SDS Page No.
Date / /

MOD - 1

1.1 ✓ SW Engg.,
✓ layered technology,
✓ Process Framework,
✓ Compatibility Maturity Model (CMMI)

1.2 Perspective Models,
Waterfall Model,
Incremental,
RAD,
Evolutionary Process Models,
Prototyping,
Spiral,
Test Driven development

1.3 Agile process,
Scrum - Industry Prespective,
Dev-Ops development practice.

SDS Page No.
Date / /

~~s/w & s/w engineering~~

s/w & s/w engineering

- s/w comprises of computer program, that provides desired features, functionalities & performance when executed.
- s/w encompasses descriptive info. available in both hard copy & in virtual form.

Types of s/w

a) System s/w ⇒ Programs written to service another programs.
eg ⇒ compilers, editors.
⇒ Processes complex & determinate info. structures.

b) Engineering s/w ⇒ characterised by no. ~~searching~~ Crunching algo.
⇒ used in astronomy, volcanology, molecular biology, etc.

c) Application s/w ⇒ stand-alone program that perform a specific task used for addressing business models.
eg ⇒ Management decision making tools.

d) Embedded s/w ⇒ resides within a product or a system, controlling features for end users & system itself.
eg: ⇒ automobile digital functions.

e) Productline software ⇒ designed for to be used by many customers, and addresses specific capabilities.
eg ⇒ Inventory control products, word processing.

f) web App ⇒ Network centric application that offer various functionalities.
eg ⇒ simple HTML files to sophisticated computing env.

3) AI slw ⇒ utilized non-numerical algos for solving complex problems.

eg ⇒ Robotics, expert systems, pattern recognition.

* Challenges in slw development

1) Growing user interest ⇒ the no. of stakeholders interested in slw has significantly increased.

2) Complexity of system ⇒ modern tech & products are highly complex, requiring meticulous attention to slw elements

3) consequences of slw failure ⇒ may range from minor failures that can be fixed quickly to catastrophic failure.

4) Increasing Perceived value ⇒ as perceived value of any app. grows their user base longevity also increases.

Layered Technology

slw engg. is a fully layered technology, to develop slw we need to go from one layer to another where all layers are interconnected & each layer demands completion of previous layer.

4 parts in which layered tech. is divided

1) A quality Focus layer

- Serves as bedrock for slw development. & continuous process of improvement principles of slw.

- Provides integrity & security to SLW and ~~data~~ manging data access control.
- Focuses on quality & promoting quality management.
- A quality focused culture fosters continuous process improvement

2) Process layer
- Forms foundation of SLW engineering, key that binds all layers together that enables development of SLW.
- Provides a framework for rational & timely development of SLW & its delivery
- Process considers all the activities, actions & tasks i.e. milestones required throughout the project.

Process activities →

a) communication ⇒ Imp thing in dev. & necessary to know the demands of the client.

b) Planning ⇒ drawing a map to reduce complications of development.

c) Modelling ⇒ Model is created accordingly, so that client can get better understanding.

d) construction ⇒ coding & testing

e) Deployment → delivery of project to the client for evaluation.

3) Method
- Provides the details for all "how to do" questions in building the SLW.
- Has info. about all tasks to be done such as communication, requirement analysis, design modelling, programming & testing.

4) Tools
- SLW development tools provide automated or semi-automated support for SLW development process & methods.
- Tools facilitate various task such as designing, modelling, coding, testing... & are integrated in system, thus can be used by another app. too.

Process Framework.

- A generic process framework for s/w engineering has 5 parts

a) communication →

- identify stakeholders and their involvement in solution
- Find the unknown questions & solution to them i.e. features & functionalities.
- divide the problem into parts for better understanding
- Explore the graphical rep (models) of.

b) planning →

- Try finding patterns & similarities in prev. soln.
- Evaluate existing project for reuse.
- define sub-problems and assess their solvability.
- create an implementation friendly rep. of soln.

c) Modeling →

- Ensure solution aligns with the plan
- verify traceability of source code to design model
- confirm correctness of each component through reviews or proof.
- Develop a comprehensive testing strategy, validate against requirements.

d) deployment →

- Take care of all the configurations before deploying the project
- All even testing should be done.

e) construction →

- code the entire Project.

