

# THE USABILITY OF CONSTRAINT DIAGRAMS

NEIL MORGAN

A thesis submitted in partial fulfilment of the  
requirements of the University of Brighton for  
the degree of Master of Philosophy

November 2011

The University of Brighton

# Abstract

The Unified Modelling Language (UML) standard supports a notation, The Object Constraint Language (OCL), for modelling constraints. As a textual language, it is expressive, providing a full specification of constraints of a system, while not requiring a symbolic mathematical background.

The UML diagrams are not precise enough on their own to provide an unambiguous specification of a system, hence the need to define further constraints on the model to reduce or remove ambiguity. Previously written in natural language or formal notations, there were still problems with ambiguity and / or understanding of the model. This set of problems gave rise to OCL.

In UML, all notations are graphical except for OCL, which is textual. Constraint Diagrams were developed jointly by the Universities of Brighton and Kent to provide a diagrammatic representation of constraints. Semantically, Constraint Diagrams and OCL are thought to be equivalent i.e. what can be expressed in OCL can be expressed in Constraint Diagrams and vice versa. A lot of effort has been put into the mathematics to ensure this. However, while the logic of both may be equivalent, one question must be addressed before Constraint Diagrams can be considered a practical modelling tool, "How usable are they?".

The question of usability is an important one. Logically, usability provides the necessary end products to improve models of systems and as such is of great merit. However, if software engineers are to use this tool effectively, we must study the usability, both in view of the notation itself and potential Constraint Diagram tool generation. A device (notation or tool) that is both easy to use and expressive enough to inform non-experts (e.g. programmers, managers, etc) is more effective than one which sacrifices either ease of use or expressiveness.

The research in this thesis explores the usability of the Constraint Diagram notation and provides a comparison with the OCL.

# Acknowledgements

I would like to thank my supervisors Professor John Howse and Dr Lyn Pemberton. John in particular afforded me with the support and enthusiasm I needed to motivate me to complete this thesis. I would also like to express my gratitude to Liz Cheek, without whom the statistics would have been merely rudimentary.

Thanks to all participants who took part in the studies.

Thanks, also, to the Visual Modelling Group for their assistance and valuable feedback when required.

Finally, thanks to Karen, Tommy and Joey for keeping me on the “Straight and Narrow”.

# Declaration

I declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the original work of the author. The thesis has not been previously submitted to this or any other university for a degree, and does not incorporate any material already submitted for a degree.

Signed .....

Dated .....

## Contents (Summary)

Contents (Summary).....	1
Contents (Full).....	2
1 Introduction.....	16
2 Study 1: Interpreting constraints using constraint diagrams.....	46
3 Quantitative analysis of the interpretation of constraint diagrams.....	53
4 Conclusions gained from the quantitative analysis.....	114
5 Qualitative analysis of the interpretation of constraint diagrams.....	121
6 Study 2: Developing constraints using constraint diagrams and OCL.....	128
7 Quantitative analysis of the development of constraint diagram and OCL operations.....	133
8 Conclusions gained from the quantitative analysis.....	164
9 Qualitative analysis of the development of constraint diagram and OCL operations.....	172
10 Conclusions and Further Work.....	184
11 Bibliography.....	189
12 Appendix I.....	193
13 Appendix II.....	204
14 Appendix III.....	207
15 Appendix IV.....	216
16 Appendix V.....	228

## Contents (Full)

Contents (Summary) .....	1
Contents (Full) .....	2
1 Introduction .....	16
1.1 Historical context .....	16
1.2 Related work.....	18
1.2.1 The Unified Modelling Language.....	18
1.2.2 The Object Constraint Language .....	21
1.2.3 Design by Contract.....	24
1.2.4 An example of UML and OCL .....	24
1.2.5 Some shortcomings of OCL.....	25
1.2.6 Constraint Diagrams.....	27
1.2.7 Euler Circles, Venn Diagrams and Spider Diagrams.....	28
1.2.8 Elements of Constraint Diagrams .....	29
1.2.9 Contours .....	30
1.2.10 Districts, Regions and Zones.....	30
1.2.11 Spiders.....	31
1.2.12 Shading.....	32
1.2.13 An example of Constraint Diagrams.....	33
1.3 Usability .....	33

1.3.1	ISO Standards for Usability .....	34
1.3.2	Quantifying Usability.....	36
1.4	The Cognitive Dimensions of Notations Framework.....	39
1.5	Empirical Study .....	43
1.6	The rest of this thesis .....	45
2	Study 1: Interpreting constraints using constraint diagrams .....	46
2.1	Hypotheses .....	46
2.2	Study focus .....	47
2.2.1	Aims .....	47
2.3	Method.....	47
2.3.1	Computer-based exercises.....	48
2.3.2	Paper-based questionnaire.....	49
2.3.3	Closeness of mapping .....	49
2.3.4	Consistency .....	49
2.3.5	Role expressiveness .....	49
2.3.6	Secondary notation.....	49
2.3.7	Overview document .....	50
2.3.8	Undertaking the “interpretation” study .....	50
2.3.9	Reducing threats to validity .....	51
2.3.10	Briefing .....	52
2.3.11	Taking the test.....	52

2.3.12	Debriefing .....	52
3	Quantitative analysis of the interpretation of constraint diagrams .....	53
3.1	Confidence Interval .....	56
3.2	Summarising the results .....	57
3.3	Question 1.....	58
3.4	Question 2.....	62
3.5	Question 3.....	66
3.6	Question 4.....	68
3.7	Question 5.....	72
3.8	Question 6.....	74
3.9	Question 7.....	78
3.10	Question 8 .....	82
3.11	Question 9.....	85
3.12	Question 10.....	89
3.13	Hypothesis testing and inference .....	93
3.14	Hypothesis tests of differences between groups.....	94
3.15	Hypothesis tests of differences between the groups using non-parametric methods	96
3.16	Modelling the probability of obtaining a correct response.....	101
3.17	Investigating relationships between group, time taken and accuracy of response	104



3.18	Further investigations of the relationships between group, time taken and accuracy of response using a log-transformed time .....	108
3.19	Modelling the time taken to complete each question .....	111
4	Conclusions gained from the quantitative analysis .....	114
4.1	The findings of the descriptive statistics .....	114
4.2	The findings of the inferential statistics .....	117
5	Qualitative analysis of the interpretation of constraint diagrams .....	121
5.1	Closeness of mapping.....	121
5.2	Consistency .....	121
5.3	Role expressiveness.....	121
5.4	Secondary notation .....	122
5.5	The paper questionnaire .....	122
5.6	A partial cognitive dimensions profile for constraint diagrams .....	122
6	Study 2: Developing constraints using constraint diagrams and OCL .....	128
6.2	Method.....	129
6.2.1	The paper-based questionnaire.....	129
6.2.2	The overview document.....	130
6.2.3	The constraint development document .....	130
6.2.4	The multiple choice questionnaire .....	130
6.2.5	Undertaking the “development” study.....	130
6.2.6	Threats to validity .....	131
6.2.7	Taking part in the study.....	131

6.2.8	Taking the test .....	132
6.2.9	Debriefing .....	132
7	Quantitative analysis of the development of constraint diagram and OCL operations.....	133
7.1	Confidence interval .....	133
7.2	Summarising the results .....	135
7.3	Metric 1 .....	137
7.4	Metric 2 .....	139
7.5	Metric 3 .....	141
7.6	Metric 4 .....	143
7.7	Metric 5 .....	145
7.8	Metric 6 .....	147
7.9	Metric 7 .....	149
7.10	Metric 8.....	151
7.11	Metric 9.....	153
7.12	Metric 10.....	155
7.13	Metric 11.....	157
7.14	Hypothesis testing and inference .....	159
7.15	Hypothesis tests of differences between groups .....	159
8	Conclusions gained from the quantitative analysis.....	164
8.1	The conduct of the study: sampling over two years .....	164
8.2	The findings of the descriptive statistics .....	164

8.3	The findings of the inferential statistics .....	170
9	Qualitative analysis of the development of constraint diagram and OCL operations.....	172
9.1	The paper questionnaire .....	172
9.2	A full cognitive dimensions profile for constraint diagrams.....	172
10	Conclusions and Further Work .....	184
10.1	Conclusions.....	184
10.2	Further Work.....	185
11	Bibliography.....	189
12	Appendix I.....	193
13	Appendix II .....	204
14	Appendix III.....	207
15	Appendix IV.....	216
16	Appendix V .....	228

## List of Figures

Figure 1: UML Class Diagram of a mortgage application .....	24
Figure 2: an example Spider Diagram with description, taken from [24] .....	29
Figure 3: example contours.....	30
Figure 4: map showing zones within contours .....	31
Figure 5: example of a spider.....	31
Figure 6: example of a strand.....	31
Figure 7: example of a tie .....	32
Figure 8: example of an arrow representing a relation or function.....	32
Figure 9: example of shading with single spider (singleton set) .....	32
Figure 10: Constraint diagram showing the number of books held by a library .....	33
Figure 11: All copies are made from one original (option 2) .....	58
Figure 12: All employed persons must be aged 18 or over (option 3) .....	62
Figure 13: A person must be either married or unmarried but not both (option 3) ...	66
Figure 14: All driving instructors hold an Advanced Driving Licence and an Instructor Certificate and give driving lessons (option 1).....	68
Figure 15: All copies of a book that are out must be associated with a current loan (option 4).....	72
Figure 16: The total number of books must equal the number of copies available plus the number of copies borrowed (option 2).....	74
Figure 17: Copies of books that are on hold are held against reservations (option 2) .....	78

Figure 18: A married adult is married to one spouse who is not a sibling (option 1)	82
Figure 19: All books with a number of copies available are InCollection while all books with no copies available are ExCollection (option 3)	85
Figure 20: The specification of a car assigned to a reservation must be the same or better than the specification reserved (option 1)	89
Figure 21: Box plot of correct responses by group	95
Figure 22: Box plot of partially correct responses by group	95
Figure 23: Box plot of incorrect responses by group	96
Figure 24: Bar graph representing correct responses by group	99
Figure 25: Bar graph representing correct and partially correct responses by group	100
Figure 26: Estimated marginal means of time against group for correct vs. incorrect responses.	106
Figure 27: Box plots of responses plotted against time and group.	107
Figure 28: Estimated marginal means of log(time) against group for correct vs. incorrect responses.	110
Figure 29: Box plots of responses plotted against log transformed time and group	111
Figure 30: Kaplan-Meier Survival Analysis Plot	113
Figure 31: Graphical representation showing the profile of responses from the questionnaire	127
Figure 32: Box plot of correct metrics by notation	160
Figure 33: Box plot of incorrect metrics by notation	160
Figure 34: Bar graph representing correct metrics by notation	163

Figure 35: Pre-condition for operation Module:SetPrerequisite( $m \rightarrow p$ ) .....	165
Figure 36: Correct post-condition for operation Module:SetPrerequisite( $m \rightarrow p$ )....	165
Figure 37: Incorrect post-condition for operation Module:SetPrerequisite( $m \rightarrow p$ ) .	165
Figure 38: OCL Statement with UML sketching.....	179
Figure 39: OCL Statement with UML sketching.....	180
Figure 40: OCL Statement with constraint diagram sketching.....	180
Figure 41: OCL Statement with constraint diagram sketching.....	181
Figure 42: Graphical representation showing the profile of responses from the questionnaire .....	183

## List of Tables

Table 1: Actual Percentage and Confidence Interval .....	56
Table 2: Descriptive Statistics for accuracy of responses to questions .....	57
Table 3: Crosstab of group by response for question 1 .....	58
Table 4: Chi-Square Tests for responses for question 1 .....	59
Table 5: Crosstab of group by transformed response for question 1 .....	60
Table 6: Chi-Square Tests for transformed responses for question 1 .....	60
Table 7: Crosstab of group by response for question 2 .....	62
Table 8: Chi-Square Tests for question 2 .....	63
Table 9: Crosstab of group by transformed response for question 2 .....	64
Table 10: Chi-Square Tests for transformed responses for question 2 .....	64
Table 11: Crosstab of group by response for question 3 .....	66
Table 12: Chi-Square Tests for question 3 .....	67
Table 13: Crosstab of group by response for question 4 .....	69
Table 14: Chi-Square Tests for question 4 .....	69
Table 15: Crosstab of group by transformed response for question 4 .....	70
Table 16: Chi-Square Tests for transformed responses for question 4 .....	70
Table 17: Crosstab of group by response for question 5 .....	72
Table 18: Chi-Square Tests for question 5 .....	73
Table 19: Crosstab of group by response for question 6 .....	75
Table 20: Chi-Square Tests for question 6 .....	75

Table 21: Crosstab of group by transformed response for question 6 .....	76
Table 22: Chi-Square Tests for transformed responses for question 6.....	77
Table 23: Crosstab of group by response for question 7 .....	79
Table 24: Chi-Square Tests for question 7 .....	79
Table 25: Crosstab of group by transformed response for question 7 .....	80
Table 26: Chi-Square Tests for transformed responses for question 7.....	81
Table 27: Crosstab of group by response for question 8 .....	83
Table 28: Chi-Square Tests for question 8 .....	83
Table 29: Crosstab of group by response for question 9 .....	86
Table 30: Chi-Square Tests for question 9 .....	86
Table 31: Crosstab of group by transformed response for question 9 .....	87
Table 32: Chi-Square Tests for transformed responses for question 9.....	88
Table 33: Crosstab of group by response for question 10 .....	90
Table 34: Chi-Square Tests for question 10 .....	90
Table 35: Crosstab of group by transformed response for question 10.....	91
Table 36: Chi-Square Tests for transformed responses for question 10.....	92
Table 37: Descriptive Statistics for responses to all 10 questions combined .....	93
Table 38: Tests for Normality for responses to all 10 questions combined.....	94
Table 39: Non-parametric tests (descriptive statistics) for Mann-Whitney test .....	96
Table 40: Mann-Whitney Test (non-parametric) Ranks .....	97
Table 41: Mann-Whitney Test (non-parametric) Test Statistics for Group.....	97



Table 42: Mann-Whitney Test (non-parametric) Means and Medians.....	98
Table 43: Logistic regression: crosstabulation .....	101
Table 44: Logistic regression: chi-square tests.....	101
Table 45: Logistic regression: Risks.....	102
Table 46: Logistic regression: variables .....	103
Table 47: Univariate ANOVA between-subjects factors.....	104
Table 48: Univariate ANOVA between-subjects descriptive statistics .....	105
Table 49: Univariate ANOVA tests of between-subjects effects .....	105
Table 50: Univariate ANOVA - log transformation for between-subjects factors..	108
Table 51: Univariate ANOVA - log transformation for between-subjects descriptive statistics.....	108
Table 52: Univariate ANOVA - log transformation for between-subjects effects ..	109
Table 53: Kaplan-Meier Survival Analysis: Case processing summary .....	112
Table 54: Kaplan-Meier Survival Analysis: Means and medians for survival time	112
Table 55: Kaplan-Meier Survival Analysis: Overall comparisons .....	112
Table 56: Actual Percentage and Confidence Interval .....	134
Table 57: Descriptive Statistics for metrics .....	135
Table 58: Crosstab of notation by metric for metric 1.....	137
Table 59: Chi-Square tests for notation by metric for metric 1 .....	138
Table 60: Crosstab of notation by metric for metric 2.....	139
Table 61: Chi-Square tests for notation by metric for metric 2 .....	140
Table 62: Crosstab of notation by metric for metric 3.....	141

Table 63: Chi-Square tests for notation by metric for metric 3 .....	142
Table 64: Crosstab of notation by metric for metric 4.....	143
Table 65: Chi-Square tests for notation by metric for metric 4.....	144
Table 66: Crosstab of notation by metric for metric 5.....	145
Table 67: Chi-Square tests for notation by metric for metric 5 .....	146
Table 68: Crosstab of notation by metric for metric 6.....	147
Table 69: Chi-Square tests for notation by metric for metric 6 .....	148
Table 70: Crosstab of notation by metric for metric 7.....	149
Table 71: Chi-Square tests for notation by metric for metric 7 .....	150
Table 72: Crosstab of notation by metric for metric 8.....	151
Table 73: Chi-Square tests for notation by metric for metric 8 .....	152
Table 74: Crosstab of notation by metric for metric 9.....	153
Table 75: Chi-Square tests for notation by metric for metric 9 .....	154
Table 76: Crosstab of notation by metric for metric 10.....	155
Table 77: Chi-Square tests for notation by metric for metric 10.....	156
Table 78: Crosstab of notation by metric for metric 11 .....	157
Table 79: Chi-Square tests for notation by metric for metric 11 .....	158
Table 80: Tests for Normality for responses to all 11 metrics combined.....	159
Table 81: Non-Parametric tests (descriptive statistics) for Mann-Whitney test.....	161
Table 82: Mann-Whitney Test (non-parametric) Ranks .....	161
Table 83: Mann-Whitney Test (non-parametric) Test Statistics.....	161

Table 84: Mann-Whitney Test (non-parametric) Means and Medians..... 162

# 1 Introduction

Software systems are among the most complex systems ever built. This complexity stems from the size of software projects together with contemporary methods of writing software. Handling that complexity is one of the challenges when aiming to improve the quality of software products. A proposed solution is not to think in terms of the software system itself but in terms of a model of the system. A model is a representation that is simpler than the system but still contains the details that we consider essential. The purpose is to raise the level of abstraction and allow thinking in high-level concepts instead of technicalities [1].

## 1.1 Historical context

In the early days of software systems development, a developer or team of developers would devise their own methods of analysing the system. This was always driven by personal influences, and while it may have been good for the individual developer or development team, it did not always produce the optimal system design, both for initial development and maintenance. Poor communication of ideas, and a lack of rigour, led to errors and omissions.

Initial efforts to resolve the problems were aimed at the programming and implementation aspects of systems development. Structured programming techniques were developed to provide consistent communication of ideas for program code development that added a certain amount of rigour. They also went a small way to address the problem of software complexity. Programs written in a structured fashion were a lot easier for another developer to maintain, therefore complexity was reduced. It also meant that programming activities could be managed more effectively. In time, these structured methods addressed the initial phases of software projects, allowing the same good communication and rigour to prevail over the whole of the project, not just the final stages. Once again, complexity was reduced.

However, over time, software systems became even more complex, as larger and larger systems were developed. Once again, developers found their systems

becoming larger and more difficult to develop and maintain, therefore more complex. A new software development paradigm was born; Object-Orientation (OO). Object-oriented programming may be seen as a collection of cooperating *objects*, as opposed to a traditional structured view in which a program may be seen as a group of tasks whose goal it is to compute (i.e. subroutines). In object-oriented programming, each object is capable of receiving messages, processing data, and sending messages to other objects.

Each object can be viewed as an independent little machine with a distinct role or responsibility. The actions, or operators, on the objects are closely associated with the object. For example, in object-oriented programming, the data structures tend to carry their own operators around with them (or at least "inherit" them from a similar object or "class"). This is termed encapsulation. The traditional approach tends to view and consider data and behaviour separately. Individual program units now became smaller, and creating larger more manageable systems was a case of providing cohesive links between these smaller objects.

Once again, software development concepts had overtaken software analysis and design concepts. To address this shortcoming, object modelling techniques were developed.

The UML is the brainchild of three software engineers: Grady Booch, Jim Rumbaugh and Ivar Jacobson. Collectively they are known as the Three Amigos. Booch's "Booch Method" was better suited for object-oriented design, while Rumbaugh's "Object Modelling Technique" (OMT) was better suited for object-oriented analysis. Jacobson's "Object-Oriented Software Engineering" later was amalgamated with the efforts of Booch and Rumbaugh to create a unified method.

Booch, Rumbaugh, and Jacobson were motivated to create a unified modelling language as their own methods were already evolving toward each other independently. It made sense to continue that evolution together rather than apart. Also, by unifying the semantics and notation, they could bring some stability to the object-oriented marketplace, allowing projects to settle on one mature modelling language. They also expected that their collaboration would yield improvements in

all three earlier methods, helping them to capture lessons learned and to address problems that none of their methods previously handled well.

The efforts of Booch, Rumbaugh, and Jacobson resulted in the release of the UML in 1996. The UML authors invited and received feedback from the general community. They incorporated this feedback into the UML, but it was clear that additional focused attention was still required [2].

Concepts from many other OO methods were also loosely integrated with UML with the intent that UML would support all OO methods. As a result, UML is useful in a variety of engineering problems, from single process, single user applications to concurrent, distributed systems, making UML rich but large.

The Unified Modelling Language is now an international standard:

ISO/IEC 19501:2005 Information technology — Open Distributed  
Processing — Unified Modelling Language (UML) Version 1.4.2.

UML has matured significantly since UML 1.1. Several minor revisions (UML 1.3, 1.4, and 1.5) fixed shortcomings and bugs with the first version of UML, followed by the UML 2.0 major revision that was adopted by the OMG in 2005.

## 1.2 Related work

We will examine the most popular software modelling language of the day, the Unified Modelling Language, along with its additional tool, the Object Constraint Language, and give comparisons with a new modelling tool, Constraint Diagrams.

### 1.2.1 The Unified Modelling Language

The Unified Modelling Language (UML) is a set of visual languages for specifying, constructing, and documenting the artefacts of software-intensive systems. It is not a software development method in itself, but is designed to be compatible with a large number of object-oriented software development methods. UML defines a model as a set of interconnected model elements, which are abstractions drawn from the system. What the user normally sees is a UML diagram, which is a graphical view to the model and shows a subset of the model elements. Different diagrams provide

different perspectives to the model with various levels of accuracy. The diagrams are largely independent and complement each other to form a general view that can be easily understood. Modelling tools are responsible for ensuring that these views consistently represent the same model [1, 3, 4].

A model contains three major categories of elements: Classifiers, events, and behaviours. Each major category models individuals in an incarnation of the system being modelled. A classifier describes a set of objects; an object is an individual thing with a state and relationships to other objects. An event describes a set of possible occurrences; an occurrence is something that happens that has some consequence within the system. Behaviour describes a set of possible executions; an execution is the performance of an algorithm according to a set of rules. Models do not contain objects, occurrences, and executions, because those things are the subject of models, not their content. Classes, events, and behaviours model sets of objects, occurrences, and executions with similar properties. Value specifications, occurrence specifications, and execution specifications model individual objects, occurrences, and executions within a particular context. The distinction between objects and models of objects, for example, may appear subtle, but it is important. Objects (and occurrences and executions) are the domain of a model and, as such, are always complete, precise, and concrete i.e. they encapsulate all necessary identity, state and behaviour to describe the object; there are no syntactic or semantic ambiguity for either implementers or users; and that the model describes real world objects and events that implementers and users can relate to. Models of objects (such as value specifications) can be incomplete, imprecise, and abstract according to their purpose in the model [5].

As with any method of modelling, there are always issues of precision. A model will contain a number of diagrams that seek to accurately depict the system, either in part or as a whole. The main drawback is that within the framework of the UML, the full set of diagrams is not yet expressive enough to provide the required level of precision. The current version of the UML specification (UML 2.1.2 Superstructure and Infrastructure) attempts to improve on this lack of precision, but still does not solve the discrepancy fully.

### 1.2.1.1 Elements of the UML

The UML is a set of 14 visual languages describing the Model, Structure, Behaviour and various aspects of Interactions, along with one textual notation that models constraints (See OCL below).

Structure diagrams emphasize what things should comprise the model. A Class diagram describes the individual classes of object available in the model and defines both their attributes and their relationship with each other. Figure 1 is an example of a class diagram. A Component diagram depicts how a software system is divided into individual components and shows the inter-dependencies of these components. A Composite Structure diagram describes the internal structure of a class and the collaborations that this structure makes possible. A Deployment diagram serves to model the hardware used in system implementations, and the execution environments and artefacts deployed on the hardware. An Object diagram shows a complete or partial view of the structure of a modelled system at a specific time. A Package diagram depicts how a system is split up into logical groupings by showing the dependencies among these groupings [3, 4].

Behaviour diagrams emphasize what must happen in the lifecycle of the system being modelled and model concrete (i.e. real, not abstract) objects. An Activity diagram represents the business and operational step-by-step workflows of components in a system. A State diagram can describe the possible states of an object as events occur. A Use case diagram shows the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases [3, 4].

Interaction diagrams, a subset of behaviour diagrams, emphasize the flow of control and data among the things in the system being modelled. A Communication diagram shows the interactions between objects or parts in terms of sequenced messages. They represent a combination of information taken from Class, Sequence, and Use Case Diagrams describing both the static structure and dynamic behaviour of a system. An Interaction overview diagram is a type of activity diagram in which the nodes represent interaction diagrams. They are a high level structuring mechanism for Sequence diagrams. Interaction overview diagrams illustrate an



overview of a flow of control in which each node can be an interaction diagram. A Sequence diagram shows how objects communicate with each other in terms of a sequence of messages. They also indicate the lifespan of objects relative to those messages. The Collaboration diagram is almost exactly the same as the Sequence diagram. The difference is the perspective. Both diagrams model interactions between objects for a specific task, but while the Sequence diagram emphasizes the sequencing of interactions over time, the Collaboration diagram models how the interactions utilize the structure of the participating objects and their relationships. A Timing diagram is a specific type of interaction diagram, where the focus is on timing constraints [3, 4].

### **1.2.1.2 Improving precision of UML**

There have also been works to improve the precision of UML by extending the notations. Farhad writes in [6, p6]:

“The UML extension mechanisms provide not only a means for communication but also a framework for the knowledge and experiences of the individuals within a development culture such that the culture can evolve. They might not meet every need that arises within the development of a project, but they do accommodate a large portion of the tailoring and customising needed by most modellers in a simple manner that is easy to implement.”

Even with these extensions to the notation, UML as a whole is not nearly as precise as is needed to accurately model a system. Enter the Object Constraint Language!

### **1.2.2 The Object Constraint Language**

The Object Constraint Language (OCL) was developed to add the extra precision to UML models. It was originally developed in 1995 by IBM as part of a business modelling project. The main strength of OCL is that it is both formal and remains easy to read and write [7] i.e. its syntax is not based on mathematical expressions, as for example the specification languages Z, Larch and VDM++ are [8]. Modellers

without a strong mathematical background will find OCL appealing; it is designed for usability, although it is underpinned by mathematical set theory and logic [8].

After the Object Management Group's (OMG) standardisation of UML, it was found that a constraint could not be easily and precisely modelled. The OCL specification from the OMG [9, p19] states:

“A UML diagram, such as a class diagram, is typically not refined enough to provide all the relevant aspects of a specification. There is, among other things, a need to describe additional constraints about the objects in the model. Such constraints are often described in natural language. Practice has shown that this will always result in ambiguities. In order to write unambiguous constraints, so-called formal languages have been developed. The disadvantage of traditional formal languages is that they are usable to persons with a strong mathematical background, but difficult for the average business or system modeller to use.”

OCL was chosen by the OMG above other formal methods for the reasons already presented i.e. a software developer did not require a strong mathematical background, they found the notation easy to use, easy to learn and easy to understand. [8] Additionally, CASE tools were made available for the UML and OCL. OCL was adopted by the OMG as a standard for specifying constraints (invariants, pre- and post conditions, and other types of constraints) in 1997 [10].

OCL is a specification language and is therefore without the common side effects associated with programming languages (e.g. a statement cannot be executed, it cannot control flow within a block of statements, it will never change the state of the system, all implementation issues are out of scope and cannot be expressed). When an OCL expression is evaluated, it simply returns a value. It cannot change anything in the model, although an OCL expression can be used to *specify* a state change (e.g., in a post-condition).

OCL is a typed language so that each OCL expression has a type. To be well formed, an OCL expression must conform to the type conformance rules of the

language. For example, you cannot compare an Integer with a String. Each Classifier defined within a UML model represents a distinct OCL type. In addition, OCL includes a set of supplementary predefined types.

A Constraint is defined by Warmer and Kleppe [8, p1] as:

“... a restriction on one or more values of (part of) an object-oriented model or system.”

Constraints convey a number of benefits to a UML model, adding additional information about model elements and their relationships, and are therefore an excellent mechanism for enhancing documentation. They also provide a standard interpretation i.e. they are unambiguous: they are interpreted identically by different people (rather than the ambiguity of a natural language). Taking the full UML model and OCL constraints goes to provide a precise description of the model or system.

Models are used to communicate among users, designers, developers and others. To have a model accompanied by a natural language explanation of the constraints, or even an informal set of instructions that show structure but are based on natural language (e.g. pseudo-code) leaves the intent of the model open to differing interpretations. Using a formal constraint language like OCL the modeller can communicate to other interested parties a precise intent, without any misunderstanding.

A constraint can be defined as having “invariance, pre-condition and post-condition facets: conditions that must hold when the method is running, fires and terminates respectively” [8, p1, 11]. An Invariant is a rule that applies throughout the life of a data structure (in this case, a class) or procedure. Each change to the data structure (an object of that class) must maintain the correctness of the invariant. The principle behind the use of pre- and post-conditions of operations is often referred to as *design by contract* [12, 13].

### 1.2.3 Design by Contract

The definition of *contract* in the design by contract principle is derived from the legal notation of a contract: a univocal lawful agreement between two parties in which both parties accept obligations and on which both parties can found their rights [8]. In object-oriented terms, a contract is a means to establish the responsibilities of an object clearly and unambiguously. An object is responsible for executing services (obligations) if and only if certain stipulations (rights) are fulfilled. A contract is an exact specification of the interface of an object, and describes the services that are provided by that object. For each service, two things are described; the conditions under which the service will be provided (the pre-condition), and the specification of the result of the service that is provided, given that the conditions are fulfilled (the post-condition) [8].

### 1.2.4 An example of UML and OCL

To demonstrate the techniques of modelling with UML and OCL, we will use as an example the short case study of an application for a mortgage. Fig 1 presents the UML class diagram.

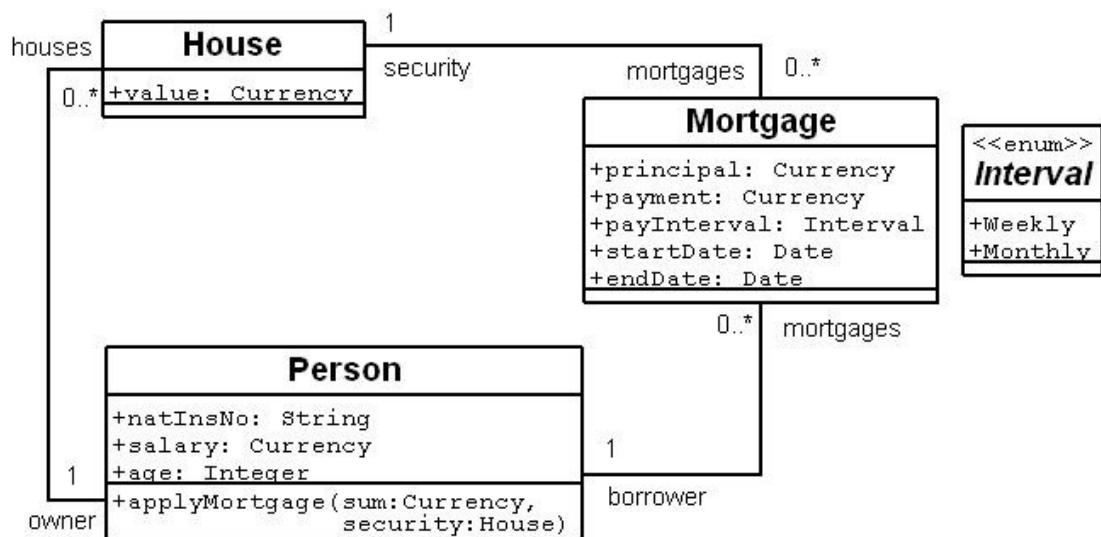


Figure 1: UML Class Diagram of a mortgage application

We can specify constraints against this model as both invariants and operations.

Example invariants and operations include:

```
Context Mortgage Inv:  
Self.security.owner = self.borrower  
Self.startDate < self.endDate
```

```
Context Person Inv:  
Self.age >= 18
```

```
Context  
Person::applyMortgage(sum:Currency, security:House)  
Pre:  
    Self.natInsNo.notEmpty and  
    Self.salary > 0  
Post:  
    Self.oclInState(Applied) and  
    Self.mortgages.principal = sum and  
    Self.houses = security
```

### 1.2.5 Some shortcomings of OCL

The OCL specification [10] describes the requirements of OCL as

1. OCL must be able to express extra (necessary) information on the models and other artefacts used in object-oriented development.
2. OCL must be a precise, unambiguous language that can easily be read and written by all practitioners of object technology and by their customers. This means that the language must be understood by people who are not mathematicians or computer scientists.
3. OCL must be a declarative language. Its expression can have no side-effects; that is, the state of a system must not change because of an OCL expression.
4. OCL must be a typed language.

According to Varizi and Jackson [14, p1]:

“We argue that, despite its numerous benefits, OCL is too implementation-oriented and therefore not suited to conceptual modelling. Moreover, it is at times unnecessarily verbose, yet far from natural language.”

A number of shortcomings are identified within the paper, especially the main fact of OCL appearing to be closer to an implementation language than a conceptual modelling language. This is argued due to the use of operations in constraints and the fact that OCL uses a type system not unlike an object-oriented programming language.

Operations have certain qualities that are incompatible with a conceptual modelling language. They can go into an infinite loop or may be undefined. In these cases, the expression containing the operation is said to be undefined. This raises concerns about precision. If we are modelling a system and cannot afford ambiguity, how can the model satisfy an undefined constraint? How can we identify in the model that the constraint is undefined? Also, if we consider a constraint to be applied to a collection of objects that are instances of a class, then it is possible that a subset of these objects may be instances of subclasses of the original class. These subclasses may redefine the original operation. It then becomes unclear as to which operation is applied to each object in the collection. Furthermore, if we assume that we only apply the redefined operation to objects which are instances of the subclass, this implies that the meaning of the constraint may change as the model evolves and further subclasses are added; very undesirable.

Types in OCL appear overly complicated. Conceptually, a class is a set of objects and a subclass is a subset of these objects. If a class were to inherit from two or more classes, then the classes must be non-disjoint sets. In OCL it is possible to have a class inherit from seemingly disjoint classes. Also, classes are not treated as collections of objects, as such we cannot use set operators to manipulate them directly, which increases the use of quantifiers. This makes frequent coercions (e.g. `oclIsKindOf`) necessary.

OCL expressions can appear somewhat verbose at times. For example, OCL uses two notations for navigating sets (->) and scalars (.). The lack of uniformity adds unnecessary complexity, especially as it can be argued that a scalar value can be a set with one element i.e. the scalar value.

One of the main shortcomings, though, is that OCL is not a stand-alone language. A model presented just in OCL would be meaningless. It must always be accompanied by its UML class diagram. There are various advantages to having a stand-alone constraint language, including (but not restricted to): [14]

1. The choice of expressing models textually or graphically can be made more flexibly
2. There is better integration between modelling and constraint languages
3. The constraint language's semantics are easier to define
4. The constraint language is more amenable to automated analysis

### **1.2.6 Constraint Diagrams**

The rationale behind OCL and Constraint Diagram notations was to construct a developer-friendly notation for expressing constraints in object-oriented modelling, as an alternative to traditional mathematical syntax [15]. Where OCL was developed to be a textual notation without mathematical symbols, Constraint Diagrams are a graphical notation based on Venn-Euler diagrams and Spider diagrams. A Spider diagram extends the Venn-Euler diagram notation by adding existential points and joins between them.

### 1.2.7 Euler Circles, Venn Diagrams and Spider Diagrams

In their paper “Towards a Formalization of Constraint Diagrams” [16, pp1-2], Gil, Howse and Kent provide a short history of Euler Circles, Venn Diagrams and Spider Diagrams.

“Constraint diagrams build on a long history of using diagrams to visualize logical or set-theoretical assertions. Circles or closed curves, which we call contours, have been in use for the representation of classical syllogisms since at least the Middle Ages [17]. Euler introduced the notation we now call *Euler circles* (or *Euler diagrams*) [18] to illustrate relations between sets. This notation uses the topological properties of enclosure, exclusion and intersection to represent the set-theoretic notions of subset, disjointedness, and intersection, respectively.

The logician John Venn used contours to represent logical propositions [19]. In Venn diagrams all contours must intersect. Moreover, for each non-empty subset of the contours, there must be a connected region of the diagram, such that the contours in this subset intersect at exactly that region. Shading is then used to show that a particular region of the resulting map is empty.

The logician Charles Peirce augmented Venn diagrams by adding X-sequences as a means for denoting elements [20]. An X-Sequence connecting a number of “minimal regions” of a Venn diagram indicates that their union is not empty. Full semantics and inference rules have only recently been developed for Venn-Peirce diagrams [21] and Euler circles [22]. Constraint diagrams bear a resemblance to Harel’s *higraphs* [23], the basis of *state charts*, in that they both represent binary relations by using arrows between closed curves. However, the semantics of the two notations are rather different.

*Spider diagrams* [24] are a natural extension of Venn-Peirce and Euler diagrams; they are based on Euler diagrams, so the topological properties of



the diagrams are important, but they also contain *spiders*, a generalization of Peirce's X-sequences, and shading.”

Gil, Howse and Kent state in their paper “Formalizing Spider Diagrams” [24, p1]:

“As a means of drawing constraints on sets and their relationship with other sets, Venn diagrams are expressive, but complicated to draw because all possible intersections have to be drawn and then some regions shaded. [...] On the other hand, Euler diagrams are intuitively easier to draw, but are not as expressive as Venn diagrams because they lack provisions for shading and for “X-Sequences”.”

A combination of the two notations indicated above would provide a far easier single notation to interpret, along with further extensions to improve the expressiveness i.e. spiders. Figure 1-2 below shows an example of a spider diagram, along with its set theoretical specification.



**Figure 2: an example Spider Diagram with description, taken from [24]**

Spider diagrams have emerged from a succession of attempts to provide the software engineer with precise, yet intuitive tools to specify a system, but are still the realm of mathematicians, and do not lend well in this current format to modelling software systems. Further evolution has occurred to provide the notation now known as Constraint Diagrams.

### **1.2.8 Elements of Constraint Diagrams**

Constraint Diagrams are a diagrammatic notation for precisely modelling constraints on an object-oriented model or system. The constraint diagram notation is a formal language that visualises first order logic. The notation is based on Euler diagrams,

which are extended by adding graphs to represent elements and arrows to represent functions and relations, thus providing a rich notation for modelling real world problems.

### 1.2.9 Contours

A contour is a simple closed plane curve. A boundary rectangle contains all other contours, although in concrete spider or constraint diagrams the existence of such a boundary rectangle is implied, and may be omitted. In Constraint Diagrams, contours denote sets of objects within the object-oriented model or system.

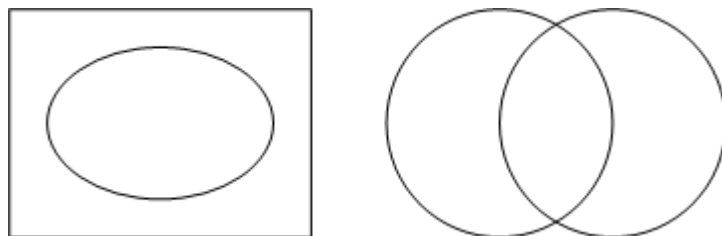
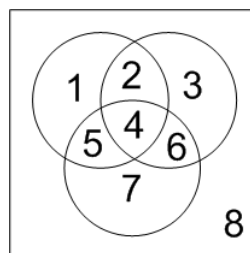


Figure 3: example contours

### 1.2.10 Districts, Regions and Zones

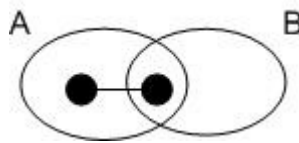
A district (or basic region) is the bounded area of the plane enclosed by a contour or by the boundary rectangle. A region is defined, recursively, as follows: any district is a region; if  $r1$  and  $r2$  are regions, then the union, intersection, or difference, of  $r1$  and  $r2$  are regions provided these are non-empty. A zone (or minimal region) is a region having no other region contained within it. Contours and regions denote sets [16]. In fig. 4, the zones are numbered. Any combination of the zones that reside next to each other represent a region e.g. zones 1 and 2 combine to form a region, as do zones 4, 5, 6 and 7.



