A Process Model for Requirements Elicitation

Debbie Richards

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

Abstract

Despite improvements in various activities making up the software development process, the elicitation, analysis and modelling of user requirements remain as one of the least explored and have the least satisfactory scientific foundations. The approach to be explored takes low-level requirements, which can be expressed in crosstable format, and uses Formal Concept Analysis to automatically generate a concept lattice. A process model is proposed which uses the lattice to assist in the identification, negotiation and reconciliation of requirements with RE practitioners and stakeholders. The early detection and correction of errors offers the greatest potential for avoiding cost overruns in the development of information systems.

Keywords

AL01 Knowledge representation; AL04 Knowledge acquisition; FB03 Information Requirements; FC15 IS models; HB26 Simulation and modelling IS; Formal Concept Analysis, Ripple Down Rules

1 REQUIREMENTS ENGINEERING – A HARD PROBLEM IN A SOFT AREA

In recent years much has been done towards improving many of the different activities making up the software development process, but elicitation, analysis and modelling of user requirements still remain as one of the least explored and have the least satisfactory scientific foundations. It is well recognised that requirements specifications are often error-prone and that it is much cheaper to detect and fix these errors early in the software development life cycle than later (Davis et. al. 1997). Clearly, early validation and correction of user requirements may alleviate many of the problems associated with software development, particularly during the maintenance phase (Boehm and Papaccio 1988, Jeffery 1992). Consequently, the RE phase has the greatest economic leverage and hence is now believed to be one of the most crucial steps in the software construction process. Empirical evidence gathered in recent years supports the need for software developers to do a substantially better job of performing requirements analysis and specification than they have been doing (Boehm and Papaccio 1988, Jeffery 1992). For these reasons Requirements Engineering (RE[1]) RE is a promising area for investing research effort.

Requirements gathering is the most communication rich activity of software development. Because social, cognitive and organisational issues are at the heart of many of the problems facing RE and because they cannot be addressed solely by the currently available software

[1] Other well known names for this field include *requirements analysis, systems analysis, requirements gathering,* and *requirements specification*.

engineering methods and techniques, novel approaches and paradigms are being sought from other disciplines. In recognition of the soft nature of this phase, we offer techniques from the fields of knowledge acquisition (KA) and data modelling that are designed to be used directly by end users. We accept that formal methods are typically impractical for widespread usage and acceptance. The proposed approach seeks to offer usable techniques that are also sound. The approach to be explored takes low-level requirements, which can be expressed in crosstable format, and uses Formal Concept Analysis (FCA) (Wille 1982, 1992) to automatically generate a *concept lattice*. The concept lattice provides a graphical abstraction hierarchy of the requirements. We intend to explore the viability and usefulness of the lattice and the proposed RE process model in the identification, negotiation and reconciliation of requirements with RE practitioners and stakeholders. The development of a more rigorous approach to requirements acquisition will offer the greatest leverage for cost saving as it is the first step in the development of computer-based systems.

2. THEORETICAL FOUNDATIONS

The approach offered in this paper is based on the use of the KA and representation technique, Ripple Down Rules, for requirements elicitation and the conceptual modelling technique, Formal Concept Analysis, for requirements reconciliation. To give the reader some familiarity with these methods, the key ideas behind both theories are described, together with some reasons for choosing these techniques.

2.1 Ripple Down Rules

Ripple Down Rules (RDR) is a hybrid case-based reasoning and rule-based approach to KA which stores knowledge as rules in an exception structure. Cases are used to prompt the acquisition of knowledge, to guide the expert in defining rules and to provide automatic validation of knowledge (Compton et. al. 1991). The success of RDR in KA has been demonstrated in the Pathology Expert Interpretative Reporting System (PEIRS) (Edwards et. al. 1993). PEIRS went into routine use in a large Sydney hospital with approximately 200 rules and grew in a four year period (1990-1994) to over 2000 rules. The system was maintained by the domain expert and the 2000 rules represent a development time of approximately 100 hours. Currently, a commercial version of Multiple Classification RDR (MCRDR) (Kang, Compton and Preston 1995) is in use in a dozen pathology laboratories. One system has over 7000 rules that were acquired at a rate of one rule per minute. The commercial version continues to be developed and advances in RDR research are being integrated as appropriate. If, we view requirements as a type of knowledge, techniques used for KA may be relevant. We were drawn to RDR in particular as they support rapid and easy acquisition, validation and maintenance of large KBS. Further, RDR offer incremental system development and KA performed directly by domain experts. This fitted with our need to handle requirements volatility and user participation. Requirements are normally elicited in natural language, during iterative interactions between the problem-owning (users) and the problem-solving (developers) communities. In our explorations we will seek to determine if MCRDR is appropriate for the capture of requirements from natural language. Further discussion of the role of RDR is given in Section 3.1.