(2)

* Umbrella activities

① Project tracking and control
② Risk management
③ Quality assurance
④ configuration management

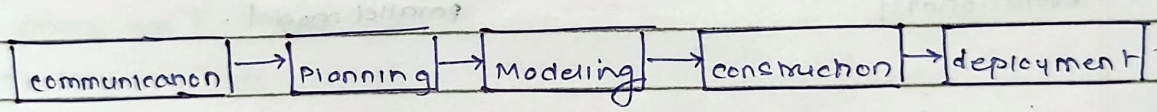
→ Task set : defines specific work to be done within sw engg. actions to achieve objectives.

→ Process patterns : describes a problem encountered in sw engg. ~~work~~ work, identify it & suggest a proven solution.

→ Process flow : how the framework activities, actions & tasks are organised in terms of sequence & time

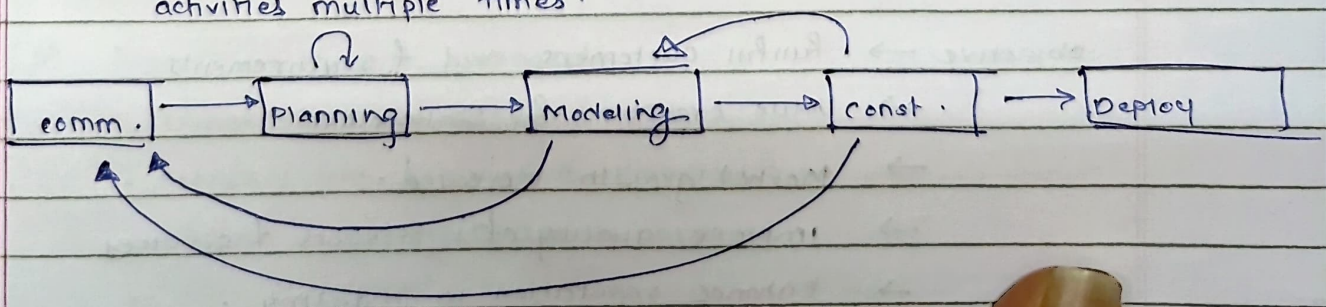
* Types of process flows

① linear ⇒ executes each of 5 framework activities in line (sequence).



a step by step process w/o revisiting any.

② Iterative ⇒ repeats one or more activities before moving to next. Also allows a feedback & refinement by revisiting activities multiple times.

