เครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล

นายปัญญา บุญยกุลศรีรุ่ง

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2554

A WEAK MUTATION TESTING TOOL FOR WS-BPEL

Mr. Panya Boonyakulsrirung

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Software Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2011

หัวข้อวิทยานิพนธ์	เครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล
โดย	นายปัญญา บุญยกุลศรีรุ่ง
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(รองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ)

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

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

..... กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร.ภัทรชัย ลลิตโรจน์วงศ์)

ปัญญา บุญยกุลศรีรุ่ง: เครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล. (A WEAK MUTATION TESTING TOOL FOR WS-BPEL) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร. ธาราทิพย์ สุวรรณศาสตร์, 371 หน้า.

การเติบโตของสถาปัตยกรรมเชิงบริการในอุตสาหกรรมซอฟต์แวร์นั้นเพิ่มขึ้นอย่างมากในหลายปีที่ผ่านมา เนื่องจากข้อดีของสถาปัตยกรรมเชิงบริการนั้นสามารถนำกลับมาใช้ใหม่ได้ ง่ายต่อการพัฒนา และการที่ไม่ขึ้นต่อกันของบริการของแต่ละบริการ ซึ่งสถาปัตยกรรมเชิงบริการนั้นถูกนำมาประยุกต์ใช้โดยเว็บเซอร์วิซดับเบิลยูเอสบีเพลซึ่งใช้ไวยกรณ์ทั้งหมดเป็นภาษาเอ็กซ์เอ็มแอล ถูกนำมาใช้ทำให้เว็บเซอร์วิซทำงานได้ซับซ้อนมากยิ่งขึ้นเพื่อตอบสนองต่อกระบวนการทางธุรกิจในแต่ละองค์กร ขณะเดียวกันการทดสอบแบบมิวเทชันซึ่งเป็นชนิดของการทดสอบซอฟต์แวร์แบบหนึ่ง และเป็นที่สนใจมาชั่วระยะเวลาหนึ่งโดยมีการนำไปทดสอบกับภาษาหลายๆ ภาษา เช่น จาวา ซี เอสคิวแอล และ เอดา เป็นต้น แต่ก็ยังถูกใช้ไม่มากในภาษาบีเพล การทดสอบแบบวีคมิวเทชันก็เป็นการทดสอบมิวเทชันอีกชนิดหนึ่งซึ่งมีข้อดีในการลดความสิ้นเปลืองในการทดสอบแบบมิวเทชัน

งานวิจัยนี้ได้นำเสนอการวิเคราะห์เกี่ยวกับตัวดำเนินการมิวเทชันสำหรับดับเบิลยูเอสบีเพลว่าสามารถประยุกต์ใช้ได้กับแนววิธีการทดสอบแบบวีคมิวเทชันได้หรือไม่ และยังเสนอเครื่องมือที่ใช้ในการสร้างมิวแตนท์โดยใช้การวิเคราะห์แบบวีคมิวเทชัน ซึ่งตัวดำเนินการมิวเทชันได้ถูกเสนอไว้ในงานที่ผ่านมา และถูกแบ่งออกเป็น 4 ประเภท คือ ตัวดำเนินการมิวเทชันเกี่ยวกับตัวระบุ ตัวดำเนินการมิวเทชันสำหรับนิพจน์ ตัวดำเนินการมิวเทชันสำหรับประพจน์ และตัวดำเนินการมิวเทชันสำหรับความผิดปกติและเหตุการณ์ เครื่องมือที่เสนอถูกเรียกว่า วีมิวท์ ซึ่งสามารถรายงานผลลัพธ์ต่างๆ เช่น จำนวนมิวแตนท์ที่กำจัดได้ จำนวนมิวแตนท์ที่ยังคงอยู่ เวลาที่ใช้ไปในการทดสอบ คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

ภาควิชา…วิศวกรรมคอมพิวเตอร์.... ลายมือชื่อนิสิต……..……

สาขาวิชา…วิศวกรรมซอฟต์แวร์…… ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก…….....……

ปีการศึกษา…2554……

5271521921 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: WEB SERVICE / WS-BPEL / MUTATION TESTING / MUTATION OPERATORS / WEAK MUTATION TESTING

PANYA BOONYAKULSRIRUNG: A WEAK MUTATION TESTING TOOL FOR WS-BPEL. ADVISOR: ASSOC.PROF. TARATIP SUWANNASART PH.D., 371 pp.

Service Oriented Architecture has dynamically increased in software industry in many years ago because many advantages such as are reusable, easy to implement and loosely coupled. Service Oriented Architecture is made more concrete with web services. Web Service Business Process Execution Language (WS-BPEL) appears to solve and support more complex business processes in many enterprises. Meanwhile, mutation testing is error-based software testing in unit level which indicates efficiency of test suites. This technique is applied in many languages such as Java, C, SQL, and Ada. Weak mutation testing can reduce execution cost more than typical mutation testing.

This thesis proposes a comprehensive analysis by using some mutation operators for WS-BPEL that can apply for weak mutation testing and also proposes a tool for generating mutants by using weak mutation technique. Those operators are proposed in previous work and classified in four categories of mutation operators. Those are Identifier replace operators, Activity operators, as well as Exceptional and Event operators. The proposed tool called WeMuTe that can identify dead mutants and live mutants, execution time, mutation score, and test cases effectiveness.

Department: Computer Engineering Student's Signature: .....
Field of Study: Software Engineering Advisor's Signature: .....
Academic Year: 2011

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ได้สำเร็จลุล่วงด้วยความเมตตาและความช่วยเหลืออย่างยิ่งมากจาก รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์ อาจารย์ที่ปรึกษา ที่เสียสละเวลาช่วยให้คำปรึกษา ข้อคิดและคำแนะนำที่มีประโยชน์ต่องานวิจัย ตลอดจนความเอาใจใส่และความเชื่อมั่นที่อาจารย์มีให้ผู้วิจัย เป็นแรงผลักดันให้ผู้วิจัยมีกำลังใจทำงานต่อไป

ขอกราบขอบพระคุณรองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ ที่กรุณาเสียสละเวลามาเป็นประธานกรรมการสอบวิทยานิพนธ์ ขอกราบขอบพระคุณผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ทองทักษ์ และผู้ช่วยศาสตราจารย์ ดร.ภัทรชัย ลลิตโรจวงศ์ ที่กรุณาสละเวลามาเป็นกรรมการสอบวิทยานิพนธ์ ทั้งนี้ยังให้คำแนะนำและข้อบกพร่องที่ต้องแก้ไขให้วิทยานิพนธ์ฉบับนี้มีความสมบูรณ์มากยิ่งขึ้น

ขอขอบพระคุณคณาจารย์ในภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัยทุกท่าน ที่ให้ความรู้อันมีค่ายิ่งแก่ผู้วิจัย

ขอขอบคุณบุคลากรในภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัยทุกท่าน ที่ให้ข้อมูล คำแนะนำและความช่วยเหลือในการดำเนินการทั้งในเรื่องการศึกษาและการสอบวิทยานิพนธ์ได้สำเร็จลุล่วง

ขอขอบคุณ เพื่อนๆ พี่ๆ และน้องๆ ทุกคนของผู้วิจัย ที่ห่วงใยและให้ความช่วยเหลือในทุกๆ ด้านจนผู้วิจัยสามารถทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วง ขอบคุณสมาชิกในห้องปฏิบัติการวิศวกรรมซอฟต์แวร์ที่คอยให้คำแนะนำและแรงสนับสนุนแก่ผู้วิจัย

ขอขอบพระคุณสมาชิกในครอบครัวทุกคน ที่ให้การสนับสนุนและให้กำลังใจแก่ผู้วิจัยเสมอมา

และท้ายที่สุดนี้ ขอกราบขอบพระคุณบิดา มารดา ที่คอยเลี้ยงดู สั่งสอนผู้วิจัยจนเติบใหญ่ ท่านทั้งสองเปรียบเหมือนแรงผลักดันให้ผู้วิจัยมีกำลังใจไม่ว่าจะเรื่องใดก็ตามที่เกิดความท้อแท้ จนทำให้วิทธยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้

สารบัญ

สารบัญตาราง

หน้า

หน้า

หน้า

สารบัญภาพ

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การเติบโตของสถาปัตยกรรมเชิงบริการซึ่งถูกเรียกว่า เอสโอเอ (SOA – Service Oriented Architecture) [1] ในอุตสาหกรรมการผลิตซอฟต์แวร์นั้นเพิ่มมากขึ้น ซึ่งซอฟต์แวร์ประเภทนี้ถูกแบ่งออกเป็นหลายๆส่วน แต่ละส่วนนั้นเรียกว่าบริการ (Services) โดยบริการเหล่านี้อยู่ในรูปแบบการไม่ขึ้นต่อกัน (Loosely Coupled) ปัจจุบันสถาปัตยกรรมเชิงบริการนั้นได้ถูกนำมาทำให้เกิดผลโดยเว็บเซอร์วิซซึ่งมีมาตรฐานในการติดต่อหรือการทำงานร่วมกันระหว่างโปรแกรมประยุกต์ (Application) ซึ่งอาจจะทำงานอยู่คนละแพลตฟอร์มหรือกรอบงานที่ต่างกัน

เว็บเซอร์วิซสามารถเข้าถึงเครือข่ายได้โดยการใช้ข้อความที่เป็นเอ็กซ์เอ็มแอล (XML – Extensible Markup Language) [2] ด้วยมาตรฐานของโซป (SOAP – Simple Object Access Protocol) [3] ซึ่งจะถูกอธิบายโดยดับเบิลยูเอสดีแอลหรือวิซเดิล (WSDL – Web Services Description Language) [4] ซึ่งเป็นเอกสารเอ็กซ์เอ็มแอลอีกเช่นกัน โดยบอกถึงรายละเอียดในการติดต่อกับเว็บเซอร์วิซเพื่อให้โปรแกรมประยุกต์ที่ต้องการเรียกใช้เว็บเซอร์วิซสามารถใช้บริการอะไรได้บ้างและจะติดต่อได้อย่างไร

ดับเบิลยูเอสบีเพล (WS-BPEL – Web Services Business Process Execution Language) หรือบีเพล (BPEL – Business Process Execution Language) [5] เป็นอีกภาษาหนึ่งซึ่งพัฒนาขึ้นมาจากภาษาเอ็กซ์เอ็มแอล กล่าวคือ บีเพลใช้ไวยากรณ์ทั้งหมดเป็นภาษาเอ็กซ์เอ็มแอลเพื่อเป็นการอธิบายกระบวนการการทำงานของบริการ ซึ่งบีเพลมักจะถูกนำมาใช้ภายในองค์กรเพื่ออธิบายการทำงานระบบที่เป็นเชิงบริการ และทำให้บริการต่างๆทำงานร่วมกันได้ซับซ้อนและมีประสิทธิภาพมากยิ่งขึ้น

การทดสอบแบบมิวเทชัน (Mutation Testing) นั้นก็เป็นที่นิยมในการทดสอบการเขียนโปรแกรม ซึ่งจากผลงานวิจัย [6] แนวโน้มการศึกษาวิธีการทดสอบนี้ก็ยังคงเป็นที่นิยม และนำไปใช้ในการทดสอบกับหลายๆภาษา ยกตัวอย่างเช่น Ada Java และ C# เป็นต้น อย่างไรก็ตาม ยังถูกใช้ไม่มากในการนำการทดสอบดังกล่าวมาใช้กับภาษาบีเพล โดยหลักการของการ

ทดสอบแบบมิวเทชันนั้นจะเป็นการใส่ข้อผิดพลาดลงไปในโปรแกรมเพื่อให้เกิดข้อผิดพลาดเพียงหนึ่งที่ต่อหนึ่งโปรแกรม และโปรแกรมที่ถูกใส่ข้อผิดพลาดลงไปจะถูกเรียกว่า มิวแตนท์ (Mutant) ซึ่งการสร้างมิวแตนท์นั้นจะพิจารณาจากตัวดำเนินการมิวเทชัน (Mutation Operator) ว่าจะใส่ข้อผิดพลาดลงไปในโปรแกรมในรูปแบบไหนได้บ้าง เช่น การใส่ข้อผิดพลาดที่เกี่ยวกับทางนิพจน์โดยการไปเปลี่ยนเครื่องหมาย <, >, <=, >= , =,!= ในนิพจน์ที่ต้องการเปลี่ยน

จากผลงานวิจัยที่ผ่านมา [7] ได้เสนอการทดสอบแบบมิวเทชันชนิดการเลือก (Selective Mutation) ซึ่งเป็นเทคนิคการลดจำนวนมิวแตนท์ (Mutant Reduction Techniques) และได้กำหนดตัวดำเนินการขึ้นมาโดยเป็นตัวดำเนินการมิวเทชันสำหรับนิพจน์ (Expression Mutation Operator) ซึ่งมีอยู่ทั้งหมด 5 ตัวดำเนินการ คือ AOR, ROR, LOR, LOD และ LOI แต่การนำมาใช้นั้นยังไม่ครบทุกตัวดำเนินการในภาษาบีเพล เนื่องจากตัวดำเนินการที่กำหนดขึ้นมายังมีตัวดำเนินการมิวเทชันประเภทอื่นๆอีก

จากงานวิจัย [8] ได้เสนอตัวดำเนินการมิวเทชันสำหรับบีเพลอีก 3 รูปแบบ ซึ่งได้พิจารณาและถูกกำหนดขึ้นมาจากงานวิจัย [9] โดยรูปแบบที่เพิ่มขึ้นมาคือ ตัวดำเนินการมิวเทชันสำหรับตัวระบุ (Mutation Operator Identifier) ตัวดำเนินการมิวเทชันสำหรับกิจกรรม (Mutation Operator Activities) และ ตัวดำเนินการมิวเทชันสำหรับเงื่อนไขที่เกี่ยวกับข้อผิดปกติและเหตุการณ์ (Mutation Operator Exceptional Condition and Event) ทั้งนี้ได้ใช้การทดสอบแบบสตรองมิวเทชัน (Strong Mutation) เพื่อวัดว่าตัวดำเนินการแต่ละตัวนั้นสร้างมิวแตนท์ได้เท่าไหร่ในแต่ละโปรแกรมที่เตรียมไว้ เพื่อให้นักทดสอบได้สร้างชุดทดสอบได้เหมาะกับโปรแกรมนั้นๆ และใช้การทดสอบแบบมิวเทชันชนิดแข็งแรงซึ่งมีข้อเสียในด้านการสิ้นเปลืองที่เกี่ยวกับการกระทำการ (Computational Cost)

งานวิจัย [10] ได้ศึกษาแนวทางการทดสอบแบบวีคมิวเทชัน (Weak Mutation Testing) ซึ่งเป็นเทคนิคการใส่ข้อผิดพลาดลงไปในโปรแกรมสำหรับการทดสอบซอฟต์แวร์ในระดับหน่วยย่อย หรือคอมโพเนนท์ (Component) ซึ่งการทดสอบแบบวีคมิวเทชันนั้น ถูกเสนอเพื่อลดความสิ้นเปลืองของการทดสอบมิวเทชัน อย่างไรก็ตามการทดสอบแบบวีคมิวเทชันก็ยังถูกมองว่ากรณีทดสอบนั้นด้อยประสิทธิภาพกว่าการทดสอบแบบสตรองมิวเทชัน คือ พิจารณาที่ผลลัพธ์ของการทดสอบ ซึ่งผลลัพธ์จากการนำการทดสอบแบบสตรองมิวเทชันไปใช้ได้เปรียบเทียบผลลัพธ์ของการทดสอบแบบวีคมิวเทชันด้วย

งานวิจัยนี้ได้วิเคราะห์ตัวดำเนินการของภาษาบีเพลในงานวิจัย [8] และศึกษากระบวนการในการทดสอบแบบวีคมิวเทชันของงานวิจัยที่ผ่านมาจาก [10 - 13] จากนั้นพิจารณาความเป็นไปได้ในการนำตัวดำเนินการมิวเทชันแต่ละตัวที่สามารถนำมาใช้ได้กับการทดสอบแบบวีคมิวเทชันสำหรับบีเพล รวมไปถึงวิเคราะห์หลักในการสร้างมิวแตนท์ที่สามารถเกิดขึ้นได้ในแต่ละตัวดำเนินการมิวเทชัน และได้เสนอเครื่องมือในการสร้างมิวแตนท์ของเอกสารบีเพลโดยใช้แนวทางการทดสอบแบบวีคมิวเทชัน โดยทดสอบมิวแตนท์ที่สร้างขึ้นมาด้วยกรณีทดสอบที่เตรียมไว้ และรายงานผลลัพธ์ผ่านทางเครื่องมือ เช่น จำนวนมิวแตนท์ที่กำจัดได้ จำนวนมิวแตนท์ที่ยังคงอยู่ และเวลาที่ใช้ไปในการทดสอบ นอกจากนี้ได้แสดงผลลัพธ์ของการทดสอบแบบวีคมิวเทชันเปรียบเทียบกับการทดสอบแบบมิวเทชันแบบทั่วไปผ่านทางเครื่องมือให้ผู้ใช้อีกด้วย

1.2 วัตถุประสงค์ของการวิจัย

1) เพื่อประยุกต์ตัวดำเนินการมิวเทชันสำหรับภาษาบีเพลมาใช้กับการทดสอบแบบวีคมิวเทชัน
2) เพื่อสร้างเครื่องมือที่ลดภาระของนักทดสอบ ในการสร้างมิวแตนท์ที่ได้จากการทดสอบแบบวีคมิวเทชันสำหรับเอกสารบีเพล

1.3 ขอบเขตการวิจัย

1.3.1 เอกสารบีเพลที่ใช้ในการทดสอบนั้นต้องเป็นไปตามมาตรฐานของโอเอซิสเวอร์ชัน 2.0

1.3.2 ผู้ใช้ต้องสร้างกรณีทดสอบที่ใช้ทดสอบโปรแกรมเอง เนื่องจากระบบยังไม่สามารถสร้างกรณีทดสอบเองได้

1.3.3 เครื่องมือต้องสามารถตรวจสอบเอกสารที่ต้องการทดสอบว่าเป็นเอกสารบีเพลหรือไม่

1.3.4 เครื่องมือสามารถเลือกประเภทของตัวดำเนินการชนิดต่างๆ และวิธีการทดสอบแบบวีคมิวเทชันชนิดทั้ง 4 แบบได้ คือ EX-WEAK/1, ST-WEAK/1, BB-WEAK/1 และ BB-WEAK/N

1.3.5 เครื่องมือสามารถสร้างมิวแตนท์ได้อย่างอัตโนมัติ

1.3.6 เครื่องมือนี้ไม่สามารถระบุมิวแตนท์สมมูลได้

1.3.7 เครื่องมือนี้ทดสอบกับโปรแกรมบีเพลอย่างน้อย 3 เอกสารบีเพล

1.3.8 เครื่องมือสามารถรายงานผลลัพธ์ของการทดสอบได้ โดยแสดงมิวแตนท์ทั้งหมดที่เกิดขึ้น, มิวแตนท์ที่ถูกกำจัด, เวลาที่ใช้ในการทดสอบ, คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

1.4 ขั้นตอนการวิจัย

ขั้นตอนการดำเนินงานของงานวิจัยนี้มีรายละเอียดดังต่อไปนี้

1.4.1 ศึกษา สำรวจ และวิเคราะห์ถึงงานวิจัยและเครื่องมือที่สามารถใช้ได้ในปัจุบัน โดยเริ่มจากงานวิจัย [7] ได้ศึกษาเกี่ยวกับการทดสอบมิวเทชันแบบการคัดเลือก และตัวดำเนินการมิวเทชันประเภทนิพจน์ที่ผู้จัยได้กำหนดขึ้นเอง รวมไปถึงเครื่องมือที่นำมาใช้ในการพัฒนาเครื่องมือ และได้ศึกษาและวิเคราะห์งานวิจัย [8] ซึ่งเสนอตัวดำเนินการมิวเทชันสำหรับภาษาบีเพลเพิ่มขึ้น คือ ตัวดำเนินการมิวเทชันประเภทตัวระบุ ตัวดำเนินการมิวเทชันประเภทกิจกรรม และตัวดำเนินการมิวเทชันสำหรับความผิดปกติและเหตุการณ์

1.4.2 ศึกษาโครงสร้างของภาษาบีเพลตามมาตรฐานของโอเอซิส ได้ทำความเข้าใจถึงนิยาม ข้อกำหนดต่างๆเกี่ยวกับสัญลักษณ์ทางภาษา ไวยกรณ์ และความสัมพันธ์ระหว่างข้อกำหนดเฉพาะอื่นๆ ยกตัวอย่างเช่น ข้อกำหนดของวิซเดิล หรือข้อกำหนดของเอ็กเอ็มแอล เป็นต้น

1.4.3 ศึกษาเครื่องมือที่ใช้พัฒนาระบบ เปรียบเทียบเครื่องมือหลายๆค่ายที่เป็นไปได้ที่เช่น อาพาเช่ทอมแคท 6.0.32 (Apache Tomcat 6.0.32) เครื่องประมวลผลเจบอส 5.1.0 (JBoss 5.1.0) และ เครื่องประมวลผลแกลสฟิช 3.0.1 (Glassfish 3.0.1) รวมไปถึงการศึกษาอาพาเช่โอดีอี 1.3.5 (Apache ODE 3.5) ซึ่งต้องติดตั้งลงบนอาพาเช่ทอมแคท 6.0.32 ด้วย

1.4.4 ศึกษาศึกษาแนวิธีการทดสอบแบบวีคมิวเทชัน โดยเรียนรู้จากงานวิจัยที่ผ่านมา [10 - 13] เกี่ยวกับวิธีการทดสอบแบบวีคมิวเทชัน

1.4.5 ศึกษาตัวดำเนินการมิวเทชันของภาษาบีเพล โดยวิเคราะห์ตัวดำเนินการมิวเทชันในงานวิจัย [8] ที่ได้กำหนดขึ้นจากนั้น นำมาประยุกต์ใช้กับการทดสอบแบบวีคมิวเทชัน

1.4.6 ออกแบบเครื่องมือที่ใช้ในการทดสอบ พิจารณาถึงความเป็นไปได้ที่จะสร้างเครื่องมือในเรื่องของการประหยัดเวลาของการประมวลผลโปรแกรมในการทดสอบ และการนำตัวดำเนินการมิวเทชันมาประยุกต์ใช้กับการทดสอบแบบวีคมิวเทชัน

1.4.7 จัดเตรียมโปรแกรมบีเพลเพื่อใช้ในการทดสอบ ซึ่งจะอ้างอิงจากงานวิจัย [7] และ [8] เพื่อที่จะเปรียบเทียบผลลัพธ์ของการประมวลผลการทดสอบแบบมิวเทชันโดยทั่วไปและแบบวีคมิวเทชัน

1.4.8 พัฒนาเครื่องมือที่ได้ออกแบบไว้ โดยพิจารณาว่าเครื่องมือที่พัฒนาขึ้นมานั้นสามารถทำงานได้ตามต้องการหรือไม่ เช่น เครื่องมือสามารถแสดงจำนวนมิวแตนท์ที่ถูกกำจัดได้หรือไม่ หรือสามารถคำนวนคะแนนมิวเทชันได้อย่างถูกต้องหรือไม่ เป็นต้น ถ้าเครื่องมือยังทำงานไม่เป็นไปตามความต้องการ อาจจะจำเป็นต้องกลับไปวิเคราะห์ในขั้นตอนการออกแบบเครื่องมืออีกครั้งหนึ่ง

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) สามารถนำวิธีการทดสอบแบบวีคมิวเทชันชนิดต่างๆและการสร้างกรณีทดสอบสำหรับดับเบิลยูเอสบีเพล ไปปรับใช้กับเว็บเซอร์วิซอื่นๆ หรือภาษาอื่นๆ ได้ ตัวอย่างเช่น PHP เป็นต้น
2) เครื่องมือที่พัฒนาช่วยลดภาระให้นักทดสอบในการสร้างมิวแตนท์ด้วยวิธีวีคมิวเทชันกับภาษาบีเพลได้

1.6 บทความวิชาการที่ได้รับการตีพิมพ์

ในการวิจัยนี้ ผู้วิจัยมีผลงานทางวิชาการร่วมกับคณะผู้วิจัย ซึ่งเป็นบทความวิชาการในระดับนานาชาติ รวมเป็น 2 บทความ ดังนี้

1) บทความวิชาการเรื่อง “A Weak Mutation Testing Framework for WS-BPEL” ซึ่งได้รับการคัดเลือกเพื่อนำเสนอและตีพิมพ์ในงาน “การประชุมวิชาการร่วมสาขาวิทยาการคอมพิวเตอร์และวิศวกรรมซอฟต์แวร์ ครั้งที่ 8 (The 8th International Joint Conference on Computer Science and Software Engineering: JCSSE 2011)” ระหว่างวันที่ 11 - 13 พฤษภาคม 2554 ณ คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล วิทยาเขตศาลายา นครปฐม ประเทศไทย

2) บทความวิชาการเรื่อง “WeMuTe - A Weak Mutation Testing Tool for WS-BPEL” ซึ่งได้รับการคัดเลือกเพื่อนำเสนอและตีพิมพ์ในงาน “International MultiConference of Engineers and Computer Scientists 2012: IMECS 2012” ระหว่างวันที่ 14 - 16 มีนาคม 2555 ณ โรงแรมรอยัลการ์เด้น เกาลูน ฮ่องกง ประเทศจีน

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ภาษาเอ็กซ์เอ็มแอล [2]

เอ็กซ์เอ็มแอลเป็นภาษามาร์กอัปสำหรับการใช้งานทั่วไป ถูกพัฒนาโดยเวิลด์ไวด์เว็บคอนซอร์เทียม (World Wide Web Consortium – W3C) [14] โดยมีจุดประสงค์เพื่อใช้เป็นสิ่งที่เอาไว้ติดต่อกันในระบบที่มีความแตกต่างกัน (เช่นใช้คอมพิวเตอร์มี่มีระบบปฏิบัติการคนละตัว หรืออาจจะเป็นคนละโปรแกรมประยุกต์ที่มีความต้องการสื่อสารข้อมูลถึงกัน) นอกจากนี้ยังเพื่อเป็นพื้นฐานในการสร้างภาษามาร์กอัปเฉพาะทางอีกขั้นหนึ่ง เอ็กซ์เอ็มแอลพัฒนามาจากเอสจีเอ็มแอล (SGML - Standard Generalized Markup Language) [15] โดยดัดแปลงให้มีความซับซ้อนลดน้อยลง เอ็กซ์เอ็มแอลใช้ในแลกเปลี่ยนข้อมูลระหว่างเครื่องคอมพิวเตอร์ที่แตกต่างกันและเน้นการแลกเปลี่ยนข้อมูลผ่านอินเทอร์เน็ต

เอ็กซ์เอ็มแอลเป็นภาษาที่ใช้เน้นส่วนที่เป็นข้อมูล โดยสามารถกำหนดชื่อแท็กและ ชื่อแอตทริบิวต์ (Attribute) ได้ตามความต้องการของผู้สร้างเอกสารเอ็กซ์เอ็มแอล โดยเอกสารนั้นจะต้องมีลักษณะที่จัดแต่งมาดีแล้ว (Well-formed) ส่วนการนิยามโครงสร้างของเอกสาร (DTD – Data Type Definition) และสคีมาจะมีหรือไม่มีก็ได้ขึ้นอยู่กับว่ามีผู้ใช้เอกสารนั้นมากน้อยแค่ไหน เอกสารเอ็กซ์เอ็มแอลเป็นแค่ไฟล์ข้อความชนิดหนึ่งที่มีแท็กเปิดและแท็กปิดครอบข้อมูลไว้ตรงกลางเท่านั้น ทำให้เอกสารเอ็กซ์เอ็มแอลถูกใช้ในการติดต่อกับระบบที่ต่างกันได้ เนื่องจากความง่ายในการสร้างเอกสาร การนำเอกสารเอ็กซ์เอ็มแอลไปใช้งานจะสนใจแต่ข้อมูลที่ถูกเน้นด้วยแท็กมากกว่า

เนื่องจากเอกสารเอ็กซ์เอ็มแอลสามารถกำหนดชื่อแท็กและชื่อแอตทริบิวต์ได้ตามความต้องการของผู้สร้างเอกสาร ทำให้ในการเน้นข้อมูลใดข้อมูลหนึ่งสามารถมีเอกสารเอ็กซ์เอ็มแอลหลายรูปแบบ (ผู้เขียนอาจใช้ชื่อแท็กต่างกันทั้งที่สื่อความหมายไปที่สิ่งเดียวกัน) หากว่าเอกสารเอ็กซ์เอ็มแอลนั้นถูกนำไปใช้ติดต่อกับระบบอื่นๆ อาจทำให้สื่อความหมายไม่ตรงกัน ดังนั้นจึงต้องมีการกำหนดรูปแบบที่เป็นมาตรฐานขึ้น (ตกลงรูปแบบระหว่างกัน) โดยการนิยามรูปแบบข้อมูลและสคีมาจะเป็นตัวกำหนดว่าเอกสารเอ็กซ์เอ็มแอลนั้นจะต้องมีแท็กอะไรบ้าง ภายในแท็กนั้นจะมี

ข้อมูลหรือแอตทริบิวต์อะไรได้บ้าง โดยการนิยามรูปแบบข้อมูลจะต่างกับสคีมาตรงที่สคีมาเป็นเอกสารเอ็กซ์เอ็มแอลด้วย

เอ็กซ์เอ็มแอลเป็นภาษาที่ไม่มีรูปแบบโครงสร้างที่กำหนดไว้ล่วงหน้า ซึ่งเป็นข้อดีที่ทำให้โปรแกรมประยุกต์ใดๆ ก็ได้สามารถใช้งานได้โดยทำตามข้อกำหนดที่ได้ตกลงกันไว้ล่วงหน้า ถึงแม้ว่าเอ็กซ์เอ็มแอลไม่มีข้อกำหนดล่วงหน้าถึงโครงสร้างข้อมูลต่างๆ แต่ก็มีรูปแบบที่เป็นที่ตกลงกัน ดังนี้

1) อีลีเมนต์ประกอบไปด้วย แท็กเปิด ข้อมูล และแท็กปิด ยกตัวอย่างเช่น

<student>Somchai</student>

ในที่นี้ <student> คือแท็กเปิด Somchai คือข้อมูล และ </student> คือแท็กปิด โดยแท็กปิดนั้นจะต้องมีชื่อเหมือนแท็กเปิดของมันแต่ตามหลังจากเครื่องหมาย '/' เอ็กซ์เอ็มแอลนั้นเป็นภาษาที่ตัวอักษรตัวเล็กตัวใหญ่มีความแตกต่างกัน (Case Sensitive) ดังนั้นการที่เราเขียนว่า <student> กับ <Student> จึงถือว่าเป็นคนละแท็กกัน นอกจากนี้แล้วอีลีเมนต์ในเอ็หกซ์เอ็มแอลสามารถบรรจุอยู่ในอีลีเมนต์อื่นๆ ได้ยกตัวอย่าง เช่น

<student>

<name>Somchai</name>

<id>555555</id>

</student>

2) อีลีเมนต์ <name> ที่บรรจุอยู่ภายในอีลีเมนต์ <student> อีลีเมนต์ไม่สามารถคร่อมกันได้ เช่น

<student>

<name>Somchai

<id></name>555555</id>

</student>

3) อีลีเมนต์สามารถว่างแบบนี้ได้ เช่น <book></book> นอกจากนี้ยังมีข้อยกเว้นสำหรับแท็กว่างจะเป็นแท็กที่ไม่ต้องมีแท็กปิดได้โดยสามารถเขียน <book/> ซึ่งมีความหมายเหมือนกัน

4) ภายในอีลีเมนต์นั้นยังมีสิ่งที่เรียกว่า แอตทริบิวต์ ด้วยโดยมีรูปแบบดังนี้

<student name=”Somchai”></student>

<student name='Somchai'></student>

จะเห็นว่าทั้งสองแบบมีความเหมือนกันแตกต่างกันเล็กน้อยคือใช้เครื่องหมาย " กับเครื่องหมาย ' ซึ่งสามารถใช้ได้ทั้งคู่

5) การประกาศเอ็กซ์เอ็มแอลโดยส่วนต่างๆ ของเอ็กซ์เอ็มแอล คือ โหนด (Node) ซึ่งปรากฏบน เอ็กซ์เอ็มแอลทุกฉบับคือ การประกาศโดยจะมีลักษณะเหมือนแท็กแต่มีเครื่องหมาย ? อยู่ด้วย และในเอกสารเอ็กซ์เอ็มแอลจะต้องมีโหนดนี้ทุกฉบับ ดังตัวอย่าง

<?xml version=”1.0” ?>

<student>

<name>Somchai</name>

<id>555555</id>

</student>

2.1.2 **เอ็กซ์เอ็มแอลสคีมา** [16]

เอ็กซ์เอ็มแอลสคีมา คือ วิธีการที่จะใช้จำแนกชนิดของเอกสารเอ็กซ์เอ็มแอลโดยจะบอกถึงโครงสร้างและชนิดของข้อมูลในเอกสารเอ็กซ์เอ็มแอลแต่ละชนิด โดยตัวสคีมาเองมีหน้าตาเหมือนเนื้อหาเอ็กซ์เอ็มแอลตามปรกติ

การนำเอกสารเอ็กซ์เอ็มแอลไปใช้นั้นไม่ว่าจะนำไปประมวลผลในลักษณะใดก็ตาม เอกสารเหล่านั้นจะต้องอยู่ในรูปแบบที่พร้อมจะนำไปประมวลผล คือ มีคุณสมบัติที่ถูกหลักไวยากรณ์ อย่างไรก็ตามไวยากรณ์นั้นเป็นคุณสมบัติทางกายภาพเท่านั้น แต่ไม่ได้ยืนยันคุณสมบัติทางตรรกะ กล่าวคือ อีลีเมนต์ต่างๆ อาจไม่ได้เป็นไปตามความมุ่งหมายของเอกสารนั้นๆ เช่น มีอีลีเมนต์หรือแอตทริบิวต์บังคับที่ขาดหายไป หรือค่าของอีลีเมนต์ไม่อยู่ในรูปแบบที่ควรจะเป็น เป็นต้น เช่นนี้แล้วหากนำเอกสารไปประมวลผล ก็จะได้ผลลัพธ์ที่ไม่ถูกต้อง การตรวจสอบความถูกต้องของโครงสร้างและข้อมูลในเอกสารเอ็กซ์เอ็มแอล ก็คือ การทำการตรวจสอบ (Validation) ดังนั้น หาก

เอกสารใดผ่านการการตรวจสอบแล้วว่าถูกต้อง เอกสารนั้นก็ได้ชื่อว่ามีคุณสมบัติที่มีความถูกต้องแล้วนั่นเอง

หากต้องการใช้เอกสารเอ็กซ์เอ็มแอลความถูกต้องของโครงสร้างเอกสารจึงมีความสำคัญเป็นอย่างมาก ซึ่งการตรวจสอบความถูกต้องได้นั้น โดยการอาศัยกฎหรือมาตรการที่ใช้ระบุความเป็นไปได้ของอีลีเมนต์ในเอกสารหนึ่งๆ ซึ่งกฎนั้นก็คือ สคีมานั่นเอง ดังนั้นเมื่อถึงกระบวนการตรวจสอบ สคีมาจึงถูกนำมาใช้เพื่ออ้างอิงในการตรวจสอบ (กระทำผ่านเครื่องมือต่างๆ) ดังตัวอย่าง การใช้ในงานฐานข้อมูลเชิงสัมพันธ์ (Relational Database) เอ็กซ์เอ็มแอลสคีมาก็เหมือนกับสคีมาของฐานข้อมูล ซึ่งเป็นตัวบอกว่าตารางหนึ่งๆ ประกอบไปด้วยฟิลด์อะไรได้บ้างและฟิลด์นั้นๆ มีรูปแบบของข้อมูลเป็นอย่างไร เป็นต้น

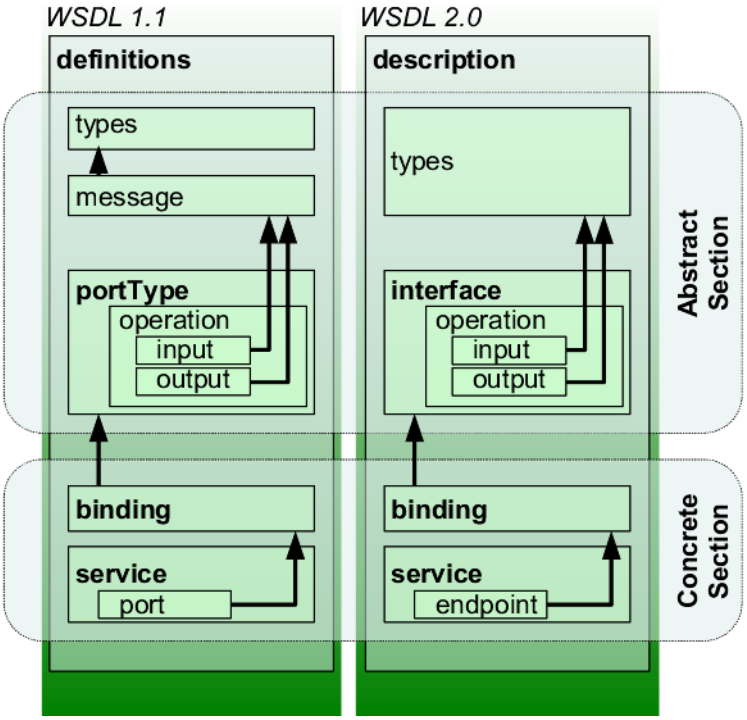
เทคโนโลยีของสคีมานั้นมีหลายประเภท เอ็กซ์เอ็มแอลสคีมาเป็นเทคโนโลยีที่เป็นที่นิยมในปัจจุบัน ซึ่งพัฒนาและปรับปรุงข้อกำหนดโดยดับเบิลยูสามซี นอกจากนี้ยังมีข้อกำหนดของเอ็กซ์เอ็มแอล และเอ็กซ์เอ็มแอลเนมสเปซ เนื่องจากเอ็กซ์เอ็มแอลสคีมานั้นเป็นเทคโนโลยีที่ใหญ่และซับซ้อนมาก จากภาพที่ 2.1 คือตัวอย่างของเอกสารเอ็กซ์เอ็มแอลสคีมา

2.1.3 **ดับเบิลยูเอสดีแอลหรือวิซเดิล** [4]

ดับเบิลยูเอสดีแอลเป็นภาษาเอ็กซ์เอ็มแอลซึ่งเป็นรูปแบบในการอธิบายเว็บเซอร์วิซ ซึ่งเวอร์ชันปัจจุบันจะเป็นเวอร์ชัน 2.0 และจากเวอร์ชัน 1.1 นั้นได้รับการอนุญาตโดยเวิลด์ไวด์เว็บคอนซอร์เทียม คือ องค์กรระหว่างประเทศทำหน้าที่จัดระบบมาตรฐานที่ใช้งานบนเวิลด์ไวด์เว็บ (WWW หรือ W3) โดยมีจุดมุ่งหมายที่จะเป็นแกนนำทางด้านพัฒนาโพรโทคอล และวิธีการใช้งานสำหรับเวิลด์ไวด์เว็บทั้งหมด นอกจากนี้ทางดับเบิลยูสามซีมีการบริการทางการศึกษาการพัฒนาซอฟต์แวร์ และเปิดให้ใช้ฟอรัมในการปรึกษาเกี่ยวกับเรื่องเว็บ WSDL 1.2 ได้ถูกเปลี่ยนเป็น WSDL 2.0 เนื่องจากการเปลี่ยนแปลงอย่างมากมายจาก WSDL ในเวอร์ชัน 1.1 ดังภาพที่ 2.2 และตัวอย่างเอกสารดับเบิลยูเอสดีแอลในภาพที่ 2.3

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns="http://kontentblue.com/ns/po"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      targetNamespace="http://kontentblue.com/ns/po"
      elementFormDefault="unqualified" >
      <xsd:element name="purchaseOrder" type="PurchaseOrderType"/>
      <xsd:element name="comment" type="xsd:string"/>
      <xsd:complexType name="PurchaseOrderType">
      <xsd:sequence>
      <xsd:element name="shipTo" type="addressType"/>
      <xsd:element name="billTo" type="addressType"/>
      <xsd:element ref="comment" minOccurs="0"/>
      <xsd:element name="items" type="Items"/>
      </xsd:sequence>
      <xsd:attribute name="orderDate" type="xsd:date"/>
      </xsd:complexType>
      <xsd:complexType name="addressType">
      <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
      <xsd:element name="street" type="xsd:string"/>
      <xsd:element name="city" type="xsd:string"/>
      <xsd:element name="state" type="xsd:string" minOccurs="0"/>
      <xsd:element name="zip" type="xsd:string"/>
      <xsd:element name="country" type="xsd:string"/>
      </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="Items">
      <xsd:sequence>
      <xsd:element name="item" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
      <xsd:sequence>
      <xsd:element name="productName" type="xsd:string"/>
      <xsd:element name="quantity">
      <xsd:simpleType>
      <xsd:restriction base="xsd:positiveInteger">
      <xsd:maxExclusive value="100"/>
      </xsd:restriction>
      </xsd:simpleType>
      </xsd:element>
      <xsd:element name="USPrice" type="xsd:decimal"/>
      <xsd:element ref="comment" minOccurs="0"/>
      <xsd:element name="shipDate" type="xsd:date" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="partNum" type="SKU" use="required"/>
      </xsd:complexType>
      </xsd:element>
      </xsd:sequence>
      </xsd:complexType>
      <!-- Stock Keeping Unit, a code for identifying products -->
      <xsd:simpleType name="SKU">
      <xsd:restriction base="xsd:string">
      <xsd:pattern value="\d{3}-[A-Z]{2}"/>
      </xsd:restriction>
      </xsd:simpleType>
</xsd:schema>
```

ภาพที่ 2.1 ตัวอย่างเอกสารเอ็กซ์เอ็มแอลสคีมา



ภาพที่ 2.2 การเปรียบเทียบอีลีเมนต์ใน WSDL 1.1 และ WSDL 2.0 [17]

```
<?xml version="1.0" encoding="UTF-8"?>
<description xmlns="http://www.w3.org/ns/wsdl"
             xmlns:tns="http://www.tmsws.com/wsdl20sample"
             xmlns:whttp="http://schemas.xmlsoap.org/wsdl/http/"
             xmlns:wsoap="http://schemas.xmlsoap.org/wsdl/soap/"
             targetNamespace="http://www.tmsws.com/wsdl20sample">
   <types>
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
                 xmlns="http://www.tmsws.com/wsdl20sample"
                 targetNamespace="http://www.example.com/wsdl20sample">
         </xs:element>
      </xs:schema>
   </types>
   <interface name="RESTfulInterface">
      <fault name="ClientError" element="tns:response"/>
      …
      <operation name="Get" pattern="http://www.w3.org/ns/wsdl/in-out">
         <input messageLabel="In" element="tns:request"/>
         <output messageLabel="Out" element="tns:response"/>
      </operation>
   </interface>

   <binding name="RESTfulInterfaceHttpBinding" interface="tns:RESTfulInterface"
            type="http://www.w3.org/ns/wsdl/http">
      <operation ref="tns:Get" whttp:method="GET"/>
      …
   </binding>
   <binding name="RESTfulInterfaceSoapBinding" interface="tns:RESTfulInterface"
            type="http://www.w3.org/ns/wsdl/soap"
            wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP/"
            wsoap:mepDefault="http://www.w3.org/2003/05/soap/mep/request-response">
   </binding>

   <service name="RESTfulService" interface="tns:RESTfulInterface">
      <endpoint name="RESTfulServiceHttpEndpoint"
                binding="tns:RESTfulInterfaceHttpBinding"
                address="http://www.example.com/rest/"/>
   </service>
</description>
```

ภาพที่ 2.3 ตัวอย่างเอกสารของดับเบิลยูเอสดีแอล

อิลีเมนต์ในโครงสร้างของเอกสารดับเบิลยูเอสดีแอลเวอร์ชัน 1.1 เปรียบเทียบกับเวอร์ชัน 2.0 (WSDL 1.1/WSDL 2.0) จะประกอบด้วยอีลีเมนต์ดังต่อไปนี้

1) Service/Service – เป็นอีลีเมนต์ซึ่งเปรียบเสมือนเป็นตัวบรรจุกลุ่มของการทำงานในระบบ ซึ่งเป็นส่วนที่กำหนด URL ของเว็บเซอร์วิซ ดังตัวอย่างภาพที่ 2.4

```
<service name='plusService'>
  <port name='plusPort' binding='plusBinding'>
    <soap:address
location='http://truehits.net/faq/webmaster/webservice/plus_server.php'
/>
  </port>
</service>
```

ภาพที่ 2.4 ตัวอย่างอีลีเมนต์ service

2) Port/Endpoint – อีลีเมนต์นี้ไม่ได้มีหน้าที่นอกเหนือจากการระบุที่อยู่หรือจุดเชื่อมต่อกับเว็บเซอร์วิซ ดังตัวอย่างภาพที่ 2.5

```
<port name='plusPort' binding='plusBinding'>
       <soap:address
       location='http://truehits.net/faq/webmaster/webservice/plus_
       server.php'/>
</port>
```

ภาพที่ 2.5 ตัวอย่างอีลีเมนต์ port

3) Binding/Binding – อีลีเมนต์จะจำเพาะไปที่ส่วนเชื่อมต่อโดยการกำหนดรูปแบบการสัมพันธ์กับโซปและการขนส่งโดยใช้โพรโตคอลของโซป ในส่วนนี้ยังมีการกำหนดหน้าที่การทำงานอีกด้วย ดังตัวอย่างภาพที่ 2.6

```
<binding name='plusBinding' type='tns:plusPortType'>
       <soap:binding style='rpc'
       transport='http://schemas.xmlsoap.org/soap/http'/>
       <operation name='plus'>
         <soap:operation soapAction='urn:xmethods-delayed-quotes#plus'/>
         <input>
         <soap:body use='encoded' namespace='urn:xmethods-delayed-quotes'
         encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
         </input>
         <output>
         <soap:body use='encoded' namespace='urn:xmethods-delayed-quotes'
         encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
         </output>
       </operation>
</binding>
```

ภาพที่ 2.6 ตัวอย่างอีลีเมนต์ binding

4) PortType/Interface – อีลีเมนต์ <PortType> ถูกเปลี่ยนเป็น <Interface> ในเวอร์ชัน 2.0 มีการกำหนดในเว็บเซอร์วิซนั้นๆ ว่า มีการดำเนินการอะไรบ้างที่สามารถทำได้ และข้อความที่จะต้องถูกใช้ในการดำเนินการนั้น ทั้งยังประกาศ messages ที่ใช้เป็นอินพุตและเอาต์พุตของโอเปอร์เรชัน ดังตัวอย่างภาพที่ 2.7

```
<portType name='plusPortType'>
 <operation name='plus'>
  <input message='tns:plusRequest'/>
  <output message='tns:plusResponse'/>
 </operation>
</portType>
```

ภาพที่ 2.7 ตัวอย่างอีลีเมนต์ portType

5) Operation/Operation – แต่ละอีลีเมนต์ <Operation> สามารถเปรียบเสมือนเป็นเมท็อดหรือฟังก์ชันในการเรียกใช้โปรแกรม โดยเว็บเซอร์วิซหนึ่งๆ สามารถมีเมท็อดได้หลายๆ เมท็อด ดังตัวอย่างภาพที่ 2.8

```
<operation name='plus'>
  <input message='tns:plusRequest'/>
  <output message='tns:plusResponse'/>
</operation>
```

ภาพที่ 2.8 ตัวอย่างอีลีเมนต์ operation

6) Message/N.A. – โดยทั่วไปแล้ว <Message> นั้นจะสอดคล้องตาม <Operation> ซึ่งจะมีข้อมูลในการที่จะกระทำการหนึ่งๆ แต่ละข้อความจะประกอบด้วยส่วนที่เป็นตรรกะมากกว่าหรือหนึ่งขึ้นไป ดังตัวอย่างภาพที่ 2.9

```
<message name="getTermRequest">
   <part name="term" type="xs:string"/>
</message>
…
<message name="getTermResponse">
   <part name="value" type="xs:string"/>
</message>
```

ภาพที่ 2.9 ตัวอย่างอีลีเมนต์ message

7) Types/Types – วัตถุประสงค์ของอีลีเมนต์ <Types> ในดับเบิลยูเอสดีแอล ก็เพื่ออธิบายถึงข้อมูลในเอกสารดับเบิลยูเอสดีแอลซึ่งเอ็กซ์เอ็มแอลสคีมาจะถูกใช้เพื่อเหตุผลนั้น บ่งบอกถึงชนิดข้อมูลที่ใช้ในดับเบิลยูเอสดีแอล ดังตัวอย่างภาพที่ 2.10

```
<types>
     <schema targetNamespace="http://example.com/stockquote.xsd"
     xmlns="http://www.w3.org/2000/10/XMLSchema">
     <element name="TradePriceRequest">
        <complexType>
          <all>
            <element name="tickerSymbol" type="string"/>
          </all>
        </complexType>
     </element>
     <element name="TradePrice">
        <complexType>
           <all>
              <element name="price" type="float"/>
           </all>
     </complexType>
     </element>
     </schema>
</types>
```

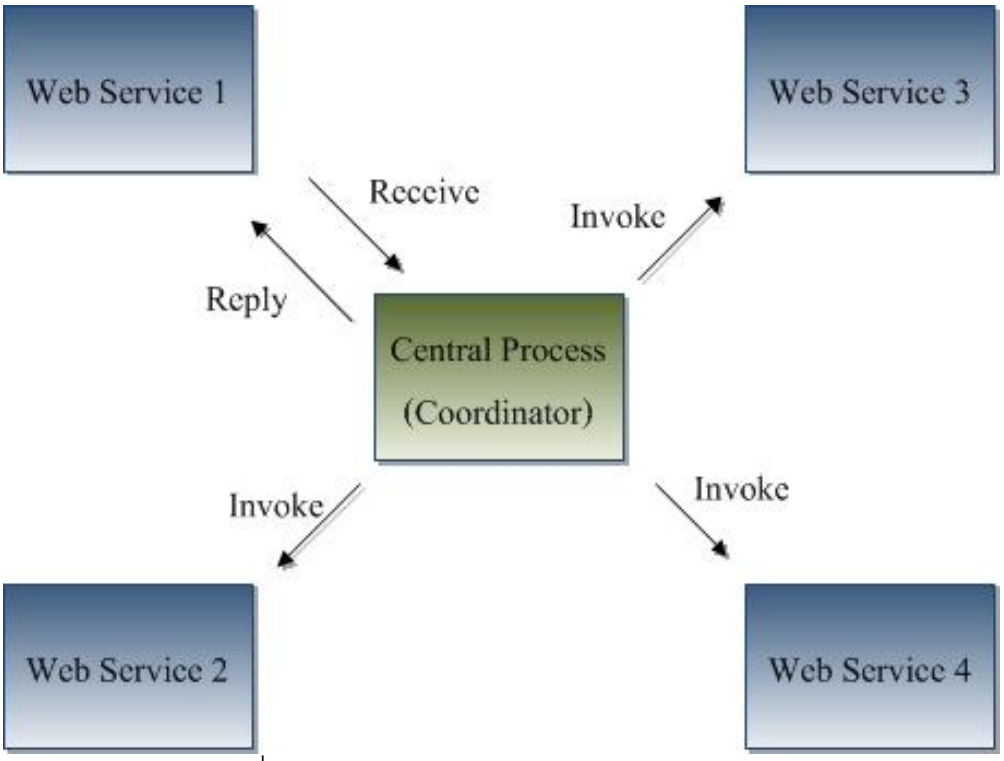
ภาพที่ 2.10 ตัวอย่างอีลีเมนต์ types

2.1.4 การประสานและการทำงานร่วมกันของเว็บเซอร์วิซ [18]

วิธีการพัฒนากระบวนการทางธุรกิจโดยมากจะใช้เทคโนโลยีเว็บเซอร์วิซที่ประกาศโดยใช้มาตรฐานดับเบิลยูเอสดีแอลมาประกอบเป็นกระบวนการทางธุรกิจใหม่โดยมีวิธีการ 2 วิธี คือ

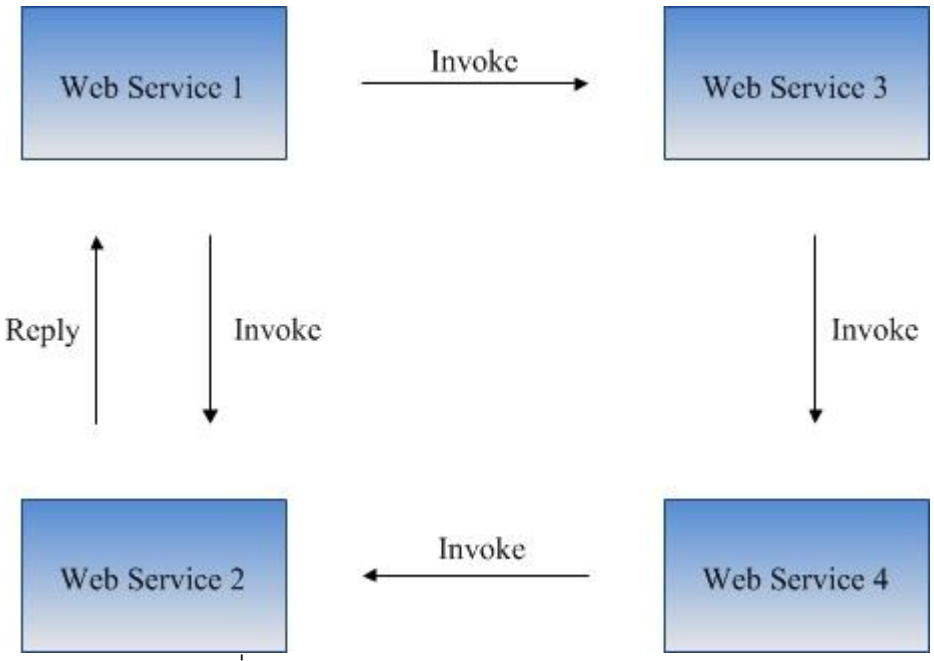
1) การประสาน (Orchestration) คือ การพัฒนากระบวนการทางธุรกิจโดยมีกระบวนการตัวกลาง (Central Process) ที่จะทำหน้าที่จัดการกับธุรกรรมทั้งหมดโดยจะส่งงานไปให้เว็บเซอร์วิซอื่นทำการประมวลผลและรับผลลัพธ์กลับมาพร้อมทั้งควบคุมการลำดับการทำงานของเว็บเซอร์วิซ ดังภาพที่ 2.11 ซึ่งกระบวนการตัวกลางนี้อาจกำหนดให้เป็นเว็บเซอร์วิซ

ใหม่อีกตัวหนึ่ง การประกอบเว็บเซอร์วิซแบบนี้โดยมากจะพัฒนาโดยใช้ภาษาบีเพล



ภาพที่ 2.11 ตัวอย่างแสดงการประสานเว็บเซอร์วิซ [18]

2) การทำงานร่วมกัน (Choreography) คือ การพัฒนาธุรกิจแบบทำงานร่วมกัน (Business Collaboration) ในกรณีนี้จะไม่มีตัวกลางที่คอยควบคุมอยู่ แต่เว็บเซอร์วิซจะทราบเองว่าเมื่อไหร่ที่จะต้องทำการประมวลผลและส่งผลลัพธ์ไปยังเว็บเซอร์วิซใดโดยการส่งข่าวสารระหว่างเว็บเซอร์วิซ ดังภาพที่ 2.12 เว็บเซอร์วิซทุกตัวที่เกี่ยวข้องกับการทำงานร่วมกันในธุรกิจจะต้องทราบถึงกระบวนการธุรกิจที่ทำความร่วมมือกันอยู่ เช่น รูปแบบของข่าวสารโอเปอร์เรชันที่ต้องประมวลผล หรือเวลาที่ต้องส่งข่าวสาร เป็นต้น ซึ่งแตกต่างกับกรณีของการทำงานแบบประสาน ซึ่งเว็บเซอร์วิซแต่ละตัวไม่ทราบรายละเอียดของกระบวนการทางธุรกิจเลย



ภาพที่ 2.12 ตัวอย่างการทำงานร่วมกันของเว็บเซอร์วิซ [18]

การประกอบเว็บเซอร์วิซแบบประสานมีข้อเด่นกว่าแบบการทำงานร่วมกัน คือ

1) การประสานงานกันระหว่างเว็บเซอร์วิซสามารถควบคุมโดยตัวประสานงานกลาง
2) เว็บเซอร์วิซแต่ละตัวสามารถถูกเรียกใช้ในกระบวนการทางธุรกิจ โดยไม่จำเป็นต้องทราบว่าเป็นส่วนหนึ่งของกระบวนการนั้นๆ
3) สามารถปรับเปลี่ยนกระบวนการนั้นสามารถทำได้ง่าย

2.1.5 ภาษาบีเพล [5]

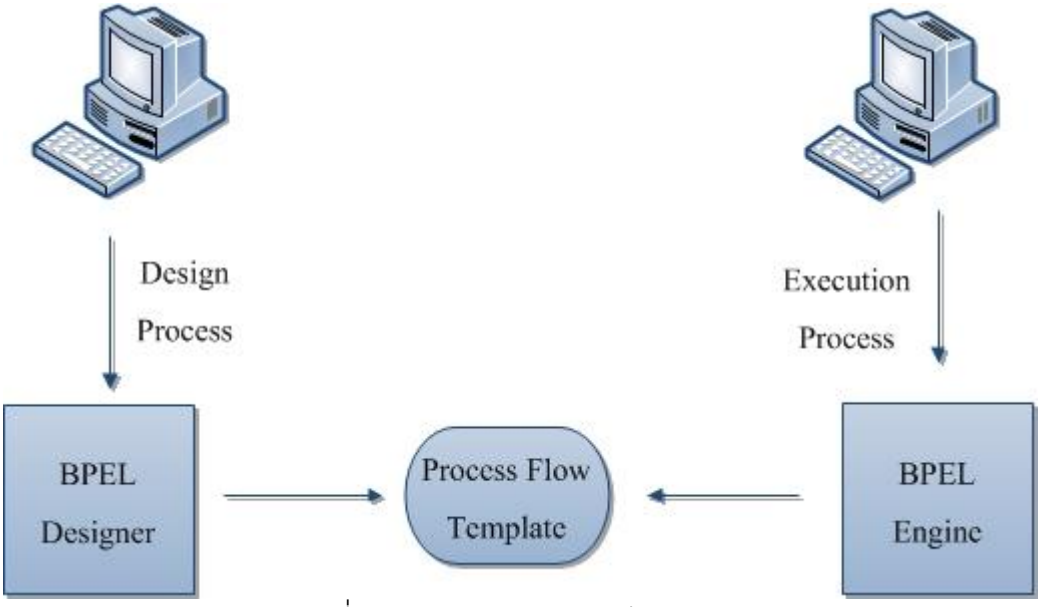
บีเพล เป็นภาษาเอ็กซ์เอ็มแอลที่ใช้พัฒนากระบวนการทางธุรกิจ (Business Process) จากเว็บเซอร์วิซที่นิยามโดยใช้ดับเบิลยูเอสดีแอล ทั้งนี้ก่อนที่จะมีภาษาบีเพล แต่ละบริษัทผู้ผลิตต่างก็มีรูปแบบของการเขียนกระบวนการทางธุรกิจที่แตกต่างกัน ดังนั้นจุดประสงค์ของการกำหนดมาตรฐานบีเพลก็เพื่อนิยามมาตรฐานกลางสำหรับการเขียนกระบวนการทางธุรกิจ โดยใช้แพลตฟอร์มที่เป็นเว็บเซอร์วิซ

บีเพลถูกพัฒนามาจากภาษาที่ใช้ในการพัฒนากระแสงาน (Workflow) 2 ภาษา คือ WSFL (Web Services Flow Language) และ XLANG โดยได้กำหนดเป็นเวอร์ชันแรกเมื่อเดือน

สิงหาคมปี 2002 ซึ่งต่อมาทางโอเอซิสได้ประกาศให้ BPEL4WS 1.1 เป็นมาตรฐานสำหรับการพัฒนากระบวนการทางธุรกิจบนเว็บเซอร์วิซเวอร์ชันล่าสุดของบีเพล คือ WSBPEL 2.0

บีเพลจะมีองค์ประกอบหลักที่เกี่ยวข้อง 3 ส่วนดังภาพที่ 2.13 คือ

1) เครื่องมือแบบบีเพล (BPEL Designer) เป็นเครื่องมือที่ให้ผู้เชี่ยวชาญด้านกระบวนการธุรกิจสามารถจำลอง Business Process โดยใช้สัญลักษณ์กราฟฟิกเพื่อสร้างไฟล์แผ่นแบบกระแสกระบวนการ โดยทั่วไปเครื่องมือเหล่านี้จะอิงตามมาตรฐาน BPMN (Business Process Modeling Notation) ในการเขียนสัญลักษณ์
2) เครื่องประมวลผลบีเพล (BPEL Engine) เป็นตัวประมวลผลไฟล์แผ่นแบบกระแสกระบวนการตามมาตรฐานบีเพล โดยจะทำงานต่างๆเช่น เรียกใช้เว็บเซอร์วิซ กำหนดเนื้อหาของข้อมูล จัดการข้อผิดพลาด หรือ ควบคุมลำดับการทำงาน โดยทั่วไปตัวจักรบีเพลจะทำงานร่วมกับเครื่องบริการโปรแกรมประยุกต์ (Application Server)
3) แผ่นแบบกระแสกระบวนการ (Process Flow Template) เป็นไฟล์ที่ระบุกระบวนการทางธุรกิจตามข้อกำหนดของบีเพลโดยจะเป็นไฟล์ที่ถูกสร้างมาจากตัวออกแบบบีเพล และจะใช้ตัวจักรบีเพลในการประมวลผล



ภาพที่ 2.13 องค์ประกอบของบีเพล [18]

โปรแกรมบีเพลจะใช้แท็กเอ็กซ์เอ็มแอล ในการประกอบเว็บเซอร์วิซเพื่อสร้างกระบวนการทางธุรกิจ โดยกระบวนการบีเพลที่สร้างขึ้นมาสามารถจะแสดงออกมาเป็นเซอร์วิซที่นิยามโดยดับเบิลยูเอสดีแอล และสามารถเรียกใช้กระบวนการบีเพลนี้ได้เหมือนการเรียกเว็บเซอร์วิซ โดยทั่วไปบีเพลจะมีชุดของคำสั่งที่ระบุภารกิจพื้นฐานที่ใช้การประกอบเว็บเซอร์วิซ ดังนี้

1) <invoke> - เป็นคำสั่งเพื่อให้กระบวนการทางธุรกิจ เรียกใช้โอเปอร์เรชัน (Operation) ภายในแท็ก portType ที่นิยามอยู่ในดับเบิลยูเอสดีแอลของเว็บเซอร์วิซ ดังตัวอย่างภาพที่ 2.14

```
<invoke inputVariable="flightReservationRequest"
       name="checkFlightReservation" operation="doFlightReservation"
       outputVariable="flightReservationResponse"
       partnerLink="FlightReservationService"
       portType="wsdl3:FlightReservationServiceImpl"
      wpc:displayName="Check Flight Reservation" wpc:id="10">
       <target linkName="link4"/>
       <source linkName="link5"/>
</invoke>
```

ภาพที่ 2.14 ตัวอย่างแอ็คติวิตี invoke

2) <receive> - คำสั่งเพื่อให้กระบวนการทางธุรกิจหยุดรอข่าวสารที่จะมาถึง ดังตัวอย่างภาพที่ 2.15

```
<receive partnerLink="shipping"
       portType="lns:shippingCallbackPT"
       operation="sendSchedule" variable="shippingSchedule">
       <documentation>Arrange Logistics</documentation>
       <sources>
       <source linkName="ship-to-scheduling" />
       </sources>
</receive>
```

ภาพที่ 2.15 ตัวอย่างแอ็คติวิตี receive

3) <reply> - คำสั่งเพื่อให้กระบวนการทางธุรกิจส่งข่าวสารเพื่อตอบกลับข่าวสารที่ได้รับมา ดังตัวอย่างในภาพที่ 2.16

```
<reply partnerLink="purchasing" portType="lns:purchaseOrderPT"
      operation="sendPurchaseOrder" variable="Invoice">
      <documentation>Invoice Processing</documentation>
</reply>
```

ภาพที่ 2.16 ตัวอย่างแอ็คติวิตี reply

4) <assign> - คำสั่งเพื่อคัดลอกข้อมูลจากตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่ง ดังตัวอย่างในภาพที่ 2.17

```
<assign>
   <!-- copy from integer expression to the variable -->
   <copy>
      <from expression="100"/>
      <to variable="output" part="payload"
query="/p:result/p:quantity"/>
   </copy>
</assign>
```

ภาพที่ 2.17 ตัวอย่างแอ็คติวิตี assign

5) <throw> - คำสั่งเพื่อระบุข้อผิดพลาดที่เกิดขึ้น ดังตัวอย่างในภาพที่ 2.18

```
<throw faultName="WrongEmployeeName" />

<throw faultName="trv:TicketNotApproved" faultVariable="TravelFault"/>
```

ภาพที่ 2.18 ตัวอย่างแอ็คติวิตี throw

6) <wait> - คำสั่งเพื่อให้กระบวนการทางธุรกิจหยุดรอตามระยะเวลาหนึ่ง ดังตัวอย่างในภาพที่ 2.19

```
<!—-duration expression-->

<wait>
  <for>'PT4H10M'</for>
</wait>

<!—-deadline expression-->

<wait>
  <until>'2004-03-18T21:00:00+01:00'</until>
</wait>
<wait>
  <until>'18:05:30Z'</until>
</wait>
```

ภาพที่ 2.19 ตัวอย่างแอ็คติวิตี wait

7) <terminate> - คำสั่งเพื่อยกเลิกกระบวนการทางธุรกิจทั้งหมด ดังตัวอย่างในภาพที่ 2.20

```
<sequence>
…
<terminate name="killClaim"/>
…

</sequence>
```

ภาพที่ 2.20 ตัวอย่างแอ็คติวิตี terminate

บีเพลยังมีชุดคำสั่งที่เป็นภารกิจแบบโครงสร้าง (Structure Task) ที่ใช้การผนวกภารกิจพื้นฐานเข้าด้วยกันเพื่อใช้ควบคุมลำดับการทำงานและสร้างกระบวนการทางธุรกิจที่ซับซ้อนขึ้น โดยมีคำสั่งต่างๆ ดังนี้

8) <sequence> - คำสั่งเพื่อนิยามการทำงานของภารกิจต่างๆแบบต่อเนื่อง ดังตัวอย่างในภาพที่ 2.21

```
<sequence>
        <!-- Invoke the web service -->
        <invoke partnerLink="AmericanAirlines"
              portType="aln:FlightAvailabilityPT"
              operation="FlightAvailability"
              inputVariable="FlightDetails" />
        <receive partnerLink="AmericanAirlines"
              portType="trv:FlightCallbackPT"
              operation="FlightTicketCallback"
              variable="FlightResponseAA" />
...
        <!-- Process the results … -->
...
        <!-- Increment the counter -->
        <assign>
          <copy>
        <from expression="bpws:getVariableData('Counter') + 1"/>
            <to variable="Counter"/>
         </copy>
        </assign>
 </sequence>
```

ภาพที่ 2.21 ตัวอย่างแอ็คติวิตี sequence

9) <if> - เป็นคำสั่งเพื่อใช้ในการเลือกทำภารกิจต่างๆ ดังตัวอย่างในภาพที่ 2.22

```
<if standard-attributes>standard-elements

      <condition expressionLanguage="anyURI"?>bool-expr</condition>

      activity

      <elseif>*<condition expressionLanguage="anyURI"?>bool-
      expr</condition>

            activity

      </elseif>

      <else>?activity</else>

</if>
```

ภาพที่ 2.22 ตัวอย่างแอ็คติวิตี if

10) <flow> - คำสั่งเพื่อระบุให้ชุดภารกิจทำงานแบบขนาน ดังตัวอย่างในภาพที่ 2.23

```
<flow standard-attributes> standard-elements

      <links>?

            <link name="NCName" />+

      </links>

      activity+

</flow>
```

ภาพที่ 2.23 ตัวอย่างแอ็คติวิตี flow

11) <switch>,<case> - คำสั่งเพื่อกำหนดให้ชุดภารกิจทำงานแบบเลือกทำ (case-switch) ตามเงื่อนไขตรรกะที่ระบุ ดังตัวอย่างในภาพที่ 2.24

```
<switch>

        <case condition="bpws:getVariableData('InsuranceAResposne',

'confirmationData','/confirmationData/Amount')
                      <= bpws:getVariableData('InsuranceBResposne',

'confirmationData','/confirmationData/Amount')">

           <!-- Select Insurance A -->
           <assign>
             <copy>
               <from variable="InsuranceAResposne" />
               <to variable="InsuranceSelectionResponse" />
             </copy>
           </assign>
        </case>
        <otherwise>
           <!-- Select Insurance B -->
           <assign>
             <copy>
               <from variable="InsuranceBResposne" />
               <to variable="InsuranceSelectionResponse" />
             </copy>
           </assign>
        </otherwise>
</switch>
```

ภาพที่ 2.24 ตัวอย่างแอ็คติวิตี switch

12) <while> - คำสั่งเพื่อกำหนดให้มีการทำงานของกลุ่มภารกิจซ้ำจนกว่าเป็นไปตามเงื่อนไขที่ระบุ ดังตัวอย่างในภาพที่ 2.25

```
<while>
 <condition> ('$moreWork' = true() ) </condition>
     <if name="gotMilk">
          <condition> ("$gotMilk" = true() ) </condition>
           <assign name="SETsomeVariables">
           </assign>
    </if>
    <if name="checkinEvent1">
          …
      </if>
      <assign>
      </asssign>
</while>
```

ภาพที่ 2.25 ตัวอย่างแอ็คติวิตี while

13) <repeatUntil> - คำสั่งเพื่อกำหนดให้ทำกิจกรรมย่อยๆ จนกว่าจะถึงเงื่อนไขที่กำหนด ดังตัวอย่างในภาพที่ 2.26

```
<repeatUntil>
  <condition
   expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:
     xpath1.0">$counter &gt; 0
  </condition>
      <sequence>
       <assign name="IncrementCounter">
        <copy>
         <from>$counter + 1</from>
         <to variable="counter"/>
        </copy>
       </assign>
       <wait name="WaitTwoSeconds">
        <for expressionLanguage=
          "urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0">
            'PT2S'
        </for>
       </wait>
      </sequence>
</repeatUntil>
```

ภาพที่ 2.26 ตัวอย่างแอ็คติวิตี repeatUntil

14) <forEach> - คำสั่งในการทำซ้ำในขอบเขตของภารกิจย่อยๆจำนวน N+1 ครั้ง โดยที่ N เป็น <finalCounterValue> ลบด้วย <startCounterValue> ดังตัวอย่างในภาพที่ 2.27

```
<forEach counterName="Counter" parallel="no">
  <startCounterValue>number(1)</startCounterValue>
  <finalCounterValue>$NoOfPassengers</finalCounterValue>
  <scope>
    <sequence>
      <!-- Invoke the web service -->
      <invoke partnerLink="AmericanAirlines"
              portType="aln:FlightAvailabilityPT"
              operation="FlightAvailability"
              inputVariable="FlightDetails" />
      <receive partnerLink="AmericanAirlines"
               portType="trv:FlightCallbackPT"
               operation="FlightTicketCallback"
               variable="FlightResponseAA" />
      <!-- Process the results ... -->
    </sequence>
  </scope>
</forEach>
```

ภาพที่ 2.27 ตัวอย่างแอ็คติวิตี forEach

15) <pick> - คำสั่งเพื่อบล็อกและรอจนกระทั่งมีข่าวสารที่เหมาะสมมาถึงหรือหมดเวลาที่รอ เมื่อคำสั่งประเภทนี้ถูกกระตุ้นกิจกรรมที่เกี่ยวพันกันจะถูกกระทำและจะสิ้นสุดการเลือก (pick) ดังตัวอย่างในภาพที่ 2.28

```
<pick createInstance="yes">
       <onMessage partnerLink="CustMsgA" ... >
       <assign .../>
       </onMessage>
       <onMessage partnerLink="CustMsgB" ... >
       <assign .../>
       </onMessage>
       <onMessage partnerLink="CustMsgC" ... >
       <assign .../>
       </onMessage>
</pick>
```

ภาพที่ 2.28 ตัวอย่างแอ็คติวิตี pick

นอกจากนี้บีเพลยังมีชุดคำสั่งในการนิยามข้อมูล ดังนี้

16) <partnerLink> - คำสั่งเพื่อกำหนด portType ของเว็บเซอร์วิซ (เรียกว่า partner) ที่จะเข้ามาร่วมในกระบวนการทางธุรกิจ ดังตัวอย่างในภาพที่ 2.29

```
<partnerLink name=”seller” partnerLinkType=”AuctionHouse_SellerLT”
        myRole=”AuctionHouse” partnerRole=”Seller”
</partnerLink>
```

ภาพที่ 2.29 ตัวอย่างการใช้ partnerLink

17) <variables> - คำสั่งเพื่อกำหนดค่าตัวแปรในกระบวนการทางธุรกิจ ดังตัวอย่างในภาพที่ 2.30

```
<variables>
      <variable name="BPELVariableName" messageType="QName"?
            type="QName"?element="QName"?>+
            from-spec?
      </variable>
</variables>
```

ภาพที่ 2.30 ตัวอย่างการใช้ variables

2.1.6 **การทดสอบแบบมิวเทชัน** [6]

การทดสอบแบบมิวเทชันหรือการวิเคราะห์มิวเทชันนั้นเป็นวิธีการทดสอบความผิดพลาดเป็นหลัก ซึ่งถูกใช้วัดประสิทธิภาพของกรณีทดสอบที่สร้างขึ้นมา จากภาพที่ 2.31 เป็นแผนภาพอธิบายวิธีการทดสอบแบบมิวเทชันโดยทั่วไปหรืออาจเรียกว่าเป็นการทดสอบแบบสตรองมิวเทชัน กล่าวคือ เริ่มต้นทดสอบ โปรแกรมต้นแบบโดยใส่ตัวดำเนินการมิวเทชันเข้าไปในโปรแกรม กล่าวคือ การไปเปลี่ยนโปรแกรมต้นฉบับเพียง 1 ที่ในโปรแกรม ตัวอย่างเช่น การเปลี่ยนเครื่องหมายทางคณิตศาสตร์ในการคำนวณในโปรแกรม หรือเปลี่ยนแปลงนิพจน์ต่างๆในโปรแกรม หรือการลบทั้งข้อความในโปรแกรมออกไป

จากนั้นก็จะได้โปรแกรมที่มีข้อผิดพลาดหรือมิวแตนท์ นำมาทดสอบด้วยกรณีทดสอบที่เตรียมไว้ ซึ่งผลลัพธ์อาจจะได้ผลลัพธ์ออกมาเหมือนกับโปรแกรมต้นฉบับหรือไม่เหมือนกับโปรแกรมต้นฉบับ สามารถพิจารณาได้ดังนี้

1) กรณีผลลัพธ์ที่ออกมาไม่เหมือนกับโปรแกรมต้นฉบับ แสดงว่า มิวแตนท์นั้นถูกกำจัด (Killed) เรียบร้อยแล้ว
2) กรณีผลลัพธ์ที่ออกมาเหมือนกับโปรแกรมต้นฉบับ แสดงว่า กรณีทดสอบที่ใช้ในตอนแรกนั้นยังไม่สามารถกำจัดมิวแตนท์ได้ ทำให้มิวแตนท์นั้นยังคงอยู่ (Live Mutant) ดังนั้นผู้ทดสอบก็จะต้องทำการเพิ่มกรณีทดสอบใหม่ๆเข้าไป หรือปรับปรุงกรณีทดสอบให้ดีขึ้นแล้วจึงทดสอบใหม่เพื่อให้มิวแตนท์ที่ยังคงอยู่นั้นถูกกำจัด ถ้าไม่มีกรณีทดสอบใดๆเลยสามารถ

กำจัดมิวแตนท์ได้ แสดงว่ามิวแตนท์นั้นจะเป็นมิวแตนท์สมมูล (Equivalent Mutant)

ในการประเมินประสิทธิภาพของชุดกรณีทดสอบนั้นสามารถวัดด้วยคะแนนมิวเทชัน (Mutation Score) ดังสมการ

$$M = \frac{D}{T - E}$$

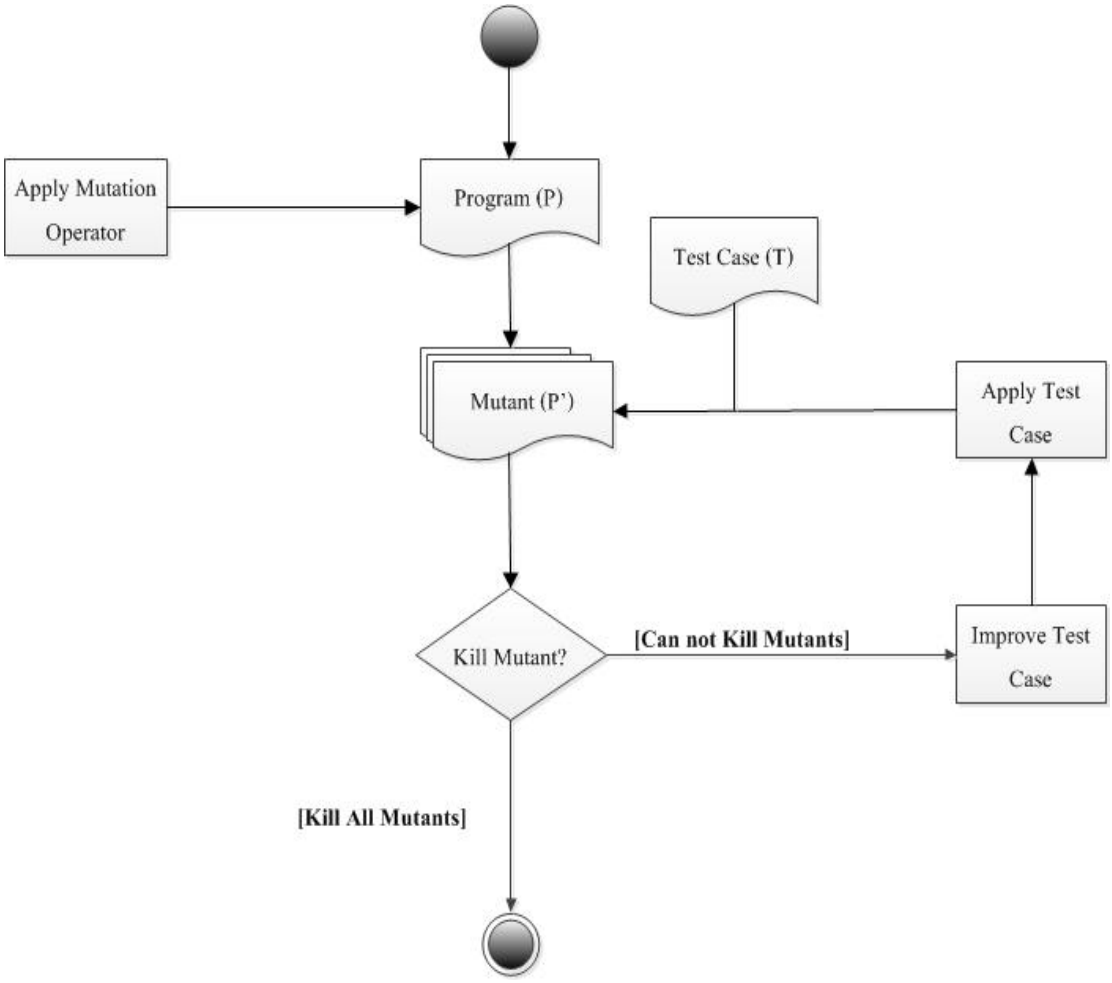
โดยที่ M คือ คะแนนมิวเทชัน

D คือ จำนวนของมิวแตนท์ที่สามารถกำจัดได้

T คือ จำนวนของมิวแตนท์ทั้งหมด

E คือ จำนวนของมิวแตนท์สมมูล

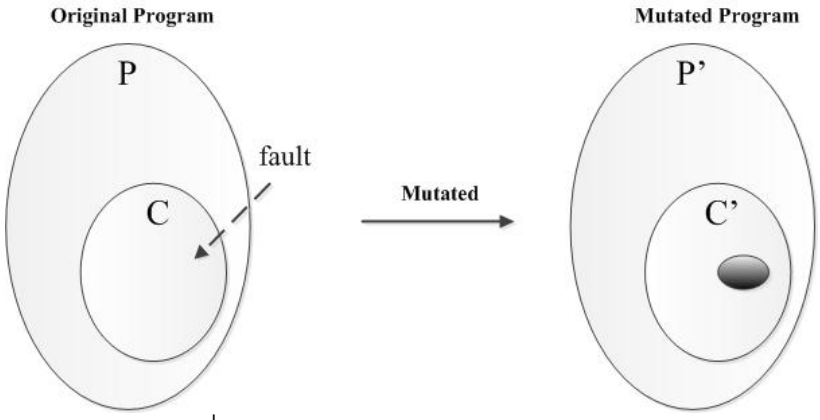
เป้าหมายของนักทดสอบนั้นมุ่งที่จะปรับปรุงคะแนนมิวเทชันให้ได้ 100% ซึ่งหมายความว่า กรณีทดสอบสามารถกำจัดมิวแตนท์ได้ทั้งหมด



ภาพที่ 2.31 วิธีการทั่วไปของการวิเคราะห์ด้วยวิธีมิวเทชัน [7]

2.1.7 การทดสอบแบบวีคมิวเทชัน [11]

ในผลงานวิจัยของฮาวเดน (W.E. Howden) [13] นั้นกำหนดให้โปรแกรม P ซึ่งประกอบด้วยคอมโพเนนท์ย่อย C และ C' เป็นคอมโพนเนท์เวอร์ชันที่ถูกใส่ข้อผิดพลาดลงไป และ P' เป็นโปรแกรม P ซึ่งมี C' อยู่ ซึ่งในการทดสอบแบบวีคมิวเทชัน ต้องการให้กรณีทดสอบ T ทำให้ C' ให้แสดงข้อผิดพลาดออกมาในการทดสอบอย่างน้อย 1 ครั้ง ซึ่ง P' อาจจะยังทำงานได้เหมือนเดิมเช่นเดียวกับ P ดังภาพที่ 2.32



ภาพที่ 2.32 การใส่ข้อผิดพลาดในคอมโพเนนท์

ซึ่งในผลงานวิจัยของฮาวเดนนั้นไม่ได้กำหนดเป็นที่ชัดเจนในส่วนของโปรแกรมคอมโพเนนท์ แต่ได้อธิบายถึงคอมโพเนนท์เหล่านั้นว่า “เป็นโครงสร้างในโปรแกรมที่สามารถคำนวณในเบื้องต้นได้” ซึ่งแบ่งออกเป็น 5 รูปแบบ ดังต่อไปนี้

1) ตัวแปรอ้างอิง (Variable Reference)
2) การกำหนดค่าตัวแปร (Variable Assignment)
3) นิพจน์คำนวณ (Arithmetic Expression)
4) นิพจน์สัมพันธ์ (Relational Expression)
5) นิพจน์แบบบูลลีน (Boolean Expression)

จากผลงานวิจัย [11] ได้กำหนดเกี่ยวกับโปรแกรมคอมโพเนนท์เป็นเฉพาะที่ในโปรแกรม โดยสถานะของโปรแกรมต้นฉบับและโปรแกรมที่ถูกใส่ข้อผิดพลาดจะถูกเปรียบเทียบกัน และได้หาจุดเปรียบเทียบออกมาได้ 4 จุด ดังภาพที่ 2.33 เพื่อเปรียบเทียบ ดังนี้

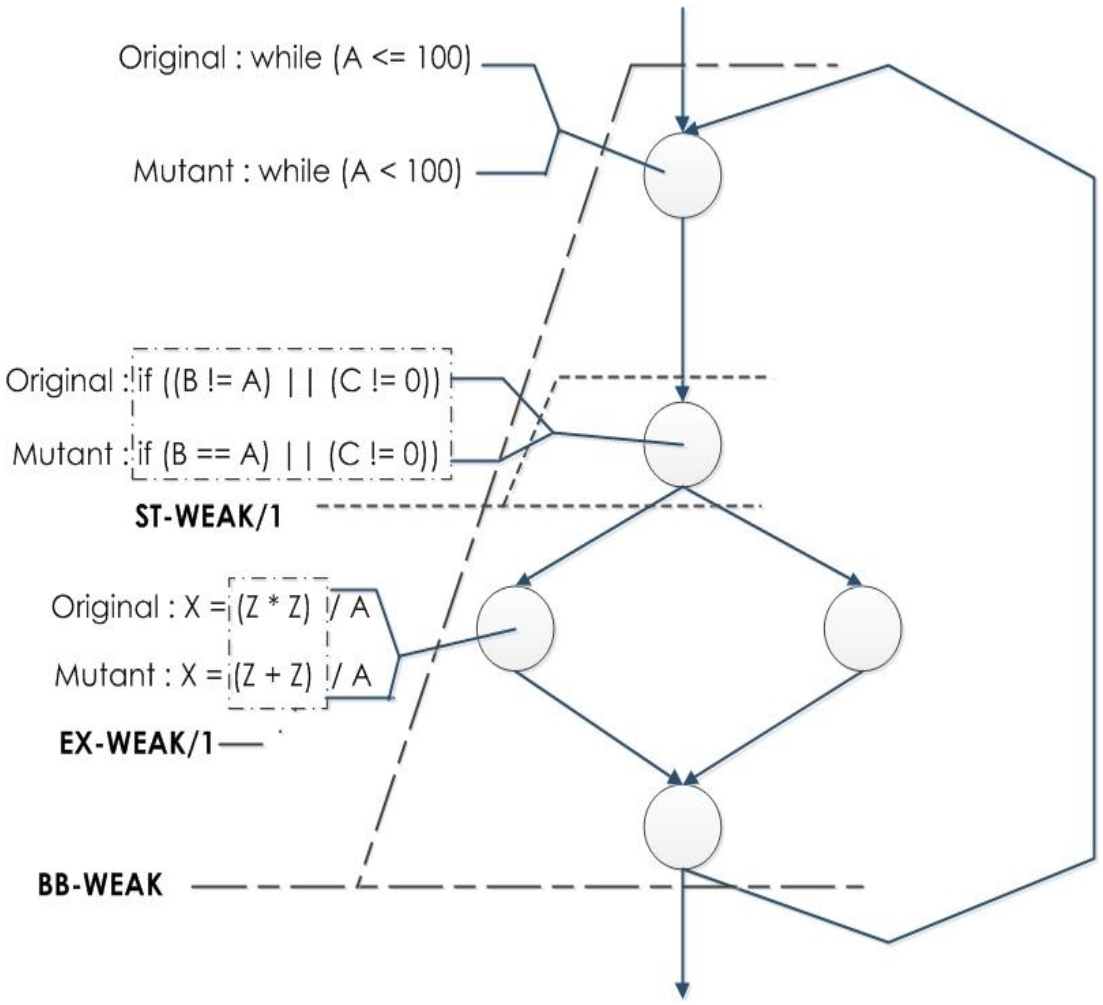
1) EX-WEAK/1 (Expression-WEAK/1) Mutation เป็นชนิดของการทดสอบแบบวีคมิวเทชัน โดยเป็นการเปรียบเทียบของสถานะ (การเปรียบเทียบผลลัพธ์หลังจากการกระทำโปรแกรมคอมโพเนนท์ส่วนนี้ไปแล้ว 1 ครั้ง) ระหว่างโปรแกรมต้นแบบและมิวแตนท์ ซึ่งการทดสอบแบบวีคมิวเทชันชนิดนี้จะพิจารณาเฉพาะนิพจน์เท่านั้น ดังภาพที่ 2.33 โปรแกรมต้นแบบมีสมการ X = (Z * Z)/A และมิวแตนท์มีสมการเป็น X = (Z + Z)/A ซึ่งหลังจากกระทำนิพจน์ดังกล่าวครั้งแรกไปแล้วจะทำการ

เปรียบเทียบค่า X เพื่อดูว่าผลลัพธ์ของค่า X เมื่อเทียบกันแล้วเท่ากันหรือไม่ระหว่างโปรแกรมต้นฉบับและมิวแตนท์

2) ST-WEAK/1 (Statement-WEAK/1) Mutation เป็นอีกชนิดของการทดสอบแบบวีคมิวเทชันโดยเปรียบเทียบผลลัพธ์ของประพจน์ระหว่างโปรแกรมต้นแบบและมิวแตนท์หลังจากผ่านการกระทำประพจน์ ซึ่งถ้าพิจารณาในภาพที่ 2.33 จะเปรียบเทียบผลลัพธ์ของประพจน์ ซึ่งในโปรแกรมต้นแบบมีประพจน์ คือ if ((B == A) || (C != 0)) และมิวแตนท์มีประพจน์ คือ if ((B != A) || (C != 0)) หลังจากผ่านการกระทำประพจน์ส่วนนี้ไปแล้ว 1 ครั้ง

3) BB-WEAK/1 (Basic-Block-WEAK/1 Execution) Mutation การทดสอบแบบวีคมิวเทชันชนิดนี้พิจารณาจากบล็อกพื้นฐาน (เป็นชุดคำสั่งที่ใหญ่ที่สุดในโปรแกรมเมื่อเปรียบเทียบกับนิพจน์หรือประพจน์ที่มีทางเข้าและทางออกเพียงทางเดียว) ในโปรแกรม เช่น ชุดคำสั่ง while หรือ repeatUntil เป็นต้น ซึ่งในแต่ละบล็อกพื้นฐานสามารถประกอบไปด้วยมิวแตนท์ของนิพจน์หรือประพจน์ในบล็อกพื้นฐานนั้นๆได้ จากภาพที่ 2.33 ชุดคำสั่ง while นั้นสามารถมีมิวแตนท์ภายในชุดคำสั่ง while ได้ การกำจัดมิวแตนท์อาจเปรียบเทียบค่าของ A ว่ามีค่าเท่ากันหรือไม่สำหรับการวนซ้ำในครั้งแรกเพียงครั้งเดียว

4) BB-WEAK/N (Basic-Block-WEAK/N Execution) Mutation เป็นการทดสอบแบบวีคมิวเทชันที่ถูกนำมาใช้เพื่อสนับสนุนวิธี BB-WEAK/1 เนื่องจากการวนซ้ำเพียง 1 ครั้ง ของวิธี BB-WEAK/1 บางกรณีไม่สามารถกำจัดมิวแตนท์ที่เกิดขึ้นในบล็อกพื้นฐานได้ การวนซ้ำแต่ละครั้งก็จะถูกเปรียบเทียบค่าตัวแปรต่างๆในบล็อกพื้นฐาน และการวนซ้ำจะหยุดเมื่อมิวแตนท์ถูกกำจัดหรือเมื่อเจอสถานะที่ไม่ถูกต้อง คือ ค่าในตัวแปรไม่ตรงกันเมื่อเปรียบเทียบระหว่างโปรแกรมต้นฉบับและมิวแตนท์ เป็นต้น หรือบล็อกพื้นฐานของมิวแตนท์มีจำนวนครั้งที่ถูกกระทำมากกว่าบล็อกพื้นฐานของต้นฉบับ



ภาพที่ 2.33 จุดที่เปรียบเทียบกันระหว่างสถานะของวิธีทดสอบแบบวีคมิวเทชัน [10]

2.1.8 ตัวดำเนินการมิวเทชัน

การทดสอบแบบมิวเทชันมีแนวคิดเพื่อที่จะใส่ข้อผิดพลาด 1 ข้อผิดพลาดลงไปในโปรแกรมต้นแบบ ทำให้เกิดมิวแตนท์ขึ้นมาจำนวนมากโดยอาศัยตัวดำเนินการมิวเทชัน โดยตัวดำเนินการมิวเทชันจะบ่งบอกถึงการเข้าไปดัดแปลงนิพจน์ ประพจน์ หรือบล็อกพื้นฐาน ซึ่งจะขึ้นอยู่กับชนิดของตัวดำเนินการมิวเทชันนั้นๆ

ตัวดำเนินการมิวเทชันเกิดจากรูปแบบของข้อผิดพลาดที่มาจากโปรแกรมเมอร์หรือนักพัฒนาซอฟต์แวร์ โดยรูปแบบของข้อผิดพลาดก็จะแตกต่างกันไปในแต่ละภาษาขึ้นอยู่กับไวยากรณ์ในภาษานั้นๆ ซึ่งจากงานวิจัยที่ผ่านมาได้มีการกำหนดตัวดำเนินการมิวเทชันในแต่ละภาษา ตามตัวอย่างดังต่อไปนี้

1) Java – งานวิจัย [25] ได้กำหนดตัวดำเนินการมิวเทชันของภาษาจาวาที่เกี่ยวกับอ็อปเจ็กต์ที่กระทำแบบพร้อมกัน (Concurrent) เช่น Threads, Synchronization methods และ Synchronization statements เป็นต้น และงานวิจัย [26] ได้กำหนดตัวดำเนินการมิวเทชันที่เป็นชนิด Collection interface, Iterator interface และ List and Vector เป็นต้น

2) C# - งานวิจัย [23] ออกแบบตัวดำเนินการมิวเทชันในภาษา C# โดยตัวดำเนินการมิวเทชันส่วนใหญ่จะสอดคล้องกับแนวคิดของอ็อปเจ็กต์ ซึ่งมีตัวดำเนินการชนิดวิธีตัวแทน (Delegate Method) และการดักจับข้อผิดพลาด เป็นต้น

3) SQL – งานวิจัย [22] ได้ใช้การวิเคราะห์แบบมิวเทชันโดยใช้ตัวดำเนินการมิวเทชันอยู่ 4 ชนิด คือ ตัวดำเนินการมิวเทชันที่เกี่ยวกับการใช้คำสั่ง SQL ตัวดำเนินการมิวเทชันที่เกี่ยวกับเงื่อนไขในคำสั่ง ตัวดำเนินการมิวเทชันที่เกี่ยวกับการดักจับ NULL และตัวดำเนินการมิวเทชันที่เกี่ยวกับการแทนที่คำสงวนในคำสั่ง SQL

4) Ada – งานวิจัย [21] ได้ใช้ตัวดำเนินการที่เกี่ยวกับภาษา Ada โดยมีทั้งหมด 5 ชนิด คือ Operand Replacement, Operators Statement, Operators Expression Operators, Coverage Operator และ Tasking Operators ซึ่งมีตัวดำเนินการมิวเทชันทิ้งสิ้น 65 ตัวดำเนินการมิวเทชัน

5) C – งานวิจัย [27] ได้กำหนดตัวดำเนินการมิวเทชันชนิด Statements Mutations, Operator Mutations และ Variable Mutations

6) Fortran – งานวิจัย [20] ได้กำหนดตัวดำเนินการมิวเทชันชนิด Statements analysis, Predicate analysis และ Coincidental correctness

7) WS-BPEL - งานวิจัย [8] ได้กำหนดตัวดำเนินการมิวเทชันขึ้นมา 26 ตัว ดำเนินการมิวเทชันสำหรับบีเพล ซึ่งตัวดำเนินการเหล่านี้สามารถจำแนกออกได้เป็น 4 ประเภท ดังต่อไปนี้

1. ตัวดำเนินการสำหรับการแทนที่ตัวระบุ (Identifier replacement operators - I)
2. ตัวดำเนินการสำหรับนิพจน์ (Expression operators - E)
3. ตัวดำเนินการสำหรับกิจกรรม (Activity operators - A)
4. ตัวดำเนินการสำหรับความผิดปกติและเหตุการณ์ (Exception and event operators - X)

แต่ละประเภทก็จะถูกกำหนดด้วยตัวอักษรใหญ่ คือ I, E, A หรือ X ตัวดำเนินการมิวเทชันทั้ง 26 ตัวดังที่กล่าวข้างต้นก็จะถูกแบ่งแยกไปตามแต่ละประเภท โดยเครื่องหมาย ✡ จะหมายถึงตัวดำเนินการนี้ถูกกำหนดขึ้นมาสำหรับภาษาบีเพลโดยเฉพาะ

ในขั้นตอนของการออกแบบตัวดำเนินการมิวเทชันสำหรับภาษาบีเพล ผู้วิจัยได้สังเกตเห็นถึงรูปแบบของข้อผิดพลาดของนักเขียนโปรแกรมที่พบบ่อยๆ เมื่อนำเว็บเซอร์วิซไปประยุกต์ใช้ ผู้วิจัยได้สมมติว่านักเขียนโปรแกรมนั้นเขียนโปรแกรมบีเพลขึ้นโดยตรงแต่ใช้เครื่องมือที่เป็นกราฟมาช่วยในการเขียนโปรแกรม ดังนั้นข้อผิดพลาดหลายๆ ตัวที่เกิดจากการเขียนโปรแกรมด้วยมือในภาษาอื่นๆ จะไม่ปรากฏในโปรแกรมบีเพล ดังนั้นผู้วิจัยจึงไม่ได้กำหนดตัวดำเนินการการลดแบบเอกภาค (Unary Minus Operator) ข้างหน้านิพจน์ เนื่องจากว่าได้ถูกพิจารณาแล้วจากงานวิจัยของโปรแกรมภาษาอื่นๆ [19 - 22]

ในขณะที่ตัวดำเนินการมิวเทชันได้ถูกกำหนดขึ้นสำหรับภาษาบีเพล ในภาษาอื่นๆ ก็ได้มีการปรับใช้เพื่อให้สามารถใช้ตัวดำเนินการได้อย่างกว้างขวาง ดังเช่นตัวดำเนินการ ISV, ERR และ ELL ซึ่งดั้งเดิมได้นำไปใช้กับภาษา C# และ Java

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัยเรื่อง “Mutation Testing for Expression Modification Operator of BPEL” [7]

จากงานวิจัยนี้ได้เสนอวิธีการทดสอบบีเพลด้วยวิธีการทดสอบมิวเทชันแบบการเลือก ซึ่งเป็นเทคนิคการลดจำนวนมิวแตนท์และได้กำหนดตัวดำเนินการมิวเทชันในแบบนิพจน์ขึ้นมาเองสำหรับการนำไปใช้กับภาษาบีเพล ซึ่งประกอบด้วยตัวดำเนินการในตารางที่ 2.1 ดังต่อไปนี้

ตารางที่ 2.1 ตัวดำเนินการมิวเทชันสำหรับนิพจน์ในภาษาบีเพล

AOR	Arithmetic Operator Replacement
ROR	Relational Operator Replacement
LOR	Logical Operator Replacement
LOD	Logical Operator Delete
LOI	Logical Operator Insertion

AOR (Arithmetic Operator Replacement) เป็นการใส่ข้อผิดพลาดลงไปในโปรแกรมโดยไปแทนที่เครื่องหมายทางคณิตศาสตร์ (+, -, *, /, %)

ROR (Relational Operator Replacement) เป็นการใส่ข้อผิดพลาดลงไปในโปรแกรมโดยการแทนที่เครื่องหมายเชิงสัมพันธ์ (=, !=, >, <, >=, <=)

LOR (Logical Operator Replacement) เป็นการใส่ข้อผิดพลาดลงไปในโปรแกรมโดยการแทนที่ดำเนินการเชิงตรรกะ (and, or)

LOD (Logical Operator Delete) เป็นการใส่ผิดพลาดลงไปในโปรแกรมโดยการลบตัวดำเนินการเชิงตรรกะที่เป็นเอกภาค (not) ออกไป

LOI (Logical Operator Insertion) เป็นการใส่ผิดพลาดลงไปในโปรแกรมโดยการเพิ่มตัวดำเนินการเชิงตรรกะที่เป็นเอกภาค (not) ลงไป

จากนั้นได้นำตัวดำเนินการดังกล่าวไปสร้างข้อผิดพลาดกับกระบวนการที่เตรียมไว้ 3 กระบวนการเพื่อสร้างมิวแตนท์ซึ่งมีกระบวนการสามเหลี่ยม, กระบวนการอนุมัติการกู้ยืม และกระบวนการซื้อสินค้า หลังจากการสร้างมิวแตนท์แล้วจึงนำกรณีทดสอบที่เตรียมไว้ไปทดสอบกับมิวแตนท์ที่สร้างขึ้น

จากงานวิจัยนี้ได้เสนอแนวทางการทดสอบแบบมิวเทชันชนิดการเลือกซึ่งเป็นเทคนิคการลดจำนวนมิวแตนท์ โดยจะเห็นได้ว่ามีการใช้ตัวดำเนินการมิวเทชันสำหรับนิพจน์เพียงอย่างเดียว ทำให้การทดสอบนั้นยังไม่ครอบคลุมนักในภาษาบีเพล ซึ่งในงานวิจัย [9] นั้นได้มีการกำหนดตัวดำเนินการมิวเทชันเพิ่มขึ้นโดยพิจารณาจากงานวิจัย [10] เพื่อทำให้การทดสอบนั้นครอบคลุมกับภาษาบีเพลมากยิ่งขึ้น

2.2.2 **งานวิจัยเรื่อง** “Quantitative Evaluation of Mutation Operators for WS-BPEL Compositions” [8]

งานวิจัยนี้ได้เสนอการประเมินเชิงปริมาณของคุณภาพของกลุ่มของตัวดำเนินการมิวแตนท์สำหรับดับเบิลยูเอสบีเพล 2.0 ได้มีการกำหนดตัวดำเนินการมิวเทชันในรูปแบบที่มากขึ้นโดยแบ่งเป็น 4 ประเภท ดังตารางที่ 2.2

ตารางที่ 2.2 รายการตัวดำเนินการมิวเทชันสำหรับภาษาบีเพล 2.0

	ตัวดำเนินการ	**คำอธิบาย**
Identifier Mutation	ISV	การแทนที่ตัวระบุตัวแปรด้วยตัวอื่นแต่เป็นชนิดเดียวกัน
Expression Mutation	EAA	การแทนที่ตัวดำเนินการเครื่องหมายทางคณิตศาสตร์(+, -, *, div, mod)
	EEU	การเอาเครื่องหมายลบที่เป็นเอกภาคออกจากนิพจน์
	ERR	การแทนที่ตัวดำเนินการความสัมพันธ์ (=, !=, <, >, <=, >=)
	ELL	การแทนที่เครื่องหมายทางทางตรรกะ (and, or)
	ECC	การแทนที่เครื่องหมายเครื่องหมายเส้นทาง (/, //)

ตารางที่ 2.3 รายการตัวดำเนินการมิวเทชันสำหรับภาษาบีเพล 2.0 (ต่อ)

	ECN	ปรับเปลี่ยนค่าคงที่ทางตัวเลขโดยการเพิ่มหรือลดค่าลงทีละ 1 หรือการเพิ่มหรือลดตำแหน่ง 1 ตำแหน่ง
	EMD	ปรับเปลี่ยนนิพจน์ช่วงเวลา (duration expression) โดยแทนที่ด้วย 0 หรือด้วยค่าครึ่งหนึ่งของค่าเริ่มต้น
	EMF	ปรับเปลี่ยนนิพจน์เส้นตาย (deadline expression) โดยแทนที่ด้วย 0 หรือด้วยค่าครึ่งหนึ่งของค่าเริ่มต้น
Concurrent Activity Mutation	AFP	การเปลี่ยนค่าแอตทริบิวต์ parallel เป็น yes ในกิจกรรม forEach ด้วย no
	ACI	เปลี่ยนลักษณะประจำ createInstance จากกิจกรรมข้อความขาเข้าเป็นไม่มี
	ASF	การแทนที่กิจกรรม sequence ด้วยกิจกรรม flow
	AIS	การเปลี่ยนลักษณะประจำ isolated ของ scope ด้วย no
	AEL	การเอากิจกรรมออก
Non-Concurrent Activity Mutation	AIE	การเอาส่วนย่อย (element) elseif หรือ else ออกจาก if
	AWR	การแทนที่กิจกรรม while ด้วย repeatUntil หรืออย่างอื่น
	AJC	การเอาลักษณะประจำ (attribute) joinCondition ออกจากกิจกรรม
	ASI	การเปลี่ยนลำดับระหว่าง sequence 2 กิจกรรม
	APM	การเอาส่วนย่อย (element) onMessage ออกจากกิจกรรม pick
	APA	การเอาส่วนย่อย (element) onAlarm ออกจากกิจกรรม pick หรือจากตัวจับเหตุการณ์ (event handler)

ตารางที่ 2.3 รายการตัวดำเนินการมิวเทชันสำหรับภาษาบีเพล 2.0 (ต่อ)

Exceptional and Event Mutation	XMC	เอาการกำหนดตัวจับการทดแทน (compensationHandler) ออกไป
	XMT	เอาการกำหนดตัวจับการสิ้นสุด (terminationHandler) ออกไป
	XMF	เอาส่วนย่อย catch หรือ catchAll ออกจากตัวจับข้อผิดพลาด (fault handler)
	XTF	การแทนที่การโยนข้อผิดพลาดด้วยกิจกรรม throw
	XER	การเอากิจกรรม rethrow ออก
	XEE	การเอาส่วนย่อย (element) onEvent ออกจากตัวจับเหตุการณ์ (eventHandler)

โดยการทดลองนั้นจะนำตัวดำเนินการไปใช้กับกระบวนการ 3 กระบวนการ คือ กระบวนการอนุมัติการกู้ยืม กระบวนการตลาดสินค้า และ กระบวนการค้นหาแบบเมตา (MetaSearch) โดยการสร้างมิวแตนท์ของแต่ละกระบวนการ จากนั้นทำการรันมิวแตนท์ด้วยชุดกรณีทดสอบที่สร้างขึ้นมา เพื่อดูประสิทธิภาพของชุดกรณีทดสอบและดูความคงทนของมิวแตนท์ที่เกิดขึ้น

อย่างไรก็ตามงานวิจัยนี้ใช้กระบวนการทดสอบแบบสตรองมิวเทชัน ซึ่งเป็นแนวทางคาดว่าผลลัพธ์จะต่างจากโปรแกรมต้นแบบ ซึ่งบางครั้งทำให้สิ้นเปลืองในส่วนที่เกี่ยวกับการคำนวน (Computational Cost) มาก ถ้านำใช้กับโปรแกรมที่มีความซับซ้อนมากๆ

บทที่ 3
แนวคิดและวิธีการดำเนินงาน

3.1 การวิเคราะห์ตัวดำเนินการมิวเทชัน

วิทยานิพนธ์นี้มีจุดประสงค์ในการวิเคราะห์ตัวดำเนินการมิวเทชันที่ถูกกำหนดขึ้นมาในงานวิจัย [8] ว่ามีตัวดำเนินการมิวเทชันอะไรบ้างที่สามารถประยุกต์ใช้ได้กับการทดสอบแบบวีคมิวเทชันทั้ง 4 แบบ ซึ่งตัวดำเนินการมิวเทชันทั้งหมด 26 ตัวดำเนินการมิวเทชัน ถูกกล่าวไว้ในบทที่ 2

สำหรับหัวข้อนี้ได้อธิบายถึงการวิเคราะห์การนำตัวดำเนินการมิวเทชันทุกประเภทกับการวิเคราะห์แบบวีคมิวเทชัน ในตารางที่ 3.1 เป็นตัวดำเนินการมิวเทชันที่สามารถใช้ได้กับแต่ละชนิดของวีคมิวเทชัน ซึ่งการวิเคราะห์นั้นจะแยกตามชนิดของการทดสอบแบบวีคมิวเทชัน โดยสามารถแบ่งออกเป็น 3 แบบ ดังนี้

1. การวิเคราะห์ที่เกี่ยวกับนิพจน์ - เกี่ยวข้องกับวิธีการทดสอบวีคมิวเทชันแบบ EX-WEAK/1 โดยพิจารณาเพียงนิพจน์เท่านั้นซึ่งจะเห็นว่า ตัวดำเนินการมิวเทชันที่สามารถประยุกต์ใช้ได้กับ EX-WEAK/1 คือ ตัวดำเนินการมิวเทชันประเภทนิพจน์ ส่วนมิวเทชันประเภทตัวระบุก็สามารถประยุกต์ใช้ด้วยวิธี EX-WEAK/1 ได้เช่นกัน ซึ่งตัวดำเนินการดังกล่าวมีดังต่อไปนี้ คือ EAA EEU ERR ELL ECC ECN EMD EMF และ ISV

2. การวิเคราะห์ที่เกี่ยวกับประพจน์ – เป็นการพิจารณาด้วยวิธีการทดสอบแบบ ST-WEAK/1 ซึ่งครอบคลุมไปถึงการวิเคราะห์นิพจน์และประพจน์ โดยตัวดำเนินการที่สามารถประยุกต์ใช้ได้กับการทดสอบ ST-WEAK/1 คือ ตัวดำเนินการประเภทตัวระบุ ตัวดำเนินการประเภทนิพจน์ทั้งหมด และตัวดำเนินการประเภทกิจกรรมบางตัว คือ AIE AJC APM และ APA ตัวอย่างเช่น AIE นั้นเป็นตัวดำเนินการที่ลบกิจกรรมย่อย elseif หรือ else ออกจากกิจกรรม if เป็นต้น ซึ่งสามารถวิเคราะห์ได้ด้วยวิธี ST-WEAK/1 ขณะเดียวกันตัวดำเนินการประเภทกิจกรรมที่เหลือ คือ AFP ASF AIS AIE AWR และ ASI ไม่สามารถใช้ ST-WEAK/1 วิเคราะห์ในการกำจัดมิวแตนท์ได้ เนื่องจากต้องอาศัยการวิเคราะห์แบบบล็อกพื้นฐาน

3. การวิเคราะห์ที่เกี่ยวกับบล็อกพื้นฐาน – เป็นการพิจารณาด้วยวิธีการทดสอบแบบ BB-WEAK/1 และ BB-WEAK/N เนื่องจากการทดสอบแบบนี้จะพิจารณาบล็อกพื้นฐานเป็นหลัก ซึ่งครอบคลุมไปถึงนิพจน์และประพจน์ด้วย เนื่องจากบล็อกพื้นฐานเป็นชุดคำสั่งที่ใหญ่ที่สุดในโปรแกรม การวิเคราะห์ด้วยวิธีนี้จึงสามารถตรวจสอบมิวแตนท์จากตัวดำเนินการประเภทกิจกรรมที่การวิเคราะห์แบบ ST-WEAK/1 ไม่สามารถวิเคราะห์ได้ คือ AFP ASF AIS AEI AWR และ ASI ส่วนตัวดำเนินการมิวเทชันประเภทตัวระบุและส่วนตัวดำเนินการมิวเทชันประเภทนิพจน์จะสร้างมิวแตนท์ของนิพจน์และประพจน์ ซึ่งเป็นองค์ประกอบหนึ่งที่อยู่ในบล็อกพื้นฐาน ด้วยเหตุนี้การกำจัดมิวแตนท์ในการวนซ้ำครั้งแรก ด้วย BB-WEAK/1 อาจไม่สามารถทำได้ จึงต้องอาศัยการทดสอบด้วยวิธี BB-WEAK/N โดยการวนซ้ำจำนวน N รอบ เพื่อพิจารณาค่าของตัวแปรในแต่ละรอบของการวนซ้ำ

ตัวดำเนินการตัวอื่นๆที่ไม่ได้กล่าวถึง คือ ตัวดำเนินการ ACI (การเปลี่ยนค่าในแอตทริบิวต์ createInstance ในกิจกรรมข้อความเป็น no) โดยตัวดำเนินการ ACI จะสร้างตัวอย่างกระบวนการ (Process Instance) ซึ่งไม่สามารถเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์ได้ ส่วนตัวดำเนินการ AIS (การเปลี่ยนค่าในแอตทริบิวต์ isolated ของกิจกรรม scope จากค่า yes เป็น no) จะเป็นการพิจารณาการเข้าถึงตัวแปร ด้วยเหตุนี้จึงไม่ถูกนำมาใช้ในแนวทางในการสร้างมิวแตนท์ในวิทยานิพนธ์นี้ ด้วยเหตุผลดังต่อไปนี้

1. ตัวดำเนินการมิวเทชัน ACI นั้นเข้าไปเปลี่ยนค่าในแอตทริบิวต์ createInstance จาก yes เป็น no ซึ่งถูกกำหนดไว้เพียงในกิจกรรม receive เท่านั้น ดังภาพที่ 3.1 ซึ่งไม่สามารถหานิพจน์ ประพจน์ หรือบล็อกพื้นฐานในกิจกรรม receive ได้

