SIMULATION MODELLING OF SPATIAL PROBLEMS

JANET SHAPIRO

A thesis submitted in partial fulfilment of the
requirements of the University of North London
for the degree of Doctor of Philosophy

January 1995

ABSTRACT

The thesis presents a simulation modelling strategy for spatial problems which uses a data structure based on spatial relationships. Using this network based approach, two domain specific data-driven models are developed in which the movement of people is modelled as a quasi-continuous process.

The development of simulation modelling technology is examined to find reasons why there should be a reluctance to use the technique. With particular reference to problems which are spatially related, the established simulation modelling techniques, together with their diagrammatic representations, are evaluated for their helpfulness at the model building stage. Using a specimen example, it is demonstrated that the commonly used approaches for digital discrete event simulation, which use a procedural paradigm, give little help with problems which involve the allocation of a resource and have spatial constraints. Two domain specific generic models are demonstrated which adopt an object-oriented approach, for which the model description, including the logical constraints, are given in the data-file.

A method for modelling the movement of people at different levels of congestion as a quasi-continuous process is validated using results from reported surveys of people's movement rates and direct observations, and this is applied in both models.

The first models the emergency evacuation of a building, using a graph structure to represent the spatial components. This is implemented using object-oriented code and test runs are compared with evacuation times from a building at the University of North London.

The second provides an experimental tool for comparing the effect upon ward function of different layouts and was influenced by a published survey of a nurse activity analysis carried out in fourteen different wards. The nurse activity model uses two graph structures and an object class to model the nurses who move, with reference to, and informed by, the spatial graph structure.

The successful application of the method in the two problem domains confirms its potential usefulness for spatial problems.

List of Contents

page

Chapter 9
The Generic Model Applied to Other Spatial Problems

Appendices

Appendix A

Floor layouts for the Eden Grove building

Appendix B

B1 Documentation for the Evacuation model

B2 Documentation for the general modelling strategy

Appendix C

Extract from 'Planning For People'

The pedestrian simulation model, PEDROUTE

Appendix D

Consultancy/Examination Rooms

Ward plans

Ward plans with nurse trails

List of Tables

List of Diagrams

List of Diagrams continued

Acknowledgements

I would like to thank Gerry Weston, London Regional Transport, Operational Research Department, for the original inspiration, and Ray Paul and David Balmer at the London School of Economics and Political Sciences, who further helped me to develop the idea for the work.

I am particularly grateful to my supervisors, Dave Saunders and Warwick Comley, for their encouragement and support.

But finally, I must thank Ian, who helped with document checking, and the rest of my family, without whose help and forbearance, the production of this thesis would not have been possible.

Author's Declaration

The work presented has been carried out by the author alone. Any section of it which refers to the work of others, has been clearly attributed.

Chapter 1. Introduction

1.1 The need for Simulation

Model building sets out to replace a complex confusing real world situation by a simpler more transparent model, which retains the desired structural essentials, and which can be analysed and manipulated to increase understanding of the complex system which it represents.

Simulation modelling is an experimental modelling technique. The documented use of Monte Carlo Methods dates from 1944-1948, developed by physicists: Von Neumann, Ulam, Harris and Herman Kahn [Curtiss et al 1951], although Statisticians had used Monte Carlo methods much earlier[Hall 1873]. Digital discrete-event simulation was introduced in the 1960's by Keith Douglas Tocher and was part of a whole range of systems modelling methodologies which developed in different disciplines and with differing interpretations[Tocher 1963]. It remains a major distinct modelling technique.

The popularity of the technique is cited by Paul [Paul 1991] to be evident in surveys of practitioners [Beasley & Whitchurch 1984] and in the literature of research centres throughout the world, but he points out that, "*The potential for simulation is undoubtedly still largely untapped.*"

This view is endorsed by a Study Report produced in consultation with the Department of Trade and Industry [Horrocks 1991] which sets out to examine the state of acceptance of the technique in UK Small Manufacturing. Enterprises (SMEs) This report concludes that the technique is widely used, but not exploited to its full potential.

"The work of the Simulation Study Group has exposed the low level of awareness and use of simulation in UK manufacturing SMEs."

Hollocks has drawn the attention of Operational Research Society members to this report [Hollocks 1992], and a recent survey of small businesses also reveals a low usage of simulation along with other Operational Research techniques [Edge & Klein].

The appendix to the Simulation Study Group report gives summary data from a study undertaken between February and July 1991 on the awareness, use and perceived benefits of simulation, gathered from over 500 interviews at manufacturing sites, professional organisations, academic institutions and software vendors. The report outlines reasons for the low level of awareness and use of simulation in SMEs, and puts forward proposals for promoting the technique. Although the study focused on the use of simulation in SMEs, it indicated a general lack of awareness, which would have implications for other areas.

A crucial impediment to increasing the use of simulation, noted in the report, is the difficulty in making modelling skills more readily accessible.

Paul makes a positive recommendation, "....*to automate as much as possible the simulation process, thereby enabling the analyst to spend more time on the problem-solving aspect of the modelling task.*" [Paul 1991]

In other words the technique should be subordinate to, but facilitate overall understanding and design. The thesis addresses the problems of model-building as experienced by the practitioner, in the context of the continuing search for a common modelling methodology for simulation. This co-ordinated approach affirms Checkland's premiss that:

"In any subject concerned with rational intervention in human affairs, theory must lead to practice; but practice is the source of theory: neither theory nor practice is prime." [Checkland 1985]

1.2 The contribution of the thesis

The thesis takes a selection of problems which have been observed to cause difficulty at the model building stage. These involve spatial layouts and allocation of resources, and have been encountered in analogous forms in a whole range of contexts:- hospital clinics, machine shops, petrol stations and underground stations. The difficulties occur in constructing a model with the correct logic and in verifying that it works. Related problems concern quasi-continuous flow in a spatial layout.

The proposed method adopts a conceptual model which is based on the physical layout, and the model definition avoids the usual difficulties with logic specification and verification. Implementation makes use of Object-Oriented Programming (OOP) which facilitates easy generation of code for similar problems.

Generic data-driven models are created for specific problem domains; an evacuation model is demonstrated which models quasi-continuous flow, and complex interactions are modelled for nurse activity on a hospital ward.

The techniques used are appropriate for modelling spatially related problems and are intended to make the simulation technique generally more accessible.

1.3 Summary of the chapters

- Chapter 2 introduces the general methods of simulation modelling.

- Chapter 3 examines three major discrete event simulation methodologies, applying them to a specimen problem as a test case in order to compare their performance.

- Chapter 4 examines the special features of spatial problems and reviews the modelling treatment used for such problems. A new approach produces a data-driven generic model which uses a graph structure to represent the spatial network.

- Chapter 5 applies the new approach to the development of a model for emergency evacuation of a building.

- Chapter 6 makes a detailed analysis of how the movement of people can be modelled within a discrete event simulation, and summarises reports on emergency egress, in order to provide parameters for the evacuation model.

- Chapter 7 reports the trial and validation of the evacuation model on a building at the University of North London.

- Chapter 8 examines the relationship between activity patterns and accommodation layouts in hospitals. The requirements for a simulation model of nurse activity on a hospital ward are outlined, and test runs of the model evaluated.

- Chapter 9 examines the application of the general modelling strategy, which is designed for spatial layout problems, to a range of problems introduced in the thesis.

- Chapter 10 evaluates the development of these models in the context of object-oriented model design and software reuse.

Chapter 2 Simulation Modelling

2.1 The general use of model-building

2.1.1 General Systems Thinking

Digital Simulation can be placed within 'Systems Thinking' which has been evolving since the 1950s as a result of both, the inadequacy of reductionist thinking to cope with the levels of complexity found in biological science, and the recognition of the role of systems by engineers [Bertalanffy 1968, Weiner 1961].

There was an attempt to unify the work in a General Systems Theory in 1955 by Bertalanffy, Boulding, Gerard and Rapoport, but more progress has been made in the separate fields of: Communication Engineering, Cybernetics, Artificial intelligence, Systems Analysis and 'Soft' Systems Methodology.

Checkland claims that, *"...systems thinking is founded upon two pairs of ideas, those of **emergence and hierarchy, and communication and control**...."* and believes that *"systems thinking and analytical thinking will come to be thought of as the twin components of scientific thinking."*
[Checkland 1985]

The technique of digital simulation is applied in accordance with the 'hard' systems thinking of the 1950s and 1960s. The characteristics of 'hard' systems thinking are:

Oriented to goal seeking,
Assumes the world contains systems which can be 'engineered',
Assumes systems models to be models of the world (ontologies),
Talks the language of 'problems and 'solutions',

[Checkland 1985]

These characteristics can be identified in digital simulation modelling - case studies using digital simulation are invariably based on well-defined systems which are bounded in space and time [Danielson, Eldridge & Brown 1991, Ceric & Hlupic 1993].

The need for less well-defined skills in simulation modelling is also recognised, as indicated by the inclusion of the word '*art*' in the titles of texts on the subject [Shannon R 1975, Tocher 1963].

Shannon justifies his title:
"The process by which a systems engineer or management scientist derives a model of a system he is studying can best be described as an intuitive art."

It is also true that there is an increasing use of simulation in environments making use of multiple methodologies [Williams, Gittins & Burke 1989, Pinkowski 1989, Williams 1992], many of which are heuristic and exploratory, having features associated with 'soft' systems thinking. Artificial Intelligence, for instance in the study of Neural Nets, makes use of simulation as an integral part of its methodology. The concept of simulation modelling is an essential component of Artificial Intelligence [Futo &Gergely 1990, Hossain 1991, Al-Amin & Tobias 1991].

Thus digital simulation plays an important role within both 'hard' and 'soft' systems thinking.

Various modelling strategies are in common use and these are evaluated in Chapter 3, but it is first necessary to examine models in general: the functions of models and the various forms they take, and the sort of world systems which might require modelling.

2.1.2 A classification of models

Model-building or modelling is a goal-oriented activity, complementary to *cognitive or design activities* [Futo & Gergely 1990].

- The **cognitive** role of model-building is to improve understanding of the functioning of the system modelled.

- The **design** element enables creativity, where something new is added to the system being modelled.

Futo and Gergely classify models at a number of levels, according to:

functions; descriptive, explanatory, illustrative, validating, predictive or normative,
medium; physical or ideal,
investigation; theoretical, experimental or mixed,
level; micro or macro,
and **character**; static or dynamic.
Any model may have one or more of these properties. **Ideal** models are further classified as having an iconic or symbolic representation mode, and symbolic models can be further broken down to include mathematical or computer models. **Dynamic** models can be organisational or have a time consideration, implicit or explicit, with discrete, continuous or combined time representation [Futo & Gergely 1990].

Diagrams 1 and 2 illustrate this taxonomy.

Diagram 1 The model medium

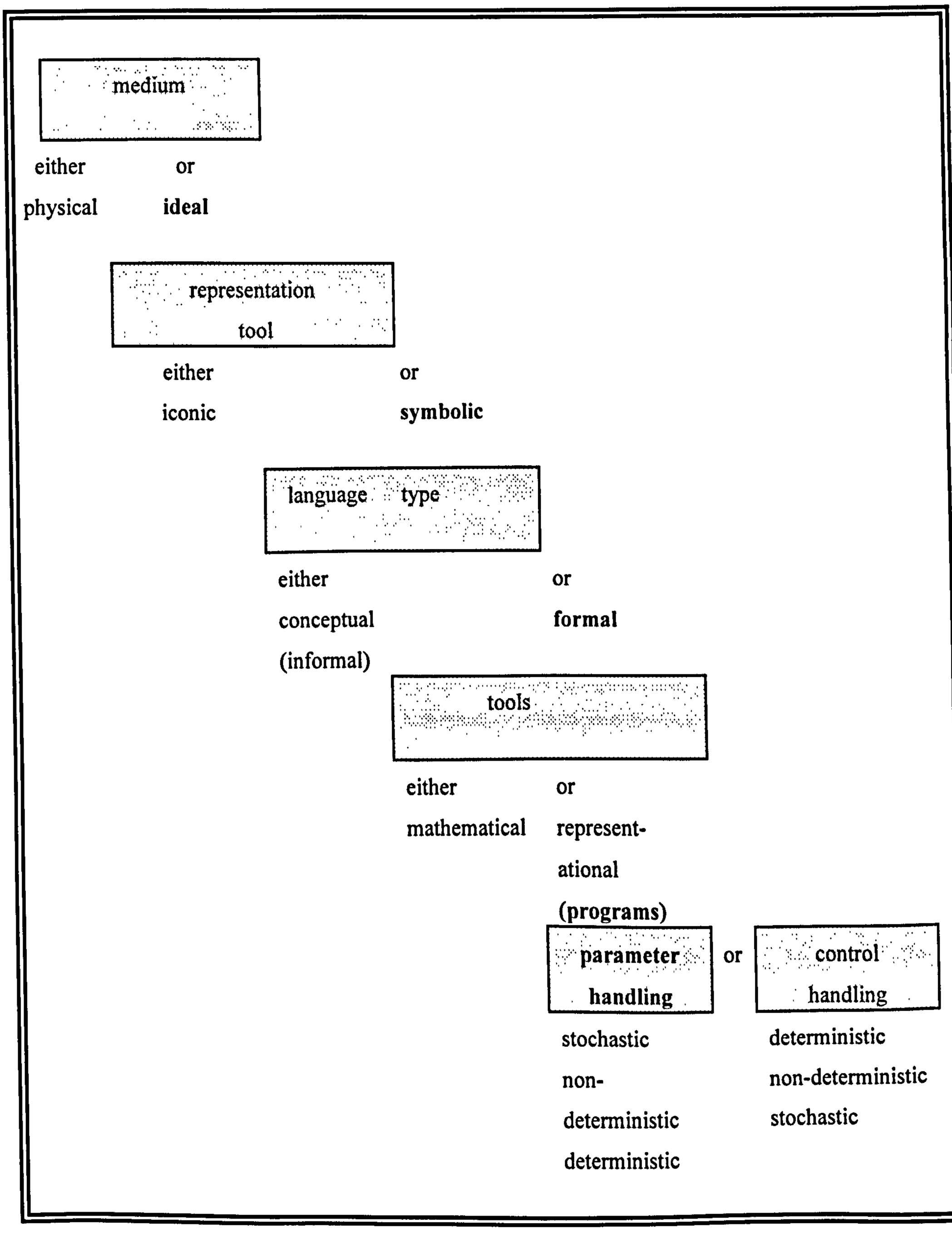
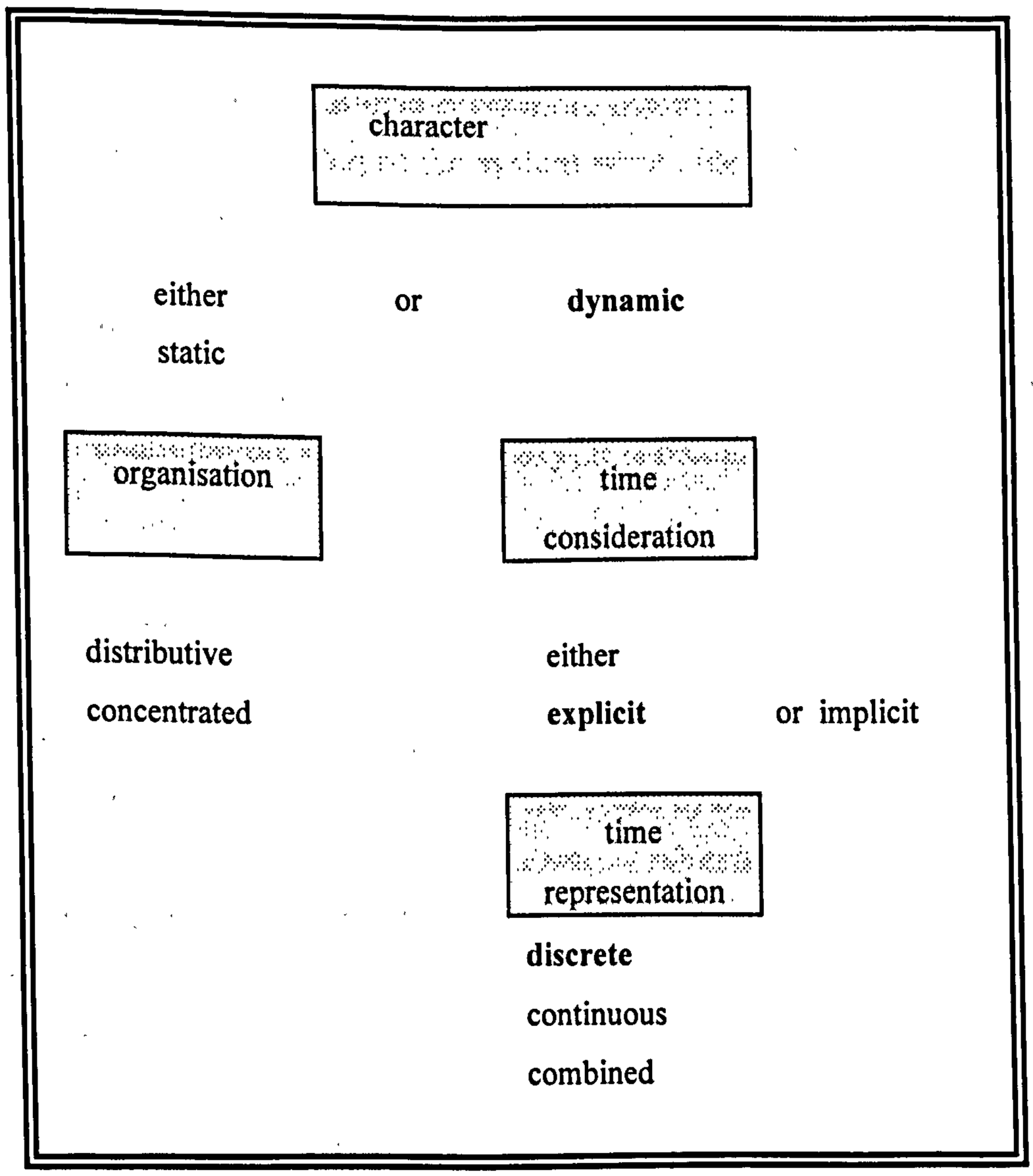


Diagram 2 The model character



The properties of digital discrete-event simulation can be identified in this taxonomy:

- A digital discrete-event model may serve any of the **functions** listed at some stage.
- It has an **ideal** medium, a **symbolic** representation tool, a **formal** language type and a **parameter handling** representational tool (program), which may be **stochastic, deterministic** or **non-deterministic.**
- The investigation will be **experimental,** but hybrid simulation may be used which links with theory [Dubois 1979].
- The level is usually **micro,** but models can be macro.
- The model will have a **dynamic** character, with an **explicit discrete** time mechanism.

The taxonomy provides the common elements for the modelling process adopted in Artificial Intelligence, Software Engineering and simulation modelling. An integrated approach is possible to Systems Modelling using this common terminology [Fishwick 1992].

2.1.3 Types of world system.

An enumeration of application domains is not appropriate for a taxonomy of frequently modelled systems, but a general grouping of problems types, such as: queueing , distribution, scheduling and allocation is more useful. These problem areas are associated with the functioning of the system which identifies both the characteristics of the major components and the objectives of the analysis.

To classify world systems by problem type is not inconsistent with a more theoretical approach, in which the term **system** denotes the portion of the world selected by the modeller for examination. In **system theory,** it is recognised that the system is itself a construct.

2.1.3.1 Definitions and terminology

Futo gives a definition:

*"From the point of view of cognition a **system,** is simply a model of a part of the world under investigation cut out according to the goals of the observer, which consists of a relatively organised and stable **conglomerate of objects** considered as a whole with respect to the remaining part of the world which is considered as the environment. Note that stability is a requirement for the **reproducibility** of the investigation."*

[Futo & Gergely 1990]

Note that the term **entity** will be substituted for **object** in this text, since object has a special reserved meaning in object-oriented programming.

Futo introduces a two way classification: static versus dynamic and micro-level versus macro-level, which is a useful base for model design, earlier recommended by Oren and Zeigler [Oren & Zeigler 1979]. At this stage the system is viewed at the micro-level, firstly at the static level. With the introduction of a time consideration the system becomes dynamic.

At the **static** level the general components of a system are the entities(objects), which have attributes and hidden internal structure. The system can be viewed as a structure of communication channels with source and target points, and the entities have defined mutual relationships.

At the **dynamic** level, activities may be performed, communication channels may be used, and changes to the state of the system i.e. events may occur. In the context of queueing systems and other problem areas, activities take the form of 'service'.

The conceptualisation of the system [Futo 1985], demands *stability* for *reproducibility*, and there will be a defined 'goal'. Apart from that, the nature of the world system to be modelled varies widely.

2.1.3.2 A classification of systems

With reference to this general framework and to the problem areas, it is possible to categorise systems according to:

- the number of entities,
- the life of the entity,
- the service mechanism,
- whether physical space is a constraint,

This list of features is by no means exhaustive for distinguishing different types of system, but each feature is known to affect the modelling strategy.

Number of the entities flowing through the system.

There may be a comparatively small number of entities, such as customers or components. For the former waiting time may be critical, for the latter, availability,

or

there may be large numbers of entities, giving a quasi-continuous system, and entities may be modelled as aggregate numbers only.

Life of the entities

Entities may have a transient life, a birth-death process. For instance customers arrive and leave, ***or***

the major entities may remain permanently in the system, such as machines in a machine-interference system, or vehicles in a shuttle service. Entities move through a closed cycle.

Service mechanism.

Processes or activities may engage entities individually ***or*** in batches. In the petrol station, cars are served individually, but in a traffic flow situation, cars move forward in groups.

Spatial constraints

There may be ample space ***or*** a scarcity of space. Location of the entities may determine which processes may or may not be carried out, and the time taken to move from one location to another may or may not be negligible.

The features can be summarised as follows:

• Number of entities	Small	or Large/infinite
• Life of entities	Transient	or Permanent
• Service mechanism	Individual	or Batch
• Spatial constraint	Ample space	or Scarcity of space

The classic simple multi-server queue, corresponds to the first column of values:

Small/Transient/Individual/Ample).

Whereas a traffic flow system is given by:

Large/Transient/Batch/Scarce,

and a machine interference system is given by:

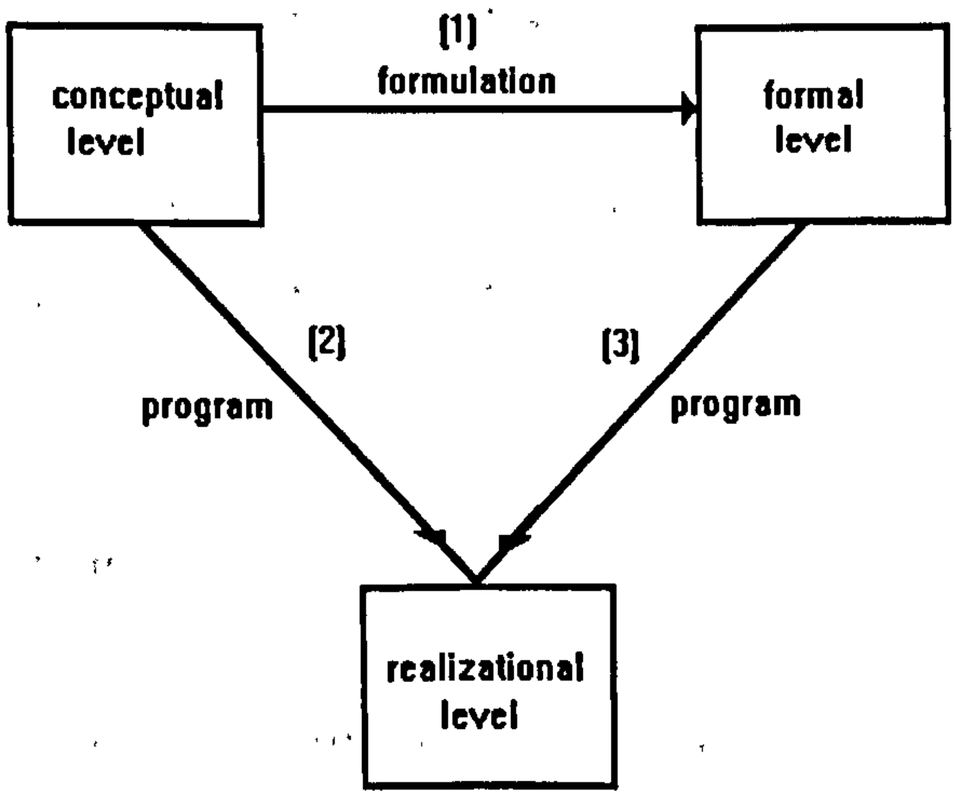
Small/Permanent/Individual/Scarce.

The thesis focuses upon the modelling strategies required to model systems of the latter two types of system. The special features of spatial problems are examined in Chapter 4.

2.1.4 The general process of modelling

The process of modelling follows three stages. Given a particular problem, *the goal*, a portion of the world is selected for investigation, and a conceptual model is constructed. This is formulated as a formal model which can be used for the problem solution. The formal model may be either an abstract theoretical model or a computer simulation model.

Diagram 3 Levels of Modelling



Process (1) denotes the formulation of the conceptual model as a theoretical model.

Process (2) denotes the implementation of the conceptual model as a computer program which can be used for experiments.

The theoretical model may be programmed, Process (3).

The formal model itself is well-defined and unambiguous, but the intuitive process of model formulation is heuristic; it progresses by trial and error, using simplification, analogy, enrichment subdivision and synthesis,[Shannon 1975].

Confusion can arise because of the different status of the various models, particularly in soft system methodology (SSM) [Lewis 1992]. Also in simulation, the model may be cognitive; intended to aid understanding of the system as it exists without intervention, in which case only one model is constructed, but it is more likely that the simulation model is needed for design purposes, and a multiplicity of models will be required for the necessary experiments.

2.2 Digital Simulation

2.2.1 General definition of digital simulation

There is a consensus of agreement for the general definition of digital simulation.

The following definition is quoted from a paper on Petri net graphs by Aimo A Torn, *"Simulation may be regarded as a general purpose method of modelling; i.e., in the absence of applicable mathematical formulas or equations, the working of important parts of the system under consideration may be mimicked, and the values of the entities of the system may be obtained by collecting data during a simulation run. Because simulation models often are complicated and because their use requires extensive calculations, they are often realised as computer programs."* [Torn 1981]

This definition identifies several aspects of the technique which will be expanded upon:

- the generality of the method,
- its experimental nature,
- the assumption of an abstract model,
- the need to use a computer program.

2.2.2 The computer program as the model

The need to produce a computer program to carry out the simulation has strongly influenced the conceptualisation process. The computer program is perceived as being the simulation model, and imposes its own modelling constraints. For this reason, several modelling methodologies have developed fairly independently, and, since the simulation modelling requires time and effort to master, practitioners have tended to refine their skills in one methodology only.

The conceptual problems of changing a familiar paradigm should not be underestimated and practitioners find it difficult to acknowledge or try out other modelling methodologies [Prince 1994]. The effect has been a divided professional base, and the need to acquire sophisticated modelling skills have discouraged potential users from what should be an essentially simple technique. This is not to say that the need to make simulation software more accessible has not been ignored. Note the development of program generators and help systems [Clementson 1982, Mathewson 1984, Crookes 1987, Paul & Chew 1987].

2.2.3 The abstract model

Digital simulation was developed by Operational Research workers with an engineering or mathematical background, who were familiar with the use of an abstract model. In the context of digital simulation, the hypothetical model is likely to take the form of a simple procedural model, univariate, as opposed to multivariate, which is more frequently used for economic or sociological systems.

The techniques of mathematical modelling and queueing theory have also been improved during this period, and Operational Research workers will regard these as a valid and inexpensive alternative to simulation for simple systems, when an adequate analytical solution can be obtained using less data, and less computation. It is recognised that theoretical analysis may also give approximate aggregate results for some complex systems.

Abstract models will be common currency as technical model representation methods for the control of complex processes. They may be iconic or symbolic, but become familiar as essential means of communication in applications domains such as: inventory control, manufacturing processing, distribution, job scheduling, traffic control etc. Several views may be taken of the system resulting in the need for several types of diagrammatic representation [Danielson, Eldridge & Brown 1991]. For the practitioner using such models, expertise linked with responsibility for good performance ensure that the methods used for diagrammatic representation accurately convey complexity and logic. The model representation is abstract but closely related to the problem on hand. In the practical context, a familiar abstract model is a useful communication aid; it is merely a tool.

2.3 Digital discrete event simulation

This thesis focuses upon digital discrete-event simulation, as opposed to analogue and continuous simulation.

2.3.1 The modelling of continuous processes.

Analogue simulation represents system changes using physical measures, usually electronic, based upon a mathematical continuous model, which is invariably formulated using differential equations. Thus continuous simulation may be carried out on an analogue computer, but can also be programmed on a digital computer, in a similar manner to numerical analysis.

Special purpose simulation languages provide for continuous simulation. GASP [Pritsker 1974], and SLAM II [Cochran & Lin 1989] combine the modelling of continuous processes and discrete events. ECSL provides a *dynamics* block in which continuous activities are updated at each time step, and WITNESS models continuous physical elements: *fluids, tanks, processors* and *pipes*. which parallel the discrete physical elements: *parts, buffers, machines* and *conveyors*.

Systems Dynamics models the variables of a system as continuous functions over time, using discrete time steps of appropriate length. The internal relationship between the variables is of interest [Richardson & Pugh 1981], but the technique is not discrete-event.

2.3.2 The modelling of discrete events

In digital discrete-event simulation an artificial time-scale is imposed upon the system, whereby time advances in finite steps. Events or instantaneous changes to the system are modelled to occur sequentially at these time-beats, the size of the step being dictated by the problem. The computational algorithm used for discrete-event simulation may advance time in fixed steps or from event to event. The method chosen for updating time will affect efficiency and possibly accuracy, but the underlying conceptual model remains the same.

2.4 Simulation in practice

2.4.1 General purpose software

Because the simulation technique is general purpose, and in theory, any system can be modelled, most simulation software is produced claiming to provide for general requirements.

A few claims are quoted:

Models in which discrete events and continuous change both occur can now be programmed with ease in ECSL.
[ECSL 1982]

STEM (Simulated Time Event Mechanism) is an object-oriented development tool for building working models of virtually any process.....,
[Lewis Artificial Intelligence Limited 1988]

The real challenge in using Pascal is to make an implementation that is....powerful in the sense that anything reasonable can be modelled.....,
[Shearn (PASSIM) 1990]

vs7 is... a flexible tool to model the operation of a wide variety of systems....,
[Syspack 1990]

see why ... can be applied to any dynamic system, eg: an oil terminal, an airport, an assembly plant, a lift system, an automated store.
[see why brochure, BL Systems Ltd.]

The software is marketed on the basis that it will be useful for any problem, but the expert in simulation modelling is well aware that contrasting systems are frequently analogous, and experience gained in modelling one situation can be transferred to another. For the expert, the abstract structure of the model will link one application with another [Maiden & Sutcliffe 1992].

2.4.2 The client's perception of the simulation model

Diagrammatic representations have been found useful, both for explaining model structure and as an aid team discussion of a problem. Tocher introduced *wheel charts* which were the precursor of the activity cycle diagram [Tocher 1964], and consultants using HOCUS used large scale charts, with icon characters to represent entities [Poole & Szymankiewicz 1977].

As early as 1960, Tocher introduced Production Gaming to communicate the idea of simulation modelling to managers [Rivett 1983], and a games oriented style is often adopted by in software design to convey the experimental nature of the technique. Unfortunately the games oriented approach, adopted without expert advice, may also have the effect of trivialising what is a serious and difficult task, and faulty conclusions can be drawn without rigorous analysis.

2.4.3 Systems which help the user

The generality of the software implies abstraction, and the novice needs either an interface which presents a familiar view of the problem, or one which gives instruction. For this reason various types of program generators and expert system interfaces have been produced to help the user. Several of these are well documented [Crookes 1982, Clementson 1982, Szymankiewicz 1977, Mathewson 1974 & 1984, Paul & Chew 1987], and simulation software is frequently marketed on the basis that it will enable the client to build his own model:

This interactive program(CAPS) is not capable of accepting every detail of every simulation model. However, it should be capable of building at least the skeleton of all simulation programs .....,
[ECSL 1982]

The client can build the model himself with little training, test the model until confident of its results and then perform the required experimentation.
[Steele (See-Why Istel Ltd.) 1984]

The benefits of the WITNESS approach are that:.....
....Models can be built and tested in small incremental stages, which greatly simplifies model building, provides the ability to identify errors in the logic and makes the model more reliable.....
[WITNESS, AT&T ISTEL Ltd. 1993]

2.4.4 Generic simulators

Generic simulators, which model a limited range of systems, are easier for practitioners to use, since part of the modelling process is already complete and only the data for the problem needs to be supplied. Pidd cites a range of such data-driven generic simulators, and introduces one developed by himself, SKIM, which models quasi-continuous manufacturing systems. The software was intended to allow design engineers to use simulation modelling without the aid of a simulation expert, but although SKIM was helpful, the engineers still found difficulty with the modelling process. [Pidd 1992a & 1992b]

In appreciation of the difficulties of the complete process, modelling experimentation and interpretation, some firms, such as PE Consultants (HOCUS), provide a combined software and consultancy service.

2.4.5 Interactive models with graphics

Simulation models were originally programmed on mainframe computers, but now most software is available on PCs with the possibility of screen graphics.

The use of PCs allowed animated display of a simulation run, together with user interaction, **Visual Interactive Simulation,** VIS. Bell reports successful use of Visual Interactive Modelling (VIM), for spatially related problems such as: "*routeing, transportation planning and facilities location"*[Bell 1985], and there are many other reports of successful use of VIS, [Hurrion 1978a 1978b, Withers & Hurrion 1982, Hollocks 1983, Mathewson 1984, Bell 1986& 1989, Danielson, Eldridge & Brown 1991, Hoare & Willis 1992].

VIS has facilitated easy model verification, and experimentation, but the effects on model construction are not so marked [Henz Luehrmann & Donald 1989]. VS7 and WITNESS provide graphics program generators but there is no strong evidence that the facility makes model building easier for the uninitiated user. Pidd provided a spreadsheet style of input for SKIM but commented that the engineers *"would have much preferred a graphical user interface(GUI) to be used"* [Pidd 1992]. This was unfortunately not tried.

2.4.6 The experimental approach in decision support software.

Computer Aided Design (CAD) is a more recently developed technique than simulation, but the Simulation Study Group report comments that it has more general acceptance in the manufacturing industry [Horrocks 1971]. The concepts used in CAD are very similar to those of simulation, but the user is in more direct control and the problem is expressed graphically.

Spreadsheet 'what if' modelling is used extensively for a wide range of applications, and like simulation, entails using an experimental technique. General software of this type can be used for simulation modelling of simple queueing problems [Przasnyski 1989/1990, Freeman 1993]. Recent software developments are expected to make spreadsheet modelling easier and more attractive to the non-programmer.

2.5 Developments in modelling strategy

Over the last thirty years simulation software and modelling strategy have developed together, and the assumed paradigm has been procedural. Recent developments have used an object-oriented approach which is appropriate for the hierarchy of models needed for modelling at the macro-level, and which is generally useful for complex structures.

Futo comments that, "*Models are very important in knowledge representation. Knowledge can be structured by establishing a connection among the constituents. Such an approach is supported by object-oriented programming.*" [Futo & Gergely 1990]

Eldredge explains that, "*In the object-oriented paradigm, the major entities in the problem domain are identified and modeled as the starting point in system development. the behaviour of objects is the focus instead of the sequence of actions that must be performed.*"[Eldredge, McGregor & Summers 1990]

The language C++ is an appropriate language to develop object-oriented simulation [McGregor & Sykes 1992]. Pascal 6 has some object oriented features [Borland, Turbo Vision Guide 1990], and some special purpose simulation languages are object-oriented.

Chapter 3 Discrete Event Simulation

3.1 Modelling strategies for digital simulation

The general process of simulation modelling is described in section 2.1.4 as heuristic. Attempts have been made to establish general but well-defined guidelines for this untidy process.

3.1.1 Unifying the strands of development

Oren and Zeigler developed a GPSS-based **comprehensive system** to structure the large number of model versions and provide assistance in model building and experimentation [Oren & Zeigler 1979].

Overstreet and Nance adopted a theoretical approach and defined a **Model Specification Formalism** which would be universal, but unambiguous. Their model specification included:

(input specifications,
output specification,
object definition set,
indexing attribute,(time)
transition specification),

where the transition specification, which defines an initial state, termination conditions and the dynamic structure, is not language independent. This specification gave a precise comprehensive model description intended *"...to introduce an intermediate form between a conceptual model ...and an implementation...."* [Oren & Zeigler 1979]

Further commonalty for discussion is provided by Fishwick, who developed a simulation modelling taxonomy that is compatible with systems modelling in Software Engineering and Artificial Intelligence, using the declarative/functional dichotomy which already exists in each discipline [Fishwick 1992].

3.1.2 Diagrams for model representation

A **diagrammatic representation** of the model would be a more effective intermediate form, and in pursuit of a common modelling strategy, a series of papers by Paul & Ceric review the most commonly used approaches for model representation. If one approach were better than others, this one would be worth developing, and might be adopted as a common approach [Ceric & Paul 1989, Paul & Ceric 1990].

The authors set out to evaluate the comparative usefulness of each method:

- does it facilitate model construction?
- does it convey model structure clearly for verification?
- does it enable further development work, and make the programming easy?

They concluded that a single simulation model representation method could not fulfil all functions, and that, *"..a hierarchy of diagrams that have a parallel algebraic equivalent will enable model comprehension at one level, and model completeness at another."*

The three stages of the hierarchy would be: conceptual model building, model logic and software implementation.

This conclusion agrees with the reported decisions on the Euromethod programme, which was comparing methods of structural (data) modelling across Europe, and is now, *"..moving away from the idea of a single 'best' method towards a process reference model- a generic model of the overall process that is more abstract and that has a greater breadth than existing methods. It should be 'an umbrella under which separate methods can sit' (Public Procurement Group 1991)."* [Flynn and Fragazo-Diaz 1993].

The problems encountered in data modelling are similar, since the models are conceptual and diagrams are used with specific terminology as part of the analysis and design.

3.1.3 The common requirements of a modelling strategy

Paul and Ceric identify the following common features of simulation modelling which give rise to conceptual problems [Ceric & Paul 1992]:

- change of system state at discrete instants of time,
- parallel asynchronous events,
- common waiting lines,
- system operation control.

The system to be modelled will be dynamic and require the use of a time-clock and data-handling techniques capable of keeping inter-related records. In other words the model will be complex.

Paul and Ceric specify the tasks to be tackled at the model formulation stage as:

- extraction of the most relevant system elements and interactions,
- making simplifications and approximations,
- system decomposition.

3.1.4 Modelling strategies in common use

Three major modelling strategies emerge as being distinct, with defined terminology and graphical representations. These are identified with simulation programming languages by Aimo Torn as follows:

- event-oriented (GASP, SIMSCRIPT),
- activity-oriented (CSL),
- process-oriented (GPSS, SIMULATION which is SIMULA-based).

[Torn 1981]

Most third generation special purpose simulation languages can be identified as following one or other of these modelling strategies.

Each modelling strategy makes use of a diagrammatic description. This may be 'neutral' [Ceric & Paul 1992], and have a broader currency. For instance augmented petri-nets graphs and activity cycle diagrams are widely used to develop models in any programming language.

The graphical terminology of each approach is demonstrated on a specimen problem in section 3.3.

3.2 A specimen spatial problem to compare strategies

In order to illustrate each of the three methods, and to evaluate whether each one satisfies the common requirements of a modelling strategy as defined above, a simple two-server queueing problem is proposed with two features:

- a spatial resource
- an allocation of resource,

both of which introduce modelling difficulty within a very simple system.

3.2.1 The petrol station forecourt problem

The chosen problem is the petrol station forecourt problem in one of its several possible forms. The spatial layout imposes constraints upon the movement of entities, but the rules are simple and the number of entities small. Other situations exist which present similar modelling difficulty, such as hospital clinics, which also involve allocation and a spatial resource [Beattie 1973, 1974, Shapiro 1976]. The system structure is analogous.

In the modelling of the evacuation of a building, spatial resource is intrinsic, but the situation is quasi-continuous, like the manufacturing systems modelled by Pidd [Pidd 1987], and other modelling difficulties are encountered. The evacuation model is considered later in Chapter 5

The **petrol station forecourt** problem is modelled using each modelling strategy in turn. Diagrams are produced using the corresponding model representation method.

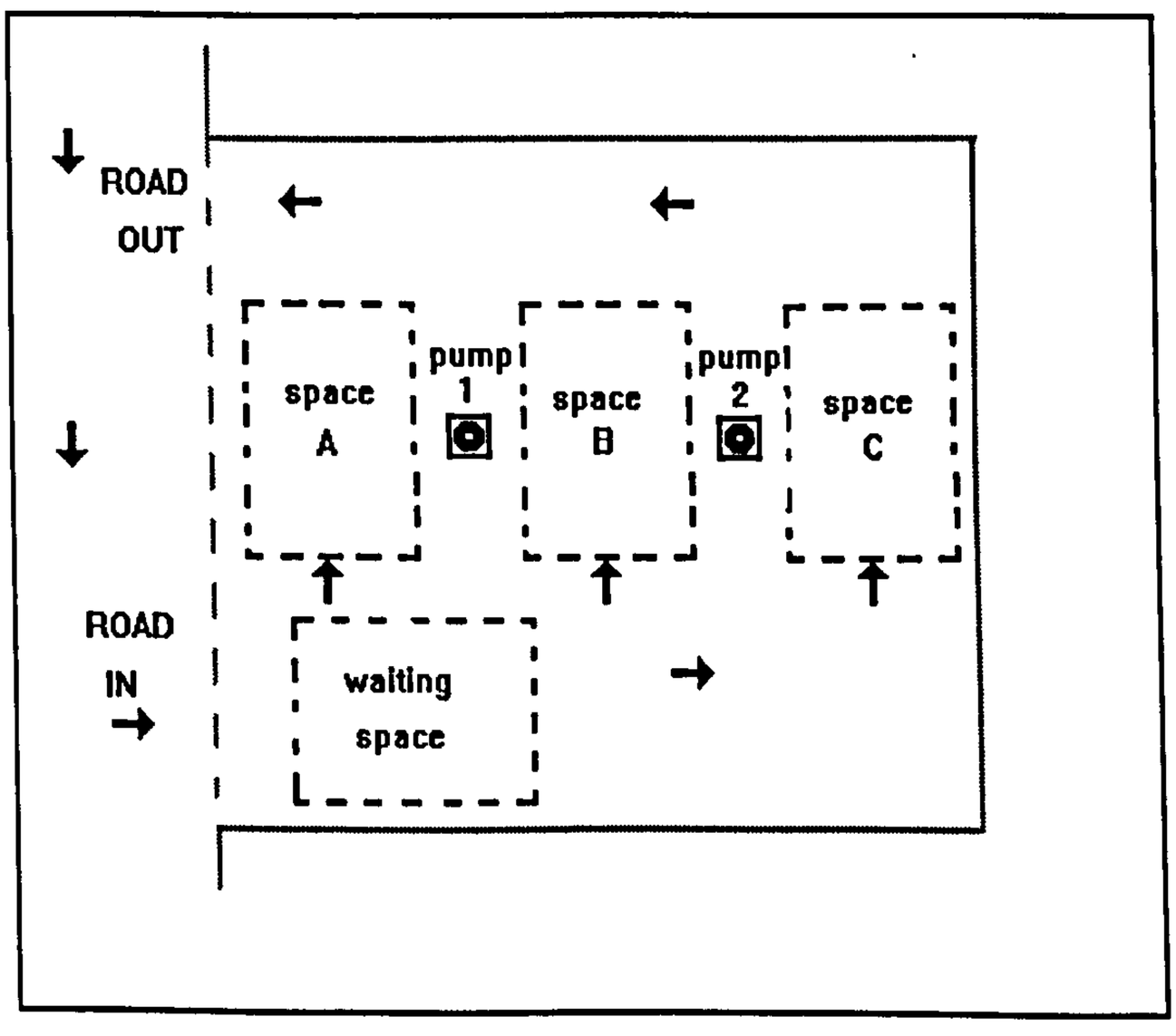
3.2.2 Problem description

The petrol station forecourt consists of a simple queueing model in which there are two servers, pump 1 and pump 2, and a stream of cars arriving to be served. There are only four standing places for cars at the forecourt; one waiting space and three spaces A, B, and C alongside the pumps. There is a restriction on the reach of the pumps: pump 1 can serve cars in spaces A and B, and pump 2 can serve cars in spaces B and C.

It is assumed that, as soon as service is finished, the pump is available to serve whichever waiting car can be reached. The space is only made available when payment is complete and the car leaves the forecourt. If there was a car in the waiting space, when a previous car had left, it could move into the free space alongside a pump. Cars arriving when the forecourt is full drive away.

The layout in shown in diagram 4.

Diagram 4 The petrol station forecourt layout



3.3 The three major modelling strategies

3.3.1. The event-based approach

In the event-based approach, the questions asked are:

- What can happen to this system that will change its state, i.e. what are the major events?

- When one of these events occur, what are the subsequent effects upon the system i.e. what is the chain of events?

An event is defined as an instantaneous change in the system. If we further distinguish between events which are 'inevitable' given the present state of the model, and events which happen only when certain conditions are satisfied, we have defined **bound events** and **conditional events**. These definitions are common to all simulation methodologies.

The **event-based** approach analyses the system in terms of the bound events which can occur, and maps all the consequences following upon each of the identified events. The bound events are conditional only upon time and can be listed on an **event calendar.** As the simulation proceeds further bound events are generated and will be entered in the event calendar.

The simulation executive will scan the event calendar and update the time-clock to the time of the soonest event. Then the event (or events), occurring at this clock time will be retrieved and the computation defined in the event flowcharts will be followed. The process is repeated until clock reaches the simulation termination time.

For the petrol forecourt problem three **bound events** can be identified:

- *car arrival,*
- *completion of the service by the pump,*
- *car leaves.*

There are other events which can be identified as **conditional**:

- *car moves into space alongside pump,*
- *car starts to be served by the pump.*

Diagrams 5, 6 and 7 show the consequences of each bound event separately, and Diagram 8 gives the control algorithm. Each adopts a convention in which dotted lines convey flow of information:

- In Diagrams 5, 6 and 7, scheduled events are written to the event calendar.
- In Diagram 8 the event calendar is searched for the soonest event to determine the next clock-time.

SCAN denotes that, at the clock-time selected, one or more events are due to happen. Control will ensure that all events which occur at that clock-time are executed.

This convention is adopted for control algorithms in the thesis, but there are alternatives such as the event graph, which is a short hand representation [Ceric & Paul 1992].

