Reasoning about Contingent Events in Distributed Systems

Ian Anthony Benson
King's College

A thesis submitted for the degree of
Doctor of Philosophy in the University of Cambridge
November 1991
Abstract

This dissertation is concerned with requirements analysis and design for open distributed systems. These are systems which are hacked about over time consisting of people, computers and other equipment collaborating on shared activities through a variety of media. The dissertation describes a theoretical framework for distinguishing and reasoning about patterns of action and interaction all the way from the computer systems to the processes of system evolution. It reports the validation of the approach in the design of a wide area communication system for an itinerant workforce of flight crew.

The structure and behaviour of software and social processes is abstracted using an object oriented language, BETA, with conventions for concurrency, object aggregation, communication, classification and generalisation. The language is given an operational semantics in Milner's process calculus CCS. A proof assistant developed by Cleaveland is used to establish temporal properties of a domain expressed in the propositional mu-calculus. The power and versatility of this modelling apparatus is demonstrated by a reconstruction of a lengthy manual mutual exclusion proof.

This machinery is used as the basis for a method of system development. The thesis identifies a taxonomy of architectural components, assembling them in a manner suggested by Winograd's language/action theory and Ciborra's transaction cost analysis. The inventory includes social roles, organisational units, distributed services and communication media. A domain is specified as a network of transactions between roles which may be performed by people or machines. The modelling apparatus allows us to systematically enumerate breakdowns, bottlenecks, gaps and inefficiencies in their performance. These perceptions allow system architects and their clients to envisage the kind of system support that may be needed to streamline transactions, improve them or even to change the organisation.

Contingency in social and machine communication is central to reasoning about the fabric of transactions that articulate a workplace. To highlight these mismatches in a form that is accessible to people who expect to play a part in the organisation I simulate transaction networks using hypercard. By giving hypercard an operational semantics in CCS the simulation becomes a specification formally related to the pseudo-code description in BETA.

A case study is reported of the application of the method to the design of a wide area multimedia system. The domain is modelled in pseudo-code as a recursive, concurrent invocation of a communicating subsystem. The hypercard simulation is analysed to identify where breakdowns might occur in the field and the results of observations are expressed in the form of a spreadsheet. The simulation is used to negotiate the resource and technical requirements of the new system which will be created by translating it into executable code. The dissertation concludes with a discussion of the design of a CASE tool and an advanced scripting environment for analysis and executable specification that is suggested by the work.
Contents

1. Coordination and Contingency ............................................................................... 1
   1.1. Theoretical Work 1
   1.2. Field Work 5
   1.3. Organisation of the Dissertation 7
   1.4. How to read the Thesis 10

2. Previous Work ...........................................................................................................13
   2.1. Theories for System Description 15
   2.2. Modelling and Verification 19
      2.2.1. Basic Concepts 20
      2.2.2. Communicating Sequential Processes 25
      2.2.3. Program Logics 27
      2.2.3. Group Element Model 38
      2.2.4. Mechanised Program Proving 41
   2.3. Construction 42
      2.3.1. Architectural Concepts 43
      2.3.2. Object Specification 47
      2.3.3. Object Class 48
      2.3.4. Groupware 49
   2.4. Enterprise and Information Models 55
      2.4.1. Organisational Contexts 55
      2.4.2. Modelling Techniques 58
      2.4.3. Transaction Cost Theory 63
   2.5. Conclusion 66

