เครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยวีคมิวเทชัน

นายชาตรี งามเบญจวงศ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2562

A Weak Mutation Testing Tool for BPMN

Mr. Chatri Ngambenchawong

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Software Engineering
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2019

หัวข้อวิทยานิพนธ์	เครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยวีคมิวเทชัน
โดย	นายชาตรี งามเบญจวงศ์
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์)

..... กรรมการ
(รองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ทองทักษ์)

..... กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร.บัณฑิต ฐานะโสภณ)

ชาตรี งามเบญจวงศ์ : เครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยวีคมิวเทชัน. (A Weak Mutation Testing Tool for BPMN) อ.ที่ปรึกษาหลัก : รศ. ดร.ธาราทิพย์ สุวรรณศาสตร์

แบบจำลองบีพีเอ็มเอ็นเป็นแบบจำลองที่ใช้สำหรับอธิบายกระบวนการทางธุรกิจ และถูกพัฒนาให้สามารถประมวลผลได้บนเครื่องประมวลผลแบบจำลอง จึงจำเป็นต้องทดสอบความถูกต้องของแบบจำลองตามเงื่อนไขทางธุรกิจ นักทดสอบจึงได้นำเสนอวิธีการสร้างกรณีทดสอบที่แตกต่างกัน รวมถึงใช้การทดสอบมิวเทชัน เพื่อประเมินคุณภาพของกรณีทดสอบ แต่จุดด้อยของการทดสอบมิวเทชันยังต้องใช้แรงงาน และทรัพยากรที่ค่อนข้างสูง

งานวิจัยนี้ได้นำเสนอการนำตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นมาประยุกต์กับการทดสอบวีคมิวเทชัน และยังเสนอเครื่องมือวีมูบีพีเอ็มเอ็นที่ใช้ในการสร้างมิวแตนท์ และทดสอบมิวแตนท์ได้อย่างอัตโนมัติกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น โดยเครื่องมือสามารถรายงานผลลัพธ์ ได้แก่ จำนวนมิวแตนท์ที่กำจัดได้ จำนวนมิวแตนท์ที่ยังคงอยู่ เวลาที่ใช้ในการทดสอบ คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ หลังจากผู้วิจัยได้ทดสอบเครื่องมือกับแบบจำลองตัวอย่างทั้งหมด 8 แบบจำลอง เครื่องมือสามารถสร้างมิวแตนท์ได้ครบถ้วน 25 ตัวดำเนินการ และทดสอบกับเครื่องประมวลผลแบบจำลองได้ทั้งหมด 13 ตัวดำเนินการ

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อนิสิต .....

ปีการศึกษา 2562 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

5971005021 : MAJOR SOFTWARE ENGINEERING

KEYWORD: BPMN, BPMN Engine, Mutation Testing, Weak Mutation

Chatri Ngambenchawong : A Weak Mutation Testing Tool for BPMN. Advisor: Assoc. Prof. Taratip Suwannasart, Ph.D.

Business Process Model and Notation (BPMN) is a model that describes a business process and is developed for processing on a BPMN Engine. It is necessary to verify the correctness of the process. Therefore, there are some proposed researches on the test case generation techniques for BPMN models. Mutation Testing is a technique to evaluate the quality of test cases. However, the major disadvantage is expensive computational cost and time.

This research proposed an analysis of the mutation operator for BPMN that can apply for weak mutation testing technique and proposes a tool called WeMuBPMN for mutant generation based on weak mutation testing technique which can generate mutants, deploy mutants on BPMN Engine automatically. The tool can report results which are dead mutants, live mutants, execution time, mutation score, and test effectiveness. We test this tool with 8 BPMN Models. The tool can generate mutant by using 25 mutation operators, but only 13 mutation operators can test on BPMN Engine.

Field of Study: Software Engineering Student's Signature .....

Academic Year: 2019 Advisor's Signature .....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ได้สำเร็จลุล่วงด้วยความเมตตา และความช่วยเหลืออย่างสูงมากจากรองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์ อาจารย์ที่ปรึกษา ที่เสียสละเวลาช่วยให้คำปรึกษา ตรวจทาน และคำแนะนำที่มีประโยชน์ต่องานวิจัย เพื่อให้งานวิจัยฉบับนี้มีความสมบูรณ์ยิ่งขึ้น ตลอดจนความเอาใจใส่ และความเชื่อมั่นที่อาจารย์มีให้ผู้วิจัยมีกำลังใจในการทำงานต่อไป

ขอกราบขอบพระคุณรองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี ที่กรุณาเสียสละมาเป็นประธานกรรมการสอบวิทยานิพนธ์ ขอกราบขอบพระคุณรองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ ผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ทองทักษ์ และ ผู้ช่วยศาสตราจารย์ ดร.บัณฑิต ฐานะโสภณ ที่กรุณาเสียสละเวลามาเป็นกรรมการสอบวิทยานิพนธ์ ทั้งนี้ยังให้คำแนะนำที่ต้องแก้ไขให้วิทยานิพนธ์ฉบับนี้มีความสมบูรณ์มากยิ่งขึ้น

ขอขอบพระคุณคณาจารย์ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ที่ได้ให้ความรู้ทางด้านวิชาการ และด้านต่าง ๆ ที่เป็นประโยชน์ต่อวิทยานิพนธ์ฉบับนี้ อีกทั้งบุคลากรทุกท่านในภาควิชาฯ ที่ช่วยประสานงานให้ข้อมูล คำแนะนำ และความช่วยเหลือระหว่างที่ผู้วิจัยกำลังศึกษาตลอดจนสอบวิทยานิพนธ์ได้สำเร็จลุล่วง

ขอขอบคุณเพื่อน ๆ พี่ ๆ และน้อง ๆ ทุกคนของผู้วิจัยที่ให้คำแนะนำ และให้ความช่วยเหลือในทุก ๆ ด้านจนผู้วิจัยสามารถทำวิทยานิพนธ์ฉบับนี้จนสำเร็จลุล่วง

ขอขอบพระคุณสมาชิกในครอบครัวทุกคนที่ให้การสนับสนุน และให้กำลังใจแก่ผู้วิจัยเสมอมา และท้ายที่สุดนี้ ขอกราบขอบพระคุณมารดา ที่คอยเลี้ยงดู และสั่งสอนผู้วิจัยจนเติบใหญ่ เป็นแรงผลักดันให้ผู้วิจัยมีกำลังใจ จนทำให้วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้

ชาตรี งามเบญจวงศ์

สารบัญ

หน้า

สารบัญตาราง

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของงานวิจัย

ในปัจจุบันองค์กรหลายแห่งได้เล็งเห็นถึงความสำคัญของการพัฒนาโครงสร้างกระบวนการเชิงธุรกิจ โดยแบบจำลองและสัญลักษณ์ของกระบวนการธุรกิจ หรือ แบบจำลองบีพีเอ็มเอ็น (Business Process Model and Notation หรือ BPMN) [1] เป็นหนึ่งในแบบจำลองที่ถูกเลือกใช้เพื่อให้ผู้ใช้ในองค์กรได้เห็นภาพรวมของกระบวนการทางธุรกิจ และเป็นสื่อกลางสำหรับสื่อสารร่วมกัน ต่อมาในปีพุทธศักราช 2554 แบบจำลองบีพีเอ็มเอ็นได้ถูกปรับปรุงข้อกำหนดให้สอดคล้องกับการใช้งานจริงมากขึ้น โดยในเวอร์ชัน 2.0 หนึ่งในการปรับปรุงที่สำคัญ คือ การพัฒนาข้อกำหนดให้แผนภาพบีพีเอ็มเอ็นสามารถดำเนินการได้อย่างอัตโนมัติผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น (BPMN Engine) ได้ และในปีพุทธศักราช 2556 แบบจำลองบีพีเอ็มได้ถูกบรรจุในมาตรฐานไอเอสโอ ISO/IEC 19510:2013 [2] จากจุดเด่นที่ได้กล่าวมาในข้างต้น ทำให้นักพัฒนากระบวนการและนักทดสอบระบบต้องทำงานร่วมกัน เพื่อให้การสร้างแบบจำลองบีพีเอ็มเอ็นสามารถดำเนินการผ่านเครื่องมือประมวลผลได้ มีการทำงานที่ถูกต้อง ครบถ้วน และมีความน่าเชื่อถือต่อผู้ใช้งาน

กระบวนการทดสอบซอฟต์แวร์เป็นกระบวนการที่สำคัญในวงจรชีวิตการพัฒนาซอฟต์แวร์ (Software Development Life Cycle) เพื่อตรวจสอบความถูกต้องของซอฟต์แวร์ให้ตรงความคาดหวัง หรือความต้องการของผู้ใช้งาน ค้นหาข้อบกพร่อง(Defect) และมีความน่าเชื่อถือ ก่อนส่งมอบให้กับผู้ใช้งาน ซึ่งการทดสอบแบบมิวเทชัน (Mutation Testing) หรือสตรองมิวเทชัน (Strong Mutation) เป็นวิธีการหนึ่งที่ใช้สำหรับใช้ทดสอบคุณภาพของชุดทดสอบ และความน่าเชื่อถือของโปรแกรม โดยการทดสอบแบบมิวเทชันเป็นการใส่ข้อผิดพลาดลงไปในโปรแกรม เพื่อให้เกิดข้อผิดพลาดเพียงหนึ่งที่ต่อหนึ่งโปรแกรม และโปรแกรมที่ถูกใส่ข้อผิดพลาดลงไปนั้นถูกเรียกว่ามิวแตนท์ (Mutant) ซึ่งการสร้างมิวแตนท์นั้นมีแนวทางที่ถูกกำหนดไว้จากตัวดำเนินการมิวเทชัน ว่าจะใส่ข้อผิดพลาดรูปแบบใดลงในโปรแกรมได้บ้าง เช่น การใส่ข้อผิดพลาดประเภทตัวดำเนินการทางคณิตศาสตร์ โดยการเปลี่ยนตัวดำเนินการทางคณิตศาสตร์ +, -, *, / หรือ มอดุลาร์ (%) ในนิพจน์ที่ต้องการปรับเปลี่ยน

จากงานวิจัย [3] พบว่าแนวโน้มการศึกษาวิธีการทดสอบนี้ยังเป็นที่นิยม แต่ทิ้งท้ายถึงปัญหาที่สำคัญของการทดสอบที่ยังต้องใช้แรงงาน และทรัพยากรที่ค่อนข้างสูง จึงได้มีงานวิจัย [4] เสนอวิธีการวีคมิวเทชัน (Weak Mutation Testing) ซึ่งเป็นการใส่ข้อผิดพลาดลงไปซอฟต์แวร์ ในระดับหน่วยย่อย หรือ คอมโพเนนท์ (Component)

จากงานวิจัย “Mutation Operators in BPMN Model” [5] ได้เสนอตัวดำเนินการสำหรับแบบจำลองบีพีเอ็มเอ็น โดยแบ่งประเภทของตัวดำเนินการทั้งหมด 4 รูปแบบ ได้แก่ ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร (Identifier Mutation Operator) ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข (Expression Mutation Operator) ตัวดำเนินการมิวเทชันเชิงกิจกรรม (Activity Mutation Operator) และ ตัวดำเนินการมิวเทชันเกี่ยวกับข้อยกเว้นและเหตุการณ์ (Exception and Event Mutation Operators) และได้ใช้การทดสอบแบบสตรองมิวเทชัน เพื่อวัดตัวดำเนินการมิวเทชันแต่ละตัวสามารถสร้างมิวแตนท์ได้เท่าไหร่จากแบบจำลองที่จัดเตรียมไว้ แต่การทดสอบสตรองมิวเทชันมีข้อเสียในด้านความสิ้นเปลืองเกี่ยวกับการกระทำการ (Computational Cost) จึงเป็นข้อจำกัดของเครื่องมือในงานวิจัยนี้ที่สามารถสร้างมิวแตนท์ที่ได้ไม่เกิน 5 ตัว ในกลุ่ม EAR, ERR และ ELR รวมถึงขาดการประเมินประสิทธิผลของตัวดำเนินการมิวเทชันที่ได้นิยามไว้

งานวิจัยนี้ได้วิเคราะห์ตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็น [5] และศึกษากระบวนการในการทดสอบมิวเทชันจากงานวิจัยที่ผ่านมา [6-20] โดยมีเป้าหมายสร้างเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยการทดสอบแบบวีคมิวเทชัน เนื่องจากเป็นวิธีการหนึ่งที่ช่วยลดความสิ้นเปลืองเกี่ยวกับการกระทำการ และเริ่มต้นจากการศึกษาตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็น จากนั้นพิจารณานำตัวดำเนินการมิวเทชันมาใช้กับการทดสอบวีคมิวเทชัน รวมถึงการนำแบบจำลองบีพีเอ็มเอ็น และชุดทดสอบไปใช้งานกับเครื่องประมวลผลแบบจำลองที่เป็นโอเพนซอร์ส (Open-Source) จากนั้นพัฒนาเครื่องมือที่มีการแก้ไขข้อจำกัดของการวิจัยก่อนหน้า [5] โดยการสร้างมิวแตนท์ให้ครอบคลุม โดยใช้การทดสอบแบบวีคมิวเทชันและเชื่อมต่อกับเครื่องประมวลผลแบบจำลองได้อย่างอัตโนมัติ เพื่อลดแรงงานของนักทดสอบแบบจำลอง โดยทดสอบมิวแตนท์ที่สร้างขึ้นมาด้วยกรณีทดสอบที่จัดเตรียมไว้ และรายงานผลผ่านเครื่องมือ เช่น จำนวนของมิวแตนท์ที่ถูกฆ่า จำนวนมิวแตนท์ที่มีชีวิต เวลาที่ใช้ทดสอบ เป็นต้น

1.2 วัตถุประสงค์ของงานวิจัย

1) เพื่อนำตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นมาประยุกต์ใช้กับการทดสอบแบบวีคมิวเทชัน
2) เพื่อสร้างเครื่องมือสร้างมิวแตนท์สำหรับการทดสอบแบบวีคมิวเทชันของแบบจำลองบีพีเอ็มเอ็น

1.3 ขอบเขตของงานวิจัย

1) แบบจำลองบีพีเอ็มเอ็นที่ใช้ในการทดสอบต้องเป็นไปตามมาตรฐานของ Object Management Group เวอร์ชัน 2.0.2 [1]
2) แบบจำลองบีพีเอ็มเอ็นที่นำมาทดสอบมีกรณีทดสอบเดิมอยู่แล้ว หรือต้องสร้างกรณีทดสอบใหม่เพิ่มเติมจากผู้ใช้งาน
3) เครื่องมือทดสอบสามารถตรวจสอบแบบจำลองบีพีเอ็มเอ็นได้ ว่าเป็นแบบจำลองบีพีเอ็มเอ็นหรือไม่
4) เครื่องมือสามารถสร้างมิวแตนท์ได้อย่างอัตโนมัติ ตามตัวดำเนินการมิวเทชันของงานวิจัย [5]
5) เครื่องมือทดสอบสามารถทดสอบวีคมิวเทชันในระดับประโยค ST-WEAK/1 (Statement-WEAK/1) และระดับกลุ่มประโยค BB-WEAK/1(Basic-Block-WEAK/1) และ BB-WEAK/N (Basic-Block-WEAK/N) ตามตัวดำเนินการมิวเทชันที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น Camunda BPMN รองรับ
6) เครื่องมือไม่รองรับการระบุมิวแตนท์สมมูลได้
7) ทดสอบเครื่องมือโดยจะทดสอบกับแบบจำลองบีพีเอ็มเอ็น 3 แบบจำลอง
8) เครื่องมือสามารถรายงานผลลัพธ์ของการทดสอบได้ โดยแสดงจำนวนมิวแตนท์ทั้งหมดที่เกิดขึ้น จำนวนมิวแตนท์ที่ถูกฆ่า จำนวนมิวแตนท์ที่มีชีวิต เวลาที่ใช้ในการทดสอบ คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

1.4 ขั้นตอนและวิธีการดำเนินงานวิจัย

1) ศึกษางานวิจัย ทฤษฎี และเครื่องมือทดสอบแบบมิวเทชันที่เกี่ยวข้อง
2) ศึกษาแบบจำลองบีพีเอ็มเอ็น
3) ศึกษาวิธีการทดสอบแบบมิวเทชัน และการทดสอบแบบวีคมิวเทชัน
4) ศึกษาตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็น โดยวิเคราะห์ตัวดำเนินการมิวเทชันในงานวิจัย [5] ที่ได้กำหนดขึ้น และประยุกต์ตัวดำเนินการมิวเทชันกับการทดสอบแบบวีคมิวเทชัน
5) ศึกษา และเลือกเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นที่เป็นโอเพนซอร์ส ที่มีการพัฒนาได้ใกล้เคียงตามมาตรฐานที่กำหนดไว้ใน [1]
6) ค้นหา หรือสร้างแบบจำลองตัวอย่าง และสร้างกรณีทดสอบสำหรับโปรแกรมตัวอย่าง
7) ออกแบบเครื่องมือ
8) พัฒนาเครื่องมือ ทดสอบ และปรับปรุงเครื่องมือ
9) บันทึก สรุปผลการวิจัย และข้อเสนอแนะ

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) นำวิธีการทดสอบวีคมิวเทชันกับแบบจำลองบีพีเอ็มเอ็น และการสร้างกรณีทดสอบไปประยุกต์ใช้กับภาษาอื่นๆ เช่น แบบจำลองซีเอ็มเอ็มเอ็น [21] เป็นต้น
2) เครื่องมือที่พัฒนาช่วยลดภาระให้นักทดสอบในการสร้างมิวแตนท์ด้วยวิธีการวีคมิวเทชันกับแบบจำลองบีพีเอ็มเอ็นได้

1.6 บทความวิชาการที่ได้รับการตีพิมพ์

งานวิจัยนี้ได้รับคัดเลือก และตีพิมพ์เป็นบทความวิชาการเรื่อง "A Weak Mutation Testing Framework for BPMN" โดย ชาตรี งามเบญจวงศ์ และ ธาราทิพย์ สุวรรณศาสตร์ ในการประชุมวิชาการ "The 27th International MultiConference of Engineers and Computer Scientists (IMECS 2019)" ระหว่างวันที่ 13-15 มีนาคม 2562 ณ โรงแรมรอยัลการ์เด้น เขตบริหารพิเศษฮ่องกง สาธารณรัฐประชาชนจีน

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ภาษาเอกซ์เอ็มแอล (Extensible Markup Language หรือ XML)

ภาษาเอกซ์เอ็มแอล (Extensible Markup Language หรือ XML) [22] เป็นภาษากลุ่มมาร์ก-อัพที่มีจุดประสงค์เพื่อให้ความชัดเจนในการให้รายละเอียดเกี่ยวกับข้อมูล หรือพฤติกรรม และถูกใช้เป็นสื่อกลางสำหรับการแลกเปลี่ยนข้อมูลผ่านระบบเครือข่ายคอมพิวเตอร์ อาทิ เช่น การแลกเปลี่ยนข้อมูลระหว่างคอมพิวเตอร์ที่ใช้ระบบปฏิบัติการ หรือโปรแกรมประยุกต์ที่แตกต่างๆ กัน เป็นต้น ซึ่งภาษาเอกซ์เอ็มแอลถูกพัฒนาโดยเวิล์ดไวด์เว็บคอนซอร์เทียม (World Wild Web Consortium หรือ W3C) [23] จากการนำภาษาเอสจีเอ็มแอล (Standard Generalized Markup Language หรือ SGML) [24] โดยลดความซับซ้อนของโครงสร้าง และปรับปรุงให้เหมาะสมกับการแลกเปลี่ยนข้อมูลผ่านระบบเครือข่ายคอมพิวเตอร์

ภาษาเอกซ์เอ็มแอลเป็นภาษาที่มีความยืดหยุ่น สามารถกำหนดรูปแบบ และโครงสร้างของเอกสารได้ จากการกำหนดชื่อแท็ก (Tag) และแอตทริบิวต์ (Attribute) โดยเอกสารถูกสร้างขึ้นมานั้นต้องอยู่ในรูปแบบที่ถูกต้อง (Well-formed) ซึ่งมีลักษณะ ดังนี้

1) เอกสารเอกซ์เอ็มแอลต้องมีการประกาศเวอร์ชัน หรือการเข้ารหัสอักขระ(Encoding) ภายใต้แท็กที่มีโครงสร้าง <? xml ?> ดังภาพที่ 2-1

```
<?xml version="1.0" encoding="UTF-8"?>
```

ภาพที่ 2-1 ตัวอย่างการประกาศเอกสารเอกซ์เอ็มแอล

2) เอกสารเอกซ์เอ็มแอลต้องมีแท็กที่เป็นรากได้ เพียง 1 แท็กต่อ 1 เอกสาร ดังภาพที่ 2-2

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
   <book id="bk101">
      <author>Bruce Silver</author>
      <title>Bpmn Method and Style, 2nd Edition</title>
      <genre>Computer</genre>
      <price>39.95</price>
      <publish_date>2011-10-17</publish_date>
   </book>
</catalog>
```

ภาพที่ 2-2 ตัวอย่างการประกาศแท็กราก

จากภาพที่ 2-2 แท็ก <catalog> ทำหน้าที่เป็นแท็กรากที่ประกอบไปด้วยแท็ก <book> จำนวน 1 ชุด

3) เอกสารเอกซ์เอ็มแอลในแต่ละอิลิเมนต์ เมื่อมีแท็กเปิด และต้องมีแท็กปิดคู่กันเสมอ โดยแท็กปิด คือ แท็กที่มีชื่อเหมือนแท็กเปิด แต่มีการเพิ่มเครื่องหมาย “/” ก่อนหน้าชื่อแท็กตามโครงสร้าง แท็กเปิด ข้อมูล และแท็กปิด ดังภาพที่ 2-3

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
   <book id="bk101">
      <author>Bruce Silver</author>
      <title>Bpmn Method and Style, 2nd Edition</title>
      <genre>Computer</genre>
      <price>39.95</price>
      <publish_date>2011-10-17</publish_date>
   </book>
</catalog>
```

ภาพที่ 2-3 ตัวอย่างแท็กเปิด และแท็กปิด

จากภาพที่ 2-3 เมื่อมีการประกาศแท็ก <author> แล้ว ต้องมี </author> คู่กันเสมอ หรือ บางอิลิเมนต์ เช่น แท็ก<price> ไม่จำเป็นต้องมีข้อมูล

4) เอกสารเอกซ์เอ็มแอลต้องไม่มีการประกาศแท็กเปิด และแท็กปิดคร่อมกัน ดังภาพที่ 2-4

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
   <book id="bk103">
      <author>Norman Fenton, James Bieman
         <title>Software Metrics: A Rigorous and Practical
Approach, Third Edition
          </author>
       </title>
      <genre>Computer</genre>
      <price>52.99</price>
      <publish_date>2014-10-01</publish_date>
   </book>
</catalog>
```

ภาพที่ 2-4 ตัวอย่างการประกาศแท็กเปิด และแท็กปิดคร่อมกัน

จากภาพที่ 2-4 แท็ก <title> ที่อยู่ภายใต้แท็ก <author> ไม่สามารถประกาศคร่อมกันได้

5) การประกาศแอตทริบิวต์ ต้องประกาศภายในแท็กเปิดเท่านั้น ดังภาพที่ 2-5

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
   <book id="bk104">
      <author>Frederick P. Brooks Jr</author>
      <title>The Mythical Man-Month: Essays on Software
Engineering, Anniversary Edition (2nd Edition)</title>
      <genre>Computer</genre>
      <price>29.88r</price>
      <publish_date>1995-08-12</publish_date>
   </book>
</catalog>
```

ภาพที่ 2-5 ตัวอย่างการประกาศแอตทริบิวต์

จากภาพที่ 2-5 แอตทริบิวต์ id ถูกประกาศ และมีการกำหนดค่าภายใต้แท็ก <book> ซึ่งเป็นแท็กเปิด โดยมีการกำหนดค่าเท่ากับ bk104

2.1.2 แบบจำลองบีพีเอ็มเอ็น (BPMN Model)

บีพีเอ็มเอ็น [1] เป็นสัญลักษณ์ที่ใช้สำหรับอธิบายกระบวนการทางธุรกิจที่มีจุดประสงค์เพื่อผู้ใช้ได้เห็นภาพรวมของกระบวนการทางธุรกิจ และเป็นสื่อกลางสำหรับสื่อสารร่วมกันระหว่างการพัฒนากระบวนการทางธุรกิจ และการพัฒนาซอฟต์แวร์ ซึ่งแบบจำลองบีพีเอ็มเอ็นถูกพัฒนาโดย Business Process Management Initiative (BPMI) ในปีพุทธศักราช 2547 จากนั้นในปีพุทธศักราช 2551 แบบจำลองบีพีเอ็มเอ็นถูกโอนให้ขึ้นกับ Object Management Group (OMG) ซึ่งเป็นผู้พัฒนา Unified Modeling Language (UML) และในปีพุทธศักราช 2554 แบบจำลองบีพีเอ็มเอ็นได้ถูกปรับปรุงข้อกำหนดให้สอดคล้องกับการใช้งานจริงมากขึ้น เป็นเวอร์ชัน 2.0 โดยมีรายละเอียดการปรับปรุงที่สำคัญ [25] ดังนี้

- กำหนดมาตรฐานการแลกเปลี่ยนข้อมูลของแบบจำลองบีพีเอ็มเอ็นสำหรับเครื่องมือออกแบบ และเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นระหว่างผู้พัฒนาเครื่องมือภายใต้การกำกับของหน่วยงาน BPMN Model Interchange Working Group (BPMN MIWG) ซึ่งเป็นหน่วยงานหนึ่งภายใน OMG
- กำหนดสัญลักษณ์เพิ่มเติม เพื่ออธิบายกระบวนการขององค์กรให้สอดคล้องกับการทำงานในปัจจุบัน
- เพิ่มตัวอย่าง และแสดงความสัมพันธ์ระหว่างแบบจำลองบีพีเอ็มเอ็น และ แบบจำลองดับเบิลยูเอส-บีเพลส

สำหรับการนำแบบจำลองบีพีเอ็มเอ็นมาใช้งานในองค์กรนั้น [26] ได้กำหนดไว้ 2 กลุ่ม และมี 3 ระดับ ได้แก่

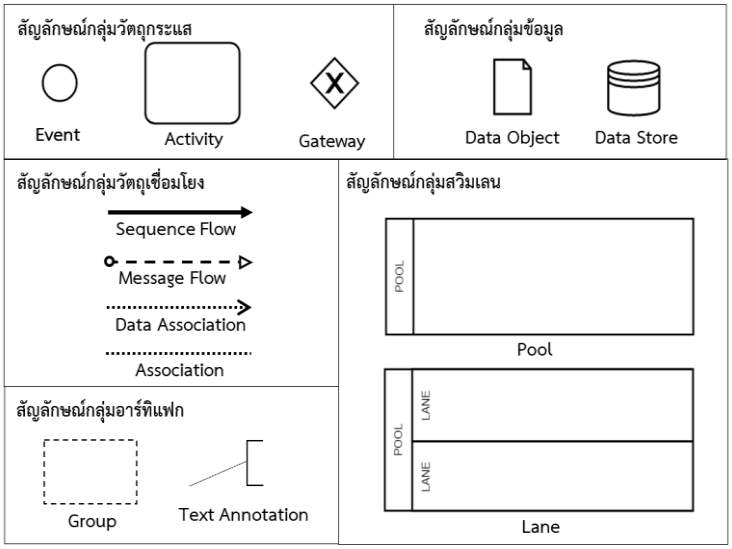
- กลุ่มที่ 1 : กระบวนการที่ไม่สามารถกระทำการ (Non-executable process)
  - ระดับที่ 1 : แบบจำลองเชิงบรรยาย (Descriptive process model) เป็นแบบจำลองที่ใช้สำหรับอธิบายกระบวนการทำงานขององค์กร เพื่อให้เห็นภาพรวม และใช้สำหรับการสื่อสาร ในมุมมองการเปรียบกระบวนทำงานที่มีอยู่ในปัจจุบัน (As-Is) และกระบวนการใหม่ (To-be) โดยองค์กรส่วนใหญ่มีการใช้งานแบบจำลองบีพีเอ็นเอ็นในระดับนี้
  - ระดับที่ 2 : แบบจำลองเชิงวิเคราะห์ (Analytic process modeling) เป็นแบบจำลองที่ขยายความสามารถของแบบจำลองเชิงบรรยาย โดยปรับให้รองรับกรณีที่เกิดข้อยกเว้น (Exceptions) หรือเหตุการณ์ (Events) และลงรายละเอียดของกระบวนการลึกขึ้น เพื่อสร้างการจำลองด้วยระบบคอมพิวเตอร์สำหรับการวิเคราะห์กระบวนการในเชิงปริมาณ และคุณภาพ

- กลุ่มที่ 2 : กระบวนการที่สามารถกระทำการ (Executable process)
  - ระดับที่ 3 : แบบจำลองดำเนินการ(Executable BPMN) เป็นแบบจำลองบีพีเอ็มเอ็นในระดับที่ 1 และ 2 ที่สามารถถูกประมวลผลผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น

ในส่วนของสัญลักษณ์ที่ใช้ในแบบจำลองบีพีเอ็มเอ็นประกอบไปด้วย 5 หมวดหมู่ ดังนี้

1) วัตถุกระแส (Flow Object) ใช้อธิบายพฤติกรรมของกระบวนการ โดยมีสัญลักษณ์ที่ใช้ได้แก่ เหตุการณ์ (Event) คือ สิ่งที่เกิดขึ้นได้ในตอนเริ่มต้น ระหว่างกลาง และสิ้นสุดกระบวนการ, กิจกรรม (Activity) คือ งานหรือสิ่งที่เกิดขึ้นในกระบวนการ และเกตเวย์ (Gateway) คือ สิ่งที่ใช้ควบคุมการไหลภายในกระบวนการ
2) ข้อมูล (Data) ใช้อธิบายข้อมูลที่จำเป็นสำหรับกิจกรรม หรือข้อมูลที่เกิดขึ้นจากกิจกรรม โดยมีสัญลักษณ์ที่ใช้ได้แก่ วัตถุข้อมูล (Data Object) และ ที่จัดเก็บข้อมูล (Data Store)
3) วัตถุเชื่อมโยง (Connecting Object) ใช้เชื่อมวัตถุกระแสเข้าด้วยกัน หรือระหว่างวัตถุกระแสกับข้อมูล โดยมีสัญลักษณ์ที่ใช้ได้แก่ กระแสลำดับ (Sequence Flow) ใช้เชื่อมโยงระหว่างวัตถุกระแส เพื่อแสดงให้เห็นขั้นตอนการทำงาน, กระแสสาร (Message Flow) ใช้เชื่อมโยงสารระหว่างผู้ส่งกับผู้รับ, การเชื่อมโยงข้อมูล (Data Association) ใช้เชื่อมโยงระหว่างวัตถุกระแสกับข้อมูล เพื่อแสดงให้เห็นถึงความสัมพันธ์ของข้อมูลกับวัตถุกระแสที่สนใจ และ การเชื่อมโยง (Association) ใช้สำหรับเชื่อมโยงระหว่างวัตถุกระแสกับอาร์ทิแฟก
4) สวิมเลน (Swimlane) ใช้สำหรับจัดกลุ่มวัตถุของแบบจำลอง โดยมีสัญลักษณ์ที่ใช้ ได้แก่ พูล (Pool) และเลน (Lane)
5) อาร์ทิแฟก (Artifacts) ใช้สำหรับอธิบายข้อมูลของกระบวนการเพิ่มเติมโดยมีสัญลักษณ์ที่ใช้ ได้แก่ การจัดกลุ่ม (Group) และข้อความหมายเหตุ (Text Annotation)

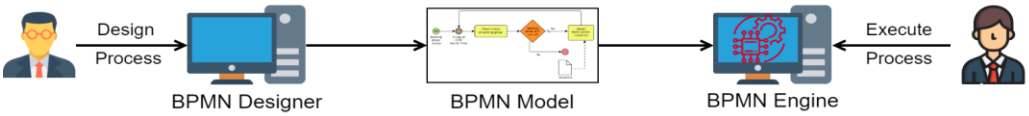
สำหรับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็นได้ถูกแสดงไว้ดังภาพที่ 2-6



ภาพที่ 2-6 สัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น

2.1.3 เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น (BPMN Engine)

เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น [27] เป็นเครื่องมือที่ช่วยลดภาระของนักพัฒนาซอฟต์แวร์ โดยนำแบบจำลองบีพีเอ็มเอ็นที่ถูกสร้างจากกระบวนการทางธุรกิจที่สนใจโดยไม่ต้องแปลงจากแบบจำลองไปเป็นซอร์สโค้ด ซึ่งองค์ประกอบของเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นมี 3 ส่วน ดังภาพที่ 2-7 ได้แก่



ภาพที่ 2-7 องค์ประกอบของเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น [27]

1) เครื่องมือสร้างแบบจำลองบีพีเอ็มเอ็น (BPMN Designer) เป็นเครื่องมือที่ช่วยให้นักวิเคราะห์เชิงธุรกิจสามารถสร้างแบบจำลอง เพื่ออธิบายกระบวนการทางธุรกิจ ได้สะดวกในรูปแบบกราฟิกไม่จำเป็นต้องเขียนคำสั่ง
2) แบบจำลองบีพีเอ็มเอ็น (BPMN Model) เป็นผลผลิตที่เกิดจากเครื่องมือสร้าแบบจำลองบีพีเอ็มเอ็น ซึ่งถูกจัดเก็บรูปแบบเอกซ์เอ็มแอลภายใต้แท็ก <bpmn:definitions> ซึ่งสามารถแยกออกเป็น 2 ส่วนย่อย ได้แก่
    2.1) ส่วนจัดเก็บรายละเอียดของกระบวนการ – เพื่อให้เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นสามารถตีความ และดำเนินการตามกระบวนการที่ได้กำหนดไว้ ซึ่งถูกจัดเก็บภายใต้แท็ก <bpmn:process>

2.2) ส่วนจัดเก็บรายละเอียดการแสดงผล – เพื่อให้เครื่องมือสร้างแบบจำลองบีพีเอ็มเอ็นสามารถสร้างแบบจำลองในรูปแบบกราฟิกได้ ซึ่งถูกจัดเก็บภายใต้แท็ก <bpmndi:BPMNDiagram> ตามข้อตกลงที่กำหนดโดย BPMN Model Interchange Working Group หรือ BPMN MIWG

จากแบบจำลองตัวอย่าง ซึ่งเป็นการกระบวนการอย่างง่าย ดังภาพที่ 2-8 ซึ่งกระบวนการประกอบไปด้วย Start Event, User Task และ End Event

ภาพที่ 2-8 ตัวอย่างกระบวนการอย่างง่าย

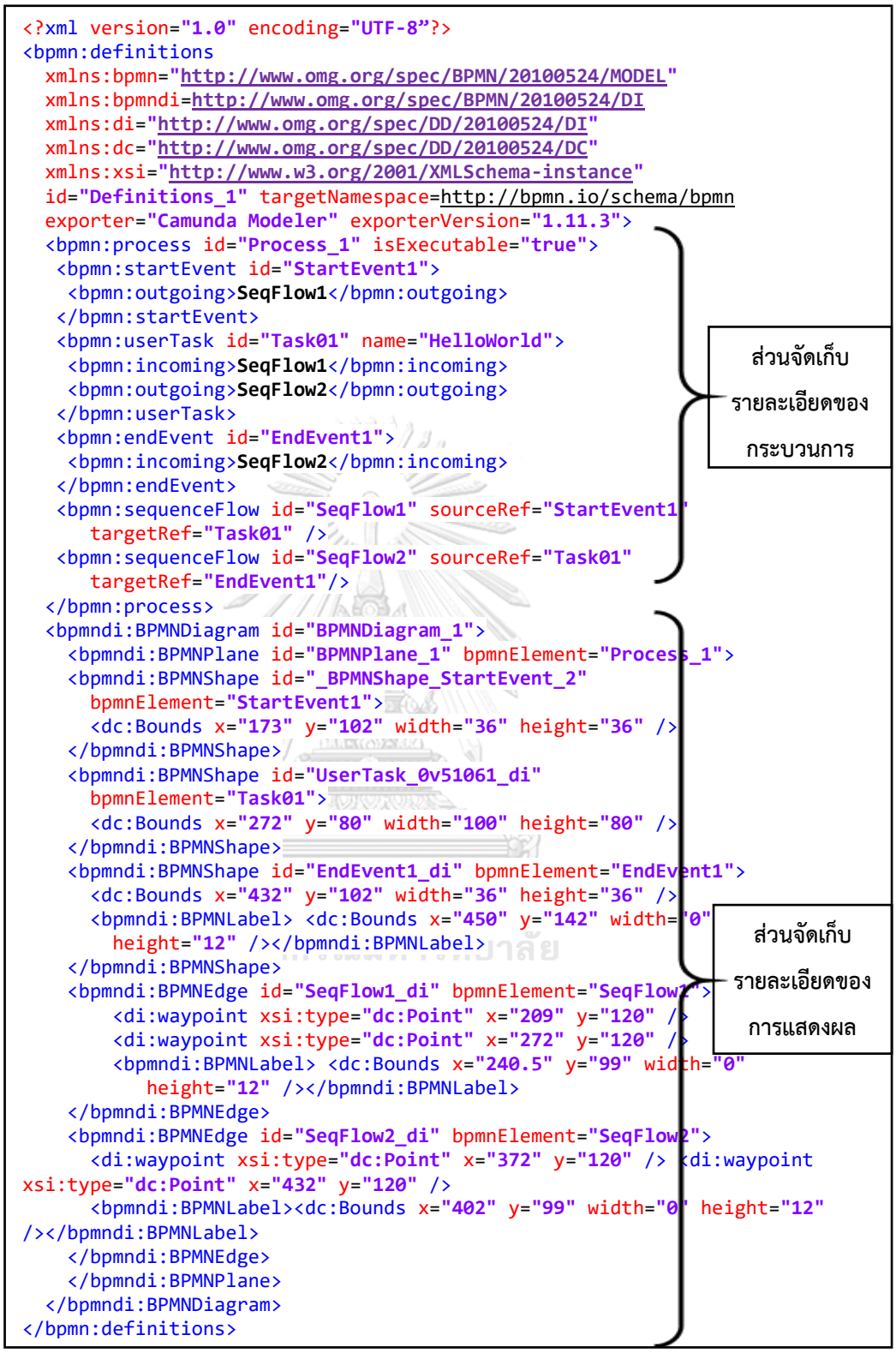
สำหรับในส่วนของแบบจำลองถูกจัดเก็บภายใต้แท็ก <bpmn2:definitions> โดยส่วนของกระบวนการถูกจัดเก็บภายใต้แท็ก <bpmn2:process> และส่วนการแสดงผลถูกจัดเก็บภายใต้แท็ก <bpmndi:BPMNDiagram> ดังภาพที่ 2-9

3) เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น (BPMN Engine) เป็นเครื่องมือที่นำแบบจำลอง บีพีเอ็มเอ็นมาประมวลผล และดำเนินงานตามกระบวนการที่ถูกนิยามไว้

2.1.4 การทดสอบมิวเทชัน (Mutation Testing)

การทดสอบมิวเทชันเป็นเทคนิคหนึ่งของการทดสอบเชิงจับผิด (fault-based testing) เพื่อทดสอบความน่าเชื่อถือของซอฟต์แวร์ และวัดประสิทธิภาพของชุดทดสอบ ซึ่งการทดสอบมิวเทชันตั้งอยู่บนสมมติฐาน 2 ข้อ จากงานวิจัย [28] ได้แก่

1) Competent Programmer Hypothesis (CPH) –ซึ่งมีรายละเอียดว่า “นักพัฒนาซอฟต์แวร์ที่มีความสามารถ พวกเขามีแนวโน้มที่จะพัฒนาซอฟต์แวร์ได้ใกล้เคียงกับเวอร์ชันที่ถูกต้อง มีข้อผิดพลาดที่ไม่ร้ายแรง สามารถแก้ไขซอฟต์แวร์เพียงเล็กน้อยเท่านั้น”
2) Coupling Effect - ซึ่งมีรายละเอียดว่า “ถ้าข้อมูลสำหรับการทดสอบสามารถเจอข้อผิดพลาดเพียงเล็กน้อยของซอฟต์แวร์ได้ ข้อมูลทดสอบชุดดังกล่าวสามารถใช้สำหรับการค้นหาข้อผิดพลาดที่ซับซ้อนได้เช่นกัน”

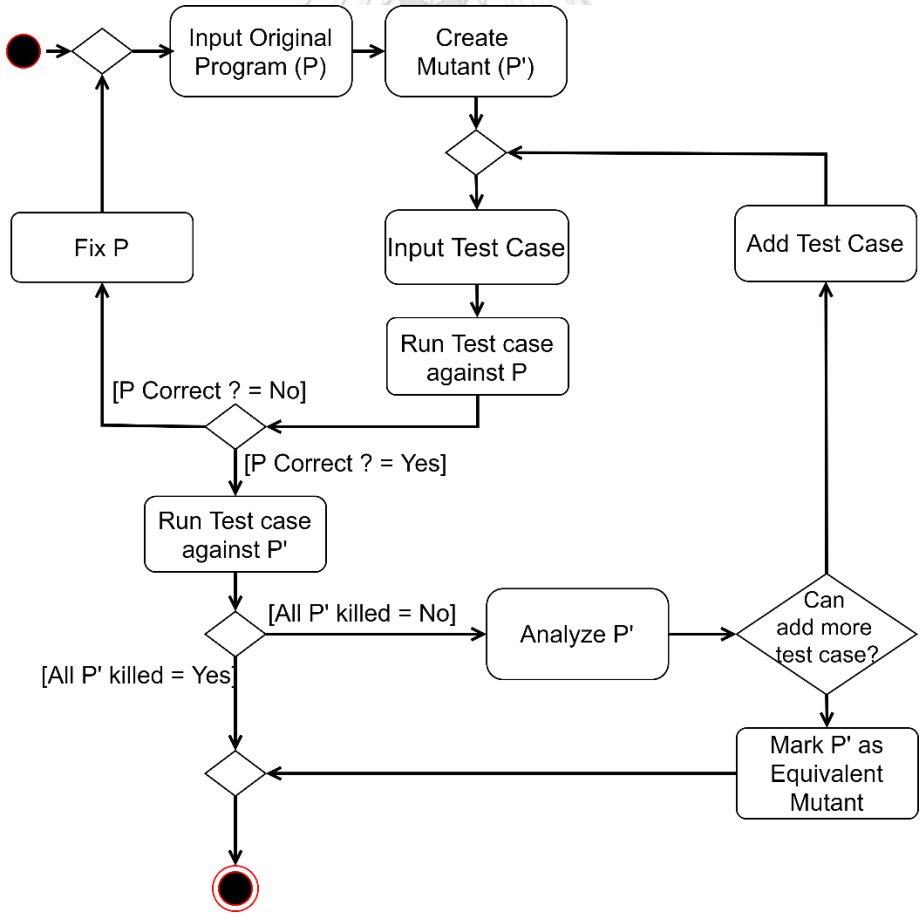



```
<?xml version="1.0" encoding="UTF-8”?>
<bpmn:definitions
  xmlns:bpmn="http://www.omg.org/spec/BPMN/20100524/MODEL"
  xmlns:bpmndi=http://www.omg.org/spec/BPMN/20100524/DI
  xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
  xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  id="Definitions_1" targetNamespace=http://bpmn.io/schema/bpmn
  exporter="Camunda Modeler" exporterVersion="1.11.3">
  <bpmn:process id="Process_1" isExecutable="true">
   <bpmn:startEvent id="StartEvent1">
    <bpmn:outgoing>SeqFlow1</bpmn:outgoing>
   </bpmn:startEvent>
   <bpmn:userTask id="Task01" name="HelloWorld">
    <bpmn:incoming>SeqFlow1</bpmn:incoming>
    <bpmn:outgoing>SeqFlow2</bpmn:outgoing>
   </bpmn:userTask>
   <bpmn:endEvent id="EndEvent1">
    <bpmn:incoming>SeqFlow2</bpmn:incoming>
   </bpmn:endEvent>
   <bpmn:sequenceFlow id="SeqFlow1" sourceRef="StartEvent1"
      targetRef="Task01" />
   <bpmn:sequenceFlow id="SeqFlow2" sourceRef="Task01"
      targetRef="EndEvent1"/>
  </bpmn:process>
  <bpmndi:BPMNDiagram id="BPMNDiagram_1">
    <bpmndi:BPMNPlane id="BPMNPlane_1" bpmnElement="Process_1">
    <bpmndi:BPMNShape id="_BPMNShape_StartEvent_2"
      bpmnElement="StartEvent1">
      <dc:Bounds x="173" y="102" width="36" height="36" />
    </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="UserTask_0v51061_di"
      bpmnElement="Task01">
      <dc:Bounds x="272" y="80" width="100" height="80" />
    </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="EndEvent1_di" bpmnElement="EndEvent1">
      <dc:Bounds x="432" y="102" width="36" height="36" />
      <bpmndi:BPMNLabel> <dc:Bounds x="450" y="142" width="0"
        height="12" /></bpmndi:BPMNLabel>
    </bpmndi:BPMNShape>
    <bpmndi:BPMNEdge id="SeqFlow1_di" bpmnElement="SeqFlow1">
        <di:waypoint xsi:type="dc:Point" x="209" y="120" />
        <di:waypoint xsi:type="dc:Point" x="272" y="120" />
        <bpmndi:BPMNLabel> <dc:Bounds x="240.5" y="99" width="0"
           height="12" /></bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="SeqFlow2_di" bpmnElement="SeqFlow2">
      <di:waypoint xsi:type="dc:Point" x="372" y="120" /> <di:waypoint
xsi:type="dc:Point" x="432" y="120" />
      <bpmndi:BPMNLabel><dc:Bounds x="402" y="99" width="0" height="12"
/></bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    </bpmndi:BPMNPlane>
  </bpmndi:BPMNDiagram>
</bpmn:definitions>
```

