

“แนวข้อสอบ/สรุปขอบเขต” สอบปลายภาค 1/2568

รายวิชา EN-012-315 | วิศวกรรมซอฟต์แวร์ สำหรับนักศึกษา กลุ่ม CE 6541

ให้นักศึกษาใช้เตรียมตัว ครอบคลุมสองหัวข้อใหญ่ ทั้งแบบปรนัยและอัตนัย รวม 40 คะแนน

- Requirements Engineering (20 คะแนน) สไลด์สอน: <https://shorturl.at/s8LgO>
- System Modeling (20คะแนน) สไลด์สอน: <https://shorturl.at/pJ1dl>
- อนุญาตให้นำกระดาษโน้ตเขียนด้วยลายมือขนาด A4 เข้าห้องสอบได้ 2 แผ่น และให้แนบส่งพร้อมข้อสอบ

1) Requirements Engineering — สิ่งที่ต้องแม่น

A. ภาพรวม RE & 4 ขั้นหลัก

- Elicitation → Analysis → Specification (SRS) → Validation (จำชื่อ+หน้าที่ของแต่ละขั้น)
- เครื่องมือเก็บความต้องการ: สัมภาษณ์, เวิร์กช็อป, แบบสอบถาม, สังเกตการณ์, ศึกษาระบบเดิม, Prototyping

B. ประเภทความต้องการ

- Functional vs Non-Functional (NFR)
- การจัดหมวด NFR: Product / Organizational / External (พร้อมตัวอย่าง)
- แนวคิด Enduring vs Volatile requirements

C. SRS (Software Requirements Specification)

- บทบาท SRS ต่อการประเมินต้นทุน/ความเสี่ยง/แผน
- โครงสร้างหัวข้อหลัก (Intro, General Description, Specific Requirements, ฯลฯ)

D. Validation & คุณภาพของความต้องการ

- เกณฑ์: Correctness, Completeness, Consistency, Clarity, Realism, Verifiability, Traceability
- เทคนิค: Review (informal/formal), Prototyping, สร้าง test cases จาก requirement

E. Metrics สำหรับ NFR

- ทำให้ “วัดได้/ตรวจสอบได้” เช่น Response time $\leq 2s$ @ 500 users, Uptime $\geq 99.9\%$ ฯลฯ

F. Requirement Management

- Identification/Versioning, **Change Management** (CR, CCB, impact analysis), **Traceability** (req ↔ design ↔ test), การจัดลำดับความสำคัญ (MoSCoW/Scoring)

ข้อควรระวังในการสอบ (มักผิด):

- เขียน NFR แบบวัดไม่ได้ (“เร็ว/ดี/ปลอดภัย” โดยไม่กำหนดเกณฑ์), สับสน Product/Organizational/External, ลืม Pre/Postcondition ใน Use Case Description, มองข้าม Traceability

2) System Modeling — สิ่งที่ต้องทำให้ได้

A. ภาพรวม/บทบาทของแบบจำลอง

- นิยาม & วัตถุประสงค์ของ System Model (สื่อสาร มองภาพรวม ลดความคลุมเครือ)
- เปรียบเทียบแนว **Structured** (DFD/ERD) กับ **OO/UML** (Use Case, Class, Sequence, Activity, Deployment)

B. DFD (Data Flow Diagram)

- **Context (Level 0):** มี 1 process แทนระบบ, แสดง External Entities และ I/O, *ไม่ใช่ Data Store*
- **Level 1+ :** แยกย่อย process, รักษาความสอดคล้องของ data flow (balancing)
- แบบฝึก: เขียน Context ของระบบพื้นฐาน (เช่น “ผลการเรียนรายวิชา”) แล้วแตกเป็น Level 1

C. ERD (Entity–Relationship Diagram)

- องค์ประกอบ: Entity, Attribute, Relationship, **Cardinality** (1:1, 1:M, M:N), Optionality
- แบบฝึก: จากโจทย์สั้น ๆ ระบุตารางหลัก/คีย์/ความสัมพันธ์

D. UML Use Case

- องค์ประกอบใน **Use Case Description** (Actor, Trigger, Pre/Post, Main/Alt flows)
- ความต่าง <<include>> (แทรกเสมอ) vs <<extend>> (มีเงื่อนไข)
- แบบฝึก: เขียนชื่อ use cases + อธิบาย flow ย่อ 1 เรื่อง

E. UML Class Diagram

- โครงสร้าง: Class–Attribute–Operation และ **visibility** (+ public, # protected, - private)