3. Modelling Language ..................................................................................................69
   3.1. Process Calculus 70
      3.1.1. Transition Semantics 72
      3.1.2. Algebraic Laws and Milner's Graphical Notation 73
      3.1.3. Modelling Value Transfer and Functions 78
      3.1.4. Behavioural Equivalence 79
   3.2. Foundations of the Domain Model 80
      3.2.1. Patterns 82
      3.2.2. Objects 83
      3.2.3. Knowledge Representation in BETA 84
      3.2.4. Local Events 84
      3.2.5. Communicating Systems 90
   3.3. BETA pseudo-code and its Behavioural Semantics 95
   3.4. Structural Abstraction 97
      3.4.1. Aggregation 98
      3.4.2. Classification 99
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.3. Hierarchical Generalisation</td>
<td>99</td>
</tr>
<tr>
<td>3.4.4. Semantics for Structural Abstraction</td>
<td>101</td>
</tr>
<tr>
<td>3.5. Conclusion</td>
<td>102</td>
</tr>
<tr>
<td>4. Inference System</td>
<td>105</td>
</tr>
<tr>
<td>4.1. Modal Mu-Calculus</td>
<td>107</td>
</tr>
<tr>
<td>4.2. Properties</td>
<td>108</td>
</tr>
<tr>
<td>4.2.1. Local Properties</td>
<td>108</td>
</tr>
<tr>
<td>4.2.2. Global Properties</td>
<td>109</td>
</tr>
<tr>
<td>4.3. Proof System</td>
<td>111</td>
</tr>
<tr>
<td>4.4. A Sample Tableau</td>
<td>114</td>
</tr>
<tr>
<td>4.5. Illustration: Database Server</td>
<td>115</td>
</tr>
<tr>
<td>4.5.1. BETA Code</td>
<td>115</td>
</tr>
<tr>
<td>4.5.2. CCS Representation</td>
<td>115</td>
</tr>
<tr>
<td>4.5.3. Mutex Property</td>
<td>116</td>
</tr>
<tr>
<td>4.6. Conclusion</td>
<td>117</td>
</tr>
<tr>
<td>5. Domain Model</td>
<td>119</td>
</tr>
<tr>
<td>5.1. Domain Description</td>
<td>121</td>
</tr>
<tr>
<td>5.1.1. Role Model</td>
<td>123</td>
</tr>
<tr>
<td>5.1.2. Link Model</td>
<td>126</td>
</tr>
<tr>
<td>5.1.3. Transactions and Contingent Events</td>
<td>127</td>
</tr>
<tr>
<td>5.1.4. Coordination Structures</td>
<td>128</td>
</tr>
<tr>
<td>5.2. Projection</td>
<td>132</td>
</tr>
<tr>
<td>5.2.1. Social Projection</td>
<td>133</td>
</tr>
<tr>
<td>5.2.2. Program Projection</td>
<td>135</td>
</tr>
<tr>
<td>5.2.3. Physical Projection</td>
<td>138</td>
</tr>
<tr>
<td>5.3. Process Model</td>
<td>140</td>
</tr>
<tr>
<td>5.3.1. Simulation of Transaction Networks</td>
<td>141</td>
</tr>
<tr>
<td>5.4. Conclusion</td>
<td>143</td>
</tr>
<tr>
<td>6. Case Study in System Development</td>
<td>145</td>
</tr>
<tr>
<td>6.1. Airline Systems</td>
<td>147</td>
</tr>
<tr>
<td>6.2. Domain Overview</td>
<td>149</td>
</tr>
<tr>
<td>6.2.1. Site Survey</td>
<td>150</td>
</tr>
<tr>
<td>6.2.2. Information Management</td>
<td>151</td>
</tr>
<tr>
<td>6.2.3. Passenger Revenue Accounting</td>
<td>152</td>
</tr>
<tr>
<td>6.2.4. Software Development</td>
<td>153</td>
</tr>
<tr>
<td>6.2.5. Yield Management</td>
<td>153</td>
</tr>
<tr>
<td>6.3. The Flight Crew</td>
<td>154</td>
</tr>
<tr>
<td>6.3.1. Organisational Dimension</td>
<td>158</td>
</tr>
<tr>
<td>6.3.2. Task Dimension</td>
<td>161</td>
</tr>
<tr>
<td>6.3.3. Economic Dimension</td>
<td>163</td>
</tr>
<tr>
<td>6.3.4. Systems Strategy</td>
<td>165</td>
</tr>
<tr>
<td>6.4. Program Projection</td>
<td>167</td>
</tr>
<tr>
<td>6.4.1. Workflow Analysis</td>
<td>167</td>
</tr>
<tr>
<td>6.4.2. Transaction Cost Spreadsheet</td>
<td>171</td>
</tr>
</tbody>
</table>
List of Figures

2.1. Integrated Systems Architecture (ISA) projections 14
2.2. Sophistication of modelling languages 20
2.3. A Kripke structure 28
2.4. Computational objects interacting through an infrastructure 43
2.5. Basic invocation model 46
2.6. A spectrum of groupware applications 50
2.7. A layered architecture for groupware 55
2.8. Information flows between an administrative bureaucracy and its customers 56
2.9. The “information intermediary” 57
2.10. The basic conversation for action 61

3.1. A capacity one buffer 73
3.2. The composite system $A|B$ 76
3.3. A transition tree 77
3.4. A transition graph 77
3.5. A capacity two buffer 78
3.6. A multiplication function 78
3.7. A folded flow graph 87
3.8. Event partial order 90
3.9. RPC flow graph 92
3.10. Transition tree for database server 94
3.11. Decomposed transition tree for database server 95