```
<process>
   <receive createInstance="yes" />
   <switch>
     <case name="msn">
       <invoke name="install_msn" />
     </case>
     <case name="skype">
       <invoke name="install_skype" />
     </case>
   </switch>
   <reply variabele="status" />
</process>
```

ภาพที่ 3.1 ตัวอย่างโปรแกรมบีเพลที่ใช้แอตทริบิวต์ createInstance

2. ตัวดำเนินการมิวเทชัน AIS ซึ่งจะเข้าไปเปลี่ยนค่าของแอตทริบิวต์ isolated จาก yes เป็น no ซึ่งถูกกำหนดไว้เพียงในกิจกรรม scope เท่านั้น ดังภาพที่ 3.2 ซึ่งไม่สามารถหานิพจน์ประพจน์ หรือบล็อกพื้นฐานในกิจกรรม scope ได้

```
<process ...>
<variables>
<variable name="TotalAmount" element="…" />
variables>
<flow>
<scope name="Merchant" isolated="yes">
<sequence>
…
…
…

</scope>
</process>
```

ภาพที่ 3.2 ตัวอย่างโปรแกรมบีเพลที่ใช้แอตทริบิวต์ isolated

3. ด้วยเหตุผลที่กล่าวในข้อที่ 1 และ 2 นั้น ไม่สามารถประยุกต์ใช้กระบวนการทดสอบแบบวีคมิวเทชันได้ ซึ่งมีการวิเคราะห์เฉพาะนิพจน์ (EX-WEAK/1) การวิเคราะห์เฉพาะประพจน์ (ST-WEAK/1) และการวิเคราะห์เฉพาะบล็อกพื้นฐาน (BB-WEAK/1 และ BB-WEAK/N) กล่าวคือ ไม่สามารถทดสอบกับมิวแตนท์ที่เกิดจากการดัดแปลงค่าในแอตทริบิวต์ createInstance และ isolated ของตัวดำเนินการมิวเทชัน ACI และ AIS ได้ ขณะเดียวกันมิวแตนท์ที่เกิดจากตัวดำเนินการเหล่านี้ สามารถใช้การทดสอบแบบสตรองมิวเทชันซึ่งเป็นการพิจารณาผลลัพธ์ของทั้งโปรแกรมได้ โดยการเปรียบเทียบผลลัพธ์ของทั้งโปรแกรมต้นแบบและมิวแตนท์

ตารางที่ 3.1 รายการตัวดำเนินการที่สามารถใช้กับวีคมิวเทชัน

ตัวดำเนินการ	การทดสอบวีคมิวเทชัน			
	EX-WEAK/1	ST-WEAK/1	BB-WEAK/1	BB-WEAK/N
มิวเทชันสำหรับตัวระบุ				
ISV	✓	✓	✓	✓
มิวเทชันสำหรับนิพจน์				
EAA	✓	✓	✓	✓
EEU	✓	✓	✓	✓
ERR	✓	✓	✓	✓
ELL	✓	✓	✓	✓
ECC	✓	✓	✓	✓
ECN	✓	✓	✓	✓

ตารางที่ 3.1 รายการตัวดำเนินการที่สามารถใช้กับวีคมิวเทชัน (ต่อ)

EMD	✓	✓	✓	✓
EMF	✓	✓	✓	✓
มิวเทชันสำหรับกิจกรรม				
ACI	-	-	-	-
AFP	-	-	✓	✓
ASF	-	-	✓	✓
AIS	-	-	-	-
AEL	-	-	✓	✓
AIE	-	-	✓	✓
AWR	-	-	✓	✓
AJC	-	✓	✓	✓
ASI	-	-	✓	✓
APM	-	✓	✓	✓
APA	-	✓	✓	✓
มิวเทชันสำหรับความผิดปกติและเหตุการณ์				
XMF	-	✓	✓	✓
XMC	-	✓	✓	✓
XMT	-	✓	✓	✓
XTF	-	✓	✓	✓
XER	-	✓	✓	✓
XEE	-	✓	✓	✓

หมายเหตุ - หมายถึง ไม่ได้นำไปประยุกต์ใช้กับการทดสอบแบบวีคมิวเทชันได้

✓ หมายถึง นำไปประยุกต์ใช้กับการทดสอบแบบวีคมิวเทชันได้

3.2 การสร้างมิวแตนท์

ในส่วนนี้จะอธิบายถึงการสร้างมิวแตนท์ในแต่ละตัวดำเนินการมิวเทชัน ดังต่อไปนี้

1) ตัวดำเนินการ ISV เป็นตัวดำเนินการมิวเทชันชนิดตัวระบุ ทำหน้าที่ดัดแปลงตัวแปรในกระบวนการของบีเพลเป็นอีกตัวแปรๆ หนึ่งที่เป็นชนิดเดียวกัน ดังตัวอย่างในภาพที่ 3.3 เป็นการสลับที่ตัวแปร $input.A ในโปรแกรมต้นแบบไปเป็นตัวแปร $input.B ในมิวแตนท์ โดยที่ $input.A และ $input.B เป็นตัวแปรชนิดเดียวกัน คือ integer

โปรแกรมต้นแบบ

```
<bpws:elseif>
<bpws:condition><![CDATA[$input.A != $input.B and $input.A != $input.C
and $input.B != $input.C ]]></bpws:condition>
      <bpws:sequence name="HiddenSequence1">
                    …
                    <bpws:from><![CDATA['Scalene']]></bpws:from>
                    …
       </bpws:sequence>
</bpws:elseif>
```

มิวแตนท์

```
<bpws:elseif>
<bpws:condition><![CDATA[$input.B != $input.B and $input.A != $input.C
and $input.B != $input.C ]]></bpws:condition>
      <bpws:sequence name="HiddenSequence1">
                    …
                    <bpws:from><![CDATA['Scalene']]></bpws:from>
                    …
      </bpws:sequence>
</bpws:elseif>
```

ภาพที่ 3.3 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ISV

2) ตัวดำเนินการ EAA เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงนิพจน์ไม่ว่าจะเป็นนิพจน์หรือประพจน์ที่มีการดำเนินการทางคณิตศาสตร์ โดยดัดแปลงตัวดำเนินการทางคณิตศาสตร์ (+, -, *, div, mod) ไปเป็นตัวดำเนินการอีกตัวหนึ่งที่เหลือ ดังตัวอย่างภาพที่ 3.4 ในโปรแกรมต้นแบบใช้เครื่องหมาย + ในนิพจน์ $input.A + $input.B และในมิวแตนท์ใช้เครื่องหมาย – ในนิพจน์ $input.A - $input.B และจะสร้างมิวแตนท์ที่เหลือเปลี่ยนจากเครื่องหมาย + เป็นเครื่องหมาย *, div และ mod ด้วยเช่นกัน

โปรแกรมต้นแบบ

```
<bpel:copy>
      <bpel:from>
            <![CDATA[$input.A + $input.B]]>
      </bpel:from>
      <bpel:to variable="calResult"></bpel:to>
</bpel:copy>
```

มิวแตนท์

```
<bpel:copy>
      <bpel:from>
            <![CDATA[$input.A - $input.B]]>
      </bpel:from>
      <bpel:to variable="calResult"></bpel:to>
</bpel:copy>
```

ภาพที่ 3.4 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ EAA

3) ตัวดำเนินการ EEU เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงเครื่องลบที่เป็นเอกภาคออกจากนิพจน์หรือประพจน์ ดังตัวอย่างภาพที่ 3.5 ในโปรแกรมต้นแบบในนิพจน์ -1 + $counter ขณะที่ในนิพจน์ของมิวแตนท์จะลบเครื่องหมาย - โดดๆ ออก คือ 1 + $counter

โปรแกรมต้นแบบ

```
<bpel:copy>
      <bpel:from>-1 + $counter</bpel:from>
      <bpel:to>$counter</bpel:to>
</bpel:copy>
```

มิวแตนท์

```
<bpel:copy>
      <bpel:from>1 + $counter</bpel:from>
      <bpel:to>$counter</bpel:to>
</bpel:copy>
```

ภาพที่ 3.5 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ EEU

4) ตัวดำเนินการ ERR เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงตัวดำเนินการความสัมพันธ์ (=, !=, <, >, <=, >=) เป็นตัวดำเนินการอีกตัวหนึ่ง ดังตัวอย่างภาพที่ 3.6 โดยในโปรแกรมต้นแบบใช้เครื่องหมาย = ในนิพจน์ $input.A = $input.B ส่วนในมิวแตนท์ใช้

เครื่องหมาย > ในนิพจน์ $input.A > $input.B และจะสร้างมิวแตนท์ที่เหลือเปลี่ยนจากเครื่องหมาย = เป็นเครื่องหมาย <, !=, <= และ >= ด้วยเช่นกัน

โปรแกรมต้นแบบ

```
<bpws:elseif>
<bpws:condition><![CDATA[($input.A = $input.B and $input.A != $input.C) or
($input.C = $input.B and $input.C != $input.A) or ($input.A = $input.C and
$input.A != $input.B)]]></bpws:condition>
       <bpws:sequence name="HiddenSequence">
              <bpws:assign name="AssignIsosceles" validate="no">
                     <bpws:copy>
                         <bpws:from><![CDATA['Isosceles']]></bpws:from>
                         <bpws:to part="result" variable="output"/>
                     </bpws:copy>
              </bpws:assign>
       </bpws:sequence>
</bpws:elseif>
```

มิวแตนท์

```
<bpws:elseif>
<bpws:condition><![CDATA[($input.A > $input.B and $input.A != $input.C) or
($input.C = $input.B and $input.C != $input.A) or ($input.A = $input.C and
$input.A != $input.B)]]></bpws:condition>
       <bpws:sequence name="HiddenSequence">
              <bpws:assign name="AssignIsosceles" validate="no">
                     <bpws:copy>
                         <bpws:from><![CDATA['Isosceles']]></bpws:from>
                         <bpws:to part="result" variable="output"/>
                     </bpws:copy>
              </bpws:assign>
       </bpws:sequence>
</bpws:elseif>
```

ภาพที่ 3.6 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ERR

5) ตัวดำเนินการ ELL เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงตัวดำเนินการทางตรรกะ (and, or) ไปเป็นตัวดำเนินการอีกตัวหนึ่ง ดังตัวอย่างภาพที่ 3.7 โปรแกรมต้นแบบใช้เครื่องหมาย and ในขณะที่มิวแตนท์ใช้เครื่องหมาย or

โปรแกรมต้นแบบ

```
<bpws:condition><![CDATA[$input.A != $input.B and $input.A != $input.C
and $input.B != $input.C ]]></bpws:condition>
<bpws:sequence name="HiddenSequence1">
       <bpws:assign name="AssignScalene" validate="no">
              <bpws:copy>
                     <bpws:from><![CDATA['Scalene']]></bpws:from>
                     <bpws:to part="result" variable="output"/>
              </bpws:copy>
       </bpws:assign>
</bpws:sequence>
```

มิวแตนท์

```
<bpws:condition><![CDATA[$input.A != $input.B or $input.A != $input.C
and $input.B != $input.C ]]></bpws:condition>
<bpws:sequence name="HiddenSequence1">
       <bpws:assign name="AssignScalene" validate="no">
              <bpws:copy>
                     <bpws:from><![CDATA['Scalene']]></bpws:from>
                     <bpws:to part="result" variable="output"/>
              </bpws:copy>
       </bpws:assign>
</bpws:sequence>
```

ภาพที่ 3.7 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ELL

6) ตัวดำเนินการ ECC เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงนิพจน์ที่มีดำเนินการเกี่ยวกับเส้นทาง (/, //) ไปเป็นตัวดำเนินการอีกตัวหนึ่ง ดังตัวอย่างภาพที่ 3.8 โดยในโปรแกรมต้นแบบใช้เครื่องหมาย / ในนิพจน์ $reserveVehicleIn.itinerary/ota:itineraryRef ขณะที่ในมิวแตนท์ใช้เครื่องหมาย // ในนิพจน์ $reserveVehicleIn.itinerary//ota:itineraryRef

โปรแกรมต้นแบบ

```
<bpel:assign name="copyVehicleCancellation">
       <bpel:copy>
       <bpel:from>$reserveVehicleIn.itinerary/ota:itineraryRef</bpel:from>
              <bpel:to part="itinerary" variable="cancelVehicleIn"/>
       </bpel:copy>
</bpel:assign>
```

มิวแตนท์

```
<bpel:assign name="copyVehicleCancellation">
       <bpel:copy>
       <bpel:from>$reserveVehicleIn.itinerary//ota:itineraryRef</bpel:from
>
              <bpel:to part="itinerary" variable="cancelVehicleIn"/>
       </bpel:copy>
</bpel:assign>
```

ภาพที่ 3.8 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ECC

7) ตัวดำเนินการ ECN เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงค่าคงที่โดยเพิ่มกับลดค่าลงทีละ 1 และการเพิ่มกับลดตำแหน่ง 1 ตำแหน่ง ดังตัวอย่างภาพที่ 3.9 โปรแกรมต้นแบบในนิพจน์ $input.A > 100 มีค่าคงที่ 100 จะถูกสร้างมิวแตนท์ 2 ชุด คือ นิพจน์ $input.A > 99 ซึ่งถูกลดค่าจาก 100 ลง 1 และนิพจน์ $input.A > 10 ซึ่งถูกลดตำแหน่ง 1 ตำแหน่ง

โปรแกรมต้นแบบ

```
<bpws:condition
expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0">
<![CDATA[$input.A > 100 or $input.B > 100 or $input.C >
100]]></bpws:condition>
<bpws:elseif>
       <bpws:condition>…</bpws:condition>
                <bpws:assign name="AssignEquilateral" validate="no">
                    <bpws:copy>
                        <bpws:from><![CDATA['Equilateral']]></bpws:from>
                        <bpws:to part="result" variable="output"/>
                    </bpws:copy>
                </bpws:assign>
</bpws:elseif>
```

มิวแตนท์ 1

```
<bpws:condition
expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0">
<![CDATA[$input.A > 99 or $input.B > 100 or $input.C >
100]]></bpws:condition>
<bpws:elseif>
       <bpws:condition>…</bpws:condition>
                <bpws:assign name="AssignEquilateral" validate="no">
                    <bpws:copy>
                        <bpws:from><![CDATA['Equilateral']]></bpws:from>
                        <bpws:to part="result" variable="output"/>
                    </bpws:copy>
                </bpws:assign>
</bpws:elseif>
```

มิวแตนท์ 2

```
<bpws:condition
expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:xpath1.0">
<![CDATA[$input.A > 10 or $input.B > 100 or $input.C >
100]]></bpws:condition>
<bpws:elseif>
       <bpws:condition>…</bpws:condition>
                <bpws:assign name="AssignEquilateral" validate="no">
                    <bpws:copy>
                        <bpws:from><![CDATA['Equilateral']]></bpws:from>
                        <bpws:to part="result" variable="output"/>
                    </bpws:copy>
                </bpws:assign>
</bpws:elseif>
```

ภาพที่ 3.9 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ECN

8) ตัวดำเนินการ EMD เป็นตัวดำเนินการมิวเทชันชนิดนิพจน์ทำหน้าที่ดัดแปลงนิพจน์ช่วงเวลาโดยการแทนที่ด้วย 0 หรือ ค่าที่ลดลงครึ่งหนึ่งของค่าเริ่มต้น ดังตัวอย่างภาพที่ 3.10 ในโปรแกรมต้นแบบในกิจกรรม onAlarm ซึ่งกำหนดให้กระบวนการรอเป็นระยะเวลา P0DT12H นั่นคือ 12 ชั่วโมง ซึ่งจะถูกสร้างมิวแตนท์ 2 ชุด คือ มิวแตนท์ 1 ลดระยะเวลาการรอลงครึ่งหนึ่งเป็น P0DT6H และมิวแตนท์ 2 กำหนดค่าเวลาการรอเป็น 0 คือ P0DT0H

โปรแกรมต้นแบบ

```
<onAlarm>
      <for>'P0DT12H'</for>
</onAlarm>
```

มิวแตนท์ 1

```
<onAlarm>
      <for>'P0DT6H'</for>
</onAlarm>
```

มิวแตนท์ 2

```
<onAlarm>
      <for>'P0DT0H'</for>
</onAlarm>
```

ภาพที่ 3.10 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ EMD

9) ตัวดำเนินการ EMF เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงนิพจน์ทำหน้าที่ดัดแปลงนิพจน์เส้นตายโดยการแทนที่ด้วย 0 หรือ ค่าที่ลดลงครึ่งหนึ่งของค่าเริ่มต้น ดังตัวอย่างภาพที่ 3.11 โปรแกรมต้นแบบกิจกรรม onAlarm ซึ่งกำหนดในกระบวนการรอจนกระทั่ง 2011-10-28T19:28:19 ซึ่งจะถูกสร้างมิวแตนท์ 2 ชุด คือ มิวแตนท์ 1 ลดระยะเวลาการรอลงครึ่งหนึ่ง 2011-09-23T19:17:15 และมิวแตนท์ 2 กำหนดค่าเวลาการรอเป็นระยะเวลาปัจจุบัน คือ 2011-08-18T19:06:11

โปรแกรมต้นแบบ

```
<onAlarm>
      <wait until=”2011-10-28T19:28:19”></wait>
</onAlarm>
```

มิวแตนท์ 1

```
<onAlarm>
      <wait until=”2011-09-23T19:17:15”></wait>
</onAlarm>
```

มิวแตนท์ 2

```
<onAlarm>
      <wait until=”2011-08-18T19:06:11”></wait>
</onAlarm>
```

ภาพที่ 3.11 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ EMF

10) ตัวดำเนินการ AFP เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงแอตทริบิวต์ parallel ในกิจกรรม forEach โดยกำหนดค่าจาก no เป็น yes ดังตัวอย่างภาพที่ 3.12 โดยในโปรแกรมต้นแบบในกิจกรรม forEach มีแอตทริบิวต์ parallel เป็น no ส่วนในมิวแตนท์ถูกกำหนดค่าในแอตทริบิวต์ parallel เป็น yes

โปรแกรมต้นแบบ

```
<forEach parallel="no" ... >
      <startCounterValue>1</startCounterValue>
      <finalCounterValue>5</finalCounterValue>
      <scope>
            <invoke name="checkAvailable" ... />
      </scope>
</forEach>
```

มิวแตนท์

```
<forEach parallel="yes" ... >
      <startCounterValue>1</startCounterValue>
      <finalCounterValue>5</finalCounterValue>
      <scope>
            <invoke name="checkAvailable" ... />
      </scope>
</forEach>
```

ภาพที่ 3.12 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AFP

11) ตัวดำเนินการ ASF เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรม sequence เป็นกิจกรรม flow ดังตัวอย่างภาพที่ 3.13 โดยในโปรแกรมต้นแบบใช้กิจกรรม sequence ขณะที่มิวแตนท์ถูกเปลี่ยนเป็นกิจกรรม flow

โปรแกรมต้นแบบ

```
<sequence name="requestMessage">
      <receive name="receiveOrder" ... />
      <invoke name="paymentCheck" ... />
      …
</sequence>
```

มิวแตนท์

```
<flow name="requestMessage">
      <receive name="receiveOrder" ... />
      <invoke name="paymentCheck" ... />
      …
</flow>
```

ภาพที่ 3.13 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ASF

12) ตัวดำเนินการ AEL เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรมออก ดังตัวอย่างภาพที่ 3.14 โปรแกรมต้นแบบในกิจกรรม sequence ประกอบด้วยกิจกรรม receive และ invoke ขณะที่มิวแตนท์กิจกรรม invoke ถูกเอาออกจากกิจกรรม sequence

โปรแกรมต้นแบบ

```
<sequence name="requestMessage">
      <receive name="receiveOrder" ... />
      <invoke name="paymentCheck" ... />
      …
</sequence>
```

มิวแตนท์

```
<sequence name="requestMessage">
      <receive name="receiveOrder" ... />

      …
</sequence>
```

ภาพที่ 3.14 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AEL

13) ตัวดำเนินการ ACI เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยเปลี่ยนค่าของแอตทริบิวต์ createInstance จาก yes เป็น no ดังภาพที่ 3.15

โปรแกรมต้นแบบ

```
<process>
   <receive createInstance="yes" />
   <switch>
     <case name="usa">
       <invoke name="install_firmware" />
     </case>
     <case name="france">
       <invoke name="install_firmware" />
     </case>
   </switch>
   <reply variabele="status" />
</process>
```

มิวแตนท์

```
<process>
   <receive createInstance="no" />
   <switch>
     <case name="usa">
       <invoke name="install_firmware" />
     </case>
     <case name="france">
       <invoke name="install_firmware" />
     </case>
   </switch>
   <reply variabele="status" />
</process>
```

ภาพที่ 3.15 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ACI

14) ตัวดำเนินการ AIE เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม else หรือ elseif ออกจากกิจกรรม if ดังตัวอย่างภาพที่ 3.16 โดยในโปรแกรมต้นแบบในกิจกรรม if ซึ่งประกอบด้วย elseif และ else และในมิวแตนท์กิจกรรม elseif ถูกลบออกจากกิจกรรม if

โปรแกรมต้นแบบ

```
<if name="majorPayment30000Baht">
      <condition>$order > 30000</condition>
      <invoke name="calculatePC" ... />
      <elseif>
            <condition>$order > 15000</condition>
            <invoke name="calculateNotebook" ... />
      </elseif>
      <else>
            <reply name="sendPicture" ... />
      </else>
</if>
```

มิวแตนท์

```
<if name="majorPayment30000Baht">
      <condition>$order > 15000</condition>
      <invoke name="calculatePC" ... />



      <else>
            <reply name="sendPicture" ... />
      </else>
</if>
```

ภาพที่ 3.16 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AIE

15) ตัวดำเนินการ AIS เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยเปลี่ยนค่าของแอตทริบิวต์ isolated จาก yes เป็น no ดังตัวอย่างภาพที่ 3.17

โปรแกรมต้นแบบ

```
<scope isolated="yes" name="Scope" ... >
   <compensationHandler>
      <invoke name="ReportCancellationToAccounting" ... >...</
invoke>
   </compensationHandler>
   ...
</scope>
```

มิวแตนท์

```
<scope isolated="no" name="Scope" ... >
   <compensationHandler>
      <invoke name="ReportCancellationToAccounting" ... >
...</invoke>
   </compensationHandler>
   ...
</scope>
```

ภาพที่ 3.17 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AIS

16) ตัวดำเนินการ AWR เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยการแทนที่กิจกรรม while ด้วย repeatUntil ดังตัวอย่างภาพที่ 3.18 โปรแกรมต้นแบบใช้กิจกรรม while ในขณะที่มิวแตนท์ถูกเปลี่ยนโดยใช้กิจกรรม repeatUntil แทนที่

โปรแกรมต้นแบบ

```
<while>
      <condition>
            $iterations > 3
      </condition>
      <invoke name="incrementCounter" ... />
</while>
```

มิวแตนท์

```
<repeatUntil>
      <invoke name="incrementCounter" ... />
      <condition>
            $iterations > 3
      </condition>
</repeatUntil>
```

ภาพที่ 3.18 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AWR

17) ตัวดำเนินการ AJC เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม joinCondition ออก ดังตัวอย่างภาพที่ 3.19 โปรแกรมต้นแบบกิจกรรม targets ประกอบด้วยกิจกรรม target และ joinCondition และในมิวแตนท์กิจกรรม joinCondition ถูกลบออกจากกิจกรรม targets

โปรแกรมต้นแบบ

```
<targets>
      <target linkName="link1" />
      <joinCondition>
            $link1 and $link2
      </joinCondition>
</targets>
```

มิวแตนท์

```
<targets>
      <target linkName="link1" />


</targets>
```

ภาพที่ 3.19 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ AJC

18) ตัวดำเนินการ ASI เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยการสลับที่กิจกรรมในกิจกรรม sequence ดังตัวอย่างภาพที่ 3.20 โปรแกรมต้นแบบในกิจกรรม sequence ซึ่งประกอบด้วยหลายกิจกรรม ขณะที่มิวแตนท์กิจกรรม invoke ทั้ง 2 จะถูกสลับลำดับการทำงาน

โปรแกรมต้นแบบ

```
<sequence name="requestMessage">
      <receive name="receiveOrder" ... />
      <invoke name="paymentCheck" ... />
      <invoke name="serviceDelivery" ... />
      <reply name="sendConfirm" ... />
</sequence>
```

มิวแตนท์

```
<sequence name="requestMessage">
      <receive name="receiveOrder" ... />
      <invoke name="serviceDelivery" ... />
      <invoke name="paymentCheck" ... />
      <reply name="sendConfirm" ... />
</sequence>
```

ภาพที่ 3.20 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ ASI

19) ตัวดำเนินการ APM เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม onMessage ออก ดังตัวอย่างภาพที่ 3.21 โปรแกรมต้นแบบในกิจกรรม pick ประกอบด้วยกิจกรรมย่อยๆ เช่น onMessage และ onAlarm ในขณะที่มิวแตนท์กิจกรรมย่อย onMessage ถูกลบออกจากกิจกรรม pick

โปรแกรมต้นแบบ

```
<pick>
      <onMessage partnerLink="buyer" operation="newArticle" ... >
      ...
      </onMessage>
      <onMessage partnerLink="buyer" operation="fullRequest" ...
      > ...
      </onMessage>
      <onAlarm>
            <for> 'P3DT10H' </for>...
      </onAlarm>
</pick>
```

มิวแตนท์

```
<pick>
      <onMessage partnerLink="buyer" operation="newArticle" ... >
      ...
      </onMessage>


      <onAlarm>
            <for> 'P3DT10H' </for> ...
      </onAlarm>
</pick>
```

ภาพที่ 3.21 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ APM

20) ตัวดำเนินการ APA เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงกิจกรรม ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม onAlarm ออก ดังตัวอย่างภาพที่ 3.22 โปรแกรมต้นแบบในกิจกรรม pick ประกอบด้วยกิจกรรมย่อยๆ เช่น onMessage และ onAlarm ในขณะที่มิวแตนท์กิจกรรมย่อย onAlarm ถูกลบออกจากกิจกรรม pick

โปรแกรมต้นแบบ

```
<pick>
       <onMessage partnerLink="buyer" operation="newArticle" ... >
       ...
       </onMessage>
       <onMessage partnerLink="buyer" operation="fullRequest" ... > ...
       </onMessage>
       <onAlarm>
              <for> 'P3DT10H' </for> ...
       </onAlarm>
</pick>
```

มิวแตนท์

```
<pick>
       <onMessage partnerLink="buyer" operation="newArticle" ... >
       ...
       </onMessage>
       <onMessage partnerLink="buyer" operation="fullRequest" ... > ...
       </onMessage>


</pick>
```

ภาพที่ 3.22 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ APA

21) ตัวดำเนินการ XEE เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม onEvent ออก ดังตัวอย่างภาพที่ 3.23 โปรแกรมต้นแบบในกิจกรรม eventHandlers ซึ่งประกอบด้วยกิจกรรมย่อย onEvent ข้างในกิจกรรม eventHandlers ขณะที่มิวแตนท์กิจกรรม onEvent ถูกลบออกจากกิจกรรม eventHandlers

โปรแกรมต้นแบบ

```
<eventHandlers>
      <onEvent partnerLink="purchase" operation="statusOrder" ... >
            <scope> ... </scope>
      </onEvent>
      <onEvent partnerLink="purchase" operation="cancelOrder" ... >
            <scope> ... </scope>
      </onEvent>
</eventHandlers>
```

มิวแตนท์

```
<eventHandlers>



      <onEvent partnerLink="purchase" operation="cancelOrder" ... >
            <scope> ... </scope>
      </onEvent>
</eventHandlers>
```

ภาพที่ 3.23 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XEE

22) ตัวดำเนินการ XER เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม rethrow ออก ดังตัวอย่างภาพที่ 3.24 โปรแกรมต้นแบบเป็นกิจกรรม faultHandlers ซึ่งประกอบด้วยกิจกรรมย่อยๆ ดังภาพ ส่วนมิวแตนท์จะลบกิจกรรมย่อย rethrow ออกจากกิจกรรม faultHandlers

โปรแกรมต้นแบบ

```
<faultHandlers>
      <catch faultName="noPaperStockException"
      faultVariable="paperInStockVar">
      <rethrow> ... </rethrow>
      </catch>
      <catchAll> ... </catchAll>
</faultHandlers>
```

มิวแตนท์

```
<faultHandlers>
      <catch faultName="noPaperStockException"
      faultVariable="paperInStockVar">

      </catch>
      <catchAll> ... </catchAll>
</faultHandlers>
```

ภาพที่ 3.24 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XER

23) ตัวดำเนินการ XMC เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม compensationHandler ออก ดังตัวอย่างภาพที่ 3.25 โดยโปรแกรมต้นแบบมีกิจกรรม scope โดยมีกิจกรรมย่อยข้างในคือ faultHandlers และ compensationHandler ส่วนในมิวแตนท์กิจกรรมย่อย compensationHandler จะถูกลบออกจากกิจกรรม scope

โปรแกรมต้นแบบ

```
<scope name="buyer" ... >
      <faultHandlers>
            <catch faultName="rejectOrder"> ...
            </catch>
      </faultHandlers>
      <compensationHandler>
            <invoke partnerLink="vendor" ... />
      </compensationHandler>
</scope>
```

มิวแตนท์

```
<scope name="buyer" ... >
      <faultHandlers>
            <catch faultName="rejectOrder"> ...
            </catch>
      </faultHandlers>


</scope>
```

ภาพที่ 3.25 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XMC

24) ตัวดำเนินการ XMF เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม catch หรือ catchAll ออก ดังตัวอย่างภาพที่ 3.26 โปรแกรมต้นแบบมีกิจกรรม faultHandlers ซึ่งประกอบด้วยกิจกรรมย่อย catch และ catchAll ขณะที่มิวแตนท์กิจกรรมย่อย catchAll ถูกลบออกจาก faultHandlers

โปรแกรมต้นแบบ

```
<faultHandlers>
      <catch faultName="bookNotAvailable" ... >
      ...
      </catch>
      <catchAll> ... </catchAll>
</faultHandlers>
```

มิวแตนท์

```
<faultHandlers>
      <catch faultName="bookNotAvailable" ... >
      ...
      </catch>

</faultHandlers>
```

ภาพที่ 3.26 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XMF

25) ตัวดำเนินการ XMT เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมโดยลบกิจกรรม terminationHandler ออก ดังตัวอย่างภาพที่ 3.27 โปรแกรมต้นแบบมีกิจกรรม scope ซึ่งมีกิจกรรมย่อย eventHandlers และ terminationHandler ซึ่งในมิวแตนท์กิจกรรมย่อย terminationHandler ถูกลบออกจากกิจกรรม scope

โปรแกรมต้นแบบ

```
<bpel:scope name="Scope">
      <bpel:eventHandlers>
            <bpel:onEvent
      messageType="intf:EventRequestMessage" ...>
                  …
            </bpel:onEvent>
            <bpel:onAlarm>
                  <bpel:for><![CDATA['PT15S']]></bpel:for>
            </bpel:onAlarm>
      </bpel:eventHandlers>
      <bpel:terminationHandler>
            …
      </bpel:terminationHandler>
</bpel:scope>
```

มิวแตนท์

```
<bpel:scope name="Scope">
      <bpel:eventHandlers>
            <bpel:onEvent
      messageType="intf:EventRequestMessage" ...>
                  …
            </bpel:onEvent>
            <bpel:onAlarm>
                  <bpel:for><![CDATA['PT15S']]></bpel:for>
            </bpel:onAlarm>
      </bpel:eventHandlers>


</bpel:scope>
```

ภาพที่ 3.27 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XMT

26) ตัวดำเนินการ XTF เป็นตัวดำเนินการมิวเทชันชนิดดัดแปลงตัวจับความผิดปกติ ทำหน้าที่ดัดแปลงกิจกรรมแทนที่ชื่อแอตทริบิวต์ของ faultName ในกิจกรรม throw ด้วยอีกตัวหนึ่ง ดังตัวอย่างภาพที่ 3.28 โปรแกรมต้นแบบมีกิจกรรม if ซึ่งมีการจับข้อผิดพลาดด้วยกิจกรรมย่อย throw ด้วย faultName=”noStock” ส่วนในมิวแตนท์จะเปลี่ยนการจับข้อผิดพลาดโดยใช้ faultName=”articleNotAvailable”

โปรแกรมต้นแบบ

