COLLEGE OF
COMPUTER STUDIES

QUEZON CITY
UNIVERSITY

WEEK 2-3 INTRODUCTION TO SOFTWARE ENGINEERING

SE101-SOFTWARE ENGINEERING

v2020

LEARNING OUTCOMES:

- Explain the concept of a software life cycle and provide an example, illustrating its phases including the deliverables that are produced.
- Select, with justification the software development models and process elements most appropriate for the development and maintenance of a diverse range of software products.
- Analyze and evaluate a set of tools in a given area of software development (e.g., management, modeling, or testing).
- Demonstrate the capability to use a range of software tools in support of the development of a software product

What is Software Engineer?

Why is it important?

Software engineering fundamentals

1. Systems should be developed using a **managed and understood development process**.
2. **Dependability and performance** are important for all types of system.
3. Understanding and managing the **software specification and requirements** (what the software should do) are important.
4. Where appropriate, you should **reuse software** that has already been developed rather than write new software.

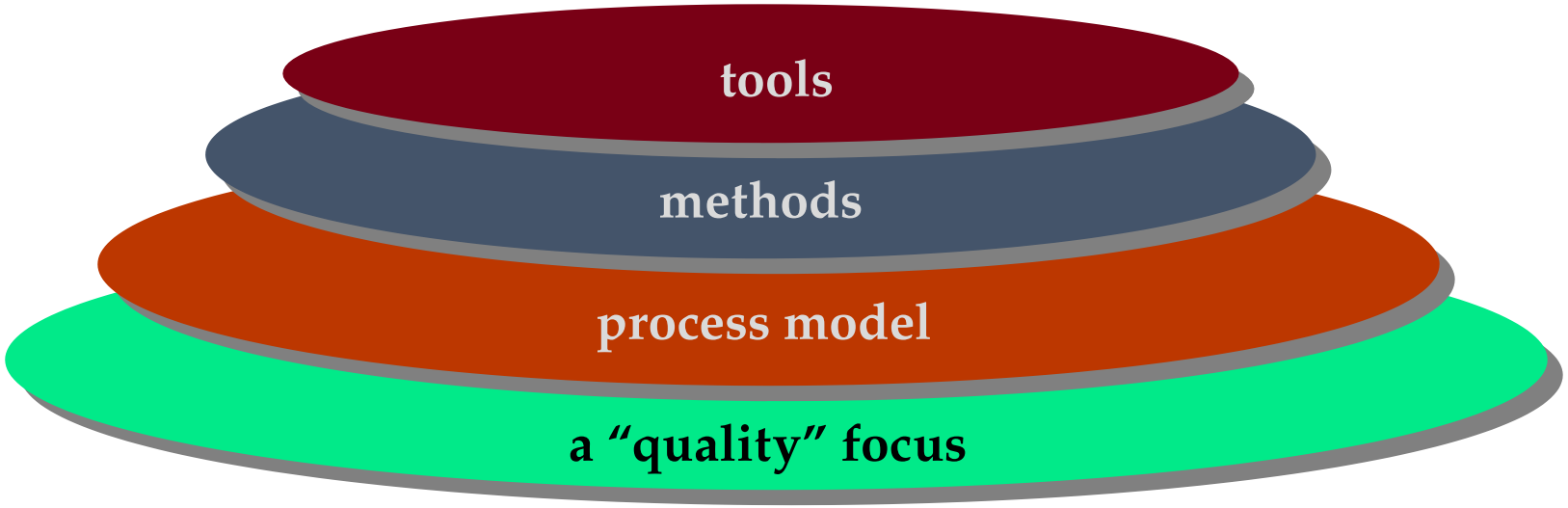
Software Engineering Ethics

- Software engineering involves **wider responsibilities** than simply the application of technical skills.
- Software engineers must behave in **an honest and ethically** responsible way if they are to be respected as professionals.
- **Ethical behaviour** is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of professional responsibility

- Confidentiality
- Competence
- Intellectual property rights
- Computer misuse

A Layered Technology



Software Engineering

What is Legacy Software? And Why must it change?

