HEWLETT-PACKARD GENERAL SYSTEMS USERS GROUP

FEBRUARY, 1980 MEETING

RESOURCE OPTIMIZATION SERIES

A MATRIX STRUCTURED DOCUMENTATION SCHEME

Brian C. Mullen

I.S.P. Information Systems Planning Corp.
4180 Lions Avenue
North Vancouver, BC
V7R 3S4
(604) 980-9101

Most people solve the difficult problem of systems documentation by avoiding it.

But if we do adopt a top-down structured approach to documentation, we face the problem of how to organize and manage the volume of graphic and narrative information that is produced?

> I have a rule of thumb, the documentation manuals from $25,000 worth of systems work occupies approximately 1 foot of shelf space. If not properly organized it is seldom used and never kept current.

There are two techniques we found useful for organizing documentation.

> The first, a matrix form of organization provides a well defined structure for filing information during all phases of the project, eliminating the redundancy and duplications that frustrate reasonable efforts to keep the documentation current.
>
> The second, A-Z workbooks provides a flexible structure for organizing manually produced documentation.

Both techniques aim at breaking the documentation problem down into more managable pieces. The magnitude of the documentation problem is underestimated.

> Systems documentation is a complex information system involving many components, relationships and attributes.
>
> In a large project environment, the contents of this information changes as the project progresses.

Information not captured concurrently with design is lost.

Conventional documentation schemes are program oriented with documentation such as record layouts duplicated many times.

This is analogous to non-database computer systems where duplicated data is difficult to update consistently.

The form of our documentation scheme reflects the structure of a typical system.

The system can be viewed in terms of layers. Working from the external boundary of the system with its environment these layers are:

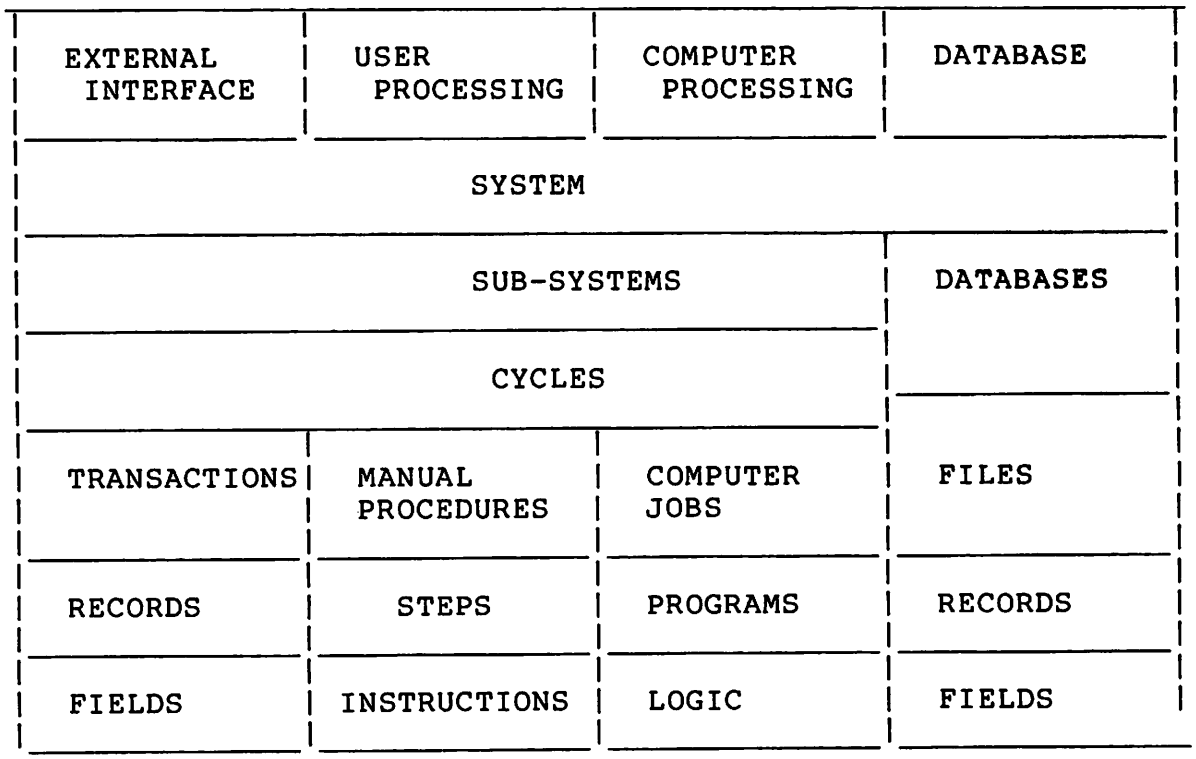
The external interfaces (primarily system transactions)
Manual user processes (both input and output)
Computer processes
Stored data(bases).

