

# Software Engineering

# 1. แนะนำรายวิชา Introduction

รหัสวิชา 03376806

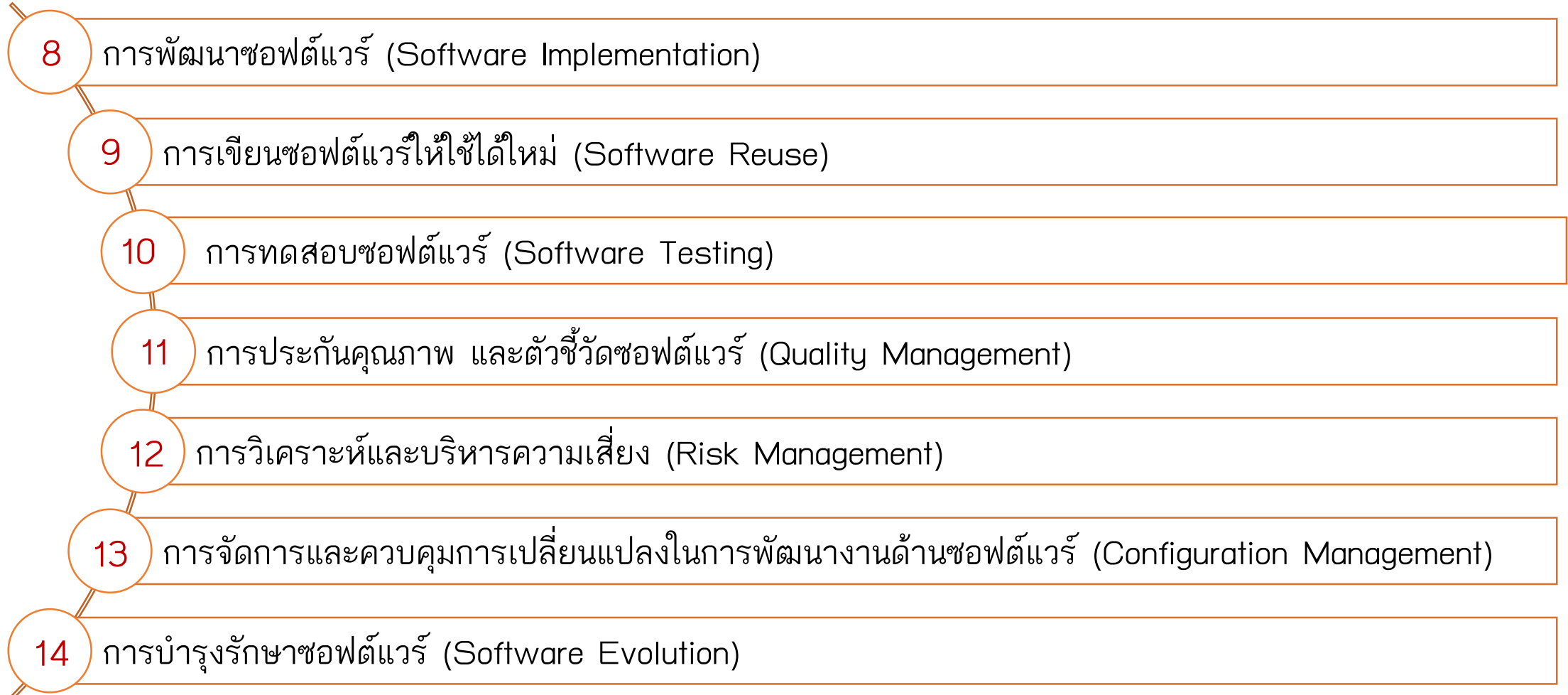
ชื่อวิชา วิศวกรรมซอฟต์แวร์ (Software Engineering)

ผู้สอน ผศ.โกศล ตราชู

# เรื่องที่จะศึกษา

- 1 วิศวกรรมซอฟต์แวร์เบื้องต้น
- 2 กระบวนการพัฒนาซอฟต์แวร์ (Software Process)
- 3 กระบวนการพัฒนาซอฟต์แวร์แบบอไจล์ (Agile Software Development)
- 4 การบริหารโครงการซอฟต์แวร์ (Project Management)
- 5 วิศวกรรมความต้องการ (Requirement Engineering)
- 6 แบบจำลองระบบ (System Modeling)
- 7 การออกแบบซอฟต์แวร์ (Software Design)

# เรื่องที่จะศึกษา

- 
- 8 การพัฒนาซอฟต์แวร์ (Software Implementation)
  - 9 การเขียนซอฟต์แวร์ให้ใช้ได้ใหม่ (Software Reuse)
  - 10 การทดสอบซอฟต์แวร์ (Software Testing)
  - 11 การประกันคุณภาพ และตัวชี้วัดซอฟต์แวร์ (Quality Management)
  - 12 การวิเคราะห์และบริหารความเสี่ยง (Risk Management)
  - 13 การจัดการและควบคุมการเปลี่ยนแปลงในการพัฒนางานด้านซอฟต์แวร์ (Configuration Management)
  - 14 การบำรุงรักษาซอฟต์แวร์ (Software Evolution)

# กิจกรรมการเรียนการสอน

- การมีส่วนร่วมในการเรียนการสอน
  - กิจกรรมต่าง ๆ
- การนำเสนองานหน้าชั้นเรียน
- การแบ่งกลุ่มย่อย ทำโครงงานวิศวกรรมซอฟต์แวร์
- การบ้าน – แบบฝึกหัด
- การสอบกลางภาคและปลายภาคเรียน

