Resolving Requirements Conflicts with Computer-Supported Negotiation

STEVE EASTERBROOK
School of Cognitive and Computing Sciences,
University of Sussex,
Falmer, Brighton, BN1 9QH, UK.
<e-mail: Easterbrook@cogs.susx.ac.uk>

1. INTRODUCTION

Conflict is an inevitable part of both requirements elicitation and system design. As McDermid points out elsewhere in this volume, requirements are negotiated, not captured. During this process, the participants will disagree over how to interpret features of the application domain, what the requirements for a new system are, and how to meet those requirements. Conventional analysis techniques tend to suppress conflict, making any resolution untraceable and adding to the communication problems. This chapter argues the need for explicit support for conflict management, and presents a model of computer-supported negotiation to address this need.

In the software engineering literature, mention of the need to handle conflict is rare. This is surprising given the importance attached to it in the social sciences. For many years it has been recognised in management science and sociology that conflict is an inevitable feature of group interaction, often to be harnessed for its positive aspects, rather than suppressed [Robbins 1974], [Deutsch 1973], [Strauss 1978]. Some recent software engineering research has identified conflict as an issue [Curtis, Krasner & Iscoe 1988], [Anderson & Fickas 1989], [Feather 1989], [Robinson 1990], although as yet little progress has been made towards understanding how conflict might be handled.

An apt example of organisational conflict is given by Robbins [1974]: a newly elected city manager has promised an immediate improvement in rubbish collection. After several months the citizens complained that there was no improvement. On investigation it turned out that the citizens regarded "improved service" to mean more frequent collection, whereas the city manager had meant earlier, quieter and more economical collection. It is not difficult to see how similar conflicts might arise during the introduction of a new software system.

The term conflict can be taken to mean any interference in one party's activities, needs or goals, caused by the activities of another party. Conflict can be characterised as disagreement among the originators of the requirements (or *problem owners*) and this disagreement may lead to inconsistencies in the specification. However, disagreements do not always lead to inconsistency, and inconsistencies do not always indicate the presence of conflict.

Typical requirements analysis methods are geared towards the development of a consistent specification. They do not allow conflicts to be expressed, let alone constructively resolved. Indeed, we could characterise existing methods as conflict avoidance, in that they prescribe particular approaches, which assist software practitioners to break problems down and resolve design decisions in particular ways. Uncertainty is reduced by the provision of the collective wisdom embodied in the methodology [Lehman 1990].

While this approach helps reduce conflict within the requirements specification, it does not help with conflicting requirements. The demand for a single consistent specification means that

requirements conflicts are suppressed when the specification is written. Formal methods do not necessarily help, even though formal languages are intended to prevent ambiguity and inconsistency [Finkelstein, Finkelstein & Maibaum 1990]. The ability to reason formally with specifications is a huge step towards detecting the presence of conflict, but carries the implication that inconsistencies are errors which must be eliminated. Methods developed to support formal specification reflect this philosophy, and miss the chance to explore conflict.

If much of software engineering is geared towards conflict avoidance, this in itself may not be a problem, for two reasons. Firstly, requirements conflict may not be as common as has been suggested. Secondly, avoidance may be a valid way of tackling conflict, especially where it prevents energy being wasted on fruitless confrontation. We argue in this chapter that requirements conflicts are extensive in socio-technical systems, and that avoidance is not a satisfactory approach to handling these conflicts. Furthermore, handling conflict in more direct ways actually improves understanding of the requirements and assists with requirements validation.

1.1. Sources of Conflict

Conflict is a part of the nature of an organisation [Robbins 1989]. In particular, it is both a source of and a response to organisational change. Software design necessarily involves organisational change, and even end-user involvement in the design process does not remove conflict [Wastell 1993]. Two obvious sources of conflict in requirements engineering are conflict between the participants' perceptions of the problem, and conflict between the many goals of a design. Other sources of conflict include conflicts between suggested solution components; conflicts between stated constraints; conflicts between perceived needs; conflicts in resource usage; and discrepancies between evaluations of priority.

Curtis *et. al.* [1988] reveal the effect of conflict in software engineering. They identified three major problem areas: the thin spread of application domain knowledge; fluctuating and conflicting requirements; and breakdowns in communication and co-ordination. Each of these areas is a source of conflict, and each depends crucially on communication between participants as a basis for any solution. A good conflict resolution approach necessarily emphasises communication between parties.

Conflicting and fluctuating requirements have many causes, from change in the organisational setting and business milieu, to the fact that the software will be used by different people with different goals and different needs. Handling constant change in requirements (*"requirements maintenance"*) requires an evolutionary approach that must be based on accurate capture of rationales and process information.

In many cases, there will be disagreement over the nature of the application [Dobson, 1993]. This could be dealt with by restricting the scope of the new system so that only a single goal (or view of the application) is addressed. For large scale, open systems this is not practical [Blum 1991]. Hence, requirements analysis must recognise and deal with the existence of multiple, conflicting perspectives. Furthermore, the occurrence of conflicting perspectives may not always be distinguishable from instances where people are describing essentially the same concepts, but using different terms. Even formal representation schemes allow enough variation in style so that there may be many different ways of saying the same thing.

1.2. Conflict Resolution

If there is no means of expressing conflict within the method, where conflicts do occur, they are likely to get suppressed. If they remain suppressed, it will lead to dissatisfaction with the requirements process. The requirements may be based on a single perspective, at the cost of any alternative perspectives. If the conflicts are eventually resolved, the resolution must be carried

out outside the framework of the method, probably at an inappropriate time, using an undesirable means. Resolution thus achieved is untraceable, making decisions irreproducible, and rationale information invalid.

Suppression of conflict will have serious effects on the remainder of the software development process. Suppressed conflict may lead to the breakdown of the requirements process, or the withdrawal of participants. Failure to recognise conflict between the perspectives of the participants will cause confusion throughout the lifecycle. Participants' understanding of the specification will differ, leading to further misunderstandings during design and implementation.

These problems make a study of conflict resolution desirable. In the next section we survey research on conflict in relevant disciplines. We will then synthesise a model of computer-supported negotiation, which can be used as a basis for the requirements process.

2. SURVEY OF RELEVANT FIELDS

2.1. Terminology

We will talk about conflict between *parties*, meaning individuals, groups, organisations, or even different roles played by one person. Similarly, we will refer to *participants* of the resolution process, to cover a similar diversity. Not all parties to a conflict need necessarily be participants in its resolution. A *Resolution Method* may be used to settle a conflict, although some conflicts will not need to be explicitly resolved. We distinguish three broad types of resolution method: *Co-operative* (or collaborative) methods, which include negotiation and education; *Competitive* methods, which include combat, coercion and competition; and *Third Party* methods, which include arbitration and appeals to authority.

Negotiation is characterised by participants exploring the range of possibilities, to find a settlement which satisfies all parties as much as possible. Such an approach has been variously termed *integrative* or *constructive* negotiation (to distinguish it from *distributive*, or *competitive* negotiation). This definition of negotiation is not universal. Authors such as De Bono [1985] restrict negotiation to its distributive variety, implying a process of bidding and concession-making, and so attack it as being inferior to an integrative approach. We prefer to give negotiation its broader definition, and call the concession-making process *Bargaining*.

There are other collaborative methods than negotiation. Some conflicts might be resolved by education, where the participants gain a better understanding of the problem, or simply learn about each other's viewpoint. This may include reformulation of the problem, so that it disappears, or becomes unimportant.