ภาพที่ 2-9 โครงสร้างของแบบจำลองบีพีเอ็มเอ็น

สำหรับกระบวนการของการทดสอบมิวเทชันได้แสดงภาพรวมของกระบวนการดังภาพที่ 2-10 โดยเริ่มต้นจากโปรแกรมต้นฉบับ และกรณีทดสอบของโปรแกรมต้นฉบับ จากนั้นเปลี่ยนแปลงบางตำแหน่งของโปรแกรมต้นฉบับด้วยการสร้างข้อผิดพลาดลงในโปรแกรม โดยเรียกโปรแกรมที่ถูกใส่ข้อผิดพลาดว่า “มิวแตนท์” แล้วจึงนำกรณีทดสอบเดิมของโปรแกรมต้นฉบับทดสอบกับมิวแตนท์ โดยผลลัพธ์ของการทดสอบมี 2 แบบ ได้แก่

1) มิวแตนท์ถูกฆ่า (Killed Mutant) หรือ ตาย (Dead Mutant) หมายถึง กรณีทดสอบที่มีอยู่สามารถหาข้อผิดพลาดที่ใส่ลงไปในมิวแตนท์ได้
2) มิวแตนท์ที่มีชีวิต (Live Mutant) หมายถึง กรณีทดสอบที่มีอยู่ไม่สามารถดักจับข้อผิดพลาดที่ใส่ลงไปในมิวแตนท์ได้ เพราะมิวแตนท์แสดงผลลัพธ์เช่นเดียวกับโปรแกรมต้นฉบับ สำหรับรูปแบบนี้สามารถตีความถึงสาเหตุได้ 2 สาเหตุ ได้แก่ (1) กรณีทดสอบเดิมที่มีอยู่ไม่เพียงพอ ซึ่งสามารถแก้ไขได้โดยการเพิ่มกรณีทดสอบโดยอิงจากมิวแตนท์ที่สร้างขึ้น หรือ (2) มิวแตนท์ที่ถูกสร้างขึ้นเป็นมิวแตนท์สมมูล (Equivalent Mutant) แม้ว่าเพิ่มกรณีทดสอบแล้ว แต่ยังไม่สามารถค้นพบข้อผิดพลาดในมิวแตนท์ได้



ภาพที่ 2-10 กระบวนการทดสอบมิวเทชัน [28]

สำหรับการทดสอบมิวเทชันมีมาตรวัดที่เกี่ยวข้องดังนี้

1) คะแนนมิวเทชัน (Mutation Score หรือ MS) ซึ่งเป็นมาตรวัดสำหรับตรวจสอบว่าชุดทดสอบนั้นเพียงพอสำหรับทดสอบโปรแกรมที่สนใจหรือไม่ โดยสามารถคำนวณคะแนนมิวเทชันได้จากสมการที่ 1

$$MS(P,T) = \frac{M_k}{M_t - M_q} \qquad \textit{สมการที่ 1}$$

โดยที่ P คือ โปรแกรมที่ถูกทดสอบ
T คือ ชุดทดสอบ
M_k คือ จำนวนของมิวแตนท์ที่ถูกฆ่า
M_t คือ จำนวนของมิวแตนท์ทั้งหมด
M_q คือ จำนวนของมิวแตนท์สมมูล

โดยเป้าหมายของนักทดสอบมีจุดมุ่งหมายที่ได้คะแนนมิวเทชันเท่ากับ 1 ซึ่งหมายความว่า กรณีทดสอบสามารถกำจัดมิวแตนท์ได้ทั้งหมด 100%

2) ประสิทธิภาพของชุดทดสอบ (Test Effectiveness หรือ E) โดยงานวิจัย [29] ได้กำหนดมาตรเพื่อวัดประสิทธิภาพของชุดทดสอบ โดยนำค่าเฉลี่ยของกรณีทดสอบที่กำจัดมิวแตนท์ $(\overline{K}_d)$ ซึ่งคำนวณได้จากสมการ

$$\overline{K}_d = \frac{\sum K_m}{D}$$

โดยที่ K_m คือ ผลรวมของจำนวนกรณีทดสอบที่สามารถกำจัดมิวแตนท์ได้
D คือ จำนวนของมิวแตนท์ที่ถูกกำจัด

เมื่อได้ค่าเฉลี่ยของกรณีทดสอบที่กำจัดมิวแตนท์แล้ว จึงคำนวณประสิทธิภาพของชุดทดสอบได้จากสมการที่ 2

$$E = MS(P,T)\ x\ \frac{\overline{K}_d}{T} \qquad \textit{สมการที่ 2}$$

โดยที่ $MS(P,T)$ คือ คะแนนมิวเทชัน
$\overline{K}_d$ คือ ค่าเฉลี่ยของกรณีทดสอบที่กำจัดมิวแตนท์
T คือ จำนวนกรณีทดสอบทั้งหมด

3) ค่าความทนทานของตัวดำเนินการมิวเทชัน (Mutation Operator Strength หรือ MOS) โดยงานวิจัย [30] ได้นำเสนอมาตรวัดนี้ เพื่อทดสอบตัวดำเนินการมิวเทชัน จากการสร้างมิวแตนท์แล้วนำไปทดสอบกับชุดทดสอบที่เตรียมไว้ จากนั้นวัดความทนทาน

ของมิวแตนท์ที่ถูกสร้างขึ้นว่าสามารถถูกกำจัด จากชุดทดสอบจำนวนเท่าใด ซึ่งความทนทานของตัวดำเนินการมิวเทชันสามารถคำนวณได้จากสมการที่ 3

$$MOS = \frac{\#minTests}{\#muts - \#notKilled} \qquad \textit{สมการที่ 3}$$

โดยที่ $\#minTests$ คือ จำนวนของชุดทดสอบที่น้อยที่สุดที่สามารถกำจัดมิวแตนท์ที่สร้างจากตัวดำเนินการที่สนใจ ในโปรแกรมที่สนใจ

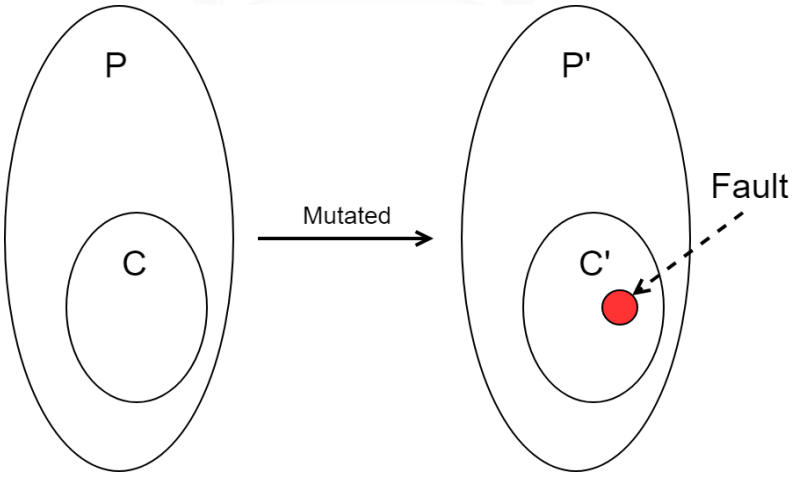
$\#muts$ คือ จำนวนของมิวแตนท์ที่สร้างได้จากตัวดำเนินการที่สนใจ

$\#notKilled$ คือ จำนวนของมิวแตนท์สมมูลที่ไม่สามารถถูกกำจัดได้จากชุดทดสอบ

โดยค่าความทนทานของตัวดำเนินการมิวเทชันมีค่าอยู่ในช่วง 0 ถึง 1 ซึ่งถ้าค่าเท่ากับ 1 นั้นแสดงว่าเป็นตัวดำเนินการมิวเทชันที่มีความทนทานสูง และสามารถถูกกำจัดด้วยจำนวนชุดทดสอบที่น้อย และมาตรวัดนี้ได้ถูกใช้งานในเครื่องมือ muJava [13] ด้วย

2.1.5 การทดสอบวีคมิวเทชัน (Weak Mutation Testing)

การทดสอบมิวเทชัน หรือ สตรองมิวเทชัน (Strong Mutation) พบว่าหากโปรแกรมความซับซ้อน อาทิ เช่น มีเงื่อนไข หรือตัวดำเนินการเป็นจำนวนมาก ซึ่งส่งผลให้ต้องใช้แรงงานและทรัพยากรที่ค่อนข้างสูง จึงมีงานวิจัย [31] ได้เสนอแนวคิดการใส่ข้อผิดพลาดลงไปซอฟต์แวร์ ในระดับหน่วยย่อย หรือ คอมโพเนนท์ (Component) ดังภาพที่ 2-11



ภาพที่ 2-11 การทดสอบวีคมิวเทชัน

โดยที่ P คือ โปรแกรมต้นฉบับ

C คือ คอมโพเนนท์ย่อยของโปรแกรม P

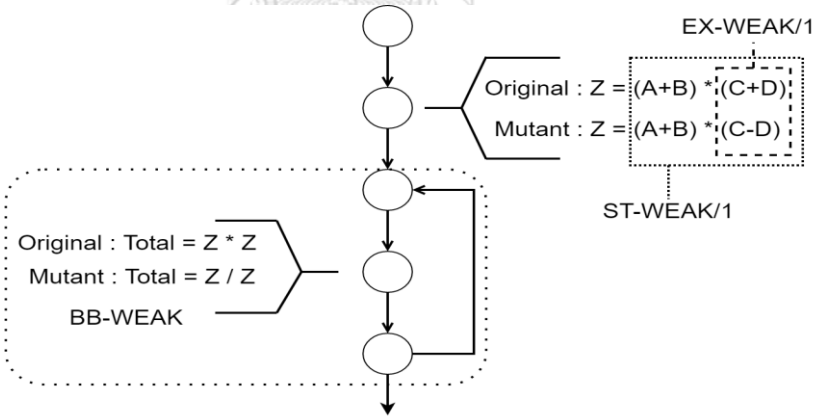
C’ คือ คอมโพเนนท์ย่อยที่ถูกใส่ข้อผิดพลาดลงไป

P’ คือ โปรแกรมที่มี C’ เป็นส่วนประกอบ

สำหรับการทดสอบนั้นต้องการให้กรณีทดสอบสามารถค้นหาข้อผิดพลาดของ C’ ออกมาให้ได้อย่างน้อย 1 ครั้ง แม้ว่า P’ ยังทำงานได้และแสดงผลลัพธ์ในขั้นตอนสุดท้ายเหมือนกับ P แต่ในงานวิจัยของ Howden [31] นั้น ยังไม่ได้กำหนดส่วนของคอมโพเนนท์ที่สามารถใส่ข้อผิดพลาดได้ แต่กำหนดลักษณะของคอมโพเนนท์ที่เข้าข่ายทั้งหมด 5 รูปแบบ ได้แก่

1) การอ้างอิงตัวแปร (Variable Reference)
2) การให้ค่าตัวแปร (Variable Assignment)
3) นิพจน์ทางคณิตศาสตร์ (Arithmetic Expression)
4) นิพจน์เชิงสัมพันธ์ (Relational Expression)
5) นิพจน์แบบบูลีน (Boolean Expression)

งานวิจัย [4] ได้มีการขยายความจากงานวิจัย [31] โดย A.J. Offutt ได้กำหนดจุดเปรียบเทียบของคอมโพเนนท์ และคอมโพเนนท์ที่เป็นมิวแตนท์ได้ 4 ระดับ ดังภาพที่ 2-12



ภาพที่ 2-12 จุดเปรียบเทียบของคอมโพเนนท์ของการทดสอบแบบวีคมิวเทชัน

1) EX-WEAK/1 (Expression-WEAK/1) เป็นการเปรียบเทียบส่วนที่ย่อยที่สุดของโปรแกรมโดยการเปลี่ยนตัวดำเนินการในระดับนิพจน์ (Expression) และเปรียบเทียบผลลัพธ์ของนิพจน์ที่ถูกแก้ไข หลังถูกดำเนินการไปแล้ว 1 ครั้ง
2) ST-WEAK/1 (Statement-WEAK/1) เป็นการเปลี่ยนตัวดำเนินการในระดับนิพจน์ (Expression) แต่เปรียบเทียบผลลัพธ์ในระดับประโยค (Statement) หลังถูกดำเนินการไปแล้ว 1 ครั้ง

3) BB-WEAK/1 (Basic-Block-WEAK/1) เป็นการเปลี่ยนตัวดำเนินการในระดับนิพจน์ (Expression) แต่เปรียบเทียบผลลัพธ์ในระดับบล็อค (Block) หลังถูกดำเนินการไปแล้ว 1 ครั้ง
4) BB-WEAK/N (Basic-Block-WEAK/N) เป็นการเปลี่ยนตัวดำเนินการในระดับนิพจน์ (Expression) แต่เปรียบเทียบผลลัพธ์ในระดับบล็อค (Block) หลังถูกดำเนินการไปแล้ว N ครั้ง

2.1.6 ตัวดำเนินการมิวเทชัน (Mutation Operator)

ตัวดำเนินการมิวเทชันเป็นสิ่งที่นักวิจัยได้สังเกตุ และค้นพบรูปแบบที่นักพัฒนาซอฟต์แวร์มักทำผิดพลาด โดยรูปแบบของข้อผิดพลาดแตกต่างกันไปในแต่ละภาษานั้นขึ้นอยู่กับไวยากรณ์ และแนวคิดที่นำมาใช้ของภาษานั้นๆ ซึ่งจากงานวิจัยที่ผ่านมาได้มีการกำหนดตัวดำเนินการมิวเทชันของแต่ละภาษาซึ่งสามารถจัดกลุ่มได้เป็น 4 กลุ่ม ดังนี้

กลุ่มที่ 1 ภาษาโปรแกรมแบบกระบวนคำสั่ง (Procedural Programming Language)

- ภาษาฟอร์แทรน (Fortran) - จากงานวิจัย [6] ได้กำหนดตัวดำเนินการมิวเทชันสำหรับภาษาฟอร์แทรน ซึ่งสามารถแบ่งออกได้ 3 กลุ่ม ได้แก่ Statement analysis, Predicate analysis และ Coincidental correctness รวมถึงมีการพัฒนาเครื่องมือ Mothra สำหรับช่วยในการทดสอบมิวเทชัน
- ภาษาซี (C) - งานวิจัย [7] ได้กำหนดตัวดำเนินการมิวเทชันสำหรับภาษาซี ซึ่งแบ่งได้เป็น 4 กลุ่ม ได้แก่ Statement Mutation, Operator Mutation, Variable Mutation และ Constant Mutation และงานวิจัย [8] มีการพัฒนาเครื่องมือ Proteum สำหรับช่วยในการทดสอบมิวเทชัน

กลุ่มที่ 2 ภาษาโปรแกรมแบบเซต (Set-Oriented Language)

- เอสคิวแอล (SQL) - งานวิจัย [9] ได้กำหนดตัวดำเนินการมิวเทชันสำหรับเอสคิวแอล ซึ่งมีแนวคิดที่ต่างจากภาษา Fortran และ C ซึ่งเป็นภาษาเชิงกระบวนความ (procedural languages) จึงได้แบ่งตัวดำเนินการมิวเทชันเป็น 4 กล่ม ได้แก่ SQL Clause (SC), Operator Replacement (OR), NULL(NL) และ Identifier Replacement(IR) รวมถึงในงานวิจัย [10] ได้มีการสร้างเครื่องมือ SQLMutation สำหรับช่วยทดสอบมิวเทชัน และมีงานวิจัย [11] ที่ได้ประเมินตัวดำเนินการมิวเทชันของภาษาเอสคิวแอล

กลุ่มที่ 3 ภาษาโปรแกรมเชิงวัตถุ (Object-Oriented Programming Language)

- ภาษาจาวา (Java) - งานวิจัย [12] ได้กำหนดตัวดำเนินการมิวเทชันที่เกี่ยวกับการเชิงโปรแกรมเชิงวัตถุไว้ และในงานวิจัย [13] ได้มีการสร้างเครื่องมือ muJava สำหรับช่วยทดสอบมิวเทชัน
- ภาษาซีชาร์ป (C#) - งานวิจัย [14] ได้กำหนดตัวดำเนินการมิวเทชันของภาษาซีชาร์ป ซึ่งประยุกต์มาจากงานวิจัย [12] และได้มีการสร้างเครื่องมือ CREAM [15] สำหรับช่วยทดสอบมิวเทชัน
- ภาษาไพทอน (Python) - งานวิจัย [16] ได้กำหนดตัวดำเนินการมิวเทชันของภาษาไพทอน ที่ประยุกต์มาจากงานวิจัย [12, 14] และได้เพิ่มตัวดำเนินการมิวเทชันที่เป็นลักษณะเฉพาะของภาษาไพทอนดังตารางที่ 2-1 ซึ่งแสดงตัวดำเนินการมิวเทชันที่เป็นลักษณะเฉพาะของภาษาไพทอน และได้มีการสร้างเครื่องมือ MutPy สำหรับช่วยในการทดสอบ [17]

  ตารางที่ 2-1 ตัวดำเนินการมิวเทชันที่เป็นลักษณะเฉพาะของภาษาไพทอน [16]

Python Specific Feature	Mutation Operators
Decorators	DDL(Decorator Deletion)
	CDI(Classmethod Decorator Insertion)
Collection slice	SDI(Static method Decorator Insertion)
	SIR(Slice Index Removal)
Loop Reversing	RIL(Reverse Iteration Loop)

กลุ่มที่ 4 ภาษาที่แลกเปลี่ยนโดยใช้เอกซ์เอ็มแอล (XML Interchange Formats)

- แบบจำลองดับเบิลยูเอส-บีเพลส (WS-BPEL) – งานวิจัย [18] ได้กำหนดตัวดำเนินการมิวเทชันของแบบจำลองดับเบิลยูเอส-บีเพลส โดยได้จัดกลุ่มของตัวดำเนินการมิวเทชันใหม่ เนื่องจากตัวภาษามีความแตกต่างจากภาษาอื่นๆ และมีลักษณะของการทำงานเป็นกิจกรรม และงานวิจัย [19] สร้างเครื่องมือ GAmera สำหรับการทดสอบมิวเทชันแบบการเลือก (Selective Mutation) และงานวิจัย [20] สร้างเครื่องมือ WeMuTe สำหรับช่วยทดสอบมิวเทชันแบบวีคมิวเทชัน

- แบบจำลองบีพีเอ็มเอ็น (BPMN) - งานวิจัย [5] ได้กำหนดตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็น ซึ่งทางผู้วิจัยได้สังเกตุถึงข้อผิดพลาดจากการนำแบบจำลองบีพีเอ็มเอ็นไปใช้งาน โดยมีตัวดำเนินการมิวเทชัน 25 ตัวดำเนินการ ซึ่งสามารถแบ่งได้ทั้งหมด 4 กลุ่ม ได้แก่
  1) ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร (Identifier Mutation Operator - I)
  2) ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข (Expression Mutation Operator - E)
  3) ตัวดำเนินการมิวเทชันเชิงกิจกรรม (Activity Mutation Operator - A)
  4) ตัวดำเนินการมิวเทชันเกี่ยวกับข้อยกเว้นและเหตุการณ์ (Exception and Event Mutation Operators - X)

จากการศึกษาพบว่าตัวดำเนินการมิวเทชันที่พบในงานวิจัย [5-20] ไม่ได้ถูกนำมาใช้ร่วมกันได้ทั้งหมด เนื่องจากแนวคิดของแต่ละภาษาแตกต่างกัน อาทิ เช่น การเชิงโปรแกรมเชิงวัตถุ หรือการทำงานที่มีลักษณะเชิงกิจกรรม เป็นต้น แต่ยังมีตัวดำเนินการมิวเทชันที่ใช้สามารถประยุกต์ใช้งานได้ ได้แก่ ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร และตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข ที่สามารถพบได้ในภาษาอื่นๆ เช่น ภาษาซี, ภาษาจาวา, ภาษาไพทอน, แบบจำลองดับเบิลยูเอส-บีเพลส และแบบจำลองบีพีเอ็มเอ็น เป็นต้น

นอกจากนี้พบว่าการทดสอบตัวดำเนินการมิวเทชันของแต่ละภาษา จำเป็นต้องสร้างเครื่องมือขึ้นเพื่อช่วยลดภาระของนักทดสอบให้ได้มากที่สุด

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัย “Mutation Operators in BPMN Model”

งานวิจัยโดย Phra Pridsadi Tadeesom และ Taratip Suwannasart (2017:45-48) [5] ได้นำเสนอตัวดำเนินการมิวเทชันสำหรับแบบจำลองบีพีเอ็มเอ็น โดยเริ่มจากการวิเคราะห์จุดผิดพลาดที่เกิดขึ้นในแบบจำลองบีพีเอ็มเอ็นที่นักพัฒนามักทำผิด ศึกษาการสร้างกรณีทดสอบที่ใช้กับแบบจำลองบีพีเอ็มเอ็น และศึกษาการสร้างตัวดำเนินการมิวเทชันสำหรับแบบจำลองดับเบิลยูเอส-บีเพลส โดยทีมวิจัยกำหนดตัวดำเนินการมิวเทชันไว้ 25 ตัวดำเนินการ และได้แบ่งตัวดำเนินการมิวเทชันออกเป็น 4 กลุ่ม ได้แก่

1) ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร (Identifier Mutation Operator) ซึ่งตัวดำเนินการมิวเทชันประเภทนี้มีจำนวน 2 ตัวดำเนินการ ดังตารางที่ 2-2

ตารางที่ 2-2 ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
IVR	Identifier Value Replacement	เปลี่ยนตัวแปรด้วยตัวแปรอื่นที่มีชนิดเดียวกันใน <extensionElement>
ITR	Identifier different Type Replacement	เปลี่ยนตัวแปรด้วยตัวแปรอื่นที่ต่างชนิดกันใน <extensionElement>

2) ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข (Expression Mutation Operator) ซึ่งตัวดำเนินการมิวเทชันประเภทนี้สามารถแบ่งได้อีก 4 ประเภทย่อย ได้แก่

2.1) ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic Operator)

2.2) ตัวดำเนินการเชิงสัมพันธ์ (Relational Operator)

2.3) ตัวดำเนินการเชิงตรรกศาสตร์ (Logical Operator)

2.4) ตัวดำเนินการเกี่ยวกับเวลา (Deadline Operator)

ตัวดำเนินการมิวเทชันประเภทนี้มีจำนวน 7 ตัวดำเนินการ ดังตารางที่ 2-3

ตารางที่ 2-3 ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
EAR	Expression Arithmetic Replacement	เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) ที่อยู่ในนิพจน์ใน <conditionExpression>
ERR	Expression Relational Replacement	เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ที่อยู่ในนิพจน์ใน<conditionExpression>
ELR	Expression Logical Replacement	เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ที่อยู่ในนิพจน์ใน <conditionExpression>
ETA	Expression Time Adjustment	ปรับระยะเวลาของ <timerDuration> ภายใต้ <timerEventDefinition> ด้วย 0 ,ครึ่งหนึ่งของเวลาเดิมและค่าเวลาที่มากกว่าเดิมหนึ่งหน่วย
EDA	Expression Date Adjustment	ปรับวันที่ของ <timerDate> ภายใต้ <timerEventDefinition> ด้วยวันในอดีต

ตารางที่ 2-3 ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข (ต่อ)

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
ERA	**E**xpression **R**epetition **A**djustment	ปรับจำนวนการทำซ้ำของ <timerCycle> ภายใต้ <timerEventDefinition> ด้วย 0 หรือครึ่งหนึ่งของจำนวนเดิม และค่าเวลาที่มากกว่าเดิมหนึ่งหน่วย
ECA	**E**xpression **C**ycle **A**djustment	ปรับช่วงระยะเวลา <timerCycle> ภายใต้ <timerEventDefinition> ด้วย 0, ค่าครึ่งหนึ่งของเวลาเดิมและค่าเวลาที่มากกว่าเดิมหนึ่งหน่วย

3) ตัวดำเนินการมิวเทชันเชิงกิจกรรม (Activity Mutation Operator) ซึ่งตัวดำเนินการมิวเทชันประเภทนี้มีจำนวน 15 ตัวดำเนินการ ดังตารางที่ 2-4

ตารางที่ 2-4 ตัวดำเนินการมิวเทชันเชิงกิจกรรม

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
ASR	**A**ctivity **S**equential **R**eplacement	เปลี่ยนค่าแอตทริบิวต์ “isSequential” ใน <multiInstanceLoopCharacteristics> ระหว่าง “true” และ “false”
ACR	**A**ctivity **C**ardinality **R**eplacement	เปลี่ยนค่า <loopCardinality> ภายใต้ <multiInstanceLoopCharacteristics> ด้วย 0, ค่าครึ่งหนึ่งของจำนวนเดิมและค่ามากกว่าเดิมหนึ่งหน่วย
AAM	**A**ctivity **A**rithmetic replacement in **M**ultiInstanceLoop	เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) ที่อยู่ในนิพจน์ ใน <completionCondition> ภายใต้ <multiInstanceLoopCharacteristics>
ARM	**A**ctivity **R**elational replacement in **M**ultiInstanceLoop	เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ที่อยู่ในนิพจน์ ใน <completionCondition> ภายใต้ <multiInstanceLoopCharacteristics>
ALM	**A**ctivity **L**ogical replacement in **M**ultiInstanceLoop	เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ที่อยู่ในนิพจน์ ใน<completionCondition> ภายใต้ <multiInstanceLoopCharacteristics>

ตารางที่ 2-4 ตัวดำเนินการมิวเทชันเชิงกิจกรรม (ต่อ)

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
ATR	**A**ctivity **T**est **R**eplacement	เปลี่ยนค่าแอตทริบิวต์ “testBefore” ใน <standardLoopCharacteristics> ระหว่าง “true” และ “false”
AMR	**A**ctivity loop **M**aximum **R**eplacement	ปรับค่าแอตทริบิวต์ “loopMaximum” ใน <standardLoopCharacteristics> ด้วย 0, ค่าครึ่งหนึ่งของจำนวนเดิมและค่ามากกว่าเดิมหนึ่งหน่วย
AAS	**A**ctivity **A**rithmetic replacement in **S**tandard loop	เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) ที่อยู่ในนิพจน์ ใน<loopCondition> ภายใต้ <standardLoopCharacteristics>
ARS	**A**ctivity **R**elational replacement in **S**tandard loop	เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ที่อยู่ในนิพจน์ ใน <loopCondition> ภายใต้ <standardLoopCharacteristics>
ALS	**A**ctivity **L**ogical replacement in **S**tandard loop	เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ที่อยู่ในนิพจน์ ใน <loopCondition> ภายใต้ <standardLoopCharacterisics>
AAA	**A**ctivity **A**rithmetic replacement in **A**dhoc subprocess	เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /,%) ที่อยู่ในนิพจน์ ใน <completionCondition> ภายใต้ <adHocSubProcess>
ARA	**A**ctivity **R**elational replacement in **A**dhoc subprocess	เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ที่อยู่ในนิพจน์ ใน <completionCondition> ภายใต้ <adHocSubProcess>
ALA	**A**ctivity **L**ogical replacement in **A**dhoc subprocess	เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ที่อยู่ในนิพจน์ ใน <completionCondition> ภายใต้ <adHocSubProcess>

ตารางที่ 2-4 ตัวดำเนินการมิวเทชันเชิงกิจกรรม (ต่อ)

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
AOR	**A**ctivity **O**rdering **R**eplacement	ปรับค่าแอตทริบิวต์ “ordering” ใน <adHocSubProcess> ระหว่าง “Parallel” และ “Sequential”
ARR	**A**ctivity cancel **R**emaining Instances **R**eplacement	ปรับค่าแอตทริบิวต์ “cancelRemainingInstances” ใน <adHocSubProcess> ระหว่าง “true” และ “false”

4) ตัวดำเนินการมิวเทชันเกี่ยวกับข้อยกเว้นและเหตุการณ์ (Exception and Event Mutation Operators) ซึ่งตัวดำเนินการมิวเทชันประเภทนี้มีจำนวน 1 ตัวดำเนินการ ดังตารางที่ 2-5

ตารางที่ 2-5 ตัวดำเนินการมิวเทชันเกี่ยวกับข้อยกเว้นและเหตุการณ์

รหัส	ชื่อตัวดำเนินการ	คำอธิบาย
XBR	e**X**ception and event **B**ehavior **R**eplacement	ปรับค่าแอตทริบิวต์ “behavior” ใน <multiInstanceLoopCharacteristics> ด้วย “None”, “One”, “All” และ “Complex”

งานวิจัยนี้ยังมีของการสร้างมิวแตนท์จากเครื่องมือที่ยังจำกัดการสร้างมิวแตนท์ในตัวดำเนินการมิวเทชันที่มีนิพจน์เชิงตรรกศาสตร์ (ELR) อย่างมาก 1 ตำแหน่ง มีนิพจน์เชิงสัมพันธ์ (ERR) อย่างมาก 2 ตำแหน่ง และนิพจน์ทางคณิตศาสตร์ (EAR) อย่างมาก 2 ตำแหน่ง และยังขาดการประเมินผลประสิทธิผลตัวดำเนินการมิวเทชันที่ได้กำหนดไว้ในงานวิจัยนี้

2.2.2 งานวิจัย “Quantitative Evaluation of Mutation Operators for WS-BPEL”

งานวิจัยโดย Antonia Estero-Botaro และคณะ (2009:97-106) [19] ได้นำเสนอการการประเมินเชิงปริมาณเชิงคุณภาพของตัวดำเนินการมิวเทชันสำหรับแบบจำลองดับเบิลยูเอส-บีเพลส 2.0 ที่ได้กำหนดไว้จากงานวิจัย [18] และมีการใช้เครื่องมือ GAmera ซึ่งเป็นเครื่องมือสำหรับทดสอบมิวเทชันด้วยวิธีการทดสอบมิวเทชันแบบการเลือก (Selective Mutation) ซึ่งเป็นเทคนิคการลดจำนวนมิวแตนท์ สำหรับการทดสอบแบบมิวเทชัน โดยการเลือกตัวดำเนินการมิวเทชันที่มีผลกับการสร้างมิวแตนท์ มีการกำหนดคำถามสำหรับการวิจัย ดังนี้

1) ตัวดำเนินการมิวเทชันสร้างมิวแตนท์ที่ไม่สามารถทดสอบได้(stillborn mutants) หรือไม่ ?
2) ตัวดำเนินการมิวเทชันไหนที่เหมาะสมกับวิธีการมิวเทชันแบบการเลือก
3) ตัวดำเนินการมิวเทชันแต่ละตัวสามารถสร้างมิวแตนท์ที่มีคุณภาพ หรือไม่ ?

โดยการทดสอบได้ทดสอบกับแบบจำลองดับเบิลยูเอส-บีเพลส 2.0 จำนวน 3 แบบจำลอง ได้แก่ แบบจำลองอนุมัติการกู้ยืม, แบบจำลองตลาดสินค้า และแบบจำลองการค้นหาเมต้า โดยใช้เครื่องมือ GAmera [19] สร้างมิวแตนท์ และทดสอบมิวแตนท์กับกรณีทดสอบที่สร้างขึ้น เพื่อดูประสิทธิภาพของกรณีทดสอบ

งานวิจัยนี้ได้นำเสนอแนวทางการทดสอบ และการประเมินตัวดำเนินการมิวเทชัน รวมถึงการกำหนดมาตรวัดสำหรับการวัดตัวดำเนินการมิวเทชันที่ได้นิยามไว้จากงานวิจัย [18] ด้วย โดยการใช้คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ ซึ่งได้มีการอ้างไว้ในบทที่ 2 หัวข้อการทดสอบมิวเทชัน – ประสิทธิภาพของกรณีทดสอบ

บทที่ 3
การสร้างมิวแตนท์

บทนี้จะเริ่มต้นด้วยการนำเสนอการจัดตัวดำเนินการมิวเทชันสำหรับการทดสอบวิคมิวเทชัน การสร้างมิวแตนท์ และภาพรวมของเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็น

3.1 การจัดตัวดำเนินการมิวเทชันสำหรับการทดสอบวีคมิวเทชัน

จากผลของงานวิจัย [5] ได้เสนอตัวดำเนินการมิวเทชันสำหรับแบบจำลองบีพีเอ็มเอ็น แต่ยังไม่มีการนำการไปประยุกต์ใช้กับการทดสอบวีคมิวเทชัน ดังนั้นในงานวิจัยนี้ได้นำตัวดำเนินการของงานวิจัย [5] และการทดสอบวีคมิวเทชัน [4] ตามแนวคิดของ A.J. Offutt มาประยุกต์ใช้ โดยมีการตรวจทานกับข้อกำหนดของแบบจำลองบีพีเอ็มเอ็น และได้แสดงรายละเอียดตามตารางที่ 3-1 ดังนี้

1) EX-WEAK/1 เป็นการเปรียบเทียบในระดับนิพจน์ มิวแตนท์สามารถถูกสร้าง และทดสอบได้จากนิพจน์ที่ถูกประกาศในแท็กของแบบจำลองบีพีเอ็มเอ็นดังนี้ <conditionExpression>, <timerDuration>, <timerDate>, <timerCycle> และ <completionCondition> โดยตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นประเภทระบุค่าตัวแปร และ ระบุนิพจน์เงื่อนไขทั้งหมด และตัวดำเนินการมิวเทชันเชิงกิจกรรมบางตัวสามารถประยุกต์ใช้กับการทดสอบวีคมิวเทชันแบบ EX-WEAK/1 ได้แก่ IVR, ITR, EAR, ERR, ELR, ETA, EDA, AAM, ARM, ALM, AAS, ARS, ALS, AAA, ARA และ ALA
2) ST-WEAK/1 เป็นการเปรียบเทียบในระดับประโยค หรือ กิจกรรมของแบบจำลองบีพีเอ็มเอ็น โดยตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นที่สามารถใช้วิธีการทดสอบวีคมิวเทชันแบบ EX-WEAK/1 สามารถประยุกต์ใช้การทดสอบวีคมิวเทชันแบบ ST-WEAK/1 ได้เช่นกัน รวมถึงตัวดำเนินการเกี่ยวกับข้อยกเว้นและเหตุการณ์ ได้แก่ XBR แต่ตัวดำเนินการมิวเทชันเชิงกิจกรรมบางตัว ได้แก่ ASR, ACR, ATR, AMR, AOR และ ARR ไม่สามารถใช้กับระดับการทดสอบนี้ได้ เนื่องจากต้องอาศัยการทดสอบแบบ BB-WEAK/1 และ BB-WEAK/N ในการทดสอบยกตัวอย่าง เช่น ตัวดำเนินการ ASR ที่เปลี่ยนพฤติกรรมของ <multiInstanceLoopCharacteristics> โดยแก้ไขแอตทริบิวต์ “isSequential” ซึ่งมีรูปแบบการทำงานในระดับบล็อคให้มีค่าระหว่าง “true” และ “false”
3) BB-WEAK/1 เป็นการเปรียบเทียบในระดับบล็อค ซึ่งในแบบจำลองบีพีเอ็มเอ็นที่เปรียบเทียบได้จากกลุ่มของกิจกรรม หรือ กิจกรรมที่มีรูปแบบทำวนซ้ำ ได้แก่ <multiInstanceLoopCharacteristics> และ <standardLoopCharacteristics> โดยพบว่าตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นที่สามารถใช้วิธีการทดสอบวีค

มิวเทชันแบบ EX-WEAK/1 และ ST-WEAK/1 สามารถประยุกต์ใช้การทดสอบวีคมิวเทชันแบบ BB-WEAK/1 ได้เช่นกัน

4) BB-WEAK/N เป็นการเปรียบเทียบในระดับบล็อคเหมือนกับ BB-WEAK/1 แต่เนื่องจากเป็นการวิเคราะห์ในขอบเขตที่กว้างกว่า เพราะทดสอบมิวแตนท์ครั้งแรกด้วย BB-WEAK/1 อาจจะไม่สามารถกำจัดมิวแตนท์ได้ จึงต้องอาศัยวิธีการทดสอบแบบ BB-WEAK/N เพื่อพิจารณาค่าของตัวแปรที่สนใจในแต่ละรอบของการทำซ้ำ

ตารางที่ 3-1 ตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นที่สามารถใช้กับวีคมิวเทชัน

ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน			
	EX-WEAK/1	ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
ตัวดำเนินการมิวเทชันที่ระบุค่าตัวแปร				
IVR	✓	✓	✓	✓
ITR	✓	✓	✓	✓
ตัวดำเนินการมิวเทชันที่ระบุนิพจน์เงื่อนไข				
EAR	✓	✓	✓	✓
ERR	✓	✓	✓	✓
ELR	✓	✓	✓	✓
ETA	✓	✓	✓	✓
EDA	✓	✓	✓	✓
ERA	-	-	✓	✓
ECA	-	-	✓	✓

ตารางที่ 3-1 ตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นที่สามารถใช้กับวีคมิวเทชัน (ต่อ)

ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน			
	EX-WEAK/1	ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
ตัวดำเนินการมิวเทชันเชิงกิจกรรม				
ASR	-	-	✓	✓
ACR	-	-	✓	✓
AAM	✓	✓	✓	✓
ARM	✓	✓	✓	✓
ALM	✓	✓	✓	✓
ATR	-	-	✓	✓
AMR	-	-	✓	✓
AAS	✓	✓	✓	✓
ARS	✓	✓	✓	✓
ALS	✓	✓	✓	✓
AAA	✓	✓	✓	✓
ARA	✓	✓	✓	✓
ALA	✓	✓	✓	✓
AOR	-	-	✓	✓
ARR	-	-	✓	✓
ตัวดำเนินการมิวเทชันเกี่ยวกับข้อยกเว้นและเหตุการณ์				
XBR	-	✓	✓	✓

3.2 การสร้างมิวแตนท์

ในส่วนนี้จะอธิบายการสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นในแต่ละตัวดำเนินการ โดยใส่ข้อผิดพลาดเพียงหนึ่งที่ต่อหนึ่งแบบจำลองดังนี้

1) ตัวดำเนินการมิวเทชัน IVR เป็นตัวดำเนินการมิวเทชันที่ทำหน้าที่เปลี่ยนตัวแปรในแบบจำลองบีพีเอ็มเอ็นจากตัวแปรที่สนใจไปเป็นอีกตัวแปรที่มีประเภทเดียวกัน โดยสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ User Task จากภาพที่ 3-1 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท IVR ได้แสดงตัวอย่างการสร้างมิวแตนท์ที่เปลี่ยนตัวแปรใน Sequence Flow จาก INPUT_A ไปเป็น INPUT_B โดยที่ตัวแปร INPUT_A และ INPUT_B มีชนิดข้อมูลประเภท String เหมือนกัน

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
        ...
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A=='complete'}]]>
</bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
        ...
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_B=='complete'}]]>
</bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

ภาพที่ 3-1 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท IVR

2) ตัวดำเนินการมิวเทชัน ITR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนชนิดของตัวแปรที่ถูกประกาศในแท็ก <extensionElement> โดยสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ User Task จากภาพที่ 3-2 ได้แสดงตัวอย่างการสร้างมิวแตนท์ที่ถูกเปลี่ยนประเภทของตัวแปร INPUT_A จากเดิมที่มีชนิดข้อมูลประเภท string ไปเป็น long

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
     <bpmn:extensionElements>
          <camunda:formField id="INPUT_A" label="INPUT_A" type="string">
          </camunda:formField>
    </bpmn:extensionElements>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
     <bpmn:extensionElements>
        <camunda:formData>
          <camunda:formField id="INPUT_A" label="INPUT_A" type="long">
          </camunda:formField>
    </bpmn:extensionElements>
</bpmn:process>
```

ภาพที่ 3-2 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ITR

3) ตัวดำเนินการมิวเทชัน EAR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) ที่อยู่ใน <conditionExpression> โดยสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Sequence Flow

จากภาพที่ 3-3 ได้แสดงตัวอย่างการสร้างมิวแตนท์จากเงื่อนไข INPUT_A + INPUT_B จากเดิมที่ใช้เครื่องหมายบวก (+) เปลี่ยนเป็น INPUTA – INPUTB ที่เป็นเครื่องหมายลบ (-) แทน

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A + INPUT_B >=
5}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A - INPUT_B >=
5}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

ภาพที่ 3-3 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท EAR

4) ตัวดำเนินการมิวเทชัน ERR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ที่อยู่ใน <conditionExpression> โดยสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Sequence Flow

จากภาพที่ 3-4 ได้แสดงตัวอย่างการสร้างมิวแตนท์ จากเงื่อนไข INPUT_A + INPUT_B >= 5 จากเดิมที่ใช้เครื่องหมายมากกว่า หรือเท่ากับ (>=) เปลี่ยนเป็น INPUT_A + INPUT_B > 5 ที่เป็นเครื่องหมายมากกว่า (>) แทน และสร้างมิวแตนท์ที่เหลือโดยเปลี่ยนเครื่องหมายเป็น น้อยกว่า (<), น้อยกว่า หรือเท่ากับ (<=), เท่ากับ (==) และ ไม่เท่ากับ (!=) แทนตามลำดับ

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A + INPUT_B >=
5}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A + INPUT_B >
5}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

ภาพที่ 3-4 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ERR

5) ตัวดำเนินการมิวเทชัน ELR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ที่อยู่ใน <conditionExpression> โดยสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Sequence Flow

จากภาพที่ 3-5 ได้แสดงตัวอย่างการสร้างมิวแตนท์ จากเงื่อนไข INPUT_A=='complete' || INPUT_A == 'incomplete' จากเครื่องหมายหรือ (||) แทนที่ด้วยเครื่องหมายและ (&&)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A=='complete' ||
INPUT_A=='incomplete'}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:sequenceFlow id="SequenceFlow_0v1ickp" sourceRef="Task_0pfwqz5"
targetRef="ExclusiveGateway_0s2b565">
      <bpmn:conditionExpression
xsi:type="bpmn:tFormalExpression"><![CDATA[${INPUT_A=='complete' &&
INPUT_A=='incomplete'}]]></bpmn:conditionExpression>
    </bpmn:sequenceFlow>
</bpmn:process>
```