2.2 Formal Concept Analysis

The rules acquired using MCRDR, as well as requirements in alternative formats that have been converted into a crosstable, will be used as the input into FCA. FCA draws on ideas from lattice and order theory (Wille 1982). A concept in FCA is comprised of a set of objects and the set of attributes associated with those objects. Knowledge is seen as applying in a restricted context and can be represented as a crosstable and defined as a formal context. A *formal context* is a triple (G,M,I) where G (for Gegenstande in German) is the set of objects which forms the extension of the concept, M (for Merkmale in German) is the set of attributes which forms the intension of the concept and I is a binary relation connecting G and M. We use the notation *gIm* (i.e. ($g,m \in I$) which is read "the object *g* has the attribute *m*". Crosstables are used to capture the relationship between objects and attributes. In a crosstable the rows are objects and the columns are attributes. An X indicates that a particular object has the corresponding attribute. Figure 1 shows a sample crosstable. Using the notion of a *galois* connection, formal concepts are found by determining the set of attributes shared by a set of objects or conversely the set of objects which share a set of attributes. Formally, a *formal concept* of the context (G,M,I) is defined to be a pair (A,B) with $A \subseteq G$, $B \subseteq M$, $A = \{g \in G | gIm \text{ for all } M \in B\}$ and $B = \{m \in M | gIm \text{ for all } M \in A\}$; A and B are called the *extent* and *intent* of the concept (A,B), respectively. The subsumption relation $\geq$ is used to find *sub-superconcept* relations and to draw a complete lattice.

The concept lattice is an abstraction hierarchy of concepts which uncovers higher level concepts from primitive concepts (Richards and Compton 1997). Users, particularly domain experts, are typically competent at describing low-level concepts or providing concrete examples of their domain but have difficulty or are reluctant to explain higher concepts. The automatic generation of concept lattices from requirements provides the automatic organisation of the requirements and the explication of relationships between them. Such structuring allows comparison of requirements based on alternative viewpoints[2].. RDR and FCA are only parts of our approach. We look at the complete process next.

3. THE PROCESS MODEL

The requirements phase of system development typically involves the activities of gathering, modeling, validation, specification and management. The process model offered is concerned with the first 3 activities. Stage 1 of our process model corresponds to the gathering activity. Since we are concerned with acquiring requirements from multiple stakeholders modeling includes comparison of models, identification and reconciliation of conflicts. Stages 2-5 support modeling and validation from multiple stakeholders. Briefly, the process model is made up of the following stages:

1. Requirements acquisition and conversion- Capture each individual viewpoint. This is the requirements gathering activity which may result in requirements in formats such as use case descriptions or interview transcripts or using the MCRDR KA technique and cases from the domain to acquire requirements into a KBS.
2. Concept generation - The crosstable is interpreted as a formal context and FCA is used to derive the concepts associated with each viewpoint.
3. Concept comparison and conflict detection - Pairwise comparison between N views is made to detect conflicts using a four state model of comparison.

[2] The use of the word viewpoint in this paper is not to be confused with the approach introduced by (Mullery 1979) and further extended by (Easterbrook and Nuseibeh 1996, Finkelstein et. al. 1994). We find the term convenient for describing the different stakeholders that are involved in the RE process and subscribe to the definition of a viewpoint as a style, an area of concern, a specification, a work plan and a work record (Finkelstein et. al 1989).

4. Negotiation - Here conflict is handled. There are a range of strategies that can be employed based on the types of conflict detected in Stage Four. The resolution operators are employed to update the individual and/or combined views.
5. Evaluation – Determine the degree of conflict to see if viewpoints are moving towards convergence and whether another cycle is needed.

The five steps are iterative. The cycle continues until the parties are satisfied with the combined specification. We will now consider in more detail each of the five stages.

3.1 Stage One: Requirements Acquisition and Conversion

A knowledge-based approach to requirements elicitation has been taken previously (e.g. Easterbrook 1989, Maiden and Rugg 1994, Ruebenstein and Waters 1989). A limitation of most earlier approaches is that they only supported a single viewpoint (Easterbrook 1991). An exception is Nii (1986) who treated the knowledge from each expert as a separate reasoning system. In the approach offered in this paper the goal is to create a shared viewpoint.