```
<if>
      <condition>$stock &gt; 100</condition>
      <flow> ... </flow>
      <elseif>
            <condition>$stock >= 0</condition>
            <throw faultName="noStock" ... />
      </elseif>
      <else>
            <throw faultName="articleNotAvailable" />
      </else>
</if>
```

มิวแตนท์

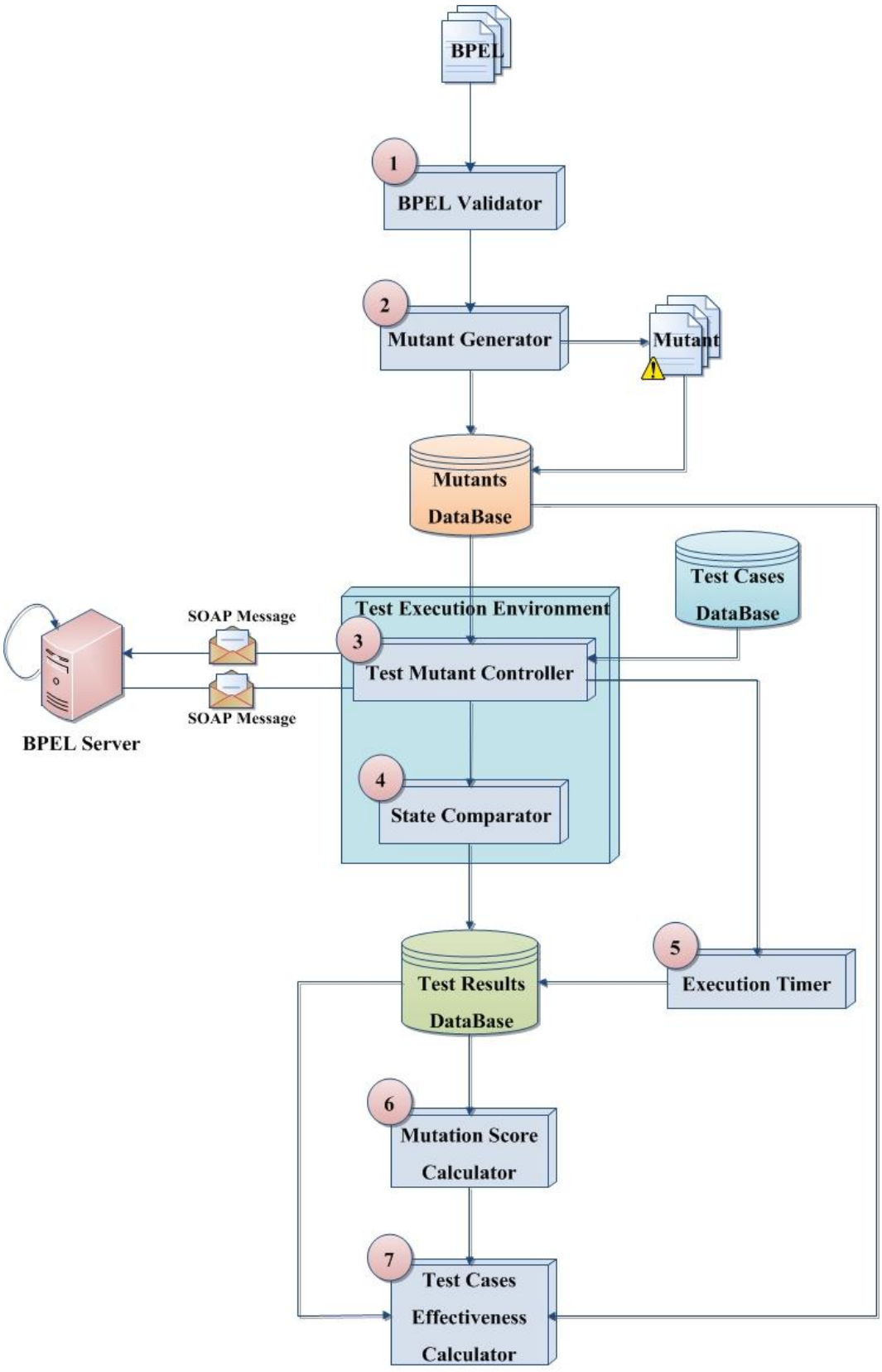
```
<if>
      <condition>$stock > 100</condition>
      <flow> ... </flow>
      <elseif>
            <condition>$stock >= 0</condition>
            <throw faultName="articleNotAvailable" ... />
      </elseif>
      <else>
            <throw faultName="articleNotAvailable" />
      </else>
</if>
```

ภาพที่ 3.28 การดัดแปลงโปรแกรมบีเพลด้วยตัวดำเนินการ XTF

3.3 การวิเคราะห์และออกแบบเครื่องมือ

3.3.1 แผนภาพเชิงแนวคิด

จากการศึกษางานวิจัย [7] ซึ่งเป็นการสร้างเครื่องมือทดสอบแบบสตรองมิวเทชันกับบีเพล พร้อมทั้งสร้างตัวดำเนินการมิวเทชันสำหรับนิพจน์ขึ้นมาใช้ในการดัดแปลงนิพจน์บีเพล และในงานวิจัย [10] ได้เสนอวิธีการทดสอบแบบวีคมิวเทชันซึ่งการทดสอบด้วยวิธีนี้มีข้อดีคือ ใช้เวลาในการทดสอบน้อยกว่าการทดสอบแบบมิวเทชันแบบดั้งเดิม ดังนั้นงานวิจัยนี้เสนอเครื่องมือที่ใช้สร้างมิวแตนท์และถูกทดสอบด้วยวิธีการทดสอบแบบวีคมิวเทชันชนิดต่างๆ ดังภาพที่ 3.29



ภาพที่ 3.29 แผนภาพเชิงแนวคิดของเครื่องมือ

จากภาพที่ 3.29 แผนภาพเชิงแนวคิดของเครื่องมือที่งานวิจัยนี้ได้สร้างขึ้น ซึ่งมีส่วนประกอบที่สำคัญอยู่ 7 ส่วน และมีการทำงานดังต่อไปนี้

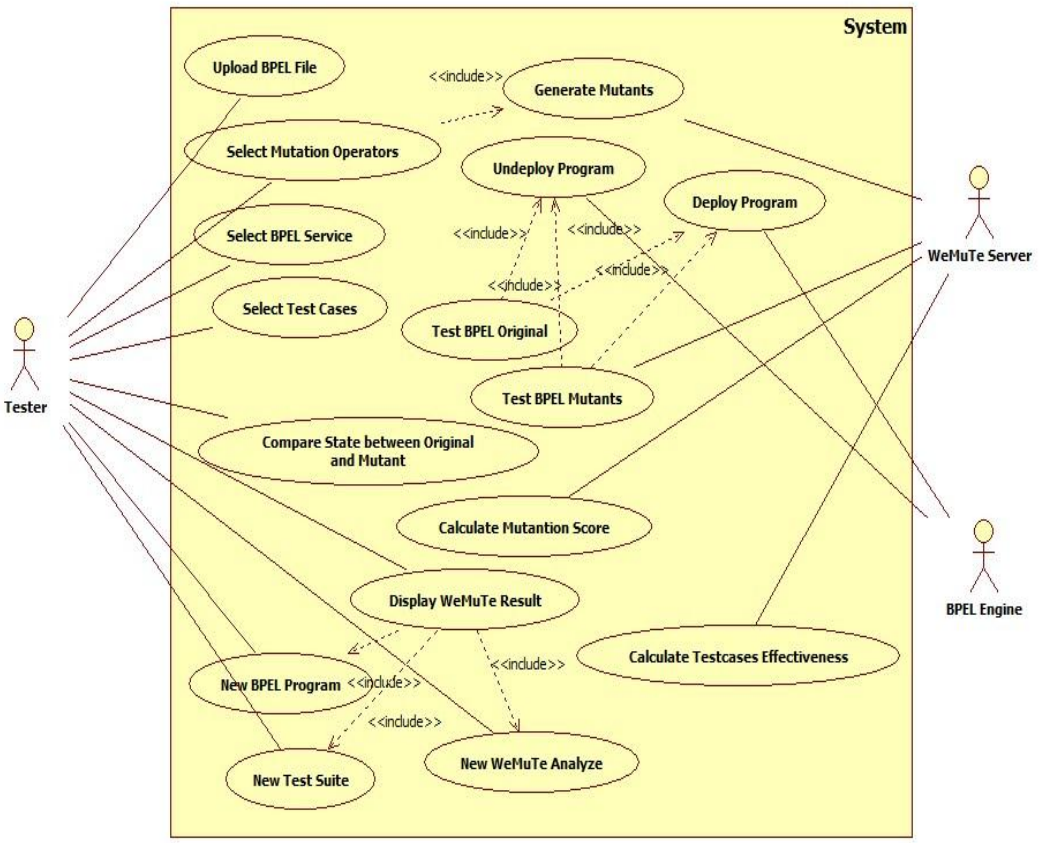
1) ตัวตรวจสอบเอกสารบีเพล (BPEL Validator) มีหน้าที่ตรวจสอบเอกสารบีเพลที่ถูกนำเข้ามาในเครื่องมือ โดยตรวจสอบว่าเอกสารบีเพลที่เข้ามามีความครบถ้วนสมบูรณ์หรือไม่ เอกสารที่ใช้ในการทดสอบประกอบด้วยเอกสารบีเพล เอกสารดับเบิลยูเอสดีแอล เอกสารดีพลอย และเอกสารเอ็กซ์เอ็มแอลสคีมาหรือไม่ เมื่อเอกสารบีเพลที่เข้ามาในระบบมีความสมบูรณ์ก็จะถูกนำไปสู่ขั้นตอนถัดไป
2) ตัวสร้างมิวแตนท์ (Mutant Generator) มีหน้าที่สร้างมิวแตนท์จากเอกสารบีเพลโดยหลักการสร้างมิวแตนท์จะพิจารณาจากตัวดำเนินการที่นักทดสอบได้เลือกใช้ ซึ่งเมื่อสร้างมิวแตนท์ออกมาจำนวนหนึ่งเครื่องมือก็จะเก็บมิวแตนท์เหล่านั้นไปไว้ที่ฐานข้อมูลมิวแตนท์ (Test Mutant Database)
3) ตัวควบคุมการทดสอบ (Test Mutant Controller) มีหน้าที่ควบคุมการทดสอบมิวแตนท์ โดยจะดึงข้อมูลมิวแตนท์จากฐานข้อมูลมิวแตนท์และกรณีทดสอบที่เตรียมไว้จากฐานข้อมูลกรณีทดสอบ จากนั้นส่งข้อมูลที่รวบรวมไปที่เครื่องประมวลผลบีเพล (BPEL Server) โดยอยู่ในรูปข้อความของโซป หลังจากที่เครื่องประมวลผลดำเนินการเสร็จเรียบร้อยแล้ว ก็จะส่งผลลัพธ์กลับมาในรูปแบบข้อความของโซปกลับมาที่ตัวควบคุมการทดสอบเพื่อเข้าสู่ขั้นตอนถัดไป
4) ตัวเปรียบเทียบสถานะ (State Comparator) มีหน้าที่เปรียบเทียบผลลัพธ์ของสถานะระหว่างมิวแตนท์และโปรแกรมต้นแบบ ซึ่งรูปแบบของการเปรียบเทียบก็จะขึ้นกับนักทดสอบที่เลือกวิธีการทดสอบ คือ EX-WEAK/1 ST-WEAK/1 BB-WEAK/1 และ BB-WEAK/N หลังจากเปรียบเทียบผลลัพธ์ระหว่างระหว่างมิวแตนท์และโปรแกรมต้นฉบับ ผลลัพธ์จะถูกเก็บไว้ที่ฐานข้อมูลของการทดสอบ (Test Results Database)
5) ตัวเก็บระยะเวลาการทำงาน (Execution Timer) ดำเนินการเก็บระยะเวลาที่ใช้ในการทดสอบมิวแตนท์ ซึ่งตัวเก็บระยะเวลาการทำงานเริ่มทำงานเมื่อตัวควบคุมการทดสอบส่งข้อมูลไปที่เครื่องประมวลผลบีเพล และหยุดเวลาเมื่อหลังจากผ่านการเปรียบเทียบผลลัพธ์เรียบร้อยแล้ว

6) ตัวคำนวนคะแนนมิวเทชัน (Mutation Score Calculator) ทำหน้าที่คำนวนคะแนนมิวเทชันโดยดึงข้อมูลมิวแตนท์ที่กำจัดได้ และมิวแตนท์ที่ยังคงอยู่จากฐานข้อมูลการทดสอบ และมิวแตนท์ที่เกิดทั้งหมดจากฐานข้อมูลมิวแตนท์มาใช้ในการคำนวน

7) ตัวคำนวนประสิทธิภาพของกรณีทดสอบ (Test Cases Effectiveness Calculator) ทำหน้าที่คำนวนประสิทธิภาพของกรณีทดสอบที่ใช้ในการทดสอบนั้น ซึ่งจะดึงข้อมูลคะแนนมิวเทชันที่คำนวนได้จากตัวคำนวนคะแนนมิวเทชัน

3.3.2 แผนภาพยูสเคส

แผนภาพยูสเคสของเครื่องมือจะอธิบายถึงภาพรวมและฟังก์ชันการทำงานของระบบ โดยจะเป็นมุมมองของนักทดสอบระบบ ดังภาพที่ 3.30 และรายละเอียดของยูสเคสดังตารางที่ 3.2 – 3.17



ภาพที่ 3.30 แผนภาพยูสเคสของเครื่องมือ

ตารางที่ 3.2 รายละเอียดยูสเคสนำเข้าโปรแกรมต้นแบบ

รหัสยูสเคส	WEMUTE001
ยูสเคส	Upload BPEL File
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	โปรแกรมต้นฉบับเก็บอยู่ในรูปเอกสารซิป
ขั้นตอน	1. นักทดสอบเรียกใช้โปรแกรมผ่านทางเว็บเบราว์เซอร์ 2. ระบบสร้างหน้าต่างสำหรับนำเข้าโปรแกรมต้นฉบับ 3. นักทดสอบอัพโหลดโปรแกรมต้นฉบับ 4. ระบบบันทึกโปรแกรมต้นฉบับไว้บนเครื่องบริการเว็บ (Web Server)
เงื่อนไขภายหลัง	โปรแกรมต้นฉบับได้รับการบันทึกอยู่บนเครื่องบริการเว็บ

ตารางที่ 3.3 รายละเอียดยูสเคสเลือกตัวดำเนินการมิวเทชัน

รหัสยูสเคส	WEMUTE002
ยูสเคส	Select Mutation Operator
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	Generate Mutant
เงื่อนไขก่อนหน้า	
ขั้นตอน	1. นักทดสอบเลือกแนววิธีการทดสอบวีคมิวเทชัน 2. ระบบจะเลือกตัวดำเนินการมิวเทชันที่เกี่ยวข้องกับการทดสอบวีคมิวเทชันชนิดนั้นๆ โดยอัตโนมัติ 3. นักทดสอบกดปุ่มเพื่อสร้างมิวแตนท์
เงื่อนไขภายหลัง	ระบบสร้างมิวแตนท์ครบทุกตัวตามตัวดำเนินการมิวเทชันที่เลือก

ตารางที่ 3.4 รายละเอียดยูสเคสสร้างมิวแตนท์

รหัสยูสเคส	WEMUTE003
ยูสเคส	Generate Mutant
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	
เงื่อนไขก่อนหน้า	โปรแกรมต้นฉบับเก็บอยู่บนเครื่องบริการเว็บ
ขั้นตอน	1. ระบบสร้างมิวแตนท์ตามตัวดำเนินการมิวเทชันที่นักสอบเลือก 2. ระบบเก็บมิวแตนท์ที่สร้างขึ้นทั้งหมดไว้ใน Mutant Database
เงื่อนไขภายหลัง	มิวแตนท์ถูกเก็บไว้ที่ Mutant Database

ตารางที่ 3.5 รายละเอียดยูสเคสเลือกโปรแกรมบีเพล

รหัสยูสเคส	WEMUTE004
ยูสเคส	Select BPEL Service
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	
เงื่อนไขก่อนหน้า	
ขั้นตอน	ระบบแสดงรายละเอียดของโปรแกรมบีเพลที่ต้องการทดสอบ
เงื่อนไขภายหลัง	นักทดสอบกดปุ่มเพื่อเข้าสู่ขั้นตอนต่อไป

ตารางที่ 3.6 รายละเอียดยูสเคสเลือกกรณีทดสอบ

รหัสยูสเคส	WEMUTE005
ยูสเคส	Select Test Cases
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	Test BPEL Origianl, Test BPEL Mutants, Deploy Program, Undeploy Program
เงื่อนไขก่อนหน้า	กรณีทดสอบได้ถูกติดตั้งไว้ที่ TestCases Database

ตารางที่ 3.6 รายละเอียดยูสเคสเลือกกรณีทดสอบ (ต่อ)

ขั้นตอน	1. นักทดสอบโหลดข้อมูลกรณีทดสอบมาเพื่อเตรียมทดสอบ 2. นักทดสอบกดปุ่มเพื่อทดสอบโปรแกรมต้นแบบ (Test BPEL Original) 3. ระบบทำการติดตั้งโปรแกรมต้นแบบ (Deploy Program) 4. ระบบเก็บผลลัพธ์ไว้ที่ Test Result Database 5. ระบบยกเลิกการติดตั้งโปรแกรมบีเพล (Undeploy Mutant) 5. นักทดสอบกดปุ่มเพื่อทดสอบมิวแตนท์ (Test BPEL Mutants) 6. ระบบทำการติดตั้งมิวแตนท์ (Deploy Program) 7. ระบบเก็บผลลัพธ์ไวที่ Test Result Database 8. ระบบยกเลิกการติดตั้งมิวแตนท์ 9. ระบบทำซ้ำในข้อ 6 – ข้อ 9 จะกว่าจะครบทุกมิวแตนท์
เงื่อนไขภายหลัง	มิวแตนท์ทุกตัวถูกทดสอบ

ตารางที่ 3.7 รายละเอียดยูสเคสทดสอบโปรแกรมต้นแบบ

รหัสยูสเคส	WEMUTE006
ยูสเคส	Test BPEL Original
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อติดตั้งโปรแกรมต้นแบบไปที่ BPEL Server
ยูสเคสที่สัมพันธ์	
เงื่อนไขก่อนหน้า	
ขั้นตอน	ระบบนำเอกสารซิปของโปรแกรมต้นแบบไปติดตั้งใน BPEL Server
เงื่อนไขภายหลัง	โปรแกรมต้นแบบถูกติดตั้งบน BPEL Server

ตารางที่ 3.8 รายละเอียดยูสเคสทดสอบบีเพลมิวแตนท์

รหัสยูสเคส	WEMUTE007
ยูสเคส	Test BPEL Mutant
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อติดตั้งมิวแตนท์ไปที่ BPEL Server
ยูสเคสที่สัมพันธ์	

ตารางที่ 3.8 รายละเอียดยูสเคสทดสอบบีเพลมิวแตนท์ (ต่อ)

เงื่อนไขก่อนหน้า	
ขั้นตอน	ระบบนำเอกสารซิปของมิวแตนท์ไปติดตั้งใน BPEL Server
เงื่อนไขภายหลัง	มิวแตนท์ถูกติดตั้งบน BPEL Server

ตารางที่ 3.9 รายละเอียดยูสเคสติดตั้งโปรแกรม

รหัสยูสเคส	WEMUTE007
ยูสเคส	Deploy Program
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	
เงื่อนไขก่อนหน้า	โปรแกรมต้นฉบับเก็บอยู่ในรูปไฟล์ซิป
ขั้นตอน	
เงื่อนไขภายหลัง	โปรแกรมต้นฉบับได้รับการบันทึกอยู่บนเครื่องบริการเว็บ

ตารางที่ 3.10 รายละเอียดยูสเคสยกเลิกการติดตั้งโปรแกรม

รหัสยูสเคส	WEMUTE007
ยูสเคส	Undeploy Program
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อนำโปรแกรมต้นแบบเข้าสู่ระบบ
ยูสเคสที่สัมพันธ์	
เงื่อนไขก่อนหน้า	โปรแกรมต้นฉบับเก็บอยู่ในรูปไฟล์ซิป
ขั้นตอน	
เงื่อนไขภายหลัง	โปรแกรมต้นฉบับได้รับการบันทึกอยู่บนเครื่องบริการเว็บ

ตารางที่ 3.11 รายละเอียดยูสเคสเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์

รหัสยูสเคส	WEMUTE008
ยูสเคส	Compare state between Original and Mutant
แอคเตอร์	ระบบ (WeMuTe Server)

ตารางที่ 3.11 รายละเอียดยูสเคสเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์ (ต่อ)

เป้าหมาย	เพื่อเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์ ที่เป็นนิพจน์และประพจน์
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	ทุกๆมิวแตนท์ถูกทดสอบ
ขั้นตอน	1. ระบบแทนที่ข้อมูลของกรณีทดสอบลงในตัวแปรในนิพจน์หรือประพจน์ของโปรแกรมต้นแบบและมิวแตนท์ 2. ระบบนำผลลัพธ์ของโปรแกรมต้นแบบและมิวแตนท์มาเปรียบเทียบกัน
เงื่อนไขภายหลัง	ผลลัพธ์ถูกเก็บไว้ใน Test Result Database

ตารางที่ 3.12 รายละเอียดยูสเคสคำนวนคะแนนมิวเทชัน

รหัสยูสเคส	WEMUTE009
ยูสเคส	Calculate Mutation Score
แอคเตอร์	ระบบ (WeMuTe Server)
เป้าหมาย	เพื่อนำคำนวนหาคะแนนมิวเทชัน
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	
ขั้นตอน	ระบบคำนวนคะแนนมิวเทชัน
เงื่อนไขภายหลัง	ระบบคำนวนคะแนนมิวเทชันเสร็จเรียบร้อย

ตารางที่ 3.13 รายละเอียดยูสเคสคำนวนประสิทธิภาพของกรณีทดสอบ

รหัสยูสเคส	WEMUTE010
ยูสเคส	Calculate TestCases Effectiveness
แอคเตอร์	ระบบ (WeMuTe Server)
เป้าหมาย	เพื่อนำคำนวนหาประสิทธิภาพของกรณีทดสอบ
ยูสเคสที่สัมพันธ์	คำนวนคะแนนมิวเทชัน
เงื่อนไขก่อนหน้า	ระบบคำนวนคะแนนมิวเทชัน
ขั้นตอน	ระบบคำนวนประสิทธิภาพของกรณีทดสอบ
เงื่อนไขภายหลัง	ระบบคำนวนประสิทธิภาพของกรณีทดสอบเสร็จเรียบร้อย

ตารางที่ 3.14 รายละเอียดยูสเคสแสดงผลลัพธ์ของการทดสอบ

รหัสยูสเคส	WEMUTE011
ยูสเคส	Display Result
แอคเตอร์	ระบบ (WeMuTe Server)
เป้าหมาย	เพื่อรายงานผลของการทดสอบ
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	ระบบแสดงรายละเอียดมิวแตนท์ที่ยังอยู่ และถูกกำจัดหรือไม่
ขั้นตอน	ระบบแสดงหน้ารายงานผลการทดสอบ
เงื่อนไขภายหลัง	-

ตารางที่ 3.15 รายละเอียดยูสเคสเริ่มทดสอบโปรแกรมบีเพลใหม่

รหัสยูสเคส	WEMUTE012
ยูสเคส	New BPEL Program
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อกลับไปเริ่มทดสอบโปรแกรมบีเพลโปรแกรมใหม่
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	ระบบแสดงผลลัพธ์ของการทดสอบ
ขั้นตอน	ระบบกลับไปที่หน้าจอเริ่มอัพโหลดโปรแกรมบีเพลใหม่
เงื่อนไขภายหลัง	-

ตารางที่ 3.16 รายละเอียดยูสเคสเริ่มการวิเคราะห์โปรแกรมบีเพลใหม่

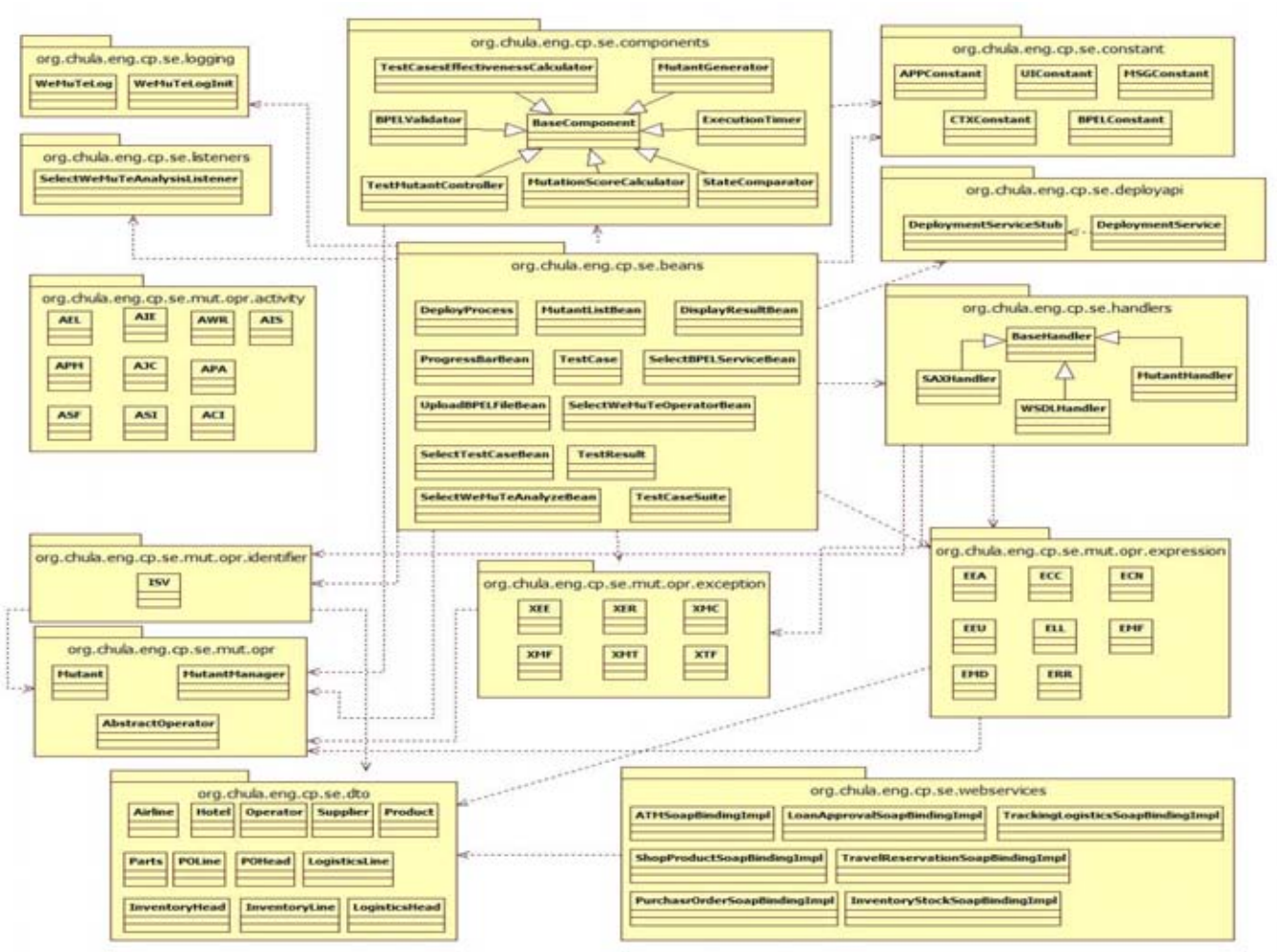
รหัสยูสเคส	WEMUTE013
ยูสเคส	New WeMuTe Analyze
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อกลับไปเริ่มกระบวนการวิเคราะห์ใหม่ด้วยโปรแกรมบีเพลเดิม
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	ระบบแสดงผลลัพธ์ของการทดสอบ
ขั้นตอน	ระบบกลับไปที่หน้าจอกลับไปเลือกกระบวนการวิเคราะห์ใหม่
เงื่อนไขภายหลัง	-

ตารางที่ 3.17 รายละเอียดยูสเคสใช้ชุดกรณีทดสอบชุดใหม่

รหัสยูสเคส	WEMUTE014
ยูสเคส	New Test Suite
แอคเตอร์	นักทดสอบ (Tester)
เป้าหมาย	เพื่อกลับไปเริ่มใช้กรณีทดสอบชุดใหม่
ยูสเคสที่สัมพันธ์	-
เงื่อนไขก่อนหน้า	ระบบแสดงผลลัพธ์ของการทดสอบ
ขั้นตอน	ระบบกลับไปที่หน้าจอเลือกกรณีทดสอบใหม่
เงื่อนไขภายหลัง	-

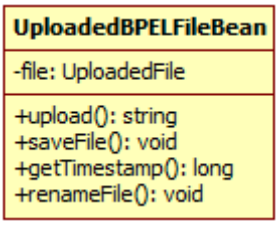
3.3.3 แผนภาพคลาส

แผนภาพคลาสใช้แสดงรายละเอียดต่างๆของคลาส และความสัมพันธ์ของคลาสในระบบ ดังภาพที่ 3.31



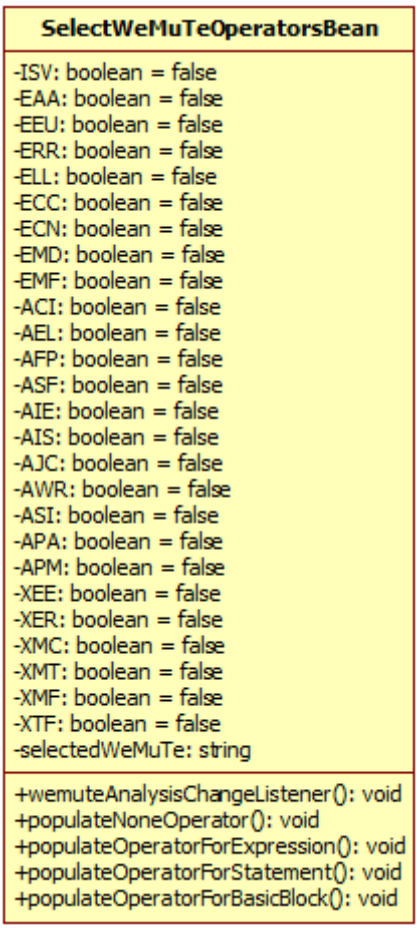
ภาพที่ 3.31 แผนภาพคลาสของเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล

1) คลาส UploadBPELFileBean คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ รับไฟล์บีเพลที่อยู่ในรูปเอกสารซิป รายละเอียดดังภาพที่ 3.32



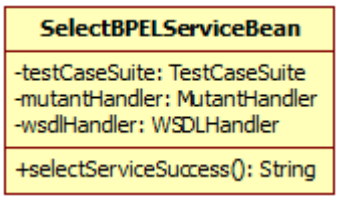
ภาพที่ 3.32 คลาส UploadedBPELFileBean

2) คลาส SelectWeMuTeOpeatorsBean คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ โดยรับชนิดวิธีการทดสอบวีคมิวเทชันจากนักทดสอบ รายละเอียดดังภาพที่ 3.33



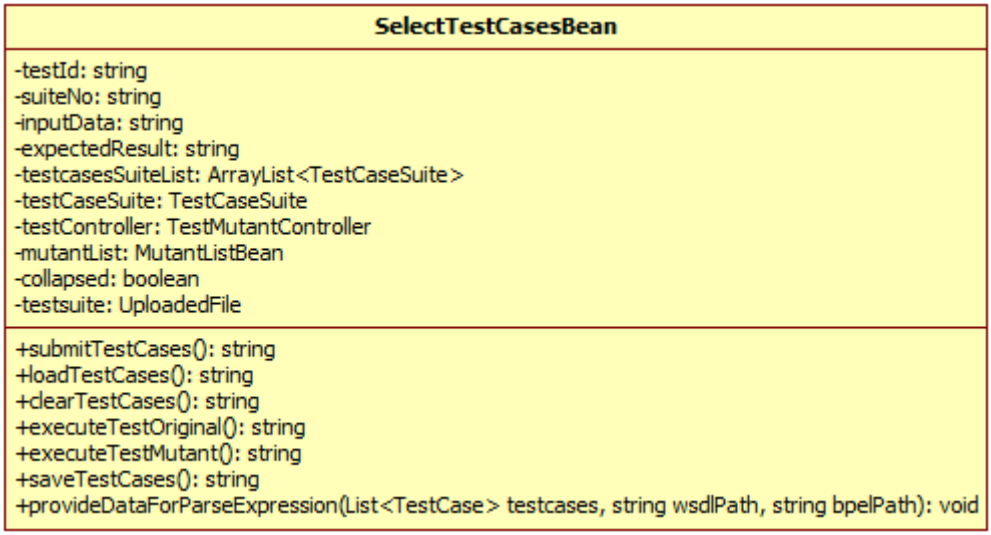
ภาพที่ 3.33 คลาส SelectWeMuTeOperatorsBean

3) คลาส SelectBPELServiceBean คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ โดยแสดงรายละเอียดของโปรแกรมบีเพลที่ต้องการทดสอบ รายละเอียดดังภาพที่ 3.34



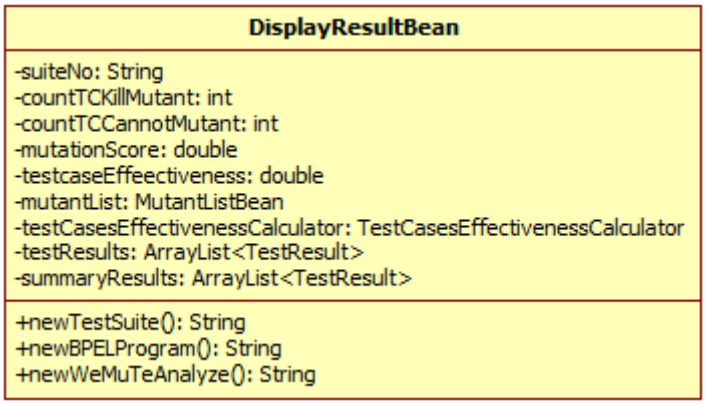
ภาพที่ 3.34 คลาส SelectBPELServiceBean

4) คลาส SelectTestCasesBean คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ ในการโหลดข้อมูลกรณีทดสอบ ทดสอบโปรแกรมต้นแบบ และทดสอบมิวแตนท์ รายละเอียดของคลาสดังภาพที่ 3.35



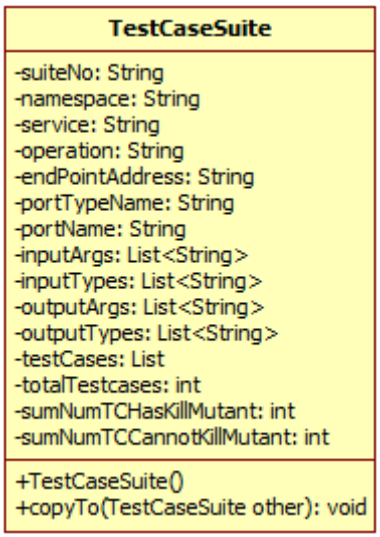
ภาพที่ 3.35 คลาส SelectTestCasesBean

5) คลาส DisplayResultBean คือ คลาสที่ทำหน้าที่ติดต่อกับนักทดสอบ แสดงผลลัพธ์ของการทดสอบ รายละเอียดของคลาสดังภาพที่ 3.36



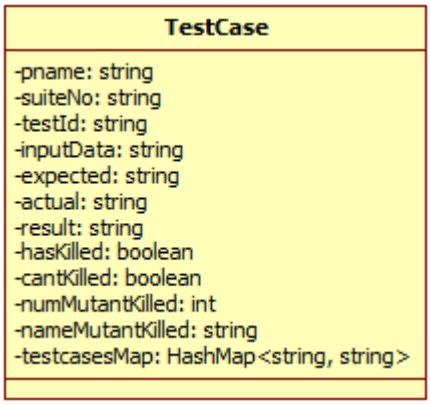
ภาพที่ 3.36 คลาส DisplayResultBean

6) คลาส TestCaseSuite คือ คลาสที่ทำหน้าที่เก็บข้อมูลของกรณีทดสอบทั้งหมด และข้อมูลของเซอร์วิซที่ต้องการเรียกใช้ รายละเอียดของคลาสดังภาพที่ 3.37



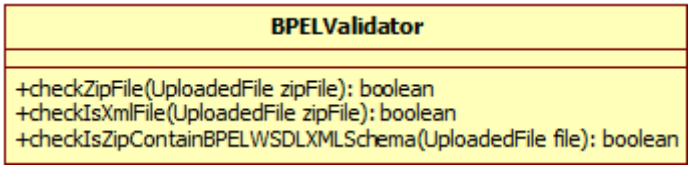
ภาพที่ 3.37 คลาส TestCaseSuite

7) คลาส TestCase คือ คลาสที่ทำหน้าที่เก็บข้อมูลของกรณีทดสอบ รายละเอียดของคลาสดังภาพที่ 3.38



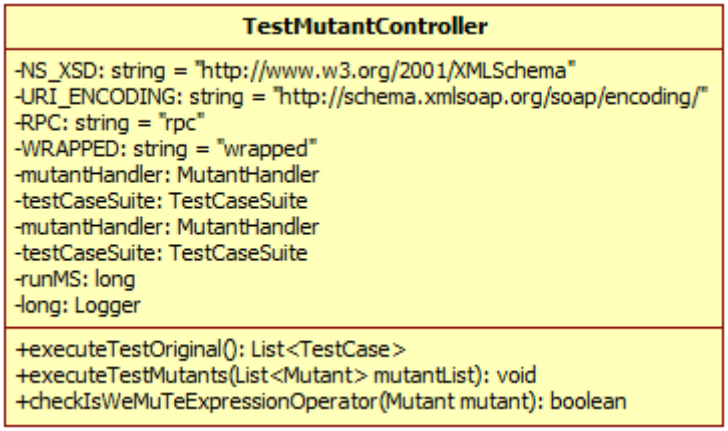
ภาพที่ 3.38 คลาส TestCase

8) คลาส BPELValidator คือ คลาสที่ทำหน้าที่ตรวจสอบเอกสารซิปของโปรแกรมบีเพลว่ามีเอกสารบีเพลและเอกสารวิซเดิล รายละเอียดของคลาสดังภาพที่ 3.39



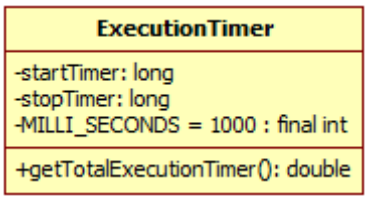
ภาพที่ 3.39 คลาส BPELValidator

9) คลาส TestMutantController คือ คลาสที่ทำหน้าที่ทดสอบโปรแกรมต้นแบบและมิวแตนท์ ซึ่งรายละเอียดของคลาสดังภาพที่ 3.40



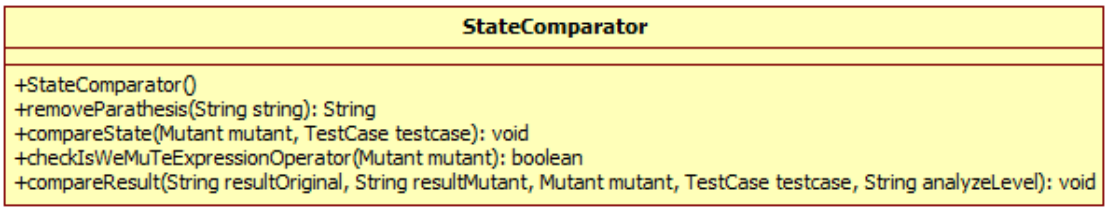
ภาพที่ 3.40 คลาส TestMutantController

10) คลาส ExecutionTimer คือ คลาสที่ทำหน้าที่เก็บเวลาที่ใช้ในการทดสอบทั้งหมด รายละเอียดของคลาสดังภาพที่ 3.41



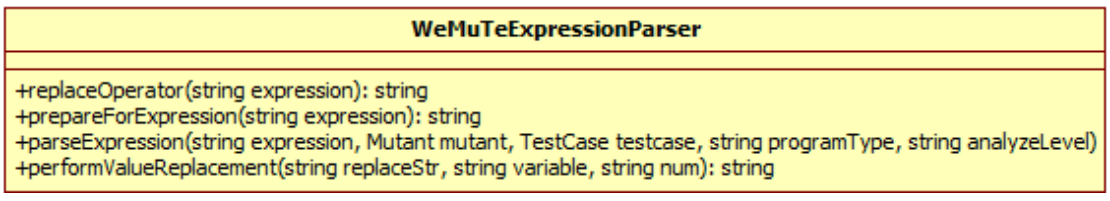
ภาพที่ 3.41 คลาส ExecutionTimer

11) คลาส StateComparator คือ คลาสที่ทำหน้าที่แทนที่ข้อมูลของกรณีทดสอบลงในนิพจน์หรือประพจน์ จากนั้นจึงเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์ รายละเอียดดังภาพที่ 3.42



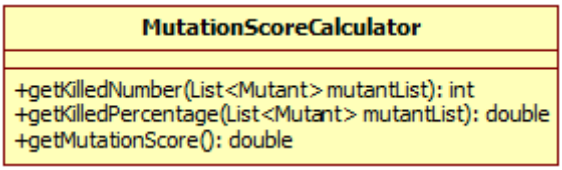
ภาพที่ 3.42 คลาส StateComparator

12) คลาส WeMuTeExpressionParser คือ คลาสที่ทำหน้าที่แทนที่ตัวดำเนินการทางคณิตศาสตร์หรือตรรกะเพื่อประมวลผลลัพธ์ของนิพจน์หรือประพจน์นั้น รายละเอียดของคลาสดังภาพที่ 3.43



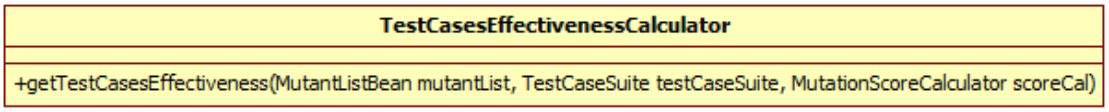
ภาพที่ 3.43 คลาส WeMuTeExpressionParser

13) คลาส MutationScoreCalculator คือ คลาสที่ทำหน้าที่คำนวนหาคะแนนมิวเทชัน รายละเอียดของคลาสดังภาพที่ 3.44



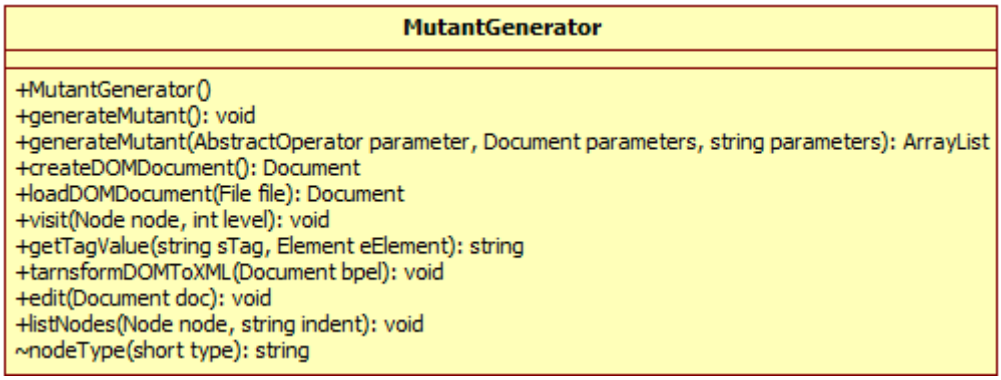
ภาพที่ 3.44 คลาส MutationScoreCalculator

14) คลาส TestCasesEffectivenessCalculator คือ คลาสที่ทำหน้าที่คำนวนหาประสิทธิภาพของกรณีทดสอบ รายละเอียดของคลาสดังภาพที่ 3.45



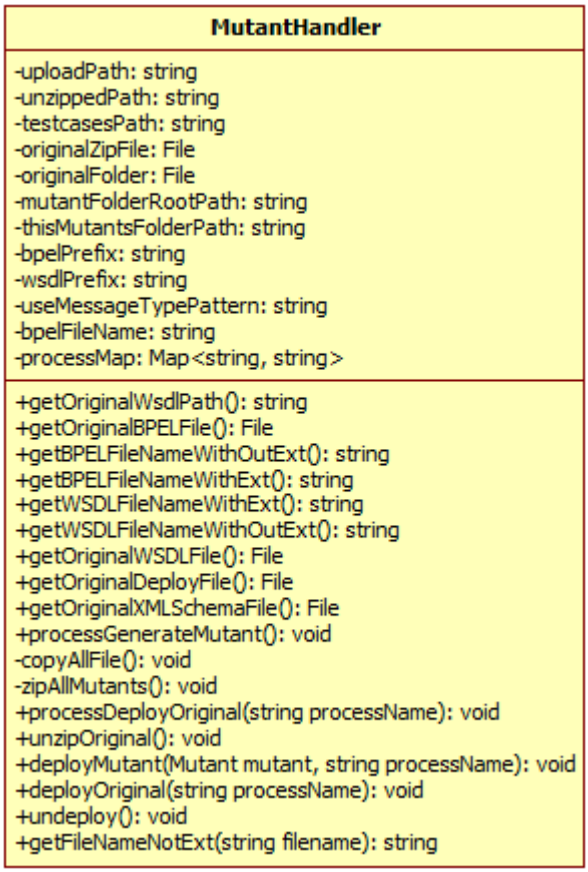
ภาพที่ 3.45 คลาส TestCasesEffectivenessCalculator

15) คลาส MutantGenerator คือ คลาสที่ทำหน้าที่เตรียมการก่อนที่จะทำการสร้างมิวแตนท์และทำการสร้างเอกสารดอมจากเอกสารเอ็กซ์เอ็มแอล รายละเอียดของคลาสดังภาพที่ 3.46



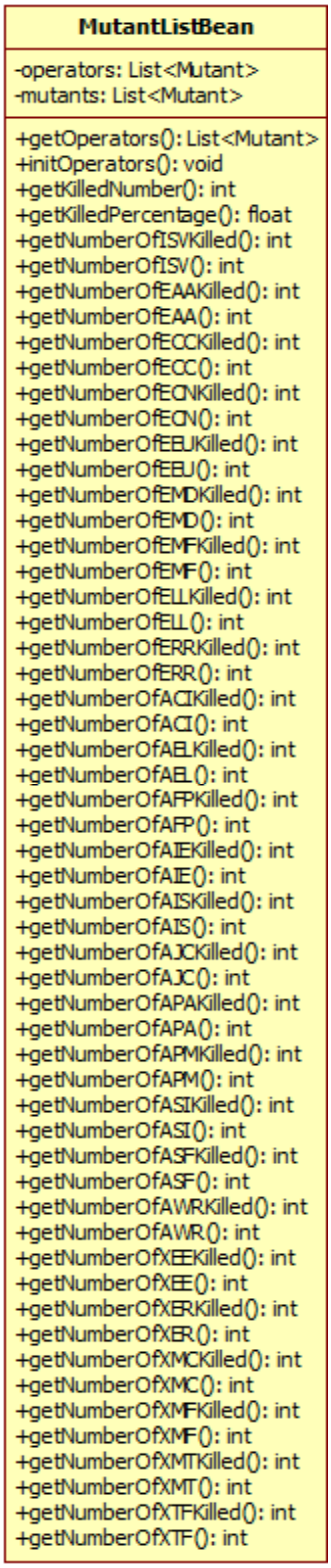
ภาพที่ 3.46 คลาส MutantGenerator

16) คลาส MutantHandler คือ คลาสที่ทำหน้าที่ติดตั้งโปรแกรมและยกเลิกโปรแกรมหาค่าต่างๆของเอกสารบีเพลและวิซเดิล รายละเอียดของคลาสดังภาพที่ 3.47



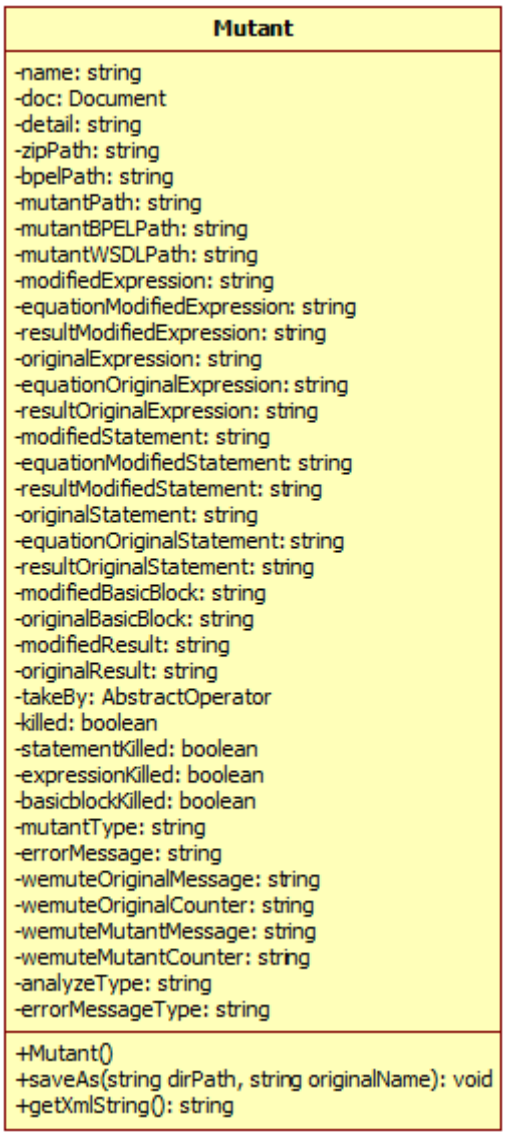
ภาพที่ 3.47 คลาส MutantHandler

17) คลาส MutantListBean คือ คลาสที่ทำหน้าที่เก็บข้อมูลของมิวแตนท์ที่สร้างขึ้นทั้งหมด และข้อมูลมิวแตนท์ที่ถูกกำจัดในแต่ละตัวดำเนินการ รายละเอียดของคลาสดังภาพที่ 3.48



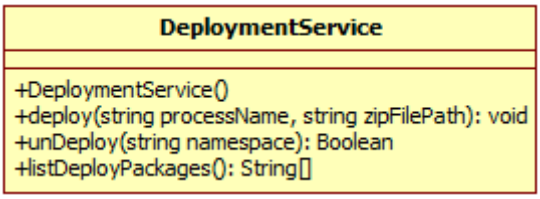
ภาพที่ 3.48 คลาส MutantListBean

18) คลาส Mutant คือ คลาสที่ทำหน้าที่เก็บรายละเอียดของมิวแตนท์ ดังรายละเอียดภาพที่ 3.49



ภาพที่ 3.49 คลาส Mutant

18) คลาส DeploymentService คือ คลาสที่ทำหน้าที่ติดตั้งและยกเลิกโปรแกรมต้นแบบและมิวแตนท์ รายละเอียดของคลาสดังภาพที่ 3.50

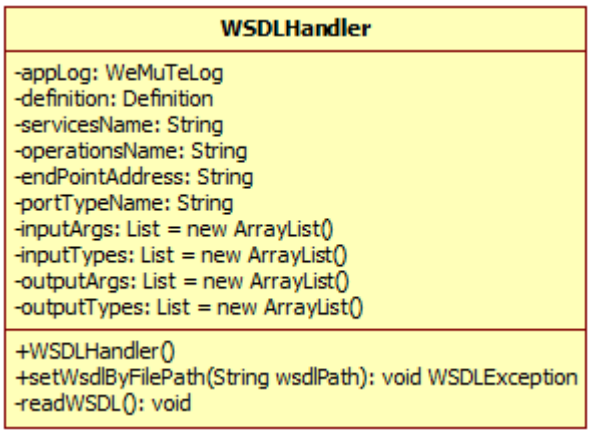


ภาพที่ 3.50 คลาส DeploymentService

19) คลาส DeploymentServiceStub คือ คลาสที่ทำหน้าที่ติดตั้งและยกเลิกโปรแกรมต้นแบบและมิวแตนท์ รายละเอียดของคลาสดังรูป 3.51

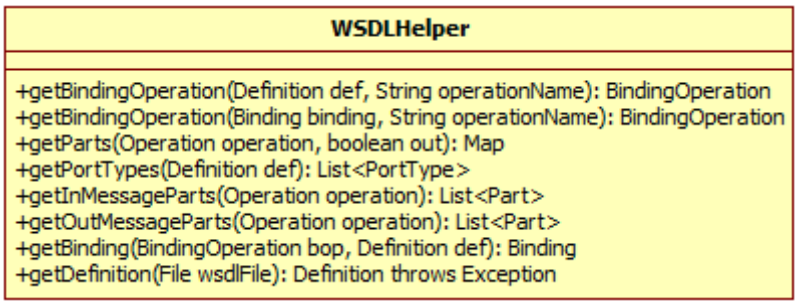
ภาพที่ 3.51 คลาส DeploymentServiceStub

20) คลาส WSDLHandler คือ คลาสที่ทำหน้าที่ในการแปลงเอกสารวิซเดิลไปเอกสารดอมเพื่อให้โปรแกรมสามารถอ่านข้อมูลในเอกสารได้ รายละเอียดของคลาสดังรูป 3.52



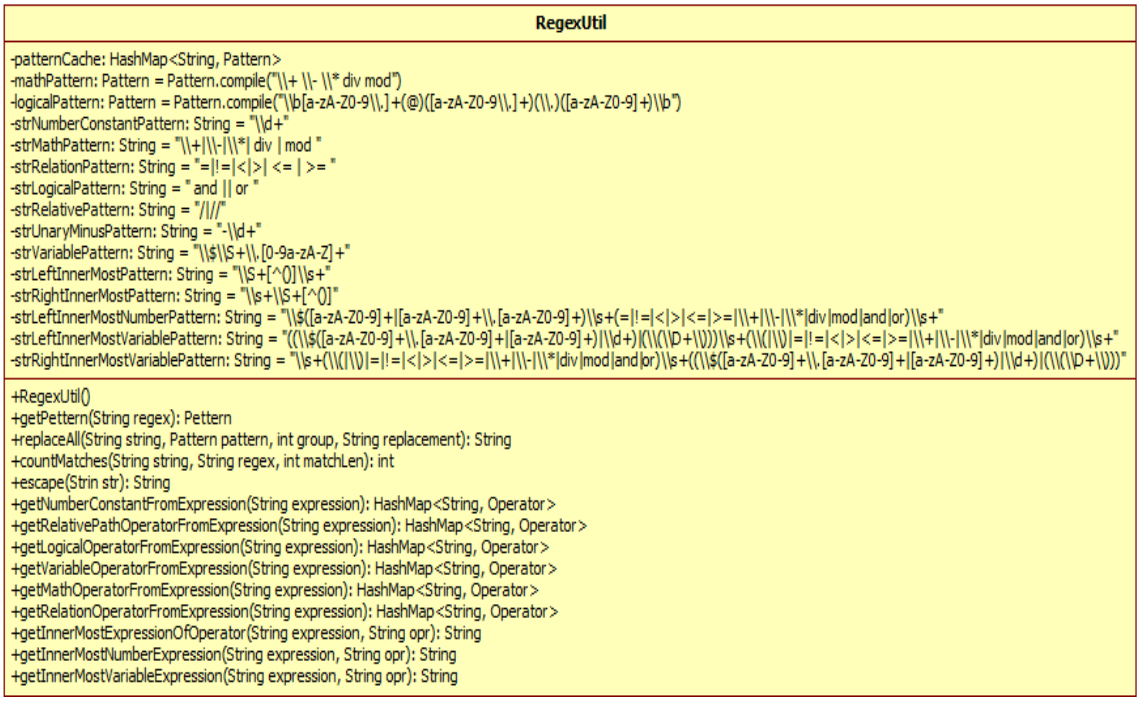
ภาพที่ 3.52 คลาส WSDLHandler

21) คลาส WSDLHelper คือ คลาสที่ทำหน้าที่ในการแปลงเอกสารวิซเดิลไปเอกสารดอมเพื่อให้โปรแกรมสามารถอ่านข้อมูลในเอกสารได้ รายละเอียดของคลาสดังรูป 3.53



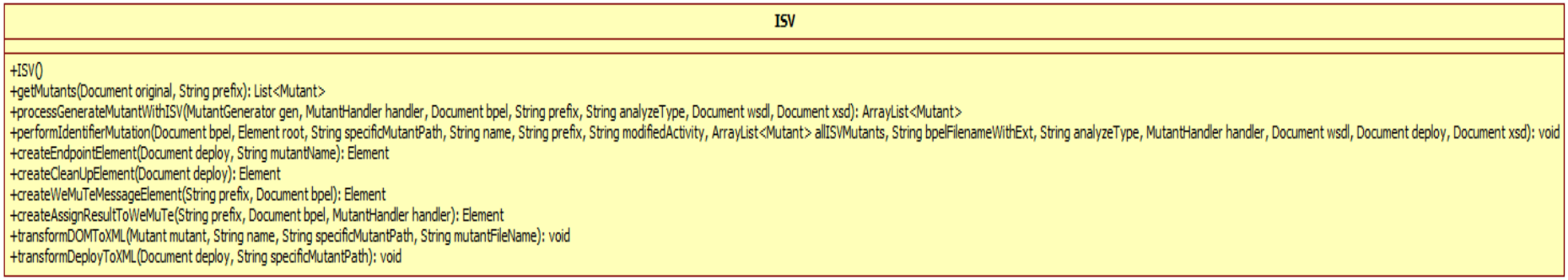
ภาพที่ 3.53 คลาส WSDLHelper

22) คลาส RegexUtil คือ คลาสที่ทำหน้าที่หาตัวดำเนินการในนิพจน์และประพจน์รวมไปถึงหานิพจน์ในสุดอีกด้วย รายละเอียดของคลาสดังรูป 3.54



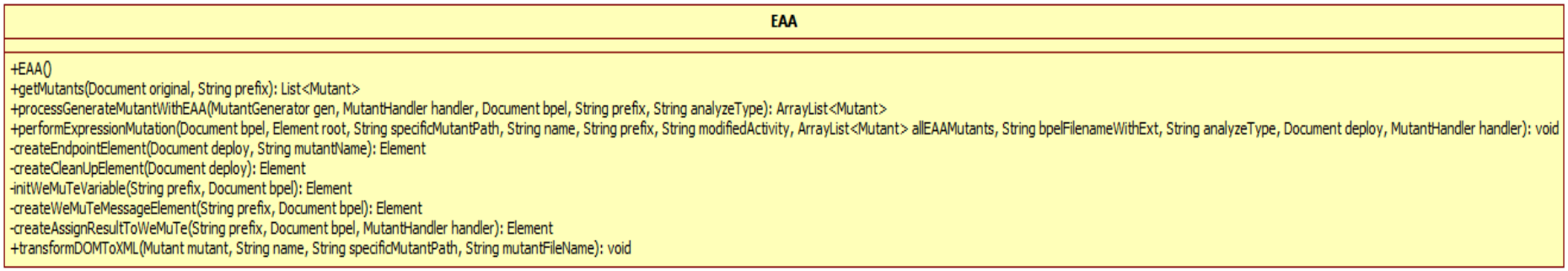
ภาพที่ 3.54 คลาส RegexUtil

23) คลาส ISV คือ คลาสที่ทำหน้าที่ดัดแปลงตัวระบุโดยการ สลับตัวแปรในนิพจน์หรือประพจน์ที่เป็นชนิดข้อมูลเหมือนกัน รายละเอียดของคลาสดังภาพที่ 3.55



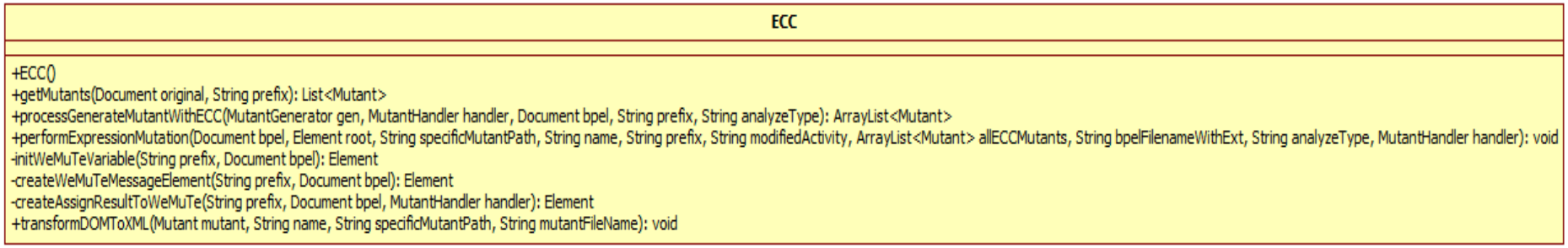
ภาพที่ 3.55 คลาส ISV

24) คลาส EAA คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ที่มีตัวดำเนินการทางคณิตศาสตร์ (+, -, *, div, mod) โดยการเปลี่ยนเป็นอีกตัวดำเนินการหนึ่ง รายละเอียดของคลาสดังภาพที่ 3.56



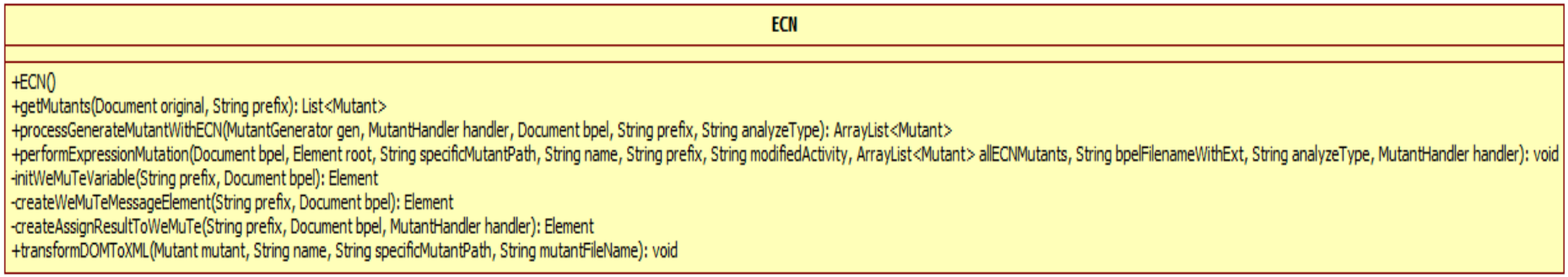
ภาพที่ 3.56 คลาส EAA

25) คลาส ECC คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ที่มีตัวดำเนินการทางเส้นทาง (/, //) โดยการเปลี่ยนเป็นอีกตัวดำเนินการหนึ่ง รายละเอียดของคลาสดังภาพที่ 3.57



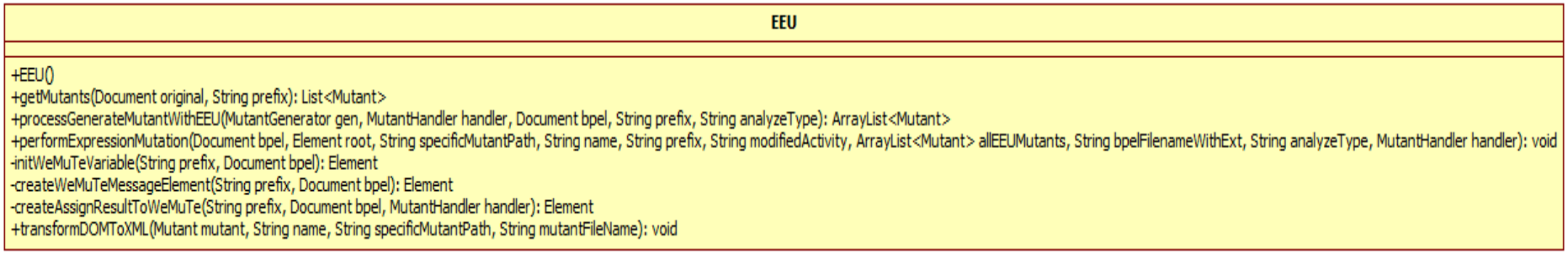
ภาพที่ 3.57 คลาส ECC

26) คลาส ECN คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ที่มีค่าคงที่ โดยลดค่าคงที่ลง 1 หรือ ลดตำแหน่งลง 1 ตำแหน่ง รายละเอียดของคลาสดังภาพที่ 3.58



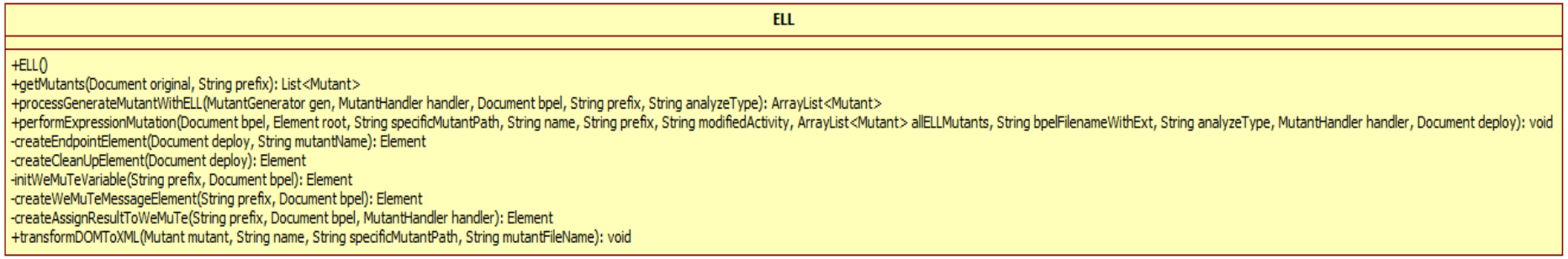
ภาพที่ 3.58 คลาส ECN

27) คลาส EEU คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ที่มีเครื่องหมาย (-) ที่เป็นเอกภาค รายละเอียดของคลาสดังภาพที่ 3.59



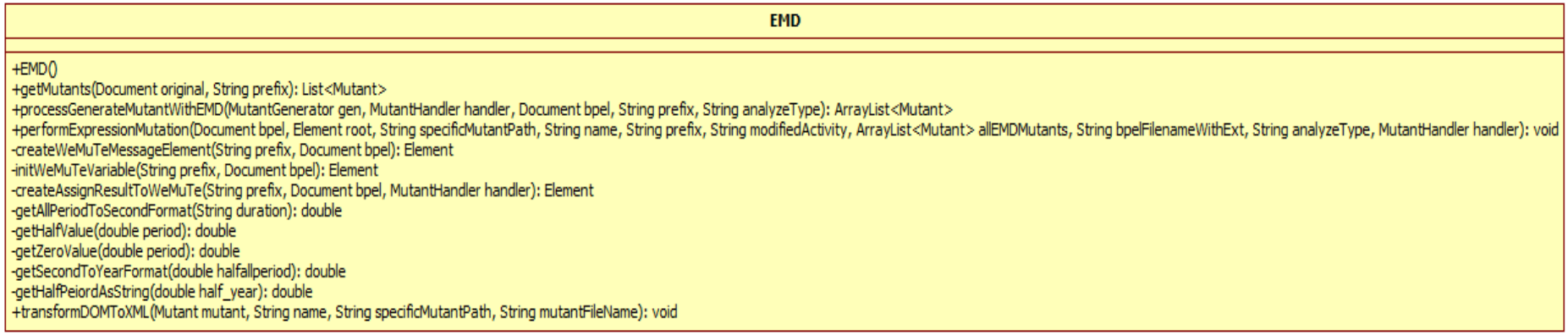
ภาพที่ 3.59 คลาส EEU

28) คลาส ELL คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์โดยเปลี่ยนตัวดำเนินการทางตรรกะ (and, or) รายละเอียดของคลาสดังภาพที่ 3.60



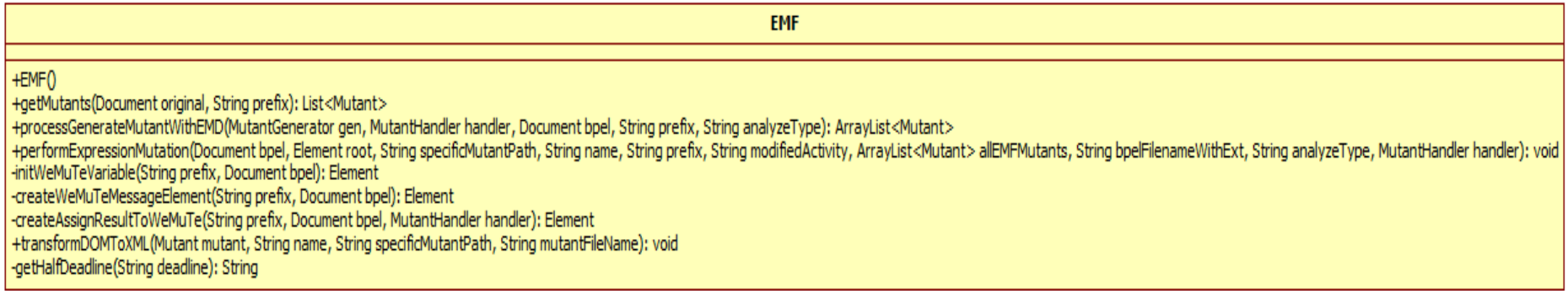
ภาพที่ 3.60 คลาส ELL

29) คลาส EMD คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ช่วงเวลาโดยการกำหนดให้เป็น 0 หรือ เป็นค่าครึ่งหนึ่งของค่าเริ่มต้น รายละเอียดของคลาสดังภาพที่ 3.61



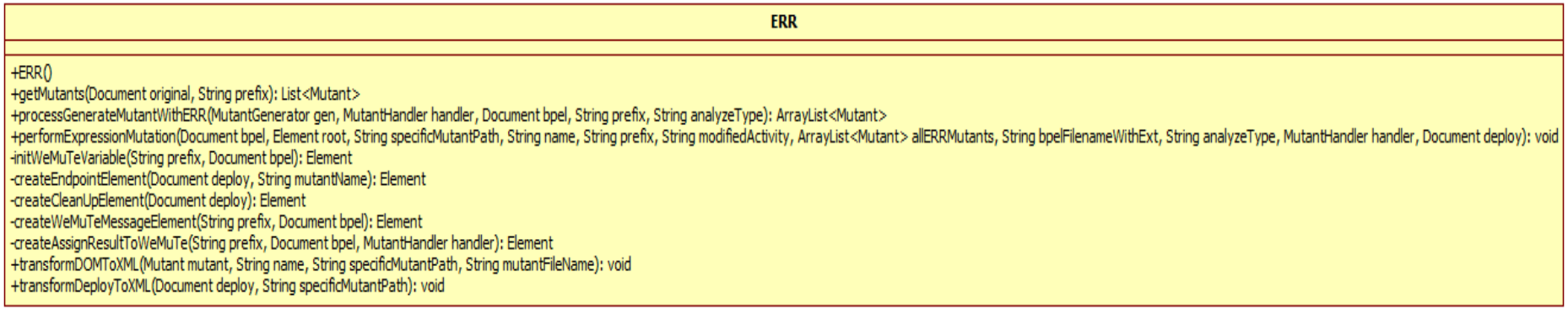
ภาพที่ 3.61 คลาส EMD

30) คลาส EMF คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์เส้นตายโดยการกำหนดให้เป็น 0 หรือ เป็นค่าครึ่งหนึ่งของค่าเริ่มต้น รายละเอียดของคลาสดังภาพที่ 3.62



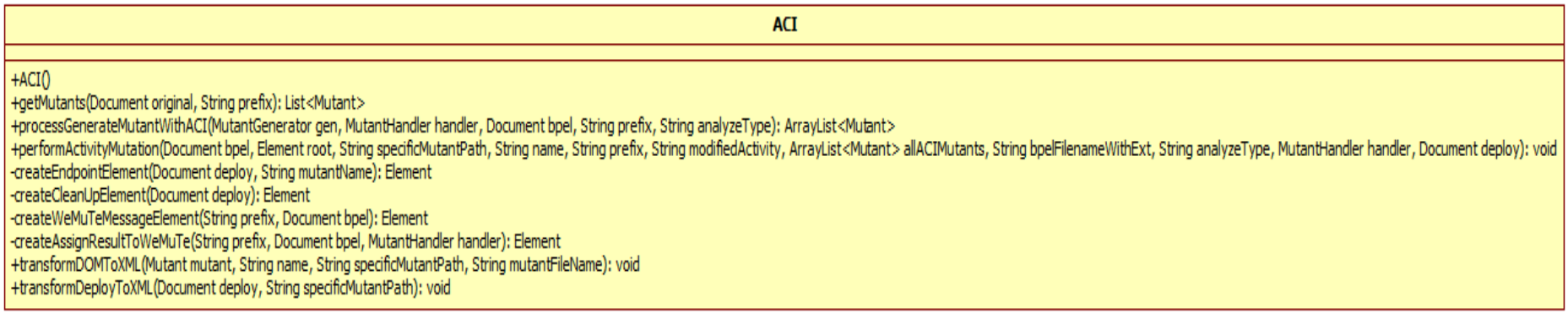
ภาพที่ 3.62 คลาส EMF

31) คลาส ERR คือ คลาสที่ทำหน้าที่ดัดแปลงนิพจน์ที่มีตัวดำเนินการความสัมพันธ์ (=, !=, <, >, <=, >=) โดยการแทนที่ด้วยตัวดำเนินการอีกตัวหนึ่ง รายละเอียดของคลาสดังรูปที่ 3.63



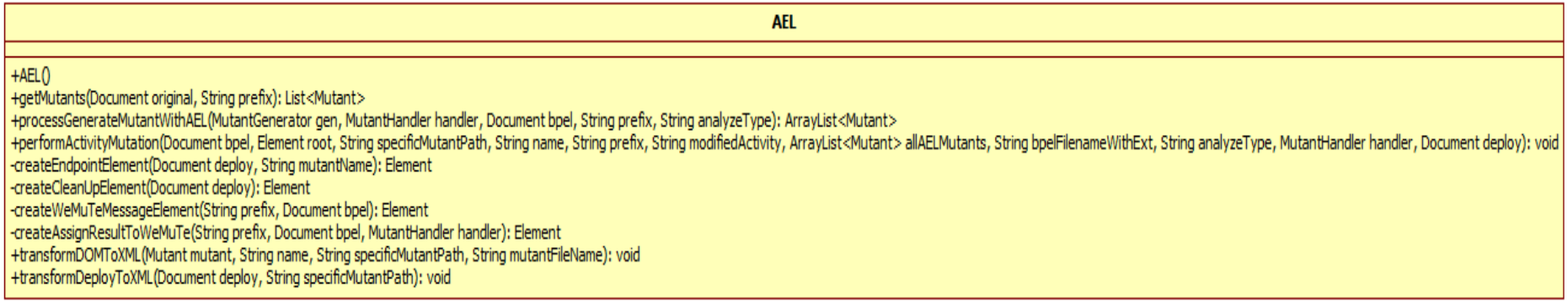
ภาพที่ 3.63 คลาส ERR

32) คลาส ACI คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยเปลี่ยนค่าในแอตทริบิวต์ createInstance จาก yes เป็น no รายละเอียดของคลาสดังภาพที่ 3.64



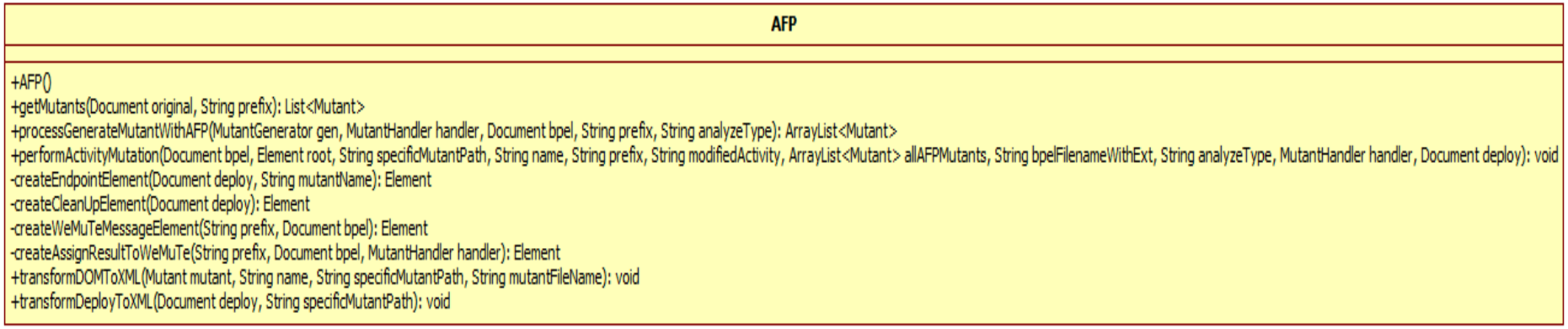
ภาพที่ 3.64 คลาส ACI

33) คลาส AEL คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรมออกจากโปรแกรมบีเพล รายละเอียดของคลาสดังภาพที่ 3.65



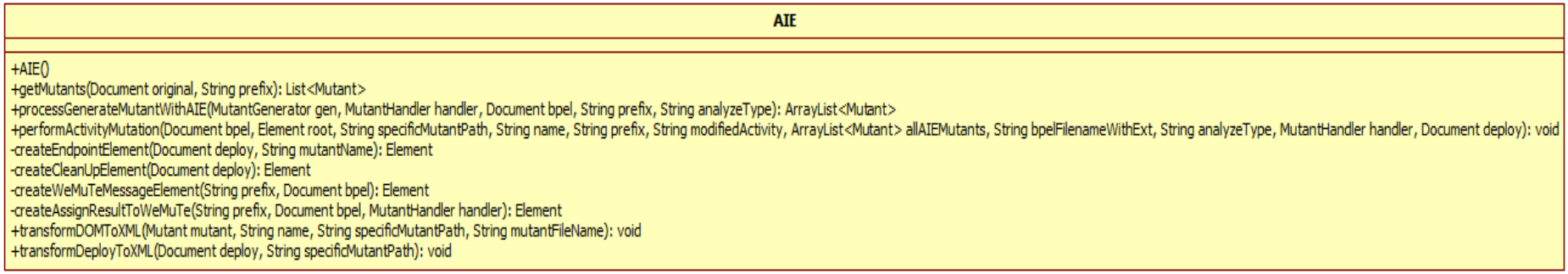
ภาพที่ 3.65 คลาส AEL

34) คลาส AFP คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการเปลี่ยนค่าในแอตทริบิวต์ parallel ของกิจกรรม sequence จาก yes เป็น no รายละเอียดของคลาสดังภาพที่ 3.66



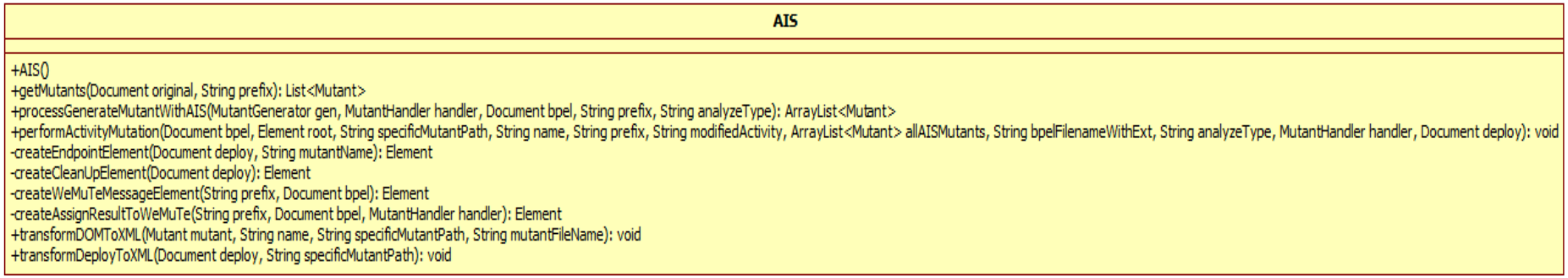
ภาพที่ 3.66 คลาส AFP

35) คลาส AIE คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรม else หรือ elseIf ออกจากกิจกรรม if รายละเอียดของคลาสดังภาพที่ 3.67



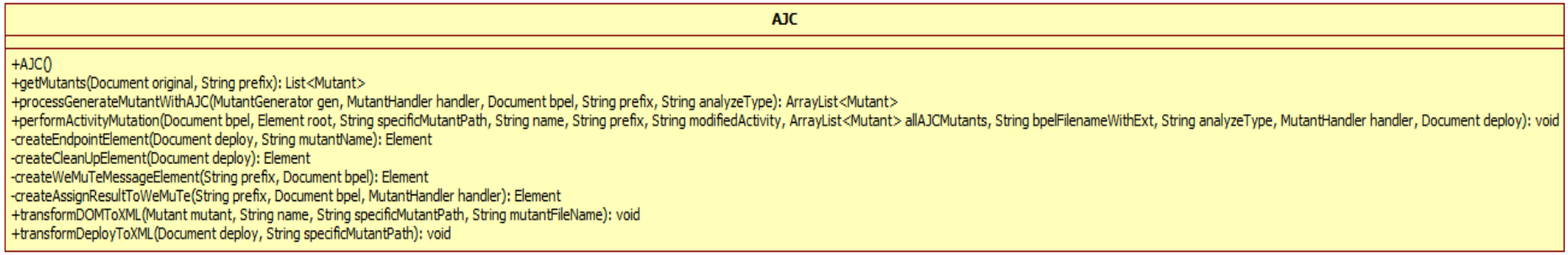
ภาพที่ 3.67 คลาส AIE

36) คลาส AIS คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการเปลี่ยนค่าในแอตทริบิวต์ isolated จาก yes เป็น no รายละเอียดของคลาสดังภาพที่ 3.68



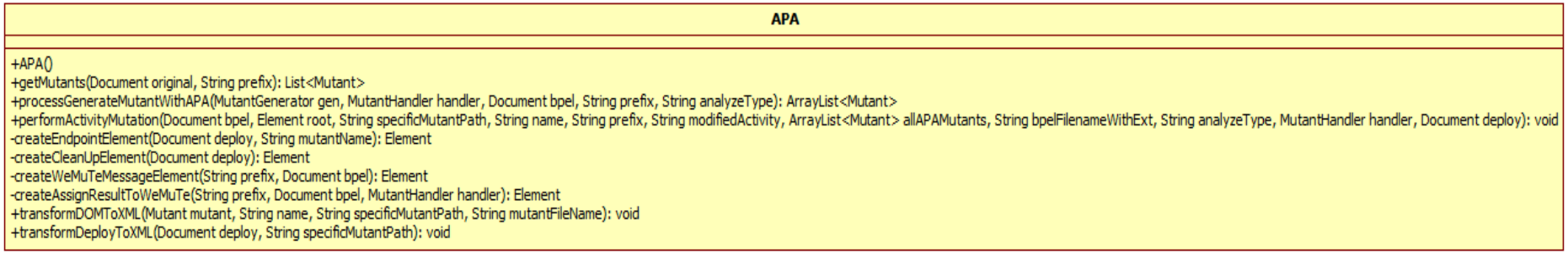
ภาพที่ 3.68 คลาส AIS

37) คลาส AJC คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรม joinCondition ออก จากโปรแกรมบีเพล รายละเอียดของคลาสดังภาพที่ 3.69



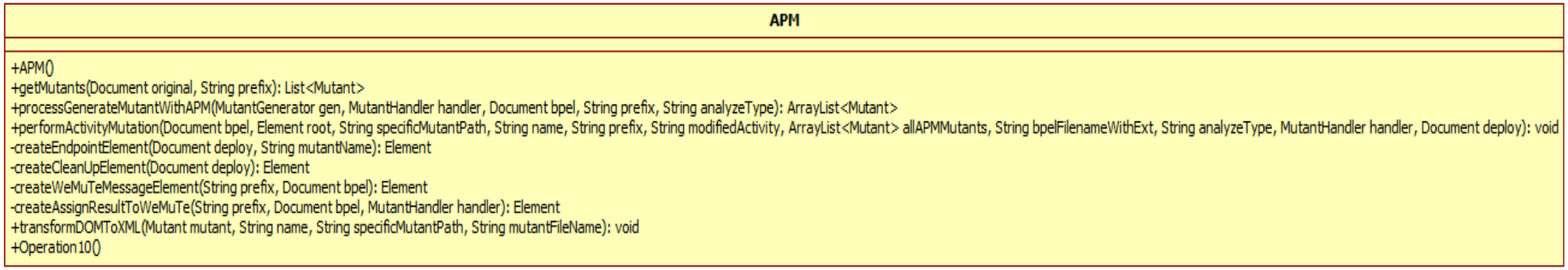
ภาพที่ 3.69 คลาส AJC

38) คลาส APA คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรม onAlarm ออก รายละเอียดของคลาสดังภาพที่ 3.70



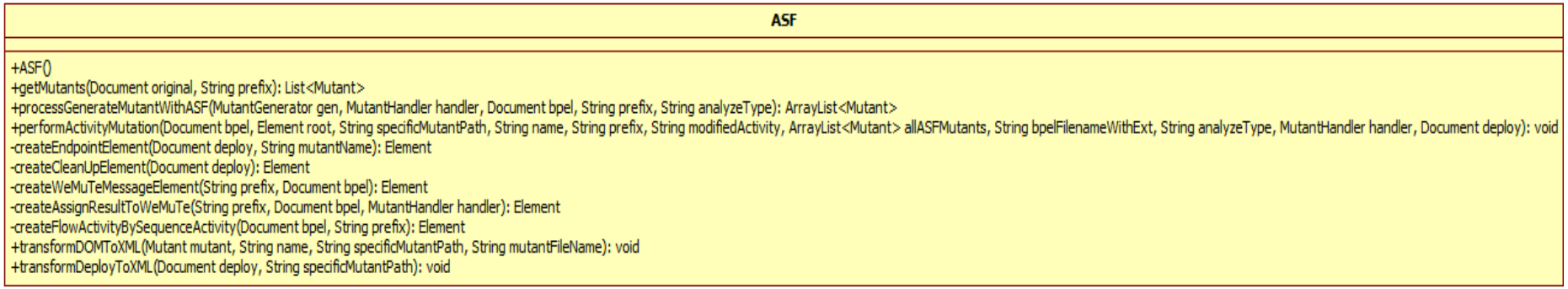
ภาพที่ 3.70 คลาส APA

39) คลาส APM คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรม onMessage ออก รายละเอียดของคลาสดังภาพที่ 3.71



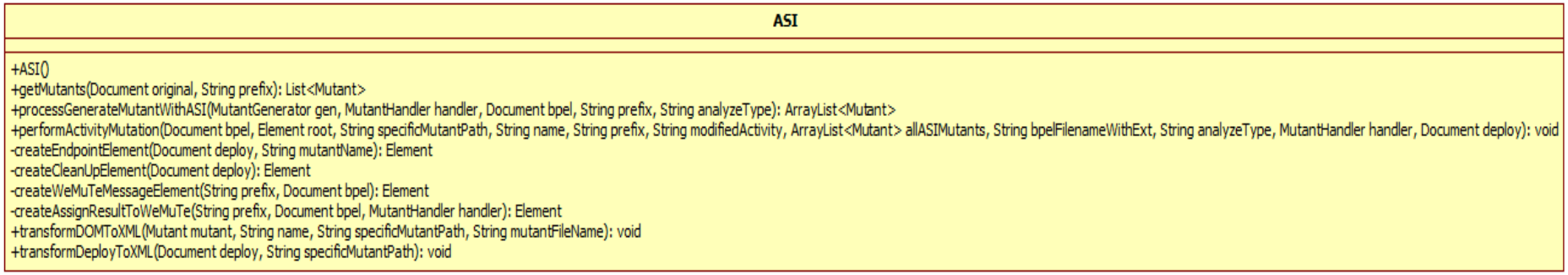
ภาพที่ 3.71 คลาส APM

40) คลาส ASF คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการเปลี่ยนกิจกรรม sequence เป็นกิจกรรม flow รายละเอียดของคลาสดังภาพที่ 3.72



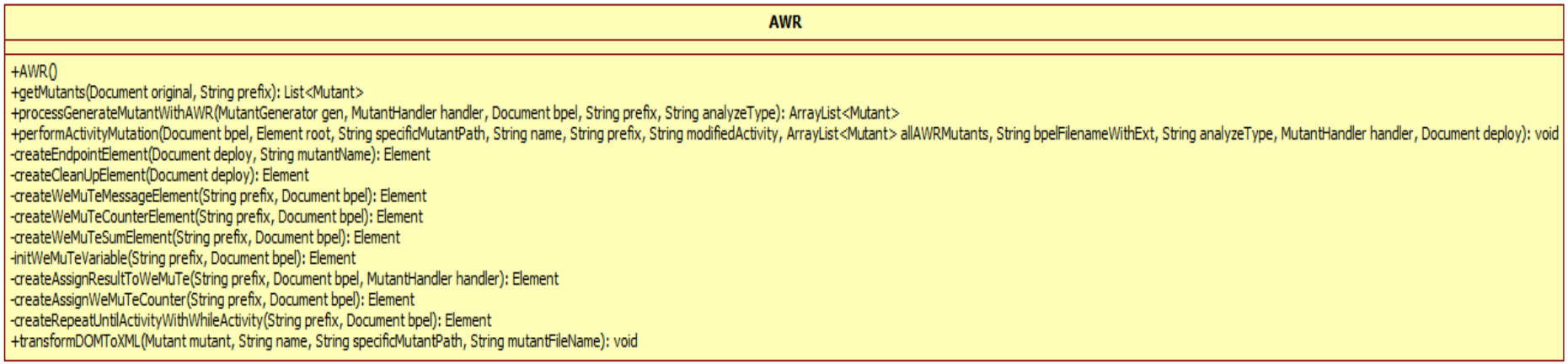
ภาพที่ 3.72 คลาส ASF

41) คลาส ASI คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยที่สลับกิจกรรมที่อยู่ในกิจกรรม sequence รายละเอียดของคลาสดังภาพที่ 3.73



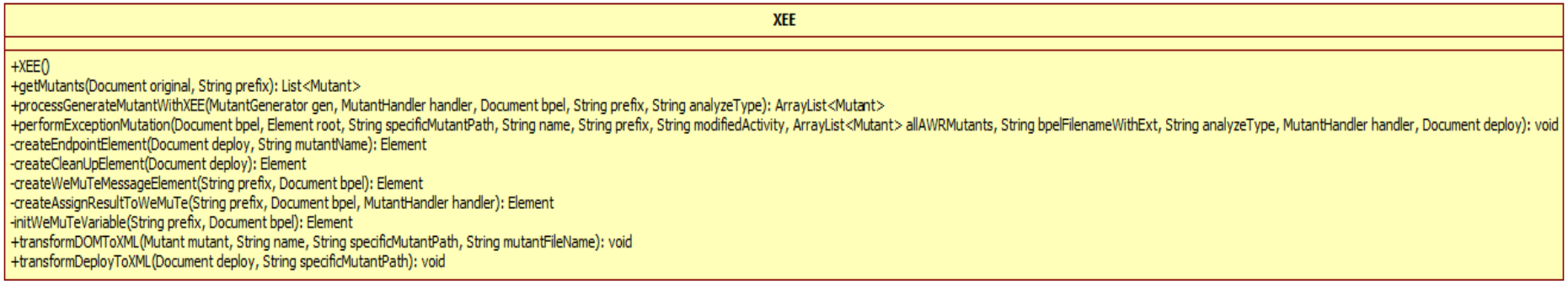
ภาพที่ 3.73 คลาส ASI

42) คลาส AWR คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการเปลี่ยนกิจกรรม while เป็นกิจกรรม repeatUntil รายละเอียดของคลาสดังรูปที่ 3.74



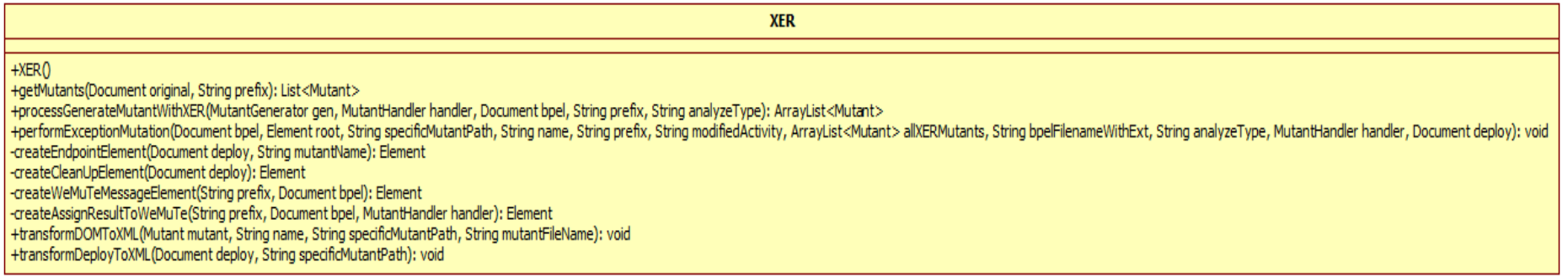
ภาพที่ 3.74 คลาส AWR

43) คลาส XEE คือ คลาสที่ทำหน้าที่ดัดแปลงความผิดปกติโดยการลบกิจกรรม onEvent ออก รายละเอียดของคลาสดังภาพที่ 3.75



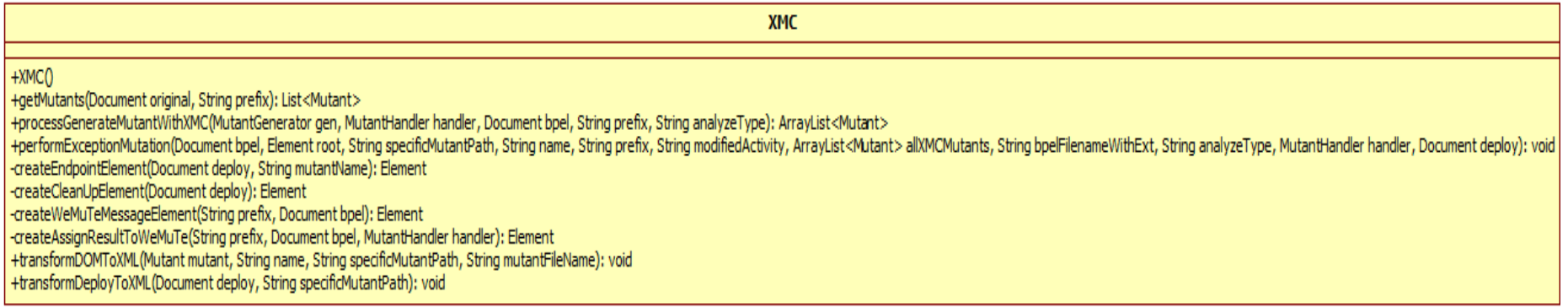
ภาพที่ 3.75 คลาส XEE

44) คลาส XER คือ คลาสที่ทำหน้าที่ดัดแปลงกิจกรรมโดยการลบกิจกรรม rethrow ออก รายละเอียดของคลาสดังภาพที่ 3.76



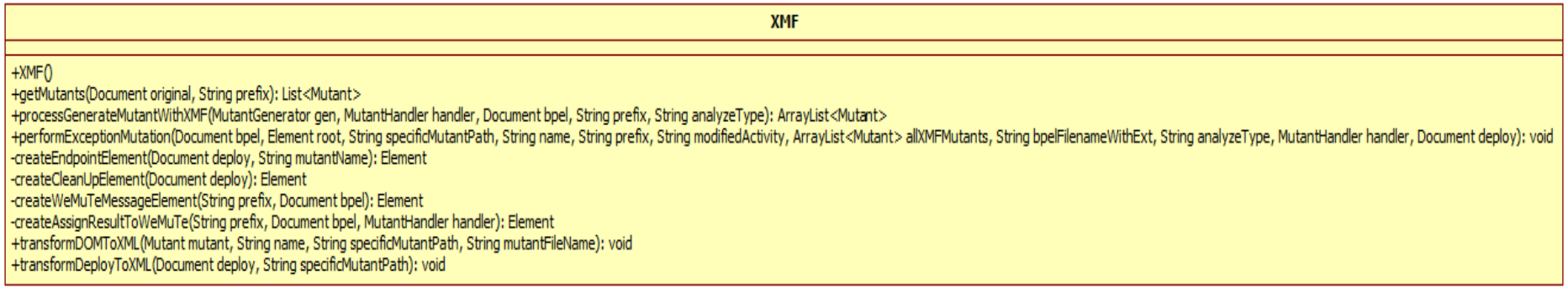
ภาพที่ 3.76 คลาส XER

45) คลาส XMC คือ คลาสที่ทำหน้าที่ดัดแปลงความผิดปกติ โดยการลบลกิจกรรม compensationHandler ออก รายละเอียดดังภาพที่ 3.77



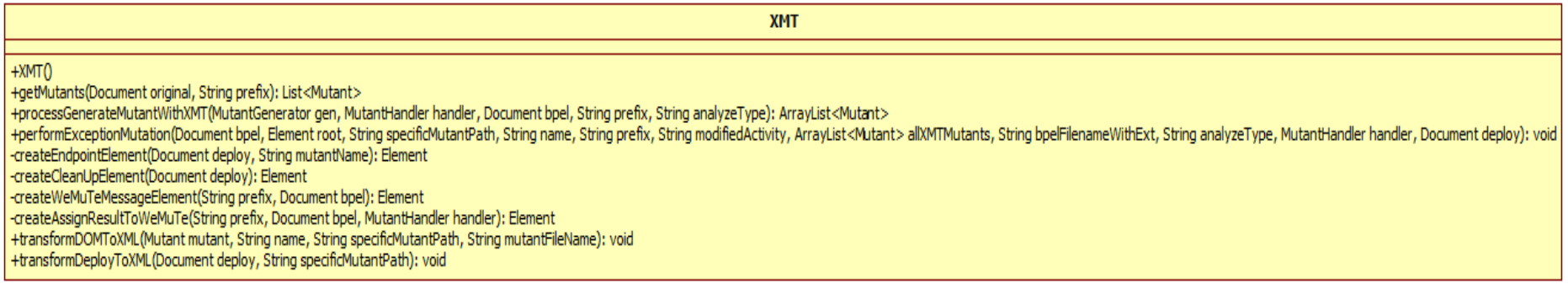
ภาพที่ 3.77 คลาส XMC

46) คลาส XMF คือ คลาสที่ทำหน้าที่ดัดแปลงความผิดปกติ โดยการลบกิจกรรม catch หรือ catchAll ออก รายละเอียดของคลาสดังภาพที่ 3.78



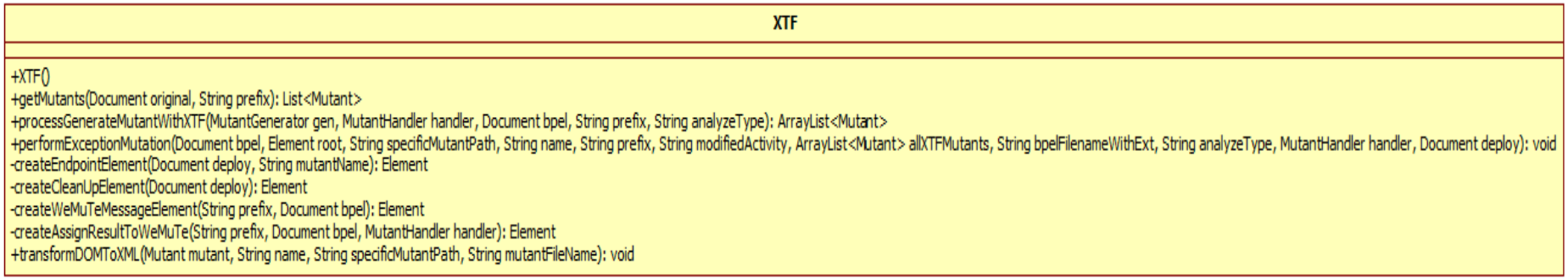
ภาพที่ 3.78 คลาส XMF

47) คลาส XMT คือ คลาสที่ทำหน้าที่ดัดแปลงความผิดปกติ โดยการลบกิจกรรม terminationHandler ออก รายละเอียดของคลาสดังภาพที่ 3.79



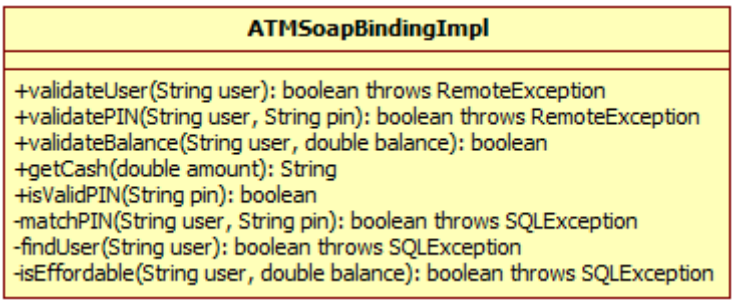
ภาพที่ 3.79 คลาส XMT

48) คลาส XTF คือ คลาสที่ทำหน้าที่ดัดแปลงความผิดปกติ โดยการเปลี่ยนค่าในแอตทริบิวต์ของ faultName ที่อยู่ในกิจกรรม throw รายละเอียดของคลาสดังภาพที่ 3.80



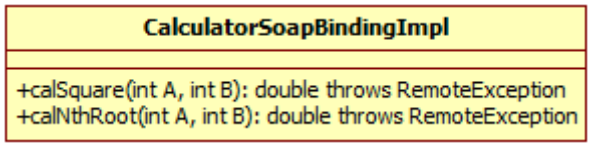
ภาพที่ 3.80 คลาส XTF

49) คลาส ATMSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบความถูกต้องของชื่อผู้ใช้ รหัสเอทีเอ็ม และยอดเงินซึ่งถูกเรียกจากมากระบวนการบีเพล ATMProcess รายละเอียดของคลาสดังภาพที่ 3.81



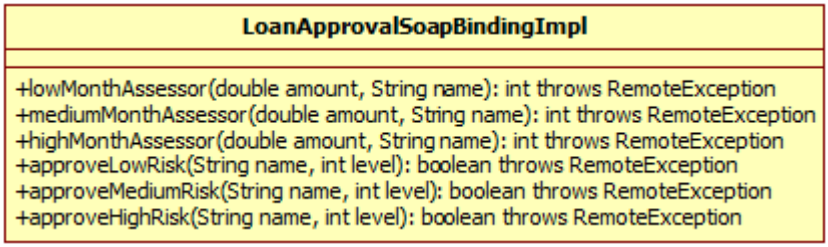
ภาพที่ 3.81 คลาส ATMSoapBingImpl

50) คลาส CalculatorSoapBindingImpl คือ คลาสที่ทำหน้าที่คำนวณการหาค่ายกกำลังและหารากที่ n ซึ่งถูกเรียกมาจากกระบวนการบีเพล SimpleCalculator รายละเอียดของคลาสดังภาพที่ 3.82



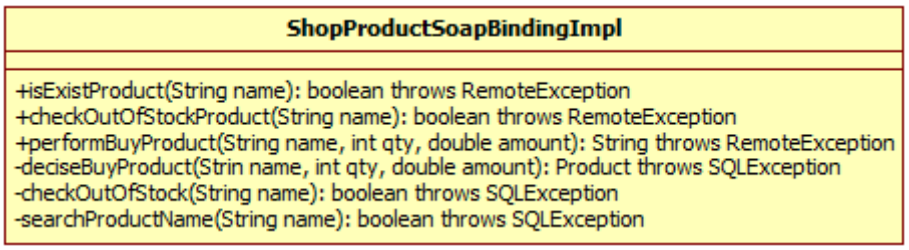
ภาพที่ 3.82 คลาส CalculatorSoapBindingImpl

51) คลาส LoanApprovalSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบระดับความเสี่ยงและการอนุมัติการกู้ยืมเงิน ซึ่งถูกเรียกมาจากกระบวนการบีเพล LoanApprovalProcess รายละเอียดของคลาสดังภาพที่ 3.83



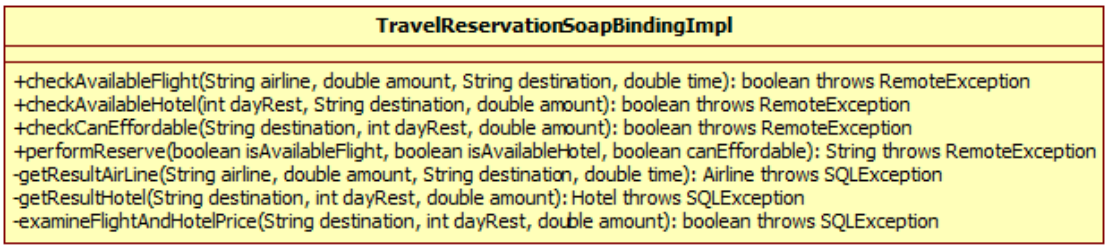
ภาพที่ 3.83 คลาส LoanApprovalSoapBindImpl

52) คลาส ShopProductSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบสินค้ามีขายในร้านขายสินค้าหรือไม่ สินค้ามีอยู่ในคลังสินค้าหรือไม่ และการซื้อสินค้าซึ่งถูกเรียกมาจากกระบวนการบีเพล ShopProductProcess รายละเอียดของคลาสดังภาพที่ 3.84



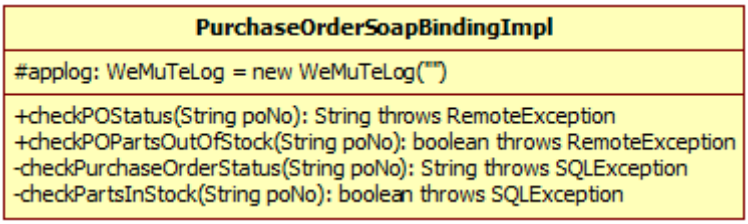
ภาพที่ 3.84 คลาส ShopProductSoapBindingImpl

53) คลาส TravelReservationSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบสายการบินที่ต้องการว่าว่างหรือไม่ ตรวจสอบโรงแรมที่ต้องการว่าว่างหรือไม่ ตรวจสอบจำนวนเงินที่ใช้ในการท่องเที่ยวเพียงพอหรือไม่ และการทำการจองตั๋ว ซึ่งถูกเรียกมาจากกระบวนการบีเพล TravelReservationProcess รายละเอียดของคลาสดังภาพที่ 3.85



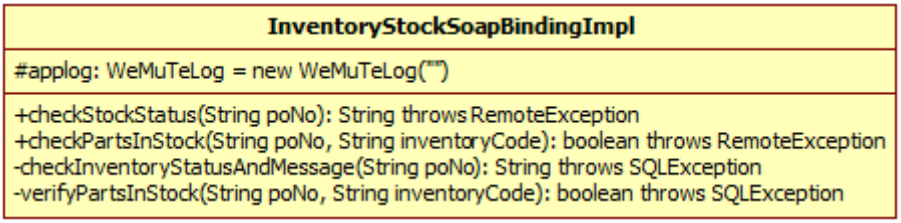
ภาพที่ 3.85 คลาส TravelReservationSoapBindingImpl

54) คลาส PurchaseOrderSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบสถานะของใบสั่งซื้อสินค้าและอะไหล่ที่สั่งซื้อในใบสั่งซื้อสินค้านั้นมีเพียงพอหรือไม่ ซึ่งถูกเรียกมาจากกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.86



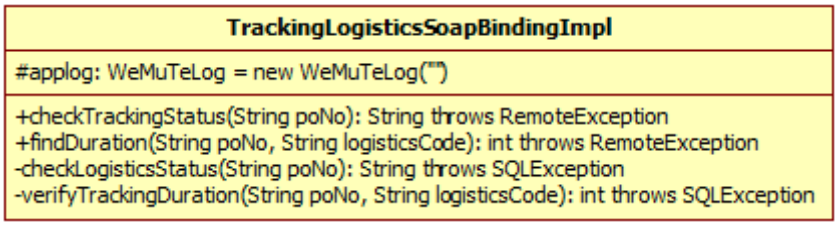
ภาพที่ 3.86 คลาส PurchaseOrderSoapBindingImpl

55) คลาส InventoryStockSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบสถานะของคลังสินค้า และตรวจสอบอะไหล่นั้นที่ต้องการมีอยู่ในคลังสินค้าหรือไม่ ซึ่งถูกเรียกมาจากกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.87



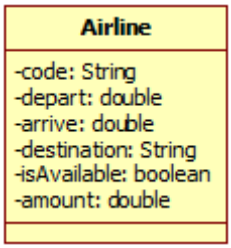
ภาพที่ 3.87 คลาส InventoryStockSoapBindingImpl

56) คลาส TrackingLogisticsSoapBindingImpl คือ คลาสที่ทำหน้าที่ตรวจสอบสถานะของการขนส่งสินค้า และตรวจสอบจำนวนวันที่ใช้ในการขนส่งสินค้า ซึ่งถูกเรียกมาจากกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.88



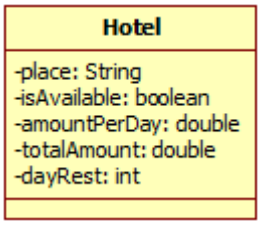
ภาพที่ 3.88 คลาส TrackingLogisticsSoapBindingImpl

57) คลาส Airline คือ คลาสที่ทำหน้าที่เก็บข้อมูลของสายการบินซึ่งใช้ในการกระบวนการบีเพล TravelReservationProcess รายละเอียดของคลาสดังภาพที่ 3.89



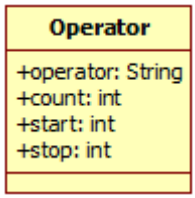
ภาพที่ 3.89 คลาส Airline

58) คลาส Hotel คือ คลาสที่ทำหน้าที่เก็บข้อมูลของโรงแรมซึ่งถูกใช้ในกระบวนการบีเพล TravelReservationProcess รายละเอียดของคลาสดังภาพที่ 3.90



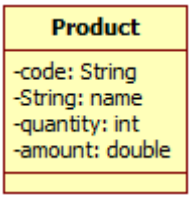
ภาพที่ 3.90 คลาส Hotel

59) คลาส Operator คือ คลาสที่ทำหน้าที่เก็บตัวดำเนินการ ตำแหน่งของตัวดำเนินการ และจำนวนของตัวดำเนินการที่อยู่ในนิพจน์และประพจน์ รายละเอียดของคลาสดังภาพที่ 3.91



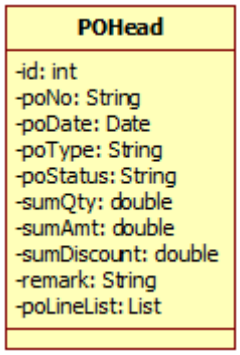
ภาพที่ 3.91 คลาส Operator

60) คลาส Product คือ คลาสที่ทำหน้าที่ข้อมูลของสินค้าซึ่งถูกใช้ในกระบวนการบีเพล ShopProductProcess รายละเอียดของคลาสดังภาพที่ 3.92



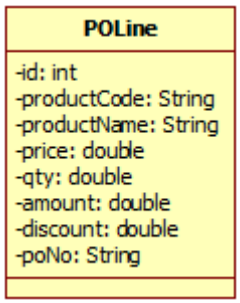
ภาพที่ 3.92 คลาส Product

61) คลาส POHead คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนหัวของใบสั่งซื้อสินค้า ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.93



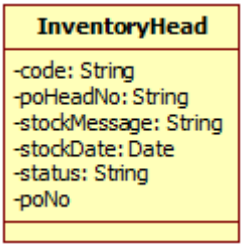
ภาพที่ 3.93 คลาส POHead

62) คลาส POLine คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนท้ายของใบสั่งซื้อสินค้า ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.94



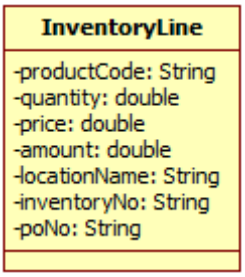
ภาพที่ 3.94 คลาส POLine

63) คลาส InventoryHead คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนหัวของคลังสินค้าซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.95



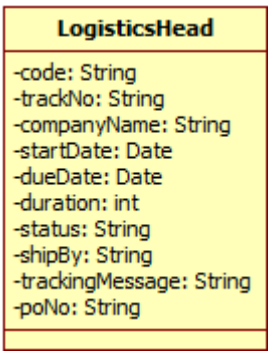
ภาพที่ 3.95 คลาส InventoryHead

64) คลาส InventoryLine คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนท้ายของคลังสินค้า ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.96



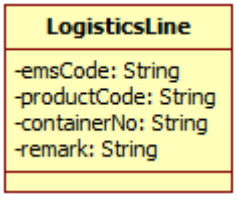
ภาพที่ 3.96 คลาส InventoryLine

65) คลาส LogisticsHead คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนหัวของการขนส่งสินค้า ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.97



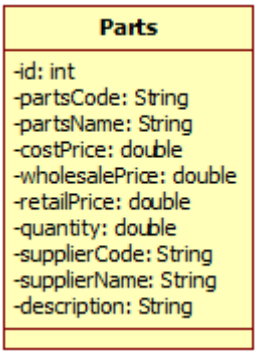
ภาพที่ 3.97 คลาส LogisticsHead

66) คลาส LogisticsLine คือ คลาสที่ทำหน้าที่เก็บข้อมูลของส่วนท้ายของการขนส่งสินค้า ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.98



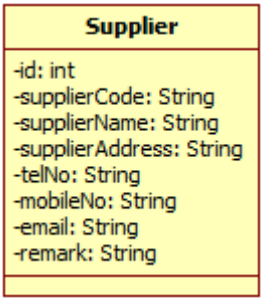
ภาพที่ 3.98 คลาส LogisticsLine

67) คลาส Parts คือ คลาสที่ทำหน้าที่เก็บข้อมูลของอะไหล่ ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.99



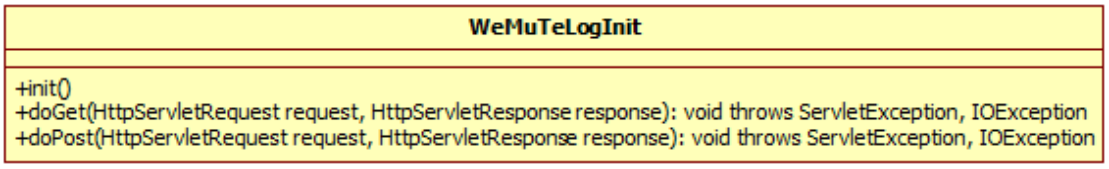
ภาพที่ 3.99 คลาส Parts

68) คลาส Supplier คือ คลาสที่ทำหน้าที่เก็บข้อมูลของผู้ขายอะไหล่ ซึ่งถูกใช้ในกระบวนการบีเพล PurchaseOrderProcess รายละเอียดของคลาสดังภาพที่ 3.100



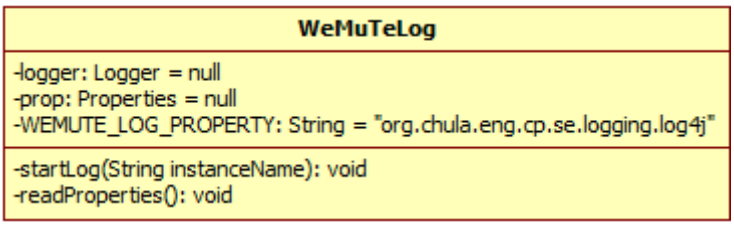
ภาพที่ 3.100 คลาส Supplier

69) คลาส WeMuTeLogInit คือ คลาสที่ทำหน้าที่กำหนดค่าเริ่มต้นให้กับคลาส WeMuTeLog รายละเอียดของคลาสดังภาพที่ 3.101



ภาพที่ 3.101 คลาส WeMuTeLogInit

70) คลาส WeMuTeLog คือ คลาสที่ทำหน้าที่กำหนดค่าเริ่มต้น ดัดแปลงความผิดปกติ โดยการเปลี่ยนค่าในแอตทริบิวต์ของ faultName ที่อยู่ในกิจกรรม throw รายละเอียดของคลาสดังภาพที่ 3.102

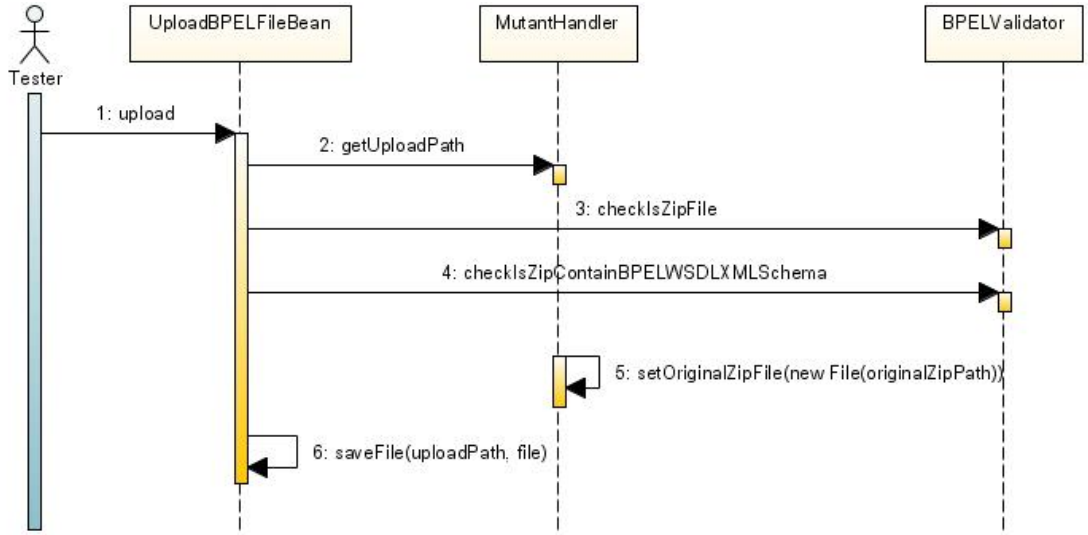


ภาพที่ 3.102 คลาส WeMuTeLog

3.3.4 แผนภาพลำดับและแผนภาพกิจกรรม

แผนภาพลำดับกิจกรรมจะอธิบายถึงความสัมพันธ์ระหว่างวัตถุต่างๆในกระบวนการนั้นๆ เป็นลำดับขั้น โดยแผนภาพลำดับของเครื่องมือมีดังต่อไปนี้

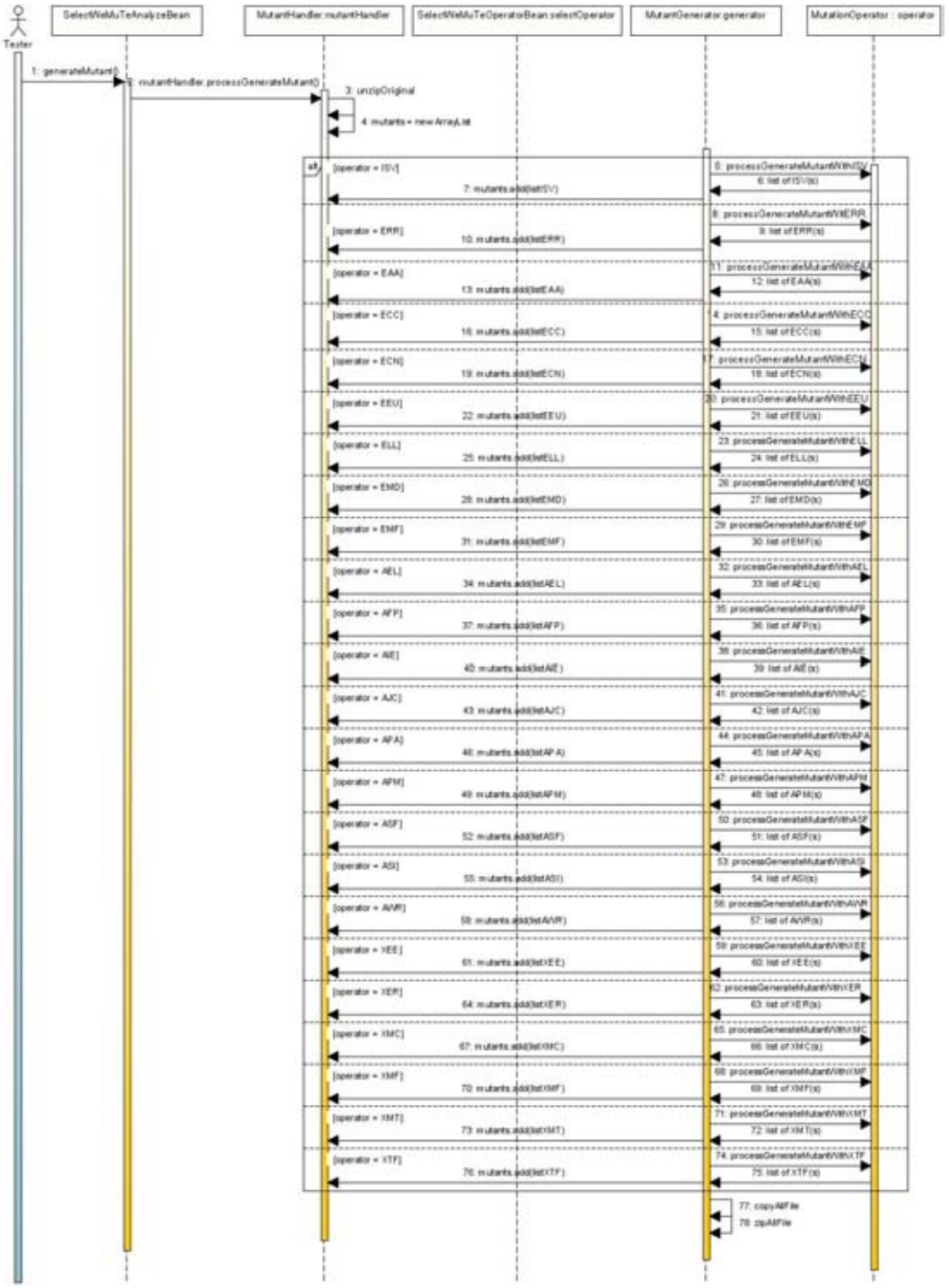
1) แผนภาพลำดับ Uploaded BPEL File เป็นการอัพโหลดโปรแกรมบีเพลในรูปของเอกสารซิปโดยนักทดสอบ จากนั้นอ็อบเจ็กต์ MutantHandler จะเก็บค่าเส้นทางของเอกสารไว้และอ็อบเจ็กต์ BPELValidator ก็จะตรวจสอบเอกสารว่าบรรจุเอกสารบีเพลและวิซเดิลไว้หรือไม่ จากนั้น MutantHandler ก็จะนำเอกสารดังกล่าวไปเก็บไว้ที่เครื่องบริการเว็บโดยเรียกเมท็อด saveFile ดังในภาพที่ 3.103



ภาพที่ 3.103 แผนภาพลำดับ Uploaded BPEL File

2) แผนภาพลำดับ Generate Mutants เป็นการสร้างมิวแตนท์จากเอกสารบีเพล เริ่มโดยการที่นักทดสอบเลือกชนิดวิธีการทดสอบของวีคมิวเทชัน (EX-WEAK/1 ST-WEAK/1 BB-WEAK/1 และ BB-WEAK/N) จากนั้นระบบจะเลือกตัวดำเนินการที่เกี่ยวข้องกับแนววิธีทดสอบที่เลือกโดยอัตโนมัติ เมื่อนักทดสอบกดปุ่มเพื่อสร้างมิวแตนท์อ็อบเจ็กต์ MutantHandler จะตรวจสอบตัวดำเนินการแต่ละตัว ถ้าตัวดำเนินการนั้นถูกเลือกก็จะสร้างมิวแตนท์ตามหลักการสร้างของตัวดำเนินการแต่ละตัว ดังภาพที่ 3.104

ในส่วนของการสร้างมิวแตนท์แต่ละตัวดำเนินการมิวเทชันจะถูกอธิบายด้วยแผนภาพกิจกรรม ดังภาพที่ 3.105 – 3.144



ภาพที่ 3.104 แผนภาพลำดับ Generate Mutants

เนื่องจากเอกสารบีเพลเป็นเอกสารเอ็กซ์เอ็มแอล ซึ่งในภาษาจาวามีฟังก์ชันในการเข้าถึงเนื้อหาของเอกสารบีเพลได้ โดยการแปลงเอกสารบีเพลไปเป็นดอมอ็อปเจ็กต์ ทำให้สามารถเพิ่ม ลบ และปรับเปลี่ยนเนื้อหาในกิจกรรมหรือค่าของแอตทริบิวต์ของกิจกรรมในกระบวนการบีเพลได้

2.1) แผนภาพกิจกรรมของ performGenerateMutantWithISV ในภาพที่ 3.105 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม if, elseif, while, repeatUntil, target และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ isvList

3. ตรวจสอบว่า isvList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

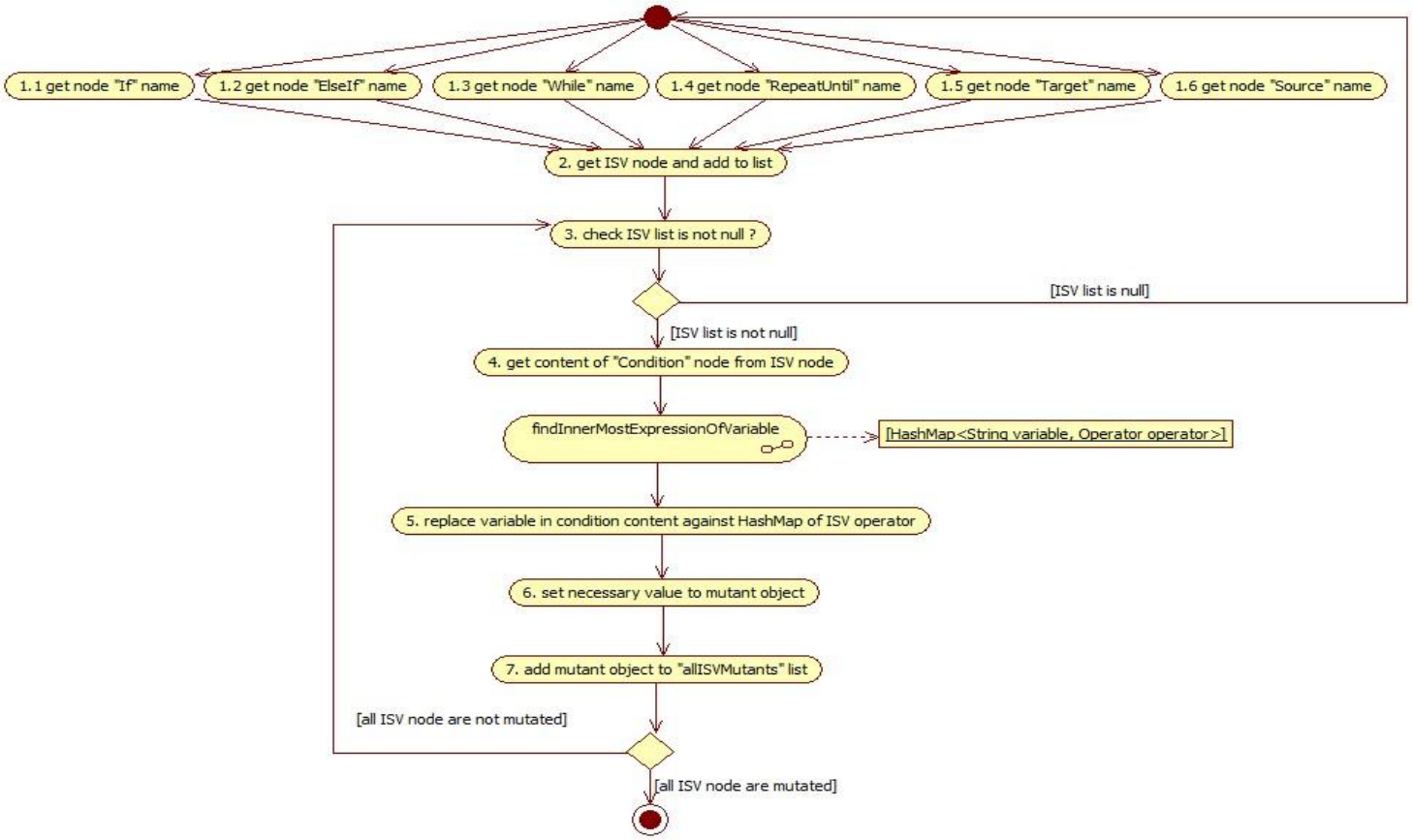
4. ถ้า isvList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ซึ่งเป็นออกมา

หมายเหตุ: หลังจากตรวจสอบว่าในรายการ isvList มีข้อมูลแล้ว โปรแกรมจะดำเนินการขั้นตอน findInnerMostExpressionOfVariable ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรมในภาพที่ 3.106

5. โปรแกรมจะหาตัวแปรภายในประพจน์ของ condition จากนั้นโปรแกรมเก็บตัวแปรไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $input.A + $input.B + $input.C โปรแกรมจะเก็บตัวแปรและตำแหน่งของตัวแปรไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการสลับที่ระหว่างตัวแปร

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allISVMutants



ภาพที่ 3.105 แผนภาพกิจกรรม performGenerateMutantWithISV

แผนภาพกิจกรรม findInnerMostExpressionOfVariable มีขั้นตอนดังต่อไปนี้

1. หาตัวแปรเสมือนเป็นตัวดำเนินการจากประพจน์ที่มาจากโหนด condition

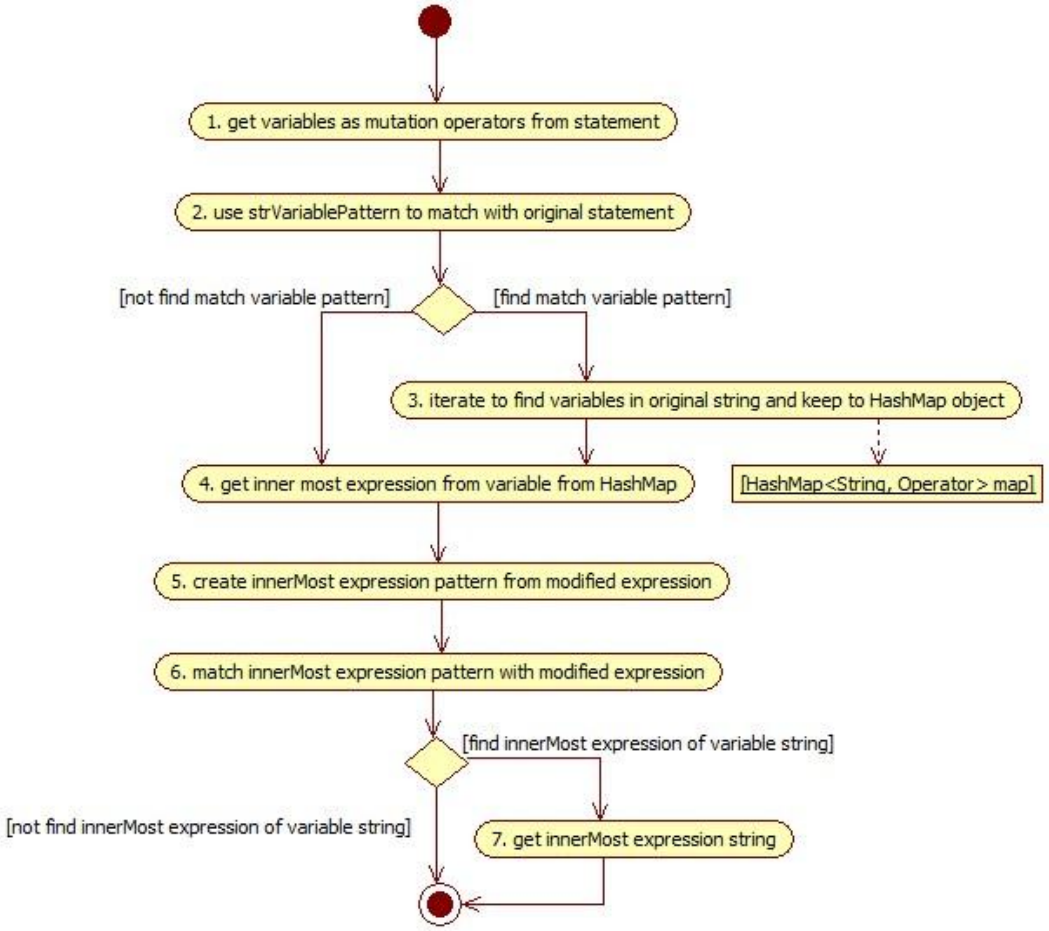
2. โดยใช้รูปแบบ \\$\\S+\\.[0-9a-zA-Z]+ เพื่อใช้ในการหาตัวแปรในประพจน์

3. วนซ้ำเพื่อหาตัวแปรในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ในอ็อปเจ็กต์ HashMap

4. วนซ้ำแต่ละตัวแปรในอ็อปเจ็กต์ HashMap เพื่อหานิพจน์ข้างในสุดของตัวแปรที่เก็บไว้ โดย

5. ใช้รูปแบบเพื่อจับรูปแบบในนิพจน์หรือประพจน์ ((\\$([a-zA-Z0-9]+\\.[a-zA-Z0-9]+|[a-zA-Z0-9]+)|\\d+)|(\\(\\D+\\)))\\s+(\\(|\\)|=|!=|<|>|<=|>=|\\+|\\|*|div|mod|and|or)\\s+ ตามด้วยตัวแปรที่อยู่ในอ็อปเจ็กต์ HashMap และตามด้วยรูปแบบ \\s+(\\(|\\)|=|!=|<|>|<=|>=|\\+|\\-|*|div|mod|and|or)\\s+((\\$([a-zA-Z0-9]+\\.[a-zA-Z0-9]+|[a-zA-Z0-9]+)|\\d+)|(\\(\\D+\\)))

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของตัวแปรแต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.106 แผนภาพกิจกรรม findInnerMostExpressionOfVariable

2.2) แผนภาพกิจกรรมของ performGenerateMutantWithEAA ในภาพที่ 3.107 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม if, elseif, while, repeatUntil, target และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ eaaList

3. ตรวจสอบว่า eaaList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

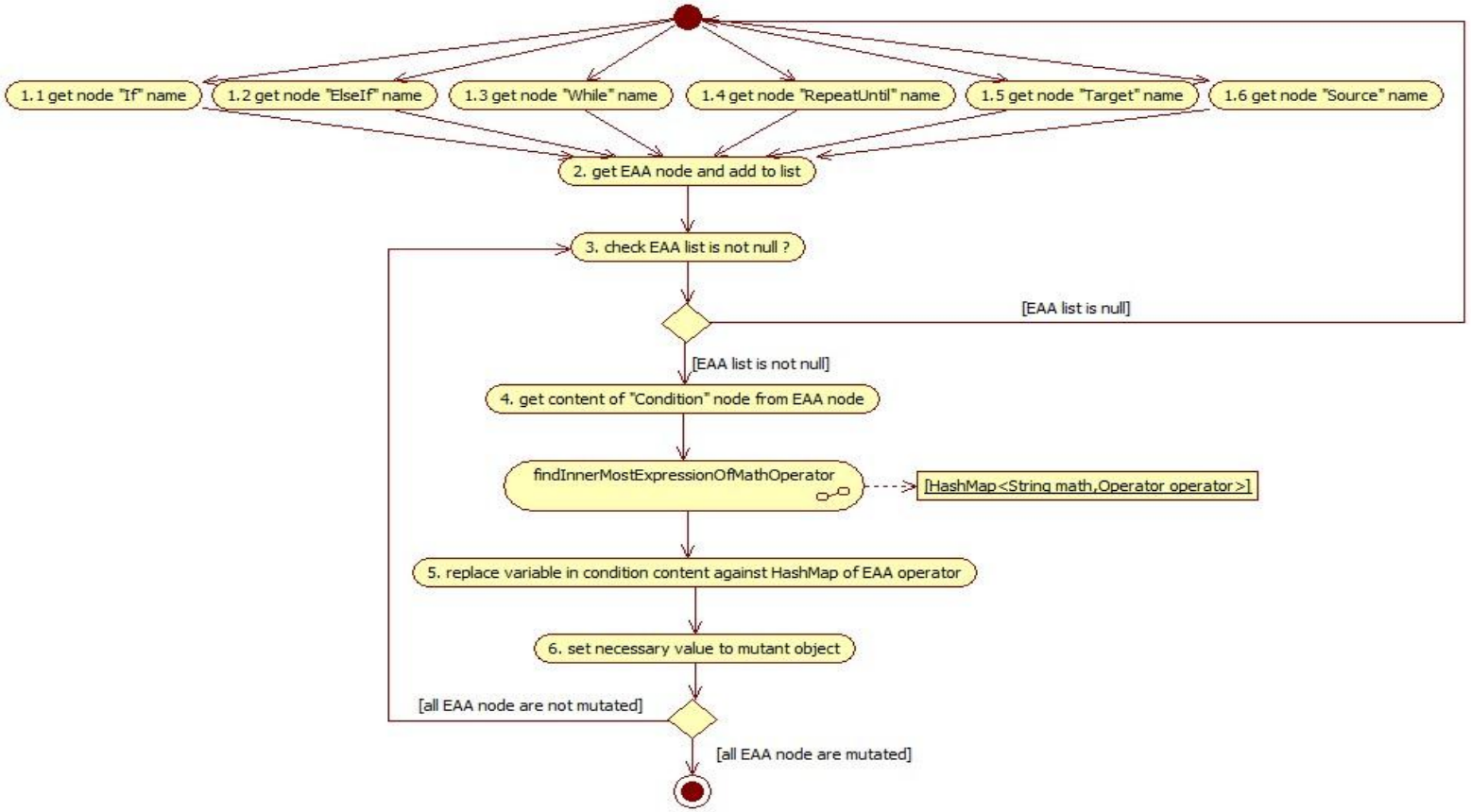
4. ถ้า eaaList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ออกมา

หมายเหตุ: หลังจากตรวจสอบว่าในรายการ eaaList มีข้อมูลแล้ว โปรแกรมจะดำเนินการ findInnerMostExpressionOfMathOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.108

5. โปรแกรมจะหาเครื่องหมายทางคณิตศาสตร์ (+, -, *, div, mod) ภายในนิพจน์หรือประพจน์ของ condition จากนั้นโปรแกรมเก็บเครื่องหมายไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $input.A + $input.B + $input.C โปรแกรมจะเก็บเครื่องหมาย + และตำแหน่งของเครื่องหมาย + ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการสลับที่เครื่องหมายทางคณิตศาสตร์ไปเป็นตัวอื่นๆ ที่ไม่ใช่เครื่องหมายในประพจน์

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEAAMutants



ภาพที่ 3.107 แผนภาพกิจกรรม performGenerateMutantWithEAA

แผนภาพกิจกรรม findInnerMostExpressionOfMathOperator ดังภาพที่ 3.108 มีขั้นตอนดังต่อไปนี้

1. ค้นหาเครื่องหมายทางคณิตศาสตร์เสมือนเป็นตัวดำเนินการจากประพจน์ที่มาจากโหนด condition

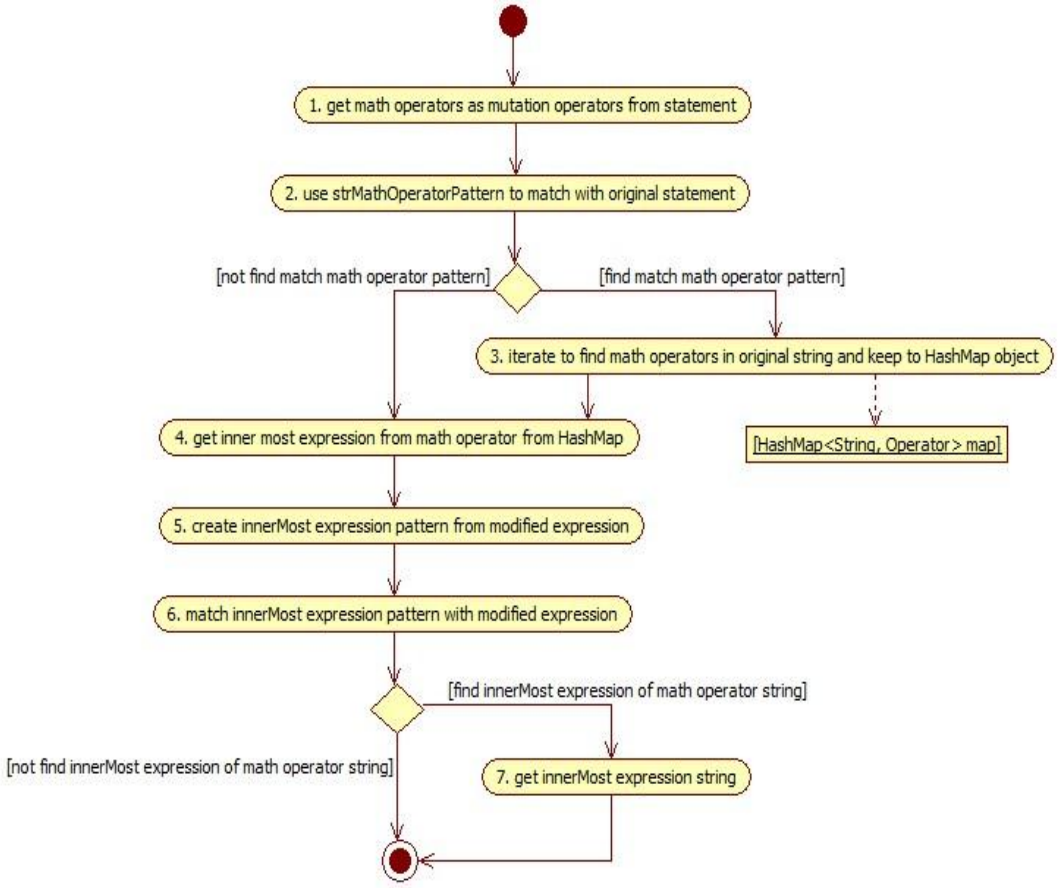
2. โดยใช้รูปแบบ \\+|\\-|*| div | mod เพื่อใช้ในการหาเครื่องหมายทางคณิตศาสตร์ในประพจน์ของ condition

3. วนซ้ำเพื่อหาตัวแปรในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละตัวแปรในอ็อปเจ็กต์ HashMap เพื่อหานิพจน์ข้างในสุดของตัวแปรที่เก็บไว้ โดย

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยเครื่องหมายทางคณิตศาสตร์ที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของตัวแปรแต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.108 แผนภาพกิจกรรม findInnerMostExpressionOfMathOperator

2.3) แผนภาพกิจกรรมของ performGenerateMutantWithECC ในภาพที่ 3.109 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม from, to, if, elseif, while, repeatUntil, targets และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ eccList

3. ตรวจสอบว่า eccList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

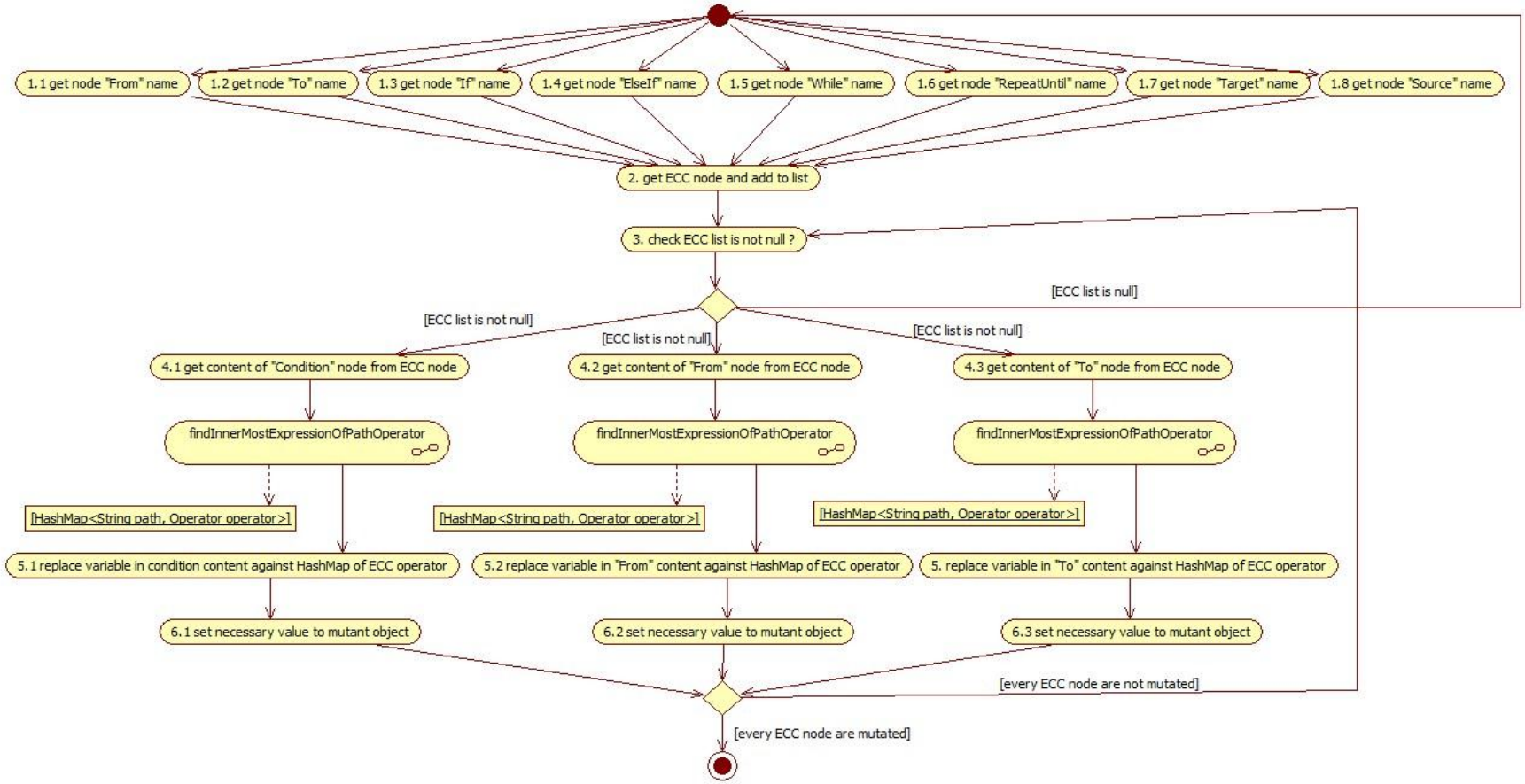
4. ถ้า eccList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ออกมา

หมายเหตุ: หลังจากตรวจสอบว่าในรายการ eccList มีข้อมูลแล้ว โปรแกรมจะดำเนินการ findInnerMostExpressionOfPathOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.110

5. โปรแกรมจะหาตัวดำเนินการเกี่ยวกับเส้นทาง (/ และ //) ภายในนิพจน์หรือประพจน์ของ condition จากนั้นโปรแกรมเก็บเครื่องหมายไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $input.request/ns0:n โปรแกรมจะเก็บเครื่องหมาย / และตำแหน่งของเครื่องหมาย / ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการสลับที่เครื่องหมายที่เกี่ยวกับเส้นทางไปเป็นตัวอื่นๆ ที่ไม่ใช่เครื่องหมายในประพจน์

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allECCMutants



ภาพที่ 3.109 แผนภาพกิจกรรม performGenerateMutantWithECC

แผนภาพกิจกรรม findInnerMostExpressionOfPathOperator มีขั้นตอนดังต่อไปนี้

1. ค้นหาเครื่องหมายที่เกี่ยวกับเส้นทางเสมือนเป็นตัวดำเนินการจากประพจน์ที่ได้จากโหนด from, to, และ condition

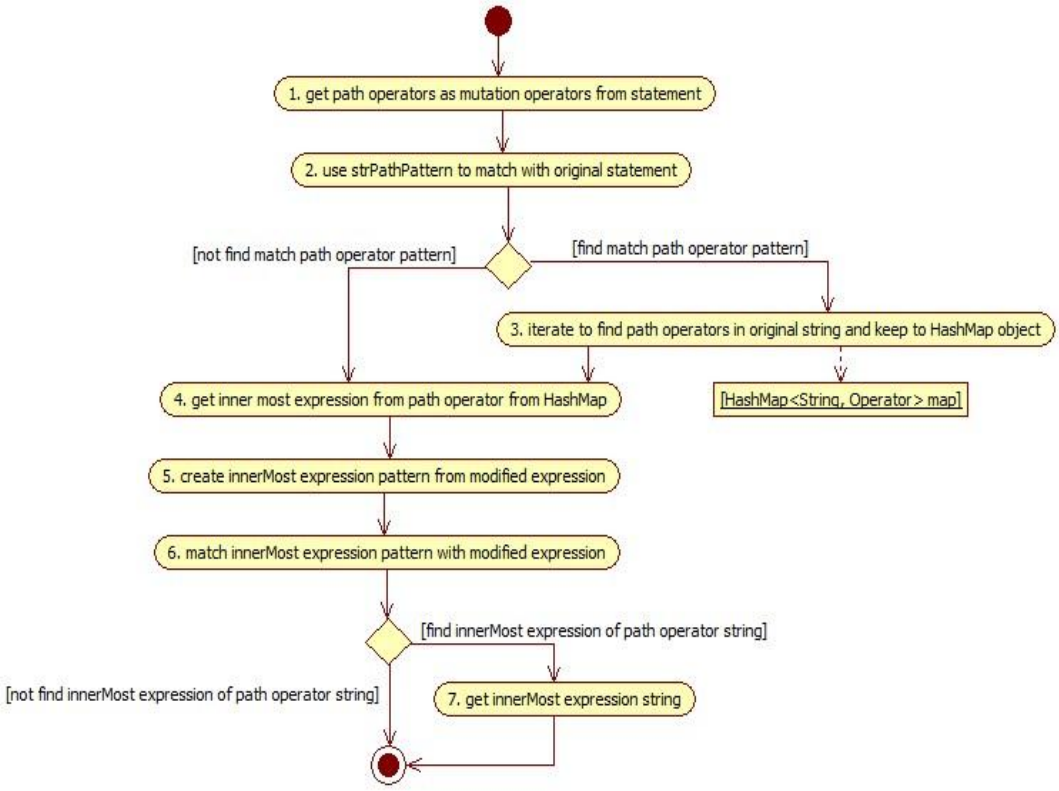
2. โดยใช้รูปแบบ /|// เพื่อใช้ในการหาเครื่องหมายที่เกี่ยวกับเส้นทางในนิพจน์หรือประพจน์

3. วนซ้ำเพื่อหาเครื่องหมายที่เกี่ยวกับเส้นทางในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละตัวดำเนินการที่เกี่ยวกับเส้นทางใน HashMap อ็อปเจ็กต์เพื่อหานิพจน์ข้างในสุดของเครื่องหมายที่เกี่ยวกับเส้นทางที่เก็บไว้

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยเครื่องหมายที่เกี่ยวกับเส้นทางที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของเครื่องหมายที่เกี่ยวกับเส้นทางแต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.110 แผนภาพกิจกรรมของ findInnerMostExpressionOfPathOperator

2.4) แผนภาพกิจกรรมของ performGenerateMutantWithECN ในภาพที่ 3.111 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม if, elseif, while, repeatUntil, target และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ ecnList

3. ตรวจสอบว่า ecnList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

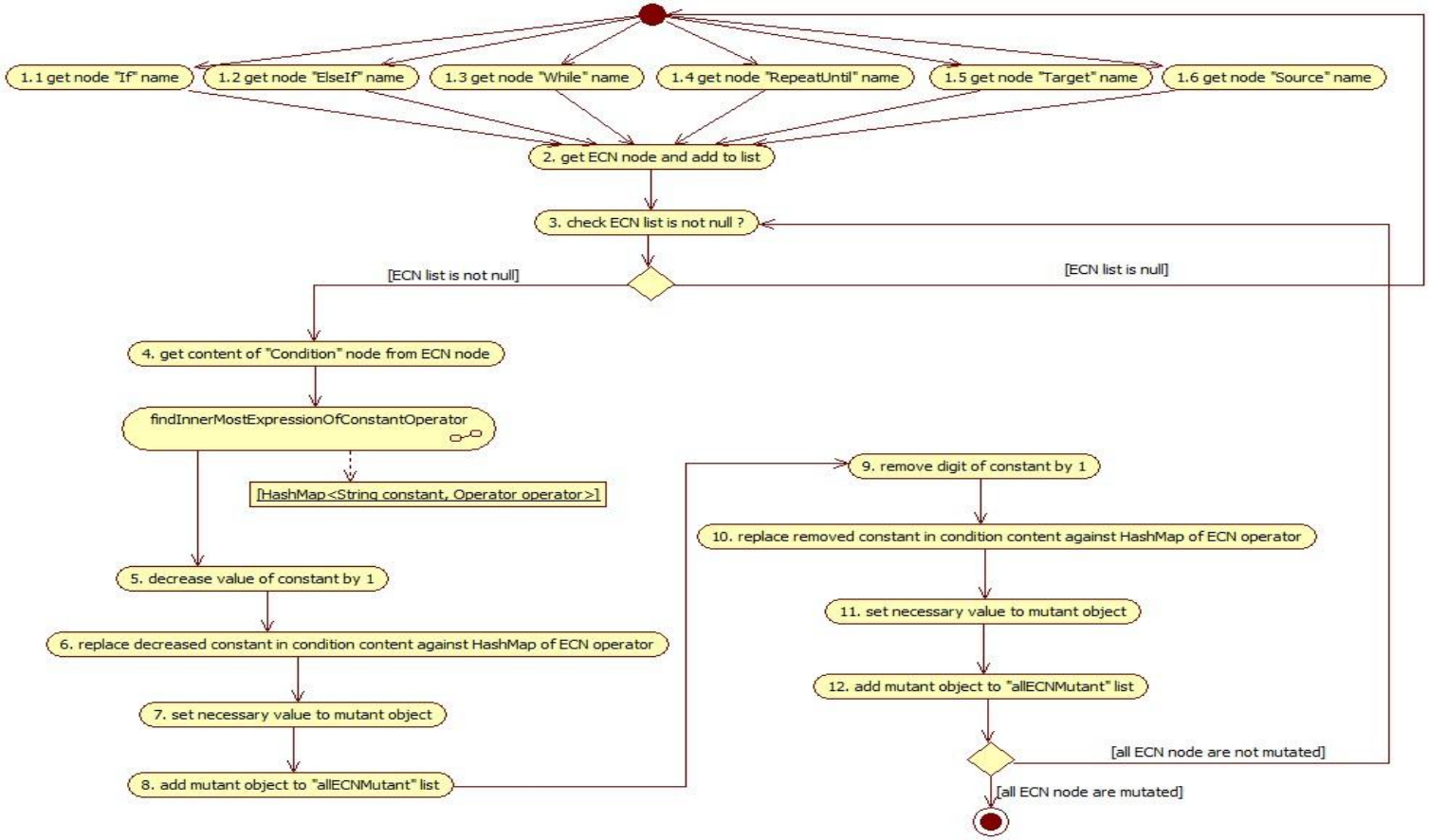
4. ถ้า ecnList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ออกมา

หมายเหตุ: หลังจากตรวจสอบว่ารายการ ecnList มีข้อมูลแล้ว โปรแกรมจะดำเนินการfindInnerMostExpressionOfConstantOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.112

5. โปรแกรมจะหาค่าคงที่ภายในนิพจน์หรือประพจน์ของเนื้อหาในโหนด condition จากนั้นโปรแกรมเก็บค่าคงที่ไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $sum <= 210 โปรแกรมจะเก็บค่าคงที่ และตำแหน่งของค่าคงที่ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการลดค่าลง 1 และการลบตำแหน่งออกไป 1 ตำแหน่ง ของค่าคงที่ในประพจน์นั้น

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allECNMutants



ภาพที่ 3.111 แผนภาพกิจกรรม performGenerateMutantWithECN

แผนภาพกิจกรรม findInnerMostExpressionOfConstantOperator มีขั้นตอนดังต่อไปนี้

1. ค้นหาค่าคงที่เสมือนเป็นตัวดำเนินการจากประพจน์ที่ได้จากโหนด condition

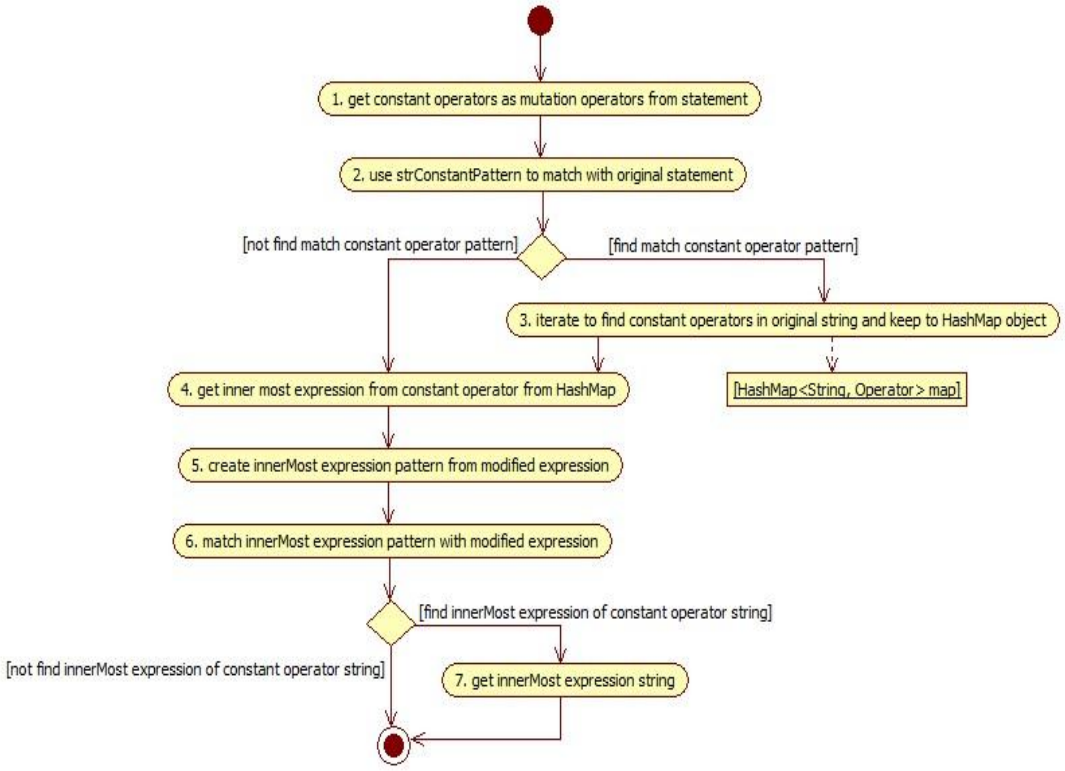
2. โดยใช้รูปแบบ \\d+ เพื่อใช้ในการหาค่าคงที่ในนิพจน์หรือประพจน์

3. วนซ้ำเพื่อหาค่าคงที่ในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละค่าคงที่ใน HashMap อ็อปเจ็กต์ เพื่อหานิพจน์ข้างในสุดของค่าคงที่ที่เก็บไว้

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยค่าคงที่ที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของค่าคงที่แต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.112 แผนภาพกิจกรรม findInnerMostExpressionOfConstantOperator

2.5) แผนภาพกิจกรรมของ performGenerateMutantWithEEU ในภาพที่ 3.113 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม from, to, if, elseif, while, repeatUntil, targets และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ eeuList

3. ตรวจสอบว่า eeuList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า eeuList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด from to หรือ condition ออกมา

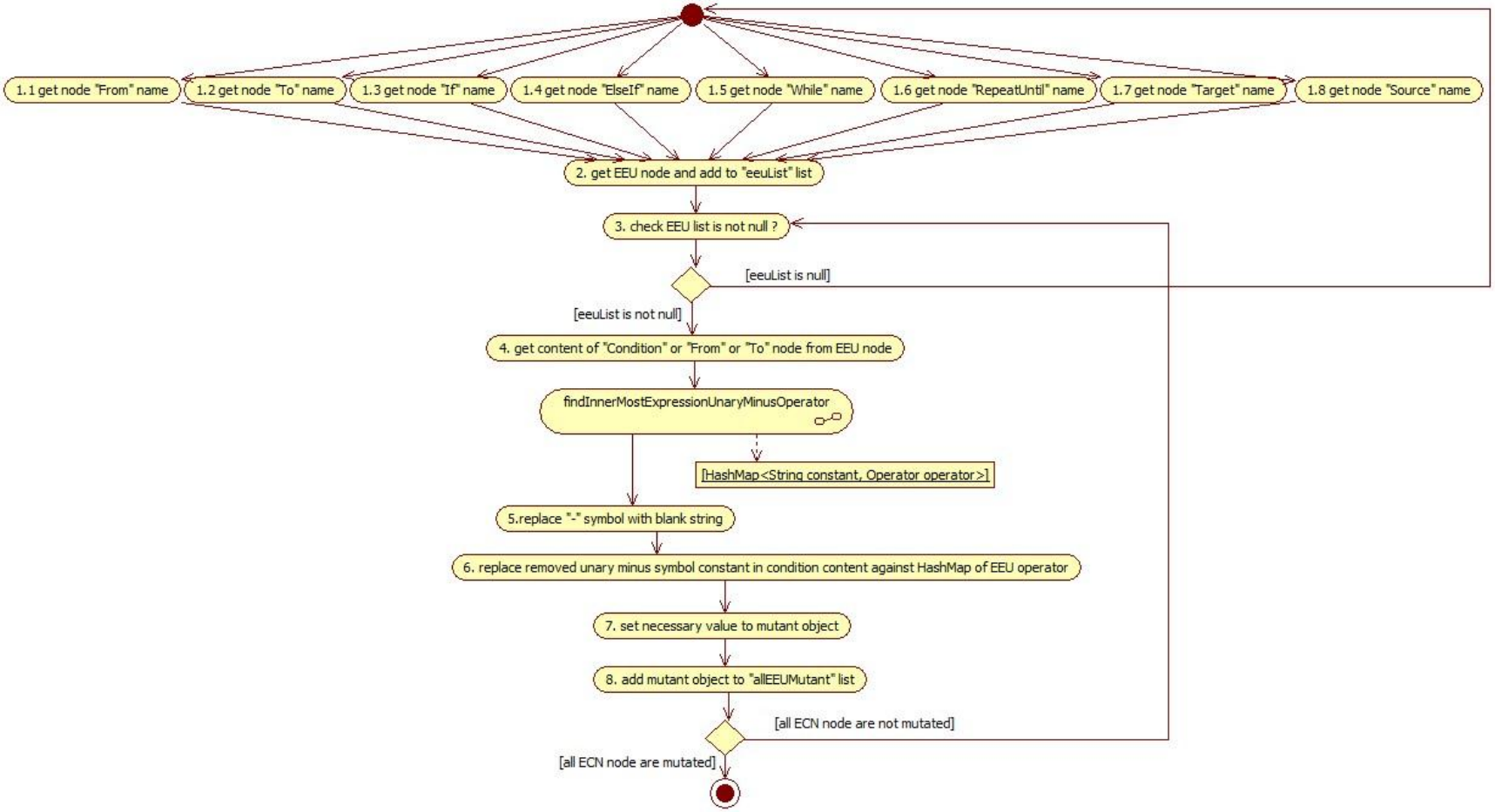
หมายเหตุ: หลังจากตรวจสอบในรายการ eeuList มีข้อมูลแล้ว โปรแกรมจะดำเนินการ findInnerMostExpressionOfUnaryMinusOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.114

5. โปรแกรมจะหาค่าคงที่ที่ติดลบภายในนิพจน์หรือประพจน์ของเนื้อหาในโหนด from to หรือ condition จากนั้นโปรแกรมเก็บค่าคงที่ที่ติดลบไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น -1 + $counter โปรแกรมจะเก็บค่าคงที่ -1 และตำแหน่งของค่าคงที่ -1 ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการลบเครื่องหมายลบของค่าคงที่ในนิพจน์หรือประพจน์นั้น

6. แทนที่ค่าคงที่ค่าใหม่ลงในนิพจน์หรือประพจน์โดยพิจารณาจากตำแหน่งของค่าคงที่ใน HashMap อ็อปเจ็กต์

7. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

8. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEEUMutants



ภาพที่ 3.113 แผนภาพกิจกรรม performGenerateMutantWithEEU

แผนภาพกิจกรรม findInnerMostExpressionOfUnaryMinusOperator มีขั้นตอนดังต่อไปนี้

1. ค้นหาค่าคงที่ที่ติดลบเสมือนเป็นตัวดำเนินการจากประพจน์ที่ได้จากโหนด from to และ condition

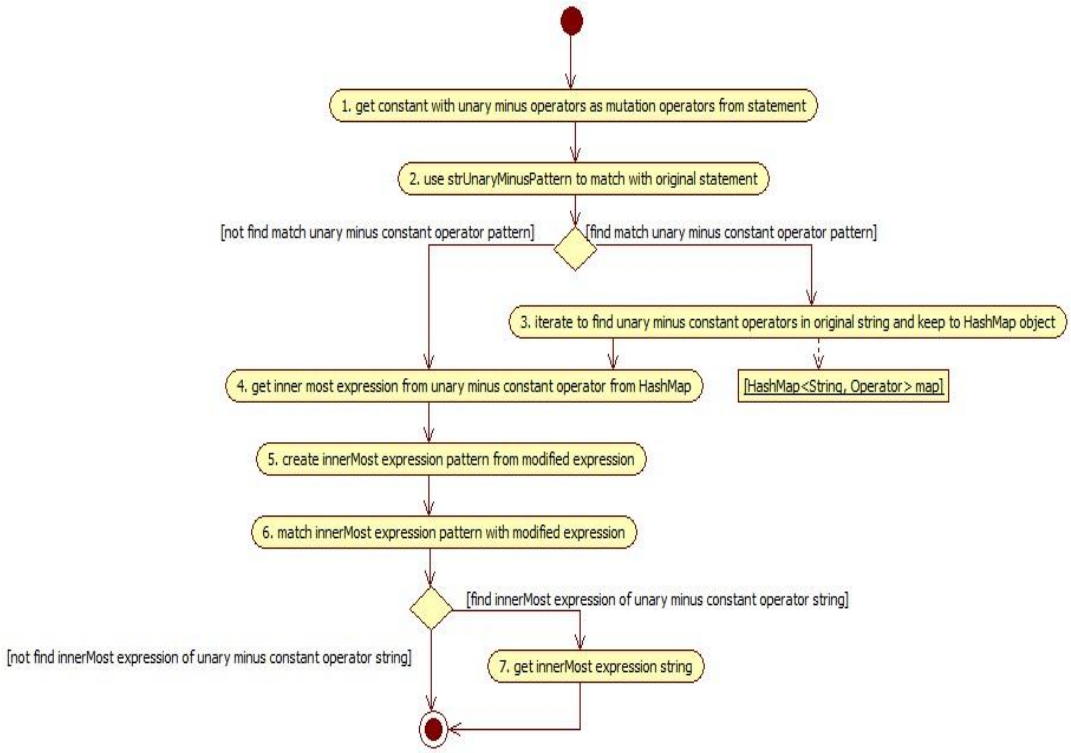
2. โดยใช้รูปแบบ -\\d+ เพื่อใช้ในการหาค่าคงที่ที่ติดลบในนิพจน์หรือประพจน์

3. วนซ้ำเพื่อหาค่าคงที่ในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละค่าคงที่ใน HashMap อ็อปเจ็กต์ เพื่อหานิพจน์ข้างในสุดของค่าคงที่ที่เก็บไว้

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยค่าคงที่ที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของค่าคงที่แต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.114 แผนภาพกิจกรรม findInnerMostExpressionOfUnaryMinusOperator

2.6) แผนภาพกิจกรรมของ performGenerateMutantWithEMD ในภาพที่ 3.115 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม wait และ repeatEvery

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ emdList

3. ตรวจสอบว่า emdList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า emdList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในแอตทริบิวต์ for ออกมา

หมายเหตุ: หลังจากตรวจสอบว่าในรายการ emdList มีข้อมูลแล้ว โปรแกรมจะดำเนินการ decreaseHalfValueOfDurationTime ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.116

5. หลังจากลดค่าของระยะเวลาในการรอลงครึ่งหนึ่งแล้ว โปรแกรมจะกำหนดค่าใหม่ให้กับแอตทริบิวต์ for

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

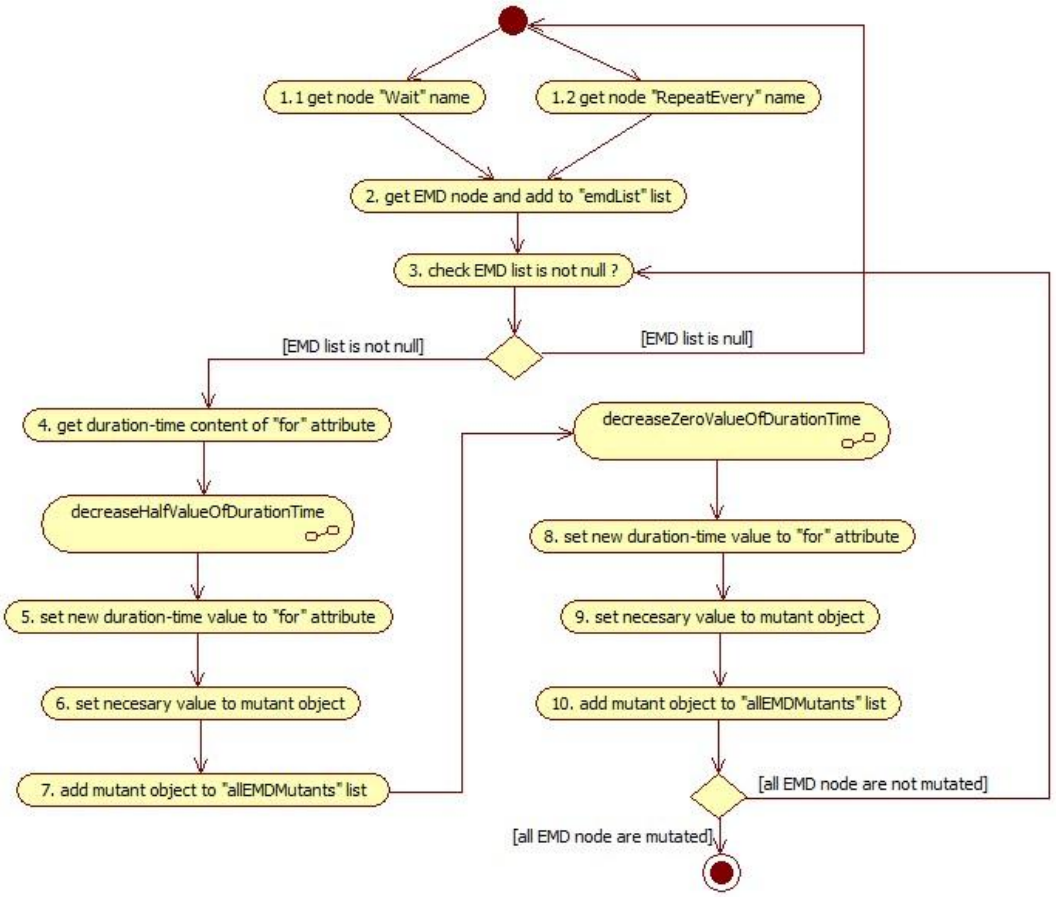
7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEMDMutants

หมายเหตุ: หลังจากตรวจสอบรายการ emdList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ decreaseZeroValueOfDurationTime ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.117

8. หลังจากลดค่าของระยะเวลาในการรอลงเป็นศูนย์แล้ว โปรแกรมจะกำหนดค่าใหม่ให้กับแอตทริบิวต์ for

9. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

10. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEMDMutants



ภาพที่ 3.115 แผนภาพกิจกรรม performGenerateMutantWithEMD

แผนภาพกิจกรรมของ decreaseHalfValueOfDurationTime ดังภาพที่ 3.116 สามารถอธิบายได้ ดังนี้

1. นำข้อมูลช่วงเวลาการรอจากแอตทริบิวต์ for เช่น “P19Y10M28DT10H37M46S”

2. แยกข้อมุลช่วงเวลาของการรอออกเป็น 2 ค่าคือ คาบเวลา “P19Y10M28D” และระยะเวลา “T10H37M46S”

3. แยกข้อมูลของคาบเวลาออกเป็นปี เดือน และวัน

4. แยกข้อมุลของระยะเวลาออกเป็นชั่วโมง นาที และวินาที

5. นำค่าทั้ง 6 มารวมกันให้อยู่ในหน่วยที่เล็กที่สุด คือ วินาที

6. จากนั้นนำค่าที่รวมกันได้ในข้อที่ 5 มาลดลงครึ่งหนึ่ง จากนั้นแปลงค่าที่ได้ให้อยู่ในหน่วยปี โดยการหารด้วย 12*30*24*60*60

7. เก็บค่าของปีไว้โดยในรูปของจำนวนเต็ม ส่วนค่าที่เป็นทศนิยมนำไปแปลงเป็นค่าของเดือนต่อไป

8. เก็บค่าของเดือนที่เป็นจำนวนเต็มซึ่งหาจากการนำจุดทศนิยมในข้อที่ 7 มาคูณด้วย 12 ส่วนค่าที่เป็นทศนิยมนำไปแปลงเป็นค่าของวันต่อไป

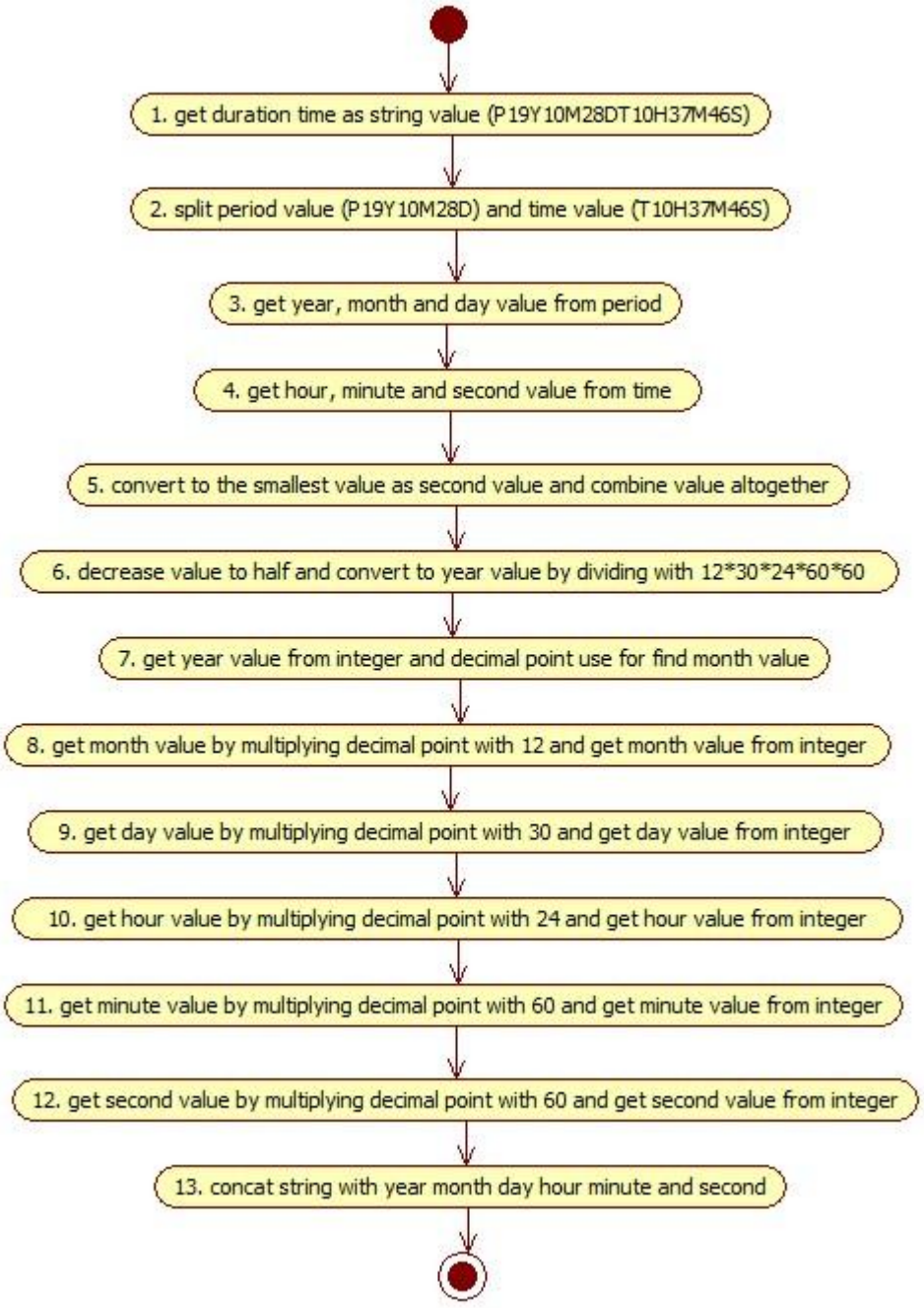
9. เก็บค่าของวันที่เป็นจำนวนเต็มซึ่งหาจากการนำจุดทศนิยมในข้อที่ 8 มาคูณด้วย 30 ส่วนค่าที่เป็นทศนิยมนำไปแปลงเป็นค่าของชั่วโมงต่อไป

10. เก็บค่าของชั่วโมงที่เป็นจำนวนเต็มซึ่งหาจากการนำจุดทศนิยมในข้อที่ 9 มาคูณด้วย 24 ส่วนค่าที่เป็นทศนิยมนำไปแปลงเป็นค่าของนาทีต่อไป

11. เก็บค่าของนาทีที่เป็นจำนวนเต็มซึ่งหาจากการนำจุดทศนิยมในข้อที่ 10 มาคูณด้วย 60 ส่วนค่าที่เป็นทศนิยมนำไปแปลงเป็นค่าของวินาทีต่อไป

12. เก็บค่าของวินาทีซึ่งหาจากการนำจุดทศนิยมในข้อที่ 11 มาคูณด้วย 60 และทำให้เป็นจำนวนเต็มโดยการปัดเศษ

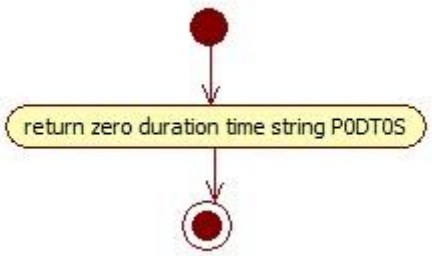
13. นำค่าของปี เดือน วัน ชั่วโมง นาที วินาที มารวมกันในรูปแบบดังนี้ “P9Y11M14DT5H18M53S”



ภาพที่ 3.116 แผนภาพกิจกรรม decreaseHalfValueOfDurationTime

แผนภาพกิจกรรมของ decreaseZeroValueOfDurationTime อธิบายได้ ดังนี้

กำหนดข้อมูลช่วงเวลาที่ไม่เกิดการรอให้กับแอตทริบิวต์ for คือ กำหนดค่าเป็น “P0DT0S”



ภาพที่ 3.117 แผนภาพกิจกรรม decreaseZeroValueOfDurationTime

2.7) แผนภาพกิจกรรมของ performGenerateMutantWithEMF ในภาพที่ 3.118 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม wait และ onAlarm

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ emfList

3. ตรวจสอบว่า emfList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า emfList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในแอตทริบิวต์ until ออกมา

หมายเหตุ: หลังจากตรวจสอบว่ารายการ emfList มีข้อมูลแล้ว โปรแกรมจะดำเนินการ decreaseHalfValueOfDeadlineTime ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.119

5. หลังจากลดค่าของระยะเวลาในการรอลงครึ่งหนึ่งแล้ว โปรแกรมจะกำหนดค่าใหม่ให้กับแอตทริบิวต์ until

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

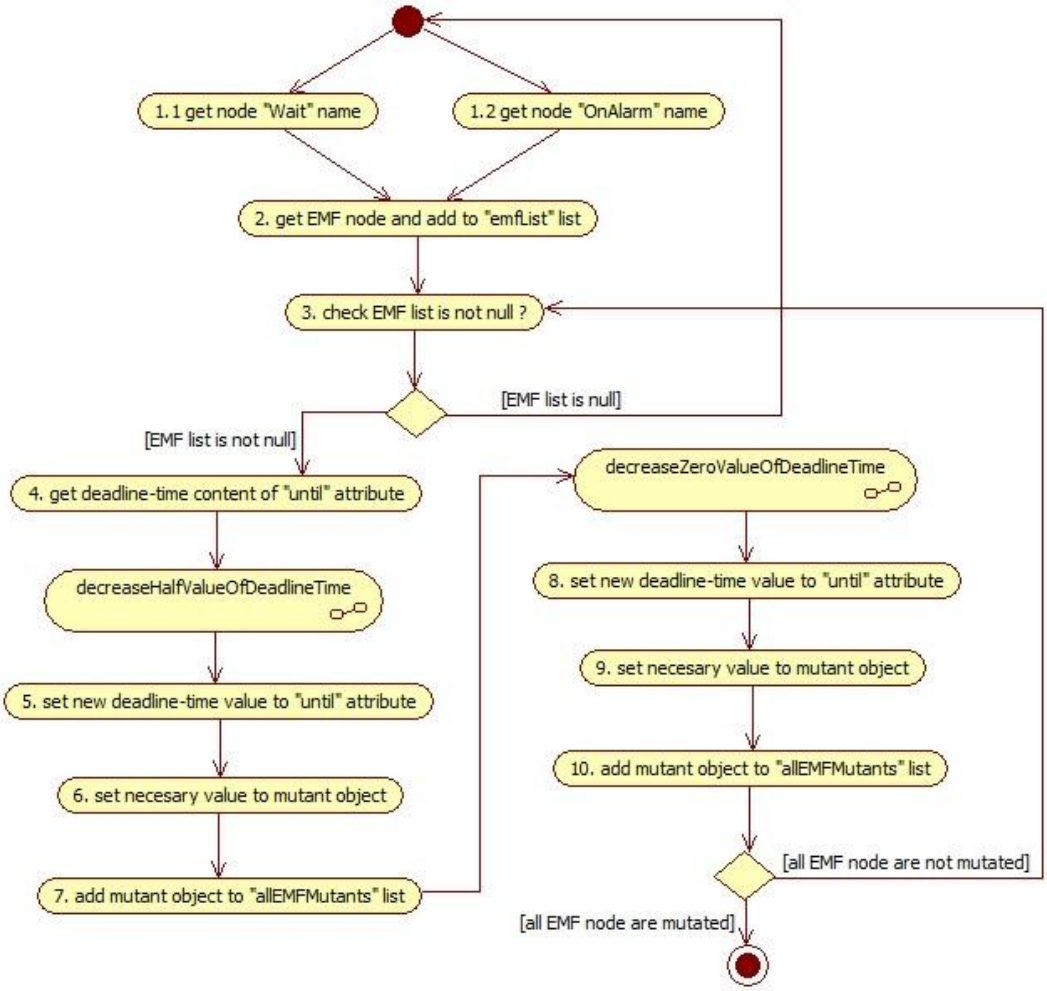
7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEMFMutants

หมายเหตุ: หลังจากตรวจสอบรายการ emfList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ decreaseZeroValueOfDeadlineTime ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.120

8. หลังจากลดค่าของระยะเวลาในการรอลงเป็นศูนย์แล้ว โปรแกรมจะกำหนดค่าใหม่ให้กับแอตทริบิวต์ until

9. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

10. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allEMFMutants



ภาพที่ 3.118 แผนภาพกิจกรรม performGenerateMutantWithEMF

แผนภาพกิจกรรมของ decreaseHalfValueOfDeadlineTime อธิบายได้ ดังนี้

1. นำข้อมูลช่วงเวลาเส้นตาย จากแอตทริบิวต์ until เช่น “2011-10-28T19:29:19”

2. แยกข้อมูลช่วงเวลาเส้นตายออกเป็น 2 ค่าคือ วันที่ “2011-10-28” และเวลา “19:29:19”

3. เก็บข้อมูลวันที่ปัจจุบันและเวลาปัจจุบันจากระบบ

4. แยกข้อมูลของวันที่เส้นตายและวันที่ปัจจุบันด้วย “-” เช่น วันที่ 2011-10-28 ถูกแบ่งออกมาได้ 3 ค่า คือ ปีเป็น 2011 เดือนเป็น 10 และวันเป็น 28 และหาผลต่างระหว่างวันที่เส้นตายและวันที่ปัจจุบันของวัน เดือน และปี ตามลำดับ

5. นำผลต่างเหล่านั้นทั้ง 3 ค่า มาหารด้วย 2 เพื่อลดค่าลงครึ่งหนึ่ง

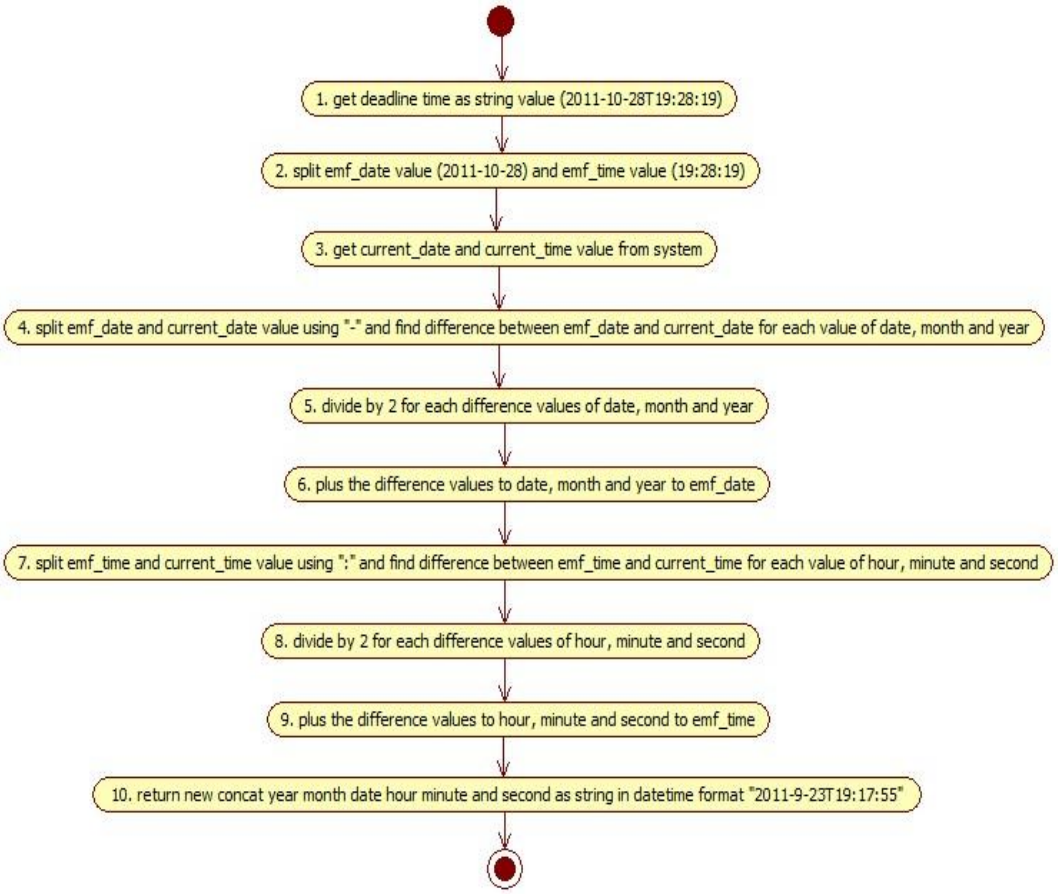
6. นำผลที่ได้จากข้อที่ 5 มาบวกเพิ่มให้กับวัน เดือน และปีของวันที่ปัจจุบัน

7. แยกข้อมูลของเวลาเส้นตายและเวลาปัจจุบันด้วย “:” เช่น วันที่ 19:29:19 ถูกแบ่งออกมาได้ 3 ค่า คือ ชั่วโมงเป็น 19 นาทีเป็น 29 และวินาทีเป็น 19 และหาผลต่างระหว่างเวลาเส้นตายและเวลาปัจจุบัน ของชั่วโมง นาที และวินาที ตามลำดับ

8. นำผลต่างเหล่านั้นทั้ง 3 ค่า มาหารด้วย 2 เพื่อลดค่าลงครึ่งหนึ่ง

9. นำผลที่ได้จากข้อที่ 8 มาบวกเพิ่มให้กับชั่วโมง นาที และวินาทีของปัจจุบัน

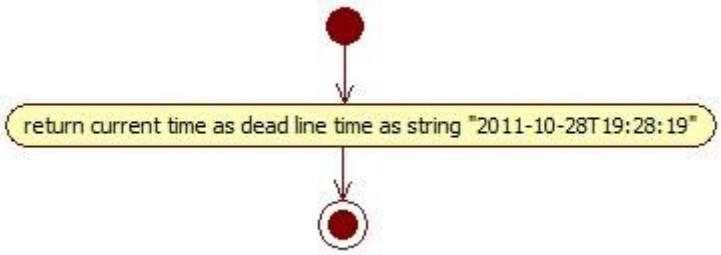
10. นำค่าที่ได้จากข้อที่ 6 และ 9 มาจัดให้อยู่ในรูปแบบวันที่และเวลา เช่น “2011-9-23T19:17:55”



ภาพที่ 3.119 แผนภาพกิจกรรม decreaseHalfValueOfDeadlineTime

แผนภาพกิจกรรมของ decreaseHalfValueOfDeadlineTime อธิบายได้ ดังนี้

กำหนดข้อมูลช่วงเวลาเส้นตายเป็นวันที่ปัจจุบันให้กับแอตทริบิวต์ until คือ กำหนดค่าเป็น “2011-9-23T19:17:55”



ภาพที่ 3.120 แผนภาพกิจกรรม decreaseZeroValueOfDeadlineTime

2.8) แผนภาพกิจกรรมของ performGenerateMutantWithELL ในภาพที่ 3.121 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม if, elseif, while, repeatUntil, target และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ ellList

3. ตรวจสอบว่า ellList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

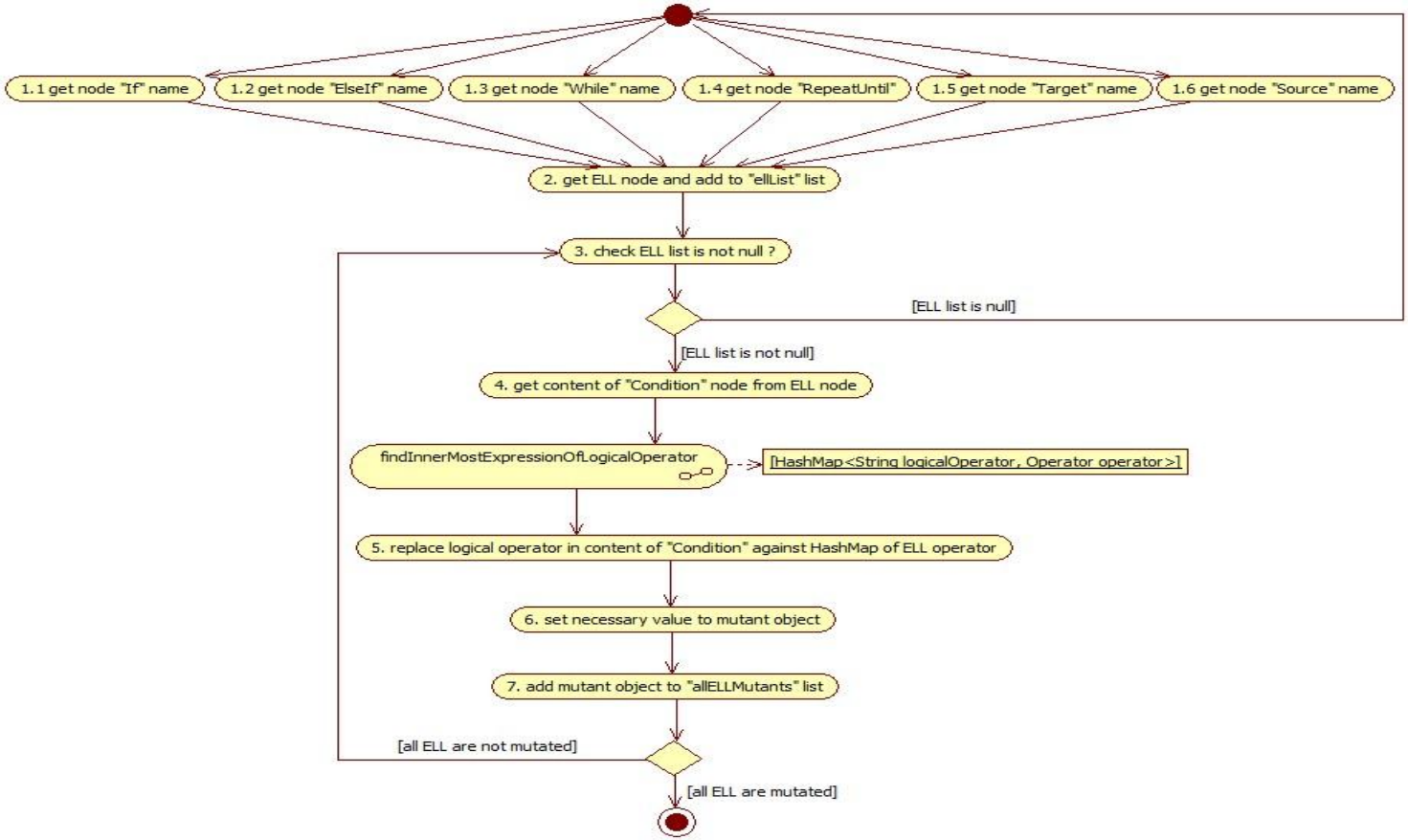
4. ถ้า ellList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ออกมา

หมายเหตุ: หลังจากตรวจสอบในรายการ ellList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ findInnerMostExpressionOfLogicalOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.122

5. โปรแกรมจะหาเครื่องหมายทางตรรกะ (and, or) ภายในนิพจน์หรือประพจน์ของ condition จากนั้นโปรแกรมเก็บเครื่องหมายไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $input.A and $input.C โปรแกรมจะเก็บเครื่องหมาย and และตำแหน่งของเครื่องหมาย and ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการสลับที่เครื่องหมายทางตรรกะไปเป็นตัวอื่นๆ ที่ไม่ใช่เครื่องหมายในประพจน์

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allELLMutants



ภาพที่ 3.121 แผนภาพกิจกรรม performGenerateMutantWithELL

แผนภาพกิจกรรม findInnerMostExpressionOfLogicalOperator ในภาพที่ 3.122 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. ค้นหาเครื่องหมายทางตรรกะเสมือนเป็นตัวดำเนินการจากประพจน์ที่ได้จากโหนด condition

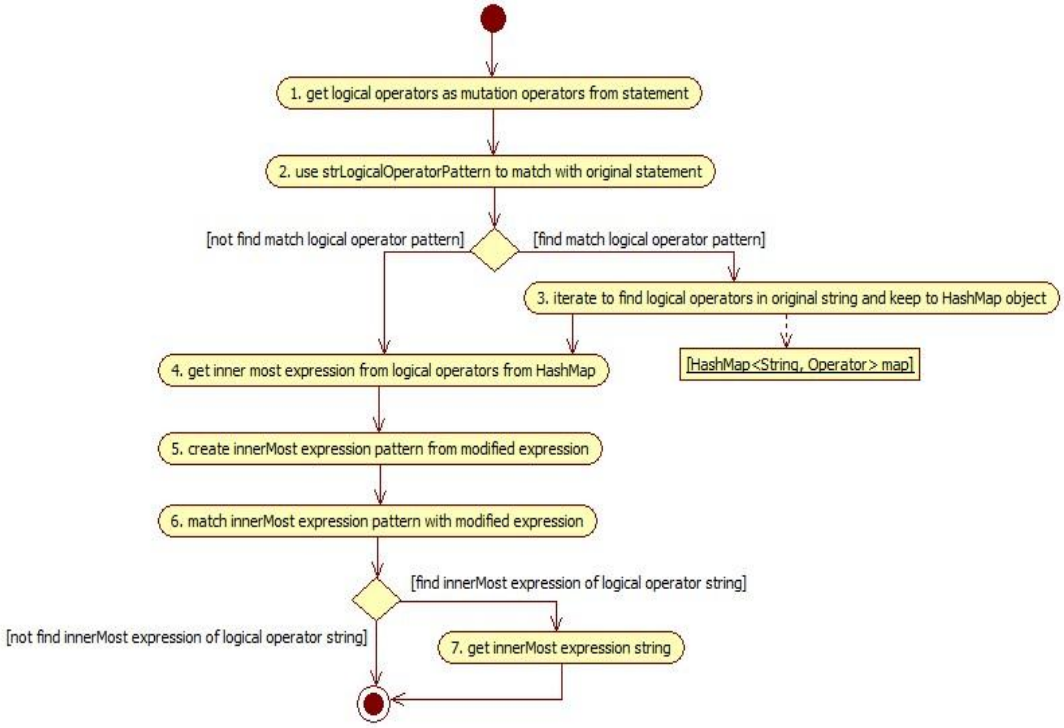
2. โดยใช้รูปแบบ and | or เพื่อใช้ในการหาเครื่องหมายทางตรรกะในประพจน์

3. วนซ้ำเพื่อหาเครื่องหมายทางตรรกะในประพจน์ จากนั้นโปรแกรมจะเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละเครื่องหมายทางตรรกะในอ็อปเจ็กต์ HashMap เพื่อหานิพจน์ข้างในสุดของเครื่องหมายทางตรรกะที่เก็บไว้

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยเครื่องหมายทางตรรกะที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของเครื่องหมายทางตรรกะแต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.122 แผนภาพกิจกรรม findInnerMostExpressionOfLogicalOperator

2.9) แผนภาพกิจกรรมของ performGenerateMutantWithERR ในภาพที่ 3.123 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม if, elseif, while, repeatUntil, targets และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ errList

3. ตรวจสอบว่า errList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

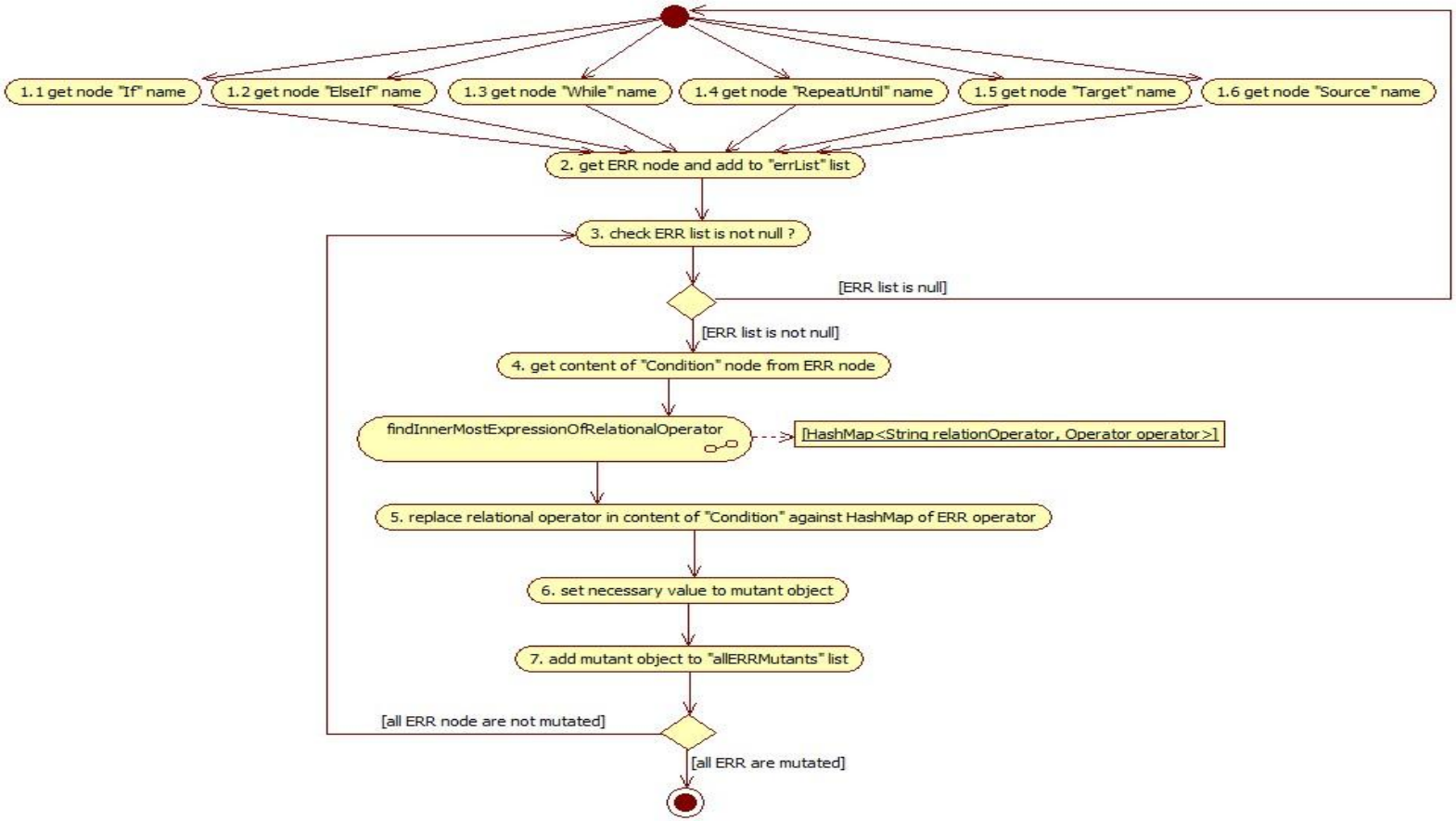
4. ถ้า errList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของนิพจน์หรือประพจน์ในโหนด condition ออกมา

หมายเหตุ: หลังจากตรวจสอบรายการ errList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ findInnerMostExpressionOfRelationOperator ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.124

5. โปรแกรมจะหาเครื่องหมายความสัมพันธ์ (=, !=, <, >, <=, >=) ภายในนิพจน์หรือประพจน์ของ condition จากนั้นโปรแกรมเก็บเครื่องหมายไว้ ตัวอย่างเช่น ในประพจน์ของ condition เป็น $input.A > $input.B = $input.C โปรแกรมจะเก็บเครื่องหมาย > และตำแหน่งของเครื่องหมาย > ไว้ใน HashMap อ็อปเจ็กต์ เพื่อใช้ในการสลับที่เครื่องหมายความสัมพันธ์ไปเป็นตัวอื่นๆ ที่ไม่ใช่เครื่องหมายในประพจน์

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allERRMutants



ภาพที่ 3.123 แผนภาพกิจกรรม performGenerateMutantWithERR

แผนภาพกิจกรรม findInnerMostExpressionOfRelationOperator มีขั้นตอนดังต่อไปนี้

1. ค้นหาเครื่องหมายความสัมพันธ์เสมือนเป็นตัวดำเนินการจากประพจน์ที่ได้จากโหนด condition

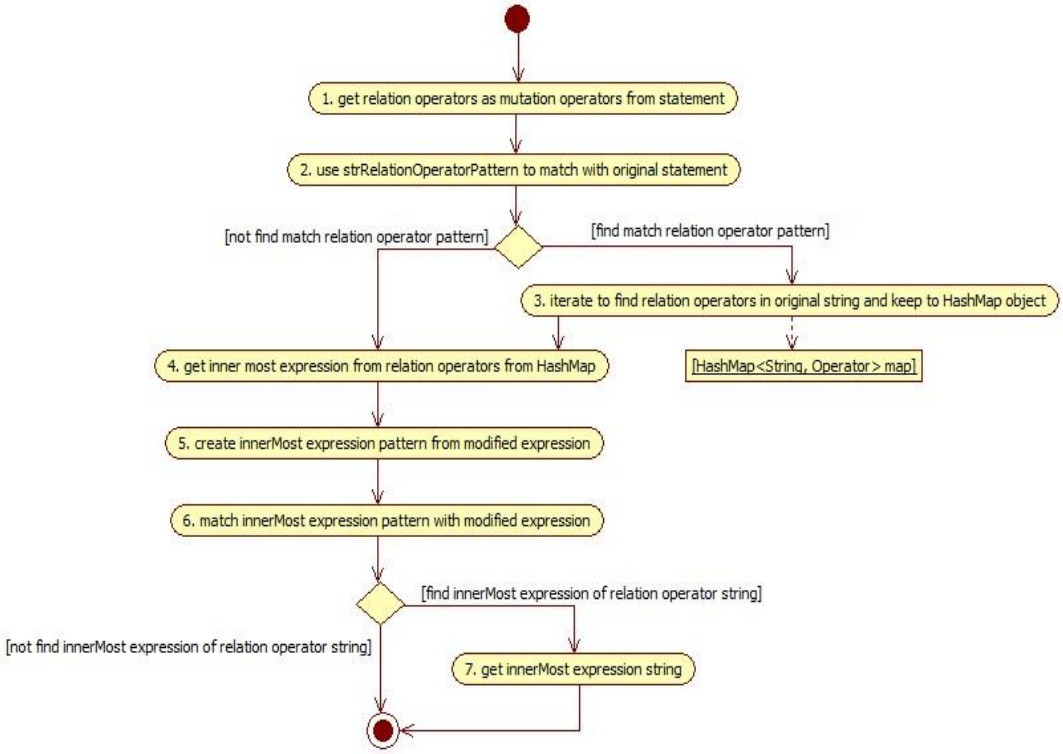
2. โดยใช้รูปแบบ =| !=| <| >| <= | >= เพื่อใช้ในการหาเครื่องหมายความสัมพันธ์ในประพจน์

3. วนซ้ำเพื่อหาเครื่องหมายความสัมพันธ์ในประพจน์ จากนั้นเก็บชื่อและตำแหน่งไว้ใน HashMap อ็อปเจ็กต์

4. วนซ้ำแต่ละเครื่องหมายความสัมพันธ์ใน HashMap อ็อปเจ็กต์ เพื่อหานิพจน์ข้างในสุดของเครื่องหมายความสัมพันธ์ที่เก็บไว้ โดย

5. ใช้รูปแบบ \\S+[^()]\\s+ ตามด้วยเครื่องหมายความสัมพันธ์ที่เก็บไว้ใน HashMap อ็อปเจ็กต์ และตามด้วยรูปแบบ \\s+\\S+[^()]

6. ใช้รูปแบบดังข้อที่ 5 เพื่อหานิพจน์ข้างในสุดของเครื่องหมายความสัมพันธ์แต่ละตัว จากนั้นเก็บนิพจน์ไว้ในมิวแตนท์อ็อปเจ็กต์



ภาพที่ 3.124 แผนภาพกิจกรรม findInnerMostExpressionOfRelationOperator

2.10) แผนภาพกิจกรรมของ performGenerateMutantWithACI ในภาพที่ 3.125 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม receive และ pick

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ aciList

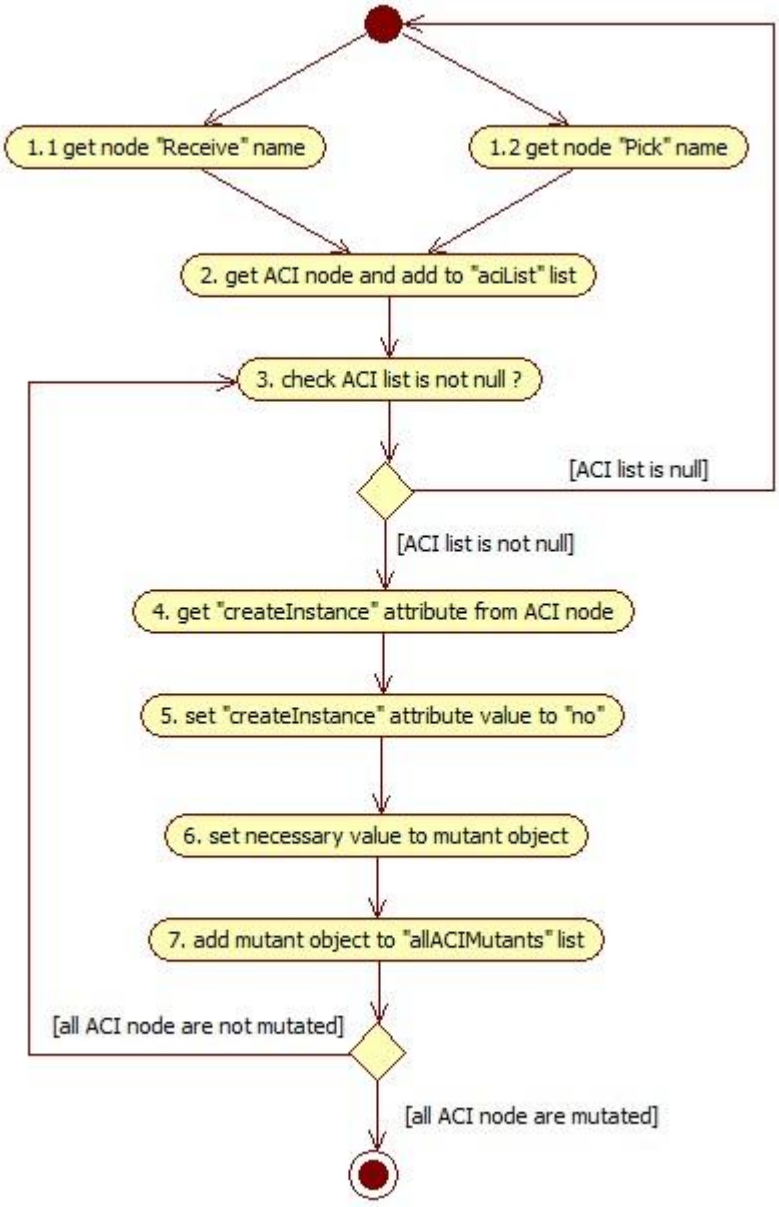
3. ตรวจสอบว่า aciList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้าในรายการ aciList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะทำการดึงค่าของแอตทริบิวต์ createInstance ออกมา

5. โปรแกรมจะกำหนดค่าแอตทริบิวต์ createInstance ให้มีค่าเป็น no

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAFPMutants



ภาพที่ 3.125 แผนภาพกิจกรรม performGenerateMutantWithACI

2.11) แผนภาพกิจกรรมของ performGenerateMutantWithAEL ในภาพที่ 3.126 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม assign, reply, receive, invoke, flow, sequence, forEach, pick, if, elseif, while, repeatUntil, targets และ source

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ aelList

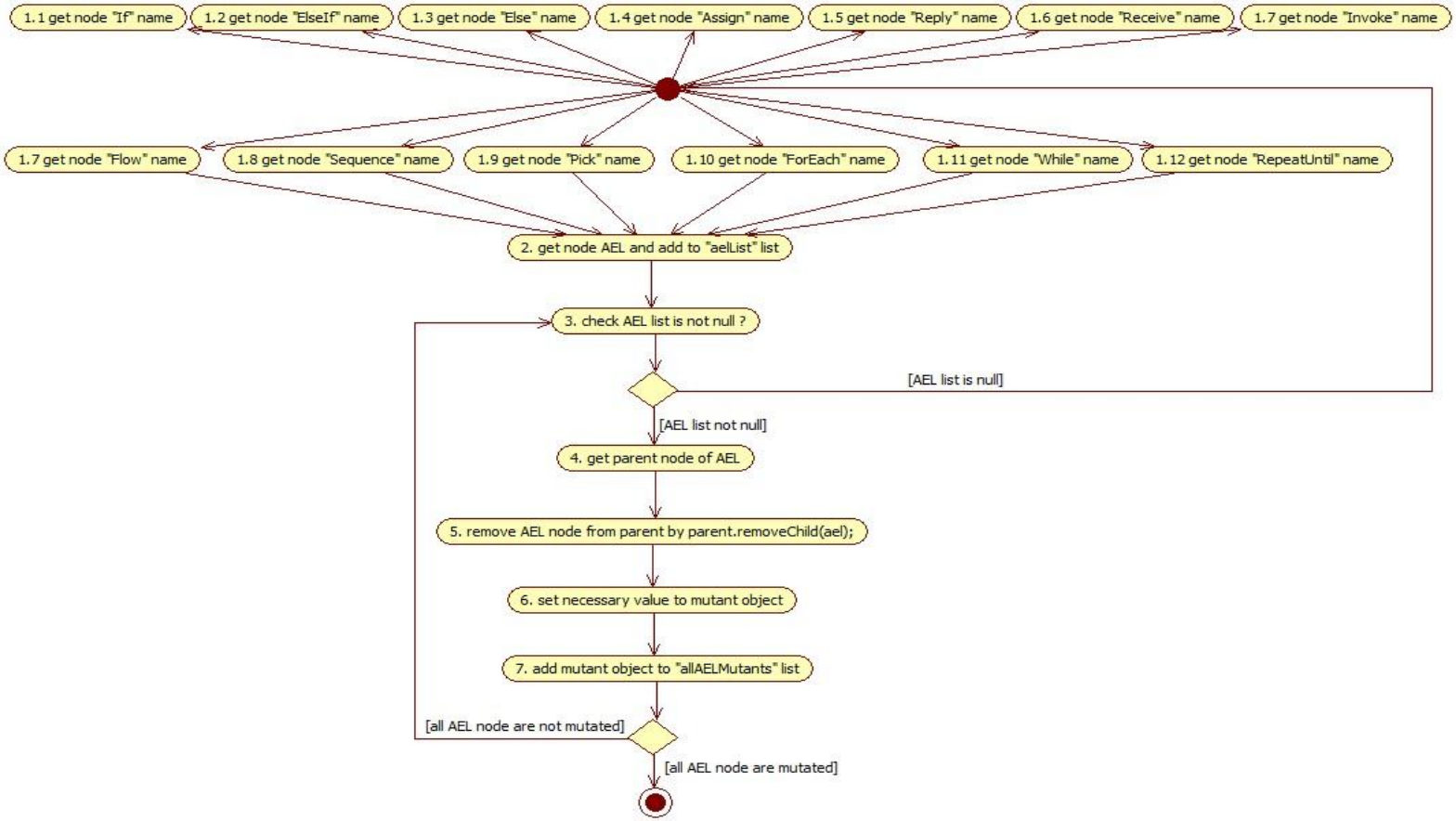
3. ตรวจสอบว่า aelList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า aelList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออกโดย ลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์ที่สร้างไปเก็บไว้ในรายการ allAELMutants



ภาพที่ 3.126 แผนภาพกิจกรรม performGenerateMutantWithAEL

2.12) แผนภาพกิจกรรมของ performGenerateMutantWithAFP ในภาพที่ 3.127 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม forEach

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ afpList

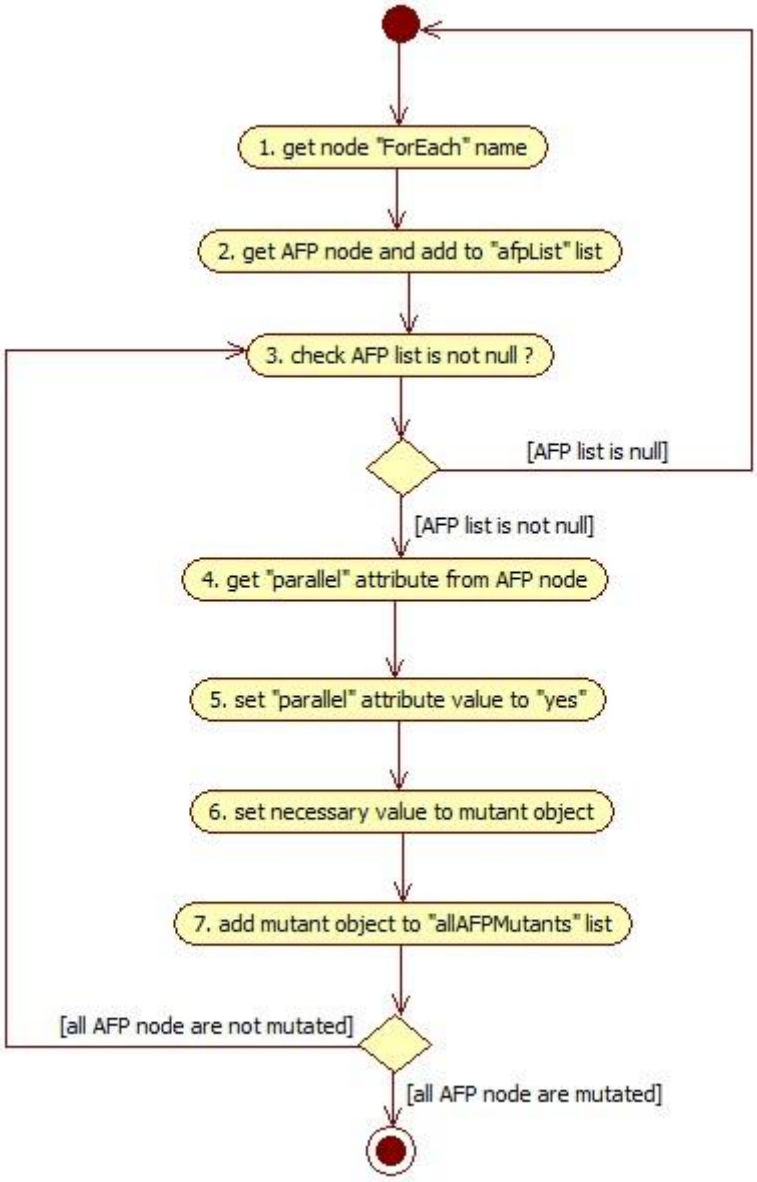
3. ตรวจสอบว่า afpList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า afpList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะดึงค่าของแอตทริบิวต์ parallel ออกมา

5. โปรแกรมจะกำหนดค่าแอตทริบิวต์ parallel ให้มีค่าเป็น yes

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAFPMutants



ภาพที่ 3.127 แผนภาพกิจกรรม performGenerateMutantWithAFP

2.13) แผนภาพกิจกรรมของ performGenerateMutantWithAIE ในภาพที่ 3.128 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม elseif และ else

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ aieList

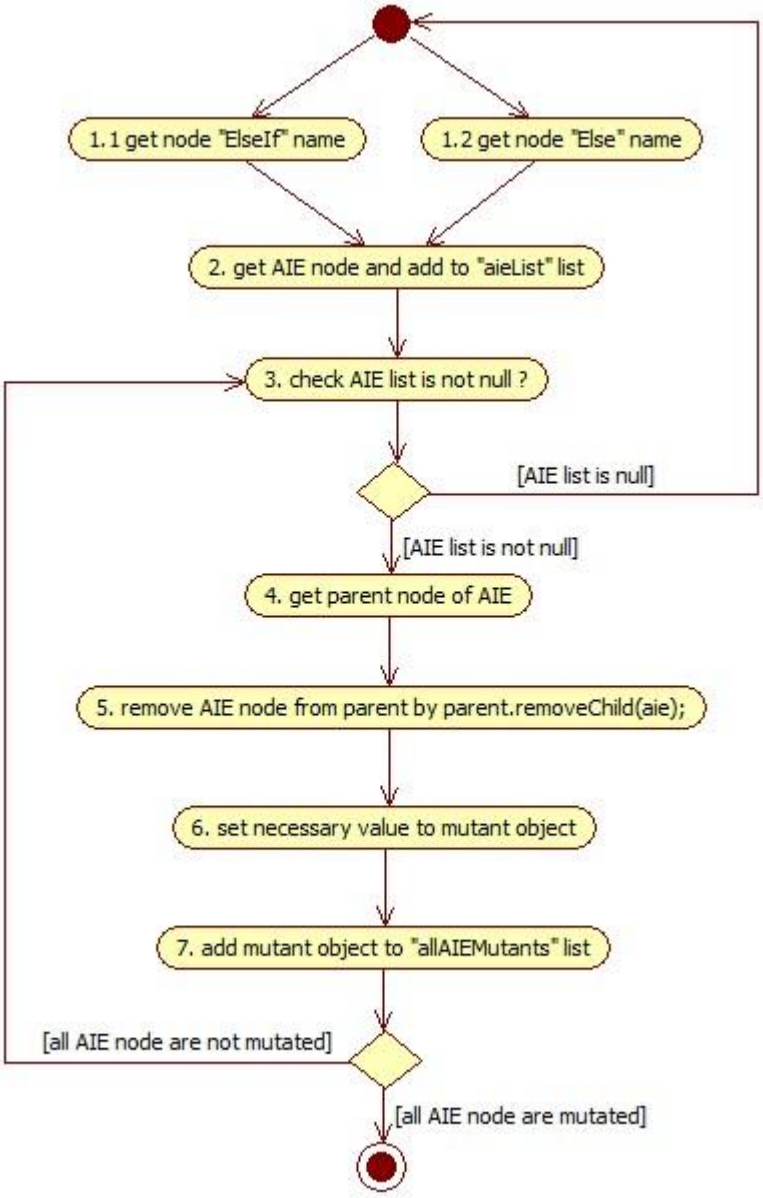
3. ตรวจสอบว่า aieList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า aieList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออกโดย ลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAIEMutants



ภาพที่ 3.128 แผนภาพกิจกรรม performGenerateMutantWithAIE

2.14) แผนภาพกิจกรรมของ performGenerateMutantWithAIS ในภาพที่ 3.129 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม scope, sequence, flow, และ if

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ aisList

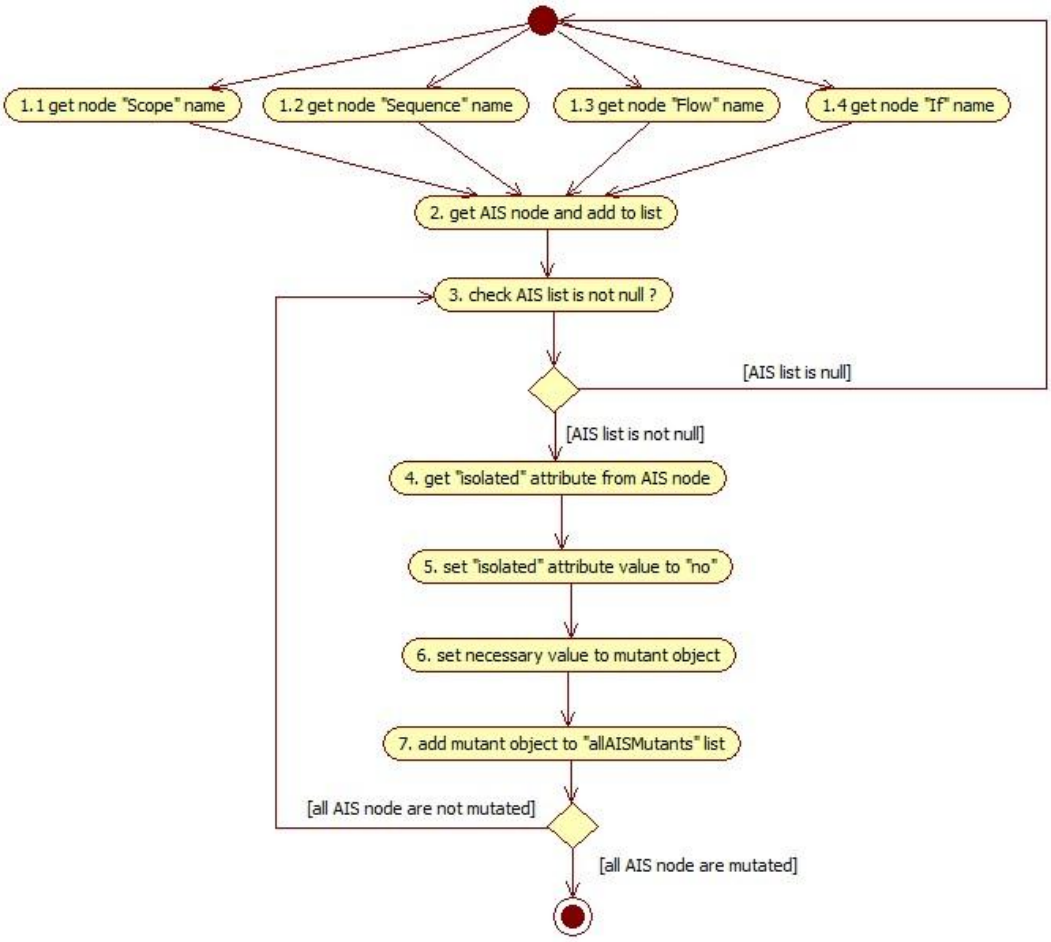
3. ตรวจสอบว่า aisList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้าในรายการ aisList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะดึงค่าของแอตทริบิวต์ isolated ออกมา

5. โปรแกรมจะกำหนดค่าแอตทริบิวต์ isolated ให้มีค่าเป็น no

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAISMutants



ภาพที่ 3.129 แผนภาพกิจกรรม performGenerateMutantWithAIS

2.15) แผนภาพกิจกรรมของ performGenerateMutantWithAJC ในภาพที่ 3.130 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม joinCondition

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ ajcList

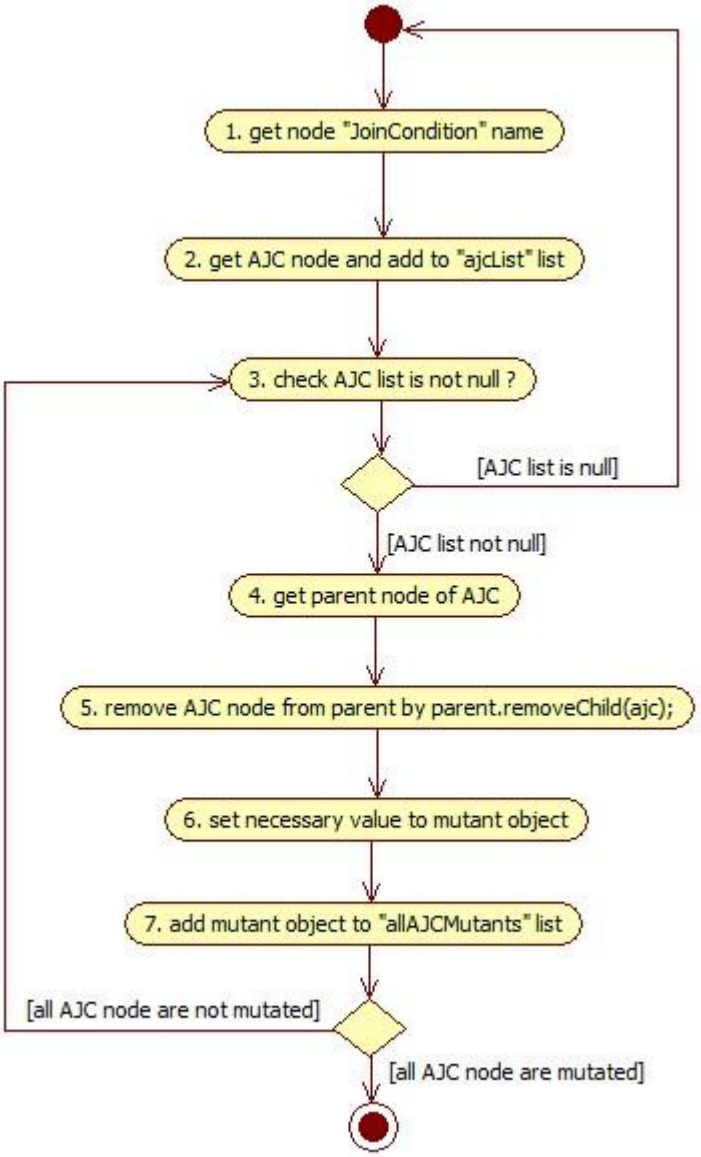
3. ตรวจสอบว่า ajcList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า ajcList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออกโดย ลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAJCMutants



ภาพที่ 3.130 แผนภาพกิจกรรม performGenerateMutantWithAJC

2.16) แผนภาพกิจกรรมของ performGenerateMutantWithAPA ในภาพที่ 3.131 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม pick และ eventHandler

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ apaList

3. ตรวจสอบว่า apaList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

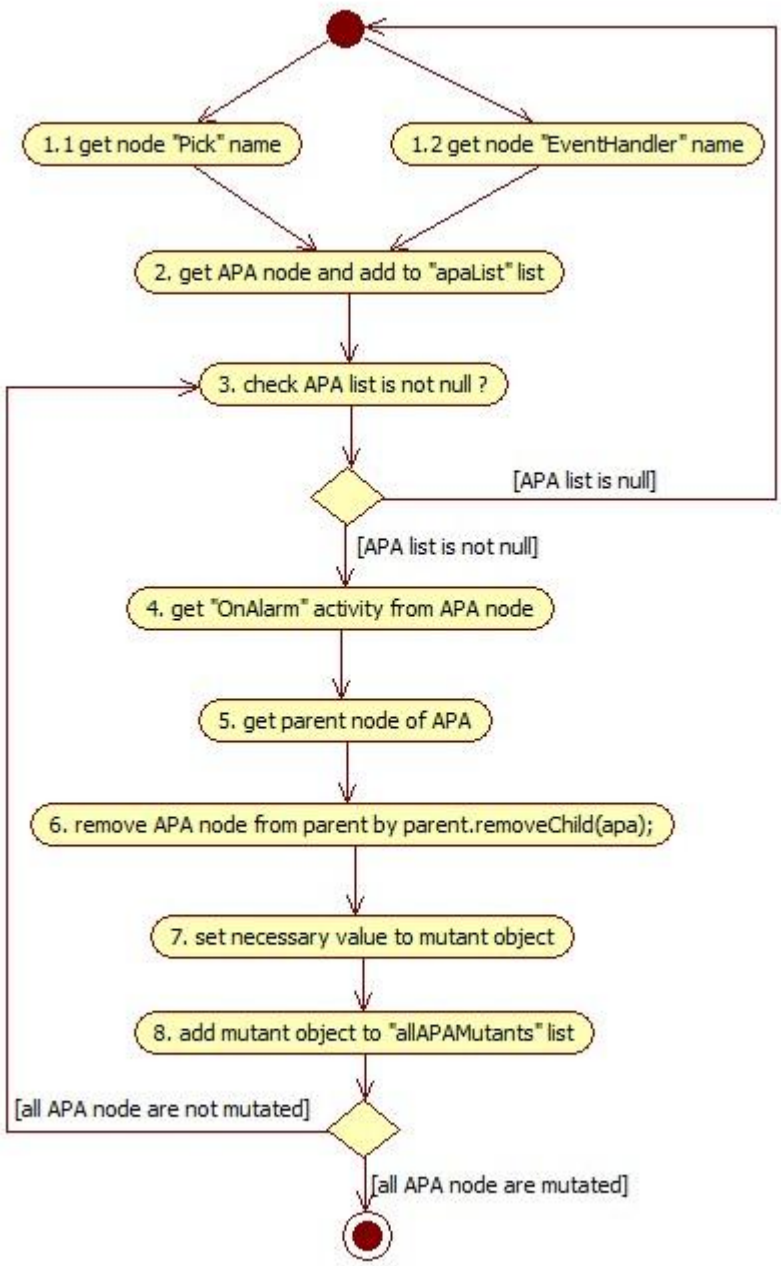
4. ถ้า apaList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำโหนดของ onAlarm ของโหนดดังกล่าวในข้อที่ 1

5. โปรแกรมจะนำข้อมูลโหนดของพ่อแม่ของโหนด onAlarm ออกมา

6. ลบโหนด onAlarm ออก โดยลบโหนดลูกออกจากโหนดพ่อแม่ที่ดึงมาในข้อที่ 5

7. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

8. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAPAMutants



ภาพที่ 3.131 แผนภาพกิจกรรม performGenerateMutantWithAPA

2.17) แผนภาพกิจกรรมของ performGenerateMutantWithAPM ในภาพที่ 3.132 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม pick

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ apmList

3. ตรวจสอบว่า apmList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

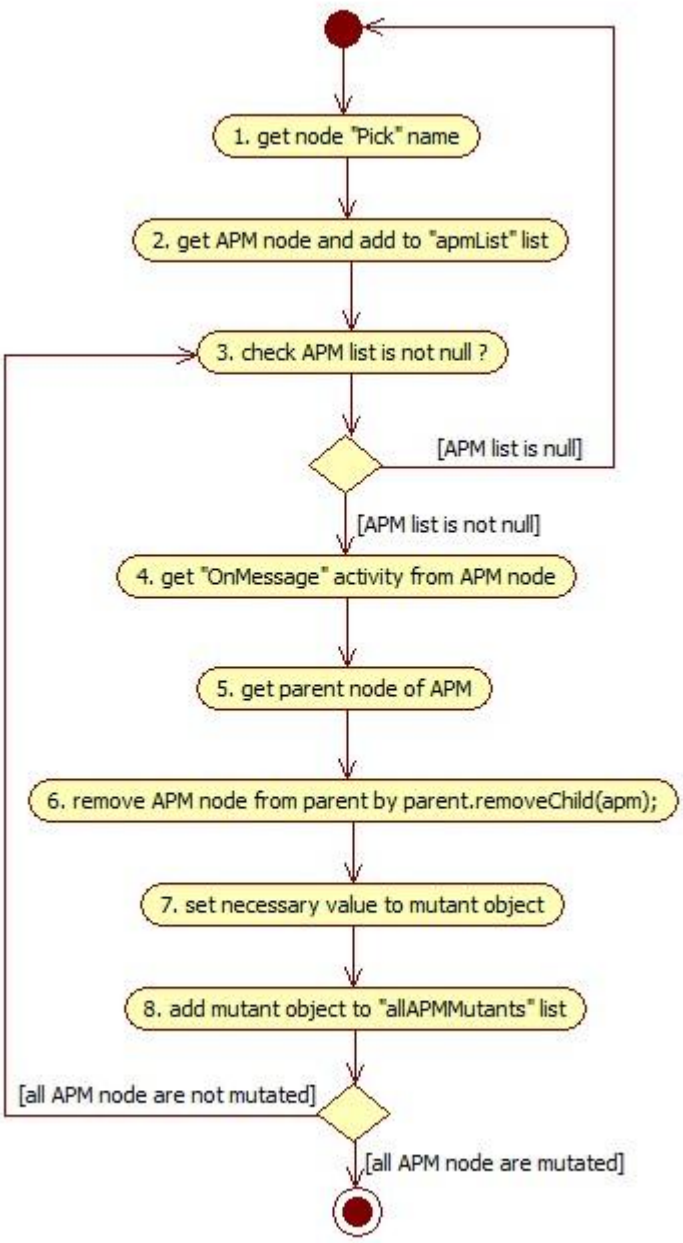
4. ถ้า apmList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำโหนดของ onMessage ของโหนดดังกล่าวในข้อที่ 1

5. โปรแกรมจะนำข้อมูลโหนดของพ่อแม่ของโหนด onMessage ออกมา

6. ลบโหนด onMessage ออก โดยลบโหนดลูกออกจากโหนดพ่อแม่ที่ดึงมาในข้อที่ 5

7. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

8. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAPMMutants



ภาพที่ 3.132 แผนภาพกิจกรรม performGenerateMutantWithAPM

2.18) แผนภาพกิจกรรมของ performGenerateMutantWithASF ในภาพที่ 3.133 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม sequence

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ asfList

3. ตรวจสอบว่า asfList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

หมายเหตุ: หลังจากตรวจสอบรายการ asfList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ createFlowActivityFromSequenceActivity ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.134

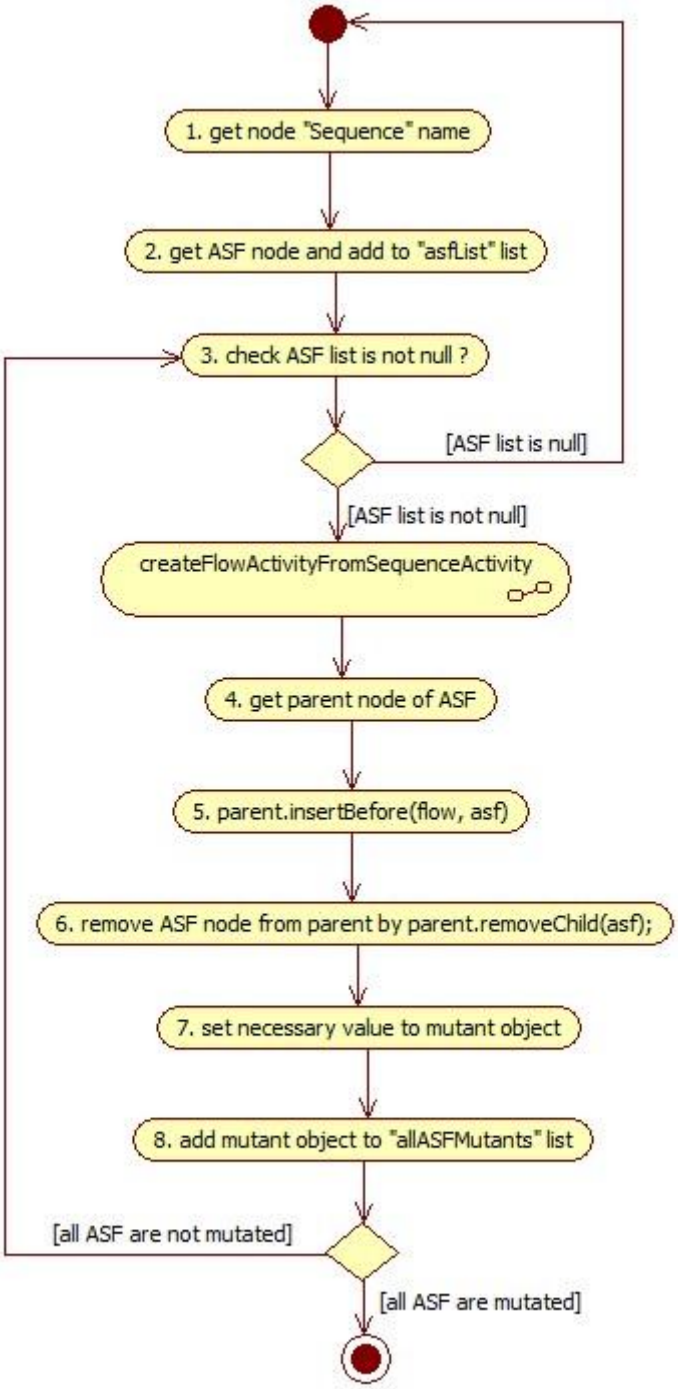
4. ถ้า asfList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะดึงข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1 ออกมา

5. นำโหนดกิจกรรม flow ที่สร้างโหนดกิจกรรม sequence จากแผนภาพกิจกรรม createFlowActivityFromSequence มาใส่ก่อนหน้าโหนด sequence ที่เราสนใจ

6. ลบโหนด sequence ที่ถูกใส่โหนด flow ก่อนหน้าออก โดยลบออกจากโหนดพ่อแม่ของโหนด sequence

7. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

8. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allASFMutants

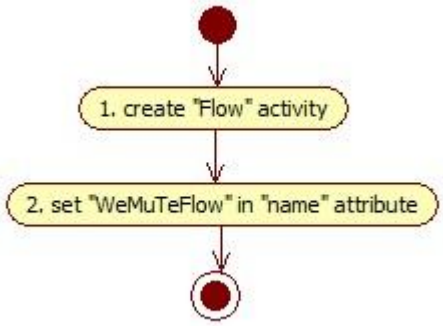


ภาพที่ 3.133 แผนภาพกิจกรรม performGenerateMutantWithASF

แผนภาพกิจกรรม createFlowActivityFromSequence มีขั้นตอนดังต่อไปนี้

1. สร้างอีลีเมนต์ flow ขึ้นมา

2. กำหนดค่าแอตทริบิวต์ name ด้วย WeMuTeFlow



ภาพที่ 3.134 แผนภาพกิจกรรม createFlowActivityFromSequenceActivity

2.19) แผนภาพกิจกรรมของ performGenerateMutantWithASI ในภาพที่ 3.135 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม sequence

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ asiList

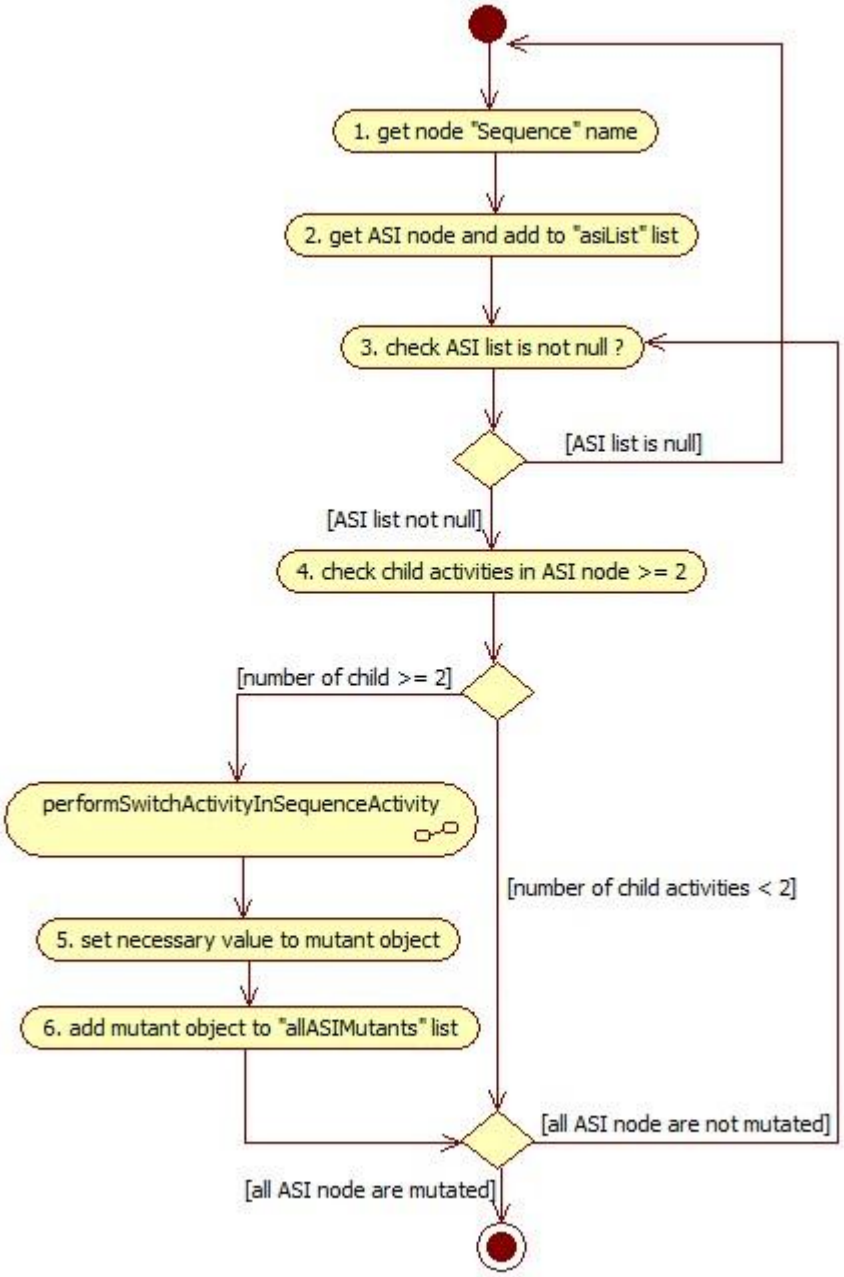
3. ตรวจสอบว่า asiList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า asiList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะตรวจสอบว่าข้อมูลใน asiList มาจำนวนมากกว่าหรือเท่ากับ 2 หรือไม่

หมายเหตุ: หลังจากตรวจสอบรายการ asiList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ performSwitchActivityInSequenceActivity ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.136

5. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

6. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allASIMutants



ภาพที่ 3.135 แผนภาพกิจกรรม performGenerateMutantWithASI

แผนภาพกิจกรรม performSwitchActivityInSequenceActivity ดังภาพที่ 3.136 มีขั้นตอนดังต่อไปนี้

1. วนซ้ำในแต่ละโหนดลูกของรายการ asiList

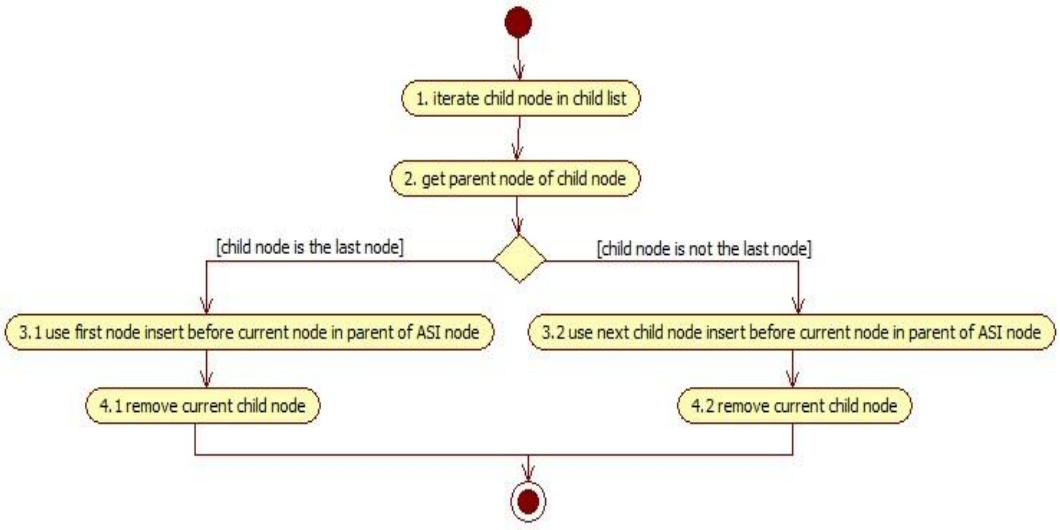
2. นำข้อมูลของโหนดพ่อแม่ออกมาจากโหนดลูก

3. โปรแกรมตรวจสอบลำดับของโหนดลูกภายในโหนดพ่อแม่ ดังต่อไปนี้

   3.1 ถ้าโหนดลูกเป็นโหนดสุดท้ายภายในโหนดพ่อแม่ โปรแกรมจะนำโหนดล่าสุดไปใส่ก่อนหน้าโหนดแรกสุด

   3.2 ถ้าโหนดลูกไม่ใช่โหนดสุดท้ายภายในโหนดพ่อแม่ โปรแกรมจะนำโหนดถัดไปมาใส่ก่อนหน้าโหนดล่าสุด

4. หลังจากสลับที่ลำดับของโหนดเรียบร้อยแล้ว โปรแกรมจะลบโหนดล่าสุดออกไป



ภาพที่ 3.136 แผนภาพกิจกรรม performSwitchActivityInSequenceActivity

2.20) แผนภาพกิจกรรมของ performGenerateMutantWithAWR ในภาพที่ 3.137 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม while

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ awrList

3. สร้างมิวแตนท์อ็อปเจ็กต์ตั้งต้นสำหรับตัวดำเนินการ AWR ขึ้นมา

4. นำมิวแตนท์อ็อปเจ็กต์ตั้งต้นไปเก็บไว้ในรายการ allAWRMutants

5. ตรวจสอบว่า awrList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

6. ดึงข้อมูลโหนดลูกทั้งหมดและโหนด condition ออกมาจากกิจกรรม while

หมายเหตุ: หลังจากตรวจสอบรายการ awrList ว่ามีข้อมูลแล้ว โปรแกรมจะดำเนินการ createRepeatUntilActivityToReplaceWhileActivity ซึ่งขั้นตอนการทำงานถูกอธิบายในแผนภาพกิจกรรม ภาพที่ 3.138

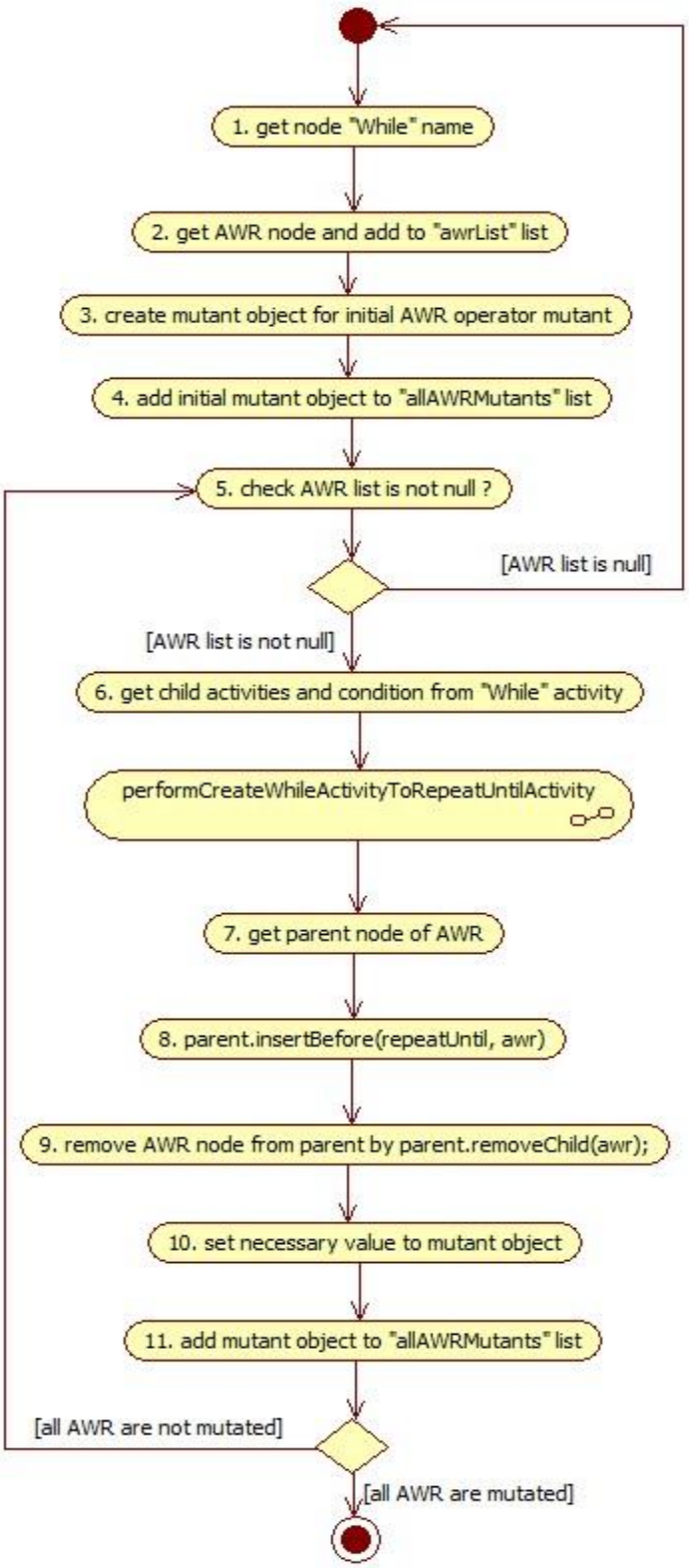
7. นำข้อมูลโหนดพ่อแม่ของกิจกรรม while ออกมา

8. นำกิจกรรม repeatUntil มาใส่ก่อนหน้ากิจกรรม while

9. จากนั้นลบโหนดกิจกรรม while ออกไป

10. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

11. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allAWRMutants



ภาพที่ 3.137 แผนภาพกิจกรรม performGenerateMutantWithAWR

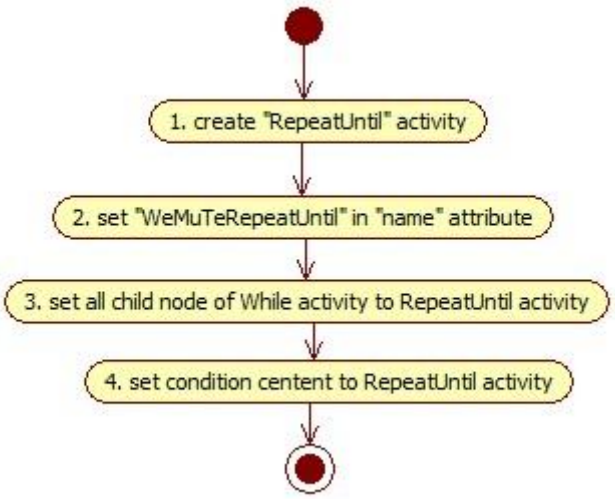
แผนภาพกิจกรรม createRepeatUntilActivityToReplaceWhileActivity ดังภาพที่ 3.138 มีขั้นตอนดังต่อไปนี้

1. สร้างกิจกรรม repeatUntil ขึ้นมา

2. กำหนดชื่อ “WeMuTeRepeatUntil” ในแอตทริบิวต์ name

2. นำข้อมูลของโหนดลูกในกิจกรรม while ทั้งหมด ใส่เข้าไปในกิจกรรม repeatUntil

4. จากนั้นนำข้อมูลของโหนด condition ใส่ให้กับกิจกรรม repeatUntil



ภาพที่ 3.138 แผนภาพกิจกรรม createRepeatUntilActivityToReplaceWhileActivity

2.21) แผนภาพกิจกรรมของ performGenerateMutantWithXEE ในภาพที่ 3.139 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม onEvent

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xeeList

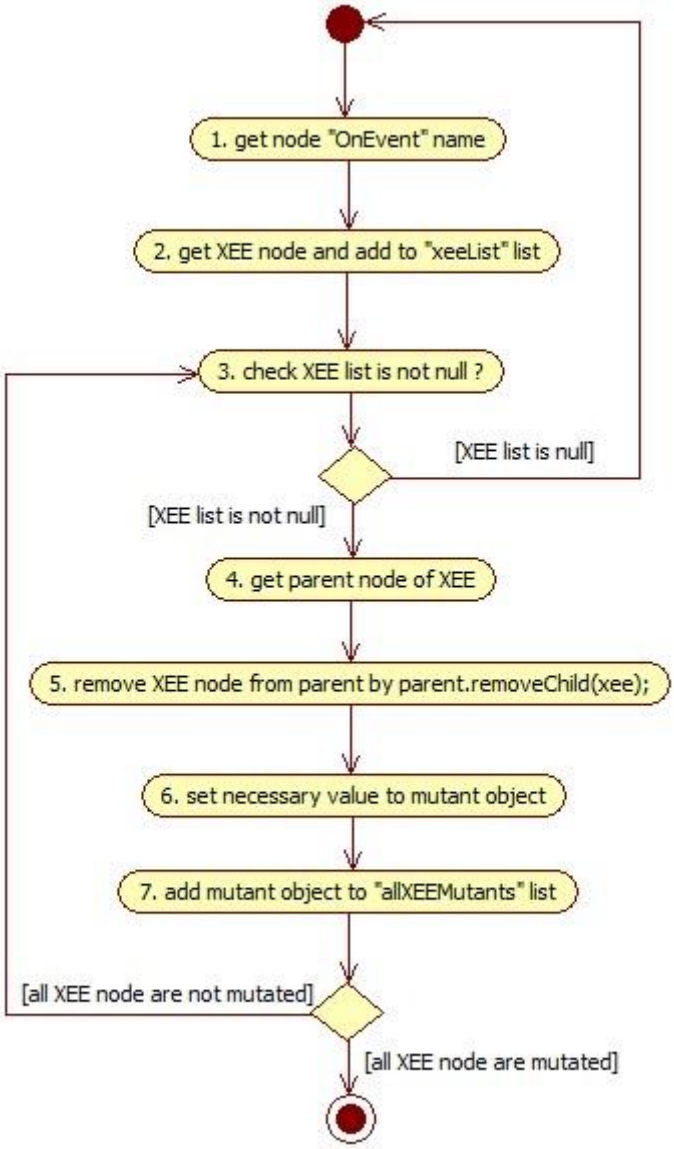
3. ตรวจสอบว่า xeeList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า xeeList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออก (onEvent) โดยลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXEEMutants



ภาพที่ 3.139 แผนภาพกิจกรรม performGenerateMutantWithXEE

2.22) แผนภาพกิจกรรมของ performGenerateMutantWithXER ในภาพที่ 3.140 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม reThrow

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xerList

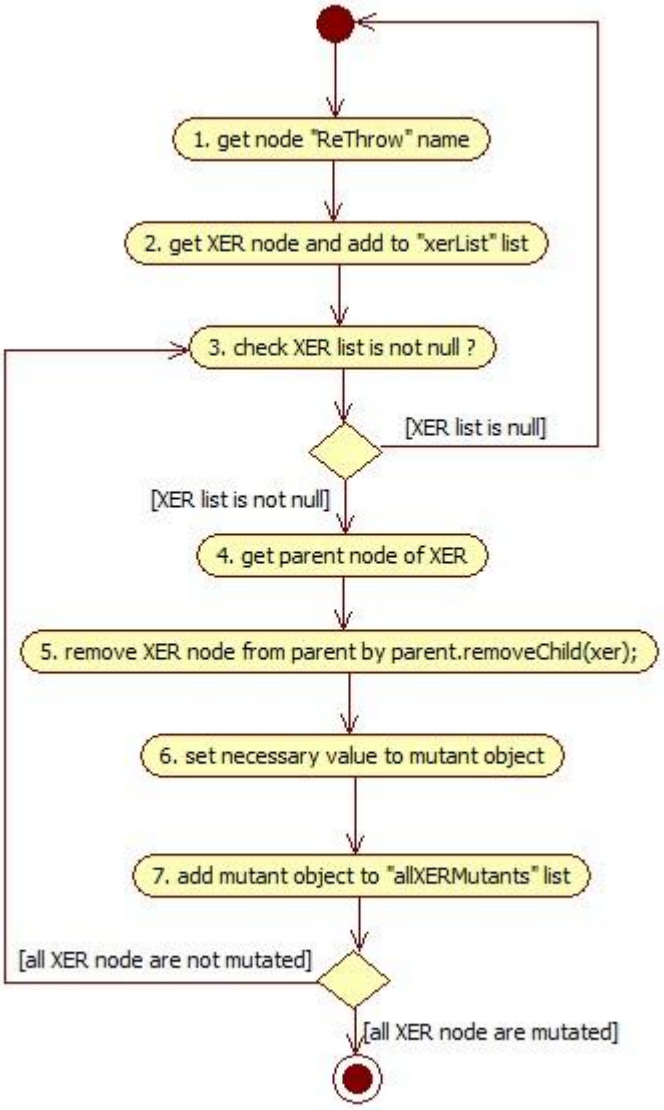
3. ตรวจสอบว่า xerList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า xerList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออก (reThrow) โดยลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXERMutants



ภาพที่ 3.140 แผนภาพกิจกรรม performGenerateMutantWithXER

2.23) แผนภาพกิจกรรมของ performGenerateMutantWithXMC ในภาพที่ 3.141 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม compensationHandler

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xmcList

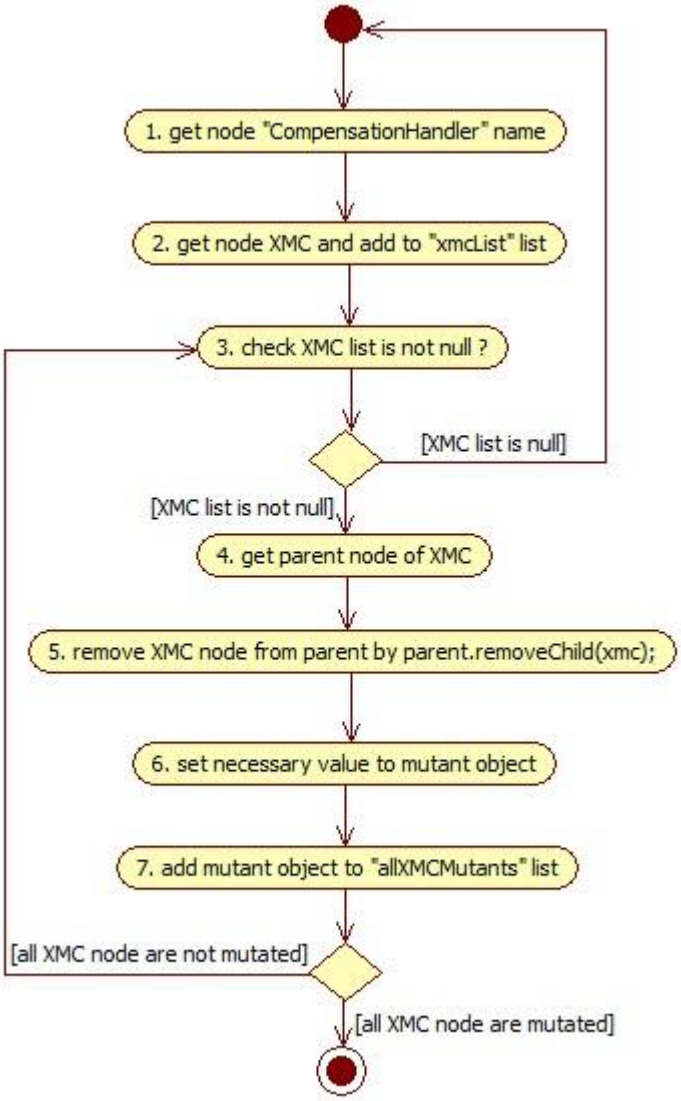
3. ตรวจสอบว่า xmcList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า xmcList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออก (compensationHandler) โดยลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXMCMutants



ภาพที่ 3.141 แผนภาพกิจกรรม performGenerateMutantWithXMC

2.24) แผนภาพกิจกรรมของ performGenerateMutantWithXMF ในภาพที่ 3.142 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม catch และ catchAll

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xmfList

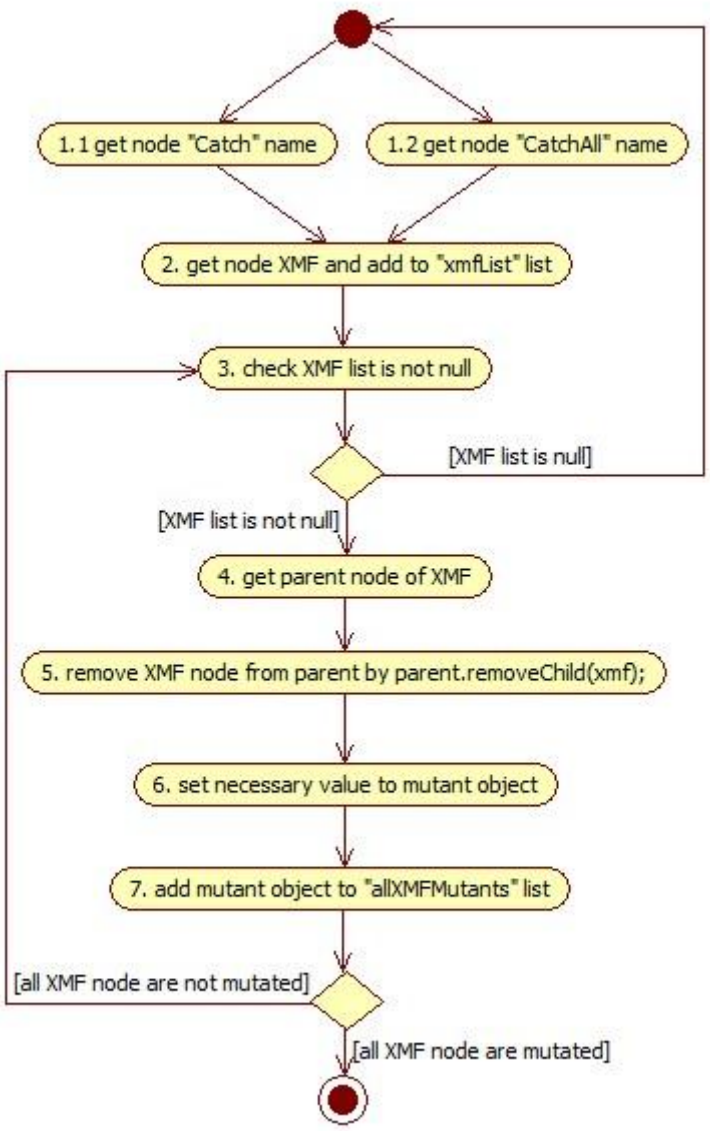
3. ตรวจสอบว่า xmfList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า xmfList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออก (catch หรือ catchAll) โดยลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXMFMutants



ภาพที่ 3.142 แผนภาพกิจกรรม performGenerateMutantWithXMF

2.25) แผนภาพกิจกรรมของ performGenerateMutantWithXMT ในภาพที่ 3.143 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม terminationHandler

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xmtList

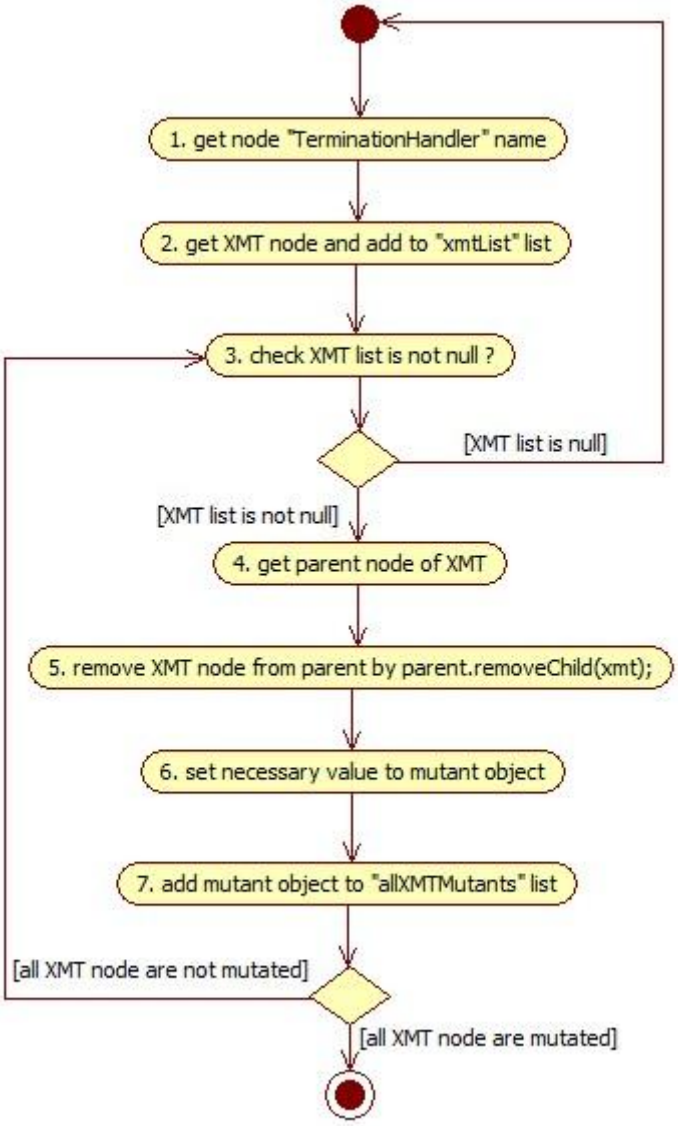
3. ตรวจสอบว่า xmtList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

4. ถ้า xmtList มีข้อมูลของโหนดดังข้อที่ 1 โปรแกรมจะนำข้อมูลของพ่อแม่ของโหนดดังกล่าวในข้อที่ 1

5. ลบโหนดที่ต้องการออก (terminationHandler) โดยลบโหนดลูกออกจากโหนดพ่อแม่

6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXMTMutants



ภาพที่ 3.143 แผนภาพกิจกรรม performGenerateMutantWithXMT

2.26) แผนภาพกิจกรรมของ performGenerateMutantWithXTF ในภาพที่ 3.144 มีขั้นตอนในการดัดแปลงกระบวนการบีเพล ดังนี้

1. โปรแกรมค้นหาโหนดในกระบวนการบีเพลโดยเลือกโหนดของกิจกรรม throw

2. นำโหนดของกิจกรรมดังกล่าวในข้อที่ 1 เก็บใส่ไว้ในรายการ xtfList

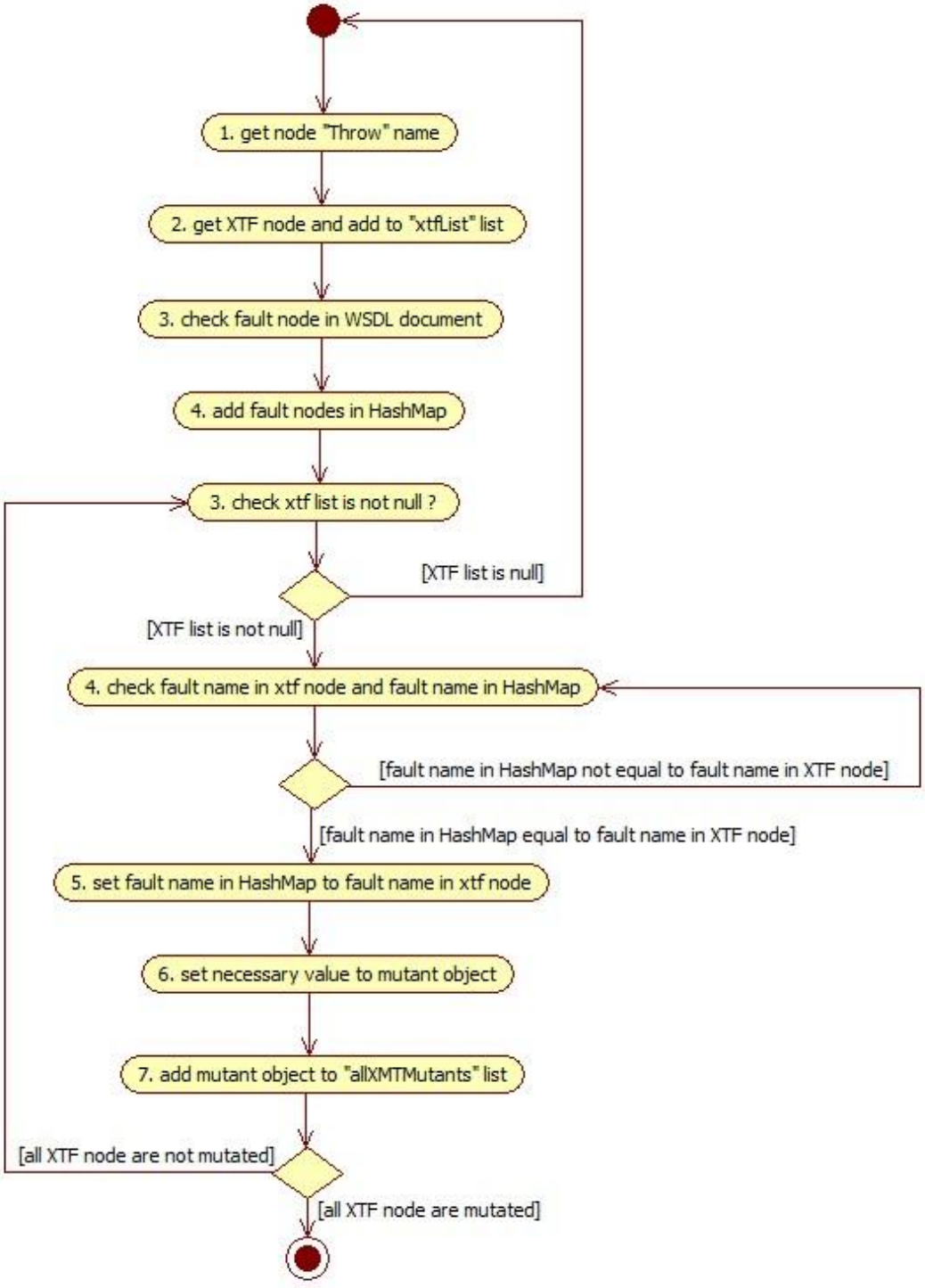
3. ตรวจสอบในเอกสารวิซเดิลในโหนด operation ว่ามีโหนด fault ที่เป็นโหนดลูกหรือไม่ ถ้ามีเก็บชื่อของโหนด fault เก็บไว้ใน HashMap อ็อปเจ็กต์

4. ตรวจสอบว่า xtfList มีข้อมูลของโหนดดังข้อที่ 1 หรือไม่ ถ้าไม่มีข้อมูลก็กลับไปเริ่มต้นใหม่

5. ทำการวนซ้ำใน HashMap อ็อปเจ็กต์พร้อมทั้งเปรียบเทียบชื่อของโหนด fault ในเอกสารวิซเดิลและในโหนด xtf ถ้าชื่อไม่ซ้ำกัน โปรแกรมจะนำชื่อนั้นใส่กลับไปที่โหนด xtf

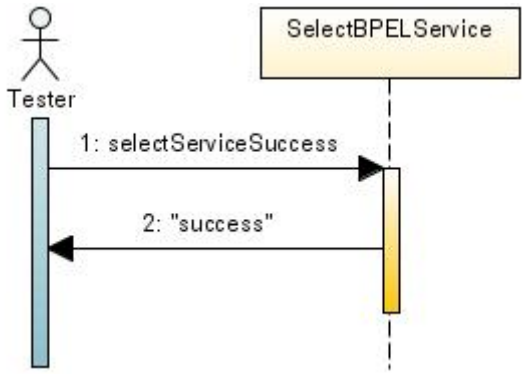
6. เก็บข้อมูลที่จำเป็นไว้ในมิวแตนท์อ็อปเจ็กต์ เช่น ชื่อของมิวแตนท์ หรือมิวแตนท์ถูกกำจัดหรือไม่ เป็นต้น

7. นำมิวแตนท์อ็อปเจ็กต์เก็บไว้ในรายการ allXTFMutants



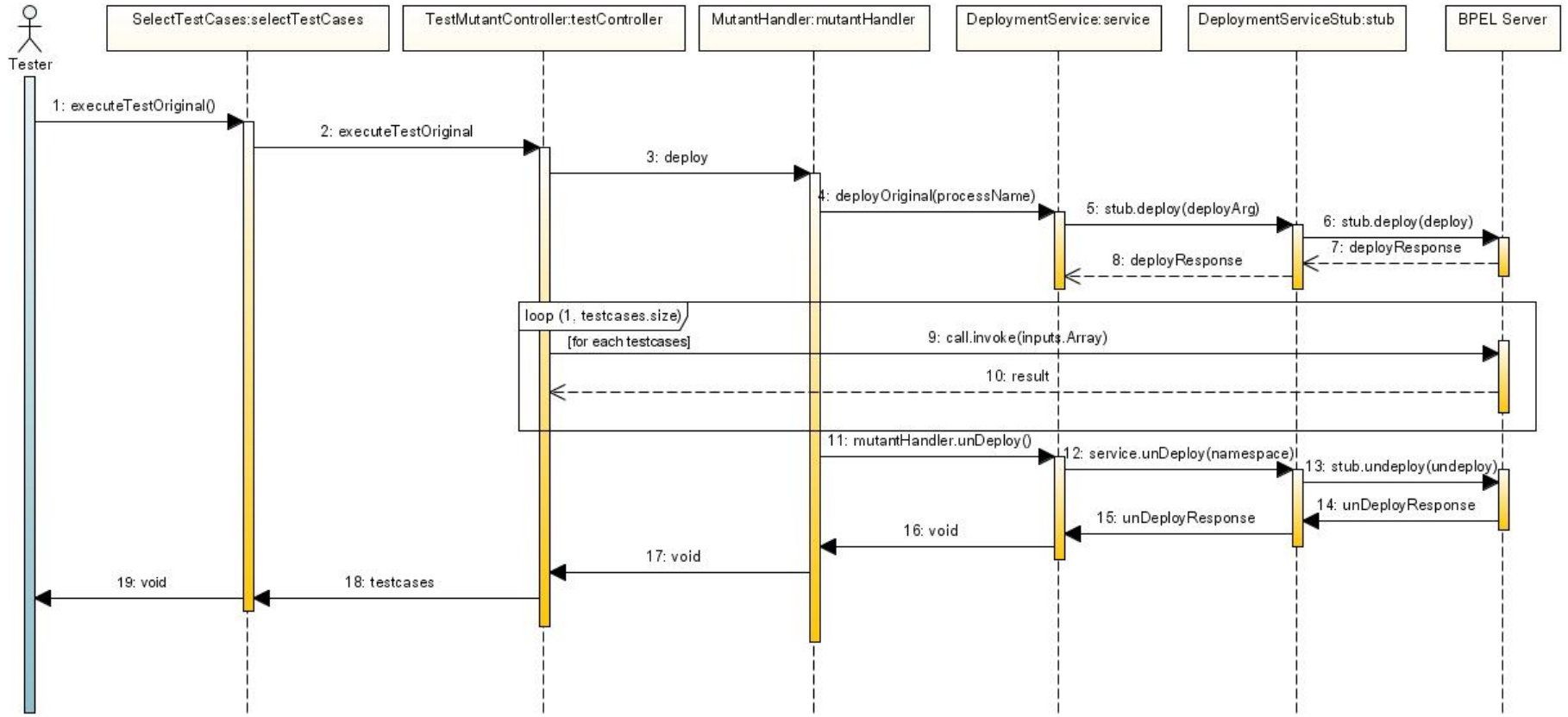
ภาพที่ 3.144 แผนภาพกิจกรรม performGenerateMutantWithXTF

3) แผนภาพลำดับ Select BPEL Service เป็นการเลือกเรียกใช้เซอร์วิซที่แสดงหน้าที่จอ จากนั้นนักทดสอบกดปุ่มเพื่อยอมรับแล้วจะเข้าสู่ขั้นตอน Test BPEL Original ดังภาพที่ 3.145



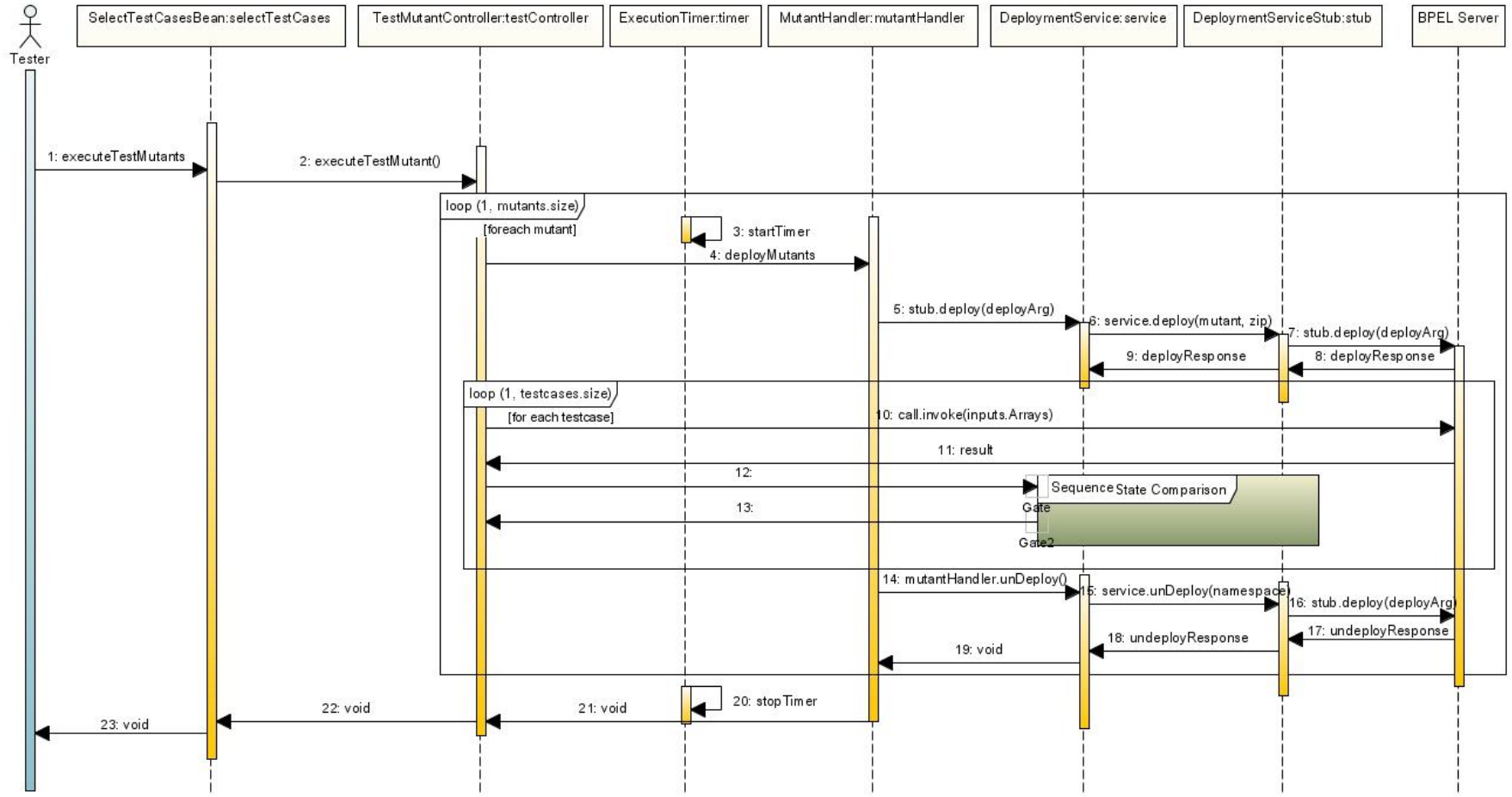
ภาพที่ 3.145 แผนภาพลำดับ Select BPEL Service

4) แผนภาพลำดับ Test BPEL Original หลังจากที่นักทดสอบโหลดข้อมูลของกรณีทดสอบจึงทำการทดสอบโปรแกรมต้นแบบ โดยอ็อบเจ็กต์ MutantHandler จะนำโปรแกรมต้นแบบที่เป็นเอกสารซิปโดยส่งไปที่อ็อบเจ็กต์ DeploymentService ก่อนการดีพลอยและ DeploymentServiceStub ตามลำดับ เพื่อติดโปรแกรมไปที่ BPEL Server จากนั้นอ็อบเจ็กต์ TestMutantController จะส่งข้อมูลกรณีทดสอบไปที่ BPEL Server เพื่อทดสอบโปรแกรมต้นแบบเมื่อวนซ้ำจนหมดกรณีทดสอบแล้ว อ็อบเจ็ก MutantHandler จะยกเลิกการติดตั้งโปรแกรมจาก BPEL Server ดังภาพที่ 3.146



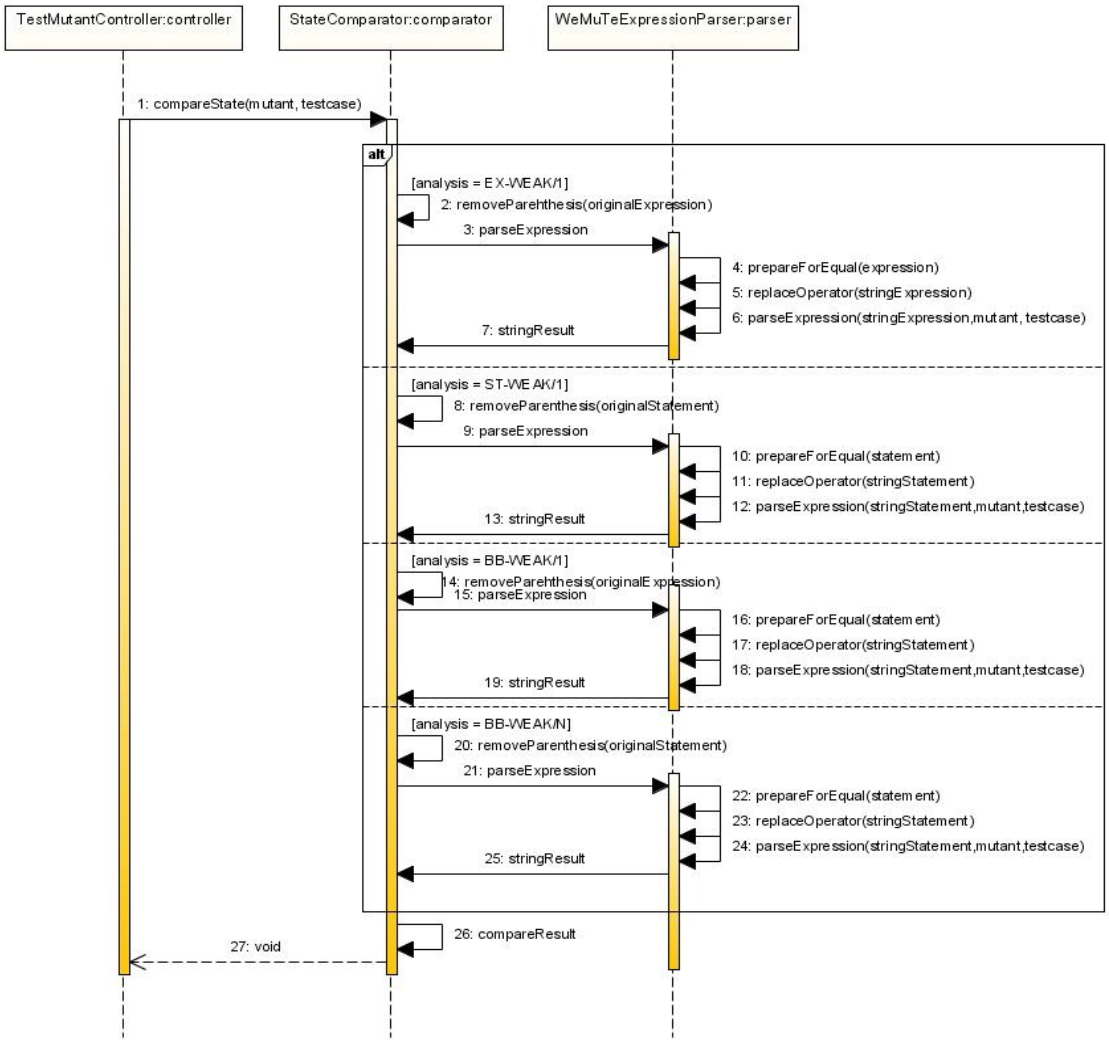
ภาพที่ 3.146 แผนภาพลำดับ Test BPEL Original

5) แผนภาพลำดับ Test BPEL Mutants ดังภาพที่ 3.147 เริ่มเมื่อนักทดสอบกดปุ่มเพื่อทดสอบมิวแตนท์ จากนั้นอ็อบเจ็กต์ ExecutionTimer จะเริ่มจับเวลาและ MutantHandler จะนำเอกสารซิปของมิวแตนท์ไปติดตั้งที่ BPELServer โดยผ่านอ็อบเจ็กต์ DeploymentService และ DeploymentServiceStub ตามลำดับ หลังจากที่ติดตั้งมิวแตนท์เรียบร้อย TestMutantController ส่งข้อมูลกรณีทดสอบเพื่อทดสอบมิวแตนท์ เมื่อผลลัพธ์ถูกส่งกลับมาจาก BPEL Server มาที่ TestMutantController จากนั้นผลลัพธ์ก็จะถูกนำไปเปรียบเทียบที่แผนภาพลำดับ State Comparison แล้วจึงยกเลิกการติดตั้งมิวแตนท์ โดยจะทำแบบนี้เรื่อยไปจนครบมิวแตนท์ทุกตัว



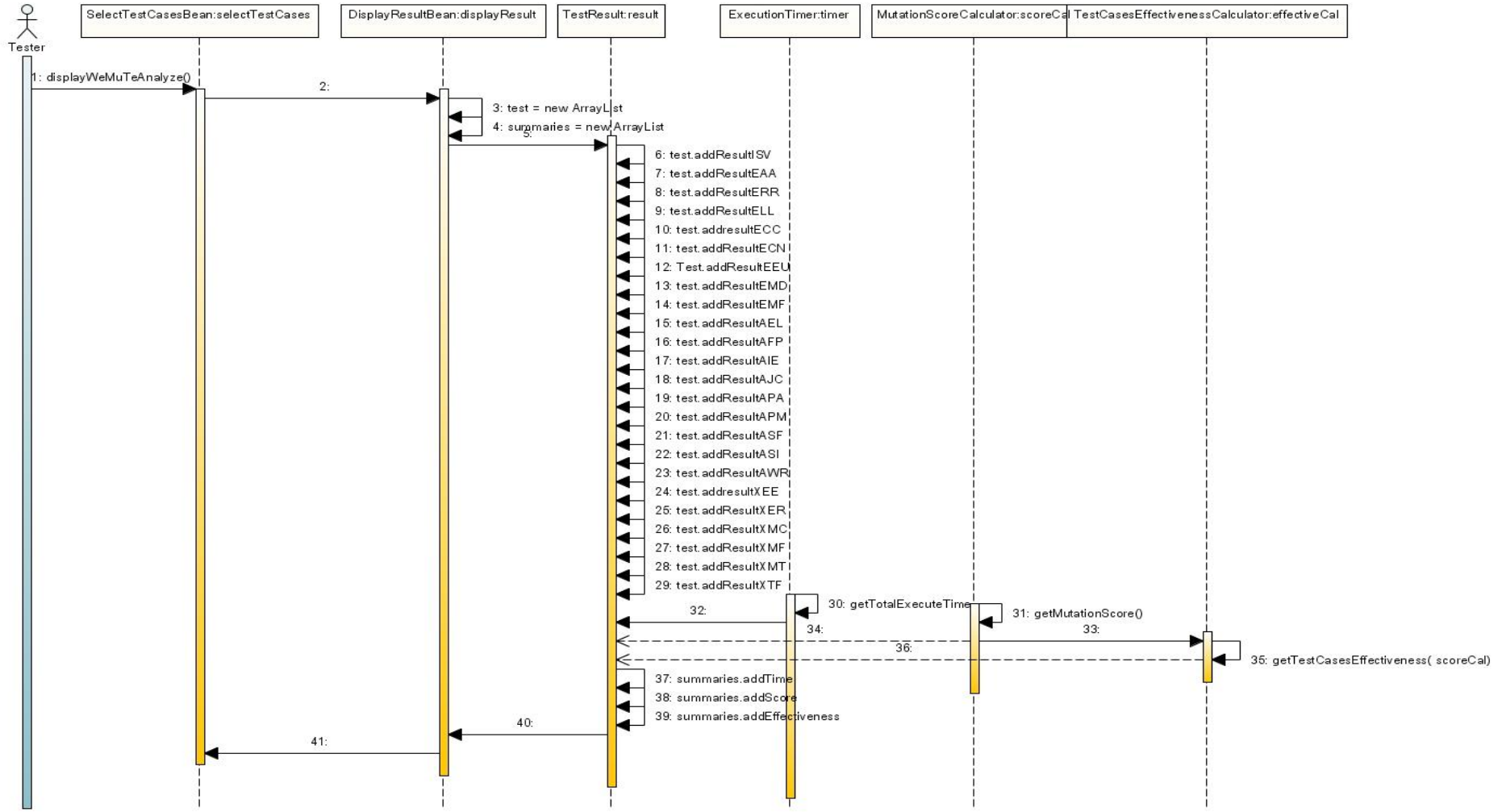
ภาพที่ 3.147 แผนภาพลำดับ Test BPEL Mutants

6) แผนภาพลำดับ State Comparison เริ่มเมื่ออ็อบเจ็กต์ TestMutantController ส่งผลลัพธ์มาที่ StateComparator จากนั้นจะพิจารณาแนววิธีการทดสอบวีคมิวเทชัน แล้วจะทำการแทนที่ตัวดำเนินการและข้อมูลของกรณีทดสอบโดยอ็อบเจ็กต์ WeMuTeExpressionParser จากนั้นจึงส่งผลลัพธ์กลับไปที่ TestMutantController ดังภาพที่ 3.148



ภาพที่ 3.148 แผนภาพลำดับ State Comparison

7) แผนภาพลำดับ Display WeMuTe Result ดังภาพที่ 3.149 จะเตรียมข้อมูลก่อนการแสดงผลลัพธ์ โดยอ็อบเจ็กต์ DisplayResultBean สร้างอ็อบเจ็ก TestResult เพื่อเก็บข้อมูลมิวแตนท์ที่เกิดจากการทดสอบทุกตัว จากนั้นจึงคำนวนคะแนนมิวเทชันกับประสิทธิภาพของกรณีทดสอบ ด้วย MutationScoreCalculator และ TestCasesEffectivenessCalculator ตามลำดับ



ภาพที่ 3.102 แผนภาพลำดับ Display WeMuTe Result

3.3.5 รูปแบบการรายงานผลของการทดสอบ

หลังจากโปรแกรมได้ทำการทดสอบด้วยวีคมิวเทชันเรียบร้อยแล้ว เครื่องมือทดสอบโปรแกรมบีเพลจะรายงานผลของการทดสอบ ดังตารางที่ 3.18

ตารางที่ 3.18 รูปแบบการรายงานผลการทดสอบ

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	*มิวแตนท์ของ ISV*	*มิวแตนท์ที่ถูกกำจัดของ ISV*
EAA	*มิวแตนท์ของ EAA*	*มิวแตนท์ที่ถูกกำจัดของ EAA*
ECC	*มิวแตนท์ของ ECC*	*มิวแตนท์ที่ถูกกำจัดของ ECC*
ECN	*มิวแตนท์ของ ECN*	*มิวแตนท์ที่ถูกกำจัดของ ECN*
ERR	*มิวแตนท์ของ ERR*	*มิวแตนท์ที่ถูกกำจัดของ ERR*
EMD	*มิวแตนท์ของ EMD*	*มิวแตนท์ที่ถูกกำจัดของ EMD*
EMF	*มิวแตนท์ของ EMF*	*มิวแตนท์ที่ถูกกำจัดของ EMF*
ELL	*มิวแตนท์ของ ELL*	*มิวแตนท์ที่ถูกกำจัดของ ELL*
EEU	*มิวแตนท์ของ EEU*	*มิวแตนท์ที่ถูกกำจัดของ EEU*
ACI	*มิวแตนท์ของ ACI*	*มิวแตนท์ที่ถูกกำจัดของ ACI*
AEL	*มิวแตนท์ของ AEL*	*มิวแตนท์ที่ถูกกำจัดของ AEL*
AFP	*มิวแตนท์ของ AFP*	*มิวแตนท์ที่ถูกกำจัดของ AFP*
AIE	*มิวแตนท์ของ AIE*	*มิวแตนท์ที่ถูกกำจัดของ AIE*
AIS	*มิวแตนท์ของ AIS*	*มิวแตนท์ที่ถูกกำจัดของ AIS*
AJC	*มิวแตนท์ของ AJC*	*มิวแตนท์ที่ถูกกำจัดของ AJC*
APA	*มิวแตนท์ของ APA*	*มิวแตนท์ที่ถูกกำจัดของ APA*
APM	*มิวแตนท์ของ APM*	*มิวแตนท์ที่ถูกกำจัดของ APM*
ASF	*มิวแตนท์ของ ASF*	*มิวแตนท์ที่ถูกกำจัดของ ASF*
ASI	*มิวแตนท์ของ ASI*	*มิวแตนท์ที่ถูกกำจัดของ ASI*
AWR	*มิวแตนท์ของ AWR*	*มิวแตนท์ที่ถูกกำจัดของ AWR*

ตารางที่ 3.18 รูปแบบการรายงานผลการทดสอบ (ต่อ)

XEE	*มิวแตนท์ของ XEE*	*มิวแตนท์ที่ถูกกำจัดของ XEE*
XER	*มิวแตนท์ของ XER*	*มิวแตนท์ที่ถูกกำจัดของ XER*
XMC	*มิวแตนท์ของ XMC*	*มิวแตนท์ที่ถูกกำจัดของ XMC*
XMF	*มิวแตนท์ของ XMF*	*มิวแตนท์ที่ถูกกำจัดของ XMF*
XMT	*มิวแตนท์ของ XMT*	*มิวแตนท์ที่ถูกกำจัดของ XMT*
XTF	*มิวแตนท์ของ XTF*	*มิวแตนท์ที่ถูกกำจัดของ XTF*

สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	*มิวแตนท์ที่สร้างทั้งหมด*	*มิวแตนท์ที่ถูกกำจัด*
เวลาที่ใช้ในการทดสอบ	*เวลาที่ใช้ (วินาที)*	*เวลาที่ใช้ (นาที)*
คะแนนมิวเทชัน	*คะแนนมิวเทชัน*	
ประสิทธิภาพของกรณีทดสอบ	*ประสิทธิภาพของกรณีทดสอบ*	

ส่วนประกอบของการรายงานผลการทดสอบของเครื่องมือทดสอบแบบวีคมิวเทชัน จะประกอบด้วยข้อมูลดังต่อไปนี้

1) ตัวดำเนินการมิวเทชันแต่ละตัวคือ ISV EAA ECC ECN EEU ELL EMD EMF ERR AEL AFP ACI AIE AIS AJC APA APM ASF ASI AWR XEE XER XMC XMF XMT และ XTF

2) จำนวนมิวแตนท์ที่สร้างได้ในแต่ละตัวดำเนินการมิวเทชัน

3) จำนวนมิวแตนท์ที่ถูกกำจัดในแต่ละตัวดำเนินการมิวเทชัน

4) จำนวนมิวแตนท์ที่สร้างขึ้นทั้งหมดและมิวแตนท์ที่ถูกกำจัดทั้งหมด

5) เวลาที่ใช้ในการทดสอบทั้งสิ้นในหน่วยวินาทีและนาที

6) คะแนนมิวเทชันสามารถหาได้จากอัตราส่วนระหว่างจำนวนของมิวแตนท์ที่ถูกกำจัด (D) ต่อด้วยจำนวนมิวแตนท์ที่ไม่สมมูล (จำนวนมิวแตนท์ที่เกิดขึ้นทั้งหมด (T) ลบด้วยจำนวนมิวแตนท์สมมูลที่เกิดขึ้น (E)) ดังสมการ

$$M = \frac{D}{T - E}$$

7) ประสิทธิภาพของกรณีทดสอบ (E) สามารถหาได้จากอัตราส่วนของค่าเฉลี่ยของจำนวนกรณีทดสอบที่กำจัดมิวแตนท์ได้ ($\overline{K}_D$) ต่อจำนวนกรณีทดสอบทั้งหมด (T) และคูณด้วยคะแนนมิวเทชัน (M) ค่าเฉลี่ยของจำนวนกรณีทดสอบที่กำจัดมิวแตนท์ได้ หาได้จากอัตราส่วนของผลรวมของกรณีทดสอบที่กำจัดมิวแตนท์ได้ (K_M) หารด้วยมิวแตนท์ที่ถูกกำจัด (D) ดังสมการ

$$\mathrm{E} = M \times \frac{\overline{K}_D}{T}$$

โดยที่ $$\overline{K}_D = \frac{\sum K_M}{D}$$

บทที่ 4
การพัฒนาเครื่องมือ

บทนี้เป็นการนำเสนอการพัฒนาเครื่องมือทดสอบแบบวีคมิวเทชัน สภาพแวดล้อมที่ใช้ในการพัฒนา คอมโพเนนท์ของส่วนต่อประสานระหว่างนักทดสอบและระบบ

4.1 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ

1) ฮาร์ดแวร์ (Hardware)

1.1) เครื่องคอมพิวเตอร์โน๊ตบุค (Notebook) หน่วยประมวลผลอินเทลคอร์ทูดูโอ พี8600 2.4 กิกะเฮิร์ซ์ (Intel® Core™ 2 Duo CPU P8600 @ 2.40GHz 2.40GHz)

1.2) หน่วยความจำสำรอง (RAM) 8.00 กิกะไบต์ (8.00 GB)

2) ซอฟต์แวร์ (Software)

2.1) ระบบปฏิบัติการ (Operating System) ไมโครซอฟท์วินโดวส์ 7 อัลติเมท เซอร์วิซแพค 1 (Windows 7 Ultimate Service Pack 1)

2.2) เครื่องมือที่ใช้พัฒนา อีคลิป ไอดีอี รุ่นสำหรับนักพัฒนาภาษาจาวาระดับองค์กร 3.7.1 (Eclipse IDE for Java EE Developers 3.7.1)

2.3) ภาษาที่ใช้พัฒนา คือ ภาษาจาวา (Java) เวอร์ชัน 6 ขึ้นไป

2.4) เฟรมเวิร์กที่ใช้พัฒนา คือ จาวาเซิร์ฟเวอร์เฟส (Java Server Faces)

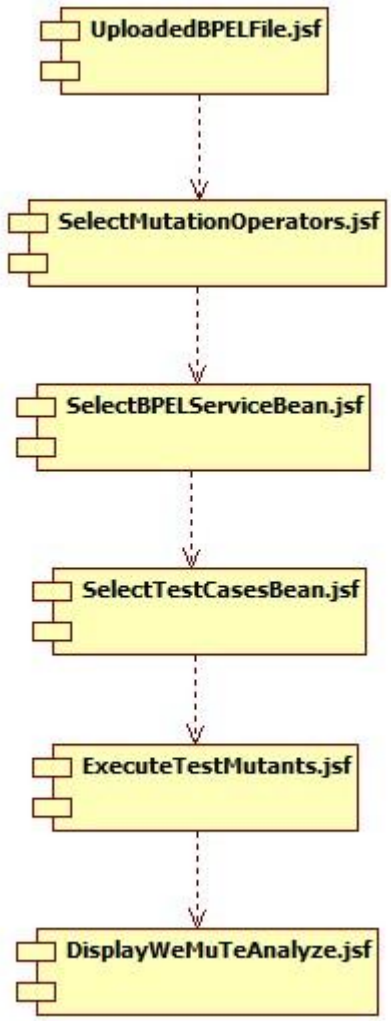
2.5) เครื่องบริการเว็บ (Web Server) อาพาเช่ทอมแคท 7.0.xx (Apache Tomcat 7.0.xx)

2.6) เครื่องประมวลผลบีเพล (BPEL Server) อาพาเช่โอดีอี 1.3.5 (Apache ODE 1.3.5) ซึ่งถูกติดตั้งบน อาพาเช่ทอมแคท 7.0.xx

2.7) เว็บเบราว์เซอร์ (Web Browser) มอสซิลล่าไฟล์ฟอกซ์ 9.0.1 (Mozilla Firefox 9.0.1) หรือ กูเกิล โครมเบราว์เซอร์ 18 ขึ้นไป (Google Chrome)

4.2 ส่วนต่อประสานกับผู้ใช้ของเครื่องมือทดสอบแบบวีคมิวเทชัน

โครงสร้างของส่วนต่อประสานกับผู้ใช้จะถูกอธิบายด้วยแผนภาพองค์ประกอบ (Component Diagram) ซึ่งได้อธิบายถึงส่วนต่อประสานทั้งหมดในระบบ และความสัมพันธ์ระหว่างส่วนต่อประสานด้วยกัน ซึ่งเริ่มจากส่วนต่อประสาน UploadBPELFile.jsf เป็นส่วนที่ติดต่อกับผู้ใช้งานเป็นอันดับแรก เรียงลำดับไปจนถึงส่วนต่อประสาน DisplayWeMuTeAnalyze.jsf เป็นอันดับสุดท้าย ดังภาพที่ 4.1



ภาพที่ 4.1 แผนภาพองค์ประกอบของเครื่องมือทอสอบวีคมิวเทชัน

ดังภาพที่ 4.1 แผนภาพองค์ประกอบในแต่ละส่วนเป็นเอกสารซึ่งถูกพัฒนามาจากจาวาเซิร์ฟเวอร์เฟส (.jsf) โดยแต่ละ component สามารถอธิบายรายละเอียดได้ดังนี้

4.2.1 หน้าจอ UploadBPELFile.jsf

หน้าจอ UploadBPELFile.jsf ติดต่อกับนักทดสอบเพื่อรับเอกสารซิปที่ประกอบด้วยเอกสารบีเพลที่เป็นไปตามมาตรฐานโอเอซิส เอกสารวิซเดิล เอกสารดีพลอย และเอกสารเอ็กซ์เอ็มแอลสคีมา (สามารถมีหรือไม่มีก็ได้) เมื่อเลือกเอกสารแล้วจึงกดปุ่มอัพโหลด จากนั้นระบบจะทำการนำเอกสารไปไว้ที่เครื่องบริการเว็บ ดังภาพที่ 4.2 ในกรณีที่อัพโหลดเอกสารอื่นที่ไม่ใช่เอกสารซิป ระบบจะแจ้งเตือนตามภาพที่ 4.3

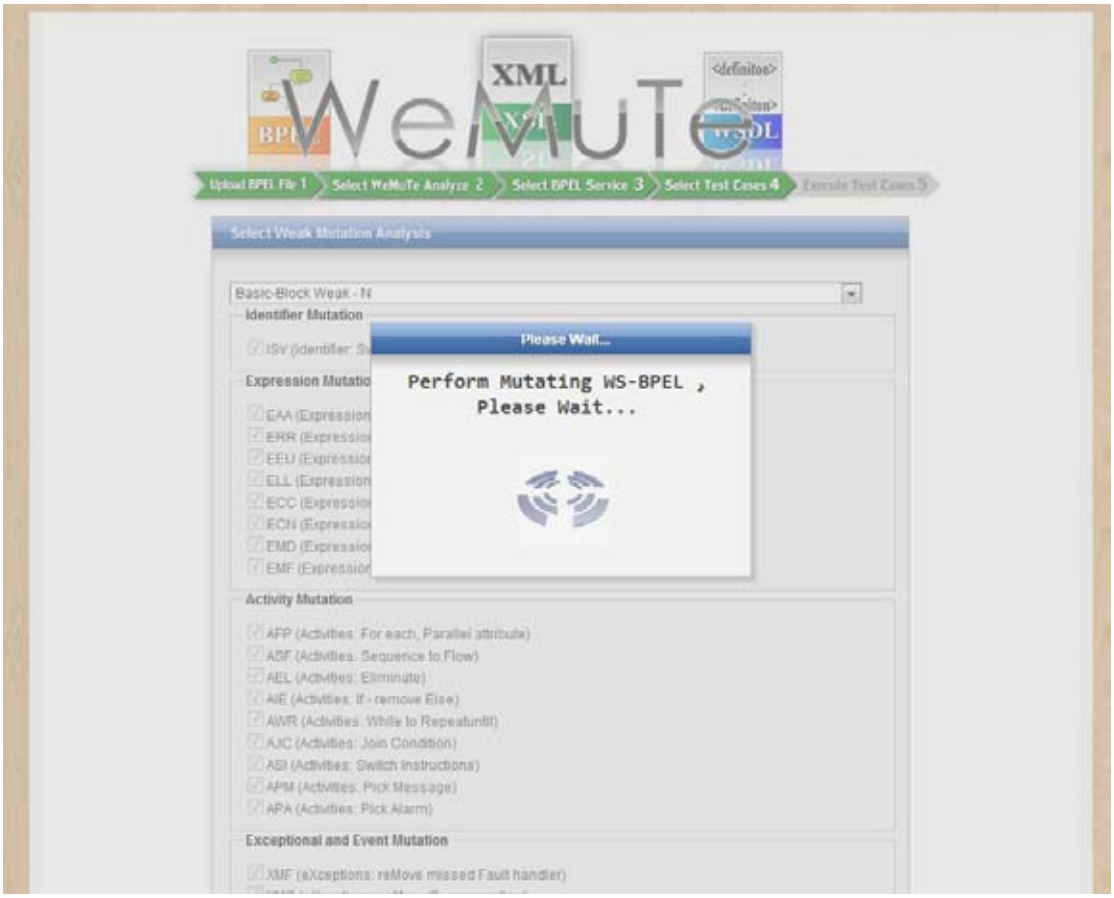
ภาพที่ 4.2 หน้าจอ UploadBPELFile.jsf

ภาพที่ 4.3 หน้าจอ UploadBPELFile.jsf (ใช้เอกสารอื่นนอกจากเอกสารซิป)

4.2.2 หน้าจอ SelectMutationOperators.jsf

หน้าจอ SelectMutationOperators.jsf ในภาพที่ 4.4 เป็นหน้าจอเพื่อให้นักทดสอบเลือกแนววิธีทดสอบแบบวีคมิวเทชัน เมื่อนักทดสอบเลือกแนววิธีทดสอบแล้วระบบก็จะเลือกตัวดำเนินการที่เกี่ยวข้องกับชนิดของวีคมิวเทชันนั้นๆ จากนั้นนักทดสอบกดปุ่ม Generate Mutants ระบบก็จะดำเนินการสร้างมิวแตนท์ ดังภาพที่ 4.5

ภาพที่ 4.4 หน้าจอ SelectMutationOperators.jsf



ภาพที่ 4.5 หน้าจอ SelectMutationOperator.jsf (ระบบสร้างมิวแตนท์)

4.2.3 หน้าจอ SelectBPELService.jsf

หน้าจอ SelectBPELService.jsf ในภาพที่ 4.6 จะแสดงรายละเอียดของโปรแกรมบีเพลที่นักทดสอบต้องการทดสอบ คือ Namespace, Service Operation, Endpoint Address, และ PortType Name

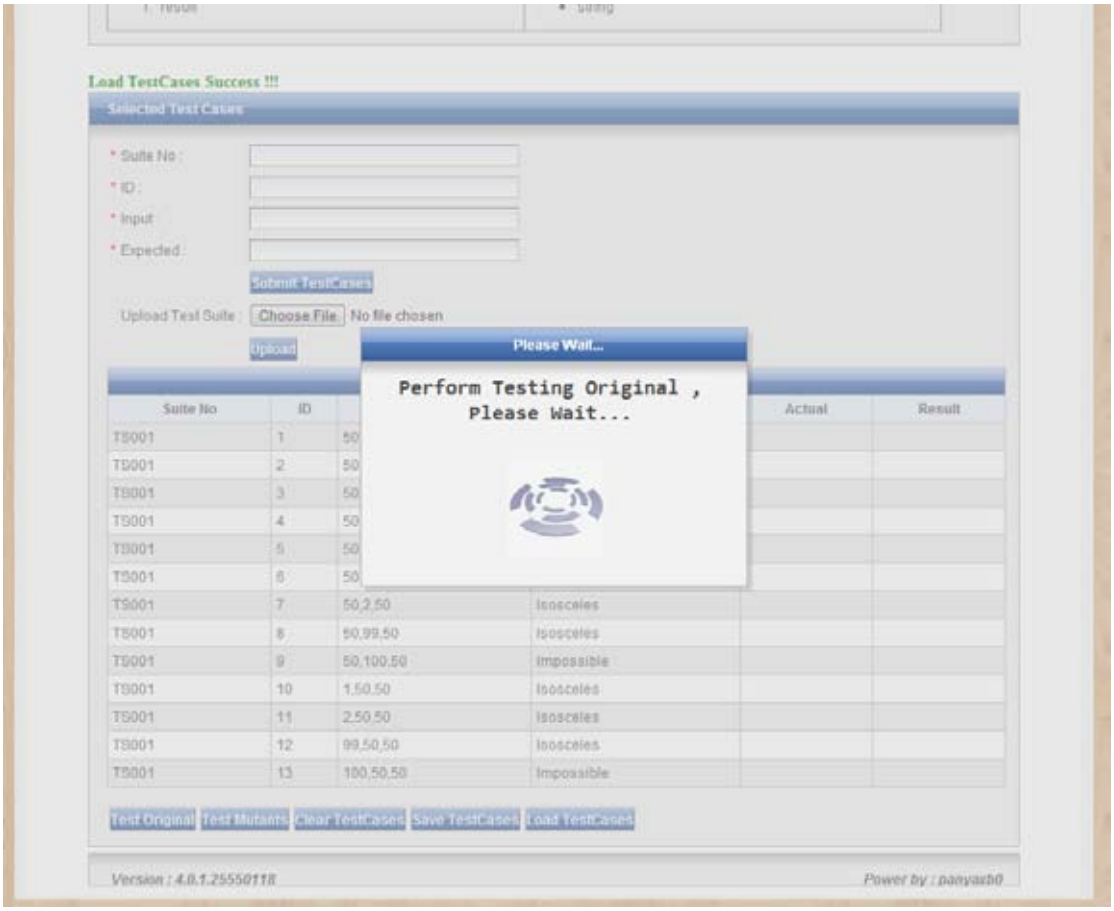
ภาพที่ 4.6 หน้าจอ SelectBPELService.jsf

4.2.4 หน้าจอ SelectTestCases.jsf

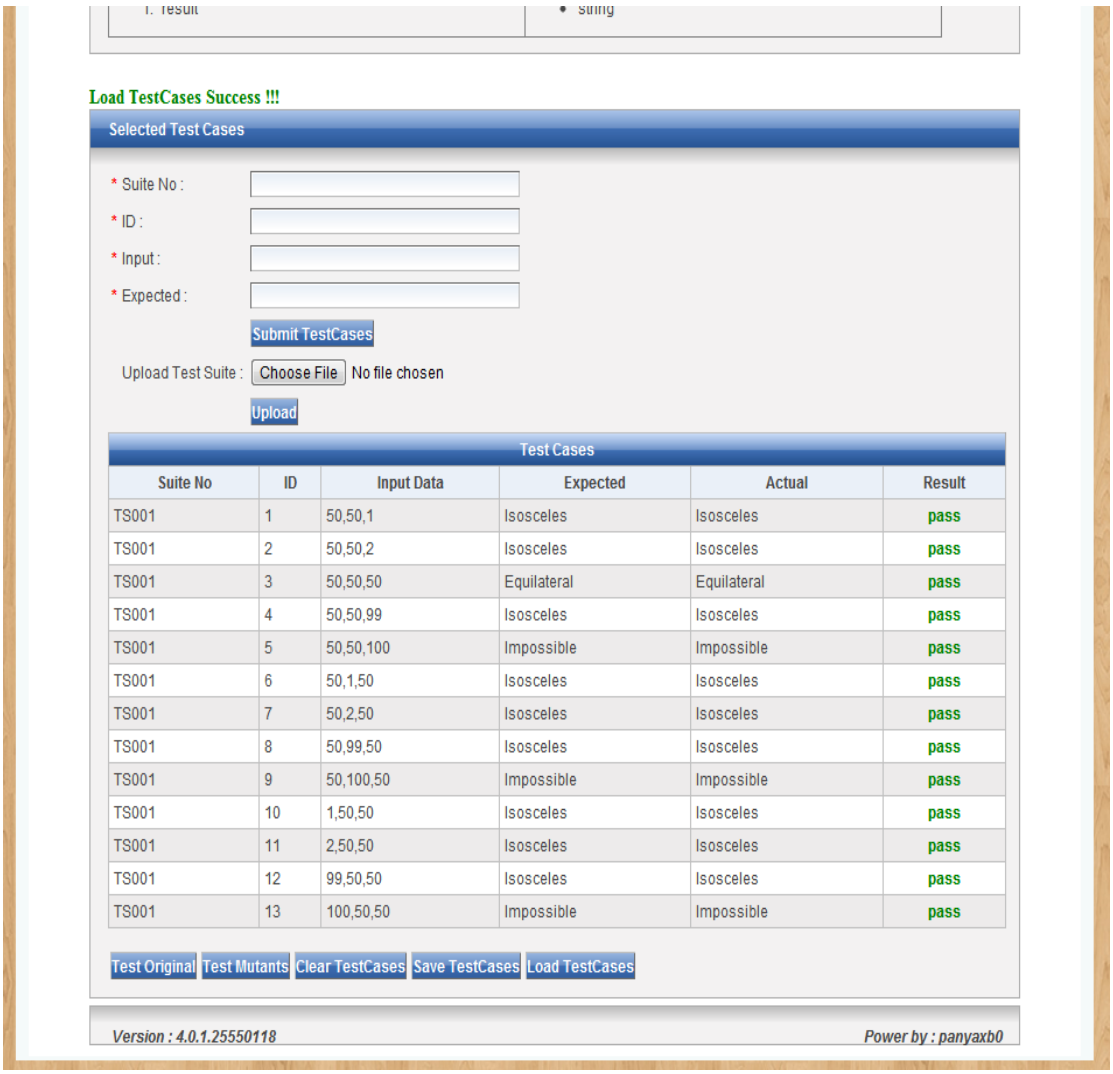
หน้าจอ SelectTestCases.jsf ดังภาพที่ 4.7 จะแสดงหลังจากนักทดสอบโหลดกรณีทดสอบแล้ว ในกรณีที่ไม่พบกรณีทดสอบระบบก็จะแสดงในภาพที่ 4.8 หลังจากนั้นนักทดสอบจะทำการทดสอบโปรแกรมต้นแบบดังในภาพที่ 4.9 และเมื่อระบบได้ทดสอบโปรแกรมเสร็จสิ้นแล้วก็จะแสดงผลดังภาพที่ 4.10 และหลังจากนั้นนักทดสอบจะทำการทดสอบมิวแตนท์ที่สร้างขึ้นมาซึ่งในระหว่างการทดสอบระบบจะแสดงหน้าจอ ดังภาพที่ 4.11

ภาพที่ 4.7 หน้าจอ SelectTestCases.jsf

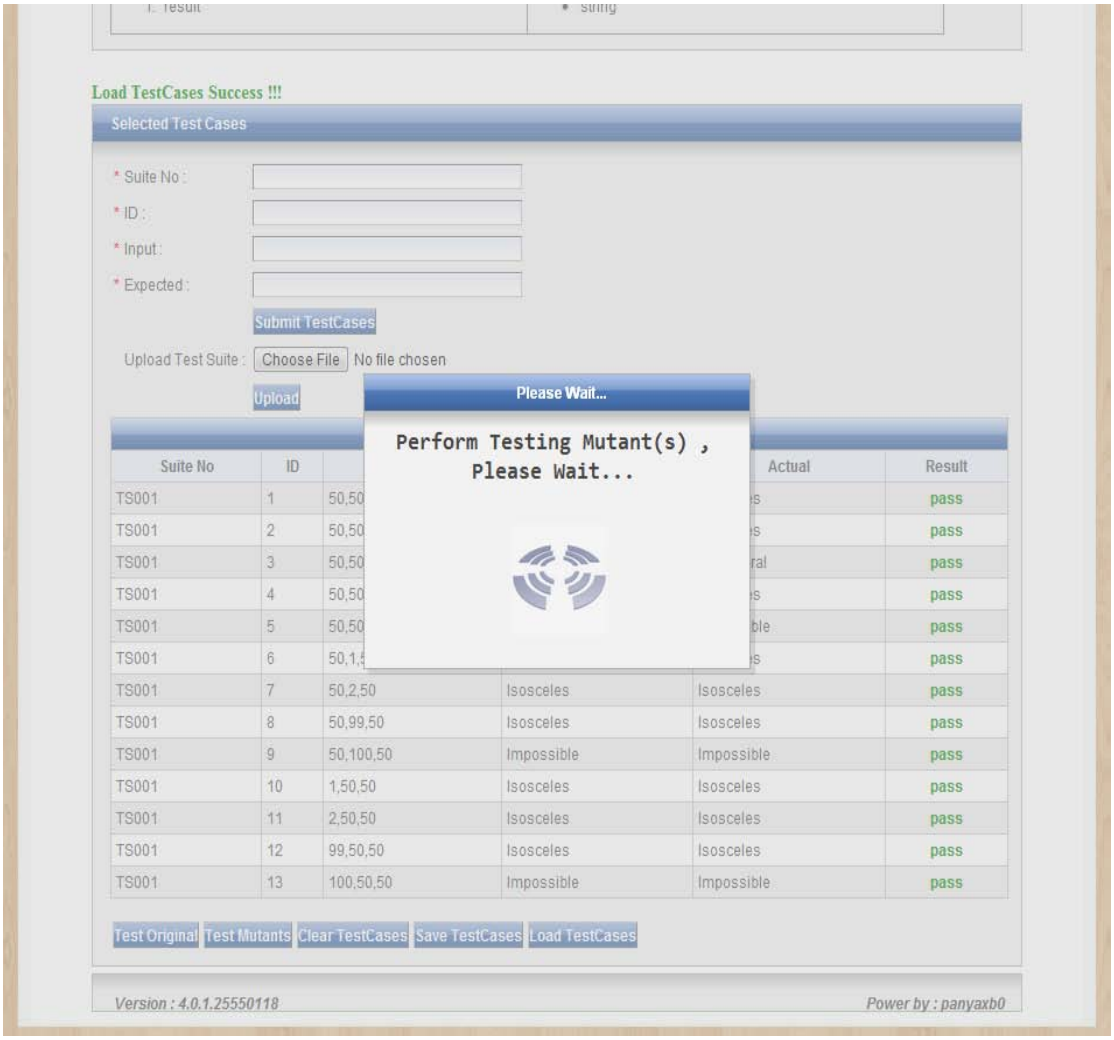
ภาพที่ 4.8 หน้าจอ SelectTestCases.jsf (ระบบไม่พบข้อมูลกรณีทดสอบ)



ภาพที่ 4.9 หน้าจอ SelectTestCases.jsf (ทดสอบโปรแกรมต้นแบบ)



ภาพที่ 4.10 หน้าจอ SelectTestCases.jsf (ทดสอบโปรแกรมต้นแบบเสร็จสิ้น)



ภาพที่ 4.11 หน้าจอ SelectTestCases.jsf (ทดสอบมิวแตนท์)

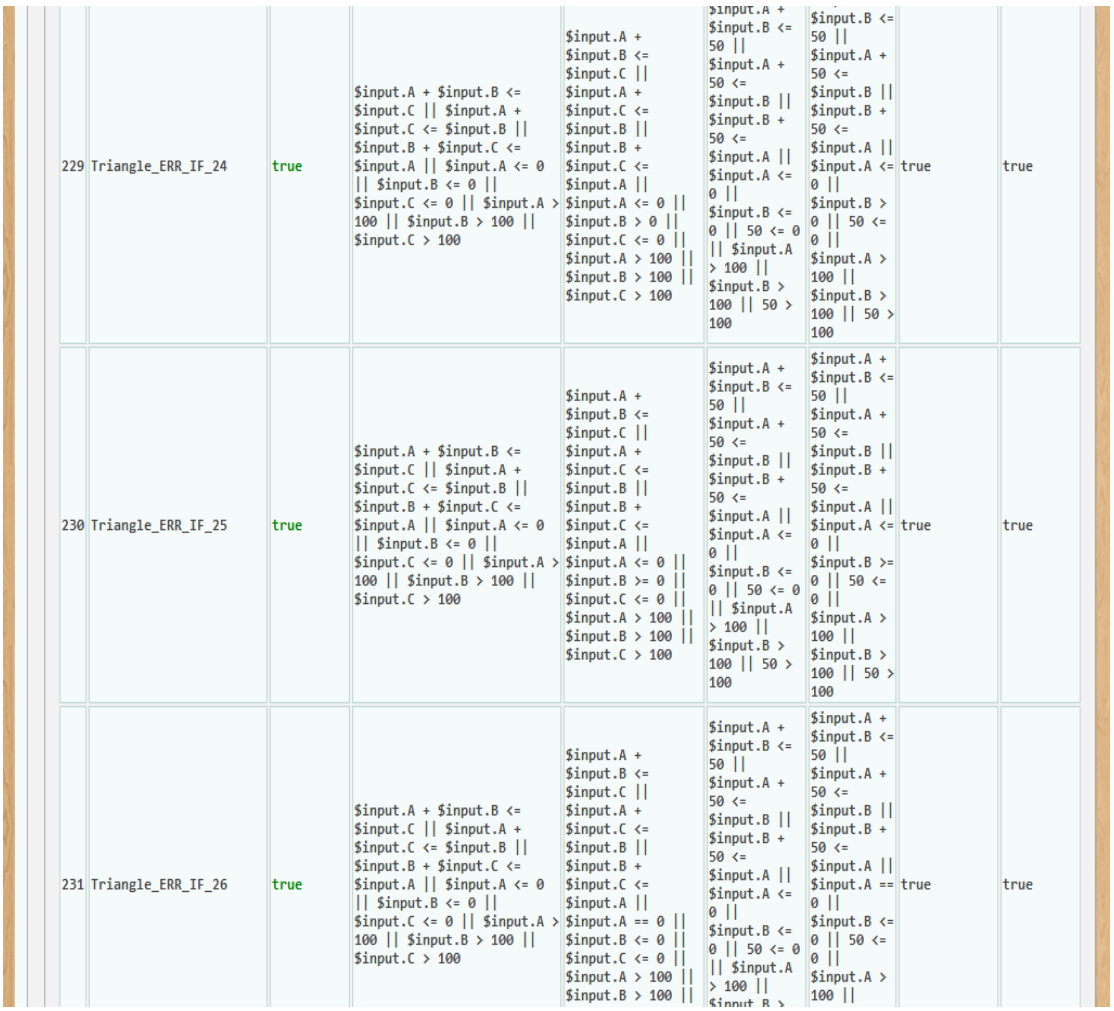
4.2.5 หน้าจอ ExecuteTestMutants.jsf

หน้าจอ ExecuteTestMutants.jsf ดังภาพที่ 4.12 4.13 จะแสดงเมื่อระบบได้ทำการทดสอบมิวแตนท์ครบทุกตัวเรียบร้อยแล้ว ซึ่งภาพที่ 4.12 แสดงผลเป็นตารางซึ่งประกอบด้วย 9 คอลัมน์ เรียงตามลำดับจากซ้ายไปขวาดังนี้ เลขที่ลำดับ ชื่อของมิวแตนท์ มิวแตนท์ถูกกำจัด นิพจน์ของโปรแกรมต้นแบบ นิพจน์ของมิวแตนท์ นิพจน์หลังการแทนที่ข้อมูลทดสอบของโปรแกรมต้นแบบ นิพจน์หลังการแทนที่ข้อมูลทดสอบของมิวแตนท์ ผลลัพธ์ของนิพจน์ของโปรแกรมต้นแบบ และผลลัพธ์ของนิพจน์ของมิวแตนท์

120	Triangle_EAA_IF_6	true	$input.A + $input.B	$input.A * $input.B	$input.A + 50	$input.A * 50	150.0	5000.0
121	Triangle_EAA_IF_7	true	$input.A + $input.B	$input.A / $input.B	$input.A + 50	$input.A / 50	150.0	2.0
122	Triangle_EAA_IF_8	true	$input.A + $input.B	$input.A % $input.B	$input.A + 50	$input.A % 50	150.0	0.0
123	Triangle_EAA_IF_9	true	$input.B + $input.C	$input.B - $input.C	$input.B + 50	$input.B - 50	100.0	0.0
124	Triangle_EAA_IF_10	true	$input.B + $input.C	$input.B * $input.C	$input.B + 50	$input.B * 50	100.0	2500.0
125	Triangle_EAA_IF_11	true	$input.B + $input.C	$input.B / $input.C	$input.B + 50	$input.B / 50	100.0	1.0
126	Triangle_EAA_IF_12	true	$input.B + $input.C	$input.B % $input.C	$input.B + 50	$input.B % 50	100.0	0.0
127	Triangle_EAA_IF_1	true	$input.B * $input.B	$input.B + $input.B	50 * 50	50 + 50	2500.0	100.0
128	Triangle_EAA_IF_2	true	$input.B * $input.B	$input.B - $input.B	50 * 50	50 - 50	2500.0	0.0
129	Triangle_EAA_IF_3	true	$input.B * $input.B	$input.B / $input.B	50 * 50	50 / 50	2500.0	1.0
130	Triangle_EAA_IF_4	true	$input.B * $input.B	$input.B % $input.B	50 * 50	50 % 50	2500.0	0.0
131	Triangle_EAA_IF_5	true	$input.B * $input.B	$input.B + $input.B	50 * 50	50 + 50	2500.0	100.0
132	Triangle_EAA_IF_6	true	$input.B * $input.B	$input.B - $input.B	50 * 50	50 - 50	2500.0	0.0
133	Triangle_EAA_IF_7	true	$input.B * $input.B	$input.B / $input.B	50 * 50	50 / 50	2500.0	1.0
134	Triangle_EAA_IF_8	true	$input.B * $input.B	$input.B % $input.B	50 * 50	50 % 50	2500.0	0.0
135	Triangle_EAA_IF_9	true	$input.B + $input.C	$input.B - $input.C	$input.B + 50	$input.B - 50	100.0	0.0
136	Triangle_EAA_IF_10	true	$input.A + $input.A	$input.A * $input.A	100 + 100	100 * 100	200.0	10000.0
137	Triangle_EAA_IF_11	true	$input.B + $input.C	$input.B / $input.C	$input.B + 50	$input.B / 50	100.0	1.0
138	Triangle_EAA_IF_12	true	$input.B + $input.C	$input.B % $input.C	$input.B + 50	$input.B % 50	100.0	0.0
139	Triangle_EAA_IF_13	true	$input.C * $input.C	$input.C + $input.C	50 * 50	50 + 50	2500.0	100.0
140	Triangle_EAA_IF_14	true	$input.C * $input.C	$input.C - $input.C	50 * 50	50 - 50	2500.0	0.0
141	Triangle_EAA_IF_15	true	$input.C * $input.C	$input.C / $input.C	50 * 50	50 / 50	2500.0	1.0

ภาพที่ 4.12 หน้าจอ ExecuteTestMutants.jsf

ส่วนในภาพที่ 4.13 จะแสดงผลเป็นตารางเช่นกันซึ่งประกอบด้วย 9 คอลัมน์ โดยเรียงจากซ้ายไปขวาดังนี้ เลขที่ลำดับ ชื่อของมิวแตนท์ มิวแตนท์ถูกกำจัด ประพจน์ของโปรแกรมต้นแบบ ประพจน์ของมิวแตนท์ ประพจน์หลังการแทนที่ข้อมูลทดสอบของโปรแกรมต้นแบบ ประพจน์หลังการแทนที่ข้อมูลทดสอบของมิวแตนท์ ผลลัพธ์ของประพจน์ของโปรแกรมต้นแบบ และผลลัพธ์ของประพจน์ของมิวแตนท์



229	Triangle_ERR_IF_24	true	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A <= 0 || $input.B <= 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 || $input.C > 100	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A <= 0 || $input.B > 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 || $input.C > 100	$input.A + $input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A <= 0 || $input.B <= 0 || 50 <= 0 || $input.A > 100 || $input.B > 100 || 50 > 100	$input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A <= 0 || $input.B > 0 || 50 <= 0 || $input.A > 100 || $input.B > 100 || 50 > 100	true	true
230	Triangle_ERR_IF_25	true	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A <= 0 || $input.B <= 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 || $input.C > 100	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A <= 0 || $input.B >= 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 || $input.C > 100	$input.A + $input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A <= 0 || $input.B <= 0 || 50 <= 0 || $input.A > 100 || $input.B > 100 || 50 > 100	$input.A + $input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A <= 0 || $input.B >= 0 || 50 <= 0 || $input.A > 100 || $input.B > 100 || 50 > 100	true	true
231	Triangle_ERR_IF_26	true	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A <= 0 || $input.B <= 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 || $input.C > 100	$input.A + $input.B <= $input.C || $input.A + $input.C <= $input.B || $input.B + $input.C <= $input.A || $input.A == 0 || $input.B <= 0 || $input.C <= 0 || $input.A > 100 || $input.B > 100 ||	$input.A + $input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A <= 0 || $input.B <= 0 || 50 <= 0 || $input.A > 100 || $input.B >	$input.A + $input.B <= 50 || $input.A + 50 <= $input.B || $input.B + 50 <= $input.A || $input.A == 0 || $input.B <= 0 || 50 <= 0 || $input.A > 100 ||	true	true

ภาพที่ 4.13 หน้าจอ ExecuteTestMutants.jsf (ต่อ)

4.2.7 หน้าจอ DisplayWeMuTeAnalyze.jsf

หน้าจอ DisplayWeMuTeAnalyze.jsf ดังภาพที่ 4.14 จะแสดงเมื่อนักทดสอบกดปุ่ม Display Result จากหน้าจอ ExecuteTestMutants.jsf โดยหน้าจอจะแบ่งออกเป็น 2 ส่วน ดังนี้

1) Summary Operators - แสดงรายละเอียดของมิวแตนท์ที่ถูกสร้างในแต่ละตัวดำเนินการทั้งหมด และมิวแตนท์ที่ถูกกำจัดในแต่ละตัวดำเนินการ

2) Summary Experiment – แสดงรายละเอียดของมิวแตนท์ที่ถูกสร้างขึ้นทั้งหมด มิวแตนท์ที่ถูกกำจัดทั้งหมด เวลาที่ใช้ในการทดสอบ คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

ภาพที่ 4.14 หน้าจอ DisplayWeMuTeAnalyze.jsf

บทที่ 5
การทดสอบเครื่องมือ

บทนี้นำเสนอแนวทางในการทดสอบเครื่องมือทดสอบแบบวีคมิวเทชันสำหรับภาษาบีเพล โดยอธิบายจากสภาพแวดล้อมที่ใช้ในการทดสอบเครื่องมือ ขั้นตอนแต่ละขั้นในการทดสอบโปรแกรมบีเพลที่ใช้ในการทดสอบ กรณีทดสอบที่ใช้ทดสอบโปรแกรมบีเพล และผลของการทดสอบโปรแกรม และสรุปผลของการทดสอบโปรแกรม

5.1 สภาพแวดล้อมที่ใช้ในการทดสอบ

การทดสอบจะทดสอบโดยใช้สภาพแวดล้อมเดียวกับเครื่องมือทดสอบดังที่อธิบายไว้ในบทที่ 4 ของวิทยานิพนธ์ฉบับนี้

5.2 ขั้นตอนในการทดสอบเครื่องมือที่พัฒนา

1) สร้างกระบวนการบีเพลขึ้นมา สร้างเว็บเซอร์วิซเพื่อใช้เป็นส่วนต่อประสานกับโปรแกรมซึ่งประกอบไปด้วยเอกสารวิซเดิล และเอกสารดีพลอย หรืออาจจะมีเอ็กซ์เอ็มแอลสคีมา (อาจมีหรือไม่มีก็ได้)

2) สร้างจาวาคลาสที่เรียกใช้เว็บเซอร์วิซในการตรวจสอบข้อมูลผ่านทางฐานข้อมูล

3) ออกแบบสคีมาฐานข้อมูลที่จะใช้ทดสอบ เช่น ตาราง และชนิดข้อมูล เป็นต้น

4) สร้างข้อมูลที่จะใช้เป็นกรณีทดสอบกับโปรแกรม

5) ทำการติดตั้งเว็บเซอร์วิซที่สร้างขึ้นไว้ที่เครื่องบริการเว็บ

6) ทำการติดตั้งโปรแกรมกระบวนการบีเพลที่ต้องการจะทดสอบไว้ที่เครื่องประมวลผลบีเพล

7) ใช้เครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลในการทดสอบโปรแกรม

8) รวมรวบผลของการทดสอบและใส่ข้อมูลตามรูปแบบที่ได้ออกแบบไว้ในบทที่ 3 ของวิทยานิพนธ์ฉบับนี้

5.3 โปรแกรมที่ใช้ในการทดสอบ

โปรแกรมที่ถูกใช้ในการทดสอบด้วยเครื่องมือทดสอบแบบวีคมิวเทชันมีทั้งหมด 9 โปรแกรม คือ โปรแกรมหาชนิดของสามเหลี่ยม โปรแกรมเครื่องคิดเลข โปรแกรมการอนุมัติการกู้ยืม โปรแกรมเอทีเอ็ม โปรแกรมการซื้อสินค้า โปรแกรมการจองตั๋วท่องเที่ยว โปรแกรมเพิ่มค่าตัวเลข โปรแกรมหาค่ากำลังสอง และโปรแกรมใบสั่งซื้อสินค้า โปรแกรมเหล่านี้จะถูกพัฒนาขึ้น

ด้วยผู้วิจัยและใช้ภาษาบีเพลในการพัฒนา ซึ่งจะเป็นโปรแกรมที่ประกอบด้วยเอกสารบีเพล เอกสารวิซเดิล เอกสารดีพลอย และเอกสารเอ็กซ์เอ็มแอลสคีมา (สามารถมีหรือไม่มีก็ได้) ทั้ง 4 เอกสารนี้จะถูกนำเข้าสู่เครื่องมือทดสอบวีคมิวเทชันในรูปของเอกสารซิป

5.3.1 โปรแกรมหาชนิดของสามเหลี่ยม (Triangle)

แผนภาพกิจกรรมดังภาพที่ 5.1 เป็นแผนภาพกิจกรรมของโปรแกรมหาชนิดของสามเหลี่ยม โดยจะรับข้อมูลนำเข้ามาเป็นความยาวทั้ง 3 ด้าน คือ A, B และ C โดยมีค่าอยู่ระหว่าง 1 ถึง 100 และตรวจสอบว่าเป็นสามเหลี่ยมชนิดไหนโดยมีขั้นตอนดังต่อไปนี้

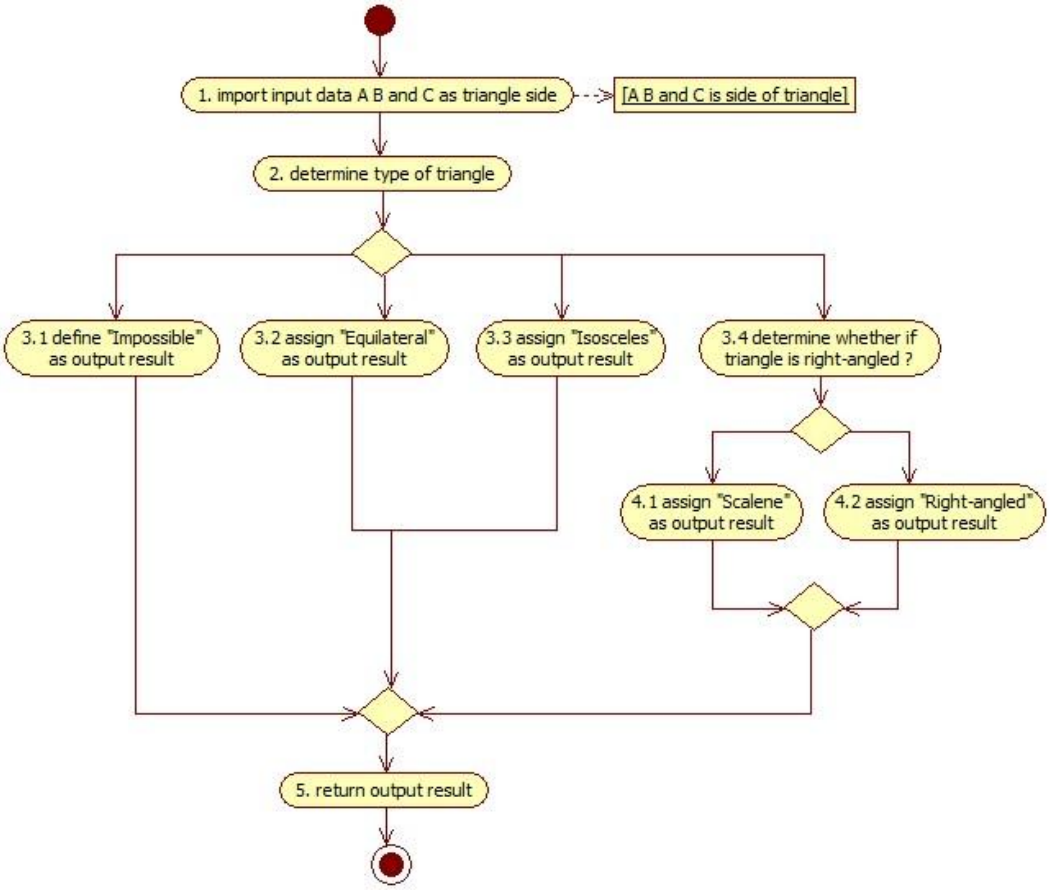
1) ตรวจสอบก่อนว่าด้านทั้ง 3 ด้านที่รับข้อมูลเข้ามาสามารถประกอบกันเป็นรูปสามเหลี่ยมได้หรือไม่ โดยใช้เงื่อนไข A > 100 or B > 100 or C >100 or A + B <= C or A + C <= B or B + C <= A or A <= 0 or B <= 0 or C <= 0 ดังนั้น ถ้าด้านทั้ง 3 ด้านไม่สามารถประกอบกันเป็นสามเหลี่ยมได้ โปรแกรมจะกำหนดค่าผลลัพธ์ให้เป็น Impossible

2) ตรวจสอบว่าด้านทั้ง 3 ด้านประกอบกันเป็นสามเหลี่ยมด้านเท่าหรือไม่ โดยการใช้เงื่อนไข C = A and B = C and A = B ถ้าเป็นสามเหลี่ยมด้านเท่าโปรแกรมจะทำการกำหนดค่าผลลัพธ์ให้เป็น Equilateral

3) ตรวจสอบว่าด้านทั้ง 3 ด้านประกอบกันเป็นสามเหลี่ยมหน้าจั่วหรือไม่ โดยใช้เงื่อนไข (A = C and A != B) or (A = B and A != C) or (C = B and C != A) ถ้าเป็นสามเหลี่ยมหน้าจั่วโปรแกรมจะกำหนดค่าผลลัพธ์ให้เป็น Isosceles

4) ตรวจสอบว่าด้านทั้ง 3 ด้านประกอบกันเป็นสามเหลี่ยมด้านไม่เท่าหรือไม่โดยใช้เงื่อนไข A != B and A != C and B != C ถ้าเป็นสามเหลี่ยมด้านไม่เท่าโปรแกรมจะกำหนดผลลัพธ์เป็น Scalene

5) ตรวจสอบว่าด้านทั้ง 3 ประกอบกันเป็นสามเหลี่ยมมุมฉากหรือไม่ โดยใช้เงื่อนไข ((A * A) + (B * B) = (C * C)) or ((B * B) + (C * C) = (A * A)) or ((A * A) + (C * C) = (A * A)) ถ้าเป็นสามเหลี่ยมมุมฉากโปรแกรมจะกำหนดค่า Right-angled



ภาพที่ 5.1 แผนภาพกิจกรรมของโปรแกรม Triangle

5.3.2 โปรแกรมการอนุมัติการการกู้ยืม (LoanApprovalProcess)

โปรแกรมการอนุมัติการกู้ยืมนี้ถูกอธิบายด้วยภาพที่ 5.2 จะรับข้อมูลเข้ามาเป็นจำนวนเงิน (amount) และชื่อ (name) โดยจะมีผู้กระทำแบ่งออกเป็น 3 ส่วน คือ ส่วนโปรแกรมบีเพล (LoanApprovalProcess) เป็นกระบวนการกลางที่ไปเรียกใช้บริการ (LoanService) จากนั้นบริการจะไปเรียกใช้งานจาวาคลาส (LoanApprovalSoapBindingImpl) อีกครั้งหนึ่ง โดยมีขั้นตอนของกระบวนการดังนี้

1) ตรวจสอบความเสี่ยงโดยทางบริการ LoanService จะรับค่าจำนวนเงิน (amount) ที่ส่งมาจาก LoanApprovalProcess เพื่อนำไปประมวลผลที่จาวาคลาส โดยมีเงื่อนไขดังนี้

- amount <= 100000 โปรแกรมจะส่งระดับความเสี่ยง (riskLevel = 1) ออกเป็นผลลัพธ์ให้กระบวนการบีเพล

- amount < 100000 and amount <= 500000 โปรแกรมจะส่งระดับความเสี่ยง (riskLevel = 3) ออกเป็นผลลัพธ์ให้กระบวนการบีเพล

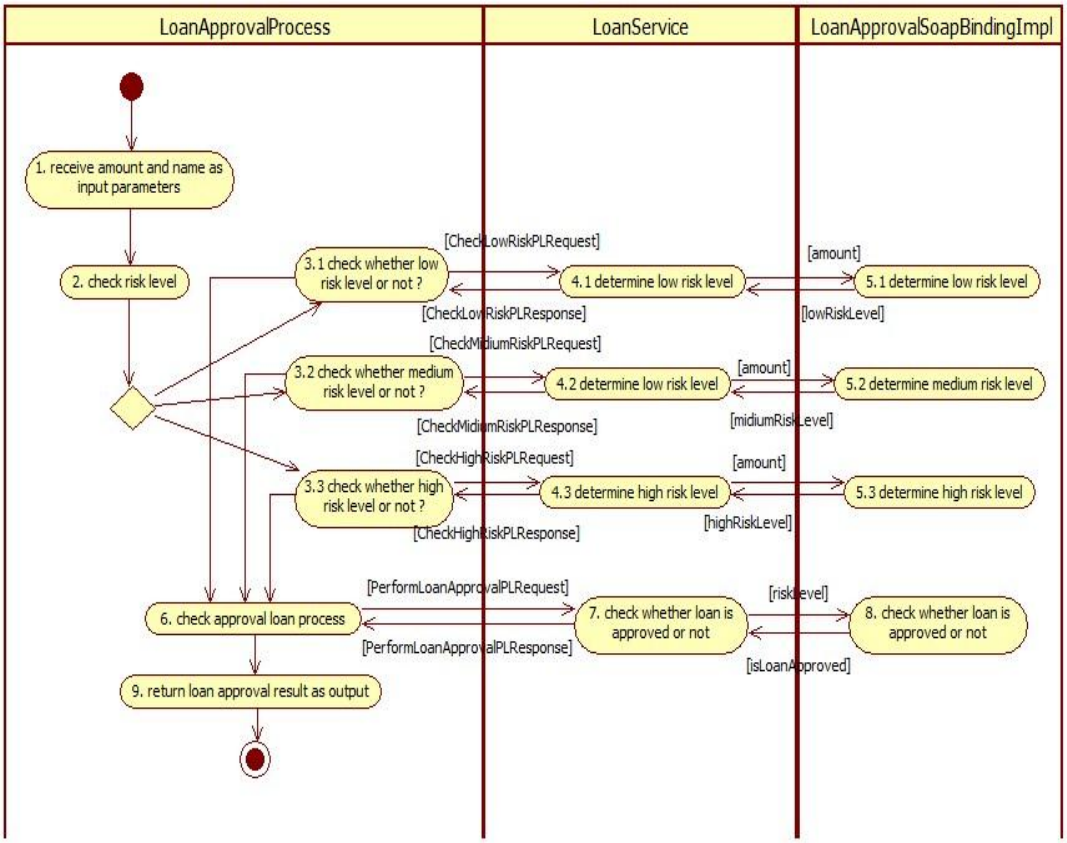
- amount > 500000 โปรแกรมจะส่งค่าระดับความเสี่ยง (riskLevel = 5) ออกเป็นผลลัพธ์ให้กระบวนการบีเพล

2) การพิจารณาการอนุมัติการกู้ยืมจะเริ่มจาก LoanApprovalProcess ส่งค่าระดับความเสี่ยงและชื่อไปที่บริการเพื่อประมวลผล โดยมีเงื่อนไขดังนี้

- level < 3 โปรแกรมจะส่งค่าการอนุมัติการกู้ยืม isApprove = true ออกเป็นผลลัพธ์

- level == 3 โปรแกรมจะส่งค่าการอนุมัติการกู้ยืม isApprove = true ออกเป็นผลลัพธ์

- level > 3 โปรแกรมจะส่งค่าการอนุมัติการกู้ยืม isApprove = false ออกเป็นผลลัพธ์



ภาพที่ 5.2 แผนภาพกิจกรรมของโปรแกรม LoanApprovalProcess

5.3.3 โปรแกรมเครื่องคิดเลข (SimpleCalculator)

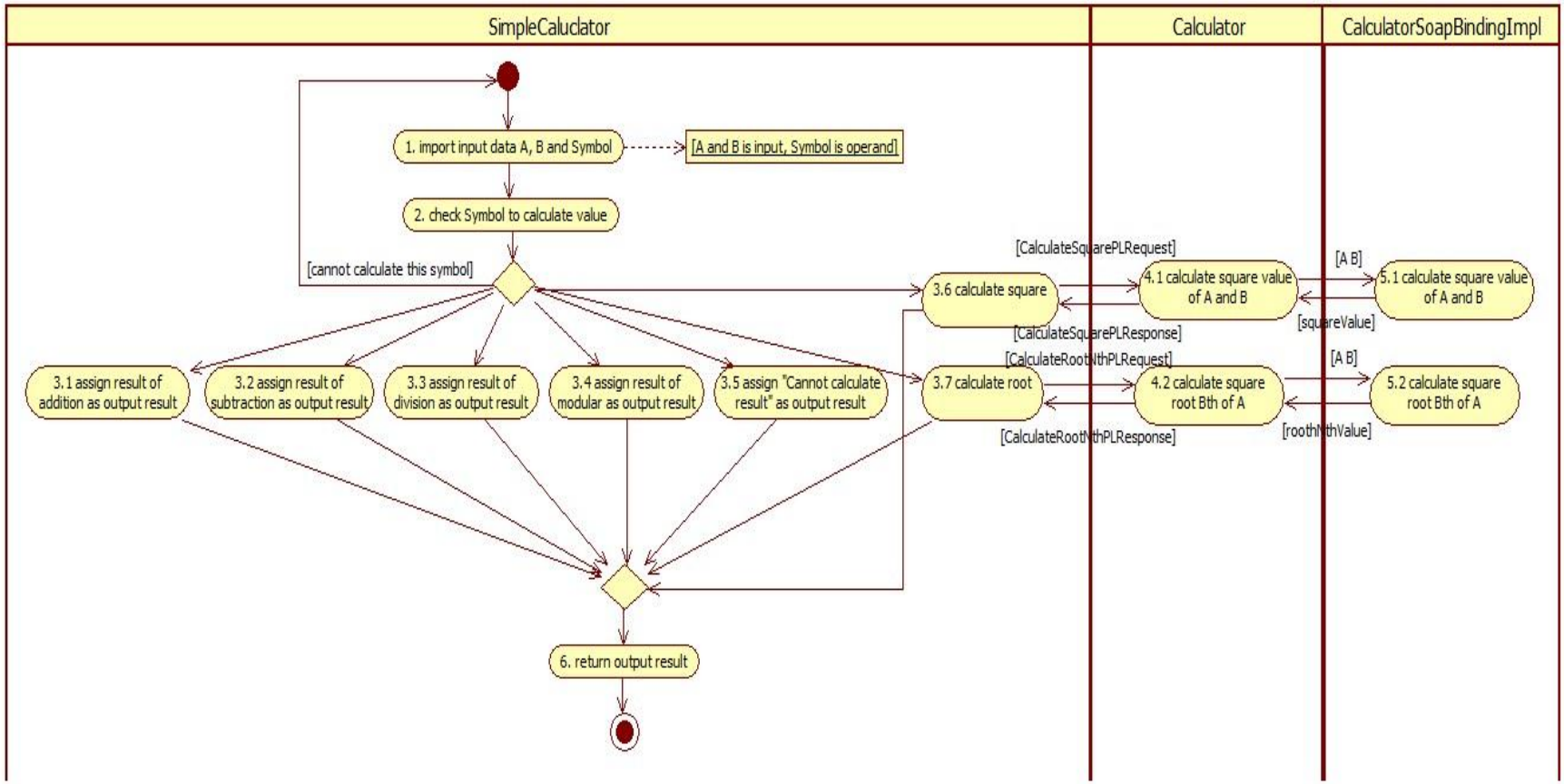
ในแผนภาพกิจกรรมของโปรแกรมเครื่องคิดเลข (SimpleCalculator) ดังภาพที่ 5.3 เป็นโปรแกรมคำนวนหาผลลัพธ์ของค่า 2 ค่าโดยกระบวนการจะรับข้อมูลนำเข้าทั้งหมด 3 ค่า คือ A, B และ Symbol ซึ่งมีขั้นตอนดังนี้

ตรวจสอบค่าของข้อมูล Symbol ว่าเป็นสัญลักษณ์อะไรเพื่อใช้ในการคำนวนค่า 2 ค่า ซึ่งค่าของ Symbol ที่รับมาจะแบ่งการพิจารณาออกเป็น 3 แบบ ดังนี้

- คำนวนผลลัพธ์ภายในกระบวนการ คือ ค่า Symbol มีค่า + - * div และ mod

- คำนวนผลลัพธ์ด้วยบริการภายนอก คือ ค่า Symbol มีค่า ^ และ nth โดยโปรแกรมบีเพลจะส่งค่า 3 ไปคำนวนด้วยบริการอื่น คือ CalculatorService และ CalculatorSoapBindingImpl ตามลำดับ

- ไม่สามารถคำนวนผลลัพธ์ได้ คือ ค่า Symbol เป็นค่าอื่นนอกเหนือจากที่กล่าวมาในเงื่อนไขข้างต้น



ภาพที่ 5.3 แผนภาพกิจกรรมของโปรแกรม SimpleCalculator

5.3.4 โปรแกรมเอทีเอ็ม (ATMProcess)

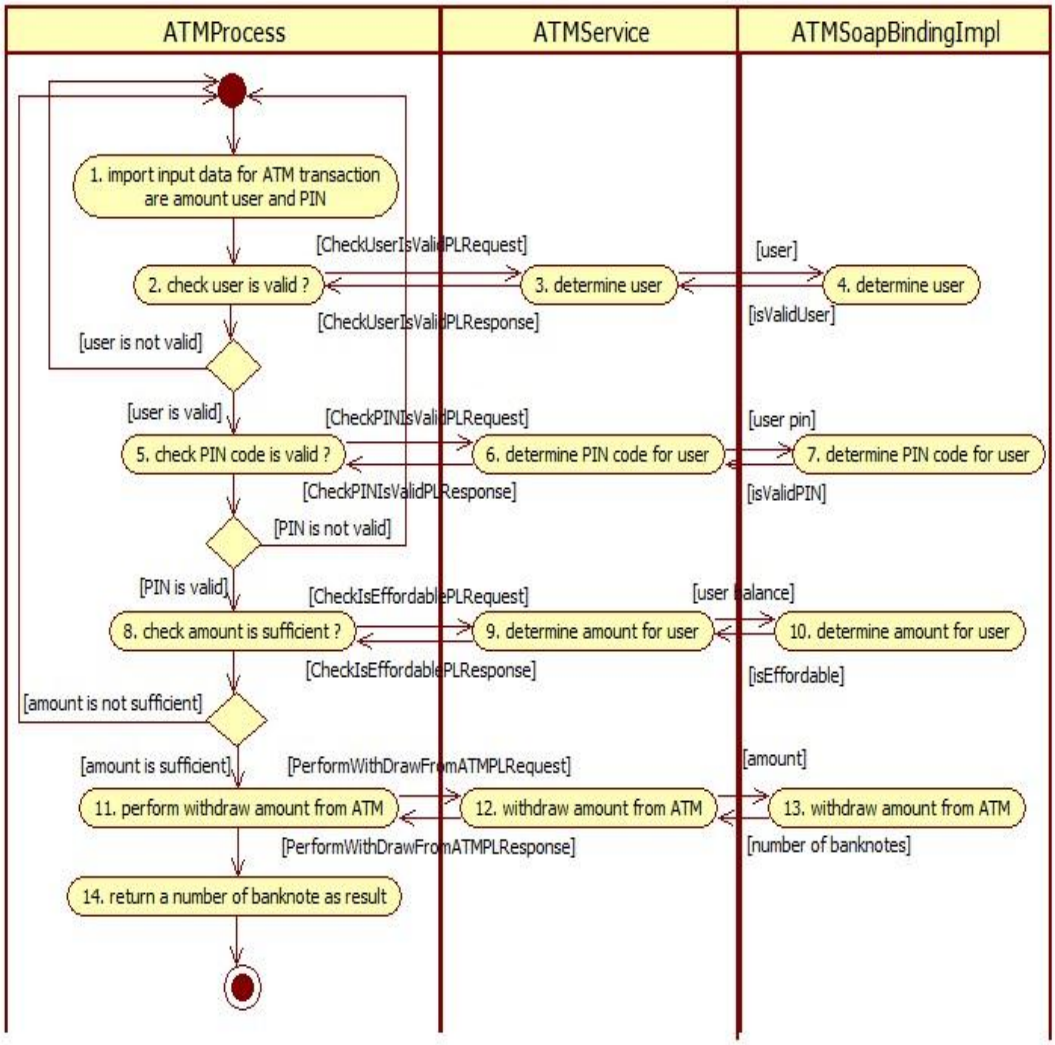
โปรแกรมอีทีเอ็มถูกอธิบายด้วยแผนภาพกิจกรรมดังภาพที่ 5.4 โปรแกรมจะรับข้อมูลนำเข้ามา 3 ค่า คือ จำนวนเงิน (amount) รหัสผู้ใช้ (user) และพิน (PIN) ซึ่งกระบวนการมีขั้นตอนดังนี้

1) ตรวจสอบว่ามีรหัสผู้ใช้อยู่ในระบบหรือไม่ โดยบีเพล ATMProcess จะส่งข้อความไปที่เว็บเซอร์วิซ ATMService และตรวจสอบกับจาวาคลาส ATMSoapBindingImpl อีกครั้ง เมื่อพบรหัสผู้ใช้อยู่ในระบบก็จะส่งผลลัพธ์ออกมาเป็น existUser = true กลับมาให้กับตัวแปร validUser ถ้าไม่ถูกต้องก็แจ้งข้อความ User does not exist แล้วกลับไปกรอกข้อมูลใหม่

2) ตรวจสอบว่ารหัสพินที่ใส่มาถูกต้องหรือไม่ โดยโปรแกรมบีเพล ATMProcess ส่งข้อความไปที่เว็บเซอร์วิซ ATMService และตรวจสอบข้อมูลผ่านฐานข้อมูลที่ ATMSoapBindingImpl ถ้ารหัสพินถูกต้องและตรงกับผู้ใช้งานก็จะส่งผลลัพธ์ validPIN = true มาให้กับตัวแปร validPIN ในกระบวนการบีเพล

3) ตรวจสอบว่าจำนวนเงินที่มีอยู่เพียงพอหรือไม่ โดย ATMProcess ส่งข้อมูลจำนวนเงินและรหัสผู้ใช้ ไปตรวจสอบที่ฐานข้อมูลโดยผ่านทางเว็บเซอร์วิซ ATMService และจาวาคลาส ATMSoapBindingImpl เมื่อตรวจสอบพบว่าจำนวนเงินเพียงพอก็จะส่งผลลัพธ์ validBalance = true มาให้กับตัวแปร validBalance ในกระบวนการบีเพล ถ้าจำนวนเงินไม่เพียงพอต้องกลับไปใส่ข้อมูลใหม่

4) ทำการถอนเงินในบัญชี โดยกระบวนการจะตรวจสอบตัวแปรทั้ง 3 ค่า ดังนี้ validUser validPIN และ validBalance ว่ามีค่าเป็นจริงหรือไม่ ถ้าเป็นจริงก็จะส่งผลลัพธ์ออกไปที่ตัวแปร atmResult



ภาพที่ 5.4 แผนภาพกิจกรรมของโปรแกรม ATMProcess

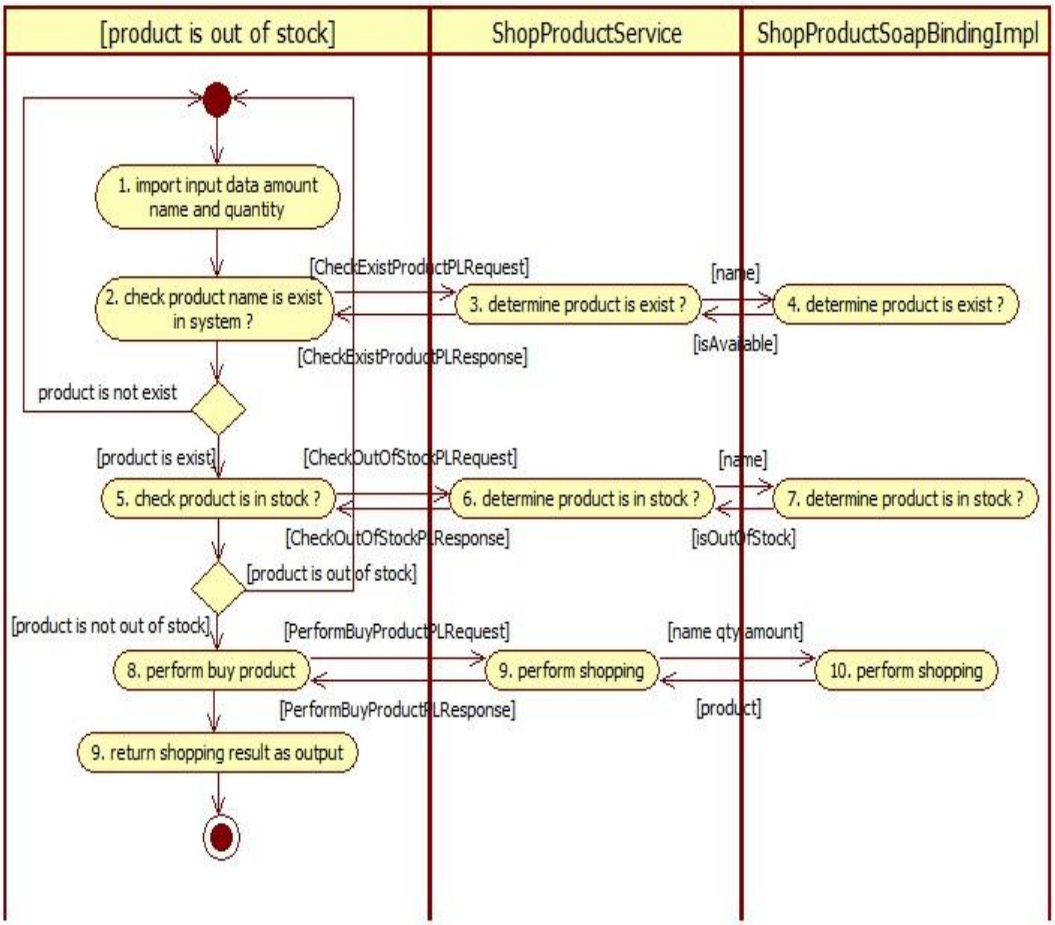
5.3.5 โปรแกรมขายซื้อสินค้า (ShopProductProcess)

โปรแกรมซื้อสินค้ามีขั้นตอนดังภาพที่ 5.5 จะรับข้อมูลนำเข้ามา 3 ค่า นั่นคือ จำนวนเงิน (amount) ชื่อสินค้า (name) และจำนวน (quantity) ซึ่งมีขั้นตอนดังต่อไปนี้

1) ตรวจสอบว่ามีสินค้าชิ้นนั้นอยู่หรือไม่ โดยกระบวนการบีเพล ShopProductProcess จะส่งข้อมูลไปที่เว็บเซอร์วิซ ShopProductService และจาวาคลาส ShopProductSoapBindingImpl เพื่อตรวจสอบสินค้าด้วยเงื่อนไขในฐานข้อมูลชื่อสินค้าไป ถ้ามีสินค้านั้นอยู่ในสต๊อกก็จะส่งค่า isAvailable = true เป็นผลลัพธ์ออกมายังบีเพล (isAvailableProduct)

2) ตรวจสอบว่ามีสินค้าในสต๊อกเพียงพอต่อความต้องการหรือไม่ ซึ่งกระบวนการบีเพลจะส่งข้อมูลชื่อสินค้าและจำนวนสินค้า ไปตรวจสอบที่ฐานข้อมูลโดยผ่านเว็บเซอร์วิซ ShopProductService และจาวาคลาส ShopProductSoapBindingImpl ตามลำดับ ถ้าสินค้ายังมีเพียงพอก็จะส่งผลลัพธ์ isOutOfStock = false มากำหนดให้ตัวแปร isOutOfStock ในโปรแกรมบีเพล

3) ทำการซื้อสินค้าโดยส่งค่า isAvailableProduct และ isOutOfStock ไปตรวจสอบว่าถ้าค่า isAvailable = true และ isOutOfStock = false จะสามารถซื้อสินค้ารายการนั้นได้



ภาพที่ 5.5 แผนภาพกิจกรรมของโปรแกรม ShopProductProcess

5.3.6 โปรแกรมการจองตั๋วการท่องเที่ยว (TravelReservationProcess)

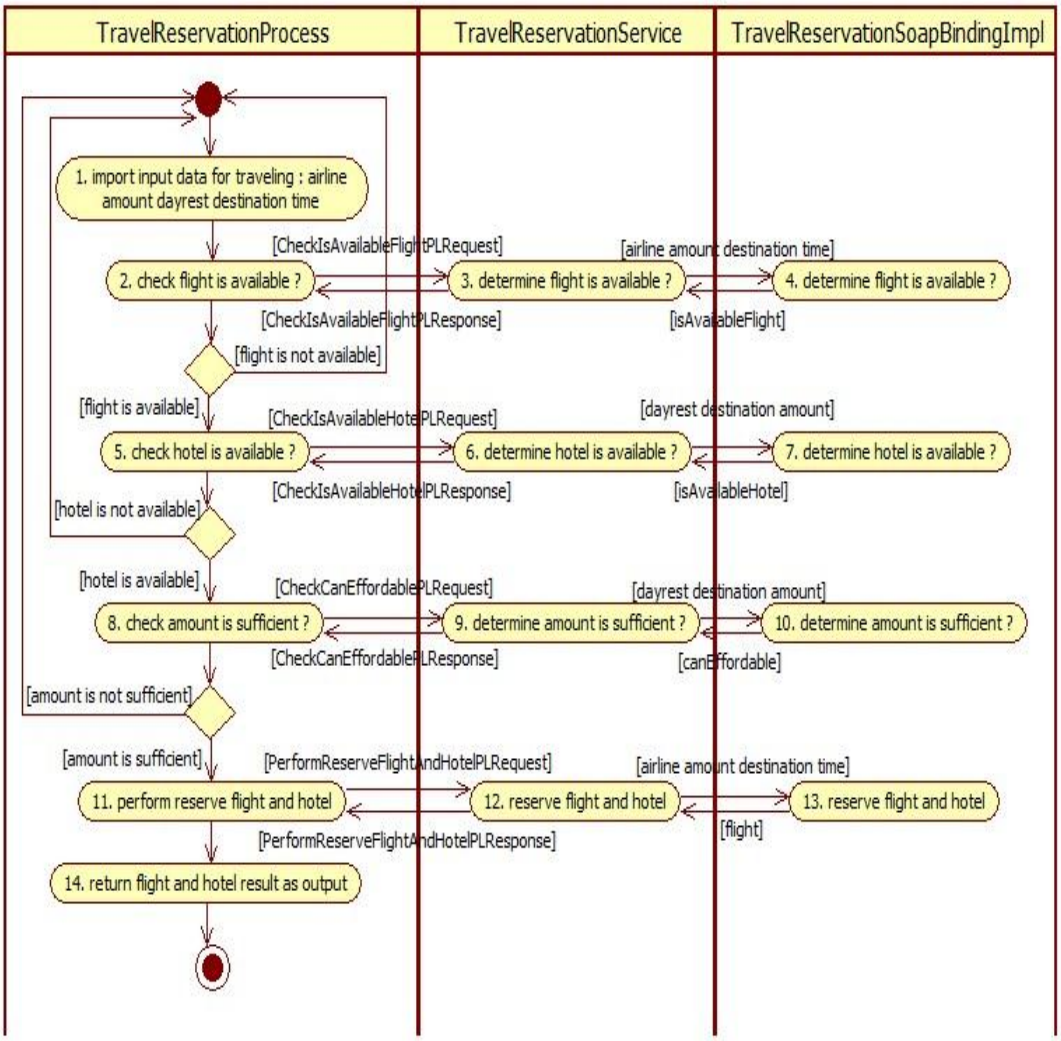
แผนภาพกิจกรรมในภาพที่ 5.6 แสดงขั้นตอนการจองตั๋วการท่องเที่ยว ซึ่งโปรแกรมนี้จะรับข้อมูลเข้ามา 5 ค่า คือ สายการบิน (airline) จำนวนเงิน (amount) จำนวนวันที่พัก (dayrest) ปลายทาง (destination) และเวลาเดินทาง(time) โดยโปรแกรมมีขั้นตอนดังต่อไปนี้

1) ตรวจสอบว่าเที่ยวบินที่ต้องการว่างหรือไม่ โดยกระบวนการบีเพลจะส่งข้อมูลนำเข้า คือ เที่ยวบิน จำนวนเงิน ปลายทาง และเวลาเดินทาง โดยส่งข้อมูลไปยังเว็บเซอร์วิซ TravelReservationService และจาวาคลาส TravelReservationSoapBindingImpl ตามลำดับ เพื่อตรวจสอบกับฐานข้อมูล ถ้าเที่ยวบินที่ต้องการนั้นว่างก็จะส่งผลลัพธ์ isAvailable = true กลับมาให้ตัวแปร isAvailableFlight ในกระบวนการบีเพล

2) ตรวจสอบว่าที่พักว่างหรือไม่ กระบวนการบีเพลจะส่งข้อมูล จำนวนวันที่พัก ปลายทาง และ จำนวนเงิน จากนั้นโปรแกรมจะตรวจสอบผ่านทางฐานข้อมูล ถ้าที่พักว่างโปรแกรมจะส่งผลลัพธ์ isAvailable = true มาที่ตัวแปร isAvailableHotel ในกระบวนการบีเพล

3) ตรวจสอบว่าจำนวนเงินเพียงพอต่อการท่องเที่ยวหรือไม่ เริ่มโดยจากกระบวนการบีเพลส่งข้อมูลของปลายทาง จำนวนวันที่พัก และจำนวนเงิน ไปตรวจสอบที่ฐานข้อมูลโดยผ่านทางบริการเว็บเซอร์วิซ TravelReservatonService และจาวาคลาส TravelReservationSoapBindingImpl ตามลำดับ ถ้าจำนวนเงินนั้นเพียงพอก็จะส่งผลลัพธ์ canEfford = true กลับมายังตัวแปร canEffordable ในกระบวนการ TravelReservationProcess

4) ทำการจองตั๋วการท่องเที่ยว โดยกระบวนการบีเพลจะตรวจสอบค่าตัวแปร isAvailableFlight isAvailableHotel และ canEffordable ซึ่งทั้ง 3 ค่าต้องมีค่าเป็น true จึงจะสามารถทำการจองตั๋วการท่องเที่ยวได้



ภาพที่ 5.6 แผนภาพกิจกรรมของโปรแกรม TravelReservationProcess

5.3.7 โปรแกรมการเพิ่มค่าตัวเลข (AddNumber)

แผนภาพกิจกรรมในภาพที่ 5.7 แสดงขั้นตอนการเพิ่มค่าตัวเลข ซึ่งโปรแกรมนี้จะรับข้อมูลเข้ามา 2 ค่า คือ ค่าเริ่มต้น (from) และจำนวนครั้งในการเพิ่มค่า (to) ซึ่งการเพิ่มจะเพิ่มค่าทีละหนึ่ง โดยโปรแกรมมีขั้นตอนดังต่อไปนี้

1) รับข้อมูลนำเข้ามาจากตัวแปร from และ to

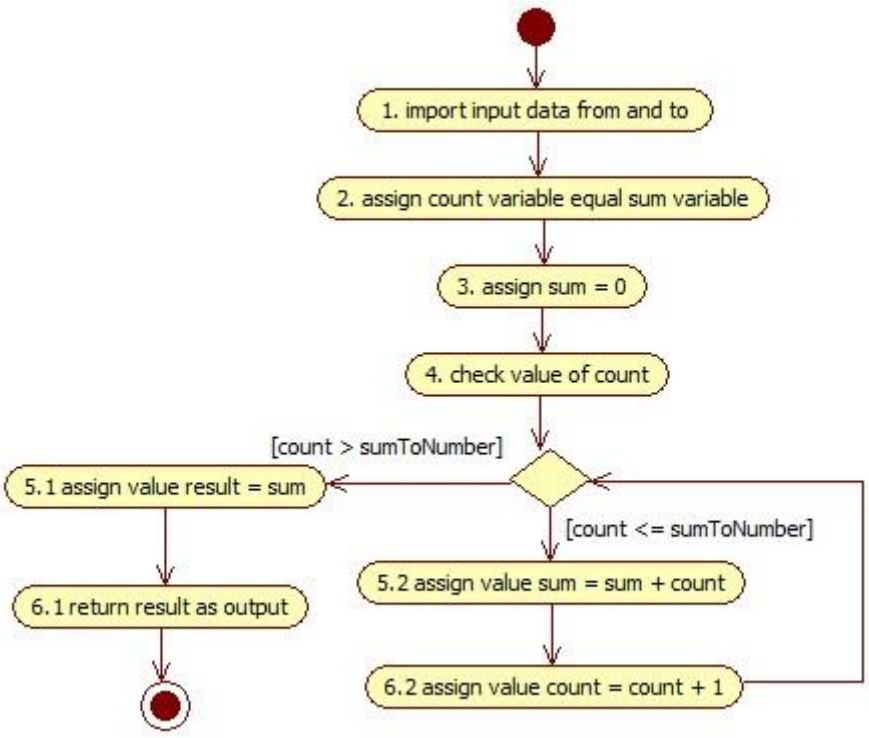
2) กำหนดค่าของตัวแปร count ให้กับตัวแปร sum

3) กำหนดค่าของตัวแปร sum เป็น 0

4) ทำการวนซ้ำไปเรื่อยๆ โดยนำค่าของตัวแปร count ไปเก็บในตัวแปร sum จากนั้นเพิ่มค่า count ทีละ 1

5) ถ้าตัวแปร count ยังน้อยกว่าหรือเท่ากับ sumToNumber จะทำในข้อ 4 ต่อไปเรื่อยๆ จนกระทั่งค่าตัวแปร count มากกว่า sumToNumber

6) แสดงผลลัพธ์ในการคำนวณ



ภาพที่ 5.7 แผนภาพกิจกรรมของโปรแกรม AddNumber

5.3.8 โปรแกรมหาค่ากำลังสอง (SquareValue)

แผนภาพกิจกรรมในภาพที่ 5.8 แสดงถึงขั้นตอนในการหาค่ากำลังสองของค่าๆ หนึ่งโดยรับข้อมูลเข้ามาเป็นตัวแปร n ซึ่งเป็นจำนวนจริงบวก และไม่สามารถคำนวนค่าที่รับมาเป็นจำนวนจริงที่ติดลบหรือข้อความได้ ซึ่งขั้นตอนของกระบวนการมีดังต่อไปนี้

1) นำค่าของ n ไปเก็บไว้ในตัวแปร totalNumbers

2) กำหนดค่าเริ่มให้กับตัวแปร sum variance และ element โดย sum = 0

3) เพิ่มค่าให้กับตัวแปร totalNumbers ทีละ 1 และกำหนดค่า 0 ให้กับตัวแปร elementSum

4) ทำการวนซ้ำโดยกำหนดให้ i = 1และมีค่าน้อยกว่าค่า n

5) เพิ่มค่า elementSum ทีละ 1 จากนั้นนำค่า i * i ไปเก็บไว้ที่ currentResult จากนั้นนำค่า sum บวกกับค่า currentResult พร้อมทั้งเก็บค่าไว้ที่ sum

6) กำหนดค่า i * i ให้กับตัวแปร currentResult

7) เพิ่มค่า sum ด้วยค่าในตัวแปร currentResult

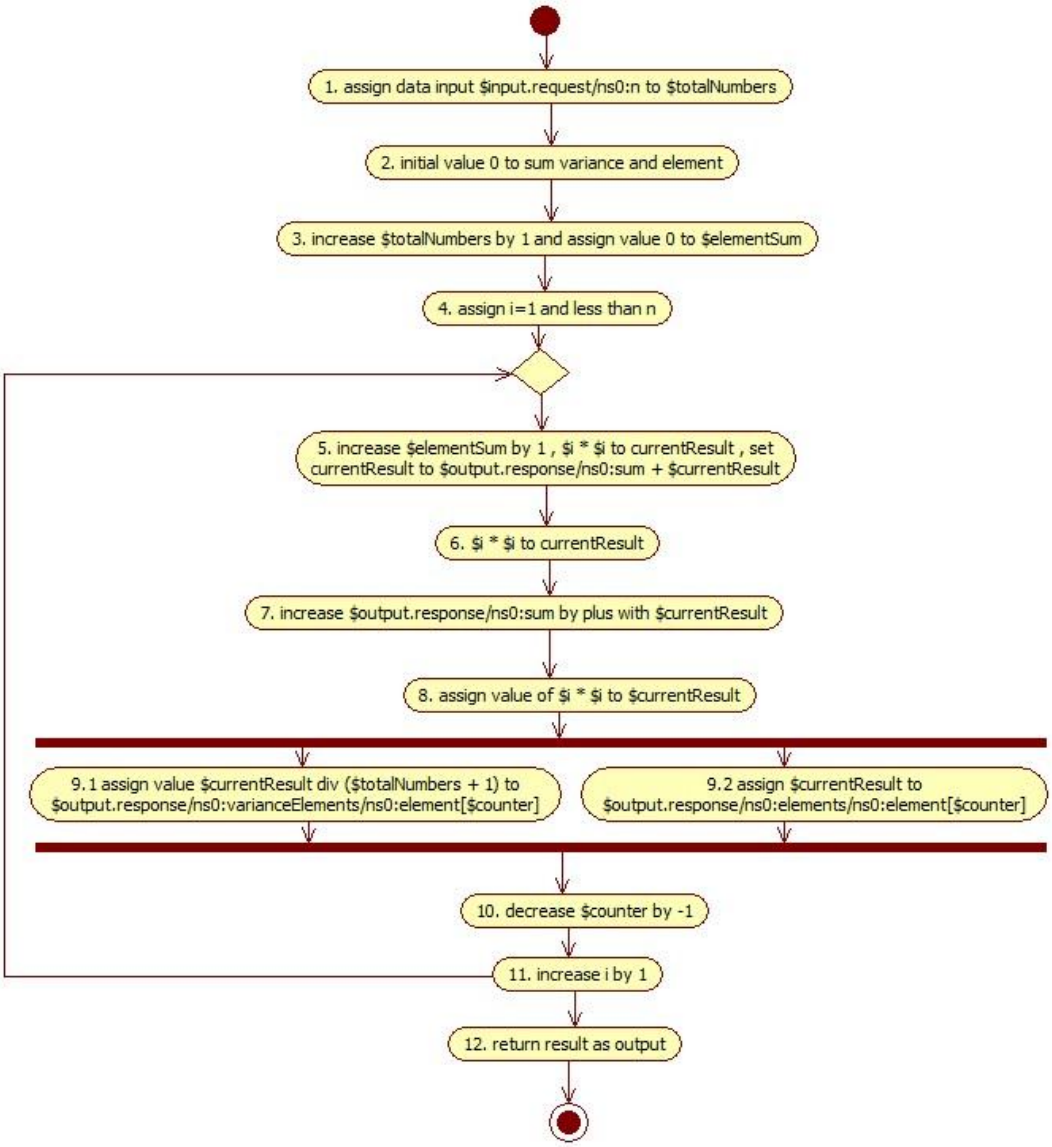
8) กำหนดค่า i * i ให้กับตัวแปร currentResult

9) กำหนดค่า currentResult div (totalNumber + 1) ให้กับ variance element และกำหนดค่า currentResult ให้กับ element

10) ลดค่าของตัวแปร counter ลง 1

11) เพิ่มค่า i ขึ้น 1

12) แสดงผลลัพธ์ในการคำนวณ



ภาพที่ 5.8 แผนภาพกิจกรรมของโปรแกรม SquareValue

5.3.9 โปรแกรมใบสั่งซื้อสินค้า (PurchaseOrderProcess)

แผนภาพกิจกรรมในภาพที่ 5.9 แสดงถึงขั้นตอนของใบสั่งซื้อสินค้าที่เป็นอะไหล่ ซึ่งโปรแกรมนี้จะรับข้อมูลเข้ามา 8 ค่า คือ รหัสใบสั่งซื้อ (poId) เลขที่ใบสั่งซื้อ (poNo) ชนิดของใบสั่งซื้อ (poType) วันที่ของใบสั่งซื้อ (poDate) จำนวนรวม (sumQty) จำนวนเงินรวม (sumAmt) ส่วนลดรวม (sumDisc) และหมายเหตุ (remark) โดยโปรแกรมมีขั้นตอนดังต่อไปนี้

1) กำหนดค่าเริ่มต้นในใบสั่งซื้อสินค้าที่ต้องการ

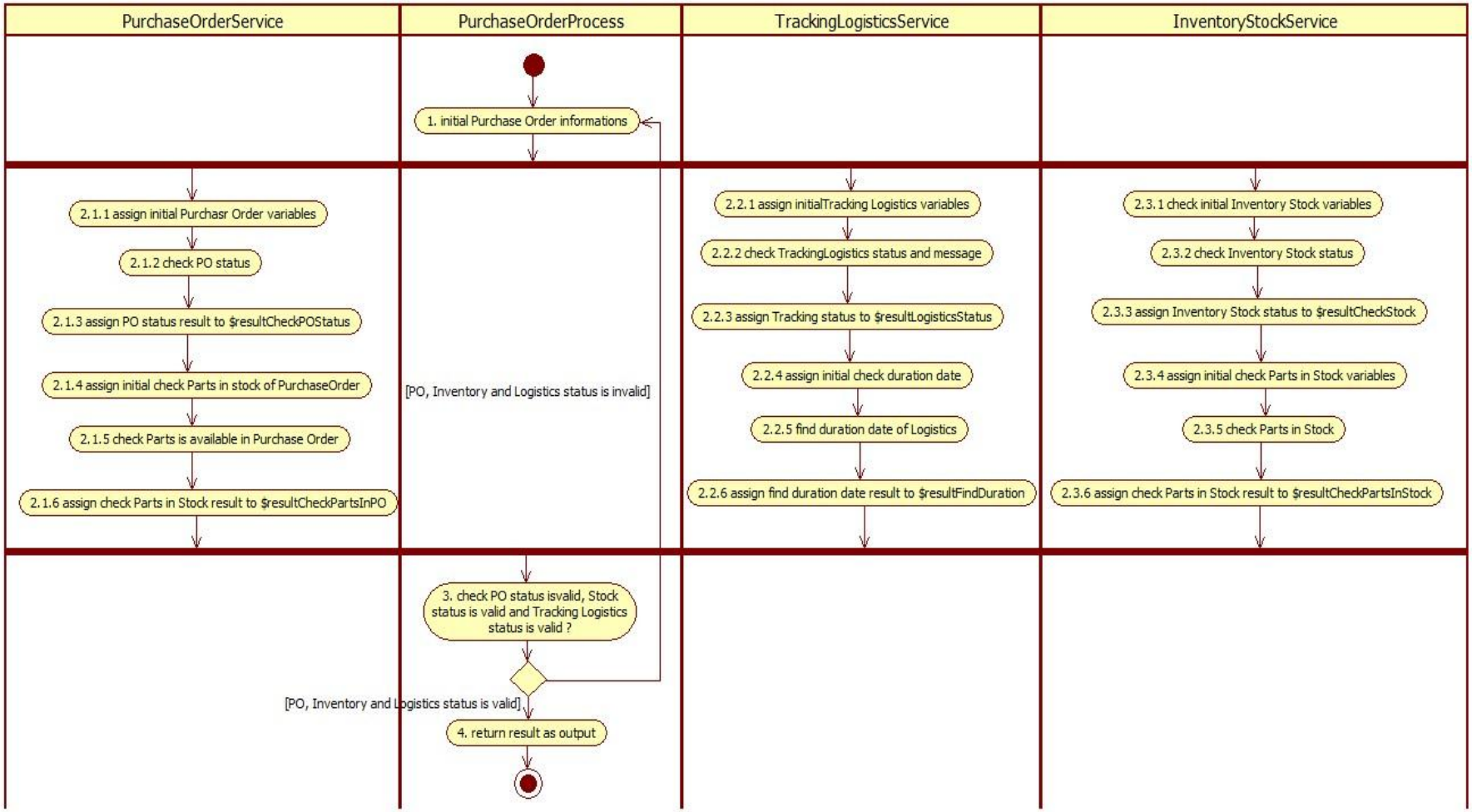
2) นำข้อมูลในใบสั่งซื้อสินค้าไปตรวจสอบในแต่ละบริการโดยมีบริการทั้งสิ้น 3 บริการ โดยมีขั้นตอนการตรวจสอบข้อมูลดังต่อไปนี้

2.1) PurchaseOrderService จะเริ่มต้นตรวจสอบสถานะของใบสั่งซื้อสินค้าว่าเป็นสถานะ Draft Order และ Invoice หรือไม่ จากนั้นตรวจสอบอะไหล่ที่สั่งซื้อนั้นมีเพียงพอในคลังสินค้าหรือไม่ โดยส่งผลลัพธ์มาที่ตัวแปร resultCheckPOStatus และ resultCheckPartsInPO ในกระบวนการบีเพล

2.2) TrackingLogisticsService จะเริ่มต้นโดยการตรวจสอบสถานะของการขนส่งสินค้าว่ามีสถานะของการขนส่งสินค้าเป็น OnTransport OnWarehouse และ OnLeaving หรือไม่ จากนั้นบริการจะตรวจสอบเวลาที่ใช้ในการขนส่ง โดยส่งผลลัพธ์มาที่ตัวแปร resultCheckLogisticsStatus และ resultFindDuration ในกระบวนการบีเพล

2.3) InventoryStockService จะเริ่มต้นตรวจสอบสถานะของคลังสินค้าว่าเป็นสถานะ StockIn StockOut และ OnGoing หรือไม่ จากนั้นบริการตรวจสอบต่อว่ามีสินค้าในคลังสินค้าหรือไม่ตามใบสั่งซื้อสินค้า โดยส่งผลลัพธ์มาที่ตัวแปร resultCheckStockStatus และ resultCheckPartsInStock ในกระบวนการบีเพล

3) เมื่อทั้ง 3 บริการตรวจสอบตรวจสอบข้อมูลใบสั่งซื้อสินค้าเรียบร้อย จากนั้นจึงพิจารณาตัวแปร resultCheckPOStatus resultCheckLogisticsStatus และ resultCheckStockStatus ว่าถูกต้องหรือไม่



ภาพที่ 5.9 แผนภาพกิจกรรมของโปรแกรม PurchaseOrderProcess

5.4 กรณีทดสอบที่ใช้ทดสอบกับโปรแกรมต้นแบบ

ในการตรวจสอบเครื่องมือทดสอบวีคมิวเทชันสำหรับดับบิลยูเอสบีเพลว่าทำงานได้ถูกต้องหรือไม่ จึงจำเป็นต้องใช้กรณีทดสอบที่เตรียมไว้ในตารางที่ 5.1 – 5.9 สำหรับโปรแกรมหาชนิดของสามเหลี่ยม โปรแกรมเครื่องคิดเลข โปรแกรมการอนุมัติการกู้ยืม โปรแกรมเอทีเอ็ม โปรแกรมการซื้อสินค้า โปรแกรมการจองตั๋วการท่องเที่ยว โปรแกรมเพิ่มจำนวน โปรแกรมหาค่ากำลังสอง และโปรแกรมใบสั่งซื้อสินค้า ตามลำดับ

ตารางที่ 5.1 กรณีทดสอบของโปรแกรมหาชนิดของสามเหลี่ยม

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	50,50,1	Isosceles
2	50,50,2	Isosceles
3	50,50,50	Equilateral
4	50,50,99	Isosceles
5	50,50,100	Impossible
6	50,1,50	Isosceles
7	50,2,50	Isosceles
8	50,99,50	Isosceles
9	50,100,50	Impossible
10	1,50,50	Isosceles
11	2,50,50	Isosceles
12	99,50,50	Isosceles
13	100,50,50	Impossible

ตารางที่ 5.2 กรณีทดสอบของโปรแกรมเครื่องคิดเลข

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	10,5,+	15.0
2	100,99,-	1.0
3	6,8,*	48.0
4	7,3,/	2.3333333333333335

ตารางที่ 5.2 กรณีทดสอบของโปรแกรมเครื่องคิดเลข (ต่อ)

5	117,17,%	15.0
6	7,4,^	2401.0
7	50,2,!	Cannot calculate by using this operand
8	1000,101,min	101
9	99,500,max	500
10	456,16,nth	1.4661732872336828
11	255,13,%	8.0
12	111,3,^	1367631.0
13	45,45,#	Cannot calculate by using this operand
14	31,0,$	Cannot calculate by using this operand
15	13,3,nth	2.3513346877207573

ตารางที่ 5.3 กรณีทดสอบของโปรแกรมการอนุมัติการกู้ยืม

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	55000,A	true
2	320000,B	true
3	499999,C	true
4	500001,D	false
5	1000000,E	false

ตารางที่ 5.4 กรณีทดสอบของโปรแกรมเอทีเอ็ม

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	50000,iPatz,5432	Your receiving cash : 50 banknote(s) of 1000
2	20000,Bill,6489	Bill does not exist
3	45000,Gates,4579	This 4579 is not valid with user
4	60000,iPatz,5432	Your balance is not enough
5	100000,Peter,1112	Your receiving cash : 100 banknote(s) of 1000

ตารางที่ 5.5 กรณีทดสอบของโปรแกรมซื้อสินค้า

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	75000.0,iPhone3GS,4	You own iPhone3GS (15000.0 ฿) with total amount 60000.0 ฿ Your change is 15000 ฿
2	80000.0,iMac,1	You cannot buy iMac
3	46000.0,Galaxy S II,2	Galaxy S II is not available in our shop.
4	28000.0,iPhone4S,1	iPhone4S is now Out of Stock

ตารางที่ 5.6 กรณีทดสอบของโปรแกรมการจองตั๋วท่องเที่ยว

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	KR-608,100000,3,Korea,20.00	You Flight and Hotel reservation are success!!
2	US-667,57000,3,USA,13.00	US-667 is not available
3	JP-798,30000,2,Japan,10.0	Room in Hotel are not available.
4	US-667,95000,5,USA,3.0	You cash 95000฿ is not enough for travel
5	UK-554,375000,7,England,10.0	UK-554 is not available
6	JP-798,30000,5,Japan,10.30	You cash 30000฿ is not enough for travel

ตารางที่ 5.7 กรณีทดสอบของโปรแกรม AddNumber

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	1, 20	210.0
2	10, 20	165.0
3	20, 20	20.0
4	20, 10	0
5	20, 1	0

ตารางที่ 5.8 กรณีทดสอบของโปรแกรม SquareValue

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	99	319181.0
2	101	354193.0
3	0	0
4	-7	Error validating request
5	test data	Error validating request

ตารางที่ 5.9 กรณีทดสอบของโปรแกรม PurchaseOrderProcess

รหัส	ข้อมูลนำเข้า	ผลลัพธ์ที่คาดหวัง
1	1, PO201202010001, PI, 2012-02-01, 10, 26000, 1400,	PO Success
2	2, PO201205050002, PR, 2012-05-05, 20, 40000, 1000,	PO Success
3	3, PO201203260003, PO, 2012-03-26, 10, 450000, 1220,	PO Success
4	4, PO201204170004, PR, 2012-04-17, 100, 45000, 3450,	PO Success
5	5, PO201204290005, PR, 2012-04-29, 230, 250000, 7680,	PO Success

5.5 ผลของการทดสอบโปรแกรม

หลังจากที่ได้ทำการทดสอบโปรแกรมตัวอย่างทั้ง 9 โปรแกรมบีเพล ซึ่งผลของการทดสอบถูกแบ่งออกเป็น 4 แบบ คือ EX-WEAK/1 ST-WEAK/1 BB-WEAK/1 และ BB-WEAK/N ซึ่งกระบวนการวิเคราะห์แบบ BB-WEAK/1 และ BB-WEAK/N ที่ทดสอบกับโปรแกรมที่เตรียมไว้ได้ผลการทดลองเหมือนกัน ดังนั้นการวิเคราะห์จึงเป็นการวิเคราะห์แบบ BB-WEAK โดยในตารางที่ 5.10 – 5.18 จะเป็นการทดสอบโดยใช้วิธี EX-WEAK/1 ส่วนตารางที่ 5.19 – 5.27 จะเป็นผลการทดสอบโดยใช้วิธี ST-WEAK/1 และตารางที่ 5.28 – 5.36 เป็นผลการทดสอบโดยใช้วิธี BB-WEAK

5.5.1 ผลการทดสอบโปรแกรมหาชนิดของสามเหลี่ยม ซึ่งใช้กรณีทดสอบในตารางที่ 5.1 และได้ผลการทดสอบดังตารางที่ 5.10 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.10 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	114	6
EAA	60	6
ECC	0	0
ECN	12	0
ERR	120	21
EMD	0	0
EMF	0	0
ELL	19	6
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0

ตารางที่ 5.10 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย EX-WEAK/1 (ต่อ)

XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	325 (มิวแตนท์ทั้งหมด)	39 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	1891 วินาที	31.52 นาที
คะแนนมิวเทชัน	12 %	
ประสิทธิภาพของกรณีทดสอบ	66.06 %	

5.5.2 ผลการทดสอบโปรแกรมเครื่องคิดเลข ซึ่งได้ใช้กรณีทดสอบในตารางที่ 5.2 และได้ผลการทดสอบดังตารางที่ 5.11 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.11 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	4	4
EAA	12	12
ECC	0	0
ECN	0	0
ERR	55	40
EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0

ตารางที่ 5.11 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย EX-WEAK/1 (ต่อ)

AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	71 (มิวแตนท์ทั้งหมด)	56 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	648 วินาที	10.8 นาที
คะแนนมิวเทชัน	78.87 %	
ประสิทธิภาพของกรณีทดสอบ	78.87 %	

5.5.3 ผลการทดสอบโปรแกรมการอนุมัติการกู้ยืม ได้ใช้กรณีทดสอบในตารางที่ 5.3 และได้ผลของการทดสอบในตารางที่ 5.12 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.12 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	14	0

ตารางที่ 5.12 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย EX-WEAK/1 (ต่อ)

ERR	40	23
EMD	0	0
EMF	0	0
ELL	1	1
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	55 (มิวแตนท์ทั้งหมด)	24 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	316 วินาที	5.27 นาที
คะแนนมิวเทชัน	43.64 %	
ประสิทธิภาพของกรณีทดสอบ	79.64 %	

5.5.4 ผลการทดสอบโปรแกรมเอทีเอ็ม ได้ใช้กรณีทดสอบในตารางที่ 5.4 และได้ผลของการทดลองดังตารางที่ 5.13 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.13 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	15	3
EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0

ตารางที่ 5.13 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย EX-WEAK/1 (ต่อ)

XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	15 (มิวแตนท์ทั้งหมด)	3 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	23 วินาที	0.38 นาที
คะแนนมิวเทชัน	20.00 %	
ประสิทธิภาพของกรณีทดสอบ	40.00 %	

5.5.5 ผลการทดสอบโปรแกรมการซื้อสินค้า ได้ใช้กรณีทดสอบในตารางที่ 5.5 และได้ผลของการทดลองดังตารางที่ 5.14 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.14 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	25	3
EMD	0	0
EMF	0	0
ELL	2	2
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0

ตารางที่ 5.14 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย EX-WEAK/1 (ต่อ)

AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	27 (มิวแตนท์ทั้งหมด)	5 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	154 วินาที	2.57 นาที
คะแนนมิวเทชัน	18.52 %	
ประสิทธิภาพของกรณีทดสอบ	44.44 %	

5.5.6 ผลการทดสอบโปรแกรมการจองตั๋วการท่องเที่ยว ได้ใช้กรณีทดสอบในตารางที่ 5.6 และได้ผลของการทดลองในตารางที่ 5.15 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.15 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0

ตารางที่ 5.15 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย EX-WEAK/1 (ต่อ)

ECN	0	0
ERR	30	2
EMD	0	0
EMF	0	0
ELL	2	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	32 (มิวแตนท์ทั้งหมด)	2 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	534 วินาที	8.9 นาที
คะแนนมิวเทชัน	6.25 %	

ตารางที่ 5.15 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย EX-WEAK/1 (ต่อ)

ประสิทธิภาพของกรณีทดสอบ	12.50 %

5.5.7 ผลการทดสอบโปรแกรมเพิ่มค่าตัวเลข ซึ่งใช้กรณีทดสอบในตารางที่ 5.7 และได้ผลการทดสอบดังตารางที่ 5.16 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.16 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	12	10
ECC	0	0
ECN	10	4
ERR	25	5
EMD	0	0
EMF	0	0
ELL	3	1
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0

ตารางที่ 5.16 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย EX-WEAK/1 (ต่อ)

XEE	0	0
XER	0	0
XMC	0	0
XMT	0	0
XMF	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	50 (มิวแตนท์ทั้งหมด)	3 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	17 วินาที	0.28 นาที
คะแนนมิวเทชัน	40.00 %	
ประสิทธิภาพของกรณีทดสอบ	38.00 %	

5.5.8 ผลการทดสอบโปรแกรมหาค่ากำลังสอง ซึ่งใช้กรณีทดสอบในตารางที่ 5.8 และได้ผลการทดสอบดังตารางที่ 5.17 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.17 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	68	54
ECC	13	13
ECN	48	44
ERR	0	0
EMD	0	0
EMF	0	0
ELL	0	0
EEU	2	2
ACI	0	0

ตารางที่ 5.17 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย EX-WEAK/1 (ต่อ)

AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMT	0	0
XMF	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	131 (มิวแตนท์ทั้งหมด)	113 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	50 วินาที	0.83 นาที
คะแนนมิวเทชัน	86.26 %	
ประสิทธิภาพของกรณีทดสอบ	66.41 %	

5.5.9 ผลการทดสอบโปรแกรมใบสั่งซื้อสินค้า ซึ่งใช้กรณีทดสอบในตารางที่ 5.9 และได้ผลการทดสอบดังตารางที่ 5.18 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ EX-WEAK/1

ตารางที่ 5.18 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย EX-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	8	8
ECC	0	0
ECN	38	37
ERR	90	7
EMD	2	0
EMF	1	1
ELL	10	0
EEU	2	2
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMT	0	0
XMF	0	0

ตารางที่ 5.18 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย EX-WEAK/1 (ต่อ)

XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	151(มิวแตนท์ทั้งหมด)	55 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	84 วินาที	1.4 นาที
คะแนนมิวเทชัน	36.42 %	
ประสิทธิภาพของกรณีทดสอบ	34.44 %	

5.5.10 ผลการทดสอบโปรแกรมหาชนิดของสามเหลี่ยม ซึ่งใช้กรณีทดสอบในตารางที่ 5.1 และได้ผลการทดสอบดังตารางที่ 5.19 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.19 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	114	26
EAA	60	60
ECC	0	0
ECN	12	5
ERR	120	62
EMD	0	0
EMF	0	0
ELL	19	4
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	2	2
AJC	0	0
AIS	0	0

ตารางที่ 5.19 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย ST-WEAK/1 (ต่อ)

APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	327 (มิวแตนท์ทั้งหมด)	159 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	1914 วินาที	31.9 นาที
คะแนนมิวเทชัน	48.62 %	
ประสิทธิภาพของกรณีทดสอบ	92 %	

5.5.11 ผลการทดสอบโปรแกรมเครื่องคิดเลข ซึ่งได้ใช้กรณีทดสอบในตารางที่ 5.2 และได้ผลการทดสอบดังตารางที่ 5.20 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.20 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	4	4
EAA	12	12
ECC	0	0
ECN	0	0
ERR	55	40

ตารางที่ 5.20 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย ST-WEAK/1 (ต่อ)

EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	6	6
AJC	0	0
AIS	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0

สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	71 (มิวแตนท์ทั้งหมด)	62 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	544 วินาที	9.07 นาที
คะแนนมิวเทชัน	80.52 %	
ประสิทธิภาพของกรณีทดสอบ	79.13 %	

5.5.12 ผลการทดสอบโปรแกรมการอนุมัติการกู้ยืม ได้ใช้กรณีทดสอบในตารางที่ 5.3 และได้ผลของการทดสอบในตารางที่ 5.21 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.21 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	14	2
ERR	40	25
EMD	0	0
EMF	0	0
ELL	1	1
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	3	3
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0

ตารางที่ 5.21 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย ST-WEAK/1 (ต่อ)

XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	58 (มิวแตนท์ทั้งหมด)	31 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	316 วินาที	5.27 นาที
คะแนนมิวเทชัน	53.45 %	
ประสิทธิภาพของกรณีทดสอบ	79.31 %	

5.5.13 ผลการทดสอบโปรแกรมเอทีเอ็ม ได้ใช้กรณีทดสอบในตารางที่ 5.4 และได้ผลของการทดลองดังตารางที่ 5.22 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.22 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	15	3
EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	2	2

ตารางที่ 5.22 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย ST-WEAK/1 (ต่อ)

AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	17 (มิวแตนท์ทั้งหมด)	5 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	13 วินาที	0.22 นาที
คะแนนมิวเทชัน	29.41 %	
ประสิทธิภาพของกรณีทดสอบ	35.29 %	

5.5.14 ผลการทดสอบโปรแกรมการซื้อสินค้า ได้ใช้กรณีทดสอบในตารางที่ 5.5 และได้ผลของการทดลองดังตารางที่ 5.23 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.23 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0

ตารางที่ 5.23 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย ST-WEAK/1 (ต่อ)

ECN	0	0
ERR	25	3
EMD	0	0
EMF	0	0
ELL	2	2
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	29 (มิวแตนท์ทั้งหมด)	7 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	164 วินาที	2.73 นาที
คะแนนมิวเทชัน	24.14 %	

ตารางที่ 5.23 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย ST-WEAK/1 (ต่อ)

ประสิทธิภาพของกรณีทดสอบ	42.24 %

5.5.15 ผลการทดสอบโปรแกรมการจองตั๋วการท่องเที่ยว ได้ใช้กรณีทดสอบในตารางที่ 5.6 และได้ผลของการทดลองในตารางที่ 5.24 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.24 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	30	2
EMD	0	0
EMF	0	0
ELL	2	0
EEU	0	0
ACI	0	0
AEL	0	0
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0

ตารางที่ 5.24 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย ST-WEAK/1 (ต่อ)

XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	34 (มิวแตนท์ทั้งหมด)	4 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	466 วินาที	7.77 นาที
คะแนนมิวเทชัน	11.76 %	
ประสิทธิภาพของกรณีทดสอบ	12.75 %	

5.5.16 ผลการทดสอบโปรแกรมเพิ่มค่าตัวเลข ได้ใช้กรณีทดสอบในตารางที่ 5.7 และได้ผลของการทดลองในตารางที่ 5.25 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.25 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	12	10
ECC	0	0
ECN	10	4
ERR	25	5
EMD	0	0
EMF	0	0
ELL	3	1
EEU	0	0
ACI	0	0

ตารางที่ 5.25 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย ST-WEAK/1 (ต่อ)

AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	50 (มิวแตนท์ทั้งหมด)	20 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	9 วินาที	0.15 นาที
คะแนนมิวเทชัน	40.00 %	
ประสิทธิภาพของกรณีทดสอบ	38.00 %	

5.5.17 ผลการทดสอบโปรแกรมหาค่ากำลังสอง ได้ใช้กรณีทดสอบในตารางที่ 5.8 และได้ผลของการทดลองในตารางที่ 5.26 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.26 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย ST-WEAK/1 (ต่อ)

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	68	54
ECC	13	13
ECN	48	48
ERR	0	0
EMD	0	0
EMF	0	0
ELL	0	0
EEU	2	2
ACI	0	0
AEL	0	0
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0

ตารางที่ 5.26 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย ST-WEAK/1 (ต่อ)

XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	131 (มิวแตนท์ทั้งหมด)	117 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	40 วินาที	0.67 นาที
คะแนนมิวเทชัน	89.31 %	
ประสิทธิภาพของกรณีทดสอบ	66.41 %	

5.5.18 ผลการทดสอบโปรแกรมใบสั่งซื้อสินค้า ได้ใช้กรณีทดสอบในตารางที่ 5.9 และได้ผลของการทดลองในตารางที่ 5.27 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ ST-WEAK/1

ตารางที่ 5.27 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย ST-WEAK/1

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	8	8
ECC	0	0
ECN	38	37
ERR	90	9
EMD	2	0
EMF	1	1
ELL	10	1
EEU	2	2
ACI	0	0
AEL	0	0
AFP	0	0
AIE	4	4
AIS	0	0
AJC	1	1

ตารางที่ 5.27 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย ST-WEAK/1 (ต่อ)

APA	3	3
APM	3	3
ASF	0	0
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	162 (มิวแตนท์ทั้งหมด)	69 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	95 วินาที	1.58 นาที
คะแนนมิวเทชัน	42.59 %	
ประสิทธิภาพของกรณีทดสอบ	32.10 %	

5.5.19 ผลการทดสอบโปรแกรมหาชนิดของสามเหลี่ยม ซึ่งใช้กรณีทดสอบในตารางที่ 5.1 และได้ผลการทดสอบดังตารางที่ 5.28 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.28 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	114	64
EAA	60	6
ECC	0	0
ECN	12	0
ERR	120	21

ตารางที่ 5.28 สรุปผลการทดสอบของโปรแกรมหาชนิดของสามเหลี่ยมด้วย BB-WEAK (ต่อ)

EMD	0	0
EMF	0	0
ELL	19	11
EEU	0	0
ACI	0	0
AEL	7	7
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0

สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	337 (มิวแตนท์ทั้งหมด)	112 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	1981 วินาที	33.02 นาที
คะแนนมิวเทชัน	33.43 %	
ประสิทธิภาพของกรณีทดสอบ	19.66 %	

5.5.20 ผลการทดสอบโปรแกรมเครื่องคิดเลข ซึ่งได้ใช้กรณีทดสอบในตารางที่ 5.2 และได้ผลการทดสอบดังตารางที่ 5.29 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.29 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	4	4
EAA	12	12
ECC	0	0
ECN	0	0
ERR	55	40
EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	18	18
AFP	0	0
AIE	6	6
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0
XEE	0	0
XER	2	0
XMC	0	0

ตารางที่ 5.29 สรุปผลการทดสอบของโปรแกรมเครื่องคิดเลขด้วย BB-WEAK (ต่อ)

XMF	2	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	100 (มิวแตนท์ทั้งหมด)	81 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	1693 วินาที	28.22 นาที
คะแนนมิวเทชัน	81 %	
ประสิทธิภาพของกรณีทดสอบ	64.93 %	

5.5.21 ผลการทดสอบโปรแกรมการอนุมัติการกู้ยืม ได้ใช้กรณีทดสอบในตารางที่ 5.3 และได้ผลของการทดสอบในตารางที่ 5.30 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.30 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	14	0
ERR	40	23
EMD	0	0
EMF	0	0
ELL	1	1
EEU	0	0
ACI	0	0
AEL	17	17
AFP	0	0
AIE	3	3

ตารางที่ 5.30 สรุปผลการทดสอบของโปรแกรมการอนุมัติการกู้ยืมด้วย BB-WEAK (ต่อ)

AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0
XEE	0	0
XER	6	6
XMC	3	3
XMF	6	6
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	91 (มิวแตนท์ทั้งหมด)	60 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	386 วินาที	6.43 นาที
คะแนนมิวเทชัน	65.93 %	
ประสิทธิภาพของกรณีทดสอบ	80.88 %	

5.5.22 ผลการทดสอบโปรแกรมเอทีเอ็ม ได้ใช้กรณีทดสอบในตารางที่ 5.4 และได้ผลของการทดลองดังตารางที่ 5.31 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.31 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0

ตารางที่ 5.31 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย BB-WEAK (ต่อ)

ECN	0	0
ERR	15	3
EMD	0	0
EMF	0	0
ELL	0	0
EEU	0	0
ACI	0	0
AEL	13	13
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0
XEE	0	0
XER	4	4
XMC	2	2
XMF	4	4
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	41 (มิวแตนท์ทั้งหมด)	29 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	37 วินาที	0.62 นาที
คะแนนมิวเทชัน	70.73 %	

ตารางที่ 5.31 สรุปผลการทดสอบของโปรแกรมเอทีเอ็มด้วย BB-WEAK (ต่อ)

ประสิทธิภาพของกรณีทดสอบ	14.63 %

5.5.23 ผลการทดสอบโปรแกรมการซื้อสินค้า ได้ใช้กรณีทดสอบในตารางที่ 5.5 และได้ผลของการทดลองดังตารางที่ 5.32 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.32 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	25	3
EMD	0	0
EMF	0	0
ELL	2	2
EEU	0	0
ACI	0	0
AEL	12	12
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0

ตารางที่ 5.32 สรุปผลการทดสอบของโปรแกรมการซื้อสินค้าด้วย BB-WEAK (ต่อ)

XEE	0	0
XER	3	3
XMC	2	2
XMF	4	4
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	51 (มิวแตนท์ทั้งหมด)	29 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	203 วินาที	3.38 นาที
คะแนนมิวเทชัน	56.86 %	
ประสิทธิภาพของกรณีทดสอบ	24.02 %	

5.5.24 ผลการทดสอบโปรแกรมการจองตั๋วการท่องเที่ยว ได้ใช้กรณีทดสอบในตารางที่ 5.6 และได้ผลของการทดลองในตารางที่ 5.33 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.33 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	0	0
ECC	0	0
ECN	0	0
ERR	30	2
EMD	0	0
EMF	0	0
ELL	2	0
EEU	0	0
ACI	0	0

ตารางที่ 5.33 สรุปผลการทดสอบของโปรแกรมการจองตั๋วการท่องเที่ยวด้วย BB-WEAK (ต่อ)

AEL	14	14
AFP	0	0
AIE	2	2
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	0	0
XEE	0	0
XER	4	4
XMC	2	2
XMF	4	4
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	59 (มิวแตนท์ทั้งหมด)	29 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	585 วินาที	9.75 นาที
คะแนนมิวเทชัน	49.15 %	
ประสิทธิภาพของกรณีทดสอบ	10.17 %	

5.5.25 ผลการทดสอบโปรแกรมเพิ่มค่าตัวเลข ได้ใช้กรณีทดสอบในตารางที่ 5.7 และได้ผลของการทดลองในตารางที่ 5.34 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.34 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	12	10
ECC	0	0
ECN	10	4
ERR	25	5
EMD	0	0
EMF	0	0
ELL	3	1
EEU	0	0
ACI	0	0
AEL	5	5
AFP	0	0
AIE	0	0
AIS	0	0
AJC	0	0
APA	0	0
APM	0	0
ASF	1	1
ASI	0	0
AWR	1	1
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0

ตารางที่ 5.34 สรุปผลการทดสอบของโปรแกรมเพิ่มค่าตัวเลขด้วย BB-WEAK (ต่อ)

XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	57 (มิวแตนท์ทั้งหมด)	27 (มิวแตนท์ที่ถูกกำจัด)
เวลาที่ใช้ในการทดสอบ	11 วินาที	0.18 นาที
คะแนนมิวเทชัน	47.37 %	
ประสิทธิภาพของกรณีทดสอบ	33.33 %	

5.5.26 ผลการทดสอบโปรแกรมหาค่ากำลังสอง ได้ใช้กรณีทดสอบในตารางที่ 5.8 และได้ผลของการทดลองในตารางที่ 5.35 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.35 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	68	54
ECC	13	13
ECN	48	44
ERR	0	0
EMD	0	0
EMF	0	0
ELL	0	0
EEU	2	2
ACI	0	0
AEL	9	9
AFP	2	2
AIE	0	0
AIS	0	0
AJC	0	0

ตารางที่ 5.35 สรุปผลการทดสอบของโปรแกรมหาค่ากำลังสองด้วย BB-WEAK (ต่อ)

APA	0	0
APM	0	0
ASF	1	1
ASI	8	8
AWR	1	1
XEE	0	0
XER	0	0
XMC	0	0
XMF	0	0
XMT	0	0
XTF	0	0
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	152 (มิวแตนท์ทั้งหมด)	134 (มิวแตนท์ที่กำจัดได้)
เวลาที่ใช้ในการทดสอบ	69 วินาที	1.15 นาที
คะแนนมิวเทชัน	88.16 %	
ประสิทธิภาพของกรณีทดสอบ	57.24 %	

5.5.27 ผลการทดสอบโปรแกรมใบสั่งซื้อสินค้า ได้ใช้กรณีทดสอบในตารางที่ 5.9 และได้ผลของการทดลองในตารางที่ 5.36 โดยใช้วิธีทดสอบวีคมิวเทชันแบบ BB-WEAK

ตารางที่ 5.36 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย BB-WEAK

สรุปผลของตัวดำเนินการมิวเทชัน		
ตัวดำเนินการมิวเทชัน	**จำนวนมิวแตนท์**	**จำนวนมิวแตนท์ที่ถูกกำจัด**
ISV	0	0
EAA	8	8
ECC	0	0
ECN	38	37
ERR	90	7

ตารางที่ 5.36 สรุปผลการทดสอบของโปรแกรมใบสั่งซื้อสินค้าด้วย BB-WEAK (ต่อ)

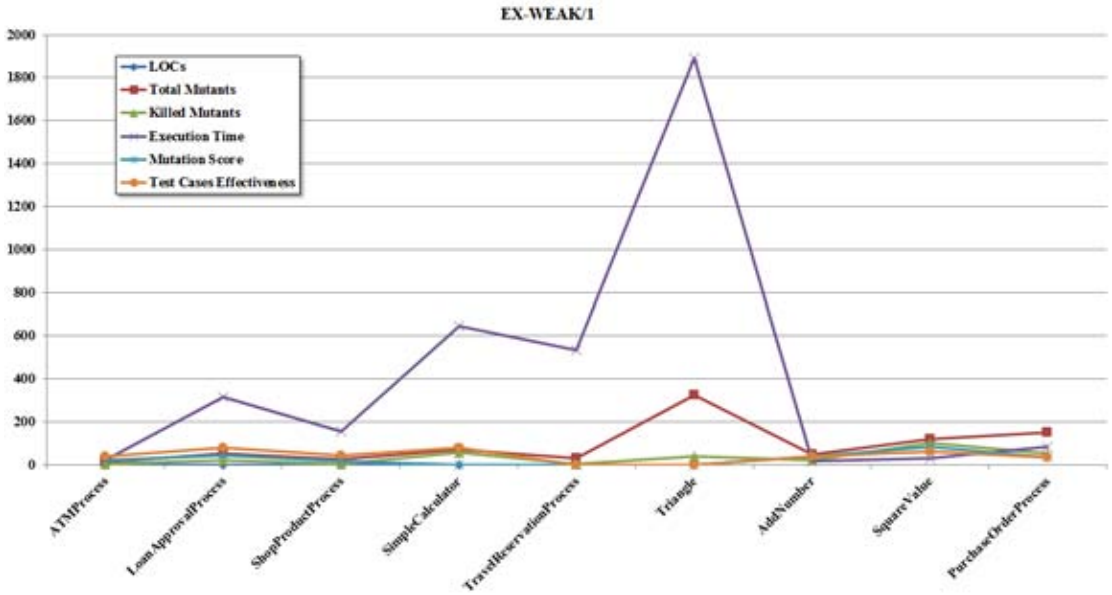
EMD	2	0
EMF	1	1
ELL	10	0
EEU	2	2
ACI	0	0
AEL	25	25
AFP	0	0
AIE	4	4
AIS	0	0
AJC	1	1
APA	3	3
APM	3	3
ASF	7	7
ASI	0	0
AWR	1	1
XEE	2	2
XER	2	2
XMC	2	2
XMF	2	2
XMT	2	2
XTF	18	18
สรุปผลการทดลอง		
จำนวนมิวแตนท์ทั้งหมด	223 (มิวแตนท์ทั้งหมด)	127 (มิวแตนท์ที่กำจัดได้)
เวลาที่ใช้ในการทดสอบ	59 วินาที	0.98 นาที
คะแนนมิวเทชัน	56.95 %	
ประสิทธิภาพของกรณีทดสอบ	23.32 %	

5.6 สรุปผลการทดสอบโปรแกรม

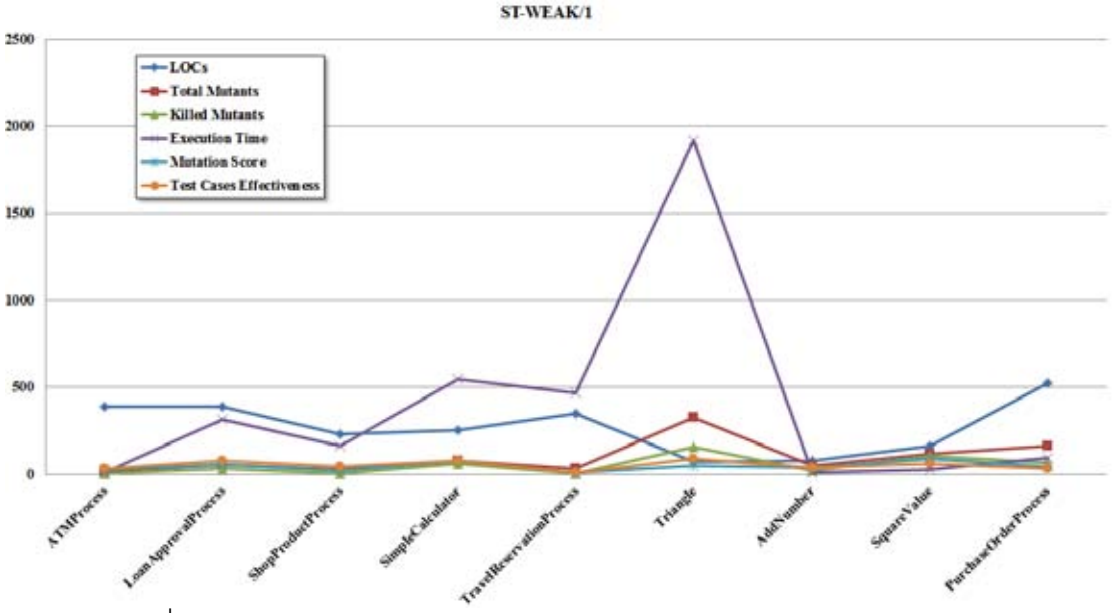
หลังจากการทดสอบโปรแกรมด้วยกรณีทดสอบที่เตรียมไว้ และได้ผลการทดลองในตารางที่ 5.10 – 5.36 ซึ่งแบ่งตามวิธีการทดสอบโดยแบ่งเป็น 3 ชนิด คือ EX-WEAK/1 ST-WEAK/1 และ BB-WEAK โดยทดสอบทุกโปรแกรมกับทุกวิธีการทดสอบ จะเห็นได้ว่าเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลสามารถสร้างมิวแตนท์ในแต่ละตัวดำเนินการได้ และสามารถแสดงผลของการทดสอบได้อย่างถูกต้อง นอกจากนี้เครื่องมือยังสามารถรายงานผลจำนวนมิวแตนท์ที่สร้างขึ้นในแต่ละตัวดำเนินการมิวเทชัน จำนวนมิวแตนท์ที่ถูกกำจัด เวลาที่ใช้ในการทดสอบโปรแกรม คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

จากภาพที่ 5.7 – 5.9 เป็นกราฟสรุปรวมทั้ง 9 โปรแกรมโดยแยกแนววิธีการทดสอบเป็น EX-WEAK/1 ST-WEAK/1 และ BB-WEAK ซึ่งลักษณะกราฟจะมีลักษณะคล้ายกัน ซึ่งสังเกตได้ว่าระยะเวลาการทดสอบมิวแตนท์ของโปรแกรมหาชนิดสามเหลี่ยม สิ้นเปลืองเวลาในการทดสอบมากซึ่งอยู่ในหน่วยวินาที เนื่องจากในโปรแกรมประกอบด้วยนิพจน์และประพจน์ในการตรวจสอบหาชนิดสามเหลี่ยมมาก ในขณะที่โปรแกรมที่เหลือมีนิพจน์หรือประพจน์ในการตรวจสอบไม่มากนัก

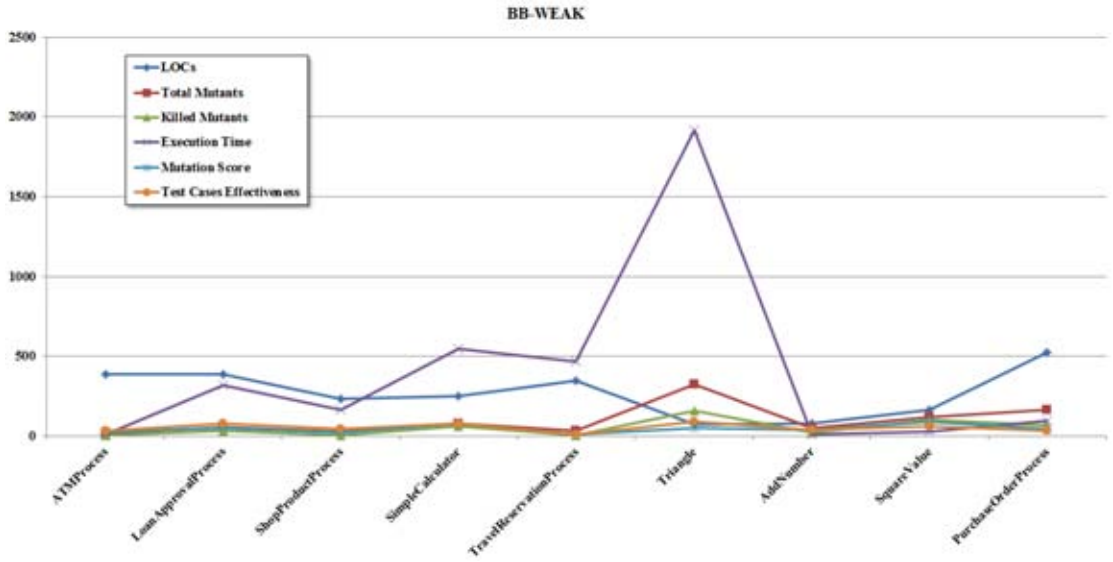
สำหรับคะแนนมิวเทชันและประสิทธิภาพของกรณีทดสอบค่าที่ได้จะอยู่ระหว่าง 0 – 100 เนื่องจากค่าที่คำนวนได้อยู่ในรูปแบบเปอร์เซนต์



ภาพที่ 5.10 กราฟผลการทดลองด้วยวิธีการทดสอบวีคมิวเทชันแบบ EX-WEAK/1



ภาพที่ 5.11 กราฟผลการทดลองด้วยวิธีการทดสอบวีคมิวเทชันแบบ ST-WEAK/1

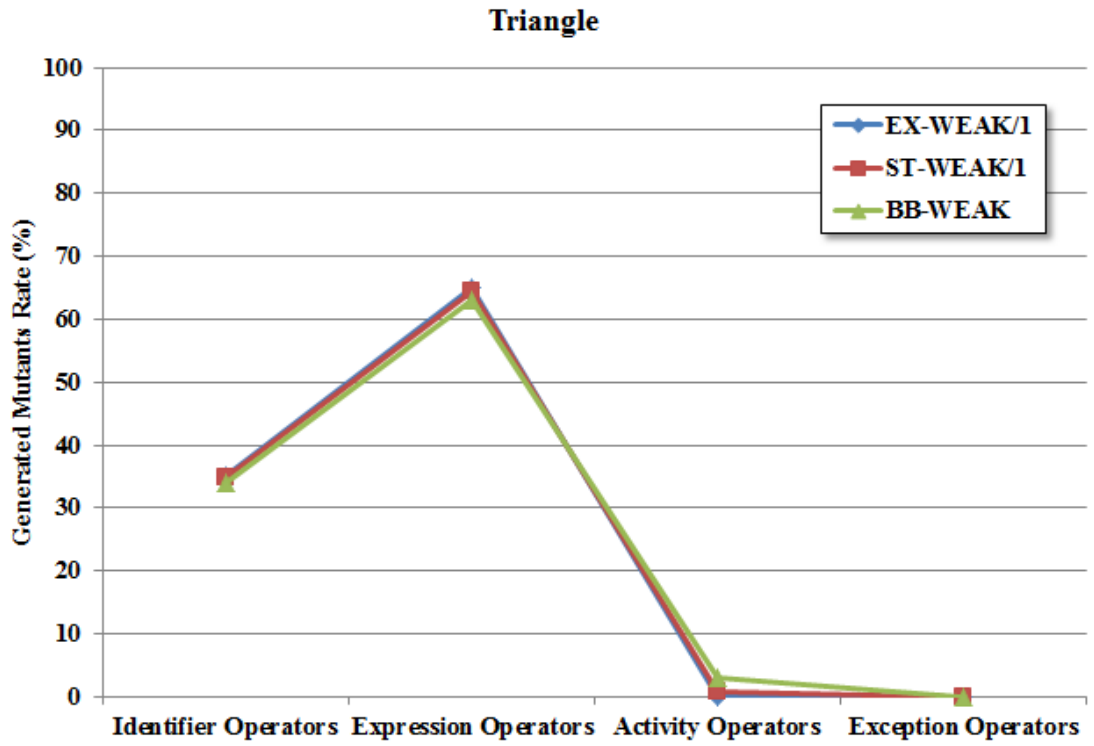


ภาพที่ 5.12 กราฟผลการทดลองด้วยวิธีการทดสอบวีคมิวเทชันแบบ BB-WEAK

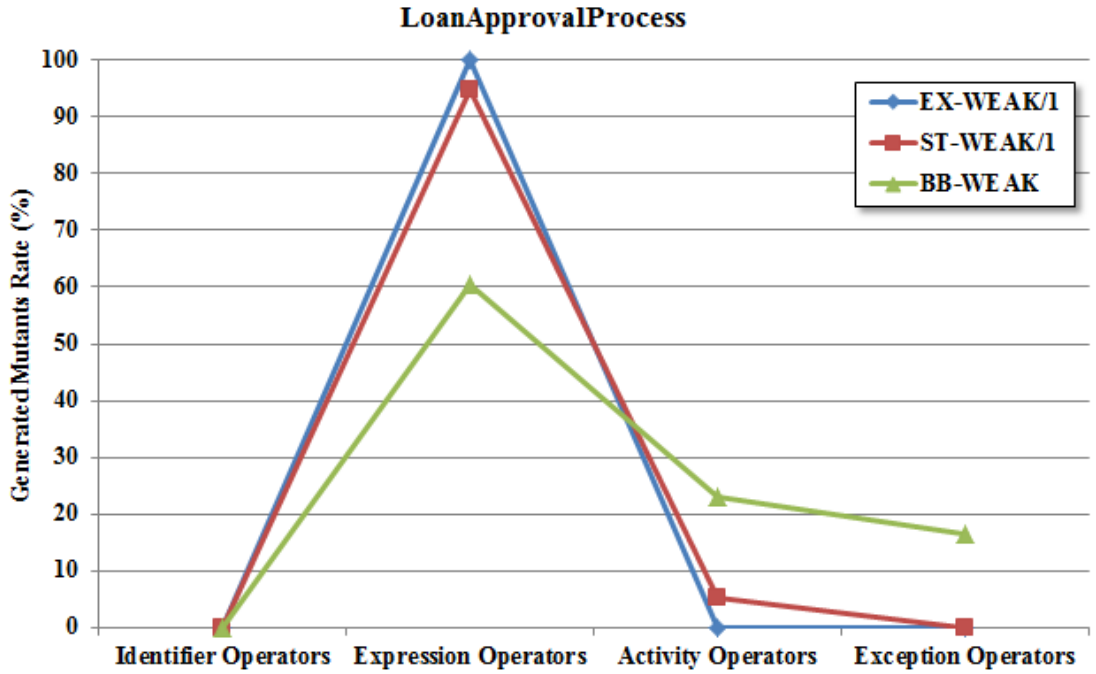
จากภาพที่ 5.13 – 5.21 แสดงอัตราส่วนการสร้างมิวแตนท์เป็นเปอร์เซนต์ในแกนตั้ง ส่วนในแกนนอนเป็นชนิดของตัวดำเนินการมิวเทชัน ซึ่งมีตัวดำเนินการมิวเทชันชนิดตัวระบุ ตัวดำเนินการมิวเทชันชนิดนิพจน์ ตัวดำเนินการมิวเทชันชนิดกิจกรรม และตัวดำเนินการมิวเทชันชนิดข้อผิดพลาด จากภาพที่ 5.13 – 5.21

แสดงให้เห็นว่ามิวแตนท์ที่ถูกสร้างขึ้นส่วนใหญ่มาจากตัวดำเนินการชนิดนิพจน์ ทั้งนี้เนื่องจากในกระบวนการบีเพลมีการใช้จำนวนนิพจน์และประพจน์เพื่อตรวจสอบตรรกะในโหนด from to หรือ condition มากกว่าจำนวนกิจกรรมที่มีในกระบวนการบีเพล

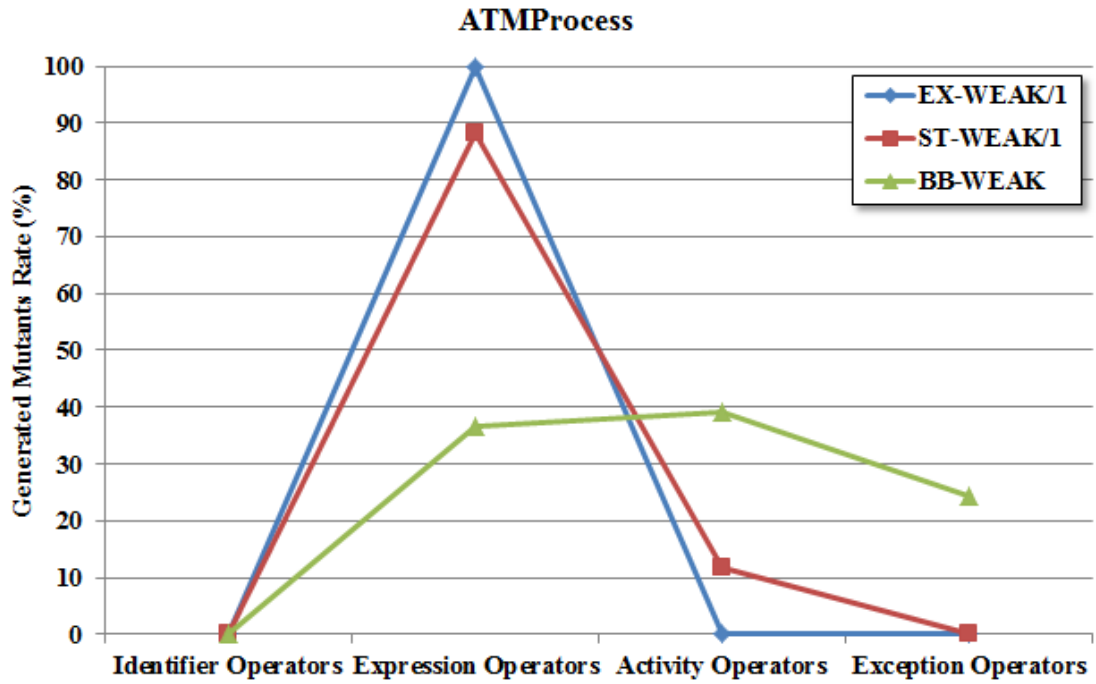
จากนั้นนำข้อมูลที่ได้จากกราฟมาสรุปเป็นตารางแสดงอัตราส่วนการสร้างมิวแตนท์ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ EX-WEAK/1 ST-WEAK/1 และ BB-WEAK ในตารางที่ 5.37 – 5.39



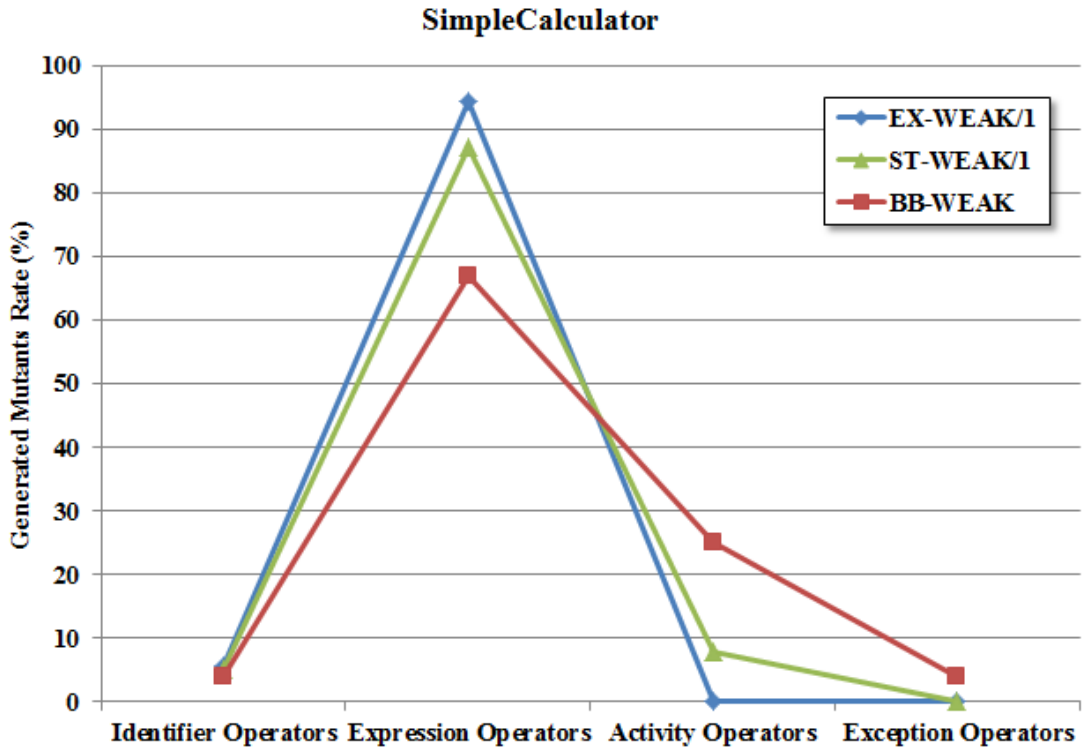
ภาพที่ 5.13 กราฟอัตราส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม Triangle



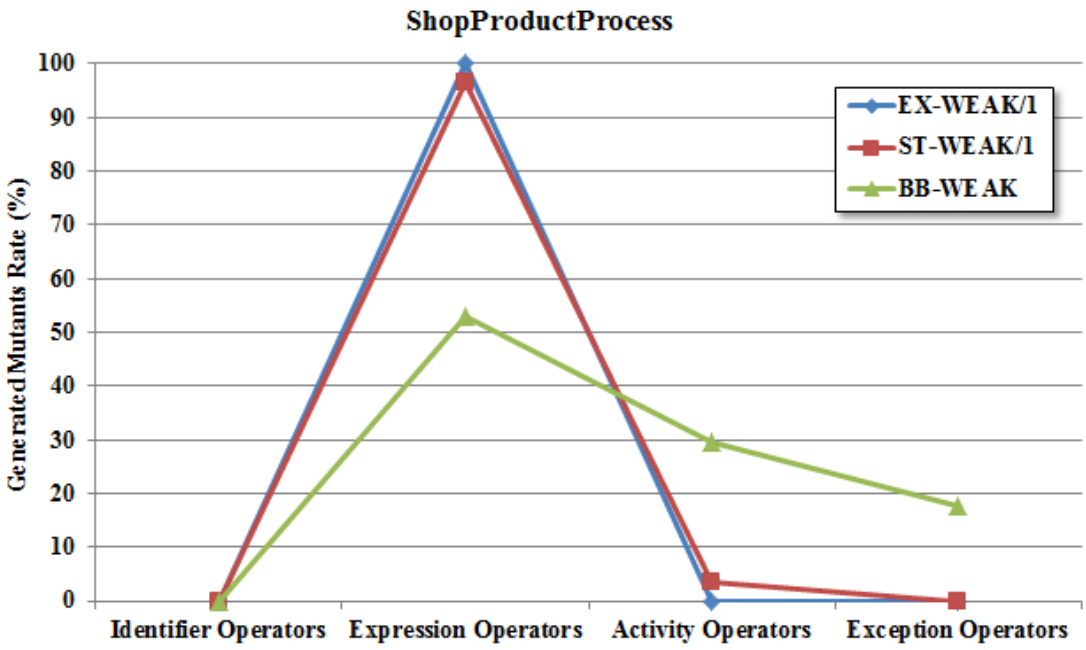
ภาพที่ 5.14 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม LoanApprovalProcess



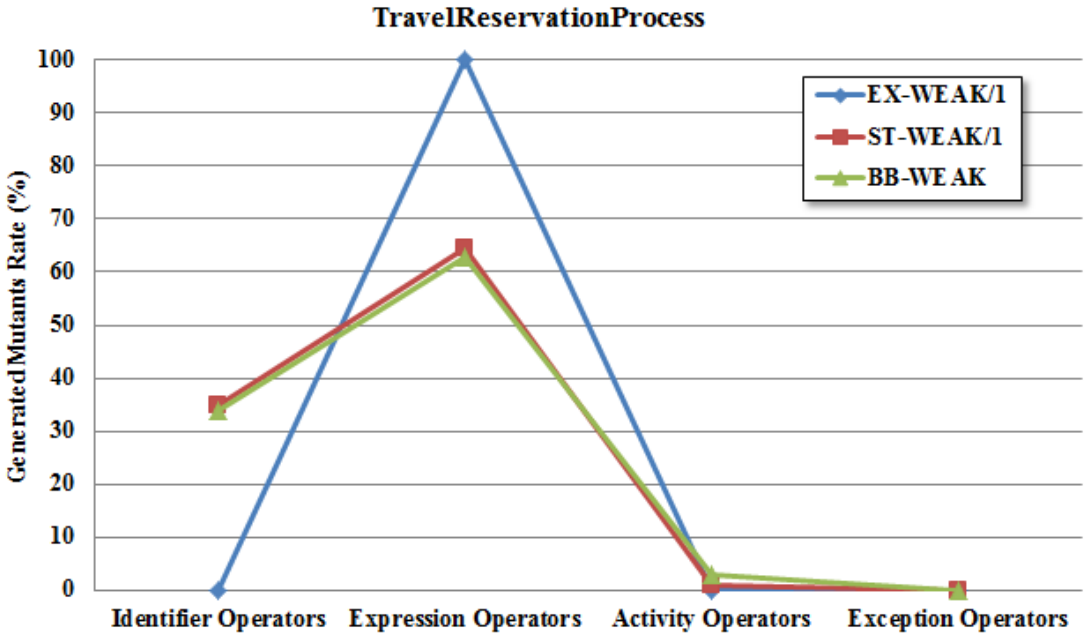
ภาพที่ 5.15 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ATMProcess



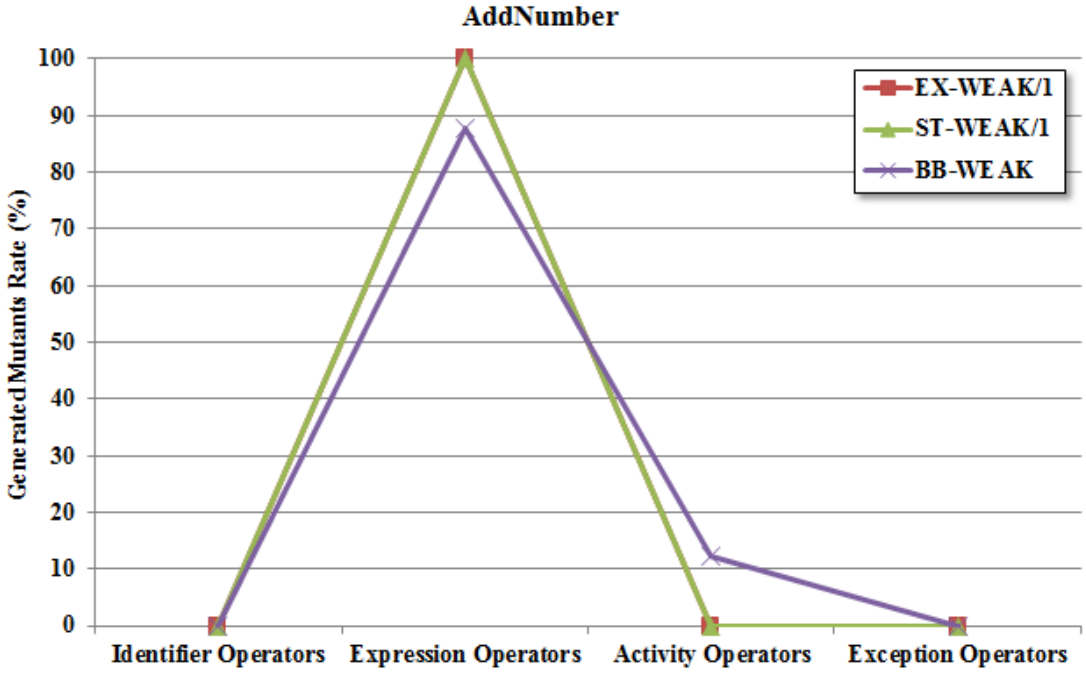
ภาพที่ 5.16 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SimpleCalculator



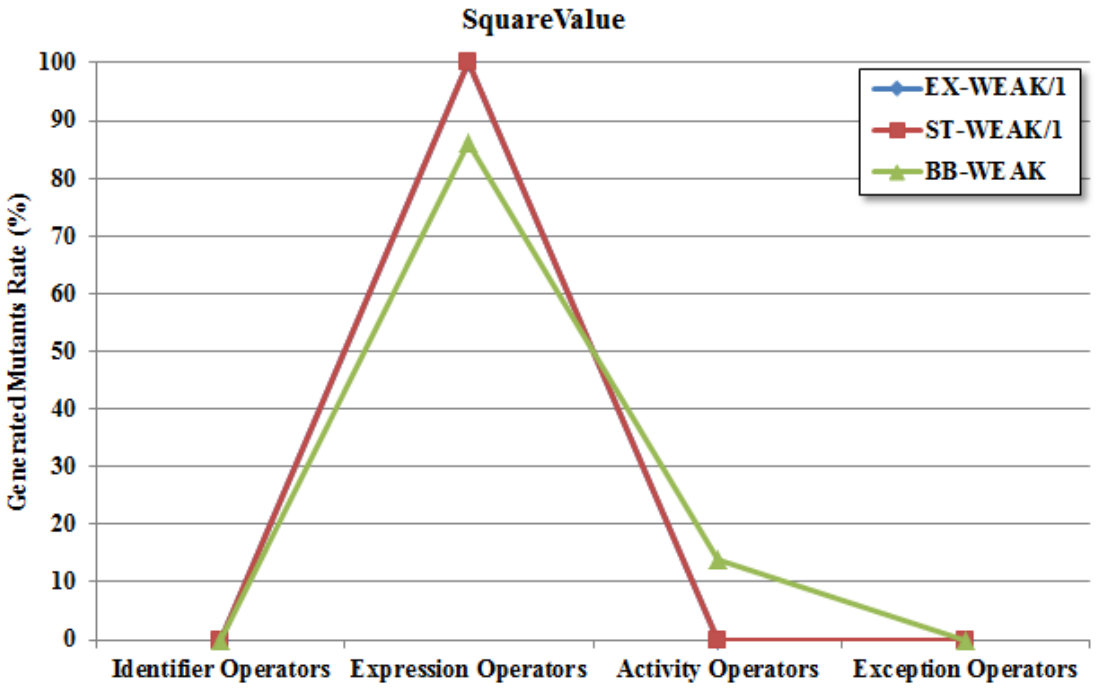
ภาพที่ 5.17 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ShopProductProcess



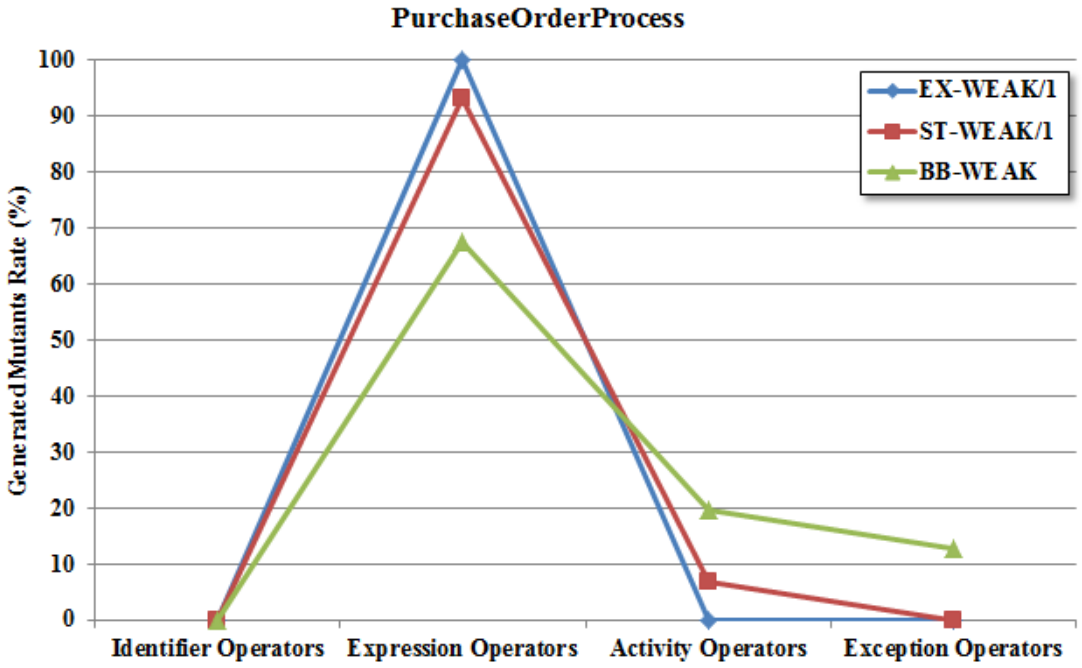
ภาพที่ 5.18 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม TravelReservationProcess



ภาพที่ 5.19 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม AddNumber



ภาพที่ 5.20 กราฟสัดส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SquareValue



ภาพที่ 5.21 กราฟอัตราส่วนการสร้างมิวแตนท์ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม PurchaseOrderProcess

ตารางที่ 5.37 อัตราส่วนการสร้างมิวแตนท์ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์ EX-WEAK/1

โปรแกรม	อัตราส่วนการสร้างมิวแตนท์ (%) ของ EX-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	35.08%	64.92%	0%	0%
อนุมัติการกู้ยืม	0%	100%	0%	0%
เอทีเอ็ม	0%	100%	0%	0%
เครื่องคิดเลข	5.63%	94.37%	0%	0%
การซื้อสินค้า	0%	100%	0%	0%
การจองตั๋วท่องเที่ยว	0%	100%	0%	0%
การเพิ่มจำนวน	0%	100%	0%	0%
การหาค่ากำลังสอง	0%	100%	0%	0%
ใบสั่งซื้อสินค้า	0%	100%	0%	0%

ตารางที่ 5.38 อัตราส่วนการสร้างมิวแตนท์ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์ ST-WEAK/1

โปรแกรม	อัตราส่วนการสร้างมิวแตนท์ (%) ของ ST-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	34.86%	64.53%	0.69%	0%
อนุมัติการกู้ยืม	0%	94.83%	5.17%	0%
เอทีเอ็ม	0%	88.24%	11.76%	0%
เครื่องคิดเลข	5.2%	87.01%	7.79%	0%
การซื้อสินค้า	0%	96.43%	3.57%	0%
การจองตั๋วท่องเที่ยว	0%	94.12%	5.88%	0%
การเพิ่มจำนวน	0%	100%	0%	0%
การหาค่ากำลังสอง	0%	100%	0%	0%
ใบสั่งซื้อสินค้า	0%	93.21%	6.79%	0%

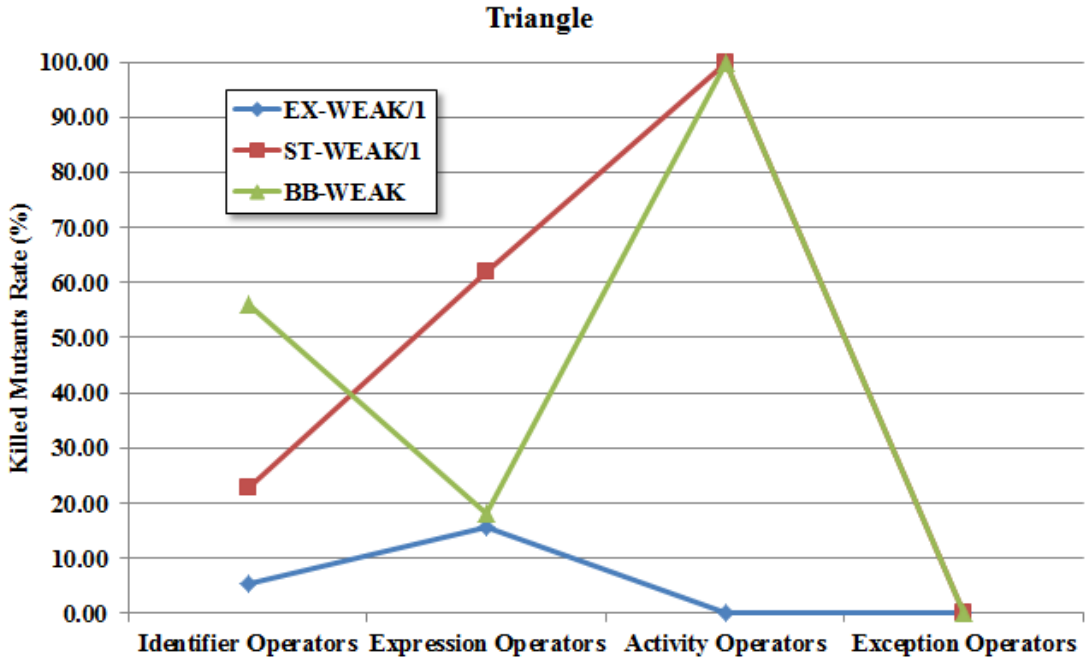
ตารางที่ 5.39 อัตราส่วนการสร้างมิวแตนท์ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์ BB-WEAK

โปรแกรม	อัตราส่วนการสร้างมิวแตนท์ (%) ของ BB-WEAK			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	34.03%	62.98%	2.98%	0%
อนุมัติการกู้ยืม	0%	60.44%	23.08%	16.48%
เอทีเอ็ม	0%	36.58%	39.02%	24.39%
เครื่องคิดเลข	4.00%	67.00%	25.00%	4.00%
การซื้อสินค้า	0%	52.94%	29.41%	17.65%
การจองตั๋วท่องเที่ยว	0%	54.24%	28.81%	16.95%
การเพิ่มจำนวน	0%	87.72%	12.28%	0%
การหาค่ากำลังสอง	0%	86.18%	13.82%	0%
ใบสั่งซื้อสินค้า	0%	67.71%	19.73%	12.56%

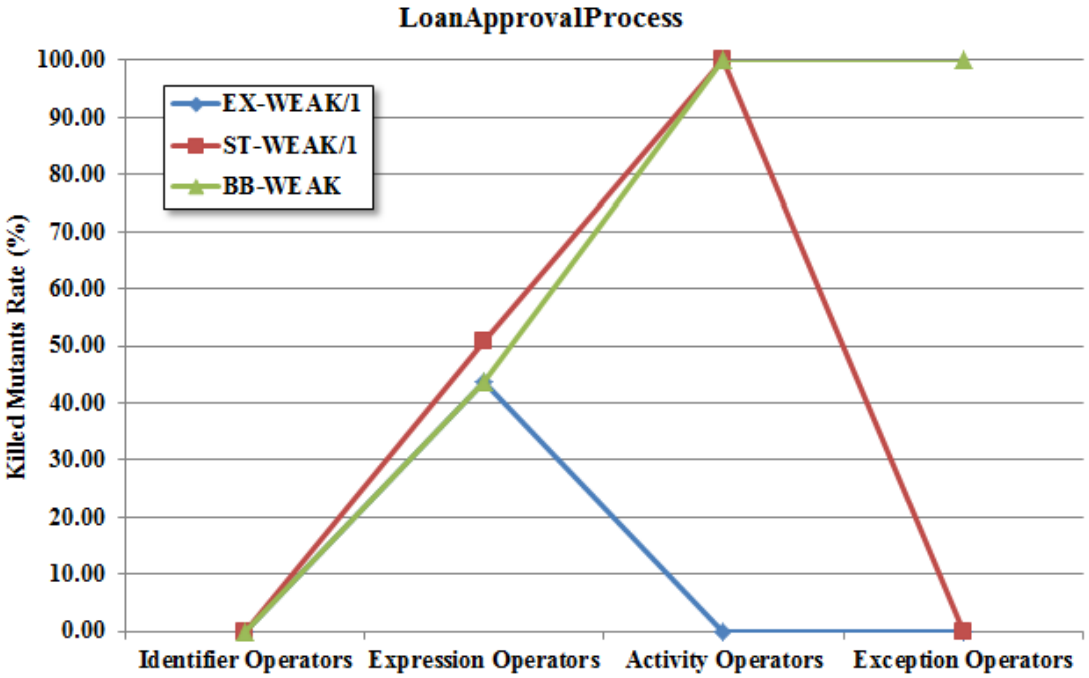
จากภาพที่ 5.22 – 5.30 แสดงอัตราส่วนมิวแตนท์ที่ถูกกำจัดเป็นรูปแบบเปอร์เซนต์ในแกนตั้ง ส่วนในแกนนอนเป็นชนิดของตัวดำเนินการมิวเทชัน ซึ่งมีตัวดำเนินการมิวเทชันชนิดตัวระบุ ตัวดำเนินการมิวเทชันชนิดนิพจน์ ตัวดำเนินการมิวเทชันชนิดกิจกรรม และตัวดำเนินการมิวเทชันชนิดข้อผิดพลาด

ดังภาพที่ 5.22 – 5.30 แสดงให้เห็นว่ามิวแตนท์ที่ถูกกำจัดส่วนใหญ่มาจากตัวดำเนินการชนิดกิจกรรมและข้อผิดพลาด ทั้งนี้เนื่องจากในกระบวนการบีเพลนั้นเมื่อถูกดัดแปลงค่าในระดับกิจกรรมหรือข้อผิดพลาด การดัดแปลงนั้นอาจทำให้ทั้งกระบวนบีเพลเกิดข้อผิดพลาดขึ้นหรือทำให้ไวยากรณ์ในกระบวนการบีเพลไม่ถูกต้อง ทำให้โปรแกรมสามารถตรวจสอบผลลัพธ์ที่ได้จากมิวแตนท์เมื่อเปรียบเทียบกับผลลัพธ์ของโปรแกรมต้นแบบ

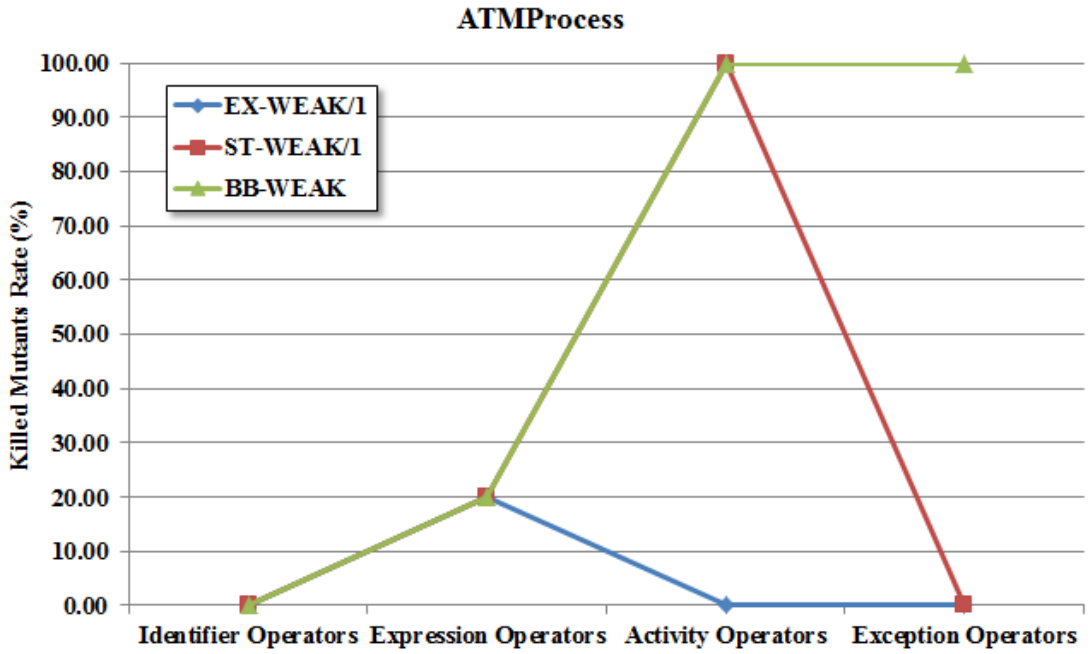
จากนั้นนำข้อมูลที่ได้จากกราฟมาสรุปเป็นตารางแสดงอัตราส่วนมิวแตนท์ที่ถูกกำจัดระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ EX-WEAK/1 ST-WEAK/1 และ BB-WEAK ในตารางที่ 5.40 – 5.42



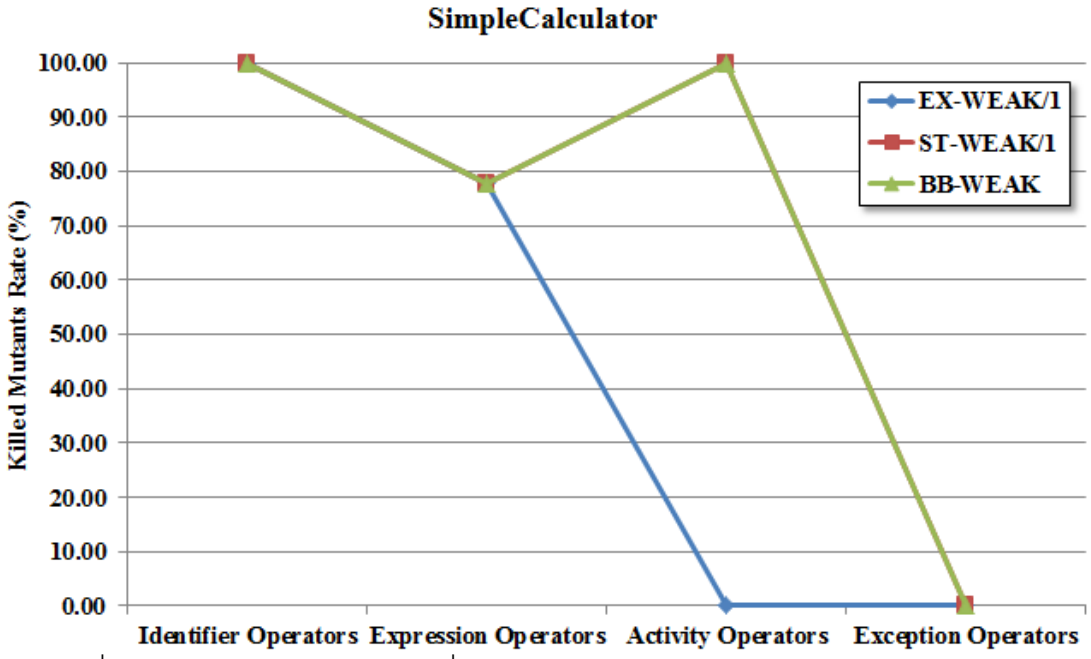
ภาพที่ 5.22 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม Triangle



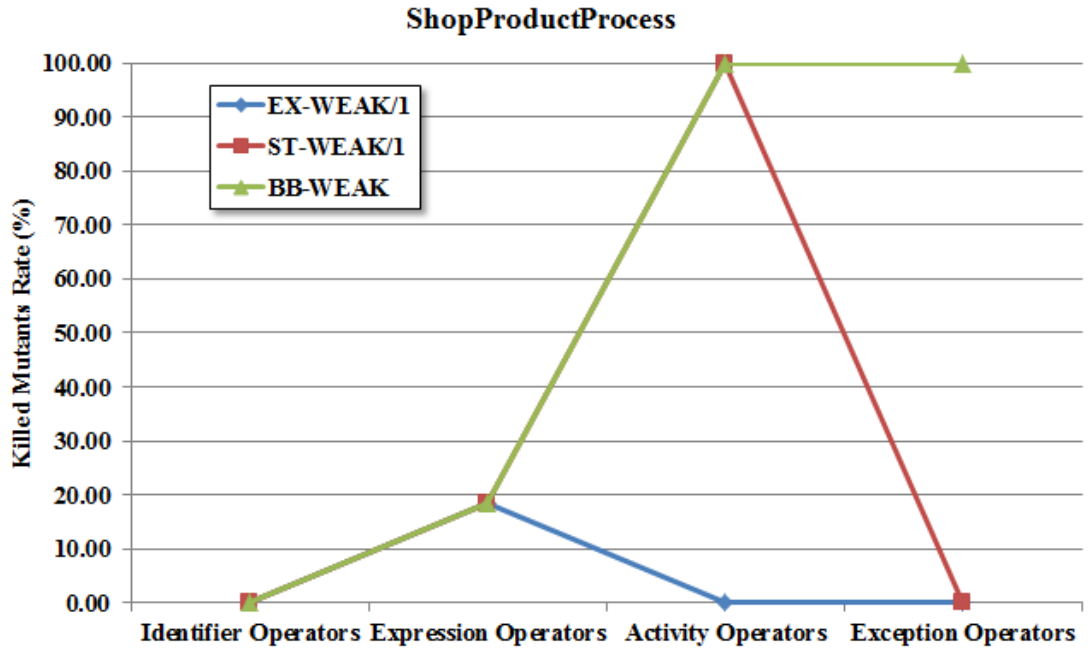
ภาพที่ 5.23 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม LoanApprovalProcess



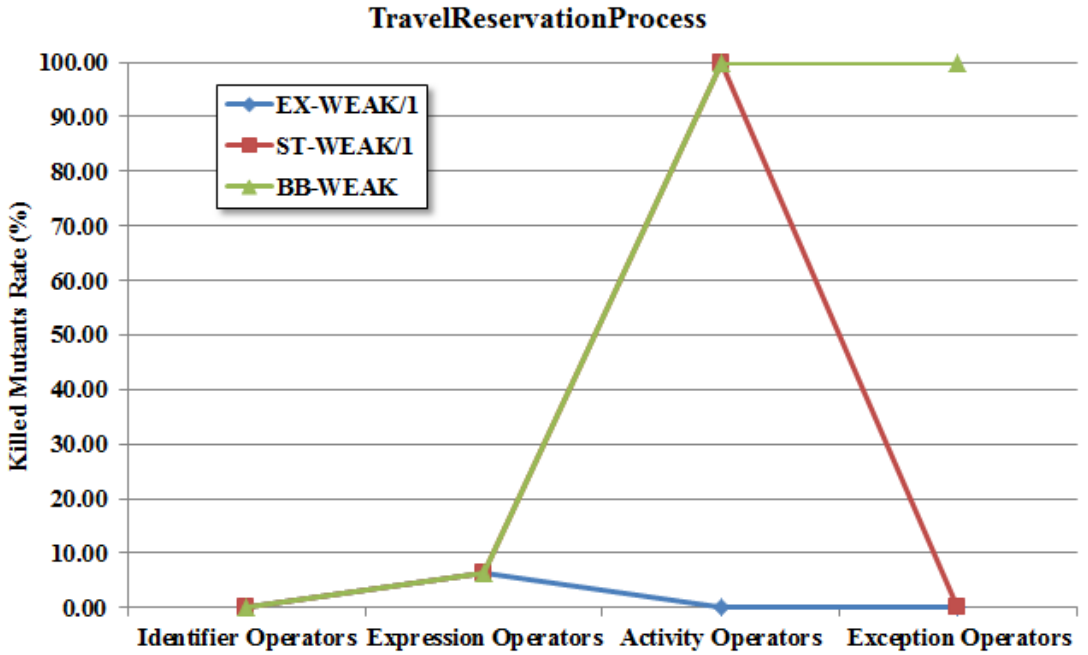
ภาพที่ 5.24 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ATMProcess



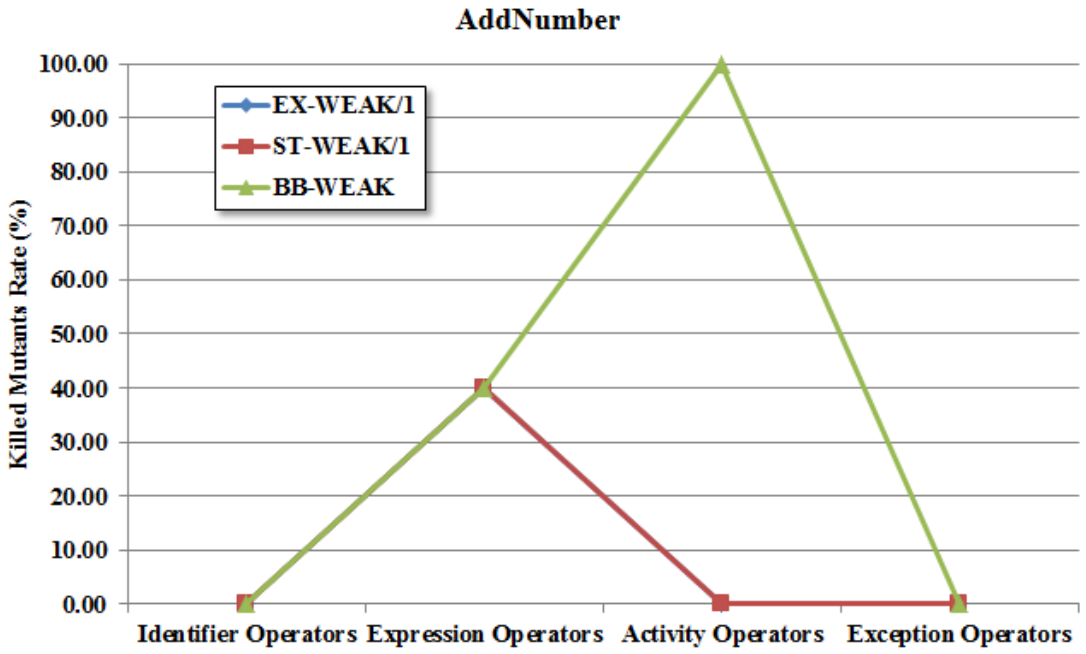
ภาพที่ 5.25 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SimpleCalculator



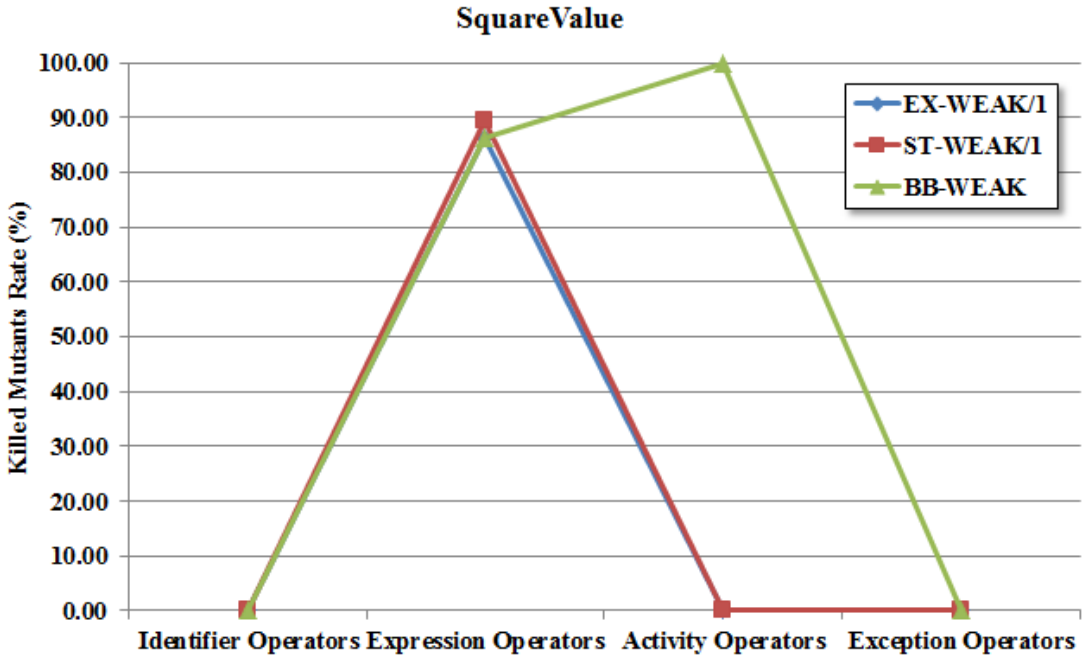
ภาพที่ 5.26 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ShopProductProcess



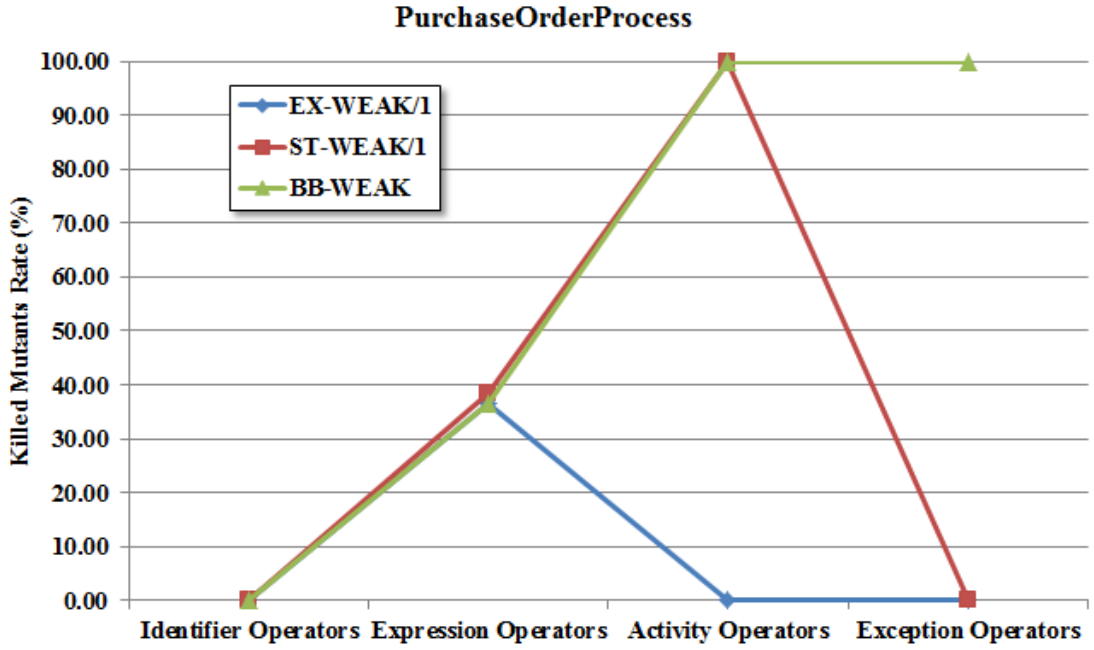
ภาพที่ 5.27 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม TravelReservationProcess



ภาพที่ 5.28 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม AddNumber



ภาพที่ 5.29 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SquareValue



ภาพที่ 5.30 อัตราส่วนของมิวแตนท์ที่กำจัดได้ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม PurchaseOrderProcess

ตารางที่ 5.40 อัตราส่วนมิวแตนท์ที่ถูกกำจัดระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ EX-WEAK/1

โปรแกรม	อัตราส่วนมิวแตนท์ที่ถูกกำจัด (%) ของ EX-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	5.26%	15.64%	0%	0%
อนุมัติการกู้ยืม	0%	43.64%	0%	0%
เอทีเอ็ม	0%	20%	0%	0%
เครื่องคิดเลข	100%	77.62%	0%	0%
การซื้อสินค้า	0%	18.52%	0%	0%
การจองตั๋วท่องเที่ยว	0%	6.25%	0%	0%
การเพิ่มจำนวน	0%	40%	0%	0%
การหาค่ากำลังสอง	0%	86.26%	0%	0%
ใบสั่งซื้อสินค้า	0%	36.42%	0%	0%

ตารางที่ 5.41 อัตราส่วนมิวแตนท์ที่ถูกกำจัดระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ ST-WEAK/1

โปรแกรม	อัตราส่วนมิวแตนท์ที่ถูกกำจัด (%) ของ ST-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	22.81%	62.08%	100%	0%
อนุมัติการกู้ยืม	0%	50.91%	100%	0%
เอทีเอ็ม	0%	20%	100%	0%
เครื่องคิดเลข	100%	77.61%	100%	0%
การซื้อสินค้า	0%	18.52%	100%	0%
การจองตั๋วท่องเที่ยว	0%	6.25%	100%	0%
การเพิ่มจำนวน	0%	40.00%	0%	0%
การหาค่ากำลังสอง	0%	89.31%	0%	0%
ใบสั่งซื้อสินค้า	0%	38.41%	100%	0%

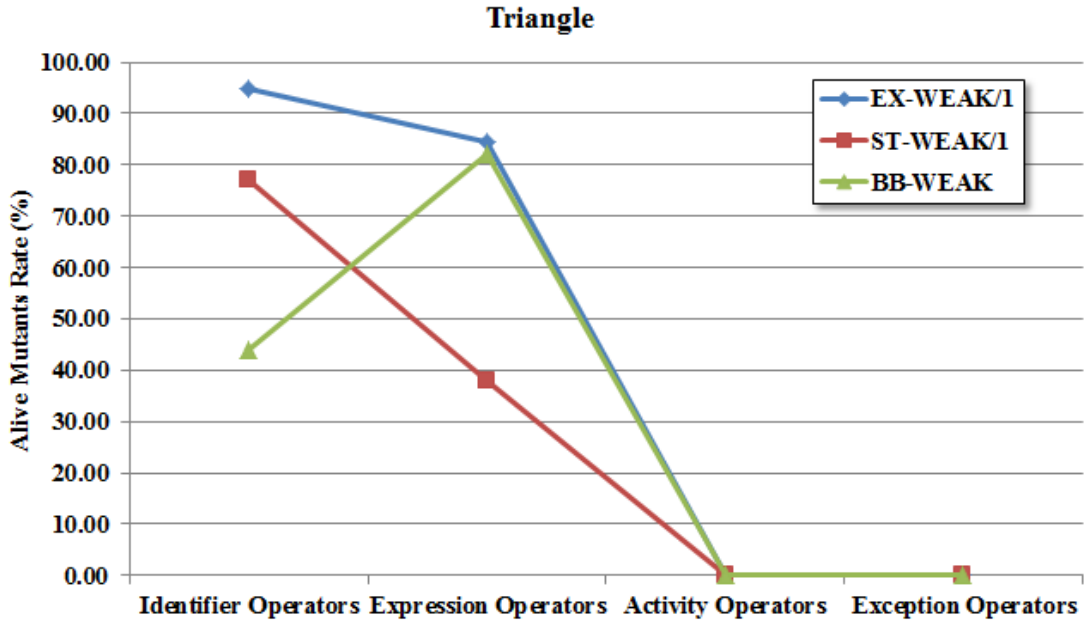
ตารางที่ 5.42 อัตราส่วนมิวแตนท์ที่ถูกกำจัดระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ BB-WEAK

โปรแกรม	อัตราส่วนมิวแตนท์ที่ถูกกำจัด (%) ของ BB-WEAK			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	56.14%	18.00%	100%	0%
อนุมัติการกู้ยืม	0%	43.64%	100%	100%
เอทีเอ็ม	0%	20.00%	100%	100%
เครื่องคิดเลข	100%	77.61%	100%	0%
การซื้อสินค้า	0%	18.52%	100%	100%
การจองตั๋วท่องเที่ยว	0%	6.25%	100%	100%
การเพิ่มจำนวน	0%	40.00%	100%	0%
การหาค่ากำลังสอง	0%	86.26%	100%	0%
ใบสั่งซื้อสินค้า	0%	36.42%	100%	100%

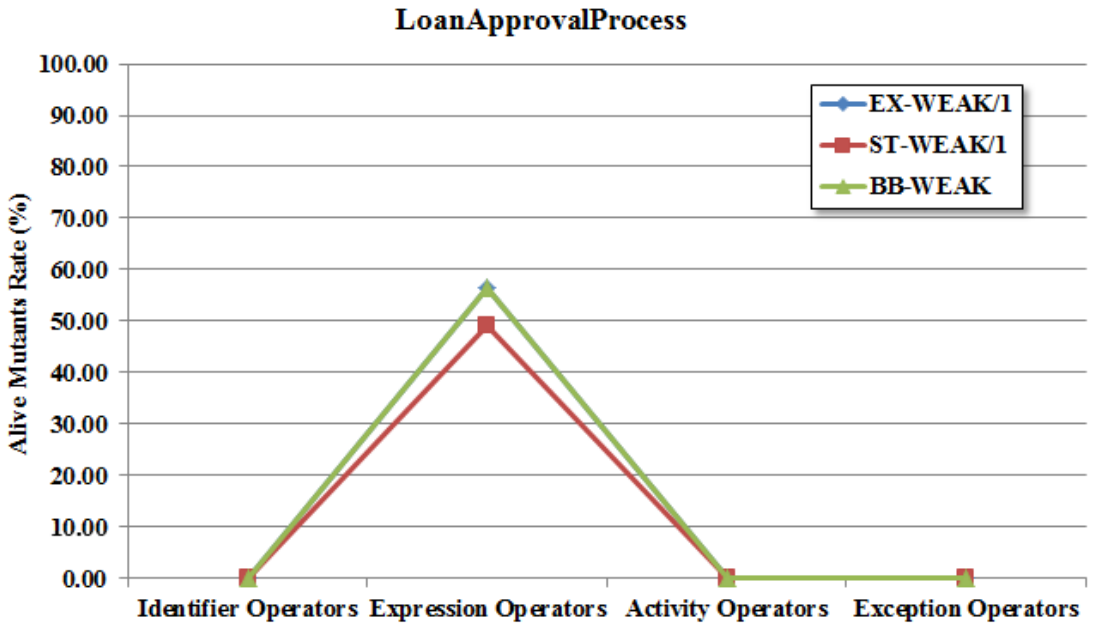
จากภาพที่ 5.31 – 5.39 แสดงอัตราส่วนมิวแตนท์ที่ยังคงอยู่เป็นรูปแบบเปอร์เซนต์ในแกนตั้ง ส่วนในแกนนอนเป็นชนิดของตัวดำเนินการมิวเทชัน ซึ่งมีตัวดำเนินการมิวเทชันชนิดตัวระบุ ตัวดำเนินการมิวเทชันชนิดนิพจน์ ตัวดำเนินการมิวเทชันชนิดกิจกรรม และตัวดำเนินการมิวเทชันชนิดข้อผิดพลาด

ดังภาพที่ 5.31 – 5.39 แสดงให้เห็นว่ามิวแตนท์ที่ยังคงอยู่ส่วนใหญ่มาจากตัวดำเนินการชนิดตัวระบุและชนิดนิพนจ์ ทั้งนี้เนื่องจากอัตราส่วนการสร้างมิวแตนท์ที่เกิดจากตัวดำเนินการมิวเทชันที่เกิดจากชนิดนิพจน์และตัวระบุมีมากกว่าชนิดกิจกรรมกับข้อผิดพลาด รวมไปถึงการดัดแปลงโปรแกรมบีเพลของตัวดำเนินการมิวเทชันชนิดกิจกรรมและข้อผิดพลาด ทำให้โปรแกรมเกิดข้อผิดพลาดขึ้นระหว่างดีพลอยหรือขณะประมวลผล ขณะที่ชนิดตัวระบุและนิพจน์นั้นผลลัพธ์อาจไม่สามารถตรวจสอบได้ เนื่องจากบางนิพจน์หรือประพจน์ไม่สามารถหานิพจน์ในสุดได้

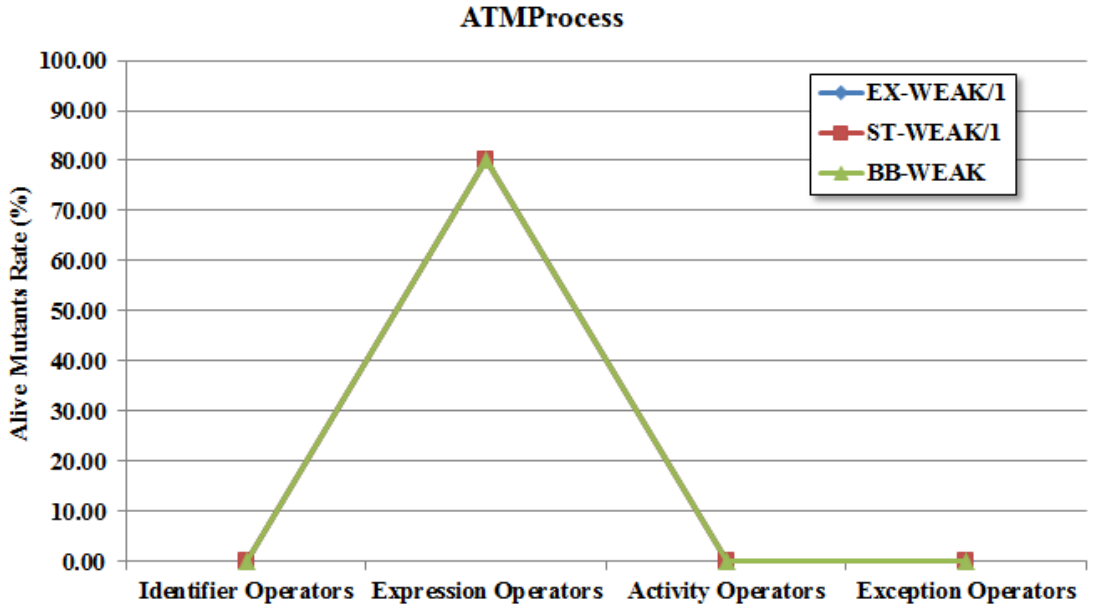
จากนั้นนำข้อมูลที่ได้จากกราฟมาสรุปเป็นตารางแสดงอัตราส่วนมิวแตนท์ที่ยังคงอยู่ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ EX-WEAK/1 ST-WEAK/1 และ BB-WEAK ในตารางที่ 5.43 – 5.45



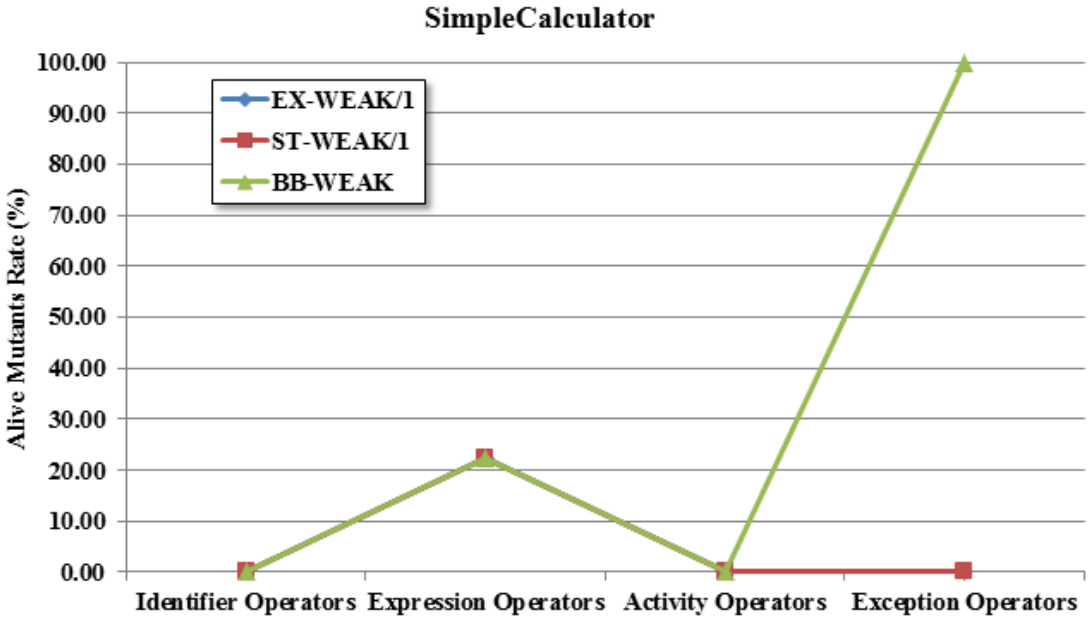
ภาพที่ 5.31 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม Triangle



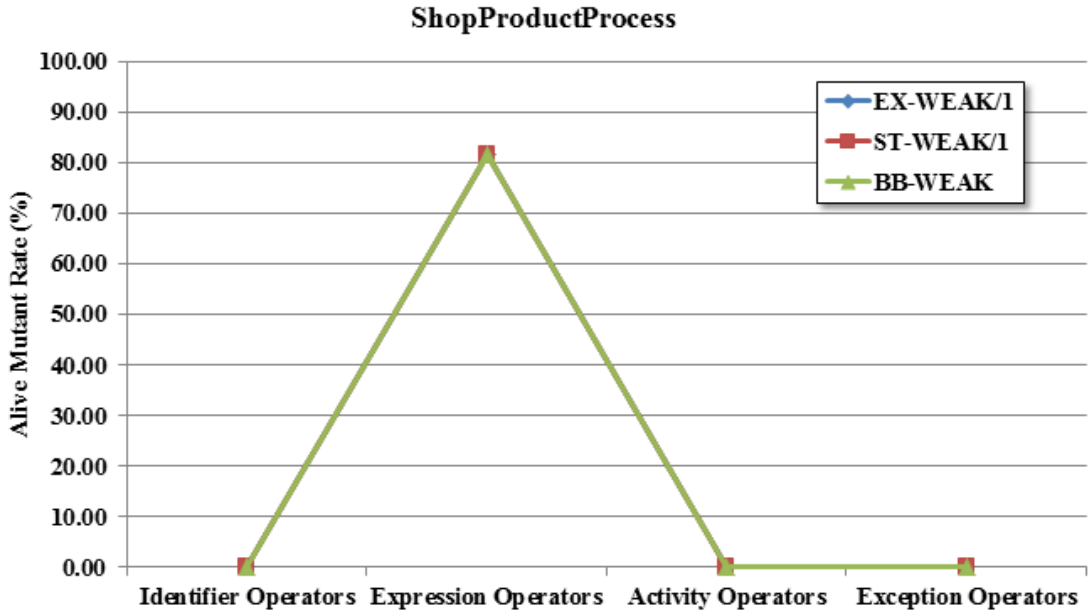
ภาพที่ 5.32 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม LoanApprovalProcess



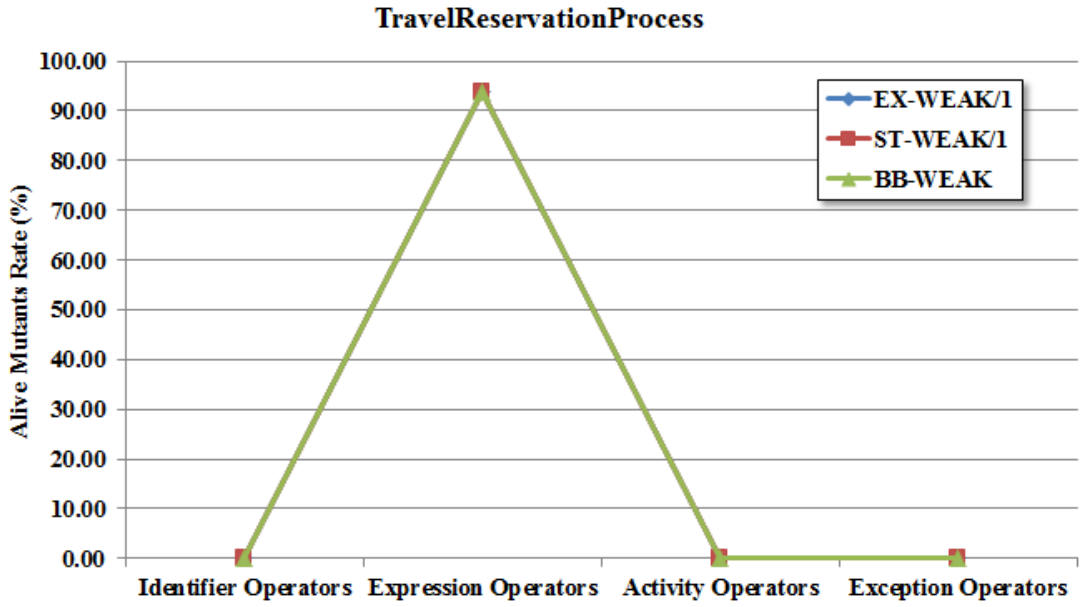
ภาพที่ 5.33 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ATMProcess



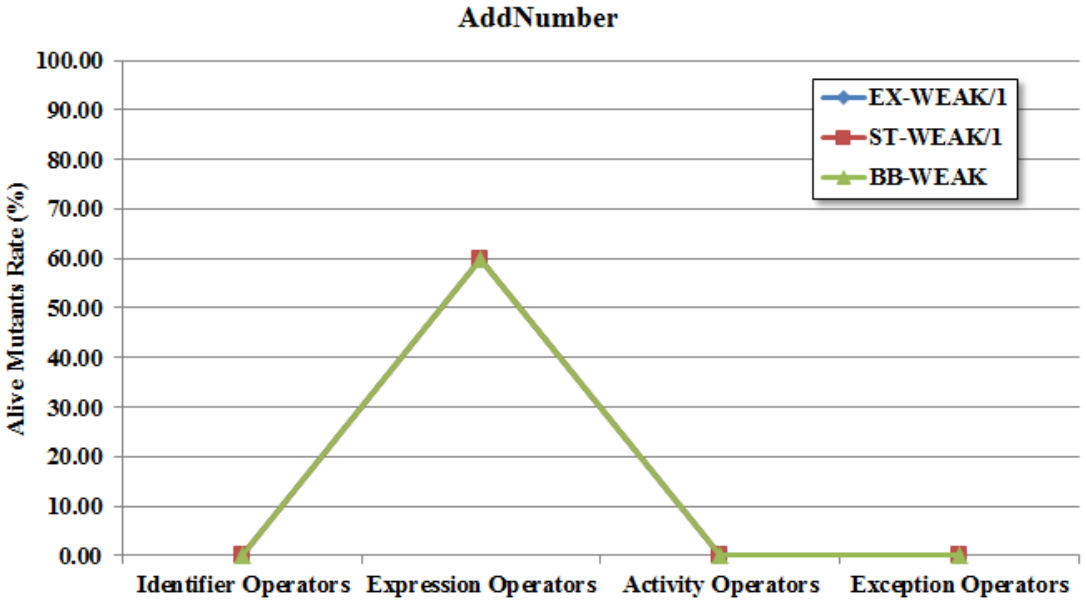
ภาพที่ 5.34 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SimpleCalculator



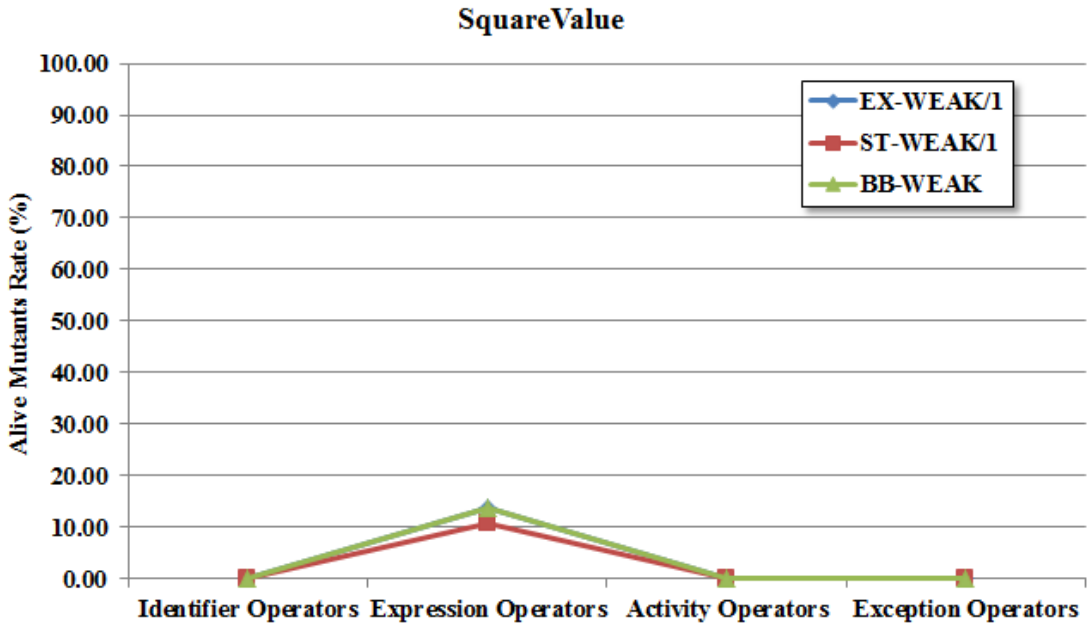
ภาพที่ 5.35 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม ShopProductProcess



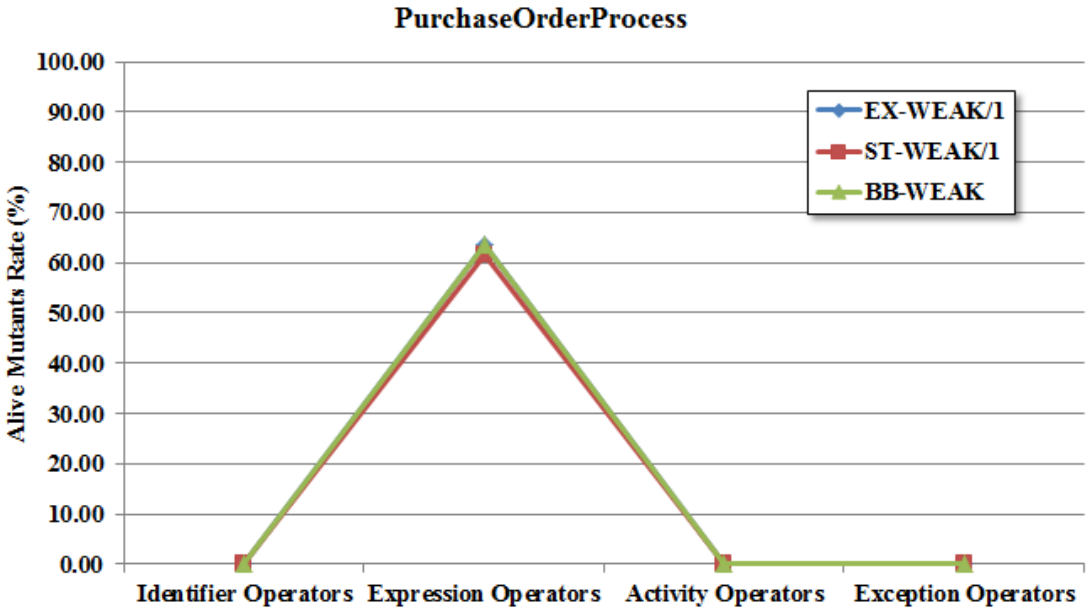
ภาพที่ 5.36 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม TravelReservationProcess



ภาพที่ 5.37 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม AddNumber



ภาพที่ 5.38 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม SquareValue



ภาพที่ 5.39 อัตราส่วนของมิวแตนท์ที่ยังคงอยู่ตามชนิดตัวดำเนินการมิวเทชันของโปรแกรม PurchaseOrderProcess

ตารางที่ 5.43 อัตราส่วนมิวแตนท์ที่ยังคงอยู่ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ EX-WEAK/1

โปรแกรม	อัตราส่วนมิวแตนท์ที่ยังคงอยู่ (%) ของ EX-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	94.74%	84.36%	0%	0%
อนุมัติการกู้ยืม	0%	56.36%	0%	0%
เอทีเอ็ม	0%	80.00%	0%	0%
เครื่องคิดเลข	0%	22.39%	0%	0%
การซื้อสินค้า	0%	81.48%	0%	0%
การจองตั๋วท่องเที่ยว	0%	93.75%	0%	0%
การเพิ่มจำนวน	0%	60.00%	0%	0%
การหาค่ากำลังสอง	0%	13.74%	0%	0%
ใบสั่งซื้อสินค้า	0%	63.58%	0%	0%

ตารางที่ 5.44 อัตราส่วนมิวแตนท์ที่ยังคงอยู่ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ ST-WEAK/1

โปรแกรม	อัตราส่วนมิวแตนท์ที่ยังคงอยู่ (%) ของ ST-WEAK/1			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	77.19%	37.91%	0%	0%
อนุมัติการกู้ยืม	0%	49.09%	0%	0%
เอทีเอ็ม	0%	80%	0%	0%
เครื่องคิดเลข	0%	22.39%	0%	0%
การซื้อสินค้า	0%	81.48%	0%	0%
การจองตั๋วท่องเที่ยว	0%	93.75%	0%	0%
การเพิ่มจำนวน	0%	60.00%	0%	0%
การหาค่ากำลังสอง	0%	10.69%	0%	0%
ใบสั่งซื้อสินค้า	0%	61.59%	0%	0%

ตารางที่ 5.45 อัตราส่วนมิวแตนท์ที่ยังคงอยู่ระหว่างโปรแกรมบีเพลและตัวดำเนินการมิวเทชันทั้ง 4 ชนิดด้วยการวิเคราะห์แบบ BB-WEAK

โปรแกรม	อัตราส่วนมิวแตนท์ที่ยังคงอยู่ (%) ของ BB-WEAK			
	Identifier	Expression	Activity	Exception
การหาชนิดสามเหลี่ยม	43.86%	82.00%	0%	0%
อนุมัติการกู้ยืม	0%	56.36%	0%	0%
เอทีเอ็ม	0%	80.00%	0%	0%
เครื่องคิดเลข	0%	22.39%	0%	100%
การซื้อสินค้า	0%	81.48%	0%	0%
การจองตั๋วท่องเที่ยว	0%	93.75%	0%	0%
การเพิ่มจำนวน	0%	60.00%	0%	0%
การหาค่ากำลังสอง	0%	13.74%	0%	0%
ใบสั่งซื้อสินค้า	0%	63.58%	0%	0%

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

วิทยานิพนธ์นี้นำเสนอวิธีการวิเคราะห์ตัวดำเนินการมิวเทชันในการที่จะประยุกต์ใช้กับแนวทางการทดสอบแบบวีคมิวเทชัน ที่ได้นำเสนอไว้ในงานที่ผ่านมา [8] และทดสอบด้วยแนววิธีมิวเทชันแบบดั้งเดิม ซึ่งตัวดำเนินการมิวเทชันมีทั้งหมด 26 ตัวดำเนินการและได้แบ่งออกเป็น 4 ชนิด คือ ตัวดำเนินการมิวเทชันชนิดตัวระบุ ตัวดำเนินการมิวเทชันชนิดนิพจน์ ตัวดำเนินการมิวเทชันชนิดกิจกรรม และตัวเนินการมิวเทชันชนิดความผิดปกติและเหตุการณ์ โดยในวิทยานิพนธ์นี้ได้นำตัวดำเนินการดังกล่าวมาใช้ทั้งหมด 24 ตัวดำเนินการ คือ ISV EAA EEU ECC ECN ELL EMD EMF ERR AEL AFP AIE AJC APA APM ASF ASI AWR XEE XER XMC XMF XMT และ XTF ส่วนอีก 2 ตัวดำเนินการ คือ ACI และ AIS ไม่ได้ถูกนำมาประยุกต์ใช้ด้วย

ตัวดำเนินการ ACI (การเปลี่ยนค่าในแอตทริบิวต์ createInstance ในกิจกรรม ข้อความเป็น no) โดยตัวดำเนินการ ACI จะสร้างตัวอย่างกระบวนการ (Process Instance) ซึ่งไม่สามารถเปรียบเทียบผลลัพธ์ระหว่างโปรแกรมต้นแบบและมิวแตนท์ได้ ส่วนตัวดำเนินการ AIS (การเปลี่ยนค่าในแอตทริบิวต์ isolated ของกิจกรรม scope เป็น no) จะเป็นการพิจารณาการเข้าถึงตัวแปร ด้วยเหตุนี้จึงไม่ถูกนำมาใช้ในแนวทางในการสร้างมิวแตนท์ในวิทยานิพนธ์นี้

ในการดำเนินการทดสอบเครื่องมือการทดสอบวีคมิวเทชันสำหรับดับเบิลเอสบีเพลในวิทยานิพนธ์นี้ ได้สร้างโปรแกรมเพื่อใช้ในการทดสอบเครื่องมือทั้งหมด 9 โปรแกรม นั่นคือ โปรแกรมหาชนิดของสามเหลี่ยม โปรแกรมเครื่องคิดเลข โปรแกรมการอนุมัติการกู้ยืม โปรแกรมเอทีเอ็ม โปรแกรมการซื้อสินค้า โปรแกรมการจองตั๋วท่องเที่ยว โปรแกรมเพิ่มจำนวน โปรแกรมหาค่ากำลังสอง และโปรแกรมสั่งซื้อสินค้า โดยโปรแกรมทั้ง 9 โปรแกรมจะถูกสร้างมิวแตนท์โดยตัวดำเนินการทั้ง 24 ตัวดำเนินการ ซึ่งการสร้างมิวแตนท์จะขึ้นกับพฤติกรรมการดัดแปลงโปรแกรมบีเพลในแต่ละตัวดำเนินการ จากนั้นทำการทดสอบมิวแตนท์ที่สร้างขึ้นโดยใช้กรณีทดสอบที่เตรียมไว้ในแต่ละโปรแกรม จากนั้นระบบแสดงผลลัพธ์ของการทดสอบมิวแตนท์ได้อย่างถูกต้อง โดยรายงานมิวแตนท์ที่สร้างขึ้นทั้งหมดในแต่ละตัวดำเนินการ มิวแตนท์ที่ถูกกำจัดในแต่ละตัวดำเนินการ เวลาที่ใช้ในการทดสอบมิวแตนท์ คะแนนมิวเทชัน และประสิทธิภาพของกรณีทดสอบ

แนวโน้มในการสร้างมิวแตนท์ของตัวดำเนินการแต่ละชนิด ดังนี้ตัวดำเนินการชนิดตัวระบุตัวดำเนินการชนิดนิพจน์ ตัวดำเนินการชนิดกิจกรรม และตัวดำเนินการชนิดข้อผิดพลาด ซึ่งชนิดที่สร้างมิวแตนท์ในอัตราส่วนที่มากที่สุดจะเป็นชนิดนิพจน์ เนื่องจากในโปรแกรมบีเพลประกอบด้วยนิพจน์และประพจน์เป็นจำนวนที่มากกว่า เมื่อเปรียบเทียบกับจำนวนกิจกรรมและจำนวนกิจกรรมที่ดักจับข้อผิดพลาด

6.2 ข้อจำกัดของงานวิจัย

1) วิทยานิพนธ์นี้พิจารณาเฉพาะบีเพลที่เป็นไปตามมาตรฐานของโอเอซิสเท่านั้น ซึ่งในเครื่องมือที่ใช้ออกแบบและพัฒนากระบวนการบีเพลมีรูปแบบที่แน่นอน ดังนั้นเมื่อโปรแกรมบีเพลไปติดตั้งเซิร์ฟเวอร์อื่นๆ เช่น ออราเคิลแอพลิเคชันเซิร์ฟเวอร์ (Oracle Application Server) หรือเครื่องประมวลแอ็คทีฟ วีโอเอส (Active VOS Engine) เป็นต้น อาจเกิดข้อผิดพลาดขึ้นในระหว่างการติดตั้งได้ จึงจำเป็นต้องมีการปรับโครงสร้างของโปรแกรมบีเพลเพื่อให้โปรแกรมทำงานได้อย่างปกติ

2) วิทยานิพนธ์นี้สนับสนุนการรับข้อมูลเข้าและนำข้อมูลออกชนิด integer, double, float, boolean, และ string เท่านั้น

3) วิทยานิพนธ์นี้ไม่ได้ประยุกต์ใช้ตัวดำเนินการมิวเทชัน ACI และ AIS ในการสร้างมิวแตนท์ ดังที่กล่าวไว้ในหัวข้อที่ 6.1

4) เครื่องมือที่ใช้ทดสอบวีคมิวเทชันนี้ไม่สามารถสร้างกรณีทดสอบได้อย่างอัตโนมัติ ดังนั้นเพื่อให้เครื่องมือทดสอบทำงานได้อย่างมีประสิทธิภาพ ควรปรับปรุงให้เครื่องมือสามารถสร้างกรณีทดสอบได้อย่างอัตโนมัติ

6.3 ข้อเสนอแนะและแนวทางการดำเนินงานต่อ

1) พัฒนาระบบให้สามารถทดสอบกระบวนการบีเพลได้หลายกระบวนการ

2) พัฒนาระบบให้สามารถสร้างกรณีทดสอบได้อย่างอัตโนมัติ

3) พัฒนาระบบให้สามารถทดสอบด้วยวิธีสตรองมิวเทชันและมิวเทชันการคัดเลือก

4) พัฒนาระบบให้สามารถทดสอบด้วยกรณีทดสอบได้หลายชุด และสามารถเลือกกรณีทดสอบที่ใช้ทดสอบได้ เนื่องจากตอนนี้ต้องนำกรณีทดสอบไปไว้ที่ฐานข้อมูลกรณีทดสอบ

รายการอ้างอิง

[1] OASIS Standard. Reference Model for Service Oriented Architecture 1.0, C. Matthew MacKenzie, Adobe Systems Incorporated, Ken Laskey, MITRE Corporation, Francis McCabe, Fujitsu Laboratories of America Limited, Peter F Brown, Rebekah Metz, Booz Allen Hamilton. Available from: http://docs.oasis-open.org/soa-rm/v1.0/ [2010, December 19]

[2] W3C Recommendation, Extensible Markup Language (XML) 1.0 (Fifth Edition), Tim Bray, Textuality and Netscape, Jean Paoli, C. M. Sperberg-McQueen, W3C, Eve Maler, Sun Microsystems, Inc., Fran?ois Yergeau. Available from: http://www.w3.org/TR/REC-xml/ [2010, December 19]

[3] W3C Recommendation, SOAP Version 1.2 Part 0: Primer (Second Edition), Nilo Mitra, Ericsson Yves Lafon, W3C, 27 April 2007, Available from: http://www.w3.org/TR/2007/REC-soap12-part1-20070427/. [2010, December 20]

[4] W3C Recommendation. Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. Roberto Chinnici, Sun Microsystems, Jean-Jacques Moreau, Canon, Arthur Ryman, IBM, Sanjiva Weerawarana, WSO2, 26 June 2007. Available from: http://www.w3.org/TR/2007/REC-wsdl20-20070626/ [2010, December 20]

[5] OASIS. Web Services Business Process Execution Language 2.0[Online]. Organization for the Advancement of Structured Information Standards. Available from: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html [2010, December 21]

[6] Y. Jia and M. Harman. An Analysis and Survey of the Development of Mutation Testing. CREST Center. King's College, London, Tech. Rep. TR-09-06. 2009.

[7] Natthapol Thaisakonpun and Taratip Suwannasart. Mutation Testing for Expression Modification Operator of BPEL. Software Engineering

Laboratory, Center of Excellence in Software Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand.

[8] Antonia Estero-Botaro, Francisco Palomo-Lozano, and Inmaculada Medina-Bulo. Quantitative Evaluation of Mutation Operators for WS-BPEL Compositions. Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference, Paris, pp. 142-150.

[9] A. Estero-Botaro, F. Palomo-Lozano, and I. Medina-Bulo. Mutation operators for WS-BPEL 2.0. ICSSEA 2008: 21th International Conference on Software & Systems Engineering and their Applications, Paris, France. 2008.

[10] A. Jefferson Offutt, and Stephen D. Lee. An Empirical Evaluation of Weak Mutation. Department of Information and Software Systems Engineering, George Mason University, Fairfax, VA 22030, Stephen D. Lee, IBM Corporation A00/062, P.O. Box 12195, Research Triangle Park, NC 27709. February 24, 1996.

[11] A. Jefferson Offutt and Stephen D. Lee. 1991. How strong is weak mutation?. Proceedings of the symposium on Testing, analysis, and verification (TAV4). ACM, New York, NY, USA, 200-213. Available from: http://doi.acm.org/10.1145/120807.120826.

[12] M. R. Girgis and M. R. Woodward. 1985. An integrated system for program testing using weak mutation and data flow analysis. Proceedings of the 8th international conference on Software engineering (ICSE '85). IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 313-319.

[13] W. E. Howden. Weak Mutation Testing and Completeness of Test Sets. IEEE Transactions on Software Engineering. 8(4):371-379, July 1982.

[14] World Wide Web Consortium (W3C). Available from: http://www.w3.org/ [2010, December 20].

[15] Dan Connolly. Overview of SGML Resources. Dan Connolly, Editor of the HTML 2.0 specification Created Nov 1995. Available from: http://www.w3.org/MarkUp/SGML/ [2010, December 15]

[16] W3C Recommendation. XML Schema Part 0: Primer Second Edition. David C. Fallside, IBM, Priscilla Walmsley - Second Edition. Available from: http://www.w3.org/TR/xmlschema-0/. [2010, November 19]

[17] Wikipedia contributors. Web Services Description Language. The Free Encyclopedia. Available from: http://en.wikipedia.org/wiki/Web_Services_Description_Language [2010, November 19]

[18] Kumar Raj Moorthy. An Introduction to BPEL. Available from: http://www.developer.com/ [2010, November 19]

[19] H. Agrawal, R. Demillo, R. Hathaway, W. Hsu, W. Hsu, E. Krauser, R. J. Martin, A. Mathur, and E. Spafford. Design of mutant operators for the C programming language. Software Engineering Research Center, Department of Computer Science, Purdue University, Indiana, Tech. Rep. SERC-TR-41-P, 1989.

[20] K. N. King and A. J. Offutt. A FORTRAN language system for mutation-based software testing. Software – Practice and Experience, vol. 21, no. 7, pp. 685 –718, 1991.

[21] A. J. Offutt, J. Voas, and J. Payne. Mutation operators for Ada. Information and Software Systems Engineering, George Mason University, Tech. Rep. ISSE-TR-96-09, 1996.

[22] J. Tuya, M. J. Suárez-Cabal, and C. de la Riva. Mutating database queries. Information and Software Technology, vol. 49, no. 4, pp. 398–417, 2007.

[23] A. Derezińska, Quality Assessment of Mutation Operators Dedicated for C# Programs, in QSIC 2006: Sixth International Conference on Quality Software. Beijing, China: IEEE Computer Society, 2006, pp. 227–234.

[24] A. Derezińska, An Experiment Case Study to Applying Mutation Analysis for SQL Queries. International Multiconference on Computer Science and Information Technology, 2009, IMCSIT 2009, pp. 559-566.

[25] J. S. Bradbury, J. R. Cordy, and J. Dingel, Mutation Operators for Concurrent Java (J2SE 5.0). Proceedings of the 2nd Workshop on Mutation Analysis (MUTATION'06). Raleigh, North Carolina: IEEE Computer Society, November 2006, pp. 83–92.

[26] R. T. Alexander, J. M. Bieman, S. Ghosh, and B. Ji, Mutation of Java Objects. In Proceedings of the 13th International Symposium on Software Reliability Engineering (ISSRE'02). Annapolis, Maryland: IEEE Computer Society, 12-15 November 2002, pp. 341–351.

[27] H. Agrawal, R. A. DeMillo, B. Hathaway, W. Hsu, W. Hsu, E. W. Krauser, R. J. Martin, A. P. Mathur, and E. Spafford, Design of Mutant Operators for the C Programming Language. Purdue University, West Lafayette, Indiana, Technique Report SERC-TR-41-P, March 1989.

ภาคผนวก

ภาคผนวก ก
อภิธานศัพท์

SOA (Service Oriented Architecture)	สถาปัตยกรรมเชิงบริการที่มองทุกอย่างเป็นลักษณะเชิงบริการ
Web Service	ระบบที่ใช้ในการติดต่อสื่อสารข้อมูลระหว่างคอมพิวเตอร์ผ่านอินเตอร์เนต ในรูปแบบเอ็กซ์เอ็มแอล
Web Service Business Process Execution Language (WS-BPEL)	เอกสารเอ็กซ์เอ็มแอลที่ใช้ในการเพิ่มประสิทธิภาพกับเว็บเซอร์วิซให้สามารถทำงานได้ซ้ำซ้อนมากขึ้น
SOAP (Simple Object Access Protocol)	โพรโตคอลที่ใช้ในการติดต่อสื่อสารข้อมูลในระดับชั้นแอพลิเคชัน ผ่านทางอินเตอร์เนตโพรโตคอล เช่น HTTP, SMTP และ FTP เป็นต้น
WSDL (Web Service Definition Language)	เอกสารเอ็กซ์เอ็มแอลที่อธิบายถึงการทำงานของบริการ หรือการติดต่อโดยใช้การส่งข้อมูลแบบไหน
Mutation Testing	การทดสอบแบบมิวเทชัน พิจารณาผลลัพธ์ของโปรแกรมต้นแบบและผลลัพธ์ของมิวแตนท์
Strong Mutation Testing	การทดสอบแบบสตรองมิวเทชัน จะเหมือนกับการทดสอบแบบมิวเทชัน
Weak Mutation Testing	การทดสอบแบบวีคมิวเทชัน พิจารณาผลลัพธ์ขององค์ประกอบในโปรแกรม (component) โดยเปรียบเทียบผลลัพธ์ระหว่างองค์ประกอบของโปรแกรมต้นแบบและมิวแตนท์
Selective Mutation Testing	การทดสอบมิวเทชันแบบคัดเลือก จะคล้ายกับการทดสอบแบบสตรองมิวเทชัน แต่จะเป็นการคัดเลือกตัวดำเนินการมิวเทชัน ว่าตัวดำเนินการมิวเทชันตัวใดมีผลต่อการสร้างมิวแตนท์มากที่สุด
Mutation Operator	ตัวดำเนินการมิวเทชันมีจุดประสงค์ในการดัดแปลงโปรแกรมต้นแบบ ซึ่งการดัดแปลงก็จะขึ้นอยู่กับชนิดของตัวดำเนินการมิวเทชันเช่นกัน

Mutant	โปรแกรมต้นแบบที่ถูกดัดแปลงโปรแกรมโดยใช้ตัวดำเนินการมิวเทชัน
Mutation Score	คะแนนมิวเทชันหรือร้อยละในการกำจัดมิวแตนท์ที่กำจัดได้ ซึ่งถูกวัดตามสัดส่วนของมิวแตนท์ที่ถูกกำจัดต่อจำนวนมิวแตนท์ทั้งหมด $$M = \frac{D}{T - E}$$
Test Cases Effectiveness	ประสิทธิภาพของกรณีทดสอบ ซึ่งอยู่ในรูปแบบความสัมพันธ์ของคะแนนมิวเทชันและประสิทธิภาพของกรณีทดสอบ $$\mathrm{E} = M \times \frac{\overline{K}_D}{T}$$ โดยที่ $$\overline{K}_D = \frac{\sum K_M}{D}$$
Web Server	เครื่องบริการเว็บถูกใช้เป็นสภาพแวดล้อมให้กับเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล
BPEL Engine	เครื่องประมวลผลบีเพลใช้เป็นสภาพแวดล้อมเพื่อให้กระบวนการบีเพลสามารถทำงานได้

ภาคผนวก ข
การติดตั้งเครื่องมือ

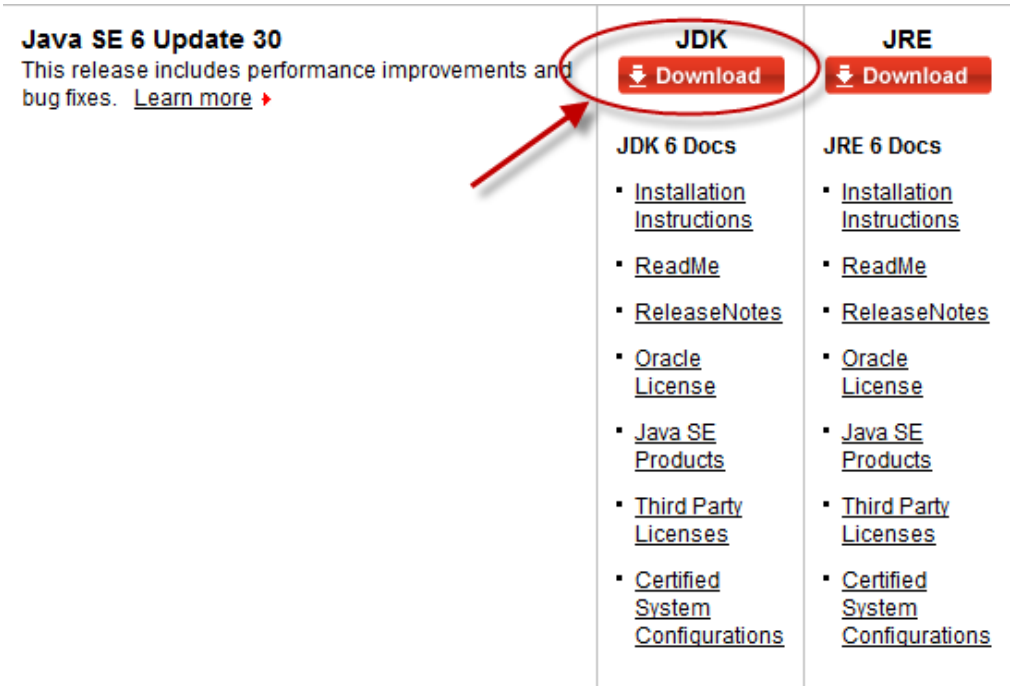
บทนี้อธิบายถึงการติดตั้งเครื่องมือที่ใช้ในการพัฒนาเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพล ซึ่งเครื่องมือที่ใช้พัฒนานั้นจะแบ่งออกเป็น 4 ส่วน คือ การติดตั้งโปรแกรมภาษาจาวา การติดตั้งสภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ การติดตั้งเครื่องบริการเว็บและเครื่องประมวลผลบีเพล และฐานข้อมูลที่เว็บเซอร์วิซเรียกใช้

ข.1 การติดตั้งโปรแกรมภาษาจาวา

เครื่องมือทดสอบแบบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลถูกพัฒนาด้วยภาษาจาวา ดังนั้นจึงจำเป็นต้องติดตั้งโปรแกรมภาษาจาวาเพื่อใช้ในการพัฒนา โดยมีขั้นตอนดังนี้

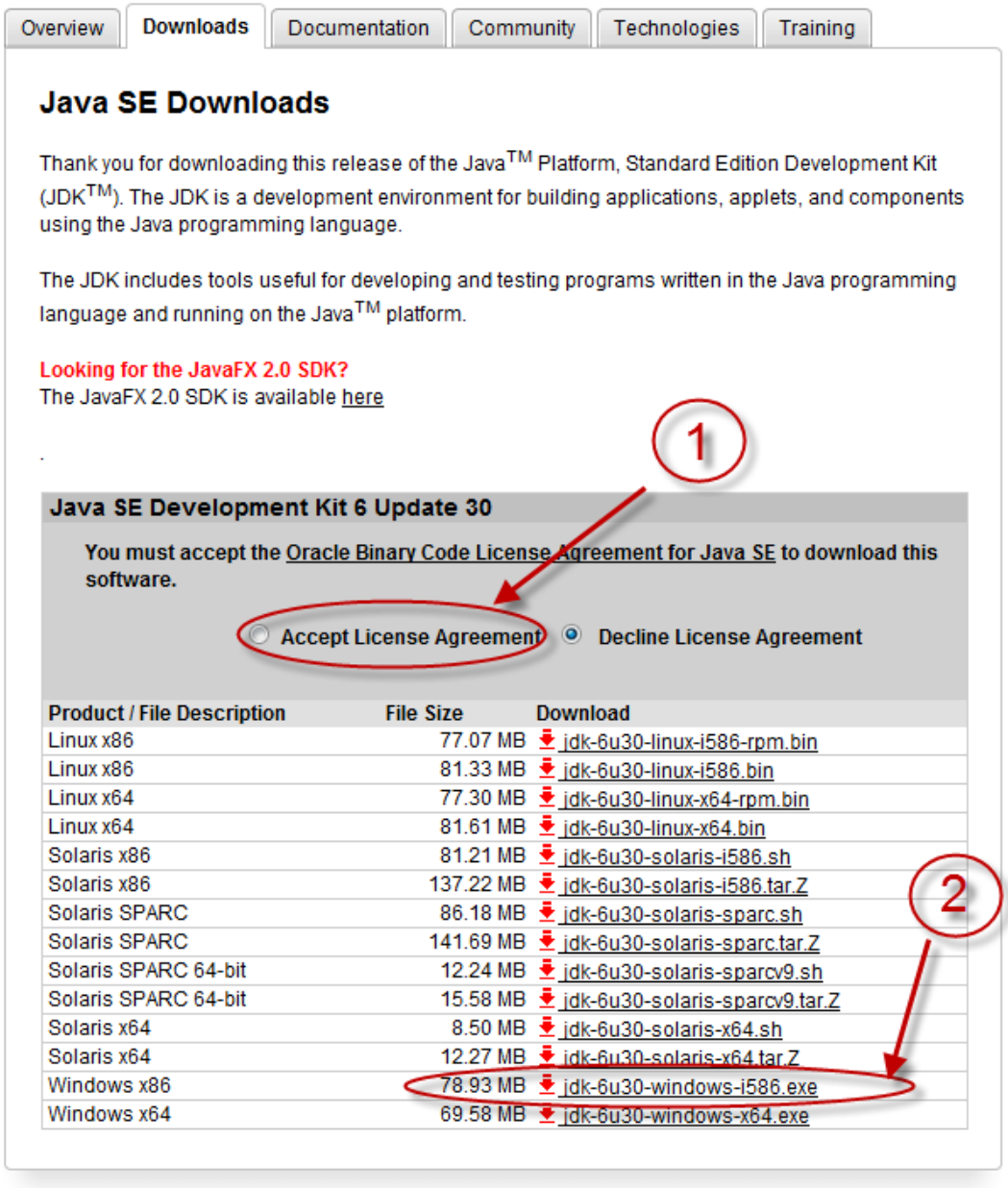
1) เปิดโปรแกรมเว็บเบราว์เซอร์แล้วไปที่ http://www.oracle.com/technetwork/java/javase/downloads/index.html เพื่อไปที่หน้าดาวโหลดโปรแกรม ซึ่งเครื่องมือทดสอบใช้ภาษาจาวาเวอร์ชัน 1.6.30 ดังภาพที่ ข.1

2) จากนั้นหน้าเว็บจะแสดงรายการโปรแกรมของภาษาจาวาในแต่ละเวอร์ชันของระบบปฏิบัติการ ซึ่งจำเป็นต้องเลือกให้ตรงกับระบบปฏิบัติการในเครื่องพีซีของนักพัฒนาด้วย ดังภาพที่ ข.2 โดยต้องยอมรับเงื่อนไขก่อนแล้วจึงสามารถดาวโหลดโปรแกรมได้

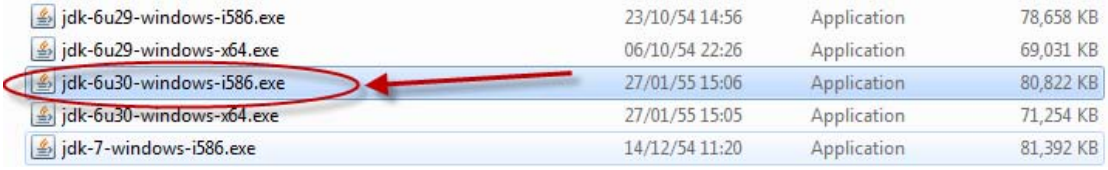


ภาพที่ ข.1 หน้าดาวโหลดโปรแกรมภาษาจาวา

3) เมื่อดาวโหลดโปรแกรมเสร็จแล้วจะได้เอกสารที่พร้อมติดตั้งโปรแกรมลงในเครื่องพีซี ดังภาพที่ ข.3

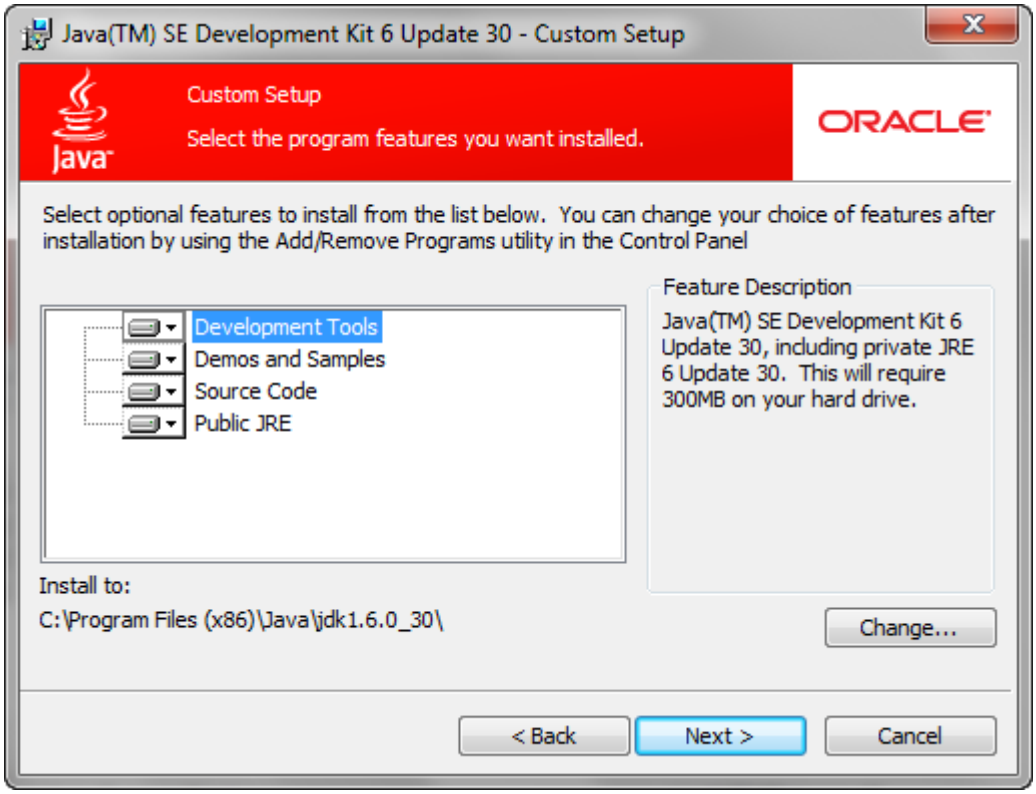


ภาพที่ ข.2 หน้าดาวโหลดโปรแกรมจาวาตามระบบปฏิบัติการ



ภาพที่ ข.3 โปรแกรมภาษาจาวาเมื่อดาวโหลดเสร็จสิ้น

4) จากนั้นทำการติดตั้งโปรแกรมภาษาจาวาโดยการดับเบิลคลิกที่โปรแกรมเพื่อติดตั้ง ดังภาพที่ ข.4 เป็นหน้าต่างแสดงไดเรกทอรี่ที่เราต้องการติดตั้งโปรแกรม

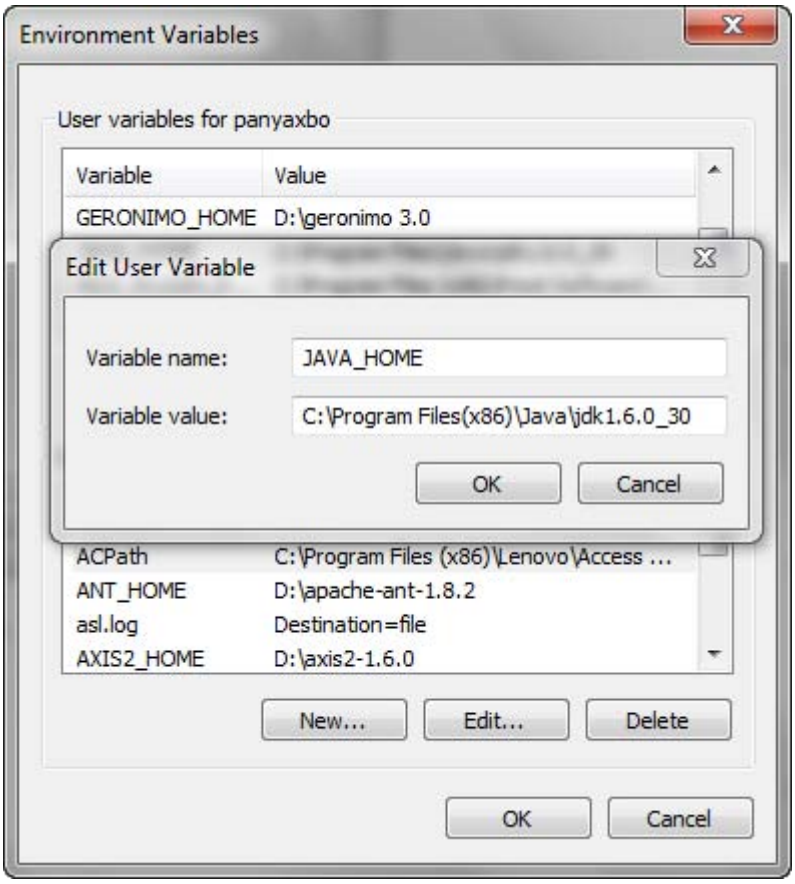


ภาพที่ ข.4 หน้าต่างการติดตั้งโปรแกรมภาษาจาวา

5) เมื่อติดตั้งโปรแกรมภาษาจาวาเสร็จสิ้น จำเป็นต้องไปกำหนดค่าตัวแปรให้สภาพแวดล้อมก่อน โดยคลิกขวาที่ My Computer -> Properties -> Advances system settings จะปรากฏหน้าต่าง Environment Variables ขึ้นมา ดังภาพที่ ข.5 จะแบ่งออกเป็น 2 ส่วน คือ User variables for *-PC-Name-* และ System variables ที่ User variables ให้กดปุ่ม New… จะปรากฏหน้าต่าง New User Variable จากนั้นให้ใส่ข้อมูล ดังนี้

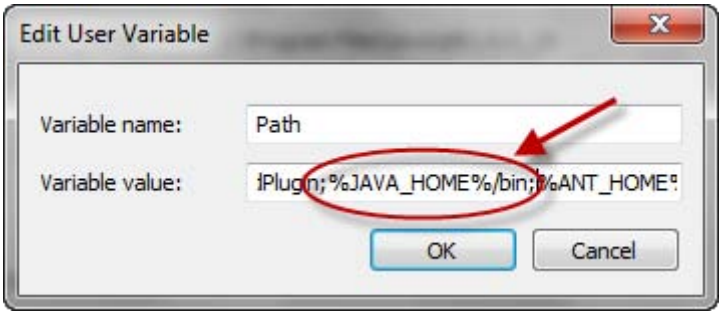
- Variable name: JAVA_HOME

- Variable value: ไดเรกทอรี่ที่เราติดตั้งโปรแกรมภาษาจาวาลงไป



ภาพที่ ข.5 หน้างต่าง Environment Variables

จากนั้นไปที่ Variable name ชื่อ Path แล้วกดปุ่ม Edit... จะปรากฏหน้าต่าง ดังภาพที่ ข.6 โดยให้เราเพิ่ม %JAVA_HOME%/bin; ในช่อง Variable value ด้วย จากนั้นกด OK เป็นการเสร็จสิ้นการติดตั้งโปรแกรมภาษาจาวา



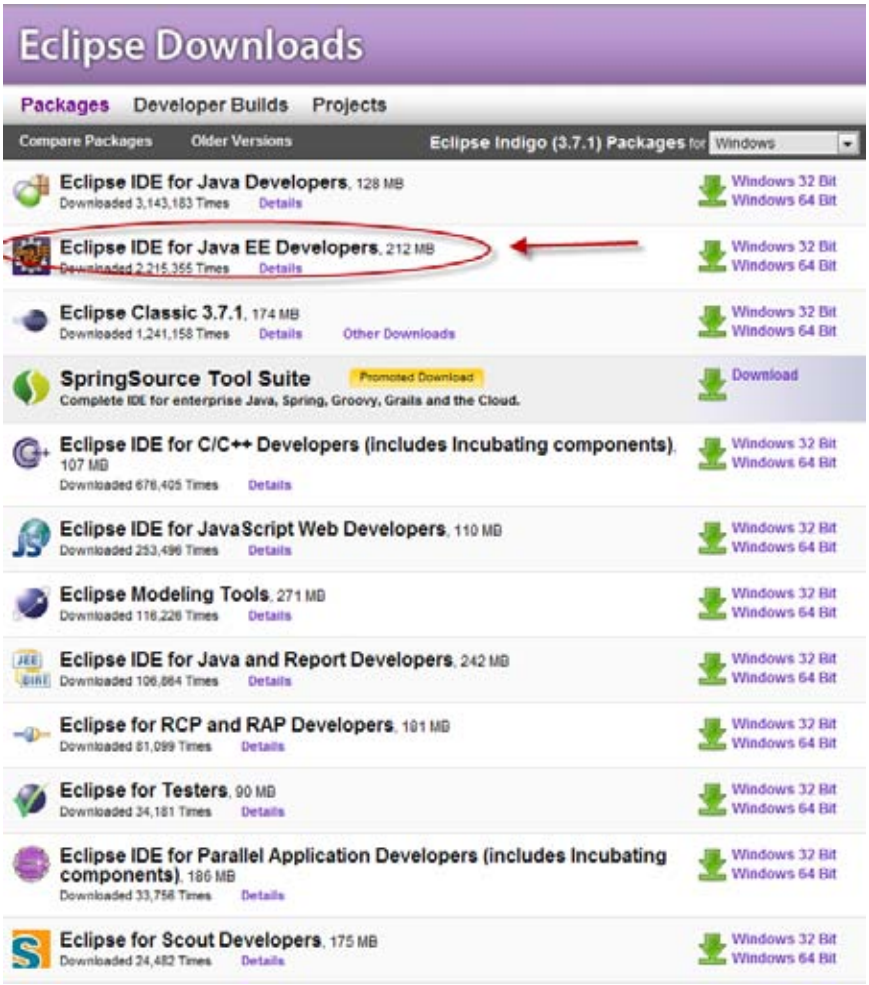
ภาพที่ ข.6 หน้าต่าง Edit User Variable

ข.2 สภาพแวดล้อมที่ใช้ในการพัฒนา (Integrated Development Environment - IDE)

ในปัจจุบันสภาพแวดล้อมที่ใช้ในการพัฒนาระบบต่างๆ มีมากมายหลายค่ายไม่ว่าจะเป็น อีคลิป เน็ทบีน ออราเคิล หรือเว็ปสเฟียร์ของไอบีเอ็ม ในวิทยานิพนธ์นี้ได้เลือกใช้อีคลิปในการพัฒนาเนื่องจากขนาดโปรแกรมไม่ใหญ่มาก ไม่ต้องติดตั้ง สามารถติดตั้งปลั๊กอินเพิ่มเติมได้ และไม่มีค่าใช้จ่าย โดยขั้นต้อนการติดตั้งมีดังต่อไปนี้

1) เปิดโปรแกรมเว็บเบราว์เซอร์เพื่อไปหน้าดาวโหลดโปรแกรมอีคลิป http://www.eclipse.org/downloads/ ของอีคลิปเพจ และดาวโหลดโปรแกรม Eclipse IDE for Java EE Developers ดังภาพที่ ข.1

2) เมื่อดาวโหลดโปรแกรมเสร็จเรียบร้อย จะได้เอกสารซิปดังภาพที่ ข.2 จากนั้นทำการคลายซิปเพื่อนำโปรแกรมออกมาใช้งาน



ภาพที่ ข.7 หน้าดาวโหลดโปรแกรมอีคลิปสำหรับนักพัฒนาภาษาจาวาระดับองค์กร

ภาพที่ ข.8 โปรแกรมอีคลิปในรูปแบบเอกสารซิป

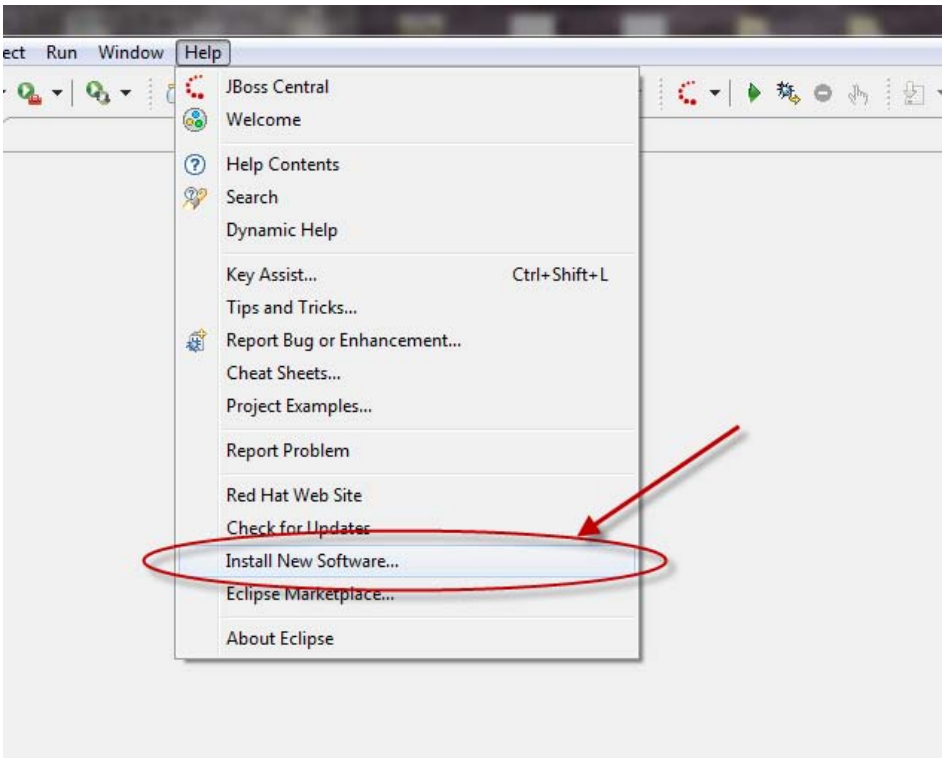
3) ทำการเปิดโปรแกรมอีคลิปขึ้นมา ดังภาพที่ ข.9 โปรแกรมจะเปิดตัวเองขึ้นมาเป็นหน้าจอการทดงานขึ้นมา แต่ในการพัฒนาเครื่องมือนั้นจำเป็นต้องติดตั้งโปรแกรมเสริม (plug-in) ซึ่งโปรแกรมเสริมที่เราต้องติดตั้งเพิ่ม คือ JBoss Tools 3.3 เพื่อใช้งานภาษาเฟรมเวิร์กของจาวาเซิร์ฟเวอร์เฟสและ BPEL Eclipse Designer เพื่อใช้ในการออกแบบกระบวนการบีเพล โปแกรมเสริมเหล่านี้จะช่วยทำให้เราพัฒนาเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลได้ง่ายขึ้น

3.1) ติดตั้งโปรแกรมเสริม JBoss Tools 3.3 โดยขั้นตอนการติดตั้งไปที่ Help -> Install New Software… ดังภาพที่ ข.10 จะปรากฏหน้าต่างดังรูป ข.11 จากนั้นกดปุ่ม Add… จะแสดงหน้าต่าง Add Repository ให้ใส่ข้อวามดังนี้

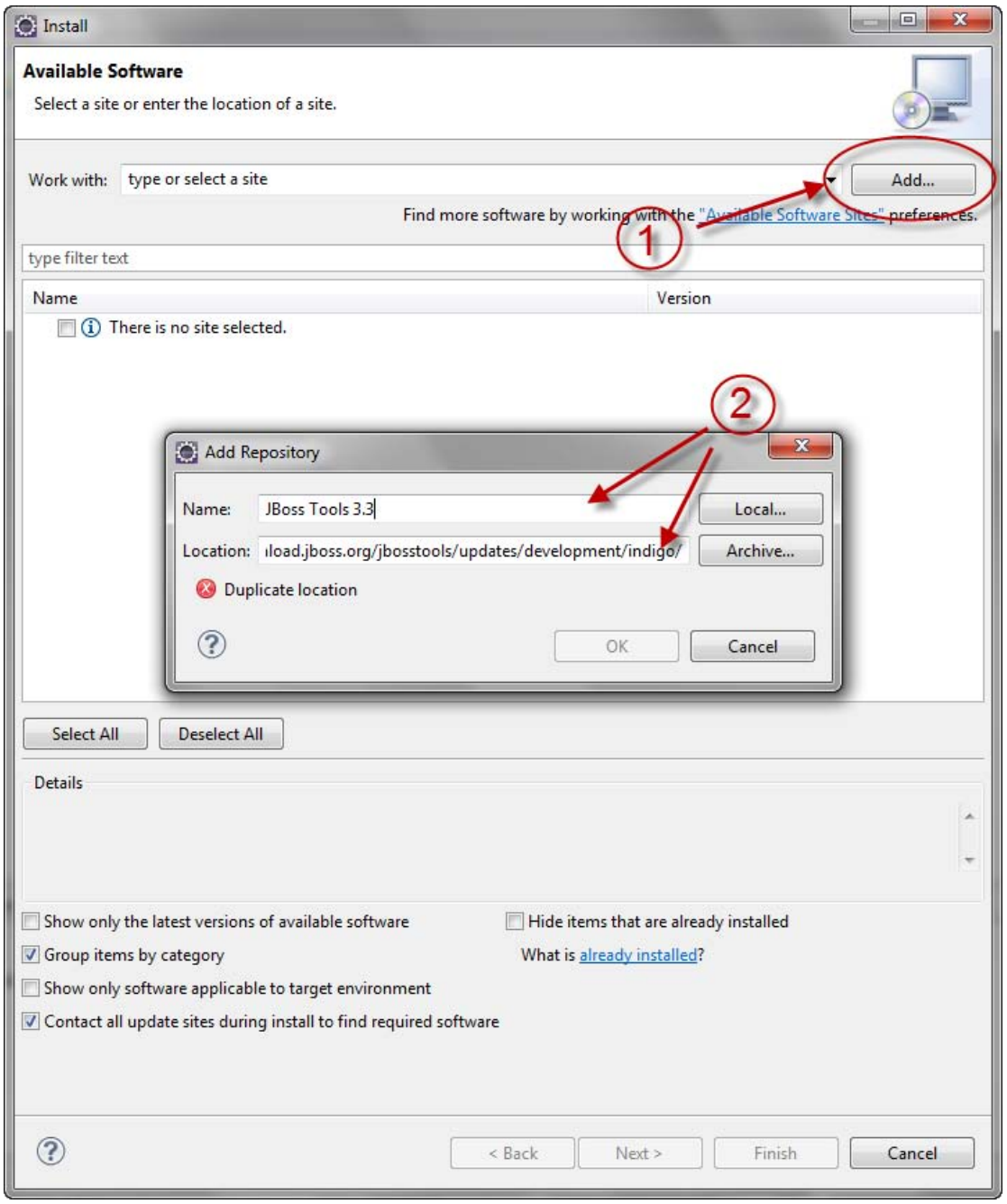
- Name: Jboss Tools 3.3
- Locations: http://download.jboss.org/jbosstools/updates/development/indigo/

จากนั้นกดปุ่ม OK โปรแกรมจะค้นหารายชื่อโปรแกรมเสริม เมื่อได้แล้วจะแสดงหน้าต่าง ดังภาพที่ ข.12 ทำการเลือกโปรแกรมเสริมที่ต้องการติดตั้งแล้วกดปุ่ม Next > โปรแกรมจะแสดงหน้าต่างแสดงารายละเอียดของโปรแกรมต่างๆ ที่จะถูกติดตั้งแล้วเลือก Next > เพื่อติดตั้งโปรแกรม รอจนกระทั่งโปรแกรมอีคลิปติดตั้งโปรแกรม JBoss Tools 3.3 เรียบร้อยจะขึ้นหน้าต่างให้ Restart Now

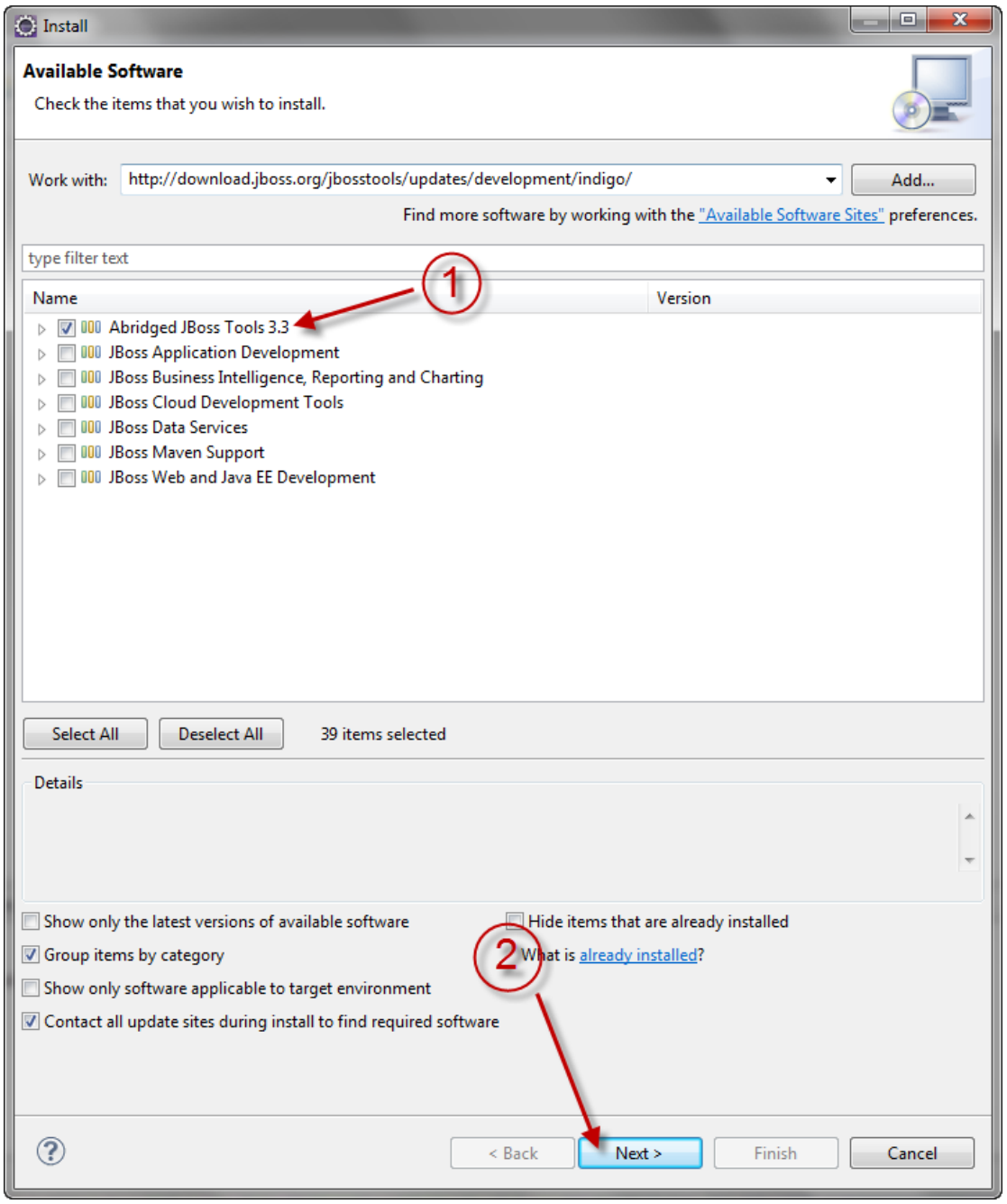
ภาพที่ ข.9 หน้าต่างโปรแกรมอีคลิป



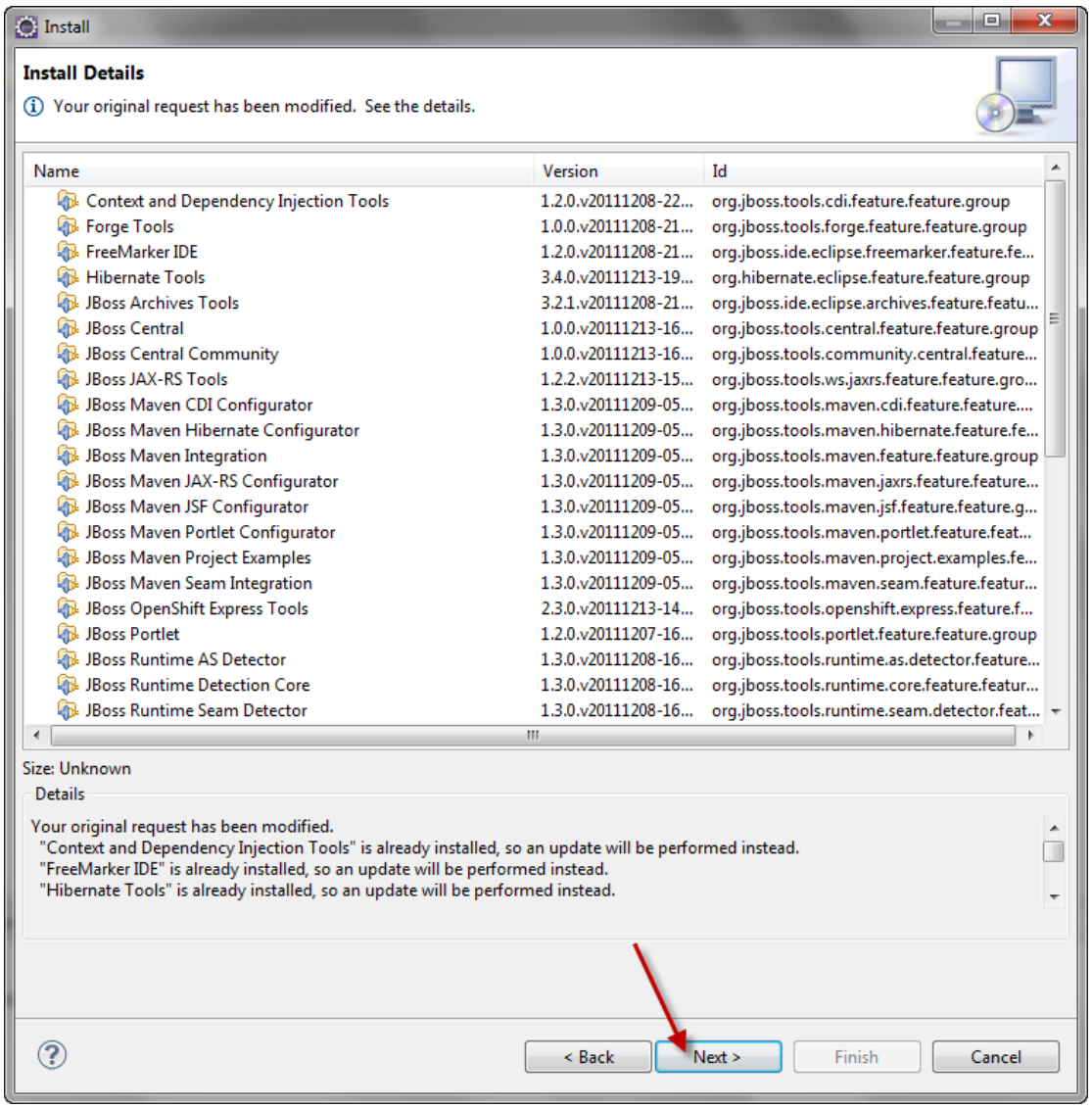
ภาพที่ ข.10 เมนูติดตั้งโปรแกรมเสริม



ภาพที่ ข.11 ติดตั้งโปรแกรมเสริม JBoss Tools 3.3



ภาพที่ ข.12 หน้าต่างรายการโปรแกรมเสริมของ JBoss Tools 3.3



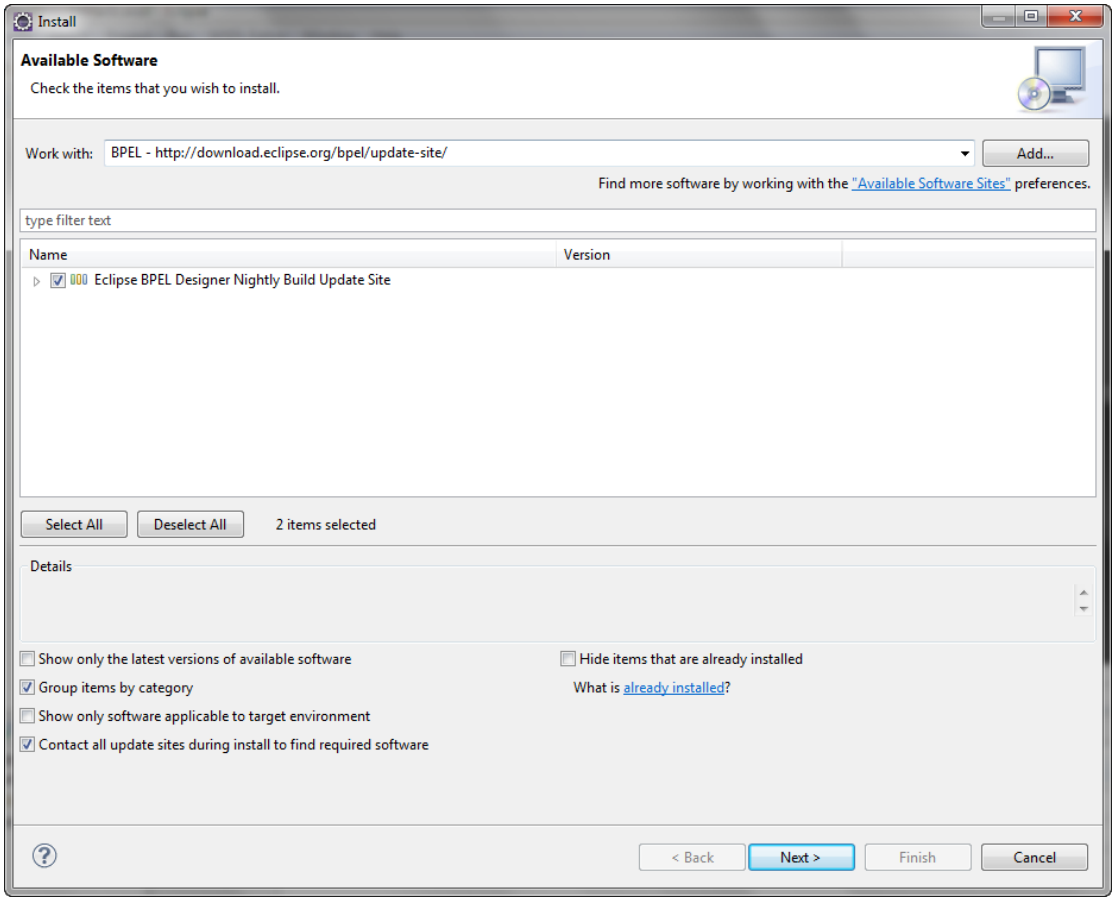
ภาพที่ ข.13 หน้าต่างรายละเอียดการติดตั้งโปรแกรม JBoss Tools 3.3

3.2) ติดตั้งโปรแกรม BPEL Eclipse Designer เป็นอีกโปรแกรมที่ทำให้นักพัฒนาสามารถออกแบบกระบวนการบีเพลได้ง่ายขึ้น โดยขั้นตอนการติดตั้งจะเหมือนกับ JBOss Tools 3.3 นั้นคือ Help -> Install New Software... จะปรากฏหน้าต่างขึ้นมาให้กดที่ Add... เพื่อติดตั้งโปรแกรมเสริมโดยใส่ข้อความดังนี้

Name: BPEL

Location: http://download.eclipse.org/bpel/update-site/

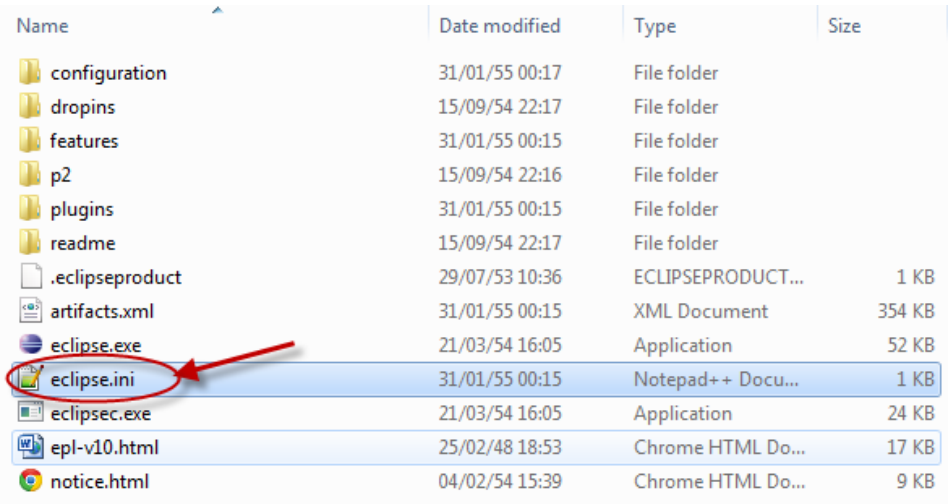
จะได้ดังภาพที่ ข.14 จากนั้นขั้นตอนการติดตั้งโปรแกรมจะเหมือนกับการติดตั้งโปรแกรม JBoss Tools 3.3



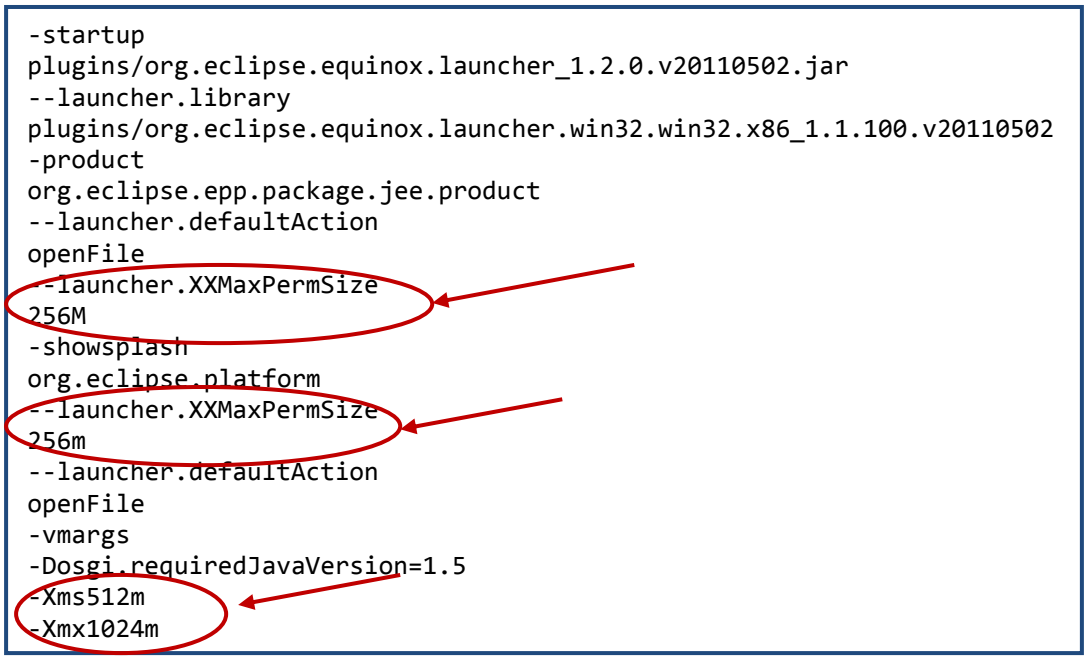
ภาพที่ ข.14 หน้าต่างการติดตั้งโปรแกรม BPEL Eclipse Designer

*** **หมายเหตุ**: ในการระหว่างการพัฒนาเครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลนั้น เนื่องจากโปรแกรมทรัพยากรดั้งเดิมของโปรแกรมอีคลิปได้ถูกกำหนดให้กินทรัพยากรไว้ค่าหนึ่ง ซึ่งถ้าเมื่อเราติดตั้งโปรแกรมอื่นๆ เพิ่มมากขึ้นการใช้งานโปรแกรมอีคลิปก็จะมากขึ้น ทำให้กินทรัพยากรมากขึ้นไปด้วยและอาจทำให้โปรแกรมอีคลิปนั้น กินทรัพยากรเกินค่าที่กำหนดไว้เริ่มต้น ดังนั้นอาจจะเกิดปัญหา Error: Java heap space ขึ้นได้ ดังนั้นการแก้ไขต้องเพิ่มทรัพยากรให้โปรแกรมอีคลิปสามารถใช้ได้มากขึ้น

เพื่อแก้ไขปัญหาดังกล่าวจำเป็นต้องไปแก้ไขโปรแกรม โดยไปที่โปรแกรมอีคลิป ECLIPSE_HOME ที่เป็นไดเรกทอรี่ของอีคลิป จากนั้นเปิดเอกสาร eclipse.ini ดังภาพที่ ข.15 เมื่อเปิดเอกสารขึ้นมาจะแสดงข้อความดังภาพที่ ข.16 โดยเราสามารถกำหนดข้อมูลของ MinPermSize และ MaxPermSize เพื่อให้โปรแกรมกินทรพยากรได้มากขึ้น



ภาพที่ ข.15 การแก้ไขทรัพยากรของโปรแกรมอีคลิป



ภาพที่ ข.16 รายละเอียดของเอกสาร eclipse.ini

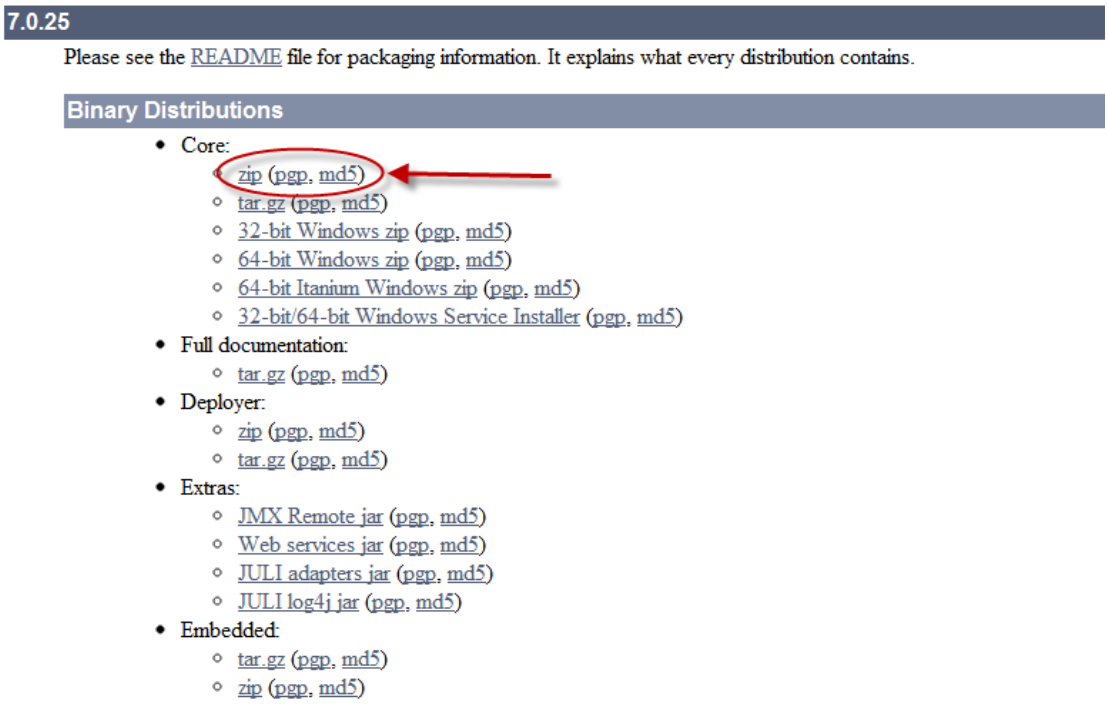
ข.3 การติดตั้งเครื่องบริการเว็บและเครื่องประมวลผลบีเพล

ในส่วนนี้จะอธิบายถึงการติดตั้งเซิร์ฟเวอร์ซึ่งจะแบ่งออกเป็นการติดตั้งเครื่องบริการเว็บ (Web Server) และเครื่องประมวลผลบีเพล (BPEL Engine)

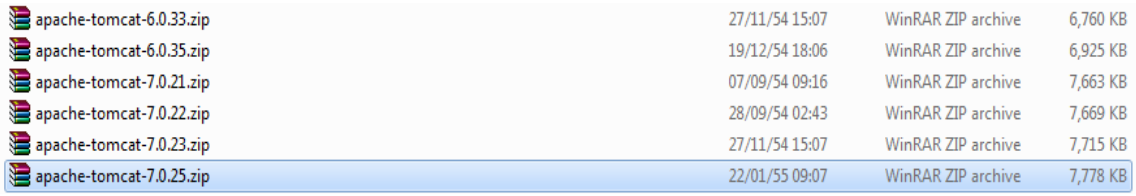
ข.3.1 เครื่องบริการเว็บ

เครื่องบริการเว็บนั้นเป็นสภาพแวดล้อมที่ให้เครื่องมือทดสอบวีคมิวเทชันสามารถทำงานได้ ซึ่งในท้องตลาดนั้นมีหลากหลายค่ายไม่ว่าจะเป็น อาพาเช่ ทอมแคท กลาสฟิช แอพลิเคชัน เซิร์ฟเวอร์ หรือเจบอส แอพลิเคชัน เซิร์ฟเวอร์ เป็นต้น แต่เครื่องบริการเว็บที่ใช้เป็นสภาพแวดล้อมการทำงานให้เครื่องมือทดสอบวีคมิวเทชันนั้น นักวิจัยได้เลือกใช้อาพาเช่ ทอมแคท เนื่องจากขนาดเอกสารไม่ใหญ่มาก หลายคนคุ้นเคยกับเครื่องบริการเว็บชนิดนี้ และง่ายต่อการใช้งาน โดยขั้นตอนการติดตั้งมีดังต่อไปนี้

1) เปิดโปรแกรมเว็บเบราว์เซอร์แล้วไปที่เพจ http://tomcat.apache.org/download-70.cgi เพื่อดาวโหลดเครื่องบริการเว็บ ซึ่งปัจจุบันอาพาเช่ ทอมแคทนั้นมีหลากหลายเวอร์ชัน โดยเวอร์ชันปัจจุบันได้อยู่ที่เวอร์ชัน 7.0.25 ดังภาพที่ ข.17 ที่ Binary Distributions ที่ Core ทำการดาวโหลดแบบซิป เมื่อดาวโหลดเสร็จสิ้นจะได้เอกสาร ดังภาพที่ ข.18



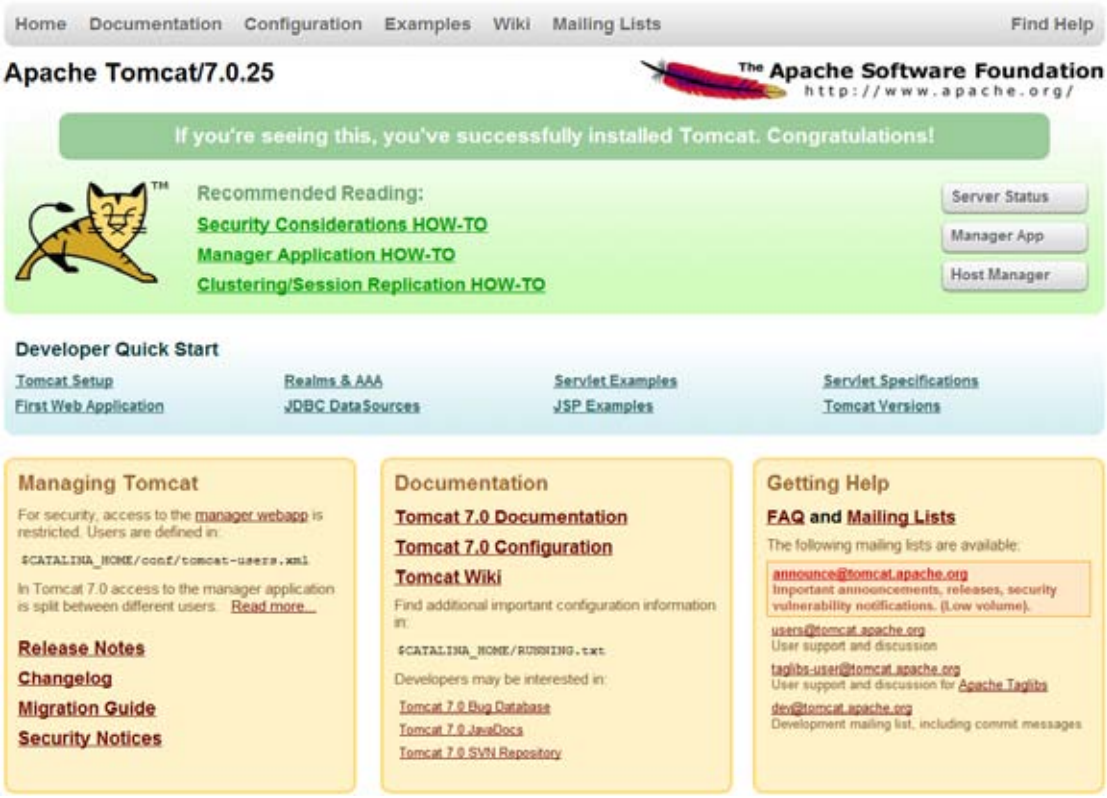
ภาพที่ ข.17 หน้าจอดาวโหลดเครื่องบริการเว็บอาพาเช่ทอมแคท



ภาพที่ ข.18 เครื่องบริการเว็บอาพาเช่ทอมแคทเมื่อดาวโหลดเสร็จสิ้น

2) หลังจากนั้นคลายเอกสารซิปเพื่อติดตั้งโปรแกรม เราต้องไปกำหนดสภาพแวดล้อมของตัวแปร (Environment Variables) ดังที่กล่าวไปในส่วนของการติดตั้งโปรแกรมภาษาจาวา โดยต้องกำหนดตัวแปรชื่อ (Variable name) เป็น CATALINA_HOME ส่วนค่าของตัวแปรจะกำหนดเป็นไดเรกทอรี่ของอาพาเช่ทอมแคท

3) ทดสอบเครื่องบริการเว็บว่าสามารถใช้งานได้หรือไม่ โดยไปที่ไดเรกทอรี่ของอาพาเช่ทอมแคท CATALINA_HOME/bin ดับเบิลคลิกเอกสาร startup.bat เมื่อเซิร์ฟเวอร์สตาร์ทเสร็จสิ้น ทำการเปิดโปรแกรมเว็บเบราว์เซอร์แล้วใส่ที่อยู่ http://localhost:8080/ หรือ http://127.0.0.1:8080/ ถ้าเบราว์เซอร์แสดงหน้าจอดังภาพที่ ข.19 แสดงว่าการติดตั้งเสร็จสมบูรณ์

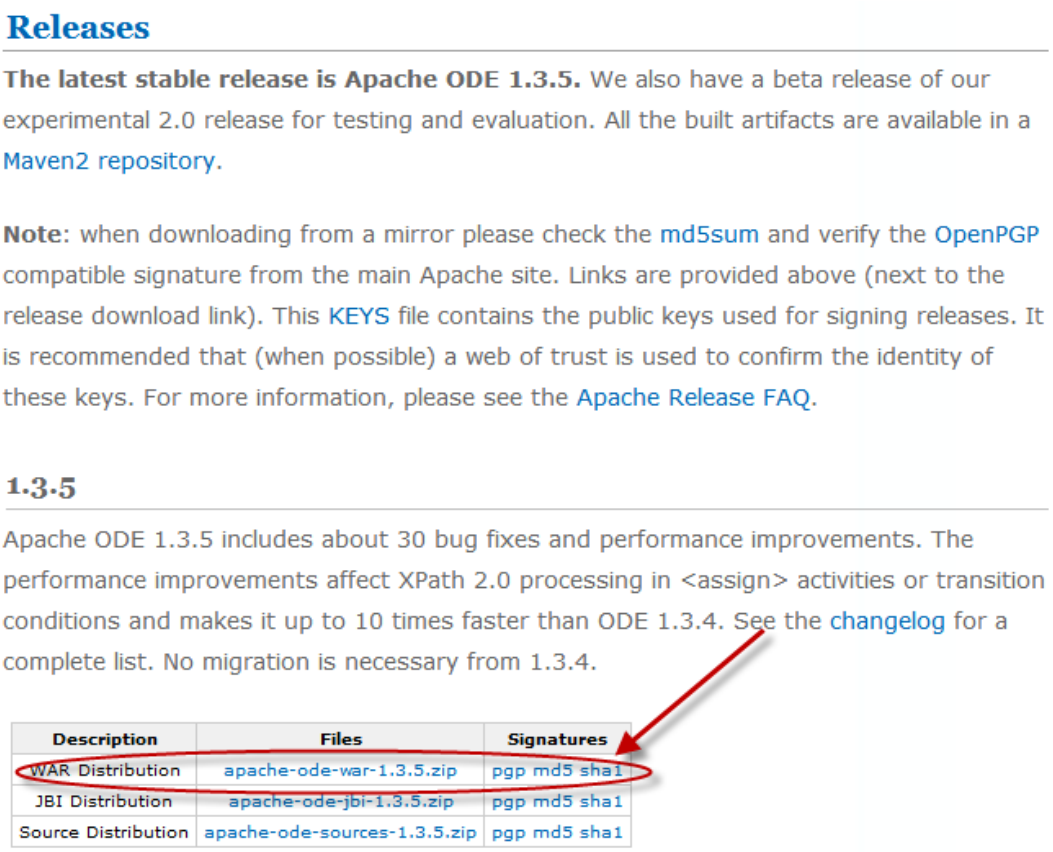


ภาพที่ ข.19 หน้าจออาพาเช่ทอมแคทเมื่อรันเซิร์ฟเวอร์

ข.3.2 เครื่องประมวลผลบีเพล

เครื่องประมวลผลบีเพลในปัจจุบันนั้นก็มีให้เลือกใช้หลากหลายค่าย เช่นเดียวกับเครื่องบริการเว็บเช่นกัน ตัวอย่างเช่น เว็บสเฟียร์ของไอบีเอ็ม ออราเคิลเอสโอเอสูท และอาพาเช่โอดีอี เป็นต้น ซึ่งในวิทยานิพนธ์นี้ได้เลือกอาพาเช่โอดีอี เนื่องจากขนาดเอกสารน้อย ติดตั้งง่าย และไม่มีค่าใช้จ่ายเพิ่มเติม เป็นต้น ในกระบวนการติดตั้งเครื่องประมวลผลบีเพลมีขั้นตอน ดังนี้

1) เปิดโปรแกรมเว็บเบราว์เซอร์เพื่อไปหน้าดาวโหลดเครื่องประมวลผลบีเพลที่ http://ode.apache.org/getting-ode.html ซึ่งในปัจจุบันอาพาเช่โอดีอีอยู่ที่เวอร์ชัน 1.3.5 เลือกดาวโหลดที่ WAR Distribution เอกสารชื่อ apache-ode-1.3.5.zip ดังภาพที่ ข.20 เมื่อดาวโหลดเสร็จสิ้นจะได้เอกสารซิป ดังภาพที่ ข.21



Releases

The latest stable release is Apache ODE 1.3.5. We also have a beta release of our experimental 2.0 release for testing and evaluation. All the built artifacts are available in a Maven2 repository.

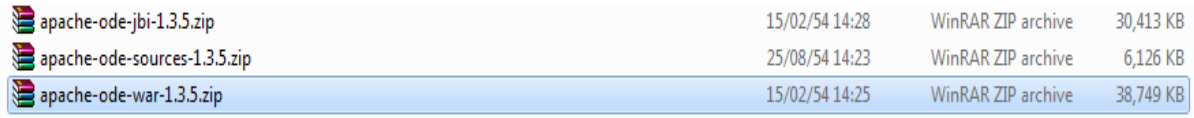
Note: when downloading from a mirror please check the md5sum and verify the OpenPGP compatible signature from the main Apache site. Links are provided above (next to the release download link). This KEYS file contains the public keys used for signing releases. It is recommended that (when possible) a web of trust is used to confirm the identity of these keys. For more information, please see the Apache Release FAQ.

1.3.5

Apache ODE 1.3.5 includes about 30 bug fixes and performance improvements. The performance improvements affect XPath 2.0 processing in <assign> activities or transition conditions and makes it up to 10 times faster than ODE 1.3.4. See the changelog for a complete list. No migration is necessary from 1.3.4.

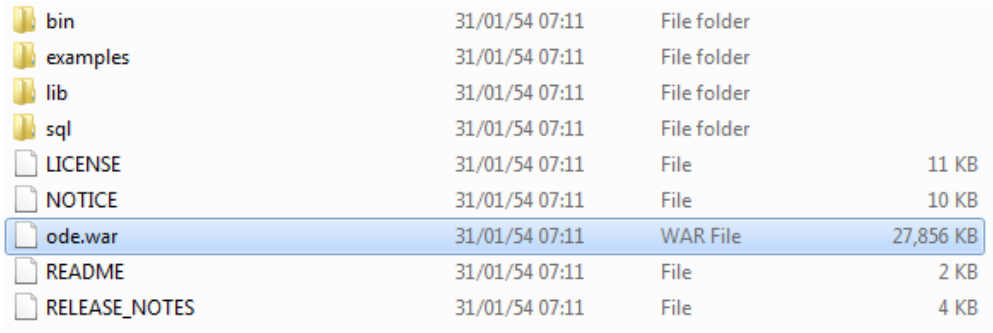
Description	Files	Signatures
WAR Distribution	apache-ode-war-1.3.5.zip	pgp md5 sha1
JBI Distribution	apache-ode-jbi-1.3.5.zip	pgp md5 sha1
Source Distribution	apache-ode-sources-1.3.5.zip	pgp md5 sha1

ภาพที่ ข.20 หน้าจอดาวโหลดเครื่องประมวลผลบีเพล

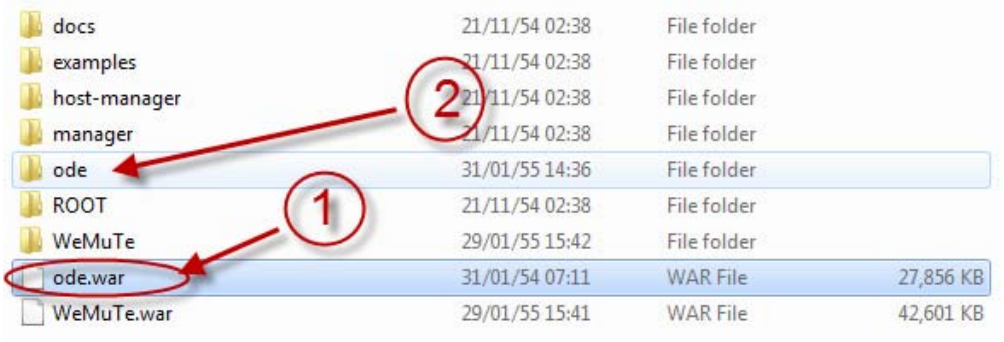


ภาพที่ ข.21 เครื่องประมวลผลบีเพลเมื่อดาวโหลดเสร็จสิ้น

2) คลายซิปแล้วจะสังเกตเห็นเอกสาร ode.war ดังภาพที่ ข.22 ในซึ่งเครื่องประมวลผลบีเพลก็เสมือนกับเว็บแอพลิเคชันตัวหนึ่ง ซึ่งต้องการสภาพแวดล้อมในการทำงานดังนั้นจึงจำเป็นต้องคัดลอกเอกสาร ode.war ไปไว้ที่อาพาเช่ทอมแคทเพื่อให้เครื่องประมวลผลบีเพลสามารถทำงานได้ โดยนำเอกสารไปไว้ที่ CATALINA_HOME/wabapps ดังภาพที่ ข.23 จากนั้นทำการรันเครื่องบริการเว็บโดยดับเบิลคลิกเอกสาร startup.bat ในไดเรกทอรี่ CATALINA_HOME/bin เมื่อเครื่องบริการเว็บรันสำเร็จเราจะได้เครื่องประมวลผลบีเพลในรูปแบบไดเรกทอรี่ ดังภาพที่ ข.23

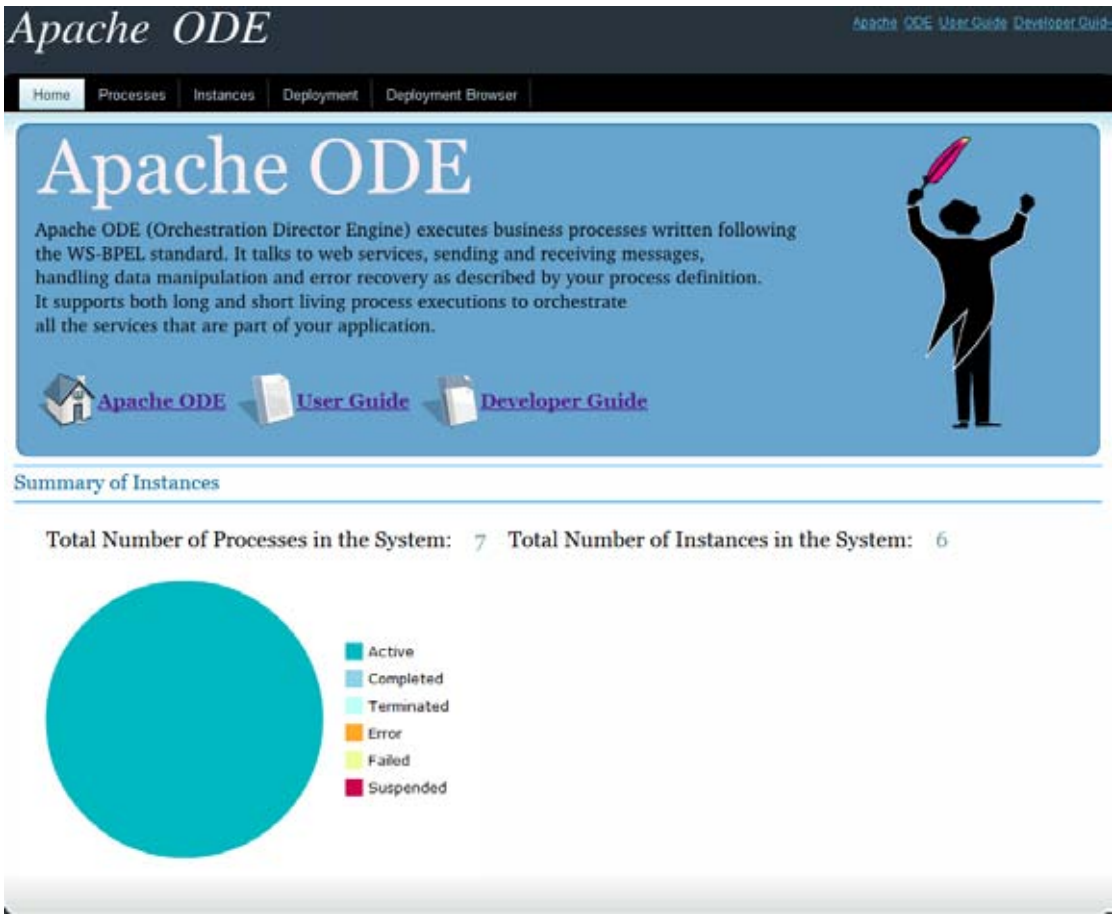


ภาพที่ ข.22 เอกสารภายในเครื่องประมวลผลบีเพล



ภาพที่ ข.23 การติดตั้งเครื่องประมวลผลบีเพลบนอาพาเช่ทอมแคท

3) ทดสอบเครื่องเครื่องประมวลผลบีเพลว่าสามารถใช้งานได้หรือไม่ โดยไปที่ไดเรกทอรี่ของอาพาเช่ทอมแคท CATALINA_HOME/bin ดับเบิลคลิกเอกสาร startup.bat เมื่อเซิร์ฟเวอร์สตาร์ทเสร็จสิ้น ทำการเปิดโปรแกรมเว็บเบราว์เซอร์แล้วใส่ที่อยู่ http://localhost:8080/ode/ หรือ http://127.0.0.1:8080/ode ถ้าเบราว์เซอร์แสดงหน้าจอดังภาพที่ ข.24 แสดงว่าการติดตั้งเสร็จสมบูรณ์



ภาพที่ ข.24 หน้าจอของเครื่องประมวลผลบีเพล

*** **ข้อควรระวัง**: ถ้าในเครื่องพีซีที่พัฒนาเกิดมีการใช้พอร์ต 8080 อยู่ จะทำให้ไม่สามารถใช้งานเครื่องบริการเว็บอาพาเช่ทอมแคทได้ ดังนั้นจึงจำเป็นต้องเปลี่ยนค่าของพอร์ตที่ใช้งานให้กับเครื่องบริการเว็บรวมทั้งเอกสารบางเอกสารในเครื่องประมวลผลด้วยเพื่อไม่ให้พอร์ตชนกัน (port conflict) และอาจมีปัญหาการใช้งานอื่นๆ ได้

จากปัญหาดังกล่าวสามารถแก้ไขโดยแบ่งออกเป็น 2 ส่วน คือ แก้ไขที่เครื่องบริการเว็บ และ แก้ไขที่เครื่องประมวลบีเพล ดังนี้

1) เครื่องบริการเว็บ

โดยต้องไปไปแก้ไขเอกสาร server.xml ในไดเรกทอรี่ CATALINA_HOME/conf เมื่อเปิดเอกสารขึ้นมา ดังภาพที่ ข.25 ในส่วนของอีลีเมนต์ Connector จะมีแอตทริบิวต์ port โดยให้แก้ไขเป็นค่าอื่นที่ไม่ใช่ 8080

