บทที่ 2
ทฤษฎีบทและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

งานวิจัยนี้ทฤษฎีที่เกี่ยวข้อง ได้แก่ แผนภาพคลาส แผนภาพซีเควนซ์ การจัดกลุ่มโอเปอเรชันในแผนภาพคลาสโดยพิจารณาจากการเรียกของโอเปอเรชัน การสร้างแอ็บสแตร็คแมชชีนบีโดยอาศัยเทคนิคการจำลองโอเปอเรชันที่เรียกและถูกเรียกระหว่างกันของคลาส และหลักการของภาษาบี โดยมีรายละเอียดดังนี้

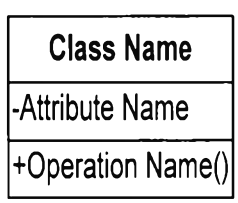
2.1.1 แผนภาพคลาส [4, 5, 6, 7]

เป็นแผนภาพพื้นฐานของระบบเชิงวัตถุ แผนภาพคลาสจะนำมาใช้ในการออกแบบโครงสร้างของระบบ โดยมีส่วนประกอบคือ คุณลักษณะ (Attribute) โอเปอเรชัน (Operation) ความสัมพันธ์ (Relation) และพฤติกรรม (Behavior) ของกลุ่มของออบเจ็ค

ในข้อกำหนดของยูเอ็มแอลรุ่นที่ 1.5 ได้กำหนดแผนภาพคลาสซึ่งประกอบด้วย 2 ส่วนหลัก ได้แก่

2.1.1.1 คลาส (Class)

อธิบายถึงชุดของออบเจ็คที่มีลักษณะคล้ายกันในเทอมของโครงสร้างและพฤติกรรม โดยสัญลักษณ์ของคลาสจะเขียนแทนด้วยรูปสี่เหลี่ยมผืนผ้า ซึ่งแบ่งส่วนประกอบของคลาสออกเป็น 3 ส่วน แสดงดังรูปที่ 2.1



รูปที่ 2.1 ส่วนประกอบของแผนภาพคลาส

จากรูปที่ 2.1 แสดงส่วนประกอบของแผนภาพคลาส ได้แก่

1) ชื่อของคลาส (Class's Name)

เป็นส่วนที่อยู่บนสุดของรูปสี่เหลี่ยมผืนผ้า แสดงด้วยตัวอักษรใช้เป็นคำนามแสดงชื่อของคลาส

2) คุณลักษณะ (Attribute)

เป็นส่วนที่อยู่ตรงกลางของรูปสี่เหลี่ยมผืนผ้า แสดงถึงคุณสมบัติของชุดของออบเจ็คที่อยู่ในคลาสเดียวกัน โดยมีส่วนประกอบต่าง ๆ ได้แก่

- ชนิดของการมองเห็นหรือการเข้าถึง (Visibility)

ได้แก่ ชนิด Private ชนิด Protected และชนิด Public เป็นต้น

- ชื่อของคุณลักษณะ (Attribute's Name)

จะแสดงถึงคุณลักษณะหรือคุณสมบัติที่มีของคลาสในแผนภาพคลาส

- ประเภทของคุณลักษณะ (Attribute's Type)

จะแสดงถึงชนิดข้อมูลของคุณลักษณะ ได้แก่ จำนวนเต็ม (Integer) สายอักขระ (String) และเปรียบเทียบเชิงตรรกะ หรือบูลีน (Boolean) เป็นต้น เพื่อทำให้คุณลักษณะมีรายละเอียดมากขึ้น

- ค่าเริ่มต้นของคุณลักษณะ (Initial Value)

เป็นค่าเริ่มต้นที่กำหนดให้กับคุณลักษณะแต่ละคุณลักษณะ โดยจะมีหรือไม่มีก็ได้

3) โอเปอเรชัน (Operation)

เป็นส่วนที่อยู่ล่างสุดของรูปสี่เหลี่ยมผืนผ้า เป็นส่วนที่แสดงพฤติกรรมของคลาส โดยมีส่วนประกอบต่าง ๆ ได้แก่

- ชนิดของการมองเห็นหรือการเข้าถึง (Visibility)

ได้แก่ ชนิด Private ชนิด Protected และชนิด Public เป็นต้น

- ชื่อของโอเปอเรชัน (Operation's Name)

แสดงชื่อโอเปอเรชันของคลาส โดยที่อธิบายคลาสว่ามีความรับผิดชอบในการทำงานใดหรือมีบริการใดบ้าง

- พารามิเตอร์ (Parameter)

เป็นตัวแปรที่รับหรือส่งของโอเปอเรชัน โดยแต่ละพารามิเตอร์ ประกอบด้วย ชื่อของพารามิเตอร์ (Parameter's Name) และชนิดของพารามิเตอร์ (Parameter's Type)

- ประเภทหรือชนิดค่าส่งคืนของโอเปอเรชัน (Return Type of Operations)

จะแสดงชนิดของโอเปอเรชันที่มีค่าส่งคืน หรือเป็น Primitive Data Type ของแต่ละโปรแกรม เช่นเดียวกับในส่วนของประเภทคุณลักษณะ

2.1.1.2 ความสัมพันธ์ (Relation)

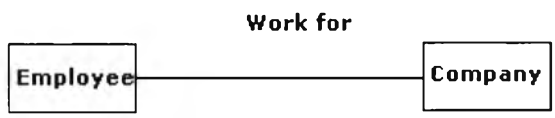
เป็นการติดต่อกันหรือสื่อสารกันระหว่างคลาส ซึ่งคลาสต่าง ๆ ในระบบสามารถมีความสัมพันธ์กันในรูปแบบต่าง ๆ โดยกลุ่มความสัมพันธ์หลักในแผนภาพคลาสมี 3 กลุ่มความสัมพันธ์ดังนี้

1) กลุ่มความสัมพันธ์แอสโซซิเอชัน (Association Relationship)

เป็นความสัมพันธ์ระหว่างคลาสแบบที่มีการติดต่อกันหรือการให้บริการกันโดยสามารถแบ่งออกเป็น 2 รูปแบบ คือ

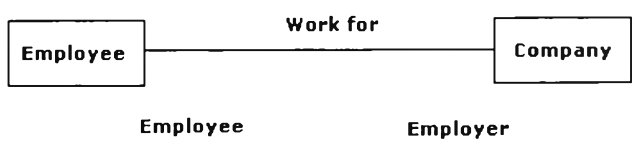
- ความสัมพันธ์แอสโซซิเอชัน (Association Relationship)