Competition concentrates on achieving maximum satisfaction for a participant, without regard for the degree of satisfaction of other parties. However, a competitive approach is not necessarily hostile.

Third Party Resolution covers any situation where participants appeal to an outside authority. There are two types of third party resolution: those in which the cases presented by each participant are taken into account, which we term *judicial*; and those where a decision is determined arbitrarily (e.g. tossing a coin), or by factors other than the cases presented (e.g. by the relative status of the participants), which we term *extra-judicial*.

2.2. Mathematical and Economic Models

Decision theory is a prescriptive approach to analysis of pre-specified alternatives. The interesting problems are concerned with resolving multiple conflicting objectives [Keeney & Raiffa 1976]. Decision theory assumes a single entity is making a choice, in contrast to conflict

where there is more than one entity, each with a different perspective. Decision theory has a role in conflict resolution in helping participants to evaluate bids, to justify such evaluations, and to persuade the other participant(s) that a solution is satisfactory.

Bargaining theory is an attempt to produce descriptive models of bargaining processes, and is especially concerned with commerce and politics. Patchen [1970] notes that the more complete models include wider concerns than bids and outcomes, including how participants influence each other's behaviour, and factors such as the cost of various actions and the cost of delaying agreement. Bargaining theory frequently makes use of the *joint outcome space* [Thomas 1976] as a tool for illustrating how a party perceives the options (figure 1). Note that there may be unperceived possibilities that provide better resolutions. Bargaining theory does not indicate how these might be found.

Game Theory is the theory of rational decision in conflict situations [Rapoport 1974]. Participants are regarded as players, and game theory examines the strategies used by the players in the process of trying to achieve particular outcomes. Game theory makes use of the payoff matrix (figure 2), reflecting the assumption that the set of outcomes is known (though not necessarily finite), and that associated with each outcome is a calculable payoff for each player. Limitations of game theory include the assumption that the payoff for any action are known with certainty by all players. However, game theory does produce some useful information about the kinds of strategy that can be used to induce co-operation and the relative success of various strategies [Axelrod 1984].

Group Decision Making is the normative study of how individual preferences can be combined into a group decision. Luce and Raiffa [1957] define the problem as that of finding a method, or *welfare function* for combining individual preference rankings into a social preference, which satisfies properties such as fairness and representativeness. Work on group decision making extends decision theory to cope with more than one decision maker, but still suffers from the assumption that all the options are known.

2.3. Behavioural Models

Conflict Theory studies conflicting pressures in society, recognising that there are many different groups with different goals, and that conflict is a frequent occurrence. Deutsch [1973] gives a list of issues involved in conflicts:
- Control over resources
- Preferences and nuisances, where the tastes or activities of one party impinge upon another.
- Values ("what should be"), where there is a claim that a value, or set of values should dominate
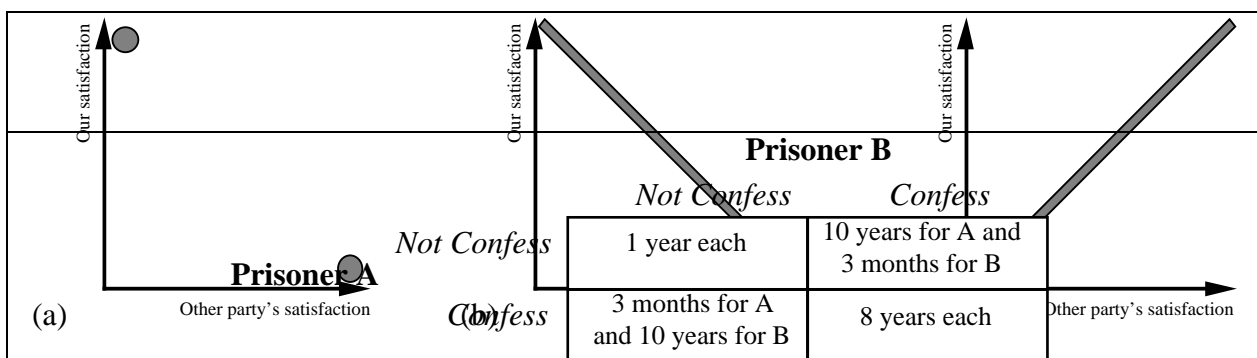

Figure 1: The Joint Outcome Space, illustrating how the options are perceived in (a) a win/lose conflict, (b) a zero-

Figure 2: The payoff matrix for the prisoner's dilemma. Each prisoner can decide independently whether to confess or to keep quiet. If they confess...

- 4 -

    – Beliefs ("what is"), when there is a dispute over facts, information, reality etc.

    – The nature of the relationship between the two parties

Robbins [1989] adds that communication problems are a major cause of what he terms *pseudo-conflicts*.

A major concern of *Organisational Psychology* is how communication and co-ordination of teams can be effected. Early work tended to assume all conflict was undesirable, and so should be eliminated, although empirical studies have demonstrated that conflict is endemic [Easterbrook *et. al.* 1993]. Moreover, Robbins [1974] advocates that conflict management include not just resolution of conflict, but stimulation of conflict too. This is a result of observations that conflict has a useful role as a stimulus to innovation, by questioning and evaluating received wisdom. It is also a major weapon against stagnation and resistance to change.

A number of models have been proposed for conducting face-to-face negotiation in a commercial setting. Scott [1988] uses a four stage model to pace the negotiation: Exploration; Bidding; Bargaining; and Settling. The exploration stage allows participants to explore a range of possibilities before any confrontation takes place. Stefik *et. al.* [1987] suggest that removing the personal attachment to positions prevents polarisation. If participants can dispense with the feeling of ownership of ideas, they can reduce the associated emotions when ideas are discarded or adopted. Similarly, Fisher & Ury [1981] recommend that rather than bargaining over positions, participants should focus on interests, and investigate options for mutual gain. This allows the participants to explain to each other their interests, and discover shared goals which were previously obscured from both.

2.4. Conflict in Computing Science

Various fields of computer science have addressed the problems of conflict. The ubiquity of conflict in the real world means that attempts to model the world, most notably in AI, have to deal with it.

Knowledge-based systems rely on consistent knowledge for their inference mechanisms to work. In most cases, this is achieved by only consulting a single expert. Inconsistencies are attributed to mistakes and are eliminated through debugging and refinement. This insistence that expertise must be consistent and rational means that knowledge acquisition becomes not so much the modelling of an expert's behaviour, but the synthesis of a rational domain model [Shaw & Woodward 1989]. This need not resemble any mental model used by the expert, and conflicts can be filtered out. Although this rationalisation process can be undertaken with a group, problems of conflict cannot be avoided so conveniently. A single expert will feel pressure to appear consistent and rational; for multiple experts, no such pressure can be applied.

Distributed Artificial Intelligence (DAI) divides up problem-solving activities among agents with specialist knowledge [Huhns 1987]. This handles conflicting knowledge by allowing different agents to develop and maintain alternative hypotheses. Most DAI systems assume agents share the same goal. Rosenschein [1985] notes that in real world situations, perfect co-operation never happens, as the goals of any two agents never coincide exactly. He uses payoff matrices from game theory to compare goals, and discusses various situations in which conflict of goals can occur, and how they can be resolved. However, DAI has not progressed much beyond work on models of belief and game theoretical studies of interaction.

