Knowledge needs analysis for simultaneously multi-agent real-time systems

This item was submitted to Loughborough University’s Institutional Repository by the/an author.

Additional Information:


Metadata Record: https://dspace.lboro.ac.uk/2134/26859

Publisher: © Philip Scown

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
This item was submitted to Loughborough University as a PhD thesis by the author and is made available in the Institutional Repository (https://dspace.lboro.ac.uk/) under the following Creative Commons Licence conditions.

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/
DECLARATION

This is to certify that I am responsible for the work submitted in this thesis, that the original work is my own except as specified in acknowledgements or in footnotes, and that neither the thesis nor the original work contained therein has been submitted to this or any other institution for a higher degree.

Philip J.A. Scown
KNOWLEDGE NEEDS ANALYSIS FOR SIMULTANEOUSLY MULTI-AGENT REAL-TIME SYSTEMS

by

Philip J.A. Scown

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of

Doctor of Philosophy

of Loughborough University

1997

© by Philip Scown
Knowledge Needs Analysis For Simultaneously Multi-Agent Real-Time Systems

Ph.D. Thesis

Philip J. A. Scown

Synopsis

A set of systems are considered where there are multiple agents simultaneously active within real-time an environment. These systems are termed SMART systems and are found in domains as diverse as office administration, process control and aviation. Such systems place particular demands on agents that are not present in non-SMART systems. Actions may be time constrained in two ways: i) an action appropriate at one time may not be appropriate at another; ii) the time available for a required action may approximate to the time in which an agent is able to perform that action. In order to be able to function in such environments agents, both human and computer, must be aware of time constraints and the actions necessary to ensure that they do not compromise system goals.

To determine the knowledge needs of users, with respect to the particular requirements of SMART systems, a modelling method is required that will make the appropriate system features explicit. A set of criteria are developed to reveal method appropriateness to knowledge needs analysis (KNA) for SMART systems. By applying the criteria a modelling method, Interaction Framework, is identified as providing significant KNA support.

Interaction Framework is applied to a demonstrator scenario taken from the domain of aviation. During the modelling of the system Interaction Framework was adapted to improve its representational power. It is thus shown to be flexible and adaptable to changes from its original description. The resultant analysis of the demonstrator is shown to support KNA and demonstrate a new way of using Interaction Framework. A method is provided that allows system designers to work from the identification of suitable scenarios, through Interaction Framework descriptions on to knowledge needs analysis based on Interaction Framework descriptions and other tools such as programmable user models (PUMs) or task analysis.

Keywords
Interaction Framework, real-time, multi-agent, SMART systems, HCI, knowledge needs analysis, system modelling, CSCW
Table of Contents

Chapter 1. Introduction: What the User in a SMART System Needs to Know
1.1. Background ................................................................. 1
1.2. Describing and Scoping Knowledge Needs Analysis ....................... 5
1.3. Methodology ............................................................... 7
1.3.1. Data collection ........................................................... 9
1.3.2. Representing multi-agent real-time systems ............................ 13
1.3.3. Analysis of systems ..................................................... 15
1.4. Framing the Question ..................................................... 16
1.5. Summary ................................................................. 19
Appendix 1.1: List of Pilot Study Questions ...................................... 20

Chapter 2. Description and definition of the domain of SMART systems
2.1 Introduction .................................................................. 21
Section A: General Description of SMART Systems
2.2. Background ............................................................... 22
2.3. The Nature of Smart Systems ........................................... 26
2.3.1. Multi-agency ............................................................. 26
2.3.2. Real-time ................................................................. 28
2.3.3. Effects of real-time on multi-agent systems ......................... 33
2.4. Conclusion to Section A ................................................ 34
Section B: Examples of Actual SMART Systems
2.5. Studies of Live Systems .................................................. 35
2.5.1. Pilot study of a manufacturing organisation ......................... 35
2.5.2. Systems reported in secondary sources ............................... 38
2.5.3. Anecdotal evidence ..................................................... 44
2.6. Discussion of SMART System Examples .............................. 48
2.6.1. Factors arising .......................................................... 48
2.7. Conclusions ............................................................... 61

Chapter 3. Review of System Modelling Methods
3.1. Introduction ............................................................... 62
3.2. General Review of Modelling Methods ................................... 64
3.2.1. Ethnography ............................................................ 67
3.2.2. Systems based approaches .......................................... 70
3.2.3. Temporal logics and notations ....................................... 72
3.2.4. Cognitive modelling ................................................... 74
3.2.5. Dialogue analysis ...................................................... 82
3.2.6. Task analysis ........................................................... 84
3.3. Comparing Methods within the Domain of HCI ......................... 90
3.3.1. Rating system .......................................................... 91
3.3.2. Results of range analysis ............................................ 93
3.4. Developing Criteria for the Domain of SMART Systems .............. 96
3.4.1. Communication criteria .............................................. 99
3.4.2. Control criteria ........................................................ 100
3.4.3. Goal criteria ............................................................ 101
3.4.4. Knowledge criteria .................................................... 101
3.4.5. Time criteria ........................................................... 102
3.5. Application of the Criteria: Assessing Method Suitability ............... 103
3.5.1. Measuring ethnography against the criteria ......................... 104
3.5.2. Application of criteria to system based approaches ................. 105
3.5.3. Application of criteria to temporal logics: XUAN .................. 107
3.5.4. Application of criteria to cognitive approaches ..................... 109
Chapter 4. Using Interaction Framework for Knowledge Needs Analysis

4.1. Introduction ....................................................................................................... 129
4.2. Selecting A Representative System. ............................................................. 131
4.2.1. Candidate systems ....................................................................................... 132
4.2.2. Outline descriptions of best candidates .................................................. 133
4.2.3. Selecting the demonstrator ....................................................................... 143
4.3. Modelling an Aviation Scenario ..................................................................... 146
4.3.1. External validation of the aviation demonstrator ................................... 146
4.3.2. Plain text description of aviation scenario ............................................... 147
4.3.3. I.F. description of aviation scenarios ....................................................... 150
4.4. Inferring Knowledge Needs from I.F. Descriptions .................................... 184
4.4.1. Explicit KNA gains for the demonstrator ............................................... 186
4.4.2. Explicit KNA gains for demonstrator variations ...................................... 192
4.5. Some Observations On The Psychology Of Time ...................................... 194
4.6. Conclusions ..................................................................................................... 198

Chapter 5. Discussion and Conclusions

5.1. Review ............................................................................................................ 201
5.2. I.F. and the Proposed Modifications ............................................................ 204
5.2.1. Characteristics of I.F. ............................................................................... 204
5.2.2. Modifications to I.F. .................................................................................. 205
5.2.3. I.F. Considered against the five themes. .................................................. 211
5.3. Applying I.F. to KNA ..................................................................................... 215
5.3.1. A method for applying I.F. ....................................................................... 217
5.3.2. The KNA gains from I.F. Application ..................................................... 229
5.3.3. Real-time issues ......................................................................................... 230
5.4. Issues Not Covered ......................................................................................... 233
5.4.1. Psychological issues .................................................................................. 233
5.4.2. Negotiation between agents ..................................................................... 233
5.4.3. Reasons for the failure of individual interaction events ......................... 234
5.5. Future Work ................................................................................................... 236
5.5.1. I.F. and ethnography .................................................................................. 236
5.5.2. The benefits of applying I.F. .................................................................... 237
5.6. Conclusions ................................................................................................... 239
Appendix 5.1 : Worked Example Demonstrating Application of the Method ...... 240

Acknowledgements .............................................................................................. 252

References ............................................................................................................. 253
Chapter 1. Introduction: What the User in a SMART System Needs to Know

1.1. Background

In the 1970s and early to mid 1980s the focus of much HCI research was on the design of single user systems such as word processors, drawing packages or databases (e.g. Dillon & Sweeney, 1988; Monk & Dix, 1987). Such systems are typically not highly dynamic. During the 1980s the importance of dynamic multi-user systems became more apparent and the field of computer supported co-operative work (CSCW) became established. While this is a very broad field, potentially covering a very wide range of systems, a great deal of effort is directed towards the study of systems which support co-operative design work (e.g. Okamura et al, 1994) or co-operative decision making (e.g. Streitz et al, 1994). A significant number of studies consider CSCW from an ethnographic perspective (e.g. Hughes et al 1994) or in a social context (e.g. Schmidt, 1994). While undoubtedly important these approaches to multi-user systems do not deal with some of the important aspects of highly dynamic systems to be found in organisations which have to be responsive. Examples are found in commercial businesses, manufacturing concerns or service and infrastructure organisations such as transportation and power supply. Work looking at real-time\(^1\) aspects of systems often focuses on single user systems or only on the interaction between a single specific user and the system in question (e.g. Decortis et al, 1991). It is the HCI issues of the domain of multi-agent real-time systems that this thesis explores. More specifically the focus is on finding ways to identify what a user needs to know to interact with and within a multi-agent real-time system.

---

\(^1\)Real-time has a number of definitions. A working definition is considered in chapter two where SMART systems are also more fully described.
In many commercial organisations there is a need for complex operational systems. These often require a number of users to carry out a range of tasks simultaneously and in real-time. In a commercial setting it is often the case that product holdings (finished stock) will fluctuate as some items are delivered to warehouses and others are allocated to customer orders. While staff responsible for maintaining inventory levels are carrying out their tasks other staff will be accepting orders from customers and informing them of price and availability. A further group of workers are involved with financial operations such as purchasing, invoicing and credit control. The work of the individuals within the business will usually be supported by information technology.

A similar situation exists within the manufacturing and service sectors. However, here there may be greater reliance on automated systems for sophisticated decision making. Many processes or operations are automated to the extent that very high level decisions may be made by human operators and managers while significant decisions about how a process is executed are delegated to machines. Power and communications networks make use of automation to ensure smooth and continuous flow. In aviation the navigational and engine management systems of commercial and military aircraft are highly automated and are relied on by flight crews and by air traffic control (ATC) to ensure safe and accurate flight. If the support provided by information technology is to be most effective then it is appropriate, as with many other systems, to spend considerable effort on getting human-computer interaction design right.

In thinking about these systems it seems that we are not just considering multi-user systems but systems that have a number of significant agents, some of which are human and some of which are automated. In addition, the situations described above have, at least in part, to deal with real-time events. These real-time events are often caused by environmental factors such as randomly distributed customer enquiries, or variable and difficult to predict demand for
infrastructure services such as power and communications. There are also real-time interrupts from within the system e.g. those that result from equipment failure. In aviation there may be a need to deviate from the original flight plan as a result of mechanical difficulties, medical emergencies, local weather problems or air traffic control traffic management difficulties.

Another common factor is that the agents involved are often simultaneously engaged in their individual tasks, i.e. they are operating in parallel. Many systems exist in which the skills of the agents within them are complementary but very different. In a manufacturing organisation there may be production managers, sales staff, financial specialists. These individuals have their own rôles, each with specific tasks and responsibilities. Yet in this situation of individual specialisation the decisions of one agent may have an effect on the tasks to be performed by another. This requires communication between agents and systems which allow for changes in task status and agent rôle switching. A financial specialist working in the rôle of budgetary planner may have to switch to a different rôle, say that of credit controller, in the light of information about the status of a customer from a sales person. In this example the sales person and financial specialist will have been separately occupied in their respective tasks - but without mutual engagement and only needing asynchronous communication. At some point rôles and tasks will change and they become engaged in a common task requiring synchronous communication.

The point of interest here is that an agent may have a number of rôles and may switch between them as conditions dictate. Each agent may also be perceived as having different rôles by others. For example, sales staff are viewed as such by customers but may be viewed as colleagues by other sales staff. The objectives of each of these rôles will be different and the adoption of a set of objectives may be regarded as defining a rôle; the rôle being supported by participation in some form of dialogue or interaction. In the present scenario the sales
staff have two roles: customer support and member of the department. Making a sale may support both roles by providing the required customer service and maximising department profits.

To design systems where there are multiple agents working simultaneously in a real-time environment it is necessary to know: i) the roles of each agent; ii) the effects of real-time interrupts and constraints; iii) how to support inter-agent communication. In considering the roles of each agent it is appropriate to consider not only the immediate tasks of each agent but how these contribute to the goals of the agent and of the system as a whole. Failure to do so may result in inefficient system design and operation through sub-optimisation. Real-time interrupts may cause agents to switch from one role to another. Such interrupts may be initiated by another agent or may come from the environment. In either case there may be a positive or negative effect on the performance of the agent and/or the system as a whole. For one agent to communicate appropriately with another there must be: motivation, means of communication and an understanding of which agent it is best to communicate with (where a choice exists). In systems where communication is computer mediated and where some of the agents are computer based there is a need to understand these issues so that appropriate HCI design decisions are made. If agents are to communicate effectively then they must have some knowledge of the roles of other agents and the functions that they perform. They must also have knowledge of the time constraints that exist. To support HCI design for multi-agent real-time systems a method is required for identifying the knowledge needs of agents. This thesis identifies one such method and shows how it may be applied.
1.2. Describing and Scoping Knowledge Needs Analysis

A number of techniques exist for determining the knowledge or information needs of computer system users. Some of these are found under the umbrella term Task Analysis; refer to Diaper (Diaper 1989a) for a review showing a range of methods. Other methods also exist which explicitly focus on the information and knowledge needs of users: Programmable User Models, PUMs (Young et al, 1989; Young and Blandford 1994; Blandford and Young 1994 abc, 1995, 1996 ab; ) and the Resources Model (Fields et al 1996 ab; Wright et al 1996).

The term knowledge needs analysis (KNA) is coined here for the process of identifying the knowledge needed by a user for functional interaction with or within a system regardless of the lower level method(s) used. In the context of designing interactive systems the terms information and knowledge are used loosely to mean some processed data that has use, meaning or purpose to the user and is required to support the interaction. While it may be possible to provide distinct definitions for both knowledge and information it is not thought that such a differentiation would prove useful in this context.

That a number of techniques exist for establishing what a user needs to know can be seen from the references given above. The approaches vary according to the user needs perceived by method developers or according to their theoretical inclinations. In the Resources Model (Fields et al 1996 ab; Wright et al 1996) a range of information resources are identified (e.g. goals, plans, action-effect mappings) and are linked to the interaction strategy of the user (e.g. plan following, exploring). In Programmable User Models the knowledge of the user is described in such terms as: device commands, objects, functions, predicates and others. Where specific items of knowledge are missing then interaction problems can be predicted. Thus action can be taken to ensure that all the required items of knowledge are available to the user, reducing the likelihood of operational difficulties. Task analysis
methods may look at how a task or tasks need to be executed to achieve a goal - the focus is on what must be done and what must be known to achieve each step.

What none of these methods addresses are the issues that arise specifically from working in a multi-agent real-time environment. Typically the target systems used when the methods are described are single user systems where time constraints are not an issue. Given that multi-agent real-time systems are widely found, though admittedly not as frequently as word processors, there is a need to identify what users need to know in these more complex situations. The approach to knowledge needs analysis taken here provides a means of describing SMART systems in a way that supports KNA by making multi-agency and real-time issues explicit.
1.3. Methodology

The initial phase of study gathered qualitative data from a range of systems. The purpose was to identify a number of issues with which any formal or semi-formal 2 representation would have to cope. In collecting data it was assumed that a scale of multi-agent real-time systems exists. At the low end we can identify simple office systems such as word processors which, while mostly operating in non-real-time mode do have some real-time components (typically during printing). In the middle of the scale more complex commercial systems, and some process control systems, can be located; while at the top end of the scale we can find aviation, power production and other complex, safety-critical systems. This scale could be said to combine complexity and criticality. It is informal and is not intended to be used for detailed analysis.

In the early stages of the search for a system representation the author was directed towards programmable user models (PUMs) (Young et al, 1989). This approach seemed to be at once both relevant to the purpose of identifying knowledge needs and inadequate for describing the multi-agent, real-time systems of interest. The approach of PUMs, in using the "designer's intended procedure" to determine what the user needs to know to achieve their goals, seemed to have immediate value. PUMs analysis is directed at identifying a range of types of knowledge: prior knowledge, inference, tracking and observation. All of which are issues for users in time constrained multi-agent systems.

The application of PUMs has so far been limited to two-agent non-real-time systems or parts of systems. Where a real-time system was

---

2Here formal representations are considered to be those which are complete and rigorous enough to allow logical reasoning about the system and support the creation and testing of proofs about system performance. Semi-formal representations are taken to be those which allow reasoning about the system but which are not complete or rigorous enough to support complete proofs. Informal representations allow for qualitative descriptions but provide little support for reasoning about the system in a logical/mathematical way.
considered ( Blandford A.E., Young R.M. (1994c and 1996a) the real-time elements were not the subject of the study and so were the target of PUMs analysis. To make PUMs work for multi-agent real-time systems a descriptive technique that would make them amenable to analysis was required. One of the PUMs researchers was able to identify a potentially useful technique: Interaction Framework. An initial examination of Interaction Framework indicated some promise. Both multiple agents and real-time issues were readily describable in Interaction Framework, but it had not been applied to systems for the purposes of identifying knowledge needs of agents within a system.

At this point it appeared that Interaction Framework could provide a multi-agent real-time description of a system that would be analogous to the designer's intended procedure used in the PUMs approach. However, before committing to this approach it would be necessary to extend the search for techniques usable to identify knowledge needs to raise confidence that a better technique did not already exist. PUMs used alone, plus a number of task analysis and other potential KNA tools were identified and compared with Interaction Framework. However, following comparison, it appeared that Interaction Framework provided the most appropriate means of describing multi-agent real-time systems so that they could then be subjected to a PUMs type analysis to identify knowledge needs.

To test this out Interaction Framework was applied to a demonstrator system to show that KNA for multi-agent real-time systems is supported. The subjective-analytical approach adopted was sufficient to demonstrate the value of Interaction Framework for some KNA purposes.

The subjective-analytic approach is the most appropriate method of research for determining the suitability of a description or modelling methods for a class of systems. A positivist / empirical approach would require the framing of one or more hypothesis that could be...
tested through the application of several modelling methods to a number of specific systems. Such an approach would require a significant amount of effort and would not yield results of higher reliability. Indeed, the requirement to find specific situations for study may reduce the reliability of the results in that the existence of experimental and contextual artefacts may make it difficult to generalise from specific, live systems to multi-agent real-time systems in general.

A grounded theory approach to knowledge needs analysis for multi-agent real-time systems would require an existing KNA method for non multi-agent real-time systems that could be applied, possibly with some modification, to the required domain. Some KNA methods were identified within Task Analysis, but it was considered that they would require too much modification to be really useful. It could be argued that Interaction Framework supports a grounded theoretical approach by allowing systems to be described in terms that accord with general systems theory. Inputs and outputs of agents may be equated to inputs and outputs of sub-systems. The neutral view of Interaction Framework in relation to agents means that viewing a system as a collection of agents is isomorphic with a view of a system constructed from sub-systems. In this sense the application of Interaction Framework to multi-agent real-time systems may be said to be a grounded theory approach to the understanding of the knowledge requirements of agents within these systems.

1.3.1. Data collection

The use of highly structured empirical data collection was considered inappropriate at this stage. Such data collection methods assume that the structure of the data to be collected is known. As there were no hypotheses to test statistically sound samples would not be required. Once a method for modelling systems had been established it was then appropriate to collect non-numerical data so that the method and
resultant model of the demonstrator system could be examined. Qualitative data was obtained from three sources:

i) a pilot study of a manufacturing organisation;
ii) a study of secondary sources reporting on multi-agent real-time systems;
iii) anecdotal evidence from the author’s own experience and from the experiences of others working in multi-agent real-time systems.

These data collection methods are further considered below.

1.3.1.1. Pilot study
A number of organisations were approached. After initial exchanges two organisations were studied. However, after only a small amount of data had been collected it became necessary for one organisation to withdraw as a result of internal reorganisation.

The remaining organisation was a tools manufacturer, part of a multinational group, but highly autonomous. The study was confined to the local company - links to the parent group were not considered in detail. Data was collected by structured interview and by informal and unstructured observation. A standard question set was used as a guide to data collection but was not used exclusively. Thus it was possible to explore interesting avenues as they occurred. The answers were recorded on a nine page interview support form, supplemented with additional pages to accommodate diagrams and text. This data was then transcribed to a HyperCard stack, for text, and MacDraw, for diagrams. The HyperCard stack allows sorting either by subject or by question. This supports data review on an intra- or inter-subject basis.

---

3 A list of the questions and question headings is given in appendix 1.1.: List of Pilot Study Questions.
In all there were seven subjects from a number of functions and at a number of levels. The following list gives job titles with brief explanations:

- **Purchasing Manager** (responsible for materials and services purchasing),
- **VDU Operator** (data entry for customer and sales support),
- **Production and Material Control Manager** (factory management & production scheduling),
- **Manufacturing Systems Co-ordinating Manager** (IT manager),
- **Financial Accounting Manager** (budget and credit monitoring and control),
- **Customer Service Manager** (maintain & develop support systems for customers),
- **Domestic Correspondent** (providing sales and customer support, but does not sell).

All subjects were direct or indirect users of the central computing system. They were assumed to be representative of all system users while accepting that a different set of users may have provided different data. The main assumption here is that the differences between different sets of users, selected on the same criteria, would not have produced substantial or significant differences. An additional justification for this assumption is that the pilot study was not to be the sole source of data.

1.3.1.2. Secondary sources
A small number of examples of systems of the target type have been reported in the literature. This has been for the purpose of reporting the findings of some specific research which, though not directed at the target systems, still has some relevance. Examples of systems that have been identified include:

- medicine (e.g. Carr et al, 1992),
- police aviation (e.g. Linde, 1988),
- process control (e.g. Hoc 1989).
1.3.1.3. Anecdotal sources
The author's employment experience, and the experiences of other users, provides anecdotal data. Any description of the target systems produced by a selected modelling method which was to be useful in their analysis, operation and design would have to be consistent with the experiences of user agents in those systems. Data from these sources relates to:

- office automation,
- process control,
- aviation.

These three system domains provide examples which are well spread along the complexity/criticality dimension. The office automation system was part of the administrative support for a financial institution. Real-time demands were often internally generated rather than imposed from outside the system. For example, a senior manager might make demands for information to be provided quickly, perhaps by the end of day. This would present a problem because the information was not within the existing resources of the supplying department and it may or may not have been within their skill to obtain the information easily from some other source. This system, while being real-time, used relatively long intervals for a measure and was not safety critical or even mission critical. As such this system would be placed at the lower end of the scale.

Around the middle of the scale is process control. The particular system was part of a food processing business, the end product of which was canned foods. The process was largely serial but at one point a "cook" would be responsible for a handful of cooking retorts each of which would have a 1.5 ton capacity of canned food. Food would be canned prior to cooking. When a retort was free it would be loaded, brought up to the correct temperature and pressure, cooked for a set period of time and then cooled down and de-pressurised. Each
retort was independent from the others. In effect the cook was dealing with a handful of almost identical retort agents. The retorts needed to be regularly polled by the cook to ensure that i) foods were not overcooked or undercooked resulting in spoilage; ii) pressure didn’t rise too far causing a retort to rupture\textsuperscript{4}. There was no IT support for this task\textsuperscript{5}.

1.3.2. Representing multi-agent real-time systems

A representation is needed that is capable of describing the system set in a way which is powerful enough to support reasoning about multi-agent real-time systems. Any system representation will need to support reasoning about real-time interactions aimed at achieving the objectives of the agents in the system. This requires that any system description must be able to handle temporal aspects of system operation and performance and the performance of agents within the system. A representation which allows logical and arithmetic operations will be required if durations and time points are to be represented and manipulated. For this reason graphical representations have limited usefulness.

A high fidelity model of any specific system might be expected to fully represent both the components of the system and the communications or interactions between them. For a system of any significant functionality this is a huge undertaking. This thesis is not concerned with specific systems except as examples of a general form of system : multi-agent with real-time. The significant system

\textsuperscript{4}Such an event had occurred and had been reported to the author. In that case the 50mm retort fastening bolts had sheared and had flown through the factory just above head height. Fortunately there had been no injuries, but production was affected!

\textsuperscript{5}The lack of IT support for this task is not thought to be significant here. The cooking task provides an example of real-time multi-agent work and as such may inform the analysis of multi-agent real-time systems, as does the non-IT police helicopter example of Linde (1988 ).
component for this study is the agent while the focus of the study is on representing temporal aspects of interactions between agents and what an agent needs to know in such environments in order that they may interact effectively.

In interacting with other agents it is not necessary for each agent to have a deep model of every other agent. What is required is skill with the interaction protocol and awareness of the goals of other agents, i.e. it is not necessary to know: i) how an agent achieves something only that it can achieve what is required and; ii) the necessary communication skills to initiate and facilitate achievement. However, in systems which are critical in some respect (e.g. aviation, process control, share trading) it is important that the human agent is able to recognise when things have gone wrong, know the likely cause(s) and be able to take corrective action. This corrective action may also require interaction with other agents. For agents to interact successfully there are certain things that they must know\(^6\). The main purpose of this thesis is to provide a framework for determining what a user must know for successful interaction to take place. This concept of “what the user needs to know” is borrowed from work on programmable user models, also known as PUMs (Young et al, 1989; Blandford & Young, 1994a). However, in order to be able to determine what a user must know it is necessary to describe the system that the user is interacting with, all be it as a component of the larger system. The representation chosen is based on the Interaction Framework (Blandford et al, 1994) although some modifications have been made to expand it to fit the current purpose.

\(^6\)It is pointed out here that human and computer agents may be said to “know” in different ways, including prior knowledge, inference, tracking and observation previously referred to. However, the focus here is on what must be known rather than how it is known and issues of epistemology.
1.3.3. Analysis of systems

It is easier to reason about the performance of the whole system if it is described in agent-neutral terms, i.e. not considering the perspective of any one agent to be of paramount importance (Blandford et al., 1994). A non-neutral view tends to consider a particular user as interacting with a system and not as an integral system component. This is the perspective of PUMs and the Resources Model. Such approaches, if applied to a multi-agent system, would require a different perspective to be considered for each user agent and would also require some mechanism for ensuring consistency between views. By considering the system as a set of interacting agents, though each may have their own objectives, it is possible to examine how interactions affect whole system performance. This performance contains objectives at a number of levels, some of which will “belong” to individual agents, higher level objectives may apply to some or all agents (Blandford et al, 1994). Thus in considering the system this way it is possible to see how the whole system performs, how that performance satisfies the needs of the agents within it, and how individual agents may affect that performance.
1.4. Framing the Question

The issue for this thesis is to find a method for determining what a user needs to know when interacting with and within a system. However, in a multi-agent system there may be human agents and computer agents involved with critical tasks. Before a general method applicable to users can be identified it seems that it would be better to consider the slightly different question: "What does an agent need to know?". This is a more general question which considers both human agents and computer agents and does not emphasise one over the other. From a generalised approach we can move to the more specific question: "what does the user need to know?". This question may be of use in a number of ways.

By identifying what a user needs to know a computer system designer, who could be described as a computer agent designer, has a better understanding of the design requirements. Overload can be avoided and an appropriate computer interface may be constructed to support user agents in their interactions. Also, where the knowledge required is diverse or complex (Kiss and Pinder, 1986) this may indicate poor task decomposition, further suggesting system redesign.

Computer agents and user agents may both execute vital tasks within a system but as system components they need different consideration by system designers. Computer agents can be designed, and to some degree tested so that their performance is known within quite well defined limits. Computer science can be applied to their design and construction. User agents are different in that they may be trained but the results of that training are not so reliably predictable as for computer components. User performance may vary with experience, temporary disposition, or as the result of individual reaction to a crisis. To some extent users may be "designed" through appropriate training,
they may also be selected according to predefined recruitment criteria such as skill and aptitude. Users are also responsible for determining system goals, they select the tasks to be done and the overall purpose of the system. It is assumed that in any system the agent responsible for the setting of system goals is a human/user agent. The point in raising these differences between user and computer agents is to show that when considering agent based systems there is a need to consider the two main types of agent differently - even if the term agent sometimes seems to be used freely.

There is a further reason to identify what it is that users of and within multi-agent real-time systems need to know and that is that their complexity is often beyond the abilities of a single designer. Smaller, less complex systems may be designed by one person or by a small, closely knit, team. To achieve a task there may be a number of user actions required which can be considered as the designer's intended procedure (Blandford & Young, 1995). For multi-agent real-time systems there may be no single design team responsible for the whole system; components and agent design or selection may be the responsibility of a number of disparate individuals and teams. In this situation there may be no designer's intended procedure for the whole system only fragments of it. Very few people, if any, may have an overall view of the system; even fewer will know it in detail. Some studies of interactions between multiple agents in context have been conducted by ethnographers (e.g. Linde, 1988; Bowers, 1994, Rouncefield et al, 1994). The results of such studies tend to give rich qualitative descriptions which, while useful, are difficult to turn into quantitative and formal descriptions needed by system designers and builders.

It would seem useful to be able to provide a whole system view and describe this quantitatively and formally. A method capable of this could also be applied to critical segments of system activity so that problem areas can be considered economically. This would enable the interactions between agents in multiple agent real-time systems to be
described and manipulated. From that point it should be possible to develop ways of answering the question "What does an agent need to know, and a user agent in particular?". This is distinct from the question "what does a user need to do?" which addresses different problems that are not dealt with here.
1.5. Summary

The target systems for this work are comprised of a number of human or computer agents interacting synchronously or asynchronously in a real-time environment. Human agents will each have their own goals to achieve through interaction with other agents, both human and computer. This is true for all such systems from office automation to aviation. By describing these systems semi-formally we can determine what it is that any agent needs to know in order to be able to interact with other agents to achieve their objectives. The term knowledge needs analysis (KNA) has been coined to describe this type of activity. To determine what a user needs to know to interact as an agent within such a system it is necessary to represent the agent’s position in that system. A number of methods for describing systems are considered. One method, Interaction Framework, is selected on its ability to assist in answering the question “what does a user need to know...”. The method is applied to a demonstrator system and conclusions are drawn for the implications for HCI.

This thesis provides a generic method for determining what user agents need to know when interacting with and within a multi-agent real-time system. This knowledge, gained from a view of the whole system, can be used to inform the requirements analysis, design and maintenance processes with respect to the knowledge needs analysis sub-domain of HCI.
Appendix 1.1: List of Pilot Study Questions

Job description
- Concise description of task
- More detailed description
- Are “time-factors” a problem?
- Importance of real-time aspects?
- Problems resulting from real-time aspects?

Other Users
- Titles and names
- Hierarchy (possible diagram)
- Tasks/roles of other users?
- Problems resulting from conflict between user goals?
- Input from other users?
- Output to others?
- How do you communicate with other users?
- What channels exist?
- Do the channels have enough capacity (bandwidth)?
- Do the channels distort or change communications?
- How formal are communications & does this vary?
- How effective is communication?

System Description
- What is the purpose of the system as a whole?
- How stable is the system - how often does it change?
- Who had influence on the design of the system?
- Who has influence on the change process?
- What inputs are there from outside the organisation?
- What inputs are real-time?
- What outputs are required in real-time?
- What outputs leave the organisation?

Diagram of whole system
Diagram of user’s part of the system

System Performance
- Good points?
- Bad points?
- How could the system be improved?
- What would be the effects of improvements on other users?
- Is there anything the system gives you that you don’t need?

Any other points
Chapter 2. Description and definition of the domain of SMART systems

2.1 Introduction

This chapter has two major sections. Section A provides a textual description of the domain of simultaneously multi-agent real-time (SMART) systems based on that of Scown (Scown, 1992). Examples are given which illustrate the range of systems and the problems sometimes experienced within them. Section B expands on the specific example systems introduced in chapter 1. These examples are used to illustrate the problems that might occur and with which any generic representation must cope. From this discussion of SMART systems, both in general and in particular, it is possible to show that there are a number of common issues which must be dealt with. These issues may have different manifestations from system to system but occur often enough that a generalised system for dealing with them will prove to be worthwhile.
Section A: General Description of SMART Systems

2.2. Background

The research reported here forms part of a programme investigating human-computer interaction issues for simultaneous multi-agent real-time systems (SMART systems). The term "agent" has been chosen on reflection. If the programme was limited to systems which had a number of human users then the work would exclude consideration of complex or intelligent machines which may act autonomously within a larger system. It is also possible that computer agents may interact directly with each other without human mediation. Hence the use of the term "agent" to cover human users of systems and other machines or sub-systems which may affect interaction with, and performance of, a system. A real-time system has elements of time criticality. Typically an event occurs which requires a response within a critical time-frame. The criticality is relative to the time it takes the necessary response to be formed or executed. Thus a given response will only be applicable for a certain period of time after which it becomes inappropriate as the time during which it would have been acceptable has passed. A more detailed discussion of "real-time" is to be found later in this chapter.

An example of a SMART system\(^1\) can be found in air traffic control. In a typical air traffic control situation there will be a flight crew plus the ground based air traffic control officer(s) of the air space(s) that the flight crosses. There may also be some complex automated flight systems. The goals of the various agents overlap but are not identical. The air traffic control officers are responsible for ensuring safe flight paths are defined and that throughput is maintained. They will have different perspectives of the aircraft depending on whether it is leaving, entering or traversing their airspace. The air crew maintain a

\(^1\) An unfortunate acronym as some systems do not seem to live up to this. However, the use of an acronym provides a useful shorthand for referring to large groups of systems.
specified course in a way which is safe and, mostly, comfortable for the passengers. In fly-by-wire aircraft the automated systems will respond to the pilot's input whilst maintaining an appropriate margin of safety. All this is happening simultaneously - the agents do not take turns in interacting except when this is explicitly required to achieve a specific task. From this example and the previous discussion it can be seen that SMART systems are different to conventional office automation: word processing, databases supporting clerical procedures, spreadsheets, etc.

Within office automation (OA) studies there has been a tendency to carry out experimental work on single user applications (Barfield & Robless, 1989; von Benda, 1987; Landau & Jaercke, 1987; Lee & Chao, 1989; Davies et al 1989 and on to Wright et al 1996). Restricting the environment to Office Automation has considerable advantages. Word processors, for example, provide an environment in which the interaction between the computer user and the hardware-software has a limited range of variability. The software may utilise an approximation to WYSIWYG\(^2\) with menus, as in many Macintosh\(^\text{TM}\) applications, control key sequences, as in older versions of WordStar\(^\text{TM}\), or embedded commands, as with troff in Unix\(^\text{TM}\). This limits the range of independent variables the experimenter has to consider and is appropriate for considering issues related to Office Automation. However one should be wary of generalising results of these studies to more complex situations. The office domain has been explored very thoroughly, but that domain is changing. An increasing number of computer supported tasks require any one user to interact with a number of different applications, users and environmental factors in real-time. While those studies that concentrate on single users of single applications are very useful and worthwhile we need to augment these with studies of the more complete system in which users operate.

A further issue, in addition to the emphasis on OA, is the non-real-time way in which these word processing and other tasks are executed. In conducting studies we may ask the user to attempt to articulate their

\(^2\) What-You-See-Is-What-You-Get
cognitive processes during a particular task. This approach assumes
that time does not matter, that if we halt the interaction there will be
no significant negative effect on the task or on the experiment. In
short, interactions are often viewed as if they are a series of
predetermined sequences of actions. Truly dynamic situations are
rarely considered. This may be acceptable for studying
straightforward OA applications but it is not acceptable for many other
areas of activity such as aviation or process control and some complex
administrative and financial systems. The organisations considered in
this study use complex computer systems to support a wide range of
commercial, administrative and manufacturing functions. The reader
is referred to work by Decortis et al (Decortis et al, 1991) for a study
of temporal aspects of complex systems. In that paper the authors
consider a single operator interacting with process control plant. Their
focus is on how an operator reasons about time; what an operator
knows about time and the predictions that may be made about
temporal errors made by operators. These issues are important to our
understanding of what a user needs to know but are not appropriate at
this point when considering system descriptions.

Work on computer supported co-operative work (CSCW) has explored
a variety of environments. Linde (Linde, 1988) examines
communication between helicopter crew members. While this does
not directly illuminate HCI research there are some relevant issues for
the study of multi-user systems. Linde examines the social structure of
crews and how this is reflected in the language used and the actions
executed by the various crew members. This issue is undoubtedly
relevant to SMART systems as a significant question is raised: who can
be said to be in charge when a number of agents are simultaneously
and independently interacting for different low level purposes but in
support of the high level goals of the organisation? CSCW has also
addressed some OA issues, extending the knowledge of that domain.
Particular attention has been paid to support for meetings (e.g. Sasse
& Fentem 1994) and support for co-operative work in general and
design in particular (e.g. Hazemi & Macaulay, 1995; Palanque &
Bastide, 1995; Neuwirth et al, 1994; Pacull et al, 1994). While such
work has relevance to SMART systems it is often narrow in focus - looking only at very specific situations and work problems. This thesis aims to provide a more generalised tool for reasoning about the range of SMART systems in a way that is domain independent.

Historically, for reasons of economy, some organisations have attached real-time systems to existing batch systems (e.g. manufacturers, various banks). This has been done in preference to fundamental analysis and redesign of systems meeting new requirements. The large scale of the necessary investment is reflected in the compromises made: function is often favoured over usability as this reduces short term investment needs. Improved HCI methods for SMART systems might be able to shift the balance of the compromise towards the users and away from the financial controllers by providing an appropriate and cost effective means of taking human factors into account early in the system life cycle.
2.3. The Nature of Smart Systems

In gathering data for the research programme a number of systems were considered and explored in a variety of ways. The systems examined in most detail were used by large commercial concerns. Aviation systems, having a number of distinct differences from standard commercial systems, were also examined. These differences include: the safety critical nature of the systems, the teamwork required and the speed with which some decisions must be made for safety to be maintained.

Interfaces for users of aviation systems have been considered by Rouse (Rouse et al 1987). Although they rightly consider aviation systems as complex systems from their examples it seems that they assume that such systems only have one user ("operator" in their terms). This author suggests that aviation systems, and other complex systems, often serve more than one user in that a number of human agents may be involved. In an aircraft there is likely to be a crew of more than one with consequently complex interaction (Linde 1988). In addition to the crew and the navigation systems that they use there will be ground based users: air traffic control. Decisions made by the air crew will have an effect on air traffic control and vice versa. While a manufacturing situation the real-time data producing activities of one department, e.g. sales, may affect marketing and stock control departments; marketing activities may affect production planning;... It will become increasingly necessary for us to be able to understand such situations and how human-computer interaction should be designed to support them.

2.3.1. Multi-agency

In the illustrative examples, further considered below, there may be many agents in the macro system3. Air crew may comprise several people with a variety of roles; in addition there will be a variety of air

3 The macro system is comprised of all the agents and sub-systems within the boundary of the study.
traffic control officers plus the various aircraft systems. In an
industrial setting plant operators may rely on other agents (operators,
users and/or sub-systems) to execute tasks co-operatively or
subordinately. They will also hope for the correct mechanical
functioning of the various non-agent machines that they use.

In large scale commercial / industrial organisations there will be
functions to support customer service, marketing, finance and others.
Each function may employ a few people or many thousands. Any or all
of these may have access to the organisations computing power
through one or more sub-systems. In one energy supply company the
systems used by five functional areas were connected to a single
customer database system accessed in real-time (see figure 2.1). The
shared database system had been developed to improve both
customer service and organisational efficiency by reducing operational
costs. In a large manufacturing concern (further considered in section
B) separate batch systems had been integrated and a real-time system
included (see figure 2.2 in section B).

