REQUIREMENTS RECONCILIATION USING A COMPUTER SUPPORTED COLLABORATIVE APPROACH

Debbie Richards[*+], Kathrin Boettger, Anne Britt Fure and Oscar Aguilera
Department of Computing, Division of Information and Communications Sciences,
Macquarie University, Sydney, Australia

Getting the requirements right is essential for good design. A currently popular technique for capturing requirements is the specification of use cases. However getting a conflict free, complete and representative set of use case descriptions can be difficult. In this paper we look at how the RECOCASE approach provides collaborative requirements RECOnciliation through the use of a computer aided software engineering (CASE) tool. In the approach, use case descriptions are entered asynchronously in natural language from individual stakeholders. The tool automatically produces a visual representation of the requirements to assist the group to identify and resolve conflicts and produce a representative requirements specification.

Keywords: *Group decision support, requirements engineering, formal concept analysis.*

1. Introduction

A good design relies on adequate requirements specification. If the requirements are inconsistent, incomplete or invalid then the design will be inappropriate or even useless. One form of requirements specification is use cases, which provide the system functionality from the users' point of view. Capturing the use case model is typically a group activity and should result in a shared description of the users' requirements. However, it is common for a few individuals to dominate the group and produce use case descriptions that do not represent the requirements of the whole group. The approach we have developed, known as *RECOCASE* as we use a computer aided software engineering (*CASE)* tool to assist requirements *RECO*nciliation, is designed for individual and group use. Our approach assists the group communication process and also results in a number of artifacts. These are used to generate new concepts and refine those already identified.

The process begins with the group brainstorming the main chunks of functionality in the form of a use case diagram using the Unified Modelling Language (UML). Ivar Jacobson was the first who applied the concept of use cases to software development as part of his object-oriented software engineering method (OOSE) (Jacobson, 1992). A use case represents a complete course of events in a

[*] Corresponding Author: Debbie Richards, Computing Department, Macquarie University, North Ryde, NSW, 2109, Australia, Ph: +61 02 9850 9567 Fax: +61 02 9850 9551, richards@ics.mq.edu.au
[+] This project is funded by grants from the Australian Research Council and Macquarie University.

system from the user's perspective. Users can often describe their needs in terms of use cases because they describe what is actually done, compared to describing what the system should support. 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 a library system include borrowing items, returning items, paying fines, maintaining members and maintaining items to be borrowed. With the high-level use case diagram before them, the group identify viewpoints and a representative, probably from within the group, for each viewpoint.

After the initial group meeting, the group leader creates a new project and enters the names of the identified use cases into the RECOCASE-tool. Later, the representative for each viewpoint can log into the system, open the project and enter descriptions for one or more use cases for their viewpoint. Use case and scenario descriptions provide a textual description of such things as use case name, actors, preconditions, postconditions, trigger, main flow and alternative flows. Our work is primarily concerned with the main flow which is the step-by-step sequence of actions from trigger to achievement of postconditions. These descriptions may be in natural language but better results are achieved with our tool when a controlled language is used. We have developed guidelines and tool support to assist the user in complying with the controlled language (Boettger *et al.* 2001). LinkGrammar is used by an answer extraction system (ExtrAns) (Molla *et al.* 2000) to translate the sentences of the use case description into flat logical forms (FLFs). FLFs are used to create crosstables. Formal Concept Analysis (FCA) (Wille 1982, 1992) is used to develop a concept lattice which can be displayed as a line diagram to graphically represent the viewpoints. This process may sound complex, but it is shielded from the user, who simply enters the steps involved in a particular scenario or use case and then views a line diagram of the concept lattice. Our group process then offers a number of resolution strategies and operators to assist development of a shared conceptual model of the requirements.

In the next section we begin with an overview of our viewpoint development process followed by a detailed description of each phase. In section 3 we discuss our group process and the nature of collaboration. In section 4 we describe evaluations we have conducted of the group process. Our conclusions are given in section 5.

2. The Viewpoint Development Process

A number of viewpoint development approaches have been proposed (e.g. Darke and Shanks 1997, Easterbrook and Nuseibeh 1996, Finkelstein *et al.* 1989 and Mullery 1979). Our approach is novel in that, in addition to capturing multiple viewpoints, it allows entry of requirements in natural language, automatic conversion into a formal representation, visualization of the requirements and resolution strategies to derive a shared and comprehensive set of requirements.

The RECOCASE development process includes six iterative steps. These steps are:

1) Requirements acquisition (describe use cases)
2) Requirements translation (convert to crosstables)
3) Concept generation (generate concepts using FCA)
4) Concept comparison and conflict detection (by viewing concept lattices)
5) Evaluation (calculating distance between viewpoints based on distance between nodes on the lattice)
6) Negotiation (applying our resolution strategies)

For the purpose of this paper we have simplified our process to the Viewpoint Development Process proposed by Darke and Shanks (1998), which includes the phases of viewpoint identification (our step

1), viewpoint representation (our steps 2 and 3), intra-viewpoint analysis and inter-viewpoint analysis (our steps 4 and 5) and viewpoint integration (our step 6). We consider each of these phases in the following subsections. Figure 1 illustrates the process. Double-bordered text boxes in figure 1 indicate manual activities. Single-bordered text boxes in Figure 1 indicate automatic processes. For a complete picture more automatic processes should be shown, but since we are concentrating on the group process we have only included those needed to go from natural language to the diagram.

