Abstract

The main goal of developing an application framework on an application domain or a business domain is to facilitate development of application systems within the same domain by reusing components through customisation and not from scratch. These components would have to be analysed and designed with reusable features so that they can be reused to develop different applications in the related domain. In the case of an object-oriented application framework in the technical view, it is made up of a set of interrelated abstract (generic) and concrete classes that underlie the components of a domain model and form the domain architecture that can be reused and customised by application developers to develop a specific application system on the same domain.

An application framework basically produces reusable code or software components that reduce software development time. The aim of producing a more reliable software by reusing code is the primary aim of object-oriented programming. Instead of writing an application from scratch, application development using a framework is a better way because the reusable software components have been tested. In this research project, we will study the ways to define the abstraction details of the library systems domain to be included into a set of generic framework classes and the means to develop wizards to assist application developers in using the framework. We will also prove the use of the framework by building a sample application that is fully object-oriented, which uses an object-oriented database for data persistency. The common and variable features of library systems domain will be identified in the requirements analysis phase and designed in the software components that exist in the layered software architecture of the framework. In the layered architectural view, interrelated components are grouped together as a logical namespace identified as "package". Each package is identified with a purpose to develop a subsystem or system for a domain. By sorting out a framework into packages, application developers can quickly identify the components for reuse.

Software reuse can be rewarding in terms of quicker application system development and reliability. The convenient reuse of software packages can help to achieve that aim.