Chapter 2

→ Process, Methods, tools + Diagram (Fig 2.1)

→ 4 types of change are encountered during support Phase (Correction, Adaption..)

→ S/w process + Diagram (Fig 2.2)

→ 5 maturity levels that are defined... (Initial, Repeatable...)

→ Each KPA is described by following characteristics... (Goals, commitment)

→ Maturity levels under KPA (All levels + points)

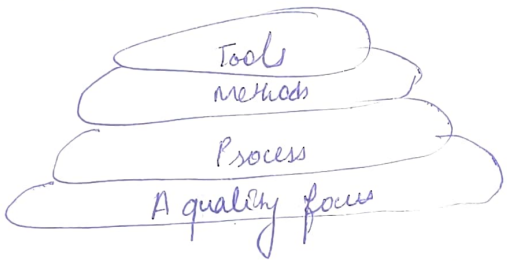
→ S/w process Model (All *)

→ Component Based Dev.

→ Product & Process

Process, Methods, Tools

Software engineering is a layered technology



The bedrock that supports software engineering is a quality focus.

Process

① It is the foundation of for software engineering.

② It holds the technology layers together and enables rational and timely development of computer software.

③ Process defines a framework for a set of key process areas that must be established for effective delivery of software engineering technology.

Methods

① It provide the technical how-to's for building software.

② Methods encompass a broad array of tasks that include requirements analysis, design, program construction, testing, and support.

③ software engineering methods rely on a set of basic princi

that govern each area of the technology and include modeling activities and other descriptive techniques.

Tools

① It provide automated or semi-automated support for the process and the methods.

② When tools are integrated so that information created by one tool can be used by another, a system for the support of software Engineering is established development, called computer-aided software engineering, is established.

③ CASE combines software, hard. and software engineering database to create a software engineering environment analogous to CAD/CAE

Generic View of Software Engineering

The work associated with software engineering can be categorized into three generic phases i.e.

① Definition Phase → what
② Development Phase → how
③ Support phase → change associated with error correction, adaptions required as the software's environment evolves, and changes due to enhancements brought about by changing customer requirements.

Four changes are encoured during support phase

① Correction → Corrective maintenance changes the software to correct defects

② Adpation → Adaptive maintenance results in modification to the software to accomodate changes to its external environment.

③ Enhancement → As software is used, the customer/user will recognize additional functions that will provide benefit. Perfective maintenance extends the software beyond its original functional requirements

④ Prevention → Computer software deteriorates due to change, and because of this, preventive maintenance, often called software reengineering, must be conducted to enable the software to serve the needs

of its end-users. In essence, preventive maintenance makes changes to computer programs so that they can be more easily corrected, adapted & enhanced

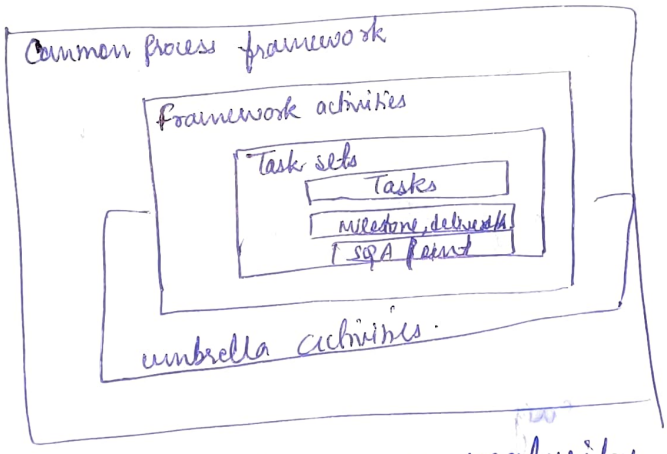
The Software Process

Software Process characterized as

① Common process framework is established by defining a small no. of framework activities that are applicable to all software projects, regardless of their size or complexity

② A no. of task sets → collection of software engineering work tasks, project milestone and quality assurance points.

③ finally umbrella activities - such as software quality assurance, software configuration measurement overlay the process model. They are independent of one framework activity and occur through out the process.