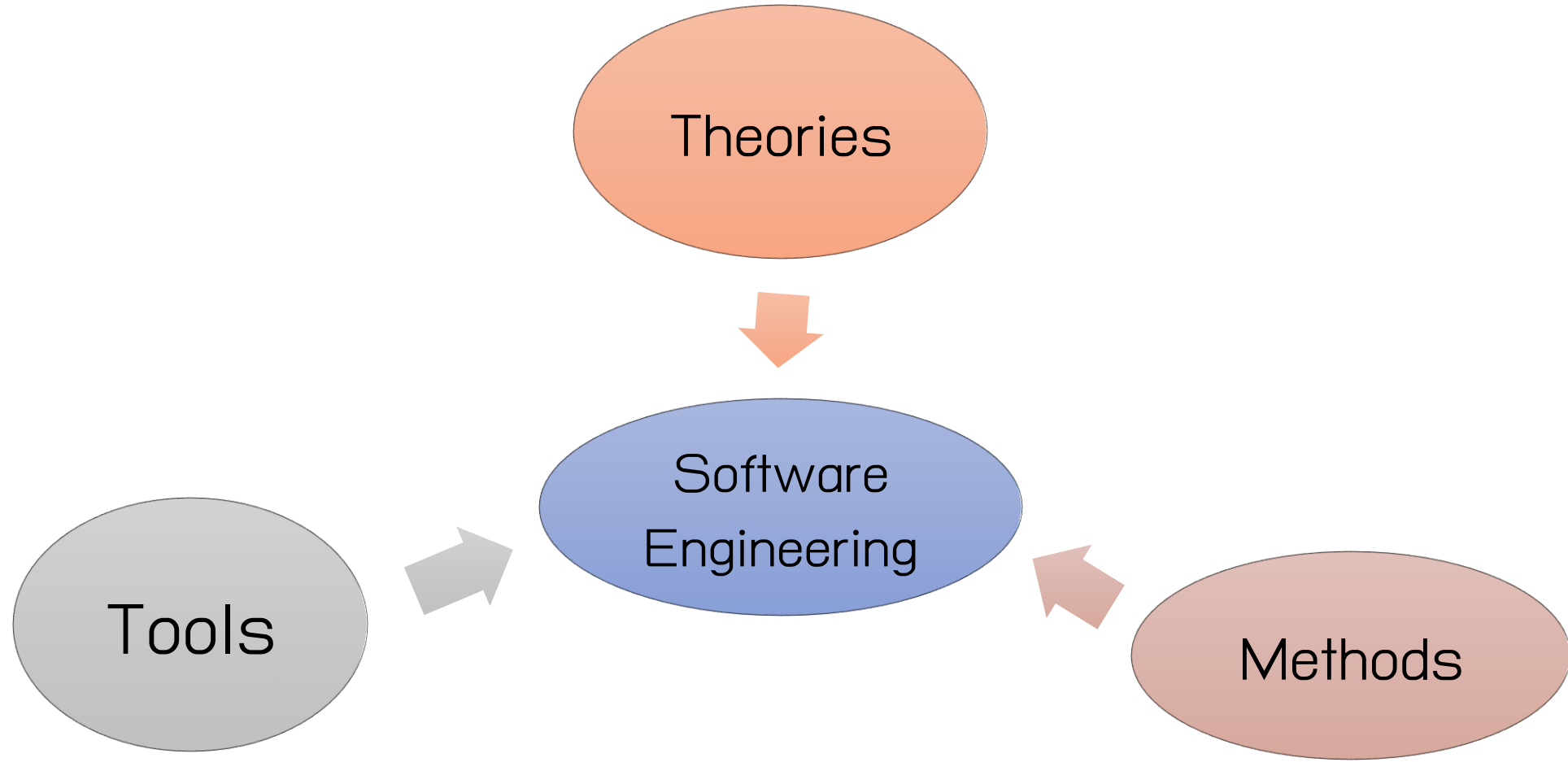
# การวัดและประเมินผล

รายการ	คะแนน ร้อยละ
แบบฝึกหัด, การบ้าน	20
งานย่อย รายบุคคล	20
โครงการงานย่อย รายกลุ่ม	20
สอบกลางภาค	20
สอบปลายภาค	20
รวม	100

# ตำรา และ เอกสารอ้างอิง

- Ian Sommerville, “Software Engineering”, 10th edition, Addison Wesley, 2015
- Roger Pressman,” Software Engineering: A practitioner’s Approach”, 7th edition, McGraw Hill, 2009
- ตำราภาษาไทย และเอกสารที่เกี่ยวข้อง
- <https://iansommerville.com/software-engineering-book/>