What is Software Process Model

Software process activities

- *Software specification*
- *Software development*
- *Software validation*
- *Software evolution*

Software Development Life Cycle models

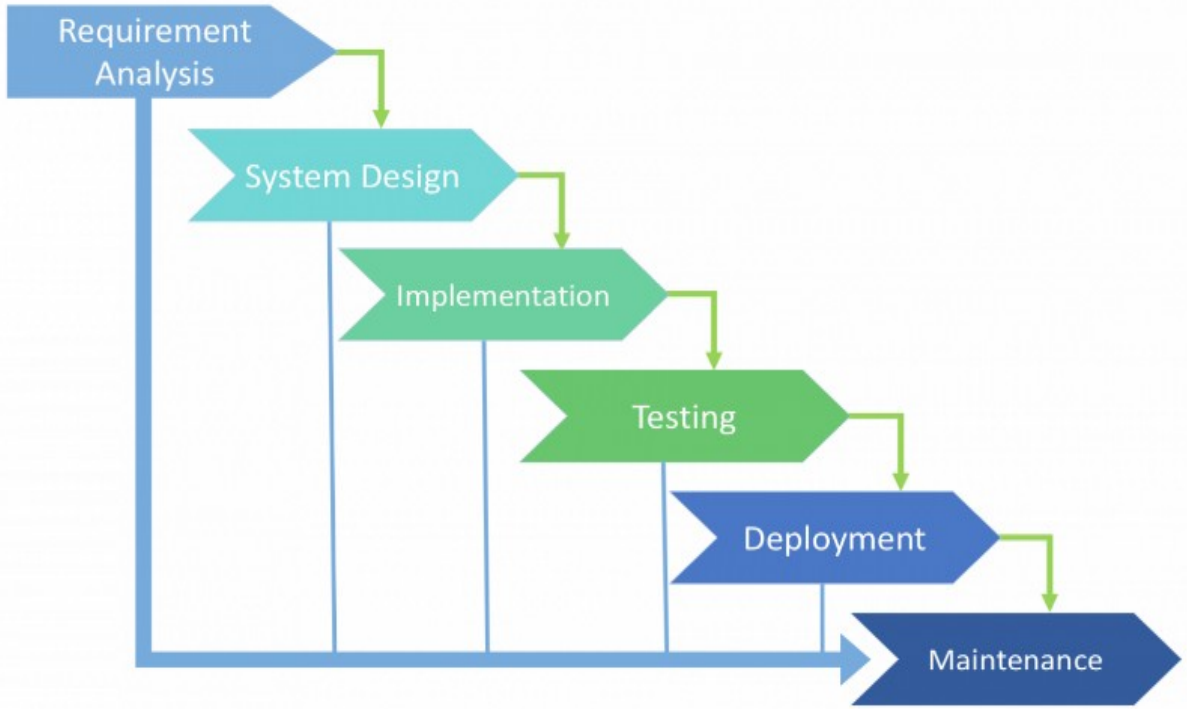
- is a continuous process, which starts from the moment, when it's made a decision to launch the project, and it ends at the moment of its full remove from the exploitation.
- The most used, popular and important SDLC models are given below:
  - Waterfall model
  - Iterative model
  - Spiral model
  - V-shaped model
  - Agile model

BASIC STAGES OF SOFTWARE DEVELOPMENT LIFE CYCLE

- Stage 1. Planning and requirement analysis
- Stage 2. Designing project architecture
- Stage 3. Development and programming
  - The programming by itself assumes four stages
    - Algorithm development
    - Source code writing
    - Compilation
    - Testing and debugging
- Stage 4. Testing
- Stage 5. Deployment

