เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส

นายวรวุฒิ ประสิทธิวุฒิศักดิ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2555

A TOOL FOR CHECKING CONSISTENCY BETWEEN SEQUENCE DIAGRAMS AND A CLASS DIAGRAM

Mr. Worawut Prasitwuttisuk

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 2012

หัวข้อวิทยานิพนธ์	เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส
โดย	นายวรวุฒิ ประสิทธิวุฒิศักดิ์
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ)

..... กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.อาทิตย์ ทองทักษ์)

..... กรรมการภายนอกมหาวิทยาลัย

(ดร. เดชานุชิต กตัญญูทวีทิพย์)

วรวุฒิ ประสิทธิวุฒิศักดิ์: เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส. (A TOOL FOR CHECKING CONSISTENCY BETWEEN SEQUENCE DIAGRAMS AND A CLASS DIAGRAM) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ.ดร.วิวัฒน์ วัฒนาวุฒิ, 104 หน้า.

วิทยานิพนธ์ฉบับนี้มีวัตถุประสงค์เพื่อออกแบบและพัฒนาเครื่องมือสำหรับตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส ซึ่งก่อนการตรวจสอบความสอดคล้องจะพิจารณาการกำหนดชื่อ กำหนดส่วนประกอบต่าง ๆ ที่จำเป็น ทั้งชื่อระบบ ชื่อแผนภาพ ชื่อคลาส ชื่อส่วนประกอบต่าง ๆ ของระบบ และส่วนย่อยอย่างเมธอด คุณลักษณะ ความสัมพันธ์ การเข้าถึง และข้อความ ส่วนการตรวจสอบความสอดคล้องจะพิจารณาจากชื่อคลาส คุณลักษณะ เมธอด พารามิเตอร์และตัวแปรในเงื่อนไขบนข้อความ ลำดับการเรียกใช้เมธอดของแผนภาพลำดับ ความสัมพันธ์ที่มีต่อกันระหว่างคลาส ทั้ง 5 ประเภท รวมไปการเข้าถึงเมธอดและคุณลักษณะต่าง ๆ อีกด้วย

วิทยานิพนธ์นี้นำเสนอขั้นตอนรวมทั้งกฎสำหรับตรวจสอบข้อมูลที่จำเป็นสำหรับแผนภาพคลาส 4 ข้อ และกฎสำหรับแผนภาพลำดับ 4 ข้อ และกฎสำหรับตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส 13 ข้อ โดยข้อมูลที่นำเข้าสำหรับเครื่องมือนี้คือข้อมูลเอ็กซ์เอ็มไอที่ถูกส่งออกจากเครื่องมือวาดแผนภาพยูเอ็มแอล ซึ่งเครื่องมือที่สร้างขึ้นนี้จะสกัดเอาข้อมูลที่จำเป็นออก แล้วส่งไปตรวจสอบข้อมูลที่จำเป็นของแผนภาพทั้งสองชนิดก่อน เมื่อตรวจพบว่าครบถ้วนพอจึงจะตรวจสอบความสอดคล้องต่อไปโดยอัตโนมัติ เครื่องมือนี้ใช้ภาษาไพธอนในการพัฒนา โดยใช้กรณีศึกษาทดสอบ รายงานที่สร้างออกมาจะช่วยให้ผู้ออกแบบซอฟต์แวร์แก้ไข เพิ่มเติมรายละเอียดของแผนภาพที่ออกแบบให้มีคุณภาพมากขึ้น

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

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

ปีการศึกษา...2555.....

5270481221 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: SEQUENCE DIAGRAM / CLASS DIAGRAM / CHECKING / COMPLETENESS / CONSISTENCY

WORAWUT PRASITWUTTISUK: A TOOL FOR CHECKING COMPLETENESS AND CONSISTENCY BETWEEN SEQUENCE DIAGRAMS AND A CLASS DIAGRAM. ADVISOR: ASSOC.PROF.WIWAT VATANAWOOD, PH.D., 104 PP.

This thesis describes design and development of a tool for checking consistency between Sequence diagram and Class diagram. Before checking consistency, the diagrams will be checked by considering their names and components; system name, diagram names, class names, methods, attributes, relations, visibilities, and messages. Consistency is checked by their names of class, attribute, method, parameter and variable on message, order of calling method, five types of relations between classes, and visibility of methods and attributes.

This thesis proposes process and rules for checking necessary data of Class diagram and Sequence diagram, 4 rules each. There are also 13 rules for checking consistency between Class diagram and Sequence diagram. Input data is in XMI file format which is exported from UML drawing tool. This XMI file will be extracted to gather the needed information. Then this information will be firstly checked if there have necessary data of all diagrams. If the diagrams pass the checking, our tool will then check consistency automatically. This tool is implemented using Python and tested by using case studies. The resulting report generated help software designer adjust or improve the quality of their designs.

Department: ...Computer Engineering.... Student's signature.....
Field of Study: ...Software Engineering... Advisor's signature.....
Academic Year: ...2012.....

กิตติกรรมประกาศ

กราบขอบพระคุณแม่และพี่ชาย สำหรับกำลังใจที่ส่งมาอย่างสม่ำเสมอไม่เคยขาด ทั้งทางตรงและทางอ้อม ทั้งต่อหน้าและลับหลัง

ขอขอบพระคุณรองศาสตราจารย์ ดร.วิวัฒน์ วัฒนาวุฒิ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้โอกาส และสละเวลาให้คำปรึกษาอย่างทุ่มเทเสมอมา วิทยานิพนธ์ฉบับนี้ไม่มีทางเสร็จสมบูรณ์ได้ หากปราศจากคำปรึกษาที่มีให้มาอย่างสม่ำเสมอนั้น

ขอขอบพระคุณ รองศาสตราจารย์ ดร.ธาราทิพย์ สุวรรณศาสตร์ ผู้ช่วยศาสตราจารย์ ดร. อาทิตย์ ทองทักษ์ และดร. เดชานุชิต กตัญญูทวีทิพย์ กรรมการสอบวิทยานิพนธ์ที่ให้ข้อแนะนำและข้อคิดเห็นที่เป็นประโยชน์อย่างยิ่ง

ขอขอบคุณ สุภา ศิรินาม ที่อยู่เคียงข้างกันเสมอ ไม่ว่ายามสุขหรือยามทุกข์ หากไม่มีกำลังใจและความช่วยเหลือที่มากมายนี้ ความท้อแท้คงทำให้วิทยานิพนธ์เล่มนี้ไม่มีวันเสร็จสมบูรณ์

และขอขอบคุณทั้งผู้เกี่ยวข้องทางตรงและทางอ้อม ไม่ว่าท่านจะรู้ตัวหรือไม่ ท่านเป็นส่วนหนึ่งที่ทำให้วิทยานิพนธ์ฉบับนี้เสร็จสมบูรณ์

สารบัญ

หน้า

บทคัดย่อภาษาไทย..... ง
บทคัดย่อภาษาอังกฤษ..... จ
กิตติกรรมประกาศ..... ฉ
สารบัญ..... ช
สารบัญตาราง..... ฌ
สารบัญรูป..... ญ
บทที่ 1 บทนำ..... 1
1.1 ความเป็นมาและความสำคัญของปัญหา..... 1
1.2 วัตถุประสงค์ของการวิจัย..... 2
1.3 ขอบเขตของการวิจัย..... 2
1.4 วิธีดำเนินการวิจัย..... 3
1.5 ประโยชน์ที่ได้รับจากการวิจัย..... 3
1.6 บทความวิชาการที่ได้รับการตีพิมพ์..... 4
บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง..... 5
2.1 ทฤษฎีที่เกี่ยวข้อง..... 5
2.1.1 แผนภาพยูเอ็มแอล (UML: Unified Modeling Language)..... 5
2.1.2 แผนภาพคลาส (Class diagram)..... 7
2.1.3 แผนภาพลำดับ (Sequence diagram)..... 11
2.1.4 มาตรฐานเอ็กซ์เอ็มไอ (XMI: XML Metadata Interchange)..... 14
2.1.5 ความครบถ้วนสมบูรณ์ (Completeness)..... 17
2.1.6 ความสอดคล้อง (Consistency)..... 18
2.2 งานวิจัยที่เกี่ยวข้อง..... 19
2.2.1 Basic Rules to Build Correct UML Diagrams..... 19
2.2.2 Impact Analysis and Change Management of UML Models..... 20

2.2.3 การออกแบบและพัฒนาเครื่องมือตรวจสอบความสอดคล้อง
ระหว่างแผนภาพคลาส แผนภาพซีเควนซ์ และแผนภาพสเตทชาร์ท..... 21
บทที่ 3 การออกแบบกฎการตรวจสอบความพร้อมของข้อมูลและความสอดคล้อง..... 23
3.1 การตรวจสอบความพร้อมของข้อมูล..... 23
3.2 การตรวจสอบความสอดคล้อง..... 24
บทที่ 4 การออกแบบแบบพัฒนาเครื่องมือการตรวจสอบความสอดคล้องของแผนภาพ
คลาสและแผนภาพลำดับ..... 44
4.1 การออกแบบเครื่องมือ..... 44
4.2 การพัฒนาเครื่องมือ..... 52
บทที่ 5 การทดสอบเครื่องมือด้วยกรณีศึกษา..... 53
5.1 สภาวะที่ใช้ในการทดสอบเครื่องมือ..... 53
5.2 ขั้นตอนของการทดสอบ..... 53
5.3 กรณีศึกษาที่ใช้ในการทดสอบ..... 54
5.3.1 กรณีศึกษาที่ 1 ระบบห้องสมุด..... 54
5.3.2 กรณีศึกษาที่ 2 ระบบเอทีเอ็ม..... 67
5.3.3 กรณีศึกษาที่ 3 ระบบลงทะเบียน..... 78
บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ..... 85
6.1 สรุปผลการวิจัย..... 85
6.1.1 สรุปการตรวจสอบความพร้อมของข้อมูล..... 85
6.1.2 สรุปกฎการตรวจสอบความสอดคล้อง..... 85
6.2 ประโยชน์ของเครื่องมือ..... 86
6.3 ข้อจำกัดของเครื่องมือ..... 87
6.4 ปัญหาและอุปสรรค..... 87
6.5 ข้อเสนอแนะ..... 87
รายการอ้างอิง..... 88
ภาคผนวก..... 91
ภาคผนวก ก การใช้งานเครื่องมือ..... 92
ภาคผนวก ข รูปแบบของรายการที่แสดงผลของการตรวจสอบ..... 95
ภาคผนวก ค ตัวอย่างของรูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอ..... 100
ประวัติผู้เขียนวิทยานิพนธ์..... 104

สารบัญตาราง

หน้า

ตารางที่ 2.1 ตัวอย่างแบบจำลองเชิงโครงสร้างแบบต่าง ๆ..... 6
ตารางที่ 2.2 ตัวอย่างแบบจำลองเชิงพฤติกรรมแบบต่าง ๆ..... 7

สารบัญรูป

หน้า

รูปที่ 2.1	ตัวอย่างคลาสแสดงให้เห็นชื่อคลาส คุณลักษณะ และเมธอด.....	9
รูปที่ 2.2	ตัวอย่างความสัมพันธ์แบบแอกกรีเกชัน.....	9
รูปที่ 2.3	ตัวอย่างความสัมพันธ์แบบคอมโพซิชัน.....	10
รูปที่ 2.4	ตัวอย่างความสัมพันธ์แบบดีเพนเดนซี.....	10
รูปที่ 2.5	ตัวอย่างความสัมพันธ์ของคลาสแบบเจนเนอรอลไลเซชัน.....	11
รูปที่ 2.6	แอคเตอร์และชื่อของแอคเตอร์ของแผนภาพลำดับ.....	11
รูปที่ 2.7	ส่วนประกอบของระบบและชื่อของส่วนประกอบของระบบของแผนภาพลำดับ.....	12
รูปที่ 2.8	ส่วนประกอบของระบบและชื่อ Parti1 พร้อมระบุชื่อคลาส ClassA.....	12
รูปที่ 2.9	เส้นชีวิตและเส้นชีวิตที่ถูกทำลายแล้ว.....	12
รูปที่ 2.10	ลำดับเวลาจากบนลงล่างของแผนภาพลำดับ.....	13
รูปที่ 2.11	เหตุการณ์ สัญญาณ และข้อความของแผนภาพลำดับอย่างง่าย.....	13
รูปที่ 2.12	แถบแอคทิเวชันของแผนภาพลำดับอย่างง่าย.....	14
รูปที่ 2.13	เฟรมแบบวนซ้ำของแผนภาพลำดับอย่างง่าย.....	14
รูปที่ 2.14	แผนภาพลำดับที่สร้างจากโปรแกรม Modelio	16
รูปที่ 2.15	ส่วนหนึ่งของโค้ดเอ็กซ์เอ็มไอ.....	17
รูปที่ 3.1	คลาสและเมธอดที่ถูกเรียกใช้เป็นส่วนประกอบของระบบและข้อความในแผนภาพลำดับ.....	25
รูปที่ 3.2	การถ่ายทอดความสัมพันธ์จากคลาสแม่สู่คลาสลูก.....	27
รูปที่ 3.3	คลาสย่อยที่ส่งข้อความแบบสร้างไปยังคลาสหลักที่มีความสัมพันธ์กันแบบแอกกรีเกชัน.....	28
รูปที่ 3.4	คลาสย่อยที่ส่งข้อความแบบสร้างไปยังคลาสหลักที่มีความสัมพันธ์กันแบบคอมโพซิชัน.....	30
รูปที่ 3.5	คลาสย่อยที่ส่งข้อความแบบทำลายไปยังคลาสหลักที่มีความสัมพันธ์กันแบบคอมโพซิชัน.....	31
รูปที่ 3.6	คลาสย่อยเรียกใช้คลาสหลัก หลังจากคลาสหลักถูกทำลายไปแล้ว.....	32

รูปที่ 3.7	คลาสด้านหางลูกศรเรียกใช้คลาสด้านหัวลูกศร.....	34
รูปที่ 3.8	เมธอดที่มีการเข้าถึงแบบสาธารณะถูกคลาสอื่น ๆ เรียกใช้.....	35
รูปที่ 3.9	เมธอดที่มีการเข้าถึงแบบป้องกันและสาธารณะถูกคลาสอื่น ๆ เรียกใช้.....	36
รูปที่ 3.10	เมธอดที่มีการเข้าถึงแบบส่วนตัวถูกคลาสตัวเองเรียกใช้.....	37
รูปที่ 3.11	พารามิเตอร์ที่ปรากฏอยู่บนข้อความในแผนภาพลำดับและแผนภาพคลาส...	38
รูปที่ 3.12	เงื่อนไขบนข้อความในแผนภาพลำดับ ซึ่งประกาศเป็นคุณลักษณะ.....	40
รูปที่ 3.13	เมธอดหลักในแผนภาพหนึ่ง กลายเป็นเมธอดย่อยในอีกแผนภาพหนึ่ง.....	42
รูปที่ 4.1	แผนภาพกิจกรรมของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ.....	45
รูปที่ 4.2	แผนภาพคลาสของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ.....	46
รูปที่ 4.3	แผนภาพยูสเคสของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ.....	47
รูปที่ 4.4	แผนภาพลำดับของเครื่องมือเมื่อข้อมูลมีความพร้อมสำหรับตรวจสอบความสอดคล้อง.....	49
รูปที่ 4.5	แผนภาพลำดับของเครื่องมือเมื่อข้อมูลไม่มีความพร้อมสำหรับตรวจสอบความสอดคล้อง.....	51
รูปที่ 5.1	ส่วนการรับข้อความชื่อไฟล์เอ็กซ์เอ็มไอ.....	54
รูปที่ 5.2	แผนภาพคลาสของระบบห้องสมุด.....	56
รูปที่ 5.3	แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือ.....	57
รูปที่ 5.4	แผนภาพลำดับแสดงเหตุการณ์จองหนังสือ.....	58
รูปที่ 5.5	แผนภาพลำดับแสดงเหตุการณ์คืนหนังสือ.....	58
รูปที่ 5.6	แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือที่จองไว้.....	59
รูปที่ 5.7	แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบหนังสือที่ค้างส่ง.....	59
รูปที่ 5.8	แผนภาพลำดับแสดงเหตุการณ์ยกเลิกหนังสือที่จอง.....	60
รูปที่ 5.9	แผนภาพลำดับแสดงเหตุการณ์สั่งซื้อหนังสือ.....	60
รูปที่ 5.10	เครื่องมือแจ้งรายละเอียดของความไม่พร้อมของข้อมูลแล้วหยุดตัวเอง.....	61
รูปที่ 5.11	แผนภาพคลาสของระบบห้องสมุดที่ได้รับการแก้ไขการเข้าถึงแล้ว.....	62
รูปที่ 5.12	รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบห้องสมุด.....	64
รูปที่ 5.13	แผนภาพคลาสของระบบห้องสมุดที่ลบคลาสที่ไม่ได้ใช้ออกแล้ว.....	65

รูปที่ 5.14	แผนภาพลำดับเหตุการณ์ยืมหนังสือที่จองไว้ที่แก้ไขแล้ว	66
รูปที่ 5.15	แผนภาพลำดับเหตุการณ์สั่งซื้อหนังสือที่แก้ไขแล้ว	66
รูปที่ 5.16	แผนภาพคลาสของระบบเอทีเอ็ม	68
รูปที่ 5.17	แผนภาพลำดับแสดงเหตุการณ์ถอนเงิน	69
รูปที่ 5.18	แผนภาพลำดับแสดงเหตุการณ์ใส่รหัสบัตรผิด	70
รูปที่ 5.19	แผนภาพลำดับแสดงเหตุการณ์ทำงานปกติ	70
รูปที่ 5.20	แผนภาพลำดับแสดงเหตุการณ์ฝากเงิน	71
รูปที่ 5.21	แผนภาพลำดับแสดงเหตุการณ์โอนเงิน	71
รูปที่ 5.22	แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบยอดเงิน	72
รูปที่ 5.23	เครื่องมือรายงานผลการตรวจสอบความพร้อมของข้อมูลและสรุปรายละเอียดของแผนภาพ	73
รูปที่ 5.24	รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบเอทีเอ็ม	74
รูปที่ 5.25	แผนภาพคลาสของระบบเอทีเอ็มที่แก้ไขตามข้อ 1-3 แล้ว	76
รูปที่ 5.26	แผนภาพลำดับแสดงเหตุการณ์ทำงานปกติที่แก้ไขแล้ว	77
รูปที่ 5.27	แผนภาพลำดับแสดงเหตุการณ์ถอนเงินที่แก้ไขแล้ว	77
รูปที่ 5.28	แผนภาพคลาสของระบบลงทะเบียน	79
รูปที่ 5.29	แผนภาพลำดับแสดงเหตุการณ์ขั้นตอนการลงทะเบียน	80
รูปที่ 5.30	แผนภาพลำดับแสดงเหตุการณ์การแสดงใบเสร็จค่าลงทะเบียน	81
รูปที่ 5.31	แผนภาพลำดับแสดงเหตุการณ์การลงทะเบียนแรกเข้า	81
รูปที่ 5.32	แผนภาพลำดับแสดงเหตุการณ์การชำระเงินผ่านธนาคาร	82
รูปที่ 5.33	เครื่องมือรายงานผลการตรวจสอบความพร้อมของข้อมูลและสรุปรายละเอียดของแผนภาพ	83
รูปที่ 5.34	รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบลงทะเบียน	84
รูปที่ 5.35	แผนภาพลำดับเหตุการณ์การลงทะเบียนแรกเข้าที่แก้ไขแล้ว	84
รูปที่ ก.1	Python Shell เมื่อเริ่มเปิดใช้	92
รูปที่ ก.2	หน้าต่าง Checking.py โค้ดของเครื่องมือตรวจสอบ	93
รูปที่ ก.3	เครื่องมือเริ่มทำงาน โดยถามชื่อไฟล์เอ็กซ์เอ็มไอที่จะตรวจสอบ	93
รูปที่ ก.4	เครื่องมือทำงานและแสดงรายงานการตรวจสอบ	94
รูปที่ ค.1	โครงสร้างไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพคลาส	100

รูปที่ ค.2 โครงสร้างไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพลำดับ..... 101
รูปที่ ค.3 พารามิเตอร์ทั้ง 5 ประเภทที่โอเอ็มจีกำหนดไว้..... 102
รูปที่ ค.4 รูปแบบการกำหนดประเภทพารามิเตอร์กำหนดเอง..... 103

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

แผนภาพยูเอ็มแอล (UML: Unified Modeling Language) เป็นภาษาหรือแผนภาพที่ได้รับความนิยมมากในปัจจุบัน ซึ่งมีแผนภาพหลากหลายชนิด ใช้อธิบายระบบเดียวแต่หลากหลายมุมมองหรือหลายกิจกรรม เหมาะแก่การนำไปใช้งานหลายประเภท และแผนภาพยังออกแบบมาโดยคำนึงถึงการสื่อความหมายให้เข้าใจง่าย และง่ายต่อการแปลงเป็นโปรแกรมมาตั้งแต่แรก

และเนื่องจากมีแผนภาพที่หลากหลาย จึงอาจมีความไม่สอดคล้อง (inconsistency) เกิดขึ้นระหว่างแผนภาพต่างชนิดกัน หรือแม้แต่ชนิดเดียวกันแต่คนละกิจกรรม อีกทั้งการออกแบบมักมีระดับชั้นนามธรรม (level of abstraction) ที่แตกต่างกันไปในแต่ละระยะของการออกแบบ แม้ระหว่างการสร้างระบบหรือหลังจากสร้างระบบเสร็จแล้ว ระบบอาจมีการพัฒนา ปรับปรุง แก้ไข อยู่เสมอ ซึ่งอาจทำให้แผนภาพมีระดับชั้นนามธรรมไม่เท่ากันและขาดความสอดคล้องได้เช่นกัน ดังนั้นการตรวจสอบความครบถ้วนสมบูรณ์เชิงโครงสร้างของการออกแบบชั้นนามธรรม (structural completeness of abstraction layer design) และความสอดคล้องจึงมีความจำเป็นมาก ทั้งก่อนการนำแผนภาพไปสร้างเป็นระบบ ระหว่างสร้างระบบ และหลังจากสร้างระบบเสร็จ แล้วระบบมีการเปลี่ยนแปลง

แผนภาพลำดับ (Sequence Diagram) เป็นแผนภาพแสดงการปฏิสัมพันธ์ (Interaction Diagram) แบบหนึ่ง ซึ่งเป็นที่นิยมที่สุด ใช้อธิบายการสื่อสารระหว่างส่วนต่าง ๆ โดยมุ่งไปที่ลำดับของการแลกเปลี่ยนข้อมูลไปยังส่วนต่าง ๆ ของซอฟต์แวร์[1] ซึ่งมีประโยชน์มากต่อการออกแบบและวิเคราะห์ระบบ เพราะมันแสดงให้เห็นความเป็นไปของระบบ แผนภาพลำดับมีความเกี่ยวข้องกับแผนภาพคลาส (Class Diagram) ซึ่งเป็นแบบจำลองเชิงโครงสร้าง (Structural Model) ทั้งในแง่ของคลาสต่าง ๆ ความสัมพันธ์ระหว่างคลาส ไปจนถึงเมธอดที่แต่ละคลาสมี แม้จะเป็นแผนภาพคนละชนิดก็ตาม

งานวิจัยนี้มุ่งเน้นไปที่การตรวจสอบความสอดคล้องกันระหว่างแผนภาพลำดับและแผนภาพคลาส โดยใช้แผนภาพคลาสเป็นหลักตรวจสอบว่าแผนภาพลำดับมีความสอดคล้องหรือไม่ แต่เพื่อป้องกันกรณีที่มีข้อมูลไม่เพียงพอแก่การตรวจสอบความสอดคล้องกัน ก่อนจะตรวจสอบความสอดคล้องกันจำเป็นจะต้องตรวจสอบความพร้อมของข้อมูลให้ผ่านเสียก่อน

1.2 วัตถุประสงค์ของงานวิจัย

เพื่อออกแบบและพัฒนาเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาสในรูปแบบเอ็กซ์เอ็มไอ

1.3 ขอบเขตของการวิจัย

1.3.1 งานวิจัยนี้ตรวจสอบความถูกต้องของแผนภาพลำดับตามมาตรฐานยูเอ็มแอลรุ่น 2.3

1.3.2 ข้อมูลที่จะนำเข้ามาตรวจสอบความถูกต้องด้วยโปรแกรมตรวจสอบ จะต้องอยู่ในรูปของเอ็กซ์เอ็มไอตามมาตรฐานเอ็กซ์เอ็มไอรุ่น 2.1 ขึ้นไป

1.3.3 การตรวจสอบแผนภาพลำดับผ่านมาตรฐานเอ็กซ์เอ็มไอนั้น จะถือว่าแผนภาพลำดับและแผนภาพคลาสจะต้องตรวจสอบความครบถ้วนสมบูรณ์จนผ่านเสียก่อน จึงจะตรวจสอบความสอดคล้องระหว่างแผนภาพได้

1.3.4 ข้อมูลแผนภาพคลาสที่จะนำมาช่วยตรวจสอบแผนภาพลำดับ จะถือว่าต้องถูกต้องอยู่แล้วในแง่โครงสร้างและความสัมพันธ์

1.3.5 แผนภาพลำดับและแผนภาพคลาสที่นำมาตรวจสอบต้องเป็นแผนภาพที่ระดับเดียวกันของระบบเดียวกันเท่านั้น และเป็นระดับที่มีความละเอียดถึงขั้นมีเมธอดของแต่ละคลาส

1.3.6 สามารถตรวจสอบความครบถ้วนสมบูรณ์ของแผนภาพลำดับและแผนภาพคลาสได้ดังนี้

1.3.6.1 การกำหนดชื่อ

1.3.6.2 การกำหนดรายละเอียดส่วนประกอบส่วนต่าง ๆ ของแผนภาพ

1.3.6.3 ความสัมพันธ์ระหว่างส่วนประกอบส่วนต่าง ๆ ของแผนภาพ

1.3.7 สามารถตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับกับแผนภาพคลาสได้แก่

1.3.7.1 ส่วนประกอบส่วนต่าง ๆ ของระบบ ได้แก่ ชื่อระบบ แอคเตอร์ ส่วนประกอบของระบบ เมธอดของข้อความ เมธอดของคลาสและการเข้าถึง

1.3.7.2 ความสัมพันธ์ของส่วนประกอบส่วนต่าง ๆ ของระบบ ได้แก่ ข้อความและความสัมพันธ์ระหว่างคลาส

1.3.7.3 ตัวแปร หรือพารามิเตอร์ของส่วนประกอบส่วนต่าง ๆ ของระบบ ได้แก่ตัวแปรหรือพารามิเตอร์ของข้อความ คุณลักษณะและเมธอด

1.3.7.4 สิทธิ์การเข้าถึงคุณสมบัติหรือความสามารถของส่วนประกอบส่วนต่าง ๆ ของระบบ

1.3.8 มีการทดสอบด้วยกรณีศึกษาอย่างน้อย 3 ระบบ

1.4 วิธีดำเนินการวิจัย

1.4.1 ศึกษางานวิจัยและเอกสารที่เกี่ยวข้อง

1.4.2 ศึกษาความสอดคล้องของการเขียนแผนภาพลำดับและแผนภาพคลาสจากหนังสือหรืองานวิจัย

1.4.3 ออกแบบกฎตรวจสอบความพร้อมของข้อมูลความสอดคล้องระหว่างแผนภาพลำดับกับแผนภาพคลาส

