Cognitive Dimensions of Use Cases - feedback from a student questionnaire

Karl Cox

Empirical Software Engineering Research Group
School of Design, Engineering and Computing
Bournemouth University
coxk@bournemouth.ac.uk

Keywords: POP-II.B. Use Cases; POP-II.B. Scenario-Based Design; POP-III.C. All Cognitive Dimensions; POP-V.B. Questionnaire Analysis

Abstract

The Unified Modelling Language (UML) has become the standard object-oriented design language in software engineering. An important part of the UML is the use-case notation. This paper reports on results of feedback from a student questionnaire about the use-case diagram notation and use-case descriptions. The questions map to Green's Cognitive Dimensions in an attempt to help understand the strengths and weaknesses of use-cases from the perspective of those actually applying the notation. Feedback from the questionnaire suggests that the notation causes confusion especially concerning the use-case relationships. However, the feedback also shows that the notation is relatively easy to change when moving from diagram to description and back again.

1. Introduction

The use-case notation became popular in software engineering with the publication of Ivar Jacobson's book (Jacobson et al 1992). Since then the most popular software engineering methods have added the use-case notation to their repertoire (e.g. Booch 1994, Rumbaugh 1995). The Unified Modelling Language version 1.1 (Rational 1997), adopted and adapted Jacobson's use-case notation. A newer version of the UML (and use-case notation) - version 1.3 - has been released (Booch et al 1999). Use cases are intended to help the developer elicit and analyse all the functional requirements for a system. The notation is simple and has the advantage that both developers *and* clients should be able to understand it, though semantic weaknesses have been suggested (Cox & Phalp 1999).

There are a large number of different scenario approaches from many fields, such as usability engineering, HCI, participatory design as well as software engineering; for examples see (Carroll 1995). It is necessary to describe why and how scenarios are useful to enable an evaluation of them to occur (Antón & Potts 1999). Recently, classification frameworks have been proposed to describe scenario approaches (Filipiddou 1998, Rolland et al 1998, Antón & Potts 1999). However, these frameworks are perhaps overly complex and are certainly time consuming to complete. The Cognitive Dimensions framework (Green 1989, Green & Petre 1996) provides a simpler and quicker means of describing scenarios than the three scenario classification frameworks referenced. The Cognitive Dimensions framework classifies visual programming languages and design notations from the perspective of the practitioner actually using the notation. This makes classification more relevant to the practitioner in that they know and understand the strengths and weaknesses of the particular notation they are using. The purpose of the dimensions is expressed by Green as: "The dimensions provide a language in which to compare the *form* and *structure* (rather than the content) of notations," (Green 1989). To help understand the use-case notation and attempt to identify its strengths and weaknesses, a questionnaire was issued to students containing questions that map to the Cognitive Dimensions. 14 students were asked to complete a questionnaire on use cases (diagram and description) as part of an assignment. The complete questionnaire can be viewed in Appendix A. The complete results data set can be viewed in Appendix B.

The paper takes the following format: Section 2 describes the use-case notation. Section 3 describes Green's cognitive dimensions. Section 4 discusses the questionnaire and issues in this study. Section 5 describes the results from the questionnaire. Section 6 discusses some interdependencies of cognitive dimensions for the use-case notation. Section 7 draws some conclusions and discusses future work.

2. Use Cases and The Use-Case Notation

In simple terms, the use case is intended to aid the requirements engineer in discovering and describing all of the functional requirements for the system they are developing. The use-case diagram notation is relatively simple, thus making it easy for clients and users to understand. Readers are referred to Booch et al (1999) for a fuller explanation.

Actors, drawn as stickmen, (see figure 1) are considered external to the system being designed. Actors can be other systems and machines as well as human users. The system is bounded to separate what is outside the system from what is inside. The actor begins a use case, interacts with the system during the use case and receives some kind of meaningful response in return as the use case ends. There are normally several use cases and several actors in one use-case diagram.

Relationships between use cases can be added to give a "clearer" understanding of connections among the functions in a system. UML version 1.3 has three types of relationship: <<include>>, <<extend>> and generalisation. The relationships for version 1.3 are shown in figures 2, 3, 4 and 5 (all actors and system boundaries are excluded from figures 3, 4 and 5 for simplification).

