



[SF220-W02]



กระบวนการพัฒนาซอฟต์แวร์

Software Development Life Cycle (SDLC)

- ❖ **ผู้สอน:** อ.ดร.อัศวภูมิ ตาคุม
- ❖ **วัน:** ศุกร์ที่ 20 มกราคม พ.ศ. 2565
- ❖ **เวลา:** 9.30-12.30
- ❖ **สถานที่:** วศ.606/1

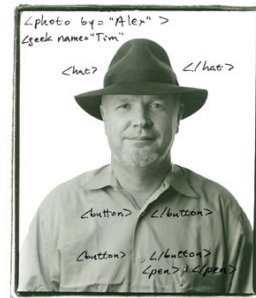
หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมซอฟต์แวร์ (Soft-en)

- **จุดเด่น** มุ่งเน้นการผลิตวิศวกรซอฟต์แวร์คุณภาพสูงที่มีความสามารถทั้งในเชิงทฤษฎีและปฏิบัติ
- **เนื้อหาโดยรวม** มุ่งเน้นให้นักศึกษาได้เรียนรู้และเข้าใจในระบบซอฟต์แวร์ทั้งในเชิงทฤษฎีและปฏิบัติ ตั้งแต่พื้นฐานจนกระทั่ง การพัฒนาการเขียนโปรแกรม การพัฒนาและทดสอบระบบ การบริหารจัดการโครงสร้างด้านซอฟต์แวร์ และมีความเข้าใจถึงการออกแบบและวิเคราะห์โปรแกรมเชิงวัสดุ และมีพัฒนาศักยภาพบัณฑิตผ่าน การฝึกงาน การทำโครงงาน/สหกิจศึกษา
- **ลักษณะพิเศษที่มุ่งพัฒนาให้มีขึ้นในตัวของนักศึกษา**
 - มุ่งพัฒนาบัณฑิตเพื่อให้เป็นวิศวกรซอฟต์แวร์ระดับชั้นนำของประเทศ ที่มีความรู้ ความสามารถทางด้านวิชาการ ด้านความคิดสร้างสรรค์ ด้านการค้นคว้าวิจัยและพัฒนา
 - ทักษะความสามารถในการผลิตซอฟต์แวร์ที่มีคุณภาพ เพื่อตอบสนองความต้องการบุคลากรที่มีคุณภาพในภาคอุตสาหกรรมซอฟต์แวร์ของไทยและนานาชาติ

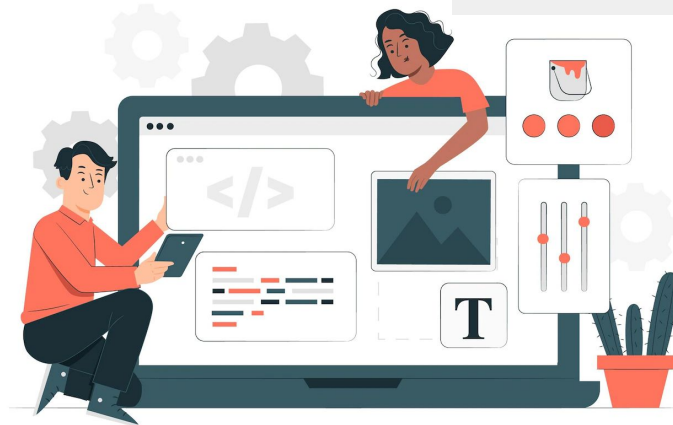









อาชีพที่สามารถประกอบได้หลังสำเร็จการศึกษา

- ❖ **Software Engineer** (วิศวกรซอฟต์แวร์) หรือ **Programmer/Developer** (นักเขียนโปรแกรม)
- ❖ **System Analyst / Designer** (นักวิเคราะห์ระบบหรือนักออกแบบระบบ)
- ❖ **Software Quality Assurance** (นักประกันคุณภาพซอฟต์แวร์)
- ❖ **Software Process Improvement Engineer** (วิศวกรปรับปรุงกระบวนการซอฟต์แวร์)
- ❖ **Software Tester** (นักทดสอบระบบ)
- ❖ **Software Architect** (สถาปนิกซอฟต์แวร์)
System Integrator (นักบูรณาการระบบ)



Tim Bray:
Father of XML, Uncle
of search engines



Software Engineer	Degree Held	Company	Application	Application Type
Tim Bray 	Bachelor of Science degrees in math and computer science	Entrepreneur; co-founded Open Text Corporation, Textuality and Antarctica Systems	XML	Developer's tool
Paul Buchheit 	Bachelor's degree in computer science	Entrepreneur; worked for Intel and Google	Gmail	Webmail service
Dave Cutler 	Bachelor's degree	DEC, Microsoft; co-founded Agrippa-Ord	Windows NT	Operating System
Max Levchin 	Bachelor's degree in computer science	Entrepreneur; co-founded NetMeridian Software, SponsorNet New Media and Confinity	PayPal	E-commerce
Pierre Omidyar 	Bachelor's degree in computer science	Claris; as an entrepreneur, co-founded Ink Development and created eBay	eBay	Online auction and shopping website
Jimmy Wales 	Bachelor's and master's degrees in finance	Teacher and Internet entrepreneur; co-founded Wikipedia	Wikipedia	Web-based encyclopedia
Michael Widenius 	None	Entrepreneur; co-founded MYSQL AB	MySQL	Open-source database

วัตถุประสงค์การเรียนรู้ | Learning Objectives

To introduce the idea of a software process — a coherent set of activities for software production.

- ❖ Understand the concepts of software processes and software process models เพื่อทำความรู้จักกับ กระบวนการพัฒนาซอฟต์แวร์
- ❖ Introduce to three general software process models and when they might be used เพื่อให้เข้าใจ แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ ในรูปแบบต่างๆ





SF220 Introduction to Software Engineering

[SF220-W02-Lecture]

กระบวนการพัฒนาซอฟต์แวร์ Software Development Life Cycle (SDLC)



SOFTWARE

Software Development Process in Software Engineering


 WIKIPEDIA
 The Free Encyclopedia

- ❖ **A software development process OR
A software development life cycle (SDLC)**
: A process of dividing software development work into smaller, parallel, or sequential steps or sub-processes to improve design and/or product management.

- ❖ **The methodology** may include the pre-definition of specific deliverables and artifacts
 - Created and completed by a project team to develop or maintain an application.

Part of a series on

Software development

Core activities [hide]

Processes · Requirements · Design · Construction · Engineering · Testing · Debugging · Deployment · Maintenance

Paradigms and models [show]

Methodologies and frameworks [show]

Supporting disciplines [show]

Practices [show]

Tools [show]

Standards and bodies of knowledge [show]

Glossaries [show]

Outlines [show]

V · T · E


- https://en.wikipedia.org/wiki/Software_development_process
- Centers for Medicare & Medicaid Services (CMS) Office of Information Service (2008). Selecting a development approach. Webarticle. United States Department of Health and Human Services (HHS).

Software & Software Process

- **Software** is the set of instructions in the form of programs
 - To govern the computer **system**
 - To process the **hardware** components
- To produce **a software product** the set of activities is used. This set is called **a software process**.