Users in both the service and manufacturing organisations investigated
may therefore be required to use several systems and in doing so they
create a larger macro-system. The macro system thus consists of a
variety of users and a variety of computer systems. The importance of
this is that as an identified user makes decisions about how they are
going to interact with the system the appropriateness of that decision
may be affected by the actions of any of the other agents, and may
itself affect those agents. This complexity of interaction may be
further compounded by the need for the system to operate in
real-time.
2.3.2. Real-time

"Real-time" is a difficult term to define. Silver and Silver (Silver & Silver, 1989) provide a glossary which contains the following entry:

"Real-time processing See Transaction-oriented processing

Transaction-oriented processing A system that processes data at the time transactions occur, in real-time; also called on-line data entry."

This seems a little vague. While the immediacy of the processing of data is conveyed there is no sense of the urgency which exists in many situations which can readily be identified as real-time.
From Yourdon (Yourdon 1989) we get:

"A real-time computer system may be defined as one which controls an environment by receiving data, processing them, and returning the results sufficiently quickly to affect the environment at that time"

This is better as it conveys the need for a response to be returned to the environment in a short time-frame.

The author suggests a number of features which, if several are present, indicate the likelihood that a real-time system exists:

1) Some system states which are continuously changing (i.e. not perceived by users as discretely changing).
2) Systems which control machinery.
3) Systems where the interval in which the response needed for proper and stable control is approximately equal to the interval in which the responder is able to respond.
4) States which cannot (easily) be reproduced.
5) Systems utilising feedback to maintain some stable, or otherwise desirable, state.

These points can be considered with reference to the systems examined by the author and others. In piloting an aircraft there are continuous changes in aircraft position and fuel load; in plant control there may be a continuous rise in temperature during a process. The way in which changes occur may affect the users cognitive model of the system, i.e. the user's structural model of the system may change as a result of his or her experiences, as when a piece of equipment fails and needs to be removed from the process. Other changes may be to data relating to the model, as in the case of a process temperature change.

---

4 In other words where the required response window is close in size to the optimum/average response performance time there is little margin for error and so a real-time response may be said to be required. Where the required response window is much larger than the optimum/average response performance time then the responder has time to consider, to plan and may draw on extra resources. This is not a real-time situation.
Some of the illustrative systems control machinery which perform some physical action. It can be said that the word processor also contains a virtual type writer - and a real printer. However, the printer operates in an environment where it is not essential that it operates continuously, a user may put it off-line if they so wish. The same cannot be said in the illustrative cases: the aircraft is continually in motion; plant often has continuous processes which cannot be temporarily halted. This has significant implications for interface designers. They must ensure that even when a system component is in an error state that the system remains stable and available as far as possible.

In aviation systems a quick response may be required. However, the need for many responses can be predicted allowing for a training based approach enabling highly trained users to respond easily and appropriately. On approaching bad weather a flight crew will have some expectations of the situations which might arise and the actions consequently required. Schank (1982) provides a script based framework for remembering and behaving. Previously learned situations may be turned into scripts for application in similar, but not necessarily identical, situations. Training may be used to provide an appropriate repertoire of scripts for a given rôle such as “pilot” or “air traffic control officer”.

Suchman (1987) considers the use of plans in the context of interaction. She puts the traditional AI view of plans, with which she disagrees as follows:

"Mutual intelligibility is a matter of the reciprocal recognizability of our plans, enabled by common conventions for the expression of intent, and shared knowledge about typical situations and appropriate actions." (ibid., pg. 27).

This assertion may take too narrow a view of interaction; it may be more accurate if “plans” is replaced by “goals”. In a multi-agent system
it is not necessary for agents to understand the plans of each other for them to co-operate. In aviation the plan of the air traffic control officer with respect to their airspace is not available to the pilot. However, the pilot understands the goals of the air traffic control officer in terms of compliance with safety rules and aircraft throughput. In a commercial setting different departments within an organisation may co-operate without knowledge of plans. And, in a very simple multi-agent system, a word processor need not know the plans of an author nor the author know the algorithms used by the software. Suchman believes the plan based view of interaction to be flawed for different reasons. It is possible to construct a plan from prior intentions, typical situations and what is known about cause and effect in the domain. However, the link between prior intentions and situated actions (i.e. those that take place in a given context regardless of what may have been previously planned) is vague as the particular circumstances in which an action is required may have much greater effect than prior intention. Thus the notion of a plan is proposed to be weaker than the traditional position; it does not cope well with circumstances outside of those defining the applicability of a plan or which pertained when it was created.

It seems appropriate to consider a view in which both the traditional plan and situated actions should sit side by side. In circumstances where negotiation and flexibility are minimised (e.g. aviation, process control) the traditional plan is most appropriate as a way of determining sequences of actions and explicitly supporting co-operation. In flexible situations the situated action may have greater prominence allowing actions to be determined according to immediate needs when uncertainty makes a detailed plan less appropriate. Given this view it may be better to refer to the traditional plan as a constrained plan.

In commercial organisations Customer Service staff are required to react to a variety of customers who have a variety of requests which present themselves in real-time. The response must be quick, accurate and presented in an appropriate manner. An incorrect response may result in a dissatisfied customer, reduced profits for the organisation.
and varying amounts of inconvenience for the staff involved. In order to be able to service the customer the supporting computer system must not only be quick itself but must also provide the necessary information in a form appropriate for the task. Difficulties arise when a situation occurs which is not only novel but which is also outside the scope of training. In such cases decisions need to be based on knowledge of "first principles", prior experience, the capacity of the individual to make appropriate judgements or by reference to some outside authority.

In all of the illustrative systems mutual feedback is a necessary part of the design. In flying there are feedback loops which exist between the pilot and the other agents in the system enabling the pilot to verify that actions have had the desired effect. The nature of the environment determines the time-frame within which feedback must be received if the flight path is to remain stable or in an otherwise desirable state. The speed of the necessary response will depend on the stability of the system and what is required to keep it in a stable state. Pilots may be required to act "immediately" in some situations, those which are less stable, and within a minute or two in others. In commercial systems decisions are made on the basis of data contained in a database. When sales staff report that a potential customer shows indications of financial difficulties they will expect the credit rating of the customer to be reduced promptly to block further purchases. In financial systems reports provide feedback to check that targets can be met and that budgets have not been exceeded.

State replicability, or the lack of this, is perhaps one of the most telling symptoms of a real-time system. In all of the illustrative situations there are some circumstances when an "undo" option cannot exist. A mis-navigated aircraft cannot return to its last known position without using more fuel; an error in processing raw materials may mean the total loss of those raw materials; customers who have found a service
to be unreliable may not return in future\(^5\). It can be argued that there are word processing errors which cannot be undone: accidental erasure of files\(^6\), accidental printing, version changes (where past versions have not been archived). The issue with word processing is that, from the point of view of the application, there is no time limit within which corrections must be made. Any such time limits are imposed from outside of the system (e.g. by user management) and are thus not design considerations. In the real-time illustrations there are time frames within which certain actions must take place otherwise the actual effect will be different from that desired by the user/operator.

### 2.3.3. Effects of real-time on multi-agent systems

What difference does real-time make to a multi-agent system? Or should the question be posed the other way around? As a result of time constraints there will be limited scope for negotiation with other agents when tasks are being delegated. Further consequences are i) that there is a need for agent autonomy; ii) that these autonomous users must have reliable world models as they do not have the time necessary for their construction. The world models will need to contain the rules of the situation, be it business, manufacturing, aviation, etc., (what is and is not allowable, constraints, etc.) and will also need to be predictive with respect to the behaviour of other agents in the macro system. The human-computer interface needs to support these models.

\(^5\) If they do return they may be looking for signs of improvement. It will be very difficult to expunge bad experiences from their memory with a result that they may seek out an alternative supplier at some future point.

\(^6\) Utility programs are available to "un-delete" files. Examples are Symantec Utilities for Macintosh™ and Norton Utilities™.
2.4. Conclusion to Section A

There is a range of systems which may be termed SMART systems. All have:

- a number of agents each with some degree of autonomy,
- agents that may operate in parallel with one another,
- a real-time component placing temporal constraints on agents.

The lowest systems in the range may have just two, or a small number of agents, and a limited real-time component (e.g. word processing). The highest have a significant number of diverse agents, complex real-time constraints and may also be safety critical (e.g. aviation, power production). In between are moderately complex systems such as may be found in the finance and manufacturing sectors.

Agents within and between systems are diverse in their capabilities. They may also be human or computer in nature. In situations where the need for control is high, as in many critical systems, agents may have highly constrained plans. More flexible systems may easily allow situated actions where plans may be varied according to the needs of the moment. Some actions may be executed purely as a result of a local situation and not as part of a plan.

In using the term "real-time" it should be remembered that this is a relative term expressing the relationship between the time in which an action must be accomplished and the ability of an agent to accomplish it within that time. There is no absolute definition of real-time in terms of required speed of response.

In section B some specific SMART systems are considered. They are at various points on the scale of complexity and serve to illustrate some of the issues encountered by users.
Section B: Examples of Actual SMART Systems

2.5. Studies of Live Systems

In chapter one a number of real SMART systems were introduced. In this section of chapter two those systems will be further described. The systems that will be described (and their source categories) are:

i) a manufacturing organisation (data gathered from pilot study);

ii) from secondary sources:
   - medicine (e.g. Carr et al, 1992),
   - police aviation (e.g. Linde, 1988),
   - process control (e.g. Hoc 1989);

iii) from anecdotal evidence:
   - office automation,
   - process control,
   - aviation.

In further describing these systems it is intended that a better understanding of the complexities of SMART systems will be achieved and that important common issues can be identified.

2.5.1. Pilot study of a manufacturing organisation

The examined system, located within the Manufacturing function of the organisation, mainly supports Production (through Materials Requirement Planning functions) but it is also used by Customer Service staff and Finance staff in addition to their own systems (see figure 2.2). While the system provides a great deal of the communications infrastructure it is not the only means of
communicating. In this organisation there is a definite company culture. The elements of the culture include:

- Shared goals
  - high company profitability
  - high quality goods
  - high levels of customer service
- Communication
  - Regular, scheduled management meetings across functions
  - Email
  - Shared data - an accessible corporate data base
Figure 2.2: Multi-mode SMART system
Within each function (Customer Service, Finance, Production, etc.) managers have their own objectives. This can sometimes lead to goal conflict. An example of such conflict exists between Purchasing, (who try to obtain the lowest price for raw materials) and Production (who want to maximise output with the lowest level of inventory). The conflict arises because the lowest prices are gained on the largest orders - which is incompatible with lowest levels of inventory. This type of conflict can be resolved by reference to the higher goals (company profitability, customer service) and by management level negotiation.

The setting of goals may be the only area where the locus of control is a significant issue. Once goals have been set individual managers appear to have appropriate levels of autonomy needed to achieve the goals of their own functional areas and to assist other managers with their goals where there is an overlap of influence on the external environment. A further point to note is that the three main areas, shown in figure 2.2, operate in batch, real-time and hybrid modes. Thus there is no consistency of interaction mode across the macro system.

2.5.2. Systems reported in secondary sources

2.5.2.1. Medicine: telepathology
Carr (Carr et al, 1992) reports work in a telepathology environment. In the reported situation a pathologist was provided with computer controlled equipment which permitted the remote examination of cell samples. This supports fast diagnosis of conditions without the need to transport patients many miles to a physician. The examination requires remote control of a camera and microscope installation so that the cell sample may be viewed from a number of positions, the cell sample being larger than the field of the microscope. The distances between the pathologist's workstation and the microscope installation could be quite large (potentially transglobal) which, combined with the characteristics of the equipment could introduce a significant
control lag - 1.5 seconds from issuing a command to microscope movement. The effect of this lag is for the pathologist to make a first guess and then successively refine this in the light of visible feedback.

One solution to this type of control lag problem is reported by Carr which makes use of a simulation which does not have the lag of the actual system. In this system the operator directs the simulation to achieve the desired result; once achieved the appropriate instructions can be sent from the simulation to the live equipment. Carr indicates that this method is well suited to a situation where spatial precision is important but that "in telepathology real-time interaction is more important" (Carr et al, 1992).

The experimental work reported examines the interaction between the following variables: operator expertise (novice, experienced user), time delay (0.5, 2.5, 4.5 seconds), and interaction mode (keyboard, touch screen & trackball). In general experienced users completed tasks in 2/3rds the time of novices. Experts had better control of the trackball, made better use of the available "stop key" to prevent overshooting, had a better internal representation of the microscope position relative to the sample slide, and were able to combine commands into efficient sequences. Other findings were that the keypad was better when delays were short but that there were no significant differences in other conditions. The touch screen was beneficial to novices in the long time delay but not in other conditions.

In considering the question "what does the user need to know?" we can see that in this domain spatial awareness is highly useful. In this situation the domain specific knowledge is concerned with the relative position of the focal point of the microscope to the cell sample. This is a 3D representation where the X-Y plane is parallel to the top surface of

---

7 It is not absolutely clear from the paper if the 1.5 second delay is the command-movement interval or whether it is the command-movement-visible feedback interval.

8 This is not dissimilar to the effect that can be seen when a workstation user is using a mouse to control the cursor on an actively multi-tasking computer. In this situation the mouse movement becomes erratic and is difficult to place precisely.
the sample and the focus depth is parallel to the Z axis. In addition to this spatial awareness experts with this system also appreciate the inherent control lag in the system and make use of the “stop key” to minimise its negative effects. Finally the expert has a good understanding of the command set which enables efficient control of the system. These three knowledge areas (spatial location, system lag, and command set) are all highly domain specific and are largely mutually independent.

2.5.2.2. Police aviation
Observations of a multi agent system are reported by Linde (Linde, 1988). In her study a police helicopter crew consists of two: a mission commander and an aircraft commander. Typical missions are “search and rescue, emergency medical services, suspect pursuit, fire spotting, transportation of personnel,...” (Linde, 1988). In these situations there are two parallel hierarchies (mission and flight control) whereas in commercial aviation there is only one hierarchy which is clearly defined. Both flight crew need to have a good knowledge of sophisticated communications technology and are highly skilled in other areas too.

During a mission it is usually necessary for control to switch from one of the crew to the other a number of times depending on the current activity. For this to work well each officer must have the required degree of expertise, must understand the role of the other and there must be a good rapport between the two. Although there are two distinct chains of command (flight and mission) the pilot is seen to have a higher degree of authority and makes decisions about mission acceptance, point of commencement, and termination. This seems to stem from the overriding imperative of safety.

A major thrust in Linde's work is the use of types of utterance to negotiate authority from task to task, moment to moment in an environment which requires collaboration between agents. From her
findings there appears to be an interaction between such factors as: who needs help from whom, long term superiority, who owns the current task. Thus, in addition to the domain knowledge for each role, each needs detailed knowledge of the rules of interaction that apply in this domain.

2.5.2.3. process control
The control of a blast furnace used for the production of iron is the subject of a study by Hoc (Hoc, 1989). Three agents are proposed: operator, computer “colleague”, and blast furnace plant. The system is clearly a real-time system as it meets nearly all of the criteria given in section A above. One feature of the blast furnace system that makes it unusual are the very long response latencies. The inertia in the system, resulting from high volumes of material and extended cooling periods, causes the physical response times to logical inputs from the operator to be extended. As a result it seems that the operators have a model of the system which is not based on knowledge of system states but which is based on knowledge of dynamic processes with consequent effect on the operator’s knowledge representation. In designing a computer colleague with optimum usability we would expect to incorporate the operator’s representation into the user interface, i.e. the supporting computer system (agent) would also be expected to have a process based representation rather than one that is state based.

The long latencies problem is compounded by the indirectness of control. Hoc (Hoc 1989) gives an example of temperature control. Raising or lowering temperature is not just a matter of turning a heater control up or down but is achieved by more indirect means. The main factors given that affect temperature are the ratio of coke to ore, air blast temperature and flow rate. The effects of operator actions on these variables may take from a few minutes up to 5 hours to achieve the desired results. In the studied system the information necessary for decision making was not always directly available but had to be inferred through backward chaining or deduced through forward chaining from available data (ibid. pg. 54). From this description it
would seem that if the operator wanted to decrease the output of the furnace that they could not just reduce the volume of ore input but would need to relate such changes to air blast temperature and the coke to ore ratio. Failure to do so could result in inappropriate temperatures within the furnace or a reduction in the quality of output. Thus the operator needs to have a good understanding of the relationships between the variables in order to effect control.

Hoc suggests that the solution is a compound one. In order to have optimum performance of the overall system:

i) the operator should be appropriately trained to well identified and defined performance goals;

ii) the support system should have an interface which supports a non-state based representation of the system;

iii) an appropriate simulation tool should be available to allow predictions to be made about the effects of operator commands.

Given the use of a simulation tool Hoc also suggests that the operator’s job could be performed by increasing the level of automation and deskilling the operator rôle but that this is not always viable and that the optimum solution is to provide a computer colleague to assist the operator. Such a computer colleague would be a significant system agent. It would need domain specific knowledge (mathematical relationships between key variables) and would need to be able to communicate with the operator. The communication protocol would need to be understood by both the operator and their computer colleague. The design of the protocol would be dependent on the rôles and relative responsibilities of the main agents. Where the operator retains over all control, he or she would require some high level domain knowledge which would be supported by low level knowledge embodied in their computer colleague and made available during decision making.
While Hoc reports on a study of a specific process control system the work of Decortis et al (Decortis et al, 1991) presents a framework for considering temporal issues in process control based on a generic model of process control systems. The focus in their work is on temporal reasoning in the context of the situations, both normal and abnormal, faced by process control operators. It is suggested that temporal issues arise from three main sources (ibid. pp 55-59):

i) the physical aspects of the process being controlled,
   • process phases and sequences,
   • characteristics of machines (performance, reliability) and their interactions,
   • possible accidents;

ii) teamwork aspects of control rooms,
    • actions within the team not affecting the system,
    • actions affecting the system,
    • actions affecting the system with operator as mediator;

iii) the “time structure” of the operator in relation to the process & the team,
    • planning and plan execution,
    • evaluation and diagnosis,
    • co-ordination.

In addition events occurring in the system may also be categorised as occurring:

• regularly or irregularly;
• frequently or infrequently (ibid. pg. 57);
• [and to which could be added predictably or unpredictably].

From considering the descriptive parameters of process control systems Decortis et al (1991) go on to develop a framework for modelling the temporal reasoning of process control operators which may be used to predict and minimise temporal reasoning errors. On the way they provide some insight into the temporal structures present in process
control scenarios which may have implications for the broader domain of SMART systems.

2.5.3. Anecdotal evidence

This subsection provides further informal data on SMART systems. The data comes from the author's own experience and from other users of and within SMART systems.

2.5.3.1. Office automation
The example to be considered was part of the back office support operation of a financial institution. While mostly operating in batch mode it was sometimes necessary for departments to provide real-time support. Typically this would mean supplying information to other departments or to senior managers very quickly - stretching the capabilities of the supplying department. In some cases the information could not be supplied within the time given, in others the quality and depth of the information was sacrificed so that the deadline could be met. This latter option was easier in situations where the information requirement was not precisely specified.

A further issue was the batch exchange of information and services between functions. Often this would take the form of the provision of information at regular intervals but sometimes there would be ad hoc requests. A problem with such requests was a misunderstanding of the structure or availability of data. Different departments would hold similar data in different ways or would collect it in different ways. This would make analysis difficult and contentious. Though this problem may be overcome through the use of appropriate database design and appropriate control by a Data Base Administrator it does provide an example of poor mutual understanding between agents. In this situation the department agents\(^9\) lacked a full understanding of the capabilities and resources of each other. Unlike the tool manufacturing

\(^9\) Atomic and compound agents will be further considered in a later chapter describing Interaction Framework (Blandford et al. 1994).
example above there was no organisational culture of co-operation which could be used to resolve difficulties easily.

2.5.3.2. Process control
The process in this example was a canned food manufacturing system. This system was described in chapter one, the relevant findings appear here. The cook was responsible for controlling the loading and unloading of retorts with batches of tinned food and was also responsible for the control of temperature, pressure and the duration of cooking, typically about 20 minutes. Training for this job was minimal - a list of instructions issued verbally and some observation of an experienced cook. Instrumentation was in the form of analogue gauges for temperature and pressure, and a 12 hour clock to mark time. To operate the system effectively the cook needed to track the process in each retort. Some cooks relied on paper & pencil notes while others were able to perform the tracking task internally; these tended to be the more experienced cooks.

The relationship between the cook and the retorts was one way. The cook had all the control. From this point of view it is arguable that the retorts were not agents but were tools, given their physical/deterministic nature. However, the variances in the system meant that each loaded retort was regarded as non-deterministic, particularly by inexperienced cooks. Such cooks found it necessary to monitor the gauges of each retort closely to ensure that correct operational parameters were maintained; they found tracking difficult. In this situation it was as if the retorts were virtual agents despite their actual deterministic natures. If this description is accepted then the novice cook has a number of independent retort agents to deal with. Although the retorts were broadly similar, being nearly identical to each other, the novice cook would need to monitor the instruments for each retort so that further input could be given at the right time. While a novice cook would have a workable model of the cooking process, and therefore of each retort agent, they had not fully internalised the model so that tracking was not automatic and the between retorts variances not learned as was the case with an
experienced cook. Difficulties could occur when the retorts had to be handled in quick succession. A novice would interact with retorts in a hurried way, having to take special care not to confuse the inputs to the various retorts. In other words a novice would need to issue instruction sequences to each retort agent with the possibility of mixing the sequences up which would result in process errors. In practice the performance of novices was close to that of the expert cooks, but experts would achieve their results with greater ease and less apparent stress.

2.5.3.3. Civil aviation
Two pilots provided data about the flying of civil passenger aircraft. Two significant issues came up with respect to agents, both automated and human.

The automated agent of concern was the NavCom (navigational computer). This device supports manual flying and provides the automatic pilot. Two versions exist: Mk I and Mk II. In the (older) Mk I the flight co-ordinates for the whole journey needed to be entered by hand by a pilot at the start of each journey. The Mk II did not need this manual input as the location and flight path of all the worlds airports were pre-programmed, the only data needing entry are the start and finish locations. The pilots reported a problem with the Mk II: it could cause the aircraft to make unexpected turns! The pilots observed that the auto-pilot status indicator was inconspicuous and that having activated the auto-pilot it was possible to forget that this was its status. It is suspected that this was compounded by the low level of task engagement brought about by the simplified user interface. The pilots were not aware that the subordinate NavCom agent was engaged in a particular task. Awareness could have been improved either by the use of a different running interface which would give a clearer indication of the agent’s status or by finding another way to increase task engagement without negating the benefits of having an auto-pilot.
The pilots also indicated that air traffic control officers are not all the same. A major difference is language. Native English speakers and air traffic control officers from Western countries could be regarded as fluent in English while those from other regions may not be. This can be significant when problems arise. An example was given of a long haul flight from an Eastern European airport to Tokyo. During the flight a passenger became seriously ill. The pilot deemed the best assistance could be gained by returning to the point of departure. Unfortunately the concepts were beyond the English capabilities of the air traffic control officers. After some attempts at negotiation the pilot had to state that he was going to return the aircraft and that the air traffic control officers would need to accommodate that decision. After some thirty minutes the air traffic control officers had found a suitable interpreter who was able to assist with the negotiations between the pilot and the air traffic control officers so that a safe landing could be achieved.

In this example the problem could be seen as one of communication. It can also be viewed at a higher level as one of ability in a particular skill, which happens to be communication. The pilots were aware that the air traffic control officer agents varied in their ability to communicate and that therefore they needed to be handled, interacted with, in a variety of ways. In the case above it meant that the pilot assumed a different, more senior, rôle and that the air traffic control officers were given a well defined task to carry out (just providing a landing slot rather than negotiating and providing a landing slot). Once the agent’s inability to perform the required task had become known then a simpler, manageable task was assigned.
2.6. Discussion of SMART System Examples

2.6.1. Factors arising

An examination of the previously described example systems reveals a number of significant issues. Each is significant in at least two of the examples, some are significant to many more. In the following sub sections each issue will be briefly described and related to the appropriate examples.

2.6.1.1. Communications skills
Communication skills are necessary for one human agent to communicate with another agent of either human or computer type. There are linguistic and social elements to these skills. When communication is between people without these skills communication is made more difficult or becomes prone to failure. Communication with computer agents requires a different kind of interaction knowledge to ensure that communication is effective and efficient.

Different systems require different communication skills. In the manufacturing example there was a requirement for high levels of social skills so that managers and non-managerial staff could negotiate with each other. The database providing information on the organisational parameters and variables had a fairly simple interface which did not require a great deal of sophistication from its users. The financial back office system was similar to the manufacturing system in these respects. In the telepathology example the focus was on the use of commands with keypad/trackball to manipulate a device. Sophisticated interpersonal communication was not an issue but device control expertise was. In the police aviation (helicopter) system language was important in reflecting the complex social and hierarchical structures of which the crew were a part. Civil aviation provides example situations where communication between individuals and with equipment is vitally important. As the example showed it is also important that equipment be suitably designed so that it is easy for the flight crew to notice and interpret output. When this is not the
case, and a failure of communication result, the effects can be alarming for those concerned.

For communication to be effective it may be necessary to have highly trained users and appropriate display design. However, in less critical systems the need for this is reduced and it may be possible to rely on the natural language skills and moderate computer literacy possessed by those people in the system. The decisions in this area will be informed by the criticality of the system and the presence or absence of time constraints which may or may not allow for negotiation to resolve problems.

2.6.1.2. Interaction protocol
Some systems have a formal or semi-formal communication protocol to be observed. This may be regarded as distinct from communication skills because the protocol rules are explicit. In the police aviation, finance and civil aviation systems there is a complex control structure such that the significant agents have different responsibilities and reporting/control hierarchies. The protocols provide a means for ensuring that the responsibilities of each agent are not subverted by another. Thus in the financial system the protocols ensure that staff from one department do not do the work of another at the expense of their own. The protocols were effectively only used at the start of interdepartmental working. Once the working rules had been established then it was not usually necessary to invoke them again.

Explicit protocols may be useful tools for ensuring that the boundaries between co-operating functions of similar status are maintained. They reduce the ambiguity about who is responsible for what tasks when co-operation is required.

2.6.1.3. locus of control
In the three systems mentioned previously plus the manufacturing system the locus of control in decision making is an issue. This has already been alluded to when considering protocols. In the manufacturing situation the locus of control became an issue with
inventory levels - functional areas of equal status had opposing views. This was resolved by reference to a higher authority. This may be done actually, by using an arbitrator, or virtually by referring to the higher level objectives of the organisation (e.g. maximise profit). In this way sub-optimisation may be avoided and explicit recognition gained that one or other side has bowed to the higher level need. In the aviation and finance systems the locus of control in any one situation may be with one agent or another and is negotiated explicitly (in the financial system) or tacitly (in aviation).

2.6.1.4. Goals and goal sharing
In the manufacturing organisation explicit use was made of goal sharing. This facilitated decision making by agents acting alone and when co-operating. In the case of an agent acting alone the sharing of goals could make the decision making process easier by helping the decision maker to know that what was being decided would comply with the needs of other parts of the organisation. In the financial system this was not the case and decisions made by one department may have been found to make difficulties for another after the decision had been made.

In the iron production process (Hoc 1989) it is suggested that the operation of the system may be improved by the creation of a computer colleague who would support the human operator. In such a system Hoc (ibid.) suggests that there will need to be a shared representation of the domain which is process based rather than state based. For this support to work there will need to be some mechanism for the computer colleague to recognise or represent the sub-goals of the operator. If this is not the case then the computer colleague will need to be guided through operations step-by-step, ceases to be a colleague and becomes a much more simple tool.

In police aviation both of the crew know the goals of the mission. Temporary mission goals are stated at the start of each mission while more permanent goals are a part of the police rôles of the crew. It seems that in this system the two crew agents know the rôles, and to
some extent the goals, of their crew partners. This assists in the
decision making process as unreasonable requests can be avoided and
goal conflict minimised.

2.6.1.5. Agents’ models of other agents
To some extent the categories listed above form part of each agent’s
model of other agents in the system - in order to communicate with an
agent some model of what that other agent will understand is required.
In addition it may be necessary to have a model of the operational
status, characteristics and capabilities of each other agent. From the
systems reviewed there appears to be support for the idea that such
models may be used for delegation of tasks and requesting assistance.
At the most obvious level if a task is to be delegated or assistance
requested then an agent needs to know which other agent is capable of
providing that support. However, as in the civil aviation and financial
office systems, the models may be faulty - the supporting agents may
not have the necessary capabilities. In these situations problems arise
which can be difficult to resolve. In the civil aviation case it may be
necessary for the pilot to over ride the requests of an air traffic control
officer where, through inadequate language fluency, they cannot
understand the problem to be resolved. In the financial system
mismatches may be resolved explicitly through negotiation or may just
result in work being completed to a lower quality level than is
required.

In addition to knowing the overall capabilities of another agent it is
necessary to know the status. For example, is the agent available to
delegate to, is the agent actively engaged in the task that was
requested, how long will the agent take to complete a given task and
with what reliability, is the agent able to work to given quality levels,
etc.? These and other modelling issues can be seen in all of the
reviewed systems.

In the manufacturing example explicit models of external agents,
suppliers and customers, were constructed. The models were used to
determine the “quality” of a supplier and considered such issues
as: price, standard of materials/services provided, delivery reliability, delivery lead times. Models of customers were formed which considered financial viability and purchase history. With respect to internal agents one manufacturing manager had a model of the behaviour of operators in the case where machinery fails. While the charge hands should refer to the production schedule to determine the next task they would in fact do the next task which would produce the greatest bonus. This would require production schedules to be changed more than would otherwise have been necessary. Another part of this model of operator behaviour was that maintaining the status quo would facilitate industrial relations and smooth production.

Models are also found in the manufacturing sales systems which perform customer credit checks. If there is inadequate credit for a particular purchase then supportive systems inform the user that the sale cannot be processed as this would take the customer over their credit limit. In such a case the rules of business are thus incorporated into an implicit model of the domain while model parameters are explicit (e.g. credit limit) and used in equations following from the model.

More explicit models are found in some systems which define some part of the macro system using one or more parameters. In the manufacturer's model of its customers and suppliers these models are of the world outside the system that would normally be directly under the organisation's control, i.e. would be within the system boundary. It is suggested here that models are sometimes employed in order that these external agents may, in some specific way, be controlled by the organisation. This has the effect of reducing the uncertainty which exists in the environment in which the organisations find themselves.

In the telepathology and iron production systems the use of explicit models in simulations is considered as a way of overcoming physical constraints. In the telepathology system the constraint is the lag inherent in the control system, in ore production it is the response latencies. In both systems the operators have high level models of their respective systems but lack the fine, low level detail that would
be necessary if they were to effect fine control. In telepathology the clinician knows that there is lag in the system but is unable to reliably predict its effect when starting a movement of the microscope. In this case the use of a “stop” button is considered preferable to a command simulator as “in telepathology real-time interaction is more important” (Carr et al, 1992, pg. 257). Hoc (Hoc 1989) is uncertain about the benefits of using a simulation which may deskill the operator’s job. He proposes the use of a computer colleague which has a low level model of the process supporting the operator’s high level model. This is thought to be the best fit with the operator’s cognitive processes than by adopting other approaches which do not make use of existing cognition or which would require a change.

In both the telepathology and iron production systems the interface used conveys something of the status of the system to the system operator. The operator’s focus of attention is on the behaviour of the system. In the civil aviation example where the crew were not aware that the auto-pilot was engaged their attention was on the behaviour of the system as the aircraft made unexpected turns. These turns would not have been unexpected in the status of the auto-pilot agent had been known to them either through a more significant display component or through increased crew engagement with the task. In this case the status of the agent as either on or off was not clearly known though the capabilities of the auto-pilot were. This is different to the case of the East European air traffic control officers whose status was known (they were engaged with the negotiation task) but whose capabilities were yet to be determined.

In police aviation the crew members have models of what each may do through their understanding of each other’s roles. Some requests may be made while others may not. Similarly in the financial office systems certain requests may be made while others may not. A senior manager may make a broader range of requests and expect them to be complied with than would be true for less senior staff. It also helps organisational efficiency if requests are placed with the right individuals or departments, those with the resources necessary to
provide the solution most directly. The same applies to file and database interrogation; the right data source has to be identified before questions can be asked. These procedural points all require some basic model of the capabilities and availability of other agents in the system.

In the food processing scenario the cook must be aware of the status of each of the retorts. In this system some data, temperature and pressure, was directly available via instrumentation. However, the time into the current cooking cycle was not. This would need to be internally tracked or monitored by use of external aids such as notes made on the finish time for each retort. This system could have been made much easier to control either by deskillling and the use of automated timers or by providing simple audible timers for each retort.

In the civil aviation scenario the air crew are required to have models of the competence of the air traffic control officers with respect to English to ensure that appropriate communication language and protocols are used. They are also required to understand the various major subsystems such as engine management and NavCom. Failure to have adequate models may result in significant difficulties or surprises during the flight as have previously been mentioned. Experience, training and appropriate instrumentation design all support the formation and use of models of other agents within the aviation system.

A further consideration is that users need to have models of the models being used. With the organisations investigated this model of the model seems to consist of: i) uses to which information from the model may be put, ii) what the model does not show, iii) knowledge of the reliability and scope of the data from which the model is constructed, iv) the reliability of the model. Without these implicit meta-models the author feels that the users would not make effective use of the models made available to them in the systems that they use.
2.6.1.6. Plans
In all of the SMART systems reviewed there was some need for planning. Plans could be extended, as in the case of manufacturing production plans spanning days or weeks; or they could be short as in the planned microscope control actions required in telepathology. In some cases, such as aviation and financial office systems, there is a need for plans to be exchanged or reviewed by several agents so that they may co-operate in expediting the plan in the most appropriate way.

In civil aviation the pilots plan to land at a given airport is explicitly communicated to the air traffic control officers he or she has to deal with. This plan may be modified by either side if circumstances dictate. In dealing with the MK I NavCom the pilot makes the flight plan explicit while in the MK II the flight plan is inferred by the NavCom from the start and end points of the journey and need never be articulated by the crew. In police aviation mission plans are negotiated between the crew members. In such cases the plan is explicit and known to both agents. In other systems, e.g. food processing and telepathology, the plan need only be known to the active agent as the equipment being interacted with does not have any facility to negotiate or alter the plan.

Castelfranchi & Falcone (1994) suggest that plan recognition is often implicit and based on the real-world knowledge of the individuals involved and that this capability does not naturally exist in computer systems designed to support co-operative work but has to be designed in. A number of approaches to plan recognition are briefly reviewed by Malinowski (Malinowski, 1992). The concern of this thesis is not with the mechanism by which plans can be formed and recognised but that plans exist and may need to be known to more than one agent.

2.6.1.7. Simulation
In two systems simulation was considered as a tool to overcome response delays in the system. In the telepathology system the delays...
were short but made precise control over the equipment very difficult. In iron production the system latencies were very long and were combined with indirect control mechanisms which also made precise control difficult. A common solution to these problems is the use of simulation which would enable control actions to be hypothesised and tested without the risk and delay of interacting with the live system. Such simulations need to be accurate and reliable and their users need to know their capabilities and limitations.

Simulations may also be used in systems where there is high risk. In aviation simulators are used to facilitate training. This reduces the risk to passenger and to the owners of aircraft while allowing the crews to gain experience of a wide range of situations. Process control may also benefit from the use of simulators as they enable operators to try out potentially dangerous sequences of operations without any risk. None of the example systems used simulators in this way but their potential use is noted.

2.6.1.8. Domain knowledge
In all of the reviewed systems domain knowledge was an issue. Without domain knowledge it would not be possible to carry out the required tasks. This issue becomes more complex when computer agents are considered. In the iron production system Hoc (1989) considers three ways of embodying domain knowledge in an automated agent: i) deskilling of the operators job, ii) simulation of the process, iii) provision of a computer colleague. From a human factors point of view the third of these was thought to be the most desirable. In civil aviation the MK II NavCom was given greater domain knowledge, which had undesirable consequences despite the reduction in load on the pilot during pre-flight procedures.

In considering the sample systems a further issue arises: that of the difference between novices and experts. Three differences emerge with respect to domain "knowledge": i) theoretical knowledge, ii) experiential knowledge, iii) skill or control knowledge. These categories do not arise in each of the example systems. In the
telepathology example there is a difference in control knowledge between novices and experts, with experts being able to control the movements of the system more accurately and efficiently than novices. In civil aviation more experience pilots will have a broader experiential knowledge of the performance of the world’s air traffic control officers than will a relatively new pilot. The manufacturing system provides an example of variable domain knowledge: sales staff who have been with the organisation long enough to remember which products were discontinued or renamed some years ago. Such knowledge would not be available to recently recruited staff.

2.6.1.9. Time windows
Another feature of all of the studied systems is the existence of “time windows”. Where time windows exist an agent may be required to carry out some action or provide a response within certain time limits outside of which the action would be inappropriate. In the telepathology case the use of the stop button only makes sense when the microscope is in the vicinity of the target. Using the stop button too soon or too late will cause under- or over shooting. In aviation the time constraint on flying time mean that certain actions have to be completed before the end of flying time, or before the end of some phase of the flight or mission. In the food processing example the reduction in temperature and pressure of food retorts must be done after the food is cooked but before it becomes over cooked.

Agents need to know when an action may be started and the time by which it must be complete. In some cases the determination of these start and end points may be explicitly time related (police aviation, food processing, finance office), in other cases start and end points will be contingent on non-time based events (such as the sequence of events during take off and landing).

10 In the situation that was observed during the pilot study one of the clerical staff received a phone call enquiring about the availability of a particular product. He was able to inform the caller that the product had been discontinued some eight years previously and that a similar product could be found in another range. This information was not available on the computer system.
A critical example of a time window can be seen in police aviation. The length of the mission is determined by police need, fuel carried and flying manoeuvres carried out. The police mission needs to be completed before there is insufficient fuel to return to base. At some point on a mission the decision to return must be made, failure to do so results in a crashed or stranded aircraft. Thus there is a time window in which the decision must be made, in the case of police aviation the closing point of the window is defined by distance from base and fuel remaining.

A less critical example was found in the manufacturing system. Some raw materials are delivered three times a week (say, Monday, Wednesday, Friday). On delivery a quality check is made. This is not instantaneous as there are inevitable logistic and technical delays. A delivery on Monday may be up to standard, so Quality Control inform Manufacturing and Purchasing that this is the case. On Wednesday there may be a problem. In this case it is necessary to inform both Manufacturing and Purchasing and the supplier so that i) a replacement can be provided and ii) so that the same problem does not occur with Friday's delivery as this would give further production and scheduling problems. In this case the time window for informing the suppliers is before the batch for delivery on Friday is made up and despatched. Time must be allowed both for delivery and for the problem to be corrected. If the supplier cannot resolve the problem in time then an alternative source must be found. Failure to act appropriately within the time window may result in considerable financial loss as a result of down time during which there is no production but for which staff must be paid.