ภาพที่ 3-5 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ELR

6) ตัวดำเนินการมิวเทชัน ETA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนระยะเวลาของ <timerDuration> ภายใต้ <timerEventDefinition> ด้วย 0 , ครึ่งหนึ่งของเวลาเดิมและค่าเวลาที่มากกว่าเดิมหนึ่งหน่วย แต่ต้องอยู่ในรูปแบบมาตรฐาน ISO 8601 โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ timer Start Event, Catching timer Intermediate Event และ Boundary timer Intermediate Event

จากภาพที่ 3-6 ได้แสดงตัวอย่างการสร้างมิวแตนท์จากกิจกรรม “Update Loan Application” ได้มี Boundary timer Intermediate Event เพื่อบอกว่า ถ้ากระบวนการค้างที่กิจกรรมนี้เกิน 5 วัน (ISO8601 = P5D) ให้ออกจากกิจกรรมนี้ โดยในแบบจำลองมิวแตนท์มีการเปลี่ยนค่าจาก 5 วันเป็นไป 6 วัน

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
       ...
    <bpmn:userTask id="Task_0pfwqz5" name="Update Loan Application &#10;"
camunda:assignee="Mary">
      <bpmn:incoming>SequenceFlow_16aequc</bpmn:incoming>
      <bpmn:outgoing>SequenceFlow_0v1ickp</bpmn:outgoing>
    </bpmn:userTask>
    <bpmn:boundaryEvent id="BoundaryEvent_0r77hdg" name="5 Days"
attachedToRef="Task_0pfwqz5">
      <bpmn:outgoing>SequenceFlow_01kbekl</bpmn:outgoing>
      <bpmn:timerEventDefinition>
        <bpmn:timeDuration
xsi:type="bpmn:tFormalExpression">P5D</bpmn:timeDuration>
      </bpmn:timerEventDefinition>
    </bpmn:boundaryEvent>
       ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
       ...
    <bpmn:userTask id="Task_0pfwqz5" name="Update Loan Application &#10;"
camunda:assignee="Mary">
      <bpmn:incoming>SequenceFlow_16aequc</bpmn:incoming>
      <bpmn:outgoing>SequenceFlow_0v1ickp</bpmn:outgoing>
    </bpmn:userTask>
    <bpmn:boundaryEvent id="BoundaryEvent_0r77hdg" name="5 Days"
attachedToRef="Task_0pfwqz5">
      <bpmn:outgoing>SequenceFlow_01kbekl</bpmn:outgoing>
      <bpmn:timerEventDefinition>
        <bpmn:timeDuration
xsi:type="bpmn:tFormalExpression">P6D</bpmn:timeDuration>
      </bpmn:timerEventDefinition>
    </bpmn:boundaryEvent>
       ...
</bpmn:process>
```

ภาพที่ 3-6 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ETA

7) ตัวดำเนินการมิวเทชัน EDA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนวันที่ใน <timerDate> ภายใต้ <timerEventDefinition> ด้วยวันในอดีต แต่ต้องอยู่ในมาตรฐาน ISO 8601 โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ timer Start Event, Catching timer Intermediate Event และ Boundary timer Intermediate Event

จากภาพที่ 3-7 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Timer Start Event ซึ่งกำหนดให้กระบวนการเริ่มทำงานในวันที่ 13 เดือน 4 ปีคริสต์ศักราช 2018 เวลา 12.13 โดยในแบบจำลองมิวแตนท์มีการเปลี่ยนวันที่จากเดิมวันที่ 13 แทนที่ด้วยวันในอดีตวันที่ 12

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">2018-04-13T12:13
        </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">2018-04-12T12:13
       </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

ภาพที่ 3-7 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท EDA

8) ตัวดำเนินการมิวเทชัน ERA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนจำนวนการทำซ้ำของ <timerCycle> ภายใต้ <timerEventDefinition> ด้วย 0 หรือครึ่งหนึ่งของจำนวนเดิม แต่ต้องอยู่ในมาตรฐาน ISO 8601 โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ timer Start Event, Catching timer Intermediate Event และBoundary timer Intermediate Event

จากภาพที่ 3-8 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Timer Start Event ซึ่งมีการทำงานจำนวน 4 รอบ ทุกๆ 10 นาที โดยแบบจำลองมิวแตนท์ได้เปลี่ยนการทำซ้ำจากเดิม 4 รอบเป็น 2 รอบ

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">R4/PT10M
        </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">R2/PT10M
        </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

ภาพที่ 3-8 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ERA

9) ตัวดำเนินการมิวเทชัน ECA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนช่วงระยะเวลา <timerCycle> ภายใต้ <timerEventDefinition> ด้วย 0, ค่าครึ่งหนึ่งของเวลาเดิมและค่าเวลาที่มากกว่าเดิมหนึ่งหน่วย แต่ต้องอยู่ในรูปแบบของมาตรฐาน ISO 8601 โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ timer Start Event, Catching timer Intermediate Event และ Boundary timer Intermediate Event

จากภาพที่ 3-9 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของตัว Timer Start Event มีการทำงานจำนวน 8 รอบทุกๆ 10 นาที โดยในแบบจำลองมิวแตนท์เปลี่ยนระยะเวลาจากเดิม 10 นาที เป็น 5 นาที

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">R8/PT10M
        </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
    <bpmn:startEvent id="StartEvent_1">
      <bpmn:timerEventDefinition>
        <bpmn:timeDate xsi:type="bpmn:tFormalExpression">R8/PT5M
       </bpmn:timeDate>
      </bpmn:timerEventDefinition>
    </bpmn:startEvent>
</bpmn:process>
```

ภาพที่ 3-9 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ECA

10) ตัวดำเนินการมิวเทชัน ASR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนค่าแอตทริบิวต์ “isSequential” ของแท็ก <multiInstanceLoopCharacteristics> ระหว่างค่า “true” กับ “false” ซึ่ง isSequential เป็นกำหนดรูปแบบการทำงานของกิจกรรม หากค่า isSequential เป็น false หมายความว่า กิจกรรมนั้นต้องทำงานตามลำดับ แต่ถ้ากำหนดค่าเป็น true หมายความว่ากิจกรรมนั้นสามารถทำงานแบบขนานได้ สำหรับตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task , Script Task, และ Sub-Process

จากภาพ 3-10 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ User Task “Review & Update” ที่มีการกำหนดค่าของแอตทริบิวต์ “isSequential” ภายใต้แท็ก <multiInstanceLoopCharacteristics> ของ User Task “Review & Update” มีค่าเป็น false แต่ในแบบจำลองมิวแตนท์เปลี่ยนค่าเป็น true

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:userTask id="Task_191p4jo" name="Review & Update">
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">3</bpmn:loopCardinality>
    </bpmn:multiInstanceLoopCharacteristics>
  <bpmnn:userTask>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:userTask id="Task_191p4jo" name="Review & Update">
    <bpmn:multiInstanceLoopCharacteristics isSequential="true">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">3</bpmn:loopCardinality>
    </bpmn:multiInstanceLoopCharacteristics>
  <bpmnn:userTask >
</bpmn:process>
```

ภาพที่ 3-10 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ASR

11) ตัวดำเนินการมิวเทชัน ACR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนค่า <loopCardinality> ภายใต้แท็ก <multiInstanceLoopCharacteristics> ด้วย 0, ค่าครึ่งหนึ่งของจำนวนเดิม และค่ามากกว่าเดิมหนึ่งหน่วย ซึ่งค่า Loop Cardinality นั้นเป็นการกำหนดจำนวนของ Instance กิจกรรมที่สามารถเกิดขึ้นได้ โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-11 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ User Task “Review & Update” ที่ค่า Loop Cardinality ภายใต้แท็ก <multiInstanceLoopCharacteristics> เท่ากับ 6 แต่ในแบบจำลองมิวแตนท์เปลี่ยนค่าเป็น 3

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:userTask id="Task_191p4jo" name="Review & Update">
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">6</bpmn:loopCardinality>
    </bpmn:multiInstanceLoopCharacteristics>
  <bpmnn:userTask>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:userTask id="Task_191p4jo" name="Review & Update">
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">3</bpmn:loopCardinality>
    </bpmn:multiInstanceLoopCharacteristics>
  <bpmnn:userTask>
</bpmn:process>
```

ภาพที่ 3-11 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ACR

12) ตัวดำเนินการมิวเทชัน AAM เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, / , %) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <multiInstanceLoopCharacteristics> โดย completion condition นั้นเป็นการกำหนดเงื่อนไขการสิ้นสุดของกิจกรรมที่สนใจ สำหรับตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-12 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ User Task “Approved” ซึ่งกำหนดว่าถ้ากรรมการจำนวนครึ่งหนึ่งเข้าร่วมอนุมัติโครงการแล้วกิจกรรม ”Approved” ถึงจะเสร็จสิ้น โดยที่จำนวนกรรมการมีทั้งหมด 8 ท่าน (ตัวแปร nrOfInstances = 8) และจำนวนของกรรมการที่เข้าร่วมอนุมัติโครงการ (ตัวแปร nrOfCompletedInstances) และมีการกำหนดเงื่อนไขของ Complete Completion คือ ${nrOfCompletedInstances/nrOfInstances >= 0.5} แต่ในแบบจำลองมิวแตนท์เปลี่ยนจากเครื่องหมายหาร (/) เป็นเครื่องหมายคูณ (*)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
    ...
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${nrOfCompletedInstances/
nrOfInstances >= 0.5 }]]></bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
       ...
 </bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
     ...
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${nrOfCompletedInstances*
nrOfInstances >= 0.5 }]]></bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
       ...
 </bpmn:process>
```

ภาพที่ 3-12 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ACR

13) ตัวดำเนินการมิวเทชัน ARM เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <multiInstanceLoopCharacteristics> โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-13 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ User Task “approved” ซึ่งกำหนด completion Condition เป็น ${nrOfCompletedInstances/nrOfInstances >= 0.5} แต่ในแบบจำลองมิวแตนท์เปลี่ยนจากเครื่องหมายมากกว่า หรือเท่ากับ (>=) เป็นเครื่องหมายมากกว่า (>)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  …
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${nrOfCompletedInstances/
nrOfInstances >= 0.5 }]]></bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
    …
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
   …
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${nrOfCompletedInstances/
nrOfInstances > 0.5 }]]></bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
    …
</bpmn:process>
```

ภาพที่ 3-13 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ARM

14) ตัวดำเนินการมิวเทชัน ALM เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <multiInstanceLoopCharacteristics> โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-14 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ User Task ”Approved” ซึ่งเป็นตัวอย่างจากภาพที่ 3-11 ซึ่งกำหนด completion Condition เป็น ${nrOfCompletedInstances/nrOfInstances >= 0.5|| nrOfCompletedInstances > 4} แต่ในแบบจำลองมิวแตนท์มีการเปลี่ยนจากเครื่องหมายหรือ (||) เป็นเครื่องหมายและ (&&)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
      ...
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[
                ${nrOfCompletedInstances/nrOfInstances >= 0.5
                            || nrOfCompletedInstances > 4}  ]]>
      </bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
         ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
      ...
  <bpmn:task id="Task_191p4jo" name="Approved">
    <bpmn:incoming>SequenceFlow_15x3kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1ggx57q</bpmn:outgoing>
    <bpmn:multiInstanceLoopCharacteristics isSequential="false">
      <bpmn:loopCardinality
xsi:type="bpmn:tFormalExpression">8</bpmn:loopCardinality>
      <bpmn:completionCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[
                ${nrOfCompletedInstances/nrOfInstances >= 0.5
                            && nrOfCompletedInstances > 4}  ]]>
      </bpmn:completionCondition>
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:task>
         ...
</bpmn:process>
```

ภาพที่ 3-14 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ALM

15) ตัวดำเนินการมิวเทชัน ATR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนค่าแอตทริบิวต์ “testBefore” ใน <standardLoopCharacterisics> ระหว่าง “true” และ “false” โดยที่ testBefore เป็นแอตทริบิวต์กำหนดลำดับการตรวจเงื่อนไขใน <loopCondition> หากค่า testBefore เป็น true หมายความว่ามีการตรวจสอบเงื่อนไขก่อนเข้ากิจกรรมในแต่ละรอบการทำงาน ถ้าในมุมมองของการเขียนโปรแกรมมีแนวคิดเหมือนคำสั่ง while-loop ในทางกลับกันหากกำหนดค่าเป็น false หมายความว่า ให้ตรวจสอบเงื่อนไขหลังจากกิจกรรมทำงานในแต่ละรอบเสร็จสิ้น โดยตัวดำเนินการมิวเทชันประเภทนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-15 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Service Task ”Generate GL File” โดยกำหนดค่าของแอตทริบิวต์ “testBefore” เป็น true แต่ในแบบจำลองมิวแตนท์มีการเปลี่ยนค่าเป็น false

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate GL File">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${LOOP_COUNTER >
5}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
 </bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="false">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate GL File">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="false" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${LOOP_COUNTER >
5}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
 </bpmn:process>
```

ภาพที่ 3-15 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ATR

16) ตัวดำเนินการมิวเทชัน AMR เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนค่าแอตทริบิวต์ “loopMaximum” ใน <standardLoopCharacterisics> ด้วย 0, ค่าครึ่งหนึ่งของจำนวนเดิมและค่ามากกว่าเดิมหนึ่งหน่วย สำหรับแอตทริบิวต์ loopMaximum เป็นการกำหนดจำนวนการทำซ้ำสูงสุดของของกิจกรรม โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-16 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Script Task ”Generate GL File” โดยกำหนดค่าของแอตทริบิวต์ “loopMaximum” เท่ากับ 8 แต่ในแบบจำลองมิวแตนท์เปลี่ยนค่าเป็น 4

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate GL File">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${LOOP_COUNTER >
5}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
 </bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate GL File">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="4">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${LOOP_COUNTER >
5}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
 </bpmn:process>
```

ภาพที่ 3-16 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท AMR

17) ตัวดำเนินการมิวเทชัน AAS เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) ของแท็ก <loopCondition> ซึ่งอยู่ภายใต้แท็ก <standardLoopCharacterisics> สำหรับแท็ก <loopCondition> เป็นการกำหนดเงื่อนไขสิ้นสุดของการทำซ้ำ เมื่อเงื่อนไขนั้นเป็นจริง โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-17 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Script Task “Generate Order” โดยเงื่อนไขของ loopCondition เป็น ${TOTAL_ORDER_AMOUNT + TOTAL_VAT_AMOUNT} แต่แบบจำลองมิวแตนท์เปลี่ยนเครื่องหมายบวก และแทนที่ด้วยเครื่องหมายลบ

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${TOTAL_ORDER_AMOUNT +
TOTAL_VAT_AMOUNT > LIMIT_AMOUNT}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${TOTAL_ORDER_AMOUNT -
TOTAL_VAT_AMOUNT > LIMIT_AMOUNT}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

ภาพที่ 3-17 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท AAS

18) ตัวดำเนินการมิวเทชัน ARS เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ของแท็ก <loopCondition> ซึ่งอยู่ภายใต้แท็ก <standardLoopCharacterisics> โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-18 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของ Script Task ”Generate Order” โดยเงื่อนไขของ loopCondition เป็น ${TOTAL_ORDER_AMOUNT + TOTAL_VAT_AMOUNT > LIMIT_AMOUNT } แต่แบบจำลองมิวแตนท์เปลี่ยนเงื่อนไขจากเครื่องหมายมากกว่า (>) แทนที่ด้วยเครื่องหมายหมายมากกว่า หรือเท่ากับ (>=)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${TOTAL_ORDER_AMOUNT +
TOTAL_VAT_AMOUNT > LIMIT_AMOUNT }]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${TOTAL_ORDER_AMOUNT +
TOTAL_VAT_AMOUNT >= LIMIT_AMOUNT }]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

ภาพที่ 3-18 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ARS

19) ตัวดำเนินการมิวเทชัน ALS เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ใน <loopCondition> ภายใต้ <standardLoopCharacterisics> โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-19 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลองต้นฉบับที่ User Task “Approve Limit” โดยเงื่อนไขของ loopCondition เป็น ${APPROVED == true && TOTAL_APPROVED_LIMIT < LIMIT_AMOUNT} แต่แบบจำลองมิวแตนท์เปลี่ยนจากเครื่องหมายและ (&&) แทนที่ด้วยเครื่องหมายหรือ (||)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${APPROVED == true &&
TOTAL_APPROVED_LIMIT <= LIMIT_AMOUNT}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:scriptTask id="Task_258phd" name="Generate Order">
    <bpmn:incoming>SequenceFlow_25y4kfv</bpmn:incoming>
    <bpmn:outgoing>SequenceFlow_1xxy57q</bpmn:outgoing>
    <bpmn:standardLoopCharacteristics testBefore="true" loopMaximum="8">
      <bpmn:loopCondition
xsi:type="bpmn:tFormalExpression"><![CDATA[${APPROVED == true ||
TOTAL_APPROVED_LIMIT <= LIMIT_AMOUNT}]]></bpmn:loopCondition>
       </bpmn:standardLoopCharacteristics>
   </bpmn:scriptTask>
   ...
</bpmn:process>
```

ภาพที่ 3-19 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ALS

20) ตัวดำเนินการมิวเทชัน AAA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /,%) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <adHocSubProcess> สำหรับแท็ก <completionCondition> เป็นการกำหนดเงื่อนไขสิ้นสุดของ AdHoc-Sub Process เมื่อเงื่อนไขนั้นเป็นจริง โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ AdHoc-Sub Process

จากภาพที่ 3-20 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดยที่ AdHoc-Sub Process ได้ถูกกำหนดเงื่อนไขไว้ ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000 && IS_PAID == true} แต่แบบจำลองมิวแตนท์เปลี่ยนจากเครื่องหมายบวก (+) แทนที่ด้วยเครื่องหมายลบ (-)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess id="adhocSubProcess" ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER + TOTAL_SHIPMENT > 50000
           && IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess     id="adhocSubProcess"     ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER - TOTAL_SHIPMENT > 50000
           || IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

ภาพที่ 3-20 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท AAA

21) ตัวดำเนินการมิวเทชัน ARA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงสัมพันธ์ (<,<=, >,>=, ==, !=) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <adHocSubProcess> โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ AdHoc-Sub Process

จากภาพที่ 3-21 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดยที่ AdHoc-Sub Process ถูกกำหนดเงื่อนไข ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000 && IS_PAID == true} แต่แบบจำลองมิวแตนท์เปลี่ยนเครื่องหมายมากกว่า (>) แทนที่ด้วยเครื่องหมายมากกว่า หรือเท่ากับ(>=)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess id="adhocSubProcess" ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000
            && IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
 </bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess id="adhocSubProcess" ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER + TOTAL_SHIPMENT >= 50,000
            && IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
 </bpmn:process>
```

ภาพที่ 3-21 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ARA

22) ตัวดำเนินการมิวเทชัน ALA เป็นตัวดำเนินการมิวเทชันที่เปลี่ยนตัวดำเนินการเชิงตรรกศาสตร์ (and, or) ของแท็ก <completionCondition> ซึ่งอยู่ภายใต้แท็ก <adHocSubProcess> โดยตัวดำเนินการสามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ AdHoc-Sub Process

จากภาพที่ 3-22 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดยที่ AdHoc-Sub Process ได้ถูกกำหนดเงื่อนไขไว้ ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000 && IS_PAID == true} แต่แบบจำลองมิวแตนท์เปลี่ยนเครื่องหมายและ (&&) แทนที่ด้วยเครื่องหมายหรือ (||)

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess id="adhocSubProcess" ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000
            && IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess id="adhocSubProcess" ordering="Sequential"
behavior="None">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition><![CDATA[
              ${TOTAL_ORDER + TOTAL_SHIPMENT > 50,000}
            || IS_PAID == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

ภาพที่ 3-22 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ALA

23) ตัวดำเนินการมิวเทชัน AOR เป็นตัวดำเนินการมิวเทชันที่ปรับค่าแอตทริบิวต์ “ordering” ใน <adHocSubProcess> ระหว่าง “Parallel” และ“Sequential” สำหรับแอตทริบิวต์ ordering เป็นการกำหนดการทำงานของกิจกรรมใน AdHoc-Sub Process ให้กิจกรรมดำเนินตามลำดับ (Sequential) หรือดำเนินการแบบขนาน (Parallel) โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ AdHoc-Sub Process

จากภาพที่ 3-23 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดยที่ AdHoc-Sub Process ค่าแอตทริบิวต์ “ordering” เท่ากับ Parallel แต่ในแบบจำลองมิวแตนท์ถูกแทนที่เป็น “Sequential”

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess      id="adhocSubProcess"      ordering="Parallel"
cancelRemainingInstances="false">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition>
              <![CDATA[${SUBMIT_DECISION == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  ...
  <bpmn:adHocSubProcess     id="adhocSubProcess"     ordering="Sequential"
cancelRemainingInstances="false">
    <bpmn:userTask id="subProcessTask" name="Task in subprocess" />
    <bpmn:userTask id="subProcessTask2" name="Task2 in subprocess" />
    <bpmn:completionCondition>
              <![CDATA[${SUBMIT_DECISION == true}]]>
       </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
   ...
</bpmn:process>
```

ภาพที่ 3-23 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท AOR

24) ตัวดำเนินการมิวเทชัน ARR เป็นตัวดำเนินการมิวเทชันที่ปรับค่าแอตทริบิวต์ “cancelRemainingInstances” ใน <adHocSubProcess> ระหว่าง “true” และ “false” สำหรับแอตทริบิวต์ cancelRemainingInstances เป็นแอตทริบิวต์ทำหน้าที่ยกเลิกการทำงานของกิจกรรมที่เกิดขึ้น กรณีที่ค่าของ <completionCondition> เป็นจริง ซึ่งถ้ากำหนดค่าเป็น “true” เมื่อเงื่อนไขใน <completionCondition> เป็นจริง กิจกรรมที่กำลังทำงานอยู่จะถูกยกเลิกไปด้วย แต่ถ้ากำหนดค่าเป็น “false” เมื่อเงื่อนไขใน <completionCondition> เป็นจริง กิจกรรมยังคงทำงานอยู่จนกว่าจะเสร็จสิ้น โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ AdHoc-Sub Process

จากภาพที่ 3-24 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดยที่ AdHoc-Sub Process ค่าแอตทริบิวต์ “cancelRemainingInstances” เท่ากับ true แต่ในแบบจำลองมิวแตนท์ถูกแทนที่เป็น false

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:adHocSubProcess id="a1" cancelRemainingInstances="true">
    <bpmn:userTask id="subProcessT1" name="Task in subprocess" />
    <bpmn:userTask id="subProcessT2" name="Task2 in subprocess" />
    <bpmn:completionCondition>
              <![CDATA[${SUBMIT_DECISION == true}]]>
    </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
</bpmn:process>
```

แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:adHocSubProcess id="a1" cancelRemainingInstances="false">
    <bpmn:userTask id="subProcessT1" name="Task in subprocess" />
    <bpmn:userTask id="subProcessT2" name="Task2 in subprocess" />
    <bpmn:completionCondition>
              <![CDATA[${SUBMIT_DECISION == true}]]>
    </bpmn:completionCondition>
  </bpmn:adHocSubProcess>
</bpmn:process>
```

ภาพที่ 3-24 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท ARR

25) ตัวดำเนินการมิวเทชัน XBR เป็นตัวดำเนินการมิวเทชันที่ปรับค่าแอตทริบิวต์ “behavior” ของแท็ก <multiInstanceLoopCharacteristics> ด้วยค่า “None”, “One”, “All” และ “Complex” โดยตัวดำเนินการนี้สามารถประยุกต์ใช้กับสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็น ได้แก่ Service Task, User Task, Script Task และ Sub-Process

จากภาพที่ 3-25 ได้แสดงตัวอย่างการสร้างมิวแตนท์ของแบบจำลอง โดย script task “BULK_ORDER” ได้กำหนดค่าแอตทริบิวต์ “behavior” มีค่าเท่ากับ None แต่ในแบบจำลองมิวแตนท์ถูกแทนที่เป็น One

แบบจำลองต้นฉบับ

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:scriptTask id="Task_191p4jo" name="BULK_ORDER">
    <bpmn:multiInstanceLoopCharacteristics    behavior="none">
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:scriptTask>
</bpmn:process>
```

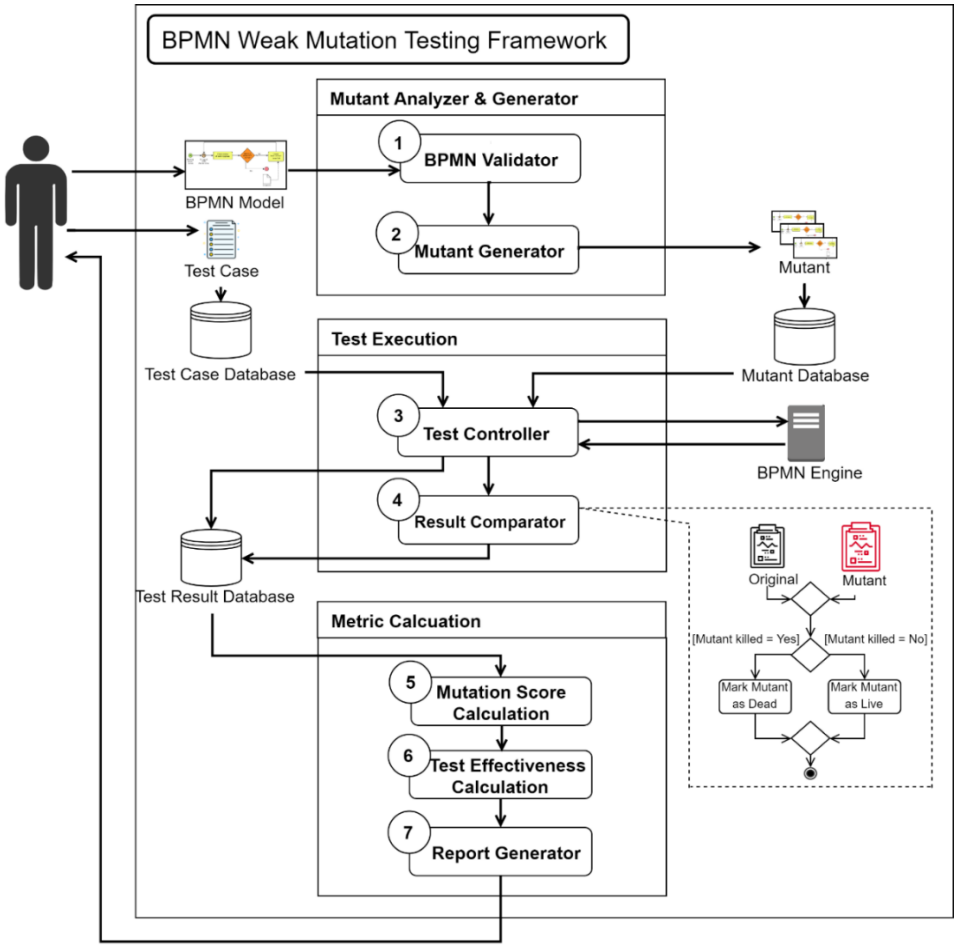
แบบจำลองมิวแตนท์