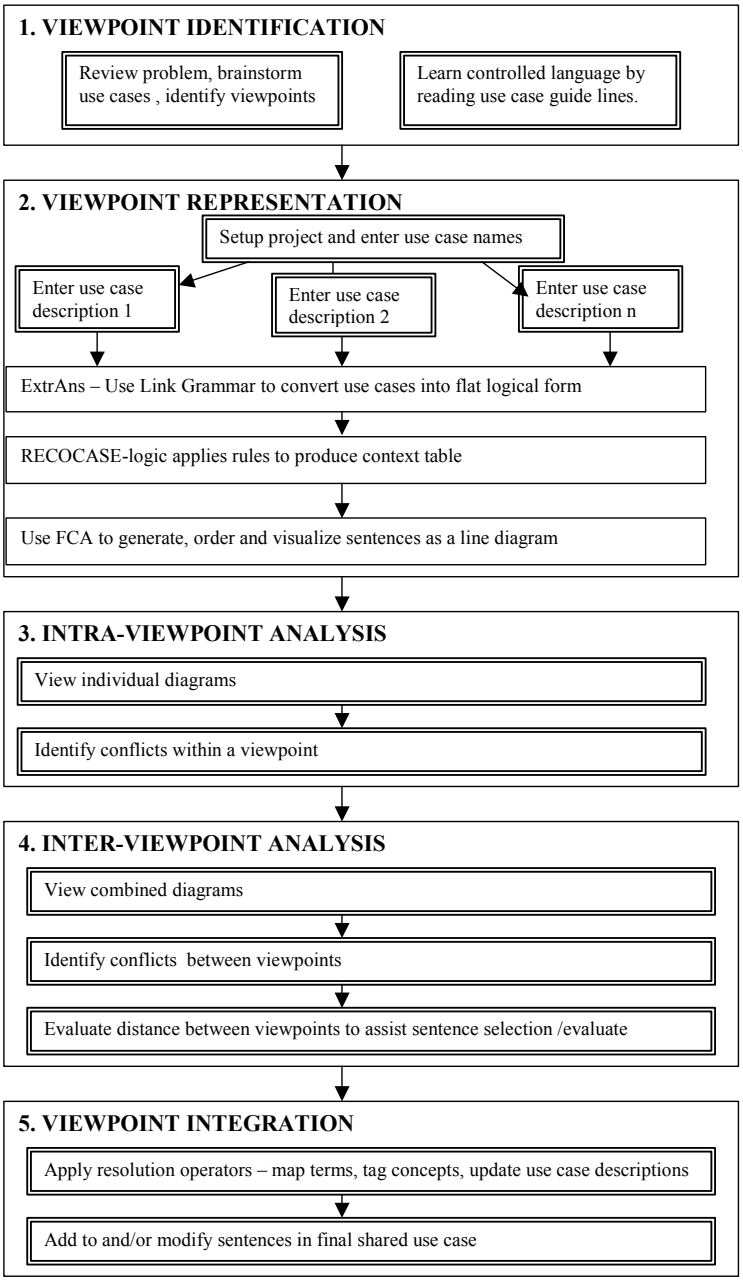

Figure 1. RECOCASE process model.

The example used in the rest of this paper concerns the Automated Video System (AVS). AVS is being designed to allow customers to self check-out videos, search for particular video titles, request new videos to be stocked by each store, and to make reservations for videos currently unavailable. Each customer has his/her own customer card, which is issued by the members of staff. Each card has a credit value and the customers can give money to staff members who then increase the card value. The customers use this card when they rent videos, and the price for renting a movie is subtracted from the credit value on the card. If the credit value is too low or the customer has any overdue films, the customer will not be allowed to rent a new movie. Any overdue fines must be paid to a staff member. By allowing customers to perform these tasks, staff will have more time to perform other duties such as issuing customer cards, registering returned videos, placing reserved videos on a special shelf, viewing the customer's video request, order new videos to the store and collecting fines from customers who did not return their videos on time.

2.1 Viewpoint Identification

Viewpoint identification starts with the creation of a viewpoint model, which is an extension of Jacobson's (1992) use case model. The development team will be comprised of a number of different types of people including users from different departments in the organization (eg. personnel and marketing), from different levels in the organization (eg. managers and operational staff) and people with different tasks and responsibilities (eg. programmers, usability engineers). A number of viewpoints can be identified for each use case representing one or more of these group members. Others outside of the organization such as suppliers and customers may also need a viewpoint agent to be assigned to represent them. All methods for the identification of Jacobson's use case model can be applied to the identification of a viewpoint model. We recommend one to five viewpoints for each use case as beyond that number would require the effort of too many individuals and reconciliation would be overly complex. If a viewpoint equates to an actor, then it would be rare to have more than five actors involved in one use case. The representative or 'viewpoint agent' is responsible for describing that viewpoint.

2.2 Viewpoint Representation

RECOCASE captures viewpoints of functional requirements in the form of use cases and scenarios. Our tool provides three alternative ways of entering and structuring use case descriptions. One possible way to describe the flow of actions is to use unstructured text (-style 1-). Cox (2001) suggests to write a use case as a list of discrete actions in the form <action#> <action description> and to use a separate line for each action (-style 2-). Figure 2 uses style 2 and shows what a use case description may look like. Wirfs-Brock (1993) proposed a structured form, which is divided into a user-action-model and a system-response-model to describe the interaction between a user and a system through a graphical user interface (-style 3-). See Table 1 for an example of style 3. The user-action model represents what the user does and the system-response model shows the system's responses to the user's actions. For the approach described in this paper the model of user-system interaction by Wirfs-Brock is extended to a model of actors-system interaction to be able to describe the interaction between the system and more than one actor. In our approach scenarios may be specified in any of the 3 styles. Use case descriptions may only be written in style 2 or 3, as shown in the tabs in Figure 2, since the free format of style 1 is too unstructured for our tool to enforce the guidelines and controlled language.