Diagram 5 Bound event CAR ARRIVES

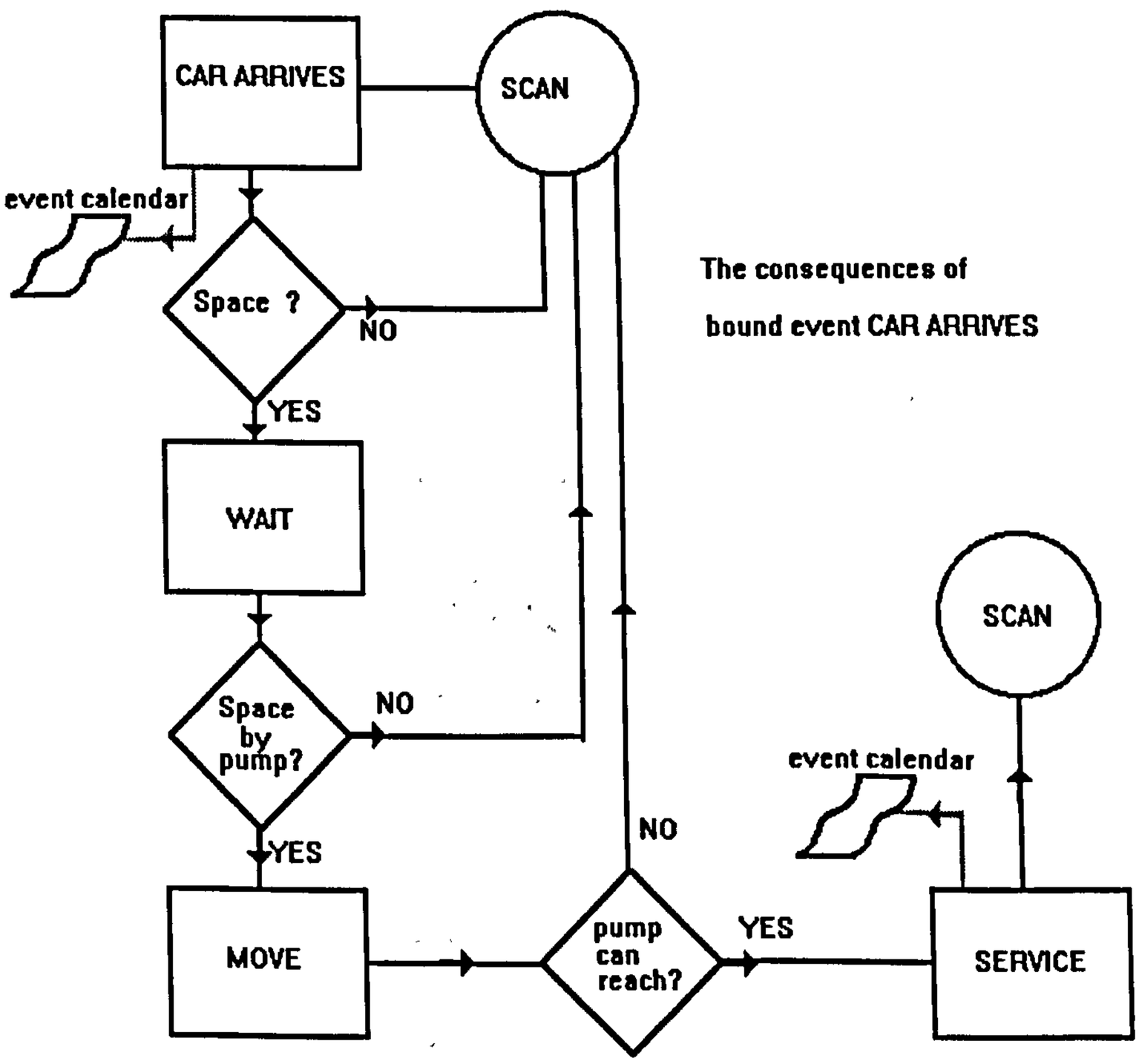


Diagram 6 Bound event PUMP COMPLETES SERVICE

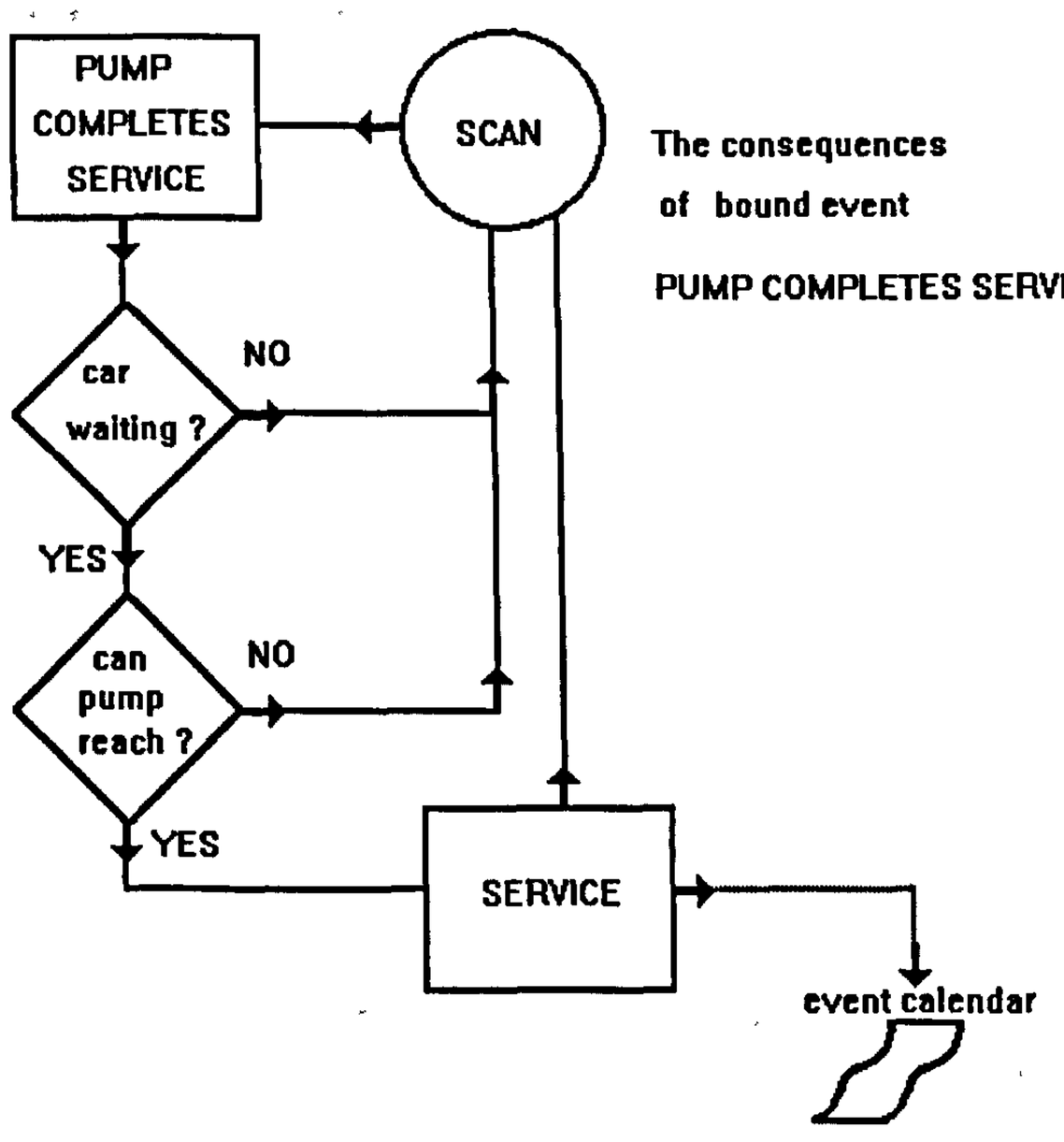
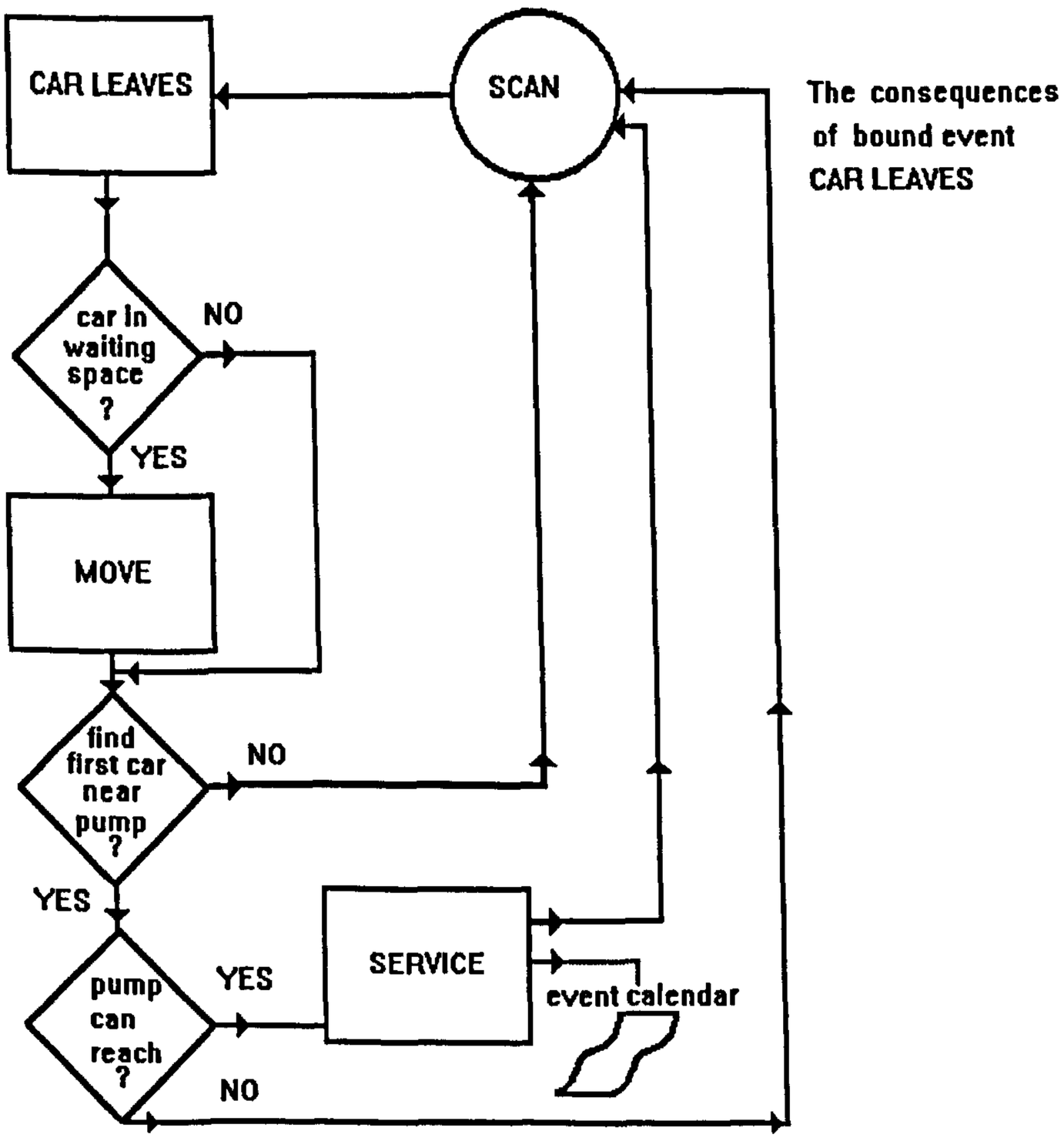
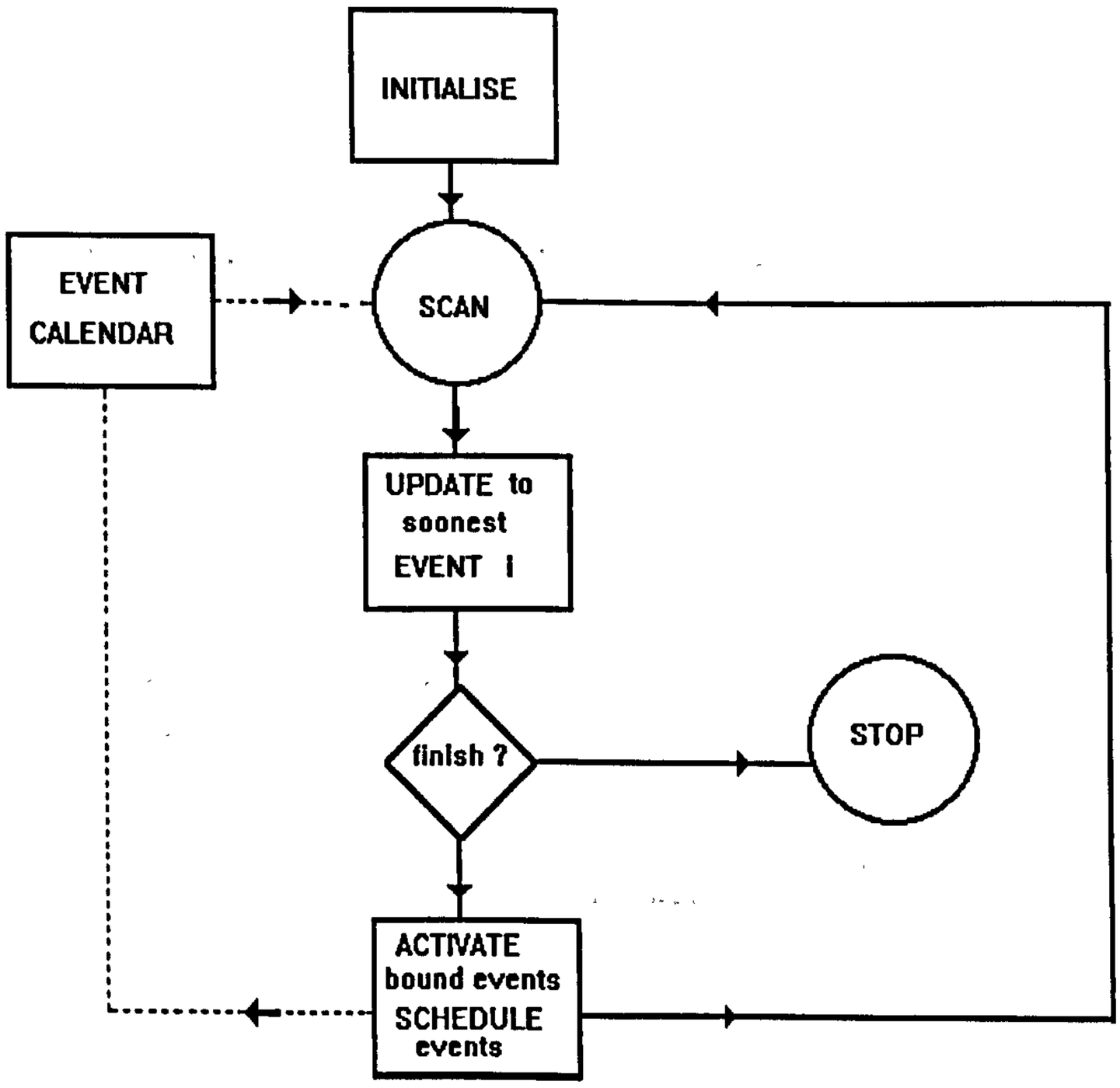


Diagram 7 Bound event CAR LEAVES



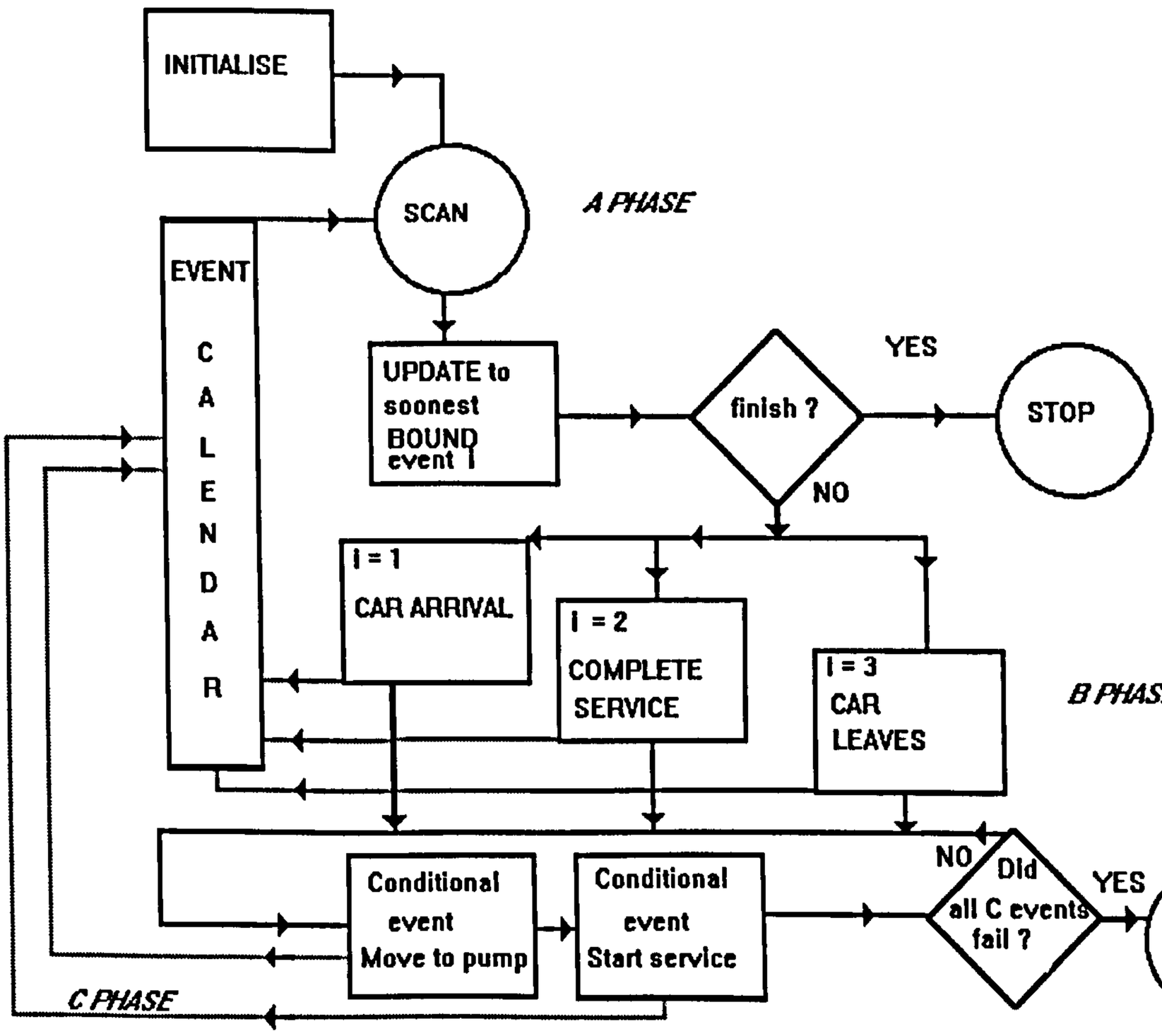
The executive will follow the algorithm shown in Diagram 8.

Diagram 8 The event-based algorithm



The algorithm can be simplified by mapping out the consequences of bound and conditional events separately, which prevents some repetition in the flow charts for the events. This produces a three-phase event algorithm shown in Diagram 9.

Diagram 9 The modified event-based algorithm



3.3.2 The process based approach

Aimo Torn [Torn 1981] gives a general description of this approach:
"In process-oriented languages all events are recognized as processes and such processes are described."

GPSS is an example of a special purpose language which uses this approach. GPSS uses a block diagram representation with symbols which are closely related to the code and to a transaction process model. GPSS is widely used and the diagrammatic tool is helpful for models of this sort.

The Petri net is widely used to model systems of asynchronous concurrent activities. In its simple form the graph consists of
"two types of nodes: circles (called places) and bars (called transitions). These nodes, places and transitions are connected by directed arcs from places to transitions and from transitions to places." [Peterson 1977].

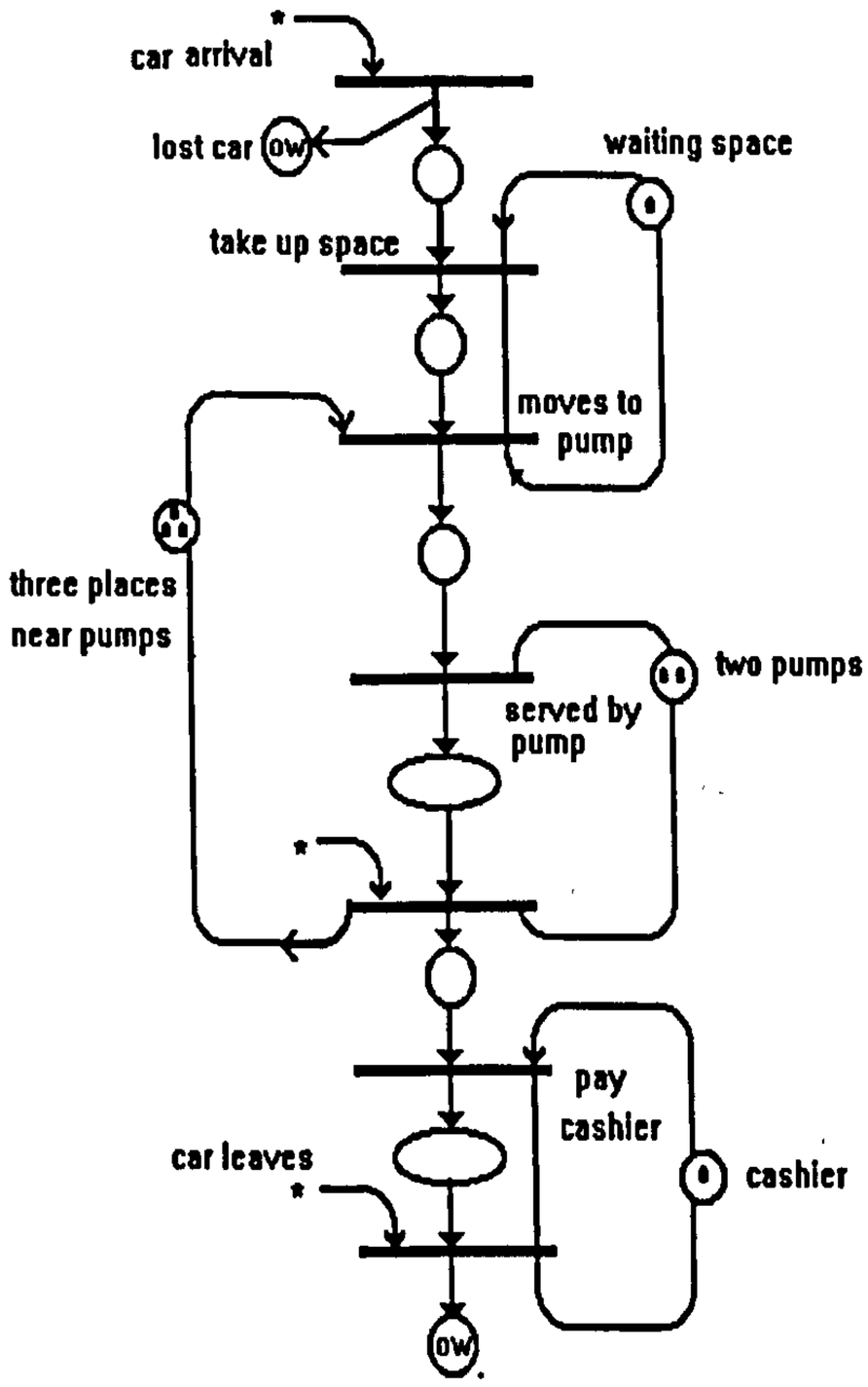
The Petri net can be used for *dry-runs* using tokens, represented by a black dot placed in a circle. Tokens show whether or not a condition is satisfied for the *firing* of a transition. In this sense dynamic Petri nets provide a general diagrammatic representation of a model which is useful for any modelling approach.

Petri net graphs have also been used extensively as an analytical tool for formal structures. Peterson quotes many references [Peterson 1977].

Aimo Torn has extended the notation further to produce *'simulation graphs'* which he claims can be universally useful for model building and verification. He introduced a **hierarchical** notation, whereby a double transition for start and finish of a service could be denoted by a double bar, which could be described in more detail in a separate graph. He also introduced **inhibitor arcs** to model priority conditions, where tokens are referenced but do not move from one place to another. Other authors have published further modification to the Petri net, for example, EPNSim graphs [Lin & Lee 1993].

The Petri net for the petrol station forecourt is given in diagram 10.

Diagram 10 The Petri net for the petrol station forecourt



For the transition *'car moves into space alongside pump'*, any space will allow the transition to fire.

For the transition *' car served with petrol'*, the pump which will allow the transition to fire is the one which matches the space.
This is not explicitly indicated on the static Petri net, but in the dynamic Petri net, the tokens for space and pump could have attributes indicated by colour or shape, which will correspond to the coding.

Alternatively the Petri net could be constructed with separate transitions for moving into each of the three spaces, and separate transitions for service by each of the two pumps. This will aid verification but coding to match the transitions would lose generality since the number of spaces and pumps could not be changed. Also, in a system with a large number of spaces and pumps, the Petri net would become impossibly complex with separately represented transitions.

3.3.3 The Activity-based approach

"In an activity-oriented language, activities (an activity is a number of closely related events) are modeled." [Aimo Torn 1981]

Aimo Torn quoted CSL, as the language which had adopted this approach, when he gave this description. CSL was developed by Esso Petroleum in 1960 with help from IBM. It was coded in FORTRAN and used bound events which were called **time dependent activities.**

In CSL, **time dependent activities** were understood to be the events starting and terminating an activity, and the CSL code was a listing of these in blocks; each block giving the conditions which needed to be satisfied for each 'activity' to happen, and the assignments and changes of state consequent upon it happening.

CSL was subsequently extended to produce ECSL, in which the term **activity** was defined to denote something which engaged entities for a period of time [Clementson 1982].

So the development of ECSL was marked by a **re-definition** of the term *activity,* and was accompanied by the use of the **Activity Cycle Diagram,** which originated form the 'wheel charts' introduced by Tocher [Tocher 1964].

In an **Activity Cycle Diagram** ACD, entities are considered to move through alternate active and passive states, in a closed cycle. For an activity to start resources may be needed, and these are released on completion of the activity. Very few notational devices are needed: a rectangle denotes an activity, a circle denotes a passive state or queue, and directed arcs indicate the flow of entities.

The ACD became a useful aid for model formulation for ECSL, CAPS (Computer Aided Programming of Simulations) and several other simulation languages including: HOCUS [Poole & Syzmankiewicz 1977], SIMON/DRAFT/DRAW/SSIM [Mathewson 1987], PASSIM [Shearn 1990]. In fact, the Activity Cycle Diagram has acquired a currency of its own and, like the Petri net, is widely used for model representation, whatever language is to be used for coding.

The system is modelled by identifying **entity classes,** and defining, for each class a sequence of **activities** and **queues.**

In this example the entities classes are *cars, waiting space, serving spaces, pumps* and *cashier*. The important entity class is *cars*, and the other four are resources.

Activities involving the *cars* are:

COMES	Arrival
MOVE	Enter if room
ENTER	Move to space alongside pump
SERVICE	Service
PAY/LEAVE	Pay cashier and leave

In this problem this is the complete list of activities. There are no other activities involving the other entity classes which do not engage *cars*.

Each activity may engage entities from more than one class, which is shown in Table 1.

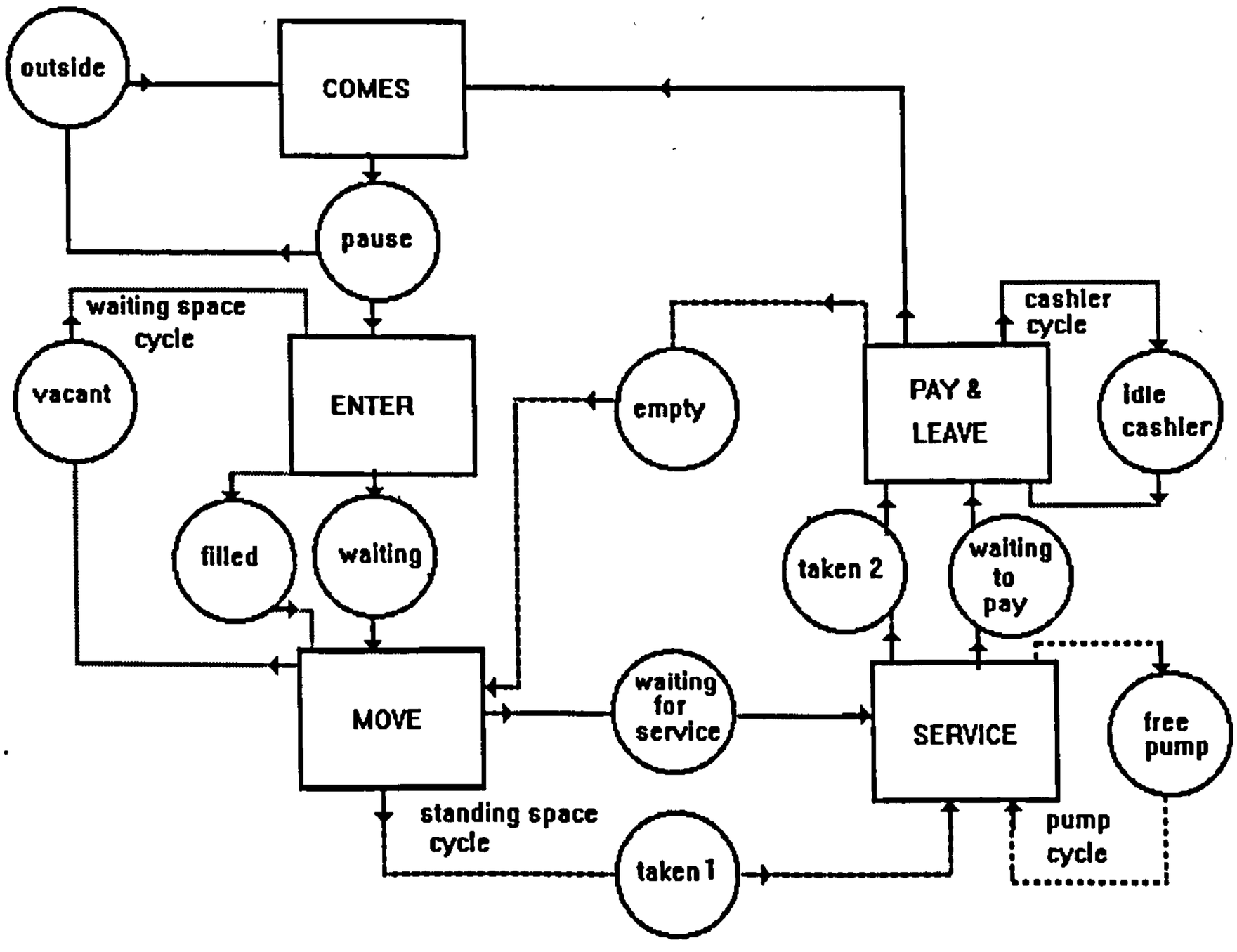
Table 1 Entities and activities

Activity Entity	COMES	ENTER	MOVE	SERVICE	PAY
Car		C	M	M	D
Waiting space		M			
Serving space	M		M	M	M
Pumps				M	
Cashier					M

Note that instances of the entity class *car* can be created(C), modified(M) or destroyed(D), but the other entity classes are permanently in the model.

The activity cycle diagram is given in diagram 11.

Diagram 11 The activity cycle diagram for the petrol station forecourt



The ACD does not show clearly that for the activity SERVICE to start a **particular** ***pump*** will be required according to the ***space*** occupied by the ***car***. There is no provision for showing, on the diagram, the specific conditions for each activity to start. It is only obvious that certain entity types must be present in the queues feeding an activity block.

The ECSL coding allows for attributes to be attached to entities, and the condition at the head of the SERVICE activity block will test that the pump matches the space. In practice faults can occur in ECSL programs in the looping structure of the condition block. For example, a common coding mistake is that: the first waiting car is selected with its space, and each pump which is free is checked for that car, but in the case of failure, the process is not repeated for another waiting car for which service could well start. The correct and faulty coding looks very similar, and very careful verification is needed to identify the mistake.

The ACD could be drawn with separate MOVE activities for different *spaces*, and separate SERVICE activities for the different *pumps*, and code could be produced to match. However this model would lack generality and could not be easily edited for a forecourt with different numbers of facilities.

3.4 Evaluation of the three strategies

The three strategies are compared according to their usefulness for the given problem using the guidelines set out in section 3.1.2.

3.4.1 Model construction

It is evident that each of the methods facilitate the initial **conceptualisation** process. The method preferred will largely depend upon prior experience. It has been suggested that those without prior experience find the flow of entities shown in Activity Cycle Diagrams more easy to understand [Shearn 1990].

3.4.2 Verification of model logic

A clear depiction of the **logic structure** in the diagrammatic representation would help the **verification** process. For this problem the diagrams failed to represent the spatial aspect. The Petri net could represent the identity of pump available, using tokens, but not whether the pump would reach.

In general the logic is specified in a procedural fashion, and little help is given for checking whether the implementation fulfils the intended specification. For instance, the **number of resources** and **required attributes** of resources are not indicated on activity cycle diagrams and static Petri nets. **Simulation nets** [Torn 1985], represent conditions and levels of resources without ambiguity but with an increase in complexity.

None of the three methods show clearly the fairly simple **spatial constraints** of the selected problem. Verification of the logic would depend upon a careful examination of model performance, which is easier when graphics provide real time animation of entity movements [Syspack (VS7) 1990]. The petrol forecourt problems have been modelled, with and without the aid of graphics, using ECSL/CAPS, VS7 and WITNESS, and graphics facilities were very helpful in verifying program logic [Tank 1992].

3.4.3 Further development work and implementation

Further development work implies that a simple model may be enriched or a complex model may be decomposed or simplified using the existing model, without making a fresh start. The modelling strategy should incorporate a method for modification using a hierarchy of models. In practice all three methods have been applied in this way, but since they follow a procedural paradigm the modification process is difficult. Aimo Torn's modification of Petri nets allow for sections of the model to be expanded in separate diagrams [Torn 1985], and in the activity cycle diagrams activities can be bound together or considered separately.

The three methods are general modelling strategies and have implementation in several special purpose languages, but once the model has been constructed in the form of a diagram, it can be directly translated into a computer program. However, for the problem illustrated, the diagrammatic representations did not help in the coding of the allocation of a pump to a car according to the space occupied.

It is claimed that simulation graphs can be shown to correspond to programs written in various languages [Torn 1981], and Activity Cycle Diagrams give the necessary data for program generators [Clementson 1982]. The modification process may require the editing of source code, or may be possible using the generator's input file [Balmer & Paul 1986].

3.5 Successful practice in applying the modelling strategies

The procedural paradigm, in the forms described in section 3.3, has been successfully used in modelling spatial problems where the movements between locations can be modelled explicitly as activities.

Such models

- involve a comparatively small number of entities, and are micro as opposed to macro, cf. sections 2.1.2. and 2.1.3.2,
- and use an algorithm in which the simulation logic is specified for the particular problem, represented in a flow chart with full details for all parts of the organisation.

A few case studies are selected in which the layout is crucially important. Each one uses software using the procedural paradigm.

3.5.1 Case studies using discrete-event simulation

A simulation study was carried out for the automated guided-vehicle system at the planning stage of a new hospital at Zagreb, Yugoslavia. [Ceric 1990]. The model included the complete transportation system for shipments of food, linen, supply materials and waste, and evaluated the robustness of the physical provision for the tasks scheduled. SIMSCRIPT II.5 was preferred to GPSS because of the complexity of the model. The data representation used 'a network-like database structure', which included spatial nodes and needed a testing module. The simulation model flowchart was associated with the hospital layout, and represented spatial components as resources needed in activities.

For planning the Stanlow and Tranmere waterfronts [Danielsen, Eldridge and Brown 1991], a detailed layout was required which included a tidal entrance bar, docks, an oil refinery and berths of various types. Entity life cycle diagrams (ELCD) were used to model the system and all locations were represented within the flow diagram. A visually interactive model was required and the VIS package GENETIK produced a successful model with graphical display.

A model of a solid waste processing system [Ceric & Hlupic 1993], carried out over eight major locations, gives adequate representation of the various movements by incorporating them within activities in the activity cycle diagrams. A generic software package, VS6 was used to produce a successful model.

A computer simulation system was developed by Bhattacharyya, Roy and Low for final car assembly on a fixed paced moving conveyor track on which manufacturing processes were carried out by teams of operatives at stations placed alongside [Bhattacharyya, Roy & Low 1993]. The fixed pace of the conveyor sometimes caused operatives who took longer than expected on one station to drift into the next and cause interference with subsequent work. For this reason the balanced man assignment to each work station was crucial to production efficiency.

A graphical representation of the assembly track was produced using SEE WHY and a discrete-event model achieved by modelling the continuous movement of the conveyor in discrete jumps. Operatives were modelled individually and their movements monitored between the moving track and the storage depots stationed alongside. The system was produced as a generic simulation model to evaluate man assignments on the assembly tracks.

3.5.2 A case study using mixed discrete-event and continuous simulation

Manufacturing systems in which the product is continuous or numerous can be modelled using a mixed discrete-event and continuous simulation.

One example is the simulation needed to determine optimal scheduling of activities and the location of equipment in the Elura Lead/Zinc Mine in New South Wales [Hoare and Willis 1992]. Congestion was anticipated on slopes deep down in the mine where trucks loaded with ore were hauled level with the crusher. The organisation of the mine is a good example of dynamic interactive processes requiring discrete-event and continuous simulation modelling as described in section 2.3.1. Mixed facilities were provided by the package SLAM II, and the model was further modified to run under SIMAN with animation provided by CINEMA.

This was a large scale complex model in which space was an important constraint. The details of the physical layout of the mine were explicitly modelled with graphical output to aid interaction. This is one example of the many successful uses of mixed discrete and continuous modelling.

Slam II is described by the authors as 'easy to learn, easy to use', but it is also said to be a 'multipurpose platform', and cannot be described as generic data-driven software. The model was specially developed for the Elura Lead/Zinc Mine and the time taken for model development is not given.

3.5.3 Common features of the methodology

The case studies are typical of the type of simulation on spatial problems, carried out using any one of a range of software products available for discrete-event simulation. In each case a model flowchart was specially designed for the problem in which the spatial elements were included as resources, but in some cases, data-driven generic software had been used to speed model development. This will be discussed in Chapter 4.

In general, model structure was not based on the spatial layout, although Ceric used a database to represent spatial data [Ceric 1990].

3.6 Spatial problems for which the modelling strategies are inadequate

There are situations where special purpose simulation software using the procedural paradigm proves to be inadequate. The problems which occur are tackled differently according to whether a macro or a micro model is required.

If the objectives of the simulation are satisfied by recording macro-performance measures, the moving entities can be represented as numerical attributes of other model components. This method is applicable when the moving entities are numerous and the resulting model is quasi-continuous. A selection of problems presented in sections 3.6.1, 3.6.2 , 3.6.3 and 3.6.4 demonstrate this approach.

The modelling difficulties are greater when it is necessary to trace the path of individual entities, and this is discussed using illustrative problems in sections 3.6.4 and 3.6.5.

3.6.1. Simulation software for manufacturing plant design

A whole range of analytical and computer algorithms have been developed for the solution of problems concerning the configurations of plant for manufacturing processes and the location of facilities [Francis, McGinnis & White 1992].

Although manufacturing problems have been used to demonstrate the usefulness of generic simulation software packages, Pidd notes that most of the packages available were 'material-oriented' rather than 'machine-oriented' [Pidd 1992]. The software was not well-suited to modelling certain manufacturing processes which converted raw material, in continuous form, into products which are discrete but numerous. The confectionery industry gives example of such quasi-continuous manufacturing processes [Pidd 1987].

Pidd developed a prototype data-driven simulation modelling tool for varying plant configurations, SKIM, for use by design engineers [Pidd 1992]. The model is essentially spatial, consisting of *"a sequence of linked processes which may include intervening buffers"*, which were defined in a spreadsheet data-entry format. There was an option to create a 'block diagram' for the graphics screen. but, as noted in section 2.4.5, Pidd commented that the engineers *"would have much preferred a graphical user interface (GUI) to be used"*.

This software is cited in section 2.4.4. as an example of a domain specific generic model.

3.6.2 Simulation of pedestrian traffic flows

Modelling the movement of people is important for building design and has been tackled in various ways and these are discussed in detail in Chapter 6. An illustrative example, which is in the from of a quasi-continuous model, is included here.

The London Regional Transport required a model for the movement of people in underground stations for station management and design. It was especially important for the model to represent passenger movement under congested conditions. The use of standard software simulation packages was ruled out since a simulation model which represented passengers individually would be unacceptably large and complex. The data-driven generic model developed by Weston, used a spatial layout in which passengers were modelled as attributes of component spaces. The specific problem domain was passenger movement in stations, and the model was developed in FORTRAN using SIMON procedures to generate the random samples required.

The congestion model could be used to model any station given a complete data-file, consisting of four parts giving information about:

- station layout, given as a list of spatial blocks with dimensions, links between blocks and limits upon passenger movement rates from one block to another,
- train frequency and capacity,
- passenger arrivals from the street,
- passenger arrivals from trains.

The last three parts would be dependent upon the time of day.

In order to model correctly the target standing position on the platform for passengers arriving from the street, and, for passengers arriving from trains, the car from which they alighted, the data-file listed all possible routes from ticket-office to platform, and from platform to street as lists of blocks with the corresponding proportion of passengers observed to select that route. For realism the simulation allowed passengers to board if a train arrived before the target platform position was reached. Since passengers were modelled as numerical attributes of spatial blocks, the implementation was efficient.

The model was used for the Angel underground station in London when advice was needed about the safe provision for projected increases in passenger traffic, and the study influenced the eventual decision to build a new tunnel for the refurbished station [Weston & Shapiro 1987].

There are reports of its use for other underground stations, [Weston & Smith 1987, Weston 1988], and the congestion model has since been produced with graphics as the package PEDROUTE, which is available for the simulation of public buildings and stadiums. [Appendix C]

There are similarities between models produced for the movement of pedestrians along corridors and vehicles along roads [MacGregor Smith 1994, Cheah & MacGregor Smith 1994]. However the congestion model involves two-way flow with either a fixed source or a fixed destination, whereas vehicular traffic is typically two-way movement on a network of roads. The simulation of vehicular traffic is discussed in section 3.6.4..

In the case of elevator traffic the movement patterns are well-defined. Two simulation models for elevator traffic are described in section 3.6.3.

3.6.3 Simulation of elevator traffic

Elevator traffic can be modelled using bulk service queueing theory, producing expected passenger waiting and journey times as performance indicators [Bailey 1954]. A macro simulation model is appropriate in this situation since passengers are numerous and need not be traced individually. Such a model involves two-way linear movement for both lifts and passengers, and two examples are described.

A procedural approach was used to produce the Advanced Elevator Traffic Simulation (ALTS) for the KONE Elevators Research Center in Helsinki [Siikonen 1993]. This was a generic model for a hypothetical building in which the number of floors, lifts, occupancy and passenger demand could be specified by the user. The model was developed in FORTRAN-77 with GKS graphics, but spatial relationships were implicit within the model logic. ALTS was verified using a systematic comparison of simulation runs with theoretical results. The generic model can be run to evaluate the effectiveness of different call allocation principles in varying situations.

Galpin and Rock also considered that a special purpose simulation language would not be helpful when planning to develop an experimental tool for examining the effects of different scheduling policies for a bank of lifts, and chose to use Pascal instead [Galpin & Rock 1995]. Their decision was affected by the fact that a measure of the overall quality of service of the lifts was required, but that the simulation model did not need to trace the path of individual lift passengers. Also the rules of the simulation were expected to be complex.

The authors quoted a recommendation of Bell and O'Keefe that it is good practice to develop the visual part of the display followed by the logic to drive the display [Bell & O'Keefe 1987]. This method of model construction is also adopted by simulation software packages such as WITNESS and VS7.

Galpin and Rock produced a fixed-time increment, stochastic model based upon a set of data structures consisting of an array of lifts and an array of floors. The visual display and the data structures correspond and define the spatial layout. Their objective was to produce a tool which would enable a person, without simulation expertise, to specify the system of lifts and carry out experiments.

Both Siikman and Galpin and Rock produced good examples of generic data-driven simulation models.

3.6.4 Simulation of vehicular traffic

Road traffic models are needed for both traffic management and for long term planning purposes. The computer software available for traffic management ranges from programs which model detailed junction design, to those which help with the planning of linked signal systems and complete road networks*. There are numerous reports which indicate that transportation planners world-wide seek methods for modelling the performance of freight carrying highway networks [Cochran & Lin 1989, Ash & Waters 1991, Hellinga & Van Aerde 1994, Pope, Rakes, Rees & Crouch 1995].

Standard operational research techniques, such as linear programming network models, dynamic programming and n-person game theory, assume simplified models which are homogeneous and repetitive. One example is the assignment algorithm enhanced by simulation, used to find optimal transport routes for coal [Ash & Waters 1991]; another is a systematic search algorithm used to evaluate the effects of road widening to remove bottlenecks at different locations of a road network [Janson 1995]. Stochastic processes and queueing theory also require a simplified model, but have been extensively developed, using linked queues and decomposition methods to model traffic network problems [MacGregor Smith 1994, Dai, Nguyen & Reiman 1994, Son, Cassidy & Modonat 1995, Alfa & Neuts 1995]

---

* Traffic Signal Design Calculation, supplied by:
The Traffic Control Systems Unit, Kings Building, Smith Square, London SW1 3HQ

Simulation is a favoured technique for modelling traffic networks because of the complexity and diversity of the problems tacked. Several authors report the rejection of generic simulation packages in favour of either a general purpose language or a simulation language with continuous facilities. The modelling methods vary according to the size and complexity of the road network, and whether a macroscopic model is thought to be adequate. For instance, the TRAFLO model developed for the Federal Highway Administration, New York State, provided, as options, three levels of modelling: microscopic, platoon based simulation and macroscopic [Eiger, Niedwski & Erikson 1983].

For modelling large scale traffic networks, where the moving entities are numerous, many authors report the use of macroscopic quasi-continuous models, in which individual vehicle characteristics are not recorded [Berger & Shaw 1977, Makigami, Nakanishi & Seill 1984, Hu & Schonfeld 1984, Pope, Rakes, Rees & Crouch 1995]. A critical feature of all models of vehicular traffic is the inverse relationship between rate of flow and traffic density. A similar property exists for pedestrian flow as shown in Table 5 section 6.1.1.

For a model of a 24 km section of Japanese expressway, traffic is modelled as a compressed fluid [Makigami, Nakanishi & Seill 1984]. This study benefited from an extensive traffic survey, in contrast to the model produced of Hampton Roads, Virginia, USA, based on a complex network and including mixed traffic (container and non-container traffic), for which complete origin-destination data was unobtainable [Pope, Rakes, Rees & Crouch 1995]. The authors depended upon traffic volumes on each route, the division of traffic between container and non-container and the branching of traffic at each intersection, supplemented by interviews with container lorry drivers. The network oriented simulation language SLAM II with Q-GERT was selected and applied to a network consisting of 350 nodes and 709 activities. The size of the network is comparable to that of the TRAFLO model which contained 60-100 freeway nodes, 200-250 street nodes and 700 links [Eiger, Niewdski & Erikson]. The Hampton Roads model was validated and claimed to be a useful tool for planning purposes with low costs on survey data collection.

The Arizona Freight Network Analysis (AFNA) model, based on a network of 81 nodes and 804 links was programmed in FORTRAN combined with SLAM II statements using a link-based conditional probability branching discrete-event method [Cochran & Lin 1989]. AFRA modelled individual vehicles attributes and was designed to be economic on survey data, using origin-destinations for freight hauling trips together

with their commodity types and gross weights from a sampled selection of companies. The model design also economised on the number of nodes, each one of which represented the complete interchange. Careful validation of model results showed that the methods used were reliable. A more detailed model for a smaller network was produced for Highway 401 freeway traffic [Hellinga & Van Aerde 1994]. This was a microscopic routing oriented simulation model of integrated freeway and surface street network, and was named INTEGRATION. It used origin destination traffic demand data and produced separate graphical output for eastbound and westbound traffic.

The AFNA and INTEGRATION are examples of how computer simulation is appropriate for large scale complex transport systems which require detailed survey results. Origin-destination matrices for trips, and traffic counts on links of the network are usually available and can be used by such models.

There is another class of models concerned with the effects of blockages and overtaking opportunities which are required for modelling two-lane rural roads with few junctions, entrances and exits. Such models require vehicle characteristics and driver behaviour to be modelled in conjunction with road topography, and are necessarily microscopic. Khasnabis and Heimbach modified the existing Franklin Institute Research Laboratory model for a two-lane roadway with passing, written in FORTRAN IV, incorporating different vehicle characteristics for freight carrying trucks, headways which vary according to traffic volume and speed, and additional exits allowing some vehicles to travel only part of the highway [Khasnabis & Heimbach 1977]. Marrall and his co-authors adopted a model originally developed by the Australian Road Research Board, TRARR, which required data in three forms: physical attributes of the road, subdivided into sections of length 0.16 km, vehicle characteristics and traffic parameters. Detailed output included the frequency of overtaking, mean vehicle speed, (overall and by vehicle category) and the percentage of following traffic [Marrall, Miller, Smith, Feuerstein & Yazdan 1995].

The modelling problems encountered in simulating traffic networks arise from:

- the size of the system,
- the large number of moving entities,
- the complexity of the system,

which are similar to the problems reported in sections 3.6.1, 3.6.2 and 3.6.3.

Survey data is needed in a suitable format for the model and at an appropriate level of detail for the investigation and projected budget. Macroscopic models are used where

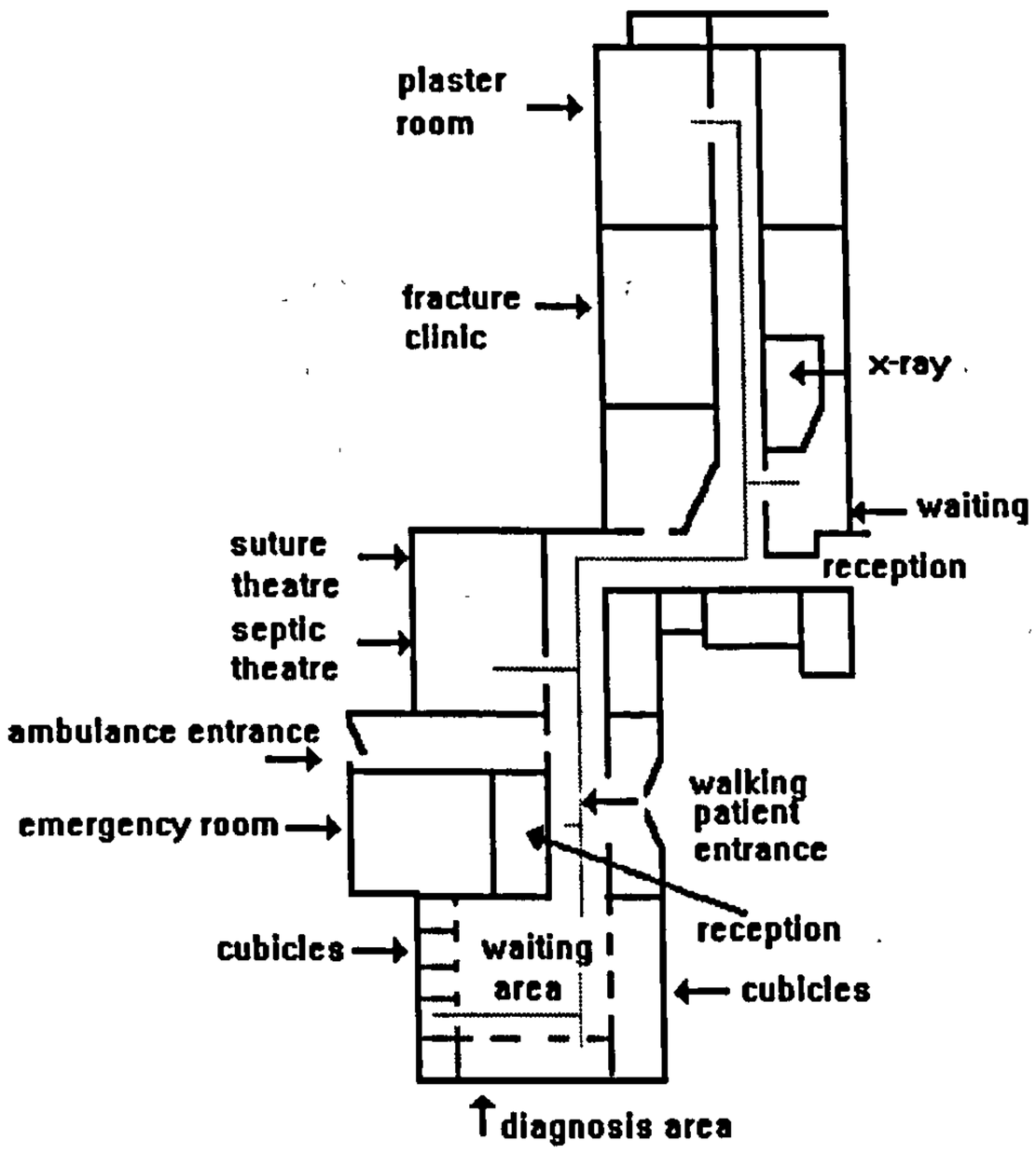
approximate results are adequate. Microscopic models tend to be expensive on both data requirements and computation and are generally applied to relatively small-scale traffic networks. One compromise is to model platoons of vehicles [Hellinga & Van Aerde 1994, Alfa & Neuts 1995], but the probabilistic approach demonstrated by Cochran and Lin gives an efficient microscopic model [Cochran & Lin 1989].

3.6.5 The King's College Hospital AED

A simulation model of a hospital department which seeks to evaluate the service given must record the individual progress of patients. The macroscopic quasi-continuous modelling methods described in sections 3.6.1, 3.6.2 and 3.6.3 will not be adequate, but analogy with the microscopic modelling strategies applied in section 3.6.4 may be helpful.

A simulation of the Accident and Emergency Department at King's College Hospital [Shapiro 1976], was required for evaluating the design and provision of the existing accommodation. The AED was housed in an old building in which the facilities for plaster and X-ray were accessed along narrow corridors some distance from the main examination and diagnosis areas, as shown in diagram 12. Patients were often accompanied by attendant relatives, and a few patients needed a porter to assist them from one location to another.

Diagram 12 The King's College Hospital Accident and Emergency Department



A model was produced in ECSL, using an activity-based approach as described in section 3.3.3, but there were difficulties in producing a model which represented the routine movements of patients and staff. As a compromise, movement times for patients between facilities were estimated and added to the appropriate treatment times. The model gave good agreement with observed patient through-put and waiting times, but information about corridor congestion and porter utilisation was lacking. The alternative strategy of adding further activities using the corridor sections to the existing model would have produced unacceptable complexity since this corridor space was used in common with other users, including hospital and ambulance staff and police.

To model corridor congestion effectively requires a different type of algorithm. The proposed methodology suited to this type of problem is introduced in Chapter 5. Hospital design is further discussed in Chapter 8.

3.6.6 Facilities for continuous simulation

The problems described in the immediately preceding sections arise from quasi-continuous properties of the system to be modelled. There are facilities for modelling continuous processes, as mentioned in section 2.3.1, and it is sometimes useful to model continuous processes within a discrete event simulation.

For instance the problems described in 3.6.2 might be solved by modelling the movement of people "en mass" as a continuous flow, and facilities are available in several simulation software packages:

- ECSL provides a DYNAMICS activity block, which is updated at each time step within the three phase algorithm.

- GASP provides a sophisticated control algorithm which allows for modification of the time step to model continuous processes to the required accuracy.

- The physical elements provided in WITNESS - fluids, tanks, pipes and processors for continuous modelling correspond to parts, buffers conveyors and machines for discrete simulation.

The movement of people in a building could be modelled in WITNESS using fluid to represent the occupants, tanks to represent rooms, pipes to represent sections of corridor and processors to represent junctions between corridors where the flows are mixed.

Several problems arise:

- It is not obvious how the relationship between flow rate and density could be modelled.
- A separate model must be created for each individual building.
- The model created would be complex and have no homogenous structure.

These disadvantages do not preclude the design of mixed discrete-continuous models for special problems; successful case-studies are described in sections 3.5.2 and 3.6.4. However the approach is rejected for the spatial problems described in Chapter 4 and Chapter 8. Designs for generic models are introduced in Chapter 5 and Chapter 9.

3.7 Implications for developing an improved strategy

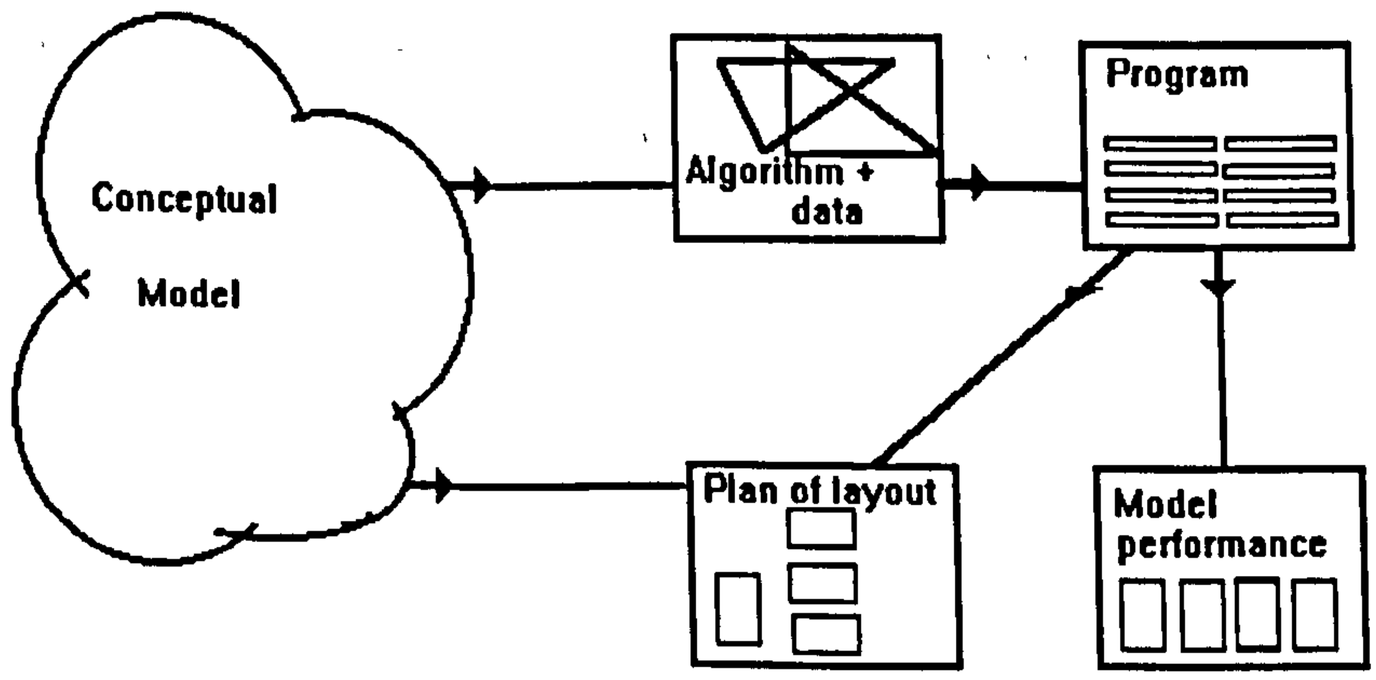
The modelling strategies have been evaluated against the guidelines set out in 3.1.2. All three follow a procedural paradigm, and there are no outstanding reasons for preferring one to another according to their diagrammatic representations; certainly the algorithms are very similar. In practice, it is the comprehensive facilities offered by software, particularly program generation, which prove to be useful for the requirements discussed. Several types of software were used in the case studies described.

The third requirement was to facilitate further model development work. This is consistent with the good software engineering practice of providing reusable modular units in a supportive environment. A modelling strategy using an object-oriented approach would make this easier to deliver [McGregor & Sykes 1992].

For the problem examined, we require a software design which incorporates the spatial interactions of the system within the data structure rules; one which binds together the **data** associated with the system entities, and the **rules and constraints** which govern them.

In each of the strategies using the procedural paradigm the model does not include spatial interactions. The representation of the layout is constructed independently, and performance data generated by the program model is mapped onto it for verification and demonstration purposes. This is illustrated in Diagram 13.

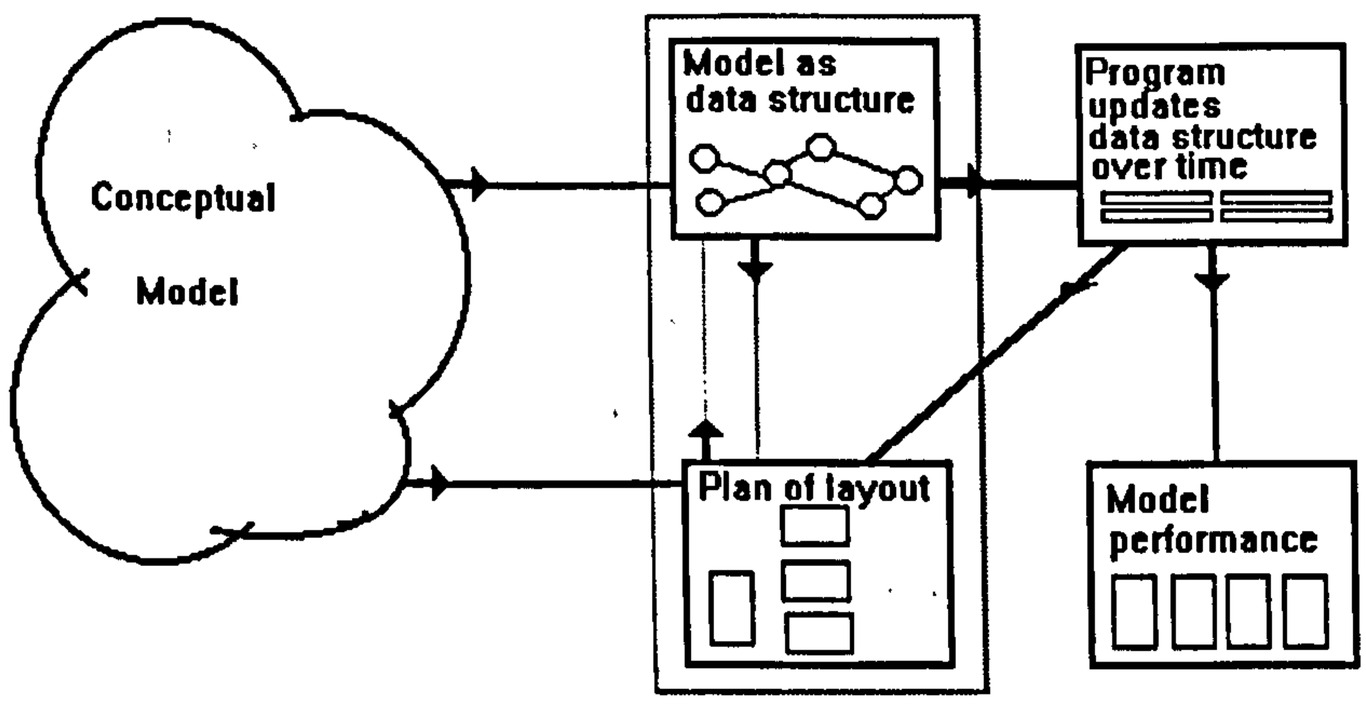
Diagram 13 A procedural modelling strategy



A modelling strategy using a graph structure for the spatial components of the system, would have the logical constraints embedded in the structure itself, and the physical layout would also be directly represented. This approach is illustrated in Diagram 14. The program would be the controlling mechanism which updates a time clock, monitors changes to the data structure, and communicates these changes to the layout model and summary performance records.