This stage is currently our least explored with many issues still outstanding. The final design of this stage will depend on various case studies and experiments that we will perform, see the Research Plan Section. We want to be able to use requirements in a variety of forms that are in current use, such as data flow diagrams, use case descriptions or decision tables. We do not plan to specify any particular type of knowledge representation or the method of acquisition. However, since phase three and five rely on using FCA, we are only able to make use of knowledge that can be represented as a crosstable. A crosstable is equivalent to a decision table in binary format. Decision tables are a commonly used representation of knowledge and not seen to be too restrictive.

Requirements conversion is the process of ensuring that all viewpoints are in comparable formats. All formats are mapped to the cross table format (to enable the use of FCA in subsequent phases) so that the largest number of mapping schemes we require for N viewpoints (i.e. each viewpoint is captured in a different representation) is 2N not N^2. In the best case, all viewpoints may be captured using the same knowledge representation requiring one mapping scheme. If the requirements are captured directly in cross table format then this phase is not necessary. If the requirements are captured using MCRDR conversion is straightforward.

While the approach does not enforce a particular knowledge representation or acquisition technique, MCRDR is proposed due to the issue of requirements change. We will explore the utility of MCRDR as it offers a number of attractive features for KA such as rapid acquisition, easy maintenance, automatic validation all of which can be performed by the user without the mediation of a knowledge engineer. An additional benefit of RDR is that the rule pathways map directly into a decision table and do not need intermediate conclusions to be mapped to primitive conditions (Richards and Compton 1997), as many rule bases require. To gain the benefits of the RDR approach it is important that a suitable source of cases be available for RE. Use cases (Rumbaugh 1994) have gained popularity as a means of eliciting and specifying requirements. We propose use cases as a possible source of cases. Cases used by RDR are often historical cases such as pathology results for a patient. Cases in a case-based reasoning sense are more general and represent stereotypical cases for the domain. Use cases are often developed based on a real situation but then generalised for reusability. Use cases tend to have more detail than would be desirable or feasible in the framework offered. To overcome this the natural language description in the use case can be converted into a more succinct representation such as shown in the diagrams in this paper. It is undesirable that simplification would result in loss of content. The transition from use cases to cases that are simplified for KA using RDR is an important step that will be investigated as part of this project.

3.2 Stage Three: Concept Generation

In the approach, concepts are generated using FCA. As described in Section 2.2, FCA begins with the definition of a formal context. A small example is introduced here and used to describe stages 2-5. As Stage 1 is still under development, we have taken the requirements for the Library Domain (Finkelstein et. al. 1994) which we will treat at the output of Stage 1. A subset of the requirements are represented in Figure 1 which shows the formal context $\mathbb{K}$ for the Clerk viewpoint n. $\mathbb{K}$ has the set of objects G = {1-check-in, 2-check-in, 3-check-out, 4-checkout} and the set of attributes M = {source=borrower, input=book, input=card, action=check-in, action=check-out, output=book, output=card, output=database_update, destination=borrower, destination =catalogue, destination library}. The crosses show which objects have which attributes, thus I = {(1-check-in, source borrower), (1-check-in, input book),…,(4-check-out, destination catalogue)}. This crosstable is used to find formal concepts as described in section 2.2. The screen dumps shown in Figures 2 and 3 are from our implementation called MCRDR/FCA, which is an enhancement of the current MCRDR for Windows system.

	Source-borrower	Input book	input card	action check-in	action check-out	Output book	output card	Output Database update	dest. Borrower	Dest catalogue	Dest librarian
1-%BCKIN	X	X		X				X		X	
2-%BCKIN	X	X		X		X					X
3-%BCOUT	X		X		X	X	X		X		
4-%BCOUT	X		X		X			X		X	

Figure 1: Context of "Library from Clerk Viewpoint"