1.4.4 ศึกษาโครงสร้างของภาษาเอ็กซ์เอ็มไอ

1.4.5 ออกแบบและสร้างส่วนสกัดข้อมูลจากข้อมูลเอ็กซ์เอ็มไอ พร้อมทั้งส่วนเก็บข้อมูลที่สกัดได้

1.4.6 ออกแบบและสร้างส่วนตรวจสอบความพร้อมของข้อมูลและความสอดคล้องตามกฎที่สร้างขึ้น

1.4.7 ทดสอบและปรับปรุงซอฟต์แวร์

1.4.8 ทดสอบด้วยกรณีศึกษา

1.4.9 สรุปผลและเขียนวิทยานิพนธ์

1.5 ประโยชน์ที่ได้รับจากการวิจัย

1.5.1 ได้เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาส

1.5.2 ช่วยให้ผู้ออกแบบแผนภาพลำดับสามารถตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาสได้

1.5.3 ซอฟต์แวร์สามารถลดเวลาที่ใช้ในการตรวจสอบความสอดคล้องระหว่างแผนภาพลำดับและแผนภาพคลาสลงได้

1.6 บทความวิชาการที่ได้รับการตีพิมพ์

"On-the-Fly Consistency Checking between Class Diagram and Sequence Diagrams using Grammar Based Approach" งานประชุมวิชาการ "The 17th International Annual Symposium on Computational Science and Engineering (ANSCSE 17)" ระหว่างวันที่ 27-29 มีนาคม 2556 ณ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น จ.ขอนแก่น

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีที่เกี่ยวข้องได้แก่ แผนภาพยูเอ็มแอล แผนภาพคลาส แผนภาพลำดับ มาตรฐานเอ็กซ์เอ็มไอ ความครบถ้วนสมบูรณ์ และความสอดคล้อง รวมทั้งงานวิจัยที่เกี่ยวข้องดังมีรายละเอียดดังต่อไปนี้

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 แผนภาพยูเอ็มแอล (UML: Unified Modeling Language) [1, 2, 3, 4, 5]

แผนภาพยูเอ็มแอลเป็นภาษามาตรฐานของการออกแบบเพื่อการพัฒนาซอฟต์แวร์หรือระบบโดยเฉพาะ ใช้ในการอธิบาย แสดงรายละเอียด หรือจำลองการทำงานของซอฟต์แวร์หรือระบบ แบ่งออกเป็นสองประเภทหลัก ๆ และแต่ละประเภทก็จะแบ่งย่อยลงไปอีก ซึ่งแต่ละประเภทจะมีความสัมพันธ์ต่อกันอย่างหลวม ๆ และถูกนำไปใช้งานต่างกันไปตามจุดประสงค์ดังนี้

2.1.1.1 แบบจำลองเชิงโครงสร้าง (Structural Model) ใช้จำลองหรือแสดงโครงสร้างโดยรวมและส่วนประกอบของระบบตลอดจนความสัมพันธ์ระหว่างส่วนต่าง ๆ ของระบบ แบบจำลองเชิงโครงสร้างมีหลายแบบ ดังจะยกตัวอย่างมา 4 แบบตามตารางที่ 2.1

ตารางที่ 2.1 ตัวอย่างแบบจำลองเชิงโครงสร้างแบบต่าง ๆ

แบบจำลองเชิงโครงสร้างแบบต่าง ๆ	
แผนภาพคลาส (Class Diagram)	ใช้แสดงโครงสร้างและความสัมพันธ์ของระบบ แสดงให้เห็นว่าระบบประกอบไปด้วยส่วนประกอบ (class) อะไรบ้าง แต่ละส่วนประกอบมีความสัมพันธ์กันอย่างไร
แผนภาพวัตถุ (Object Diagram)	ใช้แสดงวัตถุ (object) ที่เป็นส่วนประกอบของคลาส (class) และความสัมพันธ์ระหว่างวัตถุ แสดงให้เห็นโครงสร้างของคลาส
แผนภาพคอมโพเนนท์ (Component Diagram)	แสดงส่วนสำคัญของระบบ (component) และส่วนต่อประสาน (interface) ที่ถูกใช้ในการติดต่อระหว่างกัน
แผนภาพดีพลอยเมนต์ (Deployment Diagram)	แสดงตำแหน่งการทำงานของระบบ เมื่อถูกใช้งานจริง

2.1.1.2 แบบจำลองเชิงพฤติกรรม (Behavioral Model) หรือแบบจำลองเชิงพลวัตร (Dynamic Model) ใช้จำลองหรือแสดงพฤติกรรม หรือความเป็นไปของระบบตั้งแต่เริ่มจนสิ้นสุด หรืออาจใช้แสดงเฉพาะกระบวนการใดกระบวนการหนึ่งที่สนใจโดยเฉพาะ แบบจำลองเชิงพฤติกรรมมีหลายแบบเช่นกัน ดังจะยกตัวอย่าง 5 แบบ ตามตารางที่ 2.2

ตารางที่ 2.2 ตัวอย่างแบบจำลองเชิงพฤติกรรมแบบต่าง ๆ

แบบจำลองเชิงพฤติกรรมแบบต่าง ๆ	
แผนภาพกิจกรรม (Activities Diagram)	ใช้แสดงกิจกรรมที่เป็นไปในระบบจากกิจกรรมหนึ่งไปยังอีกกิจกรรมหนึ่ง
แผนภาพคอมมิวนิเคชัน (Communication Diagram)	ใช้แสดงวิธีที่วัตถุปฏิสัมพันธ์กัน (interaction) และการเชื่อมโยงที่การปฏิสัมพันธ์นั้นต้องการแสดงข้อมูลที่ส่งหากัน ไม่ได้แสดงลำดับการส่งข้อมูล
แผนภาพสถานะ (State Diagram)	ใช้แสดงสถานะของวัตถุตลอดช่วงชีวิตของมัน (lifetime) และเหตุการณ์ที่จะทำให้มันเปลี่ยนสถานะ
แผนภาพลำดับ (Sequence Diagram)	ใช้แสดงปฏิสัมพันธ์ระหว่างวัตถุ โดยเน้นไปที่การแสดงลำดับการปฏิสัมพันธ์ก่อน-หลัง ไม่ได้แสดงข้อมูลที่ส่งหากัน
แผนภาพยูสเคส (Use Case Diagram)	ใช้แสดงปฏิสัมพันธ์ระหว่างระบบกับผู้ใช้ หรือสิ่งอื่นภายนอกระบบ

แผนภาพลำดับจัดอยู่ในประเภทแบบจำลองเชิงพฤติกรรมและยังเป็นแผนภาพแสดงการปฏิสัมพันธ์แบบหนึ่งที่เป็นที่นิยมที่สุดอีกด้วย ส่วนแผนภาพคลาสเป็นแบบจำลองเชิงโครงสร้างดังจะกล่าวถึงรายละเอียดของแผนภาพทั้งสองในหัวข้อต่อไป

2.1.2 แผนภาพคลาส (Class diagram) [6, 7, 8]

เป็นแบบจำลองเชิงโครงสร้าง ใช้แสดงโครงสร้างของระบบ แสดงให้เห็นว่าระบบประกอบไปด้วยส่วนประกอบ (Class) อะไรบ้าง แต่ละส่วนประกอบมีความสัมพันธ์กันอย่างไร โดยแต่ละคลาสจะประกอบด้วย ชื่อคลาส (Class Name) คุณลักษณะ (Attribute) และเมธอด (Method) และมีความสัมพันธ์กันแบบต่าง ๆ ดังรายละเอียดดังนี้

2.1.2.1 ชื่อคลาส เป็นการบอกชื่อส่วนประกอบแต่ละส่วนของระบบ ซึ่งแต่ละส่วนประกอบจะต้องมีชื่อไม่ซ้ำกัน ชื่อคลาสจะใช้อักขระทั่วไป (String) ในการตั้งชื่อ

2.1.2.2 คุณลักษณะ แสดงคุณลักษณะที่คลาสนั้น ๆ มี โดยคลาสหนึ่ง ๆ จะมีคุณลักษณะหรือไม่ก็ได้ และสามารถมีคุณลักษณะได้มากกว่า 1 ค่า คุณลักษณะสามารถระบุการเข้าถึง (Visibility) ชนิดของข้อมูล (Data Type) และค่าเริ่มต้น (Initial Value) ได้ โดยรูปแบบของคุณลักษณะคือ Visibility Attribute_Name:Data_Type = Initial_Value

1. การเข้าถึงเป็นตัวเลือกที่จะแสดงให้เห็นว่าคุณลักษณะหรือเมธอดนั้น ๆ จะถูกมองเห็นหรือเรียกใช้ได้หรือไม่ โดยใครบ้าง ซึ่งจะแบ่งออกเป็น 3 ประเภทหลัก ๆ ได้แก่

- สาธารณะ (Public) มีสัญลักษณ์คือ “+” เป็นตัวเลือกที่เปิดที่สุด สามารถมองเห็นและเรียกใช้ได้จากคลาสอื่น ๆ ที่มีความสัมพันธ์กันได้ทั้งหมด
- ป้องกัน (Protect) มีสัญลักษณ์คือ “#” คลาสอื่น ๆ ไม่สามารถมองเห็นและเรียกใช้ได้ ยกเว้นตัวมันเองหรือคลาสที่สืบทอด (Inherit) หรือคลาสลูก (Sub Class) เท่านั้น
- ส่วนตัว (Private) มีสัญลักษณ์คือ “-” เป็นตัวเลือกที่ปิดที่สุด คลาสอื่น ๆ จะไม่สามารถมองเห็นหรือเรียกใช้ได้ ยกเว้นตัวมันเองเท่านั้น

2. ชื่อคุณลักษณะ บอกชื่อคุณลักษณะนั้น ๆ ซึ่งต้องไม่ซ้ำกับคุณลักษณะอื่น ๆ ที่มีในแผนภาพคลาส

3. ชนิดของข้อมูล บอกว่าคุณลักษณะนั้นเป็นข้อมูลชนิดใด อาจเป็นชนิดข้อมูลทั่ว ๆ ไป เช่น จำนวนเต็ม (Integer) ทศนิยม (Float) อักขระ (String) ตรรกะ (Boolean) เป็นต้น หรืออาจเป็นตัวแปรแบบกำหนดเองก็ได้ นอกจากนั้นยังสามารถใช้ชื่อคลาสเป็นชนิดของข้อมูลได้อีกด้วย

4. ค่าเริ่มต้น เป็นการกำหนดค่าเริ่มต้นให้กับคุณลักษณะ ซึ่งค่าเริ่มต้นนี้ไม่จำเป็นต้องกำหนดไว้ก็ได้

2.1.2.3 เมธอด แสดงพฤติกรรมของคลาส หรือสิ่งที่คลาสสามารถทำได้ โดยคลาสหนึ่ง ๆ จะมีเมธอดหรือไม่มีก็ได้ และสามารถมีเมธอดได้มากกว่า 1 ค่า เมธอดสามารถระบุการเข้าถึง ชนิดข้อมูลที่คืนค่า (Return Type) พารามิเตอร์ (Parameter) และชนิดของพารามิเตอร์ (Parameter Type) ได้ โดยรูปแบบของเมธอดคือ Visibility Method_Name (Parameter_List): Return_Type ดังตัวอย่างในรูปที่ 2.1

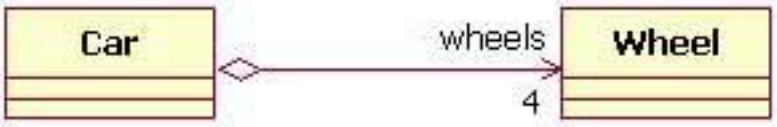
รูปที่ 2.1 ตัวอย่างคลาสแสดงให้เห็นชื่อคลาส คุณลักษณะ และเมธอด [8]

โดยปกติแล้วเมธอดจะต้องมีชื่อไม่ซ้ำกัน แต่มีกรณียกเว้นกรณีหนึ่งคือ overloading method ซึ่งมีชื่อซ้ำกันอยู่ในคลาสหนึ่ง แต่จำนวนหรือชนิดของพารามิเตอร์หรือชนิดข้อมูลจะต้องไม่เหมือนกัน

2.1.2.4 ความสัมพันธ์ เป็นการแสดงให้เห็นการติดต่อกันของคลาส แสดงด้วยเส้นเชื่อมต่อระหว่างคลาส เส้นเชื่อมต่ออาจมีลูกศรอยู่ปลายเส้นด้านใดด้านหนึ่งแสดงความสัมพันธ์ทางเดียว (Unidirectional) แต่ถ้าไม่มีลูกศรจะถือเป็นความสัมพันธ์สองทาง (Bidirectional) ความสัมพันธ์ทางเดียวหมายถึงคลาสด้านหางลูกศรสามารถเรียกใช้ คลาสด้านหัวลูกศรได้ทางเดียว ส่วนลักษณะเส้นจะแสดงแบบของความสัมพันธ์ดังนี้

1. แอซโซซิเอชัน (Association) แสดงด้วยเส้นทึบ แสดงความสัมพันธ์ที่คลาสหนึ่งสามารถเรียกใช้คุณสมบัติหรือเมธอดของอีกคลาสหนึ่งได้ ความสัมพันธ์แบบแอซโซซิเอชันอาจมีความสัมพันธ์ทางเดียวได้ โดยใช้หัวลูกศรที่ปลายข้างหนึ่ง

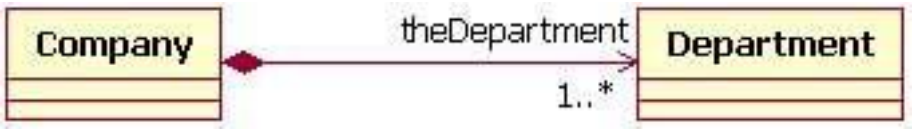
2. แอกกรีเกชัน (Aggregation) แสดงด้วยเส้นทึบ ปลายข้างหนึ่งมีสี่เหลี่ยมขนมเปียกปูนโปร่ง แสดงความสัมพันธ์แบบคลาสหนึ่งประกอบด้วยคลาสหนึ่ง โดยคลาสที่ใหญ่กว่าจะอยู่ทางด้านสี่เหลี่ยมขนมเปียกปูนโปร่ง โดยคลาสเล็กหนึ่ง ๆ สามารถเป็นส่วนประกอบของคลาสใหญ่ได้หลายคลาส ความสัมพันธ์แบบนี้จึงเป็นความสัมพันธ์แบบหลวม ๆ แม้คลาสใหญ่จะถูกทำลาย คลาสเล็กก็ยังคงอยู่ได้ และความสัมพันธ์ประเภทนี้มีทิศทางของความสัมพันธ์ได้ ดังตัวอย่างในรูปที่ 2.2



รูปที่ 2.2 ตัวอย่างความสัมพันธ์แบบแอกกรีเกชัน [8]

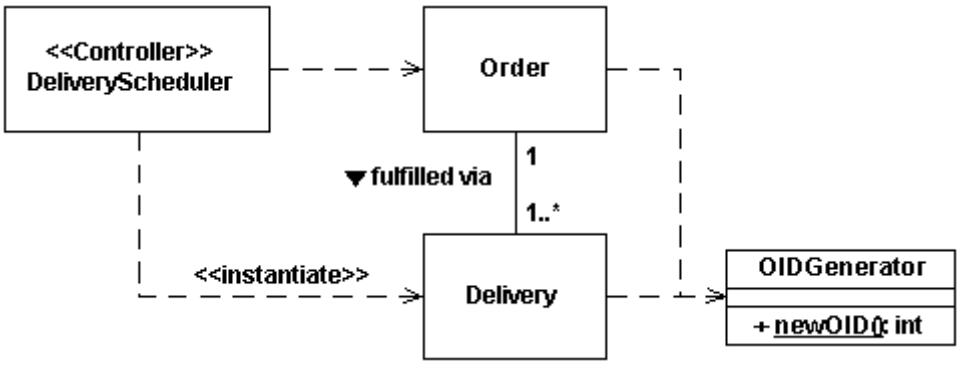
3. คอมโพซิชัน (Composition) แสดงด้วยเส้นทึบ ปลายข้างหนึ่งมีสี่เหลี่ยมขนมเปียกปูนทึบ แสดงความสัมพันธ์แบบเป็นเจ้าของ เป็นความสัมพันธ์ที่หนาแน่นที่สุด โดยคลาสเล็กหนึ่ง

คลาสจะอยู่ภายใต้คลาสใหญ่ได้เพียงคลาสเดียว และถ้าคลาสที่เป็นเจ้าของถูกทำลาย คลาสเล็กจะถูกทำลายลงไปด้วย ความสัมพันธ์ประเภทนี้มีทศทางของความสัมพันธ์ได้ ดังตัวอย่างในรูปที่ 2.3



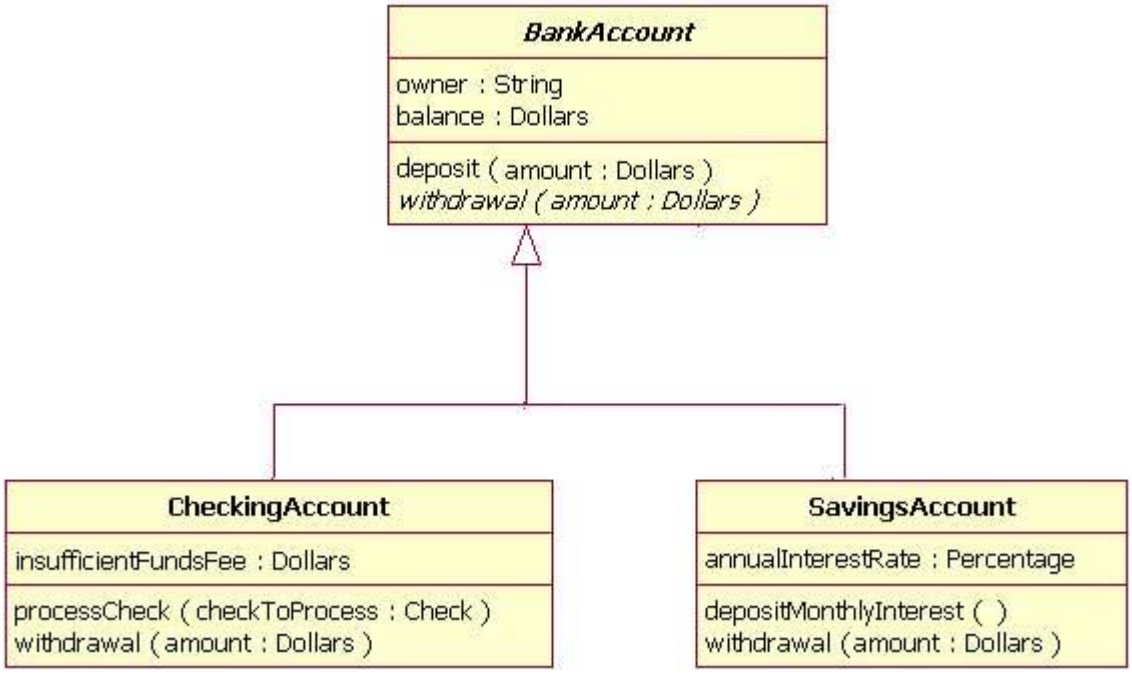
รูปที่ 2.3 ตัวอย่างความสัมพันธ์แบบคอมโพซิชัน [8]

4. ดีเพนเดนซี (Dependency) แสดงด้วยเส้นประ ปลายข้างหนึ่งมีหัวลูกศร แสดงความสัมพันธ์แบบเรียกใช้ หัวลูกศรจะชี้ไปคลาสที่ถูกเรียกใช้ การเปลี่ยนแปลงของคลาสที่ถูกเรียกใช้จะส่งผลกระทบไปถึงคลาสที่เรียกใช้ แต่จะไม่ส่งผลกระทบในทางกลับกันดังตัวอย่างในรูปที่ 2.4



รูปที่ 2.4 ตัวอย่างความสัมพันธ์แบบดีเพนเดนซี [7]

5. เจนเนอรอลไลเซชัน (Generalization) แสดงด้วยเส้นทึบ ปลายข้างหนึ่งมีสามเหลี่ยม แสดงความสัมพันธ์แบบถ่ายทอดคุณลักษณะจากคลาสแม่ โดยคลาสแม่คือคลาสที่อยู่ด้านสามเหลี่ยม จะเป็นคลาสแบบแอ็บสแตร็ค (Abstract) คลาสลูกที่อยู่อีกด้านหนึ่งจะดึงคุณสมบัติของคลาสแม่ไปใช้ได้ หากคลาสแม่ถูกทำลาย คลาสลูกจะยังอยู่ เพียงแต่คุณสมัติที่สืบทอดมาจะหายไปด้วย คุณสมบัติที่สืบทอดมาได้นั้น รวมไปถึงความสัมพันธ์กับคลาสอื่น ๆ ความเป็นคลาสหลัก-คลาสย่อยกับคลายอื่น ๆ อีกด้วย ดังตัวอย่างในรูปที่ 2.5

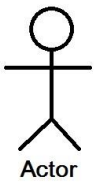


รูปที่ 2.5 ตัวอย่างความสัมพันธ์ของคลาสแบบเจนเนอรอลไลเซชัน [8]

2.1.3 แผนภาพลำดับ (Sequence diagram) [2, 9, 10]

แผนภาพลำดับเป็นแผนภาพแสดงการปฏิสัมพันธ์ที่เป็นที่นิยมที่สุด ใช้อธิบายการสื่อสารระหว่างส่วนต่าง ๆ โดยมุ่งเน้นไปที่ลำดับของการแลกเปลี่ยนข้อมูลข่าวสารไปยังส่วนต่าง ๆ ของระบบ ไม่ได้แสดงเวลาที่ใช้ในการสื่อสาร แผนภาพลำดับมีส่วนประกอบสำคัญ 7 ส่วนดังนี้

2.1.3.1 แอคเตอร์ (Actor) คือผู้ใช้ที่กระทำปฏิสัมพันธ์ระหว่างกัน หรือระหว่างระบบ อาจเป็นผู้ใช้ หรือผู้อื่นที่เกี่ยวข้องกับระบบก็ได้ วาดแทนด้วยรูปคนและระบุชื่อแอคเตอร์ไว้ด้านล่าง ดังรูปที่ 2.6

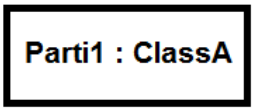


รูปที่ 2.6 แอคเตอร์และชื่อของแอคเตอร์ของแผนภาพลำดับ

2.1.3.2 ส่วนประกอบของระบบ (Participant) คือส่วนหนึ่งส่วนใดของระบบที่กระทำปฏิสัมพันธ์ระหว่างกัน หรือระหว่างแอคเตอร์ วาดแทนด้วยกรอบสี่เหลี่ยมผืนผ้า และระบุชื่อไว้ในกรอบสี่เหลี่ยมดังรูปที่ 2.7

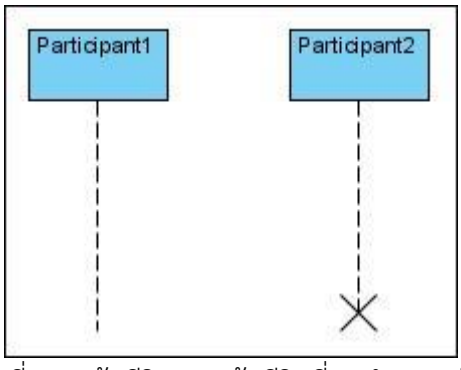
รูปที่ 2.7 ส่วนประกอบของระบบและชื่อของส่วนประกอบของระบบของแผนภาพลำดับ

ส่วนประกอบของระบบคือคลาสในแผนภาพคลาส ดังนั้นบางครั้งจะระบุชื่อคลาสลงไปด้วย โดยมีรูปแบบ ParticipantName : ClassName ดังรูปที่ 2.8



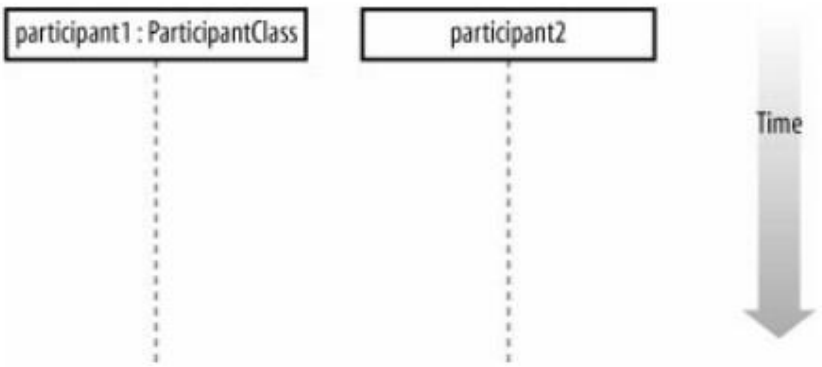
รูปที่ 2.8 ส่วนประกอบของระบบและชื่อ Parti1 พร้อมระบุชื่อคลาส ClassA

2.1.3.3 เส้นชีวิต (Lifeline) แสดงช่วงเวลาที่ส่วนประกอบของระบบ หรือแอคเตอร์ยังคงมีบทบาทอยู่ในระบบ โดยอาจจะทำงานอยู่ (Active or Busy) หรือไม่ก็ได้ ใช้เส้นประลากลงมาตั้งแต่ส่วนประกอบของระบบ หรือแอคเตอร์ตามแนวตั้ง บางส่วนของระบบอาจถูกสร้างขึ้นมาใช้ชั่วคราวและถูกทำลายไป (Terminate) ก็ใช้เครื่องหมายกากบาทวาดทับปลายเส้นประ เพื่อแสดงให้เห็นว่าหมดบทบาทแล้ว ณ ตำแหน่งนั้น ดังรูปที่ 2.9



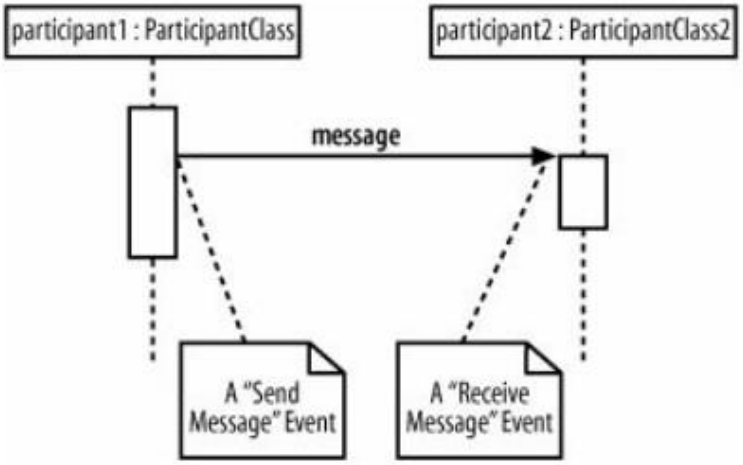
รูปที่ 2.9 เส้นชีวิตและเส้นชีวิตที่ถูกทำลายแล้ว

2.1.3.4 เวลา (Time) แผนภาพลำดับเน้นที่การแสดงลำดับการปฏิสัมพันธ์ก่อน-หลัง เวลาจึงเป็นสิ่งสำคัญ แต่แผนภาพลำดับไม่ได้แสดงเวลาที่ใช้ดังกล่าวไปแล้วในตอนต้น ดังนั้นแผนภาพลำดับจึงไม่มีเวลาบอกชัดเจนเป็นตัวเลข และระยะห่างในแผนภาพไม่ได้บอกขนาดของช่วงเวลา เพียงแต่บอกว่าลำดับการทำงานจะไล่จากบนลงล่าง หรือด้านบนทำงานก่อนด้านล่างนั่นเอง ตามรูปที่ 2.10