<https://www.geeksforgeeks.org/software-processes-in-software-engineering/>

- **The systematic approach** that is used in **software engineering** is sometimes called **a software process**.
 - A software process is a sequence of activities that leads to the production of a software product.
 - **Four fundamental activities** are common to all software processes.
[Sommerville, I. (2016)]


TU-PINE
 THAMMASAT SCHOOL OF ENGINEERING

วช.220 วิศวกรรมซอฟต์แวร์เบื้องต้น หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมซอฟต์แวร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยธรรมศาสตร์
 SF220 Introduction to Software Engineering by Akkarawoot Takhom, Ph.D. (takkhara@engr.tu.ac.th)

35

Software Engineering is related to both Computer Science and Systems Engineering.

วิทยาการคอมพิวเตอร์ (Computer Science)	วิศวกรรมระบบ (System Engineering)
<ul style="list-style-type: none"> วิทยาการคอมพิวเตอร์ เกี่ยวข้องกับทฤษฎีและวิธีการที่รองรับคอมพิวเตอร์และระบบซอฟต์แวร์ ในขณะที่ยังเกี่ยวข้องกับปัญหาในทางปฏิบัติของการผลิตซอฟต์แวร์ ความรู้บางอย่างเกี่ยวกับวิทยาการคอมพิวเตอร์ <ul style="list-style-type: none"> - จำเป็นสำหรับวิศวกรซอฟต์แวร์ในลักษณะเดียวกัน - เช่น ความรู้ทางฟิสิกส์บางอย่างเป็นสิ่งจำเป็นสำหรับวิศวกรไฟฟ้า - ทฤษฎีวิทยาการคอมพิวเตอร์มักใช้กับโปรแกรมที่มีขนาดค่อนข้างเล็ก ทฤษฎีพื้นฐานของวิทยาการคอมพิวเตอร์ <ul style="list-style-type: none"> - มักไม่ค่อยเกี่ยวข้องกับปัญหาขนาดใหญ่และซับซ้อนที่ต้องใช้กระบวนการแก้ไขทางซอฟต์แวร์ 	<ul style="list-style-type: none"> วิศวกรรมระบบ เกี่ยวข้องกับทุกแง่มุมของการพัฒนาและวิศวกรรมของระบบที่ซับซ้อน ซึ่งซอฟต์แวร์มีบทบาทสำคัญ วิศวกรรมระบบ เกี่ยวข้องกับ <ul style="list-style-type: none"> - การพัฒนาระบบ - การออกแบบนโยบายและการจัดการ และการปรับใช้ระบบ ตลอดจนวิศวกรรมซอฟต์แวร์ วิศวกรรมระบบมีส่วนร่วมใน <ul style="list-style-type: none"> - การระบุระบบ - กำหนดสถาปัตยกรรมโดยรวม - จากนั้นจึงรวมส่วนต่าง ๆ เพื่อสร้างระบบที่เสร็จสมบูรณ์

Ref*: Software engineering(2016), 10th edition, Ian Sommerville, Page 23.

[SF220-W01]

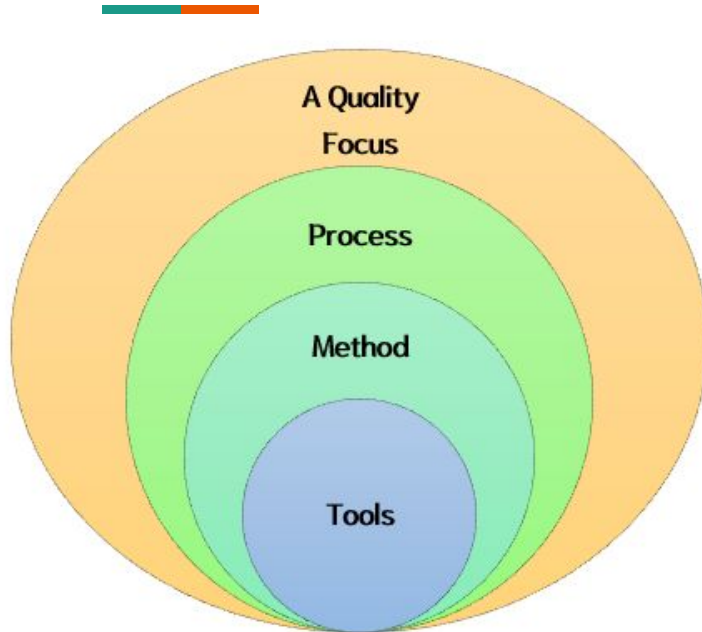
กิจกรรมพื้นฐาน 4 ประเภท | Four Fundamental Activities

There are many different kinds of **software processes**,
 BUT each and every one of them involve these **Four Types of Fundamental Activities**:

1. **Software Specification** (การจัดการข้อกำหนดซอฟต์แวร์)
 - Define what the system should do
 - Where customers and engineers define the software that is to be produced and the constraints on its operation.
2. **Software Design and Implementation** (การออกแบบและการผลิตซอฟต์แวร์)
 - Where the software is designed and programmed.
3. **Software Validation** (การตรวจสอบซอฟต์แวร์)
 - Where the software is checked to ensure that it is what the customer requires.
4. **Software Evolution** (การวิวัฒนาการของซอฟต์แวร์)
 - Where the software is modified to reflect changing customer and market requirements.



Four Parts of Layered Technology [Bruce R. and Maxim Dr., 2014]



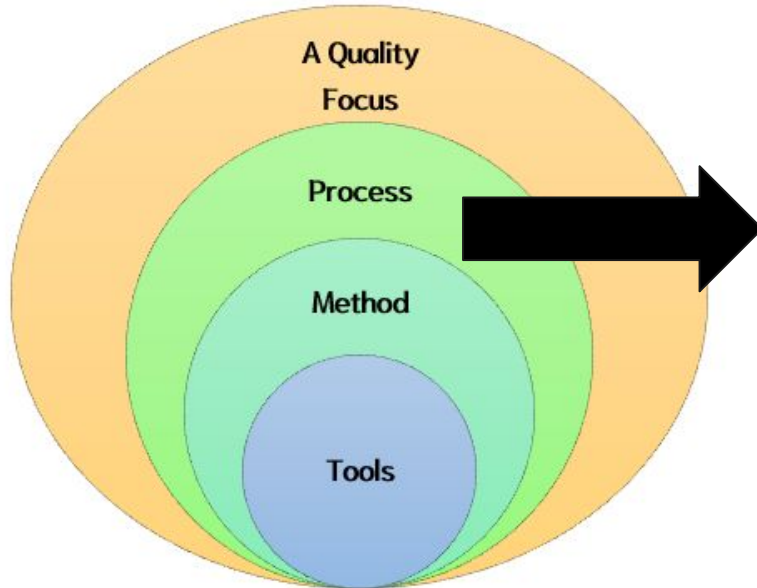
1. A Quality Focus

- Defines the continuous process improvement principles of software
- Provides integrity that means providing security to the software
- Data can be accessed by only an authorized person, no outsider can access the data
- Focuses on maintainability and usability

2. Process

- The foundation or base layer of software engineering.
- Key that binds all the layers together which enables the development of software before the deadline or on time.
- **Process** defines a framework that must be established for the effective delivery of software engineering technology.
- **The software process** covers all the activities, actions, and tasks required to be carried out for software development.