Computer-Supported Co-operative Work (CSCW) provides a number of tools intended to improve group collaboration and hence manage conflict. *Argnoter* [Stefik *et. al.* 1987] is intended for use in evolving designs. It attempts to overcome three major causes of dispute: personal attachment to positions, unstated assumptions, and unstated criteria. Proposals are

presented using webs of interconnected windows. Reasons for and against are linked to each proposal. Finally, the assumptions made by the arguments and criteria for decision making are elicited. The assumptions can be grouped together into *belief sets*, to characterise points of view, and to explore the consequences of those views. Stefik compares the tool to a spreadsheet, in that it does not understand the proposals and arguments that it manipulates, but can compute logical relationships between them.

Hypertext offers the ability to support the collaborative elicitation and organisation of ideas over longer durations. As Schuler [1988] points out, hypertext provides an excellent vehicle for supporting (but not supplanting) negotiation. Differing opinions and viewpoints can be represented and linked in the same system, encouraging plurality rather than stifling it. For example, SYNVIEW uses a model of reasoning and debate to organise many viewpoints, with any group decision-making or voting based on access to a common body of material [Lowe 1986]. Similarly, IBIS explicitly represents and links together positions, issues and arguments [Conklin 1989].

2.5. Conflict in Requirements Engineering

A number of studies have concluded that conflict has an important role to play in requirements engineering. For example, Curtis, Krasner & Iscoe [1988] identify *exceptional designers* who have a deep understanding of both the application domain and the design process which enables them "to integrate different, sometimes competing perspectives on the development process". Several recent models of the requirements process attempt to model such perspectives explicitly.

Nuseibeh & Finkelstein [1992] formalise the notion of a *ViewPoint* as having the following components:
  a style, which is the representation scheme used;
  a work plan, which describes development actions and strategy for the viewpoint, and any constraints on it;
  an area of concern, or domain;
  a specification, which is the set of statements in the viewpoint's style describing the area of concern;
  a work record, which describes how the specification developed, and its current status.
This model abstracts away from the people involved, allowing one person to have several viewpoints (as a person may have several areas of concern), and also for one viewpoint to represent several people (where people share an area of concern).

Feather [1989] uses a basic specification as a point of departure for development along separate lines of concern. At some later stage these are merged to produce a single specification, which will then reflect all the concerns. This model delays the resolution of conflict between separate concerns until after the information gathering stage. While it is not yet entirely clear how best to merge the parallel elaborations, Feather has examined the different types of conflict that occur.

One approach to easing the integration of separate specification components is through tools which support negotiation. Robinson [1990] describes tools that allow a single arbitrator to evaluate the preferences expressed by various perspectives, and guide the search for new solutions which satisfy all perspectives. A single domain model is used, expressed as a hierarchy of goals in which perspectives associate different values with the goals. Integration involves searching for novel combinations of proposals, which increase the satisfaction of all perspectives' goals. This is done using a joint outcome space on which an ideal, but probably unachievable combination of perspectives is used to stimulate consideration of other combinations that come close to this ideal.

These approaches to requirements engineering all question conventional models that ignore conflict. Work is needed to clarify how conflicts based on differing requirements can be resolved. One major issue is the need to establish common ground between viewpoints. Participants need enough common ground to communicate; indeed, such common ground is needed before conflict can be expressed and recognised. In many of the above models, the common ground is assumed.

A second issue is how the resolution is devised. Anderson & Fickas [1989] suggest that in well charted domains, the experts will be aware of typical conflicts and how to deal with them. Hence, resolutions can be elicited from experts. However, standard solutions are not necessarily the best, nor will they always work in the environment introduced by computerisation. We would also add that the organisational environments of socio-technical systems are *not* well charted domains.

3. A MODEL FOR CONFLICT RESOLUTION

We now present a model for the management of conflict in requirements analysis. The model is based on the behavioural approaches used in organisational design, and addresses the need to separate the people from the problem, in order to avoid the polarising nature of arguing from entrenched positions. The twin goals of encouraging expression of conflict and providing productive resolution methods form the basis for this model.

A support tool, *Synoptic*, has been implemented to demonstrate the feasibility of the model. *Synoptic* displays viewpoints side by side, allowing the participants to compare and annotate them. Discrepancies noted by the participants are then used to prompt for underlying assumptions and issues. The support tools are designed to provoke discussion of the conflict situations as much as elicit a suitable resolution. Hence, the model is prescriptive, without being a rigid formal process.
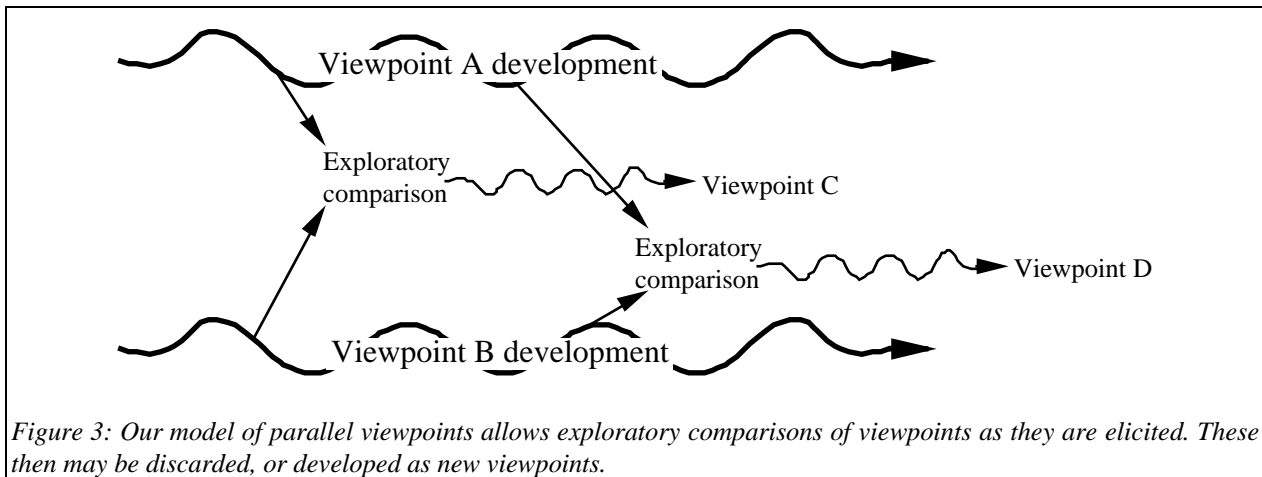
3.1. Basis for the Model

Research into group behaviour indicates that conflict can lead to higher quality solutions [Brown 1988]. In requirements analysis, exploration of the areas where participants descriptions differ can lead to a much better understanding of the domain [Eden 1989]. This is a strong argument for conflict to be carefully managed, with participants encouraged to express divergent views. This will ensure that the requirements specification does not reflect just one point of view, and does not ignore concerns which interfere with the dominant concern.

In software design, effective collaboration is essential. Hence, encouragement of conflict must be matched with resolution methods which strive to satisfy all parties. An integrative approach should be adopted, to ensure that when divergent views arise they are incorporated into the process. The ultimate goal of requirements elicitation should be to identify and represent all concerns.

3.1.1. Specification Context

We use the viewpoints model proposed by Nuseibeh & Finkelstein [1992] as a basis. Viewpoints allow the expression of conflict by providing alternative descriptions, while individually, each viewpoint remains consistent. Each viewpoint has an *originator* – the source associated with the viewpoint – and contains a self-consistent description of an area of knowledge. Easterbrook [1993] describes an architecture for eliciting viewpoints.

Figure 3: Our model of parallel viewpoints allows exploratory comparisons of viewpoints as they are elicited. These then may be discarded, or developed as new viewpoints.

