CHAPTER 3

SOFTWARE DOCUMENTATION

3.1 INTRODUCTION

Software Documentation is defined by Brockmann as "Communication designed to ease interactions between computer software and the individuals who manage, audit, operate or maintain it" (Barker Thomas T, 1991, Price 1984, Bell et al 1998). The purpose of software documentation is to improve user efficiency (and hence productivity) by answering questions, teaching techniques to overcome user's fear of software (or equipment) and to increase the retail attractiveness of the product. In this work the automatic generation of software documents has been attempted. The different types of documents, the planning and production of documents and the standards of documentation have been described in this chapter. The issues in planning and standardization has been considered in the planning and automatic generation of the software documents. In the next section the various types of software documents have been described (Bayer T 1993).

3.2 SOFTWARE DOCUMENTATION TYPOLOGY

Software documentation can be classified depending on the perspective required to be conveyed (Pressman 2000, Masse, Roger E 1985, Brockman R 1990, Guthrie, Barker, Brittan J.T. 1991). Software documents can
be classified as Internal vs. External and Reference vs. Tutorial, depending on the final consumers of the document, and the methodology of description. In this thesis an attempt has been made to generate external manuals at different levels of detail based on the status of the user.

3.2.1 Internal components

These documents are meant for internal consumption during the software development stage and are needed for software maintenance. Examples of such documents include design documents, requirement specifications, structure charts, flowcharts, prototypes, progress reports, bug reports, debugging logs, testing strategies, test results, pseudo code and program comments (Woolever et al 1994).

External documents are meant for consumers of the software and can be at different levels of detail (Coney, Mary B 1996). Examples include tutorials (interactive), user manuals, reference manuals, “getting started”, videos, context sensitive help, quick reference cards, keyboard overlays, onsite training, ‘hosted’ training, presentations. For effective software document generation, planning is necessary in order to obtain high quality (Novick D 2000).

Reference Vs Tutorial

Reference manuals rely on carefully designed reference aids; for effectiveness and emphasize system capability, possible contents, interface elements (position, function), keyboard shortcuts, menu items, customization
options and technical support. These are focused on a specific number of tightly-defined tasks like selection (and ordering) of tasks based on prioritization of users' needs. They may need to assume varying levels of ignorance of task-program-operating system-platform-computers. Almost invariably, it is best to adopt a 'step by-step' approach and to indicate specific aims before the steps. Visual support (diagrams, screen dumps etc.) is particularly important in a tutorial.

A tutorial creator needs to explain assumed prerequisite knowledge, indicate desired outcomes, relate material to user's existing knowledge, encourage necessary effort by providing incentives, point to related material (including reference) and assist in summarizing learning experience (Simpson 1988). The size of tutorial material is inversely proportional to capabilities and experience of the target audience. The aim is to help the user learn, but aim is not to teach everything, but to develop basic operating skills -- at which point tutorial becomes redundant.

A typical document needs to have some essential components to make it useful for the purpose for which it is intended (Tombre K and Antonie 1989, 1991). Different organizations adopt different standards for software documentation.

The general software development life cycle and the correspondent output documents generated are given in the following table 3.1. Document generation can be viewed from another perspective viz., the type of document generated during the different phases of the Software Development Life Cycle Model.
Table 3.1 Software development life cycle and its corresponding outputs

<table>
<thead>
<tr>
<th>Requirements Analysis</th>
<th>Feasibility Study, Outline Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Definition</td>
<td>Requirements Specification</td>
</tr>
<tr>
<td>Architectural Design</td>
<td>Design architecture specification, System test specification</td>
</tr>
<tr>
<td>Interface Design</td>
<td>Interface specification, Integration test specification</td>
</tr>
<tr>
<td>Detailed Design</td>
<td>Design Specification, Unit test specification</td>
</tr>
<tr>
<td>Coding</td>
<td>Program Code</td>
</tr>
<tr>
<td>Unit Testing</td>
<td>Unit test result report</td>
</tr>
<tr>
<td>Module Testing</td>
<td>Module test result report</td>
</tr>
<tr>
<td>Integration Testing</td>
<td>Integration test report, Final user manual</td>
</tr>
<tr>
<td>System Testing</td>
<td>System test report</td>
</tr>
<tr>
<td>Acceptance Testing</td>
<td>Final System</td>
</tr>
</tbody>
</table>

In this work emphasize has been given to generate documents at the requirement specification phase as external reference manual.

The first stage in planning a document is to describe the outline of the document.

