Copyright Statement

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand).

This thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.
- Authors control the copyright of their thesis. You will recognise the author’s right to be identified as the author of this thesis, and due acknowledgement will be made to the author where appropriate.
- You will obtain the author’s permission before publishing any material from their thesis.

To request permissions please use the Feedback form on our webpage. [http://researchspace.auckland.ac.nz/feedback](http://researchspace.auckland.ac.nz/feedback)

General copyright and disclaimer

In addition to the above conditions, authors give their consent for the digital copy of their work to be used subject to the conditions specified on the [Library Thesis Consent Form](http://researchspace.auckland.ac.nz/feedback) and [Deposit Licence](http://researchspace.auckland.ac.nz/feedback).

Note: Masters Theses

The digital copy of a masters thesis is as submitted for examination and contains no corrections. The print copy, usually available in the University Library, may contain corrections made by hand, which have been requested by the supervisor.
Methodologies for the Development of Qualitative Spatial and Temporal Reasoning Applications

Carl Schultz

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy in Computer Science,
The University of Auckland, 2010.
Abstract

A great variety of scientific, engineering-based, and commercial application domains are fundamentally grounded in concepts of space and time. Over the last three decades there has been significant interest in more human-focused and intuitive qualitative spatial and temporal reasoning (QSTR) methods, which address the inherent limitations of purely numerical approaches for reasoning about space and time. However, despite the extremely significant theoretical advances that have been made in the QSTR field, there is a distinct absence of commercial and industrial applications that utilise QSTR calculi. The central issue is that relatively little research has addressed the unique challenges of designing and developing QSTR-based applications in comparison to more traditional systems that employ numerical processing techniques.

The primary objective of this thesis is to support software engineering practitioners in the development of applications that utilise QSTR calculi. Five QSTR application case studies, which cover a range of diverse application domains, are presented and analysed throughout this thesis to motivate the development of effective methodologies. Furthermore, a comprehensive definition of QSTR applications is presented to provide a formal basis for establishing methodologies that address three major areas of QSTR application development: requirements specification, design, and validation.

Design methodologies are presented that enable developers to evaluate the efficacy of numerous QSTR calculi with respect to QSTR application functional requirements. Additionally, the design methodologies adapt object-oriented concepts and machine learning techniques to facilitate the development of custom, high level, application-specific qualitative relations and constraints.

Four key validation methodologies are adapted from well known techniques in software engineering: unit testing, integration testing, test coverage, and mutation testing. Furthermore, a novel metric called H-complexity is presented and used to define four additional test coverage classes that a developer can employ to assess the efficacy of a test suite.

Finally, a meta-validation methodology is established that enables developers and the QSTR community to empirically investigate the efficacy of QSTR validation techniques. Experiments are conducted using the meta-validation methodology and the results are analysed to identify the most effective utilisation of QSTR validation methodologies according to the software development process being employed.
For Dad.
Acknowledgements

I am deeply grateful to my supervisors Assoc. Prof. Robert Amor and Prof. Hans Guesgen for their generosity and enthusiastic support. I have found their patience and eager willingness to share their wealth of knowledge inspiring.

I would very much like to acknowledge the Tertiary Education Commission for awarding me the Bright Future Top Achiever Doctoral Scholarship, which provided financial support during my PhD.

I wish to thank my bright shining best friend Ljika for all of the beautiful fun adventures and excitement, the art and the music and the coffee, and for all the happiness that she has shared with me.

I would like to thank my Tetka Jelena, who is such a joy and a delight, for always looking after me.