**Figure 4: map showing zones within contours**

### 1.2.11 Spiders

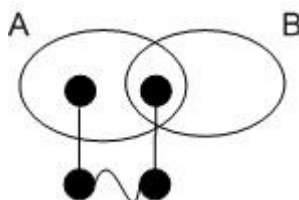
A spider is a tree with nodes (or feet) placed in different zones, which may be connected by straight lines (or legs). If a spider places one of its feet in a zone, it is said to *touch* the zone. Any given spider may touch at most one zone. A spider *inhabits* a region which is the union of all the zones it touches.



**Figure 5: example of a spider**

There are three types of spider; universal, existential and constant (or given). The universal spider represents the total set of elements within a zone, and is denoted by an asterisk or star. Existential spiders indicate that elements of the set exist, but are either unknown or undefined e.g. of the set of integers we know that at least one value exists due to the presence of an existential spider, but we are unaware of the exact integer number in question. Existential spiders are represented by a circle. The constant spider is represented by a square, and denotes the existence of a known value e.g. of our set of integers the constant spider would represent a given integer, say 1, 294 or 83,912.

Two distinct spiders denote distinct elements of the given set, unless they are connected by a *strand* or *tie*. A strand, represented by a wavy line connecting spiders in the same zone, denotes that the spiders *may* be the same element.



**Figure 6: example of a strand**

A tie, represented by a double straight line connecting spiders in the same zone, denotes that the spiders *must* be the same element. Strands and ties must be sourced and targeted at spiders.

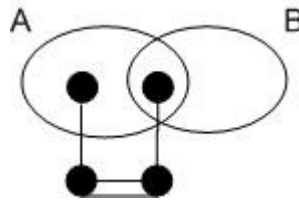


Figure 7: example of a tie

A relation or function is represented by an arrow. Arrows have a label, a source and a target. The source of an arrow may be a contour, zone or spider. The target of an arrow may be a contour, zone or spider. It is interpreted as the relational image of the set represented by the source under the relation represented by the arrow [16].

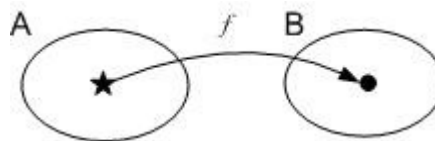


Figure 8: example of an arrow representing a relation or function

### 1.2.12 Shading

A zone that is un-shaded has zero or more elements. Shading within a zone that contains no spiders indicates that zone is empty i.e. has precisely zero elements (the empty set). Spiders existing within a shaded contour represent a quantifiable set, i.e. a shaded contour with only one spider represents that set as a singleton set consisting of one and only one element.



Figure 9: example of shading with single spider (singleton set)

### 1.2.13 An example of Constraint Diagrams

To demonstrate the techniques of modelling with constraint diagrams, we will use as an example the model for a library, specifically the invariant for number of books held by the library.

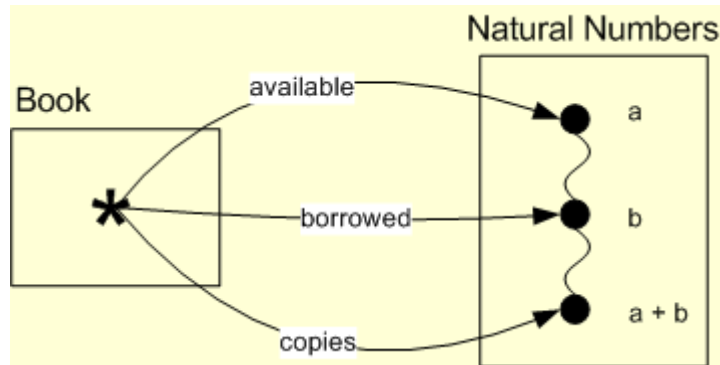


Figure 10: Constraint diagram showing the number of books held by a library

This constraint diagram is interpreted as “*The total number of books must equal the number of copies available plus the number of copies borrowed*”. The strand between each spider in the set of natural numbers indicates that the values for “a”, “b” and “a+b” may be equal. This would be the case where an instance of a book has zero copies available and zero copies borrowed.

### 1.3 Usability

The term “Usability”, when used in the context of computing, usually refers to the elegance and clarity with which a user interacts with a software application or website. This is termed “Human-Computer Interaction”, and the overall goal for improving a system’s usability is to improve this interaction. Usability is an abstract concept, but need not be vague. Researchers have reached a high degree of consensus on what usability means and on approaches to measuring it [25].

The primary notion of usability is that an object designed with the users' psychology and physiology in mind will perform better than objects designed without reference to the user, for example:

- More efficient to use – it takes less time to accomplish a particular task.
- Easier to learn – operation can be learned by observing the object.

- More satisfying to use.

Usability means that the people who use the application can do so quickly and easily to accomplish their own tasks [26]. This definition rests on four points:

1. Usability means focusing on users.
2. Users use applications to be productive.
3. Users are busy people trying to accomplish tasks.
4. Users decide when an application is easy to use.

To develop a usable software application you have to know, understand and work with the actual or potential users of the application. No one can substitute for them. Users consider an application easy to learn and use in terms of the time it takes to process each task, the number of steps each task may have and the success of predicting the required outcome.

Users connect usability with productivity. A user is seen to be more productive when they process more invoices, resolve more helpdesk queries or analyze more samples in a laboratory. Working at a computer terminal may be seen to be a barrier to productivity. Reducing the time a user spends at the computer may well improve productivity, or at least the notion of productivity.

A user will determine if the application is easy to use, not designers or developers. Users constantly balance time and effort while performing a task. They will decide if something is worth the benefit of spending their time using (or at least using correctly).

### **1.3.1 ISO Standards for Usability**

There are a number of ISO Standards that define and describe the term Usability in a multitude of contexts. From a computing viewpoint, the major standards are the ISO/IEC 9126 Software product evaluation – Quality characteristics and guidelines for their use (1991) and ISO/IEC 9241 Ergonomic requirements for office work with visual display terminals.

In the software engineering community the term usability was more narrowly associated with user interface design. ISO/IEC 9126, developed separately as a software engineering standard, defined usability as one relatively independent contribution to software quality associated with the design and evaluation of the user interface and interaction:

Usability: a set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users [27].

ISO/IEC 9126 (1991) has recently been replaced by a new four part standard that has reconciled the two approaches to usability. ISO/IEC 9126-1 describes the six categories of software quality that are relevant during product development: functionality, reliability, usability, efficiency, maintainability and portability, where:

Usability: the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions. The phrase "when used under specified conditions" was added to make it clear that a product has no intrinsic usability, only a capability to be used in a particular context.

The standard now recognises that usability plays two roles: a detailed software design activity (implied by the definition of usability), and an overall goal that the software meets user needs. ISO/IEC 9126-1 uses the term "quality in use" for this broad objective:

Quality in use: the capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use.

Quality in use is the combined effect of the six categories of software quality when the product is in use. The overall objective is to achieve quality in use, both for the end user and the support user. Functionality, reliability, efficiency and usability determine quality in use for an end user in a particular context. The support user is concerned with the quality in use of maintenance and portability tasks.

Other parts of ISO/IEC 9126 define metrics for usability and quality in use [27].

ISO/IEC 9241 provides requirements and recommendations relating to the attributes of the hardware, software and environment that contribute to usability, and the ergonomic principles underlying them. ISO/IEC 9241-11 provides the definition of usability that is used in subsequent related ergonomic standards:

“Usability: the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.” [28]

ISO 9241-11 explains how to identify the information that it is necessary to take into account when specifying or evaluating usability in terms of measures of user performance and satisfaction. Guidance is given on how to describe the context of use of the product and the measures of usability in an explicit way. It includes an explanation of how the usability of a product can be specified and evaluated as part of a quality system, for example one that conforms to ISO 9001.

It also explains how measures of user performance and satisfaction can be used to measure how any component of a work system affects the quality of the whole work system in use [27].

It is the ISO/IEC 9241-11 definition of Usability that this thesis uses to determine if Constraint Diagrams are usable in the practice of Software Specification.

### **1.3.2 Quantifying Usability**

While there are numerous methods of testing usability, when decomposed to their underlying quantitative processes, a number tend to quantify usability through some form of primitive statistics. For example, the GOMS<sup>1</sup> [29] model at its simplest counts the number of keystrokes a user performs over time to complete a task. There are other GOMS techniques to measure activity, but each decomposes a task to

---

<sup>1</sup> “Goals, Operators, Methods and Selection Rules: a model is composed of METHODS that are used to achieve specific GOALS. The METHODS are then composed of OPERATORS at the lowest level. The OPERATORS are specific steps that a user performs and are assigned a specific execution time. If a GOAL can be achieved by more than one METHOD, then SELECTION RULES are used to determine the proper METHOD.” [29]



elementary actions (such as keystrokes) that can be quantified. Each GOMS technique provides valuable information, but all have certain drawbacks. None of the techniques address user unpredictability, such as user behaviour being affected by fatigue, social surroundings, or organizational factors. The techniques are very explicit about basic movement operations, but are generally less rigid with basic cognitive actions. It is a fact that errors cannot be prevented, but none of the GOMS models allow for any type of error. Further, all of the techniques work under the assumption that a user will know what to do at any given point. This is only applicable to expert users; novices are not considered.

Functionality of the system is not considered, only the usability. If functionality were considered, the evaluation could make recommendations as to which functions should be performed by the system. User personalities and habits are not accounted for in any of the GOMS models; all users are assumed to be exactly the same [30-32].

Another method, Task analysis, is especially valuable in the context of human-computer interaction. User interfaces must be specified at an extremely low level (e.g. in terms of particular interaction styles and widgets), while still mapping effectively to users' high-level tasks. Computer interfaces are often highly inflexible (when compared to interacting with a physical environment or another person). This inflexibility magnifies the impact of interface design problems, making the close integration of task structure and interface support especially crucial [33].

Performing a task analysis verifies that the set of actions employed by the user does accomplish the task, and explicitly describes the procedure that the user actually employs since this may be different from the expected series of actions.

Task analysis is used to predict the time taken to learn a new task and become a proficient user of the particular application / machine. Task analysis may reveal how difficult one method is to learn compared to another. It can predict the time taken for a proficient user to accomplish the set task; this can reflect whether the interface is good at supporting exploration. Is it quicker to simply explore by trial and error or attempt to find out through help? It can also predict the time taken for expert

execution of the set task; how long does it take to become expert once a procedure has been discovered? This can be affected by the design of an interface [34].

A third method of analysing the usability of a software artefact is the Cognitive Walkthrough. Cognitive walkthrough builds on a task analysis that specifies the sequence of steps or actions required by a user to accomplish a task, and the system responses to those actions. The designers and developers of the software then perform a walkthrough of the steps as a group, asking themselves a set of questions at each step. Data is gathered during the walkthrough, and afterwards a report of potential issues is compiled. Finally the software is redesigned to address the issues identified.

The effectiveness of methods such as cognitive walkthroughs is hard to measure in applied settings, as there is very limited opportunity for controlled experiments while developing software. Typically measurements involve comparing the number of usability problems found by applying different methods. However, Gray & Salzman called into question the validity of those studies in their dramatic 1998 paper "Damaged Merchandise" [35], demonstrating how very difficult it is to measure the effectiveness of usability inspection methods. The consensus in the usability community is that cognitive walkthrough works well for a variety of settings and applications.

User testing is the mainstay method when it comes to finding usability problems. Nothing is more convincing than watching person after person encounter difficulties with the same part of a software or information system. The difficult areas that repeat themselves between multiple test participants reveal areas that should be studied and changed by the developers. User testing can often uncover very specific areas needing improvement, where task analysis and cognitive walkthrough often find more general areas needing improvement.

A trained observer conducts user testing often with the assistance of software developers. People who are representative of the target audience are asked to perform representative tasks with the software. The observer writes a user testing report listing the problems and offering recommendations based on their findings.

## 1.4 The Cognitive Dimensions of Notations Framework

Human-Computer Interaction (HCI) has developed over a number of years into its own discipline within Software Engineering. There is an active research community, and there are many usability techniques available for a Software Engineer to evaluate HCI. However, when we attempt to look at Software Engineering Notations from a “Usability” perspective, we find this is a new discipline with very little research. Indeed, the only framework for providing a usability evaluation is the Cognitive Dimensions of Notations [36] (henceforth “Cognitive Dimensions” or “the framework”).

Thomas Green describes the nature of Cognitive Dimensions very succinctly in the abstract for his paper [37].

“‘Cognitive dimensions’ are features of computer languages considered purely as information structures or notations. They therefore apply to many types of language—interactive or programming, high or low level, procedural or declarative, special purpose or general purpose. They are ‘cognitive’ dimensions because they control how (or whether) the preferred cognitive strategy for design-like tasks can be adopted: it has repeatedly been shown that users prefer opportunistic planning rather than any fixed strategy such as top-down development. The dimension analysis makes it easier to compare dissimilar interfaces or languages, and also helps to identify the relationship between support tools and programming languages: the support tools make it possible to use opportunistic planning with notations that would otherwise inhibit it.”

The framework is a tool for evaluating usability of systems, where a system is defined as the notation and the environment. In this context, ‘computer languages’ means both programming languages and their development environments, and modelling languages and their drawing tools.

Green and Blackwell describe Cognitive Dimensions in [38, p5] as:

“... a tool to aid non-HCI specialists in evaluating usability of information-based artefacts (summative evaluation). Since they are addressed to non-

specialists, they consciously aim for broad-brush treatment rather than lengthy, detailed analysis. Their checklist approach helps to ensure that serious problems are not overlooked.”

Cognitive Dimensions are meant to provide discussion tools, not deep analysis [36, 39]. They provide a broad brush approach to usability evaluation by non-HCI specialists. It has benefits over other, more detailed analytical techniques, such as being quick to learn and apply, is applicable at any stage of design and is comprehensible to non-specialists, but this approach does have drawbacks.

The Cognitive Dimensions framework provides a profile for a software tool or notation being evaluated. There are dangers that if the practitioner were not well versed in either Cognitive Dimensions or were not familiar with the deeper analytical methods, these profiles could become much the same for different evaluated tools. After all, many of the dimensions listed in the tutorial [39] relate mainly to software artefacts; and indeed, the majority of research using this framework has been on graphical software tools. Examples of Cognitive Dimensions studies on software engineering notations include a study on the Z formal notation [40] and a study on the comparison of sequence diagrams and collaboration diagrams from the UML [41].

[40] Describes a study of the Z notation using the software tool TranZit, “an editor that is part of a system for producing executable specifications from initial client requirements. It displays the Z specification in schema boxes, allowing free naming of all variables and has a syntax checking tool”. The examples of specifications used conform to object oriented methods i.e. an operation is specified, along with the operation’s pre-condition and post-condition. Eight out of the 14 dimensions were chosen as being more appropriate to novice users of both Z and TranZit.

[41] Describes a comparative study of sequence diagrams vs collaboration diagrams, both notations of the UML. Both notations are semantically equivalent, the key difference being the structure of data represented. Unlike the previous study, a paper-based questionnaire was used. All dimensions were used, but results from some were not very verbose, having recorded “OK” or left blank. It is interesting to

note that as part of the conclusions the authors of [41] state they used Cognitive Dimensions as a theoretical approach, rather than as an empirical approach.

The framework itself seems to provide a better evaluation of interactive tools, rather than non-interactive notations, even though “*non-interactive* artefacts, such as tables, graphs, music notation, programming languages, etc.” [39] are apparently supported. Indeed, the quote above mentions “a tool to aid non-HCI specialists”. Outside specialist paradigms such as speech, gesture or touch recognition, HCI assumes the use of a graphical user interface (GUI). This would be the interface between a software tool and the user. This would not be a suitable approach if a user (in this case, a software engineer) were presented with just a diagram. When presented with, for example, just a Constraint Diagram (an invariant in a given model), we find the framework as a whole does not apply. One diagram on a web page or a sheet of paper will need to be evaluated for usability, but the set of dimensions the tutorial describes seem not to provide any useful data. For example, the dimension Visibility and Juxtaposability asks if we can see all elements of the notation, and if we can see two or more components (i.e. diagrams) side by side. As we wish to evaluate only one diagram, and all elements within the diagram are visible, we can make assumptions about Visibility, but as we have no other diagram, or indeed, other notation to compare it with, Juxtaposability becomes an unsuitable property to evaluate.

Likewise, Hidden Dependencies becomes unsuitable for evaluating our diagram. Hidden Dependencies are relationships between elements of the notation such that one element depends on another, but the dependency is not readily observable. An example of this is a formula in a spreadsheet. The formula depends on a column(s) of values, and by looking at the structure of the formula we can determine the column(s) of values used. However, looking at the column(s) of values will not provide any hint of a formula. This would be of no value when evaluating a static diagram, as there are no hidden dependencies; this assumes an interactive tool that can change over time.

The dimension Viscosity will highlight the problem we face here best of all. The framework [39, p12] defines Viscosity as “Resistance to change: the cost of making small changes”. It further refines this definition by describing Repetition Viscosity

*(a single goal-related operation on the information structure requires an undue number of individual actions)* and *Knock-on Viscosity (one change entails further actions to restore consistency)*. This dimension, then, will provide a lot of useful information about the tool used to develop the diagram, but will not provide anything of value about a static diagram on a sheet of paper, or on a static web page.

It appears that the framework is a valuable tool for evaluating tools and notations together, especially as a lot of the dimensions covered in the tutorial do not work with just the notations themselves. However, there are a number of drawbacks to using Cognitive Dimensions in a “formal” usability setting.

The full framework is not sufficient for evaluating a notation in isolation. Most, but not all, of the dimensions are concerned with delivering a notation by means of a tool. There are a small number that are sufficient for evaluating just notations, notably Closeness of Mapping (closely representing any given domain), Consistency (similar semantics are expressed in similar syntactic forms), Role-Expressiveness (the purpose of a component is readily inferred) and Secondary Notation (extra information is carried by other means than the official syntax). While we can provide a small subset of discussion aids using these four dimensions, we cannot provide a full usability profile of the static notation.

The Cognitive Dimensions framework is an example of a heuristic evaluation. This is based on the idea that testing conforms to a predefined set of guidelines, or heuristics, and requires expert analysis to form conclusions [42]. This expert analysis tends to be subjective in its conclusions, based around the evaluator’s experiences using the tool/notation combination. For a full investigation into the usability of any artefact, whether it is a piece of software, a notation or a gadget, a study into the usability must also provide concrete data that can be interpreted using quantitative methods, as well as providing these qualitative methods.

While Cognitive Dimensions has some benefit to this program of research, it will not be sufficient to provide a full picture of the usability of constraint diagrams; it must be complemented with quantitative methods.

## 1.5 Empirical Study

Empirical studies are research studies that are based on evidence i.e. experimentation and observation, and are not based on just theory alone. They are grounded in the scientific method, with researchers typically concerned with issues such as reliability and validity [43].

To understand why empirical research was chosen for this thesis, and to understand some of the pragmatic issues around running empirical studies, we need to discuss the characteristics of experimental research. [44, p27] states “After a research hypothesis is identified, the design of an experiment consists of three components: treatments, units and assignment method [45]”. Treatments describe the different techniques, devices and procedures to compare; Units are the objects that the treatments are applied to (usually human subjects with specific characteristics e.g. age, gender and computing experience); Assignment methods describe the way in which the units are assigned different treatments [44].

“The power of experimental research lies in its ability to uncover causal relations.” [44, p44]. The complete randomization of the assignment of treatments to the experimental units is the main reason that experimental research achieves this goal [45].

This discussion will describe the processes undertaken to complete the program of empirical research, with emphasis on the choices made and reasons as to why the program of research completed the way it did.

The overall aim of the research project was to determine if the Constraint Diagram notation developed by the Visual Modelling group at the University of Brighton in conjunction with colleagues at the University of Kent could stand up to rigorous use by professional software engineers.

The original aim was to perform three studies, with study one and two investigating the various aspects of usability for Constraint Diagrams alone, and study three providing a comparison between Constraint Diagrams and OCL. Study one was to be an online questionnaire where a participant would be shown a series of ten diagrams, and would have to provide a correct answer from four possible candidate

answers. One answer would be correct, one would be incorrect, and two would be partially correct. The online questionnaire would be presented through a web browser, a common interface that most people would be familiar with. To accompany the online questionnaire, a paper based questionnaire based loosely around the Cognitive Dimensions of Notations was presented. This was a multiple choice questionnaire asking for the opinions of the participants on the notation used. It was paper based to allow the participants to relax with refreshments and complete at their leisure, rather than experience pressures of a second online questionnaire. Study one was piloted successfully and was presented to the group of participants (see chapter 3).

Study two was to have been a paper based exercise on developing constraint diagrams. The idea was to further test the interpretation of the notation, along with the ability of software engineers to draw accurate diagrams given specifications for constraints. Study three was to extend this even further and provide a comparison of development between constraint diagrams and OCL, another constraint modelling language that was very familiar (it being part of the UML standard).

Study two began by developing a paper based study guide for just constraint diagrams, showing each possible element to the notation and how these could be applied to worked examples. It was envisaged this would benefit both participants who had experience of the notation, as a refresher of their existing knowledge, and those who did not have any prior experience, as a primer for the notation. It would provide a base-line level of experience for each participant. A paper-based questionnaire was developed, to allow the participant to draw three invariants, three post-conditions of operations and three queries; a total of nine diagrams to develop in total. When ready, this was piloted at conferences and within the University of Brighton IT student body. Disappointingly, we had no interest from any potential participant.

Following on from this, it was decided to reduce the number of diagrams to one each of invariants, post-conditions of operations and queries. The cut-down study was again piloted at a conference held in Brighton, and within the University of Brighton IT student body. Again, no interest in partaking in the study was forthcoming.



Finally, a drastic decision was made, to amalgamate study two and three, and have one questionnaire to develop both constraint diagram and OCL notations (see chapter 7). The base-line primer was redesigned to incorporate equivalent information on the OCL notation. A series of six operations were specified using the constraint diagram notation, and the same constraints used to specify OCL statements. It was also decided not to present the new study to researchers at conferences, but to concentrate only on the students within the University of Brighton School of Computing, Mathematics and Information Sciences. This was piloted successfully, with participants being taken from two consecutive academic years.

## **1.6 The rest of this thesis**

The rest of this thesis describes the following:

Chapter 2 describes the method used for the study on interpretation of constraint diagrams. Chapters 3 and 4 discuss the quantitative analysis of the results of the first study, and the conclusions gained, while chapter 5 discusses the qualitative analysis of the results of the first study.

Chapter 6 describes the method used for the study on developing constraint diagrams. Chapters 7 and 8 discuss the quantitative analysis of the results of the second study, and the conclusions gained, while chapter 9 discusses the qualitative analysis of the results of the second study.

Chapter 10 provides conclusions about the thesis as a whole, and presents ideas for further research.

## **2 Study 1: Interpreting constraints using constraint diagrams**

The experiment examined the ability of software engineers to interpret and understand object-oriented software modelling diagrams, namely Constraint Diagrams.

The UML presents many diagrammatic notations for modelling within a system, e.g. Class Diagram, State Chart, etc. These are typically not precise enough to model all relevant aspects of a specification. We need to describe further constraints on the objects the model represents. For this, formal languages were developed. The disadvantage with these formal methods was their interpretation. To fully understand the model, the reader had to have a Mathematical background.

The OCL was developed to specify these constraints in a language that could be read more easily by non-mathematicians. It is expressive, being able to specify constraints in much the same way as the formal methods. It is a textual specification language (not a programming language).

While Constraint Diagrams can be used independently of the UML, and can be used solely to develop object-oriented models, they can complement the UML in much the same way that OCL does. Constraint Diagrams were developed to enhance the visualization of object structures and they build on Class Diagrams in UML. Just like OCL, Constraint Diagrams can precisely express constraints on object-oriented models.

### **2.1 Hypotheses**

- Software Engineers and Students with previous training in Constraint Diagrams could more accurately interpret Constraint Diagrams than Software Engineers and Students with no previous exposure to Constraint Diagrams.
- Software Engineers and Students with previous training in Constraint Diagrams could more rapidly interpret Constraint Diagrams than Software Engineers and Students with no previous exposure to Constraint Diagrams.

To test the hypotheses we have to look at the degree of accuracy, along with the time taken, for interpreting the diagrams. To be easy to use, a diagram must be interpreted correctly within a given period of time. For this study on invariants, that time was taken to be two minutes. We expect most participants, regardless of group, to interpret most of the expressions correctly.

If these hypotheses are proved true, any such designed system is likely to have a shorter lead-time from the point of view of implementing program code, with fewer errors made by code developers.

## **2.2 Study focus**

The focus of the study is to investigate the ability to interpret the Constraint Diagram notation for a range of expert users.

### **2.2.1 Aims**

- To measure the ability of Software Engineers to interpret a range of constraints, in this case Invariants, modelled using Constraint Diagrams.
- To compare the ability of the two study groups to interpret Constraint Diagrams empirically.
- To gather data on the attitudes and opinions to the uses of the notation relevant to the level of interpretation using elements of the cognitive dimensions framework [39] to test the hypotheses.

## **2.3 Method**

Two groups were identified for the purposes of study. Group one had some familiarity with the Constraint Diagrams notation, gained from a formal course of study on a recognised undergraduate or postgraduate qualification, or as users of the notation. Group two had some familiarity with both formal notations and/or diagrammatic notations, but have not had exposure to Constraint Diagrams.

### 2.3.1 Computer-based exercises

A set of questions and web pages were formulated for the presentation of Constraint Diagrams. A series of questions will be developed taking the form:

- An invariant taken from a hypothetical system represented in Constraint Diagram notation.
- A set of four multiple choice interpretations of the invariant. These interpretations shall have a “degree of correctness” i.e. one interpretation shall be fully correct, one shall be fully incorrect, and two shall be partially correct.

To present the questionnaire a series of web pages were developed representing invariants, their corresponding Constraint Diagrams and their respective solutions. These web pages recorded the important test criteria, namely the response to the question and the time it took for the response to be provided. Time was captured to identify the interval it takes to formulate an interpretation of an invariant, while the response was captured to identify the accuracy of the interpretation.

The web pages began with an introduction to the study, instructions for the use of the web pages (also available in PDF format for printing, to act as a “crib sheet”) (see Appendix I) and a dummy test to acclimatise the participants to the nature of the test. Each page of the test (dummy or actual) had an introduction page stating that the participant could have a break before continuing, or could even finish the test at this stage if they did not wish to continue. The page had a “Start” button/icon. When clicked, this would immediately display the question page and give the start timestamp. Upon choosing the relevant option, the page provided a finish timestamp. The difference in timestamps would be the time to interpret the diagram.

Web pages were chosen as the preferred method of presenting the questionnaire, because of their nature as a flexible rapid development and deployment medium. Using server-side scripting languages for this task (in this case PHP), we gain the computing power of some of the most popular programming languages e.g. C++ and Java. We also have the client-side browser and JavaScript readily available on most desktops, saving time on installing and configuring evaluation tools. Finally, with today’s heterogeneous desktops (Windows XP, Linux, Mac OSX, etc), writing tools

to perform the same task in the same way would require significant development and testing time; time that could be better spent on preparing the study.

### **2.3.2 Paper-based questionnaire**

As part of the study, an additional paper questionnaire was presented (see Appendix II). This consisted of multiple choice questions that reflected four dimensions from the Cognitive Dimensions of Notations framework [36, 39], namely Closeness of Mapping, Consistency, Role Expressiveness and Secondary Notation.

### **2.3.3 Closeness of mapping**

This dimension examines how close the diagrammatic representation maps to the domain being modelled. We are interested in how well the Constraint Diagrams represent our constraints. We will look at all given constraints and examine the corresponding Constraint Diagrams.

### **2.3.4 Consistency**

This dimension examines how similar semantics are expressed using similar syntactic forms. We are interested in how consistent the full range of Constraint Diagrams is when examining their syntax and semantics. We will look at the notation and examine all constraints.

### **2.3.5 Role expressiveness**

This dimension examines how well the purpose of a component, action or symbol is readily inferred. We are interested in how easily the role of a constraint can be picked up from the notation. We will look at a natural language constraint and examine the Constraint Diagrams.

### **2.3.6 Secondary notation**

This dimension examines how extra information can be presented other than in the official syntax of the notation. We are interested in annotations, maybe in diagrammatic form, maybe in textual form that can convey additional meaning to the diagram. We are also interested in whether these annotations can be used in place of

the notation to improve the interpretation of the diagram. We also investigate the use of colour as an aid to conveying additional meaning.

### **2.3.7 Overview document**

As an aid to preparing the participants for the study, a series of PowerPoint slides was developed to introduce the Constraint Diagram notations. The PowerPoint presentation presented the mechanics of the notation from the point of view of a Software Engineer who has not used either notation previously. This aided as a refresher for existing users of the notation, and served as an introduction to the notation for participants with no previous exposure. Examples of the notation were developed using a class diagram and a number of invariants that can be defined from this class diagram. The class diagram in question was taken from the Object Management Group publication The UML 2.0 OCL Specification, reproduced with permission under the terms of the licence.

### **2.3.8 Undertaking the “interpretation” study**

Two sample groups were studied, based on the relevant experience.

- Sample Group One: Participants who are familiar with the Constraint Diagram notation.
  - Participants with both a sound knowledge of Constraint Diagrams and the Set Theoretical and Logical underpinnings of the notation. These participants will come from the Visual Modelling Group at the University of Brighton.
  - Participants who have successfully completed a course at either levels 2 or 3 undergraduate or Masters level postgraduate on the topic of formal methods of modelling systems (to include ‘Z’ or similar, OCL and Constraint Diagrams). These will be the Software Engineers of tomorrow.
- Sample Group Two: Participants who are unfamiliar with the Constraint Diagram notation, but who have a familiarity with other textual, graphical or mathematical modelling languages.

- Participants with some experience of Systems Analysis using a recognised modelling framework e.g. UML/OCL, SSADM, Z, Larch, FOPL, Set Theory and Relational Algebra/Calculus, etc, but who do not have any experience of Constraint Diagrams, or their experience of Constraint Diagrams was on a course over two years ago and have not used this knowledge since in their professional or academic careers.

### **2.3.9 Reducing threats to validity**

Maturation and history effects are not evident within this study, as each participant will take part in this study only once. The effects of time on both the participant and the environment are of no consequence. There are, however, other threats that could significantly affect the results of the study, which we shall discuss further.

The experimental situation itself may threaten the validity of the study. It may be that participants do not enter into the spirit of the study, and make random selections on the test, rather than thinking about the questions posed and making informed choices. It may be that participants need to refer back to the PowerPoint guide (or PDF printout) throughout the study, to help understand certain diagrams. This will obviously impact on the time taken to interpret any given diagram. Problems of this nature are referred to as testing effects, and are one aspect of the more general issue of experimental reactivity. This refers to the fact that participants will often react to features of an experiment so that the process of making observations can change observations [46].

Instrumentation effects refer to the collection of data for the study. Who collects the data, where it is collected and what methods are used to collect the data may all have significant impact on the experiment. To help reduce instrumentation effects, where possible the same instruments will be used in collecting data, e.g. the study will take place in the same room for all participants, the same browser (and same version/build of browser) will be used, the same person will be present at all experimental stages, etc.

### **2.3.10 Briefing**

Each group will be briefed separately by the experimenter just prior to taking the test. The participants will be presented with a copy of the introductory PowerPoint slides and asked to review them. Following this, there will be a question and answer session to ensure that the preparatory material was understood. Each participant will be assigned a unique reference number for use in logging on to the online questionnaire.

Immediately before the test begins the confidentiality and voluntary nature of participation will be stressed. Any participant can drop out at any time with no repercussions. However, we will offer a small incentive (refreshments or other such).

### **2.3.11 Taking the test**

Each group of participants will be exposed to the web questionnaire within the Information Services training rooms for preference or at a pre-arranged location (e.g. work, home). If the test takes place outside the University at a pre-arranged location, the experimenter will be in attendance to provide the briefing (see above), and to see that all test protocols are adhered to. All participants will use the same web browser (Internet Explorer 6) with roughly the same build number to establish continuity of results.

The test will take a maximum of two hours. In reality it may take about an hour, but we will allow for extra time if needed. Upon completion of the questionnaire, each participant will be invited to take refreshments, when the paper questionnaire can be completed. When all participants have completed, a short de-briefing will be undertaken.

### **2.3.12 Debriefing**

Everyone will be thanked for their participation. A participant's right to confidentiality and that no personal information will be kept regarding the study will also be re-iterated.



### **3 Quantitative analysis of the interpretation of constraint diagrams**

Once completed, the results from the study will be examined using various statistical techniques. To discuss these techniques and the results we obtain from them, we will first look at the descriptive statistics.

Section 3.1 discusses the confidence interval (CI), “an estimated range of values for a population parameter. The size of the confidence interval is expressed as a percentage. If sampling is random, a 95% confidence interval has a 95% chance of containing the population parameter” [46].

Section 3.2 Descriptive Statistics below tabulates the responses to questions one through ten, tabulates them as correct, partial and incorrect responses and provides rudimentary statistical calculations on them. As well as the number of participants, range, minimum and maximum response number, we provide the mean responses along with standard error, the standard deviation and variance, as well as skewness – or lack of symmetry on the normally distributed curve (a non-zero value indicates a skewness where positive values indicate a left skewed distribution, and negative values indicate a right skewed distribution) – and kurtosis – or the extent to which the distribution departs from a normally distributed curve i.e. how pointed the shape of the curve of the distribution is.

Sections 3.3 to 3.12 discuss the individual questions and the responses given by the participants. Each question begins with a list of the four possibilities of constraint given on the web questionnaire along with the constraint diagram that the four possibilities refer to. Following on are the tables of statistics.

First we provide a crosstab of group i.e. unfamiliar and familiar with the constraint diagram notation, by response given for each question. For each group we have a count of the indicated responses. While there are four possible responses, if no participant has made a response against one possible response, this is not displayed. After the counts, the table lists the percentage within group i.e. the percentage count within unfamiliar or the percentage count within familiar that provided that

particular response; the percentage within the question presented i.e. the percentage count within the question split by the group unfamiliar or familiar; the percentage of the total i.e. the percentage count of both groups unfamiliar and familiar taken as a whole; and for the groups we list the standard residuals i.e. the residual is the difference between the observed values and the values predicted assuming no association of variables in the cross-tabulation; to achieve a standard residual we divide by the standard error of the residual i.e. the standard error is the standard deviation of a parameter estimate.

The second table lists the chi-square test results for the responses for the given question. Each table will give a value for the Pearson Chi-Square test, a statistical test for testing the null hypothesis that the distribution of a discrete random variable coincides with a given distribution; the Likelihood Ratio, aimed at testing a simple null hypothesis against a simple alternative hypothesis; Linear-by-Linear Association – a trend test to indicate if there is an underlying trend to the ordinal scale of the statistic, in this case, the given responses to the questions – and number of valid cases.

In the event of the Chi-Square tests having cells with an expected count less than five, we transform the responses to a table of correct and incorrect responses, ignoring the partially correct responses then perform the same tests on this data. When undertaking the Chi-Square tests for the transformed responses, we also see the statistic Continuity Correction. This is an adaptation of the Chi-Square test for small samples.

Section 3.13 then goes on to describe inferential statistics, i.e. those that can be used to make inferences beyond the sample given to the population as a whole, rather than those describing the sample population.

Section 3.14 describes the first of these inferential statistics, sample tests of normality to determine if the samples are normally distributed. We show the results of the Kolmogorov-Smirnov and Shapiro-Wilk tests to show normality. The Kolmogorov-Smirnov test is a conservative test and is sensitive to extreme values, so is less likely to provide a significant result. The Shapiro-Wilk test has more power to determine a non-normal distribution. This is the test we use in this thesis.

Section 3.15 describes the Mann-Whitney non-parametric test (for non-normally distributed samples). Non-parametric techniques are statistical methods that make no assumptions about the precise form of the frequency distribution from which the data are sampled. They are mainly of use for hypothesis testing using the information in the rank order within each sample. The Mann-Whitney test may be used to test whether two samples are drawn from the same distribution.

Section 3.16 describes regression analysis, which includes techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables. More specifically, regression analysis helps us understand how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are fixed. Most commonly, regression analysis estimates the conditional expectation of the dependent variable given the independent variables — that is, the average value of the dependent variable when the independent variables are fixed [47].

Sections 3.17 and 3.18 describe Univariate Analysis of Variance or ANOVA. The greater part of empirical research is concerned with the characteristics of groups, or aggregate social entities, rather than individual cases; that is, with men or women in general, rather than any particular man or woman. A range of statistical measures of association are employed to describe the features of groups, or types of case in the aggregate. Most of them are based on the normal distribution, the Binomial distribution, or the Poisson distribution, from which a number of statistical summary measures are derived. The mean, mode, and median provide measures of central tendency, the most common or typical value in the distribution, which coincide when the distribution is normal. Measures of dispersion attempt to concentrate information about the general pattern in single summary statistics. These include the range, mean deviation, quartile deviation, decile range, and the standard deviation, which is by far the most important. One of the properties of the normal distribution is that about 68 per cent of all cases are contained within one standard deviation on either side of the mean, about 95 per cent lie within two standard deviations, and some 99.73 per cent lie within three standard deviations on either side of the mean. The standard deviation of a distribution thus summarizes a great deal of information about the

overall dispersion, the degree of clustering or concentration around the mean. The variance is the square of the standard deviation, and has the property of being additive. Analysis of variance measures variance within sub-groups in the data-set, and between the averages of these sub-groups [48].

Finally, section 3.19 describes Kaplan-Meier Survival Analysis. Kaplan-Meier survival analysis is a technique that involves the generation of the tables and plots of survival or hazard function for event history data. Kaplan-Meier survival analysis is a descriptive procedure for the time to event variables in cases where time is the most prominent variable [49]. The term “survival” is a bit misleading; you can use survival curves to study times required to reach any well-defined end-point.

### 3.1 Confidence Interval

If we assume a range of 8 to 10 correct responses (80-100% of the responses) as being a successful outcome of the study, we have a total of 57 successful participants out of 68, that is, 83.8% of participants were successful. We can calculate a 95% CI (using the calculation defined in [50]) of 72.9% to 91.6%. We can then say that the true percentage of people achieving at least 80% correct answers is likely to be between 72.9% and 91.6%.

We can also calculate the CI for each individual metric. The percentages (95% confidence interval) of participants who obtained correct answers on the individual responses are given below.

Question	Number of correct answers	Number of participants	Percentage	CI (from)	CI (to)
1	54	68	79.4	67.9	88.3
2	64	68	94.1	85.6	98.4
3	65	68	95.6	87.6	99.1
4	60	68	88.2	78.1	94.8
5	66	68	97.1	89.8	99.6
6	63	68	92.6	83.7	97.6
7	63	68	92.6	83.7	97.6
8	56	68	82.4	71.2	90.5
9	64	68	94.1	82.6	98.4
10	35	68	51.5	39.0	63.8

**Table 1: Actual Percentage and Confidence Interval**

Generally, participants were successful in interpreting the invariants if at least eight responses were correct.