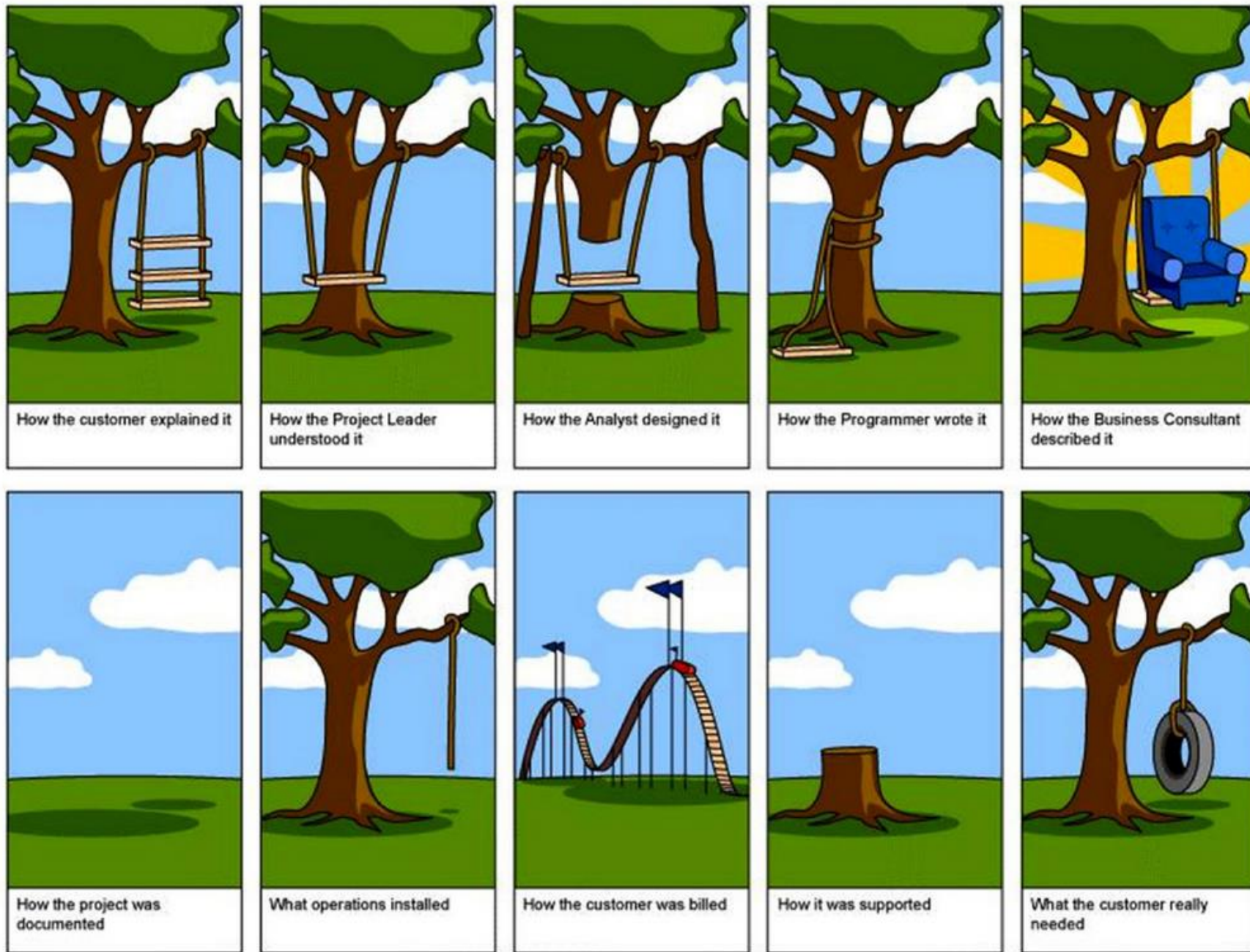
# วิศวกรรมซอฟต์แวร์





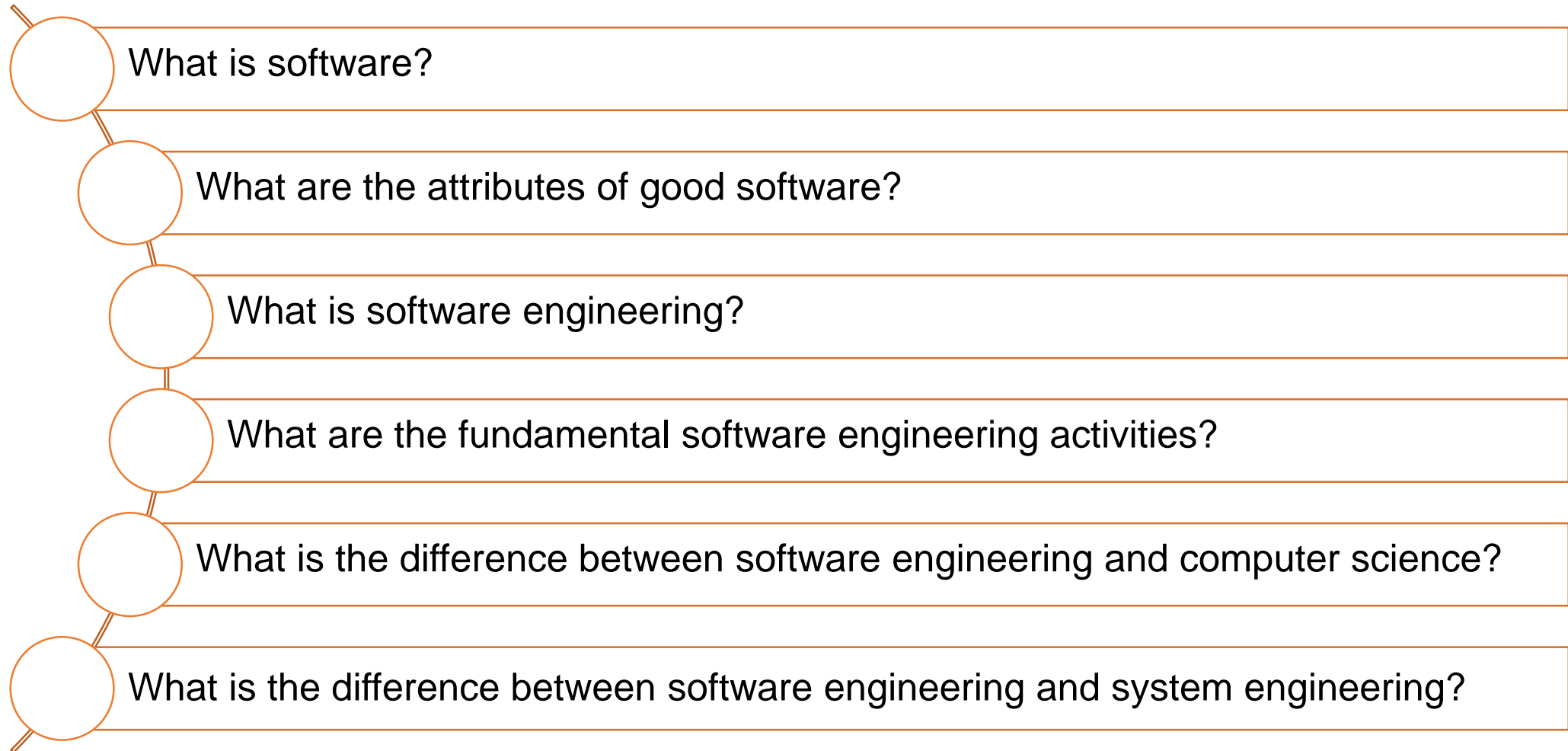
# Software engineering Fact!!!

