SEPM

ASSIGNMENT 1

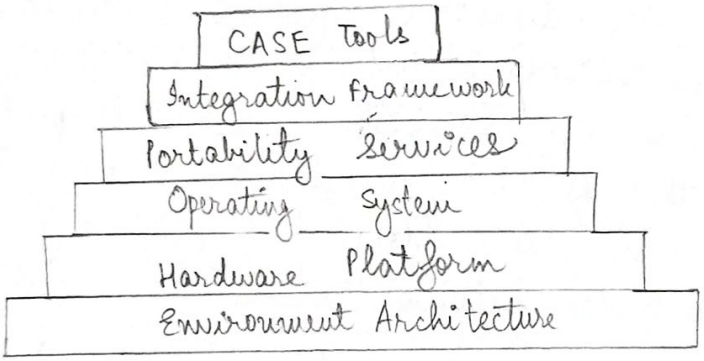
Submitted by:
Srividya Krishnakumar
CS6A
55

CASE BUILDING BLOCKS

Computer-aided software engineering (CASE) tools assist software engineering managers & practitioners in every activity associated with the software process.

CASE Building Blocks.

- CASE tools.
- Integration framework (specialized programs allowing CASE tools to communicate with one another).
- Portability services (allow CASE tools & their integration framework to migrate across different OSs & hardware platforms without significant adoptive maintenance).
- OS (database & object management services).
- Hardware platform.
- Environmental architecture (hardware & system support).



CASE Tool Components.

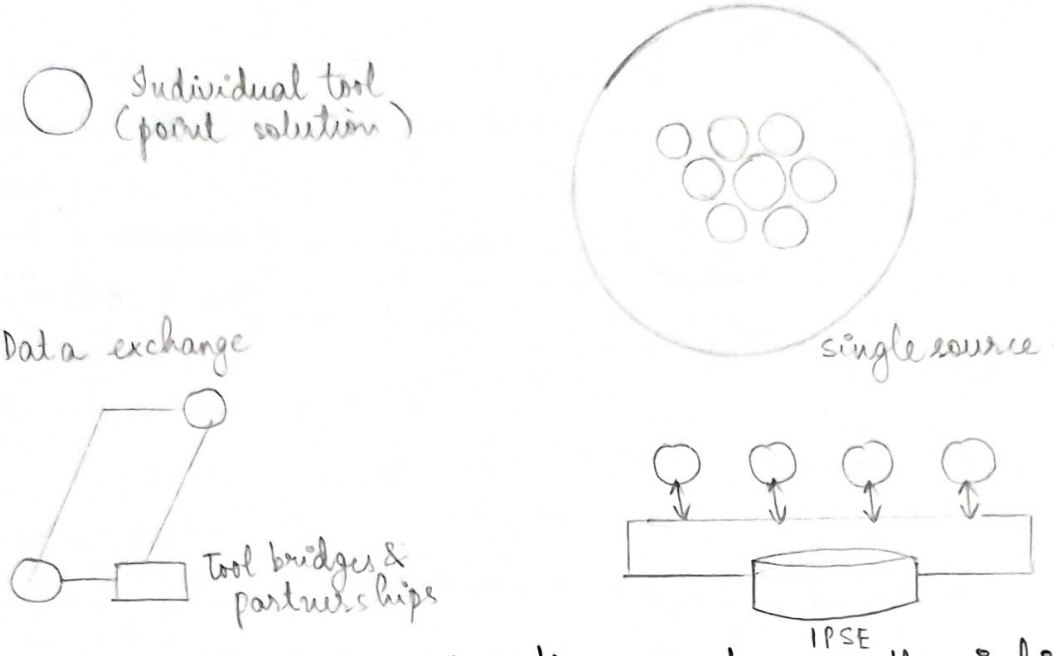
- Integration framework.
- Specialized programs allowing CASE tools to communicate.
- Portability services.
- OS
- Database & object mgmt services.
- Hardware platform.

The environment architecture, composed of the hardware platform and system support, lays the ground work for CASE. But the case environment itself demands other building blocks.

A set of portability services provides a bridge between CASE tools & their integration framework & the environment architecture.

The integration framework is a collection of specialized programs that enable individual CASE tools to communicate with one another, to create a project database & to exhibit the same look & feel to the end user.

Point-solution CASE tools can provide substantial individual benefit, but a software team needs tools that talk to one another. Integrated tools help the team develop, organize & control work products.