เป็นการอธิบายถึงความสัมพันธ์ของคลาสหนึ่งกับอีกคลาสหนึ่งในเชิงของกิจกรรม โดยความสัมพันธ์แอสโซซิเอชันจะอธิบายถึงการติดต่อกันระหว่างคลาส เช่น คลาส Employee มีความสัมพันธ์แอสโซซิเอชัน ชื่อว่า "Work for" กับคลาส Company หมายถึงมี Employee ใด ๆ จะสามารถทำงานให้ Company ใด ๆ แสดงได้ดังรูปที่ 2.2



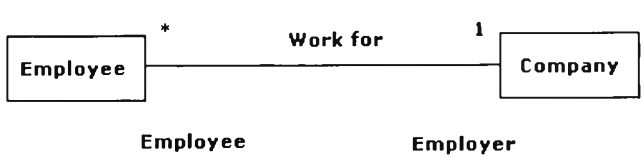
รูปที่ 2.2 ความสัมพันธ์แอสโซซิเอชันระหว่างคลาส Employee กับคลาส Company

ในบางกรณี ผู้วิเคราะห์ และออกแบบต้องการบรรยายรายละเอียดเพิ่มเติม สามารถทำการระบุบทบาท (Roles) ของแต่ละคลาสเพิ่มเติมได้ นั่นคือถ้าคลาสหนึ่งมีความสัมพันธ์กับอีกคลาสหนึ่ง แต่ละคลาสจะเกิดบทบาทระหว่างกัน โดยปกติบทบาทจะแนบมากับความสัมพันธ์ คลาสจะมีบทบาทเฉพาะในแต่ละคลาสที่มาสัมพันธ์กัน เช่น คลาส Employee มีบทบาทเป็นพนักงาน (Employee) ของคลาส Company และคลาส Company มีบทบาทเป็นนายจ้าง (Employer) ของคลาส Employee แสดงได้ดังรูปที่ 2.3



รูปที่ 2.3 บทบาทของคลาส Employee และคลาส Company

มัลติพลิซิตี (Multiplicity) เป็นการระบุถึงจำนวนออบเจ็คที่เกี่ยวข้องกันโดยมีความสัมพันธ์กับอีกออบเจ็คของอีกคลาสหนึ่ง มัลติพลิซิตีมีได้หลายรูปแบบ เช่น คลาสหนึ่งมีความสัมพันธ์กับคลาสอื่นแบบศูนย์ต่อหนึ่ง (Zero to One) หนึ่งต่อหลาย (One to Many) หลายต่อหนึ่ง (Many to One) และเพียงหนึ่ง (Exactly One) เป็นต้น ตัวอย่างของการใช้มัลติพลิซิตี แสดงได้ดังรูปที่ 2.4



รูปที่ 2.4 มัลติพลิซิตีระบุจำนวนของออบเจ็คที่มีได้ของความสัมพันธ์

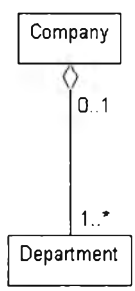
จากรูปที่ 2.4 แสดงตัวอย่างของการใช้มัลติพลิซิตีเพื่อระบุจำนวนออบเจ็คในความสัมพันธ์แอสโซซิเอชัน Work for ในระบบการทำงานในบริษัท นั่นคือ คลาส Employee แต่ละคนทำงานเพื่อหนึ่งคลาส Company

- ความสัมพันธ์แอกริเกชัน (Aggregation Relationship)

เป็นความสัมพันธ์ระหว่างคลาสหรือออบเจ็ค ที่แสดงในเชิงของการรวมกัน หรือประกอบกัน โดยสามารถแบ่งออกเป็น 2 รูปแบบ คือ

- ความสัมพันธ์แอกริเกชัน (Aggregation Relationship)

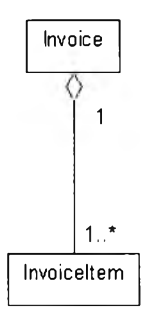
เป็นความสัมพันธ์ในเชิงของกระบวนการนำคลาสย่อย (Part Class) มารวมกัน หรือประกอบกันเป็นคลาสหลัก (Whole Class) แสดงได้ดังรูปที่ 2.5



รูปที่ 2.5 ตัวอย่างของความสัมพันธ์แอกริเกชัน

- ความสัมพันธ์คอมโพสิชัน (Composition Relationship)

เป็นความสัมพันธ์แบบแอกริเกชันรูปแบบหนึ่ง โดยจะเป็นความสัมพันธ์ในรูปแบบที่แข็งแรงกว่าความสัมพันธ์แอกริเกชัน โดยที่คลาสย่อยจะต้องมีในคลาสหลัก ถ้าคลาสหลักถูกทำลายจะทำให้คลาสย่อยถูกทำลายไปด้วย แสดงได้ดังรูปที่ 2.6

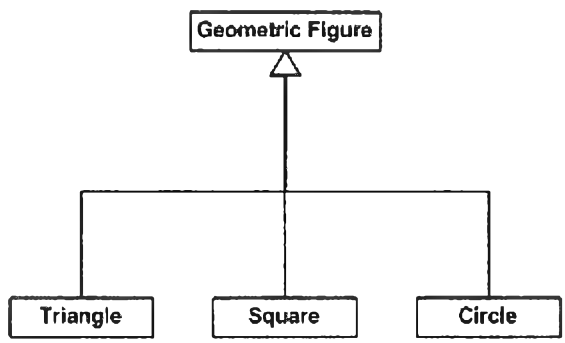


รูปที่ 2.6 ตัวอย่างของความสัมพันธ์คอมโพสิชัน

2) กลุ่มความสัมพันธ์เจเนอรัลไลเซชัน (Generalization Relationship)

เป็นความสัมพันธ์ที่มีการสืบทอดจากคุณลักษณะและโอเปอเรชันทั้งหมดจากคลาสหนึ่งที่เรียกว่าคลาสแม่ (Parent Class) หรือซูเปอร์คลาส (Super Class) ไปยังอีกคลาสหนึ่งที่เรียกว่าคลาส

ลูก (Child Class) หรือ ซับคลาส (Sub Class) ซึ่งในการเขียนคุณลักษณะ และโอเปอเรชันในแผนภาพคลาส โดยคุณลักษณะ และโอเปอเรชันที่ถูกสืบทอดจะไม่เขียนในซับคลาส เพราะได้แสดงคุณลักษณะ และโอเปอเรชันเหล่านั้นไว้ในซูเปอร์คลาสแล้ว ในกรณีที่ซับคลาสมีคุณสักษณะ และโอเปอเรชันเพิ่มขึ้น จะแสดงคุณลักษณะและโอเปอเรชันที่เพิ่มขึ้นเหล่านี้ในซับคลาส ถ้าไม่มีคุณลักษณะ และโอเปอเรชันที่เพิ่มขึ้นนี้ จะถือว่าซับคลาสมีลักษณะเช่นเดียวกันกับซูเปอร์คลาส เช่น คลาส Circle มีลักษณะรูปร่าง และการหาพื้นที่เฉพาะ ซึ่งเป็นคุณลักษณะที่ไม่มีในคลาส Geometric Figure ซึ่งเป็นซูเปอร์คลาส ตัวอย่างของความสัมพันธ์เจเนอรัลไลเซชัน แสดงได้ดังรูปที่ 2.7

