Requirements Engineering

Week 05

ดัดแปลงจาก slides ของหนังสือ Software Engineering [1]

หัวข้อที่จะศึกษา

- Functional and non-functional requirements
- Requirements engineering processes
- Requirements elicitation
- Requirements specification
- Requirements validation
- Requirements change

Requirements engineering

- กระบวนการสร้างสิ่งที่ลูกค้าต้องการจากระบบ
  - ข้อจำกัดในการดำเนินงานและการพัฒนา
- ความต้องการของระบบคืออะไร?
  - คำอธิบายสิ่งที่ระบบให้บริการ
  - ข้อจำกัด (ที่เกิดขึ้นในระหว่างกระบวนการทำ requirement)

What is a requirement?

- เป็นได้กว้างมาก
  - ตั้งแต่คำจำกัดความนามธรรมระดับสูงของบริการ
  - ข้อจำกัดของระบบ
  - ข้อกำหนดทางคณิตศาสตร์แบบละเอียด
- ข้อกำหนดมีหน้าที่หลักสองอย่าง ()
  - เป็นพื้นฐานสำหรับการเสนอราคา (สัญญาประมูลโครงการ) - ต้องเปิดกว้างสำหรับการตีความในภายหลัง
  - เป็นพื้นฐานสำหรับการทำสัญญา (ตรวจสอบตอนส่งมอบงาน) - ต้องเขียนไว้อย่างละเอียด
  - อาจต้องทำทั้งสองอย่างควบคู่กัน

Requirements abstraction (Davis)

- ถ้าบริษัทประสงค์จะทำสัญญาโครงการพัฒนาซอฟต์แวร์ขนาดใหญ่
  - ต้องกำหนด requirement ของตนในรูปแบบนามธรรมอย่างพอเพียง
    - ไม่ต้องกำหนดวิธีการที่ตายตัวไว้ล่วงหน้า
  - ต้องมีการเขียน requirement ให้ผู้รับเหมาหลายรายสามารถร่วมประมูลได้
    - มีรายละเอียดพอเพียงที่จะสามารถเสนอราคาเพื่อทำสัญญา
    - ผู้รับเหมาสามารถเสนอวิธีที่แตกต่างกัน เพื่อตอบสนองต่อ requirement
  - เมื่อชนะการประมูล ผู้รับเหมาต้องเขียนคำนิยามของระบบสำหรับลูกค้าในรายละเอียด เพื่อให้ลูกค้าเข้าใจและสามารถตรวจสอบว่าซอฟต์แวร์จะทำงานอย่างไร
    - เอกสารทั้งสองนี้ เรียกว่าเอกสารข้อกำหนดสำหรับระบบ (requirements document for the system)

Types of requirement

- ความต้องการของผู้ใช้
  - การบรรยายโดยใช้ภาษาธรรมชาติ รวมทั้งแผนผังของบริการที่ระบบให้และข้อจำกัดในการดำเนินงาน
  - เขียนขึ้นสำหรับลูกค้า
- ความต้องการของระบบ
  - เอกสารที่มีโครงสร้างที่ชัดเจน แสดงคำอธิบายโดยละเอียด
    - เกี่ยวกับฟังก์ชันที่ซอฟต์แวร์ให้บริการ
    - เกี่ยวกับข้อจำกัดในการดำเนินงานของระบบและสิ่งที่จำเป็นต้องนำมาใช้
  - เขียนขึ้นเพื่อเป็นส่วนหนึ่งของสัญญาระหว่างลูกค้ากับผู้รับเหมา

Users of requirement documents

- User requirements
  - Client managers
  - System end-user
  - Client Engineers
  - Contractor manager
  - System architects
- System requirements
  - System end-users
  - Client Engineers
  - System architects
  - Software developers

System stakeholders

- บุคคลหรือองค์กรใด ๆ ที่ได้รับผลกระทบจากระบบด้วยวิธีใดก็ตาม รวมทั้งผู้ที่มีส่วนได้เสียตามกฎหมาย
- ประเภทผู้มีส่วนได้ส่วนเสีย
  - ผู้ใช้ (End users)
  - ผู้จัดการระบบ (System managers)
  - เจ้าของระบบ (System owners)
  - ผู้มีส่วนได้เสียภายนอก (External stakeholders)

Functional and non-functional requirements

Functional and non-functional requirements

- Functional requirements
  - การให้บริการที่ระบบควรเตรียมไว้ให้
  - วิธีการที่ระบบตอบสนองต่อ input
  - วิธีการที่ระบบควรปฏิบัติในสถานการณ์เฉพาะ
  - รวมถึงอาจระบุไว้ด้วย ว่าระบบจะไม่ทำอะไรบ้าง
- Non-functional requirements
  - ข้อจำกัดของ services หรือ functions ของระบบ
    - เช่น ข้อจำกัดด้านเวลา ข้อจำกัดในการพัฒนา มาตรฐานต่าง ๆ ที่นำมาใช้ ฯลฯ
  - ใช้กับระบบโดยรวมมากกว่าใช้กับคุณลักษณะ (features) หรือบริการ (services) แต่ละอย่าง
- Domain requirements
  - ข้อจำกัดในระบบจากโดเมนของการดำเนินงาน

Functional requirements

- คำอธิบายการทำงานหรือบริการของระบบ
  - ขึ้นอยู่กับชนิดของซอฟต์แวร์
  - ขึ้นอยู่กับผู้ใช้
  - ขึ้นอยู่กับประเภทของระบบที่ใช้ซอฟต์แวร์
- ความต้องการของผู้ใช้
  - ควรเป็นคำอธิบายการทำงานในระดับสูง ว่าอะไรคือสิ่งที่ระบบควรทำ
- ความต้องการของระบบ
  - ควรเป็นคำอธิบายการบริการของระบบอย่างละเอียด

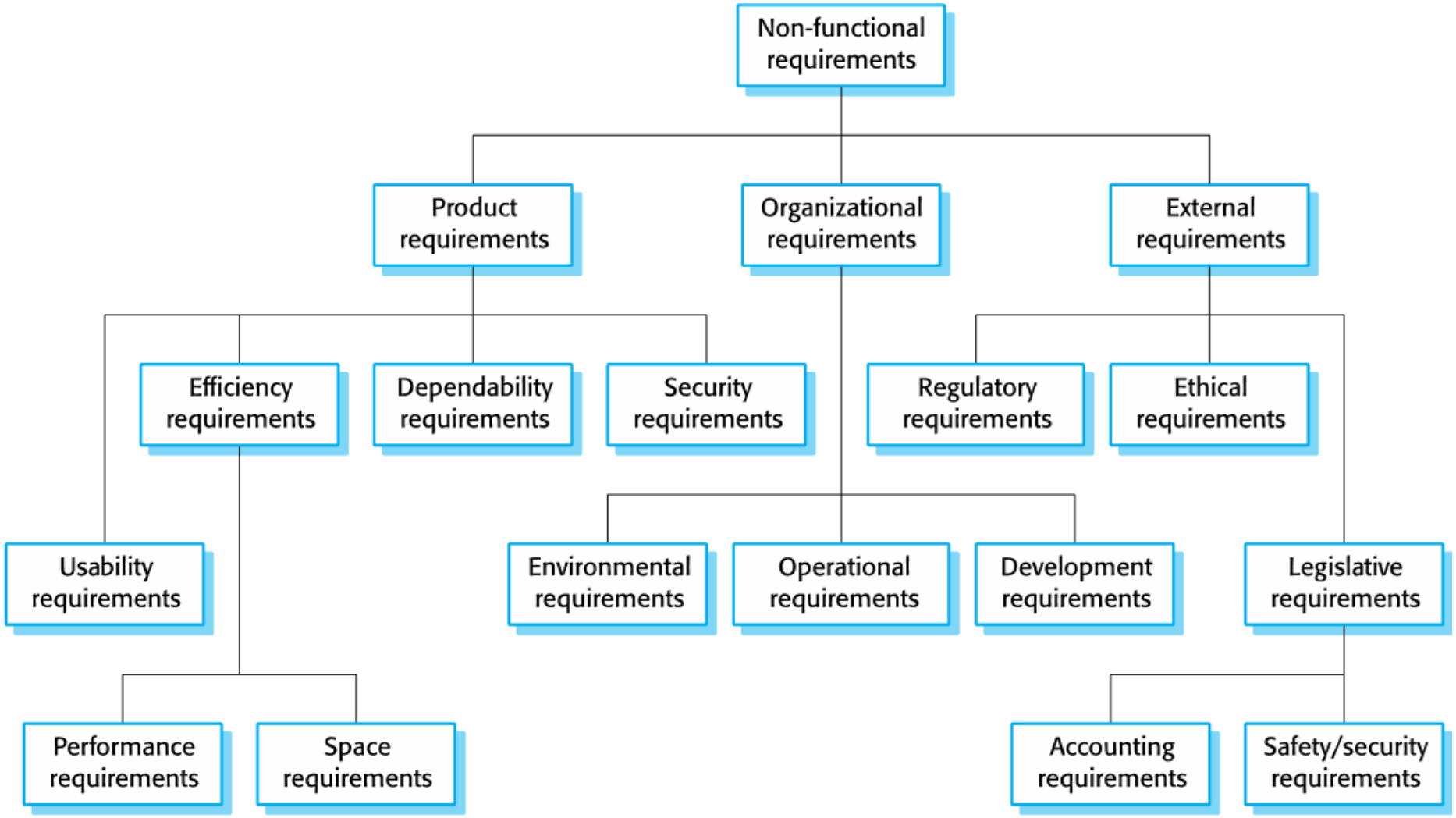
Requirements completeness and consistency