- ระบบเศรษฐกิจของประเทศที่พัฒนาแล้ว ล้วนแต่พึ่งพา Software
- ทุกระบบในโลก ต่างก็มี software controlled เพิ่มมากขึ้น มากขึ้นเรื่อย ๆ
- ราคาของ software มักจะเป็นตัวกำหนดราคาของระบบคอมพิวเตอร์
- ราคาของการ maintain มักจะสูงกว่าราคาในการ develop
  - ในระบบที่ใช้งานเป็นระยะเวลายาวนาน ค่าบำรุงรักษาจะสูงกว่าค่าสร้างซอฟต์แวร์
- Software engineering เน้นการพัฒนาซอฟต์แวร์ที่คุ้มค่า (cost-effective)

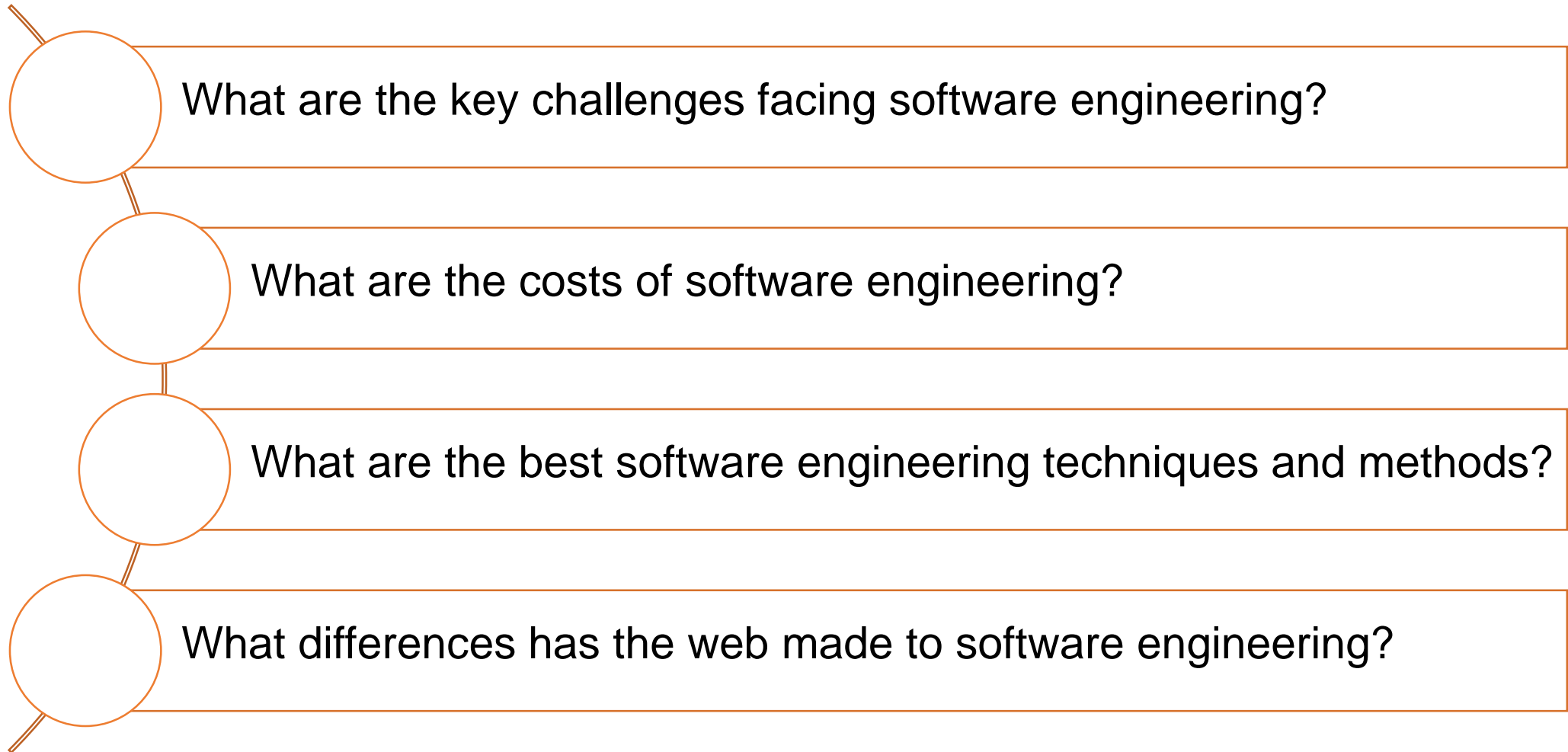


# Professional software development

# FAQ เกี่ยวกับ software engineering

- 
- What is software?
  - What are the attributes of good software?
  - What is software engineering?
  - What are the fundamental software engineering activities?
  - What is the difference between software engineering and computer science?
  - What is the difference between software engineering and system engineering?

# FAQ เกี่ยวกับ software engineering



# What is software?

Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

Software คือ โปรแกรมคอมพิวเตอร์รวมถึงเอกสารที่เกี่ยวข้อง โดยซอฟต์แวร์อาจจะถูกสร้างขึ้นตามความต้องการของผู้ใช้เฉพาะราย หรือสร้างขึ้นเพื่อวางจำหน่ายโดยทั่วไป (เช่น โปรแกรมสำเร็จรูปประมวลผลคำ ระบบปฏิบัติการ)

# What are the attributes of good software?

A: Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.

