ACIS 2002 PAPER SUBMISSION COVER-PAGE

Paper Title:	Using RECOCASE to Compare Use Cases from Multiple Viewpoints		
Author Name(s):	Debbie Richards, Kathrin Boettger and Anne Britt Fure		
Contact Author: Email address: Fax: Phone:	**Debbie Richards** **richards@ics.mq.edu.au** **02 9850 9551** **02 9850 9567**		
Paper Category (circle one):		Research in Progress	
Prior Publication (delete not applicable):	The paper has NOT been presented or published elsewhere		

AUTHOR DECLARATION

By submitting this paper to ACIS 2002 the author(s) warrant that:

- they are the copyright holders in the paper;
- the paper is accurate in all factual respects;
- the contents have been cleared for publication by any organisation having the capacity to preclude their publication;
- the approval has been secured of the copyright owner of any material which has been previously published.
- they accept full responsibility for statements made both in written papers and in presentations;
- they undertake that at least one of the authors will register for the, will present the paper, and will be available to answer questions and discuss the paper with other delegates;

(Please leave blank, reviewer use only)

Using RECOCASE to Compare Use Cases from Multiple Viewpoints

Debbie Richards
Kathrin Boettger
Anne Britt Fure

Department of Computing
Division of Information and Communications Sciences
Macquarie University
Sydney, Australia
Email: richards@ics.mq.edu.au

Abstract

Incomplete and inconsistent requirements are a major cause of the failure of computer-based projects. The capture of requirements from multiple viewpoints has been offered as a way of developing a more complete, consistent and representative set of requirements. Our viewpoint development approach, known as RECOCASE, includes a Computer Aided Software Engineering (CASE) tool to assist the capture and RECOncilation of viewpoints of functional requirements. This project seeks to offer three significant solutions to the problems of requirements elicitation, validation and reconciliation: requirements will be captured from multiple viewpoints, directly from stakeholders in natural language and then compared and reconciled through visualisation of the requirements.

Keywords

FB03 Information Requirements; FC15 IS models; HB26 Simulation and modelling IS; Formal Concept Analysis.

1. INTRODUCING THE PROJECT

Incomplete and inconsistent requirements are a major cause of the failure of computer-based projects. The capture of requirements from multiple viewpoints has been offered as a way of developing a more complete, consistent and representative set of requirements. Our viewpoint development approach, known as RECOCASE, includes a Computer Aided Software Engineering (CASE) tool to assist the capture and RECOncilation of viewpoints of functional requirements. This project seeks to offer three significant solutions to the problems of requirements elicitation, validation and reconciliation: requirements will be captured from multiple viewpoints, directly from stakeholders in natural language and then compared and reconciled through visualisation of the requirements. RECOCASE is a comprehensive viewpoint development methodology which includes use case description guidelines, a controlled language to support natural language translation, a conflict resolution process model, a group decision support approach and a tool to assist the specification and reconciliation of requirements.

1.1 Capturing multiple viewpoints

A major problem with requirements determination is that each and every stakeholder has his/her own representation of the enterprise reality. Hence, the possibility of conflicting requirements is highly common. Capturing multiple viewpoints can provide: easier future modification, enhanced communication, the ability to replay and retrace requirement's changes, a more representative specification, a means of conflict identification and resolution (Easterbrook 1991, Ramesh and Dhar 1992). A sense of ownership of the requirements and resulting system is also achieved. The RECOCASE methodology includes a process model that allows individual stakeholders to privately own and defend their own set of requirements. Our negotiation strategies and resolution operators are applied to produce a shared set of requirements that would be more complete and representative than a single set of requirements.

1.2 Automatic translation of natural language use case descriptions into computer processable format.

While a number of viewpoint development approaches exist (e.g. Darke and Shanks 1997, Easterbrook 1991, Easterbrook and Nuseibeh 1996, Kotonya and Sommerville 1998, Mullery 1979) our approach is different as it takes input in natural language directly from viewpoint agents (owners of a viewpoint) in the form of use cases and automatically outputs concept hierarchies to assist requirements reconciliation. These other approaches either do not provide automatic processing of the requirements or they expect input in tabular or logical form. Our approach offers acquisition of requirements via use cases (Jacobson 1992) which is developing as the industry standard.