During elicitation, comparisons between viewpoints can shed new light on them. Such comparisons may involve conflict resolution, but are intended to be exploratory; they do not entail merging the viewpoints. Feedback from the comparisons can be used in the development of the viewpoints, for instance to modify terminology, or to elicit information that the originator neglected. The results of any exploratory integrations are treated as new viewpoints, representing coalitions (figure 3). The process of parallel development of viewpoints – with exploratory integrations being initiated at any point – provides the context for our model of conflict resolution.

3.1.2. Detection of Conflict

The first problem for conflict resolution is to recognise that a conflict exists. This might be harder than it seems for a number of reasons. The terminology used by the participants is unlikely to match exactly [Shaw & Gaines 1988], and the styles in which knowledge about an issue is expressed will differ. Also, participants will have different areas and different amounts of knowledge, making it difficult to make comparisons. These problems make it hard to tell where participants are agreeing, let alone where they are disagreeing.

Our definition of conflict was based on interference: two parties are in conflict if the activities of one adversely affect the interest of another. Hence, viewpoints are free to differ, and only conflict when that difference matters for some reason, leading to interference. There are a number of situations in which the differences matter:
– When viewpoints need to be compared.
– When there is a need to reason with knowledge from several viewpoints.
– When the originators insist their viewpoints are "better" than others (and so perhaps should be adopted at the expense of them).
– When a coherent description is needed for further progress.
Under normal circumstances, differences between viewpoints are ignored, allowing them to develop independently. By only entering the conflict resolution process when differences between viewpoints matter, we avoid resolving conflicts unnecessarily.

Note that we deliberately ignore the distinction that Deutsch [1973] draws between real and apparent conflicts at this stage. Apparent conflicts would include: where one party has misunderstood another's position; where viewpoints use different terminology to describe the same thing; and where the interests do not interfere, and can be combined directly. All these are treated as conflicts, and part of the task of the negotiation process is to identify what type of conflict has occurred, and hence whether resolution is needed. The rationale for this approach is simple: it is impossible to tell without exploration whether a conflict is real or apparent.

3.1.3. Example

Take for example a library system. Two librarians offer different descriptions of the possible states of a book (figure 4). One gives a description based roughly on a book's physical whereabouts, whereas the other gives a description based on how a book can be accessed. There is a conflict between their views of the library.

There appear to be a number of correspondences between the two descriptions. For example, the concepts ON SHELF and AVAILABLE are similar, except that the latter includes books waiting to be shelved: it assumes that unshelved books can still be lent. OUT and LENT are also similar, except that the former includes books being used within the library, while the latter only includes such books if they are from the reserve collection. To make things worse, both could have used the same terms.

3.2. Exploration phase

The aim of the exploration phase is to arrive at a better understanding of the conflict. Additional knowledge is elicited about the conflicting descriptions. This phase involves identifying why the conflict occurred, and hence the type of conflict, the extent of the conflict, and the issues involved. Such information is represented by annotating the descriptions and linking elements together. In particular, links showing correspondences and discrepancies are used.

The exploration phase begins once a conflict has been detected. It focuses on the conflicting parts of the viewpoint descriptions. For example, given the descriptions in figure 4, imagine the analyst is trying to establish when a book is available for loan. The states ON SHELF in figure 4(a), and AVAILABLE in figure 4(b) correspond roughly, but there is conflict, as neither the names, nor the transitions attached to these states match. In this case we begin the exploration with these two diagrams and an indication that the conflict is between ON SHELF and AVAILABLE.

The result of the exploration phase is a map of the conflict. The original disparity between viewpoint descriptions is reduced to a list of items in the descriptions which correspond, and items which do not. Together, these comprise the *components* of the conflict. The exploration phase also elicits underlying issues.

3.2.1. Establishing Correspondences

The first problem is to establish common ground between the descriptions to provide participants with a basis for communication. To determine the extent of the conflict, statements related to the conflicting ones are compared, as a *context* for the conflict. Initially, the context consists of those statements in the original viewpoints which are directly connected to the ones in question. In a graphical notation, these are the arcs and nodes connected to the items in question, and for a chain of inference, immediate antecedents and consequences are used.
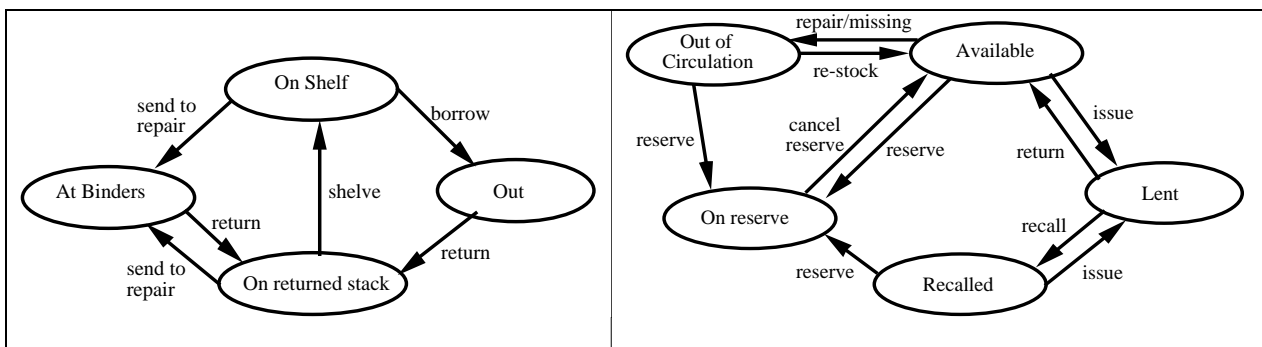

Figure 4: The possible states for a library book: (a) from the perspective of physical whereabouts; and (b): from the perspective of accessibility of a book.

The participants identify correspondences between items in the context. Correspondences may be exact, where the items are identical, or approximate, where the items are similar, but differ in certain details. Many terminological differences will be discovered at this point. The tool support provided in *Synoptic* relies on the user to identify correspondences, although methodologies for recognising terminological mismatches, such as that of Shaw & Gaines [1988], and tools for detecting graph isomorphism could be added.

To illustrate this process, consider the library books example. The arcs attached to the bubbles ON SHELF and AVAILABLE, form the context. There is an approximate correspondence between SEND TO REPAIR and REPAIR/MISSING: the former appears to be included in the latter. This raises the issue of how books going missing are handled in the first diagram, and whether this needs to be represented. The actions BORROW and ISSUE appear to be identical, but note that the return action in one diagram is the inverse of issue, while in the other, it leads to a new state, ON RETURNED STACK. In this case we can assume that the state AVAILABLE in the second diagram is the composition of ON RETURNED STACK and ON SHELF. Other correspondences can similarly be found, and items may be involved in more than one correspondence.

Figure 5 shows the different types of correspondence. For example, ISSUE and BORROW are equivalent (figure 5a). In this case, one is the name of an action described by a librarian, and the other the same action described by a borrower. Both terms are useful, and could be recorded as synonyms. The comparison raises the issue of which term should be used where.

Figure 5b shows a correspondence between a single item in one description and a group in the other. In this case the representations are at different levels of abstraction. In such cases, the correspondences may not be exact, as decomposition will reveals details about a description not considered at a coarser grain. Again, such comparisons yield issues that one description does not address.

Figure 5c shows a correspondence between a group of items in one description and a different group of items in the other. In this case, different types of decomposition have taken place. In this example, both groups are decompositions of "In the Library". The two groups will not necessarily match exactly. For example, the RECALLED state seems to include recalled books both before and after they are returned, and so is not totally captured within the group ON SHELF / ON RETURNED STACK. Alternative decompositions reveal different concerns within the system modelling process.