- โดยหลักการ ข้อกำหนดต่าง ๆ ควรมีความสมบูรณและคงเส้นคงวา
- ความสมบูรณ์
  - ควรมีคำอธิบายเกี่ยวกับสิ่งอำนวยความสะดวกทั้งหมดที่จำเป็น
- คงเส้นคงวา
  - ไม่ควรมีข้อขัดแย้งหรือข้อโต้แย้ง ในคำอธิบายเกี่ยวกับสิ่งอำนวยความสะดวกของระบบ
- ในทางปฏิบัติ เนื่องจากความซับซ้อนของระบบและสิ่งแวดล้อม จึงเป็นไปไม่ได้ที่จะจัดทำเอกสารข้อกำหนดที่ครบถ้วนและคงเส้นคงวา

Non-functional requirements

- สิ่งที่มีผลต่อคุณสมบัติและข้อจำกัดของระบบ
  - เช่น Reliability, Response time, Storage requirements
  - ข้อจำกัดคือ ความสามารถของอุปกรณ์ I/O ฯลฯ
- ในบางครั้งอาจมีความต้องการให้ใช้เครื่องมือเฉพาะ
  - เช่น IDE, ภาษาโปรแกรม หรือวิธีการพัฒนา
- Non-functional requirements อาจมีความสำคัญมากกว่า functional requirements
  - ในบางกรณี หากไม่ได้ทำตาม non-functional requirements ระบบอาจไม่มีประโยชน์

Types of nonfunctional requirement



Non-functional requirements implementation

- Non-functional requirements อาจส่งผลต่อสถาปัตยกรรมโดยรวมของระบบมากกว่าจะกระทบต่อส่วนประกอบย่อย
  - เช่น เราอาจต้องจัดระบบเสียใหม่ เพื่อลดการสื่อสารระหว่าง components
- Non-functional requirements บางอย่าง (เช่น requirements ด้านความปลอดภัย) อาจสร้าง requirements ด้านการทำงานที่เกี่ยวข้องจำนวนหนึ่ง ที่มีผลกระทบต่อ requirements ในส่วนอื่น ๆ
  - อาจไปเพิ่มข้อจำกัด ให้ requirements เดิมที่มีอยู่

Non-functional classifications

- ความต้องการของผลิตภัณฑ์ Product requirements
  - ข้อกำหนดที่ระบุว่าผลิตภัณฑ์ที่ส่งมอบจะต้องทำงานในลักษณะใดวิธีหนึ่งเช่น ความเร็วในการดำเนินการ ความน่าเชื่อถือ ฯลฯ
- ความต้องการขององค์กร Organisational requirements
  - ข้อกำหนดที่เป็นผลมาจากนโยบายและขั้นตอนขององค์กรเช่น มาตรฐานกระบวนการที่ใช้ ความต้องการในการใช้งานเป็นต้น
- ข้อกำหนดภายนอก External requirements
  - ความต้องการที่เกิดขึ้นจากปัจจัยภายนอกที่มีต่อระบบและกระบวนการพัฒนา เช่น ข้อกำหนดการทำงานร่วมกัน, ข้อกำหนดทางกฎหมายเป็นต้น

Goals and requirements

- เป็นการยากที่จะระบุ Non-functional requirements ได้อย่างแม่นยำ และอาจเป็นเรื่องยากที่จะตรวจสอบ
- เป้าหมาย
  - ความต้องการโดยทั่วไปของผู้ใช้ เช่น ความสะดวกในการใช้งาน
- การตรวจสอบ Non-functional requirements
  - ระบุวิธีการตรวจสอบโดยใช้มาตรการบางอย่างที่สามารถทดสอบได้อย่างเป็นกลาง
- การกำหนดเป้าหมาย มีประโยชน์สำหรับนักพัฒนาซอฟต์แวร์เนื่องจากมันถ่ายทอดความตั้งใจของผู้ใช้ระบบมาถึงนักพัฒนาโดยตรง

Usability requirements

- ตัวอย่าง requirement ที่ใช้ได้
- เป้าหมาย
  - เจ้าหน้าที่ทางการแพทย์ควรใช้ระบบได้โดยง่าย และควรมีการจัดระเบียบเพื่อให้เกิดความผิดพลาดจากผู้ใช้น้อยที่สุด
- การตรวจสอบ Non-functional requirements
  - เจ้าหน้าที่ทางการแพทย์สามารถใช้งานระบบทั้งหมด หลังจากผ่านการฝึกอบรมไปแล้ว 4 ชั่วโมง
  - หลังจากการฝึกอบรมนี้ จำนวนข้อผิดพลาดโดยเฉลี่ยของผู้ใช้ระบบจะต้องไม่เกินสองครั้งต่อหนึ่งชั่วโมง

Metrics for specifying nonfunctional requirements

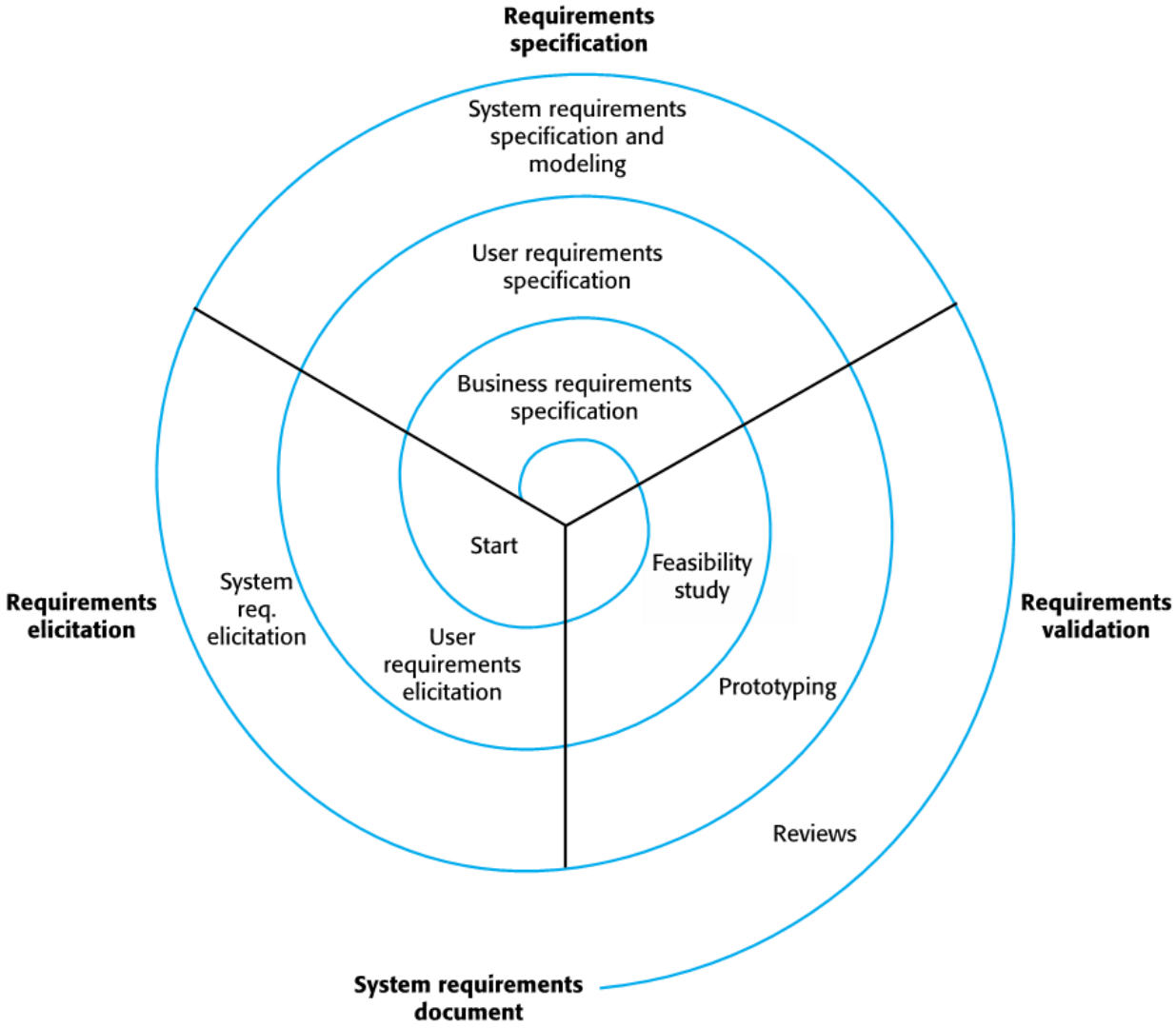
Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Requirements engineering processes

