Requirements Engineering for Cooperative Systems

Ian Sommerville and Tom Rodden

Centre for Research in CSCW

Research report : CSCW/1/1994

Computing Department, Lancaster University, LANCASTER, LA1 4YR, UK.
Phone: +44-524-593041; Fax: +44-524-593608; E-Mail: cscw-info@comp.lancs.ac.uk

Requirements engineering for cooperative systems

Ian Sommerville and Tom Rodden,
Computing Department,
Lancaster University,
LANCASTER LA1 4YR, UK.
E-mail: is@comp.lancs.ac.uk

Abstract

This paper addresses the problem of 'production-quality' CSCW software development where software is developed from an agreed statement of the system requirements. In particular, we are concerned with ways in which requirements specifications for CSCW systems can be developed and with the integration of ethnography into traditional specification methods. Existing approaches to requirements engineering are briefly described as are our experiences of using ethnographic studies in systems requirements analysis.

We suggest that existing requirements analysis methods and ethnography must both evolve to accommodate the strengths of the other approach to produce an effective and complete method of deriving cooperative system requirements. An investigation of the changes required to notations for system description and analysis methods is a long-term research goal. However, we suggest shorter-term results can be obtained by using integrated tools for ethnography and requirements capture.

Introduction

By and large, commercial CSCW products such as Microsoft Windows for work groups or Lotus Notes have not been based on an analysis of a work setting which revealed the need for cooperative systems support. Rather, they have been derived by adding cooperative features to existing personal systems or by providing neutral information exchange mechanisms. This approach is an understandable first step to CSCW systems development which can provide simple, low-level cooperative support.

Systems which are required to closely support work and which incorporate knowledge of the work itself need a systematic approach to understanding end-user and organisational requirements for CSCW. Simply adding some technical support for more than one user to existing products without an understanding of the work setting where these products will be used will result in products with severely limited applicability.

Furthermore, a disciplined and professional approach to CSCW systems development will be required to produce systems which are efficient, reliable, portable and maintainable. The software engineering approach to development involves a number of identifiable activities:

1. Requirements analysis and specification
   This involves understanding an existing work setting and specifying the required functions and properties of the supporting CSCW system in sufficient detail that these requirements can form the basis for a contract for system development.
2. System design and development
   A system must be designed and implemented in such a way that it meets its specification and can be maintained over a long lifetime. This requires the use of standards, well-documented designs, structured programming and extensive system documentation.
3. System validation against the specification and the users' needs
   The system must be validated against its specification to demonstrate that the

contract to develop the system has been fulfilled. It must also be checked against the users' requirements to demonstrate that it provides effective user support.

The specify-design-build process model is almost universal for large-scale software projects. A specification of the software is defined and a system procurer (who is responsible for developing the specification) lets the contract to develop the software to a system developer (who is responsible for designing, implementing and validating the software). Where software is developed as a product, the marketing department of the company acts as the system procurer; for bespoke software systems, the system procurer and the system developer are often separate organisations.

This model of software development was derived from the general engineering procurement model and is deeply embedded in industrial culture. Its is used because it separates the notion of specification from 'manufacture' and hence allows these activities to be carried out in different organisations. The development of a clear specification is central as it may be the basis of a legal document setting out what work will be paid for by the procurer.

The existence of a complete specification means that competitive tendering for the system design and implementation is possible. Because of the size of large systems, they can only be developed in a reasonable time by implementing sub-systems in parallel which might be developed by different contractors or by separate teams. The separation of design from implementation simplifies parallel development. The separation into stages with a definite end-point simplifies project management. For all of these reasons, this model or some variant of the model will continue for the foreseeable future as the principal process model for non-trivial software systems.

However, difficulties with this approach are well-known. Developing a specification by relatively abstract analysis and freezing this at an early stage in the process means that it often does not meet the real needs of system users. To address these difficulties, techniques such as participative and user-centred design (Greenbaum and Kyng 1991) (Norman and Draper 1986) which involve end-users in the development process have been developed. These are undoubtedly effective in some situations and result in systems which are often more attuned to user needs.