And finally I would like to say a very special thank you to my dearest mother Merlene for her kindness, love, and support.
# Contents

1 Introduction

1.1 The Application of Qualitative Spatial and Temporal Reasoning .......................... 1
1.2 Objectives of this Thesis ....................................................................................... 3
1.3 State of the Art in Applying QSTR Calculi .......................................................... 4
1.4 Contributions of this Thesis ................................................................................ 6
1.5 Organisation of this Thesis .................................................................................. 7

2 Qualitative Spatial and Temporal Reasoning .......................................................... 11

2.1 Introduction ......................................................................................................... 11
2.2 Early QSTR Calculi ............................................................................................. 11
2.2.1 Allen’s Interval Algebra .................................................................................. 12
2.2.2 Guesgen’s Spatial Orientation Calculus (Rectangle Algebra) ......................... 13
2.2.3 Region Connection Calculus ......................................................................... 15
2.2.4 Other Early QSTR Calculi Motivated by Allen’s Calculus ............................... 17
2.3 Unifying, Generalising and Augmenting QSTR Calculi ........................................ 17
2.4 Algebraic Formulation of QSTR Calculi ............................................................. 19
2.5 Underlying Domains of Interpretation ................................................................. 20
2.6 Reasoning using Qualitative Spatial and Temporal Calculi ................................. 21
2.6.1 Path Consistency, Algebraic Closure, and Weak Composition ....................... 22
2.6.2 Tractability .................................................................................................. 23
2.7 Conceptual Neighbourhoods ............................................................................... 24
2.8 Case Studies of QSTR Applications .................................................................... 26
2.8.1 QtvLight: Qualitative Decision Support for Architectural Lighting .............. 26
2.8.2 TreeSap: Geographic Information System ..................................................... 27
2.8.3 SPBD: Qualitative Interface for Robotic Programming by Demonstration .... 28
2.8.4 Ul-Qayyum’s Qualitative Image Retrieval System .......................................... 31
2.8.5 SailAway: Maritime Qualitative Navigation .................................................. 31
2.9 Summary .............................................................................................................. 32
3 Theoretical Foundations of QSTR Applications

3.1 Introduction .................................................. 35
3.2 Formal Definition of QSTR Applications ...................... 36
3.3 Actors and Roles .............................................. 39
3.4 Ambiguity ...................................................... 40
3.5 Fundamental Operations on Scenarios ......................... 40
3.6 Characteristics of QSTR Applications ......................... 41
   3.6.1 Reasoning Across a Broad Range of Abstraction Levels .... 42
   3.6.2 Continuity Assumption and Neighbourhoods for Changing Scenarios .... 42
   3.6.3 Modelling Infinite Domains ................................ 42
   3.6.4 Reasoning About Objects in Multi-Dimensional Models .......... 43
3.7 Summary ....................................................... 43

4 Formal Software Requirements for QSTR Applications ......... 45

4.1 Introduction .................................................. 45
4.2 External Interface Requirements ................................ 48
4.3 Specifying Functional Requirements with an Idealised Application ... 50
4.4 A Complete Enumeration of QSTR Tasks ..................... 51
   4.4.1 Deriving Standard QSTR Tasks Using Fundamental Operations .... 53
   4.4.2 Deriving QSTR Tasks for Unordered Scenarios .................. 56
   4.4.3 Deriving QSTR Tasks for Ordered Scenarios .................... 58
   4.4.4 Deriving QSTR Tasks for Partially Ordered Scenarios .......... 64
4.5 Characterising QSTR Application Execution Behaviour ......... 65
4.6 Performance Requirements .................................... 68
4.7 Summary ....................................................... 70

5 Organising and Visualising QSTR Applications ................. 73

5.1 Introduction .................................................. 73
5.2 Organising Relations and Constraints Into Model Fragments .......... 74
5.3 Mapping Object Oriented Concepts to the QSTR Application Domain ... 77
   5.3.1 Design Pattern: Fragment Definitions ......................... 78
   5.3.2 Design Pattern: Fragment Generalisation ....................... 80
5.4 Adapting UML Sequence Diagrams to Visualise QSTR Application Behaviour 81
5.5 Summary ....................................................... 84