```
<!-- A "Connector" represents an endpoint by which requests are received
     and responses are returned. Documentation at :
     Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
     Java AJP  Connector: /docs/config/ajp.html
     APR (HTTP/AJP) Connector: /docs/apr.html
     Define a non-SSL HTTP/1.1 Connector on port 8080
-->
<Connector port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
<!-- A "Connector" using the shared thread pool-->
<!--
<Connector executor="tomcatThreadPool"
           port="8080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
-->
```

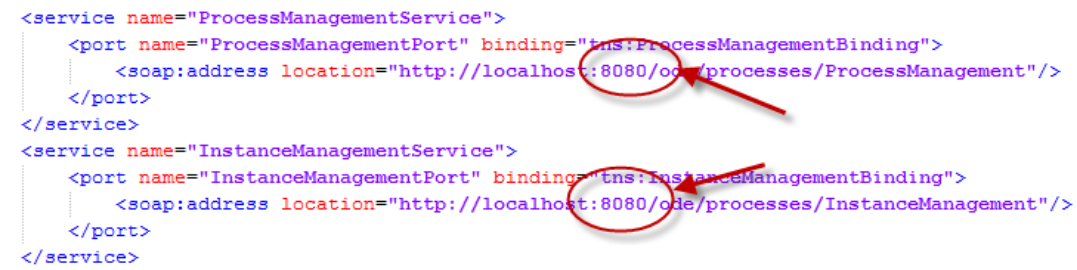
ภาพที่ ข.25 การแก้ไขพอร์ตในอาพาเช่ทอมแคทเอกสาร server.xml

2) เครื่องประมวลผลบีเพล

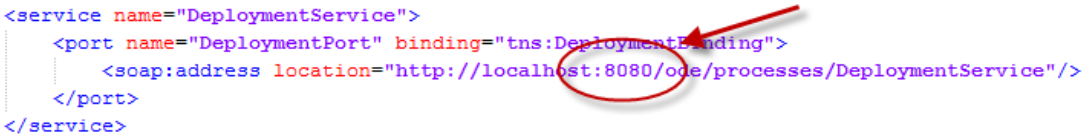
โดยต้องแก้ไขทั้งหมด 3 เอกสาร คือ fileupload.jsp ในไดเรกทอรี่ ODE_HOME/ เมื่อเปิดเอกสารขึ้นมา หาส่วนของโค้ดดังภาพที่ ข.26 แล้วเปลี่ยนค่า 8080 เป็นค่าอื่น เอกสารต่อมา คือ pmapi.wsdl ดังภาพที่ ข.27 ในอีลีเมนต์ service มีอีลีเมนต์ย่อย soap:address เปลี่ยนค่าพอร์ต 8080 เป็นค่าอื่น และเอกสาร deploy.wsdl ซึ่งอยู่ในอีลีเมนต์ soap:address เช่นกัน ดังภาพที่ ข.28