- ความสัมพันธ์: Association/Dependency/Generalization/Aggregation/Composition (จำภาพและความต่างให้ชัด)
- แบบฝึก: แปลง ERD เป็น Class Diagram แบบคร่าว ๆ

F. Sequence Diagram

- โฟกัส ลำดับข้อความตามเวลา, ผู้มีบทบาท (lifelines), เงื่อนไข/ทางเลือก
- แบบฝึก: กรณี “ถอนเงิน ATM” หรือ “สั่งซื้อสินค้า”

G. Activity Diagram (+ Swimlanes)

- โหนดสำคัญ: Initial/Final, Action, Decision/Merge, Fork/Join, Swimlanes (แบ่งความรับผิดชอบ)
- แบบฝึก: วาด flow ที่มีงานขนานและข้ามเลนอย่างน้อย 1 จุด

H. Deployment Diagram

- โหนด/อุปกรณ์/องค์ประกอบที่ติดตั้ง และการเชื่อมต่อกัน

ข้อควรระวังในการสอบ (มักผิด):

- ใส่ Data Store ใน Context, สลับ include กับ extend, วาด Composition/ Aggregation สลับกัน, Class visibility ผิด, ลำดับข้อความใน Sequence ไม่เดินตามเวลา

3) รูปแบบคำถามที่พบบ่อย (ช่วยโฟกัสอ่าน)

ปรนัย (เลือกตอบ)

- ระบุบทบาทสัญลักษณ์/ความสัมพันธ์ (DFD/ERD/UML)
- แยกประเภทความต้องการ (FR vs NFR) และจำแนก NFR เป็น 3 หมวด
- เกณฑ์ Validation และความหมาย Traceability/Change mgmt
- ความต่าง include vs extend, Aggregation vs Composition, Visibility ใน Class

อัตนัย (เขียนตอบ)

- อธิบาย 4 ชั้นของ RE พร้อมกิจกรรม
- ยกตัวอย่าง NFR ที่ “ตรวจสอบได้” (มี metrics)
- วาด Context → Level 1 (DFD) จากโจทย์สั้น, หรือ Use Case Description ที่ครบองค์ประกอบ

- ออกแบบ Activity Diagram พร้อม **Swimlanes** และอธิบายงานขนาน
- แสดงเส้นทาง **Traceability** จาก requirement → design → test

4) เช็กलिस्टก่อนสอบ (ทำให้ครบ)

- จำคำสำคัญ: include/extend, visibility (+/#/-), cardinality, Fork/Join, Swimlanes
- ฝึกวาดอย่างน้อยอย่างละ 1: Context+Level1, ERD เล็ก ๆ, Use Case (+Description สั้น), Class, Sequence, Activity
- เขียนตัวอย่าง **NFR แบบวัดได้** อย่างน้อย 5 ข้อ
(performance/security/reliability/usability/portability)
- ร่างตาราง **จัดลำดับความสำคัญ** (เช่น MoSCoW) สำหรับเคสสมมติ 6–8 ข้อ
- ฝึกสรุป **เกณฑ์ Validation** เป็น bullet 1 หน้า

5) ชุด “ฝึกเร็ว” 10 ข้อ (ทำเองให้คล่อง)

1. นิยาม RE + 4 ชั้น (ตอบสั้น)
2. แยก 6 ข้อกำหนดเป็น FR/NFR
3. เขียน NFR ด้าน performance ให้ “วัดได้” 3 แบบ
4. Context Diagram ระบบชำระเงินอย่างง่าย (สเก็ทซ์ 3 นาที)
5. Use Case “คืนสินค้า” พร้อม Pre/Post/Normal/Alt flows (ย่อ)
6. ระบุ include/extend ที่เหมาะสมในกรณี “ชำระเงิน–ยืนยันตัวตน–ใช้คุกกี้”
7. Class Diagram ย่อ: Order, Customer, Payment (ใส่ความสัมพันธ์และ visibility)
8. Sequence “ล็อกอินไม่ผ่าน” (อย่างย่อ)
9. Activity พร้อม Swimlanes: “อนุมัติคำร้องออนไลน์” (มี Fork/Join 1 จุด)
10. วาดสาย Traceability ของ **REQ-SEC-001** ไปยัง design/test

ถ้านักศึกษาทำเช็กलिस्टและ “ชุดฝึกเร็ว” ได้ครบและเข้าใจเหตุผลเบื้องหลัง จะพร้อมสำหรับทั้งข้อปรนัยและอัตนัย