TITLE : ADVANCED APPLICATION SOFTWARE FOR SPECULATIVE HOUSING COMPANIES.

AUTHOR : NEIL EWIN Bsc (Hons)

SPONSORING ESTABLISHMENT : SHEFFIELD CITY POLYTECHNIC

COLLABORATING ESTABLISHMENT : P.HASSALL LTD

DATE OF SUBMISSION : JULY 1985.

This Thesis is submitted as partial fulfilment for the requirements of a Master of Philosophy degree to the Council for National Academic Awards.

RESEARCH ABSTRACT

A wide variety of packaged applications systems for builders have recently become available, however the diversity and complexity of individual building companies has meant that many prospective users have been unable to find packages which precisely meet their needs.

The high cost, and long lead times associated with traditionally produced bespoke software has made many builders reluctant to become involved. Some builders who have commissioned bespoke software have had unsatisfactory experiences, where systems have been unreliable, or have not fully met the original objectives.

This research project was conceived to test the hypothesis that, through the use of advanced software development concepts, bespoke software could be made more acceptable to builders with smaller budgets.

The research was conducted in two stages; firstly to construct an advanced software development system, and secondly, to apply that system in a typical applications area.

The software development system (DB4GL) combines current analysis and design techniques, program generation tools, a powerful data base mechanism and a documentation facility, to yield a well structured approach to software development throughout the system life cycle.

DB4GL was used to develop a complex integrated estimating and cost control system (Spec-Builder) suitable for an actual speculative house building company.

Spec-Builder was developed over a relatively short time period, thus supporting the research hypothesis. As much of the software was generated from proven DB4GL procedures, the resulting system was reliable. The development techniques encourage extensive user participation and early appraisal of the system which can be amended easily to fulfill the user's precise requirements.

HYPOTHESIS

The shortage of suitable specialist applications packages has meant that many small and medium sized companies within the building industry have been unable to fully utilise computer technology.

It is said that through the use of advanced software development concepts and techniques, bespoke software to meet the needs of individual companies can be produced quickly, cheaply, and reliably. It would thus be possible to make bespoke software, and so computer technology, more widely available to building companies.

The purpose of the following research project is to test that hypothesis.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my research supervisors, Dr Frank Poole, and Mr Ray Oxley, for their help throughout this project. Both were exceptionally generous with their time, and provided invaluable guidance through their respective fields.

I must also thank my external supervisor, Mr G. Thorpe, Managing Director of P. Hassall Limited for his interest and guidance, and Messrs Steve Howard, Martin Ward, and John Brookes of the same company, for their time and patience during the analysis phases of this project.

I gratefully acknowledge the Local Education Authority of Sheffield City Council for their financial support during the research term.

CONTENTS

CONTENTS (CONT).

CONTENTS (CONT).

APPENDICES

CHAPTER 1.

1. INTRODUCTION

1.1. COMPUTERS IN THE BUILDING INDUSTRY

Companies in the building industry, particularly those firms of small and medium size, are generally seen as being slow to adopt computer technology [Nat01]. This slowness is usually attributed to the industries inherent conservatism. On closer analysis, this charge seems unfounded.

The industry, always a barometer of the economy, is currently in recession, and building companies are looking for ways to become more competitive. The increased efficiency and management information that can be acheived through computerised systems has been recognised, and there is an increasing demand for such systems within the industry.

In the building industry, the size of the company is not always a reflection of its complexity. For example a small building business may undertake a large number of small jobs, which will produce a modest turnover (say £500,000). However the volume of work creates a heavy load of paper work [Col01]. Small building companies may therefore require complex computer systems.

It is only since the advent of cheaper mini and micro computers that builders, other than the larger firms, have been able to afford computers of sufficient power to meet these requirements. Even then, builders have had to wait for the necessary software to become available. There are now a large number of application software packages on the market, These range from complex integrated management information systems such as 'Contractor' from ICL, and 'Siteman' from Systime, down to simple estimating programs which run on the most basic micro computers.

Despite the abundance of building applications packages (the quarterly magazine 'Construction Computing' includes a 16 page software directory listing such packages) many builders are still having difficulty obtaining systems to meet all their requirements.

The problem with the package approach for supplying software to the building industry is that there are considerable variations in the way individual building companies are run [Wes01]. These differences can be reflective of management style, organisational structure, or the type of construction business carried out.

Most building applications packages are designed for the general contractor, the most prevalent type of builder. others, such as speculative house builders, have a much smaller selection of packages available.

As the systems required are complex, few builders can afford to have bespoke software written, so if packages are not available for certain applications, a builder has little chance of fully exploiting computer technology.

1.2. COMMON PROBLEMS WITH BESPOKE SOFTWARE.

Using the traditional programming languages and methodologies, the production of bespoke software is a labour intensive exercise. The specialist labour required is expensive, and the production of software slow [Mar01], hence the resulting software is expensive.

Even after a long wait, and considerable investment, the system implemented may not entirely meet the builders requirements. However the cost of rewriting the software is often prohibitive, consequently the builder 'makes do' with the system offered.

These problems are not necessarily the fault of the people producing the application system nor the users themselves. What is at fault are the systems for producing and agreeing the systems specification and its implementation.

The specification is typically produced by a systems analyst who will gather facts about the proposed system from the builders, assess the system requirements, and finally draw up a system design and specification. Gane and Sarson [Gan01] identify 5 problems with the traditional methods used to acheive a system specification :-

i) The analyst may find it difficult to understand the business. Each business has its own terminology, (which the analyst must come to terms with), and existing systems may include a certain amount of 'intuition' which the users may have difficulty explaining.

ii) The user community may not be sufficiently aware of computer technology to be able to visualise, or explain the sort of system they want.

iii) The analyst may be overwhelmed with detail, and find it difficult to abstract the important facts from those which are less important.

iv) The system specification effectively forms the basis of a contract between the user department and the system implementors. However the document is usually written for the use of the programmers who will implement the system, and so may be almost meaningless to the users.

v) If the analyst is careful and produces a specification that the users do understand, it may not be detailed or accurate enough for the system implementors. Thus two specifications are required, one for each group.

From the builders point of view, the ideal situation would be to see exactly what the system looks like before agreeing to its purchase and implementation. This approach can be adopted when buying a standard applications package, but

when procuring bespoke systems the software producer requires the full commitment of the builder before making any substantial investment in software production.

1.3. NEW CONCEPTS IN BESPOKE SOFTWARE DEVELOPMENT.

New tools, techniques, and methodologies are becoming available which will allow bespoke software to be produced in a more satisfactory way than is currently achieved. These advances include highly structured systems analysis methodologies, and so called 'fourth generation' systems which facilitate prototyping and application program generation.

These advances allow the systems analyst, or analyst-programmer, to conduct more accurate investigations, and then quickly, and cheaply, produce prototype systems which can be shown to the user as a tangible illustration of what the system will look like. When the user is satisfied with the prototype system, the analyst can use the prototype as a part of the specification for the 'live' application system, which can be (at least partially) generated by an application generation system.

The prototyping and applications generation stages can be combined so that the final prototype is adopted as, or forms the basis of, the 'live' applications system.

1.4. THE OBJECTIVES OF THIS RESEARCH PROJECT.

At the outset of this project, there were no commercially available micro computer based software development systems with sufficient power to test the research hypothesis. It was therefore necessary to research, design, and implement an advanced software development system. The resulting system was then applied to produce bespoke software for an application typical of the housing sector of the building industry that is currently facing a shortage of software.

In chapters 3-5, an application software development system called 'DB4GL' (an acronym for Data Base 4th Generation Language) is discussed. This system combines prototype and application generation, with a structured system analysis and development methodology. DB4GL was jointly developed by the Departments of Building and Computer Studies as part of this research project.

The DB4GL system aims to support the development of software throughout the project life cycle. In particular it :-

i) provides a framework for systems analysis - giving structure to the analysis, and a formal method for recording the system specification.

ii) allows the analyst to develop system prototypes quickly from the specifications entered into the system.

iii) allows the analyst to amend the design and re-generate the prototype easily.

iv) aids the production of a formal specification from the information used to produce the prototype.

v) provides the basis for the 'live' system. Indeed in simple applications, the final prototype can be adopted as the 'live' system.

vi) provides comprehensive system documentation automatically.

vii) provides a basis and test bed for future system maintenance and development.

Speculative house builders were identified as one class of specialist builder for whom there was a shortage of suitable software, in particular, estimating and cost control packages. This shortage was demonstrated by the fact that the speculative house building company who collaborated in this project had commisioned bespoke software for these applications. The bespoke estimating system was developed during the course of this project using conventional methods.

DB4GL was used to develop an integrated estimating and cost control system for such a company. The resulting package was named "Spec-Builder" for ease of reference and is discussed in chapters 6,7 and 8 below.

Spec-Builder is a complex system. Within the estimating module, house types are described using hierarchical bills

of quantities. The bills are easily maintained, and from them house type estimates are calculated.

Site estimates are built up from an accumulation of house type estimates, and an estimate of costs pertaining to the site generally, for example roads and sewers. These site costs are apportioned to each plot. The the final estimate is expressed in terms of costs per plot. The plot costs are grouped under a variable number of headings, or cost categories as defined by the user.

The cost control module accepts input from 'invoices', 'goods received notes', and 'labour allocation sheets'. A simple code structure is used to store the costs in the same format as the estimate, that is by cost category and plot. It is then possible to print comparisons between costs and estimates at plot and site level.

The conclusion drawn from this thesis shows that given advanced software development systems, such as DB4GL, complex bespoke software, such as Spec-Builder, can be produced in a more satisfactory way than is currently acheived. Bespoke software would therefore become available to a much wider spectrum of building companies and specialist applications.

CHAPTER 2.

2. METHOD OF RESEARCH.

The research hypothesis that, through the use of advanced software development concepts and tools, it is possible to produce bespoke software to meet the needs of individual building companies in a much quicker, cheaper and satisfactory way than is currently practiced, can most easily be supported through an example.

In order to develop such an example it was necessary to investigate modern software development concepts and tools, and specify and develop such as system. Once developed, these concepts and tools were applied to an example typical of the problem areas described in the introduction. The example selected was the estimating and cost control function in a medium sized speculative house building company.

The fact finding process continued throughout the project, using a variety of methods and sources;

i) Literature searches are a traditional method for discovering information on research already carried out on and around the subject area. An initial search through the building and computing abstracts revealed much background

information, but little specific information concerning the type of software tools to be developed, or on estimating and cost control systems specifically for speculative house builders.

The lack of relevant information concerning software tools was probably caused by two factors. Firstly they form a comparitively new field, and secondly most of the research that has been carried out has taken place in commercial organisations who are developing products, and are therefore reluctant to publish their research results. Although estimating is a much researched and controversial subject, it would appear that very little published research has been carried out in the speculative housing sector of the building industry.

The literature search, although concentrated at the beginning of the project, was continued throughout. Particular attention was paid to periodicals such as Construction Computing, Computing, and IEEE publications which often contained details of new products announced in related areas.

ii) Experimentation played an important role in the development of both the software tools (DB4GL), and the estimating and cost control system (Spec-Builder). The modular construction of DB4GL made it possible to insert and try new modules with ease. Similarly the prototyping facility of DB4GL made it possible to easily experiment

with various methods and facilities in the Spec-Builder system.

System development (DB4GL or Spec-Builder) was carried out in a series of stages. As each stage reached completion it was critically evaluated along with the rest of the system. The evaluation illustrated both the good and bad aspects of the design, and helped the generation of new ideas.

If the evaluation proved unsatisfactory, two corrective actions were available; either amend the existing system, or start again, incorporating the good features of the existing system with new ideas for the less satisfactory features. In the latter case, when developing DB4GL, parts of the new system could be generated using the discarded system.

iii) Formal interviews with personnel at the collaborating building company formed the main source of information for the estimating and cost control system. These interviews were often supported by the generous provision of documentation by the interviewees.

iv) Informal discussions with both research supervisors, members of the Building and Computer Studies departments, and other interested people provided an useful framework for the formation and development of ideas which could then be tried in the actual systems.

v) The attendance at lectures provided another important source of information. The lectures include courses attended within the Polytechnic run by the Building and Computer Services departments, and lectures organised by external organisations such as the British Computer Society.

vi) There are a number of products available which are similar in purpose to the systems developed during the research project. For example, other estimating systems and fourth generation languages. Much was learnt by attending demonstrations of such systems. Demonstrations were attended within Sheffield Polytechnic, at the vendors premises and at exhibitions such as 'Interbuild' and the 'Which Computer Show'.

The major task of the project was to construct two computer systems; DB4GL and Spec-Builder, where the former is used to generate the latter. This would seem to impose a strict sequence of generation software first, generated software second. However development of the two systems was interleaved to form 5 separate stages as described in 2.1 to 2.5 below.

2.1. DESIGN OF A PROTOTYPE SOFTWARE GENERATION SYSTEM.

The objective of the first software development stage was to formulate and test ideas generated during the initial investigation phases, and develop a set of software tools

that would form the basis of the eventual DB4GL system.

The initial system development was carried out using Microsoft Basic (MBASIC), an interpreted language which allows fast system development, so that ideas could be tried and evaluated quickly.

The result of this development was a simple screen and file generation system. Screen and file definitions were formalised, and could be entered into a 'data dictionary' interactively. The definitions could then be used to generate MBASIC programs.

It soon became apparent that the system was limited by the host language, which was slow to execute, and had poor file handling characteristics. The MBASIC system was therefore discarded and a new prototype developed using Micro Focus CIS COBOL. This language has good file handling characteristics, and executes faster- but being a compiled language, results in slower program development.

The revised prototype carried over the formalised screen and file descriptions within the data dictionary, from the previous system, and introduced file handlers.

The prototype was gradually refined, introducing more detail to the data dictionary definitions, more powerful file handlers, and complex screens, eventually forming a system sufficiently powerful to provide a useful software development tool.

This prototype was then used during the second software development stage.

2.2. PROTOTYPE ESTIMATING SYSTEM.

There were a number of reasons for starting the development of the estimating system before the software development tools were complete;

i) This stage involved the first contact with the collaborating building company. It was important to maintain their initial interest in the project through regular meetings.

ii) The systems analysis stage of the estimating system development required a large amount of user co-operation. Although this was always forth-coming, people within the collaborating company were busy with their own concerns, and care was required not to over exploit this co-operation. By dividing the systems analysis into two stages, their participation was spread over a larger time period.

iii) A systems analysis methodology was to form an integral part of the 'fourth generation' software development system. It was necessary to establish and test the basis of this methodology before completing the software tools to ensure compatibility.

iv) If the prototype software development tool was to form

the basis of the eventual system, then it was important to test that basis. The protoype estimating system provided such a test and illustrated the good and weak aspects of the software tools.

v) The early application of the software tools to a building problem verified the suitability of the concepts and tools being developed to test the hypothesis in sufficient time to allow revision if necessary.

The prototype estimating system, its method of development, and the software tools used are described in the unpublished research paper entitled :- 'A computer estimating system for a speculative house builder using 4th generation software tools' [Ewi01].

The analysis of the estimating prototype was encouraging. Software development was rapid, approximately 2 weeks including some improvements made to the generating software, and although the estimating system was only a prototype, the exercise illustrated that the software tools were applicable to at least some problems occuring in building companies. The research paper also received a positive response from the collaborating establishment, see appendix B4.

2.3. DESIGN AND IMPLEMENTATION OF DB4GL.

In the light of the generation of the estimating system, the prototype software tools were reviewed. Although generally successful, some significant improvements were necessary,

in particular a more powerful data base structure.

The prototype was discarded, and a new system built incorporating the features of the prototype with a data base system which linked the file handlers (renamed 'entity handlers') via subschemas.

The data dictionary system was retained and expanded to include more detail concerning the existing system components and subschemas. The term 'object' was introduced to describe the system components, and the data dictionary was renamed the 'object dictionary'.

Many of the new system components were generated using the discarded prototype, saving considerable time.

The system was further enhanced by the inclusion of a report generator, which used the flexibility of the data base system to select, sort, and format printed reports. The screen generation system was also enhanced to exploit the revised data base. The collection of tools was combined via a simple menu driven system, and named DB4GL, Data Base 4th Generation Language [Ewi02].

The systems analysis and design technique used during the construction of the prototype estimating system was refined and formalised into the method described in chapter 5 below.

2.4. THE DESIGN AND IMPLEMENTATION OF SPEC-BUILDER.

The integrated estimating and cost control system (Spec-

Builder) is a complex and extensive application. Although consisting of the same basic operations, there is a wide variation in the way in which estimating and cost control is carried out within different companies. This is illustrated by the wide range of estimating systems available, and the difficulty that many builders still experience in obtaining a suitable system.

The application therefore offers a good example with which to test the hypothesis, that bespoke application software can be produced in a more satisfactory way than currently achieved, through the use of advanced software development concepts and tools.

Spec-Builder was developed using DB4GL and its assóciated methodologies. This development is described in detail in chapters 6 and 7 below.

2.5. DB4GL REFINEMENT.

After the development of Spec-Builder, the DB4GL software tools were re-appraised. Previous appraisals had concentrated on short term considerations, which could be implemented in the subsequent generation of DB4GL. This final appraisal was also concerned with longer term development, identifying features that would be desirable in future versions of DB4GL.

The final development stage for DB4GL included short term amendments to the system, and the identification of some

possible future enhancements, such as multi-user data base access and system security mechanisms.

The short term amendments included the addition of process partners to screen and entity objects as described in chapters 3 and 4 below, and other cosmetic changes to the object dictionary format and system presentation to improve compatibility with the associated systems analysis and design methodologies.

The final DB4GL version is described in detail in chapters 3,4 and 5 below.

CHAPTER 3.

3. THE DB4GL DATA BASE MODEL

The Data Base model was developed gradually during the course of the research project. This chapter describes the final version of the data base model, and the considerations and theories behind its conception.

3.1. THE DERIVATION OF A SUITABLE DATA BASE MODEL.

The data base mechanism is the central component of the DB4GL prototyping system. Considerable effort was spent to develop a satisfactory design.

3.1.1 DATA BASE REQUIREMENTS.

The first stage in the design was to draw up a list of requirements for the data base mechanism.

i) It must provide a powerful and flexible tool for storing and manipulating data.

ii) It must provide a consistant interface between the data base and applications programs. This interface must be suitable for both generated and manually written applications programs.

iii) The data base system must be simple in concept, and

easy to use. Simplicity should also ease system design and implementation.

iv) It must be possible to generate the data base system from specifications held in an object dictionary (see chapters 4 and 5).

v) The data base mechanism must be implementable from the standard file handlng characteristics of the COBOL host language.

vi) The data base system must fit initially into a micro-computer environment.

When the requirements were specified, the design of the data base model was commenced. The first stage in this process was to examine existing data base models, to find a basis to work from.

3.1.2. THE ANSI-X3-SPARC DATA BASE MODEL.

As a starting point, the ANSI-X3-SPARC data base model was adopted [ANS01]. This model is recognised throughout the computer industry as a data base standard, and so is widely known and understood.

The ANSI-X3-SPARC model is based on three schemas [see fig 3.1], the conceptual, internal and external schemas. The conceptual schema describes only the entities and their properties, it does not make reference to the machine environment, or specify access paths between entities.

The external schema (subschema) describes the data items, formatting information, and datasets. The external schema can be mapped onto the conceptual schema, or (to increase performance at the expense of data independence) directly onto the internal schema.

The internal schema is responsible for the physical arrangement of data in the data base.

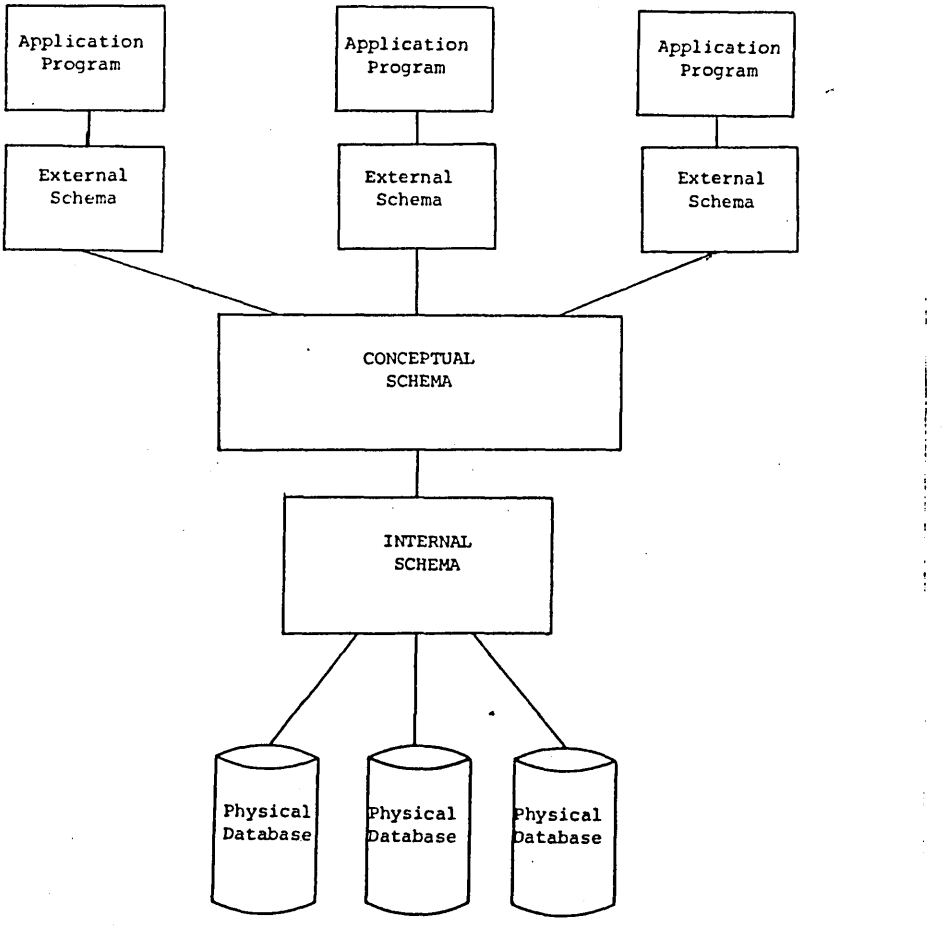


Fig 3.1. THE ANSI-X3-SPARC 3 LEVEL DATA BASE ARCHITECTURE.

3.1.3. THE ENTITY-RELATIONSHIP DATA BASE MODEL.

The ANSI-X3-SPARC specification describes a data base model,but does not detail how it should be implemented. Two

implementation methods were considered;

i) A 'CODASYL' type implementation [Inf01] [Dee01] is most commonly associated with the ANSI-X3-SPARC data base model. This uses independent schema and subschema data definition languages (DDLs) to specify the global and local views of the data base. These views include specifications of data access paths. The CODASYL method was rejected because it would have been too complex and memory dependent for a micro computer data base system, Network data base mechanisms also give poor data independence.

ii) The entity-relationship approach is based on a simple model, in which entities are expressed as normalised sets of attributes, and access paths are derived from relationships between entities. Both entities and relations can be expressed in a simple tabular form. The entity relationship model was first developed by CHEN [Chen01] [Chen02], with the objective of giving a pure and natural representation of the world, and to describe a conceptual schema in logical data base design. The entity-relationship model has potential for good data independence.

The entity-relationship approach was adopted for reasons of simplicity, and because it provides a very flexible and powerful method of data manipulation.

3.2. THE DB4GL THREE SCHEMA MODEL

Although derived from the ANSI-X3-SPARC model, the DB4GL data base system contains some differences. The three schemas types are retained, but their roles have been altered to produce a data base system founded on the entity relationship approach which gives a high level of data independence. The schemas are described in 3.2.1. to 3.2.3. below.

3.2.1. THE DB4GL CONCEPTUAL SCHEMA.

The conceptual schema is not part of the 'run time' data base system as external schemas map directly onto the internal schemas. However it is an important factor during the design, implementation and maintenance of the system.

The conceptual schema is part of the object dictionary (see chapter 4). It includes the descriptions of all entities and relations that are included, or are likely to be included, in the data base. These descriptions are used to generate the internal and external schemas (entity handlers and subschemas). The entity decriptions must be in 3rd normal form to enable the relationships to be fully exploited as mappings.

Each entity type is identified within the conceptual schema by a unique entity number. Each attribute type within an entity type is identified by an attribute number which is unique to that entity type. Any attribute type can

therefore be identified within the conceptual schema (or data base) by a pair of numbers; entity number, attribute number.

Within an entity type, there are two classes of attribute type; 'key attributes' which identify a particular entity occurrence, and 'non-key attributes' which merely describe the occurrence. For example, in an entity type 'stock item', 'stock code' would be a key attribute, but attributes such as 'stock item name', 'quantity on hand', and 're-order level' would be non-key attributes.

Key attributes are assigned attribute numbers 1 to 5, These numbers represent the significance of the keys. Attribute number 1 indicates the major, or most significant key, and attribute number 5, the least. Non-key attributes are numbered 6 to 99. Refering to the above example, 'stock code' would be attribute number 1, and 'stock item name', 'quantity on hand', and 're-order level' numbered say, 6, 7, and 8.

3.2.2. THE DB4GL INTERNAL SCHEMA (ENTITY HANDLERS).

The internal schema is implemented as a collection of 'entity handlers'. There is an entity handler for each entity in the data base, and they control and maintain the indexed sequential files in which the data is stored.

Data within the file, and the file itelf, is manipulated by calling the entity handler program and passing a parameter

which contains the instructions to be carried out. This formalised communication mechanism gives a very high level of data independence. Data manipulation instructions are divided into two classes.

INSTRUCTION	DESCRIPTION
OPEN	Open entity type (file), permitting input and output operations.
CLOSE	Close entity type (file).
GET	Get (read) the entity occurrence matching the value contained in the current entity key field. If no such occurrence exists return status.
GET NEXT	Get (read) the next entity occurence after the current entity, in key sequence.
LOCATE	Get (read) the entity occurrence matching the value contained in the current entity key field. If no such entity exists, get (read) the next entity in key sequence, and return status.
STORE	Store (write) the current entity.
DELETE	Delete the current entity.

Fig 3.2. ENTITY LEVEL OPERATION DESCRIPTIONS.

i) Entity level instructions are instructions that either change the state of the file, or the current entity (record) occurrence, for example read and write record, or open and close file.

The entity level instructions supported by the entity handler are based on the standard Micro Focus COBOL file actions, but include some enhancements.

The result of the execution of an instruction is recorded in a status field included in the return parameter. This can be interpreted by the calling program.

The full set of entity instructions, and their codes are described in Fig 3.2.

ii) Attribute level instructions manipulate the attribute values of the current entity. There are two such instructions; move-in, which inserts a new value into an attribute in the current entity, and move-out, which extracts the value of an attribute in the current entity.

a) To insert a new attribute value into the current entity (move-in), a data value is taken from the program call parameter, and inserted into an attribute in the current entity. The target attribute is identified by its attribute number, which is also passed in the parameter.

The entity handler can also validate data input. Only valid values are stored in the current entity.

The validation tests available are maximum and minimum value checking for numeric attributes, and upto 12 four character alternatives for non-numeric attributes.

If a validation test fails, an error status code is included in the return parameter.

Note, to update or insert the attribute value within the data base, the current entity must also be stored.

b) To extract an attribute value from the current entity (move-out), the value of an attribute in the current entity is entered into the program call parameter. This is then returned to the calling program. The source attribute is identified by the attribute number contained in the parameter.

