BINUS UNIVERSITY

People
Innovation
Excellence

Course : COMP6100/Software Engineering

Effective Period : Desember 2017

The Nature of Software and Software Engineering

Session 02

Acknowledgement

These slides have been adapted from Pressman, R.S. (2015). *Software Engineering : A Practioner's Approach. 8th ed*. McGraw-Hill Companies.Inc, Americas, New York. ISBN : 978 1 259 253157. Chapter 1 and 2

Learning Objectives

LO1 : Describe the concepts of software process models and the opportunity for potential business project

People
Innovation
Excellence

BINUS UNIVERSITY

Contents

- The Nature of Software
- The Changing Nature of Software
- The Software Process
- Software Engineering Practice
- Software Myths

People
Innovation
Excellence

The Nature of Software

What is Software ?

- **Instructions** (computer programs)
  - that when executed provide desired features, function, and performance;
- **Data structures**
  - that enable the programs to adequately manipulate information and
- **Documentation**
  - that describes the operation and use of the programs.

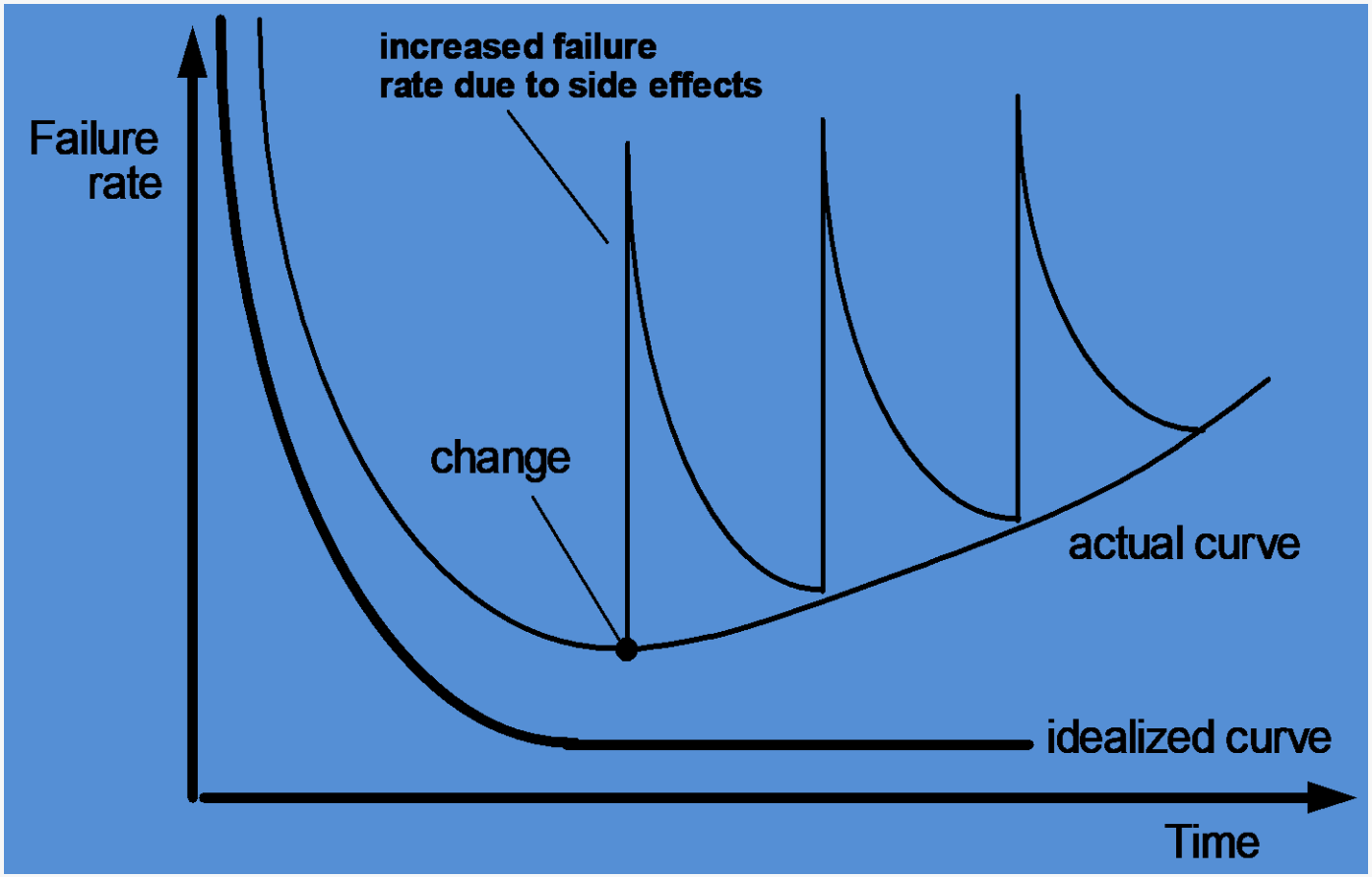
The Nature of Software

Characteristics of Software

- Software is developed or engineered it is not manufactured in the classical sense.
- Software doesn't "wear out"
- Although the industry is moving toward component-based construction, most software continues to be custom-built.

The Nature of Software

Wear vs. Deterioration



People
Innovation
Excellence

The Nature of Software

Software Application Domains

- System software
- Application software
- Engineering/scientific software
- Embedded software
- Product-line software
- Web/Mobile Application
- Arftificial intelligence software

The Nature of Software

