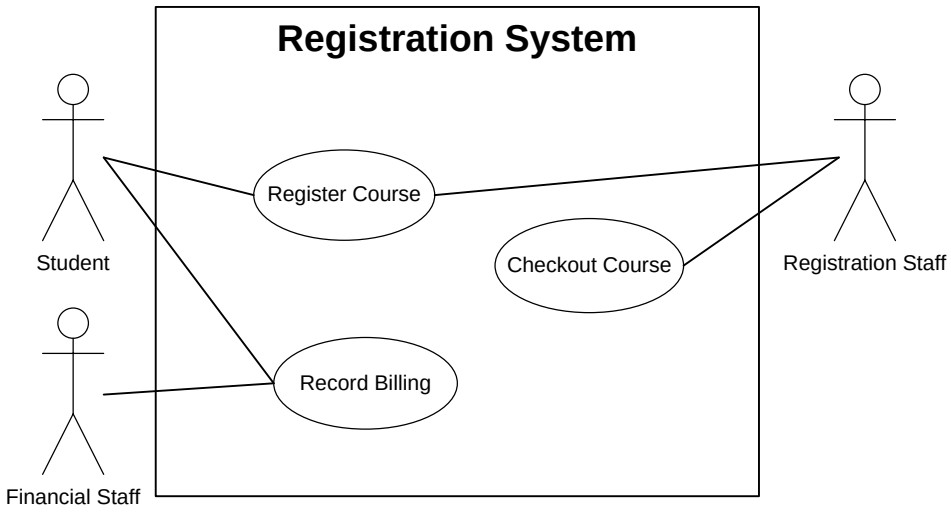
Use Case Diagram

Use Case Diagram เป็นแผนภาพที่ใช้แสดงให้ทราบว่าระบบทำงานหรือมีหน้าที่ใดบ้าง โดยมีสัญลักษณ์รูปวงรีแทน Use Case และสัญลักษณ์รูปคน (Stick Man Icon) แทน Actor สำหรับชื่อ Use Case นั้น ให้ใช้คำกริยาหรือกริยาวลี (คำกริยามีกรรมมารองรับ) เช่น ลงทะเบียนเรียน, ตรวจสอบรายวิชา, บันทึกการชำระเงิน, Generate Report, Enter Sales Data, Compute Commission เป็นต้น ส่วนการมีปฏิสัมพันธ์ระหว่าง Use Case และ Actor จะใช้เส้นตรงลากเชื่อมต่อกัน หรือจะใช้เส้นตรงมีหัวลูกศรก็ได้ ในที่นี้เลือกใช้เส้นตรงไม่มีหัวลูกศร ส่วนเส้นแบ่งขอบเขตระหว่าง Actor กับ Use Case จะใช้เส้นกรอบสี่เหลี่ยม เรียกว่า “System Boundary” และสิ่งสำคัญส่วนสุดท้ายก็คือ “ชื่อของระบบ (System Name)” ให้แสดงไว้ด้านบนสุดของแผนภาพ ตัวอย่างดังรูปที่ 1



รูปที่ 1 แสดง Use Case Diagram ของระบบลงทะเบียน

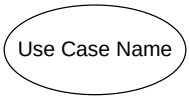
นอกจากนี้ หากกล่าวถึง Use Case Diagram ในด้านการพัฒนาระบบ นอกเหนือจากการนำมาใช้เก็บรวบรวมความต้องการต่างๆ แล้ว Use Case Diagram ยังถูกนำไปใช้เป็นพื้นฐานเพื่อการสร้างแผนภาพ (Diagram) ชนิดอื่นในขั้นตอนต่อๆ ไป และทีมงานยังสามารถใช้ Use Case Diagram เพื่อติดตามผลการดำเนินงานได้อีกด้วย

สัญลักษณ์และความสัมพันธ์ใน Use Case Diagram

สัญลักษณ์ที่สำคัญของ Use Case Diagram มีดังต่อไปนี้

- **Use Case**

  คือ หน้าที่ที่ระบบต้องกระทำ ใช้สัญลักษณ์รูปวงรี พร้อมทั้งเขียนชื่อ Use Case ซึ่งต้องใช้คำกริยาหรือกริยาวลีก็ได้



รูปที่ 2 แสดงสัญลักษณ์ Use Case

- **Actor**

  คือ ผู้เกี่ยวข้องกับระบบ ซึ่งรวมทั้ง Primary Actor และ Stakeholder Actor ที่เป็นมนุษย์ ในที่นี้จะใช้สัญลักษณ์รูปคน (Stick Man Icon) เหมือนกัน พร้อมทั้งเขียนชื่อ Actor ไว้ด้านล่างของสัญลักษณ์ด้วย แต่หากเป็น Actor ที่ไม่ใช่มนุษย์ เช่น ระบบงานอื่นที่อยู่นอกเหนือระบบที่เราสนใจ จะใช้รูปสี่เหลี่ยมแล้วเขียนคำว่า “<<actor>>” ไว้ด้านบน ดังรูปที่ 3

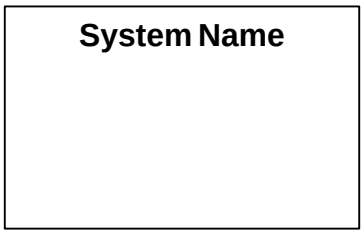
Primary & Stakeholder Actor ที่เป็นมนุษย์