Diagram 14

A modelling strategy using a data structure based on spatial relationships



The thesis presents 'domain specific' generic models using this methodology.

In Chapter 5 a model for emergency evacuation of a building is described and further demonstrated in Chapter 7.

In Chapter 9 extensions of the methodology are presented, demonstrated using a model for nurse activity on a hospital ward.

Chapter 4 Spatial Problems

4.1 Modelling spatial problems

Space is a resource in any simulation of a physical system. There are numerous examples where spatial features have been successfully modelled as a resource, or embedded within a model logic specially designed for that particular problem. Some have been referred to in the previous chapter, in sections 3.5.1 and 3.5.2. The representation of spatial characteristics has been selective and model construction of this nature is an expert task.

4.1.1 A new methodology is required

For a problem, in which:

- activities take place at different locations,
- the necessary journeys between those locations are affected by multiple changes to the environment and the movement of other entities,

it is proposed that the spatial characteristics are represented within a data-file.

The construction of this file would be a non-expert descriptive task, whereas the simulation program would contain the comparatively complex model logic.

A concept of space as a continuum in one, two or three dimensions, underlies analytical and continuous simulation models. The concept is retained in the example which uses mixed continuous and discrete simulation in Chapter 3 section 3.5.2, and in the case studies using other modelling techniques described in section 4.3.1.

The thesis recommends an interpretation of digital discrete-event simulation in which space is modelled in discrete, conveniently sized portions. This allows the spatial characteristics of the model to be listed in a simple format.

The digital model is explained in section 4.2.

4.1 2 Special features of the spatial model

A simulation model for such a system has three distinguishing features:

- It uses a large static data structure to represent the physical environment, since there will be a large number of similar spatial components with attributes which do not change over time. This may be a graph structure which can be diagrammatically represented as a network.

- It has a separate simulation engine which incorporates the general rules for change, designed for a class of problems. The rules are sufficiently simple and generalised to be repeated throughout the data structure.

- In order to represent movement between locations, the model will be quasi-continuous over space and time.

In effect, a digital mapping is imposed upon space in a similar manner to that used for time. Space and time increments are closely related to give the desired fit as described in section 4.2.

4.2 The digital model for discrete-event simulation

It is recommended that a spatial digital model is created analogous to the digital model for time. The features of digital discrete-event simulation, as introduced in 2.3.2, need to be examined more closely.

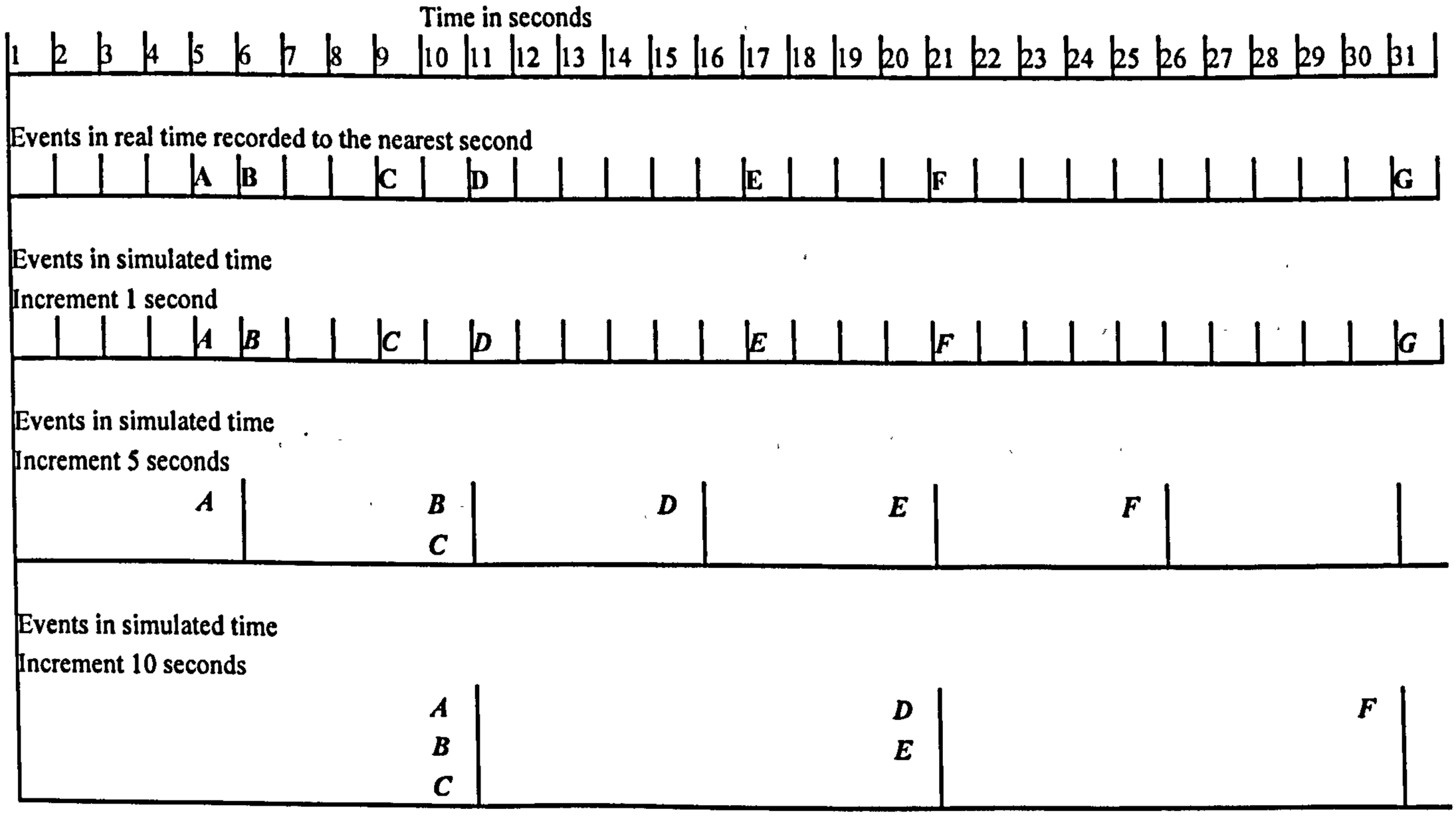
4.2.1 A digital model for time

Changes in the real system occur at points of time on a continuous scale. A digital simulation model artificially represents these events using a digital clock which assumes a finite incremental step. The choice of the size of this increment is dependent upon the problem context.

- An increment is too large if many events occur within the chosen time slot. This would model sequential events as though they were synchronous and execute them in an arbitrary order which may or may not produce an accurate model.

- An increment is too small if many steps are taken without events occurring, producing inefficient slow computation.

Diagram 15 Events in real and simulated time



The diagram shows that, when a 10 second increment is used, event A is modelled 5 seconds late, and event F, 9 seconds late. Also this model shows events A, B and C to occur simultaneously at time 10, and these events may not be executed in their correct sequential order. The model becomes progressively less accurate as the time-step is increased.

The choice of the length of the time-step is less crucial if time is updated using the three-phase method as described in 3.3.1. In effect this method allows the time clock to be updated in variable time-steps, and the unit for measuring time can be small without loss of efficiency.

4.2.2 A digital model for space

The recommended spatial model takes the form of a two-dimensional rectangular grid, and adopts a finite increment. In some situations a three dimensional grid would be needed. The problems in selecting the dimensions for each cell of this grid are similar to those for choosing a time step, and will depend upon the activities in the context of the problem.

Each cell must be small enough to define the location of the separate activities. In addition, cells represent spaces travelled through between activities. Each cell is a nodal point on a route, and needs to have dimensions appropriate to the time step chosen and realistic movement speeds. In general cells need not be of equal dimensions but careful calibration is required.

Time is modelled on an ordinal linear scale, but changes of state in space may follow many alternative routes. Spatial constraint on movement is modelled using a graph structure, in which the nodes are the cells of the spatial grid, and the arcs are the possible movements between cells.

The method can be compared with geographical information systems (GIS) which use regular rectangular grids and triangulated irregular networks (TIN), the former being more convenient for visualisation of the data [Jones, Kidner & Ware, 1994]. The graph structure, like the TIN, will not require regular equal cell dimensions, but the graphical display will reference regular rectangular co-ordinates.

4.3 The proposed model structure

To comply with 4.1.1 it is proposed that the model has two features:

- a graph structure, represented as a network, to model the spatial data,
- a data file which gives full model definition.

The resulting model will be a data driven 'domain specific' generic model

Evidence of the use of both features can be found in case studies for spatial problems. Examples are given in sections 4.3.1 and 4.3.2., and a few generic data-driven models are described in section 4.3.3.

In addition object oriented methods will be used both in model design and implementation in order to preserve model simplicity and replication of function, which is discussed in section 4.4.

4.3.1 The use of networks

Transportation and distribution problems are essentially spatial, and networks are frequently used as an analytical tool for finding and evaluating routes. Networks are used with a variety of methods, both deterministic and stochastic, including Linear Programming and Monte Carlo and discrete-event simulation.

Formulation of the problem using Linear programming is used for a range of applications [Alttinkemer 1994, Carey & Srinivasan 1994], and Linear programming is an established method, using a cost matrix representing the data for the network. This was applied to a network flow algorithm for traffic congestion with trials in Leicester [Martin 1993].

Drezner and Wesolowsky considered the solution of location problems on a regular grid of alternating one-way routes or streets where the facility and demand points are situated at nodal points of the network. The computational algorithm to produce a range of feasible solutions was tested on multiple randomly generated problems [Drezner & Wesolowsky 1995].

Network structures are also used in queueing theory and simulation models. Some examples are given of this approach.

- Complex processing problems may be modelled as networks, and decomposed into groups of interacting linked queues which can be analysed separately [Whitt 1993, Dai, Nguyen & Reiman 1994].

- MacGregor Smith demonstrated the evaluation of a design for a large regional hospital campus for pedestrian/vehicular circulation using a three-dimensional network in which each node was modelled as a queue [MacGregor Smith 1994].

- The Arizona Freight Network Analysis (AFNA) project used simulation to model the flow of freight hauling vehicles on the Arizona state-highway network system [Cochran & Lin 1989].

- For strategic choice of routes for the transportation of coal across Canada, Ash and Williams describe an evaluation of the routes by a sequential assignment of sublinks according to cost within repeated simulations. Varied data is used to represent alternative costs and constraints to give a decision making tool [Ash & Williams 1991].

There are distinct differences in the way the networks were applied in the above examples. Each arc in the networks used by MacGregor Smith and by Cochran and Lin represented a physical displacement, whereas Ash and Waters included arcs in the 'Network of possible transport routes' to represent processes, and their simulation was a decision analysis to select a feasible mix of routes with reduced total costs.

A range of network simulations and discrete-event simulations were used for freight transport systems in Arizona [Cochran and Lin 1989]. The favoured model used link-based conditional probability branching within a discrete-event simulation. This method was selected because it gave a valid model and needed data which was possible to obtain from mail surveys. The model was implemented in SLAM II, using a dBASE III database to contain the freight-flow data from the mail surveys, giving separation of the model and the topological data.

The model included both macro and micro parameters:

- macro parameters, for traffic volume and density, with intersections treated as impeding paths as in hydrodynamics,
- and micro parameters, for lorry capacity and vehicle type.

Cochran & Lin describe several model structures, some of which have similarities with the approach used by Weston to model passengers' movement in railway stations [Weston & Shapiro 1987]. These methods are discussed in Chapter 9 section 9.7.

4.3.2 The use of large data files to input model data

A feature of spatial problems is the large amount of information representing the topology of the system. For deterministic models, one method of coping with a large problem is to use heuristic search methods [Johnson & Brandeau, Altinkemer 1994], but simulation models can represent the topological data as a static database, which is particular to the system being modelled but independent of the simulation algorithm. This allows for separation of the data defining the system from the control program, and gives a data-driven generic model.

Several instances of this approach are quoted:

- A large data file is needed for the station layout and passenger routes between platform and street, for the LRT Congestion Model [Weston & Shapiro 1987], which was later developed as PEDROUTE [Appendix C].

- It is observed that Ash and Waters use the spreadsheet Lotus 1-2-3, for the input data, the simulation being coded in APL [Ash & Waters 1991], whereas Cochran and Lin use dBASE III for the input file and SLAM II for the simulation [Cochran & Lin 1989].

- The system developed by Williams Gittins and Burke was more sophisticated and incorporated an expert system. It used both a static and dynamic database for the complex system ADMIRAL, a management aid for the replenishment of ships at sea [Williams, Gittins & Burke 1989].

- Large volumes of man assignment data, recorded in a relational database, was required to support the computer simulation of man assignment on car assembly tracks [Bhattacharyya, Roy & Low 1993].

4.3.3 Data-driven generic software

There are software generating tools which allow the user to define the model in the form of a data-file. Several packages have been developed using the procedural paradigm and provide help software to build the model given a specification in terms of entities and activities, [DRAFT, ECSL CAPS, VS7, WITNESS] These are referred to in sections 2.4.3 and 2.4.4.

Although promoted for general use, most software packages were 'material-oriented' and not suitable for 'machine-oriented' quasi-continuous situations [Pidd 1992].

Domain-specific data-driven generic models were developed for quasi-continuous problems which could not be adequately modelled using the generic data-driven software available. Two examples are given.

- **SKIM,** provides a sketchpad package for design engineers, who need to model continuous and batch processes within a discrete-event time handling algorithm. SKIM is a data-driven generic model for plant design, which provides for model specification via textual input from the keyboard, working from the user's own design which incorporates a sequence of standard transformation processes [Pidd 1992].

- **PEDROUTE,** a generic data-driven model developed by Weston, to model the movement of people in congested stations. The original version, called the ***congestion model*** could be used to model any station given a complete data-file for the lay-out and passenger movements for that station. It was used for the Angel underground station in London and several other stations [Weston & Shapiro 1987, Weston & Smith 1987, Weston 1988], and has since been produced with graphics as the package PEDROUTE, which is available for the simulation of public buildings and stadiums. [Appendix C]

4.4 The use of object-oriented methods

4.4.1 The conceptual model

The generic model for a spatial problem requires a data structure that combines functionality with data, which is a fundamental provision in object-oriented system development. There are obvious physical components in the spatial layout which can be identified as objects which are stable and not subject to change. Jacobson comments that 'most object-oriented methods base their structure on the items that have a low probability of change' [Jacobson p.75 1992].

The conceptual tools of the object-oriented approach are well suited to modelling the interaction between the model components implied by the topological features of a spatial layout. The model is dynamic and frequent changes are made, but encapsulation of the data for each object maintains consistency.

4.4.2 Object-oriented programming

The advantages of object-oriented methods for the implementation of the model are the use of inheritance and polymorphism.

- Some economy of coding is possible where descendant objects inherit attributes and methods already designed and tested. Using inheritance whole groups of object classes, together with their methods, can be reused, for entirely altered purposes.
- Polymorphism allows methods to be used with adapted implementation of the function at the time of execution, which gives simplicity of coding. However Turbo Pascal is strongly typed and only limited use can be made of this facility.

4.4.3 Software reuse

Turbo Vision provides object classes with a library of methods which can be used as primitive objects. The object TCollection, which represents a list and behaves like a linked list or an array, is usefully applied to the modelling of a graph structure which calls for extensive use of lists.

The generic model itself is software reuse since it can be applied to other situations with a different data file. A model of this sort is presented in Chapter 5.

Also the methodology adopted in one generic model can be adapted to suit another developed for a different problem domain as explained in Chapter 9.

An objective of software engineering research is to devise methods, given a robust model design, for generating code to implement the model in alternative languages.

4.5 Spatial problem domains

It is proposed to develop generic models which are domain specific which will focus on the aspects of building design and management relating to building functionality, particularly as regards the movements and activities of people.

4.5.1 Published work on building management and design

Quantitative methods have been used extensively in building design to analyse circulation patterns. Published reports have focused on:

- public buildings [Tregenza 1976]
- hospital design [Rawlinson 1971, 1975, Beattie 1974, Thunhurst 1974]
- transport planning [London Transport Board 1958, Weston & Marshall 1973].

Much research has been concerned with setting guidelines for safe and comfortable movement of people within buildings and on public walkways. The study of circulation patterns in public buildings has included fire safety considerations. There are many reports of studies, carried out to assess evacuation procedures and building layouts for safe egress [Bryan, Berlin 1982, Galbreath 1969, O'Leary 1982, Chalmet 1982]. Response times to emergency alerts have been measured, since people's behavioural response is crucial in building evacuation [Pauls 1977, Stahl 1982, Ozel 1992].

Some authors have applied the simulation technique to model people's movement [Stahl 1982, Ozel 1992, Weston & Shapiro 1987], and it is recognised that simulation is a useful aid for building design and management as regards emergency egress

Other published work is concerned with algorithms to assist the design process. Several programs for 'computerised layout planning' are described by Francis, McGinnis and White in their text 'Facility Layout and Location: an Analytical Approach', which were intended for architects and layout planners [Francis, McGinnis & White 1992 pp 94-143].

These same techniques were adopted by the Medical Architecture Unit at the Polytechnic of North London. MARU examined the activities undertaken in hospitals in relation to the building provision and layout, as part of their studies for hospital building design. This included the interactive computer programs, ADAPT & ADEPT which were used, with axiomatic activity patterns and layout designs, to evaluate proposed and existing hospital designs [Beattie 1973, 1974].

Similar algorithms are reported for layout planning for production activities in refurbished buildings [Bozer, Mellor & Erlebacher 1994].

The evaluation of hospital ward layouts is discussed further in Chapter 8.

4.5.2 Simulation as a training and management aid

The published work on emergency egress research is accumulative and generally informs those in overall responsibility for building management. The nature of the response of occupants and their success rate in getting out quickly is studied, but intervening to obtain improvements in performance from the occupants is more problematic.

There have been comparatively few reports on the training potential of ***visually interactive simulation***(VIS) for safe emergency evacuation, even though VIS is well established in many contexts. [Bell 1986, 1989, Hurrion & Secker 1978, Hurrion 1978, Danielsen et al 1991]

One exception is the software developed by Keith Still which uses Virtual Reality software to simulate various emergency scenarios, in which individuals respond differently in different runs of the simulation, according to specific person types seeded in the program [Still 1992 & 1993]. The result is a very graphic illustration of the consequences of combined random responses and would be an acceptable training aid for employed personnel and public users of a building. It is preferable to videos in which similar disastrous scenarios are acted out.

Sophisticated animation is not essential to produce screen output which gives an adequate 'bird's eye view' to employees, whether air-traffic controllers, underground train drivers or ward sisters. Using simple graphics the overall effects of different procedures can be demonstrated effectively under a range of conditions. There have been reports of simulation being used in this role in transport control but no evidence has been noted in nurse management.

4.5.3 Selected problem domains

Problem domains were sought for which there would be recognised benefit from the provision of a good modelling tool. The problem domains should be spatial and involve the movement of people. The two areas which have been selected which satisfy these criteria are:

- emergency egress from a building,
- nurse activity on a hospital ward.

Generic data-driven simulation models will be developed for these two problem areas.

Alternative methods of tackling these problems are reviewed in Chapters 6 and 8, and the proposed modelling strategy is introduced in section Chapter 5.

Chapter 5 Emergency Evacuation of a Building

A simulation model for emergency evacuation has special features. The evacuation process can be described as a queueing situation which involves the movement of people in confined space. It could be treated as a chain of linked queues with batch service, but since there are distinct differences from the type of queue normally modelled using simulation it is useful to adopt another approach.

5.1 Special features of a model for emergency evacuation

Two aspects of the problem are significant:

- **The importance of the physical lay-out**
  The spatial resource is an integral part of the problem. Small changes to the layout and certain critical dimensions can have big effects on the evacuation process.

- **Changes to the system are quasi-continuous**
  The number and distribution of occupants and their responses immediately following an alert are critical. Following this early stage of the evacuation, it can be assumed that corridors become full, which imposes a continuous flow upon the queueing people, which can be modelled as a deterministic process.* The stochastic element is provided by the starting conditions, the initial response of occupants and exogenous events which may happen during the evacuation, such as blockages of escape routes.

For these reasons the model design needs to incorporate the building lay-out, including the spatial interactions and dimensions. The main components of the model will represent the building, and the occupants of the building, may be modelled as attributes of the spatial components.

---

* Note that because of the large numbers of people, who are the entities involved in this process, a global view is taken. A detailed model of movement patterns would take random samples from distributions thought to represent the range of possible movements for that individual. The global view aggregates individual movements into a fluid flow, which is modelled deterministically.

5.2 A generic data-driven model for simulating evacuation

5.2.1 The modelling approach

The model is developed for a specific problem, the evacuation of a building, but the possibility of other applications being required at a later stage influences the basic model structure. This follows good practice for software engineering [McGregor & Sykes 1992]. An object-oriented approach is adopted and primitives are defined with sufficient generality to allow for extensions to the original purpose.

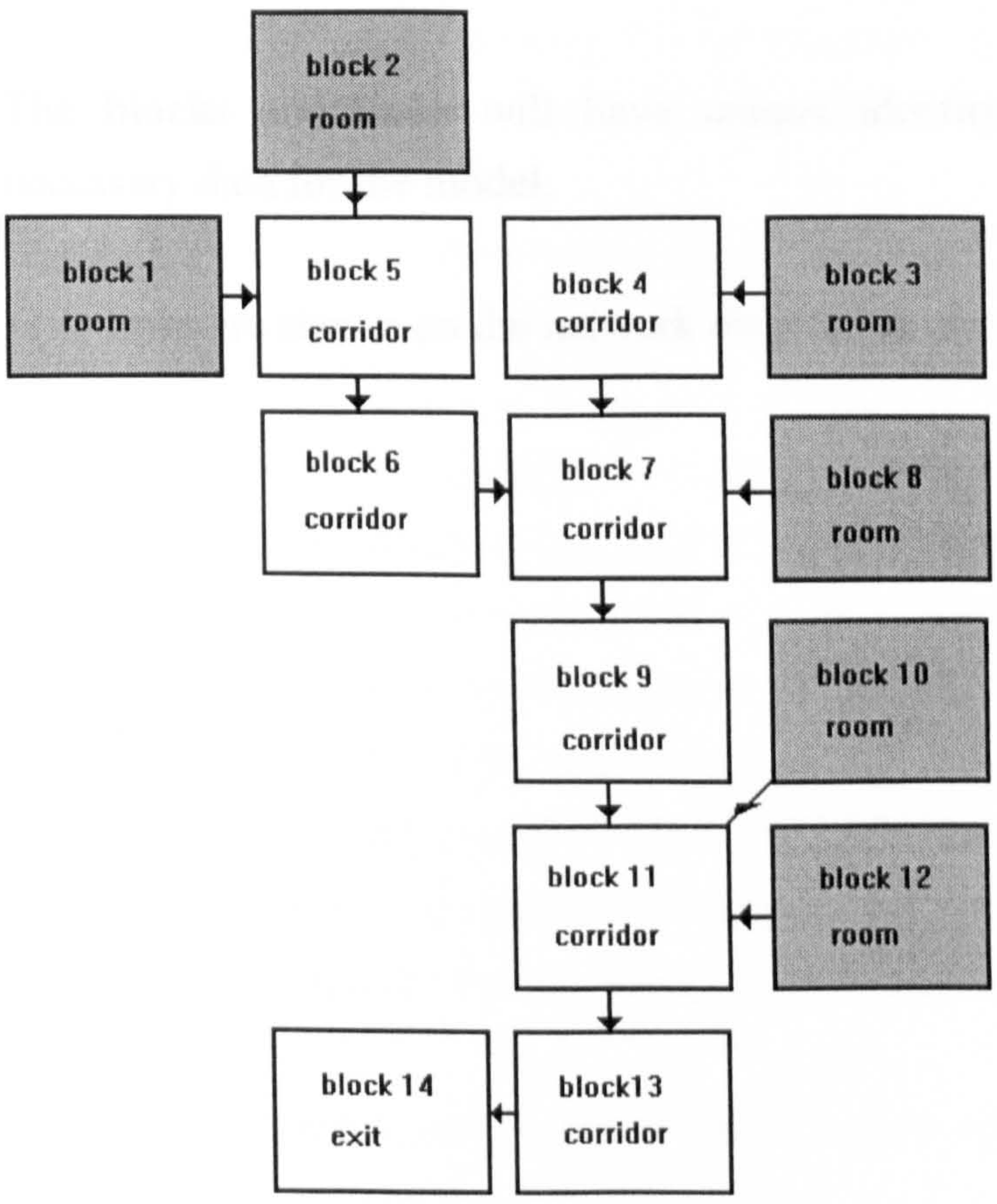
It is important that the conceptual model is closely associated with the physical system so that the model can be easily understood by users unfamiliar with simulation modelling.

5.2.2 The model structure

A simple graph structure provides the framework for modelling the evacuation process. The nodes represent the areas of physical space, either bounded, as in a room, or as a notional portion of space as in a section of corridor. The way in which people may move about the building define the links between the nodes.

A **graph structure** for a small part of a building is given in diagram 16.

Diagram 16 The graph structure for part of a building



5.2.3 The elements of the graph structure

- **The blocks**
  The floor space of the building has been divided up into **blocks**, which may be rooms, sections of corridor, sections of staircase or exits.

- **The links**
  The graph structure requires that **links** are defined between the nodes. In this case the links represent movement of people from one block to another according to the constraints of the model, i.e. from a room through a doorway into a corridor, or from one section of corridor to another.

The **blocks** and **links** will have unique identity numbers and attributes carrying necessary data for the model.

(The links are shown on the network diagram as arrows.)

5.2.4 Standard properties of the graph structure

The standard properties of a graph structure are discussed with reference to this application.*

*A graph structure is **homogeneous** if all of the nodes in it have the same number of outward pointers of the same types used for the same purposes. If a graph is not homogeneous, then it is **heterogeneous**.*

It is apparent that each node of the graph structure has only one outward pointer, which is a feature of evacuation flow. For this model the graph structure is **homogeneous,** although a model of general traffic flow in a building would have a **heterogeneous** graph structure. The property of homogeneity makes it possible to use a simple algorithm to calculate occupancy figures during evacuation.

*A graph structure is **directed** if any link between nodes is one- way: otherwise it is **undirected**.*

For the purposes of emergency evacuation it is assumed that flow out of the building is one-way, and the graph structure is directed. However two-way flow is more realistic, and to model general movement in buildings an undirected graph structure would be useful. Also in order to model imposed changes to the route taken during a simulation, links could be assigned in both directions for corridor movement, and the capacity of flow across links set to zero for the 'illegal' direction. The inclusion of two-way links would also cause the graph structure to be heterogeneous, and the algorithm for calculating occupancy figures would become more complex.

*A graph structure is **convergent** if a node can be pointed to from two or more other nodes.*

Each room in the building could not possibly have its own route to the exit, and junctions are inevitable. In the simple example given, blocks 5, 7 and 11 have two arrows entering. Any data structure of this type will be convergent.

*A graph structure is **cyclic** if there is any path in it from a node back to itself, except as a consequence of undirectedness. If a graph structure is not cyclic, then it is **acyclic**.*

---

*Definitions of standard properties are quoted from 'A practical Approach to Data Structures' by Kit Lester.

It is important for this application that the graph structure should not be **cyclic**, since it is assumed that the occupants move out of the building, and do not retrace their steps. The simple example given is **acyclic**. To include the more realistic assumption that occupants may choose their own route out, would give a graph structure which appears to be cyclic. A possible method of computation would be to superimpose multiple acyclic networks, one for each of the optional routes, which would inevitably complicate the calculations.

Theoretically the graph structure may be acyclic and still have two-way flows. In this application there is a contradiction since the two-way flow cannot apply to all links: exits cannot become entrances, and source rooms cannot become exits.

*A graph structure is **disconnected(or partitioned)** if its nodes can be separated into two or more groups such that there is no pointer from any one group to another group. If a structure is not disconnected, then it is **connected**.*

It is natural for a graph structure representing a building to be **partitioned**, so that different parts of the building are served by different exits. The graph structure will be partitioned if the routes out are defined without choice, and that all occupants of a particular room move towards one exit.

5.2.5 The static descriptive model

For modelling emergency evacuation we use a **directed acyclic graph**, and this graph structure is the **static descriptive model** for the evacuation process.

5.3 The design for implementation

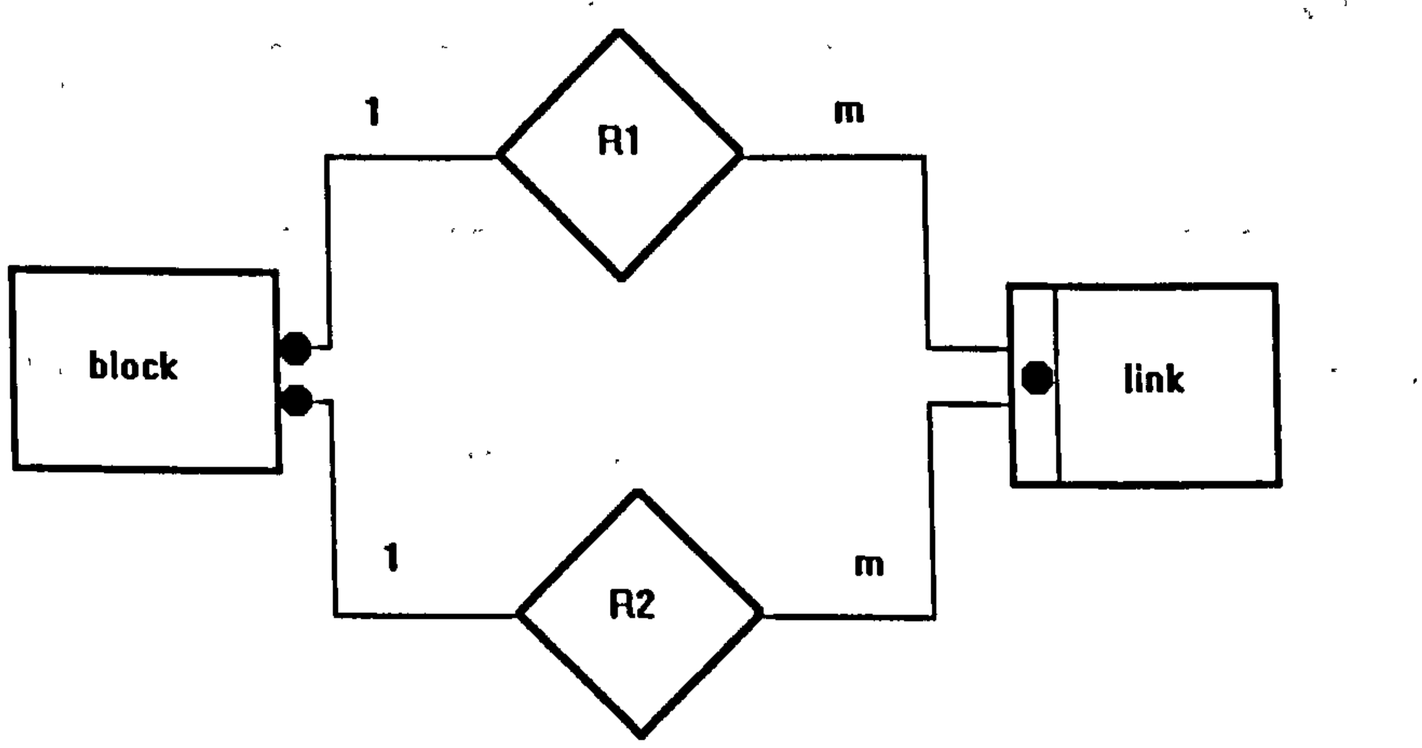
5.3.1 The entity-relationship model for the graph structure

Data analysis of the graph structure is useful to eliminate redundancy in the records, and to check the robustness of the design. The entity types for this structure are blocks and links. The E-R diagram in Diagram 17 shows the relationships between the entities, which are given in Table 2.

Table 2 The relationships between blocks and links

relationship	**R1**	**R2**
description of the relationship	a link arrow leaves block	a link arrow enters block
properties	a block may have several links leaving it (evacuation model has only one link leaving)	a block may have several links entering it
	a link emerges from one and only one block	a link enters one and only one block

Diagram 17 The entity-relationship diagram for the graph structure



Entity-relationship diagram for the graph structure

The full E-R diagram is similar to the one for critical path problems [Hutchings 1990]. The only difference is that a graph structure for the evacuation model has only one link emerging from each block, which converges the routes out to one exit.

Hutchings points out that tables for the relationships R1 and R2 may be eliminated and merged with the link data, so that the tables for the model are:

LINK TABLE (link identity, identity of block before, identity of block following)
BLOCK TABLE (block identity)

It can also be shown that block identity and link identity are determinants for necessary attributes for the graph structure, which indicates that according to the Boyce-Codd rule the data set is a well-normalised set of tables [Howe 1983].

The tables correspond to the data-file which defines the graph structure. Other data supplied is needed for the computation of evacuation times but is not significant to the model structure.

During the computation, the data in the tables is used to assign values to pointer variables, which are attributes of the links and blocks as shown in the Table 3.

Table 3 Attribute pointer variables for blocks and links

Object	Attribute pointer variables	
LINK	BLOCK before	BLOCK following
BLOCK	list of LINKS entering	list of LINKS leaving

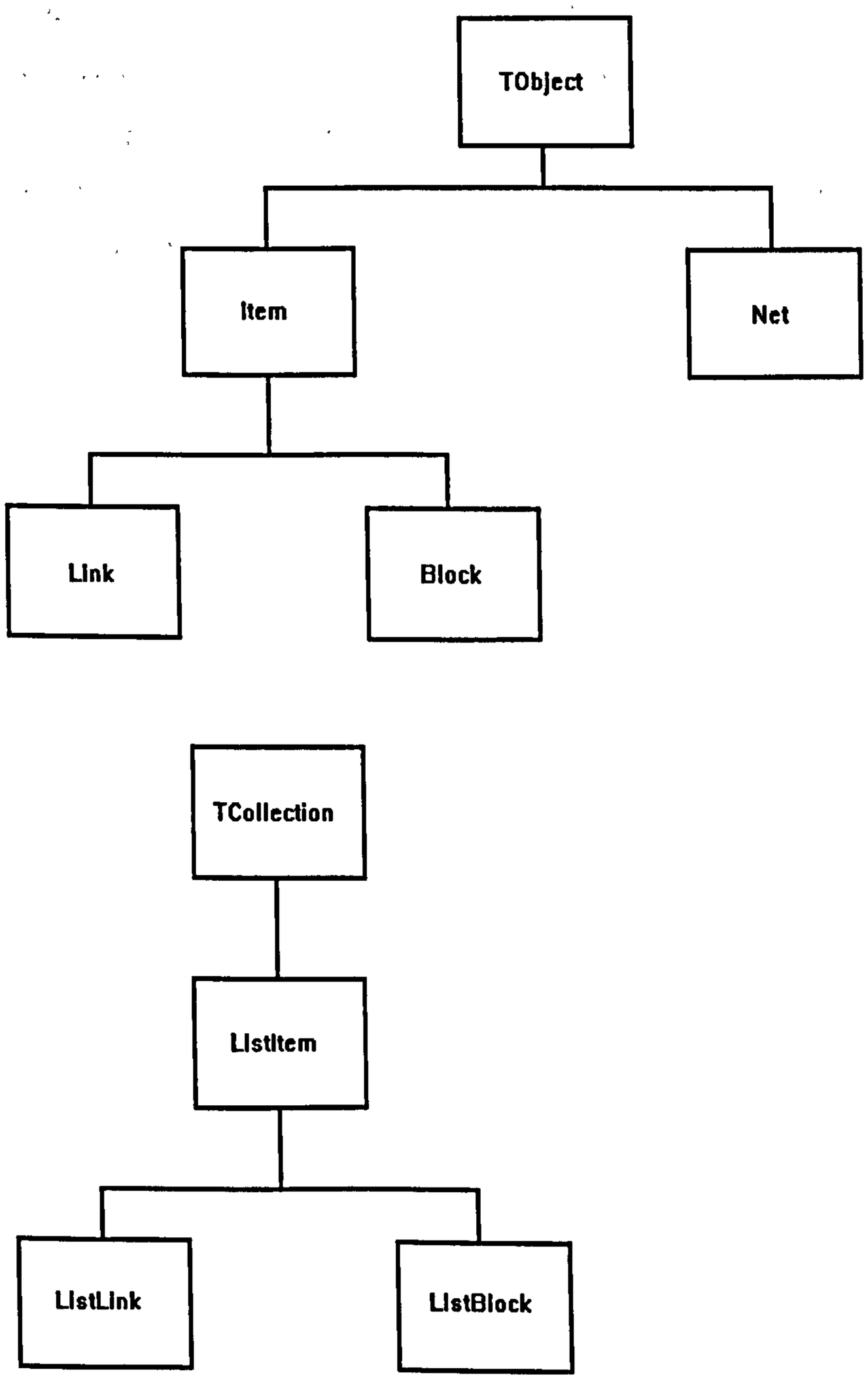
In effect, the tables supplied in the data file are the network, because they supply the mutuality of information within the component objects. Appendix B gives a detailed description of the network structure.

5.3.2 The object hierarchy for the entities

The diagrammatic representation of the object hierarchy used in the evacuation model in Diagram 18 uses,

"part of an extended notation for ER diagrams that indicates that lower boxes represent special cases of the concept in the higher-level box to which they are connected." [McGregor & Sykes p99]

Diagram 18 The object hierarchy for the evacuation model



The properties of inheritance, polymorphism and encapsulation are found useful in the program design which uses Turbo Pascal 6.0 with the object-oriented facilities of Turbo Vision. The Turbo Vision primitive objects, TObject and TCollection provide basic utility methods. The user defined primitive *Item* has descendants *Link* and *Block* which are the basic components of the graph structure. The graph structure itself is represented by the object *Net,* which has field pointers to the lists of *Blocks* and *Links*. The methods which build and check the structure are contained in Turbo Pascal units, and the documentation for these units is given in Appendix B.

The design guidelines for the design of modules [McGregor & Sykes pp124-130] have been largely observed. To comply with the *Design Guideline #8* [McGregor & Sykes p130], the preferred object hierarchy would have retained the *Listblock* object as a root class, and created a subclass to represent the sequencing of the blocks for scanning. Unfortunately Turbo Pascal 6 does not allow multiple inheritance and the method described in section 5.5.2 was adopted.

5.4 The data file for the generic model

5.4.1 Information required for constructing the model

The plan of the building can be used to enumerate the blocks, which may be rooms, sections of corridor, sections of staircase or exits. Rooms are regarded as one block if there is only one door out of that room, but otherwise as two or more blocks according to the number of doors. Sections of corridor are approximately five metres long, but may be smaller if several doors open out upon that section of corridor.

Each block is numbered, and where possible, room numbers are preserved. The maximum **capacity** of each room is given as the statutory maximum capacity. For corridors and stairs the capacity is calculated as the number which can be held at a density of 3.33 persons per square metre or 2.8 square feet per person is calculated. Dimensions of each block are not specifically included.

Links are defined wherever people may pass from one block to another. On a plan of the building adjacent sections of corridor will have the link defined in the direction of flow. The **capacity** of each link is given as the maximum rate of flow per time step. The capacity of the link implies the physical dimensions, i.e. the width of the doorway or corridor, which will limit the number of people who may pass from one block to another in a unit of time.

Movement between adjacent blocks may be possible, but not allowed according to the current rules determining the route out. The data-file must provide information about alternative routes which may be needed when one exit is blocked and the routes out are changed. For this reason, links with possible, but forbidden flows are included in the data-file with negative **capacity**, and a change in the positive/negative value of the attribute **capacity** for any link will result in a different network.

5.4.2 Creation of the data-file

On site observation plus some study of building plans is needed to construct the data-file which then constitutes the model for that particular building. The process is similar to that carried out for the congestion model constructed for the Angel underground station [Weston & Shapiro 1987].

The building lay-out is defined in the form of a graph structure, and the components **blocks** and **links**, are tabulated with essential attributes:

- The data for the **blocks**, once created, can be regarded as fixed.
- The **link** data defines the routes out and may be changed for several reasons.

In this model it is assumed that the route out from each location in the building is determined. One-way flow is assumed, and the structure is partitioned at some point mid-way along each corridor. The link data will depend upon fire-drill policies and can be easily extracted from the building plan. However, in an emergency evacuation, blockages could occur, and the simulation model must be capable of changing the link data to represent possible diversions before continuing with the calculations.

In order to demonstrate the construction of the data-file a small part of a building is modelled..

5.4.3 Constructing a data-file for a simple example

A network diagram is given in section 5.2.2. This is for part of one floor of a building. The numbered blocks, representing areas of space, which may be rooms or segments of corridor, are the **'nodes'** of the graph structure.

The specification data file given in Table 4 on the next page, lists the block data and the link data. Note that the **block** has field parameters for position, which enables an automatic screen representation. The schematic layout, which shows the screen display is given following the data file.

The **block** has field parameters for position, which enables an automatic screen representation.

The screen display as shown in Diagram 19 is a schematic layout which corresponds with the data file given in Table 4.

Table 4 Specification file for a building lay-out

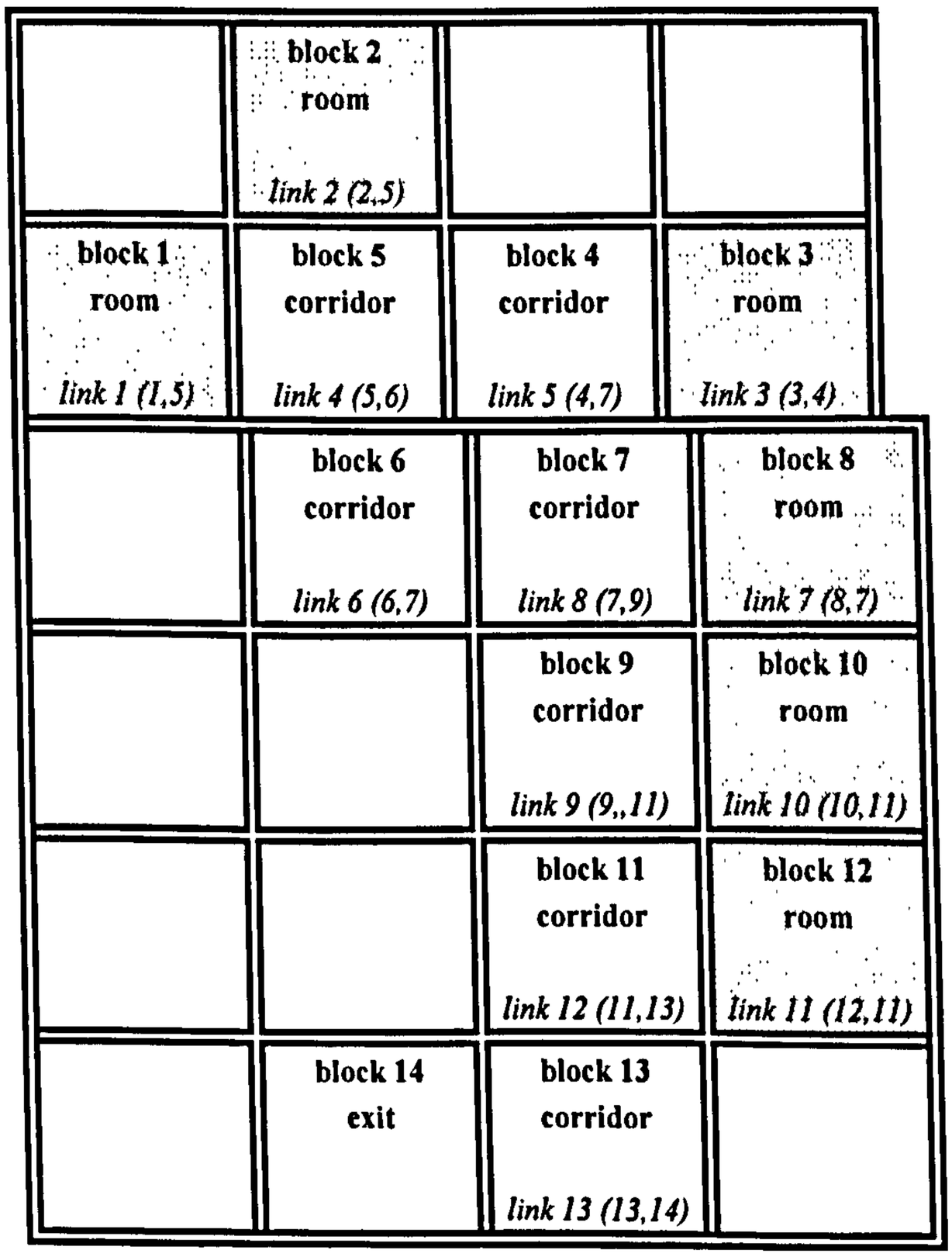
Block data

identity number	maximum capacity	location x coordinate	location y coordinate	type of area	occup-ancy
1	40	1	2	room	13
2	30	2	1	room	30
3	5	4	2	corridor	2
4	10	3	2	corridor	1
5	10	2	2	corridor	0
6	10	2	3	corridor	2
7	10	3	3	corridor	1
8	50	4	3	room	35
9	10	3	4	corridor	0
10	15	4	4	room	6
11	10	3	5	corridor	0
12	15	4	5	room	2
13	10	3	6	corridor	0
14	99999	2	6	exit	0

Link data

identit y number	from block	to block	max. flow
1	1	5	3
2	2	5	3
3	3	4	3
4	5	6	5
5	4	7	5
6	6	7	5
7	8	7	3
8	7	9	5
9	9	11	5
10	10	11	3
11	12	11	3
12	11	13	5
13	13	14	5

Diagram 19 Schematic layout corresponding to the data file



The data file can be verified by an inspection of the screen display of the schematic layout. The link data is difficult to check from the display, but errors would be evident from the summary report on the data structure produced by the program.

The report lists:

- whether each block is a source, an intermediate block, a junction or a sink,
- whether cycles are present in the structure,
- the paths followed starting from each block.

5.5 The driving mechanism for the generic model

The graph structure is the descriptive static model, but the simulation model is **dynamic.**

5.5.1 A control algorithm for simulating the evacuation of the building

The graph structure of interconnected linked lists of blocks and links enables full information to be accessed by each block and link about its immediate surroundings. In the situation where no other entities are involved, and people are represented only as the occupation number for each block, two simplifications are possible.

- The model may be updated in fixed time-steps. The time-step in this case is 5 seconds.

- The simulation can proceed with a systematic scan of the data structure at each time-step, examining each link once only*, working from exit to source.

Thus calculations are made at each time step to determine the state of the building, which, in this model, is the number of people in the various rooms and corridors. Reports can be written to file or shown as a graphical display on the screen.

The model is deterministic. Once the initial state of the building is known the calculated evacuation time is determined; it is assumed that no random events occur. For this reason, in this model, all people move at the same rate and start to move out of the building immediately the fire alarm sounds.

---

* The Congestion Model requires a repeated scan at each time step to cope with two-way flow.

5.5.2 Implementation of the data structure scan

Each block in the graph structure for the evacuation model has only one immediately following block. This makes it possible for occupation figures to be calculated in a pairwise fashion, referring to data for the current and the following block. The number moved out of the current block will be the maximum number which is allowed forward according to the flow rate determined by the link and the space available in the next block.

Provided the blocks are scanned in an order which is strictly from sink to source, each block need be visited only once. There are usually several sinks, i.e. several exits, and the graph structure will be partitioned, but it is not necessary to make separate lists for each exit.

The order for scanning the structure is obtained by allocating each block a number obtained by numbering the blocks using Fulkerson's rule, which is done by performing a repeated check of the list of blocks for the existence of entering links; blocks without entering links are numbered and the links leaving numbered blocks are temporarily removed. This numbering taken in reverse guarantees a strict sink to source sequence.

An early implementation of the model used linked lists to represented the graph structure with nested recursion to generate the required sequence of blocks. A different approach was adopted in the implementation which uses Turbo Vision TCollection objects.

TCollection objects are lists which can function either as linked lists or arrays using their own utility methods. Nested recursion using these objects used calls to ancestor methods which caused problems. Non typed pointer variables may have been one solution, but instead recursion was avoided and the *"flood-wave intuition"*, described by Kit Lestor, was used to walk or visit a structure to create the desired sequential order [Lestor 1990].

The *"flood-wave intuition"* method conducts a walk through the set of objects, by initially assigning the value *'true'* to the Boolean attribute *'dry'* for each object, and then making this attribute *'false'* when the object has been visited. When the scan of all objects is complete the attributes *'dry'* are reset to *'true'* [Appendix B]. The scanning process of all blocks in a sequence may be automatically carried out using recursion but the method described can be more efficient .

5.5.3 Modelling behaviour at junctions

At junctions, where a block has several preceding blocks, there is a conflict of flow. In its simple form the program will perform calculations in the order in sequence as the blocks are presented in the reverse order of the Fulkerson numbering. It will move people according to the maximum capacity of the link and the space available in the destination block. Since the order in which people are moved at each junction remains constant, this will result in one queue through the building being cleared before people from another are allowed through.

The simple version of the program does not model a realistic mixed flow and would not accurately predict the time to escape from a particular part of the building. It usually gives an accurate calculation of the total evacuation time when flow out is uninterrupted, but a more realistic model would be required for interactive experiments.

There are several possible methods of modelling behaviour at junctions at each time step:

- A random choice for priority queue could be taken.
- Priority could be given to each queue in turn.
- The number allowed to enter a junction block could be shared between the competing queues.
- Priority and the number allowed forward could be proportional to the numbers in following blocks.

To implement the last two methods requires a fundamental change to the algorithm. For the third method this involves an additional loop for each junction at each time step. However, to implement one of the first two methods, it is only necessary to change the order in which the blocks are examined.

The first method, which gives a random choice of priority, gives an acceptably realistic model and has been adopted. The method could be improved by attaching probabilities to the queues which represent the relative dominance of flows, but in practice the evacuation times from trial runs for the Eden Grove building, as given in Chapter 6, show no variation. For congested flow in a large building total evacuation time is unaffected by the priority of queues at junctions.

However the simple form of the model does demonstrate the quickest evacuation times possible, and shows that the maximum capacity of the links effectively imposes an upper bound upon the rate of flow out of the building.

Trial runs of the two methods using the test file in section 5.4.3 gave evacuation times 10 seconds faster for the improved method. Close examination of the results, indicates that the simpler algorithm gave priority to corridor movement at the expense of the full rooms 8, 10 and 12, which left some sections of corridor below full capacity.

Trial runs for the Eden Grove building file gave identical evacuation times.

5.5.4 A modified control algorithm

The evacuation process has been modelled using only spatial components and a fixed time step. For problems with several classes of interacting entities a more complex model will be needed.

Further instantiations of the data structure will be needed to represent other entity classes and a three-phase algorithm with a variable time step will be used.

The modified control algorithm is applied to another problem domain in Chapter 9

Chapter 6 Modelling the Movement of People

6.1 Reported observations on flow rates

Reports made of observed movement rates in emergency evacuation in varying circumstances and with different types of people, indicate general agreement. There seems to be an underlying constant mode of behaviour.

6.1.1 Observations on walking speeds

Observations made of walking speeds in non-emergency situations do indicate considerable divergence. Results for walking speed in a shopping mall are quoted by Tregenza , which approximate to a Normal distribution ranging between 0.8 and 1.8 m/sec.[Fruin 1971a/b] A more detailed breakdown indicates that walking speed varies according to the person's age and sex, the activity or situation, and whether they are alone or in groups. An important influence is the corridor capacity and the congestion, for instance where mean density is 1.4 P/m^2 walking speeds are lower and less variable than when mean density is 0.3 P/m^2.

"Under free flow conditions the range of speed in any group may extend, typically, from 0.6 m/s below the mean to 0.6 m/s above. With crowding and all pedestrians moving in one direction the range is very small." [Tregenza 1976]

For this reason, rates of flow are always quoted for a given level of congestion. Table 5 gives a range of levels of congestion feature in the literature using a variety of units.

Table 5 Congestion levels

Conditions	Density	Area per person	
	persons per sq metre	square metres	square feet
Free flow	0.434	2.3	25
	0.6	1.7	18.3
Full	1.5	0.67	7
	2.0	0.5	5.4
Congested	3.57	0.28	3
	4	0.25	2.7
Packed	5.5	0.18	2

Under the conditions of emergency evacuation, walking speeds tend to converge, slower people speed up and faster ones slow down, and Bryan recommends the assumption of a common speed of 1.26 m/sec on level ground, which is quoted as the average speed for an older woman in free flow conditions [Bryan, Tregenza 1976].

