ภาคผนวก ช

ส่วนประกอบ
ด้านการออกแบบเชิงวัตถุ

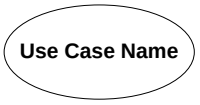
บทที่ 3
การวิเคราะห์และออกแบบระบบงาน

3.1 Use Case Diagram

Use Case Diagram เป็นไดอะแกรมที่ช่วยให้ผู้พัฒนาทราบถึงความสามารถของระบบว่าต้องทำอะไรได้บ้าง ทราบถึงผู้ใช้งานในแต่ละส่วนของระบบและเกิดความง่ายในการสื่อสารระหว่างผู้พัฒนากับผู้ใช้ระบบ

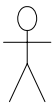
3.1.1 สัญลักษณ์ที่ใช้ใน Use Case Diagram

3.1.1.1 ยูสเคส (Use Case) คือความสามารถหรือฟังก์ชันของระบบซอฟต์แวร์ที่จะพัฒนา โดยการเขียน Use Case ใช้วงรีและคำอธิบายฟังก์ชันการทำงานอยู่ในวงรีนั้น



ภาพที่ 3.1 แสดงสัญลักษณ์ Use Case

3.1.1.2 แอ็กเตอร์ (Actor) คือ ผู้ที่กระทำกับระบบ หรือผู้ที่เกี่ยวข้อง โดยจะเป็นคนหรือไม่ก็ได้ ซึ่งเป็นผู้แลกเปลี่ยนข้อมูลข่าวสารกับระบบที่จะทำการพัฒนา โดยเราจะใช้สัญลักษณ์รูปคนแทนสัญลักษณ์ของ Actor นั้น



Actor Name

ภาพที่ 3.2 แสดงสัญลักษณ์ Actor

3.1.1.3 เส้นแสดงความสัมพันธ์ (Relationship) คือ เส้นเพื่อแสดงความสัมพันธ์ระหว่าง Actor กับ Actor หรือ Use Case กับ Use Case

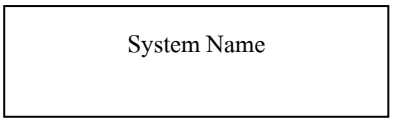
Connection ระหว่าง Actor กับ Use Case Connection ระหว่าง Use Case กับ Use Case

ภาพที่ 3.3 แสดงสัญลักษณ์เส้นแสดงความสัมพันธ์

1) ความสัมพันธ์แบบขยาย (Extend Relationship) ใช้เพื่อบอกว่ายูสเคสหนึ่ง ถูกช่วยเหลือโดยการทำงานยูสเคสอื่น โดยจะใช้ <<extend>> เป็นเครื่องหมายอ้างอิง

2) ความสัมพันธ์แบบรวม (Include Relationship) ใช้เพื่อบอกว่ายูสเคสหนึ่งถูกอาศัยการทำงานของยูสเคสอื่น ๆ โดยจะใช้ <<include>> เป็นเครื่องหมายอ้างอิง

3.1.1.4 ขอบเขต (System Boundary คือ เส้นแบ่งขอบเขตระหว่างระบบกับผู้กระทำต่อระบบ ใช้สี่เหลี่ยมเป็นสัญลักษณ์



ภาพที่ 3.4 แสดงสัญลักษณ์ขอบเขต

3.1.2 การเขียนคำอธิบาย Use Case

เนื่องจากแต่ละ Use Case ประกอบไปด้วยการทำงานหลาย ๆ อย่าง ดังนั้นเพื่อให้ทราบรายละเอียดปลีกย่อยของขั้นตอนการทำงานในแต่ละ Use Case จึงต้องมีการเขียนคำอธิบายสำหรับ Use Case ควบคู่กันไปเราเรียกคำอธิบาย Use Case ดังกล่าวว่า กระแสของเหตุการณ์ (Flow of Event)

ในการเขียนคำบรรยายกระแสของเหตุการณ์ในปัจจุบันนั้นยังมีรูปแบบที่แตกต่างกันออกไป แต่โดยส่วนใหญ่คำบรรยายกระแสเหตุการณ์นี้จะประกอบไปด้วยสองส่วนสำคัญได้แก่ เหตุการณ์หลัก (Main Flow) และ เหตุการณ์พิเศษ (Exceptional Flow) โดยเหตุการณ์ทั้งสองส่วนต้องระบุถึงสาเหตุของการเริ่มต้นและสิ้นสุดกิจกรรมด้วยเสมอ

3.1.2.1 เหตุการณ์หลัก(Main Flow) คือลำดับกิจกรรมเมือ Use Case ดำเนินกิจกรรมตามปกติ โดยการเขียนคำอธิบายในลักษณะเป็นย่อหน้าและเหตุการณ์หลักจะต้องมีเพียงหนึ่งเดียว

3.1.2.2 เหตุการณ์พิเศษ(Exceptional Flow) คือลำดับกิจกรรมเมื่อ Use Case ดำเนินกิจกรรมผิดจากปกติโดยสามารถมีได้มากกว่า 1 เหตุการณ์

3.1.3 ข้อแนะนำในการการเขียน Use Case

3.1.3.1 ใช้เพื่อเพื่อแสดงข้อมูลความต้องการของผู้ใช้ระบบเท่านั้น หรืออาจกล่าวได้ว่าเราใช้ Use Case Diagram เพื่อเก็บรวบรวมข้อมูลความต้องการจากผู้ใช้ระบบเท่านั้น ดังนั้นเป็นไปได้ว่าในการเก็บข้อมูลในรอบแรกอาจจะยังไม่ครอบคลุมความต้องการของผู้ใช้ จึงสามารถนำกลับมาปรับปรุงแก้ไขเพิ่มเติมได้จนกว่าจะได้ข้อมูลครบถ้วน