Primary & Stakeholder Actor ที่ไม่ใช่มนุษย์

รูปที่ 3 แสดงสัญลักษณ์ Actor

- **System Boundary**

เส้นแบ่งขอบเขตระหว่างระบบกับผู้กระทำต่อระบบ (Use Case กับ Actor) ใช้รูปสี่เหลี่ยมเป็นสัญลักษณ์ พร้อมทั้งเขียนชื่อระบบไว้ด้านใน ดังรูปที่ 4



รูปที่ 4 แสดงสัญลักษณ์ System Boundary

- **Connection**

คือ เส้นที่ลากเชื่อมต่อระหว่าง Actor กับ Use Case ที่มีปฏิสัมพันธ์กัน ใช้เส้นตรงไม่มีหัวลูกศรเป็นสัญลักษณ์ของ Connection ส่วน Connection ที่ใช้เชื่อมต่อระหว่าง Use Case กับ Use Case กรณีที่ Use Case นั้นมีความสัมพันธ์ซึ่งกันและกัน จะใช้สัญลักษณ์เส้นตรงมีหัวลูกศร พร้อมทั้งเขียนชื่อความสัมพันธ์ไว้ตรงกลางเส้นด้วย โดยเขียนไว้ภายในเครื่องหมาย <<...>> ดังรูปที่ 5

Connection ระหว่าง Actor กับ Use Case

Connection ระหว่าง Use Case กับ Use Case

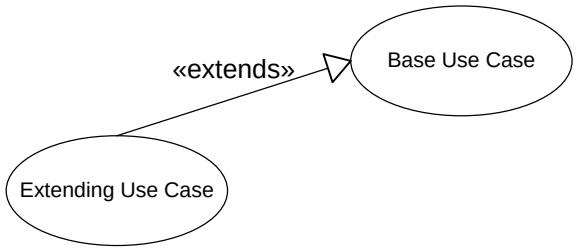
รูปที่ 5 แสดงสัญลักษณ์ Connection

- **Extend Relationship**

เป็นความสัมพันธ์แบบขยายหรือเพิ่ม เกิดขึ้นในกรณีที่บาง Use Case ดำเนินกิจกรรมของตนเองไปตามปกติ แต่อาจจะมีเงื่อนไขหรือสิ่งกระตุ้นบางอย่างที่ส่งผลให้กิจกรรมตามปกติของ Use Case นั้นถูกรบกวนจนเบี่ยงเบนไป ซึ่งเราสามารถแสดงเงื่อนไขหรือ

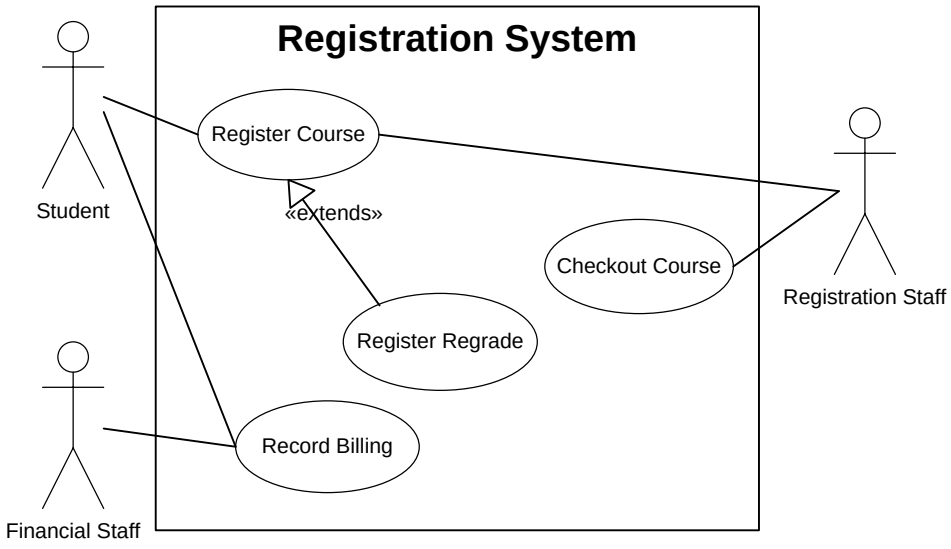
สิ่งกระตุ้นเหล่านั้นได้ในรูปของ “Use Case” และเรียกความสัมพันธ์ระหว่าง Use Case ในลักษณะนี้ว่า “Extend Relationship” โดยเรียก Use Case ที่ถูกรบกวนหรือ Use Case ที่ดำเนินงานตามปกติว่า “Base Use Case” และเรียก Use Case ที่ทำหน้าที่รบกวนหรือกระตุ้น Base Use Case ว่า “Extending Use Case”

กล่าวโดยสรุป ก็คือ Use Case หนึ่งทำหน้าที่ตามปกติ เมื่อเกิดเหตุการณ์ผิดปกติขึ้นจะต้องทำหน้าที่พิเศษเพิ่ม โดยหน้าที่พิเศษที่เพิ่มขึ้นก็คือ “Extending Use Case” นั่นเอง ดังนั้น อาจกล่าวได้ว่า Use Case ที่เป็น Extending Use Case จะเกิดขึ้นเพียงบางครั้งเท่านั้น (ไม่ได้เกิดขึ้นทุกครั้งที่ดำเนินกิจกรรมตาม Base Use Case) การวาดเส้น Connection เชื่อมระหว่าง Use Case ทั้งสอง ให้เริ่มต้นลากเส้นตรงจาก Extending Use Case หันลูกศรชี้ไปที่ Base Use Case ดังรูปที่ 6