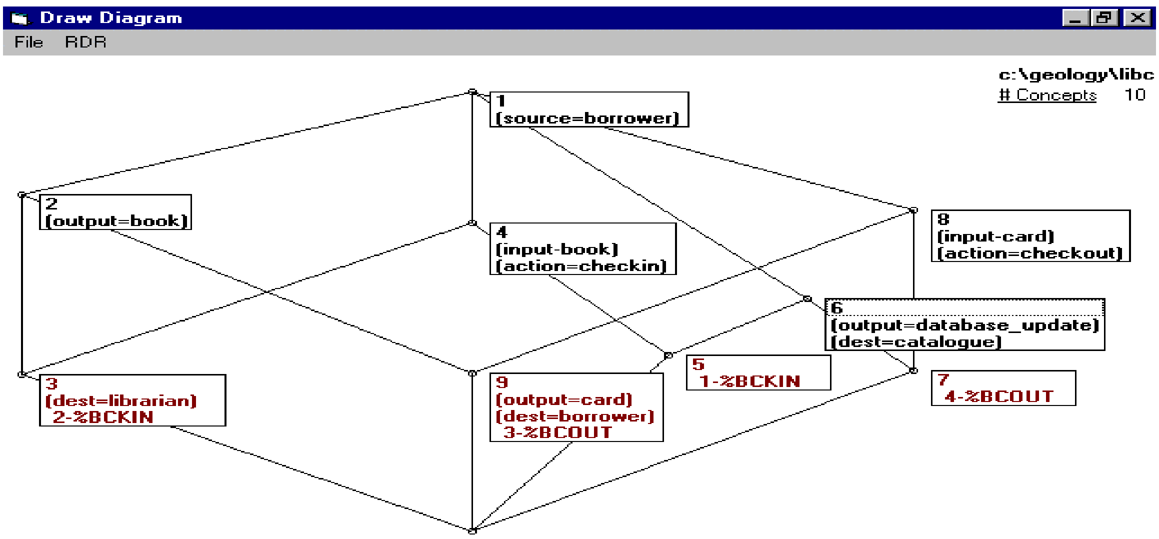


Figure 2: The Line Diagram for the Formal Context "Library from Clerk Viewpoint" in Figure 1. Each circle represents a concept where the sets of attributes and objects for that concept are reached by traversing ascending and descending paths,, respectively.

In Figure 2 the concepts are shown as small circles and the sub/superconcept relations as lines. Each concept has various attributes and objects associated with it. The labelling has been reduced for clarity. All attributes of a concept β are reached by ascending paths from β and all objects are reached by descending paths from the concept β. For example, concept number 6 in Figure 2 contains the set of attributes {source=borrower, output=database_update, dest=catalogue} for the objects {1-%BCKIN, 4-%BCOUT}. The top concept includes all objects and the set of attributes they share. The bottom concept represent the set of objects that share all attributes. In Figure 2, there is no such object. A formal context and line diagram may be developed for each of the viewpoints in the Library World. A line diagram including all viewpoints can be shown to reveal differences between viewpoints.

3.3 Stage Four: Concept Comparison and Conflict Detection

The concepts generated for each viewpoint in Stage Three are compared in this step. We have chosen to use the four quadrant model of comparison between experts developed by Shaw and Gaines (1988) as it is compatible with the FCA definition of a concept. The model of comparison classifies two conceptual models as being in one of four states:

1. *Consensus* is the situation where experts describe the same concepts using the same terminology.
2. *Correspondence* occurs where experts describe the same concepts but use different terminology.
3. *Conflict* is where different concepts are being described but the same terms are used.
4. *Contrast* is where there is no similarity between concepts or the terminology used.

States 2, 3 and 4 can all be viewed as conflict states in the more general sense of the word used in this paper. Shaw and Gaines' model, however, does offer greater precision in describing the nature of the conflict which is important in deciding how it can be handled. We will use these four states to describe how the models have been compared. Viewing concepts as being in different states is similar to the work on overlaps (Spanoudakis, Finkelstein and Til 1999) where the relation between interpretations of components of two specifications is determined and used in resolving inconsistencies. However, the nature of the overlap is considered in terms of *total*, *partial*, *inclusive* or *non-overlapping*. This does not assist in determining the cause of the overlap which is offered in Gaines and Shaw's four-state model.

