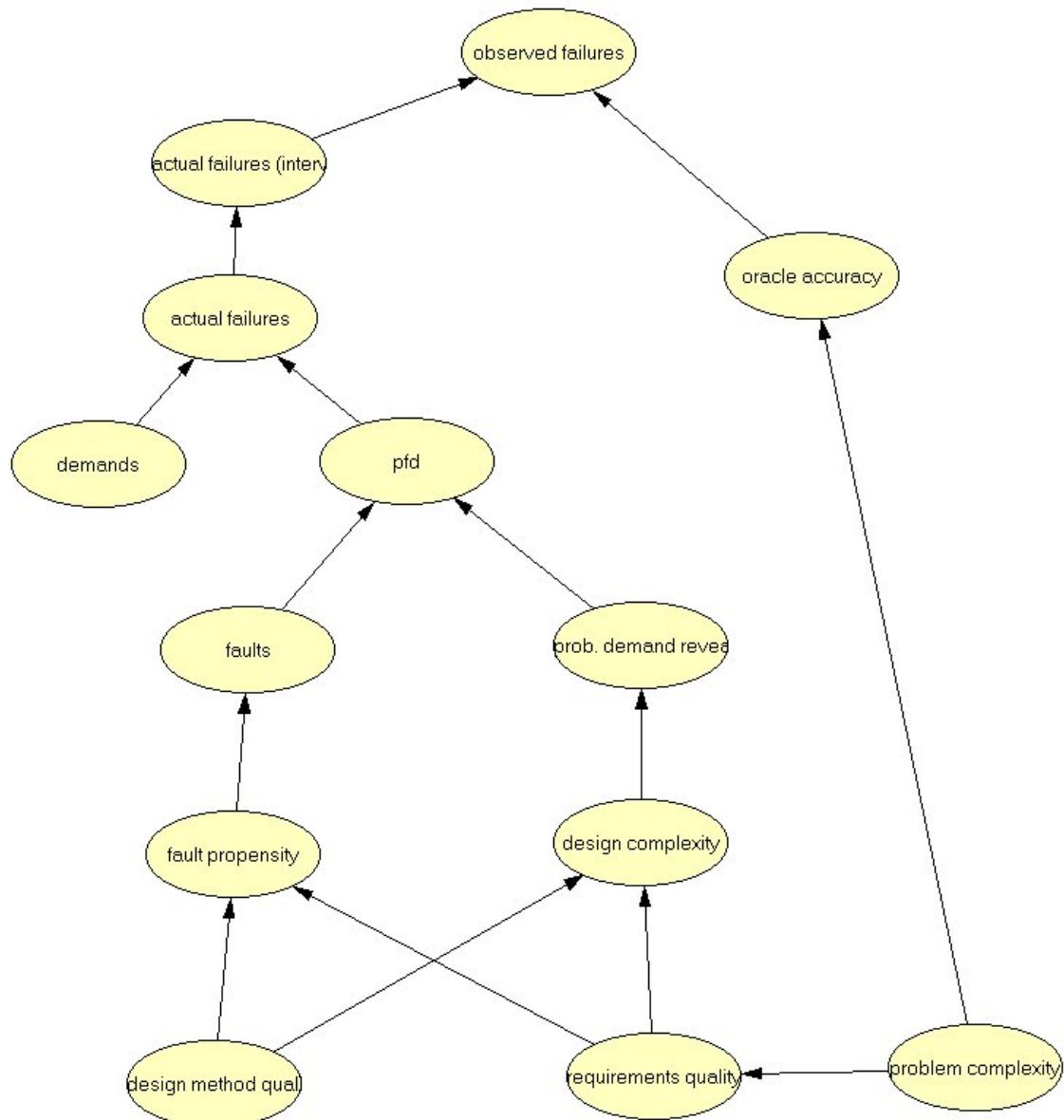
The second synthetic node, fault propensity, was used to create a parameter with which we could estimate the number of faults introduced by the development process. This is shown by idiom instantiation (I).



**Figure 4-8: Idiom Instances for Reliability Template**

## 4.2.2 Building the Reliability Argument

By joining the idiom instantiations we get the reliability template shown in Figure 4-9.



**Figure 4-9: Reliability template**

From information about the development process concerning design method quality, requirements quality and the problem complexity we can estimate the complexity of the design and the number of faults likely to be introduced into that design. The probability of failing on demand will depend on how many faults are in the system and whether these faults are more or less likely to reveal themselves. The actual number of failures observable in a campaign of testing will depend on the number of demands and the probability of failing per demand. However if the oracle used to determine failure is inaccurate the observed number of failures may underestimate the true figure. The accuracy of the testing oracle depends on the problem complexity since it is difficult to create unambiguous test criteria for difficult problems.

### 4.2.3 Reliability: Execution Scenarios

The initial states for the reliability template are shown in Figure 4-10. This shows the network when no evidence is available and represents the prior marginal probabilities for each node. We can see that the probability distribution for actual failures shows a 38% chance of observing no failures but a significant chance of imperfection. This means if we knew nothing about a product we would be more likely to expect unreliability than perfection. The other nodes can be interpreted in similar ways.

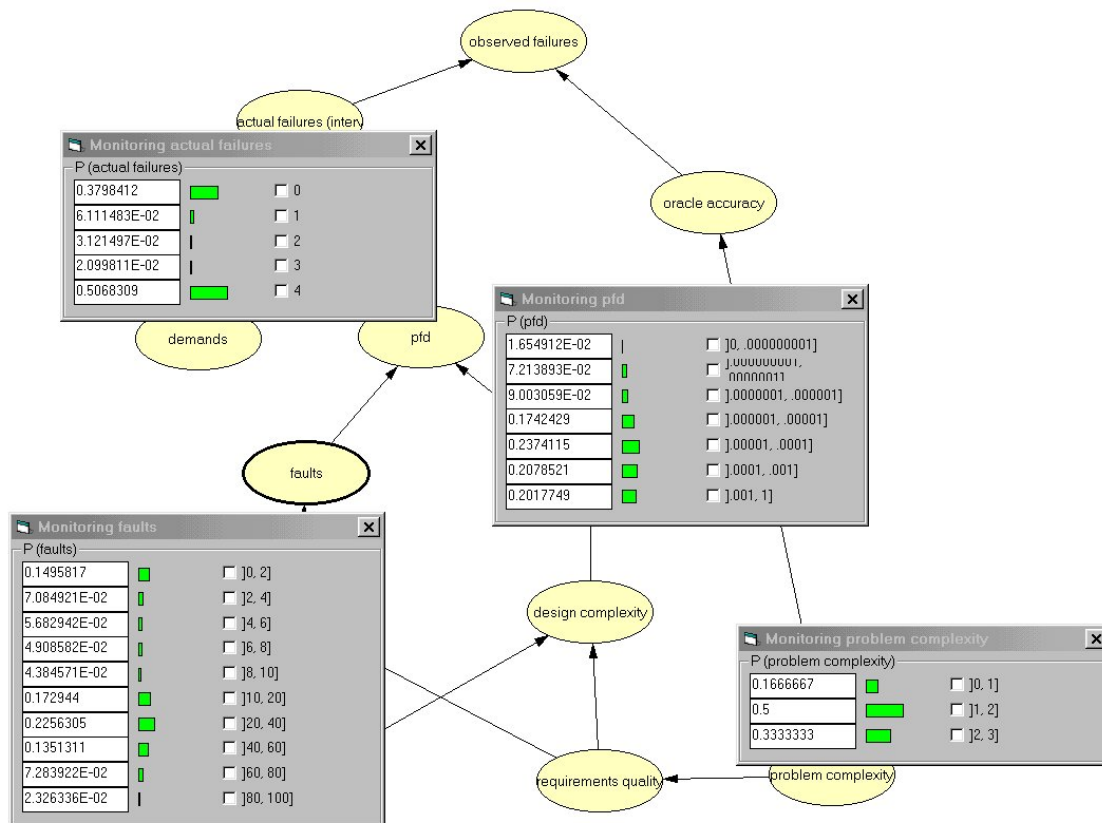
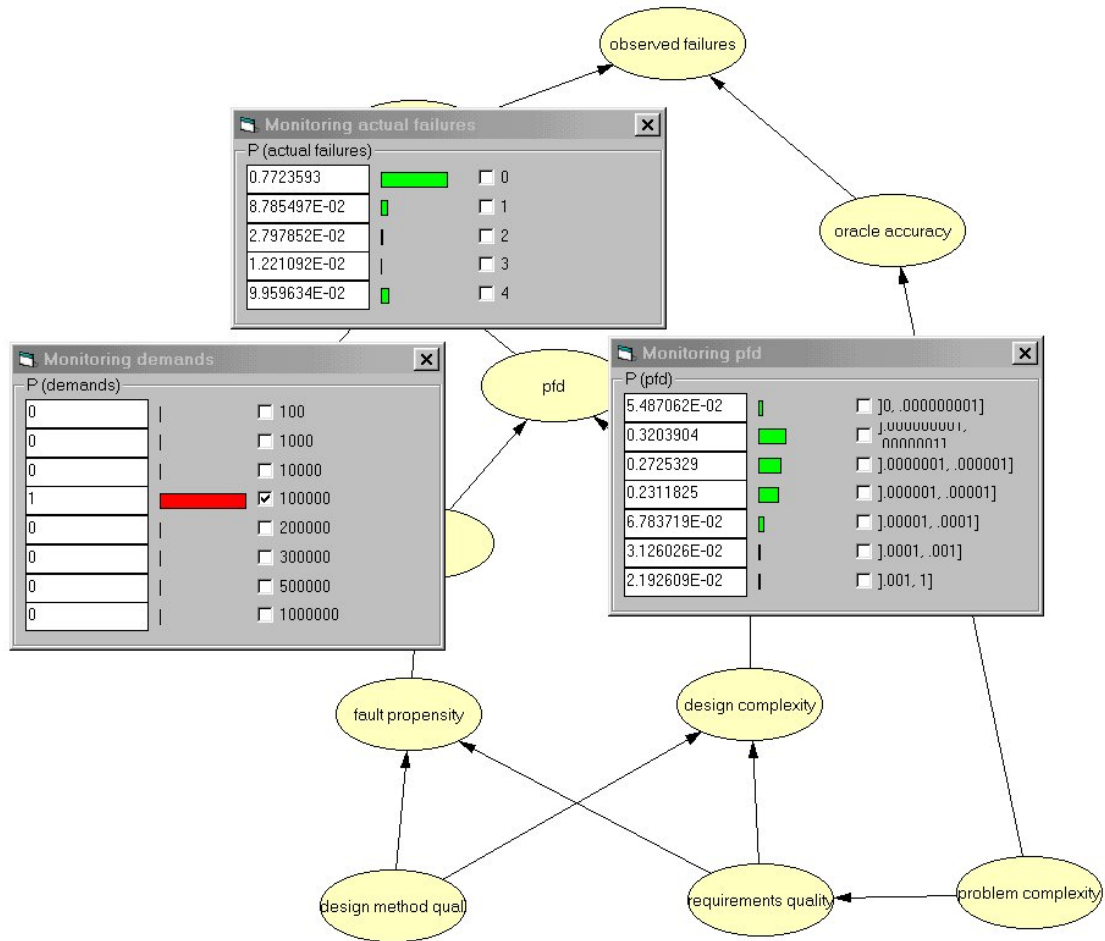


Figure 4-10: Reliability Template Initial States

If we enter evidence into the template to represent a “best case” scenario we get the scenario shown in Figure 4-11.



**Figure 4-11: Reliability Template Best Case Scenario**

In Figure 4-11 the number of demands expected is 100,000. Design complexity is low, quality of requirements is high, and problem complexity is high. The network calculates the consequences of this and forecasts a probability of failure on demand distribution with most of the probability mass around  $10^{-5}$  to  $10^{-7}$ . This is a significant change from the initial state. Alongside the forecast for probability of failure on demand the template forecasts the number of failures as 77% chance of zero failures in 100,000 demands but approx. 16% chance of greater than one.

We can compare a worst and best case scenario to evaluate the differences in forecasts. Under the worst case scenario we might expect to find poor requirements, testing oracle inaccuracy and high design complexity. This is shown in Figure 4-12.

Under this worst case scenario the reliability forecast is not very encouraging because the distribution for probability of failure on demand is now centered on  $10^{-2}$  to  $10^{-5}$ , a significant drop from the best case. Under 100,000 demands this translates into a 44% chance of more than one failure occurring and only 34% chance of zero failure. Even though no failures were discovered during testing this evidence can be dismissed as irrelevant because we do not trust the testing oracle.

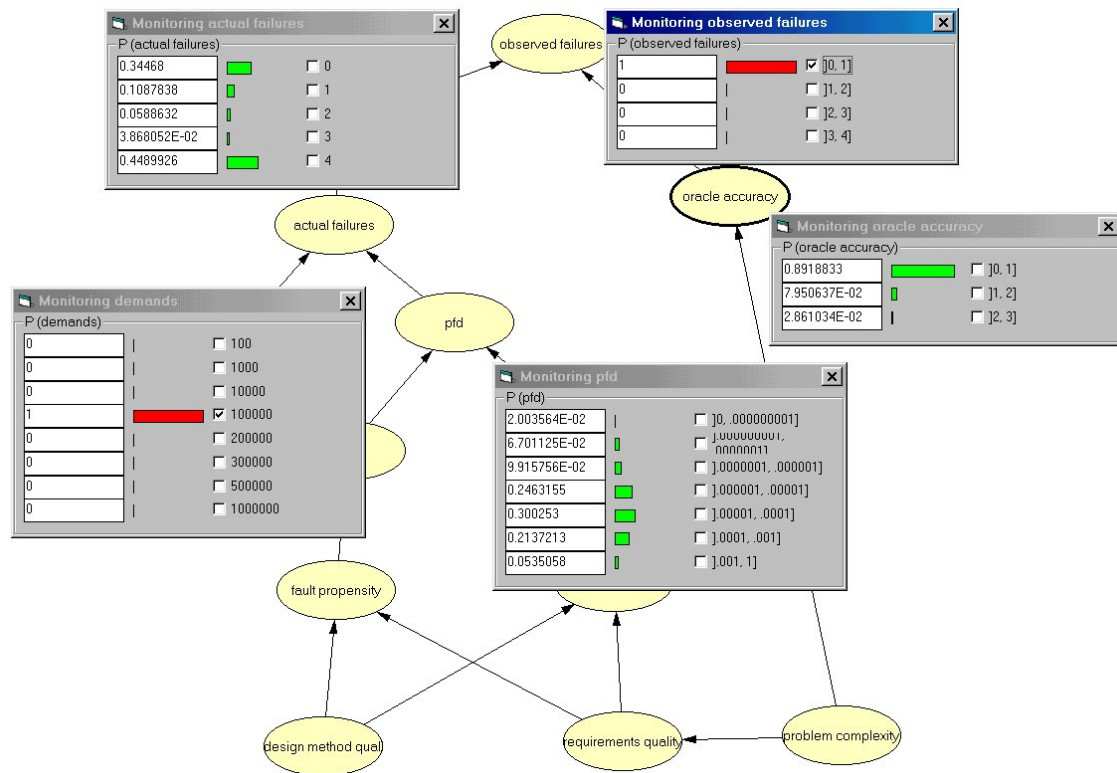


Figure 4-12: Reliability Template Worst Case Scenario

### 4.3 The Safety Risk Argument

The goal of this argument is to predict the safety risk of a piece of software. The risk of using a software system is defined as the frequency with which hazardous events might occur combined with the severity of those events. The most severe events might be those that cause physical harm and the less severe might incur inconvenience or financial loss.

The general scenario being modeled here includes the following data sources:

- Information about the accuracy/difficulty of the requirements;
- Testing data produced by an independent testing organization;
- The quality of the testing performed by these independent testers;
- The complexity of the product being produced and the impact this has on testing quality;
- The capability of the developers producing the software.

The argument is split into a number of SERENE templates each of which model specific parts of the overall argument:

1. **safety\_argument template** – this combines all of the other templates into the safety argument. The modular structure of the argument is declared within this template;
2. **safety template** – this template contains the “core” causal reasoning connecting the different data sources listed above;
3. **severity template** – contains a collection of nodes that define severity and which can be reconciled with the severity prediction contained in the safety template;

4. *test\_quality template* – contains nodes that define the quality of the testing that was done by the independent testing organization;
5. *testing\_group\_quality* template – contains nodes that define the testing capabilities of the testing organization;
6. *developer\_quality template* – contains nodes that define the quality and capabilities of the development team and organization.

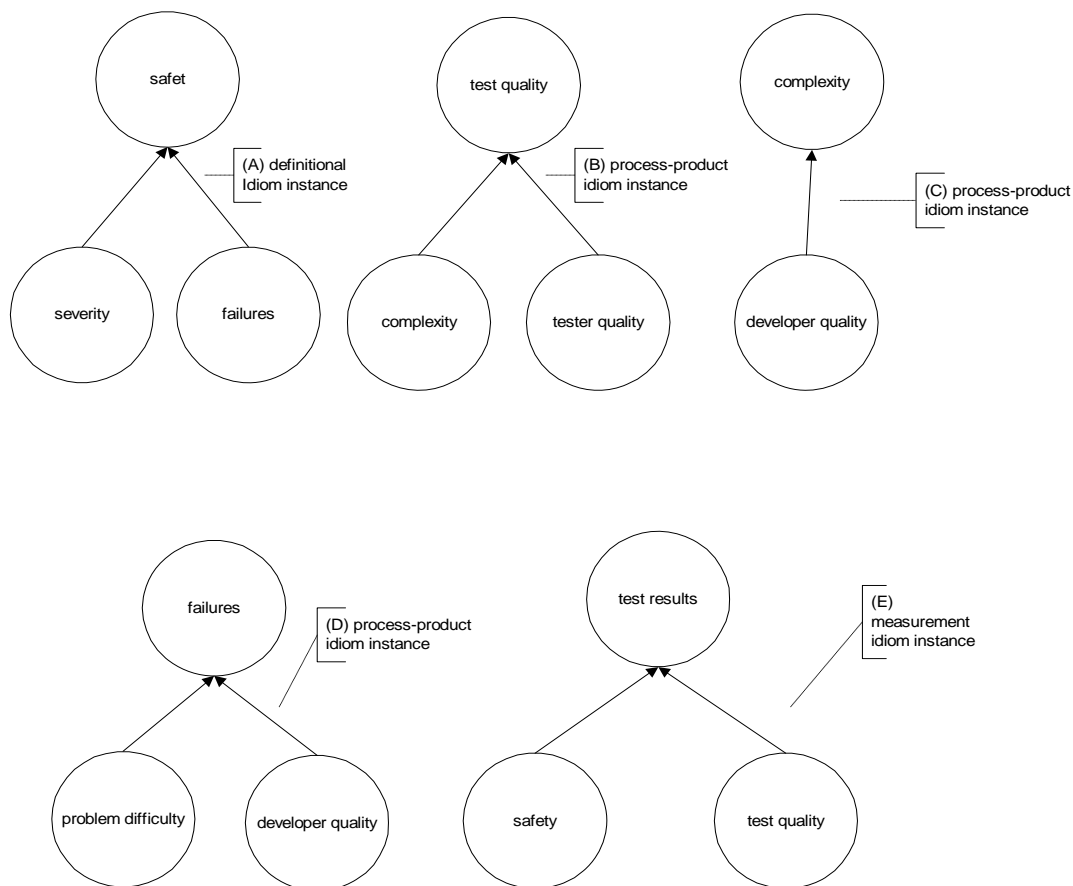
The only template that contains causal reasoning is the safety template. In many ways this is the core-reasoning component in the safety argument as a whole. All other templates are subsidiary to this and depend on it. The other templates (numbers 3-6 in the list above) involve definitional/synthesis idioms and reconciliation idioms were used to join them to the core safety template. In this way causal information can be reconciled with uncertain estimates gained by measurement rather than prediction. How this works should become evident in the examples presented later in this section.

We term the safety template the “core” template and the others the “satellite” templates.

### 4.3.1 Idiom Instances in the Safety Risk Template

We have pointed out that the safety template is the core network for this safety argument. It is therefore worthwhile to concentrate our attention on the idiom instances used to compose this template in particular.

The idiom instances are shown in Figure 4-13.

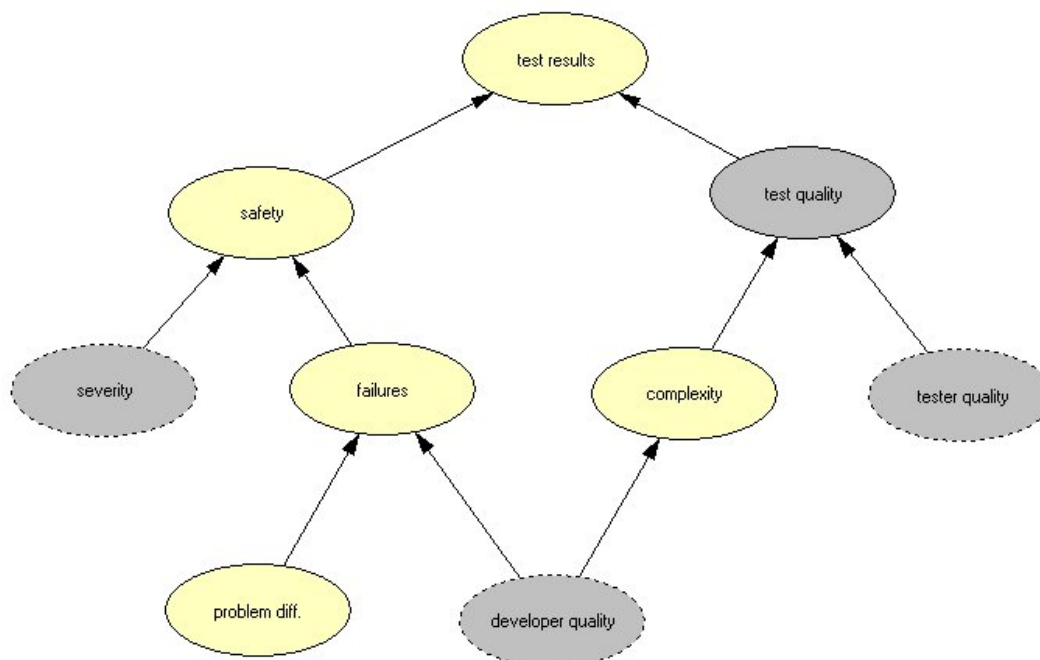


**Figure 4-13: Idiom instantiations in the safety template**

Idiom instance (A) defines the relationship between severity, failure frequency and safety according to the standard definition. Idiom instance (B) shows how the complexity of the system and the tester quality influence the quality of the test results. High complexity makes testing difficult and reduces the accuracy of any forecast of the true risk. Poor quality testing personnel also militate against accurate testing. The relationship between developer quality and the complexity of the system produced is shown by idiom instance (C): poor quality developers may be more likely to develop a system that is overly complex. The frequency of failures a system exhibits in use will depend on the problem difficulty and the quality of the developers producing that system. “Easy” problems may lead to systems with fewer failures in operation. On the other hand poor quality developers might introduce defects leading to failures. Finally, in idiom instantiation (E) the test results produced by the independent testers only provides an estimate of the true safety of the system. The extent to which this estimate can be trusted as reliable will depend on the test quality itself.

### 4.3.2 The Safety Risk Template

By joining these idioms together at the common nodes we can produce the topology of the safety template as shown in Figure 4-14.



**Figure 4-14 Safety template BBN topology**

The nodes denoted as input and output nodes are shared, by the join operation, with the other templates.

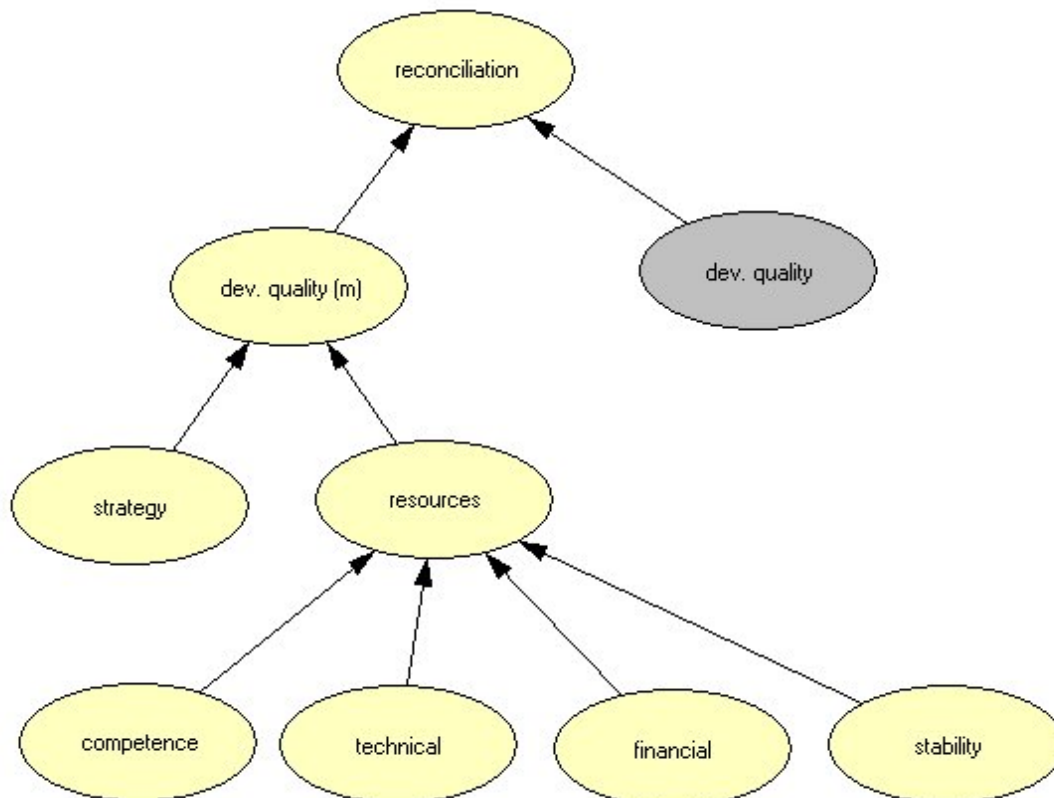
Each node is defined on an interval scale from 1-5 indicating very high to very low levels of quality or safety. Some of the causal nodes have been calculated using deterministic functions to complete the NPTs. Prior distributions were allocated to developer quality, problem difficulty and tester quality.

### 4.3.3 The Satellite Templates in Safety Risk Argument

For each major node in the core template a satellite template has been produced. These templates are mainly modelled by using the definitional/synthesis idiom. Because the method of reasoning in these templates is very similar so we will present only one template in detail and simply show the graph topologies of the others.



We will concentrate on the developer\_quality template as shown in Figure 4-15.



**Figure 4-15: Developer quality template**

The developer quality template shows how the definitional/synthesis idiom has been used to create a hierarchy of sub-attributes that define the attribute “developer quality (m)”. The (m) is used in the name to indicate that this node is inferred by measurement. Thus developer quality is defined as being composed of strategy and resources. Resources are defined as a composition of competence of personnel, technical and financial resources and the stability of the organisation.

At the top of the developer\_quality template is the triad of nodes — developer quality (m), developer quality and reconciliation. This is an instantiation of the reconciliation idiom. Here we wish to reconcile uncertain estimates arrived at by estimating the sub-attributes of developer quality with any other estimate – in this case the probability distribution of developer quality in the core network. By using the reconciliation idiom we can use the likelihoods generated by measurement as “observations” on the core safety template. In this way we reconcile different types of evidence about the same thing.

All of the other satellite networks have been constructed in the same way — a definitional/synthesis network with a reconciliation network on top.

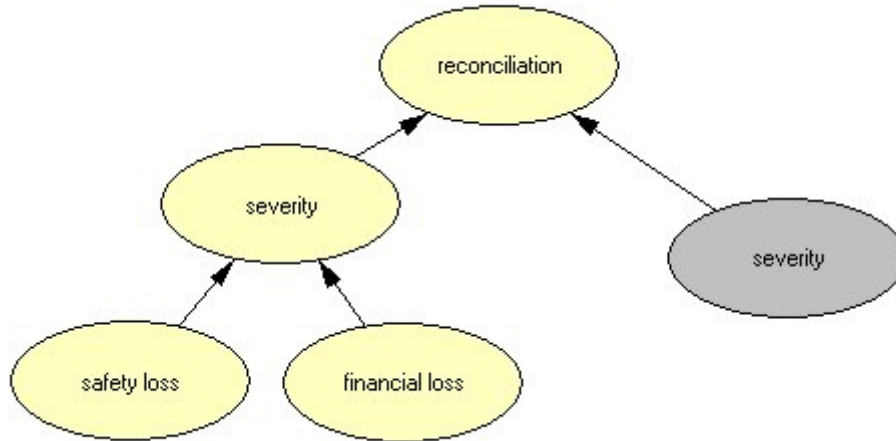
The leaf node NPTs in the developer\_quality template are defined as simple uniform probability distributions. The parent nodes are calculated by any reasonable scoring scheme. In this case we have used weighted averages with equal weights, thus:

- $\text{resources} = (\text{competence} + \text{technical} + \text{financial} + \text{stability}) / 4$
- $\text{dev\_quality\_m} = (\text{strategy} + \text{resources}) / 2$

The measurement scale for the nodes uses a simple interval or numeric scale ordered from 1 to 5, where 5 indicates good/best. We can see that the BBN model for this problem goes beyond a mere checklist in that fairly complex definitional/synthetic relations can be modelled. Also, it has the

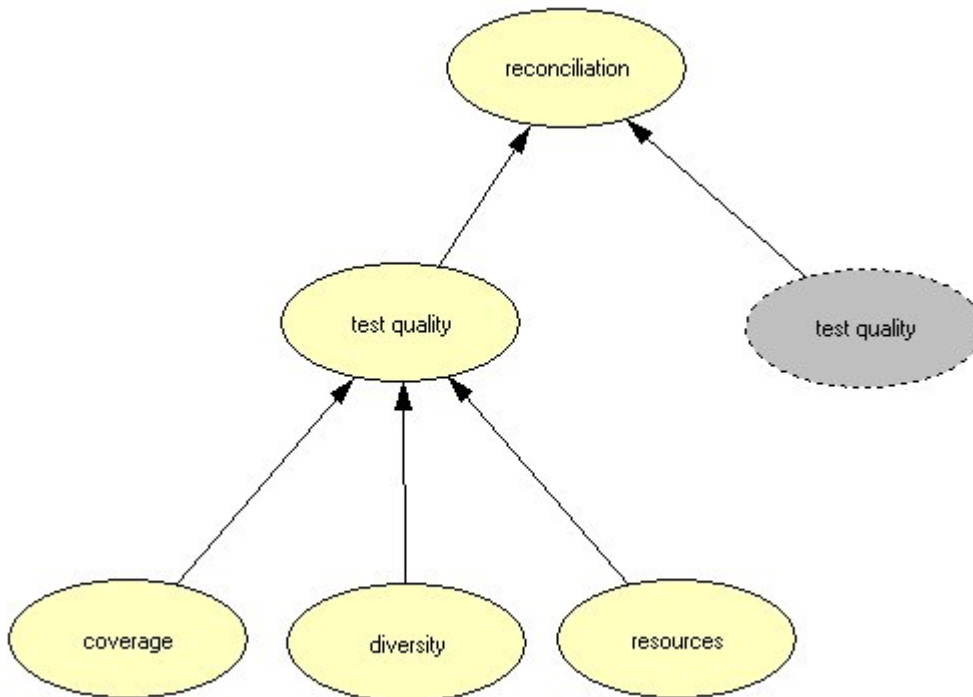
advantage that missing observations, such as knowing nothing about organisational stability, leads to uncertain measurements in the form of a likelihood distribution. A simple checklist or spreadsheet model could not provide this feature without some extra programming.

The other templates are listed in Figure 4-16 to Figure 4-18.



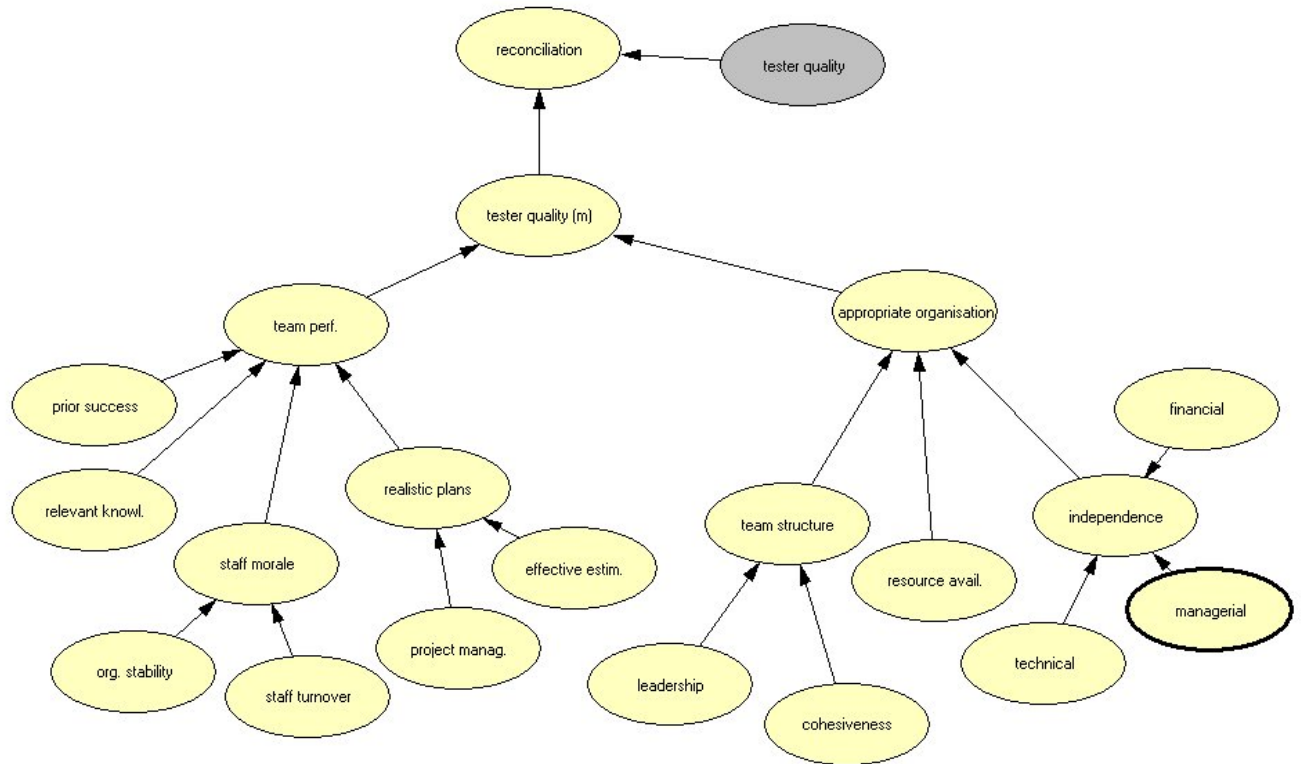
**Figure 4-16 Severity template**

The severity template shows that severity is defined as a combination of safety and financial loss.



**Figure 4-17: Test quality template**

The test quality template shows that test quality is defined as a combination of high test coverage diversity in testing and high resourcing.

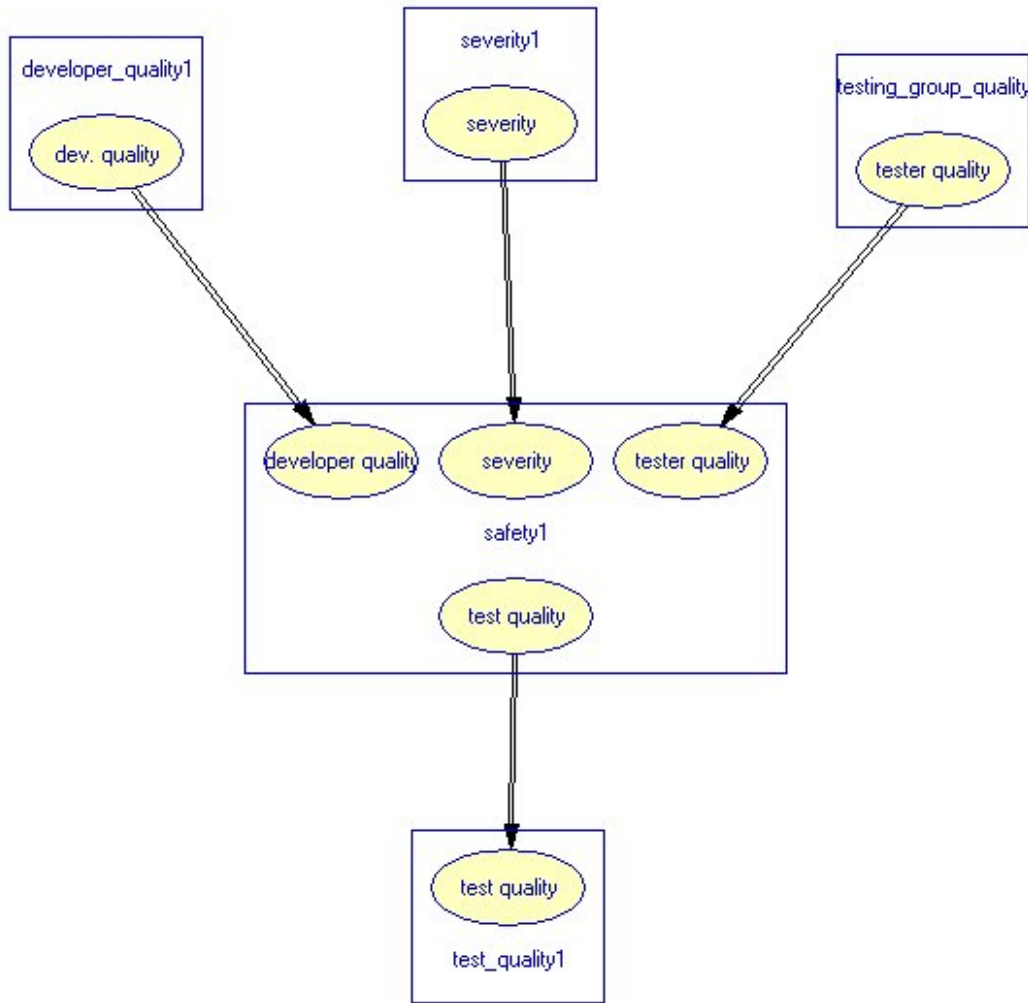


**Figure 4-18 Testing group quality template**

The testing group quality template shows how testing performance and appropriate organisation come together to define high or low testing group quality. Performance is in turn defined as the combination of realistic plans, evidence of prior success and staff morale. These in turn are decomposed further. Appropriate organisation is defined by team structure, resource availability and independence. These are also decomposed further.

**4.3.4 Creating the Safety Risk Argument from Templates**

We can combine these templates to give a complete model by joining the nodes shared between the core and satellite templates. The resulting file is called the safety\_argument template and is shown in Figure 4-19.



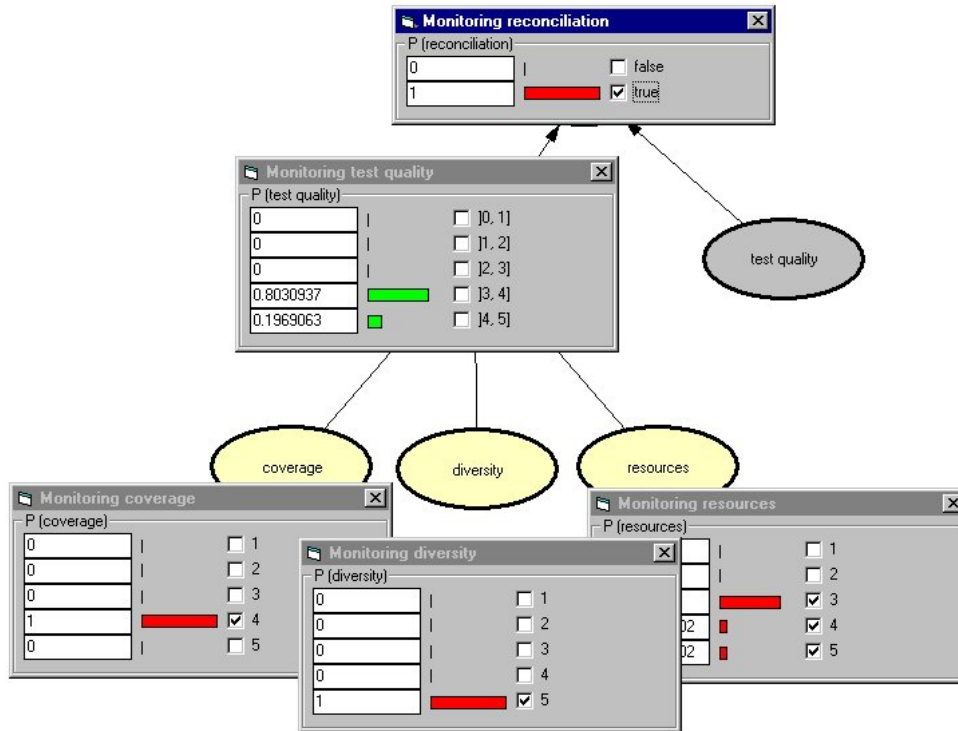
**Figure 4-19: Safety\_argument template**

We can easily see how the core template is connected to the satellite templates via the relevant join operations between the shared nodes.