รูปที่ 6 แสดงการวาดเส้น Connection เชื่อมระหว่าง Extending Use Case กับ Base Use Case

แสดงตัวอย่าง Extend Relationship ของระบบลงทะเบียนได้ดังรูปที่ 7



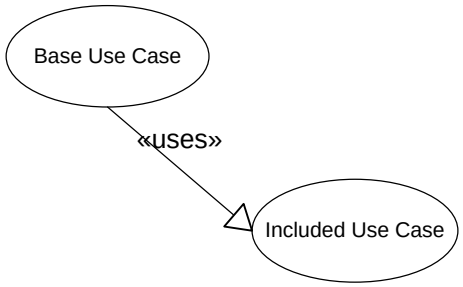
รูปที่ 7 แสดง Use Case Diagram ที่มีความสัมพันธ์แบบ Extend Relationship

จากรูปที่ 7 สังเกตที่ Use Case “Register Course” ซึ่งเป็น Base Use Case คือ ทำหน้าที่รับลงทะเบียนตามปกติ แต่เมื่อมีเงื่อนไขหรือมีเหตุการณ์พิเศษเกิดขึ้น คือ “นักศึกษาบางคนอาจมีการลงทะเบียนเรียนซ้ำเพื่อปรับเกรดด้วย (Regrade)” จึงได้เพิ่ม Extending Use Case เพื่อมารองรับหน้าที่พิเศษดังกล่าว นั่นคือ “Register Regrade”

- **Include Relationship**

  ความสัมพันธ์อีกรูปแบบหนึ่งของ Use Case Diagram ก็คือ ความสัมพันธ์แบบเรียกใช้ เกิดขึ้นในกรณีที่ Use Case หนึ่งไปเรียกหรือดึงกิจกรรมของอีก Use Case หนึ่งมาใช้เพื่อให้กิจกรรมนั้นเกิดขึ้นจริงใน Use Case ของตนเอง หรือกล่าวให้ง่ายกว่านั้นคือ กิจกรรมใน Use Case หนึ่ง อาจจะถูกผนวกเข้าไปรวมกับกิจกรรมของอีก Use Case หนึ่ง เราเรียกความสัมพันธ์ระหว่าง Use Case ในลักษณะนี้ว่า “Include Relationship” โดย Use Case ที่ทำหน้าที่ดึงกิจกรรมมาจาก Use Case อื่นๆ เรียกว่า “Base Use Case” ในขณะที่ Use Case ที่ถูกเรียก หรือถูกดึงกิจกรรมมาใช้ เรียกว่า

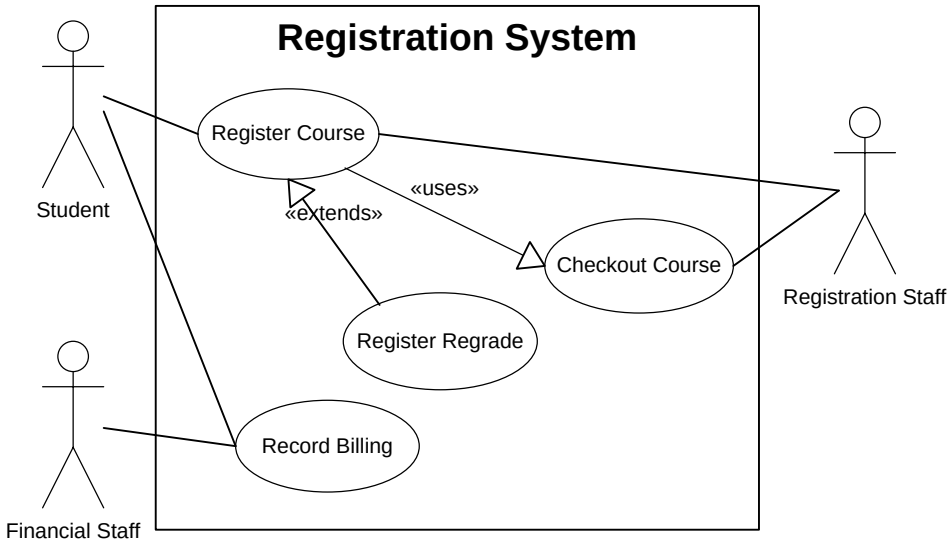
“Included Use Case” สามารถเขียนเส้น Connection ได้ในทิศทางตรงกันข้ามกับ Extend Relationship โดยเริ่มต้นลากเส้นตรงจาก Base Use Case หันลูกศรชี้ไปที่ Included Use Case แล้วเขียนชื่อ Relationship “<<uses>>” (บางตำราจะใช้คำว่า <<include>>) ไว้ตรงกลางเส้นด้วย ดังรูปที่ 8



รูปที่ 8 แสดงการลากเส้น Connection ระหว่าง Base Use Case กับ Included Use Case

ความสัมพันธ์ระหว่าง Use Case แบบ Include เป็นการสนับสนุนหลักการนำกลับมาใช้ใหม่ของ Use Case (Use Case Reusability) กล่าวคือ Use Case หนึ่งสามารถถูก Include ได้โดย Base Use Case หลายๆ ตัว และสามารถถูก Include ได้มากกว่าหนึ่งครั้งด้วย เช่น ในการทำงานของระบบเอทีเอ็ม Use Case “การตรวจสอบผู้เข้าใช้ระบบ (Validate User)” สามารถเป็น Included Use Case ให้กับ Base Use Case หลายๆ ตัว ได้แก่ Base Use Case “การถอนเงิน (Withdraw Money)” และ Base Use Case “การโอนเงิน (Transfer Money)”