Requirements engineering processes

- กระบวนการที่ใช้สำหรับ RE แตกต่างกันไปขึ้นอยู่กับ
  - Application domain
  - บุคคลที่เกี่ยวข้อง
  - องค์กรที่พัฒนาข้อกำหนด
- อย่างไรก็ตามมีกิจกรรมสากลที่เกี่ยวข้องกับกระบวนการ RE ได้แก่
  - การสอบถามความต้องการ (Requirements elicitation)
  - การวิเคราะห์ความต้องการ Requirements analysis
  - การตรวจสอบความต้องการ Requirements validation
  - การจัดการความต้องการ Requirements management
- ในทางปฏิบัติ RE กิจกรรมต่าง ๆ จะเป็นกระบวนการที่ทำควบคู่กันไปได้

A spiral view of the requirements engineering process



Requirements elicitation

Requirements elicitation and analysis

- บางครั้งเรียกว่าการกระตุ้นความต้องการหรือการค้นพบความต้องการ
- ประกอบด้วย staffs ที่ทำงานกับลูกค้า
  - เพื่อหาข้อมูลเกี่ยวกับ domain application
  - บริการที่ระบบควรให้
  - ข้อจำกัดในการดำเนินงานของระบบ
- บุคคลที่เกี่ยวข้อง ประกอบด้วย
  - end-users, managers, engineers ที่มีส่วนในการ maintenance, domain experts, trade unions, เป็นต้น
    - เรียกรวม ๆ ว่า stakeholders.

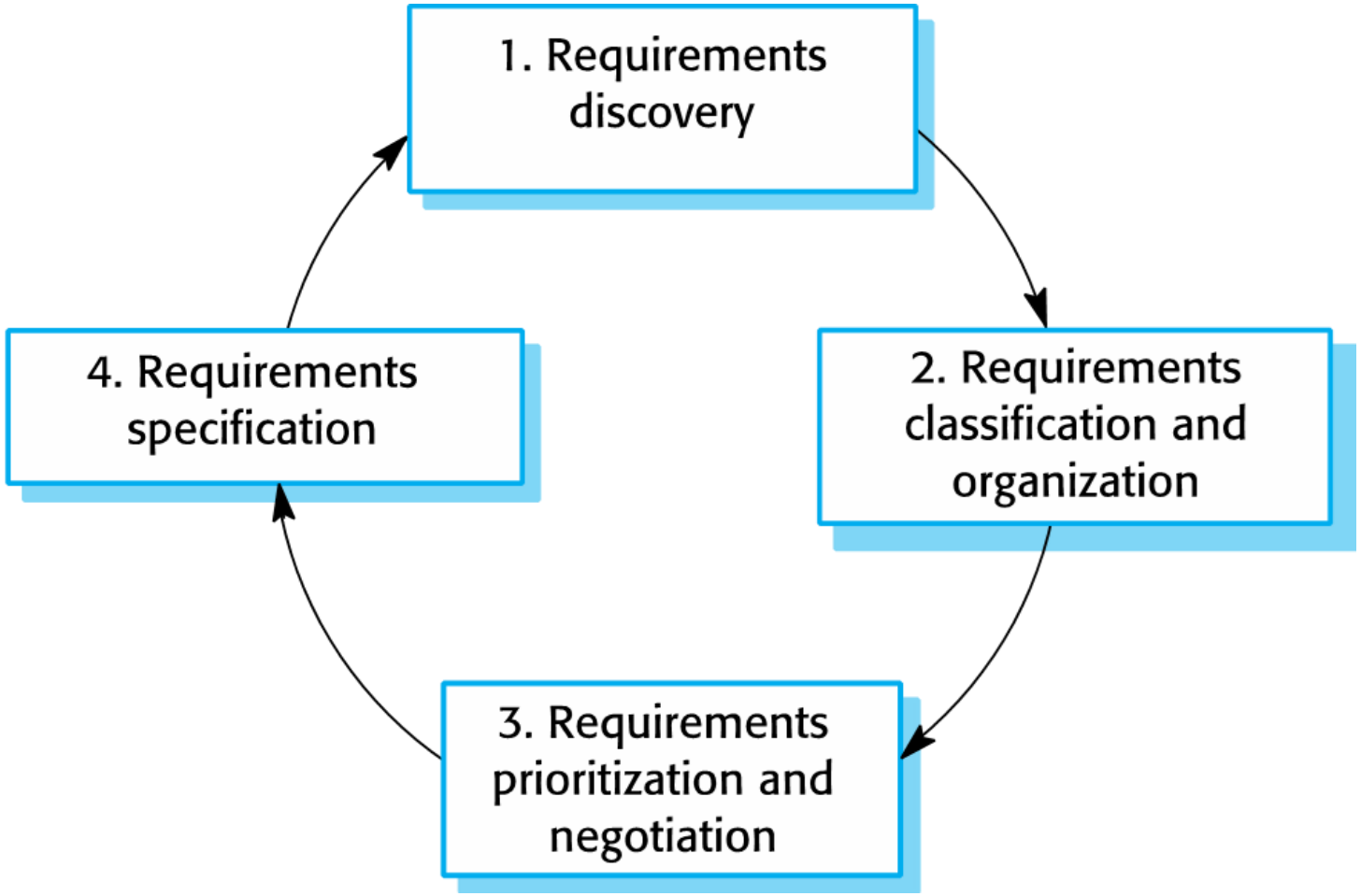
Requirements elicitation

- ขั้นตอนประกอบด้วย:
  - การค้นพบความต้องการ (Requirements discovery)
  - การจำแนกความต้องการและการจัดระเบียบ (Requirements classification and organization)
  - จัดลำดับความสำคัญและการเจรจาต่อรอง (Requirements prioritization and negotiation)
  - การกำหนดข้อกำหนด (Requirements specification)

Problems of requirements elicitation

- ผู้มีส่วนได้ส่วนเสียไม่ทราบว่าพวกเขาต้องการอะไร
- ผู้มีส่วนได้ส่วนเสียอธิบายความต้องการในภาษาของตนเอง
- ผู้มีส่วนได้ส่วนเสียแต่ละฝ่ายอาจมีความต้องการที่ขัดแย้งกัน
- ปัจจัยด้านองค์กรและการเมือง อาจส่งผลต่อความต้องการของระบบ
- มีการเปลี่ยนแปลงความต้องการในระหว่างกระบวนการวิเคราะห์
  - อาจจะมีผู้มีส่วนได้ส่วนเสียใหม่เกิดขึ้น
  - อาจจะมีการเปลี่ยนแปลงสภาพแวดล้อมทางธุรกิจ

