การสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ

นางสาวพจนา แซวประโคน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2556
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

บทคัดย่อและแฟ้มข้อมูลฉบับเต็มของวิทยานิพนธ์ตั้งแต่ปีการศึกษา 2554 ที่ให้บริการในคลังปัญญาจุฬาฯ (CUIR)
เป็นแฟ้มข้อมูลของนิสิตเจ้าของวิทยานิพนธ์ ที่ส่งผ่านทางบัณฑิตวิทยาลัย
The abstract and full text of theses from the academic year 2011 in Chulalongkorn University Intellectual Repository (CUIR)
are the thesis authors' files submitted through the University Graduate School.

SEQUENCE DIAGRAM GENERATION WITH A MODEL TRANSFORMATION TECHNOLOGY

Miss Photchana Sawprakhon

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

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 2013

Copyright of Chulalongkorn University

หัวข้อวิทยานิพนธ์	การสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ
โดย	นางสาวพจนา แซวประโคน
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร.บัณฑิต เอื้ออาภรณ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์)

.....กรรมการภายนอกมหาวิทยาลัย

(อาจารย์ ดร.ภาสกร อภิรักษ์วรพินิต)

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

พจนา แซวประโคน : การสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ. (SEQUENCE DIAGRAM GENERATION WITH A MODEL TRANSFORMATION TECHNOLOGY) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร.ญาใจ ลิ่มปิยะกรณ์, 115 หน้า.

การสร้างแผนภาพลำดับด้วยเครื่องมือยูเอ็มแอล อาจจะไม่สมบูรณ์ ไม่สอดคล้อง และไม่ถูกต้อง วิธีนี้ต้องใช้ความเชี่ยวชาญ แรงงาน และเวลา ด้วยเทคโนโลยีการแปลงตัวแบบ วิทยานิพนธ์ฉบับนี้ได้นำเสนอแนวทางการสร้างแผนภาพลำดับยูเอ็มแอลแบบอัตโนมัติจากคำอธิบายยูสเคสและแผนภาพคลาส เอทีแอลถูกใช้เป็นภาษาการแปลงตัวแบบสำหรับการแปลงเมทาโมเดลต้นทางของคำอธิบายยูสเคสและแผนภาพคลาส ให้เป็นเมทาโมเดลเป้าหมายของแผนภาพลำดับ ด้วยการเรียกใช้เอพีไอของซอฟต์แวร์โอเพนซอร์ส ไฟล์ผลลัพธ์ที่อยู่ในรูปแบบเอกซ์เอ็มไอจะถูกแปลงต่อให้เป็นไฟล์ข้อความที่เหมาะสำหรับการแสดงผลเป็นภาพแผนภาพลำดับซึ่งเป็นผลลัพธ์สุดท้าย วิธีการที่นำเสนอจะช่วยในการปรับปรุงกระบวนการซอฟต์แวร์ โดยการใช้เทคโนโลยีการแปลงตัวแบบจะสนับสนุนการนำกลับมาใช้ซ้ำของสารสนเทศที่ใช้ในการสร้างตัวแบบก่อนหน้า แทนที่จะสร้างตัวแบบใหม่ทุกครั้งในเฟสต่างๆระหว่างวัฏจักรการพัฒนา ทั้งยังเพิ่มความถูกต้องตรงกันระหว่างตัวแบบทั้งหลายที่แทนมุมมองต่างๆของระบบอีกด้วย

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2556

5570987821 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: MODEL-DRIVEN / MODEL TRANSFORMATION / SEQUENCE DIAGRAM / PROCESS IMPROVEMENT

PHOTCHANA SAWPRAKHON: SEQUENCE DIAGRAM GENERATION WITH A MODEL TRANSFORMATION TECHNOLOGY. ADVISOR: ASSOC. PROF. YACHAI LIMPIYAKORN, Ph.D., 115 pp.

Creating Sequence diagrams with UML tools can be incomplete, inconsistent, and incorrect. It also requires expertise, effort, and time. With model transformation technology, this thesis presents an approach to automate the generation of UML Sequence diagrams from Use Case Description and Class diagrams. ATL is used as the model transformation language for converting the source metamodels of Use Case description and Class diagrams to the target metamodel of Sequence diagram. With the API call of the Open Source software, the resulting file in XMI format is then transformed to another text file that suits for rendering the image of Sequence diagram as the final output. The proposed method would result in the improvement of software process. Rather than constructing the models from scratch during the different development life cycle stages, model transformations enable the reuse of information that was once modeled, as well as enhance the consistency among the models representing different views of the system.

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

Department: Computer Engineering Student's Signature .....
Field of Study: Software Engineering Advisor's Signature .....
Academic Year: 2013

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์อย่างดียิ่งของ รองศาสตราจารย์ ดร. ญาใจ ลิ่มปิยะกรณ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งได้ให้ความรู้ คำแนะนำ ข้อคิดเห็นต่างๆ และช่วยแก้ไขข้อบกพร่องต่างๆ ด้วยความเอาใจใส่ เพื่อให้การทำวิทยานิพนธ์ฉบับนี้สมบูรณ์ ผู้วิจัยมีความซาบซึ้งในความกรุณาอันดีจากอาจารย์ และขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบพระคุณ ผศ.ดร.สุกรี สินธุภิญโญ และอาจารย์ดร.ภาสกร อภิรักษ์วรพินิต กรรมการสอบวิทยานิพนธ์ ที่กรุณาเสียสละเวลา ให้คำแนะนำ ตรวจสอบ และแก้ไขวิทยานิพนธ์ฉบับนี้

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และญาติพี่น้อง ที่ให้กำลังใจและการสนับสนุนช่วยเหลือในด้านต่างๆ กราบขอบพระคุณอาจารย์ทุกท่านที่ได้อบรมสั่งสอนวิชาความรู้ให้ผู้วิจัยจนสามารถทำวิทยานิพนธ์นี้ได้สำเร็จ

สุดท้ายนี้ผู้วิจัยขอขอบคุณเพื่อนๆ ทุกคนที่คอยได้ให้กำลังใจและความช่วยเหลือต่างๆ ตลอดจนผู้มีพระคุณทุกท่านที่มิได้กล่าวถึง คุณประโยชน์และคุณค่าอันเกิดจากการศึกษาวิจัยครั้งนี้ ผู้วิจัยขอมอบแด่บิดา มารดา ครู อาจารย์ และผู้มีพระคุณทุกท่านด้วยความซาบซึ้งใจเป็นอย่างยิ่ง

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง
บทคัดย่อภาษาอังกฤษ ..... จ
กิตติกรรมประกาศ ..... ฉ
สารบัญ ..... ช
บทที่ 1 บทนำ 1
1.1 ที่มาและความสำคัญของปัญหา 1
1.2 วัตถุประสงค์ของการวิจัย 2
1.3 ขอบเขตของการวิจัย 2
1.4 ขั้นตอนและวิธีดำเนินการวิจัย 2
1.5 ประโยชน์ที่คาดว่าจะได้รับ 2
1.6 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์ 3
1.7 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์ 3
บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 4
2.1ทฤษฎีที่เกี่ยวข้อง 4
2.1.1 การพัฒนาการขับเคลื่อนด้วยแบบจำลอง (Model Driven Development : MDD) [1] 4
2.1.2 เอทีแอล (Atlas Transformation Language : ATL) [5] 6
2.1.3 เอกซ์เอ็มไอ (XML Metadata Interchange : XMI) [6] 8
2.1.4 คำอธิบายยูสเคส (Use Case Description) 8
2.1.5 แผนภาพคลาส (Class Diagram) 10
2.1.6 แผนภาพลำดับ (Sequence Diagram) 11
2.2 งานวิจัยที่เกี่ยวข้อง 12
2.2.1 Metamodeling and model transformations in modeling and simulation [1] 12
2.2.2 UML Sequence Diagram Generator System from Use Case Description Using Natural Language [7] 13
2.2.3 A Meta-Model for Textual Use Case Description [8] 14

หน้า

2.2.4 Translating Use Cases to Sequence Diagrams [9] 14
บทที่ 3 แนวทางในการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ 16
3.1 ภาพรวมของแนวทางในการสร้างเครื่องมือในการแปลงรูปแบบโมเดล 16
3.1.1 ส่วนนำเข้าข้อมูลคำอธิบายแผนภาพยูสเคสและแผนภาพคลาส 16
3.1.2 ส่วนการออกแบบเมทาโมเดลต้นทางของคำอธิบายแผนภาพยูสเคสโดยใช้เทคโนโลยีเอกซ์พาสต์ (XPath Technology) 18
3.1.3 ส่วนออกแบบและพัฒนาภาษาเพื่อใช้ในการแปลงโมเดล 22
3.1.4 ส่วนแแสดงจินตทัศน์แบบจำลองผลลัพธ์ 29
บทที่ 4 การออกแบบและพัฒนาระบบ 30
4.1 ข้อกำหนดเบื้องต้นของระบบ 30
4.2 ความต้องการเชิงหน้าที่ (Functional Requirements) 30
4.2.1 แผนภาพยูสเคส 32
4.2.2 คำอธิบายยูสเคส 32
4.3 การออกแบบสถาปัตยกรรมระบบ 38
4.4 การพัฒนาระบบ 39
4.4.1 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา 39
4.4.2 การพัฒนาวิธีการสร้างเมทาโมเดล 40
4.4.3 การพัฒนาการแปลงรูปแบบโมเดลโดยภาษาเอทีแอล 42
4.4.4 การพัฒนาวิธีการสร้างแผนภาพลำดับ 52
4.4.5 การพัฒนาส่วนต่อประสาน 52
บทที่ 5 การทดสอบและประเมินผลระบบ 56
5.1 การทดสอบระบบ 56
5.2 การประเมินผลระบบ 60
5.2.1 ระบบการจัดการร้านหนังสือ 60
5.2.1.1 ยูสเคสการซื้อหนังสือ 62
5.2.1.2 ยูสเคสการยืมหนังสือ 63
5.2.1.3 ยูสเคสการคืนหนังสือ 64
5.2.1.4 ยูสเคสการสั่งหนังสือในประเทศ 66

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

หน้า

ผลการทดสอบของระบบ 71
บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ 83
6.1 สรุปผลการวิจัย 83
6.2 ข้อจำกัด 83
6.3 แนวทางการวิจัยต่อ 83
รายการอ้างอิง 84
ภาคผนวก 85
ภาคผนวก ก. 86
ภาคผนวก ข. 97
ภาคผนวก ค. 102
ภาคผนวก ง. 107
ประวัติผู้เขียนวิทยานิพนธ์ 115

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

สารบัญตาราง

ตารางที่ 1 ตัวอย่างโครงสร้างคำอธิบายยูสเคส 8
ตารางที่ 2 ข้อมูลที่สกัดจากเอกสารเอกซ์เอ็มไอ 19
ตารางที่ 3 การสร้างกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลต้นทางและปลายทาง 25
ตารางที่ 4 คำสั่งโปรแกรม sequence editor เปรียบเทียบกับเอกซ์เอ็มไอของแผนภาพลำดับ 29
ตารางที่ 5 ความต้องการเชิงหน้าที่ของระบบการสร้างแผนภาพลำดับ 31
ตารางที่ 6 คำอธิบายยูสเคส Import Use Case with Description 33
ตารางที่ 7 คำอธิบายยูสเคส Import Class 33
ตารางที่ 8 คำอธิบายยูสเคส Extract text with XPath 34
ตารางที่ 9 คำอธิบายยูสเคส Extract text from UCD using Natural Language 34
ตารางที่ 10 คำอธิบายยูสเคส Execute Source Metamodel 35
ตารางที่ 11 คำอธิบายยูสเคส Execute Target Metamodel 35
ตารางที่ 12 คำอธิบายยูสเคส Transform Metamodel by ATL 36
ตารางที่ 13 คำอธิบายยูสเคส Execute Mapping Rule 36
ตารางที่ 14 คำอธิบายยูสเคส Generate Sequence Diagram Visualization 37
ตารางที่ 15 คำอธิบายยูสเคส Export Sequence Diagram 38
ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล 47
ตารางที่ 17 การทดสอบการการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ 57
ตารางที่ 18 ทดสอบกฎการสร้างแผนภาพลำดับ 57
ตารางที่ 19 ทดสอบการส่งออกไฟล์ที่ได้จากการประมวลผล 59
ตารางที่ 20 คำอธิบายยูสเคสการซื้อหนังสือ 62
ตารางที่ 21 คำอธิบายยูสเคสการยืมหนังสือ 64
ตารางที่ 22 คำอธิบายยูสเคสการคืนหนังสือ 66
ตารางที่ 23 คำอธิบายยูสเคสการสั่งหนังสือภายในประเทศ 67
ตารางที่ 24 คำอธิบายยูสเคสการสั่งหนังสือจากต่างประเทศ 68

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

สารบัญรูป

รูปที่ 1 ลำดับชั้นในเอ็มโอเอฟ 4
รูปที่ 2 แนวคิดของเอทีแอล 6
รูปที่ 3 Metamodeling pattern in MDD 12
รูปที่ 4 แพทเทิร์นการแปลงแบบจำลองในการพัฒนาขับเคลื่อนด้วยแบบจำลอง 13
รูปที่ 5 โครงสร้างประโยคอย่างง่ายสำหรับเขียนคำอธิบายยูสเคส 14
รูปที่ 6 ภาพรวมของงานวิจัย 16
รูปที่ 7 ตัวอย่างเอกสารเอกซ์เอ็มไอของคำอธิบายยูสเคส 17
รูปที่ 8 ตัวอย่างเอกสารเอกซ์เอ็มไอของคำแผนภาพคลาส 17
รูปที่ 9 วิธีการสกัดข้อมูลจากข้อมูลนำเข้าคำอธิบายยูสเคส 18
รูปที่ 10 ประเภทของคำที่สามารถสกัดออกมาได้ [8] 22
รูปที่ 11 การแปลงแบบโมเดลโดยภาษาเอทีแอล 22
รูปที่ 12 เมทาโมเดลของภาษาเอทีแอล [5] 23
รูปที่ 13 เมทาโมเดลของคำอธิบายยูสเคส 23
รูปที่ 14 เมทาโมเดลของแผนภาพคลาส [2] 24
รูปที่ 15 เมทาโมเดลของแผนภาพลำดับ [2] 24
รูปที่ 16 แผนภาพยูสเคสของระบบการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ 32
รูปที่ 17 สถาปัตยกรรมการทำงานของระบบ 39
รูปที่ 18 การสร้างอีคอร์โมเดล 40
รูปที่ 19 การสร้างอีคอร์ไดอะแกรม 41
รูปที่ 20 ตัวอย่างการสร้างอีคอร์ไดอะแกรม 41
รูปที่ 21 หน้าจอของระบบการสร้างแผนภาพลำดับ 53
รูปที่ 22 หน้าจอการนำเข้าไฟล์เอกซ์เอ็มไอ 53
รูปที่ 23 หน้าจอแสดงผลกรณีการนำเข้าไฟล์ผิดพลาด 54
รูปที่ 24 หน้าจอแสดงผลกรณีพบข้อผิดพลาดในการประมวลผลของระบบ 54
รูปที่ 25 หน้าจอแสดงรายการแผนภาพลำดับ 55
รูปที่ 26 หน้าจอแสดงผลการทำงานที่ได้จากการแปลงแบบโมเดล 55
รูปที่ 27 แผนภาพยูสเคสของระบบการจัดการร้านหนังสือ 61
รูปที่ 28 แผนภาพคลาสของระบบการจัดการร้านหนังสือ 70
รูปที่ 29 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการซื้อหนังสือ 71
รูปที่ 30 ผลลัพธ์การประมวลผลด้วย ATL ของยูสเคสการซื้อหนังสือ 72
รูปที่ 31 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการซื้อหนังสือ 72
รูปที่ 32 การแสดงผลแผนภาพลำดับของยูสเคสการซื้อหนังสือ 73
รูปที่ 33 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการยืมหนังสือ 73
รูปที่ 34 ผลการประมวลผลด้วย ATL ในกรณีการยืมหนังสือ 74

รูปที่ 35 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการยืมหนังสือ 74
รูปที่ 36 การแสดงผลแผนภาพลำดับในกรณีการยืมหนังสือ 75
รูปที่ 37 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการคืนหนังสือ 76
รูปที่ 38 ผลการประมวลผลด้วย ATL ในกรณีการคืนหนังสือ 76
รูปที่ 39 ผลการประมวลผลขั้นตอนการแสดงผลในกรณีการคืนหนังสือ..... 77
รูปที่ 40 การแสดงผลแผนภาพลำดับของยูสเคสการคืนหนังส 77
รูปที่ 41 ผลการประมวลผลด้วย XPath และภาษาธรรมชาติในกรณีการสั่งหนังสือภายในประเทศ. 78
รูปที่ 42 ผลการประมวลผลด้วย ATL ในกรณีการสั่งหนังสือภายในประเทศ 79
รูปที่ 43 ผลการประมวลผลขั้นตอนการแสดงผลของยูสเคสการสั่งหนังสือภายในประเทศ 79
รูปที่ 44 การแสดงผลแผนภาพลำดับของยูสเคสการสั่งหนังสือภายในประเทศ..... 80
รูปที่ 45 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการสั่งหนังสือจาก
ต่างประเทศ..... 80
รูปที่ 46 ผลลัพธ์การประมวลผลด้วย ATL ของยูสเคสการสั่งหนังสือจากต่างประเทศ 81
รูปที่ 47 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการสั่งหนังสือจากต่างประเทศ..... 82
รูปที่ 48 การแสดงผลแผนภาพลำดับของยูสเคสการสั่งหนังสือจากต่างประเทศ 82
รูปที่ 49 เครื่องมือวิชวลพาราดาม เวอร์ชันสแตนดาร์ต 10.2 86
รูปที่ 50 หน้าจอการทำงานของเครื่องมือวิชวลพาราดาม 86
รูปที่ 51 การสร้างแผนภาพยูสเคสใหม่ 87
รูปที่ 52 การวาดส่วนประกอบ System..... 87
รูปที่ 53 การวาดส่วนประกอบผู้ใช้งานระบบ 88
รูปที่ 54 การวาดฟังก์ชันการทำงานของระบบ..... 88
รูปที่ 55 การวาดเส้นความสัมพันธ์ระหว่างผู้ใช้กับยูสเคส 88
รูปที่ 56 ตัวอย่างการวาดเส้นความสัมพันธ์แบบรวมในแผนภาพยูสเคส..... 89
รูปที่ 57 ตัวอย่างการวาดเส้นความสัมพันธ์แบบขยายในแผนภาพยูสเคส 89
รูปที่ 58 ตัวอย่างการวาดความสัมพันธ์แบบสืบทอดในแผนภาพยูสเคส..... 90
รูปที่ 59 วิธีการสร้างคำอธิบายจากแผนภาพยูสเคส 90
รูปที่ 60 หน้าจอแสดงปุ่มกดเพื่อทำการสร้างคำอธิบายยูสเคส 91
รูปที่ 61 หน้าจอการเลือกรูปแบบการกรอกคำอธิบายยูสเคส 91
รูปที่ 62 ตัวอย่างวิธีการสร้างเงื่อนไขในคำอธิบายยูสเคส..... 92
รูปที่ 63 ตัวอย่างการเขียนคำอธิบายยูสเคส 92
รูปที่ 64 การเลือกเมนู Export XMI..... 93
รูปที่ 65 ตัวอย่างหน้าจอการ Export XMI..... 93
รูปที่ 66 การสร้างแผนภาพคลาสใหม่..... 94
รูปที่ 67 การวาดแผนภาพคลาสไดอะแกรม 94
รูปที่ 68 การวาดคุณลักษณะภายในคลาส..... 95
รูปที่ 69 การตั้งชื่อตัวแปรและระบุประเภทตัวแปร 95

รูปที่ 70 การวาดเมธอดภายในคลาส 95
รูปที่ 71 การตั้งชื่อเมธอดและระบุพารามิเตอร์ 95
รูปที่ 72 ตัวอย่างการวาดความสัมพันธ์แบบสืบทอดในแผนภาพคลาส 96
รูปที่ 73 หน้าจอดาวน์โหลดปลั๊กอินสำหรับติดตั้ง UML2 97
รูปที่ 74 หน้าจอ All-In-One Update Site 98
รูปที่ 75 หน้าจอแสดงปุ่มดาวน์โหลดปลั๊กอิน UML2 98
รูปที่ 76 หน้าจอสำหรับติดตั้งปลั๊กอิน UML2 99
รูปที่ 77 หน้าจอสำหรับติดตั้งค่าปลั๊กอินให้อีคลิปส์ 99
รูปที่ 78 หน้าจอ Available Software 100
รูปที่ 79 หน้าจอแสดงรายละเอียดความสามารถของ UML2 Plugin ที่ติดตั้ง 101
รูปที่ 80 หน้าจอทดสอบการติดตั้งเครื่องมือเพื่อสนับสนุนการสร้างอีคอร์ 101
รูปที่ 81 หน้าจอสำหรับดาวน์โหลดเครื่องมือติดตั้ง ATL Plugin 102
รูปที่ 82 หน้าจอแสดงตำแหน่งสำหรับดาวน์โหลด ATL Plugin 102
รูปที่ 83 หน้าจอแสดงปุ่มดาวน์โหลด ATL Plugin 103
รูปที่ 84 หน้าจอสำหรับติดตั้ง ATL Plugin 103
รูปที่ 85 หน้าจอสำหรับติดตั้ง ATL Plugin ให้อีคลิปส์ 104
รูปที่ 86 หน้าจอแสดงส่วนเสริมทั้งหมดของ ATL 104
รูปที่ 87 หน้าจอแสดงรายละเอียดความสามารถของ ATL Plugin ที่ติดตั้ง 105
รูปที่ 88 หน้าจอทดสอบการติดตั้งเครื่องมือเพื่อสนับสนุนการพัฒนาภาษาเอทีแอล 106
รูปที่ 89 หน้าจอการทำงานหลักของระบบการแผนภาพลำดับ 107
รูปที่ 90 ส่วนหน้าจอการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ 108
รูปที่ 91 ส่วนหน้าจอการแสดงสถานะการทำงานของระบบ 108
รูปที่ 92 ส่วนแสดงผลรายการแผนภาพลำดับ 109
รูปที่ 93 แท็บทั้งหมดในหน้าจอทำงานหลัก 109
รูปที่ 94 หน้าจอนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ 110
รูปที่ 95 หน้าจอแสดงสถานะการทำงานของระบบทำงานเสร็จเรียบร้อย 110
รูปที่ 96 ตัวอย่างผลลัพธ์ในขั้นตอนการแปลงค่าการแสดงผลของระบบ 111
รูปที่ 97 ตัวอย่างการแสดงผลในโปรแกรม Quick Sequence Editor 111
รูปที่ 98 วิธีการส่งออกไฟล์ในรูปแบบไฟล์รูปภาพ 112
รูปที่ 99 วิธีการบันทึกไฟล์รูปภาพ 112
รูปที่ 100 หน้าจอแสดงเครื่องหมาย “?” บนเส้นเมสเสจ 113
รูปที่ 101 หน้าจอการนำเคอร์เซอร์ไปชี้ที่ตำแหน่งที่ผิดพลาด 113
รูปที่ 102 หน้าจอแสดงปุ่ม save เมื่อมีการแก้ไขข้อมูลนำเข้า 114
รูปที่ 103 หน้าจอแสดงผลแผนภาพลำดับที่ถูกแก้ไขสำเร็จ 114

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ระหว่างขั้นตอนการพัฒนาซอฟต์แวร์ แผนภาพยูเอ็มแอล (UML Diagrams) มีความสำคัญเพราะสามารถใช้เป็นสัญกรณ์ (notations) อันเป็นหนึ่งเดียวกันที่ใช้อธิบายการออกแบบเชิงวัตถุในมุมมองต่างๆก่อนนำไปเขียนซอฟต์แวร์ และใช้สื่อสารความเข้าใจระหว่างทีมงานต่างๆ ตัวอย่างเช่น 1) ยูสเคส (Use Case) เป็นแบบจำลองเชิงตรรกะที่อธิบายฟังก์ชันการทำงานหลักของระบบ รวมถึงผู้ใช้และผู้ที่มีความเกี่ยวข้องกับระบบ โดยยังไม่มีการระบุว่ากิจกรรมต่างๆ ของระบบนั้นจะถูกพัฒนาขึ้นอย่างไร คำอธิบายยูสเคส (Use Case Description) เป็นเอกสารเชิงบรรยายประกอบการใช้งานแผนภาพยูสเคส ซึ่งจะอธิบายรายละเอียดว่าผู้ใช้จะสามารถทำอะไรได้บ้าง และระบบจะมีการตอบสนองการทำงานอย่างไร 2) แผนภาพคลาส(Class Diagram) เป็นแผนภาพที่ใช้สำหรับอธิบายโครงสร้างพื้นฐานของออบเจ็กต์และอธิบายความสัมพันธ์ระหว่างออบเจ็กต์ในภาพรวมทั้งหมดของระบบ และ 3) แผนภาพลำดับ (Sequence Diagram) เป็นแผนภาพที่อธิบายพฤติกรรมและขั้นตอนการทำงานของแต่ละยูสเคส โดยแสดงการโต้ตอบระหว่างอ็อบเจ็กต์ ซึ่งจะทำให้ผู้พัฒนาระบบมองเห็นภาพความสัมพันธ์ระหว่างอ็อบเจ็กต์ และเห็นลำดับเวลาของการเกิดก่อน-หลังได้อีกด้วย

อย่างไรก็ตาม การเขียนแผนภาพยูเอ็มแอลนับว่าเป็นกิจกรรมที่สิ้นเปลืองทรัพยากร รวมทั้งอาจเขียนขึ้นภายหลังการพัฒนาซอฟต์แวร์เพื่อจุดประสงค์สำหรับการจัดทำเอกสารในการบำรุงรักษาระบบหรือส่งมอบให้กับลูกค้า ซึ่งขั้นตอนการทำงานดังกล่าวเป็นลำดับที่ไม่ถูกต้องและอาจก่อให้เกิดข้อบกพร่อง (defect) ดังนั้น ถ้ามีเครื่องมือช่วยสนับสนุนให้ผู้วิเคราะห์ระบบสามารถสร้างแผนภาพยูเอ็มแอลได้อย่างรวดเร็วและถูกต้อง จะเป็นการส่งเสริมคุณภาพกระบวนการพัฒนาซอฟต์แวร์ขององค์กร

งานวิจัยนี้ได้ทำการศึกษาเอทีแอล (ATL: Atlas Transformation Language) ซึ่งเป็นหนึ่งในภาษาที่มีความสามารถจัดการแปลงรูปจากแบบจำลอง (model) หนึ่งไปเป็นอีกแบบจำลองหนึ่งได้ในระดับเมทาเดตา (metadata) จากความสามารถของเอทีแอลดังกล่าว ผู้วิจัยจึงมีแนวคิดการวิจัยในการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ (Model Transformation Technology) จากแหล่งข้อมูลต้นทาง คือ คำอธิบายยูสเคส และแผนภาพคลาส ทั้งนี้ ผู้วิจัยจะทำการพัฒนาระบบซอฟต์แวร์จากวิธีการที่นำเสนอ ซึ่งซอฟต์แวร์หรือเครื่องมือที่สร้างขึ้นนี้จะช่วยการปรับปรุงกระบวนการซอฟต์แวร์ให้มีประสิทธิภาพมากขึ้น กล่าวคือ ลดระยะเวลาและค่าใช้จ่ายในการสร้างเอกสารการออกแบบระบบ รวมทั้งลดข้อบกพร่อง ที่อาจเกิดขึ้นจากความไม่ถูกต้องตรงกันของเอกสารการวิเคราะห์และออกแบบระบบซอฟต์แวร์

1.2 วัตถุประสงค์ของการวิจัย

เพื่อนำเสนอวิธีการและพัฒนาเครื่องมือสำหรับสร้างแผนภาพลำดับจากคำอธิบายยูสเคสและแผนภาพคลาสโดยการใช้เทคโนโลยีการแปลงแบบจำลอง เพื่อปรับปรุงกระบวนการซอฟต์แวร์ให้มีประสิทธิภาพมากขึ้น และลดข้อบกพร่องของความไม่ถูกต้องตรงกันของเอกสารการวิเคราะห์และออกแบบระบบ

1.3 ขอบเขตของการวิจัย

1. งานวิจัยนี้ใช้เอทีแอลเป็นเทคโนโลยีการแปลงแบบจำลอง
2. เครื่องมือที่ใช้ประกอบในการทำวิจัย คือ Visual Paradigm for UML 10.2 สำหรับสร้างคำอธิบายยูสเคสและแผนภาพคลาส
3. งานวิจัยนี้สนับสนุนเฉพาะข้อมูลนำเข้าเป็นภาษาอังกฤษเท่านั้น
4. โครงสร้างภาษาที่ใช้ในประกอบคำอธิบายรายละเอียดแผนภาพยูสเคสต้องเป็นไปตามโครงสร้างไวยากรณ์ที่กำหนด

1.4 ขั้นตอนและวิธีดำเนินการวิจัย

1. ศึกษาและทำความเข้าใจเกี่ยวกับโครงสร้าง คำสั่ง และการทำงานของเอทีแอล
2. ศึกษาและทำความเข้าใจเทคโนโลยี XPath และภาษาธรรมชาติ
3. ศึกษาและทำความเข้าใจเกี่ยวกับทฤษฎีและงานวิจัยที่เกี่ยวข้อง
4. ศึกษาวิธีการใช้เครื่องมือ Visual Paradigm for UML 10.2
5. ศึกษาวิเคราะห์และกำหนดระเบียบวิธีวิจัย
6. ออกแบบและตั้งสมมติฐาน ที่เกี่ยวข้องกับงานวิจัย
7. พัฒนาระบบเพื่อใช้สำหรับการสร้างแผนภาพลำดับจากคำอธิบายยูสเคสและแผนภาพคลาส
8. ทดสอบ และนำผลที่ได้ไปปรับปรุงระบบเพื่อให้ได้วัตถุประสงค์ที่กำหนด
9. ตีพิมพ์ผลงานทางวิชาการ
10. สรุปผลการวิจัย และจัดทำวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้แนวทางและระบบต้นแบบสำหรับสร้างแผนภาพลำดับแบบอัตโนมัติ
2. ได้เครื่องมือสนับสนุนการปรับปรุงกระบวนการซอฟต์แวร์ ซึ่งช่วยลดทรัพยากรเวลาและค่าใช้จ่ายของกระบวนการพัฒนาซอฟต์แวร์ รวมทั้งลดข้อบกพร่องของความไม่ตรงกันระหว่างเอกสารการวิเคราะห์และออกแบบระบบ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

1.6 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์

วิทยานิพนธ์นี้แบ่งเนื้อหาออกเป็น 6 บท ดังต่อไปนี้ บทที่ 1 เป็นบทนำกล่าวถึงที่มาและความสำคัญของปัญหา วัตถุประสงค์ ขอบเขต และประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย บทที่ 2 กล่าวถึงกล่าวถึงทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้องกับงานวิจัยนี้ บทที่ 3 อธิบายถึงแนวทางในการสร้างแผนภาพลำดับจากคำอธิบายยูสเคสและแผนภาพคลาส บทที่ 4 อธิบายการออกแบบและพัฒนาระบบตามแนวทางการวิจัย บทที่ 5 กล่าวถึงวิธีการประเมินและวัดผลการทดลอง และบทที่ 6 กล่าวถึงสรุปผลการวิจัย ข้อเสนอแนะและแนวทางการพัฒนาต่อในอนาคต

1.7 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตีพิมพ์เป็นบทความวิชาการเรื่อง “Sequence Diagram Generation with Model Transformation Technology” โดย พจนา แซวประโคน และ ญาใจ ลิ่มปิยะกรณ์ ในรายงานสืบเนื่องการประชุมวิชาการระดับนานาชาติ The 9th International MultiConference of Engineers and Computer Scientists (IMECS 2014) ฮ่องกง ระหว่างวันที่ 12-14 มีนาคม 2557 หน้า 584-589 และ “Model-Driven Approach to Constructing UML Sequence Diagram” โดย พจนา แซวประโคน และ ญาใจ ลิ่มปิยะกรณ์ ในรายงานสืบเนื่องการประชุมวิชาการระดับนานาชาติ The 5th International Conference on Information Science & Applications (ICISA 2014) สาธารณรัฐเกาหลี ระหว่างวันที่ 6-9 พฤษภาคม 2557 หน้า 504-507

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

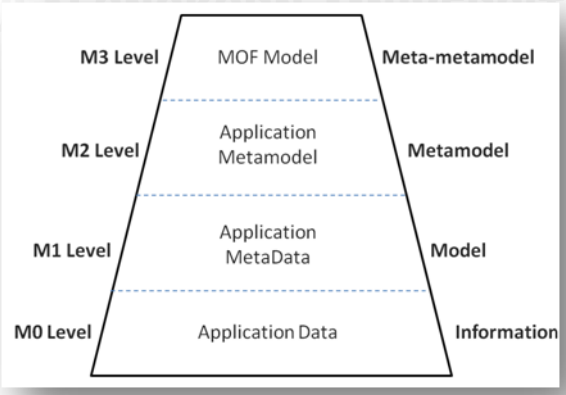
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1ทฤษฎีที่เกี่ยวข้อง

2.1.1 การพัฒนาการขับเคลื่อนด้วยแบบจำลอง (Model Driven Development : MDD) [1]

การพัฒนาการขับเคลื่อนด้วยแบบจำลองเป็นรูปแบบการพัฒนาซอฟต์แวร์ที่กำหนดขอบข่ายของปัญหาให้อยู่ในรูปแบบของโมเดลและจะมุ่งเน้นไปที่การสร้างและการใช้ประโยชน์จากโมเดลในรูปแบบของการนิยามมากกว่าการแก้ปัญหาแบบอัลกอริทึม นอกจากนี้การพัฒนาการขับเคลื่อนด้วยแบบจำลองจะมุ่งเน้นไปที่การอธิบายกระบวนการแก้ปัญหาเป็นลำดับชั้น และสนับสนุนการพัฒนาโปรแกรมแบบออโตเมติก รูปแบบการพัฒนาที่อ้างอิงวิธีการพัฒนานี้ ที่รู้จักกันอย่างแพร่หลาย ได้แก่ สถาปัตยกรรมการขับเคลื่อนด้วยแบบจำลอง (Model Driven Architect : MDA) และอีคลิปโมเดลลิงค์เฟรมเวิร์ค (Eclipse Modeling Framework : EMF) ซึ่งเป็นโอเพนซอร์สที่ถูกพัฒนาภายใต้อีคลิปโมเดลลิงค์โพรเจกต์ (Eclipse Modeling Project : EMP)

2.1.1.1 สถาปัตยกรรมการขับเคลื่อนด้วยแบบจำลอง (Model Driven Architecture: MDA) เป็นรูปแบบและวิธีในการพัฒนาซอฟตแวร์ตามแนวคิดการพัฒนาการขับเคลื่อนด้วยแบบจำลอง ซึ่งถูกนิยามโดยโอเอ็มจี (Object Management Group: OMG) [2] กล่าวได้ว่าสถาปัตยกรรมการขับเคลื่อนด้วยแบบจำลองเป็นรูปแบบการออกแบบระบบโดยการอธิบายโครงสร้างกระบวนการทำงาน และความเกี่ยวข้องทั้งหมดของระบบในรูปแบบของโมเดล อีกทั้งยังเป็นแนวคิดที่พยายามแยกส่วนของการวิเคราะห์ระบบให้ชัดเจนจากออกแบบ การวิเคราะห์ระบบจะเจาะจงไปที่ความต้องการเชิงหน้าที่ (Functional requirements) ในขณะที่การออกแบบจะมุ่งเน้นไปที่ส่วนโครงสร้างของระบบซึ่งเป็นความต้องการไม่ใช่เชิงหน้าที่ (Non-functional requirements) การพัฒนาซอฟต์แวร์ตามแนวคิดนี้เกี่ยวข้องกับมาตรฐานหลายอย่างที่รู้จักกันดี คือ Meta-Object Facility หรือเรียก เอ็มโอเอฟ (MOF) [3] ซึ่งนับว่าเป็นหัวใจสำคัญที่สุดของแนวคิดนี้



รูปที่ 1 ลำดับชั้นในเอ็มโอเอฟ

จากรูปที่ 1 เอ็มโอเอฟเป็นมาตรฐานที่กำหนดขึ้นโดยโอเอ็มจี จุดประสงค์เพื่อพัฒนาระบบด้วยแนวคิดการพัฒนาการขับเคลื่อนด้วยแบบจำลองโดยกำหนดให้แบ่งโมเดลออกเป็นลำดับชั้น 4 ลำดับที่แตกต่างกัน ได้แก่ ชั้นบนสุด เรียกว่า เมทาเมทาโมเดล (Meta-meta Model) หรือ M3 โดยที่แบบจำลองของชั้นนี้จะเป็นภาษาโปรแกรมซึ่งจะถูกใช้ในการสร้างเมทาโมเดล (Metamodel) ในลำดับชั้นล่างซึ่งเรียกว่า M2 ตัวอย่างของโมเดล M2 ที่เป็นได้ชัดที่สุดของชั้นนี้ก็คือ UML Metamodel ซึ่งเป็นโมเดลที่อธิบายตัวเองได้ โมเดล M2 ทำหน้าที่อธิบายสมาชิกของโมเดล (Model) หรือ M1 ที่อยู่ชั้นล่างถัดไป ตัวอย่างของโมเดลชั้น M1 ได้แก่ โมเดลที่ถูกสร้างอยู่ภายในยูเอ็มแอล, ชั้นสุดท้ายของเอ็มโอเอฟ เรียกว่า M0 เป็นชั้นของข้อมูล อันเป็นตัวแทนของวัตถุที่มีอยู่จริง

2.1.1.2 อีคลิปโมเดลลิ่งเฟรมเวิร์ค (Eclipse Modeling Framework : EMF) [4] หรือเรียกว่า อีเอ็มเอฟ เป็นรูปแบบโครงสร้างเพื่อการพัฒนาระบบด้วยโมเดล ภายใต้แนวคิดการพัฒนาการขับเคลื่อนด้วยแบบจำลองอีเอ็มเอฟถูกพัฒนาให้มีองค์ประกอบที่ช่วยเหลือนักพัฒนาในหลายส่วน เช่น เป็นเครื่องมือที่ช่วยในการพัฒนาโปแกรม เริ่มจากแปลงโมเดลให้อยู่ในรูปแบบมาตรฐานการแลกเปลี่ยนข้อมูลกลางในรูปแบบไฟล์เอกซ์เอ็มไอ (xmi) อีเอ็มเอฟได้จัดเตรียมเครื่องมือและการประมวลผลที่ช่วยสนับสนุนการสร้างโค้ดภาษาจาวาจากโมเดลที่ได้นิยามไว้ แล้วสร้างชุดของคลาสที่เป็นส่วนประกอบของโมเดลเพื่อเป็นเครื่องมือพื้นฐานสำหรับการแก้ไขและแสดงโมเดล. โดยโมเดลสามารถนิยามได้หลายรูปแบบอันได้แก่ จาวา ยูเอ็มแอล เอกซ์เอ็มแอล และเอกซ์เอ็มไอ ซึ่งล้วนแต่สามารถนำเข้าสู่อีเอ็มเอฟได้ หัวใจสำคัญที่สุดของอีเอ็มเอฟ เรียกว่า อีคอร์

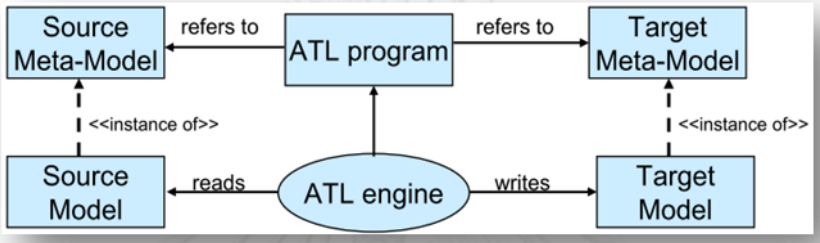
อีคอร์เป็นรูปแบบการนิยามลำดับชั้นของโมเดลในการออกแบบและพัฒนาระบบ โดยยึดแนวคิดของเอสเซนเชียลเอ็มโอเอฟ (Essential MOF : EMOF) ซึ่งจะสืบทอดคุณสมบัติมาอีกครั้ง สำหรับอีคอร์จะมีลักษณะแตกต่างจากเอ็มโอเอฟตรงที่จะแบ่งโมเดลออกเป็นแค่สามลำดับชั้น เนื่องจากลำดับชั้นสุดท้าย M3 จะอ้างอิงถึงตัวเอง การบรรยายคุณลักษณะของโมเดลด้วยอีคอร์ หรือที่เรียกว่า เมทาโมเดล จะถูกแสดงให้อยู่ในรูปคลาส ซึ่งมีสมาชิกย่อยอยู่ข้างในหลายส่วน ได้แก่

- EClass : แสดงแม่แบบของวัตถุซึ่งบรรจุ คุณลักษณะ และความสัมพันธ์กับแม่แบบอื่น ๆ
- EAttibute : แสดงคุณลักษณะอันประกอบด้วย ชื่อคุณลักษณะ และชนิดของคุณลักษณะ
- EReference : แสดงความสัมพันธ์ระหว่างแม่แบบ โดยระบุชนิดความสัมพันธ์ เช่น ความเป็นแม่ลูก, ความสัมพันธ์แบบส่วนประกอบ และอธิบายลักษณะหรือข้อกำหนดของส่วนประกอบ
- EDataType : อธิบายชนิดของคุณลักษณะ

การแปลงรูปแบบโมเดล (Model Transformation) เป็นนิยามการออกแบบและพัฒนาระบบซึ่งมองทุกอย่างเป็นแบบจำลอง ซึ่งจะมองว่าปัญหาต่าง ๆ คือแบบจำลองและแบบจำลองก็สามารถถูกปรับเปลี่ยนหรือแก้ไขโดยกระบวนการแปลงรูป (Transformation) ได้ ดังนั้นโอเอ็มจีได้กำหนดภาษามาตรฐานขึ้นมาสำหรับการแปลงรูปแบบโมเดล เรียกว่า ภาษาคิววีที (QVT Language) อย่างไรก็ดีภาษาดังกล่าวไม่ค่อยได้รับความนิยมมากนัก ประกอบกับได้มีการพัฒนาภาษาเอทีแอลขึ้นมาทีหลังเพื่อต่อยอดความสามารถที่ไม่มีจากการใช้งานในภาษาคิววีที

2.1.2 เอทีแอล (Atlas Transformation Language : ATL) [5]

เอทีแอลพัฒนาโดย AtlanMod Team ประเทศฝรั่งเศส ซึ่งภาษานี้เป็นภาษาโปรแกรมมิ่งสำหรับเปลี่ยนรูปแบบโมเดล จากกลุ่มของโมเดลต้นทางไปเป็นกลุ่มของโมเดลปลายทาง โดยอาศัยโครงร่างรูปแบบของข้อมูลที่เรียกว่า Metamodel เป็นเงื่อนไขในการเปลี่ยนรูปร่างโมเดล ภาษา ATL เป็นหนึ่งในแนวคิดของการพัฒนาซอฟต์แวร์ที่ยึดโมเดลเป็นตัวขับเคลื่อน ที่เรียกว่า Model Driven Engineering (MDE) และแนวคิดของภาษาเอทีแอลจะแสดงดังรูปที่ 2