#### 4.3.5 Safety Risk: Execution Scenario

To illustrate how the safety argument works we can explore an execution scenario by inputting some evidence and reviewing the effects on the safety prediction made.

Firstly, we enter data on the testing itself using the test\_quality template, as shown in Figure 4-20.

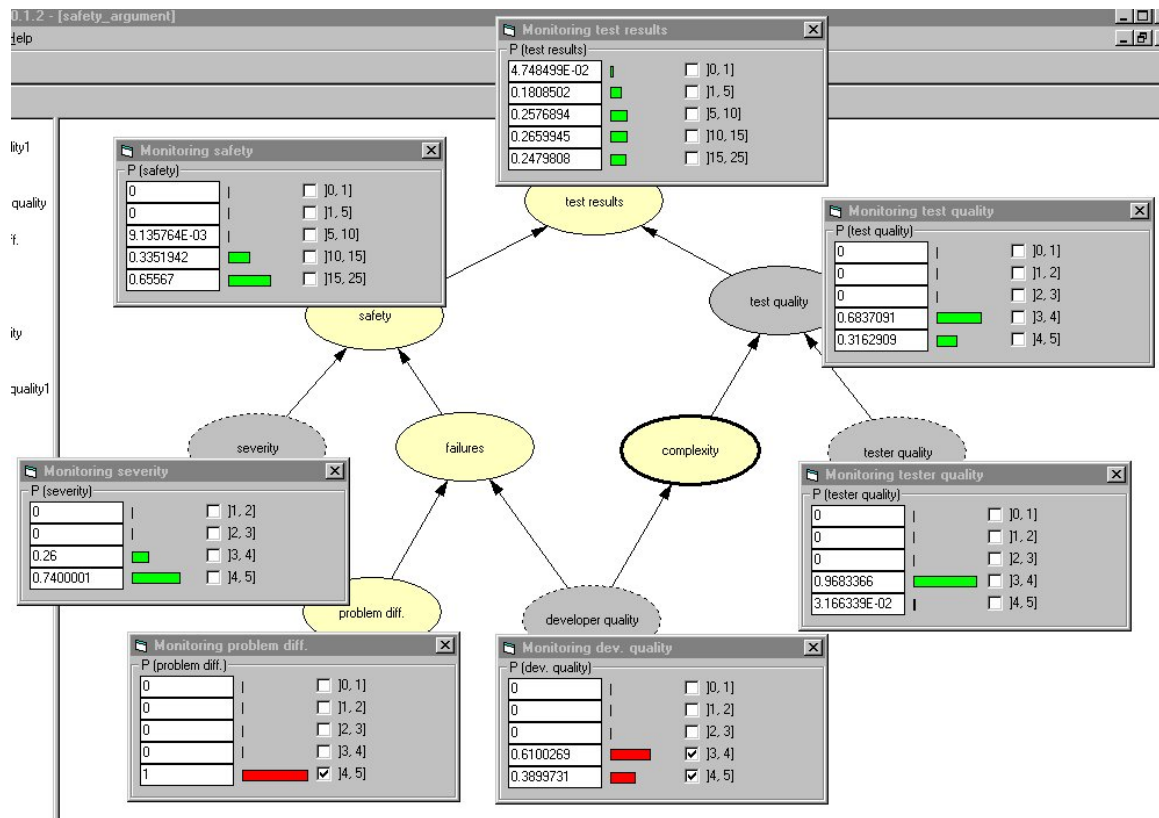


**Figure 4-20: Test quality template evidence entered**

The forecast for test quality, when the measures are reconciled with the marginal distribution in the safety template is that there is good reason to believe that high or very high quality testing has been performed.

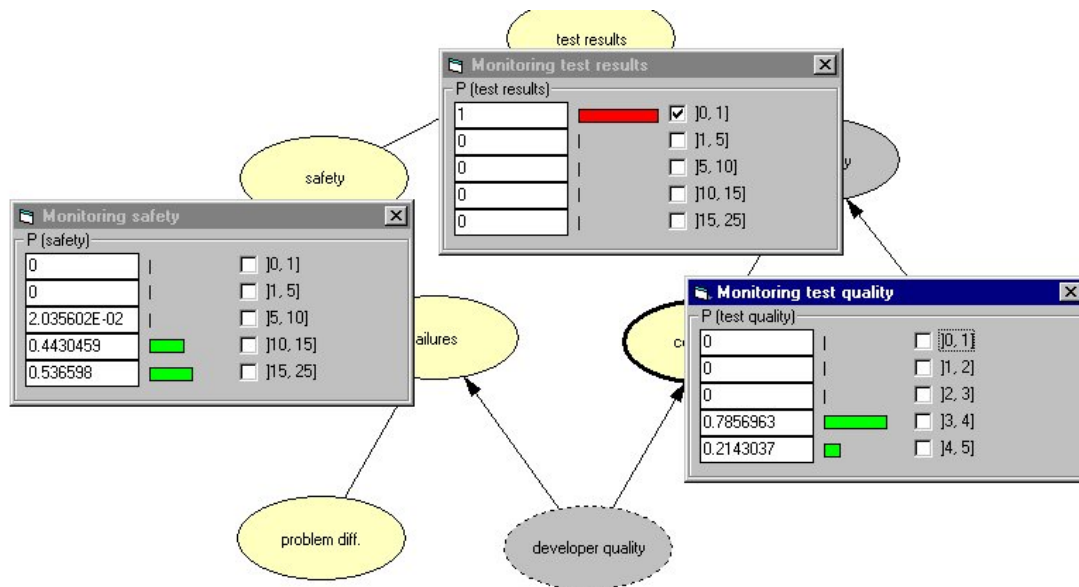
We now turn to the core net and add some additional evidence about testing group quality and severity from other satellite templates. Figure 4-21 shows the effects of adding the following additional evidence:

- the severity of the system is defined as fairly high;
- the developer quality is pretty poor;
- the problem is very difficult;
- evidence about the independent testers indicates that they are quite poor;
- the test quality also reflects this.



**Figure 4-21: Safety argument template with evidence entered**

We can see in Figure 4-21 that the forecast for safety is quite pessimistic. However if one believes that evidence from testing would improve matters one should consider the scenario displayed in Figure 4-22.



**Figure 4-22: Safety template with positive test results**

Figure 4-22 shows that because the quality of the testing was so poor the positive test results have failed to contradict the pessimistic conclusion. This result is consistent with rationale expectations. A poor quality test will accurately estimate the true level of safety. If on the other hand the testing quality was good the positive test results would overthrow the pessimistic prediction.

*This page intentionally left blank*

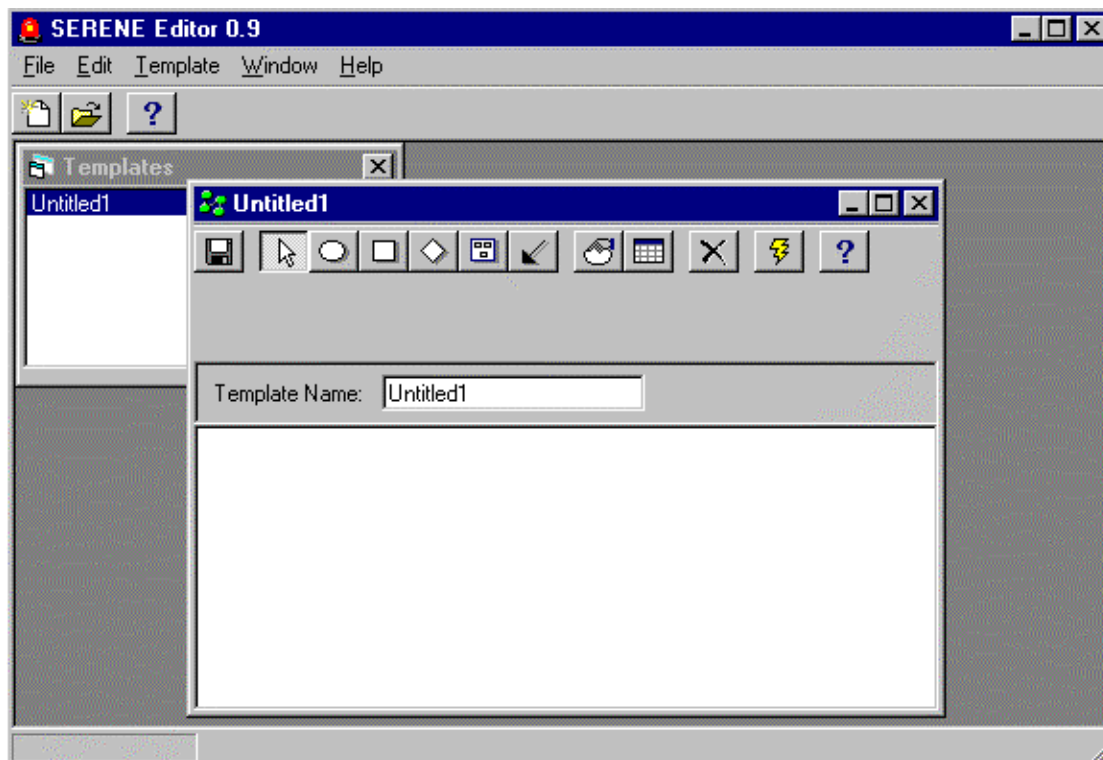
## 5 Using the SERENE tool to build a BBN from idioms

The purpose of this chapter is to demonstrate the use of the SERENE tool in constructing and using a safety argument.

The first section describes the basic functionality of the tool. The second section works through one example in detail. It describes the creation of several templates, which are then joined into one safety argument. Each template is an instantiation of an idiom.

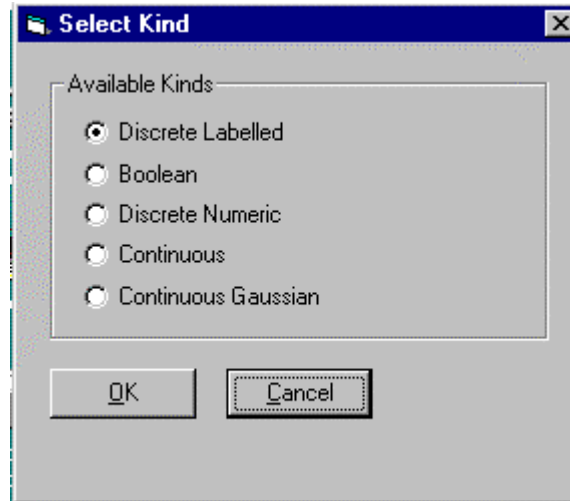
### 5.1 Basic Functionality of the Tool

1. To open a new template, select “new template” from the SERENE tool’s file menu. The template Untitled1 is created.

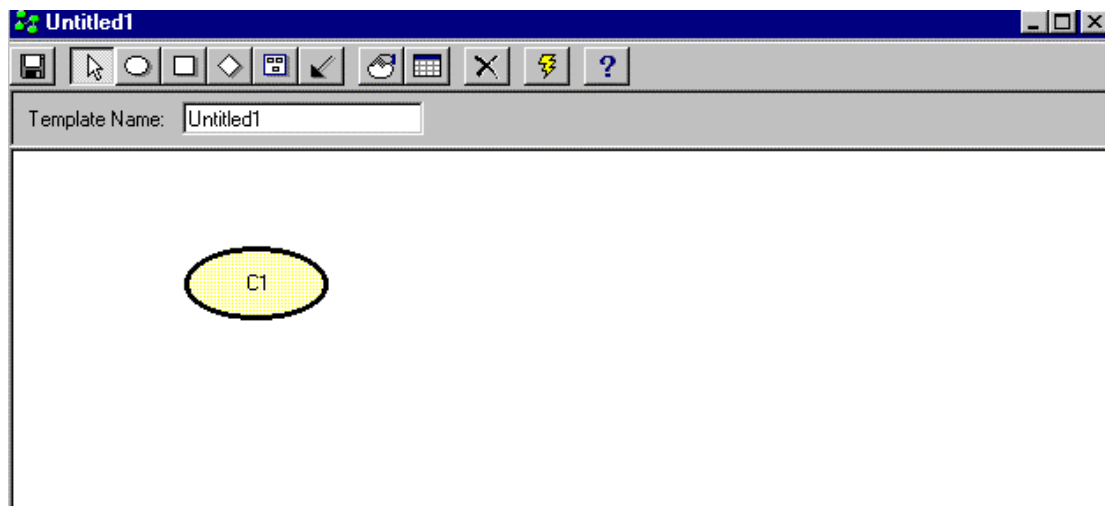


2. To create a node in the new template, select the oval shaped node icon from the template tool bar. You will be asked to select the preferred kind of node and then press OK in a dialogue box, as shown below.

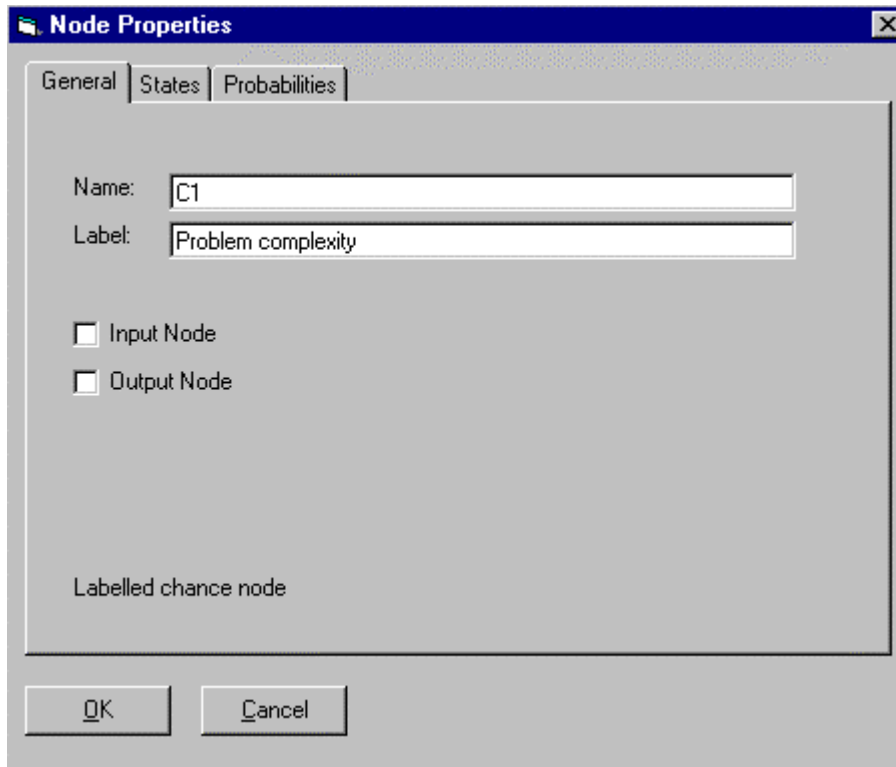




Now click inside the template and the node will appear.

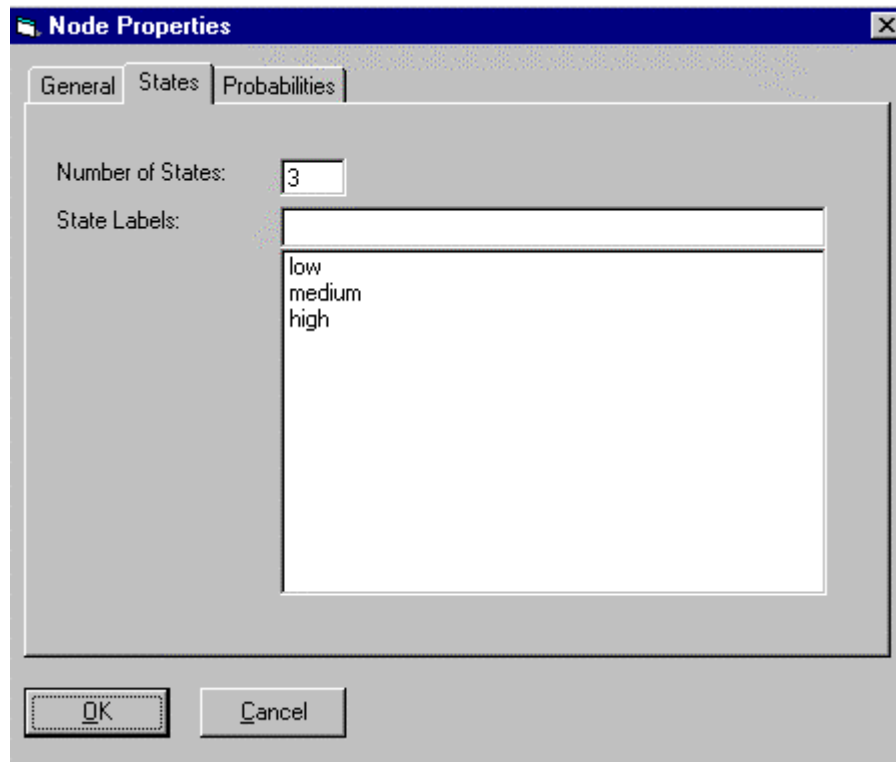


Double clicking on this node, presents you with the Node Properties window below. (Alternatively, you can select “Node Properties” from the Serene tool Edit menu).



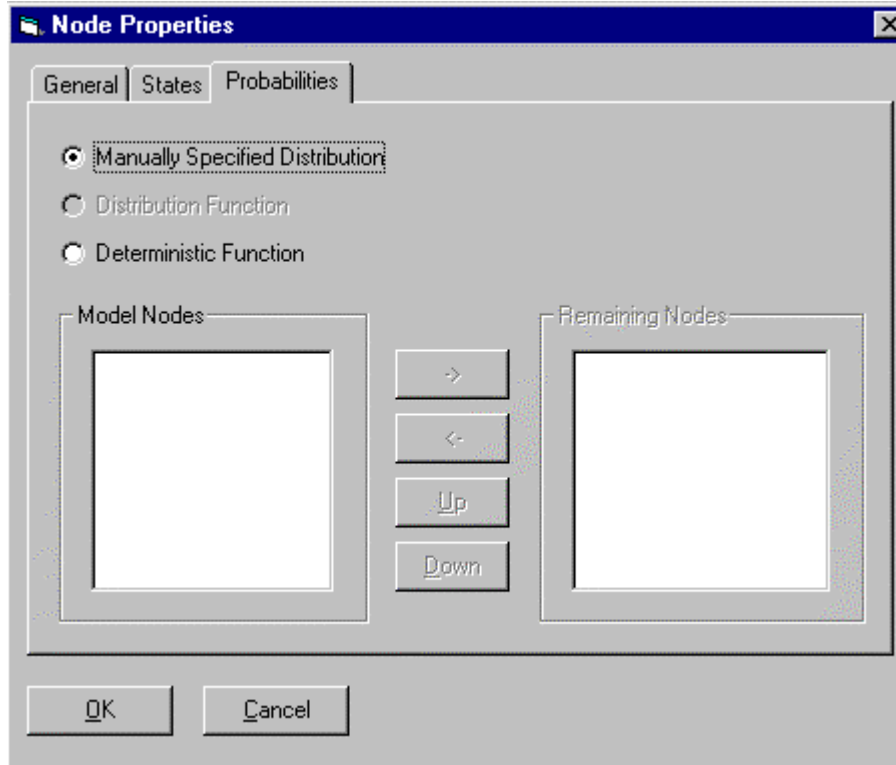
First, give the node an appropriate label. In the diagram above, this is shown as Problem complexity. You can also set the node to be an input or output node of the template to which it belongs.


3. To enter the states of the nodes, select the States option. You will be asked to enter the number of states that node has and if you have chosen the discrete labelled option, you should name them, as below:

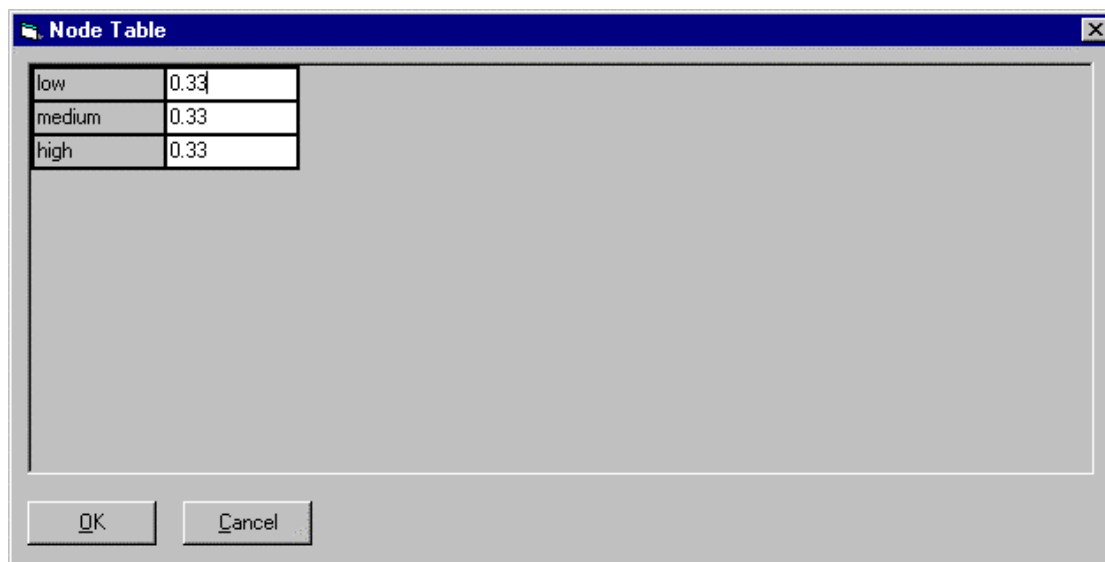


For the discrete numeric option you will be asked to insert values instead of giving names. For the continuous option you will be asked for the appropriate interval end points. Examples of each of these follow in the next section on the Defects example.

4. To enter the probabilities, select the probabilities option from the node properties dialogue box. You can select to manually specify the distribution or use a function.





For manually specifying the probabilities, you should then select the  icon from the template tool bar (or Node Table from the SERENE tool edit menu):



and enter the probabilities as required.

The Defects example in the next section describes the alternative of providing the probabilities by using a function.

5. Repeat stages 2-4 to create more nodes.
6. To create the links between the nodes, select the arrow icon  from the template tool bar. Click in one of the nodes to be joined, moving the mouse with the button still depressed and only releasing it once the cursor is positioned inside the connecting node.
7. To run the BBN, select the lightening icon  from the template tool bar.

## 5.2 Defects Example

The objective of this example is to build a BBN from idioms that enables us to make predictions about the likely number of residual defects that exist in a system on the basis of evidence about:

1. discovered defects
2. various aspects of the testing process

### 5.2.1 Step 1: Create instantiation of the definition idiom

We are first going to create an instantiation of the definition idiom which will capture the non-controversial definition:

$$residual\_defects = inserted\_defects - discovered\ defects$$

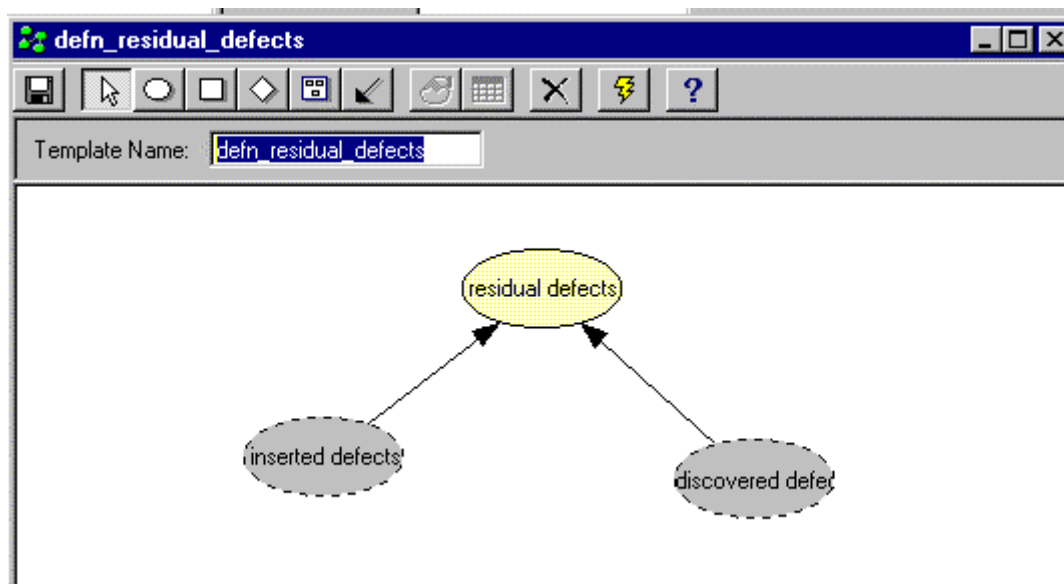
We will call the associated template: defn\_residual\_defects.

There are three main substeps you need to follow:

1. Create the nodes with the appropriate state values
2. Create the NPTs
3. Make the relevant nodes import/export nodes.

#### 1. Create the nodes with the appropriate state values

We are going to create the template that looks like:

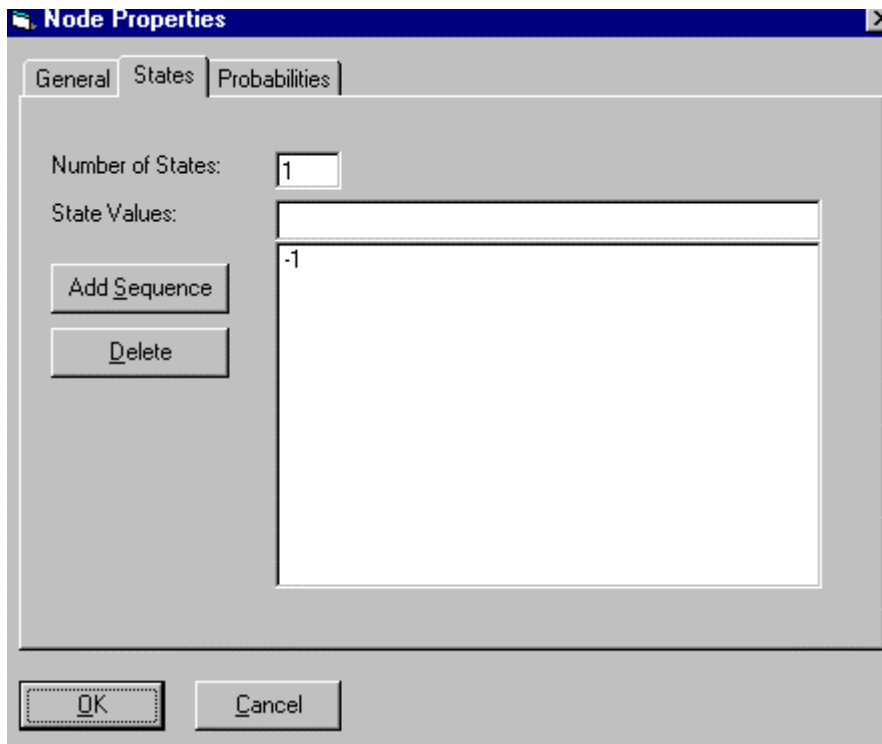


To create this template, first create three chance nodes which are all “Discrete Numeric” and add the arcs as shown. All of the three nodes are going to have 50 states:

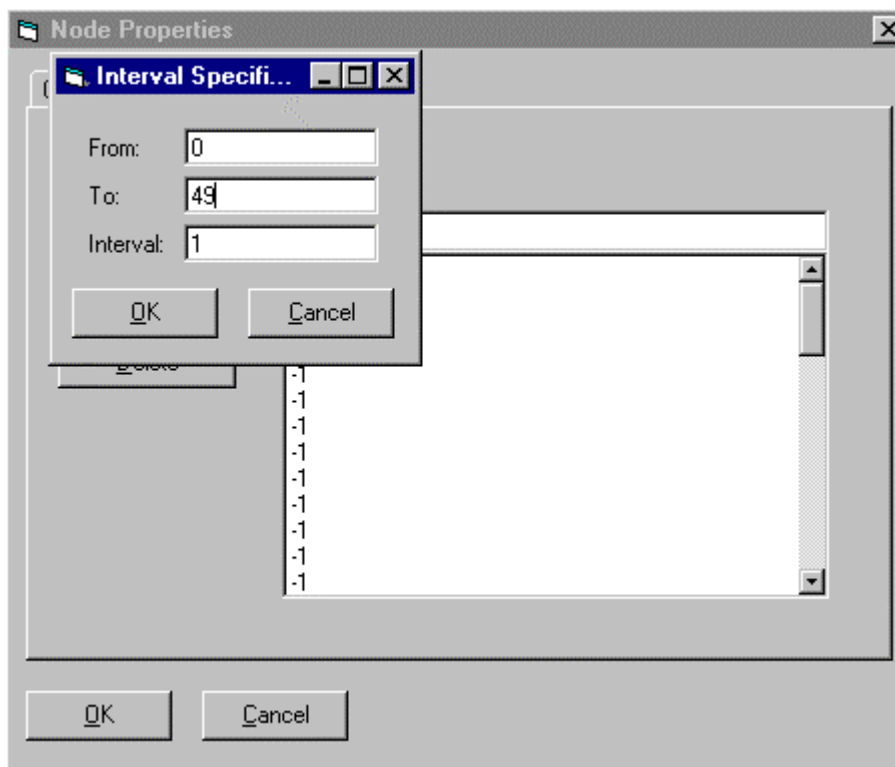
$$0,1,2,3,\dots,48,49$$

Now you could, of course, enter these states by hand, but there is in fact a very simple way to do it.

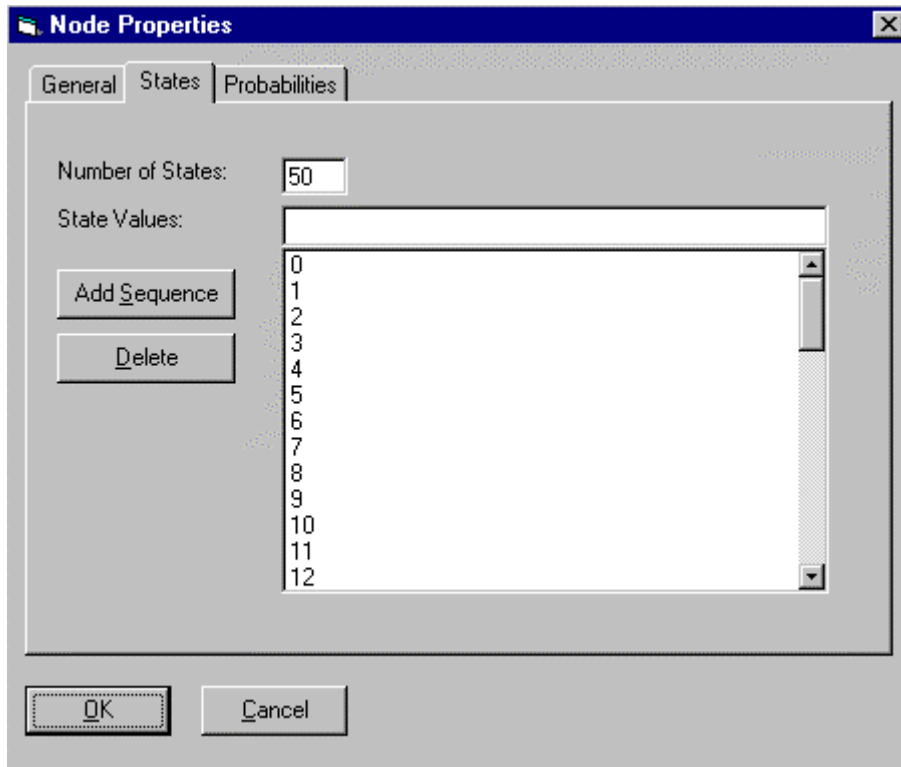
First select one of the nodes (say, “residual defects”) by double clicking on it to bring up the Node Properties window. Click the “States” tab in Node Properties:



Next, click on the “Add Sequence” button. Another small window will pop up on top as shown here:



Enter the values shown above in the “From” and “To” boxes (leave the “Interval” value to 1) and click the OK button. This will produce the states 0,1,2,...49 as shown here:



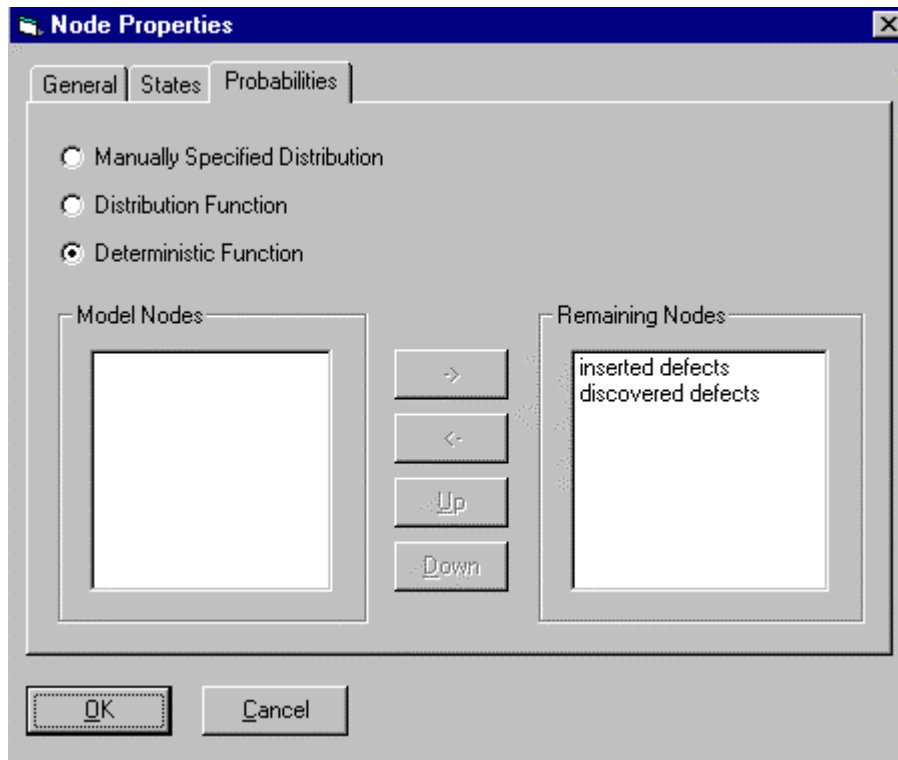
Now the node “residual defects” has the 50 states as required.

Repeat this process for each of the other two nodes.

## 2. Create the NPTs

We only need to worry about the NPT for the node “Residual defects”.

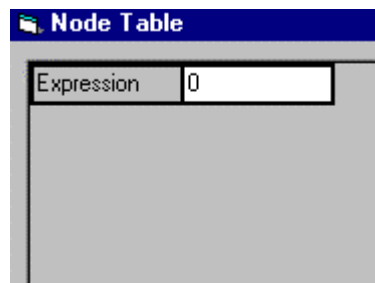
We are going to define the NPT by a deterministic function. To do this open the Node Properties window for the node “Residual defects”. Click the Probabilities tab and select “Deterministic Function”:



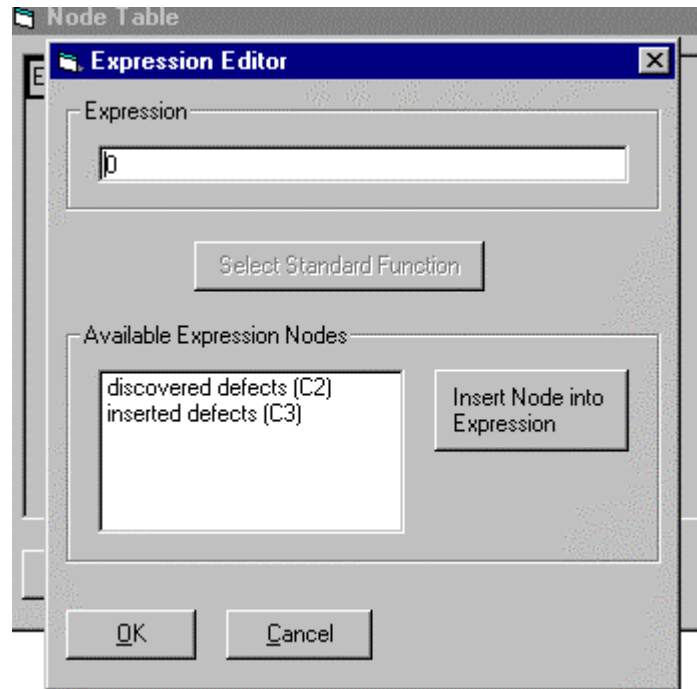
Now close the Properties window, select the node "residual\_defects" by clicking it once and then click the probabilities icon in the toolbar



This will bring up the node table window:



You are now going to enter an expression (rather than the default "0"). You can enter text directly into the box where the "0" is, but it is best to get help by calling up the Expression Editor. To do this double-click on the test field where the 0 is and the Expression Editor window appears:



The expression we want to enter is the one captures the function

$$\textit{inserted\_defects} - \textit{discovered\_defects}$$

If you double click on the available expression node “inserted defects” the label C3 appears in the expression (this is what you want since it refers to the variable “inserted defects”). If you next enter a minus sign “-” and then click on “discovered defects” you will get the expression

$$C3-C2$$

In principle this seems to be what you want, but there is a snag. If you tried to compile this template you would get errors. This is because the tool does not know that, in practice, C3 is always at least as big a number as C2. It looks at the expression C3-C2 and computes all the possible values it can take. Since both variables range from 0 to 49, the expression C3-C2 can in principle take on values which are negative. But the node “residual defects” does not have negative values. It only has values 0 to 49. Thus we have to somehow tell the tool that the value of the expression C3-C2 can never be less than 0.

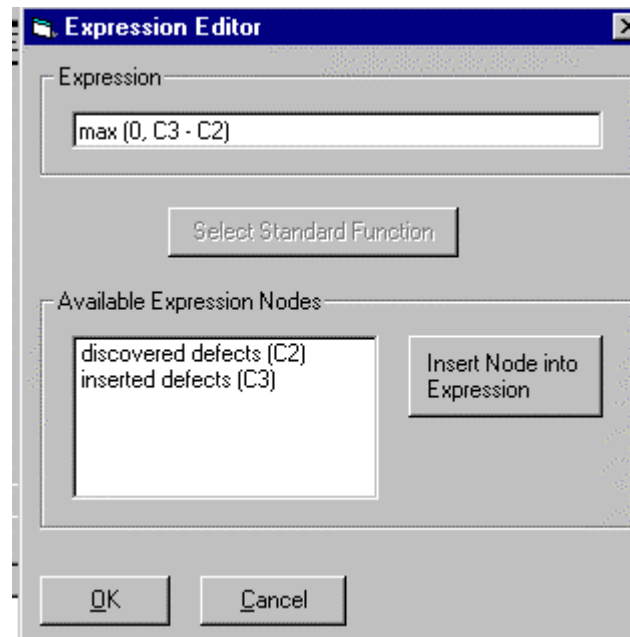
Fortunately, there is an easy way to do this because the expression editor has standard function built in. One of these functions is the max function.

