Requirement determination

as is system = existing system = ระบบที่มีอยู่แล้ว

to be system = proposed system = ระบบใหม่

system proposal – refine general idea ใน system request ให้มี detail มากขึ้น 🡪 ถ้า approve โดย committee ค่อย design

* requirement definition
* analysis model
  + functional
  + structural
  + behavioral

requirement determination

* convert high level business requirement 🡪 detailed requirement
* requirement ระบุว่าระบบทำอะไรได้
* เขียนโดย system analyst กับ business ร่วมกัน
* ประเภท requirement
  + FR = functional requirement - กระบวนการ ว่าทำงานอย่างไร, มี data อะไร, ทำอะไรได้
    - feature – นิยมใช้
    - system mode : control, emergency
    - user class : elevator
    - object
    - stimulus
    - response
  + NFR = non-functional requirement – บอกคุณภาพ, behavior property – derive goal เป็น sub goal
    - operational – physical : ทำงานบนมือถือ, ทำงานร่วมกับระบบเก่า
    - performance – speed, capacity
    - security : ประเภท user
    - reliability - ถูกต้อง ตรงกับข้อมูลจริง
    - maintainability - ต้องแก้ไข อัพเดตได้
    - usability - ใช้ง่าย
    - cultural & political : รับได้ 2 สกุลเงิน, สอดคล้องกับกฎหมายในอุตสาหกรรมนั้น

requirement definition

* text report in outline format
* define scope ของ system
* อาจมี prioritized – rank high medium low
* ให้ information needed by analysis model & subsequent workflow

requirement construct

* shall = requirement ที่สำคัญมาห ต้องทำ มีผลผูกพัน (binding)

should = preference ที่อยากได้

may = suggestion

* ใช้ is, are, was ใน descriptive text – หลีกเลี่ยง must
* ใช้ positive statement - ไม่ใช้ not
* ใช้ active voice - ไม่ใช้ passive

requirement syntax

* condition + subject + action + object t + constraint

เมื่อเกิด condition จะให้ subject ทำ action กับ object โดยมีเงื่อนไขตาม constraint

* condition + action / constraint + value

เมื่อเกิด condition จะมี action / constraint ตามค่า value

* subject + action + value

ลักษณะของแต่ละ requirement

* necessary - เขียนเฉพาะที่จำเป็น ถ้าทำไม่ได้ก็ไม่ต้องใส่
* implementation free - บอกว่าต้องการอะไร ไม่ต้องบอกว่าทำอย่างไร
* unambiguous - ไม่กำกวม
* consistent - ไม่ขัดกับ requirement อื่น
* complete - ต้องไม่ต้องตีความเพิ่ม
* singular – 1 bullet ต่อ 1 requirement
* feasible - อย่าเขียนสิ่งที่ทำไม่ได้
* traceable - ต้อง track ได้
* verifiable - ต้อง proof ได้ว่าทำเสร็จแล้ว

ลักษณะของ set of requirement

* complete - ไม่มีข้อไหนตกหล่น
* consistent – ไม่มีข้อไหนขัดกัน
* affordable
* bounded - อย่าให้ตีความเพิ่มได้

create requirement definition

* กำหนด type ของ FR กับ NFR
  + analyst ใช้ requirement analysis strategy ในการ guide user ให้ discover & explain need
  + analyst ใช้เทคนิค requirement-gathering ในการเก็บ detail
* analyst work with user เพื่อแก้ requirement เรื่อยๆ
* ช่วง analysis workflow
  + evolve requirement definition
  + ระวัง scope creep – scope ไม่ชัดแล้วขยายเรื่อยๆ

requirement analysis approach

* business process automation - มีระบบอยู่แล้ว แต่ตอนแรกใช้คนทำ
* business process improvement - ปรับปรุงกระบวนการทำงาน
* business process reengineering – รื้อทั้งระบบ

requirement analysis strategy – ใช้หลายอย่างร่วมกันได้

1. problem analysis - ถามปัญหาของ as is system 🡪 หา solution
2. root cause analysis - หาสาเหตุ 🡪 แก้ที่สาเหตุ – 1 ปัญหาอาจมีหลาย cause, 1 cause อาจมีหลายsolution
3. duration analysis - วิเคราะห์ระยะเวลาการทำงาน – อันไหน parallel, integrate ได้
4. activity-based costing – activity ไหนใช้ cost เท่าไร
5. informal benchmarking - หาตัวเทียบ
6. outcome analysis - ดู outcome ว่าลูกค้าได้สิ่งที่ต้องการไหม
7. technology analysis - หา technology ใหม่ๆที่เหมาะสม
8. activity elimination - ตัด activity ที่ไม่จำเป็น

เทคนิค requirement-gathering – uncover all requirement & build support and trust ใน user