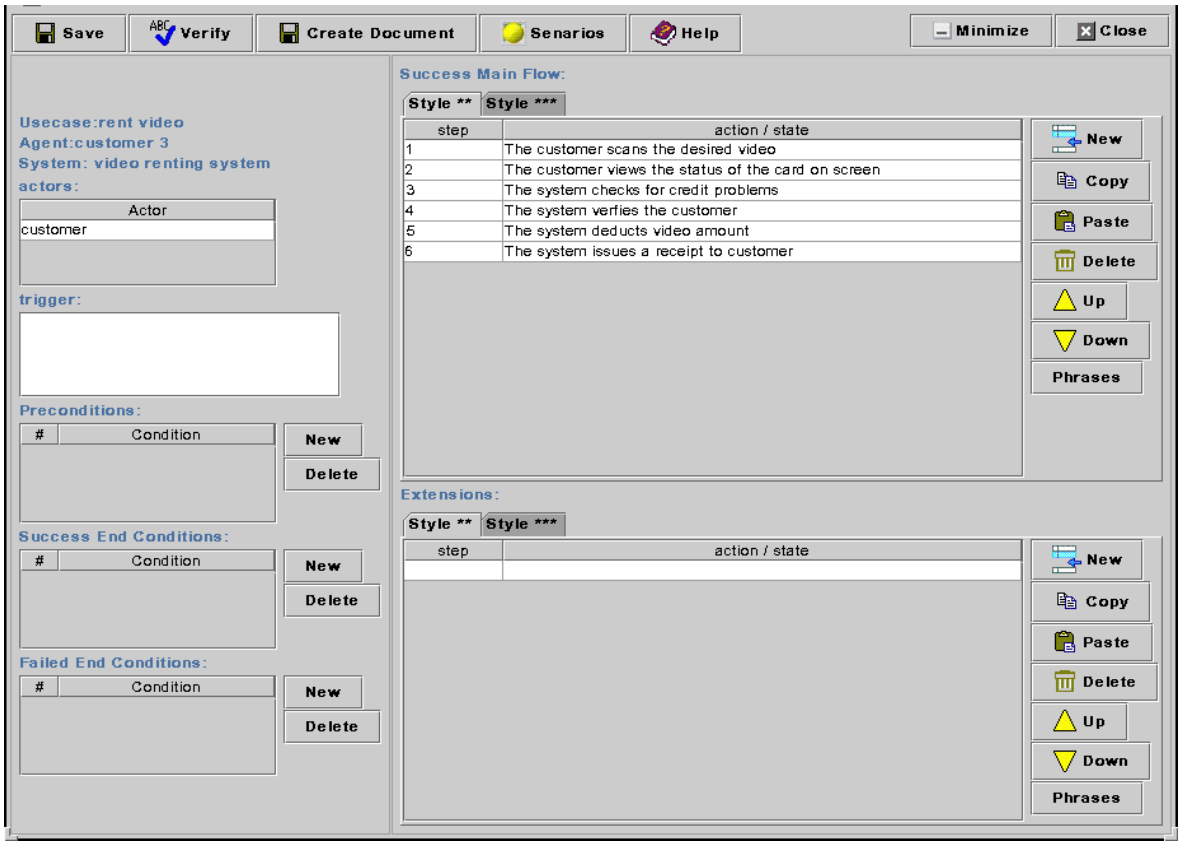

Figure 2. Graphical user interface for composition of a use case viewpoint.

Table 1. Model of user-system interaction using Wirfs-Brock (1993) approach.

User Action	System Response
1. Swipe card	2. Read card data
	3. Display credit value
4. Scan video barcode	5. Show amount payable
	6. Display due date
	7. Request rental confirmation
8. Accept transaction	9. Debit customer account
	10. Print receipt

2.3 Intra-Viewpoint Analysis

After a use case has been described for each viewpoint it should be checked to see if the viewpoint agent followed the guidelines (Boettger *et al.* 2001). This is important as we need to translate the natural/controlled language sentences into tabular format. In RECOCASE-tool we provide manual and automatic checking. The user can press the "verify" command button, as shown in the top left corner of Figure 2, to have their sentences checked before they request conversion of the sentences into a

crosstable or line diagram. To conform to our use case description guidelines, the tool looks for unknown words, modal verbs, personal and possessive pronouns and replaces them or asks the user to provide a substitute. Alternatively, the user can select "save" without first verifying the sentences. Step by step verification by the viewpoint agent is preferred as it avoids errors that may be harder for the user to identify and correct later and the process also assists the viewpoint agent in learning the controlled language. We are currently working on adding further verification features to the tool at the word and sentence level. These features directly relate to the guidelines and rules of the controlled language. At the current stage, the tool assists in finding words which are unknown for ExtrAns or which are treated as keywords. The tool also provides an output referring to the structure of each sentence (noun phrases, verb phrases, phrase sentences), which we use in breaking up the sentence into word and phrases displayed in the line diagram. The goal of this phase is to produce internally consistent viewpoints.

2.4 Inter-Viewpoint Analysis

This phase looks beyond an individual viewpoint and looks at consistency between viewpoints. The project leader, or another appointed person with experience in working with the tool and the reconciliation process, would explore the viewpoints by combining selected viewpoints. To make the task manageable and the line diagram readable, this person would select certain terms and/or sentences on which to focus. The leader would prepare a number of line diagrams around which discussions can be held. This is a very similar approach to the use of UML in object-oriented system and software design. The participants can suggest alternative interesting aspects of the requirements to view in a line diagram either during the meeting or later.

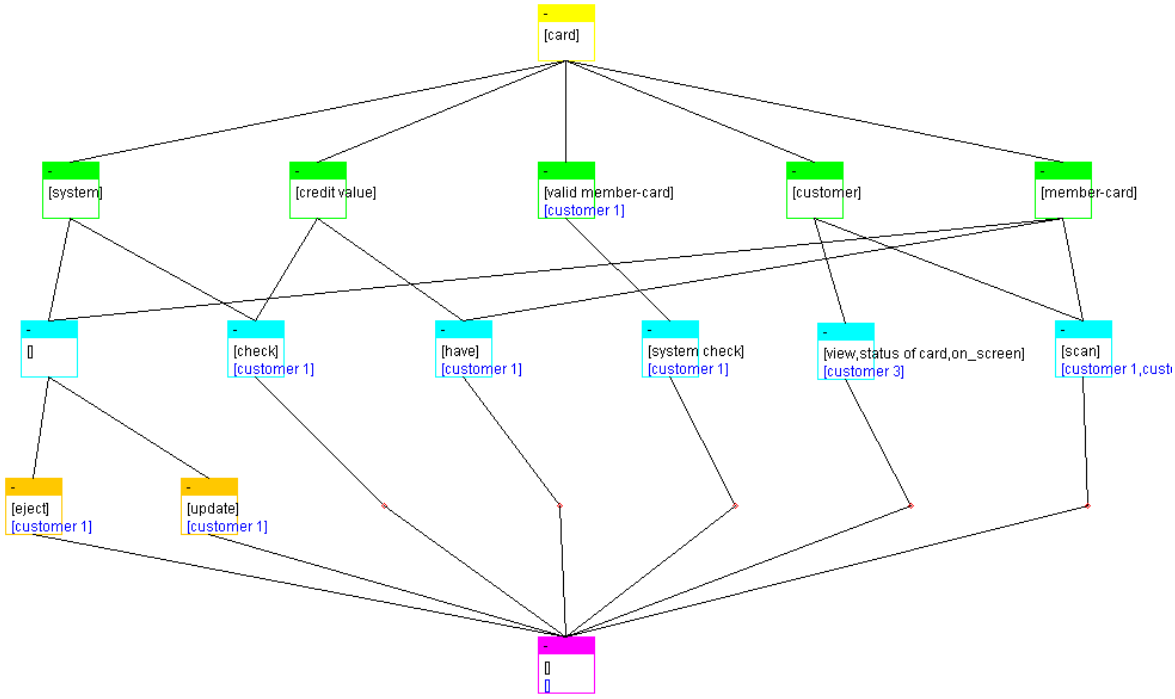

Figure 3. Line diagram for the "Rent Video" use case from Customer 1 and Customer 2 viewpoints for sentences which concern the use of a card.