รูปที่ 2 แนวคิดของเอทีแอล

รูปที่ 2 นำเสนอแนวคิดและหลักการทำงานของเอทีแอล โดยข้อมูลนำเข้า คือ แบบจำลองต้นทาง (Source Model) ส่วนที่ต้องสร้างเป็นกฎ (Rule) อธิบายองค์ประกอบของแบบจำลองต้นทางจะบรรจุอยู่ในเมทาโมเดลต้นทาง (Source Metamodel) ส่วนที่ควบคุมการทำงานการแปลงค่าระหว่าง 2 เมทาโมเดล คือ ATL Program ส่วนที่ต้องสร้างเป็นกฎอธิบายองค์ประกอบของแบบจำลองเป้าหมายจะบรรจุอยู่ในเมทาโมเดลเป้าหมาย คือ Target Metamodel ส่วนที่เป็นผลลัพธ์นำออกจะเป็นแบบจำลองเป้าหมาย คือ Target Model และส่วนที่เป็นตัวขับเคลื่อนให้ ATL Program ทำงานได้ คือ ATL Engine ซึ่งจะอ่านแบบจำลองต้นทางเข้ามาประมวลผลด้วย ATL Program ตัวอย่างที่ 1 แสดงโครงสร้างคำสั่งคำสั่งของภาษาเอทีแอล

```
rule Member2Male {                                                    →1
        from                                                          →2
            s : Families!Member (not s.isFemale())                    →3
        to                                                            →4
            t : Persons!Female (fullName <- s.firstName +' '+ s.familyName)   →5
}
```

ตัวอย่างการสร้าง Rule

บรรทัดที่ 1: “rule” เป็นคำสงวนเพื่อให้ตัวแปรภาษารู้ว่านี่คือกฎ และตามด้วยชื่อของกฎที่ต้องการ

บรรทัดที่ 2: “from”เป็นคำสงวนเพื่อบอกถึงหน่วยของข้อมูลต้นทางที่จะถูกนำมาใช้เพื่อเปลี่ยนรูปร่าง

บรรทัดที่ 3: “s : Families!Member” หมายถึงต้องการเข้าถึงข้อมูลต้นทางที่คลาส Member ซึ่งอยู่ใน package Families โดยกำหนดตัวแปรที่จะใช้แทนค่าดังกล่าวชื่อ s และเพิ่มเงื่อนไขให้จำกัดข้อมูลเฉพาะที่เงื่อนไข (s.isFemale()) เป็นจริงเท่านั้น

บรรทัดที่ 4: “to” เป็นคำสงวนเพื่อบอกถึงหน่วยของข้อมูลปลางทางที่จะถูกสร้างขึ้น

บรรทัดที่ 5: “t : Persons!Female” หมายถึงต้องการสร้างข้อมูลปลายทางที่คลาส Female ซึ่งอยู่ใน package Persons โดยกำหนดตัวแปรที่จะใช้แทนค่าดังกล่าวชื่อ t

บรรทัดที่ 6: “fullName <- s.firstName + '' + s.familyName” เป็นการสร้างข้อมูลปลายทางที่ตัวแปร “fullName” (คลาส Female) ให้มีค่าเท่ากับ ข้อมูลต้นทางที่ตัวแปร “firstName” รวมกับตัวแปร “familyName” (คลาส Member)

จากตัวอย่างที่ 1 โครงสร้างคำสั่งของภาษาเอทีแอล กฎของเอทีแอลจะกำหนดรายละเอียดองค์ประกอบการสร้างแบบจำลองเป้าหมาย โดยเริ่มต้นจากแต่ละองค์ประกอบที่ตรงกันในแบบจำลองต้นทาง ตัวอย่างที่ 2 แสดงโครงสร้างคำสั่งเอทีแอลที่มีการเรียกใช้ ATL helper จากตัวอย่างที่นำเสนอ คือ คำสั่ง isFemale()

```
helper context Families!Member def: isFemale() : Boolean =
    if not self.familyMother.oclIsUndefined() then
            true
    else
          if not self.familyDaughter.oclIsUndefined() then
                  true
          else
                  false
```

ตัวอย่างการสร้าง helper

บรรทัดที่ 1: ขึ้นต้นด้วยคำสงวน “helper context” เพื่อให้รู้ว่านี่คือการเริ่มสร้างกลุ่มของคำสั่งตามด้วย “Families!Member” เพื่อระบุตัวแปรที่จะถูกส่งเข้ามา, “def” เป็นคำสงวนเพื่อเริ่มนิยามชื่อของกลุ่มคำสั่งเหล่านี้ซึ่งจะเป็นชื่อจะถูกเรียกใช้จากที่อื่น, “Boolean” เป็นชนิดของผลลัพธ์ที่ได้จากการทำงานของกลุ่มคำสั่งนี้นั่นเอง

บรรทัดที่2 : เปลี่ยนกลุ่มคำสั่งแสดงการตัดสินใจโดยเงื่อนไขคือ “**not self**.familyMother.oclIsUndefined()” ถ้าสมาชิกมีความสัมพันธ์แบบ familyMother (ระบุใน metamodel) เงื่อนไขนี้เป็นจริง

บรรทัดที่ 5: เปลี่ยนกลุ่มคำสั่งแสดงการตัดสินใจโดยเงื่อนไขคือ “**not self.**familyDaughter.oclIsUndefined()” ถ้าสมาชิกมีความสัมพันธ์แบบ familyDaughter (ระบุใน metamodel) เงื่อนไขนี้เป็นจริง

ตัวอย่างที่ 2 การเขียนโค้ด ATL Helper จะถูกเรียกใช้จากส่วนที่เป็น ATL Rule สังเกตได้ว่า ATL helper เปรียบได้กับ method() ของการโปรแกรมเชิงวัตถุ

2.1.3 เอกซ์เอ็มไอ (XML Metadata Interchange : XMI) [6]

เอกซ์เอ็มไอเป็นรูปแบบข้อมูลของแผนภาพที่ใช้ในการในการแลกเปลี่ยนสำหรับแผนภาพยูเอ็มแอล เอกซ์เอ็มไอได้นิยามมุมมองที่เกี่ยวข้องกับวัตถุในเอกซ์เอ็มแอล (XML) ดังนี้

1. เป็นตัวแทนของของวัตถุในแง่ขององค์ประกอบของเอกซ์เอ็มแอลและแอตทริบิวต์ (Attribute) เป็นพื้นฐาน
2. เนื่องจากการเชื่อมต่อกันของวัตถุ XMI จึงได้รวมมาตรฐานในการเชื่องโยงวัตถุภายในไฟล์เดียวกันหรือข้ามไฟล์
3. การระบุวัตถุสามารถอ้างอิงได้จากวัตถุอื่นในรูปแบบ ของ IDs และ UUIDs.
4. การตรวจสอบเอกซ์เอ็มไอโดยใช้รูปแบบของเอกซ์เอ็มแอล (XML Schema)

เอกซ์เอ็มไอสามารถตรวจสอบได้โดยการระบุ EBNF product rule ในการสร้างเอกสารเอกซ์เอ็มแอลและ schema ซึ่งสามารถแลกเปลี่ยนวัตถุได้อย่างมีประสิทธิภาพ

2.1.4 คำอธิบายยูสเคส (Use Case Description)

ตารางที่ 1 ตัวอย่างโครงสร้างคำอธิบายยูสเคส

ID:	Use case type:	Importance level:
Use Case Name:		
Primary actor:		
Stakeholders and interests:		
Brief description:		
Trigger:		
Type:		
Relationships:		
Association:		
Include:		
Extend:		
Generalization:		
Normal flow of Events:		
Subflows:		
Alternate/Exceptional flows:		

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

คำอธิบายยูสเคสเป็นเอกสารเชิงบรรยายอธิบายถึงฟังก์ชันการทำงานพื้นฐานของระบบ ซึ่งจะอธิบายว่าผู้ใช้จะสามารถทำอะไรได้บ้าง และระบบจะมีการตอบสนองการทำงานอย่างไร คำอธิบายยูสเคสประกอบไปด้วยโครงสร้างข้อมูลต่างๆ ดังแสดงในตารางที่ 1 ประกอบด้วยส่วนต่างๆ ดังนี้

- Use Case Name: ชื่อยูสเคส โดยชื่อของยูสเคสควรเป็นกลุ่มคำกริยา เช่น ตรวจสอบสิทธิการเข้าใช้ระบบ
- ID: รหัสยูสเคส การกำหนดรหัสยูสเคส จะเป็นการกำหนดลักษณะเฉพาะของยูสเคส ที่ง่ายต่อการอ้างอิง รวมถึงสามารถตามรอยกลับไปยังความต้องการที่เกี่ยวข้องกันได้
- Importance level: ระดับความสำคัญของยูสเคส เมื่อเปรียบเทียบกับยูสเคสอื่นในระบบ มีไว้เพื่อช่วยในการจัดลำดับความสำคัญเร่งด่วนของงานที่ต้องทำ เช่น สูง/ปานกลาง/ต่ำ
- Primary actor: ระบุชื่อบุคคล หรือระบบที่จะก่อให้เกิดการดำเนินงานตามยูสเคส
- Use case type: ประเภทของยูสเคส โดยระบุว่า “ภาพรวม” หรือ “เชิงละเอียด” (Overview/Detail) และ “พื้นฐาน” หรือ “จริง” (Essential/Real)
- Stakeholders and interests: เป็นข้อความแสดงชื่อบุคคล หรือระบบที่ได้รับผลประโยชน์จากการทำงานของยูสเคส ได้แก่ ผู้กระทำหลัก หรือ อาจเป็นผู้ที่เกี่ยวข้องกับยูสเคส
- Brief description: เป็นข้อความสาระสำคัญของยูสเคส ซึ่งโดยทั่วไปจะเป็นการอธิบายสั้นๆ เพื่อให้เข้าใจว่ายูสเคสนี้ให้บริการใด แก่ผู้ใดบ้าง และจะมีการใช้งานเมื่อใด
- Trigger: เป็นข้อความแสดงเหตุการณ์ที่ก่อให้เกิดการดำเนินงานตามยูสเคส สามารถจำแนกได้เป็น 2 ประเภท คือ
    1. External Trigger : คือ สิ่งกระตุ้นภายนอก เช่น ลูกค้ามีความต้องการสั่งซื้อสินค้า
    2. Temporal Trigger : คือ สิ่งกระตุ้นตามเวลา เช่น เมื่อเลยกำหนดระยะเวลาการยืมหนังสือ ระบบจะต้องจัดทำระบบเตือนแก่ผู้ยืม และเจ้าหน้าที่ห้องสมุด เป็นต้น
- Relationships: ระบุความสัมพันธ์ว่า ยูสเคสมีความเกี่ยวข้องกับผู้กระทำหลัก ผู้ใช้ และยูสเคสอื่นๆอย่างไร แบ่งออกเป็น 4 รูปแบบ คือ
    1. Association: เป็นการระบุความสัมพันธ์ระหว่างยูสเคส และผู้กระทำแบบเกี่ยวเนื่อง ในที่นี้จะหมายถึงบทบาทของผู้ใช้เมื่อใช้ระบบ
    2. Include: เป็นการระบุความสัมพันธ์แบบรวมระหว่างยูสเคสว่า ยูสเคสใดมีฟังก์ชันการทำงานที่จะถูกรวมเข้าไว้กับยูสเคสนั้นๆ เสมอ
    3. Extend: เป็นการระบุความสัมพันธ์แบบขยายระหว่างยูสเคสว่า ยูสเคสใดมีฟังก์ชันการทำงานที่สามารถนำมารวมเข้าไว้เป็นทางเลือก (Optional) กับยูสเคสนั้นๆ

4. Generalization: เป็นการระบุความสัมพันธ์แบบความเป็นทั่วไปของคุณสมบัติความสัมพันธ์ระหว่างยูสเคสกับยูสเคสที่มีการสืบทอดคุณสมบัติ (Inheritance) จากยูสเคสพ่อแม่

- Normal flow of events: เป็นข้อความแสดงลำดับขั้นตอนการทำงานของแต่ละยูสเคส สามารถจำแนกได้เป็น 3 ประเภท ดังนี้

1. Normal Flow : เป็นข้อความแสดงขั้นตอนการทำงานปกติของยูสเคส
2. Subflows: เป็นข้อความแสดงขั้นตอนการทำงานที่แตกแยกย่อยลงมาจากขั้นตอนการทำงานปกติของยูสเคส เพื่อไม่ให้ส่วนของขั้นตอนปกติของยูสเคสซับซ้อนจนเกินไป
3. Alternate/Exceptional flows: เป็นข้อความแสดงเงื่อนไขที่ทำให้เกิดเหตุการณ์อื่นที่อาจเกิดขึ้นได้ในบางกรณี

เนื่องจากในงานวิจัยนี้ได้ใช้เครื่องมือวิชวลพาราดามสำหรับสร้างข้อมูลนำเข้า ดังนั้น ข้อมูลนำเข้าในส่วนที่เป็น Normal flow of events จะอยู่ในแท็ปชื่อว่า Flow of events

2.1.5 แผนภาพคลาส (Class Diagram)

แผนภาพคลาสเป็นแผนภาพที่แสดงถึงกลุ่มของออบเจ็กต์ที่มีโครงสร้างพื้นฐานและพฤติกรรมเดียวกัน ดังนั้นออบเจ็กต์ที่มีคุณสมบัติลักษณะเดียวกันก็จะรวมกลุ่มอยู่ในคลาสเดียวกัน นอกจากนี้แล้วแผนภาพคลาสเป็นแผนภาพที่แสดงถึงความสัมพันธ์ระหว่างออบเจ็กต์ในภาพรวมทั้งหมดของระบบ

ส่วนประกอบของคลาสสามารถแบ่งออกได้ 3 ส่วนคือ ประกอบด้วยส่วนต่างๆ ดังนี้

- ชื่อคลาส (Class name) โดยทุกๆ คลาสจะต้องมีการกำหนดชื่อคลาส
- แอตทริบิวต์ (Attribute) เป็นการระบุคุณสมบัติต่างๆ ของคลาส โดยที่แต่ละคลาสอาจจะมีแอตทริบิวต์หรือไม่มีก็ได้ และ แอตทริบิวต์ทุกตัวในคลาสจะต้องถูกกำกับด้วยระดับของการเข้าถึง (Visibility) ซึ่งสามารถแบ่งได้ 3 ระดับ คือ

1. Public หมายถึง แอตทริบิวต์ที่จะยอมให้คลาสอื่นเข้าถึงและเรียกใช้งานได้ ซึ่งจะมีสัญลักษณ์ + อยู่ข้างหน้า
2. Private หมายถึง แอตทริบิวต์ที่จะสามารถเรียกใช้หรืออ้างถึงได้เพียงคลาสเดียวเท่านั้น และไม่สามารถอ้างถึงหรือเรียกใช้จากคลาสลูกได้ ซึ่งจะมีสัญลักษณ์ - อยู่ข้างหน้า
3. Protected หมายถึง แอตทริบิวต์ที่จะสามารถเรียกใช้หรืออ้างถึงได้เพียงคลาสเดียวแต่อนุญาตให้คลาสลูกสามารถใช้งานได้ด้วย ซึ่งจะมีสัญลักษณ์ # อยู่ข้างหน้า

- โอเปอเรชัน (Operation) หรือเมธอด (Method) เป็นการกำหนดฟังก์ชันหรือขั้นตอนซึ่งจะถูกกระทำจากออบเจ็กต์ที่เป็นสมาชิกของคลาสนั้นๆ
- เส้นความสัมพันธ์ (Relationship) ที่ระบุความสัมพันธ์ระหว่างคลาสแบ่งออกเป็น 4 รูปแบบ คือ

    1. Association : เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบ 2 ทิศทาง เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบ “Whole-Part” หรือ “is part of” โดยจะมี Class ที่ใหญ่ที่สุดที่เป็น Object หลักและมี Class อื่นเป็นส่วนประกอบ
    2. Generalization : เป็นความสัมพันธ์ระหว่าง Object หรือ Class ในลักษณะของการสืบทอดคุณสมบัติจาก Class หนึ่ง (Superclass) ไปยังอีก Class หนึ่ง (Subclass)
    3. Aggregation : เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบขึ้นต่อกันและมีความเกี่ยวข้องกันเสมอ โดยจะมี Class ซึ่งเป็นองค์ประกอบของ Class อื่นที่ใหญ่กว่า และเมื่อ Class ที่ใหญ่กว่าถูกทำลาย Class ที่เป็นองค์ประกอบก็ยังคงอยู่
    4. Composition : เป็นความสัมพันธ์ระหว่าง Object หรือ Class แบบขึ้นต่อกันและมีความเกี่ยวข้องกันเสมอ โดยจะมี Class ซึ่งเป็นองค์ประกอบของ Class อื่นที่ใหญ่กว่า และเมื่อ Class ที่ใหญ่กว่าถูกทำลาย Class ที่เป็นองค์ประกอบก็จะถูกทำลายไปด้วย

2.1.6 แผนภาพลำดับ (Sequence Diagram)

แผนภาพลำดับเป็นแผนภาพที่แสดงให้เห็นถึงปฏิสัมพันธ์ (Interaction) ระหว่างอ็อบเจ็กต์ของคลาส โดยเฉพาะการส่งข้อความ (message) ระหว่างอ็อบเจ็กต์ตามลำดับของเวลาที่เกิดเหตุการณ์ขึ้นก่อนหลัง โดยจะมีสัญกรณ์แสดงให้เห็นลำดับของการส่งข้อความตามเวลาการส่งอย่างชัดเจน ตัวอย่างแผนภาพลำดับ

ส่วนประกอบของแผนภาพลำดับ ประกอบด้วย

- Actor : ผู้ที่เกี่ยวข้องกับระบบ
- Object : อ็อบเจ็กต์ที่ต้องทำหน้าที่ตอบสนองต่อผู้ที่เกี่ยวข้องกับระบบ
- Message :คำสั่งหรือฟังก์ชันที่อ็อบเจ็กต์หนึ่งส่งให้อีกอ็อบเจ็กต์หนึ่งซึ่งสามารถส่งกลับได้ด้วย
- Lifeline : เส้นแสดงชีวิตของอ็อบเจ็กต์ หรือคลาส

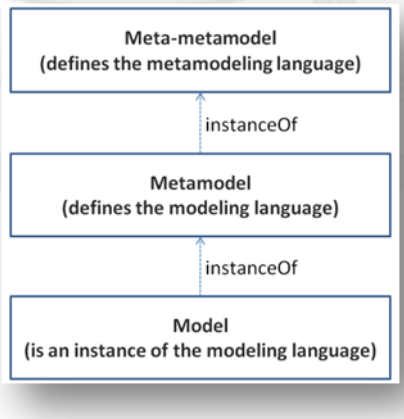
- Fragment : ลักษณะพฤติกรรมของคำสั่งหรือฟังก์ชันที่สามารถเกิดขึ้นได้ ณ บริบทนั้น ซึ่งในงานวิจัยนี้จะเลือกพฤติกรรมที่พบบ่อยในลักษณะการใช้ทำงานในปัจจุบันมานำเสนอ 4 แบบ คือ

1. Alternative : จะเกิดขึ้นเมื่อพฤติกรรมคำสั่งหรือฟังก์ชันที่สามารถมีทางเลือกได้มากกว่าหนึ่งเงื่อนไขที่สามารถเกิดขึ้นได้
2. Option : จะเกิดขึ้นเมื่อพฤติกรรมคำสั่งหรือฟังก์ชันนั้นมีเงื่อนไขแยกออกไปจากพฤติกรรมเดิมที่มีลำดับการทำงานเรียงเป็นลำดับปกติอยู่แล้ว
3. Loop : จะเกิดขึ้นเมื่อพฤติกรรมคำสั่งหรือฟังก์ชันนั้นมีลักษณะการทำงานซ้ำเดิมถ้าอยู่เงื่อนไขที่กำหนดไว้ และจะทำงานจนกระทั่งเงื่อนไขนั้นไม่อยู่ในเงื่อนไขที่กำหนดไว้
4. Reference : จะเกิดขึ้นเมื่อพฤติกรรมคำสั่งหรือฟังก์ชันนั้นได้อ้างอิงพฤติกรรมที่เกิดขึ้นที่กิจกรรมอื่นที่ไม่ได้อยู่ในขอบเขตของบริบทนี้

2.2 **งานวิจัยที่เกี่ยวข้อง**

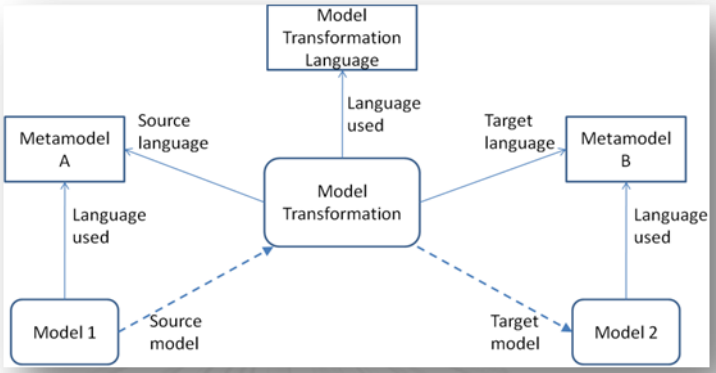
2.2.1 Metamodeling and model transformations in modeling and simulation [1]

บทความงานวิจัยนี้ได้นำเสนอแนวคิดของการพัฒนาขับเคลื่อนด้วยแบบจำลองหรือเอ็มดีดี (Model Driven Development : MDD) ซึ่งแนวคิดนี้ได้เกิดขึ้นมานานแล้ว MDD จะบอกความหมายและความสัมพันธ์ของคำศัพท์ดังนี้ คือ Model, Metamodel, Meta-metamodel, และ Model Transformation



รูปที่ 3 Metamodeling pattern in MDD

รูปที่ 3 แสดงความสัมพันธ์ของแนวคิด MDD มี 3 ระดับ คือ 1) Model เป็นอ็อบเจ็กต์ของ modeling language, 2) Metamodel เป็น Language ของ Model, และ 3) Meta-metamodel เป็นตัวควบคุม Metamodel อีกระดับหนึ่ง



รูปที่ 4 แพทเทิร์นการแปลงแบบจำลองในการพัฒนาขับเคลื่อนด้วยแบบจำลอง

รูปที่ 4 แสดงความสัมพันธ์ของ Model Transformation กับ MDD Pattern (Model, Metamodel และ Meta-metamodel) และหน้าที่ของ Model Transformation ในการแปลงข้อมูลจากแบบจำลองต้นทางไปเป็นแบบจำลองเป้าหมาย รวมทั้งมีหน้าที่เป็นตัวแปลงภาษาจากภาษาต้นทาง (Source language) ให้กลายมาเป็นภาษาเป้าหมาย (Target language) ได้ นอกจากนี้งานวิจัยนี้ได้วิจัยเกี่ยวกับวิธีการประเมินผลการเปลี่ยนแปลงรูปแบบจำลองที่ประสบความสำเร็จที่ได้กำหนดเงื่อนไขเป็นเกณฑ์สำหรับการประเมินผลที่ได้สร้างขึ้นจากการนำหลักการ MDD มาใช้เป็นแนวทางในการพัฒนาระบบ

แนวทางงานวิจัยนี้ได้ประยุกต์งานวิจัยข้างต้นในการนำแนวคิดและหลักการทำงานของ MDD มาใช้เป็นแนวทางในการแปลงรูปแบบโมเดล โดยในงานวิจัยนี้ได้พัฒนาเครื่องมือโดยใช้ภาษาเอทีแอลเป็นตัวขับเคลื่อนเมทาโมเดลของแผนภาพยูสเคสประกอบคำอธิบายและเมทาโมเดลของแผนภาพคลาสไปเป็นเมทาโมเดลของแผนภาพลำดับ นอกจากนี้ได้นำเอาเงื่อนไขที่งานวิจัยก่อนหน้าได้นำเสนอไว้มาใช้เป็นเกณฑ์ในการประเมินผลลัพธ์ของเครื่องมือที่ได้จากงานวิจัยนี้

2.2.2 UML Sequence Diagram Generator System from Use Case Description Using Natural Language [7]

บทความงานวิจัยนี้ได้นำเสนอแนวทางในการสร้างเครื่องมือที่ใช้ประกอบการวิเคราะห์และออกแบบแผนภาพไดอะแกรมที่จะใช้ประกอบการทำความเข้าใจสำหรับนักวิเคราะห์และออกแบบที่มีความสนใจเบื้องต้น ซึ่งในงานวิจัยนี้ได้นำเสนอวิธีการสร้างแผนภาพลำดับจากแผนภาพคลาสและแผนภาพยูสเคสประกอบคำอธิบาย โดยในงานวิจัยจะมุ่งเน้นวิธีการนำภาษาธรรมชาติมาใช้ในการประมวลผลข้อความจากคำอธิบายยูสเคส แล้วแบ่งประโยคออกเป็นคำแต่ละประเภท ดังแสดงในรูปที่

9 แต่งานวิจัยนี้ยังมีข้อจำกัด คือประโยคหรือข้อความที่เขียนจะต้องเป็นเฉพาะข้อความที่เขียนตามหลักไวยากรณ์ของภาษาสเปนและจะต้องหลีกเลี่ยงคำหรือประโยคที่กำกวม

แนวทางงานวิจัยนี้ได้ประยุกต์งานวิจัยข้างต้นในการสกัดข้อความจากแผนภาพยูสเคสประกอบคำอธิบายโดยใช้เทคนิคการนำภาษาธรรมชาติมาใช้และขอบเขตของงานวิจัยนี้จะยังคงคล้ายๆกับงานวิจัยข้างต้น คือ จะสนับสนุนเฉพาะภาษาอังกฤษ และโครงสร้างประโยคควรที่จะเขียนตามหลักไวยากรณ์ที่กำหนด

2.2.3 A Meta-Model for Textual Use Case Description [8]

บทความงานวิจัยนี้ได้นำเสนอเทคนิคการสร้างเมทาโมเดลของแผนภาพยูสเคสประกอบคำอธิบายเพิ่มเติมจากแผนภาพยูสเคส เนื่องจากในการกำหนดมาตรฐานส่วนประกอบของโมเดล ได้กำหนดเฉพาะเมทาโมเดลของแผนภาพยูสเคสไว้เท่านั้น ซึ่งเมทาโมเดลที่สร้างขึ้นนี้จะขึ้นอยู่กับส่วนประกอบพื้นฐานที่พบบ่อยในการใช้งานรูปแบบของแผนภาพยูสเคสประกอบคำอธิบาย และในงานวิจัยนี้ได้ใช้ภาษาโอซีแอลซึ่งเป็นหนึ่งในภาษาที่ใช้สำหรับแปลงโมเดลและตรวจสอบความสอดคล้องกันของโมเดล

แนวทางงานวิจัยนี้ได้ประยุกต์งานวิจัยข้างต้นเป็นแนวทางในการสร้างเมทาโมเดลของแผนภาพยูสเคสประกอบคำอธิบาย ซึ่งในงานวิจัยนี้การสร้างเมทาโมเดลจะออกแบบให้คล้อยตามส่วนประกอบที่ได้จากการสร้างแผนภาพยูสเคสประกอบคำอธิบายบนเครื่องมือวิชวลพาราดาม และได้พัฒนาภาษาเอทีแอลเพื่อทำการแปลงโมเดลและเพื่อตรวจสอบความสอดคล้องกันของโมเดล

2.2.4 Translating Use Cases to Sequence Diagrams [9]

บทความงานวิจัยนี้ได้นำเสนอโครงสร้างประโยคอย่างง่ายที่ใช้ในการเขียนคำอธิบายยูสเคสดังในรูปที่ 5 แล้วนำไปสร้างเป็นข้อความในแผนภาพลำดับ

No.	Syntactic Structure	Sender	Receiver	Action	Argument
1	subject verb object	subject	object or the class of object	verb	
2	subject verb object (to) verb1 (object1)	subject	object	verb1 (+ object1)	(object1)
3	subject verb object participle (object1)	subject	object	participle verb (+ object1)	(object1)
4	subject verb object adjective	subject	object	be + adjective	
5	subject verb object object1	subject	object	set + object1	(object1)
6	subject verb object conjunctive to verb1 (object1)		subject	verb	object, verb1 (+ object1)
7	subject verb gerund (object)		subject	verb	gerund verb (+ object)
8	subject verb object preposition object1	subject	object1	verb + object	(object)
9	subject verb object object1	subject	object	verb	object1
10	subject verb (for) complement		subject	verb	complement
11	subject verb		subject	verb	
12	subject be predicative		subject	be + predicative	
13	subject verb preposition object		subject	verb + preposition	object

รูปที่ 5 โครงสร้างประโยคอย่างง่ายสำหรับเขียนคำอธิบายยูสเคส

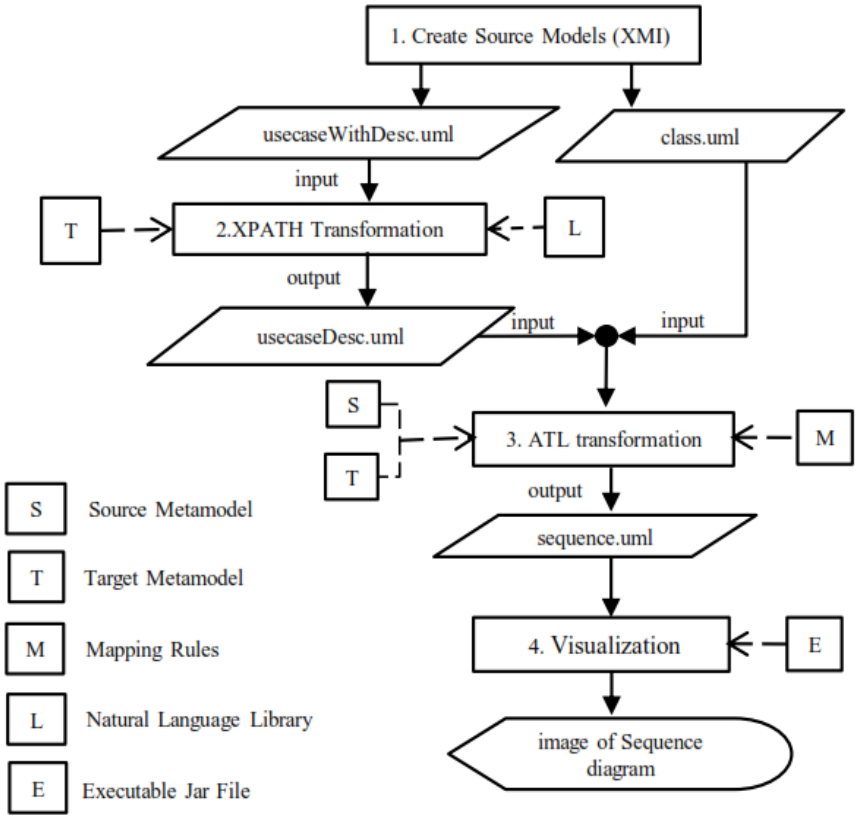
แนวทางงานวิจัยนี้ได้ประยุกต์งานวิจัยข้างต้นเป็นแนวทางในการกำหนดขอบเขตสำหรับการสร้างคำอธิบายยูสเคสที่ใช้ในงานวิจัย และสนับสนุนขั้นตอนการนำไลบรารี่ของภาษาธรรมชาติมาใช้ในงานวิจัยนี้ด้วย

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

บทที่ 3
แนวทางในการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ

3.1 ภาพรวมของแนวทางในการสร้างเครื่องมือในการแปลงรูปแบบโมเดล

งานวิจัยนี้นำเสนอแนวทางสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ แบ่งการทำงานออกเป็น 4 ส่วนหลัก คือ 1) ส่วนนำเข้าข้อมูลคำอธิบายแผนภาพยูสเคสและแผนภาพคลาส 2) ส่วนการออกแบบเมทาโมเดลต้นทางของคำอธิบายแผนภาพยูสเคสโดยใช้การแปลงแบบ XSLT 3) ส่วน ออกแบบและพัฒนาภาษาเพื่อใช้ในการแปลงโมเดล และ4) ส่วนแแสดงจินตทัศน์แบบจำลองผลลัพธ์ โดยภาพรวมของงานวิจัยแสดง ดังรูปที่ 6



รูปที่ 6 ภาพรวมของงานวิจัย

3.1.1 ส่วนนำเข้าข้อมูลคำอธิบายแผนภาพยูสเคสและแผนภาพคลาส

ข้อมูลนำเข้าของงานวิจัยนี้เป็นแผนภาพยูสเคสที่ประกอบด้วยคำอธิบายแผนภาพยูสเคสและแผนภาพคลาสยูเอ็มแอล สัญกรณ์ของแผนภาพต้องเป็นไปตามรูปแบบที่กำหนดไว้ในยูเอ็มแอล

เวอร์ชัน 2.3 แผนภาพยูเอ็มแอลนี้สามารถแปลงแผนภาพให้อยู่ในรูปแบบเอกซ์เอ็มไอ (XMI: XML Metadata Interchange) ซึ่งเป็นภาษามาตรฐานสำหรับแผนภาพยูเอ็มแอล ตัวอย่างของคำอธิบายยูสเคสแสดงดังรูปที่ 6 และตัวอย่างของคำอธิบายแผนภาพคลาสแสดงดังรูปที่ 7 เมื่อแปลงให้อยู่ในรูปแบบเอกซ์เอ็มไอแล้ว จะสามารถทำการสกัดหาเมทาโมเดลของแต่ละแผนภาพได้

รูปที่ 7 ตัวอย่างเอกสารเอกซ์เอ็มไอของคำอธิบายยูสเคส

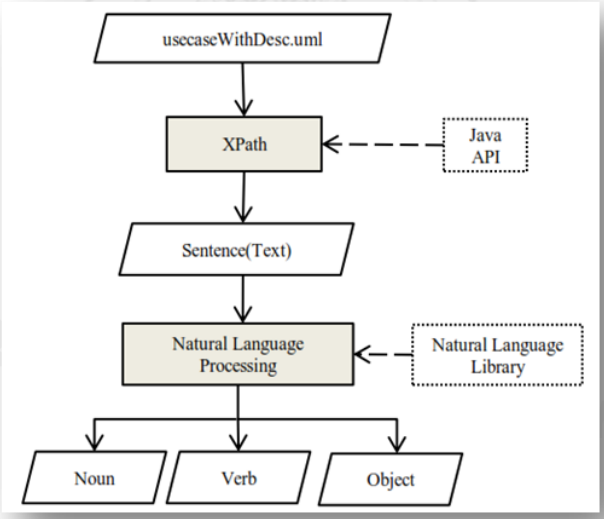
รูปที่ 8 ตัวอย่างเอกสารเอกซ์เอ็มไอของคำแผนภาพคลาส

รายละเอียดที่ต้องการสกัดในคำอธิบายยูสเคสที่ใช้ในงานวิจัยนี้ สามารถสกัดได้จาก XMI Tag ดังแสดงสรุปในตารางที่ 2 ข้อมูลที่สกัดจากเอกสารเอกซ์เอ็มไอ

3.1.2 ส่วนการออกแบบเมทาโมเดลต้นทางของคำอธิบายแผนภาพยูสเคสโดยใช้เทคโนโลยีเอกซ์พาสต์ (XPath Technology)

ขั้นตอนนี้เป็นการใช้เทคโนโลยีเอกซ์พาสต์เข้าไปสกัดเอาข้อมูลของคำอธิบายแผนภาพยูสเคสหรือ ไฟล์ usecaseWithDesc.uml จากรูปที่ 9 โดยเราจะเข้าไปสกัดค่าข้อมูลตามตารางที่สรุปไว้ในตารางที่ 1 เพื่อให้ได้ไฟล์เอกซ์เอ็มไออยู่ในรูปแบบโครงสร้างเมทาโมเดลของแผนภาพยูสเคสประกอบคำอธิบาย เมื่อทำขั้นตอนสกัดค่าที่ต้องการมาเรียบร้อยแล้ว ขั้นตอนต่อไปจะนำข้อมูลที่ได้มาประมวลผลโดยภาษาธรรมชาติ (Natural Language) ซึ่งในงานวิจัยนี้เราจะนำไลบรารี่ชื่อว่า Stanford-postagger มาช่วยสนับสนุนขั้นตอนการสกัดคำเพราะเราต้องการที่สกัดข้อความ 1 ประโยคให้แบ่งประเภทของคำออกมาเป็นคำนาม คำกริยา และ กรรม และจากความสามารถของไลบรารี่ที่นำมาประมวลผลคำจะแบ่งประเภทของคำได้ดังรูปที่ 10 ตัวอย่าง เช่น ประโยค “Officer create buying transaction.” สามารถแบ่งคำได้ ดังนี้ [Officer_NNP] [create_VB] [buying_VBG] [transaction_NN] โดย

- NNP, NN หมายถึง คำนาม
- VBP หมายถึง คำกริยา



รูปที่ 9 วิธีการสกัดข้อมูลจากข้อมูลนำเข้าคำอธิบายยูสเคส

ตารางที่ 2 ข้อมูลที่สกัดจากเอกสารเอกซ์เอ็มไอ

XMI Tag Name	ตัวอย่าง	XMI Tag Attribute	XMI Tag Value	คำอธิบาย
packagedElement	<packagedElement name="Borrow Books" xmi:type="uml:UseCase">	name	Use case name	ชื่อของ use case เช่น “Borrow Books”
		xmi:type	uml:UseCase	คุณลักษณะที่จะทำให้รู้ได้ว่า packagedElement ใดเป็น use case
useCaseDescriptions	<useCaseDescriptions> ..... </useCaseDescriptions>	-	-	Tag ภายใต้ packagedElement ที่เป็น parent ของ use case description ทั้งหมด
useCaseDescription	<useCaseDescription ... name="Full" >	Name	Use case description name	ชื่อของ use case description (หนึ่ง use case สามารถมีได้หลาย use case description)
items	<items> ..... </items>	-	-	เก็บข้อมูลจาก แท็บ“Description” ที่ export มาจาก visual paradigm เช่นข้อมูล use case id, Primary Actor เป็นต้น

ตารางที่ 2 ข้อมูลที่สกัดจากเอกสารเอกซ์เอ็มไอ (ต่อ)

item	<item allowRename="true" ..... name="Use Case ID" sequenceNumber="1" value="UCD02"/>	Name= “Use Case ID”	value	เก็บข้อมูล usecase id
item	<item allowRename="true" ..... name="PrimaryActor" sequenceNumber="2" value="Member"/>	Name= “Primary Actor”	value	เก็บข้อมูล primary actor
item	<item allowRename="true" ..... name="Secondary Actor(s)" sequenceNumber="3" value=""/>	Name=“Secondary Actor(s)”	value	เก็บข้อมูล secondary actor

ตารางที่ 2 ข้อมูลที่สกัดจากเอกสารเอกซ์เอ็มไอ (ต่อ)

stepContainers	<stepContainers> ... </stepContainers>	-	-	จุดเริ่มต้นเพื่ออธิบายข้อมูล use case description.
vpumlModel	<vpumlModel ... modelType="StepContainer">	modelType	StepContainer	จุดเริ่มต้นของแต่ละ step ที่เป็นข้อมูล use case description.
	<vpumlModel .... modelType="Step">	modelType	Step	อธิบายแต่ละขั้นตอนย่อยของ use case description.
properties	<properties>...</properties>	-	-	เก็บข้อมูลเพื่อบรรยายคุณลักษณะแต่ละส่วนของ use case description
property	<property name="name" type="string" value="Member notify borrow to Officer"/>	name	name	คุณลักษณะที่บอกลักษณะว่าเป็น Tag ที่เก็บข้อความ description
		value		เก็บข้อความที่เป็นรายละเอียดของขั้นตอนนั้น ๆ
	<property name="steps" type="models">	name	steps	คุณลักษณะที่บอกลักษณะ use case description ว่ามีรายละเอียดข้อย่อยเพิ่มเติม

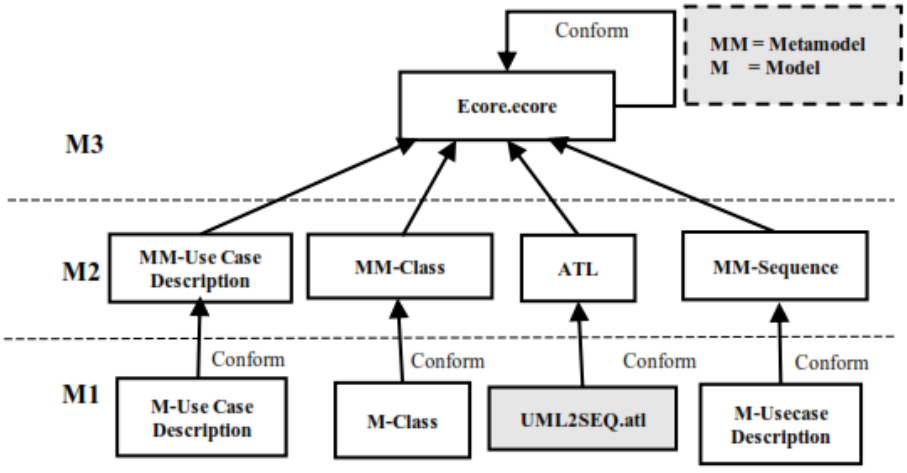
The Penn Treebank English POS tag set

1. CC	Coordinating conjunction	25. TO	to
2. CD	Cardinal number	26. UH	Interjection
3. DT	Determiner	27. VB	Verb,base form
4. EX	Existential there	28. VBD	Verb, past tense
5. FW	Foreignword	29. VBG	Verb,gerund/present
6. IN	Preposition/subordinating participle conjunction	30. VBN	Verb, past participle
7. JJ	Adjective	31. VBP	Verb, non-3rd ps. sing. present
8. JJR	Adjective,comparative	32. VBZ	Verb,3rd ps. sing. present
9. JJS	Adjective, superlative	33. WDT	wh-determiner
10. LS	Listitem marker	34. WP	wh-pronoun
11. MD	Modal	35. WP$	Possessive wh-pronoun
12. NN	Noun, singular or mass	36. WRB	wh-adverb
13. NNS	Noun, plural	37. #	Pound sign
14. NNP	Proper noun, singular	38. $	Dollar sign
15. NNPS	Proper noun, plural	39. .	Sentence-finalpunctuation
16. PDT	Predeterminer	40. ,	Comma
17. POS	Possessive ending	41. :	Colon, semi-colon
18. PRP	Personal pronoun	42. (	Left bracket character
19. PP$	Possessive pronoun	43.)	Right bracket character
20. RB	Adverb	44. "	Straight double quote
21. RBR	Adverb, comparative	45. '	Leftopen single quote
22. RBS	Adverb, superlative	46. "	Leftopen double quote
23. RP	Particle	47. '	Right close single quote
24. SYM	Symbol (mathematical or scientific)	48. "	Right closedouble quote

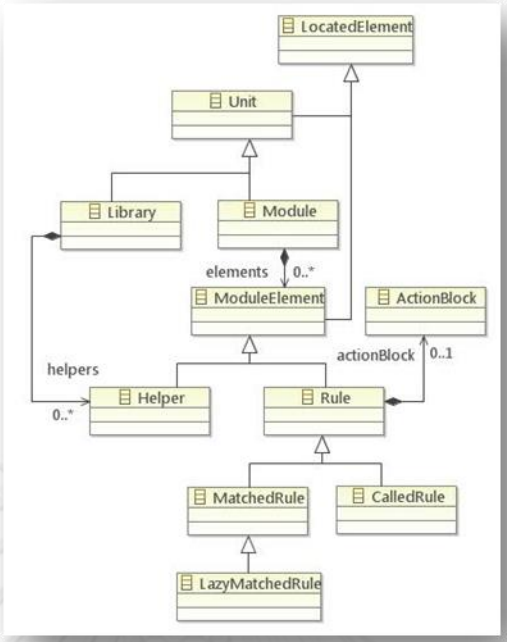
รูปที่ 10 ประเภทของคำที่สามารถสกัดออกมาได้ [8]