The requirements elicitation and analysis process



Process activities

- Requirements discovery
  - การมีปฏิสัมพันธ์กับผู้มีส่วนได้ส่วนเสีย เพื่อการค้นพบ requirements
  - requirements ของโดเมนจะถูกค้นพบในขั้นตอนนี้
- Requirements classification and organisation
  - จัดกลุ่ม requirements ที่เกี่ยวข้องและจัดระเบียบให้สอดคล้องกัน
- Prioritisation and negotiation
  - จัดลำดับความสำคัญของ requirements และแก้ไขข้อขัดแย้งของ requirement
- Requirements specification
  - นำ requirement มาสร้างเป็นเอกสารและป้อนเข้าไปในรอบต่อไป spiral

Requirements discovery

- กระบวนการในการรวบรวมข้อมูลเกี่ยวกับระบบ
  - ทั้งสิ่งที่ต้องการและที่สิ่งมีอยู่แล้ว
  - กลั่นกรองออกมาเป็น user requirement และ system requirement จากข้อมูลที่ได้
- การมีปฏิสัมพันธ์กับผู้มีส่วนได้เสียของระบบ เริ่มจากผู้จัดการไปยังหน่วยงานกำกับดูแลภายนอก
- โดยปกติ ระบบมักมีผู้มีส่วนได้เสียหลายกลุ่ม

Interviewing

- ในกระบวนการ RE มักจะมีการสัมภาษณ์ผู้มีส่วนได้ส่วนเสีย ทั้งอย่างเป็นทางการและไม่เป็นทางการ
- ประเภทของการสัมภาษณ์
  - การสัมภาษณ์แบบคำถามปลายปิด ตามรายการคำถามที่กำหนดไว้ล่วงหน้า
  - การสัมภาษณ์แบบคำถามปลายเปิดที่มีการสำรวจประเด็นต่าง ๆ กับผู้มีส่วนได้ส่วนเสีย
- การสัมภาษณ์ที่มีประสิทธิภาพ
  - สัมภาษณ์อย่างใจกว้าง -- เต็มใจที่จะฟังผู้มีส่วนได้ส่วนเสีย
  - หลีกเลี่ยง requirement ที่คิดไว้ล่วงหน้า
  - ใช้คำถามที่มีประโยชน์ เพื่อให้ได้ requirements
  - ทดลองใช้ prototype system ร่วมกัน

Interviews in practice

- โดยปกติการสัมภาษณ์จะทำทั้งแบบปลายปิดและแบบปลายเปิด
- การสัมภาษณ์เป็นสิ่งที่ดีสำหรับการทำความเข้าใจโดยรวมเกี่ยวกับสิ่งที่ผู้มีส่วนได้ส่วนเสียต้องการ และจะได้รู้ว่าพวกเขามีปฏิสัมพันธ์กับระบบอย่างไร
- ผู้สัมภาษณ์ต้องมีใจกว้างโดยไม่ต้องคิดล่วงหน้าว่าระบบควรทำงานอย่างไร
- พยายามให้ผู้ใช้อธิบายเกี่ยวกับระบบ
  - พูดจูงใจให้เขา บอกหรือแนะนำ สิ่งที่ต้องการจากระบบ
  - อย่าไปตั้งคำถามโง่ ๆ ว่า “คุณต้องการอะไรจากระบบ”

Problems with interviews

- Application specialists อาจจะพูดคนละภาษากับ requirements engineer
  - พูดคนละภาษารวมถึงใช้ภาษาพูดเดียวกันแต่ทำความเข้าใจยาก
- การสัมภาษณ์ ไม่สามารถใช้ได้ดับการทำความเข้าใจข้อกำหนดของโดเมน
  - requirements engineer มักไม่สามารถเข้าใจศัพท์เฉพาะของโดเมน (เช่นศัพท์ทางการแพทย์)
  - ความรู้เกี่ยวกับโดเมนบางอย่าง ดูเหมือนจะเข้าใจได้ง่าย แต่พบว่าเป็นการยากที่จะพูดหรือคิดว่าไม่น่าจะพูดออกมาเป็นคำพูดได้
    - ดังนั้นจะผิดพลาดมาก ถ้าเขียน requirements ออกมาโดยใช้ความเข้าใจของผู้ดำเนินการสัมภาษณ์เองล้วน ๆ

Stories and scenarios

- สถานการณ์ (Scenarios) และเรื่องราวของผู้ใช้ (user stories) เป็นตัวอย่างชีวิตจริงในการใช้ระบบ
- เรื่องราวและสถานการณ์เป็นคำอธิบายว่าระบบอาจใช้งานใดได้บ้าง
- เนื่องจากผู้มีส่วนได้เสียมักจะทำงานอยู่บนพื้นฐานของสถานการณ์จริง
  - ต้องฟังเรื่องราวของพวกเขา
  - เราสามารถแสดงความคิดเห็นเกี่ยวกับสถานการณ์หรือเรื่องราวของพวกเขา แต่ต้องทำด้วยความเคารพและให้เกียรติกันและกัน

Scenarios

- เรื่องราวของผู้ใช้ที่มีโครงสร้างอย่างเป็นรูปแบบ
- สถานการณ์ (Scenarios) ควรประกอบด้วย
  - คำอธิบายเมื่อตอนเริ่มต้นของสถานการณ์
  - คำอธิบายเกี่ยวกับการดำเนินเหตุการณ์ตามปกติของสถานการณ์
  - คำอธิบายของสิ่งผิดพลาดที่อาจเกิดขึ้นได้
  - ข้อมูลเกี่ยวกับกิจกรรมอื่น ๆ ที่อาจจะเกิดขึ้นในขณะเดียวกัน
  - คำอธิบายสถานะเมื่อสถานการณ์เสร็จสิ้น

Requirements specification

- เป็นขั้นตอนการเขียนข้อกำหนดของผู้ใช้และระบบในเอกสารข้อกำหนด
- User requirement ต้องเป็นที่เข้าใจของ end user และ customer ที่ไม่มีพื้นฐานทางเทคนิค
- System requirement เป็นข้อกำหนดที่ละเอียดขึ้น และอาจรวมถึงข้อมูลทางเทคนิคเพิ่มเติม
- Requirement อาจเป็นส่วนหนึ่งของสัญญาในการพัฒนาระบบ
  - ดังนั้นสิ่งสำคัญคือต้องทำให้ “สมบูรณ์แบบ” ที่สุด

Ways of writing a system requirements specification

Notation	Description
ใช้ภาษาธรรมชาติ	• Requirement เขียนโดยใช้ประโยคที่เรียงลำดับเลข เป็นภาษาธรรมชาติ • แต่ละประโยคควรแสดงความต้องการเพียงประเด็นเดียวเท่านั้น
ภาษาธรรมชาติที่มีโครงสร้าง	• Requirement เขียนด้วยภาษาธรรมชาติ ในรูปแบบหรือเทมเพลตมาตรฐาน • แต่ละฟิลด์จะให้ข้อมูลเกี่ยวกับข้อกำหนด ของ requirement
ภาษาคำอธิบายการออกแบบ	• วิธีนี้ใช้ภาษา เช่น ภาษาเขียนโปรแกรม แต่มีคุณสมบัติที่เป็นนามธรรมมากขึ้น เพื่อระบุrequirement โดยการกำหนดรูปแบบการดำเนินงานของระบบ • วิธีนี้ใช้งานไม่บ่อย แม้ว่าจะเป็นประโยชน์สำหรับข้อกำหนดของอินเทอร์เฟซ

Ways of writing a system requirements specification

Notation	Description
เครื่องหมายแบบกราฟิก	• โมเดลแบบกราฟิก ที่เสริมด้วยคำอธิบายประกอบแบบข้อความ • ใช้เพื่อกำหนดความต้องการในการทำงานของระบบ โดยทั่วไป มักใช้ use case และ sequence diagram
Mathematical specifications	• ใช้เครื่องหมายหรือสัญลักษณ์ทางคณิตศาสตร์ เช่น finite-state machines หรือ sets • แม้ว่าข้อกำหนดที่ชัดเจนเหล่านี้สามารถลดความคลุมเครือในเอกสารแต่ ลูกค้าส่วนใหญ่มักไม่เข้าใจข้อกำหนดที่เป็นทางการ • พวกเขามักจะยืนยันว่าไม่สามารถตรวจสอบว่าเป็น model ที่พวกเขาต้องการหรือไม่ และไม่เต็มใจที่จะยอมรับว่าเป็นสัญญาในการพัฒนาระบบ