* interview – นิยมสุด

1. เลือกคนสัมภาษณ์ - สำคัญสุด - ถาม key stakeholder ทุก level
2. คิดคำถาม - มีทั้ง open-ended, close-ended, probing question (why, example, explain)
3. เตรียมสัมภาษณ์ - นัดล่วงหน้า ให้คำถามก่อน บอกเหตุผลว่าสัมภาษณ์ทำไม
4. สัมภาษณ์
   * top-down - ถามจากภาพรวม

bottom-up - ถามจากเรื่องเล็กๆ

* + ไม่ bias, ไม่ชี้นำ, record, สรุปเรื่อยๆเพื่อ recheck, แยกความเห็นกับข้อเท็จจริง, ให้ถามกลับ, explain ว่าจะทำอะไร แต่อย่า promise

1. follow-up หลังสัมภาษณ์
   * เขียน report ภายใน 48 ชม. ให้ verify ความถูกต้อง
   * open item = สิ่งที่จะให้คนถูกสัมภาษณ์ส่งมาเพิ่มเติม

* JAD = joint application development – จัดประชุมระหว่าง user, analyst, technical lead
  + มี facilitator = คนดำเนินการ เป็นคนกลางที่ไม่เป็น participant
  + ปัญหา – facilitator ต้องจัดการ
    - domination - คนเด่นเกิน
    - noncontributor - คนหลบมุม
    - side discussion – คุยกันเอง
    - วนเรื่องเดิมที่พูดไปแล้ว
    - ข้อถกเถียง 🡪 compromise, เลื่อนไปก่อน
    - ไม่เล่นมุข แต่อย่าตึงเกิน
* questionnaire – แบบสอบถาม 🡪 กระจายคนได้เยอะ แต่ response rate ต่ำ (paper < 50%, web < 30%)
* document analysis - ขอเอกสารมาดูเอง 🡪 ข้อเสียคือได้แต่ as is system
* observation

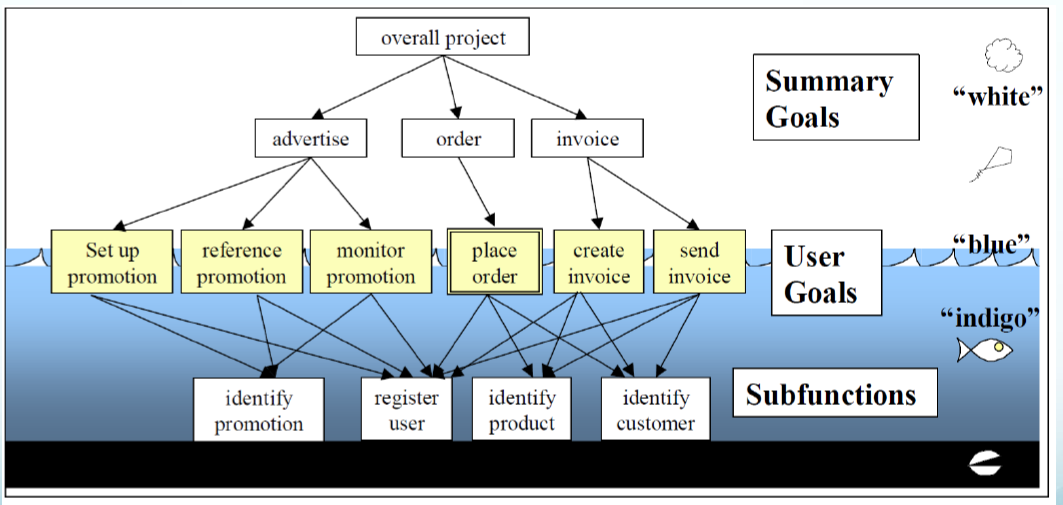
เทคนิค requirement documentation

* concept map
* user story
  + story card
  + product backlog - เอา story card มารวมเป็นตาราง

Business process and functional modeling

use case diagram - ให้ bird’s eye view - ไม่มี detail

* actor
* interaction
* ขั้นตอนสร้าง use-case diagram
  + review requirement definition
  + หา subject boundary, primary actor, major use-case
  + review current use-case
* use case level
  + summary goal – ภาพรวม, แต่ละ department ต้องการทำอะไร
  + user goal – more detailed, user ทำอะไรบ้าง
  + subfunction – ขั้นตอนย่อย - ใส่ใน use-case description กับ activity diagram



|  |  |
| --- | --- |
| Actor   * primary business - คนที่ได้ benefit * primary system - คนที่ interact กับ interface * external server - คนที่ให้ input * external receiver - คนที่รับ output ไป     คน ไม่ใช่คน | * role - ไม่ใช่ specific user - ให้ input / รับ output จาก system * ใช้ specialization/superclass association ได้ * อยู่นอก subject boundary |
| Use case | * represent system functionality * extend / include ได้ * ใช้ verb-noun phrase * อยู่ใน subject boundary |
| Subject boundary | * ชื่อ project อยู่ที่ top * represent scope |
| Association relationship | * link ระหว่าง actor กับ use case ที่ interact ด้วย * \* = multiplicity = มีได้หลายอัน * ไม่มีลูกศร = information flow two-way |
| Include relationship | * ลูกศรชี้จาก used use case 🡨 base use case * ถ้าทำ base use case 🡪 ต้องทำทุก include * reuse used use case ได้ |
| Extend relationship | * ลูกศรชี้จาก extension use case 🡪 base use case * extension เป็น exceptional case |
| Generalization relationship | * ลูกศรชี้จาก specialized 🡪 generalized * represent specialized actor / use case to อันที่ generalized |
| Package | * ระบบย่อย |
| Comment |  |