4.1. A pair of dissimilar agents 109
4.2. A sample tableau 114

5.1. A groupware perspective on distributed processing 122
5.2. Subsystems of a communicating role 124
5.3. Link model 126
5.4. ISA notation for a link 127
5.5. A transaction in ATI notation 129
5.6. Basic transaction networks 130
5.7. ATI transaction patterns 131
5.8. A transaction in the social projection 135
5.9. A request workflow 136
5.10. Negotiate transaction and its transition graph 137
5.11. Currency request workflow 138
5.12. A network planning model 139
5.13. Development process model 140
5.14. A typical hypercard stack 142
5.15. Hypercard translation of negotiate transaction networks 143

6.1. Activities and results in incremental development 145
6.2. Desktop communication media at BA 148
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3. Main dimensions of the contractual relationship</td>
<td>155</td>
</tr>
<tr>
<td>6.4. The bidline routine</td>
<td>163</td>
</tr>
<tr>
<td>6.5. The bidline transactions</td>
<td>168</td>
</tr>
<tr>
<td>6.6. The bidline simulation</td>
<td>171</td>
</tr>
<tr>
<td>6.7. The bidline task menus</td>
<td>173</td>
</tr>
<tr>
<td>6.8. Open Package, Nominate Lines, Confirm Allocation</td>
<td>174</td>
</tr>
<tr>
<td>6.9. A basic voice/data bidding system</td>
<td>177</td>
</tr>
<tr>
<td>6.10. The multimedia internetwork</td>
<td>180</td>
</tr>
<tr>
<td>8.1. An integrated CASE tool</td>
<td>205</td>
</tr>
</tbody>
</table>
Chapter 1

Coordination and Contingency

This thesis is concerned with theories, methods and tools for the design of open distributed systems. These are systems in which programs and people communicate and carry out tasks concurrently, sharing information by message passing, common workspaces or real time communication over a variety of media such as voice, video, text or data.

The dissertation reports on research to construct a unified foundation for open systems analysis. Conceptual work was needed to describe properties and activities of computer communication in a way that would enable computer supported inference during design. It was first necessary to represent the design and secure the understanding of users who expected to play a role in the organisation. The resulting simulation was implemented in hypercard. The results were validated by a field test involving the analysis and design of a system of work allocation for over three thousand pilots and flight engineers at British Airways (BA). This involved the construction of an integrated multimedia model of the system for reconciling bids and offers for work, which was used to establish a new context for the work of flight crew and their managers.

1.1. Theoretical Work

Several groups of theories have been proposed for describing the structure, behaviour and information interdependencies of computer systems, and some exploratory tools for computer support have been constructed. Although these approaches have been of value in some areas, such as Computer Aided Software Engineering (CASE) tools for maintaining the syntactic consistency of design documentation, they have been much less effective in enterprise and office modelling and in user interface design.
My approach draws on insights from three strands of research:

**institutional economics.** This is a branch of economics concerned with the nature of social organisation and coordination. It is essential to enterprise and information modelling.

**object oriented system description.** This provides a general framework for capturing regularity in information processing and communication.

**computational logics.** These provide an apparatus for describing and reasoning about behaviour.

Ciborra and his colleagues have developed an approach to systems analysis based on viewing information requirements in terms of transaction cost [Ciborra, 1987]. Their concept of transaction, which they take from institutional economics, links information, uncertainty and organisation in an original way [Williamson, 1981]. They view open systems as mediating the network of transactions between people. Transaction cost enables the analyst to suggest the best fit between an organisation and the technology. Analysis can reflect the fact that the information used to transact and coordinate may be either candid or opportunistic depending on the level of trust between the contracting parties.

By focusing on social and economic relationships, transaction cost analysis provides a basic framework for the design of groupware, a class of distributed application made possible by open computing platforms. The term *groupware* has been coined to distinguish software cooperating over a network from standalone, or transaction processing applications. Language/action theory has been proposed for the detailed design of groupware systems, such as structured electronic mail [Winograd, Flores 1986]. This embodies Searle’s work on speech acts which classify utterances into a small number of categories which fall into a particular pattern in discourse [Searle, 1969]. A typical conversation would be represented as a transition system in which arcs are labelled by person and type of speech act.
My contribution has been to take institutional economics, and give it a formal underpinning so that it can take its place as a fully fledged method for systems analysis. My approach subsumes language/action theory as a special kind of transaction network. I refine and validate the method in the case study at BA. The formal basis of my work enables us to envisage CASE tools that could be an aid in specification, construction and tailoring of distributed applications.