Where time windows are long agents may choose when they execute actions. This may not be the case when windows are short. Short windows are indicative of real-time systems in that the requirement for a response from an agent within a given time must be reasonably well matched to the ability of the agent to give that response. Long time windows may allow agents to respond in batch mode, i.e. to
provide responses at a time which is both convenient and efficient for them.

2.6.1.10. Making real-time decisions from batch data

In the companies investigated there seem to be two approaches to the use of batch data to support real-time decision making. The easiest approach, from the system designer's point of view, is to make batch data available in its unprocessed form. For example, users in a commercial organisation may look at stock levels or customer accounts, as they were recorded at the end of the previous day's trading. Using this data plus knowledge of sales made, or likely to be made (i.e. by using an internal model of the domain) users are able to answer customer queries on what is available and when.

A more complex approach is to incorporate the batch data into an explicit model. In two examples the batch data was used to compile an index of performance of a customer or a supplier. A slow paying customer would be placed in an appropriate category. In a purchasing system a supplier with a less than perfect record on supply time and quality would be given a rating which reflected that performance. Customer and supplier ratings would be used in negotiating terms of trade either in batch or real-time modes.

Both of the above approaches are interface based in that the emphasis is not on system design but on interface design: information in the system is made available to the user without major functional changes. An alternative approach would be to redesign the system to capture data in real-time and make this available in real-time too. However, this may be an expensive option requiring batch systems to be replaced by real-time systems. Where systems may be due for

---

11 Batch data is data which has been collected over some business period, bundled together and used to update a central data store at some convenient point in time.

12 Note however that programs need to be written and integrated with the macro system in order to provide the additional interface facilities.
replacement it may be appropriate to incorporate such structural changes.

Note that just making information available in real-time would not solve all problems, as indicated by the need for the compiled indices of customer and supplier performance. Users need external models to support specific tasks. These models thus tend to be reorganisations of existing data, which are updated as new, relevant, data is gathered. In the organisations examined by the author the models are qualitative rather than quantitative and tend to be relatively static.

Decisions made in real-time situations require the user to have a high degree of autonomy, or to defer to the authority of the computer system that they are using. If this were not the case then there would be a great need for users to refer decisions up the management tree. Strategic and tactical managers would be forced to become operational managers; a situation they could be expected to resist.

A further issue is the potential use of the telephone to enable real-time users to contact other users (batch or real-time) to get required information. This means that there must either be a low number of such enquiries or that there must be relatively equal numbers of users who are able to both make and answer phone enquiries.
2.7. Conclusions

Simultaneously Multi-Agent Real-Time systems are found in many domains and with wide ranging complexity. Most do or would benefit from some form of computer support in the form of automated agents that complete tasks that it would be difficult for people to do. In looking at a range of SMART systems a number of issues have arisen which are important to their use and which should, therefore, be considered during design. The ten issues so far identified are:

1) the communication skills of agents,
2) interaction protocols,
3) the locus of control,
4) goals and goal sharing by agents,
5) agents models of other agents,
6) plans,
7) the use of simulation,
8) domain knowledge,
9) time windows,
10) real-time / batch hybridisation.

For any given system the above issues need to be qualified and quantified so that appropriate design decisions may be made. The examples used in this chapter show some of the problems that can arise when there is inappropriate design. The following chapter considers ways of modelling systems which support the identification of what a human agent needs to know by making the important issues explicit.
Chapter 3. Review of System Modelling Methods

3.1. Introduction

An assumption has been made that to be able to identify what a user needs to know in interacting with a system it is useful to model that system. By producing models of systems it is possible to put users into context. This should help to identify the roles of each agent and what each user or computer agent must know in order to be able to fulfil their roles. Such information would be of assistance to system designers. With the support of suitable models designers could identify problem areas which require either a system modification, or special training and/or selection of users, so that problems not easily resolved in technical design may be addressed by human resource management. Identifying problems early in the design cycle allows appropriate solutions to be found which will reduce costs incurred through erroneous system operation, system redesign and reimplementation, and operational inefficiencies.

Appropriate modelling methods are scarce; few deal effectively with both real-time and multi-agency. A number of methods from a variety of backgrounds within HCI are reviewed from a non-SMART perspective. Each review is intended to give a high level description of a method and represents a summary of the analysis of more detailed sources.

Following the reviews those methods that may potentially support the modelling of SMART systems are assessed for suitability. However, there is a problem in that a number of different modelling techniques exist, each with different strengths and weaknesses with respect to SMART systems. This requires the prior development of criteria against which the various techniques can be tested, allowing the most appropriate to be selected. Obviously the criteria themselves need to
be appropriate for this purpose. They need to be sufficient so that the question “does the method help find out what a user needs to know” can be answered; but not so stringent that no technique can meet them. They also need to eliminate unnecessary detail from techniques that would not help to answer the question but which would require modelling effort to describe.

In the process of assessing suitability each method is given a label representing how well it fulfils each of the criteria. Brun and Beaudouin-Lafon (Brun and Beaudouin-Lafon, 1995) use labels such as “empty” or “quarter-full” to describe how well various “formalisms” used within HCI comply with each member of a set of twelve criteria. These labels are then translated into shaded areas and grouped together so that a profile for each formalism becomes visible. A similar but not identical approach is adopted here. For brevity the labels “0” to “4” have been adopted. These can be said to correspond to Brun and Beaudouin-Lafon’s “empty” and “full” labels, or to labels such as “non-compliance” and “high-compliance”. Once labels have been attributed to each method for each of the criteria, a profile across all criteria can be obtained and methods compared for suitability. This approach should allow an all embracing modelling method to be identified or, if a single method cannot be found, a set of methods may be identified so that a resultant compound method can be used to produce a model of a system that will allow the required user knowledge to be identified.
3.2. General Review of Modelling Methods

In trying to determine what an agent needs to know we must first consider the system that the agent is in. This could be done for each specific situation possible for any given system. However, such an approach would need an exhaustive and explicit consideration of all the permutations. This would not be manageable for any but the simplest systems. Each possible situation would need to be associated with the specific knowledge applicable in that situation, this would be very time consuming and would result in huge volumes of data. A higher level approach is needed which allows analysis of the whole system at a reasonable cost.

One way to approach this is to consider not just specific situational knowledge that is needed but meta-knowledge such as: knowledge of the existence of another agent that tasks may be delegated to; domain boundaries; knowledge of the meaning of time constraints. In this way a generative knowledge base may be constructed and stored for later use. This could be expected to be more compact and less time consuming than an exhaustive list of situation descriptions where the specific response by each agent in each situation would need to be identified. An example of this was seen in the pilot study conducted at the manufacturing organisation. One of the sales support staff had extensive product knowledge which covered current products and those made obsolete several years previously. In terms of having a model of the agent using meta knowledge we would describe the sales support agent as knowing about all of the company’s products from year X up to the present date. The model of the other agent would not need to contain precise and complete data on all of the products, it is sufficient to know that the other agent can access these.

A search for suitable modelling methods was conducted and revealed that most had obvious shortcomings. However, once Interaction Framework (I.F.) had been found it seemed an obvious choice: both multi-agency and temporal descriptors are fundamental components.
Having identified Interaction Framework as "the answer" it seemed appropriate to take a step back and compare it with other modelling methods, firstly within the broad context of HCI and secondly in the narrower context of SMART systems.

Many methods could have been subjected to analysis. The selection presented here includes some of the better known from the field of HCI plus a number of others that are candidates for modelling SMART systems. The methods, and the sources on which scoring is based, are as follows:

- Ethnography
  (Hughes J., King V., Rodden T., Andersen H., 1994)
- Graphical Dialogue Environment, GRADIENT
  (Alty & Ritchie, 1991)
- Interaction Framework, I.F.
  (Blandford, Harrison and Barnard, 1994 & 1995)
- Executable User Action Notation, XUAN
  (Gray, England & McGowan, 1994)
- Programmable User Models, PUMs
  (Young, Green & Simon, 1989; Blandford & Young ESPRIT 7040 reports, 1994 a,b,c)
- Resources Model, RM
  (Fields B., Wright P., Harrison M., 1996 ab; Wright P.C., Fields B, Harrison M.D., 1996)
- Goals Operations Methods and Selection, GOMS
  (Card, Moran & Newell, 1983; Johnson, 1992)
- Abstract Reasoning from Concrete Clues, ARCC
  (Decortis, de Keyser, Cacciabue & Volta, 1991)
- Dialogue analysis, Milan model
  (De Michelis & Grasso, 1994)
- Coloured timed Petri nets, CTNs
  (Furuta & Stotts, 1994)
- Hierarchical Task Analysis, HTA
  (Shepherd, 1989)
In looking at each descriptive method one or more sources have been reviewed. This is intended to provide enough data to evaluate the method's suitability for application to SMART systems. It is expected that this will provide enough information to make a decision regarding the suitability of a method rather than provide a deep understanding of all of the subtle and less subtle nuances of a method. From this a decision may be made about the value of using a method in the analysis of SMART systems.

There is some overlap between modelling approaches. For example, process modelling can be considered to be part of a systems based approach, but processes are also important to cognitive modellers such as Hoc ( Hoc, 1989 ); temporal logic, in various forms, may be used as a part of a number of modelling methods, though not so far, it seems, by ethnographers.

The modelling approaches considered are:

- ethnographic methods,
- system oriented modelling,
- temporal logic,
- cognitive modelling (including PUMs, the Resources Model, GOMS and ARCC),
- dialogue analysis,
- task analysis.
3.2.1. Ethnography

The use of ethnographic methods has increased in prominence in recent years as the field of CSCW has become more prominent. They have potential for describing multi-agent systems and so are considered at length here. The role of ethnography at various points in the design process is considered by Hughes et al (Hughes et al, 1994). Naturally, the focus of the approach is on the social aspects of systems that contribute to their success or failure. This seems appropriate given that the increasing complexity of work and the use of network technologies leads to work of a more collaborative and co-operative nature with all the social implications that follow. Hughes et al (ibid.) suggest that one problem with the take up of ethnography is that “…[it] seems too unsystematic a method, its results presented in a discursive form, design options are not clearly stated and do not attend sufficiently to [software] engineering needs.” This is countered by the argument that too often systems are built to meet engineering needs at the expense of missing the needs of their real-world settings. While both positions are correct the constraints on system designers and builders remain: designs must be quantifiable and reducible to algorithmic representation if they are to be built. This problem needs to be solved by ethnographers if they are to become a readily accepted part of the design team.

An additional problem for ethnography is the transience of teams and of the work environment. In commercial and industrial settings we can expect some domains to have a relatively high staff turnover during the life of a system. As staff leave teams and new people join the social dynamics may change. It would be expensive to recognise this in system design in terms of built in flexibility and adaptability or to be prepared to manually modify systems in accordance with changing social dynamics. This may not apply in a system with very low staff turnover or where the population is homogeneous. Additionally, the length of life that systems have is limited as organisations and their environments change. Perhaps the best purpose of ethnography is to
perform primary research which can be used to inform the design process in a general way rather than to provide specific and detailed design requirements.

However, Hughes et al (ibid.) propose ethnography as having specific rôles in the design process. Because the proposals are so specific it seems appropriate to consider this approach further. The specific rôles they propose are:

- during early rapid prototyping,
- quick and dirty ethnography,
- evaluative ethnography,
- used in the re-examination of previous studies.

3.2.1.1. Ethnography during early rapid prototyping
The application of ethnography during early rapid prototyping is seen as a method for collecting data that is not normally obtained by other methods. Hughes et al (ibid.) suggest that when run in parallel with conventional "cognitive and task analytic approaches" ethnography revealed "...moment-by-moment mutual checking..." which would otherwise have been missed. In other words the method does not provide a model of the system but additional data to describe the system, albeit in terms of the interactions between individuals. In some cases the method may augment a more substantial model produced by other means by adding communication links not otherwise found.

3.2.1.2. Quick and dirty ethnography
Trying to obtain results from the full application of ethnography to large projects is problematic for a number of reasons. Not least is that development teams and users on such projects are under pressure, there is little slack for the inevitable overhead required to support the data gathering needs of ethnographers. This is exacerbated by the technical inexperience of ethnographers with respect to the development of computer systems. It would be unusual to find an
ethnographer who was also a competent system designer because of the considerable differences in training required. Accepting this, a “quick and dirty” approach does allow key information about the social setting of a system to be gained which may be used to inform high level design decisions (ibid.).

3.2.1.3. Evaluative ethnography
Evaluative ethnography is effectively a “quick and dirty” check focused on a design prior to implementation. Hughes et al (ibid.) given an example of a study in which a design proved inadequate because it did not recognise the disrupted nature of working with the public. In their example there was a need to complete administrative tasks but that these would often be interrupted by customers seeking some, unpredictable, form of assistance. The proposed system had not recognised that its users had a need for informal information such as: where to get help with particular problems, and phone numbers of outside bodies; and had not identified the disrupted nature of the work. In this case the use of ethnography proved to be useful. However, ethnography is not the only way to identify such problems. Usability engineering or detailed task analysis could also have revealed these flaws.

3.2.1.4. Ethnography in the re-examination of previous studies
Useful lessons may be learned from the application of ethnography in the re-examination of previous studies. The study of existing systems, with their strengths and weaknesses may be used to inform the design of new systems (Hughes et al 1994). What is more, a library of systems may be stored so that a designer of one system may be able to learn from a whole range of systems. This could enable a designer to gain insight into system design and may lead them to consider new possibilities. However, this approach relies on the subjective analysis carried out by each technical designer and so may well produce varied results.
3.2.2. Systems based approaches
Systems based approaches consider the functional elements of systems: inputs, processes and outputs. Any one approach may consider some of these elements in more detail than the others. Typically the focus will either be on process, or on interaction between processes, i.e. on inputs and outputs to systems or sub-systems.

3.2.2.1. Process focused analysis
The focus of Abbott & Sarin (Abbott & Sarin, 1994) is not on agents or communication but on processes. This concept may prove to be useful to SMART systems analysis by providing an analysis method complementing those which focus on agents and communication. A key issue for work flow management is the decomposition of a process into a task hierarchy. Such decomposition may support the identification and specification of agents though they do not give this as a benefit but assume that agents are already defined: “[the]..assignment of tasks to roles that are place holders for users who will perform tasks.” They seem to assume that it is only users who are of interest in work flow analysis and not computer agents. Taken literally this would narrow the usefulness of work flow analysis, however, the substitution of “agent” for “user” would seem to be acceptable. A more significant problem is that they do not mention event timing as a significant issue. For many systems timing is a crucial issue and any form of task analysis which does not deal with this satisfactorily may not be applicable to SMART systems.

3.2.2.2. Interaction focused analysis
Not unnaturally a variety of approaches to modelling interactions within systems have been developed. Often these are directed to the development of optimised interfaces for devices with which people interact (e.g. Alty & Ritchie, 1991). While that is not the purpose of the present work it is considered that such work may be illuminating.
3.2.2.2.1. The GRADIENT project
The GRADIENT project (Alty & Ritchie, 1991) is directed towards the
development of what is, in effect, an intelligent assistant for process
controllers. As such the work focuses on developing a specialised type
of agent occupying a specific rôle in process control systems. This has
similarities with the previously mentioned work of Hoc (Hoc, 1989).
However, the GRADIENT approach is different to that of HOC in making
considerable use of formal methods such that a generalisable set of
tools for designing assistant agents becomes available. By assuming a
certain interface architecture analysis of the interaction between the
operator, the intelligent assistant and the process can be made
amenable to algebraic analysis. The result is a method for formally
specifying low level interaction events. As such it is not an appropriate
method to consider here.

3.2.2.2.2. Interaction Framework
Interaction Framework (Blandford, Harrison and Barnard, 1994 &
1995) is promising for a number of reasons. Firstly, it explicitly deals
with time in its representations of systems. Sequences and parallel
events are allowed for as are the starting points and duration of events.
IF also takes an agent neutral view in that the system description does
not centre on any one agent, such as a specific user, but describes all
agents and their associated system features in the same even-handed
way; no distinction is made between computer and human agents. This
is useful in systems where there may be no single agent that can be
said to be in charge - the question raised by Linde (Linde, 1988).
Agents are also considered to be compound or atomic. Compound
agents are those who are unified in some way and may appear as a
single agent for some purpose1. Atomic agents are single agents who
may not be further decomposed. Finally, goals are explicitly considered
and described and provision is made for them to be associated with
specific interaction events and with agents.

1A company may appear as an agent to a customer. However, in a system study it
may be useful to break down the compound company-agent into atomic agents
such as salesman, accountant, foreman, etc.
Interaction Framework does not deal with interaction protocols but with interaction events. In doing so both good and bad interactions are equally describable. This is useful in that it allows a wide range of the potential actions of agents to be considered and the consequential potential effects on the system to be estimated. The framework does assume the actions are linked to goals so that non-purposeful actions are not easily explained within Interaction Framework. However, for most systems random actions on the part of agents are not a major consideration. As it stands Interaction Framework would seem to be a very useful tool for system modeling.

3.2.2.2.3. Data flow diagrams
Data flow diagrams (DFDs) have been in use as a system description tool for many years. Their focus is on the identification of specific flows between specific locations or processes. As such they may provide useful data on what each process must be able to handle in terms of input and produce in terms of output. Unfortunately they do not contain any data on the sequence of flows nor do they contain any other temporal data. They are not thought to be useful for present purposes but have been included as an indication that traditional methods of systems analysis are not appropriate to the problem.

3.2.3. Temporal logics and notations
From the outset a general temporal logic would seem not to be sufficient to identify what an agent needs to know. While explicitly expressing temporal features such as duration, start/end, parallel activity, conditional execution of processes, etc. they are not designed to provide knowledge or goal descriptions, or to deal with exchanges between agents where the temporal component may be irrelevant. For example it may be much more important to know the content of an exchange between agents than to know when this occurred. What is required is a temporal logic which has been designed not just to
describe the temporal aspects of system interactions but which also copes with non-temporal aspects of interactions.

3.2.3.1. Executable User Action Notation (XUAN)
Gray et al (Gray et al, 1994) discuss the effects that system responsiveness or pace has on user behaviour in interacting with systems. Informed by the results of Teal & Rudnicky (Teal & Rudnicky, 1992) they conclude that "...a system which has a consistent logical behaviour but varying temporal behaviour will produce variations in user behaviour." (Gray et al, 1994, pg 302). They also report that in systems with fast responses users are more prepared to experiment without full consideration than in slow systems. Slow systems seem to encourage more reflective user interactions, greater care is taken over deciding which actions to execute. Three issues are not considered by Gray et al (Gray et al, 1994): i) the cost of making a mistake; ii) how long it takes users to learn what the speed of the system is and; iii) the stability of learning. These issues have implications for the map of user knowledge within a particular system but are not considered further here.

Putting aside the above omissions, XUAN does have some very useful notation which allows the temporal components of systems and situations to be recorded and logically manipulated (Gray et al, 1994; McGowan, 1995). Briefly, it supports parallel, sequential and conditional activities. The inclusion of notation for parallel activities implicitly supports multi-agent situations, though descriptions of agents as such are not explicitly part of XUAN. However, Gray et al do consider the notion, though without definition, when they use the term "task agents". That XUAN is applicable to multi-agent systems can be seen from the statement that "...tasks are viewed as collaborative enterprises involving one or more people (‘users’) and one or more computer systems." (Gray et al, 1994, pg 304, original emphasis).

It is interesting to note that Gray et al conclude that by considering user and system function in the same way, rather than as system
reaction to user action, we get symmetry between computer and user, “task agents” in their terms. Blandford et al (Blandford et al, 1994) see such symmetry as desirable if a system is to be considered as a group of interacting agents. As a result, Interaction Framework has been designed to be agent neutral, not differentiating between one type of agent and another. In this respect both Interaction Framework and XUAN are consistent with a systems theory view where interacting processes can be considered as synonymous with interacting agents.

3.2.4. Cognitive modelling
Cognitive modelling may be carried out for a number of purposes. In the field of expert systems the knowledge base may be based on the cognition of one or more human experts. For this to be the case it is necessary to model the cognition of those experts. In using an expert system, or other complex computer systems it may be beneficial to use a model of the users’ cognition to inform the design process so that functionality and usability are acceptable. The concern here is more with the second purpose of cognitive modelling than the first. At this point it would be useful to know if there are cognitive modelling approaches which will help us to understand what it is that a user of a SMART system must know.

In this section three cognitive modelling methods will form the main basis for discussion:

- Programmable User Models, PUMs (Young et al, 1989),
- Goals, Operators, Methods & Selection, GOMS (Card, Moran & Newell, 1983)
- Abstract Reasoning on the basis of Concrete Clues, ARCC (Decortis et al, 1991).

The first two schemes, PUMs and GOMS, are generalisable to a wide range of situations, but neither explicitly considers time factors; the third, ARCC, does.
3.2.4.1. Programmable User Models, (PUMs)
The PUMs approach, as described by Young, Blandford and others\(^2\) is intended to help with system design. By exploring the interactions between models of devices and models of users it is possible to identify design problems which may be resolved prior to implementation. A central part of the model of the user is identifying the knowledge a user gains from the explicit display of information, e.g. a visible text string in a word processor and that gained from “tracking”, e.g. knowing the contents of a copy/cut & paste buffer (Blandford & Young, 1995). This concentration on what a user knows, either by display or by tracking, is very important to the present work and played some part in its motivation. However, the PUMs approach has, so far, only considered systems in which real-time has not been made an issue and where there is one user agent and one computer agent\(^3\).

Blandford and Young (Blandford & Young, 1995) describe their model of the users problem solving capability in terms of both the knowledge the user needs and mechanisms by which the knowledge is applied. A central mechanism is means-ends analysis. This is essentially a backtracking mechanism used to identify a chain of goals to be attained in order that the needs of the highest level goal are met. The details of this mechanism are not an issue here, but the knowledge required by the user is. The following items of user knowledge have been derived from Blandford and Young (ibid.):

---


\(^3\)A real-time multi-agent system has been considered (Blandford and Young, ref. WP25) but the effects of multi-agency and real-time were removed. Though the domain was air traffic control, which may be considered as an example of a SMART system, the approach taken was to consider an individual air traffic controller (ATCO) interacting with a single display or workstation. The focus was on the information needed by and provided to the ATCO. Time windows and the effects of the actions of other agents were not considered in that study.
• current state of the macro system,
• desired state of the macro system,
• difference between current and desired state,
• whether or not the [ top level ] goal has been achieved,
• the action(s) necessary to achieve a given goal,
• whether or not an action is executable at the current time point,
• current visible state of the device,
• current tracked state of the device,
• the sub-goals necessary to satisfy a goal not currently achievable through available actions;

in addition some problem solving heuristics are suggested where a choice between courses of action can be made no other way:

• where several actions are possible choose that which is relevant to the current goal,
• where there are several actions relevant to the current goal choose that which reduces the most important difference between current and goal states,
• prefer actions which do not undo previous actions\(^4\) [ unless this is required for error correction\(^5\)].

We should also perhaps make some other items explicit:

• the user is able to read and interpret the device display,
• the user is able to recognise error states as error states.

\(^4\)This heuristic is similar to the Interaction Framework concern that event trajectories should not contain detours.
\(^5\)This has been added to allow for minor device malfunctions or misuse. This might be the case where a mechanical component has failed ( e.g. rejecting the coins input to a vending machine so that a fresh attempt may be made ) or where the user has incorrectly used a command ( e.g. put the wrong text into a cut/copy buffer when word processing ).
Once a model of the user has been constructed, i.e. user knowledge and heuristics defined, it can be run against the designer’s intended procedure for using the device. In doing this the modeller is able to see if additional knowledge is required or if the modelled knowledge fails to get the desired result. This provides information on both the model of the user and on the designer's intended procedure - either may require modification as the result of a test run. However, the purpose of PUMming is to improve the design process, i.e. to provide designers with information which will be used to improve the device interface or the device side of user-device interaction.

3.2.4.2. Resources Model (RM)
The resources model identifies a number of information resources that a user may require to support a range of interaction strategies (Wright et al, 1996). The information resources described are:

- plans (action list or sequence needed to achieve a goal)
- goals (describes the desired state)
- action affordance (set of possible [next] actions)
- interaction history (list of actions already taken)
- action-effect mapping (statement of the effect of an action)
- current system state (collected relevant attributes of objects)

Not all information resources are required for all interactions. Fields et al (Fields et al, 1996b) identify a number of interaction strategies and link these to various sub-sets of the previously identified information resources:

- plan following - plan, system state;
- planning - goals, system state, action-effects, affordances;
- semantic matching - goal, system state, action-effects, affordances;
- goal-directed exploration - goals, action history, system state, action-effects, affordances;
- learning by exploring - action history, system state, affordances.
The purpose of Resources Model analysis is to support system design. If a given interaction strategy is to be available to a user then the information resources necessary for the strategy must be available as a prerequisite. Having identified the required resources designers are able to consider the attributes of each resource (Fields et al, 1996b):

- distribution (location: user's head / interface [elsewhere]);
- explicitness (how the resource is expressed);
- accessibility (work needed during interaction to acquire a resource in usable form).

The purpose of the Resources Model is to provide a useful framework for designers to specify some features of the system that will support specific interaction strategies. In taking this approach a level of performance is implicit, though the Resources Model is not concerned with precise levels of performance of the kind that may be identified in usability engineering (Whiteside et al, 1988). It is also explicitly stated (Fields et al, 1996b) that in developing the Resources Model the focus is on how computer agents can be designed to support given users rather than an agent neutral approach that also allows user knowledge to be specified and considered as something that may be changed or developed. Though supporting system specification is the aim of the Resources Model, this is achieved through an understanding of the cognition of the user. In this sense the effort in applying the Resources Model is directed towards understanding the user, though the outcome is system oriented.

3.2.4.3. Goals, Operators, Methods & Selection, (GOMS)
A useful and concise review of GOMS (Card et al, 1983) is provided by Johnson (Johnson, 1992). A simplified model of human cognition, with an information processing perspective, is embodied in "a model human processor (MHP)". The MHP describes human long and short term memory capacities, perceptual and motor mechanisms and cognitive processing. Links between these sub-systems are also
described so that there is flow between information received and a motor response as the result of intervening cognitive activity. Knowing the performance capabilities of the various sub-systems, or being able to make informed guesses, enables overall performance of a user in a particular situation, to be estimated / predicted.

The GOMS approach focuses on the structure of the cognitive sub-system of the MHP. The components of this sub-system are initially described by Johnson (ibid.) as follows:

"A user's cognitive structure is assumed to consist of four components: a set of goals, a set of operators, a set of methods for achieving the goals and a set of selection rules for choosing among competing methods for goals." (ibid., pp 128-129)

The links between the components are described more fully. However, at this point it is worth considering a brief definition of each. This will serve as a reminder that the focus of this work is no so much on cognitive mechanisms but on knowledge or knowledge structures.

**Goals** are the highest level cognitive construct in the GOMS approach. They give purpose to action and provide a reference point against which the success or otherwise of an action can be checked. Goals may be decomposed into a number of sub-goals when to do so supports easier achievement of the top level goal. Sub-goals, or lower level goals, are more easily linked to some activity which is directly executable. In a given rôle a user must know:

- what the set of appropriate goals is,
- how to achieve them,
- how to recognise when they need to be achieved,
- when they have been achieved, and
- when an attempt to achieve them has failed.
Operators are:

"..elementary perceptual, motor or cognitive acts whose execution is necessary to change any aspect of the user's mental state or to affect the task environment [ normally undertaken to satisfy a goal]." (ibid., p130)

Thus "operation" covers such activities as reading a display message, keying input or performing a calculation respectively. Knowledge definitions are implicit here, for operations to be executable by a user that user must know how they are achieved and the effects that they will have; primary effects, side effects or both.

Methods are the ways that goals are achieved through sequences of goals and operators. These sequences may be conditional so that they can be made contingent on the state of the user and the "task environment". Methods are learned and are not ad hoc generated plans. This is a usefully explicit statement about user knowledge. It provides a statement about prior knowledge and also about what is not needed - the ability to generate plans. However, while this is acceptable for GOMS analysis it may not suit all real-world situations where it may be necessary to generate plans as new and unfamiliar situations arise. GOMS is not intended to cope with such situations but rather it is intended to provide a tool for usability analysis.

Selection rules are required for situations where there is more than one method which may be used to achieve a given goal. Johnson uses a text editing example where the user has a rule such as: if the next

---

6This phrase has been added to make explicit the link between operators and goals.
7The situation where a number of methods may be used to achieve a particular goal is different from that where there are a number of possible methods which may achieve different goals. In the former situation rules are required to select the most appropriate method while in the latter situation it is necessary to choose between competing, and possibly conflicting, goals.
edit is within 3 lines then use cursor keys else use find ( adapted from ibid. pg 131 ). Johnson goes on to say that GOMS was not designed to cope with problem solving or with error situations.

3.2.4.4. Abstract Reasoning on the basis of Concrete Clues, ( ARCC )

Decortis et al ( Decortis et al, 1991 ) start out with the assumption that time is an important issue for dynamic and interactive systems. Users ( 'operators' in their terminology ) therefore need to be able to reason about time. The degree to which this is important will vary between situations. An important factor here will be the degree of automation. Systems which are only semi-automated require more from their operators than those systems which are highly or fully automated. It is likely that there will always be a need for semi-automated systems where the operator is required to maintain some degree of control and therefore require an understanding of that system.

In dynamic systems a user may consider time as an explicit or as an implicit variable ( ibid., pg 54 ). The difference between these two approaches can be considerable. Where time is explicit then external devices such as clocks or automated timers may support the user. In the implicit situation the focus may be on state changes which occur over some period of time which may or may not be known with precision. In the latter case it is changes and the rate of change that need to be known and monitored by a user. Consequently interface support provided to a user will need to be matched to the temporal reasoning requirements.

They develop a wide ranging taxonomy of temporal issues. Then they go on to develop a structure to link the different kinds of cognitive activity of operators in complex situations and accident situations. The aim is to better understand the causes of errors and so design more supportive systems in future. Their concern is with the reasoning process rather than with the knowledge required to interact within a
KNA for SMART Systems  CH3. Approaches to Modelling

SMART system. As such the ARCC method is not directly applicable to solving the current problem. However, some forms of knowledge are a prerequisite for reasoning to take place. While Decortis et al do not present the knowledge required *explicitly* (except for a set of temporal definitions) the summary here is extrapolated from what they do present.

3.2.5. Dialogue analysis

Two different approaches are considered here based on widely different perspectives of dialogue. In the Milan model (De Michelis & Grasso, 1994), considered first, the issue is "commitment" and agreement of communication goals. In the second approach (Furuta & Stotts, 1994) the use of a form of Petri nets is considered as means of modelling multi-agent interaction where a number of rôles may be adopted by the participants.

3.2.5.1. The Milan model

De Michelis & Grasso (De Michelis & Grasso, 1994) have an ethnomethodologically oriented view of the use of language to communicate to achieve results. They focus on personal communications between human agents from a linguistic viewpoint and as such their work is better placed under a linguistic heading than under an ethnomethodological one. Major concerns are with: defining mutual commitment and negotiating commitment. The focus on human-human communication, though it may be computer mediated, does not make the Milan model, as they refer to their approach, suitable for systems where one or more of the significant agents may be non-human. Also by considering the negotiation of commitment as a central issue the approach is made less relevant to SMART systems: i) it does not directly throw light on what a user must know; ii) where one of the agents is not able to negotiate (e.g. a computer agent). In addition the Milan model as presented does not appear to support mathematical description of analysis of interactions making the analysis and description of real-time interactions difficult.
3.2.5.2. Coloured timed Petri nets

The emphasis of Furuta & Stotts (Furuta & Stotts, 1994) is on the use of a variety of Petri-net (coloured timed net, CTN) to describe synchronous protocols. Their chosen demonstration application is that of supporting meetings. While this is a very specialised form of SMART system the approach appears to be generalisable to other SMART situations such as aviation, process control and bureaucracies with real-time components. Furuta & Stotts sum up their approach as follows:

"The Trellis model serves several functions with unified notation framework: it structures shared applications; it synchronises loosely coupled parallel execution threads; it provides a repository for application information; and it provides mechanisms for joint decision making and action." (ibid.)

One significant aspect of CTNs is that they are highly visual. As such they may prove useful for representing SMART interactions. In addition Furuta and Stotts (Furuta & Stotts, 1994, pg 121) claim that their method supports the detection of flaws in a given protocol. This would allow a definition of what a user needs to know with respect to interaction protocol. However, the CTNs as described do not easily or obviously allow agents to be represented except implicitly through the presence or absence of protocol actions applicable to an agent. CTNs were chosen as an example of Petri-net because they have been applied in current CSCW studies. In addition, Furuta and Stotts (ibid. pg. 123) claim that as CTNs are "high-level" nets they are translatable into all other forms of high-level net. On the basis of these two points it seems reasonable to assume that the strengths and weaknesses of CTNs will be representative of high-level Petri-nets in general.

CTNs show protocols. As such they may prove to be a useful component of models of agents. Where agents share a protocol then communication problems can be expected to be lower than when there is no sharing. Where protocols are different we may be able to analyse
the difference to determine the faults we might expect and the way to reduce communication problems and enhance system performance.

Timing is considered explicitly. Transitions within the net have two time variables attached:

- \( t_r \): the minimum time that must pass between enabling and firing.
- \( t_m \): maximum latency: the maximum time passing after enabling before it fires automatically.

They go on to give an example of an untimed net (i.e. one where time is not an issue) where all transitions would be described with the pair: \((0, \bullet)\), i.e. there is no minimum time, \( t_r = 0 \), and they may never fire, \( t_m = \bullet \).

The overall impression of CTNs is that they are useful for analysis and design of the communication aspects of systems by prototyping but leave agents, data and objectives largely unconsidered.

3.2.6. Task analysis
Task analysis is a broad term covering a number of activities guided by different purposes (Shepherd, 1989). The term covers activities such as task decomposition so that low level units of activity or knowledge necessary to complete a task can be identified and appropriate training and selection carried out. It may also be used as a tool for identifying what existing users do or do not know in the execution of patterns of work. The use of task analysis for determining required or existing user knowledge appears to make task analysis a prime candidate for answering the question “what does a user in a SMART system need to know”. Important issues for the application of standard task analysis techniques will be their ability to cope with the demands placed on users by the multi-agent nature of system interactions and the time constraints imposed by real-time factors.
3.2.6.1. Hierarchical Task Analysis (HTA)
Some methods of task analysis do not deal with time constraints explicitly and seem to assume that they do not exist. As an example, Hierarchical Task Analysis as described by Shepherd (Shepherd, 1989) and Carey et al (Carey et al, 1989) breaks down tasks into a hierarchy of sub-tasks. The description of each task allows the structure of the overall task to be shown but does not distinguish between tasks that are time critical and those that are not. One problem described by Shepherd (Shepherd, 1989) is that HTA has a problem dealing with tasks that cannot be explicitly described (the example of changing gear in a car is given\(^8\)). In another of Shepherd’s examples the giving of a lecture is subjected to HTA. In that example a cell is included for the situation where a piece of equipment becomes faulty. The HTA given suggests that if the equipment can be fixed then it should be fixed. What Shepherd does not say is that the time taken for the repair should be taken into consideration. This could lead to a situation where students do not attend a lecture so much as observe a lecturer fixing equipment. This shortcoming of the analysis may be a result of a poorly described example which another practitioner may be able to improve upon. However, given the method’s difficulty in coping with a time critical skill such as changing gear it may be that HTA requires modification or augmentation before it can be applied to the modelling of SMART systems where the explicit representation of time is considered to be a significant issue.

3.2.6.2. Task-Action Grammar (TAG)
Task action grammar (TAG) in the context of task analysis has a number of aims (Payne & Green, 1989). The first is to formally describe the mapping between the task level and the action level. An analysis of the resulting grammar should allow predictions about learnability, types of errors, and user ability to generate task-action

\(^8\)The problem described in the example is describing the “biting point” of the clutch and that this has to be worked out by each individual and becomes an acquired skill rather than one which can be completely procedurally described.
links to be made. The central method is one of formally describing rules. The rule structure is broadly of this form:

if task X is to be achieved then action sequence Y must be done

By collecting together some or all the rules for a system a grammar may be formed. From the grammar the “psychological complexity” (Payne & Green 1989, pg 80) of the mapping may be calculated. The application of TAG analysis to a system would enable the interface to be fully described in the sense that for a given user or agent it should be possible to produce a mapping from all of the required tasks to the actions necessary to achieve those tasks. Another way to consider TAG is that the grammar for any given user is an expression of the knowledge that the user has. As with HTA, time is not considered to be an issue within TAG.

3.2.6.3. Task Analysis for Knowledge Descriptions (TAKD)
The central purpose of task analysis for knowledge descriptions (TAKD) is to identify and represent the knowledge that a user of a system should have (Diaper, 1989a). The representation can be used during requirements specification and/or system evaluation. As such this tool would seem well suited to modelling SMART systems with respect to agent knowledge requirements. The input to TAKD is described by Diaper as being low level data on objects, actions and sequences (Diaper, 1989a, pg 112). From this a task descriptive hierarchy (TDH) is constructed which “generally deals with specific objects” (Diaper, 1989a, pg 118). In the given example a taxonomy of objects is provided without any reference to associated actions. The route through the hierarchy can be used to identify objects and, potentially, the actions needed to branch through the tree. A knowledge representation grammar (KRG) is used to describe routes.

TAKD appears to be a useful way of identifying system objects and ways in which they may be manipulated and how this is understood by users. However, its application to SMART systems seems limited. The
requirement to produce a taxonomy with all end nodes defined would result in a great deal of work for complex multi-agent systems. There may be significant problems for such a methodology which relies heavily on objects when some objects may be transient, i.e. may come into being and out again. Representing this in TAKD may prove to be difficult. It may be appropriate to represent all possible actions and cope with transience in the KRG by building temporal constraint handling into this. This may be too great an addition for TAKD to survive as a usable tool for SMART analysis.

3.2.6.4. Knowledge Analysis of Tasks (KAT)
Knowledge analysis of Tasks (KAT) is a method described by Johnson (Johnson, 1989) to analyse user task knowledge. The knowledge focused on is not what a "user needs to know" but a user's task knowledge structure (TKS). As such it considers what a user has acquired but not what they may need to know to interact with a given system. Knowledge of what a user does already know may be used to inform the design process. A designer armed with an understanding of the knowledge structures possessed by a potential user [set] may use that understanding to match the new design to the users' knowledge. This should have the effect of reducing learning time and reducing the number of errors.

The approach of KAT is the reverse of that adopted here. In KAT the approach is to identify what a user knows so that the target system may be appropriately modified in the design stage. The approach taken here is to describe an existing, or potentially existing system so that what the user needs to know to interact with that system can be identified. The nature of many SMART systems is such that some of the constraints cannot be designed out. For example, real-time may be an unavoidable feature because of real world phenomena such as the weather or the physics of aviation, or as a result of economic pressures to be competitive. This is different to the situation where a software product such as a word processor or project management tool are being designed. KAT also seems well suited to the design of systems which
will automate existing non-automated processes. From this it seems reasonable to conclude that KAT is inappropriate for the analysis of SMART systems.