The speeds assumed for level ground are given in Table 6

Table 6 Walking speeds

Density P/m²	Walking speed		
0.434	1.26 m/sec	76 m/min	3 mph
1.5	0.88 m/sec	53 m/min	2 mph
2.0	0.733 m/sec	44 m/min	1.64 mph
3.57	0.38 m/sec	23 m/min	0.86 mph
5.0	stationary		

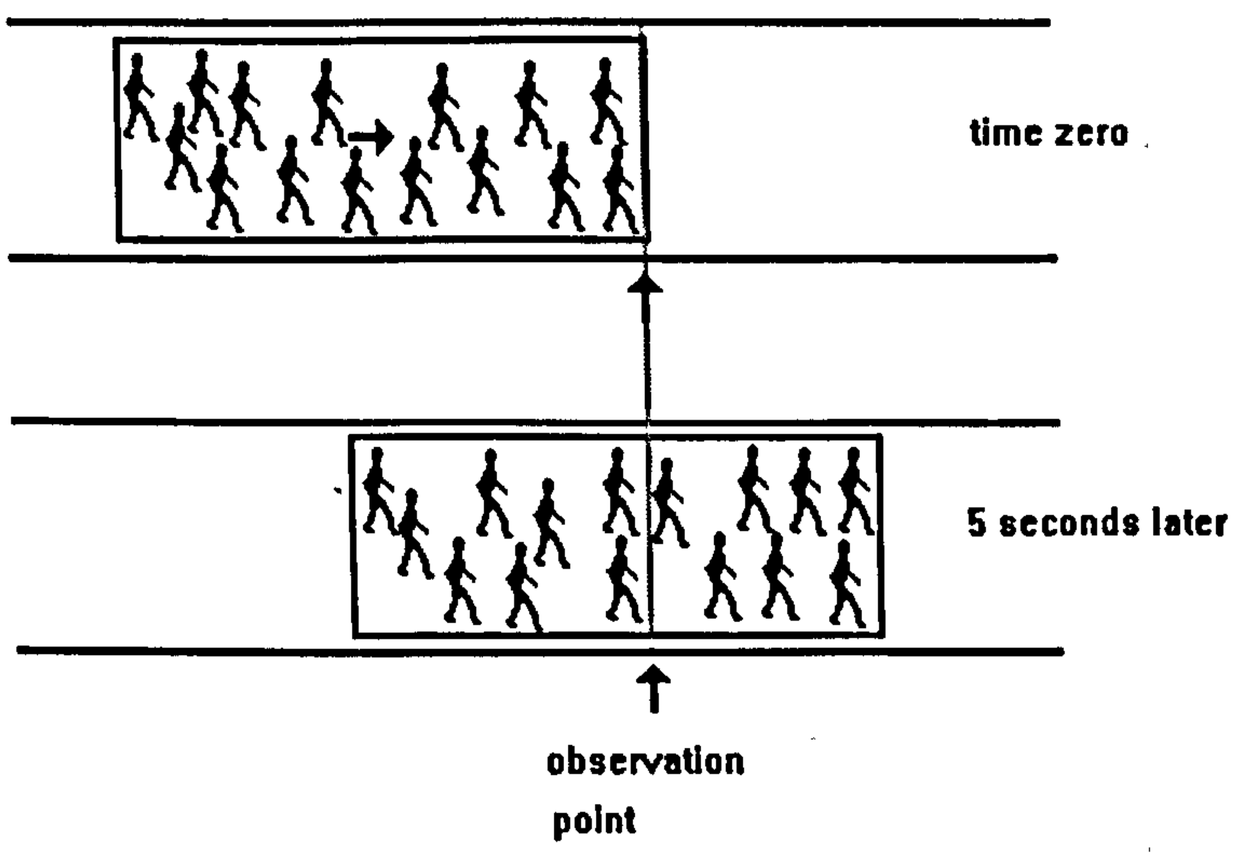
6.1.2 Reported rates of flow in emergency egress

A more useful parameter for a model for emergency egress is the number of people passing an observation point per unit time, which is also easy to observe with a fair degree of accuracy. The walking speeds given in the table above can be used to calculate rates of flow along passageways of given width to check consistency.

"A study of footways indicates that for passageways over 4 ft (1.2 m) wide, flow rates are directly proportional to width." [Bryan]

Consider a segment of corridor 1 metre wide and 10 metres long. with a density of 1.5 P/m^2 which is shown in Diagram 20. This segment can be assumed to hold 15 people.

Diagram 20 Pedestrian rate of flow



If these people are moving forward at the uniform speed 0.88 m/sec, in a time step of 5 seconds each person has moved forward 4.4 m, which is 44% of the length of this segment of corridor. An onlooker placed at the observation point marked, will be expected to count 7 people pass in 5 seconds, and this is equivalent to a rate of flow of 7 people in 5 secs, or approximately 80 people per minute.

This calculation can be repeated for density values 2. and 3.57, giving the rates of flow 88 and 86 people per minute respectively, which confirms that the rate of flow is fairly stable for a range of values of density between 1.5 and 4.

The calculated values for rate of flow agree well with those given in the London Transport Board Research Report [LTB 1958], which gives the rates of flow for a unit corridor width of 0.3 m (1 ft). The estimates in the fourth column of Table 7 were obtained from observations made at a boys' school and a Training school. [Weston & Marshall 1973]

Table 7 Rates of flow under congestion (density 3.57 P/m²)

	Number passing per minute		
	LTB estimates		Weston & Marshall estimates
Corridor width Location	0.30 m or 1 ft	1 m (LTB)	1 m
level along corridor	27	90	90
descending stairs	21	70	70
ascending stairs	19	63	62

Both the LTB report and Weston & Marshall claim that the maximum flow rates are attained when the density is 3.57 P/m² (3 square feet per person), but using empirical observations, Pauls reports that the rate of flow downstairs attains a maximum when the density has a value of about 2.17 or 2.7 P/m². [Pauls 1977].

Bryan quotes a table of speeds and rates of flows on stairs corresponding to congestion levels. [Galbreath 1968] Standard widths are quoted as 22 inches (559 mm) for an exit unit width. Taking account of the different unit width there is good agreement, and the table also gives a maximum discharge rate from the stairs when the density has the value 3.57 P/m² (3 sq. ft per person). This discharge rate is 45 persons per unit exit width per minute, and since most staircases would be two units wide, this is equivalent to a discharge rate of 90 persons per minute.

6.1.3 A model for conflicting cross flows

Standard flow rates have been established for evacuation conditions where people are moving in the same direction. Movement in opposite directions, in parallel streams has been observed to show no reduction in combined flow rates, [Weston & Marshall 1973] [Bryan]. The situation where some people are moving across the main flow is more complex, and frequently occurs in railway stations. A measure is needed for the maximum combined rates of flow where minor flows cross a major one.

From photographic records taken at a busy intersection of flows at a mainline railway station, data was compiled of crossing times and the corresponding concentrations of people. This data was used to formulate a model for crossing times in terms of concentrations of people crossing in the two directions. [Weston & Marshall 1973] A simplified version of the model produced is

$$T = 0.79 + 0.03(d - 0.56) + 0.47(d - 0.56)^2$$

- where T denotes the time in seconds to cross a distance of one metre in the direction of major flow,
- d denotes the density given as the number of persons per square metre, which is the combined total of those moving in conflicting directions.

Note that the model gives pedestrian speeds of:

1.27 m/sec for d = 0.56 corresponding to free flow,

0.20 m/sec for d = 3.56 corresponding to congestion,

which demonstrates the inverse relationship between pedestrian speed and density.

A linear model was obtained for the times of minor direction flow, and in general, the pedestrians crossing the major flow had faster speeds for all levels of congestion.

The purpose of the study was to investigate whether the usual maximum flow rate of 90 people per metre width per minute could be maintained with conflicting flows. The conclusion was that combined flows would be reduced to an estimated 70 people per metre width per minute.

6.2 Observations made at the University of North London

Observations were made in the Eden Grove building at the University of North London of movement times in different situations under crowded conditions. The measures were given as the numbers expected to move from one location to the next in one minute in congested flow.

Table 8 Observations made at the University of North London

Type of movement	Width of passage way	Observed numbers	Reported in other surveys
		Number of people/minute	
from one section of corridor to another	1.5 m	60	90 per metre width [LTB 1958]
from a room into a corridor	0.84 m	36	40-60 through 0.9 m wide swing door [Fruin 1971]
from a corridor onto the stairs	1.3 m	60	90
from one section of staircase to another	1.3 m	48	90
from the staircase out of an exit	1.3 m	48	45 per standard width 0.56 m [Galbreath]

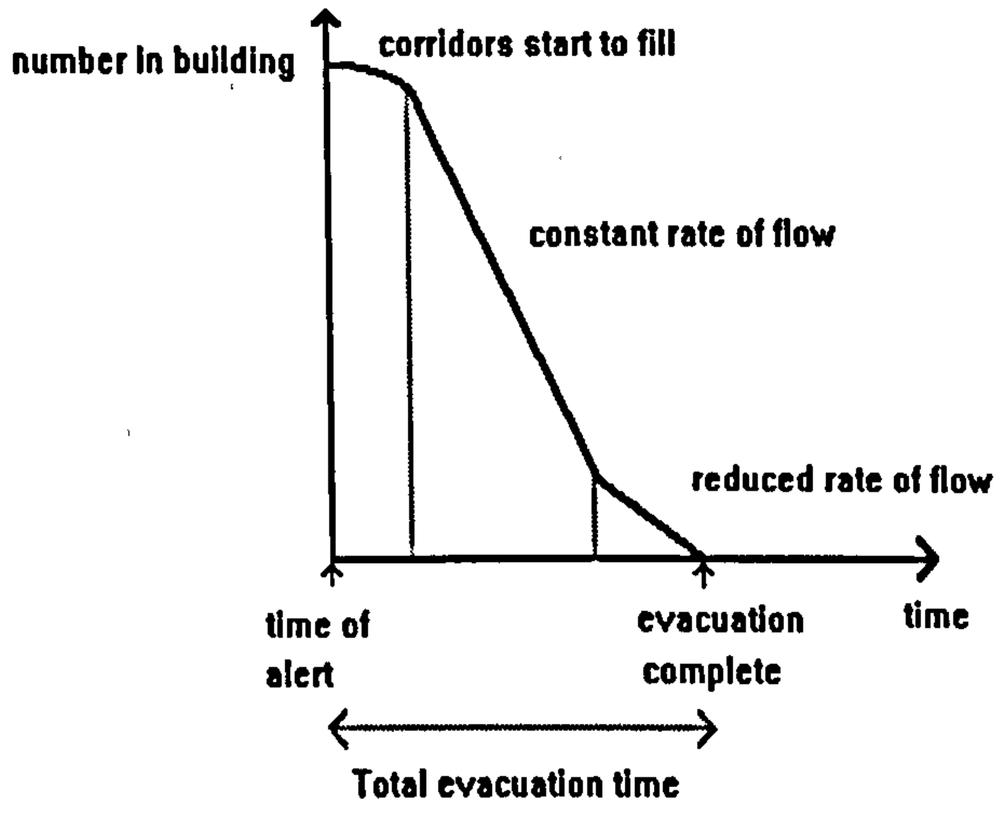
The empirical observations, which are much lower than the estimates given in the fourth column quoted from LTB, Galbreath and Fruin (section 6.1.2. Table 7), may have been made in less congested conditions.

6.3 Estimating building evacuation time by other means

6.3.1 The varying rate of evacuation

The rate at which people leave a building when flows are uninterrupted is shown in Diagram 21

Diagram 21 The number of occupants plotted against time



The evacuation process may be examined in three stages.

- During the initial period, following the alert, the rate of egress is slow but increasing. At this stage corridors will not be full; some occupants will be remote from the exit and others may take time to respond. The actual rate of egress at this stage is difficult to estimate since it depends upon the distribution of people at the time of the alert and the behavioural response of occupants.

- During the second stage, when the corridors are full, the flow out of the building can be assumed to be constant. This rate will be limited, either by the widths of the exits, or by the corridors within the building, whichever is the smaller.

- Most of the occupants will have been evacuated after the second stage. The rate of egress will decrease when all people using one of the exits have left, and this indicates that efficient evacuation procedures should ensure that numbers choosing an escape route should be in proportion to its capacity. The time taken for stragglers to evacuate cannot be estimated, but in practice could be controlled by marshalls.

6.3.2 Galbreath's formula for evacuation times

A formula for total evacuation time is given by Galbreath, [Galbreath 1969], which is explained in the Handbook of Fire Protection [Bryan].

The formula gives an estimate which is based on the constant maximum rate of egress shown on the graph with some correction for the corridor filling up time. It assumes an even distribution over the exits which in effect underestimates evacuation time. Occupant response times are also neglected.

The parameters needed are:

N	the number of persons in the building above the first floor.
n	the number of persons who can stand on the stairs at 3 sq. ft per person, or the number of persons on the floor whichever is less.
r	the rate of discharge of the stairs in persons per unit width per min.
u	the number of 22in. exit units of stair width.
T	the evacuation time to the nearest minute.

The formula $T = \frac{N + n}{r * u}$ assumes a uniform distribution of people in the building, with nearby staircases and no remote rooms.

Applying this formula to Eden Grove:

The total number in the building is 380, the sum (110+110+80+80)

$$N = 380$$

The staircases between floors,(including landings) hold 50 people on the east and 42 on the west. There are 4 flights of stairs, therefore 368 people in total can stand on the stairs and $368 < 380$

$$n = 368$$

r * u represents the maximum rate at which people could leave the stairs.
For the stated density, 3 sq. ft per person, the paper gives **r =45** persons per unit width per min. (The simulation program uses 36 people per min, i.e. 3 per 5 sec interval)

Using these figures T = 748/180 = 4 minutes 9 secs.

6.3.3 A justification of Galbreath's formula

The formula can be justified by estimating the effective flow of people out of the building. Table 9 gives the effective numbers on each floor.

Table 9 The estimated effective flow of people by floor

Source	occupants of floor	stairs 4th to 3rd	stairs 3rd to 2nd	stairs 2nd to 1st	stairs 1st to ground
4th floor	110	92	+(1/2)92	+(1/3)92	+(1/4)92
3rd floor	110		+(1/2)92	+(1/3)92	+(1/4)92
2nd floor	80			+(1/3)92	+(1/4)92
1st floor	80				+(1/4)92

The effective total flow of people from time zero is $N + 4*92$, and the evacuation time can be calculated by dividing this total by the maximum rate of egress, which agrees with the formula.

If the numbers directed to each exit are known, the formula could be applied to the building for each exit separately.

For instance if 250 of the 380 occupants chose to use one of the exits (the east exit), the evacuation time would be estimated as

$$T = \frac{250 + 4 * 50}{r * 2}$$

which gives an evacuation time of 5 minutes, and this is 51 seconds longer than the previous estimate.

6.3.4 Usefulness of the formula

The formula is useful because it gives an estimate of the best possible evacuation time to be expected, and the only data required is the width of exit routes and the numbers of people on each floor. This is an example of theoretical analysis giving an approximate estimate, as suggested in section 2.2.3. It appears to be highly sensitive to the assumed figure for the number of people who can stand on the stairs, and is known to give a crude estimate for the evacuation time.

The program can produce more accurate estimates since it represents the distances of rooms on each floor from the stairs and can model changes to the routes taken.

6.4 Conclusions

Evacuation behaviour appears to be standard, and the parameters needed for the model can be obtained by reference to the measurements of the corridors, staircases and exits of the building. Response times of the occupants remain unpredictable, but can be modelled using stochastic sampling from collated observations.

The parameters derived in this chapter are sufficiently reliable to be used in the practical trial of the evacuation model as described in Chapter 7.

Chapter 7 A Practical Trial of the Generic Model

7.1 Application of the generic model to the Eden Grove building

The Eden Grove building at the University of North London was chosen as a trial for the model. It is relatively small, has a simple layout and only two exits.

7.1.1 The Eden Grove data-file

The schematic layout, in Table 10, is equivalent to a sketch plan and shows how the building has been subdivided into blocks, which are rooms (or parts of rooms), sections of corridor and flights of stairs. The main entrance foyer at the east of the building is well-known to people, and there is a tendency for occupants to favour that exit even in emergency. The doors to the west exit are barred but can be opened from the inside in case of emergency.

Table 10 does not show the routes out, which can be changed by altering the data-file.

In the initial run, for the sake of simplicity, it has been assumed that people leave the building by the shortest route to one of the two exits referred to as *west* and *east*, and that all occupants of a room take the same route out. This, in effect, partitions the data structure approximately in the middle of each corridor.

Table 10 Schematic layout for the Eden Grove building

floor level	west stairs											east stairs	
	545	413	414			418	419	420	421	422	423	551	
fourth	9		550	549	548	547	546	544	543	542	541	19	
	8	411	410	408	407	405	404	1404	403	402	401	18	
	535		310	312		314	1314	1315	315	1316	316	530	
third	7		540	539	538	537	536	534	533	532	531	17	
	6		307	306	305	304	1304	1303	303	302	301	16	
	525					212		1214	214	216	218	520	219
second	5			529	528	527		524	523	522	521	15	
	4		208		206	205		204	203	202	201	14	
	515				112	113		1114	114	115	116	510	118
first	3		519		518	517		514	513	512	511	13	
	2		108	107	106	105		104	103	102	101	12	
west exit	1											11	east exit

Note that actual room numbers are used as block identities, but some rooms appear as two blocks. Corridor segments have been assigned numbers starting with the digit '5'.

Copies of the building plans are given in Appendix A.

The graph structure is defined for the simulation program in the data file.

Blocks are listed with:

- identity number
- maximum capacity
- three plan references (two screen co-ordinates and floor level)
- type of block (room/corridor/stairs/exit),

and links are listed with:

- identity number
- the block before
- the block after
- whether a separating wall (1:door, 0:no separating wall)
- maximum capacity
- type of link (door corridor stairs).

The occupancy data at the time of the alert, consists of a list of block identities and the numbers occupying that block.

In the implementation, the attributes, which for blocks denote the three plan references, and for links denote the identities of blocks before and after and the presence or absence of a separating wall, are stored using the same three fields, which allows some saving in the parameters stored for these objects.[McGregor & Sykes p. 123]

7.1.2 Measures assumed for the model

The data for the blocks is well-defined and can be confirmed by taking direct measurements of the building. Having divided the corridor into sections, of approximately equal area, each corridor block measures approximately 5 metres in length, and has a minimum width of 1.5 metres.

The model moves people forward by one block each time step, unless there is an obstruction or no space to move into, so the choice of time step and size of block are critical. The time step is chosen to be 5 seconds, but this can be adjusted during model calibration.

7.1.2.1 Maximum flow rates

The model required maximum rates of flow for the links between blocks, and observations were made in the Eden Grove building of movement times in different situations under crowded conditions.

The figures given are the numbers of people expected to move from one location to the next in one time step of 5 seconds. Table 11 gives the empirical and adjusted measures for the maximum rates of flow in the model:

Table 11 Maximum rates of flow assumed in the model

Type of movement	Number of people / time step	
	Observed	Adjusted
from one section of corridor to another	5	7.5
from a room into a corridor	3	4
from a corridor onto the stairs	5	7.5
from one section of staircase to another	4	6
from the staircase out of an exit (double standard width)	4	6

Maximum flow rates quoted in Chapter 6 section 6.1.2 are higher, which could be explained by the university observations being made under less hurried circumstances. The results in the literature have been taken into account and adjusted measures are given in the table.

The effect of using the adjusted measures for maximum rates of flow in the model is examined, by calculating the effect upon people's speed of movement for varying levels of congestion. Table 12 gives a range of levels of congestion, with the corresponding number of people that could expect to move into the next block if there was room, which is limited by the maximum rate of flow out of a block, (an integer value 7 or 8). The estimates of average speed are compared with reported speeds given in 6.1.1.

Table 12 Effective average speeds in the model

Density P/m^2	Estimated number in block	Number expected to enter next block in one time step	Estimated average speed	Reported average speeds
0.5	4	4	1 m/sec	1.26 m/sec
1.5	11.25	7 or 8	0.57 or 0.71 m/sec	0.88 m/sec
2.0	15	7 or 8	0.47 or 0.53 m/sec	0.733 m/sec
3	22.5	7 or 8	0.31 or 0.35m/sec	0.51 m/sec
3.57	26.77	7 or 8	0.26 or 0.30m/sec	0.38 m/sec

The calculations reveal that, at low densities, the model underestimates the movement speeds of individuals and that better estimates are given when the maximum flow rate is 8. This figure is also justified by the corridor width being greater than one metre.

However the overall rate of flow is modelled correctly, and the effect at low densities will be to overestimate evacuation time. A more serious omission is lack of information about response times.

7.1.2.2 Assessment of room occupancy

Counts were made, room by room, to obtain occupancy data for Eden Grove. Rough estimates, made on the basis of these counts, have been used in the test runs of the simulation program. Table 13 gives the numbers assumed to be in the building.

Table 13 Occupancy data

Floor of building	Observed number	Revised estimate
4th	110	108
3rd	110	144
2nd	80	128
1st	80	91
all	380	471

In the absence of reliable forecasts of numbers present in the building at any one time, it is recommended that the program is run with near full capacity figures and with varying distributions of occupancy, both of which will significantly affect total evacuation time.

7.2 Validation of the model

7.2.1 The use of fire drills

The major parameters of the model are peoples' movement rates and the initial occupancy figures. Since both of these are subject to error, it is interesting to examine how sensitive evacuation time is to the values assumed, and the validation process is most important. Fire-drills provide an opportunity to observe both total evacuation time and total occupancy for model validation, but they may give a distorted prediction of performance in a real emergency.

There are two main factors. Firstly, it is inevitable that some people are fore-warned of the likelihood of a fire-drill, and the numbers present may be reduced. Secondly, for reasons of caution, disabled people will be specifically catered for in a planned fashion, which will be difficult to do as efficiently in a real emergency evacuation. During fire-drills the first factor gives us a reduced number of people present, and the second gives us a faster evacuation time.

7.2.2 A trial of the model for the Eden Grove building

Data is available for the Eden Grove building of the University of North London, and trial runs of the model have been carried out giving evacuation times which are in close agreement with fire-drill practice times.

Table 14 Summary results for evacuation times

Fire-drill date	number evacuated	
1989	unknown	4 mins. 45 secs.
1990	unknown	4 mins. 30 secs.
1991	unknown	4 mins. 15 secs.
Simulation model		
run 2	376 people (182 west & 194 east)	3 mins.
run 3	477 people (230 west & 247 east)	3 mins. 45 secs.
run 4	477 people (165 west & 312 east)	4 mins. 40 mins.
Formula(Galbreath)*	380 people (190 west & 190 east)	4 mins. 9 secs.
	380 people (130 west & 250 east)	5 mins.

The table does not include the evacuation time for a fire-drill in 1992 which was apparently 7 minutes, an unacceptably long time, due to slow response of some occupants.

The simulation model gives the shortest evacuation time when the numbers using each exit are balanced. Runs 2 and 3 indicate the effect of too many people using the east exit, verified by Galbreath's formula. It is likely that in the fire-drills, more people used the east exit, and this would be the slowest to clear.

Overall estimates of total evacuation times agree quite well with records made during fire-drills, but, for the reasons given above, this may not be adequate.

It is interesting to note recorded observed evacuation times of 4 mins. 30 secs. and 5 mins. evacuation times from 7 storey buildings quoted by Galbreath, but these are not comparable, since these buildings have fire-protected lifts which are used for evacuation.

*Galbreath's formula is explained in Chapter 6, section 6.3.2

7.2.3 Sensitivity analysis

An examination has been made of how sensitive the model is to certain parameters which are difficult to measure precisely. The parameters which are in doubt concern the maximum congestion which would arise in corridors and stairs, and the maximum flow rates in corridors and staircases.

A maximum density of 3 P/m² is assumed, which determines the maximum capacities of blocks and the maximum flow rates from one block to the next. The affects of assuming a higher density of 3.57 P/m² are shown in Table 15.

Table 15 Model parameters for different levels of congestion

Density Parameter in model	3.0 P/m²	3.57 P/m²
Maximum capacity of corridor block	25 people	27 people
Maximum capacity of 3 staircase blocks		
East staircase	50 people	60 people
West staircase	42 people	50 people
Maximum rate of flow		
From one corridor block to the next	7.5 people/5 secs.	9 people/5 secs.
From one staircase block to the next	6 people/5 secs.	7.5 people/5 secs.

The evacuation model was run with different values for the maximum flow parameters for the staircase and corridor blocks. The results are given in Table 16.

Table 16 Model results using different parameters

Run number	Number of occupants			Maximum flow rate		Evacuation Time*
	Total	West	East	along corridors	down stairs and out of exit	
1	376	182	194	5	4	4 mins. 15 secs.
2	376	182	194	7	6	3 mins.
3	477	230	247	7	6	3 mins. 45 secs.
4	477	165	312	7	6	4 mins. 40 secs.
5	477	230	247	8	6	3 mins. 45 secs.
6	477	230	247	7	7	3 mins. 20 secs.
7	477	230	247	8	7	3 mins. 20 secs.

The results show that the evacuation time is directly proportional to the number of occupants and to the maximum flow rate from the stairs and out of the exit. Increasing the rate of flow along corridors alone does not affect evacuation time.

Thus the model demonstrates what is already well known; that the capacity of the main exits is the major factor in determining the time required to evacuate a building. It is important to obtain a good estimate for the maximum rate of flow downstairs and out of the exits, since evacuation time is sensitive to this parameter.

Since the model will also be used to estimate evacuation time for situations where some rooms are temporarily blocked, or where routes are changed, performance is likely to be sensitive to rates of flow within the building, but this is not so easily demonstrated.

* Note that both versions of the evacuation model were used and the same results were obtained using the algorithm which gave random priority at junctions.

7.2.4 Further monitoring work

More observations are needed to assess the difficulties of carrying non-walking people out of the building, and to what extent this would obstruct other people's movement along corridors and down stairs. There is a case for attempting to use the equipment in a mock up partial evacuation.

Continued monitoring of the number of people present in Eden Grove should present no problems.

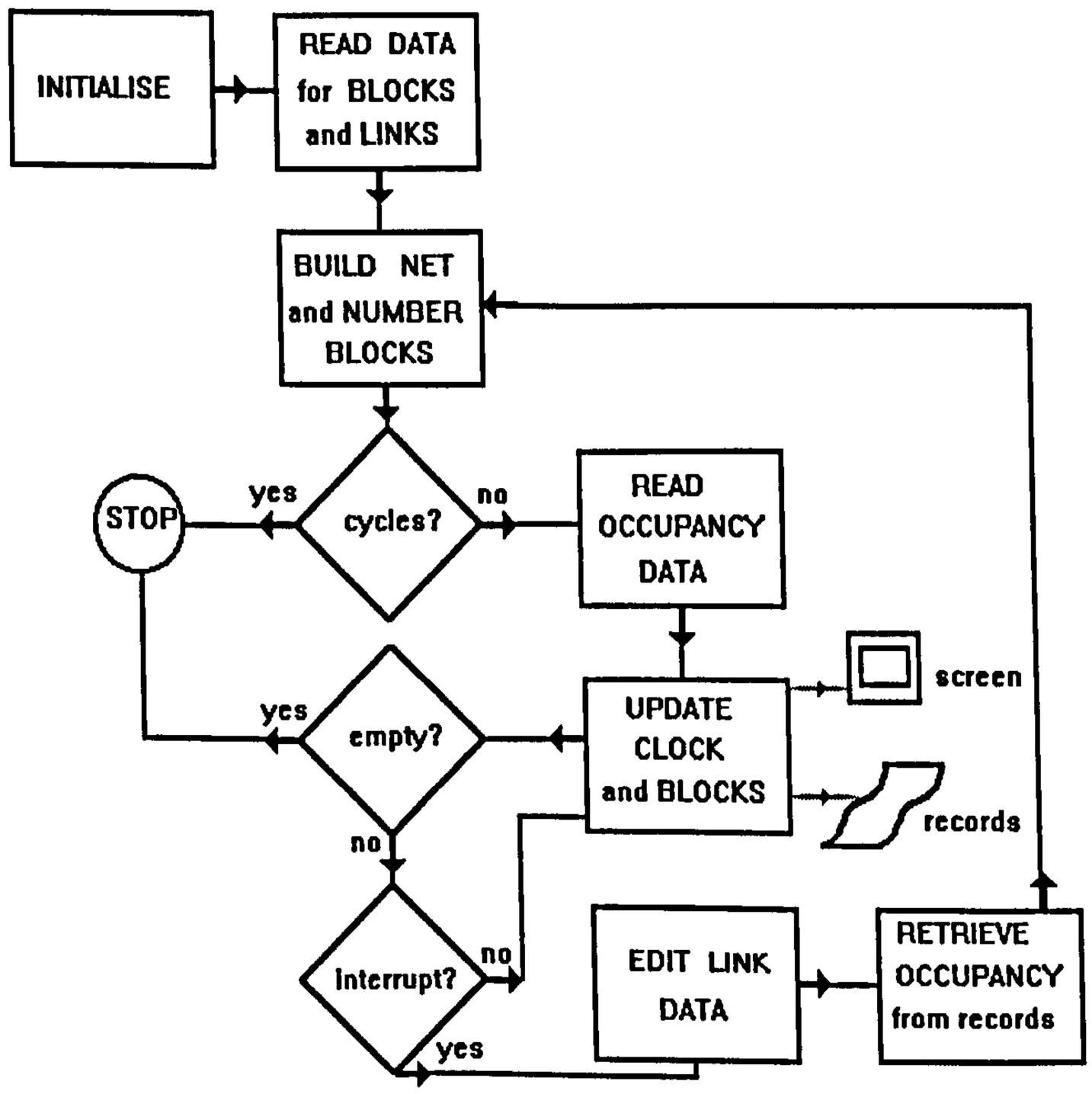
7.2.5 Experiments upon the model

It is envisaged that the model would be useful for 'what-if' modelling. The stochastic element is exogenous to the model itself, and should be introduced by the user who is familiar with the building and would have an interest in the effects of various changes.

Experiments upon the model would require records and predictions concerning the numbers occupying the building, the distribution of these people throughout the building and the recommended manner of egress. The experimenter would run the model under conditions which test the robustness of the evacuation guidelines, and for the benefit of the usual occupants of the building, the model could be run to demonstrate the effects of delayed response or of ignoring posted exit instructions.

Interactive use of the model would follow the algorithm shown in Diagram 22.

Diagram 22 Control algorithm for the interactive model



7.3 Evaluation of the usefulness of the generic model

From data collected during fire-drills it should be quite easy to identify how many people used each fire-exit. More precise information about the behaviour of occupants is more difficult to find, but this is not essential. A more important objective is to demonstrate the effects of the possible different responses from those in the building when the fire alarm is heard. In other words, the model could be used as a training aid, and could demonstrate which was the safest response in a particular set of circumstances.

For instance the model can usefully demonstrate the consequences of many people leaving the building by the least direct route. It could also demonstrate the effect of one room of people delaying their departure. There are many other possibilities.

Chapter 8 Hospital Design and Nurse Activity

8.1 Ward layout and nurse staffing

The relationship between ward layout and nurse staffing was identified by Florence Nightingale, who gave strong directives to architects on ward design.

8.1.1 The 'Nightingale' ward

Florence Nightingale [Nightingale 1863] summarised her views in '*four essential points of construction as regards nursing and discipline, in which hospitals are generally deficient, from want of due consideration on the part of the architects*', these were, using her own terminology:

1. Economy of attendance,
2. Ease of supervision,
3. Distribution of the sick in convenient numbers for attendance,
4. Position of nurses rooms.

In point 3 she refers to her preference for forty patients in one ward where they can be properly cared for than dispersed in four wards where more attendants would be needed. The nurses room referred to in point 4 would have been the bed sitting room for the head nurse, but is now the office or staff base. She wanted this placed by the ward so that the whole ward could be viewed easily day and night.

These precepts remain valid, although her objections to small wards have now been countered by moderns means of ventilation and improved service facilities provided close at hand. There is also the modern emphasis on the patient's rights to privacy which favours the accepted practice of drawing curtains round each bed bay during treatment and the option of small rooms, both of which interrupt the free view of all patients.

The 'Nightingale' open style ward is now one of many ward layouts of assorted shapes and sizes. It is common for a ward unit to consist of rooms of varying sizes, but there have been demands for higher nurse staffing levels for the divided wards, and planners and managers have sought means of evaluating the staffing requirements of the new designs. Attempts to establish measures for the effect of ward layout upon staffing requirements have been confused by adaptations in nurse management and the difficulty of establishing matched working conditions for comparison.

8.1.2 Reported studies on ward staffing

A review of the literature concerning Ward Layout and Nurse Staffing was undertaken by Jean Walker of the Medical Architecture Research Unit, at the Polytechnic of North London in 1981 [Walker 1981]. This is a comprehensive review of surveys measuring different aspects of ward layout and ward function carried out independently in Great Britain and the United States of America. A few of the surveys produced quantitative data which could be used for objective analysis.

A survey carried out by Wessex Regional Health authority [Pace & Grimshaw 1978], produced data in which nurse staffing allocations were recorded for wards categorised by layout. This data was later made available to a student at the Polytechnic of North London, who examined whether the ease of visibility of patients from the staff base for each ward was associated with the nurse staffing level assigned to that ward [Vimalasagarem 1979].

A DHSS study in 1968 was set up to investigate the variability of UK nurse staffing levels and to identify influential factors [Best, Bransby, Cornish & Simpson 1968].

A detailed work study approach was used by Wessex Regional Hospital Board in 1969, using a variety of ward layouts [Wessex R.H.B. 1969], which recommended higher staffing levels for the modified Nuffield design wards.

A comparative study of staffing implications of different ward layouts in Wycombe General Hospital and Princess Margaret Hospital Swindon [Oxford R.H.B. 1970], was an ambitious study, but it was inconclusive because of the multiplicity of factors.

A hospital authority in USA set up a new hospital in 1966, Rochester Methodist Hospital, as a laboratory to study alternative designs of nursing units, hospital systems and organisation [Trites, Galbraith, Sturdavant & Leckwart 1969]. However even this ideal situation, with standardised facilities and management could not provide adequate matched conditions for classical experimental designs, and the reported studies used methods of multiple linear regression [Aydellotte 1973].

The consensus resulting from the many surveys was that ward function as evident in nurse effectiveness appeared to be associated with ward layout, but that the nature of the relationship was not clear.

8.1.3 Measures for staffing level requirements

There have been attempts to quantify the spatially related aspects of ward function and the ward attributes which help or hinder a nurse in caring for the patients have been identified.

Quantifiable attributes of a ward which feature in surveys include:

- travel distances between staff base and patient's rooms, and nurse travel times during normal duties,
- the proportion of beds which can be seen from the staff base.

Visibility is thought to be an important feature, and this parameter was included in the Wessex RHA survey [Pace & Grimshaw 1978] and the Methodist Hospital studies [Sturdavant 1960, Huseby 1969, Trites 1969]. The latter also included a calculation of the percentage of nurse working time for which patients were visible.

The 'Aberdeen formula' computes the weekly nursing work load mainly on case load and case mix, but it includes a correction factor using parameters intended to typify ward layout features [Crompton, Mitchell & Cameron 1976]. Each parameter is converted to an index which contributes to a final score. The parameters include:

- the ratio of wards or rooms to patients,
- the distance in yards of a completed ward round, ranging from 125 yards to 975 yards,
- eight further parameters which are the mid-range distances of beds from eight locations of facilities required during normal nursing routines.

The final score proves to be unrealistically insensitive to the ratio of patients to rooms, and is generally discredited.

The 'Yale Traffic Index' was devised to measure the functional efficiency of a ward layout, based on the most frequently occurring journeys required during nursing routines neglecting travel within rooms [Pelletier & Thompson 1960]. Several modifications were made to the Yale Traffic Index. Lippert produced more sophisticated measures, differentiating between wards of different types using weighted scores based on observed journey frequency [Lippert 1971].

For management purposes it may be necessary to have an absolute score for the staffing levels required for a particular ward layout. For design purposes it is preferable to clarify the relationship between ward layout and functionality. In order to do this a nurse activity analysis was set up and this is described in section 8.1.4.

8.1.4 A nurse activity analysis

From the review outlined in section 8.1.2, it was evident that ward function was associated with ward layout and that an examination of the relationship was obscured by the multiplicity of factors. A comprehensive study was undertaken, on open and divided wards in three different hospitals, matched for case-load and management [Seelye, Ward & Walker 1981], in which direct observations were made of nurse activity using nurse trails during normal routine duties.

Objective measures of nurse effectiveness were required and the measures selected were nurse-patient contact time and travel time. The nurse-patient contact time was variously defined as: direct contact, hear and see contact, hear only contact and see only contact, each being calculated as a proportion of total work-time. Travel was recorded as travel distance in metres and as travel time as a percentage of total work time. Data was also collected on the physical characteristics, facilities available, support services and nurse management for each ward.

The analysis was carried out using cross tabulation and statistical measures of correlation to examine the association of the performance measures and a selection of ward attributes. The statistical survey package SPSS was used. It was found that travel times were similar for different ward layout, but that contact time was generally better in open wards where nurses were able to observe patients while moving from one task to another. However further analysis carried out on the same data, using generalised linear models, confirmed that nurse activity is affected by numerous factors, and that layout is not the most significant of these [Seelye 1982, Gilchrist 1982].

8.2 Hospital activity and accommodation patterns.

8.2.1 Guidelines for hospital design

The role of quantitative methods in hospital planning and space organisation is well recognised in practice. Standard configurations and modules are documented for use in design procedures [Nuffield 1955, Radford 1969 & 1971, Joseph 1970], which are incorporated into modern hospital design.

The Medical Architecture Research Unit at the Polytechnic of North London (MARU) undertook to develop automated planning tools in the form of computer-aided design as part of the unit's general objectives to provide design guidelines for practitioners [Thunhurst 1971, Kenyon 1973 & 1974, Beattie 1973 & 1974].

The same unit was responsible for the literature review, "Ward Layout and Nurse Staffing" [Walker 1981], cited in section 8.1.2.

8.2.2 Medical policy and hospital layout

How the hospital is organised is constrained by the layout, and in turn the required activity patterns influence the layout designer. This is well understood by the hospital designer; the same principles were applied in the computer-aided software, ADAPT and ADEPT.

The twin interactive programmes ADAPT (Axiomatic Department Programming Tool) and ADEPT (Axiomatic Department Evaluation Programming Tool) [Beattie 1973, Kenyon 1973] were used for experimental comparisons of medical policy and layout design.

- ADAPT generated layouts given an axiomatic accommodation pattern for a range of axiomatic activity patterns, which the designer could accept or reject.
- ADEPT evaluated layouts (which need not have been generated by ADAPT) for a selection of activity patterns.

Thus activity patterns make demands upon the hospital layout, and data for the case-load and work flow patterns must be included in any evaluation of hospital accommodation. Some examples are quoted from MARU reports in section 8.2.3.

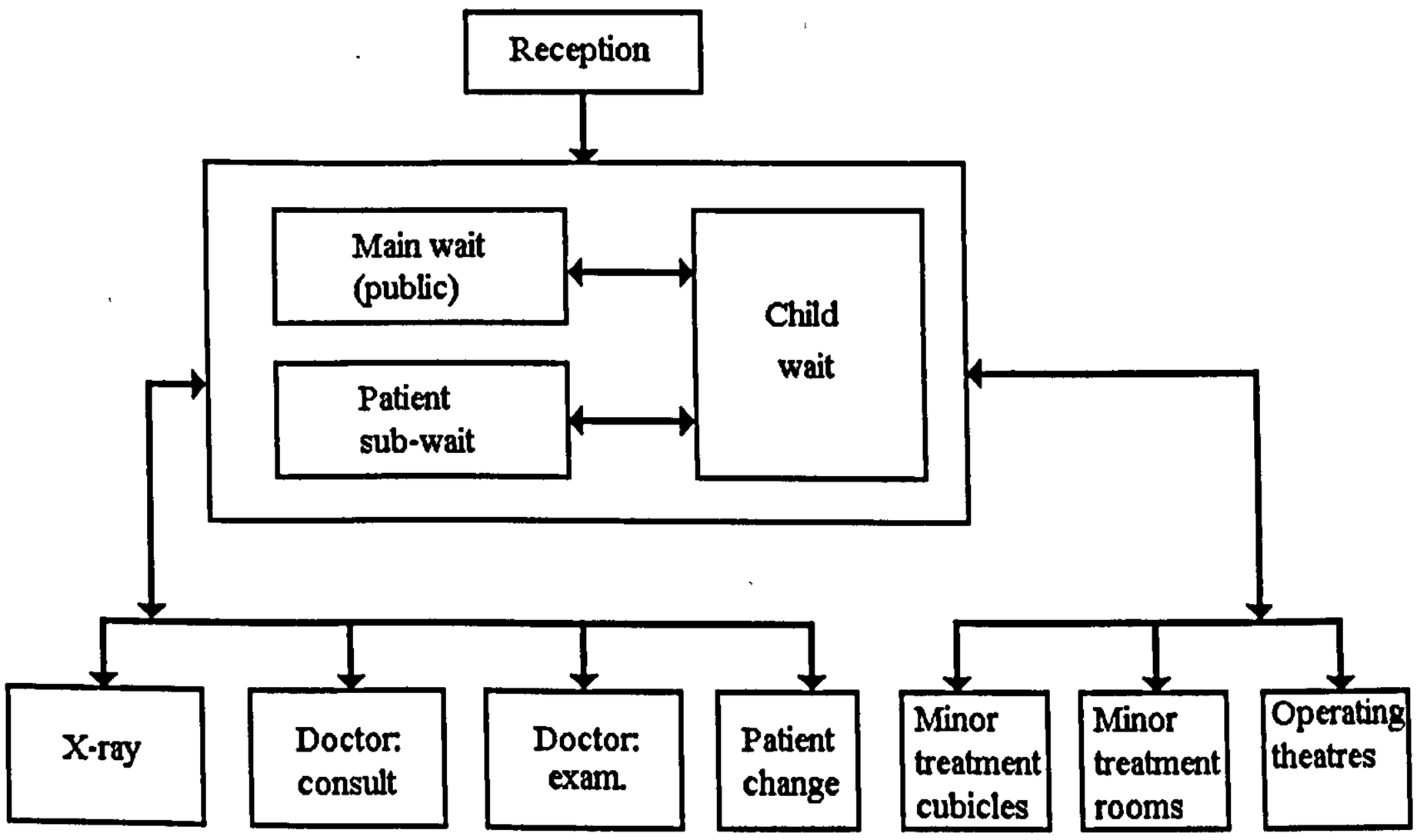
8.2.3 The implications of alternative flow patterns

Two possible flow patterns for an Accident and Emergency Department are quoted from a MARU report on Space Organisation in the Accident-&-Emergency Department [Beattie 1974]. The full diagram is reproduced in Appendix D and sections of it are given in Diagram 23

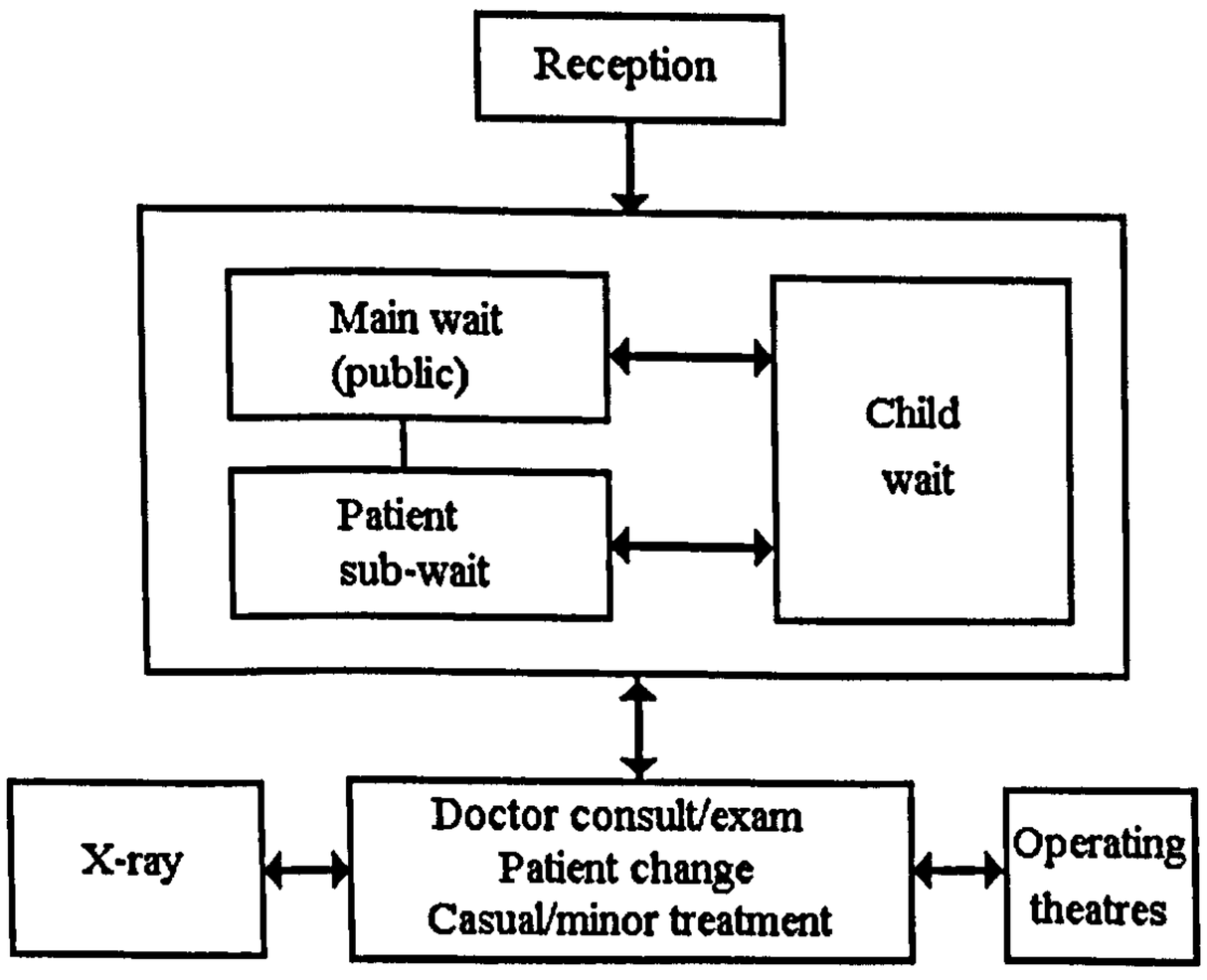
Diagram 23

Two alternative flowlines for consultation and examination in the AED

1. patient to doctor



2. doctor to patient



Each flow pattern can work efficiently only if given the appropriate accommodation.

The demands placed upon the accommodation by each of the flow patterns are examined in Table 17.

Table 17 Accommodation requirements of two different work flow patterns

Work flow pattern	Accommodation requirements			
	main-wait	consulting room	examination room	changing room
patient to doctor patients mobile doctors stationary	large	one per doctor	one per doctor	several cubicles
doctor to patient doctors mobile patients stationary	moderate size	combined consulting/ examination	combined consulting/ examination	combined with C/E or separate

For clinics where consultation and examination rooms are combined there are many possible different arrangements. It is likely that the choice of layout would depend upon the sequence of procedures needed in a particular clinic.

Designs for layouts of combined Consultancy and Examination rooms for clinic use are given in Appendix D.

8.3 The potential for simulation

At the planning and design stage for a new hospital the management and deployment of resources is open, and it would be useful to have clear guidelines as to the effects different ward layouts have on ward function for a given nurse management style and constant levels of staffing and case-load.

However, it has been shown in sections 8.1.2. and 8.1.4 that statistical analysis of data obtained from hospital surveys cannot effectively control for all extraneous factors, but using simulation, an experimental design can be set up which suits the purpose of the investigation. Given suitable survey data, the simulation technique provides the means for examining any combination of factors, and can be used to isolate the effects of the physical layout.

8.3.1 A model for nurse activity on a hospital ward

The simulation technique is frequently used for modelling hospital clinics [Shapiro 1976], but a data-driven generic model, similar to that described in Chapter 5, would be needed for a spatial model of a hospital ward and is recommended for an examination of nurse activity for alternative ward layouts. A model is needed which will simulate the normal nursing activities on a hospital ward, and which can be run for various ward layouts, staffing levels, case-load and case mix, giving performance measures which can be interpreted in terms of the nursing care achieved.

The proposed model operates with a data file which specifies the ward layout as a graph structure, and control factors, such as nurse management style, staffing levels, patient case-load and case-mix are also included in the data.

8.3.2 Performance measures for the nurse activity model

Performance measures which would be appropriate for a simulation model are listed in Table 18. These are similar to the records made in the nurse activity survey described in section 8.1.4, and include nurse-patient contact measures.

The implementation of the nurse activity model described in section 8.4 includes records for the first three measures.

Table 18 Performance measures for the simulation model

Category of measure	Units of measure	
nurse-patient direct contact time (treatment) • by nurse • average over all nurses	minutes	% proportion of total work time
nurse travel time • by nurse • average over all nurses	minutes	% proportion of total work time
nurse travel distance • by nurse • average over all nurses	metres	
patient see-contact from staff base number of patients visible	counts recorded at time intervals	average number over total work time
patient see-contact number of patients visible • by nurse • average over all nurses	counts recorded at time intervals	average number over total work time
visibility of **any** nurse to patient • by patient • average over all patients	counts recorded at time intervals	average number over total work time

The measure giving the number of patients visible from the staff base will normally be a constant figure for a given ward layout, but as part of the normal ward routines some patients receive treatment which requires privacy and curtains are drawn round the bed. Even on an open ward other patients will be obscured from view.

If the number of visible patients is recorded for nurses on normal ward duties, the sight-lines will vary according to the nurse's position, and the overall measure of patient 'see-contact' will be related to the number of staff on duty.

Diagram 24 Curtains drawn around one bed on a small ward

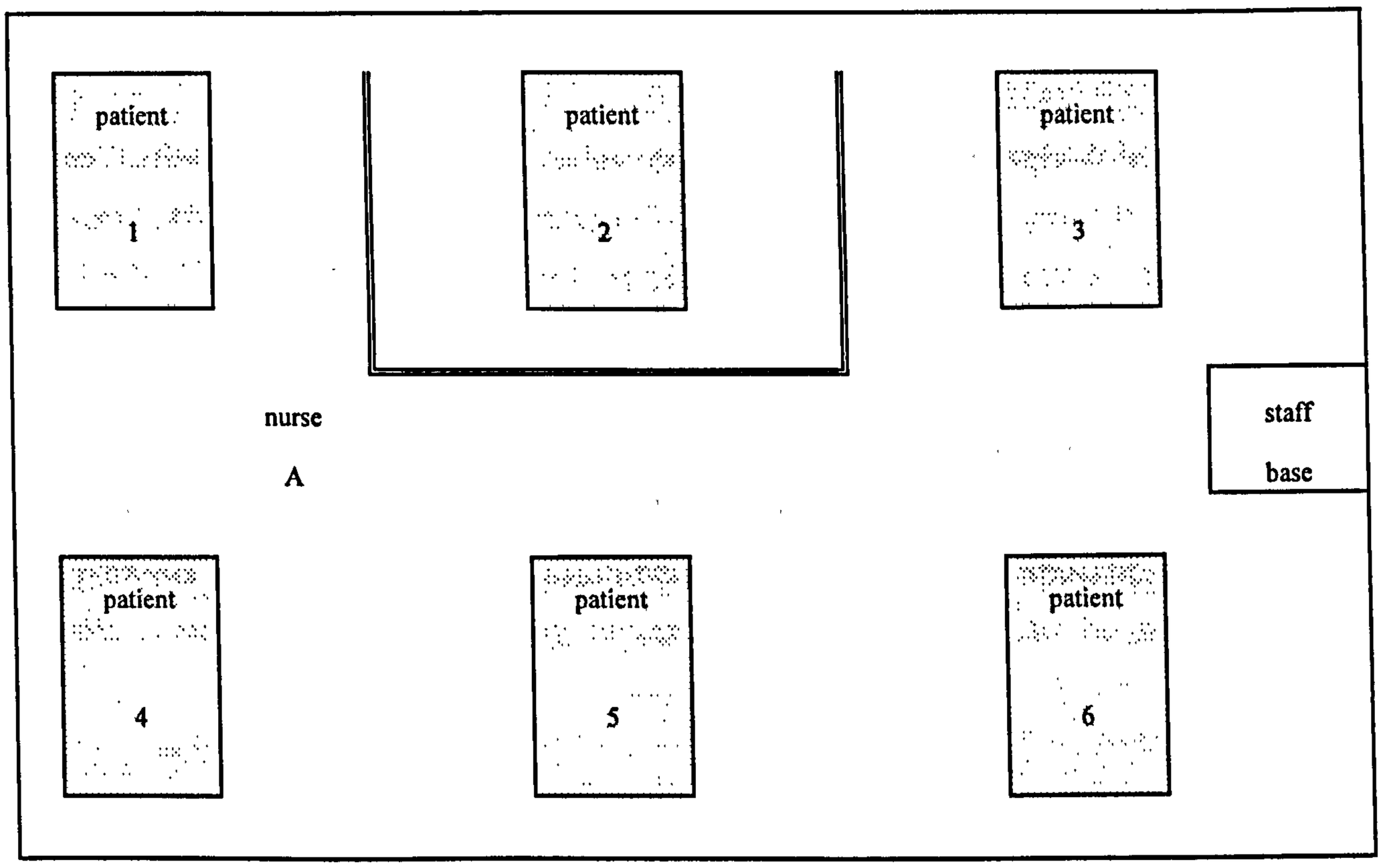


Diagram 24 illustrates the effect of drawn curtains. When curtains are drawn around the bed for patient 2, patient 1 cannot be seen from the staff base. However, nurse A, who stands at the far end of the ward, can see patient 1 but not patient 3.

In practice it is likely that curtains are drawn more frequently for patients with a high level of dependency, but this should correspond with higher staffing levels and give a comparable score for this 'see-contact' measure.

The proposed model, introduced in section 8.4, uses graph structures which refer to a spatial grid. Records for patient 'see-contact' could be made using the existing properties of the graph structure or a geometric algorithm.

8.3.3 The model design

The proposed design includes the entities: nurses, patients and tasks, all three of which are referenced to a spatial grid representing the ward.

The organisation of a typical ward includes routine tasks which are scheduled in advance and other tasks which occur randomly and which may have priority over routine work. The model includes both type of task, with patients making random calls for attention.

Constraints upon physical movements and changes of state are conveyed by graph structures as explained in Chapter 9 section 9.1.4. The graph structures, and lists of nurses, patients and routine tasks are contained within the data file. Specific allocation of patients to nurses is implemented by means of attributes attached to the patients and nurses.

Nurses are mobile entities, each of which move towards the location specified for a delegated task, retrieved from the task list. Apart from making occasional demands for attention, patients are essentially passive and stationary, their level of dependency being modelled as a probability which may vary from patient to patient.

The implementation for modelling nurses, patients and tasks, and the driving mechanism for the model is described in Chapter 9 section 9.6.

8.4 Test runs of the nurse activity model

The model is designed to give a tool for obtaining performance measures for nurse activity during normal nursing routines in wards with different layouts which could be used as an alternative to the survey method described in the MARU report on Ward Layout and Nurse Staffing [Seelye, Ward & Walker 1981] referred to in section 8.1.4. The summary results from the survey have been used to make preliminary checks to the nurse activity model.

8.4.1 Data for two different layouts

The model is demonstrated for a ward with six patients and one nurse where there are no regular duties and the nurse responds to random calls from the patients. Performance measures for the model are collected in a similar manner to the survey, in which nurses were trailed during normal working duties, and include the first three categories suggested in Table 18 section 8.3.2. The times for which each nurse is idle at staff base, moving or actively treating a patient are accumulated to give proportions of time in these three states. The total distance walked is also recorded. Records kept for the patients include the times for which each patient is comfortable, waiting for a nurse and being treated. A summary table of these measures compared with the survey results is given in section 8.5.

