Case Tools

Unit 3

Red: indicates important

Contents

- CASE tools
- Role of CASE tools in different stages of SDLC

What are CASE tools?

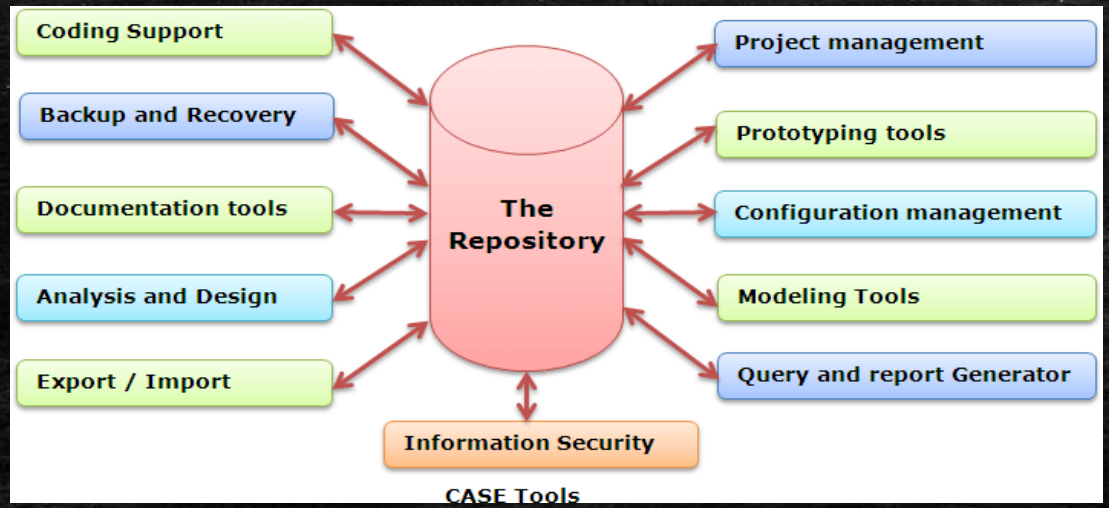
- CASE tools are software engineering tools that permit collaborative software development and maintenance.
- CASE tools support almost all the phases of the SDLC such as analysis, design etc.. Including activities like project management, configuration management etc..
- CASE tools support standard software development process like Jackson Structure Programming.
- CASE tools support the following steps for development:
  - Creation of data flow and entity models.
  - Establish relationship between requirements and models.
  - Development f top-level design.
  - Development of functional and process description.
  - Development of test cases.

- CASE tools help in generating data base tables, forms and reports, and user documentation automatically.
- CASE tools:
  - Support contemporary development of software systems, they may improve the quality of the software.
  - Helps in automating the SDLC by use of certain standard methods.
  - Create an organization wide environment that minimizes repetitive work.
  - Help developers to concentrate more on top level and more creative problem-solving tasks.
  - Support and improve the quality of documentation, testing process, project management and software maintenance.