3.2.6.5. Analysis for Task Object Modelling (ATOM)
Walsh (Walsh, 1989) describes a task analysis method designed to assist with interface specification. It is intended to support structured design methods such as Jackson Structured Design (JSD) and is itself a structured approach. ATOM would also seem to be inappropriate for modelling SMART systems as "...it starts with the specification of the tasks that the user currently undertakes, identifies the subset suitable for embodiment in the proposed system, and proceeds to produce a design that is also easy to use." (Walsh, 1989, pg 187). Again, like KAT above, this approach is better suited to the automation of existing process rather than the analysis of SMART systems.

3.2.6.6. Command Language Grammar (CLG)
Command language grammar (CLG) appears to have potential it "...attempts to separate out the conceptual model of a system from its command language and to show the relationship between them." (HCI Service, 1991). As such it may provide a method for identifying conceptual and operational knowledge needed by a user agent within a SMART system. Johnson (Johnson, 1992) describes the method for describing systems as having three components: conceptual, communication and physical. Temporal issues are not considered.

The conceptual component is comprised of two task levels: description and semantic. These are interpreted here as describing system purpose, goals and attributes of high level objects such as agents or significant system components. The communication component has two levels: syntactic and interaction. The syntactic level is concerned with descriptions of systems states and the commands and their arguments.

9 It seems from this description of the purpose of ATOM that automation is unlikely to add new tasks. This seems to be rather restrictive on the scope of automation and unrealistic for many situations.
applicable in each state. The interaction level describes the actions required, at a high level, that are required to execute commands. Both the conceptual and communication components seem to have something to offer to the SMART system modelling process. The last component, the physical, is very low level and is concerned with such issues as device layout.
3.3 Comparing Methods within the Domain of HCI

A number of scales or dimensions have been devised relevant to the broad domain of HCI. The purpose of these is to provide a background to the more targeted subsequent analysis of methods potentially useful for supporting KNA of SMART systems. The general picture will provide an appreciation of the range of issues that methods address and their relative strengths and weaknesses. The attribution of scale values to methods is based on a subjective analysis of each method. The scales are:

- **formal - pragmatic**: This scale indicates each method’s approach to describing systems. A formal method (score 0 or 1) will make use of algebraic techniques to describe systems and to manipulate knowledge of them. Pragmatic methods are more concerned with the down-to-earth issues, prosaic details, required to make an actual system work.

- **device focus - user focus**: some methods may focus on the design of the hardware-software device while others focus more on the user; some explicitly aim to be neutral.

- **cognition - performance**: where models consider user activity they may vary in how they approach this. Some are concerned with a user’s cognitive activity and internal constructs and may provide support for modelling these. Others are more concerned with the user’s visible performance (regardless of internal cognition).

- **small - large grain**: methods concerned with small grain are those that model low level device operations such as mouse clicks. Large grain methods are concerned with
high level goals and tasks rather than individual actions. A number of methods have a wide grain span.

- **descriptive - enumerative**: this scale describes the extent to which a method either describes a system or aims to measure or enumerate the performance of the system.

Two other scales were also considered but were rejected as being of less significance than the other scales. This also reduced the number of dimensions that needed to be represented in the method space, so aiding clarity. The two rejected scales were:

- **process - structure**: simply put this refers to the extent to which a method focuses on the way in which a system operates or the way its components are constructed, assembled and connected.

- **states - transitions**: some modelling methods focus on the state of a system at a given point while others are more concerned with how a system moves from one state to another. It is important to remember that here "the system" refers to the whole human-technology system.

### 3.3.1. Rating system
Brun and Beaudouin-Lafon (Brun and Beaudouin-Lafon, 1995) attempt to devise a scheme for classifying descriptive formalisms applicable to interactive systems. Their aim in doing this is to provide a reliable tool for classifying the formalisms so that the deficiencies of the set of formalisms may be clarified. This is intended to provide a starting point for the development of a formalism which will be generally applicable to the design of interactive systems and that does not also have the deficiencies Brun and Beaudouin-Lafon identify in the others. Their classification scheme is based on three views: cognitive science, calculus theory and theory of categories. The application of the
classification scheme leads to a taxonomy of formalisms. In addition to creating a taxonomy Brun and Beaudouin-Lafon (ibid.) also use a four point scale for noting how each formalism matches up against 12 criteria. Their scale is eccentric in that the points are not evenly spaced:

- "empty": the criterion is not supported by the formalism;
- "quarter full": the criterion is not properly supported by the formalism;
- "half full": the criterion is partially supported by the formalism; or
- "full": the criterion is extensively supported by the formalism."

(ibid. pg 203)

In the conclusions to their paper (ibid.) they accept that the taxonomy, the criteria, and the range of formalisms studied could be extended. In effect they present an approach to analysing how a set of formalisms meet a particular set of designers needs. The method of Brun and Beaudouin-Lafon is based on subjective-analytic assessment of formalisms against a predetermined set of criteria. Their approach to the creation of a loose set of criteria against which modelling methods may be measured seems appropriate here and describes the approach taken in this section and in section 3.3. A five point scale has been adopted (0-4). This allows extremes, a neutral point and off-centre, non-extreme positions to be scored. Though it may be argued that this approach lacks formal rigour, it is economic and gives a quick method of describing the attributes of methods relative to one another. It is not intended that the application of numbers supports scalar arithmetic. A method scoring "4" does not meet the criteria twice as well as one scoring "2", but it can be said to score considerably better, i.e. it meets many more of the components of the criterion under consideration.
3.3.2. Results of range analysis

It is important to note that the methods could be expected to vary in a number of ways. The variety stems, at least in part, from the different aims of each method’s developer(s). Thus it is to be expected that the methods will have different areas of focus to meet their various aims. The extent of this variety is shown in table 3.1 and figure 3.1. That such variety exists should not be surprising, it would be more surprising if a number of methods existed which shared many or all characteristics. This would show the presence of significant levels of redundancy and “reinvention of the wheel”.

<table>
<thead>
<tr>
<th>Ethnography</th>
<th>GRADIENT</th>
<th>If</th>
<th>XUAN</th>
<th>PUMs</th>
<th>RM</th>
<th>GOMS</th>
<th>ARCC</th>
<th>Milan</th>
<th>CTNs</th>
<th>HTA</th>
<th>TAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>2-2</td>
<td>0-3</td>
<td>0-4</td>
<td>0-4</td>
<td>0-4</td>
<td>0-4</td>
<td>2-4</td>
<td>2-4</td>
<td>2-2</td>
<td>2-2</td>
<td>2-2</td>
<td>0-1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Scales rise from 0 to 4 in moving from the first scale term to the second.

Table 3.1: Comparison of a selection of modelling methods

While the tabular form allows some differences to be seen the similarities and differences are more obvious when presented graphically.
Figure 3.1: Space of modelling methods

2 - 3 \{\text{figures show ascribed grain range}\}

Philip J.A. Scown

Page: 94
The space shows both how collections of models are formed and also how models within a group can vary. TAG and HTA are both described in the same Task Analysis text (Diaper, 1989b) and as such may be considered variants of task analysis, yet the space diagram shows that there is considerable variation between them. This general analysis also places Interaction Framework and XUAN quite close within the space. This might be taken as an indication that they would perform similar function. However, this is not always the case, as application of SMART specific criteria will show.
3.4. Developing Criteria for the Domain of SMART Systems

Ideally a single modelling technique would provide a model of a system which allows the question “what does a user need to know....?” to be answered. While some of the requirements are relatively straightforward (e.g. representing goals and objectives) other potential requirements are beyond our present understanding and may always be so (e.g. [perfect] models of agent competence and performance). This suggests that an adequate representation is sufficient, for reasons of pragmatics, rather than a comprehensive representation which would fully model all aspects of all possible SMART systems. An adequate representation will enable us to answer the question but need not do more.

The criteria for selecting a modelling method are directed at finding those methods that will assist in identifying user knowledge needs; other criteria are not required. In some respects a set of criteria themselves answer the question, or could be seen to “beg the question” of what a user needs to know. However, the selected criteria will only provide a high level qualitative framework for identifying a method which will in turn help to identify specific knowledge needs at a low level, i.e. on a system specific basis. An appropriate modelling technique will be applicable to all, or a significant sub-set, of SMART systems and will enable specific key areas of user knowledge to be identified.

The five dimensions used in the production of table 3.1 and figure 3.1 above are not of specific relevance to SMART systems but were selected for their general relevance. As such they may be used to distinguish between modelling methods, but for determining applicability to SMART systems some specific criteria are required. Earlier analysis of some example systems identified a number of significant issues:

Philip J.A. Scown
1) the communication skills of agents,
2) interaction protocols,
3) the locus of control,
4) goals and goal sharing by agents,
5) agents models of other agents,
6) plans,
7) the use of simulation,
8) domain knowledge,
9) time windows,
10) real-time / non-real-time hybridisation.

These issues are not all clearly distinct from one another but taken together indicate a number of common themes that are central to SMART systems:

- communication,
- control,
- goals,
- knowledge,
- time.

The list of themes has the appearance of a (non-exhaustive) list of primitive issues. As such it is not important how individual issues map to themes just that the significant issues are represented within the themes. The following mapping is proposed as satisfying the current need:

<table>
<thead>
<tr>
<th>issue(s)</th>
<th>become</th>
<th>theme</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &amp; 2</td>
<td>3</td>
<td>communications</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>control</td>
</tr>
<tr>
<td>5, 6, 7, 8, 9 &amp; 10</td>
<td>6, 9 &amp; 10</td>
<td>goals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>knowledge</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time</td>
</tr>
</tbody>
</table>
Different interpretations and purposes may give rise to different mappings. That some of the issues are represented more than once is not considered to be significant.

These themes lead us to criteria in a fairly direct way in that any modelling technique which is going to be able to deal with all ten issues must, as a pre-requisite, be able to handle all five of the common themes. For each theme a number of low level method descriptors need to be identified as the themes alone are too broad to be applied directly. The more descriptors apply then the more a modelling method can be said to deal with each theme and the better suited it is to modelling SMART systems. Ratings can be allocated to methods on a theme by theme basis so that they can be compared with each other both within themes and overall. This should allow the best method(s) to be identified along with potential weaknesses that may require correction.

Rating methods objectively is difficult or impossible as it requires subjective interpretation of source documents. This suggests that a fine grain scale, such as a percentage would be unsuitable, as it would imply an accuracy not actually present in any assessment. In view of this a rating method similar to that of Brun and Beaudouin-Lafon (Brun and Beaudouin-Lafon, 1995) but not identical, will be adopted. The scale adopted here has five points, effectively adding a “three quarters full” point to that of Brun and Beaudouin-Lafon (ibid.). The descriptors of each criteria, against which points are allocated, are those thought most relevant to the current problem and are based on previous considerations of a range of SMART systems.

One feature of this set of criteria is that multi-agency does not figure explicitly. This may appear to be an anomaly for a study of modelling methods of multi-agent systems. There are two reasons why this is not an anomaly; one theoretical and one pragmatic. On theoretical grounds the need for consideration of multi-agency does not arise as the focus is on what a single agent needs to know about their environment, i.e. the
rest of the system. Key knowledge areas with respect to other agents are given in the other criteria and include [domain] knowledge which may include knowledge of the behaviour of other agents. On pragmatic grounds the obvious pressure to include a multi-agency criterion is resisted because to require it would eliminate many modelling methods immediately as they are not intended to cope with multi-agency and contain no elements that do.

For each of these themes a number of more detailed descriptors are required. The descriptors selected for each theme are based only on the need to be able to assess knowledge requirements in SMART systems. While each theme could be much more fully described than this suggests to do so would not be economic and would provide unnecessary detail.

3.4.1. Communication criteria
The nature of SMART systems is such that communication between agents is essential. Any appropriate modelling technique must therefore represent communication explicitly if we are to be able to manipulate models to explore communication issues.

For a high score against this criteria the following descriptors should cover the most significant aspects:

- protocol descriptions:
  - which agents may communicate directly,
  - communication mode,
  - syntax,
  - semantics;
- content of communications (individual instances/specific surface structures);
- timing of communications: sequencing and constraints;
- situations requiring / preconditions for communication.
3.4.2. Control criteria
From earlier descriptions of systems we saw that control within a system can be a significant issue. It is necessary to identify agents that have control of a system if system and agent behaviour is to be adequately modelled. This locus of control may be fixed or may vary according to the phase or state that a system is in. This further suggests that a modelling technique needs to be able to clearly identify not only controlling agents, but also system states where these affect the locus of control. As we are looking for a generic modelling technique then provision for state modelling would seem useful.

The term "control" is often used loosely. It is noted here that control may be formal, i.e. absolute, or nominal:

formal control:
no action by other agents may affect the outcome. This is rare except in highly deterministic systems, in human-machine systems nominal control mostly applies;

nominal control:
one agent "instructs" another to perform some function(s). Nominal control exists when the initiating agent's objectives will be achieved if the subordinate action follows the given instructions. However, if the sub-agent fails, due to deliberate/accidental error or breakdown, then the instructing agent's objectives may not be achieved.

While the differences between formal and nominal control are noted here it is not intended that form part of the criteria. The use being made of the criteria does not require the distinction to be made and to do so would require superfluous effort.
For a high score against this criteria the following are required:

- the scope of control of each agent;
- the conditions under which an agent takes/relinquishes control;
- actions applying in each state.

3.4.3. Goal criteria

The view taken here is that goals give systems and the agents within them purpose, and give direction to their actions. While a more complex and complete view may be taken (e.g. Weir, 1984) the view adopted here is considered to be sufficient to meet present needs without unnecessary diversion. In some circumstances agents may share goals (e.g. increase company profitability) while in others they may have opposing goals (e.g. low stock holding vs. low unit cost/high volume purchase quantities). It would be useful then if goals could be linked to agents. In doing this the locus of control may be identified: as a particular goal becomes current then the agent charged with meeting the goal is likely to be in control of the performance of the system.

For a high score against this criteria the following are required:

- a method for describing goal hierarchies;
- connections between goals, tasks and actions;
- goal ownership;
- conditions making goals appropriate or inappropriate.

3.4.4. Knowledge criteria

Agents must have enough domain knowledge to be useful otherwise their rôle as such is questionable. As previously mentioned in dealing with others an agent needs to know the scope of their knowledge not its detailed contents. This may be handled by some high level description of an agent's rôle within a system. As such the knowledge requirements here are for surface knowledge rather than for deep knowledge. Having the necessary knowledge requirements limited in
this way enables them to be applied with reasonable equity to both human and computer agents. Surface knowledge may be represented through syllogisms or through simple predicates. This removes the need to require computer agents to exhibit intelligence and further removes the need for complex cognitive constructs and mechanisms; thus simplifying the criteria.

For a method to achieve a high score against knowledge criteria the following are required:

- descriptions of rôles;
- minimum domain knowledge requirements of each agent,
- links between goals and actions,
- knowledge of rôles of other agents;
- domain and situation specific temporal knowledge requirements,
- durations
- start/stop times
- factors affecting temporal conditions.

The ability to infer temporal knowledge requirements is important and distinct from a method’s temporal reasoning power. The latter is considered below.

3.4.5. Time criteria

Any appropriate modelling method must have a provision for temporal reasoning. This needs to be sufficient to express time constraints and to support temporal algebra. This would be enough to express event timing in absolute terms and relative to other events.

For a high score against this criterion the following are required:

- range of temporal terms ( e.g. before, after, during );
- explicit capacity for logical and arithmetic operations on temporal terms.
3.5. Application of the Criteria: Assessing Method Suitability

Having identified the criteria and some approaches to modelling we are in a position to assess which method(s) are most appropriate to support KNA for SMART systems. For each of the themes a modelling method may, as previously considered, be allocated a rating of “zero” (method does not deal with the theme) to “four” (method deals with it well and covers most or all aspects). A further point related to the issue of scoring is the comparative importance of each of the criteria: should some form of weighting be applied so that high compliance with one criteria can outweigh low performance in one or more other criteria? Two problems have been identified with such an approach. Firstly it would imply that some form of scalar arithmetic could be carried out: a four with one criterion being equivalent, say, to a one and a three in others. Secondly, it would assume that less important criteria could be dispensed with. In that case they would cease to be criteria and would become attributes. For present purposes criteria, not attributes, are required. Thus the most suitable method will be the one that has the highest compliance profile across all criteria. In the hypothetical situation where none of the methods scores in all criteria then a combination of methods is required, or modifications to an existing method so that all criteria are met.

Despite the required presence of all of the criteria for a method to be acceptable and the inappropriateness of weighting, it can be seen that some criteria have greater significance to the issues of knowledge needs analysis for SMART systems. At the most significant level are the representing of time and making knowledge explicit. As the purpose of any suitable method is to support knowledge needs analysis in a real-time environment then any suitable method must score highly against these criteria. The second rank of criteria is comprised of goals and control. The goals criterion is significant for its linking of goals and objectives to the actions of agents. Control is relevant in the domain of SMART systems where there is the potential for control over system performance to move from one agent to another. Communication forms...
the third rank of the criteria. Though communication is essential in a multi-agent system it is not a defining feature of SMART systems; even very simple systems require communication between system components. From these rankings it can be seen that the most suitable methods will score highly against time and knowledge, must score well on goals and control, while communication even though it is the least significant of the criteria should be represented. A high scoring method that scores highest on communication and lowest on time and knowledge would not have a profile appropriate for describing SMART systems.

3.5.1. Measuring ethnography against the criteria
In considering the proposals of Hughes et al (Hughes et al, 1994) it may be that much of the data they revealed may have been identifiable by non-ethnographic human factors specialists. The problems indicated by Hughes et al. seem to stem from design approaches which focus on system function rather than whole system performance. When the whole system is considered then both usability and functionality have to be considered. Such a whole system approach encourages the use of appropriate functional design linked to approaches such as usability engineering (Whiteside et al, 1988; Dillon et al, 1993; Macleod & Rengger, 1993) and the explicit defining of usability (Shackel, 1986) which supports, or even requires, appropriate data collection and proper evaluation.

Shapiro (Shapiro, 1994) reviews the place of ethnomethodology within CSCW. The description of ethnomethodology there suggests that the field provides a counterpoint to theoretical, mathematical and analytical approaches. As such it makes a useful contribution by raising the profile of the social side of multi-agent work and the need to properly consider social aspects. However, it provides only context specific observations which are specifically not transferable. This contrasts with the explicit and transferable rules and guidelines provided by other approaches.
Ethnographic methods have something to say on three of the four main points of the communications criteria. However, all of what is said is “soft” - not susceptible to formal reasoning. On “timing of communications” ethnography has little to say. Timing is not generally a significant issue for social science, real-time even less so. On this basis it seems reasonable to allocate a communications rating of 1. The position for control criteria and for goals and time is similar to that for communications, suggesting a rating of 1 for control, goals and time. Ethnographic methods have very little to say about agent knowledge, either qualitatively or quantitatively: rating 0. Overall profile: 11101 ( respective scores against each of the criteria )

While the ethnomethodological approaches may be useful they do not measure well against the present criteria. While a study using this approach may have something to say about communication, control, etc. in a given situation it is deliberately context specific and so difficult or impossible to generalise from. The models generated are historically descriptive rather than predictive or analytical. For these reasons it seems inappropriate to adopt it for the present purposes.

3.5.2. Application of criteria to system based approaches
Interaction Framework appears to be the most comprehensive of the system based approaches. Though each of the other methods has something to offer they seem very narrow when compared to Interaction Framework. On the basis of this assumption the criteria have only been applied to Interaction Framework as at this point it is expected to achieve the highest score.

On the communications theme Interaction Framework has a lot to offer, though it does not meet all of the criteria for this theme. Strengths are on identifying those agents that communicate, when they communicate and why they communicate; syntax and semantics are not issues for Interaction Framework. The timing of communications and the conditions that require them are well covered through the explicit
inclusion of temporal operators and through the linking of goals to tasks and on to interaction events. A score of 3 (out of four) is allocated.

On control Interaction Framework scores well. It explicitly links agents to actions and actions to goals. The notation allows expression of which agent is active and why, and also allows descriptions of faulty interactions and their consequences: score 4.

The first three points of the criteria for goals are well covered though the fourth, conditions making goals appropriate or inappropriate, is not. A closer inspection of Interaction Framework reveals that the achievability of goals, i.e. whether they are possible or not rather than whether or not they will be met, can be deduced from a number of sources, one of which is the trigger function. If a trigger function can be found whose conditions can be satisfied then there is some support for the goals linked to the events in the trajectory. Thus a goal is possible; if the function’s preconditions are met then the goal will be achieved. Interaction Framework also has notation for describing unachievable objectives: those for which there is no event trajectory that serves them. From this consideration of goals a score of four seems appropriate.

Knowledge requirements of agents is not an area explicitly dealt with by Interaction Framework. However, the roles of agents are well described through their relationships with other agents via the interaction events and trajectories they initiate or are affected by. The explicit connection of events to goals also allows some aspects of agent knowledge to be inferred: agents knowledge of the roles of other agents may be inferred from the definitions of event trajectories and the agents affected by them. In addition, Interaction Framework would permit a particular class of event trajectory, those directed at the gathering of information, to be defined which would permit an agent to learn the role of other agents, a secondary goal, before initiating a trajectory directed at the achievement of a primary goal.
The levels of inference required suggest a score of three against the knowledge criteria.

Interaction Framework can be given a score of four against the time criteria as it explicitly provides a range of temporal terms and functions that can easily be manipulated.

The overall profile for Interaction Framework is: 3, 4, 4, 3, 4, 18 (criteria scores, followed by their sum to provide an indication of the ranking of suitability).

3.5.3 Application of criteria to temporal logics as represented by XUAN

While XUAN is strong in the area of modelling temporal aspects of interactions (time criteria) and moderate on goals it is weak on the other three themes. XUAN has nothing explicit about communication. Only where communication is itself a task does XUAN apply. This makes the handling of communication possible, though possibly a little cumbersome. On these grounds XUAN scores 1 on communication.

XUAN provides “control structures” but these are the usual structures found in branching logics and are not explicitly about agents’ control of situations. Where the modelling of agent control is required mechanisms would need to be constructed to cover each application of the method. Score 1.

Goals are referred to obliquely by allowing for successive decomposition of tasks into sub-tasks. If we allow top level tasks to be considered as equivalent to goals then there is implicit support for a goal hierarchy. Links between goals, tasks and actions exist if the qualitative distinction between these items is ignored, or if XUAN is modified to enable the differences to be represented. Goal ownership is not considered. Conditions making goals appropriate or inappropriate are allowed for in the logical constructs of XUAN if they
may be considered as equivalent to tasks. Against the goals criteria XUAN scores 1.

User knowledge is not an issue for XUAN and its structures do not appear to support deductions about user knowledge: score 0. Temporal reasoning however, is central. Given this focus and the range of constructs provided a score of 4 seems appropriate.

XUAN profile: 1,1,1,0,4,7
3.5.4. Application of criteria to cognitive approaches

3.5.4.1. PUMs
PUMs addresses some aspects of communication well. By pragmatically considering the designer's intended procedure and exchanges of information between a device and a user in some detail a practical protocol description may be generated from which a generalisable protocol could be created. The same pragmatic approach will also describe the specific content of communication and situations requiring communication. That timing is not an issue for PUMs results in an overall communications score of 3.

Control is not an issue for PUMs though inferences may be made from the turn taking nature of the interaction analysis. This analysis should allow the scope of control of each agent, its formal/nominal nature, and actions applicable in each state to be described or describable if required. Score 3.

Goals are explicitly covered and match well against the criteria: score 4.

Knowledge is also explicitly covered by PUMs in all aspects of the criteria except temporal knowledge. For real-time systems this could present a significant difficulty for system modellers; score 3.

There is no explicit handling of time in the PUMs approach. Any temporal issues would need to be dealt with on a case-by-case basis and would be contingent on the subjective assessment of the individual using the approach; score 0.

PUMs profile: 3, 3, 4, 3, 0, 13.

3.5.4.2. Resources Model
Communication between agents is an issue for the Resources Model in two ways. Firstly, the content of communications between the
computer and the user is of direct relevance in that the identification and specification of knowledge sources is the main purpose of the Resources Model. Secondly, the style of interaction, which may be linked or related to a style of communication, is used in identifying knowledge sources. The Resources Model does not deal with low level communication issues such as protocols and specific timing consideration. A score of 3 is allotted for communication.

Control is not a significant issue for the Resources Model, though it is not completely irrelevant. In that the Resources Model is concerned with interaction strategies it can be said that each strategy differs from the others in the balance of control over the interaction. Levels of control can be linked directly to knowledge requirements: an interacting agent with little autonomy or control over the interaction requires little knowledge beyond what action it is next required to execute. Higher levels of autonomy and control require much greater access to knowledge or information. The Resources Model does not concern itself with the details of the shift of control between user and computer. Score 2 for control.

Goals are a component of the Resources Model as they are one of the information resources that it describes. While it does not concern itself with the conditions making goals appropriate or inappropriate it does allow for descriptions of goal hierarchies and, through other information resources, the connection between goals and actions can be made. The goals resource may also make ownership explicit. Given these strengths, and despite the apparent absence of goal conditionality a score of 4 seems appropriate.

The Resources Model is explicitly concerned with the knowledge needs of computer system users. Its aim is to identify what is required to support an interaction and the form that each knowledge source should take. While helping to identify high level knowledge sources it is not concerned with identifying the low-level, detailed knowledge items that a user would need. However, the identification and specification of
such low level knowledge would be a logical next step for designers, the Resources Model providing pointers in that direction. The conditions under which knowledge are needed are described at a high level and are linked to interaction strategies. However, this approach does not support the identification of lower level temporal conditions. A score of 3 is allocated against knowledge.

Time criteria are not a concern of the Resources Model. It is not concerned with making temporal requirements explicit and does not do so. The Resources Model could, in future, be adapted to make it applicable to time critical situation such as air traffic control. However, additional work is required for this to be possible (Fields et al, 1996b). Score 0 against temporal criteria.

Resources Model profile: 3,2,4,3,12.

3.5.4.3. GOMS
The GOMS approach is useful for identifying a minimum knowledge that a user must have in order to interact in an error free way. While temporal variables are a component of GOMS these are restricted to the duration of components necessary for the performance of tasks and are only intended to predict the time a user would take to perform a given task. GOMS would not cope with multi-agency or with real-time without significant modification or augmentation.

GOMS has little to say about communication except on the effects of clarity of syntax and the number syntactic options for a given semantic structure: score 1. Control is not an issue from GOMS, nor is there facility for control data to be inferred: score 0. While goals are an explicit element of GOMS the issues of goal ownership and the conditions making goals (in)appropriate are not dealt with: score 2. Descriptions of an agent’s rôle may be inferred from GOMS structures, but as GOMS is essentially a single agent method it does not easily describe the knowledge that one agent may need of another. Temporal knowledge is not described within GOMS: score 1. Time is only
considered very narrowly: as a user performance measure not something which is inherent to the situation being modelled. This is only a little better than not dealing with time at all, and does not help with multi-agent real-time issues at all: score 0.

GOMS profile: 1,0,3,1,0,5.

3.5.4.4. ARCC
ARCC does not deal with communication as a central aspect of system models but as a means of supporting operators' modelling processes. As such communication is not dealt with coherently. The timing of communications is dealt with as are the conditions requiring communication and the semantic of communication. A score of two seems appropriate as a balance between breadth of cover and lack of coherence.

Although ARCC is applied to process control systems it is not explicitly concerned with the detail of the mechanisms of control. However, some information about control may be inferable from situation descriptions: score 1.

Decortis et al (Decortis et al, 1991) suggest that goals are explicitly dealt with in a modelling module that is linked to ARCC: Fuzzy Goal Oriented Script (FUGOS). There appears to be a range of algorithms and heuristics allowing an operator to select appropriate actions given the activation of certain goals. The focus is on psychological processes through "...similarity matching...and frequency gambling." (ibid. pg 64). From their further descriptions the aspects of goals we are concerned with here appear to be: conditions making goals (in)appropriate and, implicitly, connections between goals, tasks and actions. From their limited description it is difficult to allocate a score higher than 2.
Knowledge requirements of operators are explicitly dealt with and give good coverage of the criteria: score 4. Similarly ARCC explicitly supports temporal reasoning through provision of a temporal knowledge base: score 4.

ARCC profile: 2, 1, 2, 4, 4, 13.

3.5.5. Application of criteria to dialogue analysis

3.5.5.1. Milan model
We would expect a dialogue modelling approach to be strong on communication. De Michelis & Grasso (De Michelis & Grasso, 1994) suggest that the meanings of communication are interpretative, that is, they are listener dependent rather than source dependent. This provides for multiple semantic ascriptions to a single message and allows them to be related to their social context. Syntax as applied to sentence construction or other surface structure, is not considered but the structured "syntax" of conversations is. Content and the conditions requiring communication are explicit issues for the Milan model though timing is not. A high score is suggested even though not all aspects of the communications are explicitly described: score 4.

While control is not explicit in the Milan model some aspects of it can be inferred from its modelling of commitment negotiation. A participating agent may be considered as a "doer" or a "referent" and is able to "accept" or "refuse" a commitment. The process of obtaining commitment from agents is also described: score 3. Goal hierarchies are not considered though agreement on individual goal items is important to the method. Goal ownership is explicit, but connections between goals and actions, and the conditions making goals (in)appropriate are not considered: score 2. Knowledge is not an issue for the Milan model, nor can it easily be inferred from what is included: score 0. The same applies to temporal issues: score 0.

Profile for the Milan (conversational) model: 4, 3, 2, 0, 0, 2.

Philip J.A. Scown

Page: 113
3.5.5.2. Coloured timed Petri-nets
While CTNs are strong in the area of protocols and the timing of communications they do not cover syntax or semantics well, nor do they consider surface structures: score 3. Control is strongly represented through the explicit definition of circumstances under which changes in agent rôle may occur and the scope of each agent at a given node: score 4. Goals are not considered at all but may with effort be inferred from changes in rôle and agent scope: score 1. Knowledge is constrained to issues related to timing and protocol issues. Links between goals and actions and what is known of the rôles of other agents are not described: score 2. Time criteria are only partially met but could easily be extended through the use of methods used in other Petri-net approaches: score 4.

CTN profile: 3, 4, 1, 2, 4, 14.

3.5.6. Application of criteria to task analysis methods
The Task Analysis methods considered above do not, taken as a whole, deal with communication as a distinct issue, though it may be thought of as just one of the many tasks an agent has to do, and can thus be described. As a result the way that communication is modelled will be dependent on the Task Analyst rather than on the inherent descriptive powers of Task Analysis with respect to communication: score 1. Control is not considered except as a part of a task: score 0. Goals may be considered as belonging to the top most node level in a task hierarchy. As such Task Analysis does provide a link between goals and tasks. Goal ownership is not well dealt with as the focus is on single agent analysis, thus goal owner ship would not be an issue: score 2. Knowledge is a central focus of Task Analysis. However, the temporal aspects of knowledge are generally not well described and some specific problems with this have been identified (e.g. Shepherd, 1989): score 3. Timing criteria are not dealt with consistently and it
would seem difficult to do so in Task Analysis, temporal terms are lacking: score 0.

Task Analysis profile: 1, 0, 2, 3, 0, 6.

Task analysis is the last of the methods to be measured against the criteria. The following section brings the assessment together.
3.6. Summarising the Application of the Criteria

Having reviewed the modelling methods and given each a profile against the criteria they are now more easily compared. The total column, as previously discussed, is intended to assist the ranking process rather than to provide a scalar variable supporting numerical analysis. The profiles for each of the methods, in the order they are considered above, are as follows:

<table>
<thead>
<tr>
<th>Total</th>
<th>comms</th>
<th>control</th>
<th>goals</th>
<th>knowledge</th>
<th>time</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>ethnography</td>
<td>not profiled</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>system approaches (I.F.)</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>18</td>
</tr>
<tr>
<td>temporal logic (XUAN)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>cognitive modelling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PUMs</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>RM</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>GOMS</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>ARCC</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>dialogue analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Milan</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>CTNs</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>task analysis</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 3.2: scores of methods against criteria

To ease comparison the data may be sorted on “total”. Ethnography can be removed at this stage as it has not been profiled for reasons given previously.

<table>
<thead>
<tr>
<th></th>
<th>comms</th>
<th>control</th>
<th>goals</th>
<th>knowledge</th>
<th>time</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>GOMS</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>TA</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>XUAN</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Milan</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>RM</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>ARCC</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>PUMs</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>CTNs</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>I.F.</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>18</td>
</tr>
</tbody>
</table>

Table 3.3: sorted scores of assessed methods against criteria
Table 3.3 clearly shows that Interaction Framework has best overall profile; scores for all criteria being at the three and four level. For the top ranked criteria, time and knowledge, Interaction Framework scores four and three respectively. This is only one point worse than the ARCC method. However, on the next rank, goals and control, ARCC scores only two and one respectively, while Interaction Framework scores four and four. On communications the Framework scores four while ARCC scores only two. While Interaction Framework does drop a point to ARCC on one of the two high ranking criteria its overall profile is significantly better. The graphical representation in figure 3.2 makes the advantage even clearer and also shows the relative consistency of the method over the others across the range of criteria. In order that Interaction Framework may be viewed in more detail a quick reference is provided in appendix 3.1.

![Methods X Cumulative Criteria Scores](image)

*figure 3.2: graph of method scores*
3.7. Summary & Conclusions

From a previous examination of a range of SMART systems five themes, sets of criteria, have been identified which are significant for the understanding of any one SMART system. The criteria have been described so that any modelling method in this domain could be subjectively measured against them. Those rated well would be suited to modelling SMART systems and would enable the knowledge that a user within the system requires to be identified. The modelling criteria are both subjectively described and applied. However, this is not a problem as it is only necessary to identify the overall suitability of a method and its strengths/weaknesses relative to other candidate methods.

All of the modelling methods have something to offer; though for present purposes some have more to offer than others. What is required, for ease of use, is a single method, or a very small set of methods, which will allow SMART systems to be described so that what a user needs to know can be determined. To select a method or methods the criteria given early in the chapter have been applied. However, the bald application of the criteria is not enough. In addition, as a means of selecting between overlapping methods, their economy and efficiency in practice should be used in the selection process. This should ensure that the resultant method produces worth while results in an efficient manner even if the descriptions produced are not complete. Effort needs to be balanced against payback. It is believed that the criteria have been applied consistently and that this has lead to the identification of a good candidate method for identifying user knowledge: Interaction Framework.

Interaction Framework is rated well, but not perfectly, across the criteria. In achieving the highest rating Interaction Framework is consistent; all criteria are rated three or four out of a possible maximum of four. The next three highest rated methods have ratings of zero or one against some criteria. While this lack of consistency may
make these methods unsuitable as primary modelling methods it may be the case that they will prove suitable as supplementary methods if Interaction Framework proves to have weaknesses which cannot be overcome through minor alterations. Such potential weaknesses may be identified by applying Interaction Framework to some hypothetical systems based on a number of actual systems.

Given the relative performance of the methods it seems reasonable to propose Interaction Framework as a single method applicable to identifying user knowledge needs. Although it may be possible to use a combination of methods the apparent breadth of Interaction Framework would seem to make the need for a combination unnecessary. However, if after trial application some difficulties remain then there are candidate methods which may supplement Interaction Framework.
Appendix 3.1: Interaction Framework (I.F.) Notation

The I.F. notation given here in the "Quick Reference Guide" has been derived from Blandford et al (1994 & 1995). Some modifications are suggested here and are shown in conventional square brackets for minor changes or in italics for more significant changes.

I.F. - Quick Reference Guide

The analytical headings contained within I.F can be broadly categorised as:

- agent identity;
- objectives;
- event trajectories.

1. Agents

An example based on a computer game with two human players competing using a games computer helps to clarify how agents are viewed:

"A multi-agent system consists of a set of interacting agents $\alpha \in \mathcal{A}$ which defines all the possible patterns of configuration between the atomic agents. So in this system, if the atomic agents are computer-player1 ($\alpha_1$), computer-player2 ($\alpha_2$), computer-environment ($\alpha_3$), user1 ($\alpha_4$) and user2 ($\alpha_5$) then configurations include:

$\mathcal{A} = \{\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5\}$ the set of five atomic agents;

$\mathcal{A}_1 = \{\{\alpha_1,\alpha_2,\alpha_3\}, \alpha_4, \alpha_5\}$ the set of three agents, computer-system, user1 and user2 - here, we consider $\{\alpha_1,\alpha_2,\alpha_3\}$ as a single agent with its own interactive properties;

$\mathcal{A}_2 = \{\{\alpha_1,\alpha_4\},\{\alpha_2,\alpha_5\},\alpha_3\}$ the set of three agents, player1, player2 and computer environment;

$\mathcal{A}_3 = \{\{\alpha_1,\alpha_2,\alpha_3\},\{\alpha_4, \alpha_5\}\}$ two agents, computer-system and users;

$\mathcal{A}_4 = \{\{\alpha_1,\alpha_2,\alpha_3,\alpha_4, \alpha_5\}\}$ one agent."

(Blandford et al 1994)
2. Objectives

In an objective there is: a start condition that is true before the objective is achieved and an end condition that is true after it has been achieved. Blandford et al (1995) define an objective, \( t \), as:

\[ t = (p_0, p_1) \]

where

- \( p_0 \) is true in initial state \( s_0 \),
- \( p_1 \) is true in final state \( s_1 \).

3. Event Trajectories

A trace of events followed by a specified event may be written as \( E = (e_1, e_2, ... e_6 ) > e_7 \). The operator \( > \) ('is followed by') is defined to relate the event \( e \) to all the preceding events in the set \( E \). If two events occur in parallel then we denote it by \( e_1/e_2 \). The comma separator ‘,’ indicates that interaction events may occur in any order and is in contrast with the \( > \) separator which indicates strict ordering / sequencing.

That part of the interaction trajectory that relates to the events that are initiated by agent \( \alpha \) is written as:

\[ (e_1, e_2, ... e_n ) \downarrow \alpha. \]

A significant point arising from the communicative nature of events is that they are defined by the responsiveness of the communicatee not the communicator. Additionally, the approach is agent neutral:

"...our aim is to take a neutral perspective... describing the system from an unprejudicial external viewpoint. For this purpose we use events, which we can characterise as the communications between agents as perceived by a ‘fly on the wall’. An objective is defined as a partially ordered set of events, where the events are generated by communication through action of the agents. Just as agents may be viewed at different levels of abstraction, so objectives may also be viewed at various levels. At the highest level, the entire interaction may be viewed as a single objective. At each successive level of refinement, the interaction consists of increasingly detailed objectives. The ‘lower bound’ is where the objective consists of one observable action by an agent, which is observed and interpreted by another agent in the multi-agent system. We refer to such an atomic objective as an event [figure 1]. At all levels of description, the entities which are defined relate to conjoint behaviour of two or more agents to the interaction." (Blandford et al 1994)
A question that arises from seeing these two organisations of system entities is: is there a relationship between the viewing of events and objectives and the structure of $\mathcal{A}$, the set of sets of agents\textsuperscript{10}? Where $\mathcal{A}$ is complex, i.e. made up of a relatively high number of diverse sets, we might expect difficulty in mapping from the agents to a simple hierarchy of objectives and events. Where $\mathcal{A}$ is simple the mapping would probably be easier. Given that agents may be compound or atomic and, as such, will have world views of different grain size it seems reasonable that events and objectives may be redescribed according to the agent to whom they are relevant. Thus something considered to be an event by one agent may be regarded as an objective by another. An objective for one agent may not even be visible as an event to another.

