Chapter 27
A Survey on Different Approaches to Automating the Design Phase in the Software Development Life Cycle

Sahana Prabhu Shankar
https://orcid.org/0000-0001-8977-9898
Ramaiah University of Applied Sciences, India

Harshit Agrawal
https://orcid.org/0000-0003-3515-0474
Ramaiah University of Applied Sciences, India

Naresh E.
https://orcid.org/0000-0002-8368-836X
M. S. Ramaiah Institute of Technology, India

ABSTRACT

Software design is a basic plan of all elements in the software, how they relate to each other in such a way that they meet the user requirements. In software development process, software design phase is an important phase as it gives a plan of what to do and how to do it during the implementation phase. As the technology is evolving and people's needs in the technological field are increasing, the development of software is becoming more complex. To make the development process somewhat easy, it is always better to have a plan which is followed throughout the process. In this way, many problems can be solved in the design phase, for which a number of tools and techniques are present. One is known as Design Patterns. In software engineering, a design pattern is a general solution to commonly occurring problems in software design. A design pattern isn't a finished design that can be transformed directly into code.

DOI: 10.4018/978-1-6684-3702-5.ch027

INTRODUCTION

For any software to be built, it is highly important that it satisfies the client's needs. In order to completely understand what the client wants from the software, Requirement Analysis phase plays a vital role as in this phase, the analyst tries to gain as much about the domain, functioning and basic plan of the system to be designed. Once the requirement analysis phase is completed, next is the Design phase. In design phase, a basic outline/plan is made for the software to be built which includes all the elements which will be used in the implementation phase. If there is any ambiguity in the analysis phase, the design following the requirements collected, will also have the same ambiguity. Thus, it is very important to collect and interpret all the user requirements carefully, correctly and completely.

It is suggested to always first define a basic design of the software to analyse its behaviour and also to identify the short-term risks, so that informed design decisions can be made before the system goes in the implementation phase. Not only this, but designing prior to the implementation phase can also be helpful to breakdown the complex problems in simple sub-problems and thus making the implementation part easier. From a good design phase, the software gains higher productivity, better code maintainability, higher adaptability, and Quality. Thus, a good design is recommended for success of any software project.

The advantages of having a well-defined architecture in the software development process is that it helps in making the software safer, more reliable, easy to implement and maintain. Since a well-defined software architecture breaks down the software in smaller components, it also helps in reducing the problems generally occurring while developing complex softwares. Software architecture's influence is on engineering principle and it focusses on code, object design, boxes-and-arrows and GUI of the software.

DIFFERENT PHASES IN SOFTWARE DEVELOPMENT LIFE CYCLE

But software development lifecycle (SDLC) is not only about the design phase it consists of a total of five phases:

1. Requirement Engineering
2. Design
3. Implementation
4. Integration and Testing
5. Operations and Maintenance

We already have learned so much about the design phase. Before going to the automation techniques which are used in design phase let us first have a brief introduction about the rest of the phases.

REQUIREMENT ENGINEERING

The requirement phase of software engineering serves as the architecture of the software development. It consists of analysing the needs of the stakeholders, discussing the proper approach to develop and maintain the software and finally documenting each and every process required. This is known as requirement engineering.