3.1.3 ส่วนออกแบบและพัฒนาภาษาเพื่อใช้ในการแปลงโมเดล

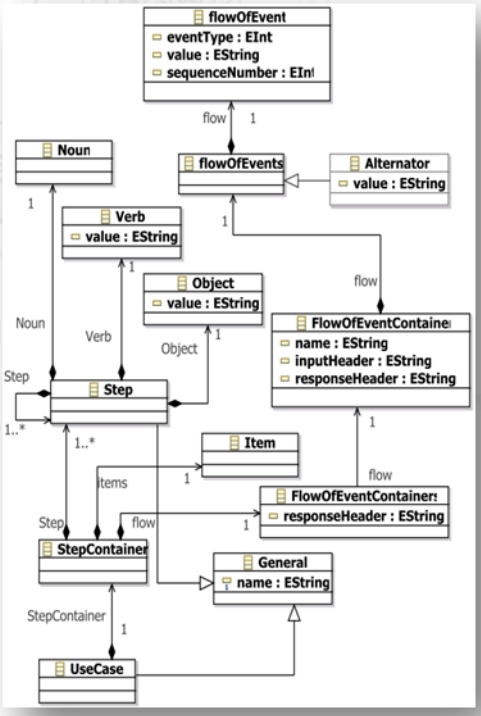
ขั้นตอนนี้เป็นการพัฒนาโปรแกรมเอทีแอลสำหรับใช้ในการแปลงจากแบบจำลองหนึ่งไปเป็นอีกแบบจำลองหนึ่ง แสดงได้ดังรูปที่ 11 และ สำหรับเมทาโมเดลของภาษาเอทีแอลสามารถแสดงได้ดังรูปที่ 12 โดยจะต้องสร้างกฎของการแปลงค่าระหว่างเมทาโมเดลต้นทาง คือ เมทาโมเดลของแผนภาพยูสเคสประกอบคำอธิบาย ดังรูปที่ 13 และ เมทาโมเดลของแผนภาพคลาส ดังรูปที่ 14 กับเมทาโมเดลเป้าหมาย คือ เมทาโมเดลของแผนภาพลำดับ ดังรูปที่ 15



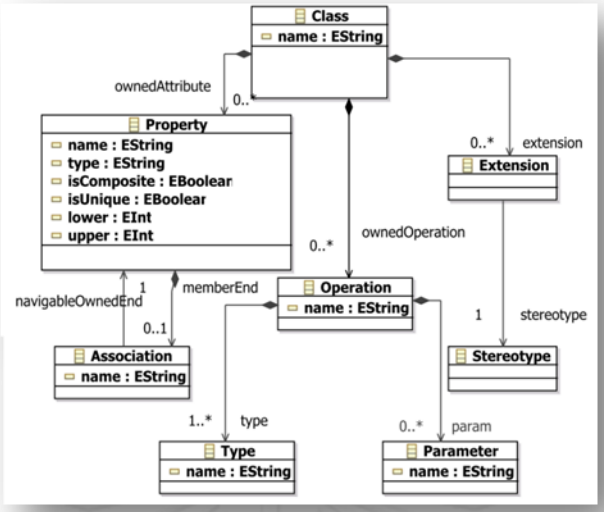
รูปที่ 11 การแปลงแบบโมเดลโดยภาษาเอทีแอล



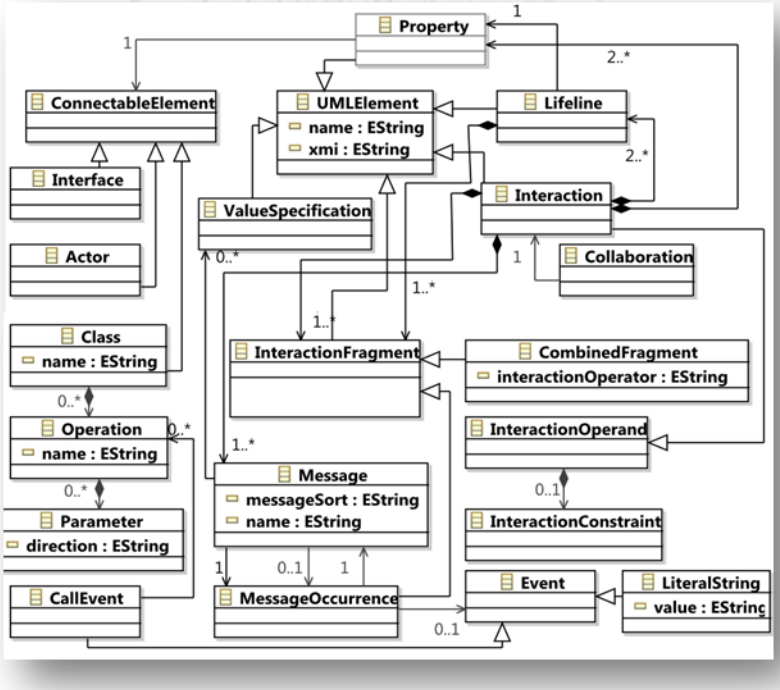
รูปที่ 12 เมทาโมเดลของภาษาเอทีแอล [5]



รูปที่ 13 เมทาโมเดลของคำอธิบายยูสเคส



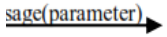
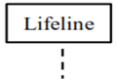
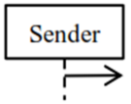
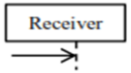
รูปที่ 14 เมทาโมเดลของแผนภาพคลาส [2]



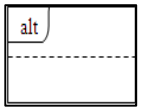
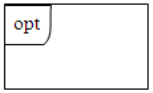
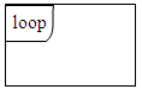
รูปที่ 15 เมทาโมเดลของแผนภาพลำดับ [2]

เมื่อทำการสร้างเมทาโทเดลทั้งต้นทางและปลายทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นขั้นตอนการสร้างกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลทั้งสอง โดยจะสรุปความสัมพันธ์ได้ดังตารางที่ 3

ตารางที่ 3 การสร้างกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลต้นทางและ

Input				Output				Rule
Class Meta-Model (CL)		Use Case Description Metamodel (UCD)		Sequence Metamodel (SEQ)				
EClass[1]	*EAttribute*[2]	*EClass*[1]	*EAttribute*[2]	*Element*	*Symbol Example*	*EClass*[1]	*EAttribute*[2]	
-	-	<Usecase>	<primaryActor>	<Actor>		<Actor>	<name>	
<Operation>	<name>	<Verb>	<value>	<Message>		<Message>	<name>	Rule 1
<Class>	<name>	-	-	<Lifeline>		<Lifeline>	<name>	Rule 2
<Class>	<name>	<Noun>	<value>	(message) <start>		<Lifeline>	<name>	Rule 3
<Class>	<name>	<Object>	<value>	(message) <finish>		<Lifeline>	<name>	Rule 4

ตารางที่ 3 การสร้างกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลต้นทางและปลายทาง (ต่อ)

Input				Output				Rule
Class Meta-Model (CL)		Use Case Description Metamodel (UCD)		Sequence Metamodel (SEQ)				
EClass[1]	*EAttribute*[2]	*EClass*[1]	*EAttribute*[2]	*Element*	*Symbol Example*	*EClass*[1]	*EAttribute*[2]	
-	-	<Alternate>	<id> <parentId>	Fragment Alternative		<Alternate>	<id> <parentId>	Rule 5
-	-	<Optional>	<id> <parentId>	Fragment Option		<Optional>	<id> <parentId>	Rule 6
-	-	<Loop>	<id> <parentId> <condition>	Fragment Loop		<Loop>	<id> <parentId> <condition>	Rule 7
-	-	<Step>	<ref>	Interaction Use		<Message>	<ref>	-

[1]EClass is represents a class, with zero or more attributes and zero or more references.

[2]EAttribue is represents an attribute which has a name and a type.

จากตารางความสัมพันธ์สามารถอธิบายเงื่อนไขในการสร้างกฎความสัมพันธ์เป็นพีซูโดโค้ด ได้ดังนี้

Rule 1: Create Message

```
IF (UCD)Sentence.Verb.value is Empty

     THEN (SEQ)Message.name = "?"

ELSE IF (UCD)Sentence.Verb.value match  any  (CL)Operation.name
     THEN

         (SEQ)Message.name = (CL)Operation.name
    ELSE
         (SEQ)Message.name = (UCD)Sentence.Verb.value

END IF
```

Rule 2: Create Lifeline

```
LOOP (CL)Class.name

         (SEQ)Lifeline.name = (CL)Class.name

 END
```

Rule 3: Create Sender receiving message

```
IF (UCD)Sentence.Noun.value is Empty

        IF (SEQ)Lifeline.name = "Dummy" is exist

                start(Lifeline).name = "Dummy"

        ELSE

                Create new (SEQ)Lifeline with name "Dummy"

        END IF

ELSE IF (UCD)Sentence.Noun.value match any (CL)Class.name

        start(Lifeline) = (CL)Class.name

        /*create new if not exist*/

     ELSE

        start(Lifeline) = (UCD)Sentense.Noun.value

        /*create new if not exist*/

END IF
```

Rule 4: Create Receiver receiving message

```
IF (UCD)Sentence.Object.value is Empty
        IF (SEQ)Lifeline.name = "Dummy" is exist
                finish(Lifeline).name = "Dummy"
        ELSE
                Create new (SEQ)Lifeline with name "Dummy"
        END IF
ELSE IF (UCD)Sentence.Object.value match any (CL)Class.name
        finish(Lifeline) = (CL)Class.name
        /*create new if not exist*/
     ELSE
        finish(Lifeline) = (UCD)Sentense.Object.value
        /*create new if not exist*/
END IF
```

Rule 5: Create Combined Fragment having value of alternatives

```
IF (UCD)Item.name match 'Alternative flows and exceptions' and (UCD)flowOfEvent.sequenceNumber
exists in Item.value
  THEN
     (SQ)Combinedfragment.interactionOperator = 'alt'
END IF
```

Rule 6: Create Combined Fragment having value of options

```
IF (UCD)Item.name match 'Alternative flows and exceptions'
and (UCD)flowOfEvent.sequenceNumber exists in Item.value
   THEN
(SQ)Combinedfragment.interactionOperator ='opt'
END IF
```

Rule 7: Create Combined Fragment having value of loops

```
IF (UCD)Step.name contain string "loop"
  THEN
    (SEQ)Combinedfragment.interactionOperator ='loop'
END IF
```

3.1.4 ส่วนแแสดงจินตทัศน์แบบจำลองผลลัพธ์

ขั้นตอนนี้จะนำผลลัพธ์ที่ได้จากการประมวลผล ซึ่งจะอยู่ในรูปแบบเอกซ์เอ็มไอของแผนภาพลำดับ แล้วนำข้อมูลที่ได้จัดให้อยู่ในรูปแบบที่เครื่องมือแสดงจินต์ทัศน์สามารถแสดงผลเป็นแผนภาพลำดับได้ ในงานวิจัยนี้ได้นำ Execute Jar File มาช่วยสนับสนุนขั้นตอนแสดงจินตทัศน์ ซึ่งจะต้องเขียนตามโครงสร้างไวยากรณ์ที่กำหนด ดังตารางที่ 4

ตารางที่ 4 คำสั่งโปรแกรม sequence editor เปรียบเทียบกับเอกซ์เอ็มไอของแผนภาพลำดับ

สัญลักษณ์ในแผนภาพลำดับ	โครงสร้างคำสั่งการแสดงรูปแผนภาพลำดับที่กำหนดโดยเครื่องมือ Sequence Editor	แท็กเอกซ์เอ็มไอไฟล์ของแผนภาพลำดับ	คำอธิบาย
Actor	ActorName:**actor**	<Actor>	สร้างผู้ใช้ระบบ
Message	ObjectSender:ObjectReceiver.**Message**	<Message>	สร้างข้อความ
Lifeline	**ObjectSender**:**ObjectReceiver**	<Lifeline>	สร้างช่วงชีวิตของอ็อปเจกต์
Object	**ObjectSender**:ObjectReceiver	<Object>	สร้างอ็อปเจกต์
Fragment (Loop)	<**c:loop** condition > .....</**c:loop**>	<Loop>	สร้างเงื่อนไข กรณีที่มีการทำงานซ้ำเดิม
Fragment (Option)	<**c:if** condition > .....</**c:if**>	<Option>	สร้างเงื่อนไขกรณีที่การทำงานเกิดขึ้นได้หลายกรณี
Fragment (Option)	<**c:ref** use case name> .....</**c:ref**>	<Reference>	สร้างเงื่อนไขกรณีที่มีการอ้างอิงถึงยูสเคสอื่น

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

บทที่ 4
การออกแบบและพัฒนาระบบ

รายละเอียดในบทนี้จะอธิบายการออกแบบและพัฒนาระบบต้นแบบเพื่อสนับสนุนแนวทางการสร้างแผนภาพลำดับที่ได้นำเสนอในบทที่ 3 โดยเนื้อหาประกอบด้วย1)ข้อกำหนดเบื้องต้นของระบบ 2)ความต้องการเชิงหน้าที่ 3)การออกแบบสถาปัตยกรรมระบบ และ 4)การพัฒนาระบบ

4.1 ข้อกำหนดเบื้องต้นของระบบ

- ผู้ใช้งาน (User)

ควรเป็นผู้มีหน้าที่ในการบูรณาการระบบ (Integrator) นักวิเคราะห์ระบบ (System Analyst) หรือผู้ทดสอบระบบ (Tester) ซึ่งเป็นผู้ที่มีความรู้เกี่ยวกับภาพรวมของระบบ ทั้งในด้านข้อกำหนดความต้องการ การออกแบบ และการทดสอบ

- ข้อมูลนำเข้า (Input)

แฟ้มเอกสารรูปแบบเอกซ์เอ็มไอที่ได้มาจากการแปลงคำอธิบายยูสเคสและแผนภาพคลาส ด้วยเครื่องมือวาดแผนภาพยูเอ็มแอล วิชวลพาราดามสำหรับยูเอ็มแอล เวอร์ชันโมเดลเลอร์ 10.2

- ข้อมูลนำออก (Output)

1. แฟ้มเอกสารของแผนภาพลำดับในรูปแบบแบบยูเอ็มแอลที่สามารถเปิดด้วยโปรแกรม Sequence Editor
2. แฟ้มเอกสารของแผนภาพลำดับในรูปแบบไฟล์รูปภาพ

- ข้อจำกัดของระบบ(Constraint)

1. ข้อมูลนำเข้าจะสนับสนุนภาษาอังกฤษเท่านั้น
2. ความถูกต้องของข้อมูลนำออกขึ้นอยู่กับความสอดคล้องของข้อมูลนำเข้า และการออกแบบเมทาโมเดลให้เหมาะสม

4.2 ความต้องการเชิงหน้าที่ (Functional Requirements)

ระบบการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบมีความต้องการเชิงหน้าที่ดังตารางที่ 5

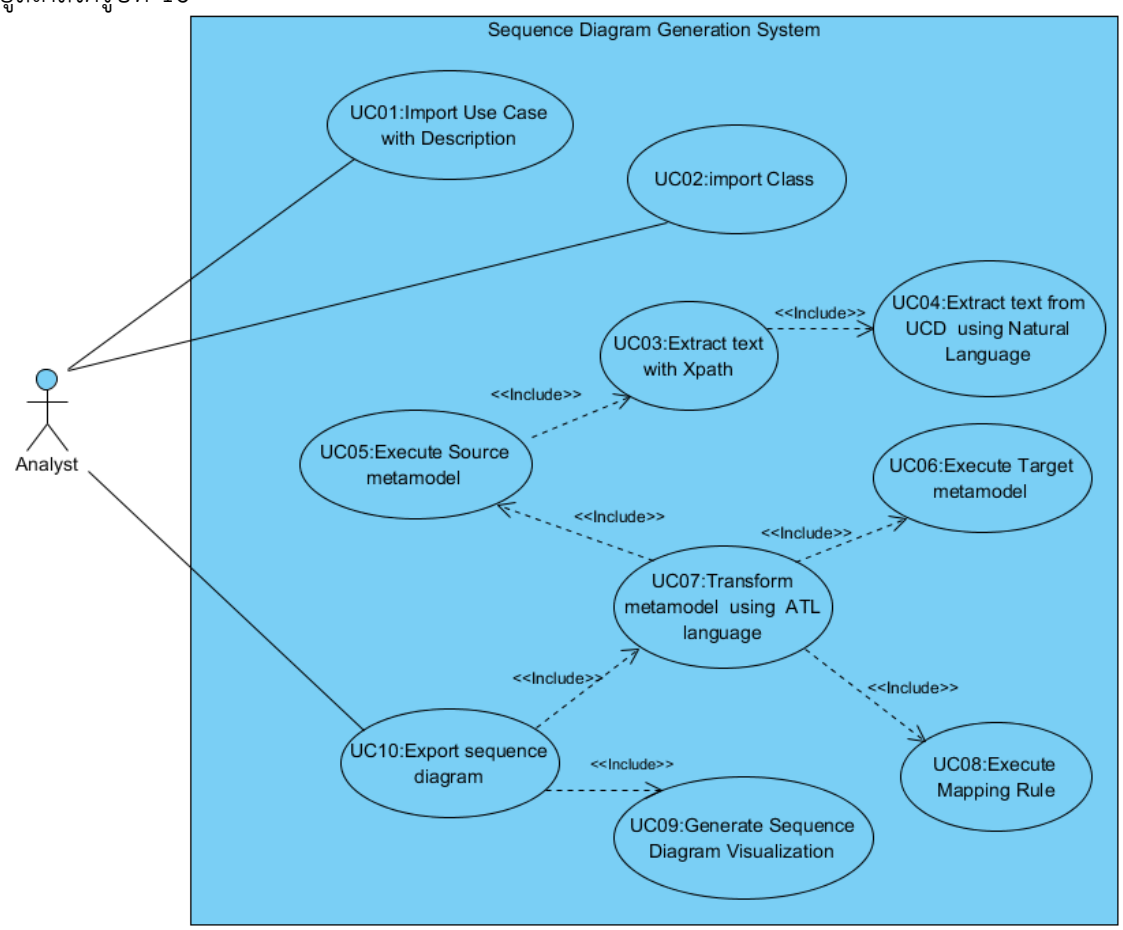
ตารางที่ 5 ความต้องการเชิงหน้าที่ของระบบการสร้างแผนภาพลำดับ

รหัสอ้างอิง	ชื่อ	คำอธิบายความต้องการเชิงหน้าที่
FREQ-01	นำเข้าแผนภาพยูสเคสประกอบคำอธิบายยูสเคสในรูปแบบไฟล์เอกซ์เอ็มไอ	ระบบสามารถนำเข้าแผนภาพยูสเคสประกอบคำอธิบายยูสเคสในรูปแบบไฟล์เอกซ์เอ็มไอได้
FREQ-02	นำเข้าแผนภาพคลาสในรูปแบบไฟล์เอกซ์เอ็มไอ	ระบบสามารถนำเข้าแผนภาพคลาสในรูปแบบไฟล์เอกซ์เอ็มไอได้
FREQ-03	สกัดคำอธิบายแผนภาพยูสเคสประกอบคำอธิ- บายยูสเคสได้	ระบบสามารถสกัดคำอธิบายแผนภาพยูสเคสประกอบคำอธิบายยูสเคสได้
FREQ-04	สกัดคำอธิบายแผนภาพคลาสได้	ระบบสามารถสกัดคำอธิบายแผนภาพคลาสได้
FREQ-05	ประมวลผลเมทาโมเดลต้นทางได้	ระบบสามารถนำเข้าและประมวลผลเมทาโมเดลต้นทางได้
FREQ-06	ประมวลผลเมทาโมเดลปลายทางได้	ระบบสามารถนำเข้าและประมวลผลเมทาโมเดลปลายทางได้
FREQ-07	ประมวลผลโดยใช้ภาษาเอทีแอลเพื่อแปลงรูปแบบโมเดล	ระบบสามารถทำการประมวลผลภาษาเอทีแอลได้
FREQ-08	ประมวลผลกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลต้นทางกับเมทาโมเดลปลายทาง	ระบบสามารถประมวลผลตามกฎในการจับคู่ความสัมพันธ์ระหว่างเมทาโมเดลต้นทางกับเมทาโมเดลปลายทางได้
FREQ-09	ส่งออกข้อมูลแผนภาพลำดับในรูปแบบไฟล์รูปภาพ	ระบบสามารถส่งออกข้อมูลแผนภาพลำดับในรูปแบบไฟล์รูปภาพได้
FREQ-10	ดำเนินสั่งการนำออกข้อมูลแผนภาพลำดับ	ระบบสามารถเริ่มปฏิบัติการตามคำสั่งที่ได้กำหนดไว้

จากความต้องการเชิงหน้าที่สามารถออกแบบและแสดงเป็นแผนภาพยูสเคส (Use Case) ประกอบคำอธิบายแผนภาพยูสเคส (Use Case Description)

4.2.1 แผนภาพยูสเคส

ระบบการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบสามารถแสดงเป็นแผนภาพยูสเคสได้รูปที่ 16



รูปที่ 16 แผนภาพยูสเคสของระบบการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ

4.2.2 คำอธิบายยูสเคส

คำอธิบายยูสเคสจะอธิบายถึงฟังก์ชันการทำงานระบบที่ปรากฏอยู่ในแผนภาพยูสเคส ซึ่งจะอธิบายว่าผู้ใช้งานสามารถทำอะไร และระบบจะตอบสนองการทำงานอย่างไร โดยคำอธิบายยูสเคสแสดงดังตารางที่ 6 ถึง ตารางที่ 15

ตารางที่ 6 คำอธิบายยูสเคส Import Use Case with Description

หมายเลขยูสเคส:	UC01	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	ผู้วิเคราะห์ระบบ		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการนำเข้าแผนภาพยูสเคสประกอบคำอธิบายในรูปแบบเอกซ์เอ็มไอ		
เงื่อนไขขั้นต้น:	แผนภาพยูสเคสประกอบคำอธิบายที่นำเข้าต้องแปลงแผนภาพให้อยู่ในรูปแบบเอกซ์เอ็มไอ		
การทำงานโดยปกติ:	1. ผู้ใช้งานคลิกปุ่ม Import 2. เลือกไฟล์เอกซ์เอ็มไอที่จะนำเข้า 3. ระบบตรวจสอบไฟล์ 4. ระบบประมวลผลไฟล์		
ทางเลือกเพิ่มเติมในการทำงาน:	3-1 นำเข้าไฟล์ไม่ถูกต้อง 3-2 ระบบแสดงข้อความเตือนการนำเข้าไฟล์ไม่ถูกต้อง 3-3 กลับไปทำขั้นตอนที่ 2 หรือออกจากระบบ		

ตารางที่ 7 คำอธิบายยูสเคส Import Class

หมายเลขยูสเคส:	UC02	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	ผู้วิเคราะห์ระบบ		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการนำเข้าแผนภาพคลาสในรูปแบบเอกซ์เอ็มไอ		
เงื่อนไขขั้นต้น:	แผนภาพคลาสที่นำเข้าต้องแปลงแผนภาพให้อยู่ในรูปแบบเอกซ์เอ็มไอ		
การทำงานโดยปกติ:	1. ผู้ใช้งานคลิกปุ่ม Import 2. เลือกไฟล์เอกซ์เอ็มไอที่จะนำเข้า 3. ระบบตรวจสอบไฟล์ 4. ระบบประมวลผลไฟล์		
ทางเลือกเพิ่มเติมในการทำงาน:	3-1 นำเข้าไฟล์ไม่ถูกต้อง 3-2 ระบบแสดงข้อความเตือนการนำเข้าไฟล์ไม่ถูกต้อง 3-3 กลับไปทำขั้นตอนที่ 2 หรือออกจากระบบ		

ตารางที่ 8 คำอธิบายยูสเคส Extract text with XPath

หมายเลขยูสเคส:	UC03	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการสกัดข้อความที่อยู่ในยูสเคสประกอบคำอธิบายโดยใช้เทคโนโลยี XPath เพื่อเข้าไปค้นหาข้อความ สร้างรูปแบบเอกซ์เอ็มไอใหม่ตามที่ต้องการ		
เงื่อนไขขั้นต้น:	นำเข้าแผนภาพยูสเคสประกอบคำอธิบายในรูปแบบเอกซ์เอ็มไอ		
การทำงานโดยปกติ:	1. ระบบอ่านไฟล์เอกซ์เอ็มไอของแผนภาพยูสเคสประกอบคำอธิบายแล้วสกัดข้อมูลที่ต้องการ โดยการจับคู่ความสัมพันธ์ของแท็กที่ปรากฏในแผนภาพยูสเคสประกอบคำอธิบายกับเมทาโมเดลที่ได้ออกแบบไว้ 2. ระบบทำการสร้างไฟล์เอกซ์เอ็มไอในรูปแบบที่ออกแบบไว้ เพื่อให้ระบบสามารถทำงานในขั้นตอนต่อไปได้		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

ตารางที่ 9 คำอธิบายยูสเคส Extract text from UCD using Natural Language

หมายเลขยูสเคส:	UC04	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการสกัดข้อความที่อยู่ในยูสเคสประกอบคำอธิบายโดยใช้ภาษาธรรมชาติ (Natural Language) มาช่วยในการแบ่งแยกชนิดของคำในแต่ละประโยคที่มีในแผนภาพยูสเคสประกอบคำอธิบาย		
เงื่อนไขขั้นต้น:	นำเข้าแผนภาพยูสเคสประกอบคำอธิบายในรูปแบบเอกซ์เอ็มไอ		
การทำงานโดยปกติ:	1. ระบบอ่านไฟล์เอกซ์เอ็มไอของแผนภาพยูสเคสประกอบคำอธิบายแล้วสกัดข้อมูลออกมาโดยแบ่งข้อความ 1 ประโยคให้เป็น ประธาน กริยาและกรรม 2. ระบบทำการสร้างไฟล์เอกซ์เอ็มไอในรูปแบบที่ออกแบบไว้ ซึ่งจะต้องมีการแบ่งแยกคำคือ ประธาน กริยาและกรรม		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

ตารางที่ 10 คำอธิบายยูสเคส Execute Source Metamodel

หมายเลขยูสเคส:	UC05	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการประมวลผลเมทาโมเดลต้นทางโดยภาษาเอทีเอล		
เงื่อนไขขั้นต้น:	ระบบต้องนำเข้าเมทาโมเดลต้นทางเรียบร้อยแล้ว		
การทำงานโดยปกติ:	1. ระบบดึงข้อมูลและจับคู่ความสัมพันธ์จากข้อมูลนำเข้าของแผนภาพประกอบคำอธิบายซึ่งอยู่ในรูปแบบไฟล์เอกซ์เอ็มไอในรูปแบบใหม่และเมทาโมเดลต้นทางที่สร้างได้ 2. ระบบทำการประมวลผลเพื่อจับคู่ความสัมพันธ์จากไฟล์เอกซ์เอ็มไอที่ได้จากขั้นตอนการออกแบบไว้กับเมทาโมเดลที่ออกแบบไว้		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

ตารางที่ 11 คำอธิบายยูสเคส Execute Target Metamodel

หมายเลขยูสเคส:	UC06	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (High)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการประมวลผลเมทาโมเดลปลายทางโดยภาษาเอทีเอล		
เงื่อนไขขั้นต้น:	ระบบต้องนำเข้าเมทาโมเดลปลายทางเรียบร้อยแล้ว		
การทำงานโดยปกติ:	1. ระบบดึงข้อมูลและจับคู่ความสัมพันธ์จากข้อมูลนำเข้าของแผนภาพประกอบคำอธิบายซึ่งอยู่ในรูปแบบไฟล์เอกซ์เอ็มไอในรูปแบบใหม่และเมทาโมเดลต้นทางที่สร้างได้ 2. ระบบทำการประมวลผลเพื่อจับคู่ความสัมพันธ์จากไฟล์เอกซ์เอ็มไอที่ได้จากขั้นตอนการออกแบบไว้กับเมทาโมเดลที่ออกแบบไว้		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 12 คำอธิบายยูสเคส Transform Metamodel by ATL

หมายเลขยูสเคส:	UC07	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการประมวลผลในการแปลงรูปแบบโมเดลโดยโครงสร้างภาษาเอทีแอล		
เงื่อนไขขั้นต้น:	ระบบต้องนำเข้าเมทาโมเดลต้นทางและปลายทางเรียบร้อยแล้ว		
การทำงานโดยปกติ:	1. ระบบทำการประมวลผลข้อมูลนำเข้าในรูปแบบไฟล์เอกซ์เอ็มไอกับเมทาโมเดลที่สร้างได้ 2. ระบบทำการประมวลผลและจับคู่ความสัมพันธ์ตามกฎที่ได้สร้างในระบบ		
ทางเลือกเพิ่มเติมในการทำงาน:	1.1 ระบบไม่สามารถทำงานได้เนื่องจากเมทาโมเดลไม่ถูกต้อง 1.2 ระบบแสดงข้อความเตือนเมทาโมเดลไม่ถูกต้อง		

ตารางที่ 13 คำอธิบายยูสเคส Execute Mapping Rule

หมายเลขยูสเคส:	UC08	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการจับคู่ความสัมพันธ์ของข้อมูลนำเข้า(แผนภาพยูสเคสประกอบคำอธิบายและแผนภาพคลาส)กับข้อมูลนำออก(แผนภาพลำดับ)		
เงื่อนไขขั้นต้น:	ระบบต้องเขียนกฎในการจับคู่ความสัมพันธ์โดยภาษาเอทีแอล		
การทำงานโดยปกติ:	1. ระบบประมวลผลตามกฎในการสร้าง actor 2. ระบบประมวลผลตามกฎในการสร้าง message 3. ระบบประมวลผลตามกฎในการสร้าง lifeline 4. ระบบประมวลผลตามกฎในการสร้าง object(sender) 5. ระบบประมวลผลตามกฎในการสร้าง object(receiver) 6. ระบบประมวลผลตามกฎในการสร้าง fragment(alternative) 7. ระบบประมวลผลตามกฎในการสร้าง fragment(option) 8. ระบบประมวลผลตามกฎในการสร้าง fragment(loop) 9. ระบบประมวลผลตามกฎในการสร้าง fragment(reference)		
ทางเลือกเพิ่มเติมในการทำงาน:	-		

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 14 คำอธิบายยูสเคส Generate Sequence Diagram Visualization

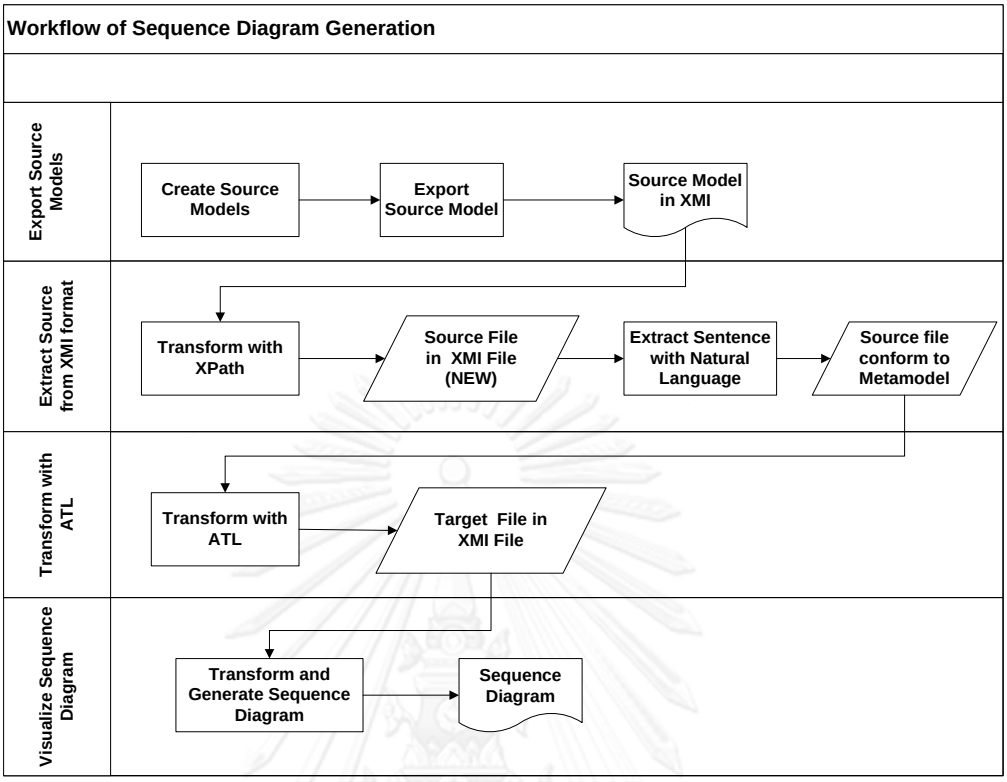
หมายเลขยูสเคส:	UC09	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	-		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการส่งออกข้อมูลกรณีทดสอบแบบบูรณาการในรูปแบบเอกซ์แอลเอส		
เงื่อนไขขั้นต้น:	ระบบต้องสร้างกรณีทดสอบแบบบูรณาการ		
การทำงานโดยปกติ:	1. ระบบสามารถสร้าง actor ในแผนภาพลำดับได้ 2. ระบบสามารถสร้าง message ในแผนภาพลำดับได้ 3. ระบบสามารถสร้าง lifeline ในแผนภาพลำดับได้ 4. ระบบสามารถสร้าง object(sender)ในแผนภาพลำดับได้ 5. ระบบสามารถสร้าง object(receiver)ในแผนภาพลำดับได้ 6. ระบบสามารถสร้าง fragment(alternative)ในแผนภาพลำดับได้ 7. ระบบสามารถสร้าง fragment(option)ในแผนภาพลำดับได้ 8. ระบบสามารถสร้าง fragment(loop)ในแผนภาพลำดับได้ 9. ระบบสามารถสร้าง fragment(reference)ในแผนภาพลำดับได้		
ทางเลือกเพิ่มเติมในการทำงาน:	2.1 ระบบไม่สามารถทำการจับคู่เพื่อหาความสัมพันธ์ตามกฎในการสร้าง message ได้ 2.2 ระบบสร้าง message เป็นคำว่า <<create>> ในแผนภาพลำดับ 5.1 ระบบไม่สามารถทำการจับคู่เพื่อหาความสัมพันธ์ตามกฎในการสร้าง object(receiver) ได้ 5.2 ระบบสร้าง object ใหม่ในแผนภาพลำดับ เป็นคำที่ได้จากการประมวลผลของระบบ		

ตารางที่ 15 คำอธิบายยูสเคส Export Sequence Diagram

หมายเลขยูสเคส:	UC10	**เวอร์ชัน:**	1.0
ความสำคัญ:	ส่วนการทำงานหลัก (Moderate)		
ผู้เกี่ยวข้อง:	ผู้บูรณาการระบบ		
รายละเอียดโดยย่อ:	ยูสเคสนี้อธิบายการสั่งการเพื่อเริ่มปฏิบัติการประมวลผลเพื่อการแปลงโมเดล		
เงื่อนไขขั้นต้น:	ระบบต้องนำเข้าไฟล์ของแผนภาพยูสเคสประกอบคำอธิบาย และแผนภาพคลาสเรียบร้อยแล้ว		
การทำงานโดยปกติ:	1. ผู้ใช้งานคลิกปุ่ม Export to Sequence Diagram 2. ระบบตรวจสอบข้อมูลนำเข้า 3. ระบบเริ่มปฏิบัติการประมวลผลโดยภาษาเอทีแอล 4. ระบบแสดงข้อความเพื่อแสดงผลสถานะการทำงานของระบบว่าอยู่ในระหว่างขั้นตอนใด 5. ระบบแสดงข้อความการประมวลผลเรียบร้อย		
ทางเลือกเพิ่มเติมในการทำงาน:	2-1 นำเข้าไฟล์ไม่ถูกต้อง 2-2 ระบบแสดงข้อความเตือนการนำเข้าไฟล์ไม่ถูกต้อง 2-3 กลับไปทำขั้นตอนที่ 1 หรือออกจากระบบ		

4.3 การออกแบบสถาปัตยกรรมระบบ

ระบบที่พัฒนาขึ้นแบ่งออกเป็นสี่มอดูลหลัก คือ มอดูลการส่งออกข้อมูลนำเข้า มอดูลการแปลงแบบเพื่อสนับสนุนมทาโมเดล มอดูลการแปลงแบบจำลอง และมอดูลการแสดงผลแผนภาพลำดับ โดยทั้ง 4 มอดูลมีความสัมพันธ์กันดังรูปที่ 17 การส่งออกข้อมูลนำเข้าจะทำหน้าที่ในการแปลงข้อมูลนำเข้าให้อยู่ในรูปแบบเอกซ์เอ็มไอ เมื่อได้ไฟล์ในรูปแบบเอกซ์เอ็มไอแล้วจะต้องนำไฟล์ไปสกัดและแปลงผลให้อยู่ในแบบฟอร์มตามเมทาโมเดลที่ได้ออกแบบไว้ แล้วให้มอดูลการแปลงแบบจำลองทำหน้าที่แปลงบทคำสั่งที่เขียนขึ้นในรูปของเอทีแอลเพื่อให้อยู่ในรูปของแบบจำลองตามเมทาโมเดลของแผนภาพลำดับ หลังจากนั้นมอดูลการสร้างแผนภาพลำดับจะทำหน้าที่ในการแปลงแบบจำลองของเมทาโมเดลแผนภาพลำดับให้อยู่ในรูปแบบที่สามารถแสดงผลในรูปของแผนภาพลำดับได้



รูปที่ 17 สถาปัตยกรรมการทำงานของระบบ

4.4 การพัฒนาระบบ

4.4.1 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา

สภาพแวดล้อมที่ใช้ในการพัฒนาระบบมีสภาพแวดล้อมทางด้านฮาร์ดแวร์และซอฟต์แวร์ดังต่อไปนี้

- ฮาร์ดแวร์

  -หน่วยประมวลผลกลาง (CPU) อินเทล คอร์ไอ 5.10 กิกะเฮิรตซ์ (CPU Intel Core 2 Duo 2.10 GHz)

  - หน่วยความจำ (RAM) 3 กิกะไบต์ (RAM 3 GHz)
  - จานบันทึกข้อมูล (Hard Disk) 500 กิกะไบต์ (Hard disk 500 GB)

- ซอฟต์แวร์

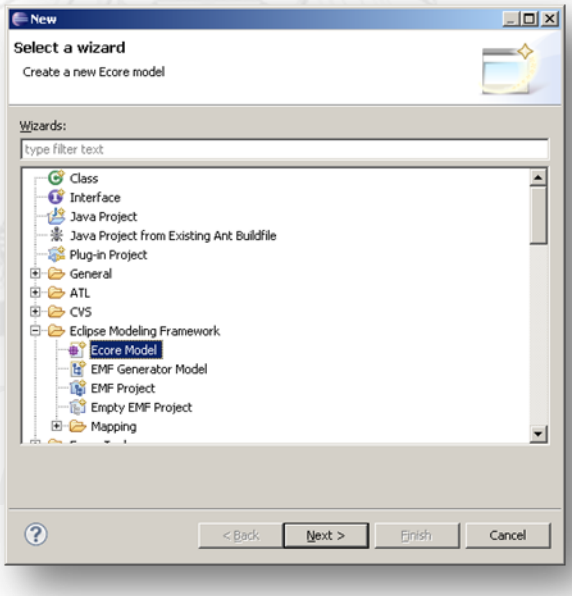
1. ระบบปฏิบัติการ ไมโครซอฟต์วินโดว์เอกซ์พี โปรเฟสชั่นนอล 32 บิต (Microsoft Windows XP Professional 32 bit)
2. เจดีเค 6 (Java JDK 6)
3. โปรแกรม อีคลิปส์ เวอร์ชัน 3.5 (Eclipse Modeling Tools)
4. เครื่องมือวาดแผนภาพยูเอ็มแอล วิชวลพาราดามสำหรับเวอร์ชันโมเดลเลอร์ 10.2 (Visual Paradigm Standard Edition 10.2)

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

5. จาวาสวิง (Swing) สำหรับสร้างส่วนต่อ
6. เครื่องมือการทดสอบแผนภาพลำดับ Quick Sequence Editor
7. เครื่องมือ Lanch4j 3.4 สำหรับแพ็คซอร์จโค้ดที่จำเป็น จากไฟล์นามสกุลจาร์ไฟล์ (.jar) เพื่อใช้สำหรับสร้างเป็นไฟล์ติดตั้ง (.exe)
8. เครื่องมือ Inno Setup 5 สำหรับนำไฟล์ที่ได้จากโปรแกรม Lanch4j 3.4 ไปสร้างเป็นไฟล์สำหรับติดตั้งได้ทันที

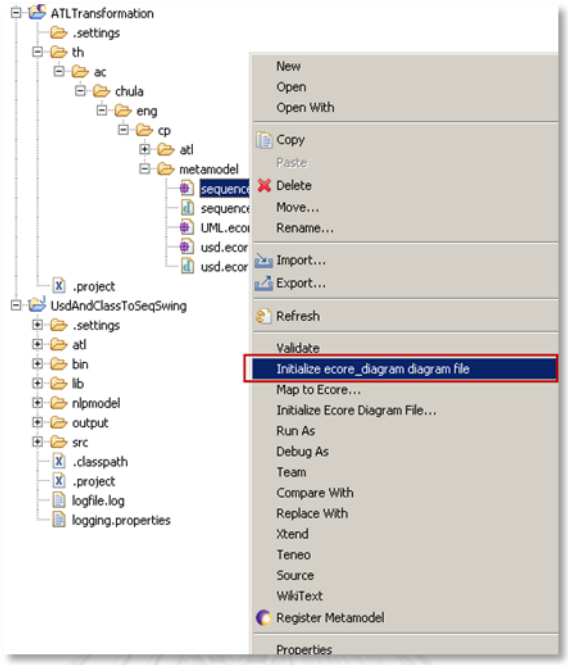
4.4.2 การพัฒนาวิธีการสร้างเมทาโมเดล

การพัฒนาเพื่อการพัฒนาเมทาโมเดลจะใช้ อีคลิปส์โมเดลลิ่งทูล และก่อนที่จะทำการสร้างอีคอร์จะต้องติดตั้งปลั๊กอินเพื่อให้อีคลิปส์สามารถสร้างอีคอร์ได้ ซึ่งวิธีการติดตั้งได้เขียนรายละเอียดไว้ที่ภาคผนวก ข เมื่อติดตั้งเรียบร้อยแล้วในการพัฒนาเราจะทำการสร้างโพรเจกต์ใหม่ แล้วไปที่หัวข้ออีคลิปส์โมเดลลิ่งเฟรมเวิร์ค (Eclipse Modeling Framework) ตามด้วยเลือกอีคอร์โมเดล (Ecore Model) จากหัวข้อในรูปที่ 18 รูปที่ 18 การสร้างอีคอร์โมเดล



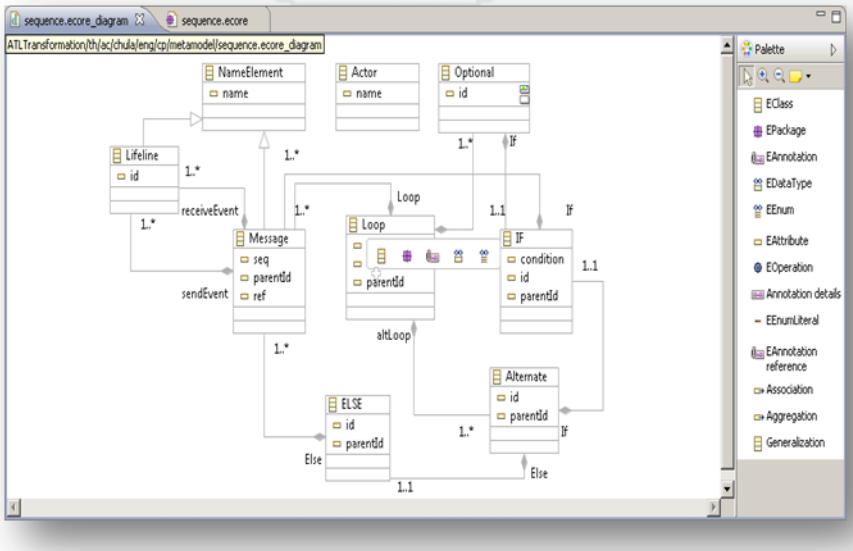
รูปที่ 18 การสร้างอีคอร์โมเดล

เมื่ออีคลิปส์สร้างไฟล์อีคอร์แล้ว เพื่อให้การสร้างอีคอร์สะดวกมากขึ้น ภายใต้ไฟล์อีคอร์ให้ทำการสร้างอีคอร์ไดอะแกรม (Initialize ecore_diagram diagram file)ตามรูปที่ 19