SDLC MODELS

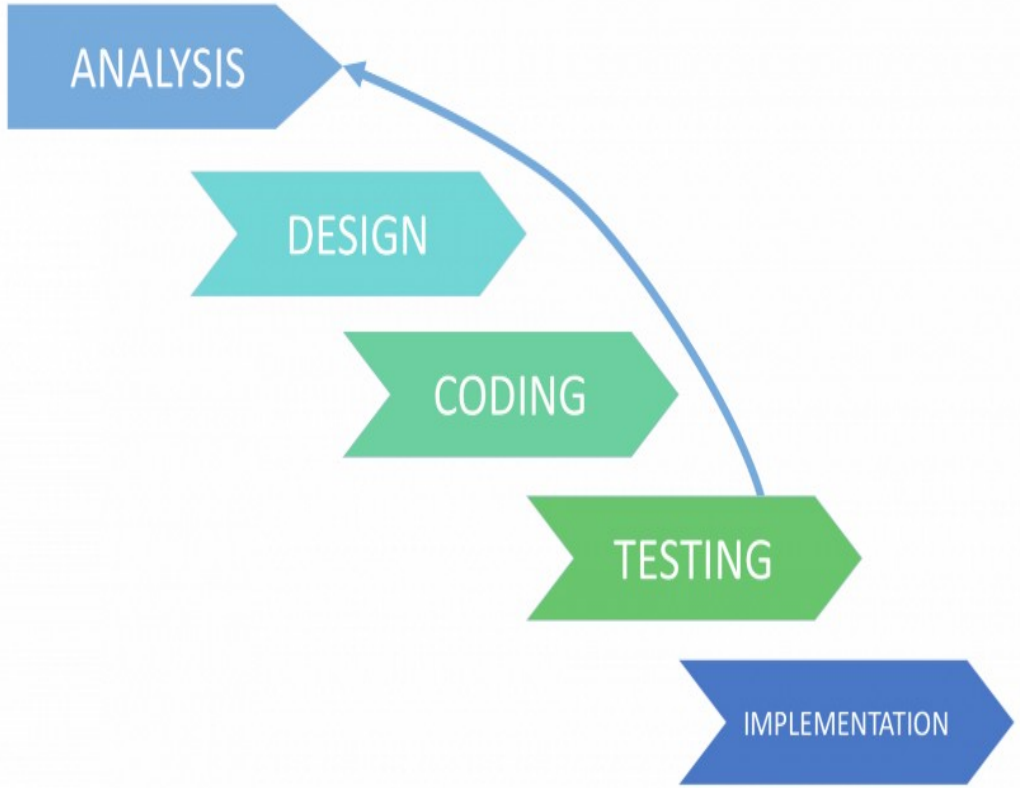
- Waterfall – is a cascade SDLC model, in which development process looks like the flow, moving step by step through the phases of analysis, projecting, realization, testing, implementation, and support.



ADVANTAGES	DISADVANTAGES
Simple to use and understand	The software is ready only after the last stage is over
Management simplicity thanks to its rigidity: every phase has a defined result and process review	High risks and uncertainty
Development stages go one by one	Not the best choice for complex and object-oriented projects
Perfect for the small or mid-sized projects where requirements are clear and not equivocal	Inappropriate for the long-term projects
Easy to determine the key points in the development cycle	The progress of the stage is hard to measure while it is still in the development
Easy to classify and prioritize tasks	Integration is done at the very end, which does not give the option of identifying the problem in advance