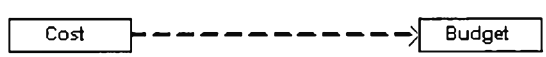


รูปที่ 2.7 ตัวอย่างของความสัมพันธ์เจเนอรัลไลเซชัน

คลาสที่ไม่มีซูเปอร์คลาส เรียกว่าคลาสราก (Root Class) ถ้าซับคลาสใดที่มีซูเปอร์คลาสเพียงคลาสเดียว เรียกว่าเป็นการสืบทอดเชิงเดี่ยว (Single Inheritance) ถ้าซับคลาสใดมีซูเปอร์คลาสมากกว่า 1 คลาส เรียกว่าเป็นการสืบทอดหลายเชิง (Multiple Inheritance)

3) กลุ่มความสัมพันธ์ดิเพนเดนซี (Dependency Relationship)

เป็นความสัมพันธ์ที่พึ่งพิง ที่ประกอบด้วยคลาสพึ่งพิง (Dependency Class) และคลาสที่ถูกพึ่งพิง (Independence Class) กล่าวคือ ถ้ามีการเปลี่ยนแปลงที่เกิดขึ้นกับคลาสที่ถูกพึ่งพิงจะส่งผลต่อคลาสพึ่งพิงด้วย เช่น คลาส Cost จะส่งผลต่อคลาส Budget แสดงได้ดังรูปที่ 2.8



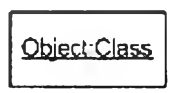
รูปที่ 2.8 แสดงความสัมพันธ์ดิเพนเดนซี

2.1.2 แผนภาพซีเควนซ์ [4, 5, 6, 7]

เป็นแผนภาพเชิงพลวัติที่แสดงถึงเหตุการณ์ที่แสดงลำดับการรับส่งข้อความหรือโอเปอเรชัน โดยที่ข้อความจะมีการรับส่งกันระหว่างวัตถุ ณ เวลาต่าง ๆ ทำให้สามารถเข้าใจข้อความในกรณีที่ระบบมีความซับซ้อน แผนภาพซีเควนซ์มีส่วนประกอบ 4 ส่วน ได้แก่

2.1.2.1 ออบเจ็ค (Object)

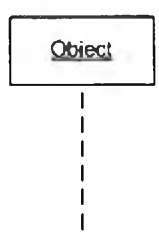
ประกอบไปด้วยชื่อของออบเจ็คที่เป็นชุดของคลาสที่มีการรับส่งข้อความในช่วงเวลาหนึ่ง โดยวางเรียงในตำแหน่งจากซ้ายไปขวาตามลำดับของข้อความในระบบ แทนด้วยรูปสี่เหลี่ยมผืนผ้า แสดงได้ดังรูปที่ 2.9



รูปที่ 2.9 ออบเจ็คในแผนภาพซีเควนซ์

2.1.2.2 เส้นชีวิต (Life Line)

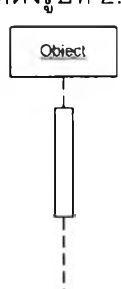
แสดงถึงช่วงชีวิตของออบเจ็ค ซึ่งแทนด้วยจุดที่เรียงในแนวดิ่งจากบนลงล่างของแต่ละออบเจ็ค แสดงได้ดังรูปที่ 2.10



รูปที่ 2.10 ช่วงชีวิตของออบเจ็คในแผนภาพซีเควนซ์

2.1.2.3 แอกทิเวชัน (Activation)

แสดงถึงช่วงของโอเปอเรชันต่าง ๆ ที่ออบเจ็คนั้นต้องกระทำ โดยที่ความยาวแอกทิเวชันจะบอกถึงช่วงเวลาที่โอเปอเรชันของออบเจ็คในการรับส่งนั้น แทนด้วยรูปสี่เหลี่ยมผืนผ้ามีลักษณะแคบๆที่วางทับเส้นชีวิต แสดงได้ดังรูปที่ 2.11



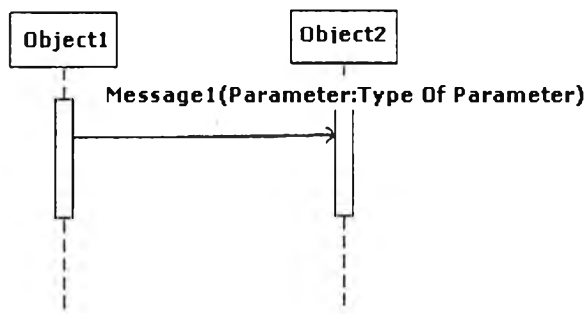
รูปที่ 2.11 ช่วงเวลาที่โอเปอเรชันของออบเจ็คต้องกระทำในแผนภาพซีเควนซ์

2.1.2.4 ข้อความ (Message)

แสดงการลำเลียงของโอเปอเรชันที่ส่งจากออบเจ็คหนึ่งไปยังอีกออบเจ็คหนึ่ง แทนด้วยลูกศร โดยมีชื่อของโอเปอเรชันและชนิดค่าส่งคืนของโอเปอเรชันกำกับบนลูกศร ซึ่งอาจจะมีพารามิเตอร์และชนิดของพารามิเตอร์ของโอเปอเรชันที่เหมาะสมอยู่ในวงเล็บที่วางถัดจากชื่อของโอเปอเรชัน แสดงได้ดังรูปที่ 2.12

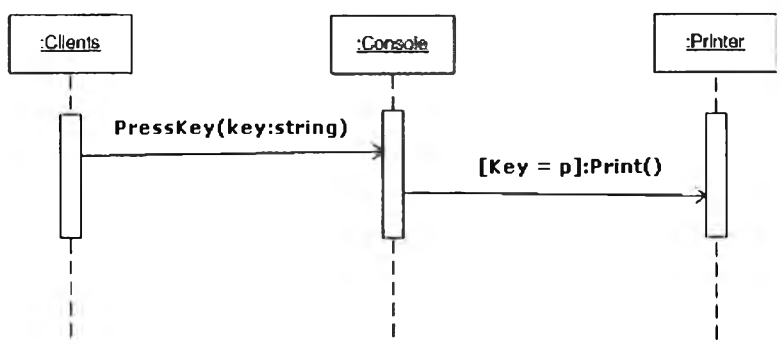
รูปที่ 2.12 ข้อความที่ส่งและรับกันระหว่างออบเจ็คในแผนภาพซีเควนซ์