The poor uptake of formal methods in RE is due to the tedious and expensive process of having a trained professional provide the inputs. Also, formal methods typically provide verification of the requirements but validation is difficult because formal requirements can not be understood by users or relies on the professional having a full understanding of the needs of the stakeholders. Our natural language work will not only benefit our research but our approach may be used as a source of formal requirements for other formal methods approaches.

1.3 Visualisation of requirements as a line diagram

The use of diagrams has become increasingly popular in the design of computer systems. The use of the Unified Modelling Language (UML) for the specification of object-oriented systems has become the standard. The specification of requirements in the UML is done via use cases. However, UML only provides visualisation of the use case at a very abstract level. To get the details of a chunk of functionality it is necessary to analyse the textual use case descriptions. While sequence diagrams are used to model the use case, they model the interaction between the objects needed to support the functionality. It is not appropriate or feasible to ask users to provide use case input in the form of a sequence diagram. Our approach allows the user to specify the steps in natural language, which is obviously feasible due to the popularity of the approach, and then to generate a model which allows comparison. In our approach we take the natural language sentences for each stakeholder viewpoint, automatically translate them into flat logical forms and use Formal Concept Analysis (FCA) (Wille 1982, 1992) for concept generation and structuring into a concept lattice. A large body of theoretical and applied international research has grown from Wille's earlier and current work. FCA has been widely and successfully applied to many problems such as information retrieval (Priss and Old 1998), data mining (Faid, Missaoui, and Godin 1997) and software reengineering (Snelting 2000). However, our application of FCA to the reconciliation of requirements is novel and seeks to take advantage of many of the nice features offered by FCA that are relevant to this application.

In the next section we introduce our approach particularly focusing on an introduction to requirements capture and the use of FCA in our approach. In Section 3 we describe some evaluation we have conducted. Our conclusions are given in Section 4.

2. APPROACH

The RECOCASE viewpoint methodology includes the following six iterative phases:

1. Requirements acquisition,
2. Requirements translation,
3. Concept generation,
4. Concept comparison and conflict detection,
5. Negotiation and
6. Evaluation,

Our process begins with the identification of use cases. Multiple use case descriptions are entered by viewpoint agents, one for each use case viewpoint (Phase 1). A viewpoint agent is the owner or representative of a viewpoint. It may correspond to an instance of an actor if each viewpoint represents an actor. The sentences are automatically translated into flat logical forms using ExtrAns (Molla et al. 2000) based on LinkGrammar (Sleator and Temperley 1991) and used to create crosstables (Phase 2). We apply FCA to the crosstables and produce concept lattices (Phase 3). Using the FCA notion of a concept, concept lattice and their visual representation as a line diagram we compare the requirements (Phase 4). To reconcile differences we apply our resolution operators and negotiation strategies (Phase 5). Using graph theory on the lattices we determine the distance between viewpoints to see whether another round of negotiations are necessary which begins another

cycle of the process (Phase 6). In the next two subsections we first consider the capture of requirements as use cases and then their representation as a concept lattice.

2.1 Capturing And Structuring Requirements.

To perform comparison we must ensure that we are comparing equivalent things. This means that the requirements must be grouped in some way. Use cases seemed a natural choice and a technique that is becoming widely accepted in theory and practice. A use case represents a complete course of events in a system from the user's perspective. A use case describes the interaction between the system and an actor. Jacobson uses the term actor to refer to the role played in relation to the system and can include an individual, group, another system or hardware device. Using the terminology of object-oriented software development, scenarios are instances of use cases. A scenario is a concrete, focused and informal description of one possible behaviour of the system interacting with an actor. Scenarios are formalized into use cases. Possible use cases for an ATM include withdrawing cash, depositing funds, transferring funds, checking balance, and validating customer. With a visualization of the use cases before them the group identify viewpoints and a representative, probably from within the group, for each viewpoint.

By getting each viewpoint agent to enter use case descriptions for each use case we could compare each description for consistency and completeness. To enhance comparison and to assist automatic conversion from natural language to tabular form, we developed a controlled language and use case structure. Our structure is compatible with most use case description formats found in object oriented system development textbooks. Use cases are divided into several parts: actors, trigger, pre- and postconditions and the flow of actions. The postconditions are further divided into success postconditions and failed postconditions. The flow of actions is divided into a main success scenario, an extension part of the main success scenario and a variation part of the main success scenario. Table 1 gives a description of each part.