At the low end of the integration spectrum is the individual (point solution) tool. When individual tools provide facilities for data exchange, the integration level is improved slightly. Such tools produce output in a standard format that should be compatible with other tools that can read the format. In some cases, the builders of complementary CASE tools work together to form a bridge between the tools.

Using this approach, the synergy between the tools can produce end products that would be difficult to create using either tool separately.

Single-source integration occurs when a single CASE tools vendor integrates a no of diff. tools & sells them as a package. Although this approach is quite effective, the closed architecture of most single source environments produces easy addition of tools from other vendors.

At the high end of the integration spectrum is the integrated project support environment (IPSE). CASE tools vendors use IPSE standards to build tools that'll be compatible with the IPSE & therefore compatible with one another.

TAXONOMY OF CASE TOOLS

CASE tools can be classified by function, by their role as instruments for managers or technical people, by their use in the various steps of the software engineering process, by the environment architecture that supports them, or even by their origin or cost. The taxonomy presented here uses functions as a primary criterion.

- Business process engineering tools – represent business data objects, their relationships & flow of the data between company business areas.
- Process modeling & management tools – represent key elements of processes & provide links to other tools that provide support to defined process activities.
- Project planning tools – used for cost & effort estimation, & project scheduling.
- Risk analysis tools – help managers build risk tables by providing detailed guidance in the identification & analysis of risks.
- Requirements tracing tools – provide systematic database-like approach to tracking requirement status beginning with specification.
- Metrics & management tools – management oriented tools capture project-specific metrics that provide an overall indication of productivity or quality, technically oriented metrics determine metrics that provide greater insight into the quality of design or code.
- Documentation tools – provide opportunities for improved productivity by reducing the amount of time needed to produce work products.
- System software tools – network system software, object management services, distributed component support, & communications software.
- Quality assurance tools – metrics tools that audit source code to determine compliance with language standards or tools that extract metrics to project the quality of software being built.
- Database management tools – RDMS & OODMS serve as the foundation for the establishment of the CASE repository.
- Software configuration management tools – uses the CASE repository to assist with all SCM tasks.
- Analysis & design tools – enable the software engineer to create analysis & design models of the system to be built, perform consistency checking b/w models.
- PRO/SIM tools – prototyping & simulation tools provide software engineers with ability to predict the behavior of real-time systems before they are built & the creation of interface mockups for customer review.

- Interface design and development tools - toolkits of interface components, often part environment with a GUI to allow rapid prototyping of UI designs.
- Prototyping tools - enable rapid definition of screen layouts, data design & report generation.
- Programming tools - compilers, editors, debuggers, OO programming environments, 4th generation languages, graphical pgming environments, application generators & DB query generators.
- Web Development tools - assist with the generation of web page text, graphics, forms, scripts, applets, etc.
- Integration & testing tools - In their directory of software testing tools, Software Quality Engineering (SQE 95) defines the foll. testing tools categories: Data acquisition, static measurement, dynamic measurement, simulation, test management, cross-functional tools.
- Static analysis tools - code-based testing tools, specialized testing languages, requirements-based testing tools.
- Dynamic analysis tools - intrusive tools modify source code by inserting probes to check path coverage, assertions, or execution flow; non-intrusive tools use a separate hardware processor running in parallel with processor containing the program being tested.
- Test management tools - co-ordinate regression training, compare actual & expected output, conduct batch testing, & serve as generic test drivers.
- Client/Server testing tools - exercise the GUI & network communications requirements for the client & server.
- Re-engineering tools:
  → Reverse engineering to specification tools - generate analysis & design models from source code.
  → Code restructuring & analysis tools - analyze program syntax, generate control flow graph, & automatically generates a structured pgm.
  → Online system reengineering tools - used to modify online DBMS.

INTEGRATED CASE ENVIRONMENT

Integration of a variety of tools & information that enables closure of communication among tools, between people & across the software process.

The benefits of integrated CASE (I-CASE) include:

(i) Smooth transfer of information from one tool to another & one software engineering step to the next

(ii) A reduction in the effort required to perform umbrella activities such as software configuration management, quality assurance & document production.