Note, the current entity must be established by a prior entity action, for the value returned to be meaningful.

3.2.3. THE DB4GL EXTERNAL SCHEMA (SUBSCHEMAS).

The external schema, or subschema, defines 'logical' relationships between the individual entity types that comprise the data base.

The subschema program, which is generated from the conceptual schema, implements these 'relationships' as physical linkages, or mappings, between attribute types in different entity types.

All data base access must pass through a subschema program, which calls the relevant entity handler to carry out the initial instruction, and then brings about any necessary mappings.

A subschema may encompass the whole data base, or just part of it. There can be any number of subschemas and subschema programs.

A subschema can therefore implement a set of linkages (mappings) that represents the 'view' of the data base required by a particular application. Alternative relationships between entity types of the same data base (or part of), can be implemented as further subschema programs, as required by other applications.

Linkages are implemented as single direction mappings. Attribute types in different entity types are paired. One attribute type is called the source attribute, the other the target attribute. The entity types containing the attribute pair are similarly called the source and target entities.

The mapping takes the current value of the source attribute and puts it in the target attribute. The target attribute must be a key field, any attribute type can be a source attribute. The source attribute is also known as a 'foreign key'.

Once a mapping has been made, the subschema program calls the target entity handler instructing a 'locate' operation. The subschema program checks to see whether the value inserted into the target attribute is still present, and that any more significant keys are unchanged. If the 'located' entity occurrence fails this test, then the

current target entity is given a null value indicating that there is no corresponding target entity occurrence to the current source entity.

The subschema controls when the mappings are carried out. These occasions occur whenever the value of a source attribute changes; ie after a 'insert' attribute action on a source attribute, or a 'locate', 'get', or 'get next' entity action is carried out on a source entity.

The target attribute in one mapping, can also be the source attribute in another. Thus when the first mapping is made, the second mapping must also be carried out, as the value of its source attribute has changed. A single entity or attribute action can therefore cause a chain reaction throughout the data base.

The subschema program also contains a deadlock avoidance mechanism which inhibits 'loops' occurring in the chained mappings.

There is no limit as to how many times a single attribute is included as the source or target attribute in data base mappings.

Mappings are limited during program execution to those entity types which have been 'opened' by the applications program. This allows the formation of program 'domains' within subschemas, ie the applications program only uses the precise area of the data base that it requires.

3.2.4. ATTRIBUTE PROCESS PARTNERS.

In order to extend the processing capabilities of the entity handlers, 'process partners' have been introduced. Each data base attribute can have associated 'process partners'. Two types of process partners have been specified, and an attribute may have one of each type directly associated.

i) pre-process partner, to produce a data value prior to presentation to the user, for example 'age' from 'date-of-birth'.

ii) post-process partner, to act on the data value recieved and generate new values for that attribute, and/or other attributes, for example 'month-number' from 'month-name'.

Process partners are implemented in the entity handlers as in-line procedures called during attribute instructions. A pre-process partner is called during value extraction instructions, and post-process partners during value insertion.

CHAPTER 4.

4. THE OBJECT ORIENTATED APPROACH.

An ´object´ represents a unit of DB4GL software; a program. Each object consists of a unit of private memory (object space), and a set of operations. Objects communicate through ´messages´. A message is a request for an object to carry out one of its set of operations. The set of messages to which an object can respond is called the ´interface´ of that object. Messages can be sent by other objects or, if the receiving object has a ´human´ interface, from the user.

Objects can be grouped into object classes, these are groups of objects which represent the same kind of system component. An individual object within a class is called an instance. For example, in DB4GL, there is an object class ´entity type´ (see below) which includes all entities, ie the entity handlers. Each entity handler is an instance of the object class ´entity type´.

The only way to interact with an object is through its interface. This ensures modularity in the system, as messages merely send instructions, and are not concerned with how they are carried out. Changes to an object will therefore not affect other objects with which it

communicates unless the object interface is changed.

Smalltalk-80 [Go101] is a well known system which pioneered the object-oriented approach. It forms a very powerful graphical interactive programming environment which supports a large number of standard object types, to which the user can add his own.

Smalltalk-80 is a programming support environment, the concepts of objects and messages are carried down to the lowest levels of programming. DB4GL is a system development support environment and so objects referred to are 'high level' objects, that is they express system components rather than program components, there are consequently fewer object classes, but those objects are complex.

DB4GL programs are written in Micro Focus Level II COBOL, which is based on the 1974 ANSI standards, and is a traditional style programming language, which has been used to implement the object oriented system structure.

DB4GL supports only 6 object classes; entity type, subschema, screen, report, data manipulation, and system. Of these object types, the first four can be generated by DB4GL, system objects are common to all DB4GL systems, so the user only has to write the data manipulation objects (programs). This simple object oriented approach leads to highly modular systems.

4.1. THE OBJECT DICTIONARY.

Objects within the classes entity type, subschema, screen and report have a very similar format. The differences between objects in each class can easily be formalised, and used as the 'specification template' for all objects in that class. For example, an instance of the object class 'entity type' can be expressed entirely in terms of entity number and name, and details of the attribute types which make up that entity type. These details therefore form the specification template for the object class 'entity type', see appendix A2 [3-4].

The 'object dictionary' contains the specification templates for each of the DB4GL object types. Each template is described as one or more normalised entity type(s). These entity types are implemented as a standard DB4GL data base, ie as entity handler and subschema programs, and constitute the object dictionary.

The object dictionary is central to DB4GL. System descriptions, in terms of object specifications, are entered into the object dictionary data base during the systems analysis and design phase of the system life cycle.

Once in the dictionary, object specifications can then be used by a suite of DB4GL system programs to generate the data base, screen and report objects which form the prototype system. Amendments to the prototype are entered into the dictionary, and the appropriate objects

regenerated. Data manipulation objects are added to the prototype manually.

Object generation is achieved by inserting the object specifications into a 'skeleton program' which represents the common aspects of the object class. The 'insertion' is done by system object programs. There is one generation program, and one skeleton program per generated object class. The object dictionary and generation procedures are shown graphically in appendix A1 [2].

When a satisfactory prototype system has been built, documentation can be produced directly from the object specifications in the dictionary to support the prototype as the system specification. The prototype can be used as the basis for the live system, and much of the system documentation produced automatically from the dictionary.

Once the live system is implemented, the object dictionary can play an important role in system maintenance. All system changes can be entered into the object dictionary and tested on the prototype before altering the live system. Revised system documentation may then be produced automatically. Example documentaion generated by DB4GL is included in appendices C2 and C4.

4.2. ENTITY TYPE OBJECTS.

Entity type objects (entity handlers), are described in chapter 3. To summarise, an entity handler program is

dedicated to the maintenance and control of a single indexed sequential file. That file being the storage medium for an entity type in the data base.

The 'object space' of an entity handler is the record format of the file within the program work space. An entity occurrence contained in the object space is termed the 'current entity'.

The operation set of an entity handler is divided into two sections;

i) operations which insert or extract attribute values in the current entity.

ii) operations which affect the status of the file, or the current entity occurrence, for example open and close file, or locate and store entity occurrence.

Operations are enacted through the message set which forms the operation interface. The only object class which calls entity type objects directly are subschema objects. These contain a copy of the standard operation set for entity type objects.

Messages are returned to the calling subschema object after the completion of an operation, these messages contain data extracted from the current entity by the 'move-out' operations (see 3.2.2 (ii) above) and/or a status value representing the degree of success (or otherwise) of the

operation.

The specification template for entity types is expressed in two object dictionary entity types;

i) entity details (entity number 1 in the object dictionary) which contains information concerning the entity type, such as entity name, description and disk drive.

ii) attribute details (entity number 2) which describes the attribute types that make up the entity type. There is one such entity occurrence per attribute.

These object dictionary entity types are fully described in appendix A2 [3-4].

Entity type specifications are entered and maintained in the object dictionary interactively, using the screen program described in appendix A3 [17].

4.3. SUBSCHEMA OBJECTS.

Subschema objects (programs) describe an area, or view, of the data base. This includes logical relationships between entities, which are implemented as single direction attribute mappings. Subschema programs are described in chapter 3.

All data base access must pass through a subschema program, and not directly to the entity handlers. Operations within the subschema operation set have two functions;

i) To reformat and pass on messages to relevant entity handler objects.

ii) To re-establish any relationship links that may have been affected by that entity handler operation. This may require sending and receiving a number of messages to and from a number of entity handlers.

The message sequence of a subschema object can be summarised as;

i) Receive message from calling object (eg screen program).

ii) Send message to entity handler specified in previous message.

iii) Receive message from entity handler

iv) Depending on received message, operation type, and the entity relationships involved, send and receive messages to and from affected entity handlers until all data base linkages are again consistant.

v) Return message to original calling object indicating the success of the requested operation.

Three entity types are required to store the subschema specification template;

i) Subschema heading details (entity number 5). There is

one such entity occurrence per subschema.

ii) Subschema area (entity number 6) specifies each entity type included in the subschema. There is one such entity occurrence per entity type in the subschema.

iii) Attribute relationships within the subschema area (entity number 7) are expressed in terms of source and target attributes. There is one such entity occurrence per relationship (attribute mapping).

The subschema specifications are described in detail in appendix A2 [8-10]. Specifications are entered into the object dictionary using the screen programs described in appendix A3 [18-19].

4.4. SCREEN OBJECTS.

Screen objects (programs) allow the user to access and update data contained in the system data base interactively.

To develop a general purpose screen object class, the requirements of screen programs within the DB4GL environment were first analysed;

i) A single screen program should allow the user to operate in create, update, enquire, and delete modes, thus reducing system duplication.

ii) It must be possible to generate the screen program from specifications, held in a data dictionary, as required by any prototype system.

iii) The screen program should be able to maintain entity types with ´1 to many´ or ´parent child´ relationships.

iv) The screen program must interface with the established data base system. The program must be able to identify and act on error messages returned from the data base mechanism.

v) The general screen design must reflect and encourage the accepted guidelines for good screen design [Gal01].

A screen program displays the image of a blank ´form´ on the VDU. The user is then invited to fill in the form. The screen program provides a set of simple commands (messages) with which the user can control the screen operation set.

A form, or screen image, is built up of two types of fields;

i) Constant fields which describe the ´blank form´. The values of these fields are constant irrespective of which, if any, entity occurrence is current, and cannot be changed during execution.

ii) Variable fields which contain data to be entered into, or has been extracted from, the data base. Each variable field relates directly to an attribute type. There are two types of variable fields; ´input fields´ into which data can be inserted and amended during program execution, and ´output fields´ which cannot, but whose value depends on the current entity.

Data is entered into the varable input 'fields' in sequence from left to right, top to bottom on the screen, which follows the natural eye movement sequence [Gal01].

The screen program prompts the user for a response by displaying the brackets '<' and '>' around the current input field. Data can only be entered between the two brackets. When the user has entered data into a field, the 'return' key is pressed, and control passes back to the screen program.

Each value input is checked against the validation criteria for the corresponding attribute type. These criteria fall into two classes;

i) Format, the field is checked to see whether it lies between the minimum and maximum lengths, and if a numeric field, does not exceed the maximum number of integer or decimal places. Non-numeric characters cannot be entered into numeric fields. Fields with a mandatory input condition are checked for a value.

ii) Content, the value of the field is checked against any validation criteria specified for the attribute type. This is acheived by interpreting messages received from the entity handler objects via the subschema.

If the data input fails a validation test, then the field is rejected, an appropriate error message displayed, and the

user asked to re-input the field.

There are two screen types within screen object class; simple screens and complex screens.

4.4.1. SIMPLE SCREEN OBJECTS.

A 'simple' screen is concerned with a single entity, that is, only one entity type can be updated from that screen. Data from other entity types can be displayed on the screen if there are the necessary mappings from the 'source', or 'projected' entity type within the specified subschema object.

When the screen program is run, a blank 'form' (the screen format), is displayed on the VDU. The user is prompted to enter data at the first input field (the field nearest the top left hand corner).

The first field input will relate to the most significant key attribute type of the source entity type. If there are more key attributes, then these fields are also requested. Once the complete key has been entered, the screen program searches the data base for an occurrence within the source entity type which matches the key value(s) input.

If no such occurrence is located, then the screen program assumes that the user wishes to create a new entity, and requests values for the remaining input fields.

If there is an existing entity occurrence which matches the

key entered, then the remaining screen fields are filled in with values from the located entity.

Once the screen is complete, either after the user has entered a new entity occurrence, or an existing occurrence has been displayed, the user is presented with three options (the simple screen object interface).

i) DELETION; The user can delete the current entity occurrence from the data base. After deletion, the screen program clears all variable fields, and returns to the first input field.

ii) EXIT; The user can exit from the current entity occurrence, saving the updated version in the data base. All variable fields are cleared, and the screen program returns to the first input (key) field.

iii) AMEND; The user can amend any of the non key input fields displayed on the screen. The fields are amended in the same sequence that they are input, left to right, top to bottom.

At each field, the current value is enclosed by the brackets '<' and '>'. The user can amend the value enclosed by over-writing it. The 'return' key moves the screen program to the next input field.

When the last field has been amended, the user is again presented with the same three choices, EXIT, DELETE, or AMEND.

The screen program is terminated by entering a null value at the first input field. The process flow of simple screens is illustrated in appendix A4 [30]. An example simple screen can be seen in appendix C4 [24].

4.4.2. COMPLEX SCREEN OBJECTS.

A 'complex' screen is the combination of two entity types within a single screen. The two entity types must be connected via 'one to many', 'parent child' or 'owner member' relationships. Such relationships are a basic construct of the entity relationship model. A single occurrence of the parent entity type is projected in the top screen area, and many occurrences of the child entity type can be projected in the bottom area.

A 'parent child' relationship exists when the key attribute type(s) of the parent entity type form the major (most significant) key attributes in the child entity type. The child entity will also have further key attribute(s), ie in a child entity type the first (major) key attributes identify the 'parent' entity occurrence to which a child belongs, the remaining key attributes identify a child entity occurrence amongst those with the same parent.

The subschema object (program) used by the screen program must contain mappings from the key attribute types in the parent entity type to those related attribute types in the child entity.

The top, or parent, screen area, is treated in much the same way as a 'simple' screen. In the bottom, or child, screen area, each related 'child' entity occurrence is presented as a series of input fields on a single line. Although several such lines can be displayed at once, if there are more related child entity occurrences than available lines, the user can 'scroll' this area of the screen up and down as required.

When a complex screen program is run, the blank 'form' is displayed and the user prompted to input the first field in the parent area of the screen (a null field stops program execution). The first fields input relate to the key attributes of the parent entity type. Once these have been input, the screen program searches the parent entity type for an occurrence matching the key entered.

If no match is found, then the screen program assumes that the user wishes to enter a new entity occurrence. The remaining input fields in the 'simple' part of the screen can be entered. The user can then enter any number of child entity occurrences, one line at a time.

If there is a corresponding parent entity occurrence to the key entered, then the located entity details are displayed on the screen. The screen program also searches for related occurrences of the child entity type. Details of these, or at least as many as there is room for, are displayed in the

bottom part of the screen. Additional child entity occurrences may be displayed by 'scrolling' the screen.

The complex screen object operation set is divided into two subsets, one each for the simple and paged (scrolled) areas of the screen. Once the screen has been filled, the user is presented with the operation subset for the paged screen area.

i) EDIT; the user can insert new child entities or amend and delete existing ones. When edit has been selected, then the user is prompted to enter the key fields of a child entity occurrence. If it is a new occurrence, then the user is prompted for the remaining field details, if the key matches an existing occurrence, then those details are displayed, and the user is given a further choice of amending or deleting the occurrence, or leaving it unchanged.

Once the current child entity has been either entered or re-displayed and updated, the screen program prompts for another key field. This continues until a null key field is entered. The screen program returns to the original option list.

ii) SCROLL DOWN; If there are too many related child entity occurrences to be displayed at once, then the remainder may be displayed by 'scrolling' down. The bottom line currently displayed in the paged screen area is moved to the top, and subsequent lines are displayed below. The

child entity occurrences are displayed in ascending key order. Groups of lines displayed together are called 'pages'.

iii) SCROLL UP; After scrolling down, the user can return to previous 'pages' by scrolling up, the reverse of scrolling down, ie the current top line in the paged screen area becomes the bottom line, and lines representing child entity occurrences with lower key values fill the top part.

iv) EXIT; This operation changes the operation subset from the paged screen area to the simple screen area.

The simple screen area operation subset allows the user to manipulate the parent entity details in the top part of the screen. This operation subset is very similar to the operation set of simple screens;

i) AMEND; The simple screen area can be amended in exactly the same way as if it were a wholly simple screen. However, once amended, the user is passed again into the operation subset for the complex screen area.

ii) DELETE; The parent entity occurrence is deleted, and also the related child entity occurrences. The screen is cleared of variable fields, and a new key requested.

iii) EXIT; The parent entity occurrence is saved, or updated, and the variable fields cleared. The user is

prompted for data at the first input (key) field.

To exit the screen program, the user must enter a null value at the first input field. The process flow of complex screens is illustrated in appendix A4 [29]. An example complex screen can be seen in appendix C4 [22].

4.4.3. SCREEN PROCESS PARTNERS.

Process partners can be introduced to increase the flexibility of screen object programs. It is possible to assign both a pre-process partner and a post-process partner to each variable field in the screen.

i) Pre-process partners are enacted before a value is displayed or input at a field. These process partners are useful for displaying special information, or further instructions concerning a particular data field.

ii) Post-process partners are enacted after a value has been displayed or input at a field. Post-process partners can act on the value which was displayed or input at the field position. They may be used for further data manipulation and to control field input sequence.

4.4.4. SCREEN OBJECT SPECIFICATION.

Three entity types are used to describe occurrences of the screen object class in the object dictionary.

i) Screen header (entity number 10), contains the specifications of general information concerning the

screen, for example, the subschema and entity type objects required, and whether the screen object is a simple or complex type. There is one screen header entity occurrence for each screen object.

ii) Constant field description (entity number 11), describes the screen co-ordinates and value of constant fields. There is an entity occurrence for each constant field in the screen format.

iii) Variable field description (entity number 12), describes the screen co-ordinates and source or target entity and attribute reference (in terms of entity number and attribute number). There is an entity occurrence for each variable field in the screen.

These entity types are fully described in appendix A2 [5-7]. Specifications are entered into the object dictionary via the screen programs described in appendix A3 [21-23].

4.5. REPORT OBJECTS.

The production of hard copy reports is an essential function of any data base system. The requirements for report objects within, and generated by, DB4GL are as follows;

i) The report programs should be able to select information from the data base which corresponds to parameters inserted at run time.

ii) It should be possible to sort the selected data into

any required order before it is printed.

iii) The report format should be flexible enough to cope with a wide range of report formats.

iv) The report program must interface with, and complement, the existing data base system.

v) It must be possible to generate report programs from specifications held in the object dictionary.

To help the construction of the report generator, which is a very complex program, a formal specification was drawn up describing the features that should be included in an ideal report object class, and how they can be implemented. This specification is included in appendix D and forms the basis of the DB4GL report generator.

Although data from more than one entity type can be included in the same report, It is necessary to identify one entity type as the 'main source entity'. This entity type contains the basic data required in the report, and from which all the other entity types required by the report can be mapped.

The production of a report takes place in two stages; the selection and sorting of the data, and the formating and printing of the report.

4.5.1. DATA SELECTION AND SORTING.

Before any data can be printed, it must be selected from the data base and sorted into order.

i) Selection :- Entity occurrences are selected from the main source entity, and their key values are stored in a temporary list file. The selection is achieved by comparing attribute values in the main source entity, and other 'linked' entity types, with values specified by the user at run time. There is no limit to the number of selection conditions included in the report progam. To be selected an entity occurrence must meet all of the specified selection criteria.

The comparison tests provided are; equal, not equal, less than, not less than, greater than and not greater than.

ii) Sorting :- Once the data hes been selected, the temporary key list is sorted into a specified order. Any number of sort keys can be included in the report program, the keys can be any of the attribute types in the main source entity, or other 'linked' entity types, The key list can be sorted into either ascending, or descending order.

If the keys are not sorted, then the report is printed in the ascending sequence of the main source entity key values.

4.5.2. REPORT FORMAT AND PRINTING.

The report format is divided into three sections or areas;

i) Report Header :- The report header area is printed at the top of every page in the report. The area may simply include the report title, however it may also include variable data fields, and titles describing the following data area.

ii) Data Area :- The data area contains repeating groups of data lines. Each group of lines describes a single entity occurrence in the main source entity.

The groups of lines are repeated until a 'break condition' occurs. These conditions occur when the end of the main source entity is reached, in which case execution halts, when the end of the page is reached, or when a specified 'break attribute' changes its value.

iii) Footing area :- The footing area is printed at the bottom of each page of the report, for example after a break condition has occurred during the printing of the data area. Constant and variable data can be included in the footing area.

4.5.3. RUN TIME CONTROL.

When a report program is run, a simple screen format is displayed, and the user is prompted to enter values for any of the run time selection parameters. The data base is

searched for occurrences of the main source entity which match the selection parameters entered. The corresponding keys are stored in a temporary file. The sort keys are also stored in the temporary file. This file is then 'bubble' sorted according to those keys.

The sorted main source entity keys are then used to sequence access to the main source entity and so consequently all those other linked entities in the subschema. Data can then be extracted, in the required sequence, from the data base for printing in the report.

4.5.4. REPORT OBJECT SPECIFICATION.

The object dictionary representation of the specification template consists of four entity types;

i) Report header (entity number 15), contains general report details, such as report name, and which subschema and entity handlers are used by the report program.

ii) Report format entity type (entity number 16) contains descriptions of the fields (both constant text and variable data) that form the report. There is an entity occurrence for each report field.

iii) The report sort entity type (entity number 17) allows the specification of sort keys. For each report sort key there is a corresponding entity occurrence.

iv) Selection details are specified in the report select

entity type (entity number 18) in a similar way to sort details. Each selection criteria is represented by an entity occurrence.

These entity descriptions are presented in more detail in appendix A2 [11-14]. Specifications are entered into the system via the screens described in appendix A3 [25-28].

4.6. DATA MANIPULATION OBJECTS.

Data manipulation objects take data from one or more entity type(s) and use it, with perhaps some transformation, to update or create one or more other entity types.

Unlike the entity, subschema, screen and report object classes, it is very difficult to formulate specification templates for data manipulation objects as there is a wide variance in their function. There is therefore no data dictionary representation for these objects, nor generation routines.

The DB4GL user must write these programs manually using COBOL, but can interface with the DB4GL data base by using the standard message format and operation codes. The data base interface was described in chapter 3.

4.7. SYSTEM OBJECTS.

A system object is a DB4GL program which is supplied in compiled form only. They are not generated by the user, and are common to all DB4GL systems. There are two kinds of

system object;

i) System objects which constitute the DB4GL system, such as the specification maintenance, object generation, system menu, and documentation programs. The MICRO FOCUS COBOL compiler and run time programs can also be included as system objects.

ii) System objects which can be used by generated programs. For example all screen field input and output is acheived by sending messages to a system object called 'ACCEPT'. This object carries out all the field operations, such as updating and extracting information from the relevant attribute value in the data base, accepting a value from the user and checking its format, displaying values from the data base, and returning the value to the calling screen object.

CHAPTER 5.

5. APPLICATION DEVELOPMENT USING DB4GL.

The DB4GL software development system should be used in a much wider role than purely as a program generator. The associated systems development methodologies utilise the software tools to enable DB4GL to play an important role throughout the system life cycle.

The system development methodologies require adoption of new approaches to system development and maintenance.

Systems development, particularly systems analysis and design, is an area of much debate in the light of new techniques and tools. Some people (Ed Yourdin et al) support structured analysis methodologies such as those described by De Marco [DeM01], and Gane and Sarson [Gan01]. Others, notably James Martin [Mar01], support the prototyping approach to systems analysis and design [Com01].

The proposed systems analysis and design methodology is a combination of these two approaches, the structured systems analysis methods, and the use of protoyping. It is recognises that both approaches have strengths and weaknesses;

i) Structured analysis provides a logical framework for the recording of analysis results, and the development of well structured solutions. However, although pictorial, the design is not easily demonstrable to users, and is poor at expressing the business meaning of the data and functions it describes.

ii) Prototyping gives a very clear picture of the proposed system, this can be used to obtain valuable feedback from the users for system verification and improvement. However, prototyping does not provide a stuctured format for initial system analysis, and often ignores the system elements that fall outside the scope of the prototyping system.

Zimmerman [Zim01], advocates a balanced combination of system development methods to give a broader insight into the problem, and in areas of overlap, provide checks. However the combination of more than one methodology to a single situation implies duplication of analysis effort.

In the development methodology described below, duplication of effort has been avoided by applying each methodology only to those aspects of analysis to which it is most suited. Initial analysis and design is carried out using structured analysis techniques, subsequent design refinement and verification is achieved by system prototyping.

The effective combination of techniques is acheived through the DB4GL object dictionary. Specifications derived from the

initial analysis and design phase are entered directly into the dictionary, and used to generate the system prototype.

Although based on a micro computer, DB4GL can be applied to the development of large systems. It can be used as a tool by systems analysts and programmer analysts to develop systems to run on large multi-user computers. With some development, DB4GL could become fully compatible with these computers thus allowing the implementation of prototypes as 'live' multi-user systems.

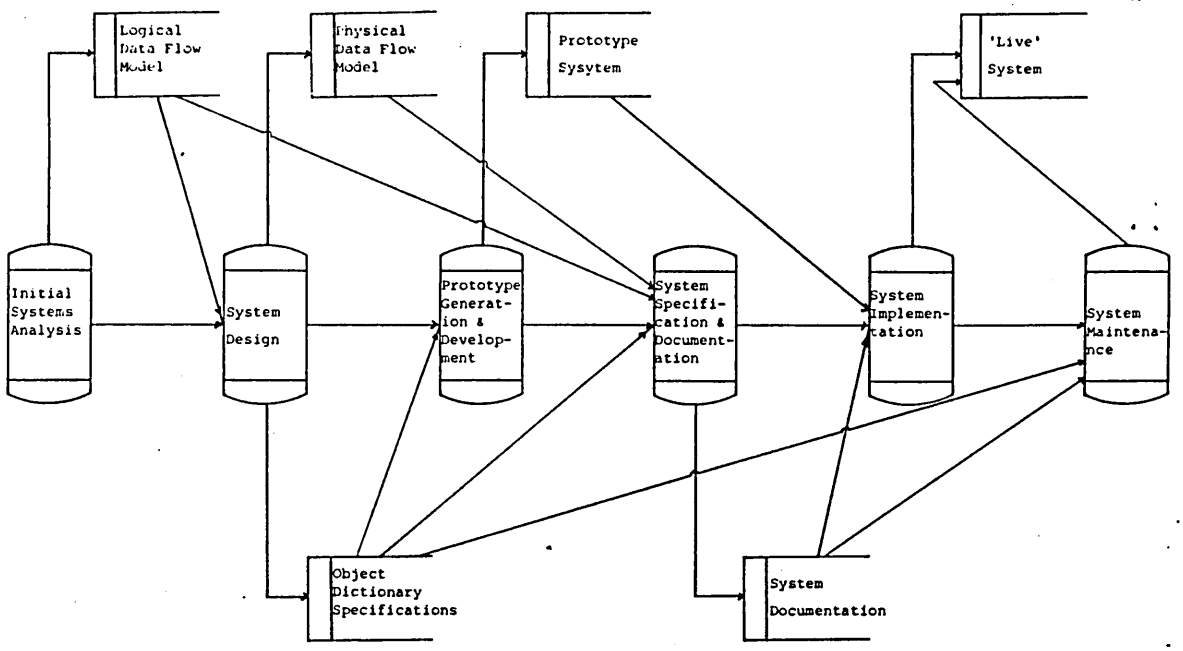


Fig 5.1. DB4GL SYSTEM DEVELOPMENT METHODOLOGY.