```
<bpmn:process id="Process_1" isExecutable="true">
  <bpmn:scriptTask id="Task_191p4jo" name="BULK_ORDER">
    <bpmn:multiInstanceLoopCharacteristics    behavior="one" >
    </bpmn:multiInstanceLoopCharacteristics>
   </bpmn:scriptTask>
</bpmn:process>
```

ภาพที่ 3-25 การสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันประเภท XBR

3.3 ภาพรวมของเครื่องมือ

งานวิจัยนี้นำเสนอเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยวิธีวีคมิวเทชัน โดยนำตัวดำเนินการมิวเทชันสำหรับแบบจำลองบีพีเอ็มเอ็นที่ถูกเสนอในงานวิจัย [5] มาประยุกต์ใช้งานกับการทดสอบแบบวีคมิวเทชัน [4] โดยแผนภาพเชิงแนวคิดของเครื่องมือได้แสดงดังภาพที่ 3-26



ภาพที่ 3-26 แผนภาพเชิงแนวคิดของเครื่องมือ

จากภาพที่ 3-26 แผนภาพเชิงแนวคิดของเครื่องมือสามารถแบ่งได้เป็น 3 ส่วน โดยมีรายละเอียดดังนี้

1) ส่วนวิเคราะห์ และสร้างมิวแตนท์ (Mutant Analyzer & Generator) สำหรับในส่วนนี้เป็นที่รับแบบจำลองบีพีเอ็มเอ็น วิเคราะห์ และสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็นโดยมีรายละเอียดดังนี้

   1.1) ส่วนการตรวจสอบแบบจำลองบีพีเอ็มเอ็น (BPMN Validator) (หมายเลข 1) ในส่วนนี้ทำหน้าที่นำแบบจำลองบีพีเอ็มเอ็นซึ่งถูกจัดเก็บในภาษาเอกซ์เอ็มแอล โดยนำไปตรวจสอบตามกฎของภาษาเอกซ์เอ็มแอล และเอกซ์เอสดีของแบบจำลองบีพีเอ็มเอ็น ถ้าผลลัพธ์แบบจำลองบีพีเอ็นเอ็นที่นำเข้าถูกต้องตามข้อกำหนดของภาษาเอกซ์เอ็มแอล และเอกซ์เอสดีของแบบจำลองบีพีเอ็มเอ็น ก็จะผ่านไปสู่ขั้นตอนถัดไปของการทดสอบ

   1.2) ส่วนการสร้างมิวแตนท์ (Mutant Generator) (หมายเลข 2) ในส่วนนี้ทำหน้าที่สำหรับการสร้างมิวแตนท์ จากตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็มเอ็น โดยใส่ข้อผิดพลาดเพียงหนึ่งที่ต่อหนึ่งแบบจำลองอาทิ เช่น ตัวดำเนินการ EAR ซึ่งเปลี่ยนตัวดำเนินการทางคณิตศาสตร์ (+, -, *, /, %) หากแบบจำลองต้นฉบับมีการใช้เครื่องหมายบวก มิวแตนท์ที่เกิดจากการสร้างของเครื่องมือเปลี่ยนจากเครื่องหมายบวก เป็นเครื่องหมายลบ คูณ หาร และมอดุลาร์ ตามลำดับ โดยจะได้แบบจำลองมิวแตนท์ทั้งสิ้น 4 แบบจำลอง เมื่อเครื่องมือสร้างมิวแตนท์ตามตัวดำเนินการมิวเทชันที่สามารถนำมาประยุกต์ได้ หลังจากนั้นเครื่องมือนำมิวแตนท์ที่สร้างได้จัดเก็บลงในฐานข้อมูลมิวแตนท์ (Mutant Database)

2) ส่วนกระทำการทดสอบ (Test Execution) สำหรับในส่วนนี้เป็นการนำมิวแตนท์ที่สร้างได้จากแบบจำลองบีพีเอ็มเอ็น นำมาทดสอบกับกรณีทดสอบที่ได้จัดเตรียมไว้โดยมีรายละเอียดดังนี้

   2.1) ส่วนการควบคุมการทดสอบ (Test Controller) (หมายเลข 3) ในส่วนนี้ทำหน้าที่สำหรับควบคุม และการจัดการทดสอบจากแบบจำลองบีพีเอ็มเอ็น ซึ่งประกอบไปด้วยมิวแตนท์จากฐานข้อมูลมิวแตนท์ และชุดทดสอบที่จัดเตรียมไว้จากฐานข้อมูลชุดทดสอบ (Test Case Database) ส่งไปที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นเพื่อติดตั้งแบบจำลอง และดำเนินการ (execute) ที่ตามตัวดำเนินการมิวเทชันที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นรองรับ จากนั้นนำผลลัพธ์ส่งต่อให้ส่วนการเปรียบเทียบผลลัพธ์ต่อไป

2.2) ส่วนการเปรียบเทียบผลลัพธ์ (Result Comparator) (หมายเลข 4) ในส่วนนี้ทำหน้าที่นำผลลัพธ์ที่ได้จากส่วนตัวควบคุมการทดสอบ มาเปรียบเทียบผลลัพธ์ตามการทดสอบวีคมิวเทชันแบบระดับประโยค (Statement-WEAK/1) ระดับกลุ่มประโยคหลังถูกดำเนินการไปแล้ว 1 ครั้ง (Basic-Block-WEAK/1) และระดับกลุ่มประโยคหลังถูกดำเนินการไปแล้ว N ครั้ง (Basic-Block-WEAK/N) จากนั้นนำผลลัพธ์ที่ได้จัดเก็บลงในฐานข้อมูลผลลัพธ์ของการทดสอบ ซึ่งมีการเก็บผลลัพธ์ของการทดสอบ และเวลาที่ใช้

3) ส่วนการคำนวณมาตรวัด (Metric Calculation) สำหรับในส่วนนี้เป็นการนำผลการทดสอบที่ได้จากส่วนกระทำการทดสอบมาสร้างรายงานของการทดสอบโดยมีรายละเอียดดังนี้

3.1) ส่วนการคำนวณคะแนนมิวเทชัน (Mutation Score Calculation) (หมายเลข 5) ในส่วนนี้ทำหน้าที่หาจำนวนของมิวแตนท์ที่ถูกฆ่า จำนวนมิวแตนท์ที่มีชีวิต และจำนวนของมิวแตนท์ทั้งหมด จากนั้นนำมาคำนวณคะแนนมิวเทชันตามสมการที่ 1 ในบทที่ 2 หัวข้อ การทดสอบมิวเทชัน – คะแนนมิวเทชัน

3.2) ส่วนการคำนวณประสิทธิภาพของชุดทดสอบ (Test Effectiveness Calculation) (หมายเลข 6) ในส่วนนี้ทำหน้าที่คำนวณประสิทธิภาพของการทดสอบจากสมการที่ 2 ในบทที่ 2 หัวข้อ การทดสอบมิวเทชัน – ประสิทธิภาพของชุดทดสอบ

3.3) ส่วนการสร้างรายงานการทดสอบ (Report Generator) (หมายเลข 7) ในส่วนนี้ทำหน้าที่สร้างรายงานของการทดสอบ เพื่อแจ้งข้อมูลแก่นักทดสอบ โดยผลลัพธ์ของรายงานประกอบไปด้วยจำนวนของมิวแตนท์ที่ถูกฆ่า จำนวนมิวแตนท์ที่มีชีวิต จำนวนของมิวแตนท์ทั้งหมด คะแนนมิวเทชัน ประสิทธิภาพของการทดสอบ และเวลาที่ใช้สำหรับการทดสอบ

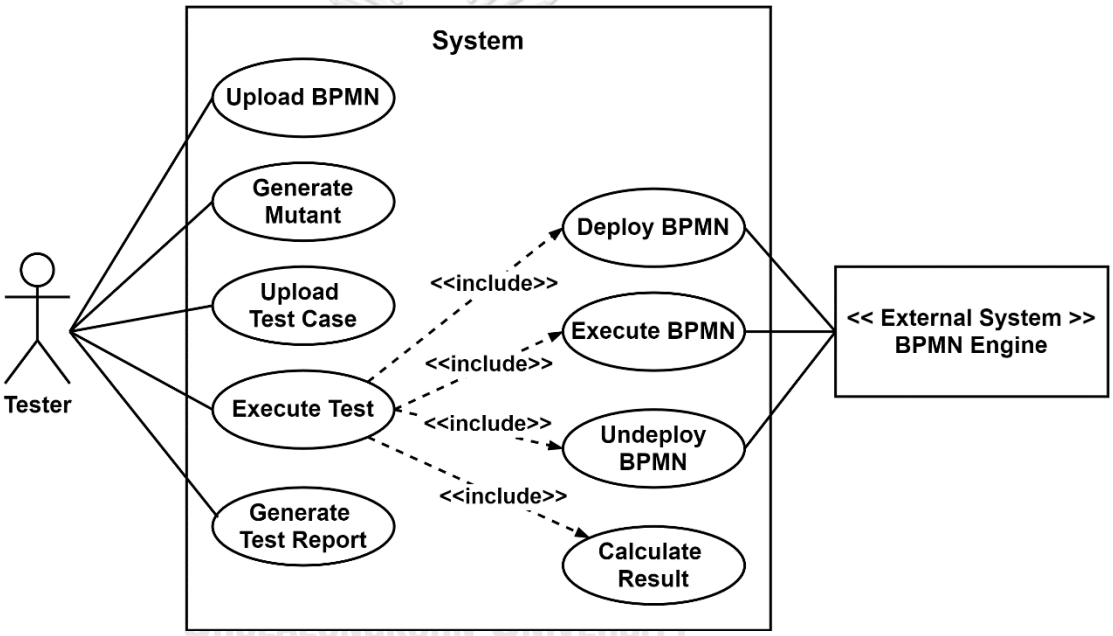
บทที่ 4
การออกแบบ และพัฒนาเครื่องมือ

บทนี้เป็นการนำเสนอการออกแบบ และพัฒนาเครื่องมือสร้างและทดสอบมิวแตนท์ด้วยการทดสอบวีคมิวเทชัน สภาพแวดล้อมที่ใช้ในการพัฒนา และโครงสร้างของส่วนต่อประสานผู้ใช้

4.1 การออกแบบเครื่องมือสร้างและทดสอบมิวแตนท์

4.1.1 แผนภาพยูสเคส (Use Case Diagram)

แผนภาพยูสเคสของเครื่องมือแสดงดังภาพที่ 4-1 โดยแสดงถึงฟังก์ชันการทำงานของเครื่องมือ



ภาพที่ 4-1 แผนภาพยูสเคสของเครื่องมือ

สำหรับยูสเคสแต่ละยูสเคสมีรายละเอียดการทำงาน ดังนี้

1) ยูสเคส Upload BPMN เป็นยูสเคสที่นักทดสอบนำแบบจำลองบีพีเอ็มเอ็นเข้าสู่เครื่องมือ
2) ยูสเคส Generate Mutant เป็นยูสเคสที่นักทดสอบสั่งให้เครื่องมือสร้างมิวแตนท์ตามการทดสอบวีคมิวเทชัน
3) ยูสเคส Upload Test Case เป็นยูสเคสที่นักทดสอบนำกรณีทดสอบเข้าสู่เครื่องมือ
4) ยูสเคส Execute Test เป็นยูสเคสที่นักทดสอบสั่งให้เครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นมิวแตนท์ กับกรณีทดสอบผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น

5) ยูสเคส Deploy BPMN เป็นยูสเคสที่เครื่องมือติดตั้งแบบจำลองบีพีเอ็มเอ็นที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น
6) ยูสเคส Execute BPMN เป็นยูสเคสที่เครื่องมือส่งคำสั่งให้เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นประมวลผลตามรายละเอียดที่ระบุไว้ในกรณีทดสอบ
7) ยูสเคส Undeploy BPMN เป็นยูสเคสที่เครื่องมือถอนการติดตั้งแบบจำลองที่เครื่องประมวลผลแบบจำลอง
8) ยูสเคส Calculate Result เป็นยูสเคสที่เครื่องมือเปรียบเทียบผลลัพธ์หลังจากประมวลผลผ่านเครื่องประมวลผลแบบจำลอง คำนวณคะแนนมิวเทชัน และคำนวณประสิทธิภาพของกรณีทดสอบ
9) ยูสเคส Generate Report เป็นยูสเคสที่นักทดสอบสั่งให้เครื่องมือสร้างรายงานสรุปผลการทดสอบ

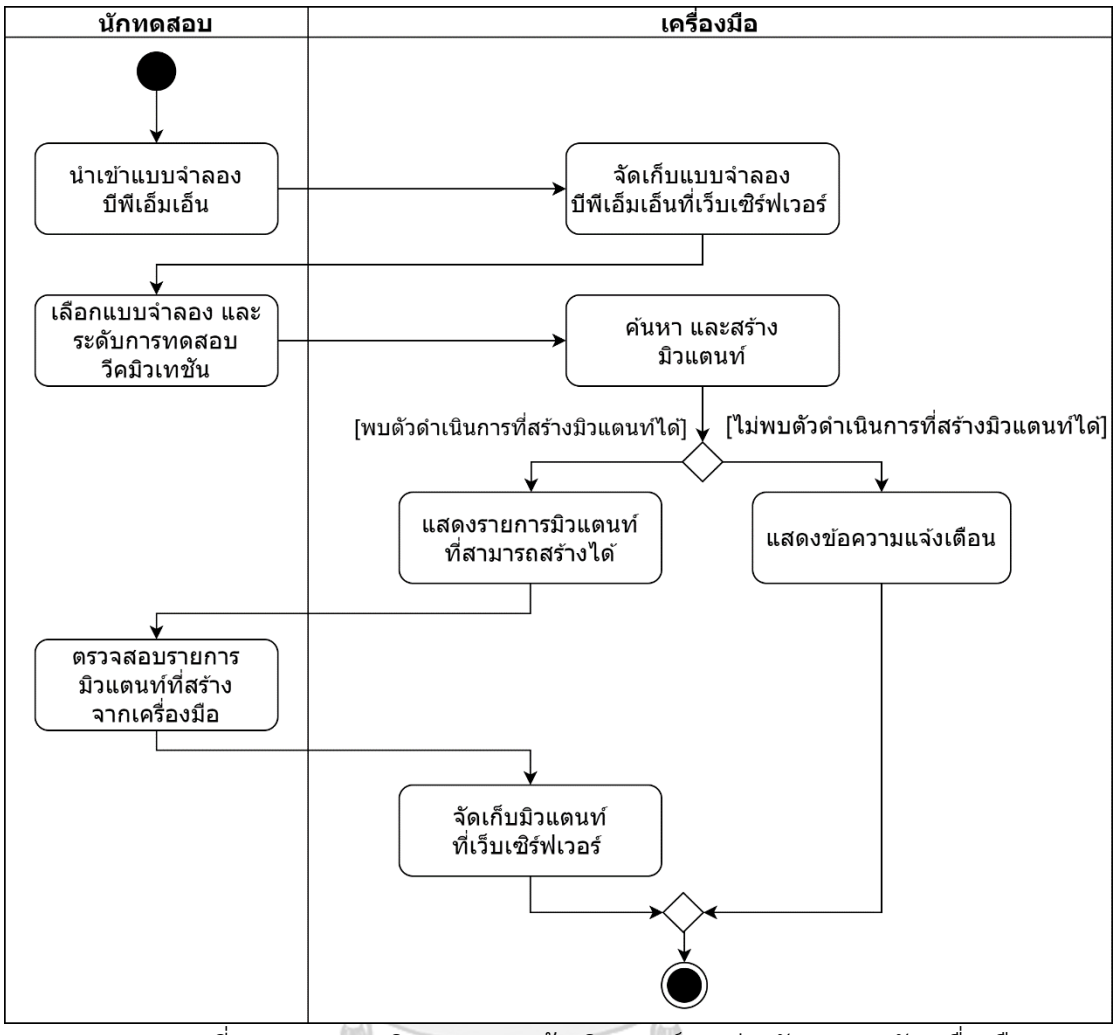
หมายเหตุ รายละเอียดของแต่ละยูสเคสสามารถดูเพิ่มเติมได้ที่ภาคผนวก ก

4.1.2 แผนภาพกิจกรรม (Activity Diagram)

แผนภาพกิจกรรมประกอบไปด้วย 6 แผนภาพกิจกรรม โดยแบ่งเป็น 2 แผนภาพกิจกรรมที่แสดงความสัมพันธ์ระหว่างนักทดสอบและเครื่องมือ และ 4 แผนภาพกิจกรรมที่แสดงถึงขั้นตอนการสร้างมิวแตนท์ของเครื่องมือตามตัวอย่างจากบทที่ 3 หัวข้อที่ 3.2 โดยมีรายละเอียด ดังนี้

1) แผนภาพกิจกรรมการสร้างมิวแตนท์ระหว่างนักทดสอบกับเครื่องมือ

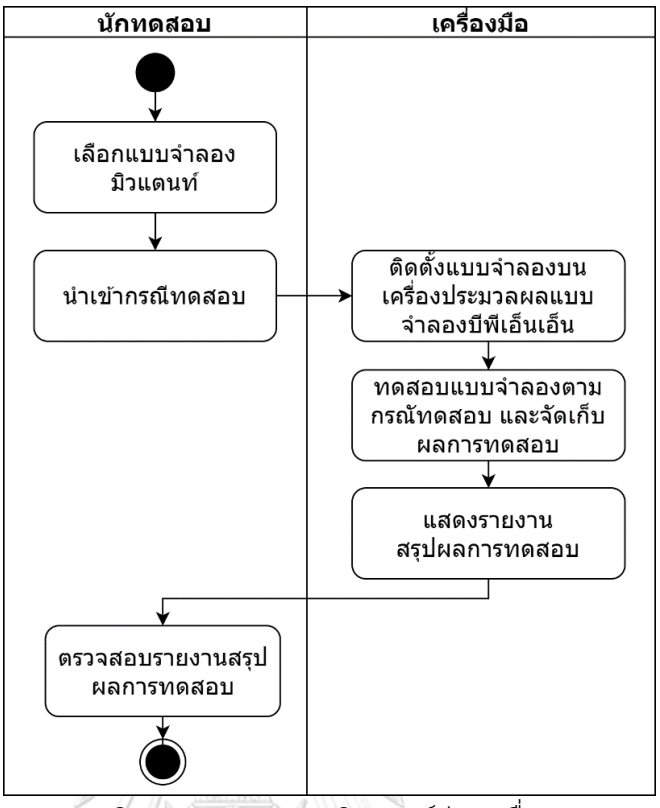
แผนภาพกิจกรรมการสร้างมิวแตนท์ระหว่างนักทดสอบกับเครื่องมือ ได้แสดงดังภาพที่ 4-2 โดยเริ่มจากที่นักทดสอบอัพโหลดไฟล์แบบจำลองบีพีเอ็มเอ็นเข้าสู่เครื่องมือ เมื่อเครื่องมือนำเข้าแบบจำลอง และจัดเก็บแบบจำลองลงที่เว็บเซิร์ฟเวอร์ (Web Server) นักทดสอบเลือกแบบจำลอง และระดับการทดสอบวีคมิวเทชัน ได้แก่ ST-WEAK/1, BB-WEAK/1 และ BB-WEAK/N เพื่อนำไปเป็นพารามิเตอร์สำหรับสร้างแบบจำลองมิวแตนท์ จากนั้นเครื่องมือค้นหา และสร้างแบบจำลองมิวแตนท์ โดยค้นหาตามตัวดำเนินการมิวเทชันทีละตัวดำเนินการจนครบ และแสดงรายการของมิวแตนท์ที่เครื่องมือสร้างได้ให้แก่นักทดสอบ ในกรณีที่เครื่องมือไม่สามารถสร้างมิวแตนท์ได้ เครื่องมือจะแสดงข้อความแจ้งเตือนให้แก่นักทดสอบ เมื่อนักทดสอบตรวจสอบรายการมิวแตนท์ที่สร้างจากเครื่องมือเสร็จสิ้น เครื่องมือจัดเก็บมิวแตนท์ที่เว็บเซิร์ฟเวอร์



ภาพที่ 4-2 แผนภาพกิจกรรมการสร้างมิวแตนท์ระหว่างนักทดสอบกับเครื่องมือ

2) แผนภาพกิจกรรมการทดสอบมิวแตนท์ผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น ระหว่างนักทดสอบกับเครื่องมือ

แผนภาพกิจกรรมการทดสอบมิวแตนท์ผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น ระหว่างนักทดสอบกับเครื่องมือได้แสดงดังภาพที่ 4-3 โดยเริ่มจากที่นักทดสอบเลือกมิวแตนท์ และอัพโหลดกรณีทดสอบเข้าสู่เครื่องมือ จากนั้นเครื่องมือติดตั้งแบบจำลองที่เครื่องประมวลผลแบบจำลอง และทดสอบแบบจำลองตามกรณีทดสอบจนเสร็จสิ้น เครื่องมือแสดงรายงานสรุปผลการทดสอบให้แก่นักทดสอบ เพื่อตรวจสอบผลลัพธ์ต่อไป



ภาพที่ 4-3 แผนภาพกิจกรรมการทดสอบมิวแตนท์ผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นระหว่างนักทดสอบกับเครื่องมือ

3) แผนภาพกิจกรรมการสร้างมิวแตนท์ของเครื่องมือ

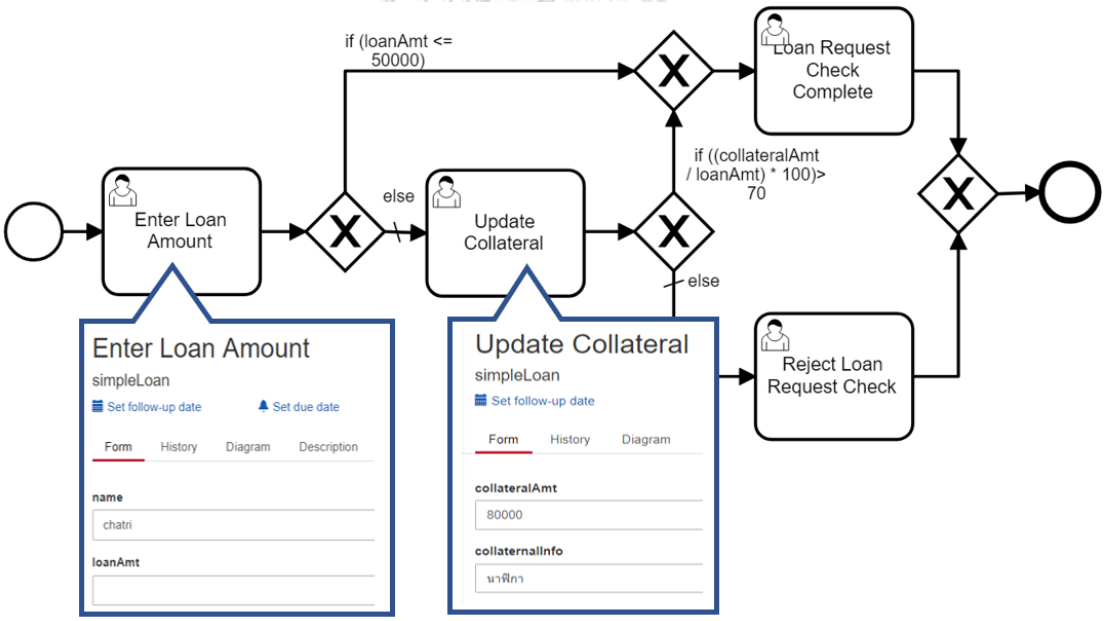
แผนภาพกิจกรรมการสร้างมิวแตนท์ของเครื่องมือเป็นแผนภาพที่แสดงการทำงานของเครื่องมือในส่วนของการสร้างมิวแตนท์ตามตัวอย่างที่ได้แสดงในบทที่ 3 หัวข้อที่ 3.2 เมื่อพิจารณาจากวิธีการที่ใช้แก้ไขไฟล์เอกซ์เอ็มเอลของแบบจำลองสามารถแบ่งวิธีการสร้างมิวแตนท์ของเครื่องมือได้ 4 วิธีการ ได้แก่

- การสร้างมิวแตนท์ด้วยการเปลี่ยนตัวแปร ได้แก่ IVR และ ITR
- การสร้างมิวแตนท์ด้วยการแก้ไขนิพจน์ ได้แก่ EAR, ELR, ERR, AAA, AAM, AAS, ALA, ALM, ALS, ARA และ ARM
- การสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแท็ก ได้แก่ ECA, EDA, ERA, ETA, ACR และ AMR
- การสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแอตทริบิวต์ ได้แก่ AOR, ARR, ASR, ATR และ XBR

สำหรับรายละเอียดการสร้างมิวแตนท์ของเครื่องมือได้มีการอธิบายเพิ่มเติมในหัวข้อที่ 3.1-3.4

3.1) การสร้างมิวแตนท์ด้วยการเปลี่ยนตัวแปร

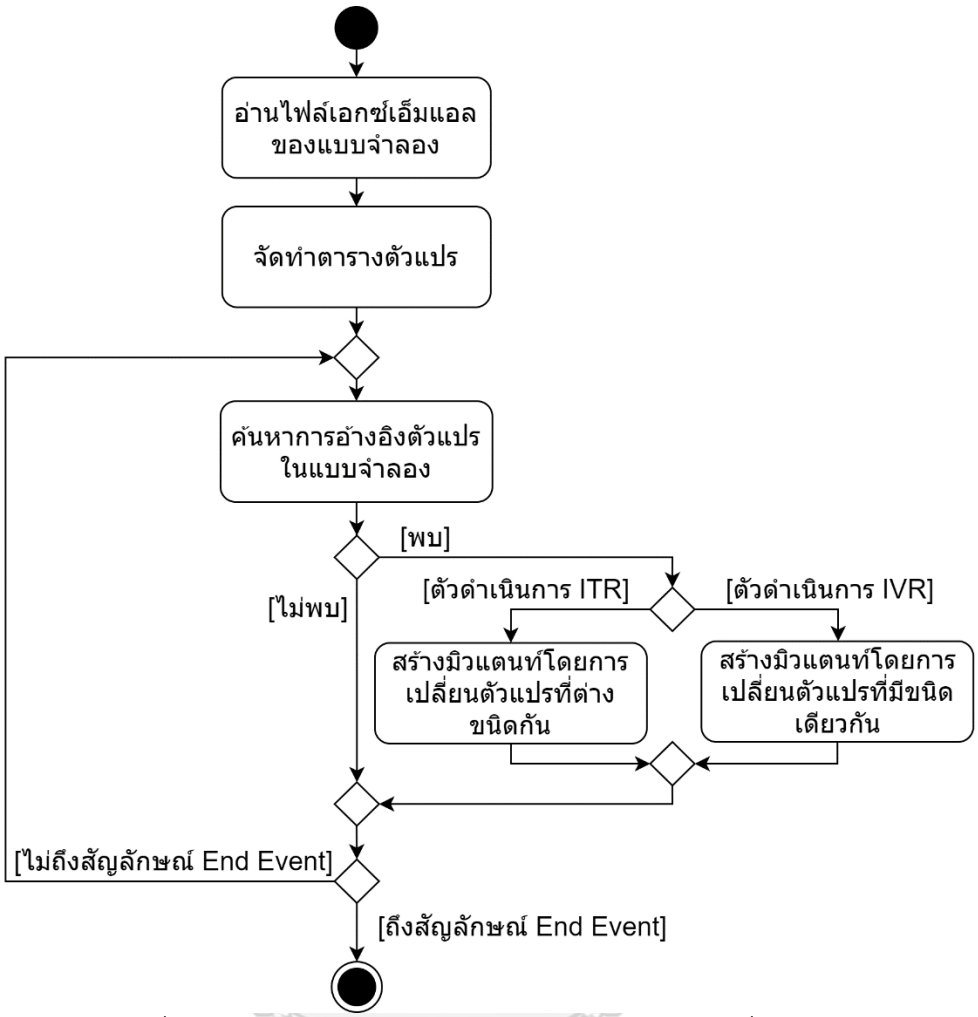
แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนตัวแปรได้แสดงดังภาพที่ 4-5 เริ่มต้นจากเครื่องมืออ่านไฟล์เอกซ์เอ็มแอลของแบบจำลองบีพีเอ็มเอ็น จากนั้นค้นหาชื่อและชนิดของตัวแปรจากแท็ก <bpmn:extensionElements> นำผลลัพธ์ที่ได้เก็บไว้ตารางตัวแปร จากภาพที่ 4-4 ได้แสดงแบบจำลองที่มีการประกาศตัวแปรในกิจกรรม Enter Loan Amount และ Update Collateral และสามารถจัดทำตารางตัวแปรได้ดังตารางที่ 4-1 เมื่อได้ตารางตัวแปร เครื่องมือค้นหาการอ้างอิงตัวแปรในแต่ละส่วนของแบบจำลอง ซึ่งถ้าพบการอ้างอิงตัวแปร เครื่องมือเปลี่ยนตัวแปรตามข้อกำหนดของตัวดำเนินการมิวเทชัน ตัวแปรที่เป็นไปได้ในการเปลี่ยนจะดูจากตารางตัวแปร ซึ่งตัวดำเนินการมิวเทชัน IVR เป็นการเปลี่ยนตัวแปรอื่นที่มีชนิดเดียวกัน และตัวดำเนินการมิวเทชัน ITR เป็นการเปลี่ยนตัวแปรอื่นที่มีชนิดต่างกัน โดยเครื่องมือวนซ้ำสร้างมิวแตนท์จนกว่าจะถึงสัญลักษณ์ End Event ของแบบจำลอง



ภาพที่ 4-4 แบบจำลองที่มีการประกาศตัวแปร

ตารางที่ 4-1 ตารางตัวแปรที่ได้จากภาพที่ 4-4

กิจกรรม	ตัวแปร	ชนิดข้อมูล
Enter Loan Amount	name	String
Enter Loan Amount	loanAmt	long
Update Collateral	collateralAmt	long
Update Collateral	CollateralInfo	String



ภาพที่ 4-5 แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนตัวแปร

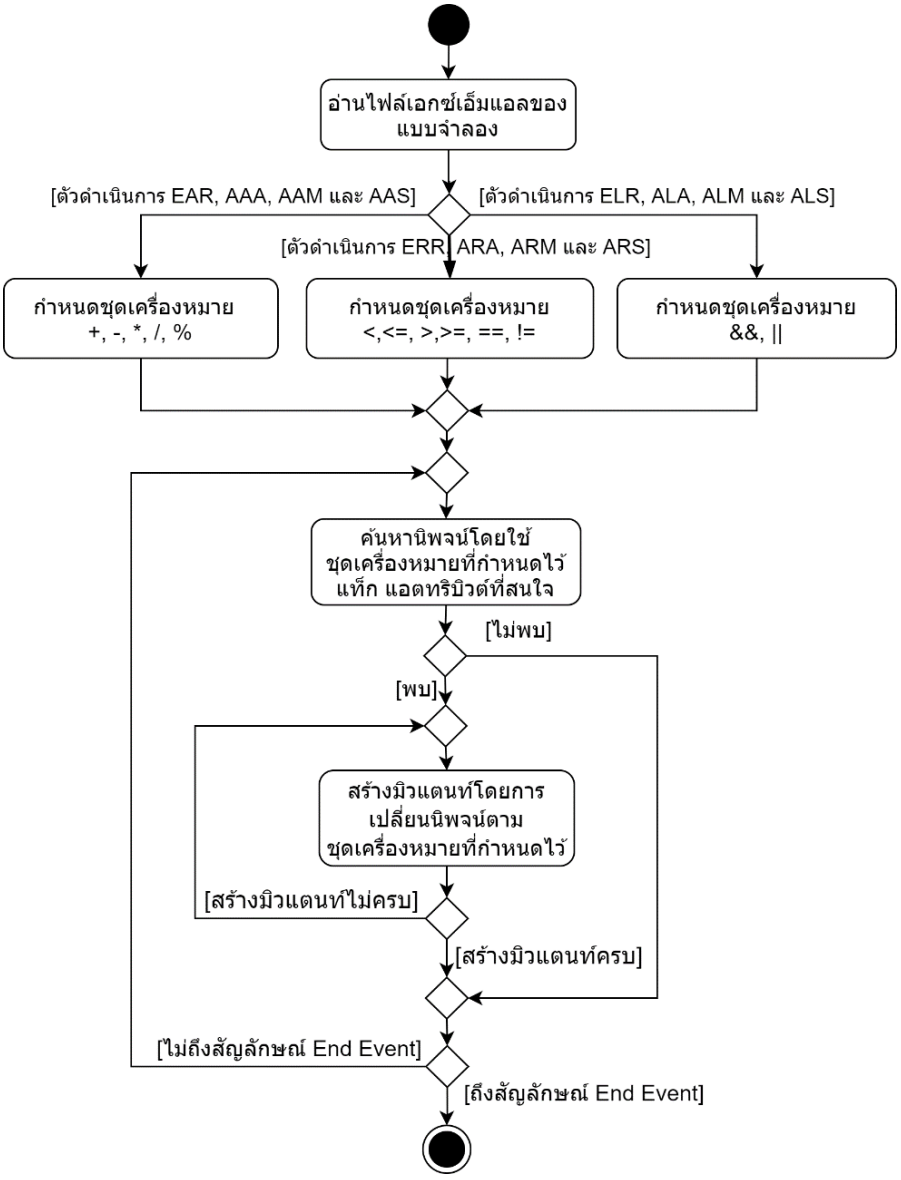
3.2) การสร้างมิวแตนท์ด้วยการแก้ไขนิพจน์

แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการแก้ไขนิพจน์แสดงดังภาพที่ 4-6 โดยเริ่มต้นจากเครื่องมืออ่านไฟล์เอกซ์เอ็มแอลของแบบจำลองบีพีเอ็มเอ็น จากนั้นเครื่องมือกำหนดเครื่องหมายสำหรับแก้ไขนิพจน์ตามตัวดำเนินการมิวเทชันดังตารางที่ 4-2

ตารางที่ 4-2 รายการเครื่องหมายสำหรับแก้ไขนิพจน์

ตัวดำเนินการ	ชุดเครื่องหมาย
EAR, AAA, AAM และ AAS	+, -, *, /, %
ERR, ARA, ARM และ ARS	<,<=, >,>=, ==, !=
ELR, ALA, ALM และ ALS	&&, ||

เครื่องมือค้นหานิพจน์โดยใช้ชุดเครื่องหมาย ร่วมกับแท็ก หรือแอตทริบิวต์ที่เป็นจุดสังเกตของแต่ละตัวดำเนินการมิวเทชัน เมื่อพบนิพจน์ที่เข้าข่าย เครื่องมือสร้างมิวแตนท์โดยการแก้ไขนิพจน์ตามเครื่องหมายที่เป็นไปได้จนครบถ้วน และทำวนซ้ำจนกว่าจะถึงสัญลักษณ์ End Event ของแบบจำลอง

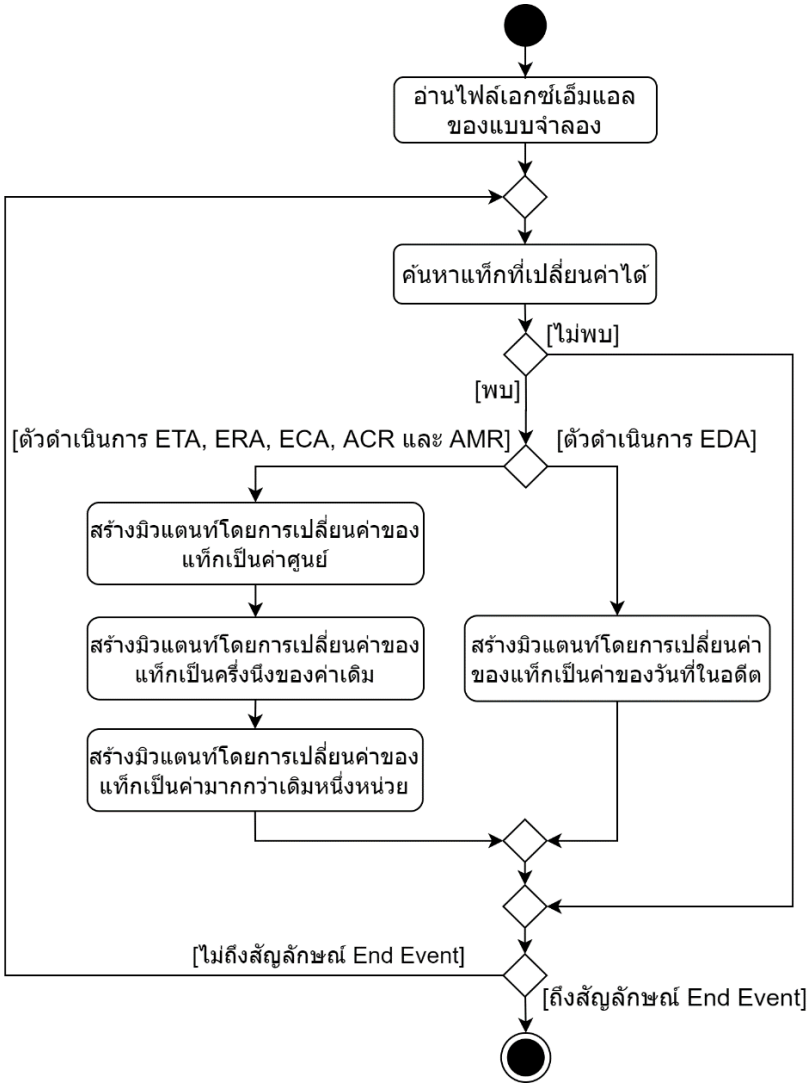


ภาพที่ 4-6 แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการแก้ไขนิพจน์

3.3) การสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแท็ก

แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแท็กแสดงดังภาพที่ 4-7 เริ่มต้นจากเครื่องมืออ่านไฟล์เอกซ์เอ็มแอลของแบบจำลองบีพีเอ็มเอ็น จากนั้นเครื่องมือค้นหาแท็กของแบบจำลองของแต่ละตัวดำเนินการมิวเทชัน ถ้าพบแท็กที่ตรงตามเงื่อนไข

กรณีที่เป็นตัวดำเนินการมิวเทชัน ETA, ERA, ECA, ACR และ AMR เครื่องมือสร้างมิวแตนท์โดยการเปลี่ยนค่าในแท็กเป็นศูนย์, ค่าครึ่งหนึ่งของค่าเดิม และค่ามากกว่าค่าเดิมหนึ่งหน่วย แต่ถ้าเป็นตัวดำเนินการมิวเทชัน EDA เครื่องมือสร้างมิวแตนท์โดยการเปลี่ยนค่าของวันที่เป็นวันในอดีต โดยเครื่องมือวนซ้ำสร้างมิวแตนท์จนกว่าจะถึงสัญลักษณ์ End Event ของแบบจำลอง



ภาพที่ 4-7 แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแท็ก

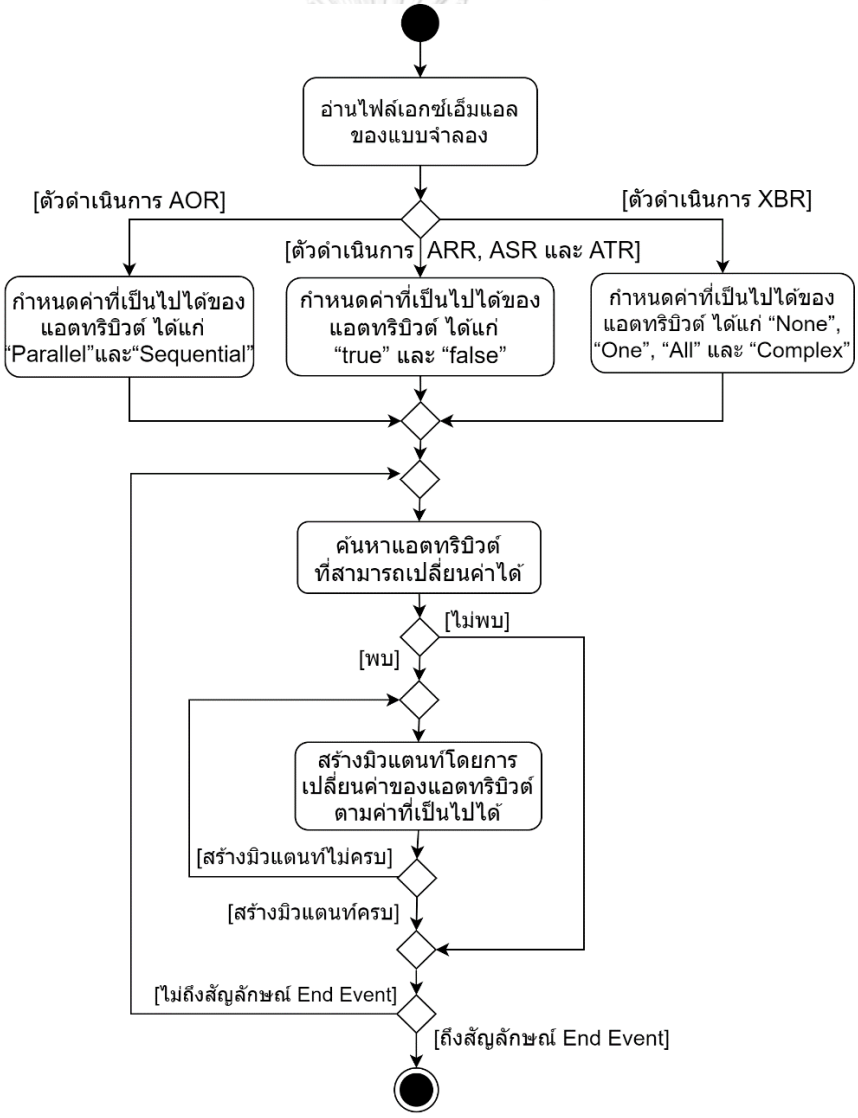
3.4) การสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแอตทริบิวต์

แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแอตทริบิวต์แสดงดังภาพที่ 4-8 เริ่มต้นจากเครื่องมืออ่านไฟล์เอกซ์เอ็มแอลของแบบจำลองบีพีเอ็มเอ็น และกำหนดค่าที่เป็นไปได้ของแอตทริบิวต์ตามตัวดำเนินการมิวเทชันดังตารางที่ 4-3 จากนั้นเครื่องมือค้นหาแอตทริบิวต์ที่เป็นจุดสังเกตของแต่ละตัวดำเนินการมิวเทชัน เมื่อพบ

แอตทริบิวต์ที่ตรงตามเงื่อนไข เครื่องมือสร้างมิวแตนท์ โดยการแก้ไขค่าของแอตทริบิวต์ตามค่าที่เป็นไปได้จนครบถ้วน และทำวนซ้ำจนกว่าจะถึงสัญลักษณ์ End Event ของแบบจำลอง

ตารางที่ 4-3 ค่าที่เป็นไปได้ของแอตทริบิวต์ตามตัวดำเนินการมิวเทชัน

ตัวดำเนินการ	ค่าที่เป็นไปได้ของแอตทริบิวต์
AOR	Parallel และ Sequential
ARR, ASR และ ATR	true และ false
XBR	None, One, All และ Complex



ภาพที่ 4-8 แผนภาพกิจกรรมการสร้างมิวแตนท์ด้วยการเปลี่ยนค่าของแอตทริบิวต์

4.1.3 แผนภาพคลาส (Class Diagram)

แผนภาพคลาสของเครื่องมือ โดยแสดงรายละเอียดต่าง ๆ ของคลาส และความสัมพันธ์ของคลาสในเครื่องมือ ดังภาพที่ 4-9 ซึ่งสำหรับแพ็คเกจแต่ละแพ็คเกจมีรายละเอียดการทำงานดังนี้

1) แพ็คเกจ controller มีหน้าที่จัดเก็บคลาสที่รับคำสั่งจากนักทดสอบ
2) แพ็คเกจ manageTest มีหน้าที่จัดเก็บคลาสที่จัดการแบบจำลอง และกรณีทดสอบ
3) แพ็คเกจ mutantGenerator มีหน้าที่จัดเก็บคลาสที่เกี่ยวกับการจัดการมิวแตนท์ สำหรับคลาสที่เกี่ยวข้องกับการสร้างมิวแตนท์ถูกแยกในแพ็คเกจย่อยชื่อ operator ซึ่งแสดงถึงคลาสที่ทำหน้าที่สร้างมิวแตนท์แยกตามตัวดำเนินการมิวเทชัน
4) แพ็คเกจ testExecution มีหน้าที่จัดเก็บคลาสที่ใช้สำหรับทดสอบแบบจำลองกับกรณีทดสอบ และคลาสที่ทำหน้าที่เชื่อมต่อกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น
5) แพ็คเกจ repositories มีหน้าที่จัดเก็บคลาสที่ใช้ควบคุม และจัดการคลาสเอ็นทิตี (Entity)
6) แพ็คเกจ entities มีหน้าที่จัดเก็บคลาสเอ็นทิตีที่ใช้แสดงแทนข้อมูลแบบจำลอง มิวแตนท์ และผลลัพธ์ของการทดสอบ

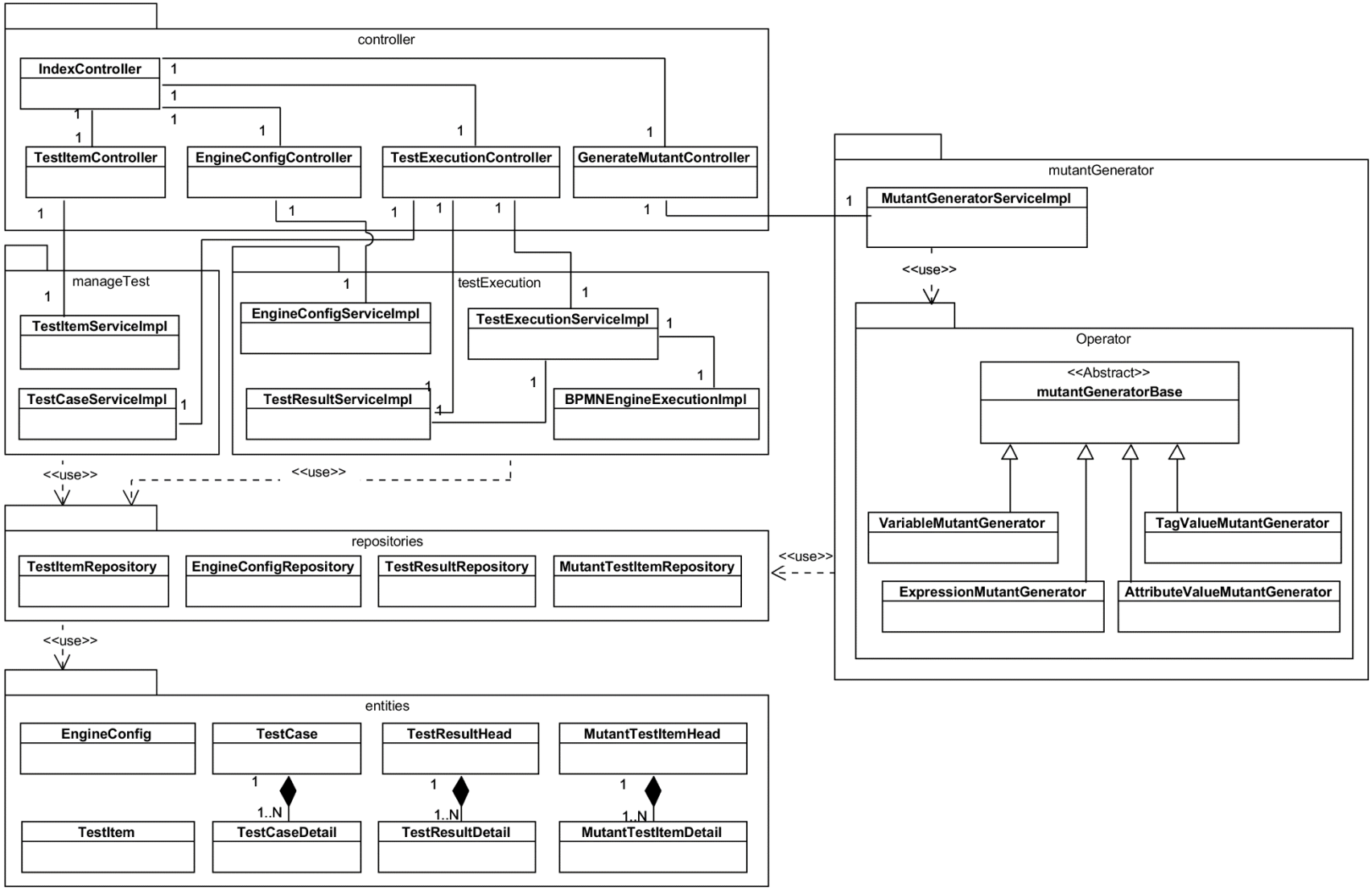
<u>หมายเหตุ</u> รายละเอียดของแต่ละคลาสสามารถดูเพิ่มเติมได้ที่ภาคผนวก ข

4.1.4 แผนภาพลำดับ (Sequence Diagram)

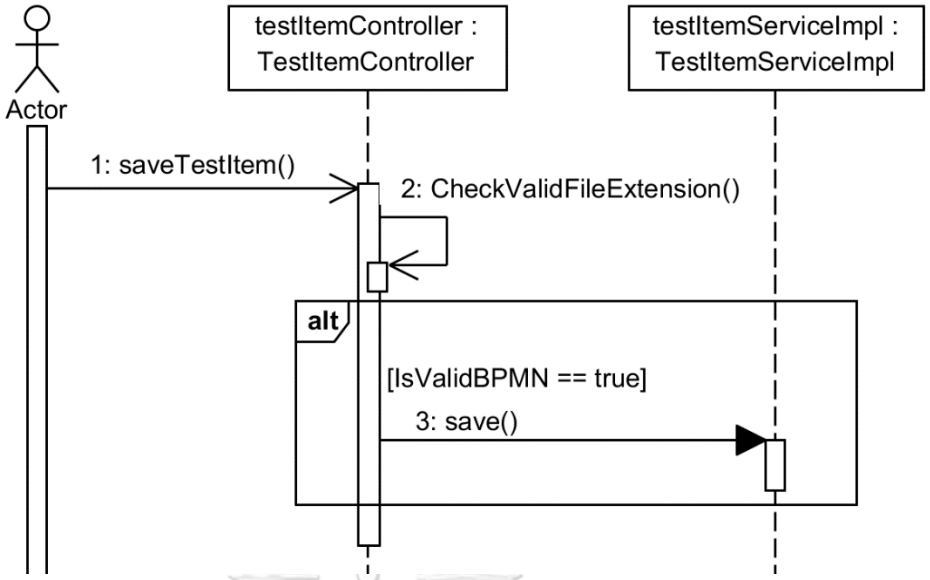
แผนภาพลำดับจะอธิบายถึงความสัมพันธ์ระหว่างวัตถุต่างๆที่ถูกกล่าวในแผนภาพคลาสเป็นลำดับขั้น โดยแผนภาพลำดับของเครื่องมือมีดังต่อไปนี้

1) แผนภาพลำดับ Upload BPMN File

แผนภาพลำดับ Upload BPMN File เป็นแผนภาพแสดงถึงการอัพโหลดแบบจำลองบีพีเอ็มเอ็นในรูปแบบไฟล์ (.bpmn) โดยนักทดสอบเรียกใช้ฟังก์ชัน saveTestItem ของคลาส testItemController ถัดมาเรียกใช้ฟังก์ชัน CheckValidFileExtension เพื่อตรวจสอบว่าเป็นแบบจำลองบีพีเอ็มเอ็น หรือไม่ ถ้าใช่คลาส TestItemServiceImpl เรียกใช้ฟังก์ชัน save เพื่อบันทึกข้อมูลแบบจำลองดังภาพที่ 4-10



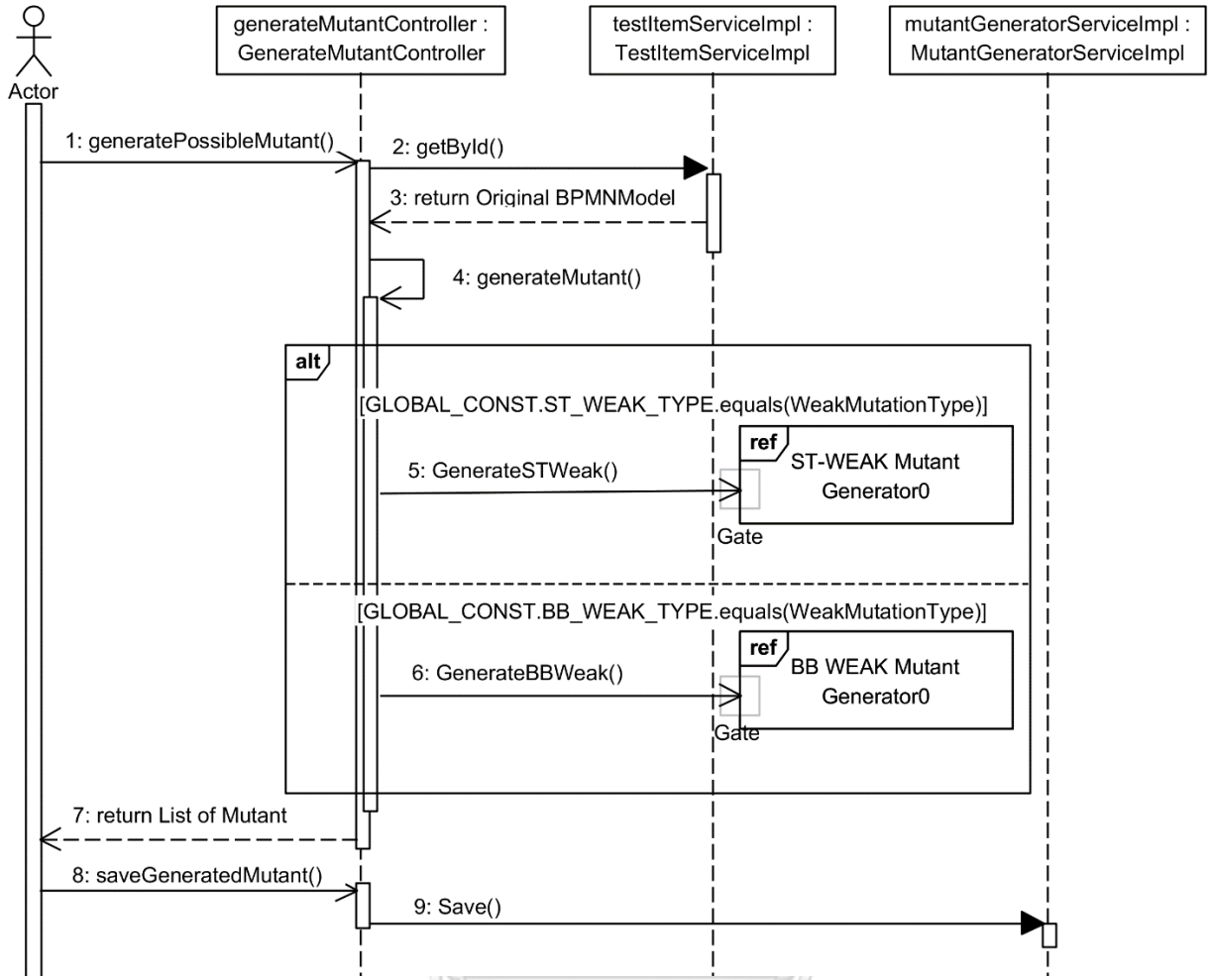
ภาพที่ 4-9 แผนภาพคลาสของเครื่องมือ



ภาพที่ 4-10 แผนภาพแผนภาพลำดับ Upload BPMN File

2) แผนภาพลำดับ Generate Mutant

แผนภาพลำดับ Generate Mutant เป็นแผนภาพที่แสดงถึงการสร้างมิวแตนท์จากแบบจำลองบีพีเอ็มเอ็นต้นฉบับดังภาพที่ 4-11 โดยนักทดสอบเลือกแบบจำลอง กับระดับการทดสอบวีคมิวเทชัน ได้แก่ ST-WEAK/1, BB-WEAK/1 และ BB-WEAK/N เมื่อนักทดสอบกดปุ่ม เพื่อสร้างมิวแตนท์เป็นการเรียกใช้ฟังก์ชัน generatePossibleMutant ของคลาส GenerateMutantController จากนั้นเครื่องมือดึงข้อมูลแบบจำลองบีพีเอ็นเอ็น โดยใช้ฟังก์ชัน getById ของคลาส TestItemServiceImpl และเรียกใช้ฟังก์ชัน generateMutant ของคลาส GenerateMutantController เพื่อสร้างมิวแตนท์ตามเงื่อนไขของระดับการทดสอบวีคมิวเทชัน ถ้ามีค่า ST-WEAK/1 เครื่องมือจะเรียกใช้ฟังก์ชัน GenerateSTWeak ของคลาส mutantGeneratorServiceImpl ซึ่งมีการอธิบายเพิ่มเติมที่หัวข้อย่อย 2.1 แต่ถ้ามีค่าเป็น BB-WEAK/1 หรือ BB-WEAK/N เครื่องมือจะเรียกใช้ฟังก์ชัน GenerateBBWeak ของคลาส mutantGeneratorServiceImpl ซึ่งมีการอธิบายเพิ่มเติมที่หัวข้อย่อย 2.2 และเมื่อนักทดสอบต้องการบันทึกมิวแตนท์จะเรียกใช้ฟังก์ชัน saveGeneratedMutant ของคลาส GenerateMutantController ซึ่งมีการส่งต่องานให้ฟังก์ชัน save ของคลาส MutantGeneratorServiceImpl ต่อไป



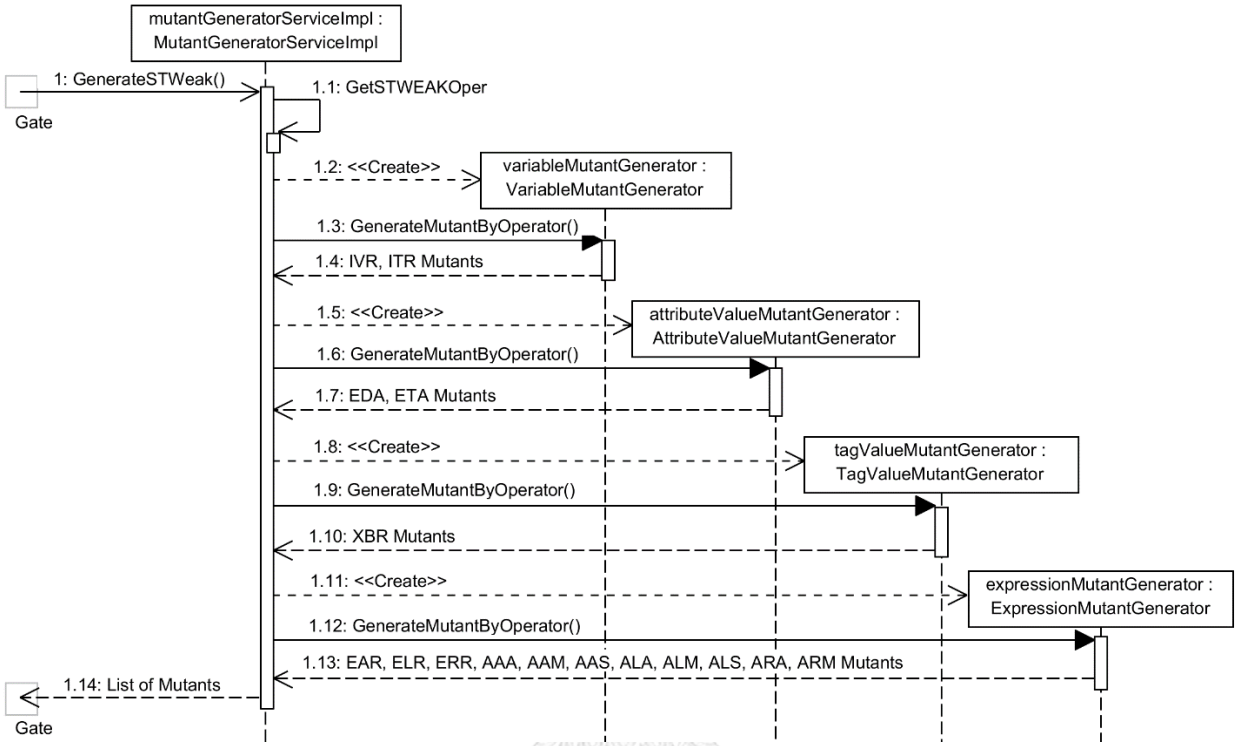
ภาพที่ 4-11 แผนภาพลำดับ Generate Mutant

2.1) แผนภาพลำดับ ST-WEAK Mutant Generator

แผนภาพลำดับ ST-WEAK Mutant Generator เป็นแผนภาพที่แสดงถึงการสร้างมิวแตนท์ตามระดับการทดสอบวีคมิวเทชัน ST-WEAK/1 ดังภาพที่ 4-12 โดยเริ่มต้นจากคลาส MutantGeneratorServiceImpl ถูกเรียกใช้ฟังก์ชัน GenerateSTWeak จากนั้นเรียกใช้ฟังก์ชัน GetSTWeakOper ของคลาส MutantGeneratorServiceImpl เพื่อดึงรายการตัวดำเนินการ และเรียกใช้ฟังก์ชัน GenerateMutantByOperator ของคลาสที่สืบทอดจากคลาล MutantGeneratorBase เพื่อสร้างมิวแตนท์ โดยมีรายละเอียดดังนี้

- คลาส VariableMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ IVR และ ITR
- คลาส AttributeValueMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ EDA และ ETA

- คลาส TagValueMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ XBR
- คลาส ExpressionMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ EAR, ELR, ERR, AAA, AAM, AAS, ALA, ALM, ALS, ARA และ ARM



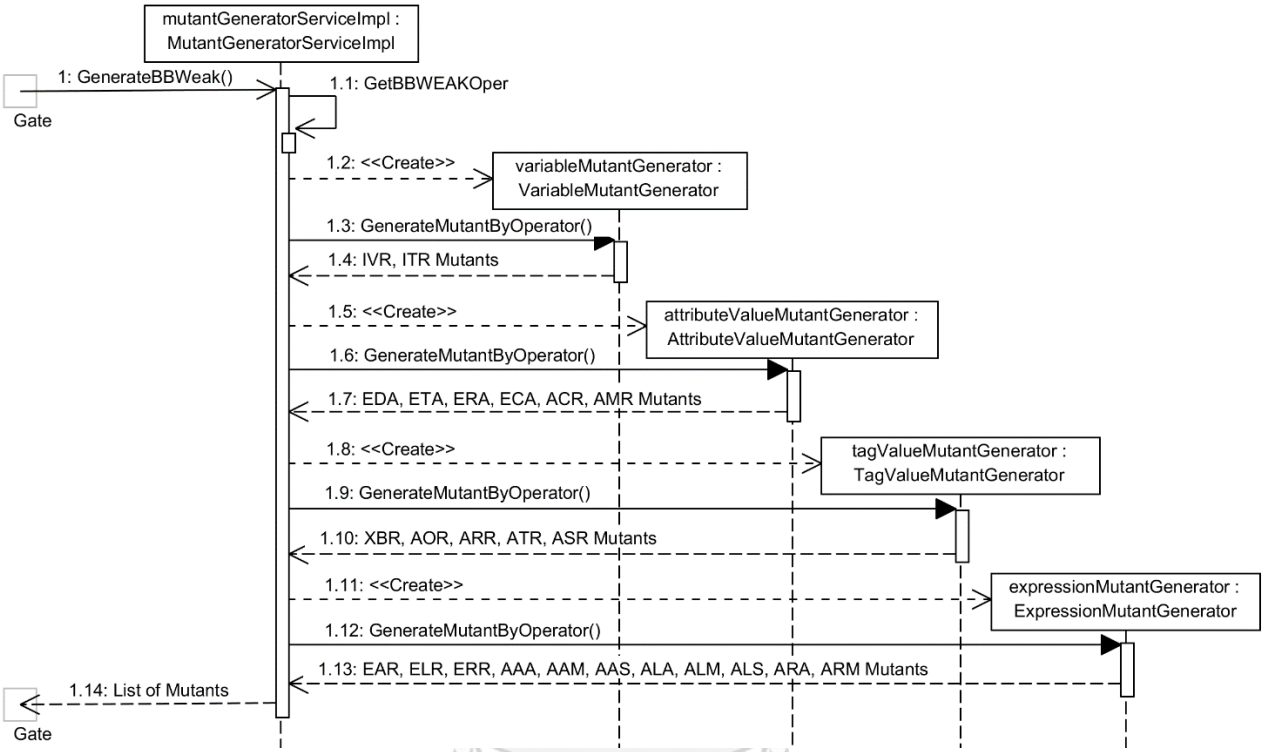
ภาพที่ 4-12 แผนภาพลำดับ ST-WEAK Mutant Generator

2.2) แผนภาพลำดับ BB-WEAK Mutant Generator

แผนภาพลำดับ BB-WEAK Mutant Generator เป็นแผนภาพที่แสดงถึงการสร้างมิวแตนท์ตามระดับการทดสอบวีคมิวเทชัน BB-WEAK/1 และ BB-WEAK/N ดังภาพที่ 4-13 โดยเริ่มต้นจากคลาส MutantGeneratorServiceImpl ถูกเรียกใช้ฟังก์ชัน GenerateBBWeak จากนั้นเรียกใช้ฟังก์ชัน GetBBWeakOper ของคลาส MutantGeneratorServiceImpl เพื่อดึงรายการตัวดำเนินการ และเรียกใช้ฟังก์ชัน GenerateMutantByOperator ของคลาสที่สืบทอดจากคลาล MutantGenerator-Base เพื่อสร้างมิวแตนท์ โดยมีรายละเอียดดังนี้