6 Designing Low Level Application Fragments: Selecting QSTR Calculi .... 87

6.1 Introduction .................................................. 87
6.2 Defining Functional Consistency ................................ 91
6.3 Determining Functional Consistency in Consistent Unambiguous Scenarios .... 92
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.4</td>
<td>Determining Functional Consistency in Consistent Ambiguous Scenarios</td>
<td>96</td>
</tr>
<tr>
<td>6.5</td>
<td>Determining Functional Consistency in Inconsistent Scenarios</td>
<td>97</td>
</tr>
<tr>
<td>6.6</td>
<td>Functional Consistency Analysis Methodology</td>
<td>98</td>
</tr>
<tr>
<td>6.7</td>
<td>Generating the Strongest QSTR Theory from the Idealised Application</td>
<td>106</td>
</tr>
<tr>
<td>6.8</td>
<td>Summary</td>
<td>107</td>
</tr>
<tr>
<td>7</td>
<td>Designing High Level Application Fragments: Utilising Data-Based Metrics and Deriving High Level Neighbourhood Graphs</td>
<td>109</td>
</tr>
<tr>
<td>7.1</td>
<td>Introduction</td>
<td>109</td>
</tr>
<tr>
<td>7.2</td>
<td>Analysing Fragment Interaction using Coupling and Cohesion</td>
<td>111</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Utilising Coupling and Cohesion to Support High Level Fragment Design</td>
<td>115</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Accuracy and Precision for Analysing Fragment Definitions</td>
<td>116</td>
</tr>
<tr>
<td>7.3</td>
<td>Generating High Level Neighbourhoods using Decision Tree Learning</td>
<td>117</td>
</tr>
<tr>
<td>7.4</td>
<td>Designing Consistent High Level Neighbourhood Graphs</td>
<td>121</td>
</tr>
<tr>
<td>7.4.1</td>
<td>Methodology for Deriving a High Level Neighbourhood Graph</td>
<td>121</td>
</tr>
<tr>
<td>7.4.2</td>
<td>Formally Defining Conceptual Neighbours as Transitions Between Low Level Relations</td>
<td>122</td>
</tr>
<tr>
<td>7.4.3</td>
<td>Refining Conceptual Neighbours: Path Restrictions</td>
<td>125</td>
</tr>
<tr>
<td>7.4.4</td>
<td>Refining Conceptual Neighbours: Transition Path Equivalence Classes</td>
<td>126</td>
</tr>
<tr>
<td>7.4.5</td>
<td>Ensuring Conceptual Neighbours are Symmetric</td>
<td>127</td>
</tr>
<tr>
<td>7.4.6</td>
<td>Dealing With Multiple Fragment Definitions</td>
<td>127</td>
</tr>
<tr>
<td>7.5</td>
<td>Summary</td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>Implementing QSTR Applications</td>
<td>131</td>
</tr>
<tr>
<td>8.1</td>
<td>Introduction</td>
<td>131</td>
</tr>
<tr>
<td>8.2</td>
<td>Languages and Architecture</td>
<td>131</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Declarative Languages: Prolog</td>
<td>132</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Relational Database Languages: SQL</td>
<td>132</td>
</tr>
<tr>
<td>8.3</td>
<td>Encoding Relations and Scenarios</td>
<td>134</td>
</tr>
<tr>
<td>8.4</td>
<td>Implementing General Reasoning</td>
<td>134</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Implementing Constraints</td>
<td>135</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Inference Rules and Deductive Closure</td>
<td>136</td>
</tr>
<tr>
<td>8.4.3</td>
<td>Checking Consistency</td>
<td>138</td>
</tr>
<tr>
<td>8.5</td>
<td>Resource Constraints</td>
<td>140</td>
</tr>
<tr>
<td>8.6</td>
<td>Summary</td>
<td>141</td>
</tr>
<tr>
<td>9</td>
<td>Test Classes and Test Quality Metrics for Validating QSTR Applications</td>
<td>143</td>
</tr>
<tr>
<td>9.1</td>
<td>Introduction</td>
<td>143</td>
</tr>
<tr>
<td>9.2</td>
<td>Overview of QSTR Validation Methodologies</td>
<td>144</td>
</tr>
</tbody>
</table>
9.2.1 Formal Definitions for QSTR Validation .................................... 146
9.3 Unit Testing ................................................................. 147
  9.3.1 Merging Decision Tables .............................................. 149
  9.3.2 Equivalence Class: Decision Table Row Combinations .......... 149
  9.3.3 Equivalence Class: Interaction of Rows With Different Condition .. 150
  9.3.4 Refined Unit Testing .................................................. 151
9.4 Integration Testing ....................................................... 152
  9.4.1 Interaction Between Expressions in The Same Constraint .......... 153
  9.4.2 Interaction Between Expressions in Different Constraints ....... 153
9.5 Test Coverage for Test Set Evaluation ................................... 154
  9.5.1 Sentence Interaction Graph .......................................... 155
  9.5.2 Test Coverage Methodology Overview ............................... 156
  9.5.3 Test Coverage Metrics ................................................ 156
9.6 Fault Seeding for Test Set Evaluation ................................... 160
  9.6.1 QSTR Mutation Testing Methodology ................................. 160
  9.6.2 Selecting and Evaluating QSTR Mutation Operators ............... 161
  9.6.3 abs Mutation Operator ............................................... 163
  9.6.4 ror Mutation Operator ............................................... 164
  9.6.5 CRT and ICE Mutation Operators ................................... 165
  9.6.6 Method Overloading: NEM and NR Mutation Operators ......... 166
  9.6.7 Remove Relation ...................................................... 167
9.7 Summary ................................................................. 168