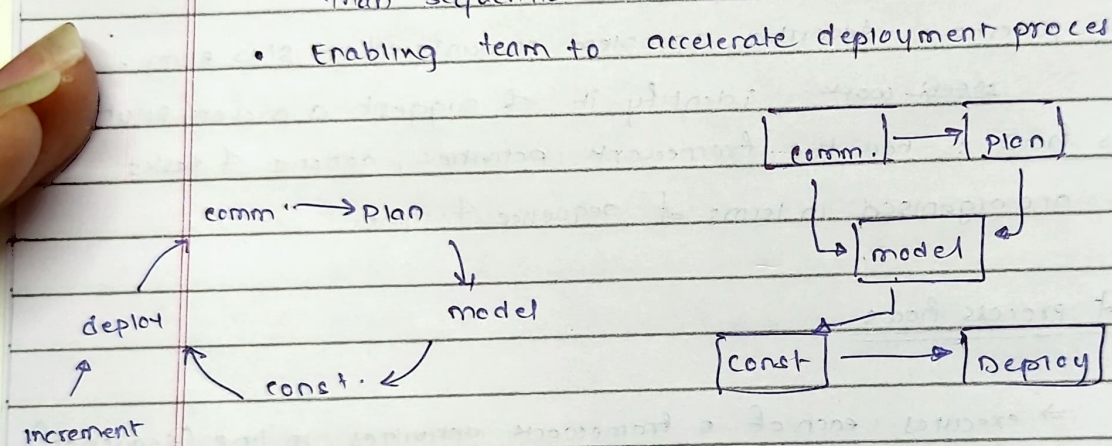


③ Evolutionary

- Executes activities in a circular manner, with each circuit through the 5 activities resulting in a more complete version of s/w.
- focuses on incremental development & continuous improvement.

④ Parallel

- one or more activities concurrently executed rather than sequential execution.
- Enabling team to accelerate deployment process.



Evolutionary parallel model.

* CMMI (capability maturity model)

- Framework used to analyse approaches & techniques followed by an organisation to develop s/w products.
Also provides guidelines to further develop the process.

objective → Fullfill customers need & requirements
→ Value creation for stakeholders
→ Market growth increased.
→ improve quality of products & services
→ Enhance reputation in industry.

SDS Page No.
Date / /

* CMMI staged representation

- Akin to climb set of stairs, where each stairs increases the maturity level of process.
- At each level, org needs to meet a criteria, increasing with levels of achievements.

→ Provides a structured approach i.e. following predefined staircases to improve the process.

* CMMI continuous representation

- Akin to use a slider with multiple positions, each representing different capability area such as project management, engg.
- organisation have flexibility to focus on specific areas of improvements at their own pace, adjust the slider according to priorities.

→ This allows customization & adaption based on unique needs of organisations.

* CMMI has 5 stages

1) Maturity level : Initial
   - processes are poorly managed & controlled
   - unpredictable outcomes from process
   - chaotic approach. & ad hoc
   - lowest quality & high risk.
   - individual performance & no grp work.

2) Maturity level : Managed:
   - requirements are managed.
   - Processes are planned & controlled.
   - Projects implemented acc. to plan.
   - risk still exists.
   - quality is improved.

Process performance quality risk procedure

3) Maturity level : Defined
- Processes are well characterized & described using proper procedures, methods, etc.
- Medium quality & medium risk
- Focus is process standardization.

4) Maturity level : Quantitatively managed
- Quantitative objectives/goals are set for performance and quality
- quantitative objectives are based on customers requirements, etc.
- Process performances are analysed quantitatively.
- higher quality of process is achieved.
- lower risk.

5) Maturity level : Optimizing
- continuous improvement in the performance.
- improvement has to be both incremental & innovative.
- highest quality of process
- lowest risk.

Perspective models

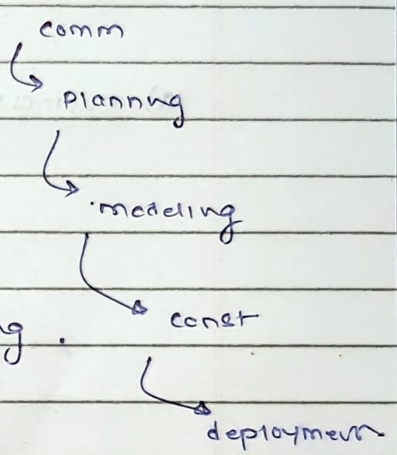
- Designed to find the best solution for proposed problem.
~~such model makes trade off betn~~
- Prescribe a set of process elements & a predictable process workflow for every project.

Types → waterfall model,
Incremental process model,
RAD model.

SDS Page No.
Date / /

① The waterfall model

- A linear sequential model, classic lifecycle model.
- Here each phase is fully completed before the start of next phase
- This model is used for small projects.
- Feedback is taken after each stage to make sure project is on right path.



Adv ⇒ simple, has sequential phases
⇒ all requirements mentioned upfront.

disadv ⇒ not good for complex projects.
⇒ issues may not be discovered until testing.
⇒ High risk.