System maintenance is simplified as amendments can be introduced by re-generating the affected objects. The highly modular nature of the applications systems developed using DB4GL means that amendments are isolated to those objects

directly affected by changes.

The complete system development methodology is summarised in Fig 5.1.

5.1. INITIAL SYSTEMS ANALYSIS.

The initial systems analysis is the first stage of the system development process. Traditional systems investigation methods are used, such as interviews, observation, and documentation searches, to build a global model of the system.

Initially, the analyst concentrates on data requirements and data flows within the system, rather than functional specifications. The aim is to produce a simple 'logical data flow diagram' [Gan01] [DeM01]. This diagram should be constructed using the four symbols (representing broad function types) illustrated in Fig 5.2.

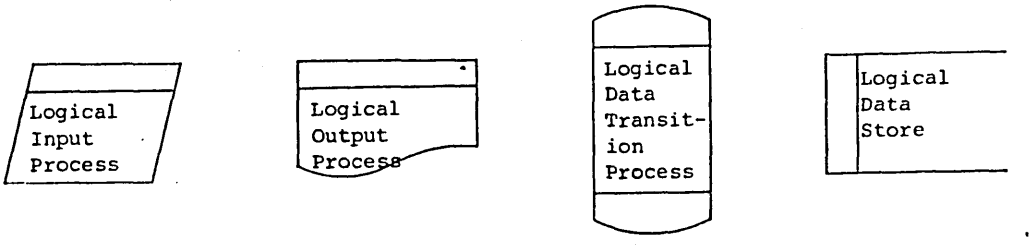


Fig 5.2. LOGICAL DATA FLOW SYMBOLS.

The symbols are joined using lines indicating the logical flow of data through the system. Each symbol should be

identified by a number unique within its type. Details of the items represented by each symbol in the logical data flow model are recorded on standard forms (see appendix B3).

The first draught logical data model should aim to identify and organise the major system components. When the first model has been completed, it forms a working document, and is successively amended and refined to include greater detail. The model is complete when the analyst is content that it is correct, and that suffiecient detail is included to allow a system prototype to be designed.

The 'top down' approach is an important aspect of structured systems analysis [Gan01] [DeM01]. It gives the analyst a logical, controlled approach to his analysis task.

5.2. SYSTEM DESIGN.

The system design phase uses the logical data flow model and its supporting documentation to develop a prototype design. The objectives are;

i) To draw up a physical data flow model, which identifies the entity, screen, report and data manipulation objects that will constitute the applications system.

ii) To enter the relevant DB4GL object specifications into the object dictionary.

The physical model is derived from the logical data flow model. It is built using the same symbol set, but with a

well defined meaning to each symbol type, see Fig 5.3. Each symbol type corresponds directly to a system object type. The objects contained in the physical data flow model can therefore be described in the DB4GL object dictionary.

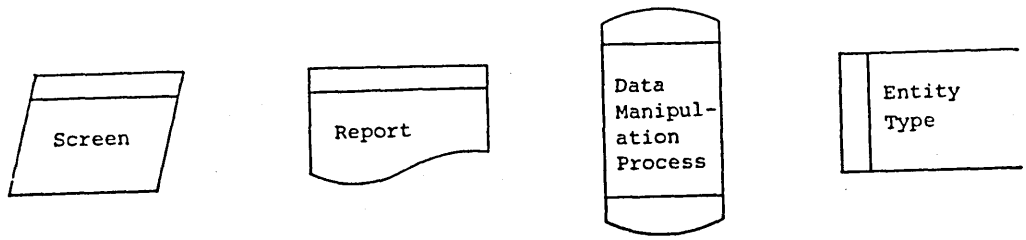


Fig 5.3. PHYSICAL DATA FLOW SYMBOLS.

The design phase is divided into 5 stages, the first three are concerned with developing the DB4GL generated system objects, the fourth with data transformation (manipulation) objects. The final stage is a 'walk through' check of the system design. These stages are described in 5.2.1. to 5.2.5. below.

5.2.1. DATA BASE DESIGN.

The data base forms the basis for all the other stages, it is therefore necessary to complete its design before starting the other design stages.

The first step for the designer is to remove data redundancy and normalise the data store descriptions contained in the logical data flow diagram. Normalisation simplifies the data

structures into simple tabular entities [Gan01] [DeM01] [Dee01] [Sak01].

One probable result of normalisation is that each logical data store will be split into more than one physical entity type object. This will be reflected on the physical data flow diagram, where each entity type object is represented as an separate symbol. For example, a data store 'order' after normalisation will be split into two entity type objects;

i) 'order header' contains details such as 'order date' and 'supplier'. There is one occurrence of 'order header' per order dispatched.

ii) 'order line', describes the items orders. There is one occurrence of 'order line' per product ordered, there may therefore be many occurrences of 'order line' relating to a single occurrence of 'order header'.

Each entity type is given a unique entity number, and recorded in the DB4GL object dictionary along with the attributes that constitute it. Decisions must be made concerning field type and size, but these can be amended at a later stage.

5.2.2. INPUT DESIGN.

All data input to the system is be made via screen programs. Each logical input process represents one or more screen program(s). The screens maintain entity types derived from

the logical data store(s) to which the logical input process is connected in the logical data flow diagram (ie the target entities).

The screen requirements for any logical input process depends upon;

i) The number of entity types required to implement the target logical data store(s). The maximum number of entity types a single screen can maintain is two, so at least one screen is required for every two target entity types.

ii) The relationships between the target entity types. If a 'parent-child' relationship exists between two entity types, then they can be maintained by a single complex screen. If no such relationship exists, then two simple screens are required.

iii) The number of input fields required to maintain each entity type. If the entity types consist of a large number of attributes, then more than one screen may be required to maintain them. Such entity types cannot be maintained by complex screens.

iv) The format of incoming data. Screen design should reflect the document(s) from which the information is taken in terms of input order and presentation [Gal01].

v) The timing and source of input data. Some entity types, or related entity types, contain data attributes which

have different sources, and so may not become available at the same time. Screen design should combine only those attribute details normally available together.

Each screen format is given a unique screen number, recorded in the object dictionary, and shown as a symbol on the physical data flow diagram. The subschema requirements of the screen are calculated, and if no suitable subschema exists, they are recorded in the object dictionary.

5.2.3. OUTPUT DESIGN.

There are two possible output formats, enquiry screens and reports. The first design task is therefore to establish which of these formats is required. Enquiry screens are designed and specified in the same way as input screens (5.2.2. above).

To design a report, it is necessary not only to establish what data is required, and the source entity types, but also the order and format in which it is to be presented. Each report is given a unique report number, and the specifications are entered into the object dictionary. Subschema requirements must also be calculated, and if no suitable subschema exists, they are recorded in the object dictionary.

5.2.4. DATA MANIPULATION PROCESS DESIGN.

The specifications of data manipulation processes are not stored in the object dictionary, and are less easy to define

than for the other DB4GL object types. Specifications are formalised to a limited extent by showing which entity types are accessed for input and output, and subschema requirements which are entered into the object dictionary if no suitable subschema exists.

The actual process procedures can be specified in the traditional narrative form, or by using structured techniques such as those described by Gane and Sarson [Gan01].

5.2.5. SYSTEM WALKTHROUGH.

The physical data flow model provides a clear picture of the complete system. This picture is used in the system walkthrough. The objective of the walkthrough is to illustrate;

i) That all system output is possible, ie all output data is derivable (by specified processes) from data entered into the system.

ii) That all system input is necessary, ie for each value input, there is a path to a corresponding output value (or values).

iii) That the system is complete, ie that all functions have the necessary source and target destinations.

Any errors detected should be amended in both the physical data flow model, and the object dictionary before proceeding

to the next stage.

5.3. PROTOTYPE GENERATION AND DEVELOPMENT.

The purpose of the prototype system in this system development methodology is fourfold;

i) To provide the program basis for the 'live' system or, at least, the basis of the system specification.

ii) As a systems analysis tool to extract system details missed by the initial analysis stage and less quantitative details such as user opinion.

iii) To test and prove the system design as a worthwhile solution to the problem.

iv) As a 'public relations' tool to encourage user participation in design, and acceptance of the eventual system.

Prototype generation is the cyclic application of three processes, which are repeated in sequence to refine and develop the prototype. This evolutionary approach is complete when a satisfactory prototype has been developed. The processes are described in 5.3.1. to 5.3.3. below.

5.3.1. PROTOTYPE GENERATION.

Prototype generation is acheived by applying a suite of generation programs (contained in the DB4GL system menu), to specifications held in the object dictionary. This

application produces source COBOL programs which can be compiled to form the prototype system.

The first system prototype should be generated immediately from the specifications entered into the object dictionary during the design phase. Data manipulation objects should only be included as simple approximations in order to build up a data base for output objects, for example, reports, that use derived data. This data base could alternatively be derived by generating temporary screen objects.

In some cases, a prototype developed in this way will be sufficient to satisfy the prototype requirements. In more complex systems however, some manual additions are required, for example data manipulation objects and process partners. These can be added to the prototype as it passes through its evolutionary development cycles.

5.3.2. PROTOTYPE ANALYSIS.

Once a prototype has been generated, it should be tested using sample data from the user department. The subsequent output data should be checked and any errors in the prototype corrected.

The prototype should then be taken into the user environment, and demonstrated to the people who will use and supervise the system. Careful note should be made of all user response, as further details concerning the system, which were missed during previous analysis phases, may

become apparent.

5.3.3. PROTOTYPE AMENDMENT.

The results of the prototype analysis phase ie user responses, are analysed, and a list of amendments produced. These amendments are made in both the physical data flow diagram, and the object dictionary. If the initial systems analysis was accurate, then these amendments should be limited to cosmetic changes to the user interface. The prototype development cycle continues with the re-generation of the prototype.

When the prototype is found to be satisfactory ie the prototype is complete (including data manipulation objects and process partners) and there are no amendments outstanding, the prototype generation phase is complete.

5.4. SYSTEM SPECIFICATION AND DOCUMENTATION.

During the analysis, design and prototyping phases, a comprehensive system specification is built up. This comprises of;

i) The logical data flow model and supporting documentation from the initial analysis phase.

ii) The physical data flow model from the design phase.

iii) Detailed object specifications from the object dictionary as entered and amended during the design and

prototype generation phases.

iv) A working model (the prototype) demonstarting the 'live' system from the users point of view.

If the prototype is to be used as the 'live' system, then the specification becomes the system documentation.

5.5. SYSTEM IMPLEMENTATION.

The prototype is a single user system, based on a micro computer. If the 'live system' is to be installed in such an environment, then the prototype can be adopted after testing.

Since much of the prototype is generated using standard DB4GL objects, which have been well tested, program testing should concentrate on the non-standard parts, such as the data manipulation objects. Thorough system testing should be carried out to check the interfaces between the system objects.

If the live system is to be installed in a more complex environment, for example on a mainframe or mini computer, or on a computer network, then the prototype will need enhancing to include such facilities as file sharing and security.

Even in complex mainframe environments the prototype may still be used as the program basis, as DB4GL generates COBOL programs which conform to the 1974 ANSI standard [Mic01].

The prototype is therefore compatible, at source code level, with most mini and mainframe COBOL compilers.

This compatibility can be enhanced considerably by two additional developments;

i) The implementation of file sharing and system security facilities in the DB4GL prototyping system.

ii) The introduction of new micro to mainframe program development aids such as 'VS COBOL WORKBENCH' recently announced by Micro Focus [B1a01]. This package gives object code compatibility between IBM mainframes and the IBM Personal Computer during the development of COBOL programs which comply with the ANSI 74 or IBM 85 standards.

5.6. SYSTEM MAINTENANCE.

System maintenance is an ever increasing and expensive burden on data processing. It has been estimated that 50% of the total costs incurred during the system life cycle are spent on the maintenance phase [G1a01].

Systems developed using DB4GL are easier to maintain than traditionally developed systems for five reasons;

i) Systems developed using DB4GL are highly modular and so most amendments can be localised to the relevant system modules.

ii) The data base mechanism offers a very high level of

data independence, so amendments to the data base need only affect those objects which use the changed data items.

iii) The user can quickly test maintenance amendments by updating and testing the prototype before interrupting the live system.

iv) If the live system has been derived directly from the prototype, then the appropriate modules can be re-generated with little manual intervention.

v) System amendments are recorded in the object dictionary in order to re-generate the live system and/or system prototype. Documentation can be updated automatically from the object dictionary.

5.7. OBSERVATIONS ON DB4GL.

An analysis of the DB4GL software development system brings forth the following observations.

i) The DB4GL system provides an integrated strategy for software development through the phases of systems analysis, design, implementation and maintenance.

ii) The DB4GL generation tools allow the rapid development and substantiation of system designs, producing prototype data base, screen and report designs quickly, which can be used to involve the users in the system design process.

iii) The use of prototypes allows a tangible demonstration of the proposed system to the user before the implementation phase begins. Many problems can be located and sorted out immediately without expensive re-programming.

iv) By using the system prototype as a basis, the 'live' system can be developed faster than more traditional programming methods alone.

v) Maintenance is eased by the modular system design, and the high level of data independence provided by the data base mechanism. Attributes can be added, amended, and deleted from entity types, and only those programs that use those attribute types need be amended or re-generated.

vi) DB4GL is self documenting, the specification of each system object is held in the 'object dictionary', and can be retrieved, formated and printed using the programs provided. System specifications and documentation can easily be produced and maintained.

vii) It was recognised that the program generation tools could not fulfill all the requirements of a complex system development strategy. Therefore flexibility was increased by the introduction of 'process partners' which allow users to include enhancements in the generated programs. The COBOL source code can be edited to incorporate further amendments.

viii) Programs which are beyond the scope of the generation tools are still included in the development strategy. They are clearly defined in the design process, and interface with the generated data base system. Such programs must be written in COBOL.

CHAPTER 6.

6. ANALYSIS OF THE ESTIMATING AND COST CONTROL FUNCTIONS.

There are a large number of packaged computer estimating systems available, some, such as the estimating system from Mandata, allow the integration of a cost control module. Most of these packages have been designed for the general contractor, and it would appear that very few have been designed specifically for speculative house builders.

Estimating systems designed for general contractors are not usually applicable to speculative house builders for two reasons;

i) Estimating systems for general contractors are usually based on the standard method of measurement (SMM), as that is the format of most bills of quantities presented to them. Speculative house builders produce their own bills of quantities (for internal use only), and so use a format most akin to their particular needs.

ii) The purpose of the estimate for the general contractor is to allow managers to produce a competitive tender for a particular contract [Ins01].

Speculative house builders do not produce tenders, but use

cost estimates to aid decision making and control. In particular;

a) To estimate the cost of developing a new site, which, when compared to the estimated site revenue, indicates the site potential.

b) To estimate the cost of building a new house type, which can be compared with the estimated selling price and other similar designs to show whether the design is cost effective.

c) To produce periodic site budgets, which form the basis for cash flow forecasts to maintain the planned level of site construction.

d) To provide information to enable accurate cost control during the site construction.

6.1. METHOD OF ANALYSIS.

The estimating and cost control package was developed from analysis carried out at a local medium size speculative house building company. This company already used computers for accounting and cost analysis, and, at the time of analysis, a bespoke estimating package was being written using traditional software development methods.

The involvement of this company was entirely altruistic. Whilst exceptionally generous with time and the supply of information, it was obviously impossible to expect the same

level of involvement that would be obtained from a prospective user department.

In order to spread the analysis phase over a wider timescale and reduce the concentration of user involvement, the system development was divided into two phases;

i) In the first phase emphasis was placed on the estimating function, an estimating system was designed and a prototype developed, using an early version of DB4GL [Ewi01].

ii) The first phase estimating system and supporting documentation formed the basis for the second analysis phase which investigated the cost control function and it's relationship to the estimating function.

During the initial analysis phases, four fact gathering methods were used;

i) A literature search into estimating and cost control in the building industry. The search provided background information only and was not specific to speculative house builders.

ii) Interviews with the personnel from the speculative house builder proved the most useful source of information. These interviews were formally arranged, and conducted using prepared lists of questions and subject headings. These structured interviews were very successful at extracting information, and only seven

interviews were required throughout the project.

a) Interview 1, with the Managing Director, to gather introductory information concerning the company, for example the company history and scale of operation, and to seek access to other members of the organisation.

b) Interview 2, with the company secretary and accountant, to discover how the accounting and cost control functions were carried out, and establish the current computer resources and how they were used.

c) Interviews 3, 4, and 5 with the chief estimator, to gather specific information concerning the estimating function, and to demonstrate initial system designs.

d) Interviews 6, and 7 with the accountant, to gather specific information concerning cost control and its relationship to estimating, and to demonstrate the initial system design.

iii) The investigation of documentation supplied during interviews to support the interview notes.

iv) The investigation of existing computer estimating packages, through attending demonstrations and exhibitions, and the analysis of supporting documentation.

When the investigation stage was complete, a logical data flow model was drawn (appendix B1 [2]) and supporting documentation produced. A report of this system was written

[Ewi01], and sent to the builder who confirmed its accuracy (appendix B4). The second analysis stage used that model as a basis, extending it to include the integration of the cost control function (appendix B1 [3]). The analysis of each logical object was recorded on standard forms (appendix B3).

6.2. THE ESTIMATING FUNCTION.

Speculative house builders have a limited number of standard house designs. There is a bill of quantities for each of these house designs which is maintained as the design is updated.

The structure of the house type bill of quantities forms a hierarchy of 'operations' and 'items';

i) An 'item' is a unit of resource which has a value, for example, a unit of material or operatives time, or a sub-contracted operation. Items are grouped according to cost type. There are seven such types;

a) Materials.

b) Sub-Contract.

c) Labour.

d) Plant and Haulage.

e) On Costs.

f) Selling.

g) Supervision.

ii) 'Operations' are the combination of resources to produce output. Operations are combined into a three level

hierarchy, each operation can be expressed in terms of items and lower level operations.

The highest operation level relates to the construction of a particular house type. This operation can be broken down into 'secondary operations' which represent the major tasks involved in the construction of that house type, for example, foundations, and brickwork to first floor. Secondary operations can be broken down into 'primary' operations and items. A primary operation is a combination of items.

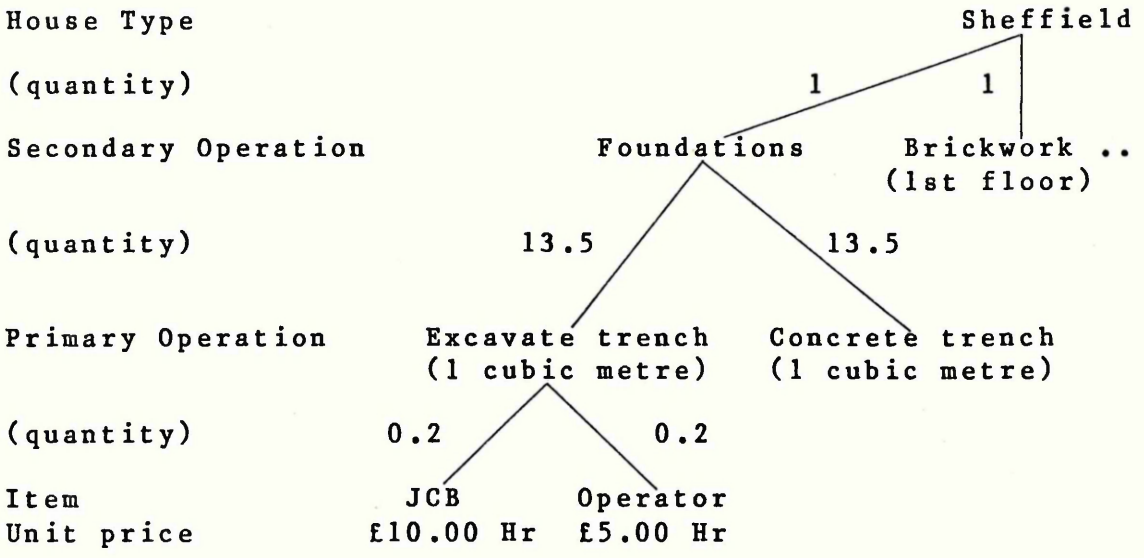


Fig 6.1. The Bill of Quantities Hierarchy

The bill of quantity structure can be easily described using an example (see Fig 6.1.); The construction of an imaginary house type, the "Sheffield", can be divided into a number of secondary operations, one of which, relates to the

construction of the house foundations. Note that the bill of quantities for each house type includes a 'standard' foundation, estimates of plot variations from this standard are included as general site costs (see below).

The house foundation secondary operation can be divided into a number of primary operations which includes the excavation of the foundation trench, and concreting the foundations.

The primary operation to excavate a trench corresponds to one cubic metre of earth removed, however the house foundation requires the excavation of 13.5 cubic metres. The secondary operation therefore associates a quantity of 13.5 with the primary operation to excavate trench.

The primary operation requires the resources, or 'items' of a JCB excavator, and its operator. These are priced at £10.00 and £5.00 per hour respectively, but to excavate 1 cubic metre of earth only takes 12 minutes, or one fifth of an hour. The primary operation therefore associates the quantity of 0.2 with each of these items.

To price the bill, the hierarchical tree structure is traversed, The value of each operation is calculated as the sum of the product of its constituent lower level operations and items multiplied by their associated quantities.

In the example, the excavation primary operation is priced at £3.00 ((£10.00 x 0.2)+(£5.00 x 0.20)), and the excavation of the foundation trench costs £40.50 (£3.00 * 13.5). This

is added to the cost of the other primary operations forming the foundations secondary operation to give the total foundations cost.

If the JCB belongs to the 'plant and haulage' item type, and operator to the 'labour' item type, then an item cost type analysis of the house foundation excavation shows £13.50 labour, and £27.00 plant and haulage.

To produce a site estimate, a bill of quantities is drawn up. This bill is a combination of two bill types;

i) Standard house type bills of quantities. Each plot is represented within the site estimate by the bill of quantities which corresponds to the house type to be constructed on that plot.

ii) A general site bill of quantities which describes the site construction and site overheads required to allow the house construction to proceed. This includes roads, sewers, site security site supervision, and some special plot costs such as the extra cost of non standard foundations.

A site bill of quantities is produced for these costs in the same way as for a house type, ie using a hierarchy of secondary and primary operations, and items. Site costs are apportioned to plots using an apportionment factor, such as plot frontage.

To allow the integration of the estimating and cost control functions, it is necessary that estimated and actual costs can be compared over the same format. This is achieved by including the same three digit code structures used for cost recording within operation descriptions. When an estimate is produced, it is structured using these cost codes rather than the bill of quantities codes

Three types of estimate are required, as described in 6.2.1 to 6.2.3 below.

6.2.1. HOUSE TYPE ESTIMATE.

When a house type bill of quantities has been drawn up for a new or amended house type, an estimate can be produced to show the cost of building that particular house type.

The estimate is summarised as a table with item cost along one axis type and operation cost code along the other. Totals and subtotals are also included.

6.2.2. NEW SITE APPRAISAL.

Before the purchase of a new site, a new site appraisal is produced. The purpose of this analysis is to help the land purchasing department, and ultimately the company management, decide whether or not to attempt the purchase, and the maximum price that they should be prepared to pay.

To prepare a site appraisal the estimator is given a plan of the proposed site, which includes the plot layout,

indicating the house type to be built on each plot.

The estimator produces a site bill of quantities, combining the separate house type bills for each plot with a bill of general site costs, for example roads, sewers, and security.

The combined site bill of quantities is priced, and the general site costs are apportioned to each plot according to plot frontage. The resulting estimate is summarised to produce a site appraisal (see appendix B2 [4] for manual format). Estimates of house selling prices are included in the appraisal for comparison, any surplus indicating the builders gross profit and the value of the land.

6.2.3. SITE BUDGET ANALYSIS.

The frequency with which site budgets are produced varies between companies, some produce them annually, or more frequently, others produce budgets to cover phases of construction (see appendix B2 [5] for manual budget format).

The purposes of a site budget are;

i) To provide expenditure estimates to be used during the formulation of cash flow forecasts.

ii) To provide a estimated costs for comparison with actual costs over a given construction phase or period.

Site budgets are produced in the same way as site appraisals. Both are site estimates combining house type

and general site cost bills of quantity. There are however three significant differences between site budgets and appraisals;

i) Site appraisals are produced before a site is purchased, budgets are only produced after purchase.

ii) Site budgets do not include any income data, ie estimated house sale value, as included in site appraisals.

iii) Site appraisals include the whole site, whereas, unless the site is small, only the portion to be constructed during the budgeted period or phase is included in the site budget.

6.3. COST CONTROL FUNCTION.

Cost control systems have two important purposes;

i) To pin point unusually high costs, and directing management effort towards solving the associated problems. Similarly practices leading to low costs can also be identified.

ii) To compare cost control reports with corresponding estimates. Estimate inaccuracies are high-lighted and the estimating data updated so that future estimates will be more accurate.

To implement a cost control system, it is necessary to

record all construction costs as they are incurred, and allocate those costs to the elements of construction for which they are required. This is achieved using a simple code structure.

The code is numeric, and divided into three parts; the first three digits form the site code, or site number, the second three digits form the plot code, or plot number, and the final three digits form the cost code catagory (the same code as recorded in the operations within the bill of quantities). General site costs, that is those costs not directly attributable to a particular plot, are recorded using a six digit cost code, ie omitting the plot code.

General site costs are apportioned to each plot, in the site cost analysis.

The cost information is extracted from three sources;

i) Operatives worksheets, which show the time spent on each cost coded section.

ii) Invoices from material suppliers, sub-contractors, plant suppliers etc. This information is combined with information from the site surveyors and foremen in order to assign cost codes to the invoiced items.

iii) Goods received notes to which cost code and provisional cost information is added on site. The cost coded notes are passed to the cost control, or accounting department, and matched with corresponding invoices, see

(ii) above. If the matching invoices are not found, then the provisional costs can be included in the cost analysis reports, and erased when the invoices are received.

Site surveyors record the amount of work done on site using 'tick sheets'. These sheets break down the construction of a house into a number of operations, and each operation is broken down into a number of equal parts. Each part is represented by a box. The surveyors assess the proportion of each operation complete, and tick the corresponding number of boxes. For example, if there are eight boxes representing the excavating of the house foundation trench, and the surveyor considers this operation half complete, then he will tick four boxes.

The tick sheets can be used to indicate operations which are likely to exceed their budget before they are complete, allowing early management response. With respect to the previous example, if the tick sheet shows the foundation trench to be half complete, yet the comparison between cost analysis and estimate shows that three quarters of the budgeted cost has already been spent, then the combination of this information pin points a problem.

6.4. THE LOGICAL DATA FLOW MODEL.

The information extracted during the two analysis phases was used to develop a logical data flow model which represents the estimating and cost control functions within a

speculative house building company (see appendix B1 [3]).

The individual symbols that comprise the model are explained in more detail on the supporting documentation, see appendix B3.

CHAPTER 7.

7. DESIGN OF THE ESTIMATING AND COST CONTROL SYSTEM.

From the basis of the logical data flow model and supporting documentation produced during the systems analysis phase, a system design was derived.