The survey was conducted with a selection of 'open' and 'divided' wards in three hospitals, and recorded patient *'see-contact'* for each nurse. This is an average number of patients visible to the nurse, calculated from regular observations of the numbers of patients who could be seen as she moved from task to task. The authors claimed that this measure varied significantly between the open and divided wards, since patients in separate rooms could not be seen by nurses working in other rooms. It is possible for the nurse activity model to include records for the number of patients visible to nurses while moving about the ward but this has not been demonstrated.

Two layouts have been used:

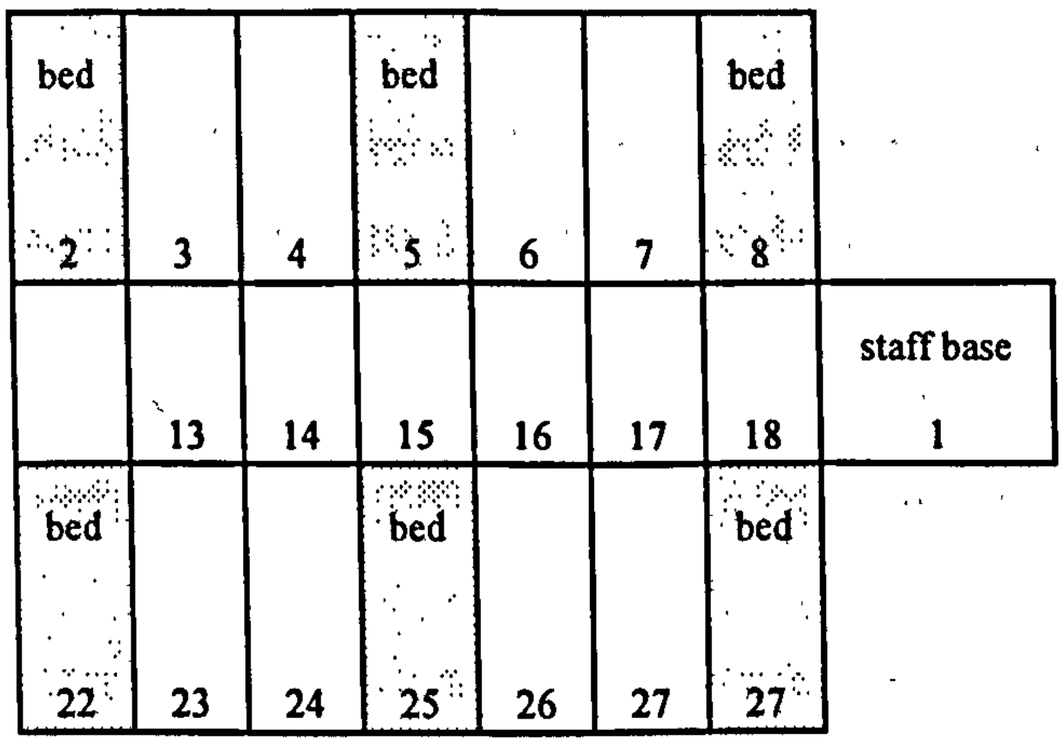
- in the first there are three beds arranged on opposite walls. This layout is further examined in Chapter 9, Diagram 27, section 9.3.2.2.
- in the second layout all six beds are arranged along one wall with the staff base at one end.

The data-files corresponding to each layout are given in Appendix B.

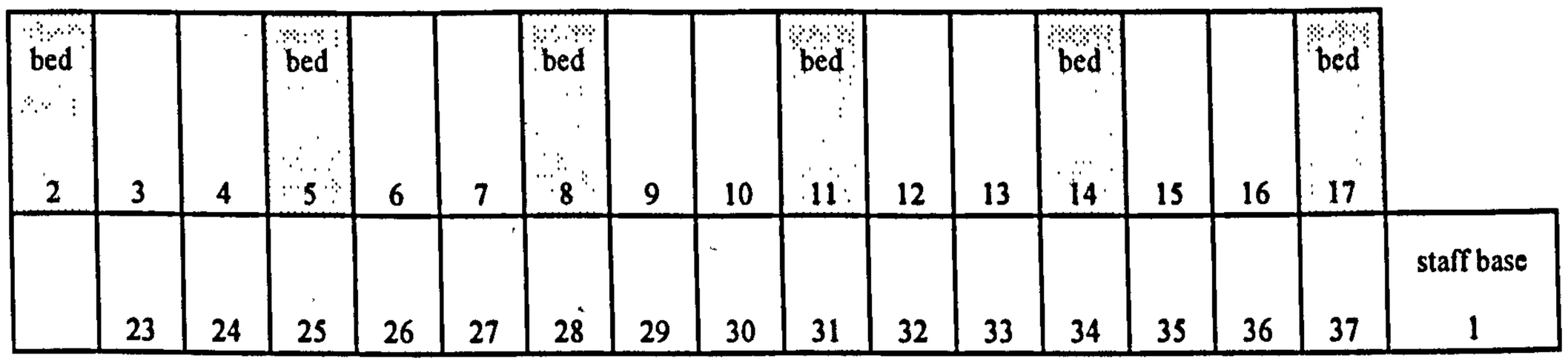
Both layouts are shown in Diagram 25. These are 'open' in the sense that all patients are visible to the nurse while she is working. The layouts for the twelve wards used in the survey would require similar but larger data-files.

Diagram 25 Two alternative layouts for a ward with six beds

Layout 1



Layout 2



During the survey nurses were observed for a three hour period. To obtain comparative results, sample runs have been carried out for both layouts, a long run of two hours and forty seven minutes, and five repeated runs of length 33 minutes.

8.4.2 Model calibration

The survey results indicate that on average nurses were moving for 8% of the time (14 minutes 24 seconds), and walked approximately 991 metres, from which their average speed can be estimated to be 1.15 metres per second. This is slightly slower than the reported average speed of 1.26 metres per second as given in Chapter 7 Table 12. In a working environment the slower speed may be caused by the nurse having to carry equipment or pausing to make quick checks of patients' condition.

A step-length of 5 seconds was used for trial sample runs of the model in which treatment times were 25 seconds and the probabilities for patients calling for attention were 1%, 3% and 5%. The short treatment time resulted in nurses moving for 36% of the time. Nurse utilisation ranged from 68% to 100%. Effective nurse movement speeds of approximately 0.46 metres per second were recorded which is unacceptably slow.

A better representation of nurse movement is obtained using a time-step of 2 seconds. Ward layouts would typically require 2 or 3 paces to move between blocks, where a pace is assumed to measure 0.7 metres. Since a nurse moves from one block to another in one time step this gives a walking speed of approximately 1 metre per second. Simulation runs using a two second time step gave acceptable results as reported in the following section .

In general treatment times would be sampled from a distribution according to information relating to ward routines, but in the test runs treatment times are assumed to have a constant duration of 200 seconds.

Patients will normally vary in dependency level, but for the test runs patients are alike and each patient has a fixed probability of calling for attention during the two second interval. For a long run of 10,000 seconds the probability for patient calls is taken to be 0.01; for the repeated runs of length 2000 seconds the probability for patient calls is 0.005.

The trial runs use a single stream of pseudo-random numbers to model random calls from the patients, and the five independent runs use five selected different seeds for the random samples. Thus conditions are matched for the two layouts.

A complete calibration would be carried out given full information about the ward which would include case-load, case-mix, nursing routines and staff available. This would produce a useful experimental model.

8.5 Evaluation of the nurse activity model

The results of the trial runs of the model are given in Appendix B. These are summarised in Table 19

Table 19 Summary model performance measures compared with survey results

Category of measure by layout of ward	Survey results		Nurse Activity Model Results (one open room)			
			Run A Single long run		Run B Average of 5 short runs	
	open	divided	layout 1 (compact)	layout 2 (long)	layout 1 (compact)	layout 2 (long)
(% of total nurse work time) nurse-patient contact time	27	27	97	92	96	91
nurse travel time	8.0	7.8	3	8	3	8
nurse rest time	24	23	0	0	1	0.5
(% of total patient time) patient being treated	*	14 (for 4 patients)	16	15.5	16	15
patient waiting for attention	*	*	62	69	46	46
patient comfortable	*	*	22	15.5	38	40
patient see & hear contact	14.8	3.9	*	*	*	*
nurse travel distance (metres over 1 hour)	330	310	115	234	134	266

NB. * indicates a missing result.

Results given in the survey report [Seelye, Walker & Ward 1981], give useful checks for the nurse activity model. The proportion of time for which nurses are moving is reported to be approximately 8% for both types of ward layout, and this figure agrees with the simulation results for the second layout in which the distance of the furthest patient from staff base is 35 paces (24.5 metres), normal for a typical hospital ward. In the compact layout the furthest patient is 17 paces away from staff base (12 metres), and the simulation records the proportion of time for which the nurse is moving to be less than 8%.

The test runs of the simulation model give a lower estimate for the distance walked. This may relate to underestimated ward dimensions but is more likely to be the effect of model simplicity. The nursing activities observed in the survey included ten broad categories taking place in fourteen types of location. The test model omits the multiplicity of tasks with the accompanying fetching and carrying, and this will underestimate total travel distance. However an experimental model would be run with routines which match those of the ward.

The survey also included measures of patient *'see-contact'*. The nurse activity model does not keep such records but the model design is capable of extension to include them. This could be achieved by the addition of a further list attribute assigned to spatial blocks to store the identities of patients visible from that block, together with methods which change and communicate this information. For instance methods would be required, by which a nurse, when drawing curtains round a bed would cause changes in the states of the spatial block attributes relating to patient visibility. Other methods would be required to enable another nurse to access this information in order that records for the number of visible patients could be made. This would be done at fixed intervals, say at every 100 time-steps.

The survey performed *'nurse trails'* and concentrated upon recording the various states of the nurses, but it is possible for the model to collate simultaneous observations for the patients. Trial runs of the model records the proportion of time for which the patient is comfortable, waiting for a nurse of being treated, but only a few patient treatment times were reported in the survey. It is also possible for patient's awareness of nurse proximity to be recorded in a similar manner to that used for patient 'see-contact'.

There are two ways in which the tabulated results indicate the usefulness of a simple model:

- The dependency of patients in run A is double that of patients in run B, which results in differences between the times for which patients are comfortable and waiting times for treatment. Overall treatment times are similar since the nurse is fully utilised.

- The test runs also indicate differences between the two layouts which are more pronounced in the run which is longer and with more demanding patients.

The survey rejected the possibility of different layouts having an effect upon nursing efficiency, but, in practice, nurse management must actively compensate for deficiencies in accommodation arrangements in order to provide good patient care. There cannot be absolute matching of nursing procedures in surveys of this sort and this throws doubt upon the conclusions.

The test runs of the nurse activity model have demonstrated that it provides a reliable tool for investigating the efficiency of different ward layouts. It can be claimed that the survey brief, which was to study the relationship between ward function and ward layout, would be more conclusively examined with the aid of the nurse activity model.

The application of the general method is further examined in Chapter 9.

Chapter 9
The Generic Model Applied to Other Spatial Problems

9.1 Extension of the modelling strategy

The model, developed and validated in chapters 5 and 7, is a special case of a generic model which imposes a time-updating mechanism upon a static descriptive model. It defines the sequence of states in a similar manner to Logical Data Modelling in SSADM [Eva 1992].

9.1.1 The features of the generic model for emergency evacuation

In the generic model for emergency evacuation the topological data and functionality of the building is modelled using a single graph structure. Under congested conditions the model is quasi-continuous; the people moving out of the building are not modelled explicitly but represented as numbers occupying each portion of space. The updating of the model is the calculation of these numbers at fixed time steps.

The model uses:

- a single graph structure to represent the spaces and routes out of a building,
- and a control algorithm which updates time in fixed steps.

9.1.2 The use of multiple graph structures

The method can be extended to model spatial situations where entities need to be modelled individually and where several classes of entities interact. This is done using multiple graph structures, each of which are referenced to the spatial grid, and which provide the movement constraints for each entity class.

For instance, a simple system in which two competitors make alternate moves such as in a game of chess, could be modelled using two graph structures and could proceed using a **client-server** model [McGregor & Sykes p273]. This is possible using an object-oriented approach since the rules for change could be embedded within the component objects resulting in a decentralised or 'stair' model [Jacobson P.225].

However, the generality of systems will require a **master-slave** model. Most systems need a control algorithm to schedule the events. For a complex model this will be the three phase discrete-event simulation which uses an event calendar, as described in section 3.3.1.

9.1.3 Modelling individual entities

It should be noted that the control algorithm is needed for computational records only; the simulation model preserves the autonomy of behaviour for active entities. This is achieved during the execution of the model by the three-fold stages of

- assigned 'goal',
- sampled individual response,
- action constrained by the environment.

The use of object classes provide a range of entity attributes to represent the assigned 'goal', and the data to specify a distribution of responses, so that the actual response can be sampled at the time of execution. The consequent action is constrained by the environment which includes the states of other entities in the system, communicated by means of the graph structure. Environmental information is effectively supplied by means of the graph structures, and in this sense the methodology is similar to that used in 'virtual reality' simulations as described in section 4.5.2 [Still 1992].

9.1.4 The graph structure as conceptual model

The basic components of the graph structure are the blocks and links. The entity-relationship model for these is given in section 5.3.1.

It was established that the structure is fully specified by the following:

LINK TABLE (link identity, identity of block before, identity of block following)
BLOCK TABLE (block identity)

Each link is uniquely determined by the block immediately preceding and the link immediately following. It can model :

- either the physical proximity of portions of space; the accessibility from one block to another,
- or a change from one state to another.

In both cases the link represents an event since physical movement is a special case of a change of state in a simulation model.

Thus a data-file consisting of tables of links and blocks, can fully specify the rule-base for a simulation model.

9.1.5 Implementation of the graph structure

The evacuation model uses a graph structure called a Net, consisting of a table of blocks, representing portions of space in a building, and a table of links representing possible movements between blocks. Object classes represent blocks, links, lists of blocks, lists of links and the Net itself.

It is necessary that each component of the structure has access to information about neighbouring components, and field attributes for the objects are created to hold this information, often in the form of a pointer to another object or to a list of objects. In the evacuation model, neighbourhood information enables the automatic construction of routes out of the building. This is fully described in Appendix B.

The data-file for the evacuation model defines a one-way situation which results in routes which are sequential lists of blocks terminating in an exit block. A more sophisticated knowledge base is required to help entities identify routes to a specified target, which, as explained in section 9.6.2, is needed in the nurse activity model.

9.2 The universality of model structures

The underlying structures of very different systems may be analogous, and this can be useful when creating models for new problem domains. The classification given in section 2.1.3.2. is useful in identifying similarities. Two pairs of examples are given of different situations which have similar model structures.

9.2.1 Models with permanent entities

Two examples are given of a model with:

- a determined number of stationary customers
- one or more servers who are mobile.

1. The hospital ward;

A determined number of patients occupy beds in a ward. When the patient needs attention a call is made to a nurse. The patient remains uncomfortable until a nurse is free to give treatment.

This situation is conceptually similar to the following example.

2. The machine interference problem;

A determined number of machines are working, but are subject to break-down. When a machine fails it remains non-functioning until a service-man is available. The machine continues working when the repair is completed.

In both cases, the spatial layout affects the movement of the servers and the organisation could include routine service calls to the customers.

9.2.2 Models with transient entities

Two examples are given of a model where customers move into one or more service locations for which a range of servers is available.

- The customers are transient but stationary for service.
- The servers are mobile and restricted to some service locations.

1. The clinic which uses combined consultancy/examination rooms.

A patient is called from main wait into a consulting/examination room. The patient undresses and waits for a doctor. When a doctor assigned to that room is free the patient is examined and a decision is made. The patient dresses and leaves for the next treatment area.

This situation is conceptually similar to the second example.

2. The petrol station forecourt;

Cars arrive and select a standing place near a pump when there is room. Service starts when a pump is free which can reach that space. Cars move away when served to the payment area.

In both situations the customers remain in a spatial location for several stages of the service process, and the assignment of the customer to service location might be crucial to how the system works.

9.3 Example models to demonstrate the extended methodology

As explained in the previous section, models with analogous structures occur in very different situations. In this section the structures of some selected models are compared.

9.3.1 The selected models

Two models are considered which are characteristic of many organisational situations:

- the petrol station forecourt problem,
- the treatment of patients in a hospital ward.

The petrol station forecourt problem was introduced in section 3.2.1 and was used to demonstrate established modelling strategies.

In Chapter 8 the evaluation of hospital ward layouts was discussed and a model for nurse activity on a hospital ward which would assist in layout evaluation was presented.

The two situations are examples of models where entities are modelled individually, but there are major differences:

- In the first example customers move and servers are stationary, while customer entities are transient.
- In the second case it is the servers who move, while customer entities are stationary and permanent.

The comparative features of the models are shown in Table 20

Table 20 The features of three example models

	Emergency evacuation model	Petrol station forecourt model	Nurse activity on hospital ward
Number of spaces in spatial net	very large	small	moderately large
Number of customers	large mobile	few at one time mobile	a constant number but not large stationary
Servers		stationary	mobile
Rules for movement	one-way	determined routes	complex

9.3.2 Graph structures for mobile entities

The graph structures provided for **mobile** entities will be similar to that used in the generic model for emergency evacuation, where the nodes represent portions of space and the arcs represent possible movement from one space to another. For the evacuation model movement was one-way resulting in an acyclic structure, but in general, the resulting structure will contain cycles.

A conceptual spatial grid exists for the whole system and mobile entities may occupy defined locations referenced on that grid. All the graph structures can be superimposed upon the spatial grid, each one using a selection of the locations.

9.3.2.1 Mobile entities in the petrol station forecourt problem

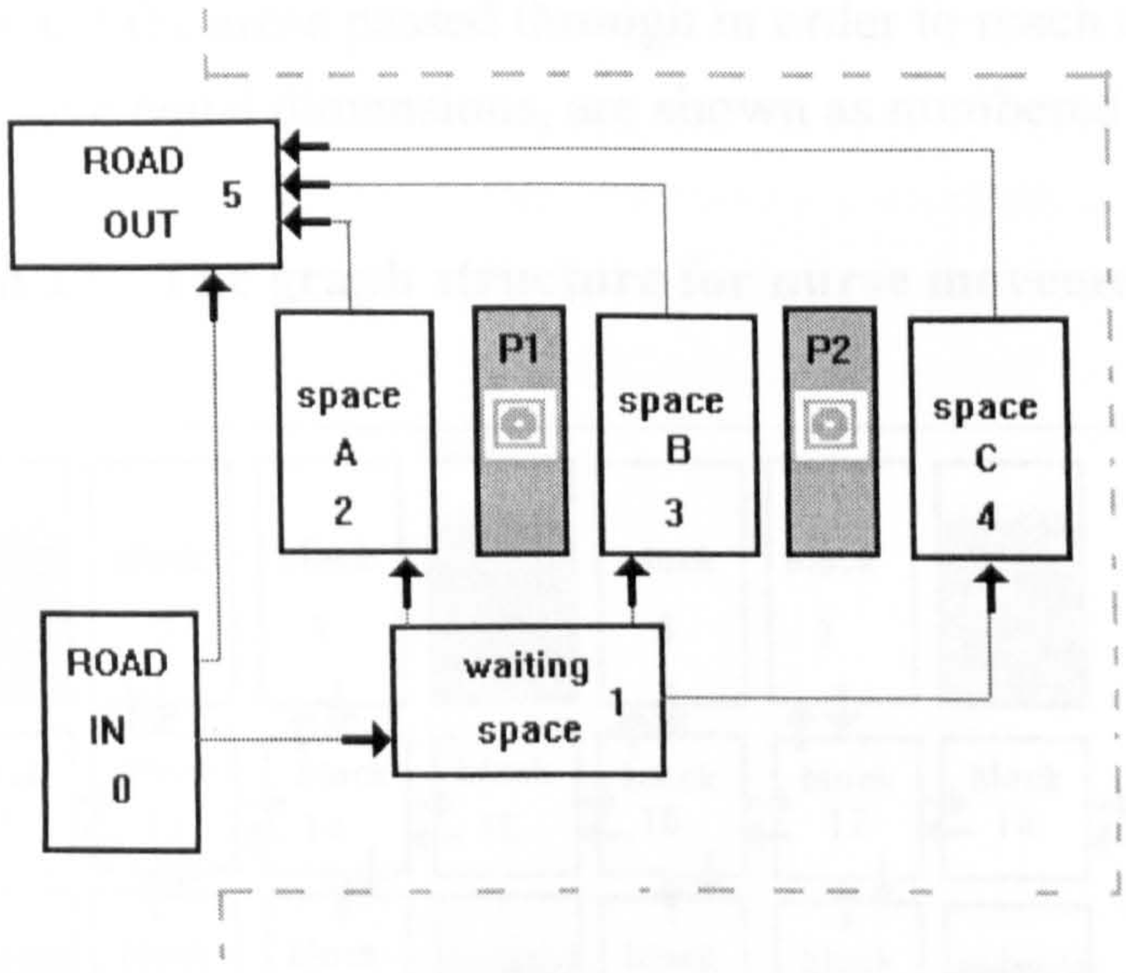
Customers, in the form of cars, are the mobile entities in the petrol station forecourt problem.

The petrol station forecourt is subdivided into spaces, big enough for a single car. In this example five spaces are identified which are listed in Table 21, and shown as numbered blocks on the graph structure in Diagram 26.

Table 21 The spaces in the petrol station forecourt model

Space	Identity number
Road in	0
Entrance/waiting space	1
Standing space by pump 1	2
Standing space between pumps 1 and 2	3
Standing space by pump 2	4
Road out	5

Diagram 26 The graph structure for the spaces in the petrol station forecourt



The graph structure, given in Diagram 26, defines the possible movement between spaces which, in this case, is one-way.

The links between blocks, indicated on the diagram by arrows, have identity numbers which are not included. Note that the structure is acyclic with a *source*, space 0, and a *sink*, space 5.

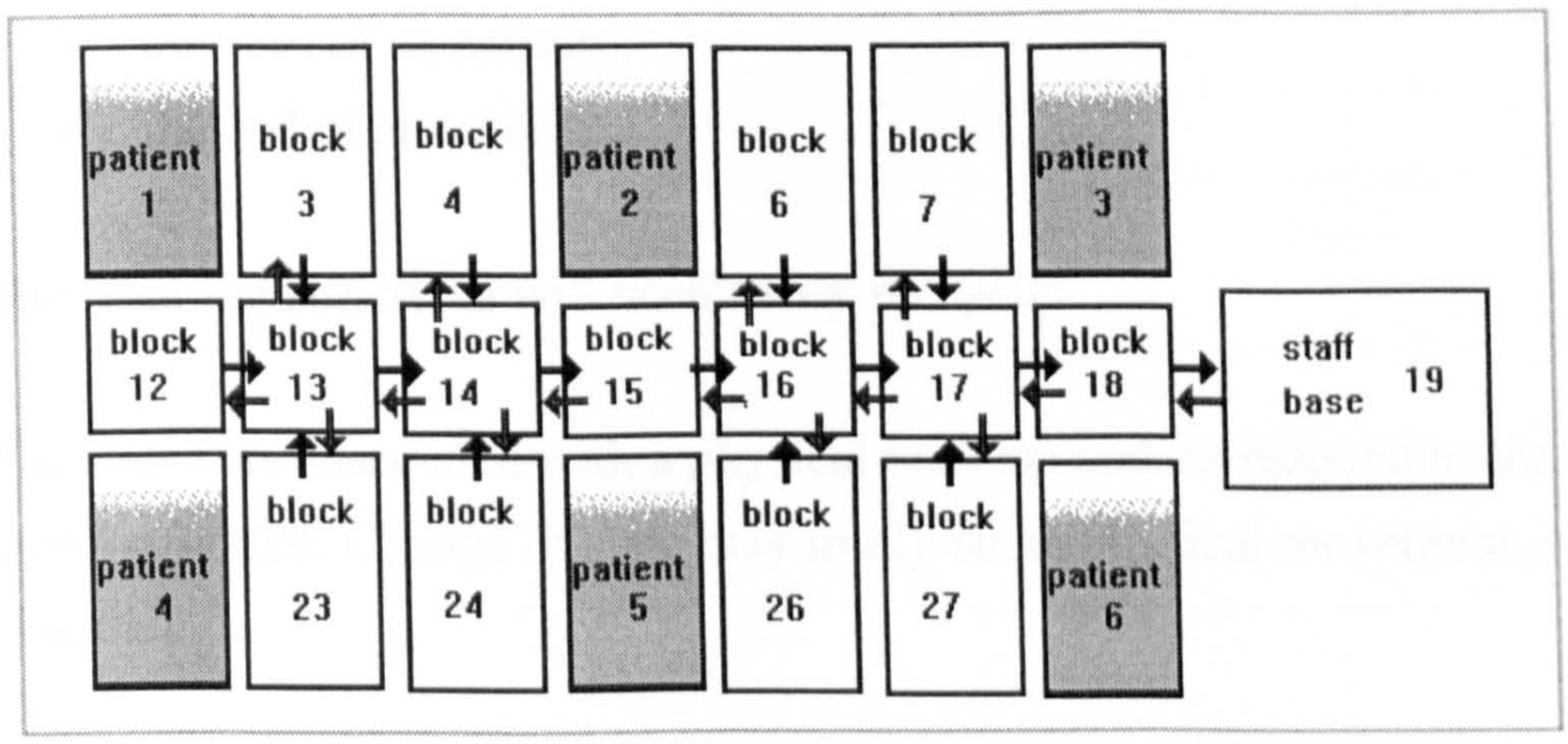
The petrol pumps are shown on the diagram but are not part of the graph structure.

9.3.2.2 Mobile entities on the hospital ward

The mobile entities on the hospital ward are nurses.

The hospital ward is subdivided into areas representing the staff base, treatment locations and the areas passed through in order to reach a destination. The areas, which may not have equal dimensions, are shown as numbered blocks in Diagram 27.

Diagram 27 The graph structure for nurse movement on a ward



Nurse movements are the links between blocks which are indicated on the diagram by arrows. The identity numbers for these are not shown.

The structure uses the same entity-relationships described in section 5.3, in which links are defined to be one-way. This gives an unambiguous representation which is useful for defining routes between locations. Unlike the graph structure in Diagram 16 section 5.2.2, the structure contains many cycles, and there is no *source* or *sink*.

Patient beds are shown but are not part of the graph structure. However the blocks which represent the spaces alongside each patient's bed will be associated with that patient. For the sake of simplicity, no links are shown for direct movement between adjacent beds without entering the corridor area. Likewise small corridor areas which act as junctions are shown as separate blocks, such as blocks 13 and 14. These could be combined to give one block with six links entering and leaving. This sort of adjustment is part of model calibration, but simplicity is preserved.

9.3.3 Graph structures for stationary entities

The graph structure for the **stationary** entity is a conceptual model for the changes of state.

The nodes represent the states:

- busy or idle for a server,
- being served or not being served for a customer.

The arcs represent changes of state, which are events corresponding to:

- the start of an activity,
- the end of an activity.

The resulting structures will necessarily be cyclic.

The states are associated with a physical location and correspond to the node of another graph structure. Change of state may imply some physical movement, but this is not essential.

9.3.3.1 Stationary entities in the petrol station forecourt problem

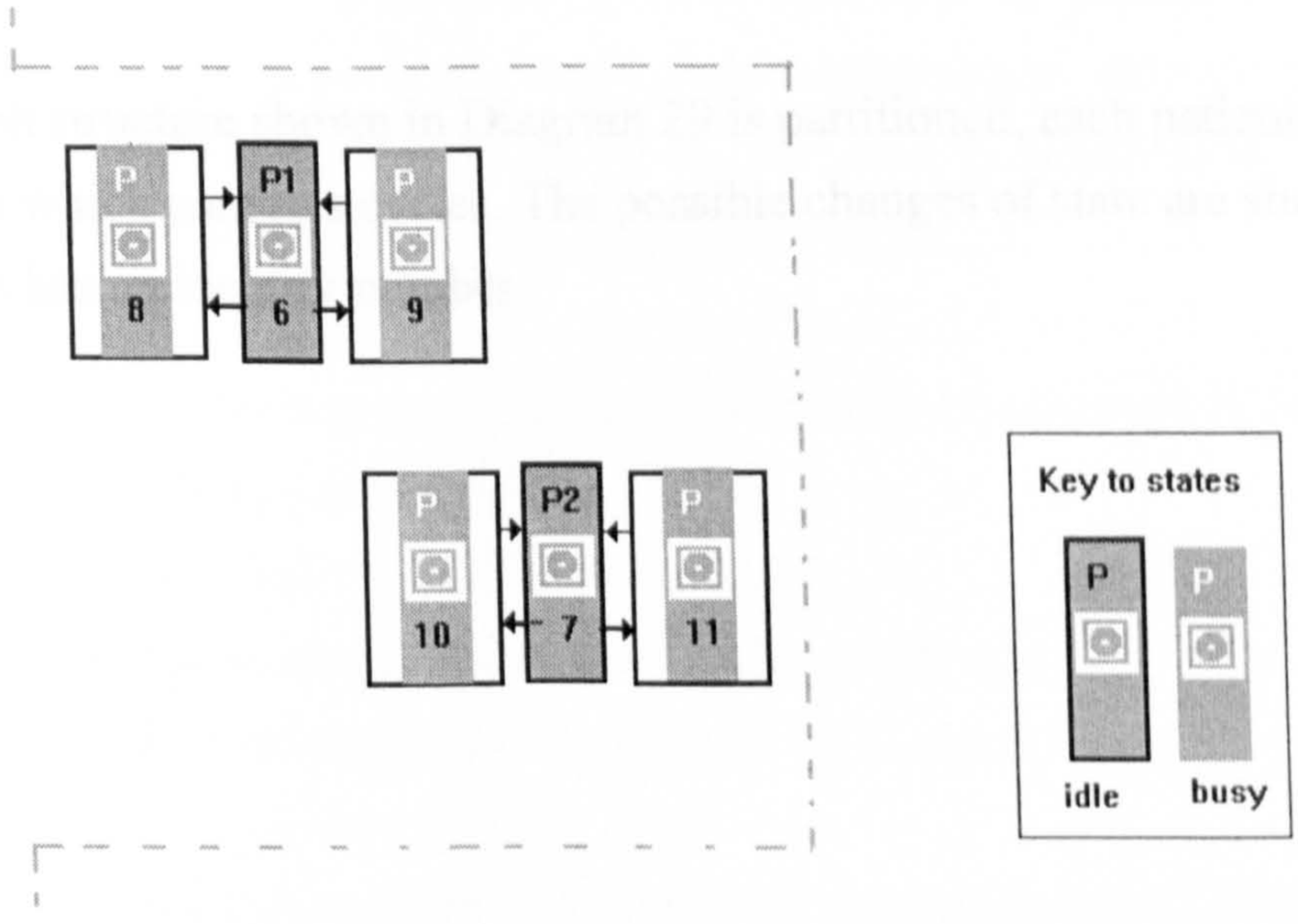
The servers for this petrol station forecourt problem are pumps, which in reality are stationary, but can serve cars occupying certain spaces.

Table 22 lists the pump states, which are shown in the graph structure in Diagram 28.

Table 22 The pumps in the petrol station forecourt model

Pump identity	State of pump	Identity of state
1	serving car in space 2	8
1	idle	6
1	serving car in space 3	9
2	serving car in space 3	10
2	idle	7
2	serving car in space 4	11

Diagram 28 The graph structure for the pumps



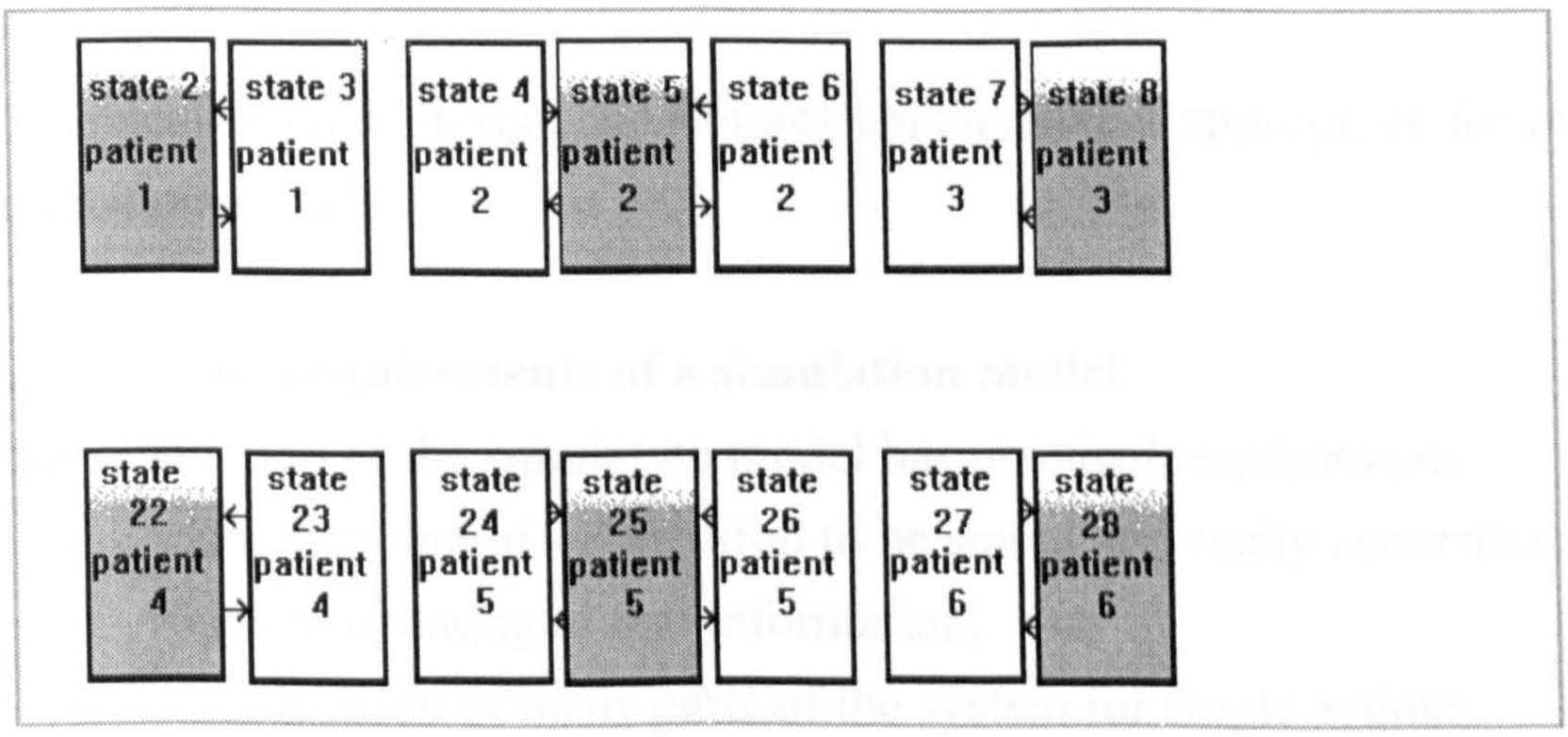
The graph structure defines how pumps may move between states. In this case the graph structure is partitioned and each part contains cycles.

The arrows denote a change of state, each of which has an identity number.

9.3.3.2 The stationary entities on the hospital ward

The patients are the customers who do not move, but remain in bed on the hospital ward.

Diagram 29 The graph structure for the patients



The graph structure shown in Diagram 29 is partitioned, each patient having a separate structure which contains cycles. The possible changes of state are shown as arrows, each of which has an identity number.

9.4 The object-oriented model design

An object class called a **Net** is used to represent a graph structure. Net has attributes which are lists of other objects, which are interrelated lists of blocks and links. The structure is described in section 5.4. and Appendix B.

The attributes and methods developed for this system of objects are duplicated for other classes of objects needed in the implementation of the simulation models. [Appendix B]

The object-oriented design has features which make it appropriate for more complex simulation models.

9.4.1 Special requirements of a simulation model

Implementation of the simulation model has practical requirements:

- a large amount of information to be stored and easily accessible,
- frequent updating of that information,
- co-ordination of many parts of the system for single actions.

All three requirements are satisfied with an object-oriented design. The objects can be assigned attribute fields for the storage of local information, and methods to retrieve and update information. With the use of pointer attributes and lists of objects, information can be retrieved quickly without replicating storage space. Thus the methods developed for OOP, in which each object controls its own data, facilitate controlled access to the object's data and controlled updating of information.

Co-ordination of the many parts of the system may be achieved in two ways. The entities may be conceptualised to be:

- closely bonded with the spatial objects,
- independent objects referenced to the spatial net.

These two options are examined in sections 9.5.1 and 9.5.2.

In both cases, the management of events will require an event calendar. For the first approach, one of the spatial nets will need to be dominant and will trigger events, while in the second approach a control algorithm schedules events.

The first approach makes use of conceptual generalisations for the block and link objects, which empower the spatial components to interact in driving the model. This is explained in sections 9.4.3. and 9.4.4.

If entities are represented by independent classes of objects, the control algorithm can co-ordinate events by

- first checking necessary conditions using the objects' query methods,
- and then calling the objects' methods for action if the conditions are satisfied.

The driving mechanism is similar for each model design.

In the following sections the various types of events and the activation of events are considered with reference to the first approach applied to the petrol station forecourt problem. The use of independent classes of objects, as applied to a model for nurse activity on a hospital ward, is considered in section 9.5.

9.4.2 The representation of events

In the graph structure for a building layout the links between blocks represented a potential movement from one block to the next, and are the events for that model. The control algorithm for the evacuation model activates events by reference to the blocks, since each block has a unique following link to be checked.

In the modified strategy links are identified as constrained or unconstrained. This corresponds to events which are **conditional** or **bound** (time dependent).

A few examples are given for the petrol station forecourt model, which refer to diagrams 26 and 28. There will be similar bound and conditional events for the nurse activity model.

- Movement from space 2 to space 5 is a bound event, since it is scheduled to happen when service in space 2 ends.
- Movement from space 1 to space 2 is conditional upon there being a car in space 1 and space 2 being empty.
- Pump 2 changing from state 10 (busy) to state 7 (idle) is a bound event, since it is scheduled to happen when service by pump 2 ends.
- Pump 2 changing from state 7 (idle) to state 11 (busy) is conditional upon pump 2 being in state 7 and there being a car in space 4.

9.4.3 Generalisations for the states of blocks

Referring to the petrol station forecourt problem, the entity classes for cars and pumps may be closely bonded with the graph structures for spaces and pumps, where links represent changes of state. Generalised concepts are introduced in order to implement a common treatment.

The concept of **readiness** is required. This occurs when a space or pump *wants* to activate its following link. The *readiness* of a block is time dependent. An attribute, in effect a time cell, is attached to each block to denote the *time of readiness*. This becomes a field of the object block.

The concept of **availability** is also required. This occurs when it is *permissible* for a space to be moved into, or when a pump is *available* for service. The property is conditional, possibly dependent upon other entity types, and is implemented using a Boolean function. Note that a *sink* is available at all times, and that a *source* is never available.

9.4.4 The activation of events

Links represent events, and these are referenced by the control algorithm for the activation of events. Spaces are the dominant entity type, and each link between spaces has the role of checking the *readiness* of the preceding space and the *availability* of the following space. Both *readiness* and *availability* of a space can be determined from attributes of the space and other entities involved.

The **master-slave** control algorithm uses the event-calendar as a scratch-pad giving current information about the system. As shown in Table 23, the events listed are each identified with a space identity and scheduled clock time. The link which will activate the event can only be determined at the clock time scheduled, since *availability* will be dependent upon the state of the model at that time.

Table 23 The event calendar

Time of scheduled event	Identity of space	Type of event
0	4	Complete service
7	2	Complete service
5	0	Car arrival

All events on the event calendar are bound, but some will be exogenous, such as *Car arrival*. Others will be scheduled as the simulation proceeds, such as the *Complete service* event.

The mechanism for the activation of events proceeds as follows:

- The control algorithm will retrieve the event with earliest scheduled clock time.
- The space identified will determine a link.
- The link first checks that it can trigger the event and then does so.

In some cases there are several links, one of which would be able to trigger.
For instance, *Car arrival* may be assigned in two ways:

i. preferably, to the link leading to *space* 1,
ii. otherwise, to the link leading to *space* 5 (which can always trigger).

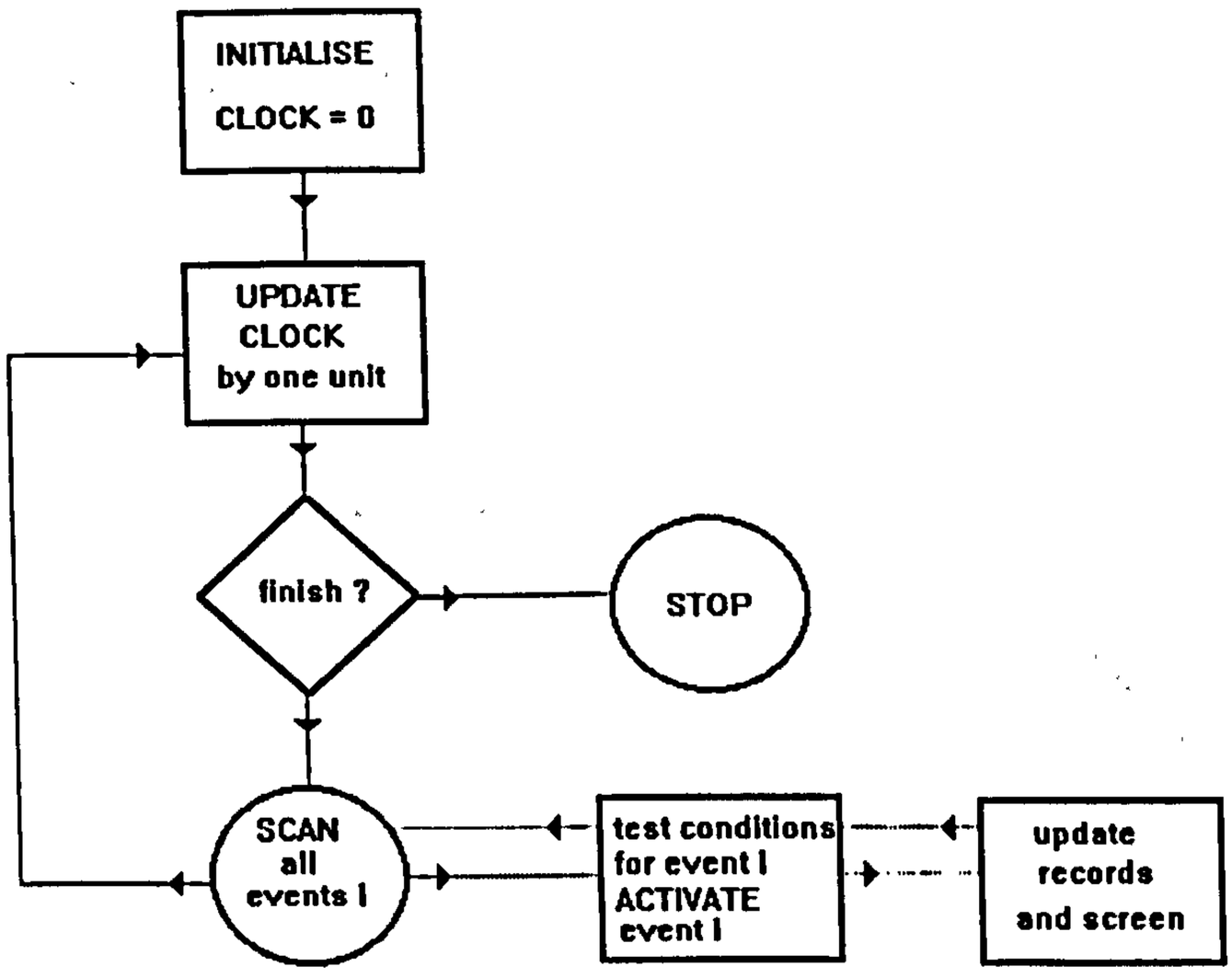
Conditional events, such as *Move to pump* and *Start service* are not included on the event-calendar.

There must be a mechanism for updating clock time. This can be done in two ways as described in sections 9.4.5 and 9.4.6

9.4.5 The master-slave algorithm which uses a fixed time step

The simulation clock time may be updated in **equal steps**, which is appropriate if screen graphics are used, or if events happen regularly at every time step. This algorithm is used for the evacuation model.

Diagram 30 The fixed time step algorithm



In the evacuation model, this algorithm scans the space components, moving from source to sink, adopting a priority order at each junction and activating movements when these are possible. This was described in Chapter 5, section 5.5.2.

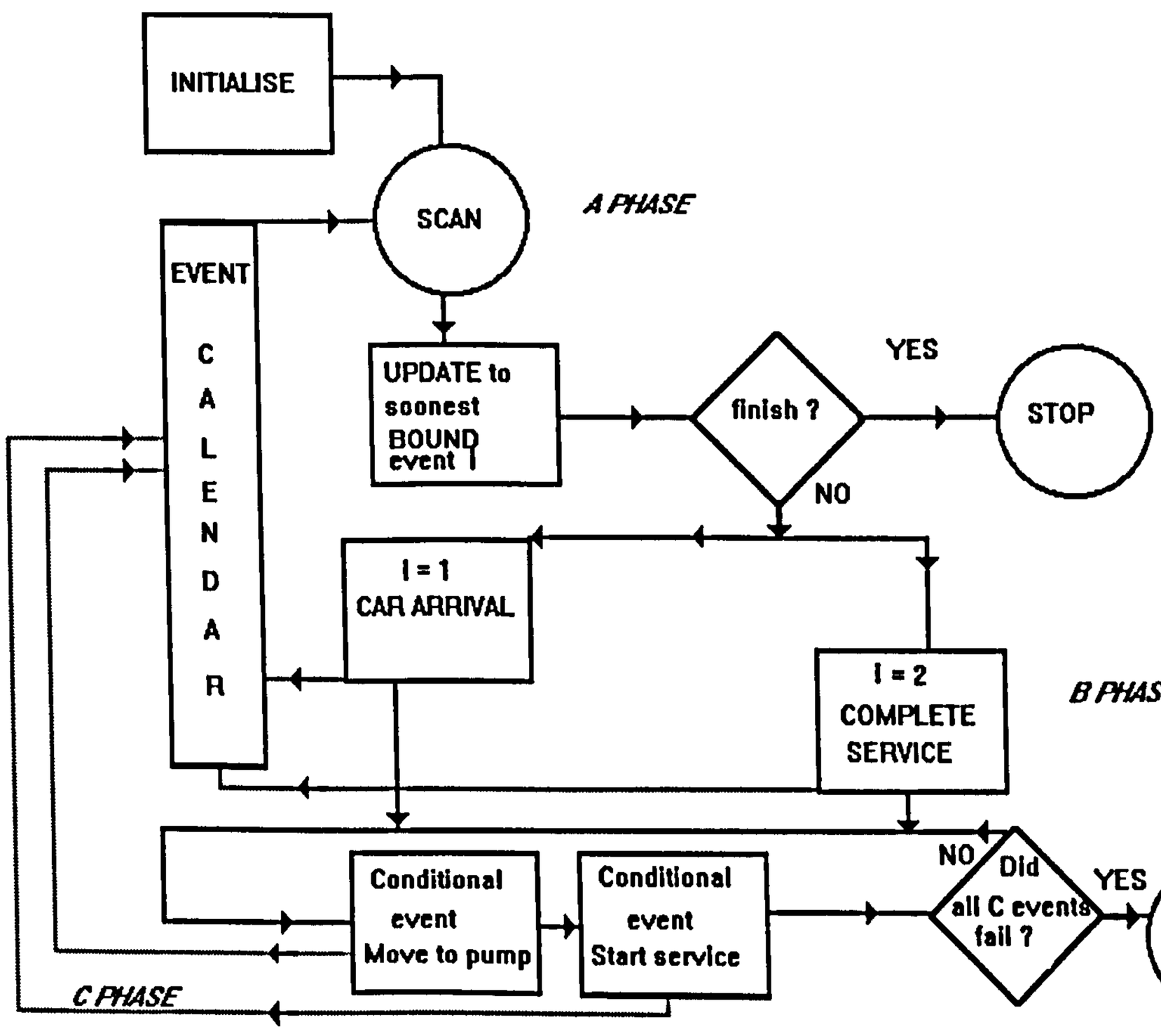
For general problems, there is a range of possible events, and the algorithm would check the conditions for each event in turn, and activate those events for which the conditions were satisfied. Since events are scanned once only at each time step, the algorithm shown in Diagram 30 will give a correct representation only if bound events are checked first. The algorithm is then equivalent to that shown in Diagram 31, but with time updated in fixed steps.

A modification is necessary for the modelling of situations such as the movement of people in two way flow, where simultaneous events are interrelated. The LRT congestion model used a repeated check for people's movement at each time step [Weston & Shapiro 1987].

9.4.6 The master-slave algorithm which uses a variable time step

The simulation clock time may be updated from **event to event.** For driving the model without a fixed time step, an **event calendar,** described in section 9.4.4., is essential. As shown in Diagram 31, the control algorithm updates clock time to the time of the soonest event, retrieves events in turn from the event calendar which are activated and in turn cause scheduled events to be written to the event calendar,.

Diagram 31 The variable time step algorithm



This is the standard algorithm for three phase discrete event simulation described in Chapter 3 (section 3.3.1), but Diagram 9 included the event *Car leaves* which, in this model, has been combined with the event *Complete service* for simplicity.

The bound events, *Car arrival* and *Complete service* are activated by methods of the object *space.* The conditional event *Move to pump* needs only a check of the standing space required, but the event *Start service* needs checks to be made of pump availability. For instance if a car was standing in *space* 3, the links for pump state, 7-10 and 6-9, would be checked. One of these could trigger the *Start service* event.

For both algorithms the activation of events would include the updating of records and screen display, and this is more straight forward for the fixed time step.

9.5 Implementation using object classes

The emergency evacuation model uses object classes to model the spatial components. It is now necessary to model the entities more explicitly.

An approach should be adopted which gives an acceptable and effective overall design

9.5.1 The requirements of a good model design

The model design must satisfy three major criteria:

- Economy and simplicity of design,
- Efficient communication of information needed to drive the simulation,
- Easy recording of performance records.

It would also be desirable for the model to preserve maximum generality which would be helpful in generating new models [McGregor & Sykes p.222]

Graph structures have been provided for the entities in the example models in sections 9.3.2. and 9.3.3. The entities may be modelled as attributes of the object blocks for these structures, or they may be represented using independent object classes.

The first option appears to offer economy and simplicity of design.

There are also good reasons for selecting the second option, in which each entity is represented as an easily identifiable object, belonging to a class consisting of individuals with common properties and unpredictable behaviour, for which data needs to be stored. Although associated with a spatial location, the entity has independence and can move from one location, or state, to another subject to spatial constraints. The autonomy provided by representing an entity as an object gives advantages in modelling the interaction between entity classes, and in the recording of performance data.

The two approaches must be considered in relation to the three criteria for good design. With regard to model simplicity the first approach would be acceptable if there were no difficulties with driving the simulation and keeping performance records. This approach will be considered using the sample models.

9.5.2 Entity classes bonded with the graph structure

For the two example models as described in section 9.3.1, it is assumed that all entities are represented by the corresponding object blocks in the graph structures.

The space or state will have an entity identity attribute, which denotes the presence of an entity. For cars, pumps and patients this is unique at any one time, but more than one nurse could occupy a spatial block and a list would be needed in this case.

However, the state of the entity is not fully described by its presence in a spatial block. This ambiguity can only be solved by creating further attributes attached to the spatial block.

Some examples are given of where ambiguities occur:

Referring to Diagram 26 for the petrol station forecourt problem, a car standing in space 2, may be:

- being served,
- waiting for service,
- or have just completed service.

Attributes for the space are needed to indicate the identity of the car occupying that space, the pump serving it, if any, and whether the car has been served.

Referring to Diagram 29 for the nurse activity on a hospital ward problem, a patient who is not under treatment in bed-state 5, may be one of three states:

- comfortable,
- waiting for treatment but not yet acknowledged,
- waiting for a designated nurse to arrive.

Attributes for a bed-state are needed for patient identity, whether comfortable, whether a nurse is designated and the nurse identity.

The provision of further attributes proliferates detail in the spatial net object, which may be acceptable if each block is associated with one entity This occurs for stationary entities which correspond to a partitioned section of the graph structure. For example, each pump and each patient is identified with a closed cycle of blocks. The unchanging association of a unique entity with a unique set of spatial blocks enables the performance records for that entity to be part of the object's data.

It is preferable to introduce independent object classes for freely moving entities, since each spatial block in the graph structure is not associated with a unique entity. For example, all additional attributes needed to specify the state of a car will change as the car moves position. The performance records for the space and for the car would be distinctly different and separate records would be needed.

The petrol station forecourt problem modelled, with cars represented by the spatial net, has a driving mechanism in which the spatial net is dominant. The modelling concepts described in sections 9.4.3 and 9.4.4 are effective but would be more appropriate for a simpler model, in which model components react to mutual stimuli with a fixed time step [McGregor and Sykes p. 273].

In general the use of an object class to represent mobile entities is preferable, and a design to illustrate the use of independent object classes to represent entities is demonstrated for the nurse activity on a hospital ward problem in section 9.6.

9.6 Implementation of the nurse activity model

It has been shown in Chapter 8 that the generic data-driven simulation model of nurse activity on a hospital ward provides an experimental tool for comparing the effect of various ward layouts on standard measures of patient care under controlled conditions.

9.6.1 A generic data-driven model for a hospital ward

The graph structures for the prototype model described in sections 9.3.2 and 9.3.3 give an essential definition for the model and which can be specified in the data-file. The prototype model data-file may specify any number of beds arranged in one open ward or divided into smaller units. In this sense it is a data-driven generic model, which can be used to compare wards of equal capacity but different layouts.

In the nurse trail survey reported in section 8.1.4, observations were made of the movement and activities of the nurses on each hospital ward. In a similar manner the model focuses upon nurse movement and activity. This design is described in sections 9.6.2 and 9.6.3.

9.6.2 Modelling the movement of nurses

Nurses are modelled using an independent object class, the **Nurse**, referenced to the spatial net for the ward.

The information resident in the graph structure, shown in Diagram 27, section 9.3.2.2, is converted to a form which can be interpreted by the **Nurse** object. Each link representing a possible movement out of a block can be interrogated for the lists of blocks accessible via that link. This information enables the nurse object to trace a path leading to a destination block, moving from one block to another at each time step.

9.6.3 Modelling nurse activity

The ward work-load results from the combination of routine duties and the number and dependency of the patients.

It is assumed that identifiable items of work can be modelled using the object class **Task**, which could be categorised in a similar manner to that used in the nurse trail survey. **Tasks** are conceptualised as being singular, i.e. a complete item of work. In reality, routine tasks often consist of many activities, undertaken by members of the nurse team working in a co-operative fashion, but a simple model is adopted in the prototype model and it assumed that each task is undertaken by a single nurse without interruption.