Finally, an item or group of items in one description might have no correspondence in the other. It may have been omitted, or because the role played by such an item has been filled in other ways (figure 5d).

The result of this stage is a list of correspondences between items in the viewpoints. Note that exact correspondences do not imply identical structure, but indicate where there is agreement, and so restrict the area of conflict. Where a correspondence is partial, there is still conflict to be resolved. In effect, the conflict has been broken down into its components: the initial rough description is replaced with a list of specific disparities between items in the descriptions.

3.2.2. Identifying the Conflict Issues

For a conflict to be resolved constructively, the reasons the parties are in conflict must be ascertained. The apparent conflict might not reflect the underlying issues, as the descriptions being compared might not be based on the same initial assumptions and motivations. In fact, there will be many assumptions, goals, and motivations involved in any description, some very trivial, and only a few will be relevant to the analyst. They are idiosyncratic in that what is obvious to one person may be an important decision to another [Kaplan 1989]. Discussion of *relevant* assumptions must be prompted in some way.

The systems gIBIS [Conklin 1989] and Argnoter [Stefik *et. al.* 1987] use *issues*, which are simply points that the design needs to address. They may take the form of suggested requirements (e.g. "The check-out process should ensure the borrower has not taken too many books"), or questions which need to be resolved (e.g. "How many books is too many?"). However, in these systems issues are elicited unprompted: in gIBIS as a prelude to identifying positions and in Argnoter as supportive arguments for proposals. Our approach is to elicit issues only in response to specific conflicts. Conflict provokes discussion of the issues, as participants raise any issues they feel other party's descriptions neglect. This avoids time wasted discussing issues on which there is already agreement, or which are irrelevant to the current context.

To assist with the elicitation of issues, four types of free-text annotation may be attached to the items in the descriptions, and to the correspondence links between items:

Comments - these are general purpose annotations, which can be attached to any item or group of items in the conflict. A typical use would be to attach to a correspondence between items to suggest a reason for the difference or similarity of items. Example: A comment might be attached to the state ON RETURNED STACK noting "librarian B's description does not include a returned stack".

Assumptions - these allow the user to note where a description appears to make some unstated assumption. They arise when comparisons reveal issues that have been neglected in either description. Example: An assumption may be attached to the comment above, to note that "librarian B's model assumes that books waiting to be shelved can be located for loan".

Issues - these are points that need to be addressed. There are many circumstances under which issues arise, but often comments and assumptions will result in an issue. Example: the assumption above might lead to the issue: "How can books that have been returned but not shelved be traced?".

Justifications - These are added to support a particular viewpoint or proposal. Often these will be added in response to assumptions and comments to provide a rationale for the original item. They will also be added in the next two phases of the process, to relate solution
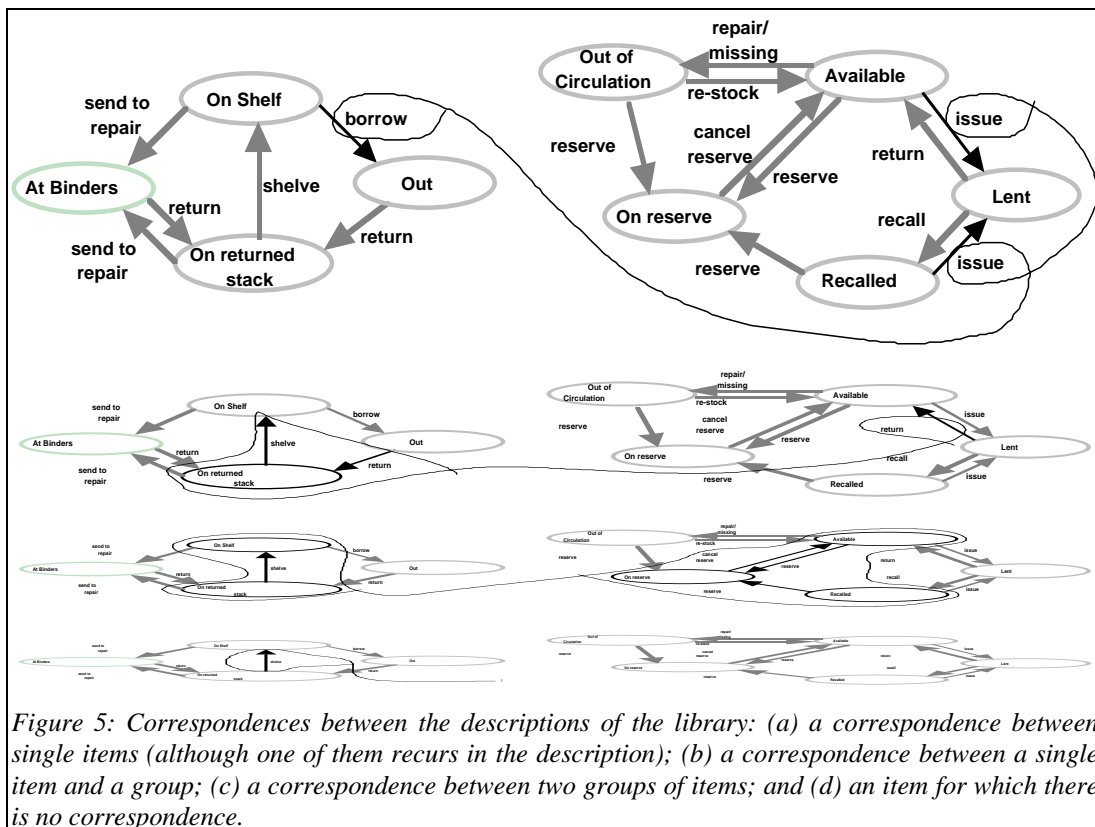

Figure 5: Correspondences between the descriptions of the library: (a) a correspondence between single items (although one of them recurs in the description); (b) a correspondence between a single item and a group; (c) a correspondence between two groups of items; and (d) an item for which there is no correspondence.

components to issues.
Several of the examples in the previous section showed how issues arise during the comparison of descriptions.

3.2.3. Agreeing Resolution Criteria

The final part of the exploration phase is establishment of criteria by which to judge possible resolutions. These represent the participants' goals for the resolution process, and will allow potential resolutions to be judged and compared. Issues represent the key points in the conflict; criteria show how the participants feel about these key points.

In our model, every issue has a related criterion describing desirable outcomes for that issue. For example, the issue might pose a question, and the criteria attached specify what would constitute a satisfactory answer, with reference to the participants' concerns. Effectively we treat issues as points that the original viewpoints left unclear, and the criteria as the clarification of these points. For some issues, participants will define opposing criteria. In this case, both are recorded, and the dispute added to the list of items in conflict.

Criteria can be used by participants to object to an issue. Issues are elicited in response to conflicts, and so will usually be agreed as being valid, if only because they are important enough to disagree over. However, occasionally, a participant will object to an issue as irrelevant. In this case, they can attach a null criteria, effectively stating "this issue can be ignored (in my opinion)". A null criteria will have either an assumption or a justification attached, explaining why. For example, a participant may object to the issue "There must be a way of locating unshelved books" because books can be assumed to be unavailable until shelved.

