9 SOFTWARE CONFIGURATION MANAGEMENT

Unit - V

Syllabus

Software configuration management : Software Configuration Management, The SCM Repository, The SCM Process, Configuration Management for any suitable software system.

Suggested Free Open Source Tools : CF Engine configuration Tool, Puppet configuration Tool.

9.1 Software Configuration Management

University Questions

Q. What is Software Configuration Management ? **SPPU - May 13, May 14, Dec. 18, 3 Marks**

Q. What do you understand by Software Configuration Management (SCM) ? **SPPU - May 19, 4 Marks**

- Basically the output of any software process contains the information based on various inputs. These output can be broadly divided into three important categories as follows :
  - The computer programs
  - The work products and
  - The data that consist of the information produced.
- All these information parts collectively called as software configuration.
- If each configuration item simply led to other items, little confusion would result. During the process, change takes place which may be unfortunate. But change can occur any time and for any undefined reason. The change is the only constant.
- The first law of system engineering says that "No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle".
- Following are four basic sources of change :
  - Every new business conditions and market conditions may cause the change in product requirement and business rules definitions.
  - Customers may require change in data presented by the information system. The customers may also demand various functionality delivered by their product of interest. The customers seek some new services. All these changes are customer oriented.
  - As the business grows or its downsizing may cause various changes in priorities of the project and also restructuring of the team happens.
  - The budget and time scheduling constraints are also important factors for change in system or product.

- SCM is a set of activities used for managing the change during the life cycle of computer software. The software configuration management can also be considered as a software quality assurance activity during the development process.

9.1.1 SCM Basics

University Questions

Q. What are elements that exist when an effective SCM system is implemented? Explain each in detail.

SPPU - May 12, 5 Marks

Q. Write short note on : Elements of a configuration management system.

SPPU - Dec. 14, 6 Marks

- Four basic elements that should exist when a configuration management system is developed :
  1. **Component elements :** The tools in the file management system uses the software configuration item.
  2. **Process elements :** The process elements or the procedures uses effective approach towards the change management in engineering and use of computer software.
  3. **Construction elements :** The automated tools are used in construction or the development process and ensuring the validated components should be assembled.
  4. **Human elements :** In order to make the effective use of SCM, the team makes the use of various tools and process feature.
- These elements are not mutually exclusive. For example, component elements work in conjunction with construction elements as the software process evolves. Process elements guide many human activities that are related to SCM and might therefore be considered human elements as well.

9.1.2 Baselines

- The change is the only constant in software development life cycle. The customer want to modify the requirement as the model gets ready. Since in the beginning, even customer is not fully aware of the product requirement. As the development begins, customers need lot of changes in the requirement.
- Once customers modify their requirement, it is now manager who modifies the project strategy.
- Actually as time passes, all the people involved in the product development process come to know exact need, the approach and how it will be done in the prescribed amount of time.
- The additional knowledge is required to know the exact requirement. It is very difficult for most of the software engineers to accept this statement that most of the changes are justified.
- The baseline is SCM that help in development process without affecting much the schedule and control the changes.
- The IEEE provides a baseline as follows :

  "A specification and requirement that is agreed upon between customer and developer is a basis for the product development and these requirements can be changed only through change control procedures".

9.1.3 Software Configuration Items (SCI)

- Basically SCI i.e. software configuration item is the integral part of software engineering development process. It is a part of large specification or we can say that one test case among large suite of test cases.

- In fact the SCI is a document or the program component like C++ functions or a Java applet.

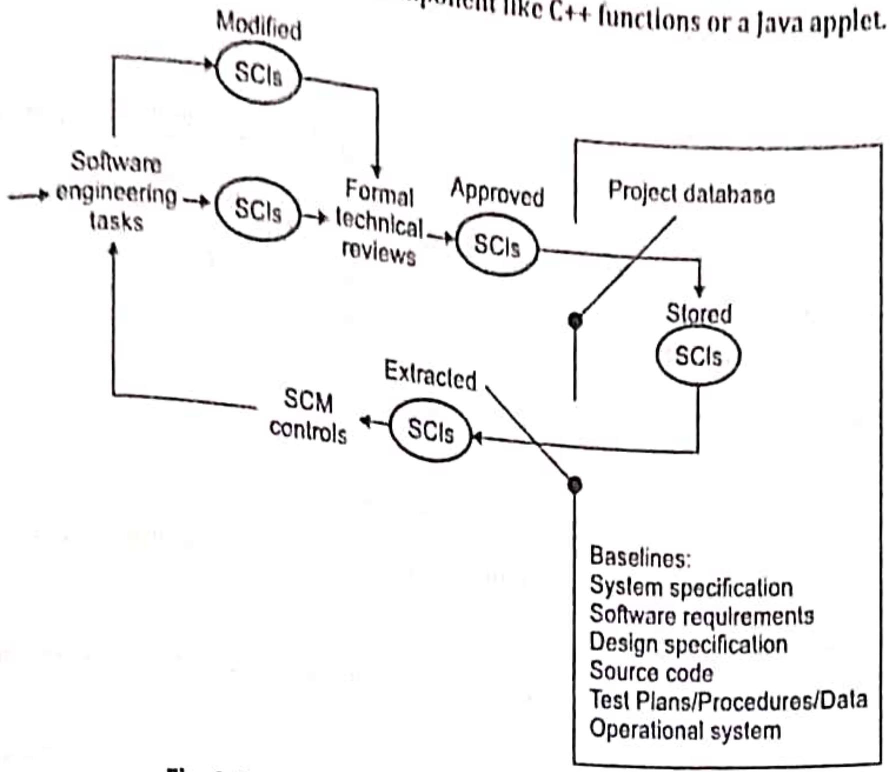


Fig. 9.1.1 : Baselined SCIs and the project database

- Most of the organizations use software tools under configuration control to help development process. In fact the editors, compilers, browsers and various automated tools are the integral part of software configuration.
- In fact the SCIs are catalogued in the project database with a single name and they form configuration objects. These objects are configuration object and it has a name, attribute and it has certain relationship with other configuration objects.
- Referring to Fig. 9.1.2, the configuration objects, Design Specification, Data Model, Component N, Source Code and Test Specification are each defined separately.