Figure 3 shows a line diagram including the viewpoints for two agents, referred to as Customer 1 and Customer 2, for the "Rent Video" use case. Just the sentences concerning the card have been included which is why the term "card" is found in the top node. 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 the bottom node on the far right represents the sentence that was written by both Customer 1 and Customer 3 and says that the "customer, scan[s], [the] member-card." This is the only sentence completely shared by the two agents.

As part of our group decision support approach we provide strategies for identifying and resolving conflicts. A number of resolution strategies have been offered but we have found that the five categories offered by Easterbrook and Nuseibeh (1996) cover the actions we have found necessary. These are:

1) Resolve, correct any errors;
2) Ignore, no action is performed;
3) Delay, identify the existence of the inconsistency but defer action until a later date;
4) Circumvent, identify the existence of the inconsistency so it can be avoided;
5) Ameliorate, reduce the degree of inconsistency. This action requires analysis and reasoning.

The last four resolution strategies are relevant for situations in which a complete resolution cannot be negotiated and each one has its appropriate usage. For example, ignoring is a useful strategy where the issue is not that important or pursuing it is not worth the effort or harm it may cause to the end solution. These approaches can be termed as living with inconsistency or 'lazy' consistency (Narayanasway and Goldman 1992) and can be compared to fault-tolerant systems that continue to function after non-critical failures occur.

2.5 Viewpoint Integration

The first step of this phase is review of the terminology used. This will allow us to determine if the viewpoint agents have a common understanding of the terminology and to be able to make concepts more similar for further analysis steps. For example, one viewpoint agent may use the term 'member card' and another viewpoint agent may use the term 'card' to refer to the object used to identify and validate the video library member. To reconcile differences in terminology we use a table of synonyms, hyponyms and hypernyms that the viewpoint agent or project leader can use to map one term to another. Using the example in Figure 3, if one viewpoint agent had used the word 'swipe' instead of 'scan' the table can be used to map the two terms so that only "scan" is displayed and both sentences appear in the same concept node.

The identification of different concepts that share the same terminology will be difficult but the lattice may be able to assist. In such a situation it is likely that some of the terms will be the same but others will not. This will suggest that the two viewpoints are referring to different things even if they use the same word/s. For example, the two sentences {member, requests, receipt}{machine, issues, receipt} in one viewpoint and the sentence {member, issues, receipt} in another viewpoint use the same terminology but represent different concepts. There is obviously an error that needs to be reconciled by the viewpoint agents.

If synonyms, hyponyms and hypernyms are defined, the second step is to determine if two concepts provide the same information. Two or more concepts described by different viewpoint agents can be in consensus if they describe the same action or state using the same terminology. This is usually the case if viewpoint agents describe their viewpoints on the same level of detail. These concepts share the same node in a concept lattice and so are easy to identify.

Partial consensus occurs if two or more formal concepts[1] share some but not all attributes. For example the sentences {machine, provide, receipt} and {machine, print, receipt} share the attributes 'machine' and 'receipt' and are thus subconcepts of the concept {machine, receipt}. This allows the same action or state to be described at varying levels of detail. For example the sentence {machine, print, receipt, receipt show transaction number and transaction type and amount and account balance} gives more information than the sentence {machine, print, receipt} but they describe the same action.

After the identification of concepts giving the same information the viewpoints can be investigated to find information not given in all viewpoints. These can be missing steps or a different sequence of actions or states. Missing steps or missing conditions are represented by concepts which are not shared by all viewpoints. There are many missing steps in Figure 3. The bottom nodes contain the identity of the viewpoint agent who wrote the sentence. Any bottom nodes with only one viewpoint agent indicate that that agent was the only one to have that (partial) sentence. Information about a different sequence of actions or states can only be derived from the 'action/state#'. In Figure 3 we have removed the action numbers from the line diagram to reduce screen clutter but they can be displayed. Where different levels of abstraction are used to specify requirements the analyst may choose to add or drop steps. However, the model that we are left with after negotiations is not expected to show all viewpoints now in total agreement but it must represent what the group are willing to accept.

We accept that living with inconsistency will be necessary and use tags to identify the status of the conflict. These tags are attached to nodes on the line diagram to mark the action taken. They can be displayed or hidden by the user. The use of tags is similar to the use of "pollution markers" (Balzer 1991) that act as a warning that code may be unstable or that the users should carefully check the output. Pollution markers can be used to screen inconsistent data from critical paths that must have completely consistent input. If it is the concept that is being circumvented, ignored or delayed, we mark the concept in the shared model. The updated shared use case and updated individual use cases are used as input in the next cycle of the process.

3. The Group Process and Collaboration using RECOCASE

Collaboration between stakeholders is essential for capturing requirements that are representative of the range of stakeholder needs. The phases presented in the previous section include a combination of individual and group processes. The key goal of this work is to address the problems which can occur in group interactions. In the group setting, people tend to fall into line and try to follow the reasoning of the individual who has the floor, unless the express purpose of the meeting is brainstorming of new ideas. In our approach, we allow the group to work together for brainstorming use cases and identifying viewpoints. However, when it comes to specifying the steps of the use case, which will become the basis of the system requirements, the process needs to be more orderly and must involve reflective as well as reflexive interaction. We believe that allowing a number of individuals, who represent the possible different viewpoints, to independently describe the use case steps will result a more complete set of requirements, encourage participation and ownership of requirements and avoid the problems of some individuals dominating the group and forcing their viewpoint on the rest of the group. The studies described in section 4.1 support our claims.

It would be difficult to classify our approach using any one of the types of systems given in (Dix et. al 1998). Our tool can be seen as a meeting and decision support system in so far as it allows individuals to record their reasoning (arguments) when used to build their own conceptual model and to support the discussion of ideas and concepts when used in face-to-face groups that are synchronously co-located. What makes our approach different to typical meeting and decision support

[1] A formal concept is a pair comprised of a set of object and the set of attribute they share. In our usage an object corresponds to a use case step/sentence and an attribute is the words or phrases that comprise the sentence.

systems is that the team members work at times alone and at other times together to develop individual as well as a co-authored system. The shared conceptual model provides structure, focus and identifies similarities and differences within the group providing a wider communication bandwidth not available when reviewing the individual viewpoints separately.