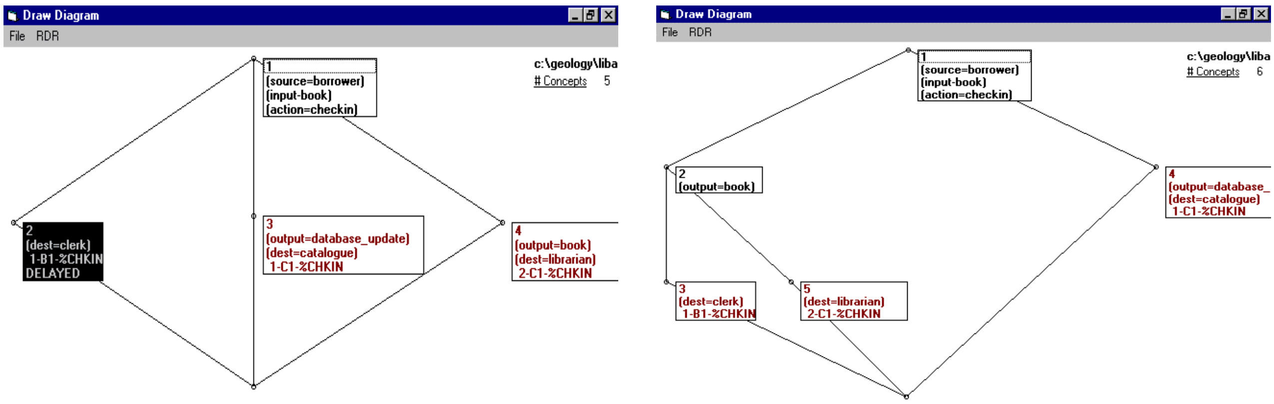
3.4 Stage Five: Conflict Negotiation

Before we can determine how to fix a detected inconsistency we need to provide a conflict resolution strategy. There are a number of resolution methods, which include negotiation, arbitration, coercion and education (Strauss 1978). Negotiation is the most appropriate within the assumed context of parties of equal status and ability. An approach to negotiation can be a general, genial chat but since automation is an important goal of requirements engineering research, it is desirable to offer as much automation as possible. Having decided on the nature of the conflict it can be handled in a number of different ways. Thomas (1976) offers five main approaches which are: competition, collaboration, avoidance, accommodation and sharing. Easterbrook (1991) has described these orientations as domination, integration, neglect, appeasement and compromise, respectively. We adopt the strategies offered by Easterbrook and Nuseibeh (1996):

- resolving (remove inconsistency),
- ignoring (take no action),
- circumventing (don't include),
- delaying (put on hold) and
- ameliorating (reduce the degree of inconsistency).

The resolution operators we employ to handle these strategies include the use of a synonym table for corresponding terms; tags to identify if a concept is ignored, delayed or circumvented; and the addition or deletion of attributes, objects or cases. These strategies are discussed in more detail in (Richards and Menzies 1998) and demonstrated using on a KE problem.

Figure 3(a) shows the concept lattice generated using MCRDR/FCA when all concepts related to the check-in action are compared from the Borrower, Library and Clerk viewpoints. There are no correspondence conflicts since the terminology has been defined and restricted according to the action tables provided in (Finkelstein et. al 1994). The object is identified by the rule number in the original knowledge base, an identifier for the knowledge base from which it originated (C1 is the Clerk viewpoint, B1 is the Borrower viewpoint) and the checkout conclusion %CKOUT. The library viewpoint does not include a requirement for the check-in action. The top node represents the concept in consensus. The Borrower and Clerk viewpoints share concept 1 which includes the attributes source=borrower, input=book and action=checkin. Concepts 2, 3 and 4 are in conflict and need to be reviewed to determine the appropriate course of action. The Clerk has two views of the check-in action one involving updating the catalogue and the other involves returning the book to the librarian. These two scenarios are acceptable and no further action is required. On the other hand the borrower sees the check in action as involving the clerk but it does not indicate what is returned to the clerk. In Figure 3(a) Concept number 2 has been tagged as delayed as the conflict needs to be reconciled at a later date after discussion with the borrower. After discussion with the borrower, it is realized they forgot to include the attribute output=book in their requirement. This attribute is added to the check-in object. The result is shown in Figure 3(b). The borrower and clerk now also share the view that the output of the check-in action is a book, although the destination is different due to the different ways these people interact with the system.



(a) *(b)*

Figure 3: The concept lattice on the left shows the borrower and clerk viewpoints with reconciliation of concept 2 delayed until the output of the action can be determined from the Borrower's viewpoint. The concept lattice on the right shows the lattice after that information has been determined. The lattice shows that the borrower and clerk now agree that the output of the check-in action is a book although the destination is different.

3.5 Stage Six: Evaluation

We are able to determine the degree of conflict by finding for each shared object the number of attributes shared divided by the total number of attributes used to describe this object. Obviously other measures are possible and the degree measured is relative. This is adequate for determining whether the degree of conflict is going up or down. We use this score to test if we are moving towards a shared view and if further iterations of the process are necessary. A related issue is evaluation of the completeness of the set of requirements. The answer to this is that requirements are never complete although it is hoped that by combining requirements from a number of viewpoints and by repeating the six phase process until a suitable state of (partial) consensus is reached that there are minimal gaps in the final requirements model.