10 H-Complexity for Analysing and Validating QSTR Applications 169
  10.1 Introduction ............................................................ 169
  10.2 H-Complexity Theory .................................................. 170
    10.2.1 Homogeneous Sets ............................................... 170
    10.2.2 H-complexity: Measuring Complexity With H Sets ............. 170
    10.2.3 Calculating H-Complexity ....................................... 171
    10.2.4 Partial Basic Queries, Partial H Sets and Partial Scenarios .. 175
  10.3 Using H-complexity to Analyse QSTR Applications .................. 176
    10.3.1 Relation Arity .................................................... 176
    10.3.2 Number of Query Variables ..................................... 178
    10.3.3 Allowable Relation States ...................................... 181
  10.4 Using H-complexity to Quantify Test Coverage ....................... 181
    10.4.1 Activity 1: Identify Component Domains ....................... 183
    10.4.2 Activity 2: Specify Conditions to Refine the Test Space ..... 183
    10.4.3 Activity 3: Calculate H-Complexity for Refined Test Spaces .. 183
10.4.4 Activity 4: Calculate the Class of a Given Test Instance ........................................ 186
10.4.5 Activity 5: H-Complexity Coverage Metrics ......................................................... 187
10.4.6 Comparison Between Adapted and H-Complexity Test Coverage Metrics .................. 189
10.4.7 Utilising Adapted and H-Complexity Test Coverage Metrics .................................... 193
10.5 Summary .................................................................................................................. 194

11 Evaluating Validation Techniques ............................................................................. 197
11.1 Introduction .............................................................................................................. 197
11.2 Evaluation Methodology ............................................................................................ 198
  11.2.1 Metrics for Assessing Test Approach Efficacy ....................................................... 199
  11.2.2 Discrete Sampling Theory ..................................................................................... 202
11.3 Generating Random Subject QSTR Applications ..................................................... 204
  11.3.1 Random Variables for Representing the Distribution of QSTR Applications .......... 206
  11.3.2 Parameters for the Structured Model of QSTR Applications ............................... 208
  11.3.3 Parameters for Constraints within a Fragment ...................................................... 209
  11.3.4 Parameters for Constraints between Fragments ................................................... 210
11.4 An Experiment for Evaluating the Presented QSTR Validation Techniques ............... 211
  11.4.1 Experiment Procedure ......................................................................................... 211
  11.4.2 Experiment Configuration and Parameter Values for Generating Random QSTR Applications .......................................................... 213
  11.4.3 Pilot Study for Estimating the Standard Deviation of Mutation Scores .................. 215
  11.4.4 Experiment Results and Analysis ........................................................................ 217
11.5 Summary .................................................................................................................. 225

12 Conclusions ................................................................................................................. 229
12.1 Future Research ........................................................................................................ 237
12.2 QSTR Application Development using Functional Consistency Analysis and Future Research ........................................................................................................... 237
12.3 Discussion of Limitations and Future QSTR Meta-Validation Research ..................... 238