ดังนั้น จากรูปที่ 7 เมื่อพิจารณาแล้ว Use Case “ตรวจสอบรายวิชา (Checkout Course)” สามารถถูกเรียกใช้จาก Use Case “ลงทะเบียนเรียน (Register Course)” ได้ ดังนั้น Use Case “Checkout Course” มีความสัมพันธ์กับ Use Case “Register Course” แบบ Include แสดง Use Case Diagram อีกครั้งดังรูปที่ 9



รูปที่ 9 แสดง Use Case Diagram ที่มี Included Relationship

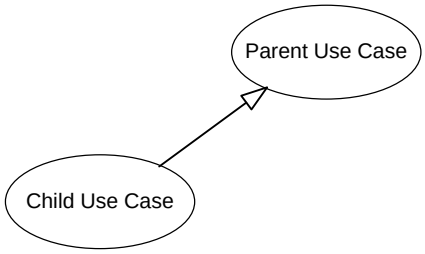
หมายเหตุ 1. ในบางตำราจะเรียกความสัมพันธ์แบบ Include ได้อีกอย่างหนึ่งว่า “Use Relationship”

2. ความแตกต่างระหว่างความสัมพันธ์แบบ Extend กับ Include คือ Extend จะเป็น Use Case ที่ถูกเรียกใช้เฉพาะบางกรณีเท่านั้น แต่ Include จะถูกเรียกใช้ทุกครั้งที่ Base Use Case มีการดำเนินกิจกรรม

- **Generalization/Specialization Relationship**

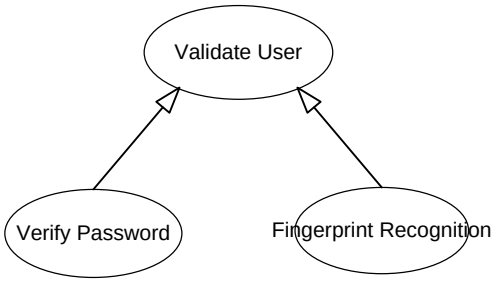
  Generalization หรือ Specialization ที่เกิดขึ้นระหว่าง Use Case มีคุณสมบัติแตกต่างจาก Generalization/Specialization ที่เกิดขึ้นระหว่างคลาส คือ **ความสัมพันธ์ลักษณะดังกล่าวที่เกิดขึ้นใน Use Case นี้จะไม่มีการถ่ายทอดคุณลักษณะ (ไม่มีการ Inherit) แต่จะใช้เพื่อแสดงความสัมพันธ์แบบจำแนกแยกแยะประเภทของ Use Case เท่านั้น** อย่างไรก็ตาม Use Case ที่เป็น Use Case หลักในการจำแนกประเภทจะเรียกว่า “Parent Use Case” ส่วน Use Case ที่ถูกจำแนกแยกแยะออกมา จะเรียกว่า “Child Use Case” ส่วนสัญลักษณ์เชื่อม

ความสัมพันธ์ ให้ใช้เส้นตรงลูกศรโปร่ง ลากจาก Child Use Case ให้ลูกศรชี้ไปที่ Parent Use Case ดังรูปที่ 10



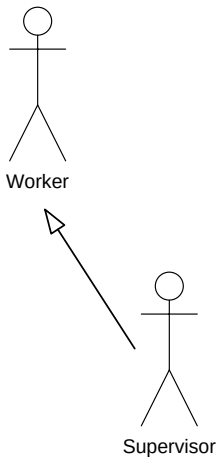
รูปที่ 10 แสดงความสัมพันธ์ระหว่าง Use Case แบบ Generalization / Specialization

ตามที่กล่าวไปแล้วว่าเราจะใช้ Generalization / Specialization ในกรณีที่ต้องการแสดงความสัมพันธ์ในเชิงการจำแนกแยกแยะประเภทของ Use Case เช่น การตรวจสอบความถูกต้องของผู้ใช้งานระบบ (Validate user) สามารถกระทำได้หลายวิธี ได้แก่ การตรวจสอบจากรหัสผ่าน (Verify Password) และการตรวจจากลายนิ้วมือ (Fingerprint Recognition) เป็นต้น แสดงความสัมพันธ์ได้ดังรูปที่ 11



รูปที่ 11 แสดงความสัมพันธะหว่าง Use Case แบบ Generalization / Specialization

นอกจากเราจะใช้ความสัมพันธ์แบบ Generalization / Specialization กับ Use Case แล้ว เรายังสามารถใช้ความสัมพันธ์ลักษณะนี้กับ Actor ได้เช่นเดียวกัน ตัวอย่างเช่น เราสามารถใช้ UML อธิบายข้อเท็จจริง “คนคุมงาน (Worker Supervisor) จัดเป็นคนงานประเภทพิเศษที่มีหน้าที่พิเศษกว่าคนงาน (Worker) ทั่วไป” ได้ดังรูปที่ 12



