

# Module 15

## Computer Aided Software Engineering

# Lesson 37

## Basic Ideas on CASE Tools

## Specific Instructional Objectives

At the end of this lesson the student would be able to:

- What is meant by CASE tool?
- Identify the primary reasons for using a CASE tool.
- What is meant by a CASE environment?
- Differentiate in between a CASE environment and a programming environment.
- Identify the benefits of a CASE environment.
- Identify the features of a prototyping CASE tool.
- Identify the features that a good prototyping CASE tool should support.
- Identify the supports that are typically available from CASE tools in order to perform structured analysis and software design activity.
- Identify the support that might be available from CASE tools during code generation.
- Identify the features of a test case generation CASE tool.

## CASE tool and its scope

A CASE (Computer Aided Software Engineering) tool is a generic term used to denote any form of automated support for software engineering. In a more restrictive sense, a CASE tool means any tool used to automate some activity associated with software development. Many CASE tools are available. Some of these CASE tools assist in phase related tasks such as specification, structured analysis, design, coding, testing, etc.; and others to non-phase activities such as project management and configuration management.

## Reasons for using CASE tools

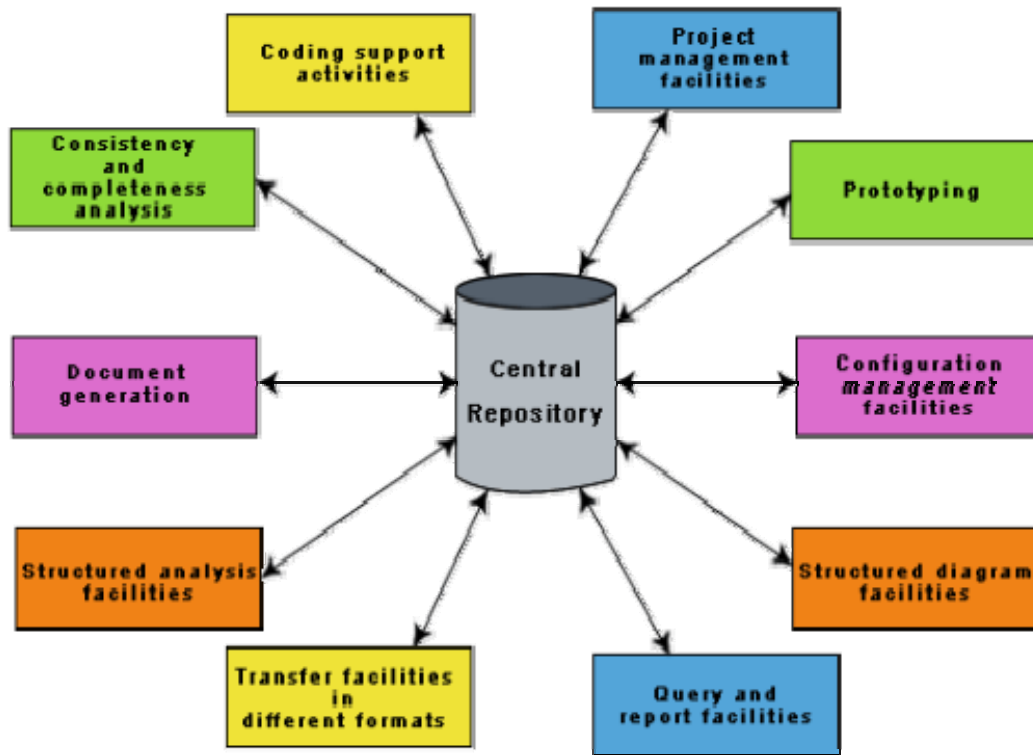
The primary reasons for using a CASE tool are:

- To increase productivity
- To help produce better quality software at lower cost

## CASE environment

Although individual CASE tools are useful, the true power of a tool set can be realized only when these set of 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 development artifacts. 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. A schematic representation of a CASE environment is shown in fig. 15.1.