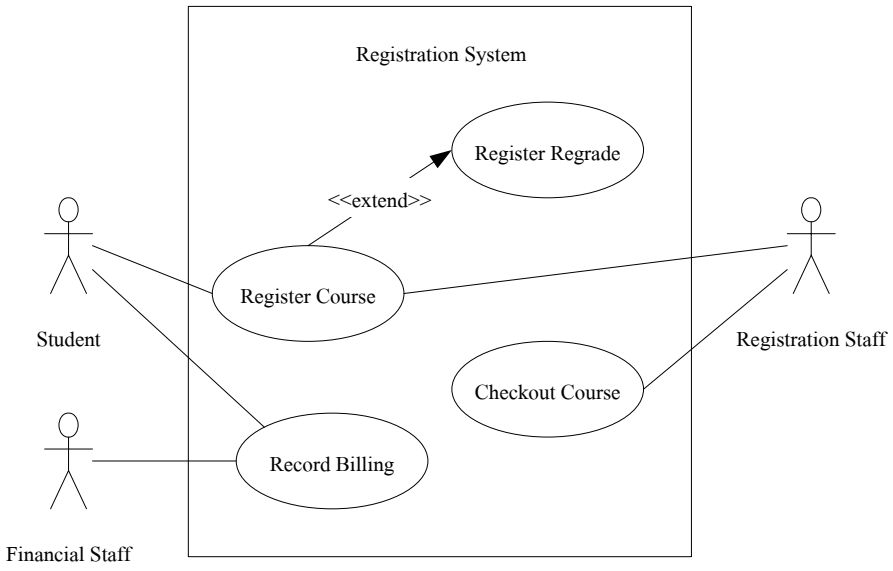
3.1.3.2 Use Case Diagram ใช้เพื่อสื่อสารระหว่างนักวิเคราะห์ระบบกับผู้ใช้ ไม่ได้ใช้สื่อสารระหว่างนักวิเคราะห์ระบบกับโปรแกรมเมอร์ ดังนั้น Use Case Diagram จะแสดงเพียงหน้าที่ที่ระบบต้องทำตามความต้องการของผู้ใช้ ทำให้มองเห็นภาพกว้าง ๆ ของระบบมากกว่าภาพเชิงลึก โดยส่วนใหญ่ Use Case Diagram จะไม่แสดงให้เห็นถึงระดับการจัดการข้อมูลในฐานข้อมูลเช่น เพิ่ม ลบ แก้ไข หรือปรับปรุงข้อมูลเป็นต้น

3.1.3.3 Use Case Diagram อาจจะมีรายละเอียดมากหรือน้อยก็ได้ ขึ้นอยู่กับมุมมอง เทคนิค และประสบการณ์ของทีมงานหรือนักวิเคราะห์ระบบจึงไม่มีข้อสรุปได้ว่าลักษณะของ Use Case Diagram แบบใดถูกหรือผิด

3.1.3.4 Actor ทุกตัวต้องมีปฏิสัมพันธ์กับ Use Case อย่างน้อยหนึ่งตัว และ Use Case ทุกตัวต้องมีปฏิสัมพันธ์อย่างใดอย่างหนึ่งกับ Actor หรือ Use Case ตัวอื่น ๆ เสมอ

ตัวอย่าง

ตัวอย่างต่อไปนี้เป็นระบบลงทะเบียนเรียน (Registration System) ประกอบด้วย 3 Actor ได้แก่ นักศึกษา(Student) เจ้าหน้าที่การเงิน(Financial Staff) และ เจ้าหน้าที่ฝ่ายทะเบียน (Registration Staff) และประกอบด้วย Use Case ซึ่งสามารถทำรายการลงทะเบียน (Register Course) ตรวจสอบวิชาที่ลงทะเบียน (Checkout Course) และบันทึกรายการชำระเงินค่าลงทะเบียน (Record Billing) นอกจากนี้อาจจะมีเหตุการณ์หรือเงื่อนไขพิเศษเช่น มีนักศึกษาลงทะเบียนรายวิชาซ้ำเพื่อทำการปรับเกรด(Register Regrade)ด้วยจึงเพิ่ม Extending Use Case เพื่อรองรับหน้าที่ดังกล่าว



ภาพที่ 3.5 ตัวอย่าง Use Case ระบบลงทะเบียน

คำอธิบาย Use case

Use Case Title : Register Course
Main Flow : ระบบลงทะเบียนมีกลุ่มบุคคลที่เกี่ยวข้อง 2 กลุ่มได้แก่ นักศึกษา และพนักงานของมหาวิทยาลัย (เจ้าหน้าที่ฝ่ายลงทะเบียนและเจ้าหน้าที่ฝ่ายการเงิน) ในแต่ละเทอมจะต้องมีนักศึกษามาลงทะเบียนเรียนของภาคเรียนปกติโดยนักศึกษาจะต้องกรอกรายวิชาเรียนในแบบฟอร์มลงทะเบียนไม่เกิน 21 หน่วยกิต แล้วนำไปยื่นกับเจ้าหน้าที่เพื่อทำการตรวจสอบวิชาที่นักศึกษาได้ลงทะเบียนในแบบฟอร์มกับประวัติการลงทะเบียนว่าถูกต้องหรือไม่ เนื่องจากบางรายวิชาของแต่ละเทอมมีเงื่อนไขว่าจะลงทะเบียนรายวิชาได้ก็ต่อเมื่อสอบผ่านรายวิชาหนึ่งมาก่อน เมื่อตรวจสอบว่าถูกต้องแล้ว เจ้าหน้าที่ฝ่ายทะเบียนจะคำนวณเงินค่าลงทะเบียนเรียน จากนั้นบันทึกลงในฐานข้อมูล ระบบแสดงข้อความบันทึกข้อมูลเรียบร้อยแล้ว สั่งพิมพ์ใบรับลงทะเบียนโดยแบ่งออกเป็น 2 ส่วน ส่วนที่ 1 นักศึกษาเก็บไว้เอง ส่วนที่ 2 นำไปชำระเงินโดยโอนฝ่ายทางธนาคาร
Exceptional Flow ที่ 1 : กรณีที่เจ้าหน้าที่ฝ่ายทะเบียนป้อนรหัสนักศึกษาผิดพลาด ระบบจะแสดงข้อความแจ้งเตือนว่า “ไม่พบข้อมูลนักศึกษา” เพื่อแสดงให้เจ้าหน้าที่ทราบว่ามีข้อผิดพลาดเกิดขึ้น และให้เจ้าหน้าที่ป้อนรหัสนักศึกษาใหม่อีกครั้ง
Exceptional Flow ที่ 2 : กรณีที่เจ้าหน้าที่ป้อนข้อมูลสำคัญไม่ครบถ้วน ระบบจะไม่สามารถบันทึกรายการลงทะเบียนเรียนได้ ดังนั้นระบบจะมีข้อความแจ้งเตือน “ป้อนข้อมูลสำคัญไม่ครบถ้วน กรุณากลับไปป้อนข้อมูลให้ครบ” ให้เจ้าหน้าที่ป้อนข้อมูลสำคัญให้ครบถ้วนระบบจึงจะสามารถรับข้อมูลต่อได้