SDLC MODELS

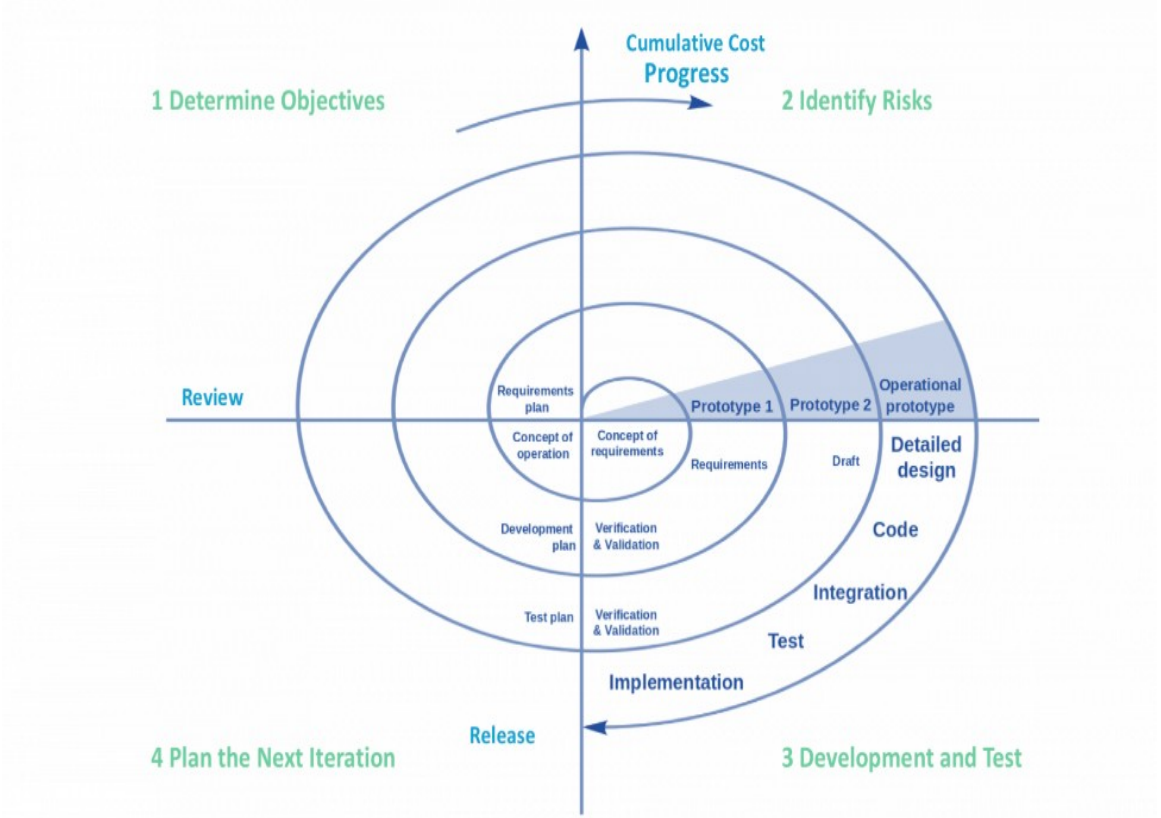
- The Iterative SDLC model does not need the full list of requirements before the project starts. The development process may start with the requirements to the functional part, which can be expanded later. The process is repetitive, allowing to make new versions of the product for every cycle.



ADVANTAGES	DISADVANTAGES
Some functions can be quickly developed at the beginning of the development lifecycle	Iterative model requires more resources than the waterfall model
The paralleled development can be applied	Constant management is required
The progress is easy measurable	Issues with architecture or design may occur because not all the requirements are foreseen during the short planning stage
The shorter iteration is - the easier testing and debugging stages are	Bad choice for the small projects
It is easier to control the risks as high-risk tasks are completed first	The process is difficult to manage
Problems and risks defined within one iteration can be prevented in the next sprints	The risks may not be completely determined even at the final stage of the project
Flexibility and readiness to the changes in the requirements	Risks analysis requires involvement of the highly-qualified specialists