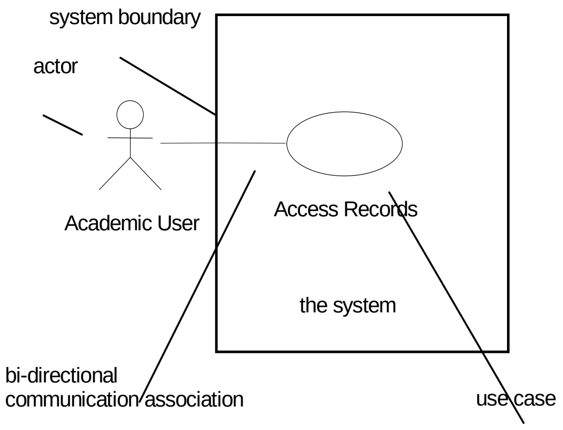

Figure 1: example use-case diagram

There is an inheritance mechanism between actors. This allows an actor to inherit all the functionality of another actor, that is, instantiate use cases, without having to draw the communication association lines another time. Figure 2 shows the Store Manager actor inherits the functionality of the Checkout Operator, thus it is not necessary to draw a line from the Store Manager to the Purchase Products use case.

Figure 3 shows that the use case Track Order makes use of the Validate User use case. Thus, when an actor wishes to Track an Order, they must also 'Validate the User' as part of that function.

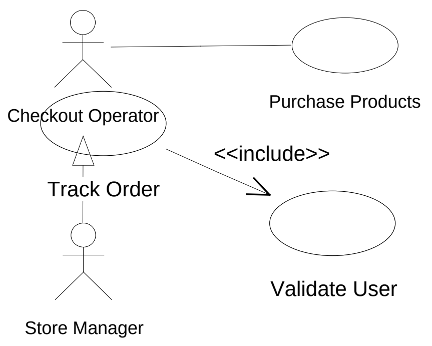

Figure 2: inheritance structure in actors

Figure 3: the <<include>> relationship; taken from (Booch et al 1999)

Figure 4 shows the normal or base use-case Sign Insurance Policy being extended by the exceptional use case Sign Car Purchase Contract. These <<extend>> use cases are exceptions to the normal behaviour of the system or alternative courses through a system that might be used, for example, five times out of a hundred.

Figure 5 shows both <<include>> and *generalisation* relationships. The *generalisation* relationships to the use cases Check Password and Retinal Scan are, in fact, specialisations of the Validate User use case. The *generalisation* relationship imposes a kind of hierarchy of use cases, like an inheritance structure in object-orientation.

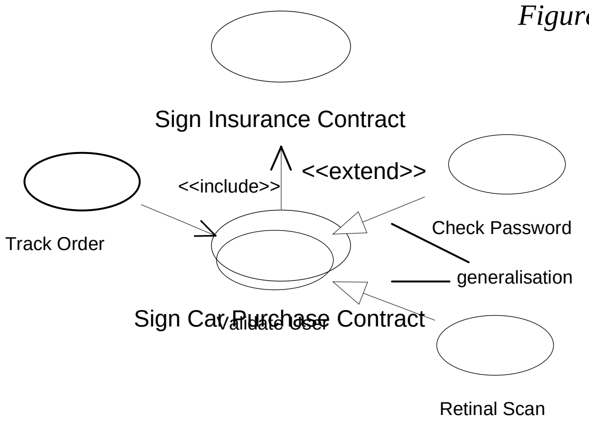

Figure 4: the <<extend>> relationship; adapted from (Eriksson & Penker 1998)

Figure 5: the generalisation relationship; taken from (Booch et al 1999)

3. Cognitive dimensions

The 13 cognitive dimensions of notations listed by Green and Petre are shown below. For "programmer" read "analyst" or "designer":

"*Abstraction gradient:* What are the minimum and maximum levels of abstraction? Can fragments be encapsulated?

Closeness of mapping: What 'programming games' need to be learned?

Consistency: When some of the language has been learnt, how much of the rest can be inferred?

Diffuseness: How many symbols or graphic entities are required to express a meaning?

Error-proneness: Does the design of the notation induce 'careless mistakes'?

Hard mental operations: Are there places where the user needs to resort to fingers or pencilled annotation to keep track of what's happening?

Hidden dependencies: Is every dependency overtly indicated in both directions? Is the indication perceptual or only symbolic?

Premature commitment: Do programmers have to make decisions before they have the information they need?

Progress evaluation: Can a partially-complete program be executed to obtain feedback on 'How am I doing'?

Role-expressiveness: Can the reader see how each component of a program relates to the whole?

Secondary notation: Can programmers use layout, colour, other cues to convey extra meaning, above and beyond the 'official' semantics of the language?