รูปแบบของเหตุการณ์การรับส่งข้อความจากแผนภาพซีเควนซ์ แสดงได้ดังรูปที่ 2.13



รูปที่ 2.13 รูปแบบของเหตุการณ์การรับส่งข้อความจากแผนภาพซีเควนซ์

ตัวอย่างของเหตุการณ์การพิมพ์งานออกทางเครื่องพิมพ์ จากแผนภาพซีเควนซ์ แสดงได้ดังรูปที่ 2.14



รูปที่ 2.14 ตัวอย่างของเหตุการณ์จากแผนภาพซีเควนซ์

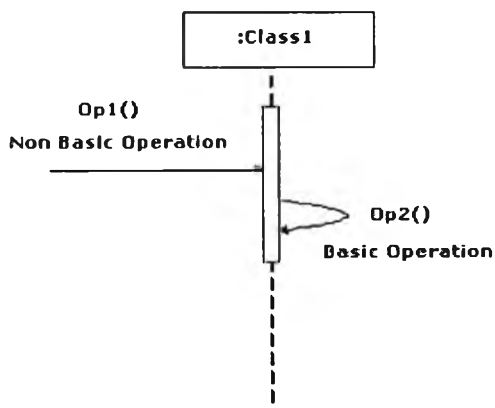
2.1.3 การจัดกลุ่มโอเปอเรชันในแผนภาพคลาสโดยพิจารณาจากการเรียกของโอเปอเรชัน [9, 10]

ในแผนภาพคลาส โดยที่คลาสหนึ่ง ๆ จะมีโอเปอเรชันจำนวนมากอาจจะยากต่อการนำไประบุในแอ็บสแตร็คแมชชีนบีได้โดยตรง จากการจัดกลุ่มโอเปอเรชันในแผนภาพคลาสโดยพิจารณาจากการ

เรียกของโอเปอเรชัน โดยงานวิจัยของฮง เลอดง [9, 10] สามารถแบ่งโอเปอเรชันโดยพิจารณาจากการเรียกโอเปอเรชันอื่น ๆ ของคลาส จากเหตุการณ์ของการรับส่งข้อความของแผนภาพซีเควนซ์ ออกเป็น 2 กลุ่ม ได้แก่

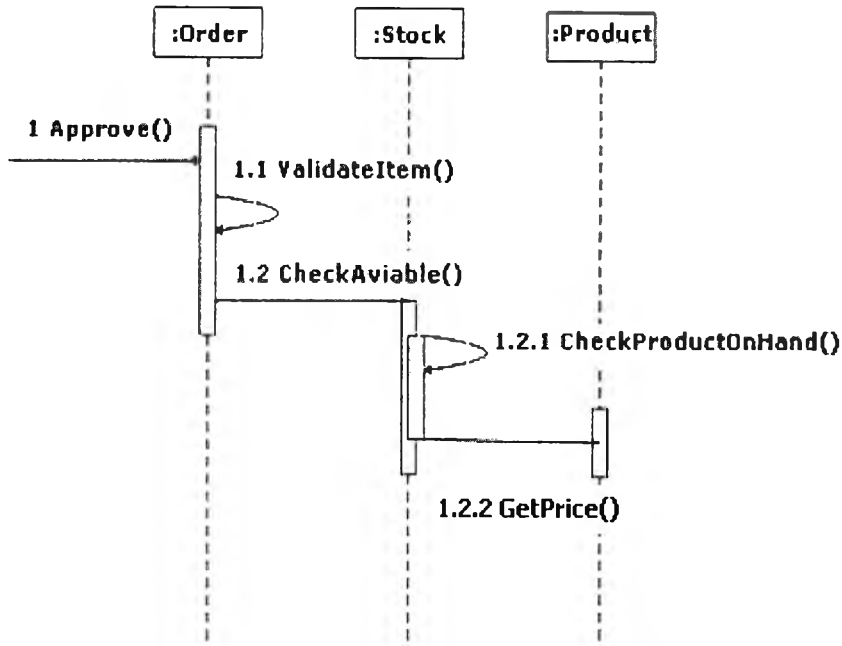
1) กลุ่มของโอเปอเรชันนอนเบซิค (Non Basic Operation)
   หมายถึง โอเปอเรชันที่มีการเรียกโอเปอเรชันอื่น ๆ ของคลาสเองหรือโอเปอเรชันอื่น ๆ ของคลาสอื่น ๆ อีกต่อหนึ่งได้
2) กลุ่มของโอเปอเรชันเบซิค (Basic Operation)
   หมายถึง โอเปอเรชันที่ไม่มีการเรียกโอเปอเรชันของคลาสใด ๆ ได้อีก

รูปแบบของโอเปอเรชันนอนเบซิค และโอเปอเรชันเบซิคในแผนภาพซีเควนซ์ แสดงได้ดังรูปที่ 2.15 โอเปอเรชัน Op1() จะสามารถทำงานได้ตามเป้าหมายนั้น จะต้องมีการสั่งให้ Class1 ทำโอเปอเรชัน Op2() นั่นคือ โอเปอเรชัน Op1() ของ Class1 จะทำการเรียกโอเปอเรชัน Op2() ของ Class1 ให้ทำงาน พบว่า โอเปอเรชัน Op1() เป็นโอเปอเรชันนอนเบซิค ส่วนโอเปอเรชัน Op2() เป็นโอเปอเรชันเบซิค เพราะว่าโอเปอเรชัน Op2() จะเป็นโอเปอเรชันที่ถูกเรียกโดยโอเปอเรชัน Op1() จึงสามารถทำงานตามเป้าหมายได้



รูปที่ 2.15 ตัวอย่างของโอเปอเรชันนอนเบซิคและโอเปอเรชันเบซิคจากแผนภาพซีเควนซ์

ตัวอย่างของเหตุการณ์การสั่งซื้อสินค้าจากร้านค้าจากแผนภาพซีเควนซ์ แสดงได้ดังรูปที่ 2.16 สามารถพิจารณาโอเปอเรชันในแผนภาพซีเควนซ์ ดังต่อไปนี้



รูปที่ 2.16 ตัวอย่างของเหตุการณ์การสั่งซื้อสินค้าจากร้านค้าจากแผนภาพซีเควนซ์

1) กำหนดให้โอเปอเรชัน Approve() เป็นโอเปอเรชันนอนเบซิค

เพราะว่าโอเปอเรชัน Approve() ของคลาส Order จำเป็นต้องสั่งให้คลาส Order ทำการ ValidateItem()โดยการเรียกโอเปอเรชัน ValidateItem() ของคลาส Order พร้อมกันนั้นจึงสั่งให้คลาส Stock ทำการ CheckAvailable() โดยการเรียกโอเปอเรชัน CheckAviable() ของคลาส Stock ดังนั้น โอเปอเรชัน Approve() จึงเป็นโอเปอเรชันนอนเบซิค