**Ballistic and Guided Trajectories:**

In exploring the characteristics of event trajectories it may help to consider the viewpoint of ballistic and guided trajectories. A ballistic trajectory is one which, once launched, cannot be further controlled. A guided trajectory can be altered after launching. It should be noted

\textsuperscript{10} The $\mathcal{A}$ notation for the set of sets of agents is an addition to the IF notation given in Blandford et al (1994).
that identifying ballistic and guided trajectories requires the loss of viewpoint neutrality, though this may only be a local issue.

"Similarly, we (Blandford et al 1994) use $E \downarrow \alpha$, to refer to the part of interaction trajectory $E$ restricted so that it contains only events that are initiated by or have destination including $\alpha$.”

In other words $E \downarrow \alpha$ is any part of the trajectory which directly involves some communicative act which $\alpha$ either generates or receives.

In order to talk about event traces (also termed “performances” by Blandford et al (1994) the source and destination functions are introduced:

source: $E \rightarrow A$

where

$E$ is the set of events and

$A$ is the set of agents in the system, and

the source mapping defines which agent in the system initiates the event $e \in E$.

[ Remember that an agent need not necessarily be atomic. ]

We shall define dest: $E \rightarrow A$ where

the dest mapping defines which agent in the system receives and interprets the event $e \in E$.

We shall define $T$ as a set of times (rational numbers), defined at a suitable level of granularity.

time: $E \rightarrow T$ defines the time at which the event begins.

duration: $E \rightarrow T$ defines the duration of an event instance.

The following notation may also be useful:

completion: $E \rightarrow T$ defines the time at which the event ends.

recent: $E \times A \rightarrow T$ defines the time [i.e. start time] of the most recent event involving an agent in $A$. 
In order to maintain consistency the following notation is suggested:

\[ \text{recent: } \mathcal{E} \times \mathcal{A} \xrightarrow{\tau} \text{defines the time (i.e. start time) of the most recent event involving an agent in } \mathcal{A}. \]

\[ \text{recent: } \mathcal{E} \times \mathcal{A} \rightarrow \tau \text{defines the time (i.e. start time) of the most recent event initiated by an agent in } \mathcal{A}. \]

Where \( \mathcal{E} \) is the set of (all possible) event types and \( \mathcal{E} \) is a set of partial orderings of event instances (a set of interaction trajectories) (Blandford et al. 1994).

Agents and objectives can be expressed at different levels of granularity, so that we can consider properties at different levels of abstraction:

\[ \Sigma \in \mathcal{I}, \]

where \( \mathcal{I} \) is the set of objectives supported by the interactive system

and \( \Sigma \) is the set of partially ordered sets of objectives

To recap:

We define a refinement mapping which defines how a high level objective \( t \in \mathcal{I}, \) where \( \mathcal{I} \) is the set of objectives supported by the interactive system and \( \Sigma \) is the set of partially ordered sets of objectives) may be described at the next level of decomposition, as a partially ordered set of lower level objectives or events \( (\mathcal{E} \subseteq \Sigma): \)

\[ \text{decomp } (t) = [u_1, u_2, u_3] \]

and

if the design is such that one objective must precede another then this is written as:

\[ \text{decomp } (t) = u_1 \rightarrow u_2 \]

Typically there will be an intermediate state such that if \( t = <p_0, p_1> \) there will be an intermediate state \( p_i \) such that \( u_1 = <p_0, p_i> \) and \( u_2 = <p_i, p_1> \).

Where an objective may be achieved in more than one way this may be written \[ \text{explicitly] as } \text{decomp } (t) = [u_1, u_2, u_3] \text{ or } \text{decomp } (t) = [v_1, v_2, v_3]. \]

\[ \text{Note that this notation requires explicit expression of every possible series of low level objectives which achieve a higher level objective.} \]
It is suggested that ◦ may be used where ordering does not matter, so that a trajectory may be written as:

\[ E = (e_1 \diamond e_2, e_3...e_6 ) \triangleright e_7 \]

where

\[ E \] is the event trajectory defined by both events \( e_1 \) and \( e_2 \) occurring in any order before \( e_3 \) and where the set of events \( e_1 \) to \( e_6 \) is followed by the terminating event \( e_7 \).

For our sales and manufacturing system above \( E_5 \) now takes the following form:

\[ E_5 = ( E_2 \diamond E_4, \text{invoice produced } ) \triangleright \text{invoice dispatched} \]

A mapping: “which takes a partially ordered event set and transforms a state:

\[ \text{transf: } E \rightarrow (S \leftrightarrow S) \]

Now given an objective \( t = (p_0, p_1) \), we say that an event trace \( E \) achieves the objective if

\[ \text{transf (E, s_0 ) = s_1} \]

for all initial states that satisfy the precondition and final states that satisfy the post condition.”

Blandford et al (1994) go on to say:

An objective may have a number of possible event trajectories associated with it.

We shall refer to a particular trajectory as \( E \rightarrow t \).

A partial achievement of an objective—an event sequence which is engaged in while performing an objective, but which does not achieve the goals of the objective— is denoted by \( E \leftarrow t \). In other words,

\[ E \leftarrow t \text{ if there exists a trace } E' \text{ such that } E \triangleright E' \rightarrow t \]

or, \( E \) leads towards \( t \) but does not reach it if the event trajectory may be connected with another event trajectory which in turn does achieve the objective\(^\text{11}\).

\(^\text{11}\) Blandford et al include the following point: “In general, the relation \( \leftarrow \) will be assumed to be non-strict.” (Blandford et al 1994).
The trigger function

"trigger : \mathcal{E} \times S \rightarrow S"  

given a state \( s_0 \in S \) and an event trajectory \( E \in \mathcal{E} \), the trigger function maps this to the state \( s_1 \in S \), that results if the trajectory \( E \) takes the state of the interactive systems from \( s_0 \) to \( s_1 \)...

\[
  s_1 = \text{trigger} ( E, s_0 )
\]

(Blandford et al, 1995)

If the trigger function allows us to say that if we have a given state, \( s_0 \), then \( E \) is the trigger trajectory for \( s_1 \) OR given \( E \) then \( s_0 \) is the trigger state for \( s_1 \).

So, we have valid states \( s_0 \) and \( s_1 \) and a valid event trajectory \( E \) that gets from \( s_0 \) to \( s_1 \) forward in time. The trigger function maps from a given state and trajectory to another given state.

Thus if any two of the triple ( \( s_0, E, s_1 \) ) are known the third can be identified. Interpreted this way the trigger function becomes the trigger relationship.

4. Formalisation of Properties [ of interaction ]

objective potential

"The objective potential of \( E \) is the set of objectives \( t \in \mathcal{T} \) ( where \( \mathcal{T} \) is the set of objectives the interactive system is designed to support ) such that \( E \) partially achieves \( t \)." (Blandford et al, 1995, emphasis added).

event potential

"Event potential describes the freedom the agent has to choose events that will continue to make progress towards a particular objective. For any \( E \ll t \), the event potential for agent \( \alpha \) is the set of events \( e \) such that \( \text{source}(e) = \alpha \) and \( E \triangleright e \ll t \) - i.e. there is an interaction continuing with event \( e \) that is canonical with respect to \( t \)." (Blandford et al, 1995, emphasis added).
unachievable objective

"If \( t = \langle p_0, p_1 \rangle \) is an objective, and the current state of the multi-agent system is \( s_0 \) (where \( p_0 \) is true in \( s_0 \)) then an objective is unachievable if there is no event ordering \( E \) or state \( s_1 \) with the property that \( \text{trigger}(E, s_0) = s_1 \) where \( p_1 \) is true in \( s_1 \)." (Blandford et al, 1995)

system reset

"Let \( s_0 \) be the state of the agents representing the multi-agent system; \( p_0 \) is true in \( s_0 \). Then for any trajectory \( E \), there is a partially ordered set of events \( F \) such that \( \text{trigger}(E \triangleright F, s_0) = s_f \) where \( p_0 \) is true in \( s_f \). (That is, the relevant predicate on the state of the system after \( E \) and \( F \) have happened is the same as that on the state of the system initially, although other aspects of the final state may differ from the initial state; \( F \) ‘resets’ the state of the system to one in which the same choices are available.)" (Blandford et al, 1995)

simple interactional detour

"\( F \) is a simple interactional detour with respect to objective \( t \) if there exists \( F', F'' \) such that \( F' \triangleright F \triangleright F'' \triangleright t \) and \( F' \triangleright F'' \triangleright t \).

Note that this definition is limited to error-and-repair trajectories that are separable from the main “flow” of the interaction. It does not, for example, cover cases where the interaction is in some other way “longer than necessary”. More generally, any interaction that is not canonical may be viewed as containing a detour." (Blandford et al, 1995)

blind alley interaction

"\( F \) is a blind alley interaction with respect to objective \( t = \langle p_0, p_1 \rangle \) if there is no trajectory \( G \) such that \( F \triangleright G \triangleright t \), that cannot be expressed as \( G = G_1 \triangleright G_2 \) where \( \text{trigger}(F \triangleright G_1, s_0) = s_f \) and \( p_0 \) is true in \( s_f \). I.e. \( t \) is unachievable without an intervening system reset, and may be unachievable even following a reset." (Blandford et al, 1995)

order-constrained events

"If \( E = E_1 \triangleright e \triangleright E_2 \triangleright f \triangleright E_3 \triangleright t \), where \( e \) and \( f \) are both initiated by agent \( a \) and \( E \) is canonical for \( t \), and it is not possible to construct a trajectory \( E_1 \triangleright f \) then the trajectory is order-constrained with respect to \( e \) and \( f \)." (Blandford et al, 1995)
order-independent events

"If \( E = E_1 \cdot e \cdot E_2 \cdot f \cdot E_3 \rightarrow t \), where \( e \) and \( f \) are both initiated by agent \( \alpha \) and \( E \) is canonical for \( t \), and it is possible to construct a trajectory \( E_1 \cdot f \) (i.e. it is possible for event \( f \) to follow \( E_1 \)) then if there exists \( E' \) such that \( E' = E_1 \cdot f \cdot E_2' \cdot e \cdot E_3 \rightarrow t \), and \( E' \) is canonical for \( t \), then the trajectory is order-independent with respect to \( e \) and \( f \) for agent \( \alpha \)." (Blandford et al, 1995)

*It was to provide notation for such situations that the \( \cdot \) notation was introduced.*

premature event

"If \( E = E_1 \cdot e \cdot E_2 \cdot f \cdot E_3 \rightarrow t \), where \( e \) and \( f \) are both initiated by agent \( \alpha \) and \( E \) is canonical for \( t \), and it is possible to construct a trajectory \( E_1 \cdot f \) (i.e. it is possible for event \( f \) to follow \( E_1 \)) but \( E_1 \cdot f \) does not partially achieve \( t \), then the event \( f \) is premature with respect to objective \( t \)." (Blandford et al, 1995)

local agent control

"Agent \( \alpha \) has local control with respect to objective \( t = (p_0, p_1) \) if the events that are initiated by agent \( \alpha \) force the outcome of \( t \), irrespective of the events initiated by other agents to the interaction. This property (referred to as ‘agent invariance’ by Harrison et al (1994)) can be expressed as follows:

for all \( E \) that achieve an objective \( t \), that is \( E \rightarrow t \), then for any \( E' \) such that \( E' \cdot a = E \cdot a \) it is the case that \( E' \rightarrow t \)." (Blandford et al, 1995)

objective-based agent control

"Agent \( \alpha \) has objective-based control with respect to objective \( t = (p_0, p_1) \) if the events that are initiated by agent \( \alpha \) force the outcome of \( t \). In this case, information communicated in events from other agents may be needed to enable \( \alpha \) to force the achievement of \( t \)." (Blandford et al, 1995)
Chapter 4. Using Interaction Framework for Knowledge Needs Analysis

4.1. Introduction

This chapter shows that Interaction Framework can be applied to SMART system modelling and that such modelling supports knowledge needs analysis (KNA). This is achieved by application of Interaction Framework to a system of moderate complexity within the aviation domain. Such an example is intended to show that Interaction Framework has some value in system modelling and that, by extrapolation, it may also be applied to systems of higher and lower levels of complexity. Examples from two other domains, office automation and process control, are also considered as candidates for demonstration purposes. However, for a variety of reasons, they provide less suitable demonstrations than aviation.

A number of stages are used to demonstrate the general usefulness of Interaction Framework. Firstly it must be shown that a single system modelling exercise has enough power to support extrapolation to all or most degrees of system complexity. Secondly, the specified system must be adequately modelled using Interaction Framework. Thirdly, it must be demonstrated that knowledge needs can be inferred from the Interaction Framework description. The structure of this chapter follows this sequence of points.

On the second and third points it is only necessary to demonstrate that Interaction Framework can produce models that support KNA. It is not necessary to exhaustively and completely model a system and show that the exhaustive model supports similarly exhaustive and complete KNA. Such an approach would have required all system options to be considered and would have become unmanageable as a result of combinatorial explosion. Interaction Framework is shown to be applicable to SMART systems, and to support KNA. The supported KNA explicitly relates to “simultaneous multi-agency” and “real-time” features of these systems. It is not claimed that Interaction
Framework can or should be used to assist with KNA outside these areas. Other techniques, such as task analysis (Diaper, 1989b), the resources model (Fields et al, 1996 ab; Wright et al, 1996) and/or programmable user models (Young et al, 1989; Blandford and Young, 1995) could be used to augment the information gained from Interaction Framework relating to non-real-time, non-multi-agent KNA.

In applying Interaction Framework to the aviation domain a number of minor extensions were required. Rather than detracting from Interaction Framework these changes show its flexibility. It is possible to add elements to Interaction Framework without reducing its coherence and without making it unwieldy or unusable.
4.2. Selecting A Representative System.

It is important to identify an appropriate system for demonstration purposes. The selected system must be complex enough to test Interaction Framework. It must also be sufficiently representative of SMART systems to support extrapolation to higher and lower levels of complexity. However, while being sufficiently representative it need not be a complete and accurate representation of any one system.

The nature of SMART systems is that multiple agents are engaged in synchronous and / or asynchronous simultaneous activity in real-time situations. From this we can see that it is necessary for the demonstration system to have at least three agents so that two may be engaged synchronously while at least one other is simultaneously engaged in asynchronous activity. Such a demonstrator system also requires interactions to be established and terminated. It would also allow for activity to take place that is relevant to an agent not directly involved in an interaction but that needs to be monitored or tracked by that agent.

A further consideration is the extent to which real-time factors affect the overall performance of the system. Some systems only have real-time influences over a small part of one phase of activity (e.g. cancelling the printing of a word processed document) while for others real-time is almost continuously an issue (e.g. floor trading at the stock market, aviation, energy production & distribution). A suitable demonstrator ought to have a significant real-time element in order to provide sufficient material to cover a range of issues and to support extrapolation to systems of higher and lower complexity. Thus the selected system will not only be a SMART system but will be representative of all SMART systems. This is necessary if any conclusions drawn from the demonstrator system will apply unambiguously to others.
4.2.1. Candidate systems

A number of candidate systems have been considered from a wide range of domains. For reasons of economy of effort it was thought appropriate to consider the simpler candidates first. Word processing, while not generally a real-time activity, does have some real-time components in the printing phase: there may be a small time window during which printing may be cancelled, otherwise printing may commence and waste both the user's time and consumable resources. From a demonstrator point of view the significance of the real-time factors may be considered to be too small to be representative of SMART systems in general.

A commonly occurring commercial example was considered: credit card sales. In this type of transaction there are often three obvious agents: customer, sales assistant, credit card dial-up service. The dial-up service may be either a single automated agent or, for larger transactions a compound agent comprising a human intermediary and a computerised information / credit scoring system. While the agent structure seems appropriate it is rejected because of poor definition of the real-time factors. Though these do exist they are mainly determined by the limited patience of the people involved. While "user patience" may be a significant issue for system performance and acceptability it is difficult to quantify and is variable within and between individuals. The same arguments may be used to reject help desks in IT Information Centres as potential demonstrator systems.

In the pilot study of the manufacturing organisation a real-time quality control issue was revealed. Unfortunately this section of the system involved only two agents (quality assessor and supplier) making it unsuitable for use as a demonstrator. Other aspects of the organisation that are applicable relate to customer support. In this domain customers make (real-time) telephone calls regarding the specification, price and availability of products. As with the credit card sales system the real-time factors are present but are not well enough defined for demonstrator purposes.
Aviation also provides a range of candidate demonstrator systems. Taken as a whole the domain would be extremely large and complex. Fortunately, it is not necessary to consider the whole domain only a suitable subsection of it, and only a suitable sub-system within that sub-domain.

In the study by Hoc (Hoc, 1989) a process control system is reported in which human and computer agents co-operate to control an industrial process. While this system clearly has at least two agents, the industrial process needs further consideration before its agency can be determined. Real-time factors are an integral part of the process as described by Hoc (ibid.).

The last three candidates from the above set are considered in more detail in the following sub-sections.

4.2.2. Outline descriptions of best candidates

The best three candidates from those considered above were taken from commerce, aviation and manufacturing. Given that there were three to choose from it seemed appropriate to consider each a little more deeply to see which, if any, would stand out as having better demonstrator potential than the others. This was done by expanding on the brief descriptions given above and looking at the relevant features of each against the five themes of SMART systems:

- communication,
- control,
- goals,
- knowledge,
- time.

The ease with which this could be done and the simplicity of each feature was an indication of each candidate's suitability as a demonstrator. A further issue is the comparative ease with which each
may be described or understood; difficulties here would make reasoning and Interaction Framework descriptions more difficult to follow.

4.2.2.1. Customer support
When potential customers make telephone enquiries they require prompt and accurate responses. It is in the supplier's interests to provide such responses as each sale, and continued custom, may depend on the speedy and accurate provision of information. Customer calls are made in real-time, they may be made at any time during the business day\(^1\) and have to be dealt with in a "reasonable" time - one that accords with the customers expectations of the number of rings before pick-up and the subsequent speed with which information is obtained.

Staff providing the information do so with reference to a computerised information system that indicates current product information and stock levels. As production batches are completed the stock levels are increased. This will be in accordance with production schedules - except where problems occur in production. This may mean that stock levels do not rise and fall in accordance with scheduled production and known or regular orders. Information on variations from planned production may be obtained from staff in the production department. Other variations may occur if orders are cancelled. Figure 4.1 is a schematic representation of the communication in this sub-system.

\(^1\)Typically the calls are not evenly distributed but have peaks and troughs depending on the time of day and various seasonal factors.
A typical scenario for this system can be represented as follows:

1) an established customer calls the company;
2) CS staff answer the call and establish the customer’s requirements;
3) CS staff use the information system to determine the availability of stock that would meet the requirements;
4) CS staff offer solutions;
5) customer decides;
6) CS staff complete the formalities.

This is a canonical example. Steps 2) and 3) require prompt and efficient handling by CS staff if customers are to be satisfied. Additionally, in step 3), the information system must be responsive enough to support CS staff. If the typical response time of the IS is slow then it may be difficult to differentiate between a system that is slow and one that is “down”. In addition to the steps of the canonical example it is possible to add parallel asynchronous activity from production who may also need to interact with the information system.
For a satisfactory interaction both customer and CS need to be aware of how long each takes to communicate. Against the communication theme it is observable that if either the potential customer or the individual customer support staff do not know how to conduct business communications by phone then the system is going to fail. In addition, CS staff must also be able to communicate with the information system.

In terms of control, the communication between customer and staff goes through several phases. In initiating the conversation the customer is in control, and is much more able to terminate the conversation if they so wish. Once the information exchange is underway then CS staff may take control as they are able to select the information that they give and may be able to steer the customer towards one product or another or towards ordering a larger quantity, etc. Having exchanged information the customer regains control as it is they that make the final decision about whether or not to buy, assuming that their credit is good.

The customer’s goal, typically, is to order an item that they require to achieve some objective of their own. CS staff have a number of goals: providing accurate information, making a sale and maximising company profits. At some point in the interaction it may become apparent that one or other of these goals cannot be achieved; in that case the agent owning that goal may decide to terminate the interaction. The goals of Production are: optimising production output per unit of cost, and fulfilling the production schedule.

For a customer to place an order they would be expected to know what they want. However, this is not always the case as customers may need some assistance in buying if they have not bought a particular product before or if they are novices. CS staff may also have specialised product knowledge that they are able to bring to bear without which the interaction may fail.

Time plays a part in a number of different ways. CS staff know they must answer the phone promptly if they are to provide a good service.
and pickup the largest number of calls. Answers to queries must be provided as quickly as possible. If it is not possible to provide an answer within a short space of time then the solution needs to be deferred, obtaining the information after the conversation at the earliest possible point. In addition the information systems needs to be responsive and needs to be updated promptly. Information that is out of date should not be used as it may result in poor customer service. Some aspects of the information are updated at routine time points, though there may also be additional irregular updates.

4.2.2.2. Aviation
In the cabin of a typical commercial airliner there may be the Captain, with overall responsibility for the aircraft, who is supported by a Co-pilot. Both are capable of flying the aircraft and will typically share tasks during any one flight. For present purposes the Captain and Co-pilot will be deemed to comprise the flight crew. The flight crew will utilise a range of instruments and sub-systems. Instruments include such items as altimeters and temperature gauges; subsystems include the navigational computer (NavCom) and flight management system (FMS). The instruments can be said to provide environmental information to the flight crew while the NavCom and FMS can be regarded as agents within the system.

Air traffic control officers can provide real-time interrupts to ongoing procedures and interactions or may cause procedures to be undertaken which would not otherwise be undertaken. For example, air traffic control officers may instruct the flight crew to alter course as a result of other traffic problems at an airport, though ultimately responsibility for the aircraft is with the Captain. The structure of this system is summarised in figure 4.2.
In addition to the normal real-time events and operations to be managed there may be a number of abnormal events. As the latter are unexpected, may be critical, and thus require prompt responses they are probably real-time events. A flight from departure point to destination will require climbs and descents as well as a series of turns. These manoeuvres have to carried out within certain time windows if the flight is to be successful: achieve its route objective, be efficient and be safe (not necessarily in that order).

A complete flight would be unnecessarily long and detailed for demonstrator purposes. What was required was a segment of flight activity that involved three or more agents in some real-time activity that would be representative of SMART systems. The following was considered suitable for a demonstrator situation:
1) flight is underway and under the direct control of the NavCom in auto pilot mode;
2) air traffic control (ATC) require the aircraft to alter course (for whatever reason) and so an air traffic control officer issues the instruction to the flight crew;
3) the flight crew need to terminate current tasks, respond to the message from the air traffic control officer and execute the required manoeuvre having taken control from the auto pilot;
4) the flight crew inform the air traffic control officer that the manoeuvre has been carried out and request and set a new course for the remainder of the flight.

For the system to work all agents must be able to communicate with each other. Failure of any element of communication can cause critical problems. While the Captain has overall control within the aircraft, s/he is constrained by law to comply with all instructions from air traffic control officers.

The air traffic control officer and the flight crew share the following goals: maintaining safe flight, getting the aircraft to its destination efficiently. The air traffic control officer also has to optimise the airspace under their control while the flight crew also need to ensure that the flight is comfortable for the passengers. Where the flight has lost time, perhaps due to delayed take-off or adverse weather, the Captain may request a quicker course from the air traffic control officer. However, this is only a request; the flight crew must abide by the air traffic control officer's decision and subsequent instructions.

That "knowledge" is important can be seen from the level of training required for all of the human agents involved and the critical nature of many aviation activities. Air traffic control officers and the flight crew must know the rules regarding safe flight and they must know how to communicate. In addition the flight crew must know about aircraft systems and how to use them. The automated agents, NavCom and FMS, must have embedded "knowledge" about navigation and engine control respectively. The NavCom must also have some knowledge of
engine control, via the FMS, if speed is to be correctly maintained as wind speeds vary or turns are made.\(^2\)

Time is important for a number of reasons: a safety margin is required to keep aircraft apart, fuel supplies are limited, schedules need to be maintained if chaos is to be avoided.\(^3\) Thus there are sound safety and commercial reasons for ensuring that time is properly understood. In addition, there are some particularly time critical phases not considered in the candidate system: take-off and landing.

4.2.2.3. Process control
The process control system considered as a potential demonstrator is described by Hoc (Hoc, 1989): a blast furnace used for the production of iron from iron ore. He proposes that it would be useful for human operators to be supported by an automated assistant, considered here to be a form of agent, that is able to reason about the domain. Such reasoning powers on the part of the Assistant would enable it to “understand” and support the reasoning of the operator and so provide better and more appropriate help (ibid.). The process is real-time, requiring consistent monitoring from the operator and well timed action to be taken to cope with both normal events or a number of possible malfunctions. The blast furnace plant behaves as an agent as:
i) the task of ore production is delegated to it; ii) it is non-deterministic in that performance is variable and a number of malfunctions may occur that need to be dealt with; iii) the plant is indirectly controlled and requires the operator to know operating procedures and to be able to interpret plant instrumentation. See figure 4.3 for a schematic representation.

\(^2\)There is a tendency for an aircraft to lose height as turns are made. Consequently engine speed may need to be increased to increase lift to compensate for this if level flight is to be maintained during a turn.

\(^3\)A completely different air traffic control paradigm is considered by Dekker (Dekker, 1996): management by exception. This aims to support an increase traffic flow by allowing the flight crew of each aircraft to optimise their route. Air traffic control officer's only intervene when a problem has occurred or the situation is such where a problem is likely to occur.
A typical phase in system performance may be:

1) Plant Interface (PI) indicates a reduction in temperature;
2) Operator (O) and Operator's Assistant (OA) observe the indication of temperature reduction;
3) PI indicates a further drop and so establishes a trend;
4) OA indicates possible courses of action involving different combinations of temperature, air flow and raw materials flow along with expected consequences of each combination;
5) O uses knowledge and current goals to make decision;
6) O manipulates PI by changing setting to get desired result;
7) O & OA observe PI to see if results are as expected: reversal of trend at expected rate of change.

In a process such as this communication will typically take place using conventional computer input/output devices and telecommunications. However, the complexity of some aspects of this domain may require some communication to be translated. The Operator's Assistant needs domain knowledge if it is to interpret the current state of the plant and communicate this to the Operator in appropriate terms. Thus
communication in some circumstances is made even more indirect than is usual with computer controlled plant.

In this situation the Operator is in nominal control of the system - there may always be plant failures that are beyond the scope of the Operator to solve. Such failures aside, it is the Operator that issues instructions and decides what action to take in response to system output.

From Hoc's description of the process it seems reasonable to summarise the Operator's goals as follows:

- optimise the flow rate,
- maintain (near) constant flow,
- avoid breakdown situations or sub-optimal performance,
- recover from breakdowns or sub-optimal performance without delay.

The goals of the proposed Operator's Assistant are related to the provision of information that would enable the Operator to achieve these goals.

The process is indirect in two distinct ways. Firstly, the process must be controlled via intermediary mechanisms. Secondly, and perhaps most significantly for the knowledge theme, there are long control latencies in the process, so that an action taken by the Operator may not achieve the desired result for some considerable time. In addition, if a trend such as high temperature is to be reversed, then account must be taken of the tendency for the trend to continue for some time before corrective action begins to take effect, and yet more time before the system reaches its target temperature. Figure 4.4 illustrates this graphically.
Figure 4.4 Graph of hypothetical temperature responses to timely and to late corrective action by blast furnace Operator

One rôle of the Operator's Assistant is to assist with decision making under these long control latency conditions. Apart from knowledge of temporal issues and performance the Operator (and therefore the Operator's Assistant) requires knowledge of the process, its operation and its energy and raw materials requirements. It is the long response latencies in this system that make time a consistently significant issue in this system. In addition, as with many large scale industrial processes, there may be emergency or safety critical situations that require very fast response if injury or damage are to be avoided.

4.2.3. Selecting the demonstrator
The Customer Support situation has the strength of being representative of a large number of commercial organisations. However, it was considered that a worked Customer Support example, using all agents, would probably be too convoluted or contrived as the Production agent interacts relatively infrequently with the others. This could have made the demonstrator example difficult to follow. A more
natural example would use three agents: customer, CS staff and information system. Real-time factors in such a situation are limited to the random timing of calls and the subjective assessment of the speed of other agents. While these are important issues, they would have been difficult to incorporate into a demonstrator in a clear and unambiguous way.

The aviation example, based on the system description above, contains five agents. It also has ample scope for introducing real-time issues. A key problem with aviation could have been limiting complexity whilst maintaining representational validity. This was made possible by restricting the scenario to the execution of a simple manoeuvre typical of those executed in many flights.

The process control scenario has three agents, the smallest number acceptable for a demonstrator. However, real-time situations should be both easy to introduce and to describe. Although the relationships between the physical variables of the system are complex it is not necessary to know what these relationships are in detail, only that a (complex) function exists linking inputs to expected outputs. However, blast furnaces are not familiar objects and may prove to be a little obscure.

At this stage the least promising candidate was the customer support system. Process control and aviation remained as candidates to be selected from. Both systems have multiple agents and scope for loosely and tightly defined real-time factors. Loosely defined factors would include, for example, the need for an aircraft to make a turn to correct for drift. This could be compared with the more tightly defined need to make a turn to avoid a near miss. In the blast furnace example ore input may be increased to compensate for a general temperature increase or reduced to avert a system shutdown resulting from a sudden system blockage. Both cases require prompt action if the desired effects are to be achieved. The description of loosely and tightly defined real-time situations revealed that aviation examples may be easier to describe to a wider audience than process control.
where more specialist knowledge would be useful to make sense of any given situation. This suggested that of the three candidates aviation would make the best demonstrator.
4.3. Modelling an Aviation Scenario

In the previous section an outline of a potential aviation scenario and its agents was given. This involved an instruction for a course change being made by an air traffic control officer and acted upon by the flight crew supported by the NavCom and FMS. In this section the scenario is described in more detail using both plain text and Interaction Framework. The scenario is also modified to explore a range of standard, non-emergency situations as well as examples that are more time critical. KNA issues are considered as they arise. Only some aspects of the demonstrator scenario are described in detail as the intention is to provide sufficient detail to demonstrate the value of Interaction Framework and the KNA gains to be made without making the description too unwieldy.

4.3.1. External validation of the aviation demonstrator

The version of the aviation sub-system described here is not the first version produced. A preliminary version was produced and subsequently submitted for review to a Human Factors Specialist with responsibility for air traffic control[^4]. Changes were made in the light of that review. These were primarily in two areas. Firstly, the highest level of authority in the original was placed with the Captain while in the revised version it is with the air traffic control officer. This authority is encoded in aviation law; the flight crew must respond to air traffic control officer instructions (in the original these were "requests"). It is noted here that while the air traffic control officer does, legally, have the highest authority an example of the Captain overriding this authority was given in chapter two. The reason was to ensure that a passenger who had been taken ill could receive appropriate medical attention.

Secondly, on a more minor point, the use of the term engine management system (EMS) was used in the original. This has been changed to flight management system (FMS). It was suggested by the

[^4]: The external validator is a Human Factors Specialist with the Defence Research Agency (DRA), ATC Systems Division.
external validator that FMS is more appropriate to commercial airliners, while EMS may be used with reference to systems in smaller private aircraft.

The external validator of the scenario also made comments on protocol issues. The interaction event e2 (flight crew returns acknowledgement) would not normally be required in the formal sense of being a statement of intention to comply. Once an instruction has been issued by an air traffic control officer it is assumed that it will be carried out. However, the flight crew will acknowledge that they have heard the instruction. Typically this will be by a simple utterance, e.g. “Roger control” or “Roger, turn left ninety” in response to an instruction to make a left turn. For this reason e2 is still required though some of the underlying semantics are different to those behind its inclusion in the first version.

The term “interrupt” is used with respect to an air traffic control officer instruction to the flight crew. The undertones associated with the term are not generally considered appropriate when the air traffic control officer has overall authority and where the flight crew are expecting, at more or less unspecified times, instructions from the air traffic control officer. However, the term is retained in the demonstrator as it is felt to represent the reality and nature of the relevant interaction events from an Interaction Framework perspective.

The external validator also provided other information that was relevant to the domain of aviation. However, that information did not apply to the demonstrator and so required no additional changes to the demonstrator or detailed responses here. No problems were indicated with interactions within the aircraft, i.e. between the flight crew, NavCom and FMS agents.

4.3.2. Plain text description of aviation scenario
In section 4.2 above, a representative scenario was presented in four steps. Having chosen aviation as a suitable demonstrator it was
appropriate to expand the four steps in greater detail and to augment them to show reasonable variations that provide a more complete test of the suitability of Interaction Framework for modelling SMART systems. The steps describing the scenario are:

s1) flight is underway and under the direct control of the NavCom in auto pilot mode (darker aircraft at bottom of figure 4.5 on heading H1);

s2) air traffic control require the aircraft to alter course (for whatever reason) and so an air traffic control officer (ATCO) issues the instruction to change headings to the flight crew;

s3) the Captain or Co-pilot acknowledges the instruction;

s4) auto pilot is deactivated;

s5) FMS is set to obtain correct speed for turn;

s6) FMS controls fuel and other inputs to engine to obtain the correct speed;

s7) turn executed until required heading reached;

s8) FMS is set to obtain correct speed for straight and level flight;

s9) crew member requests and receives a new flight path from ATCO;

s10) FMS continues to control fuel and other inputs to engine to obtain the correct speed for the new heading;

s11) new direction set on NavCom.
It is easy at this stage to draw some comparisons between the steps here, and interaction events and event trajectories that could be described in Interaction Framework. However, as yet the goals pertaining to the demonstrator have not been considered. The step-by-step listing is presented before goals as it is thought that the scenario will be easier to understand this way around, rather than presenting both the concrete and abstract goals first and attempting to show how these generate interaction events. Some high level goals, such as getting from departure point to destination, and profitable operation are not included here as part of the demonstrator as they pertain to the whole flight, or to the whole commercial operation in the case of the profit goal.
A non-exhaustive list of the high levels goals applicable to the demonstrator (and the agents to whom they are of most importance) contains the following:

- g1) maintain the rules of safe flight\(^5\) at all times (all agents except FMS);
- g2) reach destination efficiently (Captain, Co-pilot, NavCom);
- g3) optimise throughput of traffic through airspace (ATCO);
- g4) maintain passenger comfort (Captain, Co-pilot, NavCom);
- g5) cause aircraft to change course to solve air traffic control problem (ATCO).

With this level of detail modelling can begin. More detailed temporal factors are included as the Interaction Framework description of the demonstrator is developed.

4.3.3. Interaction Framework description of aviation scenarios

In using Interaction Framework, the approach taken is to work from step descriptions and goals to objectives and interaction events, through to event trajectories. This allows a description to be built up gradually and iteratively if necessary, e.g. if it should be necessary to introduce another interaction event or system variation. A straightforward scenario is described first followed by a number of alternatives included to show a wider range of Interaction Framework features. Support for KNA is noted as interaction events are analysed.

---

\(^5\) Examples of such rules: aircraft should maintain a minimum vertical separation of 1000 feet; a five mile horizontal separation should be maintained; aircraft approaching head on should both turn to the right (although there may be circumstances that prevent such a manoeuvre and require a left turn instead. These rules do not effect the FMS though other safety issues such as airframe stress limits may be significant to that agent.
4.3.3.1. Describing Interaction Framework objectives

Here the goals $g_1 - g_5$ are converted to objectives with defined start and end states:

$g_1)$

$\Rightarrow t_1 :$

$P_0$ (flight journey to be made)

$P_1$ (flight journey made, rules of safe flight not contravened)

$g_2)$

$\Rightarrow t_2 :$

reach destination *efficiently*

$P_0$ (route decision to be made)

$P_1$ (decision made, fuel use & flying time minimised consistent with $t_1$)

$g_3)$

$\Rightarrow t_3 :$

optimise throughput of traffic through airspace

$P_0$ (multiple aircraft route decisions to be made)

$P_1$ (all relevant aircraft on determined routes consistent with $t_2$)

$g_4)$

$\Rightarrow t_4 :$

maintain passenger comfort

$P_0$ (control decision to be made)

$P_1$ (aircraft manoeuvred within limits of comfort, consistent with $t_1$)

$g_5)$

$\Rightarrow t_5 :$

cause aircraft to change course to solve air traffic control problem

$P_0$ (aircraft on heading $H_1$)

$P_1$ (aircraft on heading $H_2$)

In the current scenario the only one of these goals to cause action to be taken, i.e. that has the causing of *specified* change as its purpose rather than maintaining constancy, is $g_5$. The others may be seen as constraints rather than goals. However, the existence of these constraint-goals provides a set of objectives to be achieved - albeit negative in outlook. Where choices exist then these goals should affect the decision.
Implicit in $g_5$ is a higher level goal owned by air traffic control:

$$g_5' \Rightarrow t_5': \begin{align*}
p_0 & \text{ (air traffic problem exists)} 
p_1 & \text{ (air traffic problem does not exist)}
\end{align*}$$

In order to progress to lower level objectives and on to interaction events attention is first focused on $g_5$. This appears to be the goal most directly linked to positive action and the demonstrator scenario: the initial and final states of $g_5$ correspond to the first and last steps of the demonstrator scenario ($S_1$ and $S_{11}$). To describe the scenario using Interaction Framework requires the decomposition of $t_5$ into successively lower level objectives until each may be achieved by an event trajectory describable in terms of interaction events. In this case only one further level is required containing four objectives:

- $t_{5.1}$ instruction for turn communicated by ATCO to flight crew
- $t_{5.2}$ turn executed
- $t_{5.3}$ new flight path communicated by ATCO
- $t_{5.4}$ new direction set

This can be shown as a hierarchy; only partially completed as the interaction events have not yet been identified (figure 4.6).

![Hierarchy diagram](image)

**figure 4.6**: partially completed objective and interaction event hierarchy
4.3.2.2 Interaction events and KNA for a straightforward scenario

For the objectives to be achieved it is necessary for there to be a series of interaction events linking $e_1$ to $e_n$ through a series of interaction trajectories. These, and some related KNA points, are described in following sections.

4.3.2.2.1 Interaction events for a straightforward scenario

The following event trajectories and interaction events describe the scenario:

$E_1$

$e_1$  ATCO transmits route change instruction to flight crew
$e_2$  flight crew returns acknowledgement

$t_{5.1}$ achieved $\{\text{instruction to turn communicated to flight crew}\}$

$E_2$

$e_3$  Captain de-activates auto pilot
$e_4$  Captain sets engine speed for turn
$e_5$  FMS controls engine speed for turn
$e_6$  Captain executes turn
$e_7$  Captain sets engine speed for straight and level flight
$e_8$  FMS controls engine speed for straight and level flight

$t_{5.2}$ achieved $\{\text{turn executed}\}$

$E_3$

$e_9$  Co-pilot transmits new course request to ATCO
$e_{10}$  ATCO returns new course instruction to flight crew
$e_{11}$  flight crew returns acknowledgement

$t_{5.3}$ achieved $\{\text{new flight path communicated to air crew}\}$
KNA for SMART Systems

CH.4 Testing I.F. Through Use

E4

e12 Captain sets engine speed for straight and level flight

e13 FMS maintains engine speed for new direction

e14 Captain interacts with NavCom to set new course

t5.4 achieved { new direction set }...