4. FUTURE WORK

The process model presented, particularly stage 1, requires further investigation before it could be offered as an approach usable by practitioners. This section describes the multimethodological approach we plan to use to perform these investigations, which includes a combination of experimental, survey and qualitative methods. The study had two main objectives.

Objective 1 seeks to determine whether a knowledge engineering approach to modelling and reconciling knowledge from different domain experts can be applied to modelling and reconciling requirements from different stakeholders or viewpoints. A number of earlier researchers have assumed this to be true (e.g. Maiden and Rugg 1994) but we plan to determine:

1. what features differentiate knowledge engineering from requirements engineering
2. how do these differences affect the applicability of KE techniques to RE
3. what specific adaptations are necessary to the proposed KE technique to make it suitable for RE

The findings of the first 2 questions will be of general interest to the field of RE and will provide a set of general guidelines to assist RE researchers and practitioners in determining if a KE approach will be appropriate for the RE problem at hand. The third question will yield more specific results as it concerns the investigation of the subproblem:

3.1 Is the knowledge acquisition and representation technique known as Multiple Classification Ripple Down Rules (MCRDR) appropriate for the acquisition, maintenance and validation of requirements?

If MCRDR is found to be unsuitable for the capture of requirements another technique will be applied or adapted using some form of constrained natural language which can be formatted as a crosstable for manipulation using Formal Concept Analysis (FCA). Richards (1999) has shown that the use of FCA for modelling knowledge is not restricted to knowledge in MCRDR format but is appropriate for modelling knowledge that can be mapped into a crosstable such as knowledge in decision table, decision tree or propositional form. We believe that requirements can be put into crosstable format since a number of researchers have used requirements in the form of propositions (Easterbrook and Nuseibeh 1996, Finkelstein et. al. 1994, Zowghi, Ghose and Offen 1997) which readily convert into a decision table (Colomb 1993) which in turn readily converts to a crosstable.

Objective 2 seeks to explore whether concept lattices are a feasible and useful way of visualising and reasoning about requirements by a group of stakeholders. We believe, the value of such a visualisation is the identification of commonalities and differences between viewpoints, discussion and reconciliation of conflicts and the ability to evaluate whether the requirements gathering team are converging towards a state of consensus. The resulting shared viewpoint is to be used as the specification from which the system is developed.

4.1 Project Plan

To explore objectives 1 and 2 we have devised the following project plan. The first objective is to be investigated in the early stages of this project and reviewed throughout the course of the project. The second objective is essentially an evaluation of the usefulness of the approach and will be performed in the final stages of the project. We have broken the investigations into four main phases.

Phase 1- Upgrade the existing MCRDR/FCA tool to improve exploration of the models, improve the efficiency of concept generation and fully implement some of our strategies that are only partially implemented to-date.

Phase 2. This phase focuses on Stage 1 of the process model given in Section 3. We will explore the automated conversion of controlled natural language into a crosstable. We plan to develop some clear guidelines which will make this conversion more straightforward and less biased or subjective. An integral part of this phase is the determination of the suitability of MCRDR in the acquisition phase of the requirements (subproblem 3.1). More specifically, we plan to develop some guidelines for getting initial requirements into table format and automate the process where possible. To this end we will explore tools, such as ATTEMPTO (Fuchs and Schwitter 1996), which take in constrained natural language and output propositions. To assist in developing the guidelines and determining what types of requirements can automatically be acquired, we will perform numerous case studies using our upgraded MCRDR/FCA tool from phase 1. Many case studies have been published in systems analysis and design textbooks and in the research literature (e.g. the meeting schedular system in van Lamsweerde, Darimont and Massonet 1993). We plan to use the these case studies which provide requirements in a variety of formats such as tables and use cases as sources of input into our process and for the purpose of comparison with our results.

The case studies will also give us insight into how requirements can be acquired and represented in a computer processable format. These requirements will be used by FCA to develop concept lattices. Data, in the form of concept lattices, observations, and sets of requirements for individual and shared viewpoints, will be collected at each stage and iteration of the RE process. Experiments will be performed that allow us to evaluate the impact of variables, such as the source of requirements, the format of requirements, the use of different views (these are subsets of the total set of requirements), and so on, on the efficacy of the RE process. By altering one variable at a time we plan to test the impact of that variable.