So, edit the expression as:

$$\text{max}(0, C3-C2)$$

(this defines a function which is the same as C3-C2 except when C2 is bigger than C3 in which case it will return 0). The expression editor window should look like:

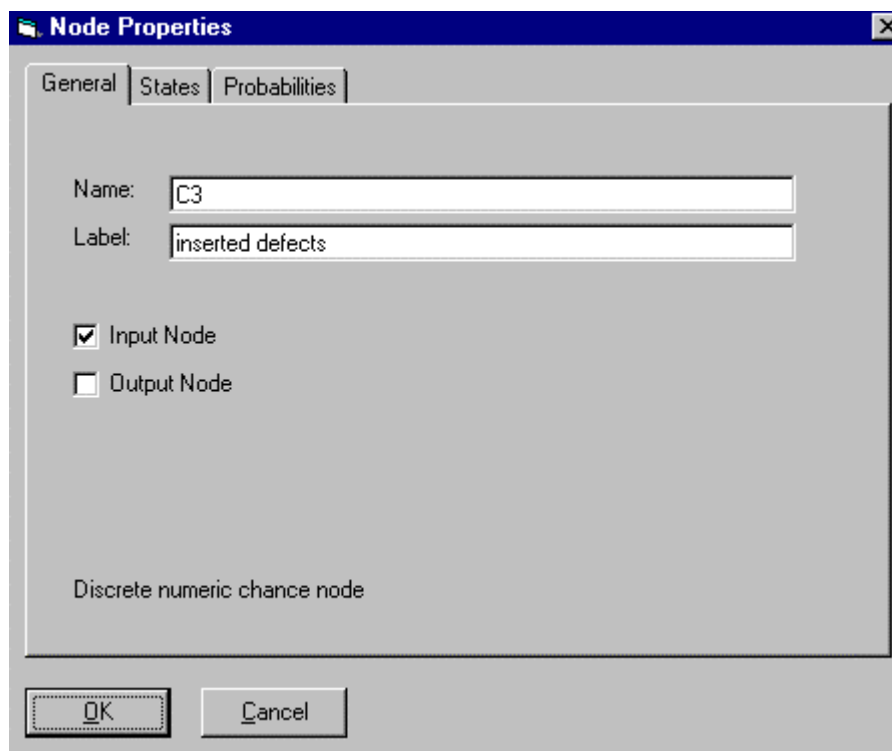




Press OK.

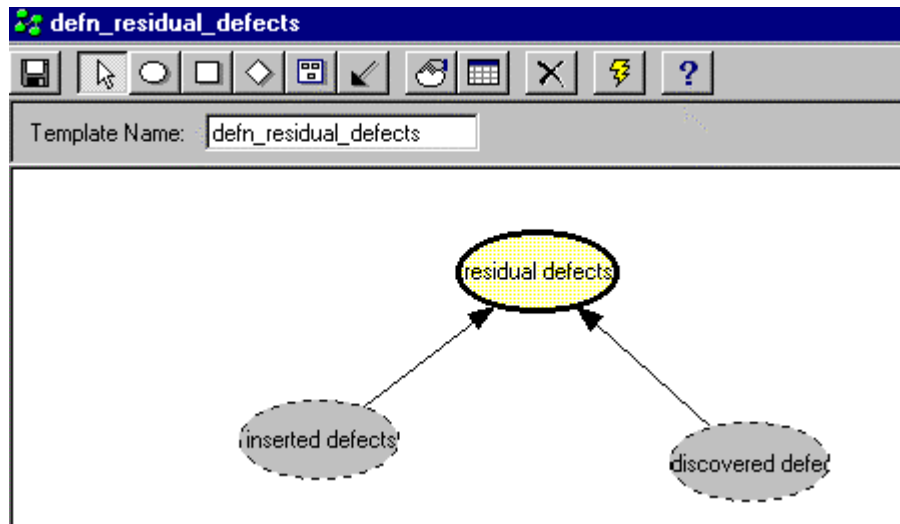
### 3. Make the relevant nodes input/output nodes


We want some of the nodes of this template to be ‘joined’ to other templates. Specifically, the nodes “discovered defects” and “inserted defects” will be joined. We therefore need to make each of these nodes “input nodes”. To do this, open up the Node Properties window for each node in turn and select “input node” as shown:



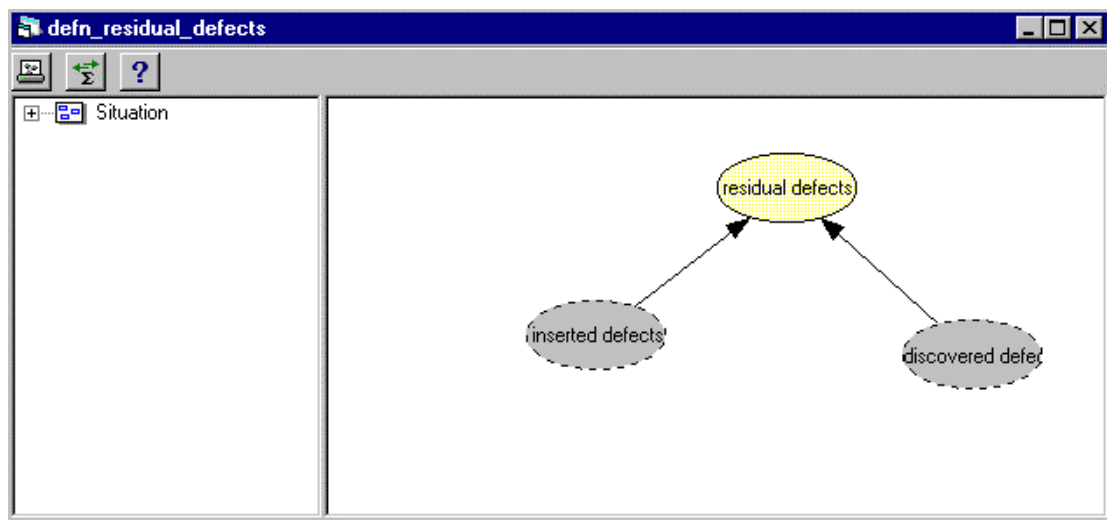
Once you press OK the node will change colour to denote that it is now recognised as an input node.

Finally, make sure you save both the template file and name it with the same name as shown:

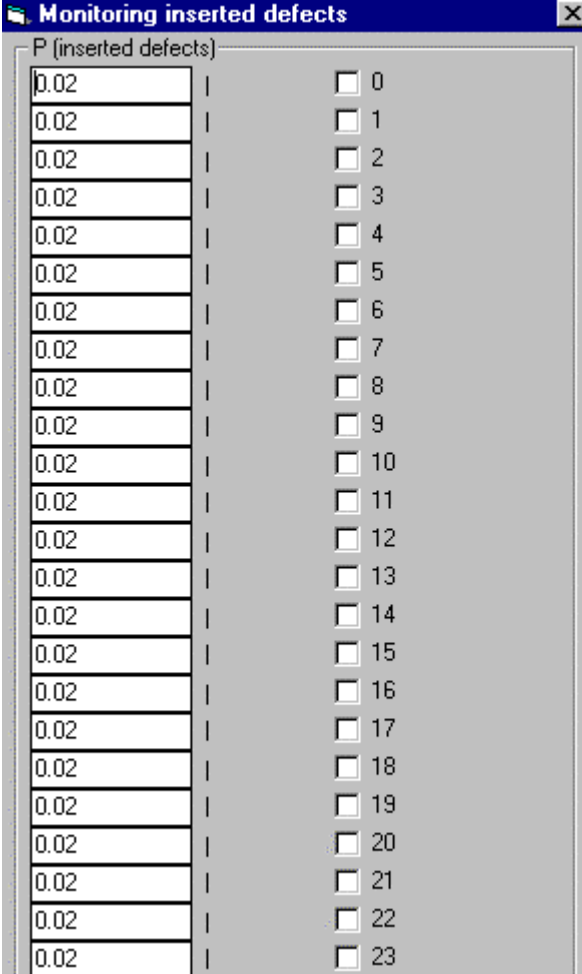


At this point you can ‘test out’ the template to see how it runs, by clicking the lightning icon  in the toolbar.

There will be a delay as the template compiles (a lot of processing goes on to compute all the values of a very large NPT). Then you should see the following display in a new window:



Double-clicking on, say the “inserted defects” node will unveil a window showing the probability distribution for this node (the diagram below only shows part of the distribution as it has too many states to fit into one screen):



The image shows a dialog box titled "Monitoring inserted defects" with a close button (X) in the top right corner. Below the title bar, the text "P (inserted defects)" is displayed. The main area of the dialog contains a list of 24 rows. Each row consists of a text input field on the left and a checkbox on the right. All input fields contain the value "0.02". The checkboxes are numbered from 0 to 23, corresponding to the rows. The dialog box has a standard Windows-style border and a grey background.

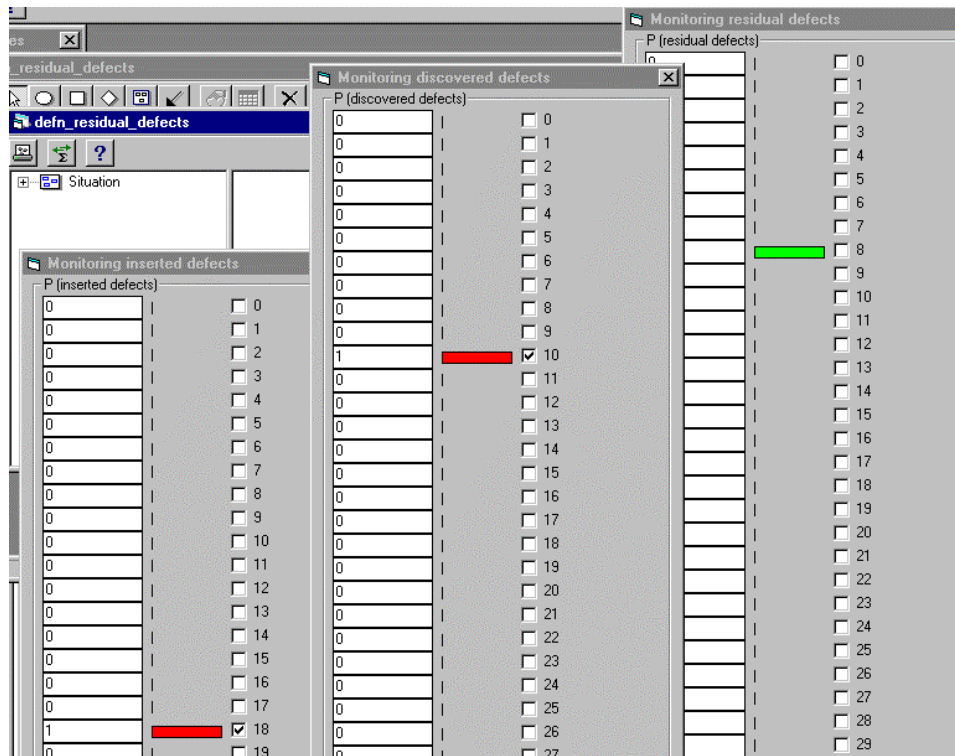
Value	Index
0.02	0
0.02	1
0.02	2
0.02	3
0.02	4
0.02	5
0.02	6
0.02	7
0.02	8
0.02	9
0.02	10
0.02	11
0.02	12
0.02	13
0.02	14
0.02	15
0.02	16
0.02	17
0.02	18
0.02	19
0.02	20
0.02	21
0.02	22
0.02	23

Open up the probability windows for each of the nodes. You are now ready to enter evidence and propagate it.

Click the value "19" for inserted defects, "11" for discovered defects and then click the button:

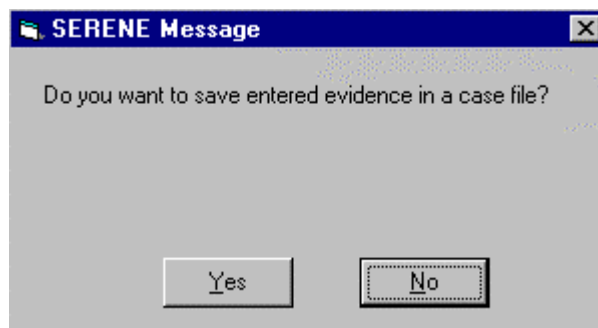


This will propagate the values and you will see the computed value of residual defects appear.



You can ‘retract’ evidence by simply clicking on an already selected value (the tick will be cleared). You can enter new evidence and propagate again. You can tick as many values as you like in any probability window. So, if for example, if you believe the inserted defects is a number between 5 and 10 but don’t know which you can select all of those values to reflect your uncertainty.

When you exit the compiled template window you will be presented with the message:

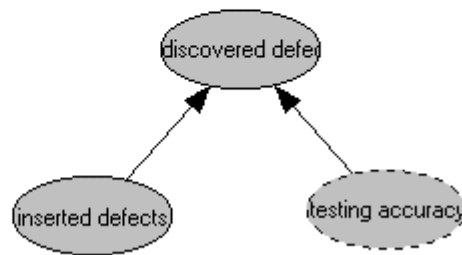


This enables you to save the evidence you entered as a case file if you wish.

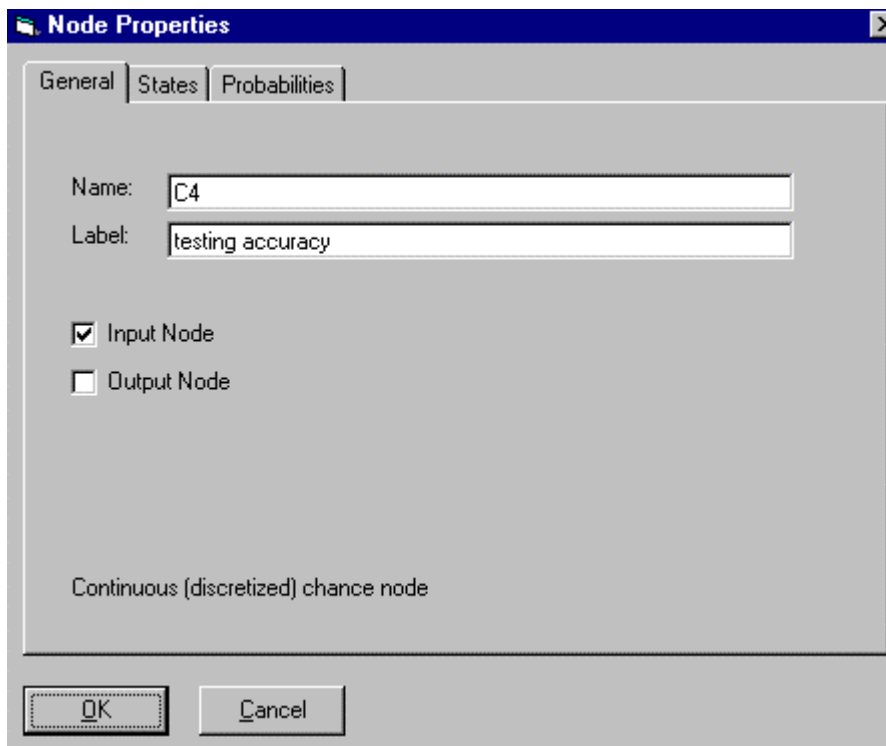
### 5.2.2 Step 2: Create an instantiation of the measurement idiom

Next we are going to create an instantiation of the measurement idiom which reflects the fact that the number of discovered defects is dependent not only on the number of inserted defects, but also on the testing accuracy.

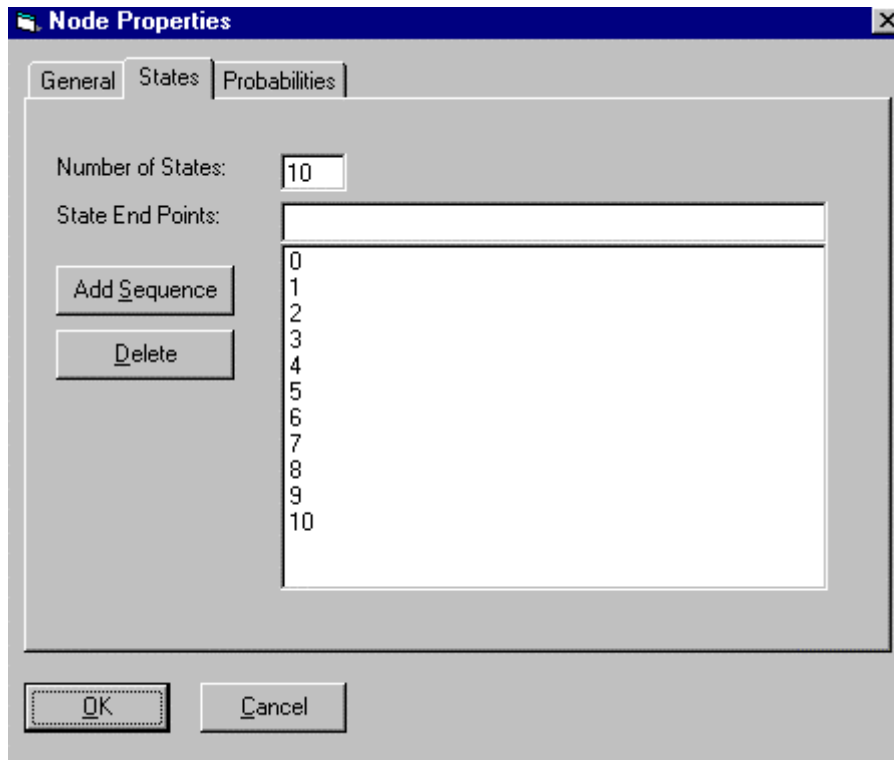
Create a new template “measurement\_testing” as shown with three nodes and two arcs.



The two nodes “discovered defects” and “inserted defects” should have exactly the same name and properties as the corresponding nodes in the previous template (i.e. discrete numeric with 50 states 0,...,49). The node “testing accuracy” is going to be an input node that will actually be properly defined in another template.



We want this node to take on values 0,1,...,10. For reasons that will be explained below, we need to make the node type continuous rather than discrete. To get the appropriate state values click on the “States” tab and use the “Add Sequence” button as before to create the states as shown:

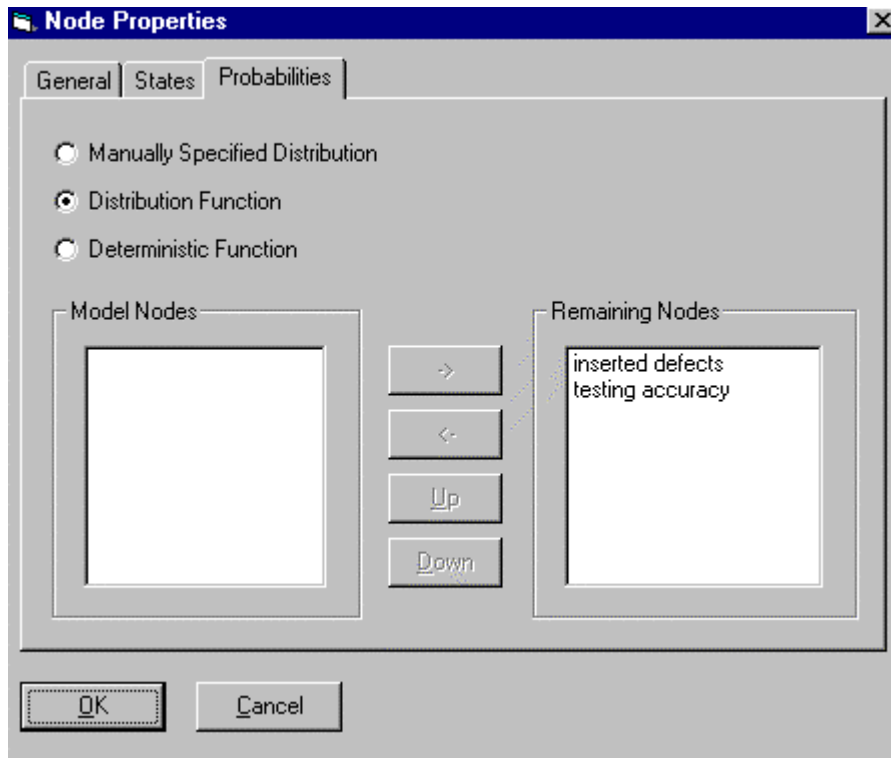


It is important to note at this point that the states here are NOT 0,1,2,...,10, but rather the 10 intervals:

]0,1], ]1,2], ..., ]9,10].

The actual values for this node can be thought of as a 10 point scale for testing accuracy where 0 means totally inaccurate and 10 means perfectly accurate.

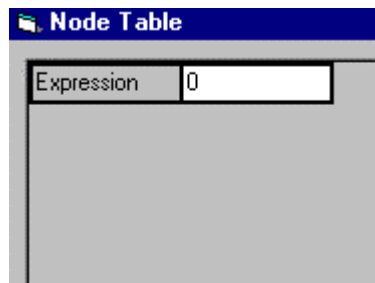
Next we need to define the NPT for the node “discovered defects”. Select the Properties window for this node and select “Distribution Function”.



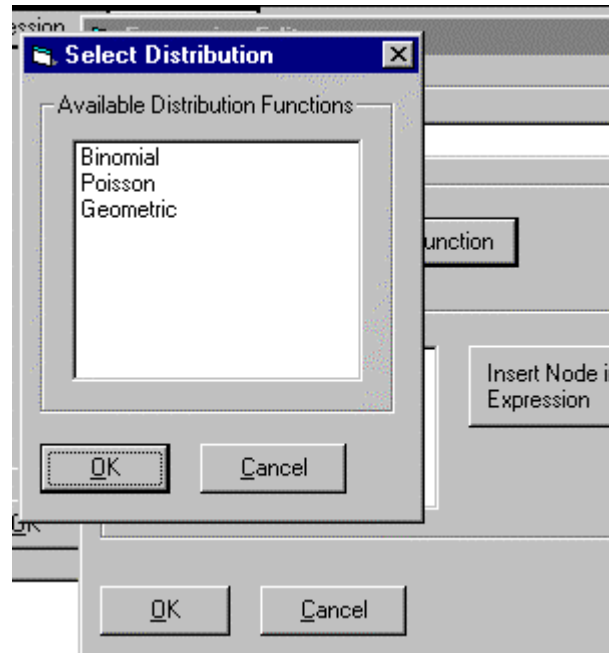
Press OK (which closes the Properties window), and select the node “discovered defects” by clicking it once. Click the probabilities icon in the toolbar



This will bring up the node table window:



Double-click on the text box (which has the default value “0”) and you will be provided with a list of available distribution functions to select:



We want the Binomial distribution, since it is reasonable to assume that the number of discovered defects is binomially  $B(n,p)$  distributed with

$n$  = number of inserted defects; and

$p$  = probability of finding defect

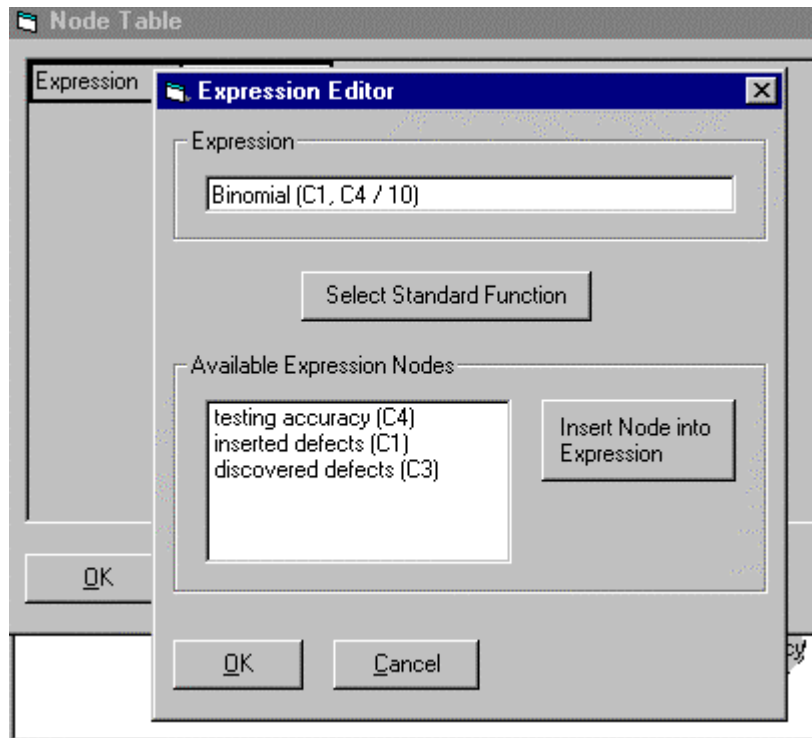
Now, we the number of inserted defects is a value already available to us, since it is one of the nodes. For the value  $p$  we are going to use the “testing accuracy” node. Since this node takes on values 0 to 10 where 0 is totally inaccurate and 10 is perfectly accurate, it is reasonable to take

$$p = (\text{testing accuracy}) / 10$$

In other words when testing is perfect the probability of finding a fault is 1 and when it is terrible the probability of finding a fault is 0.

Thus in the expression editor we enter:



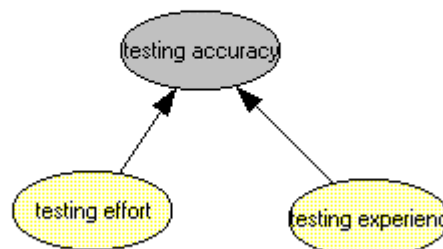


This completes the measurement\_testing template.

### 5.2.3 Step 3: Create instantiation of definition idiom: testing accuracy


Now we create a template called “testing\_accuracy” which is an instantiation of the definition idiom. We are effectively defining testing accuracy as a (deterministic) function of testing effort and testing experience.

Create the nodes and arcs as shown:

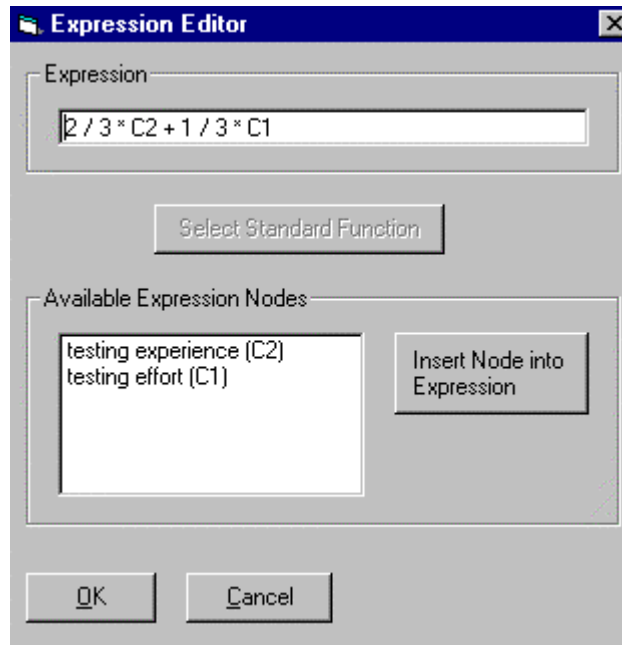


Make sure the node “testing accuracy” has exactly the same properties as in the previous idiom instantiation. Create two new nodes, “testing effort” and “testing experience” with the same properties, i.e. each as continuous nodes with 10 states.

Open the Properties window for “testing accuracy” by double-clicking it and select the Probabilities tab. Set the value to “Deterministic function”.

Close the Properties window, select the node “testing accuracy” and double click the probabilities icon  in the toolbar.

Enter the expression shown (this simply says that testing accuracy is a weighted sum of testing experience and testing effort, with experience counting twice as much as effort).



This complete the third and final basic template for the example. Next we are going to use these three templates to build a more complex BBN.


#### 5.2.4 Step 4: Combining the templates

You should now have created three idiom instantiations which are stored as the templates:

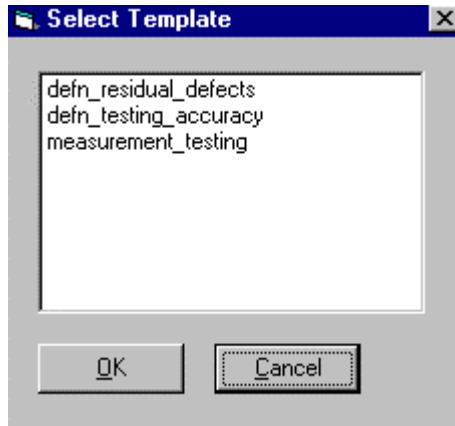
- defn\_residual\_defects
- measurement\_testing
- defn\_testing\_accuracy

We are now going to build a bigger net by combining these.

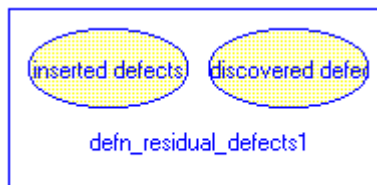
First create a new template and call it “entire\_testing”.

The ‘nodes’ in this template are going to be *abstract nodes* representing the templates we have already created. To insert a template as an abstract node click on the template icon  in the toolbar.

You will then be presented with a list of open templates:

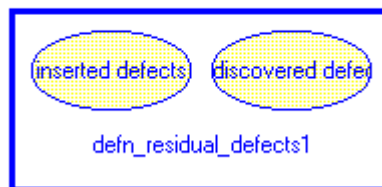
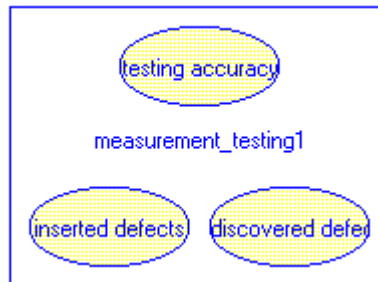


Select *defn\_residual\_defects* and click OK. Then when you click anywhere in the template window the following appears.

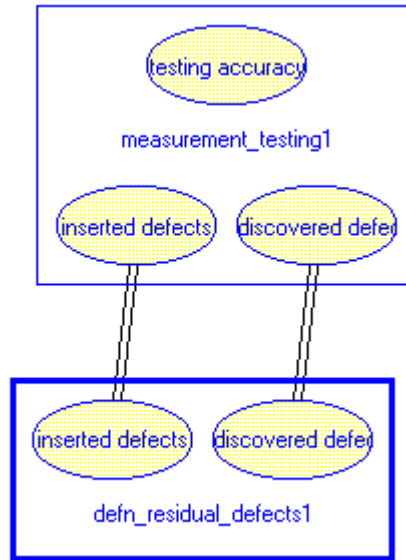


This shows that two of the nodes (“inserted\_defects” and “discovered\_defects”) which were defined as input nodes are available to be joined to other appropriate templates.

Next do the same again but select the *measurement\_testing* template. Then when you click anywhere in the template window you will now see the second template appear. Move the two templates around so they appear as:

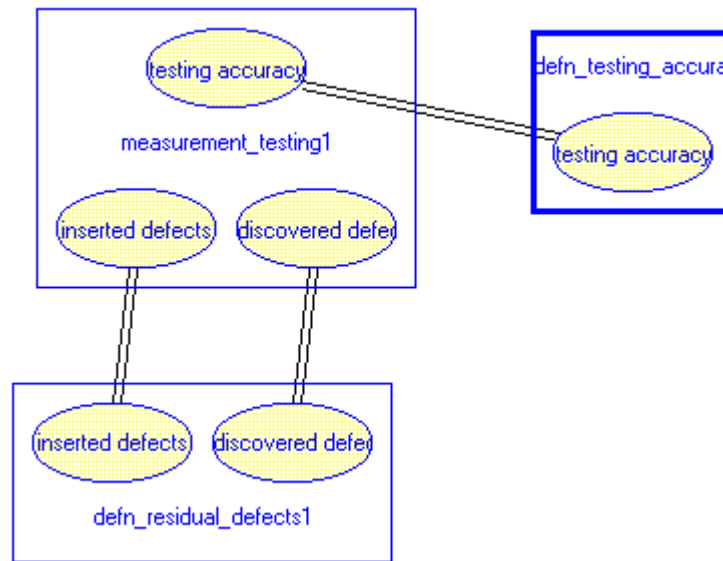


Next select the arc icon and connect the two nodes labelled “inserted defects”. Do the same to the two nodes labelled “discovered defects”. This produces:



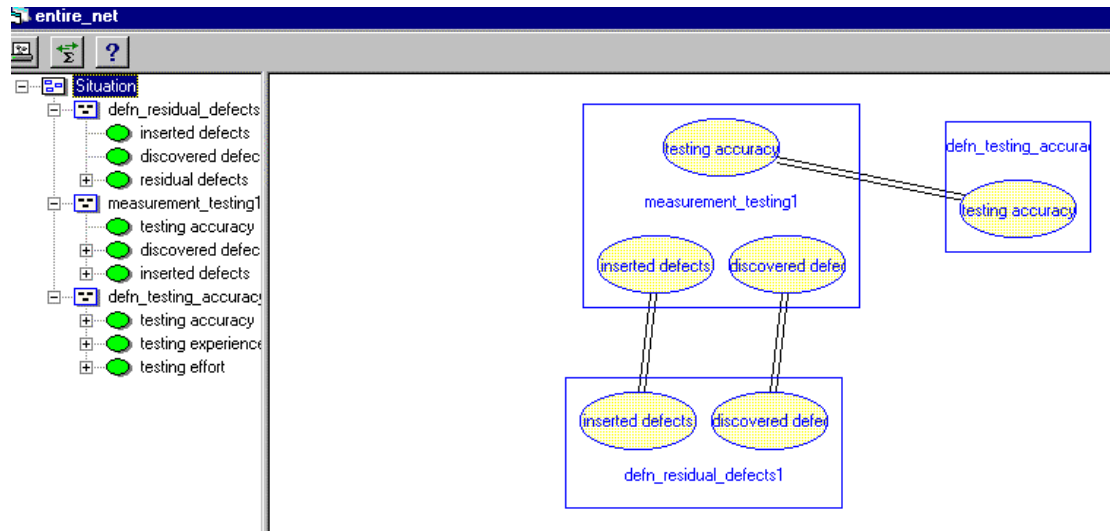
Now select the third template node “defn\_testing\_accuracy” and join the two nodes labelled “testing accuracy”.

Finally you will end up with:



This net is now ready to be compiled.

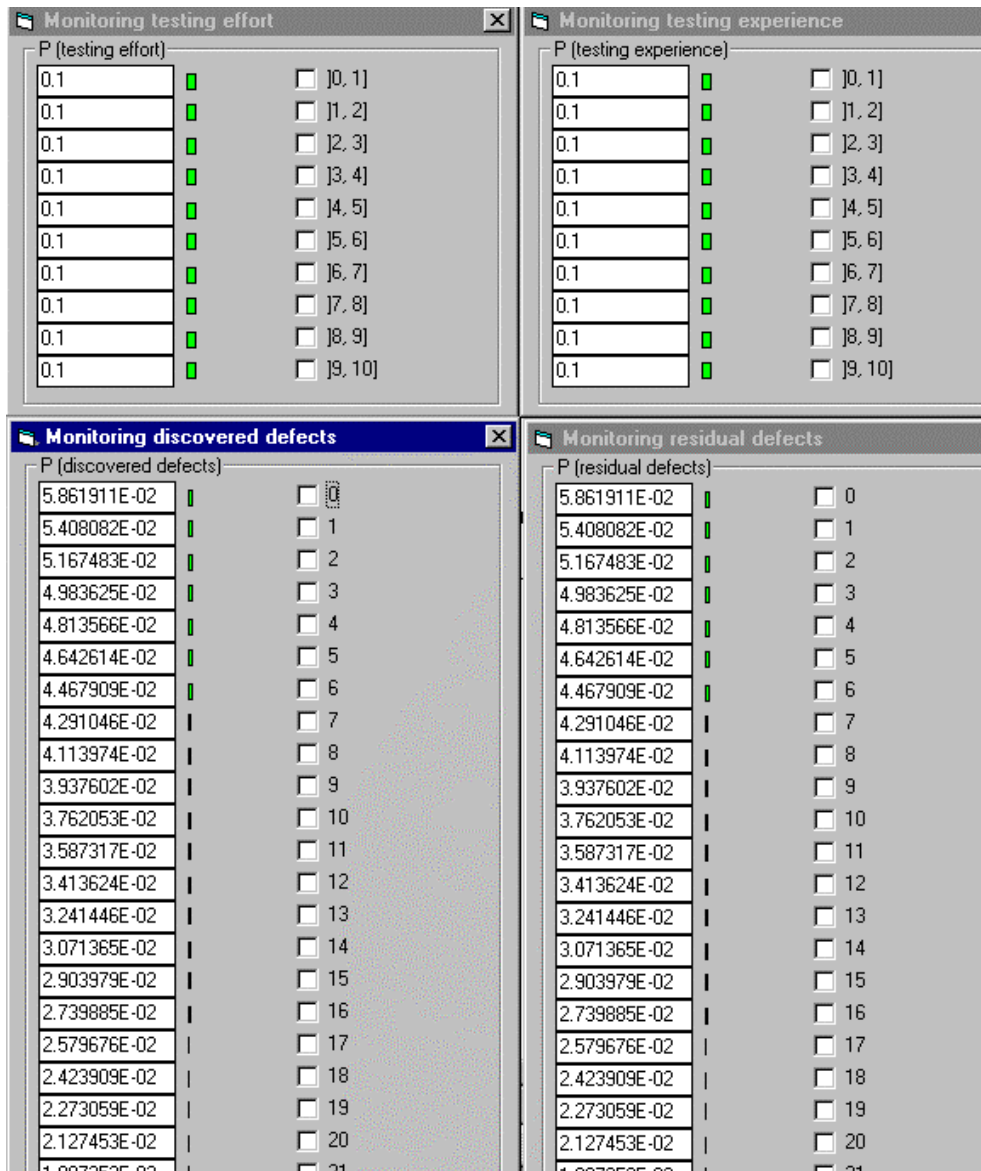
After compiling, clicking on the directory icons in the left hand pane reveals all the nodes of the net including those that are not visible in the main window.



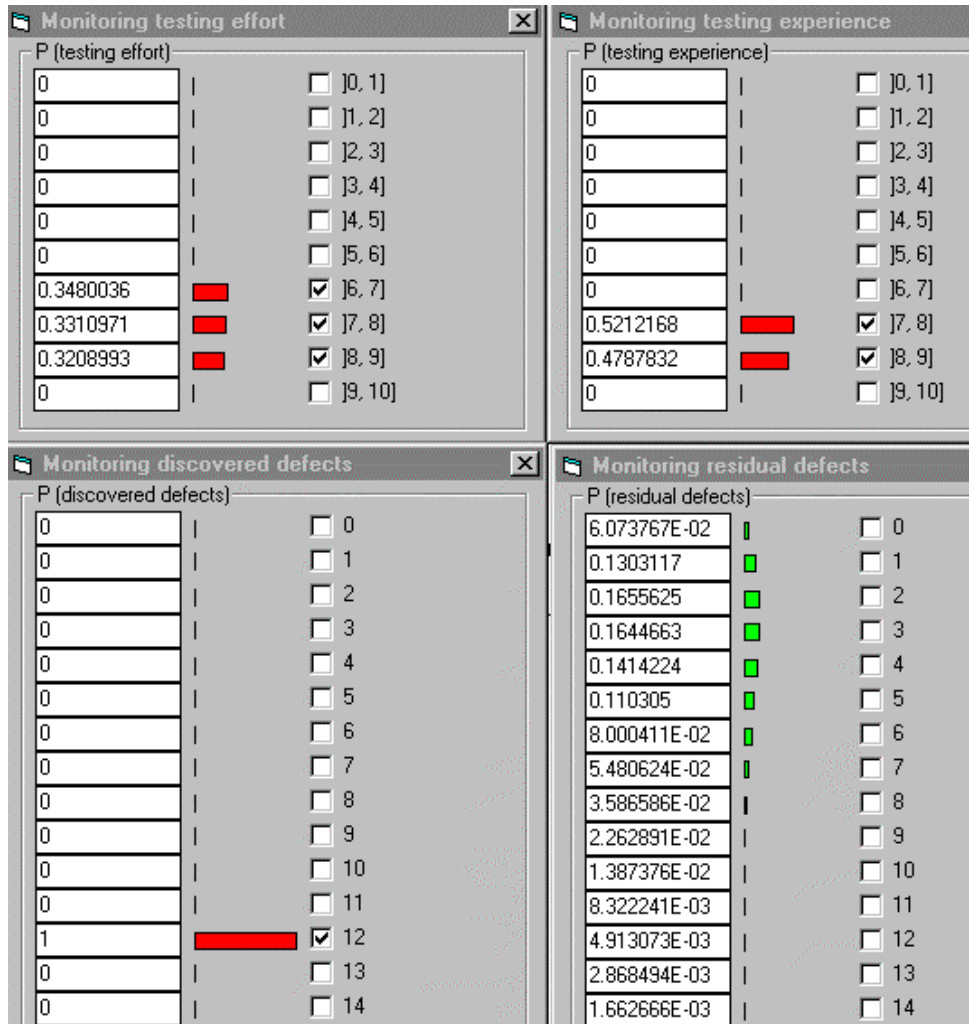
By clicking on the templates and then double-clicking the relevant nodes in them you can display the probability monitors for any nodes that you are interested in. In this example we are interested in:

- the number of residual defects (this is the thing we don't know but want to predict)
- those nodes for which we have evidence. We assume these to be “discovered\_defects”, “testing\_effort” and “testing\_experience”.