### 3.2 Summarising the results

To start, we perform descriptive statistics to describe the main features of a collection of data in quantitative terms. Descriptive statistics are distinguished from inductive statistics in that they aim to quantitatively summarize a data set, rather than being used to support statements about the population that the data are thought to represent.

Descriptive Statistics						
Group		N	Minimum	Maximum	Mean	Std. Deviation
Unfamiliar	Correct	31	5	10	8.39	1.308
	Partial	31	0	3	1.06	.892
	Incorrect	31	0	2	.55	.723
	Valid N (listwise)	31				
Familiar	Correct	37	5	10	8.92	1.187
	Partial	37	0	5	.76	1.090
	Incorrect	37	0	2	.32	.580
	Valid N (listwise)	37				

**Table 2: Descriptive Statistics for accuracy of responses to questions**

Table 2 shows that the total responses have a range five, with a minimum number of correct responses of five to a maximum of ten correct responses. A similar range of five is shown for partially correct, although the minimum is zero and the maximum is five partially correct responses. Finally, the incorrect responses demonstrate a range of two, with a minimum of zero and a maximum of two incorrect responses. The mean correct responses are far more than both partial and incorrect with a value of 8.68 (against 0.9 for partial and 0.43 for incorrect). This would seem to indicate that the majority of responses are either correct or partially correct, with only very few incorrect responses.

### 3.3 Question 1

Using the constraint diagram below, with the possible answers:

1. All copies are made from more than one original
2. All copies are made from one original
3. There is a copy made from one original
4. There is a copy made from a set of originals

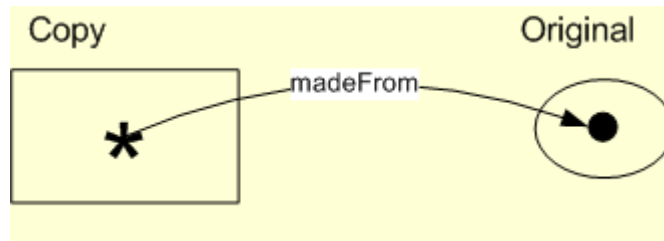


Figure 11: All copies are made from one original (option 2)

			Crosstab				
			Q1				
			1	2	3	4	Total
Group	Unfamiliar	Count	4	23	3	1	31
		% within Group	12.9%	74.2%	9.7%	3.2%	100.0%
		% within Q1	44.4%	42.6%	75.0%	100.0%	45.6%
		% of Total	5.9%	33.8%	4.4%	1.5%	45.6%
		Std. Residual	.0	-.3	.9	.8	
	Familiar	Count	5	31	1	0	37
		% within Group	13.5%	83.8%	2.7%	.0%	100.0%
		% within Q1	55.6%	57.4%	25.0%	.0%	54.4%
		% of Total	7.4%	45.6%	1.5%	.0%	54.4%
		Std. Residual	.0	.3	-.8	-.7	
Total	Count	9	54	4	1	68	
	% within Group	13.2%	79.4%	5.9%	1.5%	100.0%	
	% within Q1	100.0%	100.0%	100.0%	100.0%	100.0%	
	% of Total	13.2%	79.4%	5.9%	1.5%	100.0%	

Table 3: Crosstab of group by response for question 1

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	2.789 <sup>a</sup>	3	.425
Likelihood Ratio	3.204	3	.361
Linear-by-Linear Association	1.320	1	.251
N of Valid Cases	68		

a. 6 cells (75.0%) have expected count less than 5. The minimum expected count is .46.

**Table 4: Chi-Square Tests for responses for question 1**

We have tabulated the results for question 1 showing all four possible responses. As there are 6 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value. For this we will assume that “partially correct” results are, in fact, incorrect and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q1T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	8	23	31
		% within Group	25.8%	74.2%	100.0%
		% within Q1T	57.1%	42.6%	45.6%
		% of Total	11.8%	33.8%	45.6%
		Std. Residual	.6	-.3	
	Familiar	Count	6	31	37
		% within Group	16.2%	83.8%	100.0%
		% within Q1T	42.9%	57.4%	54.4%
		% of Total	8.8%	45.6%	54.4%
		Std. Residual	-.6	.3	
Total	Count	14	54	68	
	% within Group	20.6%	79.4%	100.0%	
	% within Q1T	100.0%	100.0%	100.0%	
	% of Total	20.6%	79.4%	100.0%	

**Table 5: Crosstab of group by transformed response for question 1**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.949 <sup>a</sup>	1	.330		
Continuity Correction <sup>b</sup>	.453	1	.501		
Likelihood Ratio	.946	1	.331		
Fisher's Exact Test				.378	.250
Linear-by-Linear Association	.935	1	.334		
N of Valid Cases	68				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 6.38.

b. Computed only for a 2x2 table

**Table 6: Chi-Square Tests for transformed responses for question 1**



Our transformed table now yields counts above the expected level of 5, thus we can use the Pearson Chi-Squared test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.330 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram. Additionally, the high number of incomplete but not incorrect answers may be caused by some of the participants showing apprehension at taking the test, or that this being the first question of the test, the participants were not fully aware of the significance of the test and made incomplete assumptions about the constraint diagram.

### 3.4 Question 2

Using the constraint diagram below, with the possible answers:

1. All persons not employed are under 18
2. All persons must be employed and be aged 18 or over
3. All employed persons must be aged 18 or over
4. All persons must be 18 or over

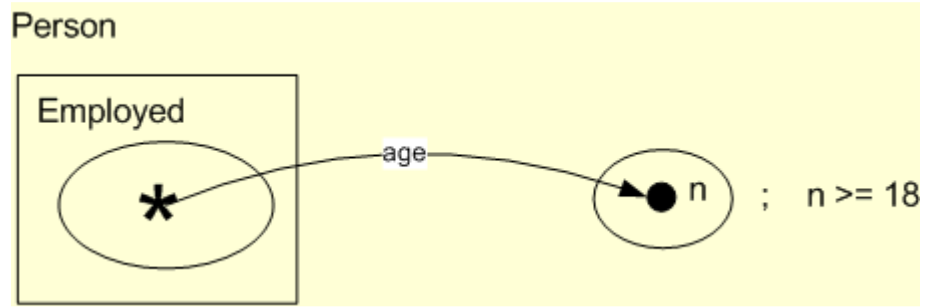


Figure 12: All employed persons must be aged 18 or over (option 3)

			Crosstab			
			Q2			
			1	2	3	Total
Group	Unfamiliar	Count	1	1	29	31
		% within Group	3.2%	3.2%	93.5%	100.0%
		% within Q2	100.0%	33.3%	45.3%	45.6%
		% of Total	1.5%	1.5%	42.6%	45.6%
		Std. Residual	.8	-.3	.0	
	Familiar	Count	0	2	35	37
		% within Group	.0%	5.4%	94.6%	100.0%
		% within Q2	.0%	66.7%	54.7%	54.4%
		% of Total	.0%	2.9%	51.5%	54.4%
		Std. Residual	-.7	.3	.0	
Total		Count	1	3	64	68
		% within Group	1.5%	4.4%	94.1%	100.0%
		% within Q2	100.0%	100.0%	100.0%	100.0%
		% of Total	1.5%	4.4%	94.1%	100.0%

Table 7: Crosstab of group by response for question 2

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.377 <sup>a</sup>	2	.502
Likelihood Ratio	1.759	2	.415
Linear-by-Linear Association	.311	1	.577
N of Valid Cases	68		

a. 4 cells (66.7%) have expected count less than 5. The minimum expected count is .46.

**Table 8: Chi-Square Tests for question 2**

We have tabulated the results for question 2 showing the given three out of a possible four responses. As there are 4 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value. For this we will assume that “partially correct” results are, in fact, incorrect and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q2T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	2	29	31
		% within Group	6.5%	93.5%	100.0%
		% within Q2T	50.0%	45.3%	45.6%
		% of Total	2.9%	42.6%	45.6%
		Std. Residual	.1	.0	
	Familiar	Count	2	35	37
		% within Group	5.4%	94.6%	100.0%
		% within Q2T	50.0%	54.7%	54.4%
		% of Total	2.9%	51.5%	54.4%
		Std. Residual	-.1	.0	
Total	Count	4	64	68	
	% within Group	5.9%	94.1%	100.0%	
	% within Q2T	100.0%	100.0%	100.0%	
	% of Total	5.9%	94.1%	100.0%	

**Table 9: Crosstab of group by transformed response for question 2**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.033 <sup>a</sup>	1	.855		
Continuity Correction <sup>b</sup>	.000	1	1.000		
Likelihood Ratio	.033	1	.855		
Fisher's Exact Test				1.000	.623
Linear-by-Linear Association	.033	1	.856		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1.82.

b. Computed only for a 2x2 table

**Table 10: Chi-Square Tests for transformed responses for question 2**

Our transformed table still does not yield counts above the expected level of 5, thus we cannot use the Pearson Chi-Squared test. Instead, we must use the Fisher's Exact Test (1-sided) to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.623 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.5 Question 3

Using the constraint diagram below, with the possible answers:

1. A person must not be unmarried
2. A divorced person must be both married and unmarried
3. A person must be either married or unmarried but not both
4. A person may be both married and unmarried

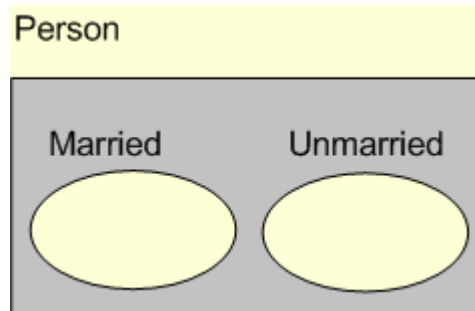


Figure 13: A person must be either married or unmarried but not both (option 3)

Crosstab					
			Q3		Total
			3	4	
Group	Unfamiliar	Count	29	2	31
		% within Group	93.5%	6.5%	100.0%
		% within Q3	44.6%	66.7%	45.6%
		% of Total	42.6%	2.9%	45.6%
		Std. Residual	-.1	.5	
	Familiar	Count	36	1	37
		% within Group	97.3%	2.7%	100.0%
		% within Q3	55.4%	33.3%	54.4%
		% of Total	52.9%	1.5%	54.4%
		Std. Residual	.1	-.5	
Total		Count	65	3	68
		% within Group	95.6%	4.4%	100.0%
		% within Q3	100.0%	100.0%	100.0%
		% of Total	95.6%	4.4%	100.0%

Table 11: Crosstab of group by response for question 3

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.562 <sup>a</sup>	1	.453		
Continuity Correction <sup>b</sup>	.025	1	.875		
Likelihood Ratio	.565	1	.452		
Fisher's Exact Test				.588	.433
Linear-by-Linear Association	.554	1	.457		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1.37.

b. Computed only for a 2x2 table

**Table 12: Chi-Square Tests for question 3**

We have tabulated the results for question 3 showing the given two out of a possible four responses. As there are 2 cells with expected count less than 5, and only two of the possible four responses were provided by the participants, we need not transform the table (it is in effect already transformed) and can use the Fisher's Exact Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

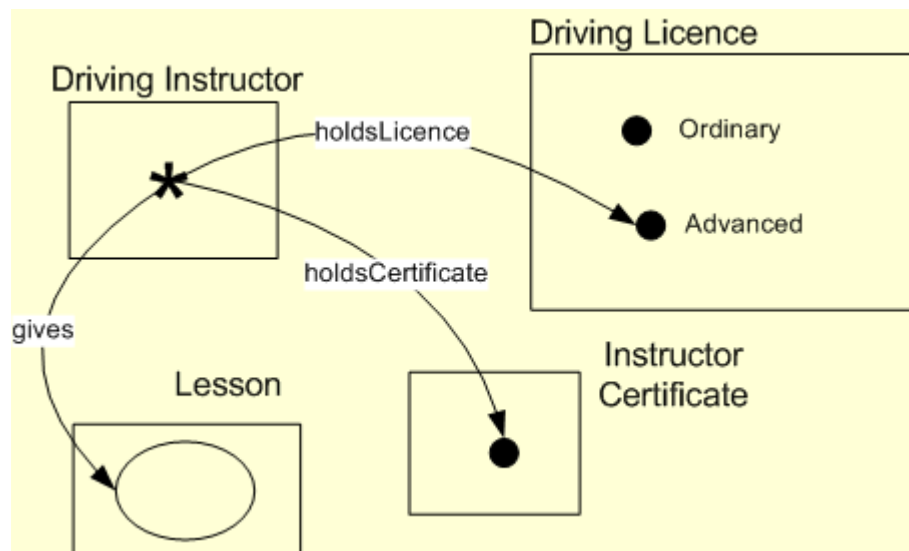
P-Value: 0.433 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.6 Question 4

Using the constraint diagram below, with the possible answers:

1. All driving instructors hold an Advanced Driving Licence and an Instructor Certificate and give driving lessons
2. All driving instructors hold a Driving Licence but not an Instructor Certificate and give driving lessons
3. A driving instructor holds an Advanced Driving Licence and an Instructor Certificate but never gives driving lessons
4. A driving instructor holds an Ordinary Driving Licence and an Instructor Certificate and gives driving lessons



**Figure 14: All driving instructors hold an Advanced Driving Licence and an Instructor Certificate and give driving lessons (option 1).**



			Q4				Total
			1	2	3	4	
Group	Unfamiliar	Count	25	1	2	3	31
		% within Group	80.6%	3.2%	6.5%	9.7%	100.0%
		% within Q4	41.7%	100.0%	50.0%	100.0%	45.6%
		% of Total	36.8%	1.5%	2.9%	4.4%	45.6%
		Std. Residual	-.4	.8	.1	1.4	
	Familiar	Count	35	0	2	0	37
		% within Group	94.6%	.0%	5.4%	.0%	100.0%
		% within Q4	58.3%	.0%	50.0%	.0%	54.4%
		% of Total	51.5%	.0%	2.9%	.0%	54.4%
		Std. Residual	.4	-.7	-.1	-1.3	
Total		Count	60	1	4	3	68
		% within Group	88.2%	1.5%	5.9%	4.4%	100.0%
		% within Q4	100.0%	100.0%	100.0%	100.0%	100.0%
		% of Total	88.2%	1.5%	5.9%	4.4%	100.0%

**Table 13: Crosstab of group by response for question 4**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	5.178 <sup>a</sup>	3	.159
Likelihood Ratio	6.690	3	.082
Linear-by-Linear Association	3.399	1	.065
N of Valid Cases	68		

a. 6 cells (75.0%) have expected count less than 5. The minimum expected count is .46.

**Table 14: Chi-Square Tests for question 4**

We have tabulated the results for question 4 showing all four responses. As there are 6 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value.

For this we will assume that “partially correct” results are, in fact, incorrect and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Group \* Q4T Crosstabulation**

			Q4T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	6	25	31
		% within Group	19.4%	80.6%	100.0%
		% within Q4T	75.0%	41.7%	45.6%
		% of Total	8.8%	36.8%	45.6%
	Familiar	Count	2	35	37
		% within Group	5.4%	94.6%	100.0%
		% within Q4T	25.0%	58.3%	54.4%
		% of Total	2.9%	51.5%	54.4%
Total		Count	8	60	68
		% within Group	11.8%	88.2%	100.0%
		% within Q4T	100.0%	100.0%	100.0%
		% of Total	11.8%	88.2%	100.0%

**Table 15: Crosstab of group by transformed response for question 4**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	3.162 <sup>a</sup>	1	.075		
Continuity Correction <sup>b</sup>	1.961	1	.161		
Likelihood Ratio	3.237	1	.072		
Fisher's Exact Test				.129	.081
Linear-by-Linear Association	3.115	1	.078		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 3.65.

b. Computed only for a 2x2 table

**Table 16: Chi-Square Tests for transformed responses for question 4**

Our transformed table still does not yield counts above the expected level of 5, thus we cannot use the Pearson Chi-Squared test. Instead, we must use the Fisher's Exact Test (1-sided) to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.129 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.7 Question 5

Using the constraint diagram below, with the possible answers:

1. All copies of a book are associated with a past loan
2. All copies of a book that are out must be associated with a past loan
3. All copies of a book are associated with a current loan
4. All copies of a book that are out must be associated with a current loan

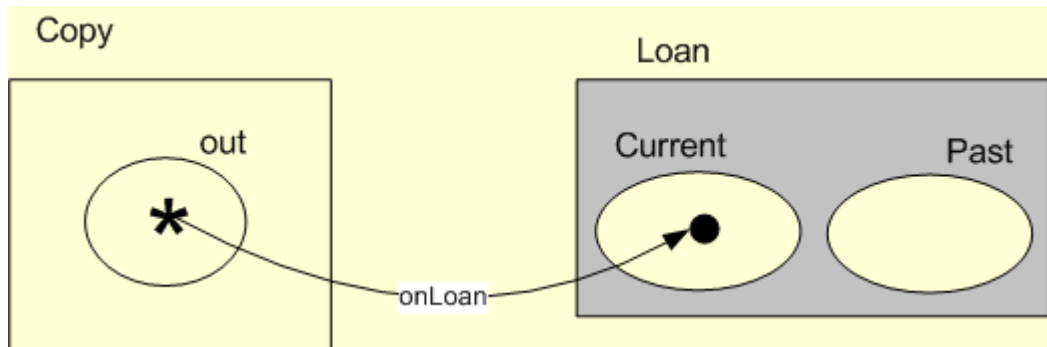


Figure 15: All copies of a book that are out must be associated with a current loan (option 4)

Crosstab					
			Q5		Total
			3	4	
Group	Unfamiliar	Count	1	30	31
		% within Group	3.2%	96.8%	100.0%
		% within Q5	50.0%	45.5%	45.6%
		% of Total	1.5%	44.1%	45.6%
		Std. Residual	.1	.0	
	Familiar	Count	1	36	37
		% within Group	2.7%	97.3%	100.0%
		% within Q5	50.0%	54.5%	54.4%
		% of Total	1.5%	52.9%	54.4%
		Std. Residual	.0	.0	
Total		Count	2	66	68
		% within Group	2.9%	97.1%	100.0%
		% within Q5	100.0%	100.0%	100.0%
		% of Total	2.9%	97.1%	100.0%

Table 17: Crosstab of group by response for question 5

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.016 <sup>a</sup>	1	.899		
Continuity Correction <sup>b</sup>	.000	1	1.000		
Likelihood Ratio	.016	1	.899		
Fisher's Exact Test				1.000	.708
Linear-by-Linear Association	.016	1	.900		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is .91.

b. Computed only for a 2x2 table

**Table 18: Chi-Square Tests for question 5**

We have tabulated the results for question 5 showing the given two out of a possible four responses. As there are 2 cells with expected count less than 5, and only two of the possible four responses were provided by the participants, we need not transform the table (it is in effect already transformed) and can use the Fisher's Exact Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

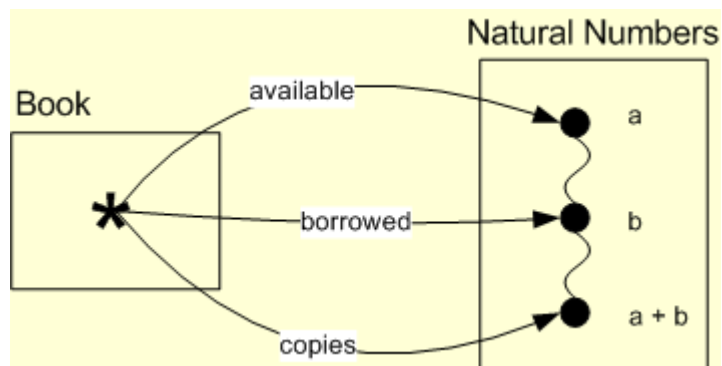
P-Value: 0.708 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.8 Question 6

Using the constraint diagram below, with the possible answers:

1. The total number of books will never equal the number of copies available plus the number of copies borrowed
2. The total number of books must equal the number of copies available plus the number of copies borrowed
3. The total number of books available will never be the same as the total number of books borrowed
4. The total number of books available will always be the same as the total number of books borrowed



**Figure 16: The total number of books must equal the number of copies available plus the number of copies borrowed (option 2)**

			Crosstab				Total
			Q6				
			1	2	3	4	
Group	Unfamiliar	Count	2	27	1	1	31
		% within Group	6.5%	87.1%	3.2%	3.2%	100.0%
		% within Q6	100.0%	42.9%	100.0%	50.0%	45.6%
		% of Total	2.9%	39.7%	1.5%	1.5%	45.6%
		Std. Residual	1.1	-.3	.8	.1	
	Familiar	Count	0	36	0	1	37
		% within Group	.0%	97.3%	.0%	2.7%	100.0%
		% within Q6	.0%	57.1%	.0%	50.0%	54.4%
		% of Total	.0%	52.9%	.0%	1.5%	54.4%
		Std. Residual	-1.0	.3	-.7	.0	
Total	Count		2	63	1	2	68
	% within Group		2.9%	92.6%	1.5%	2.9%	100.0%
	% within Q6		100.0%	100.0%	100.0%	100.0%	100.0%
	% of Total		2.9%	92.6%	1.5%	2.9%	100.0%

**Table 19: Crosstab of group by response for question 6**

Chi-Square Tests			
	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	3.786 <sup>a</sup>	3	.286
Likelihood Ratio	4.919	3	.178
Linear-by-Linear Association	.049	1	.824
N of Valid Cases	68		

a. 6 cells (75.0%) have expected count less than 5. The minimum expected count is .46.

**Table 20: Chi-Square Tests for question 6**

We have tabulated the results for question 6 showing all four possible responses. As there are 6 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded

value. For this we will assume that “partially correct” results are, in fact, incorrect and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q6T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	4	27	31
		% within Group	12.9%	87.1%	100.0%
		% within Q6T	80.0%	42.9%	45.6%
		% of Total	5.9%	39.7%	45.6%
		Std. Residual	1.1	-.3	
	Familiar	Count	1	36	37
		% within Group	2.7%	97.3%	100.0%
		% within Q6T	20.0%	57.1%	54.4%
		% of Total	1.5%	52.9%	54.4%
		Std. Residual	-1.0	.3	
Total	Count	5	63	68	
	% within Group	7.4%	92.6%	100.0%	
	% within Q6T	100.0%	100.0%	100.0%	
	% of Total	7.4%	92.6%	100.0%	

**Table 21: Crosstab of group by transformed response for question 6**



**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	2.576 <sup>a</sup>	1	.108		
Continuity Correction <sup>b</sup>	1.297	1	.255		
Likelihood Ratio	2.687	1	.101		
Fisher's Exact Test				.170	.128
Linear-by-Linear Association	2.538	1	.111		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 2.28.

b. Computed only for a 2x2 table

**Table 22: Chi-Square Tests for transformed responses for question 6**

Our transformed table still does not yield counts above the expected level of 5, thus we cannot use the Pearson Chi-Squared test. Instead, we must use the Fisher's Exact Test (1-sided) to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.128 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.9 Question 7

Using the constraint diagram below, with the possible answers:

1. All copies of books are held against reservations
2. Copies of books that are on hold are held against reservations
3. All copies of books that are not on hold are held against reservations
4. No copies of books are held against reservations

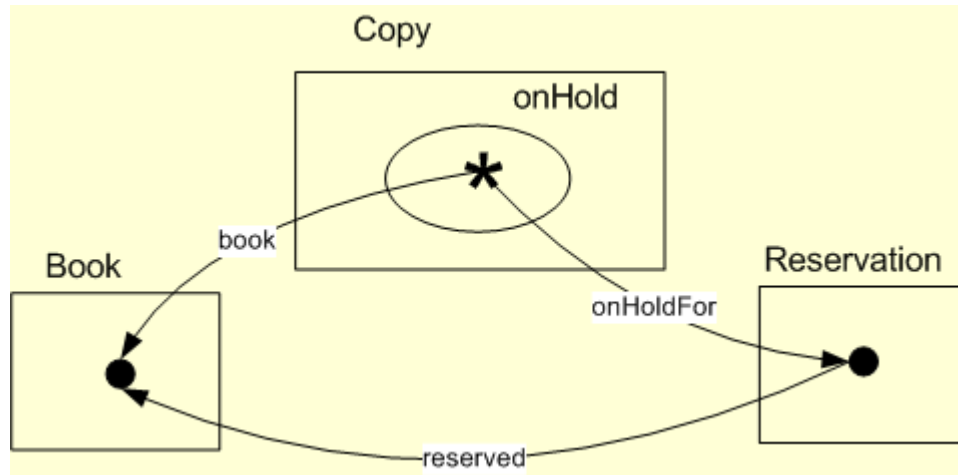


Figure 17: Copies of books that are on hold are held against reservations (option 2)

			Crosstab				Total
			Q7				
			1	2	3	4	
Group	Unfamiliar	Count	1	28	1	1	31
		% within Group	3.2%	90.3%	3.2%	3.2%	100.0%
		% within Q7	100.0%	44.4%	50.0%	50.0%	45.6%
		% of Total	1.5%	41.2%	1.5%	1.5%	45.6%
		Std. Residual	.8	-.1	.1	.1	
Familiar	Familiar	Count	0	35	1	1	37
		% within Group	.0%	94.6%	2.7%	2.7%	100.0%
		% within Q7	.0%	55.6%	50.0%	50.0%	54.4%
		% of Total	.0%	51.5%	1.5%	1.5%	54.4%
		Std. Residual	-.7	.1	.0	.0	
Total	Total	Count	1	63	2	2	68
		% within Group	1.5%	92.6%	2.9%	2.9%	100.0%
		% within Q7	100.0%	100.0%	100.0%	100.0%	100.0%
		% of Total	1.5%	92.6%	2.9%	2.9%	100.0%

**Table 23: Crosstab of group by response for question 7**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.258 <sup>a</sup>	3	.739
Likelihood Ratio	1.636	3	.651
Linear-by-Linear Association	.029	1	.864
N of Valid Cases	68		

a. 6 cells (75.0%) have expected count less than 5. The minimum expected count is .46.

**Table 24: Chi-Square Tests for question 7**

We have tabulated the results for question 7 showing all four possible responses. As there are 6 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value. For this we will assume that “partially correct” results are, in fact, incorrect

and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q7T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	3	28	31
		% within Group	9.7%	90.3%	100.0%
		% within Q7T	60.0%	44.4%	45.6%
		% of Total	4.4%	41.2%	45.6%
		Std. Residual	.5	-.1	
	Familiar	Count	2	35	37
		% within Group	5.4%	94.6%	100.0%
		% within Q7T	40.0%	55.6%	54.4%
		% of Total	2.9%	51.5%	54.4%
		Std. Residual	-.4	.1	
Total		Count	5	63	68
		% within Group	7.4%	92.6%	100.0%
		% within Q7T	100.0%	100.0%	100.0%
		% of Total	7.4%	92.6%	100.0%

**Table 25: Crosstab of group by transformed response for question 7**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.452 <sup>a</sup>	1	.501		
Continuity Correction <sup>b</sup>	.042	1	.837		
Likelihood Ratio	.451	1	.502		
Fisher's Exact Test				.653	.415
Linear-by-Linear Association	.445	1	.505		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 2.28.

b. Computed only for a 2x2 table

**Table 26: Chi-Square Tests for transformed responses for question 7**

Our transformed table still does not yield counts above the expected level of 5, thus we cannot use the Pearson Chi-Squared test. Instead, we must use the Fisher's Exact Test (1-sided) to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.415 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.10 Question 8

Using the constraint diagram below, with the possible answers:

1. A married adult is married to one spouse who is not a sibling
2. A person is married to more than one spouse
3. A person is never married to one spouse
4. A person is married to one spouse who is also a sibling

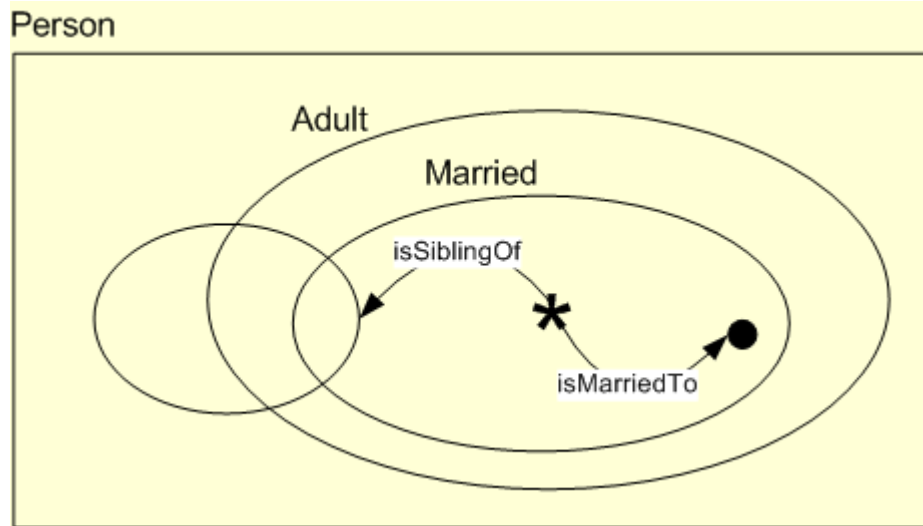


Figure 18: A married adult is married to one spouse who is not a sibling (option 1)

**Crosstab**

			Q8		Total
			1	4	
Group	Unfamiliar	Count	25	6	31
		% within Group	80.6%	19.4%	100.0%
		% within Q8	44.6%	50.0%	45.6%
		% of Total	36.8%	8.8%	45.6%
		Std. Residual	-.1	.2	
	Familiar	Count	31	6	37
		% within Group	83.8%	16.2%	100.0%
		% within Q8	55.4%	50.0%	54.4%
		% of Total	45.6%	8.8%	54.4%
		Std. Residual	.1	-.2	
Total	Count	56	12	68	
	% within Group	82.4%	17.6%	100.0%	
	% within Q8	100.0%	100.0%	100.0%	
	% of Total	82.4%	17.6%	100.0%	

**Table 27: Crosstab of group by response for question 8**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.114 <sup>a</sup>	1	.735		
Continuity Correction <sup>b</sup>	.000	1	.985		
Likelihood Ratio	.114	1	.736		
Fisher's Exact Test				.760	.490
Linear-by-Linear Association	.113	1	.737		
N of Valid Cases	68				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.47.

b. Computed only for a 2x2 table

**Table 28: Chi-Square Tests for question 8**

We have tabulated the results for question 8 showing the given two out of a possible four responses. As there are no cells with expected count less than 5, we need not transform the table (it is in effect already transformed) and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.735 (greater than 0.05 or 5%)

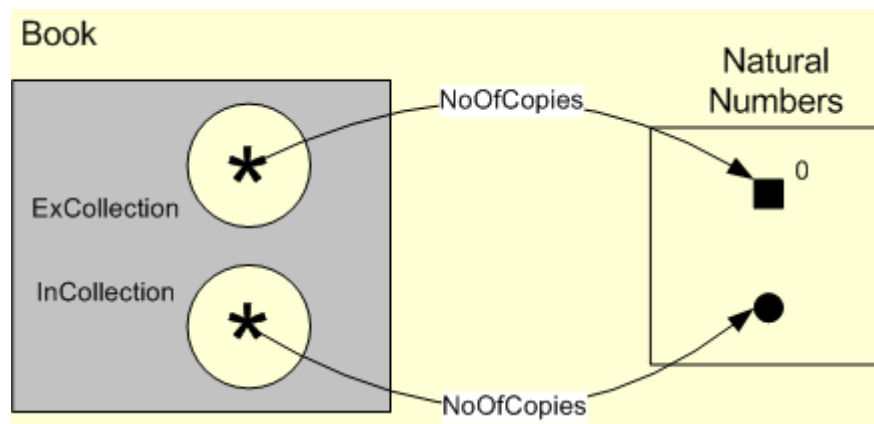
We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.



### 3.11 Question 9

Using the constraint diagram below, with the possible answers:

1. All books have a number of copies equal to zero
2. All books have a number of copies greater than zero
3. All books with a number of copies available are InCollection while all books with no copies available are ExCollection
4. All books with a number of copies available are ExCollection while all books with no copies available are InCollection



**Figure 19: All books with a number of copies available are InCollection while all books with no copies available are ExCollection (option 3)**

**Crosstab**

			Q9			Total
			2	3	4	
Group	Unfamiliar	Count	1	30	0	31
		% within Group	3.2%	96.8%	.0%	100.0%
		% within Q9	33.3%	46.9%	.0%	45.6%
		% of Total	1.5%	44.1%	.0%	45.6%
		Std. Residual	-.3	.2	-.7	
	Familiar	Count	2	34	1	37
		% within Group	5.4%	91.9%	2.7%	100.0%
		% within Q9	66.7%	53.1%	100.0%	54.4%
		% of Total	2.9%	50.0%	1.5%	54.4%
		Std. Residual	.3	-.1	.6	
Total	Count	3	64	1	68	
	% within Group	4.4%	94.1%	1.5%	100.0%	
	% within Q9	100.0%	100.0%	100.0%	100.0%	
	% of Total	4.4%	94.1%	1.5%	100.0%	

**Table 29: Crosstab of group by response for question 9**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	1.062 <sup>a</sup>	2	.588
Likelihood Ratio	1.446	2	.485
Linear-by-Linear Association	.008	1	.929
N of Valid Cases	68		

a. 4 cells (66.7%) have expected count less than 5. The minimum expected count is .46.

**Table 30: Chi-Square Tests for question 9**

We have tabulated the results for question 9 showing the given three out of a possible four responses. As there are 4 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value. For this we will assume that “partially correct”

results are, in fact, incorrect and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q9T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	1	30	31
		% within Group	3.2%	96.8%	100.0%
		% within Q9T	25.0%	46.9%	45.6%
		% of Total	1.5%	44.1%	45.6%
		Std. Residual	-.6	.2	
	Familiar	Count	3	34	37
		% within Group	8.1%	91.9%	100.0%
		% within Q9T	75.0%	53.1%	54.4%
		% of Total	4.4%	50.0%	54.4%
		Std. Residual	.6	-.1	
Total	Count	4	64	68	
	% within Group	5.9%	94.1%	100.0%	
	% within Q9T	100.0%	100.0%	100.0%	
	% of Total	5.9%	94.1%	100.0%	

**Table 31: Crosstab of group by transformed response for question 9**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.726 <sup>a</sup>	1	.394		
Continuity Correction <sup>b</sup>	.112	1	.738		
Likelihood Ratio	.767	1	.381		
Fisher's Exact Test				.620	.377
Linear-by-Linear Association	.716	1	.398		
N of Valid Cases	68				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1.82.

b. Computed only for a 2x2 table

**Table 32: Chi-Square Tests for transformed responses for question 9**

Our transformed table still does not yield counts above the expected level of 5, thus we cannot use the Pearson Chi-Squared test. Instead, we must use the Fisher's Exact Test (1-sided) to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.377 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.

### 3.12 Question 10

Using the constraint diagram below, with the possible answers:

1. The specification of a car assigned to a reservation must be the same or better than the specification reserved
2. The specification of a car assigned to a reservation must be the same as the specification reserved
3. The specification of a car assigned to a reservation must be better than the specification reserved
4. The specification of a car assigned to a reservation must not be better than the specification reserved

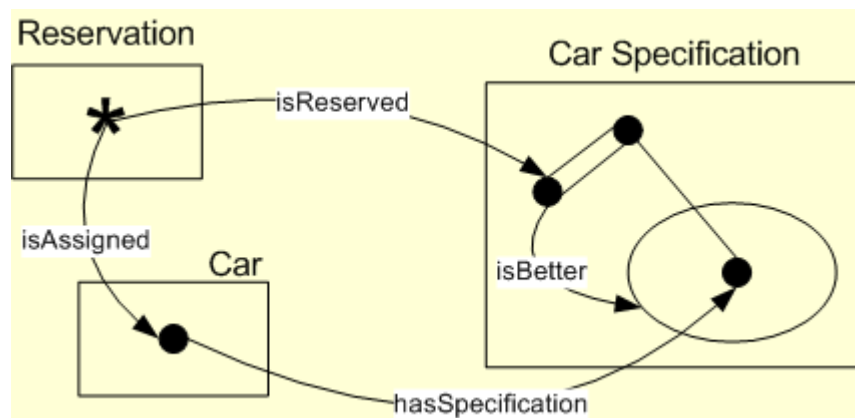


Figure 20: The specification of a car assigned to a reservation must be the same or better than the specification reserved (option 1)

			Q10				Total
			1	2	3	4	
Group	Unfamiliar	Count	14	6	8	3	31
		% within Group	45.2%	19.4%	25.8%	9.7%	100.0%
		% within Q10	40.0%	66.7%	47.1%	42.9%	45.6%
		% of Total	20.6%	8.8%	11.8%	4.4%	45.6%
		Std. Residual	-.5	.9	.1	-.1	
	Familiar	Count	21	3	9	4	37
		% within Group	56.8%	8.1%	24.3%	10.8%	100.0%
		% within Q10	60.0%	33.3%	52.9%	57.1%	54.4%
		% of Total	30.9%	4.4%	13.2%	5.9%	54.4%
		Std. Residual	.4	-.9	.0	.1	
Total	Count	35	9	17	7	68	
	% within Group	51.5%	13.2%	25.0%	10.3%	100.0%	
	% within Q10	100.0%	100.0%	100.0%	100.0%	100.0%	
	% of Total	51.5%	13.2%	25.0%	10.3%	100.0%	

**Table 33: Crosstab of group by response for question 10**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	2.089 <sup>a</sup>	3	.554
Likelihood Ratio	2.101	3	.552
Linear-by-Linear Association	.166	1	.684
N of Valid Cases	68		

a. 4 cells (50.0%) have expected count less than 5. The minimum expected count is 3.19.

**Table 34: Chi-Square Tests for question 10**

We have tabulated the results for question 10 showing all four possible responses. As there are 4 cells with expected count less than 5, we must reject this tabulation in favour of one that transforms the given responses into a correct or incorrect recoded value. For this we will assume that “partially correct” results are, in fact, incorrect

and recode as appropriate. We will then create a crosstab of the new set of data, along with a Chi-Square test, to identify other interesting results.

**Crosstab**

			Q10T		Total
			Incorrect	Correct	
Group	Unfamiliar	Count	17	14	31
		% within Group	54.8%	45.2%	100.0%
		% within Q10T	51.5%	40.0%	45.6%
		% of Total	25.0%	20.6%	45.6%
		Std. Residual	.5	-.5	
	Familiar	Count	16	21	37
		% within Group	43.2%	56.8%	100.0%
		% within Q10T	48.5%	60.0%	54.4%
		% of Total	23.5%	30.9%	54.4%
		Std. Residual	-.5	.4	
Total		Count	33	35	68
		% within Group	48.5%	51.5%	100.0%
		% within Q10T	100.0%	100.0%	100.0%
		% of Total	48.5%	51.5%	100.0%

**Table 35: Crosstab of group by transformed response for question 10**

Chi-Square Tests					
	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.908 <sup>a</sup>	1	.341		
Continuity Correction <sup>b</sup>	.503	1	.478		
Likelihood Ratio	.910	1	.340		
Fisher's Exact Test				.465	.239
Linear-by-Linear Association	.895	1	.344		
N of Valid Cases	68				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.04.

b. Computed only for a 2x2 table

**Table 36: Chi-Square Tests for transformed responses for question 10**

Our transformed table now yields counts above the expected level of 5, thus we can use the Pearson Chi-Squared test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the two groups in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between the two groups in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.341 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the two groups in the chance of correctly identifying the invariant from the diagram.