3.2 Class Diagram

เป็นแผนภาพที่ใช้ในการแสดงกลุ่มของคลาส โครงสร้างของคลาส อินเตอร์เฟส (Interface) และแสดงความสัมพันธ์ (Relationship) ระหว่างคลาส ซึ่งแผนภาพนี้เป็นแผนภาพที่จะพบมากที่สุดในทาง Object Orientation โดยสัญลักษณ์ของคลาสจะอธิบายถึงคุณสมบัติ (attribute) และ การดำเนินการ (Operation/Method) ดังภาพ

Class Name
Attribute
Operation / Method

ภาพที่ 3.6 แสดงสัญลักษณ์ของ Class

3.2.1 ส่วนประกอบของคลาส

3.2.1.1 Class Name เป็นคำนาม ขึ้นต้นด้วยตัวอักษรตัวใหญ่ ไม่มีช่องว่าง หากเป็นชื่อคลาสจะเป็นตัวหนา แต่ถ้าเป็นชื่อของ Abstract Class จะเป็นตัวเอียง

3.2.1.2 Attribute ประกอบด้วย

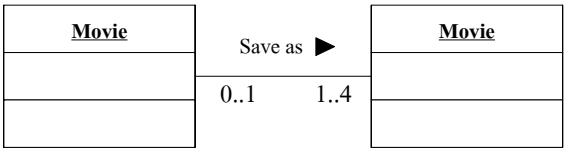
- Visibility เป็นเครื่องหมายแสดงสิทธิ์การเข้าถึง Attribute ของคลาสนั้น ได้แก่ Public(+) Private(-) Protect(#)
- ชื่อ Attribute
- เครื่องหมาย :
- ประเภทของ Attribute แบ่งได้ 2 ประเภทคือ Primitive Type และ Class Type

3.2.1.3 Operation / Method ประกอบด้วย

- Visibility เป็นเครื่องหมายแสดงสิทธิ์การเข้าถึง Operation / Method ของคลาสนั้น ได้แก่ Public(+) Private(-) Protect(#)
- ชื่อของ Operation / Method
- พารามิเตอร์ อยู่ภายในเครื่องหมายวงเล็บ เป็นตัวแปรหรือ object ที่ถูกส่งเข้าไปใน Operation / Method
- Return Type อยู่ต่อจากเครื่องหมาย : เป็นชนิดของผลลัพธ์ที่ได้จากการดำเนินงานซึ่งจะถูกส่งออกมาสู่ภายนอก

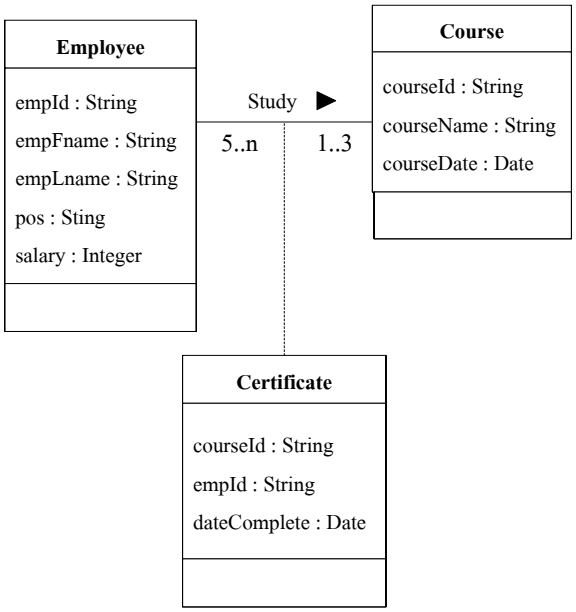
3.2.2 ความสัมพันธ์ระหว่างคลาส

3.2.2.1 Association เป็นความสัมพันธ์ในระดับเดียวกัน ไม่มีคลาสใดสำคัญมากไปกว่าคลาสใด



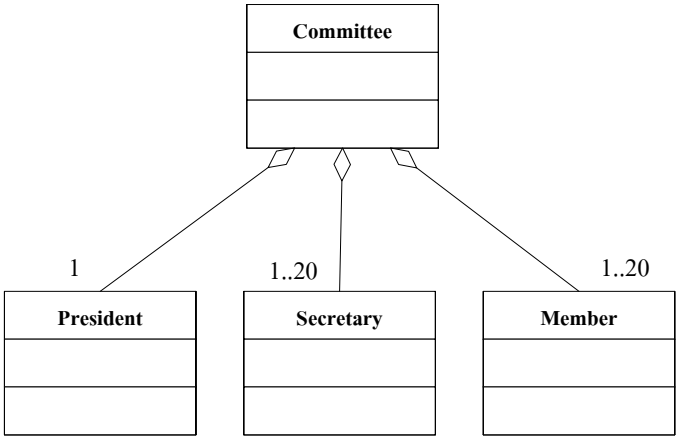
ภาพที่ 3.7 แสดงตัวอย่างความสัมพันธ์แบบ Association

Associative Class สำหรับการวิเคราะห์และออกแบบระบบเชิงวัตถุความสัมพันธ์ของคลาสแบบ Association มี 2 แบบคือ One-to-One หรือ One-to-Many ซึ่งถือว่าเป็นความสัมพันธ์แบบธรรมดา แต่หากความสัมพันธ์เป็นแบบ Many-to-Many แล้วจะต้องแตกคลาสเพิ่มอีก 1 คลาส และเรียกคลาสที่เพิ่มขึ้นนี้ว่า Associative Class