Requirements and design

- โดยหลักการ requirement ควรระบุว่าระบบควรทำอย่างไร และการออกแบบควรอธิบายถึงวิธีการเหล่านั้น
- ในทางปฏิบัติ requirement และการออกแบบจะแยกออกจากกันไม่ได้
- Requirement จะถูกเขียนเป็นประโยคภาษาธรรมชาติเสริมด้วยไดอะแกรมและตาราง
  - ภาษาธรรมชาติ ใช้สำหรับเขียนข้อกำหนดเนื่องจากใช้งานง่ายและเป็นสากล ซึ่งหมายความว่าผู้ใช้และลูกค้าสามารถเข้าใจ requirement ได้ตรงกัน

Guidelines for writing requirements

- คิดค้นรูปแบบมาตรฐานและใช้สำหรับทุก requirement
- ใช้ภาษาอย่างสม่ำเสมอ
  - ใช้คำว่า “ต้อง” เพื่อแสดง ข้อกำหนดที่บังคับ (mandatory requirements)
  - ใช้คำว่า “ควร” เพื่อแสดง ข้อกำหนดพึงประสงค์ (desirable requirements)
- ใช้ไฮไลต์ข้อความเพื่อระบุส่วนสำคัญของ requirements
- หลีกเลี่ยงการใช้ศัพท์แสงทางคอมพิวเตอร์ (computer jargon)
- ระบุเหตุผล ที่ต้องมี requirements นั้น ๆ

Problems with natural language

- ขาดความชัดเจน
  - ยิ่งทำให้มีความแม่นยำสูงขึ้น ก็จะยิ่งทำให้อ่านเอกสารได้ยากขึ้น (กลายเป็นภาษาทางกฎหมาย)
- มีความสับสนในเรื่อง requirements
  - Functional และ non-functional requirements มีแนวโน้มที่จะผสมกันมากขึ้น
- การควบรวม requirements
  - Requirements หลาย ๆ อย่างสามารถเขียนรวมกันได้ ทำให้คนรับช่วงงานมีความยากลำบากในการตีความ

Structured specifications

- วิธีการในการเขียนข้อกำหนดที่มีอิสระ มีอยู่อย่างจำกัด และ Requirement มักจะถูกเขียนขึ้นในรูปแบบมาตรฐาน
- วิธีนี้ทำงานได้ดีกับข้อกำหนดบางประเภท เช่น ข้อกำหนดสำหรับระบบควบคุมแบบฝังตัว
  - บางครั้งก็ไม่ยืดหยุ่นสำหรับการเขียนข้อกำหนดทางธุรกิจ

Form-based specifications

- นิยามของฟังก์ชันหรือ entity
- คำอธิบายของ input และแหล่งที่มา
- คำอธิบายของ output และแหล่งปลายทาง
- ข้อมูลเกี่ยวกับข่าวสารที่จำเป็นสำหรับการประมวลผล
- คำอธิบายของการดำเนินการที่จะต้องดำเนินการ
- เงื่อนไขก่อนและหลัง (ถ้ามี)
- ผลกระทบของฟังก์ชั่น (ถ้ามี)

ตัวอย่าง A structured specification of a requirement for an insulin pump

Insulin Pump/Control Software/SRS/3.3.2

Function Compute insulin dose: safe sugar level.

Description

Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.

Inputs Current sugar reading (r2); the previous two readings (r0 and r1).

Source Current sugar reading from sensor. Other readings from memory.

Outputs CompDose—the dose in insulin to be delivered.

Destination Main control loop.

ตัวอย่าง A structured specification of a requirement for an insulin pump

Action
CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.
Requirements
Two previous readings so that the rate of change of sugar level can be computed.
Pre-condition
The insulin reservoir contains at least the maximum allowed single dose of insulin.
Post-condition r0 is replaced by r1 then r1 is replaced by r2.
Side effects None.

Tabular specification

- ใช้เพื่อเสริมภาษาธรรมชาติ
- มีประโยชน์อย่างยิ่ง เมื่อต้องกำหนดจำนวนของทางเลือกที่เป็นไปได้ของการดำเนินการ
- ยกตัวอย่าง เช่น ระบบปั๊มอินซูลิน
  - คำนวณหาอัตราการเปลี่ยนแปลงระดับน้ำตาลในเลือด
  - Requirement ในรูปแบบตาราง จะอธิบายถึงวิธีคำนวณความต้องการอินซูลินในสถานการณ์ต่างๆ

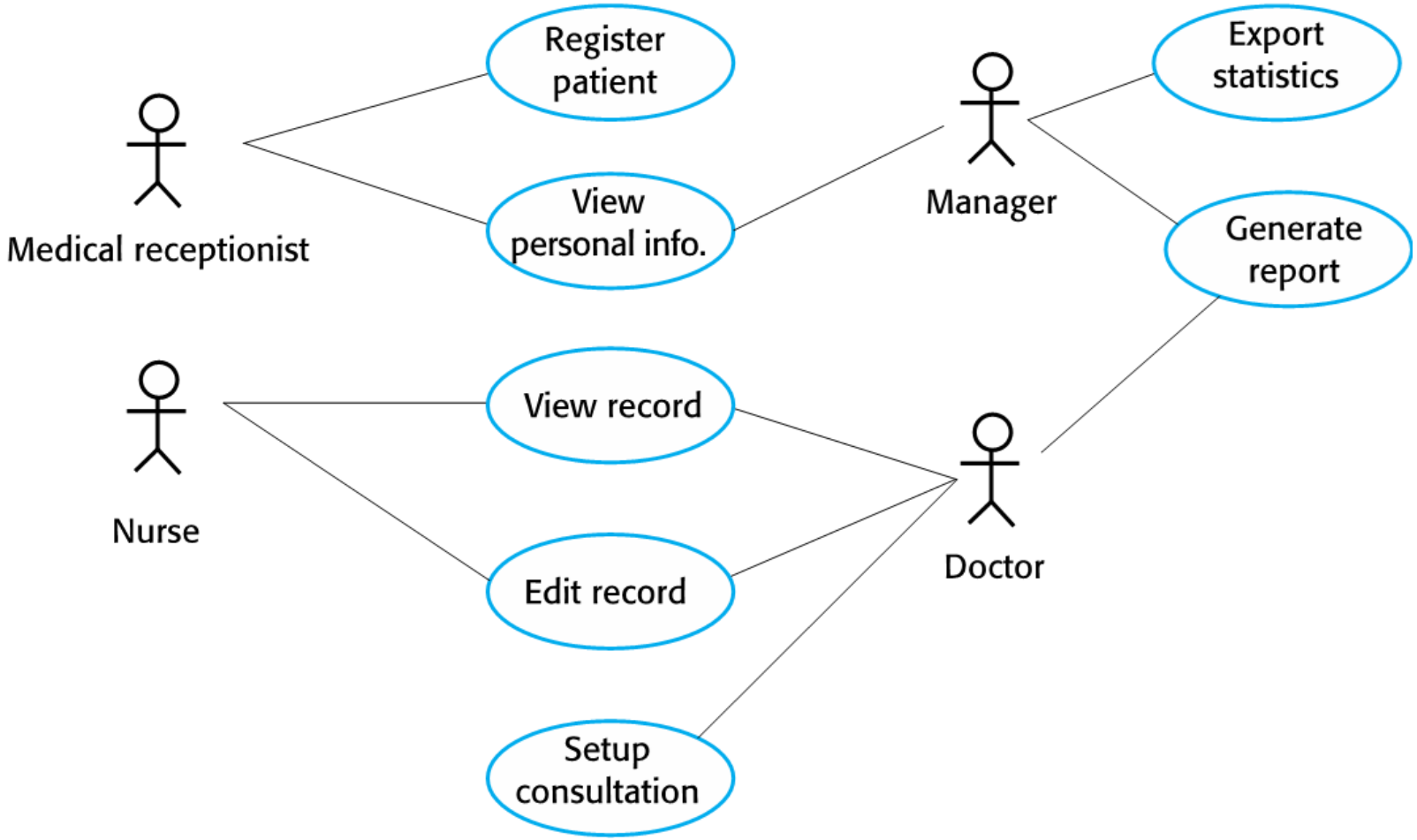
Tabular specification of computation for an insulin pump