### 3.13 Hypothesis testing and inference

“With inferential statistics, we are trying to reach conclusions that extend beyond the immediate data alone. For instance, we use inferential statistics to try to infer from the sample data what the population might think. Or, we use inferential statistics to make judgments of the probability that an observed difference between groups is a dependable one or one that might have happened by chance in this study. Thus, we use inferential statistics to make inferences from our data to more general conditions; we use descriptive statistics simply to describe what's going on in our data [47].”

Group		N	Mean	Std. Deviation	Minimum	Maximum	Percentiles		
							25th	50th (Median)	75th
Unfamiliar	Correct	31	8.39	1.308	5	10	8.00	9.00	9.00
	Partial	31	1.06	.892	0	3	.00	1.00	2.00
	Incorrect	31	.55	.723	0	2	.00	.00	1.00
	Valid N (listwise)	31							
Familiar	Correct	37	8.92	1.187	5	10	8.00	9.00	10.00
	Partial	37	.76	1.090	0	5	.00	.00	1.50
	Incorrect	37	.32	.580	0	2	.00	.00	1.00
	Valid N (listwise)	37							

**Table 37: Descriptive Statistics for responses to all 10 questions combined**

### 3.14 Hypothesis tests of differences between groups

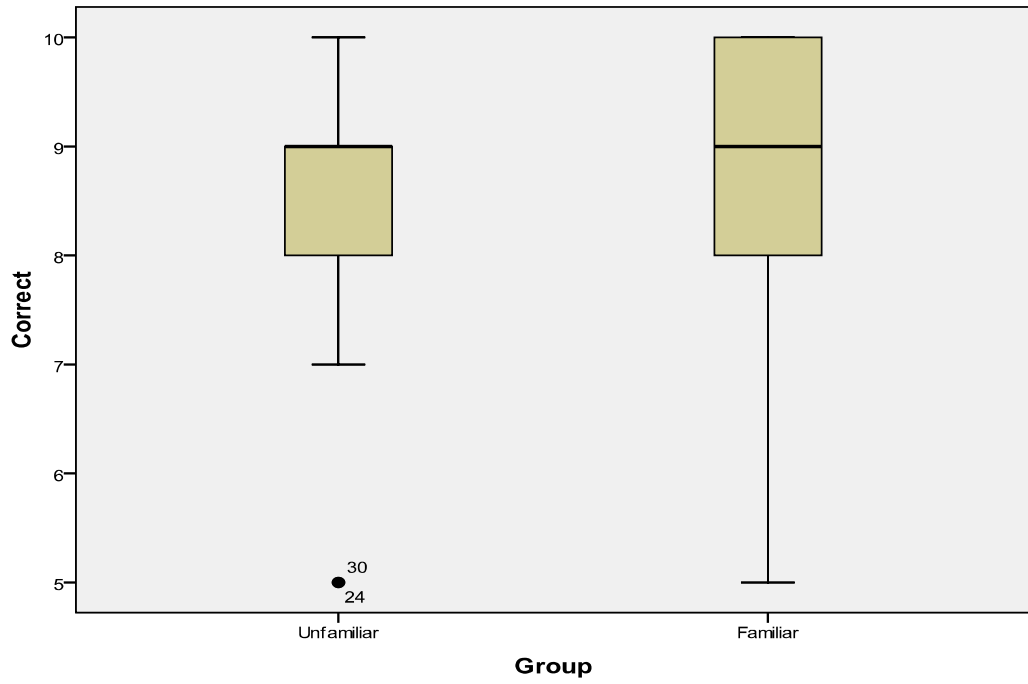
We examine the correct, partial and incorrect responses for all ten questions to determine if they are normally distributed. We are looking at two distinct groups of participants in a study with two conditions i.e. those who are familiar with the notation and those who are not. An assumption of a 2-sample t-test is that the variable is normally distributed.

		Tests of Normality					
		Kolmogorov-Smirnov <sup>a</sup>			Shapiro-Wilk		
	Group	Statistic	df	Sig.	Statistic	df	Sig.
Correct	Unfamiliar	.261	31	.000	.859	31	.001
	Familiar	.224	37	.000	.822	37	.000
Partial	Unfamiliar	.239	31	.000	.860	31	.001
	Familiar	.324	37	.000	.698	37	.000
Incorrect	Unfamiliar	.357	31	.000	.718	31	.000
	Familiar	.442	37	.000	.597	37	.000

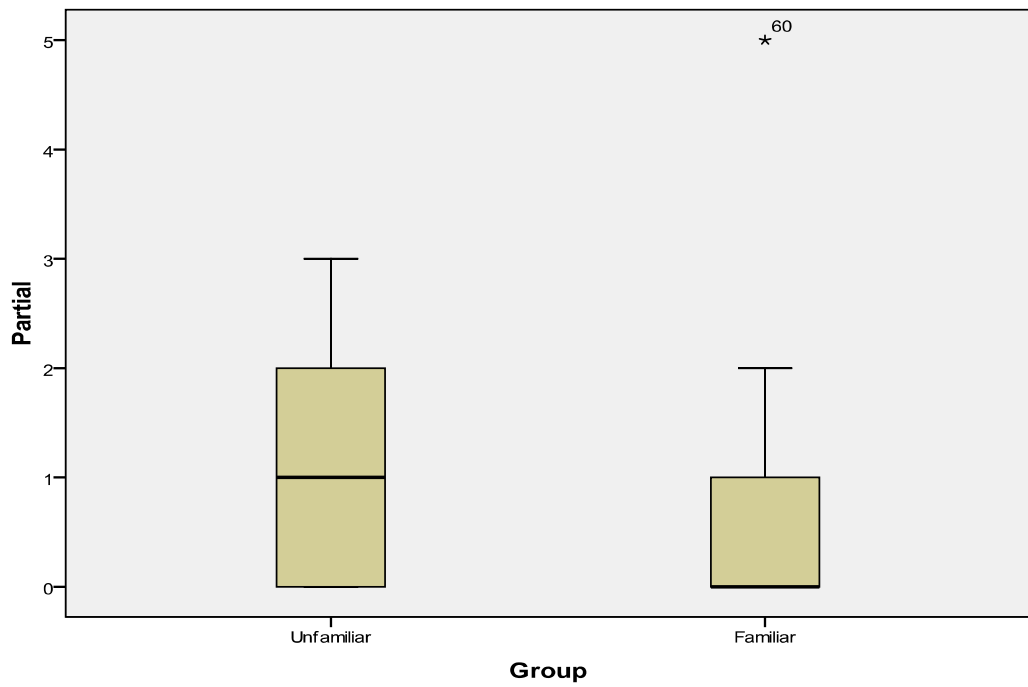
a. Lilliefors Significance Correction

**Table 38: Tests for Normality for responses to all 10 questions combined**

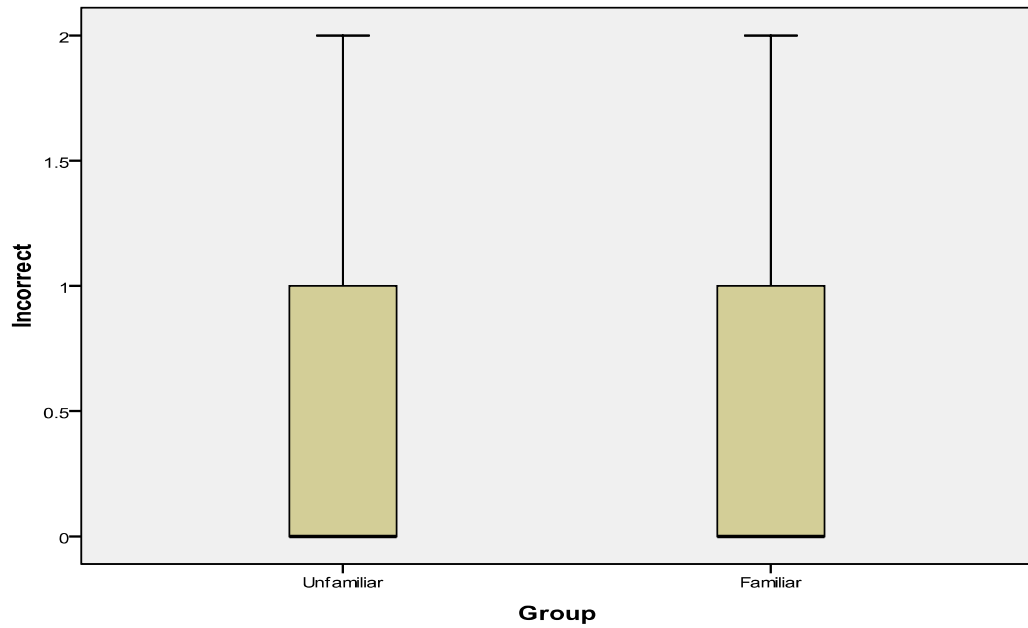
Looking at the calculated Shapiro-Wilk significance for all groups, we see the p-value is lower than 0.05 or 5%. This indicates a non-normal distribution for the samples. This result can be indicated graphically with the following box-plots.



**Figure 21: Box plot of correct responses by group**



**Figure 22: Box plot of partially correct responses by group**



**Figure 23: Box plot of incorrect responses by group**

As we have identified the distribution as non-normal, we will use non-parametric tests to infer over a larger population, i.e. all software engineers, whether the diagrams can be interpreted correctly.

### 3.15 Hypothesis tests of differences between the groups using non-parametric methods

Descriptive Statistics								
	N	Mean	Std. Deviation	Minimum	Maximum	Percentiles		
						25th	50th (Median)	75th
Correct	68	8.68	1.263	5	10	8.00	9.00	10.00
CorrectOrPartial	68	9.59	.652	8	10	9.00	10.00	10.00

**Table 39: Non-parametric tests (descriptive statistics) for Mann-Whitney test**

Ranks				
	Group	N	Mean Rank	Sum of Ranks
Correct	Unfamiliar	31	29.87	926.00
	Familiar	37	38.38	1420.00
	Total	68		
CorrectOrPartial	Unfamiliar	31	32.08	994.50
	Familiar	37	36.53	1351.50
	Total	68		

**Table 40: Mann-Whitney Test (non-parametric) Ranks**

Test Statistics <sup>a</sup>		
	Correct	CorrectOrPartial
Mann-Whitney U	430.000	498.500
Wilcoxon W	926.000	994.500
Z	-1.837	-1.123
Asymp. Sig. (2-tailed)	.066	.262

a. Grouping Variable: Group

**Table 41: Mann-Whitney Test (non-parametric) Test Statistics for Group**

**Report**

Group		Correct	CorrectOrPartial
Unfamiliar	Median	9.00	10.00
	Minimum	5	8
	Maximum	10	10
	Mean	8.39	9.48
	Std. Deviation	1.308	.724
	N	31	31
Familiar	Median	9.00	10.00
	Minimum	5	8
	Maximum	10	10
	Mean	8.92	9.68
	Std. Deviation	1.187	.580
	N	37	37
Total	Median	9.00	10.00
	Minimum	5	8
	Maximum	10	10
	Mean	8.68	9.59
	Std. Deviation	1.263	.652
	N	68	68

**Table 42: Mann-Whitney Test (non-parametric) Means and Medians**

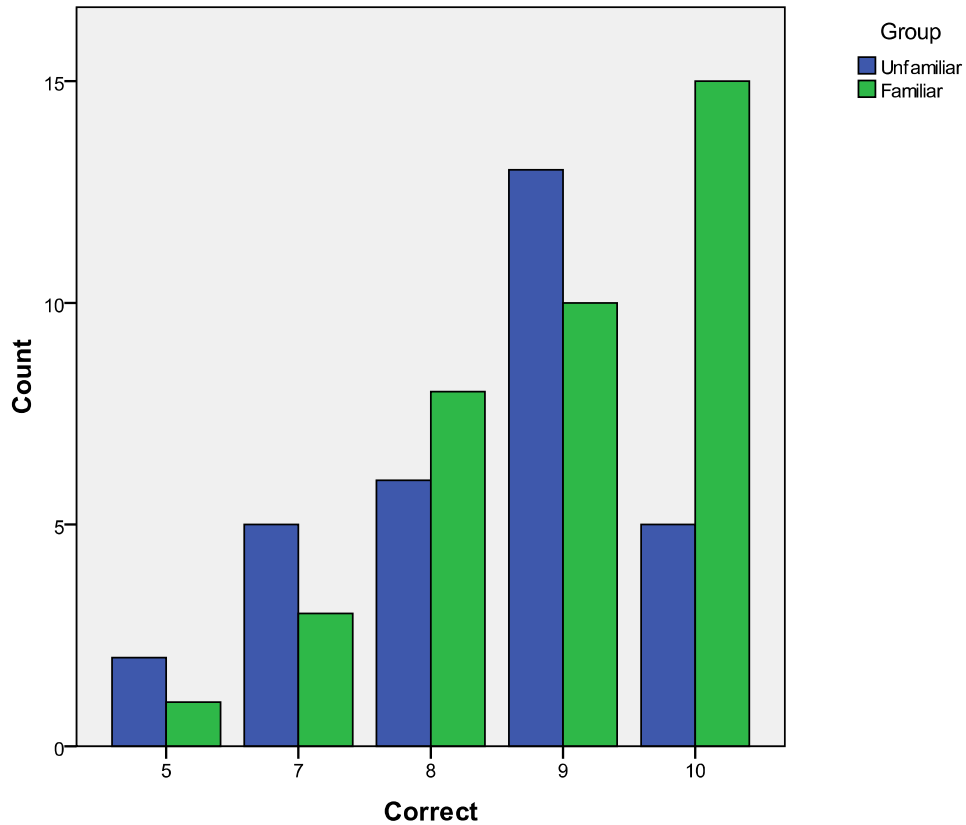
Mann-Whitney test for correct responses:

Null Hypothesis  $H_0$ : There is no difference between those familiar and those unfamiliar with the notation, in the chance of correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between those familiar and those unfamiliar with the notation, in the chance of correctly identifying the invariant from the diagram.

P-Value: 0.066 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no difference between those familiar and those unfamiliar with the notation, in the chance of correctly identifying the invariant from the diagram.



**Figure 24: Bar graph representing correct responses by group**

Looking at the graph in figure 24, we can see that those who are familiar with the notation have a higher greater propensity for correctly identifying all the invariants, while those who are unfamiliar seem to have a greater propensity for partially or incorrectly identifying one diagram. Both groups, however, seem to have a similar propensity for partially or incorrectly identifying two or more invariants.

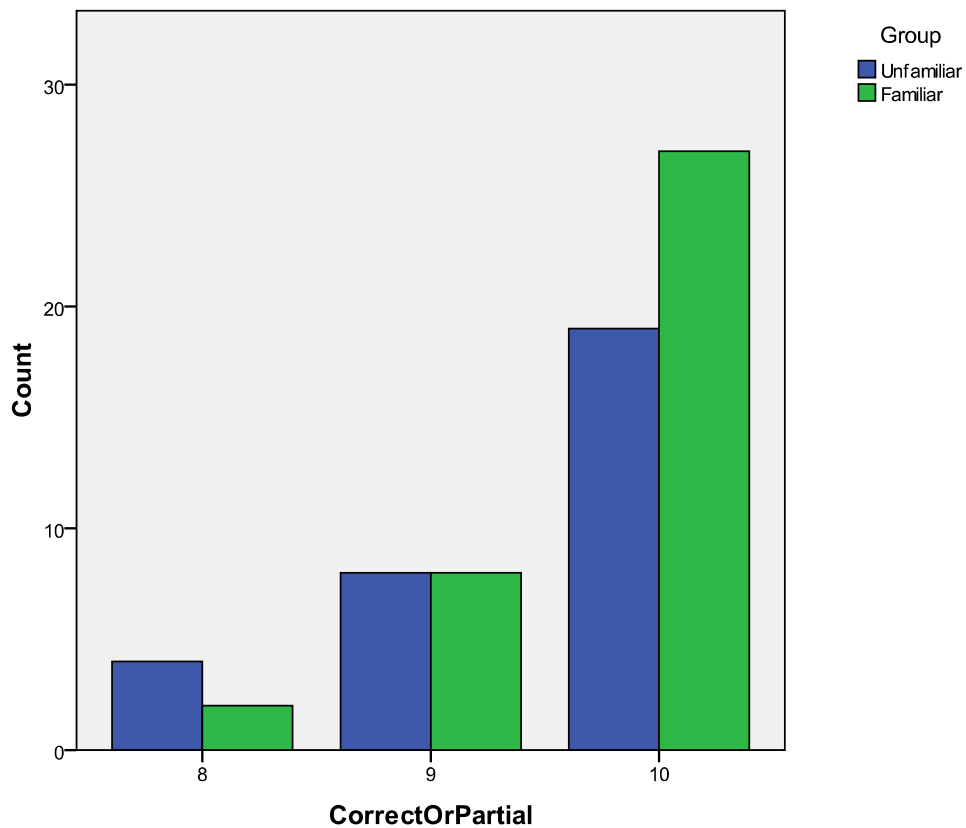
Mann-Whitney test for combined correct and partially correct responses:

Null Hypothesis  $H_0$ : There is no difference between those familiar and those unfamiliar with the notation, in the chance of correctly or partially correctly identifying the invariant from the diagram.

Alternative Hypothesis  $H_1$ : There is a difference between those familiar and those unfamiliar with the notation, in the chance of correctly or partially correctly identifying the invariant from the diagram.

P-Value: 0.262 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between those familiar and those unfamiliar with the notation, in the chance of partially identifying the invariant from the diagram.



**Figure 25: Bar graph representing correct and partially correct responses by group**

Looking at the graph in figure 25, we can see that those who are familiar with the notation have a higher greater propensity for correctly or partially identifying all ten of the invariants. However, those participants who correctly or partially identified eight or nine invariants tended to have the same approximate chance regardless of the previous experience with the notation.



### 3.16 Modelling the probability of obtaining a correct response

Logistic regression is used for prediction of the probability of occurrence of an event by fitting data to a logistic curve. It is a generalized linear model used for binomial regression. Like many forms of regression analysis, it makes use of several predictor variables that may be either numerical or categorical. Using logistic regression, we will investigate whether the group is predictive of a correct response to the constraint diagram.

First, we perform descriptive statistics to describe the main features of the collection of data in quantitative terms.

**Group \* Correct\_v2 Crosstabulation**

Count

		Correct_v2		Total
		Correct	Incorrect	
Group	Unfamiliar	260	50	310
	Familiar	322	38	360
Total		582	88	670

**Table 43: Logistic regression: crosstabulation**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	4.535 <sup>a</sup>	1	.033		
Continuity Correction <sup>b</sup>	4.060	1	.044		
Likelihood Ratio	4.524	1	.033		
Fisher's Exact Test				.039	.022
Linear-by-Linear Association	4.528	1	.033		
N of Valid Cases	670				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 40.72.

b. Computed only for a 2x2 table

**Table 44: Logistic regression: chi-square tests**

Using the Pearson Chi-Square test:

Null Hypothesis  $H_0$ : There is no association between the given response and the group.

Alternative Hypothesis  $H_1$ : There is an association between the given response and the group.

P-Value: 0.033 (less than 0.05 or 5%)

We reject  $H_0$ : there is an association between the given response and the group.

This result is also borne out using the Fisher's Exact Test (P-Value: 0.039).

Risk Estimate			
	Value	95% Confidence Interval	
		Lower	Upper
Odds Ratio for Group (Unfamiliar / Familiar)	.614	.390	.965
For cohort Correct_v2 = Correct	.938	.883	.996
For cohort Correct_v2 = Incorrect	1.528	1.031	2.265
N of Valid Cases	670		

**Table 45: Logistic regression: Risks**

We also calculate the odds as:

Unfamiliar group attaining a correct response =  $260/310 = 0.839$

Unfamiliar group attaining an incorrect response =  $50/310 = 0.161$

Observed odds of attaining a correct response =  $260/50 = 5.200$

Familiar group attaining a correct response =  $322/360 = 0.894$

Familiar group attaining an incorrect response =  $38/360 = 0.106$

Observed odds of attaining a correct response =  $322/38 = 8.474$

We also calculate the odds ratio<sup>2</sup> as:

$$\frac{\text{Observed odds of unfamiliar group attaining a correct response}}{\text{observed odds of familiar group attaining a correct response}} = \frac{5.200}{8.474} = 0.614$$

Looking at the risk estimate odds ratio for group (unfamiliar/familiar) in table 45 above, we see the computed odds ratio is 0.614. We see our 95% confidence level has a lower boundary of 0.390, while the upper boundary has a value of 0.965. As this range does not contain 1 as a value, there is a significant association between the given response and the group.

		Variables in the Equation						95% C.I. for Odds Ratio	
		B	S.E.	Wald	df	Sig.	Odds Ratio	Lower	Upper
Step 1 <sup>a</sup>	Group	-.428	.233	3.358	1	.067	.652	.413	1.030
	LogTime	1.089	.344	10.029	1	.002	2.970	1.514	5.826
	Constant	-3.122	.725	18.529	1	.000	.044		

a. Variable(s) entered on step 1: Group, LogTime.

**Table 46: Logistic regression: variables**

The odds for the unfamiliar group obtaining a correct response, when adjusted for time taken, are 0.652 x the odds for the familiar group. This is slightly greater than previously calculated as we have adjusted for time taken, and is marginally not significant as the p-value for the Wald test results in 0.067, i.e. is not statistically significant at the 5% level.

If the odds of achieving the correct result were the same for both familiar and unfamiliar, the odds ratio would be 1. As the odds ratio is less than 1, software engineers who are familiar with the notation are more likely to achieve a correct result.

---

<sup>2</sup> The odds ratio is a measure of effect size, describing the strength of association between two binary data values.

### 3.17 Investigating relationships between group, time taken and accuracy of response

As there are no readily available non-parametric tests for examining time intervals, we have used a parametric test, the Univariate Analysis of Variance and have performed a transformation of the data to give an approximation only. The time taken depends on the group, i.e. whether the group is familiar with the notation or not, and the difference between groups depends on whether the answer is correct or not.

Univariate analysis consists of describing and explaining the variation in a single variable. We will examine the time taken to respond to each question and perform a Univariate Analysis of Variance on the time taken against the group and accuracy of response.

Between-Subjects Factors			
		Value Label	N
Group	1	Unfamiliar	310
	2	Familiar	360
Correct	0	Incorrect	88
	1	Correct	582

Table 47: Univariate ANOVA between-subjects factors

### Descriptive Statistics

Dependent Variable:Time

Group	Correct	Mean	Std. Deviation	N
Unfamiliar	Incorrect	0:01:20.620	0:00:52.991	50
	Correct	0:01:06.669	0:01:08.583	260
	Total	0:01:08.919	0:01:06.440	310
Familiar	Incorrect	0:01:25.789	0:01:32.819	38
	Correct	0:00:59.134	0:01:28.579	322
	Total	0:01:01.947	0:01:29.280	360
Total	Incorrect	0:01:22.852	0:01:12.472	88
	Correct	0:01:02.500	0:01:20.287	582
	Total	0:01:05.173	0:01:19.551	670

**Table 48: Univariate ANOVA between-subjects descriptive statistics**

### Tests of Between-Subjects Effects

Dependent Variable:Time

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	40409.009 <sup>a</sup>	3	13469.670	2.139	.094
Intercept	1603002.629	1	1603002.629	254.597	.000
Group	105.111	1	105.111	.017	.897
Correct	30955.146	1	30955.146	4.916	.027
Group * Correct	3030.386	1	3030.386	.481	.488
Error	4193284.907	666	6296.224		
Total	7079544.000	670			
Corrected Total	4233693.916	669			

a. R Squared = .010 (Adjusted R Squared = .005)

**Table 49: Univariate ANOVA tests of between-subjects effects**

Hypothesis test for Group:

Null Hypothesis  $H_0$ : There is no significant difference in the chance of selecting the correct invariant based on the group of the participant.

Alternative Hypothesis  $H_1$ : There is a significant difference in the chance of selecting the correct invariant based on the group of the participant.

P-Value: 0.897 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference in the chance of selecting the correct invariant based on the group of the participant.

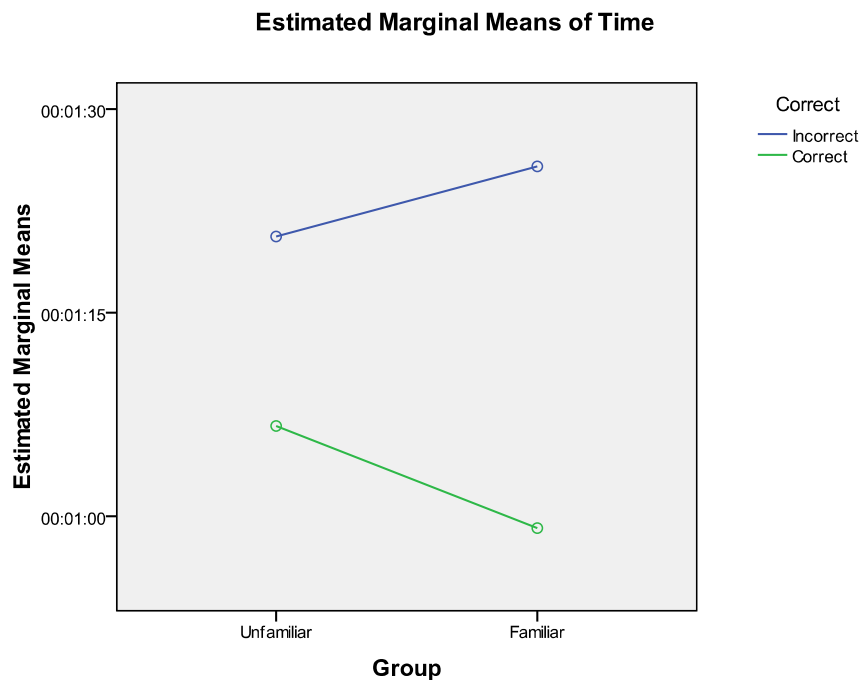
Hypothesis test for Correct/Incorrect response:

Null Hypothesis  $H_0$ : There is no difference in the chance of selecting the correct invariant, regardless of the time spent examining the diagram.

Alternative Hypothesis  $H_1$ : There is a significant difference in the chance of selecting the correct invariant, depending on the time spent examining the diagram.

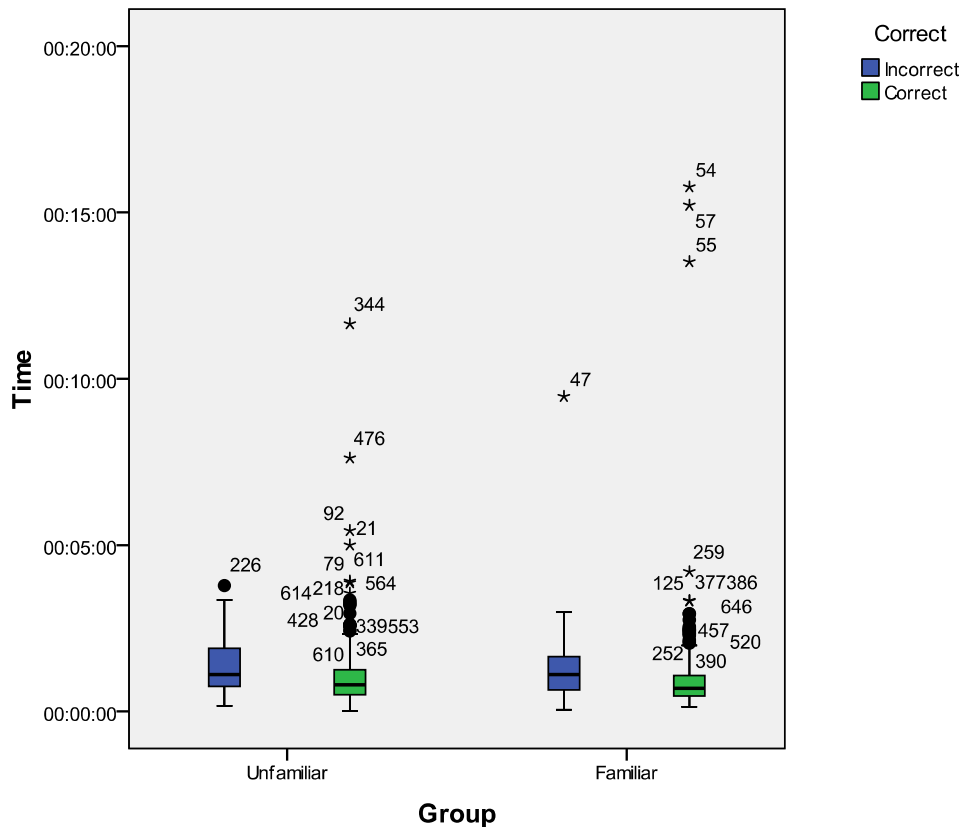
P-Value: 0.027 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference in the chance of selecting the correct invariant, depending on the time spent examining the diagram.



**Figure 26: Estimated marginal means of time against group for correct vs. incorrect responses.**

Figure 26 shows that the group familiar with the constraint diagram notation can identify the correct invariant faster than those who are unfamiliar with the notation. It also shows that the unfamiliar group provides incorrect responses faster than the familiar group.



**Figure 27: Box plots of responses plotted against time and group.**

Figure 27 represents the plot of correct and incorrect responses to the questions against the time taken to arrive at those responses. As we can see, the plots show a non-normal distribution (right skewed), with a number of outliers.

### 3.18 Further investigations of the relationships between group, time taken and accuracy of response using a log-transformed time

To check that the distribution of both groups is non-normal i.e. right skewed, and to improve the clarity of the graph, we will perform a log transformation on the time variable, and re-plot the graph.

**Between-Subjects Factors**

		Value Label	N
Group	1	Unfamiliar	310
	2	Familiar	360
Correct	0	Incorrect	88
	1	Correct	582

**Table 50: Univariate ANOVA - log transformation for between-subjects factors**

**Descriptive Statistics**

Dependent Variable: LogTime

Group	Correct	Mean	Std. Deviation	N
Unfamiliar	Incorrect	1.8044	.31999	50
	Correct	1.6842	.34827	260
	Total	1.7035	.34622	310
Familiar	Incorrect	1.7580	.44025	38
	Correct	1.6306	.31145	322
	Total	1.6440	.32901	360
Total	Incorrect	1.7843	.37501	88
	Correct	1.6545	.32920	582
	Total	1.6716	.33813	670

**Table 51: Univariate ANOVA - log transformation for between-subjects descriptive statistics**



**Tests of Between-Subjects Effects**

Dependent Variable:LogTime

Source	Type III Sum of Squares	df	Mean Square	F	Sig.
Corrected Model	1.747 <sup>a</sup>	3	.582	5.189	.002
Intercept	887.859	1	887.859	7911.373	.000
Group	.188	1	.188	1.671	.197
Correct	1.151	1	1.151	10.253	.001
Group * Correct	.001	1	.001	.009	.926
Error	74.742	666	.112		
Total	1948.565	670			
Corrected Total	76.489	669			

a. R Squared = .023 (Adjusted R Squared = .018)

**Table 52: Univariate ANOVA - log transformation for between-subjects effects**

Hypothesis test for Group (LogTime):

Null Hypothesis  $H_0$ : There is no difference between groups in mean time taken in the chance of selecting the correct invariant based on the group of the participant.

Alternative Hypothesis  $H_1$ : There is a difference between groups in mean time taken in the chance of selecting the correct invariant based on the group of the participant.

P-Value: 0.197 (greater than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference between groups in mean time taken in the chance of selecting the correct invariant based on the group of the participant.

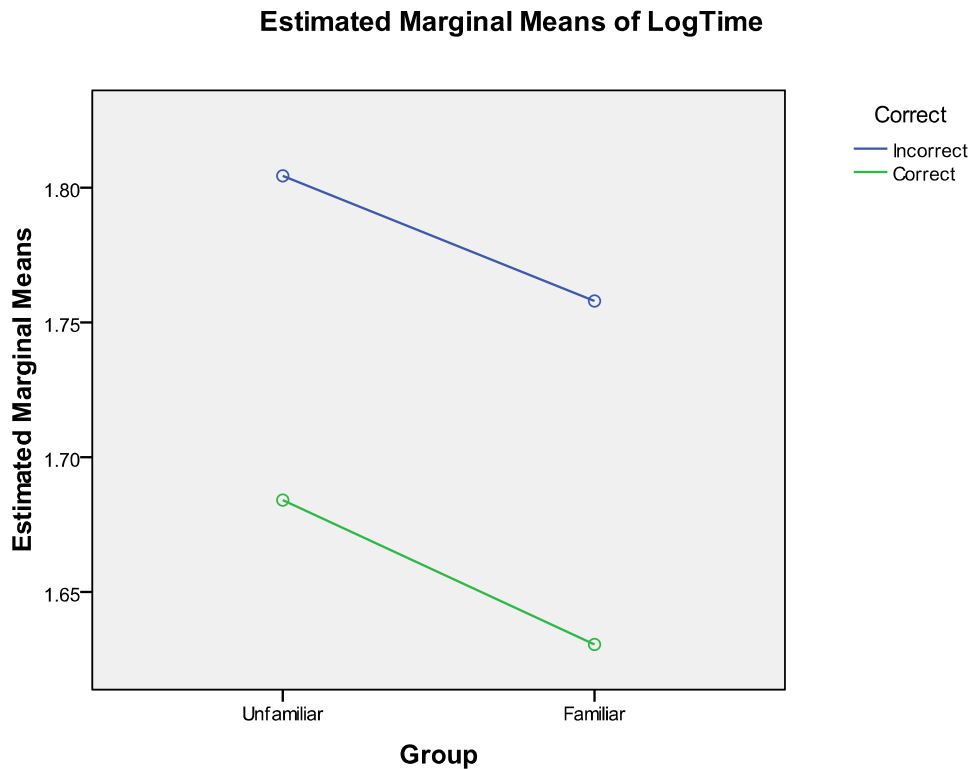
Hypothesis test for Correct/Incorrect response (LogTime):

Null Hypothesis  $H_0$ : There is no difference in the chance of selecting the correct invariant, regardless of the time spent examining the diagram.

Alternative Hypothesis  $H_1$ : There is a significant difference in the chance of selecting the correct invariant, depending on the time spent examining the diagram.

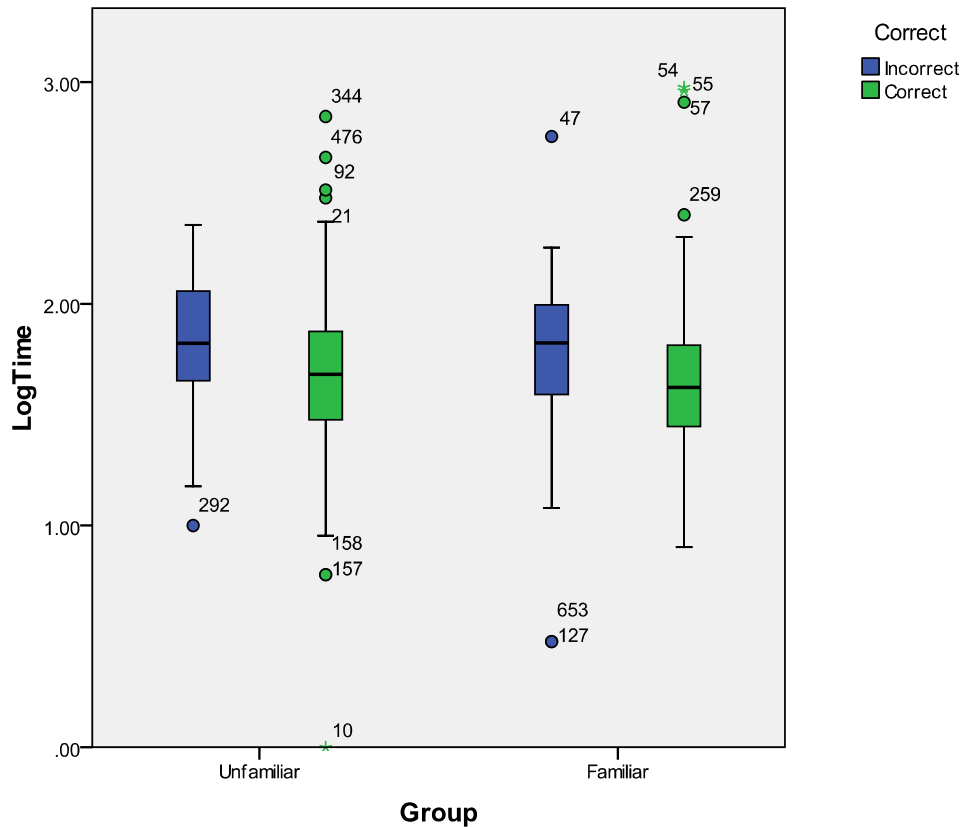
P-Value: 0.001 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference in the chance of selecting the correct invariant, depending on the time spent examining the diagram.



**Figure 28: Estimated marginal means of log(time) against group for correct vs. incorrect responses.**

Figure 28 shows that the group familiar with the constraint diagram notation can identify the correct invariant faster than those who are unfamiliar with the notation. It also shows that the familiar group provides incorrect responses faster than the unfamiliar group. This would indicate that the group familiar with the constraint diagram notation are faster in general at providing an interpretation of a constraint diagram than those who are unfamiliar with the notation.



**Figure 29: Box plots of responses plotted against log transformed time and group**

Figure 29 shows a much clearer distribution of the responses given for each group over time. We can still see a right skewed distribution on all plots.

### 3.19 Modelling the time taken to complete each question

As well as performing a logistic regression, we can further investigate the times taken to respond with an appropriate answer, using Kaplan-Meier Survival Analysis<sup>3</sup>.

<sup>3</sup> Kaplan-Meier survival analysis is a technique that involves the generation of the tables and plots of survival or hazard function for event history data. Kaplan-Meier survival analysis (KMSA) has not been designed to assess the effects of the covariates on either function. Kaplan-Meier survival analysis is a descriptive procedure for the time to event variables in cases where time is the most prominent variable [49]. The term “survival” is a bit misleading; you can use survival curves to study times required to reach any well-defined end-point [50].

**Case Processing Summary**

Group	Total N	N of Events	Censored	
			N	Percent
Unfamiliar	310	260	50	16.1%
Familiar	360	322	38	10.6%
Overall	670	582	88	13.1%

**Table 53: Kaplan-Meier Survival Analysis: Case processing summary**

**Means and Medians for Survival Time**

Group	Mean <sup>a</sup>				Median			
	Estimate	Std. Error	95% Confidence Interval		Estimate	Std. Error	95% Confidence Interval	
			Lower Bound	Upper Bound			Lower Bound	Upper Bound
Unfamiliar	83.295	6.417	70.718	95.872	53.000	3.069	46.985	59.015
Familiar	77.192	8.988	59.575	94.810	46.000	1.746	42.579	49.421
Overall	80.376	5.938	68.737	92.015	50.000	1.709	46.651	53.349

a. Estimation is limited to the largest survival time if it is censored.