Four Parts of Layered Technology [Bruce R. and Maxim Dr., 2014]

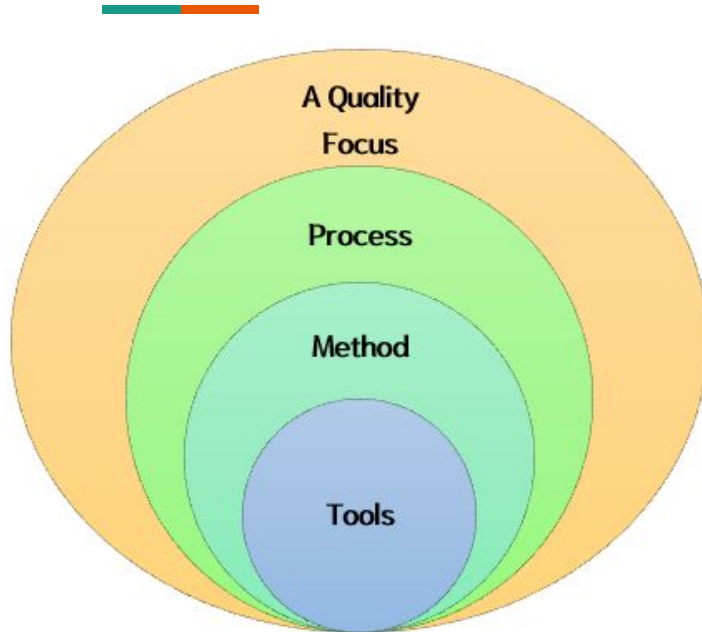


Five Process Activities of the Software Framework

1. **Communication (การสื่อสาร)**
 - Heavy communication with customers and other stakeholders
 - To gather requirements and other related activities.
2. **Planning (การวางแผน)**
 - Describe the technical tasks to be conducted, risks, required resources, work schedule etc.
3. **Modeling (การสร้างแบบจำลอง)**
 - Created a model to understand the requirements and design to achieve these requirements.
4. **Construction (การสร้างซอฟต์แวร์)**
 - Generated and tested code
5. **Deployment (การใช้งานซอฟต์แวร์)**
 - Represent a (complete/partially complete) software to the customers to evaluate and collect feedbacks based on their evaluation.



Four Parts of Layered Technology [Bruce R. and Maxim Dr., 2014]



3. Method

- During the process of software development the answers to all “how-to-do” questions are given by **method**.
- **Method** has the information of all the tasks which includes communication, requirement analysis, design modeling, program construction, testing, and support.

4. Tools

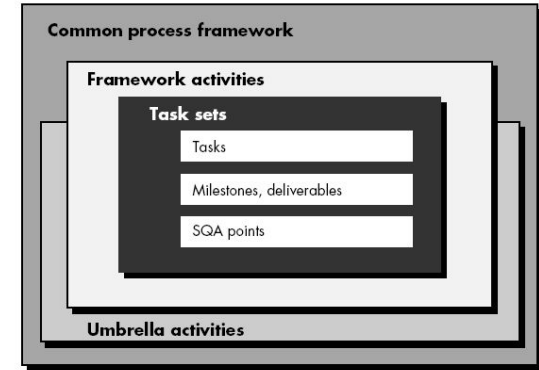
- **Software engineering tools** provide a self-operating system for **Processes** and **Methods**.
- **Tools** are integrated which means information created by one tool can be used by another.

ลักษณะของกระบวนการพัฒนาซอฟต์แวร์

1. กระบวนการต้องระบุกิจกรรมทั้งหมดอย่างชัดเจน
2. กระบวนการใช้ทรัพยากรภายใต้ข้อจำกัดต่าง ๆ เพื่อผลิตเป็นซอฟต์แวร์
3. กระบวนการหนึ่ง อาจเกิดจากกระบวนการย่อยอื่นที่มีความสัมพันธ์กัน
4. ทุกกิจกรรมของกระบวนการ มีเงื่อนไขในการเริ่มต้นและสิ้นสุดของกิจกรรม
5. ทุกขั้นตอนและทุกกิจกรรมของกระบวนการต้องมีเป้าหมายที่ชัดเจน
และมีหลักการหรือแนวทางในการปฏิบัติเพื่อให้บรรลุเป้าหมาย
6. ข้อจำกัดหรือเงื่อนไขสามารถนำมาใช้ควบคุมการดำเนินการของ
กิจกรรม หรือ จัดสรรทรัพยากรให้เกิดประโยชน์สูงสุด

Software Process Framework (กรอบงานของกระบวนการวิศวกรรมซอฟต์แวร์)

- **A process framework** establishes the foundation for a complete software process by identifying a small number of framework activities that are applicable to all software projects, regardless of size or complexity.
 - Includes a set of **umbrella activities** that are applicable across the entire software process.
 - Some most applicable framework activities are described below.

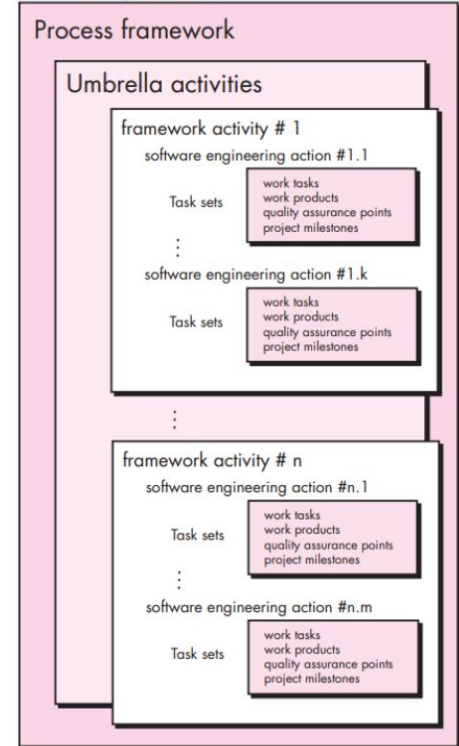


Software Process Framework (กรอบงานของกระบวนการวิศวกรรมซอฟต์แวร์)

Software Process Framework (A Generic Framework) is:

- An abstraction of the software development process.
- Details the steps and chronological order of a process.
- Since it serves as a foundation for them, it is utilized in most applications.
- Task sets, umbrella activities, and process framework activities all define the characteristics of the software development process.

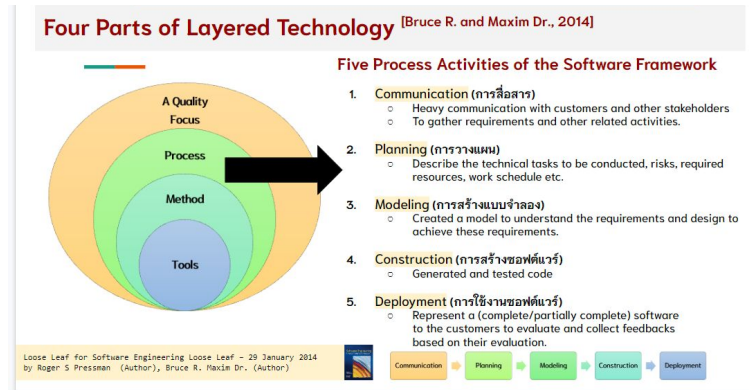
Software process



Software Process Framework (กรอบงานของกระบวนการวิศวกรรมซอฟต์แวร์)

- **The process framework** is required for representing common process activities.
- **Five framework activities** are described in a process framework for software engineering.

