CSE4708: Software Project Management

Unit II : Project Evaluation & Estimation

Topic: Software Process Models, Terminologies, Software Myths, Verification and Validation, Build and Fix

Name: Manka Sharma

Date -13th August 2020

Terminology Commonly Used

1. Deliverable – these are generated during software development. Eg. Source code, user manuals, operating procedure manuals etc.
2. Milestone – the events that refers to completion of specification or target. Eg. SRS finalization is a milestone.
3. Product – refers to what is delivered to the customer. Eg. Documents or manuals, code etc.
4. Process- the collection of steps / activities which are used to develop a product.
5. Software Process-the process through which develop the software.

Terminology Commonly Used

6. Software Metrics – used to quantitatively characterize different aspect of the software.
7. Process Metrics – To quantify the attributes of software development process and environment
8. Product Metrics –measures of the software product such as productivity, failure rate, quality, efficiency etc.

Software Myths

1. It is easy to change
2. Testing or ‘proving’ software correct can replace all the bugs / errors.
3. Reusing components always produce correct results
4. Software can work right the first time
5. Software with most features is a better software
6. Addition of more software engineers will make up for the delay
7. Aim is to develop working programs

Software Life Cycle Models

- The aim of software engineering is to produce
  - A good quality maintainable software
  - Well within the time constraint /deadline
  - Within the allocated cost / budget

- According to IEEE's definition "*Software engineering can be defined as the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software*".

"The establishment and use of sound engineering principles in order to obtain economically developed software that is reliable and works efficiently on real machines."

- By Fritz Bauer

Software Development Life Cycle

- The fundamental four phases are:
  - Planning
  - Analysis
  - Design
  - Implementation
- A software life cycle is often called as Software Development Life Cycle (SDLC).
- Different projects may emphasize on different parts of SDLC or approach SDLC phases in different ways.
- But all projects should have these phases so as to develop accurate software.

Software Development Life Cycle

- Each phase goes through some steps
- Each phase has some techniques to follow
- Each phase produces some deliverables

Software Development Life Cycle

- SDLC phases and steps proceed in a logical path from start to finish.
- Some projects the teams may move through the steps consecutively, incrementally, iteratively and in some other patterns.
- SDLC is a process of gradual refinement.
- The deliverables produced in the analysis phase are used as input to design phase, which then produce the deliverables to find out how exactly the system will be built. These deliverables are used in the implementation phase to produce the actual system.

Software Development Life Cycle

- Planning :
  - Revolves around “Why”
  - Starts with a “System request”
  - A system request presents a brief summary about the need
  - Aspects like how this system adds value
  - The feasibility analysis examines the aspects like technical feasibility, economic feasibility and organizational feasibility

Software Development Life Cycle

- Analysis:
  - Revolves around “What”
  - It is a very crucial phase of SDLC
  - It includes analysis of the current system (if any) and its problems, and then ways to design a new system
  - The requirements for the new system are gathered and modeled
  - The analyses, system concept and models are combined into a document called Software Requirement Specification (SRS)

Software Development Life Cycle

- Design
  - Revolves around “How”
  - Covers aspects such as how the system will operate in terms of hardware, software, network infrastructure, the user interface, forms, reports, databases etc.
  - A design strategy must be developed
  - This phase leads to the “architecture design” of the system
  - Architecture Design is design of the software, hardware, network, interface design, database and file storage and program design

Software Development Life Cycle

- Implementation
  - The system is actually built
  - It includes coding, testing , installation and maintenance
  - This phase gets most attention

Types of Methodologies

"The system development methodologies aim to represent the system in terms of the process of the system and the data of the system".

- **Process Centered Methodology** –
  - It focuses mainly on the processes associated with the system.
  - It represents the systems in terms of the process modeling techniques such as Data Flow Diagrams, Context Diagrams etc.
  - A process model basically represents
    - ✓ The processes of the system
    - ✓ The flow of data amongst the processes and
    - ✓ The way the system operates
- **Data Centered Methodology** –
  - It emphasis on the contents of the data and the way the data is organized.
  - It represents systems in terms of data models such as E-R diagram, Data Dictionary etc.
  - Therefore in this type of methodology, the analysts need to understand the information created and used by the system.
- **Object Oriented Methodology –** This type of methodology attempts to balance the focus between processes and data. It represents the system as objects that encapsulate both data and processes. It makes use of the modeling techniques such as class diagrams, sequence diagrams etc.