ซอฟต์แวร์ที่ดี ควรประกอบด้วยฟังก์ชันการทำงานที่ผู้ใช้ต้องการ มีประสิทธิภาพในการทำงาน สามารถบำรุงรักษาง่าย เชื่อถือได้และมีประโยชน์

# What is software engineering?

Software engineering is an engineering discipline that is concerned with all aspects of software production.

วิศวกรรมซอฟต์แวร์เป็นสาขาวิศวกรรมที่เกี่ยวข้องกับ**ทุกมิติ**ของการผลิตซอฟต์แวร์



# What are the fundamental software engineering activities?

Software specification, software development, software validation and software evolution.

การออกข้อกำหนดของซอฟต์แวร์ การพัฒนาซอฟต์แวร์ การตรวจสอบซอฟต์แวร์ และการพัฒนาซอฟต์แวร์

# What is the difference between software engineering and computer science?

Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.

วิทยาการคอมพิวเตอร์ มุ่งเน้นไปที่ทฤษฎีและพื้นฐาน

วิศวกรรมซอฟต์แวร์ เกี่ยวข้องกับการปฏิบัติในการพัฒนาและนำเสนอซอฟต์แวร์ที่เป็นประโยชน์

# What is the difference between software engineering and system engineering?

System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering. Software engineering is part of this more general process.

วิศวกรรมระบบเกี่ยวข้องกับทุกด้านของการพัฒนาระบบคอมพิวเตอร์ (ซึ่งรวมถึงฮาร์ดแวร์ ซอฟต์แวร์ และกระบวนการทางวิศวกรรม) ในขณะที่วิศวกรรมซอฟต์แวร์เป็นเพียงส่วนหนึ่งของกระบวนการทั่วไปนี้

# What are the key challenges facing software engineering?

Coping with increasing diversity, demands for reduced delivery times and developing trustworthy software.

เผชิญกับความหลากหลายที่เพิ่มขึ้น    ระยะเวลาในการจัดส่งที่ลดลง และการพัฒนาซอฟต์แวร์ที่น่าเชื่อถือ

# What are the costs of software engineering?

Roughly 60% of software costs are development costs, 40% are testing costs.  
For custom software, evolution costs often exceed development costs.

ค่าใช้จ่ายต้นทุนการพัฒนา ประมาณ 60%

ค่าใช้จ่ายในการทดสอบ ประมาณ 40%

สำหรับซอฟต์แวร์ที่กำหนดเอง ค่าใช้จ่ายในการบำรุงรักษามักจะสูงกว่าต้นทุนการพัฒนา

# What are the best software engineering techniques and methods?

While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.

ในขณะที่โครงการซอฟต์แวร์ทั้งหมดต้องได้รับการจัดการและพัฒนาอย่างมืออาชีพ เทคนิคต่าง ๆ จะเหมาะสมกับระบบต่าง ๆ กันไป ตัวอย่างเช่นเกมควรได้รับการพัฒนาโดยใช้ชุดของต้นแบบ ในขณะที่ระบบควบคุมที่เน้นความปลอดภัย ต้องมีข้อกำหนดที่ครบถ้วนและสามารถวิเคราะห์ได้ เราไม่สามารถบอกได้ว่าวิธีหนึ่งดีกว่าอีกวิธีหนึ่ง

# What differences has the web made to software engineering?

The web has led to the availability of software services and the possibility of developing highly distributed service-based systems. Web-based systems development has led to important advances in programming languages and software reuse.

นับตั้งแต่มีเว็บ ก็ทำให้ผู้คนเข้าถึงบริการด้านซอฟต์แวร์มากขึ้น และเพิ่มโอกาสในการพัฒนาระบบ distributed service-based

การพัฒนาระบบบนเว็บได้นำไปสู่ความก้าวหน้าที่สำคัญในการเขียนโปรแกรมภาษาและการนำซอฟต์แวร์มาใช้ใหม่