Viscosity: How much effort is required to make a single change?

Visibility: Is every part of the code simultaneously visible (assuming a large enough display), or is it at least possible to juxtapose any two parts side-by-side at will? If the code is dispersed, is it at least possible to know in what order to read it?" (Green & Petre 1996).

The questionnaire does not cover all the dimensions because some are not considered applicable to the use case in this study. The dimensions that are considered are consistency, error-proneness, role-expressiveness, viscosity, visibility and juxtaposability, hard mental operations and hidden dependencies (see sections 5 and 6). The secondary notation dimension was not considered because the UML allows for extensions to the notation such as adorning "notes" to the diagram. In this particular study it was not possible to get any progressive evaluation because the study was part of a student assignment. The students did not go on to design and build the checkout system so it was impossible to judge premature commitment. There are not many symbols to the notation: two types of arrows (linking use-case relationships and inheritance in actors), ellipses (the use case), guillemets <<>> (to surround text e.g. <<extend>>), lines (communication association - between actors and use cases), box ('system boundary' which is optional) and stick men (actors) so diffuseness is not a problem. We did not consider the abstraction gradient dimension in this study because there was only one (small) system under consideration. It can be supposed that, at a textual level of abstraction, the use-case descriptions could quite reasonably describe the functionality of how the final system is going to work from a user perspective and so there is a (abstract) closeness of mapping. In terms of how the final system visually looks, this is not the case.

4. The questionnaire and the students' task

The questionnaire consists of 9 questions, with many of the questions composed of several parts. The first 2 questions are general information questions. Questions 3-9 concern themselves with cognitive dimensions. Question 1 (asking about length of use-case descriptions) can be used as a check to question 7c - there are some discrepancies between the two sets of answers (see Appendix B).

The 14 students who completed the questionnaire are post-graduates studying on a MSc conversion course in Software Engineering at Bournemouth University. The majority of the students have no previous software engineering experience. There are some anomalies to this in that one student has 15 years programming experience and another has 6 years systems analysis experience. However, none of the students had (noticeable) knowledge of use cases or the UML prior to starting the course. The students received a half-day lecture on use cases, which included discussion of abstraction levels, the use-case description as well as a close examination of the UML's use-case notation version 1.3. The students drew example diagrams and wrote example descriptions in this session. As part of a follow-up experiment the students wrote a use-case description, implementing different sets of guidelines to help write the descriptions (Cox & Phalp 2000). The students then had, as part of a continuing assignment, to develop the experiment use-case diagram further to describe the complete system, as well as write descriptions for each use case (the system was a checkout machine for a supermarket). The students had four weeks to complete their assignment (as well as 3 other assignments). They could use any references or tools they considered appropriate. As part of the assignment, the students also had to complete the questionnaire discussed in this paper. There was a nominal reward of 5 marks for completed and returned questionnaires.

An obvious weakness of this study is that there are only 14 participants to get feedback from. Also, because the participants are students it is not valid to apply any generalisms across the software engineering community as a whole.

5. Feedback from the questionnaire

The complete questionnaire can be seen in Appendix A. Questions 1 and 2 are not related to cognitive dimensions so are not discussed here.

5.1 Consistency

Question 3 addresses the Consistency dimension; that is, how much of the notation can be inferred when a certain amount has been learned?

Question 3(a): From an initial understanding of the use-case notation, how much of the rest of the notation can be inferred and used correctly? Table 1 shows the responses from the questionnaire.

All	0 out of 14
Most	4 out of 14
About half	8 out of 14
Some	2 out of 14
None	0 out of 14

Table 1: Answers for question 3(a)

None of the students thought that all of the rest of the notation could be inferred from an understanding of some of it. The majority (57%) responded that about half of the notation could be inferred. It has been suggested that the notation can be counter-intuitive with respect to the relationships and directional arrows (Cox & Phalp 1999).

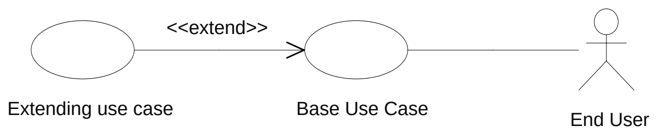

Figure 6: <<extend>>

Figure 6 shows why the <<extend>> relationship is counter-intuitive. As the Extending use case is pointing to the Base use case, we might be forgiven for thinking that the Extending use case has initial control and that the Base is dependent upon the Extending use case. Once control is passed to the Base use case we might still be forgiven for thinking that the Actor is then dependent upon the Base use case. Reality is the reverse. In fact, the Base use case has control and is interrupted by the <<extend>> use case if the condition is right for the base use case to be extended. We would imagine the arrowhead pointing towards the extending use case, not the Base. The UML is clearly counter-intuitive in this example.