- คลาส VariableMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ IVR และ ITR
- คลาส AttributeValueMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ EDA, ETA, ERA, ECA, ACR, AMR

- คลาส TagValueMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ XBR, AOR, ARR, ATR, ASR
- คลาส ExpressionMutantGenerator สร้างมิวแตนท์ของตัวดำเนินการ EAR , ELR, ERR, AAA, AAM, AAS, ALA, ALM, ALS, ARA และ ARM

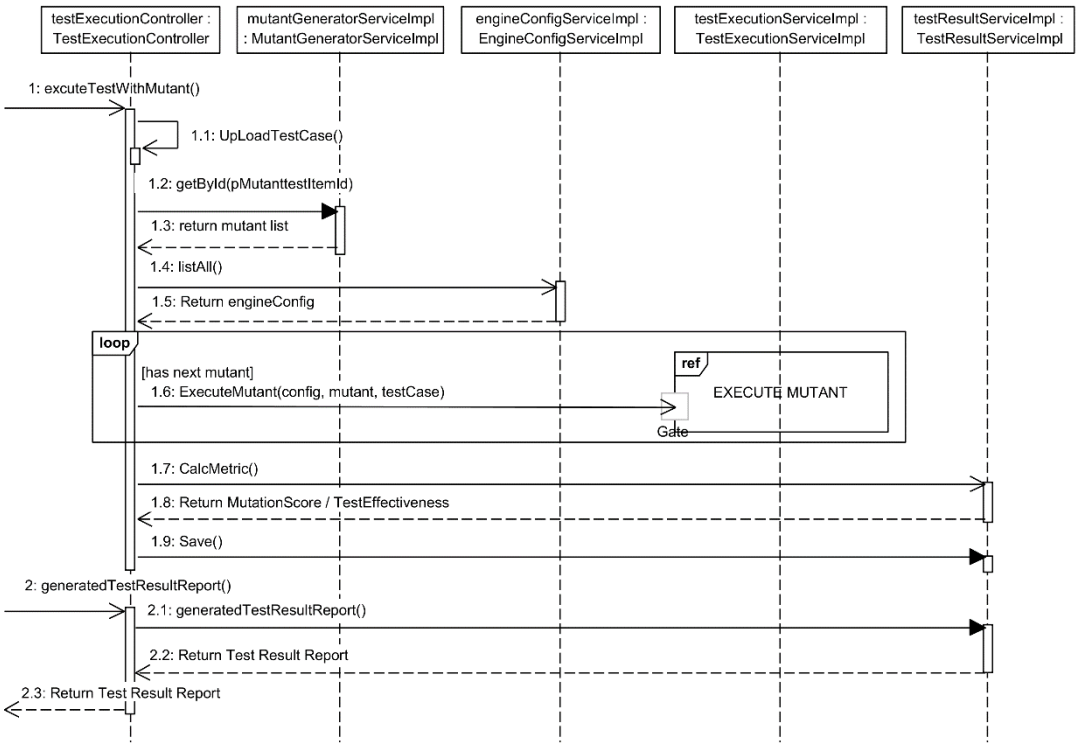


ภาพที่ 4-13 แผนภาพลำดับ BB-WEAK Mutant Generator

3) แผนภาพลำดับ Execute Test

แผนภาพลำดับ Execute Test เป็นแผนภาพที่แสดงถึงขั้นตอนการทดสอบแบบจำลองมิวแตนท์ดังภาพที่ 4-14 เริ่มต้นจากนักทดสอบเลือกแบบจำลอง และกรณีทดสอบ จากนั้น testExecutionController เรียกใช้ฟังก์ชัน excuteTestWithMutant โดยการนำกรณีทดสอบเข้าสู่เครื่องมือจากฟังก์ชัน UploadTestCase ของคลาส testExecution Controller จากนั้นเครื่องมือดึงมิวแตนท์ของแบบจำลองที่สนใจจากฟังก์ชัน getById ของคลาส MutantGeneratorServiceImpl กับข้อมูลการเชื่อมต่อเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น จากฟังก์ชัน listAll ของคลาส EngineConfigServiceImpl และเรียกใช้ฟังก์ชัน TestMutant ของคลาส TestExecutionServiceImpl เพื่อทดสอบแบบจำลองผ่านเครื่องประมวลผลแบบจำลองทีละมิวแตนท์ และทำซ้ำจนครบทุกมิวแตนท์ เมื่อทดสอบมิวแตนท์ครบถ้วนคลาส TestResultServiceImpl เรียกใช้ฟังก์ชัน CalcMetric เพื่อคำนวณ

คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบต่อไป เมื่อนักทดสอบกดปุ่ม Generate Report เครื่องมือจะเรียกใช้ฟังก์ชัน generatedTestResultReport ของคลาส testExecutionController เพื่อสร้างรายงานการทดสอบผ่านฟังก์ชัน generated TestResultReport ของคลาส testResultServiceImpl

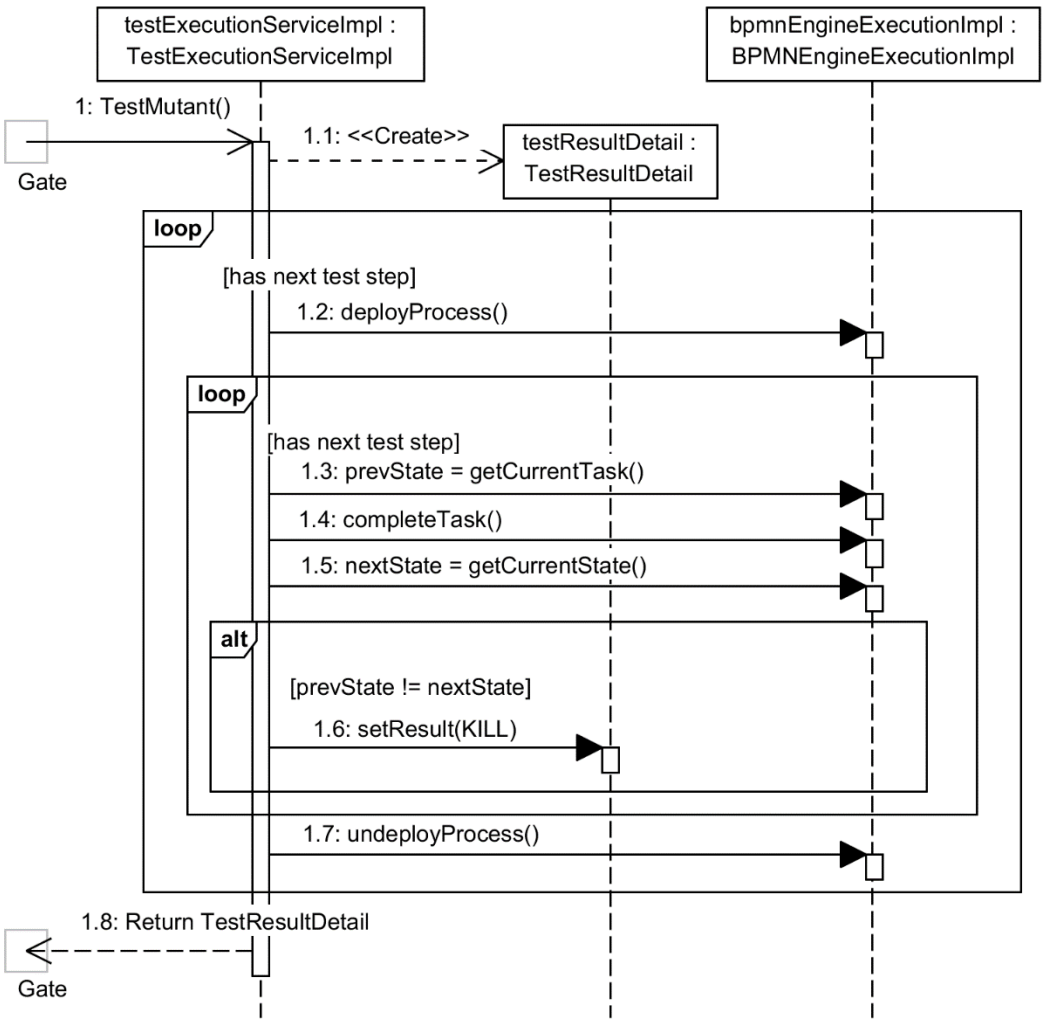


ภาพที่ 4-14 แผนภาพลำดับ Execute Test

3.1) แผนภาพลำดับ Test Mutant

แผนภาพลำดับ Test Mutant เป็นแผนภาพที่แสดงถึงขั้นตอนการทดสอบแบบจำลองผ่านเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นดังภาพที่ 4-15 โดยเริ่มต้นจากคลาส TestExecutionServiceImpl ถูกเรียกใช้ฟังก์ชัน TestMutant จากนั้นคลาส Test ResultDetail ถูกสร้างขึ้นมา เพื่อเก็บผลลัพธ์การทดสอบ เครื่องมือดำเนินการทดสอบตามกรณีทดสอบทีละกรณีทดสอบ ตั้งแต่ขั้นตอนติดตั้งแบบจำลองมิวแตนท์โดยเรียกใช้ฟังก์ชัน deployProcess ของคลาส BPMNEngineExcutionImpl และดำเนินการในแต่ละกิจกรรมจากการเรียกใช้ฟังก์ชัน completeTask ของคลาส BPMN EngineExcutionImpl เพื่อส่งข้อมูลนำเข้าของกิจกรรมตามขั้นตอนการทดสอบ (Test Step) เข้าไปเครื่องประมวลผลแบบจำลอง โดยทีก่อน และหลังการทำงานเครื่องมือได้จัดเก็บสถานะก่อนหน้า (prevState) และสถานะถัดไป (nextState) จากนั้นเครื่องมือ

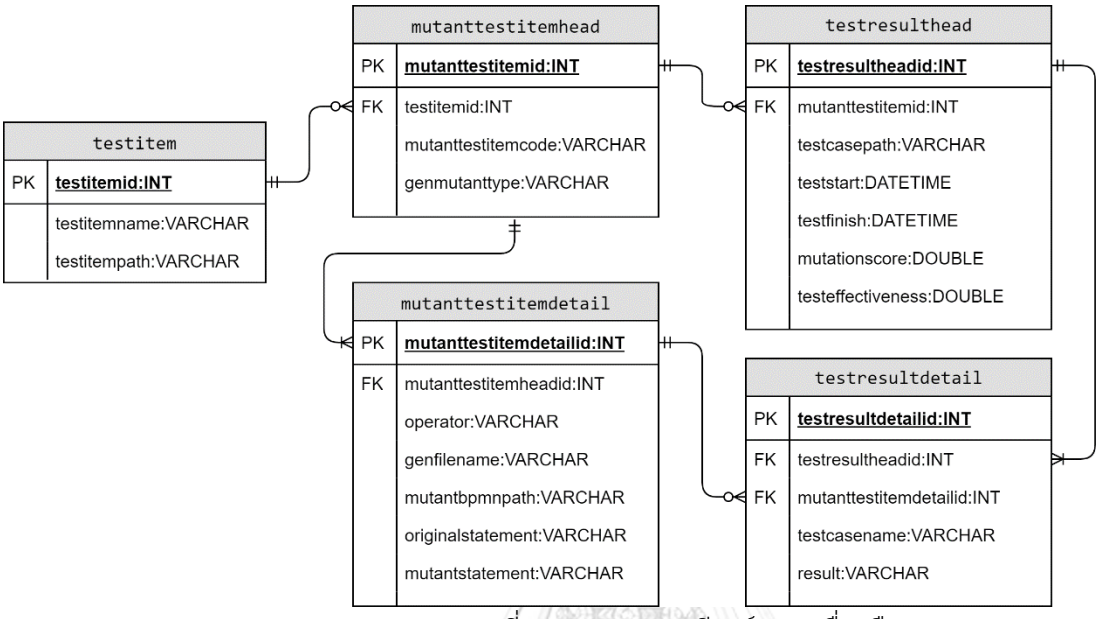
เปรียบเทียบสถานะ ถ้าสถานะก่อนหน้าแตกต่างจากสถานะถัดไป เครื่องมือบันทึกผลการทดสอบมิวแตนท์ว่าถูกฆ่าโดยเรียกใช้ฟังก์ชัน setResult ของคลาส BPMNEngineExcutionImpl และเมื่อทดสอบตามกรณีทดสอบย่อยเสร็จสิ้น เครื่องมือถอนการติดตั้งแบบจำลองโดยเรียกฟังก์ชั่น unDeployProcess ของคลาส BPMNEngineExcutionImpl



ภาพที่ 4-15 แผนภาพลำดับ Test Mutant

4.1.5 โครงสร้างฐานข้อมูล

โครงสร้างฐานข้อมูลของเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นแสดงโดยแผนภาพอีอาร์ (ER – Entity Relationship Diagram) ดังภาพที่ 4-16 โดยมีรายละเอียดดังนี้



ภาพที่ 4-16 แผนภาพอีอาร์ของเครื่องมือ

1) เอนทิตี้ testitem เป็นตารางที่ใช้เก็บข้อมูลแบบจำลองต้นฉบับ
2) เอนทิตี้ mutanttestitemhead เป็นตารางที่ใช้เก็บแบบจำลองมิวแตนท์ที่เกิดจากแบบจำลองต้นฉบับ
3) เอนทิตี้ mutanttestitemdetail เป็นตารางที่ใช้เก็บรายละเอียดของแบบจำลองมิวแตนท์แยกตามแต่ละตัวดำเนินการ
4) เอนทิตี้ testresulthead เป็นตารางที่ใช้เก็บข้อมูลสรุปผลการทดสอบ
5) เอนทิตี้ testresultdetail เป็นตารางที่ใช้เก็บผลการทดสอบของแต่ละแบบจำลองมิวแตนท์

4.2 การพัฒนาเครื่องมือสร้าง และทดสอบมิวแตนท์

4.2.1 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ

สภาพแวดล้อมที่ใช้พัฒนาเครื่องมือ มีรายละเอียด ดังนี้

4.2.1.1 ฮาร์ดแวร์ (Hardware)

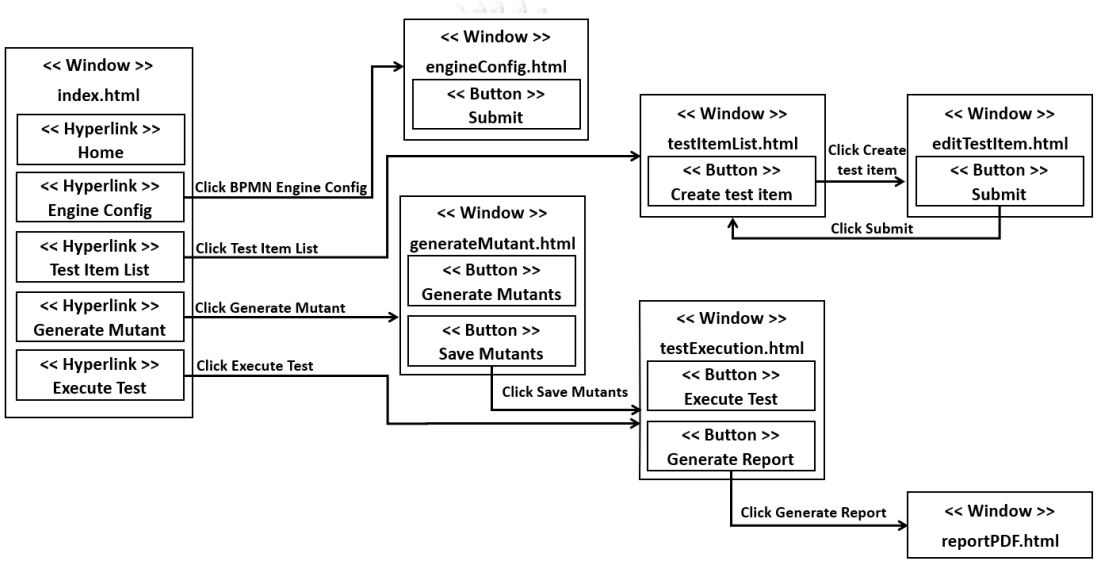
1) เครื่องคอมพิวเตอร์โน๊ตบุ๊ค (Notebook) หน่วยประมวลผลอินเทลคอร์ไอเซเว่น รุ่นที่ 8 ความเร็ว 1.8 กิกะเฮิร์ซ์ (Intel(R) Core(TM) i7-8550U 1.80 GHz)
2) หน่วยความจำหลัก (RAM) 16.00 กิกะไบต์ (16.00 GB)

4.2.1.2 ซอฟต์แวร์ (Software)

1) ระบบปฏิบัติการ (Operating System) ไมโครซอฟท์วินโดวส์ 10 โปร (Windows 10 Pro)
2) เครื่องมือที่ใช้พัฒนาไมโครซอฟท์วิชวลสตูดิโอโค้ดเวอร์ชัน 1.35.1 (Microsoft Visual Studio Code 1.35.1)
3) ภาษาที่ใช้พัฒนา คือ ภาษาจาวา (Java) เวอร์ชัน 8 ขึ้นไป
4) เฟรมเวิร์กที่ใช้พัฒนา คือ สปริงบูต (Spring Boot)
5) เว็บเซิร์ฟเวอร์อาพาเช่ทอมแคทเวอร์ชัน 8.0 (Apache Tomcat 8.0)
6) เครื่องมือประมวลผลแบบจำลองบีพีเอ็มเอ็นคามุนดาเวอร์ชัน 7.80 (Camunda BPMN Engine 7.80)
7) เครื่องมือสำหรับสร้างแบบจำลองบีพีเอ็มเอ็นคามุนดาโมเดลเลอร์เวอร์ชัน 1.11.3 (Camunda Modeler 1.11.3)
8) เครื่องมือที่ใช้ออกแบบรายงานแจสเปอร์ซอฟท์สตูดิโอเวอร์ชัน 6.8.0 (Jaspersoft Studio 6.8.0)
9) เว็บเบราว์เซอร์ (Web Browser) กูเกิลโครม (Google Chrome) เวอร์ชัน 74 ขึ้นไป

4.2.2 ส่วนต่อประสานกับผู้ใช้ของเครื่องมือ

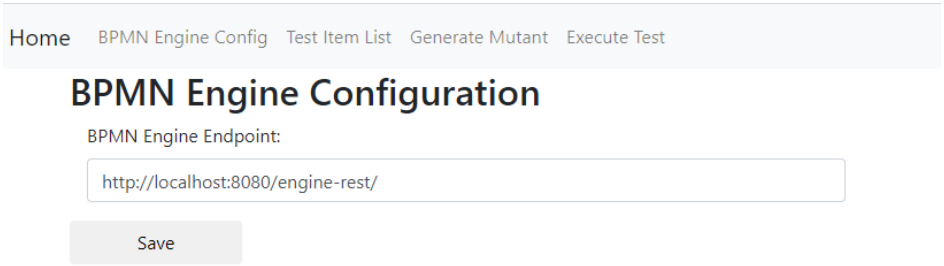
โครงสร้างของส่วนต่อประสานกับผู้ใช้ถูกอธิบายด้วยแผนภาพวินโดว์เนวิเกชัน (Window Navigation Diagram) ซึ่งได้อธิบายส่วนต่อประสานทั้งหมดของเครื่องมือ และความสัมพันธ์ระหว่างส่วนต่อประสานด้วยกัน ซึ่งเริ่มจากส่วนต่อประสาน index.html ซึ่งหน้าจอหลักที่ใช้ติดต่อกับผู้ใช้งาน ถัดมาเป็นส่วนต่อประสาน engineConfig.html เป็นส่วนที่ติดต่อกับผู้ใช้งานเป็นอันดับแรกเรียงลำดับไปจนถึงส่วนต่อประสาน testExecution.html เป็นอันดับสุดท้ายดังภาพที่ 4-17 โดยจากแผนภาพวินโดว์เนวิเกชัน สามารถอธิบายรายละเอียดแต่ละส่วนต่อประสาน ได้ดังนี้



ภาพที่ 4-17 แผนภาพวินโดว์เนวิเกชันของเครื่องมือ

1) หน้าจอ engineConfig.html

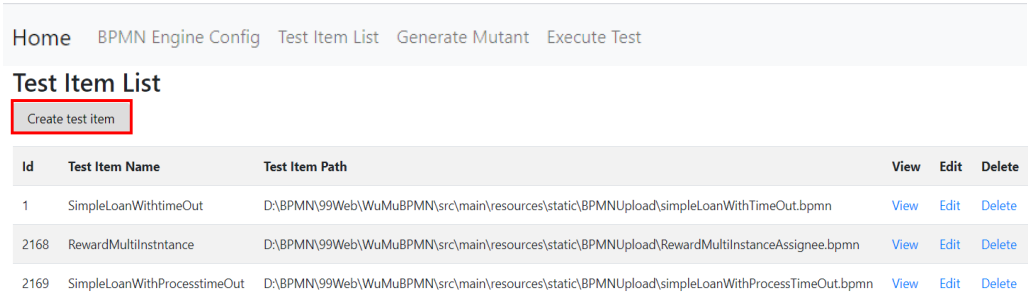
หน้าจอ engineConfig.html เป็นหน้าที่จอที่ให้นักทดสอบสามารถกำหนดที่อยู่ของเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นได้ ดังภาพที่ 4-18



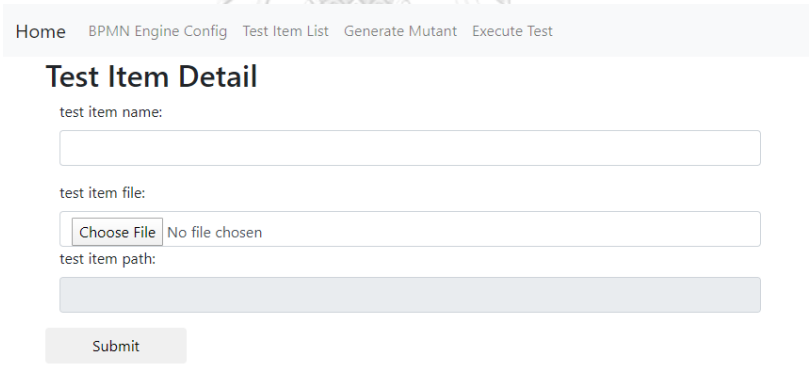
ภาพที่ 4-18 หน้าจอ engineConfig.html

2) หน้าจอ testItemList.html และหน้าจอ editTestItem.html

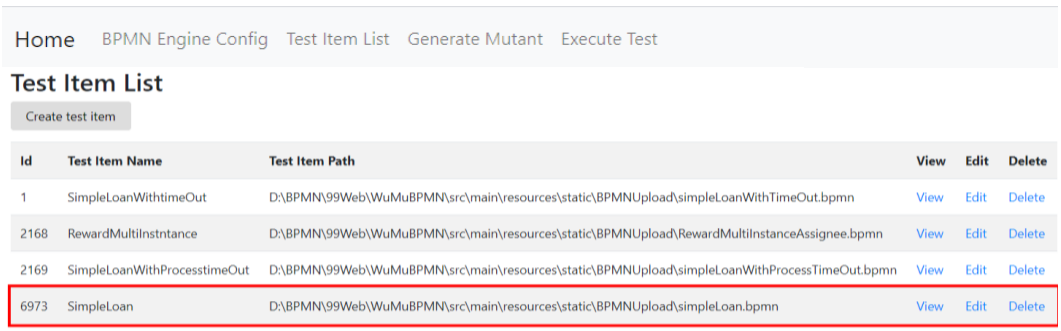
หน้าจอ testItemList.html เป็นหน้าจอที่ให้นักทดสอบสามารถจัดการแบบจำลองบีพีเอ็มเอ็นได้ โดยเมื่อกดปุ่ม Create Test Item เครื่องมือจะพาไปยังหน้าจอ editTestItem.html ดังภาพที่ 4-19 และภาพที่ 4-20 เพื่อให้นักทดสอบเลือกแบบจำลองบีพีเอ็มเอ็นที่มีนามสกุลของไฟล์ .bpmn และกรอกชื่อแบบจำลอง จากนั้นกดปุ่ม Submit เครื่องมือนำแบบจำลองบีพีเอ็มเอ็นอัพโหลดไปยังเว็บเซิร์ฟเวอร์ดังภาพที่ 4-21



ภาพที่ 4-19 หน้าจอ testItemList.html

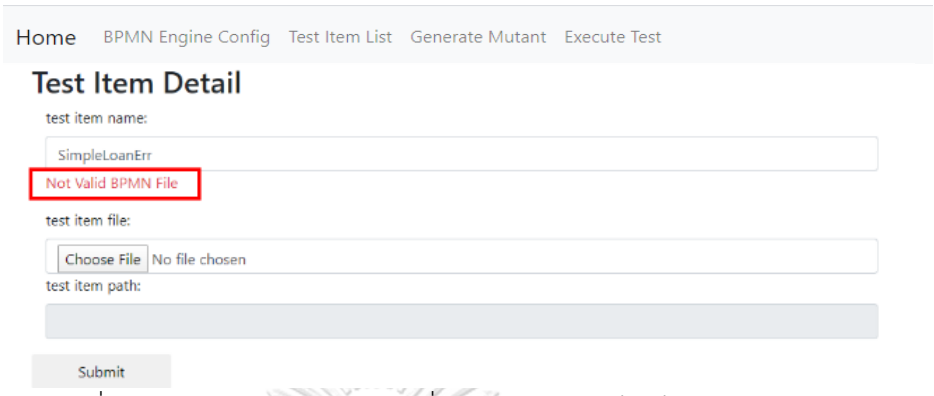


ภาพที่ 4-20 หน้าจอ editTestItem.html



ภาพที่ 4-21 หน้าจอ testItemList.html หลังกดปุ่ม Submit ที่หน้าจอ editTestItem.html

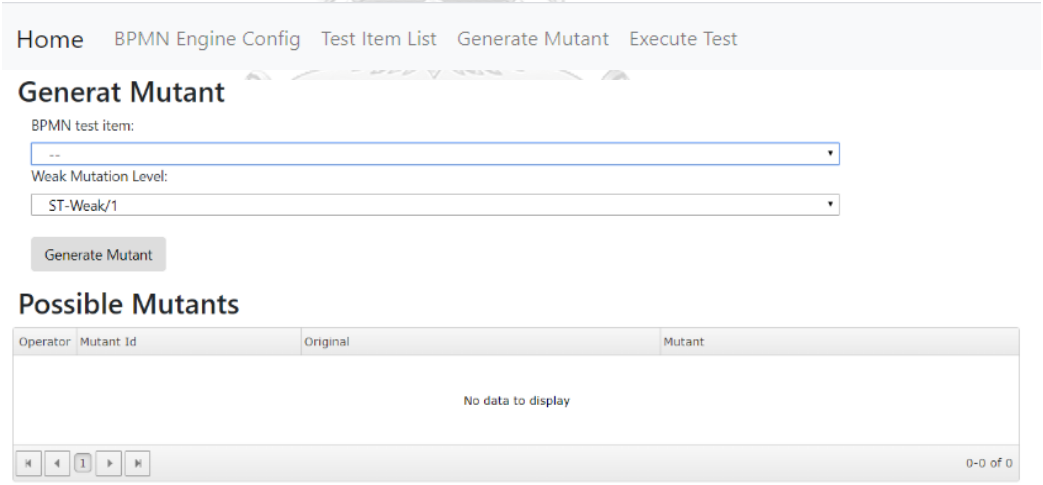
กรณีที่อัพโหลดแบบจำลองเข้าเครื่องมือแล้วพบว่าแบบจำลองนั้นไม่ตรงตามมาตรฐาน เครื่องมือมีการแสดงข้อความแจ้งเตือน ดังภาพที่ 4-22



ภาพที่ 4-22 หน้าจอแจ้งเตือนกรณีที่แบบจำลองบีพีเอ็มเอ็นไม่ตรงตามมาตรฐาน

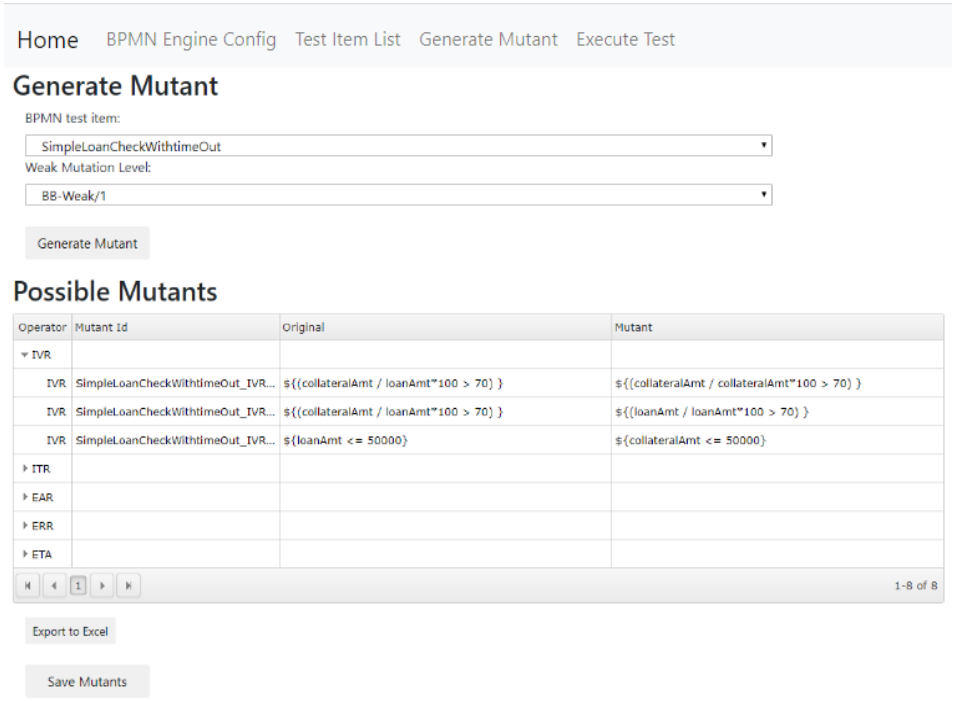
3) หน้าจอ generateMutant.html

หน้าจอ testItemList.html เป็นหน้าจอที่ให้นักทดสอบสามารถเลือกแบบจำลองที่ได้อัพโหลดไว้จากหน้าจอ testItemList.html เพื่อสร้างมิวแตนท์จากการเลือกแบบจำลองและประเภทของการทดสอบวีคมิวเทชัน ดังภาพที่ 4-23



ภาพที่ 4-23 หน้าจอการสร้างมิวแตนท์

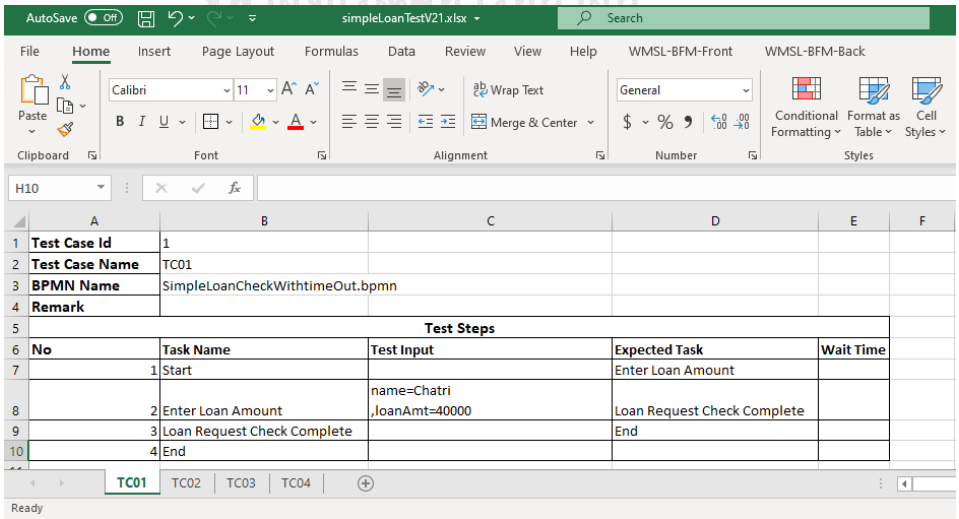
นักทดสอบกดปุ่ม Generate Mutant จากนั้นเครื่องมือประมวลผลและแสดงรายการมิวแตนท์แยกตามประเภทของตัวดำเนินการ ดังภาพที่ 4-24 หลังจากนักทดสอบตรวจทานมิวแตนท์ที่เกิดขึ้นเรียบร้อยแล้วกดปุ่ม Save Mutant เพื่อให้เครื่องมือบันทึกมิวแตนท์ที่เกิดขึ้น หรือกดปุ่ม Export to Excel เพื่อส่งออกมิวแตนท์ที่เกิดขึ้นออกมาในรูปแบบไฟล์เอกซ์เซล (.xls)



ภาพที่ 4-24 หน้าจอแสดงรายการมิวแตนท์แยกตามประเภทของตัวดำเนินการมิวเทชัน

4) หน้าจอ testExecution.html

หน้าจอ testExecution.html เป็นหน้าจอที่ให้นักทดสอบเลือกแบบจำลองมิวแตนท์ และประเภทของการทดสอบ จากนั้นนักทดสอบสามารถเลือกกรณีทดสอบที่เตรียมไว้ในรูปแบบของไฟล์เอกซ์เซล ดังภาพที่ 4-25 โดยที่หนึ่งกรณีทดสอบแทนด้วยหนึ่งชีต (Sheet) ของไฟล์เอกซ์เซล ซึ่งในแต่ละชีตมีโครงสร้าง ดังนี้



ภาพที่ 4-25 ตัวอย่างของกรณีทดสอบที่อยู่ในรูปแบบไฟล์เอกซ์เซล

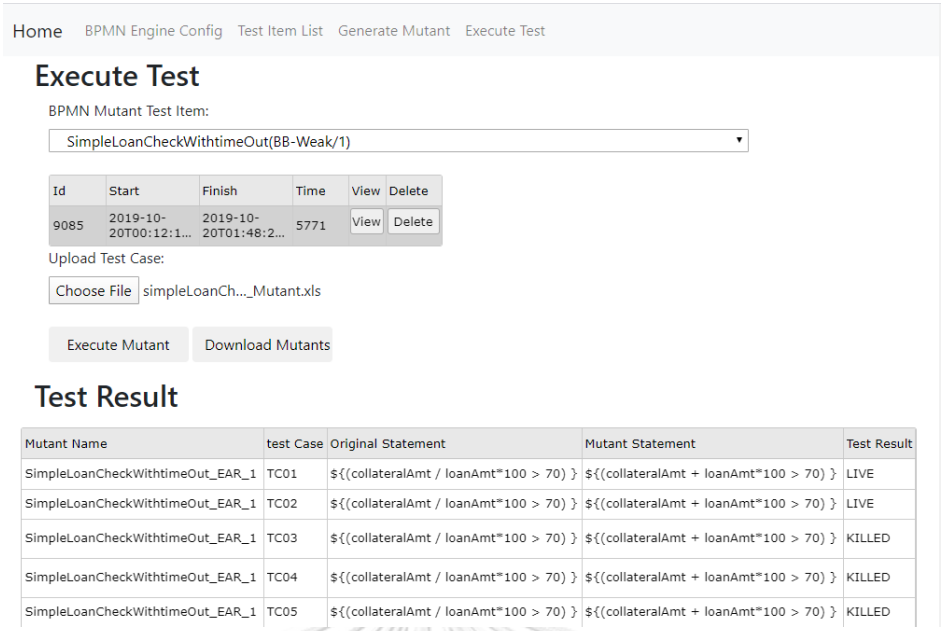
- ส่วนที่ 1: แสดงข้อมูลทั่วไปของกรณีทดสอบ ได้แก่ Test Case Id, Test Case Name, BPMN Name และ Remark
- ส่วนที่ 2: แสดงลำดับการทดสอบ (Test Steps) ในรูปแบบตาราง โดยมีรายละเอียด ดังตารางที่ 4-4

ตารางที่ 4-4 รายละเอียดของลำดับการทดสอบ

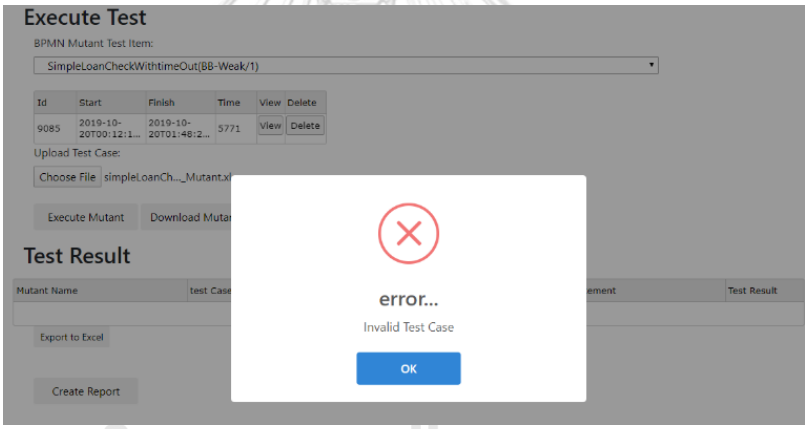
คอลัมน์	คำอธิบาย
No	ลำดับของการทดสอบ
Task Name	กิจกรรมก่อนกรอกข้อมูลนำเข้า
Test Input	ข้อมูลนำเข้าของกิจกรรม (ถ้ามี) โดยอยู่ในรูปแบบของคีย์ค่า เช่น name=Chatri
Expected Task	กิจกรรมที่คาดหวังหลังกรอกข้อมูลนำเข้า
Wait Time	เวลาที่ต้องรอในกรณีที่กิจกรรมนั้นมีการใช้งานร่วมกับสัญลักษณ์ของบีพีเอ็มเอ็นที่ต้องรอเวลาอาทิ เช่น Timer Boundary Event โดยแสดงในรูปแบบของเวลาตามมาตรฐาน ISO8601 เช่น ถ้าต้องรอเป็นเวลา 5 นาที ช่อง Wait Time ให้กำหนดเป็น PT5M

จากนั้นนักทดสอบกดปุ่ม Execute เพื่อให้เครื่องมือส่งแบบจำลองมิวแตนท์ และกรณีทดสอบเข้าไปยังเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น เมื่อเครื่องมือทดสอบแบบจำลองมิวแตนท์เสร็จสิ้น เครื่องมือแสดงผลการทดสอบลงในตาราง Test Result ดังภาพที่ 4-26

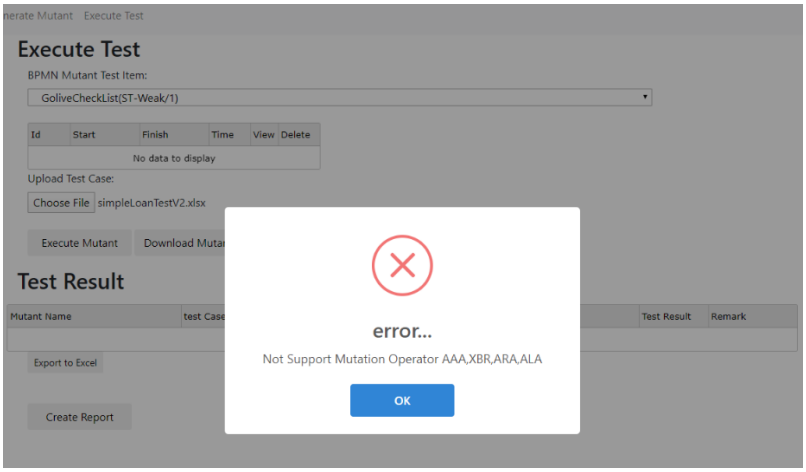
กรณีที่นักทดสอบกำหนดรูปแบบกรณีทดสอบผิดรูปแบบเครื่องมือแจ้งเตือนดังภาพที่ 4-27 และกรณีที่เลือกแบบจำลองที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นไม่รองรับเครื่องมือแสดงข้อความแจ้งเตือนดังภาพที่ 4-28



ภาพที่ 4-26 หน้าจอแสดงผลการทดสอบแบบจำลอง

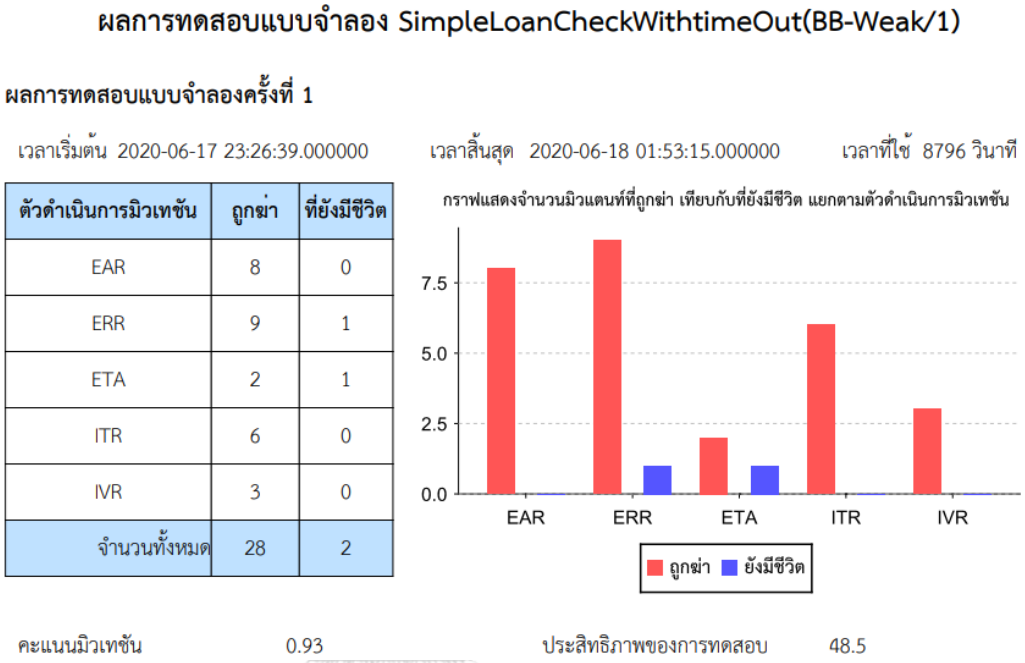


ภาพที่ 4-27 หน้าจอแจ้งเตือนกรณีที่กรณีทดสอบผิดรูปแบบ



ภาพที่ 4-28 หน้าจอแจ้งเตือนกรณีที่เครื่องประมวลผลแบบจำลองไม่รองรับตัวดำเนินการมิวเทชัน

เมื่อนักทดสอบกดปุ่ม Create Report เครื่องมือสร้างรายงาน เพื่อสรุปข้อมูลผลการทดสอบซึ่งประกอบไปด้วย เวลาที่ใช้ในการทดสอบ จำนวนมิวแตนท์ที่ถูกฆ่า จำนวน จำนวนมิวแตนท์ที่ยังมีชีวิต คะแนนมิวเทชัน และประสิทธิภาพของการทดสอบ ดังภาพที่ 4-29



ผลการทดสอบแบบจำลอง SimpleLoanCheckWithtimeOut(BB-Weak/1)

ผลการทดสอบแบบจำลองครั้งที่ 1

เวลาเริ่มต้น 2020-06-17 23:26:39.000000 เวลาสิ้นสุด 2020-06-18 01:53:15.000000 เวลาที่ใช้ 8796 วินาที

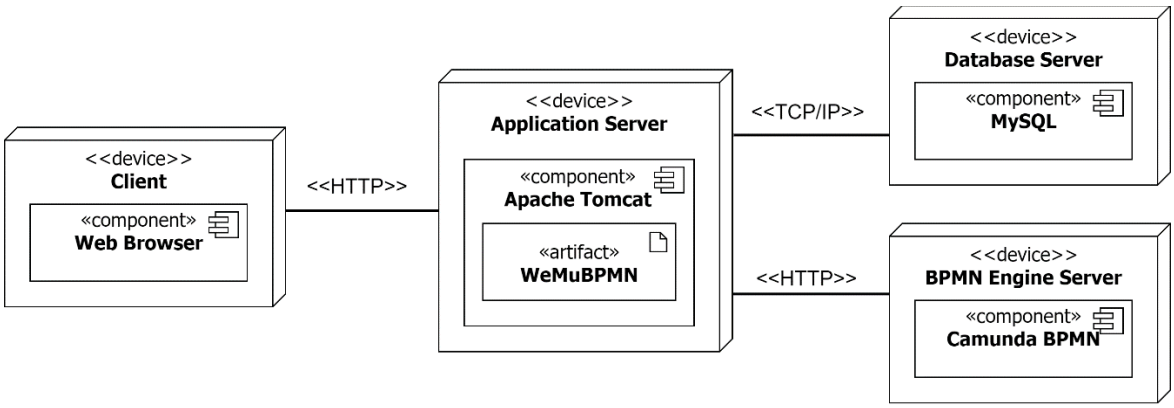
ตัวดำเนินการมิวเทชัน	ถูกฆ่า	ที่ยังมีชีวิต
EAR	8	0
ERR	9	1
ETA	2	1
ITR	6	0
IVR	3	0
จำนวนทั้งหมด	28	2

คะแนนมิวเทชัน 0.93 ประสิทธิภาพของการทดสอบ 48.5

ภาพที่ 4-29 รายงานสรุปผลการทดสอบ

4.2.3 แผนภาพการติดตั้ง (Deployment Diagram)

แผนภาพการติดตั้งของเครื่องมือแสดงดังภาพที่ 4-30 โดยแสดงให้เห็นถึงไคลเอนต์ (Client) เรียกใช้งานเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็น หรือ วีมูบีพีเอ็มเอ็น (WeMuBPMN) ถูกติดตั้งที่แอปพลิเคชันเซิร์ฟเวอร์ (Application Server) โดยทดสอบแบบจำลองกับเครื่องประมวลผลแบบจำลองบีพีเอ็นเอ็นคามุนดา และมีการใช้งานฐานข้อมูลมายเอสคิวแอล (MySQL) ในการจัดเก็บข้อมูลที่เซิร์ฟเวอร์ฐานข้อมูล (Database Server)



ภาพที่ 4-30 แผนภาพการติดตั้งของเครื่องมือ

บทที่ 5
การทดสอบเครื่องมือ

บทนี้นำเสนอแนวทางในการทดสอบเครื่องมือทดสอบวีคมิวเทชันสำหรับแบบจำลองบีพีเอ็มเอ็น โดยอธิบายจากสภาพแวดล้อมที่ใช้ในการทดสอบเครื่องมือ ขั้นตอนแต่ละขั้นในการทดสอบแบบจำลองบีพีเอ็มเอ็นที่ใช้ในการทดสอบ กรณีทดสอบที่ใช้ทดสอบแบบจำลองบีพีเอ็มเอ็น ผลของการทดสอบเครื่องมือ และสรุปผลของการทดสอบเครื่องมือ

5.1 สภาพแวดล้อมที่ใช้ในการทดสอบ

การทดสอบเครื่องมือจะทดสอบโดยใช้สภาพแวดล้อมเดียวการพัฒนาเครื่องมือดังที่อธิบายไว้ในบทที่ 4 ของวิทยานิพนธ์ฉบับนี้

5.2 ขั้นตอนในการทดสอบเครื่องมือที่พัฒนา

1) ผู้ทดสอบเตรียมแบบจำลองที่ใช้สำหรับการทดสอบ โดยอธิบายเพิ่มเติมในหัวข้อที่ 5.3
2) ผู้ทดสอบกำหนดข้อมูลการเชื่อมต่อเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น จากเมนู BPMN Engine Config ของเครื่องมือ
3) ผู้ทดสอบนำแบบจำลองเข้าสู่เครื่องมือผ่านเมนู Test Item List ของเครื่องมือ
4) ผู้ทดสอบเรียกใช้เมนู Generate Mutant เพื่อสร้างมิวแตนท์ตามระดับการทดสอบวีคมิวเทชัน ST-WEAK/1, BB-WEAK/1 และ BB-WEAK/N
5) ผู้ทดสอบเรียกใช้เมนู Execute Test เพื่อทดสอบแบบจำลองมิวแตนท์ โดยเลือกแบบจำลองมิวแตนท์ที่สร้างขึ้นจากข้อ 4 พร้อมทั้งนำเข้ากรณีทดสอบ
6) ผู้ทดสอบรวบรวมผลการทดสอบตามรายงานที่ได้แสดงในบทที่ 4 ของวิทยานิพนธ์ฉบับนี้

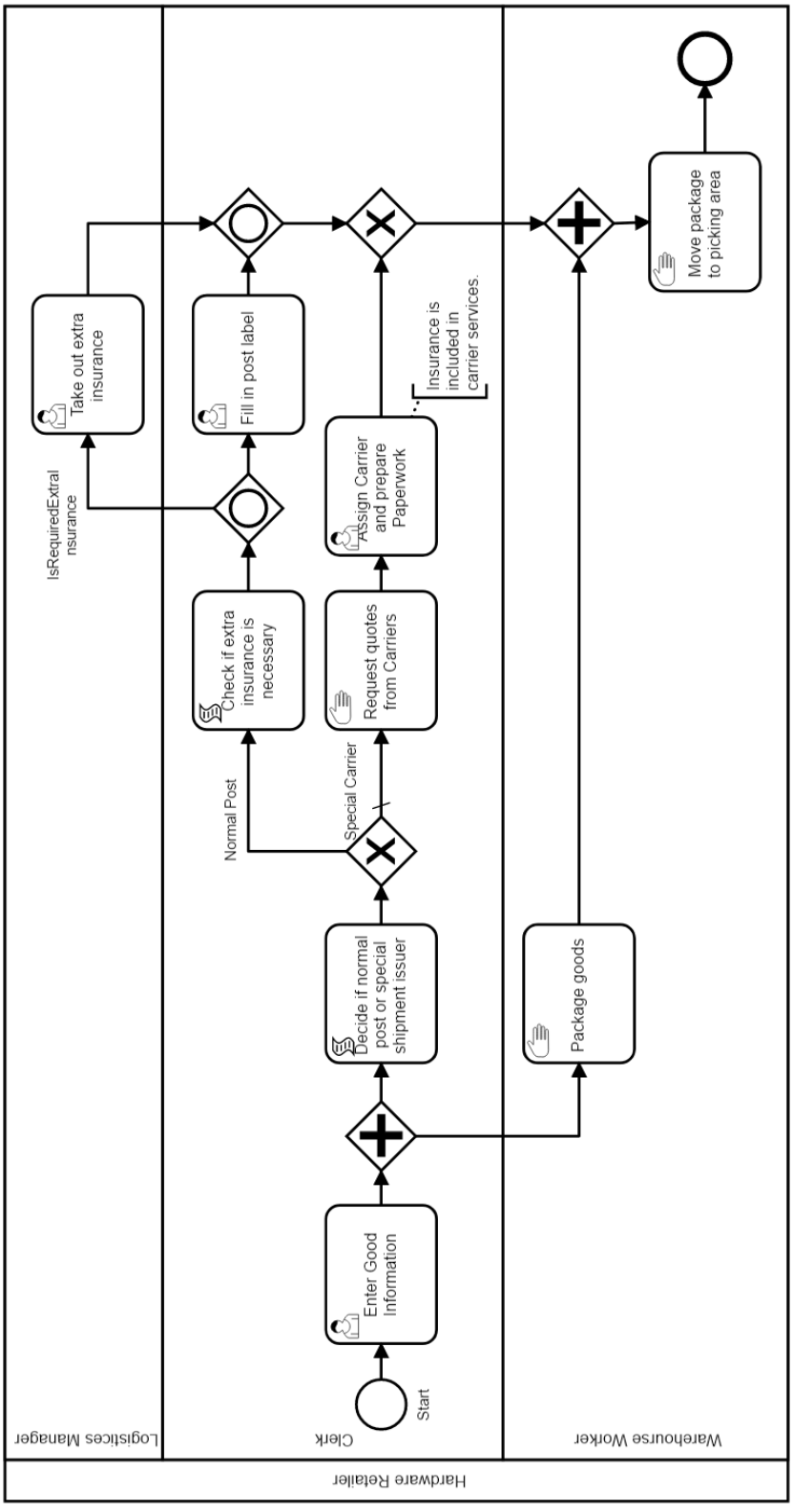
5.3 แบบจำลองที่ใช้สำหรับการทดสอบ

แบบจำลองที่ใช้สำหรับการทดสอบด้วยเครื่องมือทดสอบวีคมิวเทชันมีทั้งหมด 8 แบบจำลอง ซึ่งเป็นแบบจำลองเดิมที่มีอยู่ หรือถูกพัฒนาขึ้นโดยผู้วิจัย ได้แก่ แบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์ แบบจำลองการตรวจสอบยอดเงินกู้ แบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง แบบจำลองตรวจสอบการเบิกจ่าย แบบจำลองการสร้างรายงานข้อบกพร่องของระบบ แบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด แบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ และแบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบ โดย 2 แบบจำลองหลังจะเป็นทดสอบเฉพาะการสร้างมิวแตนท์เท่านั้น เนื่องจากเครื่องประมวลผลแบบจำลองไม่สามารถทดสอบมิวแตนท์ได้

5.3.1 แบบจำลองที่ 1 แบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์

แบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์เป็นแบบจำลองที่แสดงขั้นตอนการส่งสินค้า ดังภาพที่ 5-1 โดยมีขั้นตอน ดังนี้

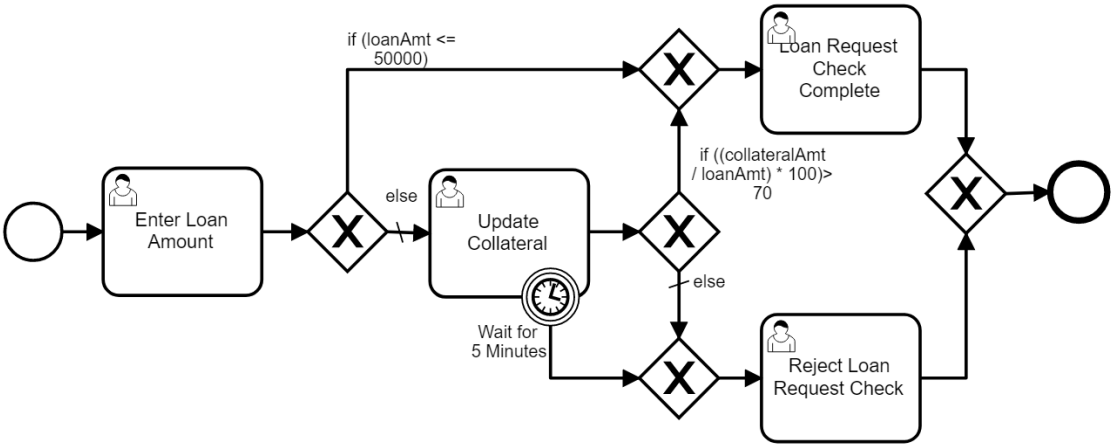
1) พนักงานขายทำหน้าที่กรอกข้อมูลสินค้าที่กิจกรรม “Enter Good Information” ได้แก่ ชื่อสินค้า, หมวดหมู่ของสินค้า, ราคาของสินค้า และผู้ผลิตสินค้า
2) กระบวนการทำกิจกรรมที่ขนานกัน (Parallel Gateway) ได้แก่
    2.1) พนักงานโกดังสินค้าทำกิจกรรม “Package goods” ซึ่งเป็นแมนนวลทาร์ก
    2.2) กิจกรรม “Decide if normal post or special shipment issuer” ซึ่งเป็นสคริปต์ทาร์กที่คำนวณภาษีมูลค่าเพิ่ม และตรวจสอบผู้ผลิตสินค้าเป็นกลุ่มที่ต้องจัดส่งสินค้าด้วยวิธีการพิเศษ โดยใช้เงื่อนไขทางเลือก (Exclusive Gateway)
    2.3) กรณีจัดส่งแบบทั่วไป (Normal Post) โดยตรวจสอบจากเงื่อนไขยอดรวมของสินค้ากับภาษีมูลค่าเพิ่มน้อยกว่า 10,000 บาท และผู้ผลิตสินค้าไม่ถูกจัดอยู่กลุ่มที่ต้องจัดส่งสินค้าด้วยวิธีการพิเศษ จะเข้ากิจกรรม “Check if extra insurance is necessary” ซึ่งเป็นสคริปต์ทาร์ก
        2.3.1) กิจกรรม “Check if extra insurance is necessary” ซึ่งเป็นสคริปต์ทาร์กที่ทำหน้าตรวจสอบหมวดหมู่ของสินค้าว่าอยู่กลุ่มที่ต้องจัดทำประกันการขนส่งสินค้า จากนั้นตรวจสอบเงื่อนไขของกิจกรรมที่ขนานกัน (Inclusive Gateway)
        2.3.2) กรณีที่ต้องจัดทำประกันการขนส่งสินค้า ผู้จัดการการขนส่งเข้ามาทำกิจกรรม “Take out extra insurance” เพื่อกรอกข้อมูลประกันภัยเพื่อเติม ได้แก่ ผู้ให้บริการประกัน และเลขที่สัญญาประกัน
        2.3.3) พนักงานโกดังสินค้าทำกิจกรรม “Fill in post label” เพื่อกำหนดข้อมูลการจัดส่ง ได้แก่ ที่อยู่ผู้จัดส่งสินค้า
    2.4) กรณีที่นอกเหนือจากข้อ 2.2.1 (Special Post) จะเข้ากิจกรรม “Request quotes from Carriers” ซึ่งเป็นแมนนวลทาร์ก และกิจกรรม “Assign Carrier and prepare Paperwork” เพื่อกำหนดข้อมูลการจัดส่ง ได้แก่ ที่อยู่ผู้จัดส่งสินค้า ที่อยู่ผู้รับสินค้า รายละเอียดบริษัทขนส่ง และเลขติดตาม
3) เมื่อเตรียมข้อมูลทุกอย่างเรียบร้อยแล้ว พนักงานโกดังสินค้าทำกิจกรรม “Move package to picking area” ซึ่งเป็นแมนนวลทาร์ก



ภาพที่ 5-1 แบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์

5.3.2 แบบจำลองที่ 2 แบบจำลองการตรวจสอบยอดเงินกู้

แบบจำลองการตรวจสอบยอดเงินกู้เป็นแบบจำลองที่ตรวจสอบยอดเงินกู้เบื้องต้นของสหกรณ์ออมทรัพย์จำลอง ดังภาพที่ 5-2 โดยมีขั้นตอนดังนี้

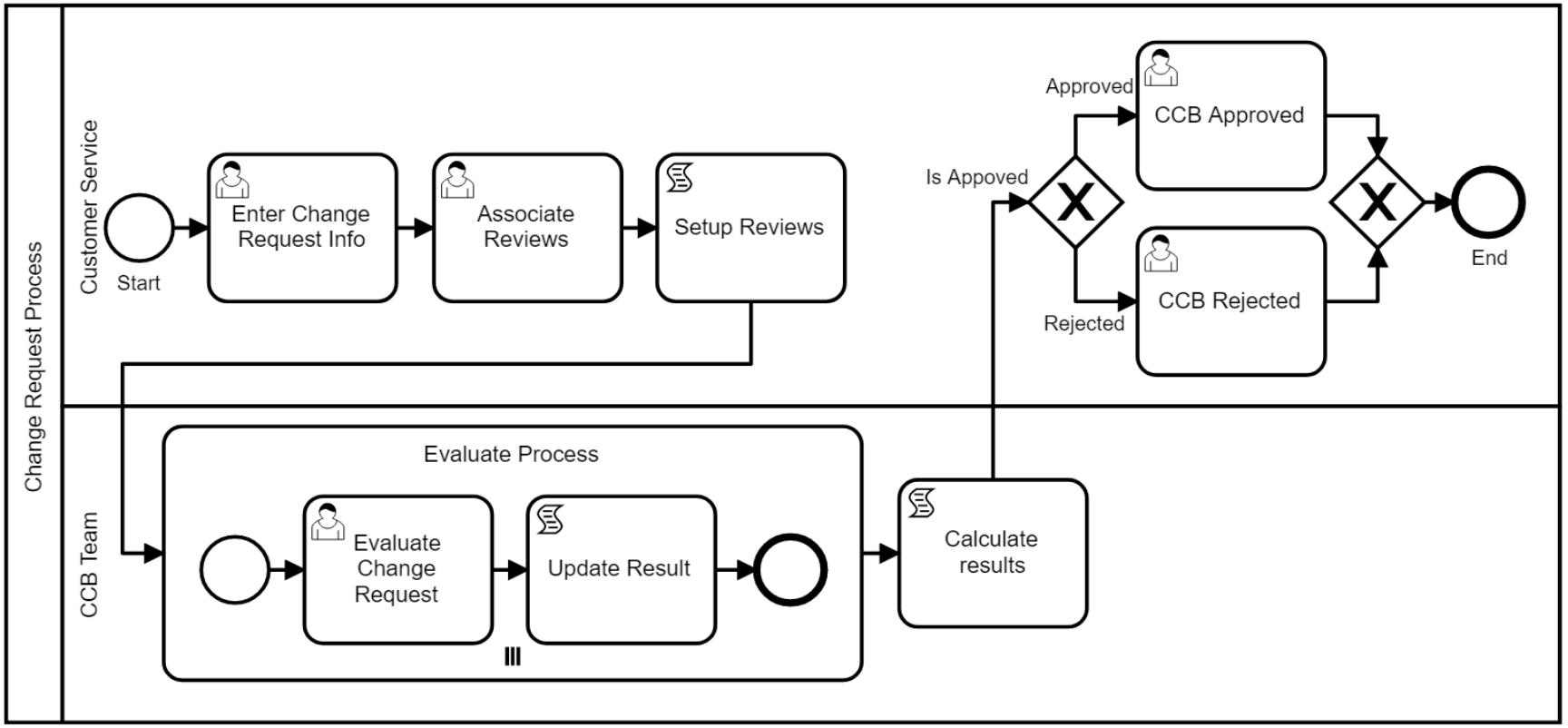


ภาพที่ 5-2 แบบจำลองการตรวจสอบยอดเงินกู้

1) ผู้กู้ต้องกรอกจำนวนเงินกู้ที่กิจกรรม “Enter Loan Amount”
2) กรณีที่ยอดเงินกู้น้อยกว่า หรือเท่ากับ 50,000 ผลลัพธ์ที่ได้ คือ สามารถกู้เงินได้ และจบกิจกรรม “Loan Request Complete”
3) กรณีที่ยอดเงินกู้มากกว่า 50,000 บาท เข้ากิจกรรม “Update Collateral” เพื่อใส่มูลค่าหลักทรัพย์ค้ำประกัน
    3.1) ถ้าหลักทรัพย์ประกันมีมูลค่ามากกว่า 70% ของเงินกู้ทั้งหมด และจบกิจกรรม “Loan Request Complete”
    3.2) ถ้าหลักทรัพย์ค่ำประกันมีมูลค่าน้อยกว่า หรือเท่ากับ 70% ของเงินกู้ทั้งหมด และจบกิจกรรม “Reject Loan Request Check”
    3.3) ถ้าผู้กู้ไม่กรอกมูลค่าหลักทรัพย์ค้ำประกันภายในเวลา 5 นาที จะไปจบที่กิจกรรม “Reject Loan Request Check”

5.3.3 แบบจำลองที่ 3 แบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง

แบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลงเป็นแบบจำลองที่แสดงกระบวนการการอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง โดยคณะกรรมการควบคุมการเปลี่ยนแปลง (Change Control Board) ของบริษัทจำลอง ดังภาพที่ 5-3 ซึ่งมีขั้นตอนการทำงาน ดังนี้



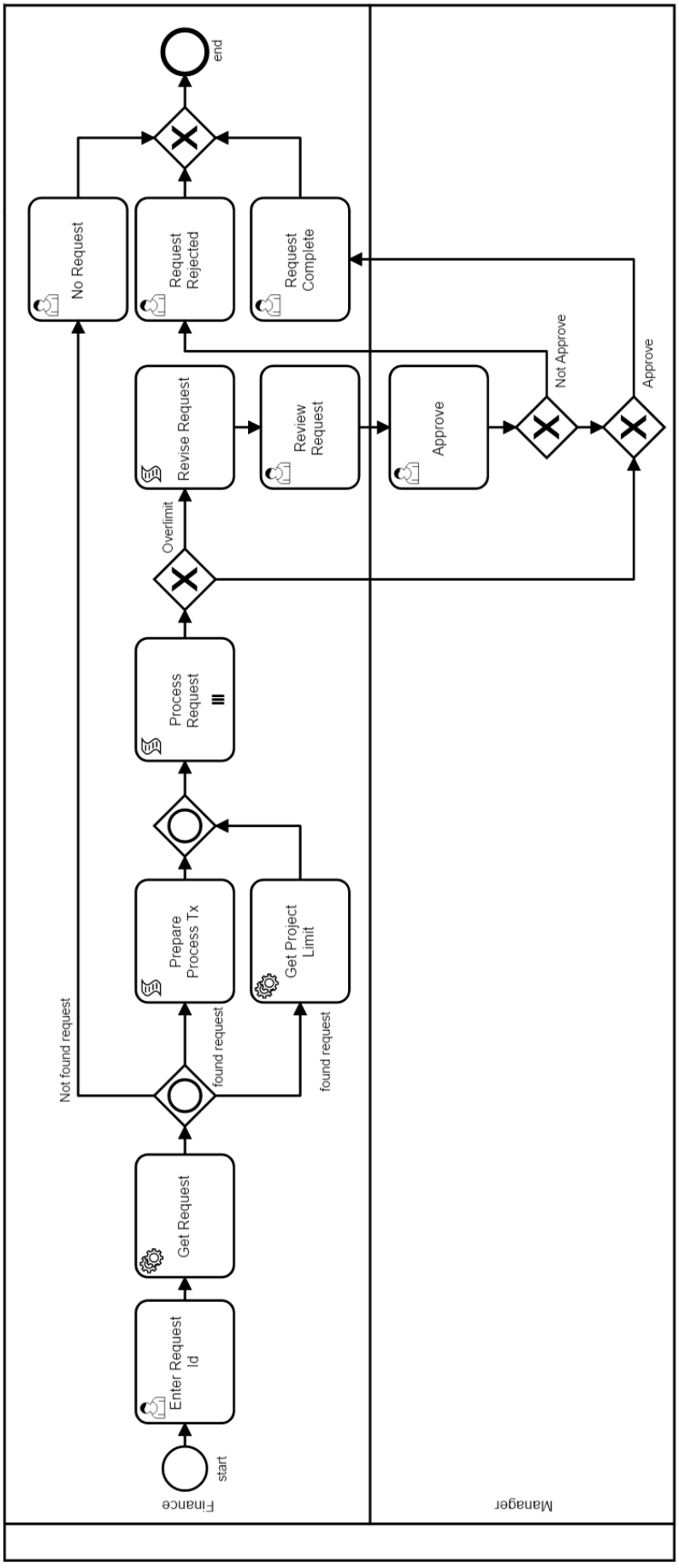
ภาพที่ 5-3 แบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง

1) ฝ่ายบริการลูกค้า(Customer Service) ทำหน้าที่กรอกข้อมูลคำร้องขอการเปลี่ยนแปลง รวมถึงข้อสรุป ผลกระทบที่เกิดขึ้น และจำนวนชั่วโมงแรงงาน (Man-day) ที่กิจกรรม “Enter Change Request Info” จากนั้นกำหนดเงื่อนไขการอนุมัติ ได้แก่ จำนวนผู้เข้าประชุม (RequiredNumOfPeers), จำนวนผู้เข้าประชุมขั้นต่ำ(RequiredMinNumof-Peer) และจำนวนผู้อนุมัติคำร้อง (RequiredNumOfApproval)
2) จากนั้นกิจกรรม “Setup Reviews” ซึ่งเป็นสคริปต์ทาร์ก ทำหน้าที่จัดการประชุมขึ้นมาและสร้างอินสแตนด์ของ “Evaluate Process” ซึ่งเป็นกระบวนการย่อย (Sub Process) เพื่อให้คณะกรรมการควบคุมการเปลี่ยนแปลง (Change Control Board) เข้ามาตรวจสอบในขั้นต่อไป