```
if (offset < bytes.length) {
    out.println("Overflow Error Occurred!");
}else{
    if(!Base64.isArrayByteBase64(bytes)){
        byte[] encodedBytes  = Base64.encodeBase64(bytes);
        String encodedString = new String(encodedBytes);
        Options opts = new Options();
        opts.setAction("http://www.apache.org/ode/deployapi/DeploymentPortType/deployRequest");
        opts.setSoapVersionURI(SOAP12Constants.SOAP_ENVELOPE_NAMESPACE_URI);
        opts.setProperty(Constants.Configuration.HTTP_METHOD,
            Constants.Configuration.HTTP_METHOD_POST);
        opts.setTo(new EndpointReference("http://localhost:8080/ode/processes/DeploymentService"));
```

ภาพที่ ข.26 แก้ไขพอร์ตในเครื่องประมวลผลบีเพล fileupload.jsp



ภาพที่ ข.27 แก้ไขพอร์ตในเครื่องประมวลผลบีเพล pmapi.wsdl



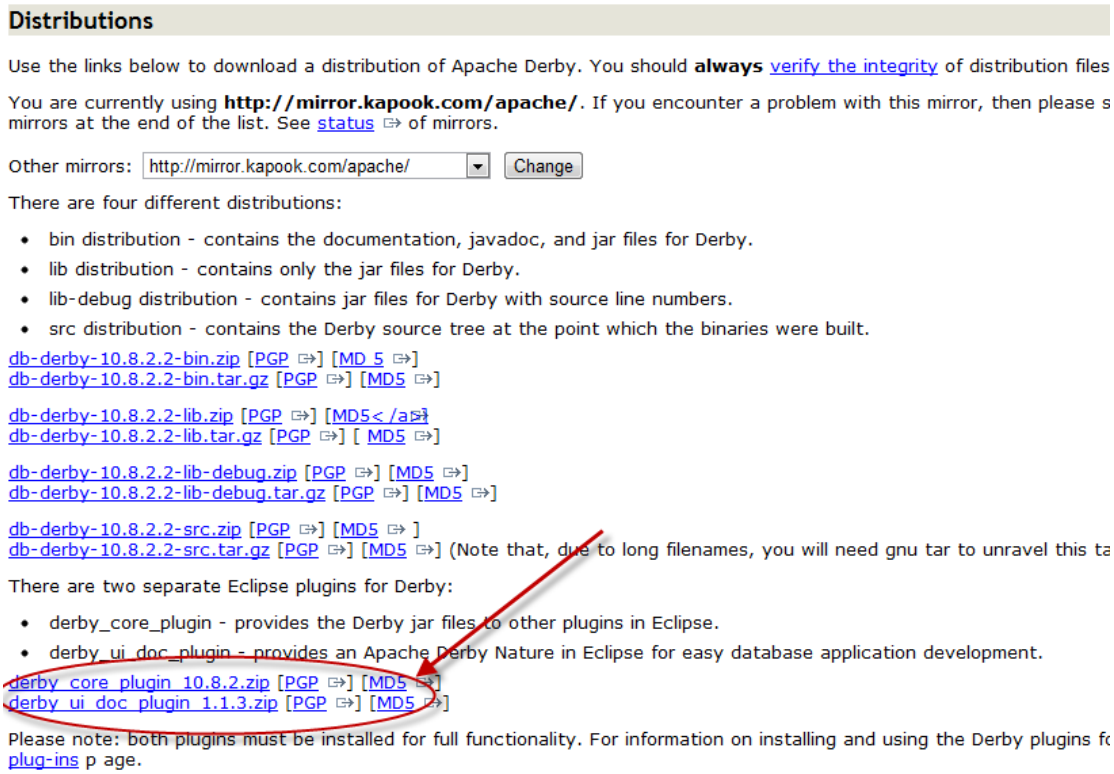
ภาพที่ ข.28 แก้ไขพอร์ตในเครื่องประมวลผลบีเพล deploy.wsdl

ข.4 ฐานข้อมูลที่เว็บเซอร์วิซเรียกใช้

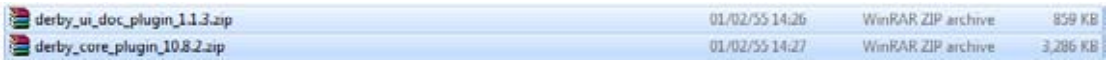
ในปัจจุบันฐานข้อมูลที่ใช้พัฒนาเว็บแอพลิเคชันนั้นมีมากมายหลายค่าย ไม่ว่าจะเป็น PostgreSQL MySQL Oracle Microsoft SQL Server Apache Derby และอื่นๆ ซึ่งในวิทยานิพนธ์นี้ได้เลือกใช้ฐานข้อมูล Apache Derby เนื่องจากติดตั้งง่ายและเป็นโปรแกรมเสริมซึ่งสามารถติดตั้งและใช้งานในโปรแกรมอีคลิปได้ ซึ่งมีขั้นตอนการติดตั้งดังนี้

1) เปิดโปรแกรมเว็บเบราว์เซอร์แล้วไปที่หน้าเพจ http://db.apache.org/derby/ จากนั้นเลือกที่แท็ป Download จะปรากฏหน้าเว็บดังภาพที่ ข.29 ซึ่งปัจจุบันจะอยู่ที่เวอร์ชัน 10.8.2.2 ให้เลือกดาวโหลด 2 เอกสาร คือ derby_core_plugin_10.8.2.zip และ derby_ui_doc_plugin_1.1.3.zip ดังภาพที่ ข.30

2) เมื่อดาวโหลดเอกสารเรียบร้อยจะได้เอกสารดังภาพที่ ข.26 จากนั้นคลายซิปออกเพื่อนำข้อมูลใน derby_core_plugin_10.8.2/plugins/… และ derby_ui_doc_plugin_1.1.3 /plugins/… นำเอกสารที่อยู่ในภาพที่ ข.31 และ ข.32 ไปไว้ในโปรแกรมอีคลิป ECLIPSE_HOME/plugins/



ภาพที่ ข.29 หน้าต่างดาวโหลดโปรแกรมเสริมของ Apache Derby

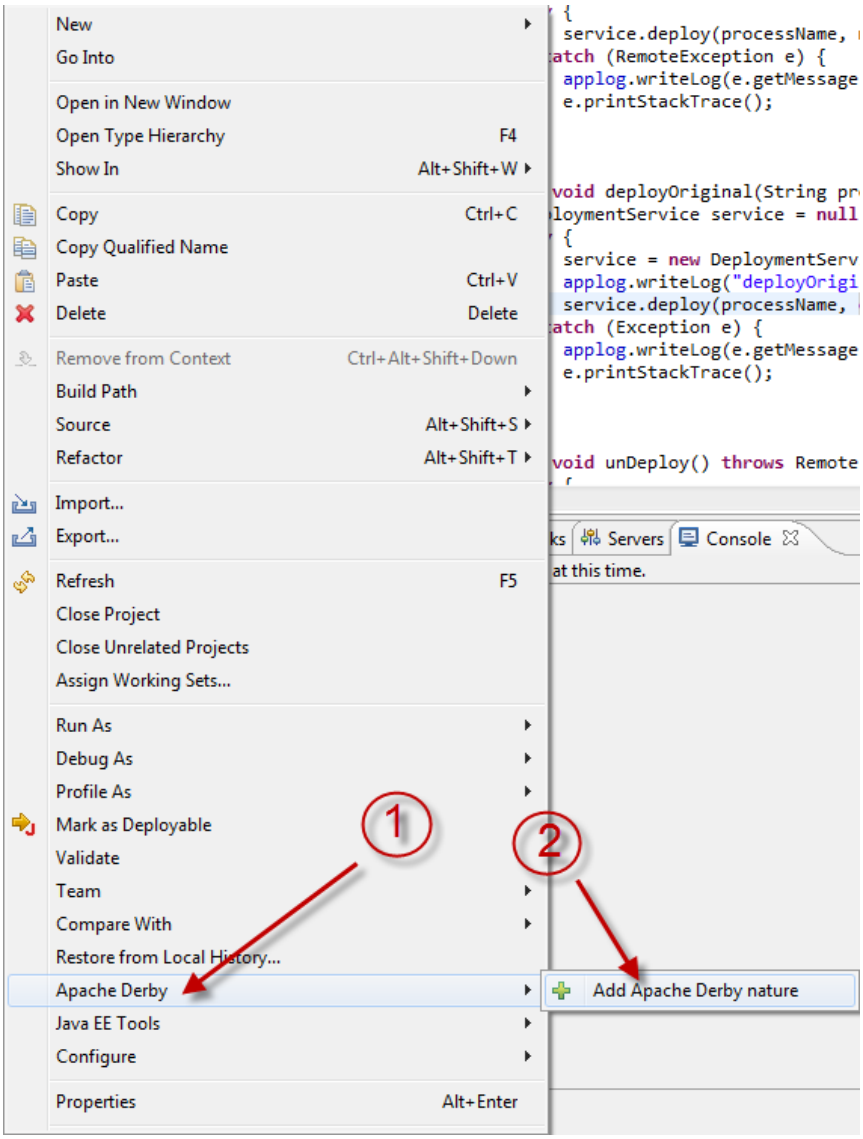


ภาพที่ ข.30 เอกสารซิปของโปรแกรมเสริม Apache Derby เมื่อดาวโหลดเสร็จสิ้น

ภาพที่ ข.31 เอกสารภายใน derby_core_plugin_10.8.2/plugins

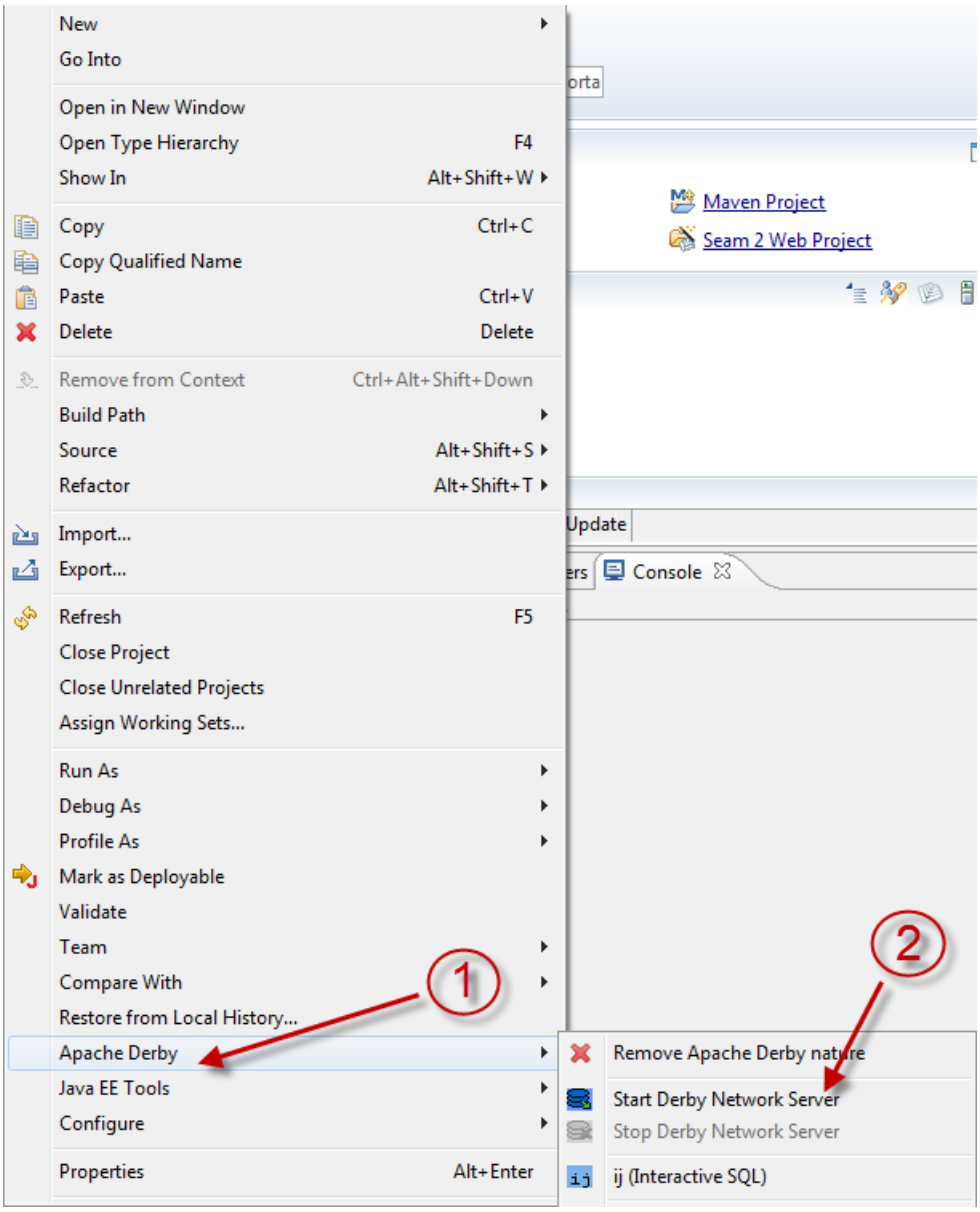
ภาพที่ ข.32 เอกสารภายใน derby_ui-plugin_1.1.3/plugins

3) เมื่อสตาร์ทโปรแกรมอีคลิปเลือกที่โครงงานที่เราต้องการ คลิกขวาเลือก Apache Derby -> Add Apache Derby nature ดังภาพที่ ข.33



ภาพที่ ข.33 การเพิ่มความสามารถ Apache Derby ลงในโครงงานในโปรแกรมอีคลิป

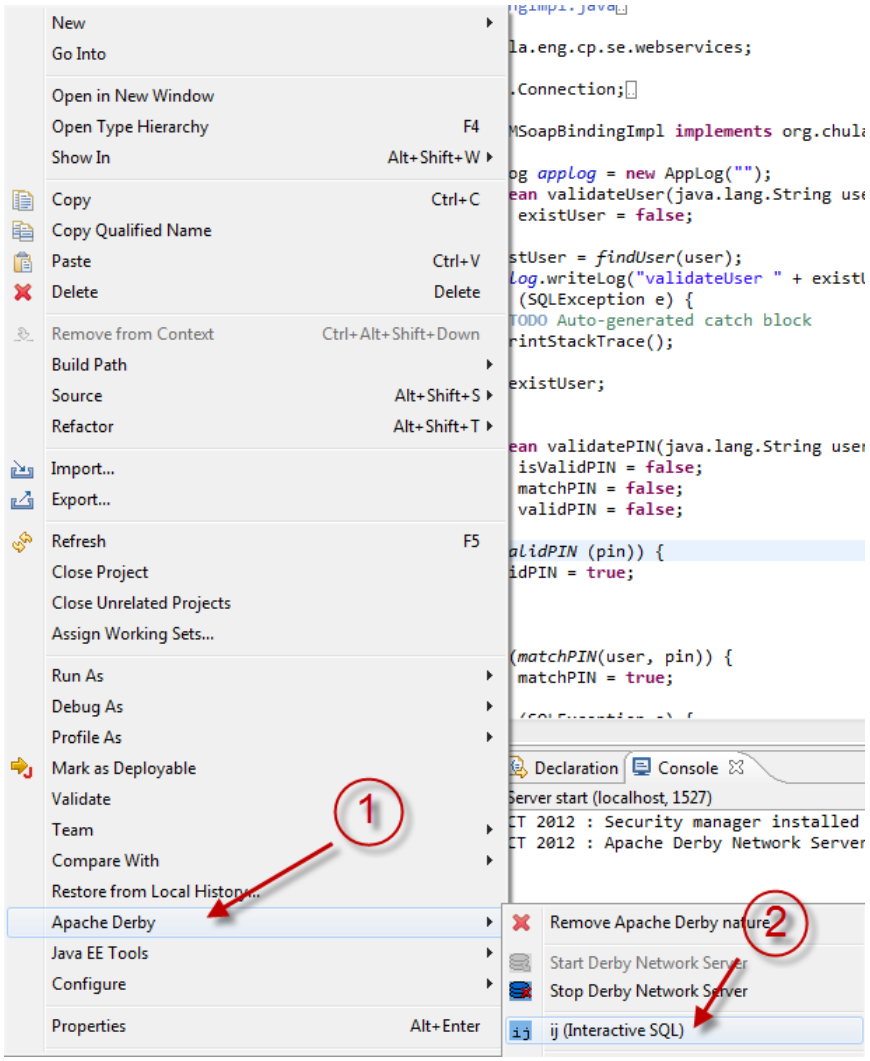
4) หลังจากเพิ่มความสามารถของ Apache Derby แล้ว เมื่อคลิกขวาอีกครั้งจะมีคำสั่ง Start Derby Network Server ดังภาพที่ ข.34 ให้เลือกคำสั่งนี้เพื่อเปิดใช้ฐานข้อมูล Apache Derby ดังภาพที่ ข.35 จากนั้นเลือกคำสั่ง ij (Interactive SQL) เพื่อป้อนคำสั่ง SQL ในการสร้างตารางและทำการใส่ข้อมูล ดังภาพที่ ข.36



ภาพที่ ข.34 เปิดใช้งานคำสั่ง Start Derby Network Server