The <<include>> relationship is more intuitive (see figure 7) in that the Base Use Case "points" to the use case it wishes to include.

5.2 Error-proneness

Questions 3(b)-3(d) relate to the Error-proneness dimension of the notation: they ask about specific parts of the notation that cause confusion. 3(b) asks about the (probably) most confusing part of the notation, the difference between <<include>> and <<extend>>. Questions 3(c) and 3(d) concern themselves with generalisation and <<extend>>. The results of the questionnaire are shown in table 2.

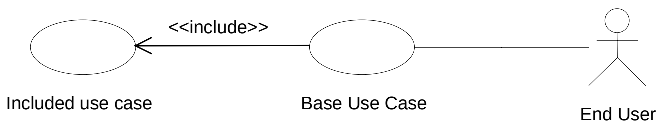

Figure 7: <<include>>

Question 3(b): Did you find you had to rely on your interpretation (at least, in part) of the differences between the <<include>> and the <<extend>> relationships?

Question 3(c): Did you find you had to rely on your interpretation (at least, in part) of how to use generalisation in use cases?

Question 3(d): Did you find you had to rely on your interpretation (at least, in part) of the differences between generalisation and <<extend>> in use cases?

Option	3(b)	3(c)	3(d)
Yes	8 out of 14	6 out of 14	6 out of 14
No	5 out of 14	2 out of 14	2 out of 14
N/A	1 out of 14	6 out of 14	6 out of 14

Table 2: answers for questions 3(b), 3(c) and 3(d)

57% of the participants had problems with the difference between <<include>> and <<extend>>. This is not surprising considering the often confusing nature of these relationships (Jacobson 1994 - here Jacobson discusses the <<uses>> relationship, the earlier version of <<include>>). It was not necessary to implement the generalisation relationship in the assignment. This is probably why 43% responded with "N/A" (not applicable) to questions 3(c) and 3(d). However, 43% also responded to "Yes" for these questions; we assume this to mean that they considered using generalisation in their assignment, though none of the students actually did use it.

Table 3 shows the percentage amount of interpretation thought necessary by participants who answered "Yes" to questions 3(b)-3(d).

Scale	3(b)	3(c)	3(d)
100% your interpretation	2 out of 8	2 out of 6	1 out of 6
75% your interpretation	2 out of 8	1 out of 6	1 out of 6
50% your interpretation	3 out of 8	1 out of 6	2 out of 6
25% your interpretation	1 out of 8	2 out of 6	1 out of 6
< 25% your interpretation	0 out of 8	0 out of 6	1 out of 6

Table 3: percentage interpretation

Table 3 shows there is a fairly even split of percentage interpretation across all questions.

Question 3(e): Did you find that you tried to avoid these relationships because of the uncertainty of their meaning?

Yes	6 out of 12
No	5 out of 12
N/A	1 out of 12

Table 4: number of participants trying to a avoid using use-case relationships

Two of the participants did not answer this question. There is some problem in the understanding of the notation and this is highlighted by the fact that 6 participants tried to avoid using the use-case relationships altogether. However, all the participants used at least one use-case relationship in their assignment answers.

Question 4 also discusses the Error-Proneness dimension by questioning individual parts of the notation. Table 5 shows how many participants had such difficulties. Table 6 shows how regularly participants made mistakes with the notational parts.

Question 4(a): Was using the <<includes>> relationship difficult?

Question 4(b): Was using the <<extend>> relationship difficult?

Question 4(c): Was using the generalisation relationship difficult?

Option	4(a)	4(b)	4(c)
Yes	5 out of 14	5 out of 14	4 out of 14
No	4 out of 14	8 out of 14	3 out of 14
N/A	5 out of 14	1 out of 14	7 out of 14

Table 5: Number of participants finding parts of the notation difficult.

It is interesting that twice as many participants had no difficulty with the <<extend>> than the <<include>> relationship. The <<extend>> is generally considered more difficult to use than the <<include>> relationship. However, 36% of participants responded to "N/A" for the <<include>> relationship, meaning that they did not use it.