- Communication,
- planning,
- modeling,
- construction,
- deployment

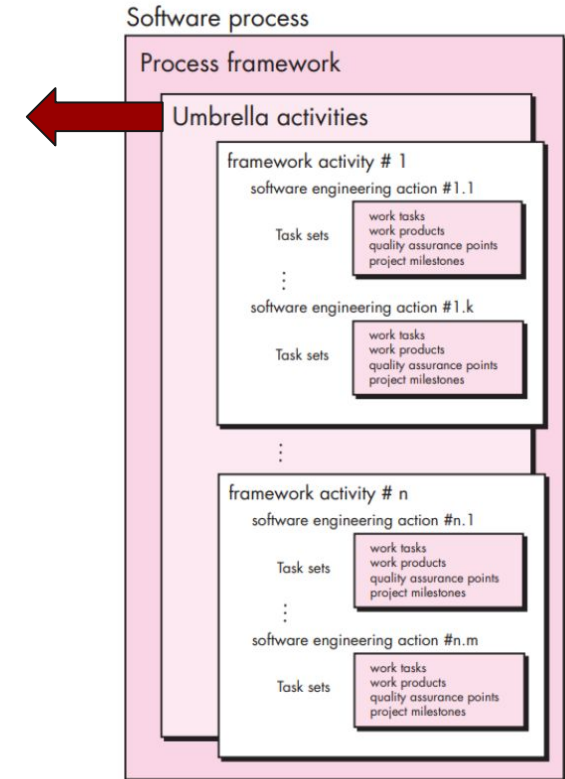


- **Each engineering action** defined by a framework activity comprises:
a list of needed work outputs, project milestones, and software quality assurance (SQA) points.

Umbrella Activities

Umbrella Activities are that take place during a software development process for improved project management and tracking.

1. **Software project tracking and control**
(การติดตามและควบคุมโครงการซอฟต์แวร์)
2. Risk management (การบริหารความเสี่ยง)
3. Software quality assurance (การประกันคุณภาพซอฟต์แวร์)
4. Formal technical reviews (บทวิจารณ์ทางเทคนิคอย่างเป็นทางการ)
5. Software configuration management
(การจัดการการกำหนดค่าซอฟต์แวร์)
6. Work product preparation and production
(การเตรียมผลิตภัณฑ์และการผลิตงาน)
7. Reusability management (การจัดการการใช้ซ้ำ)
8. Measurement (การวัดความพยายามที่จำเป็นในการวัดซอฟต์แวร์)





SF220 Introduction to Software Engineering

Models of Software Development Process

- **A process** was defined as a collection of work activities, actions, and tasks
 - Performed when some work product is to be created.
- Each of these activities, actions, and tasks reside within a framework or model that defines their relationship with the process and with one another.
- **Each software engineering action** is defined by a task set that identifies the work tasks that are to be completed, the work products that will be produced, the quality assurance points that will be required, and the milestones that will be used to indicate progress.

4 Types of Process Flow

One important aspect of the software process called **Process Flow**

- **Process Flow** describes how the framework activities and the actions and tasks that occur within each framework activity are organized with respect to sequence and time.
1. **Linear Process Flow** (การไหลของกระบวนการเชิงเส้น)
 2. **Interactive Process Flow** (การไหลของกระบวนการแบบวนซ้ำ)
 3. **Evolutionary Process Flow** (การไหลของกระบวนการแบบมีวิวัฒนาการ)
 4. **Parallel Process Flow** (การไหลของกระบวนการแบบขนาน)

4 Types of Process Flow

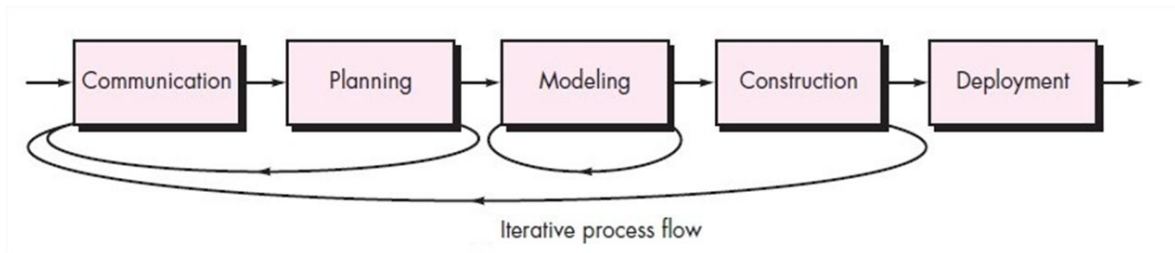
1. Linear Process Flow (การไหลของกระบวนการเชิงเส้น)

- Executes each of the five framework activities in sequence, beginning with communication and culminating with deployment.



2. Interactive Process Flow (การไหลของกระบวนการแบบวนซ้ำ)

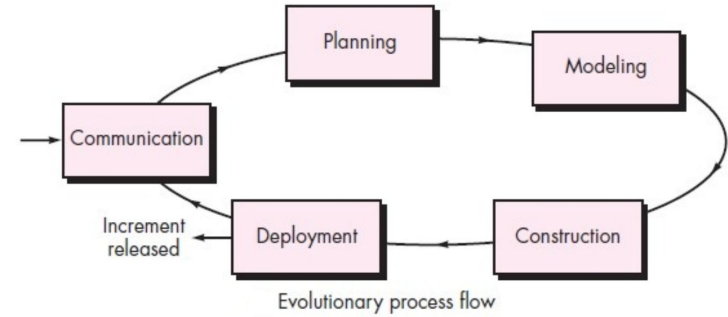
- An iterative process flow repeats one or more of the activities before proceeding to the next.



4 Types of Process Flow

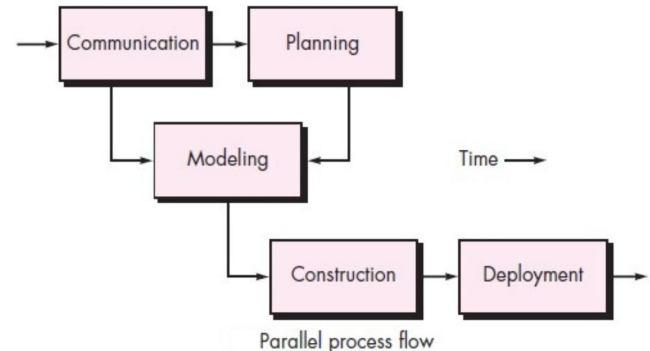
3. Evolutionary Process Flow (การไหลของกระบวนการแบบมีวิวัฒนาการ)

- An evolutionary process flow executes the activities in a “circular” manner.



4. Parallel Process Flow (การไหลของกระบวนการแบบขนาน)

- A parallel process flow executes one or more activities in parallel with other activities.



แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ | Software Process Models

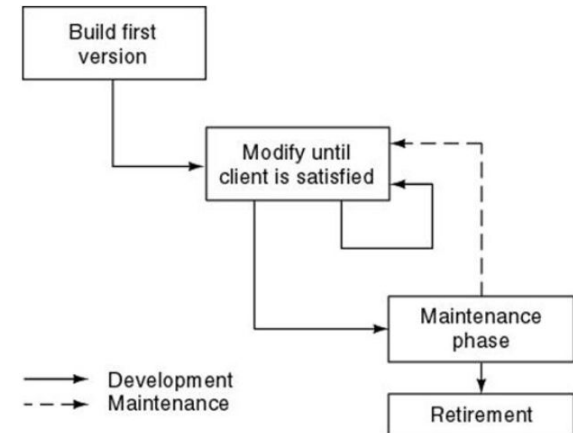
แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ คือ การจำลองภาพของกระบวนการ เพื่อให้เห็นถึงการจัดโครงสร้างลำดับขั้นตอนของกระบวนการในรูปแบบที่แตกต่างกันออกไป