รูปที่ 12 แสดงความสัมพันธ์แบบ Generalization / Specialization ระหว่าง Actor

การสร้าง Use Case Diagram

เริ่มต้นการสร้าง Use Case Diagram ด้วยการวิเคราะห์หาขอบเขตของระบบ (Problem Domain) ซึ่งประกอบไปด้วยการค้นหา Actor ที่ควรมีในระบบ และ Use Case ที่มีปฏิสัมพันธ์โดยตรงกับ Actor เหล่านั้นขึ้นมาก่อน จากนั้นจึงเพิ่มเติม Use Case อื่นๆ เข้าไปจนครบหน้าที่การทำงานของระบบ

1. ค้นหา Actor
2. ค้นหา Use Case ที่มีปฏิสัมพันธ์กับ Actor นั้นโดยตรง
3. ค้นหาและสร้างความสัมพันธ์ระหว่าง Use Case หรือ Actor (ถ้ามี) แล้วเพิ่มเติม Use Case ใหม่ซึ่งอาจเป็น Included Use Case, Extending Use Case ที่เพิ่มเติมจาก Base Use Case ที่มีอยู่แล้ว หรือจะเพิ่ม Base Use Case ใหม่ก็ได้ (ถ้ามี)
4. ต้องไม่มี Actor ใดเลยที่ไม่มีปฏิสัมพันธ์กับ Use Case
5. ต้องไม่มี Use Case ใดเลยที่ไม่มีปฏิสัมพันธ์กับ Actor
6. Use Case ทุกตัวต้องมีปฏิสัมพันธ์อย่างใดอย่างหนึ่งกับ Actor หรือ Use Case ตัวอื่นๆ เสมอ
7. เขียนคำอธิบายแต่ละ Use Case จนครบถ้วน

ข้อแนะนำในการสร้าง Use Case Diagram

ปัญหาสำคัญที่เกิดขึ้นในระหว่างการสร้าง Use Case Diagram ก็คือ การที่นักวิเคราะห์ระบบหรือทีมงานที่รับผิดชอบไม่สามารถระบุได้ชัดเจนว่าจะต้องแสดง Use Case ให้ละเอียดมากน้อยเพียงใด จะต้องแสดง Use Case ใดบ้าง ไม่แสดง Use Case ใดบ้าง หรือต้องแสดง Use Case ทั้งหมดที่เป็นหน้าที่ที่ระบบต้องกระทำ ทำให้บางครั้งทีมงานต้องเสียเวลาไปกับกระบวนการสร้าง Use Case Diagram มากเกินความจำเป็น เนื่องจากสุดท้ายแล้ว Use Case Diagram ที่ได้มาก็ไม่ได้ถูกนำไปใช้ในขั้นตอนต่อไปของการพัฒนาระบบ หรือหากถูกนำไปใช้เป็นพื้นฐานในการสร้างแผนภาพชนิดอื่นต่อไป ก็จะทำให้การดำเนินงานในขั้นตอนอื่นล่าช้าไปด้วย ดังนั้น ในที่นี้จึงมีข้อแนะนำบางประการที่จะทำให้ขั้นตอนการสร้าง Use Case Diagram เป็นขั้นตอนที่ไม่ต้องใช้เวลามากจนเกินไป ดังนี้

1. Use Case Diagram ใช้เพื่อแสดงให้เห็นถึงข้อมูลความต้องการของผู้ใช้ระบบเท่านั้น หรืออาจกล่าวได้ว่า Use Case Diagram ใช้เพื่อเก็บรวบรวมข้อมูลความต้องการจากผู้ใช้เท่านั้น เนื่องจากหาก Use Case Diagram ที่ได้ในรอบแรกยังไม่สามารถครอบคลุมความต้องการของผู้ใช้ ก็สามารถนำมาปรับปรุงแก้ไขเพิ่มเติมได้จนกว่าจะครบถ้วน เมื่อครบถ้วนแล้ว นั่นหมายความว่า ทีมงานมีความเข้าใจกับข้อมูลความต้องการในระบบใหม่ขอผู้ใช้แล้ว จึงอาจนำ Use Case Diagram ไปใช้เป็นพื้นฐานในการสร้างแผนภาพชนิดอื่นหรือไม่ก็ได้ (ขึ้นอยู่กับทีมงาน)
2. Use Case Diagram อาจมีความละเอียดมากหรือน้อยก็ได้ ขึ้นอยู่กับมุมมอง เทคนิคและประสบการณ์ของทีมงานหรือนักวิเคราะห์ระบบ จึงไม่มีข้อสรุปใดระบุได้ว่า Use Case Diagram ลักษณะใดถูกต้อง หรือลักษณะใดไม่ถูกต้อง เนื่องจาก สุดท้ายแล้วไม่ว่าจะเป็น Use Case Diagram ลักษณะใดก็ตาม ย่อมส่งผลให้ทีมงานเกิดความเข้าใจในข้อมูลความต้องการระบบใหม่ของผู้ใช้ได้อย่างถูกต้องเช่นเดียวกัน
3. ให้ตระหนักอยู่เสมอว่า Use Case Diagram นั้นใช้เพื่อการสื่อสารระหว่างนักวิเคราะห์ระบบกับผู้ใช้ ไม่ได้ใช้สื่อสารระหว่างนักวิเคราะห์ระบบกับโปรแกรมเมอร์ ดังนั้น Use Case Diagram จึงควรทำให้ผู้ใช้เห็นภาพรวมของระบบในเชิงกว้างมากกว่าเชิงลึก (ในเชิงกว้าง คือ ในระดับที่ทราบได้ว่าครอบคลุมความต้องการของผู้ใช้หรือไม่ ในเชิงลึก คือ