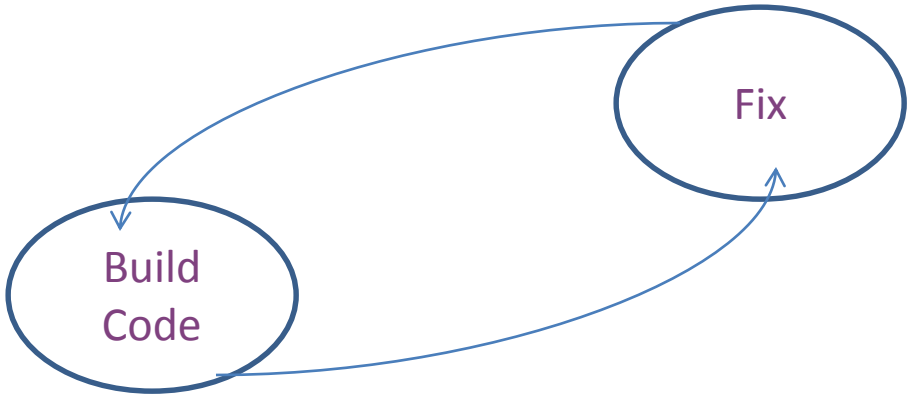
Types of Methodologies

- **Object Oriented Methodology –**
  - This type of methodology attempts to balance the focus between processes and data.
  - It represents the system as objects that encapsulate both data and processes.
  - It makes use of the modeling techniques such as class diagrams, sequence diagrams etc.

Process Models

Software Development Methodology

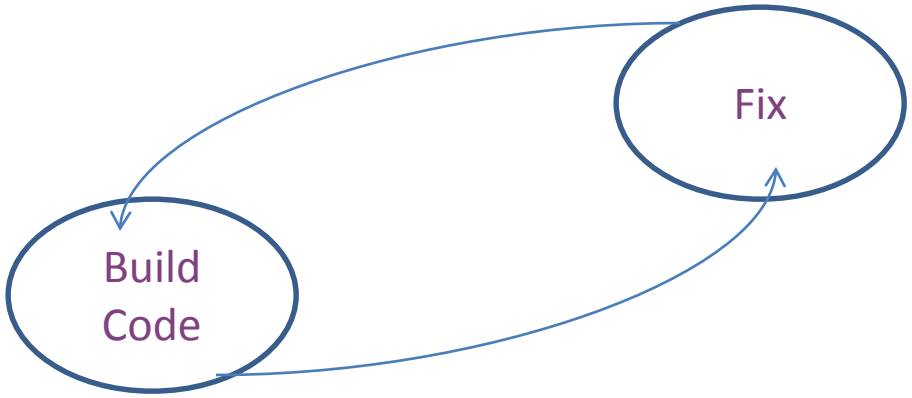
- Build and Fix Model



  - Software is constructed without concrete specifications and design
  - It is reworked several times to satisfy the client
  - It is an adhoc approach and not well defined.
  - It is a simple two phase model
  - The first phase writes code and the next phase fixes it

Software Development Methodology

- Build and Fix Model



  - Fix: error correction or adding a functionality
  - Suitable for small programming exercises with 100 – 200 Lines of Code (LOC)
  - Disadvantages : Code soon becomes unmanageable and unfixable. Maintenance is very difficult .

Verification Vs. Validation

Verification	Validation
Are we building the system right?	Are we building the right system?
Verification is the process of evaluating products of a development phase to find out whether they meet the specified requirements.	Validation is the process of evaluating software at the end of the development process to determine whether software meets the customer expectations and requirements.
The objective of Verification is to make sure that the product being develop is as per the requirements and design specifications.	The objective of Validation is to make sure that the product actually meet up the user's requirements, and check whether the specifications were correct in the first place.
Following activities are involved in Verification: Reviews, Meetings and Inspections.	Following activities are involved in Validation: Testing like black box testing, white box testing etc.

Verification Vs. Validation

Verification	Validation
Verification is carried out by QA team to check whether implementation software is as per specification document or not.	Validation is carried out by testing team.
Execution of code does not come under Verification.	Execution of code comes under Validation.
Verification process explains whether the outputs are according to inputs or not.	Validation process describes whether the software is accepted by the user or not.
Verification is carried out before the Validation.	Validation activity is carried out just after the Verification.

Verification Vs. Validation

Verification	Validation
Following items are evaluated during Verification: Plans, Requirement Specifications, Design Specifications, Code, Test Cases etc,	Following item is evaluated during Validation: Actual product or Software under test.
Cost of errors caught in Verification is less than errors found in Validation.	Cost of errors caught in Validation is more than errors found in Verification.
It is basically manually checking the of documents and files like requirement specifications etc.	It is basically checking of developed program based on the requirement specifications documents & files.

References

- Bob Hughes and Mike Cotterell, “Software Project Management ”, Tata McGraw Hill, 4th edition, 2006 .
- Software Project Management, Tutorialspoint. https://www.tutorialspoint.com/software_engineering/software_project_management.htm (accessed on 18th July 2020).