2) กำหนดให้โอเปอเรชัน ValidateItem() เป็นโอเปอเรชันเบซิค

เพราะว่าโอเปอเรชัน ValidateItem() จะถูกเรียกเฉพาะภายในคลาส Order เท่านั้น โดยที่โอเปอเรชัน ValidateItem() ไม่ได้ทำการเรียกโอเปอเรชันอื่น ๆ ดังนั้นโอเปอเรชัน ValidateItem() จึงเป็นโอเปอเรชันเบซิค

3) กำหนดให้โอเปอเรชัน CheckAvailable() เป็นโอเปอเรชันนอนเบซิค

เพราะว่าโอเปอเรชัน CheckAvailable() จำเป็นต้องสั่งให้คลาส Stock ทำการ CheckProductOnHand() โดยการเรียกโอเปอเรชัน CheckProductOnHand() ของคลาส Stock และในขณะเดียวกันนั้นจะสั่งให้คลาส Product ทำการ GetPrice() โดยการเรียกโอเปอเรชัน GetPrice() ของคลาส Product ดังนั้น CheckAvailable() เป็นโอเปอเรชันนอนเบซิค

4) กำหนดให้โอเปอเรชัน CheckProductOnHand() เป็นโอเปอเรชันเบซิค

เพราะว่าโอเปอเรชัน CheckProductOnHand() ถูกเรียกเฉพาะภายในในคลาส Stock เท่านั้นซึ่งไม่ได้เรียกโอเปอเรชันของคลาสอื่น ๆ ดังนั้นโอเปอเรชัน CheckProductOnHand() จึงเป็นโอเปอเรชันเบซิค

5) กำหนดให้โอเปอเรชัน GetPrice() เป็นโอเปอเรชันเบซิค

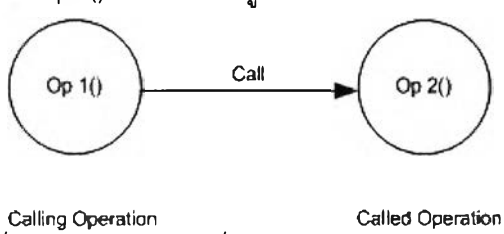
เพราะว่าโอเปอเรชัน GetPrice() ถูกสั่งโดยคลาส Product ให้ทำการ GetPrice() โดยคลาส Stock เท่านั้น นั่นคือ โอเปอเรชัน GetPrice() ไม่ได้ทำการเรียกโอเปอเรชันอื่น ๆ ดังนั้น GetPrice() จึงเป็นโอเปอเรชันเบซิค

2.1.4 การสร้างแอ็บสแตร็คแมซชีนบีจากเทคนิคการจำลองโอเปอเรชันที่เรียกและถูกเรียกระหว่างกันของคลาส [9, 10]

การสร้างแอ็บสแตร็คแมซชีนบีที่ระบุโอเปอเรชันของคลาสที่แสดงในเหตุการณ์ของการรับส่งข้อความจากแผนภาพซีเควนซ์นั้น จะใช้หลักการจัดการโอเปอเรชันของคลาสที่ขึ้นต่อกันโดยการเรียกและถูกเรียก ซึ่งเป็นส่วนหนึ่งของเทคนิคการจำลองโอเปอเรชันที่เรียก และถูกเรียกระหว่างกันของคลาส (Modeling the Calling – Called Dependency between Class Operations) นั่นคือ

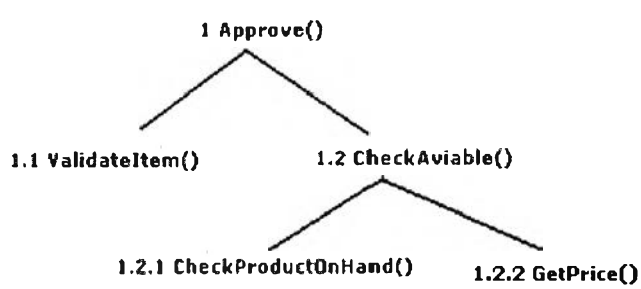
หลักการจัดการโอเปอเรชันของคลาสที่ขึ้นต่อกันโดยการเรียกและถูกเรียก (Calling – Called between Class Operation Dependency)

โอเปอเรชันที่เรียก (Calling Operation) หรือ Op1() จะทำการเรียก (Call) โอเปอเรชันที่ถูกเรียก (Called Operation) หรือ Op2() แสดงได้ดังรูปที่ 2.17



รูปที่ 2.17 โอเปอเรชันที่เรียก และโอเปอเรชันถูกเรียก

จากตัวอย่างของเหตุการณ์การสั่งซื้อสินค้าจากร้านค้าจากแผนภาพซีเควนซ์ จากรูปที่ 2.16 จะสามารถนำโอเปอเรชันทั้งหมดในแผนภาพซีเควนซ์มาทำการจัดโอเปอเรชันเป็นชั้นตามลักษณะของต้นไม้ โดยอาศัยเทคนิคการจำลองโอเปอเรชันที่เรียก และถูกเรียกระหว่างกันของคลาส แสดงได้ดังรูปที่ 2.18



รูปที่ 2.18 การจัดโอเปอเรชันทั้งหมดในแผนภาพซีเควนซ์ออกเป็นชั้นตามลักษณะของต้นไม้

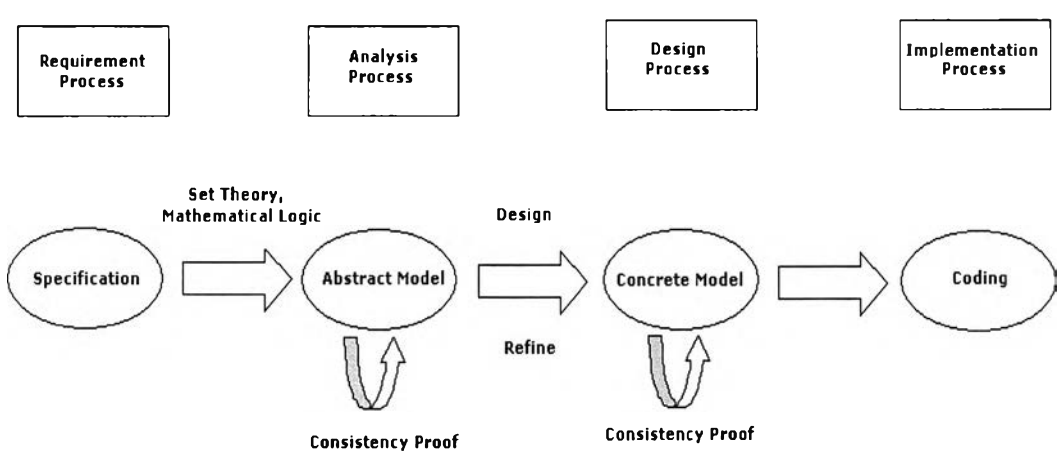
จากรูปที่ 2.18 จะแสดงการนำโอเปอเรชันทั้งหมดของเหตุการณ์การสั่งซื้อสินค้าจากร้านค้าจากแผนภาพซีเควนซ์มาทำการจัดเป็นชั้นตามลักษณะของแผนภาพต้นไม้ ซึ่งโอเปอเรชัน Approve() จะเป็นโอเปอเรชันนอนเบซิคจะทำการเรียกโอเปอเรชัน ValidateItem() ซึ่งเป็นโอเปอเรชันเบซิค และเรียกโอเปอเรชัน CheckAvailable() ซึ่งเป็นโอเปอเรชันนอนเบซิค โดยโอเปอเรชัน CheckAvailable() จะทำการเรียกโอเปอเรชัน CheckProductOnHand() ซึ่งเป็นโอเปอเรชันเบซิค และเรียกโอเปอเรชัน GetPrice() ซึ่งเป็นโอเปอเรชันเบซิค