**Table 54: Kaplan-Meier Survival Analysis: Means and medians for survival time**

**Overall Comparisons**

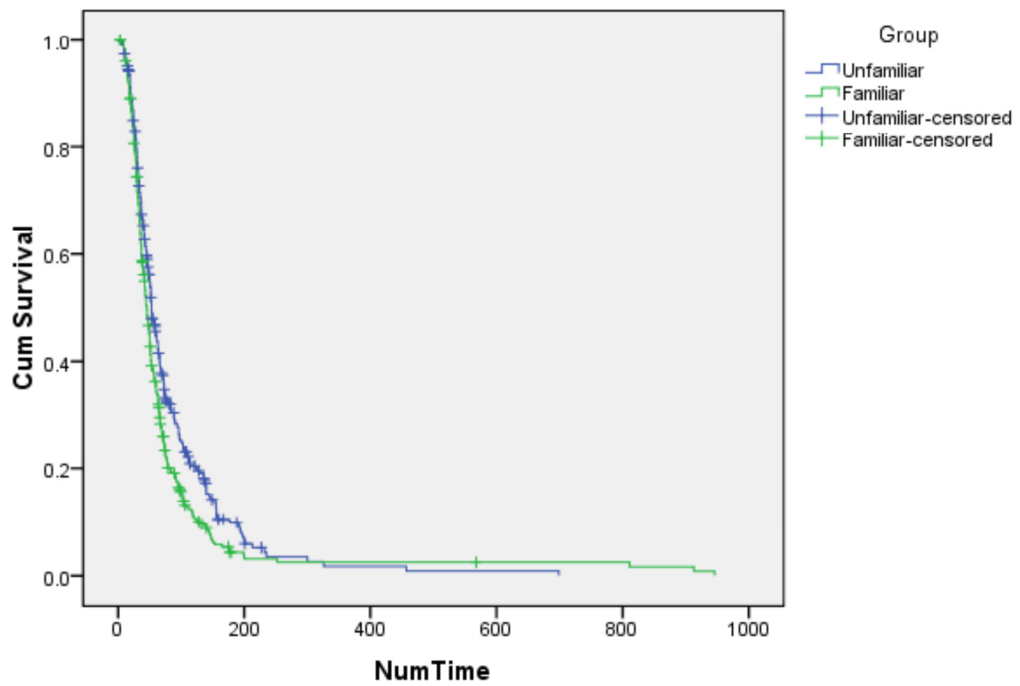
	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	9.253	1	.002

Test of equality of survival distributions for the different levels of Group.

**Table 55: Kaplan-Meier Survival Analysis: Overall comparisons**

While the P-Value above is 0.002 (less than 0.05 or 5%), in this case the P-Value does not convey information about the size of the effect [51]. Additional statistics, such as follow-up time for each group, the total number of events and the total number of participants who remain event free are important for interpreting the data.

### Survival Functions



**Figure 30: Kaplan-Meier Survival Analysis Plot**

From figure 30 we see that the group familiar with the notation take a shorter time to identify the invariant than that of the group unfamiliar with the notation. This is borne out by the median estimates of time taken in table 54 above, where the familiar group has a median of 46 seconds and the unfamiliar group has a median of 53 seconds.

## **4 Conclusions gained from the quantitative analysis**

Following on from the quantitative analysis of the responses to the 10 questions, we will now discuss our findings and draw conclusions as to whether the constraint diagram notation is more readily interpreted by software engineers.

### **4.1 The findings of the descriptive statistics**

Examining all the crosstabs, we can posit that for simple diagrams with one piece of information i.e. questions one, two and three, being represented in a straightforward way, the diagrams are interpreted correctly, within the framework of the given possible responses, by the substantial majority of participants. These questions describe many of the common features of the constraint diagram notation e.g. sets, spiders (universal and existential), and arrows.

However, question one does appear to have a rather high instance of incorrect answers, approx 25% for those who are unfamiliar with the constraint diagram notation, and approx. 16% for those who are familiar with the constraint diagram notation. Further investigation seems to show that all but one participant (from the unfamiliar group) has selected a partially correct response, rather than the fully incorrect response. This may be due to the fact that this is the first question, and participants are just beginning the test. They may have preconceived expectations as to what the first diagram may represent, especially having just read a standard text on the constraint diagram notation. Alternatively, they may have reached a conclusion as to the perceived correct response faster than normal and chosen a partially correct answer, rather than the correct answer, without fully reading the four possible responses.

We can further posit that, as complexity increases, the diagrams are also interpreted correctly by the majority of participants, although the value of the majority does tend to fluctuate slightly. As we shall see from below, this may be due to a quasi-familiarity with the particular scenario being investigated.

Question four, while increasing in complexity, is based on a real world and fairly familiar scenario, the driving instructor. Anyone who has taken driving lessons from

a registered driving instructor may, in fact, hazard a very good guess at this without having any experience of the notation. However, while it may be possible to work out the answer, it is still a valid constraint diagram, and demonstrates that this may be a very well-known domain for which to model. With the exception of one participant from the unfamiliar group who provided an incorrect response, and five participants from the unfamiliar group along with two participants from the familiar group who provided a partially correct response, this is the case.

Question five is based on a well-known academic case study, the lending library. It is common for computing students to be given the lending library case study when developing databases or applications, especially in the first year, as every University maintains at least one library of texts for students to use. This then, would become a familiar scenario to computing students, who would not only have first-hand knowledge of the scenario through borrowing books, but would also have modelled the scenario in various modelling languages. Therefore, while the diagram itself shows increasing complexity, displaying examples of all the elements shown previously, this scenario should be familiar enough for participants to achieve a fully correct response. With the exception of one participant from each study group, who both selected the same partially correct response, this is the case.

Question six extends the scenario of question five, using the lending library as a case study. It also introduces a new piece of notation, namely the strand (wavy line). While adding more complexity, building to more difficult constraints, we have also introduced the notion of simple data types as part of the constraint. As software engineers, we assume a level of mathematics that would identify a natural number as a non-negative integer, or as a primitive data type of unsigned long. We would also assume a level of understanding of the participants for the case study that would demonstrate the value of the relations “available”, “borrowed” and “copies” are at the same value (semantics of the strand) when we have zero books. Therefore, while the diagram itself introduces a new element of the notation, this scenario should be familiar enough for participants to achieve a correct response. With the exception of one participant from the familiar group, and four participants from the unfamiliar group, this is the case.

Question seven further extends the lending library case study, introducing a scenario for reserving a copy of a book. While introducing no new notation, the scenario itself adds the complexity by providing another set of objects. It is based on a familiar scenario; reserving a copy of a book. It shows a “behind the scenes” task within the system i.e. a member of the library would not need to know that a copy of a book must be on hold to be reserved. Therefore, while the diagram itself shows increasing complexity, displaying examples of all the elements shown previously but introducing an additional class or set, this scenario should be familiar enough for the majority of participants to achieve a fully correct response. With the exception of two participant from the familiar group, both of whom gave a partially correct response, and three participants from the unfamiliar group, two of which gave a partially correct response, this is the case.

Question eight describes a real world scenario; that of the status of a married couple. While the scenario itself is not complex and is well known, the modelling of the scenario is complex. Providing sets within sets and sets that cross other sets, there is only a subset of the constraint diagram notation used, equivalent to Euler diagrams. However, this diagram shows an almost 20% incorrect response rate from the participants. This may be a misunderstanding of the fact that if spiders are distinct within sets, they represent different objects, as do one spider and a set that does not contain the spider, as in the case of question eight. It does demonstrate one thing, though; the participants were examining the constraint and providing a response based on that constraint, rather than based on their real world experience. Otherwise, we could have 20% of the participants potentially wanting to marry a sibling!

Question nine returns to the lending library case study. We introduce one new piece of notation, the constant spider for the value zero. In reality, we could have used an existential spider and introduced no new notation. However, in itself adding the new notation has not introduced complexity. By saying there are only two sets of objects of the type Book, and that they must have a relation to a natural number rather plays on the mathematical knowledge of the participant. A natural number is a non-negative integer. This scenario should be familiar enough for the majority of participants to achieve a fully correct response. With the exception of one



participant from the unfamiliar group, and two from the familiar group who gave a partially correct response, and one participant from the familiar group who gave an incorrect response, this is the case.

Question ten was developed to deliberately give the most difficulty, and added a new area of the notation not seen in previous diagrams – the tie. Understanding that the tie represents the spiders being the same object as they are within the same zone (*isBetter*) is key to interpreting this diagram. The data shows that 45% of unfamiliar participants against 56% of familiar participants were able to correctly interpret the diagram, while only 9% of unfamiliar participants and 10% of familiar participants were unable to either correctly or partially interpret the diagram. The majority of participants were unable to correctly identify the significance of the tie and selected the interpretation that would give a partial result.

So far, looking at the responses as cross-tabulated, we can posit that the first hypothesis: “*Software Engineers and Students with previous training in Constraint Diagrams could more accurately interpret Constraint Diagrams than Software Engineers and Students with no previous exposure to Constraint Diagrams*” does not, in fact, hold and we see that the level of prior knowledge of constraint diagram notation software engineers have seems to be irrelevant when interpreting constraint diagrams.

While this fact is both interesting and important to note, and further studies may reinforce this, we must also observe that the study failed to demonstrate the overall objectives i.e. that the first hypothesis held. A rerun of the study may choose to use a different hypothesis, for example “*Software Engineers and Students could accurately interpret the constraint diagram notation regardless of their previous knowledge and experience of constraint diagrams*”.

We will now discuss further tests that can help us confirm the findings of the crosstab analysis.

#### **4.2 The findings of the inferential statistics**

We are trying to infer conclusions about the whole population of software engineers from the two sample groups who participated in the study.

We are testing the first hypothesis: “*Software Engineers and Students with previous training in Constraint Diagrams could more accurately interpret Constraint Diagrams than Software Engineers and Students with no previous exposure to Constraint Diagrams*”. We perform tests for normality, resulting in a non-normal distribution, so we use a Mann-Whitney Test (non-parametric),

Tests for the distribution of the samples were given using both the Kolmogorov-Smirnov and Shapiro-Wilk tests. For the benefits of the analysis of this study we will use the Shapiro-Wilk test only as it is a more accurate test. All groups show a distribution that deviates from a normal distribution; correct responses are left skewed, partial and incorrect responses are right skewed. The box plots in figures 21, 22 and 23 (p95-96) demonstrate this graphically. As such, we cannot use parametric tests, nor can we assume any particular distribution.

As the groups show a skewed distribution, we cannot perform a t-test. Instead, we must use a non-parametric equivalent. The Mann-Whitney test was used to determine whether the two groups of participants (familiar and unfamiliar) are drawn from the same distribution.

The Mann-Whitney test assigns a ranking based on the number of responses of the dependent variable (in this case the correct, or correct and partially correct responses), against the independent variable (in this case the group). For both correct and correct & partial responses, the assigned ranks for the familiar group are higher than those of the unfamiliar group. We also find that the calculated P-values for each group are above the 5% threshold, although the correct is only slightly above at 6.6%, while the P-value for the combined correct and partial is 21.2%. With this sample, therefore, there is no evidence of a true difference between the familiar and unfamiliar groups. We can therefore assume the two groups are from the same distribution. (Had one of the P-values been below the 5% threshold, we would have to assume the two groups are from different distributions, and therefore the test would have provided a different outcome.)

Overall, taking the responses to the questions in isolation, we can posit that the first hypothesis: “*Software Engineers and Students with previous training in Constraint Diagrams could more accurately interpret Constraint Diagrams than Software*

*Engineers and Students with no previous exposure to Constraint Diagrams*” does not, in fact, hold and we see that the level of prior knowledge of constraint diagram notation software engineers have seems to be irrelevant when interpreting constraint diagrams.

We are also trying to infer the time taken to complete an individual question and provide an appropriate response. We are, in fact, testing the second hypothesis: *“Software Engineers and Students with previous training in Constraint Diagrams could more rapidly interpret Constraint Diagrams than Software Engineers and Students with no previous exposure to Constraint Diagrams”*. In order to perform this test, we have performed a Logistic Regression, a Univariate Analysis of Variance (ANOVA), requiring transforming our non-parametric data into parametric data through a log transformation and a Kaplan-Meier Survival Analysis, both of which are suitable for analysing time intervals.

The Logistic Regression and calculated odds ratio demonstrate there is an association between the group and the response given. As we have already stated in section 3.16, if the odds of achieving the correct result were the same for both familiar and unfamiliar, the odds ratio would be 1. As the odds ratio is less than 1, software engineers who are familiar with the notation are more likely to achieve a correct result. This holds with the second hypothesis. It further indicates that we can infer this result to the wider population of software engineers as a whole.

The ANOVA shows us that the mean time taken to respond to the question has a relationship to 1) the group and 2) the correct or incorrect response provided as shown on tables 50 & 51 (*p108*) and figures 28 & 29 (*p110-111*). We can see from these tables and figures a correlation between groups. The mean time to report a correct response is 1.63 for familiar and 1.68 for unfamiliar groups, and incorrect responses are reported with a mean time of 1.76 for familiar and 1.80 for unfamiliar groups. We find that the familiar group has a faster mean response rate than the unfamiliar group. This holds with the second hypothesis, and also indicates that we can infer this result to the wider population of software engineers as a whole.

Interestingly, we also find a correlation within the groups i.e. we find that the mean time differences for identifying correct and incorrect results are approximately equal for both the familiar and unfamiliar groups.

The Kaplan-Meyer Survival Analysis shows us that each group has a number of censored events, in the case of our study, representing the participants' incorrect responses. For the unfamiliar group, who have a total of 310 responses, 50 are censored i.e. were incorrect. For the familiar group, who have a total of 360 responses, only 38 are censored. This gives an indication that the familiar group have a better chance of interpreting the correct invariant.

However, we are now looking at the times to identify the correct interpretation, so we need to delve further. When taking the median as our statistic, we have a 95% CI between 42.6 seconds and 49.4 seconds for the group familiar with the notation, and a 95% CI of between 47 seconds and 59 seconds for the group unfamiliar with the notation. This gives a good indication that those who are familiar with the notation are faster at interpreting the correct invariant. Another indicator is the median estimated time for taking the test. The median estimate for the unfamiliar group is 53 seconds, while the median estimate for the familiar group is 46 seconds. As the median is lower for the familiar group, it indicates an overall faster response time to obtain a correct interpretation.

Looking at the plots of Cumulative Survival (number of responses by group) against Numeric Time (number of seconds to correctly respond to the event), we see that the curve for the familiar group has a lower curve, showing the majority of the familiar group have a faster rate of time to experience the event, in this case to interpret the invariant. It should be noted also that all but one of the censored sub-group have completed their event by this time. The curve of the unfamiliar group indicates a greater time taken for the same number of responses.

Interestingly, the familiar group have several more outliers than the unfamiliar group, and have participants taking the greatest amount of time to respond to the event.

## **5 Qualitative analysis of the interpretation of constraint diagrams**

A paper questionnaire was presented to each participant. This consisted of multiple choice questions that will reflect four dimensions from the Cognitive Dimensions of Notations framework [36, 39], namely Closeness of Mapping, Consistency, Role Expressiveness and Secondary Notation. These four dimensions have been chosen for their suitability to static diagrams, rather than editable or developmental diagrams (i.e. to the study of interpretation, rather than manipulation and development, for which a full Cognitive Dimensions profile would be appropriate). The cut-down profile will use a modified version of the Cognitive Dimensions Questionnaire [52], taking into account only these four dimensions.

### **5.1 Closeness of mapping**

This dimension examines how close the diagrammatic representation maps to the domain being modelled. We are interested in how well the Constraint Diagrams represent our constraints. We will look at all given constraints and examine the corresponding Constraint Diagrams.

### **5.2 Consistency**

This dimension examines how similar semantics are expressed using similar syntactic forms. We are interested in how consistent the full range of Constraint Diagrams is when examining their syntax and semantics. We will look at the notation and examine all constraints.

### **5.3 Role expressiveness**

This dimension examines how well the purpose of a component, action or symbol is readily inferred. We are interested in how easily the role of a constraint can be picked up from the notation. We will look at a natural language constraint and examine the Constraint Diagrams.

## 5.4 Secondary notation

This dimension examines how extra information can be presented other than in the official syntax of the notation. We are interested in annotations, maybe in diagrammatic form, maybe in textual form that can convey additional meaning to the diagram. We are also interested in whether these annotations can be used in place of the notation to improve the interpretation of the diagram. We also investigate the use of colour as an aid to conveying additional meaning.

## 5.5 The paper questionnaire

We asked a series of ten questions. Each was geared to one dimension, although one dimension may have several questions. Each question could have one of five answers, graded so box one on the left was a low score while box five on the right was a high score. The responses from these questions can be amalgamated to form a profile for the constraint diagrams notation.

Questions one through six are based on the dimension closeness of mapping, question seven relates to the dimension consistency, question eight relates to the dimension role expressiveness, and questions nine and ten relate to the dimension secondary notation.

It must be stated that in all cases the only Object-Oriented modelling notation, other than constraint diagrams, that the participants have had first-hand knowledge and experience of is the UML (i.e. the only modelling language used to develop object-oriented software is the UML with OCL).

## 5.6 A partial cognitive dimensions profile for constraint diagrams

Overall responses from questions one through nine indicate a neutral to positive bias for both groups of participants. Question ten provides an even mix across positive, neutral and negative responses.

The questions asked on the paper questionnaire are:

1. How closely related is the notation to the constraints you are describing?

The group who are familiar with constraint diagrams has no negative bias for question one. They mainly have a positive bias, although there is an indication of neutrality. This is also exhibited with those who are unfamiliar with constraint diagrams. However, there are several participants from the unfamiliar group that have indicated a negative bias to the question.

2. How well does the notation depict the constraints you have chosen?

The responses from question two exhibit a very similar positive/neutral bias, for both familiar and unfamiliar groups, although there is a less negative bias for the group unfamiliar with constraint diagrams.

3. How closely does the concept of a class in this notation map to your expectations of a class represented in other Object-Oriented notations?

Interestingly, question three shows an overall positive/neutral bias as before, but the negative bias from both groups is higher. We have comments from the participants to indicate that the negative bias represents a lack of closeness of mapping for classes to, in this case, a class diagram from the UML. Representing classes as sets (rectangles or ellipses) does not seem to some to follow the usual class notation of a rectangle in UML.

4. How closely does the concept of an object in this notation map to your expectations of an object represented in other Object-Oriented notations?

Question four returns to the usual theme of a high positive/neutral bias, with only a small negative bias, although this does appear to be for both groups. However, comments from both groups have been made that indicate members of both groups are unfamiliar with other modelling notations i.e. the only ones known or used previously are UML and/or constraint diagrams.

5. How closely does the concept of a set of objects in this notation map to your expectations of a set of objects represented in other Object-Oriented notations?

Question five exhibits the usual positive/neutral bias but also displays some slightly negative bias from both groups, although the more negative bias is from the group who are familiar with constraint diagrams. This is itself seems to be a little strange, as the participants (especially those familiar with constraint diagrams) are familiar with logic and set theory, and therefore with Venn-Euler diagrams and representing sets diagrammatically. To then indicate the notation, based on Venn-Euler diagrams, does not represent sets of objects seems counter-intuitive.

6. How closely does the concept of a relation in this notation map to your expectations of a relation represented in other Object-Oriented notations?

Question six returns to the usual theme of a high positive/neutral bias, with only a small negative bias from both groups. Interestingly, we have comments from the participants to indicate the relations depicted in the constraint diagram notation have been likened to the associations in UML class diagrams, with arrows depicting the direction of either the relation or the association.

Questions one through six are based on the dimension closeness of mapping. From the overall positive/neutral bias of the participants, we can deduce that constraints modelled in the constraint diagram notation map closely to the domains they are describing. In effect, the constraints modelled provide an accurate picture of that part (or whole) of the system.

7. When reading the notation how easy is it to tell what each part is for?

Question seven presents an overall positive/neutral bias in identifying the component parts of the notation and for what each part is for. However, we have a small number of participants from both groups who expressed a negative bias. Ironically, it was the group who are familiar with the notation that provided the lowest response to the question i.e. it was difficult to identify the different parts of the notation.



8. How difficult did you find the notation overall to interpret?

(While not part of the cognitive dimensions framework in itself, this question aims to give the participant the ability to present a personal view of how well they believe the notation is to interpret, taking into account the previous questions and also any other personal factors that they may feel are pertinent, but are not previously covered.)

Question eight presents an overall positive/neutral bias overall for a participant's personal view of how difficult or not the constraint diagram notation is to interpret. There are some participants who provided a negative bias i.e. they found the notation difficult to interpret, but most of these were from the group unfamiliar with the notation, so this would be understandable. These participants had a short introduction to the notation immediately prior to the test. Perhaps these participants would have benefitted from a longer exposure.

9. How helpful would notes made in natural language or another notation be in aiding your interpretation of this notation?

Question nine presents an overall positive bias to the use of natural language or other such notation within the notation, perhaps as additional notes to the reader. There is a marked neutral bias, although this differs from all previous questions by being a lot lower than the positive bias, and there is a small negative bias over both groups. While the majority of participants expressed the notion that additional natural language symbols would help with interpretation, it was re-iterated that this depends on what the notes actually said! This, again, returns to the old adage that natural language is too imprecise to fully and precisely model a system.

10. How helpful would adding colour to the notation help your interpretation?

Finally, question ten differs in overall bias from all previous questions. There is, in fact, no bias over positive, negative or neutral; there is instead an even balance across all possible responses for both familiar and unfamiliar participants.

From these responses, we see a pattern emerging. The dimensions used in this case have been chosen for examining the notation itself in isolation, rather than using some artefact, be it software or physical, and we see a trend towards the positive. The dimensions closeness of mapping, consistency and role expressiveness all provide positive feedback, whereas the dimension secondary notation provides a mixed response. Adding notes in another notation e.g. natural language or first order predicate logic are deemed a positive thing, while adding colour to the notation has as much a negative as positive bias, and therefore needs further investigation.

We can describe the profile graphically using a box plot type notation:

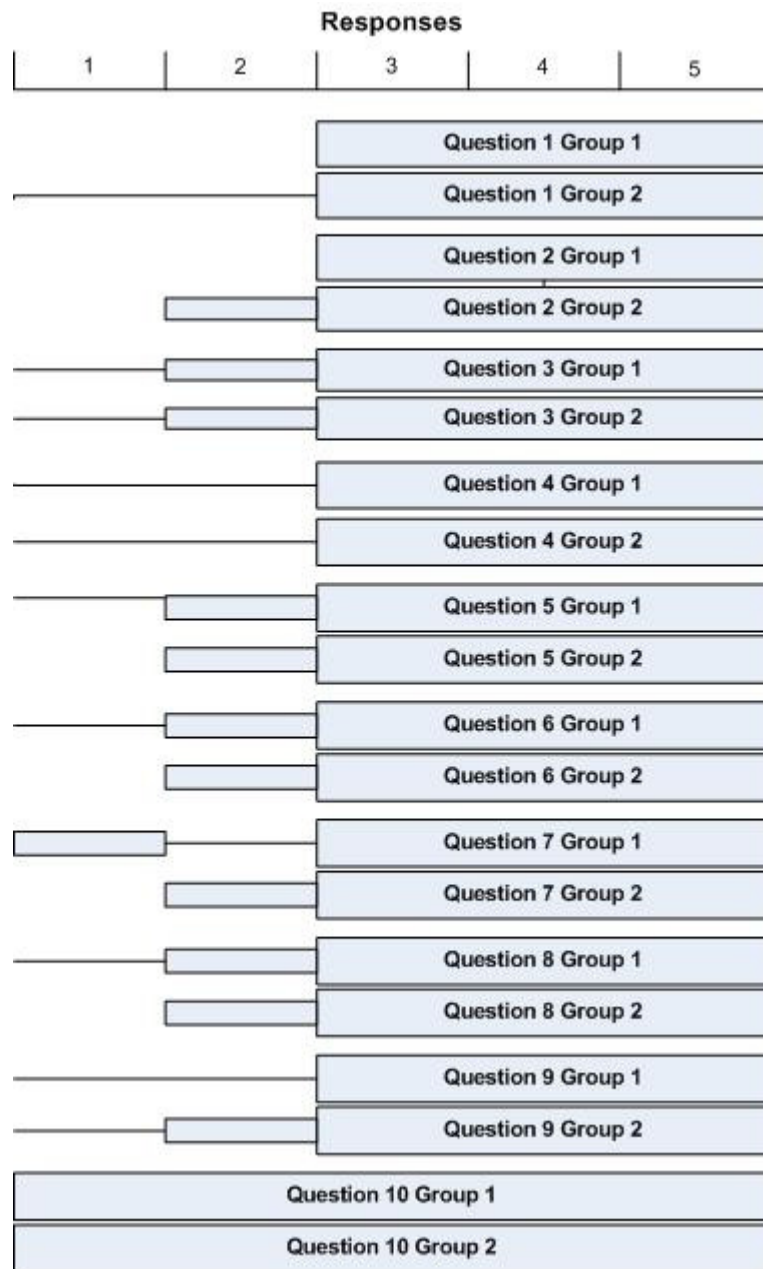


Figure 31: Graphical representation showing the profile of responses from the questionnaire

## 6 Study 2: Developing constraints using constraint diagrams and OCL

This study followed on from the interpretation of constraint diagrams, and examined the ability of software engineers to develop their own models. The focus of the study is to investigate the ability to develop constraints in the Constraint Diagram notation, and compare this to the ability to develop constraints in the OCL notation.

### 6.1 Hypothesis

For this study, we will be testing the following hypothesis:

- In general, software engineers with previous training in OCL and constraint diagrams at an equivalent level will develop constraints more accurately in constraint diagrams than in OCL.

To test the hypothesis we have to look at a number of metrics:

- All relevant objects within the constraint have been included i.e.
  - No new objects have been created and incorporated in the model unless representing primitive data types e.g. Integer, String, Date, Time, etc.
  - No irrelevant objects have been added to the constraint being modelled e.g. modelling a member of staff object in an operation for enrolling students onto modules.
  - Existing objects have not been overlooked.
- All relevant associations have been included i.e.
  - No new associations between objects have been created that do not appear on the class diagram.
  - No irrelevant associations between objects have been added to the constraint being modelled.

- Existing associations between objects have not been overlooked.
- Operation post-conditions specifications are labelled suitably.
  - All objects or sets of objects should be named consistently with the pre-condition.
  - All associations should be named consistently with either the pre-condition (in the case of existing associations) or the operation specification (in the case of a new association).
- All operation post-conditions are clear and easily readable.
  - Clear constraint diagrams or OCL statements should be evident.
  - Use of correct notational syntax should be followed.
  - The correct operation semantics should be evident.

## 6.2 Method

Whereas study one identified two groups of participants (between groups study), this study will require only one group of participants (within groups study), but will actively compare their ability to develop constraint diagrams and OCL. Each participant will have had formal training in both notations i.e. they will have taken either undergraduate study or postgraduate study in rigorous object oriented modelling using both notations, thereby giving an equivalent base knowledge for both notations.

### 6.2.1 The paper-based questionnaire

This study was fully paper-based. It consists of three artefacts; the first being an overview document (see Appendix III), the second being for developing the constraints (see Appendix IV), and the third being a more in-depth multiple choice questionnaire (see Appendix V) based on the cognitive dimensions framework [36, 39].

### **6.2.2 The overview document**

As an aid to preparing the participants for the study, a small document was developed to introduce the Constraint Diagrams and OCL notations (see Appendix III). The document presented the mechanics of the respective notations from the point of view of a Software Engineer who has not used either notation previously. This will give all participants a base knowledge and refresher of both of the notations in question.

### **6.2.3 The constraint development document**

Whereas the interpretation study used a computer-based questionnaire, the development study required the participant to draw diagrams; hence a paper-based document was developed (see Appendix IV). This is a simple document comprising two constraints, one to be completed in the constraint diagram notation, one to be completed in the OCL. The format for each is the same: an operation is specified, a pre-condition is provided, and a post-condition is required to be drawn.

### **6.2.4 The multiple choice questionnaire**

Like the interpretation study, the participants were also asked to complete a multiple choice questionnaire (see Appendix V) based on the cognitive dimensions of notations [36, 39]. However, unlike the interpretation study, which just concentrated on the dimensions Closeness of Mapping, Consistency, Role Expressiveness and Secondary Notation, the development study provided a more thorough multiple choice questionnaire examining all the dimensions in the cognitive dimensions framework.

### **6.2.5 Undertaking the “development” study**

For this study, we chose one group of participants, who had some experience at undergraduate and/or postgraduate level, of both the constraint diagram and OCL notations. Other notations may have been known, but for this study, they were deemed irrelevant. These participants were taken from the students and staff of the University of Brighton.

### **6.2.6 Threats to validity**

We aim to reduce maturation and history effects by presenting two separate operations to model. Rather than present one operation and allow the participant to gain knowledge of the operation (which could then be used to improve the outcome of the second operation), we present each participant with a different operation to model for constraint diagrams and OCL.

This study is not timed, so the effects of time on both the participant and the environment are of no consequence. There are, however, other threats that could significantly affect the results of the study, which we shall discuss further.

The experimental situation itself may threaten the validity of the study. It may be that participants do not enter into the spirit of the study, and decide to draw models that are outside the purview of the study. It may be that participants need to refer back to the overview document throughout the study, to help understand certain notations. This may make the participant feel uneasy about taking the test, especially if the experience of one or both of the notations was gained substantially before taking the test. Problems of this nature are referred to as testing effects, and are one aspect of the more general issue of experimental reactivity. This refers to the fact that participants will often react to features of an experiment so that the process of making observations can change observations [46].

Instrumentation effects refer to the collection of data for the study. Who collects the data, where it is collected and what methods are used to collect the data may all have significant impact on the experiment. To help reduce instrumentation effects, where possible the same instruments will be used in collecting data, e.g. the study will take place in the same room for all participants, the study will be given in the form of a paper questionnaire (this should be familiar to all participants), the same person will be present at all experimental stages, etc.

### **6.2.7 Taking part in the study**

Each group of participants taking the study will be briefed by the experimenter just prior to taking the test. The participants will be presented with a copy of the

questionnaire and asked to review it. Following this, there will be a question and answer session to ensure that the preparatory material was understood.

Immediately before the test begins the confidentiality and voluntary nature of participation will be stressed. Any participant can drop out at any time with no repercussions. However, we will offer a small incentive (refreshments or other such).

### **6.2.8 Taking the test**

Each group of subjects will be exposed to the questionnaire within a teaching room for preference or at a pre-arranged location (e.g. work, home). If the test takes place outside the University at a pre-arranged location, the experimenter will be available either in person or by teleconferencing to provide the briefing (see above), and to see that all test protocols are adhered to.

The test will take a maximum of two hours. In reality it may be completed within an hour, but we will allow for extra time if needed. Upon completion of the questionnaire, each participant will be invited to take refreshments. When all participants have completed, a short de-briefing will be undertaken.

### **6.2.9 Debriefing**

Everyone will be thanked for their participation. A participant's right to confidentiality and that no personal information will be kept regarding the study will also be re-iterated.



## **7 Quantitative analysis of the development of constraint diagram and OCL operations**

Once completed, the results from the study will be examined using various statistical techniques. To discuss these techniques and the results we obtain from them, we will first look at the descriptive statistics.

We present descriptive statistics that tabulate the metrics (as correct or incorrect) and provided rudimentary statistical calculations about them. As well as the number of responses to each metric, we provide the mean responses along with standard error, the standard deviation and variance, as well as skewness or lack of symmetry on the normally distributed curve (a non-zero value indicates a skewness where positive values indicate a left skewed distribution, and negative values indicate a right skewed distribution) – and kurtosis – or the extent to which the distribution departs from a normally distributed curve i.e. how pointed the shape of the curve of the distribution is.

### **7.1 Confidence interval**

“When we estimate parameters we can do one of two things. We can estimate a parameter by suggesting a particular value. This is called a point estimate. However, when we do this we still have a degree of uncertainty about the correspondence between this point estimate and the true parameter value. To deal with this uncertainty we could therefore specify a range of values around the point estimate within which we expect the true value to lie. Such a range of possible values is called a Confidence Interval (CI).” [46]

“The size of the confidence interval depends on the amount of inferential uncertainty that researchers are prepared to accept and the amount of random error present.” [46]

“A 95% CI for the sample mean would contain the true population mean 95% of the time if many samples of the same size as the empirical sample were drawn randomly from the population.” [46]

If we assume a range of 9 to 11 correct metrics (82-100% of the metrics) as being a successful outcome of the study, we have a total of 57 successful participants out of 94, that is, 60.6% of participants were successful. We can calculate a 95% CI (using the calculation defined in [50]) of 50.0% to 70.6%. We can then say that the true percentage of people achieving at least 80% correct answers is likely to be between 50.0% and 70.6%.

We can also calculate the 95% CI for each individual metric. The percentages (95% confidence interval) of participants who obtained correct answers on the individual metrics are given below.

Metric	Number of correct answers	Number of participants	Percentage	CI (from)	CI (to)
1	81	94	86.2	77.5	92.4
2	83	94	88.3	80.0	94.0
3	73	94	77.7	67.9	85.6
4	85	94	90.4	82.6	95.5
5	83	94	88.3	80.0	94.0
6	78	94	83.0	73.8	89.9
7	65	94	69.1	58.8	78.3
8	63	94	67.0	56.6	76.4
9	83	94	88.3	80.0	94.0
10	65	94	69.1	58.8	78.3
11	49	94	52.1	41.6	62.5

**Table 56: Actual Percentage and Confidence Interval**

Metrics one to nine are essentially about the process of modelling with the notations, being whether all objects that should be there are actually there and those that should not are not. The low percentages for metrics 10 and 11 can be explained by the simple fact of examining the nature of these metrics. Metrics 10 and 11 look at syntax and semantics of the operations respectively. While it is relatively easy to make sure only the required objects and associations are shown, especially if a class diagram is also provided, ensuring the syntax and semantics are correct is a lot more difficult.

Generally, participants were successful if at least nine of the metrics were correctly measured.

## 7.2 Summarising the results

To start, we perform descriptive statistics to describe the main features of a collection of data in quantitative terms. Descriptive statistics are distinguished from inductive statistics in that they aim to quantitatively summarize a data set, rather than being used to support statements about the population that the data are thought to represent.

Metrics	N	Sum	%age Correct
1. No new objects	94	81	86%
2. No irrelevant objects	94	83	88%
3. Existing objects not overlooked	94	73	78%
4. No new associations	94	85	90%
5. No irrelevant associations	94	83	88%
6. Existing associations not overlooked	94	78	83%
7. Objects named consistently	94	65	69%
8. Associations names consistently	94	63	67%
9. Clear diagrams or statements	94	83	88%
10. Correct notational syntax	94	65	69%
11. Correct notational semantics	94	49	52%
Valid N (listwise)	94		

**Table 57: Descriptive Statistics for metrics**

Table 57 shows that of eleven metrics used to test development of the constraint diagram and OCL notations, the majority of metrics gave a percentage of correct metrics over 80%, with the highest being 90% for metric 4. We see that regardless of the notation used, metrics one to six (regarding the disposition of objects and associations) provide the highest correct metrics, along with the ability to provide clear diagrams and statements. These can be thought of as the easier metrics to achieve.

Metrics seven, eight and ten are somewhat lower than metrics one to six, being in the range 67% to 69%. This may simply be due to the nature of the study itself.

After all, there was a document provided with the study paper giving the correct syntax rules for each notation. The reduced percentage may only be due to a lack of care on the part of the participants.

Metric eleven provides us with just over half of correct metrics (52%). We are studying the ability to develop constraints in two notations. While these statistics do not separate the metrics into the relevant notation, we can still see that developing the constraints to an adequate level of understanding is a difficult task.

### 7.3 Metric 1

No new objects have been created and incorporated in the model unless representing primitive data types e.g. Integer, String, Date, Time, etc.

**Crosstab**

			C1		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	8	39	47
		% within notation	17.0%	83.0%	100.0%
		% within C1	61.5%	48.1%	50.0%
		% of Total	8.5%	41.5%	50.0%
		Std. Residual	.6	-.2	
	2. OCL	Count	5	42	47
		% within notation	10.6%	89.4%	100.0%
		% within C1	38.5%	51.9%	50.0%
		% of Total	5.3%	44.7%	50.0%
		Std. Residual	-.6	.2	
Total		Count	13	81	94
		% within notation	13.8%	86.2%	100.0%
		% within C1	100.0%	100.0%	100.0%
		% of Total	13.8%	86.2%	100.0%

**Table 58: Crosstab of notation by metric for metric 1**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.803 <sup>a</sup>	1	.370		
Continuity Correction <sup>b</sup>	.357	1	.550		
Likelihood Ratio	.810	1	.368		
Fisher's Exact Test				.552	.276
Linear-by-Linear Association	.795	1	.373		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 6.50.

b. Computed only for a 2x2 table

**Table 59: Chi-Square tests for notation by metric for metric 1**

We have tabulated the results for metric 1 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for creating new objects and incorporating them into the models.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for creating new objects and incorporating them into the models.

P-Value: 0.370 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for creating new objects and incorporating them into the models.

## 7.4 Metric 2

No irrelevant objects have been added to the constraint being modelled e.g. modelling a member of staff object in an operation for enrolling students onto modules.

			C2		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	6	41	47
		% within notation	12.8%	87.2%	100.0%
		% within C2	54.5%	49.4%	50.0%
		% of Total	6.4%	43.6%	50.0%
		Std. Residual	.2	.0	
	2. OCL	Count	5	42	47
		% within notation	10.6%	89.4%	100.0%
		% within C2	45.5%	50.6%	50.0%
		% of Total	5.3%	44.7%	50.0%
		Std. Residual	-.2	.1	
Total	Count	11	83	94	
	% within notation	11.7%	88.3%	100.0%	
	% within C2	100.0%	100.0%	100.0%	
	% of Total	11.7%	88.3%	100.0%	

**Table 60: Crosstab of notation by metric for metric 2**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.103 <sup>a</sup>	1	.748		
Continuity Correction <sup>b</sup>	.000	1	1.000		
Likelihood Ratio	.103	1	.748		
Fisher's Exact Test				1.000	.500
Linear-by-Linear Association	.102	1	.750		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.50.

b. Computed only for a 2x2 table

**Table 61: Chi-Square tests for notation by metric for metric 2**

We have tabulated the results for metric 2 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for adding irrelevant objects to the constraint and incorporating them into the model.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for adding irrelevant objects to the constraint and incorporating them into the model.

P-Value: 0.748 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for adding irrelevant objects to the constraint and incorporating them into the model.



### 7.5 Metric 3

Existing objects have not been overlooked.

			C3		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	15	32	47
		% within notation	31.9%	68.1%	100.0%
		% within C3	71.4%	43.8%	50.0%
		% of Total	16.0%	34.0%	50.0%
		Std. Residual	1.4	-.7	
	2. OCL	Count	6	41	47
		% within notation	12.8%	87.2%	100.0%
		% within C3	28.6%	56.2%	50.0%
		% of Total	6.4%	43.6%	50.0%
		Std. Residual	-1.4	.7	
Total		Count	21	73	94
		% within notation	22.3%	77.7%	100.0%
		% within C3	100.0%	100.0%	100.0%
		% of Total	22.3%	77.7%	100.0%

**Table 62: Crosstab of notation by metric for metric 3**

**Chi-Square Tests**

	Value	Df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	4.967 <sup>a</sup>	1	.026		
Continuity Correction <sup>b</sup>	3.924	1	.048		
Likelihood Ratio	5.097	1	.024		
Fisher's Exact Test				.046	.023
Linear-by-Linear Association	4.914	1	.027		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 10.50.

b. Computed only for a 2x2 table

**Table 63: Chi-Square tests for notation by metric for metric 3**

We have tabulated the results for metric 3 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for overlooking existing objects from the model.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for overlooking existing objects from the model.

P-Value: 0.026 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference between the different notations for overlooking existing objects from the model.

## 7.6 Metric 4

No new associations between objects have been created that do not appear on the class diagram.

