SOFTWARE DESIGN DOCUMENT

1. Introduction

The following subsections of the Software Design Document (SDD) should provide an overview of the entire SDD.

1.1 Purpose

This subsection should explain the purpose of the SDD and specify the intended audience for it. The SDD described the software structure, software components, interfaces and data necessary for the implementation phase. Each requirement in the SRS should be traceable to one or more design entities in the SDD.

1.2 Scope

This subsection should relate the design document to the SRS and to the software to be developed.

1.3 Definitions, Acronyms and Abbreviations

This subsection should provide the definitions of all terms, acronyms and abbreviations that are used in the SDD.

2. References

This subsection should provide a complete list of all documents referenced in the SDD. It should identify these references by title, report number, date and publishing organization. It should also specify the sources from which these references are available.

3. Attributes of Design Entities

There are some attributes common to all entities, regardless of the approach utilized, whether procedural or object-oriented. These are used in subsections 4 and later.

3.1 Identification

The name of the entity should be specified. Two entities should not have the same name.

3.2 Type

The type attribute should describe the nature of the entity. It may simply name the kind of entity, such as subprogram, module, procedure, process, data item, object etc. Alternatively, design entities can be grouped, in order to assist in locating an entity dealing with a particular type of information.

3.3 Purpose

This is a description of why the entity exists. It provides the rationale for the creation of the entity. Therefore it designates the specific functional and performance requirements for which this entity was created, using the SRS.

3.4 Function

The function attribute should state the transformation applied by the entity inputs to produce the desired output. In the case of a data entity, this attribute should state the type of information stored or transmitted by the entity.

3.5 Subordinates

The subordinates attribute should identify the entities composing this entity. This information is used to trace requirements to design entities and to identify the parent/child structural relationships through a software system decomposition.

3.6 Dependencies

The dependencies attribute should identify the relationship of the entity with other entities. It describes the nature of each interaction that may involve initiation, order of execution, data sharing, creation, duplicating, usage, storage or destruction of other entities.

3.7 Interface

The interface attribute describes how other entities interact with this entity. It should describe the methods of interaction and rules governing those interactions. It provides a description of the input ranges, the meaning of inputs and outputs, the type and format of each input or output, and output error codes.

3.8 Resources

The resources attribute identifies and describes all of the resources external to the design that are needed by this entity to perform its function. It provides information about items such as physical devices (printers, discs, memory), software services (math libraries, operating system services, graphical user interface libraries), and processing resources (CPU cycles, memory allocation).

3.9 Processing

The processing attribute describes the rules used by the entity to achieve its function. It describes the algorithm used by the entity to perform a specific task. It includes sequencing of events or processes, process steps, conditions, termination criteria etc.

3.10 Data

The data attribute describes the method of representation, initial value, use, format and acceptable values of internal data.

4. Decomposition Description

4.1 General Structure

This section of the SDD should record the division of the software system into design entities. It describes the way the system is structured and the purpose and function of each entity. For each entity, it provides a reference to the detailed description. It uses the identification, type, purpose, function and subordinates attributes.

4.2 Procedural Approach

If a procedural approach is used, this includes a description of the basic modules of the system and how they relate to other modules (which modules it calls etc.) Textual descriptions should also be provided for each module that the system is decomposed into.

4.2.1 Module Decomposition

This subsection describes the decomposition information as given in 3.1 for software modules.

4.2.2 Data Decomposition

This subsection describes the decomposition information as given in 3.1 for data elements.

4.3 Object-Oriented Approach

In case an object-oriented approach is used, there should be class(inheritance) diagrams showing the classes in the system. Textual descriptions should also be provided for each class/object that the system is decomposed into. Use UML to describe the object-oriented decomposition under following sub-sections.

4.3.1 Use Case Diagrams

4.3.2 Class Diagrams

4.3.3 Sequence Diagrams

4.3.4 Statechart Diagrams

4.3.5 Activity Diagrams

5. Dependency Description

This subsection describes the dependencies between different entities. It uses the identification, type, purpose, dependencies and resources attributes.

6. Interface Description

This subsection describes everything designers, programmers and testers need to know to correctly use the functions provided by an entity. It includes the details of external and internal interfaces not provided in the SRS. It uses the identification, function and interfaces attributes.

7. Detailed Design

It contains the internal details of each design entity. These details include attribute descriptions for identification, processing and data. It contains all the details that will be needed by the programmers for implementation. Short English-like descriptions can be used to describe the algorithms utilized. Data structure details should also be given.