# Software products

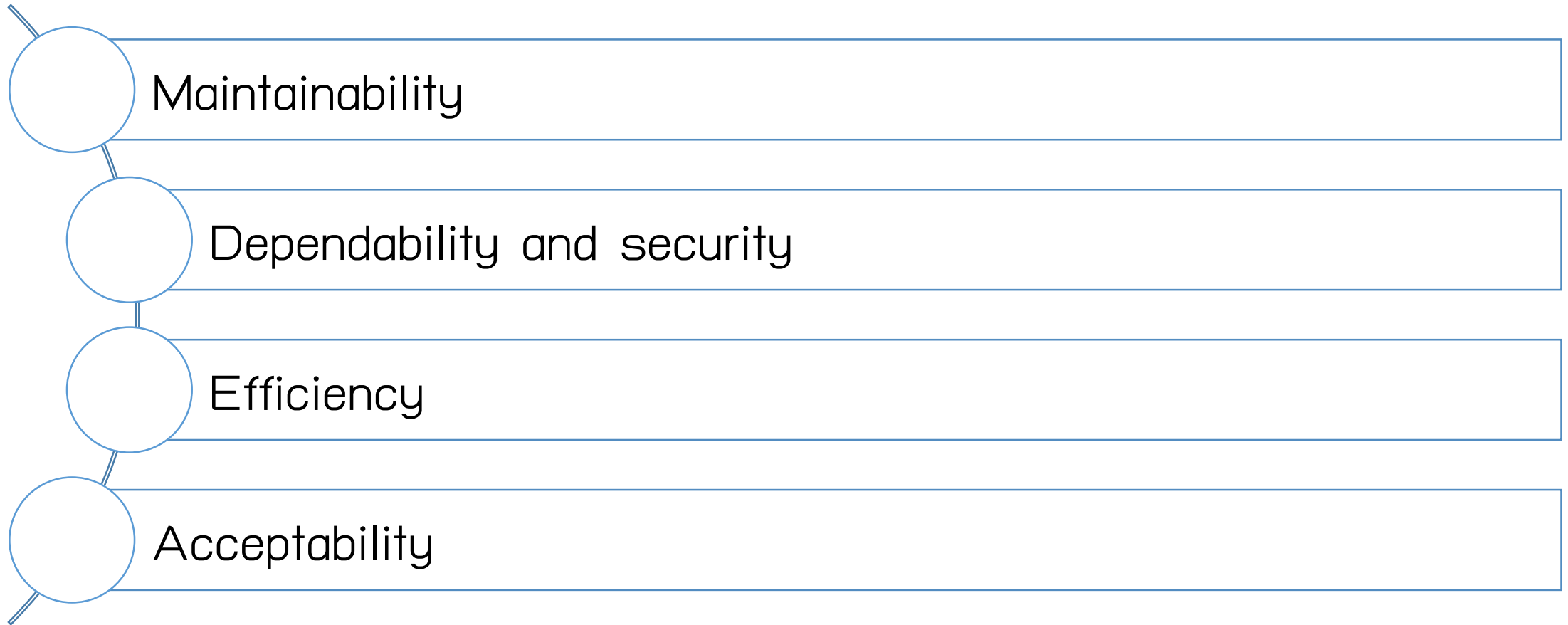
- Generic products
  - **Stand-alone systems** that are marketed and **sold to any customer** who wishes to buy them.
  - Examples – PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.
- Customized products
  - **Software that is commissioned by a specific customer to meet their own needs.**
  - Examples – embedded control systems, air traffic control software, traffic monitoring systems.



# Product specification

- Generic products
  - The **specification** of what the software should do is **owned by the software developer** and decisions on software change are made by the developer.
- Customized products
  - The **specification** of what the software should do is **owned by the customer** for the software and they make decisions on software changes that are required.

# Essential attributes of good software



## Product characteristic Description

### Maintainability

Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.

ซอฟต์แวร์ควรได้รับการเขียนในลักษณะเพื่อให้สามารถพัฒนาเพื่อตอบสนองความต้องการที่เปลี่ยนแปลงไปของลูกค้า นี่เป็นคุณลักษณะที่สำคัญเนื่องจากการเปลี่ยนแปลงซอฟต์แวร์เป็นข้อกำหนดที่หลีกเลี่ยงไม่ได้ ในสภาพแวดล้อมทางธุรกิจที่เปลี่ยนแปลงไป

### Dependability and security

Software dependability includes a range of characteristics including reliability, security and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.

ความน่าเชื่อถือของซอฟต์แวร์ประกอบด้วยลักษณะเฉพาะหลายอย่าง นับตั้งแต่ ความน่าเชื่อถือ ความมั่นคง และความปลอดภัย ซอฟต์แวร์ที่น่าเชื่อถือไม่ควรทำให้เกิดความเสียหายไม่ว่าจะเป็นทางกายภาพหรือทางเศรษฐกิจ ถึงแม้ระบบจะล้มเหลวก็ตาม ผู้ใช้ที่เป็นอันตรายไม่ควรสามารถเข้าถึงหรือทำลายระบบได้

Product characteristic	Description
Efficiency	<p>Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.</p> <p>ซอฟต์แวร์ไม่ควรใช้ทรัพยากรระบบอย่างสิ้นเปลือง ไม่ว่าจะเป็นหน่วยความจำหรือโปรเซสเซอร์ ประสิทธิภาพรวมถึงเวลาในการตอบสนอง เวลาการประมวลผล การใช้หน่วยความจำ ฯลฯ</p>
Acceptability	<p>Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.</p> <p>ซอฟต์แวร์ต้องเป็นที่ยอมรับจากผู้ใช้ประเภทต่าง ๆ ที่ออกแบบไว้ ซึ่งหมายความว่าต้องเข้าใจง่าย ใช้งานได้ และสามารถทำงานร่วมกับระบบอื่น ๆ ที่ใช้ได้</p>

# Software engineering

- วิศวกรรมซอฟต์แวร์เป็นสาขาวิศวกรรมที่เกี่ยวข้องกับทุกแง่มุมในการผลิตซอฟต์แวร์ตั้งแต่ขั้นเริ่มต้นของข้อกำหนดระบบจนถึงการบำรุงรักษาระบบหลังจากที่ได้เริ่มใช้งานแล้ว
- แง่มุมทางด้านวิศวกรรม
  - การใช้ทฤษฎีและวิธีการที่เหมาะสมในการแก้ปัญหาที่เกิดขึ้นกับข้อจำกัดภายในองค์กรและการเงิน
- แง่มุมในการผลิตซอฟต์แวร์
  - ไม่ใช่เพียงแค่กระบวนการทางเทคนิคในการพัฒนาเท่านั้น ยังรวมถึงการบริหารโครงการ การพัฒนาเครื่องมือ และวิธีการอื่น ๆ เพื่อสนับสนุนการผลิตซอฟต์แวร์