Typically the outline of a document would consist of the following components:

- Types of reference support
- Glossaries
3.3 DOCUMENTATION DEVELOPMENT

Once the outline of documents has been described documents can be produced only with effective planning. Planning is dependent on the type of document and the software standards adopted. The following are the steps taken in the planning of documentation:

- Planning
- Documentation Development
- Use of Documentation Tools

3.3.1 Planning

The material to be consulted and/or accessed include design documents, requirements analysis, the program(s) [or at least a prototype], analysts and programmers, current and potential users. All possible documentation formats must be considered, especially manuals (reference, tutorial, user...), online (help files, web based), reference (cards, glossary...) and others (video...).
The inputs to the automatic software document generation phase are extracted from the various SRS techniques such as data flow diagrams, entity relationship diagrams, state transition diagram etc., from the viewpoint of current and potential users. The documentation formats of manuals must also be an important input (Popovich et al 1988).

Understanding the user enables the planning of the tone, style, language and emphasis of the documentation. The plan should include purpose, scope and limitations of the documentation, audience analysis, style information, justification of the purpose of various types of documentation, draft tables of contents, description of the deliverables, development process, team structure (Gehani 1986) and activities, estimated costing, development schedule, implementation/installation plans and maintenance/upgrade plans. The plan must be approved before any real development work starts (Pakin, Sandra 1984).

3.3.1.1 Understanding the User

An important aspect of documentation planning is the understanding of the user profile. The user’s needs can be considered along four dimensions.

- Level of (general) computer knowledge
- General educational level
- Cultural background
- Response to various message forms
Simplistically, three basic types of users can be defined

a) Novice: has little experience, understands isolated concepts, uses concrete- not abstract examples, relies on defaults and is component oriented.

b) Intermediate: has some experience, perceives context, begins to link concepts, uses options and is task oriented.

c) Advanced (Expert): is competent, understands structure and relationships, works in the abstract, uses all features, expects performance and is goal oriented.

In another perspective Brockmann suggests two groups viz:

Parrot - has no experience, can only deal with small chunks, do not think, question or synthesize. Intermittent - may be novice, intermediate or expert, work infrequently (with this system); do not learn so rely heavily on the interface.

Users should expect documentation to use familiar words, phrases and metaphors. Documentation must be able to cope with mixtures of computer and content knowledge. For example, an experienced accountant who is a novice user (of a particular system) requires different documentation to an experienced user who is a novice accountant.

When designing documentation that caters for the less experienced the following heuristics are to be used:
• Make explicit connections to existing knowledge
• Adopt flexible approach to using various media
• When designing documentation for a culturally diverse audience, choose vocabulary carefully
• Maximize use of graphical media

Good documentation should grow with users and as user level increases so should the level of documentation that is, good documentation is transformational. Documentation structure is also critical; sections relevant to various stages need to be explicitly marked. Well-designed navigation aids are very important in assisting diverse audiences. Once the nature of the audience has been determined, it is possible to proceed to the next stage, which is developing of the documentation. The next stage is the document development stage.

3.3.2 Documentation Development

Documentation usually goes through several drafts, during which blocks have to be overcome, writing styles must be matched to users, graphics created/adapted, text edited and proofed, navigation aids developed and the packaging designed.

The client checks the technical accuracy and completeness. Usability testing is carried out to see if all (user-centred) goals have been met. Usability testing uses (wherever possible) the release version of the software. In designing the documents users expectations and specifications of the documents must be defined clearly. In this work the document produced must mirror all
entities and concepts and relations between them that are specified by the different SRS techniques.

3.3.2.1 Documentation Specifications

While stating the purpose, users expect documentation to provide one or more of the following:

Instruction -- help with learning to use the system
Reference -- detailed information about an aspect of the software
Assistance -- with a particular task, concept or interface element

The steps in specifying documentation are as follows: - First, break down documentation by tasks. Then define in detail and plan for the audience. After that, state the purpose of the documentation. Finally organize the content and develop visualization. During development, specifications can make the process more professional, allow early interaction in design, coordinate activities of the group, diminish writer's block, act like a contract, predict future performance and productivity and aid document maintenance. Once the specification has been clearly defined the next stage is documentation designing.

3.3.2.2 Documentation Design Factors

In general documentation design is carried out by documentalist who have linguistic knowledge and have also acquired knowledge about the
software to be documented and the software documents standards to be adopted.

Some of the factors to be considered while designing a document are as follows:

-- Plan the documentation process and specify the requirements of the documentation
-- Technical accuracy and completeness
-- The information must be correct
-- Logical arrangement
-- Structure and layout
-- Have an impact on both the short and long-term utility of the documentation

In automatic generation of software documentation the designing of the document was one of the most challenging tasks. In order to aid in this designing specially designed document design and standardization information to help in designing and planning the document.