The domain is described in a pseudo-code derived from BETA [Kristensen et al, 1990], a second generation object oriented language with features for the aggregation, classification and generalisation of objects required for large scale object management. Milner's process calculus CCS is used for its operational semantics [Milner, 1989]. It closely resembles BETA in its support for concurrency and synchronised message passing as the basic form of inter-object communication.

To communicate the design I found it necessary to give BETA a mathematically well founded graphical interface which I based on Milner's flow and transition graphs. Just as programming languages exist at different levels of abstraction from assembly code to abstract data types, so there is a need for a family of inter-related graphical notations.

The nature and contribution of visual abstraction should not be underestimated in working with software. For users, the desktop metaphor appears to draw its power from the apparent familiarity of its icons. In reality the power comes from the abstraction of the functions of the operating system and the applications. The desktop is more compelling than command line interfaces not because it does away with cryptic mnemonics, but because graphical input activity such as MouseDown and dragging is formally interpreted over an underlying structure of objects and methods.

Graphical notation has been found to be essential to designers too, as a means of communication within the project team. It will become even more important as the burden of system tailoring is shared with users who are not programmers.
The distributed system model extends the work of the Integrated Systems Architecture (ISA/ANSA) project [Herbert, 1989]. This project, which feeds into the open system standards process, establishes an architectural framework for the design of distributed systems and a basic set of distributed programming concepts such as service import/export (dynamic linking) and remote procedure call mechanisms.

The dissertation is the first attempt to bring these strands together. Here tools and techniques are described for the construction of integrated domain models to which organisational, program and physical projections can be formally related. Model construction focuses on representation and reasoning about transactions that give rise to contingent events.

The basic research problem is concerned with establishing two relationships: the specification relationship that holds between a program and the behaviour it engenders, and the primary relationship between a specification and the states of the world with which the program is concerned [Smith, 1987].

Behavioural specification has been an area of active research for the past decade, and many of its results are now beginning to appear in prototype analytical aids for system designers. Various logics have been investigated in which correct and erroneous behaviour can be specified in terms of required or potential event sequences. Attempts at establishing the specification relationship encounter difficulty in relating program objects to the logical framework of behavioural description. In the early Chapters I develop an apparatus for activity description which associates events and behavioural properties with objects and subsystems. This enables a composite analysis to be built up selectively from assertions concerning the behaviour of the parts.

Unfortunately specification is the least problematic of the two representation relationships encountered in systems development. The primary relationship, that is the relationship between a specification and the real world, is more intractable. The later Chapters of the dissertation show how the approach extends to developing simulation in hypercard as a specification that eases the capturing and maintenance of this relation.
The modelling language is used to delineate transactions. A primitive transaction, expressed as a BETA fragment, models a store and forward message between roles played by people or programs. A taxonomy of coordination structures is built up from these primitive transactions which subsumes Winograd’s conversation for action. The logical foundation I developed for BETA enables machine aided reasoning about breakdowns in coordination and communication in transaction network models. Transaction cost analysis has found observation of breakdown to be important in establishing where remedial system support might be introduced and these observations provide the basis for analysis in the real world.

I simulate transaction networks using hypercard. By annotating the cards to flesh out coordination and processing tasks, objects manipulated, and computer system interfaces, simulation serves to formally relate the enterprise, information and computational perspectives of ISA. Hypercard is an important tool in securing the understanding of people who expect to play a role in the organisation and forms the basis for executable specification. Computer system development can then be regarded as the incremental adjustment of the model in response to contingency in the organisation and in the physical world.

1.2. Field Work

Extensive field work was undertaken to develop and validate my approach to system description and analysis. A setting at British Airways was selected as one in which distribution was already the norm. I assembled a team consisting of Prof Ciborra and colleagues at BA and British Telecom. We started by looking at the nature of transactions between people in a variety of functional areas, using institutional economics and Winograd’s language/action theory to identify candidate domains for study [Winograd, Flores 1986].

This preliminary investigation led to the selection by BA of the flight crew domain. Flight crew managers were using traditional techniques of systems analysis to identify opportunities for computer support, and while these had led them to pursue a wide variety of discrete system developments, it had highlighted other areas where a business case did
not appear to be justified. The result was a fragmented communication and processing environment which contributed to management concern and a degree of staff alienation.