รูปที่ 19 การสร้างอีคอร์ไดอะแกรม

หลังจากนั้นให้ทำการสร้างแผนภาพอีคอร์ไดอะแกรมสำหรับเมทาโมเดลของระบบ (เมทาโมเดลที่ใช้อ้างอิงจากบทที่ 3) ตัวอย่างดังรูปที่ 20



รูปที่ 20 ตัวอย่างการสร้างอีคอร์ไดอะแกรม

หลังจากที่ทำการการสร้างเมทาโมเดลต้นทางและปลายทางเรียบร้อยแล้ว ในขั้นตอนต่อไปคือการพัฒนาภาษาเอทีแอลเพื่อแปลงแบบเป็นโมเดลปลายทาง

4.4.3 การพัฒนาการแปลงรูปแบบโมเดลโดยภาษาเอทีแอล

สำหรับการพัฒนาด้วยภาษาเอทีแอล โครงสร้างภาษาได้แสดงไว้ในตารางที่ 5 และจากตารางได้อธิบายโดยละเอียด ดังต่อไปนี้

```
-- @path classMM=/ATLProject/metamodel/UML.ecore

-- @path usdMM=/ATLProject/metamodel/usd.ecore
```

การระบุที่อยู่ของ metamodel ทั้งส่วนที่เป็น input และ output เพื่อให้อีคลิปส์สามารถทำ ให้การเขียนโค้ดสะดวกมากขึ้น (Auto-completion) เมื่อเขียนโค้ดได้ไม่ว่าจะเป็นการอ้างถึง class, attribute, operation ใน metamodel ที่ได้ประกาศไว้ ซึ่งคำสั่งลักษณะนี้จะมีหรือไม่มีก็ได้ แต่หากมีจะต้องเขียนไว้ที่บรรทัดแรกสุดเท่านั้น

```
module UsdAndClass2Seq;

create OUTSeq : seqMM from INUSD : usdMM, INClass : classMM;
```

การสร้าง code ภาษา ATL เริ่มต้นด้วย syntax “module” เพื่อระบุชื่อของโมดูลที่จะทำการพัฒนาเป็นบรรทัดแรก (หรือถัดจากบรรทัดที่เป็นการระบุที่อยู่ของ metamodel กรณีที่ระบุ) ชื่อของโมดูลจะต้องเป็นชื่อเดียวกับไฟล์ ATL เท่านั้นซึ่งหากชื่อไม่ตรงกัน plugin ATL ที่ติดตั้งใน eclipse ide ก็จะแสดงข้อผิดพลาด และไม่สามารถ compile file (เพื่อให้ได้ไฟล์ .asm) ที่เป็นได้ ถัดจากคำสั่งประกาศโมดูล จะเป็นการระบุภาพรวมของการทำ transformation อันได้แก่ metamodel ต้นทาง และ metamodel ปลายทาง โดยที่ metamodel ต้นทางหรือ input นั้นจะถูกระบุด้วย keyword “from” และตามด้วยชื่อของ metamodel ซึ่งจะใช้อ้างอิงในขั้นตอนการเขียน code ต่อไป ซึ่งส่วนที่เป็น input สามารถมีได้มากกว่า 1 โดยต้องคั่นด้วยคอมม่า ดัง code จะเห็นได้ว่ามีการระบุ input 2 ตัวคือ INUSD และ INClass

```
helper context usdMM!Step def: getMatchNounEqual() :
String =

     classMM!Class.allInstances()->

      select(e | (e.name.trim().toUpper() =
self.Noun.value.trim().toUpper()))->collect(e |
e.name)
```

Helper เปรียบได้กับ method ในภาษา java เป็นกลุ่มของคำสั่งที่สร้างขึ้นเพื่อถูกเรียกใช้งานจากจุดใด ๆ ก็ได้ใน source code เพื่อลดการเขียน code ที่ซ้ำซ้อน และแสดงการทำหน้าที่ใดหน้าที่หนึ่งเป็นสัดส่วน การสร้าง helper เริ่มต้นด้วยการระบุ keyword “helper def :” และตามด้วยชื่อ helper ที่ต้องการ ซึ่งการตั้งชื่อควรให้มีความหมายตามหน้าที่ ๆ ทำงานด้วย กรณีที่ helper

มีการรับค่าหรือพารามิเตอร์เข้ามาเพื่อประมวลผล จะถูกระบุในส่วนที่เป็นวงเล็บหลังจากกำหนดชื่อของ helper ซึ่งในตัวอย่างจะเป็นว่า helper ชื่อ getMatchNounEqual ซึ่งจะถูกเรียกได้จากชนิดข้อมูลที่เป็น usdMM!Step เท่านั้น ถัดจากการกำหนดค่าพารามิเตอร์ (ซึ่งจะมีหรือไม่มีก็ได้) จะเป็นการระบุชนิดของข้อมูลที่เป็นผลลัพธ์ เพื่อคืนค่าให้จุดที่เรียกใช้งาน helper นี้ ดังตัวอย่างจะเห็นได้ว่า helper นี้คืนค่าเป็นชนิดข้อมูล String 1 ตัว นั่นเอง สุดท้ายแล้ว helper จะเสร็จสมบูรณ์จะต้องปิดท้ายด้วย “;”

```
helper context Families!Member def: isFemale() : Boolean =
    if not self.familyMother.oclIsUndefined() then
          true
    else
          if not self.familyDaughter.oclIsUndefined() then
                true
          else
                false
```

ตัวอย่างการสร้าง helper

อธิบายโดยสังเขปสำหรับการทำงานของ helper getMatchNounEqual จะทำหน้าที่ค้นหาชื่อของ class (จาก input class metamodel) โดยถูกเรียกใช้งานมาจากโมเดล usdMM!Step เริ่มจากการดึงข้อมูลชื่อคลาสทั้งหมดจากคลาสโมเดลด้วยคำสั่ง “classMM!class.allInstances()” หลักจากนั้นระบุเงื่อนไขที่จะเป็นการคัดกรองชื่อคลาสที่ต้องการออกมาด้วยคำสั่ง “->select(e | (e.name.trim().toUpper() = self.Noun.value.trim(). toUpper()))” ซึ่งหมายถึงเลือกเฉพาะคลาสที่มีชื่อเท่ากับแอตทริบิวต์ value ของคลาส Noun ซึ่งเป็นลูกของคลาส Step (ใน usdMM metamodel) สุดท้ายจะเป็นการเก็บข้อมูลที่ได้มาทั้งหมดในรูปแบบ collection ด้วยคำสั่ง “->collect(e | e.name)” โดยเก็บเฉพาแอตทริบิวต์ name ซึ่งเป็นชนิดข้อมูล String

```
helper context usdMM!Step def: getMatchNounLike() : String =

      classMM!Class.allInstances()->select(e |
(e.name.trim().toUpper().startsWith(self.Noun.value.trim().toUpper()))

or

(e.name.trim().toUpper().endsWith(self.Noun.value.trim().toUpper())))

    ->collect(e | e.name)

;
```

ตัวอย่างการสร้าง helper

อธิบายตัวอย่างการทำงานของ helper getMatchNounLike จะทำหน้าที่ค้นหาชื่อของ class (จาก input class metamodel) โดยถูกเรียกใช้งานมาจากโมเดล usdMM!Step โดยการดึงข้อมูลชื่อคลาสทั้งหมดจากคลาสโมเดลด้วยคำสั่ง “classMM!class.allInstances()” หลักจากนั้นระบุเงื่อนไขที่จะเป็นการคัดกรองชื่อคลาสที่ต้องการออกมาด้วยคำสั่ง “->select(e |

(e.name.trim().toUpper() = self.Noun.value.trim().toUpper()))” ซึ่งหมายถึงเลือกเฉพาะคลาสที่มีชื่อเริ่มต้นหรือลงท้ายดวยแอตทริบิวต์ value ของคลาส Noun ซึ่งเป็นลูกของคลาส Step (ใน usdMM metamodel) สุดท้ายจะเป็นการเก็บข้อมูลที่ได้มาทั้งหมดในรูปแบบ collection ด้วยคำสั่ง “->collect(e | e.name)” โดยเก็บเฉพาแอตทริบิวต์ name ซึ่งเป็นชนิดข้อมูล String

Lazy rule เป็นกฎแบบหนึ่งในภาษา ATL ที่มักถูกนำมาใช้งานแบบ helper เนื่องจากเป็นการกำหนด rule แบบง่าย ๆ สามารถถูกนำมาเขียนเพื่อสร้าง object ลูกของ object หลักที่เกิดจากการทำ transformation ดังที่ใช้ในงานวิจัยนี้

Lazy rule เป็น rule ที่ถูกเรียกหรือกระทำได้มากกว่า 1 ครั้ง ขึ้นกับ rule ที่เรียก ซึ่งหมายถึง lazy rule จะทำงานก็ต่อเมื่อมีกีเรียกใช้จาก rule อื่นนั่นเอง การสร้าง lazy rule เริ่มด้วยคีย์เวิร์ด “lazy rule” ตามด้วยชื่อ ต่อด้วยการะบุต้นทางหรือ input ของข้อมูลด้วยคีย์เวิร์ด “from” และกำหนดปลายทางหรือ output ด้วยคีย์เวิร์ด “to”

ตัวอย่างการสร้าง lazy rule

```
lazy rule createEventSend{
     from   u: usdMM!Step
     to s: seqMM!Lifeline(name <- if
u.getMatchNounEqual()->size() > 0
          then
           u.getMatchNounEqual()->first()
     else if u.getMatchNounLike()->size() > 0
           thenu.getMatchNounLike()->first()
     else  u.Noun.value.trim()
endif
```

ดังตัวอย่างข้างต้นเป็นการสร้าง lazy rule ชื่อ createEventSend เพื่อสร้างลูกของ class Message โดยรับข้อมูล input เป็นชนิด usdMM!Step (class Step ที่อยู่ใน usecase metamodel) และสร้างข้อมูล output ไปเป็นชนิด seqMM!Lifeline (class Lifeline ที่อยู่ใน sequence metamodel) โดยค่าแอตทริบิวต์ “name” ของผลลัพธ์ปลายทาง ได้มากจากการทำงานของ helper “getMatchNounEqual” และ “getMatchNounLike” อีกทอดหนึ่งด้วย logic ที่กำหนด

Match Rule เป็นกฎที่นิยมเขียนเพื่อทำ transformation มากที่สุดในภาษา ATL เนื่องจากเป็นการทำ transformation ที่ตรงไปตรงมา คือ เฉพาะข้อมูลที่ตรงกับกฎหรือเงื่อนไขเท่านั้น จึงจะถูกสร้างเป็น output นั่นเอง match rule จะถูกกระทำเพียงครั้งเดียวในการ transformation 1 ครั้ง โดยไม่ต้องมีผู้เรียก (caller rule) ซึ่งเป็นจุดที่แตกต่างจาก lazy rule (ซึ่งจะต้องมีผู้เรียก) อย่างชัดเจน

```
rule Step2Message{
	from u: usdMM!Step
	to s: seqMM!Message(
		name <- u.getMatchVerb(),
		seq <- u.seq,
		ref <- u.ref,
		parentId <- u.parentId,
		sendEvent <- thisModule.createEventSend(u),
		receiveEvent <- thisModule.createEventReceive(u)
	)
}
```

ตัวอย่างการสร้าง match rule

จากตัวอย่างการสร้าง match rule เริ่มด้วย keyword “rule” (ไม่ต้องระบุคำว่า match) ตามด้วยชื่อของกฎที่ต้องการ หลังจากนั้นระบุขอบเขตเริ่มต้นและสิ้นสุดของกฎด้วยปีกกาเปิด และปีกกาปิด ภายในขอบเขตของระบุส่วนต้นทางหรือ input ด้วย keyword “from” ตามด้วย “ตัวแปร : ชนิดของข้อมูล” เช่น “from u : usdMM!Step” หมายถึงข้อมูลนำเข้าต้องเป็นชนิด usdMM!Step โดยกำหนดตัวแปรสำหรับอ้างถึงข้อมูลนำเข้านี้ชื่อว่า “u”.

ส่วนข้อมูลที่เป็นปลายทางหรือ output ระบุด้วย keyword “to” ตามด้วย “ตัวแปร : ชนิดของข้อมูล” เช่นกัน เช่น “to s : sqlMM!Message” หมายถึงข้อมูลส่งออกหรือผลลัพธ์จะเป็นชนิด sqlMM!Message โดยกำหนดตัวแปรสำหรับอ้างถึงข้อมูลส่งออกชื่อว่า “s” นั่นเอง

ในส่วนที่เป็น to จะระบุขอบเขตการทำ transformation ด้วยวงเล็บเปิดปิด และระบุข้อมูลปลายทางที่จะถูกสร้าง (โดยใช้ข้อมูลต้นทาง) ด้วยรูปแบบ “ข้อมูลปลายทาง <- ข้อมูลต้นทาง” ได้แก่

name <- u.getMatchVerb() : เป็นการระบุค่าของแอตทริบิวต์ name ของ Message ให้เท่ากับผลลัพธ์จากการทำงานของ helper getMatchVerb seq <- u.seq : เป็นการระบุค่าของแอตทริบิวต์ seq ของ Message ให้เท่ากับค่าแอตทริบิวต์ seq ชอง Step

- ref <- u.ref : เป็นการระบุค่าของแอตทริบิวต์ ref ของ Message ให้เท่ากับค่าแอตทริบิวต์ ref ชอง Step
- parentId <- u.parentId : เป็นการระบุค่าของแอตทริบิวต์ parentId ของ Message ให้เท่ากับค่าแอตทริบิวต์ parentId ชอง Step
- sendEvent <- thisModule.createEventSend(u) : เป็นการสร้างคลาส sendEvent ซึ่งเป็นคลาสลูกของคลาส Message โดยเรียกใช้ lazy rule createEventSend ซึ่งถูกอ้างถึงได้ด้วย keyword “thisModule”

- receiveEvent <- thisModule.createEventReceive(u) : เป็นการสร้างคลาส receiveEvent ซึ่งเป็นคลาสลูกของคลาส Message โดยเรียกใช้ lazy rule createEventReceive ซึ่งถูกอ้างถึงได้ด้วย keyword “thisModule”

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล

คำสำคัญ	รายละเอียด	ตัวอย่างการใช้งาน	ความหมาย
-- @path	ระบุที่อยู่ของไฟล์เมทาโมเดล	-- @path classMM=/ATLProject/metamodel /UML.ecore	ระบุ path ของ file classMM (class metamode) อยู่ที่ relative path /ATLProject/metamodel/UML.ecore
module	ระบุโมดูลของ atl ซึ่งชื่อโมดูลต้องเป็นชื่อเดียวกับไฟล์ atl เท่านั้น	module UsdAndClass2Seq;	Atl file นี้ชื่อโมดูลคือ UsdAndClass2Seq
create, from	ประกาศเจตจำนงการ transformation ของ atl นี้ โดยที่ create หมายถึงผลลัพธ์หรือ output ที่ต้องการ ส่วน from หมายถึง input ที่นำเข้าซึ่งสามารถมีได้มากกว่า 1 input โดยคั่นด้วยเครื่องหมายคอมม่า	create OUTSeq : seqMM from INUSD : usdMM, INClass : classMM;	ให้สร้างผลลัพธ์อัน conform ตาม OUTSeq metamodel โดยใช้ input สองตัวคือ usdMM (usecase metamodel) และ classMM (class metamodel).

ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล (ต่อ)

helper [context context_type]? def : helper_name(parameters) : return_type = exp;	การสร้าง helper หรือที่ในภาษาจาวาเรียกว่า method เพื่อสามารถเรียกใช้งานได้จากที่ต่าง ๆ เช่น rule หรือแม้แต่ helper ด้วยกันเอง โดยใช้ keyword helpder def:	helper context usdMM!Step def : getMatchNounEqual() : String = classMM!Class.allInstances()-> select(e | (e.name.trim().toUpper() = self.Noun.value.trim().toUpper())) ->collect(e | e.name);	สร้าง method ชื่อว่า getMatchNounEqual โดยรับ input parameter เป็นชนิด usdMM!Step และ return ผลลัพธ์กลับเป็น Set ของ String เช่นกัน
allInstances()	ดึงข้อมูลจากโมเดลและเก็บไว้ในรูปแบบชนิดข้อมูลที่เป็น collection	classMM!Class.allInstances()	ดึงข้อมูลที่เป็นชนิด classMM!Class ทั้งหมดจากโมเดล และเก็บไว้ในรูปแบบ collection
select	คำสั่งสำหรับคัดกรองเฉพาะข้อมูลที่ตรงตามเงื่อนไข	select(e | (e.name.trim().toUpper().startsWith(self.Noun.value.trim().toUpper())) or (e.name.trim().toUpper().endsWith(self.Noun.value.trim().toUpper())))	เลือก class โมเดลที่มีชื่อตรงกับข้อมูล input ซึ่งในที่นี้คือค่าที่เป็นคำนาม

ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล (ต่อ)

->collect	คำสั่งที่ใช้งานกับชนิดข้อมูล collection เพื่อเลือกดึงข้อมูล attribute ที่เฉพาะเจาะจงของโมเดล	->collect(e | e.name)	เป็นการเลือกดึงข้อมูล attribute “name” ของโมเดล
thisModule	Keyword สำหรับอ้างถึงตัวเองในระดับ global	thisModule.createEventSend(u),	เป็นการเรียกใช้ method createEventSend ภายใน module atl นั้น ๆ
self	Keyword สำหรับอ้างถึงตัวเองในระดับ local เช่น ระดับ method	helper context usdMM!Step def: getMatchVerb() : String = if self.getMatchVerbEqual()->size() > 0 ..... ;	self ที่เป็นการเรียกใน method นี้หมายถึงตัวแปรที่เป็น parameter ที่รับเข้ามาซึ่งหมายถึงตัวแปร usdMM!Step นั่นเอง

ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล (ต่อ)

lazy rule	เป็น rule ที่ทำหน้าที่คล้ายกับ helper ส่วนใหญ่สร้างเพื่อให้ match rule มาเรียกใช้อีกทอดหนึ่ง	lazy rule createEventSend{
		from
		u: usdMM!Step
		to
		s: seqMM!Lifeline(
		}
			ประกาศกฎชื่อว่า createEventReceive โดยรับ input เป็นชนิด usdMM!Step และส่งผลลัพธ์กลับเป็นชนิด seqMM!Lifeline

rule (match rule)	เป็น rule ที่ทำหน้าสำคัญในการระบุเงื่อนไขการเปลี่ยนรูปร่างของโมเดลจากต้นทางสู่ปลายทาง	rule Step2Message{ from u: usdMM!Step to s: seqMM!Message(name <- u.getMatchVerb(), seq <- u.seq, ref <- u.ref,	สร้าง rule ชื่อ Step2Message โดยรับข้อมูลเข้าเป็นชนิด usdMM!Step และส่งออกผลลัพธ์เป็นชนิด seqMM!Message ด้วยเงื่อนไขหรือกฎที่ระบุ เช่น attribute name ชองปลายทางได้มาจาก attribute value ชองคลาส Verb จากต้นทาง เป็นต้น.

ตารางที่ 16 แสดงโครงสร้างคำสั่งของภาษาเอทีแอล (ต่อ)

first	คำสั่งเพื่อดึงข้อมูลตัวแรกของชนิดข้อมูลที่เป็น collection	u.getMatchObjectLike()->first()	ดึงข้อมูล collection ตัวแรกที่ได้จากการประมวลผลของ helper “getMatchObjectLike”
trim	คำสั่งสำหรับตัดช่องว่างหน้าหลังของข้อมูลชนิด String	u.Object.value.trim()	ตัดช่องว่างหน้าหลังของข้อความที่ได้จาก u.Object.value
toUpper	คำสั่งสำหรับเปลี่ยนตัวอักษรให้เป็นตัวใหญ่ทั้งหมด	self.Noun.value.trim().toUpper()	เป็นข้อความในตัวแปร match ให้เป็นตัวใหญ่ทั้งหมดโดยตัดช่องว่างหน้าและหลังทิ้งด้วย
startsWith	คำสั่งเพื่อตรวจหาข้อมูลที่มีคำขึ้นต้นข้อความเหมือนกัน	e.name.trim().toUpper().startsWith (self.Noun.value.trim().toUpper())	ตรวจสอบว่าค่าของข้อมูล e.name มีคำขึ้นต้นด้วยข้อความที่ได้จากตัวแปร self.Noun.value.trim() จริงหรือไม่โดยเปรียบเทียบด้วยตัวอักขระตัวใหญ่ทั้งหมด
endsWith	คำสั่งเพื่อตรวจหาข้อมูลที่มีคำลงท้ายข้อความเหมือนกัน	e.name.trim().toUpper().endsWith (self.Noun.value.trim().toUpper()))	ตรวจสอบว่าค่าของข้อมูล e.name มีคำลงท้ายด้วยข้อความที่ได้จากตัวแปร (self.Noun.value.trim()จริงหรือไม่โดยเปรียบเทียบด้วยตัวอักขระตัวใหญ่ทั้งหมด

4.4.4 การพัฒนาวิธีการสร้างแผนภาพลำดับ

สำหรับการพัฒนาการแสดงจินตทัศน์เพื่อสนับสนุนไลบรารี่ของเครื่องมือ Sequence Editor ที่ได้เปรียบเทียบไว้ในตารางที่ 4 สามารถแบ่งลักษณะคำสั่งได้ 3 แบบ คือ

1. โครงสร้างคำสั่งที่ 1 คือ คำสั่งสำหรับการสร้าง Actor ซึ่งจะเขียนชื่อของ Actor ตามด้วยเครื่องหมาย “:” ตามด้วยคำว่า “Actor” เพื่อบ่งบอกความหมาย

```
ActorName : Actor
```

2. โครงสร้างคำสั่งที่ 2 คือ คำสั่งสำหรับการสร้าง Message ซึ่งจะเขียนจากชื่ออ็อปเจ็กต์ต้นทางคือ ObjectSender ตามด้วยเครื่องหมาย “:” แล้วเขียนชื่ออ็อปเจ็กต์ปลายทาง คือ ObjectReceiver ตามด้วยเครื่องหมาย “.” หลังจากนั้นให้เขียนข้อความเพื่อบ่งชี้ว่า Message นี้จะเริ่มต้นและสิ้นสุดที่ตำแหน่งใด

```
ObjectSender : ObjectReceiver.Message
```

3. โครงสร้างคำสั่งที่ 3 คือคำสั่งสำหรับการสร้างเงื่อนไข หรือข้อจำกัดในแผนภาพลำดับ กรณีที่มีการทำงานซ้ำเดิม ซึ่งจะสร้างแท็ก <C:LOOP> เพื่อบอกให้โปรแกรมทราบว่าเริ่มต้นเงื่อนไขนี้ที่ตำแหน่งใด และสร้างแท็ก </C:LOOP> เพื่อบอกให้โปรแกรมทราบว่าเงื่อนไขได้สิ้นสุดที่ตำแหน่งนั้น

```
<C:LOOP Condition>
    โครงสร้างคำสั่งที่ 2
    ....
</C:LOOP>
```

4. โครงสร้างคำสั่งที่ 4 คือคำสั่งสำหรับการสร้างเงื่อนไข หรือข้อจำกัดในแผนภาพลำดับ กรณีที่มีการทำงานที่มีตัวเลือกซึ่งจะสร้างแท็ก <C:IF> เพื่อบอกให้โปรแกรมทราบว่าเริ่มต้นเงื่อนไขนี้ที่ตำแหน่งใด และสร้างแท็ก </C:IF> เพื่อบอกให้โปรแกรมทราบว่าเงื่อนไขได้สิ้นสุดที่ตำแหน่งนั้น

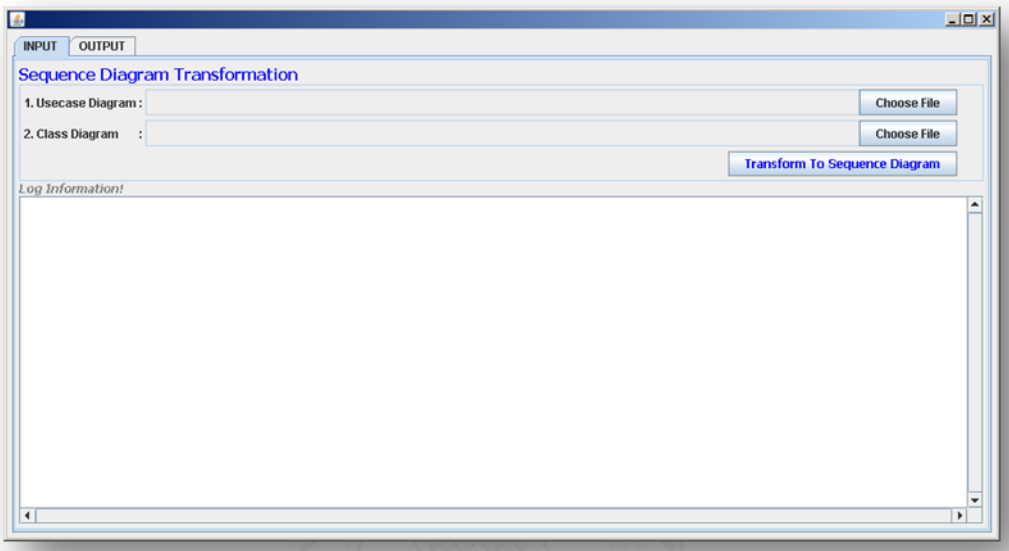
```
<C:IF Condition>
    โครงสร้างคำสั่งที่ 2
    ....
</C:IF>
```

4.4.5 การพัฒนาส่วนต่อประสาน

พัฒนาส่วนต่อประสานผู้ใช้ให้สอดคล้องกับขอบเขตของระบบจากบทที่ 1 และมีการทำงานตรงกับตามความต้องการด้านหน้าที่ที่กำหนดไว้ในบทที่ 4 ในการพัฒนาส่วนต่อประสานจะใช้ไลบรารี

สำหรับสร้างแผนภาพลำดับ (sequence editor) และ สวิง (SWING) ในการพัฒนา โดยใช้โปรแกรมอีคลิปส์ในการพัฒนา

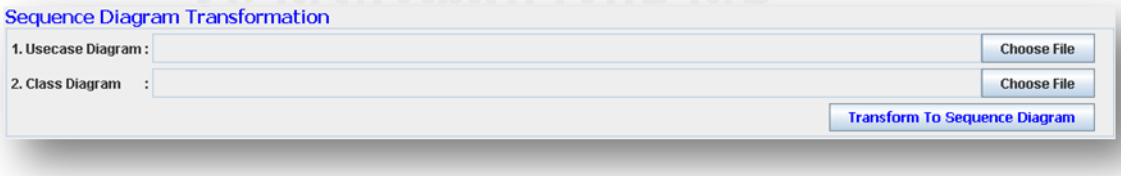
เมื่อเปิดระบบการสร้างแผนภาพลำดับด้วยเทคโนโลยีการแปลงตัวแบบ ระบบจะแสดงหน้าจอ ดังรูปที่ 21 ซึ่งระบบจะแบ่งหน้าจอการทำงานหลักออกเป็น 2 ส่วน คือ ส่วนของการประมวลผลข้อมูลนำเข้า (input) และส่วนแสดงผลลัพธ์ของการประมวลผลข้อมูลนำเข้า (output)



รูปที่ 21 หน้าจอของระบบการสร้างแผนภาพลำดับ

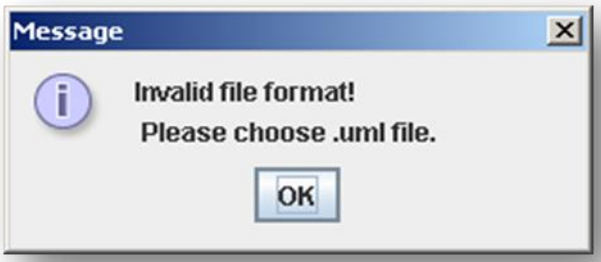
4.4.5.1 ส่วนของการประมวลผลข้อมูลนำเข้า

ส่วนนี้เป็นส่วนที่ผู้ใช้จะต้องเป็นผู้เลือกไฟล์เอกซ์เอ็มไอเพื่อนำเข้าข้อมูลของแผนภาพยูสเคสประกอบคำอธิบายและแผนภาพคลาส ดังรูปที่ 22



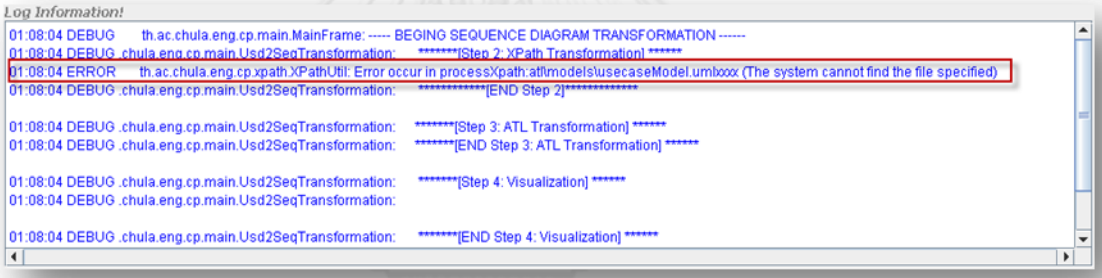
รูปที่ 22 หน้าจอการนำเข้าไฟล์เอกซ์เอ็มไอ

กรณีที่ผู้ใช้ทำการเลือกไฟล์นอกเหนือข้อกำหนดของระบบ ระบบจะทำการแจ้งเตือนเป็น Popup message ดังรูปที่ 23



รูปที่ 23 หน้าจอแสดงผลกรณีการนำเข้าไฟล์ผิดพลาด

เมื่อผู้ใช้ทำการกดปุ่มเพื่อประมวลผล ระบบจะแสดงหน้าจอเพื่อแจ้งสถานะการทำงานของโปรแกรมว่าดำเนินการถึงขั้นตอนใด และแจ้งเตือนกรณีที่พบข้อผิดพลาดของโปรแกรม ดังรูปที่ 24

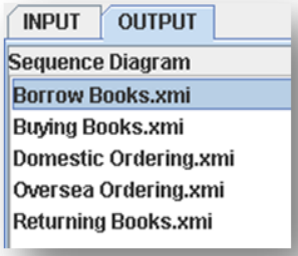


รูปที่ 24 หน้าจอแสดงผลกรณีพบข้อผิดพลาดในการประมวลผลของระบบ

4.4.5.2 ส่วนการแสดงผลลัพธ์การประมวลผลโปรแกรม

ส่วนแสดงผลลัพธ์การประมวลผลโปรแกรมนี้จะแบ่งเป็น 2 ส่วน คือ

1) ส่วนที่แสดงรายการของแผนภาพลำดับทั้งหมด โดยจะตามจำนวนของรายการของแผนภาพลำดับตามที่โปรแกรมประมวลผลได้ ซึ่งชื่อของแผนภาพลำดับจะเหมือนที่กำหนดไว้ในข้อมูลนำเข้าของแผนภาพยูสเคสประกอบคำอธิบาย ดังรูปที่ 25



รูปที่ 25 หน้าจอแสดงรายการแผนภาพลำดับ

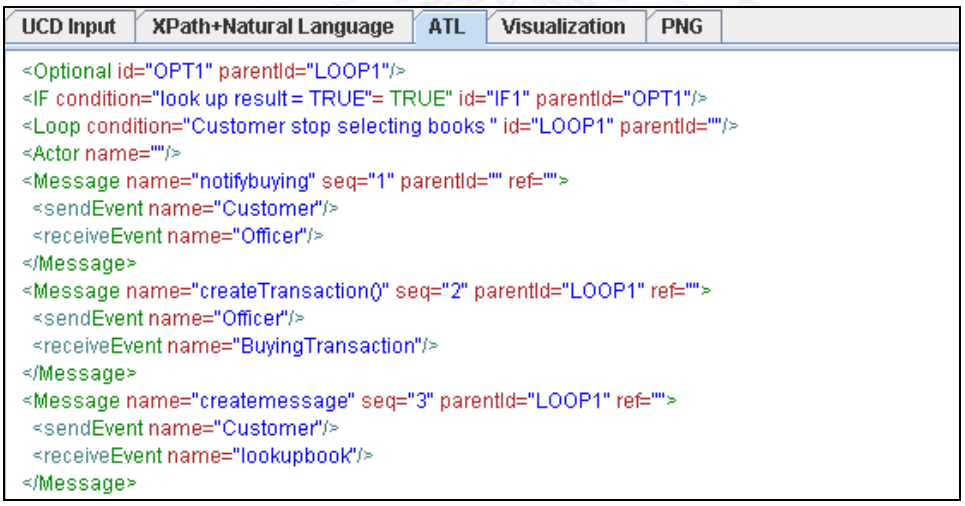
2) ส่วนที่แสดงผลลัพธ์แต่ละขั้นตอนของแต่ละแผนภาพลำดับ ซึ่งจะแสดงในรูปแบบแท็บ จากรูปที่ 26 ตัวอย่าง การแสดงผลที่ได้จากขั้นตอนการแปลงแบบโดยภาษาเอทีแอลจะประกอบด้วย 4 แท็บ คือ

- แท็บ XPath+Natural Language จะแสดงผลที่ได้จากขั้นตอนการนำเทคโนโลยี Xpath มาดึงคำจากไฟล์เอกซ์เอ็มไอและการทำงานโดยไลบรารี่

- แท็บ ATL จะแสดงผลที่ได้จากขั้นตอนการแปลงแบบโดยภาษาเอทีแอล

- แท็บ Visualization จะแสดงผลที่ได้จากขั้นตอนการแปลงรูปแบบเพื่อสนับสนุนให้สามารถใช้งานบนโปรแกรม Quick Sequence Diagram ได้

- แท็บ PNG จะแสดงแผนภาพลำดับทั้งหมดที่ได้จากการประมวลผลของระบบ



รูปที่ 26 หน้าจอแสดงผลการทำงานที่ได้จากการแปลงแบบโมเดล

บทที่ 5
การทดสอบและประเมินผลระบบ

5.1 การทดสอบระบบ

การทดสอบระบบเป็นการทดสอบหน้าที่การทำงานของระบบตามที่ได้ระบุไว้ในหัวข้อความต้องการเชิงหน้าที่ในบทที่ 4 ซึ่งการทดสอบระบบนี้ใช้วิธีการทดสอบหน้าที่การทำงาน (Black Box Testing) ตามกรณีทดสอบที่ได้ออกแบบไว้ ซึ่งมีการทดสอบ และผลการทดสอบที่ได้ ดังต่อไปนี้

1) ทดสอบการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ (ตารางที่ 17)

1.1) ทดสอบการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอจากแผนภาพยูสเคสประกอบคำอธิบาย

1.2) ทดสอบการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอจากแผนภาพคลาส

2) ทดสอบกฎการสร้างแผนภาพลำดับ

2.1) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี actor เป็นส่วนประกอบ

2.2) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี message เป็นส่วนประกอบ

2.3) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี lifeline เป็นส่วนประกอบ

2.4) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี object (sender) เป็นส่วนประกอบ

2.5) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี object (receiver) เป็นส่วนประกอบ

2.6) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี fragment (alternative) เป็นส่วนประกอบ (ตารางที่ 18)

2.7) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี fragment (option) เป็นส่วนประกอบ

2.8) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี fragment (loop) เป็นส่วนประกอบ

2.9) ทดสอบกฎการสร้างแผนภาพลำดับกรณีที่มี fragment (reference) เป็นส่วนประกอบ

3) ทดสอบการส่งออกผลลัพธ์ที่ได้จากกระบวนการแปลงแบบโมเดล (ตารางที่ 19)

3.1) ทดสอบการส่งออกในรูปแบบเอกซ์เอ็มไอ

3.2) ทดสอบการส่งออกในรูปแบบไฟล์รูปภาพ

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ตารางที่ 17 การทดสอบการการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
TC01	การนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ	นำเข้าไฟล์เอกสารเอกซ์เอ็มไอของแผนภาพส่วนประกอบ	1) แสดงชื่อไฟล์เอกสารเอกซ์เอ็มไอที่หน้าต่างแสดงรายการไฟล์เอกสารเอกซ์เอ็มไอ	ถูกต้อง
			2) แสดงข้อความระบุชื่อของไฟล์เอกซ์เอ็มไอนำเข้าได้ถูกต้อง	ถูกต้อง
			3) แจ้งเตือนกรณีที่ไม่สามารถนำเข้าเอกสารเอกซ์เอ็มไอได้	ถูกต้อง
TC02	การนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ	นำเข้าไฟล์เอกสารเอกซ์เอ็มไอของแผนภาพคลาส	1) แสดงชื่อไฟล์เอกสารเอกซ์เอ็มไอที่หน้าต่างแสดงรายการไฟล์เอกสารเอกซ์เอ็มไอ	ถูกต้อง
			2) แสดงข้อความระบุชื่อของไฟล์เอกซ์เอ็มไอนำเข้าได้ถูกต้อง	ถูกต้อง
			3)แจ้งเตือนกรณีที่ไม่สามารถนำเข้าเอกสารเอกซ์เอ็มไอได้	ถูกต้อง

ตารางที่ 18 ทดสอบกฎการสร้างแผนภาพลำดับ

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
TC03	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี actor เป็น	ในแผนภาพลำดับมี actor เป็นส่วนประกอบ กรณีที่ระบบนั้นเกี่ยวข้องกับผู้ใช้	ถูกต้อง

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
		ส่วนประกอบในแผนภาพลำดับ	ระบบ	
TC04	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี message เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี message เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น message	ถูกต้อง
TC05	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี lifeline เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี lifeline เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น lifeline	ถูกต้อง
TC06	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี object ฝั่งที่เป็น sender เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี object ฝั่งที่เป็น sender เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น object	ถูกต้อง
TC07	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี object ฝั่งที่เป็น receiver เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี object ฝั่งที่เป็น receiver เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น object	ถูกต้อง
TC08	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี fragment แบบ alternative เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี fragment แบบ alternative เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น fragment แบบ	ถูกต้อง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
			alternative	
TC09	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี fragment แบบ option เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี fragment แบบ option เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น fragment แบบ option	ถูกต้อง
TC10	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี fragment แบบ loop เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี fragment แบบ loop เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น fragment แบบ loop	ถูกต้อง
TC11	การสร้างแผนภาพลำดับ	เพื่อทดสอบการสร้างแผนภาพลำดับกรณีที่มี fragment แบบ reference เป็นส่วนประกอบในแผนภาพลำดับ	ในแผนภาพลำดับมี fragment แบบ reference เป็นส่วนประกอบ กรณีที่ระบบนั้นเป็นไปตามเงื่อนไขตามกฎการแปลงแบบเป็น fragment แบบ reference	ถูกต้อง

ตารางที่ 19 ทดสอบการส่งออกไฟล์ที่ได้จากการประมวลผล

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
TC12	ทดสอบการส่งออกไฟล์ที่ได้จากการ	เพื่อทดสอบการส่งออกไฟล์ที่ได้จากการประมวลผลในรูปแบบ	1).ระบบสามารถสร้างเอกสารในรูปแบบเอกซ์เอ็มไอได้	ถูกต้อง

หมายเลขกรณีทดสอบ	การทดสอบ	คำอธิบาย	ผลการทดสอบที่คาดหวัง	ผลการทดสอบจริง
	ประมวลผลในรูปแบบเอกซ์เอ็มไอ	เอกซ์เอ็มไอ	2).เอกสารเอกซ์เอ็มไอแสดงรายการส่วนประกอบของแผนภาพยูสเคสประกอบคำอธิบายได้ครบถ้วน	ถูกต้อง
TC13	ทดสอบการส่งออกไฟล์ที่ได้จากการประมวลผลในรูปแบบไฟล์รูปภาพ	เพื่อทดสอบการส่งออกไฟล์ที่ได้จากการประมวลผลในรูปแบบไฟล์รูปภาพ	1).ระบบสามารถส่งออกในรูปแบบไฟล์รูปภาพได้	ถูกต้อง
			2).ระบบขึ้นป๊อปอัพเพื่อให้ผู้ใช้เป็นผู้เลือกไปยังตำแหน่งที่ต้องการเก็บไฟล์รูปภาพนั้นๆได้	ถูกต้อง
			3).เมื่อไปที่ตำแหน่งที่เลือกไฟล์เพื่อจัดเก็บรูปภาพจะต้องเห็นไฟล์รูปภาพ	ถูกต้อง

5.2 การประเมินผลระบบ

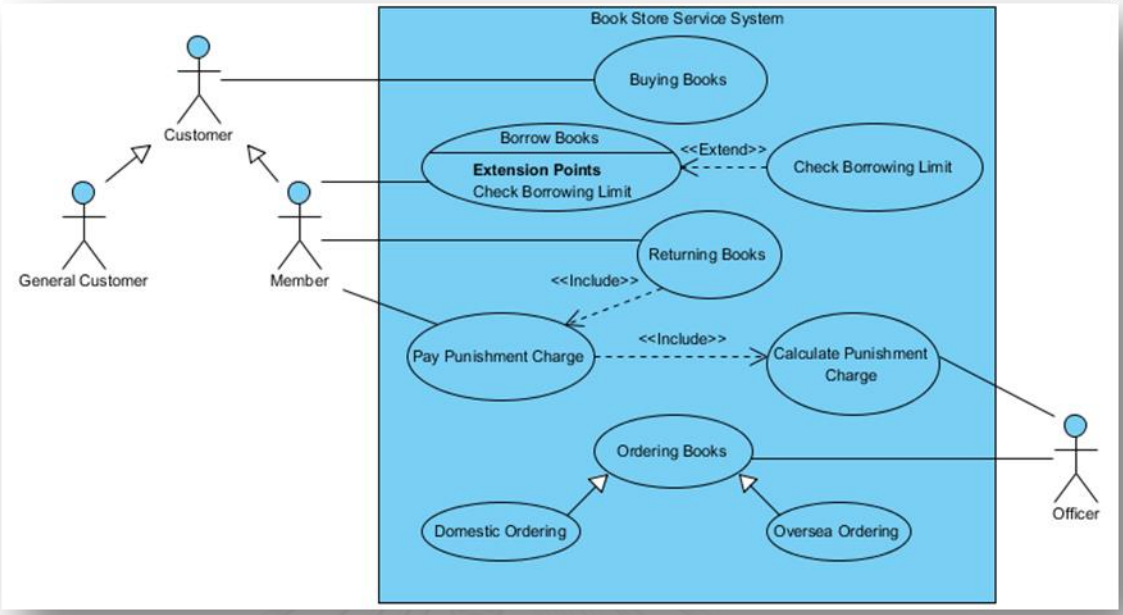
การประเมินผลระบบจะประเมินโดยทดสอบระบบการจัดการร้านหนังสือ [11] สามารถอธิบายได้ดังนี้

5.2.1 ระบบการจัดการร้านหนังสือ

- ข้อมูลนำเข้า

1).แผนภาพยูสเคสประกอบคำอธิบาย

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



รูปที่ 27 แผนภาพยูสเคสของระบบการจัดการร้านหนังสือ

แผนภาพยูสเคสของระบบการจัดการร้านหนังสือ ประกอบด้วย 9 ยูสเคส คือ

1. ยูสเคสการซื้อหนังสือ (Buying Books)
2. ยูสเคสการยืมหนังสือ (Borrow Books)
3. ยูสเคสการคืนหนังสือ (Return Books)
4. ยูสเคสการสั่งหนังสือ (Ordering Books)
5. ยูสเคสการสั่งหนังสือที่ผลิตภายในประเทศ (Domestic Ordering)
6. ยูสเคสการสั่งหนังสือที่ผลิตจากต่างประเทศ (Oversea Ordering)
7. ยูสเคสการตรวจสอบยอดจำนวนหนังสือที่สามารถยืมได้ (Check Borrowing Limit)
8. ยูสเคสการจ่ายเงินค่าปรับการยืมหนังสือ (Pay Punishment Charge)
9. ยูสเคสการคำนวณยอดค่าปรับหนังสือ (Calculate Punishment Charge)