3.3. Generative Phase

The result of the exploration phase is a "map" of the conflict, which can be used to guide the search for possible resolutions. The second phase generates these resolutions, and is essentially a design process. The aim is to propose solutions which overcome the limitations of the original viewpoints, and respond to the issues identified in the exploration phase. At this stage, the proposals are not evaluated. This prevents the creative process being stifled by pragmatic considerations [Stefik *et. al.* 1987]. The proposals might be generated in a variety of ways, from directly combining elements of existing viewpoints to techniques such as lateral thinking and brainstorming.
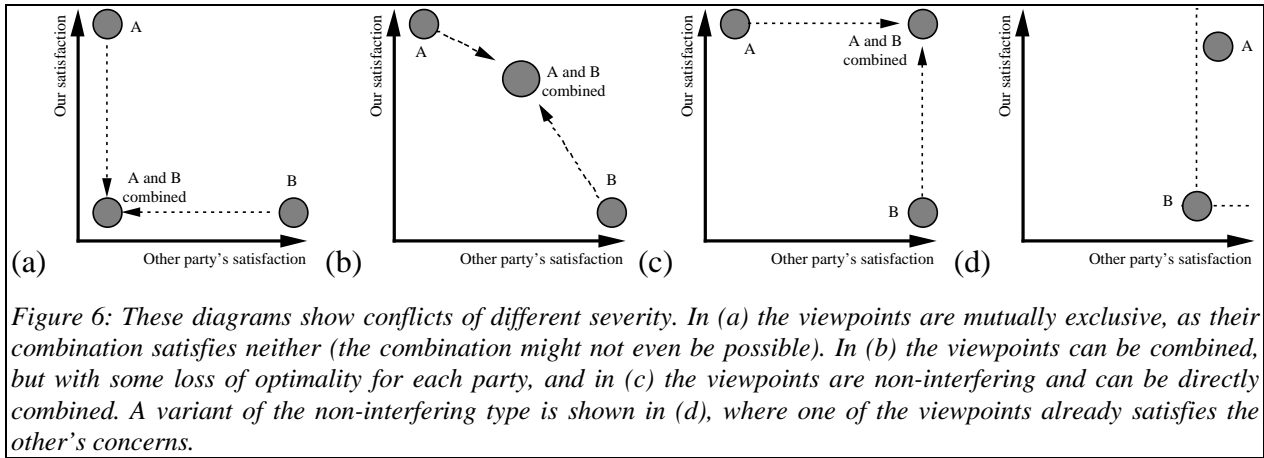
The result of the generative phase is a list of proposals for resolution. These are not intended to be complete resolutions, but might be combined in various ways to arrive at one. It is also possible that some proposals will be incompatible with one another: the evaluation phase will examine how they can be combined.

3.3.1. Types of Conflict

It is useful at this stage to characterise the type of each component of the conflict revealed by the exploration process. This will help to decide what form the generative phase will take, and what a possible resolution might consist of. We can identify three broad categories of conflict that might arise in requirements analysis, as follows:

Conflicting interpretations - descriptions of the current situation or the current requirements do not match, usually because different perspectives interpret things differently. This category corresponds to the category *Beliefs* (or "how things are"), as described by Deutsch [1973].

Conflicting designs - suggestions (or partial designs) for how the system should be do not match. This corresponds roughly to Deutsch's category *Values*, or "How things should be". While a requirements specification would not normally be expected to contain design

Figure 6: These diagrams show conflicts of different severity. In (a) the viewpoints are mutually exclusive, as their combination satisfies neither (the combination might not even be possible). In (b) the viewpoints can be combined, but with some loss of optimality for each party, and in (c) the viewpoints are non-interfering and can be directly combined. A variant of the non-interfering type is shown in (d), where one of the viewpoints already satisfies the other's concerns.

information, participants are likely to express some of their requirements as partial designs, representing their preconceptions of the system.

Conflicting terminologies - the terms in which things are described do not match. This covers the communication problems suggested by Robbins [1989] as being a major cause of conflict.

In addition to these three categories, a scale of severity is used. This ranges from *non-interference* to *mutual exclusion*. The former implies items can be combined directly without compromising either, whilst the latter indicates that each totally negates the other, and only one can be used (figure 6).

Using this schema, conflicts identified as non-interfering can be eliminated from further resolution work, as the direct combination of the two viewpoints provides an instant resolution. Where the two viewpoints provide alternative views or alternative terms, the circumstances under which each should be used still needs to be examined. For the remaining conflict types, there is plenty of scope for the design of novel resolutions which circumvent the conflict, by satisfying the underlying issues in other ways.

Table 1 describes examples of each of the categories and levels of severity. The examples are from the list of specific correspondences and differences discovered in the exploration phase. Some are phrased in a way that suggests possible resolutions; consideration of where these conflicts should appear in the table helped identify potential solutions. Note that these proposals are not exhaustive, and may obscure other possibilities. For example, exploration of this particular conflict revealed that one viewpoint was concerned with the physical whereabouts of a book, while the other is describing accessibility: it may make sense to retain both viewpoints entirely, and record for each book both its physical whereabouts and its loan status.

3.3.2. Generating Resolution Proposals

The model does not prescribe a method for generating resolutions. Consideration of the category and severity of the conflict components, as described in the previous section, provides an initial method. Beyond this, a range of design methods documented elsewhere (e.g. Finkelstein & Finkelstein [1983]) might be employed, depending on the components of the conflict, and the form of resolution required.

The categorisation of conflicts helps to determine what form a resolution should take. For example, conflicts in terminology, once detected, can be resolved by prompting for distinguishing terms. Each such suggestion is a possible resolution, to be evaluated in the same way as other resolutions. Proposals may include circumstances under which a particular term might be favoured. In many cases, negotiating terminological differences is a waste of time, and

participants should agree to differ. It may be sufficient just for the participants to be aware of such conflicts.

Conflicting interpretations are slightly harder to resolve. Sometimes these will be based on incorrect information, which can be investigated. More often, they will arise from alternative ways of looking at things. Both interpretations might be useful, and proposals might attempt to combine them, or which suggest circumstances under which one or other might be used. Proposals might also recommend that one interpretation should be discarded, in which case the issues raised by the discarded description need to be satisfied in other ways.

Conflicting designs involve a higher level of uncertainty. Often the conflict will be the result of conflicts not tackled at earlier stages, and the issues arising out of conflicting designs will indicate the concerns that lead to them. As the exploration stage has broken down the original conflict into its specific components, the designs can be examined more closely. Possible resolutions include combining the requirements underlying the designs, adapting one design to incorporate issues raised by another, or creating a new design which addresses the issues in new ways.

The result of this phase is a set of proposals for resolution. These will vary from the very specific (such as a particular change to a description), to entirely new viewpoints. Where a proposal is only applicable under certain conditions, these are described as part of the proposal. Note that the original viewpoints could be considered as possible resolutions: one or other could be accepted unaltered, if it turns out to be a satisfactory resolution. In addition, some proposals could be combined to produce a more complete resolution, while others might need to be dismantled. At this stage, the proposals have not been evaluated or compared in any way.