Part of the use case	Description
Actors	The roles which a human or nonhuman can play using the system functionality described in the use case.
Trigger	The action upon the system that starts the use case.
Preconditions	The conditions which have to be valid before the use case can be started.
Success preconditions	The state of the world after successful completion of the use case.
Failed postcondition	The state of the world after the goal was abandoned.
Main success scenario	The steps of a flow of actions from trigger to goal delivery.
Extensions	The steps which extend the main success scenario.
Variations	Variations of steps of the main success scenario.

Table 1: Structure of Use Cases

2.2 Turning Use Cases Into Concept Lattices

The potential value of FCA for RE lies in the use of term subsumption, the notion of a concept, the automatic generation of concepts and their structure into an abstraction hierarchy and the visualisation of the concepts. A formal concept in FCA is a pair of a set of objects and the set of attributes shared by those objects. The set of objects is known as the concepts *extent* and the set of attributes is the concepts *intent*. In FCA a crosstable is known as a formal context which is made up of the set of objects, the set of attributes and the 'incidence' relationship between those objects and attributes. In the crosstable shown in Fig. 2, the incidence relationship between an object and an attribute is shown by an 'X'. Each row in the crosstable is an object and a low-level concept. To find higher-level concepts we take the intersections of shared attributes and the set of objects which share those attributes. In our usage of FCA an object corresponds to a use case step/sentence. The attributes are the words and phrases output by ExtrAns which make up the sentence. ExtrAns is an answer extraction system that has been developed by Molla et al. 2000. ExtrAns uses LinkGrammar to get the syntactic structures of an input document's sentences. LinkGrammar is a parser of English based on the link grammar theory. It returns all alternative syntactic dependencies between the words of a sentence. The syntactic structure consists of a set of labeled links connecting pairs of words. We have used ExtrAns to find flat logical forms which provide the noun and verb phrases to form the attribute columns in the crosstable. The technique is described in Boettger et al. (2001).

Column 1 of Figure 1 shows the identity of object/sentence which consists of the step number and the viewpoint agent identity (e.g. "1 – agent A" is step number one written by Agent A). Figure 2 shows the line diagram for the context table in Figure 1. Higher level concepts show shared terms between sentences. Using term subsumption we can order the concepts to produce a concept lattice. Due to the limitations of the amount of data that can be displayed graphically and the limited cognitive capacity of humans, we allow the user to select which requirements they wish to model and compare. The crosstable in Figure 1 below includes the sentences using the term 'customer' from three viewpoints for the "withdrawing cash" use case. The sentences selected answer the question "What is the customer involved in?" Each sentence is a low-level concept. To find shared words and phrases we take the intersections of attributes and then determine the corresponding set of objects.

	customer	insert	ATM card	In ATM	enter	code	amount	If customer want receipt	Into card reader	PIN number	amountX	select	Account number	Into ATM	PIN	Customer want to get amount
1-A-%ATM	X	X	X	X												
6-A-%ATM	X			X	X	X										
11-A-%ATM	X			X	X		X									
16-A-%ATM	X				X			X								
1-B-%ATM	X	X	X						X							
5-B-%ATM	X				X					X						
18-B-%ATM	X				X						X					
19-B-%ATM	X											X	X			
1-C-%ATM	X	X	X											X		
3-C-%ATM	X				X										X	
5-C-%ATM	X				X	X										X

Fig. 1: The crosstable (formal context) for the question "what is the customer involved in ?" for the ATM example.