Scale	4(a)	4(b)	4(c)
Always	0 out of 5	0 out of 4	1 out of 3
Often	0 out of 5	1 out of 4	1 out of 3
Usually	1 out of 5	1 out of 4	1 out of 3
Occasionally	4 out of 5	1 out of 4	0 out of 3
Rarely	0 out of 5	1 out of 4	0 out of 3

Table 6: rate of occurrence of mistakes made with the notational parts.

For 4(a) one "Yes" participant did not answer the second part but one "No" participant did. ("No" put Occasionally).

For 4(b) two "Yes" participants did not complete the second part but one "No" did ("No" put Rarely).

For 4(c) one "Yes" participant did not complete the second part.

4 out of the 5 participants who answered "Yes" for question 4(a) only had occasional difficulties with using the <<include>> relationship, suggesting it is relatively straightforward to use.

5.3 Role-Expressiveness

Question 5 considers the role-expressiveness dimension: where do individual parts of the notation fit with other parts? Is this expressed in the notation? Table 7 shows the feedback from the questionnaire.

Question 5(a): How easy was it to recognise and know where individual parts of the use-case diagram fit within the whole diagram?

Question 5(b): How easy was it to recognise and know where individual parts of the use-case description fit within the whole description?

Question 5(c): How easy was it to recognise and know where individual parts of the use-case diagram fit in the description?

Question 5(d): How easy was it to recognise and know where individual parts of the use-case description fit in the diagram?

Scale	5(a)	5(b)	5(c)	5(d)
Easy	8 out of 14	10 out of 14	11 out of 14	9 out of 14
Difficult	6 out of 14	4 out of 14	3 out of 14	4 out of 14
Impossibl e	0 out of 14	0 out of 14	0 out of 14	1 out of 14
Don't know	0 out of 14	0 out of 14	0 out of 14	0 out of 14

Table 7: Role-expressiveness of the use-case notation

The feedback shows that the participants have more trouble recognising where parts of the diagram fit than with the description (43% and 29% respectively). One participant thought it impossible to know where parts of the description fit in the diagram. Use-case descriptions have titles that should exactly match the diagram use case. It is therefore easy to know which use case is being described. For individual events in the use case this is more difficult. If the event concerns a use-case relationship such as <<include>> then it is possible to recognise where that event fits because the relationship should be described in the diagram. However, if that event does not concern a relationship then taken in isolation it is unlikely that the event can be placed in the diagram at all.

5.4 Viscosity

Question 6 concerns the ability to make changes in the diagram and description - the Viscosity cognitive dimension. The questions specifically deal with changes from diagram to description and from description to diagram. The participants were not asked about changes in the diagram alone or the description alone because this is considered easy to do. Table 8 shows that 86% of the participants found making a change from diagram to description easy. Going the other way was also quite straightforward: 75% of the participants said this was easy.

Question 6(a): If you made a change in the diagram, how easy was it to move to the corresponding part of the description and make that change there?

Question 6(b): If you made a change in the description, how easy was it to move to the corresponding part of the diagram and make that change there?

Scale	6(a)	6(b)
Easy	12 out of 14	9 out of 14
Difficult	1 out of 14	2 out of 14
Impossible	0 out of 14	0 out of 14
Don't know	1 out of 14	1 out of 14

Table 8: ease of making changes

5.5 Visibility and Juxtaposability

Question 7 considers the Visibility and Juxtaposabililty dimensions. Table 9 shows that the majority of the participants state it is possible to see the diagram on one page, the description on one page and both the diagram and description on the same page. The system in question that the students described is quite small. As this case is entirely specific no generalisms should be made.

Question 7(a): Is it possible to view the diagram and descriptions side-by-side?

Question 7(b): Can you view the whole diagram on one screen/sheet of paper?

Question 7(c): Can you view an entire use-case description on one screen/sheet of paper?

Option	7(a)	7(b)	7(c)
Yes	11out of 14	13 out of 14	10 out of 14
No	2 out of 14	1 out of 14	4 out of 14
N/A	1 out of 14	0 out of 14	0 out of 14

Table 9: Visibility and Juxtaposability

5.6 Hard Mental Operations

Question 8 relates to the Hard Mental Operations dimension. 57% participants thought that linking use cases together was either easy or simple. 36% participants thought it difficult, however (see Table 10).

Question 8: If several use cases are linked together, for example, by some <<extend>>, <<include>> or generalisation relationships, how difficult is it to then comprehend?