รูปที่ 2.10 ลำดับเวลาจากบนลงล่างของแผนภาพลำดับ[2]

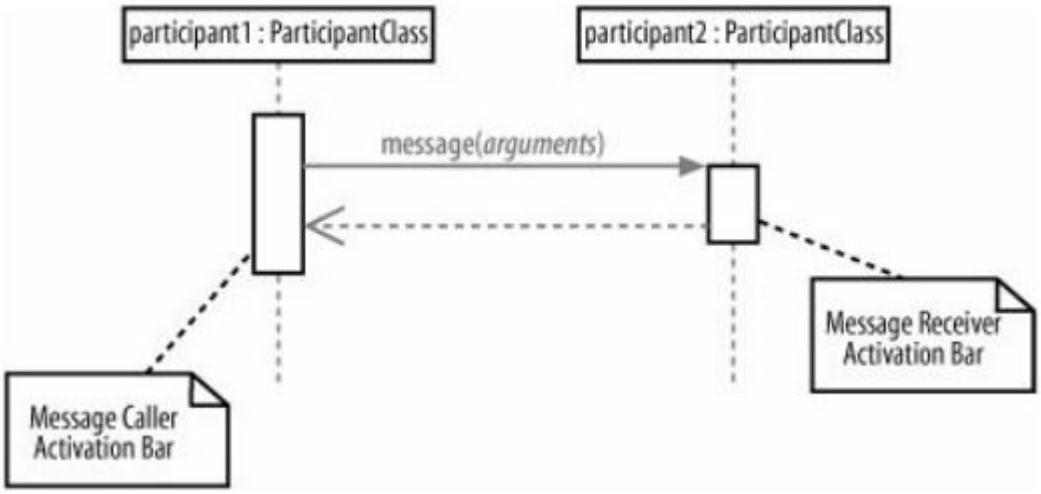
2.1.3.5 เหตุการณ์ (Event) สัญญาณ (Signal) และข้อความ (Message) เมื่อเกิดการปฏิสัมพันธ์ใด ๆ ขึ้นในแผนภาพลำดับ ส่วนที่แสดงการเกิดปฏิสัมพันธ์นั้นเรียกว่าเหตุการณ์ สิ่งที่ถูกแลกเปลี่ยนหรือไปกระตุ้นในช่วงของการเกิดปฏิสัมพันธ์เรียกว่าข้อความหรือสัญญาณตามลำดับ ซึ่งเหตุการณ์หนึ่ง ๆ จะประกอบด้วยข้อความ ผู้ส่งสัญญาณ และผู้รับสัญญาณ ดังตัวอย่างในรูปที่ 2.11



รูปที่ 2.11 เหตุการณ์ สัญญาณ และข้อความของแผนภาพลำดับอย่างง่าย[2]

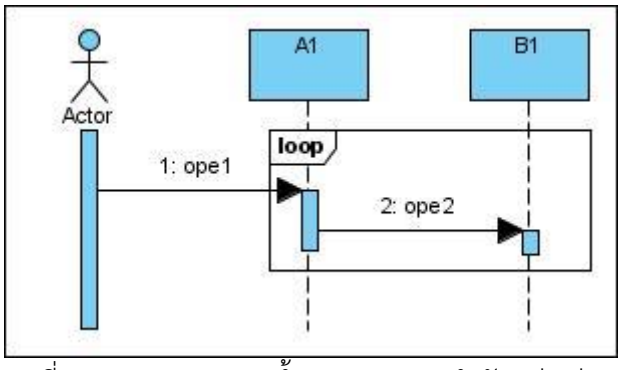
2.1.3.6 แถบแอคทิเวชัน (Activation Bar) แสดงช่วงที่แอคเตอร์หรือส่วนประกอบของระบบทำงานอยู่ (Active) ซึ่งสภาวะการทำงานนี้อาจเป็นช่วงเวลาที่รอ (Waiting) ให้ส่วนประกอบอีกส่วนหนึ่งทำงานแล้วส่งข้อความกลับมาก็ได้ แถบแอคทิเวชันวาดเป็นแท่ง

สี่เหลี่ยมผืนผ้ายาว ๆ ทาบลงบนเส้นชีวิต ใช้แสดงว่า ในขณะที่ทำงานอยู่ จะไม่สามารถทำงานอื่นได้ ต้องรอให้ทำงานเสร็จก่อนเท่านั้น ดังตัวอย่างในรูปที่ 2.12



รูปที่ 2.12 แถบแอคทิเวชันของแผนภาพลำดับอย่างง่าย[2]

2.1.3.7 เฟรม (Frame) มีลักษณะเป็นกรอบสี่เหลี่ยมผืนผ้า ใช้อธิบายบางส่วนของระบบว่าเป็นส่วนประกอบที่ทำงานร่วมกัน อาจใช้อธิบายการวนซ้ำ (Loop) เงื่อนไข (Condition) หรือละรายละเอียดเพื่อแยกไปอธิบายในแผนภาพย่อยอื่นเป็นต้น ดังตัวอย่างการวนซ้ำในรูปที่ 2.13



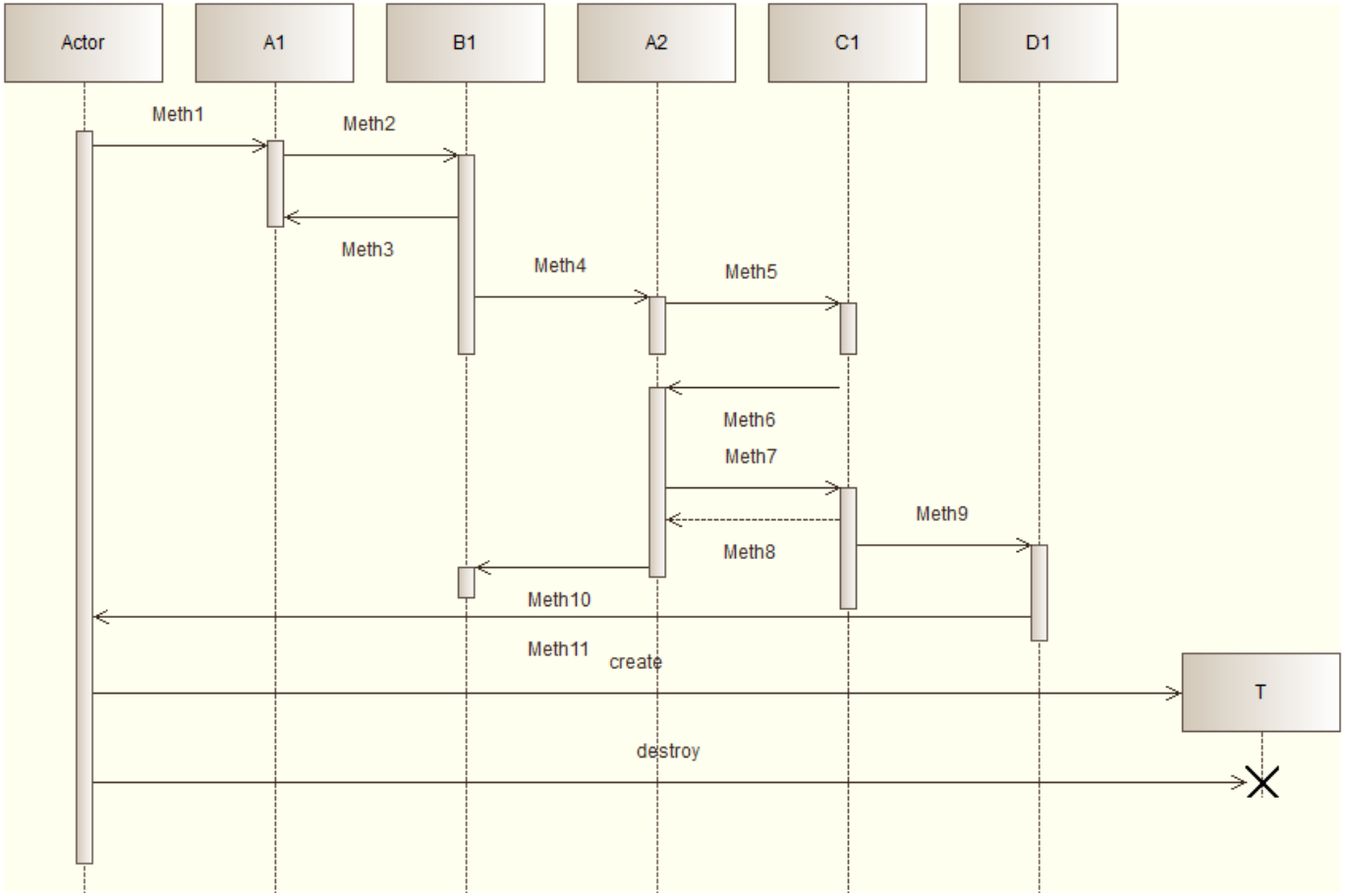
รูปที่ 2.13 เฟรมแบบวนซ้ำของแผนภาพลำดับอย่างง่าย

2.1.4 มาตรฐานเอ็กซ์เอ็มไอ (XMI: XML Metadata Interchange) [3, 11]

Object Management Group (OMG) หน่วยงานที่ออกแบบแผนภาพยูเอ็มแอลได้ออกแบบมาตรฐานเอ็กซ์เอ็มไอ (XMI) ซึ่งขยายเพิ่มเติมจากมาตรฐานเอ็กเอ็มแอล (XML: Extensible Markup Language) เพื่อใช้เป็นภาษามาตรฐานในการแลกเปลี่ยนเมตาดาตา ต่อมาก็นิยมนำไปใช้

เป็นมาตรฐานแลกเปลี่ยนแบบจำลองที่พัฒนาโดยใช้มาตรฐานยูเอ็มแอล และได้รับการตีพิมพ์เป็นมาตรฐานของ ISO/IEC 19503:2005 Information technology – XML Metadata Interchange (XMI) ปัจจุบันมาตรฐาน XMI พัฒนาจนถึงรุ่น 2.4.1 ซึ่งใช้ได้กับมาตรฐานยูเอ็มแอลรุ่น 2.0 เป็นต้นไป [12]

ภาษาเอ็กซ์เอ็มไอถูกนำมาใช้แพร่หลายในโปรแกรมสร้างแบบจำลองต่าง ๆ ทำให้สามารถแลกเปลี่ยนกันระหว่างโปรแกรมได้ เช่น แผนภาพลำดับจากโปรแกรม Modelio ดังรูปที่ 2.14 เมื่อส่งออก (Export) แล้วจะได้โค้ดเอ็กซ์เอ็มไอ ดังตัวอย่างจากโอเอ็มจีแสดงส่วนหนึ่งดังรูปที่ 2.15



รูปที่ 2.14 แผนภาพลำดับที่สร้างจากโปรแกรม Modelio

```
<?xml version="1.0"?>
<xmi:XMI xmlns:mofext="http://www.omg.org/spec/MOF/20110701" xmlns:uml="http:/
  <uml:Package xmi:type="uml:Package" xmi:id="_XMI" name="XMI" URI="http://www
    <packagedElement xmi:type="uml:Class" name="Delete" xmi:id="_XMI-Delete">
      <generalization xmi:type="uml:Generalization" xmi:id="_XMI-Delete-Genera
        <general xmi:idref="_XMI-Difference"/>
      </generalization>
    </packagedElement>
    <packagedElement xmi:type="uml:Association" name="A_addition_add" xmi:id="
      <memberEnd xmi:idref="_XMI-A_addition_add-add"/>
      <memberEnd xmi:idref="_XMI-Add-addition"/>
      <ownedEnd xmi:type="uml:Property" name="add" visibility="public" xmi:id=
        <type xmi:idref="_XMI-Add"/>
        <association xmi:idref="_XMI-A_addition_add"/>
        <upperValue xmi:type="uml:LiteralUnlimitedNatural" value="*" xmi:id="_
        <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_XMI-A_addition_add-
      </ownedEnd>
    </packagedElement>
    <packagedElement xmi:type="uml:Class" name="Documentation" xmi:id="_XMI-Do
      <ownedAttribute xmi:type="uml:Property" name="contact" visibility="publi
        <type href="http://www.omg.org/spec/UML/20110701/PrimitiveTypes.xmi#St
        <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_XMI-Documentation-c
      </ownedAttribute>
      <ownedAttribute xmi:type="uml:Property" name="exporter" visibility="publ
        <type href="http://www.omg.org/spec/UML/20110701/PrimitiveTypes.xmi#St
        <lowerValue xmi:type="uml:LiteralInteger" xmi:id="_XMI-Documentation-e
      </ownedAttribute>
```

รูปที่ 2.15 ส่วนหนึ่งของโค้ดเอ็กซ์เอ็มไอ [11]

2.1.5 ความครบถ้วนสมบูรณ์ (Completeness)

มาตรฐานIEEE 830: IEEE Recommended Practice for Software Requirements Specifications [13] ได้กล่าวไว้ว่า “รายการความต้องการซอฟต์แวร์ (SRS: Software Requirement Specification) ที่ดีจะต้องประกอบด้วยคุณลักษณะ 8 ข้อ” ซึ่งใน 8 ข้อนั้นมีความครบถ้วนสมบูรณ์และความสอดคล้องอยู่ด้วย โดยให้คำนิยามสำหรับความครบถ้วนสมบูรณ์ไว้ว่า

“รายการความต้องการซอฟต์แวร์จะมีความครบถ้วนสมบูรณ์เมื่อประกอบด้วย

- ความต้องการหลัก ๆ ครบ ไม่ว่าจะด้านฟังก์ชันการทำงาน ด้านประสิทธภาพ ด้านการออกแบบ ด้านคุณลักษณะ หรือด้านส่วนต่อประสานภายนอก
- คำจำกัดความของการตอบสนองของซอฟต์แวร์ทั้งหมด
- ชื่อและเอกสารอ้างอิงทั้งหมดของรูปภาพ ตาราง และแผนภาพ”

จะเห็นว่าความครบถ้วนสมบูรณ์จะเน้นไปที่ส่วนประกอบหลัก ๆ ที่สำคัญ หรือที่ขาดไม่ได้ ซึ่งงานวิจัยนี้ต้องการตรวจสอบความครบถ้วนสมบูรณ์เชิงโครงสร้างของการออกแบบชั้นนามธรรมที่

ต้องการ (structural completeness of abstraction layer design) เพื่อตรวจสอบความพร้อมของข้อมูลเท่านั้น ดังนั้นงานวิจัยนี้จึงให้คำจำกัดความของความครบถ้วนสมบูรณ์ของแผนภาพว่า

“แผนภาพคลาสที่มีความครบถ้วนสมบูรณ์จะต้องประกอบด้วย

• ชื่อของระบบ ชื่อคลาส และชื่อความสัมพันธ์

• ส่วนประกอบที่จำเป็นอื่น ๆ ของแผนภาพ ได้แก่คุณลักษณะ เมธอด ประเภทการเข้าถึง ชนิดของตัวแปร

แผนภาพลำดับที่มีความครบถ้วนสมบูรณ์จะต้องประกอบด้วย

• ชื่อของแผนภาพ ชื่อส่วนประกอบต่าง ๆ ของแผนภาพ และชื่อข้อความที่ส่งระหว่างส่วนประกอบต่าง ๆ ของแผนภาพ

• ส่วนประกอบที่จำเป็นอื่น ๆ ของแผนภาพ ได้แก่ ชนิดของตัวแปรและการคืนค่าบนข้อความ”

2.1.6 ความสอดคล้อง (Consistency)

มาตรฐานIEEE 830: IEEE Recommended Practice for Software Requirements Specifications [13] กล่าวถึงความสอดคล้องของรายการความต้องการซอฟต์แวร์ไว้ว่า

“ความสอดคล้องหมายถึงความสอดคล้องภายใน (internal consistency) ถ้ารายการความต้องการซอฟต์แวร์ขัดแย้งกับเอกสารระดับสูงกว่าอันใดอันหนึ่ง ดังนั้นรายการความต้องการนั้นไม่มีความสอดคล้อง”

นิยามนี้ใช้การเปรียบเทียบกับเอกสารระดับสูงกว่า ซึ่งถือว่าเป็นเอกสารที่สำคัญกว่า ถูกต้องกว่า เอกสารระดับต่ำกว่าจะขัดแย้งไม่ได้ หากมีการขัดแย้งกัน จะถือว่าเอกสารระดับต่ำกว่านั้นผิด

อย่างไรก็ตาม มาตรฐาน IEEE 830 ไม่ได้กล่าวถึงความสอดคล้องของแผนภาพยูเอ็มแอล แต่งานของ Wang, H., Feng, T., Zhang, J., และ Zhang, K. [14] กล่าวถึงความไม่สอดคล้องแทนความสอดคล้องของแผนภาพยูเอ็มแอลไว้ว่า

“ความไม่สอดคล้องระหว่างแผนภาพหมายถึงความไม่เข้ากันระหว่างความต้องการซอฟต์แวร์และการออกแบบแผนภาพ”

นิยามนี้คล้ายกับนิยามของ IEEE 830 คือการใช้ความต้องการซอฟต์แวร์ซึ่งเป็นเอกสารระดับสูงกว่าเป็นตัวเปรียบเทียบ แต่งานของ Dubauskaite, R. และ Vasilecas, O. [15] กล่าวถึงความสอดคล้องของแผนภาพยูเอ็มแอลไว้ว่า

“ความสอดคล้องหมายถึงโครงสร้างต่าง ๆ และส่วนประกอบที่ปรากฏในแผนภาพหนึ่งเข้ากันได้กับที่ปรากฏอยู่ในแผนภาพอื่น ๆ”

Simmonds, J., Van Der Straeten, R., Jonckers, V., และ Mens, T. [16] ก็กล่าวถึงความไม่สอดคล้องแทนความสอดคล้องของแผนภาพไว้ว่า

“สภาพที่แผนภาพยูเอ็มแอลสองแผนภาพใด ๆ แสดงให้เห็นว่ามีการขัดแย้งกัน หมายความว่ามันไม่สอดคล้องกัน”

สำหรับงานวิจัยนี้ไม่ได้ครอบคลุมการตรวจสอบไปถึงความต้องการซอฟต์แวร์ด้วย แต่เป็นการตรวจสอบระหว่างแผนภาพลำดับและแผนภาพคลาสที่มีอยู่เท่านั้น ดังนั้นจึงให้คำจำกัดความของความสอดคล้องของแผนภาพว่า

“แผนภาพลำดับใด ๆ ที่ไม่มีความขัดแย้งกับแผนภาพคลาส ทั้งในด้านชื่อ ความสัมพันธ์ พารามิเตอร์ และข้อห้ามต่าง ๆ เป็นแผนภาพที่มีความสอดคล้องกัน”

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 กฎพื้นฐานการสร้างแผนภาพยูเอ็มแอลให้ถูกต้อง (Basic Rules to Build Correct UML Diagrams) [17] โดย Alanazi, M. N. ปี 2009

งานวิจัยนี้นำเสนอกฎพื้นฐานของการสร้างแผนภาพ 9 ชนิดของยูเอ็มแอล ได้แก่ แผนภาพคลาส แผนภาพวัตถุ แผนภาพลำดับ แผนภาพคอลาบอเรชัน แผนภาพยูสเคส แผนภาพสถานะ แผนภาพกิจกรรม แผนภาพดีพลอยเมนต์ และแผนภาพคอมโพเนนต์ เพราะแผนภาพทั้ง 9 วาดขึ้นมาเพื่ออธิบายสิ่งเดียวกันแต่มุมมองต่างกัน ดังนั้นแผนภาพแบบต่าง ๆ จึงมีความสัมพันธ์กันไม่ทางตรงก็ทางอ้อม กฎที่งานวิจัยนี้นำเสนอมี 15 ข้อ บางข้อเป็นกฎสำหรับแผนภาพใดแผนภาพหนึ่ง บางข้อเป็นกฎที่เกี่ยวกับความสัมพันธ์ระหว่างแผนภาพ ซึ่งกฎที่เกี่ยวข้องกับแผนภาพคลาสอย่างเดียวคือ กฎข้อที่ 5 กฎที่เกี่ยวข้องกับแผนภาพลำดับอย่างเดียวคือกฎข้อที่ 13 และกฎข้อที่เกี่ยวกับความสัมพันธ์ระหว่างแผนภาพลำดับกับแผนภาพคลาสคือกฎข้อที่ 9, 10, 11, 12, 14 และ 15 ซึ่งกฎทั้ง 15 ข้อมีดังนี้

ข้อ 1 ชื่อของยูสเคสต้องเป็นคำกริยา

ข้อ 2 ยูสเคสต้องสอดคล้องกับแผนภาพปฏิสัมพันธ์

ข้อ 3 ชื่อของคลาสต้องเป็นคำนาม

ข้อ 4 คุณลักษณะ (Attribute) ต้องเป็นส่วนตัว (Private)

ข้อ 5 คลาสทุกคลาสต้องมีเมธอด (Method)

ข้อ 6 คลาสที่เป็นคลาสปลาย (Leaf Class) จะต้องไม่เป็นแอ็บสแตร็คคลาส (Abstract Class)

ข้อ 7 การขึ้นต่อกันในแผนภาพคลาสจะต้องสอดคล้องกับในแผนภาพวัตถุ

ข้อ 8 ต้องไม่มีอินสแตนซ์ (Instance) จากคลาสแบบแอ็บสแตร็ค

ข้อ 9 ข้อความของแผนภาพลำดับต้องสอดคล้องกับเมธอดของแผนภาพคลาส

ข้อ 10 ข้อความต้องส่งผ่านระหว่างคลาสที่เกี่ยวข้องกันเท่านั้น

ข้อ 11 ส่วนของระบบของแผนภาพลำดับต้องมาจากคลาสของแผนภาพวัตถุ

ข้อ 12 เมธอดสาธารณะ (Public Method) ในแผนภาพคลาสต้องถูกเรียกเป็นข้อความในแผนภาพลำดับครบทุกเมธอด

ข้อ 13 ส่วนของระบบต้องมีชื่อ

ข้อ 14 คลาสทุกคลาสในแผนภาพคลาสต้องปรากฏเป็นส่วนของระบบในแผนภาพลำดับ

ข้อ 15 ส่วนของระบบในแผนภาพลำดับต้องสอดคล้องกับความหลากหลาย (Multiplicity) ในแผนภาพคลาส

งานวิจัยนี้ชี้ให้เห็นว่าแผนภาพคลาสมีส่วนสำคัญที่ส่งผลต่อแผนภาพลำดับอย่างมาก ผู้วิจัยจึงนำแผนภาพคลาสเข้ามาใช้ช่วยตรวจสอบความสอดคล้องของแผนภาพลำดับ และในขณะเดียวกันก็ได้นำกฎข้อที่ 5 และ 13 ไปตรวจสอบความพร้อมของข้อมูลก่อนการตรวจสอบความสอดคล้องอีกด้วย

2.2.2 การวิเคราะห์ผลกระทบและการจัดการการเปลี่ยนแปลงของโมเดลยูเอ็มแอล (Impact Analysis and Change Management of UML Models) [18] โดย Briand, L.C. และ Labiche, Y. ปี 2003

งานวิจัยนี้ตรวจสอบความสอดคล้องกันระหว่างแผนภาพต่าง ๆ แล้วจึงนำแผนภาพของระบบต่างรุ่นกันมาวิเคราะห์ต่อถึงผลกระทบและความเปลี่ยนแปลงที่เกิดขึ้นโดยใช้กฎที่เขียนด้วย OCL (Object Constraint Language) ซึ่งผู้วิจัยสนใจกฎที่ใช้ตรวจสอบความสอดคล้องของแผนภาพ 75 ข้อ แต่เกี่ยวข้องกับแผนภาพลำดับและแผนภาพคลาส 5 ข้อได้แก่

ข้อ 1 คลาสใด ๆ ที่ไม่ใช่คลาสตัวเองหรือคลาสลูกไม่สามารถเรียกเมธอดแบบป้องกันได้

ข้อ 2 คลาสใด ๆ ที่ไม่ใช่คลาสตัวเองไม่สามารถเรียกเมธอดแบบส่วนตัวได้

ข้อ 3 ส่วนประกอบของระบบในแผนภาพลำดับจะต้องสอดคล้องกับคลาสในแผนภาพคลาส

ข้อ 4 แต่ละข้อความในแผนภาพลำดับ จะต้องมีความสัมพันธ์อยู่จริง ในแผนภาพอื่น ๆ

ข้อ 5 ชื่อของข้อความในแผนภาพลำดับต้องไม่ซ้ำกัน

งานวิจัยนี้มีกฎที่ตรวจสอบการเข้าถึงที่งานวิจัยก่อนหน้านี้ไม่ได้กล่าวถึงด้วย ผู้วิจัยจึงนำการเข้าถึงมาร่วมพิจารณาความสอดคล้องด้วย

2.2.3 การออกแบบและพัฒนาเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาส แผนภาพซีเควนซ์ และแผนภาพสเตทชาร์ท (Design and Development of a Tool for Consistency Checking among Class Diagrams,Sequence Diagrams, and Statechart Diagrams) [19] โดย กนิษฐา บุญคุ้ม ปี 2007

งานวิจัยนี้สร้างเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพสามแผนภาพ โดยตรวจสอบทีละคู่ แต่ละคู่จะมีกฎของการตรวจสอบของตัวเอง ซึ่งผู้วิจัยจะสนใจเฉพาะส่วนกฎการตรวจสอบแผนภาพลำดับและแผนภาพคลาส 10 ข้อ ดังนี้

ข้อ 1 ทุก ๆ ส่วนประกอบของระบบในแผนภาพลำดับจะต้องปรากฏเป็นคลาสหนึ่งๆ ในแผนภาพคลาส

ข้อ 2 ถ้าปรากฏข้อความจากส่วนประกอบของระบบใด ไปยังส่วนประกอบของระบบอื่นในแผนภาพลำดับ จะต้องมีความสัมพันธ์ระหว่างคลาสคู่นั้นในแผนภาพคลาส

ก) คลาส 2 คลาสมีความสัมพันธ์กันแบบแอกกรีเกชันในแผนภาพคลาส โดยมีคลาสหนึ่งเป็นคลาสหลัก และอีกคลาสหนึ่งเป็นคลาสย่อย ถ้ามีการส่งข้อความแบบสร้างส่วนของระบบในแผนภาพลำดับ คลาสย่อยไม่สามารถส่งข้อความแบบสร้างไปสร้างคลาสหลักได้

ข) คลาส 2 คลาสมีความสัมพันธ์กันแบบคอมโพซิชันชันในแผนภาพคลาส โดยมีคลาสหนึ่งเป็นคลาสหลัก และอีกคลาสหนึ่งเป็นคลาสย่อย ถ้ามีการส่งข้อความแบบสร้างส่วนของระบบในแผนภาพลำดับ คลาสย่อยไม่สามารถส่งข้อความแบบสร้างไปสร้างคลาสหลักได้

ค) คลาส 2 คลาสมีความสัมพันธ์กันแบบคอมโพซิชันชันในแผนภาพคลาส โดยมีคลาสหนึ่งเป็นคลาสหลัก และอีกคลาสหนึ่งเป็นคลาสย่อย ถ้ามีการส่งข้อความแบบทำลายส่วนของระบบในแผนภาพลำดับ คลาสย่อยไม่สามารถส่งข้อความแบบทำลายไปทำลายคลาสหลักได้

ง) คลาส 2 คลาสมีความสัมพันธ์กันแบบดีเพนเดนซีในแผนภาพคลาส ถ้ามีการส่งข้อความแบบทำลายส่วนของระบบในแผนภาพลำดับซึ่งเป็นคลาสผู้ให้บริการแล้ว หลังจากนั้นจะไม่สามารถเรียกใช้คลาสผู้ให้บริการผ่านคลาสผู้ใช้บริการได้

