Computer Aided Software Engineering

In this chapter, we discuss about Computer Aided Software Engineering (CASE) and how the use of CASE tools helps to reduce the effort on software development and maintenance. Of late, CASE has become a much talked about topic in software industry. Software is becoming the costliest component in any computer application. Even though hardware prices keep dropping like never, and falling below even the most optimistic expectations, software prices are zooming up due to increased manpower costs. This scenario has got most managers worried. CASE tools hold the promise of reductions in software development and software maintenance costs. Therefore, no wonder that CASE tools have become a pet subject for most software project managers. For software engineers, CASE tools have taken drudgery and routine jobs out, and helped develop quality products more efficiently.

With this brief introduction to CASE tools, we first define the terminology associated with CASE and then examine the different concepts associated with CASE.

10.1 CASE AND ITS SCOPE

We need to first define what a CASE tool is and what does a CASE environment mean. A CASE tool is a generic term used to denote any form of automated support for software engineering. In a more restrictive sense, a CASE tool can mean any tool used to automate some activity associated with software development. Many CASE tools are now available. Some of these tools assist in phase-related tasks such as specification, structured analysis, design, coding, testing, etc., and others in non-phase activities such as project management and configuration management. The primary objectives of developing or using any CASE tool are:

- to increase productivity
- to help produce better quality software at lower cost.

Although individual CASE tools are useful, the true power of a tool set can be realized only when the tools are integrated into a common framework or environment. CASE tools are characterized by the stage or stages of software development life cycle on which they focus. Since different tools covering different stages share common information, it is required that they integrate through some central repository to have a consistent view of information associated with the software. This central repository is usually a data dictionary containing the definition of all composite and elementary data items. Through the central repository all the CASE tools in a CASE environment share common information among themselves. Thus a CASE environment facilitates the automation of the step-by-step methodologies for software development. In contrast to a CASE environment, a *programming environment* is an integrated collection of tools to support only the coding phase of software development. A schematic representation of a CASE environment is shown in Fig. 10.1.

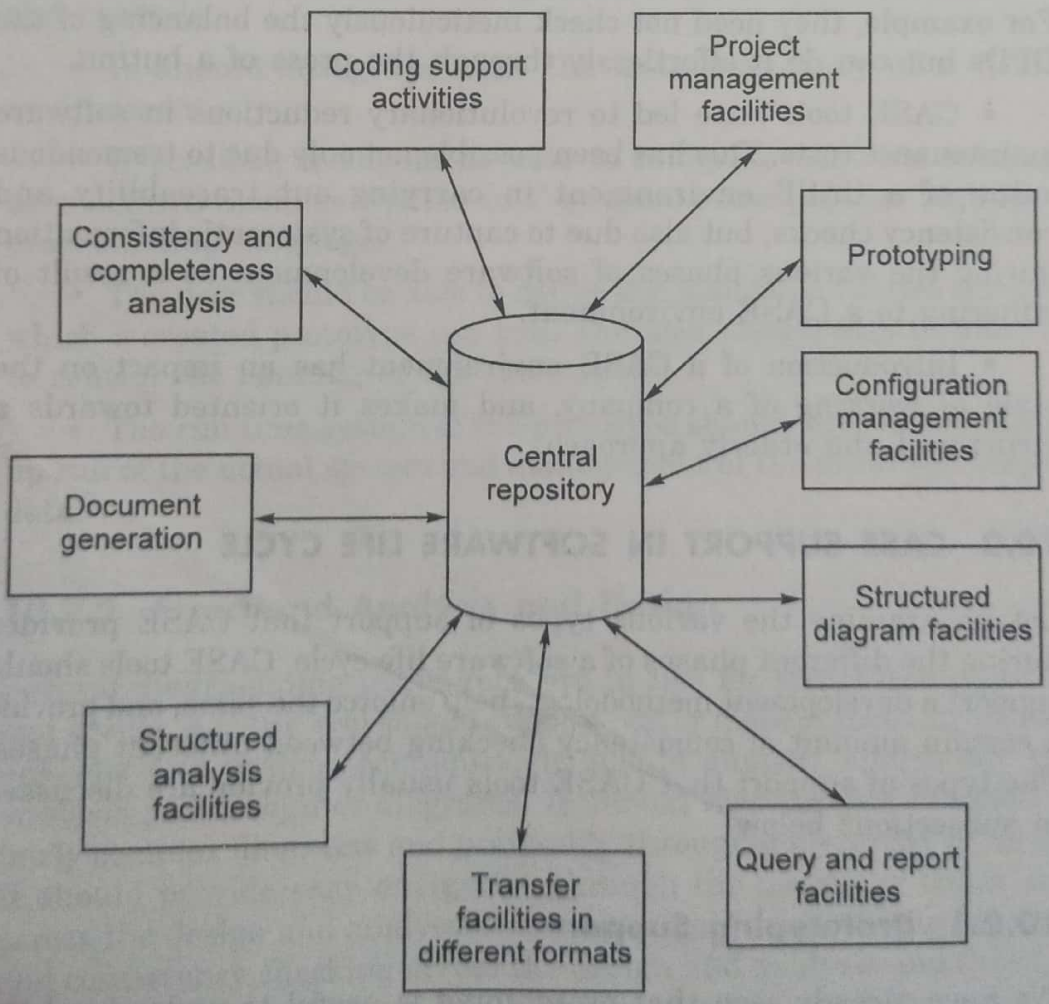

Fig. 10.1 A CASE environment.