SYSTEMS DESIGN

INTRODUCTION

The logical design which has been developed to date has been free from any constraints imposed by hardware or software. When the analyst gets down to the task of designing the actual processing and computation of the data in the system, he will face numerous problems. Many of these will be due to constraints imposed by the user or to limitations in the hardware and software that makes up the installation. As the designer moves from the logical to the physical, so his freedom will be reduced. His task, therefore, is to obtain the best logical system requirements that fit with whatever physical constraints are imposed by the environment in which he is working. The objectives of the design phase is to come up with a physical design which:

* performs the required functions to the user's satisfaction
* is cost-effective
* is the best design for the purpose

the structured approach recommended does not provide all the answers, nor is it a magic wand to wave to arrive at the perfect system or convert the logical design to the one and only correct physical design. However, it does provide a disciplined approach to help the designer in his work and ensure he receives all the information to design the system.

A GOOD DESIGN

What makes a good system? A 'good design' is one which will perform the user's required functions in a way which gives the maximum benefit, and is cost-effective. Therefore to achieve a good design of a system, the analyst should ensure that he meets the objectives that will provide this. Thus, the objective to achieve are Flexibility, Control, and Performance.

FLEXIBILITY

The flexibility of a system can be measured by it ease of maintenance and how future changes can be made. If the system designed is to be flexible then the tracking down of errors found should prove to be easier than in a less flexible system. A system with a short life span is likely to be a system that proved to be inflexible, and therefor not able to meet the user's future requirements or not easily maintained.

CONTROL

The aspects of Control relates to a number of areas of the system from the control of the inputs to and the outputs from the system and includes the various controls that cover the security aspects. The quality of input data is likely to be subject to stringent controls as to the level of degree of quality for the output information or reports. Systems that deal with monetary representation, eg cash or invoices are likely to have more control on the outputs than other systems. From the viewpoint of security and availability of the systems the designer needs to consider controls for backing up the data, for providing audit trails and also perhaps contingency plans. Other aspects may be the control of personnel having access to the system or the level of access allowed.

PERFORMANCE

How quickly will the system be able to perform the required functions? Performance, or efficiency is usually measured in terms of -

* throughput — number of transactions processed in a certain time;
* run time — the time taken to run a particular program or suite of programs in a batch system;
* response time — in an interactive system, this is the time taken for the first character of response to appear at a terminal, measured from the time when the 'transmit' key is pressed.

Various ways of optimising system performance can be considered, these include -

* minimising the number of intermediate files within the system;
* minimising the number of passes of each file;
* keeping the number of disc accesses to a minimum;
* examining the way in which the system swaps programs in and out, and minimising this;
* looking at the code used in programs and streamlining this.

INPUTS TO THE DESIGN PHASE

The starting point for the designer is the results of the analysis phase. The design cannot start until this phase is completed and the specification of the users new requirements is available.

This specification forms the basis for the design. If this document is a lengthy narrative document it is likely to need considerable interpretation by the designer which may lead to the system not reflecting what the user really wants. Using structured analysis techniques, the analysts has available to him a number of documents ready to go into the 'Structured Specification'.

The main documents on which the physical design is based will be two types: those which analyse the functions of the business and those related to the entities relevant to the business.

Documents relating to functional analysis include:

* *CONTEXT DIAGRAM* which is a Data Flow Diagram giving an overview of the whole system.
* *A LEVELLED SET OF DATA FLOW DIAGRAMS*, which describes functions at a high level.
* *PROCEDURE SPECIFICATIONS*, (Mini Specs) which are used to specify the processes contained in the lower level DFD's.
* *DATA DICTIONARY* which is built up progressively during the analysis phase and details all the data and processes that the analyst is interested in.

* *DATA ACCESS DIAGRAM*, this and the process-file structure chart describes the data stores and the processes that access, identifying method and frequency.

The results from the Entity Analysis is also available and would include:

* *ENTITY MODEL* which defines the 'things' of most importance to the user's business.

* *ENTITY LIFE CYCLES* gives a picture of the entity over a period of time.

* *LISTS OF THE ATTRIBUTES OF ENTITIES*, these are the data items which will be held about each entity and will be held in the Data Dictionary.

APPLICATION & FILE DESIGN

The physical design of the new system can be broken down into two main areas, application design, considering the system interfaces and the design of the programs, and file design, considering the type, size, access, etc of the files themselves.

Systems Interfaces

The system interfaces, the man-machine or machine-man boundary of the system can be identified from DFD's and the Data Dictionary. It is at these interfaces that dialogue designs, clerical support procedures and forms design which may be needed at the interface.

Program Design

To avoid high maintenance costs the design of the programs or program suites should be carefully considered. There are a number of ways of designing programs and their relationships to each other which can make maintenance simpler, and ultimately cheaper.

Top-down Design

The program should consist of a hierarchy of modules, each with a single entry and a single exit. Each module should provide only one function and always pass control back to the 'Controlling' module.

Small Module

Each module should be kept as small as possible. Small modules are easier to maintain and if errors are found they are easier to detect in a small module.

Black-box Approach

This implies that a module can be looked at purely in terms of its inputs and outputs, no knowledge of the code or how it works is necessary.

Isolation of Function

If a module has only one function and that function is contained completely within that module, then the modification of any aspect of that function need only affect one module. It is easy to identify where to make the change and should be simpler to test and implement.

FILE DESIGN

File design is a crucial factor in the performance of a system. The information needed to design the files can be found in DFD's, Data Dictionary, Normalised Data Structures, etc. Often, now-a-days, in preference to conventional files, databases are used. These are often built using 4GL's, their structure being hidden to the user. Notwithstanding this the designer will still need to specify the data items required, their relationships and anticipated volumes.

The system design process begins with a functional specification of what the new system is to achieve and ends with a detailed system specification from which the programs can be prepared.

In summary a 'Good System' should have the following -

Portable
Efficient
Robust
Flexible
Economic
Control
Tested

Manageable
Integrated
Security

This combination of attributes will ensure that the prime objectives of good systems design of Flexibility, Control and Performance are achieved.