จากรูปที่ 27 สามารถเขียนคำอธิบายแผนภาพยูสเคสหลักๆ ได้ 5 ยูสเคส คือ

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

5.2.1.1 ยูสเคสการซื้อหนังสือ

คำอธิบายแผนภาพยูสเคสการซื้อหนังสือ สามารถอธิบายได้ ดังนี้

1. ลูกค้า (Customer) แจ้งความจำนงว่าต้องการซื้อหนังสือกับเจ้าหน้าที่ (Officer)
2. เจ้าหน้าที่ทำการสร้างรายการขายหนังสือ (Buying Transaction)
3. เจ้าหน้าที่ (Officer) ดำเนินการหาหนังสือที่ลูกค้าต้องการโดยส่งข้อมูลเพื่อทำการค้นหา "lookup" ไปหาหนังสือ (Book) เล่มที่ต้องการ
4. ข้อมูลรหัสของหนังสือ (Book) ถูกส่งกลับมายังเจ้าหน้าที่ (Officer) หากการค้นหาไม่ประสบผลสำเร็จเจ้าหน้าที่จะไม่ได้รับข้อมูลรหัสของหนังสือ
5. ถ้าผลลัพธ์ในการค้นหาในข้อ 3 ประสบผลสำเร็จ รหัสหนังสือที่กำลังจะขายจะถูกระบุ (“setBookId”) ไปในรายการการขาย (BuyingTransaction)
6. เจ้าหน้าที่กำหนดมูลค่าของการขาย (“setPrice”) โดยการเรียกข้อมูลไปยังรายการการขาย (Buying Transaction)
7. ลูกค้าจะจ่ายเงินและปิดการขาย ("proceed") ครั้งนั้นโดยการแจ้งข้อมูลไปยังรายการการขาย (Buying Transaction)
8. กระบวนการที่ 2 ถึง 7 จะเกิดขึ้นซ้ำ ๆ (Loop) เท่ากับจำนวนหนังสือที่ถูกเลือกซื้อโดยลูกค้า (Customer stop selecting books)

คำอธิบายแผนภาพยูสเคสการซื้อหนังสือที่สร้างจากโปรแกรมวิชวลพาราดามจะเป็นดังตารางที่ 20

ตารางที่ 20 คำอธิบายยูสเคสการซื้อหนังสือ

Name	Value
Name	Buying Books
Use Case ID	UCD01
Rank	High
Primary Actors	Customer

Flow of Events

1. Customer notify buying to Officer.
2. loop
2.1. Officer create transaction to Buying transaction.

2.2. Customer notify buying to Officer.
2.3. Officer create transaction to Buying transactions.
2.4. Officer is lookup from book.
2.5. The book get book id from book profile.
2.6. The book get book id return to officer.
2.7. if look up result = TRUE
2.7.1. Officer set book id in Buying transaction.
2.7.2.Officer set price in Buying transaction.
end if
2.8. Customer proceed to Buying transactions.
until Customer stop selecting books [loop]

5.2.1.2 ยูสเคสการยืมหนังสือ

คำอธิบายแผนภาพยูสเคสการยืมหนังสือ สามารถอธิบายได้ ดังนี้

1. สมาชิก (Member) แจ้งความจำนงว่าต้องการยืมหนังสือแก่เจ้าหน้าที่ (Officer)
2. เจ้าหน้าที่ทำการสร้างรายการยืมหนังสือ (Borrow Transaction)
3. เจ้าหน้าที่ดำเนินการหาหนังสือที่สมาชิกต้องการโดยการส่งข้อมูลเพื่อไปค้นหา (“lookup”) หนังสือ (Book) ที่ต้องการ
4. ข้อมูลรหัสของหนังสือ (Book) ที่ต้องการจะถูกส่งกลับมายังเจ้าหน้าที่ หากการค้นหาไม่ประสบผลสำเร็จเจ้าหน้าที่จะไม่ได้รับข้อมูลรหัสของหนังสือ (“bId”)
5. ถ้าผลลัพธ์ในการค้นหาในข้อ 3 ประสบผลสำเร็จ รหัสหนังสือ (“bookId”) ที่กำลังจะยืมจะถูกระบุ (“setBookId”) ไปในรายการการขาย (BorrowTransaction)
6. เจ้าหน้าที่ทำการค้นหาจำนวนวันเพื่อกำหนดวันคืนหนังสือโดยการขอข้อมูล (“getBorrowDayLimit”) ไปหาแหล่งเก็บข้อมูลหนังสือ (Book Profile) เพื่อหาจำนวนวันสูงสุด (“dayLimit”) ที่สมาชิกสามารถยืมหนังสือได้
7. เจ้าหน้าที่กำหนดวันคืนหนังสือ โดยกำหนดวันที่ครบกำหนดคืนหนังสือ ("setExpireDate") ให้กับรายการยืมหนังสือ โดยวันที่กำหนดคือ วันปัจจุบันนับไปอีกเท่ากับจำนวนที่สูงสุดที่สามารถยืมหนังสือได้
8. สมาชิกทำรายการยืมสำเร็จ (“proceed”) ในครั้งนั้น โดยการแจ้งข้อมูลไปยังรายการยืมหนังสือ

9. กระบวนการที่ 2 ถึง 8 จะเกิดขึ้นซ้ำ ๆ เท่ากับจำนวนหนังสือที่ถูกเลือกโดยสมาชิกแลต้องน้อยกว่าจำนวนสูงสุดที่สมาชิกสามารถยืมได้

คำอธิบายของแผนภาพยูสเคสการยืมหนังสือที่สร้างจากโปรแกรมวิชวลพาราดามจะเป็นดังตารางที่ 21

ตารางที่ 21 คำอธิบายยูสเคสการยืมหนังสือ

Name	Value
Name	Borrow Books
Use Case ID	2
Rank	High
Primary Actors	Member

Flow of Events

1.Member notify borrow to Officer
2. loop
2.1.Officer create Buying transaction
2.2.Officer call look up Book
2.3.Book get book id from Book profile
2.4.Book return result to Officer
2.5. if Look up result = TRUE
2.5.1.Officer set price in Buying transaction
2.5.2.Officer get borrow date limit from Book profile
2.5.3.Officer set expire date
end if
2.6.Member proceed to Borrow Transaction
until less than borrowing limit [loop]

5.2.1.3 ยูสเคสการคืนหนังสือ

คำอธิบายแผนภาพยูสเคสการคืนหนังสือ สามารถอธิบายได้ ดังนี้

1. สมาชิก (Member)แจ้งความจำนงว่าต้องการคืนหนังสือไปยังเจ้าหน้าที่ (Officer)
2. เจ้าหน้าที่ทำการสร้างรายการของการคืนหนังสือ (ReturnTransaction)
3. เมื่อสมาชิกแสดงหนังสือที่ยืมไปแล้ว จะมีการจำลองการสร้างอ็อปเจ็คของหนังสือที่ถูกยืมไปในรายการยืมหนังสือ (Borrow Transaction)
4. ในรายการยืมหนังสือจะมีการส่งข้อมูลรหัสรายการหนังสือที่ยืมไปในแหล่งเก็บรายละเอียดหนังสือ (Book Profile)
5. แหล่งเก็บรายละเอียดหนังสือจะส่งค่ากลับไปให้รายการยืมหนังสือ
6. จากผลลัพธ์ที่ได้ในข้อ 4 ทำการจะกำหนดค่ารหัสหนังสือ ("setBookId") ไปยังรายการคืนหนังสือ (ReturnTransaction) เพื่อระบุรหัสหนังสือที่ถูกยืมไป
7. รายการคืนหนังสือจะค้นหา (“findTransaction”) รายการข้อมูลของการยืมหนังสือเล่มนี้ไปยังรายการยืมหนังสือ
8. รายการคืนหนังสือจะกำหนดวันคืนหนังสือ (“setReturnDate”) ให้กับตัวเอง
9. รายการคืนหนังสือจะค้นหาวันที่ครบกำหนดในการคืนหนังสือ (“getExpireDate”) ไปยังรายการยืมหนังสือ
10. ในกรณีที่วันปัจจุบันเลยจากวันที่ครบกำหนดคืนหนังสือ รายการคืนหนังสือจะทำการสร้างอ็อปเจ็คของการจ่ายค่าปรับ (Punishment Charge)
11. รายการคืนหนังสือดึงอัตราค่าปรับ ("getChargeRate") ของหนังสือไปยังแหล่งเก็บรายละเอียดหนังสือ
12. ทำการสร้างอ็อปเจ็คการคำนวณค่าปรับ (Charge Calculator) ซึ่งจะทำหน้าที่คำนวณค่าปรับ
13. ในรายการยืมหนังสือจะทำการคำนวณค่าปรับ (“calculateChartAmout”) โดยคำนวณจากวันที่เกินกำหนดและอัตราค่าปรับของหนังสือเล่มนั้น ๆ
14. ผลลัพธ์ที่ได้จากการคำนวณในข้อ 13 จะถูกกำหนดค่า (“setAmount”)เข้าไปในอ็อปเจ็คของการจ่ายค่าปรับ
15. สมาชิกดำเนินการจ่ายค่าปรับและปิดรายการคืนหนังสือ (“proceed”) ด้วยการส่งข้อมูลไปหยุดการทำงานของการจ่ายค่าปรับของหนังสือเล่มนั้น
16. กระบวนการที่ 4 ถึง 15 จะเกิดขึ้นซ้ำ ๆ อีก เท่ากับจำนวนหนังสือที่ถูกนำมาคืน
17. เจ้าหน้าที่แจ้งไปที่สมาชิกว่าการดำเนินการคืนหนังสือเรียบร้อย

คำอธิบายของแผนภาพยูสเคสการคืนหนังสือที่สร้างจากโปรแกรมวิชวลพาราดามจะเป็นดังตารางที่ 22

ตารางที่ 22 คำอธิบายยูสเคสการคืนหนังสือ

Name	Value
Name	Returning Books
Use Case ID	4
Rank	High
Primary Actors	Member

Flow of Events

1.Member notify ('return') to Officer
2.Officer create Return transaction
3.Officer get book id from Book profile
4. loop
4.1.Officer set book id to Return Transaction
4.2.Return Transaction history look up in Borrow Transaction
4.3.Return Transaction set return date (Today)
4.4.Return Transaction get expire date from Borrow Transaction
4.5. if Expire date less than Today
4.5.1.Officer create Punishment Charge
4.5.2.Return Transaction get charge daily rate from Book
4.5.3.Borrow Transaction calculate charge amount from Charge calculator
4.5.4.Charge calculator set amount to Punishment charge
4.5.5.Member proceed to Borrow Transaction
end if
4.6.Officer notify ('complete') to Member
until look up result empty [loop]

5.2.1.4 ยูสเคสการสั่งหนังสือในประเทศ

คำอธิบายแผนภาพยูสเคสการสั่งหนังสือในประเทศ สามารถอธิบายได้ ดังนี้

1. เจ้าหน้าที่สร้างรายการสั่งซื้อหนังสือเพื่อสร้างอ็อปเจ็คของรายการสั่งหนังสือ (Book Order)

2. ค้นหารายละเอียดของหนังสือที่ต้องการสั่งซื้อโดยการส่งไปค้นหา ("lookup") ไปยัง แหล่งเก็บข้อมูลหนังสือ (BookProfile) เพื่อค้นดูว่าเคยสั่งหนังสือปกนี้เข้ามาก่อนหน้านี้หรือไม่ ถ้าไม่เคยสั่งซื้อเข้ามาก่อน(lookupResult = Not Found) ให้สร้างอ็อปเจ็คในแหล่งเก็บข้อมูลหนังสือขึ้นใหม่
3. เจ้าหน้าที่จะกำหนดวันที่ซื้อหนังสือ ("setOderDate") โดยส่งข้อมูลไปยังรายการหนังสือ
4. เจ้าหน้าที่จะกำหนดมูลค่าของการสั่งซื้อ (“setOrderPrice") โดยส่งข้อมูลไปยังรายการหนังสือ
5. เจ้าหน้าที่ระบุรายละเอียดของหนังสือที่จะสั่งซื้อ ("setBookProfile") ให้แก่รายการสั่งหนังสือ
6. เจ้าหน้าที่ปิดการสั่งซื้อหนังสือ (“proceed”) ที่สั่งภายในประเทศไปที่รายการสั่งหนังสือ

คำอธิบายแผนภาพยูสเคสการสั่งหนังสือภายในประเทศที่สร้างจากโปรแกรมวิชวลพาราดามจะเป็นดังตารางที่ 23

ตารางที่ 23 คำอธิบายยูสเคสการสั่งหนังสือภายในประเทศ

Name	Value
Name	Domestic Ordering
Use Case ID	8
Rank	High
Primary Actor	Officer

Flow of Events

1.Officer create Book order
2.Book order look up from Book profile
3. if look up result = Not Found
3.1.Book order create new Book profile
end if
4.Officer set order date (Today) in Book order
5.Officer set order price in Book order
6.Officer set book profile in Book order
7.Officer proceed to Book order

5.2.1.5 ยูสเคสการสั่งหนังสือจากต่างประเทศ

คำอธิบายแผนภาพยูสเคสการสั่งหนังสือจากต่างประเทศ สามารถอธิบายได้ ดังนี้

1. เจ้าหน้าที่สร้างรายการสั่งซื้อหนังสือเพื่อสร้างอ็อปเจ็คของรายการสั่งหนังสือ (Book Order)
2. ค้นหารายละเอียดของหนังสือที่ต้องการสั่งซื้อโดยการส่งไปค้นหา ("lookup") ไปยัง แหล่งเก็บข้อมูลหนังสือ (BookProfile) เพื่อค้นดูว่าเคยสั่งหนังสือปกนี้เข้ามาก่อนหน้านี้หรือไม่ ถ้าไม่เคยสั่งซื้อเข้ามาก่อน(lookupResult = Not Found) ให้สร้างอ็อปเจ็คในแหล่งเก็บข้อมูลหนังสือขึ้นใหม่
3. เจ้าหน้าที่จะกำหนดวันที่ซื้อหนังสือ ("setOderDate") โดยส่งข้อมูลไปยังรายการสั่งหนังสือ
4. เจ้าหน้าที่จะกำหนดมูลค่าของการสั่งซื้อ (“setOrderPrice") โดยส่งข้อมูลไปยังรายการสั่งหนังสือ
5. เจ้าหน้าที่ระบุรายละเอียดของประเทศต้นทาง ("setOrderCountry") การนำเข้าให้แก่รายการสั่งหนังสือ
6. เจ้าหน้าที่ระบุรายละเอียดของหนังสือที่จะสั่งซื้อ ("setBookProfile") ให้แก่รายการสั่งหนังสือ
7. เจ้าหน้าที่ดำเนินการจ่ายภาษี (“payTax”) โดยการ ไปยังรายการสั่งหนังสือ
8. เจ้าหน้าที่ปิดการสั่งซื้อหนังสือ (“proceed”) ที่สั่งภายในประเทศไปที่รายการสั่งหนังสือ

คำอธิบายรายละเอียดของแผนภาพยูสเคสการสั่งหนังสือจากต่างประเทศที่สร้างจากโปรแกรมวิชวลพาราดามจะเป็นดังตารางที่ 24

ตารางที่ 24 คำอธิบายยูสเคสการสั่งหนังสือจากต่างประเทศ

Name	Value
Name	Oversea Ordering
Use Case ID	9
Rank	High
Primary Actor	Officer

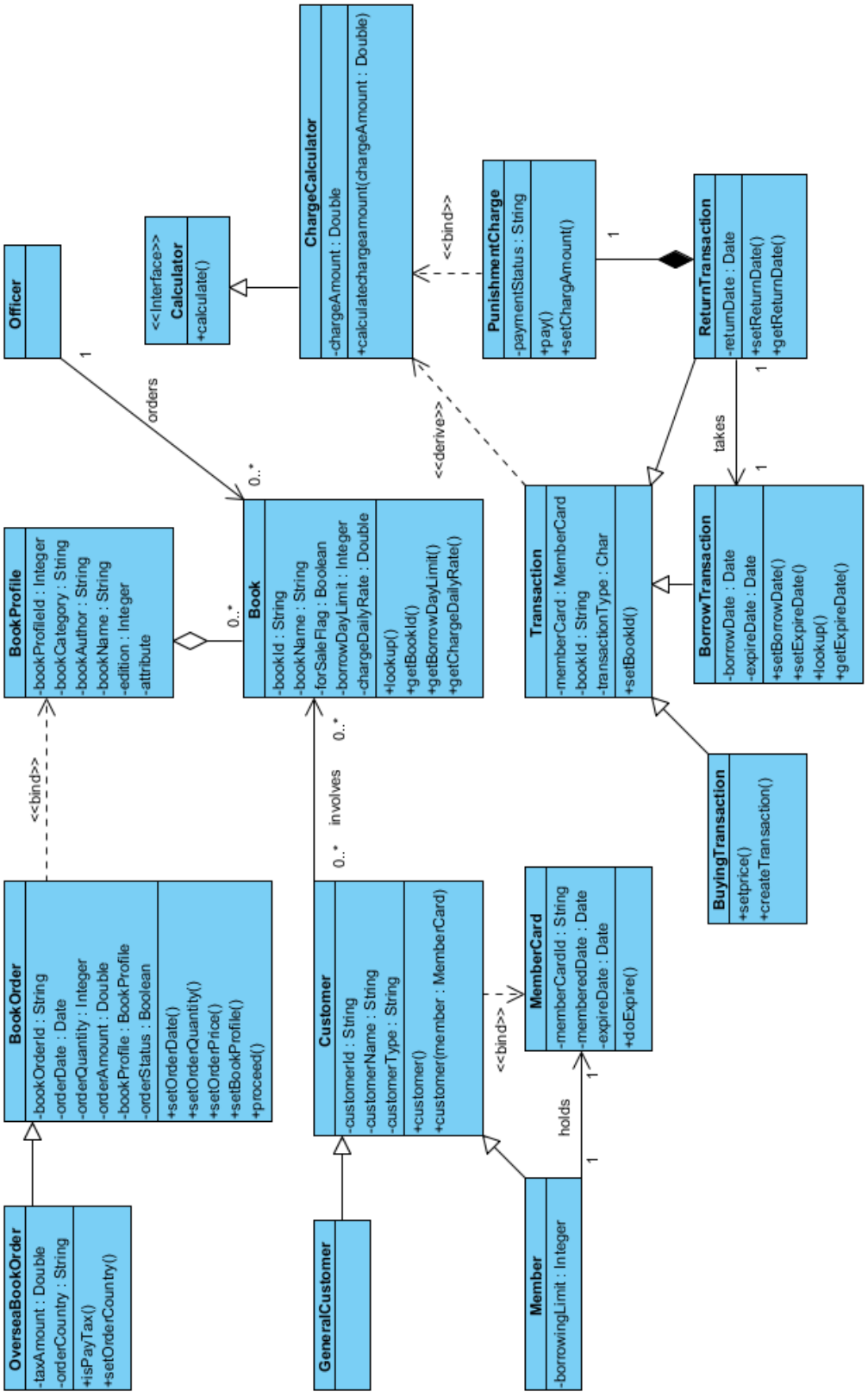
Flow of Events

1.Officer create book order
2.Book order look up from book profile
3. if look up result = Not Found
3.1.Book order create new book profile
end if

4.Officer set order date (Today) in book order
5.Officer set order country in book order
6.Officer set order price in book order
7.Officer set book profile in book order
8.Officer pay tax of book order
9.Officer proceed to book order

2). แผนภาพคลาส สามารถเขียนแผนภาพคลาสได้ดังรูปที่ 28

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



รูปที่ 28 แผนภาพคลาสของระบบการจัดการร้านหนังสือ

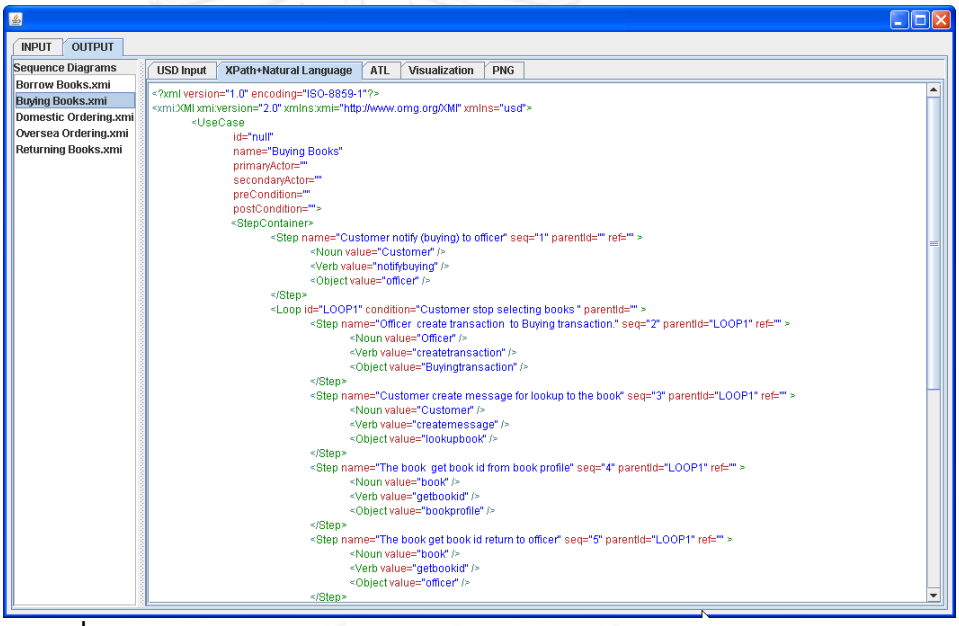
ผลการทดสอบของระบบ

1) ผลการทดสอบยูสเคสการซื้อหนังสือ

1.1 ข้อมูลจากคำอธิบายยูสเคสการซื้อหนังสือ ระบบสามารถดึงข้อมูลมาแล้วสามารถพัฒนาร่วมกับภาษาธรรมชาติ เพื่อให้ได้ผลลัพธ์ตามโครงสร้างเมทาโมเดลต้นทางที่ต้องการได้ดังรูปที่ 29 ซึ่งแต่ละแท็กขั้นตอนการทำงาน (step) ซึ่งจะแทนด้วยลำดับขั้นตอนการทำงานของแต่ละกิจกรรม และแท็กเงื่อนไขซึ่งจะแทนด้วยเงื่อนไขการเกิดเหตุการณ์ตามลำดับดังที่ระบุใน

1.2

1.3 ตารางที่ 20 จากรูปที่ 29 ประกอบด้วยแท็กลำดับขั้นตอนการทำงาน 10 ขั้นตอน และแท็กเงื่อนไข 2 เงื่อนไข



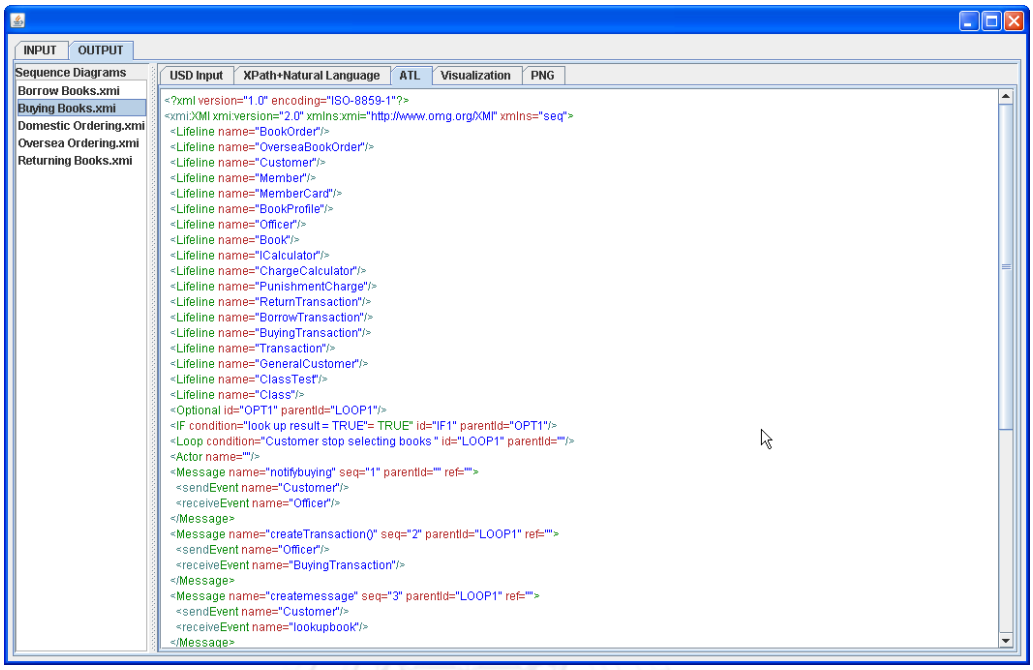
รูปที่ 29 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการซื้อหนังสือ

1.2) เมื่อข้อมูลคำอธิบายยูสเคสการซื้อหนังสือถูกสกัดให้อยู่ในรูปแบบเมทาโมเดลต้นทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นการประมวลผลโดยภาษาเอทีแอลซึ่งจะได้ผลลัพธ์ดังรูปที่ 30 ซึ่งผลลัพธ์ที่ได้จะอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับ

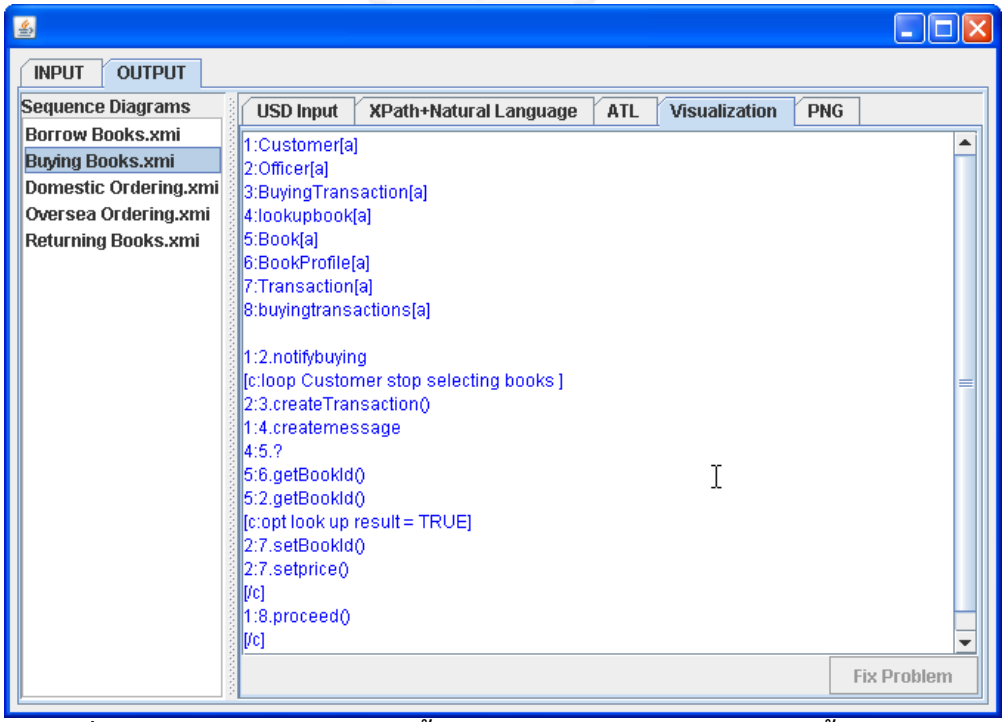
1.3) เมื่อผลลัพธ์ยูสเคสการซื้อหนังสืออยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับเรียบร้อยแล้ว เพื่อสนับสนุนให้การแสดงผลเป็นแผนภาพลำดับ จะต้องทำการแปลงค่าให้อยู่ในโครงสร้างภาษาที่โปรแกรม Quick Sequence Editor กำหนด จากผลการแปลงค่าของยูสเคสนี้จะได้ผลลัพธ์ดังรูปที่ 31สามารถนำผลลัพธ์ที่ได้ไปเปิดที่โปรแกรม Quick Sequence Editor แล้วทำการแก้ไขได้

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

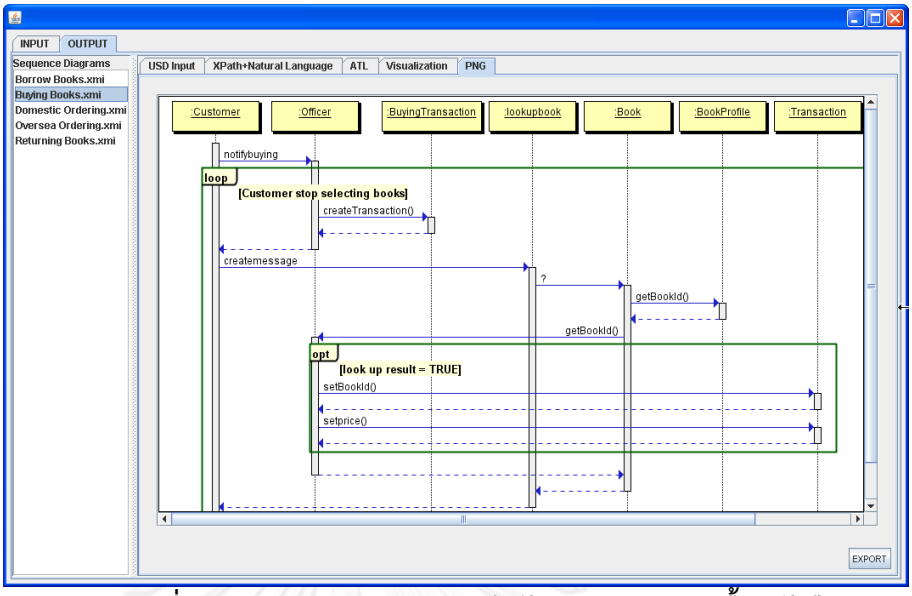
1.4) ระบบสามารถนำค่าที่ได้จากขั้นตอนการแสดงผล แล้วประมวลผลออกมาเป็นรูปแผนภาพลำดับ และสามารถส่งออกให้อยู่ในรูปแบบไฟล์พีเอ็นจี (PNG) ของแผนภาพลำดับการซื้อหนังสือ ดังรูปที่ 32



รูปที่ 30 ผลลัพธ์การประมวลผลด้วย ATL ของยูสเคสการซื้อหนังสือ



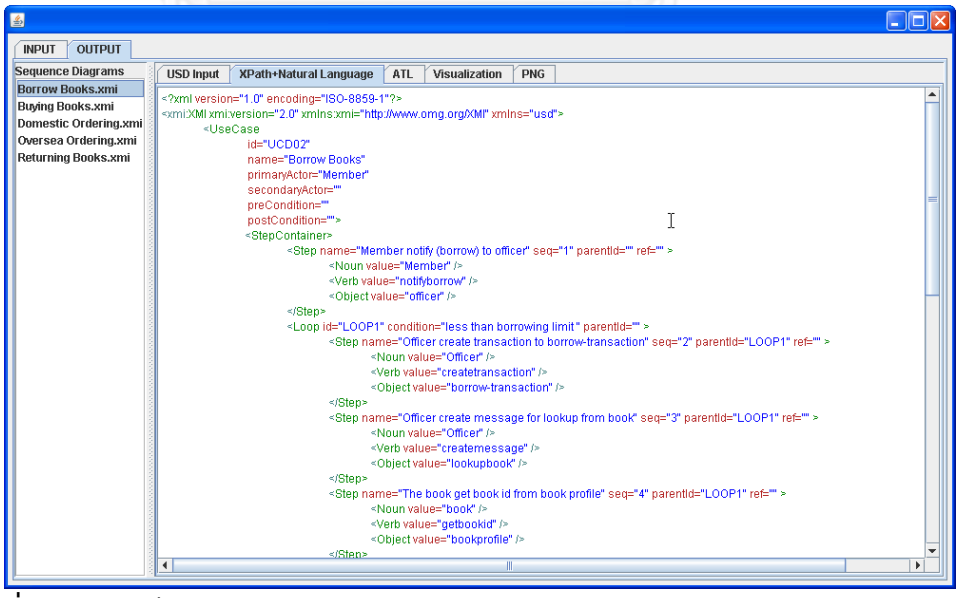
รูปที่ 31 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการซื้อหนังสือ



รูปที่ 32 การแสดงผลแผนภาพลำดับของยูสเคสการซื้อหนังสือ

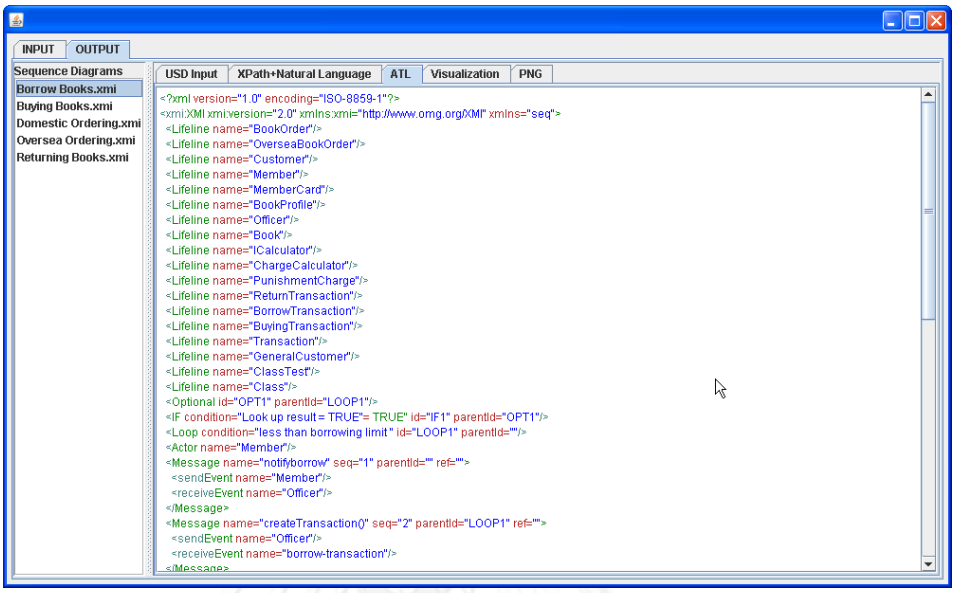
2) ผลการทดสอบยูสเคสการยืมหนังสือ

2.1) ข้อมูลจากคำอธิบายยูสเคสการยืมหนังสือ ระบบสามารถดึงข้อมูลมาแล้วสามารถพัฒนาร่วมกับภาษาธรรมชาติ เพื่อให้ได้ผลลัพธ์ตามโครงสร้างเมทาโมเดลต้นทางที่ต้องการได้ดังรูปที่ 33 ซึ่งแต่ละแท็กขั้นตอนการทำงาน (step) จะแทนด้วยลำดับขั้นตอนการทำงานของแต่ละกิจกรรม และแท็กเงื่อนไขซึ่งจะแทนด้วยเงื่อนไขการเกิดเหตุการณ์ตามลำดับดังที่ระบุในตารางที่ 21 จาก รูปที่ 33 ประกอบด้วยแท็กลำดับขั้นตอนการทำงาน 9 ขั้นตอน และแท็กเงื่อนไข 2 เงื่อนไข



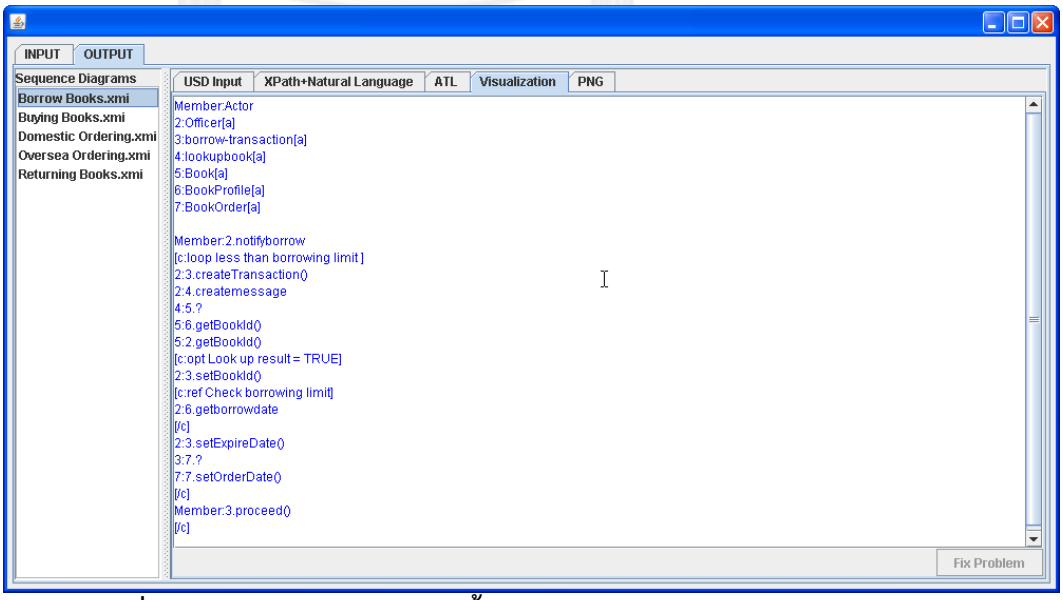
รูปที่ 33 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการยืมหนังสือ

2.2) เมื่อข้อมูลคำอธิบายยูสเคสการยืมหนังสือถูกสกัดให้อยู่ในรูปแบบเมทาโมเดลต้นทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นการประมวลผลโดยภาษาเอทีแอลซึ่งจะได้ผลลัพธ์ดังรูปที่ 34 ซึ่งผลลัพธ์ที่ได้จะอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับ



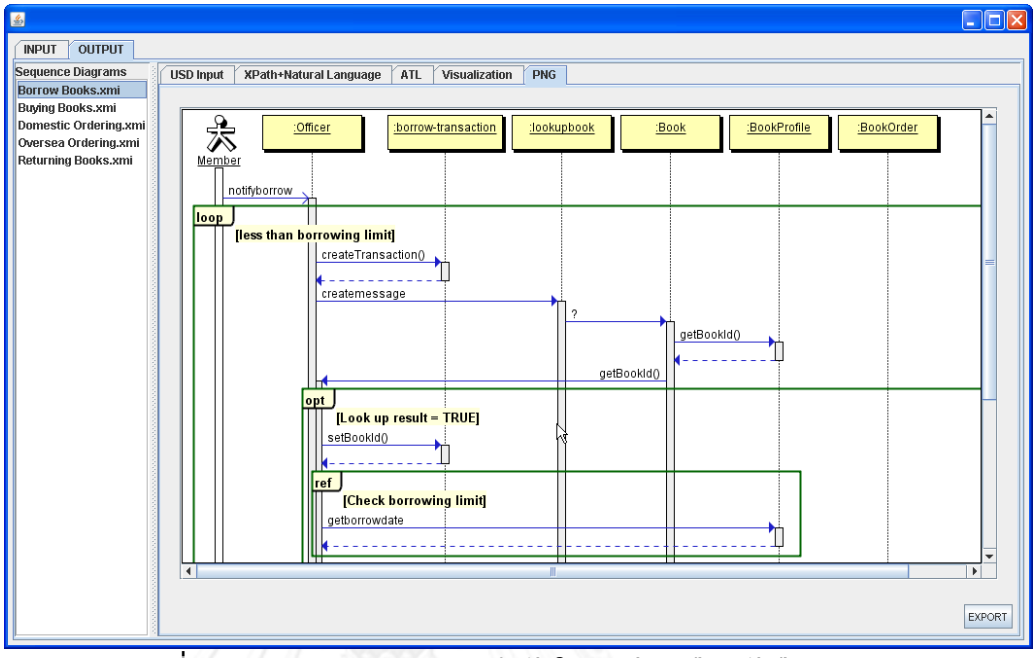
รูปที่ 34 ผลการประมวลผลด้วย ATL ในกรณีการยืมหนังสือ

2.3) เมื่อผลลัพธ์ยูสเคสการยืมหนังสืออยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับเรียบร้อยแล้ว เพื่อสนับสนุนให้การแสดงผลเป็นแผนภาพลำดับ จะต้องทำการแปลงค่าให้อยู่ในโครงสร้างภาษาที่โปรแกรม Quick Sequence Editor กำหนด จากผลการแปลงค่าของยูสเคสนี้จะได้ผลลัพธ์ดังรูปที่ 35 สามารถนำผลลัพธ์ที่ได้ไปเปิดที่โปรแกรม Quick Sequence Editor แล้วทำการแก้ไขได้



รูปที่ 35 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการยืมหนังสือ

2.4) ระบบสามารถนำค่าที่ได้จากขั้นตอนการแสดงผล แล้วประมวลผลออกมาเป็นรูปแผนภาพลำดับการยืมหนังสือ และสามารถส่งออกให้อยู่ในรูปแบบไฟล์พีเอ็นจี (PNG) ดังรูปที่ 36

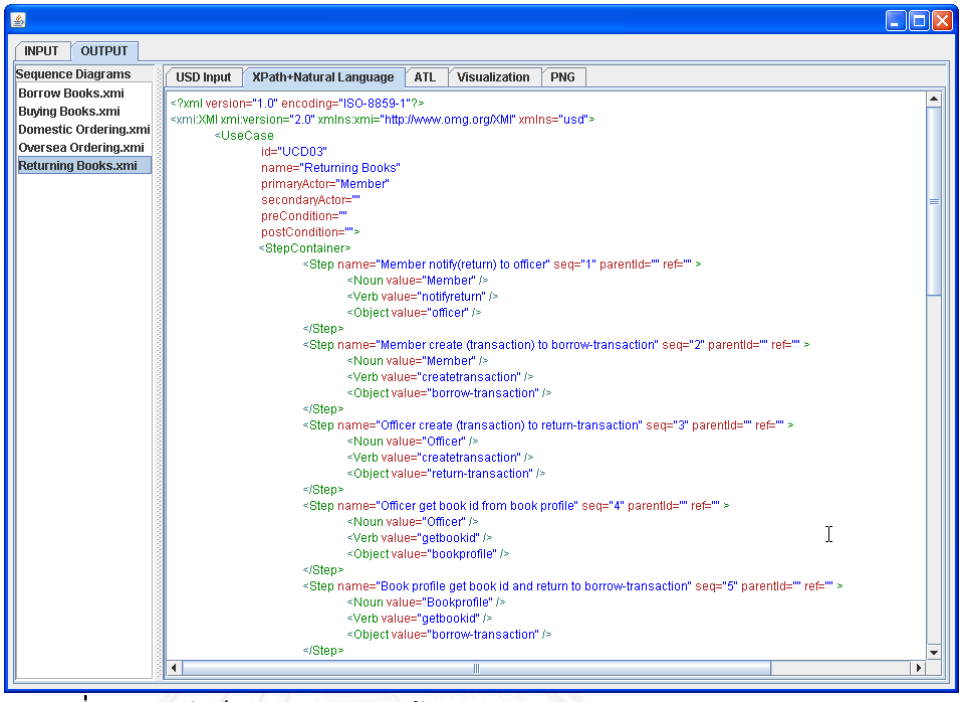


รูปที่ 36 การแสดงผลแผนภาพลำดับในกรณีการยืมหนังสือ

3) ผลการทดสอบยูสเคสการคืนหนังสือ

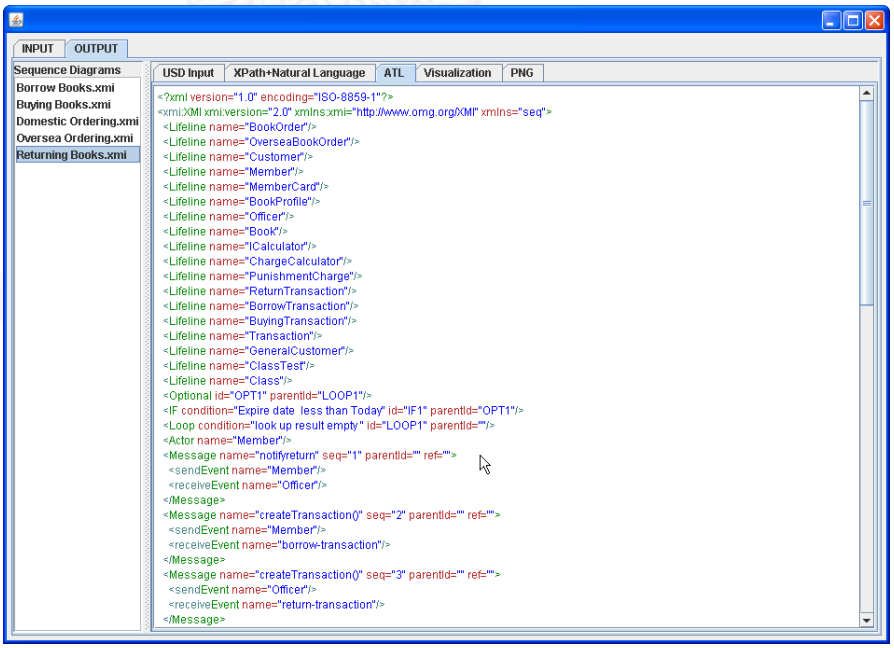
3.1) ข้อมูลจากคำอธิบายยูสเคสการคืนหนังสือ ระบบสามารถดึงข้อมูลมาแล้วสามารถพัฒนาร่วมกับภาษาธรรมชาติ เพื่อให้ได้ผลลัพธ์ตามโครงสร้างเมทาโมเดลต้นทางที่ต้องการได้ดังรูปที่ 37 แต่ละแท็กขั้นตอนการทำงาน (step) จะแทนด้วยลำดับขั้นตอนการทำงานของแต่ละกิจกรรม และแท็กเงื่อนไขซึ่งจะแทนด้วยเงื่อนไขการเกิดเหตุการณ์ตามลำดับดังที่ระบุในตารางที่ 22 จากรูปที่ 37 ประกอบด้วยแท็กลำดับขั้นตอนการทำงาน 13 ขั้นตอน และแท็กเงื่อนไข 2 เงื่อนไข