Impossible	0 out of 14
Difficult	5 out of 14
Easy	5 out of 14
Simple	3 out of 14
N/A	1 out of 14

Table 10: Hard Mental Operations

5.7 Hidden Dependencies

Question 9 considers the Hidden Dependencies dimension. There is an approximate two thirds to one third split for question 9(a) in favour of visibility in both directions in the diagram (64% to 29%). There is also a similar two thirds to one third split for question 9(b) in favour of visibility in one direction in the description (see Table 11).

Question 9(a): How visible are the dependencies in the diagram? (i.e. one part of the diagram explicitly relies upon / is determined by / uses / requires another part of the diagram.)

Question 9(b): How visible are the dependencies in the description?

Options	9(a)	9(b)
Visible in both directions	9 out of 14	4 out of 13
Visible in one direction	4 out of 14	9 out of 13
Not visible	1 out of 14	0 out of 13

Table 11: visibility of dependencies

For 9(b) one participant did not give an answer (and had responded "Not Visible" in 9(a)).

6. Interdependencies of the Dimensions

It is possible to describe some interdependencies between the dimensions and implications that arise for the use-case notation. The Consistency dimension needs to be improved: 57% of participants answered that they could understand about half the notation from what they had already learned. About half is much better than nothing but there are risks with the use-case relationships. The direction of the <<extend>> arrow to and from use cases is confused and counter-intuitive (see section 5.1). This can lead to more errors, having a negative effect on the Error-proneness dimension and possibly the Hard Mental Operations dimension (see figure 8).

The difficulty in understanding use-case relationships led to 50% (6 out of 12) participants to try to avoid using the relationships all together. This suggests that mistakes were made. The effect on Error-proneness increases and this probably negatively affects many of the dimensions.

There is interdependency between the Visibility and Juxtaposability dimension and the Viscosity dimension. It was possible to put the diagram on one page (79% "Yes"), the description on one page (93% "Yes") and both diagram and description (71% "Yes"). This no doubt made changes relatively easy. Making changes to the diagram did not cause problems (86% said "Easy"). A similar response was given to the description (75% said "Easy"). Because Visibility and Juxtaposability score well there is a positive effect on Viscosity. Making changes from diagram to description was relatively easy (86% said "Easy") and from description to diagram (71% said "Easy"). This interdependency is a strength of the use-case notation in that it is often possible to fit the diagram and description on a page especially if a drawing package is used.

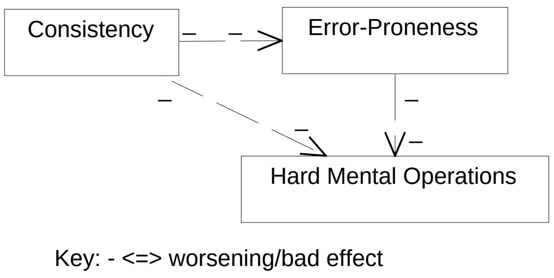

Figure 8: interdependency of Consistency, Error-Proneness and Hard Mental Operations

A goal of notations should be to show dependencies in both directions. The use-case diagram scores reasonably well in that 64% of participants said visibility was in both directions. The description scores poorly with only 29% of participants answering visibility in both directions (64% selected visibility in one direction from descriptions). One assumes that this has an effect on the Role-expressiveness dimension (and vice-versa). The relationship appears to be opposite of what one might think. 43% of participants had problems with the role-expressiveness of the use-case diagram and 29% had problems with the description. It would have been assumed that because of greater visibility of dependencies in the diagram that there would be less difficulty in understanding the expressiveness of the notation. A reason for this apparent contradiction is that the use-case descriptions developed were text-based. Understanding the role-expressiveness of text should be easier to understand than the role-expressiveness of a notation that is essentially new to the experience of the participants in this study.

7. Conclusions and future work

The paper describes an initial and small study into applying the Cognitive Dimensions framework to the UML's use-case notation version 1.3. Feedback about the notation came from a student questionnaire. The use-case notation scores relatively well with the students though there are some problems. This is the well-known problem of understanding and using the differences between the use-case relationships. This issue needs serious consideration before the next version of the UML is released. The difficulty in understanding these relationships means that other dimensions of the notation are negatively affected, such as the Hard Mental Operations dimension. The feedback from the questionnaire shows that 50% of the participants tried to avoid using the use-case relationships. The implications of this are: first, that the notation is difficult to grasp, second, many unnecessary errors will be made and third, revision of the notation needs to be made to clarify this problem.