SDLC MODEL

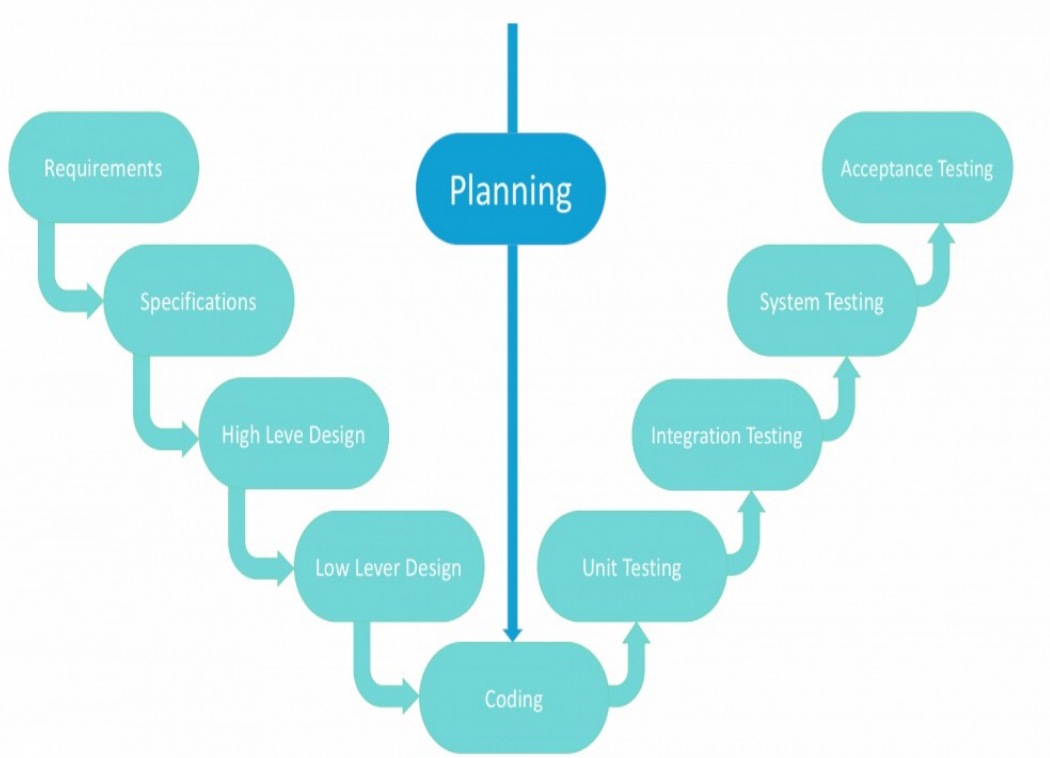
- Spiral model – is SDLC model, which combines architecture and prototyping by stages. It is a combination of the Iterative and Waterfall SDLC models with the significant accent on the risk analysis. The main issue of the spiral model – is defining the right moment to make a step into the next stage.



ADVANTAGES	DISADVANTAGES
Lifecycle is divided into small parts, and if the risk concentration is higher, the phase can be finished earlier to address the treats	Can be quite expensive
The development process is precisely documented yet scalable to the changes	The risk control demands involvement of the highly-skilled professionals
The scalability allows to make changes and add new functionality even at the relatively late stages	Can be ineffective for the small projects
The earlier working prototype is done - sooner users can point out the flaws	Big number of the intermediate stages requires excessive documentation

SDCL MODEL

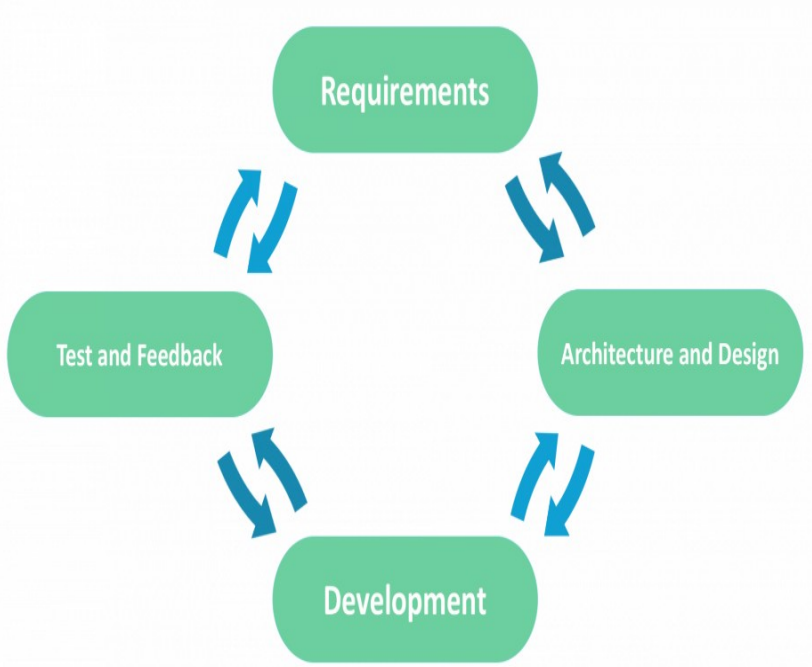
- V-shaped SDLC model is an expansion of classic waterfall model and it’s based on associated test stage for the every development stage. This is a very strict model and the next stage is started only after the previous phase. This is also called “Validation and verification” model.