② The incremental model

- Combines the elements of waterfall model and they are applied iteratively
- 1st increment in this model is a core product.
- Each increment ~~delivers~~ builds a portion of functionality & deliver to customer for feed back.
- Next increment implements customers suggestion.
  & process is repeated.

Adv ⇒ flexibility & low cost dev.
⇒ Easy testing & quick working slow.
⇒ customer feedback.

DisAdv ⇒ may overrun cost (due to increments)
⇒ carefull management, customer demands.

⑤ The RAD model (Rapid App. development).

- using RAD, a sfw is developed in very short time.
- initially starts with comm. betn customer & developer.
- planning depends on initial requirements and then requirements are divided into grps.

RAD has following steps:

collection of info.

a) Business Modeling ⇒
- All type of info. that is produced by every function & functions to handle that info.

define relationship to objs

b) Data Modeling ⇒
- The info in BM phase is refined into set of objects
- the attributes of each objects are identified & relationship is defined betn them

modify data obj.

c) Process Modeling
- Data objects moded in DM are modified to fulfill info flow to implement business model.

build & test

d) App. generation
- the actual system is built & automated tools are used to do so.

e) Testing & turnover
- prototypes are independently tested after each iteration so that overall testing time is reduced.

after study

ADV ⇒ swift development, working prototype.

DISADV ⇒ Requires through business analysis, complex data & process modeling.

SDS | Page No.
Date / /

Evolutionary Process Model

- Based on the concept of making an initial product and evolving the s/w product over time. With iterative and increamental approach.
- Product goes through several iteration until it becomes the final product & feedbacks are taken during development.
- Here customer involvement is very high & product is build as per customer needs.

Types

a) Iterative model

- 1st we take initial requirements & then enhance the product over multiple iterations.
- Each iteration, some modifications are made & functionalities are added.
- Build the final product as the user wants with multiple iterations, thus reducing the errors too.

Steps

a(i) Requirements gathering & analysis → from customers requirements are collected and analysed based on constraints (money).

(ii) Design → team designs the s/w using various diagrams, class, state, flow, etc.

(iii) Implementation → requirements are implemented i.e. s/w.

(iv) Testing → after implementation, s/w testing starts using diff. testing methods (white, gray, black) box methods.

(v) deployment → deployed on its working env.

(vi) Review → review is done to check the behaviour & validity of developed product (if any error, then restart all stages)

(vii) Maintainance → slw may have some bugs, updates, etc. maintainance involves debugging them.

(b) Incremental Model

- We 1st build the project with basic features → then evolve the project with iterations, mainly done in large projects.
- Process repeats several times until project is ready.

steps

(i) Requirement analysis → requirements are identified, and understood to ~~find~~ adjust them in incremental model.

(ii) Design & development → Design of systems functionality and development methods are done.

(iii) testing ⇒ checking the performance of each existing function as well as additional functionality.

(iv) Implementation ⇒ coding phase, developing actual slw. Final coding that converts design into app.

(c) Spiral Model

- combination of waterfall & iterative model, focused on risk handling along with developing with incremental & iterative approach

- SW is created via multiple cycles using a spiral approach. & later on an with successive development the final product will be developed.

parts.

- objective setting ⇒ each cycle starts with identification of purpose for that cycle, considering all alternatives.

- Risk assessment & reduction ⇒ calc. various alternatives based on goals & constraints. Focus of eval is risk to the project with each method.

- Development & validation ⇒ develop the strategies that resolve the risk. strategies like benchmarking, simulation etc.

- Planning ⇒ The next step is planned, proj is reviewed & a choice is made whether to continue further period of spiral or not.

ADV → suitable for complex projects
→ Enhanced flexibility
→ risk reduction
→ customer collab.
→ adapt changes

Disadv
→ complex
→ high cost
→ resource intensive

Prototyping

- Process of developing - a working replication of the ~~product~~ product (slw).