There are some plus points for use cases: the fact that descriptions and diagrams can often be viewed on one page means that making changes to the diagram and description is relatively straightforward. Also, understanding the links from the diagram to the description and from the description to the diagram makes the notation easier to use. In general, the students found the description fairly expressive with only 29% of the students having problems about where parts of the description fit. The diagram faired worse with 43% having problems understanding where parts of the diagram fit.

The fundamental problem the use-case notation needs to resolve is use-case relationships. Additional notational problems have been described in (Cox & Phalp 1999). Application of the Cognitive Dimensions shows that there are trade-offs between the various dimensions. For example, if the use-case relationships are clarified this should improve some dimensions such as Consistency and Error-proneness. This would also improve the role-expressiveness of the diagram, making Hard Mental Operations less difficult and also improve Hidden Dependencies.

7.1 Critiquing the paper

There are some issues that need further consideration. First, there has been no critique of the questionnaire itself and as such it is unclear how good its design is. Second, there was no feedback from the students about what they thought of the questionnaire. Did the "N/A" (not applicable) responses mean the students did not understand the question? Third, the process of connecting interdependencies between cognitive dimensions needs more clarification. It is highly subjective.

7.2 Future work

Our aim is to apply the Cognitive Dimensions framework to many scenario-based approaches. The key rationale for using the Cognitive Dimension framework is that it will help evaluate the scenario approaches from the point of view of the *practitioners* as opposed to simply a research perspective. We will to carry out further surveys of scenario use by means of questionnaire to both students and industry to try to build a picture of the strengths and weaknesses of different scenario approaches from the perspective of those actually using the notation.

References

Antón A.I. and C. Potts (1998), A Representational Framework for Scenarios of System Use, *Requirements Engineering Journal*, 3 (3/4), 219-241.

Booch G. (1994), Scenarios, *Report on Object Analysis and Design*, 1 (3), 3-6.

Booch G., Rumbaugh J.and I. Jacobson (1999), The Unified Modelling Language User Guide, Addison-Wesley.

Carroll J.M. ed. (1995), Scenario-Based Design: Envisioning Work and Technology in System Development, Wiley.

Cox K. and K. T. Phalp. (1999), Semantic and Structural Difficulties with the Unified Modelling Language Use-Case Notation version 1.3, *OOPSLA'99 workshop on Rigorous Modelling and Analysis of the UML: Challenges and Limitations*, Denver, Colorado, USA, 2 November 1999.

Cox K. and K. T. Phalp (2000), Use Case Authoring: Replicating the CREWS Guidelines Experiment, *EASE'2000 - 4th International Conference on Empirical Assessment and Evaluation in Software Engineering*, Keele University, Staffordshire, UK, April 17th-19th 2000.

Eriksson H. and M. Penker (1998), UML Toolkit, John Wiley.

Filippidou D. (1998), Designing with Scenarios: A Critical Review of Current Research and Practice, *Requirements Engineering Journal*, 3 (1), 1-22.

Green T.R.G. (1989), Cognitive Dimensions of Notations, in *People and Computers V, Proceedings of the Fifth Conference of the British Computer Society Human Computer Action Specialist Group*, 5-8 September 1989, edited by A. Sutcliffe and L. Macaulay, Cambridge University Press, 443-460.

Green T.R.G. and M. Petre (1996), Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework, *Journal of Visual Languages and Computing*, 7, 131-174.

Jacobson I., Christerson M., Jonsson P. and G. Oevergaard, (1992), Object-Oriented Software Engineering: A Use Case Driven Approach, Addison Wesley.

Jacobson I. (1994), Basic Use-Case Modelling (continued), *Report on Object Analysis and Design*, 1 (3), 7-9.

Rational Software Corporation (1997), UML Notation version 1.1, http://www.rational.com/uml

Rolland C., Ben Achour C., Cauvet C., Ralyté J., Sutcliffe A., Maiden N., Jarke M., Haumer P., Pohl K., Dubois E. and P. Heymans, (1998), A Proposal for a Scenario Classification Framework, *Requirements Engineering Journal*, 3 (1), 23-47.

Rumbaugh J. (1994), Getting Started: Using use cases to capture requirements, *Journal of Object-Oriented Programming*, September, 8-23.

Yourdon E. (1989), Modern Structured Analysis, Prentice Hall.