# Importance of software engineering

- บุคคลและสังคม ต้องพึ่งพาระบบซอฟต์แวร์ขั้นสูงมากขึ้นเรื่อย ๆ เราจำเป็นต้องสามารถผลิตระบบ (หมายความว่ารวมถึงซอฟต์แวร์ที่ใช้ควบคุม) ที่น่าเชื่อถือและวางใจได้ได้อย่างประหยัดและรวดเร็ว
- การใช้วิศวกรรมซอฟต์แวร์ จะช่วยให้เกิดความคุ้มค่าในระยะยาว
  - มันคุ้มค่าต่อทุกฝ่ายที่เกี่ยวข้อง หากเราใช้วิธีการด้านวิศวกรรมซอฟต์แวร์สำหรับผลิตซอฟต์แวร์ แทนการเขียนโปรแกรมแบบเก่าๆ
  - ค่าใช้จ่ายส่วนใหญ่ที่เกิดขึ้น มักจะเป็นค่าใช้จ่ายในการเปลี่ยนแปลงซอฟต์แวร์หลังจากที่ใช้งานแล้ว
  - หากขั้นตอนการพัฒนาเป็นไปตามหลักวิศวกรรม จะช่วยให้การเปลี่ยนแปลงซอฟต์แวร์หลังจากที่ใช้งานแล้ว มีความยุ่งยากลดลง

# ความหลากหลายทาง Software engineering

- ระบบซอฟต์แวร์มีหลายประเภทของ และไม่มีเทคนิคซอฟต์แวร์อเนกประสงค์หรือเทคนิคสำเร็จรูปที่สามารถใช้ได้กับการสร้างซอฟต์แวร์ได้ทุก ๆ รูปแบบ
- วิธีการและเครื่องมือด้านวิศวกรรมซอฟต์แวร์ ขึ้นอยู่กับประเภทของแอปพลิเคชันที่พัฒนาขึ้น
  - แตกต่างกันไปตามความต้องการของลูกค้า และพื้นฐานของทีมพัฒนา

# ประเภทของโปรแกรมประยุกต์ (application types)

- Stand-alone applications

- เป็นระบบ application ที่รันบนเครื่องคอมพิวเตอร์ (เช่นพีซี) ประกอบด้วยฟังก์ชันการทำงานที่จำเป็นทั้งหมดและไม่จำเป็นต้องเชื่อมต่อกับเครือข่าย

- Interactive transaction-based applications

- Application ที่รันบนคอมพิวเตอร์ระยะไกลและเข้าถึงได้โดยผู้ใช้จากเครื่องพีซีหรือ terminal ของตนเอง ซึ่งรวมถึง web application

- Embedded control systems

- เป็นซอฟต์แวร์ที่ควบคุมและจัดการอุปกรณ์ฮาร์ดแวร์แบบฝังตัว ในโลกนี้มีระบบฝังตัวมากกว่าระบบอื่น ๆ

- Batch processing systems

- เป็นระบบที่ออกแบบมาเพื่อประมวลผลข้อมูลเป็นกลุ่มใหญ่ สามารถนำข้อมูลจำนวนมากมาประมวลผลเพื่อสร้างผลลัพธ์ที่ต้องการ



# ประเภทของโปรแกรมประยุกต์ (application types)

- Entertainment systems
  - เป็นระบบที่เน้นใช้งานส่วนบุคคล และถูกออกแบบมาเพื่อความบันเทิงแก่ผู้ใช้เป็นหลัก
- Systems for modeling and simulation
  - เป็นระบบที่พัฒนาโดยนักวิทยาศาสตร์และวิศวกรในการจำลองกระบวนการหรือสถานการณ์ทางกายภาพซึ่งรวมถึงแบบจำลองของวัตถุหลายสิ่งที่แตกต่างกันแต่ทำงานประสานกัน
- Data collection systems
  - เป็นระบบที่รวบรวมข้อมูลจากสภาพแวดล้อมโดยใช้ชุดเซ็นเซอร์และส่งข้อมูลเหล่านั้นไปยังระบบอื่น ๆ เพื่อการประมวลผล
- Systems of systems
  - เป็นระบบที่ประกอบด้วยระบบซอฟต์แวร์อื่น ๆ จำนวนมาก

# Software engineering fundamentals

ถึงแม้ประเภทของซอฟต์แวร์จะมีหลากหลาย แต่หลักการพื้นฐานบางอย่าง ก็สามารถใช้ได้กับการสร้างซอฟต์แวร์ทุกประเภทโดยไม่คำนึงถึงเทคนิคการพัฒนาที่ใช้ เช่น

- ระบบควรได้รับการพัฒนาโดยมีความเข้าใจในกระบวนการพัฒนาอย่างถ่องแท้ ถึงแม้ว่าจะมีกระบวนการที่แตกต่างกันไปสำหรับซอฟต์แวร์ประเภทต่างๆ
- ความเชื่อถือได้และประสิทธิภาพเป็นสิ่งสำคัญสำหรับระบบทุกประเภท
- ทำความเข้าใจและการจัดการข้อกำหนดและความต้องการของซอฟต์แวร์ (สิ่งที่ซอฟต์แวร์ควรทำ) ให้รอบคอบ เนื่องจากมีความสำคัญเป็นลำดับต้น ๆ
- ถ้าเป็นไปได้ (และเหมาะสม) ควรใช้ซอฟต์แวร์ที่พัฒนาแล้วแทนที่จะเขียนซอฟต์แวร์ใหม่

# Internet software engineering

- ปัจจุบันนี้เว็บเป็นแพลตฟอร์มสำหรับการรัน application ที่ได้รับความนิยมมากขึ้นเรื่อย ๆ
  - มีการพัฒนาระบบบนเว็บมากกว่าระบบ stand alone
  - บริการเว็บ (web services) ช่วยให้สามารถเข้าถึง application ได้ผ่านทางเว็บ
  - บริการเว็บช่วยให้เข้าถึงข้อมูลได้หลากหลาย ทั้งที่สดใหม่หรือเก็บถาวร
  - Cloud computing เป็นแนวทางในการให้บริการคอมพิวเตอร์ซึ่ง application ทำงานจากระยะไกลบน 'cloud'
  - ผู้ใช้ไม่ได้ซื้อซอฟต์แวร์ (หรือแม้แต่ฮาร์ดแวร์) แต่เป็นระบบที่จ่ายตามการใช้งาน

