# Software Design Document

For E-banking System

Group 2 (1714086-Hritik Jaiswal,1714096- Niket Mehta ,1714097-Riddhisha Nirmal ,1714100-Vedang Paranis)

Exp 8 OOSE

Aim : To prepare Software Design Document (SDD) for your project.

Result: Attached a Document with this file.(Created using Latex Editor)

Postlab Question: Explain Architecture design patterns and styles with examples.

Ans. The software needs the architectural design to represents the design of software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectural styles.  
Each style will describe a system category that consists of :

A set of components(eg: a database, computational modules) that will perform a function required by the system.

The set of connectors will help in coordination, communication, and cooperation between the components.

Conditions that how components can be integrated to form the system.

Semantic models that help the designer to understand the overall properties of the system.

The use of architectural styles is to establish a structure for all the components of the system.

Types:

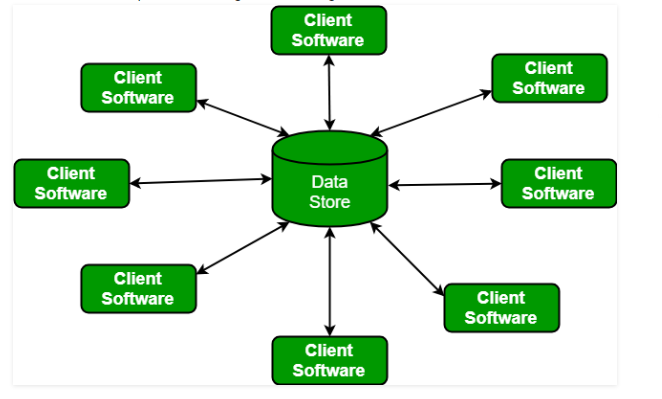
1. Data centred architectures:

A data store will reside at the center of this architecture and is accessed frequently by the other components that update, add, delete or modify the data present within the store.

The figure illustrates a typical data centered style. The client software access a central repository. Variation of this approach are used to transform the repository into a blackboard when data related to client or data of interest for the client change the notifications to client software.

This data-centered architecture will promote integrability. This means that the existing components can be changed and new client components can be added to the architecture without the permission or concern of other clients.

Data can be passed among clients using blackboard mechanism.



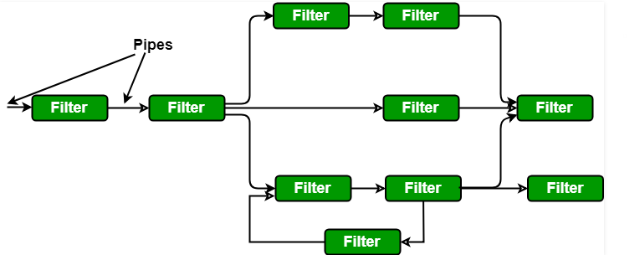
1. Data flow architectures:

This kind of architecture is used when input data to be transformed into output data through a series of computational manipulative components.

The figure represents pipe-and-filter architecture since it uses both pipe and filter and it has a set of components called filters connected by pipes.

Pipes are used to transmit data from one component to the next.

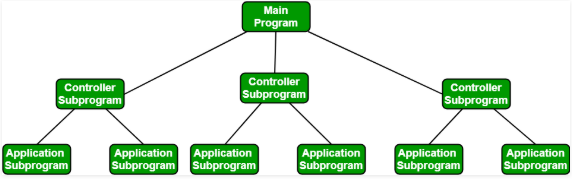
Each filter will work independently and is designed to take data input of a certain form and produces data output to the next filter of a specified form. The filters don’t require any knowledge of the working of neighboring filters.

If the data flow degenerates into a single line of transforms, then it is termed as batch sequential. This structure accepts the batch of data and then applies a series of sequential components to transform it. 

1. Call and Return architectures: It is used to create a program that is easy to scale and modify. Many sub-styles exist within this category. Two of them are explained below.

Remote procedure call architecture: This components is used to present in a main program or sub program architecture distributed among multiple computers on a network.

Main program or Subprogram architectures: The main program structure decomposes into number of subprograms or function into a control hierarchy. Main program contains number of subprograms that can invoke other components.

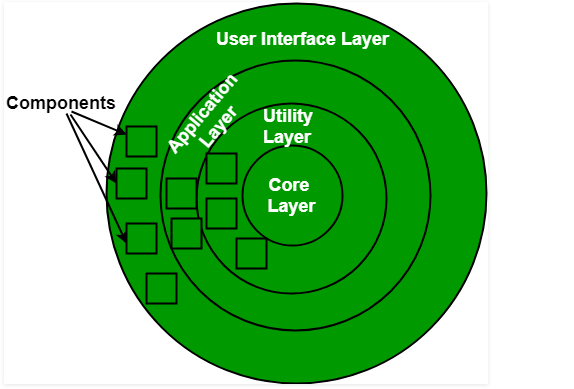


1. Object Oriented architecture: The components of a system encapsulate data and the operations that must be applied to manipulate the data. The coordination and communication between the components are established via the message passing.
2. Layered architecture:

A number of different layers are defined with each layer performing a well-defined set of operations. Each layer will do some operations that becomes closer to machine instruction set progressively.

At the outer layer, components will receive the user interface operations and at the inner layers, components will perform the operating system interfacing(communication and coordination with OS)

Intermediate layers to utility services and application software functions.



Outcome: Comprehend design pattern, architectural style usage and software planning

Paradigm

Conclusion: The Software Design Document was written for our project. Its importance and need were studied.