```
Wed Feb 01 23:42:33 ICT 2012 : Security manager installed using the Basic server security policy.
Wed Feb 01 23:42:36 ICT 2012 : Apache Derby Network Server - 10.8.2.2 - (1181258) started and ready to accept connections on port 1527
```

ภาพที่ ข.35 ข้อความทางคอนโซลของโปรแกรมอีคลิป

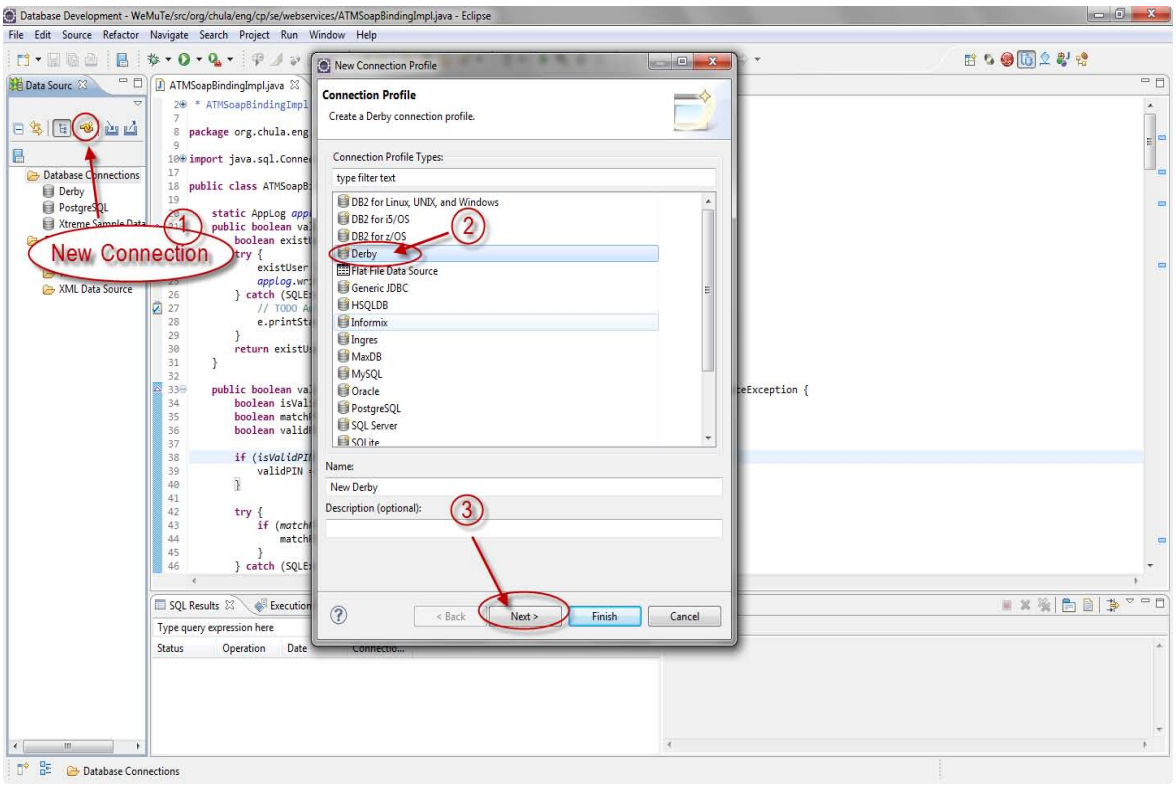


ภาพที่ ข.36 เปิดคำสั่งใช้งาน ij (Interacetive SQL)

5) ถ้ายังไม่มีการติดตั้งใช้งานฐานข้อมูลจำเป็นต้องติดตั้งก่อน โดยที่ส่วนเมนูของโปรแกรมอีคลิปเลือกที่สัญลักษณ์ Database Development ดังภาพที่ ข.37

ภาพที่ ข.37 สัญลักษณ์ Database Development

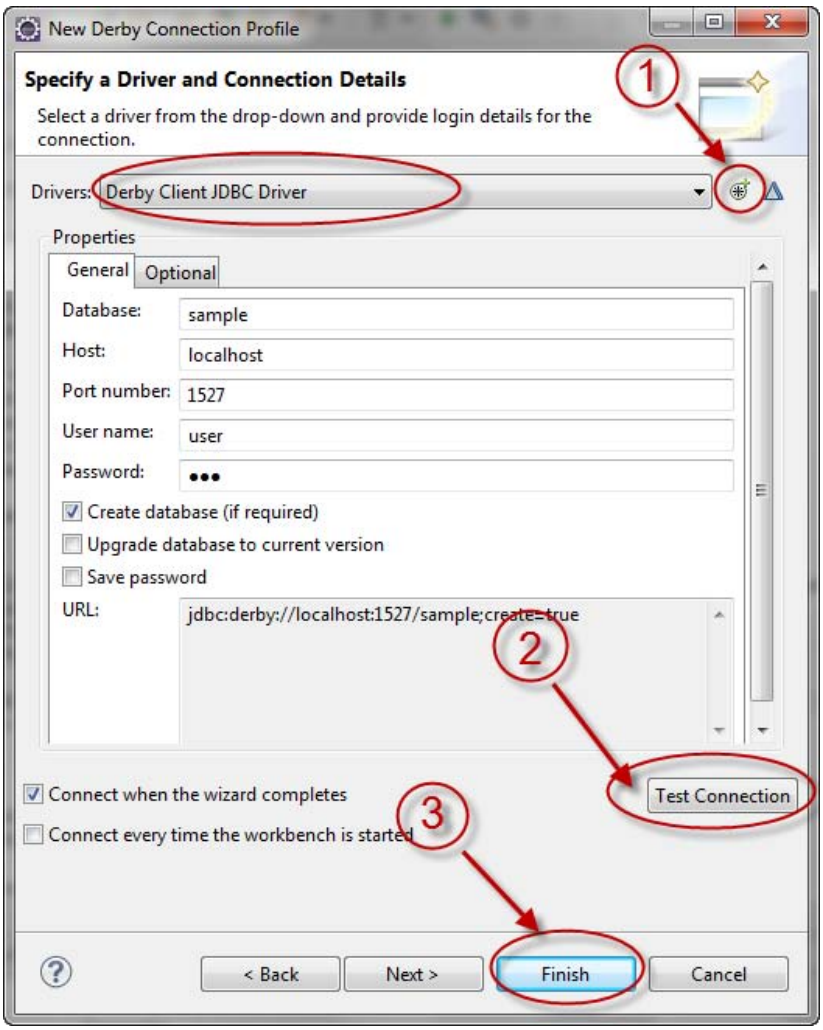
6) ในส่วนทางด้านซ้ายที่แท็ป Data Source ให้เลือกสัญลักษณ์  เพื่อ New Connection จากนั้นจะปรากฏหน้าต่าง New Connection Profile ขึ้นมาดังภาพที่ ข.38 ให้เลือก Derby แล้วเลือก Next >



ภาพที่ ข.38 หน้าต่าง New Database Connection Profile

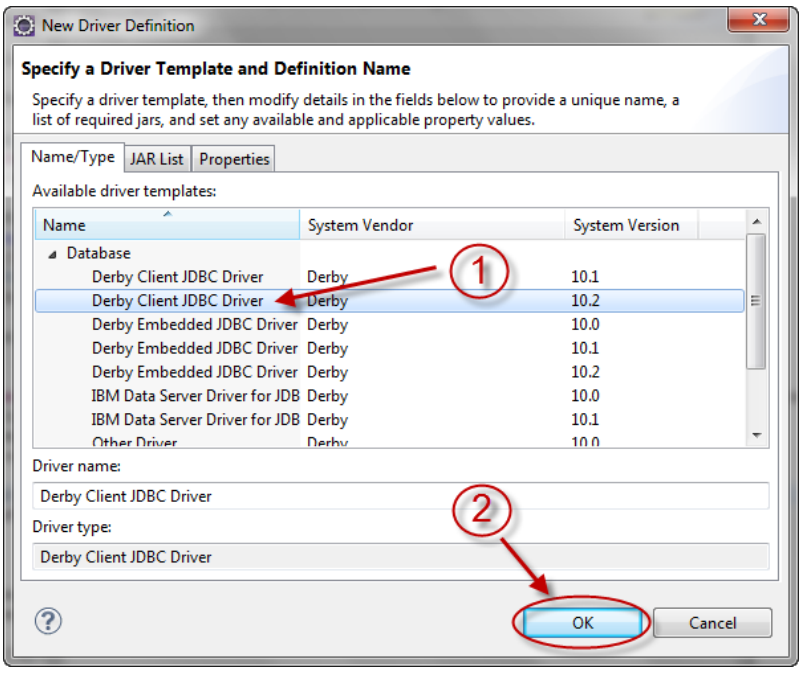
7) จากนั้นจะปรากฏหน้าต่าง Specify a Driver and Connection Details ดังภาพที่ ข.39 ซึ่งค่าข้อมูลต่างๆ โปรแกรมได้ใส่ค่ามาให้เรียบร้อยแล้ว ในส่วนของ password โปรแกรมจะบังคับให้ใส่แต่ไม่ได้ตรวจสอบ เราสามารถทดสอบการเชื่อมต่อกับฐานข้อมูลโดยการ Test Connection ถ้าเราเชื่อมต่อกับฐานข้อมูลแล้วจะขึ้นหน้าต่าง ดังภาพที่ ข.40 จากนั้นให้เลือกที่ Finish

7.1) ในขั้นตอนนี้จากภาพที่ ข.35 ในส่วนของ Drivers อาจจะไม่มี Derby Client JDBC Driver เราต้องเลือกที่สัญลักษณ์ จะปรากฏหน้างต่าง Specify a Driver Template and Definition Name ดังภาพที่ ข.41 ให้เลือกที่ Derby Client JDBC Driver แล้วเลือก OK เพื่อตกลง



ภาพที่ ข.39 หน้าต่าง Specify a Driver and Connection Details

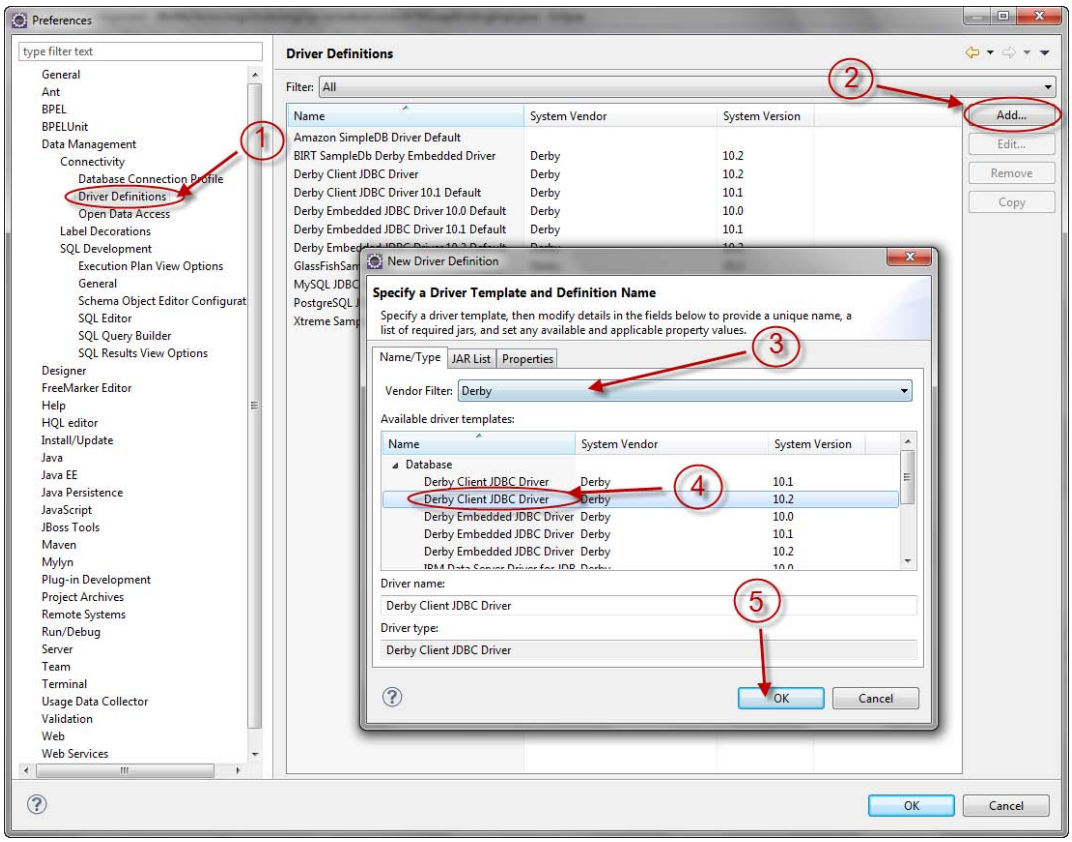
ภาพที่ ข.40 ทดสอบการเชื่อมต่อกับฐานข้อมูล



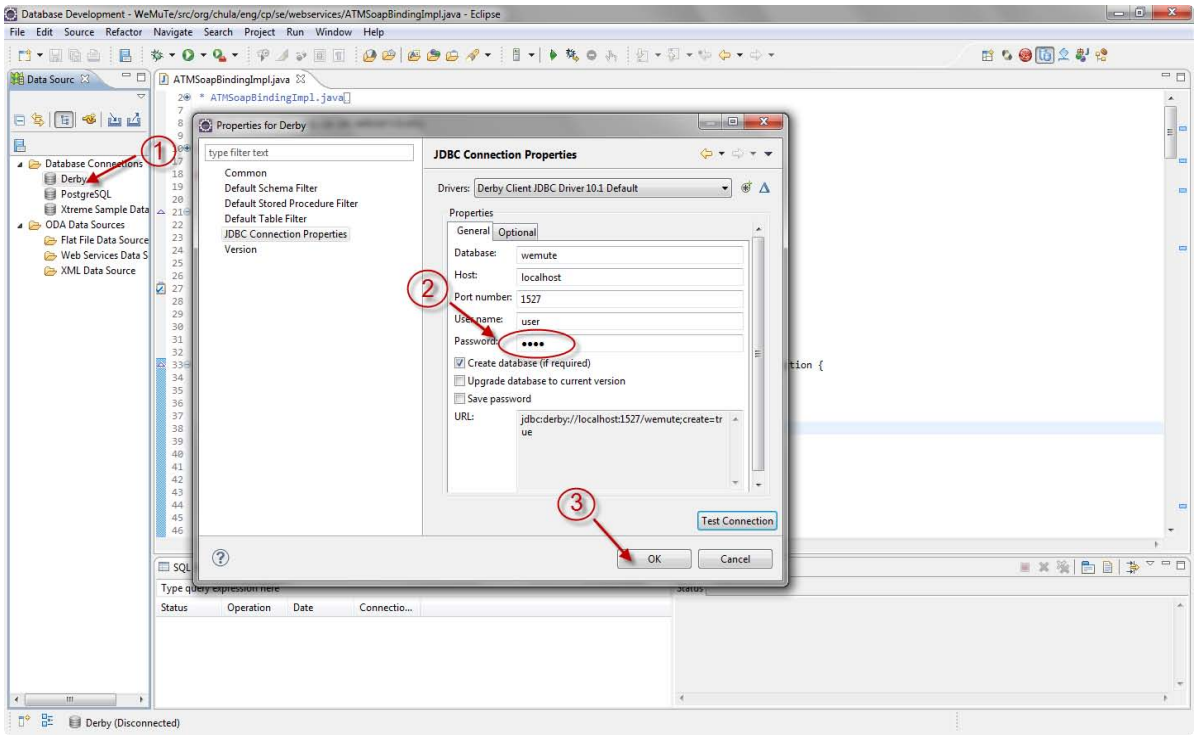
ภาพที่ ข.41 หน้าต่าง New Driver Definition

7.2) หากไม่พบ Derby Client JDBC Driver ในส่วนของ Specify a Driver Template and Definition Name เราจำเป็นต้องไป New Driver Database โดยไปที่เมนู Window -> Preferences -> Data Management -> Connectivity -> Driver Definitions เลือกที่เพิ่ม Add… จะปรากฏหน้าต่าง New Driver Definition ดังภาพที่ ข.42 และเราสามารถกรองชื่อได้ ให้เลือก Derby Client JDBC Driver แล้วตกลง

8) เมื่อเราสร้างการเชื่อมฐานข้อมูลแล้ว ต่อไปเราจะทำการเชื่อต่อกับฐานข้อมูลโดย จากรูป ข.43 คลิกขวาที่ฐานข้อมูล Derby แล้วเลือก Connect จะปรากฏหน้าต่าง Properties for Derby จากนั้นเลือกตกลงเพื่อเชื่อมต่อ

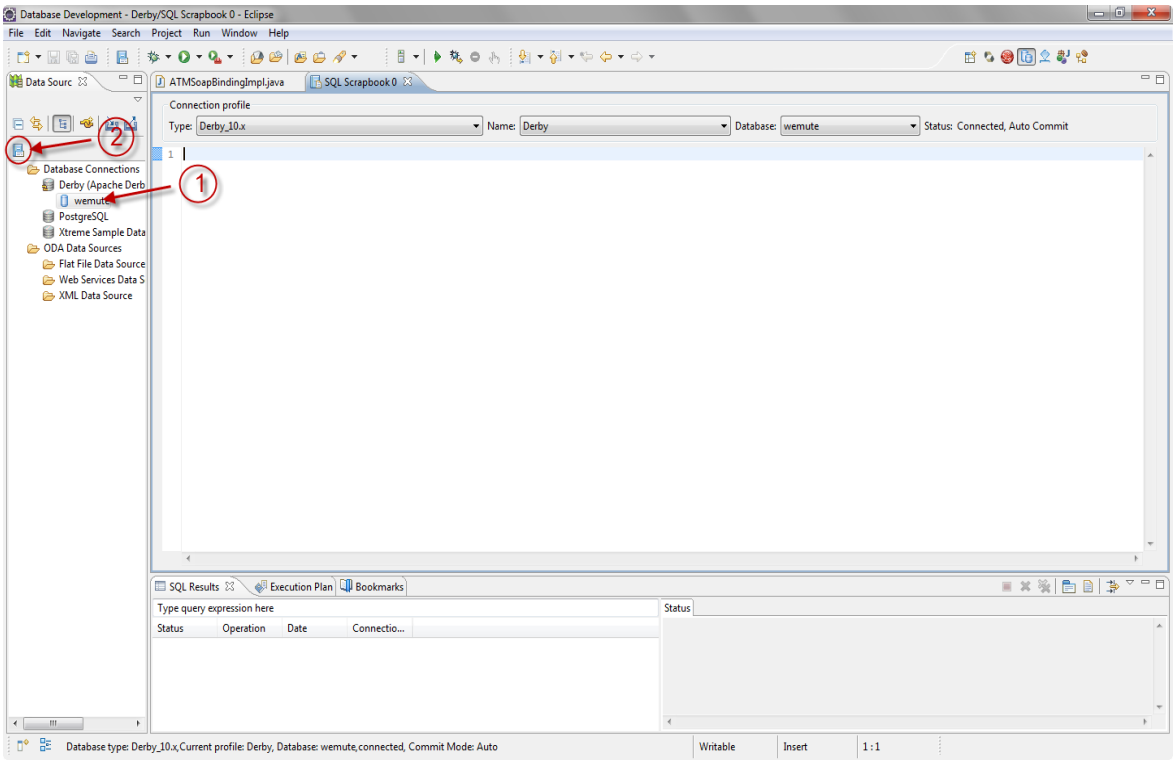


ภาพที่ ข.42 หน้าต่าง New Driver Definitions จาก Preferences



ภาพที่ ข.43 หน้าต่าง Database Connection

9) เมื่อเชื่อมต่อสำเร็จจะปรากฏโปรแกรมจะแสดงดังภาพที่ ข.44 เลือกฐานข้อมูลที่เราต้องการ จากนั้นเลือกที่สัญลักษณ์  เพื่อเตรียมพร้อมในการป้อนคำสั่ง



ภาพที่ ข.44 หน้าต่าง SQL Scrapbook

10) จากตารางที่ ข.1 – ข.13 เป็นสคีมาของตารางในฐานข้อมูลที่เครื่องมือทดสอบวีคมิวเทชันสำหรับดับเบิลยูเอสบีเพลใช้ในการตรวจสอบข้อมูล กับเว็บเซอร์วิซซึ่งจะมีตารางเอทีเอ็ม สินเชื่อ สินค้า สายการบิน โรงแรม ส่วนหัวคลังสินค้า ส่วนท้ายคลังสินค้า ส่วนหัวขนส่งสินค้า ส่วนท้ายขนส่งสินค้า ส่วนหัวใบสั่งซื้อสินค้า ส่วนท้ายใบสั่งซื้อสินค้า อะไหล่ และผู้ขาย รวมทั้งคำสั่ง SQL ในการสร้างตารางดังภาพที่ ข.45 ข.47 ข.49 ข.51 ข.53 ข.55 ข.57 ข.59 ข.61 ข.63 ข.65 ข.67 ข.69 และคำสั่ง SQL ในการใส่ข้อมูลในตารางดังภาพที่ ข.46 ข.48 ข.50 ข.52 ข.54 ข.56 ข.58 ข.60 ข.62 ข.64 ข.66 ข.68 และ ข.70

ตารางที่ ข.1 สคีมาฐานข้อมูลของตารางเอทีเอ็ม (ATM)

ตารางเอทีเอ็ม			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Name	Varchar	-	20

ตารางที่ ข.1 สคีมาฐานข้อมูลของตารางเอทีเอ็ม (ATM) (ต่อ)

PIN	Varchar	-	4
BalanceAmount	Double	-	-