			C4		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	4	43	47
		% within notation	8.5%	91.5%	100.0%
		% within C4	44.4%	50.6%	50.0%
		% of Total	4.3%	45.7%	50.0%
		Std. Residual	-.2	.1	
	2. OCL	Count	5	42	47
		% within notation	10.6%	89.4%	100.0%
		% within C4	55.6%	49.4%	50.0%
		% of Total	5.3%	44.7%	50.0%
		Std. Residual	.2	.0	
Total		Count	9	85	94
		% within notation	9.6%	90.4%	100.0%
		% within C4	100.0%	100.0%	100.0%
		% of Total	9.6%	90.4%	100.0%

**Table 64: Crosstab of notation by metric for metric 4**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.123 <sup>a</sup>	1	.726		
Continuity Correction <sup>b</sup>	.000	1	1.000		
Likelihood Ratio	.123	1	.726		
Fisher's Exact Test				1.000	.500
Linear-by-Linear Association	.122	1	.727		
N of Valid Cases	94				

a. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 4.50.

b. Computed only for a 2x2 table

**Table 65: Chi-Square tests for notation by metric for metric 4**

We have tabulated the results for metric 4 showing the given correct or incorrect constraints. As there are two cells with expected count less than 5 we use the Fisher's Exact Test to determine which of the hypotheses (below) we can use. As we are using binary values for the data to examine, if we were to transform the data for this metric using a logarithmic calculation, we would return a zero value for all values, while using a square root transformation would give us values identical to the original non-transformed values.

Null Hypothesis  $H_0$ : There is no difference between the different notations for adding new associations to the constraint and incorporating them into the model.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for adding new associations to the constraint and incorporating them into the model.

P-Value: 1.000 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for adding new associations to the constraint and incorporating them into the model.

## 7.7 Metric 5

No irrelevant associations between objects have been added to the constraint being modelled.

			C5		
			Incorrect	Correct	Total
notation	1. Constraint Diagrams	Count	5	42	47
		% within notation	10.6%	89.4%	100.0%
		% within C5	45.5%	50.6%	50.0%
		% of Total	5.3%	44.7%	50.0%
		Std. Residual	-.2	.1	
	2. OCL	Count	6	41	47
		% within notation	12.8%	87.2%	100.0%
		% within C5	54.5%	49.4%	50.0%
		% of Total	6.4%	43.6%	50.0%
		Std. Residual	.2	.0	
Total	Count	11	83	94	
	% within notation	11.7%	88.3%	100.0%	
	% within C5	100.0%	100.0%	100.0%	
	% of Total	11.7%	88.3%	100.0%	

**Table 66: Crosstab of notation by metric for metric 5**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.103 <sup>a</sup>	1	.748		
Continuity Correction <sup>b</sup>	.000	1	1.000		
Likelihood Ratio	.103	1	.748		
Fisher's Exact Test				1.000	.500
Linear-by-Linear Association	.102	1	.750		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.50.

b. Computed only for a 2x2 table

**Table 67: Chi-Square tests for notation by metric for metric 5**

We have tabulated the results for metric 5 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for adding irrelevant associations to the constraint and incorporating them into the model.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for adding irrelevant associations to the constraint and incorporating them into the model.

P-Value: 0.748 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for adding irrelevant associations to the constraint and incorporating them into the model.

## 7.8 Metric 6

Existing associations between objects have not been overlooked.

			C6		
			Incorrect	Correct	Total
notation	1. Constraint Diagrams	Count	7	40	47
		% within notation	14.9%	85.1%	100.0%
		% within C6	43.8%	51.3%	50.0%
		% of Total	7.4%	42.6%	50.0%
		Std. Residual	-.4	.2	
	2. OCL	Count	9	38	47
		% within notation	19.1%	80.9%	100.0%
		% within C6	56.3%	48.7%	50.0%
		% of Total	9.6%	40.4%	50.0%
		Std. Residual	.4	-.2	
Total		Count	16	78	94
		% within notation	17.0%	83.0%	100.0%
		% within C6	100.0%	100.0%	100.0%
		% of Total	17.0%	83.0%	100.0%

**Table 68: Crosstab of notation by metric for metric 6**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.301 <sup>a</sup>	1	.583		
Continuity Correction <sup>b</sup>	.075	1	.784		
Likelihood Ratio	.302	1	.583		
Fisher's Exact Test				.785	.392
Linear-by-Linear Association	.298	1	.585		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 8.00.

b. Computed only for a 2x2 table

**Table 69: Chi-Square tests for notation by metric for metric 6**

We have tabulated the results for metric 6 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for overlooking existing associations from the model.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for overlooking existing associations from the model.

P-Value: 0.583 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for overlooking existing associations from the model.



## 7.9 Metric 7

All objects or sets of objects should be named consistently with the pre-condition.

			C7		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	18	29	47
		% within notation	38.3%	61.7%	100.0%
		% within C7	62.1%	44.6%	50.0%
		% of Total	19.1%	30.9%	50.0%
		Std. Residual	.9	-.6	
	2. OCL	Count	11	36	47
		% within notation	23.4%	76.6%	100.0%
		% within C7	37.9%	55.4%	50.0%
		% of Total	11.7%	38.3%	50.0%
		Std. Residual	-.9	.6	
Total		Count	29	65	94
		% within notation	30.9%	69.1%	100.0%
		% within C7	100.0%	100.0%	100.0%
		% of Total	30.9%	69.1%	100.0%

**Table 70: Crosstab of notation by metric for metric 7**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	2.444 <sup>a</sup>	1	.118		
Continuity Correction <sup>b</sup>	1.795	1	.180		
Likelihood Ratio	2.462	1	.117		
Fisher's Exact Test				.180	.090
Linear-by-Linear Association	2.418	1	.120		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 14.50.

b. Computed only for a 2x2 table

**Table 71: Chi-Square tests for notation by metric for metric 7**

We have tabulated the results for metric 7 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for consistently naming all objects or sets of objects with the pre-condition.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for consistently naming all objects or sets of objects with the pre-condition.

P-Value: 0.118 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for consistently naming all objects or sets of objects with the pre-condition.

## 7.10 Metric 8

All associations should be named consistently with either the pre-condition (in the case of existing associations) or the operation specification (in the case of a new association).

			C8		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	23	24	47
		% within notation	48.9%	51.1%	100.0%
		% within C8	74.2%	38.1%	50.0%
		% of Total	24.5%	25.5%	50.0%
		Std. Residual	1.9	-1.3	
	2. OCL	Count	8	39	47
		% within notation	17.0%	83.0%	100.0%
		% within C8	25.8%	61.9%	50.0%
		% of Total	8.5%	41.5%	50.0%
		Std. Residual	-1.9	1.3	
Total	Count	31	63	94	
	% within notation	33.0%	67.0%	100.0%	
	% within C8	100.0%	100.0%	100.0%	
	% of Total	33.0%	67.0%	100.0%	

**Table 72: Crosstab of notation by metric for metric 8**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	10.829 <sup>a</sup>	1	.001		
Continuity Correction <sup>b</sup>	9.434	1	.002		
Likelihood Ratio	11.178	1	.001		
Fisher's Exact Test				.002	.001
Linear-by-Linear Association	10.714	1	.001		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 15.50.

b. Computed only for a 2x2 table

**Table 73: Chi-Square tests for notation by metric for metric 8**

We have tabulated the results for metric 8 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for consistently naming all associations with the pre-condition or operation specification.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for consistently naming all associations with the pre-condition or operation specification.

P-Value: 0.001 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference between the different notations for consistently naming all associations with the pre-condition or operation specification.

### 7.11 Metric 9

Clear constraint diagrams or OCL statements should be evident.

<b>Crosstab</b>					
			C9		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	8	39	47
		% within notation	17.0%	83.0%	100.0%
		% within C9	72.7%	47.0%	50.0%
		% of Total	8.5%	41.5%	50.0%
		Std. Residual	1.1	-.4	
	2. OCL	Count	3	44	47
		% within notation	6.4%	93.6%	100.0%
		% within C9	27.3%	53.0%	50.0%
		% of Total	3.2%	46.8%	50.0%
		Std. Residual	-1.1	.4	
Total	Count	11	83	94	
	% within notation	11.7%	88.3%	100.0%	
	% within C9	100.0%	100.0%	100.0%	
	% of Total	11.7%	88.3%	100.0%	

**Table 74: Crosstab of notation by metric for metric 9**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	2.574 <sup>a</sup>	1	.109		
Continuity Correction <sup>b</sup>	1.647	1	.199		
Likelihood Ratio	2.660	1	.103		
Fisher's Exact Test				.198	.099
Linear-by-Linear Association	2.547	1	.111		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5.50.

b. Computed only for a 2x2 table

**Table 75: Chi-Square tests for notation by metric for metric 9**

We have tabulated the results for metric 9 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for drawing clear constraint diagrams or writing clear OCL statements.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for drawing clear constraint diagrams or writing clear OCL statements.

P-Value: 0.109 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for drawing clear constraint diagrams or writing clear OCL statements.

## 7.12 Metric 10

Use of correct notational syntax should be followed.

			C10		
			Incorrect	Correct	Total
notation	1. Constraint Diagrams	Count	10	37	47
		% within notation	21.3%	78.7%	100.0%
		% within C10	34.5%	56.9%	50.0%
		% of Total	10.6%	39.4%	50.0%
		Std. Residual	-1.2	.8	
	2. OCL	Count	19	28	47
		% within notation	40.4%	59.6%	100.0%
		% within C10	65.5%	43.1%	50.0%
		% of Total	20.2%	29.8%	50.0%
		Std. Residual	1.2	-.8	
Total		Count	29	65	94
		% within notation	30.9%	69.1%	100.0%
		% within C10	100.0%	100.0%	100.0%
		% of Total	30.9%	69.1%	100.0%

**Table 76: Crosstab of notation by metric for metric 10**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	4.039 <sup>a</sup>	1	.044		
Continuity Correction <sup>b</sup>	3.192	1	.074		
Likelihood Ratio	4.090	1	.043		
Fisher's Exact Test				.073	.037
Linear-by-Linear Association	3.996	1	.046		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 14.50.

b. Computed only for a 2x2 table

**Table 77: Chi-Square tests for notation by metric for metric 10**

We have tabulated the results for metric 10 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for following the correct notational syntax.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for following the correct notational syntax.

P-Value: 0.044 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference between the different notations for following the correct notational syntax.



### 7.13 Metric 11

The correct operation semantics should be evident.

			C11		Total
			Incorrect	Correct	
notation	1. Constraint Diagrams	Count	26	21	47
		% within notation	55.3%	44.7%	100.0%
		% within C11	57.8%	42.9%	50.0%
		% of Total	27.7%	22.3%	50.0%
		Std. Residual	.7	-.7	
	2. OCL	Count	19	28	47
		% within notation	40.4%	59.6%	100.0%
		% within C11	42.2%	57.1%	50.0%
		% of Total	20.2%	29.8%	50.0%
		Std. Residual	-.7	.7	
Total		Count	45	49	94
		% within notation	47.9%	52.1%	100.0%
		% within C11	100.0%	100.0%	100.0%
		% of Total	47.9%	52.1%	100.0%

**Table 78: Crosstab of notation by metric for metric 11**

**Chi-Square Tests**

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	2.089 <sup>a</sup>	1	.148		
Continuity Correction <sup>b</sup>	1.535	1	.215		
Likelihood Ratio	2.097	1	.148		
Fisher's Exact Test				.215	.108
Linear-by-Linear Association	2.067	1	.151		
N of Valid Cases	94				

a. 0 cells (.0%) have expected count less than 5. The minimum expected count is 22.50.

b. Computed only for a 2x2 table

**Table 79: Chi-Square tests for notation by metric for metric 11**

We have tabulated the results for metric 11 showing the given correct or incorrect constraints. As there are no cells with expected count less than 5 we need not transform the table, and can use the Pearson Chi-Square Test to determine which of the hypotheses (below) we can use.

Null Hypothesis  $H_0$ : There is no difference between the different notations for providing the correct operation semantics.

Alternative Hypothesis  $H_1$ : There is a difference between the different notations for providing the correct operation semantics.

P-Value: 0.148 (greater than 0.05 or 5%)

We fail to reject  $H_0$ ; there is no significant difference between the different notations for providing the correct operation semantics.

### 7.14 Hypothesis testing and inference

We will use inferential statistics to infer conclusions about the whole population of software engineers who are modelling using constraint diagrams and OCL. We will perform tests for normality and provide independent sample tests. We will then use the Mann-Whitney Test to help us understand the modelling behaviour of the whole population of software engineers.

### 7.15 Hypothesis tests of differences between groups

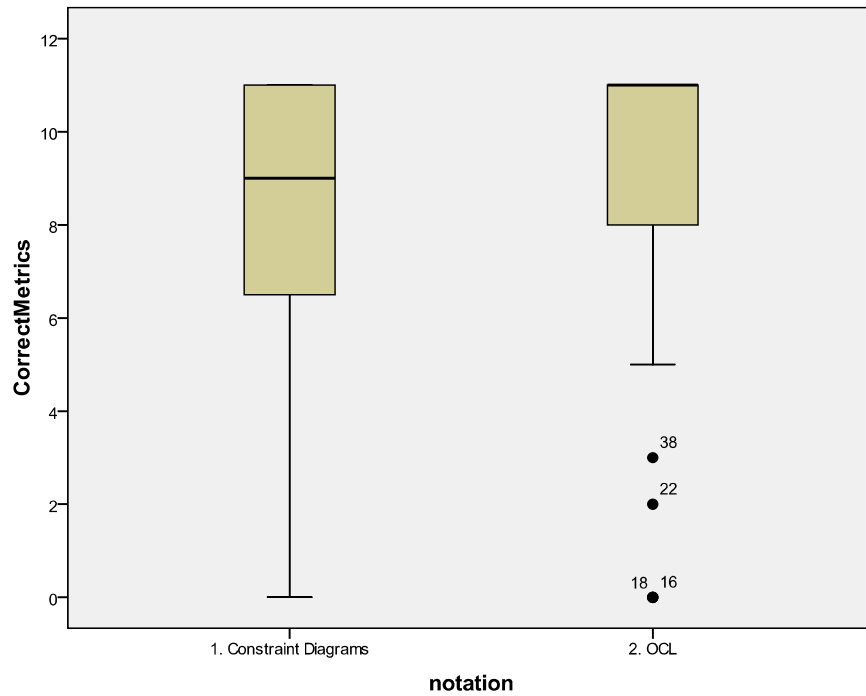
We examine the correct and incorrect metrics to determine if they are normally distributed. We are looking at the metrics against each notation i.e. constraint diagrams and OCL.

notation		Kolmogorov-Smirnov <sup>a</sup>			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
CorrectMetrics	1. Constraint Diagrams	.212	47	.000	.827	47	.000
	2. OCL	.311	47	.000	.681	47	.000
IncorrectMetrics	1. Constraint Diagrams	.212	47	.000	.827	47	.000
	2. OCL	.311	47	.000	.681	47	.000

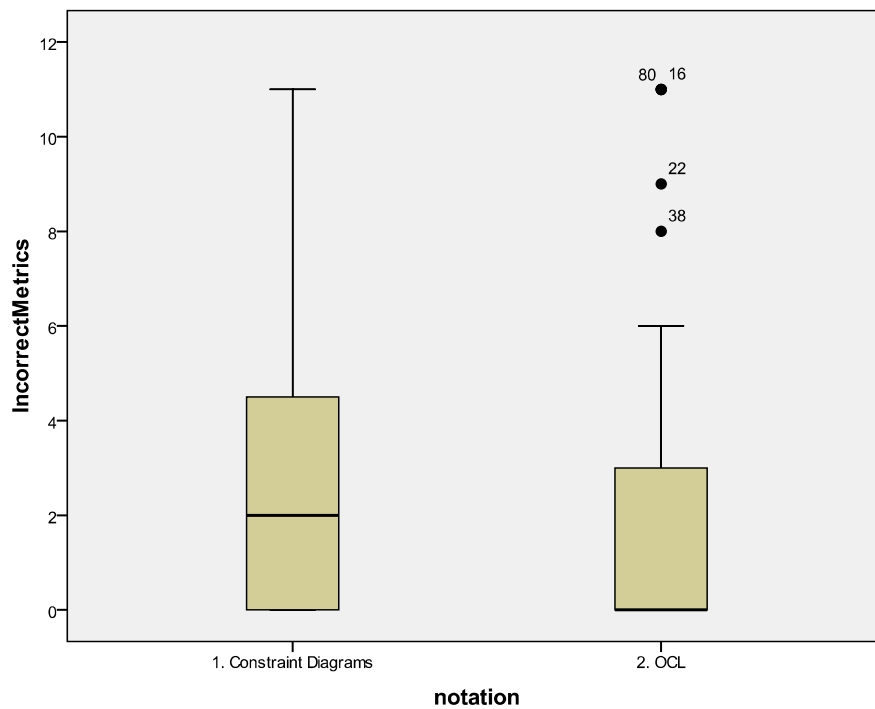
a. Lilliefors Significance Correction

**Table 80: Tests for Normality for responses to all 11 metrics combined**

Looking at the calculated Shapiro-Wilk significance for all groups, we see the p-value is lower than 0.05 or 5%. This indicates a non-normal distribution for the samples. This result can be indicated graphically with the following box-plots.



**Figure 32: Box plot of correct metrics by notation**



**Figure 33: Box plot of incorrect metrics by notation**

The correct metrics are non-normal and are left-skewed, while the incorrect metrics are non-normal and are right skewed. As we have identified the distribution as non-

normal, we will use non-parametric tests (in this case the Mann-Whitney Test) to infer over a larger population i.e. all software engineers who develop models using constraint diagrams and OCL.

	N	Mean	Std. Deviation	Minimum	Maximum	Percentiles		
						25th	50th (Median)	75th
CorrectMetrics	94	8.60	3.119	0	11	7.00	10.00	11.00
IncorrectMetrics	94	2.40	3.119	0	11	.00	1.00	4.00
notation	94	1.50	.503	1	2	1.00	1.50	2.00

**Table 81: Non-Parametric tests (descriptive statistics) for Mann-Whitney test**

notation		N	Mean Rank	Sum of Ranks
CorrectMetrics	1. Constraint Diagrams	47	41.81	1965.00
	2. OCL	47	53.19	2500.00
	Total	94		
IncorrectMetrics	1. Constraint Diagrams	47	53.19	2500.00
	2. OCL	47	41.81	1965.00
	Total	94		

**Table 82: Mann-Whitney Test (non-parametric) Ranks**

	CorrectMetrics	IncorrectMetrics
Mann-Whitney U	837.000	837.000
Wilcoxon W	1965.000	1965.000
Z	-2.117	-2.117
Asymp. Sig. (2-tailed)	.034	.034

a. Grouping Variable: notation

**Table 83: Mann-Whitney Test (non-parametric) Test Statistics**

Report			
notation		CorrectMetrics	IncorrectMetrics
1. Constraint Diagrams	Mean	8.23	2.77
	N	47	47
	Std. Deviation	2.994	2.994
	Median	9.00	2.00
	Minimum	0	0
	Maximum	11	11
2. OCL	Mean	8.96	2.04
	N	47	47
	Std. Deviation	3.230	3.230
	Median	11.00	.00
	Minimum	0	0
	Maximum	11	11
Total	Mean	8.60	2.40
	N	94	94
	Std. Deviation	3.119	3.119
	Median	10.00	1.00
	Minimum	0	0
	Maximum	11	11

**Table 84: Mann-Whitney Test (non-parametric) Means and Medians**

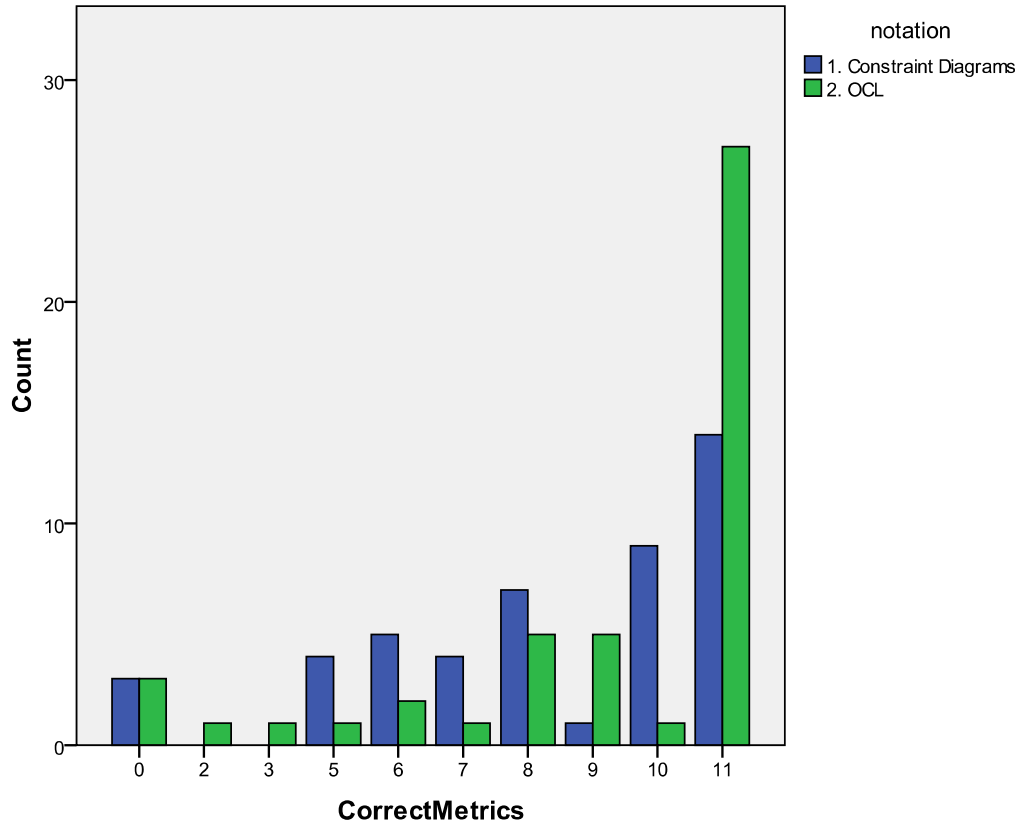
Mann-Whitney test for correct vs incorrect metrics:

Null Hypothesis  $H_0$ : There is no difference between the notations in the level of correct or incorrect metrics.

Alternative Hypothesis  $H_1$ : There is a difference between the notations in the level of correct or incorrect metrics.

P-Value: 0.034 (less than 0.05 or 5%)

We reject  $H_0$ ; there is a significant difference between the notations in the level of correct or incorrect metrics.



**Figure 34: Bar graph representing correct metrics by notation**

Examining figure 34, we can see that the constraint diagram notation displays an overall greater propensity for correct metrics when compared to OCL, when there is at least one incorrect metric experienced. However, OCL provides a greater number of counts when all eleven metrics are correct.

## **8 Conclusions gained from the quantitative analysis**

Following on from the quantitative analysis of the 11 metrics, we will now discuss our findings and draw conclusions as to whether the constraint diagram notation is more readily developed by software engineers than is OCL.

### **8.1 The conduct of the study: sampling over two years**

Overall, 47 students took part in this study, with 17 taking part in 2008-09 and 30 taking part in 2009-10. The participants were taken from the BSc (Hons) Computer Science degree course taking the Rigorous Object Oriented Modelling (ROOM) module. All possible care was taken to achieve identical study conditions for both academic years.

During the study, the ROOM module remained unchanged, with both content and lecturers (and therefore, lecturing styles) being common to both years of study. The participants were presented with the study at the end of the module teaching time as a revision mechanism. Both years were provided with an identical reference paper on both Constraint Diagrams and OCL. The set of questions from which the study papers were taken were also identical.

Thus, while combining participants from two years might have been seen as problematic from a methodological standpoint, in fact we are confident that the two samples were identical in all important points.

### **8.2 The findings of the descriptive statistics**

Examining all the crosstabs, we can posit that for most metrics, participants are able to correctly model operations regardless of the notation. Those metrics that showed any level of incorrect ability may be such that if the participants showed more care in their drawings, this may not be the case.

Please note: there are three participants who have returned blank papers. These are still included within the statistics as there is no indication that this is not the participants' valid response to the study.



Metric one shows that over 86% of participants were able to develop the operations in both notations without adding new objects that weren't on the UML class diagrams or the operation specifications provided. Looking at each individual notation, OCL fared better with over 89% as opposed to constraint diagrams with only 83%.

Interestingly, all those who created objects that did not appear on the class diagram treated associations as objects. As an illustration using the constraint diagram notation, take the pre-condition:

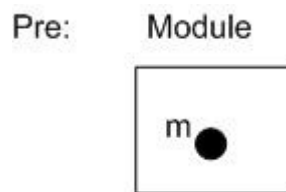


Figure 35: Pre-condition for operation `Module:SetPrerequisite(m→p)`

One correct interpretation of the post-condition would be:

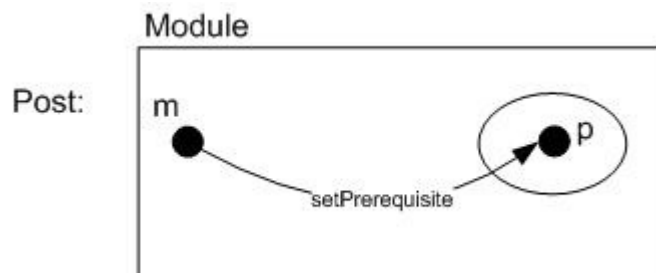


Figure 36: Correct post-condition for operation `Module:SetPrerequisite(m→p)`

Whereas the participants who incorrectly drew the operation did so using the diagram:

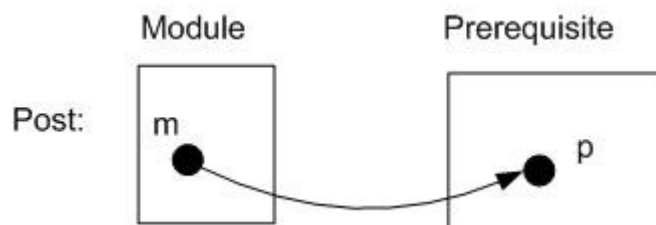


Figure 37: Incorrect post-condition for operation `Module:SetPrerequisite(m→p)`

This shows the *setPrerequisite* association as being a separate object *Prerequisite* from the original object *Module*. It is incorrect in that prerequisite modules are treated as separate from the module *m*, but are in fact a subset of the set *Module*. If the set bounding box for module had been removed, or had encapsulates the diagram fully, this could be deemed a correct interpretation of the post-condition. The same scenario was prevalent for incorrect OCL operations as well.

Metric two shows that over 88% of participants were able to develop the operations in both notations without adding irrelevant objects from the UML class diagrams or the operation specifications provided. Looking at each individual notation, OCL fared only slightly better with over 89% as opposed to constraint diagrams with over 87%. For incorrect models, unlike for metric one where there seems to be a lack of care, there appears also to have been a lack of understanding of the notation. All errors are different, with no two identical. Several can be attributed to lack of care in the development of the model, this may be because the participant may simply be unwilling to put the effort into correctly modelling the operations, or it may simply be a lack of understanding.

Metric three shows that over 77% of participants were able to develop the operations in both notations without overlooking existing objects from the UML class diagrams or the operation specifications provided. As with metric one, there was a core of participants who incorrectly overlooked existing objects and treated the associations as those missing objects. If we examine further, we find that with the sample of participants taking this study, there is a significant difference between the different notations when it comes to overlooking existing objects from the model being developed. If we break down the percentages of correct metrics per notation we see that, while other metrics have a maximum difference of approx. 10% and more likely a 2% to 5% difference, we see a difference of almost 20% for metric three. The numbers of constraint diagrams where existing objects have not been overlooked are significantly lower i.e. 68.1% whereas the OCL appears to remain at a par with other metrics i.e. 87.2%. This would seem to indicate that either due care in drawing constraint diagrams was not taken, or constraint diagrams are harder to draw than OCL statements.

Metric four shows that over 90% of participants were able to develop the operations in both notations without adding new associations between objects or sets of objects from the UML class diagrams or the operation specifications provided. Looking at each individual notation, constraint diagrams fared better with over 91% as opposed to OCL with over 89%. For incorrect models, there appears to have been a lack of understanding of the appropriate operation as well as the notation. With the constraint diagram notation, the participant in error has introduced objects and associations that have no bearing on the operation whatsoever.

Metric five shows that over 88% of participants were able to develop the operations in both notations without adding irrelevant associations between objects or sets of objects from the UML class diagrams or the operation specifications provided. Looking at each individual notation, constraint diagrams fared better with over 89% as opposed to OCL with just over 87%. For incorrect models, there appears to have been a general lack of understanding of the notation in question as general syntax is incorrect, although in one case an understanding of the notion of modelling in general may have been lacking. On the constraint diagram, the pre-condition itself was modified by the participant to enable the post-condition to appear workable.

Metric six shows that over 83% of participants were able to develop the operations in both notations without overlooking existing associations between objects or sets of objects from the UML class diagrams or the operation specifications provided. Looking at each individual notation, constraint diagrams fared better with just over 85% as opposed to OCL with over 80%. For incorrect models there appears to have been a lack of care when developing the model (all errors are different, with no two identical). This may be because the participant may simply be unwilling to put the effort into correctly modelling the operations, or it may simply be a lack of understanding.

Metric seven shows that over 69% of participants were able to develop the operations in both notations where all objects and sets of objects are named consistently with the UML class diagrams or the operation specifications provided. Looking at each individual notation, OCL fared better with over 76% as opposed to constraint diagrams with over 61%. For incorrect models, there appears to be a general lack of care when naming the objects consistently in about half of cases,

while several have provided a correct interpretation of the model and then crossed out and provided an invalid interpretation instead. The remainder appear to show a lack of understanding of modelling techniques with the notations involved.

Metric eight shows that over 67% of participants were able to develop the operations in both notations where all associations are named consistently with the UML class diagrams or the operation specifications provided. If we examine further, we find that with the sample of participants taking this study, there is a significant difference between the different notations when it comes to naming associations consistently with the pre-condition or the operation specification. If we break down the percentages of correct metrics per notation we see that, while other metrics have a maximum difference of approx. 10% and more likely a 2% to 5% difference, we see a difference of almost 22% for metric eight. The numbers of constraint diagrams where associations are not named consistently are significantly lower i.e. 51.1% whereas the OCL appears to remain at a par with other metrics i.e. 83.0%. This would seem to indicate that due care in drawing constraint diagrams was not taken, as for all diagrams in error the existing association names are not given.

Metric nine shows that over 88% of participants were able to develop the operations in both notations providing clear and concise constraint diagrams or OCL statements. Looking at each individual notation, OCL fared better with over 93% as opposed to constraint diagrams with over 83%. This is to be expected, as OCL requires a hand-written statement using non-mathematical symbolism i.e. the use of logic described in natural language. Constraint diagrams are by their very nature diagrams, and if drawn quickly or inexpertly, can lose clarity.

Metric ten shows that over 69% of participants were able to develop the operations in both notations using the correct notational syntax. If we examine further, we find that with the sample of participants taking this study, there is a significant difference between the different notations when it comes to overlooking existing objects from the model being developed. If we break down the percentages of correct metrics per notation we see that, while other metrics have a maximum difference of approx. 10% and more likely a 2% to 5% difference, we see a difference of almost 19% for metric ten. The numbers of OCL statements that do not use the correct notational syntax are significantly lower than those of constraint diagrams, with OCL having a 59.6%

correctly modelled syntactically, whereas constraint diagrams have a 78.7% correctly modelled syntactically.

This would be borne out by a comparison of the syntax for each notation. Constraint diagrams are a notation with a relatively sparse syntax. There are only a small number of ways to describe a set i.e. a rectangle or ellipse either of which could be shaded to indicate an empty set. There are only a small number of spiders describing elements in set i.e. universal spider, existential spider and constant spider. There are only a small number of connectors i.e. tie, strand and arrow. This is the basis syntax for modelling constraints using constraint diagrams.

Conversely, OCL has a rich syntax, describing primitive and complex types e.g. Boolean, Integer, Real, String, Class, etc; structured types e.g. Set, OrderedSet, Bag, Sequence, Collection, Tuple, etc; Type hierarchies e.g. for type T, OclVoid, Oclinvalid  $\leq$  T, Integer  $\leq$  Real, etc. On top of this type syntax we have operations e.g. for Boolean operations: a and b, a or b, a xor b, not a, a = b, a  $\langle$  b, a implies b, if a then b1 else b2 endif. We have operations on String types e.g. s.concat(s1), s.size(), s.toUpper(), s.toLower(), etc. These are basic syntactic structures for modelling using a textual notation; there are many others. As we can see, this richness of syntax lends itself readily to mistakes unless the notation has been used regularly and the user is intimately familiar with the notation.

Metric eleven shows that over 52% of participants were able to develop the operations in both notations using the correct operational semantics. Looking at each individual notation, OCL fared better with over 59% as opposed to constraint diagrams with over 44%.

For incorrect models, there appears to be a general lack of care when describing the semantics of the model, especially on the OCL models. In a lot of cases, the OCL statement in the pre-condition would give an indication of the statement in the post-condition. Participants have either misread the pre-condition or misread the attached literature on OCL syntax and semantics as simple errors creep in regularly. This is more pronounced in constraint diagrams although this appears to be more through a misunderstanding of the underlying set theory required for modelling with constraint diagrams.

Overall, OCL gains a small but significant edge when we examine to see if all relevant objects are included (metrics one, two and three). However, constraint diagrams tend to gain the edge where all relevant associations are examined (metrics four, five and six). When we examine the models to ensure the post-conditions are labelled suitably, OCL appears to once again have the upper hand (metrics seven and eight).

### 8.3 The findings of the inferential statistics

We are trying to infer conclusions about the whole population of software engineers and their ability to model in both OCL and constraint diagrams. We are testing the hypothesis “*In general, software engineers with previous training in OCL and constraint diagrams at an equivalent level will develop constraints more accurately in constraint diagrams than in OCL.*”. We perform tests for normality, resulting in a non-normal distribution, so we use a Mann-Whitney Test (non-parametric).

Tests for the distribution of the samples were given using both the Kolmogorov-Smirnov and Shapiro-Wilk tests. For the benefits of the analysis of this study we will use the Shapiro-Wilk test only as it is a more accurate test. Both correct and incorrect metrics show a distribution that deviates from a normal distribution; correct metrics show a left-skewed distribution, while incorrect metrics show a right-skewed distribution.

As the groups show a skewed distribution, we cannot perform a t-test. Instead, we must use a non-parametric equivalent. The Mann-Whitney test was used to determine whether the two groups of participants (familiar and unfamiliar) are drawn from the same distribution.

The Mann-Whitney test assigns a ranking based on the number of responses of the dependent variable (in this case the correct or incorrect metrics), against the independent variable (in this case the notation used). For the correct metrics, the assigned ranks for the constraint diagram notation are lower than those of the OCL notation. For the incorrect metrics, we predictably find the inverse to be evident. We also find that the calculated P-values for each group are below the 5% threshold, being 3.4% for both correct and incorrect metrics. With this sample, therefore, there

is evidence of a significant difference between the different notations. We can therefore assume the two groups are from different distributions.

Overall, taking the metrics covered in this study, we can posit that the hypothesis: *“In general, software engineers with previous training in OCL and constraint diagrams at an equivalent level will develop constraints more accurately in constraint diagrams than in OCL.”* does not, in fact, hold; OCL appears to provide a better mechanism for developing constraints.

## **9 Qualitative analysis of the development of constraint diagram and OCL operations**

A paper questionnaire was presented to each participant (see Appendix V). This consisted of multiple choice questions that reflects all dimensions from the Cognitive Dimensions of Notations framework [36, 39].

### **9.1 The paper questionnaire**

We asked a series of 20 overall questions. Each was geared to one dimension, although one dimension may have several questions. Each question could have one of five answers, graded so box one on the left was a low score while box five on the right was a high score. In several cases, one or two further questions were asked for clarification of the main question. The responses from these questions can be amalgamated to form a complete profile for the constraint diagrams notation.

It must be stated that in all cases the only Object-Oriented modelling notation, other than constraint diagrams, that the participants have had first-hand knowledge and experience of is the UML (i.e. the only modelling language used to develop object-oriented software is the UML with OCL).

### **9.2 A full cognitive dimensions profile for constraint diagrams**

Overall responses from all questions indicate a generally neutral bias for both notations. There are interesting clusters within each notation that show an overall negative bias for constraint diagrams and an overall positive bias for OCL.

The questions asked on the paper questionnaire are:

1. How easy is it to see or find the various parts of the notation while you were developing the operations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a bias towards the notation not being easy to see.



Participants developing constraints in the OCL notation provided an overall neutral response, with a bias towards the notation being easy to see.

2. When you need to change your operation, how easy is it to make that change?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with no particular bias towards the notation being easy or difficult to make changes.

Participants developing constraints in the OCL notation provided an overall neutral response, with a bias towards the notation being easy to make changes.

3. Is the notation clear and concise?

Participants developing constraints in the constraint diagram notation provided an overall neutral to not clear and concise bias, with only a small response towards being clear and concise.

Participants developing constraints in the OCL notation provided an overall unclear and not concise response, with a small bias towards the notation being clear and concise.

4. How much mental effort did you need to develop the operations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a slight bias towards having to use a lot of mental effort to develop the diagrams.

Participants developing constraints in the OCL notation provided an overall neutral response, with no particular bias towards the either having to use a little or a lot of mental effort to develop the diagrams.

4.1. Did this task come easy to you?

Those who provided responses for constraint diagrams mostly stated that the task did not come easy, mainly due to the fact that they were still learning the notation and its uses. One participant failed to respond to the question, and one indicated the task did come easy, but provided no clarification.

Conversely, those who provided responses for OCL mostly stated that the task did come easy, although there were no comments of clarification. Two participants indicated the task did not come easy, although one did comment on the constraint diagram instead of the OCL and one commented that the function was ill-defined, although this participant did attempt to re-model the UML class diagram and the pre-condition.

5. Was it complex or difficult to develop these operations in your head?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with no particular bias towards the either finding the development of the operations either difficult or not difficult to develop in the participants' head.

Participants developing constraints in the OCL notation provided an overall neutral response, with a small bias towards the finding the development of the operations not difficult to develop in the participants' head.

5.1. Would you have preferred extra space to develop these operations on paper?

All participants, regardless of the notation used, unanimously stated that there was no preference for extra space. There was enough paper to develop the operation already.

6. Did you think it was easy to make common mistakes while developing the operations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a bias towards the notation being easy to make common mistakes.

Participants developing constraints in the OCL notation provided an overall neutral response, with a bias towards the notation being easy to make common mistakes.

6.1. Did you find yourself making small slips that you felt were irritating?

Those who provided responses for constraint diagrams were split almost evenly between those who made small irritating mistakes and those who did not.

Conversely, those who provided responses for OCL mostly stated that they did not make small irritating mistakes.

7. How closely related is the notation to the constraints you are describing?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a strong bias towards the notation not being close to the constraints described.

Participants developing constraints in the OCL notation provided an overall neutral response, with a slight bias towards the notation being close to the constraints described.

8. How well does the notation depict the constraints you have chosen?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a slight bias towards the idea that the notation does not depict the constraints.

Participants developing constraints in the OCL notation provided an overall neutral response, with a strong bias towards the idea that the notation does not depict the constraints.

9. How closely does the concept of a class in this notation map to your expectations of a class represented in other Object-Oriented notations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a strong bias towards the idea that the notation provides the concept of a class, and this can be mapped to classes represented in other notations.