Appendix A - questionnaire

Questionnaire about Use-Case Notation

Please tick only one box per question. (N/A ⇔ Not Applicable)

1. How long (on average) were the use case descriptions in terms of number of pages?

- ☐ 1 page or less
- ☐ 1-2 pages
- ☐ 2-5 pages
- ☐ More than 5 pages

2. How long did you spend on average developing each use case?

- ☐ 1 day or less

- ☐ 1-2 days
- ☐ More than a 2 days

3(a) From an initial understanding of the use-case notation, how much of the rest of the notation can be inferred and used correctly?

- ☐ All
- ☐ Most
- ☐ About half
- ☐ Some
- ☐ None

3(b) Did you find you had to rely on your interpretation (at least, in part) of the differences between the <<include>> and the <<extend>> relationships?

☐ Yes ☐ No ☐ N/A

If yes, how much did you have to rely on your interpretation?

- ☐ 100% your interpretation
- ☐ 75% your interpretation
- ☐ 50% your interpretation
- ☐ 25% your interpretation
- ☐ Less than 25% your interpretation

3(c) Did you find you had to rely on your interpretation (at least, in part) of how to use generalisation in use cases?

☐ Yes ☐ No ☐ N/A

If yes, how much did you have to rely on your interpretation?

- ☐ 100% your interpretation
- ☐ 75% your interpretation
- ☐ 50% your interpretation
- ☐ 25% your interpretation
- ☐ Less than 25% your interpretation

3(d) Did you find you had to rely on your interpretation (at least, in part) of the differences between generalisation and <<extend>> in use cases?

☐ Yes ☐ No ☐ N/A

If yes, how much did you have to rely on your interpretation?

☐ 100% your interpretation
☐ 75% your interpretation
☐ 50% your interpretation
☐ 25% your interpretation
☐ Less than 25% your interpretation

3(e) Did you find that you tried to avoid these relationships because of the uncertainty of their meaning?

☐ Yes ☐ No ☐ N/A

4(a) Was using the <<includes>> relationship difficult?

☐ Yes ☐ No ☐ N/A

If yes, how often did you make mistakes with this?

☐ Always
☐ Often
☐ Usually
☐ Occasionally
☐ Rarely

4(b) Was using the <<extend>> relationship difficult?

☐ Yes ☐ No ☐ N/A

If yes, how often did you make mistakes with this?

☐ Always

☐ Often
☐ Usually
☐ Occasionally
☐ Rarely

4(c) Was using the generalisation relationship difficult?

☐ Yes ☐ No ☐ N/A

If yes, how often did you make mistakes with this?

☐ Always
☐ Often
☐ Usually
☐ Occasionally
☐ Rarely

5(a) How easy was it to recognise and know where individual parts of the use-case diagram fit within the whole diagram?

☐ Easy
☐ Difficult
☐ Impossible
☐ Don’t know

5(b) How easy was it to recognise and know where individual parts of the use-case description fit within the whole description?

☐ Easy
☐ Difficult
☐ Impossible
☐ Don’t know

5(c) How easy was it to recognise and know where individual parts of the use-case diagram fit in the description?

☐ Easy
☐ Difficult

☐ Impossible
☐ Don’t know

5(d) How easy was it to recognise and know where individual parts of the use-case description fit in the diagram?

☐ Easy
☐ Difficult
☐ Impossible
☐ Don’t know

6(a) If you made a change in the diagram, how easy was it to move to the corresponding part of the description and make that change there?

☐ Easy
☐ Difficult
☐ Impossible
☐ Don’t know

6(b) If you made a change in the description, how easy was it to move to the corresponding part of the diagram and make that change there?

☐ Easy
☐ Difficult
☐ Impossible
☐ Don’t know

7(a) Is it possible to view the diagram and descriptions side-by-side?

☐ Yes ☐ No ☐ N/A

7(b) Can you view the whole diagram on one screen/sheet of paper?

☐ Yes ☐ No ☐ N/A

7(c) Can you view an entire use-case description on one screen/sheet of paper?

☐ Yes ☐ No ☐ N/A

8. If several use cases are linked together, for example, by some <<extend>>, <<include>> or generalisation relationships, how difficult is it to then comprehend?

☐ Impossible
☐ Difficult
☐ Easy
☐ Simple
☐ N/A

9(a) How visible are the dependencies in the diagram? (i.e. one part of the diagram explicitly relies upon / is determined by / uses / requires another part of the diagram.)

☐ Visible in both directions
☐ Visible in one direction
☐ Not visible

9(b) How visible are the dependencies in the description?

☐ Visible in both directions
☐ Visible in one direction
☐ Not visible