4. Evaluation of RECOCASE

There are many different aspects of the RECOCASE framework that require validation. These include the usefulness and usability of the: guidelines and controlled language; tool; group process; resolution strategies; and the process model. Since the visual representation of the individual and shared requirements' models is the central part of our group decision support software we began our evaluations with an investigation of the learnability and comprehensibility of the concept lattice and how reasoning with a line diagram compares to reasoning with the original text. Our intial study in 1998 (Richards 1998) found that 10 out of the 12 subjects were able to learn to accurately read a line diagram within a few minutes, that the line diagram supported reasoning about the domain and that questions related to the text could be anwered faster and more accurately using the line diagram rather than the original text. The results were promising but, as noted by Kremer (1998) and evidenced in (Petre and Green 1993) and use of a visual language requires time and effort to learn and this makes evaluation of the line diagram by novices a difficult task.

In early 2002, we conducted further studies with 201 second-year students of requirements, analysis and systems design who participated in a study to evaluate our use case guidelines and the usefulness of the line diagram for reasoning about requirements. Our results showed that reading and reasoning with the line diagram could be learnt by 58% of our subjects after a 5 minute introduction (where any misinterpretations or errors were treated as failure). Four tasks were given, half of the time using the diagram and the other half using text. Answers 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. Depending on the task, answering the questions using the diagrams was 1.5 to 9.9 times faster than using the text on which the diagrams were based.

In late 2002, we designed and conducted studies to evaluate the group process and the RECOCASE-tool. We only present our results for the group process in this paper.

4.1 Design of the Evaluation of the RECOCASE Group Process

In our evaluation of the RECOCASE group process we collected data via surveys, rich text from observing the participants during the individual and group sessions, and examination of the use case descriptions. The test participants were nine 3rd year computing students who all had some knowledge about use cases from current or previous studies. We completed four separate tests, each time with three students.

In Tests 1 and 2 the test participants used the RECOCASE methodology and tool to develop use case descriptions for the automatic video system introduced in section 2.

Test 3 was conducted with the same students as in Test 1, but this time they were given a new problem to solve. We organized the tests in this way to evaluate our methodology on first time and more experienced users. Test 4 was a separate test where three test participants solved the AVS problem without using RECOCASE. We provided the test participants with use case writing guidelines since we anticipated that it would be hard to compare the sentences if they were structured differently. For example, the participants might write composite sentences by using and/or.

A group facilitator (GF) was trained over a period of a week in running all tests, and assisted the participants in drawing the use case diagram, using the guidelines and the tool. The reason for this was that a pilot study performed in June 2002 showed that the GF needs experience, both as a leader and

with the tool. A GF that did not fully understand the RECOCASE methodology or the tool would invalidate the whole test.

The tests presented in this article have been designed to follow the phases in the RECOCASE methodology given in section 2. Tests 1, 2 and 3 were conducted in three sessions as described in Table 2. Test 4 consisted of only one 1.5 hour session where the participants received the same introduction as in session 1 in Tests 1 and 2 without the description regarding the RECOCASE-tool. In Test 1, 2 and 4, the participants were given the AVS problem and brainstormed to identify use cases. The Test 4 Group then together created a use case description for the use case "rent video" on a whiteboard. Before the second session in Test 1, 2 and 3, the GF entered the project data and registered users using the RECOCASE-tool. The second group session was an individual session where the test participants entered their use case descriptions into the application. All the participants were handed a survey where they rated the usability and various features of the tool. The test participants then had a break while the GF created line diagrams by combining the different use case descriptions. In session 3 the group and the GF met and discussed the diagrams. A final shared use case description was created by reconciling the use case sentences.

The inter-viewpoint analysis phase involves the evaluation of the final shared use case. The tool can calculate if the viewpoints have become more similar by determining node by node and agent by agent

Table 2. Session overview for Test 1 and Test 2.

Sessions	Description
Session 1 -Group (75 mins)	– Presentation of the RECOCASE methodology and tool. – Explanation of concepts and the motivation behind the approach. – Explanation and examples of use of guidelines, use case descriptions and line diagrams. – Presentation of the test case. After 5 minutes individual reflection upon the test case, the participants brainstorm together to discover use cases.
Break (75 mins)	– Break while the GF enters project name, actors, agents and use cases into the RECOCASE tool
Session 2 - Individual (45 mins*3)	– Individual session with the tool. Each participant enters their use case descriptions into the tool
Break (75 mins)	– Break while the GF creates line diagrams combining the different viewpoints. The GF compares concepts and detects conflicts.
Session 3 - Group (60 mins)	– The group looks at the diagrams. Negotiate and evaluate the use case sentences. Creates a final use case description.

Table 3. The use case description for "Rent Video" written by Participant 1 in Test 2.

step	action
1	The Customer insert the member-card.
2	The systems checks the validity of the member-card.
3	The member-card is valid.
4	The system checks the credit value of the member-card.
5	The member-card has enough credit value.
6	The system shows success flag.
7	The Customer scans the Video.
8	The system stores the video title
9	The system updates the database.
10	The system updates the member-card.
11	The system ejects the member-cards

Table 4. The use case description for "Rent Video" written by Participant 2 in Test 2.

step	action
1	the customer swipes the card
2	the customer inputs the password
3	if the password is correct
4	if the customer has paid past fines
5	if the customer has returned previous videos
6	the customer scans the video
7	the customer presses the "Logout" button to end the session

how many attributes are shared by different agents divided by the total number of words that belong to that sentence (see Figure 4). Contrasting sentences score 0%, and sentences in complete consensus score 100%. The scores for each sentence for each agent are averaged to produce an overall measure of similarity. Depending on the strategy adopted, such as most similar or most different viewpoints, these distance scores can be used to decide the order in which to reconcile viewpoints.

4.2. Results of the Evaluation of the RECOCASE Group Process

To assess the RECOCASE group process, various data were collected to answer the questions of interest. We present the question and results in a separate subsection. To demonstrate the nature of the output from each test we show the individual use case descriptions for participants 1-3 and the final use case for test 2, in Tables 3-6 respectively. As shown in Tables 3-5, the lengths of the use case descriptions for participants 1, 2 and 3 are 11, 7 and 6 sentences. The final shared use case description (Table 6) contains 15 sentences and is longer and more detailed than any of the individual descriptions. It contains both sentences from the different participants' use cases and new sentences created by the participants during the reconciliation process in session 3. The participants decided not to use substeps in the final use case description. This complies with style 1 in the CP Use Case Writing Rules by Cox *et al.* (2001). The headings of each subsection indicate what was being assessed.