User-centred approaches are most effective when the system can be developed by a relatively small team of users and developers. However, they suffer from a number of fundamental limitations which means that they do not scale up well to the development of large, long-lifetime systems:

1. They rely on evolutionary prototyping to develop the software. This works very effectively for the first version of the system but this approach tends to result in relatively unstructured systems which are difficult to understand, often unreliable and expensive to maintain.
2. There is never a complete system requirements specification to act as a basis for system procurement. This means that it is very difficult to draw up a contract for the re-implementation of the prototype.
3. There are problems in a user-centred approach in understanding system cooperation. Any one user may not be able to see the 'big picture' of how the different users of the system interact with the system and with each other and how they must support the organisation.

A current research challenge is to develop a model of the software process which allows for the use of user-centred methods during a large systems development project. At the moment, these methods are sometimes used to support the development of a system prototype as part of the requirements analysis phase. However, the additional cost they impose has meant that their use has been relatively limited. We do not wish to suggest that we are hostile to the notion of participative and user-centred design. Rather, we recognise that this approach has limitations and that it is not straightforward to apply this method in the development of large-scale, CSCW systems.

Proponents of user-centred approaches recognise this difficulty and suggest that it might be resolved by modifying the way in which software is procured (Grudin 1991).

Grudin suggests that software procurement should be service-oriented rather than product-oriented i.e. you buy a service which you trust rather than write detailed product specifications. Whilst this is perhaps the best solution from a technical perspective for some classes of interactive system, we believe that it is somewhat naive. It would require immense organisational change to bring about and would require the notion of a contract between procurer and developer to be redefined. Realistically, this is unlikely to happen soon and the vast majority of software will continue to be developed according to the above software process model for the foreseeable future.

We are concerned with the integration of CSCW and professional software engineering. We recognise that conventional approaches to requirements engineering are inadequate for CSCW yet recognise also that 'human-centred' approaches do not meet the industrial need for a clear system specification. An objective of our work is to investigate how 'human-centred' analysis methods such as ethnography which recognise explicit and implicit human cooperation may be integrated with more conventional approaches to system requirements engineering.

The work on which this paper is based has been concerned with observations of air traffic controllers (Bentley et al. 1992) and systems designers cooperating to design and develop a complex hardware/software system. This work is not just concerned with understanding the work setting but has the explicit objective of identifying requirements for a software support system.

This paper reports on some of our preliminary conclusions on the advantages and disadvantages of using an ethnographic study as part of a systems requirements engineering process. To place this work in context, we briefly describe structured approaches to requirements engineering and ethnographic work which has been concerned with understanding cooperative work settings. We present our conclusions concerning the strengths and weaknesses of ethnography for understanding system requirements and then suggest a research agenda for integrating ethnography into the technical process of requirements engineering.

Requirements engineering

The most widely used approaches to system requirements engineering are based on building data-centred and process-centred models of the system.

1. Data-centred models are concerned with describing the structure of the data that is processed by the system. The most widely used data-centred model is an entity-relationship model (Chen 1976) which shows the entities in a system, their attributes and their relationships. Other models track the dependencies between data entities and the life history of data in a system.

2. Process-centred models are concerned with describing data processing as a data entity moves from one 'processing station' to another. A 'processing station' may be a program or a person and each 'processing station' is responsible for some data transformation. Thus an approach such as data flow modelling (DeMarco 1978) shows the processing stations involved in a process and names the data which 'flows' from one processing station to another.

Until recently, a process-centred model was seen as central with the data model as subsidiary to it. However, the development of object-oriented analysis (Coad and Yourdon 1990) (Rumbaugh et al. 1991) has reversed this with the objects (data) in the system being identified and specified before the processing steps.

None of the widely-used analysis methods, irrespective of whether they are based on a process-centred or a data-centred approach, explicitly allow for the inclusion of negotiation and cooperation. They do not even recognise or provide notations for denoting user interaction. It is left to the judgement of the requirements engineer to determine how much interaction and cooperation should influence the system requirements.

We are convinced that the failure of many interactive software systems to meet the real needs of their users is at least partially due to the fact that they ignore this essential

dimension of cooperation. It really depends on whether or not the individual requirements engineer implicitly applies a user-centred approach and recognises the importance of cooperation and is sufficiently sympathetic and intuitive to understand the cooperation and reflect this in the system requirements.

