IEEE STANDARD 1016: Software Design Specification

The Software Design Specification (SDS) sections provide you with guidelines related to the structure and the contents of SDS document. The Software Design Specification document includes at least these sections.

For the project, your team may have good reasons for wanting to deviate from this proposed outline. If a section is not applicable in your case, do not delete it; instead, give the topic heading and write "Not applicable".

You will note that there is some overlap in the content between different documents (i.e. the User Requirements Specification Document and the Software Design Specification Document.) This redundancy allows each document to stand on its own.

The Software Design Specification Outline

1. Introduction

1.1 Purpose of this document

Full description of the main objectives of the SDS document.

1.2 Scope of the development project

This will be similar to what was written in the SRS.

1.3 Definitions, acronyms, and abbreviations

Be sure to alphabetize!

1.4 References

This section will include technical books and documents related to design issues. Be certain that the references you give are complete and in the appropriate format.

1.5 Overview of document

A short description of how the rest of the SDS is organized and what can be found in the rest of the document. This is not simply a table of contents. Motivate and briefly describe the various parts!

2. System architecture description

2.1 Overview of modules / components

This subsection will introduce the various components and subsystems.

2.2 Structure and relationships

Make clear the interrelationships and dependencies among the various components. Structure charts can be useful here. A simple finite state machine can be useful in demonstrating the operation of the product. Include explanatory text to help the reader understand any charts.

2.3 User interface issues

This section will present the main principles of the product's user interface. Use the personas defined in section 2.1 of your SRS to make specific examples. This section should not touch on technical details. You may want to include sketches and specific text messages.

3. Detailed description of components

3.1 Component template description

This section is not part of your design. It is the pattern you will use to describe the components given in subsections 3.2 - 3.n. Each part of the template will be identified by a label. Here in 3.1, you must briefly explain the purpose of each point. To make the presentation clear, use a table or bullet list. You may adapt the template suggested below to your particular needs (although deviations from the suggested template should be minimal and well motivated).

3.2 X Component (or Class or Function ...)

Use exactly the template you define in 3.2. If a part of the template is not applicable, then mark it N/A rather than omitting it.

3.3 Y Component (or Class or Function ...)

...

3.n Z Component (or Class or Function ...)

4.0 Reuse and relationships to other products

For teams doing enhancement work, reuse is an important issue. Most enhancement work should focus on extending, rather than replacing, the design and product development from earlier semesters. For teams doing new development, reuse can also be an important strategy. In some cases, there is freeware that could be incorporated. In other cases, there are existing modules or classes that could be adapted. Another possibility is the use of special tools that produce open source results and thus permissible under the terms of this course.

This section should include the following subsections as appropriate:

- how reuse is playing a role in your product design
- how reuse is playing a role in your product implementation (and the motivation for changes)
- if you are not reusing material that is available, then give motivation for why it is being thrown out.

5.0 Design decisions and tradeoffs

Use this section to motivate any decisions that will help the reader understand the design that your team is using. This section can also capture good ideas that were abandoned and the reasons for leaving them out of the design.

6.0 Pseudocode for components

7.0 Appendices (if any)

SDS component template

The template given below suggests a reasonable structure for giving a thorough description of each component described in Part 3 of the SDS. The specific information depends in part on the design approach. Your team must adapt this template to your needs and describe it in section 3.1 of your SDS.

Identification	The unique name for the component and the location of the component in the system.
Type	A module, a subprogram, a data file, a control procedure, a class, etc
Purpose	Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS, but are implied or adjunct to formally stated SDS requirements.
Function	What the component does, the transformation process, the specific inputs that are processed, the algorithms that are used, the outputs that are produced, where the data items are stored, and which data items are modified.
Subordinates	The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part.
Dependencies	How the component's function and performance relate to other components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components.
Interfaces	Detailed descriptions of all external and internal interfaces as well as of any mechanisms for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here.
Resources	A complete description of all resources (hardware or software) external to the component but required to carry out its functions. Some examples are CPU execution time, memory (primary, secondary, or archival), buffers, I/O channels, plotters, printers, math libraries, hardware registers, interrupt structures, and system services.
Processing	The full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic.
Data	For the data internal to the component, describes the representation method, initial values, use, semantics, and format. This information will probably be recorded in the data dictionary.