3.4. Evaluation phase

The final phase, evaluation, consists of taking the proposals for resolutions and relating them both to the map of the conflict generated in the exploration phase, and to each other. The aim is

	Conflicting Interpretations	Conflicting Designs	Conflicting Terminology
non-interfering	Either interpretation can be used without affecting the other (need to find out which to use when). *Example: the possibility of books going missing has been omitted from the first viewpoint, and could be added directly if necessary.*	The design can be directly combined without compromising either. *Example: The recall facility, which is assumed to be a design suggestion, could be added directly to the first viewpoint*	Different terms have been used for the same concept (need to find out which to use when). *Example: "borrow" and "issue" apply to the same action. A borrower is more likley to use the former term, and a librarian the latter.*
partially interfering	Interpretations are not wholly consistent: if both are to be used, some resolution is required. *Example: The "shelve" action is not wholly consistent with the second viewpoint -"available" does not quite correspond to "on shelf".*	Designs can be combined but interfere, and the direct combination may not be the ideal resolution. *Example: A reserve collection could be added to the first viewpoint by splitting the "on shelf" state to indicate the type of shelf.*	The same labels have been used for similar concepts. The differences need to be resolved. *Example: "Out of circulation" and "At binders". The latter is more specific, and implies that these books will eventually return.*
mutually exclusive	Interpretations totally contradict one another, and cannot be used in conjunction. *Example: There is no "return" action for recalled books in the second viewpoint, contradicting the notion of a returned book stack.*	Designs are completely incompatible, or tend to negate one another when combined.	The same labels have been used for different concepts, and some distinguishing terms are needed. *Example: The "return" from "at binders" is indistinguishable from the "return" from "lent". These might be completely different actions.*

Table 1: Different types and severity of conflict, and for each a description of the kind of situation covered, and an example from the library books conflict.

to find a proposal that best resolves the issues involved in the conflict. The approach is similar to that of the exploratory phase, and consists of linking items together and eliciting extra information to supplement the links.

The evaluation phase begins once a sufficient number of proposals has been generated. In fact, there is no distinct end to the generative phase. When participants feel that a good range of proposals has been generated, the evaluation phase can be initiated. The generative phase and the evaluative phase are kept deliberately separate, to prevent premature evaluation of the proposals from stifling generation of new suggestions. However, it is possible that the evaluation phase will also lead to new proposals, causing a cycling of these two phases.

3.4.1. Relating Proposals to Issues

The first task is to relate the suggested resolutions to the issues underlying the conflict. This may be done by taking a proposal, and selecting the issues that it satisfies, or by taking an issue, and deciding which proposals would satisfy it. Satisfaction of issues is measured using the criteria attached to them.

Links between proposals and issues vary in the extent to which the proposal satisfies the criteria. The relationship may be either positive or negative, depending on whether the proposal contributes to the satisfaction of the issue, or frustrates the issue. Unfortunately, the complex relationship between proposals and issues cannot be satisfactorily expressed using a simple numeric scale. Instead, a qualitative scale of five values is used: fully satisfies; partially satisfies; no effect; partially frustrates; and totally frustrates. The system attaches the value "no effect" by default. The values will later be used to compare the proposals which contribute towards each issue. If the satisfaction or frustration is partial, an explanatory note is attached.

3.4.2. Relating Proposals to One Another

Individual proposals may interact in interesting ways. Some might be combined to produce a resolution which satisfies more issues than either individually: for example, the suggestion of adding a "missing" state to the first viewpoint, and the suggestion of renaming the arrow from both this state and the "at binders" state to "restock" might be combined to give a more complete solution. For other proposals, combination will negate some of the benefits: for example the suggestion of adding a reserve collection to the first viewpoint is not compatible with the suggestion of maintaining two types of state information, whereabouts and loan status. The range of interactions between proposals is analogous to the possible interactions between conflicting items (see figure 6), which were evaluated using a scale of severity.

Where two or more proposals can be combined, the combination is recorded as a new proposal. In creating the combination, the way in which the combination satisfies the issues may need to be reconsidered. In most cases the combination will satisfy all the issues that the individual proposals satisfied. However, this is not always the case, and in particular, it is not clear how proposals with differing strength links with an issue might be combined. This information need to be elicited from the participants. Additionally, the combination might only be possible under certain circumstances, which need to be recorded as conditions for the new combined proposal.

3.4.3. Choosing a Resolution

Once the proposals have been linked to the issues and to each other, the only remaining problem is to select the best proposal or combination of proposals as a final resolution. In many cases, an agreed resolution will have emerged during the process, making much of the evaluation phase redundant. In cases where there is no obvious resolution, the proposals need to be compared. If there is a proposal (or combination) which satisfies all the issues, then this is a likely candidate.

If any participants are unhappy with such a resolution, their reasons need to be elicited: these are likely to indicate issues that were missed in the exploration phase.

To a certain extent, if there is still no clear resolution at this stage, this can be seen as a failure of the negotiation process. The aim of the entire process is to explore the conflict and educate participants about each other's viewpoint: if this is successful, a resolution should emerge from the process, or the conflict should disappear. In the last resort, the participants might either agree some decision making procedure, or agree to leave the conflict unresolved. This will depend on the perceived importance of the conflict.

The chosen resolution is represented as a new viewpoint which can be used instead of the original conflicting descriptions. The original descriptions are retained as a record of the resolution process. The conflict map is recorded as a rationale for the resolution viewpoint, so that it is available for later re-examination if necessary.

4. SUMMARY

This paper has described a model for integrating conflicting domain descriptions. This forms part of a larger model of requirements engineering based on the representation of multiple viewpoints, as described in Easterbrook [1991]. The model recognises that carefully managed conflict can improve the quality of the requirements specification, and encourages the expression of conflict by allowing participants to describe their viewpoints separately. Expression of conflict needs to be balanced with productive resolution methods, to ensure that conflicts do not become counter-productive. The model was designed with this aim in mind.

The model consists of three phases: exploration of the participants' viewpoints; generation of suggestions for resolving the conflict, and evaluation of these suggestions. During the exploration phase, the conflict is broken down into its components, represented as specific correspondences and differences between items in the viewpoint descriptions. These are annotated with comments describing any assumptions they make and issues they raise. These links and annotations act as a map of the conflict to guide the later stages. Resolution takes the form of designing novel ways of satisfying the issues. In the final phase, the ideas generated are then compared with one another and measured against the issues to determine the level of satisfaction. The proposal or combination of proposals which best satisfies the issues is chosen as a resolution.

The model combines the two most co-operative methods of conflict resolution: education and negotiation. Emphasis is placed on the exploratory phase in which participants learn about other viewpoints by comparing them to their own. Participants are encouraged to compare their viewpoint descriptions with others, and this comparison facilitates the elicitation of additional information, such as hidden assumptions. In fact, the resolution is not necessarily the most important product of the negotiation process – the extra information elicited during the process, and the participants new understanding of one another's viewpoints may be far more valuable.

The entire process is highly interactive, and acts to structure the elicitation of additional information concerning the conflict. Rather than imposing a strict methodology on the participants, the various activities can be freely interleaved, so as to support discussion and exploration. Where an agreement is not reached, the arguments and understanding that have been built up aid judicial arbitration.

The model does have limitations. One weakness is that there is no way of ensuring that all relevant viewpoints participate in the conflict resolution. When a difference between particular viewpoints becomes important enough to attempt to resolve, there may be other viewpoints which contain extra information relevant for the resolution. It is not clear how these relevant viewpoints can be detected, especially in the presence of the mismatches in terminology and

style discussed in section 1.1. If other viewpoints conflict with the generated resolution, then the resolution process may have to be repeated.

It is tempting to assume the model provides a general framework for conflict resolution. However, the model was designed specifically for comparing and merging previously elicited viewpoint descriptions. The analyst would carry out most of the work, usually over a period of time, involving the viewpoint originators when necessary. The model is not intended provide a form of meeting support, such as that provided in CoLab [Stefik et. al. 1987], nor is it intended to be a problem exploration tool, such as gIBIS [Conklin 1989], although it shares some features of these systems. It is possible, however, that the model could be adapted for use in these situations.

