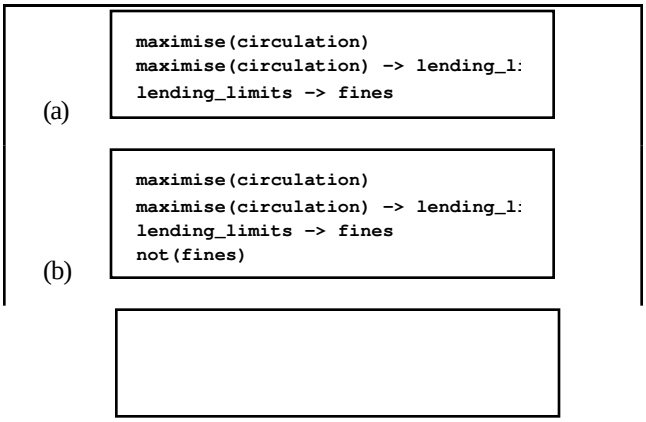
concerns of requirements engineering and knowledge acquisition. Current research in the latter focuses on models of problem solving behaviour, with the goal of implementing systems that demonstrate such behaviour. Emphasis is placed on imitation of an expert's performance. No attempt is made to model the social and organisational setting in which the behaviour is embedded, as is consistent with a purely cognitive stance. On the other hand, much of software engineering deals with systems that support human activities, and hence an understanding of the social and organisation setting is crucial.

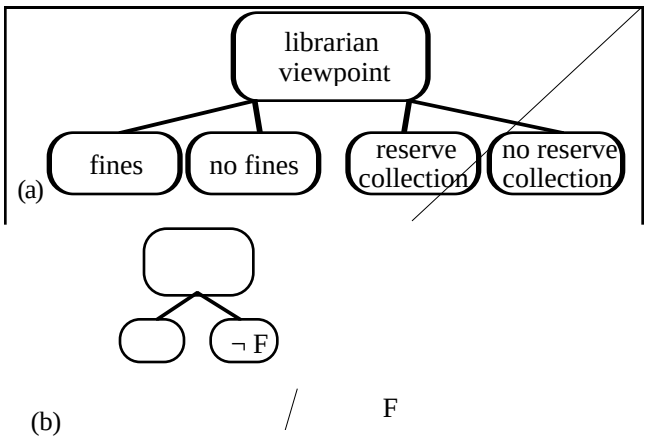
We take a 'socio-cognitive' stance, by which we mean we are interested in the interaction of cognitive and social activities, including issues of shared understanding, and the relationship of mental representations with their social and cultural settings. Instead of modelling a single problem solving agent, we are modelling an organisation. Knowledge needs to be elicited from many different sources, and hence we need to deal with many different viewpoints, and the inevitable conflicts between them. In effect, our domain model will encompass a number of different conceptual models, representing different viewpoints and different roles within an organisation.

2 Related Work

The need to deal with multiple views is central to requirements engineering, and a number of approaches


maximise(circulation)
maximise(circulation) -> lending_l
lending_limits -> fines
(a)
maximise(circulation)
maximise(circulation) -> lending_l
lending_limits -> fines
not(fines)
(b)
(c)


Librarian
Viewpoint
Description


librarian viewpoint
fines
no fines
reserve collection
no reserve collection
(a)

¬ F

(b)

F

no

Inheritance Rules for New Descendants

1) If no previous descendants exist, the usual algorithm (figure 2) is used (a, b).

2) If the item is inconsistent with the viewpoint, then it follows that it is inconsistent with all descendants. In this case, two new descendants, **X** and **Y** are created. **X** will contain the new item, while **Y** represents the status quo. Any previous descendants become descendants of **Y** (c).

3) If the item is consistent with the viewpoint, it might be inconsistent with some existing descendants. For each family, test whether the new item is consistent with each descendant. The following situations are possible:

i) The new item is consistent for all existing descendants. In this case it can be added directly to the original viewpoint (d).

ii) The new item is inconsistent with all existing descendants. In this case rule 2 above applies (c)

iii) The new item is consistent with some descendants and not others. If only one descendant in each pair is inconsistent with the new item, it is placed in the alternative to this descendant (e, f). Otherwise, two new descendants, **X** and **Y** are created, as in rule 2. Pairs that are consistent with the new item become descendants of **X**

transition diagrams. The predicate calculus is supported with a simplified set of rules for detecting conflicts, by generating contradictions through the application of rules such as modus ponens. The graphical representations include validity constraints in their conflict detection rules. For example, in a state transition diagram, a state with two identically labelled transitions from it is treated as a conflict.

6 Discussion

We presented an approach to domain modelling which facilitates the identification and elaboration of viewpoints, and introduces a method of representing conflicting knowledge explicitly, using hierarchies of viewpoints. Each viewpoint contains a description in some suitable representation, and has a unique originator. Each piece of knowledge exists within the context of a viewpoint, and this context provides extra information about the reliability and applicability of the knowledge. *Analyser* currently exists as a prototype implementation.

6.1 Remaining Problems

One problem we have not addressed is how to recognise and handle the use of different terminology. This is a difficult problem when combining contributions from many people [12]. There are, however, some mitigating factors in *Analyser*. For instance, we assume that to a certain extent participants will recognise instances of mismatching terminology, either during translation into a structured representation, or when the viewpoints are presented back to them. Other features could be added to ease the problem, for example, allowing viewpoints to define synonyms. However, these techniques do not constitute a satisfactory solution.

The problem of differing terminologies raises another question. Interpretation of natural language utterances into formal or semi-formal representations involves the formulation of a suitable ontology. Different viewpoints will use different terms to build their description, and there might not be a simple correspondence between the sets of terms. Certainly there is unlikely to be any pre-existing common ontology. However, the viewpoints must use the same terms, for comparisons, and to allow communication between participants. In fact, there does not need to be a shared ontology, as long as correspondences between terms can be found. The conflict resolution model described in [3] addresses these problems in more detail.

Finally, we have glossed over the relationship between inconsistency and conflict. Conflicts are detected if the rules of a representation scheme are broken, or an inconsistency is generated. However, there are conflicts which might not surface in this way. For example, *Analyser* is unable to detect conflicts between a person's goals unless they generate contradictions. Detection of conflict is a difficult problem, and it is likely that a collection of heuristics is needed. We have not attempted to develop such heuristics.

References

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

[2] Easterbrook, S. M. (1991a) *Elicitation of Requirements from Multiple Perspectives*. PhD Thesis, University of London.

[3] Easterbrook, S. M. (1991b) Resolving Conflicts Between Domain Descriptions with Computer-Supported Negotiation. *Knowledge Acquisition: An International Journal*, Vol 3, pp 255-289.

[4] Feather, M. S. (1987) The Evolution of Composite System Specifications. *Proceedings, Fourth IEEE International Workshop on Software Specification and 19on.*
198on.
dP 9 88V v i P w p o i n Or i e s e n t e n