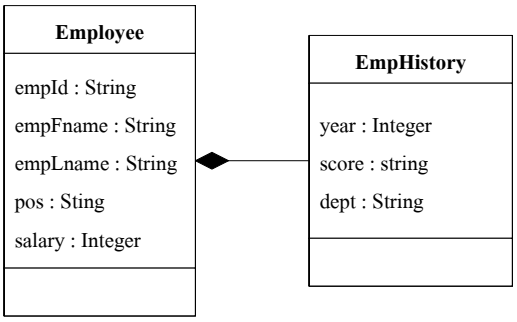
ภาพที่ 3.8 แสดงตัวอย่าง Associative Class

3.2.2.2 Aggregation เป็นความสัมพันธ์ระหว่างคลาสแบบต่างระดับ ในลักษณะของการเป็นองค์ประกอบ โดยคลาสที่เป็นองค์ประกองเรียกว่า “Part Class” ส่วนคลาสที่เกิดจากจากการรวมกันขององค์ประกอบต่าง ๆ เรียกว่า “Whole Class” โดยใช้สัญลักษณ์เส้นตรงหัวข้าวหลามตัดโปร่งลากจาก Part Class ไปยัง Whole Class ลักษณะสำคัญของความสัมพันธ์แบบ Aggregation คือ เมื่อลบ Whole Class ทิ้งไป คลาสที่เป็น Part Class จะไม่ถูกลบไปด้วย จะยังคงอยู่ต่อไปโดยไม่จำเป็นต้องพึ่งพา Whole Class



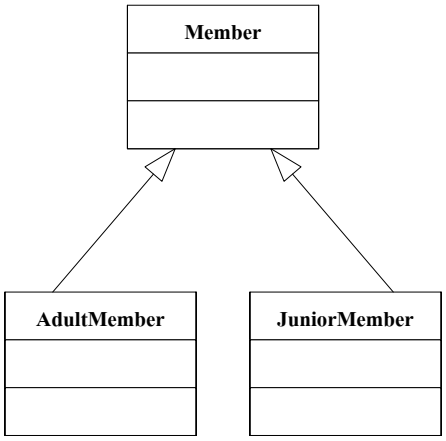
ภาพที่ 3.9 แสดงตัวอย่างความสัมพันธ์แบบ Aggregation

3.2.2.3 Composition เป็นความสัมพันธ์แบบ Aggregation ชนิดพิเศษที่ Whole Class มีผลต่อการดำรงอยู่ของ Part Class กล่าวคือหาก Whole Class ถูกทำลายหรือลบทิ้งไป Part Class ก็จะถูกทำลายหรือลบทิ้งไปด้วย หรือหากกล่าวอีกนัยหนึ่ง Part Class จะต้องทำงานร่วมกับ Whole Classไปจนตลอดอายุ Whole Class สัญลักษณ์ที่ใช้คือเส้นตรงหัวข้าวหลามตัดทึบลากจาก Part Class ไปยัง Whole Class



ภาพที่ 3.10 แสดงตัวอย่างความสัมพันธ์แบบ Composition

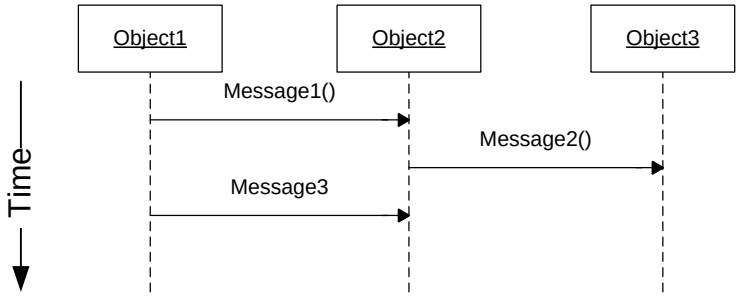
3.2.2.4 Generalization / Specialization เป็นการอธิบายความสัมพันธ์ระหว่างคลาสในลักษณะจำแนกชนิด การจำเพาะเจาะจงรายละเอียด หรือการหาลักษณะร่วมกันของคลาสต่างชนิดกัน เพื่อสร้างคลาสที่เป็นตัวแทนของกลุ่มคลาสเหล่านั้น



ภาพที่ 3.11 แสดงตัวอย่างความสัมพันธ์แบบ Generalization / Specialization

3.3 Sequence Diagram

เป็นแผนภาพที่แสดงให้เห็นถึงการปฏิสัมพันธ์ (Interaction) ระหว่างอ็อบเจ็กต์โดยเฉพาะการส่ง Message ระหว่างอ็อบเจ็กต์ตามลำดับของเวลา (Sequence) ที่เกิดเหตุการณ์ขึ้นจากน้อยไปมาก โดยจะมีสัญลักษณ์แสดงให้เห็นลำดับของการส่ง Message ตามเวลาส่งอย่างชัดเจน แสดงลักษณะของ Sequence Diagram



ภาพที่ 3.12 แสดงลักษณะของ Sequence Diagram

3.3.1 ส่วนประกอบของ Sequence Diagram

3.3.1.1 Actor คือ ผู้กระทำต่อระบบ

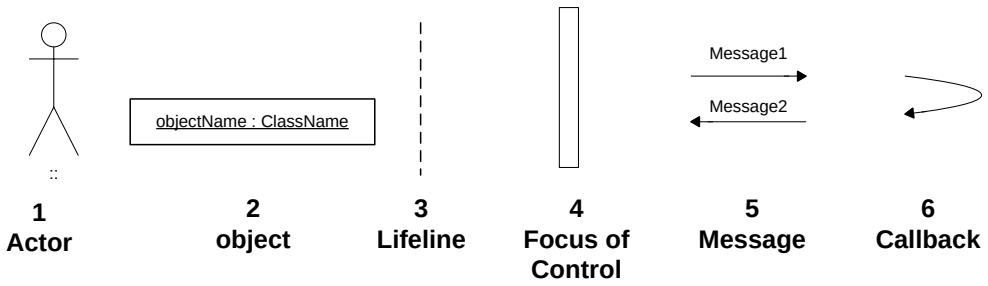
3.3.1.2 Object คือ อ็อบเจ็กต์ที่ต้องทำหน้าที่