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



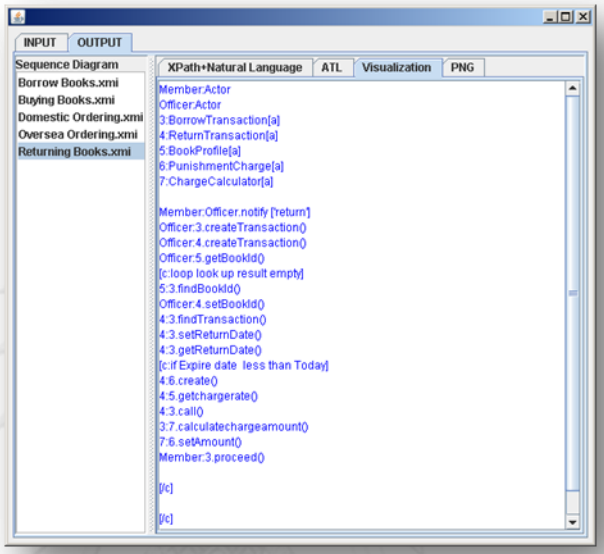
รูปที่ 37 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการคืนหนังสือ

3.2) เมื่อข้อมูลคำอธิบายยูสเคสการคืนหนังสือถูกสกัดให้อยู่ในรูปแบบเมทาโมเดลต้นทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นการประมวลผลโดยภาษาเอทีแอลซึ่งจะได้ผลลัพธ์ดังรูปที่ 38 ผลลัพธ์ที่ได้จะอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับ



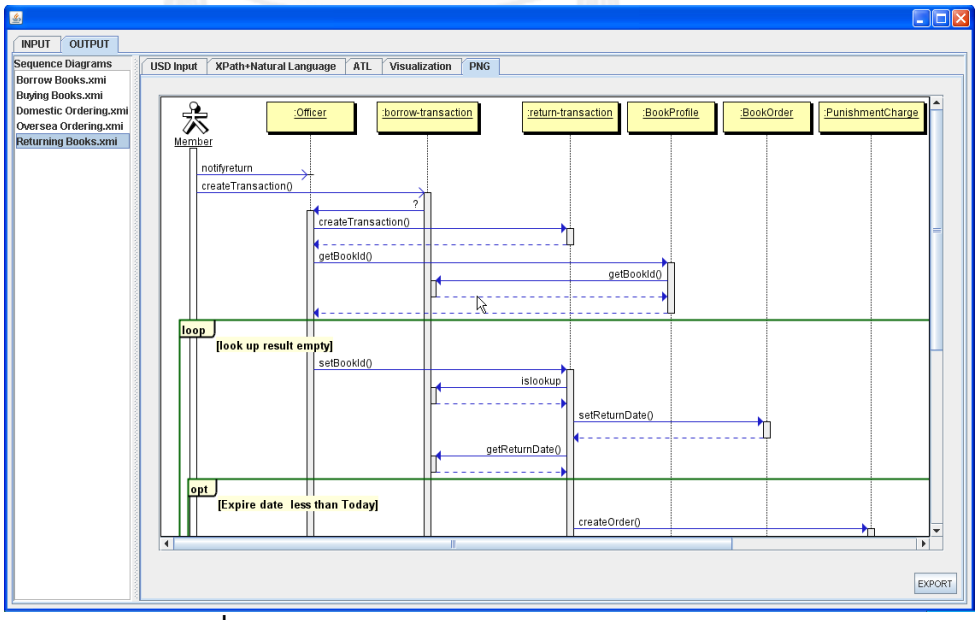
รูปที่ 38 ผลการประมวลผลด้วย ATL ในกรณีการคืนหนังสือ

3.3) เมื่อผลลัพธ์ยูสเคสการคืนหนังสืออยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับเรียบร้อยแล้ว เพื่อสนับสนุนให้การแสดงผลเป็นแผนภาพลำดับ จะต้องทำการแปลงค่าให้อยู่ในโครงสร้างภาษาที่โปรแกรม Quick Sequence Editor กำหนด จากผลการแปลงค่าของยูสเคสนี้จะได้ผลลัพธ์ดังรูปที่ 39 สามารถนำผลลัพธ์ที่ได้ไปเปิดที่โปรแกรม Quick Sequence Editor แล้วทำการแก้ไขได้



รูปที่ 39 ผลการประมวลผลขั้นตอนการแสดงผลในกรณีการคืนหนังสือ

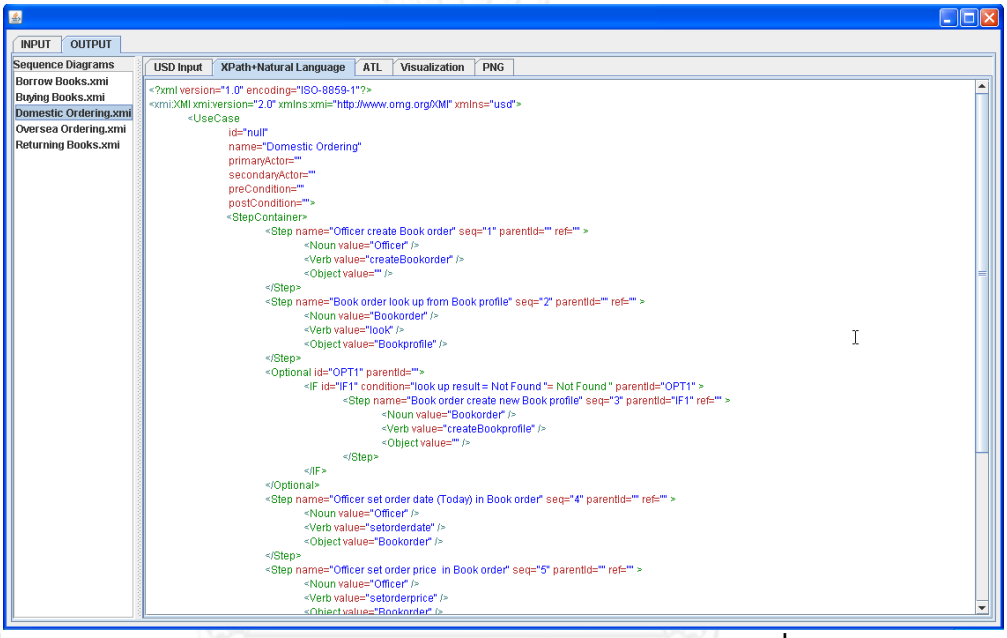
3.4) ระบบสามารถนำค่าที่ได้จากขั้นตอนการแสดงผล แล้วประมวลผลออกมาเป็นรูปแผนภาพลำดับการคืนหนังสือ และสามารถส่งออกให้อยู่ในรูปแบบไฟล์พีเอ็นจี (PNG) ดังรูปที่ 40



รูปที่ 40 การแสดงผลแผนภาพลำดับของยูสเคสการคืนหนังส

4) ผลการทดสอบยูสเคสการสั่งหนังสือภายในประเทศ

4.1) ข้อมูลจากคำอธิบายยูสเคสการสั่งหนังสือภายในประเทศ ระบบสามารถดึงข้อมูลมาแล้วสามารถพัฒนาร่วมกับภาษาธรรมชาติ เพื่อให้ได้ผลลัพธ์ตามโครงสร้างเมทาโมเดลต้นทางที่ต้องการได้ดังรูปที่ 41 ซึ่งแต่ละแท็กขั้นตอนการทำงาน (step) จะแทนด้วยลำดับขั้นตอนการทำงานของแต่ละกิจกรรม และแท็กเงื่อนไขซึ่งจะแทนด้วยเงื่อนไขการเกิดเหตุการณ์ตามลำดับดังที่ระบุในตารางที่ 23 จากรูปที่ 41 ประกอบด้วยแท็กลำดับขั้นตอนการทำงาน 7 ขั้นตอน และแท็กเงื่อนไข 1 เงื่อนไข

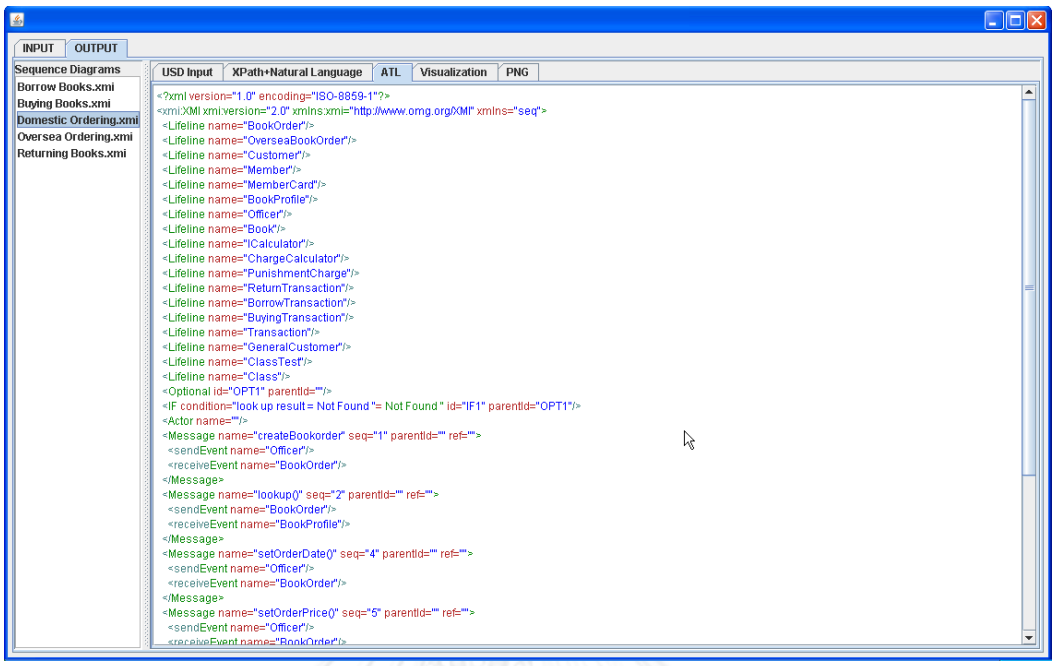


รูปที่ 41 ผลการประมวลผลด้วย XPath และภาษาธรรมชาติในกรณีการสั่งหนังสือภายในประเทศ

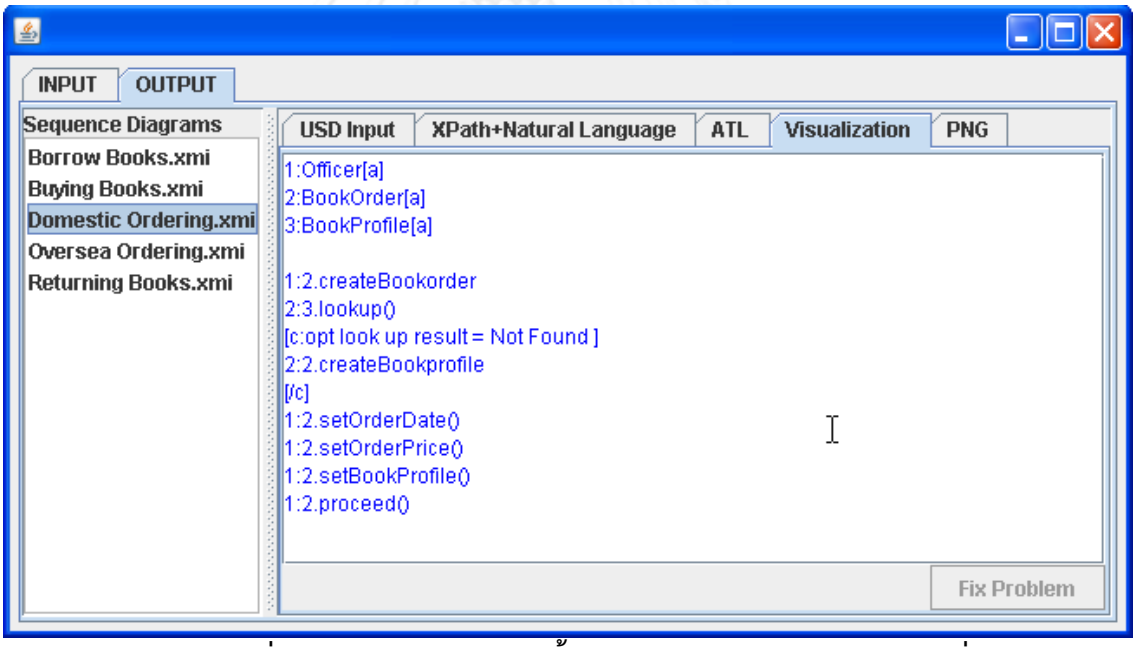
4.2) เมื่อข้อมูลคำอธิบายยูสเคสการสั่งหนังสือภายในประเทศถูกสกัดให้อยู่ในรูปแบบเมทาโมเดลต้นทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นการประมวลผลโดยภาษาเอทีแอลซึ่งจะได้ผลลัพธ์ดังรูปที่ 42 ซึ่งผลลัพธ์ที่ได้จะอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับ

4.3) เมื่อผลลัพธ์ยูสเคสการสั่งหนังสือภายในประเทศอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับเรียบร้อยแล้ว เพื่อสนับสนุนให้การแสดงผลเป็นแผนภาพลำดับ จะต้องทำการแปลงค่าให้อยู่ในโครงสร้างภาษาที่โปรแกรม Quick Sequence Editor กำหนด จากผลการแปลงค่าของยูสเคสนี้จะได้ผลลัพธ์ดังรูปที่ 43 สามารถนำผลลัพธ์ที่ได้ไปเปิดที่โปรแกรม Quick Sequence Editor แล้วทำการแก้ไขได้

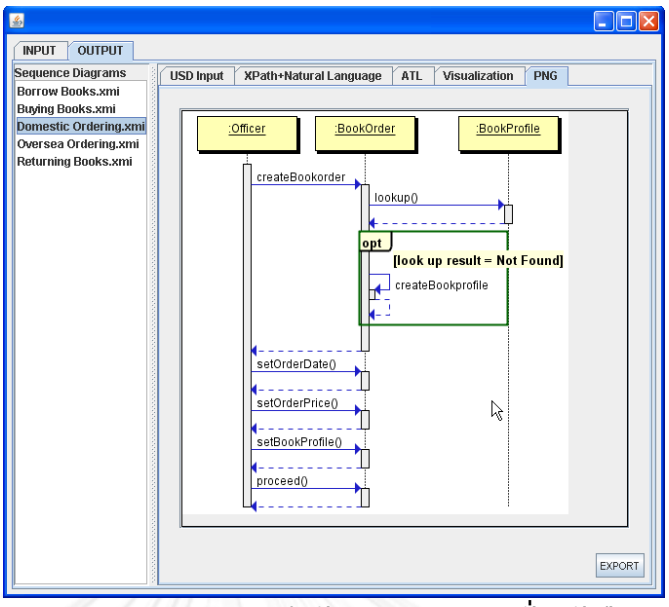
4.4) ระบบสามารถนำค่าที่ได้จากขั้นตอนการแสดงผล แล้วประมวลผลออกมาเป็นรูปแผนภาพลำดับการสั่งหนังสือภายในประเทศ และสามารถส่งออกให้อยู่ในรูปแบบไฟล์พีเอ็นจี (PNG) ดังรูปที่ 44



รูปที่ 42 ผลการประมวลผลด้วย ATL ในกรณีการสั่งหนังสือภายในประเทศ



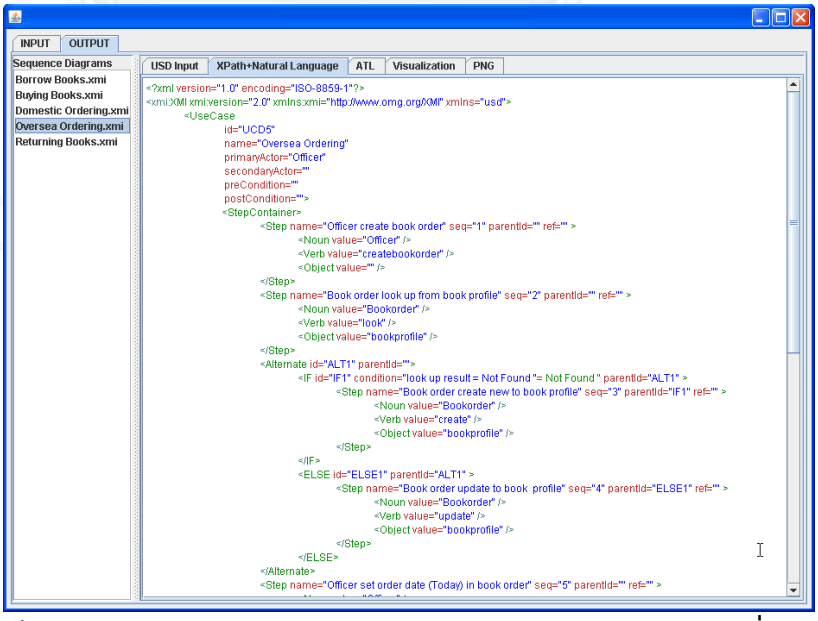
รูปที่ 43 ผลการประมวลผลขั้นตอนการแสดงผลของยูสเคสการสั่งหนังสือภายในประเทศ



รูปที่ 44 การแสดงผลแผนภาพลำดับของยูสเคสการสั่งหนังสือภายในประเทศ

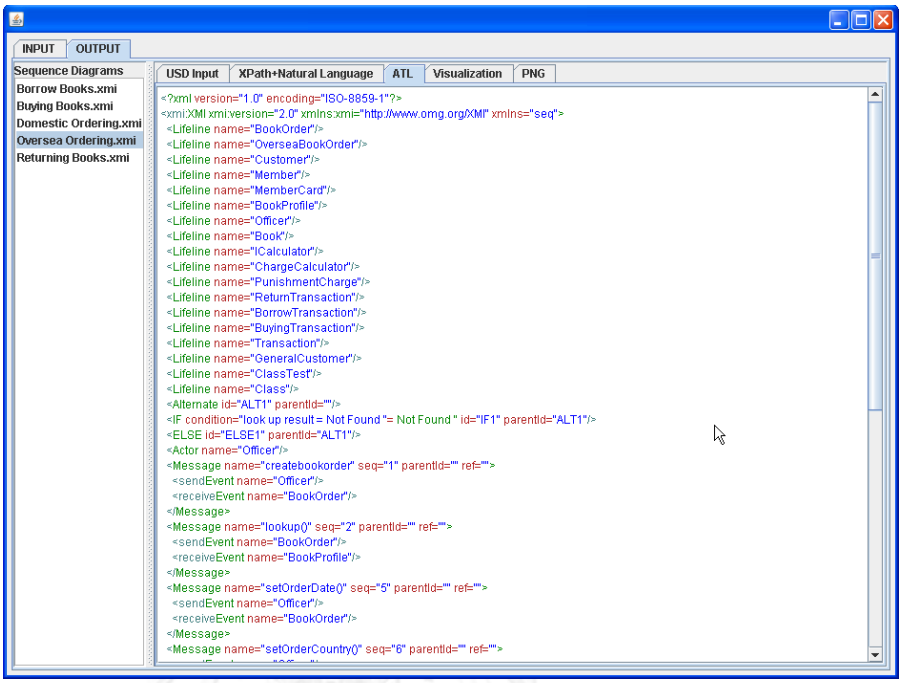
5) ผลการทดสอบยูสเคสการสั่งหนังสือจากต่างประเทศ

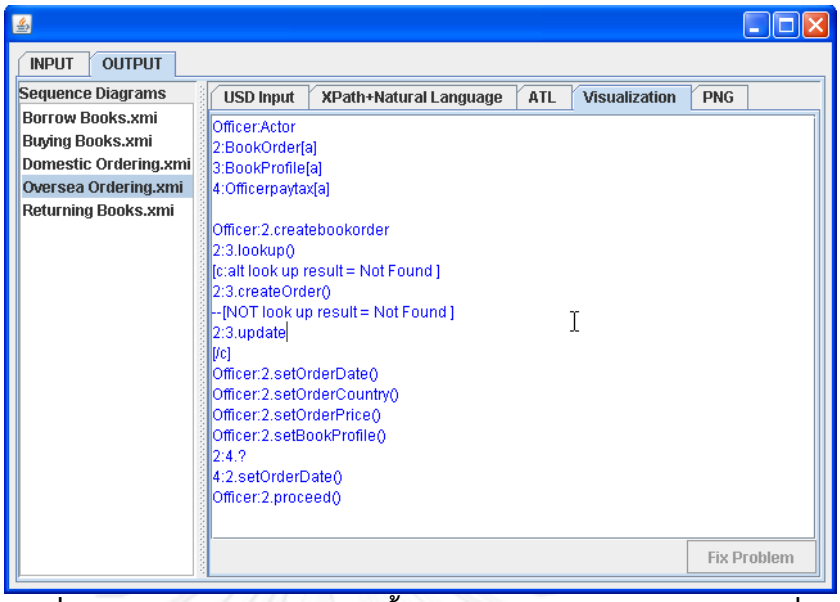
5.1) ข้อมูลจากคำอธิบายยูสเคสการสั่งหนังสือจากต่างประเทศ ระบบสามารถดึงข้อมูลมาแล้วสามารถพัฒนาร่วมกับภาษาธรรมชาติ เพื่อให้ได้ผลลัพธ์ตามโครงสร้างเมทาโมเดลต้นทางที่ต้องการได้ดังรูปที่ 45 ซึ่งแต่ละแท็กขั้นตอนการทำงาน (step) จะแทนด้วยลำดับขั้นตอนการทำงานของแต่ละกิจกรรม และแท็กเงื่อนไขซึ่งจะแทนด้วยเงื่อนไขการเกิดเหตุการณ์ตามลำดับดังที่ระบุในตารางที่ 24 จากรูปที่ 45 ประกอบด้วยแท็กลำดับขั้นตอนการทำงาน 9 ขั้นตอน และแท็กเงื่อนไข 1 เงื่อนไข



รูปที่ 45 ผลลัพธ์การประมวลผลด้วย XPath และภาษาธรรมชาติของยูสเคสการสั่งหนังสือจากต่างประเทศ

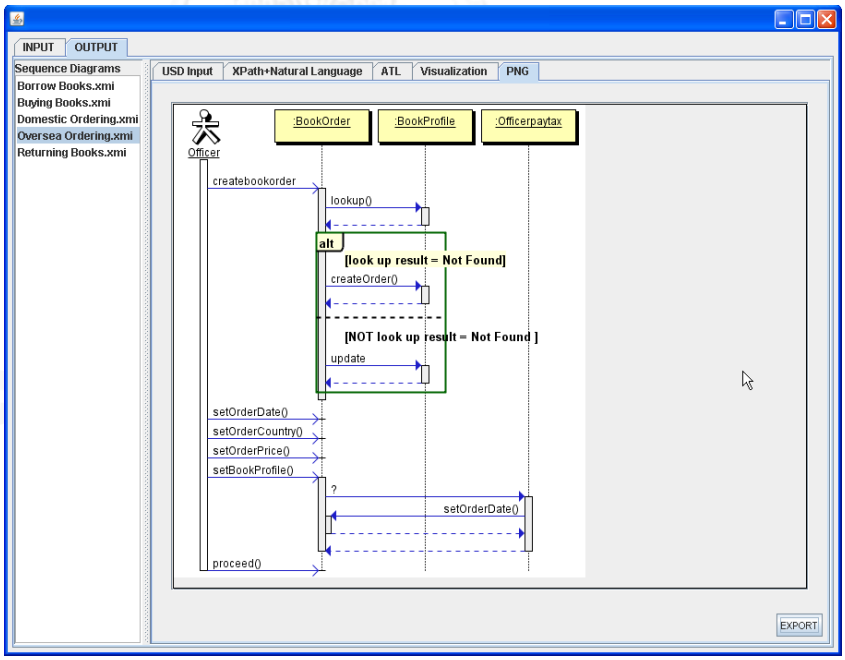
5.2) เมื่อข้อมูลคำอธิบายยูสเคสการสั่งหนังสือจากต่างประเทศถูกสกัดให้อยู่ในรูปแบบเมทาโมเดลต้นทางเรียบร้อยแล้ว ขั้นตอนต่อไปเป็นการประมวลผลโดยภาษาเอทีแอลซึ่งจะได้ผลลัพธ์ดังรูปที่ 46 ซึ่งผลลัพธ์ที่ได้จะอยู่ในรูปแบบเมทาโมเดลของแผนภาพลำดับ


INPUT
OUTPUT
Sequence Diagrams
Borrow Books.xmi
Buying Books.xmi
Domestic Ordering.xmi
Oversea Ordering.xmi
Returning Books.xmi
USD Input
XPath+Natural Language
ATL
Visualization
PNG
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns="seq">
<Lifeline name="BookOrder"/>
<Lifeline name="OverseaBookOrder"/>
<Lifeline name="Customer"/>
<Lifeline name="Member"/>
<Lifeline name="MemberCard"/>
<Lifeline name="BookProfile"/>
<Lifeline name="Officer"/>
<Lifeline name="Book"/>
<Lifeline name="ICalculator"/>
<Lifeline name="ChargeCalculator"/>
<Lifeline name="PunishmentCharge"/>
<Lifeline name="ReturnTransaction"/>
<Lifeline name="BorrowTransaction"/>
<Lifeline name="BuyingTransaction"/>
<Lifeline name="Transaction"/>
<Lifeline name="GeneralCustomer"/>
<Lifeline name="ClassTest"/>
<Lifeline name="Class"/>
<Alternate id="ALT1" parentId=""/>
<IF condition="look up result = Not Found "= Not Found " id="IF1" parentId="ALT1"/>
<ELSE id="ELSE1" parentId="ALT1"/>
<Actor name="Officer"/>
<Message name="createbookorder" seq="1" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="lookup()" seq="2" parentId="" ref="">
<sendEvent name="BookOrder"/>
<receiveEvent name="BookProfile"/>
</Message>
<Message name="setOrderDate()" seq="5" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="6" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="7" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="8" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="9" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="10" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="11" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="12" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="13" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="14" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="15" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="16" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="17" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="18" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="19" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="20" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="21" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="22" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="23" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="24" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="25" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="26" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="27" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="28" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="29" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="30" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="31" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="32" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="33" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="34" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="35" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="36" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="37" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="38" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="39" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="40" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="41" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="42" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="43" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="44" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="45" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="46" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="47" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="48" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="49" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="50" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="51" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="52" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="53" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="54" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="55" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="56" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="57" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="58" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="59" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="60" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="61" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="62" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="63" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="64" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="65" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="66" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="67" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="68" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="69" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="70" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="71" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="72" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="73" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="74" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="75" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="76" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="77" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="78" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="79" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="80" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="81" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="82" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="83" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="84" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="85" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="86" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="87" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="88" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="89" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="90" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="91" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="92" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="93" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="94" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="95" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="96" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="97" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="98" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="99" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="100" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="101" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="102" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="103" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="104" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="105" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="106" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="107" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="108" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="109" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="110" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="111" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="112" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="113" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="114" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="115" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="116" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="117" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="118" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="119" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="120" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="121" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="122" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="123" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="124" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="125" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="126" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="127" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="128" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="129" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="130" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="131" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="132" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="133" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="134" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="135" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="136" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="137" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="138" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="139" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="140" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="141" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="142" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="143" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="144" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="145" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="146" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="147" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="148" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="149" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="150" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="151" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="152" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="153" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="154" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="155" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="156" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="157" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="158" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="159" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="160" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="161" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="162" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="163" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="164" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="165" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="166" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="167" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="168" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="169" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="170" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="171" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="172" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="173" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="174" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="175" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="176" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="177" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="178" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="179" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="180" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="181" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="182" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="183" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="184" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="185" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="186" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="187" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="188" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="189" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="190" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="191" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="192" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="193" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="194" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="195" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="196" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="197" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="198" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="199" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="200" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="201" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="202" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="203" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="204" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="205" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="206" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="207" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="208" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="209" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="210" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="211" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="212" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="213" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="214" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="215" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="216" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="217" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="218" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="219" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="220" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="221" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="222" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderTotalPrice()" seq="223" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderStatus()" seq="224" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderDate()" seq="225" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderCountry()" seq="226" parentId="" ref="">
<sendEvent name="Officer"/>
<receiveEvent name="BookOrder"/>
</Message>
<Message name="setOrderPrice()" seq="227



รูปที่ 47 ผลลัพธ์การประมวลผลขั้นตอนการแสดงผลของยูสเคสการสั่งหนังสือจากต่างประเทศ

5.4) ระบบสามารถนำค่าที่ได้จากขั้นตอนการแสดงผล แล้วประมวลผลออกมาเป็นรูปแผนภาพลำดับการสั่งหนังสือจากต่างประเทศ และสามารถส่งออกให้อยู่ในรูปแบบไฟล์พีเอ็นจี (PNG) ดังรูปที่ 48



รูปที่ 48 การแสดงผลแผนภาพลำดับของยูสเคสการสั่งหนังสือจากต่างประเทศ

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

ผลลัพธ์โดยสรุปที่ได้จากงานวิจัยนี้ มีดังนี้

1. ได้แนวทางสำหรับการสร้างแผนภาพลำดับจากแผนภาพยูสเคสประกอบคำอธิบายและแผนภาพคลาส โดยใช้เทคโนโลยีการแปลงตัวแบบ
2. ได้ระบบต้นแบบในการสร้างแผนภาพลำดับ
3. ได้เครื่องมือสนับสนุนการปรับปรุงกระบวนการซอฟต์แวร์ ซึ่งช่วยลดทรัพยากรและข้อผิดพลาดในกระบวนการพัฒนาซอฟต์แวร์

6.2 ข้อจำกัด

- เอกสารเอกซ์เอ็มไอที่นำเข้าเข้าในงานวิจัยนี้รองรับเฉพาะที่สร้างขึ้นมาจากเครื่องวาดภาพวิชวลพาราดามเท่านั้น
- ข้อมูลนำเข้าในงานวิจัยนี้จะสนับสนุนเฉพาะภาษาอังกฤษเท่านั้น และจะต้องมีความสัมพันธ์กันระหว่างคำอธิบายแผนภาพยูสเคสกับแผนภาพคลาส
- คำอธิบายแผนภาพยูสเคสจะต้องเป็นข้อความหรือประโยคที่เขียนถูกต้องตามโครงสร้างภาษาตามที่ผู้วิจัยเป็นผู้กำหนด

6.3 แนวทางการวิจัยต่อ

- สำหรับขั้นตอนการประมวลผลของภาษาธรรมชาติมาใช้ในงานวิจัยนี้ เนื่องจากได้นำไลบรารี่มาใช้จึงอาจจะมีความสามารถไม่เพียงพอตามที่เราต้องการ ถ้าเราสามารถพัฒนาวิธีการประมวลผลของภาษาธรรมชาติหรือหาไลบรารี่ที่มีความสามารถมากกว่านี้มาใช้ประกอบงานวิจัย แผนภาพลำดับที่ได้จะมีความถูกต้องและสมบูรณ์เพิ่มมากขึ้น
- สำหรับการนำเข้าไฟล์เอกซ์เอ็มไอ เมทาโมเดลจะสนับสนุนเฉพาะการนำเข้าไฟล์จากโปรแกรมวิชชวลพาราม เท่านั้น ถ้าเราสามารถสร้างเมทาโมเดลที่สนุบสนุนการนำเข้าไฟล์จากโปรแกรมอื่นมากขึ้น จะส่งผลให้งานวิจัยนี้มีประสิทธิภาพมากยิ่งขึ้น
- ภาษาเอทีแอลเป็นภาษาสำหรับการแปลงแบบโมเดล ถ้าเรารู้โครงสร้างเมทาโมเดลต้นทางและ โครงสร้างเมทาโมเดลปลายทางแล้ว เราสามารถนำไปประยุกต์ใช้กับแผนภาพอื่นๆที่มีความสัมพันธ์กันได้

รายการอ้างอิง

1. Cetinkaya, D. and A. Verbraeck. Metamodeling and model transformations in modeling and simulation. in Simulation Conference (WSC). Proceedings of the 2011 Winter. Phoenix, AZ: IEEE.
2. Object Management Group, in Unified Modeling Language (OMG UML) ,Superstructure Version 2.4.1. 2011, OMG, Inc.
3. Object Management Group, in Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification version 1.1. 2011, OMG, Inc.
4. Steinberg, D., et al., EMF Eclipse Modeling Framework 2ed. 2009, Boston: Addison-Wesley Professional.
5. Eclipse Foundation, ATL Transformation Language. 2014 [cited 2013 Dec.]; Available from: http://www.eclipse.org/atl/atlTransformations.
6. Object Management Group, in MOF/XMI Mapping version 2.4.1. 2013, OMG, Inc.
7. Laura Mendez, M.C., C.R. Romero, and K.Y. Perez Herrera, UML Sequence Diagram Generator System from Use Case Description Using Natural Language, in Electronics, Robotics and Automotive Mechanics Conference (CERMA2007). 2007. p. 360 - 363.
8. Some, S.S., A Meta-Model for Textual Use Case Description. Journal of Object Technology, 2009. vol.8: p. 87–106.
9. Li, L., Translate Use Cases to Sequence Diagrams, in ASE '00 2000, Proceedings of the 15th IEEE international conference on Automated software engineering: Washington, DC, USA. p. 293.
10. Stanford University, Natural Language Processing. [cited 2014 Feb]; Available from: http://nlp.stanford.edu/software/index.shtml.
11. กลมกล่อม, ก., การวิเคราะห์และออกแบบระบบเชิงวัตถุด้วย UML. 2012, (กิตติ ภักดีวัฒนะกุล, บรรณาธิการ): กรุงเทพมหานคร : สำนักพิมพ์เคทีพี.

CHULALONGKORN UNIVERSITY

ภาคผนวก

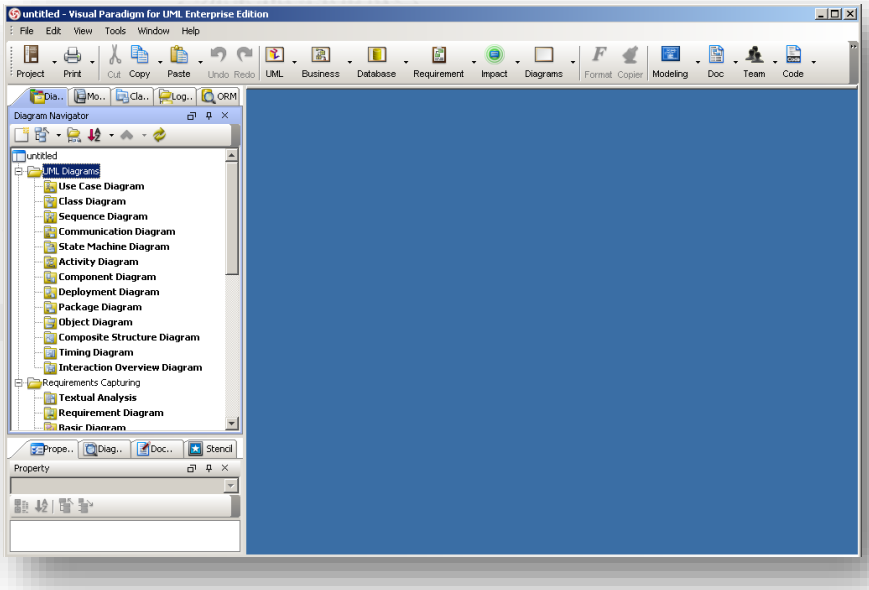
จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ภาคผนวก ก.

การสร้างแผนภาพส่วนประกอบโดยเครื่องมือวิชวลพาราดาม

รูปที่ 49 เครื่องมือวิชวลพาราดาม เวอร์ชันสแตนดาร์ต 10.2

เครื่องมือวิชวลพาราดามเวอร์ชันสแตนดาร์ต 10.2 ดังรูปที่ 49 เป็นเครื่องมือที่ใช้สร้างแผนภาพยูสเคส เพื่อนำเข้าในงานวิจัยนี้ หน้าจอการทำงานหลักแสดงดังรูปที่ 50

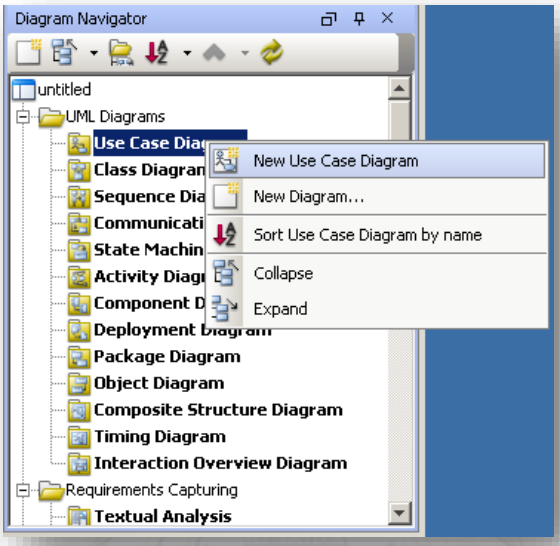


รูปที่ 50 หน้าจอการทำงานของเครื่องมือวิชวลพาราดาม

วิธีการสร้างแผนภาพยูสเคส

1) การสร้างแผนภาพ

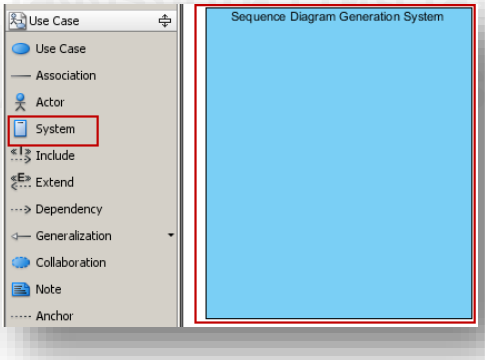
เลือกที่แถบเมนูด้านบนดังนี้ UML Diagrams > Use Case Diagram >New Use Case Diagram ดังรูปที่ 51



รูปที่ 51 การสร้างแผนภาพยูสเคสใหม่

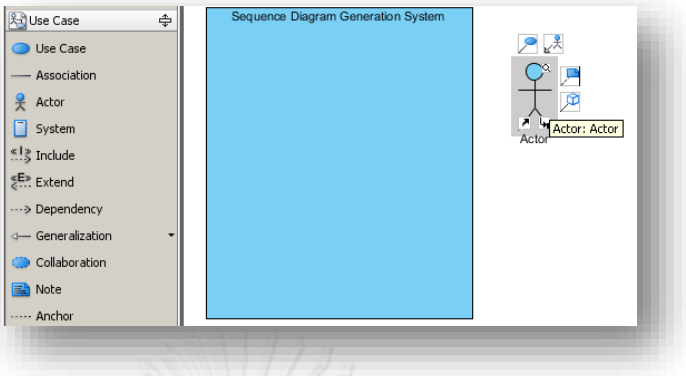
2) การวาดแผนภาพยูสเคส

- เลือกการสร้างแผนภาพประเภท “Use Case”
- กรณีการวาดระบบ (System) ที่แถบเครื่องมือด้านซ้าย ให้ทำการลาก System มายังหน้าจอการวาด แล้วตั้งชื่อระบบ ดังรูปที่ 52



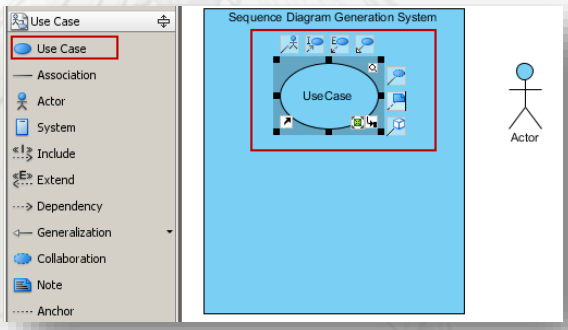
รูปที่ 52 การวาดส่วนประกอบ System

- กรณีการวาดผู้ใช้งานระบบ (Actor) ที่แถบเครื่องมือด้านซ้าย ให้ทำการลาก Actor มายังหน้าจอการวาด แล้วตั้งชื่อผู้ใช้งานระบบ ดังรูปที่ 53



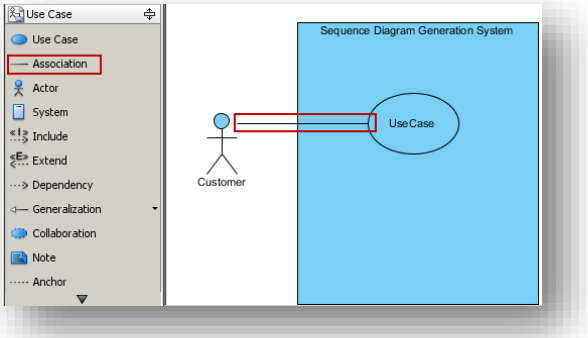
รูปที่ 53 การวาดส่วนประกอบผู้ใช้งานระบบ

- กรณีการวาดฟังก์ชันการทำงาน (Use Case) ที่แถบเครื่องมือด้านซ้าย ให้ทำการลาก Use Case มายังหน้าจอการวาด แล้วตั้งชื่อฟังก์ชันการทำงานของระบบ ดังรูปที่ 54



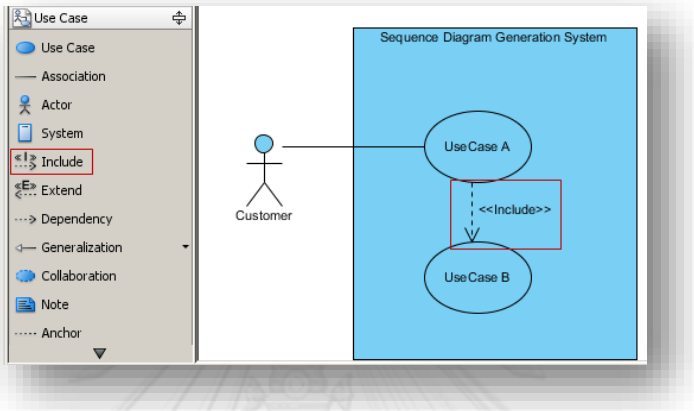
รูปที่ 54 การวาดฟังก์ชันการทำงานของระบบ

- กรณีการวาดเส้นความสัมพันธ์ในแผนภาพยูสเคส กรณีเป็นเส้นความสัมพันธ์ (Association) ระหว่างผู้ใช้กับยูสเคส วิธีใช้คือเลือกไอคอน Association ซึ่งจะมีสัญกรณ์เป็นเส้นตรง ทำการลากเส้นระหว่าง Actor กับ Use Case ดัง รูปที่ 55