ADVANTAGES	DISADVANTAGES
Every stage of V-shaped model has strict results so it's easy to control	Lack of the flexibility
Testing and verification take place in the early stages	Bad choice for the small projects
Good for the small projects, where requirements are static and clear	Relatively big risks

SDLC MODEL

- In the agile methodology after every development iteration, the customer is able to see the result and understand if he is satisfied with it or he is not. This is one of the advantages of the agile software development life cycle model. One of its disadvantages is that with the absence of defined requirements it is difficult to estimate the resources and development cost. Extreme programming is one of the practical use of the agile model. The basis of such model consists of short weekly meetings – Sprints which are the part of the Scrum approach.

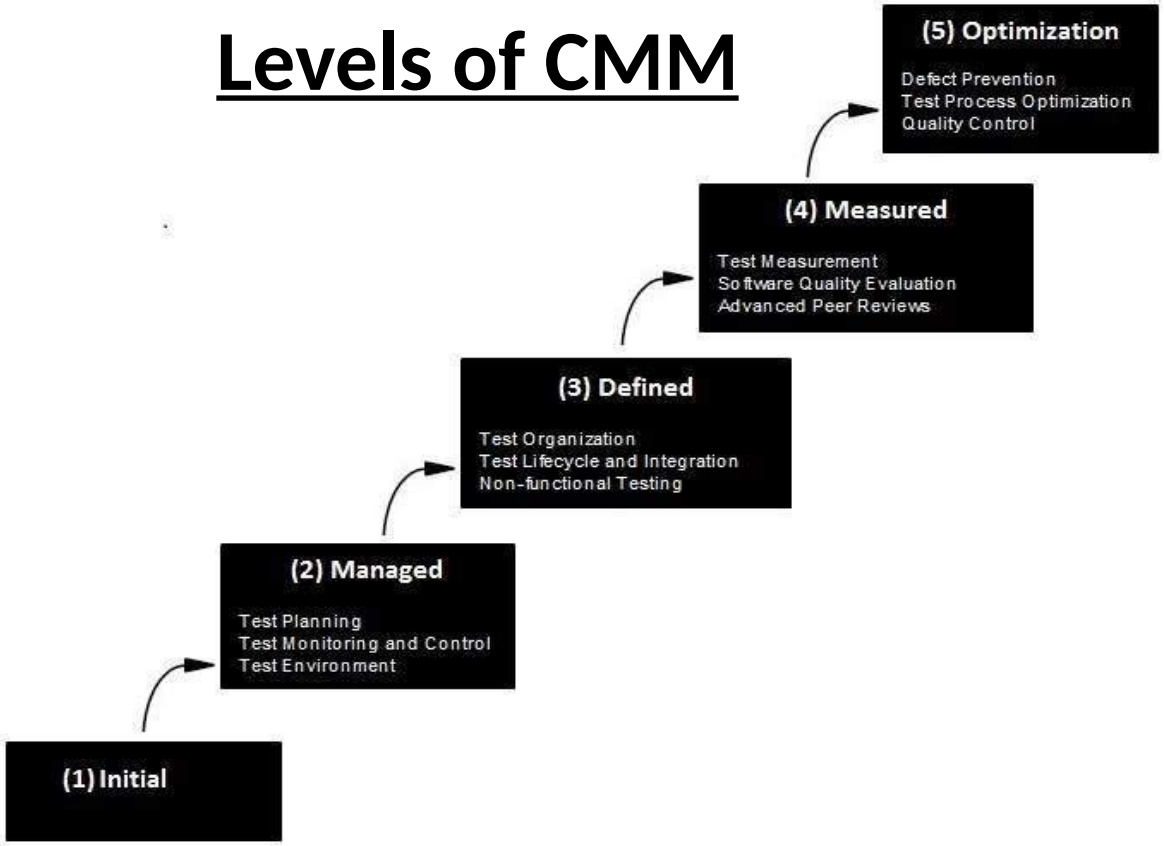


ADVANTAGES	DISADVANTAGES
Corrections of functional requirements are implemented into the development process to provide the competitiveness	Difficulties with measuring the final cost because of permanent changes
Project is divided by short and transparent iterations	The team should be highly professional and client-oriented
Risks are minimized thanks to the flexible change process	New requirements may conflict with the existing architecture
Fast release of the first product version	With all the corrections and changes there is possibility that the project will exceed expected time

What is Capability Maturity Model?

- The Software Engineering Institute (SEI) Capability Maturity Model (CMM) specifies an increasing series of levels of a software development organization. The higher the level, the better the software development process, hence reaching each level is an expensive and time-consuming process.