The task object specifies the patient who needs treatment, where it should take place, which nurse is designated and further information relating to the nature of the task, its urgency and how long it should take.

The ward work-load will be represented by the object **Service**, which consists of the lists of tasks in existence at any one time, those in progress, assigned to a nurse or pending. Routine tasks are listed in the data-file, but others will be generated during a simulation run from patients unexpected needs for attention.

9.6.4 The object hierarchy

Thus the model consists of several object classes:

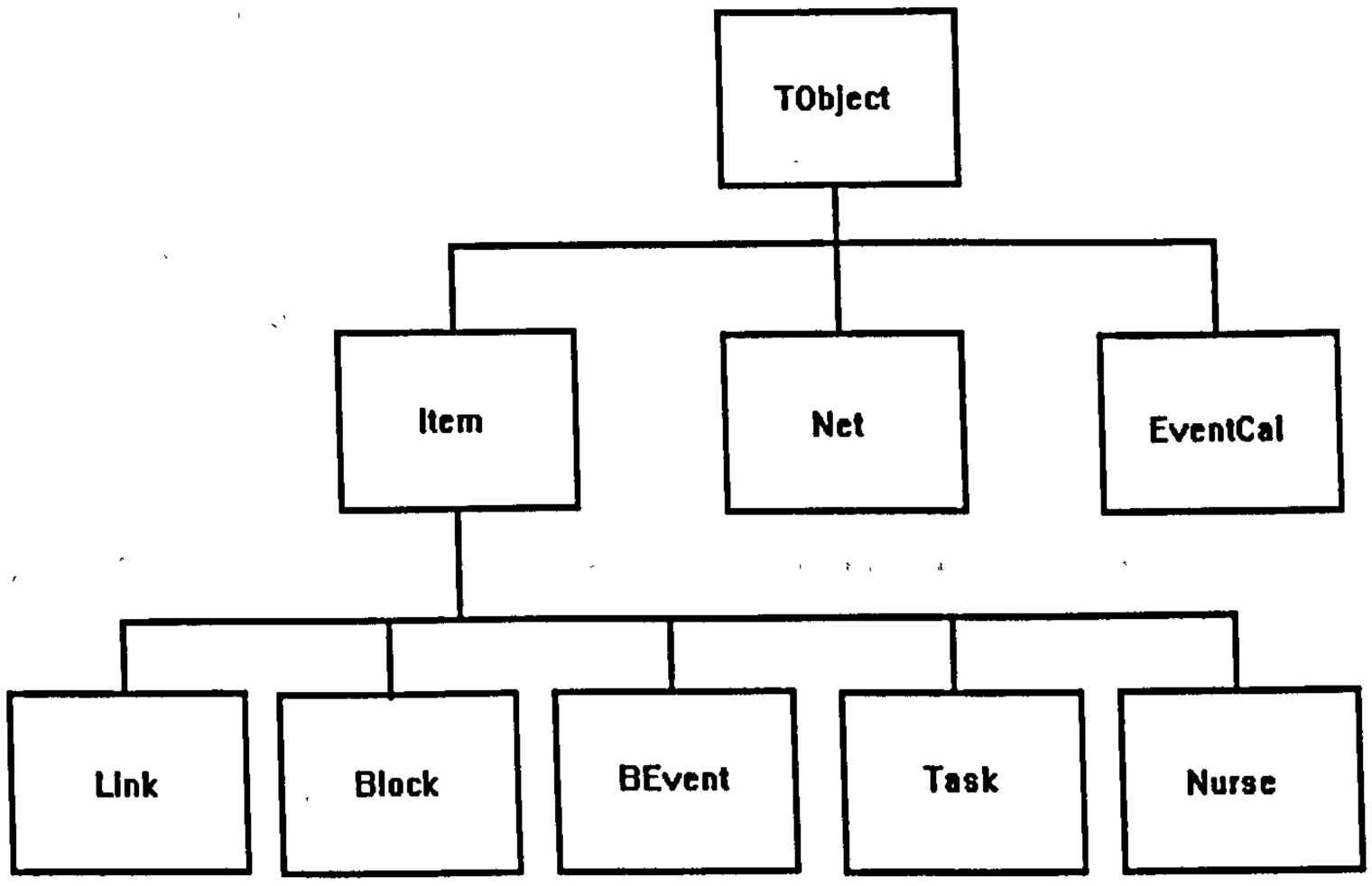
- **Nurse**, which is referenced to the spatial net for the ward layout,
- **Patient**, represented by the overlapping graph structure for the beds.
- **Task**, the work undertaken by nurses.

In addition all models require an event scheduling mechanism which requires an object representing a bound event, **Bevent.** This will specify a time when the event is due to happen and the space in which it will occur. The event calendar, the object **EventCal** has attributes which are lists of bound events.

The new objects inherit attributes and methods from those developed for the emergency evacuation model. In particular the object Patient is like the object Net and has an analogous structure. It uses an object Bed instead of Block, ListBed instead of ListBlock, Change instead of Link and ListChange instead of ListLink.

The object hierarchy is given in Diagram 32, but the objects representing patients are not explicitly included.

Diagram 32 The modified object hierarchy

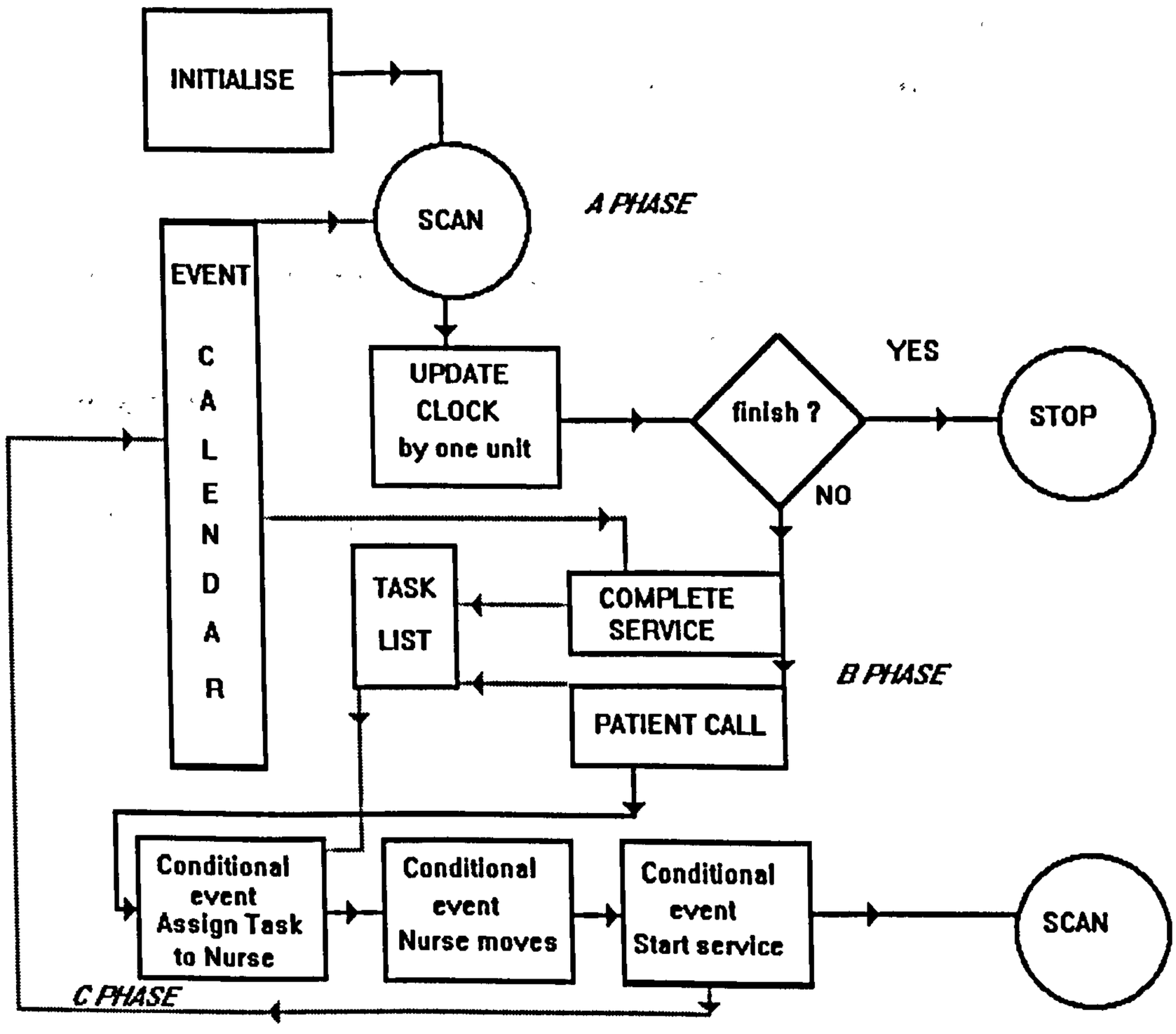


The control algorithm for the model, described in section 9.6.5, controls the clock and prompts the objects for a response. However the objects have control of their own data. The model is partially centralised in this respect [Jacobson p 225].

9.6.5 The driving mechanism for the model

The three-phase discrete-event simulation algorithm is used, as shown in Diagram 33. It is similar to the algorithm in Diagram 31 section 9.4.6, but has a fixed time-step. There is no loss of efficiency since nurses are continually moving from one location to another and events will occur at most clock times. However the choice of time-step and the dimensions of the spatial blocks need to be chosen to give an accurate representation of nurse movement. For the emergency evacuation model, as explained in Chapter 7 section 7.1.2., block dimensions of approximately 5 metres by 1.5 metres and a time-step of 5 seconds gave a satisfactory fit. Calibration of the nurse activity model, discussed in Chapter 8 section 8.4.2, gave satisfactory results using a time-step of two seconds.

Diagram 33 The control algorithm for the nurse activity model



The control algorithm deals with bound events at the 'B Phase', and conditional events at the 'C Phase' as shown in diagram 33.

The bound events are of two kinds:

1. scheduled events on the event calendar, such as the completion of activities,
2. time-dependent activities, such as a patient calling for attention.

The conditional events are of three kinds:

1. the assignment of a task to a nurse, which requires:
    - a task
    - and an available nurse,
2. a nurse moving one block, which requires:
    - a nurse who wishes to move,
3. a nurse starting treatment, which requires:
    - a nurse who is adjacent to a patient,
    - and that this patient needs treatment.

Conditional events are considered in the order given above, which will test and activate events which release resources before other events which may require them.

For example:

- a nurse who has just moved to a location can start treatment immediately,
- a nurse who has been delegated to a task can start moving towards that task at the same clock time.

The complete list of objects and methods are given in Appendix B.

9.6.6 Modelling patient calls

It is assumed that patient calls for attention are *'random'* in the same way that customer arrivals or machine breakdowns are said to be random. Patient calls follow a Markov process, in the sense that the probability of a patient call occurring during a time interval $(t, t + l)$ of length l is dependent upon the length of the time interval l but independent of the time t and events occurring before t [Cox & Smith, 1961, P6]. A well-known property of the simple Markov process is that the number of events in a unit interval follows a Poisson process and that the interval between events follows an exponential distribution. For this reason there are two statistically equivalent methods for modelling patients' calls.

One method is to sample the time which elapses between a patient becoming comfortable and needing attention, which may be assumed to follow a negative exponential distribution. In the algorithm, the execution of the bound event *'Complete Service'* will require a random sample from that distribution to calculate the scheduled time for that patient's next call. An event at that scheduled time is added to the event calendar.

Alternatively it can be assumed that each patient who is presently comfortable has a fixed probability of needing attention during the time interval, equal to one time-step. This can be modelled by taking a random sample at each clock time for a patient who is comfortable to determine whether that patient is going to call for attention at that moment, which is like modelling machine failures [Emshoff & Sisson, 1970, P16]. The probability of calling for attention may vary from one patient to another according to the level of dependency.

The second method is used since the nurse activity model is updated with a fixed time-step. The chosen probability must be adjusted according to the length of the time-step. The first method can be implemented since the model uses an event-calendar, and this approach would be preferable, i.e. more efficient, when nurses have a fixed routine and patient calls are unlikely.

9.6.7 Summary of the model implementation

Like the evacuation model, the nurse activity model is a special case of a generic model which imposes a time-updating mechanism upon a static descriptive model, but both the descriptive model and the updating mechanism are more complex.

The nurse activity model requires separate records for individual entities, which applies to both nurses and patients. To satisfy this requirement, object classes have been created for nurses and for patients, and each are referenced to graph structures representing the constraints upon movement and change, referenced to a common spatial grid. A further object class was provided for the scheduling of nursing activities.

Thus the complete descriptive model consists of records for all object classes and both graph structures. The method is not limited to two graph structures; further entity classes which may reference different graph structures can be introduced.

At each time step, changes to the evacuation model were limited to updating the numbers occupying each part of the building, but a variety of events are modelled in the nurse activity model, facilitated by the three-phase control algorithm shown in Diagram 33. The method can provide more complexity with the addition of conditional or bound events as required.

As explained in section 9.5, the model updating process has been facilitated by the use of object classes.

9.7 The practical advantages of the proposed methodology

The proposed modelling strategy has been examined in more detail in this chapter and can now be evaluated according to the guidelines set out in section 3.1.2. which were used to assess the performance of established modelling strategies in section 3.4.

9.7.1 Evaluation of the proposed methodology using the standard criteria

The three guidelines are considered in turn:

- does it facilitate model construction?

The model structure is referenced directly to the spatial layout as indicated in Diagram 14 in section 3.7, which aids the conceptualisation process. This is in agreement with VIS guidelines; that is to set up the visual display for the model before developing the model logic [Bell and O'Keefe 1987].

Using the proposed methodology, model data, structure and visual display all correspond, making development straight-forward.

- does it convey model structure clearly for verification?

The spatial constraints are self-evident in the graph structures for the model. More complex rules can be checked since the visual display for the animation will correspond with the graph structures.

- does it enable further development work, and make programming easy?

The static model is specified in the data file which is separate from the simulation engine. This enables changes to be made to the physical layout within the data and experiments can be carried out on the model. The method is appropriate for developing generic models where the same simulation can be run for similar but different problems.

9.7.2 Comparison of model structures

The methodology is compared with that used in simulations of spatial problems reported in the literature and judged, in particular, for:

- functionality, whether a calibrated model is produced which delivers appropriate performance data evaluation,
- data requirements, whether the model is economic in its requirements for survey data of the system to be modelled.

In general, a simple model is favoured. For instance, if aggregate performance measures are sufficient there will be no need to trace the path of individual entities, and these can be represented as attributes of spatial model components, which results in a macro model.

This approach was possible in the LRT congestion model [Weston & Shapiro 1987], the evacuation model described in Chapter 7; the prototype Lift Simulation model [Galpin & Rock] and several traffic simulation models reported in section 3.6.4, but a macro model could not be used for nurse activity on a hospital ward.

In simulations where entities move according to origin-destination survey data, the routes followed can be modelled in different ways. The routes for each entity can be specified as a list of locations, or random sampling can be used at branch points to assign the path taken. In the LRT congestion model the distribution of passengers along the platform was important, and observations were made of the passenger boarding and alighting positions. The data-file specified distinct routes taken by passengers in both directions, together with the proportions of passengers assigned to each route. For a more complex traffic network this model structure was rejected; separate representation of the trips made by freight haulage vehicles, given as lists of nodes passed through, gave a simple computation algorithm but was heavy on data retrieval and needed survey information which could not be obtained [Cochran & Lin 1989].

In the nurse activity model, although routes to and from locations on the ward could be specified as lists of blocks, the nurse activity trails [Appendix D] show that in practice the variety of routes taken preclude this possibility [Seelye, Ward & Walker]. Moreover, it is not necessary, since the implementation of the model uses an object class for nurse, which, at each step, retrieves the next link to suit the current destination from the spatial graph structure.

Where aggregate data is available, random samples taken from a prescribed probability distribution, over a large number of trials, can give a satisfactory model. The varying priority of movement forward from sections of corridor at junctions was modelled in this way in the evacuation model described in section 5.5.3. with satisfactory results. It was also one of the modelling approaches tried for the Arizona Freight Network Analysis in the form of unconditional probability branching at each node [Cochran & Lin]. The algorithm was easily applied and the data was available for this method, but the simulated trucks wandered aimlessly. The modified approach finally adopted for the ARNA system used random sampling for each trip at each node, conditional upon its

incoming link, to determine whether it terminated there or continued to an assigned link. This was combined with assignments of new trips from each node according to survey data percentages. The method gave a valid model and was economic on survey data.

In the nurse activity model random sampling is not required for nurse movement since the nurse object is assigned a target location. The implementation refers to attributes of the spatial link objects, which are themselves pointers to linked lists of objects. There are no data-retrieval problems and the computation is efficient.

However the unpredictability of the nurse's work sequence can be modelled by random samples being used to model patient calls, each call adding a task to the potential work of the nurse as described in 9.6.3.

The model structure for the spatial models considered in this section has been influenced by performance objectives and practical considerations such as data availability and the size and complexity of the problem. All of these factors are interrelated. In particular, the two models presented in this thesis have been shown to have appropriate functionality and data requirements.

9.7.3 Features of the proposed method

In summary the proposed method

- creates a data-driven, domain specific generic model,
- bases the visual, conceptual and formal model on a spatial grid representing the physical layout, using graph structures for the formal data structures,
- makes use of Object-Oriented Programming (OOP) to model the moving entities, which also facilitates easy generation of code for similar problems.

A data-driven generic model imposes a separation between the static model and the simulation engine. This is a well-established approach, frequently adopted for large scale network based simulations as discussed in Chapter 4, section 4.3.2. This method provides software which is efficient and easy to use, and the resulting model can be run and maintained by people without simulation skills,

The formal data-structure proposed provides a robust framework. It represents spatial and logical constraints and facilitates internal communication between different model entities. The network representation gives a good conceptual model and is capable of modelling greater complexity.

In addition the method uses OOP which has advantages for model development and software reuse.

Chapter 10 Concluding Remarks

10.1 Developments in modelling methodology

Developments in discrete event simulation modelling have been considerable, and these are documented in Chapters 2 and 3. The technique is acknowledged to be both powerful and flexible; general purpose software is available and yet simulation methods are not exploited to the full. The thesis contributes to recent efforts made to promote the technique by providing support for users who are not themselves experienced in simulation modelling.

10.1.1 Data-driven generic models

Help at the initial model building stage is most realistically provided by personal advice, but data-driven generic models are a genuine alternative. Crookes prefers to leave the client with a flexible generic model [Crookes 1987]. Pidd has developed a tool, SKIM, which helps engineers develop models for manufacturing processes [Pidd 1992a &b]. Ozdemirel & Mackulak go further and propose a module generator using a classification scheme which would help the user select an appropriate generic model [Ozdemirel & Mackulak 1993].

This type of model has been selected for development in this thesis and two are demonstrated for specific domains.

10.1.2 The evacuation model

The evacuation model is a data-driven generic model which is intended for use with any building given a data file to match that building. It is similar to the congestion model (Station Capacity Model), produced by LUL and described in Appendix C [Shapiro & Weston], and relies upon the same documented theory for modelling people's movement under congestion which is explained in Chapter 6.

The evacuation model has been demonstrated on a small building in Chapters 5 and 7, for which the data file was easily constructed from building plans and direct observation. It is anticipated that the model could be applied equally well to other buildings and that it would be a useful interactive tool.

10.1.3 The nurse activity model

The nurse activity model is a data-driven generic model developed to compare the effectiveness of nursing routines carried out in different ward layouts. Unlike the hospital design evaluation software ADAPT/ADEPT [Beattie 1973, Kenyon 1973], which produced aggregate measures, it can produce a full range of performance data for patients and nurses.

The nurse activity model is described in Chapter 8 section 8.3 and is demonstrated on a small ward. The model can be run for any ward for which data is available and will give comparable results for different layouts in which the conditions are matched. Such results are difficult to obtain from direct observation.

10.2 Model structure

10.2.1 The conceptual framework

Programming languages, such as FORTRAN and ALGOL, which were major high level languages thirty years ago, had the advantage of using subscripted arrays which provided a conceptual grid for the data structure of the model. The popularity of FORTRAN, long after alternative programming languages became available, may be explained by the acceptability of subscripted arrays for modelling static structures. The acceptability of spreadsheet modelling may also be partly due to the visible grid which is used to store data.

Both program generators and data-driven generic models provide a rigid framework to support the model definition, which may be specified by a question answer routine, or on specially prepared forms. For a problem which concerns physical space, it is natural that the data structure for the model should be spatially related, which can be specified with reference to a grid, as in the evacuation model. The thesis proposes that the two-dimensional grid is an attractive framework for conceptualising the data structure for any model, and that this format is appropriate for a general modelling strategy.

10.2.2 The modelling strategy

The general application of the spatial data structure described in Chapter 9 has advantages for modelling several types of system.

Two examples have been demonstrated:

- The evacuation model is an example of a quasi-continuous system, which is not easily modelled using a standard three-phase discrete-event algorithm without a spatially related data structure.
- The nurse activity model integrates the quasi-continuous features of the evacuation model with a representation of nurses performing their normal routine duties.

An Object Oriented design is adopted and, in both models, graph structures represent the static model. The evacuation model required only one.

Data-driven generic models are recommended, which can be implemented for any problem in the same domain, given a data-file which gives a complete description of the system to be modelled. For both models, the operating rules of the system and the initial conditions are determined within the data-file and interpreted by the control algorithm.

The general method is applicable to a wide range of spatially related problems. In Chapter 9, examples are given of how the graph structures would be designed to provide the spatial reference required by servers and customers. The petrol station forecourt problem, introduced in Chapter 3, involves spatial allocation which is explicitly modelled by means of the graph structure.

There are also systems in which the spatial components and interrelationships are more abstract, such as communication networks, and these can be modelled using an analogous data structure. The method is expected to reduce the conceptual problems usually associated with model building, since the entities defined are static and there is less ambiguity in the terminology adopted.

Thus for a whole class of problems the spatially related data structure, coupled with the control algorithm, gives an improved modelling strategy. It provides a simple model building structure without loss of efficiency.

10.2.3 Software reuse

The thesis presents a test case in software reuse. The Turbo Vision modules have been implemented, from the published software documentation, starting from a base of inexperience, to produce a working simulation model which will in its turn serve the use of others as the evacuation model

The nurse activity model makes use of the modules produced for the evacuation model to represent the spatial graph structures. The further structures required to represent the freely moving entities and abstract lists of events and tasks, use an analogous design and methods inherited from the same primitive objects. Program development requires the writing of very few sections of new code, and the inherited methods are already tested.

In the completed models, the object-oriented code enables the modules to be used as they stand, but the object hierarchy can be extended by the author or the user to provide further facilities.

Turbo Vision also provides the whole range of window presentation methods used by the system itself. These enable the model to be run within a windows environment, and is appropriate where a range of options is presented to the user.

10.3 Future developments

The creation of the data-file for a new implementation can be a large task. The listing of the links between blocks is particularly onerous. The model carries out checks on the validity of the graph structure, but an interface program which generates the data-file automatically using key-board input would make the model more accessible.

The trial runs of both models created no problems and results were obtained quickly. However the algorithm contains methods which are called at every time-step and for larger problems efficiency can be improved by amending the model so that some of these methods are executed less frequently. This is possible for methods for record collection routines and the modelling of patient calls referred to in section 9.5.6.

Various difficulties have been described concerning 'polymorphism' in section 4.4.2, and 'inheritance' in sections 5.5.2 and 9.4.1. The difficulties are due to the restrictions imposed by the object oriented design of Turbo Pascal 6, and it would be desirable to create another implementation of the model in a more flexible alternative language. The language C++, although strongly typed, allows multiple inheritance and is a possible choice. Using a strictly object-oriented language would assist in improving model design, and a comprehensive analysis using the methods of Logical Data Modelling could be carried out to ensure that each model was robust and efficient. A model destined for common use needs to be both and simulation models in particular are expected to store a large amount of data.

Finally, the method has proved successful for the models presented in the thesis. The use of a referential rectangular grid in simulation modelling will be further tested in developing models for other problem domains. This will be carried out with the help of those who are learning simulation techniques to confirm the conceptual benefits of this approach.

List of References

ALFA Attahiru Sule & NEUTS Marcel F., 1995, *Modelling Vehicular Traffic Using the Discrete Time Markovian Arrival Process*, Transportation science, Vol. 29, No. 2, May, pp 109-117.

ALTINKEMAR Kemal, 1994, *Topological Design of Ring Networks*, Computers Ops. Res., Vol. 21, No. 4. pp 421-431

ASH L. and WATERS C D J, 1991, *Simulating the Transport of Coal Across Canada-Strategic Route Planning*, J. Opl. Res.Soc. Vol. 42, No. 3, pp 195-203

AYDELLOTTE M. K., 1973, *Nurse Staffing Methodology: A review and criticism of selected literature.* D. H. E. W. Publication, No. NH 73-4333, Washington, U. S. Govt. Printing Office, V.

BAILEY N. T. J.,1954, *On Queuing Process with Bulk Service*, Journal Royal Statistical Society, B, Vol 16, No. 1, pp 80-87

BALCI Osman & NANCE Richard E., 1987, *Simulation Model Environments: A Research Prototype,* J.Opl. Res. Soc. Vol. 38, No. 8, August.

BALMER David W. & PAUL Ray J.,1986, *CASM - The right environment for simulation,* J. Opl. Res. Soc. Vol. 37, pp 443-452

BEASLEY J. E. & WHITCHURCH G., 1984, *O.R. education- a survey of young O.R. workers* J.Opl. Res. Soc. Vol. 35, pp 281-288.

BEATTIE A., 1972a, *Layout in the Accident-&-Emergency Dept.* MARU Report 2/72 PNL.

BEATTIE A., 1972b, *Methods for the Experimental Comparison of Alternative Generic Layout Types.* MARU Report 4/72 PNL.

BEATTIE Alan, 1973, *The Accident & Emergency Dept. in Harness Hospitals: experimental comparisons of medical policy and layout design options, using the interactive computer programmes 'ADAPT' and 'ADEPT',* (internal report) MARU 4/73, Medical Architecture Research Unit, Polytechnic of North London,March.

BEATTIE Alan, 1974, *Space Organization in the Accident & Emergency Department,* (internal report) MARU 4/74, Medical Architecture Research Unit, Polytechnic of North London,July.

BELL Peter C., 1985, *Visual interactive modelling in operational research: successes and oportunities,* J. Opl. Res. Soc. Vol 36, No. 11, pp 975-982.

BELL P. C. & O'KEEFE R. M., 1987, *Visual Interactive Simulation _ History, recent developments and major issues,* SIMULATION, Vol. 49, No. 3, pp 109-116

BELL Peter C., 1989, *Stochastic Visual Interactive Simulation Models,* J.Opl. Res. Soc., Vol. 40, No.7, July, pp. 615-624

BEST J., BRANSBY E. R.,CORNISH J. & SIMPSON H. M., 1968, *The Hospital Service Enquiry into the Deployment of Nursing and Midwifery Staff,* Unpublished Report. London Ministry of Health, J/G 82/32

BERGER C. R. & SHAW L., 1977, *Discrete-event simulation of freeway traffic.* In *An Overview of simulation in Highway Transportation*, pp 85-93. Society for Computer simulation.

BERLIN Geoffrey N., 1982, *A Simulation Model for Assessing Fire Safety,* Fire Technology, Vol. 18, No. 1, Feb., pp 66-75

BERTALANFFY Ludwig von, 1968, *General System Theory, Foundations Development Applications,* (text) Penguin Books Ltd.

BHATTACHARYYA S. K., ROY R. & LOW M. J., 1993, *A Computer Simulation System for the Evaluation of Man Assignments on Car Assembly Tracks,* SIMULATION Vol. 61, No. 2, Aug., pp 124-133

BOZER Yuvuz A., MELLER Russell D.and ERLEBACHER Steven J., 1994, *An Improvement-type Layout Algorithm for Single and Multiple-floor Facilities,* Management Science Vol 40, No. 7, July, pp 918-932

BRYAN John L., post 1984, *Concepts of Egress Design,* (revised) chapter 3 in section 7-34, Firesafety in Building Design and Construction, Fire Protection Handbook 16th Edition, pp 20-40, National Fire Protection Association.

BUTLER T. W., KARWAN K. R., SWEIGART J. R. & REEVES G. R., 1992, *An Integrative Model-based Approach to Hospital Layout,* IEE Transactions, Vol. 24, No. 2, pp. 144-152.

CAREY Malachy & SRINIVASAN Ashok, 1994, *Solving a class of network models for dynamic flow control*, Eur. J. Opl. Res., 75, pp 151-170

CERIC Vlatko, *Simulation Study of an Automated Guided-Vehicle System in a Yugoslav Hospital*, J. Opl. Res. Soc., Vol. 41, No. 4, pp. 299-310

CERIC Vlatko & PAUL Ray J., 1992, *Diagrammatic Representations of the Conceptual Simulation-Model for Discrete Event Systems,* Mathematics and Computers in Simulation, Vol. 34, No 3-4, pp. 317-324

CERIC V. & HLUPIC V., 1993, *Modelling a solid-waste processing system by discrete-event simulation,* J.Opl. Res. Soc., Vol. 44, No. 2, Feb., pp. 107-114

CHALMET L G et al, 1982, *Network Models for Building Evacuation,* Management Science Vol. 28, No. 1, pp86-105, Jan 1982.

CHEAH Jen Yeng & MACGREGOR SMITH, 1994, *Generalised M / G / C / C state dependent queueing models and pedestrian traffic flows,* Queueing Systems 15 pp. 365-386.

CHECKLAND Peter, 1985, *From Optimising to Learning: A Development of Systems Thinking for the 1990s,* J. Opl. Res. Soc., Vol. 36, No. 9, Sept., pp 757-767

CLEMENTSON A T, 1966, *Extended Control and Simulation Language,* Computer Journal, Vol. 9, No. 3, pp 215-220.

COCHRAN Jeffery K and LIN Li, 1989, *Application of Computer Simulation to Freight Transport Systems*, J.Opl.Res.Soc. Vol 40, No. 5 pp 433-441

COX D. R. & SMITH Walter L., 1961, *Queues,* (text) General Rditor M.S.Bartlett, Chapman and Hall, London.

CROMPTON H. M., MITCHELL H., & CAMERON J. McL., 1976, *The Aberdeen Formula.* Nurs. Times. Occ. Papers. August 26 pp. 121-124, Sept. 2 pp. 125-128.

CROOKES J. C., 1982, *Simulation in 1981,* Eur. J. Opl. Res. Vol. 9, No. 1, pp 1-7.

CROOKES John & VALENTINE, 1982, *Simulation on micro-computers,* J.Opl. Res. Soc. Vol. 33, pp 855-858.

CROOKES John G., BALMER David W., SEW TEE CHEW & PAUL Ray J., 1986, *A Three-Phase Simulation Written in Pascal*, J. Opl. Res. Soc., Vol. 37, No. 6, pp 603-618

CROOKES John G., 1987, *Generators, Generic Models & Methodology,* J. Opl. Res. Soc., Vol. 38, No. 8, Aug., pp.765-768

CURTISS J. H. et al, 1951, *Monte Carlo method,* National Bureau of Standards Applied Mathematics Series, 12.

DAI J G, NGUYEN Vien & REIMAN Martin, 1994, *An Approximation Method for Generalised Jackson Networks*, Operations Research, Vol. 42, No. 1, Jan.-Feb., pp. 119-136

DANIELSEN Paul, ELDRIDGE David & BROWN Steven, 1991, *Visually-interactive discrete-event simulation helps planning on the Stanlow and Tranmere Waterfronts,* OR Insight, Vol 4, issue 1, Jan.-March, pp. 6-14

DOMSCHKE Wolfgang & DREXI Andreas, *Location and Layout Planning, An International Bibliography,* Lecture Notes in Economics and Mathematical Systems, (text) Springer-Verlag Berlin Heidelberg NewYork Tokyo

DREZNER Z & WESOLOWSKY G. O., 1995, *Location on a one-way rectilinear grid,* J. Opl. Res. Soc., Vol. 46, No 6, pp 735-746

EDGE Chris & KLEIN Jonathan, 1993, *No Job too small? A questionnaire survey indicates the kind of use that small businesses make of standard operational research techniques,* OR Insight, Vol. 6, Issue 3, July-Sept., pp.8-12

EIGER A, NIEDOWSKI R. & ERIKSON D., 1983, *Large scale traffic system simulation: an approach of the TRAFLO model*, I.T.E. Journal, pp 22-28.

ELDREDGE David L., MCGREGOR John D. & SUMMERS Marguerite K., 1990, *Applying the object-oriented paradigm to discrete event simulations using the C++ language,* SIMULATION, Feb., pp 83-91

EMSHOFF James R. and SISSON Roger L., 1970, *Design and Use of Computer Simulation Models*,(text) The Macmillan Co.

EVA Malcolm, 1992, *SSADM Version 4: A User's Guide,* (text) The McGraw-Hill International Series in Software Engineering.

EVANS John B., 1988, *Structures of Discrete Event Simulation* (text) Ellis Horwood Series in Artificial Intelligence

FISHWICK Paul A., 1992, *An Integrated Approach to System Modeling Using a Synthesis of Artificial Intelligence, Software Engineering and Simulation Methodologies,* ACM Transactions on Modeling and Computer Simulation, Vol 2, No. 4, October 1992 pp 307-330

FLYNN D. J. & FRAGASO-DIAZ O., 1993, *Conceptual EuroModelling: how do SSADM and MERISE compare?,* Eur. J. Inf. Systs. Vol. 2, No. 3, July, pp 169-193

FRANCIS R. L., McGINNIS L. F. Jr & WHITE J. A., 1992, *Facility Layout and Location: an Analytical Approach,* (text) 2nd Edition. Prentice Hall.

FREEMAN Jim, 1993, *Spreadsheet gaming and management skills development,* OR Insight, Vol. 6, Issue 1, Jan.-Mar., pp. 9-14

FRUIN J. J., 1971a, *Designing for pedestrians: a level -of-service concept,* Highway Research Record no 355.

FRUIN J. J., 1971b, *Pedestrian planning and design,* New York, Metropolitan Association of Urban Designers and Environmental Planners.

FUTO Ivan & GERGELY Tamas, 1990, *Artificial Intelligence in Simulation* (text) Ellis Horwood Series in Artificial Intelligence.

GALBREATH M., 1969, *Time of Evacuation by Stairs in High Buildings Ontario Fire Marshall,* Vol. 5, No. 2, First Quarter.

GALPIN V. C. & ROCK S. T.,1995, *A Lift Simulation Prototype,* Software-Practice and Experience, Vol. 25 (3) Mar., pp 251-270

GILCHRIST R. 1982, *An Analysis of Continuous Proportions,* COMPSTAT Physica-Verlag, Vienna for IASC (International Association for Statistical Computing) pp 236-241

HALL A., 1873, *On an experimental determination of* π Messeng. Math. 2, pp113-114.

HELLINGA Bruce & VAN AERDE Michael, 1994, *An overview of a simulation study of the Highway 401 freeway traffic management system,* Canadian Journal of Civil Engineering, Vol. 21, No. 3, June, pp 439-454

HENZ LUEHRMANN Monica & BYKETT Donald L., 1989, *A Pilot study of the impact of animation on decision making,* SIMULATION Oct., pp 153-158

HLUPIC Vlatka & PAUL Ray J., 1994, *Simulation Modelling of Flexible Manufacturing Systems Using Activity Cycle Diagrams,* J. Opl. Res. Soc., Vol 45, No 9, pp. 1011-1023.

HOARE R. T. & WILLIS R. J., 1992, *A case study of animated computer simulation in the Australian mining industry,* J.Opl. Res. Soc., Vol. 43, Dec., pp1113-1120

HOLLOCKS Brian, 1983, *Simulation and the Micro,* J.Opl. Res. Soc. Vol 34, No.4, April., pp. 331-343

HOLLOCKS Brian, 1992, *A well-kept secret?,* OR Insight, Vol. 5, Issue 4, Oct.-Dec., pp. 12-17

HORROCKS Rod (editor), 1971, *The Simulation Study Group Simulation in UK Manufacturing Industry* The Management Consultancy Group Barclays Venture Centre, University of Warwick Science Park

HOSSAIN Md. Al-Amin & TOBIAS Andrew, 1991, *WITNESS in the hands of an expert system* OR Insight, Vol. 4, issue 3, July-Sept., pp. 10-14

HOWE D. R., 1983, *Data Analysis for Data Base Design,* (text) Edward Arnold.

HU Y. C. & SCHONFELD, 1984 *Simulation and optimization of regional road network,* Journal of Transportation Engineering, vol. 110, pp 431-443.

HURRION R. D. & SECKER R. J. R., 1978, *Visual Interactive Simulation an Aid to Decision Making,* Omega 6, pp. 419-426.

HURRION Robert H., 1978, *An Investigation of Visual Interactive Simulation Methods Using the Job-Shop Scheduling Problem,* J.Opl. Res. Soc., Vol. 29, No.11, Nov., pp 1085-1094

HUSEBY R. D., 1969, *Impact of Nursing Design on Patients' Opinion,* Unpublished Ph.D Thesis. University of Minnesota. (Summary in Aydellote, M. K.)

HUTCHINGS A. R., 1990, *Introduction to Entity-relationship Modelling for OR Analysts,* J.Opl. Res. Soc., Vol. 41, No. 3. pp 191-200.

JACOBSON Ivar, 1992, *Object-Oriented Software Engineering, A Use Case Driven Approach,* (text) Addison Wesley.

JANSON Bruce N., 1995, Network Design Effects of dynamic Traffic Assignment, Journal of Transportation Engineering, Vol. 121, No. 1, Jan-Feb, pp. 1-13

JOHNSON M Eric & BRANDEAU Margaret L, 1993, *An Analytic Model for Design of a Multivehicle Automated Guided Vehicle System*, Management Science, Vol. 39, No. 12, Dec.

JONES Christopher B, KIDNER David B and WARE J Mark, 1994, *The Implicit Triangulated Irregular Network and Multiscale Spatial Databases,* The Computer Journal, Vol 37, No. 1, pp 43-57

JOSEPH B. M.,1970, *Planning and Design of Best-Buy Hospitals and Development of a Harness Design for District General Hospitals.* Hosp. Eng. 1970, June, pp 132-144.

KENYON B.,1973, *The ADAPT/ADEPT Suite of Programmes.* MARU Report 2/73 PNL.

KENYON Bob, 1975, *Simulating Hospital Departments,* MARU 2/75, PNL.

KHASNABIS Snehamay & HEIMBACH Clinton L., 1977, *Traffic Simulation as Highway Design Tool,* Transportation Engineering Journal, May 1977 pp 369-384.

KISKO T. M. & FRANCIS R. L., 1984, *Network Models of Building Evacuation: Development of Software Systems,* NBS-GCR-84-457, National Bureau of Standards, Gaithersburg MD.

LESTOR Kit, 1990, *A Practical Approach to Data Structures,* (text) Ellis Horwood Series in Artificial Intelligence.

LEWIS P. J., 1992, *Rich picture building in the soft systems methodology*, Eur. J. Inf. Systs. Vol. 1, No. 5, pp 351-360, May.

LIN James T. & LEE Chia-Chu, 1993, *A Three-phase Discrete Event Simulation with EPNSim Graphs,* SIMULATION Vol. 60, No. 6, June., pp 382-392

LIPPERT S., 1971, *Travel in Nursing Units,* Human Factors, 13, 3 pp. 269-282.

LONDON TRANSPORT BOARD, 1958, *Second report of the Operational Research Team on the Capacity of Footways,* Report No 95 LTB.

MACGREGOR SMITH J., 1994, *Application of State-dependent Queues to Pedestrian/vehicular Network Design*, Operations Research Vol. 42, No. 3, May-June, pp. 414-427

MAKIGAMI Yusuji, NAKANISHI Tsunehiko, & SEILL Kim, 1984, *Simulation Model Applied to Japanese Expressway,* Jnl. of Transportation Engineering, Vol. 110, pp. 94-111

MAIDEN Neil A. & SUTCLIFFE Alistair G., 1992, *Exploiting Reusable Specifications Through Analogy CASE,* Commun. ACM. Vol. 35, No. 4, pp. 55-64

MARRALL John, MILLER Jr Edward, SMITH Gerald, FEUERSTEIN John & YAZDAN Fred, *Planning and Design of Passing Lanes using Simulation Model*, Journal of Transportation Engineering Jan/Feb 1995, Vol. 121, No. 1, pp. 50-62

MARTIN Peter T., 1993, *Can Operational Research Tackle Traffic Congestion?* OR Insight Vol. 6 Issue 1, Jan.-Mar., pp 4-8

MATHEWSON Stephen C., 1974, *Simulation program generators,* SIMULATION Vol. 23, Dec., pp 181-189

MATHEWSON S. C., 1985, *Simulation program generators: code and animation on a P.C.,* J. Opl. Res. Soc., Vol. 36, No. 7, July, pp. 583-5899

MATHEWSON S. C., 1987, *DRAFT/DRAW/SSIM- an integrated network based toolkit for simulation,* paper given at UK Simulation Conference, Bangor.

MCGREGOR John D. & SYKES David A., 1992, *Object-Oriented Software Development: Engineering Software for Reuse,* (text) Van Nostrand Reinhold New York.

MELINEK S. J. & BOOTH S., 1975, *An Analysis of Evacuation Times & the Movement of Crowds in Buildings*, CP 96175 Building Research Establishment, (Fire Research Station).

NIGHTINGALE F., 1863, *Notes on Hospitals.* (text) London-Longman, Green, Longman Roberts, and Green, p. 50.

NUFFIELD PROVINCIAL HOSPITALS TRUST, 1955, *Studies in the Functions and Design of Hospitals.* London. Oxford University Press.

NUFFIELD PROVINCIAL HOSPITALS TRUST, 1955, *Studies in the Function and Design of Hospitals.* Oxford University Press.

O'KEEFE Robert M. & HADDOCK Jorge, 1991, *Data driven generic simulation for flexible manufacturing systems*, International Journal of Production Research, Vol. 29, No. 9, pp 1795-1810

O'LEARY Timothy & GRATZ Jerre M., 1982, *An Analysis of Fire Evacuation Procedures Using Simulation,* Fire Journal, Vol. 76, No. 3, May, pp. 119-121

OREN Tuncer I. & ZEIGLER Bernard P., 1979, *Concepts for advanced simulation methodologies,* SIMULATION Vol. 32, part 3, Mar., pp 69-82

OVERSTREET C. Michael & NANCE Richard E., 1985, *A Specificatiion Language to Assist in Analysis of Discrete Event Simulation Models*, Commun. ACM, Vol. 28, No. 2, Feb., pp 190-201

OXFORD R. H. B.,1970 *The Evaluation of a Deep Ward Plan.* Oxford Regional Hospital Board.

OZDEMIREL Nur E. & MACKULAK Gerald T., 1993, *A Generic Simulation Module Architecture Based on Clustering Group technology Model Codings,* SIMULATION Vol. 60, No. 6, June, pp 421-433

OZEL Feliz, 1992, *Simulation Modeling of human behaviour in buildings,* SIMULATION Vol. 58, No. 6, June, pp. 377-384

PACE A. J., & GRIMSHAW E.,1978, *Regional Nurse Staffing.* Nurs. Times, Occ Papers. 74 No. 25 pp. 101-104, No. 26 pp.105-108.

PAUL R. & CHEW S., 1987, *Simulation Modelling using an Interactive Simulation Program generator,* J.Opl. Res. Soc.,Vol. 38, No. 8, Aug., pp 735-752

PAUL Ray J., 1991, *Recent developments in simulation modelling,* J.Opl. Res. Soc.,Vol. 42, No. 3, Mar., pp.217-226

PAULS J. L., 1977, *Movement of People in Building Evacuations, Human Response to Tall Buildings,* Dowden Hutchinson & Ross Inc. Stroudsurg P.A.

PELLETIER R. J. & THOMPSON J. D., 1960, *Yale Index Measures Design Efficiency.* The Modern Hospital, Vol 95, No. 5. pp. 73-77.

PETERSON James L., 1977, *Petri Nets,* ACM Computing Surveys Vol. 9, No. 3, Sept., pp. 223-252

PIDD Michael, 1987, *Simulating Automated Food Plants :Current Simulation Research,* J.Opl. Res. Soc.,Vol. 38, No. 8, Aug., pp 683-692

PIDD M., 1992, *SKIM: a simulation sketchpad for plant design,* J.Opl. Res. Soc.,Vol. 43, No. 12, Dec., pp. 1121-1133

PIDD M. & TUNNICLIFFE-WILSON J., 1992, *Implementing a computer aided production management system,* Eur. J. Inf. Systs. Vol. 1, No. 6, Sept., pp. 409-415

PIDD Michael, 1992, *Guidelines for the design of data driven generic simulators for specific domains,* SIMULATION ,Oct., pp. 237-243

PIDD Michael, 1995, *Object-orientation, Discrete Simulation and the Three-Phase Approach,* J. Opl. Res. Soc.,Vol. 46, pp. 362-374.

PINKOWSKI Ben, 1989, *Clustert- A simulation-based expert,* SIMULATION, May, pp. 179-185

POOLE T. & SZYMANKIEWICZ J., 1977, *Using simulation to solve problems,* (text) McGraw-Hill Book Company (UK) Limited.

POPE James A., RAKES Terry R., REES Loren Paul & CROUCH Ingrid W.M., 1995, *A Network Simulation of High-congestion Road-traffic Flows in Cities with Marine Container Terminals*, J. Opl. Res. Soc., Vol. 46, pp 1090-1101.

PRINCE Frank A., 1994, *The Paradigm Shift Process,* Creativity and Innovation Management, Vol. 3 No. 1, Mar., pp 29-32

PRITSKER A. Alan B., 1974, *The GASP IV simulation language*, (text) Wiley - Interscience.

PRITSKER A. Alan B. & PEGDEN Claude Dennis, 1979, *Introduction to simulation and SLAM*, (text) Wiley.

PRZASNYSKI Z. H., 1989, *On Using Spreadsheets to Model Decision Problems,* Computer Educ. Vol. 13, No. 2, pp 117-128

PRZASNYSKI Zbigniew H., 1990, *Spreadsheet Simulation Applications in Operations Management,* Microcomputer Applications, Vol. 9, No. 1

RADFORD R.,1971, *Systems, Cybernetics, Computers and Health Buildings.* Computer Aided Design 1971, Autumn, pp 16-22.

RAWLINSON Carole & DOIDGE Charles, 1971, *Dynamic Space Allocation,* Architectural Research and Teaching, Vol. 1, No. 3, April, pp. 4-10

RAWLINSON Carole, 1975, *The Development of a Classification Framework for Selecting Case Study Hospitals,* (internal report) MARU 2/76, Medical Architecture Research Unit, Polytechnic of North London. Dec.

RICHARDSON George P. & PUGH, III Alexander L., 1981, *Introduction to Systems Dynamics with DYNAMO* , (text) MIT Press.

RIVETT B. H. P., 1983, *Professor K. D. Tocher,* J. Opl. Res. Soc., Vol. 34, No. 4, April.

SEELYE Alan, WARD Sylvia & WALKER Jean, 1981, *Ward Layout and Nurse Staffing - A report on Three Case Study Hospitals,* Unpublished report DHSS Commission No H78/R/A/614A, MARU Polytechnic of North London.

SEELYE Alan, 1982, *Ward Layout and Nurse Staffing - Nurse activity patterns and nurse-patient see and hear contact analyses,* Unpublished Working Paper 1/82, MARU Polytechnic of North London.

SHANNON Robert E., 1975, *Systems Simulation: The Art and Science,* (text) Prentice Hall.

SHAPIRO Janet, 1976, *Report on the Simulation of the Accident and Emergency Department of Kings College Hospital,* (internal report) PNL.

SHEARN D. C. S., 1990, *PASSIM - A Pascal discrete event simulation program generator,* SIMULATION July, pp. 31-38

SIIKONEN Marja-Liisa, 1993, *Elevator Traffic Simulation*, SIMULATION, Vol. 61, No. 4, Oct., pp 257-267

SON Yung Tae, CASSIDY MIchael J. & MODONAT Samer M, 1995, *Evaluating Steady-state Assumption for Highway Queueing System*, Journal of Transportation Engineering, Mar/April Vol. 121, No. 2, pp 182-190.

SMITH J MacGregor, 1982, *An Analytical Queing Network Computer Program for the Optimal Egress Problem,* Fire Technology, Vol. 18, No. 1, Feb., pp 18-33

STAHL Fred, 1982, *FiresII: A Behaviour Based Computer Simulation of Emergency Egress During Fires,* Fire Technology, Vol.18, Feb., pp 49-65

STEELE Will, 1991, *Taking the 'O' out of OR, Model Building Made Easy* AT & T Istel Limited, paper presented at The Operational Research Society National Event, Discrete Event Simulation, 19th Nov.

STILL Keith, 1992, *The Lemming Factor,* Health & Safety at Work, Dec., p. 23

STILL Keith, 1993, *New computer system can predict human behavioural response to building fires, (VEGAS),* FIRE Jan., pp. 40, 42

STURDAVANT M., 1960, *Intensive Nursing Service in Circular and Rectangular Units compared.* Hospitals, 34, 14 pp. 46-48, 71-78.

TANK Arvind, 1992, *Evaluation of Various Simulation Software through Petrol Station Forecourt Models,* final year project for BSc Honours Degree in Business and Computing, PNL.

THUNHURST C.,1971, *An Axiomatic Hospital Department Design Model applied to an OPD.* MARU Report 3/71, PNL

THUNHURST Colin, 1974, *Quantitative Analysis and the Planning of the Outpatient Department,* (internal report) MARU 3/74, Medical Architecture Research Unit, PNL London.

TOCHER Keith Douglas, 1963, *The Art of Simulation* (text) English Universities Press.

TOCHER K. D., 1964, *Some Techniques of Model Building,* in Proceedings of IBM Scientific Computing Symposium on Simulation Models and Gaming, New York, pp 119-155.

TORN Aimo A., 1981, *Simulation Graphs: A General Tool for Modelling Simulation Designs,* SIMULATION, Vol. 37, pp. 187-194.

TORN Aimo A., 1985, *Simulation nets, a simulation modeling and validation tool,* SIMULATION, Vol. 45, part 2, Aug., pp. 71-75

TREGENZA Peter, 1976, *The Design of Interior Circulation,* (text) Crosby Lockwood Staples London.

TRITES D. K., GALBRAITH F. D., STURDAVANT M., & LECKWART J. L.,1969, *Radial Nursing Units Prove Best in a Controlled Study.* Modern Hospital, April, 112, 4, pp. 94-99.

VIMALASEGARAM K., 1979, *Programming to Produce Tables Related to the Design of the Ward.* Polytechnic of North London. Department of Mathematics.

WALKER Jean, 1981, *Ward Layout and Nurse Staffing - A Review of the Literature.* Unpublished report, DHSS Commission No : H78/R/A/6/4A, MARU Polytechnic of North London

WESSEX R. H. B., O. & M. WORK STUDY DEPT. *Measurement and Quality Assessment of the Work of Three Ward Teams.* Poole General Hospital, Winchester, Wessex Regional Hospital Board.

WESTON J. G., 1988, *Victoria Congestion Signing: Monitoring,* Operational Research Note 88/45, London Regional Transport.

WESTON J. G. & MARSHALL J., 1973, *Pedestrian Movement in Crossing Flows* , Transportation Planning and Technology, Vol. 1, pp 49-54.

WESTON J. G. & SHAPIRO J., 1987, *Analysis of Passenger Congestion at Angel,* Operational Research Note 87/23, December, London Regional Transport.

WESTON J. G. & SMITH R. G., 1987, *Analysis of Passenger Congestion at Liverpool Street (Central Line),* Operational Research Note 87/12, October, London Regional Transport.

WHITT Ward, 1993, *Large Fluctuations in a Deterministic Multiclass Network of Queues*, Management Science Vol. 39, No. 8, Aug., p.1020

WIENER N., 1961, *Cybernetics,* (text) MIT Press Cambridge, Mass. 1948 (enlarged edition 1961) Wiley New York.

WILLIAMS T. M., GITTINS R. P. & BURKE D. M., 1989, *Replenishment at Sea,* J.Opl. Res. Soc.,Vol. 40, No. 10, pp. 881-887

WILLIAMS T. M., 1992, *Heuristic Scheduling of Ship Replenishment at Sea,* J.Opl. Res. Soc.,Vol. 43, No. 1, Jan., pp. 11-18

WITHERS Stephen J. & HURRION Robert D., 1982, *The interactive development of visual simulation models.* J.Opl. Res. Soc.,Vol. 33, No. 11, Nov.

WOOD P., 1988, *Effect of Platform Management on Station Stoptimes,* Operational Research Note 88/43, October, London Regional Transport.

Computer Software

ECSL

CLE.COM Ltd.,1982, *ECSL User Manual,* Clementson A.T., 8, Stanley Road, King's Heath, Birmingham, B14 7NB.

TURBO PASCAL 6.0

BORLAND International, 1990, *Turbo Pascal Version 6.0, Turbo Vision Guide,* (text) Corporate Headquarters: 1800 Green Hills Road, P.O. Box 660001, Scotts Valley, CA 95067-0001.

VS7

SYSPACK LTD, 1990,*Syspack vs7 user guide*, Chew See Tee,149 Victoria Park Road, London E9 7JZ.

WITNESS

AT&T ISTEL, 1993, *WITNESS*, Visual Interactive Systems Ltd., Highfield House, Headless Cross Drive, Reddithch, Worcs., B97 5EQ.

Bibliography

The following texts and papers have been referred to.

BERTALANFFY Ludwig von, 1968, *General System Theory, Foundations Development Applications,* (text) Penguin Books Ltd.

BREEDAM Alex Van, RAES Jan and VAN DE VELDE Karel, 1990, *Segmenting the simulation software market,* OR Insight, Vol. 3, No. 2, April-June, pp. 9-13

BREUER Melvin A., 1977, (editor) *Digital System Design Automation: Languages, Simulation & Data Base,* (text) Pitman.

CHECKLAND Peter, 1981, *Systems Thinking Systems Practice,* (text) John Wiley & Son.

CONOLLY Brian, 1975, *Lecture Notes on Queueing Systems,* (text)Ellis Horwood Ltd.

CONOLLY Brian, 1981, *Techniques in OR Vol 2 Models, Search and Randomisation,* (text) Ellis Horwood Ltd.

DOUKIDIS G., 1987, *An Anthology on the Homology of Simulation with A I :Current Simulation Research,* J. Opl. Res. Soc., Vol. 38, No. 8, August.

DOUKIDIS G. & PAUL R., 1985, *Research in Expert Systems to Aid Simulation Model Formation,* J.Opl. Res. Soc., Vol. 36, No. 4, April

DOUKIDIS Georgios, 1989, *The Expert systems product: The symbiosis with OR,* OR Insight, Vol. 2, issue 4, Oct-Dec.