พบว่าโอเปอเรชัน Approve() เป็นโอเปอเรชันที่เรียกโอเปอเรชัน ValidateItem() โอเปอเรชัน CheckProductOnHand และโอเปอเรชัน GetPrice() เป็นโอเปอเรชันที่ถูกเรียก ส่วนโอเปอเรชัน CheckAviable() เป็นทั้งโอเปอเรชันที่เรียกและถูกเรียก โดยโอเปอเรชัน CheckAviable() เป็นโอเปอเรชันที่ทำการเรียกโอเปอเรชัน CheckProductOnHand() และโอเปอเรชัน GetPrice() ในทำนองเดียวกันโอเปอเรชัน CheckAviable() เป็นโอเปอเรชันที่ถูกเรียกโดยถูกเรียกจากโอเปอเรชัน Approve() ด้วย

2.1.5 หลักการของภาษาบี (B Methods) [2, 13]

เป็นหลักการนำแบบจำลองข้อกำหนดรูปนัยภาษาบีที่สามารถนำไปใช้งานในกระบวนการพัฒนาซอฟต์แวร์ตั้งแต่กระบวนการวิเคราะห์ความต้องการของระบบจนถึงกระบวนการสร้างโปรแกรมของระบบซึ่งได้รับการพัฒนาโดย เจ อาร์ อาเบรียล (J.R Abrial) ในปี ค.ศ. 1980

กระบวนการพัฒนาซอฟต์แวร์ภายใต้ข้อกำหนดรูปนัยของภาษาบีมีกระบวนการดังรูปที่ 2.19



รูปที่ 2.19 กระบวนการพัฒนาซอฟต์แวร์โดยใช้ข้อกำหนดรูปนัยของภาษาบี

จากรูปที่ 2.19 แสดงขั้นตอนกระบวนการพัฒนาซอฟต์แวร์ภายใต้ข้อกำหนดของภาษาบี โดยเริ่มจาก

1. กระบวนการวิเคราะห์ความต้องการ (Requirement Process)

จะทำการเขียนข้อกำหนดของความต้องการโดยใช้สัญลักษณ์ทางคณิตศาสตร์เพื่อนำเสนอความต้องการจากกระบวนการวิเคราะห์ความต้องการให้อยู่ในรูปของโมเดลแอ็บสแตร็ค (Abstract Model) ในกระบวนการวิเคราะห์ (Analysis Process)

2. กระบวนการวิเคราะห์ (Analysis Process)

จะทำการตรวจสอบความสอดคล้องของความต้องการจากกระบวนการวิเคราะห์ในรูปแบบของโมเดลแอ็บสแตร็ค จากนั้นทำการออกแบบความต้องการจากกระบวนการวิเคราะห์ในรูปแบบของโมเดลคอนครีต (Concrete Model) ในกระบวนการออกแบบ (Design Process)

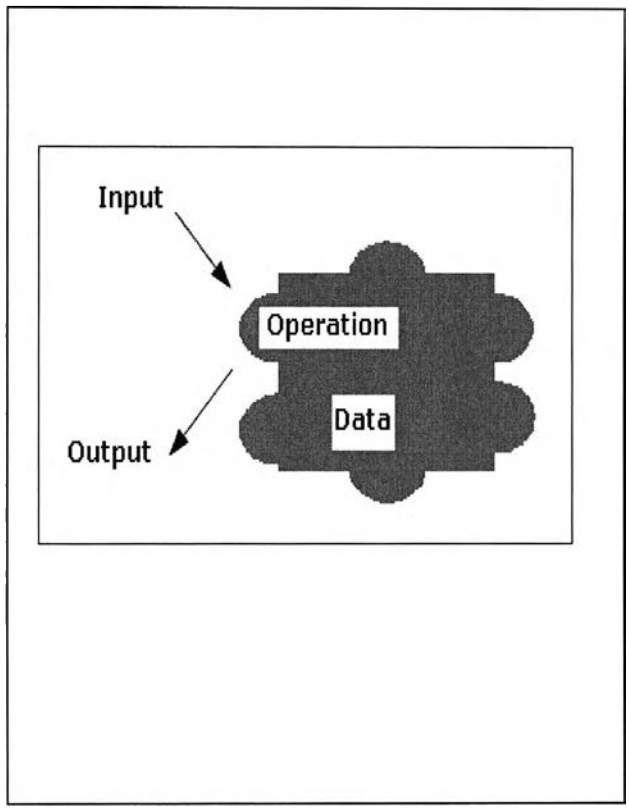
3. กระบวนการออกแบบ (Design Process)

จะทำการตรวจสอบความสอดคล้องของความต้องการจากกระบวนการวิเคราะห์ในรูปของโมเดลคอนครีต ซึ่งในระหว่างรูปแบบของโมเดลแอ็บสแตร็คและรูปแบบของโมเดลคอนครีตจะมีการออกแบบและการเพิ่มรายละเอียดของการออกแบบหรือการรีไฟน์เมนต์ (Refinement) จากรูปของโมเดลแอ็บสแตร็คไปสู่โมเดลคอนครีต โดยที่รูปของโมเดลคอนครีตยังคงความหมาย และความถูกต้องเหมือนรูปแบบของโมเดลแอ็บสแตร็คเช่นเดิม

4. กระบวนการสร้างโปรแกรม (Coding Process)

เป็นขั้นตอนสุดท้ายของกระบวนการพัฒนาซอฟต์แวร์ภายใต้ข้อกำหนดของภาษาบี โดยสามารถทำการพัฒนาเป็นโปรแกรมของภาษาได้