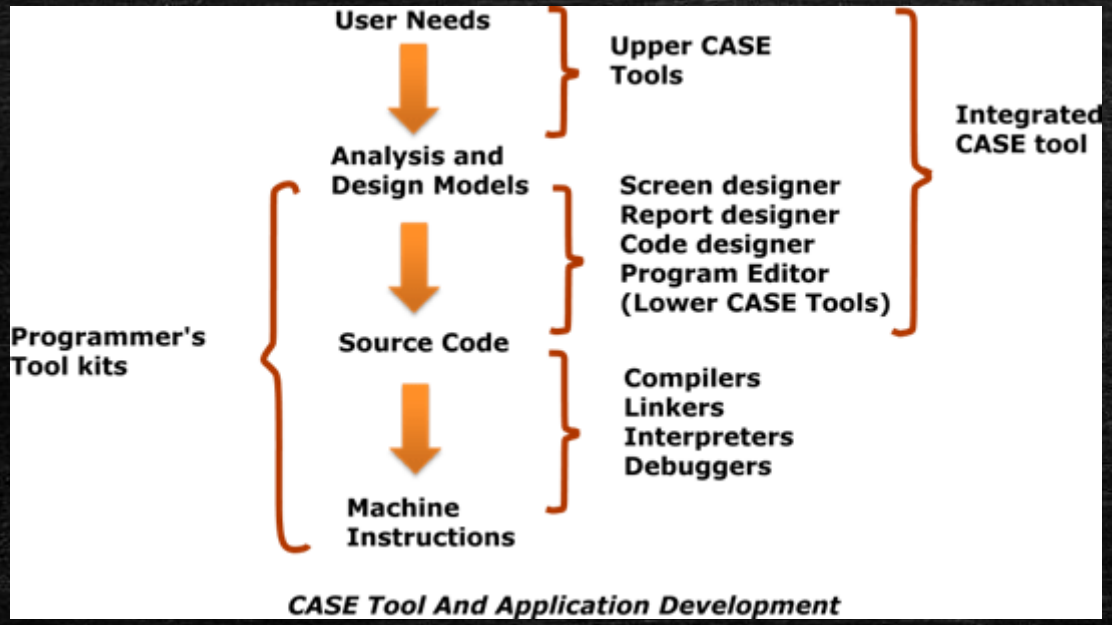
Types of tools in CASE tools

- Analysis tools
- Repository to store all diagrams, forms, models and report definitions etc..
- Diagramming tools
- Screen and report generators
- Code generators
- Documentation generators
- Reverse engineering tools(that take source code as input and produce graphical and textual representations)
- Re-engineering tools(that take source code as the input analyse it and interactively alters an existing system to improve quality and/or performance)

- Features to be supported by CASE tools :
  - It should have Security of information. The information will be visible/changeable by authorized users only.
  - Version Control for various products
  - A utility to import/export information from various external resources in a compatible fashion.
  - Process of backup and recovery as it contains very precious data.



- Case tools are categorized into the following categories:
  1. Upper CASE tools
  2. Lower CASE tools
  3. Integrated CASE tools



CASE Tool And Application Development

Upper CASE tools:

- Mainly focuses on analysis and design phases of software development.
- Includes tools for analysis modelling, reports and forms generation.

Lower CASE tools:

- It supports implementation of system development(coding phase, configuration management).
- They include tools for coding, configuration management, etc..

Integrated CASE tools:

- Also known as I-CASE.
- Helps in providing linkages between the lower and upper case tools.
- It creates a cohesive environment for software development where programming by lower CASE tools may automatically be generated for the design that has been developed in an upper CASE tool.

Need of CASE tools

- It provides integrated homogenous environment for the development of complex projects.
- It creates a shared repository of information that can be utilized to minimize the software development time.
- It also provides environment for monitoring and controlling projects in such a way that team leaders are able to manage complex projects.
- CASE tools are used to:
  - Development of cost effective software: Reduce the cost as they automate many repetitive manual tasks.
  - Minimization of development time: Reduce development time of the project as they support standardization and avoid repetition and reuse.
  - Development of better quality product: Develop complex projects with better quality as they provide greater consistency and coordination.
  - Create good quality documentation

- Create systems that are maintainable because of proper control of configuration item that support traceability requirements.
- Standardizing the software development process
- Avoiding repetition and maximizing reuse.
- Collaborative developments

- CASE tools cannot do the following:
  - Automatic development of functionally relevant system.
  - Force system analysts to follow a prescribed methodology.
  - Change the system analysis and design process.

Disadvantages of CASE tools

- Complex functionality
- Many project management problems are not amendable. Therefore CASE tools cannot be used in such situations.
- Not easily maintainable
- Require more clear definitions of user needs and requirement.
- Difficult to customize
- Require training of maintenance staff

Factors that affect deployment of CASE tools

- A successful CASE implementation requires the following considerations:
  1. Training all the users in typical CASE environment that is being deployed, also giving benefits of CASE tools.
  2. Closeness of CASE tools methodology to the SDLC.
  3. Compulsory use of CASE initially by the developers.
  4. Compatibility of CASE tools with other development platforms that are being used in an organization.
  5. Timely support of vendor with respect to various issues relating to CASE tools:
     - Low cost support
     - Easy to use and learn CASE tools having low complexity and online help.
     - Good graphic support and multiple users support.
  6. Reverse engineering support by the CASE tools: Its that a CASE tool supports complicated nature of reverse engineering.