3) คณะกรรมการควบคุมการเปลี่ยนแปลงเข้ามาตรวจสอบคำร้องในกิจกรรม “Evaluate Change Request” และอนุมัติ หรือไม่อนุมัติคำร้องขอการเปลี่ยนแปลง
4) เมื่อคณะกรรมการควบคุมการเปลี่ยนแปลงลงความเห็นตามจำนวนผู้เข้าประชุมขั้นต่ำเรียบร้อย ถัดไปเป็นหน้าที่ของกิจกรรม “Calculate results” ซึ่งเป็นสคริปต์ทาร์ก ทำหน้าที่สรุปผลการอนุมัติคำร้อง และสรุปผลในกิจกรรม “CCB Approved” ในกรณีที่อนุมัติ และถ้าไม่อนุมัติสรุปผลในกิจกรรม “CCB Rejected”

5.3.4 แบบจำลองที่ 4 แบบจำลองตรวจสอบการเบิกจ่าย

แบบจำลองตรวจสอบการเบิกจ่ายเป็นแบบจำลองที่แสดงกระบวนการตรวจสอบการเบิกจ่ายเงินของแต่ละโครงการของบริษัทจำลอง ดังภาพที่ 5-4 โดยมีขั้นตอนการทำงาน ดังนี้

1) กิจกรรม “Enter Project Id” เป็นกิจกรรมที่ฝ่ายการเงินต้องใส่หมายเลขของโครงการเข้ามาในแบบจำลอง และส่งงานต่อให้กิจกรรม “Get Request” ซึ่งเป็นเซอร์วิสทาร์กที่ทำหน้าที่ติดต่อกับเว็บเซอร์วิส เพื่อนำเข้าข้อมูลร้องขอการเบิกจ่าย
2) ถ้าหากพบข้อมูลร้องขอการเบิกจ่ายจะมีกิจกรรมที่ทำงานขนานได้ ได้แก่
    2.1) กิจกรรม “Prepare Process Tx” ที่ทำหน้าที่เตรียมข้อมูล และ กิจกรรม “Get Project Limit” ที่ทำหน้าที่ดึงข้อมูลวงเงินของแต่ละโครงการ ให้เรียบร้อยก่อนเข้ากิจกรรม “Process Request“
    2.2) ในกรณีที่ไม่มีร้องขอการเบิกจ่าย จบที่กิจกรรม “No Request”

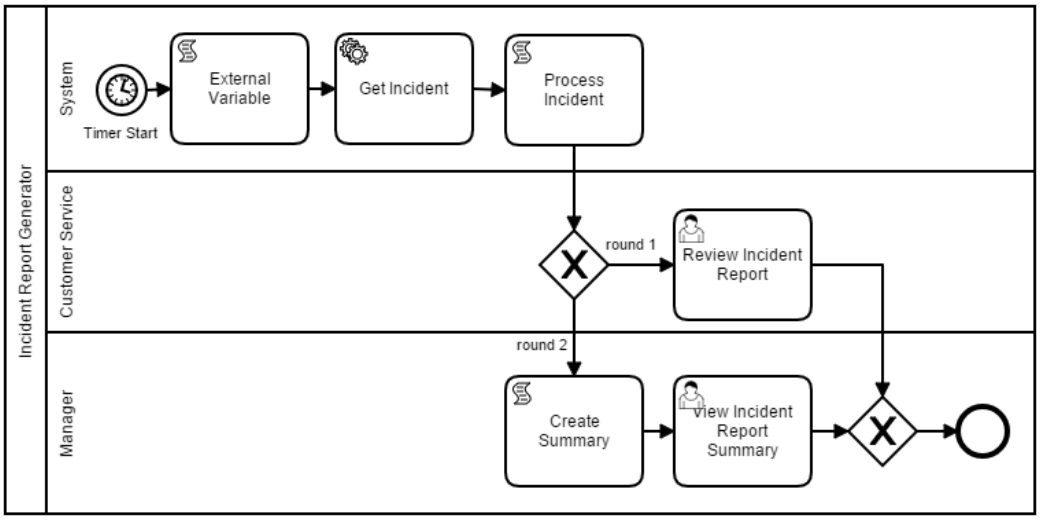


ภาพที่ 5-4 แบบจำลองตรวจสอบการเบิกจ่าย

3) กิจกรรม “Process Request“ เป็นกิจกรรมที่มีพฤติกรรมแบบ Multi-Instance ทำหน้าที่อ่านแต่ละรายการในข้อมูลร้องขอการเบิกจ่าย เพื่อหายอดรวมของสินค้า และบริการ โดยมีเงื่อนไขที่การหยุดการทำงาน (Complete Condition) ดังนี้
   3.1) กิจกรรม “Process Request“ ได้ประมวลผลทุกรายการในข้อมูลร้องขอการเบิกจ่ายครบถ้วน
   3.2) หรือ กิจกรรม “Process Request“ประมวลผลรายการในข้อมูลร้องขอการเบิกจ่าย แล้วพบว่างบประมาณที่ขอเกินจากวงเงินที่กำหนดไว้ของโครงการ
4) ในกรณีที่ประมวลผลรายการในข้อมูลร้องขอการเบิกจ่ายแล้วพบว่างบประมาณที่ขอเกินจากวงเงินที่กำหนดไว้ของโครงการ กิจกรรมถัดไปที่ถูกทำ ได้แก่ “Revise Request” และ “Review Request” เพื่อให้ฝ่ายการเงินรับทราบ และส่งข้อมูลต่อให้ผู้จัดการ เพื่ออนุมัติในกิจกรรม “Approve”
   4.1) ในกรณีที่ผู้จัดการอนุมัติ จะเข้าไปที่กิจกรรม “Request Complete”
   4.2) ในกรณีที่ผู้จัดการไม่อนุมัติ จะเข้าไปที่กิจกรรม “Request Rejected”

5.3.5 แบบจำลองที่ 5 แบบจำลองการสร้างรายงานข้อบกพร่องของระบบ

แบบจำลองการสร้างรายงานข้อบกพร่องของระบบ เป็นแบบจำลองที่ทำหน้าที่สร้างรายงานข้อบกพร่องของระบบ เริ่มต้นจาก Timer Start Event ซึ่งถูกกำหนดการทำงานอยู่ในรูปแบบ R2/PT1M ซึ่งสามารถตีความได้ว่ากระบวนการมีการทำงานทั้งหมด 2 รอบ แต่ละรอบมีระยะเวลาห่างกัน 1 นาที ดังภาพที่ 5-5

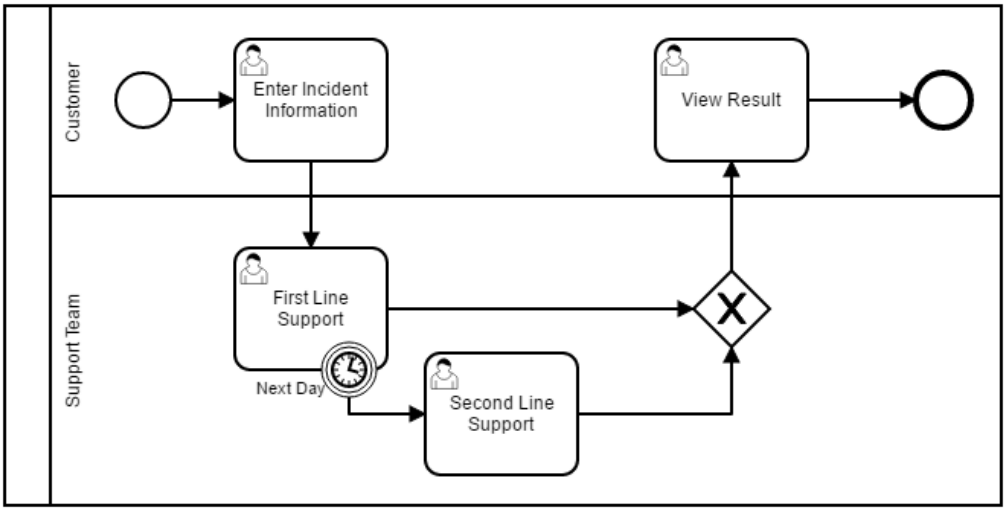


ภาพที่ 5-5 แบบจำลองการสร้างรายงานข้อบกพร่องของระบบ

1) การทำงานรอบที่ 1 กิจกรรม “Get Incident” และ “Process Incident” ทำงาน โดยดึงข้อมูลข้อบกพร่องของระบบให้กับฝ่ายบริการลูกค้า เพื่อตรวจสอบในกิจกรรม “Review Incident Report”
2) การทำงานรอบที่ 2 กิจกรรม “Get Incident” และ “Process Incident” ทำงาน โดยดึงข้อมูลข้อบกพร่องของระบบให้กับผู้จัดการ เพื่อตรวจสอบในกิจกรรม “View Incident Report Summary” โดยสรุปข้อมูลข้อบกพร่องของระบบในกิจกรรม “Create Summary” ให้เรียบร้อย

5.3.6 แบบจำลองที่ 6 แบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด

แบบจำลองการจัดการคำร้องขอการตรวจสอบช้อผิดพลาด เป็นแบบจำลองที่แสดงถึงลำดับ ขั้นตอนในกรณีที่เกิดข้อผิดพลาดจากตัวซอฟต์แวร์หลังการใช้งานบนสภาพแวดล้อมจริง (Go-Live) ดังภาพที่ 5-6



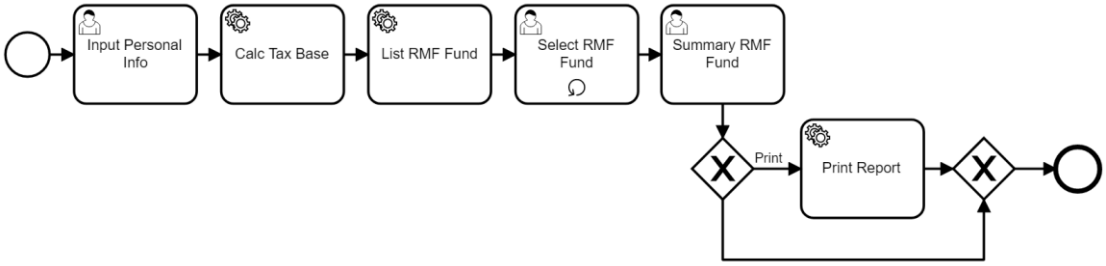
ภาพที่ 5-6 แบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด

1) เมื่อเกิดข้อผิดพลาดในระบบ ลูกค้าสามารถสร้างคำร้องขอการตรวจสอบข้อผิดพลาดที่กิจกรรม “Enter Incident Information” โดยกรอกข้อมูลที่จำเป็น ได้แก่ หัวข้อคำร้องขอการตรวจสอบช้อผิดพลาด และรายละเอียดของข้อผิดพลาด
2) หลังจากนั้นฝ่ายบริการลูกค้า (Support Team) ตรวจสอบคำร้องขอการตรวจสอบที่กิจกรรม “First Line Support” และคำแนะนำกับลูกค้า ในกรณีที่กิจกรรม First Line Support”ไม่ได้ถูกตรวจสอบภายใน 1 วันงานจะถูกส่งต่อไปให้กิจกรรม Second Line Support เพื่อทำหน้าที่รับช่วงการตรวจสอบต่อ

3) หลังจากฝ่ายบริการลูกค้าได้ตรวจสอบ และให้คำแนะนำเบื้องต้นแล้ว กิจกรรมถัดมาเป็นกิจกรรม “View Result” เพื่อสรุปข้อมูลให้ลูกค้ารับทราบ

5.3.7 แบบจำลองที่ 7 แบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ

แบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ เป็นแบบจำลองที่จำลองกระบวนการการแนะนำการเลือกกองทุนอาร์เอ็มเอฟ ดังภาพที่ 5-7

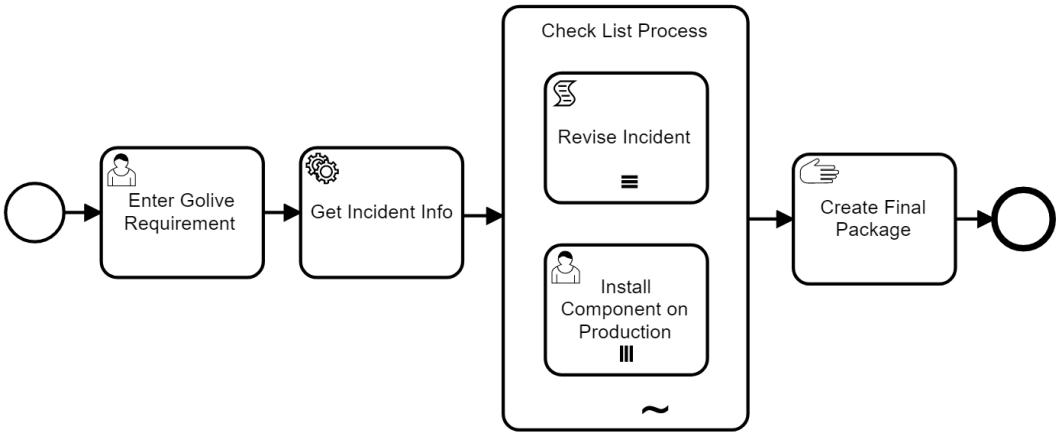


ภาพที่ 5-7 แบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ

1) ผู้ใช้งานกรอกข้อมูลส่วนตัวที่กิจกรรม “Input Personal Info” เมื่อกรอกข้อมูลเสร็จเรียบร้อยแล้วจะเข้าสู่การทำงานของเซอร์วิสทาสก์ “Calc Tax Base” เพื่อคำนวณฐานภาษี จากนั้นเข้าสู่กิจกรรม “List RMF Fund” ซึ่งเป็นเซอร์วิสทาร์กทำหน้าที่ดึงข้อมูลกองทุน
2) จากนั้นเป็นกิจกรรม “Select RMF Fund” ที่ให้ผู้ใช้งานวนเลือกกองทุนได้มากที่สุด 5 กองทุน โดยเงื่อนไขการออกทำงาน คือ มูลค่าที่ซื้อกองทุนนั้นต้องไม่เกิน 15% ของรายได้ที่ต้องเสียภาษีต่อปี และต้องไม่เกิน 500,000 บาทเมื่อรวมกับกองทุนสำรองเลี้ยงชีพ กบข. ประกันชีวิตแบบบำนาญ และกองทุนสงเคราะห์ครูโรงเรียนเอกชน หรือผู้ใช้งานระบุว่าเป็นรายการสุดท้ายแล้ว หลังจากนั้นเข้าสู่กิจกรรม “Summary RMF Fund” แสดงข้อมูลสรุป
3) หลังจากได้ตรวจสอบข้อมูลสรุปแล้ว ถ้าผู้ใช้งานเลือกพิมพ์รายงานจะเข้าสู่กิจกรรม “Print Report” ซึ่งถ้าไม่เลือกพิมพ์รายงานจะเป็นการจบกระบวนการ

5.3.8 แบบจำลองที่ 8 แบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบ

แบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบเป็นกระบวนการที่ถูกใช้งานในบริษัทจำลอง เพื่อใช้ระหว่างช่วงการบำรุงรักษาซอฟต์แวร์ ก่อนนำซอฟต์แวร์ที่ปรับปรุงแก้ไขข้อผิดพลาดเรียบร้อยแล้วติดตั้ง และใช้งานจริง ดังภาพที่ 5-8



ภาพที่ 5-8 แบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบ

1) ผู้ใช้งานกรอกข้อมูลที่กิจกรรม “Enter Golive Requirement” ได้แก่ AcceptNumOfOpenIssue (จำนวนข้อผิดพลาดที่ยอมรับได้) และ ClientCount (จำนวนเครื่องที่ต้องติดตั้งรันไทม์(Runtime) เพิ่มเติม เมื่อกรอกข้อมูลเสร็จกิจกรรม “Get Incident Info” ซึ่งเป็นเซอร์วิสทาร์กทำหน้าที่ติดต่อกับเว็บเซอร์วิส เพื่อดึงข้อมูลผิดพลาดในระบบออกมา
2) จากนั้นเข้าสู่กิจกรรม “Check List Process” ที่มีรูปแบบการทำงานแบบ Adhoc-Sub Process โดยมีกิจกรรมย่อยภายใน ได้แก่ กิจกรรม “Revise Incident” และ Install Component on Production”
3) กิจกรรม “Revise Incident” ซึ่งเป็นสคริป์ทาร์กทำหน้าที่นับจำนวนข้อผิดพลาดที่มีสถานะเปิด (Open) โดยแยกตามงานในส่วนของการบำรุงรักษาลงในตัวแปร OpenMAIncidentCount และงานของโครงการลงในตัวแปร OpenProjectIncidentCount ซึ่งมีลักษณะการทำงานแบบ Multi-Instance โดยเงื่อนไขการสิ้นสุดกิจกรรม(Completion Condition) คือ การตรวจสอบข้อมูลผิดพลาดครบถ้วน
4) กิจกรรม “Install Component on Production” ซึ่งเป็น User Task โดยทีมติดตั้งขององค์กรต้องลงติดตั้งรันไทม์ (Runtime) ให้กับเครื่องคอมพิวเตอร์ตามจำนวนที่กำหนด
5) เงื่อนไขการสิ้นสุดของกิจกรรม “Check List Process” คือ จะต้องมีจำนวนข้อผิดพลาดของช่วงการบำรุงรักษา รวมกับงานของโครงการ ต้องน้อยกว่าหรือเท่ากับจำนวนข้อผิดพลาดที่ยอมรับได้ และเครื่องคอมพิวเตอร์ต้องมีการติดตั้งรันไทม์ได้ครบถ้วน

5.4 ผลการทดสอบเครื่องมือ

จากแบบจำลองทั้ง 8 แบบจำลองข้างต้น ผู้ทดสอบได้ใช้เครื่องมือสร้างมิวแตนท์ขึ้น และทดสอบด้วยกรณีทดสอบที่เตรียมไว้ พบว่าเครื่องมือสามารถสร้างมิวแตนท์ขึ้นมาได้ครบถ้วนทั้ง 25 ตัว ดำเนินการ โดยผลการทดสอบแบบจำลองจากการทดสอบวีคมิวเทชันแบบ ST-WEAK/1 , BB-WEAK/1 และ BB-WEAK/N ได้แสดงดังตารางที่ 5-1, 5-2 และ 5-3 ตามลำดับ

ตารางที่ 5-1 ผลการทดสอบแบบจำลองจากวิธีการทดสอบมิวเทชันแบบ ST-WEAK/1

แบบจำลอง	จำนวนมิวแตนท์ที่ถูกสร้าง	จำนวนมิวแตนท์ที่ถูกฆ่า	จำนวนมิวแตนท์ที่มีชีวิต	เวลาที่ใช้ทดสอบ (วินาที)	คะแนนมิวเทชัน	ประสิทธิภาพของการทดสอบ
แบบจำลองที่ 1	33	33	0	109	100.00%	66.67%
แบบจำลองที่ 2	30	28	2	8790	93.00%	48.50%
แบบจำลองที่ 3	37	33	4	196	89.00%	53.40%
แบบจำลองที่ 4	92	92	0	477	100.00%	75.00%
แบบจำลองที่ 5	10	10	0	1638	100.00%	100.00%
แบบจำลองที่ 6	1	1	0	86366	100.00%	50.00%
แบบจำลองที่ 7	84	-	-	-	-	-
แบบจำลองที่ 8	38	-	-	-	-	-

ตารางที่ 5-2 ผลการทดสอบแบบจำลองจากวิธีการทดสอบมิวเทชันแบบ BB-WEAK/1

แบบจำลอง	จำนวนมิวแตนท์ที่ถูกสร้าง	จำนวนมิวแตนท์ที่ถูกฆ่า	จำนวนมิวแตนท์ที่มีชีวิต	เวลาที่ใช้ทดสอบ (วินาที)	คะแนนมิวเทชัน	ประสิทธิภาพของการทดสอบ
แบบจำลองที่ 1	33	33	0	104	100.00%	66.67%
แบบจำลองที่ 2	30	28	2	8857	93.00%	48.50%
แบบจำลองที่ 3	40	36	4	220	90.00%	72.00%
แบบจำลองที่ 4	95	94	1	478	99.00%	74.25%
แบบจำลองที่ 5	16	16	0	1768	100.00%	100.00%
แบบจำลองที่ 6	1	1	0	86623	100.00%	50.00%
แบบจำลองที่ 7	88	-	-	-	-	-
แบบจำลองที่ 8	47	-	-	-	-	-

ตารางที่ 5-3 ผลการทดสอบแบบจำลองจากวิธีการทดสอบมิวเทชันแบบ BB-WEAK/N

แบบจำลอง	จำนวนมิวแตนท์ที่ถูกสร้าง	จำนวนมิวแตนท์ที่ถูกฆ่า	จำนวนมิวแตนท์ที่มีชีวิต	เวลาที่ใช้ทดสอบ (วินาที)	คะแนนมิวเทชัน	ประสิทธิภาพของการทดสอบ
แบบจำลองที่ 1	33	33	0	115	100.0%	66.67%
แบบจำลองที่ 2	30	28	2	8855	93.00%	48.50%
แบบจำลองที่ 3	40	36	4	220	90.00%	72.00%
แบบจำลองที่ 4	95	94	1	478	99.00%	74.25%
แบบจำลองที่ 5	16	16	0	1799	100.00%	100.00%
แบบจำลองที่ 6	1	1	0	86630	100.00%	50.00%
แบบจำลองที่ 7	88	-	-	-	-	
แบบจำลองที่ 8	47	-	-	-	-	

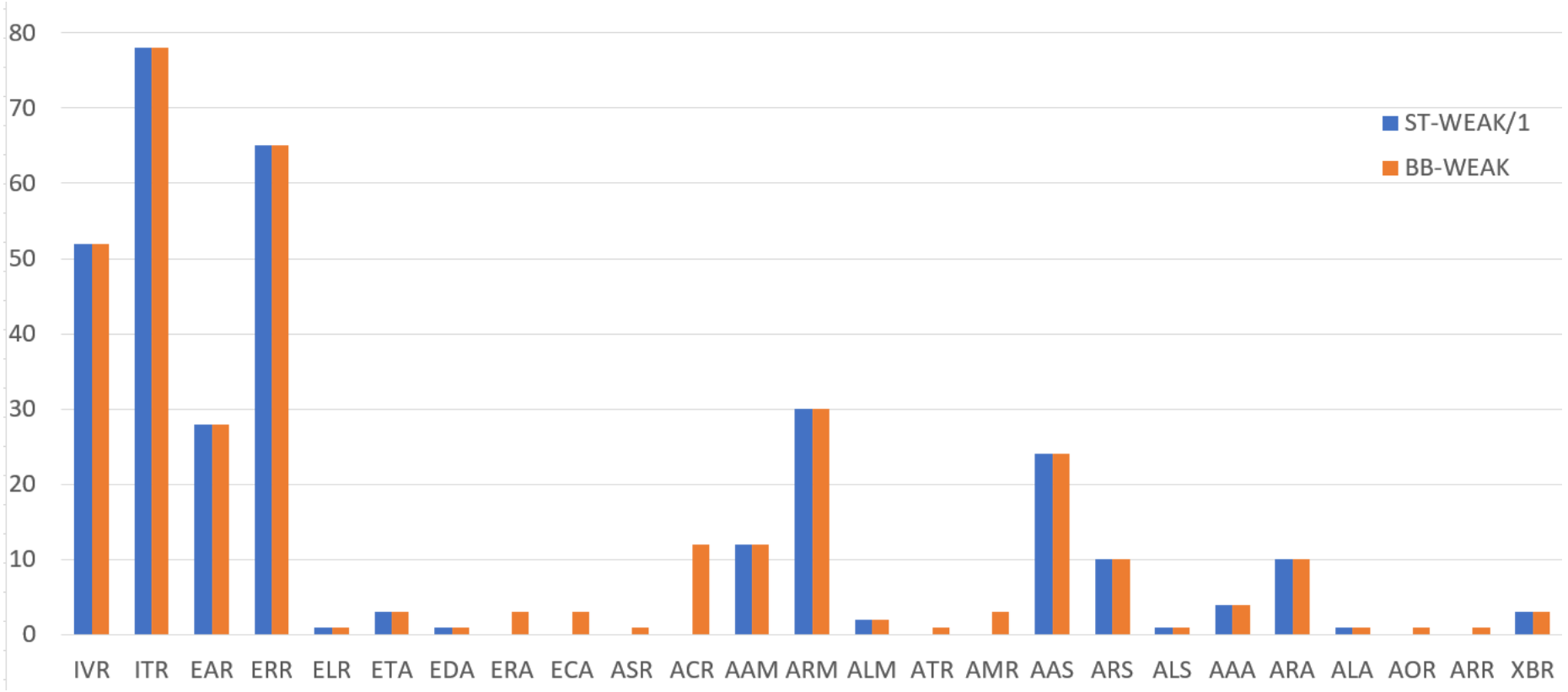
หมายเหตุ

1. ตารางที่ 5-1 ถึง 5-3 ในแบบจำลองที่ 7 และ 8 จะไม่มีผลการทดสอบแบบจำลอง เนื่องจากเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นไม่รองรับการทดสอบมิวแตนท์
2. กรณีทดสอบของแบบจำลองสามารถดูรายละเอียดเพิ่มเติมได้ที่ภาคผนวก ค
3. ผลการสร้างมิวแตนท์แยกตามตัวดำเนินการมิวเทชันสามารถดูรายละเอียดเพิ่มเติมได้ที่ภาคผนวก ง

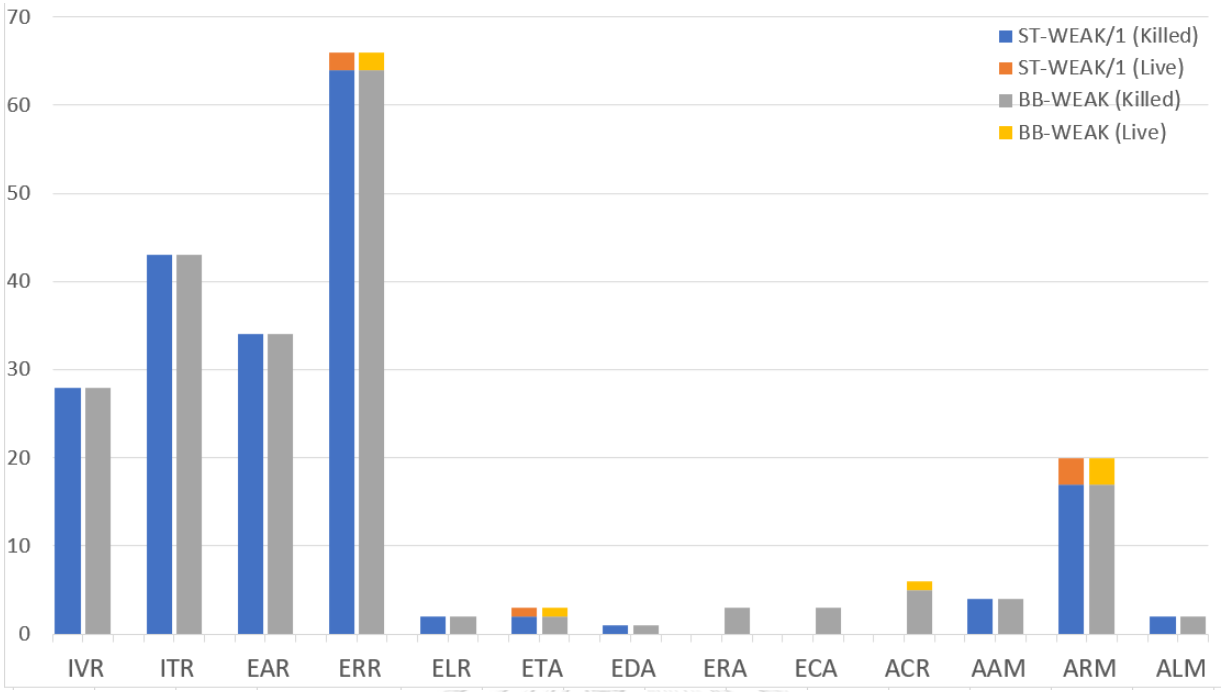
เมื่อนำข้อมูลในตารางที่ 5-1 ถึง 5-3 มาตรวจสอบจะพบว่าในส่วนของการทดสอบแบบ BB-WEAK/1 และ BB-WEAK/N ได้ผลการทดสอบที่เหมือนกัน เนื่องจากมิวแตนท์ถูกฆ่าในรอบแรกของการทดสอบ จึงขอสรุปผลการทดลองรวมกันในกลุ่ม BB-WEAK

จากภาพที่ 5-9 เมื่อแจกแจงการสร้างมิวแตนท์แยกตามตัวดำเนินการมิวเทชันพบว่าตัวดำเนินการมิวเทชัน ITR ที่เน้นการเปลี่ยนชนิดของตัวแปรนั้นสามารถสร้างมิวแตนท์ได้มากที่สุด ถัดมาเป็นตัวดำเนินการมิวเทชัน ERR และ ARM ที่เน้นการเปลี่ยนตัวดำเนินการเชิงสัมพันธ์ และ ตัวดำเนินการมิวเทชัน IVR ที่เน้นการเปลี่ยนชนิดตัวแปรที่มีชนิดเดียวกันเป็นลำดับถัดมา

เมื่อตรวจสอบผลการทดสอบ โดยมีตัวดำเนินการมิวเทชันที่สามารถทดสอบเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นทั้งหมด 13 ตัวดำเนินการ ได้แก่ IVR, ITR, EAR, ERR, ELR, ETA, EDA, ERA, ECA, ACR, AAM, ARM และ ALM พบว่ามิวแตนท์ที่เกิดจากตัวดำเนินการมิวเทชัน ERR, ITR, IVR และ EAR มีโอกาสถูกฆ่าได้ด้วยกรณีทดสอบมากที่สุดดังภาพที่ 5-10



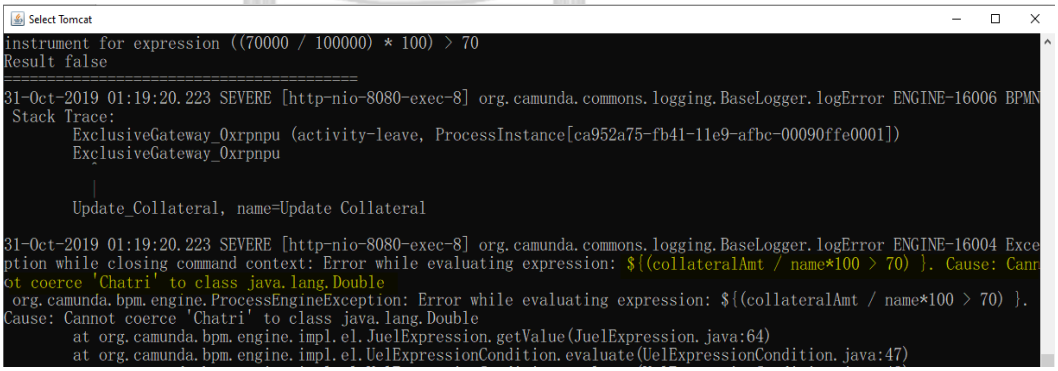
ภาพที่ 5-9 กราฟแท่งแสดงจำนวนมิวแตนท์แยกตามตัวดำเนินการ และประเภทการทดสอบวีคมิวเทชัน



ภาพที่ 5-10 กราฟแท่งแสดงจำนวนมิวแตนท์ที่ถูกทดสอบตามตัวดำเนินการ และสถานะของมิวแตนท์

หากตรวจสอบผลลัพธ์ของทดสอบพบว่าตัวดำเนินการมิวเทชันควรมีการพิจารณาถึงการนำมาใช้งาน ได้แก่

1) ตัวดำเนินการมิวเทชัน ITR ซึ่งเป็นเปลี่ยนตัวแปรที่มีชนิดต่างกัน เมื่อนำมาทดสอบกับเครื่องประมวลผลแบบจำลองได้เกิดข้อผิดพลาดขึ้น ดังภาพที่ 5-11

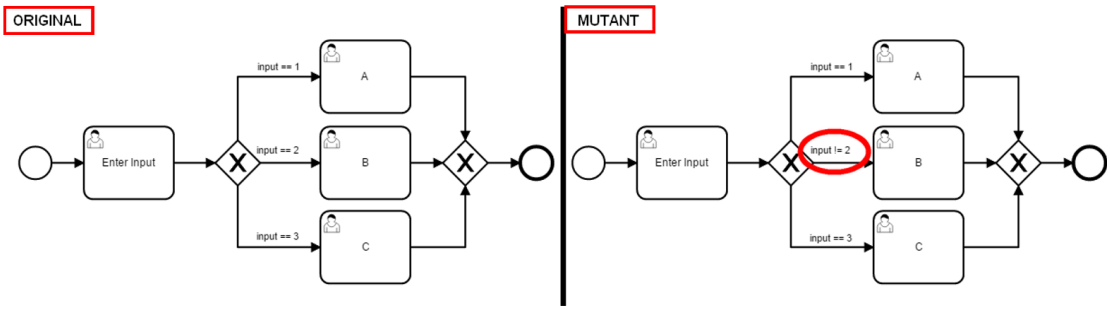


ภาพที่ 5-11 ข้อผิดพลาดจากเครื่องประมวลผลแบบจำลอง
เมื่อมีการประมวลผลนิพจน์ที่มีตัวแปรต่างชนิด

จากภาพที่ 5-11 มิวแตนท์ ${collateralAmt / name * 100 > 70)} โดยดัดแปลงจากต้นฉบับ ${(collateralAmt / loanAmt*100 > 70) } เมื่อนำมาประมวลผลผ่านเครื่องประมวลผลแบบจำลองพบว่าเกิดข้อผิดพลาดขึ้น เนื่องจากตัวแปร loanAmt มี

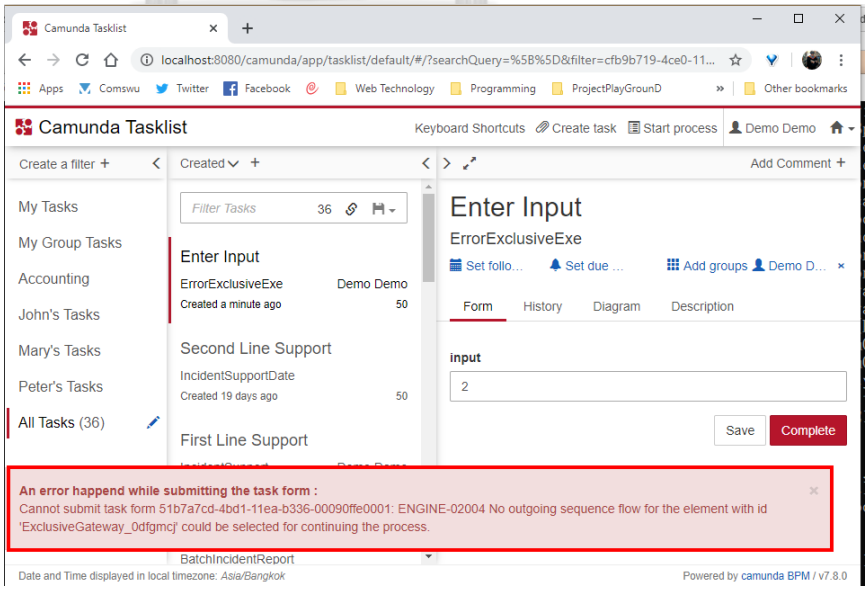
การเก็บข้อมูลประเภทตัวเลข แต่ตัวแปร name มีการเก็บข้อมูลประเภทสตริง เมื่อนำสตริงเข้ามาทำกระบวนการทางคณิตศาสตร์กับตัวเลข เช่น คูณ หรือ หาร จึงได้เกิดข้อผิดพลาดขึ้น

2) ตัวดำเนินการมิวเทชัน EAR, ELA และ ERA มีข้อจำกัดกรณีที่รูปแบบของแบบจำลองนั้นมีการกำหนดเงื่อนไขในทุกเส้นทางที่ออกจากสัญลักษณ์ Gateway และถ้านำแบบจำลองไปสร้างแบบจำลองมิวแตนท์ เมื่อเครื่องประมวลผลแบบจำลองตีความเงื่อนไขข้อผิดพลาด เพราะไม่มีเส้นทางที่สามารถทำให้จบกระบวนการได้ ดังรูปที่ 5-12



ภาพที่ 5-12 แบบจำลองต้นฉบับ (ซ้าย) เปรียบแบบจำลองมิวแตนท์ (ขวา) ที่มีการใช้งาน Exclusive Gateway แล้วไม่มีเส้นทางที่สามารถทำให้จบกระบวนการได้

จากภาพที่ 5-12 ในกรณีที่ตัวแปร INPUT มีค่าเท่ากับ 2 ในแบบจำลองมิวแตนท์ เมื่อเครื่องประมวลผลแบบจำลองการประมวลผลเงื่อนไขที่ Exclusive Gateway จะพบข้อผิดพลาดดังภาพที่ 5-13



ภาพที่ 5-13 ข้อผิดพลาดจากเครื่องประมวลผลแบบจำลองแล้วไม่มีเส้นทางที่สามารถทำให้จบกระบวนการได้

3) ตัวดำเนินการมิวเทชัน EDA, ETA ซึ่งมีความเกี่ยวข้องกับเวลามีความสิ้นเปลืองเวลาในการทดสอบมากที่สุดเนื่องจากต้องรอเวลาจริง ถึงตรวจสอบกิจกรรมถัดไปตามที่ระบุไว้ในกรณีทดสอบได้

5.5 สรุปผลการทดสอบ

จากการทดสอบเครื่องมือกับแบบจำลองตัวอย่างทั้ง 8 แบบจำลองตามที่ได้แสดงสรุปผลการทดสอบดังตารางที่ 5-1 ถึง 5-3 ได้แสดงให้เห็นว่าเครื่องมือทดสอบแบบจำลองบีพีเอ็มเอ็นด้วยวีคมิวเทชันสามารถสร้างมิวแตนท์ของแต่ละตัวดำเนินการมิวเทชันได้ครบถ้วนทั้ง 25 ตัวดำเนินการ และทดสอบมิวแตนท์กับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นที่เป็นโอเพนซอร์สได้ทั้งหมด 13 ตัวดำเนินการ ได้แก่ IVR, ITR, EAR, ERR, ELR, ETA, EDA, ERA, ECA, ACR, AAM, ARM และ ALM อย่างไรก็ตามเมื่อนำผลการทดสอบมาวิเคราะห์พบว่าตำดำเนินการมิวเทชันมิวเทชันบางตัวควรพิจารณาถึงการนำมาใช้งาน ได้แก่ ตัวดำเนินการ ITR เนื่องจากพบว่ามิวแตนท์ที่สร้างขึ้นไม่สามารถทำงานได้จริงบนเครื่องประมวลผลแบบจำลอง ตัวดำเนินการมิวเทชัน EAR, ELA และ ERR ซึ่งต้องมีการพิจารณาการใช้งานกับสัญลักษณ์ Exclusive Gateway เพื่อป้องกันปัญหาไม่มีเส้นทางที่สามารถทำให้จบกระบวนการ และตัวดำเนินการมิวเทชัน EDA, ETA และ ECA ที่ยังมีข้อจำกัดของการทดสอบอยู่ เพราะกินเวลาในการทดสอบมาก

บทที่ 6
ผลสรุปการวิจัยและข้อเสอแนะ

6.1 สรุปผลการวิจัย

งานวิจัยนี้เริ่มต้นจากการนำตัวดำเนินการมิวเทชันของแบบจำลองบีพีเอ็นเอ็นทั้งหมด 25 ตัว ดำเนินการมาวิเคราะห์ เสนอการจัดตัวดำเนินการมิวเทชันสำหรับการทดสอบวีคมิวเทชัน และพัฒนาเครื่องมือที่ใช้สร้าง และทดสอบมิวแตนท์ได้อย่างอัตโนมัติกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น

เพื่อที่จะทดสอบเครื่องมือสำหรับงานวิจัยนี้ ได้มีการนำแบบจำลองทั้งหมด 8 แบบจำลองพบว่าเครื่องมือสามารถสร้างมิวแตนท์จากตัวดำเนินการมิวเทชันทั้งหมด 25 ตัวได้ และทดสอบมิวแตนท์ได้อย่างอัตโนมัติกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นที่เป็นโอเพนซอร์สได้ทั้งหมด 13 ตัวดำเนินการ ได้แก่ IVR, ITR, EAR, ERR, ELR, ETA, EDA, ERA, ECA, ACR, AAM, ARM และ ALM ทำให้ทราบถึงประสิทธิภาพของตัวดำเนินการมิวเทชันแต่ละตัว อาทิ เช่น ตัวดำเนินการมิวเทชัน ITR ที่ทางผู้วิจัยมีความเห็นว่าไม่ควรนำตัวดำเนินการมิวเทชันนี้ไปใช้สำหรับการสร้างมิวแตนท์ เนื่องจากแบบจำลองที่ได้เมื่อนำไปทดสอบกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น พบว่าตัวมิวแตนท์ที่สร้างไม่สามารถทำงานได้จริง รวมถึงตัวดำเนินการมิวเทชันที่เกี่ยวกับเวลาได้แก่ EDA และ ETA ที่ยังมีข้อจำกัดของการทดสอบอยู่ และตัวดำเนินการมิวเทชันที่สามารถสร้างมิวแตนท์ได้มากที่สุดเป็นตัวดำเนินการมิวเทชันในกลุ่มที่เน้นการเปลี่ยนตัวดำเนินการเชิงสัมพันธ์ ตัวดำเนินการทางคณิตศาสตร์ และ ตัวดำเนินการที่เน้นการเปลี่ยนตัวแปรที่มีชนิดเดียวกัน

6.2 ข้อจำกัดงานวิจัย

1) วิทยานิพนธ์นี้ได้ทดสอบตัวดำเนินการมิวเทชันที่เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นคามุนดารองรับเท่านั้น
2) เครื่องมือที่ใช้ทดสอบมีข้อจำกัดเกี่ยวกับตัวดำเนินการมิวเทชันที่เกี่ยวกับวันที่ และเวลา ซึ่งต้องรอการทำงานจริง ไม่สามารถแก้ไขเวลาของสภาพแวดล้อมที่ใช้ทดสอบได้
3) เครื่องมือที่ใช้ทดสอบวีคมิวเทชันนี้ไม่สามารถสร้างกรณีทดสอบได้อย่างอัตโนมัติ
4) มิวแตนท์ที่เกิดจากตัวดำเนินการมิวเทชัน EAR, ERR และ ELR เมื่อนำไปใช้กับสัญลักษณ์ Exclusive Gateway มีบางกรณีที่ไม่มีเส้นทางที่สามารถทำให้จบกระบวนการได้
5) มิวแตนท์ที่เกิดจากตัวดำเนินการมิวเทชัน ITR ไม่สามารถดำเนินการบนเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็นได้ เนื่องจากชนิดข้อมูลของตัวแปรต่างชนิดกัน

6.3 แนวทางการพัฒนาต่อ

1) พัฒนาเครื่องมือสร้างมิวแตนท์ที่รองรับการทดสอบมิวเทชันรูปแบบอื่นๆ
2) พัฒนาเครื่องมือที่สามารถเชื่อมต่อกับเครื่องประมวลผลแบบจำลองบีพีเอ็นเอ็นของทีมพัฒนาค่ายต่างๆได้
3) ประเมินประสิทธิผลของตัวดำเนินการมิวเทชันที่ยังไม่ได้ทดสอบในงานวิจัยนี้ ได้แก่ ASR, ATR, AMR, AAS, ARS, ALS, AAA, ARA, ALA, AOR, ARR และ XBR
4) นิยามตัวดำเนินการมิวเทชันเพิ่มเติม โดยดูสัญลักษณ์ของแบบจำลองบีพีเอ็มเอ็นที่ยังไม่ถูกนำมาประยุกต์ใช้งาน เช่น compensation, signal event, condition event เป็นต้น
5) พัฒนาเครื่องมือให้รองรับแบบจำลองบีพีเอ็มเอ็นที่มีความเกี่ยวโยงจากสัญลักษณ์ Call Activity
6) พัฒนาเครื่องมือให้สามารถเรียนรู้ และลดการสร้างมิวแตนท์ที่ไม่จำเป็นได้

บรรณานุกรม

1. OMG. *Business Process Model and Notation V.2.0.2 [Online]*. 2014; Available from: https://www.omg.org/spec/BPMN/2.0.2/ [2020, May 15].
2. ISO/IEC, *ISO/IEC 19510:2013 – Information technology - Object Management Group*

Business Process Model and Notation. v2.0.2.[2020, March 15] ed. 2013: ISO.

3. Mike Papadakis, M.K., Jie Zhang, Yue Jia, Yves Le Traon and Mark Harman, *Advances in Computers*, in *Mutation Testing Advances: An Analysis and Survey*. 2017.
4. Offutt, A.J. and S.D. Lee, *An empirical evaluation of weak mutation.* IEEE Transactions on Software Engineering, 1994. **20**(5): p. 337-344.
5. Tadeesom, P.P. and T. Suwannasart. *Mutation Operators in BPMN Model*. in *ICIDE 2017*. 2017.
6. King, K.N. and A.J. Offutt, *A Fortran language system for mutation-based software testing.* Softw. Pract. Exper., 1991. **21**(7): p. 685-718.
7. Agrawal, H.D., Richard & Hathaway, Bob & Hsu, William & Hsu, Wynne & Krauser, and R.M. E.W. & J. Martin, Aditya & Spafford, Eugene, *Design Of Mutant Operators For The C Programming Language.* 1999.