... t5 achieved { aircraft was on heading H1 and is now on heading H2, and prepared for H3 }

The above event trajectories, E1 to E4, are intended to be both simple and manageable while being representative: an aircraft is requested to change course (E1), it changes course (E2), information on further course changes is requested (E3), and maintaining the new course (E4). In this form the Interaction Framework description does not provide any new and useful information that would help with knowledge needs analysis. To be useful the addition of descriptions relating to the timing of events and actions and the parallel/sequential nature of some events is required. A manageable way to add this detail is to consider each interaction event individually:

e1 ATCO transmits route change instruction to flight crew

The air traffic control officer becomes aware of a need for a route change. Having determined what the change should be it is communicated to the flight crew. Remember that the interaction event e1 is not deemed to have occurred until it is received and correctly interpreted by the flight crew. In addition it should be noted that the success of the communication is not known by the air traffic control officer until e2 has taken place.

(1) completion (e1) = time(e1) + duration(e1) < time(e2)
where "completion" is used to signify the end time of an event and "time" is used to signify the start time. Thus $e_2$ cannot start until $e_1$ is complete.

(a1) source ( $e_1$ ) $\rightarrow$ ATCO

(a2) dest ( $e_1$ ) $\rightarrow$ flight crew

Statements (a1) and (a2) relate to agent involvement with the interaction event: the originator or source of the event is the air traffic control officer, while it is intended for the (compound) agent "flight crew". The event $e_1$ would not be appropriate for any other agents in the given circumstances and so no other sources or destination options need be considered.

$e_2$ flight crew returns acknowledgement

The semantics of the communication in $e_2$ are defined by those of $e_1$. For $e_2$ to be an acknowledgement for $e_1$, i.e. to meet its purpose, its content must be appropriate to that in $e_1$. If that is not the case then the air traffic control officer may reasonable assume that the attempt at $e_1$ has failed and needs re-transmitting. The completion of $e_2$ achieves the first minor objective.

(2) $E_1 = < e_1 \triangleright e_2 > \leftarrow t_{5,1}$ \{ instruction to turn communicated to flight crew \}

(3) $T_{t_{5,1}} = \text{completion}( e_2 ) = \text{completion}( e_1 ) + \text{duration}( e_2 ) = \text{etc.}$

where $T_{t_{5,1}}$ is the time that objective $t_{5,1}$ is achieved
The time at which objective \( t_{5.1} \) is achieved can be calculated in a number of ways provided that enough is known of the start and end times or durations of the interaction events in an event trajectory.

(a3) \( \text{source (e2)} \rightarrow \text{flight crew} \)

(a4) \( \text{dest (e2)} \rightarrow \text{ATCO} \)

Statements (a2) and (a3) are not equally specific in terms of agent identity. When the air traffic control officer sends the message it is sent to the flight crew, which member receives it does not matter; the interaction event is successful if either or both receives it. In (a3) the effect, for the air traffic control officer, is that the flight crew have acknowledged the message and therefore it has been, in some sense, successful. However, the message has to be sent by a member of the flight crew in that only one may transmit the message. So, at first glance it may appear that the use of the term "flight crew" in (a2) and (a3) has two different meanings and could, therefore, be ambiguous. In fact the ambiguity is not really present - within this scenario it has been assumed that the Captain and the Co-pilot both have to agree acknowledgement and compliance and are thus both occupied in the event. In such circumstances though one may operate the radio equipment the flight crew may be considered as a single agent. This assumption could be considered as arbitrary, in that an alternative assumption is possible: the Co-pilot transmits the response while the Captain begins to carry out the manoeuvre as a parallel activity. The more complex issue of parallel activity is returned to later.

KNA: Statements (1) and (a3) together also suggest that for \( e_2 \) to be successful the flight crew must know that \( e_1 \) is complete. If this is not known then time(\( e_2 \)) is likely to be too early or
too late. This may create a need for e1 to be repeated, so reducing the efficiency and reliability of the interaction and making the goals g2, g3 and g5 more difficult to achieve.

**e3**

**Captain de-activates auto pilot**

Comparing e3 with e1 and e2 reveals a structural difference between the events: e2 is essentially an explicit acknowledgement of e1 while in e3 the acknowledgement is implicit. This is appropriate because for e3 the NavCom is behaving as a deterministic device with acknowledgement as a determined output. This can be compared to interactions such as e1 and e2 where the response of the flight crew agent is non-deterministic. During auto pilot tasks (i.e. after e3) the NavCom will be non-deterministic in that it will be responding to inputs not provided by the flight crew and thus, potentially, not known to them.

\[(4) \quad \text{time}(e_3) > \text{completion } (e_2)\]

**KNA:** e3 cannot be started until after t5.1 (flight crew successfully instructed to turn) has been achieved.

However, this only gives the relative temporal positions of e2 & e3 and says nothing about the actual commencement and completion times. These actual times may need to be generated. It is possible that they may be inferred from i) the high level goals:

- g1) maintain the rules of safe flight at all times;
- g3) optimise throughput of traffic through airspace;

and, ii) from the possible speed of performance of the flight crew and NavCom agents with respect to completing the tasks necessary to support e3.
KNA: In other words the start time of \( e_3 \) should be prompt to support \( g_1 \) and \( g_3 \) and the completion time is determined by the task performance of the agents involved. This could be used to identify flight crew training needs and NavCom design objectives and so can be seen to assist with knowledge needs analysis. As the interaction needs to occur promptly the NavCom interface needs to be such that speedy interaction is facilitated. The combination of flight crew training and NavCom interface design needs to make the extraction of information about the state of the autopilot component quick and reliable.

\[
E_2 = \langle e_3 \triangleright e_4 \triangleright e_5 \triangleright e_6 \triangleright e_7 \triangleright e_8 \rangle^{t_{5.2}}
\]

\( e_3 \) is not an objective but only the first interaction event in the trajectory terminating with the achievement of \( t_{5.2} \).

KNA: The important performance point is that the objective \( t_{5.2} \) is to be achieved quickly; no one interaction event is as important as this objective. The importance of the current objective and the higher level objective \( t_5 \) suggest that the interaction events should be carried out as quickly as possible while complying with safety related goals.

Returning to the issue of temporal values for \( e_3 \), the start time is given in (4) above while the completion time can now be approximated from the ( variable ) performance ( abbreviated to “perf.” ) of the agents involved. In the case where their respective tasks are performed serially...

\[
\text{(6) completion (} e_3 \text{) } = \text{time (} e_3 \text{) } + \text{perf. (Captain, } e_3 \text{)} + \text{perf. (NavCom, } e_3 \text{)} = \text{time (} e_3 \text{) } + \text{duration (} e_3 \text{)}
\]
Similar statements may be produced for all interaction events in an event trajectory so that, given knowledge of the performance limits of agents or tools, the performance limits for a complete event trajectory may be known. By anticipating some interaction events it can be seen that the expected time for the satisfaction of the terminating objective may be found. For E2 this is:

\[
(7) \text{completion (e8)} = \text{time (e8)} + \text{perf. (FMS, e8)} + \text{completion (e7)} = \text{completion (e7)} + \text{duration (e8)} = Tt_{5,2} \{ \text{turn executed} \}
\]

Thus statements (6) and (7) provide a simple arithmetical means of calculating the estimated time of achievement of an objective. The more precisely performances are known the more precise will be the calculation.

Returning to the current interaction event, e3.....

KNA: (a5) recent(Captain) → time (e3)

The time of the (start of) the most recent event initiated by the Captain is given by the start time of e3. Similar statements for other agents are:

(a6) recent(flight crew) → time (e2)

(a7) recent(Co-pilot) → unknown

Note that though the Captain is an atomic agent, a member of the compound agent "flight crew" this does not mean that events initiated by the Captain must be synonymous with events initiated by the flight crew, or vice versa. The rôles of these agents may overlap but are not equivalent. It is
possible for the flight crew agent, as viewed by the air traffic control officer, to transmit a response while the Captain carries out some other task.

KNA: The agent related statement (a7) states that the start time of the most recent event initiated by the Co-pilot is unknown. Though the Co-pilot is involved in e2 we cannot say from the description if the event was initiated by the Co-pilot or not. Thus we cannot say with certainty:

(a8) recent(Co-pilot) △ time (e2)

We could make the following statements:

(a9) recent (flight crew) → time (e2),

from the air traffic control officer's perspective:

(a10) source (e2) → flight crew

and from the Captain / Co-pilot's individual perspectives:

(a11) source (e2) → Captain (... hypothetically)

This suggests that perspective may be an issue for multi-agent systems where agents may be simultaneously engaged. If perspective is not taken into account in this way then potentially useful information may be lost. To avoid the loss of information that may occur where perspective is not taken into account the following additions to Interaction Framework are proposed:

(a10') ATCO ↦ (e2) → flight crew

where ↦ is used to mean view or viewpoint and
(a11') Captain, Co-pilot $\rightarrow$ (e2) $\rightarrow$ Captain

(...hypothetically)

Further, where an agent is compound this could be made explicit in the notation:

$$\text{flight crew}^2$$

where, for simplicity, the digit showing the number of compound agents need not always be shown. Explicitly showing the compound nature of some agents has the advantage that while that agent may be involved in an interaction event this does not mean that all of the atomic agents of which it is composed are, taken individually, involved in that same interaction. Indeed, some may not be involved in that interaction but in another.

KNA: Returning to the main flow of the event trajectory... For KNA purposes (a8) may tell designers and other agents something about how the Co-pilot is occupied and that particular agent's availability for supporting interaction events initiated by other agents. To be sure on this point we would need a corresponding statement giving the completion time of the last event involving the Co-pilot:

(a12) \[
\text{completion(Co-pilot)} = \text{completion}(e_2)
\]

The time of completion of the last interaction event involving the Co-pilot is given by the completion time of $e_2$. While this may not be especially useful information for the interaction events so far, awareness of such issues may be absolutely essential when two agents are competing for the resources of a third. In such situations overload and consequent error situations may be predicted. Interruptions are dealt with more fully in a later section.
Captain sets engine speed for turn

In this element of E2 the Captain interacts with the FMS to establish the air speed for the aircraft through the turn. This is structurally similar to e3 and generates similar statements for the Captain and for the FMS.

\[(8) \text{ time}(e_4) > \text{ completion}(e_3)\]

\[(9) \text{ completion}(e_4) = \text{ time}(e_4) + \text{ perf.}(\text{Captain}, e_4) + \text{ perf.}(\text{FMS}, e_4) = \text{ time}(e_4) + \text{ duration}(e_4)\]

(a13) recent(FMS) → unknown

KNA: This is the first interaction event, within the scenario, that involves the FMS so the time of the last event initiated by the FMS is unknown, but will be some time prior to the start of the demonstration scenario. We can identify the start time of the most recent event involving the FMS agent:

(a14) recent(FMS) = time(e_4)

(a15) source(e_4) → Captain

FMS controls engine speed for turn

The FMS controls the aircraft systems to maintain the required speed for the manoeuvre and the prevailing conditions. That this has occurred is relayed back to the flight crew via FMS and other instrumentation (e.g. engine revolutions and air speed) and by direct observation of events (e.g. changing tone of engine noise). This feedback
is required so that the agent receiving the information is aware of the changes that have occurred. In this case the recipient will be the flight crew.

The duration of e5 will depend on the performance of FMS, flight crew and the sub systems required to actuate the change (e.g. engines, fuel pumps and control systems). It is possible that the completion of e5 will be anticipated and that e6 may commence before the required engine speed has been achieved. In such a case there will be a degree of concurrency between e5 and e6 and the agents involved will be simultaneously engaged in activity:

(10) \( \text{time(e}_6\) > \text{time(e}_5\) \\

and, in this case...

(11) \( \text{completion(e}_6\) > \text{completion(e}_5\) \\

This allows for e5 to be fully or partially completed prior to the commencement of e6. If it is necessary for e5 to be completed before the start of e6 then we also need to write:

(12) \( \text{time(e}_6\) > \text{completion(e}_5\) \\

If (12) is true then (10) is also true, but not necessarily the reverse.

KNA: Statement (12) may be used for KNA where the engine speed must have reached its target speed prior to the commencement of the turn. However, where this is not the case, i.e. where engine speed response latency is known (implicitly or explicitly) and can be anticipated so allowing earlier execution of the turn, then it is necessary to define the amount of anticipation that is both acceptable (with reference to goals g1, g2 and g4) and that...
supports the objective (t_{5.2}). Again, this supports KNA by making temporal requirements explicit.

Where e_5 and e_6 can be carried out in parallel then this can be noted:

(13) \[ E_2 = < e_3 \triangleright e_4 \triangleright e_5//e_6 \triangleright e_7 \triangleright e_8 >^\dagger t_{5.2} \]

Where e_5 and e_6 can be carried out in any order then this can be written as:

(14) \[ E_2 = < e_3 \triangleright e_4 \triangleright e_5, e_6 \triangleright e_7 \triangleright e_8 >^\dagger t_{5.2} \]

A more complex possibility also exists; where e_5 and e_6 may commence in any order and then run parallel until e_7 commences this may be written as:

(14') \[ E_2 = < e_3 \triangleright e_4 \triangleright e_5\triangleright//e_6 \triangleright e_7 \triangleright e_8 >^\dagger t_{5.2} \]

If it were also the case that the two events could finish in any order then \(\dagger//\dagger\) notation is suggested in place of \(\dagger/\dagger\).

**e_6 Captain executes turn**

The Captain uses the directional controls (e.g. control yolk / joystick) of the aircraft to cause the aircraft to turn through the required number of degrees. In doing so the Co-pilot and NavCom are able to observe the Captain's actions via a number of channels: through various instruments and by observation of the changed attitude and direction of the aircraft. The turn may also be observable by the air traffic control officer via radar.
e7  Captain sets engine speed for straight and level flight

Structurally similar interaction event to e4. The turn having been completed the Captain needs to return air speed to that required for straight and level flight. To achieve this the Captain interacts directly with the FMS....

\[(15) \text{time}(e7) > \text{completion}(e6)\]

\[(16) \text{completion}(e7) = \text{time}(e6) + \text{perf. (Captain, e6)} + \text{perf. (FMS, e6)} = \text{time}(e6) + \text{duration}(e6)\]

e8  FMS controls engine speed for straight and level flight

Structurally similar interaction event to e5. However, e8 terminates E2 with the achievement of t5.2 \{ turn executed \}. Statements (7) and (9) from above (assuming e5 and e6, to be strictly sequential, otherwise select (14) or (14')) :

\[(7) \quad E2 = < e3 \triangleright e4 \triangleright e5 \triangleright e6 \triangleright e7 \triangleright e8 \triangleright t5.2 \{ \text{turn executed} \}\]

and

\[(9) \quad \text{completion}(e8) = (\text{time}(e8) + \text{performance (FMS, e8)} + \text{completion}(e7)) = Tt5.2 \{ \text{turn executed} \}\]

While the purpose of the FMS is to control low level systems to ensure appropriate engine performance the
interaction event is directed to the flight crew to ensure that they know that the FMS and sub-systems under its control are functioning correctly.

(a16) source(e8) → FMS

(a17) FMS ↦ source(e8) → flight crew*

Co-pilot transmits new course request to ATCO

For illustrative purposes a change of active agent is introduced here. While the Captain has initiated several of the previous interactions the present interaction event is initiated by the Co-pilot. Note that the Captain does not request that the Co-pilot obtain the information, the Co-pilot has tracked events and having seen that the turn is completed the Co-pilot is aware that the information is required.

(17) time(e9) > completion(e2)

KNA: This interaction event may not be required - the air traffic control officer may provide new course information without it being requested. In such a situation the air traffic control officer will have tracked events and / or anticipated them - making this a virtual interaction event. Such events are further considered in a later section.

(a18) flight crew* ↦ source(e9) → Co-pilot

(a19) dest(e9) → ATCO

(a20) ATCO ↦ source(e9) → flight crew*

(a21) recent (Captain) = time(e7)
(a22) completion (Captain) = completion (e7)
--comment on the Captain's
--availability, useful for KNA.

e_{10} \text{ ATCO returns new course information}

Having received a request for course information the air
traffic control officer provides it. In the scenario the start
of this event is dependant on the termination of the
preceding event:

\( (18) \ \text{time}(e_{10}) > \text{completion}(e_9) \)

and

(a23) source(e_{10}) \rightarrow \text{ATCO}

(a24) ATCO \leftrightarrow \text{dest}(e_{10}) \rightarrow \text{flight crew*}

\text{e}_{11} \text{ flight crew returns acknowledgement}

Structurally similar to e_2; and leads to the achievement of
an objective:

\( (19) \ E_3 = \langle e_9 \rangle, e_{10} \rangle, e_{11} \rangle \uparrow_{t} t_{5.3} \)
{new flight path communicated to flight crew}

(a25) ATCO \leftrightarrow \text{source}(e_{10}) \rightarrow \text{flight crew*}

(a26) dest(e_{10}) \rightarrow \text{ATCO}
e₁₂  Captain sets engine speed for straight and level flight

Structurally similar to e₄; significant statements are:

(20) time(e₁₂) > completion (e₁₁)

(21) completion (e₁₂) = time (e₁₁) + perf. (Captain, e₁₁) + perf.(FMS, e₁₁)
    = time (e₁₁) + duration (e₁₁)
    = Tₜ₅.₃ + duration (e₁₁)
    = etc.

(a27) source(e₁₂) → Captain

(a28) dest(e₁₂) → FMS

e₁₃  FMS maintains engine speed for new direction

(22) time(e₁₃) > time (e₁₂)

(23) time (e₁₃) > completion (e₁₂)

(24) completion (e₁₃) > completion (e₁₂)

One observation about this event, and others like it involving the FMS, is that while it terminates after the preceding event the time of termination is determined by the interruption of an as yet unspecified interaction event. The duration of e₁₃ may be quite short or long and is determined by the requirements of other agents.
e_{14} \hspace{1cm} \textbf{Captain interacts with NavCom to set new course}

The Captain interacts with NavCom to establish the new course and engages the auto pilot on that course. This can only be done after the new course has been received by the air crew and after the turn has been executed:

\[
(25) \quad \text{time (e}_{14}\text{)} > \text{completion (e}_{10}\text{)} \\
\quad \text{and} \\
(26) \quad \text{time (e}_{14}\text{)} > \text{completion (e}_{8}\text{)}
\]

This event also terminates the event trajectory and achieves the higher level objective supported by the trajectory:

\[
(26) \quad E_4 = < e_{12} \triangleright e_{13} \triangleright e_{14} >^{-1} t_{5,4} \\
\quad \{ \text{new direction set ( H3 )} \}
\]

\[
= < e_{12} \triangleright e_{13} \triangleright e_{14} >^{-1} t_{5} \\
\quad \{ \text{aircraft was on heading H1} \\
\quad \text{and is now on heading H2, and} \\
\quad \text{prepared for H3 } \}
\]

\[
(a29) \quad \text{source}(e_{14}) \rightarrow \text{Captain}
\]

\[
(a30) \quad \text{dest}(e_{14}) \rightarrow \text{NavCom}
\]

4.3.2.2.2 some KNA issues for the straightforward scenario
A number of interaction events require one agent to interrupt the activity of another. At e_{4} the issue of agent interruptability can be introduced. The reason for not introducing interruptions earlier ( e.g. at e_{1} ) is that e_{4} involves the FMS and is thus perhaps simpler to discuss than the previous interaction event, e_{3}, involving the NavCom agent or one involving human agents. The FMS controls the engines with well defined responsibilities and relatively directly: engine speed is raised or lowered and fuel is pumped to meet consequent demand.
As such an agent the FMS is easily interruptable in the sense that requests for engine speed changes may be made at almost any time. Other agents may not be interruptable, or may not respond immediately to an interaction event as they have other responsibilities that may not be interrupted. This raises the question of how to represent the interruptability of an agent? It is suggested here that the interruptability is context dependent, i.e. whether or not an agent may be interrupted depends on the activity that the agent is engaged in at the point of potential interruption. More precisely, interruptability depends on the consequences of halting or changing the activity. While this applies to the demonstrator it may not apply to other systems where an agent may lock out interruptions by disabling input mechanisms or channels.

In a generalised situation there may be two agents, one engaged in ongoing activity, the other that wishes or needs to interrupt that activity in order to meet their own objective. Both are part way through event trajectories leading towards the objectives that each agent supports:

\[ \alpha_r \text{ is the recipient of the interruption} \]
\[ \alpha_i \text{ is the interrupting agent} \]

recent ( \( \alpha_r \)) \( \uparrow \) \( e_r \) --the start time of the most recent event
--involving \( \alpha_r \) is by the start of \( e_r \)

\[ \mathbb{E}_r = < \ldots e_r, \ldots > \uparrow t_r \]
--\( e_r \) is part of an event trajectory satisfying \( t_r \)

recent ( \( \alpha_i \)) \( \uparrow \) \( e_i \)

\[ \mathbb{E}_i = < \ldots e_i, \ldots > \uparrow t_i \]

source ( \( e_i \)) \( \rightarrow \) \( \alpha_i \)
--\( \alpha_i \) is the source of the interrupting event
completion ( e₁ ) < completion ( eᵣ )

--the interrupting event finishes before the ongoing event, or the time that the event would have finished without the interruption.

One observation that follows from this description is that the two agents have two objectives tᵣ & tᵰ and these may or may not be the same, or of the same priority. In the aviation scenario there is a hierarchy of goals that are supported by different objectives. Examples of potentially conflicting goals of different priority are (repeated from section 4.3.1 above):

\[ \begin{align*}
g₁ & \text{ maintain the rules of safe flight at all times (all agents except FMS);} \\
g₂ & \text{ reach destination efficiently (Captain, Co-pilot, NavCom);} \\
g₃ & \text{ optimise throughput of traffic through airspace (ATCO).} \\
\end{align*} \]

It is not necessary to give these goals detailed consideration to see that g₁ is of higher priority than both g₂ and g₃ taken individually or together. It can also be seen that g₂ and g₃ may conflict. In the demonstrator scenario we would expect that the air traffic control officer would request a course change for sound reasons, i.e. in order to achieve some valid objective. It would be of some help to the air traffic control officer if they were able to be certain that their objective in making the change is of higher priority than the current objective of the flight crew, or does not impede the attainment of higher objectives. How can Interaction Framework represent these issues?

The attainability of objectives, and thence of goals, may be represented by Interaction Framework in a number of ways to reflect different situations. Sometimes objectives may be delayed, while at other times they may become unachievable. Interaction Framework expresses these possibilities in the following ways (taken from “I.F. Quick Reference Guide”, appendix to chapter 3):
unachievable objective

"If t = < p_0, p_1> is an objective, and the current state of the multi-agent system is s_0 (where p_0 is true in s_0) then an objective is unachievable if there is no event ordering E or state s_1 with the property that trigger(E, s_0) = s_1 where p_1 is true in s_1."

( Blandford et al, 1995 )

The definition of an "unachievable objective" makes it explicit that an objective is not tied to a specific event trajectory is such a way that if an event in the trajectory fails then the objective cannot be achieved. There may be several event trajectories - all of which fail. Another way of viewing this definition is that it states that where there is an unachievable objective any attempt to backward chain from the objective to the current state via a series of interaction events will fail - no continuous route can be found either by simple interactional detour or by system reset (both of which are further considered below). In the demonstrator scenario this might occur if, for example, time constraints, agent unavailability or communication problems prevent the completion of the necessary event trajectories for the intended manoeuvre.

Simple interactional detour

"F is a simple interactional detour with respect to objective t if there exists F', F'' such that F'\triangleright F \triangleright F'' \triangleright t and F'\triangleright F'' \triangleright t."

Note that this definition is limited to error-and-repair trajectories that are separable from the main "flow" of the interaction. It does not, for example, cover cases where the interaction is in some other way "longer than necessary". More generally, any interaction that is not canonical may be viewed as containing a detour.”

( Blandford et al, 1995 )

With a "simple interactional detour" an objective may still be attainable after an interruption - but the effective event trajectory will be longer than would otherwise be necessary. Where an interactional detour occurs in a real-time situation then time constraints need to be
considered. It is easy to conceive of situations such that a simple interactional detour occurs where there is no constraint but that when a time constraint is introduced the original objective becomes unachievable or becomes a blind alley interaction (see below). This may be the case where \( p_1 \), the end state of the objective, includes a requirement for the change to have occur before a specific time.

When considering interactional detours a distinction needs to be made between increasing the number of interaction events and increasing the duration of the trajectory. The definition of a "simple interactional detour" given above is concerned with the number of events in a trajectory. While this may give some indication of expected cognitive load on human agents the effects on duration are implicit. The temporal duration of a trajectory with a detour must be greater than one without. In a time constrained situation it is not enough to consider whether a series of interaction events, taken one by one, can reach an objective; the time taken for each event and the time available must be taken into account. A delay in any event trajectory may cause it to become unachievable; in the case of a detour the unachievability becomes more likely just because it must be of longer duration due to the additional events over and above those in a non-detour trajectory.

This raises the issue of trajectory abandonment. Once it becomes clear that a given trajectory has an unachievable objective then abandonment of that trajectory must be considered. In some other situations it may be that no current objective would be achieved by continuing with the first trajectory and that another must be identified and pursued. However, it is possible that a trajectory may support more than one objective so that it should be continued despite certain failure of one or more of those objectives. This will depend on the relative and absolute importance of each objective associated with the trajectory. Where an objective is only unattainable if the present trajectory is persevered with then it would be appropriate to consider a simple interaction detour or a more complex detour, that of a "system reset".
It could be said that the demonstrator scenario is itself an example of interaction detour in that the rerouting of the aircraft was not intended at the start of the journey. Additional interaction events were required to achieve the "safe flight" objective. This makes explicit the order of precedence of objectives: safe flight is more important than economic flight.

**system reset**

"Let $s_0$ be the state of the agents representing the multi-agent system; $p_0$ is true in $s_0$. Then for any trajectory $E$, there is a partially ordered set of events $F$ such that $\text{trigger}(E \triangleright F, s_0) = s_f$ where $p_0$ is true in $s_f$. (That is, the relevant predicate on the state of the system after $E$ and $F$ have happened is the same as that on the state of the system initially, although other aspects of the final state may differ from the initial state; $F$ 'resets' the state of the system to one in which the same choices are available.)" (ibid.)

This bears some resemblance to a simple detour. However, there are many situations in SMART systems where the same choices are *not* always available following a detour; aviation is one. An aircraft that is put into a stack, perhaps awaiting a suitable landing slot, uses fuel so that after a period of time it is not possible to divert to another airport, or indeed, to keep the aircraft circling in a stack for the same period of time as was previously possible! Thus, as time passes fewer options remain open to the flight crew and to the air traffic control officer and system resets are rarely possible. In the demonstrator scenario the aircraft is diverted from its original course. This effects the options available for the remainder of the flight as extra fuel is consumed and the heading of the aircraft is altered. These effects can be countered by ensuring that the aircraft is carrying considerable excess fuel over that required for the planned route. This reduces the cost efficiency of the flight and makes explicit the conflict between safety and economy goals. Failure to carry extra fuel may require that provision for "system reset" be made. An analogy for aviation would be for the aircraft to land and refuel. In either case the flight crew* needs to be made aware by the FMS of the limits of fuel supplies.
Blind alley interaction

"F is a blind alley interaction with respect to objective \( t = \langle p_0, p_1 \rangle \) if there is no trajectory \( G \) such that \( F \rightarrow G \xrightarrow{t} \), that cannot be expressed as \( G = G_1 \rightarrow G_2 \) where \( \text{trigger}(F \rightarrow G_1, s_0) = s_f \) and \( p_0 \) is true in \( s_f \). I.e. \( t \) is unachievable without an intervening system reset, and may be unachievable even following a reset." (Blandford et al, 1995)

If the interruption causes an interaction event involving \( \alpha_r \) to be abnormally terminated then it is possible that the event trajectory to which it belonged becomes a blind alley interaction, i.e. the objective \( t_r \) has become blocked, achievement is prevented. Also, if \( \alpha_r \) is not aware of this then the agent may pursue activity that cannot achieve the desired objective. This suggests that in systems where interrupts may result in blind alley interactions there should be some mechanism for enabling agents to identify them as soon as possible. This is particularly true in real-time systems where continuance with a blind-alley interaction may prevent other objectives from being achieved. In the demonstrator scenario this might occur if the diversion took the aircraft into a position where its destination could not be reached directly. Air traffic control officer and flight crew* agents would benefit from knowing this prior to deciding on the change of heading. If it is not possible to know this before the change of course then it needs to be known soon after so that a new destination objective may be set.

Local agent control

"Agent \( \alpha \) has local control with respect to objective \( t = \langle p_0, p_1 \rangle \) if the events that are initiated by agent \( \alpha \) force the outcome of \( t \), irrespective of the events initiated by other agents to the interaction. This property (referred to as 'agent invariance' by Harrison et al (1994)) can be expressed as follows:
for all $E$ that achieve an objective $t$, that is $E \leftarrow t$, then for any $E'$ including events from other agents such that $E \downarrow \alpha = E' \downarrow \alpha$, it is the case that $E' \leftarrow t$ and $\alpha$ has local [i.e. local formal] control.

(adapted from Blandford et al, 1995)

In an interruption $\alpha_i$ has control if their objective $t_i$ is achieved regardless of the events initiated by $\alpha_r$. However, this does not adequately describe possibilities such as the termination of $E_r$ by an interruption or the possibility for convergence of $E_r$ & $E_i$. Trajectory convergence can be said to exist when:

$$E_1 = <e_1...e_n> \leftarrow t_1$$

$$E_2 = <e'_1...e'_n> \leftarrow t_2$$

$E_1 \downarrow \alpha_1$

$E_2 \downarrow \alpha_2$

$E_1 \parallel E_2$

$t_1 = t_2$

By comparison forced trajectory convergence may be described as:

$$E_3 = <e_1...e_n> \leftarrow t_3$$

$$E_4 = <e'_1...e'_n> \leftarrow t_4$$

$t_3 + t_4$

$E_3 \downarrow \alpha_3$

$E_4 \downarrow \alpha_4$

$E_3 \parallel \alpha_4$

$$E_4 = <e'_1...e'_m \not\sim \triangleright_4 t_4$$

- There are two interaction trajectories...
- each initiated by an agent...
- the trajectories are parallel...
- and have the same objective.

- the associated objectives are not the same...
- each trajectory is initiated by an agent...

- $E_3$ requires the trajectory agent resource $\alpha_4$...
- $E_3$ causes the abandonment of $E_4$ and $t_4$ becomes unachievable.
Trajectory convergence is not explicit in Interaction Framework as presented by Blandford et al (Blandford et al, 1994, 1995). That it is possible to introduce such a construct, in order to be able to describe such a situation, is an indication of the flexibility and generative power of the Interaction Framework.

4.3.3.3. Scenario variations
In the modified scenario some new events are introduced and existing ones modified so that an additional range of situations can be considered. This allows more of the features of Interaction Framework to be demonstrated and also demonstrates further support for knowledge needs analysis.

\[ e_1 \quad \text{ATCO transmits route change request to flight crew} \]

Immediately we have the potential for a problematic event. In communicating with the flight crew the air traffic control officer may need to make several attempts, i.e. the communicative actions may need to be executed several times before the air traffic control officer receives a response (\( e_2 \)) and can know that communication (\( e_1 \)) has taken place. If fortunate the communication (\( e_1 \)) may be successful on the first attempt. In either case there will be an interval between the end of \( e_1 \) and the start of \( e_2 \):

\[
(27) \quad \text{completion (} e_1 \text{)} < \text{time (} e_2 \text{)}
\]

where "completion" is used to signify the end time of an event and "time" is used to signify the start time.

While this is fine for success on the first attempt it does not convey the complexity of communication actions where multiple attempts are required. In that case both (1) and (27) remain true but some additional information is needed.
What are the possible causes for $e_1$ failing:

i) ATCO fails to operate radio equipment correctly,
ii) flight crew may not be receiving on the correct frequency,
iii) there may be an equipment failure (extremely unlikely but possible),
iv) some other cause(s).

How can Interaction Framework be used to handle this type of problem? The first observation relates to how Interaction Framework does not handle it: Interaction Framework does not describe actions that support interaction events. However, by supporting temporal operations we may be able to circumvent the problem of describing a failed interaction event in the following way:

(28) $D_i = T_n - \text{completion}(e_1)$

where $D_i = \text{length of interval}$, $T_n = \text{time at point n}$.

(29) if $D_i > D_a$ then $e_1$ is considered to have failed by the air traffic control officer

where $D_a = \text{acceptable duration}$, variable according to circumstances and standard procedures;

From our consideration of $e_1$ we can see that an acceptable start time, i.e. the available time window, for $e_2$ is defined by the completion of the necessary actions by the ATCO and $D_a$, the acceptable delay...

(30) if $\text{time}(e_2) - \text{completion}(e_1) > D_a$

then $e_1$ has failed or $e_2$ has failed (or both)

That the air traffic control officer needs to know how to communicate, i.e. how to manipulate communication equipment controls, is obvious. That the air traffic control officer needs to know when to make second
and subsequent attempts is less obvious. Statement (30) makes explicit the link between air traffic control officer behaviour in making multiple attempts and the behaviour of the flight crew agent and the need for a defined, or at least understood, acceptable duration between e1 and e2, which, if exceeded, will require e1 to be initiated again. Furthermore, the involvement of both air traffic control officer and flight crew agents suggests that both agents must have an understanding of the relationship expressed in statement (30) and that this is an aspect of communication protocol and timing. The air traffic control officer only has nominal control as the flight crew may not receive the message and if received may have some reason for not complying with it.

\[ e9 \quad \text{Co-pilot transmits new course request to ATCO} \]

It should also be noted that interaction event e9 could take place in parallel with any of the earlier events from e2 and before e10, which must be before e14. Notation is required that allows this to be shown clearly, e.g. :

\[
(31) \quad E_1 \supset E_2 \hspace{1em} // \hspace{1em} E_3 \supset E_4 \rightarrow t_5 \quad \text{-- trajectory where } E_2 \text{ & } E_3
\]

\[
\text{-- must be parallel, but where}
\]

\[
\text{-- } E_2 \text{ starts before } E_3, \text{ and } E_3
\]

\[
\text{-- finishes after } E_2
\]

or

\[
(31') \quad E_1 \supset E_2 \bullet \hspace{1em} // \hspace{1em} E_3 \supset E_4 \rightarrow t_5 \quad \text{-- trajectory where } E_2 \text{ & } E_3
\]

\[
\text{-- may be parallel but may}
\]

\[
\text{-- start in any order, though}
\]

\[
\text{-- } E_3 \text{ finishes after } E_2.
\]

\[
(31'') \quad E_1 \supset E_2 \bullet \hspace{1em} // \hspace{1em} \bullet E_3 \supset E_4 \rightarrow t_5 \quad \text{-- trajectory where } E_2 \text{ &}
\]

\[
\text{-- } E_3 \text{ may be parallel but}
\]

\[
\text{-- may start or stop in}
\]

\[
\text{-- any order.}
\]
The possibility of events occurring in any order or in parallel affects how agent involvement is represented. In the simple sequential trajectory $E_1 > E_2 > E_3 > E_4$ we have:

\[(32) \text{ time } (E_3) > \text{ completion } (E_2)\]

and thus

\[(33) \text{ time } (e_9) > \text{ completion } (e_8)\]

However, even where strict ordering is lost, as in $E_2 || E_3$, then the following statements are still true:

\[(34) \text{ time } (e_9) > \text{ completion } (e_2)\]

\[(35) \text{ time } (e_3) > \text{ completion } (e_2)\]

This is possible because the flight crew is a compound agent and the Captain and Co-pilot can divide tasks between them. Otherwise parallel trajectories involving both the Captain and Co-pilot, as members of the flight crew agent, would not be possible. This may be particularly useful in a severely time constrained situation, i.e. one where the real-time nature of the system becomes significant. If there is a requirement for a manoeuvre to be completed quickly then sharing tasks between multiple agents is useful. For this to occur each agent needs to know what the objective is, what needs to be done to achieve it and how the work may be factored between capable and available agents. In a real-time environment it is also necessary to know what time constraints apply to the desired course of action and any others that may become necessary.

The straightforward description, given in section 4.3.2.2.1 above, contains references to temporal factors but not to real-time factors, i.e. those where there is a time constraint such that the time available to perform an action approximates the time taken to perform it. A possible cause for such a situation is suggested in figure 4.5: the air
traffic control officer's request for a change of course may stem from the need to avoid a collision / near miss situation that would go against \( g_1 \) ( maintain rules of safe flight ). The air traffic control officer is informed by instruments of the course, altitude and probable speed of the aircraft in their airspace. This means that they are more likely to be aware of converging courses of aircraft, and from an earlier point, than flight crews just using visual detection or proximity detectors\(^6\). The training and experience of air traffic control officers may tell them how long they have to take appropriate action, i.e. they will quickly appreciate the size of the time window. Obviously a large window gives more options than a small one.

Interaction Framework provides a framework for analysing such situations. The time interval from observation of the potential for a dangerous situation ( contravention of \( g_1 \) ) and when that event would occur in the absence of preventative action is the duration of the time window ( \( D_w \) ). The time for evasive action to be taken is given by the sum of the durations of \( E_1 \) and \( E_2 \). Thus for \( E_1 \) and \( E_2 \) to be effective:

\[
(36) \quad D_w > D_{E1} + D_{E2}.
\]

While the air traffic control officer may not know precise durations they may be able to estimate accurately and reliably enough for the manoeuvre to be effective on most occasions. The most variable part of the equation, relying on the responsiveness of the human flight crew in initiating \( E_2 \) will be \( D_{E1} \); \( D_{E2} \) being largely determined by aircraft performance. Thus if \( D_{E1} \) passes a certain threshold some other action ( here unspecified ) will be required. The implications of this for KNA are that: i) the flight crew must know that they must be aware of, and respond promptly to, air traffic control officer messages; ii) the air traffic control officer must be aware of the approximate relative

---

\(^6\)Proximity detectors ( Traffic Collision Avoidance System, or TCAS ) are short range radio systems fitted to aircraft that are designed to inform flight crews of the very close proximity of other aircraft on a closing trajectory, and the manoeuvre needed to avoid a collision.
4.3.3.4. Additional points

Specific atomic agents have been identified, such as Captain or air traffic control officer. However, the choice of Captain as agent for some interaction events is not always obvious, as the actions of the Captain, in the demonstrator scenario, could be carried out by the Co-pilot. So, in the demonstrator, “a member of the flight crew” could be substituted for “Captain”. Earlier, in e9, it was proposed that it may be useful to show that a compound agent is known to be such from a specific viewpoint. By making this explicit, as in the case of the air traffic control officer’s view of the flight crew, we can express the air traffic control officer’s awareness of the flight crew’s ability to multi-task. There may be a number of circumstances where such awareness may be useful if certain key objective and goals are to be met.

There are also some interaction events that do not appear to be interactions between agents but that do have corresponding interaction events. This is because an action executed by an agent and subsequently observed by another agent constitutes an interaction event - even if the executing agent is not interacting directly but through the medium of a tool or other non-agent. There need be not intention to interact for interaction to occur. The observation of the action in this context makes it an interaction event. For example, the FMS controls engines and communicates with the flight crew via instrumentation.