Condition	Action
Sugar level falling (r2 < r1)	CompDose = 0
Sugar level stable (r2 = r1)	CompDose = 0
Sugar level increasing and rate of increase decreasing ((r2 – r1) < (r1 – r0))	CompDose = 0
Sugar level increasing and rate of increase stable or increasing ((r2 – r1) ≥ (r1 – r0))	CompDose = round ((r2 – r1)/4) If rounded result = 0 then CompDose = MinimumDose

Use cases

- Use-case เป็นสถานการณ์สมมติที่ผนวกอยู่ใน UML
- Use-case ระบุ actor ที่อยู่ในปฏิสัมพันธ์และอธิบายการปฏิสัมพันธ์ด้วยตัวเอง
- กลุ่มของ use-case อธิบายปฏิสัมพันธ์ที่เป็นไปได้ทั้งหมดของระบบ
- โมเดลกราฟิกระดับสูงนี้ อาจเสริมด้วยรายละเอียดเพิ่มเติมแบบตาราง
- Sequence diagram ของ UML อาจใช้เพื่อเพิ่มรายละเอียดให้กับ use-case โดยการแสดงลำดับของการประมวลผลเหตุการณ์ในระบบ

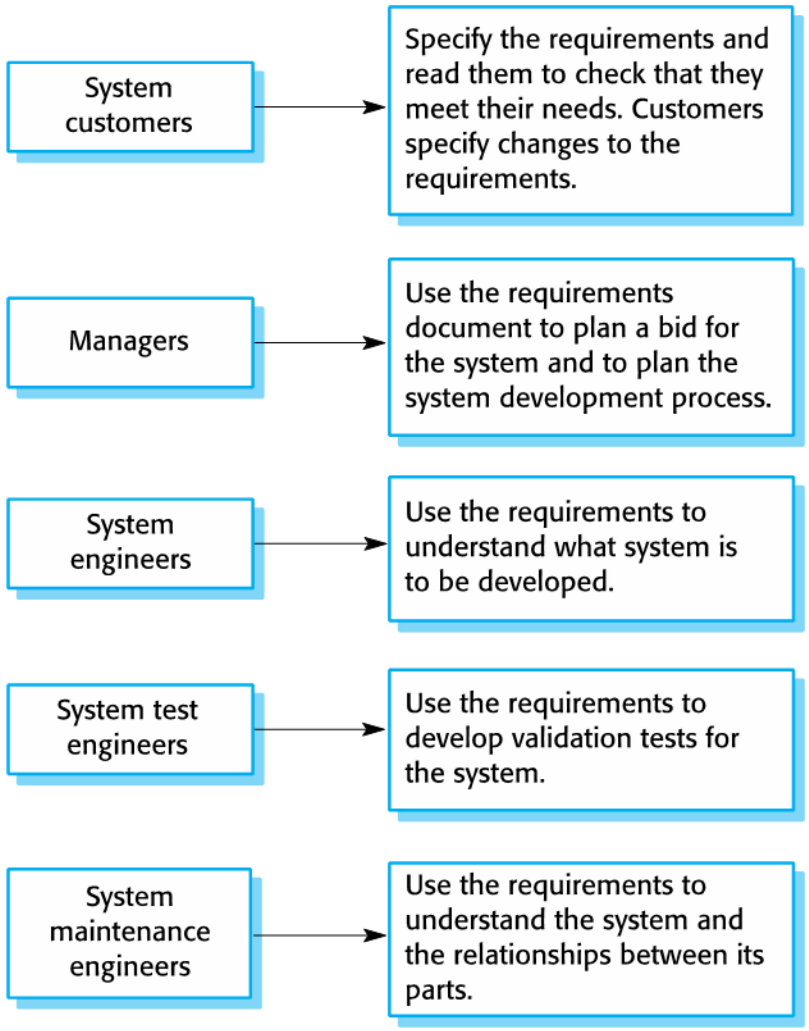
ตัวอย่าง use-case diagram



The software requirements document

- เอกสารข้อกำหนดของซอฟต์แวร์เป็นคำอธิบายอย่างเป็นทางการว่านักพัฒนาระบบต้องใช้อะไรบ้าง
- ควรมีทั้งนิยามของ user requirements และข้อกำหนดของ system requirements
- มันไม่ใช่เอกสารการออกแบบ
  - ควรบอกว่าระบบควรทำอะไร (WHAT the system should do)
  - ไม่ควรบอกว่าระบบควรทำงานอย่างไร (HOW the system should do)

Users of a requirements document



Requirements document variability

- ข้อมูลในเอกสาร requirement ขึ้นอยู่กับชนิดของระบบและแนวทางการพัฒนาที่ใช้
- ระบบที่พัฒนาขึ้นเรื่อย ๆ (incremental) จะมีรายละเอียดในเอกสาร requirement น้อยกว่า
- เอกสาร requirement ได้รับการออกแบบให้เป็นมาตรฐาน เช่น มาตรฐาน IEEE
  - สามารถใช้งานได้กับ requirement สำหรับโครงการวิศวกรรมระบบขนาดใหญ่

The structure of a requirements document

Chapter	Description
Preface (คำนำ)	• ควรกำหนดผู้อ่านที่คาดหวังไว้ในเอกสาร • อธิบายประวัติของเวอร์ชันรวมถึง • เหตุผลสำหรับการสร้างเวอร์ชันใหม่ • สรุปการเปลี่ยนแปลงที่ทำในแต่ละเวอร์ชัน
Introduction (บทนำ)	• ความจำเป็นของระบบ ควรอธิบายสั้น ๆ เกี่ยวกับหน้าที่ของระบบ • อธิบายว่ามันจะทำงานร่วมกับระบบอื่นได้อย่างไร • ควรอธิบายว่าระบบนี้เหมาะสมกับธุรกิจโดยรวม • วัตถุประสงค์เชิงกลยุทธ์ขององค์กรที่ได้รับการว่าจ้างซอฟต์แวร์
Glossary (อภิธานศัพท์)	• ควรกำหนดศัพท์ทางเทคนิคที่ใช้ในเอกสาร • ไม่ควรตั้งสมมติฐานเกี่ยวกับประสบการณ์หรือความเชี่ยวชาญของผู้อ่าน

The structure of a requirements document

Chapter	Description
User requirements definition (ข้อกำหนดความต้องการของผู้ใช้)	• อธิบายบริการที่มีให้สำหรับผู้ใช้ • ควรอธิบาย nonfunctional system requirements ในส่วนนี้ด้วย • คำอธิบายนี้อาจใช้ภาษาธรรมชาติ ไดอะแกรม หรือข้อมูลอื่น ๆ ที่เข้าใจได้ง่ายสำหรับลูกค้า • ควรระบุมาตรฐานผลิตภัณฑ์และกระบวนการที่ต้องปฏิบัติตาม
System architecture (สถาปัตยกรรมระบบ)	• นำเสนอภาพรวมระดับสูงของสถาปัตยกรรมระบบที่คาดการณ์ไว้ • แสดงการกระจายของฟังก์ชันต่าง ๆ ในโมดูลระบบ • ควรเน้นองค์ประกอบทางสถาปัตยกรรมที่นำกลับมาใช้ใหม่
System requirements specification (ข้อกำหนด requirement ของระบบ)	• ควรอธิบาย functional และ non-functional requirement ในรายละเอียดเพิ่มเติม • หากจำเป็น เพิ่มเติมรายละเอียด non-functional requirement เข้าไปอีก • อาจมีการเชื่อมต่อกับระบบอื่น ๆ