Before system design could be considered, code structures required formulating;

i) The cost, site and plot code structures were all derived from the manual system (three digit numbers).

ii) House type codes were formed as 5 character codes of the form XXXnn, where XXX represents the first three characters of the house type name, for example SHE for the house type SHEffield, and nn represents the variation number within that house type.

iii) Bill of quantity operation codes were also formulated. Each operation was identified using a two part code; operation level and operation code. Operation level is a number 1 or 2, indicating whether the operation is primary or secondary level, and operation code is of the form XXXnnn, where XXX represents the first three characters of the operation class, for example BRI for

BRIckwork, and nnn is a sequence number to identify the operation within its class.

iv) Item codes were also structured XXXnnn, where XXX represents the first three characters of the item name for example CAR for CARpenter, and nnn is a sequence number to identify the individual item within its item group.

7.1 PROTOTYPE DESIGN.

The initial prototype design for Spec-Builder was developed following the 5 stage design methodology described in chapter 5 above.

i) The data base requirements were specified within the data stores described in the logical data flow model and supporting documentation. To design the data base, the data store descriptions were normalised [Gan01] [DeM01] [Dee01] [Sak01] to form simple tabular data structures called entity types. The resulting entity types were examined for inconsistancy and unnecessary data redundancy which was removed.

For example logical data store 1 (see appendix B3 [6]) describes house types in terms of house details (eg description, and house class), and the major operations required to build it.

When logical data store 1 was normalised, it formed two entity types; entity 1 containing the descriptive house type details, and entity 2 containing the major operations

required to build it, see appendix C2 [3-4]. There was only one occurrence of the entity 1 per house type, but there was an occurrence of entity 2 for each major operation required to build each house type.

The normalised entity types were numbered and recorded in the object dictionary. An analysis of these entity types, generated from the data dictionary is included in appendix C2.

ii) The arrangement of input screens was derived from both the input processes of the logical data flow model, and the newly designed data base entity types.

There are two types of DB4GL generated screens; simple screens which update a single entity type, and complex screens which can update two entity types which have a 'parent-child' relationship (see chapter 4 above). The normalisation of the logical data base split many of the logical data stores into more than one entity type. The logical input processes which update those data stores may require spliting into more than one input screen.

Screen design was carried out to following the guidelines described in chapter 5 above. The four logical input processes produced seven screens. These screen descriptions were entered into the object dictionary from which the screen documentation included in appendix C4 was generated.

The subschema requirements for each screen were also noted, and recorded in the object dictionary as additions to an existing subschema, or as a new subschema.

iii) The four output processes contained in the logical data flow model related directly to the estimate and cost control reports identified as being required during the systems analysis phase (see chapter 6).

It was noted that the format of the site budget and site appraisal were very similar. Implementation was simplified by combining the two output processes to form a single report object. The logical distiction between site appraisal and site budget remained (see 6.2.3 above).

The format and content of each report was recorded in the object dictionary.

The subschema requirements for each report were also noted, and recorded in the object dictionary as additons to an existing subschema, or as a new subschema.

iv) In the initial prototype, the data manipulation processes were ommitted. Temporary screens were designed and recorded in the object dictionary, which enabled the output from the data manipulation process to be simulated for system testing purposes. These temporary screens were gradually replaced by data manipulation programs during the prototype development phase.

v) As the prototype passed through the design stages, a physical data flow model was built up (see appendix C1). Using this model it was possible to follow the path of data through the system. Inconsistancies, for example output values that could not be derived from the data input, were identified and removed.

7.2. PROTOTYPE DEVELOPMENT.

The initial prototype design was recorded in the object dictionary as it was developed. Once complete, the prototype was generated.

The objective of the initial prototype was to demonstrate and test the input and output procedures to ensure that they were in a suitable format, and expressed, or requested, the right data attributes.

In order to test the generated processes, example data was entered into the prototype screens, and into temporary screens to put data into the source entity types of the prototype reports. Apart from minor alterations, to include totals in reports for example, the input and output processes were satisfactory.

Most of the prototype development was concerned with the data manipulation processes. The first process developed was to calculate house type estimates. The resulting program accepted a house code from a screen, and traversed the corresponding bill of quantities tree to calculate the

estimate. This process was tested using source data input through the house type and operation input screens and by printing the results using the house type estimate report.

The data transition process in the logical data flow model to calculate site estimates, was divided into three seperate stages;

i) To calculate the estimate of general site costs from the general site cost bill of quantities.

ii) To copy plot estimates from the corresponding house type esimate generated by data manipulation process 1.

iii) To apportion the general site costs to each plot, according to plot frontage.

Each stage was implemented as a separate data manipulation processes. There were four reasons for the process division.

i) To simplify the programs required to calculate a site estimate.

ii) To give the user greater flexibility during the calculation of site estimates. The calculation of site estimates from a large bill of quantities can be time consuming, and so by splitting the process, the user could select partial re-calculation, and so save time.

iii) An intermediate entity type was required to hold the site estimate before it was apportioned. This entity type could be used as a source for an unapportioned site

estimate report, ie an estimate that shows details of general site costs.

The new entity type was entered into the object dictionary, and an entity handler generated.

iv) Micro Focus level II COBOL programs can use a maximum of six index sequential files simultaneously. To combine the site estimate as a single program would require more than this. Even with the division it was necessary to combine the operation header and item entity types to reduce the number of files required to calculate the estimate of general site costs.

The unapportioned site estimate report was implemented, and used to analyse the results of the two estimate calculation processes when applied to test data. Test output from the apportionment process was output using the site budget report.

The logical data transition process to calculate site costs was divided into two stages, or processes, the calculation of site costs, and the apportionment of general site costs by plot. This division also required a intermediate entity type, which contained details of general site costs. A report which selected and compared the actual and estimated direct site costs could be easily introduced. The apportionment process was copied from the equivalent site estimate program.

Test results of the site cost calculation process were listed via the cost v estimate report process.

When the data manipulation processes had been implemented in the prototype, which then formed a complete system, the prototype design was appraised, and amendments which were necessary, or desirable noted.

Ideally the prototype should have been demonstrated to the collaborating building company at this stage, however this was not possible due to the logistic problems of moving the computer system, and time shortages. The system design was however demonstrated to the accountant during the final interview, albeit in a diagramatic form.

From the results of the prototype appraisal, two amendments were made to the prototype system.

i) The inclusion of operation estimates. The estimator is frequently faced with 'rush jobs', in which estimates have to be produced very quickly. Breaking down the bill of quantity into a hierarchy of operations and items is a time consuming process.

The facility introduced allows the estimator to use his experience to price operations rather than breaking them down to item level. This can save time at the expense of accuracy.

ii) Site cost variations. The collaborating building

companies area of operation covers an area from Yorkshire to Northamptonshire. There can be considerable variances in costs at the different sites throughout this area. These variances can be incorporated into the site estimate using factors introduced into the site description entity type. There is a factor for each item cost type, and when a site estimate is calculated, all item costs are multiplied by the corresponding factors. It is therefore possible to increase (or decrease) all labour charges on a particular site, by say 10%, by entering the relevant factor in the site description.

Allowances for inflation can be incorporated in site estimates in the same way.

On completion of these amendments the prototype would, in normal circumstances, be ready for implementation.

7.3. SYSTEM DEVELOPMENT DURATION.

The initial analysis phase of the system development process was carried out in two stages, as described in chapter 6. It is difficult to quantify the time taken to complete the analysis as it was carried out concurrently with development work on the DB4GL system. An estimate drawn from the research diary indicates that around two weeks was spent on each analysis phase.

The development of the Spec-Builder prototype system from the analysis results took just over five weeks. This was

spent as follows :-

i) System design - One week.

ii) The generation and testing of data base, screen and report programs - Three days.

iii) The development of data manipulation processes, ie the estimate calculation and apportionment programs - Two weeks.

iv) Prototype testing and development to form a system which would be suitable for implementation in a 'live' environment - One week and three days.

This time estimate does not include time spent on literature searches, attending lectures, or product demonstrations that were part of the investigation process.

It is not possible to make a direct comparison between the development time taken using DB4GL and traditional system development procedures, without developing identical systems using both techniques. However the development time was obviously extremely short for a complex system.

CHAPTER 8.

8. USING SPEC-BUILDER.

Spec-Builder forms a comprehensive estimating and cost control system. In terms of functional capability it exceeds the bespoke systems that were written for the collaborating company. However it does not contain the refinements such as multi-user capability, and security considerations that would be necessary for a commercial system.

A two level hierarchical menu system is used to combine the components of Spec-Builder into a packaged system. Menus are lists of options displayed on the screen. For each option there is a description, and a three character code. To select an option, the code is entered in the position indicated;

ENTER OPTION, OR 'X' TO EXIT < >

This causes the corresponding program to be executed. When the program execution is completed, control returns to the menu program, and the menu is redisplayed.

The menu at the top of the hierarchy is the main system menu, which is displayed when the system is started (see appendix C3 [18]).

This menu divides the package into three sections; data base maintenance screens, system processes, and data base reports. There is a menu option for each of these sections which lead to sub-menus forming the second level of the menu hierarchy.

8.1. DATA BASE MAINTENANCE.

When the data base maintenance option is selected in the system menu, a sub-menu is displayed. The sub-menu contains an option for each screen program in the system, see appendix C3 [21].

8.1.1. HOUSE DETAIL MAINTENANCE.

The house detail maintenance screen is a complex screen which is used to maintain the two house detail entity types, (see appendix C4 [22-23]).

The parent entity details describe the general house details, the child entity details describe the operations carried out to build the house type.

8.1.2. OPERATION DETAIL MAINTENANCE.

Operation details are maintained by a complex screen. The parent entity details describe the operation, and the child entity section contains the details of the sub-operations and items which make up that operation, see appendix C4 [26-27].

When entering details in the parent section, the cost

'CATEGORY' displayed after a value has been entered into the 'COST CODE' field.

The value entered against 'ESTIMATE (Y/N)' is used by the estimate calculation processes to determine whether the value of the operation is to be calculated from it's sub-operations and items, or the value entered in the 'ESTIMATED VALUE' field is to be used. When (and if) the operation cost is calculated, the process stores the value in the entity type. This value is displayed in the 'ESTIMATED VALUE' field.

The 'WASTAGE%' value is a percentage value entered to allow for estimated wastage of each sub-operation or item during the operation construction. For example if an operation requires 10 cubic metres of concrete, and the estimator expects a 5% wastage, then he enters a 5 in the wastage field. When the estimate is calculated, the cost of the sub-operation or item is increased by 5%, thus the estimate is calculated for 10.5 cubic metres.

8.1.3. ITEM DETAIL MAINTENANCE.

Item details are maintained through a simple screen, see appendix C4 [24-25]. The field 'COST TYPE' must be one of the following;

i) 'LAB' - Labour item

ii) 'MAT' - Material item

iii) 'SUB' - Sub-contracted item

iv) 'PLA' - Plant or haulage item

v) 'ONC' - Oncosts

vi) 'SEL' - Selling

vii) 'SUP' - Supervision

Some items, such as sand, are always calculated as site costs, even though they may be included in the house type bills of quantities. These items can be denoted by entering the value 'Y' against the field 'APPORTIONED'. This value is tested by the house estimate calculation process, and if positive, the item is ommitted from the estimate.

8.1.4. SITE HEADER MAINTENANCE.

Site heading details are entered using a simple screen see appendix C4 [28-29]. The site address and telephone number are for documentation purposes only. Site weightings allow local cost anomalies to be incorporated in the estimate without adjusting the operation data base. If for example the cost of labour on a particular site was 10% higher than usual, the by entering the value 1.100 against 'LABOUR', all labour item costs are increase by 10% when the site cost is estimated (the value 1.000 represents no weighting).

8.1.5. SITE OPERATION MAINTENANCE.

The site operation details are maintained through a complex screen. The parent entity is the site header entity type, but only the site code is entered in this part. The site name is displayed if the corresponding site header details

have been entered. The child entity details (the site operations) are entered in the same way as operations in the house detail maintenance screen above, see appendix C4 [32-33].

8.1.6. PLOT MAINTENANCE.

The plot maintenance screen is similar to the site operation maintenance screen (above) in format, see appendix C4 [30-31]. It is a complex screen but the only field entered in the parent entity section of the screen is the site code. The site name is displayed. There is a child entity occurrence for each plot within the site, and the details entered against the plot are the code for the house type to be constructed, and its estimated sale value.

8.1.7. COST DETAIL MAINTENANCE.

The cost detail maintenance screen was designed to reflect the format of invoices and goods received notes which are the source documents, see appendix C4 [34-35]. The 'SOURCE REF' field is the identification code of the invoice, goods received note, or labour worksheet. This code should also be recorded on the source document for future reference. The 'PROVISION OR INVOICE' field will accept the values 'P' or 'I' only. This field indicates whether the following cost items are actual costs or provisional costs. Provisional costs can easily be updated to actual costs on receipt of an invoice, by replacing the 'P' with an 'I', and recording the source reference on the invoice.

Cost items are recorded in the child entity part of the screen in the same sequence as they appear on the source document. Each item is allocated to a site, and if applicable, a plot. A cost code is also assigned. A brief description of the item is recorded, and the item cost catagory, ie MATerial LABour etc. Finally the gross value of the item is entered.

8.1.8. COST CATAGORY MAINTENANCE.

Cost codes and cost code descriptions can be maintained via a simple screen (see appendix C4 [36-37]). Meaningful names can be assigned to the numeric cost codes, which can then be used in their place to make reports more easily understood. The cost code description is used by the operation maintenance screen above, as a visual reference to enable the operator to check that the correct cost code was entered.

8.2. SYSTEM PROCESSES.

The system process menu is a sub-menu of the main system menu, see appendix C3 [20]. It contains an option for each of the data manipulation processes which facilitate the calculation of estimated and actual costs.

A complete site estimate is built up from the sequential application of four estimate processes.

i) The calculation of house type estimates.

ii) The calculation of general site costs.

iii) The calculation of plot estimates, using the house type estimates.

iv) The apportionment of general site costs to each plot.

Site cost analyses are produced through the application of two processes.

i) The calculation of total site costs.

ii) The apportionment of general site costs to each plot.

All of the data manipulation processes consist of the same three execution stages.

i) A simple screen is displayed, and the user requested to enter the code representing the object (site or house type) to be calculated or estimated. If an invalid code is entered, then an error message is displayed, and the user invited to re-enter the code. When a valid code is entered, the object name is displayed, and the process continued. If a null code is entered, then the process is stopped, and control returns to the menu.

When a data manipulation process is completed, the screen data fields are cleared, and the user is prompted for another code. To exit the process, a null field is entered.

ii) When a valid code has been entered, then any data generated by a previous application of the process to that

code is deleted.

iii) The data manipulation process updates the target entity type using information drawn from source entity types.

The data manipulation processes are described in detail below.

8.2.1. HOUSE TYPE ESTIMATE.

The house type estimate process is used to create, or update, information held in the house type estimate entity type.

The estimate is calculated by traversing the bill of quantities hierarchy, multiplying operation and item quantities by item values and estimated operation values. Wastage factors are also applied.

8.2.2. SITE ESTIMATE :- DIRECT COST.

The direct cost site estimate process calculates the estimated cost of the general site costs, that is those costs not directly attributable to a plot.

Direct site costs are estimated in the same way as house types, the bill of quantities is traversed, multiplying item and operation quantities by item values and estimated operation values. Wastage factors are also applied. The resulting values are then multiplied by the relevant site variation factors, and stored in the unapportioned site

estimate entity type.

8.2.3. SITE ESTIMATE :- PLOT COST.

The site plot cost estimate follows the same three stages as the previous processes, ie accept valid site code, delete existing estimate details, calculate new estimate.

To calculate plot estimates, the house type estimate, as generated by the house type estimate process, is used. Site variation factors are applied to the plot estimates as they are calculated. The results are stored in the unapportioned site estimate entity type.

8.2.4. SITE ESTIMATE APPORTIONMENT.

To complete the site estimate, the analysis should be expressed in terms of cost per plot. This allows direct comparison between site income, from house sales, and site costs. Therefore a share of general site costs for example roads and sewers, must be apportioned to each plot. To achieve a fair distribution, plot frontage is used to calculate an apportionment factor.

The apportionment process totals the plot frontages for the site, and calculates the apportionment factor for each plot as; INDIVIDUAL PLOT FRONTAGE / TOTAL PLOT FRONTAGE. The resulting factors are recorded in the plot detail entity type.

Each general site cost item is 'shared' between the plots,

under the collective cost code 'APPORTIONED', as the product of apportionment factor and total general site cost.

8.2.5. SITE COST CALCULATION.

The site cost calculation process calculates the total actual and provisional costs entered into the system, through the screen described in 8.1.7. above, for a given site. The resulting data is stored in the unapportioned site cost entity type.

8.2.6. SITE COST APPORTIONMENT.

The site cost apportionment process functions in the same way as the estimate apportionment process (see 8.2.4. above), except that it also apportions provisional cost information.

8.3. DATA BASE REPORTS.

The report menu is a sub-menu of the main system menu. It contains an option for each of the reports that can be generated from the data base.

8.3.1. HOUSE TYPE ESTIMATE.

The house type estimate report program produces a tabular analysis of the estimated costs, see appendix C5 [38]. The report program prompts the user for a house code, which it uses to extract the relevant details from the house type estimate entity type. These are then sorted into cost code

order, and printed. Obviously, a house type estimate must have been calculated previously (see 8.2.1 above).

8.3.2. SITE ESTIMATE :- DIRECT COSTS.

The direct cost site estimate report includes an analysis of the unapportioned general site costs. It also includes an estimate analysis for each plot, see appendix C5 [39-41]. The report program prompts the user for a site code, and then selects, sorts and prints the corresponding details from the unapportioned site estimate entity type. The source data must have been previously created by the direct site cost, and plot cost estimate processes (see 8.2.2. and 8.2.3. above).

8.3.3. SITE ESTIMATE :- BUDGET.

The site budget is an analysis of estimated site costs, broken down by plot (see appendix C5 [42-43]). All general site costs are apportioned between the plots. The report program prompts the user for a site code, and then selects, sorts and prints the corresponding details from the apportioned site estimate entity type. The source data must have been previously created by the direct site cost, plot cost estimate, and estimate approtionment processes (see 8.2.2, 8.2.3, and 8.2.4 above).

8.3.4. COST ANALYSIS :- BY SITE.

The cost analysis report is a list of all cost items entered into the cost control system. It is ordered (in order of

significance) according to site code, plot code, and cost code. It also includes the reference identifying the source document, and a description of the cost item (see appendix C5 [46]).

The purpose of the cost analysis report is to aid the detection of abnormal costs. Such costs are highlighted in the cost v estimate comparison report, but only in terms of plot number, cost code, and item type. Using that information, and the cost analysis report, the cost item(s) can be traced back to their source document. The report program prompts the user for a site code, and then selects, sorts, and prints data from the cost source and cost item detail entity types.

8.3.5. COST V ESTIMATE COMPARISON.

The site cost v estimate comparison report is an analysis of actual and provisional site costs and includes site estimate details for comparison (see appendix C5 [44-45]). The report program prompts the user for a site code, and then selects, sorts and prints the corresponding details from the apportioned site cost entity type. The source data must have been previously created by the site cost calculation and apportionment processes (see 8.2.5, and 8.2.6 above).

CHAPTER 9.

9. CONCLUSIONS.

Many building companies, although keen to exploit computer technology, have difficulty in obtaining suitable software. There are two ways in which companies without internal data processing expertise can acquire suitable software;

i) The purchase of 'packaged' software. Such software is designed to be used within a large number of organisations, thus exploiting economies of scale. However, the specialist requirements of individual companies are not catered for.

ii) The commission of bespoke software. This software is traditionally expensive, and can be unreliable. There is no guarantee that the resulting software will fully satisfy the users' requirements.

The research hypothesis stated that through the use of advanced software development concepts and tools, reliable bespoke software can be written for building companies quickly and cheaply.

When the project was started, there were no micro-computer based software development systems available with sufficient

flexibility to test the hypothesis. It was therefore necessary to specify, design, and implement such a system. This strategy offered an opportunity to develop and test novel approaches to data base structures.

The resulting system was named DB4GL (Data Base 4th Generation Language). The DB4GL system combined current systems analysis and design techniques with program generation tools, a powerful data base mechanism and a self documentation facility. This combination formed a structured approach to software development throughout the system life cycle, from the initial phase of systems analysis to system maintenance. Thereafter DB4GL was used to develop an integrated estimating and cost control system, which would be suitable for small to medium sized speculative house building companies. The resulting applications system was called 'Spec-Builder'.

At the start of the research project there were already a large number estimating packages available [Con01]. However many companies with specialist requirements still could not find suitable packages, this included the collaborating company who had already commissioned a bespoke estimating system that was developed during the course of this project.

The estimating and cost control system was therefore seen as a difficult application area where the wide variety of approaches adopted by individual housing companies made it virtually impossible for software package suppliers to

satisfy individual company requirements.

9.1. THE DB4GL SOFTWARE DEVELOPMENT SYSTEM.

The DB4GL system was an ambitious attempt to produce a comprehensive software development strategy, encompassing the whole application system life cycle, that is; systems analysis, design, prototyping, implementation and maintenance.

The core of the DB4GL system is an object dictionary into which system design specifications are entered, and from which system prototypes, and eventually 'live' systems are generated. Hence the DB4GL development methodology forms a guideline for the analyst, or analyst-programmer to follow. At specified points certain documents, such as data flow diagrams, must be produced, and certain tasks, such as entering information into the object dictionary, must be carried out.

Prototyping tools must allow rapid program development, and provide the flexibility to amend prototypes easily. Using DB4GL experience confirms that screen, report and data base programs can be generated very quickly from object dictionary descriptions. The modular prototype construction, and high degree of data independence provided by the data base mechanism means that the prototypes are easily amended and are resiliant to change.

As the prototyping system generates standard COBOL source

code programs, it is easily possible to edit and extend these programs to enhance the prototype. It is therefore possible to evolve 'live' application systems through the stepwise refinement of less sophisticated prototypes. This facility is supported by the process partner concept.

The prototyping approach allows users to become involved with the design process, and appraise the system as it is developed. This provides better project control and reduces the possibility of logical errors in the system through the misunderstanding of user requirements.

The data base mechanism is the foundation of the whole prototype generation system. It is a novel implementation of sound academic principles contained in the ANSI-X3-SPARC [ANS01] and entity relationship [Che01] models. This combination gives a simple data base structure, with the performance characteristics of a network data base, and yet retains the flexibility and high level of logical data independence normally associated with a relational data base.

The maintenance of 'live' systems developed from generated prototypes is simplified. Such systems retain the highly modular structure, and the high level of data independence provided by the data base mechanism. This means that changes can be localised to the area of the system directly affected by the amendments. If maintenance amendments are made initially to the object dictionary descriptions, then they

may be tested on the prototype before interrupting the 'live' system. As DB4GL can be used to generate system specifications and documentation from the object dictionary, these can be updated automatically.

9.2. THE SPEC-BUILDER ESTIMATING AND COST CONTROL SYSTEM.

Spec-Builder is an integrated estimating and cost control system that would be suitable for a small to medium sized speculative house builder. The package enables;

i) The estimator to build up a hierarchical bill of quantities for each house type.

ii) The compilation of site estimates by combining house type bills of quantities, with similar bills of quantities for general site costs.

iii) The production of estimates for sites and house types. All estimates are expressed in terms of cost code and item type.

iv) The entry of cost details, whether actual or provisional, into the system interactively.

v) The direct comparisons between estimated and actual costs.

vi) The identification of the sources of deviant construction costs.

Spec-Builder gives the estimator considerable flexibility

when building up estimates, in particular;

i) Estimated costs for high level bill of quantities operations can be inserted if the estimator has insufficient time to break the bill of quantities down to elemental items.

ii) Site estimates can be adjusted easily to allow for local variations in the cost of items such as labour and materials.

iii) Wastage can be included as a percentage factor at all levels in the bill of quantities hierarchy.

Spec-Builder is a complex applications system. In terms of functional specification, it exceeded the bespoke software written for the collaborating company, particularly with regard to the integration of the estimating and cost control functions. However, Spec-Builder does not include the multi-user capabilities and security considerations that would be required of such a system in a medium sized company. The package therefore remains a prototype system, however as a single user system, Spec-Builder could be of use to small speculative house builders where estimating and cost control are carried out by the same person.

9.3. RESEARCH CONCLUSIONS AND RECOMMENDATIONS.

In terms of functional capability, Spec-Builder is a complex applications package. Moreover, the software was developed in just over five weeks, a very short period, which would

confirm the validity of the hypothesis that current software shortages within the building industry could be significantly reduced through the use of advanced software development concepts and tools.

The major limitation of Spec-Builder, that it is not suitable for a multi-user environment, is a reflection on the lack of development time available for DB4GL rather than weakness of the hypothesis. Future versions of DB4GL could include such facilities.

The otherwise satisfactory development of Spec-Builder also illustrates that the DB4GL software development system achieves its objective of providing an integrated strategy for the software life cycle. As such DB4GL could form the foundation of a powerful software development system although further research and development would be necessary.

Such was the confidence gained in this approach that it is recommended that future research programmes should focus on the following areas.

i) The current development methodology forms a support environment in which various systems analysis and design methodologies, such as those proposed by Gane and Sarson [Gan01], can be practised. A useful research area would be the identification and formalisation of a systems analysis and design methodology which would most

efficiently utilise the support environment.

ii) The development of software tools to allow the automatic generation of more sophisticated prototypes. This could include the formulation of a very high level language by which process partners and data manipulation procedures can be described.

iii) Enhancements to include multi-user capabilities and system security.

iv) The introduction of advanced 'expert' interfaces to both the DB4GL development system and generated prototypes. These interfaces could include "query by example" data base enquiry facilities, and "expert" screen and report design aids.

REFERENCES.

[ANS01] ANSI-X3-SPARC/DBMS study group : interim report February 1985

[Bla01] 'Micro Focus advances COBOL'
George Black
Computer Weekly, February 28, 1985

[Che01] The Entity-Relationship model : Towards a unified view of data
ACM Transactions on Data base systems Vol 1, No 1.
P.P.Chen

[Che02] Entity-Relationship to systems analysis and design (North-Holland, Amsterdam, 1980).
P.P.CHEN ed.

[Col01] Building a kind of success story
Sarah Colley
Computing, The magazine February 28 1985

[Con01] Software Directory
Construction Computing October 1984

[DeM01] Structured Analysis and system specification
Tom DeMarco

[Dee01] Fundamentals of Data Base Systems
S.M.Deen

[Gan01] Structured Systems analysis : tools and techniques
Chris Gane and Trish Sarson

[Ewi01] A Computer Estimating System for a Speculative House Builder using 4th Generation Software Tools.
N.A.Ewin
Internal Research Paper R/D/84/3
Department of Building Sheffield Polytechnic

[Ewi02] DB4GL :- A Fourth Generation System Prototyping tool
N.A.Ewin
Internal Research Paper R/D/85/2
Department of Building Sheffield Polytechnic

[Gal01] Handbook of screen format design
Wilbert O. Galitz.

[Gla01] Modern Programming Practices; a report from industry
Robert L. Glass

[Go101] SMALLTALK-80 The language and its implementation
Adele Goldberg and David Robson.

[Inf01] Data Base Systems
Infotech State of the Art Report 1975.

[Ins01] Code of estimating practice.
Estimating practice committee
Institute of Building.