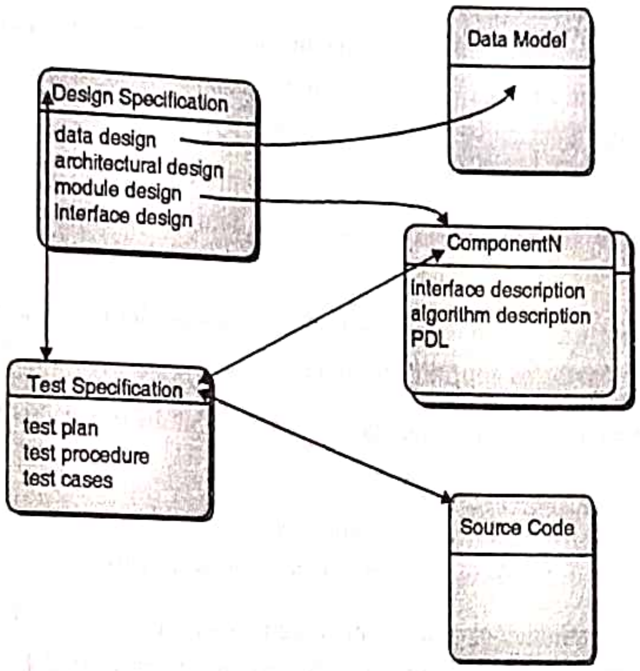


Fig. 9.1.2 : Configuration objects

9.2 SCM Repository

University Questions

Q. What is SCM repository in detail? **SPPU - May 15, Dec. 15, May 16, May 18, 5 Marks**

Q. What is software SCM repository? **SPPU - Dec. 18, 3 Marks**

- In the early days of software engineering, software configuration items were maintained as paper documents (or punched computer cards), placed in file folders or three-ring binders, and stored in metal cabinets.
- This approach was problematic for many reasons :
  - To find a SCI is a difficult task when it is needed.
  - To determine which items were changed and by whom. It is always challenging.
  - To develop a new version from an existing program is prone to errors and time consuming too.
  - To describe detailed relationship is actually impossible.
- As discussed earlier that SCIs are catalogued in the project database with a single name, they reside in the repository.
- The repository is a database that stores the software engineering information. The software developer or engineer interacts with the repository by using built in tools within repository.

9.2.1 The Role of the Repository

University Question

Q. Discuss role of SCM repository. **SPPU - May 15, Dec. 15, 5 Marks**

The SCM repository is the set of mechanisms and data structures that allow a software team to manage change in an effective manner. The repository will perform all the fundamental operations of database management system and in addition it will perform the following operations :

- **Data integrity :** Data integrity validates all the entries to the repository and make sure that the consistency among various objects intact and takes care of all the modifications takes place. It will also ensure cascading modifications i.e. change in one object causes change in a dependent object also.
- **Information sharing :** It is mechanism for sharing information among various developers and between various tools. These tools manage and control multi-user access to data, and locks or unlock objects to retain its consistency.
- **Tool integration :** Tool integration is a data model that can be used by many software engineering tools to control access to the data, and performs appropriate configuration management functions.
- **Data integration :** Data integration provides database functions that allow various SCM tasks to be performed on one or more SCIs.
- **Document standardization :** Document standardization is an important task for defining the objects. This standardization is a good approach for making software engineering documents.
- **Methodology enforcement :** Methodology enforcement defines an (E-R model) i.e. entity-relationship model available in the databases i.e. repository. This model may be used as a process model for software engineering. It is mandatory that the relationships and objects must define before building the contents of the repository.

- To achieve these functions, the database is used as a meta-model. This meta-model exhibits the information and how this information is stored in the databases i.e. repository. This meta-model also checks data security and integrity.

9.2.2 General Features and Content

University Question

Q. Discuss features of SCM repository.

SPPU - May 15, Dec. 15, 5 Marks

- The contents of databases and features of databases are considered from two perspective :
  - o What data is to be stored in the databases
  - o What services are provided by the databases.
- A detailed breakdown of types of representations, documents, and work products that are stored in the repository is presented in Fig. 9.2.1.

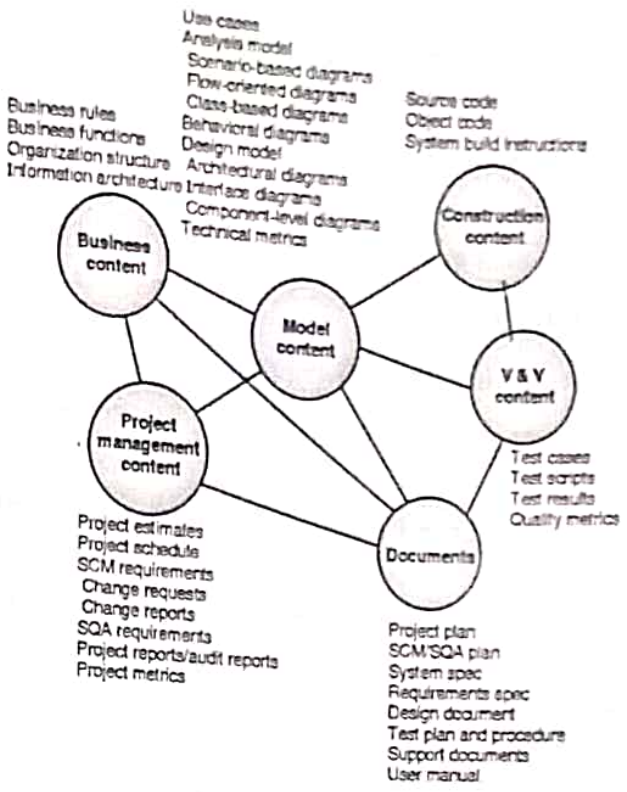


Fig. 9.2.1 : Content of the repository

- A robust repository provides two different classes of services :
  1. The same types of services that might be expected from any sophisticated database management system and
  2. Services that is specific to the software engineering environment.
- A repository that serves a software engineering team should :
  - o Integrate with or directly support process management functions;
  - o Support specific rules that govern the SCM function and the data maintained within the repository;
  - o Provide an interface to other software engineering tools; and
  - o Accommodate storage of sophisticated data objects (e.g., text, graphics, video, audio).