4.2.1 Completeness of use case descriptions

A more complete set of requirements is one of the goals of the RECOCASE approach. We were interested in comparing the length of individual and shared use case descriptions as relative length can be an indicator of a more complete set of requirements. Table 7 shows the length of the individual and final shared use case descriptions. In Tests 1 and 2, all the participants wrote shorter and less detailed use case descriptions than the final shared use case description. This is in contrast to Test 3 (which involved the participants from Test 1), where two of the test participants wrote longer use case

Table 5. The use case description for "Rent Video" written by Participant 3 in Test 2.

step	action
1	The customer scans the desired video
2	The customer views the status of the card on screen
3	The system checks for credit problems
4	The system verfies the customer
5	The system deducts video amount
6	The system issues a receipt to customer

Table 6. The final shared description for the "Rent Video" use case in Test 2.

step	action
1	The customer insert the member-card
2	The system verifies the member-card
3	The customer enters the password
4	The system verifies the password
5	The customer views the status of the card on screen
6	If the member-card has enough credit value
7	The customer scans the video
8	The system stores the video id
9	The system updates the credit value of the member-card
10	The customer opts to log out
11	The system updates the customer record
12	The system given an option for issuing a receipt
13	If the customer selects a yes option
14	The system issues a receipt to customer
15	The system logs the customer off

Table 7. The number of sentences in the participants' use case descriptions and the final shared use cases.

Test / Participants	Number of Sentences in Use Case Description			
	Part 1	Part 2	Part 3	Final Shared Use Case
Test 1	7	7	6	12
Test 2	11	7	6	15
Test3	17	6	20	16
Test 4	N/A	N/A	N/A	10

descriptions than the final shared use case. When the group reconciled the use case sentences in Test 3, they discovered that some of the steps in the long use cases were too detailed and actually described a low level design. Some of the sentences could also be classified as belonging to other use cases rather than this specific one, and were removed from the diagrams by the participants. In this way, the process was successful in not only capturing a more complete set but also acted as a filter for inappropriate steps. Even though the second problem appears to have been more difficult to specify than the first, the participants were more confident in writing use case descriptions and also more effective at using the tool. This is presumably because they were not first time users in this test.

Test 4 was conducted with the same test problem as in Tests 1 and 2 but without using the RECOCASE-tool. The resulting use case description from Test 4 consists of 10 lines and is shorter than any of the other use case descriptions. A comparison of the different use case descriptions showed that the description produced in Test 4 described the same main parts of the use case as Test 1 and 2, but provided fewer steps and less detail. These results indicate that RECOCASE provides a means to gain a longer and more detailed use case description. A possible reason for this can be that by using the tool individually the participants think on their own before they meet the group to decide the final description and that this prevents group thinking.

4.2.2 Representativeness of the final shared use case description

Another goal of RECOCASE is development of a more representative set of requirements. The final shared use case descriptions are mainly created by extracting sentences from the participants' individual use case descriptions. The results from the evaluation of the similarity between viewpoints in Test 2 are shown in Figure 4. The column "rent video" shows the comparison between the different viewpoints and the final shared use case description. Participants 1 and 3 had written descriptions that were 62% and 70% similar to the final description. Although the description by participant 2 in Test 1 only had a similarity of 57% to the final description, they indicated that they were satisfied that the final use case description adequately covered their viewpoint (see section 4.2.5).

Figure 5 provides a summary of Tests 1-3 of the percentages of each individual use case found in the final use case description. It can be seen that all viewpoints were considered and included to some extent.

4.2.3 Comprehensibility of the Line Diagrams for requirements negotiation

Although the GF guides the group members in the reconciliation process, understanding the diagrams and their role in reconciliation should improve user participation and satisfaction. To measure the appropriateness of the line diagram for reasoning about requirements a number of questions were asked. As shown in Figure 6, all the participants in Tests 1 and 2 answered that the diagrams were easy to understand. Surprisingly, they found the diagrams harder to understand in Test 3. Table 7 showed that the participants wrote longer use case descriptions in Test 3 and this again made it necessary to construct more diagrams. Four diagrams were constructed compared to three in Test 1 and two in Test 2. The diagrams in Test 3 were larger than those constructed in the other tests and this probably caused the difficulties in reading the diagrams.

The participants in Test 1 and 2 found the diagrams useful in visualizing the system requirements and they also thought they were useful in the negotiation process. Again the ratings are lower in Test 3, and we conclude that too many or too large diagrams decrease the diagrams' usefulness and should therefore be avoided.

Vp Name	customer 1	customer 2	customer 3	rent video
customer 1	100%	56%	35%	62%
customer 2	41%	100%	36%	57%
customer 3	44%	38%	100%	70%
rent video	50%	33%	35%	100%

REPORT:

rent video against customer 2 = 33.727856

****

rent video compare with customer 3 : the system logs the customer off = 0.6666667 -
rent video compare with customer 3 : If the customer selects a yes option = 0.083333336 -
rent video compare with customer 3 : The system gives an option for issuing a receipt = 0.083333336 -
rent video compare with customer 3 : The system updates the customer record = 0.14285715 -
rent video compare with customer 3 : The customer opts to log out = 0.14285715 -
rent video compare with customer 3 : The system updates the credit value of the member-card = 0.14285715 -
rent video compare with customer 3 : The system stores the video id = 0.25 -
rent video compare with customer 3 : If the member-card has enough credit value = 0.25 -
rent video compare with customer 3 : the system verifies the password = 0.2 -
rent video compare with customer 3 : the customer enters the password = 0.2 -
rent video compare with customer 3 : The system verfies the member-card = 0.33333334 -
rent video compare with customer 3 : The system issues a receipt to customer = 1.0 -
rent video compare with customer 3 : The customer views the status of the card on screen = 1.0 -
rent video compare with customer 3 : The Customer scans the Video. = 0.6666667 -

Figure 4. The RECOCASE-tool displays the similarities between the different viewpoints and the final shared use case description in Test 2.

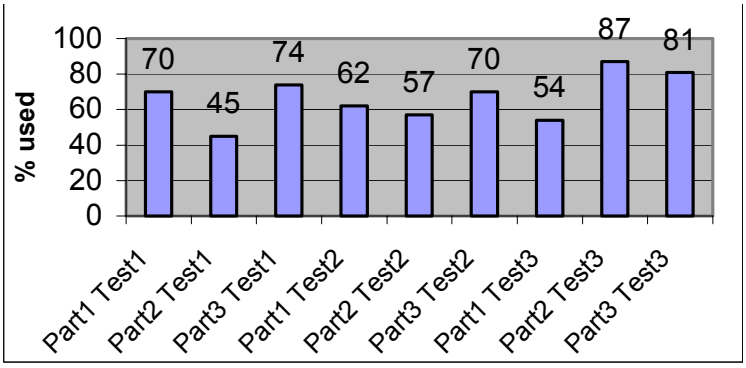

Figure 5. The percentage of individual descriptions retained in the shared use case.