Done when cust. do not know exact project requirements.

steps:

① Requirement gathering & analysis is initial for desiging a prototyping model.
user is asked about what the expect from slw.

② Quick design → basic design done (made for a quick ~~over~~overview of description.

③ Build a prototype → help building an ~~action~~ actual prototype from designed prototype.

④ Initial user eval → testing performances of the model, as customer tests strengths & weaknesses., then sent to developer.

⑤ Refining prototype → based on feed back from user (client), the final system is approved (reworked).

⑥ Implement & maintain product → final system is tested & distributed to production.

SDS | Page No.
Date / /

* Types of prototyping

a) Rapid throwaway prototyping →

- Quick prototype created for each idea, allowing customer feedback.
  Prototype may not become final product but helps get feedback.

b) Evolutionary Prototyping →

- Initial prototype is refined incrementally based on customer feedback saving energy to rework from scratch.

c) Incremental prototyping →

- Final prod. is divided into small prototypes developed individually, then merged into single prod.
- Reducing complexity & proj duration.

d) Extreme Prototyping →

- Start with basic coding, then add functional screen, with data then final services, for final prototyping.

* ADV (prototyping)

→ early customer feedback.
→ easy accomodation of new requirements.
→ design flexibility
→ testing & validation of design.
→ Risk reduction & improved comm.

DISADV

→ Time & cost insensitive
→ variation in requirements
→ difficulty in accomodating changes
→ limited scalability
→ disagree customer not satisfied.
→ additional costs.

Test Driven Development (TDD)

- Test cases are written before the code that validates those test cases.
- Depends on repetition of very short dev. cycle.

→ automated unit tests are used to drive the design & free decoupling of dependencies.

Process sequences

1. Add a test → test case to describe functionality
2. Run all test cases & make sure new test case fails
3. write a code that passes new test case.
4. Run test cases
5. Refactor code (remove duplication.)
6. Repeat steps.

SDS | Page No.
Date / /

Agile Development

- way of working on a project where team collaborates closely and adapts to changes.
- Emphasized on team work, flexibility and delivering quality slw quickly
- Agile methods focus on frequent checks and improvements, with goal of meeting customer needs and company goals.

Benefits of Agile for customers

- Faster response to development req.
- Quick delivery
- High cust. satisfaction.

Benefits of Agile for vendors:

- high value features reduces wastages
- less time-to-market compared to waterfall.
- better cust. satisfaction.

Benefits of Agile for development teams.

- Enjoyment in doing work.
- less non-productivity.
- Time for valuable work.

Benefits of Agile for product managers:

- Easier alignment of development work with customer needs.
- enhanced delivery of value to customers.

Page No.
Date

Benefits of Agile for project Managers

- More concrete planning & tracking
- task level tracking for better awareness.
- Daily scrum meetings
- Quick detection & resolution of issue.

Scrum Development

- Scrum is a type of agile framework.
- In which people can address complex adaptive problem, while productivity & creativity of delivering the products is at highest possible values.

- Scrum uses iterative process

Features:

- light weighted
- emphasises self organisation
- simple to understand
- framework to help team work together.

* Sprint → time of one month or less. A new sprint immediately starts on completion of previous.

* Release → product is completed & is in release deployment phase.

* sprint review → products still have some non-achieveable features, that will be checked currently & then passed to sprint retrospective stage.

SDS | Page No.
Date / /

* sprint retrospective stage → quality or status of the product is checked

* Product backlog → based on priority the features are now implemented.

* sprint backlog → divided into two parts, product assigned features to sprint & sprint planning meeting.

ADV of using Scrum:

- Fast moving & money efficient
- Divide large product into small product.
- customer satisfaction is important (relies on feedback)
- adaptive because it has a short sprint.
- better quality is the aim

Disadv of Scrum:

- Does not allow changes into sprint.
- Not a fully described model.
- hard to plan the structure, if proj definition is not clear
- Daily scrum meetings.