Developers, analysts, programmers need documentation to help with managing and maintaining the code, understanding program structure and behavior, understanding language syntax and grammar (BNF diagrams, language reference).
3.3.3 Use of Documentation Tools

General document tools are word processors, layout programs, graphics programs, with some special features templates / master pages and automated creation of navigation aids. Documentation specialized tools are converters, help compilers, hypertext editors, groupware systems and CASE tools. There is a need to decide on which documentation tools are to be used.

An implementation for structured documents usually uses different kinds of programs in the way shown in the following figure 3.1. DTD tools are used to write structure definitions, editors and structure editors are used to key in the content. Parsers are used to check the validity and page layout programs and DSSSL tools to produce a printed version. Search programs, databases, database front-ends and electronic delivery tools allow the user to query and retrieve documents. API tools can do integration of programs.

A short description of each component is given below:

- **text editor**
  Text editors are usually programs that are only able to input, update and output an ASCII text with the help of a human user.

- **structure editor**
  A program that is used to write text.

- **desktop publishing software**
  These programs are used for producing page layout for paper and electronic documents.
Figure 3.1 Documentation tools
• **formatter/page layout program**
  This is a program to generate a formatted form of a text for printing. The text is made with the use of a text editor.

• **search program**
  The search program usually preprocesses a text in order to improve the search process. Different preprocessing methods are used. These programs contain a query language and a search engine. The query language is used to express the queries and the search engine processes the query and produces an answer. Query languages for structured text usually contain the following expressions as their basic components:

  ➢ Search "program"
    a character string program is searched anywhere.

  ➢ Search "program" inside element "description"
    A character string program is searched inside a structure element description.

  ➢ Search "program" inside "description" whose "type"-attribute is "editor"
    a character string program is searched inside a structure element description that has an attribute whose name is type and whose value is editor.

Depending on the query language it is possible to form more complicated queries from these basic query components.
- **electronic delivery tool**
  Electronic delivery tools are used to define the display layout of structured texts.

- **browser**
  Browsers are programs that are used to browse different kinds of files, e.g., SGML files, DTDs. Browsers operate either on the file system or they can show data given by Web servers.

- **Database**
  In relational databases the data is modeled by relations and the operations on the data are modeled by relational algebra or relational calculus. The query language is the SQL (Structured Query Language) ISO standard.

- **Text database**
  A database management program for text. Data models and query languages vary. The text can be to some extent structured, documents can be subdivided into parts.

- **Structured text database**
  is group contains very different kinds of programs whose implementation can be based on a relational database, a text database or an object-oriented database.

- **Document database**
  The queries of document databases will return only whole documents. Search for parts of documents is not possible.
• Database front-end  
These programs are used for viewing and sometimes also for editing the information stored in the database.

• Conversion program  
These programs or language interfaces are used to make conversion programs from one text format to another.

• parser  
Parser is a program that parses the text confirming to a grammar. Because SGML does not involve a specified grammar but can be used to describe texts for different kinds of grammars, SGML parsers are actually meta parsers. They take as input a grammar and generate a parser for this grammar.

• API (Application Programmer's Interface) tool  
Mostly a set of functions or procedure libraries are used to allow a program to interface with other programs.

• DTD tool (structure design tool)  
Programs to process (create, edit, show) document type definitions (DTD's) for structured documents.
• DSSSL tool (layout design tool)

A DSSSL tool contains functions to create and edit DSSSL specifications, to check their correctness or/and to apply specifications to SGML documents to output formatted texts.

These documentation tools help the documentalist to format the document while automatically generating document emphasize has been given to the content of the document and not its formatting. The next section describes the various standards of software document.

3.4 STANDARDS IN SOFTWARE DOCUMENTATION

This glossary includes terms and definitions from the following ISO standards for the presentation, identification and description of documents.

• ISO 4:1997, Information and documentation -- Rules for the abbreviation of title words and titles of publications.
• ISO 215:1986, Documentation - Presentation of contributions to periodicals and other serials.
• ISO 690-2:1997, Information and documentation - Bibliographic references -- Part 2: Electronic documents or parts thereof
• ISO 832:1994, Information and documentation - Bibliographic description and references - Rules for the abbreviation of bibliographic terms.

• ISO 999:1996, Information and documentation -- Guidelines for the content, organization and presentation of indexes

• ISO 1086:1991, Information and documentation - Title leaves of books

• ISO 2146:1988, Documentation - Directories of libraries, archives, information and documentation centers, and their databases.

• ISO 2788:1986, Documentation - Guidelines for the establishment and development of monolingual thesauri.


• ISO 5122:1979, Documentation - Abstract sheets in serial publications.