Some methods of requirements engineering have adopted the notion of viewpoints (Kotonya and Sommerville 1992) where it is explicitly recognised that requirements for a system derive from different sources who may have quite different perceptions of the system. These 'viewpoint-based' approaches are principally concerned with looking at viewpoints in isolation rather than as cooperating entities. They provide a way of reconciling conflicting needs (which is a key problem in requirements engineering) rather than identifying viewpoint interactions and cooperation. However, proponents of these methods recognise the importance of cooperation (Finkelstein et al. 1992) and these viewpoint-oriented approaches may, in future, include explicit cooperation specifications.

Understanding work using ethnography

Ethnography involves an observer spending an extended period of time (sometimes several years) living in a society and making detailed observations of its practices. Subsequent analysis of these observations reveals information about the structure, organisation and practices of these societies. Key characteristics of ethnography are that (in principle at least) there are no pre-suppositions about the society being studied, there is no list of questions to be answered and the ethnographer does not try and impose his or her value judgements on the practices which are observed.

These techniques have also been applied to an analysis of work where an ethnographer spends several months in a working environment observing and noting practices, cooperation and processes. The rationale for these studies is that actual work practices often differ quite markedly from the 'prescribed practices' set out in company manuals and handbooks. It is usually the case that a 'working division of labour' (Anderson et al. 1989) evolves where a team organises itself to carry out a task irrespective of the job descriptions and job titles defined by the organisation. This working division of labour is not static. It is continually re-negotiated depending on circumstances, resource availability and priorities.

The potential value of ethnography in deriving computer system requirements was first identified in seminal work by Suchman (Suchman 1983; Suchman 1987). Subsequently, further studies concerned with air traffic control (Harper et al. 1991), police database systems (Ackroyd et al. 1992), underground railway control (Heath and Luff 1991) and financial dealing systems (Heath et al. 1993) have confirmed that an ethnographic study can give real insights into working processes which should be taken into account when deriving computer system requirements.

The notion that there is a fixed process or procedure for most tasks which can be automated is, of course, an over-simplification. The existence of this 'working division of labour' rather than the prescribed organisation is one important reason why the requirements for a software system are often such that the system does not meet the real needs of end-users. The system requirements are defined according to documented procedures and standards but don't take into account actual working practices.

Involving prospective end-users of a computer system in the requirements analysis does not solve this problem. We know from work in knowledge acquisition that experts find it very difficult to articulate their expertise. It is equally if not more difficult for end-users to describe the working division of labour which is, in fact, informal and dynamic. In some cases, the actual work practices may be quite contrary to organisational standards and the end-users of the technology will simply not admit that these practices go on.

Requirements engineering with ethnography

Our work using ethnography for requirements analysis has been mostly concerned with two different types of domain:

1. The relatively narrow domain of air traffic control where all of the participants have a common goal and share common representations. These representations act as a focus for cooperation and are used to mediate and coordinate the work involved.
2. The broader domain of systems design. Although in principle the participants share a common goal (the production of a working system), in practice, the goals of the different teams working on the designs are concerned with their specific part of the system. There is no single shared representation; indeed differences in the interpretation of specialised representations are often a source of misunderstandings.

Curtis *et al* (Curtis et al. 1988) identified the lack of application domain knowledge as the most significant problem in requirements engineering. Ethnography is important in developing this knowledge as the ethnographer's viewpoint is not that of a system engineer. He or she is not limited by trying to fit an application domain into a context of data-centred and process-centred models but deliberately approaches the understanding of the domain with an open mind.

We have found (perhaps unsurprisingly) that the ethnographic studies in the focused application domain have been more fruitful in deriving requirements for a support system based around the shared representation. During our work on design, we have learned much about the design process. However, this information, while identifying cooperation in the process, is not detailed enough to allow us to derive detailed tool support requirements. While it is dangerous to draw general conclusions from our limited experience, our work suggests that, in the near future, ethnographic studies whose objective is requirements analysis are likely to be more cost-effective in these focused domains rather than in more general design work. Hughes *et al.* (Hughes et al. 1994) discuss this and other uses for ethnography in the systems design process.