The structure of a requirements document

Chapter	Description
System models	• อาจรวมถึงรูปแบบระบบกราฟิกที่แสดงความสัมพันธ์ระหว่างส่วนประกอบของระบบกับระบบและสภาพแวดล้อม • ตัวอย่างของรูปแบบที่เป็นไปได้คือ object models, data-flow models หรือ semantic data models โมเดลอ็อบเจ็กต์ โมเดลการไหลของข้อมูล แบบจำลองข้อมูลความหมาย
System evolution	• ควรจะอธิบายสมมติฐานพื้นฐานเกี่ยวกับระบบที่ใช้และการเปลี่ยนแปลงที่คาดว่าจะเกิดขึ้น • เช่น จากวิวัฒนาการของฮาร์ดแวร์ การเปลี่ยนแปลงความต้องการของผู้ใช้ เป็นต้น • ส่วนนี้มีประโยชน์สำหรับนักออกแบบระบบ เพราะอาจช่วยให้พวกเขาตัดสินใจในการออกแบบ ซึ่งอาจจำกัดการเปลี่ยนแปลงของระบบในอนาคต

The structure of a requirements document

Chapter	Description
Appendices	• ควรให้รายละเอียดข้อมูลเฉพาะที่เกี่ยวข้องกับแอปพลิเคชันที่กำลังพัฒนา ตัวอย่างเช่น ฮาร์ดแวร์และคำอธิบายฐานข้อมูล • ข้อกำหนดฮาร์ดแวร์ จะกำหนดฮาร์ดแวร์ที่น้อยที่สุดและเหมาะสมที่สุดสำหรับระบบ • ความต้องการฐานข้อมูล กำหนดองค์กรเชิงตรรกะของข้อมูลที่ใช้โดยระบบและความสัมพันธ์ระหว่างข้อมูล
Index	• อาจรวมดัชนีหลาย ๆ อย่างไว้ในเอกสาร • เช่นเดียวกับดัชนีตัวอักษรปกติ อาจมีดัชนีของไดอะแกรม ดัชนีของฟังก์ชันและอื่น ๆ

Requirements validation

Requirements validation

- เกี่ยวข้องกับการแสดงให้เห็นว่า requirement ได้กำหนดระบบที่ลูกค้าต้องการจริงๆ
- ค่าใช้จ่ายเนื่องจากความผิดพลาดของ requirement มีค่าสูง ดังนั้นการตรวจสอบความถูกต้องจึงมีความสำคัญมาก
  - การแก้ไขข้อผิดพลาดจาก requirement หลังจากการจัดส่ง อาจเสียค่าใช้จ่ายถึง 100 เท่าของค่าใช้จ่ายในการแก้ไขข้อผิดพลาดขณะ implementation ระบบ

Requirements checking

- ความถูกต้อง (Validity)
  - ระบบมีฟังก์ชันที่รองรับความต้องการของลูกค้าได้ดีที่สุดหรือไม่?
- ความมั่นคง (Consistency)
  - มีข้อขัดแย้งเรื่องข้อกำหนดหรือไม่?
- ความสมบูรณ์ (Completeness)
  - ฟังก์ชั่นทั้งหมดที่ลูกค้าต้องการ?
- สัจนิยม (Realism)
  - ความต้องการสามารถดำเนินการได้ตามงบประมาณและเทคโนโลยีที่มีอยู่
- ตรวจสอบได้ (Verifiability)
  - สามารถตรวจสอบความต้องการได้หรือไม่?

Requirements validation techniques

- บทวิจารณ์ความต้องการ (Requirements reviews)
  - การวิเคราะห์ด้วยตนเองของระบบตาม Requirements
- การสร้างต้นแบบ (Prototyping)
  - ใช้แบบจำลองปฏิบัติการของระบบเพื่อตรวจสอบความต้องการ
- การสร้างกรณีทดสอบ (Test-case generation)
  - พัฒนา test-driven สำหรับ Requirements ในการตรวจสอบ

Requirements reviews

- ควรมีการทบทวนเป็นประจำในขณะที่มีการกำหนด requirement
- ทั้งลูกค้าและนักพัฒนาควรมีส่วนร่วมในการ review
- Review อาจเป็นทางการ (พร้อมเอกสารฉบับสมบูรณ์) หรือไม่เป็นทางการ
  - การสื่อสารที่ดีระหว่างนักพัฒนาลูกค้าและผู้ใช้สามารถแก้ปัญหาได้ในระยะเริ่มต้น

Review checks

- ตรวจสอบได้ (Verifiability)
  - ความต้องการที่สมจริงสามารถทดสอบได้หรือไม่?
- ความสามารถเข้าใจ (Comprehensibility)
  - เข้าใจข้อกำหนดถูกต้องหรือไม่?
- ตรวจสอบย้อนกลับ (Traceability)
  - ที่มาของ requirement ได้ถูกระบุไว้อย่างชัดเจนหรือไม่?
- การปรับตัว (Adaptability)
  - ความต้องการสามารถเปลี่ยนแปลงได้โดยไม่มีผลกระทบอย่างมากต่อ requirement อื่น ๆ หรือไม่?

Requirements change

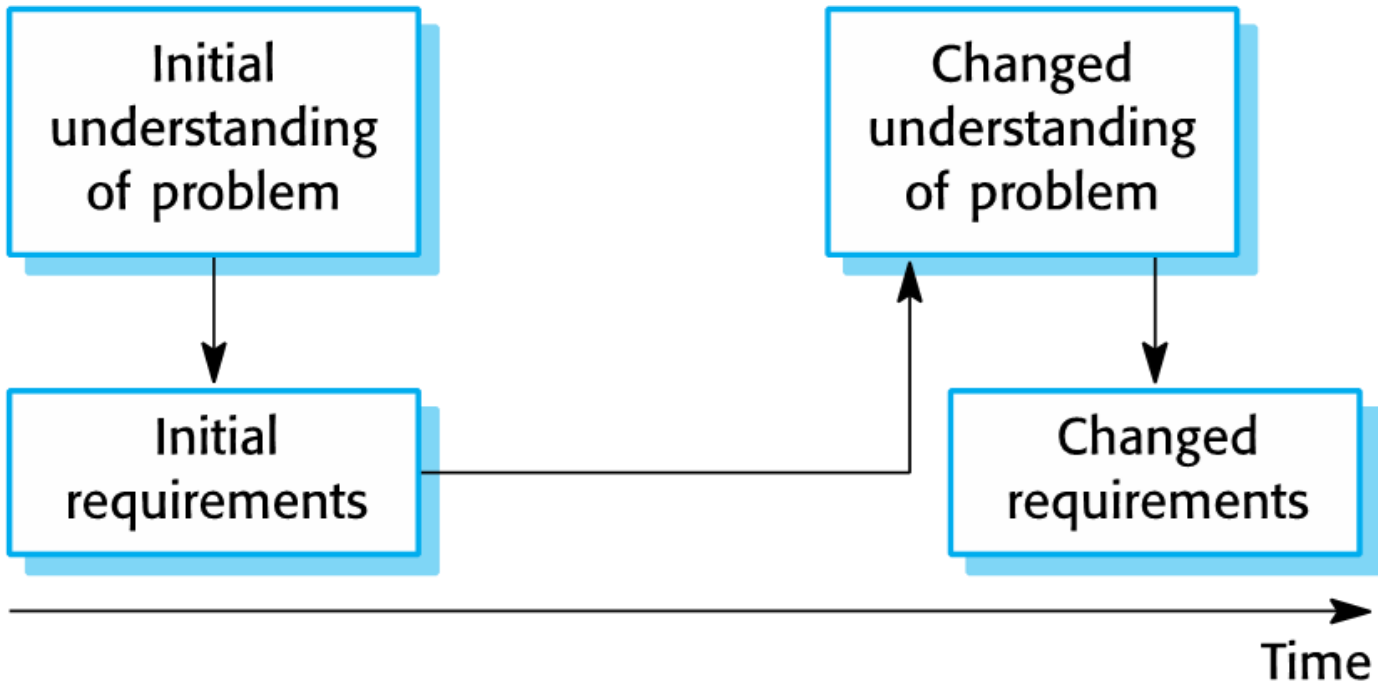
Changing requirements

- สภาพแวดล้อมทางธุรกิจและทางเทคนิคของระบบจะเปลี่ยนแปลงไปตลอดเวลาหลังจากติดตั้ง
  - อาจมีฮาร์ดแวร์ใหม่ ๆ ออกมาให้ใช้
  - อาจจำเป็นต้องเชื่อมต่อระบบกับระบบอื่น ๆ
  - ลำดับความสำคัญทางธุรกิจอาจมีการเปลี่ยนแปลง
  - อาจมีกฎหมายและข้อบังคับใหม่ ๆ ที่จำเป็นต้องปฏิบัติตาม
- คนที่จ่ายเงินสำหรับระบบและผู้ใช้ระบบนั้นแทบจะไม่เหมือนกัน
  - ลูกค้าระบบกำหนด requirement ตามข้อจำกัดขององค์กรและงบประมาณ ซึ่งอาจขัดแย้งกับความต้องการของ end-user
  - หลังจากส่งมอบแล้วอาจมีความต้องการเพิ่มคุณลักษณะใหม่ ๆ เพื่อให้บรรลุเป้าหมายของระบบ