Different levels of process maturity

Level 1 : Initial

The software process is characterized as ad hoc and occasionally even chaotic. Few processes are defined, and success depends on individual effort.

Level 2 : Repeatable.

Basic project management process are established to track cost, schedule, and functionality. The necessary process discipline

that concern each area of the technology and include modeling is in place to repeat earlier successes on projects with similar application.

Level 3: Defined

① The software process for both management and engineering activities is documented, standardized, and integrated into an organization-wide software process. All projects use a documented and approved version of the organization's process for developing and supporting software. This level includes all characteristics defined for level 2.

Level 4: Managed

Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood & controlled using detailed measures. This level includes all characteristics defined for level 3.

Level 5: Optimizing

Continuous process improvement is enabled by quantitative feedback ~~the~~ from the process and from testing innovative ideas and technologies. This level ~~too~~ includes all characteristics for level 4.

The software engineering associated key process areas (KPAs) with each of the maturity levels. The KPAs describe those software engineering functions that must be present to satisfy good pratice at a particular level.

* Each KPA is described by identifying the following characteristics:

① Goals → the overall objectives that the KPA must achieve

② Commitments → requirements that must be met to achieve the goals or provide proof of intent to comply with the goals.

Abilities → those things that must be in place to enable the organization to meet the commitments.

Activities → the specific tasks required to achieve the KPA function

Methods for monitoring implementation
the manner in which the activities are monitored as they are put into place.

Methods for verifying implementation → the manner in which proper practice for the KPA can be verified.

KPAs should achieved at each process maturity level

Process maturity level 2
1. Software configuration management
2. Software quality assurance
3. Software subcontract management
4. Software project tracking & oversight
5. Software project planning
6. Requirements management

Process maturity level 3
1. Peer reviews
2. Intergroup coordination
3. ~~Software product~~ project Software product engineering
4. Integrated software management
5. Training program
6. Organization process definition
7. Organization process focus.

of the technology and include modeling
info in place to repeat earlier successes on projects with similar

Process maturity level 4

1. Software quality management
2. Quantitative process management

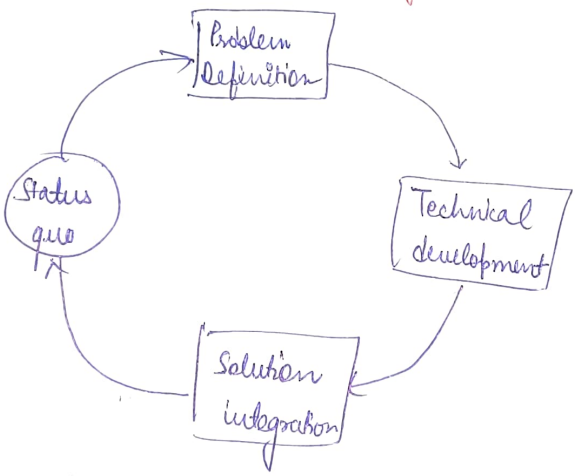
Process maturity level 5

1. Process change management
2. Technology change management
3. Defect prevention

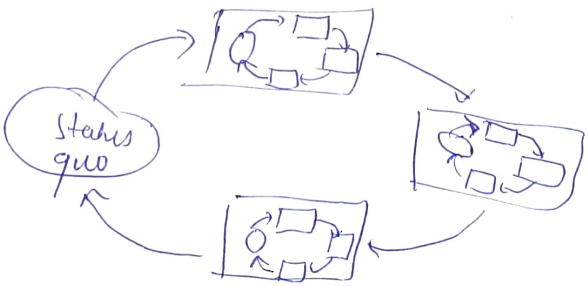
Software Process Models

A process model for software engineering is chosen based on the nature of the project and application, the methods & tools to be used, and the controls and deliverables that are required.

(a) Phases of a problem solving loop.



(a) Phases within problem solving loop



Four distinct stages are encountered

(1) Status quo → represents the current state of affairs

(2) problem definition → identifies the specific problem to be solved

(3) technical development → solves the problem through the application of some technology

(4) Solution integration → delivers the results to those who requested the solution in the first place.