EMSHOFF James R. and SISSON Roger L., 1970, *Design and Use of Computer Simulation Models*,(text) The Macmillan Co.

EVA Malcolm, *SSADM Version 4: A User's Guide,* (text) The McGraw-Hill International Series in Software Engineering.

EVANS John B., 1988, *Structures of Discrete Event Simulation,* (text) Ellis Horwood Series in Artificial Intelligence.

FLITMAN & HURRION R. D., 1987, *Linking Discrete-Event Simulation Models with Expert systems,* J. Opl. Res. Soc., Vol. 38, No. 8, August

FUTO Ivan & GERGELY Tamas, 1990, *Artificial Intelligence in Simulation,* (text) Ellis Horwood Series in Artificial Intelligence.

HAMMERSLEY J. M. & HANDSCOMB D. C., 1964, *Monte Carlo Methods*, (text) Methuen Monographs.

HORROCKS Rod (editor), 1991, *The Simulation Study Group Simulation in UK Manufacturing Industry* The Management Consultancy Group Barclays Venture Centre, University of Warwick Science Park, September.

HOWE D. R., 1983, *Data Analysis for Data Base Design,*(text) Edward Arnold.

JACOBSON Ivar, 1992, *Object-Oriented Software Engineering, A Use Case Driven Approach,* (text) Addison-Wesley Publishing Company.

KAC Mark, 1964, *Probability*, The Scientific American, 1964 Sept. pp.92-108.

KREUTZER Wolfgang, 1986, *System Simulation. Programming Styles & Languages,*(text) Addison-Wesley.

KRUSE Robert L., 1984, Data Structures & Program Design, Prentice-Hall.

LESTOR Kit, 1990, *A Practical Approach to Data Structures,* (text) Ellis Horwood Series in Artificial Intelligence.

MCGREGOR John D. & SYKES David A. 1992, *Object-Oriented Software Development: Engineering Software for Reuse,* (text) Van Nostrand Reinhold New York.

O'BRIEN Stephen, 1991, *Turbo Pascal 6, The Complete Reference (text)* Borland-Osbourne/McGraw-Hill.

PIDD M., 1984, *Computer Simulation in Computer Science,* (text) Wiley, Chichester.

PIDD Mike, 1989, *Choosing discrete simulation software: Useful features to look for and what to ask the salesman,* OR Insight, Vol. 2, Issue 3, July-Sept.

POOLE T. & SZYMANKIEWICZ J., 1977, *Using simulation to solve problems* (text) McGraw-Hill Book Company (UK) Limited.

SCHOEMAKER S., (editor) *Computer Networks & Simulation,* (text) North Holland Publishing Co.

SHANNON Robert E., 1975, Systems *Simulation: The Art and Science,* (text) Prentice Hall.

SIMON H. A., 1981, *The Sciences of the Artificial,* (text) (2nd edition) MIT Press Cambridge Mass.

STEWART Ian, 1989, *Does God Play Dice: The New Mathematics of Chaos* (text) Penguin Books.

TOCHER Keith Douglas, 1963, *The Art of Simulation,* (text) English Universities Press.

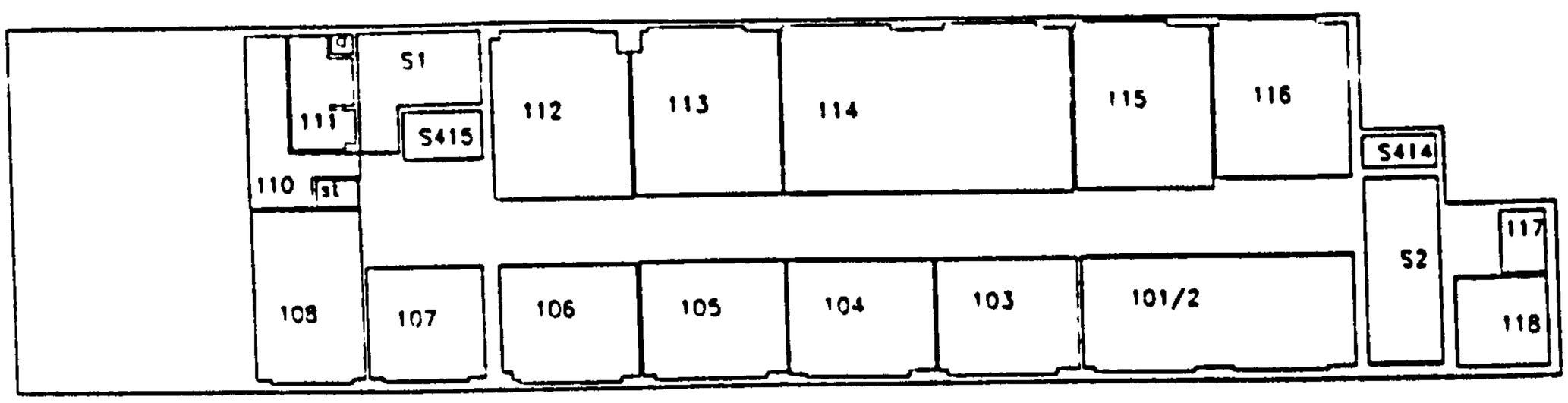
TREGENZA Peter, 1976, *The Design of Interior Circulation,* (text) Crosby Lockwood Staples London.

ZEIGLER B. P., 1984, *Multifacetted Modelling & Discrete Event Simulation,* (text) Academic New York.

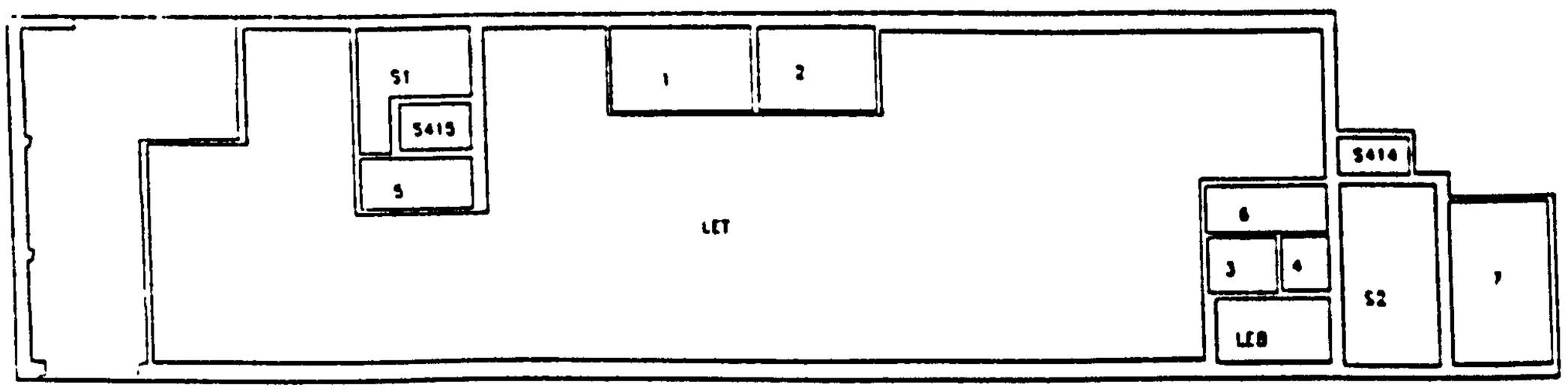
Appendix A

Floor layouts for the Eden Grove building University of North London

Ground floor and first floor

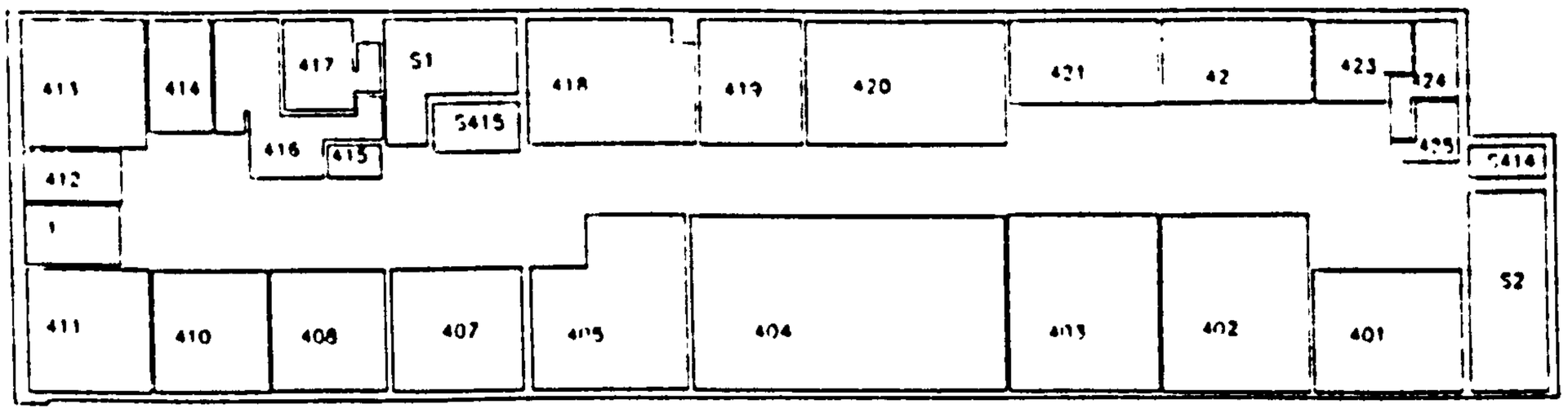


PLAN: 1 [FIRST]

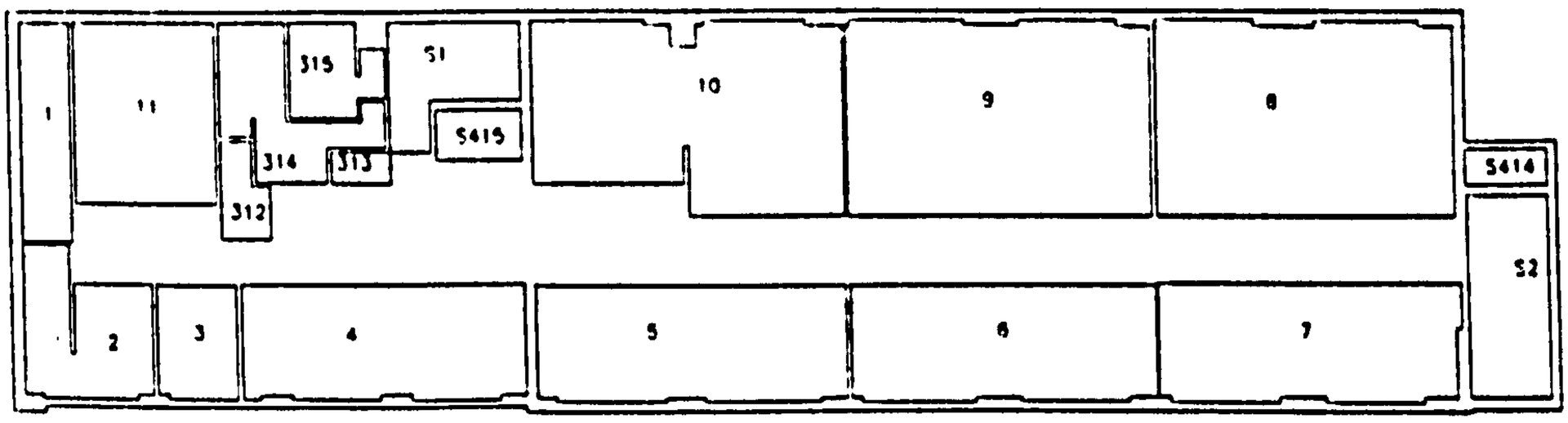


PLAN: 1 [GROUND]

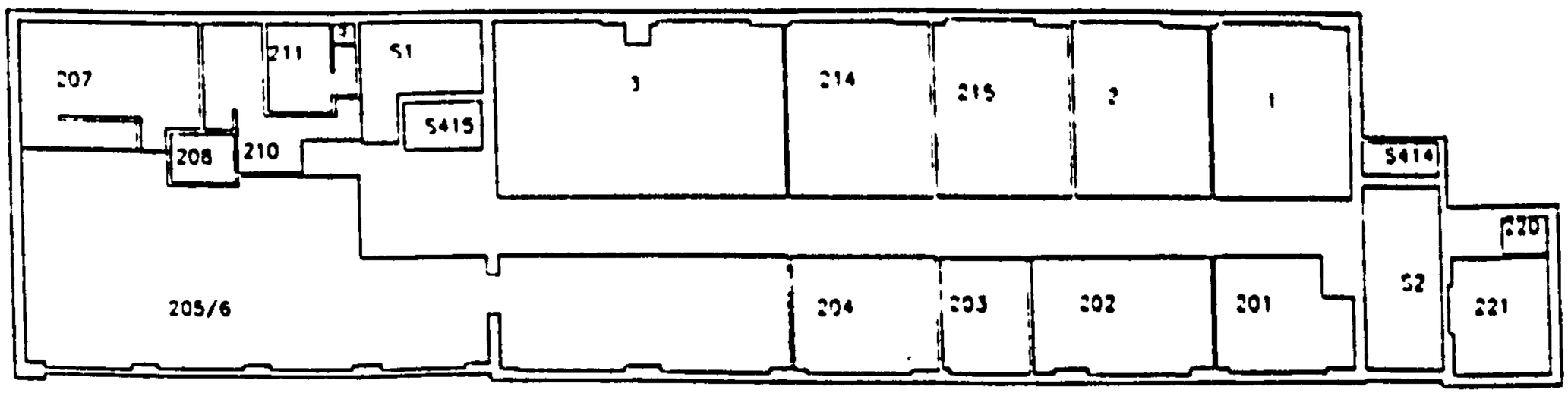
Second, third and fourth floors.



PLAN 1 (FOURTH)



PLAN: 1 (THIRD)



PLAN: 1 (SECOND)

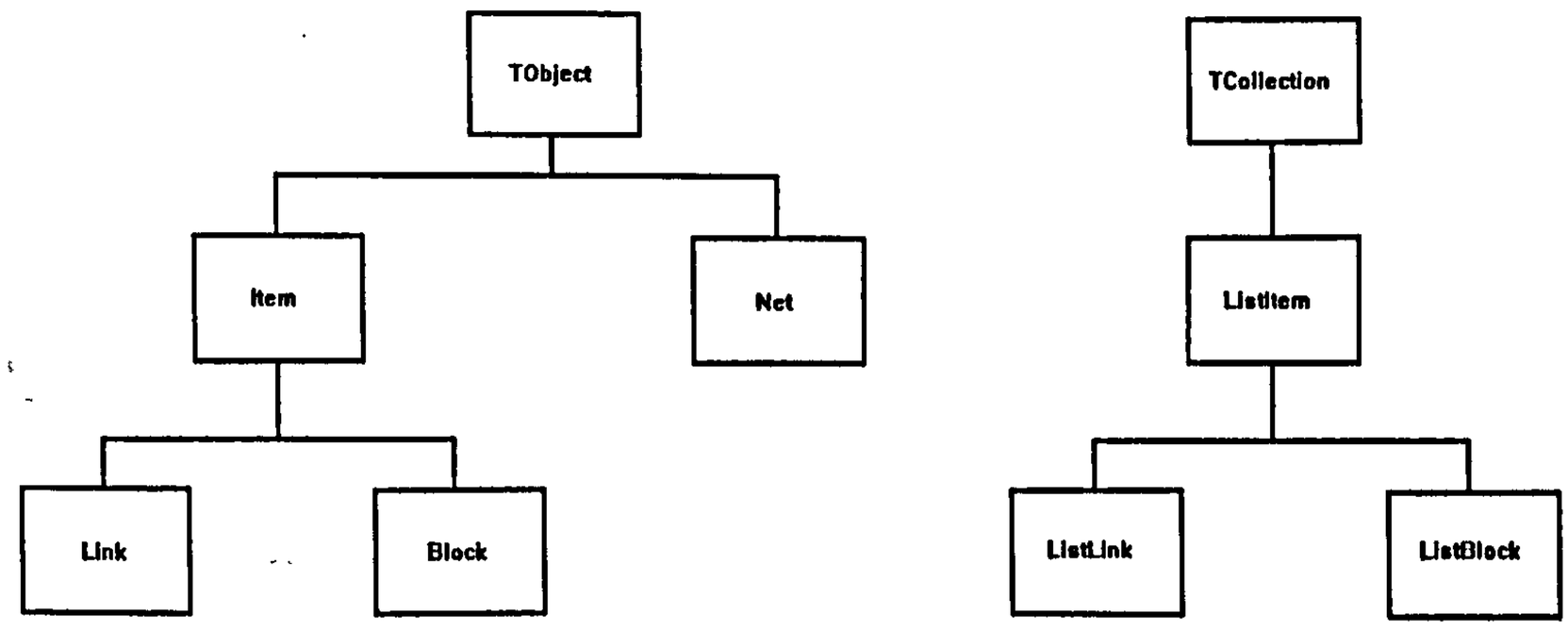
Appendix B

CONTENTS

B1. Documentation of the Evacuation Model

The evacuation model uses TVision objects, with the hierarchy described in Diagram 18, in section 5.3.2.

The object hierarchy



Note that TCollection is an abstract type for implementing any collection of items, including other objects, a more general concept than the traditional array, set or list. Utility methods are provided for TCollection.

The object **Net** is defined to represent the graph structure. It has fields which represent the list of blocks and the list of links, and other lists of blocks which are used to partition the graph structure.

Table I gives the fields for Item and the two descendants Link and Block.

Table I Objects and their fields

Field	type	Item	Link	Block
• id	integer	identity	identity	identity
• capacity	integer		maximum flow	maximum capacity
• bid[1] • bid[2] • bid[3]	integer array	location	block before block after door or not door	location (x) location (y) location (floor)
• linktype • blocktype	string		door/corr/doorout	room/corr/stairs/exit
adjacent[1] adjacent[2]	block pointers		to block before to block after	
ls[1] ls[2]	pointers (lists of links)			to following links to preceding links
goesto comesfrom	pointers (lists of blocks)			to blocks following to blocks before
number	integer	index	index	Fulkerson's number
flag	integer			check flag
dry	Boolean			not visited
• num	integer			occupancy
tmp_num tmp_spare poss_move act_move	integer			temporary variables used by block method update

Fields indicated with bullets are given as data. The initial value of the number occupying each block is given as data.

The objects' methods

Methods of the objects are used to:

- build the graph structure,
- check whether it cycles,
- update the states of blocks as clock is updated,
- check whether the building is empty.

Building the graph structure

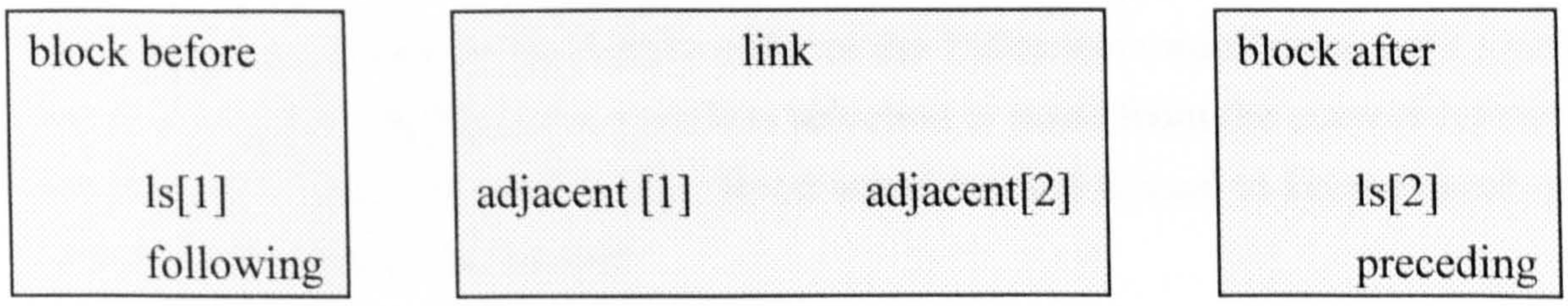
This method **Build** is called by Net.

The list of links is scanned.

For each link, which is legal, the list of blocks is scanned

1. to find the one indicated by the link as the block before;
   - the link is added to this block's list of links following,
   - and the link's *adjacent[1]* pointer is assigned to this block,
2. to find the one indicated by the link as the block after;
   - the link is added to this block's list of links preceding,
   - and the link's *adjacent[2]* pointer is assigned to this block,

This is illustrated by the diagram:



Summarise

This method is called by Net.

The list of blocks is scanned.

1. A report is written to file giving the type of block,
2. The block method *assignneighbours* is called, which:
   - refers to *ls[1]*, the list of following links, and creates goesto, the list of following blocks
   - refers to *ls[2]*, the list of preceding links, and creates, comesfrom, the list of preceding blocks.

Numbering blocks and checking whether the structure is a cyclic

A method is called by Net which **splits** the complete list of blocks into further lists:

starts, inters, exits, which are created according to whether each block is a *source, an intermediate block or a sink.*

For each of these sublists, which partition the structure, methods are called by Net, which assigns numbers to each block which correspond to Fulkerson's numbering.

The method which numbers the blocks for the intermediate blocks will loop indefinitely if the structure has a cycle. This method assigns the Boolean variable *cycles* to true or false.

Updating the structure

The method *Listupdate* is called by the object ListBlock.

The blocks are visited in the reverse order of the Fulkerson numbering, and the block method update is called for each.

The block method *update,* checks how many people can move forward into the next block nearer the exit, moves that number forward, updating the attributes for the next block and the current one.

The **alternative** random update method is called by the object ListBlock.
The blocks are visited in the reverse order of the Fulkerson's numbering until a junction block is reached. At this point a random selection is made from the *comesfrom* blocks for the next block to be visited. The 'flood wave' method is used to keep a check on which blocks have been visited.

Checking whether the building is empty

The method *evacdone* is called by the object Net.
This is a Boolean function which is assigned to be true when the contents of the list of blocks exits is equal to the contents of the entire list of blocks.

The main program

The main program uses units containing the methods for all the objects, and a unit which gives a screen display.

The tasks carried out by the program are as follows:

- the input and output files are set up,
- the data file is read,
- the graph structure is built,
- the blocks are numbered,
- if a cycle exists STOP

  otherwise read occupancy data,
- update the blocks, update file and screen,

  until the building is empty and STOP.

B2 Documentation of the Nurse Activity Model

B2.1 Objects for the referential spatial grid

The object **Block** is conceptualised as a physical space or a state. This implies that the physical location may be associated with more than one block. The object **Bed** is used for the state of a patient. Block and bed have location references which are rectangular co-ordinates corresponding with positions on the plan of the building.

The object **Link** can signify a movement from one location to another or a change of state. In both senses this represents an event. The object **Link**is used for movement and **Change** is used for the change of state for a patient.

The object **Net** is the complete set of blocks and links, and **lists** of blocks and links (using TCollection objects). **Net** is the graph structure which is the model for the building with spatial constraints on movement.

In a similar manner to the evacuation model, the blocks for the nurse activity model represent (rectangular) portions of space, which may be corridor sections in the ward or outside the ward, utility or office locations, the space alongside a bed or the bed itself. Dimensions of the corridor spaces relate to the choice of time-step and movement speeds of nurses.

The attributes and methods for Net are given in the units TVenvbld.pas and TVnet.pas for the evacuation model, and Hosp_bld.pas and Hosp_net.pas for the nurse activity model.

The object **Patient** is the complete set of **beds** and **changes**, and lists of beds and changes. Like Net, **Patient** is the model for what happens to patients; it stores the event rules and is a partitioned graph structure.

B2.2 Objects for event handling

BEvents are objects representing bound events. **BEvent** has attributes time the event is due, and the place at which *it will occur*. Another attribute denoting the change in state corresponding to the event, a pointer to change, is assigned *at execution time*.
The control algorithm uses an object **EventCal**, which is a descendant of TObject with fields which are descendants of ListItem (TCollections). It has methods to retrieve scheduled events and to add new events to the calendar

B2.3 Objects for the ward routines

Tasks are conceptualised as singular, which is an approximate representation, since routine task consists of many activities, each of which could be undertaken by one of the nurse team. The model could be extended to model such sequences.

Task is an object which has field attributes location, patient identity and time reported. with methods to create a new task and to assign the task to a nurse. The list of tasks, **ListTask** is another object which has methods for selecting which task should be assigned to a nurse and for deleting completed tasks.

Service is an object like EventCal which is the entire set of ward routines. This object can sort the tasks according to whether they are pending, delegated or active.

B2.4 The Nurse object

The object **Nurse** is a descendant of **Item,** and is modelled to be a mobile object, associated, at any particular time, with a unique spatial object **block. Nurse** inherits the integer array attribute for location, and this is interpreted to be the block identities for present position, next position and destination. New attributes are the pointer to the destination block, **target,** the pointer to the task assigned, **job_del** and the boolean variable, **busy.**

The object Nurse has methods to retrieve a task from the list, to move one block, to start a task and to finish a task.

The object **NurseTeam** represents the entire staff for the ward, and has methods for selecting nurses who are ready to treat, on the move or available for a new task.

Further notes:
The nurse activity model allow nurses to move freely according to the spatial constraints. The nurses have a **target** destination according to the delegated task. Attributes are needed in order to give information as to which blocks can be reached via each link. A Boolean attribute for **link** identifies which link leads to staff-base. Some links are on paths to a variety of destinations, including resource locations, such as sluice, kitchen etc. and the patients, who will be grouped in various wards.
The field or attribute **via** is a pointer to a list of blocks which are accessible using this link. The Net method **'SetRoutes'** assigns values to via for each link, and this information is available to nurses when searching though alternative links to move towards the target destination.

B2.5 Inheritance
The object classes **Block, Link, Nurse, Task** and **BEvent** are all descendants of **Item.** Lists of these objects are also descendants of **ListItem** and inherit the methods of the primitive object.

Patient, Bed and **Change** and lists of beds and changes have the same object hierarchy as **Net, Block, Link** and lists of blocks and links. The hierarchy is shown in diagram 31 section 9.5.4. Additional attributes and methods are required which are provided in the unit Pas_bld.pas.

B2.6 Interaction between objects
Objects have methods which will deliver information about field attributes and will make changes to the states of its own attributes.
The control algorithm is given in diagram 32 section 9.5.5. Interaction is controlled by the main program, which retrieves pointers to objects and activates the object for a response. Each object controls its own data giving encapsulation.

B2.7 The program units
The hierarchy of the units is as follows:

Calling program	NURSE_CA.PAS
Nurse object and Methods	NURSE_BLD.PAS
Task object, Service and Methods	TVTASK.PAS
Event objects, EventCal and Methods	HOSP_EV.PAS
Bed, Change, list objects and Patient	PAT_BLD.PAS
Net, Block, Link and list objects and Methods	HOSP_NET.PAS
	HOSP_BLD.PAS.

Methods in higher units may call methods in lower units or in those at the same level.

The public sections of the units are listed below in reverse order.

```
Unit Hosp_bld;
interface
uses Objects;

type
  File_name = string[14];
  block_type = string[6];
  link_type  = string[6];
  integerP   = ^integer;

  itemP = ^Item; blockP = ^block;  linkP = ^link;
  lsitemP = ^ListItem;  lslinkP = ^ListLink; lsBlockP =
^ListBlock;
  pairlslinkP = array[1..2] of lslinkP;
  pairblockP = array[1..2] of blockP;

 Item = Object(TObject)
   id , capacity, number{of node},scan_num  : integer;
   bid                          :array [1..3] of integer;
   dry  {not checked}                  :boolean;
   constructor init;
   procedure setnumber(newnumber : integer);
   procedure setup ; virtual;
   procedure display; virtual;
   procedure update; virtual;
   destructor done; virtual;
   destructor cancel; virtual;
  end {object Item};

  link = Object(Item)
{inherits id cap number scan_num bid[] bid[3]
 1=door 0=not door dry}
    lktp                : link_type;
    arrow {true if capacity>0}: boolean;
    {passable, denoted by arrow true when id>0 &cap >0 }
    toSB                 : boolean; {leading to Staff Base}
    via                 : lsBlockP; {blocks led to }
    adjacent             : pairblockP;
    constructor init;
    constructor neighinit;
    constructor viainit;
    procedure
getlinkdata(new_id,new_cap,new_loc1,new_loc2,

new_loc3,new_number:integer;new_lktp:link_type);
    function isonpath(dest: BlockP) : boolean;
    {true if link leads to the block indicated}
    procedure setup; virtual;
    function isdedicated: boolean;
    procedure display; virtual;
    procedure routesdisplay;
    function stepinto:blockP;
    function steplength: integer;
    procedure update;  virtual;
    destructor done; virtual;
    destructor cancel; virtual;
  end {object link};


  block = Object(Item)
    {inherits id cap number bid[]}
    flag                    : integer;
    blktp{exit,room,corr,stair} : block_type;
    ls                      : pairlslinkP;
    homegoesto, homecomesfrom   : LsBlockP; {for object
Net}
    awaygoesto, awaycomesfrom   : LsBlockP; {for object
Net}

{***** fields for needed hospital activity
**********************}
    num {number of people }     : integer;
    nurse_id          : integer;
{**************************************}
    constructor init;
    constructor listsinit;
    procedure setup; virtual;
    procedure occupancy;
    procedure display; virtual;
    procedure update; virtual;
    function isStaffBase: boolean;
    function isstart: boolean;
    function issource: boolean;
    function issink: boolean;
    function isbetween: boolean;
    function isjunction: boolean;
    function isculdesac: boolean;
    function wayhome : linkP;
    function nearerhome : blockP;
    function steplink(destination:blockP): linkP;
    procedure identify;
    procedure assignneighbours;
    function traceback: lslinkP;
    destructor done; virtual;
    destructor cancel; virtual;
  end {object block};

ListItem = Object(TCollection)
  checks : boolean;
  constructor init;
  procedure listsetup ; virtual;
  procedure listdisplay; virtual;
  procedure listupdate; virtual;
  procedure resetdry;
  function identity(ipoint : ItemP): integer;
  function isempty:boolean;
  function notempty:boolean;
  procedure diff(list2: lsItemP) ;
  procedure additem(newitem : itemP);
  procedure replace(Anitem : itemP);
  function head : itemP;
  procedure tail;
  function before(thisitem : itemP): itemP;
  function next(thisitem : itemP): itemP;
  destructor done; virtual;
  destructor alldone; virtual;
  destructor listdone; virtual;
Private
  currentitem : integer;
  end {ListItem};


ListLink = Object(ListItem)
   {inherits currentitem from ListItem}
   {inherits isempty notempty tail}
   constructor init;
   procedure listsetup ; virtual;
   procedure listdisplay; virtual;
   procedure addlink(newlink : linkP);
   procedure listupdate; virtual;
   function headl : linkP;
   function reverse(ip: linkP): linkp;
   function linkhome :linkP;
   function findway(dest:blockP): linkP;
   procedure CrossoutArrows;
   function NoArrows : boolean;
   function CountArrows : integer;
   procedure ResetArrows;
   function search_link(k :integer) : linkP;
   destructor done; virtual;
   destructor alldone; virtual;
   destructor listdone; virtual;
end {ListLink};
```

```
ListBlock = Object(ListItem)
  {inherits currentitem from ListItem}
  {inherits isempty notempty tail}
  constructor init;
  procedure listsetup ; virtual;
  procedure listdisplay; virtual;
  procedure listupdate; virtual;
  procedure randlistupdate;
  function headb : blockP;
  function memberb(bp : BlockP) : boolean;
  function search_blk(blkid : integer) : blockP;
  function findblockid(bp:blockP) : integer;
  function search_fulk(k : integer) : blockp;
  function contents : integer; {sum of num in each block}
  procedure printout;
  destructor done; virtual;
  destructor alldone; virtual;
  destructor listdone; virtual;
end {ListBlock};


var
  Infile   : Text;
  Outfile  : Text;
  Input_file_name  : File_name;
  Output_file_name : File_name;
  clock : integer;
  printcheck : boolean;
  function Getminimum(i,j :integer) : integer;
{*****************************************}
```

```
Unit Hosp_net;
interface
uses Objects, hosp_bld;

type

 NetP = ^Net;
 Net = Object(TObject)
   spacesP     : lsBlockP;
   connectP    : lsLinkP;
      {pointer to the list of links}
   destinations,inters,starts, ends  : lsBlockP;
{pointers to lists of blocks of different types}
{destinations are patients and facilities}
{starts is Staff Base}
   constructor init;
   procedure netsetup ;
   procedure netdisplay;
   procedure netupdate;
   procedure listblocksbefore; virtual;
   constructor initblocklinks;
   procedure connectlinks;
   procedure splits;
   procedure splittwoways;
   procedure StartsFulks(Totnode : integer);
   procedure IntersFulks(Totnode :integer;var cycles:
boolean);
   procedure DestFulks(TotNode : integer);
   procedure build{(blklist :lsBlockP; lnklist : lslinkP)};
   procedure setroutes;
   procedure reorder(LastNo :integer;var Fulks :
LsBlockP);
   procedure summarise;
   function SpacesList: LsBlockP;
   function LinksList: LsLinkP;
   procedure OrderScan(Totalnode :integer);
   function evacdone: boolean;
   destructor blocklinksdone;
   destructor done; virtual;
   destructor shutdown;
 end {object Net};

implementation
```

```
Unit Pat_bld;
interface
uses Hosp_bld, Hosp_net, Objects;

type
  bedP = ^Bed;   {patient state}
  changeP = ^Change; {change of state}
  PatientP = ^Patient;   {Patient structure}
  lsbedP = ^ListBed;
  lschangeP = ^Listchange;
  change_type = string[6];
  bed_type    = string[6];
  pairpatP  = array[1..2] of bedP;
  pairlschangeP = array[1..2] of lschangeP;

  change = Object(Item)
{inherits id cap number scan_num bid[] bid[3] 1=door
0=not door dry}
    chgetp              : change_type;
    valid {like arrow}      : boolean;
    nearby              : pairPatP;
    constructor init;
    constructor nearinit;
    procedure setup; virtual;
    procedure display; virtual;
    procedure update;  virtual;
    procedure start_treat(np : pointer);
    procedure finish_treat;
    destructor done; virtual;
    destructor cancel; virtual;
  end {object change};


  bed = Object(Item)
 {inherits id cap number bid[]}
    timecomfortable: integer;
    timewaiting    : integer;
    timetreated  : integer;
 { records  }
    bedtp       : bed_type;
    pat_id      : integer;
    nurse_trt    : pointer;
    comfortable   : boolean;
    treatment     : boolean;
    potential     : pairlschangeP;
    nextstate     : lsbedP;
    prevstate     : lsbedP;
    constructor init;
    constructor listsetup {initialise lists};
    procedure setup; virtual;
    function isabed:boolean;
    function isoccupied: boolean;
    function iscomfortable: boolean;
    function undertreatment: boolean;
    procedure display; virtual;
    procedure update; virtual;
    function iswaiting: boolean;
    procedure begintreated(kpat :integer;np :pointer);
    procedure endtreated;
    procedure suspend;
    procedure patcall;
    procedure patcomfort(kp :integer);
    procedure identify;
    procedure assignnear;
    function wherebed: integer;
    function occupant : integer;
    {return patient id given bed id}
    function findchange: ChangeP;
    function findstartchange:ChangeP;
    function findnextstate: bedp;
    destructor done; virtual;
    destructor cancel; virtual;
  end {object bed};


ListChange = Object(ListItem)
  {inherits currentitem from ListItem}
  {inherits isempty notempty tail}
  constructor init;
  procedure listsetup ; virtual;
  procedure listdisplay; virtual;
  procedure addchange(newlink : ChangeP);
  procedure listupdate; virtual;
  function headch : changeP;
  function search_change(k :integer) : changeP;
  destructor done; virtual;
  destructor alldone; virtual;
  destructor listdone; virtual;
end {Listchange};

ListBed = Object(ListItem)
  {inherits currentitem from ListItem}
  {inherits isempty notempty tail}
  constructor init;
  procedure listsetup ; virtual;
  procedure listoccupancy;{hosp_bld might not use this}
  procedure listdisplay; virtual;
  procedure listupdate; virtual;
  procedure listreport(due :integer);
  function headbed : bedP;
  function memberbed(bp : BedP) : boolean;
  function search_bed(bdid : integer) : bedP;
  function search_pat(bdid :integer) : integer;
  function ScanOccup: BedP;
  function ScanComfortable: BedP;
  destructor done; virtual;
  destructor alldone; virtual;
  destructor listdone; virtual;
end {ListBed};

Patient = Object(TObject)
  statesP     : lsBedP;
  changesP    : lschangeP;
    {pointer to the list of links}
  occupied,waiting,satisfied,attended  : lsBedP;
{pointers to lists of states of different kinds}
  constructor init;
  procedure patsetup ;
  function BedsList: lsBedP;
  function ChangesList: lsChangeP;
  procedure patdisplay;
  procedure patupdate;
  procedure liststatesbefore; virtual;
  constructor initbedchanges;
  procedure connectchanges;
  procedure splitstates;
  procedure build;
  procedure report;
  function pointtochange(k: integer) : ChangeP;
  destructor bedchangesdone;
  destructor done;  virtual;
  destructor complete;
end {object Pat};
{*****************************}


implementation
```

```
Unit Hosp_ev;
interface
uses Objects, hosp_bld, hosp_net, pat_bld;

type
  event_type = string[6];
  ptrBevent = ^Bevent;
  Bevent = Object(Item)
{inherit id, number = index,   scan_num = unique index}
{when assigned index must be unique}
    time_due : integer;
        {event will be executed by a link method}
    ptrChange : changeP;
        {inherit dry  = whether checked and executed}
        {event may affect 2 spaces, indicate first space}
    ptrspace : blockP;
        {events may be exogenous or endogenous}
    What_hap  : event_type;
    Constructor init;
    function duenow(due:integer):boolean;
    function earlier(due:integer):boolean;
    function later(due:integer):boolean;
    function Wherehap: integer;
    function WhenHap:  integer;
    procedure tag;
    Procedure evcreate(due, k :integer; bs :lsblockp );
    Procedure Execute(clock : integer; bs : lsblockp; Patmix
: PatientP);
    procedure display; virtual;
    Destructor cancel; virtual;
    Destructor done; virtual;
  end {Bevent record} ;

  ptrListEvent = ^ListEvent;

  ListEvent = Object(ListItem)
    constructor init;
  procedure startup(bs:lsblockp);
  procedure evsetup(due, k:integer; bs : lsblockP);
  function latestevent(due : integer): ptrBevent;
  function earliest(due :integer): ptrBevent;
  procedure listdisplay; virtual;
  procedure renumber;
  function memberev(evp:ptrBevent): boolean;
  procedure delete_event(pastevent : ptrBevent);
  procedure knockoff;
  procedure wipe;
  destructor alldone; virtual;
  destructor listdone; virtual;
  destructor done; virtual;
  end {ListEvent};

ptrEventCal = ^EventCal;

EventCal = Object(TObject)
   AllEvents                    : ptrListEvent;
   beforelis, simultaneous, afterlis : ptrListEvent;
   { soonest : ptrBevent;}
  constructor init;
  function NoEvents : boolean;
  procedure start(bs:lsblockp);
  procedure remove;
  procedure evsplits(due:integer);
  procedure oldaddevent(due, k :integer;bs : lsblockp);
  procedure addevent(due, k :integer;bs : lsblockp);
  function Retrieve(due: integer) : ptrBevent;
  procedure update(due : integer; var newtime : integer);
  procedure eventrep;
  destructor eventslistdone;
  destructor mopup;
  destructor done; virtual;
  end  {EventCal};

var
plist : lsblockp;

implementation
```

```
Unit TVtask;
interface
uses Objects, Hosp_ev, hosp_bld,hosp_net,pat_bld;

type
  task_type = string[6];
  ptrTask = ^Task;
  ptrListTask = ^ListTask;
  ServP   = ^Service;

  Task = Object(Item)
  {inherit id,capacity used for priority}
  {number to be used for cumulative number}
    time_call:integer;
  {inherit dry  = whether checked and put on list}
    ptrlocation : blockP;
    nurse_del : pointer;
    active   : boolean;
{task associated with patient in block}
{pointer to nurse not used but integer only}
    activity  : task_type;
    Constructor init;
    function isdelegated: boolean;
    function taskspace: integer;
    function taskidentity: integer;
    Procedure AssignTask(np : pointer);
    Procedure create(call_no, due, k, priority :integer; bs
:lsblockp );
    procedure undertake(np :Pointer);
    Procedure abandon;
    procedure display; virtual;
    Destructor cancel; virtual;
    Destructor done; virtual;
  end {Task record};

  ListTask = Object(ListItem)
    constructor init;
    function nearestcall(k : integer): ptrTask;
    function mosturgent: ptrTask;
    procedure listdisplay; virtual;
    function  membertk(tp: Ptrtask): boolean;
    procedure delete_task(completetask : ptrTask);
    procedure setup(call_no,due,k,priority :integer;
bs:lsblockP);
    procedure addtask(tsk:ptrtask);
    procedure wipe;
    destructor alldone; virtual;
    destructor listdone; virtual;
    destructor done; virtual;
    end {ListEvent};

  Service  = Object(TObject)
    alltasks, Pending,Delegated, InAction   :ptrListTask;
    constructor init;
    procedure tasksplits;
    procedure tsksplits;
    procedure taskreport;
    procedure emergscan(patlist :lsbedP);
    destructor taskslistdone;
    destructor sweep;
    destructor done; virtual;
    end  {Service};

var
plist : lsblockp;

implementation
```

```
Unit Nurse_bld;
interface
uses Objects, hosp_bld, hosp_net, TVtask, Hosp_ev,
pat_bld;

type
 nurse_type = string[6];
 nurseP = ^nurse;
 lsnurseP = ^nurseteam;

 Nurse = Object(Item)
 {inherit: id ,capacity=seniority, number ? ,scan_num ?}
 {bid[1] is present block id, bid[2] is next, bid[3] is
destination}
 { dry  i.e.not checked}
   busy : boolean;
   present_location : BlockP;
{ where nurse is at present}
   target  : BlockP;  {the block moving towards}
   which_pat  : integer;
   job_del  : PtrTask;
{the job nurse intending to do next}
{ records  }
   log          : integer;
   timeidle     : integer;
   timewalking   : integer;
   timetreating  : integer;
{ records  }
   constructor init;
   constructor pointerinit;
   procedure locate(bs : lsBlockP);
   procedure adjust_loc(bs : lsBlockP);
   procedure setup ; virtual;
   procedure display; virtual;
   procedure update; virtual;
   function isavail : boolean;
   procedure get_task(bs: lsblockP;jp : ptrTask);
   {k obtained using ListTask and Task methods}
   function nursetask: ptrtask;
   function isonmove: boolean;
   procedure move;
   {one step from present block to one nearer the target}
   function isready: boolean;
   {is at location for treating patient}
   function tasklocate: integer;
   procedure Start_Task(k:integer);
{now at the target block start treatment
 and schedule end time}
   Procedure Finish_Task;
{Bound event: nurse becomes not busy. this method deals
with nurse}
{parameters for job_del and present_location are fields of
nurse}
    destructor done; virtual;
    destructor cancel; virtual;
 end {object Nurse};

NurseTeam = Object(ListItem)
{inherits checks boolean and methods isempty notempty}
  constructor init;
  procedure listsetup ; virtual;
  procedure putinplace(bs:lsblockP);
  procedure adjust(bs :lsblockp);
  procedure listdisplay; virtual;
  procedure staffupdate; virtual;
  procedure listreport;
  {may need method to select nurse}
  function  find_nurse(nursenum:integer):nurseP;
  function  get_nurse(blkid :integer; bs: lsblockP): NurseP;
  function ScanMove: NurseP;
{get pointer to first nurse who is moving}
  function ScanReady: NurseP;
{get pointer to first nurse who is ready to start treatment}
  function ScanAvail: NurseP;
{get pointer to first nurse who is available}
  destructor done; virtual;
  destructor alldone; virtual;
  destructor listdone; virtual;
end {ListItem};

{*****************************************}
implementation
```

```
unit hosp_dr;
{Procedures for setting up the screen for plan of building}
{ data id, capacity, num, ref across, ref down, block type }
interface
uses
  crt, Graph, hosp_bld, hosp_net;

var
  clock,Gd,Gm, num_acc,num_dwn : integer;
  bpp,bp       : blockp ;

function SetNumAcc: integer;
{gives the number of cells across, call before switchOn
after OpenFile}
function SetNumDwn : integer;  var j :integer;
{gives the number of cell down, call before SwitchOn after
OpenFile}
procedure SwitchOn( Gd,Gm :integer);
procedure SwitchOff;
function IntToStr(i: integer): String;
Function GridXstep(i:integer):integer;
Function GridYstep(j:integer):integer;
procedure Displayclock(due :integer);
procedure DrawGrid(i,j :integer);
{in graph mode draws a grid of cells i across and j down}
Procedure LabelGrid(bs : lsblockp; num_acc, num_dwn
:integer);
Procedure Occupshow(bs : lsblockp; num_acc, num_dwn
:integer);

implementation
```

Algorithm for Nurse activity model

As given in Diagram 33 in Chapter 9 section 9.6.5

INITIALISATION

Input data file and construct graph structures for nurses and patients

Set Clock to zero

Set initial conditions including a list of routine Tasks

Goto B Phase

A PHASE Update clock by one unit.

If not termination time goto B Phase

otherwise Stop

B PHASE Bound events

Check Event Calendar and execute events due at Clock time and delete them.

Check through all patients and generate calls for treatment, adding to the list of Tasks.

C PHASE Conditional Events

1. **Nurse assigned to task.**

   Nurses who are free located. Select task appropriate from list. Assign nurse delegation and target. Assign state of patient.

2. **Nurse Movement:**

   Each nurse is located (space block occupied), checked for delegation and target location. Search through outgoing links and check the via lists for target; select valid link and next block. Move nurse into that block.

3. **Start treatment.**

   Patients waiting for treatment are located. For each one check if nurse in block adjacent. Execute change for nurse and patient. Sample treatment time. Schedule end of treatment on event calendar.

Note: The conditional events are ordered so that other events are made possible at the same clock time.

For example;

A nurse may have just moved to a location and can start treatment.

A nurse delegated to a task can start movement at same the clock time.

Data-files for Nurse activity model

- list of blocks accessed by nurses
- list of links for nurses' movement
- list of blocks for patient states
- list of links for patients' changes of state
- blocks and corresponding patient identities
- nurses on duty with initial location

Layout 1 (compact)

```
1 9 9 2 1 sbase
2 1 2 1 1 pat
3 9 3 1 1 bybed
4 9 4 1 1 bybed
5 1 5 1 1 pat
6 9 6 1 1 bybed
7 9 7 1 1 bybed
8 1 8 1 1 pat
13 9 3 2 1 corr
14 9 4 2 1 corr
15 9 4 2 1 corr
16 9 6 2 1 corr
17 9 7 2 1 corr
18 9 8 2 1 corr
22 1 2 3 1 pat
23 9 3 3 1 bybed
24 9 4 3 1 bybed
25 1 5 3 1 pat
26 9 6 3 1 bybed
27 9 7 3 1 bybed
28 1 8 3 1 pat
-1
```

```
998 13 3 3 -3 topat
997 13 23 3 -3 topat
996 14 13 2 -3 topat
995 14 4 3 -3 topat
994 14 24 3 -3 topat
993 15 14 2 -3 topat
992 16 15 2 -3 topat
991 16 6  2 -3 topat
990 16 26 3 -3 topat
989 17 16 2 -3 topat
988 17 7  3 -3 topat
987 17 27 3 -3 topat
986 18 17 2 -3 topat
985 1  18 4 -3 topat
1  3 13  3 3 tosb
2  23 13  3 3 tosb
3  13 14  3 3 tosb
4  4 14  3 3 tosb
5  24 14  3 3 tosb
6  14 15  2 3 tosb
7  15 16  2 3 tosb
8  6 16  3 3 tosb
9  26 16  3 3 tosb
10 16 17  2 3 tosb
11  7 17  3 3 tosb
12  27 17  3 3 tosb
13  17 18  2 3 tosb
14  18 1  4 3 tosb
-1
```

```
2 1 2 1 1 pat
3 0 3 1 1 bybed
4 0 4 1 1 bybed
5 2 5 1 1 pat
6 0 6 1 1 bybed
7 0 7 1 1 bybed
8 3 8 1 1 pat
22 4 2 3 1 pat
23 0 3 3 1 bybed
24 0 4 3 1 bybed
25 5 5 3 1 pat
26 0 6 3 1 bybed
27 0 7 3 1 bybed
28 6 8 3 1 pat
-1
998 2 3 0 3 start
997 5 4 0 3 start
996 5 6 0 3 start
995 8 7 0 3 start
994 23 22 0 3 start
993 25 24 0 3 start
992 25 26 0 3 start
991 28 27 0 3 start
1 3 2 0 3 finis
2 4 5 0 3 finis
3 6 5 0 3 finis
4 7 8 0 3 finis
5 22 23 0 3 finis
6 24 25 0 3 finis
7 26 25 0 3 finis
8 27 28 0 3 finis
-1
2 1
5 2
8 3
22 4
25 5
28 6
-1 0
1  1 1 1 1
-1
```

Layout 2 (long)

```
1 2 1 1 pat
3 9 3 1 1 bybed
4 9 4 1 1 bybed
5 1 5 1 1 pat
6 9 6 1 1 bybed
7 9 7 1 1 bybed
8 1 8 1 1 pat
9 9 9 1 1 bybed
10 9 10 1 1 bybed
11 1 11 1 1 pat
12 9 12 1 1 bybed
13 9 13 1 1 bybed
14 1 14 1 1 pat
15 9 15 1 1 bybed
16 9 16 1 1 bybed
17 1 17 1 1 pat
22 9 2  2 1 corr
23 9 3  2 1 corr
24 9 4  2 1 corr
25 9 5  2 1 corr
26 9 6  2 1 corr
27 9 7  2 1 corr
28 9 8  2 1 corr
29 9 9  2 1 corr
30 9 10 2 1 corr
31 9 11 2 1 corr
32 9 12 2 1 corr
33 9 13 2 1 corr
34 9 14 2 1 corr
35 9 15 2 1 corr
36 9 16 2 1 corr
37 9 17 2 1 corr
1  9 18 2 1 sbase
-1
```

```
1  3 23 3 3 tosb
2 23 24 2 3 tosb
3  4 24 3 3 tosb
4 24 25 2 3 tosb
5 25 26 2 3 tosb
6  6 26 3 3 tosb
7  2 27 3 3 tosb
8  7 27 3 3 tosb
9 27 28 2 3 tosb
10 28 29 2 3 tosb
11  9 29 3 3 tosb
12 29 30 2 3 tosb
13 10 30 3 3 tosb
14 30 31 2 3 tosb
15 31 32 2 3 tosb
16 12 32 3 3 tosb
17 32 33 2 3 tosb
18 13 33 3 3 tosb
19 33 34 2 3 tosb
20 34 35 2 3 tosb
21 15 35 3 3 tosb
22 35 36 2 3 tosb
23 16 36 3 3 tosb
24 36 37 2 3 tosb
25 37 1  4 3 tosb
998 23  3 3 -3 topat
997 24 23 2 -3 topat
996 24  4 3 -3 topat
995 25 24 2 -3 topat
994 26 25 2 -3 topat
993 26  6 3 -3 topat
992 27  2 3 -3 topat
991 27  7 3 -3 topat
990 28 27 2 -3 topat
989 29 28 2 -3 topat
987 30 29 2 -3 topat
986 30 10 3 -3 topat
985 31 30 2 -3 topat
984 32 31 2 -3 topat
983 32 12 3 -3 topat
982 33 32 2 -3 topat
981 33 13 3 -3 topat
980 34 33 2 -3 topat
979 35 34 2 -3 topat
978 35 15 3 -3 topat
977 36 35 2 -3 topat
976 36 16 3 -3 topat
975 37 36 2 -3 topat
974  1 37 4 -3 topat
-1
```

```
2 1 2 1 1 pat
3 0 3 1 1 bybed
4 0 4 1 1 bybed
5 2 5 1 1 pat
6 0 6 1 1 bybed
7 0 7 1 1 bybed
8 3 8 1 1 pat
9 0 9 1 1 bybed
10 0 10 1 1 bybed
11 4 11 1 1 pat
12 0 12 1 1 bybed
13 0 13 1 1 bybed
14 5 14 1 1 pat
15 0 15 1 1 bybed
16 0 16 1 1 bybed
17 6 17 1 1 pat
-1
98  2  3 0 3 start
97  5  4 0 3 start
96  5  6 0 3 start
95  8  7 0 3 start
94  8  9 0 3 start
93 11 10 0 3 start
92 11 12 0 3 start
89 17 16 0 3 start
1  3  2 0 3 finis
2  4  5 0 3 finis
3  6  5 0 3 finis
4  7  8 0 3 finis
5  9  8 0 3 finis
6  10 11 0 3 finis
7  12 11 0 3 finis
8  13 14 0 3 finis
9  15 14 0 3 finis
10 16 17 0 3 finis
-1
2 1
5 2
8 3
11 4
14 5
17 6
-1 0
1 1 1 1 1
-1
```

Nurse activity model
Results of run A

Time-step	2 seconds
Probability of patient call	0.01
Treatment times	100 clock ticks (200 seconds or 3 minutes 20 seconds)
Length of run	5000 clock ticks (10,000 seconds or 167 minutes)

Layout 1 (compact)

Patient number	Time comfortable			Time waiting			Time being treated		
	clock ticks	minutes	%	clock ticks	minutes	%	clock ticks	minutes	%
1	796	27	15.92	3346	112	66.92	858	29	17.16
2	1529	51	30.58	2771	92	55.42	700	23	14.00
3	1167	39	23.34	3033	101	60.66	800	27	16.00
4	885	30	17.70	3295	110	65.90	900	30	18.00
5	1324	44	26.48	2876	96	57.52	800	27	16.00
6	863	29	17.26	3337	111	66.74	800	27	16.00
mean	1100	37	22.00	3100	103	62.00	800	27	16.00

Nurse measures

Time spent treating patients	Time spent walking	Time spent idle	Distance walked
4858/5000 = 97.16%	142/5000 = 2.84%	0	492 steps = 344m

Layout 2 (long)

Patient number	Time comfortable			Time waiting			Time being treated		
	clock ticks	minutes	%	clock ticks	minutes	%	clock ticks	minutes	%
1	865	29	17.30	3428	114	68.56	707	24	14.14
2	769	26	15.38	3531	118	70.62	700	23	14.00
3	940	32	18.80	3260	109	65.20	800	27	16.00
4	848	28	16.95	3352	112	67.04	800	27	16.00
5	809	27	16.18	3391	113	67.82	800	27	16.00
6	399	13	7.98	3801	128	76.02	800	27	16.00
	775	26	15.50	3450	115	69.00	775	26	15.50

Nurse measures

Time spent treating patients	Time spent walking	Time spent idle	Distance walked
4607/5000 = 92.14%	393/5000 = 7.86%	0	1004 steps = 703m

Nurse activity model
Results of run B
Performance measures for nurses

Time-step	2 seconds
Probability of patient call	0.005
Treatment times	100 clock ticks (200 seconds or 3 minutes 20 seconds)
Length of run	1000 clock ticks (2000 seconds or 33.33 minutes)

Layout 1 (compact)

Run number	Time spent in each state in clock ticks			Distance walked
	Treating patients	Walking	Idle	number of paces
1	973	27	0	97
2	976	24	0	91
3	953	34	13	117
4	974	26	0	96
5	919	39	42	127
mean of 5 runs				
clock ticks	959	30	11	106 paces
minutes	32	1	22 secs	74 metres
% of work time	96	3	1	

Layout 2 (long)

Run number	Time spent in each state in clock ticks			Distance walked
	Treating patients	Walking	Idle	number of paces
1	900	100	0	244
2	894	80	26	209
3	913	87	0	221
4	912	88	0	223
5	918	82	0	221
mean of 5 runs				
clock ticks	907	87	5	224 paces
minutes	30	3	10 secs	157 metres
% of work time	91	8.5	0.5	

Nurse Activity model
Results of run B

Performance measures on patients

Time-step 2 seconds
Probability of patient call 0.005
Treatment times 100 clock ticks
200 seconds or 3minutes 20 seconds
Length of run 1000 clock ticks
2000 seconds or 33.33 minutes
Each run uses an independent seed for the patient's random calls.
Layout 1 (compact)

Average results over the 5 runs, giving % of total work time in each state.