9.2.3 SCM Features

University Question

SPPU - Dec. 18, 6 Marks

Q. Explain the features of tool set supporting SCM repository.

- To support SCM, the repository must have a tool set that provides support for the following features :

  1. Versioning
  2. Dependency tracking and change management
  3. Requirements tracing
  4. Configuration management
  5. Audit trails

1. Versioning

- In the development process, as the project progresses, various versions of the product will be developed and the database will save all these versions.
- The repository will keep track of these versions in order to make effective management of the product delivery or the product releases.
- The history of all releases will be used by the developers to make effective testing and debugging.

2. Dependency tracking and change management

- The databases or the repository stores various relationships among the configuration objects.
- The relationships may be between entities and processes, or between application design and component design and between all the design elements etc.
- Some relationships are optional and some of the relationships are mandatory relationships that have various dependencies.
- So to keep the track of previous history and relationships is very important for the consistency of the information present in the databases. The new releases of the final product are also dependent on the history stored in the repository. This will be useful in the improvement process.

3. Requirements tracing

- The requirement tracing will provide the ability to trace all the design components and releases. This tracing results from a specific requirement called as forward tracing.
- It will also useful in finding that which requirements are fulfilled properly from the ready product. This is called as backward tracing.

4. Configuration management

- The configuration management is a facility to keep the track of various configurations under development.
- The series of configurations is used as the project milestones and future releases.

5. Audit trails

- The audit trails keeps additional information (i.e. meta-data) like the changes made by whom and when. Also it stores the reasons why changes have been made.
- The source of each change in the development must be stored in the repository.

.3 SCM Process

University Questions

Q. Explain SCM process in detail. **SPPU - Dec. 12, May 16, May 19, 6 Marks**

Q. Which are the layers of SCM process ? Explain each in detail. **SPPU - May 18, Dec. 19, 6 Marks**

The SCM (Software Configuration Management) process consists of series of task to monitor the control on changes being occurred. The main objectives of these tasks are as follows :

1. To identify all individual items that can define software configuration collectively.
2. Manage the changes taking place in various individual items.
3. To handle different versions or releases of product.
4. To maintain the quality of the software under construction over the period of time.

A process that achieves these objectives must be characterized in a manner that enables a software team to develop answers to a set of complex questions :

- o How does a software team identify the discrete elements of a software configuration?
- o How an organization manages the changes being done in existing release or the existing version? The modification should be incorporated efficiently.
- o How an organization keeps the track and control of new releases?
- o In an organization, who is responsible for authorizing all these changes?
- o The mechanism used to let others know the changes taking place and implemented?

These questions lead us to the definition of five SCM tasks - identification, version control and change control, configuration auditing, and reporting, as illustrated in Fig. 9.3.1.

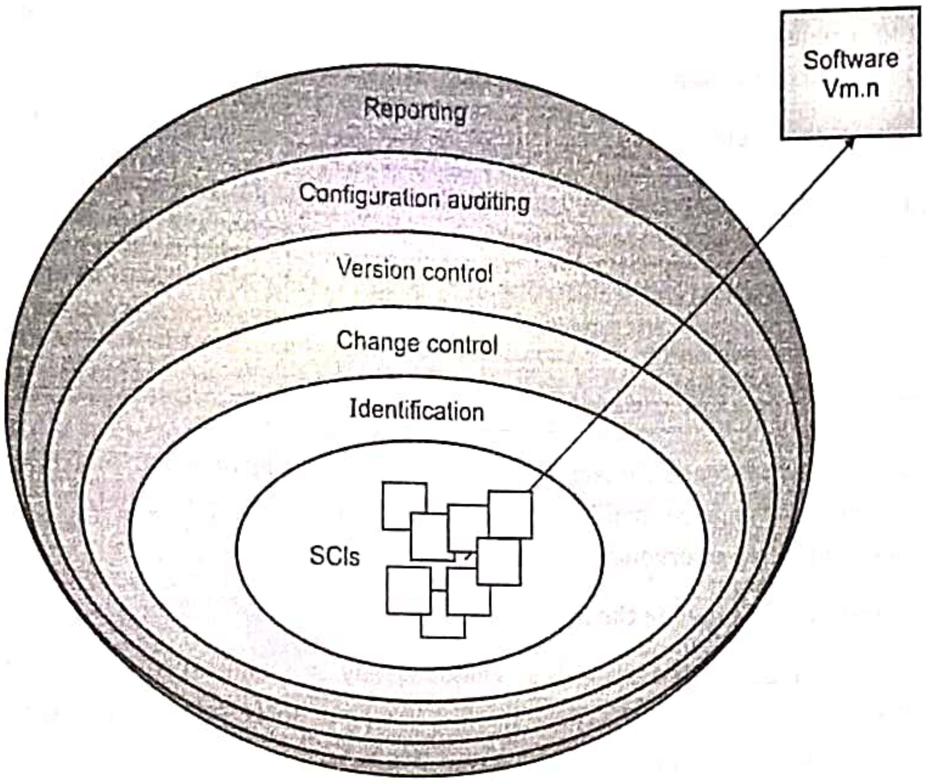


Fig. 9.3.1 : Layers of the SCM process

9.3.1 Identification of Objects in the Software Configuration

- To control and manage software configuration items, each should be separately named and then organized using an object-oriented approach. Two types of objects can be identified :

1. Basic objects and
2. Aggregate objects

1. **Basic object :** A basic object is a unit of information that has been created by a software engineer during analysis, design, code or test.
2. **Aggregate object :** An aggregate object is a collection of basic objects and other aggregate objects.