I constructed a transaction network model to characterise the breakdowns, gaps and inefficiencies in the relationships between pilots and the airline. My model of the economic transactions, the bidline market, took the form of a concurrent, recursive invocation of a communicating subsystem described in BETA pseudo-code.

This approach is a significant advance over previous techniques for organisational modelling grounded in institutional economics and language/action theory. While subsuming these methods, it goes on to expose subtle breakdowns such as processing failure arising from a lack of information sharing across concurrent threads, it scales to accommodate the hundreds of thousands of conversations in the domain and it provides executable specifications. While allowing that the unit of analysis is one-to-one dialogue, as it is in language/action theory, the general purpose nature of BETA allows much richer communication structures to be modelled, and directs the analyst’s attention towards issues of regulation and control which are not touched on by language/action.

A hypercard stack constructed from the model was used to simulate transactions from the perspective of the pilots and other staff. This enabled them to carry out a systematic evaluation of the consequences of breakdowns suggested by the model, so that we could create both the functional requirements for a system of computer support, and a successful case for the investment of corporate funds.

The contribution of the thesis lies in the following areas:

- extending a formulation for representing communicating concurrent systems to model a new setting
- applying techniques for reasoning about programs to a new programming language
- applying economic theory in a new situation
- designing a computer tool to support the method.

1.3. Organisation of the Dissertation

Chapter 2 reviews previous research. The central core lies in the considerable body of work on modelling communicating concurrent systems. The thesis draws together these approaches to reasoning about behaviour with work on object oriented analysis and design. It uses this framework to describe distributed architectures both from an engineering and an application perspective. The concept of groupware is introduced as a new class of computer application made possible by peer to peer and client/server architectures in which emphasis is put on social interaction through shared objects, such as messages, screens or databases.

The review highlights the two problems which I address in the dissertation. It shows how previous attempts at reasoning about programs encounter the *correspondence problem* - the difficulty in relating program objects to the logical framework of a behavioural description. Even when this difficulty is overcome by ad hoc means, formal proof that an implementation satisfies its specification is error prone, lengthy and tedious.

The second problem concerns the representation of the relationship between the information system and the organisation it supports. This is even more problematic, and has been the subject of study from many different research perspectives in addition to computer science. These studies have given rise to a variety of insights some of which draw on ideas from operating system and programming language theory in order to represent communication and coordination.

There is a general consensus among the research communities that users and designers need to understand both the form and the context of open systems. Design techniques need to distinguish communication from other activities and support concurrency and reasoning about contingent as well as routine events. Systems analysis needs to go beyond traditional data flow and decision support approaches to comprehend the ongoing enquiry, research and analysis actually involved in carrying out
procedures. Tools are needed to support design and user involvement in developing and tailoring systems.

Chapter 3 reviews CCS and the flow graph and derivation rules which are used to represent the operational semantics of BETA. BETA is introduced through an extended example in which a monitor is constructed by encapsulating a database within a guard. BETA is given an operational semantics in CCS in a translation which establishes an exact correspondence between objects in the domain and the action alphabet of CCS. This provides the basis for overcoming the correspondence problem faced by CCS [Stirling, Walker 1989], temporal logic [Manna, Pneuli 1982] and ad hoc formalisms such as [Lansky, 1983] with her GEM language.

Along with CCS, Milner developed with Hennessy a relativised modal logic which enabled local properties of CCS agents to be expressed. Recent research has extended Hennessy-Milner logic to a temporal logic in which both local and global properties can be expressed. In Chapter 4 I describe the resulting logic, modal mu-calculus, and its proof theory. A decision procedure developed by [Cleaveland, 1989] is applied to mechanical verification of the mutex property discussed earlier using a prototype analysis aid, the Concurrency Workbench (CWB). Although this is not a new result, by representing the domain in BETA and CCS I was able to recast both the modelling framework and a lengthy manual proof of mutual exclusion of monitor events developed by Lansky in her dissertation [Lansky, 1983]. This demonstrates the power and generality of my technique.

In Chapter 5 the apparatus is put to work as the basis for capturing the primary relationship. I approach the problem by developing a system development method for groupware. Domain description is based on the concepts of a role and a link. A role consists of a set of multitasking processes communicating by shared local memory. It may be performed by a person or a program. Links are channels supplied by the infrastructure that enable communication between roles. A channel can be data, video, fax or even a shared application.