**Fig. 15.1: A CASE Environment**

## CASE environment vs programming environment

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.

## Benefits of CASE

Several benefits accrue from the use of a CASE environment or even isolated CASE tools. Some of those benefits are:

- A key benefit arising out of the use of a CASE environment is cost saving through all development phases. Different studies carry out to measure the impact of CASE put the effort reduction between 30% to 40%.

- Use of CASE tools leads to considerable improvements to quality. This is mainly due to the facts that one can effortlessly iterate through the different phases of software development and the chances of human error are considerably reduced.
- CASE tools help produce high quality and consistent documents. Since the important data relating to a software product are maintained in a central repository, redundancy in the stored data is reduced and therefore chances of inconsistent documentation is reduced to a great extent.
- CASE tools take out most of the drudgery in a software engineer's work. For example, they need not check meticulously the balancing of the DFDs but can do it effortlessly through the press of a button.
- CASE tools have led to revolutionary cost saving in software maintenance efforts. This arises not only due to the tremendous value of a CASE environment in traceability and consistency checks, but also due to the systematic information capture during the various phases of software development as a result of adhering to a CASE environment.
- Introduction of a CASE environment has an impact on the style of working of a company, and makes it oriented towards the structured and orderly approach.

## Requirements of a prototyping CASE tool

Prototyping is useful to understand the requirements of complex software products, to demonstrate a concept, to market new ideas, and so on. The important features of a prototyping CASE tool are as follows:

- Define user interaction
- Define the system control flow
- Store and retrieve data required by the system
- Incorporate some processing logic

## Features of a good prototyping CASE tool

There are several stand-alone prototyping tools. But a tool that integrates with the data dictionary can make use of the entries in the data dictionary, help in populating the data dictionary and ensure the consistency between the design data and the prototype. A good prototyping tool should support the following features:

- Since one of the main uses of a prototyping CASE tool is graphical user interface (GUI) development, prototyping CASE tool should support the

user to create a GUI using a graphics editor. The user should be allowed to define all data entry forms, menus and controls.

- It should integrate with the data dictionary of a CASE environment.
- If possible, it should be able to integrate with external user defined modules written in C or some popular high level programming languages.
- The user should be able to define the sequence of states through which a created prototype can run. The user should also be allowed to control the running of the prototype.
- The run time system of prototype should support mock runs of the actual system and management of the input and output data.

## Structured analysis and design with CASE tools

Several diagramming techniques are used for structured analysis and structured design. The following supports might be available from CASE tools.

- A CASE tool should support one or more of the structured analysis and design techniques.
- It should support effortlessly drawing analysis and design diagrams.
- It should support drawing for fairly complex diagrams, preferably through a hierarchy of levels.
- The CASE tool should provide easy navigation through the different levels and through the design and analysis.
- The tool must support completeness and consistency checking across the design and analysis and through all levels of analysis hierarchy. Whenever it is possible, the system should disallow any inconsistent operation, but it may be very difficult to implement such a feature. Whenever there arises heavy computational load while consistency checking, it should be possible to temporarily disable consistency checking.

## Code generation and CASE tools

As far as code generation is concerned, the general expectation of a CASE tool is quite low. A reasonable requirement is traceability from source file to design data. More pragmatic supports expected from a CASE tool during code generation phase are the following:

- The CASE tool should support generation of module skeletons or templates in one or more popular languages. It should be possible to include copyright message, brief description of the module, author name and the date of creation in some selectable format.
- The tool should generate records, structures, class definition automatically from the contents of the data dictionary in one or more popular languages.
- It should generate database tables for relational database management systems.
- The tool should generate code for user interface from prototype definition for X window and MS window based applications.

## Test case generation CASE tool

The CASE tool for test case generation should have the following features:

- It should support both design and requirement testing.
- It should generate test set reports in ASCII format which can be directly imported into the test plan document.