แบบไม่เป็นระบบ (Unsystematic) * เหมาะกับโครงการขนาดเล็กทำใช้เอง	แบบเป็นระบบ (The Systematic Approach) *Traditional Models กระบวนการที่ใช้ต้องอธิบายรูปแบบของแบบจำลองได้ (*อาจมีมากกว่านี้)
<ol style="list-style-type: none"> 1. ไม่มีการ<u>วางแผน</u> 2. ไม่มีการ<u>จัดการ</u>ที่ดี 3. ไม่มีการ<u>จัดทำเอกสาร</u> 	<ol style="list-style-type: none"> 1. Waterfall Model / Document-Driven Model (แบบจำลองน้ำตก) 2. Incremental Model แบบจำลองส่วนเพิ่ม 3. Rapid Application Development: RAD Model แบบจำลองแบบเร่งรัด 4. Evolutionary Model/Prototyping แบบจำลองเชิงวิวัฒนาการหรือต้นแบบ 5. Spiral Model / Risk-Driven Model แบบจำลองเวียนกันหอย 6. Rational Unified Process แบบจำลอง

Software Process Models:

0. Built – and – Fix Model

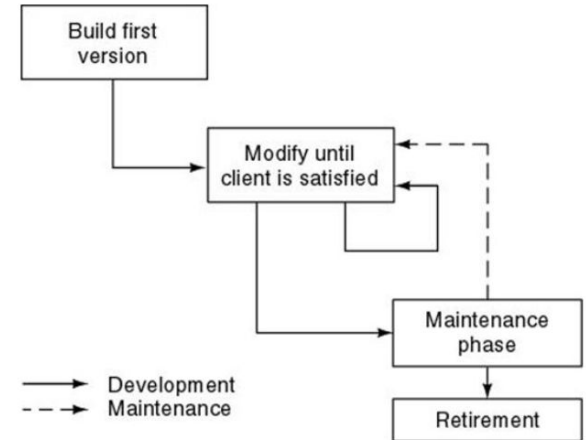
- เป็นโมเดลที่มีอายุเก่าแก่มากที่สุด
- แต่ไม่ค่อยได้รับความนิยมมากเท่าที่ควร
- เนื่องจาก
 - เป็นรูปแบบโมเดลที่เป็นลักษณะการเขียนโปรแกรม และแก้ไขปรับปรุงโปรแกรมไปเรื่อย ๆ
 - ลองผิดลองถูกไปจนกระทั่งคิดว่าพอใจหรือตรงกับความต้องการมากที่สุด
- เป็นโมเดลที่ไม่มีการวิเคราะห์ความต้องการ
- ไม่มีการกำหนดคุณลักษณะของซอฟต์แวร์
- ไม่มีการออกแบบ เหมาะกับงานที่ทำงานซ้ำ หลายๆ ครั้ง
- ซอฟต์แวร์มีความยาวไม่มากนัก ประมาณ 100- 200 บรรทัด



Software Process Models:

0. Built – and – Fix Model (Ad Hoc Model)

- The software is developed without any specification or design.
- An initial product is built, which is then repeatedly modified until it (software) satisfies the user.
- The software is developed and delivered to the user. The user checks whether the desired functions ‘are present.
 - If not, then the software is changed according to the needs by adding, modifying or deleting functions.
- This process goes on until the user feels that the software can be used productively.
- However, the lack of design requirements and repeated modifications result in loss of acceptability of software.
- Thus, software engineers are strongly discouraged from using this development approach.



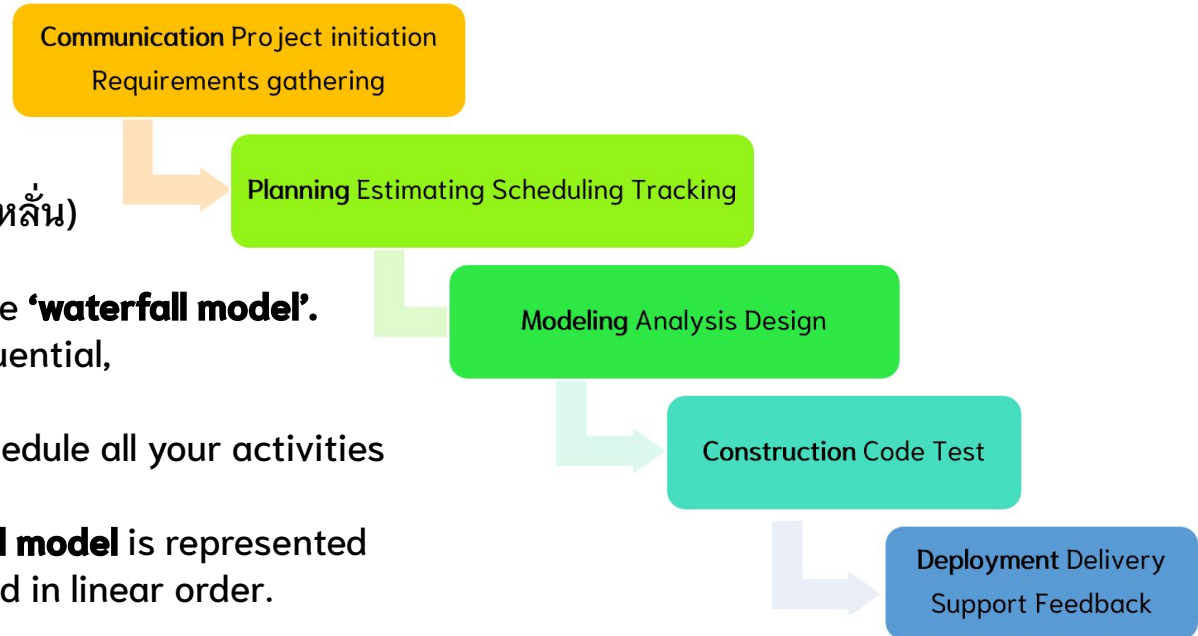
This model includes two phases.

1. **Build:** In this phase, the software code is developed and passed on to the next phase.
2. **Fix:** In this phase, the code developed in the build phase is made error free. Also, in addition to the corrections to the code, the code is modified according to the user's requirements.

Software Process Models:

1. Waterfall Model (แบบจำลองน้ำตก) / Document-Driven Model

- **The waterfall model** is an example of a plan-driven process—in principle, we must plan and schedule all of the process activities before starting work on them.



- **Phases** are cascaded (ถูกลดหลั่น) from one phase to another, that is while it is known as the **‘waterfall model’**.
- **The waterfall model** is a sequential, plan driven-process where you must plan and schedule all your activities before starting the project.
- **Each activity in the waterfall model** is represented as a separate phase arranged in linear order.

Software Process Models:

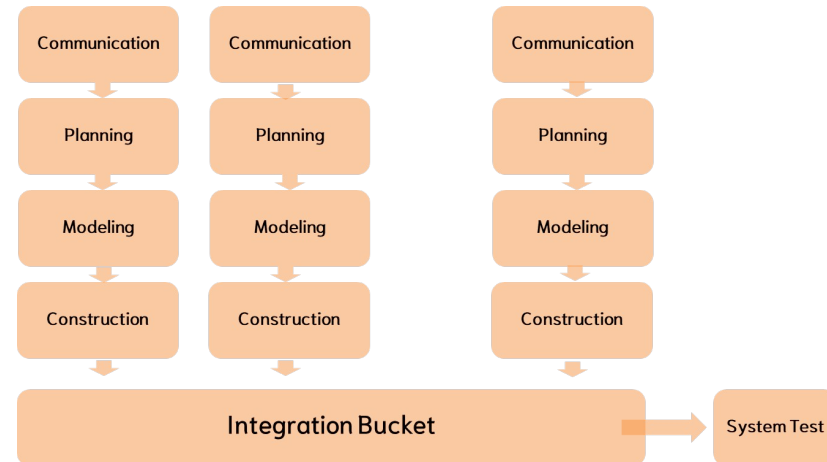
2. Incremental Model (แบบจำลองส่วนเพิ่ม/ก้าวหน้า)

The incremental model is a model of software development where the product is designed, implemented and tested incrementally (a little more is added each time) until the product is finished.