Characteristics of a successful CASE Tools

- A standard methodology: Must support a standard software development methodology and standard modelling techniques. In the present situation most of the CASE tools are moving towards UML.
- Flexibility: Must be flexible in use of editors and other tools.
- Strong Integration: Should be integrated to support all stages. For e.g. if a change is made at any stage, say model, it should be reflected in the code documentation and all related design and other documents. This provides a cohesive environment for development.
- Integration with testing Software: Must provide interface for automatic testing tools, to take care of regression and other kind of testing software under dynamic requirement.
- Support for reverse engineering: Must be able to generate complex models from already generated code.
- Online Help: Must provide online tutorial.

- Allows integration of information
- Allow traceability
- Helps improving quality of software
- Reduce cost
- Checks for inconsistency of information
- Provides tools for configuration management and project management

Case Software Development Environment

- CASE tools support various activities during software development.
- Some of the functional features that is provided by CASE tools for software development are:
  1. Creating SRS
  2. Creation of design specification
  3. Creation of cross references
  4. Verifying/ Analyzing the relationship between requirement and design
  5. Perform project and configuration management
  6. Build system prototypes
  7. Containing code and associated documents
  8. Validation and Verification

- Major features that should be supported by CASE tools :
  - Strong visual support
  - Predicting and reporting errors
  - Generate content repository
  - Support structured methodology
  - Integration of various life cycle stages
  - Consistent information transfer across SDLC stages
  - Automating coding/ prototype generation
- Present CASE tools support UML(Unified Model Language)

- CASE tools are useful for design, development and implementation of the web site development.
- Web engineering requires the tools in the following categories:
  - Site content management tools
  - Site version control tools
  - Server management tools
  - Site optimization tools
  - Web authoring and deployment tools
  - Site testing tools(load and performance testing)
  - Link checkers
  - Program checkers
  - Web security test tools

CASE tools and Requirements Engineering

- A good and effective requirement engineering tool needs to include the best practices of requirements definition and management.
- It should be highly iterative with the goal of establishing managed and effective communication and collaboration.
- Following features of CASE tools:
  - A dynamic, rich editing environment for team members to capture and manage requirements.
  - To create a centralized repository
  - To create a task-driven workflow to do change management and defect tracking.
- Four - step process in requirements engineering for CASE :
  1. Requirements Elicitation
  2. Specification
  3. Validation
  4. Requirement Management

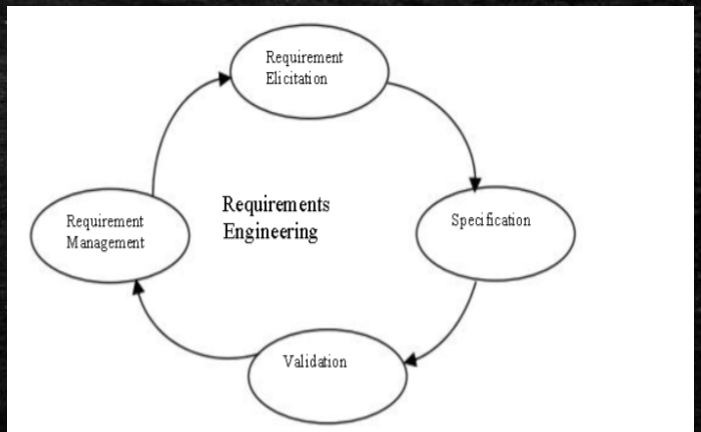


Figure 3.3: The four-step requirement engineering process

Requirements Elicitation :

- A simple technique for requirements elicitation is to ask “why”.
- CASE tools support a dynamic, rich editing environment for team members to capture and manage requirements.
- Requirements elicitation is the stage where CASE tools are least helpful as this stage requires interaction with users.
- Features available for requirements elicitation are:
  - Reusable requirements and design templates for various types of system
  - Keeps track of systems attributes like performance and security
  - Supports a common vocabulary of user-defined terms that can be automatically highlighted to be part of glossary
  - Supports features for the assessment of quality of requirements
  - Handles separate glossary for ambiguous terms that are flagged for additional clarification.

- What do we expect from the tool?
  - It should have rich support for documentation that is easily understandable to stakeholders and development teams.
  - It should have the capability of tracking of requirements during various SDLC systems.
  - It should help in the development of standard technical and business terminology for end clients.