Phase 3- Based on the outcomes of the case studies in Phase 2, the tool and RE process model will be extended and formalised. If MCRDR is found to be appropriate for capturing requirements, we will need to extend and formalise the use of MCRDR in the approach. If MCRDR is inappropriate, we will concentrate on acquisition of requirements directly from more conventional RE sources such as data flow diagrams, use case descriptions and class diagrams. This phase will require revisiting some of the case studies or rerunning and possibly modifying some of the experiments performed in Phase 2. Our findings will also involve enhancements to the MCRDR/FCA tool and making it sufficiently robust and user-friendly for use in the evaluations to be performed in the next stage.

Phase 4 - This phase will focus on evaluation of the tool and process model. The evaluation in Phase 4 is to extend beyond feasibility, which can be measured in phases 2 and 3 using the case studies and experiments. The evaluation in Phase 4 is focused on the usefulness of the approach. We want to answer questions such as:

- ✓ Can we get people to use the approach?
- ✓ Can the approach not only identify commonalities and conflicts between stakeholders but also assist in resolution of these conflicts so that a representative set of requirements can be developed?

Evaluation will explore the usefulness of the approach for group decision making and will thus involve comparison to other similar groupware tools that may be available at the time.

Evaluation with real subjects is always problematic, the biggest problem being access to subjects. We currently have access to a number of sources of subjects including undergraduate students in a second year Requirements, Analysis and Systems Design unit (possibly for testing input of requirements in different formats) and third year Software Engineering unit, postgraduate students in a Postgraduate Professional Development Program and participants in the Software Requirements Engineering (SRE) Mailing List[3] which is comprised of academics, students, researchers and RE practitioners. Different tests (survey, case study or experiment) will need to be designed depending on the method of access; time, experience and knowledge of the subjects; analysis techniques and the goals of the particular test. An example of a test will be to give two test groups, Groups A and B, two case studies, Case Study 1 and 2. Each group will have to produce a set of requirements for both case studies which represent the groups combined viewpoint. For Case Study 1, Group A will use our tool and Group B will use group discussion to identify and reconcile conflicts. For Case Study 2, Group B will use our tool and Group B will use group discussion. Prior to this experiment we will perform experiments to determine how quickly people can learn to use the tool and make modifications based on the problems encountered. We are not able to fully specify the experiments or case studies to be performed in this phase since these will depend on the changes we make to the tool and what we learn from phases 2 and 3. Since we will be using human subjects we also need to have our experiments approved by the ethics committee. At this stage we acknowledge that test development, evaluation and application and result evaluation will be essential to determine if our second objective can be achieved.

5 SUMMARY

This paper has introduced a process model for requirements elicitation and reconciliation from multiple stakeholders. As future work we intend to extend the process and evaluate it with practitioners. This work is novel in two respects:

1. We propose to tackle the more soft and socially-situated problem of requirements acquisition rather than assume we have already acquired these in a suitable format and
2. In contrast to traditional approaches to modelling which spend considerable time building the model we perform minimal *a priori* analysis and use FCA to automatically generate and formalise a model of our stakeholder's viewpoints.

REFERENCES

Boehm, B.W. and Papaccio, P.N. (1988) Understanding and Controlling Software Costs *IEEE Transactions of Software Engineering* 14(10):1462-1477.

Colomb, Robert.M. (1993) Decision Tables, Decision Trees and Cases: Propositional Knowledge-Based Systems *Technical Report No. 266* Key Centre for Software Technology, Department of Compter Science, The University of Queensland, Australia.

Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Menzies, T., Preston, P., Srinivasan, A. and Sammut, C. (1991) Ripple Down Rules: Possibilities and Limitations *6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems Workshop*, Banff, 6.1 - 6.18.

[3] SRE mailing list is an electronic discussion bulletin board that is managed and moderated by Didar Zowghi at Macquarie University. It has close to 700 subscribers world wide.

Davis, A.M., Jordan, K. and Nakajima, T. (1997) Elements Underlying the Specification of Requirements. *Annals of Software Engineering, Spec. Issue on Software Requirements Engineering*, 3:63-100.

Easterbrook, S. M. (1989) Distributed Knowledge Acquisition as a Model for Requirements Elicitation, In *Proceedings of the 3rd European Workshop on Knowledge Acquisition for Knowledge Based Systems*, Paris, July 1989.

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 SW7 2BZ.

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

Edwards, G., Compton, P., Malor, R, Srinivasan, A. and Lazarus, L. (1993) PEIRS: a Pathologist Maintained Expert System for the Interpretation of Chemical Pathology Reports *Pathology 25*: 27-34.

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.

