**SYSTEM DESIGN DOCUMENT[1]**

The System Design Document (SDD) is written after the initial system decomposition is done, and updated throughout the development. SDD describes the services provided by each subsystem. Although this section is usually empty or incomplete in the first versions of the SDD, this section serves as a reference for teams for the boundaries between their subsystems.

SDD is used to define interfaces between teams of developers and serve as a reference when architecture-level decisions need to be revisited. The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem.

1. **Introduction**

Provide a brief overview of the software architecture and the design goals. It also provides references to other documents and traceability information (e.g., related requirements analysis document, references to existing systems, constraints impacting the software architecture)

* 1. **Purpose of the System**
  2. **Design Goals**
  3. **Definitions, Acronyms, and Abbreviations**
  4. **References**

References to existing systems, etc.

1. **Current Software Architecture**

Describe the architecture of the system being replaced (remember hepsiburada from RAD)**.**

If you are working on a scientific problem, describe the solutions in the literature. (In RAD, you have mentioned only the problem)

If you answer no for both cases, remove this part.

1. **Proposed Software Architecture**

Documents the system design model of the new system

* 1. **Overview**

Present a bird’s-eye view of the software architecture and briefly describes the assignment of functionality to each subsystem.

* 1. **System Decomposition**

Describe the decomposition into subsystems and the responsibilities of each. This is the main product of system design.

* 1. **Hardware/Software Mapping**

Describe how subsystems are assigned to hardware and off-the-shelf components. It also lists the issues introduced by multiple nodes and software reuse.

* 1. **Persistent Data Management**

Describe the persistent data stored by the system and the data management infrastructure required for it. This section typically includes EER, the description of data schemes, the selection of a database, and the description of the encapsulation of the database.

* 1. **Access Control and Security**

Describe the user model of the system in terms of an access matrix. This section also describes security issues, such as the selection of an authentication mechanism, the use of encryption, and the management of keys.

* 1. **Global Software Control**

Describe how the global software control is implemented. In particular, this section should describe how requests are initiated and how subsystems synchronize. This section should list and address synchronization and concurrency issues.

* 1. **Boundary Conditions**

Describe the start-up, shutdown, and error behavior of the system. (If new use cases are discovered for system administration, these should be included in the requirements analysis document, not in this section.)

1. **Subsystem Services**

Describe the services provided by each subsystem. Although this section is usually empty or incomplete in the first versions of the SDD, this section serves as a reference for teams for the boundaries between their subsystems. The interface of each subsystem is derived from this section and detailed in the Object Design Document.

1. **References**

The following is an example of listing a book in this section. Check the text to see how it is cross referenced (The whole document is based on [1]).

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.