Software Process Introduction

Lecturer: Ngo Huy Bien
Software Engineering Department
Faculty of Information Technology
VNUHCM - University of Science
Ho Chi Minh City, Vietnam
nhbien@fit.hcmus.edu.vn

Objectives

- To present what is *software*
- To present *general software development life cycle model*
- To present *software process* concepts
- To *define a process* for a project

References

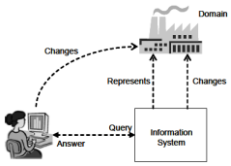
1. Roger S. Pressman. Software Engineering -- A Practitioner's Approach. 7th Edition. McGraw-Hill. 2010.
2. Antoni Olive. Conceptual Modeling of Information Systems. 2007.
3. Silvia T. Acuna et al. A process model applicable to software engineering and knowledge engineering. 1999.
4. NASA , Recommended Approach to Software Development. 1992.
5. Bill Curtis et al. Process Modeling. 1992.
6. R. M. Hillyer. Models of Software Evolution - Life Cycle and Process. 1990.
7. Silvia T. Acuna and Xavier Ferre. Software Process Modelling. 1998.

What is Software? [1]

- Software is (1) *instructions* (computer programs) that when executed provide desired function and performance, (2) *data structures* that enable the programs to adequately manipulate information, and (3) *documents* that describe the operation and use of the programs.

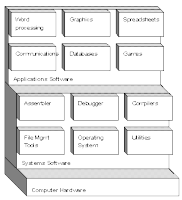
Information System [2]

- An *information system* is a designed system that collects, stores, processes, and distributes information about the state of a domain.
- A system is therefore considered to have three main functions:
  - *Memory*: to maintain a representation of the state of a domain.
  - *Informative*: to provide information about the state of a domain.
  - *Active*: to perform actions that change the state of a domain.

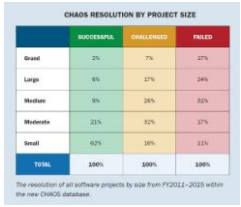


Software Applications

- System software
- Application software
- Engineering/scientific software
- Embedded software
- Product-line software
- Web applications
- AI software



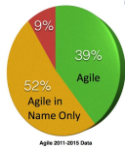
What Went Wrong?



CHAOS RESOLUTION BY PROJECT SIZE

	SUCCESSFUL	CHALLENGED	FAILED
Grand	2%	7%	27%
Large	6%	17%	24%
Medium	9%	26%	31%
Moderate	21%	32%	17%
Small	62%	18%	11%
TOTAL	100%	100%	100%

The resolution of all software projects by size from FY2011–2015 within the new CHAOS database.

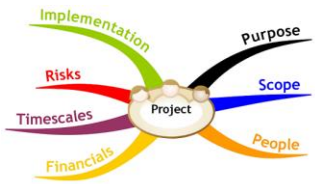


Managers are recognizing that *to be successful*, they need to be conversant with and use modern project management techniques. But *what* are the techniques?

Basic Management Questions

- Which step should we do **next**?
- How **long** will it take?
- **How** to perform the step?
- Which **artifacts** will it produce?
- **Who** is responsible for doing the step?

Software Project Planning [4, 6]



Roles and Products

Project Manager
Business Analyst
Technical Architect

System and Operations Concept Document
Project Plan

Clients, Users

Software Requirements [1]

The *elicitation*, *analysis*, *specification*, and *validation* of requirements for software.

1. Requirements *elicitation*
2. Requirements *analysis* and negotiation
3. Requirements *specification*
4. System *modeling*
5. Requirements *validation*
6. Requirements *management*

Roles and Products

- Business Analyst
- Users
- Project Manager

- Glossary
- Business rules
- Domain model
- Project vision
- Functional requirement specification
- Supplementary specification

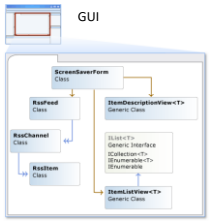
Software Design

- The *process of problem-solving* and planning for a software solution.
- After the purpose and specifications of software are determined, software developers *plan for a solution*.
- It includes *low-level component* and *algorithm implementation* issues as well as the *architectural view*.