Changing requirements

- ระบบขนาดใหญ่มักมีชุมชนผู้ใช้ที่มีความหลากหลาย
  - ผู้ใช้จำนวนมากมีความต้องการและลำดับความสำคัญที่แตกต่างกัน
    - อาจขัดแย้งหรือไปคนละทางกัน
  - ความต้องการของระบบอย่างสุดท้ายคือ การประนีประนอมระหว่างผู้ใช้ที่แตกต่าง

Requirements evolution



Requirements management

- Requirements management เป็นกระบวนการจัดการความต้องการที่เปลี่ยนแปลงไป
  - เกิดขึ้นระหว่าง requirements engineering process และ system development
- ความต้องการใหม่ ๆ มักจะเกิดขึ้นเสมอ ทั้งในขณะที่ระบบกำลังมีการพัฒนาและหลังจากที่เริ่มใช้งานแล้ว
- ต้องมีการติดตามความต้องการของแต่ละบุคคลและรักษาความเชื่อมโยงระหว่าง requirement
  - เพื่อให้สามารถประเมินผลกระทบจากการเปลี่ยนแปลง requirement ได้
  - จำเป็นต้องสร้างกระบวนการอย่างเป็นทางการ สำหรับการทำข้อเสนอการเปลี่ยนแปลง (change proposals) และเชื่อมโยงสิ่งเหล่านั้นเข้ากับ requirement ของระบบ

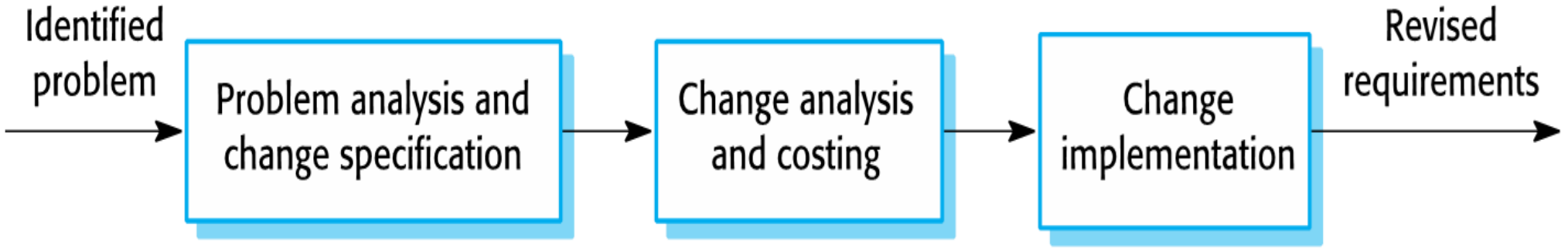
Requirements management planning

- กำหนดระดับของรายละเอียด requirements management ที่จำเป็น
- การตัดสินใจในการจัดการความต้องการ:
  - การระบุความต้องการ (Requirements identification) แต่ละ requirement ต้องมีการระบุเฉพาะ เพื่อให้สามารถอ้างอิงข้ามกับ requirement อื่น ๆ ได้
  - ขั้นตอนการจัดการการเปลี่ยนแปลง (A change management process) เป็นชุดของกิจกรรมที่ประเมินผลกระทบและค่าใช้จ่ายของการเปลี่ยนแปลง
  - นโยบายการตรวจสอบย้อนกลับ (Traceability policies) นโยบายเหล่านี้กำหนดความสัมพันธ์ระหว่าง requirement แต่ละข้อและระหว่าง requirement และการออกแบบระบบ ควรที่จะต้องบันทึกไว้
  - การสนับสนุนเครื่องมือ (Tool support) เครื่องมือที่สามารถใช้งานได้ อาจจะเป็นระบบการจัดการความต้องการเฉพาะ ไปจนถึงสเปรดชีต และระบบฐานข้อมูลแบบง่ายๆ

Requirements change management

- การตัดสินใจว่าควรเปลี่ยนแปลงข้อกำหนดหรือไม่
  - การวิเคราะห์ปัญหาและข้อกำหนดการเปลี่ยนแปลง
    - ในระหว่างขั้นตอนนี้ จะมีการวิเคราะห์ปัญหาหรือข้อเสนอการเปลี่ยนแปลง เพื่อตรวจสอบว่าถูกต้องหรือไม่
    - ถ้าไม่ถูกต้อง จะส่งผลการวิเคราะห์กลับไปยังผู้ร้องขอ ซึ่งอาจตอบสนองโดยการเปลี่ยนแปลงข้อกำหนดที่เฉพาะเจาะจงมากขึ้นหรือตัดสินใจถอนคำขอ
  - วิเคราะห์การเปลี่ยนแปลงและต้นทุน
    - ผลของการเปลี่ยนแปลงที่เสนอจะได้รับการประเมินโดยใช้ข้อมูลการตรวจสอบย้อนกลับและความรู้ทั่วไปเกี่ยวกับข้อกำหนดของระบบ
    - เมื่อการวิเคราะห์เสร็จสิ้นแล้วจะมีการตัดสินใจว่าจะดำเนินการเปลี่ยนแปลงข้อกำหนดหรือไม่
  - Implement การเปลี่ยนแปลงข้อกำหนดนั้น
    - ดำเนินการปรับเปลี่ยนเอกสารข้อกำหนด
    - ควรมีการจัดระเบียบเอกสารเพื่อให้การเปลี่ยนแปลงสามารถทำได้อย่างง่ายดาย

Requirements change management



Key points

- Requirement สำหรับระบบซอฟต์แวร์ กำหนดสิ่งที่ระบบควรทำและกำหนดข้อจำกัดในการปฏิบัติการและการดำเนินงาน
- ความต้องการของระบบคือคำอธิบายของบริการที่ระบบต้องระบุหรืออธิบายถึงวิธีการคำนวณบางอย่างที่ต้องทำ
- Non-functional requirements มักเป็นตัวกำหนดหรือจำกัดการพัฒนาระบบและกระบวนการพัฒนาที่ใช้อยู่
- Non-functional requirements มักจะเกี่ยวข้องกับคุณสมบัติที่เกี่ยวกับความปลอดภัยของระบบและถูกนำไปใช้กำหนดระบบโดยรวม

Key points

- กระบวนการ requirements engineering เป็น iterative process ประกอบด้วย
  - requirements elicitation
  - requirements specification
  - requirements validation
- requirements elicitation เป็นกระบวนการซ้ำซ้อนที่สามารถแสดงเป็น spiral ของกิจกรรม ประกอบด้วย
  - requirements discovery
  - requirements classification and organization
  - requirements negotiation
  - requirements documentation

Key points

- Requirements specification คือกระบวนการของการจัดทำเอกสาร user requirements และ system requirements อย่างเป็นทางการ สิ่งที่ได้คือ เอกสารข้อกำหนดซอฟต์แวร์ (software requirements document)
- เอกสารข้อกำหนดซอฟต์แวร์ (software requirements document) เป็นคำอธิบายที่ระบุไว้ใน system requirements ควรจัดให้ทั้งลูกค้าระบบและนักพัฒนาซอฟต์แวร์สามารถใช้งานได้

Key points

- Requirements validation คือกระบวนการตรวจสอบข้อกำหนดสำหรับ ประกอบด้วย
  - ความถูกต้อง validity
  - ความสอดคล้อง consistency
  - ความสมบูรณ์ completeness
  - ความสมจริง realism
  - การตรวจสอบได้ verifiability
- การเปลี่ยนแปลงทางธุรกิจและทางเทคนิคย่อมนำไปสู่การเปลี่ยนแปลงความต้องการสำหรับระบบซอฟต์แวร์
  - การจัดการความต้องการ (requirements management) เป็นกระบวนการในการจัดการและควบคุมการเปลี่ยนแปลงเหล่านี้

คำถาม???