This problem solving loop applies to software engineering work at many different levels of resolution.

It can be used at macro level when the entire application is considered, at a mid-level when program components are being engineered and even at line of code level

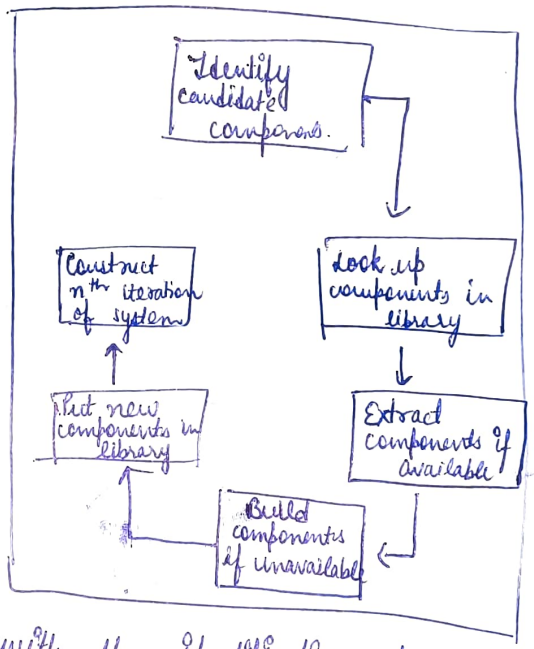
These stages are recursive, meaning each phase contains smaller versions of the same loop. This reflects how, at every level of detail, the same problem-solving process is repeated. The "Chaos model" highlights the inherent complexity and non-linearity of software development, emphasizing that despite attempts to organize the process, it remains dynamic & interconnected

Component-based Development

① The Component based development model incorporates many of the characteristics of the spiral method

② It is evolutionary in nature, demanding an iterative approach to the creation of software.

③ It composes applications from prepackaged software components (called classes).



The engineering activity begins with the identification of candidate classes. This is accomplished by examining the data to be manipulated by the application and the algorithm that will be applied to accomplish the manipulation. Corresponding data and algo. are packaged into a class.

After identification, class library is searched to determine if these classes already exist. If they do, they are extracted from the library and reused. If a candidate class does not reside in the library, it is engineered using object-oriented methods.

The first iteration of the application to be built to meet the unique needs of the application. Process flow then returns to the spiral & will ultimately re-enter the component assembly iteration during subsequent passes through the engineering activity.

The unified software development process is representative of a no. of component-based development models that have been proposed in the industry

Product & Process

Margaret Davis discusses the important duality b/w product and process in software development, arguing both elements are essential and interdependent. She notes that the software industry has a tendency to swing between focusing on product (such as programming languages or data encapsulation) and process (like structured analysis or the Capability Maturity Model), creating confusion for developers and failing to address core issues. Product & process are not opposites but rather two sides of the same coin.

She uses the analogy of the dual nature of light - accepted to be both a particle and a wave - to illustrate how software development must embrace the duality of product and process. Ignoring this duality obscures opportunities for innovation and reuse, as viewing an artifact solely as either a product or a process limits its potential and application in future software activities.

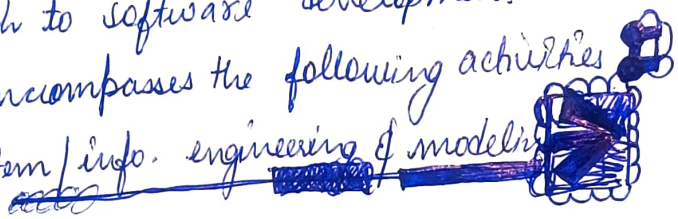
Software developers should derive as much fulfilment from the development process as from the end result. Recognizing this duality is key to maintaining developer engagement and satisfaction as software development evolves from programming to a more mature field of software engineering

Linear Sequential Model

(1) It sometimes called the classic life cycle or the waterfall model, the linear sequential model suggests a systematic, sequential approach to software development.

(2) It encompasses the following activities

(i) System / info. engineering & modeling



1561
424