- It involves both development and maintenance.
- The product is defined as finished when it satisfies all of its requirements.
- This model combines the elements of the **waterfall model** with the iterative philosophy of prototyping.

Characteristics of Incremental Model

1. System is broken down into many mini development projects.
2. Partial systems are built to produce the final system.
3. First tackled highest priority requirements.
4. The requirement of a portion is frozen once the incremented portion is developed.

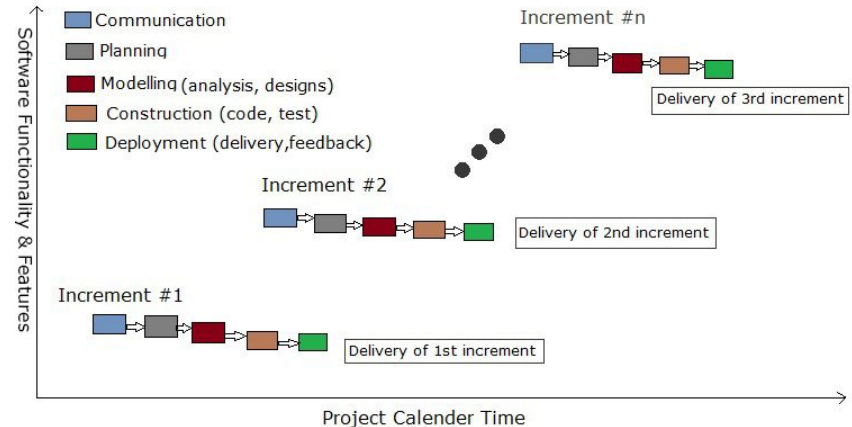


Software Process Models:

2. Incremental Model (แบบจำลองส่วนเพิ่ม/ก้าวหน้า)

These tasks are common to all the models

1. **Communication:** helps to understand the objective.
2. **Planning:**
required as many people (software teams) work on the same project but different function at same time.
3. **Modeling:**
involves business modeling, data modeling, and process modeling.
4. **Construction:**
this involves the reuse software components and automatic code.
5. **Deployment:**
integration of all the increments.



Software Process Models:

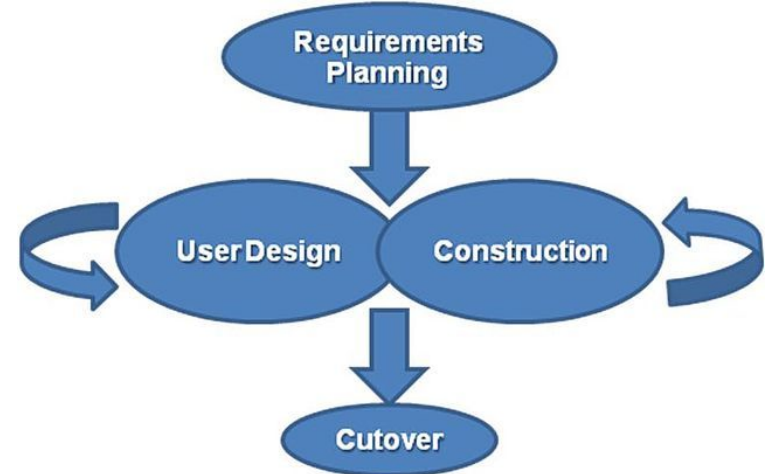
3. Rapid Application Development: RAD Model (แบบจำลองเร่งรัด)

Rapid application development (RAD), also called **rapid application building (RAB)**

- The name for James Martin's method of rapid development.
- In general, RAD approaches to software development put less emphasis on planning and more emphasis on an adaptive process.
- Prototypes are often used in addition to or sometimes even instead of design specifications.

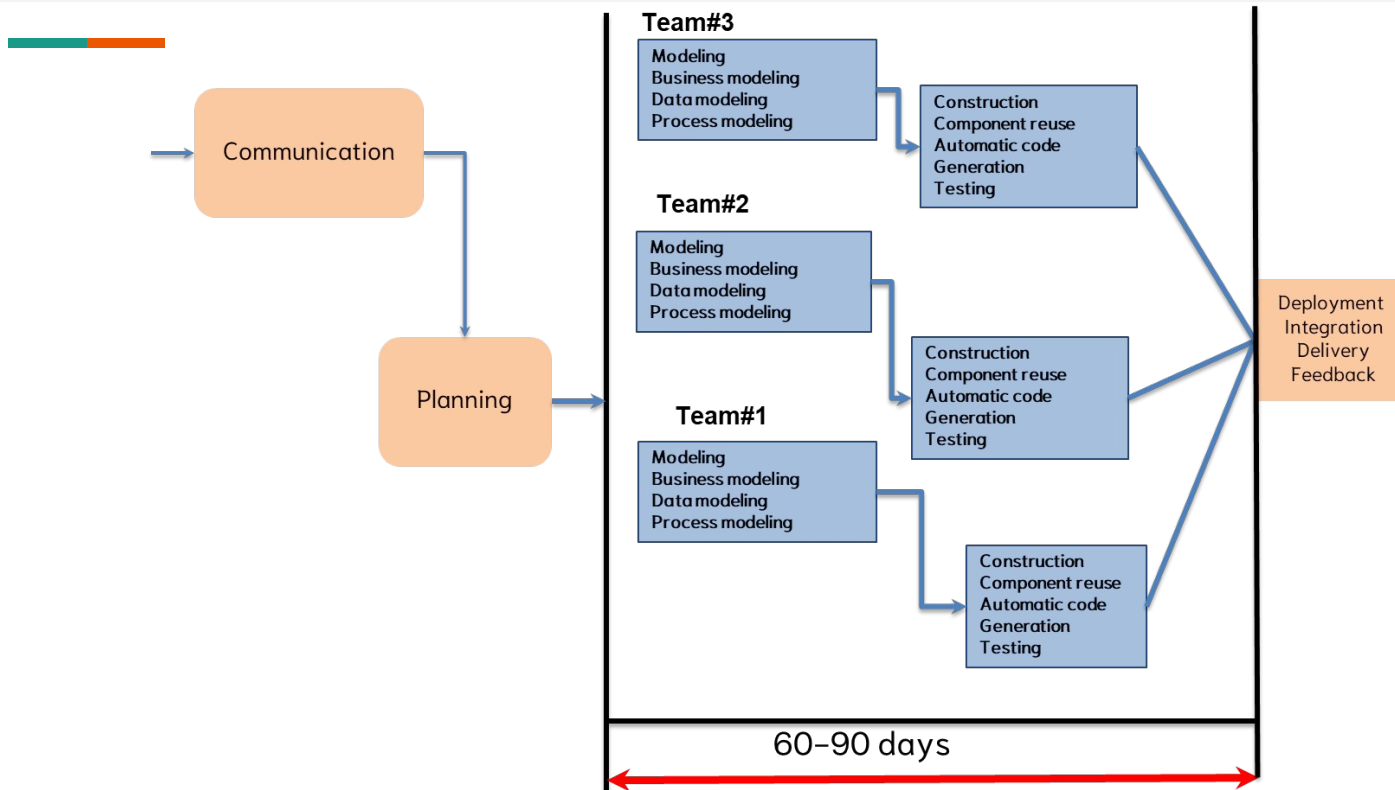
The James Martin approach to RAD divides the process into 4 distinct phases:

1. Requirements planning phase
2. User design phase
3. Construction phase
4. Cutover phase



Software Process Models:

3. Rapid Application Development: RAD Model (แบบจำลองเร่งรัด)

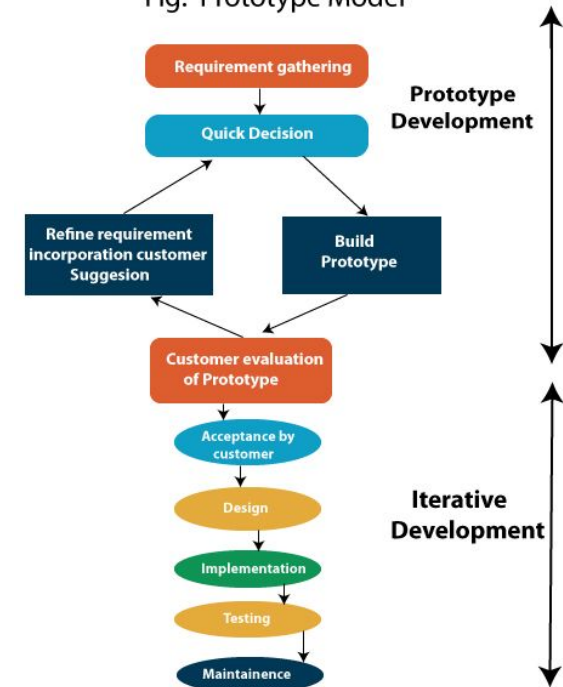


Software Process Models:

4. Evolutionary Model / Prototyping (แบบจำลองเชิงวิวัฒนาการ หรือ ต้นแบบ)

- **The prototype model** requires that before carrying out the development of actual software, a working prototype of the system should be built.
- **A prototype** is a toy implementation of the system.
- A prototype usually turns out to be:
 - A very crude version of the actual system,
 - Possible exhibiting limited functional capabilities,
 - Low reliability, and
 - Inefficient performance as compared to actual software.
- In many instances, the client only has a general view of what is expected from the software product.
 - In such a scenario where there is an absence of detailed information regarding the input to the system, the processing needs, and the output requirement, the prototyping model may be employed.

Fig: Prototype Model



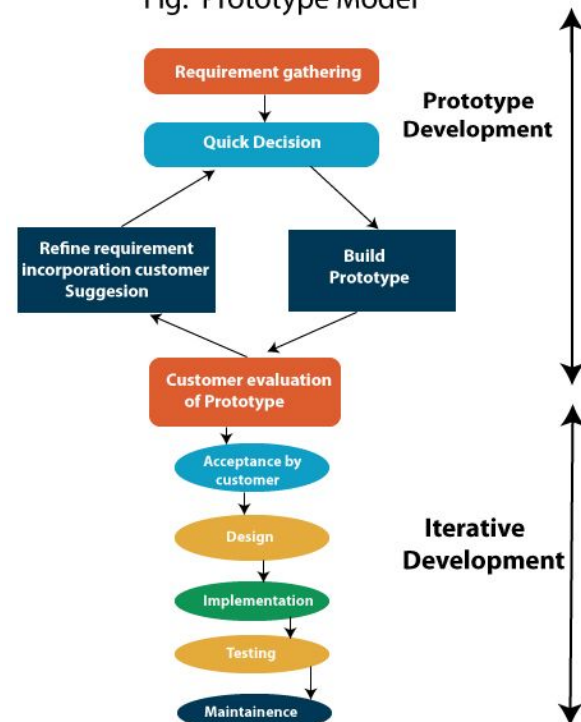
Software Process Models:

4. Evolutionary Model / Prototyping (แบบจำลองเชิงวิวัฒนาการ หรือ ต้นแบบ)

Steps of Prototype Model

1. Requirement Gathering and Analyst
2. Quick Decision
3. Build a Prototype
4. Assessment or User Evaluation
5. Prototype Refinement
6. Engineer Product

Fig: Prototype Model

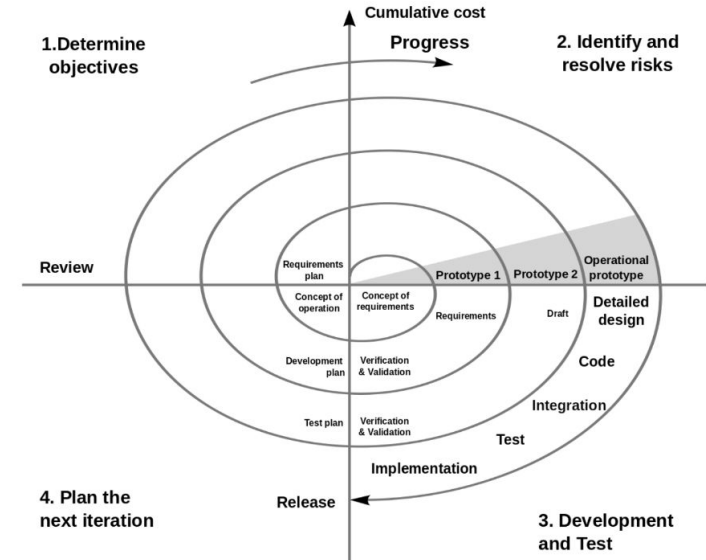


Software Process Models:

5. Spiral Model / Risk-Driven Model (แบบจำลองเวียนกันหอย)

Each cycle in the spiral is divided into four parts:

1. **Objective setting:**
Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.
2. **Risk Assessment and reduction:**
The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.
3. **Development and validation:**
The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.
4. **Planning:**
Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.



Spiral model (Boehm, 1988).

A number of misconceptions stem from oversimplifications in this widely circulated diagram (there are some errors in this diagram).

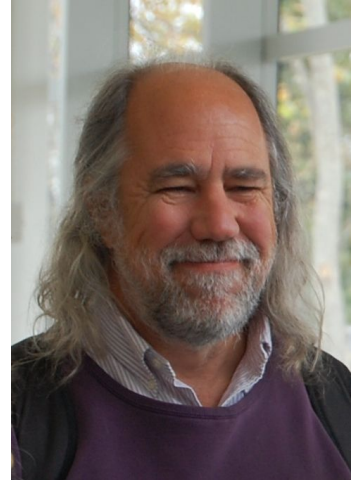
Software Process Models:

6. Rational Unified Process (RUP) (แบบกระบวนการรวม): Iterative + Evolution

- **Rational Unified Process (RUP)**

is a software development process for **object-oriented (OO) models**.
aka the **Unified Process Model**.

- Created by Rational corporation
Designed and documented using **UML (Unified Modeling Language)**.
- This process is included in IBM Rational Method Composer (RMC) product.
- IBM (International Business Machine Corporation) allows us to customize, design, and personalize the unified process.
- RUP is proposed by Ivar Jacobson, **Grady Bootch**, and James Rumbaugh.
- Some characteristics of RUP include
 - use-case driven,
 - Iterative (repetition of the process), and
Incremental (increase in value) by nature,
delivered online using web technology, can be customized or tailored in modular and electronic form, etc. RUP reduces unexpected development costs and prevents wastage of resources.