- Since each of the object in the product has some distinct features that make the object different from other objects. The object has a unique name, description and list of resources associated with it. The name of the object should be very clear and distinct.
- The description of the object should identify the type of software configuration item i.e. SCI represented by that object.

9.3.2 Version Control

- The version control actually controls the new releases or new versions. It combines the procedures and tools in order to control various versions of configuration objects.
- Any version control management system has four major capabilities that is integrated in the version control system itself :
  1. The repository will store all the related configuration objects.
  2. The repository will store all the versions of configuration objects.
  3. It has a facility to provide the related information about configuration objects so that a software engineer will construct a new version based on those information.
  4. To track all the issues in development process by using a special tracking facility in the version control.

9.3.3 Change Control

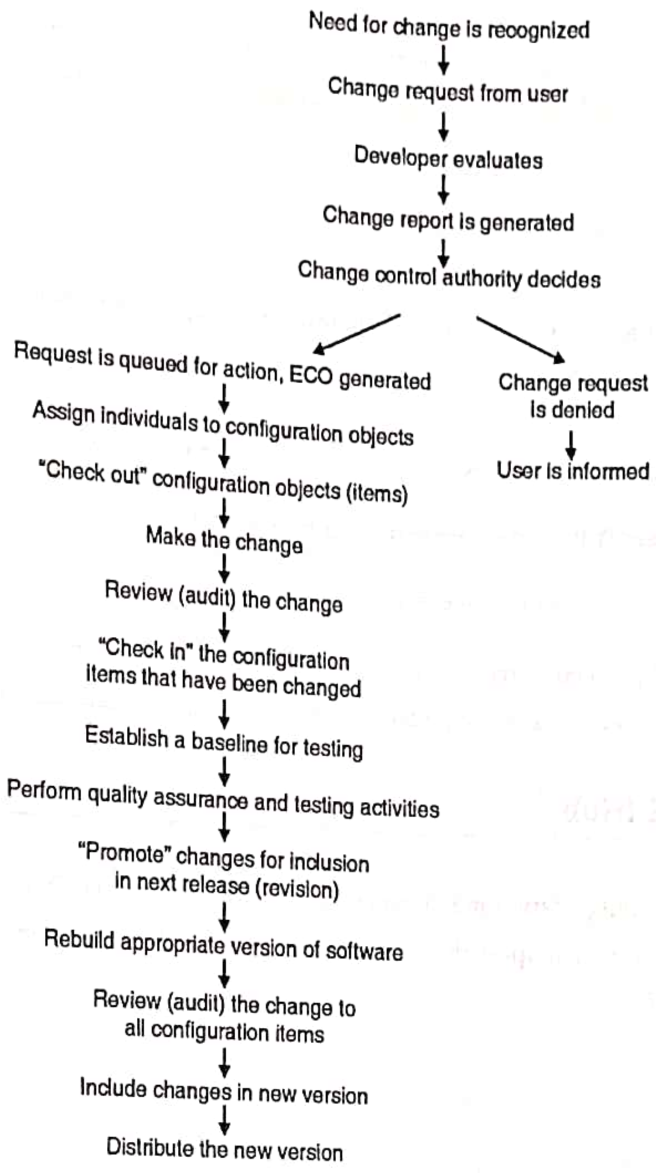
University Questions

Q. Write short note on : Change control process. **SPPU - May 12, May 14, 5 Marks**

Q. Explain the change control mechanism in software configuration management? **SPPU - Dec. 18, Dec. 19, 3 Marks**

- In a development of a larger software system, the changes may be uncontrolled and it leads to a complex situation. In such projects the change control is done partially by human and partially by some automated tools. In such a complex situation human intervention is very much necessary.
- The change control process is explained in the following Fig. 9.3.2.
- The change request is first submitted and then evaluated by a technical support staff by taking into consideration its potential side effects and the overall impact on other objects in the product. The other parameters to be evaluated are system functions, the cost of project etc.

- Based on the result of the evaluation treated as a change report, the implementation is taken into consideration. This report is submitted by change control authority i.e. CCA. The CCA is a person or a group who has the final authority to take decision on any changes and their priority.
- After approval from CCA, a change order called ECO (engineering change order) is generated for each of the change.
- The object to be changed can be placed in a directory that is controlled solely by the software engineer making the change.
- These version control mechanisms, integrated within the change control process, implement two important elements of change management :
  - o access control and
  - o Synchronization control.
- Before an SCI become a baseline, the changes should be applied. The developer will look after whether the changes are justified or not by project. The technical requirement must check properly.
- After approval from CCA, a baseline may be created and change control is implemented.



Need for change is recognized
Change request from user
Developer evaluates
Change report is generated
Change control authority decides
Request is queued for action, ECO generated
Assign individuals to configuration objects
"Check out" configuration objects (items)
Make the change
Review (audit) the change
"Check in" the configuration items that have been changed
Establish a baseline for testing
Perform quality assurance and testing activities
"Promote" changes for inclusion in next release (revision)
Rebuild appropriate version of software
Review (audit) the change to all configuration items
Include changes in new version
Distribute the new version
Change request is denied
User is informed

Fig. 9.3.2 : The change control process

- Once the final product is released, the formal changes must be instituted as per the Fig. 9.3.2. This formal change is outlined.
- The CCA plays an active role in second and third layers of change control based on size of the project.