In e9, the Co-pilot transmits new course request to the air traffic control officer, the possibility of virtual interaction events was considered. In PUMs the comparable issue of tracking is raised (e.g. Blandford & Young, 1995). This occurs when the actions of an agent or object are not directly observable but are inferred or internally modelled by the interacting agent. In the demonstrator an example of tracking would be where the Co-pilot continues with previous activity
(step S₃b, e.g. map reading) while the Captain interacts with other agents (e.g. E₂ - turn execution). We would expect the Captain to have tracked, possibly approximately, the activity of the Co-pilot. As a result the Captain may expect the Co-pilot to have obtained the necessary information even if the Co-pilot has not initiated an actual interaction event to inform the Captain of this. Thus the actions of the Co-pilot could be said to be virtual interaction events, even though these may be subject to more error than would be the case with actual interaction events.
4.4. Inferring Knowledge Needs from Interaction Framework Descriptions

The Interaction Framework description of the demonstrator has made many features of the system explicit. Primarily these relate to: agent availability, temporal issues and constraints, and the links between low level interaction events and objectives. Making these features explicit enables us to see the consequences of agents lacking certain aspects of knowledge about the SMART system that they are involved with. The main mechanism for establishing consequences is through the link between interaction events and objectives via event trajectories. A failed interaction event that halts or diverts an event trajectory may delay or prevent an objective from being met. Interaction Framework provides a means of making this reasoning explicit.

For an interaction event to occur there are several preconditions. Three are explicit in Interaction Framework:

- an initiating agent must know that an interaction event is required to meet an objective;
- an agent must take action to initiate an interaction event;
- there must be another agent that recognises the interaction event as such.

The first two of these points say something of the knowledge requirements of the source agent. They need to know that an event is required and they must know how to initiate the event (as well as have the means to do so). We should modify the first point by adding that though some interaction events may be intentionally initiated there is scope for virtual interaction events to exist. These result from conscious or unconscious tracking carried out by the recipient agent.

Real-time situations complicate interactions by putting time constraints on both the time available for interaction events to occur and their appropriateness to a given situation. This means that the list of
preconditions for the conscious initiation of an interaction event should be extended to include the following point:

- there must be sufficient time available for the objective supported by the interaction event to be achieved.

Note that this last point does not just require enough time for the interaction event but also enough time for all the subsequent events leading to an objective. This requires that an initiating agent has some awareness not only of their own performance but also of the performance of all other agents subsequently involved in the event trajectory. Interaction Framework provides a way of describing this knowledge and any explicit or tacit calculations necessary to obtain it. Examples of this are provided at various points in the aviation demonstrator.

As mentioned above, Interaction Framework makes explicit the links between objectives and interaction events. As such the description is causal and provides a direct knowledge requirement, i.e. for an agent to achieve an objective they must know the interaction events required to achieve it. Such knowledge helps agents to recognise when objectives are unachievable, or if a detour is required in order to achieve them. One condition that may make an objective unachievable, or mean that a detour is required, has already been referred to: shortage of time for one or more given interaction events. Another such condition is the unavailability of another agent. In SMART systems it is possible that an agent necessary for an interaction event is otherwise engaged.

In the demonstrator scenario the original course of the aircraft is deviated from. However, the goal of reaching the destination remains despite the deviation. The flight crew are aware of this and are able to generate an interaction trajectory that will enable the goal to be met: E3. That interaction trajectory leads to the easy acquisition of the new heading without which it would be impossible for the aircraft to reach its destination and the goal may need to be abandoned, unless the new course could be obtained by other means. This is another example of
the need for knowledge of the links between interaction events and objectives. In this case it can be seen that an agent, in the case the air crew, has had to generate an event trajectory in order to be able to achieve a high priority objective (reaching the desired destination). To do this they require domain knowledge and a knowledge of the rôles of another agent in providing information, namely the air traffic control officer.

Any given interaction event may support one or more goals and objectives. This is obviously true where goals and objectives exist in a hierarchy so that supporting an objective also supports connected higher level objectives and goals.

4.4.1. Explicit KNA gains for the demonstrator
Some of the KNA gains identified from consideration of the demonstrator are reconsidered here as examples of explicit gains made from the application of Interaction Framework. This is followed by a further consideration of the demonstrator with the addition of some numerical values so that the gains made from specific situations may be clarified.

The list of identified gains includes the following:

- human agents must know response latencies of all other agents so that: i) estimates of the durations of event trajectories may be made and appropriate planning decisions made; ii) at a given time point, where a response is awaited, it can be known if the response is due to come or if it is delayed/late. As a specific example of this point we can consider $E_1$. Both the air traffic control officer and flight crew* agents must be aware, probably through training, of the acceptable limits to delays in responses to communications. This is necessary to avoid delays in taking action and to avoid having to repeat transmissions.

- Human agents must know the rôles of each other so that only appropriate requests are made. Knowledge of rôles needs to
include the responsibilities of other agents in addition to their likely performance times and speeds. Thus there is a requirement for training that takes this into account. While instrumentation may display useful temporal data it is unlikely to be able to adequately display role information suitable for a real-time system. When considering the type and location of knowledge, particularly when looking at device design, the application of PUMs, Resources Model and Task Analysis methods may assist. Interaction Framework can provide the temporal and objective parameters for the application of these other methods.

- The NavCom interface and flight crew training needs to support quick and reliable extraction of information about the state of the autopilot. If this isn’t done then the event trajectories involving flight crew - NavCom interactions may fail possibly resulting in blocked objectives. The availability of agents needs to be known if inappropriate requests are to be avoided. For computer agents this may be through a combination of display and lock-out, so that information on unavailability is available and if this is ignored or unobserved then inappropriate actions are blocked. Human agents may also be trained to use a combination of tracking and direct observation applied to all other agents.

- Human agents need to know the sequences of event trajectories and interaction events needed to achieve their various objectives. They also need to be aware of any time constraints that may effect choices. Event trajectories may be pre-learned or may be formed “on-the-run”. The Resources Model may be employed here to identify the information or knowledge sources required ( Fields et al, 1996b ), the two approaches being analogous to following a plan or creating a plan.

- The use of training to support tracking / virtual interaction events would allow for quicker and more efficient event
trajectories and therefore supports objective achievement. This can be seen in the demonstrator where the Captain and Co-pilot may track each others actions while each is engaged in parallel activity. Absence of tracking would make parallel activity more difficult to sustain as each agent would need to direct their resources towards monitoring rather than acting.

- Agents need to be aware of the hierarchy or order of precedence of objectives so that when it arises goal conflict may be resolved. In aviation there is an obvious ordering of the objectives of safe flight and profitable flight. In a commercial, non-aviation setting, the relative priorities of objectives may be more difficult to order. For example: the need to maximise throughput of customers against the need to provide good customer care.

- Where it is necessary to choose between event trajectories information should be available that allows the success or failure of each to be predicted. This may relate to time constraints, to agent availability, or to other factors that may affect the outcome. This has implications both for the training of agents and for the design of interfaces.

It is instructive at this point to consider how the Interaction Framework statements describing the demonstrator scenario lead to quantitative KNA gains. This may be done by moving from a generic description to a specific one by adding numeric data. This can be done by adding data such as specific aircraft speeds and numeric elements of the rules of safe flight to the demonstrator. In the example considered here a further item of data is added: that the situation of two aircraft closing head-on is observed when the aircraft are 12 miles apart. While this figure is arbitrary, it would be possible to identify specific case related figures by reference to domain experts. In aviation, specific locations have their own weather, traffic and restrictions that generate numeric data that may be used in calculations. Consideration
of how to use experts to identify cases and scenarios is done elsewhere 7.

So, for the demonstrator the following data are applied:

- Assumed air speed of closing aircraft (Boeing 747) in mph 8: 550 mph.
- Horizontal separation rule applied by air traffic control officers: 5 miles.
- Horizontal separation at which potential conflict is observed: 12 miles

From the above it can be calculated that the time available to avoid compromising the five mile separation rule is the time taken for two aircraft travelling at 550 mph towards each other to cover 7 miles. That time interval is approximately 23 seconds. Such a situation may be caused by weather patterns, communications failure or non-commercial air activity. If we are concerned with the avoidance of a collision or very near miss, then the time interval is approximately 39 seconds. How can this data be used with the Interaction Framework statements to support KNA?

Initial analysis of the event trajectories reveals that the significant trajectories with respect to achieving $g_1$ (maintain the rules of safe flight at all times) are $E_1$ and $E_2$. Further consideration of these event trajectories indicates that although $E_2$ terminates with $e_8$ the significant interaction event is $e_6$ (Captain executes turn). Subsequent interaction events have no further effect on the horizontal separation

---

7 Dekker (Dekker, 1996) gives an example of an aircraft without functioning communications with an uncertain heading. Part of this uncertainty stems from a radar update period of twelve seconds. In such an interval an aircraft travelling at c.550 knots may cover approximately 2 miles.
8 The difference between nautical miles and miles, and knots and mph is acknowledged. Land based equivalents are used for simplicity.
of the aircraft. Thus, for this example, the period from initiating $e_1$ to completing $e_6$ must be 23 seconds or less.

If we make one further assumption then we are in a position to identify knowledge needs. Assuming that the aircraft has been functioning correctly then the air traffic control officer can reasonably expect the required interaction events to be executed successfully and within the required time (of 23 seconds). The need to assume this is clearer if we considered the reverse position, i.e. one where the aircraft is known to have problems with communication or FMS systems that would make some interaction events impossible or highly unpredictable. In this situation it seems unlikely that the air traffic control officer would try to redirect this aircraft; the other aircraft would be a more reasonable prospect for redirection. This is the second item of knowledge required: that the aircraft is likely to be able to respond in the time available; the first item being the (approximate) amount of time available. If we are to expect event trajectories $E_1$ and $E_2$ to be able to succeed then the two aforementioned items of knowledge are prerequisites.

Some of the interaction events contained in $E_1$ and $E_2$ are relatively deterministic, but not all. Perhaps the least deterministic is $e_1$ (air traffic control officer transmits route change instruction to flight crew). Remembering that the completion of an interaction event is dependent upon effective reception by the target agent, the air traffic control officer must be aware that the flight crew might not receive the message for a number of reasons such as communications equipment set to receive on the wrong channel, or the flight crew might be engaged in some other activity that slows their response. It is not until $e_2$ has been successfully completed that the air traffic control officer knows the flight crew have received the message and are able to take the required action (turn left 90 degrees).

Once $e_2$ has been completed then the time available for the completion of interaction events $e_3$ to $e_6$ is defined. These interaction events are also the most deterministic; with an appropriately skilled flight crew
and a functioning aircraft the time for the aircraft to make the turn could be reliably predicted.

We can represent such facts in Interaction Framework statements as follows:

for goal $g_1$ to be achieved...

$\text{completion}(e_6) - \text{time}(e_1) \leq 23$ seconds

$\text{completion}(e_6) - \text{time}(e_3) = d_t$

$\text{completion}(e_2) - \text{time}(e_1) = d_c$

If the situation reaches a state where $d_c + d_t$ exceeds 23 seconds then the goal becomes unachievable. If the air traffic control officer does not know this then they may persist with inappropriate activity: persistent attempts at $e_1$. Furthermore, the air traffic control officer must know what other courses of action are available and the duration of each. Knowing the duration of each alternate course of action is required so that the most appropriate may be initiated at a time point that leaves sufficient time for completion.

In this example the four interaction events, $e_3$ to $e_6$, must be completeable in a short time: 23 seconds minus $d_c$. This type of information can be used to inform the design of the systems that the air crew must operate. A PUMs type analysis of the interaction between user and device focuses on what the user must know to be able to achieve the objectives. That analysis is time independent. However, in a SMART system we can see that there are time constraints and that objectives are temporally sensitive. Thus the knowledge that a user must have to operate a device must be knowable within the time constraints imposed on the user (or on any of the agents involved). By considering the specific time constraints obtained from well chosen scenarios not only can useful statements be
made about the interaction as a whole but components of that interaction can also be analysed.

That areas of airspace have their own characteristics determined by such factors as weather and traffic patterns makes the experience of air traffic control officers location specific and less transferable than would otherwise be the case. From the above example it seems likely that an Interaction Framework analysis of key scenarios for a given airspace could provide useful information for trainers. Critical time periods and interaction events could be identified and used as the focus of training episodes. This would ensure that air traffic control officers internalise the knowledge necessary to handle events in their airspace. An alternative view is that the information available to air traffic control officers through instrumentation could include temporal data such as temporal separation of aircraft or the time since the last display update.

4.4.2. Explicit KNA gains for demonstrator variations
Previously consideration was given to variations from canonical event trajectories. These variations included problems with air traffic control officer to flight crew communications, the introduction of parallel event trajectories and the introduction of explicit time constraints.

The first and third of the variations listed here require air traffic control officers to know about the amount of time available and the normal duration of interaction events. If either of these is not known with adequate precision then the performance of the system as a whole is likely to be compromised: interaction events will not be initiated quickly enough, delayed or missing interaction events will not be recognised as such. Recognition of these characteristics of the air traffic control situation enables trainers and system designers to ensure that the human agents involved are trained in such a way that the significant temporal knowledge is acquired. It may also be appropriate to consider how support may be provided for human agents to ensure when cognitive load is high, such as might be the case for air traffic control officers in abnormal or emergency situations, that key temporal
information is tracked and is available to the agent. Interaction Framework may help to identify such situations by quantifying the demands on the agent by identifying the interaction events involving them and the time window in which they need to occur. However, Interaction Framework does not say how such information should be presented or what the precise content should be. For example, it may be necessary to conduct usability experiments to determine if information should be presented textually or graphically and what the content of information should be.

Where parallel event trajectories occur, or are required, then Interaction Framework may be used to make explicit the availability of agents to perform parallel activities. In addition, the Interaction Framework description may identify the need for parallel activity in order that the difficulties imposed by time constraints may be eased.
4.5. Some Observations On The Psychology Of Time

In this short section some experimental findings on psychological aspects of time are presented. While it is clear that agents may need to have some awareness of time, may need to use temporal knowledge, and make temporal judgements it is not clear how well this can be done. The work reported here on the psychology of time shows some of the problems faced by individuals in making temporal judgements and considers some of the implications this may have for SMART systems. It is not the purpose of this section to give a complete and full account of the psychology of time but to give an indication of the complexity of the issues involved.

Analysis of the demonstrator situation reveals that it can be useful to know about temporal issues. The air traffic control officer expects the flight crew to respond promptly, i.e. within a few seconds (e.g. interaction events e1 and e2). The flight crew also expect the air traffic control officer to be responsive (e.g. e9 and e10). When the Captain deactivates the auto pilot the NavCom display is expected to indicate this promptly, within fractions of a second. Knowing how long an action supporting an interaction can take and the time available may be essential in selecting the right actions for a given situation. Interactions taking place in time constrained circumstances may go wrong, preventing objective achievement, if time factors are not well understood. Unfortunately, ensuring that agents actually know what they need to know is problematic.

The evidence for internal clocks with precise and linear scales is weak (Block, 1990) though there is evidence from both human (Zakay, 1990) and animal experiments (Roitblat & Young, 1990) for mechanisms relating to the judgement of intervals and the due point of events. These judgements seem to be qualitative rather than quantitative and precise. A finding reported by Zakay is that "...observers tend to overestimate brief intervals and underestimate long ones" (Zakay, 1990, pg. 62). This situation is further complicated by the context in which time judgements need to be made. Block (Block,
1990) identifies four factors affecting psychological and cognitive aspects of time:

- personal characteristics (e.g. past experience and personality)
- structure of the time period (e.g. complexity of events, duration of interval)
- activity during the interval (e.g. non-attendance versus attendance to events)
- time related behaviours and judgements (e.g. estimation, comparison)

From these four points we could expect to see considerable variation in performance within and between individuals making time related judgements. As experience in an environment is gained we would expect increased accuracy in estimation tasks. However, the perceived complexity of a sequence of events within an interval will affect judgements of the length of the interval and the sub-intervals between events. Individuals focusing on temporal factors will make different judgements from those who are not focusing on temporal factors but who are asked to judge them afterwards. Such judgements may be made more difficult where events occur in close proximity (Patterson, 1990). In such cases information relating to earlier events may be masked by that of later events. Though Patterson’s work was with visual stimuli it seems possible that the same effect would apply to a range of stimuli and events occurring within SMART systems. This will make knowledge of earlier events less reliable and may result in faulty judgements. This effect is reduced where events occur further apart. This suggests that the performance of SMART systems would deteriorate where human agents need to make temporal judgements about events occurring close together.

Another factor effecting the accuracy of temporal estimates is the method used. Zakay (Zakay, 1990) lists four methods: verbal...
estimation, production of a requested interval, reproduction of a demonstrated interval, and comparison or relative lengths of intervals. The accuracy of subjects varies between methods. Zakay (ibid.) infers that each method relies on different psychological processes and that this accounts for the variability. A further complication is that well learned tasks, i.e. those that are automatic, do not rely on cognitive manipulation of time in the way that non-automatic tasks do (Michon, 1990). In addition, the execution of automatic tasks interferes with subjects attempts at making explicit cognitive representations of time. This could present problems in environments containing a mix of learned, i.e. automatic, and unlearned tasks. When an unlearned task is encountered the necessary temporal information from a previously executed automatic task may not be available for reasoning with.

Animal experiments (Roitblat & Young, 1990) show evidence for some internal awareness of the passing of time that has some clock like properties - but not in a regular / regulated way. In an experiment with conditioned stimulus / pause / unconditioned stimulus structure (bell, fixed interval, food) the animal would rise closer and closer to the unconditioned stimulus point. In early trials the animal would rise soon after the bell; while in later trials the animal would delay rising until close to the time of food presentation. What are the implications of this data for SMART systems? Initially there may be a tendency for human agents to react quickly to a situation (analogous to the conditioned stimulus) where a negative event requiring a response is anticipated (unconditioned stimulus). After further experience or training the response may be safely delayed. An advantage could be to allow greater thinking time in which the agent may bring experience, training and problem solving skills to bear so that the most appropriate response is selected from the set of possible responses. Training will increase user knowledge in terms of increased automation of appropriate responses, awareness of likely intervals, and responses requiring problem solving techniques. Where tasks are well learned, to the point where they are automatic, there is no need for an explicit knowledge of time. If this is the case then a reasonable conclusion would be that for critical tasks there
should be a high degree of learning so that processes become automatic. Human agents should be well drilled so that critical aspects are well learned. Selection criteria for staff would need to take this into account. Where cognitive problem solving is needed then users should be supported in making temporal judgements. This would be an issue for the design of automated agents. Alternatively, the need to make such judgements should be minimised or eliminated as, from the above evidence, human performance may be unreliable.
4.6. Conclusions

The use of an appropriate demonstrator, from the aviation domain, and the use of Interaction Framework to support KNA makes it possible to propose that Interaction Framework could be used for systems that are either more complex or less complex than the demonstrator. The complexity of multi-agency, and the aviation scenario has five atomic agents and one compound agent, makes it easy to see that a reduction in the number of agents would not invalidate the approach. Any system simpler that the demonstrator in terms of numbers of agents will just be describable in fewer statements. Systems that are more complicated that the demonstrator, in agent and / or temporal dimensions, can be modelled by the addition of the requisite agents, and by using the required descriptive statements. Admittedly complexity of the description will grow at least as rapidly as system complexity grows, probably more so, but the system will be describable and knowledge needs inferable albeit with greater effort.

Temporal aspects of systems that are vague but none the less have limits, can be described using Interaction Framework. The interaction event e1 was described twice. The second description introduced an indeterminate time interval. In identifying the possible existence of such a time period a mechanism for handling it, with respect to KNA, was also identified. The identification of such an interval shows a need for training of one or more agents and, possibly, a need for tightly defining the interval so that the reliability of system performance is improved. Other KNA gains were also made.

The customer support system was rejected as a demonstrator because of apparent problems with clarity and ambiguity. We are now in a position to see how such a system could be modelled using Interaction Framework if the need arose. The uncertain intervals in that system may be modelled using simple statements of the type used to describe interaction event e9 (statement (36)). Such a description, and the subsequent analysis, supports decision making with respect to system design, the training of staff and operational procedures.
Interaction Framework supports the description of SMART systems in a way that is susceptible to mathematical and logical analysis. This analysis supports KNA for agents within the system, with a particular focus on temporal issues. This includes the availability, or non-availability, of other agents within the system that may be necessary for co-operative working. Such knowledge may be necessary for decision making and optimal system operation.

In applying Interaction Framework some modifications have been found to be useful. This does not detract from Interaction Framework, more it indicates the flexibility of the Framework to be adapted to the requirements of particular situations. The modifications suggested would seem to be in the spirit of Interaction Framework. Further application of the framework may reveal other additions that will make it easier to apply to a wide range of systems and domains.

Although Interaction Framework describes what must be done in order that an event trajectory reaches its objective normally, and necessary knowledge can be inferred from this, nothing is said about how things are known. Possibilities include:

- learned patterns of behaviour / procedures (e.g. learned manuals),
- tracking (based on a model of the behaviour of another agent),
- display / explicit statement or representation,
- problem solving, e.g. inference (most apt in novel situations).

Interaction Framework's contribution is to make the knowledge requirements explicit. From such explicit statements system designers or maintainers may select from the options available on the basis of agent limitations and the requirements of the situation. Other KNA related tools such as the Resources Model, Task Analysis and PUMs can be applied to interactions between two agents. Such work may use the information made explicit through the application of Interaction Framework. The requirements of aviation, with its safety critical
requirements, may be different to the requirements of a trading or manufacturing systems that are less safety critical and have different degrees or freedom for manoeuvre.
Chapter 5. Discussion and Conclusions

5.1. Review

A class of systems in which multiple human and computer agents are simultaneously active in real-time environments has been described. These systems have been termed “SMART systems”. The need for human agents to interact with one or more other agents, to use their resources in time constrained situations, poses human factors problems that are not found in all other, non-SMART, systems. The nature of the time constraints of real-time systems is that agents have a limited amount of time in which they may perform necessary actions and the appropriateness of actions changes with time due to the existence of time-windows.

In examining the human factors issues of SMART systems the focus has been on what it is that a user needs to know. In particular, attention has been paid to the special requirements of SMART systems resulting from time constraints and agent availability. Other techniques supporting knowledge needs analysis (KNA), such as programmable user models, PUMs (Young et al. 1989; Young and Blandford, 1994), Resources Model, RM (Fields B., Wright P., Harrison M., 1996 ab; Wright P.C., Fields B, Harrison M.D., 1996), and task analysis (Diaper, 1989b) do not support multi-agency or real-time well despite providing detailed support in other areas of user knowledge needs. Where time is considered as an important factor the focus tends to be on issues such as system responsiveness or the pace of interaction (Wickens, 1984; Dix, 1992).

To support KNA for SMART systems an understanding of the particular system in which human agents are operating is required. Interaction Framework has been identified as a suitable system modelling method, bringing out the important factors of SMART systems. Its selection was based on its overall profile on a set of criteria in five themes. Under
each theme heading more detailed criteria were provided. Each method was then subjectively assessed to produce a ranking of method suitability. Interaction Framework was top of this ranked list and was thus selected.

The themes are:

- communication,
- control,
- goals,
- knowledge,
- time.

Interaction Framework is flexible in that a manageably small notation set can be used to describe a number of different agent configurations, their objectives and the interactions required to meet objectives. The notation explicitly caters for temporal factors and allows statements to be made about the past, present and future availability of agents. The purpose of Interaction Framework, as given by its authors (Blandford et al., 1995) is to provide:

- a design and evaluation aid for interactive systems with multiple agents,...
- that has a neutral perspective with respect to both human & computer agents,...
- making features and qualities of the interaction explicit.

( adapted from ibid. )

Having identified Interaction Framework as a suitable tool for describing SMART systems it was applied to a demonstrator. The purpose of this was firstly to show the ease with which systems may be described with Interaction Framework and secondly, to show that KNA is supported by such descriptions. This use of Interaction Framework is additional to that envisaged by its designers. In selecting a demonstrator from a specialist domain it was necessary to
seek expert validation to confirm the content and structure of the scenario. This was done and the feedback provided useful information on a number of minor points that were used to improve the demonstrator. Though the demonstrator was from a specialised domain, aviation, it is apparent that Interaction Framework may be applied to systems of greater or lesser complexity and that it is domain independent. However, although Interaction Framework provides useful support for KNA, it needs to be supported by other techniques, such as PUMs, the Resources Model, or task analysis, if a full set of user knowledge requirements is to be identified covering such aspects as equipment use or operational procedures.

Identifying areas of system operation where time is particularly critical, i.e. where temporal demand is closely matched to temporal ability, allows designers to focus on potentially sensitive areas. Interaction Framework supports the segmenting of systems into time sensitive and non-time sensitive phases so that Interaction Framework modelling effort may be focused where it can be of greatest benefit. This was shown in the demonstrator by modelling a segment of a flight and not the whole flight. The KNA and performance analysis would provide input to the design and modification of systems. Gains could be expected in the form of lower rates of operational errors and, potentially, reduced operator stress (Wickens, 1984); though this issue has not been considered in this thesis.

Other issues that have not been considered, but which may be valid areas of interest are: dynamic modelling strategies that agents employ in interacting with other agents; stress effects on performance resulting from real-time factors; differences in interactions between two human agents, and between computer and human agents.
5.2. Interaction Framework and the Proposed Modifications

As a result of familiarity gained with Interaction Framework, during the method selection and application phases of this work, a number of characteristics have become apparent. In addition, the application phase revealed some gaps in the notation. These were easy enough to fill, so that there was little difficulty in applying a modified Interaction Framework to the demonstrator scenario. Some high level characteristics of Interaction Framework and a number of proposed modifications are described below.

5.2.1. Characteristics of Interaction Framework

Interaction Framework provides a way of making some of the key consequences of actions explicit - particularly where time and multiple agency are significant system features. Consequences of particular interest are those relating to the meeting of objectives and the availability of agents. Those of particular interest for a given system can be given qualitative labels such as: “interaction detour”, “system reset”, “order-independent events”, etc. Explicit awareness of these consequences is a useful aspect of knowledge for both human and computer agents if they are to be able to make informed choices in a range of situations. The approach of Interaction Framework can be broadly categorised as reductionist. An interaction between agents that could be simply described at a high level as agents co-operating to achieve an objective, can be reduced to a number of lower level statements. An interaction between agents may be considered as an event trajectory, with each of the constituent interaction events named and labelled. Each interaction event can be further described in a series of statements that make explicit the temporal factors and the involvement of each of the agents taking part.

Qualitative aspects of the human factors dimensions of interactions do not feature in Interaction Framework descriptions of systems. Issues such as learnability, effectiveness (Shackel, 1986) and subjective
evaluation (Whiteside et al, 1988) are not directly addressed by Interaction Framework descriptions of systems. This is true even though usability engineering approaches, such as those of Whiteside et al (ibid.) share a reductionist approach with Interaction Framework. However, the reductionism of Interaction Framework may indirectly support other approaches to usability by helping to define the conditions under which human agents are expected to work. For example, knowing the extent to which knowledge needs to be automatic has some bearing on learning issues; the time available for cognitive operations affects ease of use and may also affect subjective evaluation.

Time factors were considered in describing the aviation system. However, they were considered in an abstract way, one susceptible to mathematical manipulation; the actual values of time points and intervals do not make any difference to those manipulations. This makes Interaction Framework suitable for modelling SMART systems operating under a wide range of temporal regimes, i.e. where time is measured in fractions of seconds to durations of hours, days, or weeks.

5.2.2. Modifications to Interaction Framework

During the modelling method selection process Interaction Framework was studied. The study revealed some areas of the notification that required clarification. In addition some concepts were added that seemed to follow on from the description of Interaction Framework but that had not been explicitly included. These points are:

- completion time of an event added to start and duration

  completion: \( \tau - \pi \) defines the time at which the event ends

In the original description of Interaction Framework the start time and duration of interaction events were explicitly covered by the notation. While these two items are, arithmetically, sufficient to determine the end time of an interaction event they
do not support all the necessary reasoning. In time constrained situations it may be necessary to work back from the time that an interaction event completes to determine its start time. Also, to make reasoning about agent availability easier, the identification of when an interaction event has completed will clearly identify the point at which an agent is available.

- consistency point, i.e. notation of arrows is subject to minor alterations to make it internally consistent

\[
\text{recent: } \varepsilon \times A \ni T \text{ defines the time [i.e. start time] of the most recent event involving an agent in } A.
\]

\[
\text{recent: } \varepsilon \times A \rightarrow T \text{ defines the time [i.e. start time] of the most recent event initiated by an agent in } A.
\]

A more minor point was that some of the arrow elements of the notation seemed to be applied inconsistently. This has been made consistent.

- ballistic / guided trajectories

During the description of Interaction Framework given by Blandford et al (Blandford et al, 1995) an example was used to illustrate the point that event trajectories may sometimes be interrupted (Greatbach et al, 1993). Further consideration of this point leads to the conclusion that some event trajectories could be classified as "guided", others as "ballistic". Guided trajectories may be initiated by an agent, observed and further controlled to ensure that the objective is met. This contrasts with ballistic trajectories where further interaction on the part of the initiating agent is not possible after the first interaction event. Where agents misclassify trajectories, i.e. where their knowledge of them is flawed in this respect, performance of the system may be sub-optimal. This is linked to the concept of "local agent control" (Blandford et al, 1995). However, it is
believed that by pairing ballistic and guided trajectories the difference between types of control is clarified.

These trajectories may be formalised as follows:

**Ballistic trajectories**

A ballistic trajectory occurs when an event trajectory that is initiated by an agent to achieve an objective *may not* be subsequently altered by that agent should it appear that the trajectory is deviating away from that objective.

For an agent not to be able to exert further influence on an interaction trajectory it is necessary that the agent is excluded from initiating any further interaction events after the first in the trajectory. The agent may be the target of interaction events but is not able, for whatever reason, to respond in the event of an apparent deviation from the objective. This may be described as follows:

\[
\begin{align*}
(37) & \text{ } E^\equiv (e_1, e_2, ... e_n) - t & \text{event trajectory } E \text{ achieves } t \\
(38) & \text{ } E = E_1 > E_2 & \text{E may be considered as two event trajectories: } E_1 \text{ & } E_2 \\
(39) & \text{ } E_1 = (e_1) > e_2 & \text{E}_1 \text{ has one interaction event, or terminates in an interaction event} \\
(40) & \text{ } e_1 \downarrow \alpha_1 & \text{...that is initiated by } \alpha_1 \\
(41) & \text{ } (e_2, ... e_n) \downarrow \mathcal{A} & \text{all subsequent interaction events are initiated by a set of agents...} \\
(42) & \text{ } \alpha_1 \in \mathcal{A} & \text{...that does not include } \alpha_1
\end{align*}
\]
Guided trajectories

A guided trajectory occurs when an event trajectory that is initiated by an agent to achieve an objective may be subsequently altered by that agent should it appear that the trajectory is deviating away from that objective.

The initiation of a guided trajectory may follow the same first few steps as a ballistic trajectory (37 to 40). However, after initiation, the trajectory may be modified by \( \alpha_1 \) if a deviation from the objective becomes apparent. This results in an interaction detour but may achieve the objective. Note that the intervention of \( \alpha_1 \) following the deviation does not guarantee that the objective is achieved, but does increase its chances in that if there was no intervention the objective would certainly be missed. We begin the description with the assumption of interaction detour:

\[
\begin{align*}
(43) \quad & F' \triangleright F \triangleright F'' \leftarrow t \text{ and } F' \triangleright F'' \leftarrow t \\
(44) \quad & F' = \langle e_1 \rangle \triangleright e_2 \\
(45) \quad & e_1 \downarrow \alpha_1 \\
(46) \quad & F = \langle e_2, ...e_h \triangleright e_i, ...e_n \rangle \\
(47) \quad & F = G' \triangleright G'' \\
(48) \quad & G' = \langle e_2, ...e_h \rangle \\
(49) \quad & G'' = \langle e_i, ...e_n \rangle \\
(50) \quad & e_i \downarrow \alpha_1
\end{align*}
\]

there is a simple interactional detour \( F \)

\( F' \) has one interaction event, or terminates in an interaction event...

...that is initiated by \( \alpha_1 \)

the interaction detour, \( F \), contains a number of interaction events

the simple detour may be considered as having two parts \( G' \) and \( G'' \)

\( G' \) is the part of the detour before \( \alpha_1 \) takes remedial action...

\( G'' \) starts with that remedial action...

... initiated by \( \alpha_1 \)
This may be summed up as follows. The intention was for a canonical interaction (F'=F' \rightarrow t). However, an interaction detour occurs (F). The detour is in two parts (G' and G'') where G' is that part of the detour before action is taken (either because of lack of perception or inability to intervene) and G'' is the event trajectory required to bring the whole interaction back towards its objective.

In applying Interaction Framework to the demonstrator a number of other areas were found where changes and additions were identified that could improve its power and expressiveness. These changes, further described below, are not so radical that they change the nature of Interaction Framework. They allow for some aspects of systems to be described in more detail, and may prove very useful in supporting KNA. The changes identified from the application of Interaction Framework are:

- view point notation: to clarify the availability of atomic agents from compounds

ATCO \rightarrow \text{dest(e10)} \rightarrow \text{flight crew}

The need to differentiate between view points became clear when considering interaction events involving compound agents. In the demonstrator a message was sent by a compound agent (the flight crew) to an atomic agent (air traffic control officer / ATCO). While it is reasonable for the air traffic control officer to view the flight crew as an agent in this interaction event it is not so reasonable for member agents of the flight crew to do the same. That they have a different view point is clear. The significance of their particular view is that it gives additional information about the availability or non-availability of the atomic agents comprising the flight crew that is not available to the air traffic control officer.
Interaction Framework does not have a notation that can easily be used to describe perspective. In the case of $e_2$ there are two distinct views of the event: that of the air traffic control officer agent and the views of the members of the flight crew; Captain and Co-pilot. The view of the air traffic control officer is that the flight crew has responded, and in doing so have achieved objective $t_{5.1}$. Just considering this view may lead to the conclusion that the flight crew were busy and unable to carry out any other tasks in parallel with $e_2$. This is not the case. While one member of the flight crew is involved with $e_2$ the other may be committed to some other action or may be uncommitted. This affects the apparent availability of agents and so may affect other agents’ views of what may be done, e.g. the interruptability of an agent.

- marking of compound agents: the use of $**$ notation

$flight\ crew^{**2}$

Linked to the clarification of viewpoint is the addition of notation to differentiate between compound and atomic agents. This is useful where an interaction event is directed at a compound agent in situations where action may only be required by one of its atomic agents. This helps to avoid situations where incorrect assumptions might be made about the availability of agents that might be thought to be atomic when they are actually compound.

- explicit order independence: $\bullet$ notation

$e_5\bullet e_6$ or $e_5\bullet//e_6$ or $e_5\bullet//\bullet e_6$
In the original Interaction Framework notation interaction events are listed to show either that events are partially ordered or that they occur in parallel. The • notation was added to show the start and finishing sequence of parallel interaction events. This allows for some choice on the part of initiating agents and enables activity to continue rather than have to wait for another agent to become available. This may be particularly useful in a real-time environment where time is short and a number of interaction events need to be completed for an objective to be achieved.

These changes and additions to Interaction Framework increase the range of SMART system features and situations that can easily be described. The ease with which Interaction Framework can be augmented is an indication that it could, if required, be augmented in further applications to SMART systems should the need arise. This may be done without compromising the basic form of Interaction Framework, retaining its constructs of agent, objective, interaction event and event trajectory.

5.2.3. Interaction Framework considered against the five themes.

Five important themes relevant to the effective functioning of SMART systems were introduced in chapter three:

- communication,
- control,
- goals,
- knowledge,
- time.

These themes were used to create criteria against which all of the system modelling methods could be compared. As a result of this comparison Interaction Framework was chosen as being the method
with the greatest potential for revealing the knowledge requirements of agents in SMART systems. Having applied Interaction Framework it may now be appropriate to review the relevance or appropriateness of the criteria.

The consideration of Interaction Framework against the communications theme given in chapter three still seems reasonable at this point. While the syntax, mode and specific surface structures of communications are not revealed by Interaction Framework models they do provide the purposes of communications and specific information about timing. The semantics of communications are linked to their immediate and long term objectives. The mechanisms in Interaction Framework for linking low level interaction events to high level objective and goals make the semantics an inherent part of interaction event descriptions.

Control can be viewed in two ways once an Interaction Framework description has been produced. For a canonical trajectory control can be said to rest with the initiating agent. When an agent has an objective that is to be achieved through a particular trajectory, that they may or may not choose to initiate, then that agent may be said to have control over the achievement of the objective. However, there is a deeper way of looking at control. The concepts of local agent control and forced trajectory convergence were considered in chapters three and four respectively. The control concepts discussed in those chapters reflect the diversity of situations where control may be fixed or may move according to the characteristics of the system or local context. Interaction Framework allows systems to be described in ways that allow control issues to be explored and thus appropriately inform the design process. An example of an important control issue is goal conflict. This occurs when there is forced trajectory convergence. By making such issues explicit it is more likely that they will come to light during design rather than the later, and less convenient, operational stages.
Although the language of Interaction Framework is more explicit about objectives these are readily linked to the less explicitly described goals. This enables Interaction Framework descriptions to span a broad range of activity: from high level, goal oriented views of activity; down to individual interaction events. By linking goals and objectives to interaction events we can make a link between high level interests and specific, low level activity. The termination of event trajectories with an achieved objective, coupled with the necessity to link agents with interaction events, makes a specific link between agents and objectives / goals inevitable. It can be said that agents initiating events that support objectives have some ownership of the objective. It is not so clear if the same can be said of agents that are targets of the same events. Where an agent can and does block an event trajectory then we can say that they do not own that objective, or that they own some objective with higher priority. Also, where an event trajectory causes the achievement of one objective at the expense of a higher priority objective we can say that the trajectory is inappropriate. An example of this could be seen in the demonstrator. One goal is to reach the destination efficiently, while another is to maintain safety. It is clear that interaction events supporting efficient routing over safe routing are to be avoided.

Interaction Framework does not explicitly consider knowledge to be an issue. However, the descriptions of event trajectories contained in an Interaction Framework model of a scenario make certain knowledge requirements obvious. It is necessary for an agent to know which other agents are available, what their capabilities are and how interacting with them will support system objectives. The consequences of not having this knowledge can also be seen in that failed or detoured trajectories may prevent objective achievement.

The temporal operators in Interaction Framework make the description of temporal boundaries (time windows), sequencing and concurrency easy to describe. This enables the temporal properties of a system design or of an operational scenario to be explored with relative
formality. That is, the behaviour of the system as a whole and that of the agents within it can be treated mathematically. Where real-time is the case then design decisions can be made to ensure that the system performs as required by taking appropriate human factors into account.