Finkelstein, A., Gabbay, D., Hunter, A, Kramer, J. and Nuseibeh, B. (1994) Inconsistency Handling in Multi-Perspective Specifications *IEEE Transactions on Software Engineering* 20(8):569-578.

Fuchs, N.E. and Schwitter, R. (1996) Attempto Controlled English (ACE) In *Proceedings of the First International Workshop on Controlled Language Applications (CLAW'96)* Leuren, Belgium.

Jeffery, D.R. (1992) Software Process Quality: Requirements for Improvement from Industry In *Rome Air Development Centre Software Quality Workshop*, New York.

Kang, B., Compton, P. and Preston, P. (1995) Multiple Classification Ripple Down Rules: Evaluation and Possibilities *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995, Vol 1: 17.1-17.20.

van Lamsweerde, A, Darimont, R. and Massonet, Ph. The Meeting Scheduler System: Preliminary Definition *Internal Report,* University of Louvain, Belgium.Maiden, N. A. M. and Rugg, G. (1994) Knowledge Acquisition Techniques for Requirements Engineering *In Proceedings of the Workshop on Requirements Elicitation for Systems Specification*, Keele, UK, July 1994.

Maiden, N. A. M. and Rugg, G. (1994) Knowledge Acquisition Techniques for Requirements Engineering *In Proceedings of the Workshop on Requirements Elicitation for Systems Specification*, Keele, UK, July 1994.

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.

Nii, H.P. (1986) Blackboard System (Part 2): Blackboard Application Systems, Blackboard Systems from Knowledge Engineering Perspective *AI Magazine* Summer 1986.

Reubinstein, H.B. and Waters, R.C. (1989) The Requirements Apprentice: An Initial Scenario *Proceedings of the fifth IEEE International Workshop on Software Specification and Design* Pittsburgh, Penn.

Richards, D. (1999) Visualising Knowledge-Based Systems *Proceedings of the Third Workshop on Software Visualisation*, Dec 3-4, 1999, University of Technology, Sydney, 1-8.

Richards, D. and Compton, P. (1997) Uncovering the Conceptual Models in Ripple Down Rules In Dickson Lukose, Harry Delugach, Marry Keeler, Leroy Searle, and John F. Sowa, (eds) (1997), *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS'97),* August 3 - 8, University of

Washington, Seattle, USA, Lecture Notes in Artificial Intelligence, Number 1257, Springer Verlag, Berlin, 198-212.

Richards, D. and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirements Engineering Task In Gaines, B. and Musen, M. (eds) *11th Workshop on Knowledge Acquisition, Modeling and Management (KAW'98),* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:SIS-6.

Richards, D. and Zowghi, D. (1999) Maintaining and Comparing Requirements *Proceedings of the Fourth Australian Conference on Requirements Engineering (ACRE'99)* 29-30 September, 1999, Macquarie University, Sydney, 115-130.

Rumbaugh, James (1994) Getting Started: Using Use Cases to Capture Requirements *JOOP* Sept. 1994, 8-23.

Shaw, M.L.G. and Gaines, B.R., (1988) A Methodology for Recognising Consensus, Correspondence, Conflict and Contrast in a Knowledge Acquisition System in *Proceedings of the 3rd Knowledge Acquisition for Knowledge-Based Systems Workshop,* November 1988, Banff, Canada.

Spanoudakis, G., Finkelstein, A. and Till, D. (1999) Overlaps in Requirements Specifications Automated Software Engineering Journal (to appear) available from ftp.cs.city.ac.uk/users/gespan/ase99.ps.gz

Strauss, A. (1978) *Negotiation: Varieties, Contexts, Processes and Social Order* Jossey-Bass Publishers, San Francisco, CA.

Thomas, K. (1976) Conflict and Conflict Management In Duneette (ed) *Handbook of Industrial and Organisational Psychology* Rand McNally College Publishing Co.

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 Systems *Computers Math. Applic. (23)* 6-9: 493-515.

Zowghi, D., Ghose, A. and Offen, R. (1997) Computer Assisted requirements Evolution Tool: Formal Foundation and Architecture, *Proceedings of the Second Australian Workshop on Requirements Engineering* (AWRE97), Macquarie University, Sydney, Australia, October 1997, 37-49.

ACKNOWLEDGEMENTS

Thanks to Tim Menzies who suggested that the MCRDR/FCA approach could be applied to the problem of requirements engineering and who participated in a number of useful discussions to this end. Thanks to Didar Zowghi for her contribution to this paper particularly in the development of the research plan.

COPYRIGHT