[Mar01] Applications development without programmers,
James Martin

[Mic01] Micro Focus 'Level II' COBOL language manual
Micro Focus Ltd.

[Nat01] Computing and communication in the construction industry
National Consultative Council of the Building and Civil Engineering industries
Standing committee on computing and data co-ordination

[Sak01] Entity-Relationship approach to logical database design
Hirotaka Sakai
pp Entity-Relationship approach to software engineering
ed C.G.Davis et al

[Wes01] Research carried out in the Department of Building, Sheffield City Polytechnic by S.E.Westgate

[Zim01] Phases, Methods, and Tools - A Triad of system development
Rodney P. Zimmerman
pp Entity-Relationship approach to software engineering
ed C.G.Davis et al

BIBLIOGRAPHY

Program Development as a Formal Activity
M. Broy & P.Pepper
IEEE Trans Software Jan 1981.

Computers in Construction
Building (suppliment) December 4, 1981

Information Systems: Theory and Practice
J.G.Burch & F.R.Strater
Hamilton

Application Generation : Rapid Development of Applications Systems Without Conventional Programming.
Central Computer & Telecommunications Agency
(Information Technology in the Civil Service No 3)
London 1983.

Methodology & Tools for Data Base Design
ed S.Ceri
North Holland, Amsterdam 1983

Micro-Computer Programs for the Construction Industry
Chartered Institute of Builders
From the Proceedings of the C.I.O.B. 3rd Synposium on Organisational Management of Construction, 1981.

To Develop a Cost-Monitoring package using a Micro-Computer
Chartered Institute of Builders
From the Proceedings of the C.I.O.B. 3rd Synposium on Organisational Management of Construction, 1981.

Entity-Relationship Approach to Systems Analysis & Design
Ed P.P.Chen
North-Holland

Entity-Relationship Approach to Information modeling & Analysis
Ed P.P.Chen
North-Holland 1983

Structured systems development techniques :
Stategic Planning to System Testing
G. Collins & G.Blay
Pitman

COBOL : Still winning after all these years
Computing, The Magazine June 1985

Micro-Computers for Integrated Accounting
Construction Computing July 1983

Estimating By Computer
Construction Computing July 1983

Computer Aided Estimating
Construction Computing April 1983

Micros in Construction
Construction Industry Computing Association 1981

Entity-Relationship Approach to Software Engineering
ed C.G.Davis, S.Jajodia, P.Ann-Beng, R.T.Yeh
North Holland 1983

Use of Computers in Construction Management
R.H.Day

An Information System for the Construction Industry
Department of the Environment
Directorate General Development (Housing & Construction)
Working Party on Data Co-ordination.

Computing & Communication in the Construction Industry
Department of the Environment
National Consultative Council of the Building & Civil Engineering Industries
Standing Committee on Computing & Data Co-ordination.

Construction Site Studies - Production, Administration and Personnel
G.Forster
Longman Technician Series.

The Fourth Generation
Infotech State of the Art Report 1 1971.

Future Programming
Infotech State of the Art Report
Vols 1 & 2, 1978.

Data Base Technology
Infotech State of the Art Report
Vols 1 & 2, 1978.

Structured Analysis and Design
Infotech State of the Art Report
Vols 1 & 2, 1978.

Small Computer Systems & Their Applications in Construction
Institute of Civil Engineers
Proceedings from I.C.E. Conference 1980.

Software Development - A Rigorous Approach
Cliff B. Jones
Prentice-Hall

Expert Systems : Their impact on the Construction Industry
J.Lansdown
John Lansdown & RIBA Conference Fund 1982.

Application Program Generators - A state of the Art Survey
R.F.Lobell
N.C.C.

An Introduction to Data Base Design
J.K.Lyon

A Simplified Guide to Structured COBOL Programming
D.D.McCracken
John Wiley & Sons

Computers in the Building Industry - What the NFTBE members have to say
National Builder, November 1982.

Choosing and Using Program Generators
C. Naylor
Sigma Technical Press

The CODASYL Approach to Data Base Management
T.W.Olle

Systems Analysis
A.Parkin
Edward Arnold

Modern Methods for COBOL Programmers
J.R.Pugh & D.H.Bell
Prentice-Hall

Micro-Computers for the Builder
QS Weekly, April 22, 1982 pp6-7.

Build Program Technique
John G.Rice
Wiley-Interscience 1981.

Housebuilder's Computer
J.Sellers
Building Technology & Management, February 1983

The CODASYL Data Base Approach : A COBOL example of design & use of a Personnel File
Edgar H. Sibley

Micro-Computers in Construction
G.Trimble
Building Technology & Management, February 1982

Micro-Computers & the control of Sub-Contracted Work
E.Trimble & N.Clark
Building Technology & Management, March 1982

Data Base Design Methodology
M.Vetter & R.N.Maddison
Prentice International.

APPENDIX A.

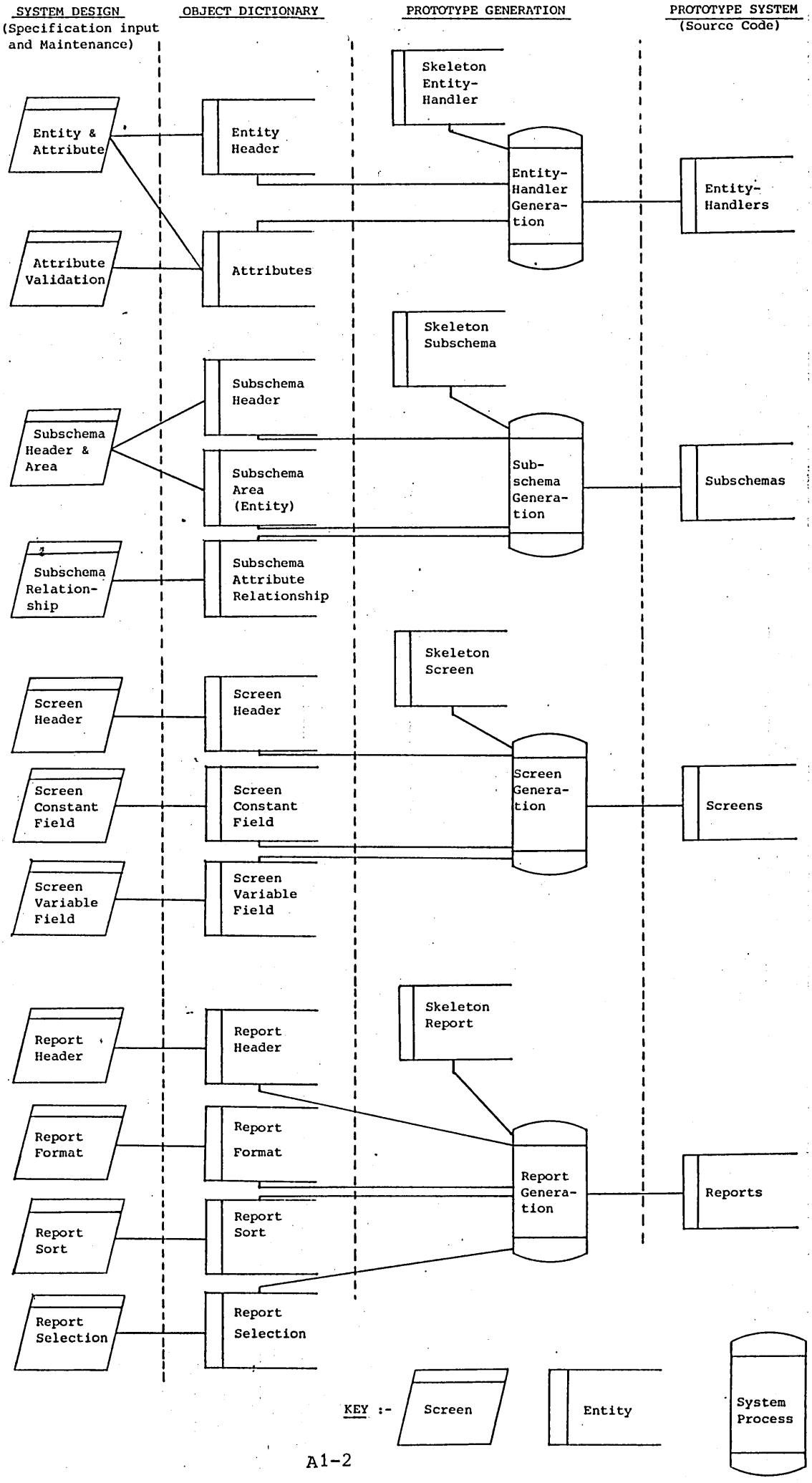
DB4GL SPECIFICATIONS

A1. DB4GL SYSTEM BREAKDOWN (DIAGRAM)

A2. DB4GL OBJECT DICTIONARY ANALYSIS

A3. DB4GL SCREEN FORMATS

A4. SCREEN OBJECT FLOW DIAGRAMS


SYSTEM DESIGN
(Specification input and Maintenance)
OBJECT DICTIONARY
PROTOTYPE GENERATION
PROTOTYPE SYSTEM
(Source Code)
Entity & Attribute
Entity Header
Skeleton Entity-Handler
Entity-Handler Generation
Entity-Handlers
Attribute Validation
Attributes
Skeleton Subschema
Subschema Header
Subschema Header & Area
Subschema Area (Entity)
Sub-schema Generation
Subschemas
Subschema Relation-ship
Subschema Attribute Relationship
Skeleton Screen
Screen Header
Screen Header
Screen Constant Field
Screen Constant Field
Screen Generation
Screens
Screen Variable Field
Screen Variable Field
Skeleton Report
Report Header
Report Header
Report Format
Report Format
Report Generation
Reports
Report Sort
Report Sort
Report Selection
Report Selection
KEY :-
Screen
Entity
System Process

DB4GL OBJECT DICTIONARY ANALYSIS

ENTITY SPECIFICATION

ENTITY NO : 1

ENTITY NAME : ENT-DSC

DESCRIPTION : DATA DICTIONARY :- ENTITY HEADER

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	ENTITY NO	9	0	2	2	0	N	N
6	ENTITY NAME	X	1	12	0	0		
7	DATA DISK	X	1	1	0	0		
8	DESCRIPTION	X	1	40	0	0		

ENTITY SPECIFICATION

ENTITY NO : 2

ENTITY NAME : ATTRIBUTE

DESCRIPTION : DATA DICTIONARY :- ATTRIBUTE

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	ENTITY NO	9	0	2	2	0	N	N
2	ATTRIBUTE NO	9	0	2	2	0	N	N
6	SHORT NAME	X	1	12	0	0		
7	TYPE	X	1	1	0	0		
8	MIN LENGTH	9	0	2	2	0	N	N
9	MAX LENGTH	9	0	2	2	0	N	N
10	INT PLACES	9	0	1	1	0	N	N
11	DEC PLACES	9	0	1	1	0	N	N
12	SIGNED	X	1	1	0	0		
13	PACKED	X	1	1	0	0		
14	START CHAR	X	0	1	0	0		
15	FILL CHAR	X	0	1	0	0		
16	IN PROCESS	9	0	4	4	0	N	N
17	OUT PROCESS	9	0	4	4	0	N	N
18	ACT OR VIRT	X	1	1	0	0		
19	MIN VALUE	9	0	20	9	9	Y	N
20	MAX VALUE	9	0	20	9	9	Y	N
21	VALUE 1	X	0	4	0	0		
22	VALUE 2	X	0	4	0	0		
23	VALUE 3	X	0	4	0	0		
24	VALUE 4	X	0	4	0	0		
25	VALUE 5	X	0	4	0	0		
26	VALUE 6	X	0	4	0	0		
27	VALUE 7	X	0	4	0	0		
28	VALUE 8	X	0	4	0	0		
29	VALUE 9	X	0	4	0	0		
30	VALUE 10	X	0	4	0	0		
31	VALUE 11	X	0	4	0	0		
32	VALUE 12	X	0	4	0	0		

ENTITY SPECIFICATION

ENTITY NO : 5

ENTITY NAME : SCREENHD

DESCRIPTION : DATA DICTIONARY :- SCREEN HEADER

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCREEN NO	9	0	2	2	0	N	N
6	SCREEN NAME	X	1	12	0	0		
7	SCREEN DESC	X	1	40	0	0		
8	SCREEN TITLE	X	1	30	0	0		
9	SCREEN TYPE	X	1	1	0	0		
10	SCREEN DISK	X	1	1	0	0		
11	SCHEMA NO	9	0	2	2	0	N	N
12	PARENT ENT	9	0	2	2	0	N	N
13	CHILD ENT	9	0	2	2	0	N	N
14	LINK ENT 1	9	0	2	2	0	N	N
15	LINK ENT 2	9	0	2	2	0	N	N
16	LINK ENT 3	9	0	2	2	0	N	N
17	LINK ENT 4	9	0	2	2	0	N	N
18	NO OF PAGES	9	0	2	2	0	N	N
19	SCROLL START	9	0	2	2	0	N	N

ENTITY SPECIFICATION

ENTITY NO : 6

ENTITY NAME : SCREENFD

DESCRIPTION : DATA DICTIONARY :- SCREEN FIELD

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCREEN NO	9	0	2	2	0	N	N
2	LINE NUMBER	9	0	3	3	0	N	N
3	COLUMN	9	0	3	3	0	N	N
6	ENTITY NO	9	0	2	2	0	N	N
7	ATTRIBUTE NO	9	0	2	2	0	N	N
8	INPUT OUTPUT	X	1	1	0	0		
9	OPT MAND	X	1	1	0	0		
10	PRE PROCESS	9	0	4	4	0	N	N
11	POST PROCESS	9	0	4	4	0	N	N

ENTITY SPECIFICATION

ENTITY NO : 7

ENTITY NAME : SCREENTX

DESCRIPTION : DATA DICTIONARY :- SCREEN TEXT

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCREEN NO	9	0	2	2	0	N	N
2	LINE NUMBER	9	0	3	3	0	N	N
3	COLUMN	9	0	3	3	0	N	N
6	TEXT VALUE	X	1	40	0	0		
7	TEXT LENGTH	9	0	2	2	0	N	N

ENTITY SPECIFICATION

ENTITY NO : 10

ENTITY NAME : SCHEMAHD

DESCRIPTION : DATA DICTIONARY :- SCHEMA HEADER

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCHEMA NO	9	0	2	2	0	N	N
6	SCHEMA NAME	X	1	12	0	0		
7	SCHEMA DESC	X	1	40	0	0		
8	SCHEMA DISK	X	1	1	0	0		

ENTITY SPECIFICATION

ENTITY NO : 11

ENTITY NAME : SCHEMAET

DESCRIPTION : DATA DICTIONARY :- SCHEMA ENTITIES

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCHEMA NO	9	0	2	2	0	N	N
2	ENTITY NO	9	0	2	2	0	N	N
6	ENTITY DISK	X	1	1	0	0		

ENTITY SPECIFICATION

ENTITY NO : 12

ENTITY NAME : SCHEMALK

DESCRIPTION : DATA DICTIONARY :- SCHEMA LINKAGES

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SCHEMA NO	9	0	2	2	0	N	N
2	SOURCE ENT	9	0	2	2	0	N	N
3	TARGET ENT	9	0	2	2	0	N	N
4	TARGET ATTR	9	0	2	2	0	N	N
5	SOURCE ATTR	9	0	2	2	0	N	N

ENTITY SPECIFICATION

ENTITY NO : 15

ENTITY NAME : REPORTHD

DESCRIPTION : DATA DICTIONARY :- REPORT HEADER

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	REPORT NO	9	0	2	2	0	N	N
6	REPORT NAME	X	1	12	0	0		
7	REPORT DESC	X	1	40	0	0		
8	SCHEMA NO	9	0	2	2	0	N	N
9	SOURCE ENT	9	0	2	2	0	N	N
10	LINK ENT 1	9	0	2	2	0	N	N
11	LINK ENT 2	9	0	2	2	0	N	N
12	LINK ENT 3	9	0	2	2	0	N	N
13	LINK ENT 4	9	0	2	2	0	N	N
14	BREAK ENT	9	0	2	2	0	N	N
15	BREAK ATTR	9	0	2	2	0	N	N
16	DATA START	9	0	2	2	0	N	N
17	FOOT START	9	0	2	2	0	N	N
18	FORM WIDTH	X	2	3	0	0		

ENTITY SPECIFICATION

ENTITY NO : 16

ENTITY NAME : REPORTSE

DESCRIPTION : DATA DICTIONARY :- REPORT SELECTION

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	REPORT NO	9	0	2	2	0	N	N
2	SELECT NO	9	0	2	2	0	N	N
6	ENTITY NO	9	0	2	2	0	N	N
7	ATTR NO	9	0	2	2	0	N	N
8	SELECT COND	X	1	4	0	0		
9	SELECT DESC	X	1	20	0	0		

ENTITY SPECIFICATION

ENTITY NO : 17

ENTITY NAME : REPORTSO

DESCRIPTION : DATA DICTIONARY :- REPORT SORT

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	REPORT NO	9	0	2	2	0	N	N
2	SORT NO	9	0	2	2	0	N	N
6	ENTITY NO	9	0	2	2	0	N	N
7	ATTR NO	9	0	2	2	0	N	N
8	ASCND DESND	X	1	1	0	0		

ENTITY SPECIFICATION

ENTITY NO : 18

ENTITY NAME : REPORTFD

DESCRIPTION : DATA DICTIONARY :- REPORT FORMAT

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	REPORT NO	9	0	2	2	0	N	N
2	REPORT AREA	X	1	1	0	0		
3	LINE NO	9	0	2	2	0	N	N
4	COLUMN	9	0	2	2	0	N	N
6	FIELD TYPE	X	1	1	0	0		
7	TEXT VALUE	X	1	40	0	0		
8	ENTITY NO	9	0	2	2	0	N	N
9	ATTR NO	9	0	2	2	0	N	N

DB4GL SCREEN FORMATS

MENU :

OPTION : MAIN SYSTEM MENU

```
                    SYSTEM MENU                          ACTION : SELECT
-------------------------------------------------------------------------------

                OPTION                              CODE

                DATA BASE MAINTENANCE.....          DAT
                SCREEN MAINTENANCE.......           SCR
                REPORT MAINTENANCE.......           REP

                ENTER OPTION, OR 'X' TO EXIT    <    >
```

MENU : MAIN SYSTEM MENU

OPTION : DATA BASE MAINTENANCE

ENTITY & SCHEMA MAINTENANCE ACTION : SELECT

OPTION	CODE
ENTITY MAINTENANCE.....	ENT
ATTRIBUTE MAINTENANCE.....	ATT
ENTITY GENERATION.....	EGN
SCHEMA ENTITY MAINTENANCE.....	SEN
SCHEMA ATTRIBUTE LINKAGES.....	SLK
SCHEMA GENERATION.....	SGN
ENTITY SPECIFICATION REPORT...	REP

ENTER OPTION, OR 'X' TO EXIT < >

MENU : DATA BASE MAINTENANCE

OPTION : ENTITY-MAINTENANCE

```
AREA : SCREEN                    ENTITY MAINTENANCE                    ACTION : ENQUIRE
-------------------------------------------------------------------------------------------

          ENTITY NO :  1       NAME : ENTITY          DISK : B

          DESCRIPTION : DB4GL ENTITY OBJECT DESCRIPTION

 ATTR                          MIN     MAX     INT     DEC                    IN    OUT
  NO    SHORT NAME    TYPE LENGTH LENGTH PLACE PLACE SIGN PACK              PROC  PROC  A/V

   1    ENTITY NO        9      0      2     2     0    N                     0     0    A
   6    SHORT NAME       X      1     12     0     0    N                     0     0    A
   7    DESCRIPTION      X      0     40     0     0    N                     0     0    A
   8    DISK             X      1      1     0     0    N                     0     0    A




-------------------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : SCROLL (U)P, SCROLL (D)OWN, (E)DIT, OR E(X)IT
```

MENU : DATA BASE MAINTENANCE

OPTION : SCHEMA ENTITY MAINTENANCE

```
AREA : SCREEN                  SCHEMA ENTITIES                  ACTION : ENQUIRE
-------------------------------------------------------------------------------

          SCHEMA NO :  1            SCHEMA NAME : HASSALL    DISK : A

          DESCRIPTION : HASSALL SYSTEM SCHEMA

          ENTITY NO    ENTITY NAME      DISK

              1          ENTITY01         A
              2          ENTITY02         A
              3          ENTITY03         A
              4          ENTITY04         A
              5          ENTITY05         A
              6          ENTITY06         A
              7          ENTITY07         A
             10          ENTITY10         A
             11          ENTITY11         A
             12          ENTITY12         A
             20          ENTITY20         A

-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : SCROLL (U)P, SCROLL (D)OWN, (E)DIT, OR E(X)IT
```

MENU : DATA BASE MAINTENANCE

OPTION : SCHEMA ATTRIBUTE LINKAGES

AREA : SCREEN SCHEMA LINKAGE ACTION : ENQUIRE

SCHEMA NO : 1 SCHEMA NAME : HASSALL

SOURCE		TARGET	
ENTITY	ATTRIBUTE	ENTITY	ATTRIBUTE
1 ENTITY01	1 HOUSE CODE	2 ENTITY02	1 HOUSE CODE
1 ENTITY01	1 HOUSE CODE	11 ENTITY11	1 HOUSE CODE
2 ENTITY02	1 HOUSE CODE	1 ENTITY01	1 HOUSE CODE
2 ENTITY02	2 OP LEVEL	6 ENTITY06	1 OP-ITEM TYPE
2 ENTITY02	3 OP CODE	6 ENTITY06	2 OP-ITEM CODE
3 ENTITY03	1 SITE CODE	4 ENTITY04	1 SITE CODE
3 ENTITY03	1 SITE CODE	5 ENTITY05	1 SITE CODE
3 ENTITY03	1 SITE CODE	10 ENTITY10	1 SITE CODE
4 ENTITY04	6 HOUSE CODE	1 ENTITY01	1 HOUSE CODE
4 ENTITY04	1 SITE CODE	3 ENTITY03	1 SITE CODE
5 ENTITY05	1 SITE CODE	3 ENTITY03	1 SITE CODE
5 ENTITY05	2 OP LEVEL	6 ENTITY06	1 OP-ITEM TYPE
5 ENTITY05	3 OP CODE	6 ENTITY06	2 OP-ITEM CODE

ENTER OPTION : < >
OPTIONS : SCROLL (U)P, SCROLL (D)OWN, (E)DIT, OR E(X)IT

MENU : MAIN SYSTEM MENU

OPTION : SCREEN MAINTENANCE

```
                    SCREEN MAINTENANCE                ACTION : SELECT
--------------------------------------------------------------------------------

                OPTION                                CODE

                SCREEN HEADER MAINTENANCE.....        HDR
                SCREEN FIELD MAINTENANCE......        FLD
                SCREEN TEXT MAINTENANCE.......        TXT
                SCREEN GENERATION.............        GEN
                SCREEN FORMAT PRINT...........        PRI

                ENTER OPTION, OR 'X' TO EXIT    <    >
--------------------------------------------------------------------------------
```

MENU : SCREEN MAINTENANCE

OPTION : SCREEN FIELD MAINTENANCE

AREA : SCREEN SCREEN FIELD MAINTENANCE ACTION : ENQUIRE

SCREEN NO : 1 SCREEN NAME : HSE-MNT

LINE	COLUMN	ENTITY	ATTRIBUTE	I/O	O/M	PROCESS1	PROCESS2
3	19	1 ENTITY01	1 HOUSE CODE	I	O	0	0
3	36	1 ENTITY01	6 HOUSE NAME	I	M	0	0
3	66	1 ENTITY01	10 ABBREVIATION	I	M	0	0
5	20	1 ENTITY01	7 HOUSE CLASS	I	M	0	0
5	40	1 ENTITY01	8 DESCRIPTION1	I	M	0	0
6	40	1 ENTITY01	9 DESCRIPTION2	I	O	0	0
8	19	1 ENTITY01	11 FLOOR AREA	I	M	0	0
8	46	1 ENTITY01	12 PLOT FRONT	I	M	0	0
8	64	1 ENTITY01	13 GARAGE CODE	I	O	0	0
8	78	1 ENTITY01	14 HEATING CODE	I	O	0	0
99	8	2 ENTITY02	2 OP LEVEL	I	O	0	0
99	15	2 ENTITY02	3 OP CODE	I	M	0	0
99	24	6 ENTITY06	7 DESCRIPTION1	O	O	0	0

ENTER OPTION : < >
OPTIONS : SCROLL (U)P, SCROLL (D)OWN, (E)DIT, OR E(X)IT

MENU : SCREEN MAINTENANCE

OPTION : SCREEN HEADER MAINTENANCE

```
AREA : SCREEN                    SCREEN HEADER                    ACTION : OPTION
-------------------------------------------------------------------------------

                SCREEN NUMBER       :   1
                       NAME         :  HSE-MNT
                       DESCRIPTION  :  HASSALL :- HOUSE DETAIL MAINTENANCE
                       TITLE        :  HOUSE DETAIL MAINTENANCE
                       TYPE         :  C
                       DISK         :  A
                SCHEMA NUMBER       :   1     HASSALL
                PARENT ENTITY NO    :   1     ENTITY01
                CHILD  ENTITY NO    :   2     ENTITY02
                LINK 1 ENTITY NO    :   6     ENTITY06
                LINK 2 ENTITY NO    :   0
                LINK 3 ENTITY NO    :
                LINK 4 ENTITY NO    :
                NUMBER OF PAGES     :   6
                SCROLL START LINE   :  13


-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : (A)MEND, (D)ELETE, OR E(X)IT
```

MENU : SCREEN MAINTENANCE

OPTION : SCREEN TEXT MAINTENANCE

```
AREA : SCREEN                SCREEN TEXT FIELD MAINTENANCE    ACTION : ENQUIRE
-------------------------------------------------------------------------------

          SCREEN NO :  1              SCREEN NAME : HSE-MNT

          LINE       COLUMN     TEXT VALUE

            3            6      HOUSE CODE :
            3           29      NAME :
            3           51      ABBREVIATION :
            5            6      HOUSE CLASS :
            5           26      DESCRIPTION :
            8            6      FLOOR AREA :
            8           30      PLOT FRONTAGE :
            8           55      GARAGE :      HEATING :
           10            6      OPERATION BUILD UP DETAILS :-
           11            6      LEVEL    CODE      DESCRIPTION
           11           59      QUANTITY  WASTAGE


-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : SCROLL (U)P, SCROLL (D)OWN, (E)DIT, OR E(X)IT
```

MENU : MAIN SYSTEM MENU

OPTION : REPORT MAINTENANCE

```
                        REPORT MAINTENANCE                ACTION : SELECT
-------------------------------------------------------------------------------

                OPTION                                  CODE

                REPORT HEADER MAINTENANCE.....   HDR
                REPORT FIELD MAINTENANCE......   FLD
                REPORT SORT MAINTENANCE.......   SOR
                REPORT SELECTION MAINTENANCE..   SEL
                REPORT GENERATION.............   GEN

                ENTER OPTION, OR 'X' TO EXIT    <   >
-------------------------------------------------------------------------------
```