ข้อ 3 ถ้าส่วนของระบบ A ไปร้องขอส่วนของระบบ B ให้ทำเมธอด X โดยการส่งข้อความในแผนภาพลำดับแล้ว คลาสของส่วนของระบบ B ในแผนภาพคลาสจะต้องมีเมธอด X ปรากฏอยู่เสมอ

ข้อ 4 จำนวน ลำดับ และชนิดของพารามิเตอร์ของเมธอดที่ปรากฏบนข้อความในแผนภาพลำดับ จะมีจำนวน ลำดับ และชนิดของพารามิเตอร์ของเมธอด เหมือนกับเมธอดที่ประกาศไว้ในคลาสของแผนภาพคลาส

ข้อ 5 ตัวแปร และชนิดของตัวแปรที่เป็นตัวกำหนดเงื่อนไขในเครื่องหมาย '[]' บนข้อความในแผนภาพลำดับ จะประกาศเป็นคุณลักษณะในคลาส

ข้อ 6 เมธอดบนข้อความที่ส่งระหว่างส่วนของระบบในแผนภาพลำดับ โดยให้ส่วนของระบบที่เรียกส่วนของระบบอื่นเป็นส่วนของระบบหลัก และส่วนของระบบที่ถูกเรียกเป็นส่วนของระบบย่อยแล้ว ถ้ามีแผนภาพลำดับอื่นที่มีส่วนของระบบตัวเดียวกันกับแผนภาพลำดับแรก ส่วนของระบบย่อยไม่สามารถเรียกส่วนของระบบหลักได้

งานวิจัยนี้มีกฎค่อนข้างจะครอบคลุม ผู้วิจัยจึงยึดเป็นหลักในการวิจัย และได้แก้ปัญหาที่พบในงานวิจัยนี้คือ หากมีข้อความคลาสใดไม่ได้ระชื่อหรือเมธอด จะทำให้ตรวจสอบความสอดคล้องไม่ได้ ผู้วิจัยจึงตรวจสอบความพร้อมของข้อมูลเสียก่อนเพื่อแก้ปัญหานี้ อีกทั้งเพิ่มการตรวจสอบเรื่องทิศทางความสัมพันธ์และการเข้าถึงซึ่งงานวิจัยนี้ไม่ได้กล่าวถึง

บทที่ 3

การออกแบบกฎการตรวจสอบความพร้อมของข้อมูลและความสอดคล้อง

ในบทนี้จะกล่าวถึงการออกแบบกฎการตรวจสอบความพร้อมของข้อมูลและความสอดคล้องของแผนภาพคลาสและแผนภาพลำดับ และแสดงกฎการตรวจสอบความพร้อมของข้อมูลและความสอดคล้องทั้งหมดด้วย

3.1 การตรวจสอบความพร้อมของข้อมูล

เพื่อที่จะตรวจสอบความสอดคล้องระหว่างแผนภาพได้ครบถ้วนตามขอบเขตที่ออกแบบไว้ ข้อมูลบางอย่างเป็นสิ่งจำเป็นที่จะต้องมี ในขณะที่แผนภาพยูเอ็มแอลออกแบบให้มีลักษณะยืดหยุ่นมากในแง่ของรายละเอียด กล่าวคือแผนภาพยูเอ็มแอลเพียงกำหนดไว้ว่ารายละเอียดของโครงสร้างของแผนภาพต่าง ๆ มีอะไรได้บ้าง แต่ไม่บังคับให้มีรายละเอียดเหล่านั้นให้ครบถ้วน เพียงกำหนดไว้เป็นเกณฑ์ขั้นต่ำเท่านั้น การตรวจสอบความพร้อมของข้อมูลของงานวิจัยนี้ นอกจากจะเป็นการตรวจสอบระดับข้อมูลตามความต้องการขั้นต่ำแล้ว ยังเป็นการบังคับให้แผนภาพมีระดับความละเอียดถึงระดับที่ต้องการ มีข้อมูลเพียงพอต่อการนำไปตรวจสอบความสอดคล้องต่อไปได้

การตรวจสอบความพร้อมของข้อมูลของงานวิจัยนี้ จะตรวจสอบส่วนประกอบต่าง ๆ และความสัมพันธ์แผนภาพทั้งสองดังสรุปได้ดังนี้

แผนภาพคลาส

- แผนภาพคลาสจะต้องมีชื่อของระบบที่อธิบายอยู่
- คลาสทุกคลาสที่ปรากฏในแผนภาพต้องมีชื่อคลาส
- คลาสทุกคลาสที่ปรากฏในแผนภาพต้องมีเมธอดและการเข้าถึงของเมธอดระบุไว้
- คลาสทุกคลาสต้องมีความสัมพันธ์กับคลาสอื่น

แผนภาพลำดับ

- แผนภาพลำดับทุกแผนภาพจะต้องมีชื่อแผนภาพ
- แอคเตอร์และส่วนประกอบของระบบที่ปรากฏในแผนภาพต้องมีชื่อ
- ข้อความที่ส่งหากันทุกข้อความจะต้องมีชื่อเมธอด
- ส่วนประกอบของระบบต้องมีความสัมพันธ์กับส่วนอื่น

3.2 การตรวจสอบความสอดคล้อง

งานวิจัยนี้จะตรวจสอบความสอดคล้องของแผนภาพทั้งสองโดยพิจารณาให้ครอบคลุมสิ่งต่อไปนี้

- ชื่อของส่วนประกอบต่าง ๆ ได้แก่ ชื่อคลาส ชื่อคุณลักษณะ ชื่อเมธอด ชื่อพารามิเตอร์ และชื่อข้อความ เพื่อใช้
- ชนิดของส่วนต่าง ๆ ของระบบ ได้แก่ พารามิเตอร์ คุณลักษณะ เงื่อนไข และการคืนค่าของเมธอด
- ลำดับในการส่งข้อความ ทั้งนี้เพื่อใช้ประกอบการตรวจสอบการเรียกใช้พารามิเตอร์และการคืนค่าต่าง ๆ อีกทั้งยังใช้ตรวจสอบการเข้าถึงเมธอดบางอย่างที่อาจถูกทำลายไปแล้ว
- ความสัมพันธ์ระหว่างคลาสต่าง ๆ โดยพิจารณาทั้งทิศทางของความสัมพันธ์และประเภทของความสัมพันธ์ทั้ง 5 ประเภทด้วย ได้แก่ แอสโซซิเอชัน แอกกรีเกชัน คอมโพซิชัน ดีเพนเดนซี และเจนเนอรอลไลเซชัน เพื่อตรวจสอบใช้งานคุณสมบัติบางอย่างที่ถ่ายทอดจากคลาสแม่สู่คลาสลูกหรือคลาสผู้ให้บริการสู่คลาสผู้ใช้บริการ
- การเข้าถึงของเมธอด เพื่อใช้พิจารณาประกอบกับความสัมพันธ์กรณีที่มีการถ่ายทอดคุณสมบัติจากคลาสแม่สู่คลาสลูก หรือคลาสผู้ให้บริการสู่คลาสผู้ใช้บริการ

เมื่อพิจารณาข้อบังคับและแนวทางการสร้างแผนภาพคลาสและแผนภาพลำดับร่วมกับงานวิจัยทั้งสามที่เกี่ยวข้อง จะสรุปออกมาเป็นเรื่องและกฎได้ดังต่อไปนี้

3.2.1 แผนภาพคลาสใช้อธิบายโครงสร้างทั้งหมดของระบบ ในขณะที่แผนภาพลำดับ ใช้อธิบายเหตุการณ์ต่าง ๆ โดยใช้ส่วนประกอบทั้งหมดหรือเพียงบางส่วนของระบบเป็นส่วนประกอบ ดังนั้นแอคเตอร์และส่วนประกอบของระบบจึงต้องอ้างอิงจากคลาสในแผนภาพคลาส ดังสรุปเป็นกฎได้ดังนี้

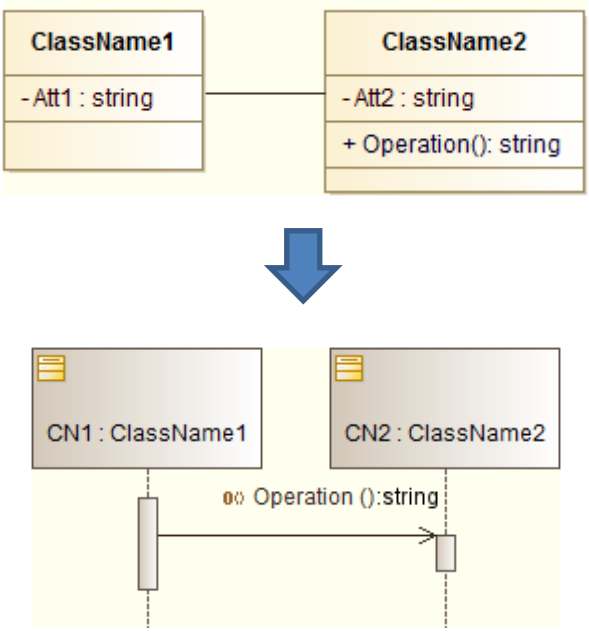
กฎข้อที่ 1 แอคเตอร์และส่วนของระบบในแผนภาพลำดับจะต้องปรากฏเป็นคลาสในแผนภาพคลาส

จากกฎข้อที่ 1 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) นำชื่อแอคเตอร์และส่วนประกอบของระบบในแผนภาพลำดับทุกแผนภาพ มาตรวจสอบกับชื่อคลาสทั้งหมดในแผนภาพคลาส

2) ถ้ามีชื่อใดไม่ตรงกับชื่อคลาสในแผนภาพคลาสเลย แอคเตอร์หรือส่วนประกอบของระบบนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.2 แผนภาพคลาสหนึ่ง ๆ อธิบายโครงสร้างของระบบหนึ่งระบบ แต่ระบบหนึ่งระบบ อาจทำงานได้หลายอย่าง จึงแสดงเป็นแผนภาพลำดับได้หลายเหตุการณ์ ซึ่งถ้าอธิบายเหตุการณ์ทั้งหมดของระบบได้ครบถ้วน โครงสร้างต่าง ๆ ของระบบก็ต้องถูกนำไปใช้ในแผนภาพลำดับให้ครบด้วย โครงสร้างดังกล่าวคือคลาสที่จะถูกนำไปเป็นส่วนประกอบของระบบในแผนภาพ และเมธอดก็จะถูกเรียกใช้ในฐานะข้อความในแผนภาพลำดับดังรูปที่ 3.1 ยกเว้นคลาสแม่ที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชันและดีเพนเดนซีที่ทำหน้าที่เป็นแม่แบบเท่านั้นที่ไม่ต้องถูกเรียกใช้ในแผนภาพคลาส



รูปที่ 3.1 คลาสและเมธอดที่ถูกเรียกใช้เป็นส่วนประกอบของระบบและข้อความในแผนภาพลำดับ

จากรูปที่ 3.1 แผนภาพคลาสมีคลาสอยู่ 2 คลาสคือ ClassName1 และ ClassName2 ซึ่งปรากฏเป็นส่วนประกอบของระบบในแผนภาพลำดับในชื่อ CN1 และ CN2 นอกจากนี้เมธอด Operation ของคลาส ClassName2 ยังถูก CN1 เรียกใช้เป็นข้อความในแผนภาพลำดับด้วย

นอกจากนี้การตรวจสอบเมธอดจะต้องตรวจสอบ overloading method ซึ่งมีชื่อเหมือนกันแต่ข้อกำหนดด้านพารามิเตอร์ต่างกันด้วย ดังนั้นจึงสรุปกฎได้ 2 ข้อดังนี้

กฎข้อที่ 2 เมธอดทุกเมธอดในแผนภาพคลาสจะต้องถูกเรียกใช้ในแผนภาพลำดับ

จากกฎข้อที่ 2 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) นำชื่อข้อความทุกข้อความในแผนภาพลำดับทุกแผนภาพ มาตรวจสอบกับชื่อเมธอดทั้งหมดในแผนภาพคลาส โดยตรวจสอบข้อความเฉพาะกับเมธอดในคลาสที่ข้อความระบุไว้ ไม่ตรวจสอบข้ามคลาส

2) หากในคลาสที่ตรวจสอบมี overloading method ก็ต้องตรวจสอบ overloading method ทั้งหมดด้วยว่าถูกใช้ครบหรือไม่

3) ถ้ามีชื่อใดตรงกับชื่อเมธอดในคลาสที่ระบุแผนภาพคลาสก็จะสะสมคะแนนไว้

4) เมื่อตรวจสอบครบหมดแล้ว ถ้ามีเมธอดใดที่ไม่ได้คะแนนสะสมเลยอย่างน้อย 1 เมธอด แผนภาพลำดับเหล่านั้นไม่สอดคล้องกับแผนภาพคลาส

กฎข้อที่ 3 คลาสทุกคลาสในแผนภาพคลาสจะต้องปรากฏเป็นแอ็คเตอร์หรือส่วนของระบบในแผนภาพลำดับ

จากกฎข้อที่ 3 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

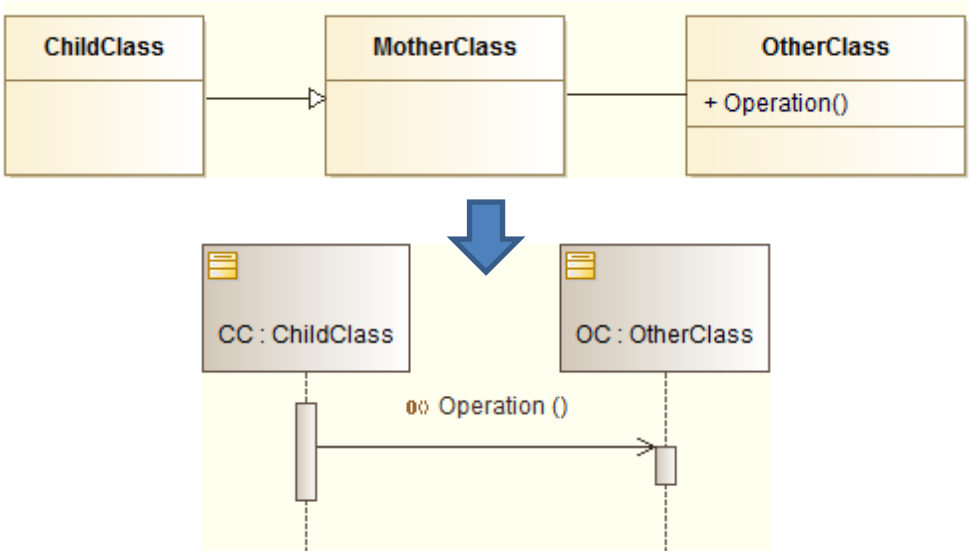
1) นำชื่อแอคเตอร์หรือส่วนประกอบของระบบทุกส่วนในแผนภาพลำดับทุกแผนภาพ มาตรวจสอบกับชื่อคลาสทั้งหมดในแผนภาพคลาส

2) ถ้ามีชื่อใดตรงกับชื่อคลาสในแผนภาพคลาสก็จะสะสมคะแนนไว้

3) ถ้ามีคลาสใดเป็นคลาสแม่ในความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือดีเพนเดนซีในแผนภาพคลาสก็ให้สะสมคะแนนไว้

4) เมื่อตรวจสอบครบหมดแล้ว ถ้ามีคลาสใดที่ไม่ได้คะแนนสะสมเลยอย่างน้อย 1 คลาส แผนภาพลำดับเหล่านั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.3 ในแผนภาพลำดับที่อธิบายเหตุการณ์ต่าง ๆ ที่เกิดขึ้นในระบบ ส่วนต่าง ๆ ของระบบส่งข้อความหากันแสดงถึงการเรียกใช้เมธอดของคลาสต่าง ๆ ซึ่งคลาสต่าง ๆ ดังกล่าวต้องมีความสัมพันธ์กันอยู่จริงดังรูปที่ 3.2



รูปที่ 3.2 การถ่ายทอดความสัมพันธ์จากคลาสแม่สู่คลาสลูก

จากรูปที่ 3.2 ChildClass เป็นคลาสลูกของ MotherClass จึงได้รับการถ่ายทอดความสัมพันธ์กับ OtherClass มาด้วย ดังนั้นในแผนภาพลำดับ ChildClass จึงส่งข้อความเรียกใช้เมธอด Operation ของ OtherClass ได้ ซึ่งจะสรุปกฎได้ดังนี้

กฎข้อที่ 4 ความสัมพันธ์ของแอคเตอร์และส่วนของระบบในแผนภาพลำดับจะต้องมีอยู่จริงในแผนภาพคลาส

โดยกฎข้อนี้พิจารณาการถ่ายทอดความสัมพันธ์แบบเจนเนอรอลไลเซชันร่วมด้วย กล่าวคือ คลาสลูกใดที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชันกับคลาสแม่ จะได้รับการถ่ายทอดความสัมพันธ์ของคลาสแม่ที่มีต่อคลาสอื่น ๆ มาด้วย เรียกความสัมพันธ์แบบนี้ว่าความสัมพันธ์โดยปริยาย

จากกฎข้อที่ 4 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) เก็บรวบรวมคู่การส่งข้อความในแผนภาพลำดับทั้งหมดเป็นคู่ ๆ

2) เก็บรวบรวมความสัมพันธ์จากแผนภาพคลาสเป็นคู่ ๆ

3) นำคู่การส่งข้อความมาตรวจสอบกับคู่ความสัมพันธ์จากคลาสมาตรวจสอบ

4) ถ้าพบว่าสอดคล้องกันคู่การส่งข้อความก็จะสะสมคะแนนไว้

5) ถ้าไม่สอดคล้องกันให้ตรวจสอบว่าคู่การส่งข้อความเป็นคลาสใดคลาสหนึ่งในคลาสลูกของความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือไม่

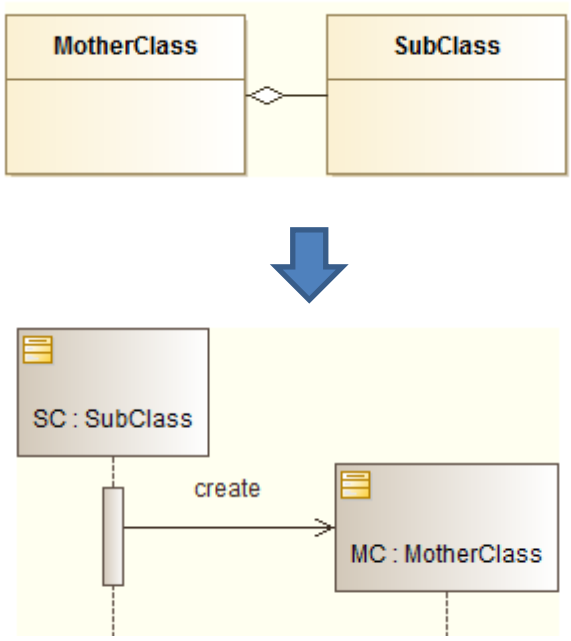
6) ถ้าเป็น ให้เป็นคลาสลูกเป็นคลาสแม่แทน แล้วตรวจสอบกับคู่ความสัมพันธ์จากคลาสอีกครั้ง

7) ถ้าพบว่าสอดคล้องกันคู่การส่งข้อความก็จะสะสมคะแนนไว้

8) เมื่อตรวจสอบจนครบหมดแล้ว ถ้ามีคู่การส่งข้อความของแผนภาพลำดับใดไม่มีคะแนนสะสมเลย แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.4 เมื่อพิจารณาถึงประเภทความสัมพันธ์ 3 ประเภทของแผนภาพคลาส ได้แก่ แอกกรีเกชัน คอมโพซิชัน และดีเพนเดนซี จะพบว่าการสร้างและทำลายคลาสมีข้อบังคับบางอย่างที่ต้องตรวจสอบดังนี้

ความสัมพันธ์แบบแอกกรีเกชันเป็นความสัมพันธ์แบบส่วนหลัก-ส่วนประกอบ กล่าวคือคลาสหลักคลาสหนึ่งมีส่วนประกอบได้หลายอย่าง และคลาสที่เป็นส่วนประกอบนั้นยังสามารถเป็นส่วนประกอบของคลาสหลักได้หลายคลาส คลาสย่อยของความสัมพันธ์แบบนี้จึงไม่สามารถส่งข้อความแบบสร้างไปสร้างคลาสหลักดังรูปที่ 3.3 ได้



รูปที่ 3.3 คลาสย่อยที่ส่งข้อความแบบสร้างไปยังคลาสหลักที่มีความสัมพันธ์กันแบบแอกกรีเกชัน

จากรูปที่ 3.3 แสดงคลาส MotherClass และ SubClass ที่มีความสัมพันธ์กันแบบแอกกรีเกชัน และ SubClass ซึ่งเป็นคลาสย่อยส่งข้อความแบบสร้างไปสร้าง MotherClass ซึ่งเป็นคลาสหลักในแผนภาพลำดับ ซึ่งไม่สามารถทำได้

นอกจากนี้ยังจำเป็นต้องตรวจสอบความสัมพันธ์แบบเจนเนอรอลไลเซชันด้วย เนื่องจากถ้าคลาสแม่ของความสัมพันธ์แบบเจนเนอรอลไลเซชันเป็นคลาสย่อยของความสัมพันธ์แอกกรีเกชัน คลาสลูกของความสัมพันธ์แบบเจนเนอรอลไลเซชันจะสืบทอดความเป็นคลาสย่อยของความสัมพันธ์แอกกรีเกชันมาด้วยดังนั้นจึงสรุปออกมาเป็นกฎได้ดังนี้

กฎข้อที่ 5 คลาสย่อยไม่สามารถส่งข้อความแบบสร้างในแผนภาพลำดับไปสร้างคลาสหลักได้ ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบแอกกรีเกชันในแผนภาพคลาส

จากกฎข้อที่ 5 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบข้อความแบบสร้างในแผนภาพลำดับ และเก็บรวบรวมไว้เป็นคู่ส่ง-รับข้อความ

2) นำไปตรวจสอบว่าเป็นการส่งจากคลาสที่มีฐานะเป็นคลาสย่อยและคลาสหลักเป็นผู้รับหรือไม่ ถ้าใช่แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

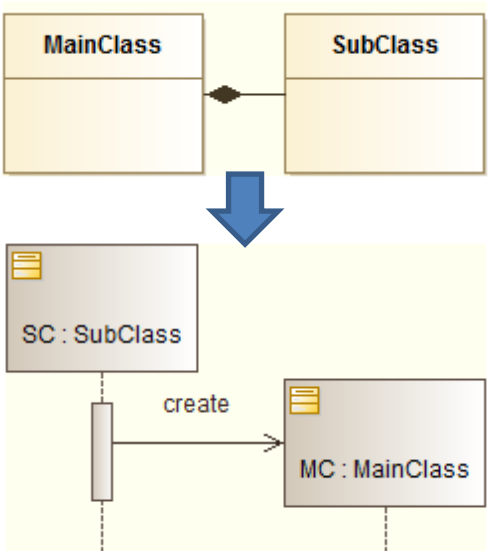
3) ถ้าไม่ใช่ ให้ตรวจสอบว่าคลาสที่ส่งเป็นคลาสลูกของความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือไม่

4) ถ้าใช่ ให้ตรอจสอบว่าคลาสแม่นั้นเป็นคลาสย่อยของความสัมพันธ์แบบแอกกรีเกชันหรือไม่

5) ถ้าใช่ ให้ตรวจสอบว่าข้อความนั้นส่งไปยังคลาสหลักของความสัมพันธ์แบบแอกกรีเกชันดังกล่าวหรือไม่

6) ถ้าใช่ แผนภาพลำดับนั้น ไม่สอดคล้องกับแผนภาพคลาส

ความสัมพันธ์แบบคอมโพซิชันต่างจากความสัมพันธ์แบบแอกกรีเกชันที่แสดงความเป็นเจ้าของมากกว่า กล่าวคือคลาสหลักอาจประกอบด้วยคลาสย่อยได้หลายคลาส แต่คลาสย่อยไม่สามารถไปเป็นส่วนประกอบของคลาสหลักได้หลายคลาส ต้องเป็นส่วนประกอบของคลาสใดคลาสหนึ่งเท่านั้น และเมื่อคลาสหลักถูกทำลายลง คลาสย่อยก็ต้งถูกทำลายลงเช่นกัน ดังนั้นคลาสหลักจึงไม่สามารถส่งข้อความทั้งแบบสร้าง และแบบทำลายไปยังคลาสหลักดังรูปที่ 3.4 และ 3.5 ได้



รูปที่ 3.4 คลาสย่อยที่ส่งข้อความแบบสร้างไปยังคลาสหลักที่มีความสัมพันธ์กันแบบคอมโพซิชัน

จากรูปที่ 3.4 แสดงคลาส MainClass และ SubClass ที่มีความสัมพันธ์กันแบบคอมโพซิชัน และ SubClass ซึ่งเป็นคลาสย่อยส่งข้อความแบบสร้างไปสร้าง MainClass ซึ่งเป็นคลาสหลักในแผนภาพลำดับ ซึ่งไม่สามารถทำได้

เช่นเดียวกันกับความสัมพันธ์แบบแอกกรีเกชัน การตรวจสอบนี้จำเป็นจะต้องตรวจสอบความสัมพันธ์แบบเจนเนอรอลไลเซชันด้วย เพราะอาจมีการถ่ายทอดความเป็นคลาสย่อยไปสู่คลาสลูกได้ดังสรุปออกมาเป็นกฎได้ดังนี้

กฎข้อที่ 6 คลาสย่อยไม่สามารถส่งข้อความแบบสร้างในแผนภาพลำดับไปสร้างคลาสหลักได้ ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส

จากกฎข้อที่ 6 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบข้อความแบบสร้างในแผนภาพลำดับ และเก็บรวบรวมไว้เป็นคู่ส่ง-รับข้อความ

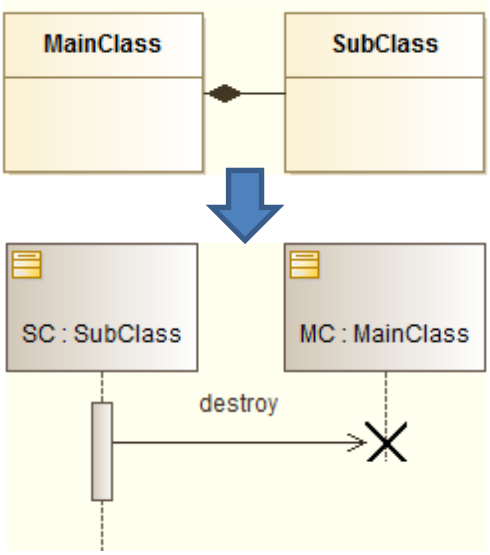
2) นำไปตรวจสอบว่าเป็นการส่งจากคลาสที่มีฐานะเป็นคลาสย่อยและคลาสหลักเป็นผู้รับหรือไม่ ถ้าใช่แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3) ถ้าไม่ใช่ ให้ตรวจสอบว่าคลาสที่ส่งเป็นคลาสลูกของความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือไม่

4) ถ้าใช่ ให้ตรอจสอบว่าคลาสแม่นั้นเป็นคลาสย่อยของความสัมพันธ์แบบคอมโพซิชันหรือไม่

5) ถ้าใช่ ให้ตรวจสอบว่าข้อความนั้นส่งไปยังคลาสหลักของความสัมพันธ์แบบคอมโพซิชันดังกล่าวหรือไม่