8. Delamaro M. E., M.J.C. *Proteum - A Tool for the Assessment of Test Adequacy for C Programs*. in *In: Proc. of the Conf. on Performability in Computing Systems (PCS 96)*. 1996.
9. Tuya, J., M.J. Suarez-Cabal, and C.d.l. Riva, *Mutating database queries.* Inf. Softw. Technol., 2007. **49**(4): p. 398-417.
10. Tuya, J., M.J. Suarez-Cabal, and C.d.l. Riva. *SQLMutation: A tool to generate mutants of SQL database queries*. in *Second Workshop on Mutation Analysis (Mutation 2006 - ISSRE Workshops 2006)*. 2006.
11. Derezinska, A. *An experimental case study to applying mutation analysis for SQL queries*. in *2009 International Multiconference on Computer Science and Information Technology*. 2009.

12. Yu-Seung, M., K. Yong-Rae, and J. Offutt. *Inter-class mutation operators for Java*. in *13th International Symposium on Software Reliability Engineering, 2002. Proceedings.* 2002.
13. Ma, Y.-S., J. Offutt, and Y.R. Kwon, *MuJava: an automated class mutation system: Research Articles.* Softw. Test. Verif. Reliab., 2005. **15**(2): p. 97-133.
14. Derezinska, A. *Quality Assessment of Mutation Operators Dedicated for C# Programs*. in *2006 Sixth International Conference on Quality Software (QSIC'06)*. 2006.
15. Szustek, A.D.A. *CREAM - a system for object-oriented mutation of C#*. in *red. S.Szczepanski, M. Klosowski, Z. Felendzer*. 2007. Annals Gdansk University of Technology Faculty of ETI.
16. Derezinska, A. and K. Hałas. *Analysis of Mutation Operators for the Python Language*. 2014. Cham: Springer International Publishing.
17. halas, k. *MutPy, mutation testing tool for Python 3.x*. [Source Code] 2014; Available from: https://github.com/mutpy/mutpy [2020, May, 23].
18. A. Estero-Botaro, F.P.-L., and I. Medina-Bulo. *Mutation operators for WS-BPEL 2.0.* in *ICSSEA 2008: 21th International Conference on Software & Systems Engineering and their Applications*. 2008. . Paris, France.
19. Domínguez-Jiménez, J.J., et al. *GAmera: An Automatic Mutant Generation System for WS-BPEL Compositions*. in *2009 Seventh IEEE European Conference on Web Services*. 2009.
20. Boonyakulsrirung, P. and T. Suwannasart. *WeMuTe–a weak mutation testing tool for WS-BPEL*. in *Proceedings of The International MultiConference of Engineers and Computer Scientists (IMECS'12)*. 2012.
21. OMG. *Case Management Model and Notation Standard*. 2014; First:[Available from: https://www.omg.org/spec/CMMN/About-CMMN/ [2020, May 20].
22. Yergeau, F., et al. *Extensible Markup Language (XML) 1.0*. W3C Recommendation 2013 2018, February 19 [cited 2018, February 19; Fifth:[Available from: http://www.w3.org/TR/REC-xml/ [2020, May 19].
23. *World Wide Web Consortium (w3c)*. Available from: http://www.w3.org/ [2018, February 19].

24. Connolly, D. *Overview of SGML resources*. World Wide Web Consortium 1995; Available from: http:/www.w3.org/MarkUp/SGMLS [2018, February 19].

25. Völzer, H. *An overview of BPMN 2.0 and its potential use*. in *International Workshop on Business Process Modeling Notation*. 2010. Germany: Springer.

26. Silver, B., *BPMN Method and Style, with BPMN Implementer's Guide: A structured approach for business process modeling and implementation using BPMN 2.0*. 2011: Cody-Cassidy Press Aptos.

27. Polančič, G. *Supporting Core BPM Principles With BPMN 2.0 – 4th Principle: "Information Technology Is An Essential Enabler For BPM*. Available from: http://blog.goodelearning.com/bpmn/supporting-core-bpm-principles-bpmn-2-0-4th-principle-information-technology-essential-enabler-bpm/ [2020, May 19].

28. DeMillo, R.A., R.J. Lipton, and F.G. Sayward, *Hints on test data selection: Help for the practicing programmer.* Computer, 1978. **11**(4): p. 34-41.

29. Estero-Botaro, A., F. Palomo-Lozano, and I. Medina-Bulo. *Quantitative evaluation of mutation operators for WS-BPEL compositions*. in *Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on*. 2010. IEEE.

30. Hu, J., N. Li, and J. Offutt. *An analysis of OO mutation operators*. in *Software Testing, Verification and Validation Workshops (ICSTW), 2011 IEEE Fourth International Conference on*. 2011. IEEE.

31. Howden, W.E., *Weak mutation testing and completeness of test sets.* IEEE Transactions on Software Engineering, 1982(4): p. 371-379.

ภาคผนวก

ภาตผนวก ก
รายละเอียดยูสเคสของเครื่องมือ

ในภาคผนวก ก เป็นการอธิบายเพิ่มเติมเกี่ยวกับรายละเอียดของยูสเคส แต่ละยูสเคสเพิ่มจากหัวข้อที่ 4.1.1 ดังตารางที่ ก-1 ถึง ก-9

ตารางที่ ก-1 รายละเอียดของยูสเคส Upload BPMN

รหัสยูสเคส	WEMUBPMN001
ยูสเคส	Upload BPMN
แอคเตอร์	นักทดสอบ
เป้าหมาย	เพื่อนำแบบจำลองบีพีเอ็มเอ็นเข้าสู่เครื่องมือ
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	แบบจำลองอยู่ในรูปแบบไฟล์ .bpmn
ขั้นตอน	1. นักทดสอบเรียกใช้โปรแกรมผ่านทางเว็บเบราว์เซอร์ 2. เครื่องมือสร้างหน้าต่างสำหรับนำเข้าแบบจำลองต้นฉบับ 3. นักทดสอบอัพโหลดแบบจำลองต้นฉบับ 4. เครื่องมือบันทึกแบบจำลองต้นฉบับ
เงื่อนไขภายหลัง	แบบจำลองได้รับการบันทึกอยู่บนเว็บเซิร์ฟเวอร์

ตารางที่ ก-2 รายละเอียดของยูสเคส Generate Mutant

รหัสยูสเคส	WEMUBPMN002
ยูสเคส	Generate Mutant
แอคเตอร์	นักทดสอบ
เป้าหมาย	เพื่อสร้างมิวแตนท์ตามประเภทการทดสอบวีคมิวเทชัน
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	แบบจำลองได้รับการบันทึกอยู่บนเว็บเซิร์ฟเวอร์
ขั้นตอน	1. นักทดสอบเลือกแบบจำลองที่บันทึกไว้ในเครื่องมือ 2. นักทดสอบเลือกประเภทการทดสอบวีคมิวเทชัน 3. นักทดสอบกดปุ่ม เพื่อให้เครื่องมือสร้างมิวแตนท์ 4. เครื่องมือสร้างมิวแตนท์ตามประเภทการทดสอบวีคมิวเทชัน 5. เครื่องมือบันทึกมิวแตนท์ที่สร้างขึ้นทั้งหมดไว้ใน Mutant Database
เงื่อนไขภายหลัง	มิวแตนท์ได้รับการบันทึกไว้ที่ Mutant Database

ตารางที่ ก-3 รายละเอียดของยูสเคส Upload Test Case

รหัสยูสเคส	WEMUBPMN003
ยูสเคส	Upload Test Case
แอคเตอร์	นักทดสอบ
เป้าหมาย	เพื่อนำกรณีทดสอบเข้าสู่เครื่องมือ
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	-
ขั้นตอน	1. นักทดสอบเลือกแบบจำลองที่บันทึกไว้ในเครื่องมือ 2. เครื่องมือจัดเก็บกรณีทดสอบไว้ที่ Test Case Database
เงื่อนไขภายหลัง	กรณีทดสอบได้รับการบันทึกไว้ที่ Test Case Database

ตารางที่ ก-4 รายละเอียดของยูสเคส Execute Test

รหัสยูสเคส	WEMUBPMN004
ยูสเคส	Execute Test
แอคเตอร์	นักทดสอบ
เป้าหมาย	เพื่อทดสอบผลลัพธ์ของแบบจำลองมิวแตนท์
ยูสเคสที่เกี่ยวข้อง	Include: Deploy BPMN Include: Execute BPMN Include: Undeploy BPMN Include: Calculate Result
เงื่อนไขก่อนหน้า	1. แบบจำลองต้นฉบับได้รับการบันทึกอยู่บนเว็บเซิร์ฟเวอร์ 2. มิวแตนท์ถูกสร้างจากแบบจำลองต้นฉบับและได้รับการบันทึกอยู่บนเว็บเซิร์ฟเวอร์ 3. กรณีทดสอบได้รับการบันทึกไว้อยู่บนเว็บเซิร์ฟเวอร์
ขั้นตอน	1. นักทดสอบเลือกกลุ่มของมิวแตนท์ถูกสร้างตามประเภทการทดสอบวีคมิวเทชัน 2. นักทดสอบเลือกกรณีทดสอบ เพื่อเตรียมทดสอบ 3. นักทดสอบกดปุ่ม เพื่อให้เครื่องมือดำเนินการทดสอบ 4. นักทดสอบเลือกกลุ่มของมิวแตนท์ถูกสร้างตามประเภทการทดสอบวีคมิวเทชัน 5. นักทดสอบเลือกกรณีทดสอบ เพื่อเตรียมทดสอบ

ตารางที่ ก-4 รายละเอียดของยูสเคส Execute Test (ต่อ)

ขั้นตอน (ต่อ)	6. นักทดสอบกดปุ่ม เพื่อให้เครื่องมือดำเนินการทดสอบ 7. เครื่องมือดึงมิวแตนท์จากกลุ่มของมิวแตนท์ 8. เครื่องมือติดตั้งแบบจำลองมิวแตนท์ 9. เครื่องมือดำเนินการทดสอบแบบจำลองมิวแตนท์ 10. เครื่องมือถอนการติดตั้งแบบจำลองมิวแตนท์ 11. เครื่องมือเปรียบเทียบผลลัพธ์ 12. เครื่องมือทำซ้ำในข้อที่ 7 – 11 จนกว่าจะครบทุกมิวแตนท์ 13. เครื่องมือคำนวณหาคะแนนมิวเทชัน 14. เครื่องมือคำนวณประสิทธิภาพของกรณีทดสอบ 15. เครื่องมือจัดเก็บผลลัพธ์การทดสอบไว้ที่ Test Result Database
เงื่อนไขภายหลัง	ผลลัพธ์ของการทดสอบถูกจัดเก็บไว้ใน Test Result Database

ตารางที่ ก-5 รายละเอียดของยูสเคส Deploy BPMN

รหัสยูสเคส	WEMUBPMN005
ยูสเคส	Deploy BPMN
แอคเตอร์	เครื่องมือ, เครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น
เป้าหมาย	เพื่อติดตั้งแบบจำลองบีพีเอ็มเอ็นที่เครื่องประมวลผลแบบจำลอง
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	แบบจำลองได้รับการบันทึกอยู่บนเว็บเซิร์ฟเวอร์
ขั้นตอน	1. เครื่องมืออัพโหลดแบบจำลองไปยังเครื่องประมวลผลแบบจำลอง
เงื่อนไขภายหลัง	แบบจำลองบีพีเอ็มเอ็นถูกติดตั้งเครื่องประมวลผลแบบจำลอง

ตารางที่ ก-6 รายละเอียดของยูสเคส Execute BPMN

รหัสยูสเคส	WEMUBPMN006
ยูสเคส	Execute BPMN
แอคเตอร์	เครื่องมือ, เครื่องประมวลผลแบบจำลอง
เป้าหมาย	เพื่อให้เครื่องประมวลผลแบบจำลองกระทำการกับแบบจำลองตามรายละเอียดที่ระบุไว้ในกรณีทดสอบ
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	แบบจำลองบีพีเอ็มเอ็นถูกติดตั้งเครื่องประมวลผลแบบจำลอง
ขั้นตอน	1. เครื่องมืออ่านข้อมูลกรณีทดสอบ 2. เครื่องมือส่งข้อมูลที่อ่านได้จากกรณีทดสอบไปยังเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น และรอรับผลลัพธ์ที่ได้กลับมา
เงื่อนไขภายหลัง	แบบจำลองถูกดำเนินการตามรายละเอียดที่ระบุไว้ในกรณีทดสอบ

ตารางที่ ก-7 รายละเอียดของยูสเคส Undeploy BPMN

รหัสยูสเคส	WEMUBPMN007
ยูสเคส	Undeploy BPMN
แอคเตอร์	เครื่องมือ, เครื่องประมวลผลแบบจำลอง
เป้าหมาย	เพื่อถอนการติดตั้งแบบจำลองบีพีเอ็มเอ็นที่เครื่องประมวลผลแบบจำลอง
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	แบบจำลองบีพีเอ็มเอ็นถูกติดตั้งเครื่องประมวลผลแบบจำลอง
ขั้นตอน	1. เครื่องมือส่งคำสั่งถอนการติดตั้งแบบจำลองไปยังเครื่องประมวลผลแบบจำลอง
เงื่อนไขภายหลัง	แบบจำลองบีพีเอ็มเอ็นถูกลบจากเครื่องประมวลผลแบบจำลอง

ตารางที่ ก-8 รายละเอียดของยูสเคส Calculate Result

รหัสยูสเคส	WEMUBPMN008
ยูสเคส	Calculate Result
แอคเตอร์	เครื่องมือ
เป้าหมาย	เพื่อจัดทำสรุปผลการทดสอบ
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	มิวแตนท์ทั้งหมดถูกทดสอบ
ขั้นตอน	1. เครื่องมือนำผลลัพธ์ที่ได้จากทดสอบผ่านเครื่องประมวลผลแบบจำลองมาตีความสถานะของมิวแตนท์ว่าถูกฆ่า หรือมีชีวิต 2. เครื่องมือคำนวณคะแนนมิวเทชัน 3. เครื่องมือคำนวณประสิทธิภาพของกรณีทดสอบ
เงื่อนไขภายหลัง	ผลลัพธ์ถูกจัดเก็บไว้ใน Test Result Database

ตารางที่ ก-9 รายละเอียดของยูสเคส Generate Test Report

รหัสยูสเคส	WEMUBPMN009
ยูสเคส	Generate Test Report
แอคเตอร์	นักทดสอบ
เป้าหมาย	เพื่อสร้างรายงานสรุปให้กับนักทดสอบ
ยูสเคสที่เกี่ยวข้อง	-
เงื่อนไขก่อนหน้า	ผลลัพธ์ของการทดสอบถูกจัดเก็บไว้ใน Test Result Database
ขั้นตอน	1. เครื่องมืออ่านผลลัพธ์ของการทดสอบ 2. เครื่องมือสร้างรายงานสรุปให้กับนักทดสอบ
เงื่อนไขภายหลัง	รายงานการทดสอบถูกสร้างขึ้น

ภาตผนวก ข
รายละเอียดคลาสของเครื่องมือ

ในภาคผนวก ข เป็นการอธิบายเพิ่มเติมเกี่ยวกับรายละเอียดของคลาส แต่ละคลาสเพิ่มเติมจากหัวข้อที่ 4.1.3

1) แพ็คเกจ controllers

1.1) คลาส IndexController คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบเป็นส่วนแรกถูกเรียกใช้งานเมื่อเข้าหน้าจอแรกของเครื่องมือ รายละเอียดดังภาพที่ ข-1

IndexController
+index() : string

ภาพที่ ข-1 คลาส IndexController

1.2) คลาส TestItemController คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ เพื่อรับไฟล์แบบจำลองบีพีเอ็มเอ็นเข้าสู่เครื่องมือ รายละเอียดดังภาพที่ ข-2

TestItemController
+listTestItem(Model : Model) : string +viewTestItem(id : int, Model : Model) : string

ภาพที่ ข-2 คลาส TestItemController

1.3) คลาส GenerateMutantController คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ โดยรับประเภทการทดสอบวีคมิวเทชันจากนักทดสอบรายละเอียดดังภาพที่ ข-3

GenerateMutantController
+listTestItem(ModelMap : ModelMap) : String +generatePossibleMutant(model : ModelMap , pParam : Map<String, String>) : List<mutantTestItem> +saveGeneratedMutant(pGeneratedMutantInfo : GeneratedMutantInfo) : String -generateMutant(pTestItemEntry : testItem) : List<mutantTestItem>

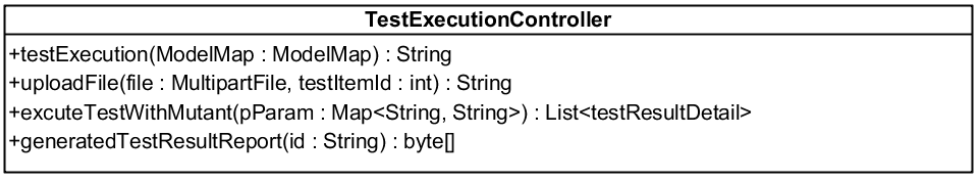
ภาพที่ ข-3 คลาส GenerateMutantController

1.4) คลาส EngineConfigController คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ โดยรับข้อมูลการเชื่อมต่อกับเครื่องประมวลผลแบบจำลองจากนักทดสอบรายละเอียดดังภาพที่ ข-4

EngineConfigController
+viewEngineConfig(id : int, Model : Model) : String +saveTestItem(engineconfigEntry : engineConfig) : String

ภาพที่ ข-4 คลาส EngineConfigController

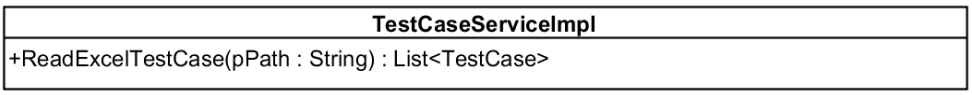
1.5) คลาส TestExecutionController คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ โดยทดสอบแบบจำลองมิวแตนท์ กับกรณีทดสอบรายละเอียดดังภาพที่ ข-5



ภาพที่ ข-5 คลาส TestExecutionController

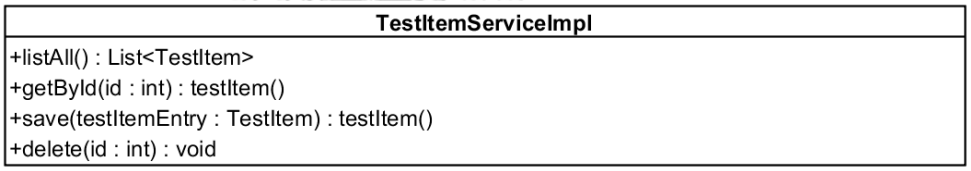
2) แพ็คเกจ manageTest

2.1) คลาส TestCaseServiceImpl คือ คลาสที่ทำหน้าที่จัดการกรณีทดสอบ รายละเอียดดังภาพที่ ข-6



ภาพที่ ข-6 คลาส TestCaseServiceImpl

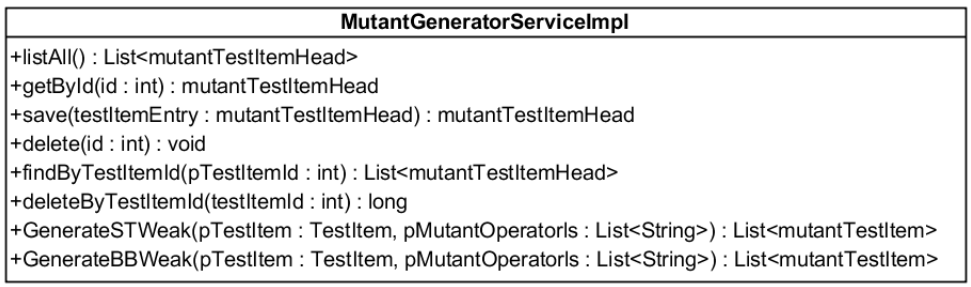
2.2) คลาส TestItemServiceImpl คือ คลาสที่ทำหน้าที่จัดการแบบจำลองบีพีเอ็มเอ็นต้นฉบับ รายละเอียดดังภาพที่ ข-7



ภาพที่ ข-7 คลาส TestItemServiceImpl

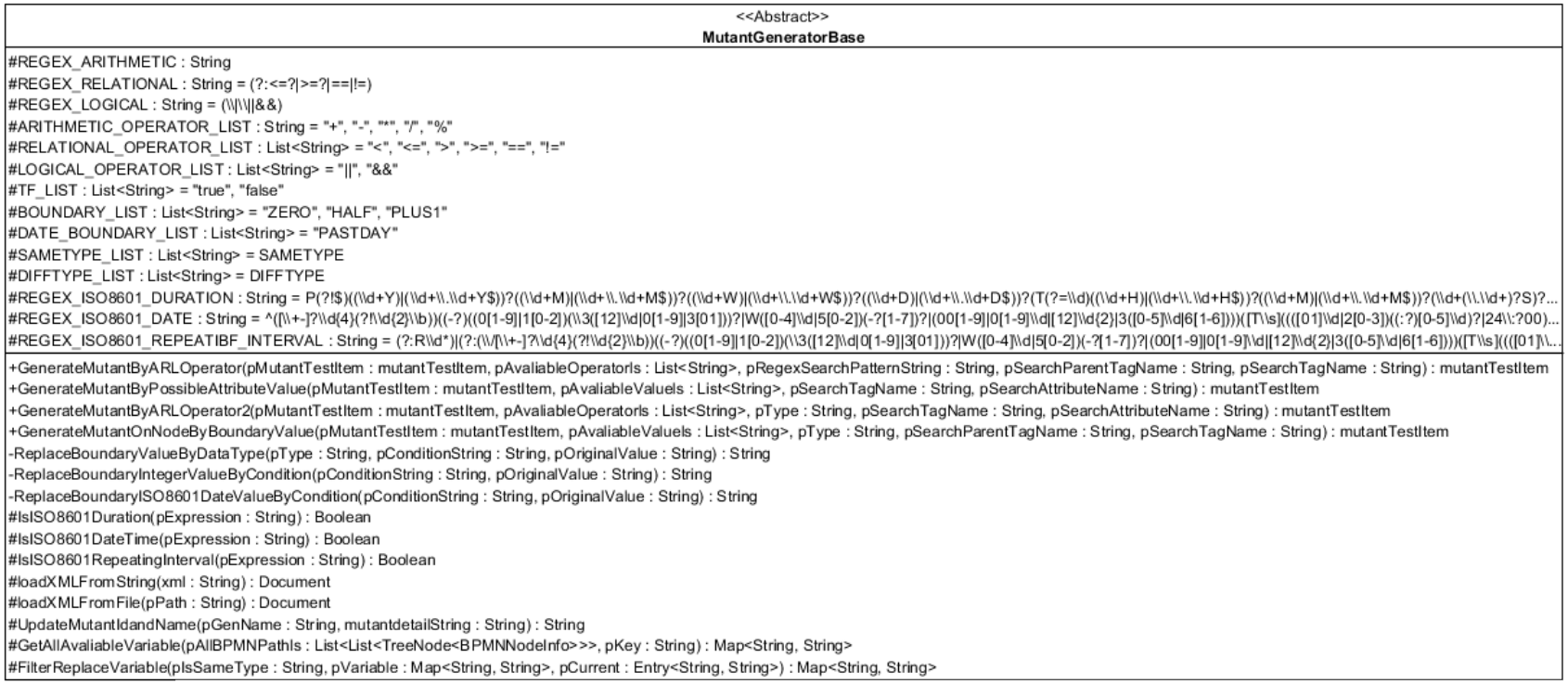
3) แพ็คเกจ mutantGenerator

3.1) คลาส MutantGeneratorServiceImpl คือ คลาสที่ทำหน้าที่จัดการมิวแตนท์ รายละเอียดดังภาพที่ ข-8



ภาพที่ ข-8 คลาส MutantGeneratorServiceImpl

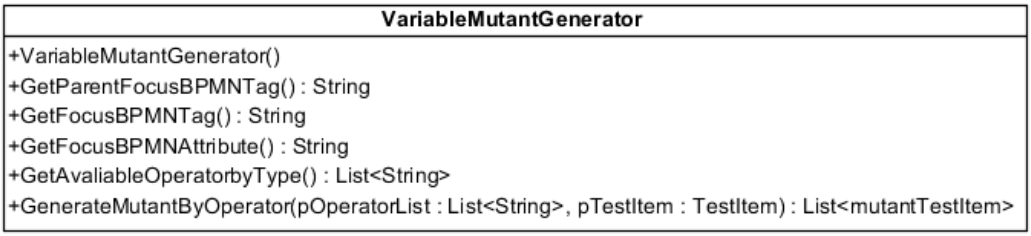
3.2) คลาส MutantGeneratorBase คือ คลาสแม่ที่เก็บการขั้นตอน วิธีการสร้างมิวแตนท์ที่สามารถใช้งานร่วมกันได้ของแต่ละตัวดำเนินการ รายละเอียดดังภาพที่ ข-9



ภาพที่ ข-9 คลาส MutantGeneratorBase

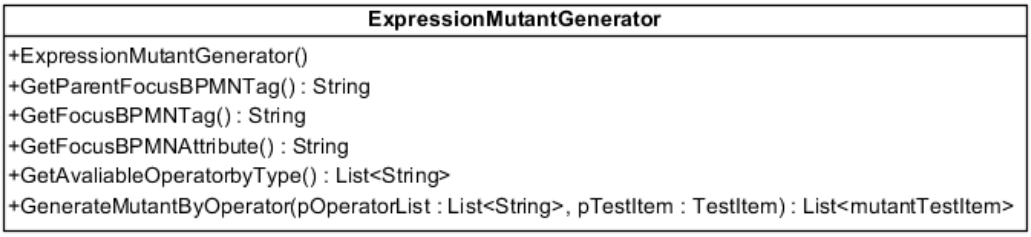
4) แพ็คเกจ mutantGenerator.operator

4.1) คลาส VariableMutantGenerator คือ คลาสสืบทอดจาก MutantGeneratorBase โดยคลาสนี้ทำหน้าที่สร้างมิวแตนท์ โดยการแทนที่ตัวแปรสำหรับตัวดำเนินการ IVR และ ITR รายละเอียดดังภาพที่ ข-10



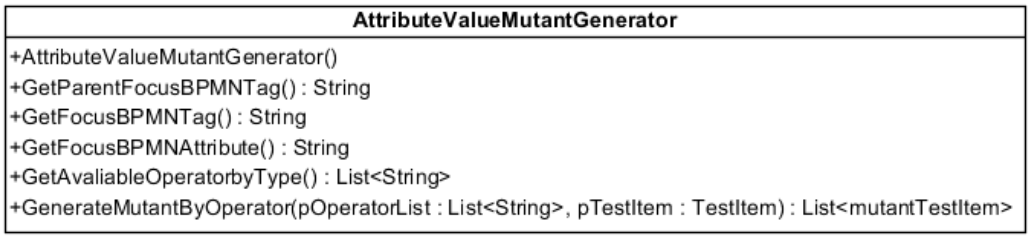
ภาพที่ ข-10 คลาส VariableMutantGenerator

4.2) คลาส ExpressionMutantGenerator คือ คลาสสืบทอดความสามารถจาก MutantGeneratorBase โดยคลาสนี้ทำหน้าที่สร้างมิวแตนท์ที่แก้ไขนิพจน์สำหรับตัวดำเนินการ EAR, ELR, ERR, AAA, AAM, AAS, ALA, ALM, ALS, ARA และ ARM รายละเอียดดังภาพที่ ข-11



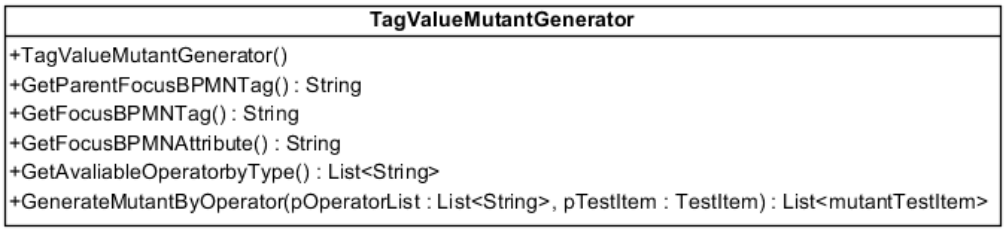
ภาพที่ ข-11 คลาส ExpressionMutantGenerator

4.3) คลาส AttributeValueMutantGenerator คือ คลาสสืบทอดความสามารถจากคลาส MutantGeneratorBase โดยคลาสนี้ทำหน้าที่สร้างมิวแตนท์ โดยการแก้ไขค่าของแอตทริบิวต์สำหรับตัวดำเนินการ ECA, EDA, ERA, ETA, ACR และ AMR รายละเอียดดังภาพที่ ข-12



ภาพที่ ข-12 คลาส AttributeValueMutantGenerator

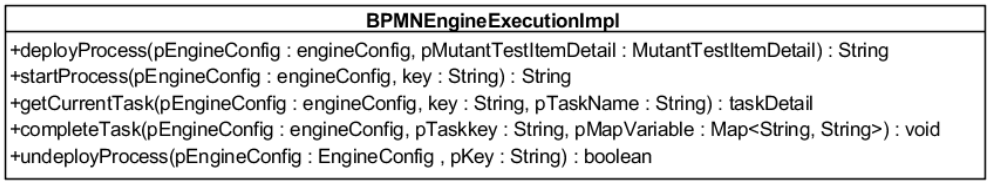
4.4) คลาส TagValueMutantGenerator คือ คลาสสืบทอดความสามารถจากคลาส MutantGeneratorBase โดยคลาสนี้ทำหน้าที่สร้างมิวแตนท์ โดยการแก้ไขค่าของแท็กสำหรับตัวดำเนินการ AOR, ARR, ASR, ATR และ XBR รายละเอียดดังภาพที่ ข-13



ภาพที่ ข-13 คลาส TagValueMutantGenerator

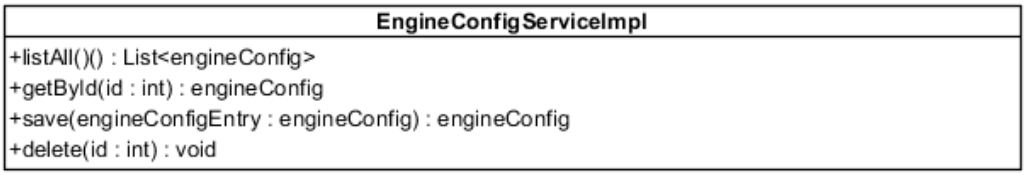
5) แพ็คเกจ testExecution

5.1) คลาส BPMNEngineExecutionImpl คือ คลาสที่ทำหน้าที่จัดการการเชื่อมต่อเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น รายละเอียดดังภาพที่ ข-14



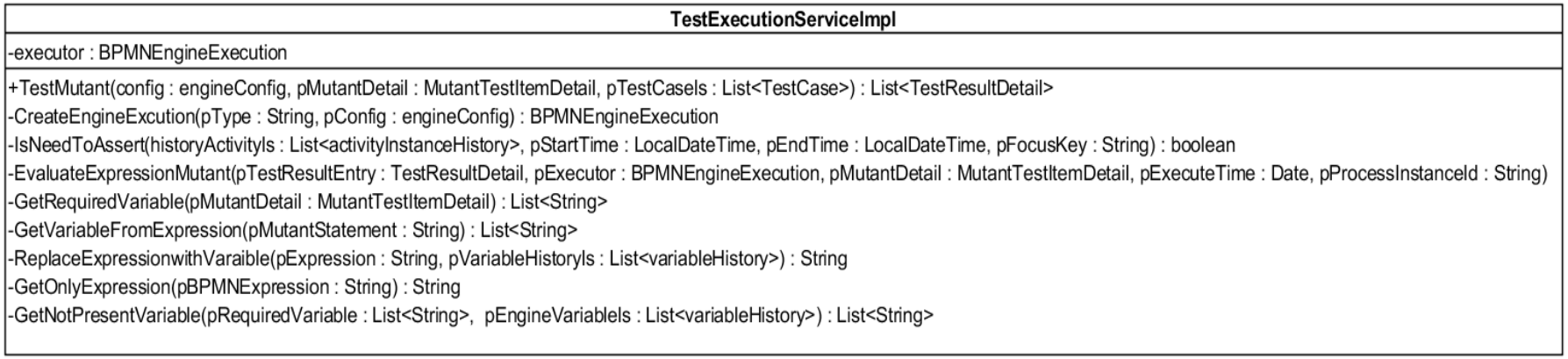
ภาพที่ ข-14 คลาส BPMNEngineExecution

5.2) คลาส EngineConfigServiceImpl คือ คลาสที่ทำหน้าที่จัดการข้อมูลการเชื่อมต่อกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น รายละเอียดดังภาพที่ ข-15



ภาพที่ ข-15 คลาส EngineConfigServiceImpl

5.3) คลาส TestExecutionServiceImpl คือ คลาสที่ทำหน้าที่ทดสอบแบบจำลองมิวแตนท์ กับกรณีทดสอบผ่านเครื่องประมวลผลแบบจำลองรายละเอียดดังภาพที่ ข-16



ภาพที่ ข-16 คลาส TestExecutionServiceImpl

5.4) คลาส TestResultServiceImpl คือ คลาสที่ทำหน้าที่จัดการผลลัพธ์การทดสอบ รายละเอียดดังภาพที่ ข-17

TestResultServiceImpl
+listAll()() : List<TestResultHead> +getById(id : int) : TestResultHead +save(testResultEntry : TestResultHead) : TestResultHead +delete(id : int) : void +deleteByMutantTestItemId(pMutantTestItemId : int) : void +CalcMutationScore(pTestResultIs : List<TestResultDetail>) : double +CalcTestEffectiveness(pTestCaseIs : List<TestCase>, pTestResultIs : List<TestResultDetail>) : double +findByMutantTestItemId(pMutantTestItemId : int) : List<TestResultHead> +generatedTestResultReport(pMutantTestItemId : int) : byte[]

ภาพที่ ข-17 คลาส TestResultServiceImpl

6) แพ็คเกจ repositories

6.1) คลาส MutantTestItemRepository คือ คลาสที่ทำหน้าที่จัดการด้านฐานข้อมูลในของคลาส MutantTestItemHead และ MutantTestItemDetail รายละเอียดดังภาพที่ ข-18

MutantTestItemRepository
+deleteByTestItemIdAndGenMutantType(testItemId : String, genMutantType : String) : Long +findByTestItemId(pTestItemId : int) : List<mutantTestItemHead> +deleteByTestItemId(testItemId : int) : Long

ภาพที่ ข-18 คลาส MutantTestItemRepository

6.2) คลาส EngineConfigRepository คือ คลาสที่ทำหน้าที่จัดการด้านฐานข้อมูลของคลาส EngineConfig รายละเอียดดังภาพที่ ข-19

EngineConfigRepository
+listAll() : List<EngineConfig>

ภาพที่ ข-19 คลาส MutantTestItemRepository

6.3) คลาส TestResultRepository คือ คลาสที่ทำหน้าที่จัดการด้านฐานข้อมูลของคลาส TestResultHead และ TestResultDetail รายละเอียดดังภาพที่ ข-20

TestResultRepository
+deleteByMutantTestItemId(pTestItemId : int) : Long +findByMutantTestItemId(pMutantTestItemId : int) : List<TestResultHead>

ภาพที่ ข-20 คลาส TestResultRepository

7) แพ็คเกจ entities

7.1) คลาส TestCase คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของกรณีทดสอบ รายละเอียดดังภาพที่ ข-21

TestCase
-testCaseId : int -testCaseName : String -testCaseDetail : List<TestCaseDetail>

ภาพที่ ข-21 คลาส Testcase

7.2) คลาส TestCaseDetail คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของกรณีทดสอบ ในส่วนของขั้นตอนการทดสอบ รายละเอียดดังภาพที่ ข-22