Software Analysis and Specification :

- Reason for documenting requirements is to remove ambiguity of information.
- A good requirement specification is testable for each requirement.
- Features supported by CASE tools for design specification is that the design and implementation should be traceable to requirements. Good way to do this is to support a label or a tag to the requirements.

- Additional features the CASE should support for specifications :
  - Must have features for storing and documenting of requirements.
  - Enable creation of models that are critical to the development of functional requirements.
  - Allow development of test cases that enable the verification of requirements and their associated dependencies.
  - Test cases that help in troubleshooting the correlation between business requirements and existing system constraints.
- What do we expect from the tool?
  - It should allow development of a labeled requirements document that helps in traceability of requirements.
  - It should allow both functional and non-functional requirements with related quality attributes to be made.
  - We should be able to develop the associated models.

Validation :

- Features available for validation are :
  - it allows collaborative yet customizable workflows for the software development team members.
  - It facilitates approvals and electronic signatures to facilitate audit trails.
  - Assigning owner of requirement may be useful if any quality attributes may need changes. Hence, a prioritized validated documented and approved requirements can be obtained.

Managing Requirements :

- The requirements document should have visibility and help in controlling the software delivery process.
- Features available in CASE tools for Managing requirements:
  - Estimation of efforts and costs
  - Specification of project schedule such as deadline, staff requirements and other constraints.
  - Specification of quality parameters.

- Software Change Management :If an incomplete or ambiguous requirement is detected in the early analysis phase, it can be changed easily with minimum cost.
- Once they are converted to baselines after requirements validation such changes should be controlled.
- Major requirements of change management:
  - Any change to the requirements in baselines should follow a process which is defined as the ability to track software requirements and their associated models/ documents. This helps in finding the components that will be affected due to the change. This also helps tracking and managing change. The whole process starts with labeling the requirements properly.
  - CASE tools store requirement baselines which include type, status, priority and change history of a software item. This traceability is bi-directional in nature.
  - CASE tools are familiar with environments like Microsoft Word, with added communication methods , such as email. Thus CASE is very important tool for requirements engineering.

CASE tools and design and implementation

- CASE tools support the analysis and design phases of software development.
- Tools supported by the design tools are:
  - Structured Chart
  - Program Document Language (PDL)
  - Optimization of ER and other models
  - Flow charts
  - Database design tools
  - File design tools
- It support standard representation of program architecture.
- It contains testing items related to the design and debugging.
- Automatic support for maintenance is available if any of requirements or design items is modified using these diagrams.

- It also provides error-checking stages.
- CASE tools have strong support for models. Helps in resolving conflicts and ambiguity among the models and helps in optimizing them to create excellent design architecture and process implementation.
- Model Vs Code :
  - Model enhances communication, as its more pictorial and less code.
  - A model : helps the user, reduces the cost of project, and conveys large spectrum of information.
  - Standard model represents the following :
    - System architecture
    - Software dependencies
    - Flow of information
    - Database organization and structure
    - Deployment configuration, etc..

- Models help in better planning, reduction of risk and controls complexity.
- Characteristics of a good modeling tool :
  - CASE tools provide continuously synchronized models and code.
  - It helps in consistent understanding of information and documentation.
  - It helps other software developers to understand the portions and relationships to portions other team members are working on.
  - It helps in management of source code through a visual model.
  - Modeling can help in creating good patterns including modeling patterns, language specific patterns and basic patterns.
  - It also facilitates reverse engineering.
- CASE Repository:
  - It stores software system information. It includes analysis and design specifications and helps in analyzing these requirements and converting them into program code and documentation.

- Contents of a CASE repository:
  1. Data : information and entities/object relationship attributes, etc..
  2. Process : Support structured methodology, link to external entities, document structured software development process standards.
  3. Models : Flow models, state models, data models, UML document etc..
  4. Rules/Constraints : Business rules, consistency constraints, legal constraints.
- CASE repository has two primary segments :
  1. Information repository : It includes information about an organization's business information and its applications.
  2. Data dictionary : contains all the data definitions for all organizational applications, along with cross-referencing if any. Its entries have a standard definition namely, element name and alias; textual description of the element; related elements; element type and format; range of acceptable values and other information unique to the proper processing of the element.