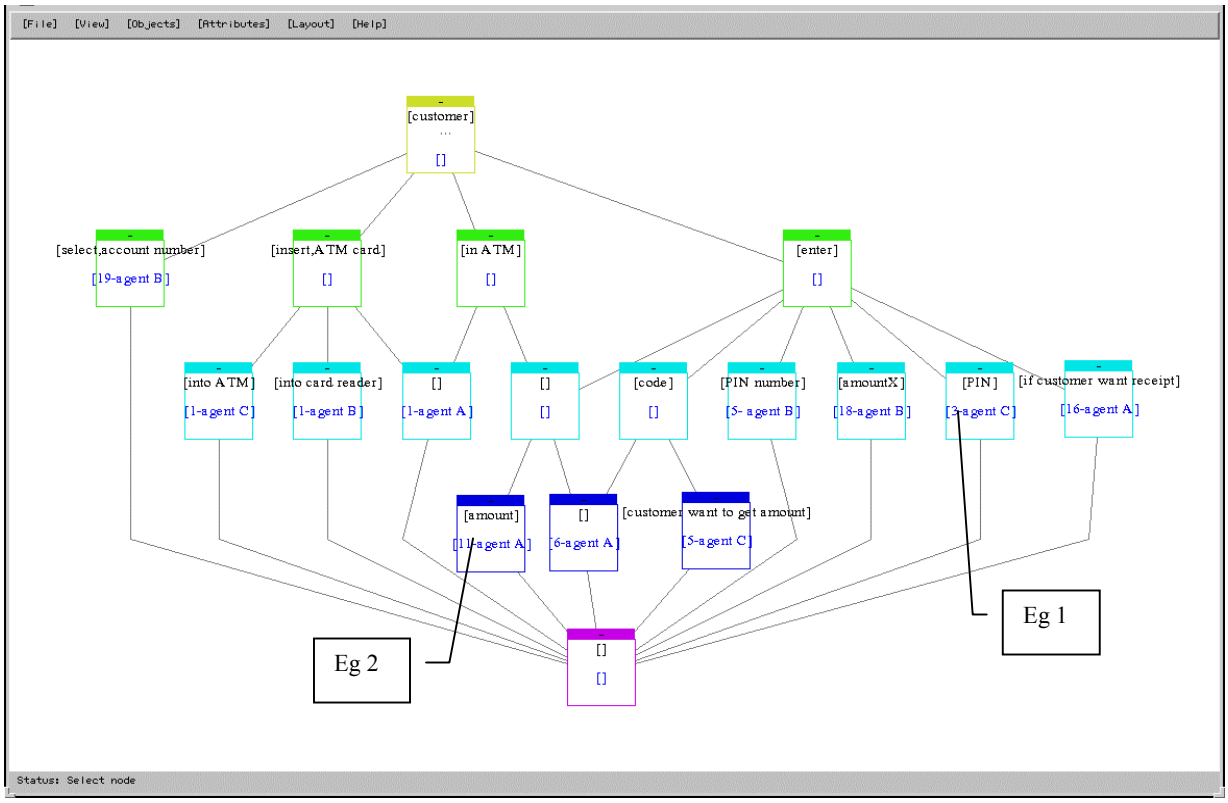

Fig. 2: The concept lattice for the question "what is the customer involved in ?" for the ATM example.

The project leader, or someone else familiar with the approach, would draw a number of line diagrams based on the combined viewpoints which would then be reviewed by the project team. By looking at the lattice we can determine what concepts are shared and what differences exist between the viewpoints. To read the line diagram start at the bottom nodes to find the agent who is the owner of the sentence, pick up the term in that node and then pick up all terms that can be reached by all ascending paths to get the complete sentence. For example, look at the two nodes we have labeled "eg. 1" and "eg. 2". "Eg. 1" represents sentence number 3 written by agentC "The customer enters [the] PIN". "Eg. 2" has more a complex structure because several parts of the sentence are shared by other sentences. "Eg. 2" represents sentence number 11 by agentA "The customer enters [the] amount in [the] ATM". We can look at the diagram to see what conflicts exist. If we look at the left side of Fig. 2 we can see that only Agent B (in sentence 19) states that the customer select an account number. Agent B would need to clarify what they mean. Perhaps they mean the type of account such as savings, cheque or credit. Or perhaps they forgot that the ATM would get the account number from the card. Just to the right of that node, we see a number of shared concepts. Agents A, B and C all agree that the customer inserts the ATM card. However, agentC states it is inserted "into [the] ATM", agentB specifies "into [the] card reader" and agentA specifies "in [the] ATM". These sentences are clearly expressing the same notion. We can either update the viewpoints with a shared term or enter a mapping between terms in a table of synonyms, hypernyms and hyponyms that the user can pop up. Our approach also allows concepts to be tagged as "ignored, delayed or circumvented" so that the conflicts can be managed in one of these three ways (Richards and Menzies 1998). After fixing the left side, we can then go on to detecting and reconciling conflicts on the right side of the diagram. Other line diagrams can be drawn and compared by specifying which sentences and/or terms to include in the crosstable.