TestCaseDetail
-taskName : String -testInput : Map<String, String> -expectedTask : String

ภาพที่ ข-22 คลาส TestCaseDetail

7.3) คลาส TestItem คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของแบบจำลองบีพีเอ็มเอ็นต้นฉบับรายละเอียดดังภาพที่ ข-23

TestItem
-testItemId : int -testItemPath : String -testItemName : String

ภาพที่ ข-23 คลาส TestItem

7.4) คลาส MutantTestItemHead คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของมิวแตนท์ตามประเภทการทดสอบวีคมิวเทชัน เมื่อจัดเก็บลงฐานข้อมูล รายละเอียดดังภาพที่ ข-24

MutantTestItemHead
-mutantTestItemId : int -mutantTestItemCode : String -testItemId : int -genMutantType : String -mutantTestItemDetailIs : List<mutantTestItemDetail>

ภาพที่ ข-24 คลาส MutantTestItemHead

7.5) คลาส MutantTestItemDetail คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของมิวแตนท์เมื่อจัดเก็บลงฐานข้อมูล รายละเอียดดังภาพที่ ข-25

MutantTestItemDetail
-mutantTestItemDetailId : int -GenFileName : String -FoundTagId : String -FoundTagName : String -FocusKey : String -Operator : String -OriginalStatement : String -MutantStatement : String -MutantBPMNPath : String

ภาพที่ ข-25 คลาส MutantTestItemDetail

7.6) คลาส EngineConfig คือ คลาสที่หน้าที่แสดงรายละเอียดการเชื่อมต่อเครื่องประมวลผลแบบจำลอง รายละเอียดดังภาพที่ ข-26

EngineConfig
-engineConfigId : int -baseUrl : String

ภาพที่ ข-26 คลาส EngineConfig

7.7) คลาส TestResultHead คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของผลลัพธ์การทดสอบ รายละเอียดดังภาพที่ ข-27

TestResultHead
-testResultHeadId : int -mutantTestItemId : int -testStart : Date -testFinish : Date -executionTime : int -mutationScore : double -testEffectiveness : double -testCasePath : String -testResultDetailIs : List<testResultDetail>

ภาพที่ ข-27 คลาส TestResultHead

7.8) คลาส TestResultDetail คือ คลาสที่ทำหน้าที่แสดงรายละเอียดของผลลัพธ์การทดสอบของแต่ละมิวแตนท์ รายละเอียดดังภาพที่ ข-28

TestResultDetail
-testResultDetailId : int -mutantTestItemId : int -mutationOperator : String -startTime : Date -endTime : Date -executionTime : int -mutantName : String -original : String -evalOriginal : String -mutant : String -evalMutant : String -result : String -remark : String

ภาพที่ ข-28 คลาส TestResultDetail

ภาคผนวก ค
กรณีทดสอบสำหรับแบบจำลองที่สามารถทดสอบผ่านเครื่องประมวลผลแบบจำลอง

ในภาคผนวก ค เป็นการอธิบายกรณีทดสอบสำหรับแบบจำลองตัวอย่างที่สามารถทดสอบได้จริงกับเครื่องประมวลผลแบบจำลองบีพีเอ็มเอ็น

1 กรณีทดสอบของแบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์

ตารางที่ ค-1 กรณีทดสอบของแบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์ TC01

Test Case Id	1			
Test Case Name	TC01			
BPMN Name	HardwareRetailer.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Good Information	
2	Enter Good Information	Goods_Name=Computer ,Goods_Category=electrical_supplies ,Goods_Price=5000 ,Goods_Issuer=AAA	Fill in post label	
3	Fill in post label	Source=HKD ,Destination=BKK	Take out extra insurance	
4	Take out extra insurance	insurance_provider=SAMSUNG ,insurance_serviceno=SS1234	End	
5	End			

ตารางที่ ค-2 กรณีทดสอบของแบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์ TC02

Test Case Id	2			
Test Case Name	TC02			
BPMN Name	HardwareRetailer.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Good Information	
2	Enter Good Information	Goods_Name=electric drill ,Goods_Category=building_materials ,Goods_Price=3000 ,Goods_Issuer=AAA	Fill in post label	
3	Fill in post label	Source=HKD ,Destination=BKK	End	
4	End			

ตารางที่ ค-3 กรณีทดสอบของแบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์ TC03

Test Case Id	3			
Test Case Name	TC03			
BPMN Name	HardwareRetailer.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Good Information	
2	Enter Good Information	Goods_Name=Computer ,Goods_Category=electrical_supplies ,Goods_Price=5000 ,Goods_Issuer=ABC	Assign Carrier and prepare Paperwork	
3	Assign Carrier and prepare Paperwork	Carries_Provider=KERRY ,Carrier_TrackingNumber=KER152358 ,Source=HKD ,Destination=BKK	End	
4	End			

2. กรณีทดสอบของแบบจำลองการตรวจสอบยอดเงินกู้

ตารางที่ ค-4 กรณีทดสอบของแบบจำลองการตรวจสอบยอดเงินกู้ TC01

Test Case Id		1		
Test Case Name		TC01		
BPMN Name		SimpleLoanCheckWithtimeOut.bpmn		
Remakr				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Loan Amount	
2	Enter Loan Amount	name=Chatri ,loanAmt=40000	Loan Request Check Complete	
3	Loan Request Check Complete		End	
4	End			

ตารางที่ ค-5 กรณีทดสอบของแบบจำลองการตรวจสอบยอดเงินกู้ TC02

Test Case Id		2		
Test Case Name		TC02		
BPMN Name		SimpleLoanCheckWithtimeOut.bpmn		
Reamrk				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Loan Amount	
2	Enter Loan Amount	name=Chatri ,loanAmt=100000	Update Collateral	
3	Update Collateral	collaternalInfo=watch , collateralAmt=80000	Loan Request Check Complete	
4	Loan Request Check Complete		End	
5	End			

ตารางที่ ค-6 กรณีทดสอบของแบบจำลองการตรวจสอบยอดเงินกู้ TC03

Test Case Id	3			
Test Case Name	TC03			
BPMN Name	SimpleLoanCheckWithtimeOut.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Loan Amount	
2	Enter Loan Amount	name=Chatri ,loanAmt=100000	Update Collateral	
3	Update Collateral	collaternalInfo=watch , collateralAmt=10000	Reject Loan Request Check	
4	Reject Loan Request Check		End	
5	End			

ตารางที่ ค-7 กรณีทดสอบของแบบจำลองการตรวจสอบยอดเงินกู้ TC04

Test Case Id	4			
Test Case Name	TC04			
BPMN Name	SimpleLoanCheckWithtimeOut.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Loan Amount	
2	Enter Loan Amount	name=Chatri ,loanAmt=100000	Update Collateral	
3	Update Collateral		Reject Loan Request Check	PT5M
4	Reject Loan Request Check		End	
5	End			

3. กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง

ตารางที่ ค-8 กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง TC01

Test Case Id	1			
Test Case Name	TC01			
BPMN Name	CCBApproveProcess.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Change Request Info	
2	Enter Change Request Info	RedmineId=#246092 ,Subject=Extend Field paid-up share on Master File Equity (Old 13 digit/ New 15 Digit) ,Impact1=Check DB field ,Impact2=a good opportunity, migrate VB6 to .NET ,Impact3=Recheck Module BOT Report and PTI Report ,Manday=15	Associate Reviews	
3	Associate Reviews	RequiredNumOfPeers=3 ,RequiredMinNumofPeer=2 ,RequiredNumOfApproval=2	Evaluate Change Request	
4	Evaluate Change Request	reviewResult=APPROVE	Evaluate Change Request	
5	Evaluate Change Request	reviewResult=APPROVE	CCB Approved	
6	CCB Approved		End	
7	End			

ตารางที่ ค-9 กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง TC02

Test Case Id	2			
Test Case Name	TC02			
BPMN Name	CCBApproveProcess.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Change Request Info	
2	Enter Change Request Info	RedmineId=#246092 ,Subject=Extend Field paid-up share on Master File Equity (Old 13 digit/ New 15 Digit) ,Impact1=Check DB field ,Impact2=a good opportunity, migrate VB6 to .NET ,Impact3=Recheck Module BOT Report and PTI Report ,Manday=15	Associate Reviews	
3	Associate Reviews	RequiredNumOfPeers=3 ,RequiredMinNumofPeer=3 ,RequiredNumOfApproval=3	Evaluate Change Request	
4	Evaluate Change Request	reviewResult=APPROVE	Evaluate Change Request	
5	Evaluate Change Request	reviewResult=APPROVE	Evaluate Change Request	
6	Evaluate Change Request	reviewResult=APPROVE	CCB Approved	
7	CCB Approved		End	
8	End			

ตารางที่ ค-10 กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง TC03

Test Case Id	3			
Test Case Name	TC03			
BPMN Name	CCBApproveProcess.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Change Request Info	
2	Enter Change Request Info	RedmineId=#246092 ,Subject=Extend Field paid-up share on Master File Equity (Old 13 digit/ New 15 Digit) ,Impact1=Check DB field ,Impact2=a good opportunity, migrate VB6 to .NET ,Impact3=Recheck Module BOT Report and PTI Report ,Manday=15	Associate Reviews	
3	Associate Reviews	RequiredNumOfPeers=3 ,RequiredMinNumofPeer=3 ,RequiredNumOfApproval=2	Evaluate Change Request	
4	Evaluate Change Request	reviewResult=APPROVE	Evaluate Change Request	
5	Evaluate Change Request	reviewResult=REJECT	Evaluate Change Request	
6	Evaluate Change Request	reviewResult=APPROVE	CCB Approved	
7	CCB Approved		End	
8	End			

ตารางที่ ค-11 กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง TC04

Test Case Id	4			
Test Case Name	TC04			
BPMN Name	CCBApproveProcess.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Change Request Info	
2	Enter Change Request Info	RedmineId=#246092 ,Subject=Extend Field paid-up share on Master File Equity (Old 13 digit/ New 15 Digit) ,Impact1=Check DB field ,Impact2=a good opportunity, migrate VB6 to .NET ,Impact3=Recheck Module BOT Report and PTI Report ,Manday=15	Associate Reviews	
3	Associate Reviews	RequiredNumOfPeers=3 ,RequiredMinNumofPeer=2 ,RequiredNumOfApproval=2	Evaluate Change Request	
4	Evaluate Change Request	reviewResult=REJECT	Evaluate Change Request	
5	Evaluate Change Request	reviewResult=REJECT	CCB Rejected	
6	CCB Approved		End	
7	End			

ตารางที่ ค-12 กรณีทดสอบของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง TC05

Test Case Id	5			
Test Case Name	TC05			
BPMN Name	CCBApproveProcess.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Change Request Info	
2	Enter Change Request Info	RedmineId=#246092 ,Subject=Extend Field paid-up share on Master File Equity (Old 13 digit/ New 15 Digit) ,Impact1=Check DB field ,Impact2=a good opportunity, migrate VB6 to .NET ,Impact3=Recheck Module BOT Report and PTI Report ,Manday=15	Associate Reviews	
3	Associate Reviews	RequiredNumOfPeers=3 ,RequiredMinNumofPeer=3 ,RequiredNumOfApproval=2	Evaluate Change Request	
4	Evaluate Change Request	reviewResult=APPROVE	Evaluate Change Request	
5	Evaluate Change Request	reviewResult=REJECT	Evaluate Change Request	
6	Evaluate Change Request	reviewResult=REJECT	CCB Rejected	
7	CCB Approved		End	
8	End			

4. กรณีทดสอบของแบบจำลองตรวจสอบการเบิกจ่าย

ตารางที่ ค-13 กรณีทดสอบของแบบจำลองตรวจสอบการเบิกจ่าย TC01

Test Case Id	1			
Test Case Name	TC01			
BPMN Name	DisburseProcessWS.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Request Id	
2	Enter Change Request Info	RequestId=99	No Request	
3	No Request		end	
4	End			

ตารางที่ ค-14 กรณีทดสอบของแบบจำลองตรวจสอบการเบิกจ่าย TC02

Test Case Id	2			
Test Case Name	TC02			
BPMN Name	DisburseProcessWS.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Request Id	
2	Enter Change Request Info	RequestId=1	Review Request	
3	Review Request		Approve	
4	Approve	Approve=yes ,ApproveReason=OK	Request Complete	
5	Request Complete		end	
6	end			

ตารางที่ ค-15 กรณีทดสอบของแบบจำลองตรวจสอบการเบิกจ่าย TC03

Test Case Id	3			
Test Case Name	TC03			
BPMN Name	DisburseProcessWS.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Request Id	
2	Enter Change Request Info	RequestId=1	Review Request	
3	Review Request		Approve	
4	Approve	Approve=no ,ApproveReason=over budget	Request Rejected	
5	Request Rejected		end	
6	end			

ตารางที่ ค-16 กรณีทดสอบของแบบจำลองตรวจสอบการเบิกจ่าย TC04

Test Case Id	4			
Test Case Name	TC04			
BPMN Name	DisburseProcessWS.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Request Id	
2	Enter Change Request Info	RequestId=2	Request Complete	
3	Request Complete		end	
4	End			

5. กรณีทดสอบของแบบจำลองการสร้างรายงานข้อบกพร่องของระบบ

ตารางที่ ค-17 กรณีทดสอบของแบบจำลองการสร้างรายงานข้อบกพร่องของระบบ TC01

Test Case Id	1			
Test Case Name	TC01			
BPMN Name	BatchInidentReport.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Timer Start		Review Incident Report	PT1M
2	Review Incident Report		end	
3	Timer Start		View Incident Report Summary	PT1M
4	View Incident Report Summary		end	
5	end			

6. กรณีทดสอบของแบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด

ตารางที่ ค-18 กรณีทดสอบของแบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด TC01

Test Case Id	1			
Test Case Name	TC01			
BPMN Name	IncidentSupport.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Incident Information	
2	Enter Incident Information	Subject=Not Found Market Price of UOB , Description=Not Found Market Price of UOB.	First Line Support	
3	First Line Support	Workaround=Please check price code and edit price code Same as Security Code , NeedToFixedSW=false	View Result	
4	View Result		End	
5	End			

ตารางที่ ค-19 กรณีทดสอบของแบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด TC02

Test Case Id	2			
Test Case Name	TC02			
BPMN Name	IncidentSupport.bpmn			
Remark				
Test Steps				
No	**Task Name**	**Test Input**	**Expected Task**	**Wait Time**
1	Start		Enter Incident Information	
2	Enter Incident Information	Subject=Not Found Market Price of UOB , Description=Not Found Market Price of UOB.	First Line Support	
3	First Line Support		Second Line Support	P1DT0M0S
4	Second Line Support	Workaround=Please check price code and edit price code Same as Security Code , NeedToFixedSW=false	View Result	
5	View Result		End	
6	End			

ภาคผนวก ง
ผลการสร้างมิวแตนท์ของเครื่องมือกับแบบจำลองที่ใช้ในการทดสอบ

ในภาคผนวก ง เป็นการแสดงผลการสร้างมิวแตนท์ของเครื่องมือกับแบบจำลองที่ใช้ในการทดสอบ

<u>แบบจำลองที่ 1</u> แบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์

ตารางที่ ง-1 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการส่งสินค้าของร้านฮาร์ดแวร์

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	EAR	4	4	4
2	ELR	1	1	1
3	ERR	10	10	10
4	ITR	16	16	16
5	IVR	2	2	2
จำนวนมิวแตนท์ทั้งหมด		33	33	33

<u>แบบจำลองที่ 2</u> : แบบจำลองการตรวจสอบยอดเงินกู้

ตารางที่ ง-2 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการตรวจสอบยอดเงินกู้

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	IVR	3	3	3
2	ITR	6	6	6
3	ERR	10	10	10
4	EAR	8	8	8
5	ETA	3	3	3
จำนวนมิวแตนท์ทั้งหมด		**30**	**30**	**30**

<u>แบบจำลองที่ 3</u> : แบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง

ตารางที่ ง-3 แสดงผลการสร้างมิวแตนท์ของแบบจำลองอนุมัติการแก้ไขตามคำร้องขอการเปลี่ยนแปลง

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	ALM	1	1	1
2	ARM	10	10	10
3	ERR	5	5	5
4	ITR	15	15	15
5	IVR	6	6	6
6	ACR	-	3	3
จำนวนมิวแตนท์ทั้งหมด		37	40	40

<u>แบบจำลองที่ 4</u> : แบบจำลองตรวจสอบการเบิกจ่าย

ตารางที่ ง-4 แสดงผลการสร้างมิวแตนท์ของแบบจำลองตรวจสอบการเบิกจ่าย

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	AAM	4	4	4
2	ALM	1	1	1
3	ARM	10	10	10
4	EAR	16	16	16
5	ERR	25	25	25
6	ITR	21	21	21
7	IVR	15	15	15
8	ACR	-	3	3
จำนวนมิวแตนท์ทั้งหมด		92	95	95

<u>แบบจำลองที่ 5</u> : แบบจำลองการสร้างรายงานข้อบกพร่องของระบบ

ตารางที่ ง-5 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการสร้างรายงานข้อบกพร่องของระบบ

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	ERR	10	10	10
2	ERA	-	3	3
3	ECA	-	3	3
จำนวนมิวแตนท์ทั้งหมด		10	16	16

<u>แบบจำลองที่ 6</u> : แบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด

ตารางที่ ง-6 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการจัดการคำร้องขอการตรวจสอบข้อผิดพลาด

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	EDA	1	1	1
จำนวนมิวแตนท์ทั้งหมด		1	1	1

<u>แบบจำลองที่ 7</u> : แบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ

ตารางที่ ง-7 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการคำนวณการซื้อกองทุนอาร์เอ็มเอฟ

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	IVR	24	24	24
2	ITR	20	20	20
3	ERR	5	5	5
4	AAS	24	24	24
5	ARS	10	10	10
6	ALS	1	1	1
7	ATR	-	1	1
8	AMR	-	3	3
จำนวนมิวแตนท์ทั้งหมด		84	88	88

<u>แบบจำลองที่ 8</u> : แบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบ

ตารางที่ ง-8 แสดงผลการสร้างมิวแตนท์ของแบบจำลองการตรวจสอบความพร้อมก่อนการขึ้นระบบ

ลำดับที่	ตัวดำเนินการมิวเทชัน	การทดสอบวีคมิวเทชัน		
		ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
1	IVR	2	2	2
2	AAM	8	8	8
3	ARM	10	10	10
4	AAA	4	4	4
5	ARA	10	10	10
6	ALA	1	1	1
7	AOR	-	1	1
8	ARR	-	1	1
9	XBR	3	3	3
10	ASR	-	1	1
11	ACR	-	6	6
จำนวนมิวแตนท์ทั้งหมด		38	47	47

ประวัติผู้เขียน

ชื่อ-สกุล	ชาตรี งามเบญจวงศ์
วัน เดือน ปี เกิด	23 เมษายน 2533
สถานที่เกิด	โรงพยาบาลราชวิถี
วุฒิการศึกษา	1.วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยศรีนครินทรวิโรฒ (พ.ศ. 2552 - 2555) 2.วิทยาศาสตรมหาบัณฑิต สาขาวิศวกรรมซอฟต์แวร์ จุฬาลงกรณ์มหาวิทยาลัย (พ.ศ. 2559 - 2562)
ที่อยู่ปัจจุบัน	47 ซอย ชัยพฤกษ์ 18 ถนน ชัยพฤกษ์ แขวงตลิ่งชัน เขตตลิ่งชัน กรุงเทพฯ 10170
ผลงานตีพิมพ์	Chatri Ngambenchawong and Taratip Suwannasart. 2019. A Weak Mutation Testing Framework for BPMN. In Proceedings of the International MultiConference of Engineers and Computer Scientists 2019 (IMECS 2019), March 13-15, 2019, Hong Kong, 571-575