- CASE tools manage and control access to repository. Such information can also be stored in corporate database.
- It assists the project management tasks, aids in software reusability by enabling software modules in a manner so that they can be used again.
- Implementation tools and CASE :
- It provides following features for implementation :
  - Diagramming tools enable visual representation of a system and its components.
  - Represents process flows.
  - Helps in implementing the data structures and program structures.
  - Support automatic creation of system forms and reports.
  - Ready prototype generation.
  - Create both technical and user documentation.
  - Create master templates used to verify documentation and its conformance to all stages of the SDLC.
  - Enable automatic generation of program and database from the design documents, diagrams, forms and reports stored in the repository.

Software Testing

- CASE tools also support software testing.
- Testing tools can help in automated unit testing, functional regression testing and performance testing.
- Testing tool ensures the high visibility of test information, types of common defects, and application readiness.
- Feature needed for testing :
  - It must support all testing phases, namely, plan, manage and execute all types of tests namely, functional, performance, integration, regression testing from the same interface.
  - It should integrate with other third party testing tools.
  - It should support local and remote test execution.
  - It should help and establish and manage traceability by linking requirements to test cases.
  - It should create a log of test run.
  - It should output meaningful reports on test completeness, test analysis.
  - It may provide automated testing.

Software Quality and CASE tools

- Software quality is sacrificed by many developers for faster development and low cost.
- A good quality product actually enhances the speed of software development. It reduces the cost and allows enhancement and functionality with ease as it's a better structured product.
- Software quality involves functionality for software usability, reliability, performance, scalability, support and security.
- Integrated CASE tools:
  - It helps the development of quality product as they support standard methodology and process of software development.
  - Supports an exhaustive change management process.
  - Contains easy to use visual modeling tools incorporating continuous quality assurance.
- Quality is essential in all life cycle phases.

Analysis :

- CASE tools help in reflecting the system requirements clearly, accurately and in a simple way.
- It supports the requirements analysis.
- It helps in ambiguity resolution of the requirements. Thus making high quality requirements.

Design:

- Prime focus of the quality starts with the testing of the architecture of the software.
- CASE tools help in detecting, isolating and resolving structure deficiency during the design process.
- CASE tools provides a set of automatic run time language tools for development of reliable and high performance applications.

Testing :

- Functionality and performance testing is done to ensure high quality product.
- CASE support automated testing tools that help in testing the software, this in turn, helps in improving the quality of software.
- CASE tools enhances the speed breadth and reliability of these design procedures.
- Design tools are very important in web based system where scalabilty and reliability are two major issues of design.

Deployment :

- Deployment is the phase where the system is made operational.
- System failure should not result in a complete failure of the software on restart.
- It also support configuration management to help any kind of change.

- Quality is teamwork : It involves integration of workflow of various individuals. It establishes a traceability and communication of information, all this can be done by sharing workload documents keeping their configuration items.

Software Configuration Management

- SCM is extremely important from the view of deployment of software applications.
- SCM controls deployment of new software versions.
- SCM is integrated with an automated solution that manages distributed deployment. It helps in companies to bring new releases more efficiently and effectively.
- It also reduces cost, risk and accelerates time.
- It should have facilities of automatic version control, access control, automatic re-building of software, build audit, maintenance and deployment.
- SCM should have following facilities:
  - Creation of configuration
  - This documents a software build and enables versions to be reproduced on demand
  - Configuration lookup scheme that enables only the changed files to be rebuilt. Thus, entire application need not be rebuilt.

- Dependency detection features even hidden dependencies, thus ensuring correct behavior of the software in partial rebuilding.
- Ability for team members to share existing objects, thus saving time of the team members.