Roles and Products

Architectural design specification

Technical Architect, Designer, Developer



Detailed design specification

Software Construction (Implementation)

The *construction of software* through the use of programming languages.

Roles and Products

Code File, Unit Tests

Module

Product

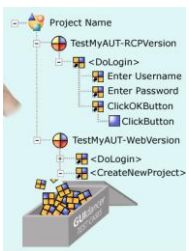
Development Plan
Integration Plan

Developer
Technical Architect

Software Testing

The *empirical investigations* conducted to provide stakeholders with information about the <u>quality</u> of the product or service under test, with respect to the context in which it is intended to operate.

Roles and Products



- ❑ Test Plan
- ❑ Test Data
- ❑ User Guide
- ❑ Test Result

QA/QC and Tester

Acceptance Testing Roles And Products

System Delivery

- ❑ Acceptance Test Plan
- ❑ Test Result

Clients, Users

Supporter and Development Team

Software Maintenance

- Software systems often have *problems* and need *enhancements* for a long time after they are first completed. This subfield deals with those problems.
- *Software maintenance* is the process of enhancing and optimizing deployed software (software release), as well as remedying defects.

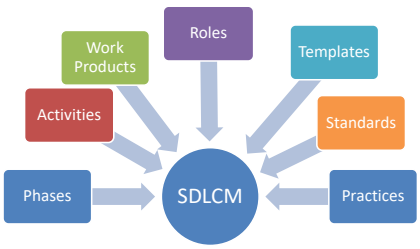
Roles and Products

Clients, Users, Supporters and Development Team

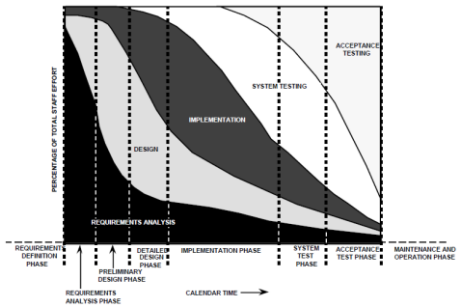
Why Take These Steps?

Software Development Life Cycle Model [3]

Software life cycle: all the states through which the software evolves.
A *software development life cycle model* is either a descriptive or prescriptive characterization of how software is or should be developed.

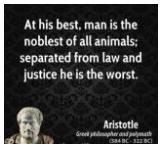


NASA/SEL SDLC Model [4]



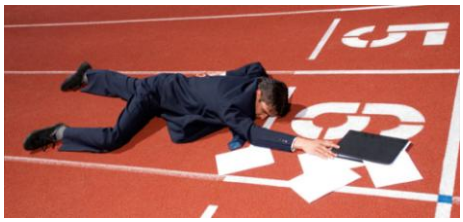
Why Describe or Define a SDLC Model?

- As *comparative descriptive* or *prescriptive* accounts for how software systems come to be the way they are.
- Establish a framework for building, implementing and enhancing systems that *all personnel* have to follow.
- As prescriptive outlines for *what products* to produce for approvals and delivery to client.
- As a basis for determining what software engineering *tools* and *methodologies* will be most appropriate to support different life cycle activities.
- As frameworks for analyzing or estimating patterns of *resource allocation* and consumption during the software life cycle.
- To organize, plan, staff, budget, schedule and manage *software project work* over organizational time, space, and computing environments.
- As a basis for conducting empirical studies to determine what affects *software productivity*, *cost*, and *overall quality*.
- Regulatory *compliance*



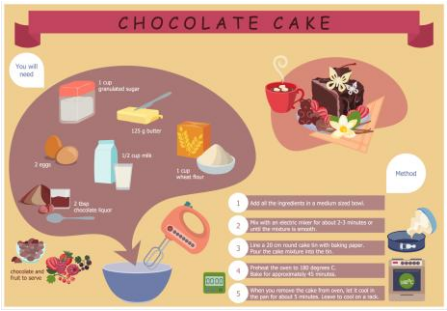
A Way to Guarantee the Failure

- Don't use a specific *methodology* because coding is all that is really important.



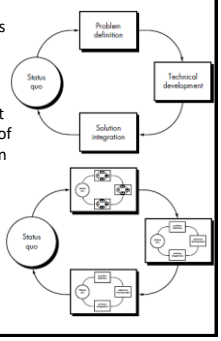
How to Execute a Phase?