Software Process Models:

6. Rational Unified Process (RUP) (แบบกระบวนการรวม): Iterative + Evolution

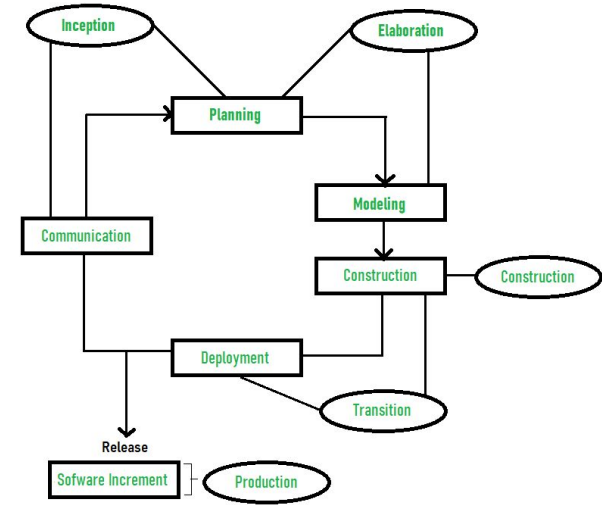
- Some characteristics of RUP include
 1. use-case driven,
 2. Iterative (repetition of the process), and
 3. Incremental (increase in value) by nature,
 4. delivered online using web technology,
 5. can be customized or tailored in modular and electronic form, etc.
- RUP reduces unexpected development costs and prevents wastage of resources.

Software Process Models:

6. Rational Unified Process (RUP) (แบบกระบวนการรวม): Iterative + Evolution

Phases of RUP: There is total of five phases of the life cycle of RUP:

1. **Inception (เริ่มต้นวิเคราะห์)** – Communication and planning are the main ones.
2. **Elaboration (ลงรายละเอียด)** – Planning and modeling are the main ones.
3. **Construction (สร้าง)** – The project is developed and completed.
4. **Transition (เปลี่ยนแปลง)** – The final project is released to the public.
5. **Production (ขึ้นผลิตภัณฑ์)** – The final phase of the model.



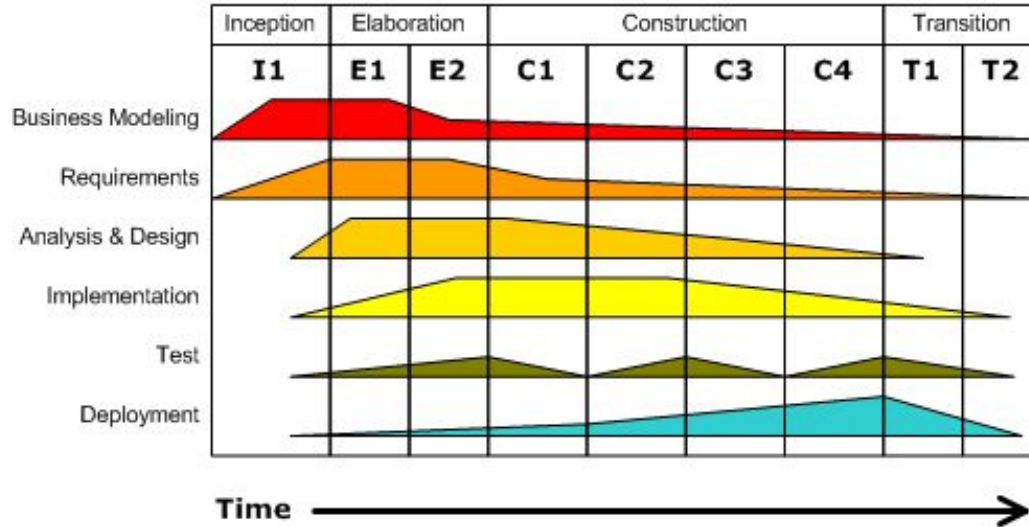
Software Process Models:

6. Rational Unified Process (RUP) (แบบกระบวนการรวม): Iterative + Evolution

RUP phases and disciplines.

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.





SF220 Introduction to Software Engineering

[SF220-W02-GrpWork] กระบวนการพัฒนาซอฟต์แวร์ Software Development Life Cycle (SDLC)

[SF220-W02-GrpWork]

3. งานกลุ่ม (Group Work): แบบฝึกปฏิบัติในชั้นเรียน (60 นาที; 5 คะแนน)

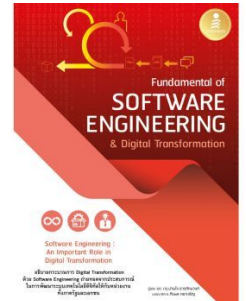
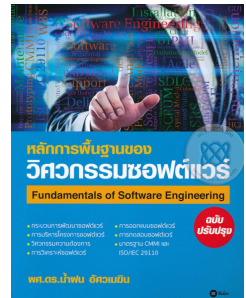
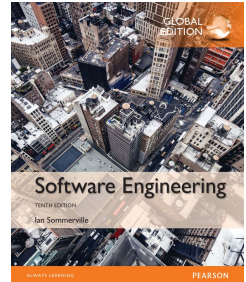
3.1 ให้นักศึกษาทำการค้นคว้าและศึกษาโมเดลการพัฒนาซอฟต์แวร์ (Software Development Models) ตามหัวข้อที่นักศึกษาได้ทำการจับฉลากและ ทำการนำเสนอหน้าชั้นเรียน โดยต้องมีเนื้อหาครอบคลุมดังนี้

- ชื่อหัวข้อโมเดลที่ได้ และ รายชื่อสมาชิกในกลุ่มของนักศึกษา
- คำอธิบายโมเดลการพัฒนาซอฟต์แวร์
- ภาพประกอบแบบจำลองของ โมเดลการพัฒนาซอฟต์แวร์
- ข้อดี-ข้อเสียของโมเดลการพัฒนาซอฟต์แวร์
- แหล่งอ้างอิงข้อมูลที่นักศึกษาได้ทำการค้นคว้า
- นำเสนอกลุ่มละ 5 นาที (ตามช่วงเวลาที่กำหนด)
กรณีนำเสนอเกินเวลาที่กำหนด ตัดคะแนน 80% ของคะแนนที่ได้



เอกสารประกอบการเรียน | Book/eBook References

- Sommerville, I. (2016)
Software Engineering. 10th Edition, Pearson Education Limited, Boston.
- ผศ.ดร. น้ำฝน อัครเมสิน (2560)
หลักการพื้นฐานของวิศวกรรมซอฟต์แวร์
(Fundamentals of Software Engineering).
ซีเอ็ดยูเคชั่น, บมจ.
- รศ. ดร.ปานใจ ธารทัศนวงศ์. (2565)
Fundamental of Software Engineering & Digital Transformation,
Infopress.





**EDUCATING
THE MIND**

WITHOUT

**EDUCATING
THE HEART**

IS NO EDUCATION AT ALL.

ARISTOTLE

Reference:
<https://www.redbubble.com/i/art-board-print/Educating-the-mind-without-educating-the-heart-is-no-education-at-all-by-proofreading/36956096.JUXJO>

Thank you
for
your attention.

“การให้ความรู้แก่สมองโดยที่หัวใจปราศจากการเรียนรู้ มีค่าเท่ากับการไม่ได้เรียนรู้อะไรเลย。”

-อริสโตเติล-

<https://www.goodreads.com/author/quotes/2192.Aristotle>