6) ถ้าใช่ แผนภาพลำดับนั้น ไม่สอดคล้องกับแผนภาพคลาส



รูปที่ 3.5 คลาสย่อยที่ส่งข้อความแบบทำลายไปยังคลาสหลักที่มีความสัมพันธ์กันแบบคอมโพซิชัน

จากรูปที่ 3.5 แสดงคลาส MainClass และ SubClass ที่มีความสัมพันธ์กันแบบคอมโพซิชัน และ SubClass ซึ่งเป็นคลาสย่อยส่งข้อความแบบทำลายไปทำลาย MainClass ซึ่งเป็นคลาสหลักในแผนภาพลำดับ ซึ่งไม่สามารถทำได้ ดังนั้นสรุปออกมาเป็นกฎได้ดังนี้

กฎข้อที่ 7 คลาสย่อยไม่สามารถส่งข้อความแบบทำลายในแผนภาพลำดับไปทำลายคลาสหลักได้ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส

จากกฎข้อที่ 7 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบข้อความแบบทำลายในแผนภาพลำดับ และเก็บรวบรวมไว้เป็นคู่ส่ง-รับข้อความ

2) นำไปตรวจสอบว่าเป็นการส่งจากคลาสที่มีฐานะเป็นคลาสย่อยและคลาสหลักเป็นผู้รับหรือไม่ ถ้าใช่แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

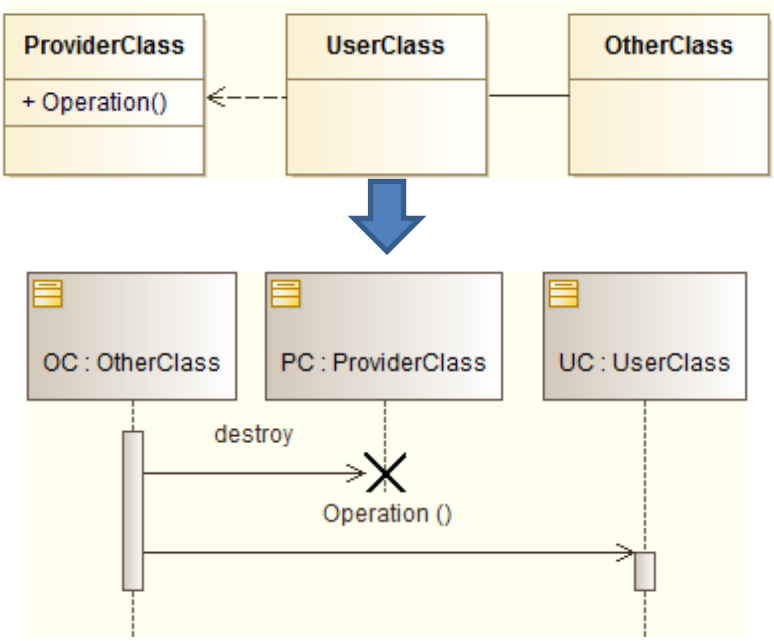
3) ถ้าไม่ใช่ ให้ตรวจสอบว่าคลาสที่ส่งเป็นคลาสลูกของความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือไม่

4) ถ้าใช่ ให้ตรอจสอบว่าคลาสแม่นั้นเป็นคลาสย่อยของความสัมพันธ์แบบคอมโพซิชันหรือไม่

5) ถ้าใช่ ให้ตรวจสอบว่าข้อความนั้นส่งไปยังคลาสหลักของความสัมพันธ์แบบคอมโพซิชันดังกล่าวหรือไม่

6) ถ้าใช่ แผนภาพลำดับนั้น ไม่สอดคล้องกับแผนภาพคลาส

ความสัมพันธ์แบบดีเพนเดนซีเป็นความสัมพันธ์แบบผู้ใช้บริการ-ผู้ให้บริการ แม้คลาสผู้ใช้บริการจะไม่มีปัญหาในการส่งข้อความแบบสร้างหรือทำลายไปยังคลาสผู้ให้บริการ แต่เมื่อคลาสผู้ให้บริการถูกทำลายลงไปแล้ว คลาสผู้ใช้บริการจะเรียกใช้เมธอดของคลาสผู้ให้บริการดังรูปที่ 3.6 อีกไม่ได้ เช่นเดียวกับความสัมพันธ์แบบคอมโพซิชัน เมื่อคลาสหลักถูกทำลาย คลาสย่อยก็จะถูกทำลายลงไปด้วย



รูปที่ 3.6 คลาสย่อยเรียกใช้คลาสหลัก หลังจากคลาสหลักถูกทำลายไปแล้ว

จากรูปที่ 3.6 คลาส ProviderClass และคลาส UserClass มีความสัมพันธ์กันแบบดีเพนเดนซี โดยหลังจากส่วนของระบบ PC ซึ่งเป็นคลาสผู้ให้บริการถูกทำลายไปแล้วในแผนภาพลำดับ เมธอด Operation ซึ่งเป็นเมธอดของ PC กลับถูกเรียกใช้ผ่าน UC ซึ่งเป็นคลาสผู้ใช้บริการ ซึ่งไม่สามารถทำได้

เช่นเดียวกันกับความสัมพันธ์แบบแอกกรีเกชันและคอมโพซิชัน การตรวจสอบนี้จำเป็นจะต้องตรวจสอบความสัมพันธ์แบบเจนเนอรอลไลเซชันด้วย เพราะอาจมีการถ่ายทอดความเป็นคลาสย่อยไปสู่คลาสลูกได้ ดังนั้นจึงสรุปออกมาเป็นกฎได้ดังนี้

กฎข้อที่ 8 ไม่สามารถเรียกคลาสผู้ให้บริการผ่านคลาสผู้ใช้บริการได้ ถ้าคลาสผู้ให้บริการถูกทำลายไปแล้ว

จากกฎข้อที่ 8 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบหาความสัมพันธ์แบบดีเพนเดนซีในแผนภาพคลาส แล้วรวบรวมคู่ความสัมพันธ์ไว้เป็นคู่ผู้ให้บริการ-ผู้ใช้บริการ

2) ตรวจสอบหาข้อความแบบทำลายในแผนภาพลำดับ เทียบกับคู่ความสัมพันธ์ที่เก็บข้อมูลไว้ ถ้ามีการทำลายคลาสผู้ให้บริการ ให้เก็บค่าลำดับการเรียกใช้ไว้

3) ตรวจสอบการเรียกใช้คลาสผู้ให้บริการผ่านคลาสผู้ใช้บริการที่เกิดขึ้นทั้งหมด แล้วเทียบลำดับการเรียกใช้ หากลำดับการเรียกใช้ใดมีค่ามากกว่า (เรียกใช้หลังจากถูกทำลายไปแล้ว) แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

4) ตรวจสอบการเรียกใช้คลาสผู้ให้บริการผ่านคลาสลูกที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชัน ว่าคลาสแม่มีความสัมพันธ์กับเป็นคลาสผู้ใช้บริการหรือไม่

5) ถ้าใช่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

กฎข้อที่ 9 ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส เมื่อคลาสหลักถูกทำลายในแผนภาพลำดับ คลาสย่อยจะถูกทำลายด้วย

จากกฎข้อที่ 9 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบหาความสัมพันธ์แบบคอมโพซิชันในแผนภาพคลาส แล้วรวบรวมคู่ความสัมพันธ์ไว้เป็นคู่คลาสหลัก-คลาสย่อย

2) ตรวจสอบหาข้อความแบบทำลายในแผนภาพลำดับ เทียบกับคู่ความสัมพันธ์ที่เก็บข้อมูลไว้ ถ้ามีการทำลายคลาสหลัก ให้เก็บค่าลำดับการเรียกใช้ไว้

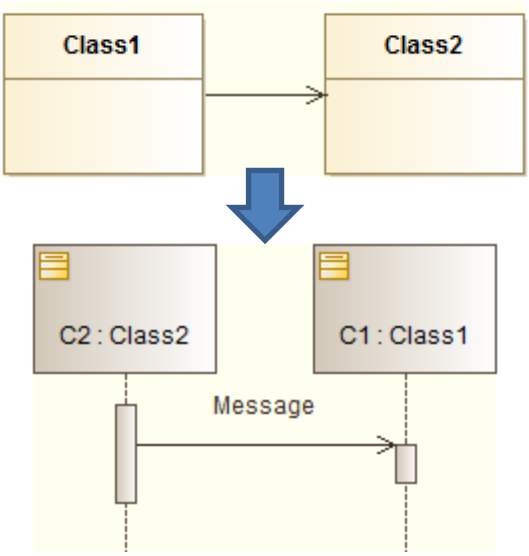
3) ตรวจสอบการเรียกใช้คลาสหลักผ่านคลาสย่อยที่เกิดขึ้นทั้งหมด แล้วเทียบลำดับการเรียกใช้ หากลำดับการเรียกใช้ใดมีค่ามากกว่า (เรียกใช้หลังจากถูกทำลายไปแล้ว) แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

4) ตรวจสอบการเรียกใช้คลาสหลักผ่านคลาสย่อยที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชัน ว่าคลาสแม่มีความสัมพันธ์กับเป็นคลาสย่อยหรือไม่

5) ถ้าใช่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.5 แม้จะพิจารณาการมีอยู่จริงของความสัมพันธ์ระหว่างคลาสต่าง ๆ ไปแล้วในหัวข้อที่ 3.2.3 แต่สิทธิ์การเข้าถึงอื่น ๆ ก็จะต้องนำมาพิจารณาร่วมด้วยได้แก่ ทิศทางของความสัมพันธ์ การถ่ายทอดคุณสมบัติของความสัมพันธ์แบบเจนเนอรอลไลเซชัน และการเข้าถึงของเมธอดซึ่งเป็นส่วนที่เพิ่มเติมเข้ามาจากงานวิจัยอื่น ๆ ดังนี้

ทิศทางของความสัมพันธ์มีสองแบบคือแบบทางเดียวและสองทาง ถ้าความสัมพันธ์เป็นแบบทางเดียว คลาสที่หางลูกศรจะเรียกคลาสที่หัวลูกศรได้เท่านั้น เรียกกลับกันดังรูปที่ 3.7 ไม่ได้



รูปที่ 3.7 คลาสด้านหางลูกศรเรียกใช้คลาสด้านหัวลูกศร

จากรูปที่ 3.7 คลาส Class1 มีความสัมพันธ์แบบทางเดียวกับคลาส Class2 ในแผนภาพคลาส แต่ในแผนภาพลำดับส่วนประกอบบระบบ C2 ซึ่งเป็นคลาสด้านหัวลูกศรกลับส่งข้อความหา C1 ซึ่งเป็นคลาสด้านหางลูกศร ซึ่งไม่ถูกต้อง ดังนั้นจึงสรุปออกมาเป็นกฎข้อที่ 10.1 ได้ดังนี้

กฎข้อที่ 10.1 ทิศทางของความสัมพันธ์ ถ้าเป็นความสัมพันธ์ทางเดียว คลาสด้านหัวลูกศรจะเรียกคลาสอีกด้านไม่ได้

จากกฎข้อที่ 10.1 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบหาความสัมพันธ์แบบทางเดียวในแผนภาพคลาส แล้วรวบรวมคู่ความสัมพันธ์ไว้เป็นคลาสผู้ส่ง-ผู้รับ

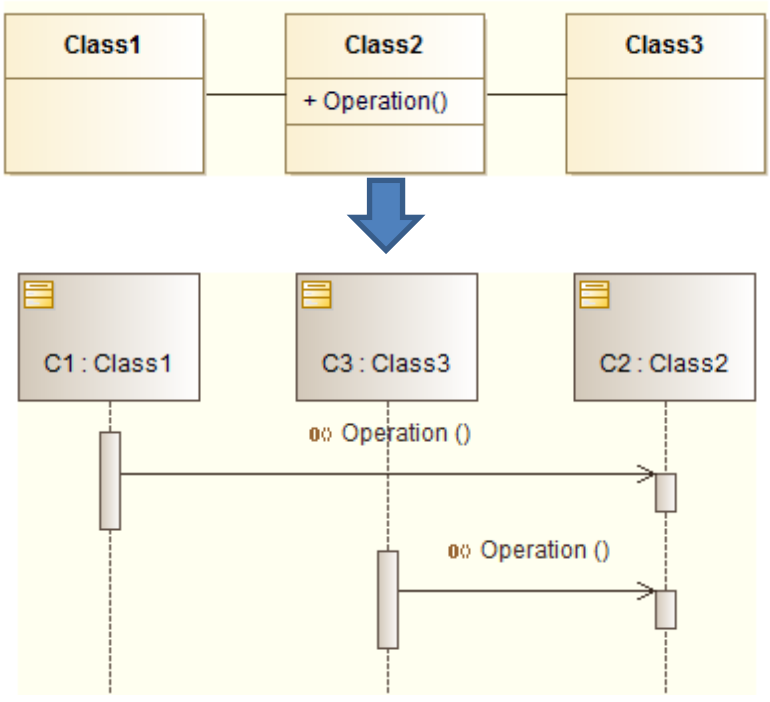
2) ตรวจสอบข้อความทั้งหมดในแผนภาพลำดับ แล้วรวบรวมคู่ผู้ส่ง-ผู้รับของข้อความทั้งหมดไว้

3) เปรียบเทียบคู่ผู้ข้อความกับคู่ความสัมพันธ์ทางเดียวว่าผู้ส่งข้อความเป็นผู้รับของความสัมพันธ์ทางเดียวหรือไม่

4) ถ้าใช่ ตรวจสอบผู้รับข้อความเป็นผู้ส่งของความสัมพันธ์ทางเดียวหรือไม่

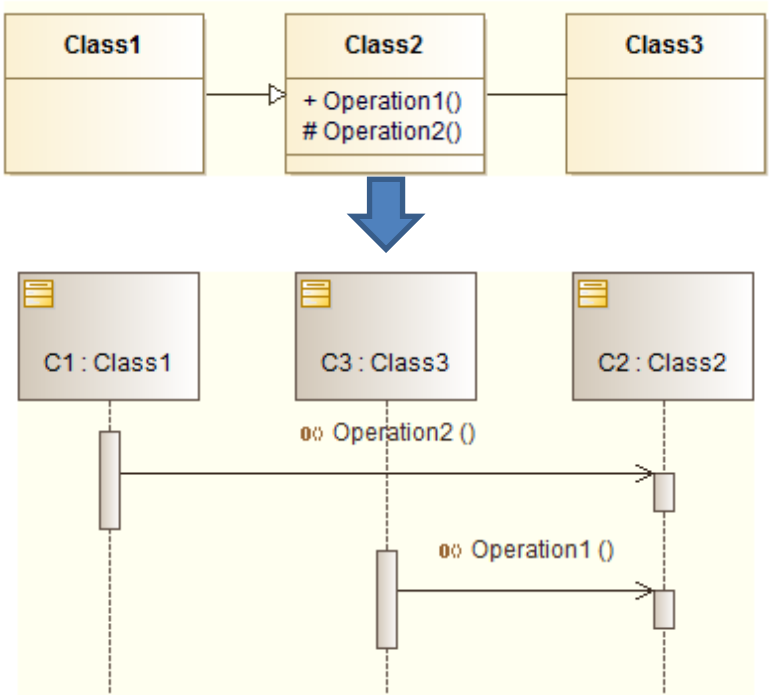
5) ถ้าใช่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

การเข้าถึงเมธอดบ่งบอกสิทธิ์ของการเรียกใช้เมธอดนั้น ๆ แก่คลาสที่เรียก กล่าวคือ ถ้าเป็นการเข้าถึงแบบสาธาณะ คลาสทุกคลาสที่มีความสัมพันธ์กันจะมีสิทธิ์เรียกใช้ได้หมดดังรูปที่ 3.8 ถ้าเป็นการเข้าถึงแบบป้องกัน คลาสลูกที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชันเท่านั้นที่จะถูกถ่ายทอดคุณสมบัติมาให้คลาสลูกเรียกใช้ได้ดังรูปที่ 3.9 และถ้าเป็นการเข้าถึงแบบส่วนตัว จะมีแต่คลาสตัวเองเท่านั้นที่มีสิทธิ์เรียกใช้ได้ดังรูปที่ 3.10



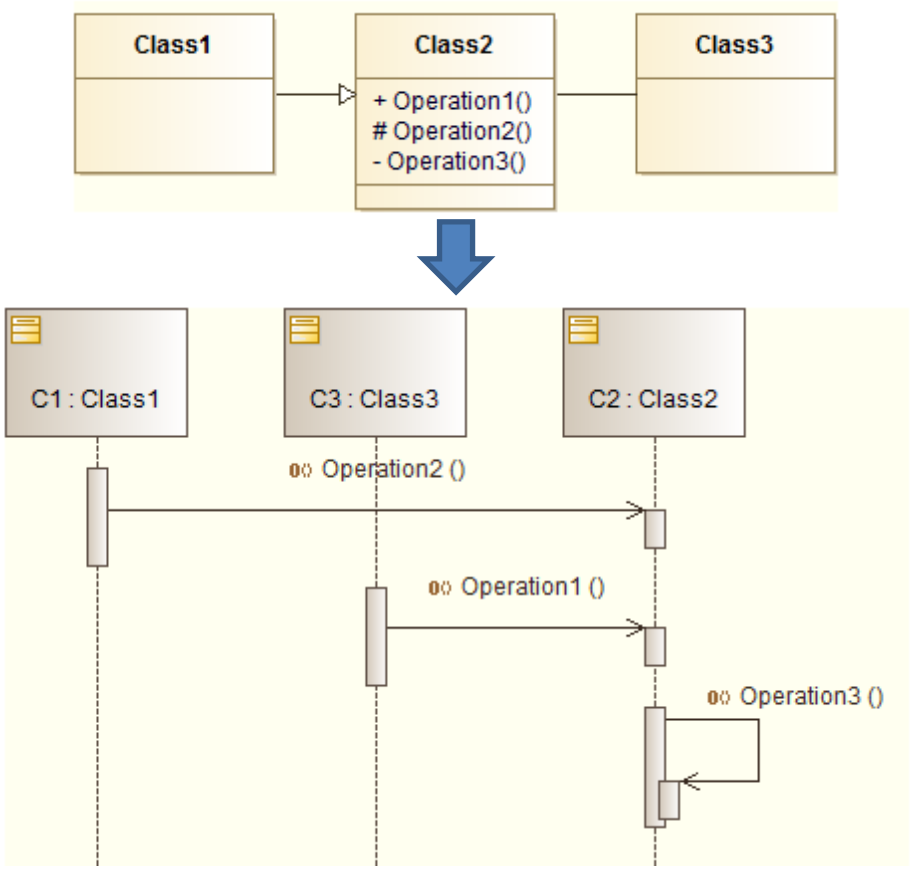
รูปที่ 3.8 เมธอดที่มีการเข้าถึงแบบสาธารณะถูกคลาสอื่น ๆ เรียกใช้

จากรูปที่ 3.8 แสดงเมธอด Operation ของคลาส Class2 ในแผนภาพคลาส ที่มีการเข้าถึงเป็นสาธารณะ ดังนั้นในแผนภาพลำดับ ส่วนประกอบของระบบ C1 และ C3 สามารถส่งข้อความเรียกใช้เมธอด Operation นี้ได้ โดยไม่ต้องมีความสัมพันธ์แบบพิเศษใด ๆ ในแผนภาพคลาส



รูปที่ 3.9 เมธอดที่มีการเข้าถึงแบบป้องกันและสาธารณะถูกคลาสอื่น ๆ เรียกใช้

จากรูปที่ 3.9 คลาส Class2 มีเมธอด Operation1 ที่มีการเข้าถึงเป็นสาธารณะ และ Operation 2 ที่มีการเข้าถึงเป็นป้องกันในแผนภาพคลาส ดังนั้นในแผนภาพลำดับส่วนประกอบของระบบ C1 ซึ่งมีความสัมพันธ์กับ C2 เป็นคลาสแม่แบบเจนเนอรอลไลเซชันจึงจะส่งข้อความเรียกใช้เมธอด Operation2 ได้ ส่วน C3 ซึ่งมีความสัมพันธ์แบบแอสโซซิเอชันกับ C2 ก็ส่งข้อความเรียกใช้ได้เพียงเมธอด Operation1 เท่านั้น



รูปที่ 3.10 เมธอดที่มีการเข้าถึงแบบส่วนตัวถูกคลาสตัวเองเรียกใช้

จากรูปที่ 3.10 คลาส Class2 มีเมธอด 3 เมธอดในแผนภาพคลาส ได้แก่ Operation1 ที่มีการเข้าถึงเป็นสาธารณะ Operation2 ที่มีการเข้าถึงเป็นป้องกัน และ Operation3 ที่มีการเข้าถึงเป็นส่วนตัว ในแผนภาพลำดับส่วนประกอบของระบบ C1 สามารถส่งข้อความเรียกเมธอด Operation2 ไปยัง C2 ซึ่งมีความสัมพันธ์เป็นคลาสแม่แบบเจนเนอรอลไลเซชันได้ ส่วนประกอบของระบบ C3 ส่งข้อความเรียกเมธอด Operation1 ได้อย่างเดียว เพราะมีความสัมพันธ์กับ C2 แบบแอสโซซิเอชัน ส่วนเมธอด Operation3 จะถูกเรียกใช้ผ่านการส่งข้อความหาตัวเองของ C2 ได้เท่านั้น ดังนั้นจึงสรุปออกมาเป็นกฎข้อที่ 10.2 ได้ดังนี้

กฎข้อที่ 10.2 การเข้าถึงของเมธอด ถ้าการเข้าถึงเป็นแบบสาธารณะคลาสที่สัมพันธ์กันจะมีสิทธิ์เข้าถึง ถ้าเป็นการเข้าถึงแบบป้องกัน คลาสลูกและตัวเองเท่านั้นที่มีสิทธิ์เข้าถึง ถ้าเป็นการเข้าถึงแบบส่วนตัว คลาสตัวเองเท่านั้นที่มีสิทธิ์เข้าถึง

จากกฎข้อที่ 10.2 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบการเข้าถึงของเมธอดทั้งหมดในแผนภาพลำดับ เก็บข้อมูลการเข้าถึงของเมธอดไว้

2) ตรวจสอบการส่งข้อความทั้งหมดในแผนภาพลำดับ ถ้าเป็นข้อความที่เรียกใช้เมธอดมีการเข้าถึงเป็นส่วนตัว ให้ตรวจสอบว่าผู้รับและผู้ส่งเป็นคลาสเจ้าของเมธอดหรือไม่

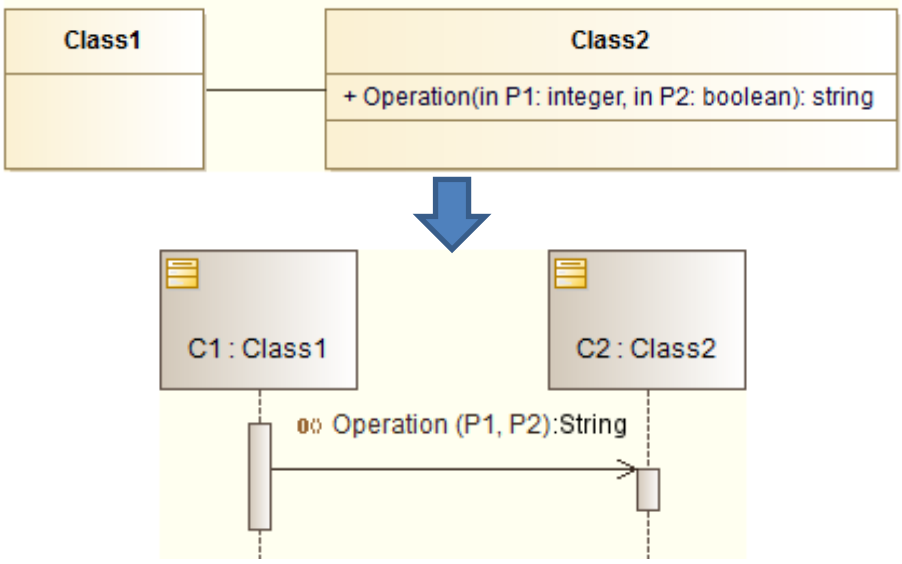
3) ถ้าไม่ใช่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

4) ถ้าข้อความที่ใช้เรียกเมธอดมีการเข้าถึงเป็นป้องกัน ให้ตรวจสอบว่าผู้รับเป็นคลาสเจ้าของเมธอดหรือไม่

5) ถ้าไม่ ให้ตรวจสอบต่อว่าผู้รับเป็นคลาสลูกของคลาสเจ้าของเมธอดที่มีความสัมพันธ์แบบเจนเนอรอลไลเซชันหรือไม่

6) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.6 ข้อความในแผนภาพลำดับ อาจมีพารามิเตอร์กำกับอยู่ได้ ทั้งพารามิเตอร์เข้าและคืนค่า ทั้งนี้พารามิเตอร์เหล่านั้นจะต้องสอดคล้องกับที่กำหนดไว้ในแผนภาพคลาส ทั้งจำนวน ลำดับ (กรณีมีพารามิเตอร์เข้าหลายตัว) และชนิด ดังรูปที่ 3.11



รูปที่ 3.11 พารามิเตอร์ที่ปรากฏอยู่บนข้อความในแผนภาพลำดับและแผนภาพคลาส

จากรูปที่ 3.11 คลาส Class2 ในแผนภาพคลาสมีเมธอด Operation ที่ต้องการค่าพารามิเตอร์เข้า P1 และ P2 แล้วคืนค่าเป็นชนิด String ในแผนภาพลำดับ ส่วนประกอบของระบบ C1 จึงส่งข้อความเรียกใช้เมธอด Operation โดยระบุพารามิเตอร์เข้า P1 และ P2 และยังต้องระบุการคืนค่าเป็น String อีกด้วย

โดยกฎข้อนี้ต้องพิจารณาเมธอดแบบ overloading method ร่วมด้วย เนื่องจาก overloading method มีชื่อเหมือนกันแต่ชนิดหรือจำนวนตัวแปรไม่เท่ากัน ดังนั้นจึงสรุปออกมาเป็นกฎข้อที่ 11 ได้ดังนี้

กฎข้อที่ 11 จำนวน ลำดับ และชนิดของพารามิเตอร์ของเมธอดบนข้อความในแผนภาพลำดับจะต้องเหมือนกับที่ประกาศไว้ในคลาสของแผนภาพคลาส

จากกฎข้อที่ 11 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบเมธอดแบบ overloading method และเก็บรวบรวมข้อมูลชื่อ ชนิด ลำดับ และจำนวนพารามิเตอร์ไว้ ทั้งพารามิเตอร์เข้าและคืนค่า

2) ตรวจสอบพารามิเตอร์บนข้อความทุกข้อความว่าเรียกใช้เมธอดแบบ overloading method หรือไม่

3) ถ้าใช่ ให้ตรวจสอบจำนวนพารามิเตอร์ทั้งหมดของข้อความ ว่ามีเท่ากับจำนวนพารามิเตอร์ของ overloading method ใดอันใดอันหนึ่งหรือไม่

4) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส ถ้าใช่ตรวจสอบชื่อและชนิดของพารามิเตอร์ ว่าสอดคล้องกันหรือไม่

5) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

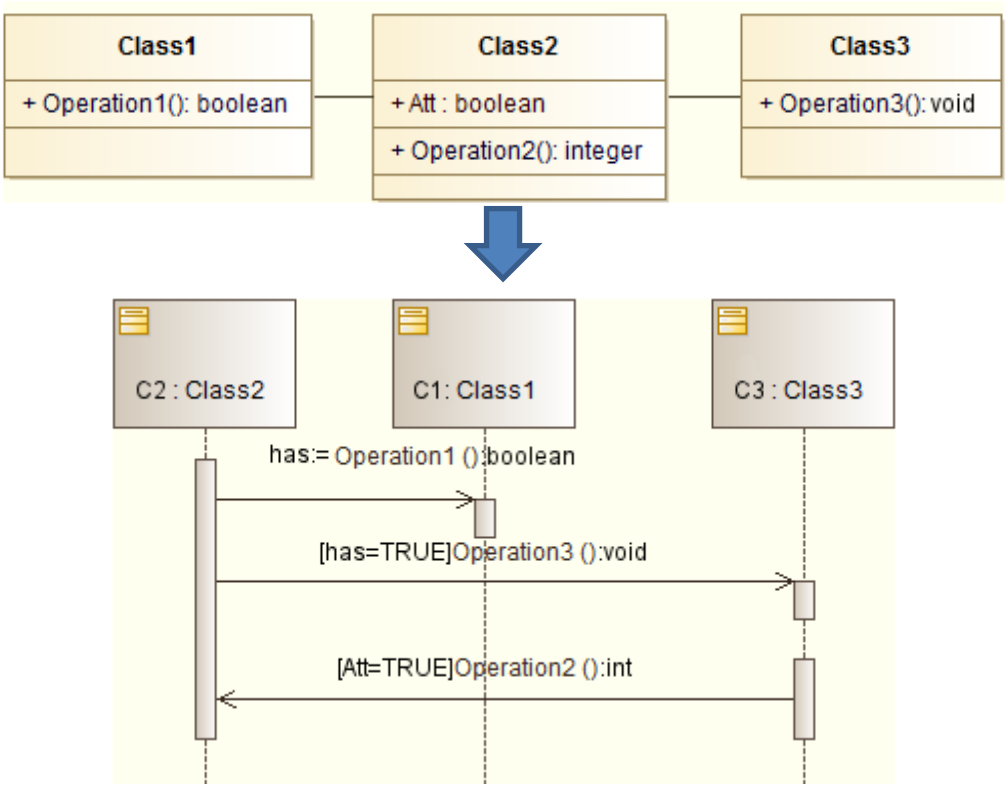
6) แต่ถ้าข้อความนั้นไม่ได้เรียกใช้เมธอดแบบ overloading method ให้ตรวจสอบชื่อข้อความว่าตรงกับชื่อเมธอดบนคลาสนั้นหรือไม่

7) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส ถ้าใช่ตรวจสอบว่าจำนวนพารามิเตอร์บนข้อความเท่ากับจำนวนพารามิเตอร์ที่ประกาศไว้ในเมธอดนั้นหรือไม่

8) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส ถ้าใช่ตรวจสอบชื่อและชนิดของพารามิเตอร์ว่าสอดคล้องกันหรือไม่

9) ถ้าไม่ แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.7 นอกจากพารามิเตอร์เข้าหรือคืนค่าที่ได้ตรวจสอบในข้อ 3.2.7 แล้ว ข้อความบนแผนภาพลำดับยังอาจมีเงื่อนไขซึ่งเขียนอยู่ในเครื่องหมาย “[]” ซึ่งเงื่อนไขดังกล่าวอาจกำหนดมาจากตัวแปรจากข้อความก่อนหน้า หรือประกาศไว้เป็นคุณลักษณะบนคลาสที่ถูกเรียก ดังรูปที่ 3.12 ทั้งนี้ยังพิจารณาการถ่ายทอดคุณสมบัติของความสัมพันธ์แบบเจนเนอรอลไลเซชันร่วมด้วย



รูปที่ 3.12 เงื่อนไขบนข้อความในแผนภาพลำดับ ซึ่งประกาศเป็นคุณลักษณะ

จากรูปที่ 3.12 มีคลาส Class1 Class2 และ Class3 ในแผนภาพคลาส โดยแต่ละคลาสมีเมธอดคลาสละ 1 เมธอดได้แก่ Operation1 Operation2 และ Operation3 ตามลำดับ และคลาส Class2 มีคุณลักษณะชื่อ Att ชนิด boolean อยู่ 1 คุณลักษณะ ในแผนภาพลำดับ เริ่มต้นจากส่วนประกอบของระบบ C2 ส่งข้อความเรียกเมธอด Operation1 ไปยัง C1 โดยกำหนดตัวแปร has ขึ้นมาให้รับค่าคืนเป็น boolean ต่อมา C2 ส่งข้อความไป C3 เรียกใช้เมธอด Operation3 โดยมีเงื่อนไขว่าค่าตัวแปร has ที่รับค่ามาก่อนหน้าต้องเป็น TRUE และสุดท้าย C3 ส่งข้อความไป C1 เรียกใช้เมธอด Operation2 โดยมีเงื่อนไขว่าคุณลักษณะ Att ของ C1 ต้องเป็น TRUE

เนื่องจากกฎข้อนี้เกี่ยวข้องกับเมธอด จึงจำเป็นต้องตรวจสอบความสัมพันธ์แบบเจนเนอรอลไลเซชันด้วย เพราะอาจมีคลาสลูกสืบทอดคุณลักษณะจากคลาสแม่เข้ามาด้วยนั่นเอง ซึ่งสรุปออกมาเป็นกฎข้อที่ 12 ได้ดังนี้

กฎข้อที่ 12 ตัวแปร และชนิดของตัวแปรที่เป็นตัวกำหนดเงื่อนไขในเครื่องหมาย '[]' บนข้อความในแผนภาพลำดับ จะต้องประกาศเป็นคุณลักษณะในคลาสที่ถูกเรียก หรือตัวแปรจากข้อความก่อนหน้า

จากกฎข้อที่ 12 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบข้อความในแผนภาพลำดับทั้งหมดว่าข้อความใดมีการกำหนดเงื่อนไขไว้บ้าง เก็บข้อมูลไว้ทั้งชื่อและชนิดของเงื่อนไข

2) ตรวจสอบข้อความในแผนภาพลำดับทั้งหมดว่าข้อความใดมีการคืนค่าบ้าง เก็บข้อมูลการคืนค่าไว้ทั้งชื่อและชนิดของการคืนค่า

3) นำชื่อเงื่อนไขไปตรวจสอบกับคุณลักษณะทั้งหมดของคลาสที่ถูกเรียก

4) ถ้าชื่อตรงกันให้ตรวจสอบชนิดของคุณลักษณะนั้น

5) ถ้าชนิดตรงกัน ให้สะสมคะแนนไว้

6) ถ้าคลาสที่ถูกเรียกเป็นคลาสลูกในความสัมพันธ์แบบเจนเนอรอลไลเซชัน ให้ไปตรวจสอบชื่อและชนิดของคุณลักษณะของคลาสแม่ด้วย ถ้าตรงกันให้สะสมคะแนนไว้

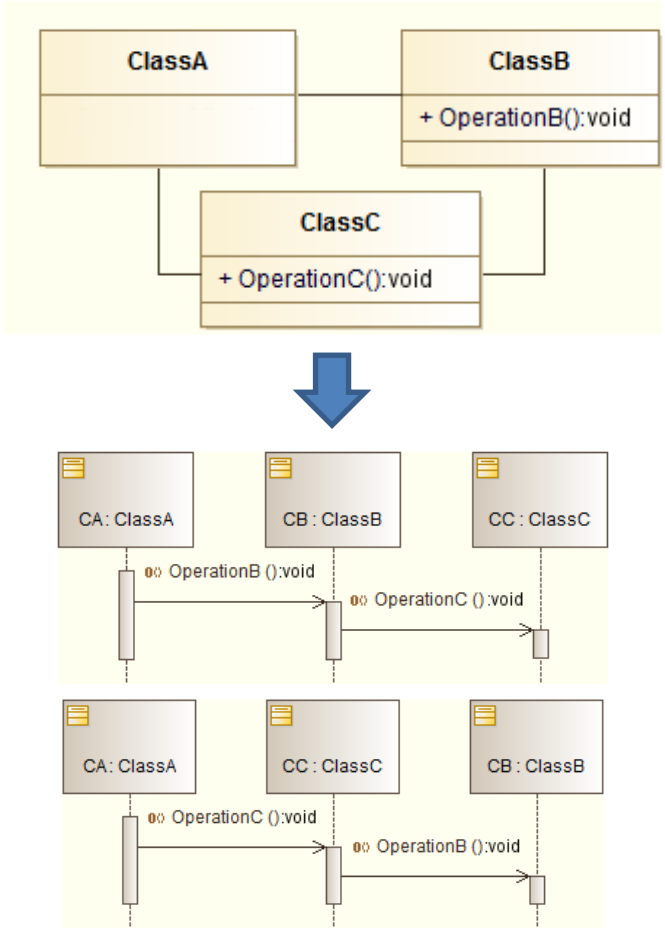
7) นำชื่อเงื่อนไขไปตรวจสอบกับชื่อตัวแปรคืนค่าในแผนภาพลำดับเดียวกัน

8) ถ้าชื่อตรงกันให้ตรวจสอบชนิดของตัวแปรคืนค่านั้น

9) ถ้าชนิดตรงกัน ให้สะสมคะแนนไว้

10) เงื่อนไขใดไม่มีคะแนนสะสมอยู่เลย แผนภาพลำดับนั้นไม่สอดคล้องกับแผนภาพคลาส

3.2.8 เมื่อพิจารณาลำดับการส่งข้อความในแผนภาพลำดับแล้ว เราอาจแบ่งข้อความที่เรียกส่วนประกอบของระบบสามส่วนต่อกันเป็นข้อความหลักและข้อความย่อย โดยข้อความที่ส่งเรียกส่วนประกอบของระบบที่หนึ่งไปยังส่วนประกอบของระบบที่สองถือเป็นข้อความหลัก และข้อความที่ส่งเรียกเมธอดที่สองไปยังส่วนประกอบของระบบที่สามถือเป็นข้อความย่อย ในแผนภาพลำดับหนึ่งข้อความหลักจะกลายเป็นข้อความย่อยในแผนภาพลำดับอีกแผนภาพหนึ่งไม่ได้ เพราะจะเกิดการทำงานแบบเรียกกันไปเรื่อย ๆ คล้ายการเวียนเกิด (recursive) ดังรูปที่ 3.13



รูปที่ 3.13 เมธอดหลักในแผนภาพหนึ่ง กลายเป็นเมธอดย่อยในอีกแผนภาพหนึ่ง

จากรูปที่ 3.13 คลาส ClassA ClassB และ ClassC มีความสัมพันธ์กันแบบแอสโซซิเอชันในแผนภาพคลาส โดยแต่คลาส ClassB และ ClassC มีเมธอดคลาสละหนึ่งเมธอดได้แก่ OperationB และ OperationC ตามลำดับ ในแผนภาพลำดับด้านซ้าย ส่วนของระบบ CA ส่งข้อความเรียกใช้เมธอด OperationB ไปยัง CB แล้ว CB ส่งข้อความเรียกใช้ OperationC ไปยัง CC ต่อ ทำให้ OperationA เป็นเมธอดหลัก และ OperationC เป็นเมธอดย่อย แต่ในแผนภาพลำดับด้านขวา ส่วนของระบบ CA ส่งข้อความเรียกใช้เมธอด OperationC ไปยัง CC แล้ว CC ส่งข้อความเรียกใช้เมธอด OperationB ไปยัง CB ต่อ ทำให้เมธอด OperationC ซึ่งเป็นเมธอดย่อย กลายเป็นเมธอดหลัก และ OperationA ซึ่งเป็น เมธอดหลัก กลายเป็นเมธอดย่อย ดังนั้นจะสรุปออกมาเป็นกฎข้อที่ 13 ได้ดังนี้

กฎข้อที่ 13 ข้อความที่ส่งต่อกันระหว่างส่วนของระบบสามส่วนในแผนภาพลำดับ ที่มีลักษณะเป็นข้อความหลักและข้อความย่อยแล้ว ถ้ามีแผนภาพลำดับอื่นที่มีข้อความตัวเดียวกันกับแผนภาพลำดับแรก ข้อความย่อยจะต้องไม่เรียกข้อความหลัก

จากกฎข้อที่ 13 สามารถอธิบายขั้นตอนการตรวจสอบความสอดคล้องของแผนภาพลำดับกับแผนภาพคลาสได้ดังนี้

1) ตรวจสอบการส่งข้อความบนแผนภาพลำดับ และเก็บข้อมูลคู่ผู้รับ-ผู้ส่งและลำดับการส่งไว้

2) ตรวจสอบว่ามีข้อความใดมีลักษณะการส่งต่อเป็นทอด ๆ หรือไม่ โดยดูไล่ตามลำดับการส่งว่าผู้รับในลำดับก่อนหน้าเหมือนผู้ส่งในลำดับถัดไปหรือไม่

3) ถ้าใช่ ถือว่าข้อความมีลักษณะการส่งเป็นทอด ๆ ให้เก็บข้อมูลไว้เป็นคู่เมธอดหลัก-เมธอดย่อย

4) เมื่อตรวจสอบข้อความจนครบทุกแผนภาพแล้ว ให้ตรวจสอบว่ามีเมธอดหลักของข้อมูลที่เก็บมา เหมือนกับเมธอดย่อยใดในแผนภาพลำดับอื่น ๆ หรือไม่

5) ถ้ามี ให้ตรวจสอบว่าเมธอดย่อยเหมือนเมธอดหลักในคู่เดียวกันหรือไม่

6) ถ้าใช่ แผนภาพลำดับสองแผนภาพนั้นไม่สอดคล้องกัน

บทที่ 4

การออกแบบและพัฒนาเครื่องมือการตรวจสอบความสอดคล้องของแผนภาพคลาสและแผนภาพลำดับ

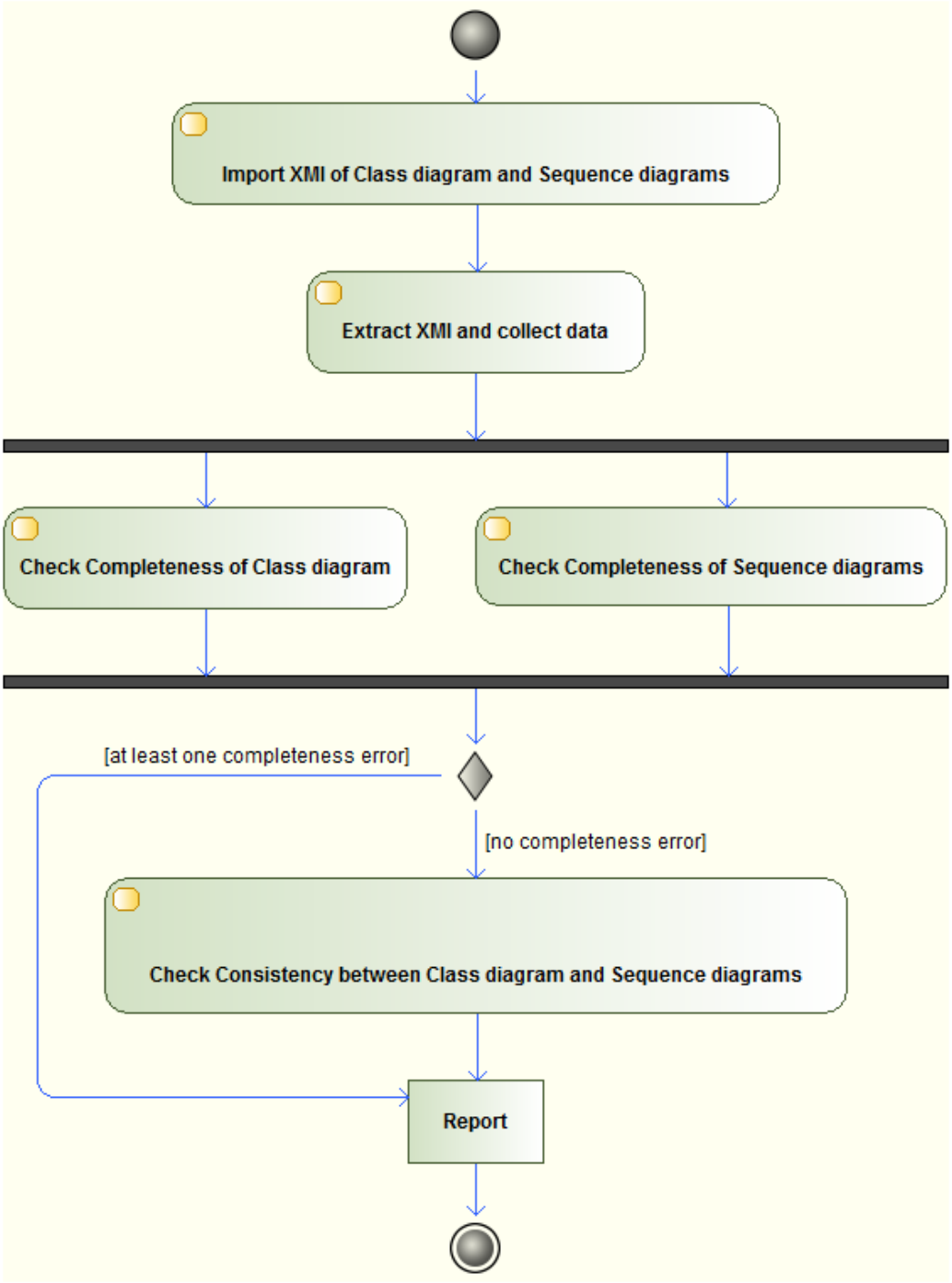
ในบทนี้จะกล่าวถึงการออกแบบและพัฒนาเครื่องมือสำหรับตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ มีรายละเอียดดังต่อไปนี้

4.1 การออกแบบเครื่องมือ

เครื่องมือในการตรวจสอบนี้มีส่วนประกอบหลัก ๆ 3 ส่วนได้แก่

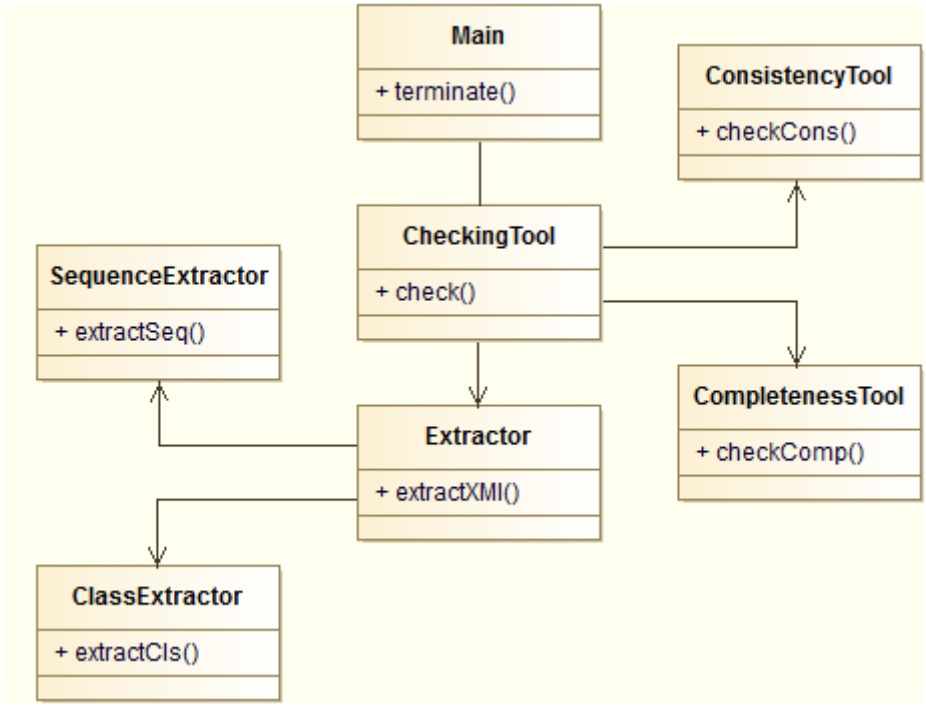
- ส่วนสกัดข้อมูลจากเอ็กซ์เอ็มไอ ทำหน้าที่รับข้อมูลเอ็กซ์เอ็มไอเข้ามาสกัดเอาข้อมูลของแผนภาพคลาสและแผนภาพลำดับ แล้วเก็บรวบรวมข้อมูลส่งให้ส่วนตรวจสอบความพร้อมของข้อมูลและความสอดคล้องต่อไป
- ส่วนตรวจสอบความพร้อมของข้อมูล ทำหน้าที่ตรวจสอบความครบถ้วนสมบูรณ์เชิงโครงสร้างของการออกแบบชั้นนามธรรมที่ต้องการของแผนภาพคลาสและแผนภาพลำดับ และรายงานผลแล้วส่งต่อข้อมูลให้ส่วนตรวจสอบความสอดคล้อง ตลอดจนหยุดการทำงานของเครื่องมือเมื่อตรวจสอบพบความไม่ครบถ้วนสมบูรณ์เชิงโครงสร้างของการออกแบบชั้นนามธรรมที่ต้องการ
- ส่วนตรวจสอบความสอดคล้อง ทำหน้าที่ตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ แล้วรายงานผล

จากส่วนประกอบหลัก 3 ส่วน ผู้วิจัยออกแบบให้เครื่องมือทำงานอย่างต่อเนื่องโดยอัตโนมัติ กล่าวคือ เพียงนำข้อมูลเอ็กซ์เอ็มไอเข้ามา เครื่องมือก็จะสกัดข้อมูลของแผนภาพทั้งสอง ตรวจสอบความครบถ้วนสมบูรณ์ของแผนภาพทั้งสอง ถ้าไม่ผ่านจะรายงานผลและหยุดเครื่องมือ ถ้าผ่านจะตรวจสอบความสอดคล้องระหว่างแผนภาพทั้งสองต่อ และรายงานผลออกมาอย่างอัตโนมัติตามลำดับ ดังรูปที่ 4.1



รูปที่ 4.1 แผนภาพกิจกรรมของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ

เมื่อพิจารณาจากส่วนประกอบหลักทั้ง 3 ออกแบบแผนภาพคลาสของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับได้ดังรูปที่ 4.2



รูปที่ 4.2 แผนภาพคลาสของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ

จากรูปที่ 4.2 แผนภาพคลาสประกอบได้ด้วยคลาสทั้งหมด 7 คลาสได้แก่

- คลาส Main เป็นคลาสหลักของเครื่องมือ มีหน้าที่เริ่มต้นการทำงาน และหยุดทำงานเมื่อตรวจพบว่าระดับข้อมูลของแผนภาพไม่เพียงพอจะตรวจสอบความสอดคล้อง มีความสัมพันธ์แบบแอสโซซิเอชันกับคลาส CheckingTool

- คลาส CheckingTool ตัวกลาง มีหน้าที่เรียกส่วนทำงานหลัก 3 ส่วนให้ทำงาน มีความสัมพันธ์แบบแอซโซซิเอชันกับ 3 คลาสทำงานหลัก

- คลาส Extractor เป็นคลาสทำงานหลัก มีหน้าที่สกัดข้อมูลจากไฟล์เอ็กซ์เอ็มไอ โดยเรียกใช้คลาสย่อย 2 คลาส

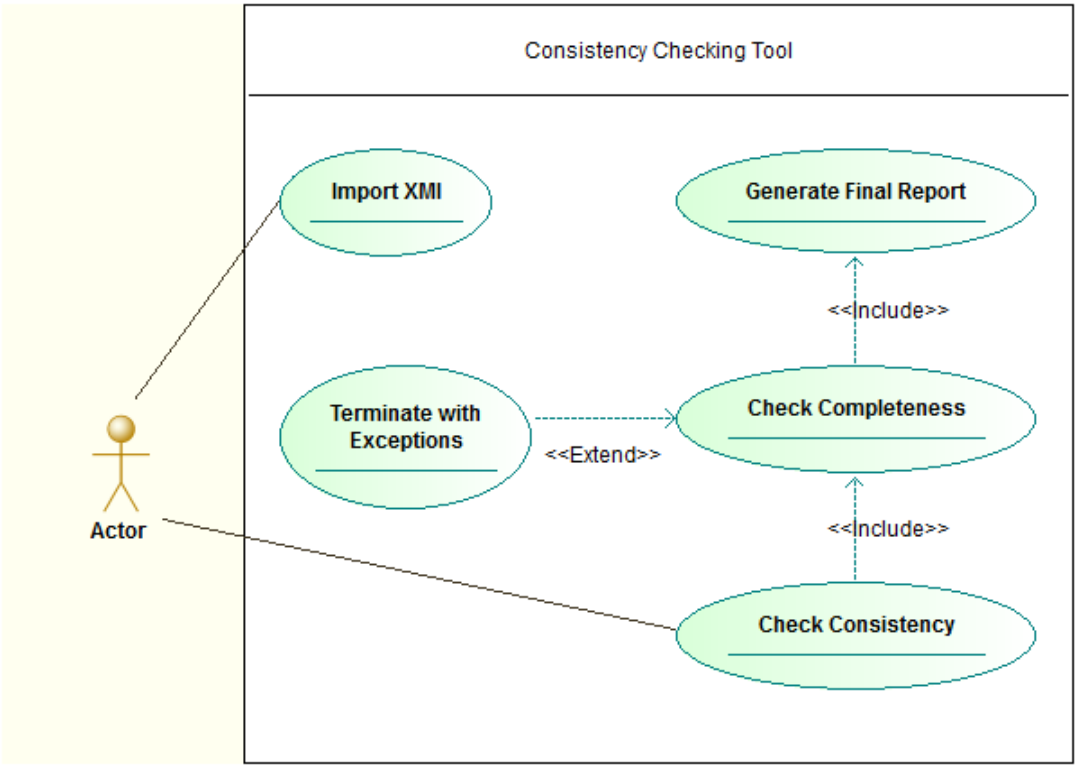
- คลาส ClassExtractor มีหน้าที่สกัดข้อมูลแผนภาพคลาสและส่งข้อมูลคืนให้คลาส Extractor

- คลาส SequenceExtractor มีหน้าที่สกัดข้อมูลแผนภาพลำดับและส่งข้อมูลคืนให้คลาส Extractor

- คลาส CompletenessTool มีหน้าที่ตรวจสอบความครบถ้วนสมบูรณ์เชิงโครงสร้างของแผนภาพคลาสและแผนภาพลำดับ โดยใช้ข้อมูลที่ Extractor สกัดออกมา

- คลาส ConsistencyTool มีหน้าที่ตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ โดยใช้ข้อมูลที่ Extractor สกัดออกมา

เนื่องจากการออกแบบให้เครื่องมือทำงานอย่างต่อเนื่องโดยอัตโนมัติ ผู้ใช้จึงทำหน้าที่เพียงป้อนข้อมูลเอ็กซ์เอ็มไอเข้าเครื่องมือตรวจสอบ แล้วเครื่องมือจะทำงานจนเองจนแสดงผลออกมาดังแผนภาพยูสเคสรูปที่ 4.3