So, get the following monitors arranged on the screen:



Now you can start entering evidence for the nodes you have information about and see its effect on the thing you want to predict “residual\_defects”. In the following example we enter evidence which shows we believe the testing effort is pretty high and the testing experience is good. We find 12 defects. Once we propagate this we get:



This gives us a prediction for residual defects. Specifically, the value is almost certainly between 0 and 8 with values 2 and 3 being most likely.

## 6 References

1. Alpert M & Raiffa H In: Kahneman D Slovic P & Tversky A (eds)  
Judgement under Uncertainty: Heuristics and Biases  
Cambridge University Press, (1982)
2. Bache R, Mullerburg M  
Measures of testability as a basis for quality assurance  
Software Eng J Vol 5 (2), 86-92, (1990)
3. Basili VR and Rombach HD, The TAME project: Towards improvement-oriented software environments, IEEE Transactions on Software Engineering 14(6), pp 758-773, 1988.
4. Basili V R and Rombach H D  
The TAME project: Towards Improvement Oriented Software Environments  
IEEE Trans. Soft. Eng. 14(6), pp 758-773.GQM, (1988)
5. Bennett P A  
Software development for the channel tunnel: a summary  
High Integrity Systems 1 (2), 213-220 (1994)
6. Bertolino A and Marre M  
How many paths are needed for branch testing?  
J Systems and Software (1995)
7. Brocklehurst S and Littlewood B  
New ways to get accurate software reliability modelling  
IEEE Software, July
8. Bunge M  
Causality and Modern Science  
Dover Publications, New York (3<sup>rd</sup> Revised Edition) (1979)
9. Chapman I & Chapman J  
Illusory Correlation as an obstacle to the use of valid psychodiagnostic signs  
Journal of Abnormal Psychology 74, 271-280 (1969)
10. DATUM  
Dependability Assessment of safety critical systems through Unification of Measurable evidence  
EPSRC/DTI Safety Critical Systems Programme, (Project IED/1/9314) (1993)
11. Fenton N E and Pfleeger S L  
Software Metrics: A Rigorous and Practical Approach (2<sup>nd</sup> Edition)  
International Thomson Computer Press, (1996)
12. Fenton N E  
How to improve safety-critical standards, in 'Safer Systems' (Ed: Redmill F and Anderson T)  
Proceedings 5th Annual Safety Critical Systems Symposium, pages 96-111, (1997)
13. Fenton N E  
Multi-criteria Decision Aid; with emphasis on its relevance in dependability



- assessment  
DATUM/CSR/02, CSR (1995)
14. Fenton N E  
Software measurement: a necessary scientific basis  
IEEE Trans Software Eng 20 (3), 199-206 (1994)
  15. Fenton N E  
Software Metrics: A Rigorous Approach  
Chapman and Hall (1991)
  16. Fenton N E  
When a software measure is not a measure  
Softw Eng J 7(5), 357-362, (1992)
  17. Fenton N E, Iizuka Y, and Whitty RW (Editors)  
Software Quality Assurance and Metrics: A Worldwide Perspective  
International Thomson Computer Press, (1995)
  18. Fenton NE and Pfleeger SL, Software Metrics: A Rigorous and Practical Approach  
(2nd Edition), International Thomson Computer Press, 1996.
  19. Finkelstein L  
A review of the fundamental concepts of measurement  
Measurement Vol 2(1), 25-34., (1984)
  20. Fischhoff B, Slovic P, Lichtenstein S  
Knowing with certainty: The appropriateness of extreme confidence  
Journal of Experimental Psychology: Human Perception and Performance, 3. 552-564  
(1977)
  21. Halstead M  
Elements of Software Science  
North Holland, (1977)
  22. Hamer P, Frewin G, Halstead's  
software science: a critical examination  
Proc 6th Int Conf Software Eng, 197-206, (1982)
  23. Hamlet D and Voas J  
Faults on its sleeve: amplifying software reliability testing  
Proc ISSTA '93, Boston, pp 89-98, (1993)
  24. Hatton L  
C and Safety Related Software Development: Standards, Subsets, testing, Metrics,  
Legal issues  
McGraw-Hill, (1994)
  25. Hatton L  
Static inspection: tapping the wheels of software  
IEEE Software, May, 85-87, (1995)

26. Hatton, L., & Hopkins, T. R  
Experiences with Flint, a software metrication tool for Fortran 77  
Symposium on Software Tools, Napier Polytechnic, Edinburgh, (1989).
27. Hogarth R.M  
Cognitive Processes and the assessment of subjective probability distributions (with discussion)  
Journal of American Statistical Association, 70. 271-294 (1975)
28. International Electrotechnical Commission (IEC)  
Draft European Standard IEC 61508 - Functional Safety : Safety Related Systems (in 7 parts), Ref. 65A/179/CDV to 65A/185/CDV,  
IEC Geneva, June (1995)
29. Kahneman D. Slovic P. & Tversky A.(eds)  
Judgement under Uncertainty: Heuristics and Biases  
Cambridge University Press, (1982).
30. Keller T  
Measurements role in providing 'error-free' onboard shuttle software  
3rd Intl Applications of Software Metrics Conference, La Jolla, California, pp 2.154-2.166, Proceedings available from Software Quality Engineering, (1992).
31. Laprie J-C (ed)  
Dependability: basic concepts and terminology  
Springer Verlag, (1992).
32. Laprie J-C (ed), Dependability: basic concepts and terminology, Springer Verlag, 1992.
33. Leveson NG and Turner CS  
An investigation of the Therac-25 accidents  
IEEE Computer, July, 18-41, (1993).
34. Littlewood B and Strigini L  
Validation of Ultra-High Dependability for Software-based Systems  
CACM, vol. 36, no 11, (1993).
35. Littlewood B  
Limits to evaluation of software dependability, in Software Reliability and Metrics (Ed Littlewood B, Fenton N)  
Elsevier, (1991).
36. Lytz R  
Software metrics for the Boeing 777: a case study  
Software Quality Journal, 4(1), 1-14, (1995).
37. McCabe T  
A Software Complexity Measure, IEEE Trans  
Software Engineering SE-2(4), 308-320, (1976).
38. Meyer M. & Booker. J  
Eliciting and Analyzing Expert Judgement. A Practical Guide.  
Academic Press, (1991).

39. MISRA  
The Motor Industry Software Reliability Association, Development Guidelines for Vehicle Based Software, November (1994)
40. Murphy A.H. & Winkler R.  
Probability forecasting in meteorology  
Journal of the American Statistical Association. 79. 489-500. (1984).
41. Myers, Glenford J  
Reliable Software through Composite Design  
Van Nostrand Reinhold, (1975).
42. Neil M. and Fenton N  
Predicting software quality using Bayesian belief networks  
Proceedings of the 21st Annual Software Engineering Workshop at NASA Space Flight Centre, Washington, USA, 4-5 December (1996).
43. Neil M. and Fenton N  
Software Defect Density Modelling  
Proceedings of the Applications of Software Metrics Conference in Atlanta, USA, November 11-15, (1997).
44. Neil, M.D.  
Multivariate Assessment of Software Products  
Journal of Software Testing, Verification and Reliability, Vol 1(4), pp 17-37, (1992).
45. Nejme, BA  
NPATH: A measure of execution path complexity and its applications  
Comm ACM, 31(2), 188-200, (1988).
46. Oulsnam G  
Cyclomatic numbers do not measure complexity of unstructured programs  
Inf Processing Letters, 207-211, Dec, (1979).
47. Oviedo EI,  
Control flow, data flow, and program complexity,  
Proc COMPSAC 80, IEEE Computer Society Press, New York, 1980, 146-152,  
(1980).
48. Pearl J  
Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufman, (1988).
49. Peterson, C. & Beach, L.  
Man as an intuitive statistician  
Psychological Bulletin, 68 (1), 29-46 (1969)
50. Riley P  
Towards safe and reliable software for Eurostar  
GEC Journal of Research 12 (1), 3-12, (1995).
51. Roberts FS, Measurement Theory with Applications to Decision Making, Utility, and the Social Sciences, Addison Wesley, 1979.

52. Roberts FS,  
Measurement Theory with Applications to Decision Making, Utility, and the Social Sciences  
Addison Wesley, (1979).
53. Saaty T  
The Analytic Hierarchy Process  
McGraw Hill, New York, (1980).
54. Saaty T, The Analytic Hierarchy Process, McGraw Hill, New York, 1980.
55. Shaw, R,  
System Integrity and Risk Management Department, Lloyd's Register, 'Safety Cases - How Did We Get Here?'  
Proceedings of 'Safety and Reliability of Software-based Systems, 12th Annual CSR Workshop, September 1995, Springer-Verlag, (1997).
56. Shepperd MJ  
A critique of cyclomatic complexity as a software metric  
Softw. Eng. J. vol 3 (2), pp 30-36, (1988).
57. Spetzler C.S. & Stael von Holstein C S.  
Probability encoding in decision analysis  
Management Science, 22. 340-358. (1975)
58. Stark G, Durst RC and Vowell CW,  
'Using metrics in management decision making',  
IEEE Computer, 42-49, Sept, (1994).
59. Vincke P,  
Multicriteria Decision Aid,  
J Wiley, New York, (1992).
60. Vincke P, Multicriteria Decision Aid, J Wiley, New York, 1992.
61. Voas JM and Miller KW,  
Software testability: the new verification,  
IEEE Software, pp 17-28, May, (1995).
62. Walters, G.F. and McCall, J.A.  
Development of Metrics for Reliability and Maintainability.  
Proceedings Annual Reliability and Maintainability Symposium, IEEE (1978).
63. Wilson A.G.  
Cognitive Factors Affecting Subjective Probability Assessment.  
Institute of Statistics and Decision Sciences Discussion Paper No. 94-02. Duke University, Durham, USA (1994).
64. Winkler R.L and Poses R. M.  
Evaluating and combining physicians' probabilities of survival in an intensive care unit.  
Unpublished. Durham NC. Duke University. (1991)

65. Winkler R.L.  
Good probability appraisers  
In: P.Goel and A. Zellner (eds). Bayesian Inference and Decision Techniques. 265-278 (1986)
66. Winkler R.L. Hora SC, Baca R. G.  
The quality of expert judgement elicitations.  
Nuclear Regulatory Commission Contract NRC-02-88-005. San Antonio, TX: Center for Nuclear Waste Regulatory Analyses (1992)
67. Woodward MR, Hennell MA, Hedley D,  
A measure of control flow complexity in program text,  
IEEE Trans Soft. Eng, SE-5 (1), 45-50, (1979).
68. Wright G. & Ayton P. (eds).  
Subjective Probability.  
John Wiley & Sons Ltd., (1994)
69. Zuse H,  
Software Complexity: Measures and Methods,  
De Gruyter. Berlin, (1991).

## 7 Appendix A: Context of the SERENE Method

The objective of this appendix is to describe the relationship between the SERENE Method and other methods and techniques used for ensuring the safety of electronic and programmable systems. The SERENE Method contributes to only a small part of the overall process of justifying the safety of complex systems; it is intended to be compatible with and enhance the existing processes.

During the development of the SERENE Method it has been apparent that these existing processes, which form the context for the use of the SERENE Method, vary widely. Although the practice of safety assessment in Europe is converging under the influence of standards such as IEC 61508, there are still many national and industry sector differences. The SERENE Method cannot be based on a specific safety lifecycle, or a particular safety principle such as GAMAB or ALARP and still be widely applicable across Europe.

In this appendix, the context of the SERENE Method is reviewed so that readers can determine how the SERENE Method could be applied in their specific circumstances. It is intended to show that since the task addressed by the SERENE Method - that of synthesising safety evidence into a safety argument - is common to the different regulatory regimes and safety engineering standards applicable in many countries and industries, the SERENE Method is widely applicable.

The appendix covers the following topics:

- Safety development and assessment
- Safety regulations and standards
- Risk assessment
- Safety principles (including ALARP)
- Safety lifecycles
- Relationship to safety analysis techniques

### 7.1 Safety Development and Assessment

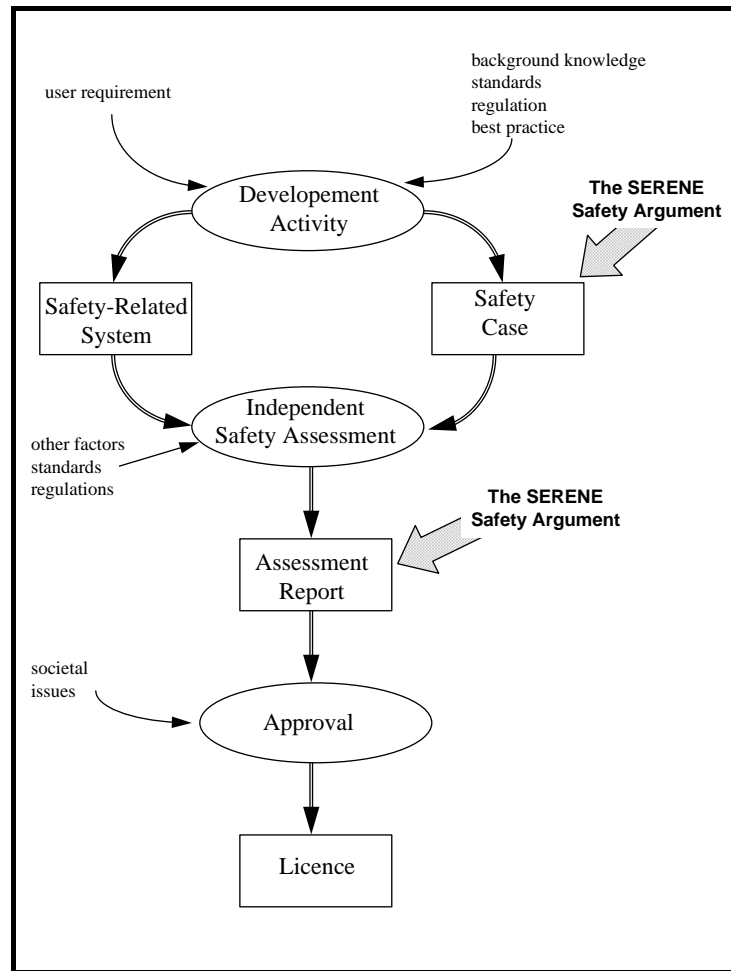
Figure 7-1 shows a representation of the overall process for the development of a safety-related system. The following steps are distinguished.

**Development** A system is developed to meet the requirements of some user, taking into account relevant regulations and standards. The products of this step are the system itself and the safety case.

**Safety Assessment** Both the system and the safety case are scrutinised in the safety assessment process, which is normally carried out independently of the development. The assessment team report on the safety of the proposed system. The assessment report recommends acceptance or rejection of the system. An argument prepared by an assessor will consider factors such as the extent of the information made available to the assessor which would not be relevant to an argument prepared by a developer.

**Approval** A regulator or statutory body must make a decision to approve or licence a system. This decision cannot be taken on the basis of a technical argument alone, but must consider other 'societal factors'.

A safety argument represented by a BBN could be used as part of the safety case prepared during the development of a system. In this case the developer is the user of the SERENE Method. The assessment report could also be based on an argument represented using a BBN. In this case the safety assessor is the user of the SERENE Method.



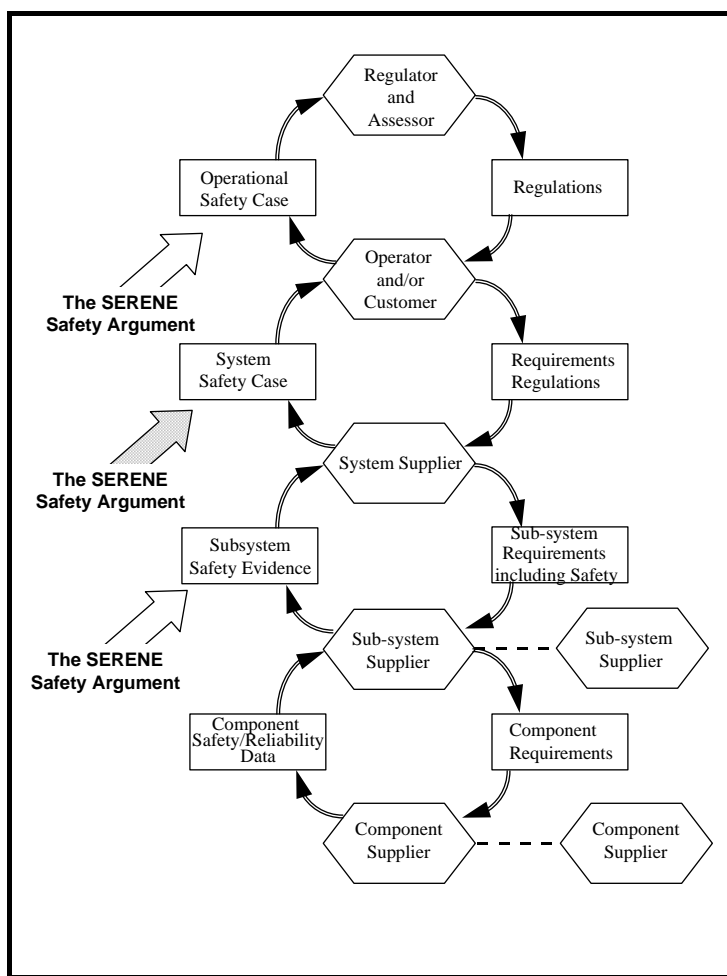
**Figure 7-1: Overall Safety Process**

Figure 7-1 shows only a single safety case arising from the system development process; it is possible to have multiple safety cases. This occurs because a system is a hierarchy with layers of subsystems and components: each subsystem or component can be considered to be a system in its own right. Figure 7-2 illustrates the system hierarchy, showing how each level contributes to the final safety case. The following levels of safety case are distinguished:

**Operational Safety** is a property of a system in an environment. This is illustrated by observing that an aeroplane is safest on the ground, while a car is at its most dangerous on the wrong side of the road. The final stage of safety argument must therefore consider how a system is to be operated, maintained and eventually disposed of. In addition to procuring a system which is capable of being operated safely, the operators will be responsible for ensuring that it actually is operated safely: for example, the staff must be suitably trained. A SERENE safety argument could be used in the operational safety case.

**System** The system developer produces a system safety case. This demonstrates that the design and manufacture of the system makes it fit for purpose. The SERENE Method of representing a safety argument using BBNs is most applicable to this safety case. The system safety case forms a major component of the final operational safety case.

**Subsystem** When a subsystem is purchased from a separate supplier, safety evidence will also need to be supplied. Depending on the scope of the subsystem, this safety evidence might constitute a safety case, in which a SERENE safety argument could be used.



**Figure 7-2: Hierarchical Development of Safety Systems**

**Component Safety** is not normally a property which is applicable to a component. However, other properties of components are important to the safety of the system. For example, a valve may be highly reliable. In addition, the valve may be designed so that it is much more likely to fail in one mode (e.g. open) rather than the other (e.g. closed). This property can be used to design a safe system.

If the use of SERENE Safety Arguments is agreed between the different parties in the supply chain then the BBN used at one level can be incorporated into the argument at the next level.

## 7.2 Safety Regulations and Standards

Documents that are commonly referred to as ‘standards’ may be of several types. For example, the following table shows the top-level documents applicable to the development of an electronically controlled railway switch, for use on the UK railway.

Title	Type
Railway Safety Case Regulations	Regulation
Railway Safety Critical Work Regulations	Regulation
The Construction (Design and Management) Regulations	Regulation
Railway and Other Transport Systems (Approval of Works, Plant and Equipment) Regulations	Regulation
Functional Safety: Safety-related Systems, Draft IEC 61508 (Parts 1-7)	Standard
Railway Applications: The Specification and Demonstration of	



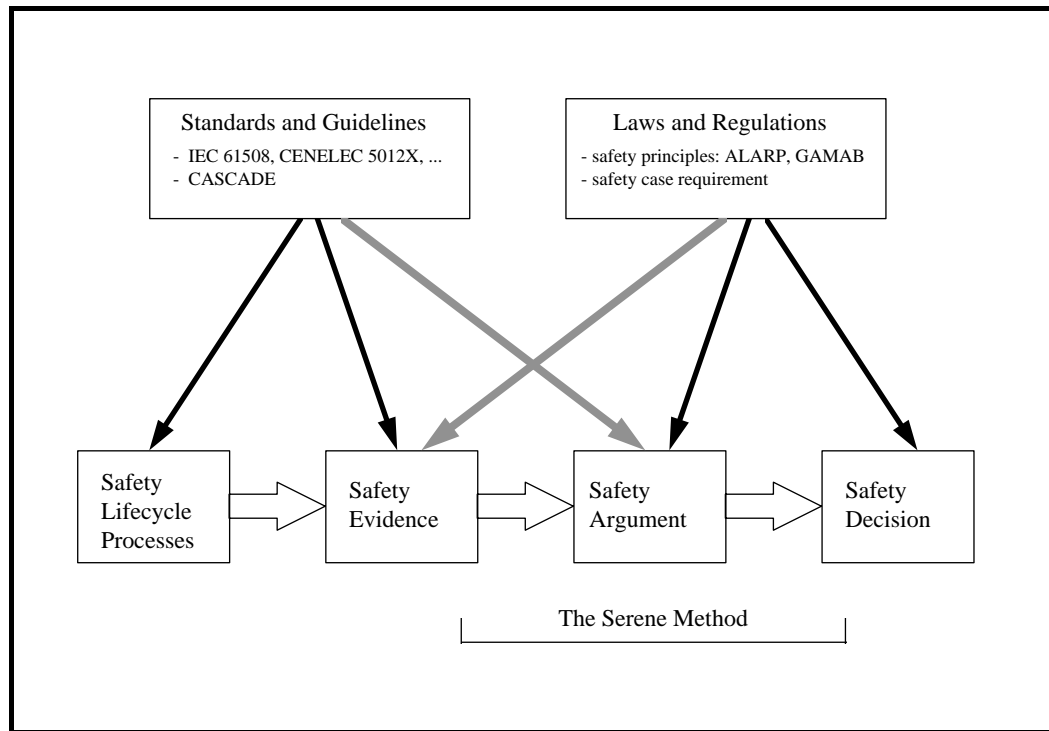
Title	Type
Dependability - Reliability, Availability, Maintainability and Safety (RAMS), CENELEC prEN 50126	Standard
Railway Applications: Software for Railway Control and Protection Systems, CENELEC prEN 50128	Standard
Railway Applications: Safety Related Electronic Systems for Signalling, CENELEC prEN 50129	Standard
Model for Quality Assurance in Design, Development, Production, Installation and Servicing, BS EN ISO 9001	Standard
Electrical Engineering and Control Systems, Engineering Safety Management, EECS-ESM-X-001	Railtrack Procedures

As the example shows, a hierarchy of legal requirements, regulations, standards and guidelines apply to a development project. Although international standards are increasingly being formulated, there is still considerable variation between the legal requirements of different countries and the regulatory requirements of different industries.

**Regulations** The regulations cited in the example derive from UK legal requirements. In some industry sectors, EC Directives are giving rise to common regulations across Europe, replacing existing national regulations. The particular railway regulations cited in this example embody the UK requirements for a safety case to be maintained by Railtrack, the operator of the UK Rail infrastructure. Other countries do not necessarily have this requirement.

**Standards** The standards cited in this example are international (ISO or IEC) or European (CENELEC). In Europe, national standards are increasingly being replaced by European or International standards. However, this process is not yet complete.

**Procedures** The procedures cited in the example have been produced by Railtrack, in order to guide suppliers on the steps necessary to fulfil the requirement to produce a safety case. The intention is for the procedure to supplement, rather than supplant the standards. However, standards are often drafted to be compatible with more than one viewpoint and experience has shown that Railtrack's procedures are not always easily understood outside the UK. It is likely that procedures will continue to be produced by major procurement organisations to guide suppliers.



**Figure 7-3: The Influence of Standards and Regulation on the SERENE Method**

Figure 7-3 illustrates how the laws, regulations, standards and guidelines affect the use of the SERENE Method. Standards and guideline are most concerned with specifying the safety lifecycle and the safety analysis which should be carried out. The laws and regulations address the arguments which are acceptable for reaching a safety decision. As a result, both regulations and standards impact the safety argument and the application of the SERENE Method. The SERENE Method is not itself specific to any particular standard.

### 7.3 Risk Assessment

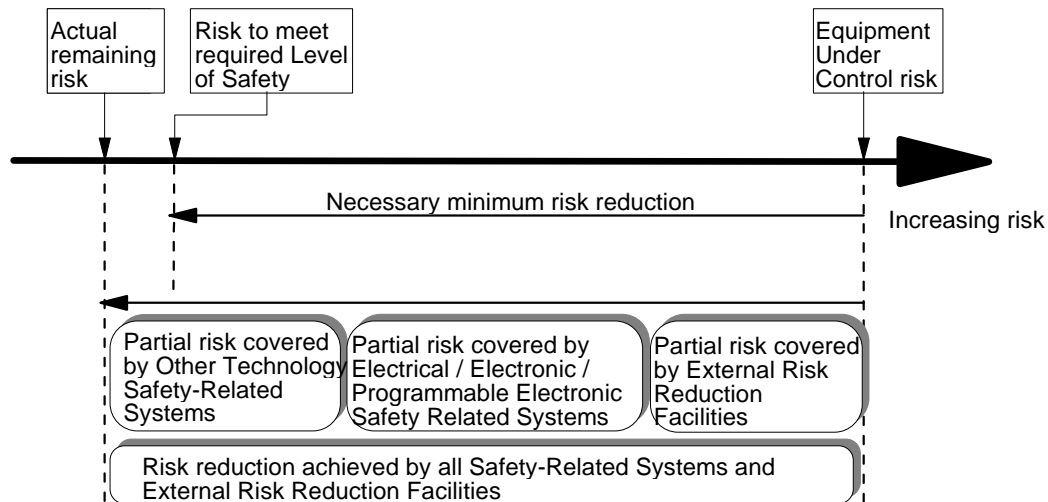
The concept of *risk* is defined in IEC 61508 Part 4 (IEC, 1995) as the ‘probable rate of occurrence of a hazard causing harm and the degree of severity of harm’. Two points of note arise:

1. risk is a property of a hazard and accident relationship
2. risk is determined via a combination of the frequency of an accident and its severity.

Standards such as IEC 61508 are based on a risk assessment principle. Risk assessment is the process of identifying the risks arising from the use of a system prior to its introduction and reducing the risks to acceptable limits. Figure 7-4, taken from IEC 61508, illustrates the concept of risk reduction.

A safety argument developed in compliance with a standard based on risk reduction will be an argument about the remaining risk, or the extent of the risk reduction. The SERENE Method is suitable for use as part of a safety case based on Risk Assessment. Both hazard severity and hazard frequency can be considered to be properties of a system (within its environment) and are therefore amenable to prediction using the SERENE Method.

### Risk Reduction: General Concepts



**Figure 7-4: Risk Reduction**

Although the use of risk assessment is required in IEC 61508 and is widespread, both variations and alternatives exist. For example, a risk based approach is followed in Civil Aviation, but within more restricted parameters. For civil avionics software developed to RTCA DO-178B, frequency of failure is not considered, only severity. For civil avionics systems, risk targets are specified for aircraft-level functions. A simplified risk assessment framework is also proposed in the MISRA Guidelines (MISRA, 1994). The SERENE Method can be applied to these variations of risk assessment, since they are based on a requirement to predict system properties.

## 7.4 Safety Principles

### 7.4.1 Principles to determine acceptability of risk

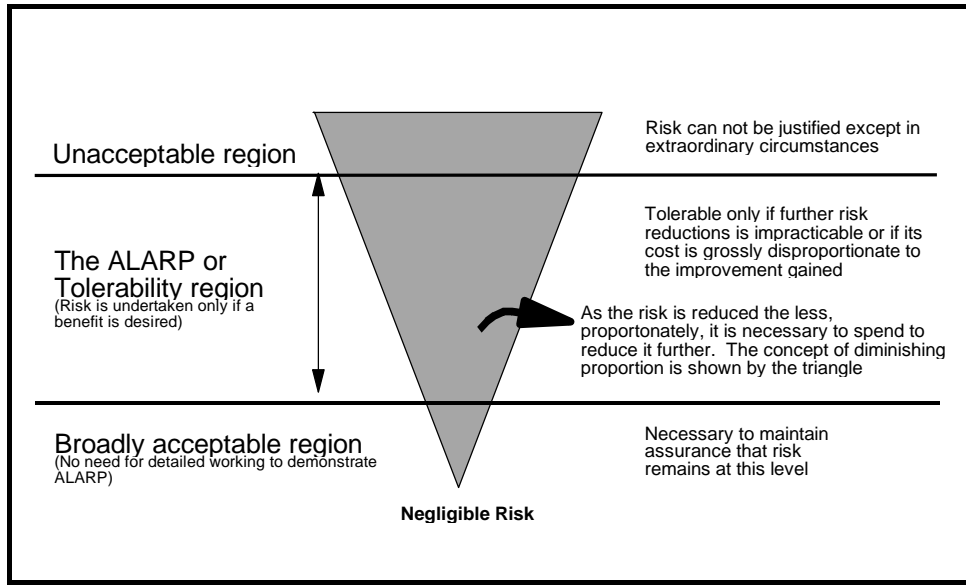
The concepts of risk assessment and risk reduction do not determine the level of risk that is acceptable. We mention here three principles that are used to determine the acceptability of risk:

	Country	Meaning
ALARP	UK	As low as reasonably practicable
GAMAB	France	Globalement au moins aussi bon
MEM	Germany	Minimum endogenous mortality

These principles are similar, but not identical. Typically, the principles are determined by regulations or legislation, not by the standards. The draft IEC 61508 contains guidance material on the ALARP principle, but it is not normative. A safety argument represented using the SERENE Method cannot determine the acceptability of risk: this is a concern of decision making. The safety argument provides input to the decision making process. However, provided that the system property predicted by the SERENE safety argument is chosen appropriately, the SERENE Method can be applied whichever safety principle is applied.

### 7.4.2 The ALARP Principle

The requirement for the operation of safety-related systems is that the level of risk associated with their deployment must be either acceptable, or tolerable and *As Low As Reasonably Practicable* (ALARP).



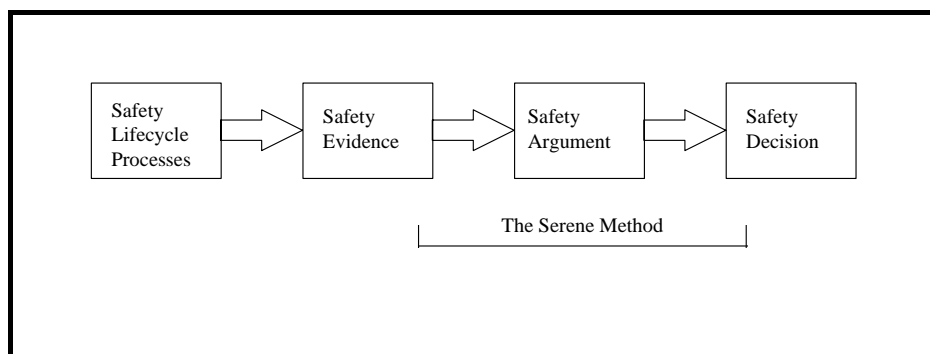
**Figure 7-5: The ALARP Triangle**

A SERENE Safety Argument based on the ALARP Principle could predict the residual risk of a system. Once this risk had been shown to be within the tolerable region, the effect of further risk reduction could be evaluated by modifying the safety argument and compared with the cost.

If the level of risk of operating the hazardous equipment is intolerable or tolerable, then some form of *risk reduction* must be performed in order to make the risk acceptable or to reduce tolerable risks as far as possible within reasonable constraints. The ALARP triangle with intolerable, tolerable and acceptable risk regions is presented in Figure 7-5. The triangular shape is indicative of how the amount of effort to reduce the level of risk deemed to be ‘reasonably practicable’ decreases as risk decreases. The key concept is that, within the ALARP region, the further level of risk reduction required is not absolute, but also takes account of the effort required to achieve the risk reduction.

**7.5 Safety Lifecycles**

The safety lifecycle specifies the activities carried out during the development of a safety-related system and therefore determines the forms of safety evidence which is available for use in the safety argument. This relationship is illustrated in Figure 7-6, which distinguishes the different concepts: safety plans and processes making a safety lifecycle, safety evidence and the compilation of safety evidence into a safety argument.



**Figure 7-6: Relationship of Safety Lifecycle and Argument**

This relationship can be established in two ways.

**From Argument to Lifecycle**

In this approach, the relationship is established 'top-down', working from the safety argument to the safety lifecycle. The need to provide the safety evidence to support the desired form of safety argument determines the safety analyses which are required as part of the lifecycle. The advantage of this approach is to avoid safety analysis or other development activities which do not contribute towards the safety argument.

### **From Lifecycle to Argument**

In this approach, the relationship is established 'bottom-up'. The safety lifecycle is taken as the starting point: the safety evidence produced by following the safety lifecycle form the building blocks of the safety argument. The advantage of this view is that safety lifecycles are much more widely understood than safety arguments. Indeed the safety lifecycle may be laid down in a standard, obliging a developer to treat the lifecycle as the starting point for the development.

In practice, both top-down analysis of the argument and bottom-up analysis of the lifecycle are likely to be required, in order to achieve consistency between the evidence needed in the argument and the evidence produced from the lifecycle.

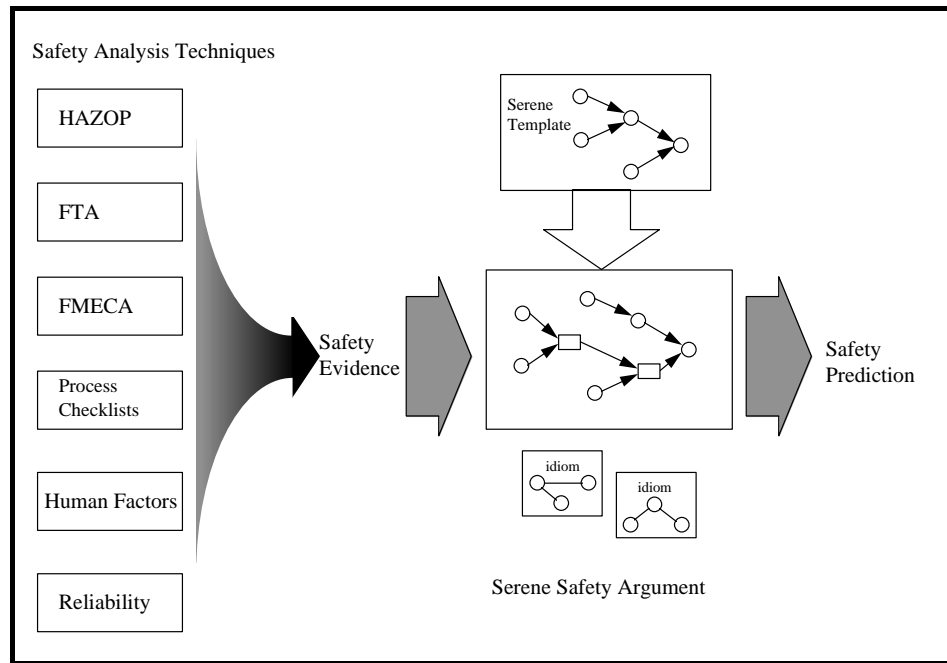
## **7.6 Relationship to Safety Analysis Techniques**

In this section we look at:

- Failure analysis techniques
- Process quality
- Metrics

### **7.6.1 Failure Analysis Techniques**

The SERENE Method is not an alternative to the existing safety and reliability analysis techniques. Instead the SERENE Method depends on the safety evidence generated using existing safety analysis techniques. This is illustrated in Figure 7-7. This shows how failure analysis techniques such as Fault Tree Analysis (FTA) and Failure Modes, Effects and Criticality Analysis (FMECA) are used, in the standard way, to generate evidence which is included in a SERENE Safety Argument. It is notable that FTA and reliability modelling techniques such as Reliability Block Diagrams and Markov Nets all manipulate probabilities. It is possible that Bayesian Nets could be used as an alternative to these techniques, however, that is not the objective of the SERENE Method. The SERENE Method proposes the use of Bayesian Nets for representing a safety argument: none of the existing safety analysis techniques is used in this way.



**Figure 7-7: SERENE and Safety Analysis Techniques**

### 7.6.2 Process Quality

The SERENE Safety Argument should take account of all factors which contribute to safety, including (it is widely believed) the properties of the processes used for safety management and software development. To provide evidence for a safety argument, the processes are evaluated using checklists. The checklists could be based on the requirements of standards such as ISO 9000 and IEC 61508. The competency of the engineers responsible for the development and history of the project should also be evaluated. The SERENE Method does not specify particular checklists which should be used. However, there must be sufficient safety evidence to populate all the nodes in the BBN model of the safety argument.

### 7.6.3 Metrics

Both product and process metrics could be used to provide safety evidence. The BBN safety argument is based on inferences from measurable properties of a system, including its development history, to predict properties which cannot be measured directly. Although the use of metrics is not a requirement of the SERENE Method, the discipline of rigorously defining the properties which need to be measured and determining the measurement method would help to ensure that the inputs to the Bayesian inference process are correct.

*This page intentionally left blank*

## 8 Appendix B: Fundamentals of Probability Theory

This chapter provides a detailed introduction to Bayesian probability as required for BBNs. It covers the following topics:

- Different approaches to probability (Frequentist and Bayesian)
- Probability axioms
- Variables and probability distributions
- Joint events and marginalisation
- Conditional probability
- Bayes rule
- Chain rule
- Independence and conditional independence

### 8.1 Different approaches to probability

There are two fundamentally different approaches to probability:

- The Frequentist approach
- The Bayesian approach

#### 8.1.1 Frequentist approach to probability

Probability theory is the body of knowledge that enables us to reason formally about uncertain events. The populist view of probability is the so-called frequentist approach whereby the probability  $P$  of an uncertain event  $a$ , written  $P(a)$ , is defined by the frequency of that event based on previous observations. For example, in the UK 50.9% of all babies born are girls; suppose then that we are interested in the event  $a$ : 'a randomly selected baby is a girl'. According to the frequentist approach  $P(a)=0.509$ .

#### 8.1.2 Bayesian approach to probability

The frequentist approach for defining the probability of an uncertain event is all well and good providing that we have been able to record accurate information about many past instances of the event. However, if no such historical database exists, then we have to consider a different approach. Suppose, for example, we want to know the probability that a newly developed flight control system contains a critical fault. Since there are no previous instances of such systems we cannot use the frequentist approach to define our degree of belief in this uncertain event.

Bayesian probability is a formalism that allows us to reason about beliefs under conditions of uncertainty. If we have observed that a particular event has happened, such as Spurs winning the FA Cup in 1991, then there is no uncertainty about it. However, suppose  $a$  is the statement

"Spurs win the FA Cup in the year 2001"

Since this is a statement about a future event, nobody can state with any certainty whether or not it is true. Different people may have different beliefs in the statement depending on their specific knowledge of factors that might effect its likelihood. For example, Norman may have a strong belief in the statement  $a$  based on his knowledge of the current team and past achievements. Alan, on the other hand, may have a much weaker belief in the statement based on some inside knowledge about



the status of Spurs; for example, he might know that the club is going to have to sell all of its best players in the year 2000.

Thus, in general, a person's subjective belief in a statement  $a$  will depend on some body of knowledge  $K$ . We write this as  $P(a|K)$ . Norman's belief in  $a$  is different from Alan's because they are using different  $K$ 's. However, even if they were using the same  $K$  they might still have different beliefs in  $a$ .

The expression  $P(a|K)$  thus represents a *belief measure*. Sometimes, for simplicity, when  $K$  remains constant we just write  $P(a)$ , but you must be aware that this is a simplification.

## 8.2 Probability axioms

While different people may give  $P(a)$  a different value there are nevertheless certain axioms which should always hold for internal consistency. These are the axioms of probability theory (which can be *proved* to be valid when  $P(a)$  represents the frequentist approach):

1.  $P(a)$  should be a number between 0 and 1
2. If  $a$  represents a certain event then  $P(a)=1$ .
3. If  $a$  and  $b$  are mutually exclusive events then  $P(a \text{ or } b) = P(a)+P(b)$

(mutually exclusive means that they cannot both be true at the same time; for example, if  $a$  represents the proposition that our control system contains 0 faults, while  $b$  represents the proposition that our control system contains 1 fault)

For any event  $a$  we denote the negation of  $a$  as  $\neg a$ . For example, if  $a$  represents the proposition:

"our control system contains 0 faults"

then  $\neg a$  represents the proposition:

"our control system contains at least 1 fault"

Clearly for any event  $a$ , the event

$a$  or  $\neg a$

is a certain event, and hence from axiom 2:

$$P(a \text{ or } \neg a) = 1$$

Since the events  $a$  and  $\neg a$  are mutually exclusive It follows from axiom 3 that

$$1 = P(a \text{ or } \neg a) = P(a) + P(\neg a)$$

and hence that

$$P(a) = 1 - P(\neg a)$$

Sometimes this is called the 4<sup>th</sup> axiom, but it follows from the others.