MENU : REPORT MAINTENANCE

OPTION : REPORT HEADER MAINTENANCE

```
AREA : SCREEN                REPORT HEADER INFORMATION        ACTION : ENQUIRE
--------------------------------------------------------------------------------

                    REPORT NUMBER       :  1
                    REPORT NAME         : EST1-PLT
                    DESCRIPTION         : SITE ESTIMATE BY PLOT, DIRECT HOUSE COST
                    SCHEMA NUMBER       :  1     HASSALL
                    SOURCE ENTITY       : 10     ENTITY10
                    LINK ENTITY 1       :  1     ENTITY01
                    LINK ENTITY 2       :  4     ENTITY04
                    LINK ENTITY 3       :  3     ENTITY03
                    LINK ENTITY 4       :  0     ENTITY01
                    BREAK ENTITY        : 10     ENTITY10
                    BREAK ATTRIBUTE     :  2
                    DATA START LINE     :  9
                    FOOTING START LINE  : 55
                    FORM WIDTH 80/120   : 120
                    PAGE NUMBERS Y/N    : N


--------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : (A)MEND, (D)ELETE, OR E(X)IT
```

MENU : REPORT MAINTENANCE

OPTION : REPORT FIELD MAINTENANCE

```
AREA : SCREEN                REPORT FIELD DETAILS              ACTION : CREATE
-------------------------------------------------------------------------------

                    REPORT NUMBER :  1

                    AREA (1/2/3)  : 1

                    LINE NUMBER   :  1

                    COLUMN NUMBER :  30

                    TYPE (T/D)    : T

                    ENTITY NO     :  0

                    ATTRIBUTE NO  :  0

                    TEXT VALUE    : REPORT TITLE


-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : (A)MEND, (D)ELETE, OR E(X)IT
```

MENU : REPORT MAINTENANCE

OPTION : REPORT SORT MAINTENANCE

```
AREA : SCREEN                REPORT SORT DETAILS                ACTION : ENQUIRE
-------------------------------------------------------------------------------

                    REPORT NUMBER     :   1
                    SEQUENCE NUMBER   :   1
                    ENTITY NUMBER     :  10   ENTITY10
                    ATTRIBUTE NUMBER  :   2   PLOT NO
                    ASCEND/DESCEND    :  A


-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : (A)MEND, (D)ELETE, OR E(X)IT
```

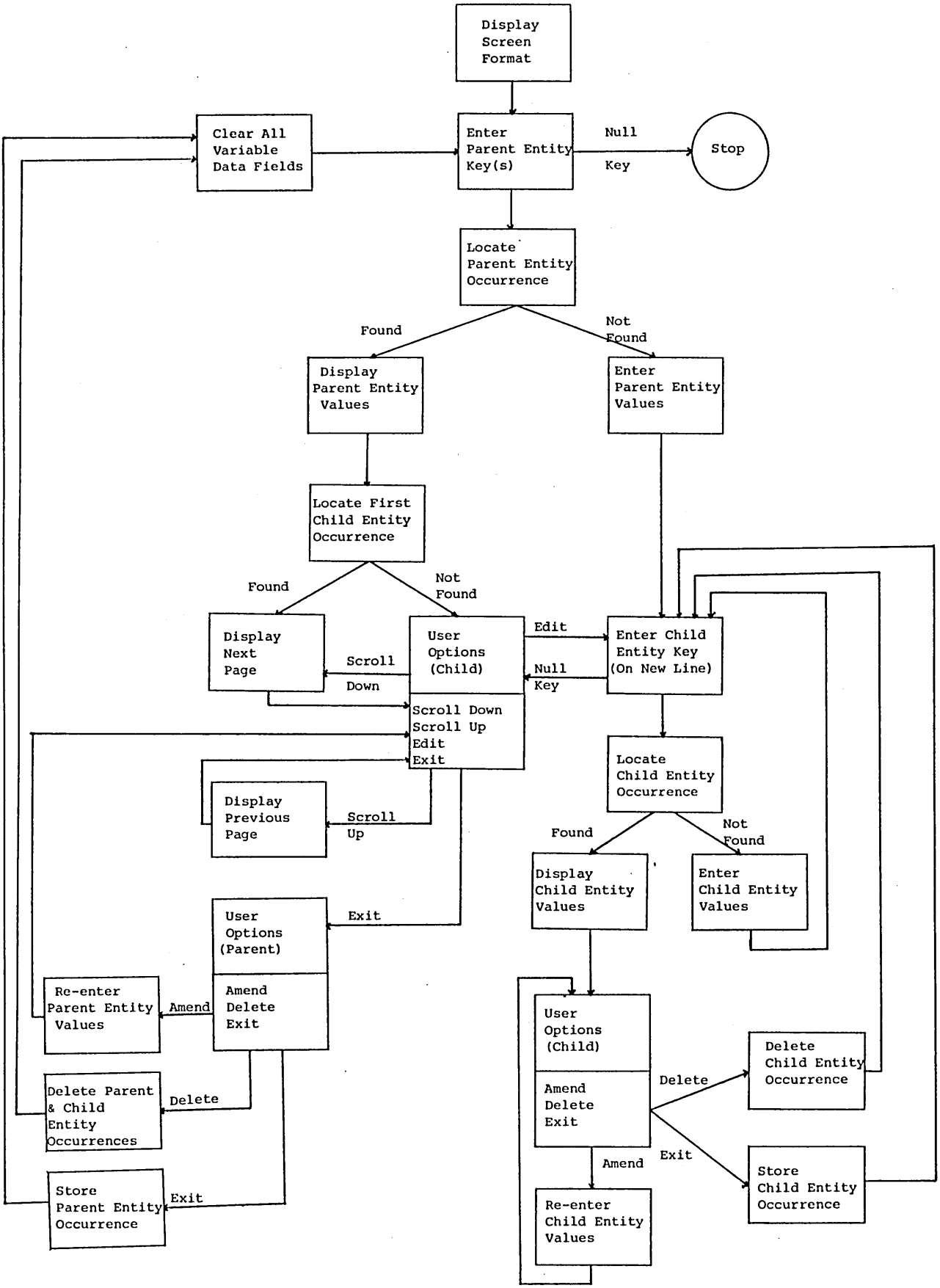
MENU : REPORT MAINTENANCE

OPTION : REPORT SELECTION MAINTENANCE

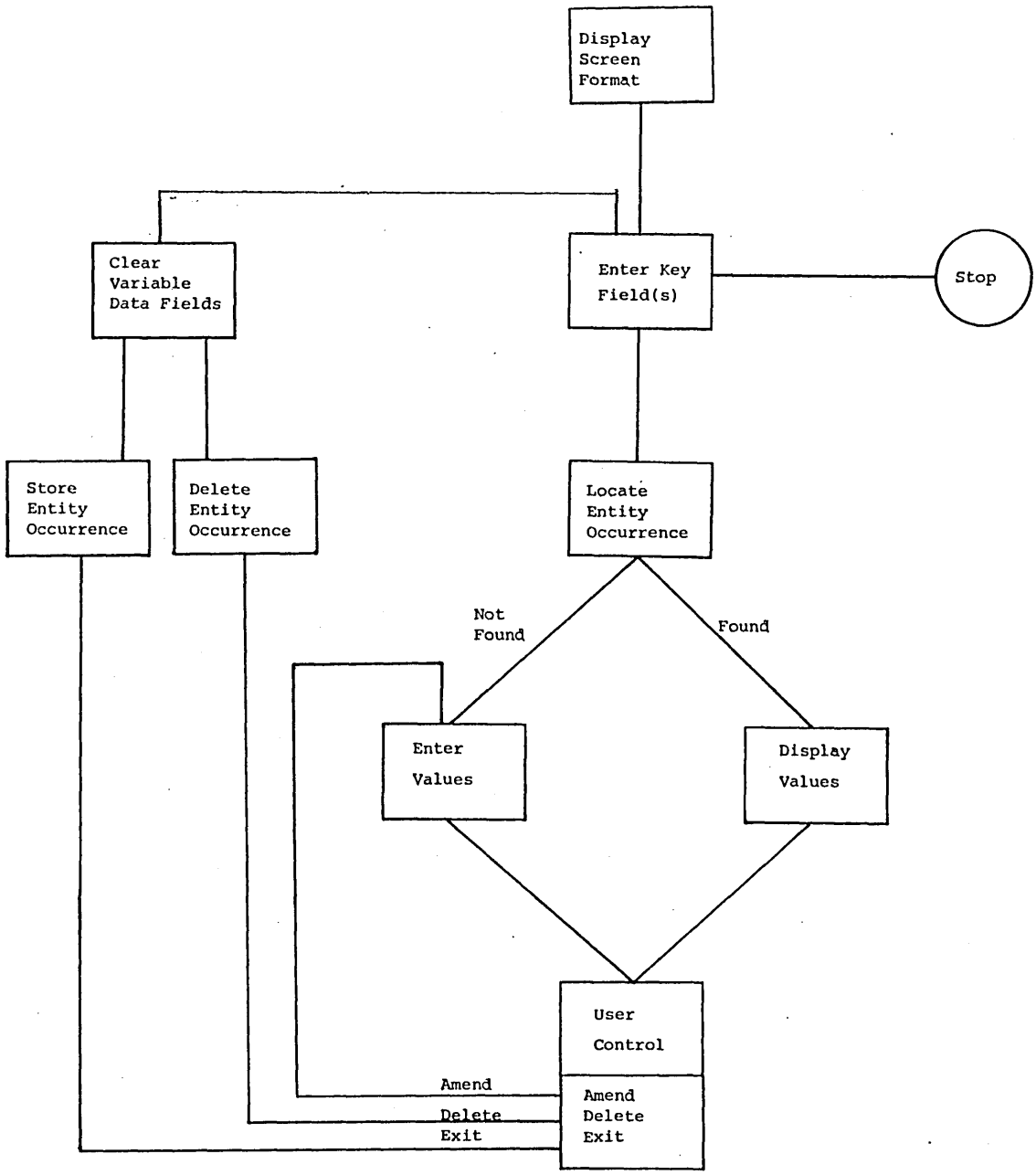
```
AREA : SCREEN              REPORT ENTITY SELECTION          ACTION : ENQUIRE
-------------------------------------------------------------------------------

                    REPORT NUMBER          :  1
                    SEQUENCE NUMBER        :  1
                    ENTITY NUMBER          : 10  ENTITY10
                    ATTRIBUTE NUMBER       :  1  SITE CODE
                    PARAMETER DESCRIPTION  : SITE NUMBER
                    SELECT CONDITION       : =

-------------------------------------------------------------------------------
ENTER OPTION : < >
OPTIONS : (A)MEND, (D)ELETE, OR E(X)IT
```


Display Screen Format
Clear All Variable Data Fields
Enter Parent Entity Key(s)
Null Key
Stop
Locate Parent Entity Occurrence
Found
Not Found
Display Parent Entity Values
Enter Parent Entity Values
Locate First Child Entity Occurrence
Found
Not Found
Display Next Page
User Options (Child)
Scroll Down
Edit
Null Key
Enter Child Entity Key (On New Line)
Scroll Down
Scroll Up
Edit
Exit
Locate Child Entity Occurrence
Display Previous Page
Scroll Up
Found
Not Found
Display Child Entity Values
Enter Child Entity Values
User Options (Parent)
Exit
Re-enter Parent Entity Values
Amend
Amend
Delete
Exit
User Options (Child)
Amend
Delete
Exit
Delete
Delete Child Entity Occurrence
Delete Parent & Child Entity Occurrences
Delete
Amend
Exit
Store Child Entity Occurrence
Re-enter Child Entity Values
Store Parent Entity Occurrence
Exit

SIMPLE SCREEN OBJECT FLOW DIAGRAM



APPENDIX B.

ESTIMATING AND COST CONTROL ANALYSIS RESULTS

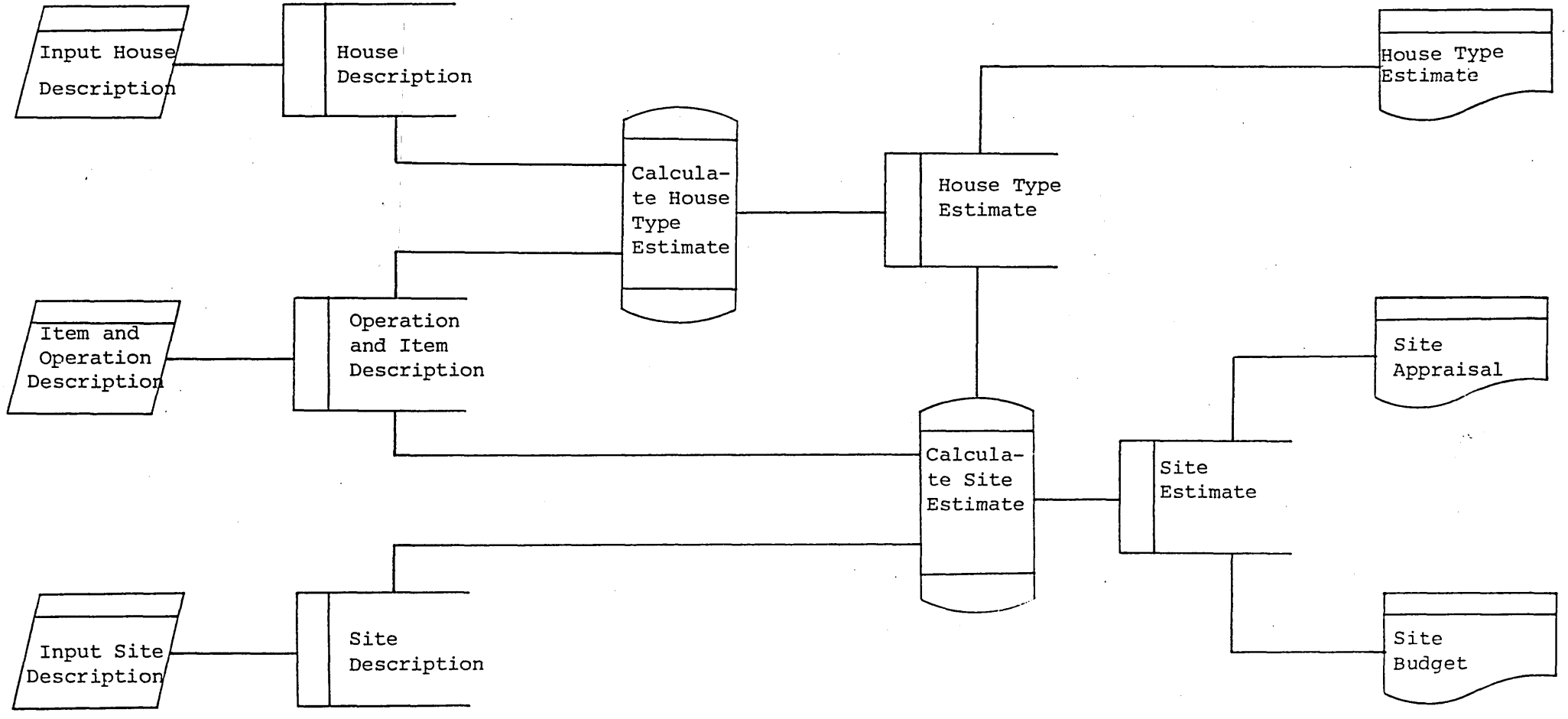
B1. LOGICAL DATA FLOW MODELS (DIAGRAMS)

B2. MANUAL REPORT FORMATS

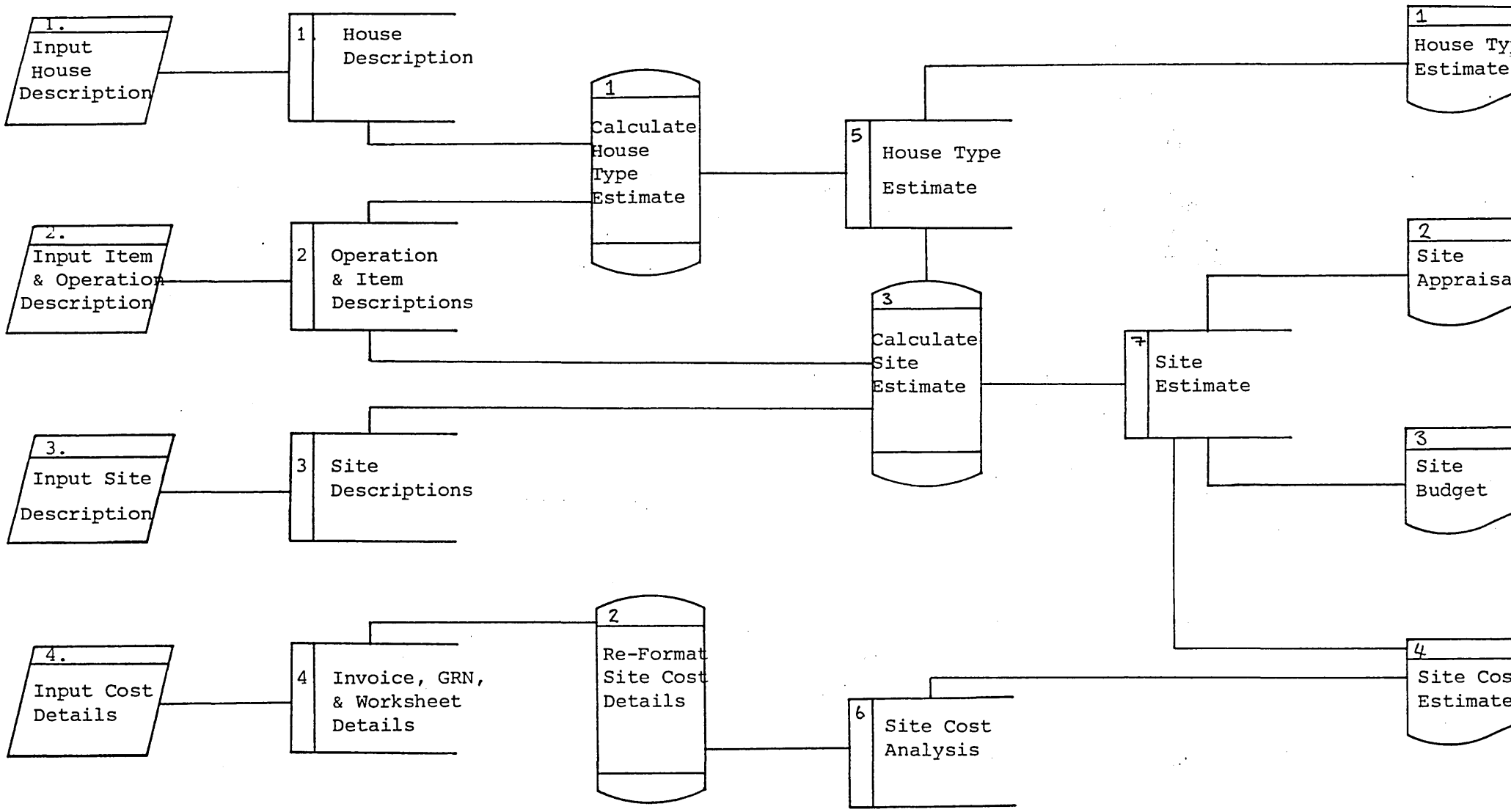
B3. LOGICAL SYSTEM OBJECT ANALYSIS

B4. ESTIMATING SYSTEM ACKNOWLEDGMENT

Logical Data Flow Describing the Estimating Function in a Speculative House Builder



Logical Data Flow Model Describing the Estimating & Cost Control Functions in a Speculative House Builder



PROJECT APPRAISAL

Site	
Location	
Date	
No of units	
Site area	
Density	

	Specification details	House type	House type	House type	House type	House type	House type	House type	Totals
	Floor area								
	G.F. area:								
	Plot front								
	No on site								
SUGGESTED SELLING PRICE									
INCLUSIVE OF :- CENTRAL HEATING									
GARAGE									
BUILDING COSTS									
1. FOUNDATIONS									
2. DRAINAGE									
3. SUPERSTRUCTURE									
4. EXT. WORKS & SERVICES									
5. DIRECT COSTS									
6. SITE ONCOSTS									
SUB-TOTAL 1									
ABNORMAL COSTS									
1.									
2.									
3.									
4.									
5.									
6.									
SUB-TOTAL 2									
ROADS & SEWERS									
R&S ABNORMALS									
OFF SITE WORKS									
SUB-TOTAL 3									
SELLING COSTS @ % of SALES									
PROFESSIONAL FEES :- ARCHITECT									
QS. ENGINEER									
SUB-TOTAL 4									
LAND COST INCLUDING FEES									
TOTAL 1									
SELLING PRICE									
LESS TOTAL 1									
N.P. REAL/ANTICIPATED @ % on	SALES PRICE								
LAND VALUE									

SUMMARY OF BUDGET COST

SITE:-
NO. OF UNITS:-

PROG:-
ISSUE:-

PRICE DATE:-
PREP DATE:-

SECTION												TOTAL COST
FOUNDATIONS												
DRAINAGE												
SUPERSTRUCTURE												
GARAGE												
EXTERNAL WORKS												
DIRECT COSTS												
SITE ON COSTS												
TOTAL												
ROADS & SEWERS												
SPECIAL WORKS												
DEVELOPMENT COST												
MATERIALS												
SUBCONTRACT												
WAGES												
PLANT & EQUIP.												
HAULAGE												
SITE EXPENSES												
SUPERVISION												
TOTAL												
ROADS & SEWERS												
SPECIAL WORKS												
DEVELOPMENT COST												

LOGICAL SYSTEM OBJECT ANALYSIS

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	Spec-Builder		
DATE			
DATA STORE NO	1	PAGE	1
NAME	House Description		

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
House NAME	X	12		
House Description	X	60		
House Class	X	3		
Abbreviation	X	2		
Floor Area	9		4	2
Plot Frontage	9		4	2
Garage Code	X	2		
Central Heating Code	X	2		
Major Operations				
Quantity	9		6	3
Operation Level (Primary/Secondary)	X	1		
Wastage %	9		2	2

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
DATA STORE NO	2.
PAGE	1
NAME	OPERATION AND ITEM DESCRIPTIONS

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
OPERATION LEVEL OF ITEM TYPE	X	1		
OPERATION OR ITEM NAME	X	8		
DESCRIPTION	X	60		
UNIT	X	8		
COST CODE	9		3	0
ITEM COST/PRICE	9	6	6	3.
SUB-OPERATION OR ITEM DETAILS				
QUANTITY	9		6	3
WASTAGE %	9		2	2
OPERATION OR ITEM	X	1		

-- D B 4 G L --

LOGICAL DATA STORE ANALYSIS

PROJECT	Spec-Builder
DATE	
DATA STORE NO	3.
PAGE	1
NAME	Site Description

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
Site Number	9		3	0
Site Name	X	30		
Address	X	130		
Plot Details				
Plot Number	9		3	0
House Type	X	5		
Sale Price	9		6	2
Site Operations				
Quantity	9		6	3
Operation Level (Primary/Secondary)	X	1		
Wastage %	9		2	2

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	Spec Builder
DATE	
DATA STORE NO	4
PAGE	1
NAME	Site Cost Details

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
Source Reference	X	10		
Provisional or Actual Cost	X	1		
Supplier/Operative/Sub-Contractor Name	X	30		
Invoice/Worksheet/G.R.N. Date	X	10		
Invoice/Worksheet/G.R.N. Item Details				
Site No	9		3	0
Plot No	9		3	0
Cost Code	9		3	0
Item Description	X	20		
Item Type (Lab/Mat/Sub/etc)	X	3		
Cost	9		6	2

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
DATA STORE NO	5.
PAGE	1
NAME	HOUSE TYPE ESTIMATE

VALUE NAME		TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
HOUSE TYPE.					
COST CODE	ITEM TYPES	9		3	0
MATERIAL COST		9		6	2
SUB CONTRACT COST		9		6	2
LABOUR COST		9		6	2
PLANT + HAULAGE COST		9		6	2
ON COSTS		9		6	2
SELLING COST		9		6	2
SUPERVISION COST		9		6	2

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
DATA STORE NO	6
PAGE	1
NAME	SITE COST ANALYSIS

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
SITE NUMBER	9		8	Ø
PLOT NUMBER	9		3	Ø
COST CODE	9		3	Ø
MATERIAL COST	9		6	2
SUB CONTRACT COST	9		6	2
LABOUR COST	9		6	2
PLANT/HAULAGE COST	9		6	2
ONCOSTS	9		6	2
SELLING COST	9		6	2
SUPERVISION COST	9		6	2
MATERIAL PROVISIONS	9		6	2
SUB CONTRACT PROVISIONS	9		6	2
LABOUR PROVISIONS	9		6	2
PLANT/HAULAGE PROVISIONS	9		6	2
ONCOST PROVISIONS	9		6	2
SELLING PROVISIONS	9		6	2
SUPERVISION PROVISIONS	9		6	2

-- D B 4 G L --
LOGICAL DATA STORE ANALYSIS

PROJECT	SPEC-BUILDER		
DATE			
DATA STORE NO	7	PAGE	1
NAME	SITE ESTIMATE		

VALUE NAME	TYPE (9/X)	CHARS	INT PLACES	DEC PLACES
SITE NUMBER	9		3	Ø
PLOT NUMBER	9		3	Ø
COST CODE	9		3	Ø
MATERIAL COST	9		6	2
SUB-CONTRACT COST	9		6	2
LABOUR COST	9		6	2
PLANT + HAULAGE COST	9		6	2
ON COSTS	9		6	2
SELLING	9		6	2
SUPERVISION	9		6	2

-- D B 4 G L --
LOGICAL DATA TRANSITION ANALYSIS

PROJECT	Spec-Builder
DATE	
DATA TRANS NO	1
PAGE	1
NAME	Calculate House Type Estimate.

SOURCE VALUES	DATA STORE
House Name/Type	1
Operation Details	1 + 2
Level	..
Cost Code	..
Sub-Operation/Item	..
Quantity	..
Wastage	..
Item Details	2
Cost/Price	..
Item Type	..

TARGET VALUES	DATA STORE
House Name/Type	5
Cost Code	
Material Cost	
Sub Contract Cost	
Plant + Haulage Cost	
On Costs	
Selling Cost	
Supervision Cost.	

LOGICAL DATA TRANSITION PROCESS DESCRIPTION.

PROCESS NO : 1.

TITLE : CALCULATE HOUSE TYPE ESTIMATE.

OBJECTIVE : TO CALCULATE A HOUSE TYPE ESTIMATE FROM INFORMATION CONTAINED IN THE BILL OF QUANTITIES.

SUMMARY.

1. Break down the secondary operations specified in logical data store 1 into primary operations and items, using the bill of quantities hierarchy, contained in logical data store 2.

2. Multiply secondary and primary operation quantities by item prices to calculate estimated operation costs.

3. Arrange estimate results by cost code and item type in logical data store 5.

-- D B 4 G L --

LOGICAL DATA TRANSITION ANALYSIS

PROJECT	SPEC-BUILDER		
DATE			
DATA TRANS NO	2	PAGE	1
NAME	RE-FORMAT SITE COST DETAILS		

SOURCE VALUES	DATA STORE
SOURCE REFERENCE	4
PROVISIONAL or ACTUAL COST	..
INVOICE/WORKSHEET/GRN ITEM DETAILS	..
SITE NUMBER	..
PLOT NUMBER	..
COST CODE	..
ITEM TYPE	..
COST	..
HOUSE TYPE APPORTION FACTOR (PLOT FRONTAGE)	1

TARGET VALUES	DATA STORE
SITE NUMBER	6.
PLOT NUMBER	
COST CODE	
ACTUAL COSTS BY ITEM TYPE	
PROVISIONAL COSTS BY ITEM TYPE	

LOGICAL DATA TRANSITION PROCESS DESCRIPTION.

PROCESS NO : 2.

TITLE : RE-FORMAT SITE COST DETAILS.