The application domains which we have studied are discussed elsewhere (Bentley, Rodden et al. 1992). In the course of these studies, we have learned a great deal about using ethnography to understand system requirements. The general advantages of studying an application domain which requires automated support using ethnography are:

1. Explicit identification of cooperation
   An objective of our research was to discover if ethnography was an appropriate technique for identifying subtle, often implicit, cooperation which was central to the functioning of some system. In this respect, the ethnographic work was very successful. The ethnographic studies of air traffic controllers, for example, revealed that cooperation involving 'at a glance' understanding of other controller's displays was an essential part of the process. It showed us that the often-stated requirement for user tailorability (Fischer and Girgensohn 1990) would be a dangerous facility to add to the system as controllers would need to take more time to understand other controller's displays.
2. Identification of process and data variability
   A central assumption which underlies conventional requirements engineering is that both processes and data change relatively slowly (over months or years). In fact, our studies have revealed that processes particularly are rapidly re-configured by participants in these processes to cope with exceptions, other demands on their time or organisational changes which invalidate some of the assumptions on which process models are based. Even the structure of data may change informally as individuals cope with deficiencies in the given data organisation.
3. Identification of organisational influences on system requirements
   Data and process-centred approaches to requirements universally ignore the organisation and organisational culture in which the system is to be delivered. Our ethnographic work has revealed that the organisational structure and culture is a critical factor in making a system work. While we have not yet studied how it might influence system requirements, we are convinced that this is a critical factor.

4. Rationale provision
   A serious defect of all current requirements methods is that they abstract rationale out of the description. System requirements are specified but no record is maintained of why a particular requirement is necessary. This leads to misunderstandings of the requirements and to significant problems when requirements change. Often changes impact the system in unpredictable ways because the person changing the system does not realise why a particular facility has been provided. An ethnographer not only records actual activities but also looks for the underlying reasons why these activities have evolved. This information can and should be associated with specific system requirements. In the short-term, we believe that this provision of rationale is one of the most important contributions that ethnography can make to system specifications.

A positive 'side-effect' of ethnography is that end-users feel more involved in the process. When structured methods are used, end-users may feel left out of the requirements analysis process as the process is designed to treat them as 'processing stations' rather than as intelligent participants in a complex process. Ethnography has the advantage that is involves users and allows them to volunteer system requirements and to identify shortcomings of existing systems.

These advantages have convinced us that ethnographic studies have a role to play in the requirements engineering process. However, the work also revealed several problem areas which must be tackled before ethnography can be used systematically in commercial projects:

1. The ethnographic process
   A model of the process which we used to understand system requirements from ethnographic studies involved developing a prototype system in parallel with the ethnography. Regular debriefing sessions involving the ethnographer and the system developers were held and the prototype system was modified according to information derived from these sessions. This approach, which took place over a period of about 18 months, allowed us to derive useful but incomplete system requirements.

   While a prolonged ethnographic study is possible for very large system development projects where the specification phase may last 2 or 3 years, it is unrealistic for the majority of system specification activities. These must be completed in a relatively short time (typically 2 or 3 months) if the customer's delivery time is to be satisfied. We need to telescope the ethnographic process so that effective results can be delivered quickly. We also need to find ways in which observation-based analysis can be integrated effectively with data and process based analysis with both of these analyses carried out in parallel.

2. The nature of the ethnographic record
   The results of an ethnographic analysis are usually recorded as unstructured text with inevitable overlaps, repetitions, etc. It is difficult for this text to be used by anyone apart from the ethnographer who was involved in the process. Furthermore, the collected data may be in the form of hand-written notes, electronic text, printed documents and diagrams, audio and video tape recordings etc. The heterogeneous nature of this record compounds the problems of finding information.

   There is a need for methods, notations and tools which allow the ethnographic record to be structured and organised so that it can act as an effective supplement to the system requirements document. As discussed in the previous section, the ethnographic record is not just a source of requirements, it can also act as rationale for system requirements. We need to be able to move quickly from a specific statement of a requirement to its associated rationale.

3. Inter-disciplinary communications and education
   Most ethnographic work is currently carried out by anthropologists or sociologists with most requirements analysis carried out by computer scientists or engineers. These disciplines have little in common in that they adopt quite different methodological approaches to a problem, use mutually incomprehensible jargon and

suffer from mutual distrust which, for historical reasons, has developed between 'soft' and 'hard' sciences.