• ISO 5963:1985, Documentation - Methods for examining documents, determining their subjects, and selecting indexing terms.
• ISO 5964:1985, Documentation - Guidelines for the establishment and development of multilingual thesauri.

• ISO Committee Draft 5966 (1999), Information and documentation - Guidelines for the presentation of technical reports. [A revision of ISO 5966:1982].

• ISO 6357:1985, Documentation - Spine titles on books and other publications.

• ISO 7144:1986, Documentation - Presentation of theses and similar documents.

• ISO 7154:1983, Documentation - Bibliographic filing principles.

• ISO 7275:1985, Documentation - Presentation of title information of series.


• ISO 9115:1987, Documentation - Bibliographic identification (biblid) of contributions in serials and books [withdrawn].

• ISO 10324:1997, Information and documentation - Holdings statements - Summary level

3.5 DOCUMENTATION STANDARDS

In order to produce a good document that follows standards it is necessary to understand the various details that go into a standard document. In this work the standard has been used at the backbone in both the design of the document design patterns and in the planning stage.

Any documentation standard has four sections.

3.5.1 Section 1
3.5.1.1 Scope

i) Identification

This Documentation Standard establishes the required standard for specifications, standards, plans, procedures, manuals, reports records, software development files and other data items to be prepared during the design,
development of Systems, subsystems, Hardware Configuration Items (HWCI), Computer Software Configuration Items (CSII).

ii) Purpose

The purpose of this standard is to detail requirements for the documentation to be prepared during the system life cycle of the identified project. The documents described shall be in accordance with the requirements specified in the SOW.

This document meets the requirement of the contractual standards and provides a document strategy/standard for the documentation to be prepared and evaluated during the total system life cycle.

iii) Document overview

The 'Documentation Standard' consists of the following sections:

- Section 1 identifies, states the purpose and provides a brief summary;
- Section 2 lists all referenced documents;
- Section 3 provides a requirements definition of the document preparation, style, layout, format, etc., together with the procedures for documentation change, revision, and amendment;
- Section 4 identifies the documentation groups to be prepared during the system and its development life cycles;
• Section 5 briefly explains the purpose of each of the documents to be prepared and provides a brief summary overview;
• Section 6 provides a list of definitions, abbreviations, and acronyms, and other non-mandatory information used throughout the document.

Appendixes to this document contain example texts for the document to be prepared during the system and its development life cycles. These shall be in accordance with data item descriptions identified by the referenced documents.

iv) Relationship to other standards and plans

The Documentation Standard shall be used in conjunction with the 'System Engineering Management plan' and 'Configuration Management Plan' to provide a fully integrated Project Management System in accordance with the 'Project (Program) Plan for the system.

This document provides a means of achieving the concept of minimum documentation required by Engineering Management (MIL-STD-499A) by identifying required plans, reports, and relevant data items.
3.5.2 Section 2

3.5.2.1 Referenced Documents

i) Government documents

- MIL-STD-961
- MIL-STD-962
- MIL-STD-970
- MIL-STD-490
- DEF STAN 05-28

SPECIFICATIONS

- MIL-S-83490 Specification, types and forms
- etc.,

ii) Non-Government documents

*Identify any non-government documents used as reference for documentation standards for example, IEEE, BS, etc.*

iii) Order of precedence

In the event of conflict between this standard and other documents referenced herein, the documentation requirements of this standard shall apply. All documents referenced are to the following identified issues unless otherwise stated. Where no issue is quoted the issue in force at the date of actual contract let shall apply.
iv) Source of documents
2.4.1 Government specifications, standards, and handbooks.
2.4.2 Non-Government documents.
3.5.3 Section 3
3.5.3.1 Style, format and identification

i) General

This paragraph includes the requirements for style, format, and general instructions for preparing and evaluating documentation. This includes material arrangement, titling, paragraphing, numbering, heading, security, binding, checks and concluding material.

ii) Sectional arrangement of documentation

Documentation shall normally contain six numbered sections, and appendices as required, titled and numbered as follows:

1. SCOPE
2. APPLICABLE DOCUMENTS
3. REQUIREMENTS
4. QUALIFICATION REQUIREMENTS
5. PREPARATION FOR DELIVERY
6. NOTES

A. APPENDIX

Subject matter shall be kept within the scope of the section so that the same kind of requirements or information will always appear in the same section of every document.
Each section shall be divided into numbered and titled paragraphs and subparagraphs as necessary.

iii) Language style

iv) Paragraph numbering

Each paragraph and subparagraph shall be numbered consecutively within each section of the document, using a full stop (.) - period- to separate the number representing each breakdown. Each section, paragraph and subparagraph shall be numbered consecutively using a decimal notation as follows:

3. SECTION
3.1 First paragraph.
3.1.1 First subparagraph.
3.1.2 Second paragraph.
3.2 Second paragraph.
3.2.1 First subparagraph.
3.2.2 Second paragraph.