การเขียนข้อกำหนดของภาษาบีได้นั้น จะอาศัยใช้สัญลักษณ์ทางทฤษฎีเซต (Set Theory) และคณิตตรรกศาสตร์ (Mathematical Logic) สามารถนำเสนอข้อกำหนดของภาษาบีในรูปแบบของสัญกรณ์แอ็บสแตร็คแมชชีนบี (B Abstract Machine - BAM) โดยแนวคิดของ BAM จะสอดคล้องกับแนวคิดของหลักการเชิงวัตถุ นั่นคือ 1 BAM ถือว่าเป็น 1 โมดูล (Module) แต่ละโมดูลจะเป็นอิสระต่อกันและภายในโมดูลจะมีการปกปิดข้อมูล (Encapsulation) ทำให้สามารถนำมาพัฒนากับระบบที่มีความซับซ้อนได้ โครงสร้างของแอ็บสแตร็คแมชชีนบีแสดงได้ดังรูปที่ 2.20



MACHINE
IMPLEMENTATION
REFINES
CONSTRAINTS
USES
SEES
INCLUDES
PROMOTES
EXTENDS
SETS
CONSTANTS
PROPERTIES
VARIABLES
INVARIANT
ASSERTION
DEFINITIONS
INITIALISATION
OPERATIONS
END

(a) (b)

รูปที่ 2.20 โครงสร้างของแอ็บสแตร็คแมชชีนบี

(a) การซ่อนข้อมูล (Encapsulation) และ (b) อนุประโยคที่ใช้ในแอ็บสแตร็คแมชชีนบี

อนุประโยคหลัก ๆ ของแอ็บสแตร็คแมชชีนบี ได้แก่

1) MACHINE เป็นอนุประโยคในแอ็บสแตร็คแมชชีนบี ที่ระบุชื่อของแอ็บสแตร็คแมชชีนบี

2) SETS เป็นอนุประโยคในแอ็บสแตร็คแมชชีนบี ทำให้แอ็บสแตร็คแมชชีนบีสามารถนิยามเซตที่อ้างถึงชุดของตัวแปรสเตท (State Variables) ได้

3) SEES เป็นอนุประโยคในแอ็บสแตร็คแมชชีนบี ที่ทำให้แอ็บสแตร็คแมชชีนบีสามารถมองเห็น หรืออ่านค่าของตัวแปรสเตท แต่ไม่สามารถเปลี่ยนแปลงค่าของตัวแปรสเตทนั้นผ่านทางอนุประโยค OPERATIONS นั่นคือในอนุประโยค INVARIANT ของแอ็บสแตร็คแมชชีนบีที่ SEES จะไม่สามารถอ้างถึงตัวแปรสเตทของแอ็บสแตร็คแมชชีนบีที่ถูก SEES

4) USES เป็นอนุประโยคในแอ็บสแตร็คแมชชีนบี ที่ทำให้แอ็บสแตร็คแมชชีนบีนั้นสามารถเห็น หรืออ่านค่าของตัวแปรสเตท แต่ไม่สามารถเปลี่ยนแปลงค่าตัวแปรสเตทผ่านทางอนุประโยค OPERATIONS เช่นเดียวกับ SEES แต่ USES ต่างจาก SEES คือ อนุประโยค

INVARIANT ของแอ็บสแตร็คแมชชีนบีที่ USES จะสามารถอ้างถึงตัวแปรสเตทของแอ็บสแตร็คแมชชีนบีที่ถูก USES ได้

5) VARIABLES เป็นอนุประโยคในแอ็บสแตร็คแมชชีนบีที่สามารถเก็บค่าของตัวแปรสเตทที่จำเป็นทั้งหมดในแอ็บสแตร็คแมชชีนบี

6) INVARIANT เป็นอนุประโยคที่ระบุสัจพจน์ และเงื่อนไขที่เป็นจริงในแอ็บสแตร็คแมชชีนบี โดยจะแสดงความสัมพันธ์ที่เกี่ยวข้องระหว่างตัวแปรสเตทกับชนิดของตัวแปรสเตทนั้นต้องเป็นจริงเสมอ

7) INITIALIZATIONS เป็นการกำหนดสถานะเริ่มต้นการทำงานให้กับตัวแปรสเตทในแอ็บสแตร็คแมชชีนนั้น ๆ โดยการกำหนดค่าเริ่มต้นของตัวแปรสเตทในแอ็บสแตร็คแมชชีนบีนั้น จะต้องสอดคล้องกับอนุประโยค INVARIANT

8) OPERATIONS เป็นอนุประโยคที่อธิบายโอเปอเรชันและเงื่อนไขการทำงานของโอเปอเรชันในแอ็บสแตร็คแมชชีนบี โดยโอเปอเรชันจะทำการรับข้อมูลนำเข้าและส่งข้อมูลออก ผลการเปลี่ยนแปลงจะเกิดขึ้นภายในแอ็บสแตร็คแมชชีนบี ซึ่งโอเปอเรชันของแมชชีนจะต้องสอดคล้องสัจพจน์ และเงื่อนไขที่เป็นจริงในอนุประโยค INVARIANT ตลอดของการทำงานของโอเปอเรชันภายในแอ็บสแตร็คแมชชีนบี

9) IMPLEMENTATIONS เป็นการรีไฟน์เมนต์รูปแบบหนึ่ง (Refinement) ที่สามารถนำแอ็บสแตร็คแมชชีนบีที่ทำการอิมพลีเมนต์แล้วมาพัฒนาเป็นโค้ด (Code) ของโปรแกรมภาษาได้แก่ ภาษาซี (C Language) ภาษาซีพลัซพลัซ (C++ Language) และเอดา (Ada) ที่สามารถนำไปเอ็กซีคิวส์ (Execute) ได้ การอิมพลิเมนเทชันแอ็บสแตร็คแมชชีนบี โดยมีหลักการคือ IMPLEMENTATIONS เป็นการอิมพลีเมนต์เทชันแอ็บสแตร็คแมชชีนบี เพื่อทำการรีไฟน์เมนต์แอ็บสแตร็คแมชชีนบีในอนุประโยครีไฟน์ (REFINES) โดยแสดงชื่อของแอ็บสแตร็คแมชชีนบีที่จะทำการอิมพลีเมนต์ในอนุประโยครีไฟน์ ซึ่งในแอ็บสแตร็คแมชชีนบีจะมีโอเปอเรชันที่เรียกเพื่อแสดงการเรียกโอเปอเรชันที่ถูกเรียกอื่น ๆ ของแอ็บสแตร็คแมชชีนบีโดยการใช้อนุประโยค SEES โดยแอ็บสแตร็คแมชชีนบีในอนุประโยค SEES จะต้องมีโอเปอเรชันที่ถูกเรียก