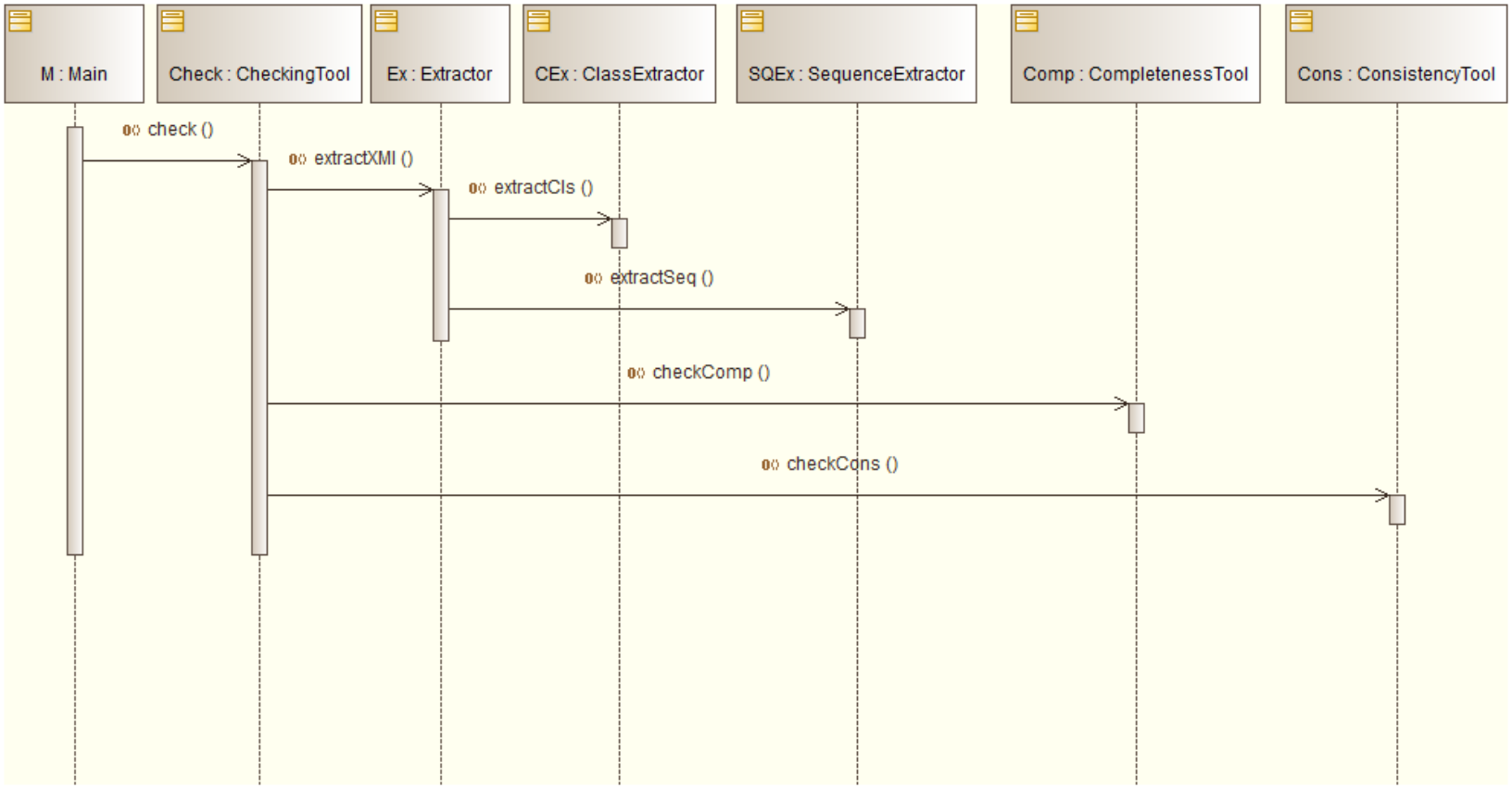


รูปที่ 4.3 แผนภาพยูสเคสของเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ

จากรูปที่ 4.3 แผนภาพยูสเคสมียูสเคส 5 ยูสเคส ได้แก่ Import XMI, Check Consistency, Check Completeness, Terminate with Exceptions และ Generate Final Report โดยผู้ใช้จะนำเข้าไฟล์เอ็กซ์เอ็มไอ เพื่อให้ได้ผลการตรวจสอบความสอดคล้องของแผนภาพคลาสและแผนภาพลำดับทั้งหมดในไฟล์เอ็กซีเอ็มไอนั้น โดยก่อนจะตรวจสอบความสอดคล้องระหว่างแผนภาพต่าง ๆ เครื่องมือจะตรวจสอบความครบถ้วนสมบูรณ์เชิงโครงสร้างเพื่อดูความพร้อมของ

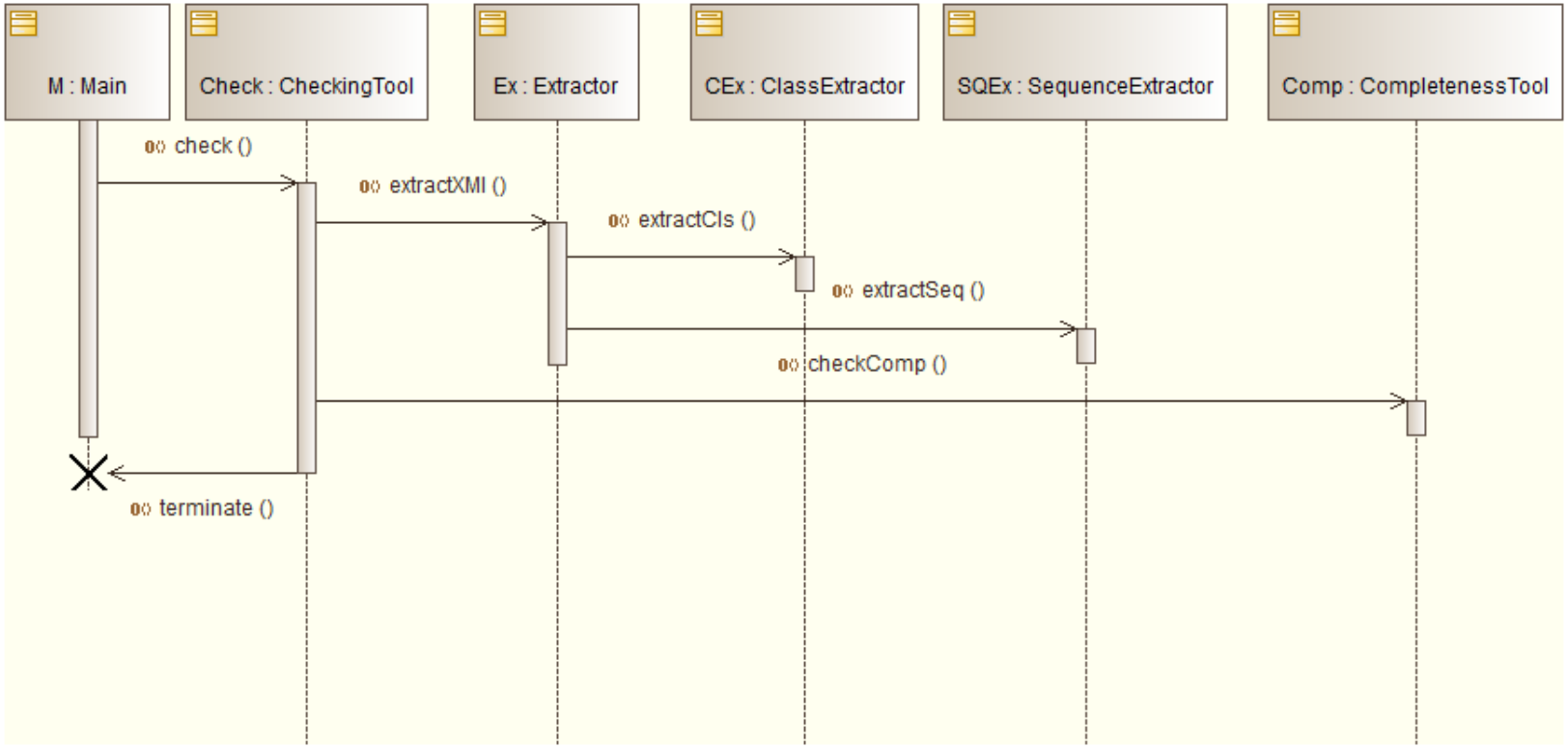
ข้อมูลก่อน ถ้าแผนภาพใดแผนภาพหนึ่งไม่มีความพร้อมของข้อมูล เครื่องมือจะหยุดตัวเอง ถ้าแผนภาพมีความพร้อมจะตรวจสอบความสอดคล้องและสร้างรายงานต่อไป

จากยูสเคสที่ออกแบบไว้ เหตุการณ์ที่จะเกิดขึ้นมี 2 เหตุการณ์คือการตรวจสอบพบว่าข้อมูลมีความพร้อมสำหรับการตรวจสอบความสอดคล้อง และข้อมูลไม่มีความพร้อมสำหรับตรวจสอบความสอดคล้อง ซึ่งนำมาออกแบบแผนภาพลำดับได้ดังรูปที่ 4.4 และรูปที่ 4.5 ตามลำดับ



รูปที่ 4.4 แผนภาพลำดับของเครื่องมือเมื่อข้อมูลมีความพร้อมสำหรับตรวจสอบความสอดคล้อง

จากรูปที่ 4.3 มีส่วนประกอบของระบบทั้งหมด 7 ส่วน ได้แก่ Main, CheckingTool, Extractor, ClassExtractor, SequenceExtractor, CompletenessTool, และ ConsistencyTool เริ่มจาก Main ส่งข้อความเรียกใช้เมธอด check() เพื่อเริ่มการทำงาน CheckingTool จะส่งข้อความเรียกเมธอด extractXMI ไปยัง Extractor เพื่อให้ Extractor สั่ง ClassExtractor และ SequenceExtractor ให้สกัดข้อมูลแผนภาพคลาสและแผนภาพลำดับตามลำดับ แล้วส่งข้อมูลที่สกัดได้กลับมายัง Extractor และส่งต่อไปยัง Checking อีกต่อหนึ่ง ต่อมาเมื่อได้ข้อมูลของแผนภาพคลาสและแผนภาพลำดับทั้งหมดแล้ว Checking ก็ส่งข้อความเรียกใช้เมธอด checkComp ไปตรวจสอบความพร้อมของข้อมูลของแผนภาพทั้งสอง แล้วส่งรายงานกลับมา เมื่อพบว่าแผนภาพทั้งหมดมีความพร้อมสำหรับตรวจสอบความสอดคล้อง Checking จะะส่งข้อความเรียกใช้เมธอด checkCons ไปตรวจสอบความสอดคล้องและรายงานผลต่อไป ซึ่งรายงานทั้งหมดก็จะถูกส่งกลับมายัง Main เพื่อแสดงผลต่อไป



รูปที่ 4.5 แผนภาพลำดับของเครื่องมือเมื่อข้อมูลไม่มีความพร้อมสำหรับตรวจสอบความสอดคล้อง

จากรูปที่ 4.4 มีส่วนประกอบของระบบทั้งหมด 6 ส่วน ได้แก่ Main, CheckingTool, Extractor, ClassExtractor, SequenceExtractor, และ CompletenessTool เริ่มจาก Main ส่งข้อความเรียกใช้เมธอด check() เพื่อเริ่มการทำงาน CheckingTool จะส่งข้อความเรียกเมธอด extractXMI ไปยัง Extractor เพื่อให้ Extractor สั่ง ClassExtractor และ SequenceExtractor ให้สกัดข้อมูลแผนภาพคลาสและแผนภาพลำดับตามลำดับ แล้วส่งข้อมูลที่สกัดได้กลับมายัง Extractor และส่งต่อไปยัง Checking อีกต่อหนึ่ง ต่อมาเมื่อได้ข้อมูลของแผนภาพคลาสและแผนภาพลำดับทั้งหมดแล้ว Checking ก็ส่งข้อความเรียกใช้เมธอด checkComp ไปตรวจสอบความพร้อมของข้อมูลของแผนภาพทั้งสอง แล้วส่งรายงานกลับมา เมื่อพบว่าแผนภาพทั้งหมดไม่มีความพร้อมสำหรับตรวจสอบความสอดคล้อง Checking จะส่งข้อความเรียกใช้เมธอด terminate() ไปทำลาย Main เพื่อหยุดการทำงานของเครื่องมือ

4.2 การพัฒนาเครื่องมือ

เครื่องมือนี้พัฒนาด้วยภาษาไพธอน โดยใช้ Python Shell บนระบบปฏิบัติการณ์วินโดวส์ โดยมีเครื่องคอมพิวเตอร์และซอฟต์แวร์ที่ใช้ดังนี้

- หน่วยประมวลผลกลาง Intel Core 2 Duo 6420 ความเร็ว 2.13 GHz
- หน่วยความจำหลัก 4.00 GB
- ฮาร์ดดิสก์ความจุ 80 GB ความเร็วจานหมุน 7200 RPM
- หน่วยประมวลผลภาพ Nvidia GeForce 8500GT หน่วยความจำ 256MB
- ระบบปฏิบัติการ Windows 7 Professional 64 bit
- Python 3.2.3
- Notepad++ 6.3.2

บทที่ 5

การทดสอบเครื่องมือด้วยกรณีศึกษา

การทดสอบเครื่องมือที่พัฒนาขึ้นด้วยกรณีศึกษา 3 กรณี ได้แก่ ระบบห้องสมุด ระบบเอทีเอ็ม และระบบการลงทะเบียน ผลที่ได้สามารถทดสอบความสอดคล้องของแผนภาพคลาสและแผนภาพลำดับได้ตามขอบเขตของงานวิจัย

5.1 สภาวะที่ใช้ในการทดสอบเครื่องมือ

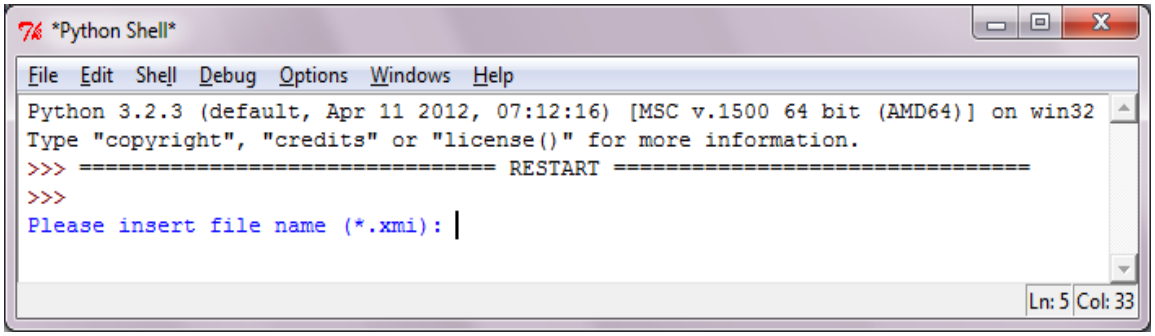
เครื่องคอมพิวเตอร์และระบบที่ใช้ในการทดสอบมีรายละเอียดดังนี้

- หน่วยประมวลผลกลาง Intel Core 2 Duo 6420 ความเร็ว 2.13 GHz
- หน่วยความจำหลัก 4.00 GB
- ฮาร์ดดิสก์ความจุ 80 GB ความเร็วจานหมุน 7200 RPM
- หน่วยประมวลผลภาพ Nvidia GeForce 8500GT หน่วยความจำ 256MB
- ระบบปฏิบัติการ Windows 7 Professional 64 bit
- Python 3.2.3

5.2 ขั้นตอนของการทดสอบ

การทดสอบทำโดยป้อนไฟล์เอ็กซ์เอ็มไอของแผนภาพคลาสและแผนภาพลำดับของกรณีศึกษาทั้ง 3 กรณีเข้าสู่เครื่องมือทีละไฟล์ ซึ่งมีขั้นตอนภายใน 3 ขั้นตอนดังอธิบายไว้ในบทที่ 3 โดยข้อมูลที่ป้อนเข้ามาเป็นข้อมูลแผนภาพที่เป็นไปตามมาตรฐานยูเอ็มแอลรุ่น 2.3 ซึ่งสร้างจากโปรแกรมโมเดลิโอรุ่น 2.2.1 แล้วส่งออกเป็นไฟล์มาตรฐานเอ็กซ์เอ็มแอลรุ่น 2.1

เมื่อเตรียมไฟล์เอ็กซ์เอ็มไอเสร็จแล้ว รันเครื่อมมือตรวจสอบดังรูปที่ 5.1 แล้วจึงใส่ชื่อไฟล์ที่เตรียมไว้ถ้าไฟล์อยู่ในแฟ้มเดียวกันกับโปรแกรม Python แต่ถ้าไฟล์อยู่คนละแฟ้มก็ใส่พาธของไฟล์ด้วย



รูปที่ 5.1 ส่วนการรับข้อความชื่อไฟล์เอ็กซ์เอ็มไอ

เมื่อป้อนชื่อไฟล์เสร็จแล้วเครื่องมือจะอ่านไฟล์เอ็กซ์เอ็มไอที่ระบุและตรวจสอบตามกฎการตรวจสอบความพร้อมของข้อมูลของแผนภาพคลาสและแผนภาพลำดับทั้งหมดและรายงานผลทันที หากแผนภาพมีความพร้อมของข้อมูล เครื่องมือก็จะตรวจสอบความสอดคล้องระหว่างแผนภาพทั้งหมดต่อทันที

5.3 กรณีศึกษาที่ใช้ในการทดสอบ

กรณีศึกษาทั้ง 3 กรณี นำมาจากงานกรณีศึกษาที่ใช้ในงานวิจัยของกณิษฐา บุญคุ้ม [19] โดยเพิ่มรายละเอียดของข้อมูลเรื่องการเข้าถึงลงไป

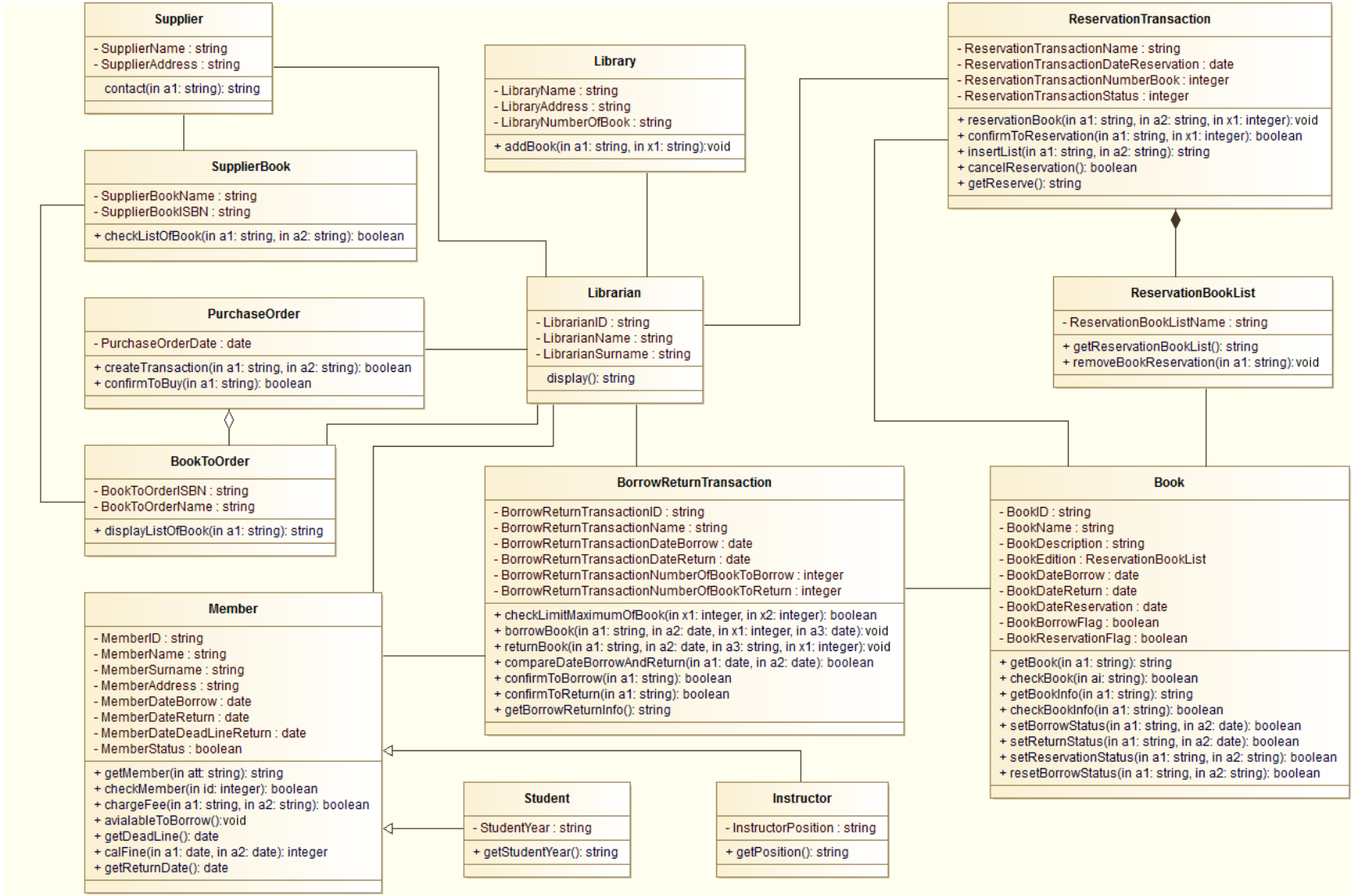
5.3.1 กรณีศึกษาที่ 1 ระบบห้องสมุด

เป็นระบบที่ใช้ในการบริหารงานภายในห้องสมุด โดยห้องสมุดจะมีบรรณารักษ์หนึ่งคนขึ้นไป สมาชิกห้องสมุดจะสามารถใช้บริการยืม คืน และจองหนังสือได้ ส่วนบรรณารักษ์หนึ่งคนจะสามารถทำรายการยืม-คืนให้แก่สมาชิก โดยทำผ่านรายการยืม-คืนหนังสือ และสามารถทำรายการจองให้แก่สมาชิก โดยทำผ่านรายการจองหนังสือ ซึ่งรายการจองหนังสือหนึ่งรายการจะประกอบด้วย รายชื่อและจำนวนของหนังสือที่จะจอง ทั้งนี้บรรณารักษ์สามารถตรวจสอบหนังสือที่ถูกยืมออกไปจากรายการยืม-คืนหนังสือ ส่วนสมาชิกสามารถดูหนังสือได้เช่นกันว่าหนังสือว่างหรือสามารถยืมได้หรือไม่ รวมทั้งบรรณารักษ์ยังยกเลิกการจองของสมาชิกได้ นอกจากนี้บรรณารักษ์ยังทำข้อตกลงซื้อหนังสือกับผู้ขายได้อีกด้วย

5.3.1.1 แผนภาพคลาสและแผนภาพลำดับของระบบห้องสมุด

1. แผนภาพคลาส

แผนภาพคลาสของระบบห้องสมุดประกอบด้วยคลาส 13 คลาส มีคลาสที่มีบทบาทหลัก ๆ เช่น Library, Librarian, Member, BorrowReturnTransaction และ ReservationTransaction ดังรูปที่ 5.2

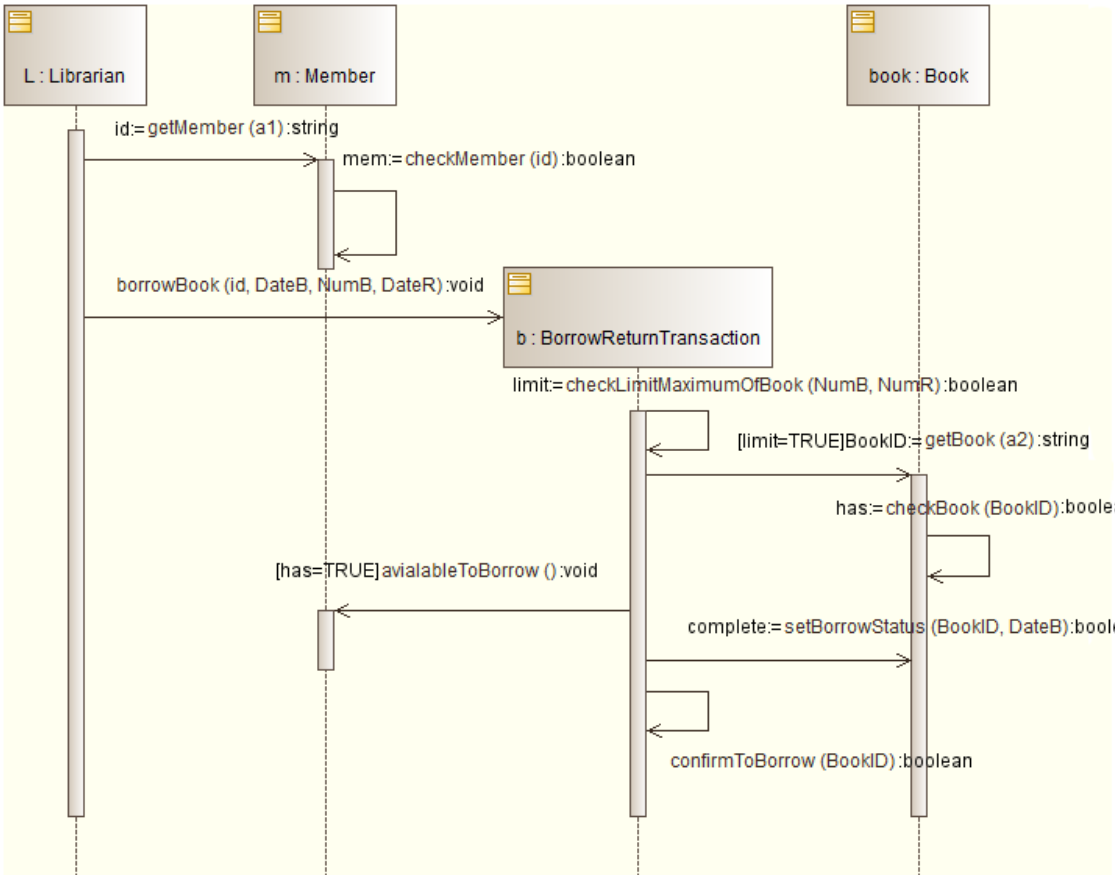


รูปที่ 5.2 แผนภาพคลาสของระบบห้องสมุด

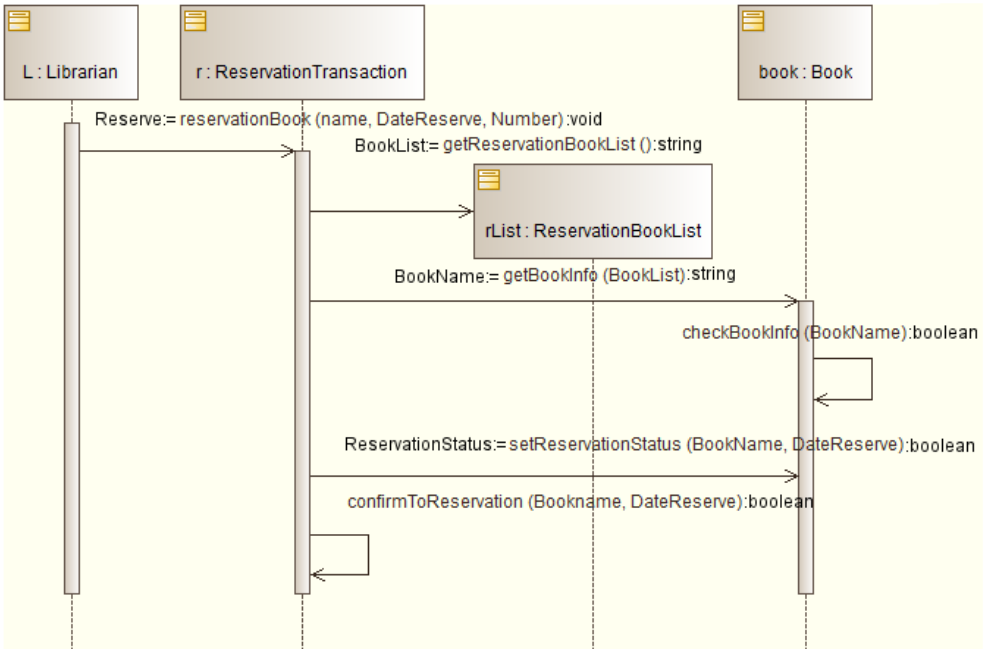
2. แผนภาพลำดับ

แผนภาพลำดับแสดงเหตุการณ์ที่เกิดขึ้นในระบบห้องสมุด 7 เหตุการณ์ แสดงดังรูปที่ 5.3 ถึง 5.9 ตามลำดับดังนี้