3. EVALUATION RESULTS-TO-DATE

We have chosen to use a visual representation of the individual and shared requirements models as a central part of the RECOCASE methodology. The utitlity of the approach thus hinges on the usefulness and usability of the concept lattice. We have just conducted evaluations of our use case guidelines, the readability and usefulness of the line diagram for reasoning about requirements with 201 second year requirements, analysis and systems design students. We chose students as we had access to them and they were likely future members of development teams that would use the RECOCASE approach. We believe if we can get students at the beginning of second year who are just learning about analysis and design to successfully use the technique we could expect better results in industry. At the start of the evaluation students were given a one paragraph problem description concerning a student accommodation booking service. We did not give a comprehensive problem statement as the task to be performed only concerned one part of the system functionality and we did not want to overload the students with unnecessary details. As a group we brainstormed some use cases. For **task 1** we asked the student to write the main flow of events for the "Booking Room" use case based on the problem description. We specified not to include processing which may occur before or after this use case (e.g. logging into a network, starting up a web browser, updating room availability, processing of the credit card, contacting the resort, etc). Half of the students were given extra time to read the guidelines that we created to improve the output of ExtrAns. The other half were not given these guidelines. The entry and analysis of results to determine if *better* descriptions were provided with or without the guidelines are not complete and will be presented in a future paper. However, our results-to-date show 40/79 felt confident using the guidelines, 25/79 were neutral and 14/79 lacked confidence. 52/79 found the guidelines helpful, 18/79 were neutral and 27/79 did not find the guidelines helpful. Another 18 students had the guidelines but did not fill in the comments section. We measured if having the guidelines affected whether subjects were more likely to conform to the fourteen points in our guidelines. We found that subjects with guidelines were more likely to use the same word to refer to the same thing and avoid use of pronouns, modal verbs, adverbs, conjunctions and disjunctions. We will be performing comparisons of our results with the guideline evaluations conducted by Cox 2001.

Once we had collected use case descriptions from each student, we showed the students a possible solution taken from one of the responses to the pilot study we had conducted earlier with 11 students. We then showed and explained two concept lattices containing those sentences. The first lattice showed all the sentences. The second lattice showed selected sentences from that viewpoint plus another viewpoint so that the sentences could be compared. For **task 2** we gave the students a different lattice based on two other student viewpoints and asked them to write the sentences and the owner/source of that sentence so that we could test whether and how quickly they could learn to read a concept lattice. The results of this task would indicate if the results of tasks 3 and 4 were likely to be valid.

The students were divided into Group A and Group B. For **task 3** Group A were given sentences for the Agent A and Agent B viewpoints. Group B were given a diagram of the sentences. Both groups were asked to answer the same questions and record the time it took to answer each question. For **task 4** Group B were given sentences for the Agent D and Agent E viewpoints, shown in Figure 3 and Group A were given a diagram, shown in Figure 4. The same questions were asked this time regarding Agents D and E. The rationale behind this design was to evaluate whether a textual or graphical representation would be quicker and/or more accurate for comparison of requirements. We got everyone to try both representations on similar but not identical sentences (as they would know the answers from the first task) since some people are more visually oriented than others. Further anaylsis is being performed, but our results-to-date show that reading and reasoning with the line diagram could be learnt by 58% of our subjects after a 5 minute introduction, questions were up to 80% more likely to be correct when using the diagram as opposed to textual sentences and that 61% of students preferred using the line diagram over sentences to answer the questions. Answering the questions using the diagrams was up to 9.9 times faster. In the coming months an evaluation will be designed and conducted to test how well the RECOCASE-tool supports our group decision process and assists requirements reconciliation. We will perform indepth comparison of related work concerning requirements and natural language (eg. Ambriola and Gervassi 1998) and group processes (eg. Al-Ani, Leaney, and Lowe 1999).