Participants developing constraints in the OCL notation provided an overall neutral response, with a slight bias towards the idea that the notation provides the

concept of a class, and this can be mapped to classes represented in other notations; although there was one participant who did not think the concept of classes in OCL mapped to other notations that depicted classes.

10. How closely does the concept of an object in this notation map to your expectations of an object represented in other Object-Oriented notations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response towards the idea that the notation provides the concept of an object, and this can be mapped to objects represented in other notations.

Participants developing constraints in the OCL notation provided a strong positive response, with only a slight bias towards the idea that the notation does not provide the concept of an object, and this can be mapped to objects represented in other notations.

11. How closely does the concept of a set of objects in this notation map to your expectations of a set of objects represented in other Object-Oriented notations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a strong bias towards the idea that the notation provides the concept of a set of objects, and this can be mapped to sets of objects represented in other notations.

Participants developing constraints in the OCL notation provided an overall neutral response, with a bias towards the idea that the notation provides the concept of a set of objects, and this can be mapped to sets of objects represented in other notations.

12. How closely does the concept of a relation in this notation map to your expectations of a relation represented in other Object-Oriented notations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a strong bias towards the idea that the notation provides the concept of a relation, and this can be mapped to a relation represented in other notations.

Participants developing constraints in the OCL notation provided an overall neutral response, with a slight bias towards the idea that the notation provides the concept of a relation, and this can be mapped to a relation represented in other notations.

13. When reading the notation how easy is it to tell what each part is for?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a strong bias towards the idea that the notation does not provide a mechanism for easily identifying what each part is for.

Participants developing constraints in the OCL notation provided an overall positive response, with a slight bias towards the idea that the notation does not provide a mechanism for easily identifying what each part is for.

13.1. Were there some parts that were particularly difficult to interpret?

Those who provided responses for constraint diagrams were split almost evenly between those who thought that some parts of the notation were particularly difficult to interpret, and those who thought that the notation was not difficult to interpret.

Conversely, those who provided responses for OCL unanimously stated that they thought that the notation was not difficult to interpret.

13.2. Were there parts of the notation you used that you didn't know what they meant?

Those who provided responses for constraint diagrams mostly stated that they understood all parts of the notation, with the exception of one participant who stated they did not know what parts of the notation meant.

Conversely, those who provided responses for OCL unanimously stated that they understood all parts of the notation.

14. How difficult did you find the notation overall to interpret?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with no particular bias towards the notation being easy or difficult to interpret.

Participants developing constraints in the OCL notation provided an overall positive response, with only a slight neutral bias towards the notation being easy or difficult to interpret.

15. How helpful would notes made in natural language or another notation be in aiding your interpretation of this notation?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a positive bias towards the use of a secondary notation in a natural language or other formal notation being helpful in aiding the interpretation of constraint diagrams.

Participants developing constraints in the OCL notation provided an overall positive response, with a neutral bias towards the use of a secondary notation in a natural language or other formal notation being helpful in aiding the interpretation of OCL statements.

16. How helpful would adding colour to the notation help your interpretation?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a positive bias towards adding colour to the diagrams being helpful in aiding the interpretation of constraint diagrams.

Participants developing constraints in the OCL notation provided an overall positive response, with a slight neutral to negative bias towards adding colour to the statements being helpful in aiding the interpretation of OCL.

17. How easy is it to sketch things out when playing with ideas for solving the operations?

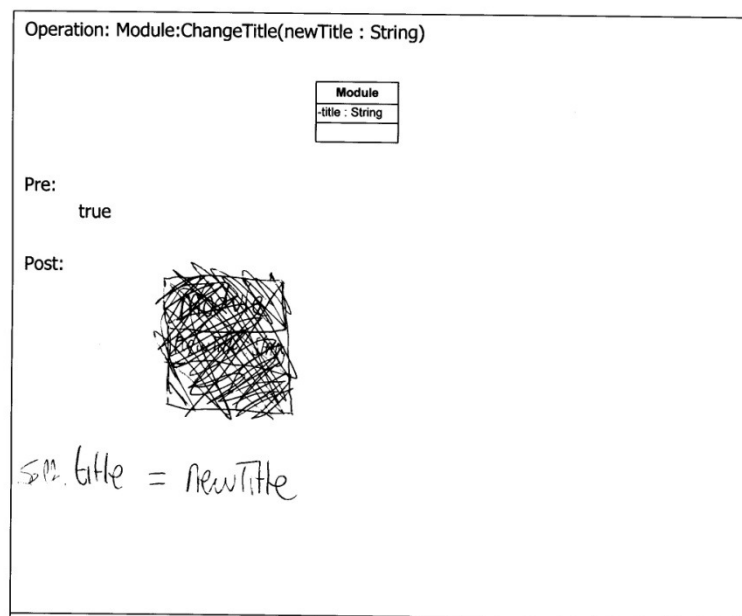
Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a positive bias towards finding it easy to sketch out ideas to help solve operations.

Participants developing constraints in the OCL notation provided a neutral to positive response towards finding it easy to sketch out ideas to help solve operations.

17.1. Are there features of the notation to help you do this?

Of all participants, only two responded positively to this question, both from the constraint diagram group of participants, and both failed to indicate what features of the notation would help when sketching out ideas.

However, it must be noted here that on several papers, participants completing an OCL statement have sketched out ideas using both constraint diagram notation and UML to give them a feel for the operation. See below for scans of these OCL statements with diagrams.



**Figure 38: OCL Statement with UML sketching**

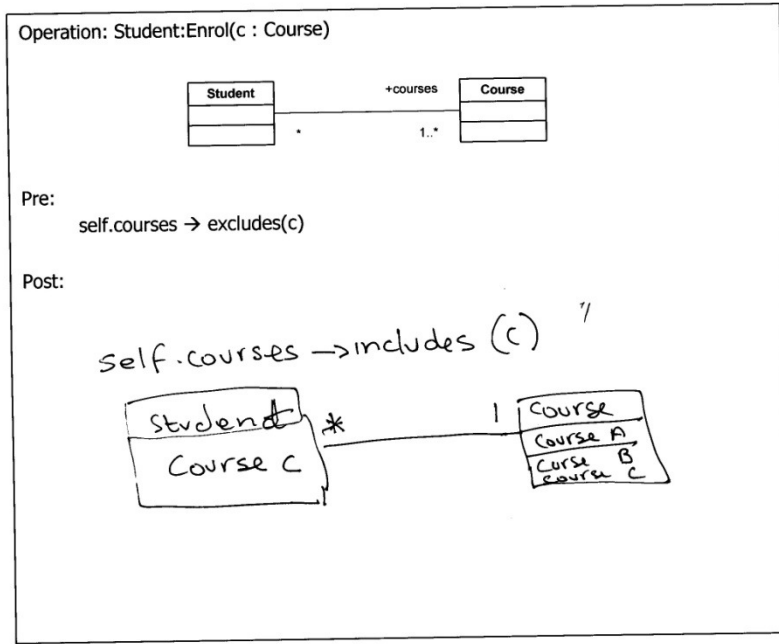


Figure 39: OCL Statement with UML sketching

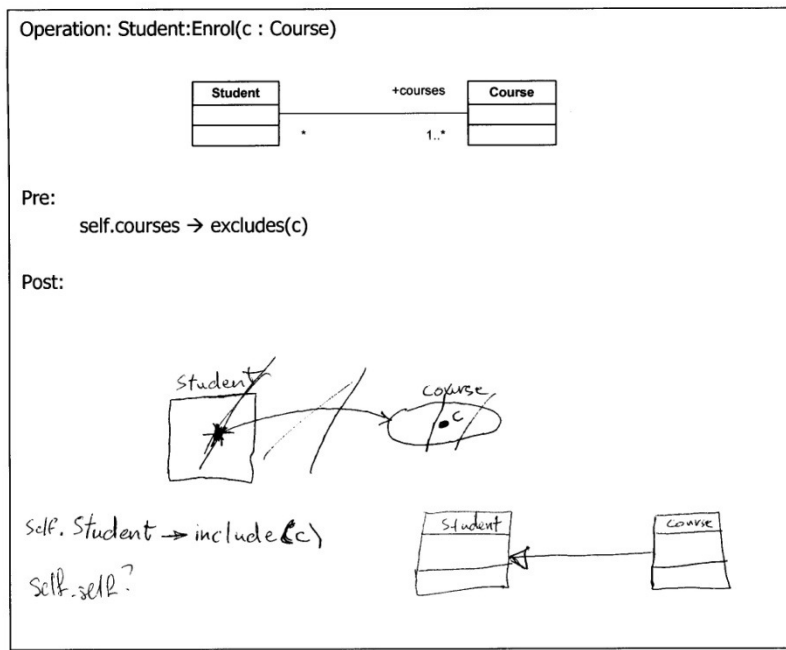
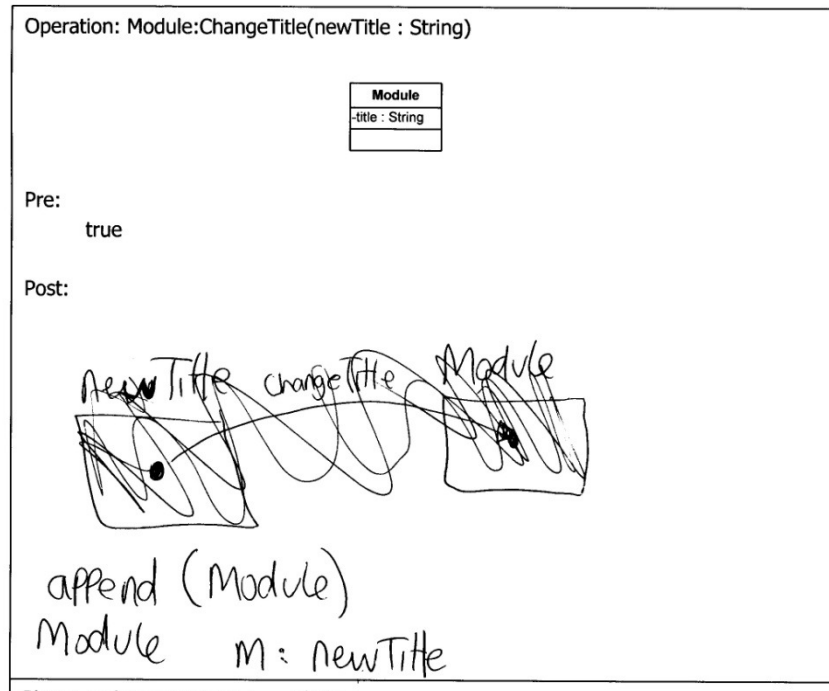


Figure 40: OCL Statement with constraint diagram sketching





**Figure 41: OCL Statement with constraint diagram sketching**

18. Do you think you are constrained in any way as to the order you chose for developing your operations?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with a positive bias towards not being constrained as to the order of developing the operations.

Participants developing constraints in the OCL notation provided an overall positive response, with a slight neutral to negative bias towards not being constrained as to the order of developing the operations.

19. Did you think that different parts of the notation were similar, i.e. meaning similar things?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with no particular bias towards indicating that different parts of the notation were either similar or dissimilar.

Participants developing constraints in the OCL notation showed no particular bias (positive, negative or neutral) towards indicating that different parts of the notation were either similar or dissimilar.

19.1. Did you think any similarities were clear?

All participants, regardless of the notation being examined, responded negatively to this question i.e. that those similarities were not clear. Indeed, in most cases, the overall response was that there were no similarities between different parts of the notation.

19.2. Were there places where some things ought to be similar but the notation makes them appear different?

All participants, regardless of the notation being examined, responded negatively to this question i.e. that there were no places where some things should be similar, but the notation made them appear different.

20. How easy do you think it would be to extend the syntax of the notation to incorporate other Object-Oriented or UML constructs?

Participants developing constraints in the constraint diagram notation provided an overall neutral response, with no particular bias towards indicating the difficulty of extending the syntax to incorporate other object-oriented constructs.

Participants developing constraints in the OCL notation provided an overall positive response, with regard to the ease of extending the syntax to incorporate other object-oriented constructs.

We can describe the profile graphically using a box plot type notation:

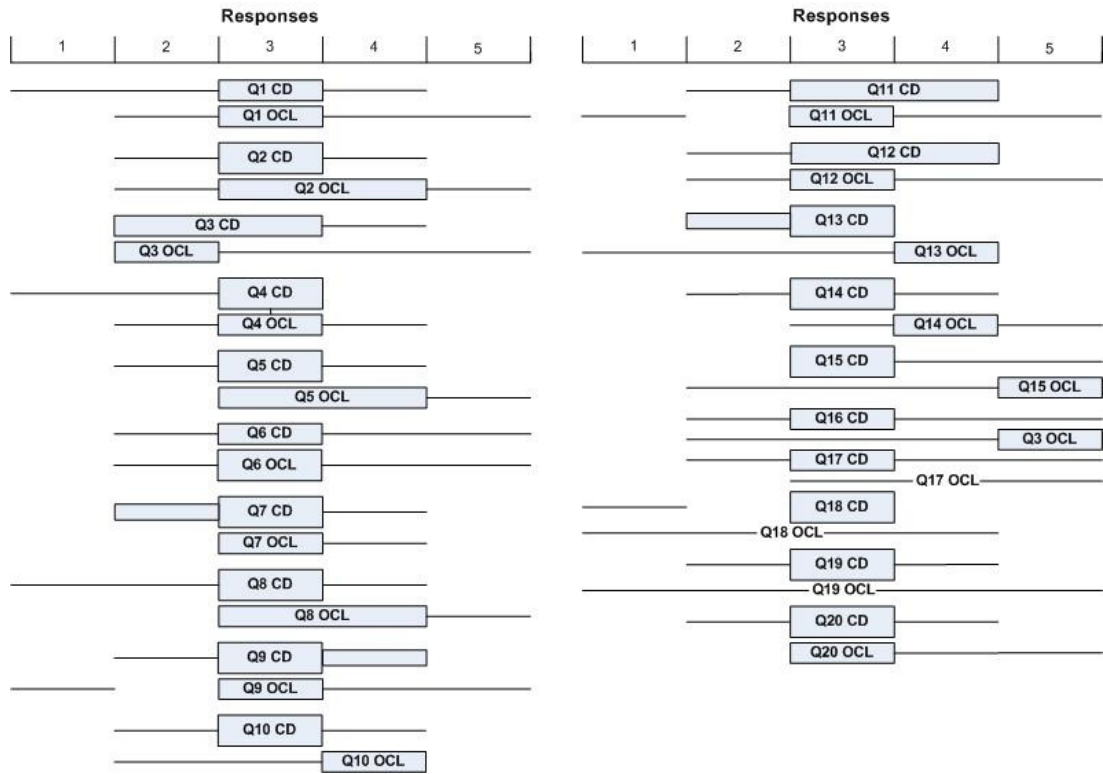


Figure 42: Graphical representation showing the profile of responses from the questionnaire

## 10 Conclusions and Further Work

### 10.1 Conclusions

Due to nature of the education and training of software engineers, where a whole range of mathematical and diagrammatic notations are introduced at a relatively early stage, we find that the correct interpretation of constraint diagrams becomes a readily attainable task. Regardless of whether a software engineer has prior knowledge of the constraint diagram notation, if they are given a model to develop an application from they should find little problem understanding the model and therefore developing said application.

This should be consistent regardless of the specialities of the software engineer. For example, a computer scientist specialising in embedded systems could have detailed knowledge of symbolic notations such as Petri Nets for concurrency and 'Z' for formal specification of programs. A business systems application developer could have knowledge of UML with OCL for developing user-centric applications as well as set theory and relational algebra for database development. There may also be skills overlap between the various fields.

The knowledge of these various notations, whether they are diagrammatic, symbolic or textual, aids in the understanding and interpretation of, in most cases, this new notation.

Developing constraint diagrams provides a more complex situation. With the study undertaken, we see that developing constraint diagrams appears to be more difficult than using OCL. There may be many factors that influence this, from individual preference to a notation for developing the constraints, to the sparseness of the syntax for constraint diagrams, as opposed to the richness of the syntax of OCL.

Software engineers are taught to develop applications in textual languages, such as Java, C++, ADA, SQL, etc. This predominance for textual languages may bias the individual to a preference to the textual modelling OCL. Perhaps if visual programming languages were taught as well, e.g. ProGraph, then a particular individual may not demonstrate this bias as readily.

Constraint Diagrams only have a small number of syntactic elements to model with. Each syntactic element can be used in a number of situations. Indeed, to get the richness of a language like OCL, we have to “reuse” each element in a specific context, each of which may be interpreted differently for separate models.

OCL, on the other hand, has a rich syntax with a large number of syntactic elements, each of which has a specific use; therefore by breaking down the model into its component parts, we find one specific OCL statement for each of these component parts. This may make the software model easier to develop.

One possible way to increase the efficiency of developing constraint diagram models would be to provide intelligent tool support. Currently, there are only a small handful of rudimentary tools, really specialist diagram editors, most of which are outdated. The notation has evolved since these tools were written. However, rather than have just diagram editors, perhaps a full CASE tool (or set of CASE tools), similar to Microsoft® Visio or IBM Rational Rose, would aid the overall modelling process with constraint diagrams.

To conclude, we find that constraint diagrams are usable by software engineers with varied backgrounds i.e. they are easily interpreted, although to develop constraint diagrams accurately requires rather more training or education than for OCL.

## 10.2 Further Work

This programme of research has investigated some usability aspects of the constraint diagram notation, and has made comparison with OCL, a well known and widely used modelling language that is part of the UML. While OCL must reflect a UML class diagram, constraint diagrams may reflect a UML class diagram (a top-down approach) or may not reflect UML at all i.e. it is possible to build models of systems using just constraint diagrams (a bottom-up approach). Using this programme of research as a basis, several areas of further research are evident.

It is possible to perform more of the same type of usability research, with larger groups of participants, or with the same size groups, but having more studies, to determine whether the groups used in this programme of research are really indicative of the software engineering population as a whole. It may also be possible

to perform larger studies using just the cognitive dimensions framework, also using new dimensions as they are introduced to the framework. Rather than using a “cut-down” version of the standard questionnaire for multiple choice responses, a full cognitive dimensions study would follow the standard questionnaire, while also adding additional questions for new dimensions, and give verbose responses as appropriate.

Further notations could be introduced as a comparison to constraint diagrams, either replacing or complementing the OCL operations. Examples of these notations include the relatively new notations of Visual OCL and JML (The Java Modelling Language), or the more widely known and used ‘Z’, Larch or First Order Predicate Logic (FOPL). It may even be prudent to test these new notations against themselves, without using constraint diagrams, to determine the efficacy of the studies this program of research has used. To further test the interpretation of constraint diagrams, we could increase the number of constraints used, and introduce other notations as a comparison. To further test the development of constraint diagrams we could do the same i.e. introduce additional constraints and ask for completion in other notations. The issue here, though, would be the prior knowledge of these notations for development purposes. The focus of the groups of participants may need to be moved from software engineers, perhaps to computer scientists who should have a more mathematical background as a lot of these notations have at least a basis in mathematical symbolic notations.

As constraint diagrams evolve, for example with the addition of reading trees [53, 54], or any other subsequent notation, similar studies could be undertaken to determine the effective usability of these new artefacts, and help determine if their use would be of benefit to a software engineer using the constraint diagram notation in the field.

A recent evolution of the constraint diagram is the Concept Diagram [55, 56], developed initially to examine ontologies. An ontology is an explicit specification of a conceptualization [57]. Ontologies are the structural frameworks for organizing information and are used as a form of knowledge representation about the world or some part of it. An ontology model comprises a set of statements (called axioms) that capture properties of individuals, concepts and roles [56]. Concept diagrams

were proposed in [58] for the purpose of Ontology Modelling. Concept diagrams use the majority of the ‘atomic’ syntax of the constraint diagram notation, but the manner in which the syntax is used is (sometimes subtly) different.

One major area of research that is lacking, though, is that of formulating a new framework or methodology to standardise the study of the usability of software engineering notations. Currently, for HCI, there are a number of complementary methods to provide an overall picture of the interface and how to better tailor it for the particular user base it is aimed at. However, for usability of notations there is only the cognitive dimensions framework. Additional frameworks and methods would add depth and clarity to the testing of notations to determine their suitability and usability for use in real-world software engineering projects.

As part of this new framework, metrics would be decided upon, not dissimilar to the ones used in study two of this programme of research. A more rounded list of metrics would need to be defined, perhaps encompassing the various types of notation i.e. diagrammatic, textual and mathematically symbolic. These metrics may relate to one or more of the types of notation, and may be used for either specific or general purposes.

It may be prudent to investigate using patterns as a usability tool e.g. having a pattern that is common in any notation would allow for clearer interpretation as well as faster and more accurate development, especially if the diagrams could be provided with a full description of the constraint to be modelled. Software engineers could be taught the meanings of these patterns as part of an undergraduate or postgraduate programme of study, or even as part of a professional training programme.

It would be interesting to investigate the phenomenon of drawing constraint diagrams to describe OCL statements. As part of the second study, participants were asked to write OCL statements. In some cases, as noted earlier in chapter 9, some participants drew constraint diagrams and other UML diagrams to aid in the interpretation and understanding of the operation prior to completing the OCL statement for that operation.

It must be noted that this phenomenon was not wide-spread, but has also been noted in other cases, for example in examination papers on modelling using both constraint diagrams and OCL. It may be that some software engineers have an affinity for modelling in diagrams, while others have an affinity for modelling in textual or symbolic notations. If we could understand this phenomenon more, perhaps we could provide tools for modelling that would allow the development of one preferred notation, but could translate to a number of other notations based on personal preference of both the software engineer performing the modelling exercise and the software engineer tasked to build the software artefact from the model.



## 11 Bibliography

1. Dubrovin, J., Jumbala — An Action Language For UML State Machines, Master's thesis, Department of Engineering Physics and Mathematics, 2006, Helsinki University of Technology, Helsinki.
2. The Object Management Group, Unified Modeling Language Specification 1.5, 2003, pp 1-12. Available from: <http://www.omg.org/spec/UML/1.5/>
3. Stevens, P. and Pooley, R., *Using UML: Software Engineering with Objects and Components*, Updated ed. 2000, Pearson Education Ltd.
4. Pender, T., *The UML Bible*. 2003, Wiley Publishing Inc.
5. The Object Management Group, Unified Modeling Language (OMG UML), Superstructure, V2.1.2. 2007. Available from: <http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF/>
6. Farhad, J. The UML Extension Mechanisms, 2002. Available from: <http://www.cs.ucl.ac.uk/staff/ucacwxe/lectures/3C05-02-03/aswe15-essay.pdf>.
7. The Object Management Group, The Object Constraint Language Specification 2.0, 2005. Available from: <http://www.omg.org/spec/OCL/2.0/PDF>
8. Warmer, J. and Kleppe, A., *The Object Constraint Language: Precise Modelling with UML*, 1st ed. 1999, Addison Wesley Longman Ltd.
9. The Object Management Group, Object Constraint Language Specification 2.0, 2005. Available from <http://www.omg.org/spec/OCL/2.0/PDF>
10. Object Constraint Language Specification 1.1, 1997, OMG document ad970808.
11. Graham, I., *Migrating to Object Technology*, 1995, Addison Wesley.
12. Mandrioli, D. and Meyer, B., eds. *Advances in Object-Oriented Software Engineering*, 1991, pp. 1-50, Prentice Hall.
13. Meyer, B., Applying "Design By Contract", *Computer (IEEE)*, 1992, pp. 40 - 51.
14. Vaziri, M. and Jackson, D., Some Shortcomings of OCL, the Object Constraint Language of UML, 1999. Available from: <http://www.omg.org/docs/ad/99-12-05.pdf>

15. Flower, J., Howse, J., Taylor, J. and Kent, S., A Visual Framework for Modelling with Heterogeneous Notations, IEEE Symposia on Human-Centric Computing Languages and Environments, 2002, pp. 71-73.
16. Gil, J., Howse, J., and Kent, S., Towards a Formalization of Constraint Diagrams, IEEE Symposia on Human-Centric Computing Languages and Environments, 2001, pp. 72-79, IEEE.
17. Lull, R., *Ars Magma*. 1517, Lyons.
18. Euler, L., *Lettres a Une Princesse d'Allemagne*, in Letters No. 102-108, 1761.
19. Venn, J., On the diagrammatic and mechanical representation of propositions and reasonings, 1880: Phil.Mag.
20. Peirce, C., *Collected Papers*, 1933, Harvard University Press.
21. Shin, S.-J., *The Logical Status of Diagrams*, 1994, Cambridge University Press.
22. Hammer, E., *Logic and Visual Information*, 1995, CSLI Publications.
23. Harel, D., *On visual Formalisms, in Diagrammatic Reasoning*, Glasgow, J., Narenarayanan, N. N., Chandrasekaran, B., Eds. 1998, pp. 235-272, MIT Press.
24. Gil, J., Howse, J., and Kent, S., Formalizing Spider Diagrams, IEEE Symposium on Visual Languages, 1999, pp.130-137, IEEE.
25. Redmond-Pyle, D. and Moore, A., *Graphical User Interface Design and Evaluation*, 1995, Prentice Hall.
26. Dumas, J.S. and Redish, J.C., *A Practical Guide to Usability Testing*, 1999, Intellect.
27. UsabilityNet International Standards. Available from:  
[http://www.usabilitynet.org/tools/r\\_international.htm](http://www.usabilitynet.org/tools/r_international.htm).
28. ISO 9241-11: Guidance on Usability. 1998. Available from:  
[http://www.usabilitynet.org/tools/r\\_international.htm](http://www.usabilitynet.org/tools/r_international.htm).
29. GOMS Summary. Available from:  
[http://ei.cs.vt.edu/~cs5724/g2/summary.html?&lang=en\\_us](http://ei.cs.vt.edu/~cs5724/g2/summary.html?&lang=en_us).
30. GOMS. Available from: <http://www.usabilityfirst.com/usability-methods/hci-design-approaches/>.
31. GOMS Model Work. Available from:  
<http://www.eecs.umich.edu/~kieras/goms.html>.

32. Usability First: Method: GOMS. Available from:  
<http://www.usabilityfirst.com/methods/goms.txt>.
33. Crystal, A. and Ellington, B., Task analysis and human-computer interaction: approaches, techniques, and levels of analysis. *Proceedings of the Tenth Americas Conference on Information Systems*, New York, 2004. Available from:  
<http://tc.eserver.org/33300.html>
34. The use of Task Analysis in HCI. 1998. Available from:  
<http://www.psy.gla.ac.uk/~steve/HCI/cscln/trail1/Lecture8.html>.
35. Gray, W.D. and Salzman, M.C., Repairing Damaged Merchandise: A rejoinder, *Human-Computer Interaction*. Vol. 13 no. 3, pp. 325-335.
36. Green, T. and Blackwell, A., Cognitive Dimensions of Notations. Available from:  
<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/>.
37. Green, T. and Blackwell, A., ed. Cognitive Dimensions of Notations. *People and Computers V*, ed. A. Sutcliffe and L. Macaulay. 1989, pp. 443-460, Cambridge University Press.
38. Green, T. and Blackwell, A., Cognitive Dimensions of Information Artefacts: a tutorial. 1998 [cited; Available from:  
<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDtutorial.pdf>.
39. Green, T. and Blackwell, A., Cognitive Dimensions of Information Artefacts: a tutorial. Available from:  
<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDtutorial.pdf>.
40. Triffitt, E. and Khazaei, B., A Study of Usability of Z formalism Based on Cognitive Dimensions, 14th Workshop of the Psychology of Programming Interest Group, 2002 pp.15-28, Brunel University.
41. Kutar, M., Britton, C., and Barker, T., A Comparison of Empirical Study and Cognitive Dimensions Analysis in the Evaluation of UML Diagrams, 14th Workshop of the Psychology of Programming Interest Group, 2002, pp. 1-14, Brunel University.
42. Faulkner, X., *Usability Engineering*, 2000, Macmillan Press Ltd.
43. Englefield, P. A pragmatic framework for selecting empirical or inspection methods to evaluate usability, 2005. Available from:  
[http://echo.iat.sfu.ca/library/englefield\\_00\\_inspection\\_methods\\_usability.pdf](http://echo.iat.sfu.ca/library/englefield_00_inspection_methods_usability.pdf).
44. Lazar, J., Feng J. H., and Hochheiser H., *Research Methods in Human Computer Interaction*, 1st ed., 2010, John Wiley & Sons Ltd.

45. Oehlert, G.W., *A first course in design and analysis of experiments*, 1st ed, 2000, W.H. Freeman (New York)
46. Haslam, S.A. and McGarty, C., *Research methods and statistics in psychology*, 2003, Sage Publications.
47. Trochim, W.M.K. *Research Methods Knowledge Base: Inferential Statistics*, 2006. Available from: <http://www.socialresearchmethods.net/kb/statinf.php>.
48. Marshall, G. *A Dictionary of Sociology*, 1998. Available from: <http://www.encyclopedia.com/doc/1O88-variationstatistical.html>.
49. Kaplan-Meier Survival Analysis. Available from: <http://www.statisticssolutions.com/methods-chapter/statistical-tests/kaplan-meier-survival-analysis/>.
50. Altman, D.G., Machin, D., Bryant, T.N. and Gardner, M.J., *Statistics with confidence*, 2nd ed., 2000, BMJ Books.
51. Peat, J. and Barton, B., *Medical Statistics: A guide to data analysis and critical appraisal*, 1st ed., 2005, Blackwell Publishing.
52. Green, T. and Blackwell, A., *A Cognitive Dimensions Questionnaire*, 2007. Available from: <http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDquestionnaire.pdf>.
53. Fish, A., Flower, J., and Howse, J., A reading algorithm for constraint diagrams, IEEE symposium on human centric computing languages and environments, 2003, pp. 161-168, Auckland, New Zealand.
54. Fish, A., Flower, J., and Howse, J., The semantics of augmented constraint diagrams, *Journal of Visual Languages & Computing*, 2005.
55. Howse, J., Stapleton, G., and Oliver, I., Visual Reasoning about Ontologies, in *International Semantic Web Conference, CEUR*, 658:5-8, 2010, Shanghai, China.
56. Chapman, P., Stapleton, G., Howse, J., and Oliver, I., Deriving Sound Inference Rules for Concept Diagrams, *VLHCC*, 2011, pp. 87-94.
57. Gruber, T.R., *A Translation Approach to Portable Ontology Specifications Knowledge Acquisition*, 1993, pp. 199-220.
58. Oliver, I., Howse, J., and Stapleton, G., Visualising and Specifying Ontologies using Diagrammatic Logics, *5th Australasian Ontologies Workshop, CRPIT*, pp. 37-47, 2009.

## **12 Appendix I**

Introductory material used in study one – slides.

The slides were presented as a Microsoft® PowerPoint presentation, to be made available while the participants took part in the study.

# **Constraint Diagrams**

## **An Introduction**

**By Neil Morgan**  
**n.a.morgan@brighton.ac.uk**

# Constraints

What are Constraints?

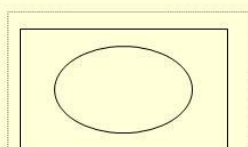
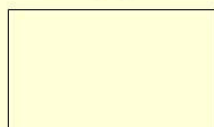
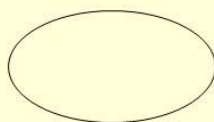
A constraint is a restriction on one or more values of (part of) an object-oriented (OO) model or system (Warmer & Kleppe, 1999).

There are two types of constraint in OO models: *Invariants* and *Operation Pre- and Post-conditions*. Invariants are rules that must apply to the model or system, regardless of the state it is in. These will be examined in detail within this first study.

Pre- and Post-conditions refer to a design principle, Design By Contract (for more information on Design By Contract, please refer to *The Object Constraint Language: Precise Modelling with UML*, Warmer & Kleppe, 1999.). These will be examined in later studies.

# Notation

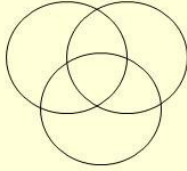
Constraint Diagrams are a diagrammatic notation for precisely modelling constraints on an OO model or system



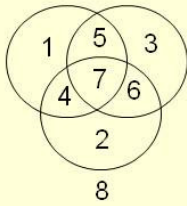
An example of how contours are used.

These structures are termed Contours. Contours are used in Constraint Diagram notation to denote sets. In our examples we will use rectangles to represent all objects of a class and ellipses to represent subsets of objects, either objects in different states or objects with different identity.

# Notation



A District is the set of points lying on or within a given contour. The example opposite has 3 contours (circles), and therefore 3 districts.



The Regions of a diagram are the non-empty sets which can be formed from its districts by means of set union, set intersection and set difference operations.

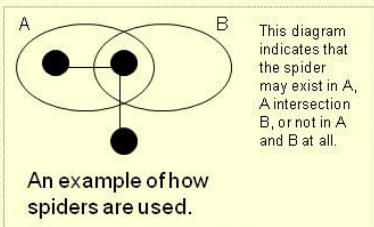
A Zone is a region that has no other region as a proper subset.

Therefore in our diagram opposite, each individually numbered area is a Zone, whereas areas 1, 4 & 7 would represent a Region, as would 4, 5, 6, 7 & 8.

# Notation

These structures are termed Spiders. Spiders denote elements of sets. Spiders are represented as a tree-like structure, with lines (called legs) connecting the shapes opposite (called feet).

- A square represents a Unitary Spider i.e. a named element e.g. in the set of Integers, element Zero would be represented by a square.
- A circle represents an Existential Spider i.e. at least one element e.g. in the set of Integers, there exists at least one positive Integer and would be represented by a circle.
- \* A star represents a Universal Spider i.e. all elements e.g. a star would represent all elements in the set of Integers.



# Notation

Distinct spiders represent distinct elements unless they are connected by a *strand* or a *tie*.



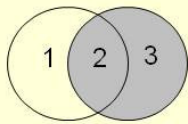
A strand (the wavy line) denotes that spiders may be the same if they occur within the same zone.

A tie (the double straight line) denotes that spiders must be the same if they occur within the same zone.

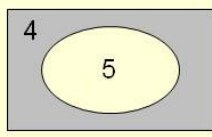
An arrow is used to connect one spider to another spider or zone.

<p><b>A</b></p> <p>An example of how strands are used.</p>	<p><b>B</b> This diagram indicates that if the spider is not in A and B at all, it <b>MAY</b> be the same element, but is not the same if it is within A, or A intersection B.</p>	<p><b>A</b></p> <p>An example of how ties are used.</p>	<p><b>B</b> This diagram indicates that if the spider is not in A and B at all, it <b>MUST</b> be the same element, but is not the same if it is within A, or A intersection B.</p> <p>An example of how connectors are used.</p>
--	--	---	---

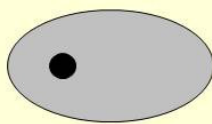
# Notation



Shading within a zone, that contains no spiders, indicates that zone is empty i.e. has precisely zero elements (the empty set). A zone that is unshaded has zero or more elements.



In the first example zones 2 & 3 are empty while zone 1 may contain zero or more elements. In the second example zone 4 is empty while zone 5 contains zero or more elements.

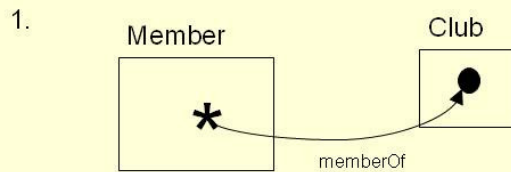


It is also possible to quantify exactly the number of elements in a set, by adding spiders to an empty set. This example indicates the set has exactly one element. Additional spiders would increase the number of set members.



## Examples

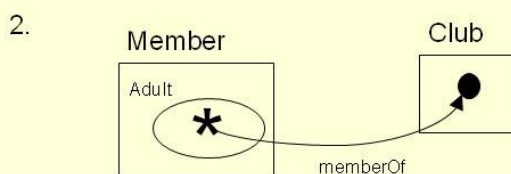
We introduce the notation with a few simple examples.



Here, all elements within the set "Member" (using universal spider '\*') are related to one element in the set "Club" (using existential spider '●') by the relation "memberOf". Our constraint would be "All Members are members of a Club"

## Examples

We introduce the notation with a few simple examples.



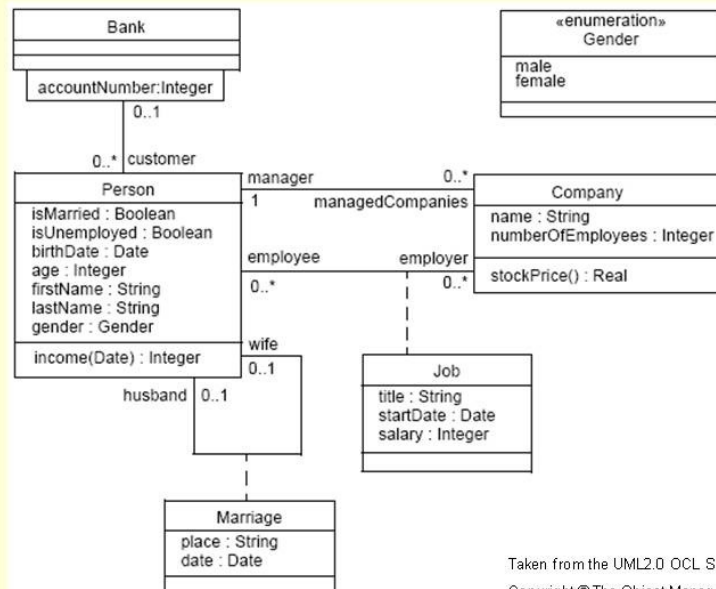
Here, all elements within the subset "Adult" (using universal spider '\*') are related to one element in the set "Club" (using existential spider '●') by the relation "memberOf". Our constraint would be "All Members who are Adults are members of a Club"

# Case Study

To introduce the use of the Constraint Diagram Notation in a Software Engineering context, we will take an example class diagram and model some constraints. These constraints will be arbitrarily chosen for their suitability of demonstrating the notation at work. Because of this, their relevance to a “Real System” may well be questionable.

The Class Diagram is taken from the Object Management Group's specification of OCL 2.0 (available from the OMG website).

# Class Diagram



Taken from the UML2.0 OCL Specification  
Copyright © The Object Management Group, 1997-2003  
Reproduced with permission under the terms of the licence

## Constraints

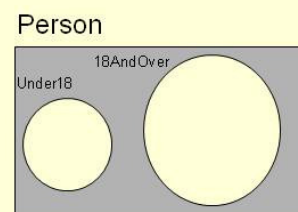
To demonstrate Constraint Diagrams, we will model the following invariant constraints.

- A person under the age of 18 cannot be an employee of a company
- An unemployed person has an income of under £100
- A manager of a company is not unemployed
- The gender of a wife must be female
- The gender of husband must be male

## Modelling a Constraint

### A person under the age of 18 cannot be an employee of a company

Our Class Person (see class diagram previously) has a property Age. With this we can identify members of Person with an age of under 18 and an age of 18 and over.

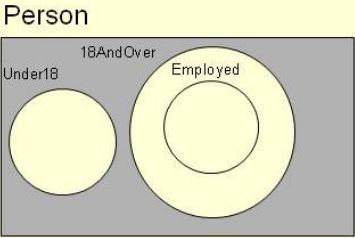


We can model this as depicted right. The set Person is shaded representing an empty set, i.e. a set with no elements. There are, however two subsets, 'Under18' and '18AndOver'. There is no shading so we can say both these subsets have Zero or more elements. We can also state that a Person falls into one of two subsets 'Under18' or '18AndOver'.