Example



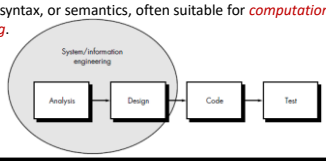
Process and Process Model [2, 5]

- A *process* is a set of partially ordered steps intended to reach a goal.
- A *process model* is an abstract description of an actual or proposed process that represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or machine.
- *Software process* is a partially ordered set of activities undertaken to manage, develop and maintain software systems.
- A *software process model* is an abstract representation of the software process.



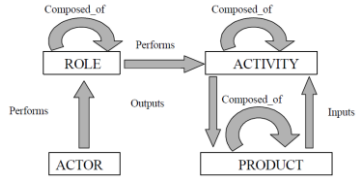
Software Process vs. SDLC

- The life cycle centers on the *product*, defining the states through which the product passes from the start of construction (the initial state is the user need) until the software is in operation (this product state is the deployed system) and finally retired (the state is the retired system).
- The software process centers on the *construction process* rather than on the product(s) outputted.
- Software process can be used to develop more *precise and formalized* descriptions of software life cycle activities.
- Software process power emerges from their utilization of a sufficiently rich notation, syntax, or semantics, often suitable for *computational processing*.



Elements of Process Model [7]

- *Activity* is the stage of a process that produces externally visible changes of state in the software product.
- *Artifact* or Product is the (sub)product and the "raw material" of a process.
- *Agent* or *Actor* is an entity that executes a process.
- *Role* describes a set of agent or group responsibilities, rights and skills required to perform a specific software process activity.



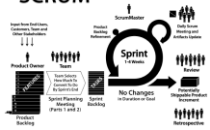
Software Process Objective

- The software process has a common objective: to *build* and *maintain* a software product that satisfies a need detected by a user.
- The integral software process model seeks to do just this: *define* a series of activities to be performed to produce software.

How to Define a Process?

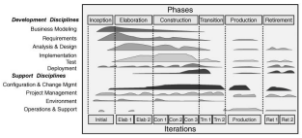
REVIEW

SCRUM



REVIEW

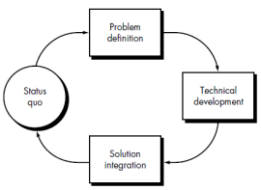
RUP (Rational Unified Process)

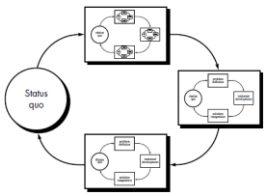


Process Definition Template

- *Life cycle* (hierarchical/overlapped phases)
- Phase
  - Purpose
  - *Entry criteria*
  - Inputs
  - Roles
    - Tasks
    - Process flow
    - Deliverables
  - Checkpoints
  - Outputs
  - *Exit criteria*

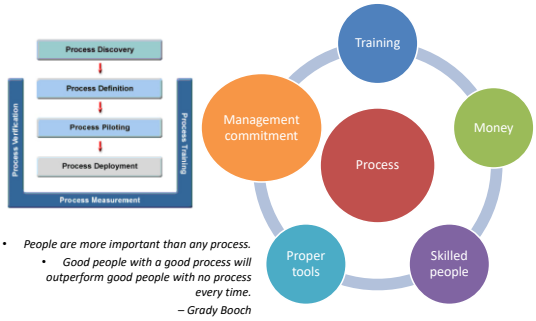
Problem Solving Loop [1]





Regardless of the process model that is chosen for a software project, all of the stages—status quo, problem definition, technical development, and solution integration—*coexist simultaneously* at some level of detail.

How to Use a Process?



- *People are more important than any process.*
  - *Good people with a good process will outperform good people with no process every time.*

– Grady Booch

Why Software Process?

- Bringing *discipline* to various tasks in software development.
  - This discipline leads to *consistency* and uniformity in products delivered at the end of these tasks.
  - Project *predictability* (framework for software development plan)
  - Better *communication* and learning
  - Preventing *wasted effort* and repeated errors
- Having *good processes* to follow vs. hiring only *brilliant people*
- Having people work *smarter* vs. having people work *harder*
- *Guarantee* for customers (schedule, budget and quality)

Thank You for Your Time