OBJECTIVE : TO RE-ARRANGE SITE COST INFORMATION INTO A FORM WHICH MAY BE COMPARED DIRECTLY WITH ESTIMATING DATA.

SUMMARY.

1. Extract data from logical data store 4, and change format from source format into cost code format, ie by site number, plot number and cost code.

2. Add re-formated data to corresponding data items in logical data store 6.

3. Apportion indirect site costs to plots according to plot frontage.

-- D B 4 G L --

LOGICAL DATA TRANSITION ANALYSIS

PROJECT	SPEC-BUILDER		
DATE			
DATA TRANS NO	3	PAGE	1
NAME	CALCULATE SITE ESTIMATE.		

SOURCE VALUES	DATA STORE
HOUSE TYPE	5
COST CODE	"
ESTIMATED COSTS BY ITEM TYPE	"
PLOT NUMBER	2
SITE OPERATION DETAILS	2
LEVEL	"
COST CODE	"
SUB-OPERATION/ITEM	"
QUANTITY	"
WASTAGE %	"
ITEM DETAILS	"
COST/PRICE	"
APPORTION TYPE FACTOR (PLOT FRONTAGE)	1

TARGET VALUES	DATA STORE
SITE NUMBER	7
PLOT NUMBER	
COST CODE	
ESTIMATED COSTS BY ITEM TYPE.	

LOGICAL DATA TRANSITION PROCESS DESCRIPTION.

PROCESS NO : 3.

TITLE : CALCULATE SITE ESTIMATE.

OBJECTIVE : TO CALCULATE A SITE ESTIMATE BY COMBINING HOUSE TYPE ESTIMATES WITH AN ESTIMATE FOR INDIRECT SITE COSTS DERIVED FROM THE INDIRECT SITE COST BILL OF QUANTITIES.

SUMMARY.

1. Sreak down the secondary operations specified in logical data store 3 into primary operations and items, using the bill of quantities hierarchy contained in logical data store 2.

2. Multiply secondary and primary operation quantities by itme prices to calculate estimated operation costs.

3. Apportion indirect site costs to plots according to plot frontage.

4. Calculate direct plot estimates using house type estimates from logical data store 5.

5. Arrange estimate results by plot number, cost code and item type in logical data store 7.

-- D B 4 G L --
LOGICAL INPUT PROCESS ANALYSIS

PROJECT	Spec-Builder		
DATE			
INPUT PROC NO	1	PAGE	1
NAME	House Description		
INPUT SOURCE	'TAKEN OFF' House Type Drawings		
TARGET DATA STORES	1.		

INPUT VALUES	DATA STORE
House Name	1
House Description	
House Class	
Abbreviation	
Floor Area	
Plot Frontage	
Garage Code	
Central Heating Code	
Major Operations	
Quantity	
Operation Level (Primary/Secondary)	
Wastage %	

-- D B 4 G L --
LOGICAL INPUT PROCESS ANALYSIS

PROJECT	Spec-Builder		
DATE			
INPUT PROC NO	2.	PAGE	1
NAME	ITEM + OPERATION DESCRIPTIONS.		
INPUT SOURCE	BILLS OF QUANTITIES (TAKEN OF' HOUSE TYPE DRAWING)		
TARGET DATA STORES	2.		

INPUT VALUES	DATA STORE
OPERATION LEVEL / OR ITEM	2.
OPERATION / ITEM NAME	
DESCRIPTION	
UNIT	
COST CODE	
ITEM COST / PRICE	
SUB - OPERATION OR ITEM DETAILS	
QUANTITY	
WASTAGE %	
OPERATION OR ITEM	

-- D B 4 G L --
LOGICAL INPUT PROCESS ANALYSIS

PROJECT	SPEC-BUILDER		
DATE			
INPUT PROC NO	3	PAGE	1
NAME	SITE DESCRIPTIONS		
INPUT SOURCE	SITE PLAN, SITE BILLS OF QUANTITIES		
TARGET DATA STORES	3		

INPUT VALUES	DATA STORE
SITE NUMBER	3.
SITE NAME	
SITE ADDRESS	
PLOT DETAILS	
PLOT NUMBER	
HOUSE TYPE	
SALE PRICE	
SITE OPERATIONS	
QUANTITY	
OPERATION LEVEL (PRIMARY/SECONDARY)	
WASTAGE %	

-- D B 4 G L --
LOGICAL INPUT PROCESS ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
INPUT PROC NO	4
PAGE	1
NAME	COST DETAILS
INPUT SOURCE	WORKSHEETS, INVOICES, GOODS RECEIVED NOTES
TARGET DATA STORES	4.

INPUT VALUES	DATA STORE
SOURCE REFERENCE	4.
PROVISIONAL or ACTUAL COSTS	
SOURCE NAME	
DATE	
ITEM DETAILS	
SITE NO USED AT	
PLOT NO	
COST CODE	
ITEM DESCRIPTION	
ITEM TYPE (LAB/MATERIAL/etc)	
COST	

-- D B 4 G L --

LOGICAL OUTPUT PROCESS ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
OUTPUT PROC NO	1
PAGE	1
NAME	HOUSE TYPE ESTIMATE
OUTPUT TARGET	ESTIMATOR, DRAWING OFFICE.
OUTPUT FORMAT	FORMAT AS SITE BUDGET.
SOURCE DATA STORES	5, 1

OUTPUT VALUES	DATA STORE
HOUSE NAME	1
HOUSE DESCRIPTION	1
HOUSE ESTIMATE ANALYSIS, BY COST CODE, BY COST TYPE	5.

-- D B 4 G L --

LOGICAL OUTPUT PROCESS ANALYSIS

PROJECT	SPEC-BUILDER
DATE	
OUTPUT PROC NO	2
PAGE	1
NAME	SITE APPRAISAL
OUTPUT TARGET	ESTIMATOR, LAND BUYING DEPT, MANAGEMENT
OUTPUT FORMAT	BASED ON MANUAL APPRAISAL DOCUMENT.
SOURCE DATA STORES	1

OUTPUT VALUES	DATA STORE
SITE NUMBER	7
PLOT NUMBER	7
HOUSE NAME	1
SITE ESTIMATE ANALYSIS	
BY PLOT	7.
BY COST CODE	
BY COST TYPE	

-- D B 4 G L --
LOGICAL OUTPUT PROCESS ANALYSIS

PROJECT	Spec-Builder
DATE	
OUTPUT PROC NO	3
PAGE	1
NAME	Site Budget
OUTPUT TARGET	Estimator, Accountant, Management
OUTPUT FORMAT	Based on Manual Site Budget
SOURCE DATA STORES	, 1

OUTPUT VALUES	DATA STORE
Site Number	7
Plot Number	7
House Name	1
Site Estimate Analysis	
By Plot	7.
By Cost Code	
By Cost Type	

-- D B 4 G L --
LOGICAL OUTPUT PROCESS ANALYSIS

PROJECT	Spec-Builder
DATE	
OUTPUT PROC NO	4
PAGE	1
NAME	Site Estimate 'v' Cost Comparison.
OUTPUT TARGET	Estimator, Cost Control, Management.
OUTPUT FORMAT	Based on Cost Report + Site Budget Format
SOURCE DATA STORES	6, 1, 7

OUTPUT VALUES	DATA STORE
Site Number	6
Plot Number	6
House Name	
Site Actual Cost, Provisional Cost and Estimate Comparison	
By Plot	6, 7
By Cost Code	
By Cost Type	

HASSALL HOMES

P. Hassall Limited
500 Charlotte Road,
Sheffield S2 4ER
Telephone (0742) 760191

MW/DB

19th December, 1984

Mr. R. Oxley,
Dept. of Building,
Sheffield City Polytechnic,
Pond Street,
SHEFFIELD. S1 1WB

Dear Sir,

Computer Estimating for Speculative House Builder and Data Base Systems

We consider that Mr. N.A. Ewin has produced an extremely comprehensive report that would suit any house builder of sufficient size to warrant the cost of the software and the necessary staff to keep the system up to date. Mr. Ewin has obviously spent a great deal of time on this matter and appears to have covered all the essential points.

Yours faithfully,
P. HASSALL LIMITED

M. WARD
Company Secretary

Registered Office: 500 Charlotte Road, Sheffield S2 4ER
Registered No. 535148 England
A Subsidiary Company of Raine Industries PLC

APPENDIX C.

SPEC-BUILDER DESCRIPTION

C1. PHYSICAL DATA FLOW MODEL

C2. SPEC-BUILDER OBJECT DICTIONARY ANALYSIS

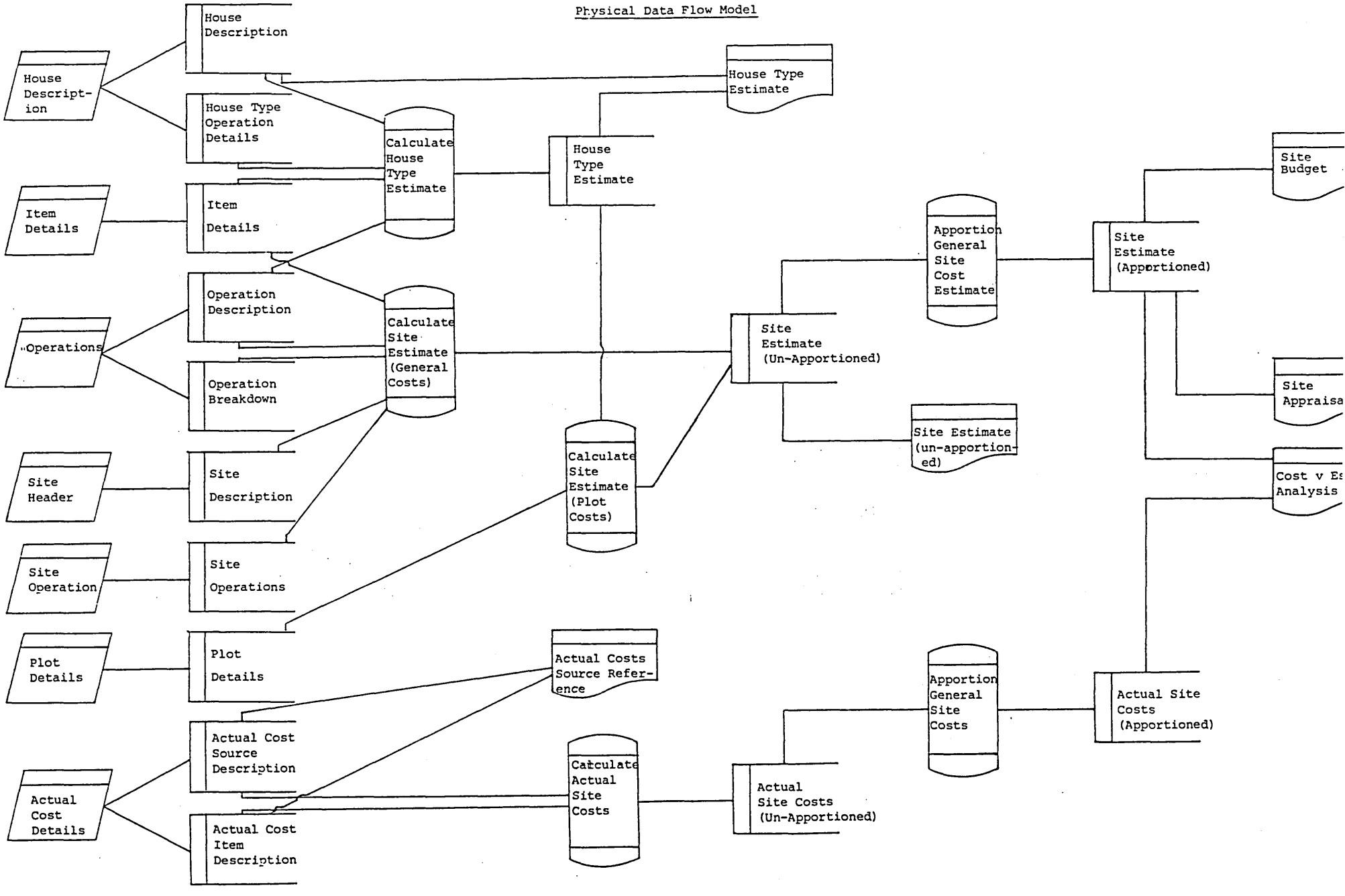
C3. SPEC-BUILDER MENUS

C4. SPEC-BUILDER SCREEN FORMATS

C5. SPEC-BUILDER REPORT FORMATS

Integrated Estimating and Cost Control system for Speculative House Builders

Physical Data Flow Model



SPEC-BUILDER OBJECT DICTIONARY ANALYSIS

ENTITY SPECIFICATION

ENTITY NO : 1

ENTITY NAME : ENTITY01

DESCRIPTION : HOUSE DESCRIPTION

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKE
1	HOUSE CODE	X	1	5	0	0		
6	HOUSE NAME	X	1	12	0	0		
7	HOUSE CLASS	X	1	3	0	0		
8	DESCRIPTION1	X	1	30	0	0		
9	DESCRIPTION2	X	1	30	0	0		
10	ABBREVIATION	X	1	2	0	0		
11	FLOOR AREA	9	0	7	4	2	N	Y
12	PLOT FRONT	9	0	7	4	2	N	Y
13	GARAGE CODE	X	1	2	0	0		
14	HEATING CODE	X	1	2	0	0		

ENTITY SPECIFICATION

ENTITY NO : 2

ENTITY NAME : ENTITY02

DESCRIPTION : HOUSE OPERATION AND ITEM BUILD UP

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	HOUSE CODE	X	1	5	0	0		
2	OP LEVEL	X	1	1	0	0		
3	OP CODE	X	1	6	0	0		
6	QUANTITY	9	0	10	6	3	N	Y
7	WASTAGE	9	0	5	2	2	N	Y
8	COST CODE	9	0	3	3	0	N	N

ENTITY SPECIFICATION

ENTITY NO : 3

ENTITY NAME : ENTITY03

DESCRIPTION : SITE DESCRIPTION

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE CODE	9	0	3	3	0	N	N
6	SITE NAME	X	1	30	0	0		
7	ADDRESS 1	X	1	30	0	0		
8	ADDRESS 2	X	1	30	0	0		
9	ADDRESS 3	X	1	30	0	0		
10	ADDRESS 4	X	1	30	0	0		
11	POST CODE	X	1	9	0	0		
12	TEL NO	X	1	12	0	0		
13	LABOUR RATE	9	0	5	1	3	N	Y
14	MATERIAL RTE	9	0	5	1	3	N	Y
15	SUB CON RATE	9	0	5	1	3	N	Y
16	PLANT RATE	9	0	5	1	3	N	Y
17	ON COST RATE	9	0	5	1	3	N	Y
18	SELLING RATE	9	0	5	1	3	N	Y
19	SUPERVIS RTE	9	0	5	1	3	N	Y
20	ESTIMATE RTE	9	0	5	1	3	N	Y

ENTITY SPECIFICATION

ENTITY NO : 4

ENTITY NAME : ENTITY04

DESCRIPTION : SITE PLOT DETAILS

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE CODE	9	0	3	3	0	N	N
2	PLOT NO	9	0	3	3	0	N	N
6	HOUSE CODE	X	1	5	0	0		
7	SALE VALUE	9	0	9	6	2	N	Y
8	COMMENCE DTE	X	6	8	0	0		
9	APP FACTOR	9	0	11	1	9	N	Y

ENTITY SPECIFICATION

ENTITY NO : 5

ENTITY NAME : ENTITY05

DESCRIPTION : SITE OPERATION DETAILS

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE CODE	9	0	3	3	0	N	N
2	OP LEVEL	X	1	1	0	0		
3	OP CODE	X	1	6	0	0		
6	QUANTITY	9	0	10	6	3	N	Y
7	WASTAGE	9	0	5	2	2	N	Y

ENTITY SPECIFICATION

ENTITY NO : 6

ENTITY NAME : ENTITY06

DESCRIPTION : OPERATION AND ITEM DETAILS

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	OP-ITEM TYPE	X	1	1	0	0		
2	OP-ITEM CODE	X	1	6	0	0		
6	NAME	X	1	8	0	0		
7	DESCRIPTION1	X	1	30	0	0		
8	DESCRIPTION2	X	0	30	0	0		
9	UNIT	X	1	8	0	0		
10	COST CODE	9	0	3	3	0	N	N
11	ESTIMATED	X	1	1	0	0		
12	VALUE	9	0	10	6	3	N	Y
13	ITEM TYPE	X	3	3	0	0		
15	CONV FACTOR	9	0	9	4	4	N	Y
16	DATE UPDATED	X	6	8	0	0		
17	APPORTIONED	X	1	1	0	0		

ENTITY SPECIFICATION

ENTITY NO : 7

ENTITY NAME : ENTITY07

DESCRIPTION : OPERATION BUILD UP

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	OP LEVEL	X	1	1	0	0		
2	OP CODE	X	1	6	0	0		
3	SUB OP LEVEL	X	1	1	0	0		
4	SUB OP CODE	X	1	6	0	0		
6	QUANTITY	9	0	10	6	3	N	Y
7	WASTAGE	9	0	5	2	2	N	Y

ENTITY SPECIFICATION

ENTITY NO : 8

ENTITY NAME : ENTITY08

DESCRIPTION : INVOICE & PROVISIONAL COST HEADER DETAIL

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SOURCE REF	X	1	10	3	0	N	N
6	PROV OR INV	X	1	1	0	0		
7	SUPPLIER	X	1	30	6	3	N	Y
8	DATE	X	0	8	0	0		

ENTITY SPECIFICATION

ENTITY NO : 9

ENTITY NAME : ENTITY09

DESCRIPTION : INVOICE & PROVISION COST ITEM DETAIL

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SOURCE REF	X	1	10	3	0	N	N
2	ITEM NO	9	0	3	3	0	N	N
6	SITE NO	9	0	3	3	0	N	N
7	PLOT NO	9	0	3	3	0	N	N
8	COST CODE	9	6	3	3	0	N	N
9	DESCRIPTION	X	1	20	0	0		
10	COST CAT	X	3	3	0	0		
11	VALUE	9	0	10	6	2	Y	Y

ENTITY SPECIFICATION

ENTITY NO : 10

ENTITY NAME : ENTITY10

DESCRIPTION : SITE ESTIMATE

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE CODE	9	0	3	3	0	N	N
2	PLOT NO	9	0	3	3	0	N	N
3	COST CODE	9	0	3	3	0	N	N
6	MATERIAL	9	0	9	6	2	N	Y
7	SUB CONTRACT	9	0	9	6	2	N	Y
8	LABOUR	9	0	9	6	2	N	Y
9	PLANT-HAUL	9	0	9	6	2	N	Y
10	ON COSTS	9	0	9	6	2	N	Y
11	SELLING	9	0	9	6	2	N	Y
12	SUPERVISION	9	0	9	6	2	N	Y
13	ESTIMATE	9	0	9	6	2	N	Y
14	TOTAL	9	0	10	7	2	N	N

ENTITY SPECIFICATION

ENTITY NO : 11

ENTITY NAME : ENTITY11

DESCRIPTION : HOUSE ESTIMATE

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	HOUSE CODE	X	1	5	0	0		
2	COST CODE	9	0	3	3	0	N	N
6	MATERIAL	9	0	9	6	2	N	N
7	SUB CONTRACT	9	0	9	6	2	N	N
8	LABOUR	9	0	9	6	2	N	N
9	PLANT HAUL	9	0	9	6	2	N	N
10	ON COSTS	9	0	9	6	2	N	N
11	SELLING	9	0	9	6	2	N	N
12	SUPERVISION	9	0	9	6	2	N	N
13	ESTIMATE	9	0	9	6	2	N	N
14	TOTAL	9	0	9	6	2	N	N

ENTITY SPECIFICATION

ENTITY NO : 12

ENTITY NAME : ENTITY12

DESCRIPTION : SITE ESTIMATE BY PLOT

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE CODE	9	0	3	3	0	N	N
2	PLOT NO	9	0	3	3	0	N	N
3	COST CODE	9	0	3	3	0	N	N
6	MATERIAL	9	0	9	6	2	N	Y
7	SUB CONTRACT	9	0	9	6	2	N	Y
8	LABOUR	9	0	9	6	2	N	Y
9	PLANT-HAUL	9	0	9	6	2	N	Y
10	ON COSTS	9	0	9	6	2	N	Y
11	SELLING	9	0	9	6	2	N	Y
12	SUPERVISION	9	0	9	6	2	N	Y
13	ESTIMATE	9	0	9	6	2	N	Y
14	TOTAL	9	0	10	7	2	N	Y

ENTITY SPECIFICATION

ENTITY NO : 14

ENTITY NAME : ENTITY14

DESCRIPTION : COST DETAILS BY SITE (NOT APPORTIONED)

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE NO	9	0	3	3	0	N	N
2	PLOT NO	9	0	3	3	0	N	N
3	COST CODE	9	0	3	3	0	N	N
6	MAT COST	9	0	10	6	2	Y	Y
7	SUB COST	9	0	10	6	2	Y	Y
8	LAB COST	9	0	10	6	2	Y	Y
9	PLA COST	9	0	10	6	2	Y	Y
10	ONC COST	9	0	10	6	2	Y	Y
11	SEL COST	9	0	10	6	2	Y	Y
12	SUP COST	9	0	10	6	2	Y	Y
13	MAT PROV	9	0	10	6	2	Y	Y
14	SUB PROV	9	0	10	6	2	Y	Y
15	LAB PROV	9	0	10	6	2	Y	Y
16	PLA PROV	9	0	10	6	2	Y	Y
17	ONC PROV	9	0	10	6	2	Y	Y
18	SEL PROV	9	0	10	6	2	Y	Y
19	SUP PROV	9	0	10	6	2	Y	Y
20	TOTAL	9	0	11	7	2	Y	N

ENTITY SPECIFICATION

ENTITY NO : 15

ENTITY NAME : ENTITY15

DESCRIPTION : COST DETAILS BY SITE & PLOT (APPORTIONED

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	SITE NO	9	0	3	3	0	N	N
2	PLOT NO	9	0	3	3	0	N	N
3	COST CODE	9	0	3	3	0	N	N
6	MAT COST	9	0	10	6	2	Y	Y
7	SUB COST	9	0	10	6	2	Y	Y
8	LAB COST	9	0	10	6	2	Y	Y
9	PLA COST	9	0	10	6	2	Y	Y
10	ONC COST	9	0	10	6	2	Y	Y
11	SEL COST	9	0	10	6	2	Y	Y
12	SUP COST	9	0	10	6	2	Y	Y
13	MAT PROV	9	0	10	6	2	Y	Y
14	SUB PROV	9	0	10	6	2	Y	Y
15	LAB PROV	9	0	10	6	2	Y	Y
16	PLA PROV	9	0	10	6	2	Y	Y
17	ONC PROV	9	0	10	6	2	Y	Y
18	SEL PROV	9	0	10	6	2	Y	Y
19	SUP PROV	9	0	10	6	2	Y	Y
20	TOTAL	9	0	11	7	2	Y	Y

ENTITY SPECIFICATION

ENTITY NO : 20

ENTITY NAME : ENTITY20

DESCRIPTION : COST CATAGORY ANALYSIS

ATTRIBUTE NUMBER	SHORT NAME	TYPE	MIN LENGTH	MAX LENGTH	INT PLACES	DEC PLACES	SIGNED	PACKED
1	COST CODE	9	0	3	3	0	N	N
6	COST NAME	X	1	12	0	0		

```
COST CONTROL & ESTIMATING SYSTEM      ACTION : SELECT
-------------------------------------------------------------------------------

OPTION                                  CODE

DATA BASE MAINTENANCE.....              DAT
SYSTEM PROCESSES..........              PRO
DATA BASE REPORTS.........              REP

ENTER OPTION, OR 'X' TO EXIT    <    >

-------------------------------------------------------------------------------
```

COST CONTROL & ESTIMATE REPORTS ACTION : SELECT

OPTION	CODE
HOUSE ESTIMATE ANALYSIS.....	HSE
SITE ESTIMATE :- DIRECT COST..	DIR
SITE ESTIMATE :- BUDGET.....	BUD
COST ANALYSIS :- BY SITE.....	CST
COST V ESTIMATE COMPARISON....	COM

ENTER OPTION, OR 'X' TO EXIT < >

SYSTEM PROCESS & CALCULATION ACTION : SELECT

OPTION	CODE
HOUSE TYPE ESTIMATE.....	HSE
SITE ESTIMATE :- DIRECT COST..	DIR
SITE ESTIMATE :- PLOT COST....	PLO
SITE ESTIMATE APPORTIONMENT...	EAP
SITE COST CALCULATION.....	CST
SITE COST APPORTIONMENT.....	CAP

ENTER OPTION, OR 'X' TO EXIT < >

DATA BASE MAINTENANCE ACTION : SELECT

OPTION	CODE
HOUSE DETAIL MAINTENANCE.....	HSE
OPERATION DETAIL MAINTENANCE..	OPE
ITEM DETAIL MAINTENANCE.....	ITM
SITE HEADER MAINTENANCE.....	SHD
SITE OPERATION MAINTENANCE....	SOP
PLOT DETAIL MAINTENANCE.....	PLO
COST DETAIL MAINTENANCE.....	CST
COST CATAGORY MAINTENANCE.....	CAT

ENTER OPTION, OR 'X' TO EXIT < >

SPEC-BUILDER SCREEN FORMATS

Format print for screen number 1
****

HOUSE DETAIL MAINTENANCE

```
HOUSE CODE : XXXXX      NAME : XXXXXXXXXXXX    ABBREVIATION : XX

HOUSE CLASS : XXX    DESCRIPTION : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

FLOOR AREA : 9999.99     PLOT FRONTAGE : 9999.99  GARAGE : XX  HEATING : XX

OPERATION BUILD UP DETAILS :-
LEVEL     CODE      DESCRIPTION                              QUANTITY   WASTAGE

  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
  X       XXXXXX    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX          999999.999   99.99
```

Screen name :   HSE-MNT

Screen description : HASSALL :- HOUSE DETAIL MAINTENANCE

Schema number :   1

Screen documentation (field analysis) for screen number 1

****

Header Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY01	HOUSE CODE	I	O
2	N	ENTITY01	HOUSE NAME	I	M
3	N	ENTITY01	ABBREVIATION	I	M
4	N	ENTITY01	HOUSE CLASS	I	M
5	N	ENTITY01	DESCRIPTION1	I	M
6	N	ENTITY01	DESCRIPTION2	I	O
7	N	ENTITY01	FLOOR AREA	I	M
8	N	ENTITY01	PLOT FRONT	I	M
9	N	ENTITY01	GARAGE CODE	I	O
10	N	ENTITY01	HEATING CODE	I	O

Scrolled Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
11	Y	ENTITY02	OP LEVEL	I	O
12	Y	ENTITY02	OP CODE	I	M
13	N	ENTITY06	DESCRIPTION1	O	O
14	N	ENTITY02	QUANTITY	I	M
15	N	ENTITY02	WASTAGE	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 2
****