## 8.3 Variables and probability distributions

We have seen the example of the uncertain event  $a = \text{"Spurs win the FA Cup in the year 2001"}$ . We can think of this event as just one state of the *variable*  $A$  which represents "FA Cup winners in 2001". In this case  $A$  has many states, one for each team entering the FA Cup. We write this as

$$A = \{a_1, a_2, \dots, a_n\}$$

where  $a_1 = \text{"Spurs"}$ ,  $a_2 = \text{"Chelsea"}$ ,  $a_3 = \text{"West Ham"}$ , etc.

Since in this case the set  $A$  is finite we say that  $A$  is a finite discrete variable.

As another example, suppose we are interested in the number of critical faults in our control system. The uncertain event is  $A = \text{"Number of critical faults"}$ . Again it is best to think of  $A$  as a variable which can take on any of the discrete values  $0,1,2,3,\dots$  thus

$$A = \{0,1,2,3,\dots\}.$$

Let us define  $a_1$  as the event " $A=0$ ", and  $a_2$  as the event " $A=1$ ".

Clearly the events  $a_1$  and  $a_2$  are mutually exclusive and so  $P(a_1 \text{ or } a_2) = P(a_1) + P(a_2)$ . However, we cannot say that

$$P(a_1 \text{ or } a_2) = 1$$

because  $a_1$  and  $a_2$  are not *exhaustive*. That is, they do not form a complete partition of  $A$ . However, if we define  $a_3$  as the event " $A > 1$ " then  $a_1, a_2$ , and  $a_3$  are complete and mutually exhaustive and in this case

$$P(a_1) + P(a_2) + P(a_3) = 1$$

In general if  $A$  is a variable with states  $a_1, a_2, \dots, a_n$ :

$$\sum_{i=1}^n P(a_i) = 1$$

The *probability distribution* of  $A$ , written  $P(A)$ , is simply the set of values  $\{P(a_1), P(a_2), \dots, P(a_n)\}$

#### 8.4 Joint events and marginalisation

Suppose that our control system  $X$  is made up of two subsystems. Let  $A$  be the number of critical faults in the first subsystem and let  $B$  be the number of critical faults in the second subsystem.

Suppose that

$$A = \{a_1, a_2, a_3\} \text{ where } a_1=0, a_2=1, a_3=">1"$$

$$B = \{b_1, b_2, b_3\} \text{ where } b_1=0, b_2=1, b_3=">1"$$

If we are interested in the overall number of critical faults in the system, then we speak about the joint event  $A$  and  $B$ . We write the probability of this event as

$$P(A, B)$$

$P(A, B)$  is called the *joint probability distribution* of  $A$  and  $B$ . Specifically,  $P(A, B)$  is the set of probabilities:

$$\{P(a_1, b_1), P(a_1, b_2), P(a_1, b_3), P(a_2, b_1), P(a_2, b_2), P(a_2, b_3), P(a_3, b_1), P(a_3, b_2), P(a_3, b_3)\}$$

where for any  $i$  and  $j$ ,  $P(a_i, b_j)$  is the probability of the event  $a_i$  and  $b_j$ .

In general, if  $A$  and  $B$  are variables with possible states  $\{a_1, a_2, \dots, a_n\}$  and  $\{b_1, \dots, b_m\}$  respectively then the joint probability distribution  $P(A, B)$  is the set of probabilities

$$\{P(a_i, b_j) \mid i=1, \dots, n, j=1, \dots, m\}$$

If we know the joint probability distribution  $P(A, B)$  then we can calculate  $P(A)$  by a formula (called *marginalisation*) which comes straight from the third axiom, namely:

$$P(a) = \sum_i P(a, b_i)$$

This is because the events  $(a, b_1), (a, b_2), \dots, (a, b_m)$  are mutually exclusive. When we calculate  $P(A)$  in this way from the joint probability distribution we say that the variable  $B$  is *marginalised out* of  $P(A, B)$ . It is a very useful technique because in many situations it may be easier to calculate  $P(A)$  from  $P(A, B)$ .

## 8.5 Conditional probability

In the introduction to Bayesian probability we explained that the notion of degree of belief in an uncertain event  $A$  was *conditional* on a body of knowledge  $K$ . Thus, the basic expressions about uncertainty in the Bayesian approach are statements about conditional probabilities. This is why we used the notation  $P(A|K)$  which should only be simplified to  $P(A)$  if  $K$  is constant. Any statement about  $P(A)$  is always conditioned on a context  $K$ .

In general we write  $P(A|B)$  to represent a belief in  $A$  under the assumption that  $B$  is known. Even this is, strictly speaking, shorthand for the expression  $P(A|B,K)$  where  $K$  represents all other relevant information. Only when all such other information is irrelevant can we really write  $P(A|B)$ .

The traditional approach to defining conditional probabilities is via joint probabilities. Specifically we have the well-known 'formula':

$$P(A | B) = \frac{P(A, B)}{P(B)}$$

This should really be thought of as an axiom of probability. Just as we saw the three probability axioms were 'true' for frequentist probabilities, so this axiom can be similarly justified in terms of frequencies:

**Example:** Let  $A$  denote the event 'student is female' and let  $B$  denote the event 'student is Chinese'. In a class of 100 students suppose 40 are Chinese, and suppose that 10 of the Chinese students are females. Then clearly, if  $P$  stands for the frequency interpretation of probability we have:

$$P(A,B) = 10/100 \text{ (10 out of 100 students are both Chinese and female)}$$

$$P(B) = 40/100 \text{ (40 out of the 100 students are Chinese)}$$

$$P(A|B) = 10/40 \text{ (10 out of the 40 Chinese students are female)}$$

It follows that the formula above for conditional probability 'holds'.

In those cases where  $P(A|B) = P(A)$  we say that  $A$  and  $B$  are *independent*.

If  $P(A|B,C) = P(A|C)$  we say that  $A$  and  $B$  are *conditionally independent* given  $C$ .

The notions of independence and conditional independence are crucial for BBNs and are examined in more detail in Section 8.8.

## 8.6 Bayes theorem

True Bayesians actually consider conditional probabilities as more basic than joint probabilities. It is easy to define  $P(A|B)$  without reference to the joint probability  $P(A,B)$ . To see this note that we can rearrange the conditional probability formula to get:

$$P(A|B) P(B) = P(A,B)$$

but by symmetry we can also get:

$$P(B|A) P(A) = P(A,B)$$

It follows that:

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}$$

which is the so-called **Bayes Rule** or **Bayes Theorem**.

It is common to think of Bayes rule in terms of updating our belief about a hypothesis  $A$  in the light of new evidence  $B$ . Specifically, our *posterior* belief  $P(A/B)$  is calculated by multiplying our *prior* belief  $P(A)$  by the *likelihood*  $P(B/A)$  that  $B$  will occur if  $A$  is true.

The power of Bayes' rule is that in many situations where we want to compute  $P(A|B)$  it turns out that it is difficult to do so directly, yet we might have direct information about  $P(B|A)$ . Bayes' rule enables us to compute  $P(A/B)$  in terms of  $P(B/A)$ .

For example, suppose that we are interested in diagnosing cancer in patients who visit a chest clinic.

Let  $A$  represent the event "Person has cancer"

Let  $B$  represent the event "Person is a smoker"

Suppose we know the probability of the prior event  $A$  is 0.1 on the basis of past data (10% of patients entering the clinic turn out to have cancer). Thus:

$$P(A)=0.1$$

We want to compute the probability of the posterior event  $P(A/B)$ .

It is difficult to find this out directly. However, we are likely to know  $P(B)$  by considering the percentage of patients who smoke – suppose  $P(B)=0.5$ . We are also likely to know  $P(B/A)$  by checking from our records the proportion of smokers among those diagnosed with cancer. Suppose  $P(B/A)=0.8$ .

We can now use Bayes' rule to compute:

$$P(A/B) = (0.8 \times 0.1)/0.5 = 0.16$$

Thus, in the light of *evidence* that the person is a smoker we revise our prior probability from 0.1 to a posterior probability of 0.16. This is a significance increase, but it is still unlikely that the person has cancer.

The denominator  $P(B)$  in the equation is a normalising constant which can be computed, for example, by marginalisation whereby

$$P(B) = \sum_i P(B, A_i) = \sum_i P(B|A_i) \cdot P(A_i)$$

Hence we can state Bayes rule in another way as:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{\sum_i P(B|A_i) \cdot P(A_i)}$$

### 8.6.1 Bayes Theorem Example

Suppose that we have two bags each containing black and white balls. One bag contains three times as many white balls as blacks. The other bag contains three times as many black balls as white. Suppose we choose one of these bags at random. For this bag we select five balls at random, replacing each ball after it has been selected. The result is that we find 4 white balls and one black. What is the probability that we were using the bag with mainly white balls?

**Solution:** Let  $A$  be the random variable "bag chosen" then  $A=\{a_1, a_2\}$  where  $a_1$  represents "bag with mostly white balls" and  $a_2$  represents "bag with mostly black balls". We know that  $P(a_1)=P(a_2)=1/2$  since we choose the bag at random.

Let B be the event "4 white balls and one black ball chosen from 5 selections".

Then we have to calculate  $P(a_1|B)$ . From Bayes' rule this is:

$$P(a_1|B) = \frac{P(B|a_1) \cdot P(a_1)}{P(B|a_1) \cdot P(a_1) + P(B|a_2) \cdot P(a_2)}$$

Now, for the bag with mostly white balls the probability of a ball being white is  $\frac{3}{4}$  and the probability of a ball being black is  $\frac{1}{4}$ . Thus, we can use the Binomial Theorem, to compute  $P(B|a_1)$  as:

$$P(B|a_1) = \binom{5}{1} \left(\frac{3}{4}\right)^4 \left(\frac{1}{4}\right)^1 = \frac{405}{1024}$$

Similarly

$$P(B|a_2) = \binom{5}{1} \left(\frac{1}{4}\right)^4 \left(\frac{3}{4}\right)^1 = \frac{15}{1024}$$

hence

$$P(a_1|B) = \frac{405/1024}{405/1024 + 15/1024} = \frac{405}{420} = 0.964$$

### 8.6.2 Likelihood Ratio

We have seen that Bayes' rule computes  $P(A|B)$  in terms of  $P(B|A)$ . The expression  $P(B|A)$  is called the likelihood of A. In the example above A had two values  $a_1$  and  $a_2$ . The ratio

$$\frac{P(B|a_1)}{P(B|a_2)}$$

is called the likelihood ratio. In the example the likelihood ratio computes to  $405/15 = 27$ . This tells us that the 'odds' on the bag being the one mainly with white balls is 27 to 1.

### 8.7 Chain rule

We can rearrange the formula for conditional probability to get the so-called *product rule*:

$$P(A,B) = p(A|B) p(B)$$

We can extend this for three variables:

$$P(A,B,C) = P(A|B,C) P(B,C) = P(A|B,C) P(B|C) P(C)$$

and in general to n variables:

$$P(A_1, A_2, \dots, A_n) = P(A_1|A_2, \dots, A_n) P(A_2|A_3, \dots, A_n) P(A_{n-1}|A_n) P(A_n)$$

In general we refer to this as the *chain rule*.

This formula is especially significant for Bayesian Belief Nets. It provides a means of calculating the full joint probability distribution ; in BBNs many of the variables  $A_i$  will be conditionally independent which means that the formula can be simplified (see Section 9.7).

## 8.8 Independence and conditional independence

We already introduced the notion of independence in Section 8.5 . In this subsection we explain these concepts in more detail since they are central to understanding BBNs.

The conditional probability of A given B is represented by  $P(A|B)$ . The variables A and B are said to be *independent* if  $P(A) = P(A|B)$  (or alternatively if  $P(A,B) = P(A) P(B)$  because of the formula for conditional probability ).

**Example 1** Suppose Norman and Martin each toss separate coins. Let A represent the variable "Norman's toss outcome", and B represent the variable "Martin's toss outcome". Both A and B have two possible values (Heads and Tails). It would be uncontroversial to assume that A and B are independent. Evidence about B will not change our belief in A.

**Example 2** Now suppose both Martin and Norman toss the same coin. Again let A represent the variable "Norman's toss outcome", and B represent the variable "Martin's toss outcome". Assume also that there is a possibility that the coin is biased towards heads but we do not know this for certain. In this case A and B are not independent. For example, observing that B is Heads causes us to increase our belief in A being Heads (in other words  $P(a|b) > P(b)$  in the case when  $a = \text{Heads}$  and  $b = \text{Heads}$ ).

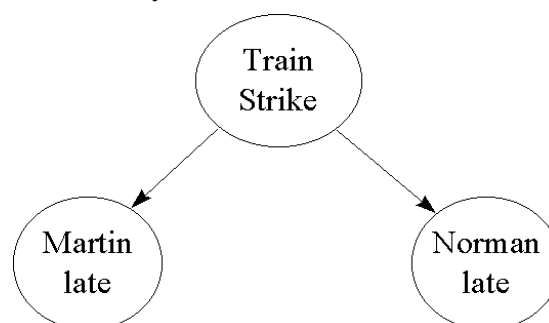
In Example 2 the variables A and B are both dependent on a separate variable C, "the coin is biased towards Heads" (which has the values True or False). Although A and B are not independent, it turns out that once we *know* for certain the value of C then any evidence about B cannot change our belief about A. Specifically:

$$P(A|C) = P(A|B,C)$$

In such case we say that *A and B are conditionally independent given C*.

In many real life situations variables which are believed to be independent are actually only independent conditional on some other variable.

**Example 3** Suppose that Norman and Martin live on opposite sides of the City and come to work by completely different means, say Norman comes by train while Martin drives. Let A represent the variable "Norman late" (which has values true or false) and similarly let B represent the variable "Martin late". It would be tempting in these circumstances to assume that A and B must be independent. However, even if Norman and Martin lived and worked in different countries there may be factors (such as an international fuel shortage) which could mean that A and B are not independent. In practice any model of uncertainty should take account of all *reasonable* factors. Thus while, say, a meteorite hitting the Earth might be reasonably excluded it does not seem reasonable to exclude the fact that both A and B may be affected by a Train strike (C). Clearly  $P(A)$  will increase if C is true; but  $P(B)$  will also increase because of extra traffic on the roads. Thus the situation is represented in the following diagram (which is actually a BBN)



Now, "Martin late" and "Norman late" are conditionally independent given "Train strike". Once we know there is a train strike the events "Norman late" and "Martin late" are independent. See also the section on transmitting evidence in BBNs (Section 9.2).

## 9 Appendix C: Bayesian Belief Nets Tutorial

This appendix provides a detailed tutorial on BBNs (if you are seeking only a broad overview of BBNs, then go to Section 2.2.1 of the main report- it is self-contained).

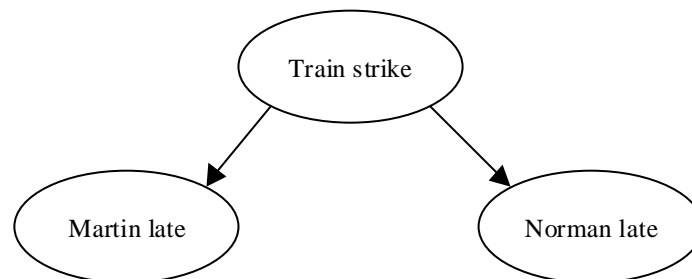
The appendix covers:

- Definition of BBNs
- Analysing a BBN: entering evidence and propagation
- Explaining away evidence in a BBN
- Why do we need BBNs in probability computations
- General case of joint probability distribution in BBN
- Dealing with increases in the number of variables
- Why we should use BBNs
- How BBNs deal with evidence
- Types of reasoning permitted in BBNs

### 9.1 Definition of BBNs: graphs and probability tables

A BBN is a special type of diagram (called a graph) together with an associated set of probability tables.

The graph consists of nodes and arcs as shown in the simple example:



The nodes represent variables, which can be discrete or continuous. For example, a node might represent the variable 'Train strike' which is discrete, having the two possible values 'true' and 'false'. The arcs represent causal relationships between variables. For example, suppose we have another variable 'Norman late' which is also discrete with values 'true' and 'false'. Since a train strike can *cause* Norman to be late we model this relationship by drawing an arc from the node 'Train strike' to the node 'Norman late'

The key feature of BBNs is that they enable us to model and reason about *uncertainty*. In our example, a train strike does not imply that Norman will definitely be late (he might leave early and drive), but there *is* an increased probability that he will be late. In the BBN we model this by filling in a probability table for each node. For the node 'Norman late' the probability table (also called the Node Probability Table or NPT) might look like this:

Norman late	Train Strike	
	True	False
True	0.8	0.1
False	0.2	0.9

This is actually the conditional probability of the variable 'Norman late' given the variable 'train strike'. The possible values (true or false) for 'Norman late' are shown in the first column. Note that we



provide a probability for each combination of events (four in this case), although the rules of probability mean that some of this information is redundant. Informally, the particular values in this table tell us that: Norman is very unlikely to be late normally (that is, the probability Norman is late when there is no train strike is 0.1), but if there is a train strike he is very likely to be late (the probability is 0.8). Now suppose that Norman has a colleague, Martin, who usually drives to work. To model our uncertainty about whether or not Martin arrives late we add a new node 'Martin late' to the graph and an arc from 'Train strike' to this node. A train strike can still cause Martin to be late because traffic is heavier in that case. However, the probability table for 'Martin late', shown here, is very different in content to the one for 'Norman late':

<b>Martin late</b>	<b>Train Strike</b>	
	True	False
True	0.6	0.5
False	0.4	0.5

Informally, Martin is often late, but a train strike only increases the likelihood of his lateness by a small amount. In the event of a train strike Martin is less likely to be late than Norman.

The probability table associated with the node 'Train strike' is somewhat different in nature. This node has no 'parent' nodes in this model (we call it a **root** node), and therefore we only have to assign a probability to each of the two possible values 'true' and 'false'. In the following table the actual values suggest that a train strike is very unlikely.

<b>Train strike</b>	
True	0.1
False	0.9

There may be several ways of determining the probabilities of any of the tables. For example, in the previous table we might be able to base the probabilities on previously observed frequencies of days when there were train strikes. Alternatively, if no such statistical data is available we may have to rely on subjective probabilities entered by experts. The beauty of BBNs is that we are able to accommodate both subjective probabilities and probabilities based on objective data.

## 9.2 Analysing a BBN: entering evidence and propagation

Having entered the probabilities we can now use Bayesian probability to do various types of analysis. For example, we might want to calculate the (unconditional) probability that Norman is late:

$$\begin{aligned}
 p(\text{Norman late}) &= p(\text{Norman late} \mid \text{train strike}) * p(\text{train strike}) + p(\text{Norman late} \mid \text{no train strike}) \\
 &= (0.8 * 0.1) + (0.1 * 0.9) \\
 &= 0.17
 \end{aligned}$$

This is called the marginal probability (see Section 8.4).

Similarly, we can calculate the marginal probability that Martin is late to be 0.51.

However, the most important use of BBNs is in *revising* probabilities in the light of actual observations of events. Suppose, for example, that we *know* there is a train strike. In this case we can **enter the evidence** that 'train strike' = true. The conditional probability tables already tell us the revised probabilities for Norman being late (0.8) and Martin being late (0.6). Suppose, however, that we do not know if there is a train strike but do know that Norman is late. Then we can enter the evidence that 'Norman late' = true and we can use this observation to determine:

- a) the (revised) probability that there is a train strike; and

b) the (revised) probability that Martin will be late.

To calculate a) we use Bayes theorem :

$$p(\text{Train strike} | \text{Norman late}) = \frac{p(\text{Norman late} | \text{train strike}) \times p(\text{train strike})}{p(\text{Norman late})} = \frac{0.8 \times 0.1}{0.17} = 0.47$$

Thus, the observation that Norman is late significantly increases the probability that there is a train strike (up from 0.1 to 0.47). Moreover, we can use this revised probability to calculate b):

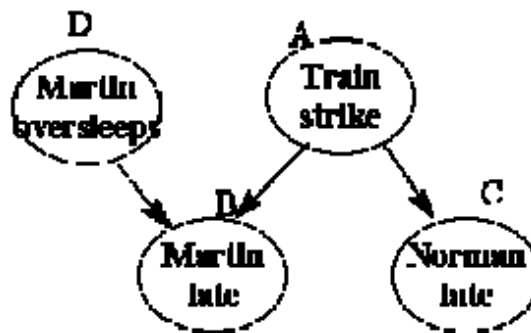
$$\begin{aligned} p(\text{Martin late}) &= p(\text{Martin late} | \text{train strike}) * p(\text{train strike}) + p(\text{Martin late} | \text{no train strike}) \\ &= (0.6 * 0.47) + (0.5 * 0.53) \\ &= 0.55 \end{aligned}$$

Thus, the observation that Norman is late has slightly increased the probability that Martin is late. When we enter evidence and use it to update the probabilities in this way we call it **propagation**.

For a detailed look at how BBNs transmit evidence for propagation (including the notions of *d-connectedness* and *separation*) see Section 9.8.

### 9.3 The notion of 'explaining away' evidence

Now consider the following slightly more complex network:



In this case we have to construct a new conditional probability table for node B ('Martin late') to reflect the fact that it is conditional on *two* parent nodes (A and D). Suppose the table is:

		Martin oversleeps		Train strike	
		True	False	True	False
Martin late	True	0.8	0.5	0.6	0.5
	False	0.2	0.5	0.4	0.5

We also have to provide a probability table for the new root node D ('Martin oversleeps').

Martin oversleeps	
True	0.4
False	0.6

We have already seen that in this initialised state the probability that Martin is late is 0.51 and the probability that Norman is late is 0.17.

Suppose we find out that Martin *is* late. This evidence increases our belief in both of the possible causes (namely a train strike A and Martin oversleeping B). Specifically, applying Bayes theorem yields a revised probability of A of 0.13 (up from the prior probability of 0.1) and a revised

probability of D of 0.41 (up from the prior probability of 0.4). However, if we had to bet on it, our money would be firmly on Martin oversleeping as the more likely cause. Now suppose we also discover that Norman is late. Entering this evidence and applying Bayes yields a revised probability of 0.54 for a train strike and 0.44 for Martin oversleeping. Thus the odds are that the train strike, rather than oversleeping, have caused Martin to be late. We say that Martin's lateness has been 'explained away'.

#### 9.4 Why do we need a BBN for the probability computations?

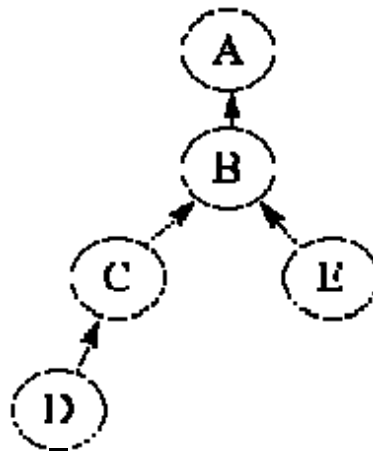
BBNs make explicit the dependencies between different variables. In general there may be relatively few direct dependencies (modelled by arcs between nodes of the network) and this means that many of the variables are conditionally independent. In the simple example above the nodes 'Norman late' and 'Martin late' are conditionally independent (there is no arc linking them); once the value of 'train strike' is known knowledge of 'Norman late' does not effect the probability of 'Martin late' and vice versa.

The existence of unlinked (conditionally independent) nodes in a network drastically reduces the computations necessary to work out all the probabilities we require. In general, all the probabilities can be computed from the joint probability distribution. Crucially, this joint probability distribution is far simpler to compute when there are conditionally independent nodes.

Suppose, for example, that we have a network consisting of five variables (nodes) A,B,C,D,E. If we do not specify the dependencies explicitly then we are essentially assuming that all the variables are dependent on each other. The chain rule enables us to calculate the joint probability distribution  $p(A,B,C,D,E)$  as:

$$p(A,B,C,D,E) = p(A|B,C,D,E)*p(B|C,D,E)*p(C|D,E)*p(D|E)*p(E)$$

However, suppose that the dependencies are explicitly modelled in a BBN as:



Then the joint probability distribution  $p(A,B,C,D,E)$  is much simplified:

$$p(A,B,C,D,E) = p(A|B)*p(B|C,E)*p(C|D)*p(D)*p(E)$$

#### 9.5 General case of joint probability distribution in BBN

Suppose the set of variables in a BBN is  $\{A_1, A_2, \dots, A_n\}$  and that  $\text{parents}(A_i)$  denotes the set of parents of the node  $A_i$  in the BBN. Then the joint probability distribution for  $\{A_1, A_2, \dots, A_n\}$  is:

$$P(A_1, \dots, A_n) = \prod_{i=1}^n P(A_i | \text{parents}(A_i))$$

## 9.6 Dealing with the increases in the number of variables

Even in the simple example above it is tricky to work out all the probabilities and the revised probabilities once evidence is entered. Imagine a larger net with many dependencies and nodes that can take on more than two values. Doing the propagation in such cases is generally very difficult. In fact, there are no universally efficient algorithms for doing these computations (the problem is NP-hard). This observation, until relatively recently, meant that BBNs could not be used to solve realistic problems. However, in the 1980s researchers discovered propagation algorithms that were effective for large classes of BBNs. With the introduction of software tools that implement these algorithms (as well as providing a graphical interface to draw the graphs and fill in the probability tables) it is now possible to use BBNs to solve complex problems without doing any of the Bayesian calculations by hand. This is the reason why the popularity of BBNs has mushroomed in recent years. They have already proven useful in practical applications including medical diagnosis, diagnosis of mechanical failures, and adaptive human interfaces for computer software.

## 9.7 Why we should use BBNs

BBNs on their own enable us to model uncertain events and arguments about them. The intuitive visual representation can be very useful in clarifying previously opaque assumptions or reasonings hidden in the head of an expert. With BBNs, it is possible to articulate expert beliefs about the dependencies between different variables and BBNs allow an injection of scientific rigour when the probability distributions associated with individual nodes are simply 'expert opinions'. BBNs can expose some of the common fallacies in reasoning due to misunderstanding of probability.

However, the real power of BBNs comes when we apply the rules of Bayesian probability to propagate consistently the impact of evidence on the probabilities of uncertain outcomes. A BBN will derive all the implications of the beliefs that are input to it; some of these will be facts that can be checked against observations, or simply against the experience of the decision makers themselves.

## 9.8 How BBNs deal with evidence

In this section we look at the way that evidence is transmitted in BBNs. We consider two types of evidence:

- **Hard evidence (instantiation)** for a node *X* is evidence that the state of *X* is definitely a particular value. For example, suppose *X* represents the result of a particular match for a football team {win, lose, draw}. Then an example of hard evidence would be knowledge that the match is definitely won. In this case we also say *X* is *instantiated* as the value 'win'.
- **Soft evidence** for a node *X* is any evidence that enables us to update the prior probability values for the states of *X*. For example, if we know that the team is winning the match 3-0 at half-time, then the probability of *win* would be quite high, while the probabilities of both *lose* and *draw* would be quite low (compared with ignorance prior values).

We distinguish three types of connection in a BBN. In a serial connection (Section 9.8.1) we will see that any evidence entered at the beginning of the connection can be transmitted along the directed path providing that no intermediate node on the path is instantiated (which thereby blocks further transmission). In a diverging connection (Section 9.8.2) we will see that evidence can be transmitted between two child nodes of the same parent providing that the parent is not instantiated. In a converging connection (Section 9.8.3) we will see that evidence can only be transmitted between two parents when the child (converging) node has received some evidence (which can be soft or hard).

These notions of transmitting evidence enable us to describe the important (but quite tricky) notion of 'explaining away' evidence in a BBN. Also the rules for transmitting evidence for serial, diverging and converging connections are sufficient for us to describe a completely general procedure for

determining whether any two nodes of a BBN are dependent or not. This is the formal notion of d-separation (Section 9.8.4). This notion is crucial for understanding how the algorithms for probability propagation in BBNs actually work.

### 9.8.1 Serial Connection

Consider the following BBN



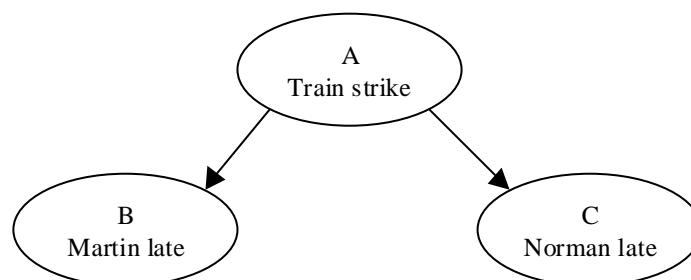
Suppose we have some evidence that a signal failure has occurred (A). Then clearly this knowledge increases our belief that the train is delayed (B), which in turn increases our belief that Norman is late (C). Thus evidence about A is transmitted through B to C. In general any evidence about A will be transmitted through B to C.

However, now suppose that we know the true status of B; for example, suppose we know that the train is delayed (that is, we have hard evidence for B). In this case any knowledge about A is irrelevant to C because our knowledge of B essentially 'overwrites it'; any new information about the likelihood of a signal failure (A) is not going to change our belief about Norman being late once we know that the train is delayed. In other words the evidence from A cannot be transmitted to C because B **blocks** the channel.

In summary, in a serial connection evidence can be transmitted from A to C unless B is instantiated. Formally we say that **A and C are d-separated given B**.

### 9.8.2 Diverging connection

Consider the following BBN:



Any evidence about A is transmitted to both B and C. For example, if we have evidence which increases our belief in a train strike (A) then this in turn will increase our belief in both Martin being late (B) and Norman being late (C).

Of more interest is whether information about B can be transmitted to C (and vice versa). Suppose we have no hard evidence about A (that is, we do not know for certain whether or not there is a train strike). If we have some evidence that Martin is late (B) then this increases our belief in A (that is, we also reason in the opposite direction of the causal arrows). This in turn increases our belief in C. In other words, evidence about B (Martin late) is transmitted through to C (Norman late).

However, suppose now that we have hard evidence about A, that is we know for certain whether or not there is a train strike (we also say that A is instantiated). In this case any evidence about B

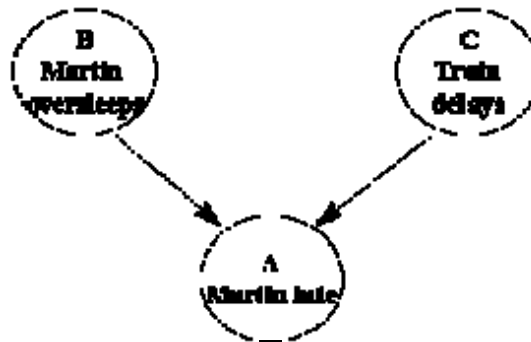
(Martin late) does not change in any way our belief about C (Norman late); this is because the certainty of A blocks the evidence from being transmitted (it becomes irrelevant once we know A for certain). The value of C is only influenced by the certainty of A. Thus, when A is known for certain B and C become independent.

Because the independence of B and C is conditional on the certainty of A, we say formally that B and C are *conditionally independent* (given A). (For a more basic explanation of independence and conditional independence [click here](#) ).

In summary: evidence can be transmitted from B to C through a diverging connection A unless A is instantiated. We say that **B and C are d-separated given A**.

### 9.8.3 Converging connection

Consider the BBN in the figure below.



Clearly any evidence about B or C is transmitted to A. However, we are now concerned about whether evidence can be transmitted between B and C.

If we have no information about whether or not Martin is late, then clearly whether or not Martin oversleeps and whether or not there are train delays are independent. In other words if nothing is known about A then A's parents (B and C) are independent, so no evidence is transmitted between them.

However, if *anything* is known about A (even so-called soft evidence) then the parents of A become dependent. For example, suppose that Martin usually hangs up his coat in the hall as soon as he gets in to work. Then if we observe that Martin's coat is not hung up after 9.00am our belief that he is late increases (note that we do not know for certain that he is late - we have soft as opposed to hard evidence - because on some days Martin does not wear a coat). Even this 'soft' evidence about Martin being late increases our belief in both *B:Martin oversleeping* and *C:Train delays* (see also the principle of 'explaining away'). We say that B and C are conditionally dependent (on A).

It follows that in a converging connection evidence can only be transmitted between the parents B and C when the converging node A has received some evidence (which can be soft or hard).

### 9.8.4 The notion of d-separation

What we have seen above is that:

1. In a serial connection from B to C via A, evidence from B to C is blocked only when we have hard evidence about A.
2. In a diverging connection where B and C have the common parent A evidence from B to C is blocked only when we have hard evidence about A.
3. In a converging connection where A has parents B and C any evidence about A results in evidence transmitted between B and C.

In cases 1 and 2 we say that the nodes B and C are d-separated when there is hard evidence of A. In case 3 B and C are only d-separated when there is *no* evidence about A. In general two nodes which are not d-separated are said to be d-connected.

These three cases enable us to determine in general whether any two nodes in a given BBN are dependent (d-connected) given the evidence entered in the BBN. Formally:

**Definition of d-separation:** Two nodes X and Y in a BBN are *d-separated* if, for all paths between X and Y, there is an intermediate node A for which either:

1. the connection is serial or diverging and the state of A is known for certain; or

2. the connection is diverging and neither A (nor any of its descendants) have received any evidence at all.

## 9.9 Types of reasoning permitted in BBN's

When using BBNs we are interested in making predictions about uncertain quantities conditioned upon some evidence. Mathematically the meaning of the “|” in  $p(A | B)$  is straightforward but when modelling the real world we must be careful to ensure that conditioning is employed only to model sensible statements about the world. There are a number of types of reasoning that can be considered as valid conditional propositions in a BBN and the purpose of this section is to provide an overview of what the permitted reasoning types are and outline their differences. The different modes of reasoning described are:

*Causal*: effect determined by cause, e.g. accident caused by fault introduced by system designer (Section 9.9.1);

*Statistical*: chance of event determined by population of possible events and sampling method, e.g. probability of failure determined by number of demands and number of failures observed (Section 9.9.2);

*Structural*: phenomena determined by overall structure to which it belongs or object/event or determined by properties of class of objects/events to which it belongs. E.g. failure of module X determined by error introduced in module Y or reliability of system A similar to reliability of system B where A and B belong to class of systems C (Section 9.9.3);

In Section 9.9.4 we explain how these different types of reasoning are handled by the SERENE idioms.

### 9.9.1 Causal determination

Causal reasoning has been advocated as a central tenet of BBN modelling (Pearl 88). From this viewpoint the topology of a BBN is said to reflect the causal structure of the real world and as such provides an explanatory framework for inference and prediction. This interpretation of causality has its roots in Locke's epistemological view of causality as a property of the world rather than simply a way of describing the world. Here causation is not only a component of experience but also an objective form of inter-dependence between objects and events. This view contrasts with that of Hume which has enjoyed considerable influence in modern science and holds that causation is a purely mental construct where “causes” and “effects” are merely connected rather than produced one by the other. Philosophical debates about the true nature of causality are outside the scope of this work but the interested reader might wish to turn to (Bunge 79) for a detailed discussion of the pros and cons of each school of thought. It is sufficient to say that for our purposes the idea that causation as a property of the real world holds some attraction since we are interested in predicting/explaining properties of real systems.

Causal structuring in BBNs helps embody theories about how the world operates as well as encoding uncertainties about its operation. Such a view contrasts sharply with classical statistical analysis where correlation between variables indicates mere phenomenological connection rather than causation. Any causal explanation is then regarded as purely subjective. It is this key distinction that gives BBNs the edge over statistical analysis. Structuring models using causal ideas is more natural for the expert since descriptions of reasoning are easily explained by comparison with reality. On the other hand in statistical modelling statements about the world can often be hidden in mathematical parameters which have no direct physical meaning. Likewise BBNs offer an advantage over rule based expert systems in that causality offers different experts a common medium to compare their reasoning. Rule



based expert systems encode the mental rules of the expert independently of whether these rules are consistent with the real world.

### 9.9.1.1 Using Time and Physical Relationships to structure causal relations

Time forms a key component in causal reasoning. We organise our explanations of events according to their occurrence *in* time. Thus effects follow causes because causes occurred at an earlier time or concomitant with the cause. Such a seemingly trivial observation should not be overlooked. It is all too easy when involved in BBN modelling to get lost in a web of possible relationships between variables without recognising that the time sequence underlying the creation of artefacts and events can be used to organise the cause-effect chain.

Example: We might wish to know the probability that a system is safe based on available information {design complexity, requirements quality, designer competence and testing quality}. If we rank these according to some notional time line such as the software life-cycle we might acknowledge that

- a) requirements quality and designer competence might be causes of design complexity since they both precede it;
- b) testing quality and design complexity might be causes of system safety since they both precede it.

Another important component in causal reasoning is physical connection. Events, artefacts and causal agents (E.g. people, machines) must be connected in order to interact. After all it would make no sense to say that the requirements quality directly caused a system to be unsafe because there is no physical connection between the requirements specification and the system's unsafe state. Instead we can say it is an indirect cause of the unsafe system because the intermediate effects partly caused by the requirement specification quality caused the system to be unsafe. Where we can identify direct physical connections we should use these to help identify the causal structure of the BBN.

### 9.9.1.2 Necessary and Sufficient conditions

In BBNs we model causality by use of the conditional statement  $p(A | B)$ , meaning the probability of A given we know B is true. In causal language this translates "B causes A with probability  $p$ ", where  $p$  indicates the degree of belief or frequency with which B indeed causes A. However, knowing B causes A, from the BBN graph topology, is not enough to know exactly *how* A causes B; that is when an effect is a necessary consequence of a cause or where the cause is a sufficient condition for an effect.

Example: consider the causal propositions  $p(\text{failure in software} | \text{fault in software})$ . If we had with hindsight experienced a failure in the software we would say it was because of a fault although beforehand we might acknowledge situations where other variables might cause the failure, such as operator error or where the fault does not cause the failure because of some internal fault tolerance mechanism.

We can think of the number  $p$  as an indication of how often B by itself is sufficient to cause A and  $(1-p)$  as an indication of other factors that might determine A but are unknown. The extent of causal determination is therefore indicated by both the topology and the values for the probabilities within the BBN. Let us consider a more complex proposition such as  $p(A | B, C)$ . From this information alone we know that B and C cause A but we cannot tell which combinations of B and C events cause A. The necessary condition for A might be B AND C to occur or even B NOT C. The power of BBNs partly lie in their ability to encode the many possibilities of causal interaction amongst variables within manageable, conditional probability structures.

### 9.9.1.3 Types of causal determination

There are a number of types of causal determination we might want to model in BBNs. Again these groupings serve to form convenient labels rather than provide an exhaustive list of mutually independent categories.

*Natural*: here we model the effects of natural, physical processes which operate without motive or intent.

Example: smoking causes lung cancer, or an earthquake causes a nuclear protection system to fail.

*Productive*: here we model the effects of an action, by man or machine, on the transformation or creation of an artefact or event. Here we might name the man or machine the causal agent i.e. that which directly caused the effect.

Example: The designer designs a piece of software to meet a requirements specification or a programmer updates a piece of code to remove some faults. We might model these as  $p(\text{quality of software} \mid \text{quality of requirements, ability of designer})$  and  $p(\text{\# dormant faults} \mid \text{\#fault removed, \#previously dormant faults})$

*Accidental*: here we model the effects of an unintended action - mistake, slip or error.

Example: For instance we might find that an unintended consequence of a design process is the introduction of faults in the product. This might be represented by  $p(\text{\# faults introduced} \mid \text{design process is poor})$ .

*Purposive*: here we model the effects of a deliberate intervention in a situation.

Example: For instance we might realise that  $p(\text{system failure} \mid \text{system complexity}) > c$ , where  $c$  represents some acceptability threshold. Thus we decide to do one of two mitigation actions a) introduce a new feature into the system thus changing the system's complexity and by inference the probability of system failure, or b) introduce a new defence such as training the operator to avoid system failures when certain functions are used within the system). In this way the BBN would be updated to reflect the new situation either by withdrawing an observation and replacing it with a new one or introducing a whole new variable and updating the conditional probability model.

#### 9.9.1.4 Multiple views of causal structure

We may consider causality to be an objective property of the real world but this does not mean to say that different observers might not have different causal models for the same phenomena. The extent and richness of a BBN will be determined by how much experience the expert has of the situation and the availability of information about that situation.

Example: A software developer may have direct experience of his project and be able to identify the causal chain of events:

poor requirements spec. -> poor design -> poor code -> poor product.