ในระดับรายละเอียดการทำงานของระบบ) ซึ่งจะทำให้ผู้ใช้ทราบว่านักวิเคราะห์ระบบเข้าใจความต้องการของตนเองได้อย่างครบถ้วนหรือไม่นั่นเอง

4. Use Case Diagram โดยส่วนใหญ่จะไม่แสดงให้เห็นถึงการทำงานในระดับการจัดการข้อมูลในฐานข้อมูล เช่น การเพิ่ม ลบ แก้ไข หรือปรับปรุงข้อมูล เป็นต้น ทั้งนี้ เนื่องจากโปรแกรมของระบบงานทุกระบบจะต้องมีหน้าที่ในส่วนของการจัดการข้อมูลเป็นหน้าที่พื้นฐานอยู่แล้ว ไม่จำเป็นต้องนำมาแสดงให้เห็นใน Use Case Diagram ก็ได้
5. จากข้อ 3 สิ่งที่ควรนำมาแสดงใน Use Case Diagram ก็คือ หน้าที่หลักๆ หรือหน้าที่ที่เป็นจุดเด่นของระบบที่ผู้ใช้งานต้องการให้ระบบกระทำได้อย่างแท้จริงเท่านั้น (เป็นการสนับสนุนข้อความที่ว่า “หน้าที่ในการเพิ่ม ลบ แก้ไข หรือปรับปรุงข้อมูลนั้นเป็นหน้าที่พื้นฐานที่ทุกระบบต้องมีอยู่แล้ว”)
6. จากข้อ 4 คำว่า “หน้าที่หลักหรือหน้าที่ที่เป็นจุดเด่นของระบบ” ในที่นี้หมายถึง หน้าที่ที่ระบบจะต้องกระทำ (System Operate) ตามความต้องการของผู้ใช้ ไม่ใช่หน้าที่ที่ผู้ใช้จะต้องกระทำ (Human Operate) อันเนื่องจากการทำงานของระบบ

การเขียนคำอธิบาย Use Case

เช่นเดียวกับการวิเคราะห์ระบบแบบเดิมที่จะต้องมีการเขียนคำอธิบาย Context Diagram เพื่อให้ทราบรายละเอียดปลีกย่อยของแต่ละระบบย่อยด้วย สำหรับในแต่ละ Use Case ตามที่กล่าวไปแล้วว่าประกอบไปด้วยการกระทำหลายๆ อย่างต่อเนื่องกันเป็นลำดับขั้นตอน ดังนั้นการแสดงแผนภาพแทนความคิดของนักวิเคราะห์ระบบที่มีต่อระบบเพียงอย่างเดียวนั้นอาจไม่เพียงพอ จำเป็นต้องมีการเขียนอธิบายรายละเอียดควบคู่กันไปด้วย เรียกคำอธิบาย Use Case ดังกล่าวว่า “กระแสของเหตุการณ์ (Flow of Event)”

การเขียนคำอธิบาย Use Case หรือ Flow of Event นั้น ปัจจุบันมีรูปแบบแตกต่างกันออกไป แต่ในที่นี้จะเขียนคำอธิบายโดยมีส่วนประกอบ 2 ส่วนสำคัญ ได้แก่ Main Flow และ Exceptional Flow

1. Main Flow

   คือ ลำดับกิจกรรม เมื่อ Use Case ดำเนินกิจกรรมตามปกติ โดยการเขียนคำอธิบายในลักษณะเป็นย่อหน้า (Paragraph) และ Main Flow จะต้องมีเพียงหนึ่งเดียวเท่านั้น

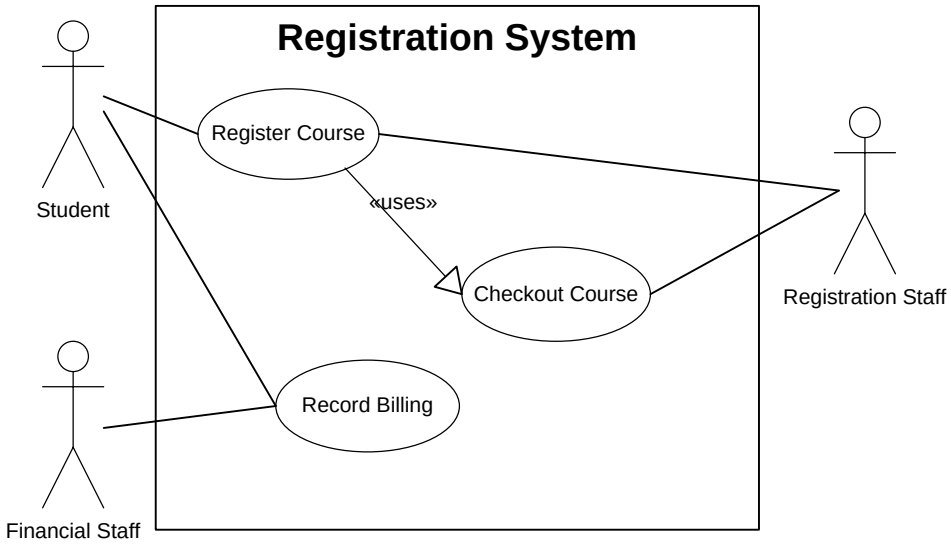
2. Exceptional Flow

   คือ ลำดับกิจกรรม เมื่อ Use Case ดำเนินกิจกรรมผิดจากปกติ โดยสามารถมีมากกว่า 1 Flow ได้