9.3.4 Configuration Audit

- A software configuration audit addressees the following questions :
  - Whether the change mentioned in the ECO applied or not ? Incorporated any additional modifications ?
  - Whether the formal technical review conducted or not to assess technical correctness ?
  - Whether the software process followed or not and software engineering standards properly applied ?
  - Whether the changes are "highlighted" in the SCI ?
  - Whether SCM procedures for noting the change, recording it, and reporting it been followed or not ?
  - Whether all SCIs properly updated ?
- In some of the cases, the configuration audit questions are asked as part of a FTR (formal technical review). Sti SCM is a formal activity. The SCM audit is conducted separately. These activities are performed by the qualit assurance group.

9.3.5 Status Reporting

- Configuration status reporting is a SCM task that has following questions to answer :
  - What had happened? (Specify the changes made).
  - Who did it? (Specify the responsible person or authority approving the changes).
  - When did it happen ? (Specify the time of occurrence of the change)
  - Anything else is affected based on the changes made?
- The CSR report is generated on regular basis to keep the developers aware of the changes made and the hist of the changes made. It is very much useful in new releases or constructing new versions.

9.4 SCM Tools such as GitHub

- The most popular software configuration management tool is Git. Git is a distributed version control tool was invented by Linus Torvalds to support the development of Linux. Linus named the version control toc which is an old English word.

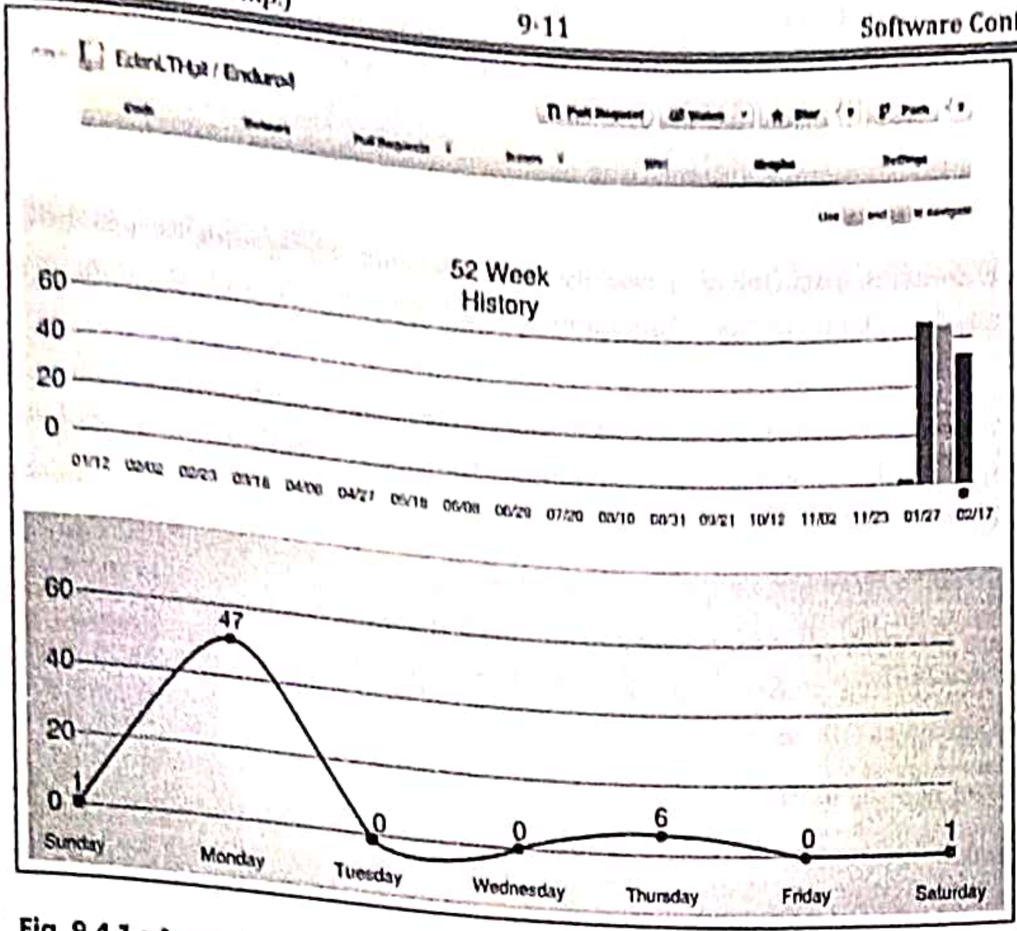


Fig. 9.4.1 : A graph produced by Github showing how code only gets committed

- The purpose of Git is to provide a reliable and versatile version control and configuration management, it does this with a little different approach than some other version control tools. Git also enables people to work together and empowers teamwork.
- Github is a Git hosting service where a person can make his own repository accessible for the public. Github is free unless you want your code to be private. It also offers a lot of socially focused features.
- Github supports Git and makes it more accessible to the public. There are a lot of code hosting services on the internet for all kinds of SCM tools if you do not want to trouble yourself with setting up servers and configuring the tools.
- Github is very focused on the collaboration between users. Each project on Github get its own wiki-page where people active in the project can document various information. Github also tracks commits and makes statistics accessible via graphs. As a user you can follow other users and enter projects.
- This is surely part of why Github is so successful, its features gives clear feedback on projects progress and allows for easy communication between developers. Many projects on Github are only used to store code and provide backup for single person projects.

9.5 Computer-Aided Software Engineering (CASE)

- Computer-Aided Software Engineering (CASE) tools are developed to help software engineers, managers and all the software practitioners in all the software activities related to the software development process.
- The CASE tools are actually built to automate software management activities. It also assists a software engineer in requirement analysis, design, coding and testing, debugging.

- Software engineering is considered to be very difficult. The CASE tools reduce the amount of effort required to develop a product. In most of the project, these tools play very important role to achieve the benefits. In addition to these benefits, tools also provide different way of looking at software engineering information that improves the software quality.
- CASE tools help a software engineer or a practitioner in developing high-quality products. It also produces additional customized work due to these automation tools. Without CASE tools the thing might have been too tough.
- Computer-aided software engineering provides a platform where a software engineer automates all the manual activities. This improves the insight of engineering to produce better products. Thus CASE ensures the quality before the actual product is built.