ITEM DETAIL MAINTENANCE

CODE : XXXXXX

COST TYPE : XXX

DESCRIPTION : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

UNIT : XXXXXXXX

PRICE : 999999.999

APPORTIONED : X

CONV FACTOR : 9999.9999

Screen name : ITEM-MNT

Screen description : HASSALL :- ITEM DETAIL MAINTENANCE

Schema number : 1

Screen documentation (field analysis) for screen number 2

****

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY06	OP-ITEM CODE	I	O
2	N	ENTITY06	ITEM TYPE	I	M
3	N	ENTITY06	DESCRIPTION1	I	M
4	N	ENTITY06	DESCRIPTION2	I	O
5	N	ENTITY06	UNIT	I	M
6	N	ENTITY06	VALUE	I	O
7	N	ENTITY06	APPORTIONED	I	M
8	N	ENTITY06	CONV FACTOR	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 3
****

```
                          OPERATION MAINTENANCE
------------------------------------------------------------------------------

   LEVEL : X    CODE : XXXXXX    DESCRIPTION : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                                               XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

   UNIT : XXXXXXXX    COST CODE : 999    CATAGORY :   XXXXXXXXXXXX

           ESTIMATE (Y/N) : X    ESIMATED VALUE :   999999.999

   LEVEL  CODE     DESCRIPTION                       QUANTITY  WASTAGE%

     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
     X    XXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  999999.999     99.99
------------------------------------------------------------------------------
```

Screen name : OP-MNT

Screen description : HASSALL :- OPERATION MAINTENANCE

Schema number : 1

Screen documentation (field analysis) for screen number 3

****

Header Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY06	OP-ITEM TYPE	I	O
2	Y	ENTITY06	OP-ITEM CODE	I	M
3	N	ENTITY06	DESCRIPTION1	I	M
4	N	ENTITY06	DESCRIPTION2	I	O
5	N	ENTITY06	UNIT	I	M
6	N	ENTITY06	COST CODE	I	O
7	N	ENTITY20	COST NAME	O	O
8	N	ENTITY06	ESTIMATED	I	M
9	N	ENTITY06	VALUE	I	O

Scrolled Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
10	Y	ENTITY07	SUB OP LEVEL	I	O
11	Y	ENTITY07	SUB OP CODE	I	M
12	N	ENTITY06	DESCRIPTION1	O	O
13	N	ENTITY07	QUANTITY	I	M
14	N	ENTITY07	WASTAGE	I	O

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 4
****

SITE HEADER

SITE CODE : 999
NAME : XXXXXXXXXXXXXXXXXXXXXXXXXXXX
ADDRESS : XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
POST CODE : XXXXXXXXX
TEL NO. : XXXXXXXXXXX

SITE WEIGHTINGS :-

LABOUR	MATERIAL	SUB CON	PLANT	ON COST	SELLING	SUPERVIS	ESTIMATE
9.999	9.999	9.999	9.999	9.999	9.999	9.999	9.999

Screen name : SITE-HDR

Screen description : HASSALL :- SITE HEADER DESCRIPTION

Schema number : 1

Screen documentation (field analysis) for screen number 4

****

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY03	SITE CODE	I	O
2	N	ENTITY03	SITE NAME	I	M
3	N	ENTITY03	ADDRESS 1	I	M
4	N	ENTITY03	ADDRESS 2	I	O
5	N	ENTITY03	ADDRESS 3	I	O
6	N	ENTITY03	ADDRESS 4	I	O
7	N	ENTITY03	POST CODE	I	O
8	N	ENTITY03	TEL NO	I	O
9	N	ENTITY03	LABOUR RATE	I	M
10	N	ENTITY03	MATERIAL RTE	I	M
11	N	ENTITY03	SUB CON RATE	I	M
12	N	ENTITY03	PLANT RATE	I	M
13	N	ENTITY03	ON COST RATE	I	M
14	N	ENTITY03	SELLING RATE	I	M
15	N	ENTITY03	SUPERVIS RTE	I	M
16	N	ENTITY03	ESTIMATE RTE	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 5
****

PLOT MAINTENANCE

SITE CODE : 999 SITE NAME : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

PLOT NO	HOUSE CODE	DESCRIPTION	SALE PRICE
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99
999	XXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.99

Screen name : S-HOUSE

Screen description : HASSALL :- SITE DESCRIPTION; PLOT DETAIL

Schema number : 1

Screen documentation (field analysis) for screen number 5
****

Header Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY03	SITE CODE	I	O
2	N	ENTITY03	SITE NAME	I	M

Scrolled Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
3	Y	ENTITY04	PLOT NO	I	O
4	N	ENTITY04	HOUSE CODE	I	M
5	N	ENTITY01	DESCRIPTION1	O	O
6	N	ENTITY04	SALE VALUE	I	O

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 6
****

SITE OPERATIONS.

SITE CODE : 999   SITE NAME : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

LEVEL	CODE	DESCRIPTION	QUANTITY	WASTAGE
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99
X	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	999999.999	99.99

Screen name :   S-OPER

Screen description : HASSALL :- SITE OPERATION MAINTENANCE

Schema number :   1

Screen documentation (field analysis) for screen number 6

****

Header Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY03	SITE CODE	I	O
2	N	ENTITY03	SITE NAME	I	M

Scrolled Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
3	Y	ENTITY05	OP LEVEL	I	O
4	Y	ENTITY05	OP CODE	I	M
5	N	ENTITY06	DESCRIPTION1	O	O
6	N	ENTITY05	QUANTITY	I	M
7	N	ENTITY05	WASTAGE	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 7
****

COST MAINTENANCE

SOURCE REF : XXXXXXXXXX PROVISION OR INVOICE : X

SUPPLIER : XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

DATE : XXXXXXXX

ITEM NO	SITE NO	PLOT NO	COST CODE	DESCRIPTION	COST CAT	VALUE
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99
999	999	999	999	XXXXXXXXXXXXXXXXXXXX	XXX	-999999.99

Screen name : COST-MNT

Screen description : COST (INVOICED & PROVISIONAL) MAITENANCE

Schema number : 2

Screen documentation (field analysis) for screen number 7

****

Header Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY08	SOURCE REF	I	O
2	N	ENTITY08	PROV OR INV	I	M
3	N	ENTITY08	SUPPLIER	I	M
4	N	ENTITY08	DATE	I	M

Scrolled Area

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
5	Y	ENTITY09	ITEM NO	I	O
6	N	ENTITY09	SITE NO	I	M
7	N	ENTITY09	PLOT NO	I	M
8	N	ENTITY09	COST CODE	I	M
9	N	ENTITY09	DESCRIPTION	I	M
10	N	ENTITY09	COST CAT	I	M
11	N	ENTITY09	VALUE	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

Format print for screen number 20
****

COST CATAGORY MAINTENANCE

COST CODE : 999

CATAGORY : XXXXXXXXXXXX

Screen name : CC-MNT

Screen description : COST CATAGORY MAINTENANCE

Schema number : 1

Screen documentation (field analysis) for screen number 20
****

Field No.	Key	Entity	Attribute	Input/Output	Mandatory/Optional
1	Y	ENTITY20	COST CODE	I	O
2	N	ENTITY20	COST NAME	I	M

(N.B. Field numbers are allocated by horizontally traversing the screen)

********** End of screen documentation **********

HOUSE ESTIMATE :- DIRECT COSTS

HOUSE CODE : SHE01 HOUSE NAME : SHEFFIELD

COST CODE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ON COSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	1850.00	0.00	745.00	239.00	0.00	0.00	0.00	0.00	2834.00
DRAINS	190.30	0.00	157.50	83.76	0.00	0.00	0.00	0.00	431.56
SUPER-STRUCT	6975.34	3100.30	3456.50	0.00	0.00	0.00	0.00	450.30	13982.44
EXTERNAL WKS	367.20	190.40	190.70	307.30	0.00	0.00	0.00	0.00	1055.60
ONCOSTS	0.00	24.25	0.00	0.00	114.00	0.00	0.00	0.00	138.25
HOUSE TOTAL	9382.84	3314.95	4549.70	630.06	114.00	0.00	0.00	450.30	18441.85

SITE ESTIMATE (DIRECT COSTS) :- TEST SITE

PLOT NO : 0 HOUSE CODE : HOUSE NAME :

COST CODE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ONCOSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
SAND & THSLE	356.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	356.30
SITE HAULAGE	0.00	0.00	0.00	90.00	0.00	0.00	0.00	0.00	90.00
SUPERVISION	0.00	0.00	0.00	0.00	0.00	0.00	770.00	0.00	770.00
WATCHMAN	0.00	0.00	670.00	0.00	0.00	0.00	0.00	0.00	670.00
SEWERS	0.00	2350.00	0.00	0.00	0.00	0.00	0.00	0.00	2350.00
ROADS	0.00	5600.00	0.00	0.00	0.00	0.00	0.00	0.00	5600.00
PLOT TOTAL	356.30	7950.00	670.00	90.00	0.00	0.00	770.00	0.00	9836.30

(general site costs)

SITE ESTIMATE (DIRECT COSTS) :- TEST SITE

PLOT NO : 1 HOUSE CODE : SHE01 HOUSE NAME : SHEFFIELD

COST CODE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ONCOSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	1850.00	0.00	745.00	239.00	0.00	0.00	0.00	0.00	2834.00
DRAINS	190.30	0.00	157.50	83.76	0.00	0.00	0.00	0.00	431.56
SUPER-STRUCT	6975.34	3100.30	3456.50	0.00	0.00	0.00	0.00	450.30	13982.44
EXTERNAL WKS	367.20	190.40	190.70	307.30	0.00	0.00	0.00	0.00	1055.60
ONCOSTS	0.00	24.25	0.00	0.00	114.00	0.00	0.00	0.00	138.25
PLOT TOTAL	9382.84	3314.95	4549.70	630.06	114.00	0.00	0.00	450.30	18441.85

SITE ESTIMATE (DIRECT COSTS) :- TEST SITE

PLOT NO : 2 HOUSE CODE : WAK01 HOUSE NAME : WAKEFIELD

COST CODE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ONCOSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	1942.50	0.00	797.15	262.90	0.00	0.00	0.00	0.00	3002.55
DRAINS	211.23	0.00	189.00	96.32	0.00	0.00	0.00	0.00	496.55
SUPER-STRUCT	11858.07	4030.39	4942.79	0.00	0.00	0.00	0.00	0.00	20831.25
EXTERNAL WKS	422.28	220.86	228.84	334.95	0.00	0.00	0.00	0.00	1206.93
ONCOSTS	0.00	0.00	0.00	0.00	262.20	0.00	0.00	0.00	262.20
PLOT TOTAL	14434.08	4251.25	6157.78	694.17	262.20	0.00	0.00	0.00	25799.48
SITE TOTAL	24173.22	15516.20	11377.48	1414.23	376.20	0.00	770.00	450.30	54077.63

SITE BUDGET (APPORTIONED) :- TEST SITE

PLOT NUMBER : 1 HOUSE CODE : SHE01 NAME : SHEFFIELD APPORTION FACTOR : 0.454545454

COST CENTRE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ON COSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	1850.00	0.00	745.00	239.00	0.00	0.00	0.00	0.00	2834.00
DRAINS	190.30	0.00	157.50	83.76	0.00	0.00	0.00	0.00	431.56
SUPER-STRUCT	6975.34	3100.30	3456.50	0.00	0.00	0.00	0.00	450.30	13982.44
EXTERNAL WKS	367.20	190.40	190.70	307.30	0.00	0.00	0.00	0.00	1055.60
ONCOSTS	0.00	24.25	0.00	0.00	114.00	0.00	0.00	0.00	138.25
APPORTIONED	161.95	3613.63	304.54	40.90	0.00	0.00	349.99	0.00	4471.01
PLOT TOTAL	9544.79	6928.58	4854.24	670.96	114.00	0.00	349.99	450.30	22912.86

SITE BUDGET (APPORTIONED) :- TEST SITE

PLOT NUMBER : 2 HOUSE CODE : WAK01 NAME : WAKEFIELD APPORTION FACTOR : 0.545454545

COST CENTRE	MATERIALS	SUB CON	LABOUR	PLNT & HLG	ON COSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	1942.50	0.00	797.15	262.90	0.00	0.00	0.00	0.00	3002.55
DRAINS	211.23	0.00	189.00	96.32	0.00	0.00	0.00	0.00	496.55
SUPER-STRUCT	11858.07	4030.39	4942.79	0.00	0.00	0.00	0.00	0.00	20831.25
EXTERNAL WKS	422.28	220.86	228.84	334.95	0.00	0.00	0.00	0.00	1206.93
ONCOSTS	0.00	0.00	0.00	0.00	262.20	0.00	0.00	0.00	262.20
APPORTIONED	194.34	4336.35	365.45	49.09	0.00	0.00	419.99	0.00	5365.22
PLOT TOTAL	14628.42	8587.60	6523.23	743.26	262.20	0.00	419.99	0.00	31164.70
SITE-TOTAL	24173.21	15516.18	11377.47	1414.22	376.20	0.00	769.98	450.30	54077.56

SITE COST ANALYSIS (APPORTIONED) :-TEST SITE

PLOT NUMBER : 1 HOUSE CODE : SHE01 APPORTION FACTOR : 0.454545454

COST CENTRE		MATERIALS	SUB CON	LABOUR	PLNT & HLG	ON COSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	COST	655.70	45.30	840.00	357.00	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		1898.00
	ESTIMATE	1850.00	0.00	745.00	239.00	0.00	0.00	0.00	0.00	2834.00
DRAINS	COST	274.90	0.00	149.00	82.67	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		506.57
	ESTIMATE	190.30	0.00	157.50	83.76	0.00	0.00	0.00	0.00	431.56
SUPER-STRUCT	COST	6845.78	3090.00	3267.00	0.00	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		13202.78
	ESTIMATE	6975.34	3100.30	3456.50	0.00	0.00	0.00	0.00	450.30	13982.44
EXTERNAL WKS	COST	428.90	187.80	197.00	310.80	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		1124.50
	ESTIMATE	367.20	190.40	190.70	307.30	0.00	0.00	0.00	0.00	1055.60
ONCOSTS	COST	0.00	0.00	0.00	0.00	127.50	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		127.50
	ESTIMATE	0.00	24.25	0.00	0.00	114.00	0.00	0.00	0.00	138.25
APPORTIONED	COST	145.68	3730.67	327.49	22.72	0.00	0.00	375.45		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		4602.01
	ESTIMATE	161.95	3613.63	304.54	40.90	0.00	0.00	349.99	0.00	4471.01
PLOT TOTAL	COST	8350.96	7053.77	4780.49	773.19	127.50	0.00	375.45		21461.36
	ESTIMATE	9544.79	6928.58	4854.24	670.96	114.00	0.00	349.99	450.30	22912.86

SITE COST ANALYSIS (APPORTIONED) :-TEST SITE

PLOT NUMBER : 2 HOUSE CODE : WAK01 APPORTION FACTOR : 0.545454545

COST CENTRE		MATERIALS	SUB CON	LABOUR	PLNT & HLG	ON COSTS	SELLING	SUPERVISN	ESTIMATE	TOTAL
FOUNDATIONS	COST	583.50	53.45	925.00	390.50	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		1952.45
	ESTIMATE	1942.50	0.00	797.15	262.90	0.00	0.00	0.00	0.00	3002.55
DRAINS	COST	264.50	0.00	162.00	97.50	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		524.00
	ESTIMATE	211.23	0.00	189.00	96.32	0.00	0.00	0.00	0.00	496.55
SUPER-STRUCT	COST	11370.20	4001.70	4837.00	0.00	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		20208.90
	ESTIMATE	11858.07	4030.39	4942.79	0.00	0.00	0.00	0.00	0.00	20831.25
EXTERNAL WKS	COST	450.75	223.80	225.40	312.80	0.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		1212.75
	ESTIMATE	422.28	220.86	228.84	334.95	0.00	0.00	0.00	0.00	1206.93
ONCOSTS	COST	0.00	0.00	0.00	0.00	240.00	0.00	0.00		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		240.00
	ESTIMATE	0.00	0.00	0.00	0.00	262.20	0.00	0.00	0.00	262.20
APPORTIONED	COST	174.81	4476.80	392.99	27.27	0.00	0.00	450.54		
	PROVISION	0.00	0.00	0.00	0.00	0.00	0.00	0.00		5522.41
	ESTIMATE	194.34	4336.35	365.45	49.09	0.00	0.00	419.99	0.00	5365.22
PLOT TOTAL	COST	12843.76	8755.75	6542.39	828.07	240.00	0.00	450.54		29660.51
	ESTIMATE	14628.42	8587.60	6523.23	743.26	262.20	0.00	419.99	0.00	31164.70
SITE TOTAL	COST	21194.72	15809.52	11322.88	1601.26	367.50	0.00	825.99		51121.87
	ESTIMATE	24173.21	15516.18	11377.47	1414.22	376.20	0.00	769.98	450.30	54077.56

COST & PROVISIONAL COST ANALYSIS

SITE NO : 1 SITE NAME : TEST SITE

PLOT NO	COST CODE	SOURCE REFERENCE	ITEM NO	DESCRIPTION	COST CATAGORY	COST PROV	VALUE
0	100	INV003	1	SAND AND THISTLE	MAT	I	320.50
0	101	PLA001	1	JCB	PLA	I	50.00
0	101	PLA001	2	JCB OPERATOR	LAB	I	38.00
0	102	WKS2001	1	SITE FOREMAN	SUP	I	155.00
0	102	WKS2002	1	SITE FOREMAN	SUP	I	155.50
0	102	WKS2003	1	SITE FOREMAN	SUP	I	155.00
0	102	WKS2004	1	SITE FOREMAN	SUP	I	155.00
0	102	WKS2005	1	SITE FOREMAN	SUP	I	155.50
0	102	WKS2006	1	SITE FOREMAN BONUS	SUP	P	50.00
0	103	WKS1001	1	WATCHMAN	LAB	I	134.30
0	103	WKS1002	1	WATCHMAN	LAB	I	134.50
0	103	WKS1003	1	WATCHMAN	LAB	I	134.50
0	103	WKS1004	1	WATCHMAN	LAB	I	139.50
0	103	WKS1005	1	WATCHMAN	LAB	I	139.70
0	700	INV002	1	SEWER CONSTRUCTION	SUB	I	2557.50
0	800	INV001	1	ROAD CONSTRUCTION	SUB	I	5650.00
1	1	INV004	1	BRICKS (COMMONS)	MAT	I	350.00
1	1	INV004	2	READY MIX	MAT	I	260.00
1	1	INV004	3	AIR BRICKS	MAT	I	45.70
1	1	INV005	1	JCB & OPERATOR	PLA	I	357.00
2	1	INV004	4	BRICKS (COMMONS)	MAT	I	323.00
2	1	INV004	5	READY MIX	MAT	I	223.00
2	1	INV004	6	AIR BRICKS	MAT	I	37.50
2	1	INV005	2	JCB & OPERATOR	PLA	I	390.50

DB4GL REPORT GENERATOR SPECIFICATIONS.

APPENDIX D

REPORT GENERATION

INTRODUCTION

This report forms the specification for a report generator. It documents the ideas and concepts that would ideally be included in such a program, and outlines how it may be implemented. The report generator will form a part of the fourth generation programming package developed as part of the research project conducted within the building department.

The report generator must obviously complement and interface with the existing data dictionary and data base systems that form the basis of the fourth generation system.

The specification of the report generator has been divided into three sections;

i) generated report format and control

ii) report specification

iii) report generation

1. REPORT FORMAT AND CONTROL

The form and control of the reports it is intended to generate must obviously be clearly established before any attempt to generate them can be made.

The production of a report, in any form, consists of two stages;

i) the selection and sorting of the data to be reported,

ii) the formating and printing of the report.

These two stages are treated separately in the analysis and production of the generated reports.

1.1. REPORT FORMAT

The generated report is divided into three physical areas:

i) the header area,

ii) the data area,

iii) the footing area.

1.1.1. HEADER AREA

The header area contains constant and variable fields, and is printed at the top of each page of the report. At the simplest level the header area may contain nothing more than the report title. It may also contain variable data obtained from the data base which helps describe the data area below,e.g. in an order report, the data area may include details of individual order lines, and the header area contain details of the customer who placed the order.

1.1.2. DATA AREA

The data area contains variable data only. It consists of repeating groups of lines of data fields. Usually there is only 1 line per group, more are necessary only when the paper is not wide enough to include all the required data.

The lines, or groups of lines, are directly proportional, ie 1:1, to occurrences of the main source entity. The groups are repeated until a break condition occurs (see breaks below).

1.1.3. FOOTING AREA

The footing area also contains constant and variable fields, it is printed at the bottom of the page when a break condition has occurred. An additional footing area is produced at the end of the report.

The footing may include totals of numeric data attributes in the data area between the last breaks. The end of report footing contains totals for the whole report.

EXAMPLE REPORT FORMAT

```
                                                    PAGE : 999

H                        ORDER REPORT
E
A     CUSTOMER NO : A0001      CUSTOMER NAME : ACME LTD
D     ORDER NO    : 000001     ORDER DATE    : 01/01/84
E
R     STOCK    ITEM    DATE REQD    PRICE    QUANTITY     VALUE

D
A     A123     WIDGET  01/03/84     1.34     1000.00     1340.00
T     B234     WOTSIT  03/03/84     2.13     2000.00     4260.00
A

F
O
O     TOTAL VALUE                                        5600.00
T
```

1.1.4. BREAKS

A control break is an event which occurs during the printing of the report. It causes a break in the printing of the data area, and the footing area to be printed, followed by a page throw, and the heading area, before the printing of the data area is resumed (if it can be).

There are two types of breaks, those enforced by end of file conditions, and those which can be imposed by the user. The user implements breaks by specifying a particular data attribute, such that when the value of that attribute changes, a break event occurs. In the above example, the user may specify customer or order number as the break attribute.

1.2. REPORT CONTROL

Before any data can be printed, it must be selected from the data base and sorted into the required order.

1.2.1. SELECTION

The report is built from data held in a source entity type and other 'linked' entity types. Entities are selected from the source entity type. The selection is achieved by comparing attribute values from both the source and the linked entity types, with control values. The control values may be inserted at run time, or be dependent on the value of other attribute values in the data base. The selected entities are recorded by placing the source entity keys into a temporary list file.

The selection criteria must allow the use of 'and' and 'or' logical operators, and the conditions; 'equal', 'not equal', 'greater than', and 'less than'.

1.2.2. SORTING

Once the required entities have been selected, the keys stored in the temporary file are sorted. This is done according to the users specification. Any attribute in the source entity type, or a linked entity type can be specified, in ascending or descending

order. Any number of sort keys can be specified in order of priority.

Once the entity keys have been selected and sorted, the temporary file of keys is passed to the next stage, the report output.

2. REPORT SPECIFICATION

In order to generate a report, a precise unambiguous specification of the report is required, which will form part of the system data dictionary. The specification of reports can therefore be expressed in terms of data dictionary entity types.

The action of generating a report is again considered as two separate stages;

i) the selection and sorting of the required entity details.

ii) the specification of entity selection and sorting criteria.

iii) the specification of the report format.

2.1. REPORT HEADER

The report header information can be stored in a single entity type. There is only one entity type occurrence per report. It contains information relevant to both stages of this report production.

REPORT NUMBER	KEY FIELD
REPORT NAME	
REPORT DESCRIPTION	
SCHEMA NUMBER	
MAIN SOURCE ENTITY	
LINK ENTITY 1	

LINK ENTITY 2

LINK ENTITY 3

LINK ENTITY 4

BREAK ENTITY NO

BREAK ATTRIBUTE NO

DATA AREA START LINE

FOOTING AREA START LINE

FORM WIDTH (80 OR 120 COLUMNS)

PAGE NUMBERS REQUIRED

2.2. SELECTION AND SORTING SPECIFICATION

There are separate entity types for recording selection and sorting criteria. There may be many occurrences of each entity type for a single report. Both are keyed by report number and a sequence number which indicates the order of the entities.

2.2.1. SELECTION

REPORT NUMBER	KEY FIELD
SELECTION SEQUENCE NUMBER	KEY FIELD
BRACKET OPEN	
CURRENT ENTITY NUMBER	
CURRENT ATTRIBUTE NUMBER	
RELATIONSHIP	
CONTROL ENTITY NUMBER	
CONTROL ATTRIBUTE NUMBER	
RUN-TIME PARAMETER NAME	
BRACKET CLOSE	
AND/OR/STOP	

2.2.2. SORTING

REPORT NUMBER	KEY FIELD
SORT SEQUENCE NUMBER	KEY FIELD
SORT ENTITY NUMBER	
SORT ATTRIBUTE NUMBER	

2.3. REPORT FORMAT

The report format specification is held in two entity types; one each for text field details and variable field details (attribute values). There is an entity occurrence for each field in the report structure.

2.3.1. TEXT FIELDS

REPORT NUMBER	KEY FIELD
AREA (HEADER OR FOOTING)	KEY FIELD
LINE NUMBER	KEY FIELD
COLUMN NUMBER	KEY FIELD
TEXT VALUE	

2.3.2. VARIABLE FIELDS (ATTRIBUTES)

REPORT NUMBER	KEY FIELD
AREA (HEADING, DATA OR FOOTING)	KEY FIELD
LINE NUMBER	KEY FIELD
COLUMN NUMBER	KEY FIELD
ENTITY NUMBER	
ATTRIBUTE NUMBER	
TOTAL? (IF NUMERIC IN FOOTING)	

3. REPORT GENERATION

The generation of report programs is divided into the same two sections as report specification and report production, ie;

i) the selection and sorting of the required entity details

ii) the reporting of the selected and sorted information.

Two separate (object) programs are generated to produce each report. The first object program selects and sorts the required entities according to the specifications in the data dictionary. The sorted and selected source entity keys are stored in a temporary file which is passed to the second object program. This program produces the hard copy report. The two object programs are linked at run time, the second following imediately after the first.

There are also two separate report generating programs, one for each of the object program types. Both generating programs function by encrypting the specifications held in the data dictionary into skeleton programs.

3.1. SORT AND SELECTION

The sort and selection object program initially selects the required entities, stores their keys and then sorts them. This sequence reduces the amount of sorting (and therefore time) required.

The select procedure searches the main source entity sequentially, from start to finish, locating entity occurrences that meet the selection criteria. The attributes which can be tested are not limited to the source entity, but also include attributes contained in any of the entities that are contained in the current scheme, and are specified in the report specification header, eg. in the order report

example, it would be possible to report only on those customers in a certain area. The main source entity is the order line entity, but the customer area code is held within the customer entity. So long as the customer entity has been specified in the specification header, and it is contained within the specified schema, then this is allowed.

There are two sources of control value against which the attributes are compared during selection. The first are control values that are contained within the data base, these are attribute values contained in entity types which must part of the program sub-schema, ie within the schema, and the report specification header. The second type of control values are those which are input, or can be changed, at run time. These values are prompted for every time the generated report program is run.

As entities are selected their keys are stored in a sequential file. The record format of this file is defined by the first generation program. When the file is opened it is opened as type 'output' only, thus any previous occurrence of the file is over-written. Once the selection stage has been completed, the file is closed and re-opened as 'input-output' for sorting.

The sorting algorithm 'bubble sorts' the keys. The keys are read from the temporary file and passed, via the schema, to the source entity enabling links to the other entities in the sub-schema. The comparisons are made between the current schema attribute values, and a set of previous values held within the sorting program. The key to the lower set is stored and, if necessary, the program values exchanged for those in the schema. The next key is then read from the

temporary file.

The sorting algorithm passes repeatedly through the temporary file swapping keys around until the keys are in the required order. The temporary file is then closed, and the program halts. The report output program is then called.

3.2. REPORT OUTPUT

The report output object program is generated from the report format specification contained in the data dictionary.

The object program uses the temporary key file output by the select and sort object program to access the relevant source entities in sequence. The keys are copied from the temporary file into the main source entity via the sub-schema which links the other entities. The attribute values can then be accessed for inclusion in the report.