Our work has shown that the barriers between disciplines can be breached (Sommerville et al. 1992) given that each group is motivated to do so. However, introducing 'soft' techniques into 'hard' engineering will require a new approach to the training of requirements engineers.

Cooperative systems requirements engineering

The current procurement model for software systems is deeply embedded in our organisational structures. Because of the dominance of this model, it is inconceivable that existing approaches to requirements engineering will be discarded in favour of 'user-centred design', ethnography or any other 'human-centred' approach. CSCW systems will have to be developed within this framework. The key challenge is not the development of new methods to replace existing methods but the adaptation of existing approaches to requirements engineering to take account of human cooperation and the 'working division of labour'.

The integration of ethnography with requirements engineering can be addressed under three closely related headings:

1. *Methodology*. How can we derive an approach to ethnographic observations which can be integrated into structured methods of requirements engineering and which can be taught to engineers involved in requirements analysis?
2. *Notations*. How can we develop structured, concise and precise notations to describe cooperation and the working division of labour?
3. *Tools*. What tools are required to help organise the ethnographic record and to integrate ethnographic observations with structured requirements?

Before such integration is possible, the disciplines involved must be convinced that the viewpoints and perspectives of the other disciplines are relevant and of importance. Sadly, there is often a vast gulf of misunderstanding between 'pragmatic' software engineers and 'people-oriented' disciplines such as sociology. These latter groups have had little exposure to engineering projects and find it difficult to relate to the design problems which are tackled by the engineers. Similarly, engineers find it difficult to come to terms with the use of very detailed natural language rather than formal notations and the lack of abstraction which characterises social science disciplines.

This reflects the fundamental methodological differences between the disciplines. Ethnographers and other behavioural scientists are trained in analysis and evaluation. They try to avoid making judgements during the analysis process. By contrast, engineers are trained in design and synthesis. Making judgements and formulating abstractions are fundamental aspects of design. These perspectives must be merged if ethnography is to contribute to requirements engineering.

Bridging this gap is not easy. Few organisations are large enough to require a dedicated team of social scientists working as ethnographers in requirements analysis. It seems likely then that ethnography and other 'human-centred' approaches must therefore be undertaken by non-specialists (such as systems engineers and end-users) as part of the requirements process. This will require the expertise of anthropologists and sociologists to be applied to the training of requirements engineers. This training could be simplified if a more systematic ethnographic 'method' for those less-expert users is developed. This should incorporates hints and guidelines about how to perform an ethnographic study.

Methods

In engineering terms, the 'ethnographic method' is very informal and (perhaps simplistically) seems to involve an ethnographer 'hanging around' with the group being studied. This, of course, reflects the underlying precept of ethnography that it should

let understanding of a culture emerge rather than be biased by some methodological pre-conceptions.

This approach contrasts strongly with the increasing trend in software development to make use of 'structured methods' for deriving the system requirements. Typically, a structured method will include a standard set of notations to express the system model, rules defining a 'correct' model, guidelines on good practice and the process to be used to derive the model and a standard set of report formats. In short, the trend is towards prescription in deriving and denoting the analysis.

Structured methods have the advantages that they lead to extensive system documentation expressed in a standard notation and there is usually a clear route from the methodically derived description to implementation. However, the guidance provided by the method is incomplete for many work settings. In particular, support for cooperation or indeed any form of user interaction with the software is a critical omission of current structured methods. The practical result of this is that these methods are rarely applied as defined. In a study which confirmed some of the imperfections of Structured Analysis (a widely-used method), Bansler and Bødker (Bansler and Bødker, 1993) state:

> "In conclusion, our interpretation of what happens in practice is that experienced designers - instead of following the rules and procedures of Structured Analysis - pick and choose among the various formalisms given in the method, adapt them for their own purposes and integrate them into their own design processes."

Applying structured methods may sometimes reduce rather than increase the quality of the requirements specification. Because of omissions and imperfections in the method, the analyst may be guided away from rather than towards key system requirements. Nevertheless the use of these methods has significant benefits to an organisation in that they provide a basis for quality assurance (in the sense that quality means following a defined method) and they provide standardised documentation which is essential for systems which will evolve over many years. In spite of their problems, these methods are unlikely to be discarded in favour of less structured alternatives. Rather than argue for their rejection, we prefer to look at how they can be supplemented with more informal knowledge.