Legacy Software

Why must software change?

- software must be **adapted** to meet the needs of new computing environments or technology.
- software must be **enhanced** to implement new business requirements.
- software must be **extended** to make it interoperable with other more modern systems or databases.
- software must be **re-architected** to make it viable within a network environment.

The Changing Nature of Software

Four broad categories of software are evolving to dominate the industry.

1. **WebApps**
2. **Mobile Applications**
3. **Cloud Computing**
4. **Product Line Software**

The Changing Nature of Software

WebApps

- In the early days of the World Wide Web, website consisted of little more than a set of limited hypertext files
- Today, WebApps have evolved into sophisticated computing tools that have been integrated with corporate databases and business applications
- They provide full computing potential in many of the application categories as explained before

The Changing Nature of Software

Mobile Applications

- The term app has evolved to connote software that has been specifically designed to reside on mobile platform (e.g., iOS, Android, or Windows Mobile)
- It is important to recognize that there is a subtle distintion between mobile web applications and mobile apps.
- A mobile web application (WebApp) allows a mobile device to gain access to web-based content via browser that has been specifically designed to accommodate the strengths and weaknesess of the mobile platforms.
- A mobile app can gain direct access to the hardware characteristics of the device (e.g., accelerometer or GPS location) and then provide the local processing and storage capabilities noted earlier

