Unit 6: Design and Implementation

Software design and implementation is the stage in the software engineering at which an executable software system is developed. Software design is a creative activity in which you identify software components and their relationships, based on a customer's requirements. Implementation is the process of realizing the design as a program. This is the stage in which we design the software either using structured design or an object oriented design approach and only then an executable software system is developed. These activities are invariably interred leaved. Software design is a creative activity in which you identify software components and their relationships, based on a customer's requirements. Implementation is the process of realizing the design as a program. The level of detail in the design depends on the type of system being developed and methods such as plan driven or agile approach that you are using. Design and implementation are closely linked and you should take implementation issues into account when developing a design. While implementing a system, the most important decisions that have to be made at an early stage of software project is whether or not you should buy or build application. It is now possible to buy off the shelf system that can be adapted to the user's requirement. When you are developing a system using COTS components then design process becomes concerned with how to use the configuration features of that system to deliver the system requirements.

Object-Oriented Design Using the UML

- Object-oriented design processes involve developing a number of different system models.
- They require a lot of effort for development and maintenance of these models and, for small systems, this may not be cost-effective.
- However, for large systems developed by different groups design models are an important communication mechanism.
- There are a variety of different object-oriented design processes that depend on the organization using the process.
- Common activities in these processes include:
    - Define the context and modes of use of the system
    - Design the system architecture

- Identify the principal system objects
- Develop design models.
- Specify object interfaces.

(Note: Use different types of UML diagram from unit 4)

Design Patterns

Design pattern is a general repeatable solution to a commonly occurring problem in software design. A design pattern isn't a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations. They define a common language that helps your team communicate more efficiently. It is a description or template for how to solve a problem that can be used in many different situations. Patterns are formalized best practices that the programmer can use to solve common problems when designing an application or system. Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved

Design patterns can speed up the development process by providing tested, proven development paradigms. Effective software design requires considering issues that may not become visible until later in the implementation. Reusing design patterns helps to prevent subtle issues that can cause major problems and improves code readability for coders and architects familiar with the patterns. Design patterns are usually associated with object oriented design that relies on object characteristics such as inheritance and polymorphism. Hence pattern is a way of reusing abstract knowledge about a problem and its solution.

Implementation Issues

A critical stage of SDLC is the system implementation in which we create an executable code of the software using high level programming language from scratch or using off the shelf systems to meet the specific requirements of the user. Some aspects of implementation issue are:

- **Reuse**: Most modern software is constructed by reusing existing components or systems. When you are developing software, you should make as much use as possible of existing code.

- **Configuration management**: During the development process, you have to keep track of the many different versions of each software component in a configuration management system.
- **Host-target development:** Production software does not usually execute on the same computer as the software development environment. Rather, you develop it on one computer (the host system) and execute it on a separate computer (the target system).

Open-Source Development

- Open source development is an approach to software development in which the source code of a software system is published and volunteers are invited to participate in the development process.
- Its roots are in the Free Software Foundation (www.fsf.org), which advocates that source code should not be proprietary but rather should always be available for users to examine and modify as they wish.
- Open source software extended this idea by using the Internet to recruit a much larger population of volunteer developers. Many of them are also users of the code.
- ″ The best-known open source product is, of course, the Linux operating system which is widely used as a server system and, increasingly, as a desktop environment.
- Other important open source products are Java, the Apache web server and the mySQL database management system.

Open source licensing

A fundamental principle of open-source development is that source code should be freely available; this does not mean that anyone can do as they wish with that code.

- Legally, the developer of the code (either a company or an individual) still owns the code. They can place restrictions on how it is used by including legally binding conditions in an open source software license.
- Some open source developers believe that if an open source component is used to develop a new system, then that system should also be open source.
- Others are willing to allow their code to be used without this restriction. The developed systems may be proprietary and sold as closed source systems.