The informal nature of the ethnographic approach means that many software engineers are likely to reject ethnography because of their inherent suspicion of informality (the trend in software engineering since the 1970s has been towards formality). We have written elsewhere of the need to support informality in the software process (Sommerville and Monk 1994) so we do not share this suspicion of development methods which are inherently reliant on human judgement. Nevertheless, we believe that the need for ethnography to be a flexible and opportunistic process does not preclude the incorporation of more focused, deterministic approaches to analysis.

The development of a more structured approach to ethnography which would allow practitioners who were not themselves behavioural scientists to apply an ethnography effectively in requirements engineering. This, of course, means that the 'pure' ethnographic approach must evolve to accommodate more direction and explicit focus so that the time and effort required for analysis is significantly reduced. However, we believe that compromising 'pure' ethnography is worthwhile if that is what is required to make it useful in an engineering context. If we can get 80% of the positive value of 'pure' ethnography with 20% of the effort, the changes are certainly worth making.

We would also like to see this more structured ethnography incorporated into existing structured methods for requirements engineering. Although this must be a long-term rather than a short-term objective, we believe that the approaches to requirements engineering based on viewpoints (Kotonya and Sommerville 1992) can be an effective starting point for the development of such a method. We are currently exploring this possibility where we identify viewpoints in the domain being studied and analyse cooperation across and within viewpoints. Initial results are encouraging but more extensive experience will be required before we can draw any conclusions about the approach.

Notations

An ethnographic record of work practice is inherently unstructured. It consists of observations of work processes made over an extended period of time. Inevitably, there is a significant amount of duplication and the information collected ranges from specific observations of particular activities to anecdotes and 'war stories' told by workers to the ethnographer. Ethnography is almost completely dependent on natural language for expressing knowledge of a work setting. When this ethnographic record is used by non-specialists who are involved in requirements analysis, 3 problems can arise:

1. The readers and writers of the description may not use the same words for the same concept. This can lead to misunderstandings because there may be no shared vocabulary for the work being studied.
2. The sequential nature of the record may mean that similar requirements are expressed in completely different ways. The reader has to find related requirements with the consequent likelihood of error and misunderstanding.
3. The ethnographic record is not partitioned so the inter-relationships between observations can only be discovered by examining all observations.

By contrast to the natural language used to express ethnographic observations, requirements specification typically uses a mixture of structured diagrams expressed using different graphical notations, natural language, and, increasingly, formal mathematics. The trend is away from natural language towards more formal notations because of the difficulties identified above.

Because of the excessive richness and flexibility of natural language descriptions, there is a strong case for developing a more structured notation for expressing ethnographic observations. This would serve the purpose of structuring and indexing the corpus of field material collected during an empirical study.

The development of such a notation is obviously a long-term research goal which might be carried out in conjunction with the development of methodical approaches to ethnography. We are not suggesting, of course, that such a notation should be the sole means of describing cooperation. There are circumstances where only natural language will suffice. Rather, we are suggesting that a notation with natural language annotations will simplify the problem of ethnography becoming acceptable to software engineers.

Tools

We cannot realistically expect short-term results in the development of more structured methods and notations for ethnographic observations. To develop methods and notations, we need more studies on the use of ethnography in requirements engineering to provide basic data for this research. However, we believe that ethnography can make a more immediate contribution to the requirements engineering process, given that we can effectively organise the ethnographic record.

The need for a more structured ethnographic record is an immediate one if the results of an ethnographic study are to be part of the requirements for a CSCW system. The approach which we are currently investigating is to structure the ethnographic record using software tool support.

Of course, there have been previous efforts to provide tool support for the ethnographer. 'The Ethnograph' (Seidel and Clark 1984)is an example of a computer-based tool to support ethnographic record management. However, Davies (Davies 1990)comments on this system and comparable tools:

> "... impose many unwelcome constraints on the researcher and s/he has to significantly alter the methods and techniques of analysis to fit in with a given system"

We believe that forcing ethnographers (or anyone else for that matter) to change their work to fit available tools is unacceptable. More flexible tools are required which can be adapted to a particular way of working and which can present information in a

number of different forms. We are now experimenting with a flexible information management system called the PPIS (Process and Product Information System) which has been developed from our previous work on supporting the design process (Haddley and Sommerville 1990; Twidale et al. 1993). This is a hypertext-based system with a rich vocabulary which allows both entities and relationships of interest to be given types and selective overall views of a system to be produced.