References ..................................................................................................................... 241

A1 ........................................................................................................................................ 257
  A.1 Unit Tests .................................................................................................................. 257
  A.2 Vertex Tests ............................................................................................................. 257
  A.3 Test Script ................................................................................................................ 258
10.4.4 Activity 4: Calculate the Class of a Given Test Instance ............ 186
10.4.5 Activity 5: H-Complexity Coverage Metrics .......................... 187
10.4.6 Comparison Between Adapted and H-Complexity Test Coverage Met-
rics .................................................................................. 189
10.4.7 Utilising Adapted and H-Complexity Test Coverage Metrics ........ 193
10.5 Summary ................................................................. 194

11 Evaluating Validation Techniques ........................................... 197
11.1 Introduction ............................................................... 197
11.2 Evaluation Methodology ................................................ 198
  11.2.1 Metrics for Assessing Test Approach Efficacy ..................... 199
  11.2.2 Discrete Sampling Theory ......................................... 202
11.3 Generating Random Subject QSTR Applications ...................... 204
  11.3.1 Random Variables for Representing the Distribution of QSTR Applica-
tions ............................................................................. 206
  11.3.2 Parameters for the Structured Model of QSTR Applications ...... 208
  11.3.3 Parameters for Constraints within a Fragment ..................... 209
  11.3.4 Parameters for Constraints between Fragments ................. 210
11.4 An Experiment for Evaluating the Presented QSTR Validation Techniques .... 211
  11.4.1 Experiment Procedure ............................................. 211
  11.4.2 Experiment Configuration and Parameter Values for Generating Ran-
dom QSTR Applications .................................................. 213
  11.4.3 Pilot Study for Estimating the Standard Deviation of Mutation Scores . 215
  11.4.4 Experiment Results and Analysis .................................. 217
11.5 Summary ................................................................. 225

12 Conclusions ................................................................. 229
12.1 Future Research .......................................................... 237
12.2 QSTR Application Development using Functional Consistency Analysis and
  Future Research ............................................................. 237
12.3 Discussion of Limitations and Future QSTR Meta-Validation Research .... 238

References ................................................................. 241

Appendix A ................................................................. 257
  A.1 Unit Tests .............................................................. 257
  A.2 Vertex Tests ............................................................ 257
  A.3 Test Script ............................................................... 258
# List of Figures

2.1 An example scenario of blocks illustrating Guesgen’s qualitative relations between objects (reproduced from [78]). ........................................ 15
2.2 The conceptual neighbourhood of Allen’s basic interval relations. ........... 25
2.3 Screenshot of the interface used to specify a query in TreeSap GIS. ............ 28
2.4 Screenshot of the transparency method used in TreeSap GIS to visualise results of the query “Find all Roads near a Specific Building (black circle)” .... 29
2.5 Screenshot of the display threshold method used in TreeSap GIS to visualise results of the query “Find all Roads near a Specific Building (black circle)” for threshold values 100%, 75%, and 50%. ................................. 29
2.6 Sonar configuration in the Pioneer 3DX robot sonar ring. .......................... 30
4.1 A selection of associated external interface QSTR problem characteristics, ordered according to their dependencies. ................................. 49
4.2 Use case relationships for basic QSTR tasks. ........................................... 52
4.3 QSTR application behaviour during the execution of a software application. .... 66
4.4 A selection of associated functional and performance QSTR problem characteristics, ordered according to their dependencies. ................................. 68
5.1 Example adapting sequence diagram notation to allow classifiers to represent multiple fragments, and to allow objects to appear in multiple classifiers. ...... 82
5.2 Using standard sequence diagram notation to illustrate semantics of the leftmost multi-fragment classifier in Figure 5.1. ................................. 82
5.3 Example using standard sequence diagram notation for classifiers with single fragments. ................................................................. 84
6.1 Scenario for which the idealised application produces output $\alpha$. Interval $x$ must always be before $z$. ........................................... 89
6.2 Scenario for which the idealised application produces output $\beta$. Region $z$ is necessarily connected to $x$. ........................................... 90
7.1 Mean, upper quartile, and lower quartile cohesion and coupling distances of image categories. Landscape with mountains is abbreviated to lwm.