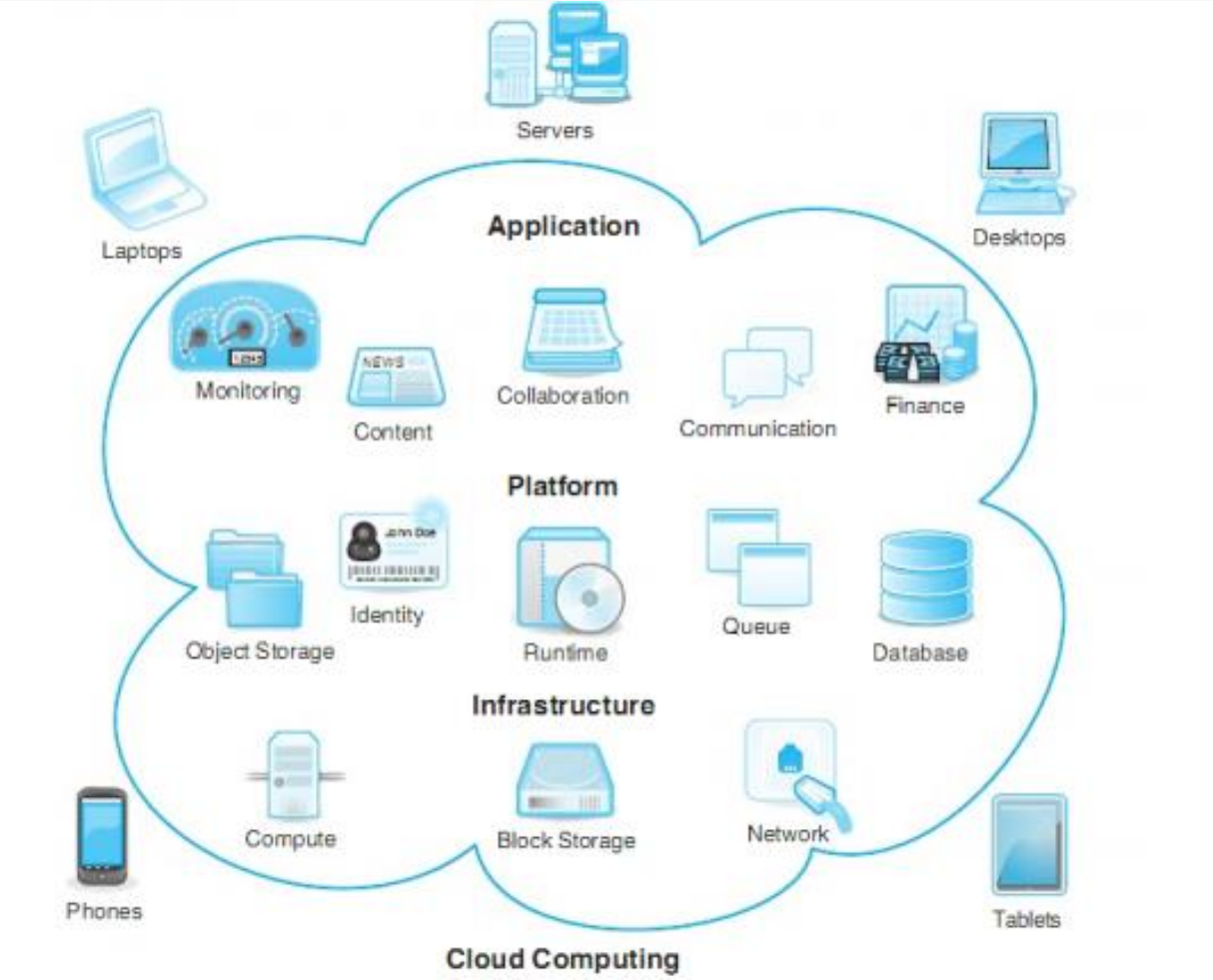
The Changing Nature of Software

Cloud Computing

- Cloud computing encompasses an infrastructure or 'ecosystem' that enables any user, anywhere, to use a computing device to share computing resources on a broad scale
- The computing devices reside outside the cloud and have access to a variety of resources within the cloud.
- These resources encompass applications, platforms, and infrastructure.
- In its simplest form, an external computing device access the cloud via a Web browser or analogous software.

The Changing Nature of Software

Cloud Computing



People
Innovation
Excellence

The Changing Nature of Software

Product Line Software

- The Software Engineering Institute defines a software product line as " a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of particular market segment or mission and that are developed from a common set of core assets in a prescribed way" [SEI13]
- In essence, a software product line results in the development of many products that are engineered by capitalizing on the comonality among all the products within the product line

Software Engineering

- **The IEEE definition:**

  (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and *maintenance of software; that is, the application of engineering to software.*

  (2) The study of approaches as in (1).