The support system, *Synoptic*, is a prototype, built to demonstrate the working of the model. The support it provides is often minimal. Its main deficiency is that it requires the user to do much of the work. By incorporating more knowledge about the conflict resolution process, the system would provide much more guidance. For example, by comparing the issues which resolution proposals satisfy, the system might generate the most likely combinations of proposals, and use them to prompt the user for more information. Such a technique was used successfully in the knowledge acquisition system KSS0 [Shaw & Gaines 1987]. Another, major shortcoming of *Synoptic* is that it only allows two viewpoints to be compared at once.

We have described a solution to the problem of comparing viewpoints and resolving conflicts between them. We have demonstrated how an example conflict might be resolved using the model. Additional advantages include the ability to mix representations, and elicitation of additional information. The former is important as it is notoriously difficult to compare knowledge represented in different ways. Synoptic provides a means to explore correspondences between different representations. In using the comparison of existing descriptions as a basis for exploration, the system is able to elicit underlying assumptions which might otherwise have remained hidden. Furthermore, the comparison process draws out the issues that the descriptions address.

5. ACKNOWLEDGEMENTS

6. REFERENCES

Anderson, J. S., and Fickas, S. (1989) *A Proposed Perspective Shift: Viewing Specification Design as a Planning Problem*, Proceedings, Fifth IEEE International Workshop on Software Specification and Design, Pittsburg, Penn.

Axelrod, R. (1984) *The Evolution of Co-operation*, Basic Books Inc, NY.

Blum, B. I. (1991) Integration Issues Elucidated in Large-Scale Information System Development. *Journal of Systems Integration*, Vol 1, Pp. 35-53.

Brown, R. (1988) *Group Processes: Dynamics within and between Groups*, Oxford: Basil Blackwell Ltd.

Conklin, J. (1989) *Design Rationale and Maintainability*, Proceedings of the Twenty-Second Annual IEEE International Conference on System Sciences, Vol 2, Hawaii.

Curtis, B., Krasner, H., and Iscoe, N. (1988) *A Field Study of the Software Design Process for Large Systems*, Communications of the ACM, 31 (11).

De Bono, E. (1985) *Conflicts: A Better Way to Resolve Them*, Penguin Books.

Deutsch, M. (1973) *The Resolution of Conflict*, Yale University Press, New Haven.

Dobson, J. (1993) *The structure of the Requirements Engineering Process and its implications for requirements analysis*. Proceedings of the DRA colloquium on Analysis of Requirements for Software Intensive Systems, 19-20th May 1993, Malvern, UK.

Easterbrook, S. M. (1989) *Distributed Knowledge Acquisition as a Model for Requirements Elicitation*, Proceedings, Third European Workshop on Knowledge Acquisition for Knowledge Based Systems (EKAW-89), Paris.

Easterbrook, S. M. (1991) *Elicitation of Requirements from Multiple Perspectives*, PhD Thesis, Imperial College, University of London.

Easterbrook [1993] *Domain Modelling with Hierarchies of Alternative Viewpoints*. Proceedings, First IEEE International Symposium on Requirements Engineering, San Diego, California, 4-6 January 1993

Easterbrook, S.M., Beck, E.E., Goodlet, J.S., Plowman, L., Sharples, M. and Wood, C.C. (1993) A Survey of Empirical Studies of Conflict. In S. M. Easterbrook (ed) *CSCW: Co-operation or Conflict?* London: Springer-Verlag.

Eden, C. (1989) Using Cognitive Mapping for Strategic Options Development and Analysis (SODA). In J. Rosenhead (ed) *Rational Analysis for a Problematic World: Problem Structuring Methods for Complexity, Uncertainty and Conflict*. Chichester: J. Wiley.

Feather, M. S. (1989) *Constructing Specifications by Combining Parallel Elaborations*, IEEE Transactions on Software Engineering, 15 (2), Feb 1989, p198-208.

Finkelstein, A. C. W., Finkelstein, L. and Maibaum, T. S. E. (1990) *Engineering-In-The-Large: Software Engineering and Instrumentation*, Proceedings, UK IT '90, pp 1-8, Peter Peregrinus.

Finkelstein, L., and Finkelstein, A. C. W. (1983) *Review of Design Methodology*, IEE Proceedings, 130 (4), June 1983.

Fisher, R., and Ury, W. (1981) *Getting to Yes: Negotiating Agreement Without Giving in*, Hutchinson.

Huhns, M. N. (ed) (1987) *Distributed Artificial Intelligence*, Morgan Kaufmann Publishers Inc, Los Altos CA..

Kaplan, S. M. (1989) *COED: Conversation-Oriented Software Environments*, Draft Report, University of Illinois at Urbana-Champaign.

Keeney, R. L., and Raiffa, H. (1976) *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, J. Wiley & Sons, NY.

Lehman, M. M. (1990) *Uncertainty in Computer Application,* Technical Correspondence, Communications of the ACM, 33 (5), May 1990.

Lowe, D. G. (1986) *SYNVIEW: The design of a system for co-operative structuring of information*, Proceedings, Conference on Computer-Supported Co-operative Work, Austin, Texas, p376-385.

Luce, D. L., and Raiffa, H. (1957) *Games and Decisions: Introduction and Critical Survey*, J. Wiley & Sons, NY.

Nuseibeh, B., and Finkelstein, A. C. W. (1992) ViewPoints: A vehicle for Method and Tool Integration. Proceedings of the IEEE International Workshop on Computer-Aided Software Engineering (CASE-92), Montreal, Canada, 6-10th July 1992.

Patchen, M. (1970) *Models of Co-operation and Conflict: A Critical Review*, Journal of Conflict Resolution, 14, (3), Sept 1970.

Rapoport, A., (ed) (1974) *Game Theory as a Theory of Conflict Resolution*, D. Reidel Publ. Co., Dordrecht, Holland.

Robbins, S. P. (1974) *Managing Organizational Conflict: A Non-traditional Approach*, Prentice Hall, NJ.

Robbins, S. P. (1989) *Organizational Behaviour: Concepts, Controversies, and Applications*, (fourth edition) Prentice Hall, NJ.

Robinson, W. N. (1990) *Negotiation Behaviour During Multiple Agent Specification: A Need for Automated Conflict Resolution*, To appear, ICSE-90.

Rosenschein, J. S. (1985) *Rational Interaction: Co-operation Among Intelligent Agents*, Ph.d. Thesis, Report No STAN-CS-85-1081, Dept of Computer Science, Stanford University, Stanford, CA..

Schuler, D. (1988) *AI and Hypertext in Support of Negotiation*, in Bernstein, M., (ed) (1988) Proceedings, AAAI-88 Workshop on AI and Hypertext: Issues and Directions .

Scott, B. (1988) *Negotiating: Constructive and Competitive Negotiations*, Paradigm Publishing, London.

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

Shaw, M. L. G., and Woodward, J. B. (1989) *Mental Models in the Knowledge Acquisition Process*, Proceedings, Fourth Knowledge Acquisition For Knowledge-Based Systems Workshop, Banff, October 1989.

Stefik, M., Foster, G., Bobrow, D. G., Kahn, K., Lanning, S., and Suchman, L. (1987) *Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings*, Communications of the ACM 30 (1).

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

Thomas, K. (1976) *Conflict and Conflict Management*, in Dunnette (ed), Handbook of Industrial and Organizational Psychology, Rand McNally College Publ. Co.

Wastell, D. G. (1993) The Social Dynamics of Systems Development: Conflict, Change and Organisational Politics. In S. M. Easterbrook (ed) *CSCW: Cooperation or Conflict?* London: Springer-Verlag.