7.2 Graph of the near and far clustering from Table 7.1.

7.3 Partial decision Tree representation of the fragment definition of coast images.

7.4 Derived neighbourhood graph from generated fragment definitions presented in Table 7.2.

7.5 Reference domain fragment $F_2$ containing relations $R_1, \ldots, R_8$ and $R'_1, \ldots, R'_6$ with simple ordered neighbourhoods (left). The fragment definition space (right) consists of fragment definitions that specify one relation from $R_1, \ldots, R_8$ and one relation from $R'_1, \ldots, R'_6$.

7.6 Refined fragment definition spaces. (a) direct path restrictions, (b) critical path restrictions, (c) equivalence class of direct paths, (d) equivalence class of critical paths in conjunction with a direct path restriction.

8.1 Architectural overview of a Prolog implementation of a QSTR application.

8.2 Architectural overview of a relational database implementation of a QSTR application.

9.1 Flow graph of the QSTR validation process.

9.2 Sentence Interaction Graph (SIG) of a selection of domain rules for architectural lighting.

9.3 Overview of the process of calculating test coverage.

9.4 Partial ordering of coverage criteria from weakest to strongest (if full coverage is achieved).

9.5 Sentence interaction graph of the example application.

9.6 Overview of the process of calculating mutation score.

10.1 Two graphs representing binary relations $R_1$ and $R_2$.

10.2 Graph consisting of three disconnected subgraphs. Subgraphs $\{a, b\}$ and $\{c, d\}$ correspond to basic queries, where $\{e, f, g\}$ contains more than one basic query as induced subgraphs.

10.3 Overview of the process of calculating H-complexity coverage scores (regarding the input “get app design”, note that the developer may only be analysing a portion of the full QSTR application).

11.1 Overview of evaluation methodology for assessing QSTR validation techniques.

11.2 The estimated probability mass function, $f_{X|R_1}$, of the total number of relations in a QSTR application, generated by first specifying the number of fragments, and then incrementally populating each fragment with a random number of relations, uniformly distributed between 1 and 10.
List of Tables

2.1 The thirteen basic qualitative relations between pairs of time intervals in Allen’s Interval Algebra. ................................................................. 12
2.2 The nine basic qualitative relations between pairs of objects in Guesgen’s Rectangle Algebra. ................................................................. 14
2.3 The eight basic qualitative relations between pairs of regions in Region Connection Calculus. ................................................................. 16

3.1 Comparing the domains of model theory, QSTR applications, and the roles of QSTR application designers and users. ................................. 40
3.2 Permitted operations on QSTR scenarios based on the combination of application components that are variable. ........................................ 41

4.1 Comparing the QSTR application requirements methodologies presented in this chapter with IEEE Recommended Practice for Software Requirements Specifications (IEEE 830-1998) [179]. ......................................................... 47
4.2 Tasks that can be performed on a scenario with respect to the underlying parameters. In the state diagram, states are black circles, tasks (composed of states) are ovals with a task label, terminating states are double-lined circles, arrows are state transitions annotated with QSTR operations, and the arrow with no source state is the task entry point. ............................................. 55
4.3 Tasks that can be performed on an unordered set of scenarios with respect to the underlying parameters. ................................................. 57
4.4 Icons illustrating the important permutations of scenarios in an ordered sequence that are used to define a set of basic QSTR tasks. .................. 59
4.5 Tasks that can be performed on an ordered sequence of scenarios with respect to the underlying parameters. ............................................. 63
4.6 Tasks that can be performed on a partially ordered set of scenarios with respect to the underlying parameters. ............................................. 65
4.7 Enumerating relevant paths through the QSTR application statechart in Figure 4.3 to determine basic application behavioural patterns. Characteristic values that uniquely define each behavioural class are bolded in the right column.

4.8 Each of the six application behaviour classes derived in Table 4.7 can be uniquely determined by their change source and task category characteristics.