3.3.1.3 Lifeline คือ เส้นแสดงชีวิตของอ็อบเจ็กต์หรือคลาส

3.3.1.4 Focus of Control / Activation จุดเริ่มต้นและจุดสิ้นสุดของแต่ละกิจกรรมในระหว่างที่มีชีวิตอยู่

3.3.1.5 A Message คำสั่งหรือฟังก์ชันที่คลาสหนึ่งส่งให้อีกคลาสหนึ่ง ซึ่งสามารถส่งกลับได้ด้วย

3.3.1.6 Callback / Self Delegation คือ การประมวลผลและคือค่าที่ได้ภายในอ็อบเจ็กต์เดียวกัน



ภาพที่ 3.13 แสดงสัญลักษณ์ภายใน Sequence Diagram

3.3.2 Stereotype

เป็นเทคนิคที่ใช้ในการเพิ่มชนิดของสัญลักษณ์พิเศษในภาษา UML จากสัญลักษณ์เดิมที่มีอยู่แล้ว ให้กลายเป็นสัญลักษณ์ชนิดใหม่ สำหรับคลาสที่นำมาใช้ในกระบวนการสร้าง Sequence Diagram ได้แก่ Entity Class Boundary Class Control Class

3.3.2.1 Entity Class คือคลาสที่เก็บข้อมูลสำคัญของระบบไว้ หรือกล่าวอีกนัยหนึ่งก็คือคลาสที่ใช้เป็นตัวแทนของฐานข้อมูลระบบ ลักษณะสำคัญของ Entity Class คือข้อมูลที่ถูกเก็บอยู่ใน Entity Class จะเป็นข้อมูลที่คงที่อยู่ตลอดเวลาแม้ว่าเครื่องคอมพิวเตอร์หรือระบบจะถูกปิดลงก็ตาม

Entity Class Name

ภาพที่ 3.14 แสดงสัญลักษณ์ Entity Class

3.3.2.2 Boundary Class คือ คลาสที่ถูกใช้โดย Actor เพื่อปฏิสัมพันธ์กับระบบ เป็นสื่อกลางตัวกลางระหว่าง Actor กับระบบ เช่น คลาสของฟอร์มใบสั่งซื้อ หน้าจอแสดงผล กล่องข้อความ เป็นต้น

Boundary Class Name

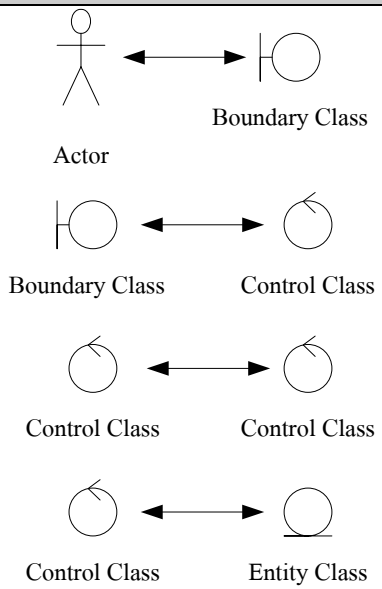
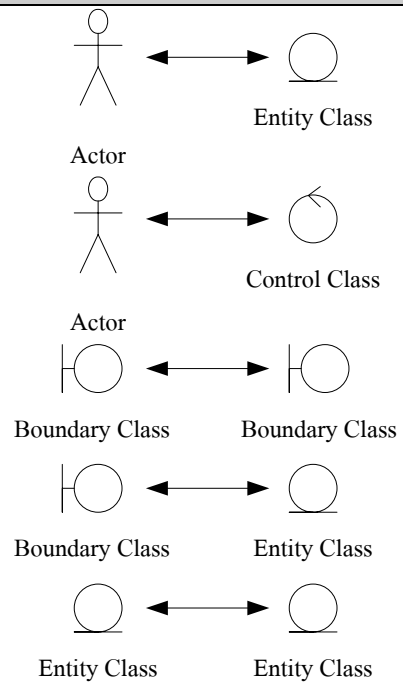
ภาพที่ 3.15 แสดงสัญลักษณ์ของ Boundary Class

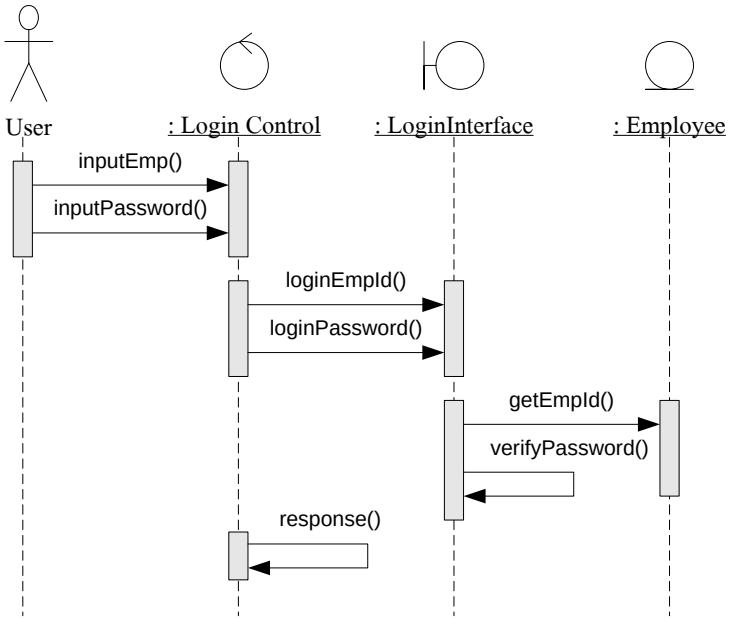
3.3.2.3 Control Class คือ คลาสที่คอยกำหนดกฎเกณฑ์และควบคุมการทำงานของระบบในแต่ละ Use case เมื่อมีการทำงานใด ๆ ก็ตามที่เกี่ยวข้องกับข้อมูลสำคัญของระบบ เช่น การเพิ่ม ลบ หรือ แก้ไขข้อมูลในฐานข้อมูล เป็นต้น จะต้องอยู่ภายใต้การควบคุมของ Control Class เพื่อป้องกันความเสียหายที่อาจเกิดขึ้นกับข้อมูล