However an independent assessor of the resulting product might not have access to the code or the design or may even have no understanding of the role of the design in the developer's life-cycle. To the assessor the causal chain would look like :

poor requirements spec. -> poor product.

Each model is correct w.r.t their particular viewpoint although the developer's is more correct w.r.t reality since it is a more complete description of the true situation.

Similar considerations apply when we consider the probability numbers themselves. One person's probability number for a conditional event will differ from another's simply because their experience of that event may differ. Thus we must expect different individuals/organisations to arrive at different BBNs based safety arguments to reflect differing experience.

In a BBN it *might* therefore make sense to assume there is a correct BBN i.e. one that is a complete model of all causes and effects. But such an assumption would demand omniscience. Underlying any macroscopic model of causality we might find underlying microscopic relationships determining the macroscopic behaviour. Examples of this abound in economics, sociology, physics and

thermodynamics. The question of appropriate model granularity will be as much determined by practical constraints as theory.

The trade-off between what can practically be modelled and what could possibly be modelled is illustrated by the problem of infinite regression. Every cause could conceivably be considered an effect of some other cause and this an effect of another, and so on *ad infinitum*.

Example: Consider the causal chain:

requirements spec. quality -> code quality -> system reliability -> chance of accident.

We could regress this further to introduce :

analyst training -> spec. quality, and then :

company training policy -> analyst training.

From there we could introduce :

economic conditions -> company training policy, and then :

government policy -> economic conditions, etc.

We would only stop identifying and modelling new causes at the point of diminishing returns. There are practical limits to any holistic approach.

### 9.9.2 Statistical Determination

BBNs can be used as valid representations of statistical determination. Under statistical determination the probabilities of events are determined by the chance of experiencing or selecting a particular event from a population of possible events or from a stochastic process. There are obvious overlaps between statistical and causal models of determination. Statistical models of determination are subsumed by causal models since causal models must also admit to chance (subjective probabilities might reflect the randomness of experience).

Statistical models employ statistics that measure aggregates of multiple instances of individual phenomena. E.g. means, medians and standard deviations are used to characterise populations and samples from populations. Such statistics do not represent direct physical and hence objective quantities and as such do not offer causal explanations for individual events. Also parameters in statistical distributions might also fail to admit to physical interpretation since they may merely be mathematical contrivances.

Example: a piece of software may be subjected to repeated demands. When a demand fails the failure is noted and the testing continues. Each demand is a Bernoulli trial and is sampled without replacement from an infinitely large population of possible demands. The chance of  $m$  failures from  $n$  demands is *defined* by the binomial distribution with parameter  $p$ , probability of failure per demand.

Here the probability of  $m$  failures is wholly determined by the chance,  $p$  of encountering  $m$  failures in  $n$  demands sampled. Each individual failure may have been *caused* by combinations of particular faults and triggering events but this is not admitted within the statistical model. Instead we can reason either about the distribution of probability of failures for the sample or individual statistics characterising the distribution, such as sample variance.

In statistical models the probabilities are wholly defined by the parameters of the statistical model adopted. Other variables cannot be admitted except by extending the model by conditioning the parameters on other, perhaps causal, variables.

Example: Consider again the example above. We might chose to condition the probability of failure,  $p$ , on the faults in the software and the probability with which those faults would be triggered. In this way we extend the statistical model by introducing prior variables which might be considered causal in nature.

Despite the differences between causal and statistical determination there are no special dangers presented by considering statistical and causal determination to be identical. However the clear advantage of statistical models is that they can be used to generate the node probability tables in a BBN using chances generated by the model. Causal determination might complement this advantage by providing sensible interpretations for the parameters.

When encoding statistical distributions in BBNs we must be careful about the introduction of contradictions. For instance, when using the binomial distribution the proposition

$$p(n+1 \text{ failures} \mid n \text{ trials}, p)$$

is a contradiction, since we could never have more failures than trials. Under these circumstances

$$p(n+1 \mid n \text{ trials}, p)$$

would be set to zero since contradictions are impossible.

### 9.9.3 Structural Determination

Structural determination can be modelled in BBNs by employing suitable logical or probabilistic relations. We identify various types of structural determination:

1. deterministic - where relations between nodes are logical or functional;
2. definitional - where node relations define the meaning of other nodes
3. architectural - where nodes are related according to some physical or conceptual pattern
4. analogical - where nodes inherit the attributes of a node class.

#### 9.9.3.1 Deterministic Relations

There may be cases where we might wish to encode logical or functional relationships within a BBN. There are a number of reasons for wishing to use the BBN uncertainty calculus to model deterministic relationships:

1.  $p(A \mid B, C)$  is deterministic but the values for  $B$  and  $C$  may be uncertain. We may wish to calculate the distribution of uncertainty on  $A$  given the uncertainties in  $B$  and  $C$ . However when  $B$  and  $C$  are fixed, known values the value of  $A$  is dependent on the values of  $B$  and  $C$ ;
2. it is convenient to display information such as statistics and functions with the BBN representation rather than resort to some other formalism simply for the sake of it.

In functional/logical determination the state values of the child node are wholly determined by the values of the parent nodes. All of the classical arithmetic and logical operators can be brought to bear when defining functions within BBNs.

Example: the number of residual faults in a software product,  $A$ , is simply the number of faults introduced,  $B$ , minus the number of faults discovered and fixed,  $C$ . A BBN fragment could be produced as:

$$p(A, B, C) = p(A \mid B, C)p(C \mid B)p(B)$$

Both of the contingent relations,  $p(C \mid B)$  and  $p(B)$  are causal or statistical but the relation between  $A$ ,  $B$  and  $C$  is logical because  $A = B - C$ , where  $B \geq C$ .

Note that with functional/logical relations the direction of the arrows in a BBN cease to make sense. However in some cases it is better to favour one direction over another in order to combine BBN subnets without undue complication.

Also note that when modelling functional or logical relations in a BBN it is possible to introduce contradictions into the network. When modelling arithmetical statements such as  $C = A - B$  where  $A, B, C > 0$  we find that the probability of *all* values of  $C$  are zero when  $B > A$  i.e. they are impossible. Clearly this causes problems within a BBN since all conditional probability entries for  $p(C \mid B > A)$  in

an NPT must sum to one. Under these circumstances we are clearly conditioning C on a contradiction. But we can use a trick. We set any value of C conditioned on the contradiction to one and the rest to zero. Upon propagation the contradictions themselves should be impossible and these arbitrary values are multiplied throughout by zero thus maintaining consistency and meaning.

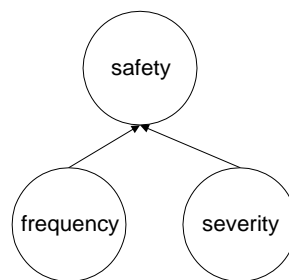
### 9.9.3.2 Definitional Relations

As well as encoding deterministic relations a BBN can model relations between variables which are definitional. (See section 6.5.4.3 - definitional idioms). Here the attribute of a variable may be defined in terms of sub-attributes of the attribute. Again the same reasons hold for representing these in BBNs as held for deterministic relations.

An example of a definitional relation:

Safety is defined as the severity and frequency of failures in a system. This can be modelled in a BBN by

$p(\text{safety} \mid \text{frequency and severity}) = 1$  when  $\text{frequency} \times \text{severity} = \text{safety}$ .



Taxonomies of quality factors such as the Walters and McCall model, (Walters 78) or GQM-type (Basili 88) models fall within the realm of definitional determination.

Note that as with functional/logical relations the direction of the arrows in a definitional part of a BBN also cease to make sense. However in some cases it is better to favour one direction over another in order to combine BBN sub-nets without undue complication. In the above complexity example if the arrows were placed in the opposite direction we might not easily be able to condition complexity on causal variables.

### 9.9.3.3 Architectural Determination

In architectural determination the relations between nodes are defined by some physical or conceptual organisational principle. Here the nodes in a BBN reflect this pattern and the relations show the inter-dependence between them.

When building safety arguments we might be able to exploit information about the architectural organisation of a product in order to structure events or propositions.

Examples: A system consists of modules which dynamically interact to deliver the product's service. These modules may be organised in a hierarchy where failures in modules at the lower levels cause failures in the sub-system level in the hierarchy and so on until they propagate to the highest level. Here we might model the relations using an IS-A or IS-PART-OF relations between the system entities.

Another example of architectural determination is employed in fault tree analysis (FTA) to show how fault modes in different sub-systems can combine to develop into fault modes using AND and OR relations.

### 9.9.3.4 Analogical Determination

Analogical determination involves the use of analogy or metaphor to carry across some of the properties from one class of objects or events to the current object or event under study. Strictly speaking statistical determination also involves extrapolation from one situation to another on the

assumption that the sampled subjects under study remain homogeneous across samples or that there is some reason to expect statistical regularity between situations. However analogical determination differs in that we might know what the differences between situations are - that is the differences between them are not due to random effects influenced by chance but by fixed factors whose effects might reasonably be supposed.

Reasoning from analogy does not imply any physical connection between the situations. Rather we assume that they belong to a class of situations which have similar properties. Knowing the properties of one item from the class, and the differences between this item and the one under study might enable us to inherit many of these properties from the known situation.

Example: The reliability of a database system used by company X is known and has a particular statistical distribution. We are faced with the problem of predicting the reliability of the same database system for company Y. We know that company X and Y are using the database system to perform similar functions and so might reasonably expect that the reliability data gained by company X could be used to predict database reliability in company Y's case.

Clearly being able to assume that, in both cases, reliability data are samples from the same population (a class of companies using the database system perhaps) would be highly beneficial. However rather than employ straightforward induction the use of analogy demands that an assessment be made of the similarity between members of the class, otherwise we would achieve absurd conclusions.

Example: We might know that company Y intend to use the database under a 10 fold increase in the transaction count. Also we might learn that company Y intend to only use the most complex functions of the database system. Therefore despite surface similarities between situations we might decide that straight adoption of company X's reliability data would be unwise.

The assessment of similarity involves the invocation of additional and fixed causal factors, shared by both situations but differing in their intensities, that explain the differences between the situations. We can either use these as additional nodes in the BBN or, for simplicity, group their effects into one single node and call this, for convenience "degree of similarity" or some such like.

Example: Upon recognising the differences between company X and Y's use of the database system we might introduce two causal factors into the BBN to show how these factors would explain the likely differences between them whilst still allowing some of the experience to carry across.

Degree of similarity suggests the use of some multidimensional distance metric to measure the difference between both situations, X and Y. Each causal factor that differs between the situations would be represented by a dimension of the metric. Large differences between causal factors imply a large value for the distance metric,  $D(X,Y)$ , and so low similarity and vice versa.

Using the distance metric approach the types of statements that might be encoded in a BBN would fall into one of three categories:

1. Partial similarity -  $p(X | Y, \text{zero} < D(X,Y) < \text{infinity}) = kp(Y)$ , where k increase or decreases  $p(Y)$  according to how favourable the differences are.
2. Similarity -  $p(X | Y, D(X, Y) = 0) = p(Y)$
3. Dissimilarity -  $p(X | Y, D(X, Y) = \text{infinity}) = p(X)$ .

#### 9.9.4 Idioms and types of determination

The SERENE idioms embody different types of determination discussed above:

1. Statistical and analogical determination is modelled by the induction idiom. Here aggregates of repeated instances of homogenous events are used to predict the likely behaviour of a new

- event. The extent to which this can be done successfully depends on the degree of the similarity between the events ;
2. Causal reasoning is modelled by the process-product and measurement idioms. In the process/product idiom our reasoning connects causes (processes) to consequences (the product of those causes). The measurement idiom relies on the concomitant interaction of the measurement instrument and the artefact under observation to produce an estimate (the consequence);
  3. Deterministic reasoning is modelled by the synthesis/definitional idiom. Here definitions of things are modelled by deterministic functions. Also different variables may be combined in some way for practical purposes (synthesis), where the combination rule is completely certain.

## 10 Appendix D: BBNs and decision making

### 10.1 Introduction

The SERENE toolset is focused on the use of BBNs to support safety argumentation. Given a particular safety-critical system and its associated documentation and test results, the SERENE toolset is intended to help assessors and developers take account of the diverse information about the system and reason about its likely dependability if and when it is used.

In its most basic use the SERENE toolset should therefore provide a prediction of the likely safety of the system under investigation. However, while BBNs provide considerable support for decision making, they do not allow us to incorporate the notion of *preference*. Because of this BBNs cannot, alone, provide a complete solution for the kind of wider decision problems in which a system safety assessment exercise inevitably fits. For example, suppose we wish to determine whether a proposed software-controlled protection system should be deployed in a particular reactor. We could use the SERENE toolset to construct a safety argument of the system, but the result of this exercise is merely one component of the information that a regulator will use before reaching a decision about deployment. The regulator will be interested in other criteria like cost (both economic and political) and functionality. These criteria may have a heavier weighting than the predicted safety level when it comes to making a decision about the nature of deployment. In other words, the safety assessment problem is attempting to predict just a single criteria in what is a multi-criteria decision problem.

In this appendix we provide a model of reasoning about the broader context in which safety assessment is performed and identify the specific role of the SERENE method and toolset. In doing this we clarify misunderstandings about important concepts in dependability argumentation and show how to avoid the confusion and ambiguity associated with much work in this area.

This appendix covers the following areas, using two examples (one safety critical decision problem, and one everyday decision problem) to illustrate the concepts:

- identifying motives/objectives from a given perspective
- the notion of a decision problem and its constituent parts; we use the accepted terminology of multi-criteria decision aid (MCDA). Thus, we introduce the key notions of *criteria*, *constraints* and *actions*.
- defining and measuring criteria
- the key notion of uncertain criteria and inference
- analogy with GQM (Goal-Question-Metric)

### 10.2 The example problems

We will structure this appendix around two examples. One is a safety critical decision problem, and one is an everyday decision problem to which everyone can relate easily. It is important that the everyday example is included in juxtaposition to the safety-critical one because the concepts are much more widely understood and accepted in such a concrete example. The two problem examples are presented in Table 10-1.



**Table 10-1: Two example problems**

Safety critical problem	'Everyday' problem
<p>I am a Government-appointed Regulator for Nuclear Power. It is my job to license computerised equipment for nuclear power plants. It is proposed to deploy a new software controlled protection system in an existing reactor to replace the existing mechanical controlled system. I have the authority to inspect every aspect of the new system's design and test as well as the company that produced it. The problem is to decide whether to:</p> <ul style="list-style-type: none"> <li>• Deploy the new system</li> <li>• Deploy with specified minor changes</li> <li>• Deploy only after specified major changes</li> <li>• Do not deploy, but retain existing mechanical system</li> <li>• Decommission plant</li> </ul> <p>Obviously I have to be assured that the new system is sufficiently safe, but I also have to take account of the cost of the new system and any proposed changes to it, and I have to take account of political requirements and the cost of maintenance (which is expected to be lower with the new system).</p>	<p>I have to get to Heathrow in time for a 10.00am flight to Rome. The problem is to choose both the mode of transport and the departure time. The modes of transport are:</p> <ul style="list-style-type: none"> <li>• Car (i.e. drive myself and park)</li> <li>• Taxi</li> <li>• Train</li> </ul> <p>For simplicity we take the departure times to be 3 discrete intervals :</p> <ul style="list-style-type: none"> <li>• early (meaning between 5 and 6 AM)</li> <li>• medium (meaning between 6 and 7 AM)</li> <li>• late (meaning between 7 and 8 AM)</li> </ul> <p>Obviously I want the journey to be as comfortable and cheap as possible (within certain constraints) and I would seek to minimise both the journey time and the waiting time (again within certain constraints). But I have to take account of certain conflicts and also factors like how much luggage I am carrying as well as the weather, roadworks, and the rush-hour traffic building up after 6AM).</p>

The key concepts to be defined in this appendix are shown in summary form in Table 10-2 for each of the two examples. In the rest of the report we explain these concepts in more detail.

**Table 10-2: The key concepts summarised**

	Safety assessment example	Travel example
<b>Objective/Motive</b>	To ensure that a safe protection system is installed in a nuclear plant at reasonable cost	To get to Heathrow cheaply, comfortably and quickly in good time to catch the Rome flight
<b>Perspective</b>	Decision maker: The regulator Key stakeholders: the Government and the local community	Decision maker: The traveller Key stakeholders: The people meeting the traveller in Rome
<b>Decision problem</b>	To decide if the proposed computer protection system is appropriate for deployment	To determine the most suitable mode of transport and start time
<b>The set of possible actions</b>	Deploy; Deploy with specified minor changes; Deploy only after specified major changes; Do not deploy, but retain existing mechanical system; Decommission plant	The set of pairs of the form (A,B) where A is the transport type (own_car, taxi, train) and B is a start-time (early, medium, late).
<b>Criteria (functions defined on actions)</b>	Safety, functionality, cost (financial), cost (political) For example, safety might be defined as the probability of failure on demand (pfd); functionality might be defined as either 'satisfactory' or 'unsatisfactory'; financial cost might be the cost in pounds including life-cycle maintenance costs. Note that the value of some criteria for some actions may never been known with certainty.	Journey_time, wait_time, cost, comfort For example: journey_time might be defined as the elapsed time in minutes between leaving home and arriving at the check-in desk; comfort might be defined as one of 'low', 'medium', or 'high'.
<b>Constraints (properties of criteria that you specify as desirable)</b>	Examples: Safety < 10 <sup>-3</sup> pfd Cost < £10 Million	Examples: Wait_time > 10 minutes Cost < £50
<b>External factors (variables you cannot control, but which can influence the value of criteria for a given action)</b>	Test results Test effort Experience of development team Quality of methods used	Weather Roadworks Train strike
<b>Internal factors (variables you may be able to control and which can influence the value of criteria for a given action)</b>	Examples: System load (you could insist that the system be deployed providing that it is subject to a maximum number of hours of continuous use) System environment (you could specify that it can be used for reactor A but not reactor B).	Examples: Leaving time (if there is bad weather you could leave earlier) Amount of luggage to take (given information about roadworks you might be able to cut down on luggage sufficiently for you to be able to go by train)

### 10.3 Identifying objective and perspective

The *objective* of a decision problem is the ultimate reason you are interested in the problem. In other words it is your motive for solving it. The objective is always with respect to a particular *perspective*, the most important component of which is the person or party making the decision. The regulator will have an entirely different perspective of the problem of ensuring that a safe protection system is

installed in a nuclear plant compared to the system supplier. Similarly, the traveller will have an entirely different perspective of the problem of getting to Heathrow compared to a London taxi driver.

**Example:** The objective (from the perspective of the traveller) of the travel problem is the one shown in Table 10-2; thus the objective is *not* to have a comfortable journey, even though this is one of the factors we may consider in our final choice. The objective of the safety assessment problem (from the perspective of the Regulator) is the one shown in Table 10-2; it is *not* to deploy the new system, even though this is one of the options available.

The perspective of the problem incorporates not only the decision maker, but also the *stakeholders*. These are the parties most affected by the chosen outcome *and* whose viewpoints will need to be considered in arriving at a decision. Different stakeholders may have quite different interests, which in turn may be quite different from those of the decision maker.

**Example** In our travel problem let us assume that the traveller is attending an important business meeting in Rome. The most important stakeholders are a) the other people (that is, the Romans) who will be at the meeting and b) the traveller's boss. The Romans are really only interested in the traveller getting to the airport in time for the flight; if it was their choice alone they would insist on the traveller leaving as early as possible by the quickest mode of transport. The traveller's boss on the other hand is interested in cost, while the traveller himself is interested in the comfort of the journey and not having to wait too long.

**Example:** In the safety assessment problem (from the perspective of the regulator) the Government and local community are the main stakeholders; they have similar overall objectives, but they may have radically different ways of judging the best outcome. For example, the local community probably does not consider cost as an important factor at all, but the Government certainly will. The Regulator's challenge is to take account of these different considerations as well as his own.

It is just as important to ensure that we know who are *not* considered to be stakeholders, as this is a crucial step in scoping and simplifying the problem. Generally a party which is affected by the decision should be excluded from being considered a stakeholder if either

1. their viewpoints/needs are not relevant; or
2. their viewpoints are fundamentally inconsistent with that of the decision maker or an accepted stakeholder (there is no point in attempting to solve a decision problem when there is *no* solution which could be accepted by *all* the stakeholders).

**Example** In the travel problem it is reasonable to exclude the London taxi-drivers from the set of stakeholders. Although they *may* be affected by the outcome (in the sense that one of them may benefit from a high-paying job) there is no need for the traveller to consider the needs of the taxi drivers. The traveller certainly does not owe them a living. On the other hand the viewpoint of the traveller's husband certainly is relevant. But, if the husband is fundamentally opposed to her travelling abroad on business then there is little point in including him as one of the stakeholders; his only interest is in stopping the trip completely and this is incompatible with the interests of the traveller and other stakeholders. It would be impossible to arrive at a decision that satisfied them all.

**Example** In the safety problem it is reasonable to exclude the system developers from the set of stakeholders. They will be affected by the outcome but it is not necessary to consider their needs (which in this context is simply to sell the system). The Regulator does not owe the developers a living.

The above examples confirm the importance of identifying a clear and appropriate objective from a clearly defined perspective. Many real-life decision problems fail on this first hurdle. If not done properly the entire safety assessment process could be a costly waste of time. We believe that in many cases where safety assessment is being performed the motive and perspective is not at all clear.

## 10.4 The decision problem and its constituent parts

Having identified the objective and perspective our next task is to define the decision problem that we need to solve to meet the objective from the given perspective. Although it is useful to express the decision problem in the kind of summary prose shown in the third row of Table 10-2, the decision

problem is only truly well-defined once we identify the following (using the standard terminology of multi-criteria decision theory [Vincke 1992]):

- the set of possible (mutually exclusive) *actions* we can take (these are the alternatives)
- a set of *criteria*, which are functions defined on actions
- a set of *constraints* which are properties of the criteria; these can also be thought of as *preferences*

In both of the examples we have a finite set of actions, but generally the set of actions could be infinite and even continuous (in the travel example, if we decided that the departure time should be the *real* time in the interval between 6.00 and 9.00AM then the set of actions is continuous).

If there were only a single criterion with which to judge the actions it would not be difficult to solve the decision problem - we would just choose the action that returned the 'best' value for the criterion. In the safety example, if safety (defined as the probability of critical failure on demand) really *were* the only criterion on which we had to choose our actions then we would simply choose the action with the highest value of safety. Unfortunately, we also have to consider other criteria like cost (both economic and political) and functionality. Inevitably some of these will be conflicting; the safest system may not be either the cheapest or the one with the most functionality. Generally, we wish to optimise a number of possibly conflicting criteria. We may be guided in our decision choice by a number of constraints (which may be thought of as preferences). These are properties of the attributes that we regard as necessary (from the chosen perspective) - any action which fails to satisfy a constraint for any criteria is automatically rejected. Unfortunately, introducing constraints generally provides only a small amount of help for us to choose the optimal action.

There is an extensive body of work, called multi-criteria decision aid (MCDA) [Vincke 1992] which does provide concrete help for solving decision problems. MCDA includes such well known techniques as linear programming (only relevant when the criteria all have equal weighting and can be measured on a ratio scale) and other more recent techniques which help us to solve problems in more general cases when we do not have such ideal criteria. For example, the Analytical Hierarchy Process (AHP) [Saaty 1980] is a popular (albeit very crude) technique that includes a means of weighting criteria against each other at one level and actions with respect to particular criteria at a lower level. More rigorous approaches, such as *outranking methods*, avoid the theoretical limitations of AHP, but do not guarantee a linear ordering of the actions; in other words you may still end up with having to find some other method of choosing between 'equally acceptable' best actions.

MCDA plays a complementary role to the SERENE method, but it has limitations that we must (and fortunately can) take account of. Specifically, the vast body of MCDA techniques makes two critical assumptions:

1. That the relevant criteria are well defined (and hence for a given action  $a$  it is obvious how you can compute  $g(a)$  for a given criteria  $g$ )
2. That the relevant criteria are certain (and hence for a given action  $a$  and criteria  $g$  the value  $g(a)$  is deterministic rather than stochastic).

**Example:** The following is a classical MCDA problem: Choose one from a set of cars to buy based on the criteria: age, price, engine size, petrol consumption at 30mph, maximum speed. For a given car  $x$  each of the values  $age(x)$ ,  $price(x)$ ,  $engine\_size(x)$ ,  $petrol\_consumption(x)$ ,  $maximum\_speed(x)$  are both well-defined and certain. Thus, for each 'action' (in other words each car) we can construct a well-defined vector of values corresponding to the various criteria.

The SERENE method provides precisely the ammunition for dealing with the important cases when these assumptions are not valid.

## 10.5 Defining criteria

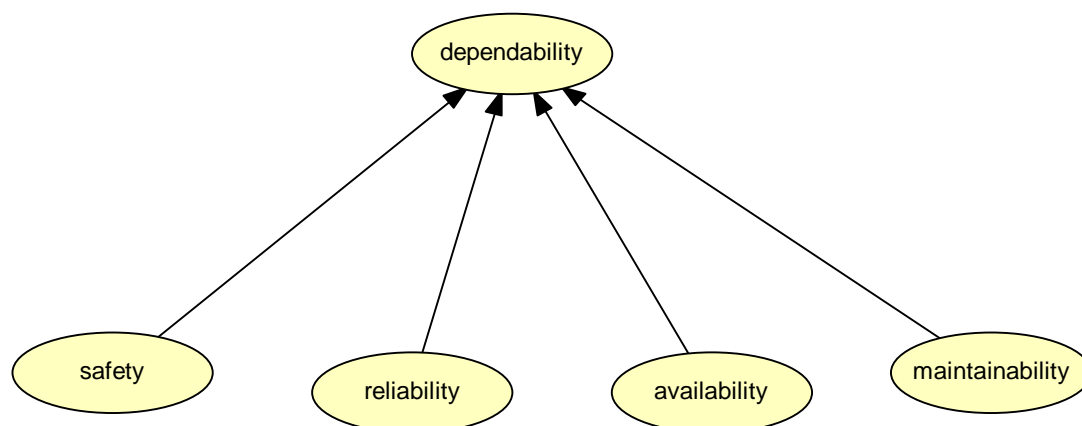
The theory of MCDA assumes that criteria are always well-defined. Our examples confirm that this is not true for real-life problems.

**Example** In the travel example the criteria are *journey\_time*, *wait\_time*, *cost* and *comfort*. Recall that, formally, a criterion is a function from the set of actions into some ordered set. Thus, for a well-defined criterion we first need to define the ordered set (which is the range of the function and which may also be thought of as the ‘measurement scale’). For *journey\_time* and *wait\_time* this might be the set of positive real numbers or we might be content with a simple ordinal scale like {short, medium, high}. However, in either case the value of the function for each action must be unambiguous. For example, we might define *journey\_time* as the elapsed time in minutes from leaving our front door until arriving at the airport check-in desk. For a given action, such as (taxi, early) we could then compute the value *journey\_time*(taxi, early) as the actual time in minutes taken by taxi when we leave early. The criterion *wait\_time* does not present too many problems, but even an apparently well understood criterion like *cost* must be very carefully defined. For example, does ‘cost’ mean just price paid on the day or does it also include overheads (such as a sum for ‘wear and tear’ when the transport type is your own car)? When it comes to the criterion ‘comfort’ it is not at all clear how the criterion should be defined even if we could agree on an appropriate measurement scale.

**Example** In the safety example none of the criteria *safety*, *functionality*, *cost (financial)*, *cost (political)* are easily defined. For example, we have suggested that safety might be defined as the probability of failure on demand (pfd). However, this is clearly a function that cannot be properly computed for all of the possible actions - in fact it is only known for certain for one of the possible actions - ‘Decommission’ where safety is perfect (pfd=0) simply because there are no demands.

In summary the problem is that, in most real-world decision problems we will be interested in criteria which are not necessarily well-defined (in the sense of MCDA). For simplicity let us call such criteria ‘vague’. It is beyond the scope of this document to provide a set of guidelines on how to define and hence measure vague criteria (readers should consult [Roberts 1979] for a good general account and [Fenton and Pflieger 1996] for an account in the context of software engineering). But the following points are especially relevant for SERENE:

1. Vague criteria are often decomposed into lower level attributes that are assumed to be well-defined. For example, according to [Laprie 1992], *system dependability*, is decomposed into *safety*, *reliability*, *availability*, and *maintainability*. It is tempting to draw this as shown in Figure 10-1.



**Figure 10-1: Definition of system dependability (can be viewed as an instantiation of the definitional/synthesis idiom)**

It is important to note that the decomposition alone is not *sufficient* to define the higher level criterion (for example, there may be many ways to define system dependability as a combined measure of the lower level attributes). What you must also *not* do is confuse the decomposition (and any subsequent refined definition) with the notion of *causal dependence*. Figure 10-1 is an

instantiation of the definitional/synthesis idiom. Dependability is not *caused* by safety, reliability, etc. but is merely defined in terms of these attributes.

2. Defining criteria is the same thing as defining measures for attributes. The rules of measurement theory (notably the representation condition) govern when we have truly defined a measure for an attribute and what the appropriate scale type is. Often a simple ordinal scale may be sufficient for our purposes.
3. A measure for an attribute should never be seen as *defining* an attribute (this is one of the most important lessons of measurement theory). Thus, for example, the notions of ‘comfort’ and ‘dependability’ exist independently of any means of measuring them. While it may be sufficient for our purposes to measure comfort on the simple scale {low, medium, high} this measurement does not replace all existing intuition about comfort and therefore does not re-define it. This is a surprisingly difficult concept to grasp. For example, many people assume that the weight of a person is *defined* by the number of pounds that an accurate set of scales will record when the person steps on them; this is quite wrong - weight is an attribute that can be measured in many different ways, of which the pounds and scales case is just one.
4. In some situations we may need to define a very crude measure of a vague or complex attribute. For example, rather than defining an indirect measure of *dependability* using the Laprie decomposition above, it maybe be sufficient to provide a crude direct ordinal scale measure such as {low, moderate, average, high, very high}.
5. Vague and complex attributes are ones of whose *definition* we are uncertain. This must not be confused with uncertain inference about the attribute (we deal with this key notion of uncertainty in the next section). For example, there is no uncertainty about how to define and measure a person’s weight, but if we wanted to predict a person’s weight in two years time then that value is uncertain. Conversely, although we may be unsure how to define and measure system safety, there is no uncertainty about the safety of a system that has been built, used, and decommissioned (it is just that we may not agree on how to measure it). What we need to be careful about is to distinguish between the different types of ‘uncertainty’ that arise in decision problems:
  - Uncertainty in meaning - where we have a vague criteria that we do not properly define
  - Imprecision - where our measurement process is inaccurate even though it may be well-defined
  - Uncertainty in inference

## 10.6 Uncertain criteria and inference

In this section we consider:

- Criteria that require uncertain inference
- External factors
- Internal factors

### 10.6.1 Criteria that require uncertain inference

In classical MCDA, once a criterion  $g$  is defined (even if it is vague in the sense discussed in the previous section) it is assumed that for a given action  $a$  the value of  $g(a)$  is certain. For example, it is reasonable to assume that the age, price, and engine size, of each of a set of cars that we may wish to buy, are defined with certainty. However, in general many key criteria cannot be computed with any kind of certainty. Rather, they require some kind of uncertain inference. Even in our car example a criteria like petrol consumption will be uncertain, being dependent on (among other factors) the speed and road conditions.

**Example** In our travel example each of the criteria is uncertain (although for simplicity our later example assumes that the *cost* is certain for a given choice). For example, *journey\_time* for a specific choice of

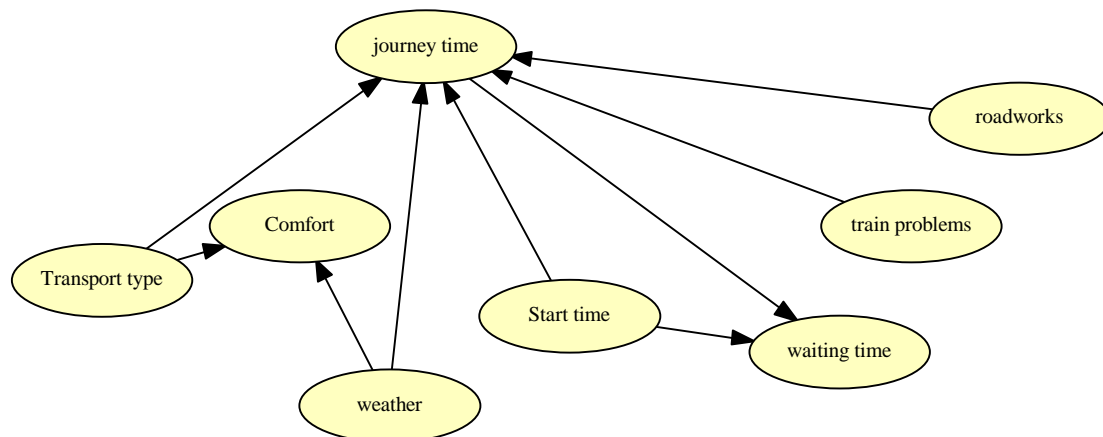
action will vary according to whether or not there are train problems (like a signal failure or industrial action), roadworks, or bad weather. Similarly, in our nuclear example, the safety criteria uncertain (for a given choice of action) because it will vary according to many factors relating both to its development and conditions of future operational use.

Whereas traditional MCDA assumes that all criteria can be measured with certainty, it is clear that any interesting problem will involve key criteria that are inherently uncertain. Having a specific method for handling this uncertainty is where, of course, the BBNs come into the SERENE method. We next consider the different types of factors that lead to the need for uncertain inference.

### 10.6.2 External factors

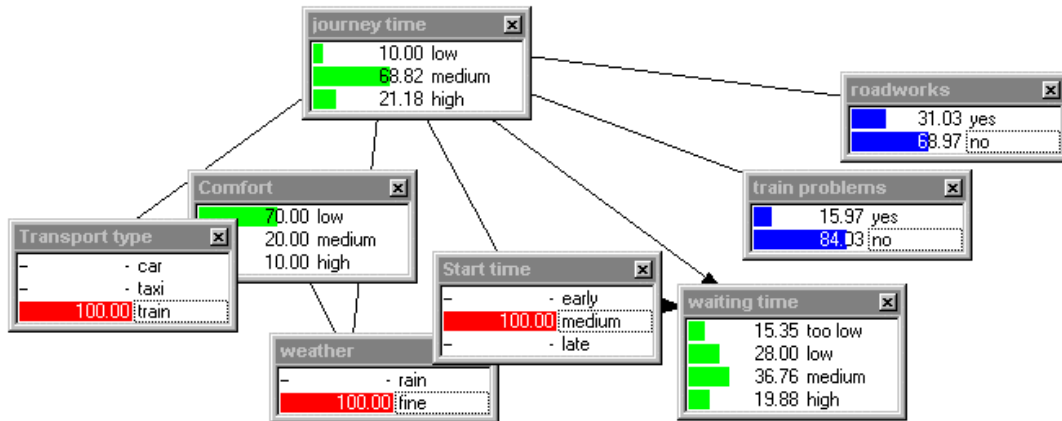
External factors are variables that the decision-maker cannot control, but which can influence the value of a criterion for a given action. These factors, along with the uncertain criteria themselves, will form the set of nodes in a BBN for predicting the values of the uncertain criteria. Often external factors can be thought of as *risk factors*.

**Example** In our travel Example there are three uncertain criteria (journey\_time, comfort, and waiting\_time). In Figure 10-2 we have produced a single BBN which incorporates all of these uncertain criteria with the factors which impact on them. The external factors are *weather*, *train\_problems*, and *roadworks*.



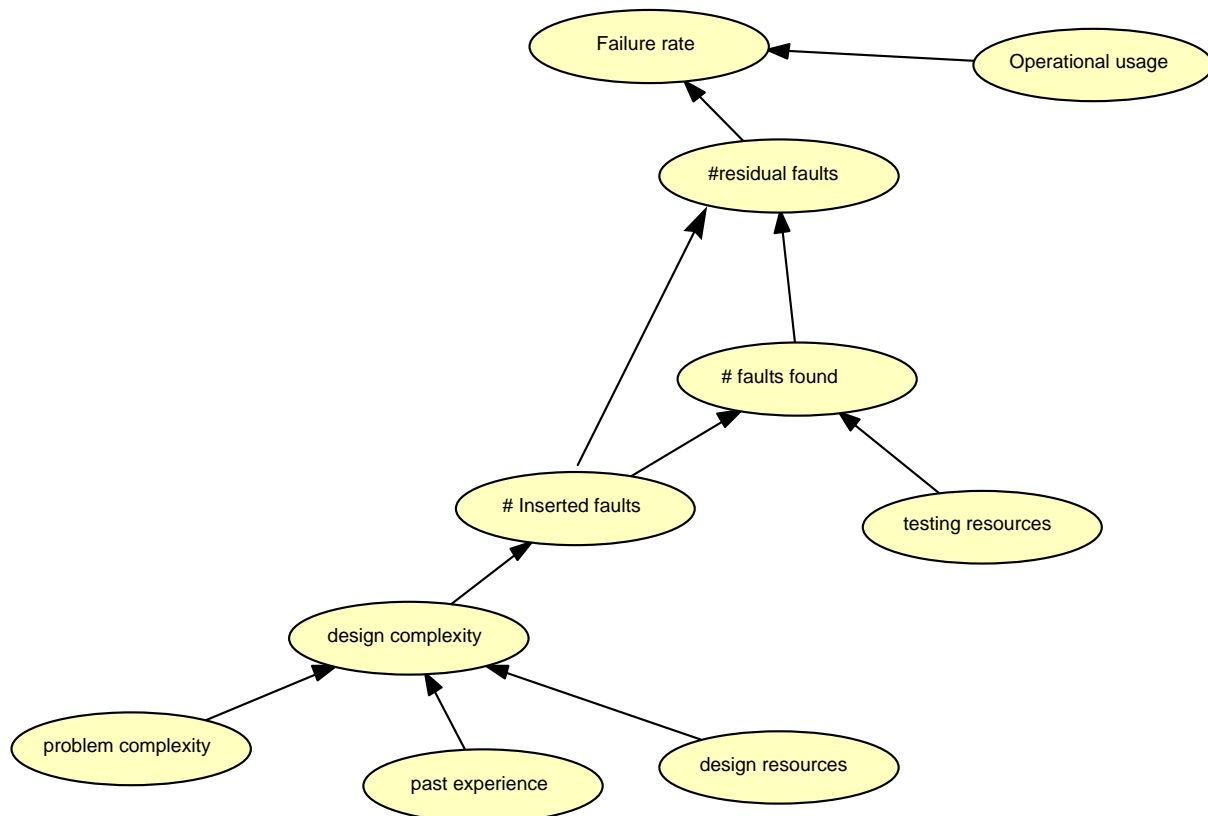
**Figure 10-2: BBN for predicting the uncertain criteria in the travel example**

In Figure 10-3 we use the BBN of Figure 10-2 to calculate values of the uncertain criteria. In this scenario we decide to go by train and leave early. We do not know for sure if there are any train problems but enter some likelihood values. The weather is fine. In this scenario the most likely waiting time will be medium, but note that there is 0.15 probability that we will arrive too late.



**Figure 10-3: Calculating values of the uncertain criteria**

**Example** In a safety-critical software system, the key uncertain criteria that we wish to predict might be the failure rate in operation. Clearly this is going to be dependent on the external factor operational\_usage as shown in the BBN of Figure 10-4. Note that it is also directly influenced of course by the number of residual faults, which in turn is influenced by other factors. From the viewpoint of the system developer the only other factor which is external is problem\_complexity. From the viewpoint of a regulator all of the factors are external.