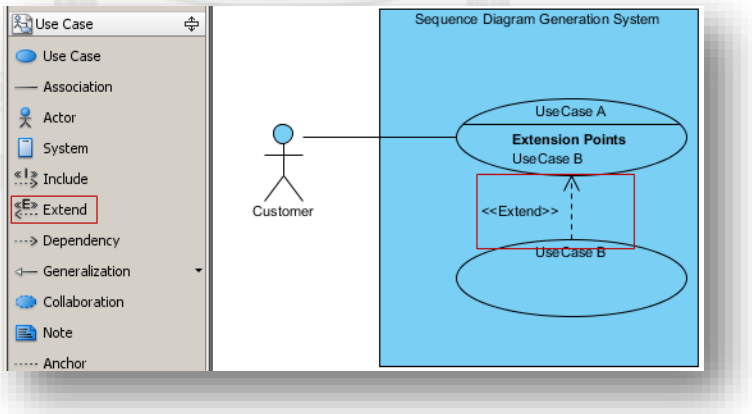
รูปที่ 55 การวาดเส้นความสัมพันธ์ระหว่างผู้ใช้กับยูสเคส

- กรณีการวาดเส้นความสัมพันธ์แบบรวม (Include) ในแผนภาพยูสเคส กรณีเป็นเส้นความสัมพันธ์แบบรวม จะเกิดขึ้นเมื่อยูสเคสหนึ่งเรียกใช้งานอีกยูสเคสหนึ่ง ซึ่งจะมีสัญกรณ์เป็นเส้นประตรงที่มีลูกศรชี้ไปยังยูสเคสที่ถูกเรียกใช้และมีคำว่า <<include>> อยู่บนเส้นลูกศรวิธีใช้ให้เลือกไอคอน Include ซึ่งจะมีลักษณะเป็นเส้นประตรงและมีสัญกรณ์ <<I>> อยู่บนเส้น ทำการลากเส้นระหว่าง Use Case ดังรูปที่ 56 จากรูปอธิบายได้ว่า ยูสเคสบีถูกเรียกใช้โดยยูสเคสเอ



รูปที่ 56 ตัวอย่างการวาดเส้นความสัมพันธ์แบบรวมในแผนภาพยูสเคส

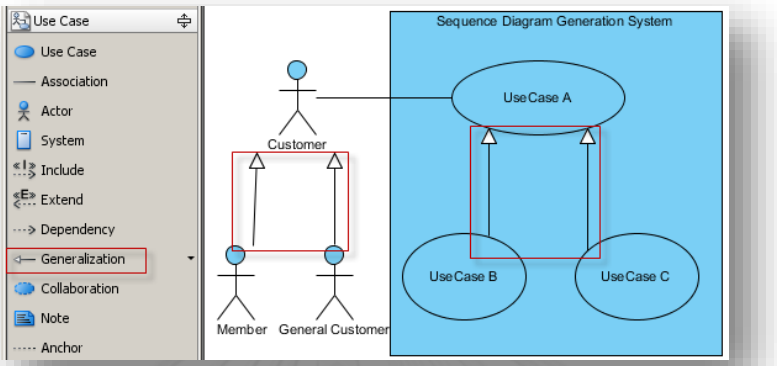
- กรณีการวาดเส้นความสัมพันธ์แบบขยาย (Extend) ในแผนภาพยูสเคส กรณีเป็นเส้นความสัมพันธ์แบบขยาย เกิดขึ้นเมื่อยูสเคสหนึ่งไม่สามารถทำงานได้ตามปกติจึงมียูสเคสอีกอันทำงานแทนเพื่อเป็นทางเลือก ซึ่งจะมีสัญกรณ์เป็นเส้นประตรงที่มีหัวลูกศรชี้จากยูสเคสที่ถูกขยายและมีคำว่า <<extends>> กำกับบนเส้นลูกศร วิธีใช้คือเลือกไอคอน Extend ซึ่งจะมีลักษณะเป็นเส้นประตรงและมีสัญกรณ์ <<E>> อยู่บนเส้น ทำการลากเส้นระหว่าง Use Case ดังรูปที่ 57 จากรูปอธิบายได้ว่า ยูสเคสเอมีความสัมพันธ์แบบขยายกับยูสเคสซี



รูปที่ 57 ตัวอย่างการวาดเส้นความสัมพันธ์แบบขยายในแผนภาพยูสเคส

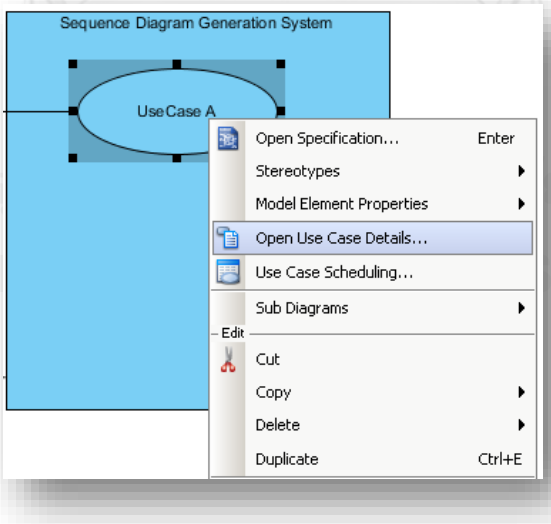
- กรณีการวาดเส้นความสัมพันธ์แบบสืบทอด (Generalization) ในแผนภาพยูสเคส กรณีเป็นเส้นความสัมพันธ์แบบสืบทอด เกิดขึ้นเมื่อมีการสืบทอดระหว่างผู้ใช้ระบบ หรือมีการสืบทอด

ระหว่างยูสเคส ซึ่งจะมีสัญกรณ์เป็นเส้นตรงที่มีหัวลูกศรเป็นรูปสามเหลี่ยมโปร่ง ชี้ไปยังผู้ใช้ระบบหรือยูสเคสต้นแบบ วิธีใช้คือเลือกไอคอน Generalization แล้วทำการลากเส้นความสัมพันธ์แบบสืบทอดของผู้ใช้ระบบ หรือยูสเคส ดังรูปที่ 58 จากรูปอธิบายได้ว่า Member และ General Customer สืบทอดมาจาก Customer และ ยูสเคสบีและยูสเคสซีสืบทอดมาจากยูสเคสเอ



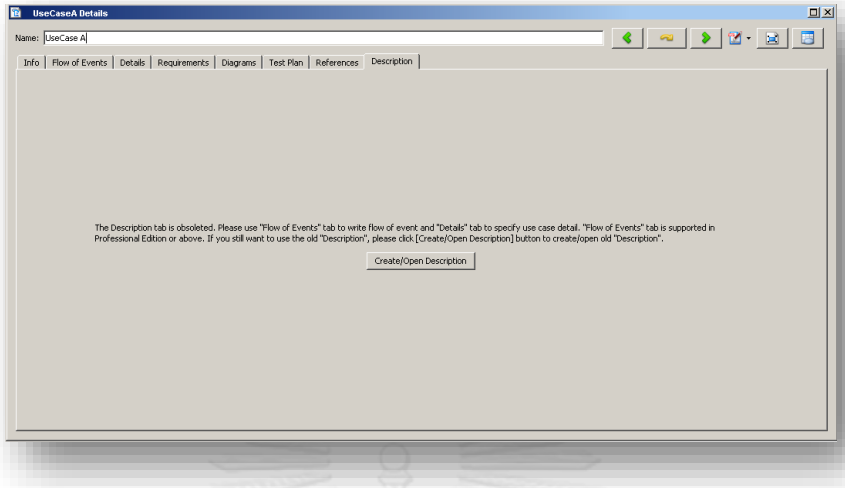
รูปที่ 58 ตัวอย่างการวาดความสัมพันธ์แบบสืบทอดในแผนภาพยูสเคส

ขั้นตอนต่อไปเป็นขั้นตอนการเขียนคำอธิบายจากแผนภาพยูสเคสแต่ละฟังก์ชันงาน วิธีการเขียนคือ เลือกฟังก์ชันงานที่ต้องการแล้วคลิกขวาเลือกไปที่ Open Use Case Detail... ดังรูปที่ 59

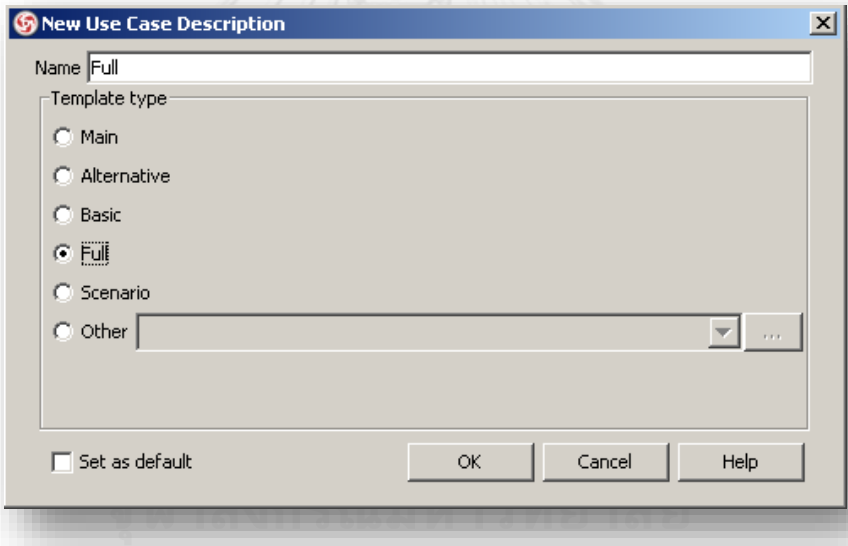


รูปที่ 59 วิธีการสร้างคำอธิบายจากแผนภาพยูสเคส

– ในแท็ป Description จะพบหน้าจอดังรูปที่ 60 ให้กดปุ่ม Create/Opene Description แล้วเลือกรูปแบบการกรอกคำอธิบายแบบ Full ดัง รูปที่ 61



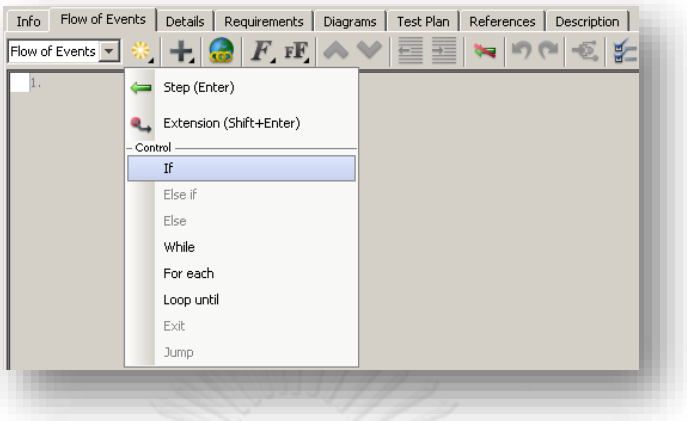
รูปที่ 60 หน้าจอแสดงปุ่มกดเพื่อทำการสร้างคำอธิบายยูสเคส



Chulalongkorn University

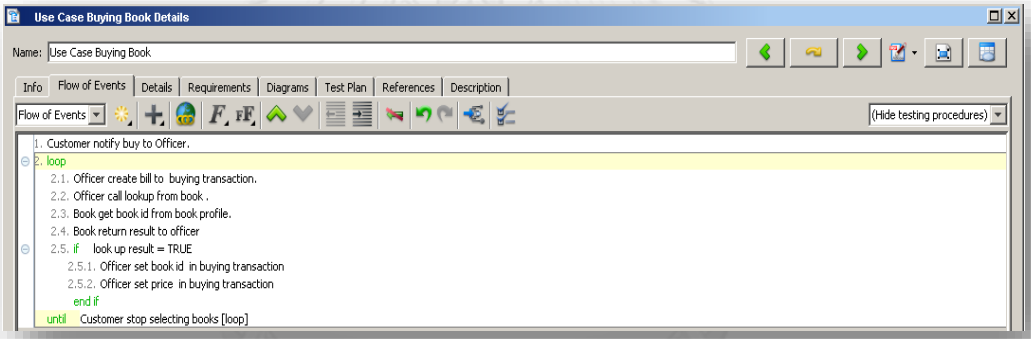
รูปที่ 61 หน้าจอการเลือกรูปแบบการกรอกคำอธิบายยูสเคส

- ในแท็ป Flow of Events ให้เขียนอธิบายฟังก์ชันงานโดยละเอียดเป็นข้อเรียงลำดับตามเหตุการณ์ที่เกิดก่อนหลัง และกรณีที่ต้องการเขียนเงื่อนไขในฟังก์ชันงานให้ไปเลือกที่ไอคอนและเลือกเงื่อนไขดัง รูปที่ 62



รูปที่ 62 ตัวอย่างวิธีการสร้างเงื่อนไขในคำอธิบายยูสเคส

- คำอธิบายยูสเคสที่ใช้ในงานวิจัยจะมีลักษณะดังรูปที่ 63 ซึ่งข้อความที่เขียนควรที่จะเป็นไปตามขอบเขตและเงื่อนไขที่กำหนดในบทที่ 1

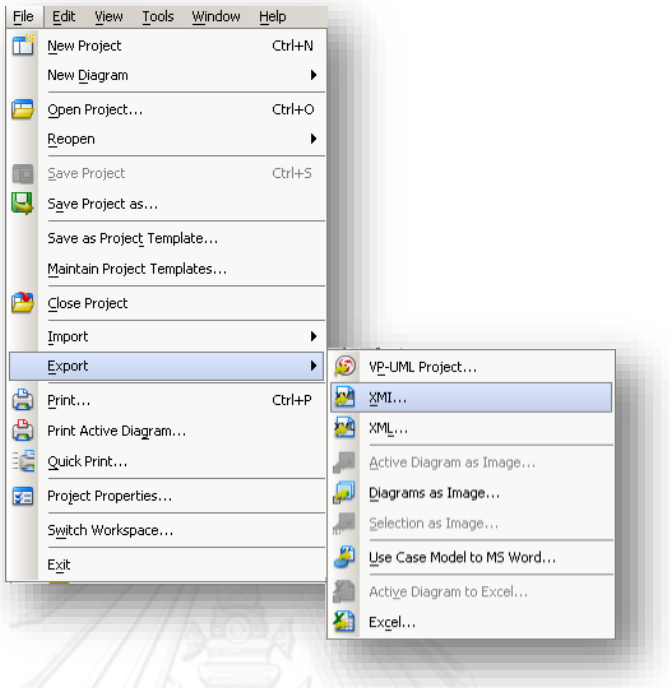


รูปที่ 63 ตัวอย่างการเขียนคำอธิบายยูสเคส

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

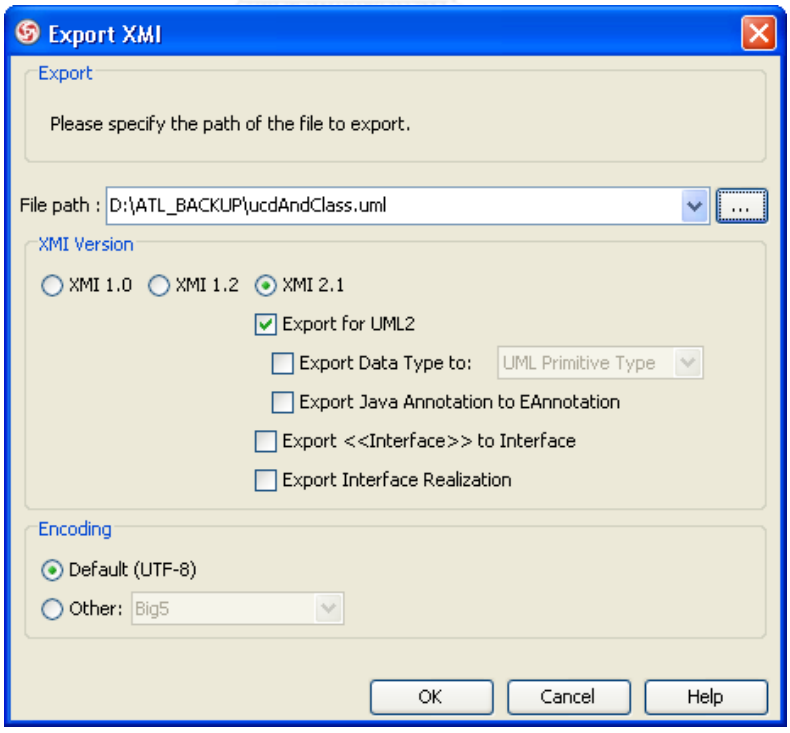
3) การแปลงแผนภาพยูสเคสประกอบคำอธิบายให้เป็นเอกสารเอกซ์เอ็มไอ

- ในขั้นตอนการพัฒนาเครื่องมือที่ใช้ในงานวิจัย ข้อมูลนำเข้าจะต้องเป็นไฟล์ที่อยู่ในรูปแบบเอกซ์เอ็มไอ ซึ่งวิธีการ คือ ให้เข้าไปที่แถบเมนู คลิก File > Export > XMI ดังรูปที่ 64



รูปที่ 64 การเลือกเมนู Export XMI

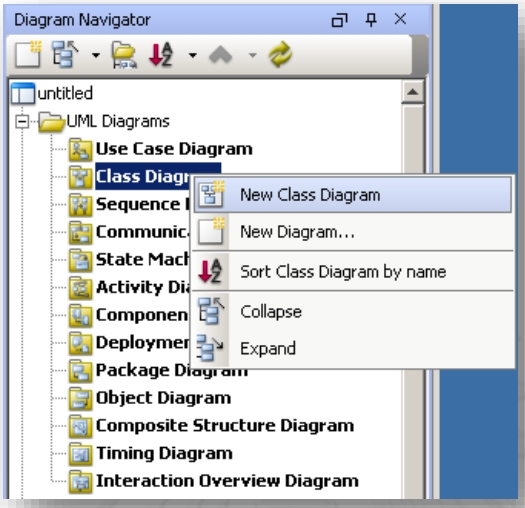
- ระบุที่อยู่และตั้งชื่อไฟล์ที่ File path แล้วตั้งค่าต่างๆ ตามรูปที่ 65 แล้วคลิก “OK”



รูปที่ 65 ตัวอย่างหน้าจอการ Export XMI

วิธีการสร้างแผนภาพคลาส

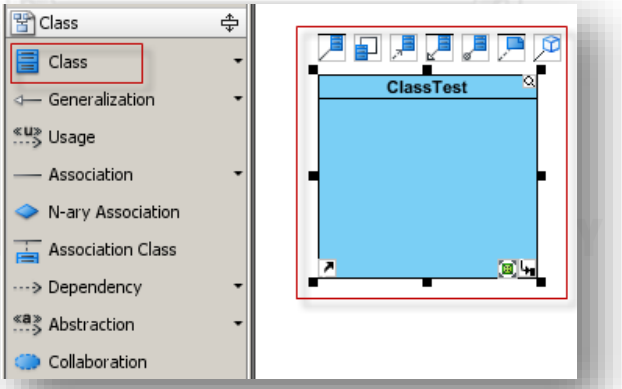
1) การสร้างแผนภาพคลาส เลือกที่แถบเมนูด้านบนดังนี้ UML Diagrams > Class Diagram >New Class Diagram ดังรูปที่ 66



รูปที่ 66 การสร้างแผนภาพคลาสใหม่

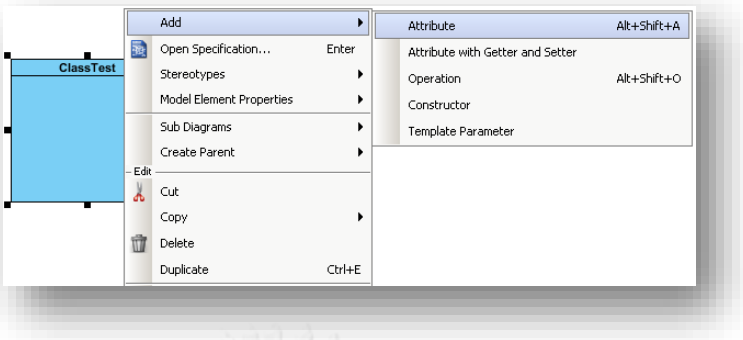
2) การวาดแผนภาพคลาส

- กรณีการวาดคลาส (Class) ที่แถบเครื่องมือด้านซ้าย ให้ทำการลาก Class มายังหน้าจอการวาด แล้วตั้งชื่อระบบ ดังรูปที่ 67

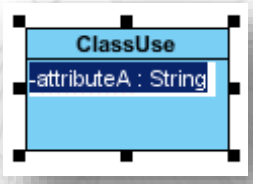


รูปที่ 67 การวาดแผนภาพคลาสไดอะแกรม

- กรณีการวาดคุณลักษณะ (Attribute) ที่คลาส ให้คลิกขวาที่คลาสที่ต้องการ แล้วเลือก Add>>Attribute ดังรูปที่ 68 ทำการตั้งชื่อตัวแปรและระบุประเภทของตัวแปรแอตทริบิวต์ให้เรียบร้อย ดังรูปที่ 69

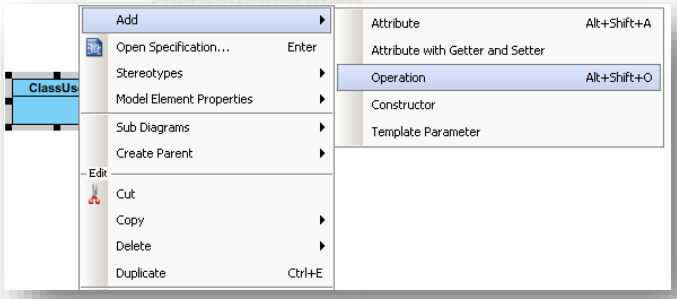


รูปที่ 68 การวาดคุณลักษณะภายในคลาส

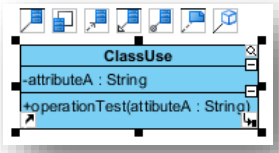


รูปที่ 69 การตั้งชื่อตัวแปรและระบุประเภทตัวแปร

- กรณีการวาดเมธอด (Operation) ที่คลาส ให้คลิกขวาที่คลาสที่ต้องการ แล้วเลือก Add>>Operation ดังรูปที่ 70 ทำการตั้งชื่อเมธอดและระบุพารามิเตอร์ให้เรียบร้อย ดังรูปที่ 71

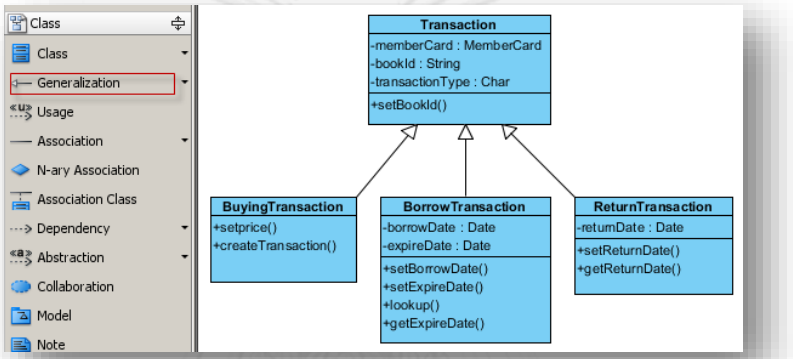


รูปที่ 70 การวาดเมธอดภายในคลาส



รูปที่ 71 การตั้งชื่อเมธอดและระบุพารามิเตอร์

- กรณีการวาดเส้นความสัมพันธ์แบบสืบทอด (Generalization) ในแผนภาพคลาส จะเกิดขึ้นในกรณีที่ความสัมพันธ์ระหว่างคลาสมีลักษณะของการสืบทอดคุณสมบัติ จากโครงสร้างคลาสหนึ่งไปยังโครงสร้างอีกคลาสหนึ่ง ซึ่งจะมีสัญกรณ์เป็นเส้นตรงที่มีหัวลูกศรเป็นรูปสามเหลี่ยมโปร่งชี้ไปยังคลาสต้นแบบ (Super Class) วิธีใช้คือเลือกไอคอน Generalization แล้วทำการลากเส้นความสัมพันธ์แบบสืบทอดจากคลาสต้นแบบ (Super Class) ไปยังคลาสลูก (Sub Class) ดังตัวอย่างดังรูปที่ 72 จากรูปอธิบายได้ว่า Class BuyingTransaction, Class BorrowTransaction และ Class Return Transaction สืบทอดคุณสมบัติมาจาก Class Transaction



รูปที่ 72 ตัวอย่างการวาดความสัมพันธ์แบบสืบทอดในแผนภาพคลาส

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

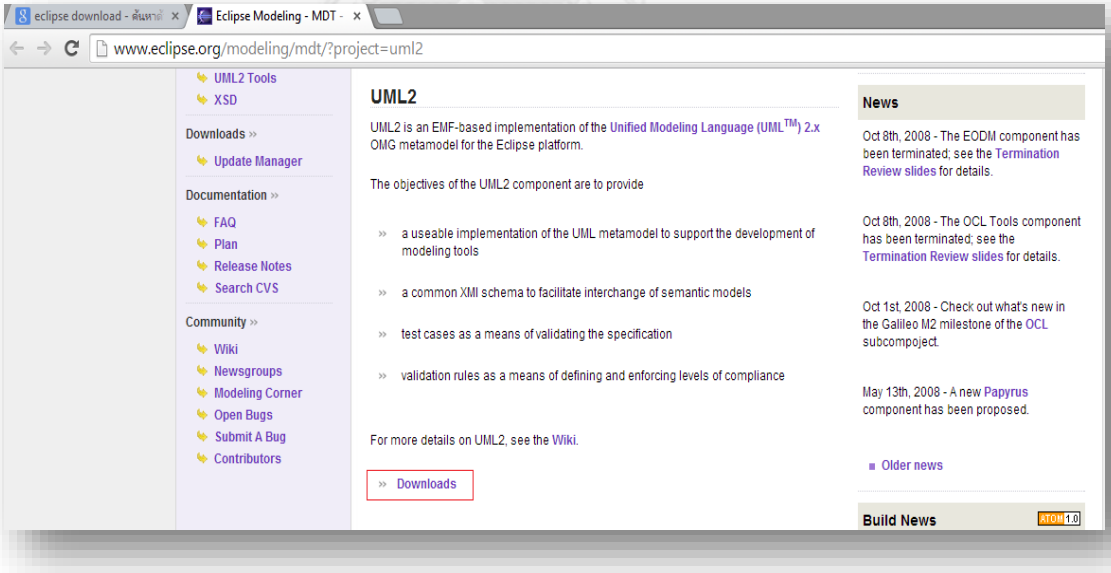
ภาคผนวก ข.
การติดตั้งเครื่องมือเพื่อสนับสนุนการทำสร้างอีคอร์

เครื่องมือสำหรับสร้างเมทาโมเดลบนพื้นฐานของ Eclipse platform เพื่อพัฒนาระบบตามแนวคิด EMF โดยรองรับมาตรฐาน UML 2.x ซึ่งถูกกำหนดรูปแบบโดย OMG. คุณลักษณะที่นำเสนอโดย UML2 นั้นประกอบไปด้วย

1. สามารถสร้าง UML metamodel สำหรับสนับสนุนการพัฒนาระบบในแนวคิดที่ยึดโมเดลเป็นพื้นฐาน
2. ใช้มาตรฐาน xmi อันเป็นมาตรฐานกลางในการแลกเปลี่ยนข้อมูลระหว่างโมเดล
3. สร้างโมเดลที่อ้างอิงตามเมทาโมเดลที่ได้กำหนดไว้แล้ว
4. ประกอบด้วยเครื่องมือสำหรับตรวจสอบความผิดพลาดของโมเดลในขั้นตอนการพัฒนา

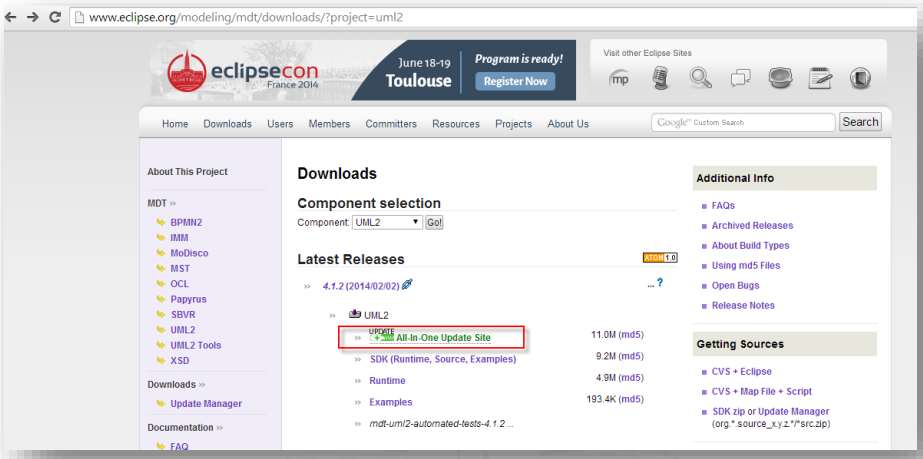
ขั้นตอนการติดตั้ง UML2 Plugin สำหรับ Eclipse 3.5 (Galileo) มีรายละเอียดดังนี้

1.เริ่มขั้นตอนการดาวน์โหลด UML2 plugin สำหรับอีคลิปส์ โดยสามารถเข้าไปที่ http://www.eclipse.org/modeling/mdt/?project=uml2 ดังรูปที่ 73 แล้วกด “Download”



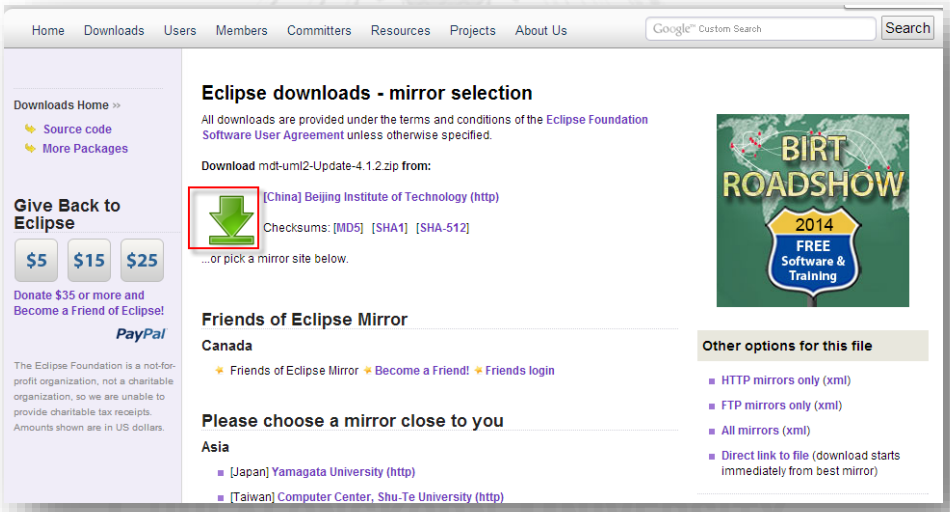
รูปที่ 73 หน้าจอดาวน์โหลดปลั๊กอินสำหรับติดตั้ง UML2

1. เลือก “All-In-One Update Site” ดังรูปที่ 74



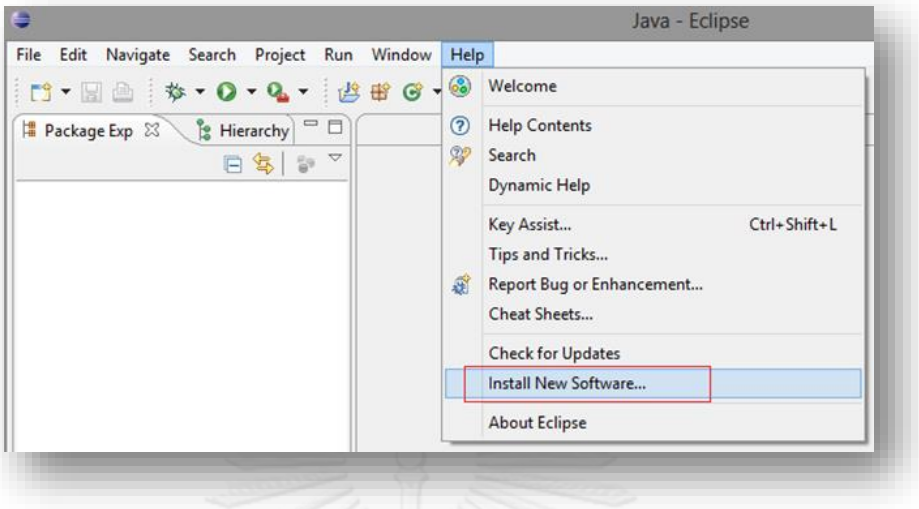
รูปที่ 74 หน้าจอ All-In-One Update Site

2. กดที่รูป download ตามรูปที่ 75 แล้วจะได้ไฟล์ “mdt-uml2-Update-4.1.2.zip”



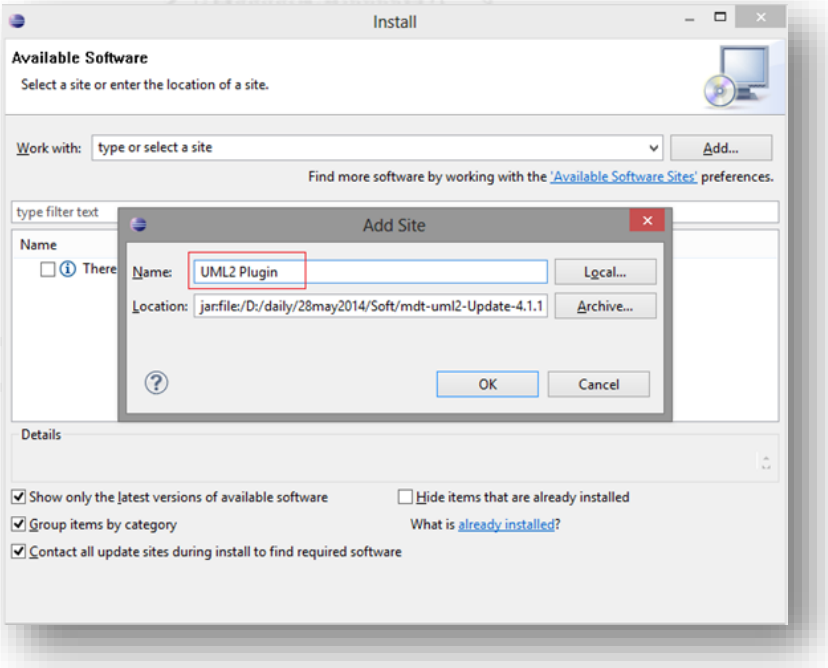
รูปที่ 75 หน้าจอแสดงปุ่มดาวน์โหลดปลั๊กอิน UML2

3. เริ่มขั้นตอนการติดตั้ง UML2 plugin ให้เปิด eclipse ขึ้นมา แล้วไปที่เมนู “Help” ตามด้วย “Install New Software”



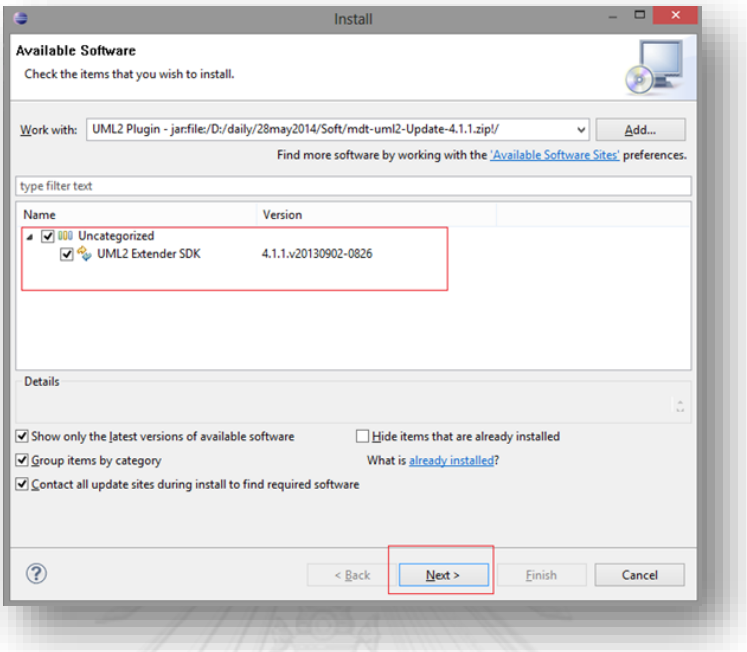
รูปที่ 76 หน้าจอสำหรับติดตั้งปลั๊กอิน UML2

4. กดปุ่ม “Add” เพื่อทำการตั้งชื่อของปลั๊กอิน จากรูปที่ 77 ได้ตั้งชื่อปลั๊กอินที่ใช้สำหรับการพัฒนาเครื่องมือในงานวิจัยนี้ว่า UML2 Plugin สำหรับข้อมูลในช่อง Location ให้กดปุ่ม “Archive” แล้วเลือกไดเรคทอรี่ที่จัดเก็บไฟล์ “mdt-uml2-Update-4.1.1.zip” ซึ่งเป็น UML2 plugin ที่ได้ดาวน์โหลดมาแล้ว



รูปที่ 77 หน้าจอสำหรับติดตั้งค่าปลั๊กอินให้อีคลิปส์

5. ทำการเลือก “UML Extender SDK” และกด “Next” ดังรูปที่ 78

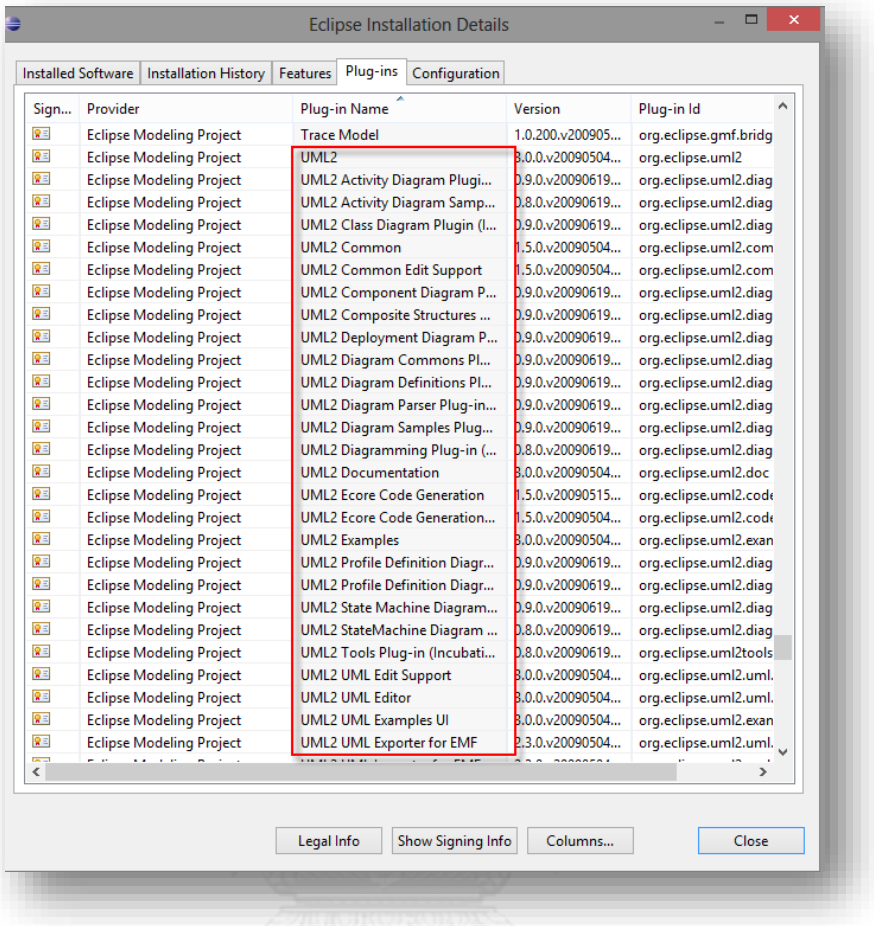


รูปที่ 78 หน้าจอ Available Software

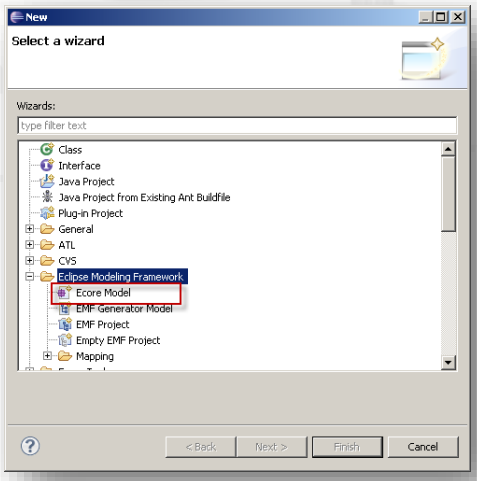
6. เมื่อทำการติดตั้งปลั๊กอินเรียบร้อยแล้วดังรูปที่ 79 จะได้เครื่องมือที่เป็น UML2 เพื่อใช้สำหรับการสร้างอีคอร์ในขั้นตอนการสร้างเมทาโมเดลต้นทางและเมทาโมเดลปลายทางได้

เมื่อติดตั้งปลั๊กอิน UML2 เรียบร้อยแล้ว เพื่อทดสอบว่าการติดตั้งนั้นสมบูรณ์ ควรที่จะทำการรีสตาร์ท Eclipse หนึ่งครั้งแล้วทดสอบโดยไปที่ตำแหน่งที่ต้องการสร้าง Ecore แล้วคลิกขวาเลือก New > Other > ในหน้าต่างโปรแกรมจะมี Eclipse Modeling Framwork เพิ่มเข้ามาเป็นตัวเลือกเพื่อสนับสนุนการสร้างอีคอร์ คลิกเลือก Ecore Model ดังรูปที่ 80

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



รูปที่ 79 หน้าจอแสดงรายละเอียดความสามารถของ UML2 Plugin ที่ติดตั้ง



รูปที่ 80 หน้าจอทดสอบการติดตั้งเครื่องมือเพื่อสนับสนุนการสร้างอีคอร์

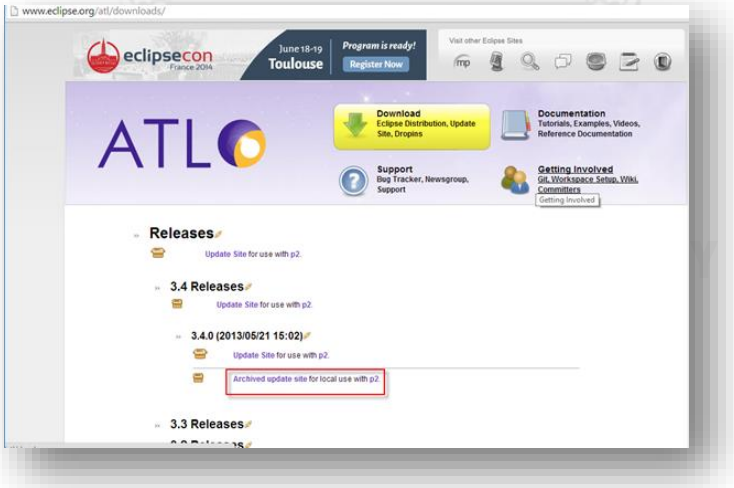
ภาคผนวก ค.
การติดตั้งเครื่องมือเพื่อสนับสนุนการพัฒนาภาษาเอทีแอลบนอีคลิปส์