Agent D

step	action
1	the system displays list of resorts
2	the student views the list of resorts
3	the student selects a resort
4	the system asks for the number of people
5	the student enters the number of people
6	the system displays a list of rooms and rates
7	the student selects a room
8	the system asks for any special requests
9	the student enters any special requests
10	the system asks the user to reconfirm all the details
11	the student reconfirms the details
12	the system sends an email to the student to confirm booking

Agent E

step	action
1	the student enters contact details
2	the student enters the time and destination
3	the student enters number of people
4	the student enters special requests
5	the student selects a payment method
6	the system checks the availability of the room requested
7	the system sends an email to the student to confirm the booking

Fig. 3: Use case descriptions for the “Booking Room” use case from Agent D and E viewpoints.

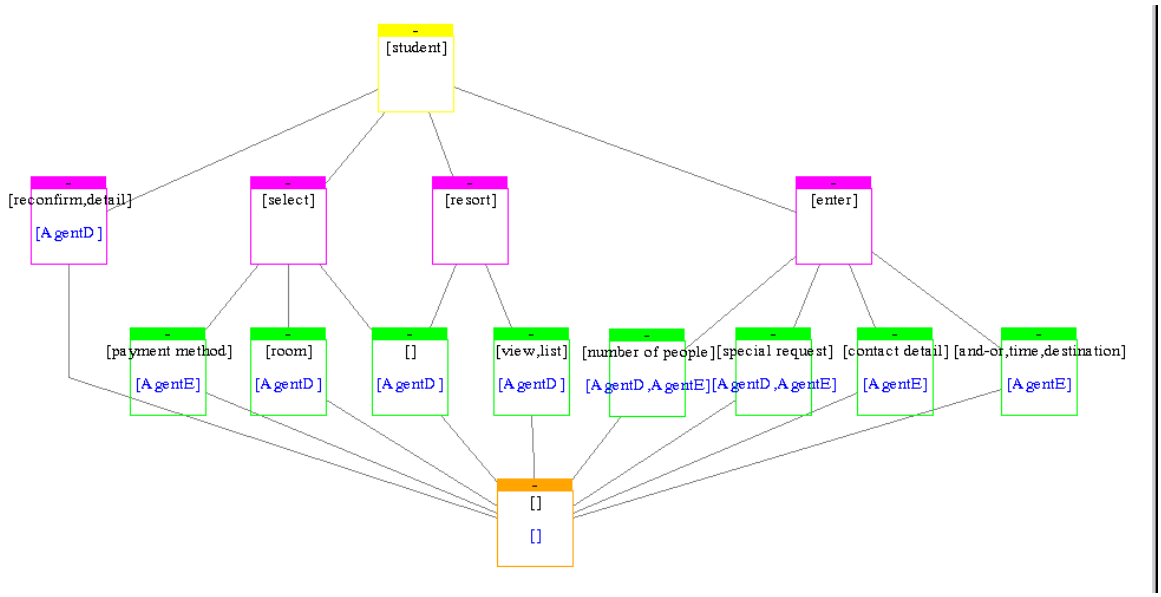

Fig. 4: Line diagram for the sentences using the use case descriptions for the “Booking Room” use case from Agent D and E viewpoints.

4. CONCLUSION

In Section One we mentioned the benefits offered by the RECOCASE approach including handling multiple viewpoints, accepting and formalising requirements in natural language and visualisation of requirement's viewpoints. In addition RECOCASE is a comprehensive methodology including use case description guidelines, a controlled language to assist natural language translation, a conflict resolution process model, a group decision support approach and a tool.

Expected benefits to research in the field of RE in general include: a set of use case guidelines, an automated technique to support the conversion of NL specifications into table format that could be applied to other formal approaches that assume requirements in logical or tabular form and comparative studies and experiments of all aspects of the RECOCASE methodology. Through empirical evaluation this study will also reveal more about the intrinsic nature of requirements and conflict resolution, the nature of differences between requirements and what we can realistically expect of stakeholders.

REFERENCES

Al-Ani, B., Leaney, J., and Lowe, D. (1999) A Taxonomy of Partially Excluded Service Descriptions *Proceedings of the Fourth Australian Conference on Requirements Engineering (ACRE'99)* 29-30 September, 1999, Macquarie University, Sydney, 15-26.