The advantage of our current approach is that the PPIS supports both informal information capture and structured notations (Sommerville et al. 1993). We can therefore seamlessly link information captured during the ethnographic analysis with more structured system representations such as data-flow diagrams or object descriptions.

We have adopted a working practice whereby the ethnographer does not work on site for long, uninterrupted periods but returns regularly to report on the progress of the work. Interim records can be entered into the PPIS. Entering these into the PPIS focuses the ethnographer's attention on the records and often suggests useful structuring which can take place. The records then become immediately available to the software requirements engineers. Thus, ethnography and conventional analysis can be carried out in parallel with the stream of ethnographic information integrated into the specification as it becomes available.

This tool support also helps us to address the problem of prolonged ethnography which may be carried out in parallel with other analyses. In essence, the ethnographic studies generate 'nuggets' of useful information at unpredictable intervals. In conventional ethnography, these are made explicit in an analysis phase where the ethnographic record is analysed after the field studies have been completed. With the PPIS, they can be immediately integrated and linked to system entities and processes.

Conclusions

In systems where there are multiple end-users, cooperating either explicitly or implicitly, we believe that there is always a dynamic and informal working division of labour which is unlikely to conform to formal organisational structures or job descriptions. If a software system is to be successfully used, it must support actual rather than formal work. Ethnography is one way of revealing this working division of labour so, potentially at least, can contribute to the process of deriving requirements for cooperative systems.

Our work has been concerned with investigating how this can actually be achieved and our general conclusions are:

1. Observational methods such as ethnography have an important role in informing systems requirements capture and analysis. However, there is not a clear and simple correspondence between an observational record and a systems requirement document. An important use of the ethnographic record is to serve as rationale for system requirements so some means of linking these documents is required.
2. It is extremely difficult to fit ethnographic observations into current structured methods of requirements analysis as these methods factor out an immense amount of (useful) information which is collected during the ethnographic studies. Because their fundamental principles (and indeed their strengths) are based on limiting the analysts choice, existing analysis methods cannot readily be adapted to incorporate ethnography. We can only use ethnography with existing methods if we devise procedures for systematically annotating the models produced by these methods with information derived from the ethnographic study.
3. Trends in requirements engineering which recognise the importance of multiple perspectives when deriving system requirements are leading this discipline towards CSCW and these new methods are the most likely candidates to serve as a basis for CSCW systems requirements capture. In the longer-term, we foresee the development of more structured approaches to ethnographic analysis incorporating

viewpoints which will reduce the costs of ethnography and simplify its integration with other methods of analysis.

4. Ethnography can be used now with structured methods by providing software tool support which allows all relevant information to be collected and linked. This linking allows the collection of ethnographic data as background material to emerging requirements. The tool must not embed any 'method' but must support a transition from informal to formal information organisation.

Acknowledgements

The work described here has been partially funded by the UK Joint Research Council Initiative in Cognitive Science and HCI and by the European Commission in the ESPRIT projects Proteus and Comic. Thanks are due to our collaborators in the Departments of Computing and Sociology namely Pete Sawyer, Richard Bentley, Michael Twidale, Simon Monk, Dan Shapiro, Dave Randall and Val King.

References

Ackroyd, S., Harper, R., Hughes, J. A. and Shapiro, D. (1992). *Information Technology and Practical Police Work.* Milton Keynes: Open University Press.

Anderson, R. J., Hughes, J. A. and Sharrock, W. W. (1989). *Working for Profit: The Social Organisation of Calculability in an Entrepreneurial Firm*. Aldershot: Avebury.

Bansler, J.P. and Bødker, K. (1993). "A Reappraisal of Structured Analysis: Design in an Organizational Context". *ACM Trans. on Information Systems*, **11** (2): 165-193.