In addition to its characteristics with respect to the five themes Interaction Framework also has another of importance: agent neutrality. This neutrality allows the performance of all system agents to be considered equally. Where the Interaction Framework model provides useful design information this may relate to either computer or human agents. Thus design decisions about computer or human components of systems are supported coherently. Computer design specialists, human factors specialists and human resource management specialists could be brought together, and their spheres of operation linked, through an Interaction Framework based analysis. This could reduce the risk of conflicting views giving rise to conflicting design decisions, but remains to be tested.
5.3. Applying Interaction Framework to KNA

By using Interaction Framework to describe systems areas of activity can be identified where time may be constrained, i.e. where there may not be enough time to complete actions or where actions have limited time span of appropriateness. We can describe sequences of interaction events, and the availability of agents necessary for objectives to be achieved. If agents do not have knowledge of these things then it is unlikely that the system will function correctly. Thus by making the sequence and timing of interaction events explicit, along with the objectives they are to achieve and the identities of all agents involved, it is possible to make explicit the knowledge requirements of all of the agents that support an objective. How the knowledge is known is not made explicit. The way of knowing needs to be determined by designers, some of whom may need an awareness of psychological aspects of human performance under a range of conditions as they apply to the specific system being considered. Where time is very highly constrained then it may be appropriate to embody knowledge in an automated agent. In less constrained situations human agents may be expected to acquire the necessary knowledge, either to the point where it is automatic or to where it may be reasoned with. Knowledge of the capabilities of other agents is required so that the correct event trajectories may be initiated. Where this knowledge is not available then a high proportion of interaction detours may exist such that objectives are blocked. The status of agents may be known through tracking, inference or direct observation. Systems need to be designed so that mechanisms of knowing, appropriate to possible situations, are supported.

The structure of Interaction Framework descriptions is such that the failure of an agent with a particular interaction event (or its associated task actions) can be linked to specific consequences, such as the non-achievement of specific objectives. The consequences of interrupting an engaged agent can be seen in the delay to that agent's goals
For Interaction Framework to be useful it is only necessary to identify aspects of system operation that are of interest - it is not necessary to model a complete operational cycle. This was demonstrated in the selection of a segment of activity from a larger aviation system. Useful information about knowledge needs was obtained from a small segment without the need to know about all of the preceding events. However, where some aspects of a target segment do require knowledge of previous events then Interaction Framework provides a mechanism for looking at the preceding events. For example, if it is necessary to know about the availability of an agent then Interaction Framework enables us to consider the prior interaction events either initiated by, or involving, that agent and the time of completion of those events.

The understanding of systems gained by Interaction Framework description and subsequent KNA has applications in macro systems design, the design of user interfaces for computer agents and in the selection and training of user / human agents. This is a new use for Interaction Framework. Applied to macro system design Interaction Framework may be able to highlight sequences of operation where there is a prolonged period of time constrained activity. In such situations the margin for error may be slim, possibly too slim to be acceptable. Errors may occur because of component failure: the failure of some item of equipment, or the sub-optimal performance of a human agent (Wickens, 1984, ch. 9). Where such problems are identified at the design stage it may be appropriate to redesign the system and / or the structure of the tasks to be carried out.

In an earlier chapter reference was made to the inspiration for considering KNA that was gained from the PUMs approach (Young et al, 1989; Young & Blandford, 1994). An idea central to PUMming is the designer's intended procedure. This refers to the way that designer's
intend a device to be used, and includes the various states of the device and the user actions required to move from one state to another. By examining the designer's intended procedure and the user's goals at various points in the interaction inferences can be made about what it is the user needs to know in order to be able to use the device and achieve their main objective. This approach is problematic when applied to SMART systems. While it may be appropriate to consider individual computer agents and the designer's intended procedure for each of these, there may not be a designer's intended procedure for compound agents, especially where these contain one or more human agents. Interaction Framework provides a way of describing the operation of agents in SMART systems that has parallels with the designer's intended procedure and that allows operation to be better understood. This understanding is a prerequisite for making improvements.

5.3.1 A method for applying Interaction Framework

Describing a method for supporting knowledge needs analysis

This section describes a method for knowledge needs analysis. The focus is on the sequence of activities so that a designer working on SMART systems can work through the method in support of knowledge needs analysis. A trivial example is provided in appendix 5.1 to provide an illustration. The method contains a number of steps:

1) identify key scenarios within the target system;
2) describe each scenario in Interaction Framework terms: objectives, agents, interaction events;
3) identify constraints: temporal factors, factors preventing canonical event trajectories;
4) apply tools (Resources Model, task analysis, PUMs and others as appropriate) to complete KNA;
5) evaluation
This sequence allows the identification of problem areas or areas of interest and supports their description and analysis in a way that takes account of multi-agency and real-time constraints. Each of these steps is considered in further detail below. It should be noted that this sequence of steps has been extrapolated from work done on the demonstrator but remains to be tested on a live system.

**Identify key scenarios**

Dekker (Dekker, 1996) states that "The design of an incident scenario must be driven by a predefined phenomenon of interest." This is interpreted here as meaning that the selection of scenarios should not be haphazard but should be focused on specific issues. When the aviation demonstrator scenario was selected as a vehicle for exploring the suitability of Interaction Framework it was done with great care. The needs of a demonstrator were considered and a number of potential demonstrators were identified. The need for multi-agency and real-time were important features as was the need for the scenario to be understandable. In the examples used by Dekker (ibid.) issues were domain specific rather than related to the identification of an analysis tool, as is the case here.

For a live system the "phenomenon of interest" would be defined by the needs of two sets of people: the designers of the system and those with a direct stake in its operation and use (who are not the designers). Thus if either of these sets of people identifies a point of significant interest then a scenario must be designed that includes material allowing the relevant issues to be fully explored. In Dekker's method, where rule changes are under consideration, the scenario is designed with the proposed new rules in operation to act as probes. In using a scenario based approach to support knowledge needs analysis the same approach may be used for identifying scenarios: one or more system designers become sufficiently familiar with the domain; system designers and other stakeholders identify the significant issues; other constraints and enduring problems are identified; a scenario is developed that supports exploration of the target issues whilst also
ensuring that standing constraints and problems can be taken into account. From this point the scenario may be described using Interaction Framework.

Describe each scenario in Interaction Framework terms
The Interaction Framework description needs to link goals and objectives of the scenario(s) to event trajectories and interaction events. These items have to be linked to atomic and compound agents. Interaction Framework does not generate information on time constraints or agent structures - these must be provided by those with a stake in system design. As with Dekker's document based method it is necessary to refer descriptions back to system operators and specialists. This allows for misinterpretations and other corrections to be made. It is to be expected that the process will be iterative and will continue until no further changes are required.

While the process of building an Interaction Framework description may start with a description of goals and objectives and work down to interaction events this is not a requirement. It may be that the special characteristics of the key phenomena direct the style and order of development. In the demonstrator scenario the analysis was driven by the need to maintain safe flight. In a system taken from a commercial domain where the key issue might be agent availability then the initial focus might be on agent structures. However, whatever the circumstances, it is necessary to obtain sufficient information to describe the full set of Interaction Framework features.

A significant task in forming Interaction Framework descriptions is the identification of agents. While atomic agents may be easy to identify, being either individual human operators / users or significant sub-systems, the identification of appropriate compound agents is more problematic. In some situations there will be obvious groupings. For the demonstrator the flight crew* is an obvious agent entity: the flight crew* agent is a target for air traffic control officer communications. In other systems the identification of compound agents may not be so
clear. One of the systems rejected for demonstrator purposes was credit card transactions. In this situation the customer and sales assistant are obvious atomic agents. The credit agency or card company is less easy to categorise. Is it atomic for the purpose of the transaction in that it is a single point contacted by telephone. The credit clerk being called may use a computer system to make a decision, but that will be communicated back by the clerk to the sales assistant. A further problem is whether or not the sales assistance and the credit agency form a compound agent as together, from the customer's point of view, they control the sale (an example of local agent control).

Potential solutions to the problem of compound agent definition are:
1) assume all possible combinations of atomic agents may have meaning;
2) do not assume any compound agents until their existence is unavoidable;
3) allow intuition and prior knowledge to define compound agents until this is found not to work.

The first assumption is problematic as it suffers from combinatorial explosion. A system with three atomic agents may have a further three compound agents of two atomic agents. We could also continue to combine compound agents, which in this case would add another compound agent of three atomic agents, making a total of seven atomic and compound agents to be considered. For systems with four or more atomic agents the rate of agent growth increases rapidly. On these grounds it seems appropriate to reject this method of grouping agents.

The second assumption has a further assumption embedded in it: where there is a genuine need for a compound agent then this will become clear during the modelling process. It is not clear at this stage if there is sufficient experience with Interaction Framework to believe that this embedded assumption is reasonable. If it is not reasonable then it may be the case that useful compound agents are not identified.
during the modelling process. A further consequence of this may be that sub optimal design decisions are made as a result.

The third assumption has a higher degree of face validity than the other two. We might expect system designers to have some initial intuitions about reasonable agent groupings. This could be a reasonable starting point for agent groupings. As analysis progresses the initial selections may be refined.

In considering these three approaches to compound agent identification it would seem that a reasonable approach is to use the third assumption, or the second together with ethnography, for the following reasons. Where intuitions are available then the third approach will prove to be economical. The combinatorial explosion of the first assumption is avoided and the effort wasted initially modelling without compound agents is also avoided by going straight to assumption three. Where intuitions are not available, or are considered to be suspect, then it may be more economical to use the second assumption plus ethnographic analysis to identify reasonable compound groupings.

Identify constraints
The Interaction Framework description may be manipulated to explore both canonical and abnormal event trajectories under the conditions to be explored. This will reveal the constraints that result from temporal factors or agent availability or performance. In this step we can also begin the knowledge needs analysis by identifying the consequences of the various agents not knowing the most appropriate sequence of interaction events or the consequences of their actions.

At the end of this stage key items of knowledge should have been identified. These will related to temporal constraints and the results of normal and abnormal event trajectories. This list of knowledge requirements can be used as a starting point for more detailed analysis carried out in the final step.
Apply tools to complete KNA
With the aid of the list of knowledge requirements and the Interaction Framework description additional KNA oriented tools such as programmable user models (PUMs), the Resources Model and Task Analysis can be applied to the scenario. It is expected that tools such as these, when used with Interaction Framework descriptions, could be used to inform the detailed design of devices, procedures, staff selection and training programmes. The tools, applied on their own, would not reveal the real-time and multi-agency constraints that apply to SMART systems; these elements are supplied by Interaction Framework. The KNA contribution from the three identified methods is outlined here and placed in a SMART context.

PUMs and PUMming makes use of IL, a programming language, to define an executable model of the user (Blandford and Young, 1995 & 1996). The execution relies on an assumption of means-ends analysis and various items of knowledge that the user is believed to have. These items of knowledge are variously categorised and include:

Knowledge of objects

- conceptual types and specific instances of object. In the demonstrator the air traffic control officer could be expected to have classes of aircraft as objects (e.g. commercial, private, military) and specific instances (e.g. the re-routed aircraft in the demonstrator). Other agents could also be objects. Objects may also be abstract items such as deadlines, temporal constraints.

relationships between objects

This includes, but is not limited to, temporal relationships or physical relationships; examples of which can be seen in the demonstrator scenario.

functions and predicates

In the demonstrator the air traffic control officer must be aware of what it is possible to ask flight
crews to do and what things hold true about the aircraft under their control.

User knowledge specification
conceptual operations
This, depending on the identity of each knowledge item, could include such things as: issuing instructions, making manoeuvres, changing air speed, stacking aircraft, etc.

initial knowledge
functions and predicates that hold true for the domain: aviation law, aviation physics, some aspects of operational procedures.

task [or task set]
functions and predicates that apply in the goal state, e.g. aircraft fuel load, heading, separation from other aircraft.

Device(s)
device commands
This includes command availability, effects and side effects. For a SMART system there will be a number of "devices" to consider. The set of devices will include computer equipment and also human agents. Interaction Framework analysis provides useful information on command availability as it provides temporal information rather than non-temporal state information.

initial state
In Interaction Framework terms this will be a description of the device (agent) in state p0. For the demonstrator this will include the heading of the aircraft to be re-routed and knowledge of its capabilities.

display
Information directly displayed by the device. For the aircraft, from the air traffic control officer's
perspective, this could include current radio transmissions or its radar image. For the flight crew instruments will have displays and human agents will be observable as busy or not busy.

The human agents in the system may know in a number of ways: prior learning, direct observation, tracking, and inference. An Interaction Framework analysis carried out prior to a PUMs analysis will provide information that could be encoded in IL so that, with a suitable temporally aware execution environment, the limits of the knowledge that human agents have can be explored in a real-time environment.

RM links the information resources required (plan, system state, goal(s), etc.) by the user for successful interaction given a particular style of interaction (planning, plan following, exploration, etc.) (Fields et al. 1996b). In addition consideration is given to the form, or physical expression, of each information resource. The strategies and information resources described in the Resources Model can be considered to apply to SMART systems as follows:

**Information resources**

**plans**  action list or sequence needed to achieve a goal. In a SMART system this could be a procedure to follow or a previously known event trajectory. Plans may be written in procedure manuals, exist in the heads of human agents, or be embedded in computer systems.

**goals**  describes the desired state. Corresponds to $p_1$ in the Interaction Framework description of objectives. In the demonstrator the air traffic control officer's goal is partly enduring (maintain the rules of safe flight, etc.) and partly contextual (avoid a near miss/collision).

**action affordance**  set of possible [next] actions. In the aviation scenario many of the action affordance will be a
limited set of actions defined by the physics of aviation, aviation law and the context of the situation.

interaction history
list of actions already taken. Typically this will exist as a mental trace - the memory of recent events existing in the heads of human agents. In some cases an on-line log of events may be available.

action-effect mapping
statement of the effect of an action. As with interaction history this is most likely to reside in the heads of human agents and results from training and experience. Exceptions to this are where some augmentation is provided in the form of simulation that allows the effects of actions to be previewed prior to execution. While simulation is not generally available in live air traffic control situations it has been used in space exploration and in the control of nuclear power plant.

current system state
collected relevant attributes of objects. In the demonstrator this is likely to be a combination of key knowledge internalised by human agents and instrumentation displays to be referred to when required. Interaction Framework describes key information about the current system state when agent availability is defined. This knowledge may be internalised by agents or it may be directly visible to them, as is mutually the case with the flight crew.

The mapping between interaction strategies and information resources (Fields et al, 1996b), first listed in chapter three, is:
plan following: plan, system state;
planning: goals, system state, action-effects, affordances;
semantic matching: goal, system state, action-effects, affordances;
goal-directed exploration: goals, action history, system state, action-effects, affordances;
learning by exploring: action history, system state, affordances.

In many SMART systems the criticality of system operation, system value and temporal constraints may make some of these strategies inappropriate. It is unlikely that an air traffic control officer will use a live flight to try out a "learning by exploring" interaction strategy. However, plan based and semantic matching approaches may be appropriate. Plan based approaches may rely on previously learned procedures and event trajectories, or on the development of appropriate event trajectories as may be required in a novel situation. Novel situations may also make semantic matching or goal-directed exploration viable methods, given sufficient domain knowledge on the part of the human agents.

Task analysis is a broad heading that could be applied to a range of methods, a number of which were considered in chapter three. Those considered to be potentially useful in support of KNA are:

- Hierarchical Task Analysis, HTA (Shepherd, 1989)
- Task Action Grammar, TAG (Payne & Green 1989)
- Task Analysis for Knowledge Descriptions, TAKD (Diaper, 1989a)
- Knowledge Analysis of Tasks, KAT (Johnson, 1989)
- Analysis of Task Object Modelling, ATOM (Walsh, 1989)

By briefly reviewing the KNA aspects of each of these in turn an overview of the field of task analysis can be obtained.
HTA
plans and sequences  The focus here is on the knowledge of plans and sequences of actions that a user must know to achieve a given goal. Some consideration is given to branching: conditional steps, repetition, etc. While not very strong on temporal issues there may be some support here for generating the objective interaction event hierarchy and as such may support the design of training programmes.

TAG
mapping between tasks and actions
The creation of a grammar, to describe either the full set of mappings or just those known to a given user, can be used as a partial description of possible interactions. In a SMART context this seems to be appropriate at the level of individual interaction events rather than event trajectories. A TAG could be used to describe any interaction between human agents and either other human agents or computer agents.

generative capability
For a TAG to be usefully generative its rules need to be regular, consistent and easily understood. When such a grammar exists it will be easier for a human agent to generate a mapping where one is needed that has not previously been learned.

TAKD
objects  may be compound or atomic (e.g. a keyboard). One of the listed problems for this method is distinguishing between important and unimportant objects. A knowledge representation grammar (KRG) is used to provide a taxonomy of objects.
actions are directed towards specific objects. The full content of some actions can be difficult to determine as they sometimes contain obscure cognitive or perceptual elements that may need some later cognitive walkthrough to identify. Actions are typically low level device operations rather than higher level cognitive processes.

sequences sequence representation grammars (SRG) are applied to the objects described in a KRG, defining common and allowable sequences of actions.

CLG conceptual component concerned with the purpose of the system (task level) and system objects. This can be related to the objectives and agents described in Interaction Framework. This component of CLG also describes conceptual operations on objects.

communication component describes the structure of the language used to communicate between user and device (syntactic level). The interaction level of the component is concerned with lower level dialogue descriptions based on “interaction constituents”.

physical component describes the physical layout and design of interaction devices. While this may apply to computer agents it is difficult to see how it would apply to human agents, except perhaps in terms of ergonomics. It is not seen as an issue for KNA.

These task analysis methods can be summarised so that an overview, albeit a sparse overview, can be obtained. Firstly, it seems appropriate to reiterate that none of the task analysis methods provides a well structured means of dealing with temporal issues. Secondly, there are
a number of common features within the methods described above. System objects are described, as are interaction grammars. These two components describe the conceptual objects a human agent has to know about and interact with and the legitimate ways that those interactions can take place. This knowledge, when coupled with known goals or objectives leads to the forming of plans or the identification of appropriate procedures to apply in given circumstances. Interaction Framework places agents, objectives and interaction events into a temporal context that may then be further resolved using task analysis tools so that at a given time the objects and agents available and the allowable actions can be determined.

Evaluation
It is unrealistic to expect that the KNA steps detailed above, as with any other design method, would provide full and complete results the first time through. In order to increase confidence in the results, and to reveal pre-production faults, it seems appropriate to expect the results of the KNA process will be evaluated through prototyping or desk checking. In either case this would be supported by the domain specialists involved in scenario selection, or specialists of the same type.

5.3.2. The KNA gains from Interaction Framework application

A number of knowledge needs analysis gains can be made through the application of Interaction Framework to target systems:

- the interaction events required to achieve the current objective;
- the identity of the agent or agents that need to be interacted with;
- knowledge of the availability of agents over time;
- the amount of time available for interaction events / tasks.
By making these requirements explicit Interaction Framework provides an analogy for a designer's intended procedure that takes account of the needs of multi-agent systems operating under time constraints. Sequences of interaction events can be determined and the need for ordering and concurrency can be made clear. Where agents are required to make decisions regarding the use of other agents to achieve objectives then this knowledge is essential if interaction events are not to be misdirected.

Interaction Framework does not provide a definition of all of the knowledge needs of an agent but provides a scheme for identifying essential knowledge about interaction. Specific knowledge needs about how to interact with specific agents can be gained by applying other methods. However, the framework provides some input to these methods by making time constraints explicit and clarifying the objectives to be achieved.

5.3.3. Real-time issues

The definition of real-time adopted in this thesis is that the time required for performance, the time constraints, is approximately equal to the time taken by the agent to perform. This was first considered in chapter two. Interaction Framework provides a mechanism for making time constraints explicit. Thus if an essential part of an objective is that it must be achieved by a given time then system designers must ensure that this is possible. However, there may be occasions when some condition, such as a fault or other unforeseen circumstance, either reduces the time available or increases the performance time. The agents in the system must be aware of such changes if the objective is to be achieved. Where the objective cannot be achieved then consideration must be given to whether safety or liveness are the overriding requirements.

This can be seen if we consider a situation where an agent would normally initiate an event trajectory to achieve a specific objective. If
the time constraints have been tightened then the agent needs to know this so that an alternative trajectory may be launched if the available time is insufficient. Alternatively, if one of the other agents that would be involved in the trajectory is unavailable, or their performance is below what is required then yet another trajectory may need to be identified. These decisions can only be made if the initiating agent has knowledge of relevant temporal factors. Some of this temporal knowledge may be enduring, such as the performance of other agents and the time taken to execute certain procedures. Other temporal knowledge is transient in that it only applies in the context of the current situation.

In designing systems where temporal factors are significant we can take account of psychological knowledge so that appropriate design decisions are made. Where time is very highly constrained systems should not be designed that require complex cognitive processing on the part of human agents. This suggests automation should be used or the human users should be appropriately trained so that decision making in these circumstances is automatic or highly proceduralised. It is also necessary to ensure that agents have the information they require at the time it is required. This may be achieved by making information directly available or by ensuring that agents have appropriate internal mechanisms, such as inference or tracking. Where inference or tracking are required then designers should be aware of the potential for task context to affect these internal processes. The structure of events within a time period and environmental factors such as noise may affect perception of time and other cognitive performance; resulting in sub-optimal performance.

From a designer’s point of view these issues mean that consideration needs to be given to what an agent needs to know and to the circumstances under which this knowledge is needed. From this point the designer can consider how the knowledge is to be available to the agent. For computer agents this is a matter of knowledge representation and algorithms. The situation is more complicated for
human agents where internal processes and their manifestation in performance are less certain.
5.4. Issues Not Covered

A number of issues relevant to SMART systems have been identified, but not explored in depth. The most prominent of these is the psychological aspects of time; a large and complex area only briefly considered previously. Another issue is negotiation between agents. This may exist where human agents need to plan and compromise, but is not yet a feature of interactions with computer agents. A third issue is specific causes of failure of individual interaction events.

5.4.1. Psychological issues

In the preceding chapter some aspects of the psychology of time were briefly discussed. However, the purpose of that discussion was as a pointer to unanswered questions rather than as an attempt to resolve problems. There are other psychological issues that have not been addressed that are potentially relevant to designers of SMART systems. Neither Interaction Framework nor KNA address the psychological issues that may be significant in SMART systems. However, by identifying areas of system operation where time is significant it allows for psychological knowledge to be applied in a focused way. For example, where significant events occur in close proximity the effects of masking can be explored (Patterson, 1990). Describing systems using Interaction Framework could, potentially, reveal the possibility of time-constrained non-standard situations, i.e. those where the user has to make cognitive judgements where the time allowed may be inadequate. Such situations are likely to result in performance errors.

5.4.2. Negotiation between agents

One strength of human agents is their ability to co-operate and negotiate; particularly in novel situations, or when goals conflict. An example of this was identified in the pilot study where there was conflict between the need for both low inventory value and low unit costs (see chapter 2). Another reason for negotiation is when the best
solution to a problem needs to be found. This may occur when a resource, such as a tool or another agent, is not available to support the originally desired event trajectory. Thus there may be times in the operation of SMART systems when negotiation is required. However, Interaction Framework does not describe such negotiations, it only describes the (time constrained) circumstances under which it may be needed. While negotiation may be necessary to find a solution to a problem the nature of the problem may change and the time available to find a solution may be limited as a result of real-time factors. By using Interaction Framework representations of a system a range of scenarios that may require negotiation may be identified and explored, e.g. trajectory convergence; and the consequences of each outcome of negotiation tested. Such an approach has similarities with the sensitivity analysis carried out by business managers (Davis and Olson, 1984).

5.4.3. Reasons for the failure of individual interaction events

Interaction events may fail for a number of reasons. For example, the first two interaction events in the demonstrator have a semantic link - e2 is a reply to e1. If the semantics of e2 are not appropriate then, as was previously discussed, the air traffic control officer may consider that e1 did not succeed. The reasons for this vary from the technical (transmitter or receiver frequencies were inappropriately set), to the psychological (the flight crew did not register or understand the meaning of the message), and the social (the flight crew decided that they had a more important message and that an appropriate response to e1 would follow later). The reasons for the failure of an interaction event are more appropriately considered once the significance or otherwise of event failure is known. Interaction Framework assists with this analysis and may be used to reveal those events that are highly significant (e.g. that may prevent objectives being reached) and that, subsequently, can be studied to identify potential causes for failure. Targeted events may be subject to analysis from a systems point of view by examining component performance, system coupling...
and feedback issues. Psychological studies may reveal the cognitive issues around specific events as situated in their trajectory context. Social studies may reveal important aspects of interactions between multiple agents that will affect performance and that can be improved.
5.5. Future work

A number of areas for future work are considered. These are as diverse as linking Interaction Framework to ethnographic approaches, identifying and quantifying the benefits of applying Interaction Framework and the automation of Interaction Framework.

5.5.1. Interaction Framework and ethnography

Hughes et al (Hughes et al, 1994) suggested that ethnographic studies are too unsystematic, do not support the needs of software engineers and give results that are not transferable between contexts. The studies are highly qualitative. This contrasts with Interaction Framework which is quantitative, highly reductionist and relies on a formal approach to describe systems. The two approaches do have one feature in common, each considers "interaction" to be highly significant to system performance. This suggests an area for future work would be a study of the potential for bringing these two approaches together. Such a dualistic approach to system modelling may provide new information on how systems work; usable links between qualitative and quantitative factors may be formed. By linking a socially oriented approach, such as ethnography, to a formal method, Interaction Framework, it may be possible to formalise the effects on systems of such phenomena as person, i.e. agent, perception (Insko & Schopler, 1972) and group coherence (Secord & Backman, 1974). Such work on coherence would examine how factors affecting coherence determine the achievement of objectives. Recent work on coherence by Furuta and Kondo (Furuta and Kondo, 1992) has taken a highly mathematical approach to modelling system performance in real-time environments; although the effects of real-time have not been a factor for their consideration. This suggests that there is room for an Interaction Framework - ethnographic approach.
5.5.2. The benefits of applying Interaction Framework

Interaction Framework should be applied to live systems in design and maintenance phases. This would provide further data on its application and consequent benefits. A range of domains from financial and commercial to industrial and safety critical should be used to provide evidence for the limits of applicability and usefulness of Interaction Framework. In carrying out such work further extensions to Interaction Framework may be identified.

The relatively small notation set and the mathematical approach of Interaction Framework may make it suitable for automation. The development of computational models of systems based on Interaction Framework structures should support the modelling of large systems and the automatic description of knowledge requirements. Such computational models may also identify other properties of SMART systems that are beyond the capabilities of those that are manually manipulated. Further work is required to identify suitable representations for Interaction Framework objects and an inference mechanism.

If the Framework is to be applied by the HCI community some further work is needed to identify an appropriate delivery mechanism. A number of issues have been identified (Buckingham Shum & Hammond, 1994) relating to the difficulty of transferring potentially useful tools and techniques into practically useful tools:

"• How can we communicate what different models do?
• How do different models relate to each other?
• How do different models fit into the development process?
• What process is involved in applying a modelling/analytic approach?
• What knowledge and expertise is needed to make effective use of a modelling approach?
• How useful is the modelling to development teams?"

(ibid. pg. 22)
While this thesis addresses some of these issues it does so from the point of view of a specific domain and, sometimes, from the point of view of a particular method: Interaction Framework. The framework developed by Buckingham Shum & Hammond (ibid.) examines four "gulfs" between what is required by designers and what is provided by modellers. A step in the direction of making Interaction Framework more useful would be to subject it to detailed study under the gulf framework. Should a study would allow the benefits of Interaction Framework to be communicated to a wider community.

The original purpose of the Framework was to provide an agent neutral description of multi-agent systems that would highlight such system properties as interaction detours or the need for a system reset. This analysis would be used by designers to improve design, particularly from a human factors perspective. In this thesis an additional purpose has been found: support for knowledge needs analysis. However, in considering the benefits of Interaction Framework in the context of what is provided by other analytical modelling methods a further purpose has come to light. The Framework may provide a much broader support for analysis by indicating where other methods might be used. In chapter three a "holy grail" methodology is mentioned in the context of Brun and Beaudouin-Lafon's (Brun and Beaudouin-Lafon, 1995) approach to system classification. It seems unlikely that such an all embracing methodology, answering all design questions, could exist. In contrast, Interaction Framework may provide a useful, if not universal, framework that enables specific tools or sets of tools to be applied most appropriately to those parts of systems that could most benefit. Where interaction between agents is complex and may require negotiation or conflict resolution then techniques developed in ethnomethodology or artificial intelligence may be applied. Where a human user needs to use particular devices then PUMs, the Resources Model, task analysis or other techniques may be applied. Real-time elements of system operation may benefit from detailed study by cognitive engineers or human performance specialists.
5.6. Conclusions

In this thesis a group of interactive systems, SMART systems, has been described. For these systems to function it is necessary for their agents to have both specific domain knowledge and skills, and to have knowledge of the requirements and constraints caused by real-time factors. To support the identification of this knowledge a modelling method, Interaction Framework, has been identified using criteria developed for the purpose. The Framework has also been modified to extend its usefulness beyond the original intention of its designers. The Interaction Framework was designed to provide an understanding of the properties of interactive systems, it can also be used, as has been demonstrated, to support knowledge needs analysis with respect to real-time factors. The Framework is flexible and neither computer nor user centric; the modifications do not alter this "neutral view" characteristic. While Interaction Framework has been demonstrated to be applicable to aviation it may also be applied to systems of lesser or greater complexity. The models of the systems produced using Interaction Framework, and the resultant KNA assist with the selection and training of human agents and with the design of computer agents. As such Interaction Framework provides a range of benefits to systems developers and provides another tool to make working on a set of difficult problems easier.
Appendix 5.1

Worked Example Demonstrating Application of the Method

The pages in this appendix give an example of the output of each step of the method as applied to a trivial example. The purpose of this is to provide an indication of the activity of each step and the type of output that can be expected. It is not intended that the example should demonstrate all of the features of the Interaction Framework, knowledge needs analysis, or the potential of the method to handle complex examples.

The example is based on a hypothetical telephone sales operation. A number of sales staff use an information system to process customers orders. Customers wish to have their orders processed promptly. However, the operating speed of the information system is variable, and it sometimes stalls. A more complete description is contained in the step illustrations.
Method Step 1) Identify Key Scenarios.

i) Identify Personnel To Be Used
Personnel to be involved in developing and describing a scenario are: representatives of the sales staff and representatives of the customers, for their views on the “customer experience”.

ii) Arrange Discussion/Focus Groups To Identify Key Problems
The views of the people involved were:

Sales Staff
Monday mornings are used by our wholesale customers to order items that they have found out they need over the weekend, either as a result of demand or from high sales volumes. In effect, there is a two day backlog of needs that customers want to turn into orders. On Thursdays they want to get orders in for delivery before the weekend. So, orders are bigger and from a greater number of customers. When they do get through, the information system is often busy and getting busier. Though we have several sales staff, who answer phones quickly, the customers complain that service is slow. It’s not the fault of the staff, the system is slow and sometimes stalls without us noticing.

Customers
After a weekend of selling there are usually a number of items we need to order so that we can restock as soon as possible. The trouble is that when we phone in it can be difficult to get through; and when we do the computer is quite slow. We’d like the order over and done with in five minutes. Occasionally we wait for ages before the sales person says that the computer has gone down and they’ll call back later when the system is up.
iii) Develop Scenario That Contains The Key Problems

When customers make telephone enquiries they require prompt and accurate responses. It is in the supplier's interests to provide such responses as each sale, and continued custom, may depend on the speedy and accurate provision of information. Customer calls are made in real-time, and they may be made at any time during the business day. However, there are points in the week when demand peaks: Monday morning, Thursday afternoon. A significant system upgrade is not part of the information systems strategy.

The several sales staff support customers by using a computerised information system that indicates current product information and stock levels and also supports the ordering process. Figure 5.1 is a schematic representation of the system.

![Figure 5.1: Sales system](image-url)
A typical situation is as follows:

1) a customer calls the company;
2) the call is answered and the sales process begins, the performance of the system is acceptably fast;
3) a second customer calls, the sales process begins and the performance of the system begins to degrade;
4) further customers call and performance degrades to the point where it does not meet customers' needs for prompt service;
5) occasionally, for non-sales related reasons, the system stalls and ceases to provide any service; screen output is frozen and there is no response to input.

Sometimes sales staff will be waiting for a slow machine that may have stalled, or time may be wasted working a slow system when it would be better to call back when the system is working more quickly.
Method Step 2) Describe Scenario Using Interaction Framework.

i) List Agents, Objectives

**agents**
customer
g1) order all the necessary items within a five minute phone call

\[ t_1 \]
\[ p_0 : \text{items not ordered} \]
\[ p_1 : \text{items ordered within five minutes} \]

sales person
information system
company\(^2\) : sales person + information system
g2) support customer needs

\[ t_2 \]
\[ p_0 : \text{items not ordered} \]
\[ p_1 : \text{required and orderable items are processed} \]

g3) provide businesslike service (efficient, courteous, etc.)

\[ t_3 \]
\[ p_0 : \text{customer starts ordering process} \]
\[ p_1 : \text{ordering process completed & customer has positive image of the organisation} \]

ii) Produce Event Trajectories
Steps in the scenario:

s1) customer makes call to sales staff
s2) account located on information system
s3) customer provides details of order
s4) details of order confirmed
s5) sales process closed
In 3), 4) and 5) both the other goals are also blocked: g2 and g3, supporting customer needs and providing businesslike service. This is because interaction events e3, e5 and e6 respectively terminate abnormally. In these cases it may be more appropriate to introduce a new interaction event:

\[ e_8 : \text{information system signifies that it has stalled.} \]

This results from a perception on the part of the sales staff that the interval between information system input and output is unacceptably long and probably stalled. This leads 3), 4) and 5) to be rewritten as:

3') \[ E'_2 = < e_1 \triangleright e_2 \triangleright e_8 \triangleright A_t \triangleright t_1 ( \& \ A_t \& \ A_t \& t_3 ) \]

4') \[ E'_3 = < e_1 \triangleright e_2 \triangleright e_3 \triangleright e_4 \triangleright e_5 \triangleright e_8 \triangleright A_t \triangleright t_1 ( \& \ A_t \& \ A_t \& t_3 ) \]

5') \[ E'_4 = < e_1 \triangleright e_2 \triangleright e_3 \triangleright e_4 \triangleright e_5 \triangleright e_6 \triangleright e_7 \triangleright e_8 \triangleright A_t \triangleright t_1 ( \& \ A_t \& \ A_t \& t_3 ) \]

Where the system is performing at an intermediate level, i.e. slow, but not yet stalled then we have:

6) \[ E_5 = < e_1 \triangleright e_2 \triangleright e_3 \triangleright e_4 \triangleright e_5 \triangleright e_6 \triangleright e_7 \triangleright e_8 > t_2 ( \& \ A_t \& \ A_t \& t_3 ) \]

\{ order items processed ............. \}

7) completion(e8) - time(e2) > five minutes

...BUT NOT within five minutes

The critical interaction events for slow performance of the information system are: e3, e5 and e6 given that the other agents are motivated to perform quickly. In these same interaction events, the duration will be determined both by the performance of the system and the time taken for sales staff to perceive the output.
Where performance is unacceptably slow or stalled then if \( t_3 \) is to be achieved, in addition to \( t_1 \) and \( t_2 \), the sales staff need to detect the level of performance and abandon the current trajectory with a view to restarting it (\( e_a \)). The interaction will be restarted by the sales staff (\( e_r \)) when the information system is performing at an acceptable level. For example:

8) \( E_6 = < e_1 \ e_2 \ e_3 \ e_5 \ e_6 \ e_7 \ e_8 \supset t_1 \)
   \[ ( \& \ \neg t_2 \ \& \ \neg t_3 ) \]

\( E_6 \) is preferable to \( E_3 \), \( E_4 \) or \( E_5 \) as all objectives are met despite stalled or slow performance.
Method Step 3) Identify Constraints On Canonical Event Trajectories.

i) List Problem Areas and the Factors Causing Them.

The main factor preventing canonical event trajectories is the poor performance of the information system. This cannot be resolved with the current Information Systems strategy.

The next preferable event trajectory after the canonical E₁ is E₆. Instances of E₆ are limited by the ability of sales staff to perceive poor system performance, stalled or slow running, as signified by E₅. Where they fail to observe poor performance and terminate the interaction gracefully then faulty interactions occur, e.g. E₅ such that interactions take too long and make all objectives unachievable.
Step 4) Apply Non-Interaction Framework Tools To Complete KNA.

i) Identify Characteristics Of The Problem.
The difficulty with the system lies in the user (sales staff) use of the information system. The system may fail in two ways which are not always easy to distinguish: the system may be slow or it may be stalled. The nature of the sales task is one of following standard procedures to support customers.

ii) Select Most Apt Tool For The Problem
The Resources Model can be applied here. The following of sales procedures can be translated in Resources Model terms as a “plan following” strategy.

iii) Apply Tool To Complete KNA
The knowledge resources required for a plan following strategy are: the plan (in this case the sales procedure) and the system state.

The Resources Model makes the issue of system state explicit. For an acceptable interaction, with a plan following interaction such as this, the state of the system must be available to the user. In this case it is not. It is not clear to the user of a stalled system that it has stalled and thus it is not clear that trajectory abandonment or system reset is the most appropriate action to take. This stems from the graceful degradation of the system; which in turn makes tracking both difficult and unreliable. Awareness of a stall using inference provides the necessary knowledge - but too late to support objectives. These problems, taken together, suggest that the information should be displayed directly rather than relying on the users' internal resources.

To eliminate this problem some output from the system could indicate that it is not stalled. This could be in the form of a flashing data item, e.g. date/time. When the system stalls the item will not flash and will so alert the user to a stalled system.
The proposed solution is that the system should have an additional display element that conveys the following information:

- presence / absence of stalling;
- an "analogue" display allowing users to differentiate fast system operation from slow, unacceptably slow, and stalled.
Method Step 5) Evaluation.

i) refer findings back to domain specialists or make use of prototyping methods.
ii) determine if KNA is reasonable or if steps 2) to 5) need to be repeated.

The proposal to include a "performance-o-meter" in the interface has been received favourably by sales staff. A prototype should be produced to determine if this would be acceptable in practice. If the performance-o-meter is acceptable then the prototype will also allow us to determine the fine detail of its implementation.

Criteria to be applied for testing the new feature will include a significant improvement in the detection rate of slow and stalled systems. Assuming it is adopted on the live system then customer perceptions of service before and after the change will also need to be determined.
ACKNOWLEDGEMENTS

The support of Dennis Dunn, Head of the Department of Business Information Technology is particularly acknowledged.

Thanks to Professor Ernest Edmonds for supervising and providing clear and concise advice. Thanks also to Doctor Helmut Bez for pointing out potential pitfalls in the system of working.

The support of Ann Blandford is gratefully acknowledged in giving her time to discuss many of the theoretical issues. Thanks also to Sid Dekker for his assistance.

To all those individuals that gave their time in interviews and allowed themselves to be observed I also offer thanks.

Finally, my greatest thanks to my wife Chris who has patiently let me work and who has sometimes had to give me more vigorous encouragement over the years.
References


Fields B., Wright P., Harrison M. (1996) "Objectives, strategies and resources as design drivers" obtained from the following world-wide-web site http://www.cs.york.ac.uk/~bob


HCI Service (1991) "HCI tools and methods handbook", DTI.


KNA for SMART Systems

References