Ambriola, V. and Gervassi, V. (1998) The Case for Cooperative Requirement Writing. ECOOP Workshops 1998: 477-479

Boettger, K., Schwitter, R., Richards, D., Aguilera, O. and Molla, D. (2001) Reconciling Use Cases via Controlled Language and Graphical Models, The Proceedings of the 14th International Conference on Applications of Prolog, (INAP'2001), 20-22 October, 2001, University of Tokyo, Japan, pp. 186-195.

Cox, K. (2001) Experimental Material, http://dec/bournemouth.ac.uk/staff/kcox/UCwriting.htm.

Darke, P. and Shanks, G. (1997) Managing User Viewpoints in Requirements Definition, 8th Australasian Conference on Information Systems.

Easterbrook, Steve (1991) Elicitation of Requirements from Multiple Perspectives PhD Thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, London.

Easterbrook, S. and Nuseibeh, B. (1996) Using Viewpoints for Inconsistency Management BCSEEE Software Engineering Journal January 1996, pp. 31-43.

Faid, M., Missaoui, R. and Godin, R. (1997) Mining Complex Structures Using Context Concatenation in Formal Concept Analysis, *International KRUSE (Knowledge Retrieval and Storage for Efficiency) Symposium* August 11-13, Vancouver, BC.

Finkelstein, A.C.W., Goedicke, M., Kramer, J. and Niskier, C. (1989) Viewpoint Oriented Software Development: Methods and Viewpoints in Requirements Engineering In Proceedings of the Second Meteor Workshop on Methods for Formal Specification Springer Verlag, LNCS.

Jacobson, I. (1992) *Object-Oriented Software Engineering*, Addison-Wesley.

Kotonya, G. and Sommerville, I. (1998) Requirements Engineering: Process and Techniques Chichester, UK, Wiley and Sons.

Molla, D., Schwitter, R., Hess, M. and Fournier, R. (2000) Extrans, an answer extraction system *T.A.L special issue on Information Retrieval oriented Natural Language Processing*.

Mullery, G. P. (1979) CORE - a method for controlled requirements expression In Proceedings of the 4th International Conference on Software Engineering (ICSE-4), IEEE Computer Society Press, 126-135.

Mullery, G. P. (1979) CORE - a method for controlled requirements expression *In Proceedings of the 4th International Conference on Software Engineering (ICSE-4)*, IEEE Computer Society Press, 126-135.

Priss, U. and Old, J. (1998) Information Access through Conceptual Structures and GIS In *Information Access in the Global Information Economy Proceedings of the 61st Annual Meeting of ASIS'*98 :91-99 also http://php.indiana.edu./~upriss/papers/asis98.ps

Ramesh, B. and Dhar, V. (1992) Supporting Systems Development by Capturing Deliberations During Requirments Engineering IEEE Transactions on Software Engineering 18(6):498-510.

Richards, D. and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirement Engineering Task, *11th Workshow on Knowledge Acquisition, Modeling and Management*, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Banff, Canada, 18th-23rd April.

Richards, D. and Zowghi, D. (1999) Maintaining and Comparing Requirements, *Proceedings of the Fourth Australian Conference on Requirement Engineering* ACRE'99, Macquarie University, Sydney, 29-30 September, 1999

Sleator, D. D. and Temperley, D. (1991) Parsing English with a Link Grammar, *Technical Report* CMU-CS-91-196, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA.

Snelting, G. (2000) Software Reengineering Based on Concept Lattices Invited Contribution, *Proceedings of 4th European Conference on Software Maintenance and Reengineering.*

Wille, R. (1982) Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts In I. Rival (ed), Ordered Sets Reidel, Dordrecht, Boston, 445-470.

Wille, R. (1992) Concept Lattices and Conceptual Knowledge*, Computers and Mathematics with Applications,* 23: 493-522.

ACKNOWLEDGEMENTS

Thanks to Oscar Aguilera for his work on the RECOCASE-tool, Rolf Schwitter and Diego Molla-Alloid for their assistance with ExtrAns and other language aspects of this project. This work is funded by a grant from the ARC.

COPYRIGHT

Richards and Boettger © 2001. The authors assign to ACIS an educational and non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The authors also grant a non-exclusive licence to ACIS to publish this document in full in the Conference Papers and Proceedings. Those documents may be published on the World Wide Web, CD-ROM, in printed form, and on mirror sites on the World Wide Web. Any other usage is prohibited without the express permission of the authors.