```
CREATE TABLE APP.ATM (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS
   IDENTITY
  (START WITH 1, INCREMENT BY 1),
  Name VARCHAR(20),
  PIN VARCHAR(4) NOT NULL
  BalanceAmount DOUBLE
 );
```

ภาพที่ ข.45 คำสั่งสร้างตารางเอทีเอ็ม

```
INSERT INTO APP.ATM(Name, PIN, BalanceAmount)
VALUES('Peter','1112', 500000);

INSERT INTO APP.ATM(Name, PIN, BalanceAmount)
VALUES('Steve','6758', 2400000);

INSERT INTO APP.ATM(Name, PIN, BalanceAmount)
VALUES('iPatz','5432', 55000);

INSERT INTO APP.ATM(Name, PIN, BalanceAmount)
VALUES('Gates','4578', 3500000);
```

ภาพที่ ข.46 คำสั่งใส่ข้อมูลในตารางเอทีเอ็ม

ตารางที่ ข.2 สคีมาฐานข้อมูลของตารางสินเชื่อ (Loan)

ตารางสินเชื่อ			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Code	Varchar	-	5
Name	Varchar	-	10
Age	Integer	-	-
Credit	Double	-	-
IsBlackList	Boolean	-	-
HasExperience	Boolean	-	-
IsNPL	Boolean	-	-

```
CREATE TABLE APP.Loan (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  Code VARCHAR(5),
  Name VARCHAR(10),
  Age INTEGER,
  Credit DOUBLE PRECISION,
  IsBlackList BOOLEAN,
  HasExperience BOOLEAN,
  IsNPL BOOLEAN
);
```

ภาพที่ ข.47 คำสั่งสร้างตารางสินเชื่อ

```
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U001', 'Jame', 24, 500000, FALSE,TRUE, TRUE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U002', 'Jimmy', 19,45000, TRUE, TRUE, TRUE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U003', 'Kathy', 36, 1500000, FALSE, TRUE, FALSE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U004', 'Lilly', 10, 3500, FALSE, FALSE, FALSE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U005', 'Jeorge', 46, 3567400, TRUE, TRUE, FALSE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U006', 'Hotz', 15, 130000, FALSE, FALSE, FALSE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U007', 'Jay', 50, 43500, FALSE, FALSE, TRUE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U008', 'Patrick', 4, 1000, TRUE, FALSE, TRUE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U009', 'John', 39, 79800, TRUE, TRUE, TRUE);
INSERT INTO APP.Loan(Code, Name, Age, Credit, IsBlackList, HasExperience,
   IsNPL) VALUES('U010', 'Albert', 7, 1300, TRUE, FALSE, FALSE);
```

ภาพที่ ข.48 คำสั่งใส่ข้อมูลในตารางสินเชื่อ

ตารางที่ ข.3 สคีมาฐานข้อมูลของตารางสินค้า (Product)

ตารางสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Code	Varchar	-	5
Name	Varchar	-	100
Quantity	Double	-	-
Amount	Double	-	-