ขั้นตอนการติดตั้ง ATL Plugin สำหรับ Eclipse 3.5 (Galileo) มีรายละเอียดดังนี้

1. เข้าไปดาวน์โหลด ATL Plugin จากลิงค์ http://www.eclipse.org/atl/ ดังรูปที่ 81

รูปที่ 81 หน้าจอสำหรับดาวน์โหลดเครื่องมือติดตั้ง ATL Plugin

2. เลือก “Archived update site for local use with p2.” ดังรูปที่ 82

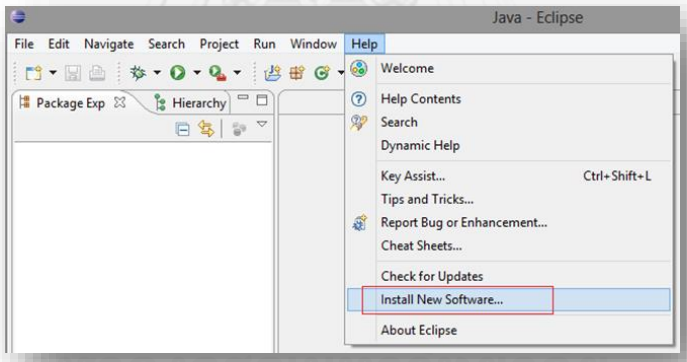


รูปที่ 82 หน้าจอแสดงตำแหน่งสำหรับดาวน์โหลด ATL Plugin

3. จากรูปที่ 83 กดปุ่มตามรูปเพื่อทำการดาวน์โหลดไฟล์ จะได้ไฟล์ “m2m-atl-Update-3.4.0.zip”

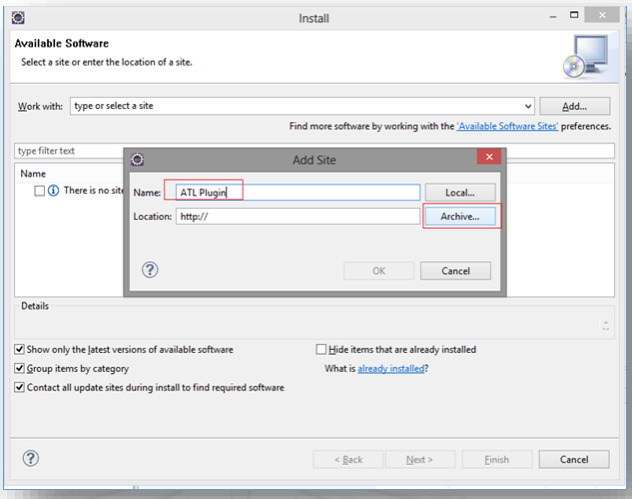
รูปที่ 83 หน้าจอแสดงปุ่มดาวน์โหลด ATL Plugin

4. เริ่มขั้นตอนการติดตั้ง ATL plugin ให้เปิด eclipse ขึ้นมา แล้วไปที่เมนู “Help” ตามด้วย “Install New Software” ดังรูปที่ 84



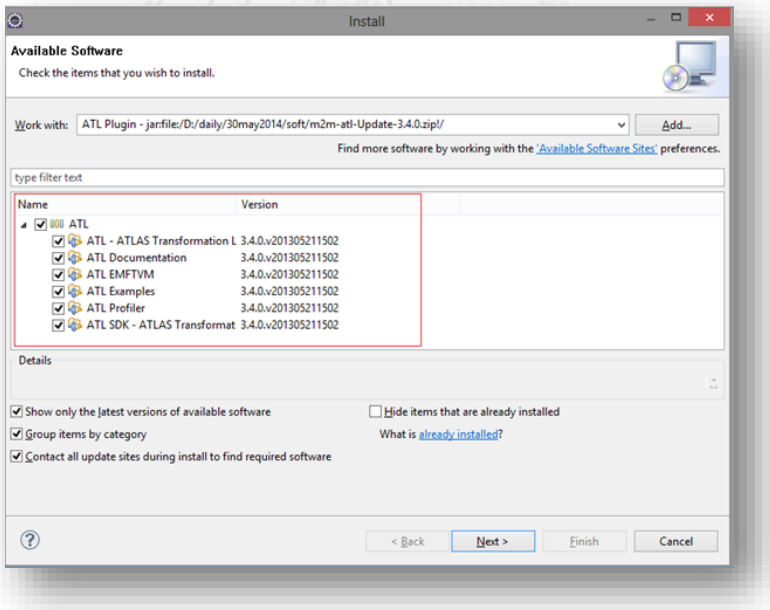
รูปที่ 84 หน้าจอสำหรับติดตั้ง ATL Plugin

5. กดปุ่ม “Add” เพื่อทำการตั้งชื่อของปลั๊กอิน จากรูปที่ 85 ได้ตั้งชื่อปลั๊กอินที่ใช้สำหรับการพัฒนาเครื่องมือในงานวิจัยนี้ว่า ATL Plugin สำหรับข้อมูลในช่อง Location ให้กดปุ่ม “Archive” แล้วเลือกไดเรกทอรี่ที่จัดเก็บไฟล์ “m2m-atl-Update-3.4.0.zip” ซึ่งเป็น ATL plugin ที่ได้ดาวน์โหลดมาแล้ว



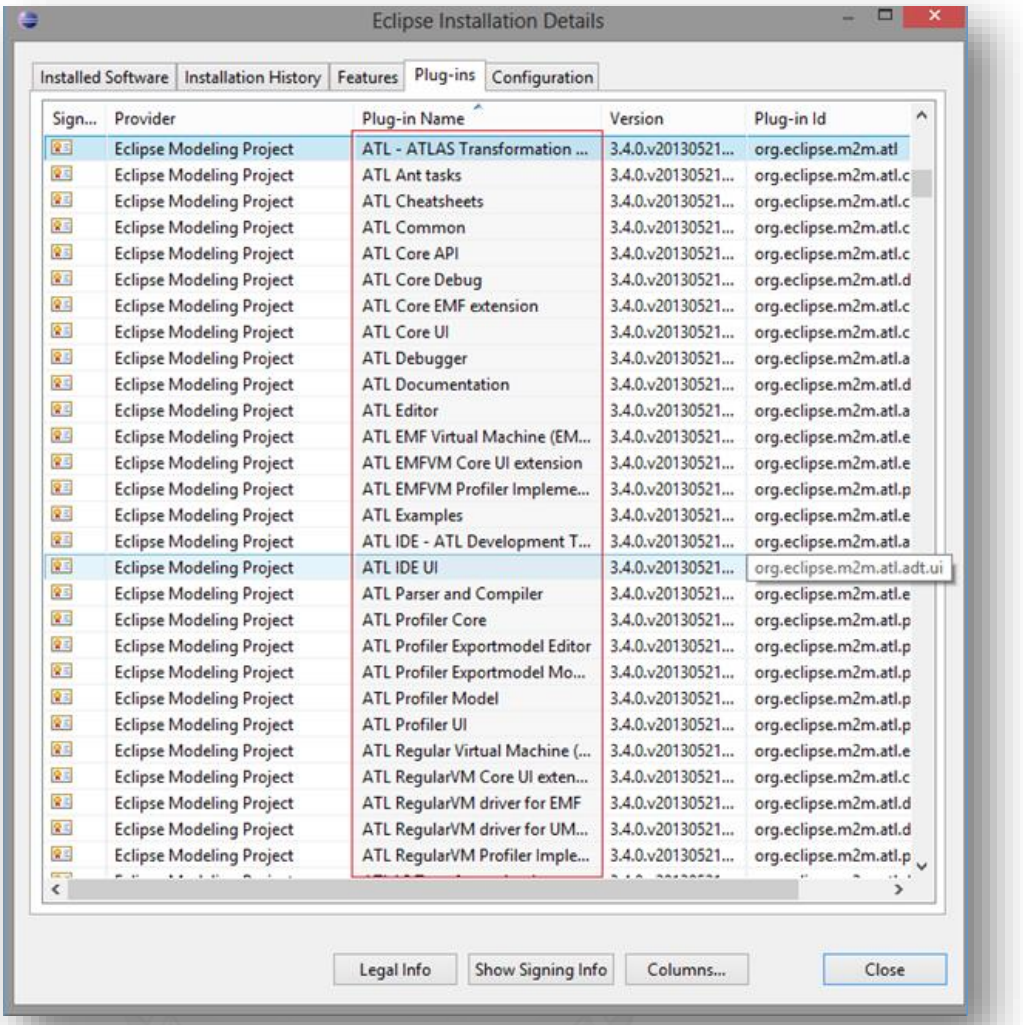
รูปที่ 85 หน้าจอสำหรับติดตั้ง ATL Plugin ให้อีคลิปส์

6. ทำการเลือกส่วนเสริมทั้งหมดของ ATL เสร็จแล้วกดปุ่ม “Next” ดังรูปที่ 86



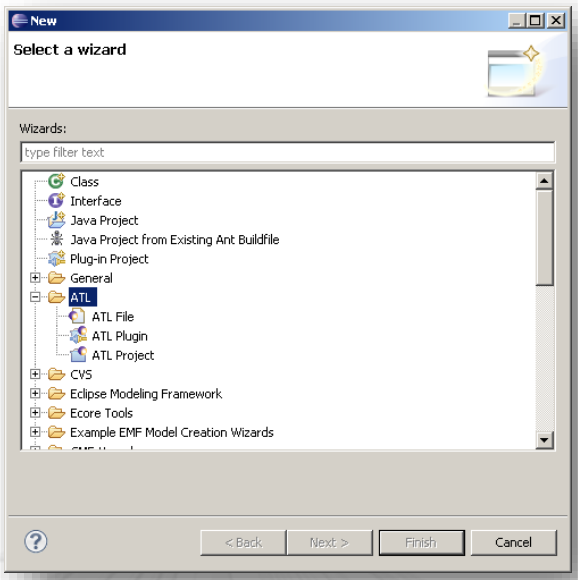
รูปที่ 86 หน้าจอแสดงส่วนเสริมทั้งหมดของ ATL

7. เมื่อทำการติดตั้งปลั๊กอินเรียบร้อยแล้วจะสามารถพัฒนาภาษาเอทีแอลบนอีคลิปส์ได้ ผลการติดตั้ง ATL จะแสดงดังรูปที่ 87



รูปที่ 87 หน้าจอแสดงรายละเอียดความสามารถของ ATL Plugin ที่ติดตั้ง

เมื่อติดตั้ง ATL Plugin เรียบร้อยแล้ว เพื่อทดสอบว่าการติดตั้งนั้นสมบูรณ์ ควรที่จะทำการรีสตาร์ท Eclipse หนึ่งครั้งแล้วทดสอบโดยไปที่ตำแหน่งที่ต้องพัฒนาภาษาเอทีแอล แล้วคลิกขวาเลือก New > Other > ในหน้าต่างโปรแกรมจะมี ATL เพิ่มเข้ามาเป็นตัวเลือกเพื่อสนับสนุนการพัฒนาภาษาเอทีแอล คลิกเลือก ATL File ดังรูปที่ 88



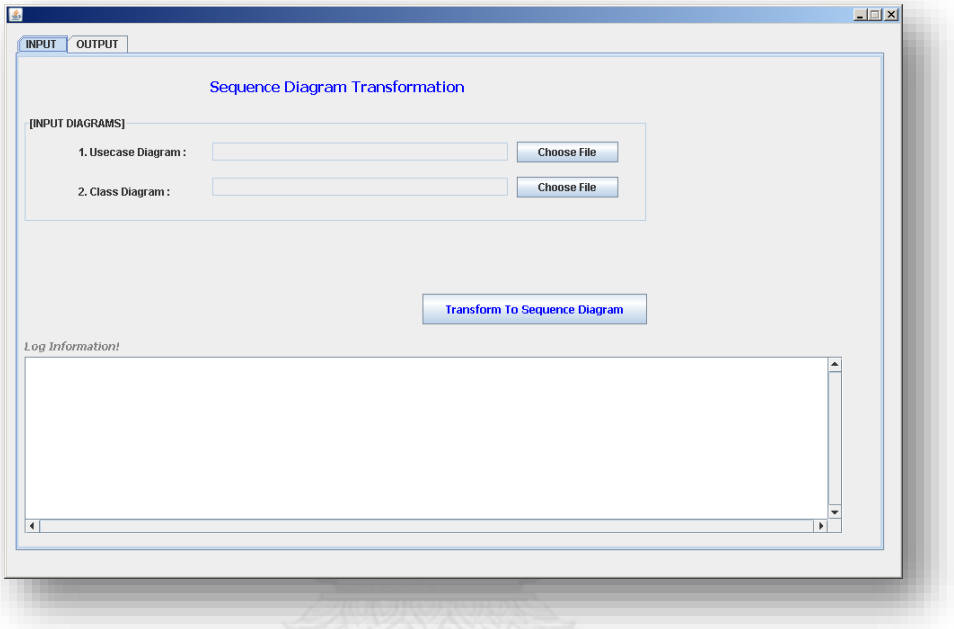
รูปที่ 88 หน้าจอทดสอบการติดตั้งเครื่องมือเพื่อสนับสนุนการพัฒนาภาษาเอทีแอล

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University

ภาคผนวก ง.

การใช้งานระบบ

ระบบการสร้างแผนภาพลำดับจากแผนภาพยูสเคสประกอบคำอธิบาย เป็นระบบที่ช่วยสร้างแผนภาพลำดับโดยอัตโนมัติจากคำอธิบายแผนภาพยูสเคสและชื่อโอเปอร์เรชั่นในแผนภาพคลาสที่อยู่ในรูปแบบเอกซ์เอ็มไอ ซึ่งหน้าจอของระบบดังรูปที่ 89



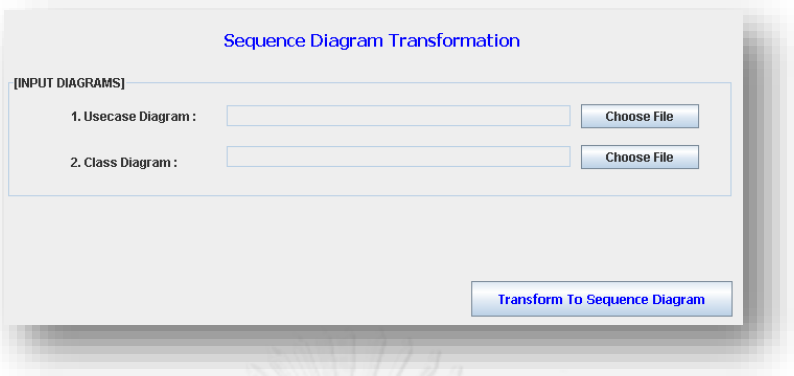
รูปที่ 89 หน้าจอการทำงานหลักของระบบการแผนภาพลำดับ

ระบบการสร้างแผนภาพลำดับ แบ่งการทำงานออกเป็น 3 ส่วน คือ ส่วนหน้าจอการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ ส่วนหน้าจอแสดงสถานะการทำงานของระบบ และส่วนของหน้าจอการส่งออกข้อมูล

1) ส่วนหน้าจอการนำเข้าข้อมูลเอกสารเอกซ์เอ็มไอ

หน้าจอการนำเข้าข้อมูลของระบบประกอบด้วย 3 ส่วน ดังรูปที่ 90 1)ส่วนนำเข้าเอกสารเอกซ์เอ็มไอของแผนภาพยูสเคสประกอบคำอธิบาย 2)ส่วนนำเข้าเอกสารเอกซ์เอ็มไอของแผนภาพคลาส 3)ส่วนสั่งการประมวลผลโปรแกรม

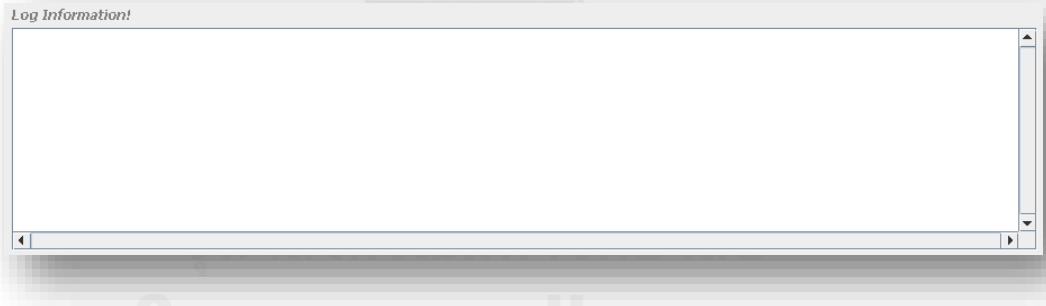
จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University



รูปที่ 90 ส่วนหน้าจอการนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ

2) ส่วนหน้าจอแสดงสถานะการทำงานของระบบ

หน้าจอการแสดงสถานะการทำงานของระบบประกอบด้วย 4 ขั้นตอน คือ 1)ส่วนแสดงสถานะการทำงานขั้นตอนการนำเข้าข้อมูล 2)ส่วนแสดงสถานะการทำงานขั้นตอนประมวลผลด้วยเทคโนโลยีเอกซ์พาสต์และการประมวลผลโดยภาษาธรรมชาติ 3)ส่วนแสดงสถานะการทำงานขั้นตอนการแปลงผลด้วยภาษาเอทีแอล 4)ส่วนแสดงสถานะการทำงานขั้นตอนการแปลงค่าการแสดงผลรูปที่ 91



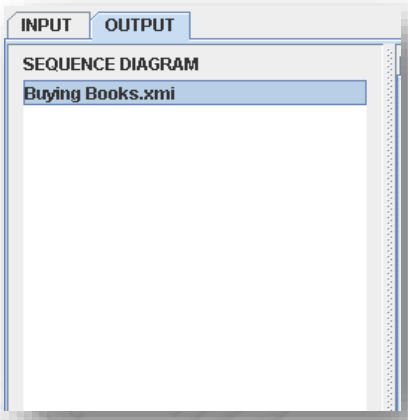
รูปที่ 91 ส่วนหน้าจอการแสดงสถานะการทำงานของระบบ

3) ส่วนของหน้าจอการส่งออกข้อมูล

หน้าจอการแสดงผลการส่งออกข้อมูล ประกอบด้วย 2 ส่วน คือ

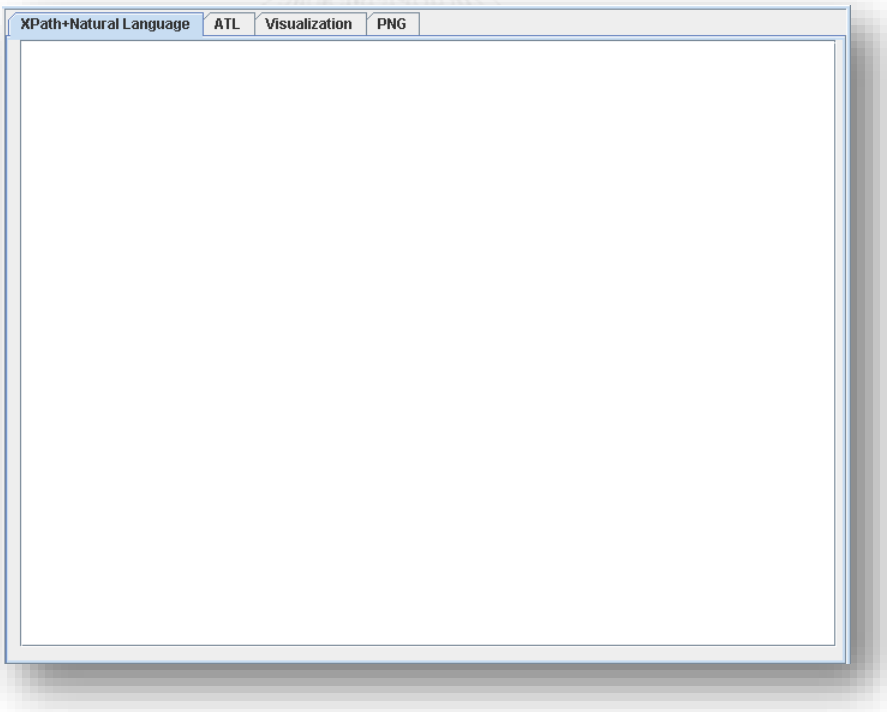
1.หน้าจอแสดงผลรายชื่อแผนภาพลำดับทั้งหมดที่ได้หลังจากการประมวลผลโปรแกรมตัวอย่างรายชื่อแผนภาพลำดับแสดงดังรูปที่ 92

CHULALONGKORN UNIVERSITY



รูปที่ 92 ส่วนแสดงผลรายการแผนภาพลำดับ

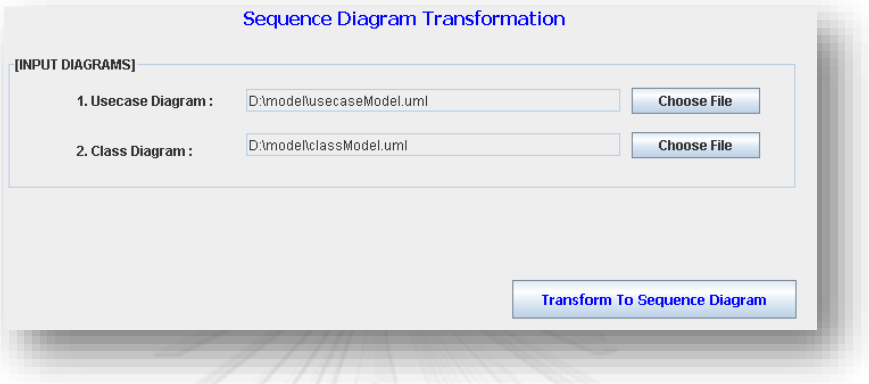
1. หน้าจอแสดงผลลัพธ์ของการประมวลผลที่ได้ในแต่ละขั้นตอนแสดงดังรูปที่ 93 ซึ่งประกอบด้วย 4 แท็บ ดังนี้
    - แท็บ XPath+Natural Language เป็นส่วนที่แสดงเอกสารเอกซ์เอ็มไอของแผนภาพยูสเคสประกอบคำอธิบาย
    - แท็บ ATL เป็นส่วนที่แสดงผลการประมวลผลโดยภาษาเอทีแอล
    - แท็บ Visualization เป็นส่วนที่แสดงผลภาษาสำหรับแสดงแผนภาพลำดับ
    - แท็บ PNG เป็นส่วนแสดงภาพแผนภาพลำดับ



รูปที่ 93 แท็บทั้งหมดในหน้าจอทำงานหลัก

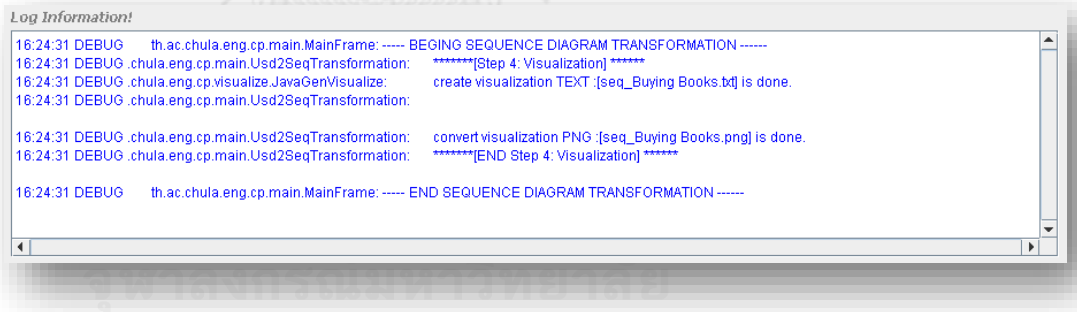
วิธีการใช้งานระบบ

ในหน้าจอการนำเข้าเอกสารเอกซ์เอ็มไอ ให้คลิกปุ่ม “Browse” เพื่อเลือกไฟล์รูปแบบเอกซ์เอ็มไอของแผนภาพยูสเคสประกอบคำอธิบาย และไฟล์ในรูปแบบเอกซ์เอ็มไอของแผนภาพคลาส ตัวอย่างการเลือกไฟล์เป็นดังรูปที่ 94



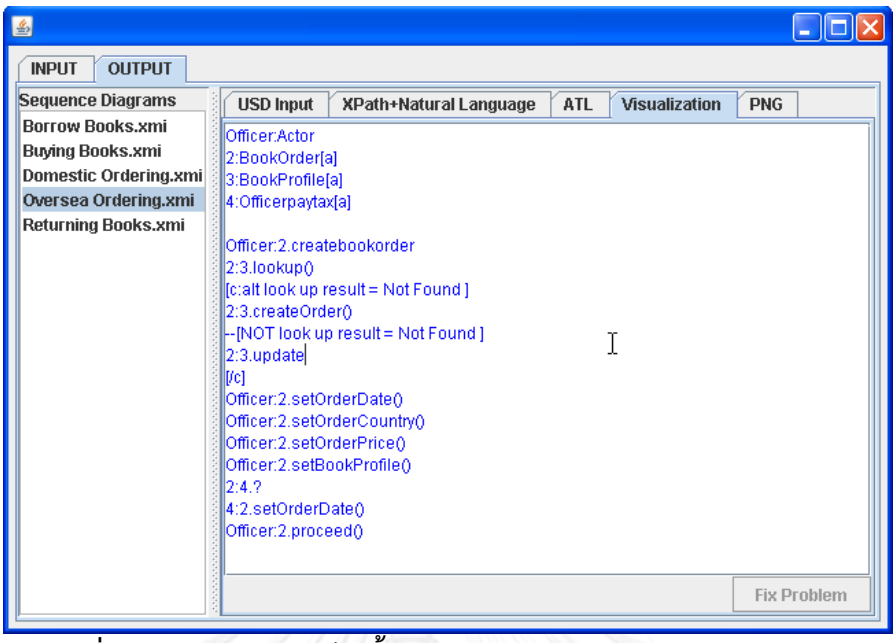
รูปที่ 94 หน้าจอนำเข้าไฟล์เอกสารเอกซ์เอ็มไอ

กดปุ่ม “Transform To Sequence Diagram” ในรูปที่ 94 แล้วตรวจสอบสถานะการทำงานของระบบว่าประมวลผลเสร็จทุกขั้นตอนแล้วหรือไม่ ดังรูปที่ 95 กรณีที่ระบบทำงานเสร็จเรียบร้อย ระบบจะแจ้งให้ทราบเป็นข้อความ “END SEQUENCE DIAGRAM TRANSFORMAION”

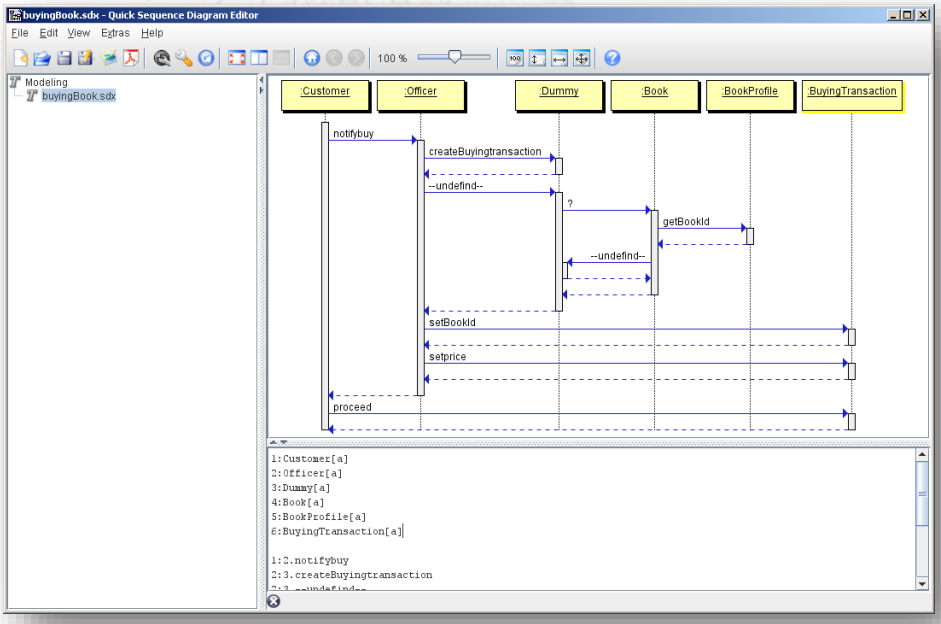


รูปที่ 95 หน้าจอแสดงสถานะการทำงานของระบบทำงานเสร็จเรียบร้อย

เมื่อระบบทำงานเรียบร้อยแล้วให้คลิกเลือกที่แท็บ “Output” เพื่อดูผลลัพธ์ของระบบ ซึ่งระบบจะทำการสร้างแผนภาพลำดับตามข้อมูลนำเข้า ให้ทำการคลิกเลือกแผนภาพลำดับแต่ละข้อแล้วเข้าไปดูผลลัพธ์แต่ละขั้นตอนของแผนภาพลำดับนั้นๆ จากตัวอย่างใน รูปที่ 96 ถ้าต้องการดูผลลัพธ์ของระบบในขั้นตอนการแปลงค่าการแสดงผล ให้เลือกที่ Buying Books > Visualization นอกจากนี้ถ้าต้องการแก้ไขแผนภาพลำดับ สามารถนำค่าผลลัพธ์ที่ได้ไปเปิดและแก้ไขค่าตามโครงสร้างภาษาของโปรแกรม Quick Sequence Editor ได้ ดังรูปที่ 97

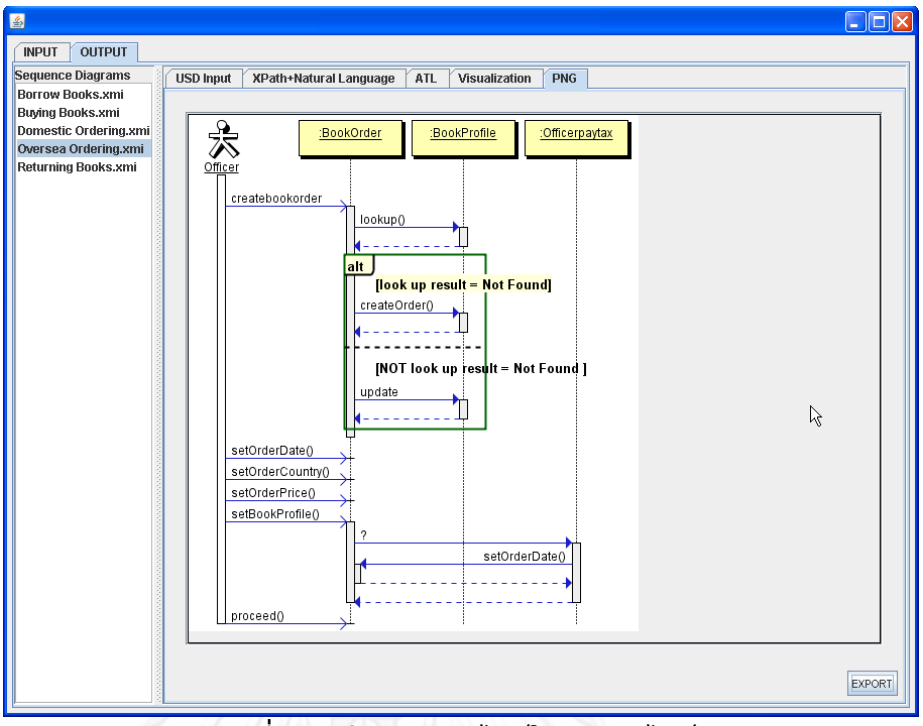


รูปที่ 96 ตัวอย่างผลลัพธ์ในขั้นตอนการแปลงค่าการแสดงผลของระบบ

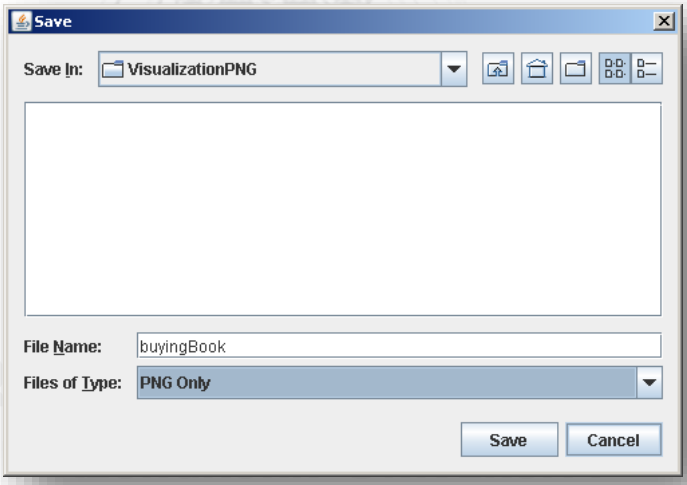


รูปที่ 97 ตัวอย่างการแสดงผลในโปรแกรม Quick Sequence Editor

ระบบสามารถส่งออกแผนภาพลำดับเป็นไฟล์รูปภาพนามสกุลพีเอ็นจี (.PNG) โดยการเลือกที่แผนภาพลำดับ > PNG >กดปุ่ม Export ดังรูปที่ 98 หลังจากนั้นให้ทำการเลือกตำแหน่งที่ต้องการเก็บไฟล์ดัง รูปที่ 99

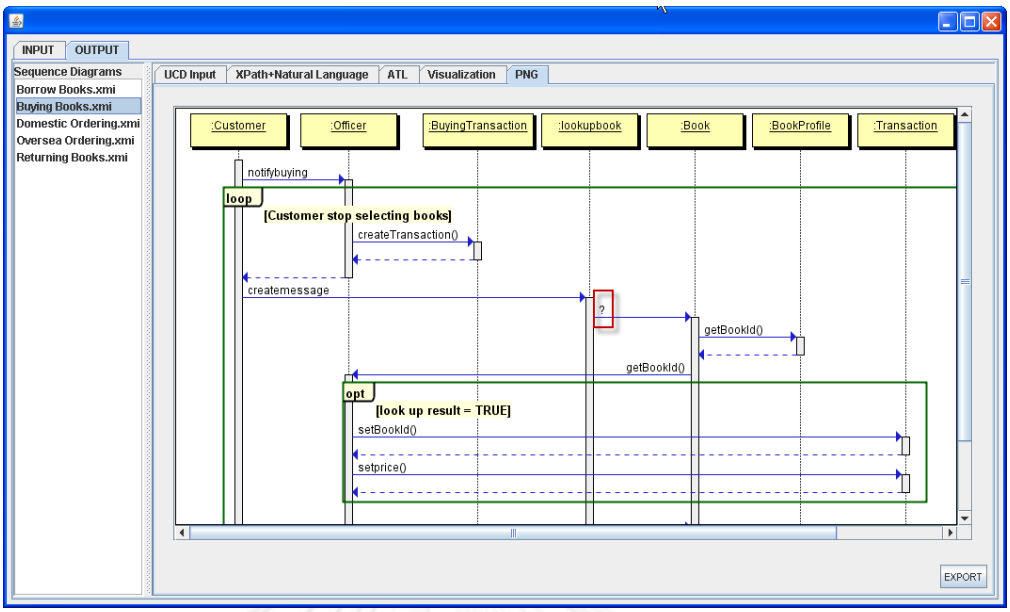


รูปที่ 98 วิธีการส่งออกไฟล์ในรูปแบบไฟล์รูปภาพ

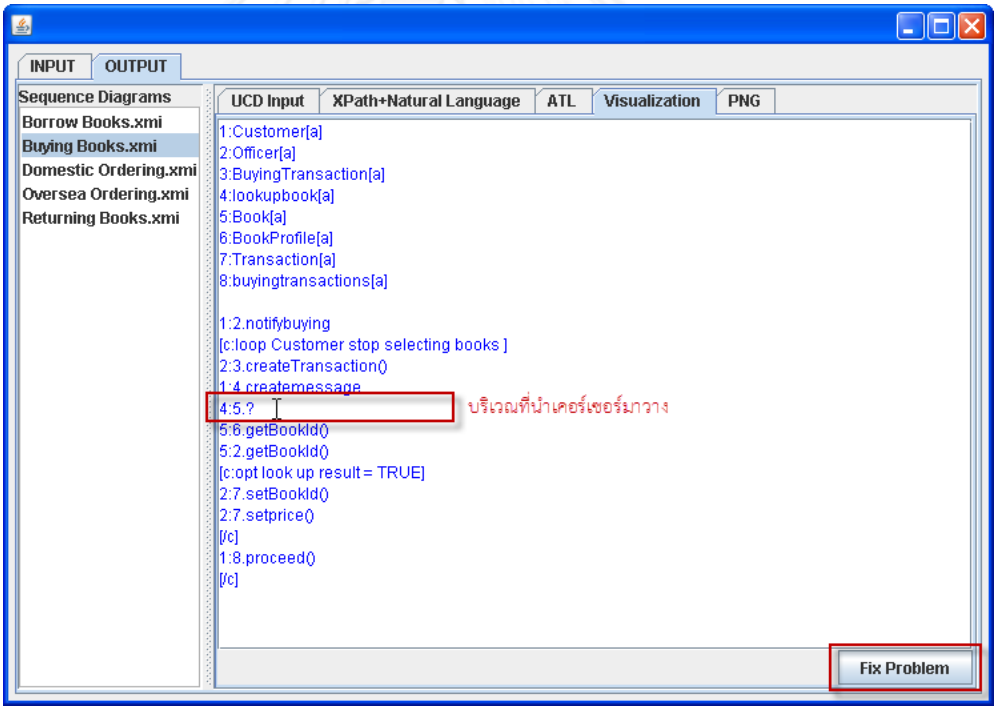


รูปที่ 99 วิธีการบันทึกไฟล์รูปภาพ

กรณีที่ระบบสร้างแผนภาพลำดับแล้วปรากฏเครื่องหมาย “?” บนส่วนที่เป็นเส้นเมสเสจ (message) ดัง รูปที่ 100 เพื่อช่วยสนับสนุนความง่ายในการใช้งานของผู้ใช้ ผู้ใช้จะต้องไปที่แท็ป Visualization แล้วค้นหาเครื่องหมาย “?” นำเคอร์เซอร์ไปชี้ที่บรรทัดที่ปรากฏเครื่องหมาย “?” แล้วกดปุ่ม Fig Problem ระบบจะค้นหาและแจ้งเตือนตำแหน่งที่ระบบประมวลผลแล้วมีข้อผิดพลาดดังรูปที่ 101

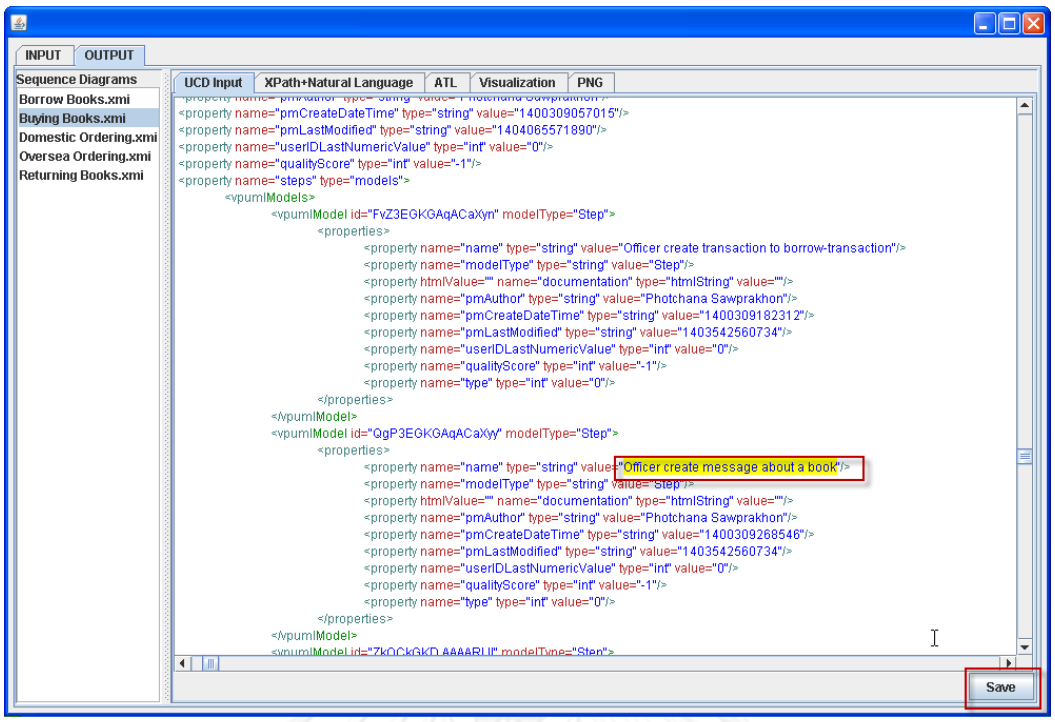


รูปที่ 100 หน้าจอแสดงเครื่องหมาย “?” บนเส้นเมสเสจ

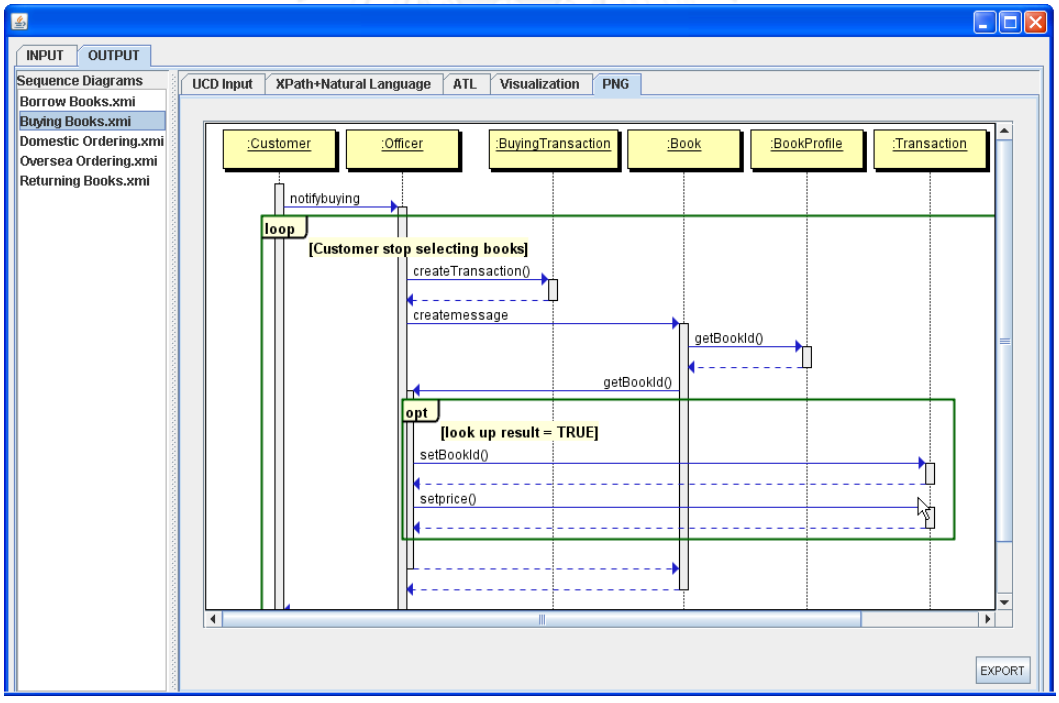


รูปที่ 101 หน้าจอการนำเคอร์เซอร์ไปชี้ที่ตำแหน่งที่ผิดพลาด

เมื่อผู้ใช้ทำการแก้ไขข้อมูลนำเข้าเรียบร้อยแล้วให้ทำการกดปุ่ม save ดังรูปที่ 102 เพื่อสั่งให้ระบบทำการประมวลผลอีกครั้ง แล้วจะได้ผลลัพธ์ดังรูปที่ 103 เป็นตำแหน่งที่แจ้งเตือนว่ามีความผิดพลาดได้ถูกแก้ไขแล้ว



รูปที่ 102 หน้าจอแสดงปุ่ม save เมื่อมีการแก้ไขข้อมูลนำเข้า



รูปที่ 103 หน้าจอแสดงผลแผนภาพลำดับที่ถูกแก้ไขสำเร็จ

ประวัติผู้เขียนวิทยานิพนธ์

นางสาว พจนา แซวประโคน เกิดเมื่อวันที่ 07 ตุลาคม พ.ศ. 2526 ที่จังหวัดบุรีรัมย์ สำเร็จการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี ในปีการศึกษา 2549 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมซอฟต์แวร์ ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2555

จุฬาลงกรณ์มหาวิทยาลัย
Chulalongkorn University