**Figure 10-4: BBN to predict software failure rate**



### 10.6.3 Internal factors

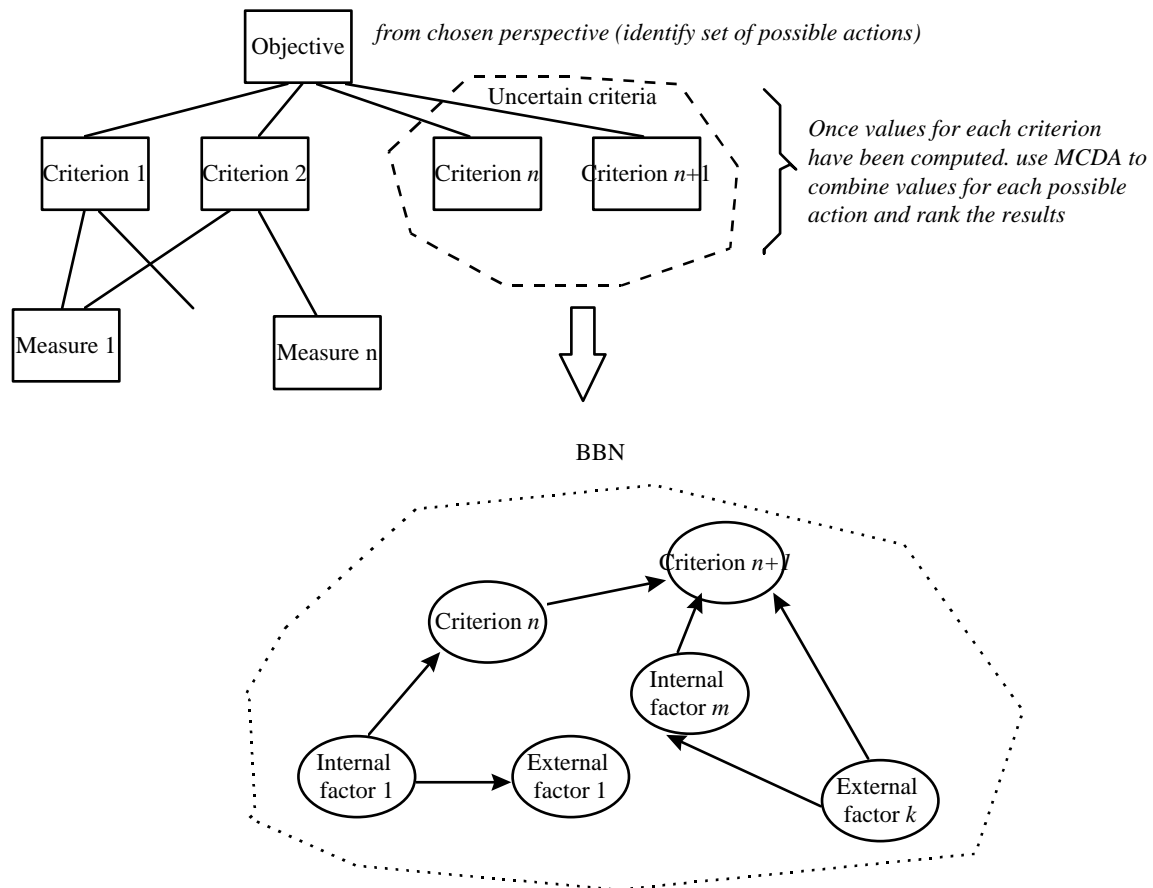
Internal factors are variables that the decision maker can control, and which can influence the value of a criterion for a given action. Normally internal factors can be regarded as risk minimisation factors. In the nuclear example the Regulator is interested in the uncertain safety criterion. He cannot control factors relating to the completed development and testing process, but he could insist on certain modifications being done before licensing the equipment, and he could insist on certain restrictions on its operational use. Thus factors like 'operational use' and 'product modifications' are, from the perspective of the Regulator internal (risk reduction) factors. Interestingly, if we consider this example from the perspective of the system developer, then the roles of the external and internal factors are mostly reversed. The developer has control over the quality of the design and testing process (including of course how to assign resources), but has no control over its proposed operational usage.

In constructing a BBN for the uncertain criteria, both the external and internal factors that affect the criteria will be included. We will also generally include one or more nodes representing the actions to be chosen, since obviously these will affect the criteria (they are, in a sense the 'ultimate' controllable variables, but we do not classify them as internal, because our definition of internal assumes a fixed action has been chosen).

**Example** The BBN for our travel example (Figure 10-2) includes two nodes *transport-type* and *start\_time* which together represent the choice of actions available to us - recall that our actions consisted of the set of tuples (transport-type, start-time). This BBN does not include any genuine internal factors, but it is not difficult to think of examples that could have been included like *amount\_of\_luggage*. (which is generally not a fixed constraint - if we receive evidence of roadworks we could reduce the amount of luggage we take to enable us to travel by train.

## 10.7 Analogy with GQM

The approach to decision problem solving we have described has a close analogy with GQM (Goal Question Metric) [Basili and Rombach 1987]. You start by asking what are your goals - that is, the objective for your decision. Next you have to consider the perspective (for example, the Regulator as opposed to the Developer). Next you ask 'questions' which we think of as identifying the set of possible actions and then the set of criteria that distinguish these actions. At this point traditional GQM would simply insist that you define the underlying measures for your chosen criteria and traditional MCDA would then provide a means of combining the resulting measures for each action and provide a means of ranking the actions as a result. The key difference we have is that while some criteria may be certain, and hence depend on a traditional approach to measurement, many key criteria will require uncertain inference. These criteria will depend on various external and internal factors that we have to identify. Having identified them we use them to make predictions of the values of the uncertain criteria for the different actions. We do this by using a BBN. This enables us to compute values for each criterion for a given action and we can then apply traditional MCDA techniques to combine the values and rank the actions. The process is shown schematically in Figure 10-5.



**Figure 10-5: How the SERENE approach fits in with GQM and MCDA**

## 10.8 SERENE and decision making: concluding remarks

The use of BBNs in the SERENE method helps us to make predictions about uncertain factors like safety of a proposed system. While this is extremely important this is only one component of a broader decision making process. In this appendix we have described the broader decision making context and have provided a rigorous method for tackling it. In summary this method consists of the following:

1. Agree on the *objective* for your decision problem
2. Make sure you know *from whose perspective* the problem must be solved. Thus identify carefully both the *decision maker* and the *stakeholders*.
3. Identify the set of possible *actions* that will form the set of alternatives available to you.
4. Identify the set of *criteria*, that is the attributes of actions, that will determine your choice.
5. Identify any fixed constraints, that is properties of criteria that must be satisfied for any chosen action.
6. Determine which criteria are uncertain (that is, can only be calculated for a given action using uncertain inference) and which criteria can be calculated with certainty.
7. For the certain criteria ensure that you have appropriate definitions that enable an unambiguous mapping of actions into a totally ordered set. There is no harm if the ordered set is a simple ordinal scale as long as clear rules are defined for the mapping. If a criterion is vague or complex, it may be necessary to decompose it into lower level attributes. However, all definitions of the certain criteria (including any decomposition) must be done separately from the BBN.

8. For the uncertain criteria, identify the factors that will affect them. There will generally be external factors that you cannot control and some internal ones that you can control. Having identified them construct one or more BBNs for the various factors and uncertain criteria. The BBNs should also include nodes corresponding to the actions themselves.
9. As a result of steps 7 and 8 you will be able calculate a value (within some probability bounds in the case of the uncertain criteria) for each criterion for a given action. This means that you can apply traditional MCDA techniques to combine the values for a given action and then to rank the set of actions. In the case of the uncertain criteria you could, for example, apply values for 'most likely' as well as the upper and lower bounds. If the result of the MCDA analysis produces a unique 'best' action which satisfies all of the defined constraints then you are done. If not you will have to relax various constraints or introduce new actions (MCDA deals with these issues and it is beyond the scope of this paper).

## 11 Appendix E: Biases and fallacies in reasoning about probability

A critical part of any BBN model are the node probability tables. In many situations we have to rely on experts to provide the (subjective) probability values. In fact people are not very good when it comes to estimating or reasoning about probability. In this section we consider the common problems, notably types of bias, that affect probability judgements. It is important to be aware of these biases in order to make adjustments when eliciting probabilities from experts. We cover the following:

- Representativeness
- Denial of certainty
- Availability
- Adjustment and Anchoring
- Conjunction fallacy
- Hindsight bias
- Difficulties in assessing variance, covariance and correlation
- Conservatism
- Overconfidence
- Fallacies associated with causal and diagnostic reasoning

### 11.1 Representativeness

*Representativeness* is the collective term used to describe the following range of fallacies people make when judging probabilities.

- The problem of base-rate neglect
- Insensitivity to prior probability of outcomes
- Insensitivity to sample size
- Misperception of chance and randomness

We consider these in turn:

#### 11.1.1 The problem of base-rate neglect

Consider the following problem

*A particular heart disease has a prevalence of 1/1000 people. A test to detect this disease has a false positive rate of 5%. Assume that the test diagnoses correctly every person who has the disease. What is the chance that a randomly selected person found to have a positive result actually has the disease?*

This question was put to 60 students and staff at Harvard Medical School.

Almost half gave the response 95%.

The average answer was 56%.

The correct answer is 2%. It was given by just 11 participants. An informal way of explaining this result is to think of a population of 10,000 people. We would expect just 10 people in this population to have the disease. If you test everybody in the population then the false positive rate means that, in addition to the 10 people who do have the disease, another 500 will be wrongly diagnosed as having it. In other words only about 2% of the people diagnosed positive actually have the disease. When

people give a high answer like 95% they are ignoring the very low probability (i.e. rarity) of having the disease. In comparison the probability of a false positive test is relatively high.

A formal Bayesian explanation is as follows:

let  $A$  be the event 'person has the disease'

let  $B$  be the event 'positive test'.

We wish to calculate  $p(A|B)$ .

First we note that:

$$p(A|B) = 1 - p(\text{NOT } A|B)$$

In fact it is easier to calculate  $p(\text{NOT } A|B)$ . By Bayes this is:

$$p(\text{NOT } A|B) = \frac{p(B|\text{NOT } A) \cdot p(\text{NOT } A)}{p(B)} = \frac{p(B|\text{NOT } A) \cdot p(\text{NOT } A)}{p(B|A) \cdot p(A) + p(B|\text{NOT } A) \cdot p(\text{NOT } A)}$$

Now, we know the following:

$$p(A) = 0.001$$

$$p(\text{NOT } A) = 0.999$$

$$p(B|\text{NOT } A) = 0.05$$

$$p(B|A) = 1$$

Hence:

$$p(\text{NOT } A|B) = \frac{0.05 \cdot 0.999}{0.001 + 0.05 \cdot 0.999} = 0.9804$$

Hence  $p(A|B)$  is approximately 0.02.

### 11.1.2 Insensitivity to prior probability of outcomes

Suppose you are given the following description of a person:

*'He is an extremely athletic looking young man who drives a fast car and has an attractive blond girlfriend.'*

Now answer the following question:

*Is the person most likely to be a premiership professional footballer or a nurse?*

If you answered *professional footballer* then you were sucked into this particular fallacy. You made the mistake of ignoring the base-rate frequencies of the different professions simply because the description of the person better matched the stereotypical image. In fact there are only 400 premiership professional footballers in the UK compared with many thousands of male nurses, so in the absence of any other information it is far more likely that the person is a nurse.

The hypothesis that people evaluate probabilities by representativeness in this way (thereby ignoring the prior probabilities) was tested by [Kahneman and Tversky 1973]. Subjects were shown brief personality descriptions of several individuals, allegedly sampled at random from a group of 100 professionals - all engineers or lawyers. The subjects were asked to assess, for each description, the probability that it belonged to an engineer rather than a lawyer. There were two experimental conditions:

1. Subjects were told the group consisted of 70 engineers and 30 lawyers
2. Subjects were told the group consisted of 30 engineers and 70 lawyers

The probability that a particular description belongs to an engineer rather than a lawyer should be higher in 1 than in 2. However, in violation of Bayes rule, the subjects produced essentially the same probability judgements. Subjects were apparently evaluating the likelihood of a description being an engineer rather than a lawyer by the degree to which it was representative of the two stereotypes; they were paying little or no regard to the probabilities of the categories.

When subjects were given no personality sketch, but were simply asked for the probability that an unknown individual was an engineer the subjects correctly gave the responses 0.7 and 0.3 in 1 and 2 respectively. However, when presented with a totally uninformative description the subjects gave the probability to be 0.5 in both experiments 1 and 2.

Kahneman & Tversky concluded that when no specific evidence is given, prior probabilities are used properly; when worthless evidence is given, prior probabilities are ignored.

### 11.1.3 Insensitivity to sample size

Consider the problem of two hospitals of different sizes in the same town. In the large hospital, 45 babies are born each day, whereas only 15 are born in the smaller hospital. 50% of all babies are boys, but on some days the percentage will be higher and on other days, it will be lower.

Which hospital would you expect to record more days per year, when over 60% of the babies born were boys?

The answer is the smaller one. A large sample is less likely to stray from the 50%.

Most people answer that it would be the same for both hospitals. They assume the events are described by the same statistic and therefore equally representative of the whole population. This fallacy derives from a lack of an intuitive "law of large numbers".

### 11.1.4 Misperception of chance and randomness

This is an error of "local" randomness known as "the gambler's fallacy" or a belief in the "law of small numbers". People believe that when flipping a coin for example, after several "heads" the next flip will surely be "tails". The sequence H-T-H-T-T-H is considered more likely than H-H-H-T-T-T, for example. It seems to be more random, or it is more representative of the expected sequence generated by such a random process.

In other words, the fallacy is that the characteristics of a process will not only be represented globally in an entire sequence, but also locally in each of its parts, and it is a fallacy with which even experienced researchers frequently expose themselves. For example, by expecting 10 samples to provide statistically significant results in an experiment or trial, in the same way as if there were 10,000 samples.

## 11.2 Denial of Uncertainty

Where does uncertainty lie?

When a race starts, does the uncertainty concerning the eventual winner of the race reside in the spectators, or is it a property of the horses and riders?

Does uncertainty come from within yourself, or is it an intrinsic property of events in the environment?

If you opted for the second option i.e. that uncertainty is a property of events in the environment, then you are subject to this fallacy. You are denying uncertainty. You believe you can control it. You may be a clever business manager who believes she can avoid uncertainty and actually reduce risk by skillful action, for example. But the answer is that uncertainty is attributable to YOU.

Acknowledgement of this fallacy is vital to any attempts to quantify subjective probabilities about uncertain events.

### 11.3 Availability

*Availability* is the collective term used to describe the following range of fallacies people make when judging probabilities.

- Retrievability of instances
- Illusory Correlation
- Biases due to the Effectiveness of a search set
- Biases due to Imaginability

We consider these in turn:

#### 11.3.1 Retrievability of instances

If you have just witnessed a car accident, your estimate of the subjective probability of having a car accident will temporarily rise. The event is salient, and so more available. Similarly, in an experiment by Tversky and Kahneman, subjects who were asked whether a list of well-known personalities contained more men than women, responded positively if the men in the list were better known than the women, whereas the numbers of each gender were in fact the same. In this case it was increased familiarity which made the male names more available and hence caused the error of judgement.

#### 11.3.2 Illusory Correlation

Here the co-occurrence of two events is judged on the strength of their association. This is a very important fallacy because subjective probability assessors are often asked to estimate joint or conditional probabilities that depend on the correlations between two events.

Chapman and Chapman first described this bias after performing an experiment in which information about hypothetical mental patients was presented to a selection of naive judges. For each patient there was a clinical diagnosis and a drawing of a person that the patient had drawn. Later, the judges were asked to estimate the frequency of co-occurrence of a diagnosis such as paranoia for example, with a feature of the drawing, such as peculiar eyes. The result was a marked overestimation of the frequency of co-occurrence, an effect that was found to be extremely resistant to contradictory data. It also interfered with the detection of other relationships that were in fact present.

The explanation: If the associative bond between two events is very strong, then it is easy to conclude that the events have more frequently occurred together than in reality, they have.

#### 11.3.3 Biases due to the effectiveness of a search set

Think of any English text including words of three letters or more. Is it more likely that a word picked at random and containing the letter "r" would start with the "r", or that "r" would be the third letter?

The answer is that "r" is more frequently found in the third position of English words, than the first. The fallacy is that the reverse is true.

The reason for this is that it is easier to search for words by their first letter than by their third.

#### 11.3.4 Biases of imaginability

Imagine planning an adventurous expedition. How to evaluate the risks involved? You must imagine contingencies with which the expedition may not be equipped to cope. If many of these are vividly portrayed, the expedition would seem to be very dangerous indeed. However, the actual likelihood of any of these events occurring may be very small.

Alternatively, you may wish to assess the frequency of a class whose instances are not stored in memory but can be generated according to a rule. It is usual to generate several instances and evaluate frequency or probability according to the ease by which these instances can be constructed. Frequency is assessed here by imaginability or availability for construction. For example:

Consider a group of ten people who form committees of  $k$  members,  $2 \leq k \leq 8$ . How many different committees of  $k$  members can be formed?

The correct answer to this problem is given by the binomial coefficient

$$\binom{n}{k}$$

which reaches a maximum of 252 for  $k = 5$ . The number of committees of  $k$  members equals the number of committees of  $(10 - k)$  members, because any committee of  $k$  members defines a unique group of  $(10 - k)$  non-members.

Answering this question without computation involves mentally constructing committees of  $k$  members and evaluating their number by the ease with which they come to mind. Committees of few members such as 2, are easier than committees of many members, such as 8. The simplest scheme for the construction of committees is a partition of the group into disjoint sets. It is easy to construct five disjoint committees of 2 members, but impossible to generate even two disjoint committees of 8 members. So, using imaginability alone, the small committees will seem far more numerous than larger committees, whereas in fact there is a bell-shaped function.

#### 11.4 Adjustment of uncertainty

When asked to make predictions, people often select a salient (but not necessarily relevant) starting point and adjust their guesses from there. This adjustment may be insufficient. It has been found that different starting points yield different estimates that are biased towards the initial starting values. This phenomenon is called anchoring.

Starting points can be given, or they may be the result of some incomplete computation. For example, estimate within 5 seconds the product of :

$$8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$$

Have another go, but this time, estimate within 5 seconds the product of:

$$1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8$$

The correct answer is 40,320.

Presumably your answer was much smaller than this. You would have performed a few steps of computation and then estimated the product from there, underestimating because such adjustments typically are insufficient. Also, because the result of the multiplication of the first few steps in the descending sequence is higher than with the second ascending sequence, your answer to the first estimation was most likely a larger number than the second. In an experiment, the median estimate for the descending sequence was found to be 2,250 whilst for the ascending sequence it was 512.

#### 11.5 Conjunction Fallacy

Examine the following personality sketch:

Bill is 34 years old. He is intelligent, but unimaginative, compulsive and generally lifeless. In school, he was strong in mathematics but weak in social studies and humanities.

Which statement is more probable:



A. Bill is an accountant that plays jazz for a hobby, or

B. Bill plays jazz for a hobby?

From elementary probability theory, the probability of a conjunction  $P(A \& B)$  cannot exceed the probability of either of its constituents,  $P(A)$  or  $P(B)$ . This is the conjunction rule. However, it is often the case that the conjunction is more representative of its class than either of its constituents, or more available in some way, and therefore judgements of its probability are subject to one of the representativeness or availability heuristics.

In the example, A. was erroneously selected by 92% of subjects including those who were informed in matters of statistics.

## 11.6 Hindsight bias

This is the tendency of people with the benefit of hindsight, to falsely believe they would have predicted the outcome of an event. This affects probability elicitation because once outcomes are observed, the assessor may assume that they are the only outcomes that could have happened and underestimate the uncertainty in the outcomes that could have happened, but didn't. By doing this we are preventing ourselves in future episodes, from learning from the past.

Warning people of the dangers of this bias has little effect. In hindsight we are anchored, and cannot truly reconstruct our foresightful state of mind. It is better to argue against the inevitability of the reported outcome and convince oneself that it might have turned out otherwise.

## 11.7 Difficulties in assessing variance, covariance, and correlation

It has been shown that people have great difficulty estimating statistical variance. Estimates are influenced by the mean of the stimuli. Instead of estimating variance, it is the coefficient of the variation (the standard deviation divided by the mean) that is estimated. The explanation given by Peterson and Beach (1967) is as follows:

"Think of the top of a forest. The tree tops seem to form a fairly smooth surface, considering that the trees may be 60 or 70 feet tall. Now, look at your desk top. In all probability it is littered with many objects and if a cloth were thrown over it the surface would seem very bumpy and variable. The forest top is far more variable than the surface of your desk, but not relative to the sizes of the objects being considered."

Experiments concerning bivariate observations include people's ability to recognise functional relationships presented in simple 2 x 2 contingency tables. Typically, these summarise the number of instances of the presence and absence of some variable X, apparently associated with the presence or absence of some variable, Y (eg. X as a disease and Y as a symptom). In many cases, it is found that people's judgemental strategies ignore one or more of the four cells. Most commonly, there is a virtually exclusive reliance on the size of the "present-present" cell relative to the entire population. In other words, if there are more people with the disease that also have the symptom than those with the disease but without the symptom, then the conclusion is that the relationship is positive. But valid inferences in such cases can only be made by considering *all* of the four cells, for example by comparing the proportion of diseased people showing the symptom with the proportion of non-diseased people also showing the symptom.

A different example of this in terms of everyday inference concerns answers to the question: "Does God answer prayers?"

If you consult the "present-present" cell only, you may answer "yes", if when you've asked God for something it has happened. The skeptic may query this asking how often you had asked for something and it did not happen. But this comparison of only two cells is inadequate. Although it seems crazy,

data from the "absent-absent" cell i.e. things did not happen and that were not prayed for must still to be considered, as well as the things that did happen and were not prayed for.

Investigating covariance assessment, experiments have also compared data-based correlation estimates where the data is pairs of numbers or sounds, and theory-based estimates where no data is presented. Subjects used a simple rating scale to describe their subjective impression of the strength and direction (positive or negative) of relationships between pairs of variables.

It was found that for data-based experiments, subjects had difficulty recognising positive relationships with correlations of less than 0.6-0.7. Correlations in the range of 0.2-0.4 were barely detected and correlations of 0.6-0.8 were underestimated. Only correlations of over 0.85 were consistently rated positive. With theory-based estimates on the other hand, positive and negative correlations were correctly recognised and so were the relative magnitude of the correlations. In the theory-based experiments, covariation estimates were based on subjects' prior expectations or theories rather than any immediately available data. eg. alternative measures of honesty, or personal attitudes, habits or preferences. These were compared with "objective" correlations taken from previous empirical studies.

## 11.8 Conservatism

Suppose that we have two bags each containing black and white balls. One bag contains 30 white balls and 10 black balls. The other bag contains 30 black balls and 10 white. Suppose we choose one of these bags at random. For this bag we select five balls at random, replacing each ball after it has been selected. The result is that we find 4 white balls and one black. What is the probability that we were using the bag with mainly white balls?

If you are a typical subject, you would have answered between 0.7 and 0.8. The answer is in fact, 0.96, which can be computed as shown in Section 8.6.1.

Bayes' is a formally optimum rule about how to revise opinions in the light of evidence. Typically subjects revise their probabilities less than Bayes's rule suggests. We are "conservative" processors of information.

It seems that the major cause of conservatism is human misaggregation of the data. We perceive each datum accurately and are aware of its individual diagnostic meaning, but are unable to combine its diagnostic meaning well with the diagnostic meaning of other data when revising opinions.

## 11.9 Overconfidence

Confidence in one's knowledge can be assessed with questions of the following kind:

Which city has more inhabitants?

Hyderabad

Islamabad

How confident are you that your answer is correct?

50%, 60%, 70%, 80%, 90%, 100%

If you answer 50%, then you are guessing. If you answer 100%, then you are absolutely sure of your answer.

Two decades of research into this topic has demonstrated that in all cases where subjects said they were 100% certain of an answer, the relative frequency of correct answers was 80%. Where subjects said they were 90% certain of an answer, the relative frequency of correct answers was about 75%. If subjects said they were 80% confident of an answer, the relative frequency of correct answers was in fact 65%, and so on.

In other words, values for confidence were systematically higher than relative frequencies. This is the overconfidence bias which has been demonstrated using a variety of tasks, including those that are impossible or nearly impossible. Examples include predicting the winners in 6-furlong horse races or diagnosing the malignancy of ulcers.

Overconfidence also occurs when accumulating evidence, for example case-study material about some demonstrable effect or other, from which certain predictions are then made. There is a point in the information-gathering process when predictive accuracy reaches a ceiling. Nevertheless, confidence in one's conclusions continues to rise as more information is received. Towards the end of the information-gathering process, most judges are overconfident about their judgements.

### 11.10 Fallacies associated with causal and diagnostic reasoning

Firstly, let's clarify the main concepts. Consider the following example:

*Jack is an introvert. He is shy and unworldly. He stays at home most nights watching TV.*

The above description of Jack's character as a shy introvert could be considered *causal data* for predicting his behaviour i.e. that he spends his nights at home watching TV.

On the other hand, if spending the nights at home watching TV is seen as a possible cause of Jack's introverted nature, then this behaviour is *diagnostic information* about his introverted personality.

These different relations concern judgements of conditional probability  $P$ , defined as  $P(X/D)$  of some target event  $X$  (Jack's character), on the basis of some evidence or data  $D$  (Jack's behaviour).

Alternatively, if Jack's behaviour  $D$  is neither the cause nor the effect of his personality  $X$ , but both are perceived to be consequences of some other factor such as that Jack lives alone in the middle of nowhere, then  $D$  is referred to as *indicational data* for how Jack may behave in some other also lonely situation.

But if  $D$  and  $X$  do not seem to be related either by a direct or indirect causal link, then  $D$  is referred to as *incidental*.

People strive to achieve a coherent interpretation of the events that surround them. The organisation of events by *schemas* of cause-effect relations serves to achieve this goal.

But whereas with a normative treatment of conditional probability the data  $D$  and an event  $X$  can be equally informative, psychologically, causal data tends to have a far greater impact than other data of equal informativeness. So much so, that in the presence of data that evokes a causal schema, incidental data which does not fit that schema is given little or no consideration.

**Example 1** Here inferences from causes to consequences are made with greater confidence than inferences from consequences to causes.

Which of the following events is more probable?

- a. That a girl has blue eyes if her mother has blue eyes.
- b. That the mother has blue eyes, if her daughter has blue eyes
- c. That the two events are equally probable.

Most people answer correctly (c). In an experiment conducted by Tversky and Kahneman, 75 out of 165 subjects chose this answer. However, nearly as many – 69 -chose (a) and only 21 chose (b). This indicates the expected asymmetry of inference. We generally perceive the daughter as more similar to the mother than vice versa, and we attribute properties of the daughter to the mother with greater confidence than vice versa.

Conclusion: the impact of causal data on the judged probability of a consequence is greater than the impact of diagnostic data on the judged probability of a cause.

Why?

It seems it is easier and more natural for us to reason from causes to consequences than to reason from consequences to causes. (See [availability - biases of imaginability](#)).

**Example 2** Here, when the same data has both causal and diagnostic significance, the former is generally given more weight than the latter in judgements of conditional probability.

Which of the following two probabilities is higher?

(a)  $P(R/H)$  The probability that there will be rationing of fuel for individual consumers in the UK in the next millenium, if you assume that a marked increase in the use of solar energy for home heating will occur during the last few years of this millenium.

(b)  $P(R/\sim H)$  The probability that there will be rationing of fuel for individual consumers in the UK in the next millenium, if you assume that no marked increase in the use of solar energy for home heating will occur during the last few years of this millenium.

Note that the event H has both causal and diagnostic significance:

**Causal:** A marked increase in use of solar energy should alleviate a fuel crisis. The direct causal relationship with R therefore indicates that it is less likely that there will be fuel rationing if we adopt a strategy for solar energy use. This reasoning makes (a) the more probable.

**Diagnostic:** The diagnostic implications of H increase R, because a marked increase in use of solar energy in the next few years implies there must be an impending energy crisis for this to be a worthwhile strategy. This reasoning makes (b) the more probable.

In an experiment of this type, 68 of 83 respondents stated that (a) was the more probable. However, the amount of fuel saved by increased use of solar energy for home heating is unlikely to be sufficient to avert an impending crisis, whereas the shortage of fuel implied by H is indeed indicative of a forthcoming energy crisis. This line of reasoning makes (b) the more probable.

**Example 3** Here it is easier to assimilate a new fact within an existing causal model than to revise the model using diagnostic inference, in the light of this new fact.

Prediction and explanation represent two different types of causal inference. Models or schemas are often used to explain or predict outcomes which in turn are then used to revise or update the models. Model revision is an example of diagnostic inference.

The strength of causal reasoning and the weakness of diagnostic reasoning are evident from:

- People over-predicting from uncertain models. For example predicting academic performance of an individual from a brief personality sketch.
- People constructing causal accounts for outcomes they could not predict.
- People having great difficulty revising uncertain models to accommodate new data.

Consider the following description written by a clinical psychologist on the basis of projective tests:

Tom W. is a student of high intelligence, although lacking in true creativity. He has a need for order and clarity, and for neat and tidy systems in which every detail finds its appropriate place. His writing is rather dull and mechanical, occasionally enlivened by somewhat corny puns and by flashes of imagination of the sci-fi type. He has a strong drive for competence. He seems to have little feel and little sympathy for other people and does not enjoy interacting with others. Self-centred, he nonetheless has a deep moral sense.

What subject(s) would you predict to be Tom W.'s most likely field of graduate study?

If you put Computer Science or Engineering, then you are in agreement with the majority who took part in this experiment.

Now answer the following :

In fact, Tom W. is a graduate student in the School of Education and he is enrolled on a special programme of training for the education of handicapped children. Please outline very briefly the theory which you consider most likely to explain the relation between Tom W.'s personality and his choice of career.

Solution: Well, did you even consider revising your model based on believing the description may be unreliable or inaccurate? In this example, Tom's vocational choice is unlikely given the personality description. But that description was only provided on the basis of projective tests. A reasonable diagnostic inference should therefore lead to a substantive revision of the image of Tom W.'s character to make it more compatible with the stereotype of his profession.

Explanation:

In general, if explanations can be made with minimal and local changes to existing conceptions, people rarely drastically revise their opinions.

## 12 Appendix F: BBNs and Bayesian Probability Resources

This section contains the following resources:

- References on Bayesian probability
- References on eliciting probabilities
- Books about BBNs
- Papers about BBNs
- BBN related websites
- BBN tools
- BBN projects
- BBN teaching/learning material on the web
- BBN people

### 12.1 Probability References

Kahneman D. Slovic P. & Tversky A.(eds). *Judgement under Uncertainty: Heuristics and Biases*. Cambridge University Press, (1982).

Lee PM, 'Bayesian Statistics: An Introduction, 2nd Edn' , Arnold, London UK, 1997.

Quite a good account of Bayesian statistics, but still not for the feint hearted (Lindley's book is better for beginners)

Lindley DV, 'Making Decisions' , John Wiley and Sons, 1985.

Something of a classic book. Argues that Bayesian probability is the natural basis for decision making in all walks of life.

Wilson A.G. Cognitive Factors Affecting Subjective Probability Assessment. *Institute of Statistics and Decision Sciences Discussion Paper No. 94-02*. Duke University, Durham, USA (1994).

Wright G. & Ayton P. (eds). *Subjective Probability*. John Wiley & Sons Ltd., (1994)

### 12.2 References on eliciting probabilities

Cooke, R.M. "Experts in Uncertainty. Opinion and Subjective Probability in Science". Oxford University Press, 1991.

Meyer, M. and Booker, J. "Eliciting and Analyzing Expert Judgement. A Practical Guide." Knowledge Based Systems Volume 5. Academic Press, 1991.

### 12.3 Books about BBNs

Jensen FV, 'An Introduction to Bayesian Networks' , UCL Press, 1996. ISBN 1-85728-332-5 HB.

A reasonable introduction that comes with a demo version of Hugin.

Pearl J, 'Probabilistic reasoning in intelligent systems' , Morgan Kaufmann, Palo Alto, CA, 1988.

This book contains a thorough, but very hard going, account of BBNs and their propagation

## 12.4 Papers about BBNs

There are literally hundreds of papers about BBNs- most are highly technical. For a really extensive set of BBN references visit the website: "A Roadmap to Research on Bayesian Networks and other Decomposable Probabilistic Models" hosted by Lonnie Chrisman, School of Computer Science, Pittsburgh

<http://www.cs.cmu.edu/afs/cs.cmu.edu/user/ldc/WWW/bayes-net-research/bayes-net-research.html>

Also worth looking at are:

Special Issue on Bayesian Networks: Communications of the ACM., March, 1995, vol 38, no. 3.

Special Issue on Data Mining: Communications of the ACM., November, 1996, vol 39, no. 11.

D. Heckerman. A tutorial on learning Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, March, 1995.

## 12.5 BBN related web sites

**Agena** is a company which specialises in decision analysis for critical systems, with special emphasis on BBN solutions. This site contains extensive material on BBNs.

<http://www.agenaco.uk>

**Hugin Expert A/S - Home Page:** Site for the company that produces the leading BBN tool

<http://www.hugin.com/>

**Roadmap to Research on Bayesian Networks and other Decomposable Probabilistic Models .** Really extensive set of BBN references hosted by Lonnie Chrisman, School of Computer Science, Pittsburgh

<http://www.cs.cmu.edu/afs/cs.cmu.edu/user/ldc/WWW/bayes-net-research/bayes-net-research.html>

**Association for Uncertainty in Artificial Intelligence** The main purpose of AUAI is running the annual Conference on Uncertainty in Artificial Intelligence (UAI). Their web site is a very good source of BBN material

<http://www.auai.org/>

**Knowledge Industries** is a company which specialises in building expert systems based on BBNs. This is a very good web site with examples of BBN applications.

<http://kic.com/>

**Microsoft's Decision Theory & Adaptive Systems Group** Extensive BBN activity here

<http://www.research.microsoft.com/research/dtg/>

**BNets -- FAQs & NewsGroups** currently contains a small set of FAQ's (hosted by IEI Pisa).

<http://rep1.iei.pi.cnr.it/projects/SHIP/talk/BNHold/faqs.html>

**Learning Bayesian Networks** This is work done primarily by Wai Lam for his Ph.D. research. It investigates the application of the MDL (Minimum Description Length) principle to the problem of constructing Bayesian Networks from data.

<http://logos.uwaterloo.ca/fbacchus/B-nets.html>

**Marco Ramoni's home page research projects and publications** Extensive BBN material

<http://kmi.open.ac.uk/~marco/>

### **Applicability of Genetic Algorithms for Abductive Reasoning in Bayesian Belief Networks**

[http://www.mi.fgg.eur.nl/FGG/MI/annrep94/p\\_08.html](http://www.mi.fgg.eur.nl/FGG/MI/annrep94/p_08.html)

### **Bayesian Network Interchange Format Home Page**

<http://www.research.microsoft.com/research/dtg/bnformat/default.htm>

Norsys company that specializes in making BBN tools

<http://www.norsys.com/>

### **Tony O'Hagen's tool University of Nottingham**

<http://www.maths.nott.ac.uk/personal/aoh/1b.html>

### **Bayesian Network Links**

<http://www.lis.pitt.edu/~jeroen/bayesian.htm>

**Statistical Decision Theory and Uncertainty Reasoning on the Web:** Good links here

<http://www.cs.ruu.nl/research/projects/tetrad/related/theory.html>

## **12.6 BBN tools**

**Hugin.** The leading BBN tool

<http://www.hugin.com/>

**Bayesian Knowledge Discoverer (BKD)** is a computer program able to learn Bayesian Belief Networks (BBNs) from (possibly incomplete) databases. BKD is based on a new estimation method called Bound and Collapse and it has been developed within the Bayesian Knowledge Discovery project.

<http://kmi.open.ac.uk/~marco/projects/bkd/software/>

Norsys company that specializes in making BBN tools

<http://www.norsys.com/>

## **12.7 BBN projects**

**IMPRESS** (IMproving the software PRocESS using bayesian nets) EPSRC Project GR/L06683. 1 Jan 1997 - 31 Dec 1999.

[http://www.csr.city.ac.uk/csr\\_city/projects/impres.html](http://www.csr.city.ac.uk/csr_city/projects/impres.html)

**SERENE** (SafEty and Risk Evaluation using Bayesian NEts) ESPRIT Framework IV Collaborative Project 22187. 1 June 1996 - 1 June 1999.

<http://www.hugin.dk/serene/>

**TRACS** DERA contract for CSR, LSF/E20173. Sept 1996 - February 1999.

[http://www.csr.city.ac.uk/csr\\_city/projects/dra.html](http://www.csr.city.ac.uk/csr_city/projects/dra.html)

**BBNs for robotics** This is an interesting Japanese project

<http://www.etl.go.jp/etl/robotics/People/hara/bayes.html>

**European Union, Project INTAS 93-725** Multivariate Statistical Analysis and Bayesian Belief Networks for the Development of Intelligent Decision Support Systems with Applications in Medicine and Agriculture



<http://www.gsi.dit.upm.es/~jcg/prj/intas.html>

## **12.8 Relevant Journals for BBNs**

### **Decision Support Systems**

<http://cism.bus.utexas.edu/CISM/DSS/Dss.html>

**Machine Learning Online** Online version of journal with numerous papers on BBNs

<http://mlis.www.wkap.nl/>

## **12.9 Lectures/Teaching material for BBNs**

CSR's BBN and probability web site

<http://www.csr.city.ac.uk/people/norman.fenton/bbns/bbnframes.html>

A nice introductory lecture on Bayesian networks by Stuart M. Speedie University of Minnesota:

<http://yoda.cis.temple.edu:8080/UGAIWWW/lectures/bnets.html>

## 13 Appendix G: SERENE Tool Reference Section

This appendix contains information about the following SERENE tool commands

- [About](#)
- [Add Node](#)
- [Add Link](#)
- [Case File](#)
- [Delete](#)
- [Domain](#)
- [Domain Menu](#)
- [Domain Window](#)
- [Edit Menu](#)
- [Evidence](#)
- [Exit](#)
- [Export Domain As HKB](#)
- [Export Domain As Net](#)
- [Export Template As Net](#)
- [Expression Building](#)
- [Expression Editor](#)
- [File Menu](#)
- [Generate Domain Documentation](#)
- [Generate Template Documentation](#)
- [Help](#)
- [Help Menu](#)
- [Import Template From Net](#)
- [Joint Probability](#)
- [Make Domain](#)
- [Monitor Window](#)
- [Links](#)
- [Load Case File](#)
- [New Project](#)
- [New Template](#)
- [Node Categories](#)
- [Node Kinds](#)
- [Node Properties](#)
- [Nodes](#)
- [Node Table](#)
- [Open Project](#)
- [Open Template](#)
- [Propagate](#)
- [Remove Template](#)
- [Retract All Evidence](#)
- [Save Case File](#)
- [Save Project](#)
- [Save Project As](#)
- [Save Template](#)
- [Save Template As](#)
- [Sensitivity Analysis](#)
- [Template](#)
- [Template Documentation](#)
- [Template Menu](#)
- [Templates Window](#)
- [Template Window](#)
- [Tools Menu](#)
- [Window Menu](#)

### 13.1 About

The About box is opened by selecting "About" in the [Help Menu](#).

### 13.2 Add Link

You can add a new [link](#) from one node to another to a template by selecting "Add Link" from the [Edit menu](#). Then, you can add the link by dragging from one node to another.

Several links can be added quickly if you hold down the shift key.

In the tool bar you can also find a tool button for adding links:



### 13.3 Add Node

You can add a new node to a template by selecting "Add Node" from the [Edit menu](#). When you do this, you are prompted for the [node category](#) and [node kind](#). Then, you can add the node by clicking in the network of the [Template Window](#).

In the tool bar you can also find a series of tool buttons for adding nodes.

Several nodes can be added quickly if you hold down the shift key.

You can also use the different node buttons in the tool bar to add nodes:



There is one node button for each of the [categories](#): Chance node, decision node, utility node, and abstract node.

### 13.4 Case File

A case file is a file storing the [evidence](#) of a [domain](#). Using case files you are able to store interesting *cases* and quickly [load](#) them again when it is needed (eg. for a demonstration).

Case files are stored with the .snc file extension.

### 13.5 Delete

You can delete a node or link by selecting it in the network of the [Template Window](#) and then selecting "Delete" from the [Edit menu](#).

You can select several nodes and/or links if you hold down the shift key.

You can also use the delete button from the tool bar:



### 13.6 Domain

A domain in the SERENE tool represents a BBN or an influence diagram specified by a set of [templates](#). In a domain you are able to calculate the probability distribution of any chance node or the expected utility of any decision node given a set of evidence.

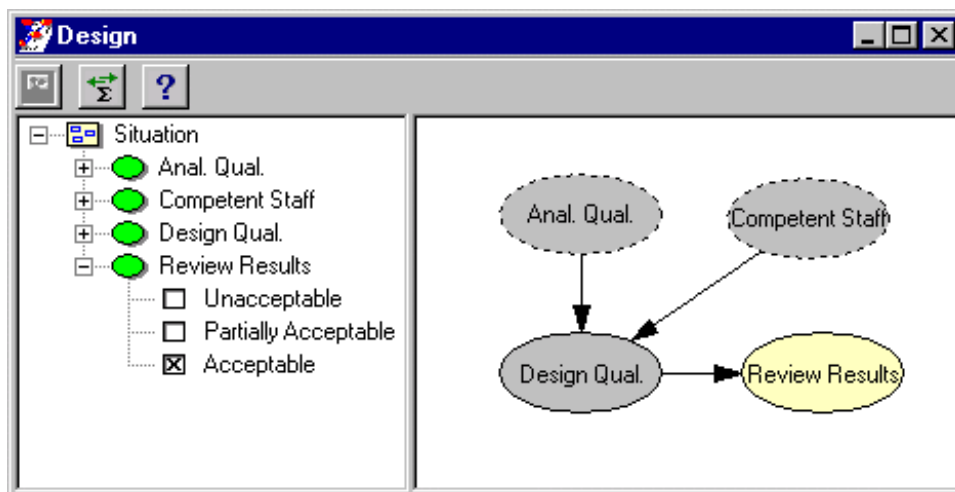
A domain is monitored in a [Domain Window](#).