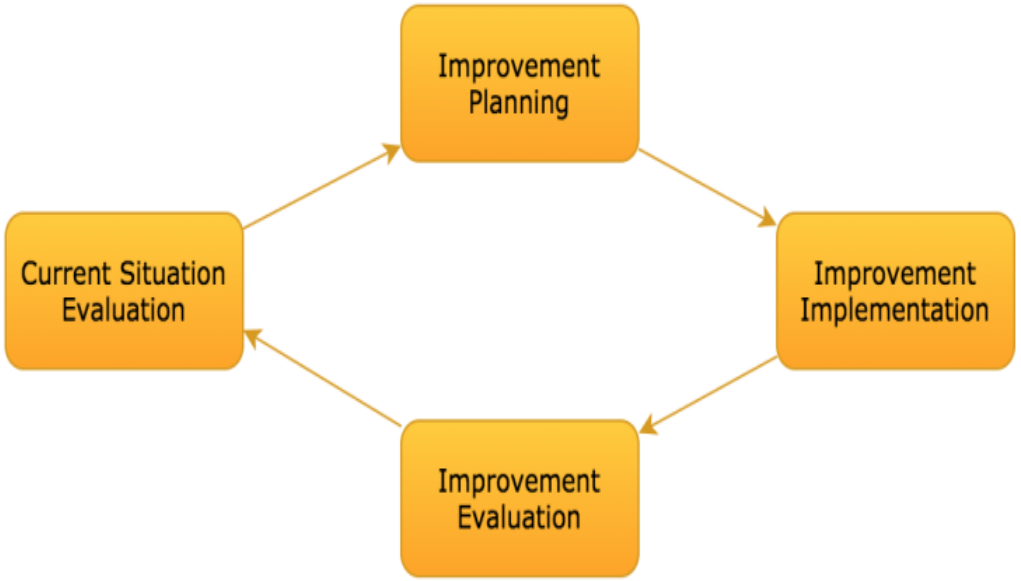
Levels of CMM



What is SPI?

- Software Process Improvement (SPI) methodology is defined as a sequence of tasks, tools, and techniques to plan and implement improvement activities to achieve specific goals such as increasing development speed, achieving higher product quality or reducing costs.
- SPI can be considered as process re-engineering or change management project to detect the software development lifecycle inefficiencies and resolve them to have a better process. This process should be mapped and aligned with organizational goals and change drivers to have real value to the organization.

SPI mainly consists of 4 cyclic steps



Why are Companies Seeking SPI – The motivators?

1. Standardization and Process consistency
2. Cost Reduction
3. Competitive Edge
4. Meeting targets and reduce time to market
5. Improve customers satisfaction
6. Job satisfaction, Responsibilities, and Resource Management
7. Automation and Autonomy
8. Proven outcome

Software Measurement in Software Engineering

- Measurement helps in estimation, quality control, productivity assessment and project control throughout a software project. Also, measurement is used by software engineers to gain insight into the design and development of the work products. In addition, measurement assists in strategic decision-making as a project proceeds.
- Software measurements are of two categories:
  - Direct measures
  - Indirect measures
- Measurement process is characterized by a set of five activities:
  - **Formulation**
  - **Collection**
  - **Analysis**
  - **Interpretation**
  - **Feedback**