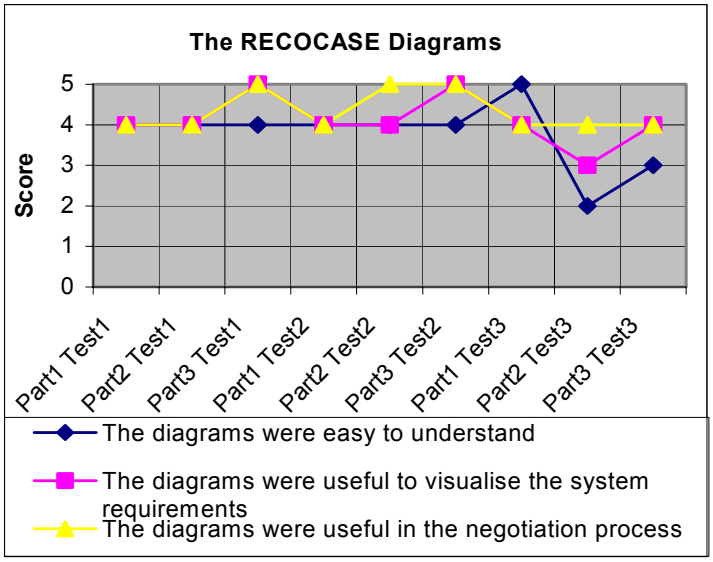

Figure 6. The test participants' ratings of the diagrams created by the RECOCASE tool.

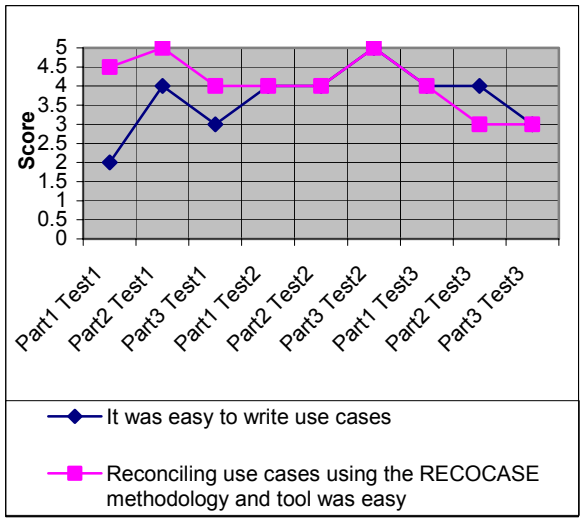

Figure 7. The test participants' ratings of the difficulty of writing use cases and the reconciliation process.

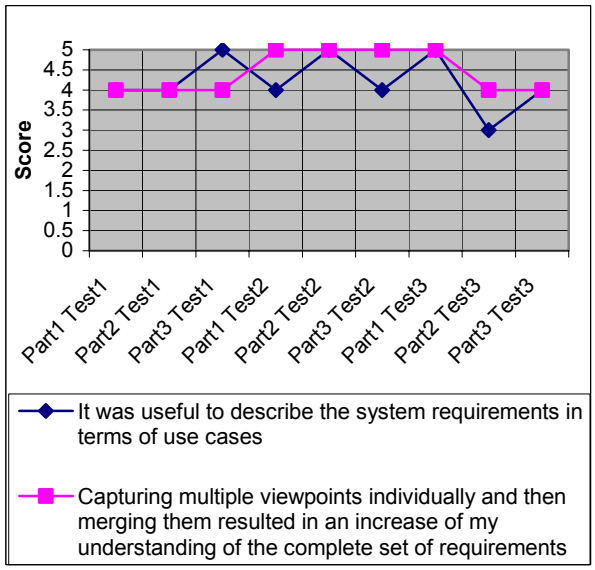

Figure 8. The participants' ratings of the usefulness of use cases in describing requirements and the RECOCASE process.

4.2.4 Complexity of the RECOCASE process

If the process provided is too complex, time consuming or requires highly skilled users then the likelihood of the process being adopted is low. The low acceptance of formal methods for requirements specification is evidence of this. Some of the participants found it difficult to write use case descriptions. This is illustrated in figure 7 and was also observed by the authors during the tests. The group members expressed that it was hard to get started writing the use case descriptions, and observations and time measurements from Test 1 and 2 where the participants wrote two descriptions

showed that they were usually faster and more confident when writing the second description. The participants had then "learned" how to structure and compose the sentences. The group members in Tests 1 and 2 found it easy to reconcile the use case sentences by using the RECOCASE methodology and tool. Again, the participants in Test 3 described the process as more difficult, probably because of the size and the number of diagrams.

4.2.4 Usefulness of the RECOCASE process

We decided a couple of years ago to apply our reconciliation process to use case descriptions. Our reason for this was its growing acceptance and that focusing on one use case at a time provided a manageable chunk of requirements to work with. To test whether our decision was reasonable, we asked the participants to assess the use of use cases and, in particular, our use of them in the RECOCASE process. In Figure 8 we see that the group members found it useful to describe the system requirements in terms of use cases. The participants also liked the viewpoint approach that RECOCASE supports and said that the process of capturing multiple viewpoints individually by using the tool and then merging them resulted in an increase in their understanding of the complete set of requirements.

4.2.5 Satisfaction with the final requirements and the process.

Even though all the participants in Tests 1, 2 and 3 expressed that the resulting use case description covered their viewpoint, not all of them were completely satisfied with the resulting use case description (Figure 9). This was interesting and not easy to explain. Initially we thought that the less satisfied participants may have had less of their own use case included in the final description but this is not the case. It is possible that the participant felt that the final use case had the appropriate concepts but could have been worded or structured better. Another reason could simply be the subjective nature of the notion of satisfaction and that some event during the activity, such as someone disagreeing with them or feeling tired at the end of the day, made them feel less than satisfied. All the test participants agreed or strongly agreed to the statement saying the process of reconciling requirements from different viewpoints had been successful. This is a very positive and encouraging result.