Williamson's notion of a transaction is defined in terms of juxtaposed primitive transactions each of which is a BETA fragment consisting of two
communicating roles. A taxonomy of typical communication structures is developed. Transaction cost analysis characterises an organisational setting in terms of transaction networks. It catalogues breakdowns, bottlenecks, gaps and inefficiencies, and these enable the designer to envisage what kind of groupware support may be needed. The representation of the structure and behaviour of simple transactions is illustrated and I show how a simulation in the form of a hypercard stack can be formally related to the domain description.

In Chapter 6 an empirical study is reported of the application of the method to the design and specification of a wide area multimedia system. We show how the hypercard simulation provides a framework for analysis and specification. It leads to the construction of the transaction cost spreadsheet which is completed by observations by users themselves in the field. The simulation is used both to negotiate requirements for the new system and to specify its user interface.

The domain is modelled in BETA pseudo-code as a concurrent, recursive invocation of a single communicating subsystem. This finding involved a detailed transaction cost analysis across multiple communication media. It can therefore be regarded as a significant validation of the thesis that groupware domains consisting of multimedia communication structures can be successfully delineated by applying the techniques developed in this dissertation.

Chapter 7 describes how the findings in the thesis relate to previous work. The relationship of the dissertation to the fields of Management Information Systems and Computer Supported Cooperative Work is elaborated.

Chapter 8 contains conclusions and suggests directions for further work on integrated CASE tools. Contemporary systems design is characterised by a variety of graphical and pseudo-code notations which soon exhaust the possibilities of consistent and reliable manual calculation and description. My approach is no exception and has led me to use various prototype aids, such as the CWB. Concluding remarks suggest ways in which computer support might go beyond syntactic consistency checking to offer dynamic analysis with better explanatory features. This would form the basis for a
second generation scripting environment, such as Apple Hypercard or Microsoft Windows Toolbook, which together with commercial application program modules would facilitate a further shift in development capabilities towards users.

1.4. How to read the Thesis

The dissertation brings together work in several subareas of computer science. Amongst these are object oriented design, temporal formalisms for reasoning about programs, groupware development processes, information flows in human organisations, theory of coordination mechanisms etc. It is important that systems architects can understand these areas as they constitute the emerging field of open distributed processing. At present understanding is obscured as many different languages, notations and theories are involved. The thesis draws these into a unifying mathematical framework for the first time. It is able to do so because of the achievement of Milner and his colleagues in giving operational semantics to parallel languages. I have extended his work to object oriented system description, so that it serves the principled construction of graphical notations, languages and tools and the conduct of a general technique for system development.

The introductory Chapter 2 highlights requirements that an exercise in formal modelling of open systems needs to meet. The mathematical requirements are addressed in the three initial chapters, 3-5. Chapter 3 gives BETA pseudo-code its semantics. Chapter 4 shows the extensive range of properties that can be expressed in the logic, and how they can formally related to pseudo-code domain descriptions. Chapter 5 proposes a class library of components in BETA which can be combined to specify the synchronisation skeleton of distributed applications. The chapter shows how this specification can be abstracted and simulated using hypercard, and how simulation can be used as a tool in a general method of system development.

Having laid down the logical underpinnings, the mathematical argument gives way in Chapter 6 to narrative, to record how the method was used at British Airways. Here the novelty lies in showing how the correlation between breakdowns in the physical communication model, and breakdowns in the social and economic relationships in the workplace
helped users establish their requirements. While simulation is used to make such an analysis more concrete, it is the existence of new engineering platforms, the product of ongoing research, that enables us to imagine how systems will assist the solution to these problems.

The simulation is extended to show how such applications might look by incorporating emerging technologies for sound and video communication. In designing an infrastructure of workstations and communication networks the system architect needs such an indication of how usage will grow. The marginal cost of adding multimedia communication is much lower if this has been conceived *ab initio*. For this to happen systems analysis must look far wider for its requirements than it has done hitherto. Chapter 6 is a step in broadening its horizons.

The work described in the thesis opens up a line of research, and gives a snapshot of progress made so that its outlines can be distinguished. Enough has been achieved, both in mathematical modelling and in validating new research and development methods, to show that the approach has great promise. There is of course much yet to be done. The final two Chapters draw together these lessons, relate what has been learned to previous work, and discuss how the research might usefully be carried forward. Here issues that are introduced in the earlier chapters, such as graphical notations and user interface simulations, are revisited to show how they combine with the mathematical foundations in pointing the direction to a new generation of CASE tools to support architects and their clients in analysis, design and system generation.