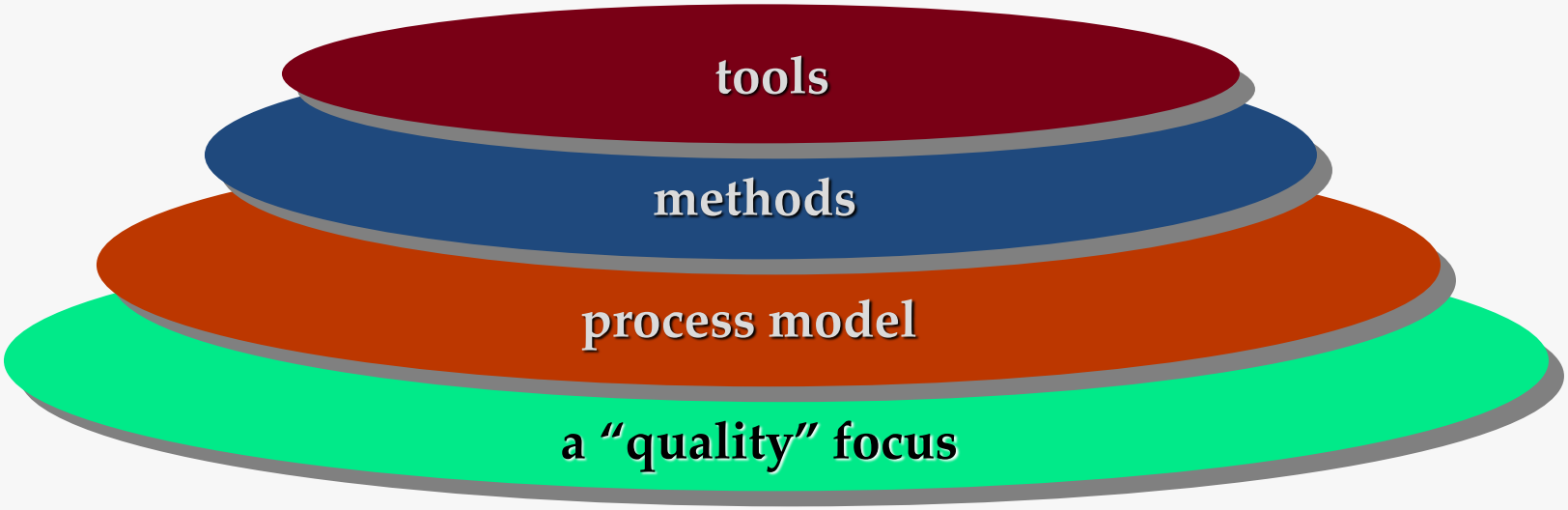
People
Innovation
Excellence

BINUS UNIVERSITY

People
Innovation
Excellence

The Software Process

A Layered Technology



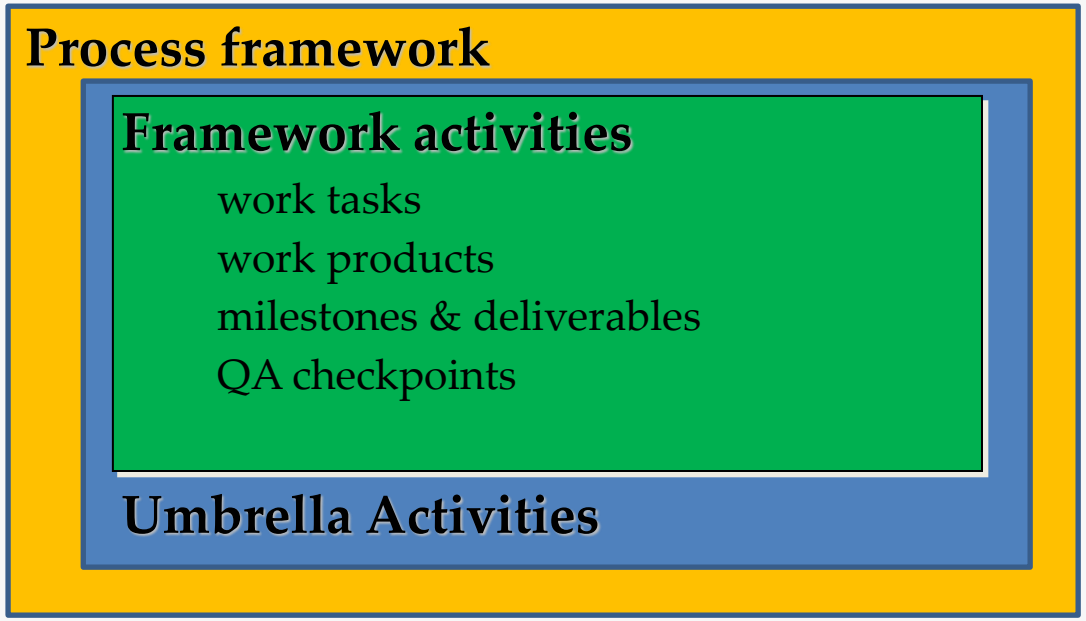
Software Engineering

BINUS UNIVERSITY

People
Innovation
Excellence

The Software Process

A Process Framework



The Software Process

Framework Activities

- Communication
- Planning
- Modeling
  - Analysis of requirements
  - Design
- Construction
  - Code generation
  - Testing
- Deployment