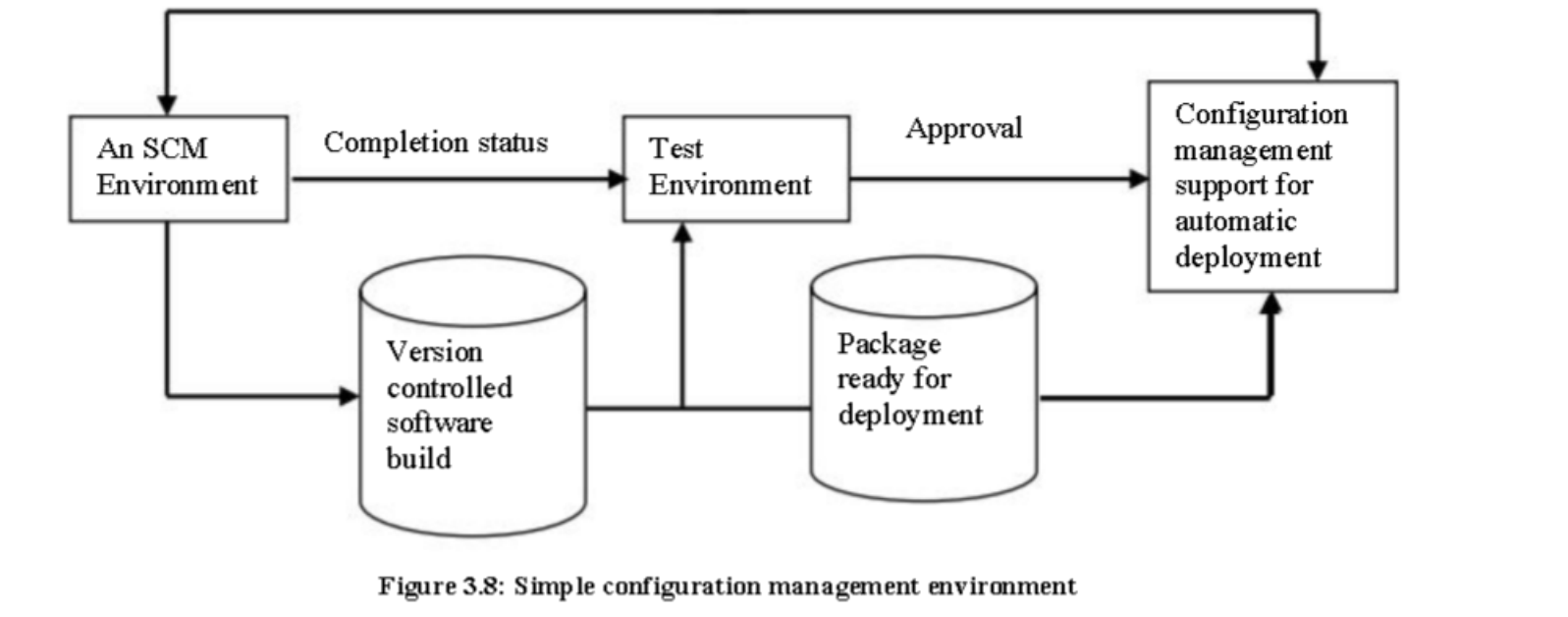


Figure 3.8: Simple configuration management environment

Software Project Management and CASE tools

- Development of a software is a team activity. Development is too risky, or too large for a single developer to deliver the software in a stipulated time and quality frame.
- CASE tools help in management of teams and projects.
- Features of CASE in this aspect:
  - Sharing and securing the software using username and passwords.
  - Allowing reading of project related documents
  - Allowing exclusive editing of documents
  - Linking the document for sharing
  - You can read change requests for yourself and act on them accordingly.
  - Setting of revision table so that versioning can be done.
  - Addition or deletion of files from repository is indicated.
  - Any updating of files in repository is automatically available to users.
  - Conflicts between versions are reported and avoided.

- Differences between versions can be visualized.
- The linked folder, topics and change requests to an item can be created and these items can be accessed if needed.
- It must have reporting capabilities of information.

- Project management tools provide the following benefits :
  - They allow control of projects through tasks so control complexity.
  - It allows tracking of project events and milestones.
  - The progress can be monitored using Gantt chart.
  - Web based project related information can be provided.
  - Automatic notifications and mails can be generated.

- Features of a Project Management software :
  - It should support drawing of schedules using PERT and GANTT chart.
  - It should be easy to use such that tasks can be entered easily, the links among the tasks should be easily desirable.

- Milestones and critical path should be highlighted.
- It should support editing capabilities like adding/ deleting/ moving tasks.
- Should map timeline information against a calendar.
- Should allow marking of durations for each task graphically.
- It should provide views tasks, resources, or resource usage by task.
- Should be useable on network and should be able to share information through network.