Vertically a system can be viewed at several levels of detail,
System
Sub-systems
Cycles
Procedures/Jobs
Programs, modules
Detailed logic

The corresponding levels of detail for the data components are

Logical databases
Physical databases
Transactions/Files
Records
Segments
Fields
Code values

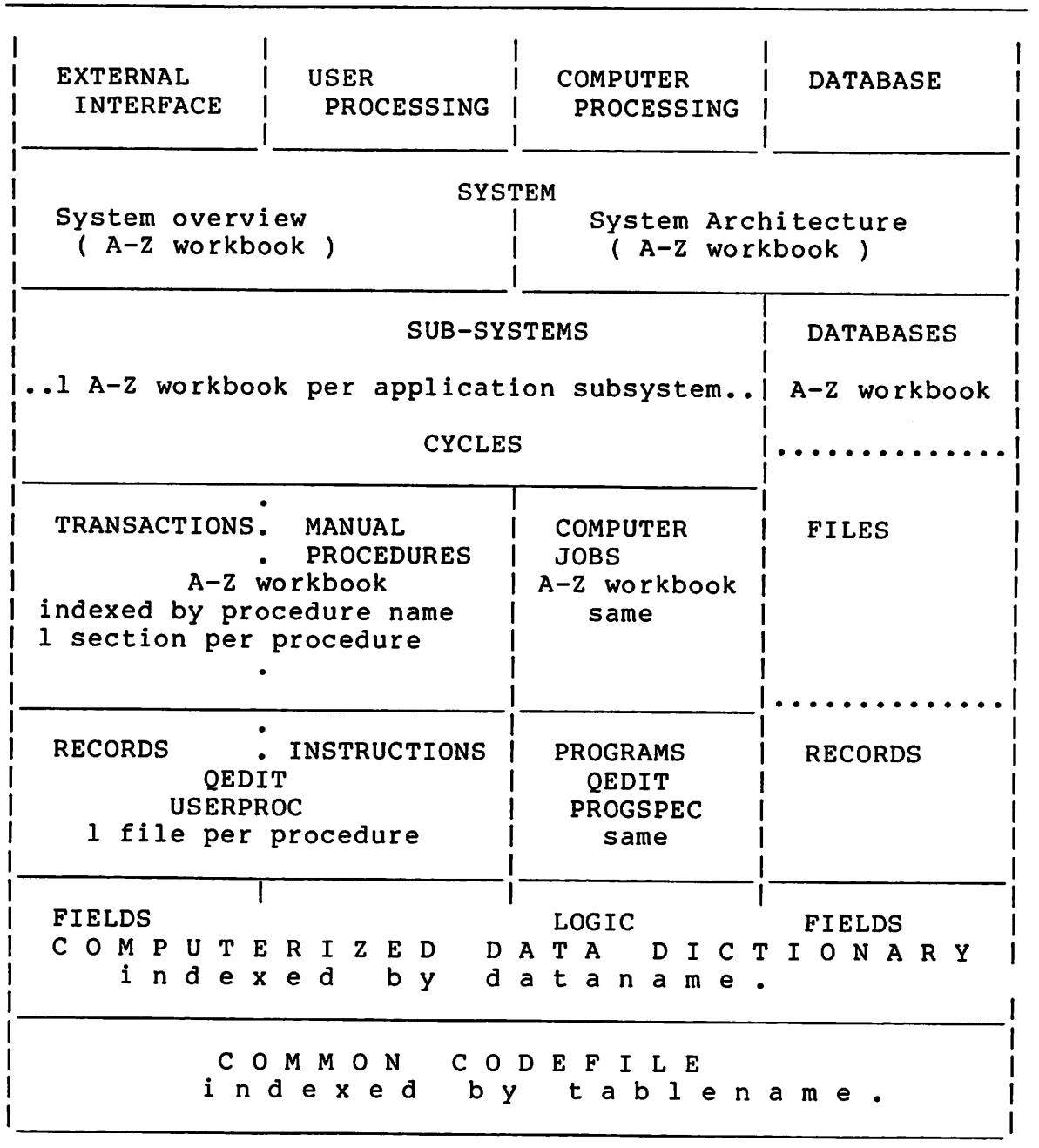
By segmenting the system documentation problem first into layers and then by level of detail we arrive at a matrix as shown below.



In practise we have not implemented a sperate module for each 'box' in the matrix. The database manuals include specifications for each database and the datasets and entries contained therein.

In addition, it does not make sense to seperate the definition of the external transactions from the manual procedures that process these transactions so these are combined in a User Procedure manual.

The documentation structure which corresponds to this system model is shown below.



Users of the documentation system must understand the structure to locate the appropriate manual. Then the table of contents is used to locate the actual section in the manual.

A comprehensive naming convention is used to identify components.

Each workbook is segmented by an A-Z set of dividers with a table of contents in the front of each manual.

The index consists of the letters A ->Z down the left hand margin with the title of each section filled out in pencil opposite the corresponding letter.

The A->Z dividers with the index is used for for the same reason that logical keys are preferable in a disc file, it uncouples the physical allocation of space in the manual from the contents This allows easy reorganization of the sections within or between manuals

new sections can be added

existing sections can be split up

two sections can be combined

In each case, the index is updated to reflect the new status .

Because of the flexibility of the table of contents and the ease of re-organization, the initial placement if not critical.

The information within each section is periodically reviewed and re-organized. Sometimes a section is set up that turns out not to be justified, the amount of material under that topic is too small or it really should have gone into another section. The section is removed and the entry in the table of contents at the front of the manual is erased.

The section is now available for re-use for another topic.

The development of large, complex systems without adequate concurrent documentation is a high risk venture.

It allows designer's to explore and improve their decisions on paper prior to committing their design to program code.

Users gain insight into how the system will work allowing meaningful input during design, reducing later costly rework.

Documentation also gives the system development process better visibility to management, reducing the pressure to 'do something' and commence programming too soon.

The learning curve for new systems team members is reduced.

With the organization techniques discussed here, we have been able to develop concurrent documentation in an organized manner that is useable and maintainable.