CASE tools

CASE tools are class of software which is useful to automate the different activities in life cycle. They are useful in different software engineering phases. They are useful in requirement analysis, design, coding, testing, etc. Different CASE tools are given as follows :

- Business process engineering tools
- Project planning tools
- Risk analysis tools
- Project management tools
- Requirement tracing tools
- Documentation tools
- System software tools
- Quality assurance tools
- Database management tools
- Prototyping tools
- Programming tools
- Web development tools
- Static analysis tools
- Dynamic analysis tools
- Test management tools
- Client server testing tools
- Re-engineering tools

Functions of CASE tools

- Analysis
- Design
- Code generation
- Documentation

Types of CASE Tools

- The general types of CASE tools are listed as follows :
  - Diagramming tools
  - Computer display and report generators
  - Analysis tools
  - Central repository
  - Documentation Generators
  - Code generators

CASE - taxonomy

Different terms involved in the CASE tools are as follows. These tools are useful to understand the CASE in details.

Workbenches

- Workbench integrates different CASE tools in one application.
- It is useful to support certain process activity.
- They generally have homogeneous and consistent interface
- They have easy invocation of tools and tools chains.
- CASE workbenches can be classified into following categories.
  - Business planning and modeling.
  - Analysis and design
  - User interface development
  - Programming
  - Verification and validation
  - Maintenance and reverse engineering
  - Configuration management
  - Project management
  - Design management.

Tool-kits

- It is collection of products which are loosely integrated.
- Support provided by tool-kits is limited to programming, configuration management, project management.
- It is extended from basic set of operating system tools.

Environments

- It is nothing but the collection of CASE tools and workbenches.
- It is useful in supporting software process.

- It is classified into different categories based on basis of integration.
  - Toolkits
  - Language centered
  - Integrated
  - Fourth generation
  - Process centered

Components of CASE

- Components of CASE tools is shown in Fig. 9.5.1.

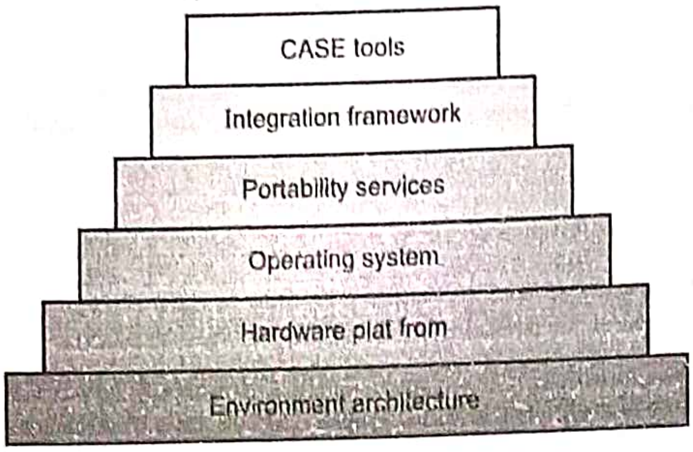


Fig. 9.5.1 : Components of CASE Tools

- It consists of different blocks like environment architecture, hardware platform, operating system, portabilit services, integration framework and lastly CASE tools.

Categories (upper, lower and Integrated CASE tools)

- It is used in wide variety of software development life cycle methods.
- It is also useful in project identification and selection.
- It is useful in project initiation, planning and design.
- Components of CASE tools are classified into 3 main categories which are as follows :
  - Upper CASE tools
  - Lower CASE tools
  - Integrated CASE tools
- Upper CASE tools
  - These are the tools which supports software development from implementation.
  - It mainly focuses on analysis phase.
  - It also focus on design phase.
  - It includes diagramming tools, report and form generators and analysis tools.
- Lower CASE tools :
  - It is useful in supporting implementation and integration tasks.

- It is useful in database schema generation, program generation, implementation, testing, configuration, etc.

- Integrated CASE tools :
  - It is also called as ICASE.
  - It supports both upper CASE tools and lower CASE tools.
  - If we consider the example of designing the form and building the database to support it at the same time.
  - Different tools are available for creating diagrams, forms, and reports.
  - It is also supporting analysis, reporting, code generation, etc.

9.6 Emerging Software Engineering Trends

- It is style of software development which is focused on public availability and communication.
- Usually communication happens using internet.
- It is used in freeware, open source software and commons based peer production.
- It is also used in agile development model.
- It is generally used in development of free software.
- It is very compatible with free software.
- They meets online for software development.
- It is evolution of integrated development environment.
- Typical functionalities are as follows :
  - Version control system
  - Bug tracking system
  - Todo list
  - Mailing list
  - Document management system
  - Forum
- They also involve users in the collaborative development.
- It is used in most technological disciplines.

9.6.1 Model-driven Development

- It is software design approach for development of software system.
- It provides guidelines for structuring of specifications.
- It is kind of domain engineering.
- It is launched by object management group.
- It defines system functionalities using platform independent model.
- Related standards are as follows.

- Unified modeling language (UML)
- Meta object factory (MOF)
- XML metadata interchange (XMI)
- Enterprise distributed object computing (EDOC)
- Software process engineering metamodel (SPEM)

- Different tools are used in model driven architectures.
  - Creation tools
  - Analysis tools
  - Transformation tools
  - Composition tools
  - Test tool
  - Simulation tools
  - Metadata management tools
  - Reverse engineering tools
- Model driven development is shown in Fig. 9.6.1.