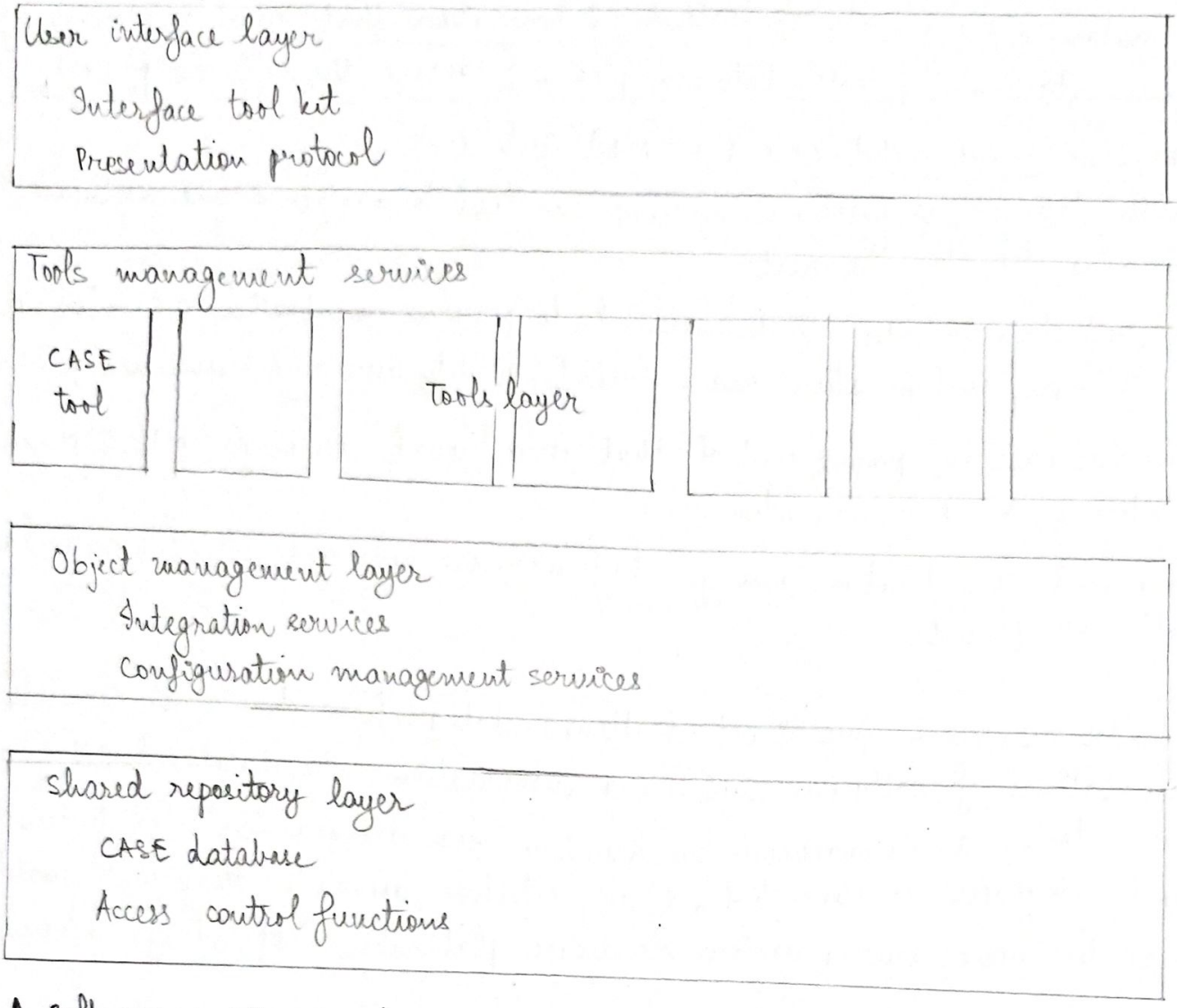
(iii) An increase in project control that is achieved through better planning, monitoring & communication.

(iv) Improved co-ordination among staff members who are working on a large software project.

But I-CASE also poses significant challenges. Integration demands consistent representations of software engineering information, standardized interfaces between tools, a homogeneous mechanism for communication between the software engineer & each tool, & an effective approach that will enable I-CASE to move among various hardware platforms & operating systems.

The term integration implies both combination & closure. I-CASE combines a variety of different tools & a spectrum of information in a way that enables closure of communication among tools, between people, & across software process. Tools are integrated so that software engineering info. is available to each tool that needs it; usage is integrated so that a common look & feel is provided for all tools; a development philosophy is integrated, implying a standardized software engineering approach that implies modern practice & proven methods.