### 13.7 Domain Menu

The Domain menu contains actions to manipulate the current [Domain](#):

- [Propagate](#)
- [Retract All Evidence](#)

### 13.8 Domain Window



The Domain Window monitors a [domain](#). The left pane of the window is a tree view of the domain structure. Here, you can browse down to the sub domains specified by separate [templates](#). The right pane is the network pane where the template structure of the currently selected sub domain is monitored.

## 13.9 Edit Menu

The Edit menu contains actions to manipulate the current [template](#):

- [Add Node](#)
- [Add Link](#)
- [Delete](#)
- [Node Properties](#)
- [Node Table](#)

## 13.10 Evidence

Evidence is information about the state of one or more nodes in a [domain](#). Eg. if one of the nodes of a domain for determining diseases at a clinique is **Fever** with states **false** and **true**, observing that the patient has a temperature of 39.5&deg; C (= 103.1&deg; F) will make you able to enter *evidence* into the **Fever** node.

Evidence can be entered in SERENE in two ways:

4. Browse down into the node using the tree view of the [Domain Window](#). If the node belongs to the current sub domain, you can select one or more states of the node.
5. Double-click a node, and a [Monitor Window](#) will appear. In the monitor window, you can select the state of the node.

Evidence in continuous chance nodes can only be entered using monitor windows.

## 13.11 Exit

You can exit the SERENE tool by selecting "Exit" from the [File Menu](#).

When you exit the SERENE tool, you are prompted to save the state of the current project:

- If you have a running [domain](#), and you have entered [evidence](#), you are asked if you want to save a [case file](#).
- If any of the templates have changed since the last save or since they were opened, you are asked if you want to save them.
- If the templates have changed or new templates have been added, you are asked if you want to save the current project. If you have answered "no!" to saving some of the templates, the project cannot be saved.

### 13.12 Export Domain As HKB

You can export a [domain](#) to the Hugin HKB format by selecting "Export Domain As HKB" from the [File menu](#) of a [Domain Window](#).

### 13.13 Export Domain As Net

You can export a [domain](#) to the Hugin Net format by selecting "Export Domain As Net" from the [File menu](#) of a [Domain Window](#).

### 13.14 Export Template As Net

You can export a [template](#) to the Hugin Net format by selecting "Export Template As Net" from the [File menu](#) of a [Template Window](#).

### 13.15 Expression Building

To specify expressions for a conditional probability table or utility table of a node, you need to enter [Node Properties](#) and under the "Probabilities" tab select "Function Expressions". Then, click "OK" and open the [Node Table](#) of the node.

Now, you can either type in the expression yourself in each cell or you can use the [Expression Editor](#).

The functions specified by expressions differ in that some are deterministic functions specifying the exact state of a node, eg:  $n\text{People} = n\text{Men} + n\text{Women}$  whereas others are distribution functions specifying the probability distribution of the states, eg:  $P(n\text{Six}) = \text{Binomial}(n\text{Dice}, 1/6)$ . There are a limited number of distribution functions available so SERENE will know how to interpret the expression you write.

The building blocks available for expressions are

- Constant values
- Discrete distribution values
- Continuous distribution functions
- Arithmetic functions
- Boolean functions
- Comparison operators
- If-then-else function

#### 13.15.1 Constant Values

The following kinds of constants can be used in expressions: state labels, numeric values, and Boolean values.

State labels must be encapsulated in quotation characters (").

Numeric values are typed in as you would probably expect to do - without any encapsulating characters or other formatting stuff: Eg. "X + 3.4" (here, the quotation characters are used only to separate the expression from the rest of the text - it has nothing to do with state labels as discussed above).

There are two Boolean constant values: "false" and "true". Notice that the Boolean constants must be lower case.

### 13.15.2 Discrete Distribution Functions

These distribution functions are available for numbered nodes only. You type a distribution function as the function name followed by a comma separated list of arguments in brackets. Eg. "Binomial(4, 0.2)".

Function	Node requirements	Comments	Arguments	Arg. range
Binomial	Numbered: 0, 1, 2, ... , n		n	0, 1, 2, ...
			p	[0,1]
Poisson	Numbered: 0, 1, 2, ... , n	n will get prob. mass of n, n+1, ...	Mean	]0, inf[
Geometric	Numbered: 0, 1, 2, ... , n	n will get prob. mass of n, n+1, ...	p	[0, 1]

### 13.15.3 Continuous Distribution Functions

These distribution functions are available for interval nodes.

Function	Node requirements	Comments	Arguments	Arg. range
Normal	Interval. First state must start in $-\infty$ . Last interval must end in $\infty$ .		Mean	$]-\infty, \infty[$
			Variance	$]0, \infty[$
Beta	Interval. First state must start below Lower. Last interval must end after Upper (see arguments).	If not specified, arguments Lower and Upper will be 0 and 1, respectively	Alpha	$]0, \infty[$
			Beta	$]0, \infty[$
			Lower (optional)	$]-\infty, \text{Upper}[$
			Upper (optional)	$]\text{Lower}, \infty[$
Gamma	Interval: First state must start below 0. Last interval must end in $\infty$ .		Shape	$]0, \infty[$
			Scale	$]0, \infty[$
Exponential	Interval: First state must start below 0. Last interval must end in $\infty$ .		Lambda	$]0, \infty[$
Weibull	Interval: First state must start below 0. Last interval must end in $\infty$ .		Shape	$]0, \infty[$
			Scale	$]0, \infty[$
Uniform	Interval: The state intervals must cover Lower and Upper - at least in end points (see arguments).		Lower	$]-\infty, \text{Upper}[$
			Upper	$]\text{Lower}, \infty[$



#### 13.15.4 Arithmetic Functions

The arithmetic functions can be used in deterministic expressions for numbered nodes, interval nodes, and utility nodes. They can also be used in all kinds of functions taking numeric arguments. Below are listed the arithmetic functions:

+ (sum)

- (subtract)

\* (multiply)

/ (divide)

power

min

max

log

negate

log

exp

sqrt

The first four arithmetic functions (operators) use infix notation: "A + B". The rest have prefix notation: "max(A, B, 0.7)".

#### 13.15.5 Boolean Functions

The Boolean functions can be used in deterministic expressions for Boolean nodes. They can also be used in all kinds of functions taking Boolean arguments. The Boolean functions are:

and

or

not

Boolean functions have prefix notation: "and(or(X, Y), A, B)".

### 13.15.6 Comparison Operators

The comparison operators can be used in deterministic expressions for Boolean nodes. They can also be used in all kinds of functions taking Boolean arguments. Below are listed the comparison operators:

= (equals)

!= (equals not)

< (less than)

> (greater than)

<= (less than or equals)

>= (greater than or equals)

All comparison operators can be used to compare numeric values. The equality comparison operators (= and !=) can also be used to compare Boolean values and labels. All comparison operators use infix notation: "A = 14".

### 13.15.7 The if-then-else Function

The if-then-else function takes three arguments: The first argument is the Boolean condition determining if the function should return the second or third argument as the result. Eg. "if(false, "yes", "no")" would return "no".

The if-then-else function can be used to combine both deterministic functions and distribution functions. Eg. if you are building a system for predicting how many times you will roll six with a number of dice, you could be in a situation where you were uncertain about whether the dice were fair or not, having probability 1/6 and 1/2 for rolling six respectively. If you have a Boolean node "DiceFake" representing this uncertainty, you could specify one of these two expressions:

"Binomial(nDice, if(DiceFake, 1/2, 1/6))"

or

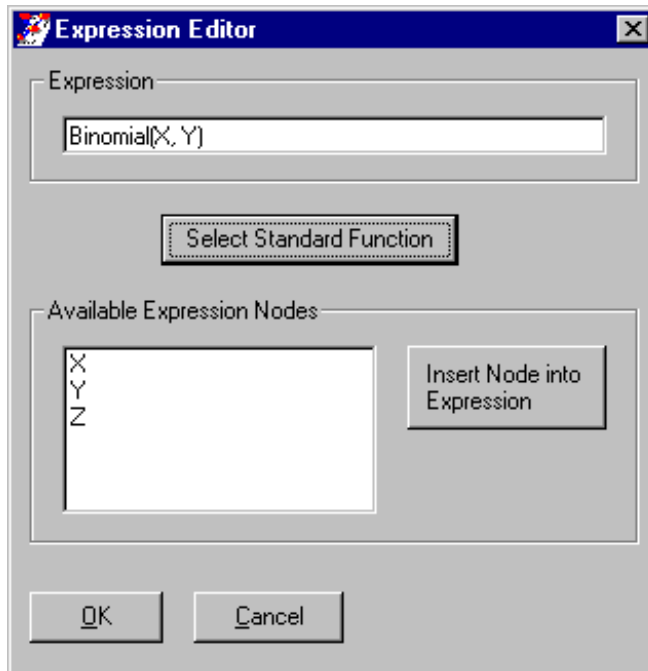
"if(DiceFake, Binomial(nDice, 1/2), Binomial(nDice, 1/6))"

### 13.16 Expression Editor

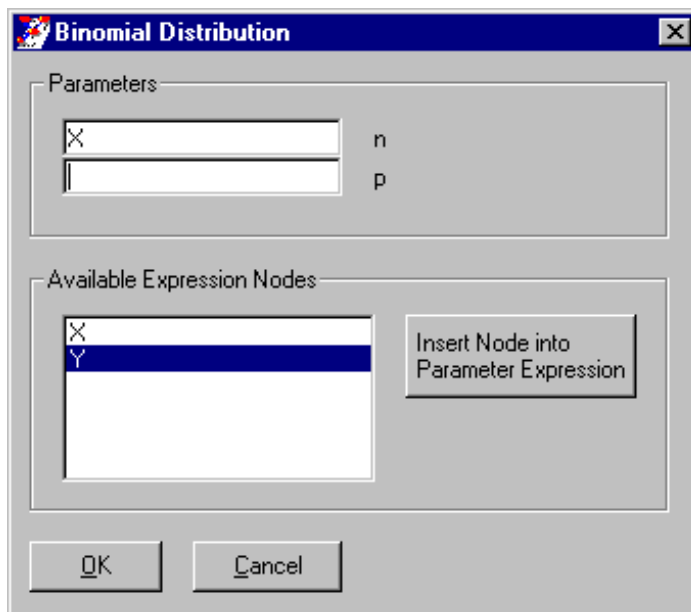
The Expression Editor is opened by double-clicking the cells of the [node table](#) when "Function Expressions" have been specified in [Node Properties](#).

At the top you can edit the expression yourself. This is necessary if you want to specify deterministic expressions using arithmetic, Boolean, or comparison operators. If the node is a numbered or interval chance node, you can also select standard distribution functions from a list.

First you select the distribution function.



Then, you specify the arguments.



At the bottom of the Expression Editor you can see the list of nodes that can be used in the expression (parent nodes of the current node). You can insert them at the position of the cursor by clicking the "Insert Node into Expression" button.

See also Expression Building

### 13.17 File Menu

The file menu contains entries for loading and storing data externally to the SERENE tool. The File menu is different depending on whether the current window is a [Template Window](#) or a [Domain Window](#).

#### Template Window File Menu

- [New Template](#)
- [Open Template](#)
- [Remove Template](#)
- [Import Template From Net](#)
- [Export Template As Net](#)
- [Save Template](#)
- [Save Template As](#)
- [New Project](#)
- [Open Project](#)
- [Save Project](#)
- [Save Project As](#)
- [Generate Template Documentation](#)
- [Exit](#)

#### Domain Window File Menu

- [Load Case File](#)
- [Save Case File](#)
- [Export Domain As Net](#)
- [Export Domain As HKB](#)
- [Generate Domain Documentation](#)
- [Exit](#)

### 13.18 Generate Domain Documentation

You can generate documentation for the current [domain](#) by selecting "Generate Domain Documentation" from the [File menu](#).

The domain documentation is an HTML document containing a list of tables showing the probability distribution of the nodes having an open [Monitor Window](#) when starting the documentation generator.

### 13.19 Generate Template Documentation

You can generate documentation for the current [template](#) by selecting "Generate Template Documentation" from the [File menu](#).

The template documentation can be specified to contain:

- The template graph
- The template description
- Node Documentation:
  - Node tables
  - Node descriptions
  - Node states
  - State descriptions

It can also be specified that each template used by the current template should be documented. This will ensure that there are links from an abstract node in the current template to the documentation of the template it was created from.

### 13.20 Help

You can open the SERENE tool help by selecting "Help" from the [Help menu](#).

### 13.21 Help Menu

The Help menu contains entries to gain further information about the SERENE tool.

- [Help](#)
- [Template Documentation](#)
- [About](#)

### 13.22 Import Template From Net

You can import a [template](#) from the Hugin Net format by selecting "Import Template From Net" from the [File menu](#) of a [Template Window](#).

### 13.23 Joint Probability

In a domain you can get the joint probabilities of a set of discrete nodes using the Joint Probability tool found in the [Tools menu](#).

When the Joint Probability tool is started, it first prompts for a list of nodes using the Node Selection dialog. Move nodes into the Selected Nodes list box and press "OK". This will make a table window appear showing the joint probability of the nodes selected.

### 13.24 Links

You can drag links from one [node](#) to another. We say that the link goes from the *parent* to the *child*. There are different meanings of the links you create. Specifically the links can be:

- Causal links
- Information links
- Join links
- Utility links

### 13.24.1 Causal Links

(links to [chance nodes](#))

Causal links goes from chance nodes or decision nodes to chance nodes. This means that the state of the child depends on the state of the state of the parent. The conditional probability table of the child will change when you add or remove parents.

### 13.24.2 Information Links

(links to [decision nodes](#))

Links into decision nodes have a special meaning: They say that before you can read a valid expected utility in the decision node, the state of the parent node must be known (entered as [evidence](#)).

The reason why you need to add information links is that in an influence diagram (a Bbn with decisions and utilities) there must be an explicitly defined order of the decisions.

Also, it must be explicit when a chance node will be known (between which two decisions will you know the state of the chance node) if it will be known at all. The underlying Hugin inference engine cannot calculate the right expected utilities and probabilities if it doesn't know this order.

### 13.24.3 Join Links

(links to input nodes of [abstract nodes](#))

You connect external nodes with input nodes of an abstract node using *join links*. This means that the external parent will be the one used internally in the abstract node when the input node is used as a parent of another node.

Since the parent and child in a join link are somehow the same node, they must have similar types and have the same number of states. Otherwise, you will not be able to create a domain from the template containing the join link.

### 13.24.4 Utility Links

(links to [utility nodes](#))

Links from chance nodes or decision nodes to a utility node define that the utility function is determined by the state of those parent nodes.

The utility table of the utility node child will change when you add or remove parents.

### 13.25 Load Case File

You can load a [case file](#) by selecting "Load Case File" from the [File menu](#) of a [Domain Window](#).

### 13.26 Make Domain

Any [template](#) can be *instantiated* to a [domain](#) by selecting "Make Domain" from the [Template menu](#).

Creating a domain for a template will disable the user from editing the templates of the domain as long as the domain is still running. To edit the templates, you must first close the [Domain window](#).

You can also use the Make Domain button from the tool bar to perform this action:

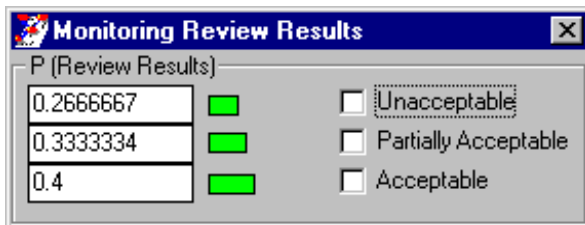


### 13.27 Monitor Window

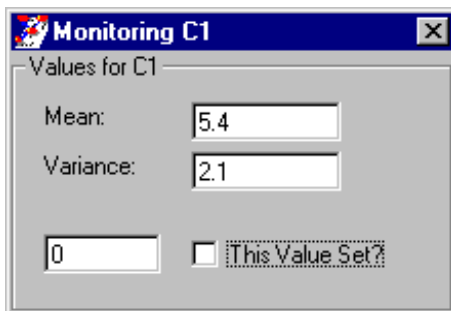
Monitor windows can be opened for chance nodes and decision nodes to monitor the probability distribution or expected utility given any evidence entered.

Monitor windows can also be used to enter [evidence](#) (clicking the states of a discrete node or entering the value of a continuous node).

Below is shown the monitor window of a discrete node.



Below is shown the monitor window of a continuous node.





### 13.28 New Project

You remove the current project (all templates) and create a new one by selecting "New Project" from the [File menu](#) of a [Template window](#).

The SERENE tool will prompt you to save the old project before leaving it.

### 13.29 New Template

You add a new [template](#) to the current project by selecting "New Template" from the [File menu](#) of a [Template window](#).

New templates will be given the name "UntitledX" where X is some number.

## 13.30 Node Categories

When [adding a node](#) you must first specify the *category* of the new node. This can be one of:

- Chance Node
- Decision Node
- Utility Node
- Abstract Node

### 13.30.1 Chance Nodes

Chance nodes are random variable nodes representing some attribute of the world. You'll need to specify how a chance node depends on its parents - this is done in the [node table](#) which is either a conditional probability table (discrete nodes) or a conditional Gaussian distribution (continuous nodes).

After specifying that the category of a node is *chance node*, you must specify the [kind](#) of the node.

### 13.30.2 Decision Nodes

Decision nodes are variables representing decisions. In SERENE decision nodes can only have a finite discrete set of states each having a state label.

When decision nodes are contained in a [domain](#), there must be an explicit order of decisions and chance nodes. This is specified by [information links](#).

### 13.30.3 Utility Nodes

Each utility node in a [domain](#) specify a term of the joint utility function of the domain. The [utility table](#) contain a utility value for each configuration of the parents of the utility node.

### 13.30.4 Abstract Nodes

Abstract nodes are nodes with a more complex structure than just a set of states. They are abstractions over a set of other nodes specified by a [template](#).

After specifying that the category of a node is *abstract node*, you must specify which template is used to create this abstract node.

Abstract nodes have a set of *input nodes* and a set of *output nodes*. These are specified by the template of the abstract node.

*Output nodes* are nodes from the inside of the abstract node visible to the outside. You can use output nodes [as parents](#) of external nodes. You can NOT make links from external nodes to output nodes.

*Input nodes* are nodes from the outside of the abstract node that are used inside the internal structure of the abstract node as parents. You can make [join links](#) to an input node of an abstract node. The join link means that the parent node of the join link will be the one used inside the abstract node in the place of the input node.

### 13.31 Node Kinds

When [adding a node](#) you must first specify the [category](#) of the new node. Then, if the category was *chance node*, you must specify the kind of the node:

- Discrete Labelled
- Boolean
- Discrete Numbered
- Interval
- Continuous

(Continuous is only available for chance nodes)

#### 13.31.1 Discrete Labelled

Discrete labelled nodes have a set of states each having a label (eg. "blue" or "green"). If such a node is used in a [function expression](#) it can only be used with comparison operators

#### 13.31.2 Boolean

Boolean nodes have two states: **false** and **true**. In function expressions they can be used with both comparison operators and Boolean operators.

#### 13.31.3 Discrete Numbered

Discrete numbered nodes have a set of states each having a numeric value (eg. -1, 0.2, 5, or 194.8). The values of the states must be ordered with the lowest value first. Numbered nodes can be used with comparison operators and arithmetic operators. For discrete numbered chance nodes different standard discrete distribution functions can be used to specify the [node table](#).

#### 13.31.4 Interval

Interval nodes are discrete nodes where each state represent an interval on the real axis. The interval of state  $n+1$  must start where the interval of state  $n$  ended. It is possible for the first state to start in  $-\infty$  and for the last state to end in  $\infty$ .

Interval nodes can be used with arithmetic operators and standard continuous distribution functions.

#### 13.31.5 Continuous

Continuous nodes in the SERENE tool are random variables with a infinite set of states from  $-\infty$  to  $\infty$ . The dependency from the parent nodes to the continuous child node is specified by a [conditional Gaussian function](#).

There are a few restrictions related to continuous nodes:

- Continuous chance nodes cannot be parents of discrete nodes
- Continuous chance nodes cannot be used in the same domain as decision nodes and utility nodes.

## 13.32 Node Properties

You can enter the Node Properties dialog by selecting the node and then selecting "Node Properties" from the [Edit menu](#).

The Node Properties dialog has three tabs:

- General
- States
- Probabilities

You can also use the Node Properties button from the tool bar to perform this action:



### 13.32.1 General tab

In the states tab you can edit the name and label of the node. The name is a unique identifier of the node in the template. It can contain no spaces or funny characters. There are no restrictions in what the label can be. There can be several nodes in the same template with the same label.

In the General tab you can also specify if the node should be an input or an output node of the [template](#).

### 13.32.2 States tab

In the States tab you can edit the states of discrete nodes.

### 13.32.3 Probabilities tab

In the Probabilities tab you can specify how you want to edit the [node table](#) of the node. You can choose to do it manually - specify each value in the table, or by using [function expressions](#) - specifying the table values as some function of the parent states.

If you choose to use function expressions, you can also specify which nodes should be used in the *expression model* (in the node table you will need to specify an expression for each configuration of the model nodes - if there are no model nodes, you only specify one expression).

### 13.33 Node Table

You can edit the node table of a chance node or utility node by opening the Node Table window. You do this by selecting "Node Table" from the [Edit menu](#).

In the node table you can edit the table values manually or you can specify one or more [function expressions](#) for the table. You need to select this in the Probabilities tab of the [Node Properties](#) dialog.

If you specify expressions, you can edit an expression cell by double-clicking it. This will open the [Expression Editor](#).

You can also use the Node Table button from the tool bar to open the node table:



### 13.34 Nodes

The nodes in a [template](#) are characterized by their [category](#) and for chance nodes, their [kind](#).

### 13.35 Open Project

You can open a project by selecting "Open Project" from the [File menu](#). Project files are stored with the .snp extension. Opening a project will close the old one. You will be prompted to save the old project.

### 13.36 Open Template

You open a [template](#) to the current project by selecting "Open Template" from the [File menu](#) of a [Template window](#).

It is illegal to open a template which has the same name as an existing template.

### 13.37 Propagate

After you have entered [evidence](#) into a domain, you need to propagate it to be able to read new updated probabilities and expected utilities. You propagate by selecting "Propagate" from the [Domain menu](#).

### 13.38 Remove Template

You can remove a [template](#) from the current [domain](#) by making its [window](#) active and selecting "Remove Template" from the [File menu](#).

Note that you can close a template window without removing it from the current project. You can open the template window again by clicking the template in the [Templates window](#).

### 13.39 Retract All Evidence

You can retract all evidence entered in a [domain](#) by selecting "Retract All Evidence" from the [Domain menu](#).

### 13.40 Save Case File

You can save a [case file](#) by selecting "Save Case File" from the [File menu](#) of a [Domain Window](#).

### 13.41 Save Project

You can save the current project by selecting "Save Project" from the [File menu](#) when a [template window](#) is active.

Project files get the .snp file extension.

### 13.42 Save Project As

You can save the current project under a new name by selecting "Save Project" from the [File menu](#) when a [template window](#) is active.

Project files get the .snp file extension.

### 13.43 Save Template

You can save the current [template](#) by selecting "Save Template" from the [File menu](#) when a [template window](#) is active.

Template files get the .snt file extension.

### 13.44 Save Template As

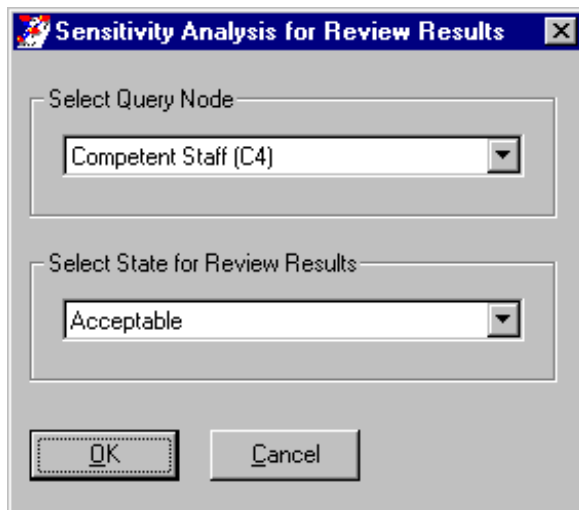
Template files get the .snt file extension.

### 13.45 Sensitivity Analysis

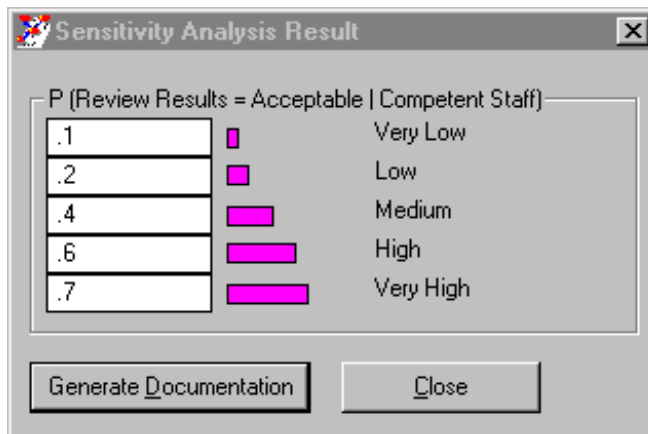
You can perform a sensitivity analysis by selecting "Sensitivity Analysis" from the [Domain menu](#) when a [domain window](#) is active.

Sensitivity can give you an idea of how sensitive a node is to evidence about another node. Eg. you might want to know how sensitive **Review Results** in state **Acceptable** is to **Competent Staff**:

Select the **Review Results** node in the [domain window](#) and activate Sensitivity Analysis.



Then, select **Competent Staff** as *query node* and **Acceptable** as *state of Review Results* and press "OK".



What you get is the conditional probability of **Review Results** in state **Acceptable** given the different states of **Competent Staff**.

### 13.46 Template

A template is a BBN fragment which can be *instantiated* one or more times in other templates as [abstract nodes](#). People familiar with object-oriented programming can think of templates as *classes* and abstract nodes as *objects*.

So, a template contains a set of [nodes](#) with [links](#) between them. To be able to compose a large Bbn from a set of templates, there must be an interface between the templates. This is achieved by defining some of the nodes as *input nodes* and *output nodes*.

*Output nodes* are nodes belonging to the inside of an [abstract node](#) created from this template that are visible to the outside. In the external scope, you will then be able to use output nodes [as parents](#) of external nodes. You can NOT make links from external nodes to output nodes.

*Input nodes* are nodes belonging to the external scope of an abstract node created from this template. They can be used inside the template as parents of internal nodes (including output nodes). When you make a [join link](#) to an input node of an abstract node this will make the external node the real parent of the nodes inside the template having the input node as parent (an input node is a reference to an external node).

### 13.47 Template Documentation

If template documentation is stored, you can view it by selecting "Template Documentation" from the [Help menu](#) when a [template window](#) is active.

If a template is stored in the file "x.snt", then the documentation file must be stored in the same directory ("folder") with the same name as the template file but with the ".html" extension in stead of ".snt".

Template documentation can be [generated](#) by the SERENE tool.

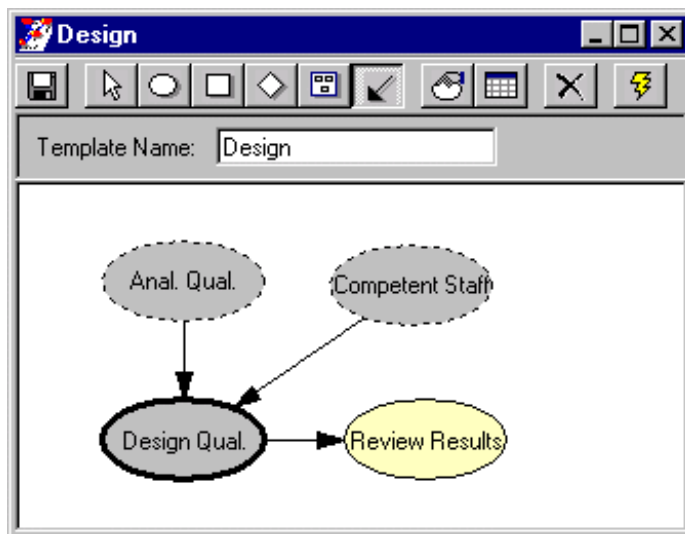


### 13.48 Template Menu

The Template menu contains one entry which allows you to create a [domain](#) from the current [template](#).

- [Make Domain](#)

### 13.49 Template Window



A template window monitors a [template](#). Using the template window you can build and edit a template: [add nodes](#), create [links](#), and change the properties of the nodes (using [and node table](#) dialogs).

### 13.50 Templates Window



You can view all [templates](#) of a project in the Templates window. The Templates window is by default placed in the upper left of the SERENE main window. If it has been closed you can open it by selecting "Templates" from the [Window menu](#).

By double-clicking a template in the templates window you will activate that template (open its [template window](#)).

### 13.51 Tools Menu

The tools menu contains actions to perform special calculations in a [domain](#).

- [Sensitivity Analysis](#)
- [Joint Probability](#)

### 13.52 Window Menu

The Window menu contain one action to open the [Templates window](#).

- [Templates](#)

*This page intentionally left blank*

## 14 Index

- about box, 172
- abstract node, 13, 23, 48, 99, 187
- actions (in MCDA), 149
- add link, 172
- add node, 172
- add sequence button, 86
- adjustment of uncertainty, 161
- Agena, 168
- AHP. *See* Analytical Hierarchy Process
- ALARP, 116
- analogical determination, 143
- Analytical Hierarchy Process, 13, 149
- architectural determination, 142
- arithmetic functions, 178
- attribute, 13, 34
- availability, 160, 161
- axioms, 122
  
- base-rate neglect, 157
- Bayes Rule. *See* Bayes theorem
- Bayes theorem, 22, 124, 131
  - example, 125
  - extended version, 125
- Bayesian approach to probability, 121
- Bayesian Belief Net, 13, 16
  - BBNs and decision making, 145
  - BBNs within the SERENE method, 22
  - books, 167
  - dealing with increasing number of variables, 133
  - defining the topology, 23
  - definition, 129
  - entering evidence and propagation, 130
  - how BBNs deal with evidence, 133
  - introduction to BBNs, 20
  - journals, 170
  - Microsoft use of, 22
  - object oriented approach to BBNs, 23
  - papers, 168
  - projects, 169
  - resources, 167
  - SERENE partner BBNs, 29
  - teaching material, 170
  - to represent a safety argument, 20
  - tools, 169
  - Tutorial on BBNs, 129
  - types of reasoning permitted in BBNs, 137
  - web sites and links, 168
  - why BBNs are needed for probability computations, 132
  - why we should use BBNs, 133
- Bayesian Knowledge Discoverer (BKD), 169
- BBN. *See* Bayesian Belief Net
- belief measure, 121
- Beta distribution, 177
- beta function, 61, 65
- biases due to the effectiveness of a search set, 160
- biases of imaginability, 161
- binomial coefficient, 161
- binomial distribution, 24
- Binomial distribution, 176
- blocked evidence, 134
- Boolean functions, 178
- Boolean node, 188
  
- CASCADE project, 19
- case file, 172, 185
- causal and diagnostic reasoning fallacies, 164
- causal dependence, 151
- causal determination, 137
  - types of, 139
- causal links, 184
- causal nets, 13
- causal relations
  - using time and physical relationships to structure them, 138
- causal structure
  - multiple views, 139
- cause-effect relations, 164
- CENELEC, 114
- chain rule, 126, 132
- chance node, 187
- comparison operators, 179
- conditional independence, 127, 135
- conditional probability, 124
- conditionally independent events, 124
- conjunction fallacy, 162
- conservatism, 163
- constant values, 176
- constraints, 149
- constraints (in decision problems), 149
- continuous distribution functions, 177
- continuous node, 188
- continuous variable, 129
- converging connection, 136
- correlation, 162
- covariance, 162
- criteria, 149
  
- d-connectedness, 122, 123, 126, 132
- decision node, 187
- decision problem, 145
  - constituent parts, 148
  - examples, 145
  - objective, 147
  - perspective, 147
- defect density, 59
- defect density argument, 59
  - creating from templates, 60

- defining criteria, 150
- definitional relations, 142
- definitional/synthesis idiom, 35
- delete, 173
- denial of uncertainty, 159
- density template, 61
- density\_template, 60
- dependability, 150
- deterministic function, 87
- deterministic relations, 141
- difficulties in assessing variance, covariance, correlation, 162
- discrete distribution functions, 176
- discrete numbered node, 188
- discrete variable, 123, 129
- distribution functions, 176, 177
- diverging connection, 134
- divorcing, 37
- domain, 173, 185
- domain documentation, 182
- domain menu, 173
- domain window file menu, 181
- d-separated, 134, 135
- d-separation, 136
  
- EDF safety argument, 30
- edit menu, 174
- eliciting probabilities
  - references on, 167
- entering observations in BBN, 54
- entity, 34
- Equipment under control (EUC), 13
- ERA safety argument, 29
- evidence, 174
- evidence case file, 93
- examples of SERENE method, 59
- execution scenario
  - defect density examples, 63
  - reliability examples, 68
  - safety risk examples, 76
- explaining away evidence, 131
- Exponential distribution, 177
- export domain as HKB, 175
- export domain as net, 175
- export template as net, 175
- expression building, 175
- expression editor, 25, 88, 97, 175, 180
  - window, 89
- external attributes, 34
- external factors, 152
  
- failure analysis techniques, 118
- Failure Modes, Effects and Criticality Analysis (FMECA), 118
- Fault Tree Analysis (FTA), 118
- file menu, 181
  
- frequentist approach to probability, 121
- functional safety, 13, 15
  
- GAM, 19
- GAMAB, 116
- Gamma distribution, 177
- general tab, 189
- generate documentation, 182
- Geometric, 176
- glossary, 13
- Goal-Question-Metric, 13, 145, 154
- GQM. *See* Goal-Question-Metric
- graphical networks, 13
- graphical probability tables, 13
  
- hard evidence (instantiation), 133
- hazard analysis, 29
- help, 182
- help menu, 182
- hindsight bias, 162
- HKB format, 175
- Hugin, 22, 168, 169
- Hugin net format, 175, 183
  
- idiom, 13, 23, 137
  - choosing the right one, 44
  - instances in reliability template**, 64
  - instances instances in defect density adrgument, 59
  - instantiation, 13, 35
  - joining, 45
  - rationale for using, 33
  - relating types of reasoning to, 144
  - specifications of, 34
- IEC. *See* International Electrotechnical Commission
- IEC 61508, 14, 15, 19, 111, 119
- if-then-else function, 179
- illusory correlation, 160
- import template from net, 183
- IMPRESS, 169
- independence, 127
- independence of events, 124
- induction idiom, 41
- influence diagrams, 13
- information links, 184
- input node, 90
- insensitivity to prior probability of outcomes, 157, 158
- insensitivity to sample size, 159
- INTAS, 170
- internal attributes, 34
- internal factors, 154
- International Electrotechnical Commission, 13
- interval node, 188

- join links, 184
- join operation, 45, 53
  - formal definition, 47
- joint events, 123
- joint probability distribution
  - general case in BBN, 133
- joint probability distribution, 123
- joint probability tool, 183
  
- likelihood, 125, 126
- Likelihood Ratio, 126
- link, 172
- links, 183
- load case file, 185
  
- make domain, 185
- marginal probability, 130
- marginalisation, 123
- Markov Nets, 118
- MCDA. *See* Multi Criteria Decision Aid
- measurement idiom, 40
- measurement scale, 150
- measurement theory, 151
- metrics, 119
- Microsoft, 22, 168
- misperception of chance and randomness, 159
- MISRA Guidelines, 116
- monitor window, 185
- Multi-Criteria Decision Aid, 13, 149
- Multi-Criteria Decision Aid, 145
- mutually exclusive events, 122
  
- necessary and sufficient conditions, 138
- new project, 186
- new template, 186
- node, 190
- node category, 187
- node kind, 188
- node probability table, 13, 21
  - creating in SERENE tool, 87
  - defining, 53
  - help with defining, 24
  - icon, 53
  - NPTs for defect density argument, 61
  - NPTs in safety risk template, 74
- node properties, 189
- node properties window, 82
- node table, 190
- node table window, 88
- Normal distribution, 177
- Norsys, 169
- NP-hard problem, 133
- NPT. *See* node probability table
  
- one\_phase\_rework\_template, 60
- open project, 190
  
- open template, 190
- output node, 90
- outranking methods, 149
- overconfidence, 163, 164
  
- parent nodes, 133
- PES. *See* Programmable Electronic System
- Poisson distribution, 176
- posterior belief, 125
- prediction/reconciliation idiom, 42
- preference, 145
- preferences (in decision problems), 149
- prior belief, 125
- probabilities tab, 189
- probability
  - resources, 167
- probability biases, 157
- probability distribution, 122, 123
- probability distribution function, 24
- probability references, 167
- probability theory
  - axioms, 122
  - Bayesian approach, 121
  - different approaches, 121
  - frequentist approach, 121
  - fundamentals, 121
- process, 34
- process quality, 119
- process-product idiom, 38
- product, 34
- product rule, 126
- Programmable Electronic System, 13, 15
- propagate, 190
- propagation, 22, 131
  
- Railtrack, 114
- reconciliation idiom
  - in safety risk template, 73
- references on eliciting probabilities, 167
- reliability argument, 67
- Reliability Block Diagrams, 118
- reliability template, 64, 67
  - execution scenarios, 68
- representativeness, 158
- representativeness biases, 157
- resource, 34
- retract all evidence, 191
- retrievability of instances, 160
- rework process, 60
- risk assessment, 115
- risk factors, 152
- risk minimisation factors, 154
- root node, 14, 130
  
- safety
  - definition, 14

- safety analysis techniques, 14
- safety argument, 14, 19, 111
- safety case, 14, 19
- safety claim, 14
- safety development and assessment, 111
- safety evidence, 14
- safety integrity, 14
- safety lifecycle, 14, 117
- safety objective, 14
- safety risk argument
  - creating from templates, 75
  - execution scenarios, 76
- safety risk template, 70, 72
  - idiom instances, 71
- safety standards, 113
- safety-related system, 14
- satellite templates (in safety risk example), 72
- save project, 191
- save project as, 191
- save template, 191
- save template as, 191
- schemas, 164
- sensitivity analysis, 192
- separation, 122, 123, 126, 132
- SERENE, 14, 169
  - activities in the method, 25
  - argument construction, 26
  - argument preparation, 25
  - argument use, 27
  - BBNs and decision making, 155
  - benefits of, 30
  - constructing safety arguments, 33
  - context for, 111
  - decision making, 23
  - example of using the method, 49
  - examples of method, 59
  - how it fits in with GQM and MCDA, 155
  - objectives, 16
  - partner BBNs, 29
  - relationship to safety analysis techniques, 118
  - summary, 3
  - tool, 27, 81
- serial connection, 134
- soft evidence, 133
- Spurs, 121, 122
- stakeholders (in decision problem), 148
- standards, 113
- states tab, 189
- statistical determination, 140
- structural determination, 141
  
- template, 14, 23, 35, 186, 193
  - combining, 99
  - defect density example, 60
  - safety risk example, 70, 72
- template documentation, 182, 193
- template menu, 194
- template window, 194
- template window file menu, 181
- tools menu, 195
- TRACS, 169
- two\_phase\_rework template, 62
  
- uncertain criteria and inference, 151
- uncertain inference, 151
- uncertainty, 21
  - different types, 151
- Uniform distribution, 177
- utility links, 184
- utility node, 187
  
- vague criteria, 150
- variable, 122, 123
- variance, 162
  
- Weibull distribution, 177
- window menu, 195

## Distribution

European Commission	(EC)	(Accepted Report only)
ERA Technology	(ERA)	
Centre for Software Reliability	(CSR)	
Electricité De France	(EDF)	
HUGIN Expert A/S	(HE)	
Objectif Technologie	(OT)	
TÜV Nord	(TÜV)	
ERA Project Archives	(ERA)	