ทั้ง Main Flow และ Exceptional Flow จะต้องระบุถึงสาเหตุของการเริ่มต้นและสิ้นสุดกิจกรรมด้วยเสมอ นอกจากการระบุถึง Main Flow และ Exceptional Flow แล้ว เราสามารถเพิ่มเติมส่วนประกอบอื่นๆ ได้ตามความเหมาะสม โดยในที่นี้จะเพิ่ม Use Case Title, Use Case Id, Primary Actor และ Stakeholder Actor ด้วย

ตัวอย่าง Use Case Diagram ของระบบลงทะเบียน

ระบบลงทะเบียนมีกลุ่มบุคคลที่เกี่ยวข้อง 2 กลุ่ม ได้แก่ นักศึกษา และพนักงานของมหาวิทยาลัย (เจ้าหน้าที่ฝ่ายทะเบียนและเจ้าหน้าที่ฝ่ายการเงิน) ในแต่ละเทอมจะต้องมีนักศึกษามาลงทะเบียนเรียนของภาคเรียนปกติ โดยนักศึกษาจะต้องกรอกแบบฟอร์มลงทะเบียนให้เรียบร้อยแล้วนำไปยื่นกับเจ้าหน้าที่ฝ่ายทะเบียนในวันและเวลาที่ประกาศไว้ เมื่อเจ้าหน้าที่รับแบบฟอร์มลงทะเบียนมาแล้ว จะทำการตรวจสอบวิชาที่นักศึกษาได้ลงไว้ในแบบฟอร์มกับประวัติการเรียนว่าถูกต้องหรือไม่ เนื่องจาก บางวิชาของแต่ละเทอมมีเงื่อนไขว่าจะลงทะเบียนได้ก็ต่อเมื่อสอบผ่านอีกวิชาหนึ่งมาก่อน เมื่อตรวจสอบพบว่าถูกต้องแล้ว เจ้าหน้าที่ฝ่ายทะเบียนจะคำนวณเงินค่าลงทะเบียนเรียน แล้วบันทึกลงในฐานข้อมูล สั่งพิมพ์ใบรับลงทะเบียนโดยแบ่งออกเป็น 2 ส่วน ส่วนที่ 1 นักศึกษาเก็บไว้เอง ส่วนที่ 2 นำไปชำระเงินโดยโอนผ่านทางธนาคาร แล้วนำใบรับชำระเงินกลับมาให้เจ้าหน้าที่ฝ่ายการเงินบันทึกสถานะการชำระเงินเป็นขั้นตอนสุดท้าย



รูปที่ 13 แสดง Use Case Diagram ของระบบลงทะเบียนเรียน

จากรูปที่ 13 เป็น Use Case Diagram ที่สร้างเสร็จเรียบร้อยแล้ว เราสามารถทราบได้ว่าระบบลงทะเบียนจะต้องมีหน้าที่หลักๆ อยู่ 3 หน้าที่ ได้แก่ ลงทะเบียนเรียน ตรวจสอบรายวิชา และบันทึกการชำระเงินค่าลงทะเบียน โดยผู้ที่มีหน้าที่ลงทะเบียนก็คือ “นักศึกษา” ส่วนผู้ที่มีหน้าที่รับลงทะเบียนก็คือ “เจ้าหน้าที่ฝ่ายทะเบียน” และผู้ที่มีหน้าที่บันทึกการชำระเงินค่าลงทะเบียน ก็คือ “เจ้าหน้าที่ฝ่ายการเงิน” ซึ่งต่างก็เป็นพนักงานของมหาวิทยาลัยเช่นเดียวกัน

<u>คำอธิบาย Use Case ระบบลงทะเบียน</u>

จากรายละเอียดของ “ระบบลงทะเบียน” และ Use Case Diagram ที่แสดงในรูปที่ 13 คำอธิบายของแต่ละ Use Case มีดังนี้

Use Case Title : Register Course	**Use Case Id :** 1
Primary Actor : Registration Staff	
Stakeholder Actor : Student	
Main Flow : ระบบลงทะเบียนมีกลุ่มบุคคลที่เกี่ยวข้อง 2 กลุ่ม ได้แก่ นักศึกษา และพนักงานของมหาวิทยาลัย (เจ้าหน้าที่ฝ่ายทะเบียนและเจ้าหน้าที่ฝ่ายการเงิน) ในแต่ละเทอมจะต้องมีนักศึกษามาลงทะเบียนเรียนของภาคเรียนปกติ โดยนักศึกษาจะต้องกรอกแบบฟอร์มลงทะเบียนไม่เกิน 8 วิชา แล้วนำไปยื่นกับเจ้าหน้าที่ฝ่ายทะเบียนในวันและเวลาที่ประกาศไว้ เมื่อเจ้าหน้าที่รับแบบฟอร์ม	