Integration Framework Design.

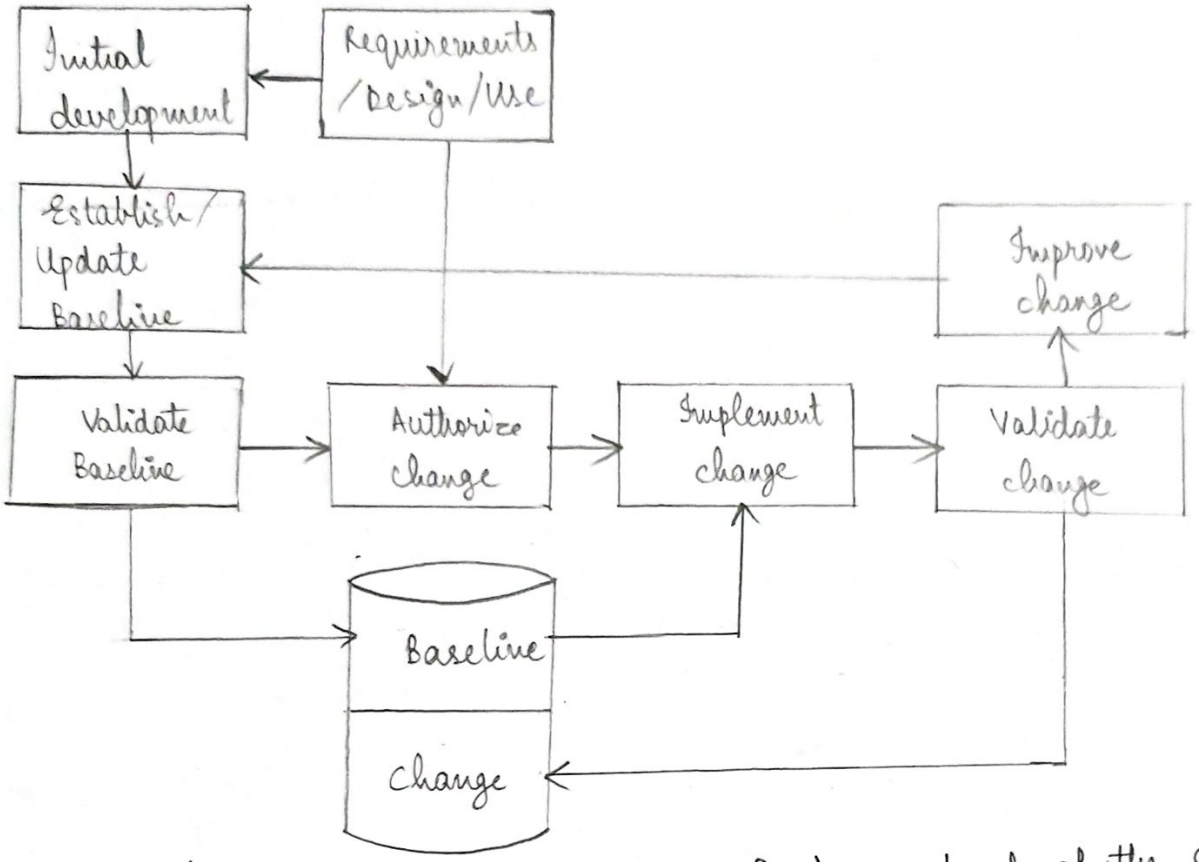


A software engineering team uses CASE tools, corresponding methods & a proper framework to create a pool of software engineering information. The integration framework facilitates transfer of information into & out of the pool.

SOFTWARE CONFIGURATION MANAGEMENT

Software configuration management (SCM) is a set of management disciplines within the software engineering process to develop a baseline.

SCM encompasses the disciplines & techniques of initiating, evaluating & controlling change to software products during & after the software engineering process.



Configuration managers are responsible for keeping track of the differences b/w software versions, for ensuring that new versions are derived in a controlled way & for releasing new versions to the right customers at the right time.

SCM Activities.

- Configuration item identification
- Promotion management
- Release management
- Branch Management
- Variant management
- change management

SCM Roles

- Configuration manager.
- Change control board member.
- Developer.
- Auditor.

SCM Directories

- Programmer's Directory.
- Master directory
- Software repository.