In order to avoid confusion with paragraph numbers itemization (lists) within a paragraph or subparagraph shall be identified by lower-case letters followed by a full stop.
3.5.4 Section 4

3.5.4.1 GENERAL CONSIDERATIONS

This section discusses the considerations to be addressed when preparing documentation of a project.

i) System/software development life cycle

A system/software development life cycle shall be established as defined in the 'Software Development Standards' document and shall consist of the following phases:

- System requirements analysis and system design;
- Software requirements analysis;
- Preliminary design;
- Detailed design;
- Coding and Unit testing;
- Unit integration and testing;
- CSCI testing
- System integration and testing.

Detailed descriptions of the above phases are provided in the 'Software Development Standards'.

ii) System/hardware development life cycle

A system/hardware development life cycle shall be established and comprise the following:
• System/hardware requirements analysis;
• Hardware requirements analysis;
• Preliminary design;
• Detailed design;
• Fabrication;
• HWCI testing;
• System integration and testing.

iii) Need for documentation and standardization

Some of the purposes that documentation and standardization serve are to:

• Provide managers with documents to evaluate a significant developmental milestones to determine that requirements have been met and that resources should continue to be expended;
• Record technical information to allow coordination of later development, use, and modification;
• Provide authors of documents and managers of the project development a guide to follow in preparing and checking documentation;
• Provide uniformity of style, format and content of documentation throughout the project;
• Prevention of variety and promotes variety reduction.

iv) General considerations on documentation planning and preparation.
Manager

The manager with responsibility for the specific discipline program plan shall ensure:

- A documentation plan should be developed early in the process of project development, including:
  - Which document types will be prepared and the appropriate level of detail;
  - The dates that the documents must be available;
  - Document evaluation procedures, as applicable.
- The prepared documents are updated properly and in a timely manner.

Authors

The author of a document should ensure that they have:

- An understanding of its relationship to other associated documentation;
- An understanding of the overall content required in the document that is to be prepared. Frequently, a section in a document that is intended to be very general is written before a later section that is intended to provide specifics. This often leads to the inclusion of too much detail in the general section. This can be avoided by reviewing the outline of the document type before writing has started;
An understanding of the audience who will use the document. For example, although "input" is discussed in the operation and support documents the detail presented in each is different since each is intended for a different audience. Audience types are, individuals, a group of individuals, user groups, and development groups.

**Document audiences**

Authors when preparing documentation shall consider the following. When each document is written the function of the audience that will use the document must be considered to ensure that the information is presented using appropriate terminology and to an adequate level of detail.

**Audience types**

Each document type is prepared for and read by a specific audience. The audience may be an individual or group of individuals. The two audiences involved in all projects will be the development Group (Team) and User groups.

1. The development group may be the acquirer or a supplier. As a minimum, the group shall perform the design and coding functions;
2. The User Group is usually the acquirer or is agent that requested the project. As a minimum, the User Group comprises of those who provide inputs and those who utilize the outputs from the project.
• Audience composition

Regardless of the organizational relationships between the development team and the user group, the functions of the two are separate. Both of these groups may consist of personnel with separate titles and functions that include the following:

Document types

Document types are produced during the evolutionary development of the system and its comprising hardware and software.

Standard generalized mark-up language (SGML)

When documentation is required to be marked-up (tagged) it shall be as a minimum to ISO 8879 (SGML). ISO 8879 defines a basic set of requirements for the digital data form of page-oriented technical parameters. This data shall be prepared in conformance to these requirements and will facilitate the automated storage, retrieval, interchange, and processing of technical documents (Uma G.V., Geetha T.V 2001d).

CONCLUSION

Producing software documentation manually often results in inconsistent and incomplete documents because engineering information is always changing during the course of software development. Furthermore, manual documentation is very labor intensive and error-prone. As the complexity of software grows, organizations have an increasing need to
automate and consistently maintain software documentation that captures the entire software development process. Even while automatically generating software documents, software standards have to be followed in order to have a uniform and consistent structuring of documents. Hence automatic software documentation is a challenging task of natural language generation, that considers the various stages and inputs used by the documentalist in generating a document that caters to standards and satisfies users needs by adopting linguistically correct words syntax and organization. In order to effectively generate software documents inputs have to be identified and represented in a proper manner. Hence the content determination phase plays an important role in identifying the inputs for generating any natural language text.