ลงทะเบียนมาแล้ว จะป้อนรหัสนักศึกษาเพื่อทำการตรวจสอบวิชาที่นักศึกษาได้ลงไว้ในแบบฟอร์มกับประวัติการเรียนว่าถูกต้องหรือไม่ เนื่องจาก บางวิชาของแต่ละเทอมมีเงื่อนไขว่าจะลงทะเบียนได้ก็ต่อเมื่อสอบผ่านอีกวิชาหนึ่งมาก่อน เมื่อตรวจสอบพบว่าถูกต้องแล้ว เจ้าหน้าที่ฝ่ายทะเบียนจะคำนวณเงินค่าลงทะเบียนเรียน จากนั้นบันทึกลงในฐานข้อมูล ระบบแสดงข้อความบันทึกข้อมูลเรียบร้อยแล้ว สั่งพิมพ์ใบรับลงทะเบียนโดยแบ่งออกเป็น 2 ส่วน ส่วนที่ 1 นักศึกษาเก็บไว้เอง ส่วนที่ 2 นำไปชำระเงินโดยโอนผ่านทางธนาคาร
Exceptional Flow ที่ 1 : กรณีที่เจ้าหน้าที่ฝ่ายทะเบียนป้อนรหัสนักศึกษาผิดพลาด ระบบจะแสดงข้อความแจ้งเตือน “ไม่พบข้อมูลนักศึกษา” เพื่อให้เจ้าหน้าที่ทราบว่ามีข้อผิดพลาดเกิดขึ้น และให้เจ้าหน้าที่ป้อนรหัสนักศึกษาใหม่อีกครั้ง
Exceptional Flow ที่ 2 : กรณีที่เจ้าหน้าที่ป้อนข้อมูลสำคัญไม่ครบถ้วน ระบบจะไม่สามารถบันทึกรายการลงทะเบียนเรียนได้ ดังนั้นระบบจะมีข้อความแจ้งเตือน “ป้อนข้อมูลสำคัญไม่ครบถ้วน กรุณากลับไปป้อนข้อมูลให้ครบ” ให้เจ้าหน้าที่ป้อนข้อมูลสำคัญให้ครบถ้วนระบบจึงจะสามารถบันทึกข้อมูลได้

Use Case Title : Checkout Course	**Use Case Id :** 2
Primary Actor : Registration Staff	
Stakeholder Actor : -	
Main Flow : การตรวจสอบรายวิชา หลังจากเจ้าหน้าที่ได้รับแบบฟอร์มลงทะเบียน และป้อนรหัสนักศึกษาได้อย่างถูกต้องแล้ว เจ้าหน้าที่ต้องป้อนรายวิชาที่นักศึกษาลงทะเบียน จากนั้นระบบจะนำรหัสวิชาที่ได้รับมาตรวจสอบรายวิชาที่ต้องผ่านมาก่อน โดยเมื่อทราบรายวิชาที่ต้องผ่านมาก่อนของวิชาที่นักศึกษาลงทะเบียนแล้ว ระบบจะเปรียบเทียบกับรายวิชาที่นักศึกษาเคยเรียนผ่านมาทั้งหมด เมื่อระบบตรวจสอบพบว่ารายวิชาที่นักศึกษาลงทะเบียนนั้นถูกต้อง ระบบจึงทำการคำนวณเงินค่าลงทะเบียนเป็นลำดับต่อไป	
Exceptional Flow ที่ 1 : กรณีการตรวจสอบรายวิชาที่ลงทะเบียนไม่ถูกต้อง เจ้าหน้าที่แจ้งนักศึกษาว่ามีบางรายวิชายังไม่สามารถลงทะเบียนได้เนื่องจากยังไม่ผ่านบางรายวิชามาก่อน ให้นักศึกษากลับไปแก้ไขแล้วนำใบลงทะเบียนมายื่นที่ฝ่ายทะเบียนอีกครั้งภายในระยะเวลาที่กำหนด	

Use Case Title : Record Billing	**Use Case Id :** 3
Primary Actor : Financial Staff	
Stakeholder Actor : Student	
Main Flow : เจ้าหน้าที่ฝ่ายการเงินรับสำเนาใบรับชำระเงินค่าลงทะเบียนจากนักศึกษา ป้อนรหัสนักศึกษาเพื่อให้ระบบแสดงข้อมูลการลงทะเบียน ตรวจสอบจำนวนเงินถูกต้องตรงกัน แล้วทำการบันทึกข้อมูลการชำระเงินและสถานการณ์ชำระเงิน ระบบแสดงข้อความยืนยันการบันทึกข้อมูลเรียบร้อยแล้ว	
Exceptional Flow ที่ 1 : กรณีเจ้าหน้าที่ป้อนรหัสนักศึกษาผิดพลาด ระบบจะแสดงข้อความแจ้งเตือน “ไม่พบข้อมูลนักศึกษา” เพื่อให้เจ้าหน้าที่ทราบว่ามีข้อผิดพลาดเกิดขึ้น และให้เจ้าหน้าที่ป้อนรหัสนักศึกษาใหม่อีกครั้ง	
Exceptional Flow ที่ 2 : กรณีจำนวนเงินที่นักศึกษาชำระไม่ตรงกับข้อมูลที่แสดงบนหน้าจอคอมพิวเตอร์ เจ้าหน้าที่จะต้องสอบถามนักศึกษา และแจ้งวันที่ชำระเงินงวดต่อไปให้นักศึกษาทราบ จากนั้นเจ้าหน้าที่บันทึกข้อมูลการชำระเงินและสถานะการชำระเงิน ระบบแสดงข้อความยืนยันการบันทึกข้อมูลเรียบร้อยแล้ว	
Exceptional Flow ที่ 3 : กรณีที่ระบบไม่สามารถบันทึกข้อมูลได้ เนื่องจาก เจ้าหน้าที่ฝ่ายการเงินป้อนข้อมูลการชำระเงินไม่ครบถ้วน ให้เจ้าหน้าที่ป้อนข้อมูลสำคัญให้ครบถ้วน แล้วบันทึกข้อมูลอีกครั้ง	