The Software Process

Umbrella Activities

- Software project tracking and control
- Risk management
- Software quality assurance
- Technical reviews
- Measurement
- Software configuration management
- Reusability management
- Work product preparation and production

The Software Process

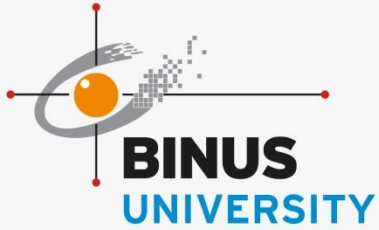
Process Adaptation

- overall flow of activities, actions, and tasks and the interdependencies among them
- degree to which actions and tasks are defined within each framework activity
- degree to which work products are identified and required
- manner which quality assurance activities are applied
- manner in which project tracking and control activities are applied
- overall degree of detail and rigor with which the process is described
- degree to which the customer and other stakeholders are involved with the project
- level of autonomy given to the software team
- degree to which team organization and roles are prescribed

Software Engineering Practice

- **Polya** suggests:
  1. ***Understand the problem***
     (communication and analysis).
  2. ***Plan a solution***
     (modeling and software design).
  3. ***Carry out the plan***
     (code generation).
  4. ***Examine the result for accuracy***
     (testing and quality assurance).

People
Innovation
Excellence



Software Engineering Practice

Understand the Problem

- *Who has a stake in the solution to the problem?*
  - That is, who are the stakeholders?
- *What are the unknowns?*
  - What data, functions, and features are required to properly solve the problem?
- *Can the problem be compartmentalized?*
  - Is it possible to represent smaller problems that may be easier to understand?
- *Can the problem be represented graphically?*
  - Can an analysis model be created?

Software Engineering Practice

Plan the Solution

- ***Have you seen similar problems before?***
  - Are there patterns that are recognizable in a potential solution? Is there existing software that implements the data, functions, and features that are required?
- ***Has a similar problem been solved?***
  - If so, are elements of the solution reusable?
- ***Can subproblems be defined?***
  - If so, are solutions readily apparent for the subproblems?
- ***Can you represent a solution in a manner that leads to effective implementation?***
  - Can a design model be created?

Software Engineering Practice

Carry Out the Plan

- ***Does the solution conform to the plan?***
  - **Is source code traceable to the design model?**
- ***Is each component part of the solution provably correct?***
  - **Has the design and code been reviewed, or better, have correctness proofs been applied to algorithm?**

Software Engineering Practice

Examine the Result

- ***Is it possible to test each component part of the solution?***
  - **Has a reasonable testing strategy been implemented?**
- ***Does the solution produce results that conform to the data, functions, and features that are required?***
  - **Has the software been validated against all stakeholder requirements?**

BINUS UNIVERSITY

People
Innovation
Excellence

Hooker's General Principles

1: *The Reason It All Exists*

2: *KISS (Keep It Simple, Stupid!)*

3: *Maintain the Vision*

4: *What You Produce, Others Will Consume*

5: *Be Open to the Future*

6: *Plan Ahead for Reuse*

7: *Think!*

Software Myths

- Affect managers, customers (and other non-technical stakeholders) and practitioners
- Are believable because they often have elements of truth,

but …

- Invariably lead to bad decisions,

therefore …

- Insist on reality as you navigate your way through software engineering

References

- Pressman, R.S. (2015). ***Software Engineering : A Practioner's Approach. 8th ed***. McGraw-Hill Companies.Inc, Americas, New York. ISBN : 978 1 259 253157
- Did You Know, http://www.youtube.com/watch?v=cl9Wu2kWwSY
- Introduction to Software Engineering, http://www.**youtube**.com/watch?v=Z6f9ckEElsU

BINUS UNIVERSITY

Q & A

People
Innovation
Excellence