5.1 Comparison of QSTR applications and libraries based on the number of fragments and the number of relations per fragment. For each application the fragments are ordered by the level of abstraction, from highest to lowest.

5.2 Comparison of QSTR applications and libraries based on the number of high, intermediate, and low level fragments, and the number of relations per fragment.

5.3 Comparison of the custom qualitative components of QSTR applications and libraries (i.e. excluding QSTR calculi), based on the number of high, intermediate, and low level fragments, and the number of relations per fragment.

5.4 Adapting standard UML class diagram notation and concepts [141] to the QSTR application domain.

5.5 Adapting standard UML sequence diagram notation and concepts [141] to the QSTR application domain.

6.1 Six cases, defined as allowable combinations of four conditions.

6.2 Determining functional consistency depending on whether the scenario is guaranteed to be consistent or the scenario is possibly inconsistent, and whether the scenario is unambiguous (the network is atomic) or ambiguous (the network is non-atomic).

6.3 Classes of scenarios for which the QSTR application using IA will be functionally inconsistent.

6.4 Classes of scenarios for which the QSTR application using RCC8 will be functionally inconsistent.

6.5 Classes of scenarios for which the tank-valve application by [126] using IA will be functionally inconsistent.

7.1 Near and far clusters formed using mean coupling distance thresholds. Near threshold is cohesion mean +10%, far threshold is greatest coupling mean −10%.

7.2 Fragment definitions generated using the J48 decision tree learning algorithm, implemented in Weka [168], with confidence 0.25.

7.3 QSTR application developer guidelines for deriving effective neighbourhood graphs.

9.1 Decision tables for conditions in the exists clause (upper) and not exists clause (lower) of the left hand side set expression in the example constraint.
9.2 Two three-dimensional matrices representing the left hand side set expression of the architectural lighting constraint in Example 9.3.1. 150
9.3 Decision tables for conditions in the exists clause (upper) and not exists clause (lower) of the relevant portions of the example set expression that separate triggers and overrides. 152
9.4 One-dimensional matrix representing the example set expression after merging the decision tables in Table 9.3. 152
9.5 Adapted coverage criteria for the example application. 159
9.6 Adapting mutation operators for imperative and OO software to QSTR applications. 164

10.1 Basic queries are derived from different combinations of relation state tuples. 174
10.2 Homogeneous (H) sets are derived from different combinations of basic queries. 174
10.3 Scenarios are derived from different combinations of H sets being non-empty. 175
10.4 Common conditions for refining test spaces. 184

11.1 Mean and standard deviation of mutation scores (to 4 decimal places) for five testing approaches resulting from the pilot study with sample size \( n_p = 30 \). The cell with the highest standard deviation (Random Testing, Instance Creation Expression) is highlighted. 216
11.2 Mean and standard deviation of the number of components per QSTR application (to 4 decimal places). 217
11.3 Mean and standard deviation of the number of components per QSTR application to 4 decimal places. 218
11.4 Mean and standard deviation of the number of mutants generated per QSTR application to 4 decimal places. 219
11.5 Linear correlation between the number of mutants generated and the number of different application components. 220
11.6 Estimated means \((\bar{y}_S \sim \bar{y}_U)\) and standard deviations \((s \sim S)\) of mutation scores for the five test approaches for four different fault types. For each fault type, highlighted cells contain the highest mean mutation score. 221
11.7 Minimum, mean, standard deviation of, and maximum number of tests in a testset for each of the five test approaches, where one testset is used to test one application. 223
11.8 Mean and standard deviation of the functional efficacy percentage scores for each test approach with respect to different fault types. The highlighted cells indicate the highest functional efficacy scores. 223
11.9 Minimum, mean, standard deviation of, and maximum number of assertions in a testset for each of the five test approaches. 224
11.10 Mean and standard deviation of the number of inference steps required to determine an expected test result, and the mean and standard deviation of the number of constraints with non-empty domains in the initial premise scenario of a test (thus requiring further inspection by a human oracle). . . . . . . . . . . . . . . 225

11.11 Mean and standard deviation of the processing time in minutes per application (including all mutants) required by each validation method. . . . . . . . . . . . . . . 225