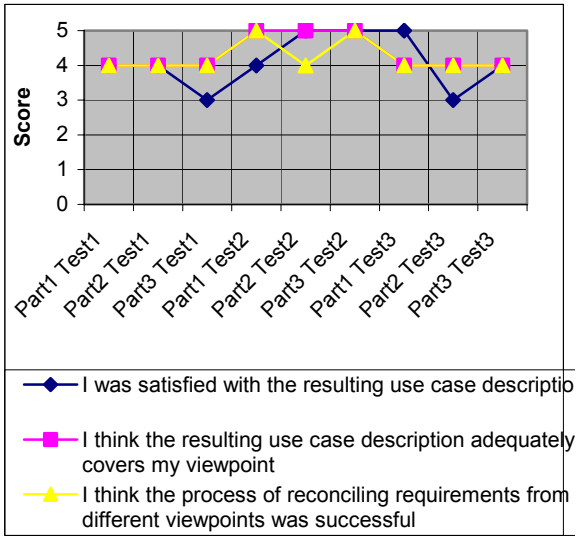

Figure 9. The test participants' ratings of the final shared use case and the reconciliation process.

4.2.6 Final observation regarding the group process

The most notable difference between writing use case descriptions with the tool (tests 1, 2 and 3) and without the tool (test 4) concerned the group process. We found that all participants in Tests 1, 2 and 3 were involved in the discussion, though sometimes needing to be asked a direct question regarding their viewpoint. In contrast, the group in Test 4 did not work well together. Participant 2 would not get involved even when asked for his opinion. Had he already expressed an opinion by entering a viewpoint, the GF would have had specific questions that could be asked. Using the tool, participants could write whatever they wanted at their own pace and using their own terms. Their sentences would be displayed in diagrams together with the other participants' sentences. This may have given them a sense of ownership of their sentences and resulted in increased involvement in the discussion (and hopefully the project overall) compared to group brainstorming of use cases as in test 4. We would need to conduct further studies without the tool to see if similar behaviour emerged. Our focus in these studies had been on the tool and the RECOCASE process.

5. Conclusion

As described in the previous section we have performed initial but substantial evaluation studies of the value of the line diagram and the feasibility of our group process. We have found that a visualization of the requirements is preferred and more accurate and quicker than text for reasoning and comparison purposes. The group process and tool seem to work well and produce more complete descriptions of the system requirements. We are just commencing an indepth comparison with similar groupware tools. In particular we will explore the natural language and group process work done by (Al-Ani *et al.* 1999 which uses the gIBIS tool (Conklin and Begeman 1989)) and (Ambriola and Gervasi 2000). To allow distributed users to participate in group decision making we would need to address the issue of distributed meeting rooms. We may be able to integrate our approach with the work by Greenberg and Roseman (1998) which uses a room metaphor to allow work to occur individually and as a group as well as synchronously and asynchronously. We need to consider the numerous issues that differentiate face-to-face communication from text-based communication. The findings of (Damian *et al.* 1999) indicate that the group may more successfully achieve their goals by working in a distributed mode without the emotional complications of face-to-face interaction. Our results showed that groups generate more complete requirements than individuals on their own. Our findings also show that even more comprehensive requirements can be obtained by capturing requirements individually first and then merging them into one shared specification.

We are currently seeking an industry partner to further develop this work into a practical and industry-relevant approach. However, our evaluations show that the approach already offers a viable solution to capturing use case viewpoints in natural language from multiple stakeholders which can then be visualized and compared resulting in a more complete and representative set of requirements upon which the system design can be based.

6. 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), Macquarie University, Sydney, pp. 15-26.

Ambriola, V. and Gervasi, V., 1998, "The Case for Cooperative Requirement Writing," Proceedings of ECOOP Workshops, pp. 477-479

Balzer, R., 1991, "Tolerating Inconsistency," Proceedings of 13th International Conference on Software Engineering (ICSE-13), Austin, Texas, USA, IEEE Computer Society Press, pp. 158-165.

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

Conklin, J. and Begeman, M., 1989, "gIBIS: A Tool for all Reasons," Journal of the American Society for Information Science, 40(3):200-213.

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

Damian, D., Eberlein, A. Shaw, M. and Gaines, B., 1999, "Studies in Distributed Software Requirements Engineering," Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management (KAW'99), Banff, AB, Canada.

Darke, P. and Shanks, G., 1997, "Managing User Viewpoints in Requirements Definition," Proceedings of 8th Aust. Conf. on Info. Systems.

Dix, A., Finlay, J., Gregory, A. and Russell, B., 1998, "Human Computer Interaction," 2nd Edition, Prentice Hall, London, UK.

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

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.

Greenberg, S. and Roseman, M., 1998, "Using a Room Metaphor to Ease Transitions in Groupware," Research Report 98/611/02, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, http://cpsc.ucalgary.ca/grouplab/papers/

Jacobson, I., 1992, "Object-Oriented Software Engineering," Addison-Wesley.

Kremer, R., 1998, "Visual Languages for Knowledge Representation," Proceedings of the Eleventh Workshop on Knowledge Acquisition Modeling and Management (KAW'98), Banff, Canada.

Molla, D., Schwitter, R., Hess, M. and Fournier, R., 2000, "Extrans, an answer extraction system" T.A.L spec. issue on Info. Retrieval oriented Natural Language Processing.

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

Narayanaswamy, K. and Goldman, N., 1992, "Lazy Consistency: A Basis for Cooperative Software Development," Proceedings of International Conference on Computer-Supported Cooperative Work (CSCW'92), Toronto, Ontario, Canada, ACM SIGCHI & SIGOIS, pp. 257-264.

Petre, M. and Green, T.R.G., 1993, "Learning to Read Graphics: Some Evidence that 'Seeing' an Information Display is an Acquired Skill," Journal of Visual Languages and Computing, 4(1):55-70.

Richards, D., 1998, "An Evaluation of the Formal Concept Analysis Line Diagram," In Slaney, J., Antoniou. G and Maher, M.J. (eds), Poster Proc. of 11th Australian Joint Artificial Intelligence Conf. AI'98, Griffith University, Nathan Campus, Brisbane, Australia, pp. 121-133.

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

Wille, R., 1992, "Concept Lattices and Conceptual Knowledge Systems," Computers Math. Applic., (23) 6-9:493-515.

Wirfs-Brock, R., 1993, "Designing Scenarios: Making the Case for a Use Case Framework," Smalltalk Report Nov-Dec 1993. NY, NY: SIGNS Publications.