```
CREATE TABLE APP.Product (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
  (START WITH 1, INCREMENT BY 1),
  Code VARCHAR(5),
  Name VARCHAR(100) NOT NULL,
  Quantity DOUBLE PRECISION,
  Amount DOUBLE PRECISION
);
```

ภาพที่ ข.49 คำสั่งสร้างตารางสินค้า

```
INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC001', 'Macbook Air 13"', 6, 38000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC002', 'iPad2 Black 3G Wifi 64Gb', 2, 25000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC003', 'iPhone4 White 32 Gb', 7, 15000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC004', 'iPhone4 Black 16 Gb', 11, 14500);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC005', 'Mackbook Pro 17"', 5, 70000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC006', 'iPod Nano 16 Gb', 10, 10000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC007', 'iMac 27"', 1, 99000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC008', 'Apple TV 2nd Gen', 7, 8900);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC009', 'iPod Touch 32 Gb', 10, 7500);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC010', 'iPhone3GS Black 16 Gb', 5, 15000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC011', 'iPad Black Wifi 32 Gb', 12, 17000);

INSERT INTO APP.Product(Code, Name, Quantity, Amount)
VALUES('PC012', 'iPhone4S White 64 Gb', 0, 26700);
```

ภาพที่ ข.50 คำสั่งใส่ข้อมูลในตารางสินค้า

ตารางที่ ข.4 สคีมาฐานข้อมูลของตารางสายการบิน (Airline)

ตารางสายการบิน			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Code	Varchar	-	6
Depart	Double	-	-
Arrive	Double	-	-
Destination	Varchar	-	10
IsAvailable	Boolean	-	-
Amount	Double	-	-

```
CREATE TABLE APP.Airline (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  Code VARCHAR(6),
  Depart DOUBLE PRECISION,
  Arrive DOUBLE PRECISION,
  Destination VARCHAR(10),
  IsAvailable BOOLEAN,
  Amount DOUBLE PRECISION
);
```

ภาพที่ ข.51 คำสั่งสร้างตารางสายการบิน

```
INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('KR-608', 20, 22, 'Korea', TRUE, 13500);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('KR-401', 2.3, 4.3, 'Korea', FALSE, 10500);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('US-667', 4.3, 16.35, 'USA', TRUE, 78000);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('US-433', 10.4, 22.50, 'USA', FALSE, 81000);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('JP-102', 5.35, 9.45, 'Japan', FALSE, 25000);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('JP-798', 11.2, 15.3, 'Japan', TRUE, 27500);

INSERT INTO APP.Airline(Code, Depart, Arrive, Destination, isAvailable,
   Amount)
VALUES('UK-554', 12.1, 21.45, 'England', FALSE, 68900);
```

ภาพที่ ข.52 คำสั่งใส่ข้อมูลในตารางสายการบิน

ตารางที่ ข.5 สคีมาฐานข้อมูลของตารางโรงแรม (Hotel)

ตารางโรงแรม			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Place	Varchar	-	10
IsAvailable	Boolean	-	-
AmountPerDay	Double	-	-

```
CREATE TABLE APP.Hotel (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  Place VARCHAR(10),
  IsAvailable BOOLEAN,
  AmountPerDay DOUBLE PRECISION
);
```

ภาพที่ ข.53 คำสั่งสร้างตารางโรงแรม

```
INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Korea', TRUE, 2500);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Korea', TRUE, 4500);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('USA', FALSE, 15000);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('USA', TRUE, 6500);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Japan', TRUE, 3500);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Japan', FALSE, 3000);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('England', FALSE, 7600);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Plague', FALSE, 7600);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('Korea', FALSE, 3000);

INSERT INTO APP.Hotel(Place, IsAvailable, AmountPerDay)
VALUES('USA', TRUE, 5500);
```

ภาพที่ ข.54 คำสั่งใส่ข้อมูลในตารางโรงแรม

ตารางที่ ข.6 สคีมาฐานข้อมูลของตารางส่วนหัวคลังสินค้า (InventoryHead)

ตารางส่วนหัวคลังสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
PoNo	Varchar	-	20

ตารางที่ ข.6 สคีมาฐานข้อมูลของตารางส่วนหัวคลังสินค้า (InventoryHead) (ต่อ)

StockMessage	Varchar	-	50
Status	Varchar	-	10
StockDate	Date	-	-
StockCode	Varchar	-	20

```
CREATE TABLE APP.InventoryHead (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  PoNo VARCHAR(20),
  StockMessage VARCHAR(50),
  Status VARCHAR(10),
  StockDate DATE,
  StockCode VARCHAR(20)
);
```

ภาพที่ ข.55 คำสั่งสร้างตารางส่วนหัวคลังสินค้า

```
INSERT INTO APP.InventoryHead(PoNo, StockMessage, Status, StockDate,
   StockCode)
VALUES('PO201202010001', 'In Stock', 'StockIn','2012-02-01','ST0001');
INSERT INTO APP.InventoryHead(PoNo, StockMessage, Status, StockDate,
   StockCode)
VALUES('PO201205050002', 'In Stock', 'StockIn','2012-05-05','ST0002');
INSERT INTO APP.InventoryHead(PoNo, StockMessage, Status, StockDate,
   StockCode)
VALUES('PO201203260003', 'Out of Stock', 'StockOut','2012-03-
   26','ST0003');
INSERT INTO APP.InventoryHead(PoNo, StockMessage, Status, StockDate,
   StockCode)
VALUES('PO201204170004', 'Out of Stock', 'StockOut','2012-04-
   17','ST0004');
INSERT INTO APP.InventoryHead(PoNo, StockMessage, Status, StockDate,
   StockCode)
VALUES('PO201204290005', 'Going', 'OnGoing','2012-04-29','ST0005');
```

ภาพที่ ข.56 คำสั่งใส่ข้อมูลในตารางส่วนหัวคลังสินค้า

ตารางที่ ข.7 สคีมาฐานข้อมูลของตารางส่วนท้ายคลังสินค้า (InventoryLine)

ตารางส่วนท้ายคลังสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
ProductCode	Varchar	-	10

ตารางที่ ข.7 สคีมาฐานข้อมูลของตารางส่วนท้ายคลังสินค้า (InventoryLine) (ต่อ)

Quantity	Double	-	-
Price	Double	-	-
Amount	Double	-	-
LocationName	Varchar	-	10

```
CREATE TABLE APP.InventoryLine (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  ProductCode VARCHAR(10),
  Quantity DOUBLE PRECISION,
  Price DOUBLE PRECISION,
  Amount DOUBLE PRECISION,
  LocationName VARCHAR(10)
);
```

ภาพที่ ข.57 คำสั่งสร้างตารางส่วนท้ายคลังสินค้า

```
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0010', 100, 200, 20000, 'A01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0009', 60, 250, 15000, 'A01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0011', 55, 170, 9350, 'B01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0002', 80, 4000, 320000, 'B01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0003', 60, 900, 54000, 'B01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0004', 250, 850, 212500, 'B01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0005', 450, 200, 90000, 'C01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0006', 440, 250, 110000, 'C01');
INSERT INTO APP.InventoryLine(ProductCode, Quantity, Price, Amount,
   LocationName)
VALUES('PT0009', 700, 150, 105000, 'C01');
```

ภาพที่ ข.58 คำสั่งใส่ข้อมูลในตารางส่วนท้ายคลังสินค้า

ตารางที่ ข.8 สคีมาฐานข้อมูลของตารางส่วนหัวขนส่งสินค้า (LogisticsHead)

ตารางส่วนหัวขนส่งสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
Code	Varchar	-	10
CompanyName	Varchar	-	30
StartDate	Date	-	-
DueDate	Date	-	-
Duration	Integer	-	-
TrackingMessage	Varchar	-	50
ShipBy	Varchar	-	20
PoNo	Varchar	-	20
TrackingStatus	Varchar	-	20

```
CREATE TABLE APP.LogisticsHead (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  Code VARCHAR(10),
  CompanyName VARCHAR(30),
  StartDate DATE,
  DueDate DATE,
  Duration INTEGER,
  TrackingMessage VARCHAR(50),
  ShipBy VARCHAR(20),
  PoNo VARCHAR(20),
  TrackingStatus VARCHAR(20)
);
```

ภาพที่ ข.59 คำสั่งสร้างตารางส่วนหัวขนส่งสินค้า

```
INSERT INTO APP.LogisticsHead(Code, CompanyName, StartDate, DueDate,
   Duration, TrackingMessage, ShipBy, PoNo, TrackingStatus)
VALUES('LG0001', 'Narin Motor', '2012-02-01', '2012-02-03', 3, '-',
   'Civa', 'PO201202010001', 'OnTransport');
INSERT INTO APP.LogisticsHead(Code, CompanyName, StartDate, DueDate,
   Duration, TrackingMessage, ShipBy, PoNo, TrackingStatus)
VALUES('LG0002', 'Narin Motor', '2012-05-05', '2012-05-08', 4, '-',
   'Civa', 'PO201205050002', 'OnWarehouse');
INSERT INTO APP.LogisticsHead(Code, CompanyName, StartDate, DueDate,
   Duration, TrackingMessage, ShipBy, PoNo, TrackingStatus)
VALUES('LG0003', 'Ruam Charoen Arai Co,.Ltd.', '2012-03-26', '2012-03-
   30', 5, '-', 'Civa', 'PO201203260003', 'OnWarehouse');
INSERT INTO APP.LogisticsHead(Code, CompanyName, StartDate, DueDate,
   Duration, TrackingMessage, ShipBy, PoNo, TrackingStatus)
VALUES('LG0004', 'Lee Mong Huad Tire Co.,Ltd.', '2012-04-17', '2012-
   04-20', 4, '-', 'Civa', 'PO201204170004', 'OnLeaving');
INSERT INTO APP.LogisticsHead(Code, CompanyName, StartDate, DueDate,
   Duration, TrackingMessage, ShipBy, PoNo, TrackingStatus)
VALUES('LG0005', 'Ruam Charoen Arai Co,.Ltd.', '2012-04-29', '2012-04-
   30', 2, '-', 'Civa', 'PO201204290005', 'OnLeaving');
```

ภาพที่ ข.60 คำสั่งใส่ข้อมูลในตารางส่วนหัวขนส่งสินค้า

ตารางที่ ข.9 สคีมาฐานข้อมูลของตารางส่วนท้ายขนส่งสินค้า (LogisticsLine)

ตารางส่วนท้ายขนส่งสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
EmsCode	Varchar	-	20
ProductCode	Varchar	-	10
ContainerNo	Varchar	-	10
Remark	Varchar	-	100

```
CREATE TABLE APP.LogisticsLine (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  EmsCode VARCHAR(20),
  ProductCode VARCHAR(10),
  ContainerNo VARCHAR(10),
  Remark VARCHAR(100)
);
```

ภาพที่ ข.61คำสั่งสร้างตารางส่วนท้ายขนส่งสินค้า

```
INSERT INTO APP.LogisticsLine(EmsCode, ProductCode, ContainerNo,
   Remark)
VALUES('EA235567801TH', 'PT0010', 'CN00230', '-');
INSERT INTO APP.LogisticsLine(EmsCode, ProductCode, ContainerNo,
   Remark)
VALUES('EA235567801TH', 'PT0009', 'CN00230', '-');
INSERT INTO APP.LogisticsLine(EmsCode, ProductCode, ContainerNo,
   Remark)
VALUES('EA114567909TH', 'PT0011', 'CN04550', '-');
INSERT INTO APP.LogisticsLine(EmsCode, ProductCode, ContainerNo,
   Remark)
VALUES('EA114567909TH', 'PT0015', 'CN04550', '-');
INSERT INTO APP.LogisticsLine(EmsCode, ProductCode, ContainerNo,
   Remark)
VALUES('EA114567909TH', 'PT0016', 'CN04550', '-');
```

ภาพที่ ข.62 คำสั่งใส่ข้อมูลในตารางส่วนท้ายขนส่งสินค้า

ตารางที่ ข.10 สคีมาฐานข้อมูลของตารางส่วนหัวใบสั่งซื้อสินค้า (POHead)

ตารางส่วนหัวใบสั่งซื้อสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
PoNo	Varchar	-	20
PoDate	Date	-	-
PoType	Varchar	-	10
PoStatus	Varchar	-	15
SumQty	Double	-	-
SumAmt	Double	-	-
SumDiscount	Double	-	-
Remark	Varchar	-	50

```
CREATE TABLE APP.POHead (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  PoNo VARCHAR(20),
  PoDate DATE,
  PoType DOUBLE PRECISION
  PoStatus VARCHAR(15),
  SumQty DOUBLE PRECISION,
  SumAmt DOUBLE PRECISION
  SumDiscount DOUBLE PRECISION,
  Remark VARCHAR(50)
);
```

ภาพที่ ข.63 คำสั่งสร้างตารางส่วนหัวใบสั่งซื้อสินค้า

```
INSERT INTO APP.POHead(PoNo, PoDate, PoType, PoStatus, SumQty, SumAmt,
   SumDiscount, Remark)
VALUES('PO201202010001', '2012-02-01','PI', 'Invoice', 10, 26000,
   1400, '-');
INSERT INTO APP.POHead(PoNo, PoDate, PoType, PoStatus, SumQty, SumAmt,
   SumDiscount, Remark)
VALUES('PO201205050002', '2012-05-05','PR', 'Draft', 20, 40000, 1000,
   '-');
INSERT INTO APP.POHead(PoNo, PoDate, PoType, PoStatus, SumQty, SumAmt,
   SumDiscount, Remark)
VALUES('PO201203260003', '2012-03-26','PO', 'Order', 10, 450000, 1220,
   '-');
INSERT INTO APP.POHead(PoNo, PoDate, PoType, PoStatus, SumQty, SumAmt,
   SumDiscount, Remark)
VALUES('PO201204170004', '2012-04-17','PR', 'Draft', 100, 45000, 3450,
   '-');
INSERT INTO APP.POHead(PoNo, PoDate, PoType, PoStatus, SumQty, SumAmt,
   SumDiscount, Remark)
VALUES('PO201204290005', '2012-04-29','PR', 'Draft', 230, 250000,
   7680, '-');
```

ภาพที่ ข.64 คำสั่งใส่ข้อมูลในตารางส่วนหัวใบสั่งซื้อสินค้า

ตารางที่ ข.11 สคีมาฐานข้อมูลของตารางส่วนท้ายใบสั่งซื้อสินค้า (POLine)

ตารางส่วนท้ายใบสั่งซื้อสินค้า			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
ProductCode	Varchar	-	20
ProductName	Varchar	-	50
Price	Double	-	-
Qty	Double	-	-

ตารางที่ ข.11 สคีมาฐานข้อมูลของตารางส่วนท้ายใบสั่งซื้อสินค้า (POLine) (ต่อ)

Amount	Double	-	-
Discount	Double	-	-
PoNo	Varchar	-	20

```
CREATE TABLE APP.POLine (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  ProductCode VARCHAR(20),
  ProductName VARCHAR(50),
  Price DOUBLE PRECISION,
  Qty DOUBLE PRECISION,
  Amount DOUBLE PRECISION,
  Discount DOUBLE PRECISION,
  PoNo VARCHAR(20)
);
```

ภาพที่ ข.65 คำสั่งสร้างตารางส่วนท้ายใบสั่งซื้อสินค้า

```
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0010', 'Front Sprocket Jomthai Dream 14T', 55, 200, 11000, 0,
   'PO201202010001');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0009', 'Rear Sprocket Jomthai Wave 35T', 60, 250, 15000, 0,
   'PO201202010001');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0011', 'Chain Jomthai 420 120L', 55, 175, 9625, 0,
   'PO201202010001');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0002', 'V-Belt 82 B 3 stars', 75, 4000, 300000, 0,
   'PO201205050002');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0003', 'Inner Tube Veerubber 2.50-17', 58, 900, 52200, 0,
   'PO201203260003');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0004', 'Outer Tube Camel 2.25-17 CM2', 220, 850, 187000, 0,
   'PO201203260003');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0005', 'Front-Hub Wave 110i', 450, 200, 90000, 0,
   'PO201204170004');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0006', 'Rear-Hub Nova-Tena', 440, 250, 110000, 0,
   'PO201204170004');
INSERT INTO APP.POLine(ProductCode, ProductName, Price, Qty, Amount,
   Discount, PoNo)
VALUES('PT0009', 'Carbulator Wave 125', 610, 150, 91500, 0,
   'PO201204290005');
```

ภาพที่ ข.66 คำสั่งใส่ข้อมูลในตารางส่วนท้ายใบสั่งซื้อสินค้า

ตารางที่ ข.12 สคีมาฐานข้อมูลของตารางอะไหล่ (Parts)

ตารางอะไหล่			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
PartsCode	Varchar	-	10
PartsNameEn	Varchar	-	50
CostPrice	Double	-	-
WholesalePrice	Double	-	-

ตารางที่ ข.12 สคีมาฐานข้อมูลของตารางอะไหล่ (Parts) (ต่อ)

RetailPrice	Double	-	-
Quantity	Double	-	-
SupplierCode	Varchar	-	10
SupplierNameEn	Varchar	-	30
Description	Varchar	-	100

```
CREATE TABLE APP.Parts (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  PartsCode VARCHAR(10),
  PartsName VARCHAR(50),
  CostPrice DOUBLE PRECISION,
  WholesalePrice DOUBLE PRECISION,
  RetailPrice DOUBLE PRECISION,
  Quantity DOUBLE PRECISION,
  SupplierCode VARCHAR(10),
  SupplierNameEn VARCHAR(30),
  Description VARCHAR(100)
);
```

ภาพที่ ข.67 คำสั่งสร้างตารางอะไหล่

```
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0001', 'Carbulator Wave 125', 550, 610, 680, 980, 'SP0001',
   'Ruam Charoen Arai Co,.Ltd.', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0002', 'V-Belt 82 B 3 stars', 70, 80, 85, 3500, 'SP0002',
   'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0003', 'Inner Tube Veerubber 2.50-17', 55, 58, 65, 1400,
   'SP0003', 'Lee Mong Huad Tire Co.,Ltd', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0004', 'Outer Tube Camel 2.25-17 CM2', 210, 220, 240, 1700,
   'SP0003', 'Lee Mong Huad Tire Co.,Ltd', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0005', 'Front-Hub Wave 110i', 430, 450, 520, 670, 'SP0002',
   'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0006', 'Rear-Hub Nova-Tena', 420, 440, 500, 450, 'SP0002',
   'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0007', 'Complete Set Gasket Sonic', 80, 85, 90, 250, 'SP0002',
   'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0008', 'Piston Kit Set Dream 0.75', 125, 135, 145, 2500,
   'SP0002', 'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0009', 'Rear Sprocket Jomthai Wave 35T', 110, 130, 145, 2500,
   'SP0002', 'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0010', 'Front Sprocket Jomthai Dream 14T', 50, 55, 65, 1500,
   'SP0002', 'Narin Motor', '-');
INSERT INTO APP.Parts(PartsCode, PartsName, CostPrice, WholesalePrice,
   RetailPrice, Quantity, SupplierCode, SupplierNameEn, Description)
VALUES('PT0011', 'Chain Jomthai 420 120L', 140, 175, 185, 2000, 'SP0002',
   'Narin Motor', '-');
```

ภาพที่ ข.68 คำสั่งใส่ข้อมูลในตารางอะไหล่

ตารางที่ ข.13 สคีมาฐานข้อมูลของตารางผู้ขาย (Supplier)

ตารางผู้ขาย			
ชื่อ	**ชนิดข้อมูล**	**คีย์**	**ขนาด**
Id	Integer	PK	-
SupplierCode	Varchar	-	10
SupplierName	Varchar	-	50
SupplierAddress	Varchar	-	100
TelNo	Varchar	-	15
MobileNo	Varchar	-	15
Email	Varchar	-	30
Remark	Varchar	-	100

```
CREATE TABLE APP.Supplier (
  Id INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY
   (START WITH 1, INCREMENT BY 1),
  SupplierCode VARCHAR(10),
  SupplierName VARCHAR(50),
  SupplierAddress VARCHAR(100),
  TelNo VARCHAR(15),
  MobileNo VARCHAR(15),
  Email VARCHAR(30),
  Remark VARCHAR(100)
);
```

ภาพที่ ข.69 คำสั่งสร้างตารางผู้ขาย

```
INSERT INTO APP.Supplier(SupplierCode, SupplierName, SupplierAddress,
   TelNo, MobileNo, Email, Remark)
VALUES('SP0001', 'Ruam Charoen Arai Co,.Ltd.', '11-13 Soi.Wongsawang15
   Bangsue Bangkok 10800');
INSERT INTO APP.Supplier(SupplierCode, SupplierName, SupplierAddress,
   TelNo, MobileNo, Email, Remark)
VALUES('SP0002', 'Narin Motor', '999 Uechuleang Building Bangruk
   Bangkok 10300','025545657', '0891235469', 'narin.motor@gmail.com',
   '-');
INSERT INTO APP.Supplier(SupplierCode, SupplierName, SupplierAddress,
   TelNo, MobileNo, Email, Remark)
VALUES('SP0003', 'Lee Mong Huad Tire Co.,Ltd.', 2500);
```

ภาพที่ ข.70 คำสั่งใส่ข้อมูลในตารางผู้ขาย

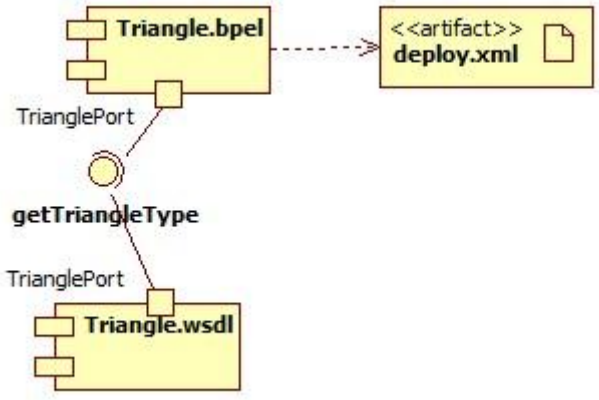
ภาคผนวก ค
การสร้างเว็บเซอร์วิซ

บทนี้อธิบายถึงเว็บเซอร์วิซที่ถูกนำมาใช้ในงานวิจัยนี้ ซึ่งจะอธิบายถึงคอมโพเนนท์ในกระบวนการบีเพล ข้อมูลและชนิดของข้อมูลที่ใช้ในการติดต่อระหว่างเว็บเซอร์วิซ และการสร้างเว็บเซอร์วิซ ตามลำดับ

ค.1 คอมโพเนนท์ของกระบวนการบีเพล

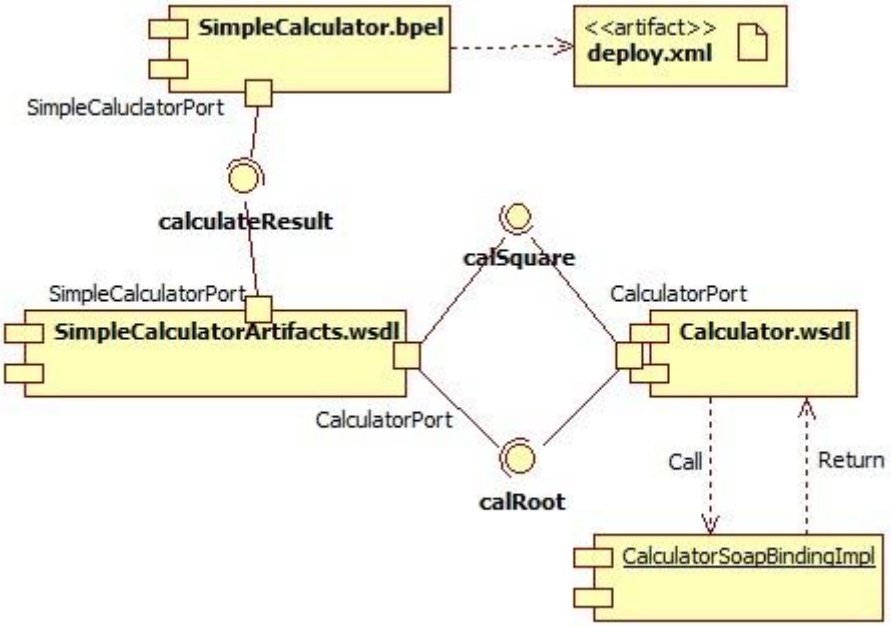
ในกระบวนการบีเพลประกอบด้วยหลายๆ เอกสารที่เกี่ยวข้องที่ต้องอาศัยการทำงานร่วมกัน ซึ่งมีกระบวนการบีเพลที่ใช้ทั้งหมด 9 โปรแกรมบีเพล คือ โปรแกรมหาชนิดของสามเหลี่ยม โปรแกรมเครื่องคิดเลข โปรแกรมการอนุมัติการกู้ยืม โปรแกรมเอทีเอ็ม โปรแกรมการซื้อสินค้า โปรแกรมการจองตั๋วท่องเที่ยว โปรแกรมเพิ่มค่าตัวเลข โปรแกรมหาค่ากำลังสอง และโปรแกรมใบสั่งซื้อสินค้า ดังภาพที่ ค.1 – ค.9

1) โปรแกรม Triangle ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 3 คอมโพเนนท์ คือ deploy.xml Triangle.bpel และ Triangle.wsdl โดย Triangle.bpel กับ Triangle.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง TrainglePort ด้วย getTriangleType ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร



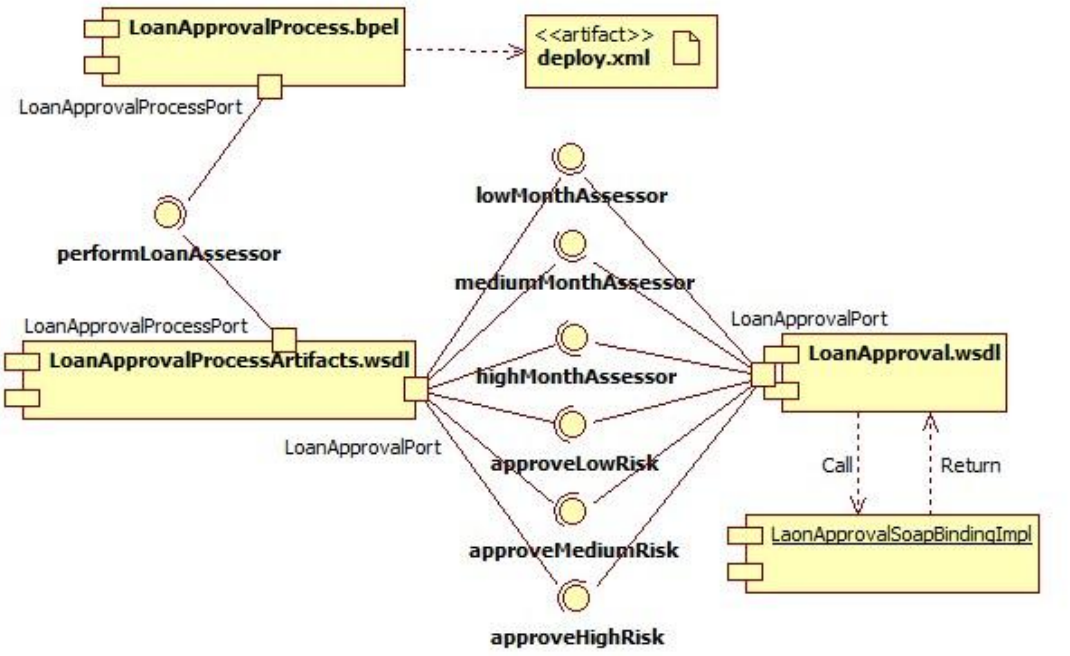
ภาพที่ ค.1 แผนภาพคอมโพเนนท์เว็บเซอร์วิซของโปรแกรม Triangle

2) โปรแกรม SimpleCalculator ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 5 คอมโพเนนท์ คือ deploy.xml SimpleCalculator.bpel SimpleCalculatorArtifacts.wsdl Calculator.wsdl และ CalculatorSoapBindingImpl โดย SimpleCalculator.bpel กับ SimpleCalculatorArtifacts.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง SimpleCalculatorPort ด้วย getTriangleType ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ SimpleCalculatorArtifacts.wsdl กับ Calculator.wsdl ยังติดต่อสื่อสารกันผ่านช่องทาง CalculatorPort โดยใช้ calSquare และ calRoot จากนั้น Calculator.wsdl จะไปเรียกใช้ CalculatorSoapBindingImpl ในการหาผลลัพธ์



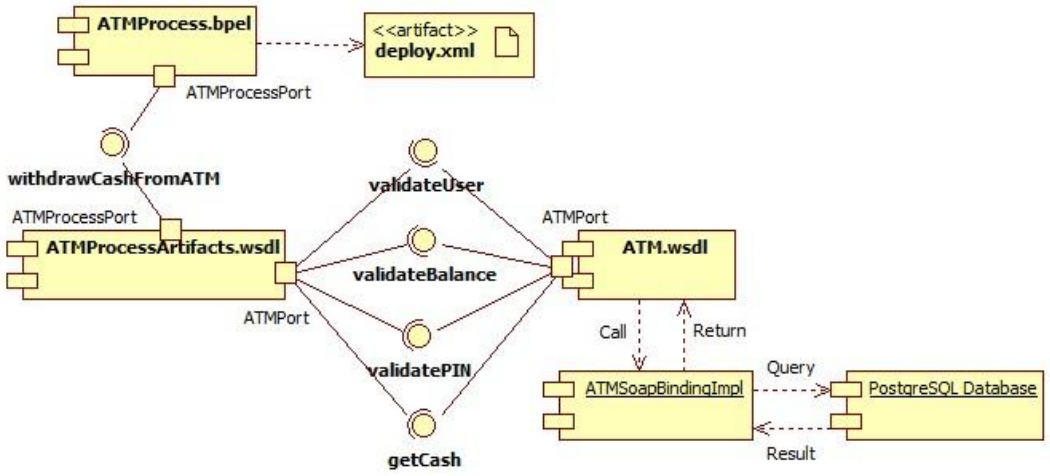
ภาพที่ ค.2 แผนภาพคอมโพเนนท์ของโปรแกรม SimpleCalculator

3) โปรแกรม LoanApprovalProcess ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 5 คอมโพเนนท์ คือ deploy.xml เอกสารบีเพล LoanApprovalProcess.bpel เอกสารวิซเดิล LoanApprovalProcessArtifacts.wsdl เอกสารวิซเดิล LoanApproval.wsdl และจาวาคลาส LoanApprovalSoapBindingImpl โดยกระบวนการ LoanApprovalProcess.bpel กับเว็บเซอร์วิซ LoanApprovalProcessArtifacts.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง LoanApprovalProcessPort โดยใช้ performLoanAssessor ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ LoanApprovalProcessArtifacts.wsdl กับ LoanApproval.wsdl ยังติดต่อสื่อสารกันผ่านช่องทาง LoanApprovalPort โดยใช้ lowMonthAssessor mediumMonthAssessor highMonthAssessor approveLowRisk approveMediumRisk และ approveHighRisk จากนั้น LoanApproval.wsdl จะไปเรียกใช้ LoanApprovalSoapBindingImpl ในการหาผลลัพธ์



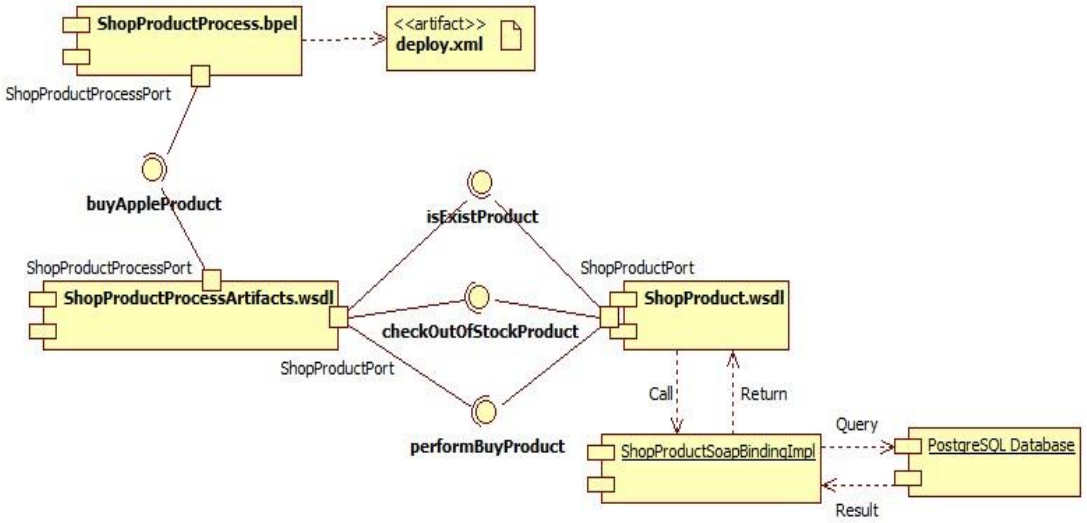
ภาพที่ ค.3 แผนภาพคอมโพเนนท์ของโปรแกรม LaonApprovalProcess

4) โปรแกรม ATMProcess ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 6 คอมโพเนนท์ คือ deploy.xml ATMProcess.bpel ATMProcessArtifacts.wsdl ATM.wsdl ATMSoapBindingImpl และ PostgreSQL Database โดย ATMProcess.bpel กับ ATMProcessArtifacts.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง ATMProcessPort ด้วย withdrawCashFromATM ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ ATMProcessArtifacts.wsdl กับ ATM.wsdl ยังติดต่อสื่อสารกันผ่านช่องทาง ATMPort โดยใช้ validateUser validateBalance validatePIN และ getCash จากนั้น ATM.wsdl จะไปเรียกใช้ ATMSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์



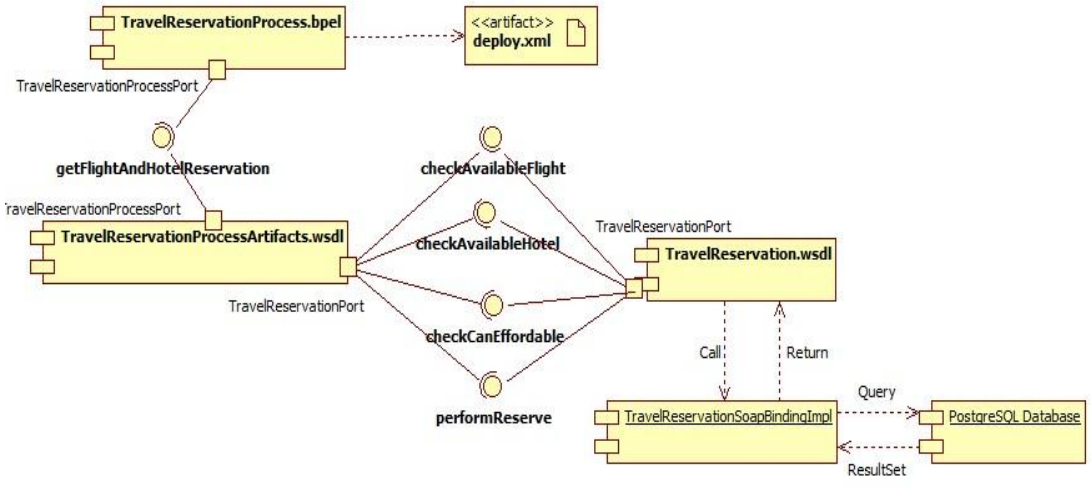
ภาพที่ ค.4 แผนภาพคอมโพเนนท์ของโปรแกรม ATMProcess

5) โปรแกรม ShopProcessProcess ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 6 คอมโพเนนท์ คือ deploy.xml เอกสารบีเพลShopProductProcess.bpel ShopProductProcessArtifacts.wsdl ShopProduct.wsdl ShopProductSoapBindingImpl และฐานข้อมูลซึ่งใช้ PostgreSQL Database โดยกระบวนการบีเพล ShopProductProcess.bpel กับเว็บเซอร์วิซ ShopProductProcessArtifacts.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง ShopProductProcessPort ด้วย buyAppleProduct ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ ShopProductProcessArtifacts.wsdl กับ ShopProduct.wsdl ยังติดต่อสื่อสารกันผ่านช่องทาง ShopProductPort โดยใช้ isExistProduct checkOutOfStockProduct และ performBuyProduct จากนั้น ShopProduct.wsdl จะไปเรียกใช้ ShopProductSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์



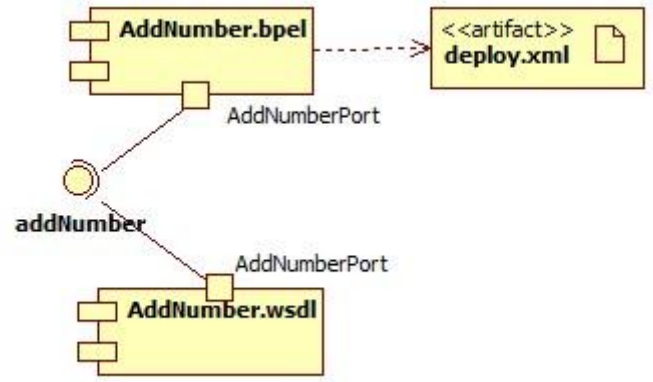
ภาพที่ ค.5 แผนภาพคอมโพเนนท์ของโปรแกรม ShopProductProcess

6) โปรแกรม TravelReservationProcess ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 6 คอมโพเนนท์ คือ deploy.xml TravelReservationProcess.bpel เอกสารวิซเดิล TravelReservationProcessArtifacts.wsdl เอกสารวิซเดิล TravelReservation.wsdl TravelReservationSoapBindingImpl และฐานข้อมูลซึ่งใช้ PostgreSQL Database โดย TravelReservationProcess.bpel กับ TravelReservationProcessArtifacts.wsdl จะติดต่อสื่อสารระหว่างกันผ่านช่องทาง TravelReservationProcessPort ด้วย getFlightAndHotelReservation ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ TravelReservationProcessArtifacts.wsdl กับ TravelReservation.wsdl ยังติดต่อสื่อสารกันผ่านช่องทาง TravelReservationPort โดยใช้ checkAvailableFlight checkAvailableHotel checkCanEffordable และ performReserve จากนั้น TravelReservation.wsdl จะไปเรียกใช้ TravelReservationSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์



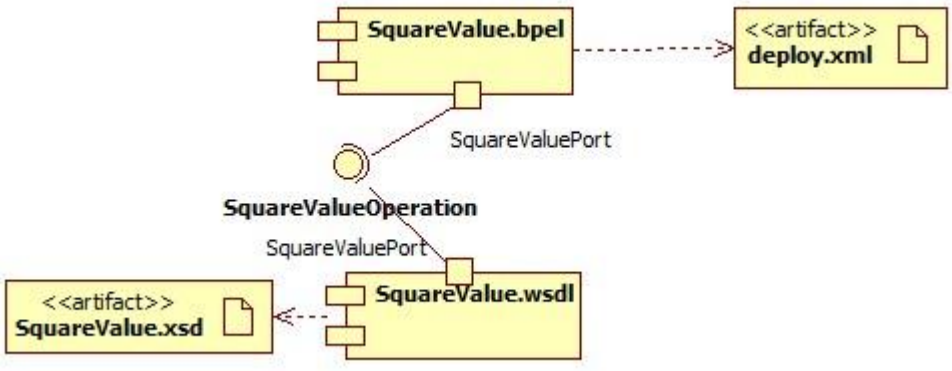
ภาพที่ ค.6 แผนภาพคอมโพเนนท์ของโปรแกรม TravelReservationProcess

7) โปรแกรม AddNumber ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 3 คอมโพเนนท์ คือ deploy.xml AddNumber.bpel และ AddNumber.wsdl โดย AddNumber.bpel กับ AddNumber.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง AddNumberPort ด้วย addNumber ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร



ภาพที่ ค.7 แผนภาพคอมโพเนนท์ของโปรแกรม AddNumber

8) โปรแกรม SquareValue ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 4 คอมโพเนนท์ คือ deploy.xml SquareValue.bpel SquareValue.wsdl และ SquareValue.xsd โดย SquareValue.bpel กับ SquareValue.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง SquareValuePort ด้วย SqureValueOperation โดยมี SquareValue.xsd ในการบอกรูปแบบและชนิดของข้อมูลที่ใช้ในการติดต่อสื่อสารกัน ส่วน deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร



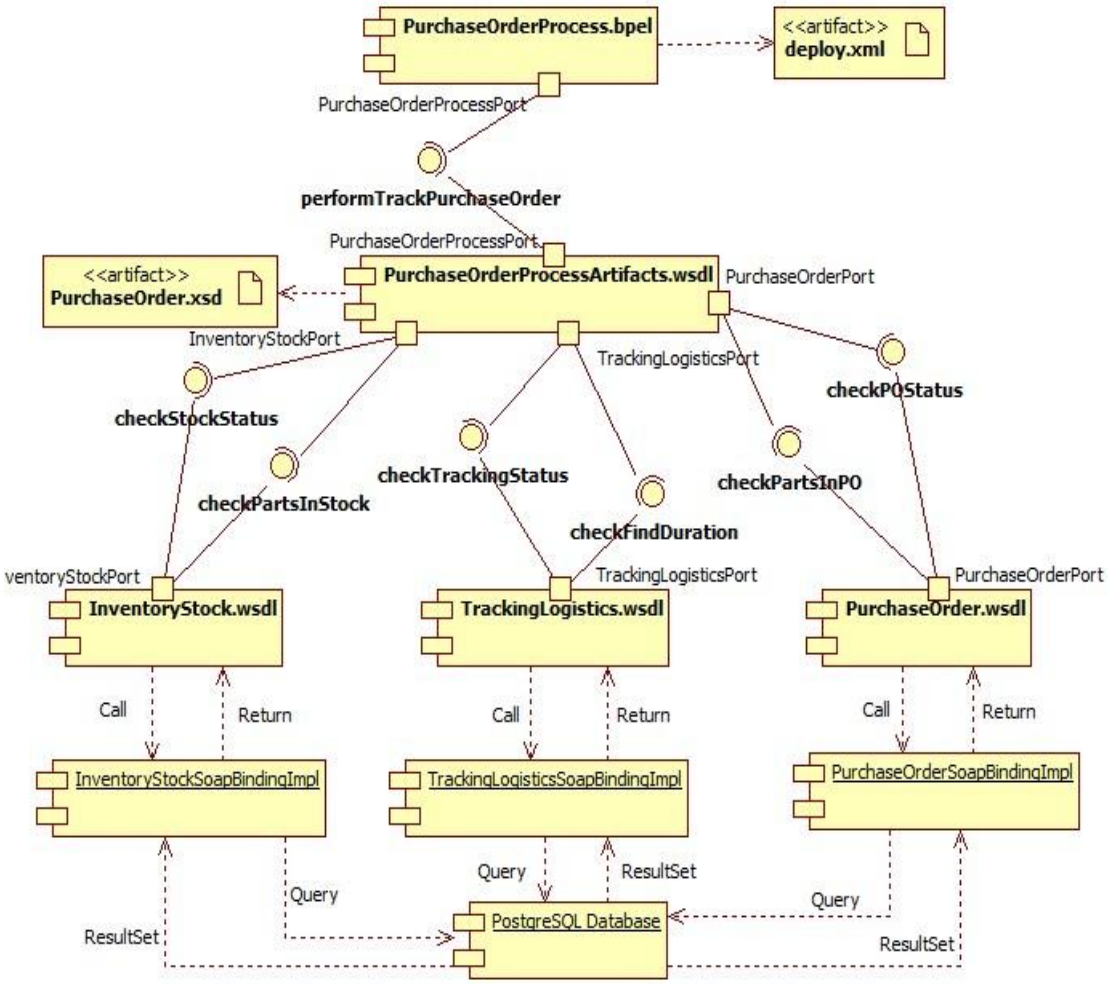
ภาพที่ ค.8 แผนภาพคอมโพเนนท์ของโปรแกรม SquareValue

9) โปรแกรม PurchaseOrderProcess ประกอบด้วยคอมโพเนนท์ที่ทำงานร่วมกันได้อยู่ 11 คอมโพเนนท์ คือ deploy.xml โปรแกรมบีเพล PurchaseOrderProcess.bpel เอกสารวิซเดิล PurchaseOrderProcessArtifacts.wsdl เอกสารสคีมา PurchaseOrder.xsd เอกสารวิซเดิล PurchaseOrder.wsdl เอกสารวิซเดิลInventoryStock.wsdl เอกสารวิซเดิล TrackingLogistics.wsdl จาวาคลาสเว็บเซอร์วิซของ PurchaseOrderSoapBindingImpl และ InventoryStockSoapBindingImpl TrackingLogisticsSoapBindingImpl และ PostgreSQL Database โดย PurchaseOrderProcess.bpel กับ PurchaseOrderProcessArtifacts.wsdl จะติดต่อสื่อสารกันผ่านช่องทาง PurchaseOrderProcessPort ด้วย performTrackPurchaseOrder โดยมี PurchaseOrder.xsd เพื่อบอกรูปแบบและชนิดของข้อมูลที่ใช้ในการติดต่อสื่อสารกัน ซึ่งมี deploy.xml อธิบายถึงกระบวนการบีเพลและช่องทางการติดต่อสื่อสาร ทั้งนี้ PurchaseOrderProcessArtifacts.wsdl ยังติดต่อกับคอมโพเนนท์อื่นๆ ดังต่อไปนี้

1. PurchaseOrder.wsdl ติดต่อสื่อสารกันผ่านช่องทาง PurchaseOrderPort โดยใช้ checkPOStatus และ checkPartsInPO โดยจะไปเรียกใช้ PurchaseOrderSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์

2. InventoryStock.wsdl ติดต่อสื่อสารกันผ่านช่องทาง InventoryStockPort โดยใช้ checkStockStatus และ checkPartsInStock โดยจะไปเรียกใช้ InventoryStockSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์

3. TrackingLogistics.wsdl ติดต่อสื่อสารกันผ่านช่องทาง TrackingLogisticsPort โดยใช้ checkTrackingStatus และ checkFindDuration โดยจะไปเรียกใช้ TrackingLogisticsSoapBindingImpl ซึ่งต่อผ่านฐานข้อมูล PostgreSQL Database ในการหาผลลัพธ์



ภาพที่ ค.9 แผนภาพคอมโพเนนท์ของโปรแกรม PurchaseOrderProcess

ค.2 ข้อมูลและชนิดของข้อมูลที่ใช้

ในหัวข้อเป็นการแสดงข้อมูลที่นำเข้าและส่งออกเพื่อใช้ในการติดต่อสื่อสารกันไปมาภายในกระบวนการบีเพล ซึ่งข้อมูลที่นำเข้าและส่งออกนั้นใช้ในกระบวนการบีเพลทั้งหมด 9 โปรแกรมบีเพล คือ โปรแกรมหาชนิดของสามเหลี่ยม โปรแกรมเครื่องคิดเลข โปรแกรมการอนุมัติการกู้ยืม โปรแกรมเอทีเอ็ม โปรแกรมการซื้อสินค้า โปรแกรมการจองตั๋วท่องเที่ยว โปรแกรมเพิ่มค่าตัวเลข โปรแกรมหาค่ากำลังสอง และโปรแกรมใบสั่งซื้อสินค้า ดังตารางที่ ค.1 – ค.9

ตารางที่ ค.1 ชื่อและชนิดข้อมูลเข้าข้อมูลออกของโปรแกรม Triangle

TriangleRequestMessage	
Name	Type
A	int
B	int
C	int
TriangleResponseMessage	
Name	Type
Result	string

ตารางที่ ค.2 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม LoanApprovalProcess

LoanApprovalProcessRequestMessage	
Name	Type
amount	double
name	string
LoanApprovalProcessResponseMessage	
Name	Type
result	string

ตารางที่ ค.3 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม ATMProcess

ATMProcessRequestMessage	
Name	Type
amount	double

ตารางที่ ค.3 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม ATMProcess (ต่อ)

name	string
pin	string
ATMProcessResponseMessage	
Name	**Type**
result	string

ตารางที่ ค.4 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม SimpleCalculator

SimpleCalculatorRequestMessage	
Name	**Type**
A	int
B	int
Symbol	string
SimpleCalculatorResponseMessage	
Name	**Type**
result	string

ตารางที่ ค.5 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม ShopProductProcess

ShopProductProcessRequestMessage	
Name	**Type**
amount	double
name	String
quantity	double
ShopProductProcessResponseMessage	
Name	**Type**
result	string

ตารางที่ ค.6 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม TravelReservationProcess

TravelReservationProcessResponseMessage	
Name	Type
airline	string
amount	double
dayrest	int
destination	string
time	double
TravelReservationProcessResponseMessage	
Name	Type
Result	string

ตารางที่ ค.7 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม AddNumber

AddNumberInputMessage	
Name	Type
from	int
to	Int
AddNumberOutputMessage	
Name	Type
result	string

ตารางที่ ค.8 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม SquareValue

SquareValueOperationRequest	
Name	Type
n	unsignedInt
SquareValueOperationResponse	
Name	Type
sum	double
variance	double

ตารางที่ ค.9 ชื่อและชนิดข้อมูลเข้าและข้อมูลออกของโปรแกรม PurchaseOrderProcess

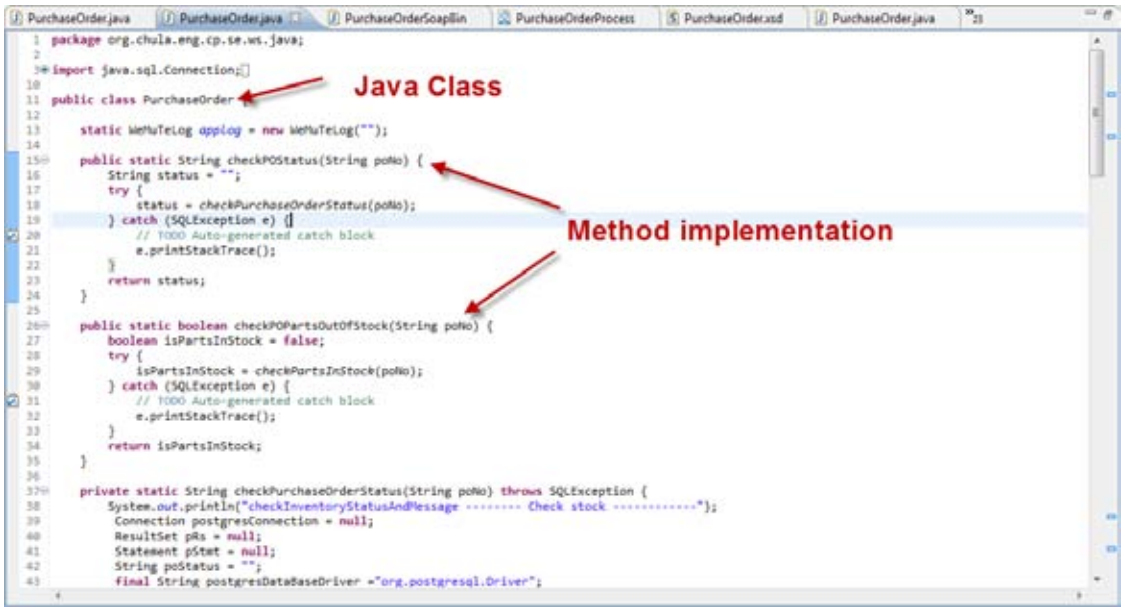
PurchaseOrderProcessRequest	
Name	**Type**
poId	int
poNo	string
poType	string
poDate	date
sumQty	double
sumAmt	double
sumDisc	double
remark	string
PurchaseOrderProcessResponse	
Name	**Type**
result	string

ค.3 การสร้างเว็บเซอร์วิซ

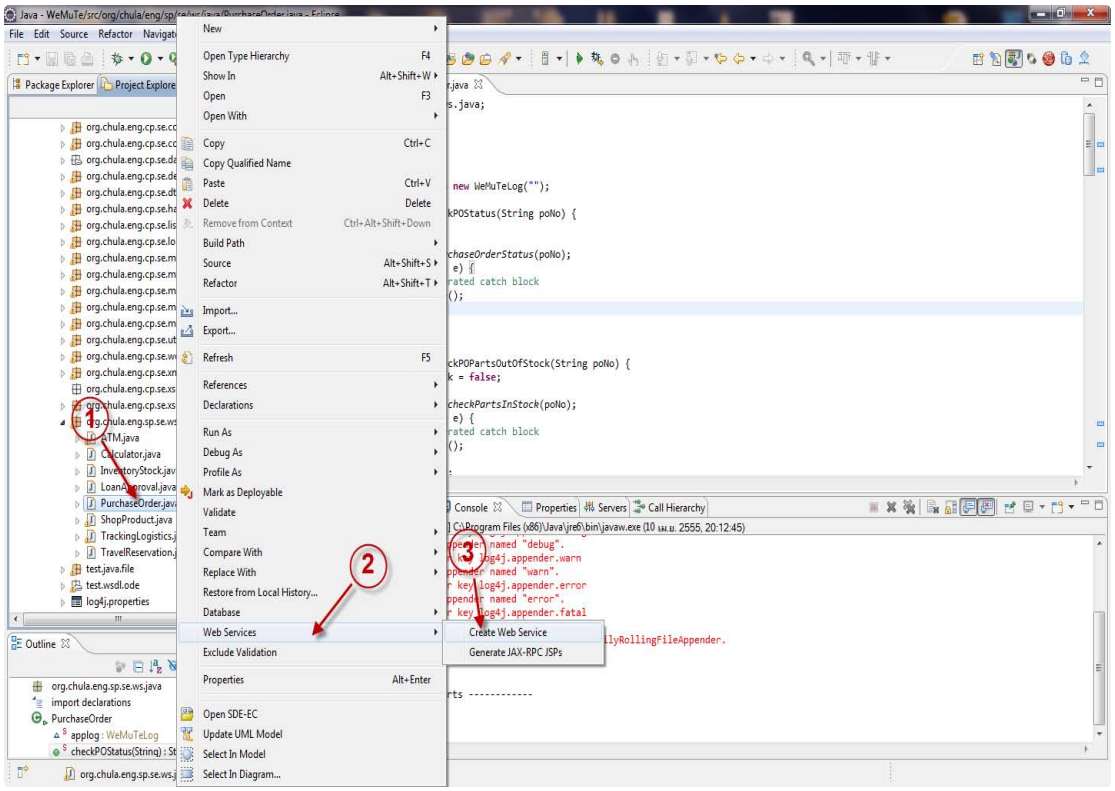
ในหัวข้อนี้อธิบายถึงการสร้างเว็บเซอร์วิซจากการเขียนจาวาคลาสโดยใช้โปรแกรมอีคลิป ซึ่งขั้นตอนทั้งหมดสามารถอธิบายได้ดังต่อไปนี้

1) ผู้ใช้จำเป็นต้องพัฒนาโปรแกรมโดยใช้ภาษาจาวาขึ้นมาก่อน ดังภาพที่ ค.10 สร้างคลาสชื่อ PurchaseOrder โดยมีฟังก์ชันที่ใช้ภายในคือ checkPOStatus และ checkPOPartsOutOfStock

2) เมื่อพัฒนาโปรแกรมเสร็จเรียบร้อยแล้ว คลิกขวาที่คลาส PurchaseOrder เลือก Web Services -> Create Web Service เพื่อเริ่มสร้างเว็บเซอร์วิซ ดังภาพที่ ค.11



ภาพที่ ค.10 จาวาคลาสที่ใช้สร้างเว็บเซอร์วิซ



ภาพที่ ค.11 การสร้างเว็บเซอร์วิซจากจาวาคลาส

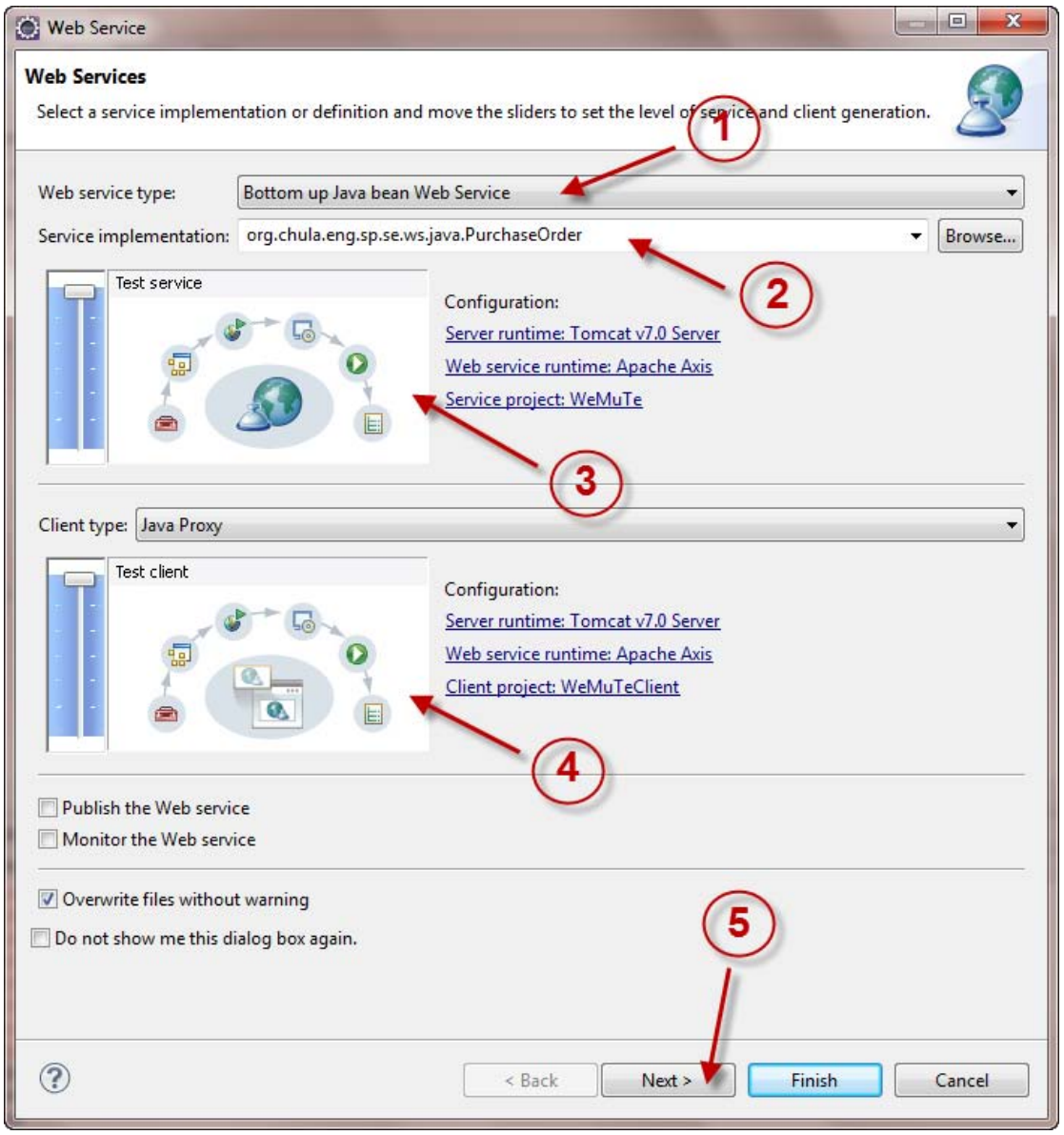
3) โปรแกรมอีคลิปจะแสดงหน้าต่าง Web Service ดังภาพที่ ค.12 จากนั้นเลือก Web service type เป็น Bottom up Java bean Web Service และในส่วน Service implementation โปรแกรมอีคลิปจะเลือกโปรแกรมที่เราต้องการสร้างเว็บเซอร์วิซให้อยู่แล้ว และเลือก Configuration เป็นแบบ Test service ซึ่งมีอยู่ทั้งหมด 6 แบบดังตารางที่ ค.10 ในส่วนของ Client type เลือกเป็น Java Proxy เลือก Configuration เป็นแบบ Test client ซึ่งมีอยู่ทั้งหมด 6 แบบดังตารางที่ ค.11 เมื่อเลือกค่าที่ต้องการเรียบร้อยจึงกดปุ่ม Next

ตารางที่ ค.10 Service implementation configuration

Configuration	
Test service	✓
Start service	✓
Install service	✓
Deploy service	✓
Assemble service	✓
Develop service	✓

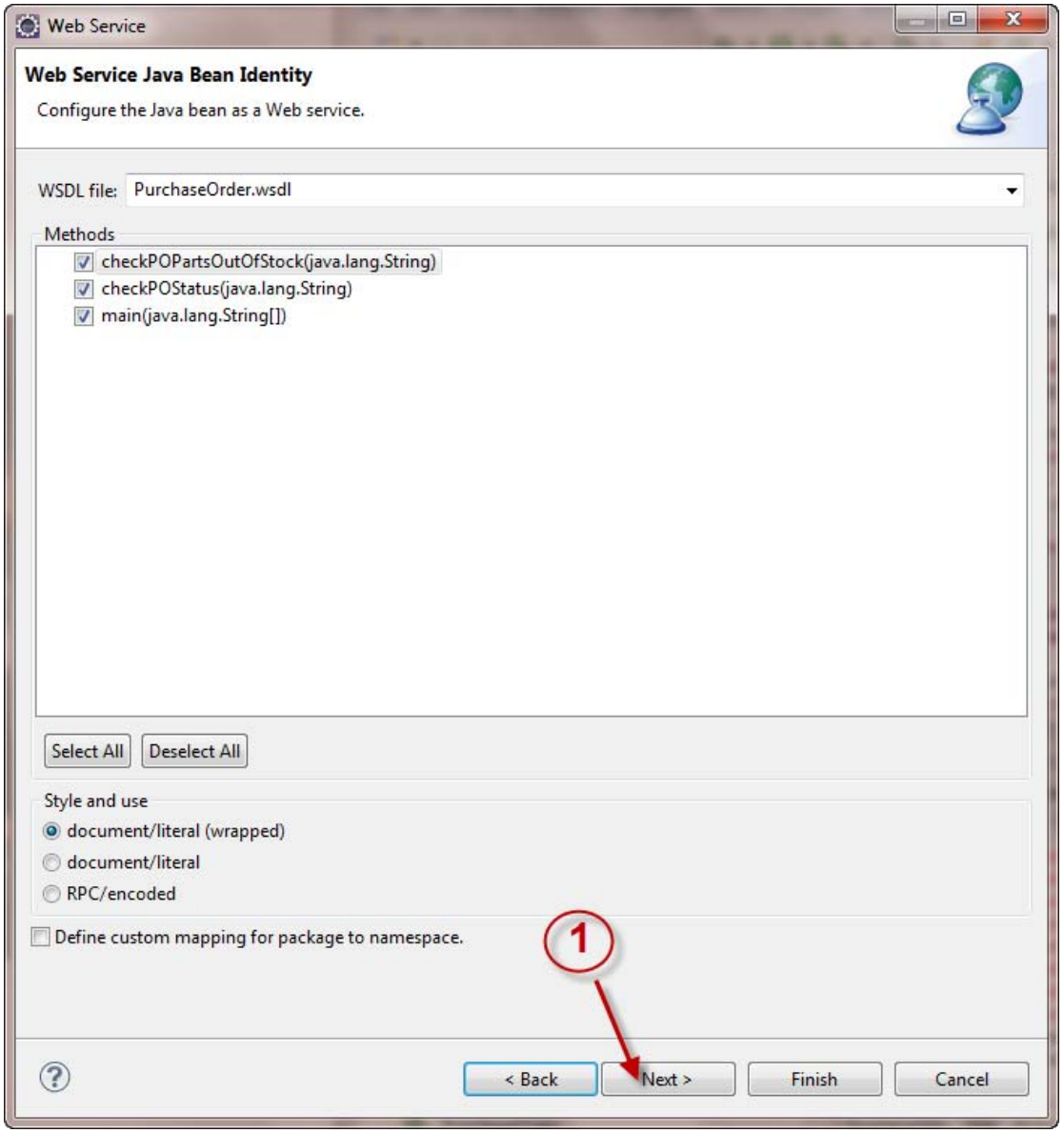
ตารางที่ ค.11 Client type configuration

Configuration	
Test client	✓
Start client	✓
Install client	✓
Deploy client	✓
Assemble client	✓
Develop client	✓



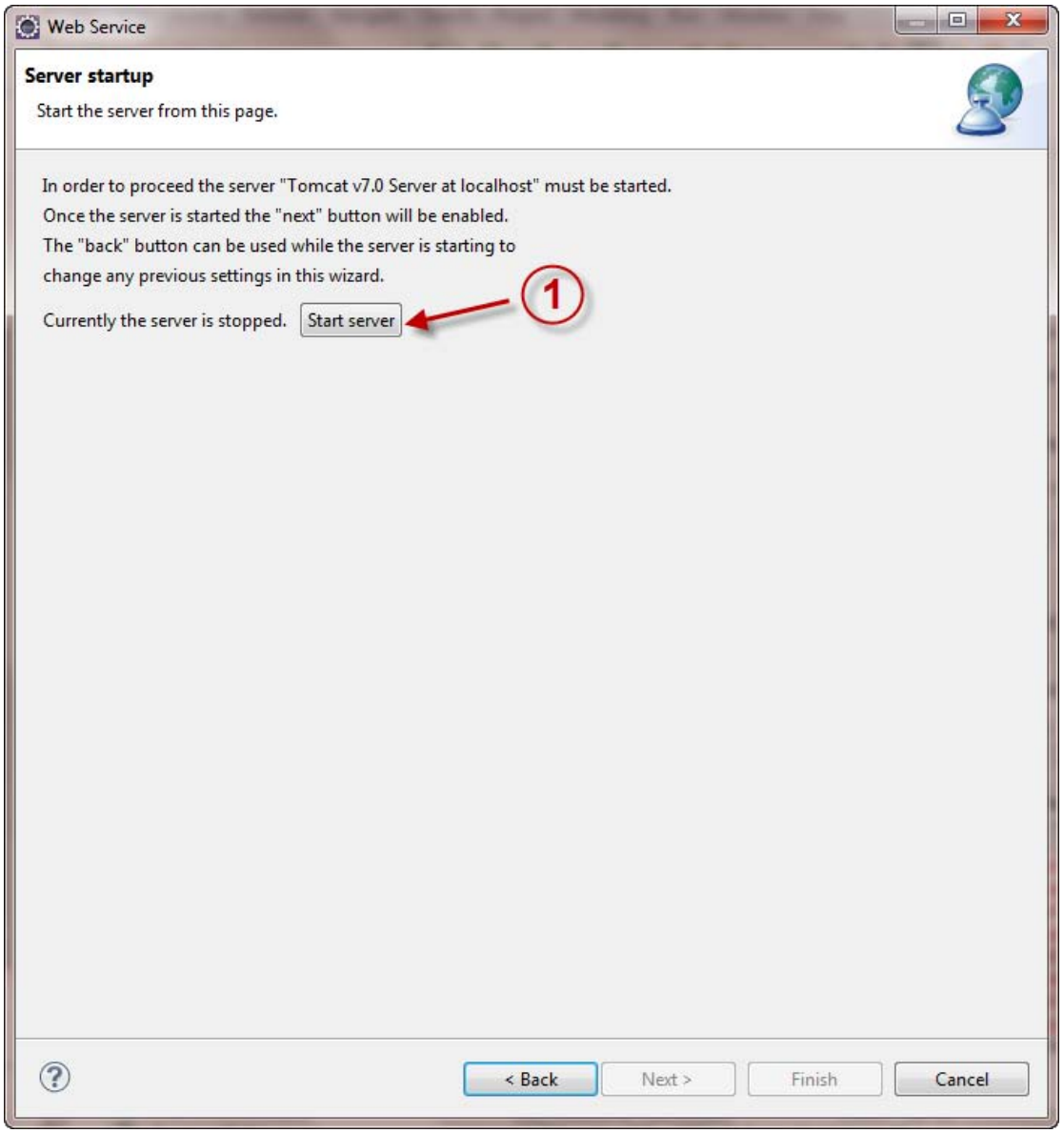
ภาพที่ ค.12 หน้าต่าง Web Services

4) โปรแกรมอีคลิปจะแสดงหน้าต่าง Web Service Java Bean Indentity เพื่อสร้างเอกสารวิซเดิล ซึ่งสามารถกำหนดชื่อเอกสารวิซเดิลในส่วน WSDL file จากนั้นในหัวข้อ Methods โปรแกรมจะให้เลือกฟังก์ชันที่เราต้องการจะนำไปสร้างในเว็บเซอร์วิซ เมื่อเลือกฟังก์ชันเรียบร้อยแล้ว ในส่วน Style and use จึงเลือก document/literal (wrapped) จากนั้นกดปุ่ม Next ดังภาพที่ ค.13

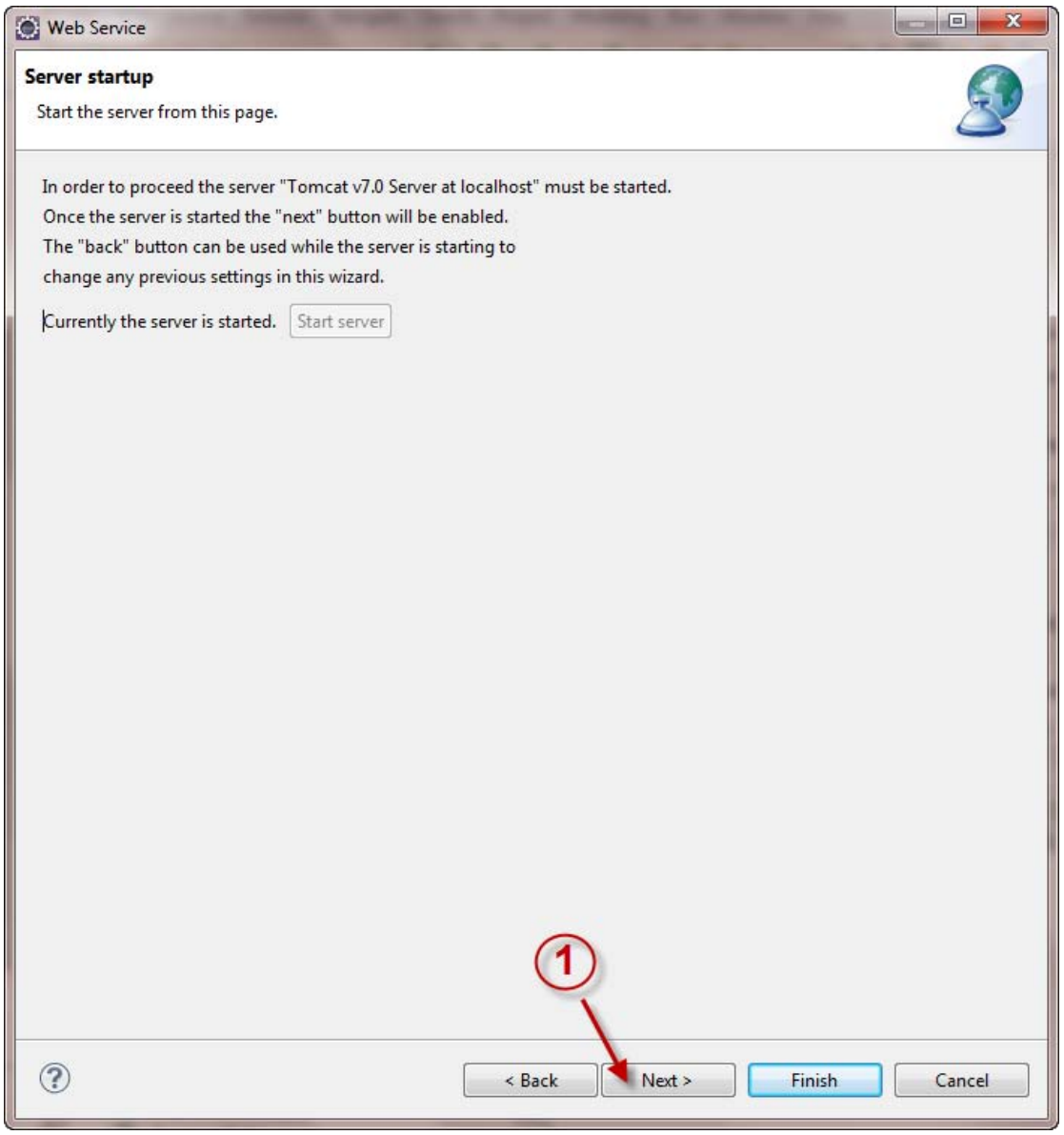


ภาพที่ ค.13 หน้าต่าง Web Service Java Bean Identity

5) จากนั้นโปรแกรมอีคลิปจะแสดงหน้าต่าง Server startup ถ้าในโปรแกรมยังไม่ได้ start Runtime server จากนั้นกดปุ่ม Start server ดังภาพที่ ค.14 เมื่อโปรแกรมอีคลิป start server เสร็จเรียบร้อยดังภาพที่ ค.15 กดปุ่ม Next

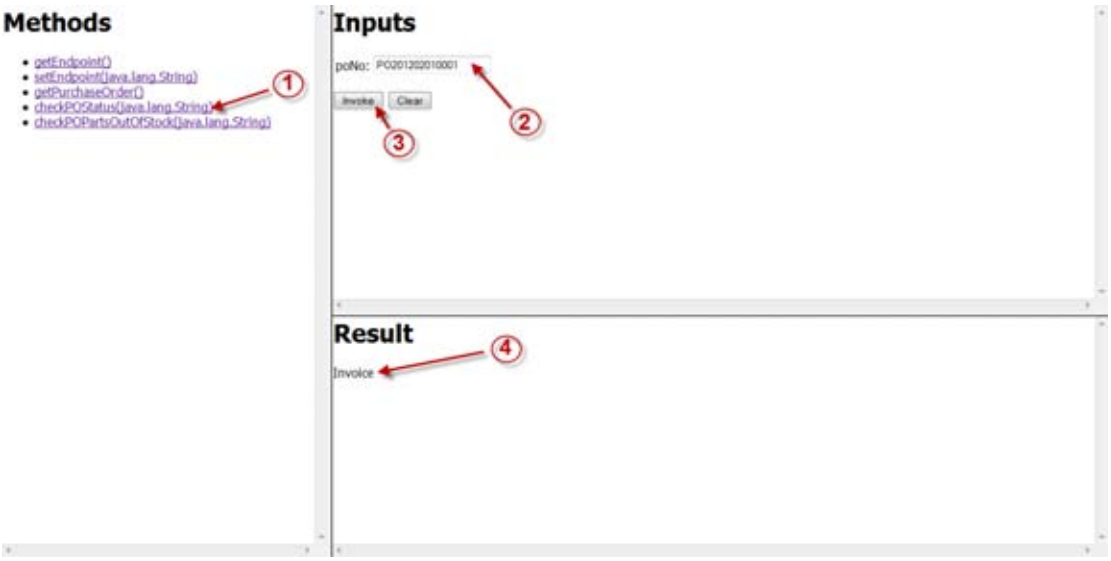


ภาพที่ ค.14 หน้าต่าง Server startup

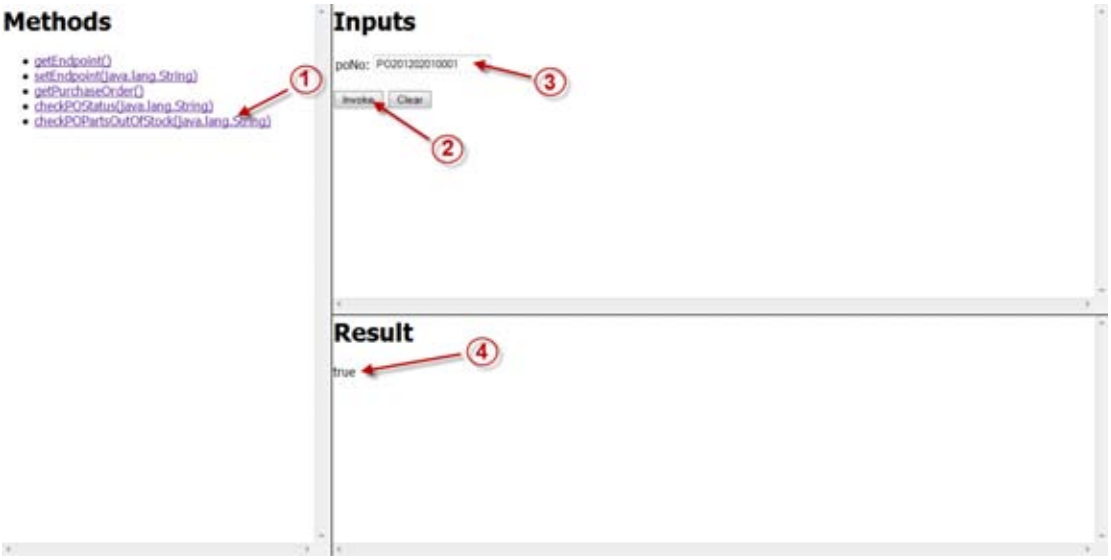


ภาพที่ ค.15 หน้าต่าง Server startup เรียบร้อย

6) จากนั้นโปรแกรมอีคลิปจะแสดงบราวเซอร์ในโปรแกรมอีคลิปเพื่อให้ผู้ใช้สามารถทดสอบลองใช้เรียกเซอร์วิซที่สร้างขึ้นได้ ดังภาพที่ ค.16 – ค.17 โดยจะมีส่วนที่เป็น Methods ในการเรียกใช้ทดสอบ มีส่วน Inputs เพื่อใส่ข้อมูลเพื่อใช้ทดสอบ และส่วนของ Results เพื่อให้เว็บเซอร์วิซแสดงผลลัพธ์

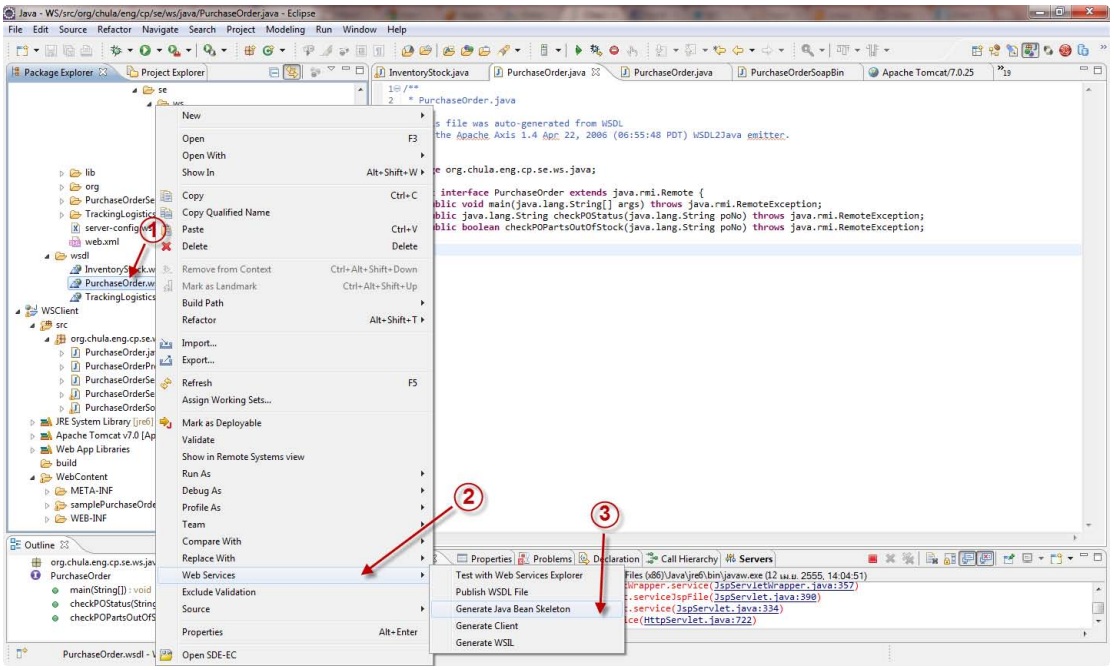


ภาพที่ ค.16 ทดสอบการเรียกใช้ checkPOStatus



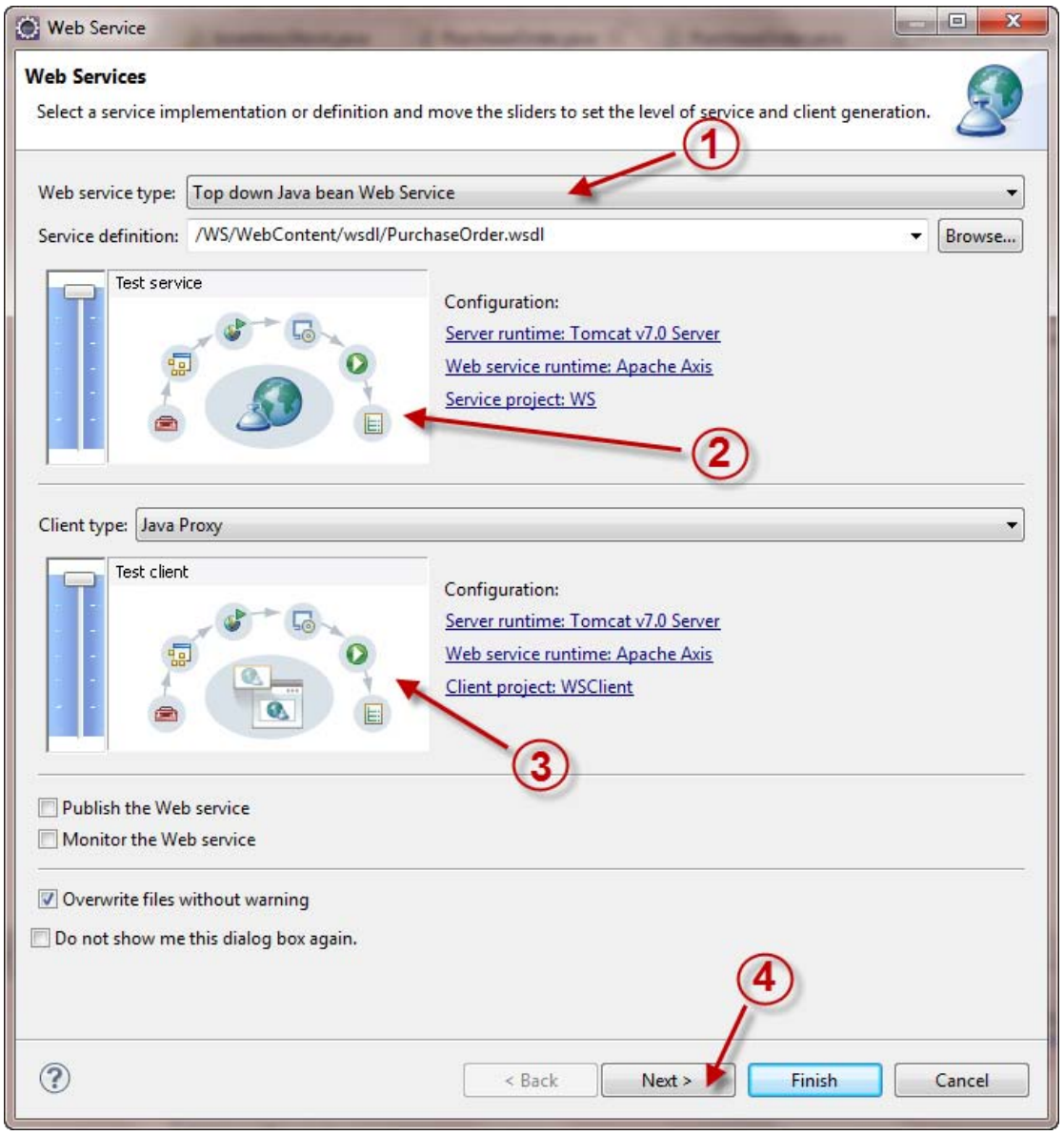
ภาพที่ ค.17 ทดสอบการเรียกใช้ checkPOPartsOutOfStock

7) จากนั้นกดคลิกขวาที่เอกสารวิซเดิลที่โปรแกรมอีคลิปสร้างขึ้นมาเพื่อสร้างจาวาคลาส ดังภาพที่ ค.18 โดยเลือก Web Services -> Generate Java Bean Skeleton



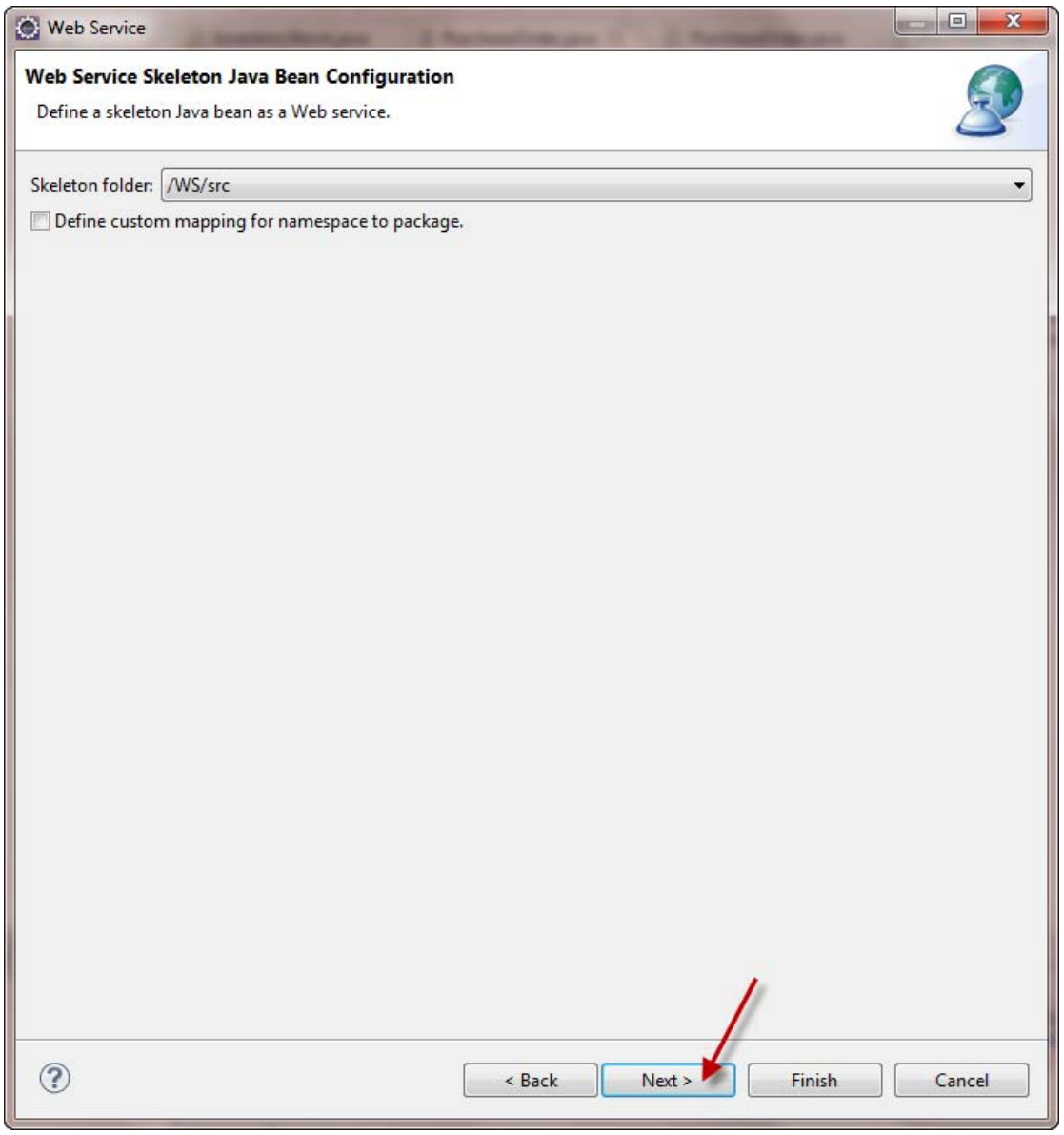
ภาพที่ ค.18 คำสั่งสร้างจาวาคลาสจากเอกสารวิซเดิล

8) จากนั้นโปรแกรมอีคลิปจะแสดงหน้าต่าง Web Services เพื่อให้เลือก Web service type ซึ่งเลือกเป็น Top down java bean Web service ถัดมาจะเป็นส่วน Service definition ซึ่งโปรแกรมอีคลิปจะเลือกให้กับผู้ใช้อยู่แล้ว จากนั้นในส่วน Service และ Client ให้เลือกที่ระดับ Test service และ Test client ตามลำดับ แล้วจึงกดปุ่ม Next เพื่อดำเนินการต่อ ดังภาพที่ ค.19



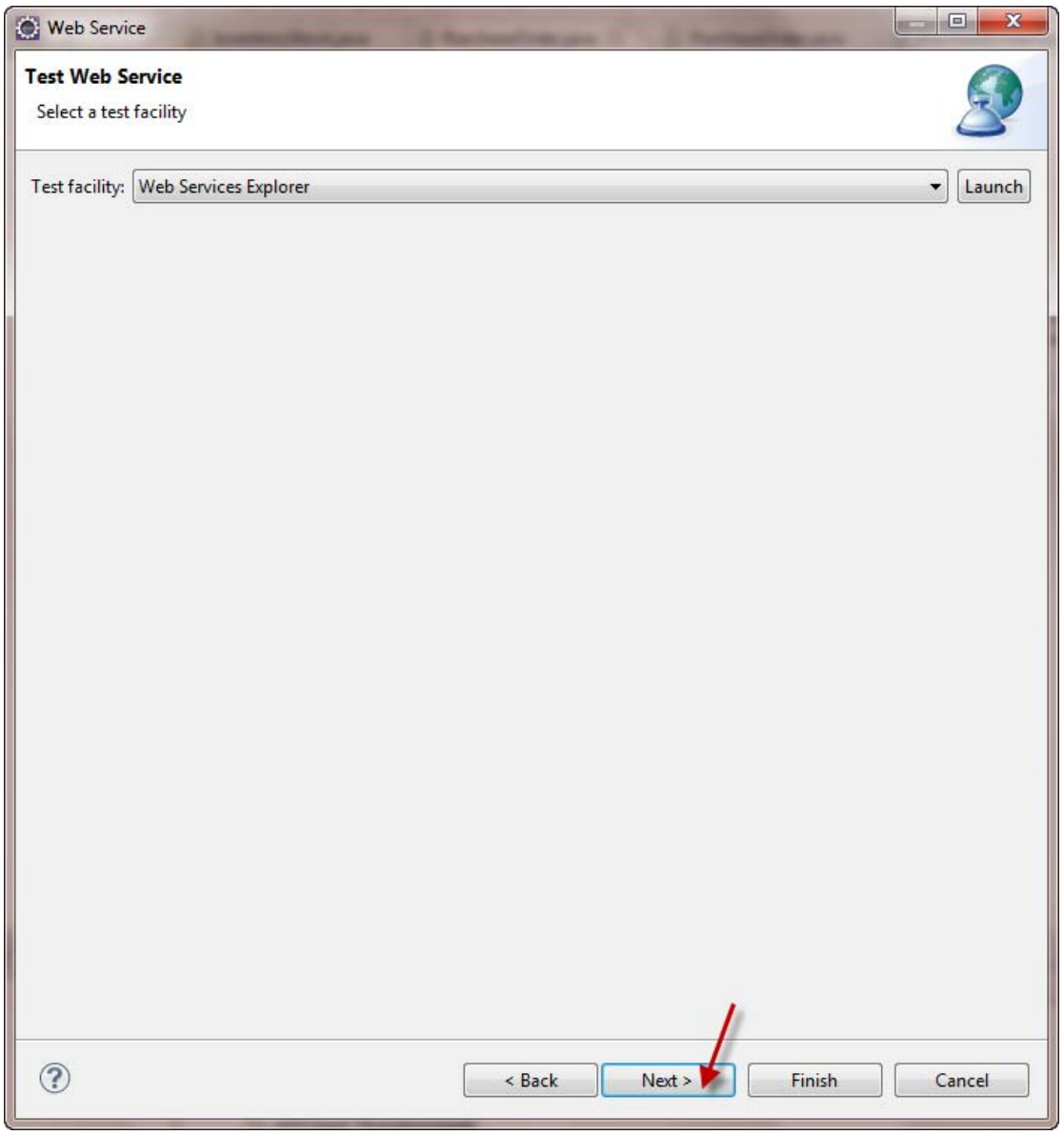
ภาพที่ ค.19 หน้าต่าง Web Services สร้างจาวาคลาสจากเอกสารวิซเดิล

9) จากนั้นโปรแกรมอีคลิปจะแสดงหน้าต่าง Web Service Skeleton Java Bean Configuration เพื่อเลือกไดเรกทอรี่ที่ใช้สร้างจาวาคลาสจากเอกสารวิซเดิล จากนั้นกดปุ่ม Next ดังภาพที่ ค.20



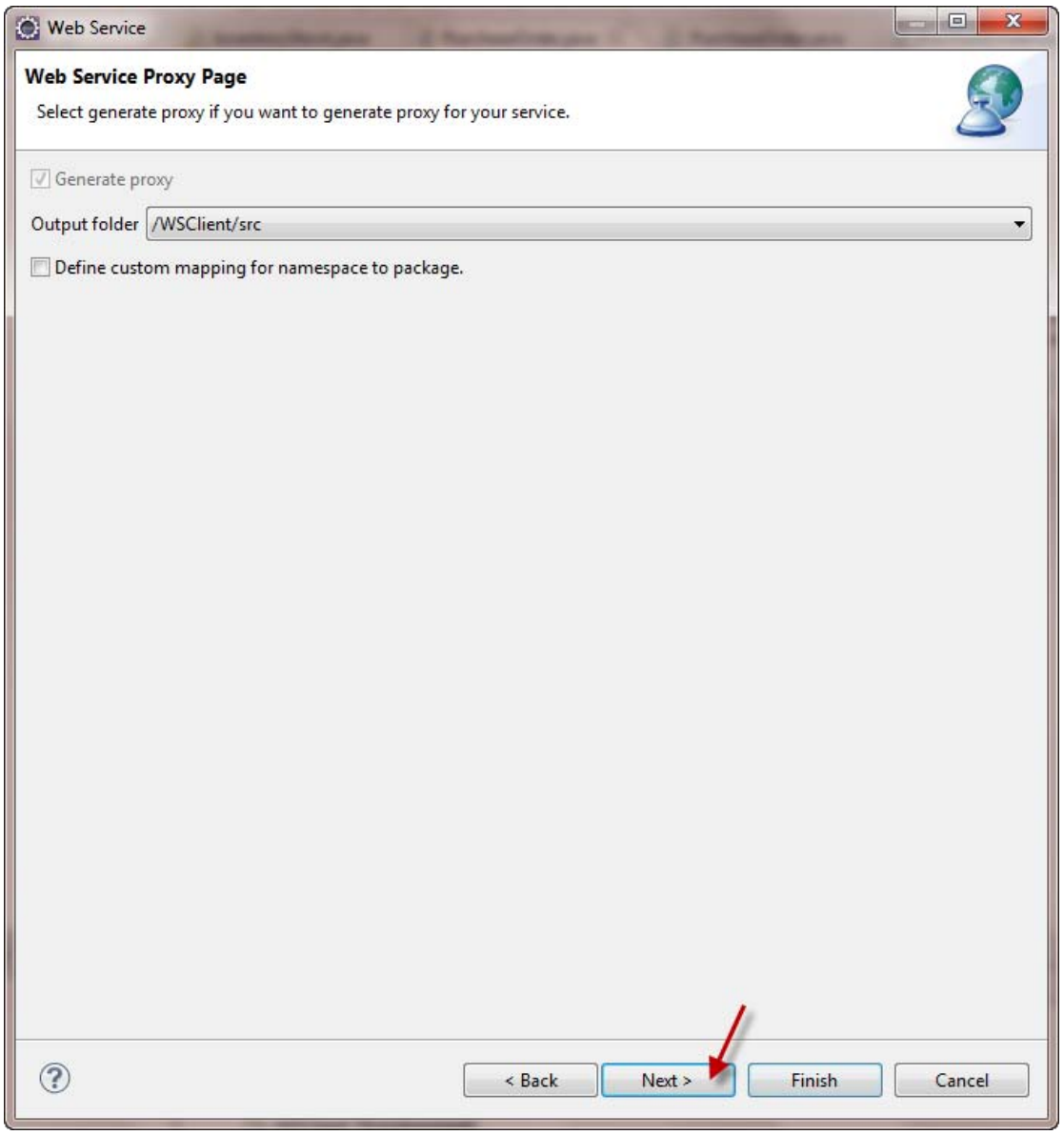
ภาพที่ ค.20 หน้าต่าง Web Service Skeleton Java Bean Configuration

10) โปรแกรมอีคลิปแสดงหน้าต่าง Test Web Service เพื่อเลือก Test facility ซึ่งโปรแกรมจะเลือก Web Services Explorer ให้ผู้ใช้อยู่แล้ว ดังภาพที่ ค.21



ภาพที่ ค.21 หน้าต่าง Test Web Service

11) จากนั้นที่หน้าต่าง Web Service Proxy Page เลือกไดเรกทอรี่เพื่อให้โปรแกรมสร้างไฟล์ที่ใช้ติดต่อกับเว็บเซอร์วิซ ดังภาพที่ ค.22

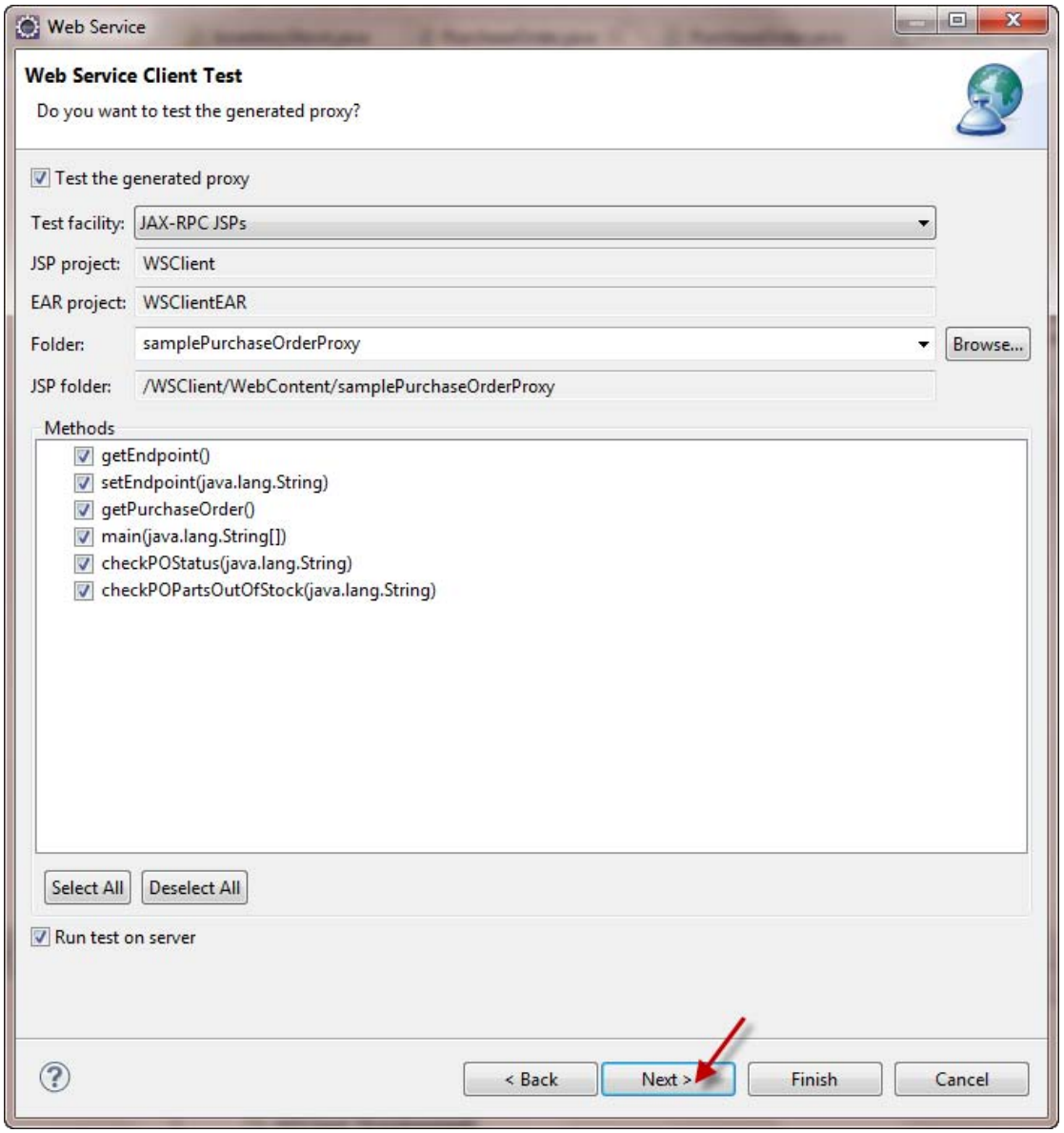


ภาพที่ ค.22 หน้าต่าง Web Service Proxy Page

12) จากนั้นโปรแกรมอีคลิปจะแสดงหน้าต่าง Web Service Client Test เพื่อให้ผู้ใช้เลือกค่าที่จำเป็นในการสร้าง ดังนี้

- Test facility: JAX-RPC JSPs
- JSP Project: WSClient
- EAR Project: WSClientEAR
- Folder: samplePurchaseOrderProxy
- JSP folder: /WSClient/WebContent/samplePurchaseOrderProxy

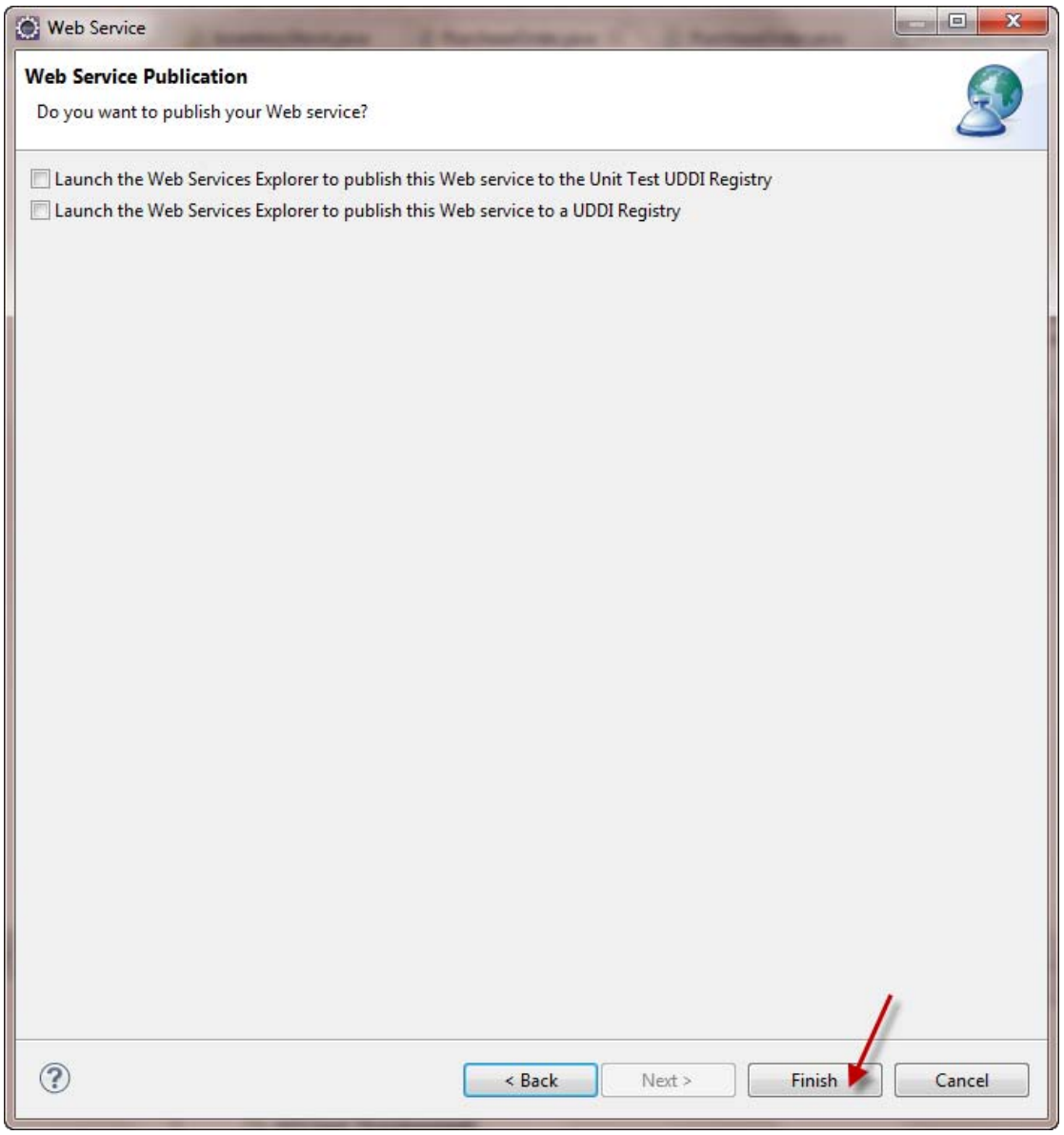
ซึ่งโปรแกรมอีคลิปได้กำหนดค่าที่จำเป็นให้กับผู้ใช้เรียบร้อยแล้ว จากนั้นกำหนดค่าที่ผู้ใช้ต้องการแล้วจึงกดปุ่ม Next ดังภาพที่ ค.23



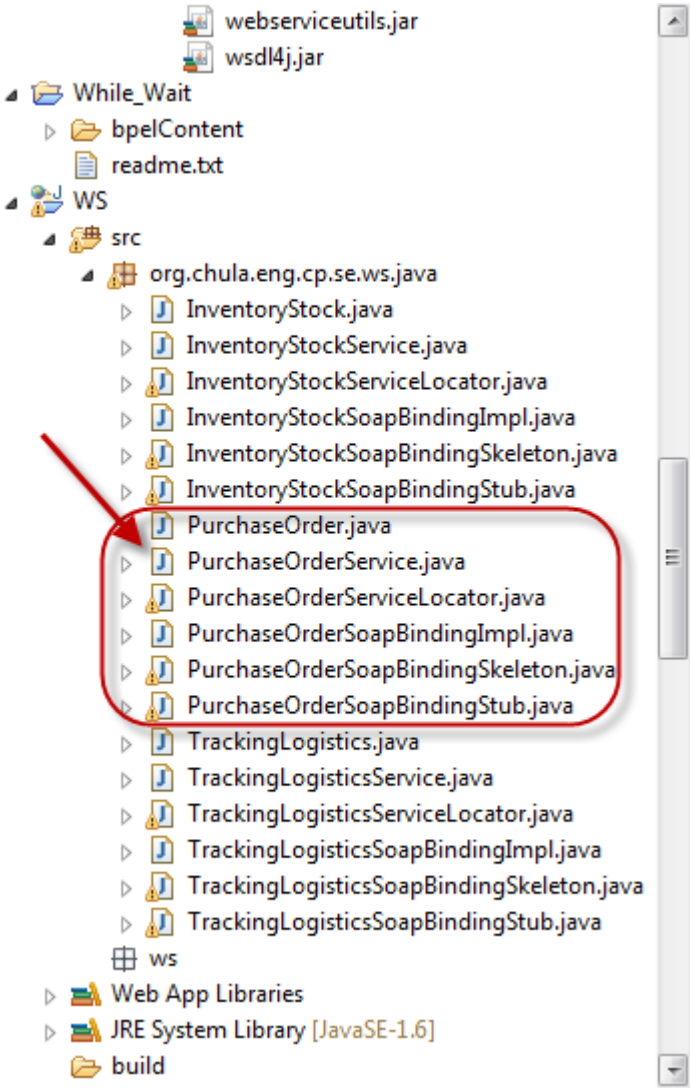
ภาพที่ ค.23 หน้าต่าง Web Service Client Test

13) จากนั้นโปรแกรมจะแสดงหน้าต่าง Web Service Publication ซึ่งในขั้นตอนนี้ผู้ใช้สามารถกดปุ่ม Finish ได้เลย ดังภาพที่ ค.24

14) เมื่อโปรแกรมดำเนินการเสร็จเรียบร้อย ผู้ใช้สามารถตรวจสอบเอกสารที่โปรแกรมสร้างขึ้นมาได้ ดังภาพที่ ค.25



ภาพที่ ค.24 หน้าต่าง Web Service Publication



ภาพที่ ค.25 แสดงการสร้างเว็บเซอร์วิซเสร็จเรียบร้อย

ประวัติผู้เขียนวิทยานิพนธ์

นายปัญญา บุญยกุลศรีรุ่ง เกิดเมื่อวันที่ 6 มกราคม พ.ศ. 2526 สำเร็จการศึกษาระดับปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรศาสตร์ จากคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ บางเขน ในปีการศึกษา 2544 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2552

ขณะที่ศึกษานั้น ผู้วิจัยได้ร่วมทำบทความกับอาจารย์ที่ปรึกษา ซึ่งมีบทความที่ได้รับการคัดเลือกเพื่อนำเสนอและตีพิมพ์ในงานประชุมวิชาการในระดับนานาชาติ รวมทั้งสิ้น 2 บทความ โดยมีรายละเอียดดังต่อไปนี้

1) บทความวิชาการเรื่อง “A Weak Mutation Testing Framework for WS-BPEL” ซึ่งได้รับการคัดเลือกเพื่อนำเสนอและตีพิมพ์ในงาน “การประชุมวิชาการร่วมสาขาวิทยาการคอมพิวเตอร์และวิศวกรรมซอฟต์แวร์ ครั้งที่ 8 (The 8th International Joint Conference on Computer Science and Software Engineering: JCSSE 2011)” ระหว่างวันที่ 11 - 13 พฤษภาคม 2554 ณ คณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยมหิดล วิทยาเขตศาลายา นครปฐม ประเทศไทย

2) บทความวิชาการเรื่อง “WeMuTe - A Weak Mutation Testing Tool for WS-BPEL” ซึ่งได้รับการคัดเลือกเพื่อนำเสนอและตีพิมพ์ในงาน “International MultiConference of Engineers and Computer Scientists 2012: IMECS 2012” ระหว่างวันที่ 14 - 16 มีนาคม 2555 ณ โรงแรมรอยัลการ์เด้น เกาลูน ฮ่องกง ประเทศจีน