Control Class Name

ภาพที่ 3.16 แสดงสัญลักษณ์ของ Control Class

ในปี ค.ศ. 1999 Rosenberg และ Scott ได้กำหนดเงื่อนไขบางประการในการวางตำแหน่งของ Actor และคลาสชนิดพิเศษทั้ง 3 ชนิดใน Sequence Diagram ขึ้นเพื่อเพิ่มความถูกต้อง

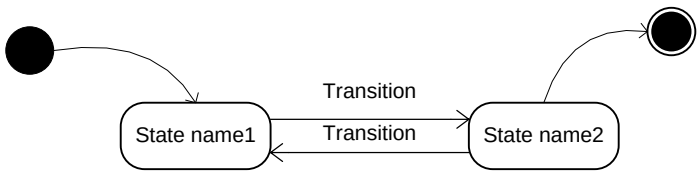
ถูกต้อง	ไม่ถูกต้อง
	



ภาพ 3.17 แสดงตัวอย่างการใช้งาน Sequence Diagram

3.4 Statechart Diagram (ถ้ามี)

Statechart Diagram เป็นแผนภาพที่แสดงให้เห็นพฤติกรรมของอ็อบเจ็กต์เช่นเดียวกับแผนภาพในกลุ่ม Behavioral Diagram อื่น ๆ แต่ Statechart Diagram จะเน้นที่การแสดงให้เห็นถึงสถานะ (State) การเปลี่ยนสถานะ (Transition) ที่มีเหตุการณ์ (Event) ที่เกิดขึ้นในช่วงชีวิตของอ็อบเจ็กต์ 1 ช่วง (1 Sequence) แสดงลักษณะของ Statechart Diagram **ดังภาพที่ 3.18**



ภาพที่ 3.18 แสดงลักษณะของ Statechart Diagram

3.4.1 สัญลักษณ์ของ Statechart Diagram

3.4.1.1 Initial State คือ จุดเริ่มต้นการเปลี่ยนสถานะ

ภาพที่ 3.19 แสดงสัญลักษณ์ของ Initial Stage

3.4.1.2 Final State คือ จุดสิ้นสุดของการเปลี่ยนสถานะ

ภาพที่ 3.20 แสดงสัญลักษณ์ของ Final Stage

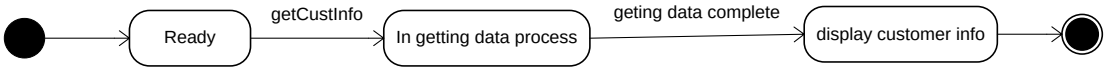
3.4.1.3 Transition คือ เส้นกระตุ้นให้เปลี่ยนสถานะซึ่งจะมีชื่อของเหตุการณ์ที่ทำให้เกิดการเปลี่ยนสถานะกำกับอยู่

ภาพที่ 3.21 แสดงสัญลักษณ์ของ Transition

3.4.1.4 State คือ สถานะของอ็อบเจ็กต์ ภายในเป็นชื่อของสถานะซึ่งมักเป็นวิลีหรือคำที่แสดงกริยาของสถานะ

ภาพที่ 3.22 แสดงสัญลักษณ์ของ State

ตัวอย่าง

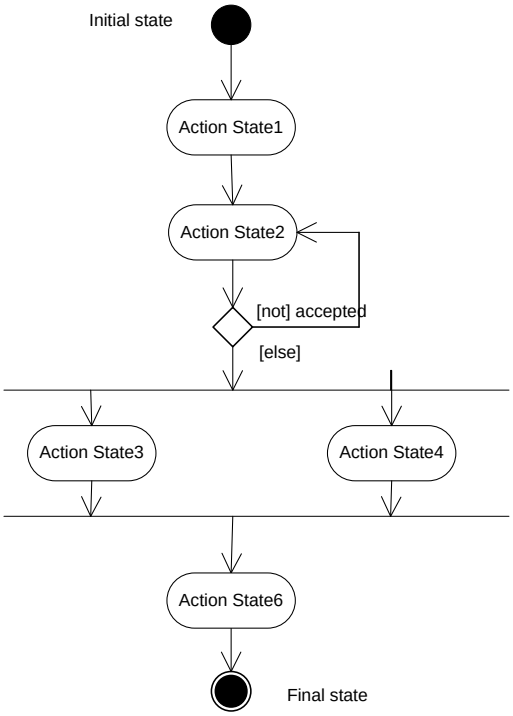


ภาพที่ 3.23 แสดงตัวอย่าง Statechart Diagram

แสดงให้เห็นสถานะที่เปลี่ยนแปลงไปของอ็อบเจ็กต์ “Customer” โดยเริ่มต้นที่สถานะ “เตรียมพร้อม (Ready)” จากนั้นมีอ็อบเจ็กต์อื่นส่ง Massage “getCusInfo” มากระตุ้นทำให้ “Customer” เกิดการเปลี่ยนสถานไปเป็น “อยู่ในกระบวนการดึงข้อมูล (in getting data process)” จากนั้นเมื่อกระบวนการดึงข้อมูลเสร็จสิ้น (getting data complete) ทำให้อ็อบเจ็กต์ “Customer” เปลี่ยนสถานะไปเป็น “แสดงสินค้า (display customer info)” และสิ้นสุดการเปลี่ยนแปลงสถานะ

3.5 Activity Diagram(ถ้ามี)*

เป็นแผนภาพที่แสดงให้เห็นลำดับการดำเนินกิจกรรม (Activity) จากกิจกรรมหนึ่งไปยังกิจกรรมหนึ่งภายในระบบที่เกิดจากการทำงานของออบเจ็กต์ ลักษณะของแผนภาพจะคล้ายกับ Flow Chart ดังรูปที่ 3.2.4