รูปแบบของการอิมพลิเมนเทชันแอ็บสแตร็คแมชชีนบี สามารถแสดงได้ดังรูปที่ 2.21

```
IMPLEMENTATIONS
B Abstract Machine_imp
REFINES
B Abstract Machine have Calling Operations
SEES
B Abstract Machine have Called Operations
OPERATIONS
        Return Value of Calling Operations ← Calling Operation (Parameter) =
        VAR
        Return Value of Called Operations
        IN
        Return Value of Called Operations ← Called Operation (Parameter) =
        END

END
```

รูปที่ 2.21 โครงสร้างของอิมพลิเมนต์เทชันแอ็บสแตร็คแมชชีนบี

2.2 งานวิจัยที่เกี่ยวข้อง

งานวิจัยที่เกี่ยวข้องกับการแปลงแผนภาพคลาส และแผนภาพซีเควนซ์ของยูเอ็มแอลไปเป็นแอ็บสแตร็คแมชชีนบี มีรายละเอียดดังนี้

2.2.1 ระบบการแปลงแผนภาพแบบจำลองเชิงเทคนิคของวัตถุไปเป็นข้อกำหนดของภาษาบี (A Sysmetic Approach to Transformation Diagram to a B Specification) [8] โดย Eric Meyer และ Jeanine Souquières

งานวิจัยนี้นำเสนอระบบการแปลงของแผนภาพแบบจำลองเชิงเทคนิค (OMT – Object Modeling Technique) ไปเป็นข้อกำหนดของภาษาบี (B Formal Specification) โดยงานวิจัยนี้ได้นำเสนอการแปลงแบบจำลองเชิงวัตถุ (Object Model) ได้แก่ คลาส และความสัมพันธ์ และแบบจำลองเชิงพลวัติ (Dynamic Model) ได้แก่ การเปลี่ยนแปลงของสถานะ (Transition) ไปเป็นข้อกำหนดรูปนัยของภาษาบี และทำการพิสูจน์อินแวร์เรียน (Invariant) ได้นำบีทูลคิต ซึ่งเป็นเครื่องมือมาช่วยในการพิสูจน์ความถูกต้อง โดยอาศัยหลักการกำหนดหัวข้อในการพิสูจน์ (Proof Obligation) และการพิสูจน์โดยอัตโนมัติ (Auto Proof)

2.2.2 เทคนิคการรวมยูเอ็มแอลและข้อกำหนดของภาษาบี (Integrating UML and B Specification Techniques) [9] โดย Hung Ledang และ Jeanine Souquières

งานวิจัยนี้นำเสนอวิธีการแปลงความต้องการจากแผนภาพยูเอ็มแอล ได้แก่ แผนภาพคลาส และแผนภาพคลอแลบอเรชัน (Collaboration Diagram) เป็นข้อกำหนดรูปนัยของภาษาบีโดยใช้กระบวนการ 2 กระบวนการ คือ กระบวนการแบ่ง (Division Procedure) และกระบวนการจำลอง (Duplication Procedure) เป็นเทคนิคในการจัดตำแหน่งของการโอเปอเรชันของคลาสออกเป็นชั้น 3

ชั้น ได้แก่ ชั้นบนสุด (Top Layer) ชั้นกลาง (Intermediate Layer) และชั้นล่างสุด (Bottom Layer) แล้วนำแต่ละชั้นมาทำการแปลงเป็นแอ็บสแตร็คแมชชีนบีทำให้สามารถได้รายละเอียดของข้อกำหนดของภาษาบีจากแผนภาพยูเอ็มแอลที่สามารถอธิบายความหมายของระบบในเชิงโครงสร้าง และความหมายของระบบในเชิงพฤติกรรม

2.2.3 **การจำลองโอเปอเรชันของคลาสในข้อกำหนดของภาษาบี โดยใช้การทดลองจากกรณีศึกษาของส่วนประกอบของปั๊ม** (Modeling class operation in B: a case study on the pump component) **[10] โดย** Hung Ledang **และ** Jeanine Souquières

งานวิจัยนี้เป็นงานวิจัยที่ต่อเนื่องมาจากงานวิจัยเรื่องเทคนิคการรวมยูเอ็มแอลและข้อกำหนดของภาษาบี (Integrating UML and B Specification Techniques) โดย Hung Ledang และ Jeanine Souquières โดยงานวิจัยนี้ได้นำเสนอการจำลองโอเปอเรชันของแผนภาพคลาสและแผนภาพคลอแลบอเรชัน แล้วทำการแปลงเป็นแอ็บสแตร็คแมชชีนบี ซึ่งทำการทดลองกับกรณีศึกษาของส่วนประกอบต่างๆของปั๊ม

2.2.4 **เครื่องมือสนับสนุนการใช้ยูเอ็มแอลสำหรับพัฒนาโครงงานข้อกำหนดของภาษาบี** (Tools – Supported Use of UML for Developing B Project) **[11] โดย** Colin Snook **และ** Michael Butler

งานวิจัยนี้นำเสนอเครื่องมือที่ช่วยในการแปลงแผนภาพคลาสไปเป็นข้อกำหนดรูปนัยของภาษาบี โดยเครื่องมือนี้เรียกว่า ยูทูบี (U2B Class Diagram Translater) ที่ช่วยในการแปลงแผนภาพคลาสที่ถูกสร้างโดยโปรแกรมเรชันนัลโรซ ไปเป็นข้อกำหนดของภาษาบี ซึ่ง U2B เป็นสคริปต์ไฟล์ (Script File) ที่สามารถรัน (Run) โดยโปรแกรมเรชันนัลโรซ และสามารถแปลงแบบแผนภาพที่กำลังเปิดอยู่ขณะนั้นไปเป็นแอ็บสแตร็คแมชชีนบีได้ ในงานวิจัยนี้ได้นำเสนอตัวอย่างจากกรณีศึกษาจากระบบจราจรบนถนน

2.2.5 **การแปลงรูปแบบที่เหมาะสมที่อยู่บนพื้นฐานของยูเอ็มแอลและข้อกำหนดของภาษาบี** (Formalizing Pattern Applicability – An Approach based on UML and B) **[12] โดย** Rafael Marcano-Kamenoff

งานวิจัยนี้นำเสนอการแปลงแผนภาพยูเอ็มแอลซึ่งเป็นข้อกำหนดกึ่งรูปนัยไปเป็นข้อกำหนดรูปนัยของภาษาบี ซึ่งได้ทำการทดลองโดยใช้หลักการหาความสัมพันธ์ของกฎการแปลงแผนภาพยูเอ็มแอลไปเป็นข้อกำหนดของภาษาบี ซึ่งได้พยายามหารูปแบบที่เหมาะสมเพื่อช่วยสนับสนุนความสัมพันธ์ของกฎการแปลง และได้ทำการทดลองโดยใช้ตัวอย่างของกรณีศึกษาของแบบจำลองของระบบการกระจาย (Distributed System) กับระบบการรับและการให้บริการ (Client-Server)