Software Process Assessment

- is a disciplined examination of the software processes used by an organization, based on a process model. The assessment includes the identification and characterization of current practices, identifying areas of strengths and weaknesses, and the ability of current practices to control or avoid significant causes of poor (software) quality, cost, and schedule.
- Software Assessment (or audit) can be of three types.
  - **self-assessment (first-party assessment)**
  - **second-party assessment**
  - **third-party assessment**

Software Process Maturity Assessment

- The scope of a software process assessment can cover all the processes in the organization, a selected subset of the software processes, or a specific project. Most of the standard-based process assessment approaches are invariably based on the concept of process maturity. When the assessment target is the organization, the results of a process assessment may differ, even on successive applications of the same method.
- There are two reasons for the different results:
  - The organization being investigated must be determined. For a large company, several definitions of organization are possible and therefore the actual scope of appraisal may differ in successive assessments.
  - Even in what appears to be the same organization, the sample of projects selected to represent the organization may affect the scope and outcome..

Software Process Assessment Cycle According to Paulk and colleagues (1995), the CMM-based assessment approach uses a six-step cycle:

- Select a team
- The representatives of the site to be appraised complete the standard process maturity questionnaire.
- The assessment team performs an analysis of the questionnaire responses and identifies the areas that warrant further exploration according to the CMM key process areas.
- The assessment team conducts a site visit to gain an understanding of the software process followed by the site.
- The assessment team produces a list of findings that identifies the strengths and weakness of the organization's software process.
- The assessment team prepares a Key Process Area (KPA) profile analysis and presents the results to the appropriate audience.

Software Engineering Tools and Methods

Software Engineering Tools

A. Software Requirements

- *Requirements modeling tools*
- *Traceability tools*

B. Software Design Tools.

C. Software Construction Tools.

- *Program editors*
- *Compilers and code generators*
- *Interpreters*
- *Debuggers*

D. Software Testing Tools.

- *Test generators*
- *Test execution frameworks*
- *Test evaluation tools*
- *Test management tools*
- *Performance analysis tools*

E. Software Maintenance Tools.

- *Comprehension tools*
- *Re-engineering tools*

F. Software Engineering Process Tools

- *Process modeling tools*
- *Process management tools*
- *Integrated CASE environments*
- *Process-centered software engineering environments*

Software Engineering Tools

G. Software Quality Tools

- *Inspection tools*
- *Static analysis tools*

H. Software Configuration Management Tools.

- *Defect, enhancement, issue and problem tracking tools*
- *Version management tools*
- *Release and build tools*

I. Software Engineering Management Tools.

- *Project planning and tracking tools*
- *Risk management tools*
- *Measurement tools*

J. Infrastructure support tools.

- *Interpersonal communication tools*
- *Information retrieval tools*
- *System administration and support tools*

- **K. Miscellaneous tool issues.**
  - *Tool integration techniques*
  - *Meta tools*
  - *Tool evaluation*

Software Development Methods

- The software development section is divided into four subsections: *heuristic methods* dealing with informal approaches,
- *formal methods* dealing with mathematically based approaches,
- *prototyping methods* dealing with software development approaches based on various forms of prototyping, and
- *miscellaneous method issues*. The first three subsections are not disjoint; rather they represent distinct concerns.

References:

- eBook
  - Software Engineering A Practioner's Approach by Pressman and Maxim
  - Software Engineering by Sommerville
- Journal:
  - Software Engineering Tools and Methods by Carrington
- Online Resources:
  - https://www.existek.com/blog/sdlc-models/
  - https://www.tutorialspoint.com/software_testing_dictionary/capability_maturity_model.htm
  - https://melsatar.blog/2018/06/26/the-software-process-improvement-spi-reward-or-risk/
  - **http://ecomputernotes.com/software-engineering/software-measurement**
  - https://www.tutorialspoint.com/software_quality_management/software_quality_management_process_assessment.htm

COLLEGE OF
COMPUTER STUDIES

QUEZON CITY
UNIVERSITY

End of Lesson

v2020