ROW	avcomf	avwt	avtrt
1	37.06	47.88	15.06
2	43.98	42.50	13.52
3	31.78	44.76	23.46
4	24.96	59.02	16.00
5	34.78	47.54	17.86
6	56.94	33.06	10.00

Results for each run
times spent in each state in clock ticks

Run 1

ROW	comf1	wait1	trt1
1	337	463	200
2	623	277	100
3	161	566	273
4	213	687	100
5	249	551	200
6	641	259	100

Run 2

ROW	comf2	wait2	trt2
1	226	574	200
2	279	545	176
3	341	459	200
4	154	645	200
5	318	482	200
6	660	340	0

Run 3

ROW	comf3	wait3	trt3
1	315	532	153
2	280	520	200
3	304	496	200
4	261	639	100
5	255	545	200
6	546	354	100

Run 4

ROW	comf4	wait4	trt4
1	457	443	100
2	215	585	200
3	379	421	200
4	314	486	200
5	219	607	174
6	496	404	100

Run 5

ROW	comf5	wait5	trt5
1	518	382	100
2	802	198	0
3	404	296	300
4	306	494	200
5	698	192	119
6	504	296	200

Nurse Activity model
Results of run B

Performance measures on patients

Time-step 2 seconds
Probability of patient call 0.005
Treatment times 100 clock ticks
200 seconds or 3 minutes 20 seconds
Length of run 1000 clock ticks
2000 seconds or 33.33 minutes
Each run uses an independent seeds for the patient's random calls.
Layout 2 (long)

Average results over the 5 runs, giving % of total work time in each state.

ROW	wavcmf	wavwt	wavtrt
1	32.16	51.84	16.00
2	38.32	45.68	16.00
3	30.98	48.66	20.36
4	42.48	45.26	12.26
5	39.42	48.46	14.12
6	53.78	34.22	12.00

Results for each run
times spent in each state in clock ticks

Run 1

ROW	wcomf1	wwait1	wtrt1
1	240	560	200
2	511	389	100
3	408	392	200
4	373	527	100
5	301	599	100
6	452	348	200

Run 2

ROW	wcomf2	wwait2	wtrt2
1	520	280	200
2	493	307	200
3	770	130	100
4	534	266	200
5	539	367	194
6	996	4	0

Run 3

ROW	wcomf3	wwait3	wtrt3
1	153	647	200
2	316	584	100
3	202	598	200
4	77	810	113
5	274	526	200
6	317	583	100

Run 4

ROW	wcomf4	wwait4	wtrt4
1	276	624	100
2	366	434	200
3	70	630	300
4	431	469	100
5	567	421	12
6	243	557	200

Run 5

ROW	wcomf5	wwait5	wtrt5
1	419	481	100
2	230	570	200
3	99	683	218
4	709	191	100
5	290	510	200
6	681	219	100

Appendix C

Extract from Planning For People:

The pedestrian simulation model, PEDROUTE

SAFETY AND EFFICIENCY ARE THE TWIN PRIORITIES FOR RAILWAY AUTHORITIES, FACED WITH BUOYANT TRAFFIC AND YET CONSTRAINED BY FINANCE. A NEW PLANNING TOOL HAS BEEN DEVELOPED IN LONDON OFFERING MAJOR ADVANCES ON PREVIOUS PRACTICE.

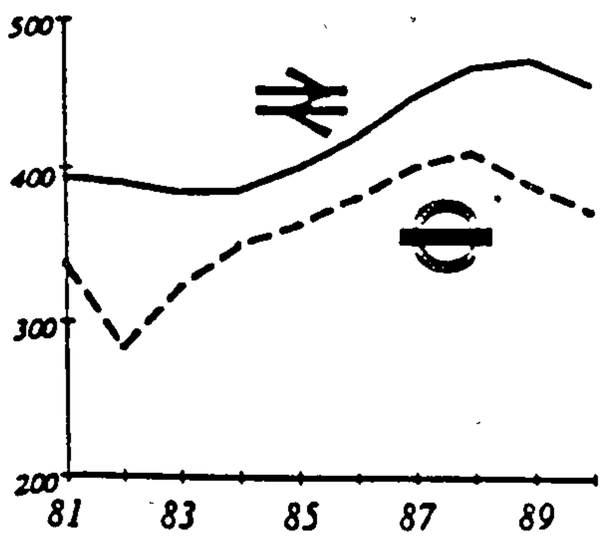
PEDROUTE

Halcrow Fox and Associates

In railway and bus stations, airport terminals and building complexes, large pedestrian movements are the common experience. So too are delay and frustration, and increasingly, concerns for safety. London Underground Limited (LUL) is a prime example of an undertaking faced with improving existing high standards of safety at stations to cope with increasing volumes of pedestrians, concentrated in short peak periods. To assist in the planning of stations, both existing and new, Halcrow Fox and Associates, in collaboration with LUL, developed a pedestrian simulation model, PEDROUTE. This paper describes the background to the development of the model, how it works, and outlines some of its applications.

BACKGROUND

The rail network in London has developed for historical reasons as a *hub and spoke* system, with many radial lines (the *spokes*) feeding into a central core (the *hub*). Accordingly, the focus of activity is at a handful of key termini which serve large parts of London and the South-East, and where considerable volumes of passengers transfer to Underground services for their onward movement throughout the city's core.



Trend in Morning Peak Carryings (000s)

	1989
	389,000
	473,000

Arrivals between 07.00 and 10.00 in central area.

Some 40% of BR arrivals transfer to LUL at the main termini.

With the resurgence in rail commuting following the imposition of stringent parking controls in the capital, the development of the Travelcard scheme, and a general growth in employment levels during the 1980's, the system, and more particularly the main termini, have become increasingly congested.

One element of LUL/BR's response to the growth in patronage has been to upgrade existing lines and services through new stock, resignalling to improve frequencies and the restructuring of services such as the Thameslink network. However, a major programme of Underground station capacity improvements has also been launched to tackle over-crowding at the main central London stations. In addition a series of new lines have been proposed to provide for longer-term requirements, including Crossrail, the Jubilee Line Extension and the Chelsea-Hackney Line, which aim to distribute the passenger burden over more stations and open up new areas of employment.

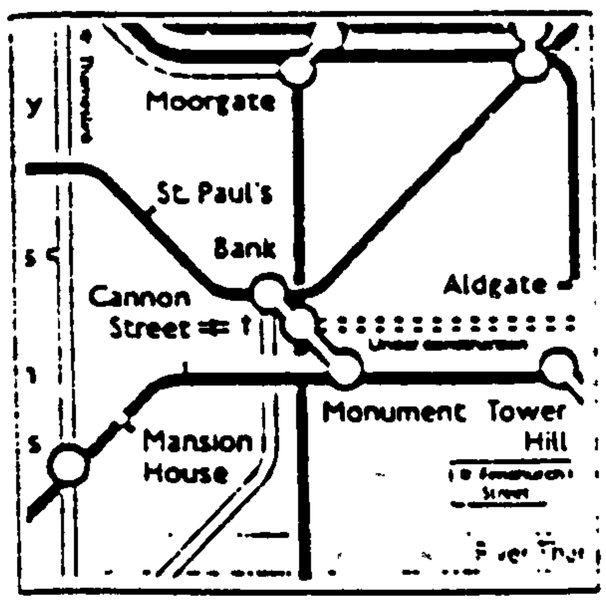
It is with this in mind that LUL decided to develop a pedestrian simulation model to assist with the on-going task of improving the service levels in the London Underground. PEDROUTE simulates pedestrian behaviour in congested conditions and quantifies the benefits of changing levels of congestion. The package has three distinct roles:

- Developing, and evaluating the performance of, options to increase capacity at congested stations, then providing the economic justification for such new investment;
- Optimising the design of new station facilities in economic and safety terms to ensure their suitability for predicted passenger volumes; and
- Demonstrating compliance with safety regulations, for example in meeting a required evacuation time for a station.

This new approach to station planning has been successfully applied at over 25 stations, ranging from some of the largest interchanges in London at Waterloo and Liverpool Street, to smaller stations such as Knightsbridge and Bayswater.

EARLY TECHNIQUES

HFA's experience of pedestrian modelling developed from early work on behalf of The City of London which commissioned the Consultants to appraise London Regional Transport's proposals to extend the Docklands Light Railway to Bank, and to identify alternative locations for the siting of the terminal in central London.

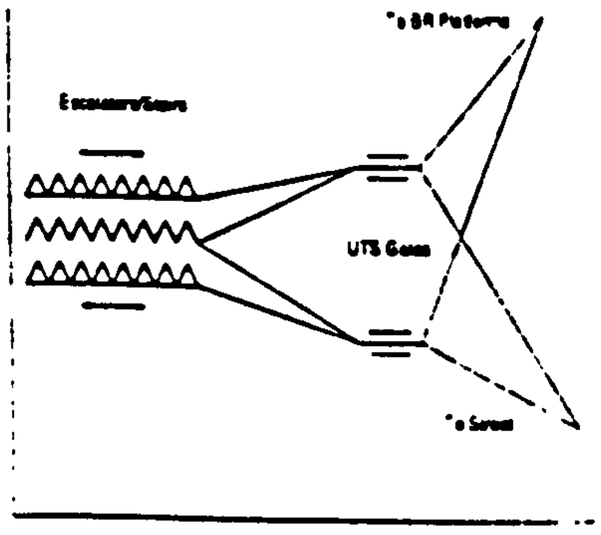


The extension of the DLR to Bank Station provided an early application of the pedestrian modelling techniques

The City's concern centred on the pedestrian congestion that may arise at the Bank complex, so as one element of the study a simple link-node model of the pedestrian network for Bank, Monument and Cannon Street stations was developed. Using link capacities derived from work by Fruin[1], the existing system was examined to identify overloaded links. Subsequently, networks were constructed to represent the proposals from LRT and the City for the development of the station, and passenger flows were forecast based on passenger surveys. Assignments were carried out on an "all or nothing" basis using fixed time trees. The results indicated areas of potential congestion and the relative accessibility afforded by the alternative schemes.

When HFA was appointed to undertake a review of Victoria Underground Station in 1987, it was clear that an "all or nothing" assignment could not accurately represent the pedestrian system in which so many alternative routes between points are available and where both regular and sporadic congesuon effects route choices.

Accordingly, speed/flow relationships were built into the time trees so that as the shortest route became congested passengers diverted to the longer, but now quicker, route. The TRANSPORT modelling suite had been used in this and the earlier Bank model, and had performed reasonably well. But the limitations in terms of specifying speed-flow relationships and the general inefficiency of the process led to the choice of the SATURN congested assignment model as the basis for further pedestrian modelling.



Stations are represented by simple link node networks

In the pedestrian modelling work the morning peak one hour was modelled as being the worst period for congestion, with benefits (of schemes) factored to represent the full day. Throughout this peak hour demand was assumed to present a flat demand profile (due to the limitations of the model), although checks of the peak of the peak five minutes were carried out to examine the capacity of facilities such as UTS gates to handle peak flows.

As before the network was described in terms of links with a given length and width, the length determining the walk time in free flow conditions, while the width determined the capacity of the link. In addition, the walking speed at capacity and the maximum capacity were derived from Fruin and observation, while a power term describing the manner in which congestion affects walk speeds was developed from observations of pedestrian behaviour.

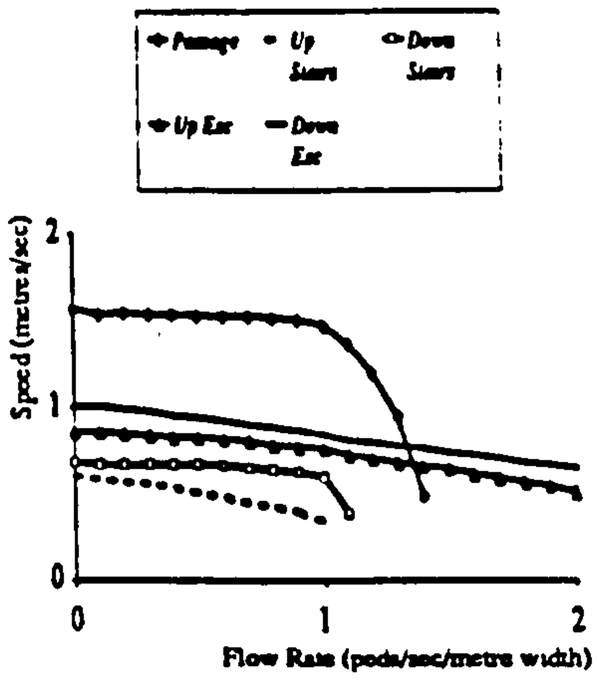
PEDROUTE

The PEDROUTE package interfaces the SATURN model, as used in earlier work, with LUL's own pedestrian simulation model, SCM (Station Capacity Model). SCM was developed as a means of evaluating station improvement options in terms of passenger time savings, but it lacked the route assignment capabilities of SATURN. The linking of the two models was designed to address the limitations of both approaches. The package has been enhanced

[1] Fruin Pedestrian Planning & Design, 1971

by the addition of a 3D colour graphics program that forms a powerful and effective analytical and presentational tool for use with the model output, enabling rapid visual analysis of the test results.

The simulation approach to the model is vital in the context of planning for highly peaked pedestrian flows, as the *platoon* movements of passengers from trains to station entrances or other platforms places great pressure on the pedestrian facilities for relatively short periods of time. Previous models based upon average values over, typically, the peak hour, would not identify these short-lived peaks.



Speed-Flow relationships used in the model.

Note: Escalators flow rate is in pedestrians per second per escalator.

The model simulates the arrival and departure of trains, allowing the impact of train loads of passengers on platforms, escalators, stairs and passages to be examined - for example, where an island platform is occasionally fed simultaneously by two arriving trains. The model simulates the build-up and decay of queues throughout the station complex. This reveals the effects of blocking back which can interfere with other pedestrian movements, and may ultimately 'lock' large parts of the station.

The passenger volume/capacity relationships for different elements of the station complex are critical to the success of the PEDROUTE simulation. These have been derived from extensive surveys of passenger movements at LUL's existing stations and provides a sound behaviourial basis for the analysis.

PEDROUTE produces a detailed simulation of the movements of pedestrians around the station, and assesses their journey times, and the delay and congestion they experience. Congestion is categorised by service level, a measure based upon the density of people using a pedestrian block at any one time. Information concerning the train service at each platform is also given, as is an assessment of the social costs incurred in terms of free movement in the system, waiting time and delay, which form the basis for evaluating congestion relief options.

The graphical package constructs a 2D or 3D image of the station from a series of blocks, which can then be manipulated so as to view the station, or any part of the station, from any angle and at any level of magnification. Data can then be plotted on this base diagram, either in terms of values or coloured bands, with control over the period examined down to as little as an one minute interval. In addition, other data series can be plotted in the form of bar or line charts that indicate changes in key measures throughout the simulation period.

Simulations can be carried out for any time period specified by the user and for any number of days. A typical model run might involve simulating pedestrian flows in 5 minute intervals over a 3 hour period on 5 days, the days varying due to a random function which varies train arrivals, passenger turn-up, train loads, late-running and so on, within specified ranges. The output file from the model run can be interrogated using the graphics facilities to determine average and worst case conditions over the whole station or at particular bottlenecks.

CONGESTION RELIEF

LUL is conducting a systematic programme of congestion relief at stations in central London. For each station a range of improvement options are produced for easing conditions experienced during peak periods. These involve both small scale improvements, such as increased escalator capacity at key bottlenecks, and radical solutions such as a completely new ticket hall.

XXX

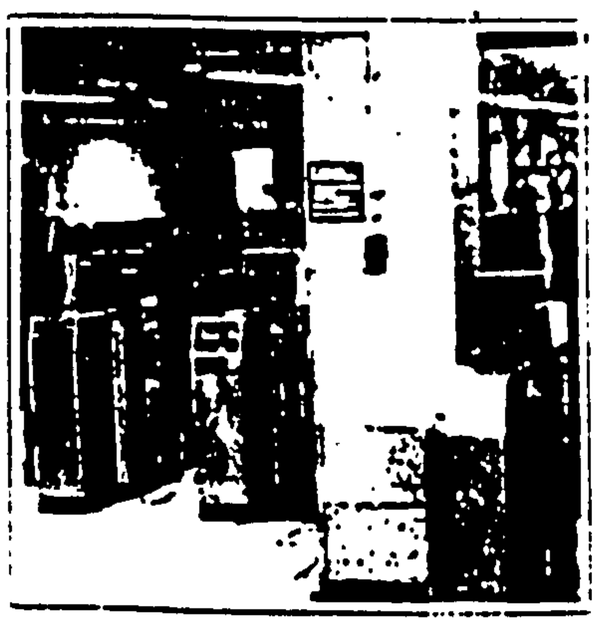
C

PEDROUTE produces the information necessary to enable the selection of the most effective option, by determining the value of the time saved by passengers for comparison with the costs of the improvements and ensuring specified station standards are maintained.

The arrival of Crossrail at Paddington will result in a substantial increase in passenger movements.

The model is first used in validation mode to check that an accurate representation of the current movement of passengers in the station can be simulated, model results being compared with surveys of passenger flows and journey times. The coded station layout within the model is then amended to reflect the changed layout including the proposed improvements under consideration, along with any changes in train frequency, capacity, expected traffic growth, etc, that may be planned within the time frame being examined. Future passenger flows can then be simulated both *with* and *without* the measures being tested to provide a "do-nothing" and one or more "do something" options.

The key feature of the model is its ability to represent short-lived peaks in passenger demands, thereby providing a thorough test of the capabilities of pedestrian facilities at the time it matters and assessing the standard of service experienced by passengers using the station. For example, although an escalator has a theoretical capacity of around 7,200 passengers per hour, this is based on a steady stream of people arriving at the base of the escalator matched to the throughput of the escalator. Where the arrival rate is highly peaked, periods of excess capacity will arise ensuring the effective capacity may well be only a fraction of the theoretical capacity.



Knightsbridge station was one of the stations included in the early stages of the station upgrade programme

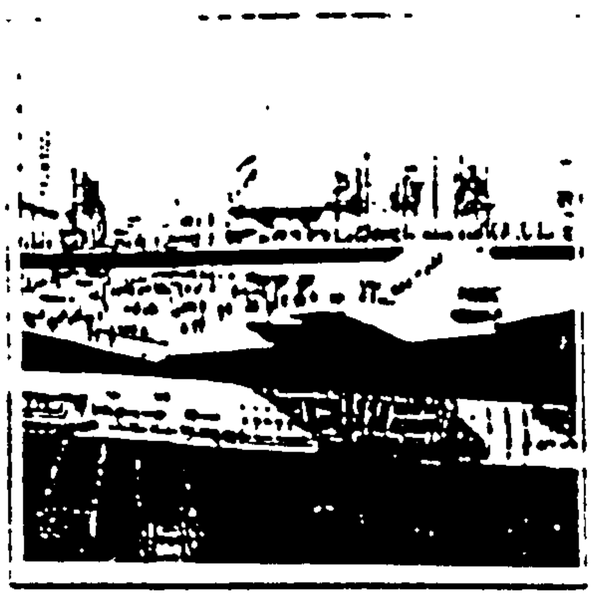
The results from the model include the passenger time and congestion relief savings that may be expected from implementing the improvements, together with information about the passengers groups which would benefit from the improvements. The time savings are then set against the costs of the improvements to produce the economic net present value. By systematically analysing a range of feasible improvement measures, the preferred station improvement strategy can be identified.

Waterloo Underground station was one of the first major applications of the model. Construction of the Waterloo International Terminal (for the Channel Tunnel services) and the Jubilee Line Extension are both on-going and will result in a major increase and reorientation of traffic flows within the existing station. The station layout was assessed against the forecast passenger flows and using the upgraded service patterns expected to examine what would happen without major capacity expansion. This demonstrated that severe and protracted congestion would result during both peak periods if no action was taken to upgrade the existing station fabric.

Various combinations of measures were developed and assessed using PEDROUTE to study the influence particular measures had individually and in combination. Thus complementary and conflicting options are identified, enabling the options to be ranked and evaluated. An improvement strategy was then developed and justified in economic terms, which forms the basis for the Waterloo Underground station of the future.

DESIGN OPTIMISATION

PEDROUTE has also been used to 'drive' the design process for stations on London's two major new rail projects - the Jubilee Line Extension and Crossrail. In both cases the model is being used to develop economically optimal and operationally viable designs.



Tottenham Court Road is set to develop as a major interchange, with Crossrail and Chelsea-Hackney services connecting with the existing Central and Northern Lines.

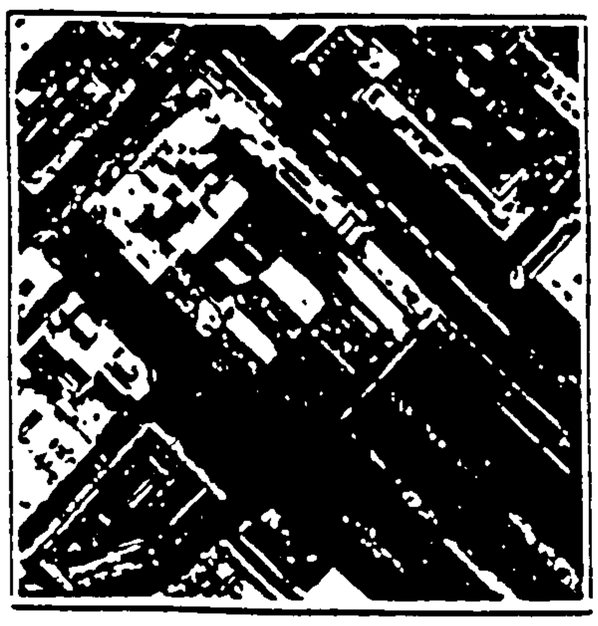
Preliminary designs for the new or upgraded stations were produced using standard station design parameters which indicate average throughput and requirements as laid down by LUL. These designs were then tested using PEDROUTE against passenger forecasts to identify potential boulenecks in the design, and to assess the standards of service provided to passengers using the station. The results were fed back into the design process so that amendments in the layout could be devised. This iterative process was repeated a number of times until a satisfactory layout was produced.

Escalator capacity has been one of the key issues in the design of Canary Wharf station on the Jubilee Line Extension. Planned as the showcase station on the line, but with peak passenger numbers expected to make it the busiest single line station in London, the design for the station had to cope with a number of constraints, not least the narrow "box" in which the station must fit to avoid the deep piles associated with the buildings being constructed above and the old wharfs. Modelling of the preliminary design revealed major congestion caused by the limited and poorly located escalator capacity at the western end of the station, resulting in a mean delay per passenger for the whole morning peak of over 100 seconds.

Suggested amendments have seen the evolution of the design so that it now incorporates two western ticket halls - one on top of the other; the resiting of the eastern ticket hall some 100 metres further west (to abstract some of the heavy westbound passenger flow); and the orientation of all escalators to face west. Consequently, the mean passenger delay has fallen to just 9 seconds per passenger for the same period.

SAFETY ASSESSMENT

The LUL review into safety which followed the Kings Cross Underground fire confirmed the need to be able to test and validate the design of stations to demonstrate that they meet the required standards for passenger evacuation. PEDROUTE was identified as a potential solution to the problem of accurately assessing the capabilities of a station, and enhancements were introduced to allow the model to respond to the differing circumstances that may apply in an emergency, not least the possibility that particular escape routes might be unavailable due to the location of the fire itself.



Aerial view of Paddington and Eastborne Terrace, the site of the Crossrail station

One problematic aspect of the model enhancement was establishing the changes in pedestrian behaviour that would probably occur in evacuation conditions, especially where the source of the fire is not visible to the passenger (i.e. perceive no danger, route choice may lead to the source of the fire, and a reluctance to use emergency escape routes). In addition, it was assumed that passengers evacuating a platform would select the nearest exit (emergency or normal) if they were on the affected platform, and thereafter use the quickest route to the surface based on the route conditions and crowding resulting.

Two main approaches to the assessment of evacuation procedures have been applied. The first is to generate a series of starting scenarios using PEDROUTE in its conventional congestion relief mode, with the model assessing what would happen if an emergency was to occur at that point in time. The second approach is to specify a "worst case" scenario, as agreed with the regulatory authorities, and model the evacuation of the station using these parameters. In practice both approaches are often used, with the operational scenarios applied to determine a distribution of evacuation times, which can then be compared with times resulting from the "worst case" scenario.

The modelling approach offers significant benefits in the assessment of evacuation potential, as it provides detailed information concerning the performance of the station design in a wide range of conditions and enables strategies for dealing with the problems to be developed and tested. Conversely, the model may also be used to establish appropriate standards for evacuation, in the light of realistic pedestrian behaviour under emergency conditions, and with reference to the prevailing guidelines which are based upon standards developed for application in American transit stations (and derived by the National Fire Protection Association).

The exercise has also enabled further refinement of the PEDROUTE model so that the route assignment element, previously provided by SATURN, is now an integral part of PEDROUTE. This permits dynamic reassignment based on SCM's assessment of journey times between points in the station, providing a more accurate measure of pedestrian movements when confronted by short-term and unexpected delays.

FUTURE DEVELOPMENTS

Although the model has been developed for use in examining congestion within a station, the general principles on which it is based are applicable to many other situations. Most obviously it has applications in other types of transport terminal such as bus stations and airports, where similar questions about passenger comfort, delay and safety arise. Similar considerations arise in the design of any large building or stadium (as witnessed by the Hillsborough Stadium disaster), as well as in restricted pedestrian malls. For example, at the micro level the model has been used to assess the viability of the design of an university lecture theatre complex, where there was concern that lecture changeovers could not be accommodated within specified time limits.

With the model's use in an ever wider range of situations, the model, and the techniques for applying it, are becoming increasingly sophisticated. PEDROUTE offers planners the means to optimise building layouts from the twin perspectives of operational efficiency and safety. Experience shows that major changes in conventional design practice have been justified in cost benefit terms, and that initial design guidance can be improved to reflect the knowledge gained from examination of earlier designs, thus shortening the design process. The consequences for station design are considerable.

In railway and bus stations, airport terminals and building complexes large pedestrian movements are the common experience. So too are delay, frustration and, increasingly, concerns for safety. London Underground Limited (LUL) as the carrier of 2.5 million passengers a day and in the wake of Kings Cross station fire is seeking to improve standards of safety at stations in order to cope with large volumes of pedestrians.

LUL has, in collaboration with Consultants, Halcrow Fox and Associates (HFA), developed a pedestrian simulation model, PEDROUTE, to assist with the task of improving passenger service levels in the London Underground. PEDROUTE simulates pedestrian behaviour in free flow and congested conditions and can quantify the benefits of changing levels of congestion. It has three distinct roles:

developing and evaluating the performance of options to increase capacity at congested stations. Then providing the economic justification for such new investment.

Optimising the design of new station facilities in economic and safety terms to ensure their suitability for predicted passenger volumes.

Demonstrating compliance with safety regulations, for example in meeting the required evacuation time from a station.

This approach to station planning and design has been successfully applied at over 25 stations, ranging from some of the largest interchanges in London at Waterloo and Liverpool Street to smaller stations such as Knightsbridge and Bayswater.

PEDROUTE

The PEDROUTE package interfaces a dynamic assignment model (SATURN) with a pedestrian simulation model. The package has been enhanced by the addition of a 3D colour graphics program that forms a powerful and effective analytical and presentational tool for use on PC compatible microprocessors.

The simulation approach is vital in the context of planning for highly variable and peaked passenger flows; for instance "platoon" movements of passengers from trains to station entrances or other platforms place great pressure on pedestrian facilities for short periods of time. The model simulates the arrival and departure of trains, allowing the impact of train loads of passengers on platform, escalator/stair and corridor facilities to be examined. The model simulates the build-up and decay of queues throughout the station complex. This reveals the effects of blocking back which can interfere with other pedestrian movements, and may ultimately "lock" large parts of the stations.

The passenger volume / delay relationships for different elements of the station complex are critical to the PEDROUTE simulation. These have been derived from extensive surveys of passenger movements at LUL's existing stations and provide the sound behavioural basis for analysis

PEDROUTE assesses journey times and delay and congestion experienced by pedestrians. Congestion is categorised by "Service Level", a measure based upon the density of people using a pedestrian facility. (Derived from concepts originally formulated by J J FRUIN). Information concerning the train service at each platform is also given , as is an assessment of the social cost incurred by passengers; in terms of free movement time, waiting time and delay. This forms the basis for evaluating congestion relief options.

The graphical package constructs a 2D or 3D image of the station , which can be manipulated so as to view the station or any part of it, from any angle and at any level of magnification. Data from the simulation can then be plotted on this base diagram either in terms of values or coloured bands, with detail down to one minute intervals.

Simulations can be carried out for any time period and for any number of days. For example, a typical model run might involve simulating pedestrian flows over the 3 hour peak period for five separate days.

CONGESTION RELIEF AT EXISTING STATIONS

LUL is conducting a systematic programme of congestion relief at stations in Central London. For each station LUL architects and Engineers produce a range of improvement options for easing congestion during peak periods. These involve both smallscale improvements , such as increased escalator capacity at key bottle-necks, and radical solutions such as a completely new platform or Booking Hall. PEDROUTE produces the information to assist in the selection of the preferred option by comparing the effectiveness of each option in reducing congestion and by calculating the time saved by passengers.

The model is first used in validation mode to check that an accurate representation of the current movement of passengers can be simulated; model results being compared with surveys of passengers flows and journey times. The station layout within the model is then amended to incorporate the improvement under consideration, along with any changes to train frequency, capacity etc that may be planned. Future passenger flows are then simulated.

Green Park station is one example wnere PEDROUTE has been applied. Construction of the Jubilee Line extension will lead to a major increase and reorientation of traffic flows within tne station. Assessment of the existing station layout showed that severe and protracted congestion would result if no action was taken. As a results of evaluating options a new package has been devised comprising a new ticket hall and interchange passage. This could be justified in economic terms and has been submitted for Parliamentary authority.

DESIGN OPTIMISATION AT NEW STATIONS

PEDROUTE has 'driven' the design of stations on London's two major new rail projects - the Jubilee Line Extension and Crossrail. In both cases the model is being used to develop designs which are both cost-effective and operationally viable.

Preliminary designs for the new or amended stations were produced using standard station design parameters. These designs were then tested by PEDROUTE against passenger forecasts to identify "bottle-necks" in the design, and to assess the standards of service provided to passengers using the station. The results were fed back into the design process so that amendments to the layout could be devised. This iterative process has been repeated a number of times until a satisfactory layout was produced.

Escalator capacity has been one of the key issues in the design of the Canary Wharf station on the Jubilee Line Extension. Planned to be the show-case station on the line, but with peak passenger numbers expected to make the station the busiest single line station in London, the design for Canary Wharf has had to cope with a number of constraints, not least the narrow "box" in which the station must fit. Modelling of the preliminary design revealed major congestion caused by the limited escalator capacity to the Western ticket hall, resulting in a mean delay of over 100 seconds.

The design as it has been amended and now put forward for Parliamentary authority incorporates two western ticket halls - one on top of the other with other major consequential changes. As a result the mean modelled delay has been reduced to 9 seconds.

SAFETY ASSESSMENT

The LUL review into safety which followed the King's Cross Underground fire confirmed the need to be able to test and validate the design of stations in order to demonstrate that they meet required standards for evacuation. Enhancements have been introduced into PEDROUTE to allow it to respond to the different circumstances that may apply in an emergency.

Two main approaches to the assessment of evacuation procedures have been applied. The first is to generate a series of starting situations using PEDROUTE in its conventional congestion assessment mode, with the model assessing the numbers and locations of people needing to be evacuated. The second is to generate a "worst case" scenario, as agreed with the regulatory authorities. In both cases PEDROUTE is used in evacuation mode to assess for each scenario the distribution of evacuation times from various parts of the station.

The modelling approach offers significant benefits in the assessment of evacuation potential, as it provides detailed information concerning the performance of station design under a wide range of situations and enables alternative strategies for dealing with the problems to be tested quickly and easily.

FUTURE DEVELOPMENTS

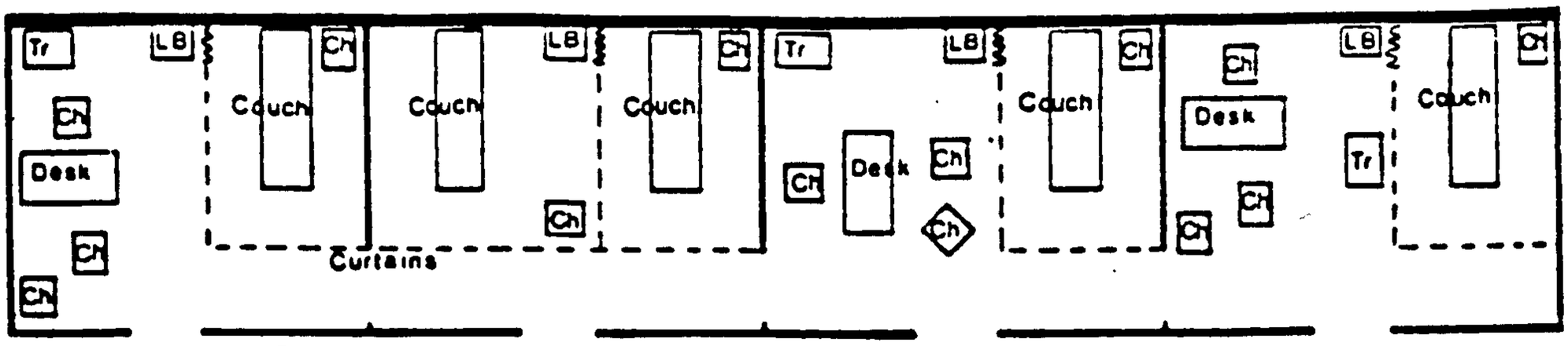
Although the model has been developed to examine congestion within a station, the general principles on which it is based are applicable to many other situations. Most obviously it has applications in other types of transport terminals such as bus stations and airports, where similar questions of pedestrian comfort, delay and safety arise. Similar considerations are important in the design of any large building or stadium as well as in restricted pedestrian malls. For example the model has been successfully applied to the design of a University lecture complex as well as examining problem with the HADJ in Saudi Arabia

LUL and HFA are continuing to collaborate to provide further enhancements to PEDROUTE.

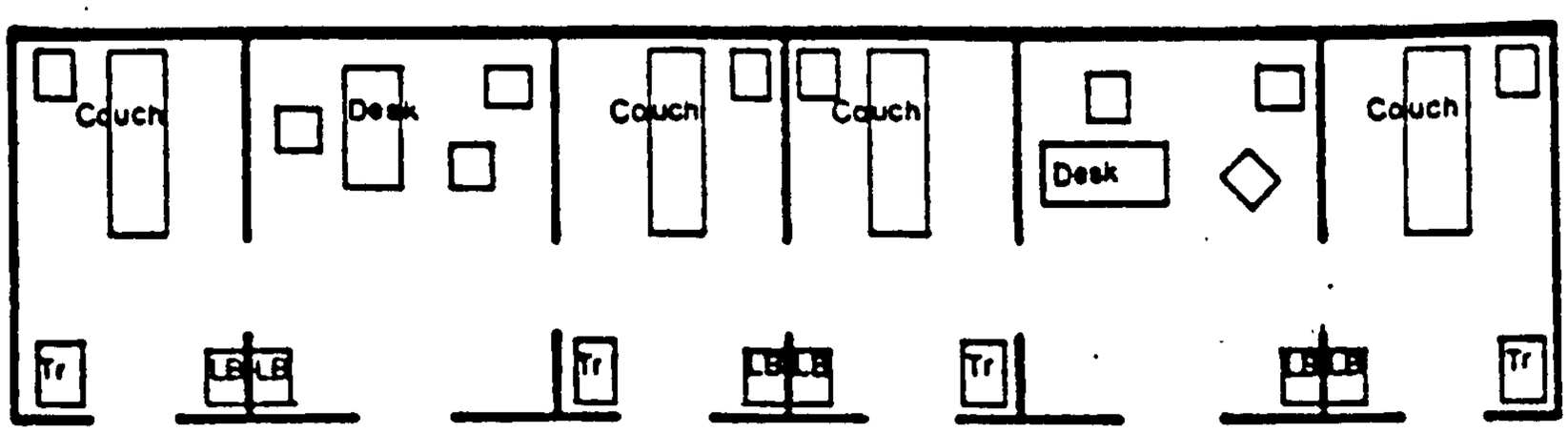
PEDROUTE is marketed by HALCROW FOX and ASSOCIATES

Appendix D

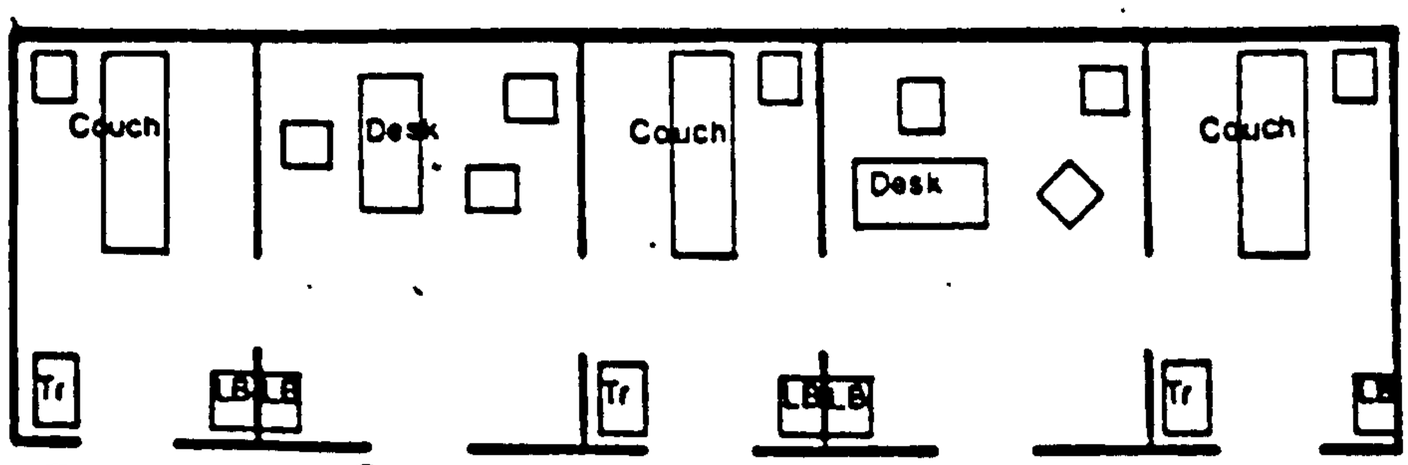
EXTRACTS FROM HOSPITAL SURVEYS



Group of 4 combined consulting examination rooms



Group of 2 consulting with 4 examination rooms

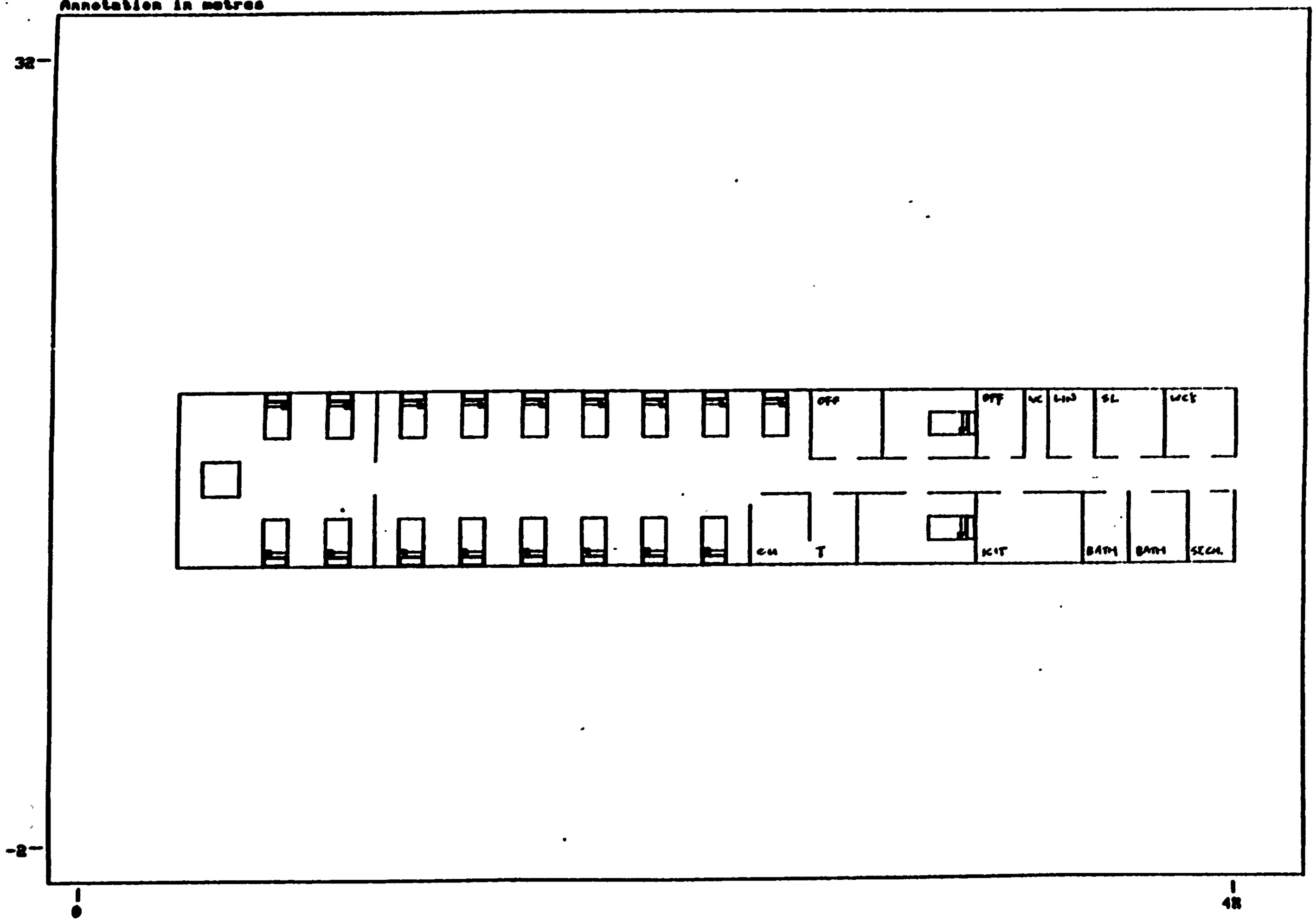


Group of 2 consulting with 3 examination rooms

fig 6 C/E Rooms Suitable for 18 doctor sessions per week

D

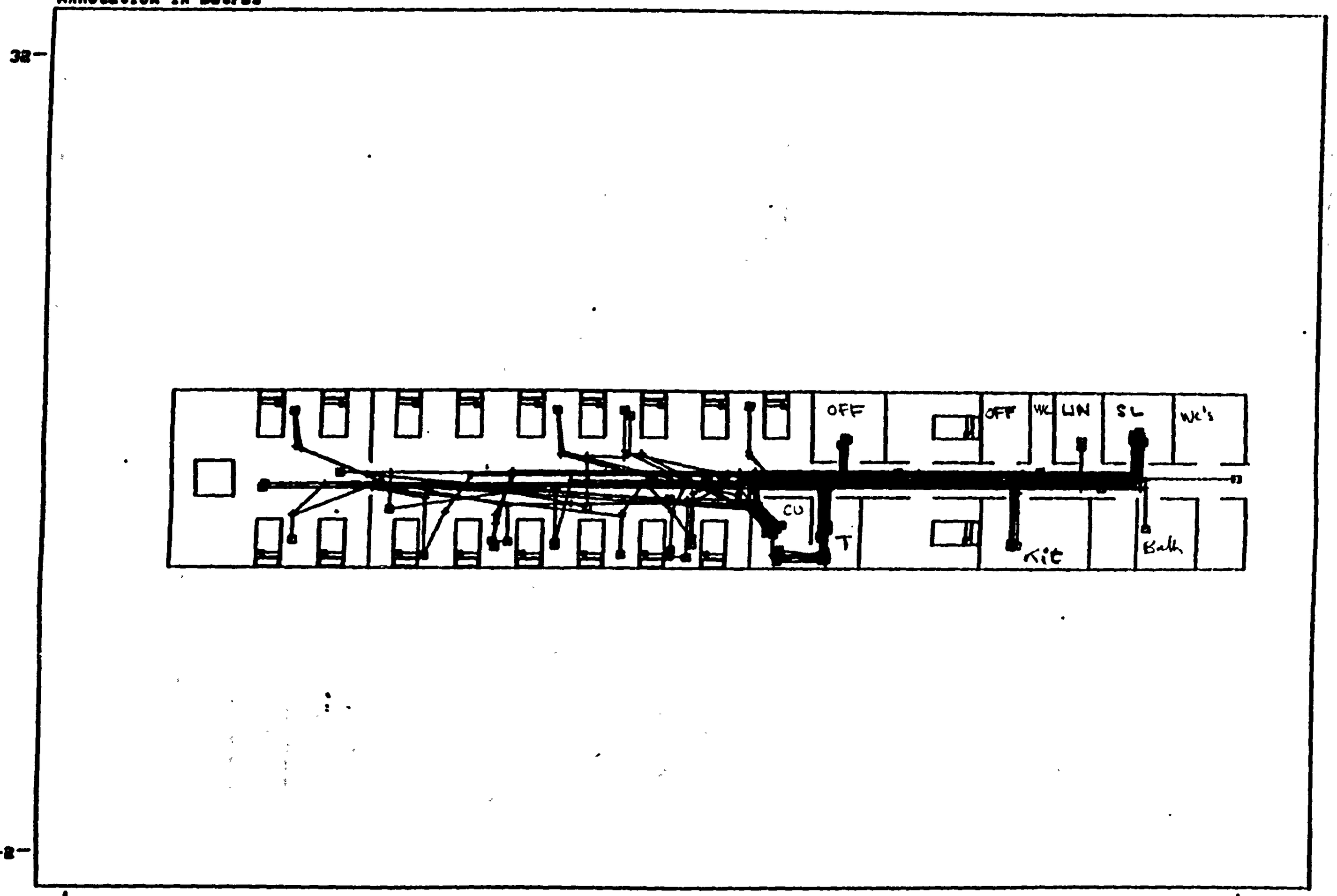
WARD-ANALYSIS-PROGRAM Scale 1:150 WED 13 May 81 11.57
Annotation in metres



No grid Nothing is drawn

Plan 10 Ward 05X

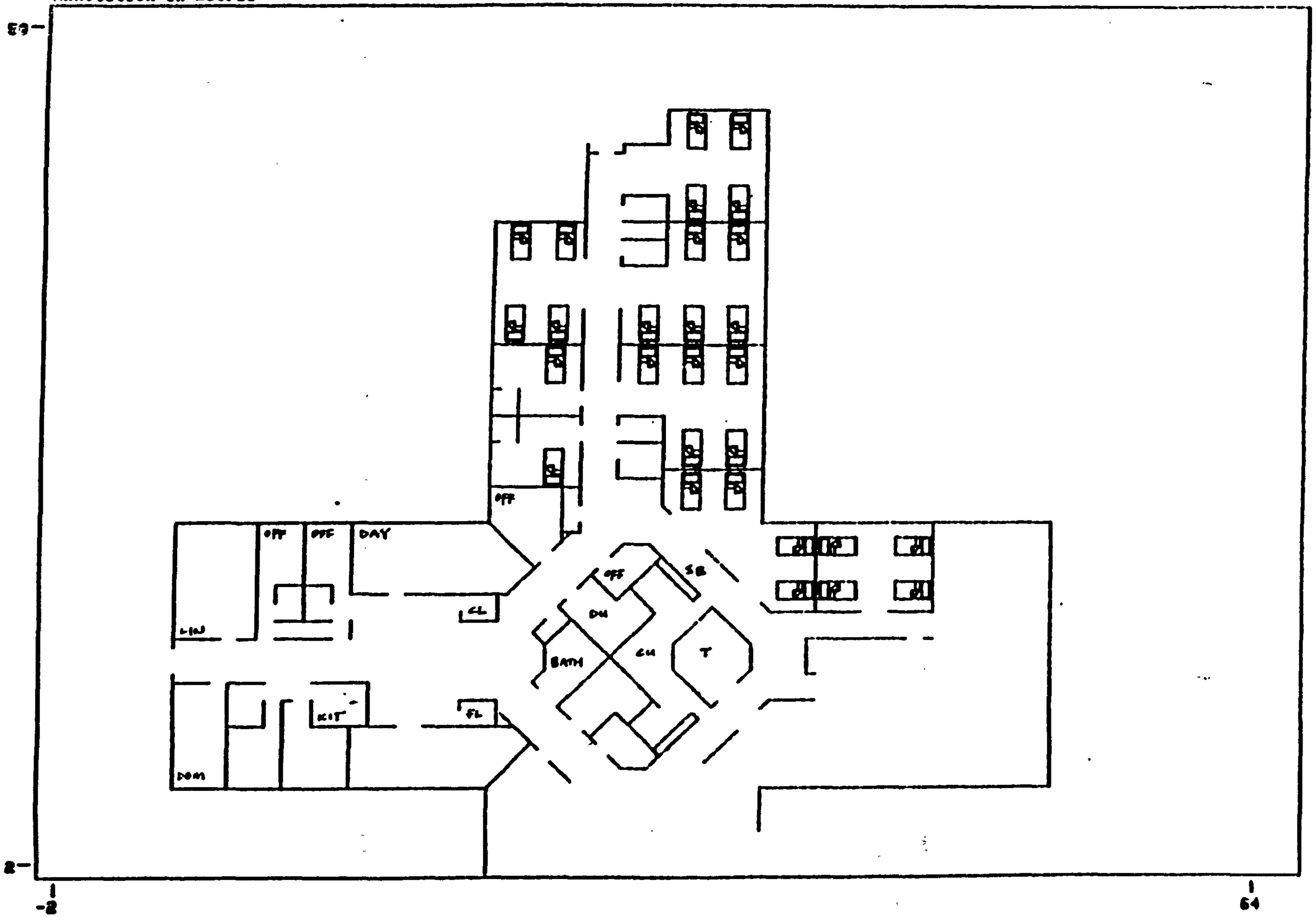
UARD-ANALYSIS-PROGRAM Scale 1:150 WED 13 May 81 11.57
Annotation in metres



No grid Nothing is drawn

Trail TR05X4 1.2

UARD-ANALYSIS-PROGRAM Scale 1:200 WED 13 May 81 14.35
Annotation in metres



No grid Nothing is drawn

Plan 8 Ward 04Z

AIL NAME? TR04Z5

AIL NAME?

RD = 04Z
TE = 31-10-80
IFT = A
RSE HOURS = 3.000

URSE	PATIENTS TIMES HOURS S & H	H ONLY	S ONLY	DISTANCE	% TRAVELLING TIME
	3.36	3.09	0.00	740.06	5.79
	3.36	3.09	0.00	740.06	5.78

DE NAME	PATIENTS TIMES HOURS S & H	H ONLY	S ONLY	NO. OF VISITS	% TIME OF NURSE
ED 26	0.91	0.97	0.00	5	15.67
FF WARD	0.00	0.00	0.00	2	17.52
3	0.05	0.45	0.00	5	2.27
4	0.03	0.03	0.00	1	0.25
R	0.00	0.00	0.00	1	0.96
ED 27	1.83	1.33	0.00	10	26.39
ATH1	0.24	0.00	0.00	1	3.16
U	0.00	0.00	0.00	5	6.92
IT	0.00	0.00	0.00	1	0.77
U	0.06	0.07	0.00	5	1.07
7	0.05	0.00	0.00	1	0.42
ED 25	0.07	0.20	0.00	1	2.17
B	0.27	0.01	0.00	3	2.33
ED 28	0.35	0.03	0.00	2	3.14
REAT	0.00	0.00	0.00	1	0.09
FF	0.00	0.00	0.00	1	6.11
	3.36	3.09	0.00	45	94.22

PATIENT	% TIME WITH NURSE
26	15.67
27	34.55
25	2.17
28	3.14

ACTIVITY	% TIME OF NURSE
8C	31.14
20	17.52
4WS	0.43
9M	0.96
2L	1.74
8G	15.75
2D	0.34
2I	1.20
11	5.91
2M	0.77
3F	5.49
12	0.51
16L	0.42
2H	0.09
14	1.90
3M	1.01
28	2.13
9P	0.42
13	0.37
9R	6.11

FIG 1 TRAIL ANALYSIS-TRAIL TR04Z5

TABLE 10 MEAN VALUES FOR TRAVEL AND CONTACT PER WARD TYPE AND HOSPITAL

	HOSPITALS			
TRAVEL TIME (%)	03	04	05	ALL
OPEN	8.1	6.3	9.7	8.0
DIVIDED	8.7	7.7	7.1	7.8
TRAVEL DISTANCE (m)				
OPEN	1030	878	1064	991
DIVIDED	1083	865	843	930
CONTACT: SEE & HEAR				
OPEN	20.42	18.00	6.09	14.84
DIVIDED	3.71	4.50	3.33	3.85
HEAR ONLY				
OPEN	25.17	12.25	10.25	15.89
DIVIDED	8.32	7.34	9.17	8.28
AVERAGE				
OPEN	22.80	15.13	8.17	15.37
DIVIDED	6.01	5.92	6.25	6.06