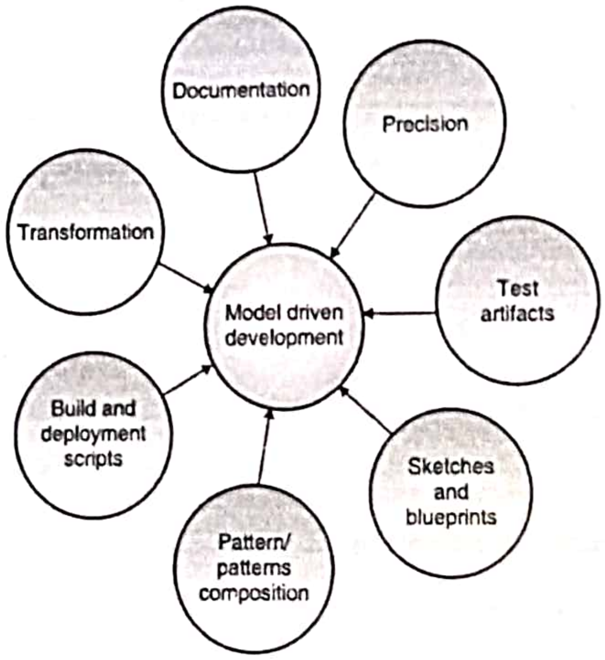


Fig. 9.6.1 : Model driven architecture

9.6.2 Test-driven Development

- It is software development process which relies on repetition of very short development cycle.
- First, developer writes test cases which define a desired improvement.
- It is related to test first programming concept of extreme programming.
- Test driven development cycle is shown in Fig. 9.6.2

1. Add test
2. Run all tests
3. Write some code
4. Run tests
5. Refactor code
6. Repeat the process.

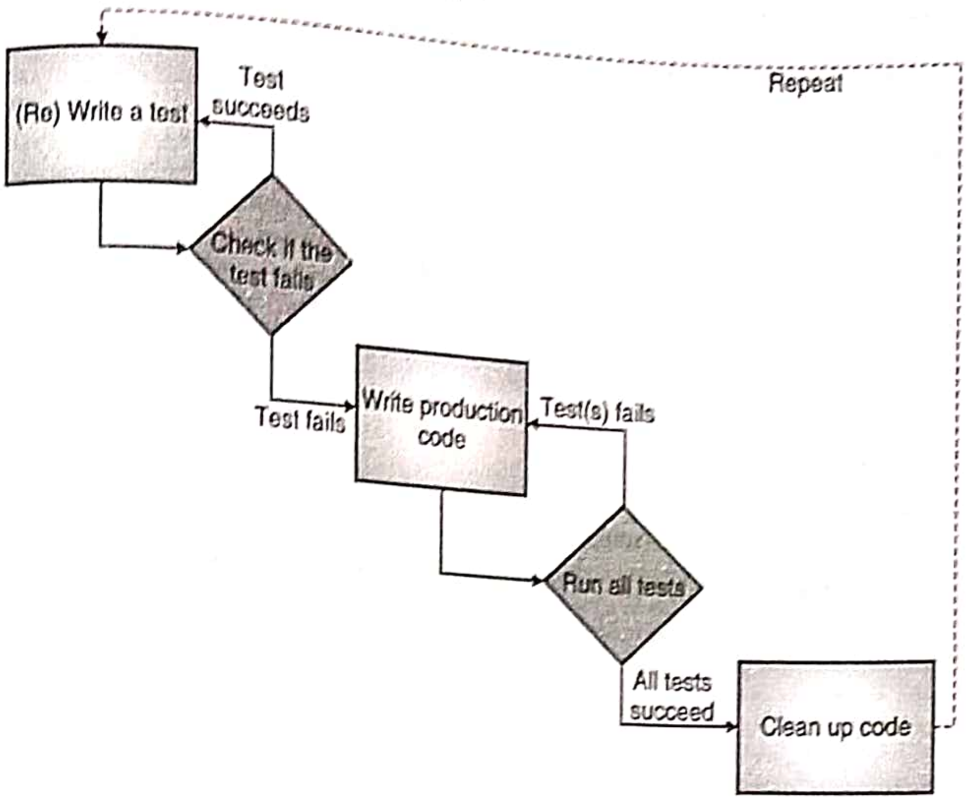


Fig. 9.6.2 : Test Driven development

Challenges of global software development

- Communication breakdown
- Coordination breakdown
- Control breakdown
- Cohesion barriers
- Culture clash
- Ability to gain market share.
- Greater flexibility and variable staffing model.
- Lower cost labour.
- Access to broader set of skilled workers.
- Ability to leverage outsourcing providers with specialized skill.
- Possibility of establishing a presence in geographies that may become new market for product.
- Ability to gain competitive edge.
- Business driver and global delivery challenges are shown in Fig. 9.6.3.