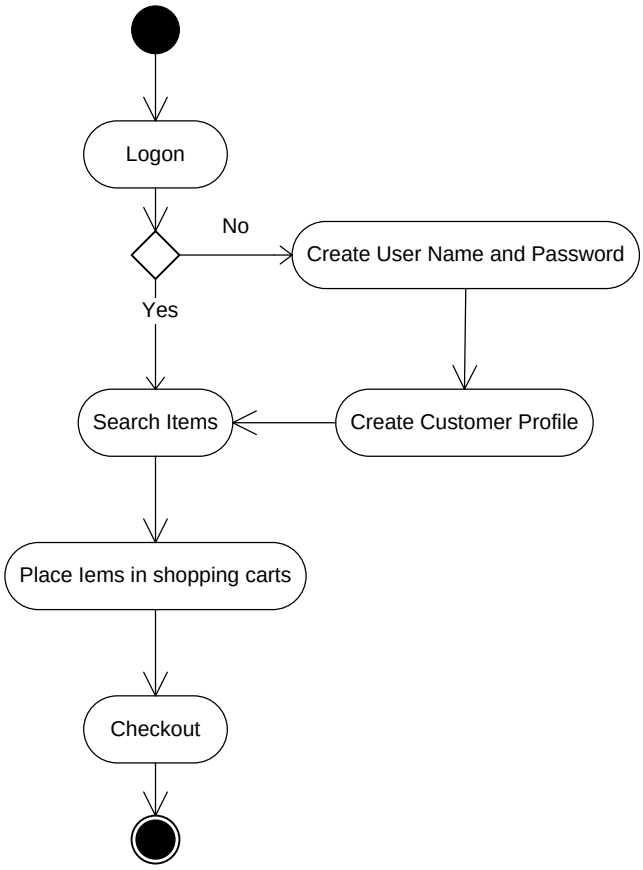


ภาพที่ 3.24 แสดงลักษณะของ Activity Diagram

สำหรับสัญลักษณ์ของ Activity Diagram จะคล้ายๆ กับ Statechart Diagram ได้แก่ จุดเริ่มต้น จุดสิ้นสุด และกิจกรรม แต่สำหรับ Activity Diagram จะไม่แสดงให้เห็นการเปลี่ยนแปลง สถานะแต่จะแสดงให้เห็นลำดับของกิจกรรมต่างๆ ซึ่งสามารถเขียนได้หลายรูปแบบดังนี้

3.5.1 แบบทางเลือกตัดสินใจ

การเขียน Activity Diagram แบบทางเลือกตัดสินใจ สามารถเขียนได้โดยลากลูกศรผ่านสัญลักษณ์ข้าวหลามตัดก่อนแล้วจึงลากไปยังแต่ละทางเลือก ดังภาพ 13 เป็นลำดับขั้นตอน “การสั่งซื้อสินค้าผ่านทางอินเตอร์เน็ต”

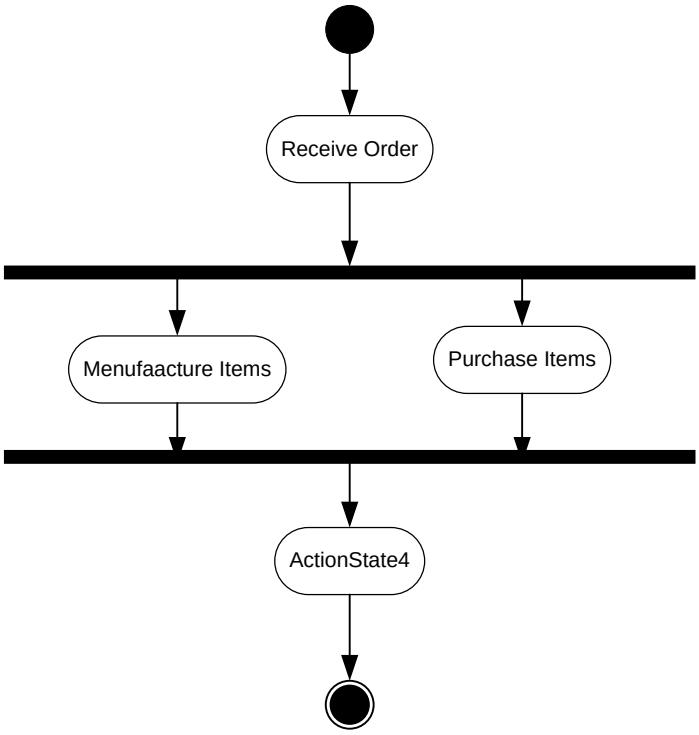


ภาพที่ 3.25 Activity Diagram แบบมีทางเลือกตัดสินใจ

จากรูป เป็นลำดับขั้นตอนการสั่งซื้อสินค้าทางอินเตอร์เน็ต เริ่มต้นเมื่อลูกค้าเข้ามาที่ เว็บไซต์ (Logon) โดยมีทางเลือกให้ดำเนินการต่อไปหากลูกค้ามีชื่อและรหัสสมาชิกแล้ว แต่หากยังไม่มีให้ไปลงทะเบียนเพื่อกำหนดชื่อและรหัสสมาชิกใหม่เสียก่อน จึงสามารถเลือก รายการสินค้าที่ต้องการได้แล้วทำการวางสินค้าที่เลือกไว้ในตระกร้า เมื่อเลือกสินค้าครบแล้ว ระบบจะทำการคำนวณราคาสินค้าพร้อมกับรายงานผล จากนั้นลูกค้าก็สามารถออกจากระบบได้

3.5.2 แบบมีการทำงานพร้อมกัน

สำหรับลักษณะการทำงานพร้อมกันจะเรียกว่า “Transition Fork” ซึ่งหมายถึง จุดเปลี่ยนแยก โดยจะต้องลากเส้นตรงขวางแนวนอนก่อนแยกสู่กิจกรรม ที่จะต้องกระทำพร้อมกัน เมื่อเสร็จสิ้นแล้วก็ลากเส้นตรงขวางแนวนอนก่อนแล้วจึงลากลูกศร รวมมายัง กิจกรรมอื่นต่อไป ลักษณะรวมนี้ เรียกว่า “Transition Join” ซึ่งหมายถึงจุดเปลี่ยนรวม ดังภาพ 12เป็นลำดับขั้นตอน “การผลิตสินค้าตามสั่ง” Activity Diagram ในลักษณะนี้สามารถมี Decision ได้ด้วย ดังตัวอย่างในภาพ 14