# Modelling a Constraint

**A person under the age of 18 cannot be an employee of a company**

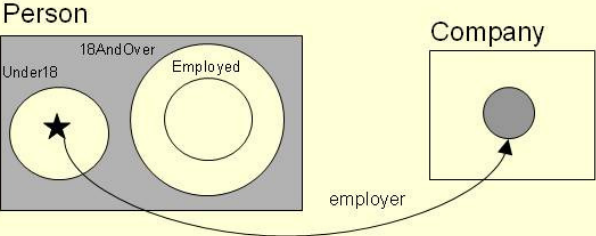
Our invariant states that a person under the age of 18 cannot be employed by a company. We can then identify a subset of **Person**, **18AndOver**, Employed. There is no subset of Employed for the subset Under18, so we begin to enforce our invariant.



# Modelling a Constraint

**A person under the age of 18 cannot be an employee of a company**

We can then identify the relationship between all employed persons and a company as seen here. The asterisk represents 'all members of the set or subset', in this case all persons who under 18. The arrow indicates the relationship: i.e. all under 18s have no employer.



# Modelling a Constraint

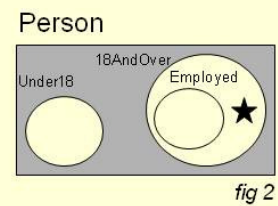
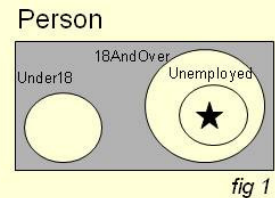
## An unemployed person has an income of under £100

Using our previously defined class we can also model the income of an unemployed Person. There are two ways of achieving this goal.

We can either explicitly define a set of Unemployed persons in the age group 18AndOver and quantify the members (see *fig. 1*).

Or we can continue as in the previous diagram by defining the set of Employed persons and quantify the members outside this set as being Unemployed (see *fig 2*).

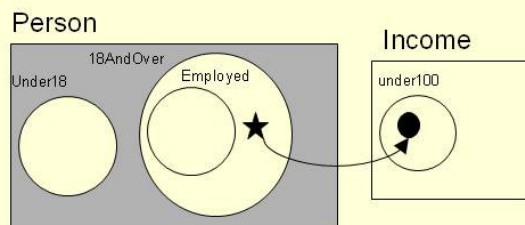
Either would work as the property isUnemployed is a boolean value, i.e. true or false, indicating one of two states.



# Modelling a Constraint

## An unemployed person has an income of under £100

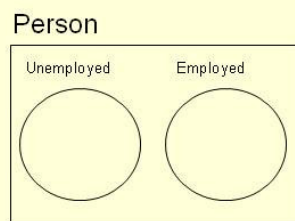
We can then identify the relationship between all unemployed persons and an income of under £100. There is no defined salary for unemployed persons, they may earn any amount that is under £100. Therefore we use an existential spider to denote one value within the set, rather than connecting to the contour which indicates all values within the set.



## Modelling a Constraint

### A manager of a company is not unemployed

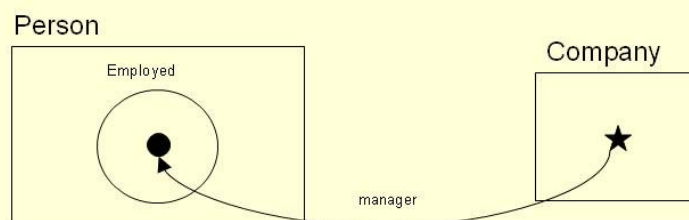
Using our boolean property **Person.isUnemployed**, we can identify two possible states of employment (Employed and Unemployed). We can model this as the diagram below. Person is not shaded as we take into account the possibility of persons in the set Under18, although adding the extra contours would unnecessarily complicate the diagram, while providing no additional information pertinent to the invariant, so we can omit them.



## Modelling a Constraint

### A manager of a company is not unemployed

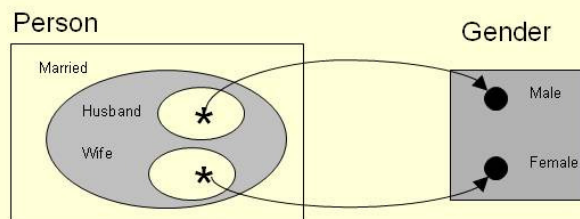
Our invariant states that a manager is not unemployed. Each company has one manager and that person is employed.



## Modelling a Constraint

**The gender of a wife must be female**  
**The gender of a husband must be male**

We can model these two invariants within the same diagram. We know from the class diagram that a person may be either Married or Unmarried. We model the enumeration class Gender as being a set with exactly two elements. Hence we can model the invariant that all husbands are male and all wives are female.



## Thank You

**If you require a break, please take one.**

**Refreshments are available at the Megabytes Café on the first floor of  
Watts building.**

**When you are ready, please continue to the questionnaire.**

## 13 Appendix II

Study material used in study one – paper based questionnaire.

The paper-based questionnaire is based loosely on a subsection of the Cognitive Dimensions framework.

### The Usability of Diagrammatic Notations: Interpreting Constraint Diagrams

This questionnaire collects your views as to how easy it is to interpret Constraint Diagrams. The questions encourage you to think about the ways that you see Constraint Diagrams, and how they would benefit you when modelling Object-Oriented systems.

Question 1:

How closely related is the notation to the constraints you are describing?

Not Close		(please tick one box)		Very Close
1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>

Question 2:

How well does the notation depict the constraints you have chosen?

Not Well		(please tick one box)		Very Well
1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>

Question 3:

How closely does the concept of a class in this notation map to your expectations of a class represented in other Object-Oriented notations?

Not Close		(please tick one box)		Very Close
1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>

Question 4:

How closely does the concept of an object in this notation map to your expectations of an object represented in other Object-Oriented notations?

Not Close		(please tick one box)		Very Close
1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>

Question 5:

How closely does the concept of a set of objects in this notation map to your expectations of a set of objects represented in other Object-Oriented notations?

Not Close		(please tick one box)		Very Close
1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>



## The Usability of Diagrammatic Notations: Interpreting Constraint Diagrams

Question 6:

How closely does the concept of a relation in this notation map to your expectations of a relation represented in other Object-Oriented notations?

Not Close (please tick one box) Very Close

1  2  3  4  5

Question 7:

When reading the notation how easy is it to tell what each part is for?

Not Easy (please tick one box) Very Easy

1  2  3  4  5

Question 8:

How difficult did you find the notation overall to interpret?

Very Difficult (please tick one box) Not Difficult

1  2  3  4  5

Question 9:

How helpful would notes made in natural language or another notation be in aiding your interpretation of this notation?

Not Helpful (please tick one box) Very Helpful

1  2  3  4  5

Question 10:

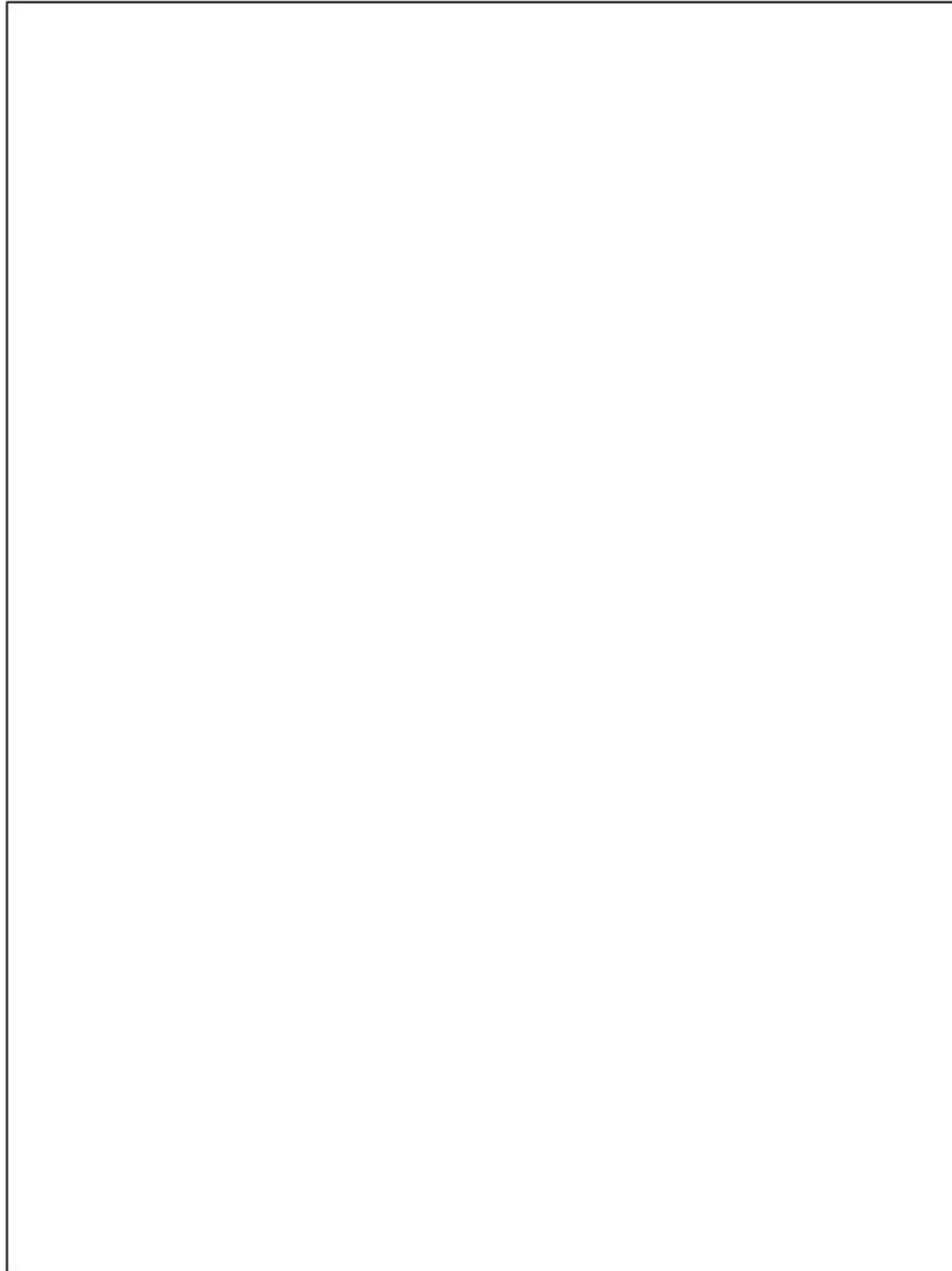
How helpful would adding colour to the notation help your interpretation?

Not Helpful (please tick one box) Very Helpful

1  2  3  4  5

## **The Usability of Diagrammatic Notations: Interpreting Constraint Diagrams**

Thank you for your participation. We would appreciate any comments or feedback you may have about the experiment, the materials used (PowerPoint slides, web pages, questionnaire, etc) or the venue.



## **14 Appendix III**

Introductory material used in study two – introductory material.

The introductory material is paper-based, and is adapted from the slides in study one. It begins by asking some demographic questions, e.g. are you a Professional, Academic or Student Software Engineer. We then expand on the previous study's slides and provide a tutorial on both Constraint Diagrams and OCL.

# The Usability of Constraint Diagrams in a Software Engineering Context

Please return this questionnaire to:

Neil Morgan  
Admin Computing  
University of Brighton  
Watts 137  
Lewes Road  
Brighton  
East Sussex  
United Kingdom  
BN2 4GJ

Tel: +44 (0) 1273 643930  
Fax: +44 (0) 1273 642666

Email: [n.a.morgan@brighton.ac.uk](mailto:n.a.morgan@brighton.ac.uk)

This questionnaire will remain confidential.  
However, If you would like to take part in  
further studies, please provide your name  
and email address.

Name:

Email:

Please tell us a little about yourself.

Are you a: (please tick)

Professional	<input type="checkbox"/>
Academic	<input type="checkbox"/>
Researcher	<input type="checkbox"/>
Student	<input type="checkbox"/>

How did you gain your experience of Object-  
Oriented technologies?

From University study	<input type="checkbox"/>
Self-taught or Independent training	<input type="checkbox"/>
I have no experience of Object- Oriented technologies	<input type="checkbox"/>

(This could be experience of design using e.g.  
UML etc, or programming using e.g. Java, C++,  
SmallTalk, Python, ADA, etc)

Did you complete this questionnaire yourself or  
in collaboration with others?

Myself

Collaboration

## An Introduction to the Constraint Diagram Notation

### What are Constraints?

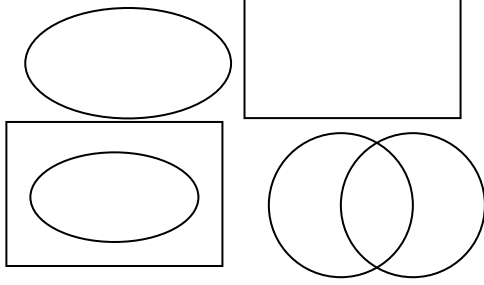
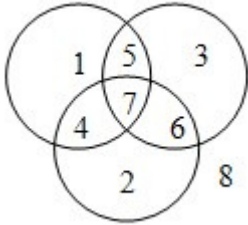
A constraint is a restriction on one or more values of (part of) an object-oriented (OO) model or system (Warmer & Kleppe, 1999). There are two types of constraint in OO models: *Invariants* and *Operations*. Invariants are rules that must apply to the model or system, regardless of the state it is in. These will be examined in detail within this first study. Operations are further categorised as being Events or Queries. Events are changes to an object or set of objects and are modelled using a Design-by-Contract approach, specifying pre- and post-conditions of the event. Queries do not affect the state of an object or set of objects.

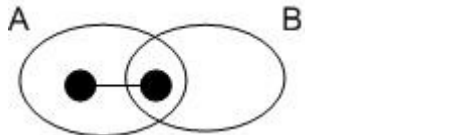
For more information on Design-by-Contract, please refer to *The Object Constraint Language: Precise Modelling with UML*, Warmer & Kleppe, 1999.

### What are Constraint Diagrams?

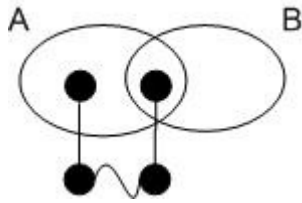
Constraint Diagrams are a diagrammatic notation for precisely modelling constraints on an OO model or system. It takes its notation from that of Set Theory, combining Venn and Euler diagrams effectively with spider diagrams (a more recent notation), extends it, and provides a rich notation for modelling real world problems.

### Elements of the Constraint Diagram Notation

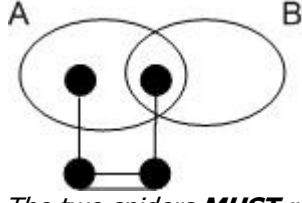
	<p>Contours.</p> <p>These denote sets of objects. A diagram may have any number of contours, and contours may be nested.</p> <p>Disjointedness, enclosure and intersection of contours represent disjoint sets, subset, and set intersection respectively.</p>
	<p>Regions and Zones.</p> <p>The Regions of a diagram are the areas that can be formed from its contours.</p> <p>A Zone is a region that has no other region as a proper subset.</p> <p>Therefore in our diagram opposite, each individually numbered area is a Zone, and any non-empty combination of zones is a region.</p>



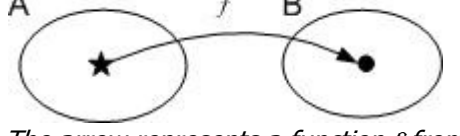
The spider represents an element in A (it is in A but not in B or in the intersection of A and B)



The two spiders **may** represent the same element but only if they both represent elements outside A.



The two spiders **MUST** represent the same element but only if they both represent elements outside A.



The arrow represents a function  $f$  from A to B

**Spiders.**



These structures are termed Spiders. Spiders denote elements of sets. Spiders are represented as a tree-like structure, with lines (called legs) connecting the shapes above (called feet). The first example diagram represents one spider, while examples two and three represent two spiders joined by strands or ties (see below).

A square represents a Constant Spider i.e. a named element e.g. in the set of Integers, element Zero would be represented by a square.

A circle represents an Existential Spider i.e. at least one element e.g. in the set of Integers, there exists at least one positive Integer and would be represented by a circle.

A star represents a Universal Spider (forall / foreach) i.e. all elements e.g. a star would represent all elements in the set of Integers.

**Connectors.**



Distinct spiders represent distinct elements unless they are connected by a *strand* or a *tie*.

A strand (the wavy line) denotes that the elements represented by spiders may be the same if they occur within the same zone.

A tie (the double straight line) denotes that the elements represented by spiders must be the same if they occur within the same zone.

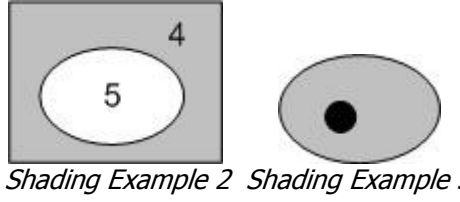
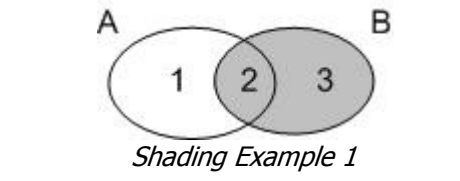
An arrow, representing a relation or function, is used to connect one spider to another spider or zone. Arrows can be sourced or targeted on contours or spiders.

**Shading.**

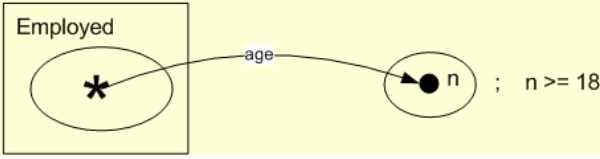
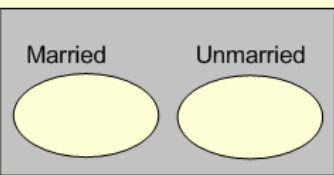

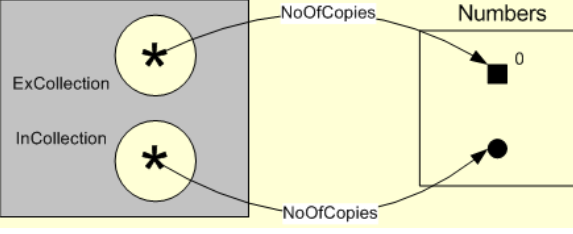
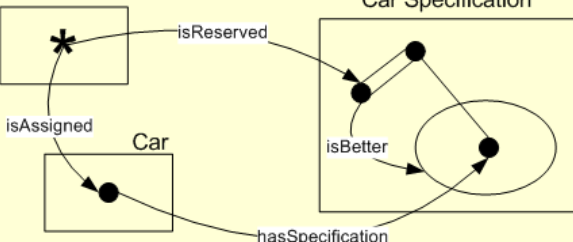
Shading within a zone that contains no spiders indicates that zone is empty i.e. has precisely zero elements (the empty set). A zone that is un-shaded has zero or more elements.

In the first example zones 2 & 3 are empty while zone 1 may contain zero or more elements. In the second example zone 4 is empty while zone 5 contains zero or more elements.

Shading indicates that no elements other than those represented by spiders exist in a shaded region. Example 3 indicates the set has exactly one element.



## Examples of Constraint Diagrams

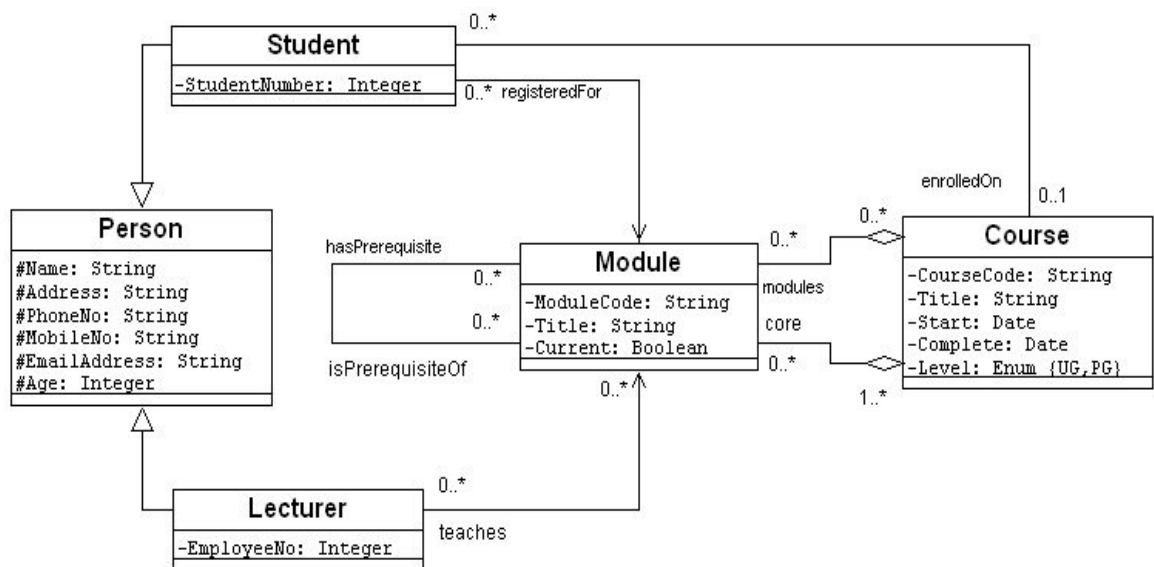
<p>Person</p> 	<p><b>All employed persons must be aged 18 or over.</b></p> <p>This diagram demonstrates the use of universal spider (all employed), existential spider (<math>n</math>, quantified by <math>n \geq 18</math>) and the arrow labelled age which is a function.</p>
<p>Person</p> 	<p><b>A person must be either married or unmarried but not both.</b></p> <p>This diagram demonstrates the use of shading. It denotes that an object of the class person must be in either state married or state unmarried (and not both).</p>
<p>Book</p> 	<p><b>The total number of books must equal the number of copies available plus the number of copies borrowed.</b></p> <p>This diagram additionally demonstrates the use of strands. This would represent the fact that <math>a = b = a + b</math> may hold (true if both <math>a</math> and <math>b</math> have a value of zero).</p>
<p>Book</p> 	<p><b>All books with a number of copies available are InCollection while all books with no copies available are ExCollection.</b></p> <p>This diagram demonstrates the use of constant spider. A book in state ExCollection must have zero, and only zero, copies. A book with any other number of copies must be in state InCollection.</p>
<p>Reservation</p> 	<p><b>The specification of a car assigned to a reservation must be the same or better than the specification reserved.</b></p> <p>The diagram shows for each reservation, the specification of the reserved car and the set of better specifications than this (following the arrows isReserved and isBetter). The specification of the assigned car (following the arrows isAssigned and hasSpecification) is represented by a two-footed spider. The foot in the elliptical contour represents a better specification than the one reserved; the foot outside the contour is connected by a tie to the spider representing the specification of the reserved car, indicating that these specifications must be the same.</p>

## An Introduction to the OCL Notation

### What is OCL?

OCL is a textual notation for precisely modelling constraints on a UML model. While most formal languages (e.g. Z, Larch, etc) are designed for people with a strong mathematical background, OCL is designed for software engineers using object oriented methods. OCL is designed for usability, although it is underpinned by mathematical set theory and logic (Warmer & Kleppe, 1999). OCL is a declarative language where constraints have no side effects i.e. the state of the system does not change due to the evaluation of an OCL expression.

In OCL, unlike Constraint Diagrams, all constraints are written in the context of a UML model, and never stand alone. To help us discuss OCL we will provide an example class diagram.



### Invariants

An invariant is a constraint that states a condition that must always be met by all instances of the class, type or interface. An invariant is described using an expression that evaluates to true if the invariant is met (Warmer & Kleppe, 1999). Invariants are always written in the form:

Context *<classname>* inv:  
    *<Boolean expression>*  
    [ *<Boolean expression>* ... ]

Invariants can be applied to attribute(s) of a class. You must provide a class name (the context of the invariant). All attributes of the context class may be used in the invariant. In an OCL expression the reserved word *self* may be used to refer to the contextual instance of the class, or you may specify a concrete instance e.g.



Context Student inv:                    example using self  
    Self.age >= 18

Context s: Student inv:                example using concrete instance  
    s.age >= 18

You can also apply invariants to attributes of instances on one class or on attributes of associated classes. Often it is necessary to associate an object with more than one object from an associated class. OCL has a number of collection operations to deal with such. These are identified with an arrow “->”. Collections of objects can be identified as Sets (an element appears only once), Bags (elements may appear more than once) and Sequences (a bag in which the elements are ordered) e.g.

Context Course inv:  
    Self.core->forAll( current = true )

Sometimes a UML class diagram defines an enumeration type as an attribute type. To identify the values of the enumeration type in OCL we prefix the value with a '#' symbol.

## Operations

An operation is specified using pre- and post-conditions that state the effect of the operation without stating the algorithm or implementation. The context of pre- and post-conditions is always an operation or a method. The parameters of the contextual operation are accessible in the OCL expression constituting the pre- and post-conditions.

Examples of OCL operations are:

Context Module::NewPrerequisite(m : Module) : Boolean  
Pre: self.hasPrerequisite = false  
Post: self.hasPrerequisite = m.ModuleCode  
    Self.Current = true

Context Lecturer::SetTaughtModule(m : Module) : Boolean  
Pre: not ( self.teaches->includes(m) )  
Post: self.teaches = self.teaches@pre->including(m) and  
    self.teaches.ModuleCode = m.ModuleCode

# Additional OCL Information

Slides courtesy of Dr Ali Hamie, University of Brighton, © 2003

### OCL basic and model types

- Predefined primitive types
  - Boolean: true, false
  - Integer: -3, 0, 6, 18
  - Real: -1.2, 0.0, 12.87
  - String: 'hello world'
- Types from UML model
  - Classifier types, eg.
    - Member
  - Enumeration types, eg.
    - Status: status::banned, #banned
- Other types
  - OclAny: super type of primitives and classifiers
  - OclVoid: null
  - OclInvalid: invalid

### OCL structured types

- Collection types
  - Set(T): sets where elements are unique and not ordered
  - OrderedSet(T): sets where elements are ordered
  - Bag(T): like sets but an element may occur more than once
  - Sequence(T): sequences where elements are ordered (lists)
  - Collection(T): abstract super type of other collection types
- Tuple types (records)
  - Tuple(a<sub>1</sub> : T<sub>1</sub>, ..., a<sub>n</sub> : T<sub>n</sub>)
- Examples
  - Set{1, 2} : Set(Integer)
  - Set{Set{1}, Set{3, 5}} : Set(Set(Integer))
  - Bag{-1.0, 6.3, 5} : Bag(Real)
  - Sequence{1..10} : Sequence(Integer)
  - Sequence{1.2, 5, 6.8} : Sequence(Real)
  - Tuple(x = 2, y = true) : Tuple(x : Integer, y : Boolean)

### OCL type hierarchy

- Type conformance relation  $\leq$ 
  - OclVoid, OclInvalid  $\leq T$  for all types T
  - Integer  $\leq$  Real
  - $T \leq T' \Rightarrow C(T) \leq C(T')$  for collection type C
  - $C(T) \leq \text{Collection}(T)$  for collection type C
  - $B \leq \text{OclAny}$  for all primitives and classifiers B
  - Generalization hierarchy from UML model
- Examples
  - String  $\leq$  OclAny
  - Set(Integer)  $\leq$  Set(Real)

### Operations on Boolean type

Operation	Notation	Result type
and	a and b	Boolean
or	a or b	Boolean
exclusive or	a xor b	Boolean
negation	not a	Boolean
equals	a = b	Boolean
not equals	a <> b	Boolean
implication	a implies b	Boolean
conditional	if a then b1 else b2 endif	type of b1

Example:  
true and false = false  
true or false = true

### Operations on Integer and Real types

Operation	Notation	Result type
equals	a = b	Boolean
not equals	a <> b	Boolean
less	a < b	Boolean
greater	a > b	Boolean
less or equal	a <= b	Boolean
greater or equal	a >= b	Boolean
plus	a + b	Integer/Real
minus	a - b	Integer/Real
multiply	a * b	Integer/Real
divide	a / b	Real
integer division	a.div(b)	Integer
absolute value	a.abs()	Integer/Real
maximum	a.max(b)	Integer/Real
minimum	a.min(b)	Integer/Real
round	a.round()	Integer
floor	a.floor()	Integer
modulus	a.mod(b)	Integer

Example: 8.7.floor() = 8

### Operations on String type

Operation	Notation	Result type
concatenation	s.concat(s1)	String
size	s.size()	Integer
to upper case	s.toUpper()	String
to lower case	s.toLower()	String
equals	s1 = s2	Boolean
not equals	s1 <> s2	Boolean
substring	s.substring(n,m)	String

Example:  
'John'.concat(' Smith') = 'John Smith'  
'John'.substring(1,2) = 'Jo'

### Operations on OclAny type

Operation	Notation	Result type
equals	a = b	Boolean
not equals	a <> b	Boolean
type of	a.oclIsTypeOf(T)	Boolean
Kind of	a.oclIsKindOf(T)	Boolean
in state	a.oclInState(state)	Boolean
is new	a.oclIsNew()	Boolean

30

### Operations on all collections (Collection)

Operation	Description
size()	the number of elements in the collection
count(o)	the number of occurrences of object o in the collection
includes(o)	true if object o is an element of the collection
includesAll(c)	true if all the element in collection c are present in the current collection
excludes(o)	true if object o is not an element of the collection
excludesAll(c)	true if all the element in collection c are not present in the current collection
isEmpty()	true if the collection contains no elements
notEmpty()	true if the collection contains one or more elements
sum()	the addition of all the elements in the collection

Operations on collections are denoted using the arrow syntax as `c->op(...)`, where c is a collection and op is a collection operation

31

### Specialised operations on collection types

Set	OrderedSet	Bag	Sequence
intersection(s)	first()	union(b)	first()
Intersection(b)	last()	union(s)	last()
union(s)	at(i)	intersection(s)	at(i)
union(b)	append(e)	intersection(b)	append(e)
including(e)	including(e)	including(e)	including(e)
excluding(e)	excluding(e)	excluding(e)	excluding(e)
asBag()	asBag()	asSet()	asBag()
asSequence()	asOrderedSet()	asSequence()	asOrderedSet()
flatten()	...	...	...
...			

b is a bag, s is a set and e is an element

32

### Iterator expressions

- Quantification
  - `c->forall( e : T | exp )` -- true if all elements of c satisfy expression exp
  - `c->exists( e : T | exp )` -- true if at least one element in c satisfy exp
- Example
  - `Set{1, 2, 3} ->forall( n : Integer | n > 3 )` evaluates to false
  - `Set{-1, 5, 8} ->exists( n : Integer | n + 1 > 10 )` evaluates to false
- Values selection
  - `c->select( e : T | exp )` -- all elements of c satisfying exp
  - `c->any( e : T | exp )` -- some element of c satisfying exp
- Example
  - `Set{1, 2, 3} ->select( n : Integer | n >= 2 )` evaluates to `Set{2, 3}`
- Collecting values
  - `c->collect( e : T | exp )` -- collection of elements with exp applied to each element of c
  - `c.p` -- collection of elements e.p for each e in c (short-hand notation for collect)

33

### Iterator expressions

- Uniqueness
  - `c->isUnique( e : T | exp )` -- true if exp has a unique value for all elements in the collection
- Sorting
  - `c->sortedBy( e : T | exp )` -- Sequence of all the elements in the collection in the order specified (< must be defined on exp)
- Iteration
  - `c->iterate( e : T1; result : T2 = exp1 | exp2 )`
    - Iterates over all the elements of the collection and accumulates the result in the variable result.
  - Example:
    - `Set{1, 2, 3} ->iterate( i : Integer; sum : Integer = 0 | sum + i )` evaluates to 6.
  - All other collection operations can be defined in terms of iterate

34

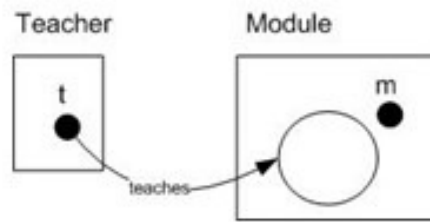
## **15 Appendix IV**

Study material used in study two – paper based modelling exercise.

A set of diagrams representing six constraints were developed; each constraint being represented in Constraint Diagram notation and in OCL notation. These diagrams are presented here.

Operation: Teacher:AddModule(m)

Pre:

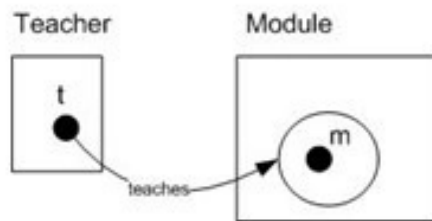


Post:

Please make any notes you wish here

Operation: Teacher:RemoveModule(m)

Pre:



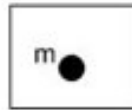
Post:

Please make any notes you wish here

Operation: Module:SetPrerequisite(s : Set(Module))

Pre:

Module

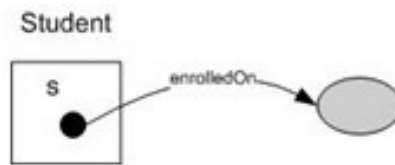


Post:

Please make any notes you wish here

Operation: Student:Enrol(c : Course)

Pre:



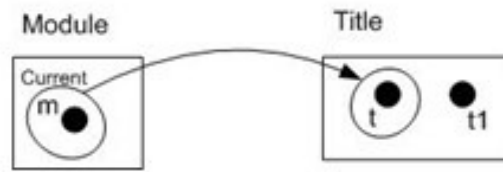
Post:

Please make any notes you wish here



Operation: Module:ChangeTitle(t1 : String)

Pre:



Post:

Please make any notes you wish here

Operation: Teacher:AddModule(m)



Pre:

self.modules → excludes(m)

Post:

Please make any notes you wish here

Operation: Teacher:RemoveModule(m)



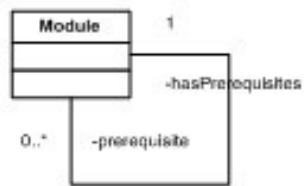
Pre:

self.modules → includes(m)

Post:

Please make any notes you wish here

Operation: Module:SetPrerequisite( $m \rightarrow p$ )



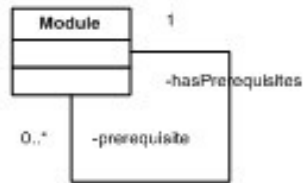
Pre:

$\text{self.prerequisite} \rightarrow \text{excludes}(p)$

Post:

Please make any notes you wish here

Operation: Module:SetPrerequisite(s : Set(Module))



Pre:

$\text{self.prerequisite} \rightarrow \text{excludesAll}(s)$  and  $\text{not}(s \rightarrow \text{isEmpty}())$

Post:

Please make any notes you wish here

Operation: Student:Enrol(c : Course)



Pre:

self.courses → excludes(c)

Post:

Please make any notes you wish here

Operation: Module:ChangeTitle(newTitle : String)

Module
-title : String

Pre:  
    true

Post:

Please make any notes you wish here

## **16 Appendix V**

Study material used in study two – paper based questionnaire.

The paper-based questionnaire is based loosely on the Cognitive Dimensions framework provided by [52].



## The Usability of Diagrammatic Notations: Developing Constraint Diagrams and OCL

This questionnaire collects your views as to how you developed the Constraint Diagrams and OCL statements. The questions encourage you to think about the ways that you see Constraint Diagrams and OCL, and how they would benefit you when modelling Object-Oriented systems.

When developing the operations, what proportion of your time (as a rough percentage) did you spend:

- Searching for information in the accompanying document \_\_\_\_\_ %
- Searching for information within the notation itself \_\_\_\_\_ %
- Translating substantial amounts of information from one notation to the other \_\_\_\_\_ %
- Making small amendments to one notation based on your work on the other \_\_\_\_\_ %

When completing the questions below, please tick or cross one of the numbered boxes for each question. Where appropriate, also tick or cross the Yes/No response and provide a short description as to why you have chosen this. If necessary, please write this on a separate sheet of paper (provided).

1	How easy is it to see or find the various parts of the notation while you were developing the operations?	Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Easy	
2	When you need to change your operation, how easy is it to make that change?	Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Easy	
3	Is the notation clear and concise?	Not Clear 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Clear	
4	How much mental effort did you need to develop the operations?  Did this task come easy to you?	A Lot 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>  <input type="checkbox"/> Yes <input type="checkbox"/> No If not why _____	A Little	
5	Was it complex or difficult to develop these operations in your head?  Would you have preferred extra space to develop these operations on paper?	Very Difficult 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>  <input type="checkbox"/> Yes if so why _____ <input type="checkbox"/> No	Not Difficult	
6	Did you think it was easy to make common mistakes while developing the operations?  Did you find yourself making small slips that you felt were irritating?	Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>  <input type="checkbox"/> Yes <input type="checkbox"/> No	Easy	
*	7	How closely related is the notation to the constraints you are describing?	Not Close 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Close
8	How well does the notation depict the constraints you have chosen?	Not Well 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Well	
9	How closely does the concept of a class in this notation map to your expectations of a class represented in other Object-Oriented notations?	Not Close 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Close	
10	How closely does the concept of an object in this notation map to your expectations of an object represented in other Object-Oriented notations?	Not Close 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Close	
11	How closely does the concept of a set of objects in this notation map to your expectations of a set of objects represented in other Object-Oriented notations?	Not Close 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Close	
12	How closely does the concept of a relation in this notation map to your expectations of a relation represented in other Object-Oriented notations?	Not Close 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/>	Close	

## The Usability of Diagrammatic Notations: Developing Constraint Diagrams and OCL

13	<p>When reading the notation how easy is it to tell what each part is for?</p> <p>Were there some parts that were particularly difficult to interpret?</p> <p>Were there parts of the notation you used that you didn't know what they meant?</p>	<p>Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Easy</p> <p><input type="checkbox"/> Yes If so which _____ <input type="checkbox"/> No</p> <p><input type="checkbox"/> Yes If so which _____ <input type="checkbox"/> No</p>
14	<p>How difficult did you find the notation overall to interpret?</p>	<p>Very Difficult 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Not Difficult</p>
15	<p>How helpful would notes made in natural language or another notation be in aiding your interpretation of this notation?</p>	<p>Not Helpful 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Helpful</p>
16	<p>How helpful would adding colour to the notation help your interpretation?</p>	<p>Not Helpful 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Helpful</p>
17	<p>How easy is it to sketch things out when playing with ideas for solving the operations?</p> <p>Are there features of the notation to help you do this?</p>	<p>Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Easy</p> <p><input type="checkbox"/> Yes If so which _____ <input type="checkbox"/> No</p>
18	<p>Do you think you are constrained in any way as to the order you chose for developing your operations?</p>	<p>Not Constrained 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Constrained</p>
19	<p>Did you think that different parts of the notation were similar, i.e. meaning similar things?</p> <p>Did you think any similarities were clear?</p> <p>Were there places where some things ought to be similar but the notation makes them appear different?</p>	<p>Not Similar 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Similar</p> <p><input type="checkbox"/> Yes If so which _____ <input type="checkbox"/> No</p> <p><input type="checkbox"/> Yes If so which _____ <input type="checkbox"/> No</p>
20	<p>How easy do you think it would be to extend the syntax of the notation to incorporate other Object-Oriented or UML constructs?</p>	<p>Not Easy 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> Easy</p>
<p>After completing this questionnaire, can you think of any obvious ways to improve the design of the study?</p>		