Bentley, R., Rodden, T., Sawyer, P., Sommerville, I., Hughes, J., Randall, D., et al. (1992). *Ethnographically-informed Systems Design for Air Traffic Control*. CSCW'92, Toronto, Canada, 123—29.

Chen, P. (1976). "The entity relationship model - Towards a unified view of data." *ACM Trans on Database Systems* , **1**(1): 9-36.

Coad, P. and Yourdon, E. (1990). *Object-oriented Analysis*. Englewood Cliffs, NJ: Prentice-Hall.

Curtis, B., Krasner, H. and and Iscoe, N. (1988). "A Field Study of the Software Design Process for Large Systems." *Comm ACM* , **31** (11): 1268-87.

Davies, J. R. (1990). "A methodology for the design of computerised qualitative research tools." *Interacting with Computers* , **2**(1): 33-58.

DeMarco, T. (1978). *Structured Analysis and System Specification*. New York: Yourdon Press.

Finkelstein, A., Kramer, J., Nuseibeh, B. and Goedicke, M. (1992). "Viewpoints: A Framework for Integrating Multiple Perspectives in System Development." *Int J of Software Engineering and Knowledge Engineering* , **2**(1): 31-58.

Fischer, G. and Girgensohn, A. (1990). *End-user Modifiability in Design Environments*. CHI'90, Seattle, USA, ACM Press, 183-92.

Greenbaum, J. and Kyng, M. (1991). *Design at Work: Cooperative Design of Computer Systems*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Grudin, J. (1991). "Interactive Systems: Bridging the Gap between Developers and Users." *IEEE Computer* , **24**(4): 59-69.

Haddley, N. and Sommerville, I. (1990). "Integrated Support for Systems Design." *IEE/BCS Software Eng J* , **5**(6): 331-38.

Harper, R., Hughes, J. and Shapiro, D. (1991). Harmonious Working and CSCW:Computer Technology and Air Traffic Control. *Studies in Computer-Supported Cooperative Work*. Amsterdam: Kluwer. 225-34.

Heath, C., Jirotka, M., Luff, P. and Hindmarch, J. (1993). *Unpacking collaboration: the interactional organisation of trading in a city dealing room*. ECSCW'93, Milan, 155-70.

Heath, C. and Luff, P. (1991). *Collaborative Activity and Technological Design: Task coordination in th London Underground control room*. ECSCW'91, Amsterdam, Kluwer, 65-80.

Hughes, J., Rodden, T., King, V. and Andersen, H. (1994). *Moving out from the control room: ethnography in system design*. CSCW'94, Greensborough, North Carolina, To appear.

Kotonya, G. and Sommerville, I. (1992). "Viewpoints for requirements definition." *IEE/BCS Software Eng J* , **7**(6): 375-87.

Norman, D. A. and Draper, S. W. (1986). *User-centered System Design*. Hillsdale, N.J: Lawrence Erlbaum.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W. (1991). *Object-oriented Modeling and Design*. Englewood Cliffs, N.J.: Prentice-Hall.

Seidel, J. V. and Clark, J. A. (1984). "The Ethnograph: a computer program for the analysis of qualitative data." *Qualitative Sociology* , **7**(2): 110-25.

Sommerville, I. and Monk, S. (1994). *Supporting informality in the software process*. 3rd European Workshop on Software Process Technology, Villard-de-Lans, France, Springer, 114-9.

Sommerville, I., Rodden, T., Sawyer, P., Bentley, R. and Twidale, M. (1993). *Integrating ethnography into the requirements engineering process*. RE'93, IEEE International Symposium on Requirements Engineering, San Diego, CA., 165-73.

Sommerville, I., Rodden, T. A., Sawyer, P. and Bentley, R. (1992). *Sociologists can be Surprisingly Useful in Interactive Systems Design*. HCI'92, York, UK, 341-54.

Suchman, L. (1983). "Office procedures as practical action." *ACM Trans on Office Information Systems* , **1**(3): 320-28.

Suchman, L. (1987). *Plans and Situated Actions*. Cambridge: Cambridge University Press.

Twidale, M., Rodden, T. A. and Sommerville, I. (1993). *The Designer's Notepad: Supporting and Understanding Cooperative Design*. ECSCW'93, Milan, 93-108.