# Web software engineering

- Software reuse
  - การใช้ซอฟต์แวร์ซ้ำเป็นวิธีการที่สำคัญสำหรับการสร้างระบบบนเว็บ เมื่อจะสร้างระบบเหล่านี้เราควรนึกถึงการใช้งานส่วนประกอบและระบบซอฟต์แวร์ที่มีอยู่เดิมเป็นอันดับแรก
- Incremental and agile development
  - ระบบเว็บได้รับการพัฒนาและแจกจ่ายออกมาเรื่อย ๆ ตามความต้องการที่เปลี่ยนไปทุกวัน
  - ปัจจุบันนี้ได้รับการยอมรับโดยทั่วไปว่า เราไม่สามารถระบุข้อกำหนดทั้งหมดสำหรับระบบใดๆ ได้ล่วงหน้า ต้องทำไปแก้ไขไปพัฒนาเพิ่มเติมไปเรื่อย ๆ
  - เทคนิคการพัฒนาแบบ Incremental และ agile จะช่วยให้พัฒนาซอฟต์แวร์ได้ทันความต้องการของตลาด
- ระบบที่มุ่งเน้นบริการ (Service-oriented systems)
  - ซอฟต์แวร์สามารถสร้างโดยใช้วิศวกรรมซอฟต์แวร์เชิงบริการ ซึ่งส่วนประกอบซอฟต์แวร์เป็นบริการเว็บแบบ stand-alone
- Rich interfaces
  - เมื่อเทคโนโลยีการพัฒนาอินเทอร์เน็ตเฟสเช่น AJAX และ HTML5 เกิดขึ้น จะสนับสนุนการสร้าง interface ที่หลากหลายภายในเว็บเบราว์เซอร์

# Software engineering ethics

- วิศวกรรมซอฟต์แวร์เกี่ยวข้องกับ ความรับผิดชอบ มากกว่าเพียงแค่การใช้ ทักษะทางเทคนิค
- วิศวกรซอฟต์แวร์จะต้องปฏิบัติตนอย่าง ซื่อสัตย์และมีจริยธรรม หากต้องได้รับการ ยอมรับว่าเป็นมืออาชีพ
- พฤติกรรมทางจริยธรรมเป็น มากกว่าแค่การปฏิบัติตามกฎหมาย แต่เกี่ยวข้องกับ การปฏิบัติตามหลักการที่ถูกต้องตามหลักศีลธรรม

# Issues of professional responsibility

- Confidentiality (ความลับ)
  - วิศวกรควรให้ความสำคัญกับการรักษาความลับของนายจ้างหรือลูกค้า โดยไม่คำนึงถึงว่าได้มีการลงนามในข้อตกลงการรักษาความลับอย่างเป็นทางการหรือไม่
- Competence (ความสามารถ)
  - วิศวกรไม่ควรบิดเบือนความสามารถของตน ไม่ควรที่จะรับงานที่ตนไม่มีความสามารถ
- Intellectual property rights (สิทธิในทรัพย์สินทางปัญญา)
  - วิศวกรควรตระหนักถึงกฎหมายท้องถิ่น ที่เกี่ยวกับการใช้ทรัพย์สินทางปัญญา เช่น สิทธิบัตร ลิขสิทธิ์ ฯลฯ ควรระมัดระวังเพื่อให้แน่ใจว่าทรัพย์สินทางปัญญาของนายจ้างและลูกค้าได้รับความคุ้มครอง
- Computer misuse (การใช้คอมพิวเตอร์ในทางที่ผิด)
  - วิศวกรซอฟต์แวร์ไม่ควรใช้ทักษะทางเทคนิคในการใช้คอมพิวเตอร์ของผู้อื่นในทางที่ผิด การใช้คอมพิวเตอร์ในทางที่ผิดจากระดับง่าย ๆ (เช่น เล่นเกมบนเครื่องของนายจ้าง) ไปจนถึงการกระทำที่รุนแรงมาก (เช่น การเผยแพร่ไวรัส)

# สรุป

- วิศวกรรมซอฟต์แวร์เป็นสาขาวิศวกรรมที่เกี่ยวข้องกับทุกด้านของการผลิตซอฟต์แวร์
- คุณลักษณะผลิตภัณฑ์ซอฟต์แวร์ที่จำเป็นคือ ความสามารถในการบำรุงรักษา เชื่อถือได้ ความมั่นคง ความปลอดภัย ประสิทธิภาพ และการยอมรับ
- กิจกรรมเกี่ยวกับการออกข้อกำหนด การพัฒนา การตรวจสอบ และการบำรุงรักษา เป็นส่วนประกอบของกระบวนการซอฟต์แวร์
- ความคิดพื้นฐานของวิศวกรรมซอฟต์แวร์ จะสามารถใช้งานได้กับทุกประเภทของการพัฒนาระบบซอฟต์แวร์
- ระบบซอฟต์แวร์มีหลายประเภท ต้องการเครื่องมือและเทคนิคทางวิศวกรรมซอฟต์แวร์ที่เหมาะสมสำหรับการพัฒนา
- ความคิดพื้นฐานของวิศวกรรมซอฟต์แวร์สามารถใช้ได้กับระบบซอฟต์แวร์ทุกประเภท
- วิศวกรซอฟต์แวร์มีความรับผิดชอบต่อวิชาชีพด้านวิศวกรรมและสังคม ไม่ควรสนใจแค่เพียงปัญหาทางเทคนิค
- สังคมระดับมืออาชีพ มีจรรยาบรรณไว้คอยกำหนดมาตรฐานเกี่ยวกับพฤติกรรมที่คาดหวังของสมาชิก ในสังคมของวิศวกรซอฟต์แวร์ก็เช่นเดียวกัน

คำถาม???