Business drivers and global delivery challenges

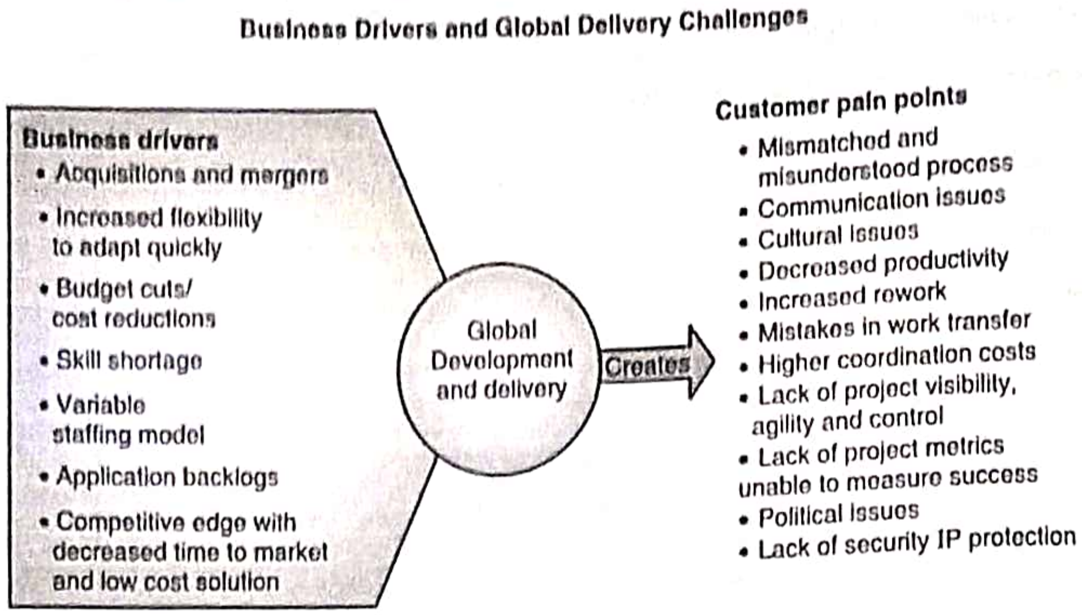

Business Drivers and Global Delivery Challenges
Business drivers
• Acquisitions and mergers
• Increased flexibility to adapt quickly
• Budget cuts/ cost reductions
• Skill shortage
• Variable staffing model
• Application backlogs
• Competitive edge with decreased time to market and low cost solution
Global Development and delivery
Creates
Customer pain points
• Mismatched and misunderstood process
• Communication issues
• Cultural issues
• Decreased productivity
• Increased rework
• Mistakes in work transfer
• Higher coordination costs
• Lack of project visibility, agility and control
• Lack of project metrics unable to measure success
• Political issues
• Lack of security IP protection

Fig. 9.6.3 : Business Driver and global delivery challenges

- Misunderstood processes or mismatched processes.
- Communication issues can lead to misunderstanding, omissions, errors and rework.

9.7 Case Study : CFEngine Configuration Tool

9.7.1 Introduction : CFEngine

- CFEngine is a configuration management system that provides a framework for automated management of IT infrastructure. CFEngine is decentralized and highly scalable. It is powered by autonomous agents that can continuously monitor, self-repair, and update or restore an entire IT system, with negligible impact on system resources or performance.
- CFEngine is a distributed system for managing and monitoring computers across an IT network. Machines on the network that have CFEngine installed, and have registered themselves with a policy server (see Installation and Configuration), will each be running a set of CFEngine component applications that manage and interpret textual files called policies. Policy files themselves contain sets of instructions to ensure machines on the network are in full compliance with a defined state. At the atomic level are sets, or bundles, of what are known in the CFEngine world as Promises. Promises are at the heart of Promise Theory, which is in turn what CFEngine is all about.

9.7.2 Policy Language and Compliance

- For many users, CFEngine is simply a configuration tool – i.e. software for deploying and patching systems according to a policy. Policy is described using promises. Every statement in CFEngine 3 is a promise to be kept at some time or location. More than this, however, CFEngine is not like other automation tools that "roll out" an image of some software once and hope for the best. Every promise that you make in CFEngine is continuously verified and maintained. It is not a one-off operation, but a self-repairing process should anything deviate from the policy.
- CFEngine ensures that the actual state of a system is in compliance with the predefined model of desired state for the system. If it is not in compliance CFEngine will bring it into compliance. This is known as convergence.

- That model is represented by one or more policies that have been written using the declarative CFEngine policy language. The policy language has been designed with a vocabulary that is intuitive, yet at the same time can still support the design of highly complex IT systems.
- Those policies are distributed across all hosts within the system via download from the policy server. Every host will then interpret and execute each of the instructions it has been given in a predetermined order.
- CFEngine continually monitors all of the hosts in real-time, and should the system's current state begin to drift away from the intended state then CFEngine will automatically take corrective action to bring everything back into compliance.

9.7.3 CFEngine Policy Servers and Hosts

- There are basically two categories of machines in a CFEngine environment : policy servers and their client hosts. Policy servers are responsible for making policy files available to each of the client hosts that have registered with it (a.k.a. bootstrapped), including itself. Hosts on the other hand are responsible for ensuring they continuously pull in the latest policies, or changes to policies, from the policy server. They are additionally responsible for ensuring they remain fully compliant with the instructions contained within the policy files, at all times.
- The role of a particular machine where CFEngine is deployed determines which of the components will be installed and running at any given moment.

9.7.4 CFEngine Component Applications and Daemons

- There are a number of components in CFEngine, with each component performing a unique function: components responsible for implementing promises, components responsible for organizing large networks of agents, and other components responsible for providing the infrastructure of CFEngine.
- These components form the basis of automation with CFEngine. They are independent software agents running on the various systems that make up your infrastructure. They communicate with one another as shown in the following figure, using a protocol that allows each host to distribute promises, act upon them, and report status to a central server.

9.7.5 CFEngine Features

- Defines the configuration of an entire IT system, including: Devices, Users, Applications, and Services.
- Helps maintain that system over time.
- Checks the system state at any given moment.
- Ensures compliance with a desired system state.
- Propagates real-time modifications or updates across the system.

9.7.6 Choose a CFEngine Version

CFEngine Enterprise is a licensed edition for enterprises that plan to use the tool in production environments. The Enterprise edition comes in several variants, including one that can be evaluated for free (up to 25 servers).

9.7.7 Installation

- There are several steps to bring up a CFEngine installation within an organization:
  1. Prepare all appropriate machines for installation.
  2. Configure your network and security.
  3. Download the CFEngine software.
  4. Install CFEngine on the Policy Server(s).
  5. Bootstrap the Policy Server to itself.
  6. Initiate post-install configuration on the Policy Server.
  7. Install CFEngine on the Host machine(s).
  8. Bootstrap the Host(s) to a Policy Server.

Review Questions

1. Write short note on : Elements of a configuration management system.
2. What is software configuration management repository?
3. Discuss role of SCM repository.
4. What are elements that exist when an effective SCM system is implemented? Explain each in detail.
5. Discuss features of SCM repository.
6. Explain SCM process in detail.
7. Explain change control process.

□□