use-case description

* ข้อมูล
  + input
  + validate
  + process
  + output
* ประเภท use case
  + real - บอก technology, design commitment, user interface ของ real system

essential - บอก implementation-independent issue ที่จำเป็น

* + overview – high level overview ของ requirement – overview กับ relationship

detail - บอกทุก part – บอก flow of event ตาม activity diagram ด้วย

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name:** verb-noun phrase (ชื่อในวงรี) | | **ID:** | **Importance Level:** High/Medium/Low |
| **Primary Actor:** actor ที่เป็นคนทำ | **Use Case Type:** Detail/Overview, Essential/Real | | |
| **Stakeholders and Interests:** actor + คนที่อาจใช้ข้อมูลนี้ในอนาคต  ใคร - ต้องการทำอะไร | | | |
| **Brief Description:** This use case support <actor> to | | | |
| **Trigger:** เริ่มต้นการทำงานทำงานเมื่อไร  **Type:** External (ทำงานเมื่อมีคนสั่ง) / Temporal (ทำงานเมื่อถึงเวลา) | | | |
| **Relationships:**  **Association:** actor ที่ interact ด้วย  **Include:**  **Extend:**  **Generalization:** | | | |
| **Precondition:** | | | |
| **Postcondition:** | | | |
| **Normal Flow of Events:** ขั้นตอนปกติ - ลำดับ 1,2, ... - Keep It Simple Stupid   1. If …, the S-1: <name> is performed | | | |
| **SubFlows:** ขั้นตอนย่อย - อาจเป็น include use case  S-1: <name> ลำดับ 1, 2, ... | | | |
| **Alternate/Exceptional Flows:** อาจเป็น extend use case  4a1: ... | | | |

activity diagram - อธิบายรายละเอียด use case แบบ graphical view

* ขั้นตอนสร้าง activity diagram
  + เลือก business process – review requirement definition, use-case diagram, documentation
  + กำหนด set of activity 🡪 connect ด้วย control flow, object flow, node
  + กำหนด decision, parallel
  + วาด diagram – เส้นไม่ตัดกัน, ซ้าย🡪ขวา / บน🡪ล่าง

|  |  |
| --- | --- |
| Action / Activity | * verb หรือ noun * short แต่ understandable   Action   * decomposable ไม่ได้   Activity   * represent set of action * 1 activity : 1 use case |
| Object node | * ใช้เมื่อมีการส่งข้อมูลในรูป object ระหว่าง action / activity |
| Comment |  |
| Control flow | * แสดงลำดับการ execute |
| Object flow | * เชื่อม object / node * แสดง flow ของ object / information |
| Initial node | * เริ่มต้น * อยู่บน/บนซ้าย |
| Final-activity node | * สิ้นสุด * อยู่ล่าง/ล่างขวา |
| Final-flow node | * จบ flow * path อื่นที่ parallel อยู่ยังทำงานต่อได้ |
| Decision node | * เงื่อนไข |
| Merge node | * merge 2 เงื่อนไข |
| Fork node | * parallel * ทำทั้งคู่ |
| Join node | * merge parallel flow |
| Swimlane | * ได้ทั้งแนวตั้ง/แนวนอน |

verify and validate business process & functional model

* verification – check ว่า
  + ตรงกับ spec – requirement, model
  + สร้างถูกต้อง
* validation – check กับลูกค้าว่าตรงกับ purpose
* walkthrough - ไล่เช็ค doc กับลูกค้า
  + purpose
    - test fidelity ของ functional model กับ functional requirement
    - ensure ว่า model consistent
  + โดย development team กับ client – interactive
    - facilitator - อำนวยความสะดวกการประชุม นัดคน
    - presenter - คนนำเสนอ
    - recorder – บันทึกการประชุม
* rule
  + เทียบแต่ละ step ของ use-case description กับ activity diagram
  + เทียบ object ของ use-case description กับ activity diagram
  + เทียบ sequence ของ use-case description กับ activity diagram
  + ใน 1 use-case (1 วงรี) มี 1 ตาราง use-case description
  + actor ใน use-case description ต้องอยู่ใน use-case diagram
  + stakeholder
    - ถ้ามีใน use-case diagram 🡪 ต้องมีใน use-case description
    - ถ้ามี use-case description 🡪 อาจมี/ไม่มีใน use-case diagram ก็ได้
  + relationship ใน use-case description (include, extend, generalization) ต้องมีใน use-case diagram
  + ถูกต้องตาม diagram-specific rule