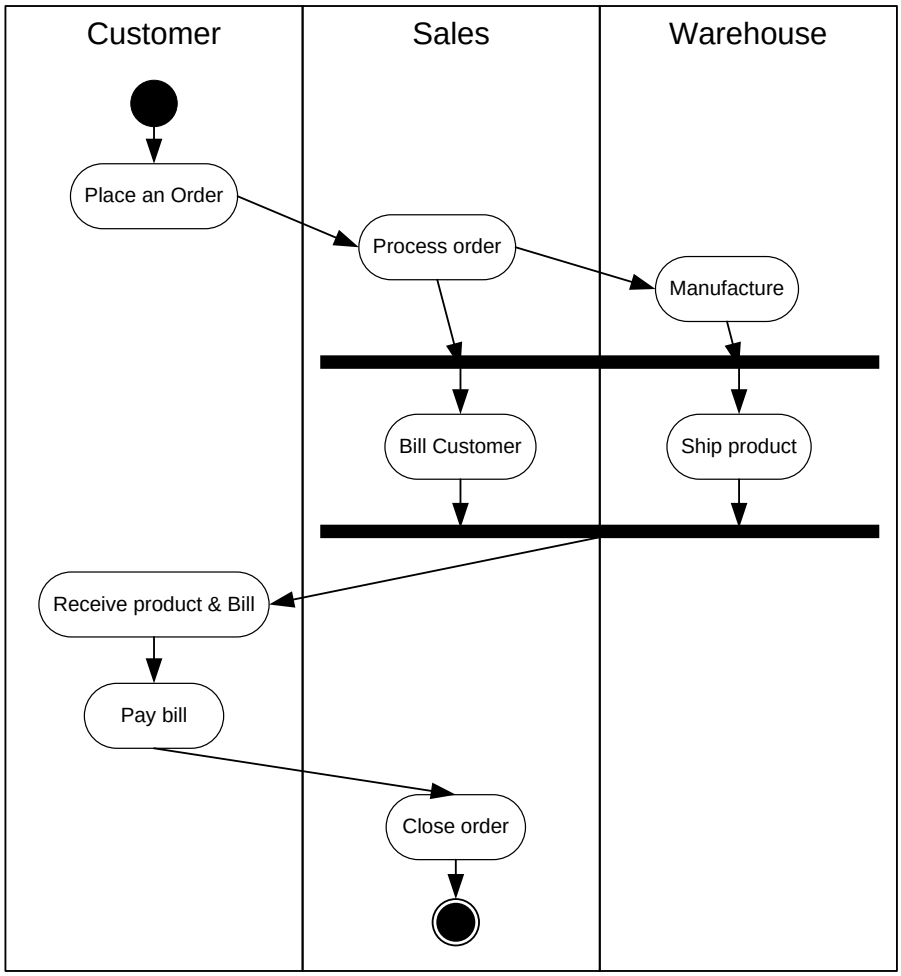


ภาพที่ 3.26 Receive Order

ลำดับต่อไปจะเป็นการผลิตสินค้าซึ่งบางส่วนอาจผลิตตามใบสั่งซื้อ (Manufacture Item) หรือบางส่วนอาจสั่งซื้อจากที่อื่น (Purchase Item) ก็ได้ หลังจากนั้นจึงจบกระบวนการรับใบสั่งซื้อ (Complete Order)

3.5.3 แบบแบ่งส่วนด้วย Swimlanes

เป็นการแบ่งส่วนการทำงานออกเป็นหลายๆ ส่วนที่เกี่ยวข้องกันใน 1 กิจกรรมโดยจะแบ่งเป็นแนวตั้งคล้ายกับลู่ในสระว่ายน้ำ แต่ละลู่คือแต่ละส่วน (ซึ่งอาจเป็นได้ทั้งบุคคล หน่วยงาน แผนก หรือสถานที่) ดังภาพที่ เป็นการแบ่งส่วนการทำงาน 3 ส่วน ได้แก่ ลูกค้า (Customer) งานขาย (Salse) และคลังสินค้า (Warehouse) ที่มีการทำงานเกี่ยวข้องกันในกิจกรรม “การขายสินค้า” โดยจากภาพกิจกรรมเริ่มต้นที่ “การสั่งซื้อสินค้า (Place an order)” ซึ่งเกิดขึ้นที่ส่วนของลูกค้า (Customer) จากนั้น เมื่อฝ่ายขาย (Sales) ได้รับคำสั่งซื้อจึงดำเนินการงานขาย (Process Order) ส่งเรื่องต่อไปยังฝ่ายงานคลัง (Warehouse) เพื่อจัดทำสินค้าที่ลูกค้าต้องการ แล้วส่งสินค้าไปยังลูกค้า (Ship product) พร้อมกับที่ฝ่ายขายแนบเอกสารการเก็บเงิน (Bill Customer) ไปด้วย เมื่อลูกค้าได้รับสินค้า (Receive product) แล้ว ก็ทำการชำระเงินตามระยะเวลาที่ตำลงกันไว้ (Pay Bill) ฝ่ายขายก็สามารถปิดใบสั่งซื้อรายการนี้ได้ (Close Order) สำหรับ Activity Diagram ในรูปแบบนี้สามารถมี Decision ได้



ภาพที่ 3.27 Activity Diagram แบบแบ่งส่วนการทำงานด้วย Swimlanes

**** หมายเหตุ ****

1. Statechart Diagram และ Activity Diagram อาจจะไม่จำเป็นต้องมีอยู่ในการวิเคราะห์ระบบงานบางระบบก็ได้ แต่ในบางระบบงาน เช่น การสร้างเกมส์ ระบบงานด้านฮาร์ดแวร์ เป็นต้น อาจจะมีความจำเป็นต้องมี Statechart Diagram และ Activity Diagram เข้ามาช่วยเพื่อให้สามารถมองเห็นวิธีการทำงานได้ชัดเจนมากยิ่งขึ้น
2. บทที่ 4 อ้างอิงตามรูปแบบบทที่ 4 ด้านดาต้าเบส
3. บทที่ 5 อ้างอิงตามรูปแบบบทที่ 5 ด้านฮาร์ดแวร์