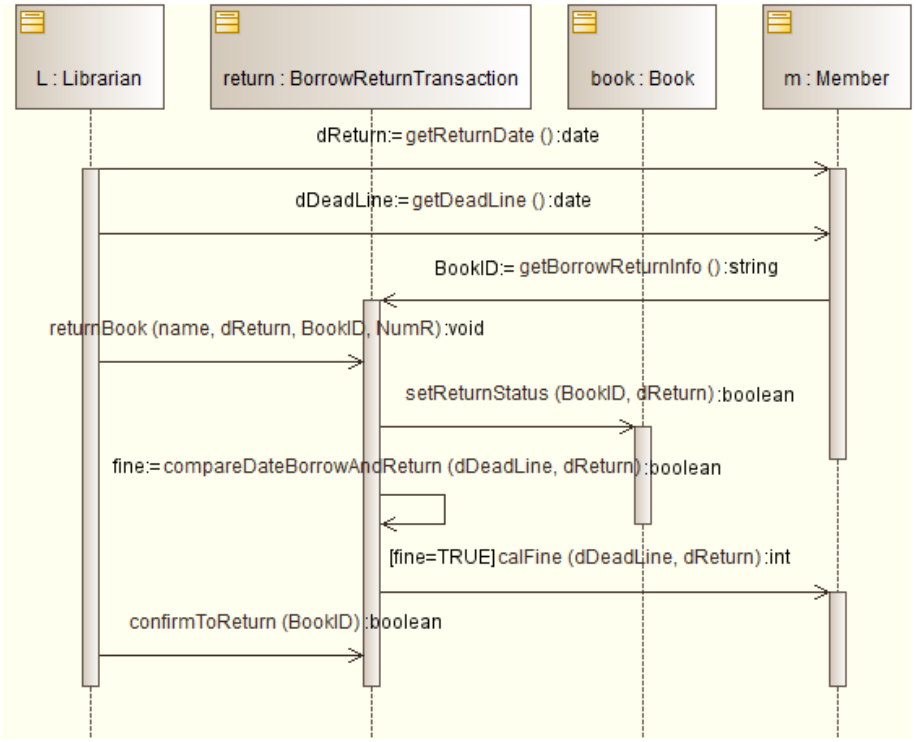
- แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือ
- แผนภาพลำดับแสดงเหตุการณ์จองหนังสือ
- แผนภาพลำดับแสดงเหตุการณ์คืนหนังสือ
- แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือที่จองไว้
- แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบหนังสือที่ค้างส่ง
- แผนภาพลำดับแสดงเหตุการณ์ยกเลิกหนังสือที่จอง
- แผนภาพลำดับแสดงเหตุการณ์สั่งซื้อหนังสือ



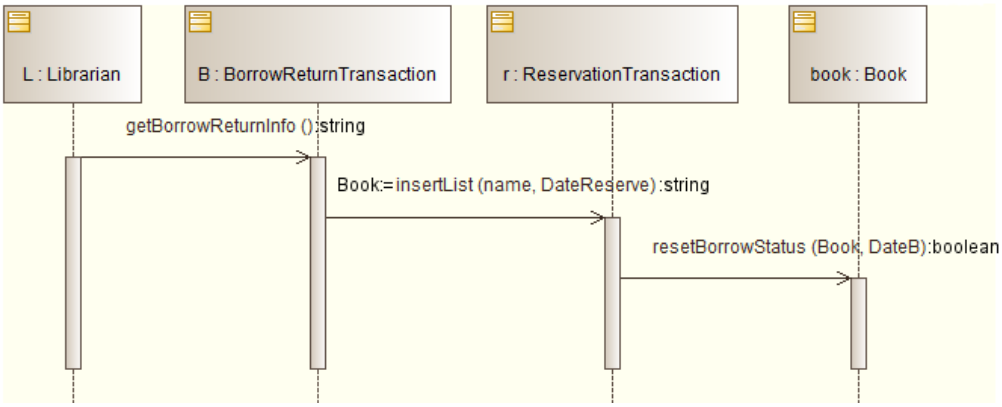
รูปที่ 5.3 แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือ



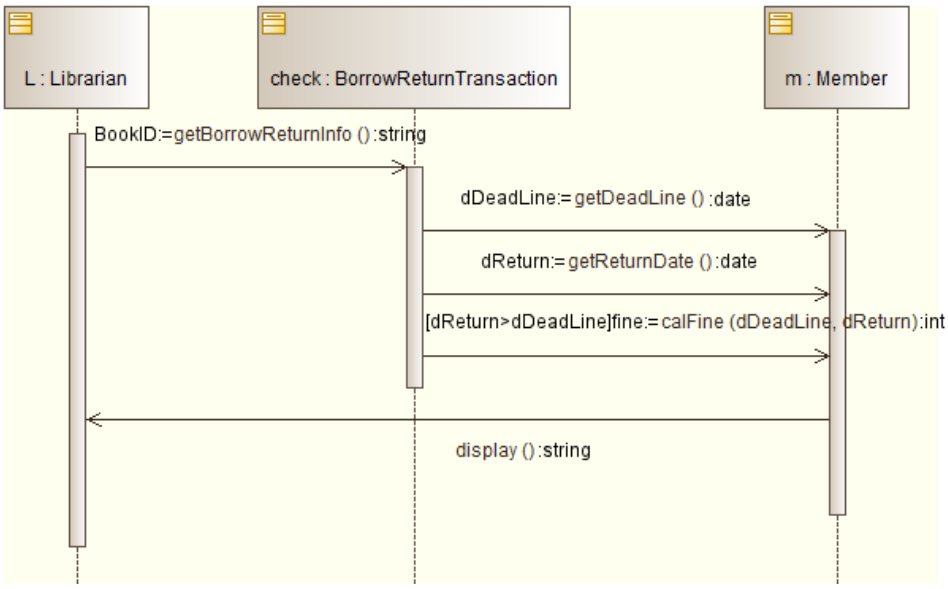
รูปที่ 5.4 แผนภาพลำดับแสดงเหตุการณ์จองหนังสือ



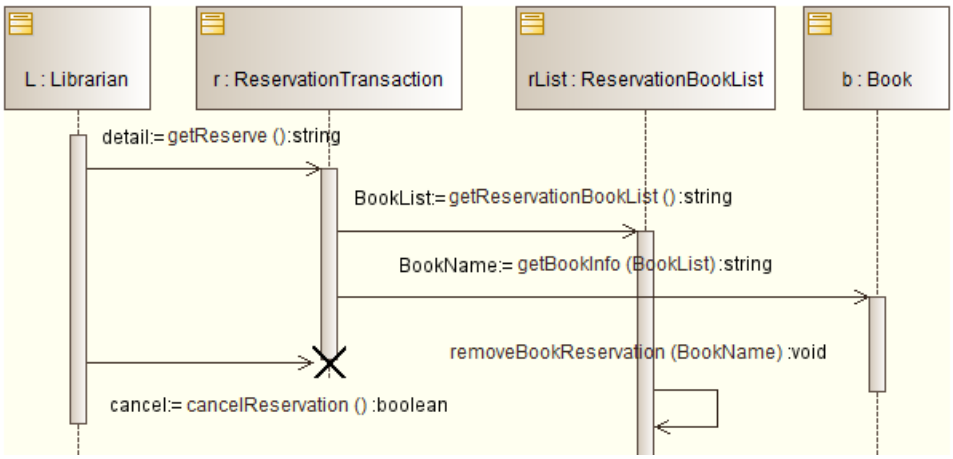
รูปที่ 5.5 แผนภาพลำดับแสดงเหตุการณ์คืนหนังสือ



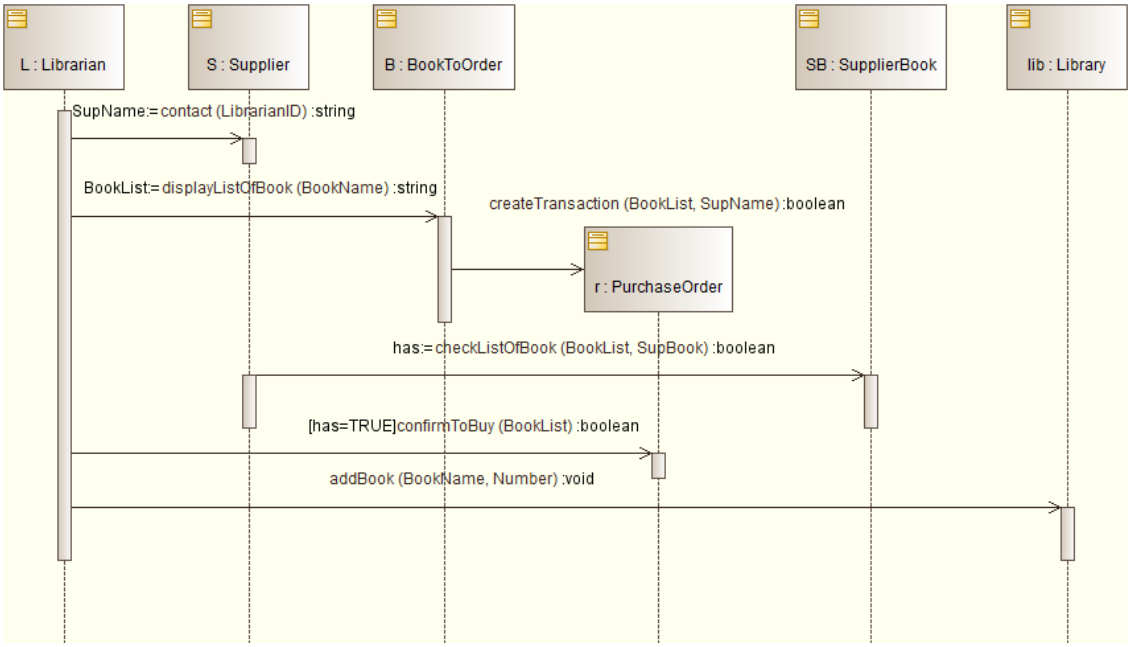
รูปที่ 5.6 แผนภาพลำดับแสดงเหตุการณ์ยืมหนังสือที่จองไว้



รูปที่ 5.7 แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบหนังสือที่ค้างส่ง



รูปที่ 5.8 แผนภาพลำดับแสดงเหตุการณ์ยกเลิกหนังสือที่จอง



รูปที่ 5.9 แผนภาพลำดับแสดงเหตุการณ์สั่งซื้อหนังสือ

5.3.1.2 ผลการตรวจสอบความครบถ้วนสมบูรณ์

1. แผนภาพคลาส

เครื่องมือตรวจพบความไม่พร้อมของข้อมูลสองจุด คือเมธอด contact ของคลาส Supplier และเมธอด display ของคลาส Librarian ไม่ได้ระบุการเข้าถึงไว้ แล้วหยุดตัวเองดังรูปที่ 5.10

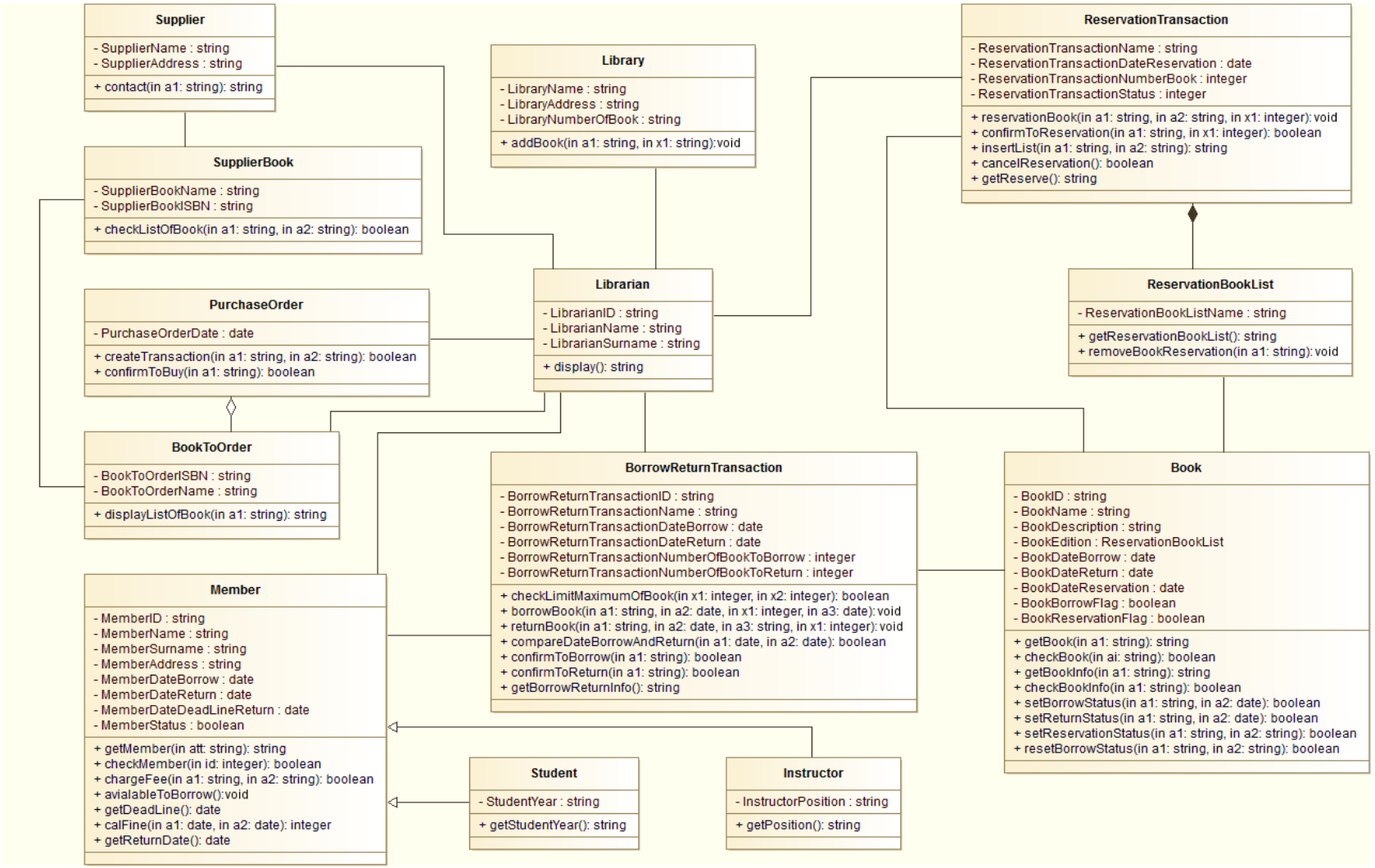
```
Please insert file name (*.xmi): Simple Library.xmi
Entering Readieness Checking Module
.
..
...
....
.....
Checking result...
<--There's  2 Class Methods errors which have no given Method or Visibility.
----Class ID: < _aexMi4OxEeKK4eHVRkUeZw > Class name: < Supplier > Meth ID: < _aex
Mm4OxEeKK4eHVRkUeZw > Meth Name: < contact > doesn't have visibility.
----Class ID: < _aezpAYOxEeKK4eHVRkUeZw > Class name: < Librarian > Meth ID: < _ae
zpF4OxEeKK4eHVRkUeZw > Meth Name: < display > doesn't have visibility.

### Readiness checking not pass. Please fix problem before checking consistency be
tween diagrams. ###
>> You have 2 Class diagram Readiness problems to fix.
---Program will be terminated---
Traceback (most recent call last):
  File "C:\Python32\Checking.py", line 384, in <module>
    exit('Readiness problem')
  File "C:\Python32\lib\site.py", line 362, in __call__
    raise SystemExit(code)
SystemExit: Readiness problem
>>>
```

รูปที่ 5.10 เครื่องมือแจ้งรายละเอียดของความไม่พร้อมของข้อมูลแล้วหยุดตัวเอง

เมื่อแก้ไขแผนภาพคลาสของระบบโดยระบุการเข้าถึงให้ครบแล้ว ตามรูปที่ 5.11 สร้างไฟล์เอ็กซ์เอ็มไอให้เครื่องมือตรวจสอบอีกครั้งไม่พบความไม่พร้อมของข้อมูลของแผนภาพคลาสของระบบห้องสมุด เครื่องมือตรวจสอบสรุปรายละเอียดของแผนภาพคลาสออกมาได้ดังนี้

- มีคลาสทั้งหมด 13 คลาส
- มีคุณลักษณะทั้งหมด 43 คุณลักษณะ
- มีเมธอดทั้งหมด 38 เมธอด
- มีเส้นความสัมพันธ์กันทั้งหมด 17 เส้น
- มีประเภทของตัวแปรทั้งหมด 6 ประเภท



รูปที่ 5.11 แผนภาพคลาสของระบบห้องสมุดที่ได้รับการแก้ไขการเข้าถึงแล้ว

2. แผนภาพลำดับ

ไม่พบความไม่พร้อมของข้อมูลของแผนภาพลำดับของระบบห้องสมุดใด ๆ เครื่องมือตรวจสอบจึงสรุปรายละเอียดของแผนภาพลำดับออกมาได้ดังนี้

- มีแผนภาพลำดับทั้งหมด 7 แผนภาพ
- มีส่วนประกอบของระบบทั้งหมด 29 ส่วนประกอบ
- มีข้อความทั้งหมด 42 ข้อความ

5.3.1.3 ผลการตรวจสอบความสอดคล้อง

ตรวจสอบด้วยเครื่องมือแล้ว รายงานผลแจ้งว่าเกิดความไม่สอดคล้องขึ้น 6 จุด ซึ่งขัดกับกฎข้อ 2 3 4 และ 5 ดังรูปที่ 5.12 ดังมีรายละเอียดดังนี้

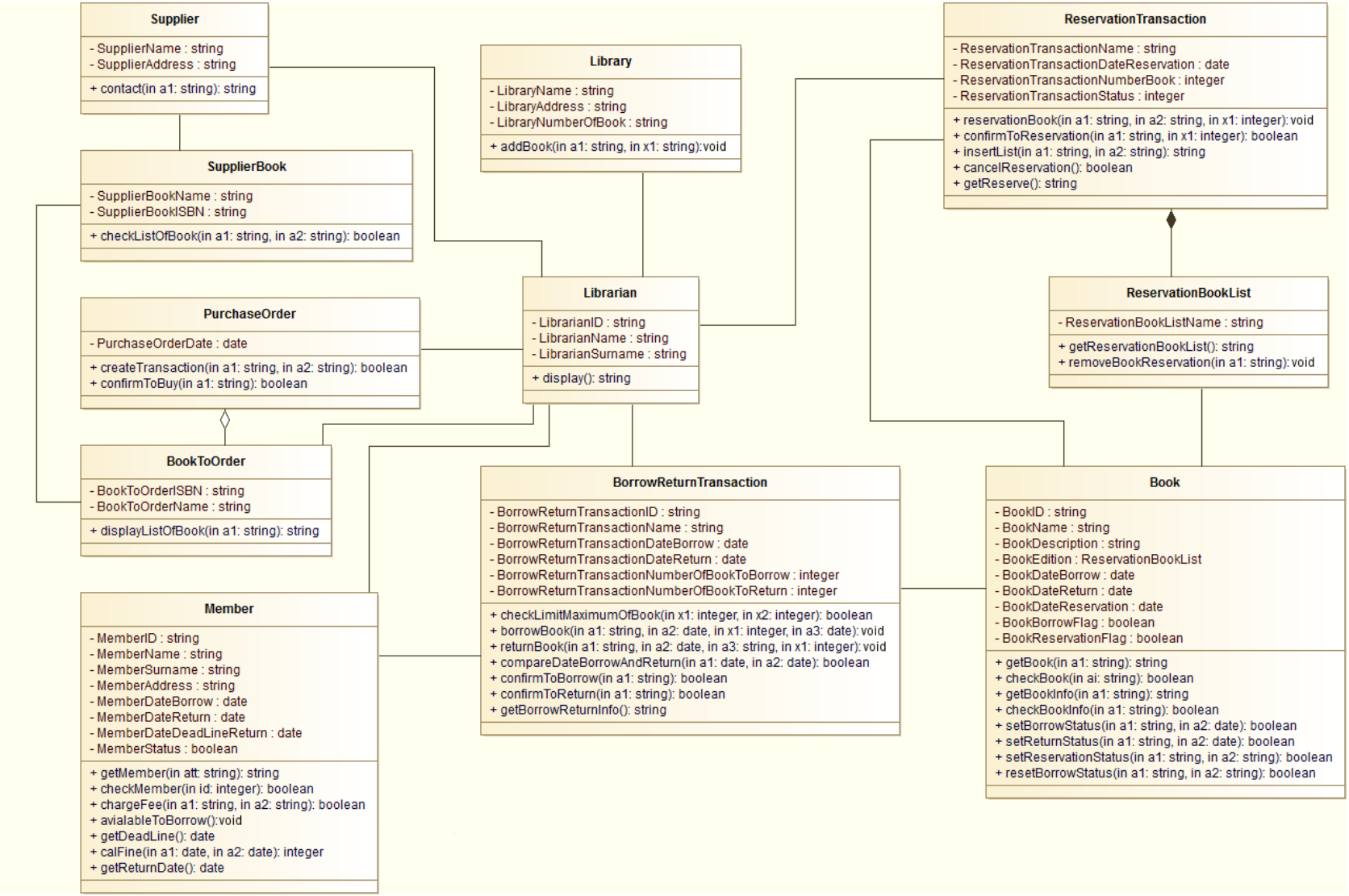
1. เมธอด getStudentYear ของคลาส Student ไม่ถูกเรียกใช้ในแผนภาพลำดับใด ๆ เลย
2. เมธอด getPosition ของคลาส Instructor ไม่ถูกเรียกใช้ในแผนภาพลำดับใด ๆ เลย
3. คลาส Student ไม่ถูกเรียกใช้ในแผนภาพลำดับใด ๆ เลย
4. คลาส Instructor ไม่ถูกเรียกใช้ในแผนภาพลำดับใด ๆ เลย
5. ข้อความ insertList ในแผนภาพลำดับเหตุการณ์ยืมหนังสือที่จองไว้ เป็นข้อความที่รับ-ส่งกับคลาสที่ไม่มีความสัมพันธ์กันในแผนภาพคลาส
6. ข้อความ createTransaction ในแผนภาพลำดับเหตุการณ์สั่งซื้อหนังสือ ส่งข้อความแบบสร้างไปสร้างคลาสแม่ที่มีความสัมพันธ์กันแบบแอกกรีเกรชัน

```
---Entering Consistency Checking Module---
.
..
...
....
.....
Rule 1: all participants in Sequence Diagrams must be valid as class in Class Diagram
---pass---
Rule 2: all methods in Class Diagram must be used
--not pass: There're Method(s) that don't be used.
<--error code: Class Name: < Student > Method ID: < _ae027YOxEeKK4eHVRkUeZw > Method Name: <
getStudentYear >
<--error code: Class Name: < Instructor > Method ID: < _ae03AIOxEeKK4eHVRkUeZw > Method Name:
< getPosition >
Rule 3: all classes in Class Diagram must be used as participants in Sequence Diagrams
--not pass: There're Class(es) that don't be used.
<--error code: Class ID: < _ae024oOxEeKK4eHVRkUeZw > Class name: < Student >
<--error code: Class ID: < _ae029YOxEeKK4eHVRkUeZw > Class name: < Instructor >
Rule 4: relation in Sequence Diagrams must be valid in Class Diagram
--not pass: There're Relation(s) between Actor or Paticipant that don't appear in Class Diagr
am.
<--error code: SQD Name: < Borrow Reserved Book > Message ID: < _ae3Tl4OxEeKK4eHVRkUeZw > Mes
sage name: < insertList >
Rule 5: subclass must not create motherclass if their relationship is aggregation
--not pass: There're subclass(es) sending 'CreateMessage' to motherclass(es). [Aggregation]
<--error code: SQD Name: < Order Book > Message ID: < _ae36rIOxEeKK4eHVRkUeZw > Message Name:
< createTransaction >
Rule 6: subclass must not create motherclass if their relationship is composition
---pass---
Rule 7: subclass must not destroy motherclass if their relationship is composition
---pass---
Rule 8: after provider was destroyed, user cannot use it's services
---pass---
Rule 9: after motherclass was destroyed, subclass must be destroyed too. If their relationshi
p is composition
---pass---
Rule 10: relationship between participants in Sequence Diagram must be valid and have right i
n Class Diagram
---pass--- : all messages is valid
---pass--- : all messages send in right direction
---pass--- : all messages sent have right to do so
Rule 11: variables in messages must match in methods
---pass---
Rule 12: variables in guard condition must match in attributes or grobal variables
---pass---
Rule 13: there must not be recursion between Sequence Diagrams
---pass---
You still have 6 consistency error(s) to correct.
>>>
```

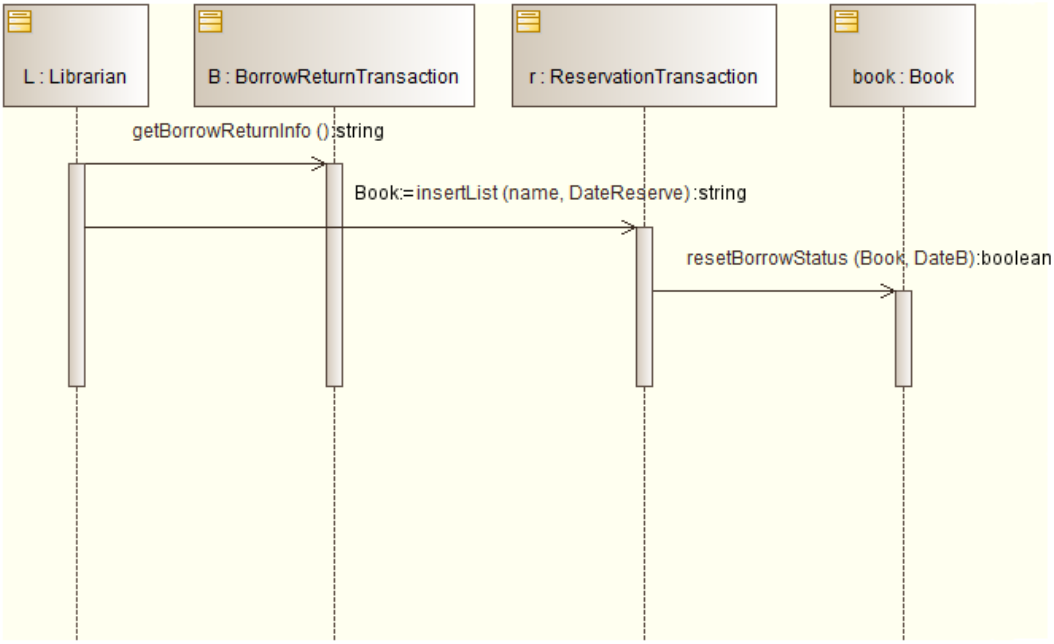
รูปที่ 5.12 รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบห้องสมุด

แก้ไขความไม่สอดคล้องทั้ง 6 จุดดังนี้

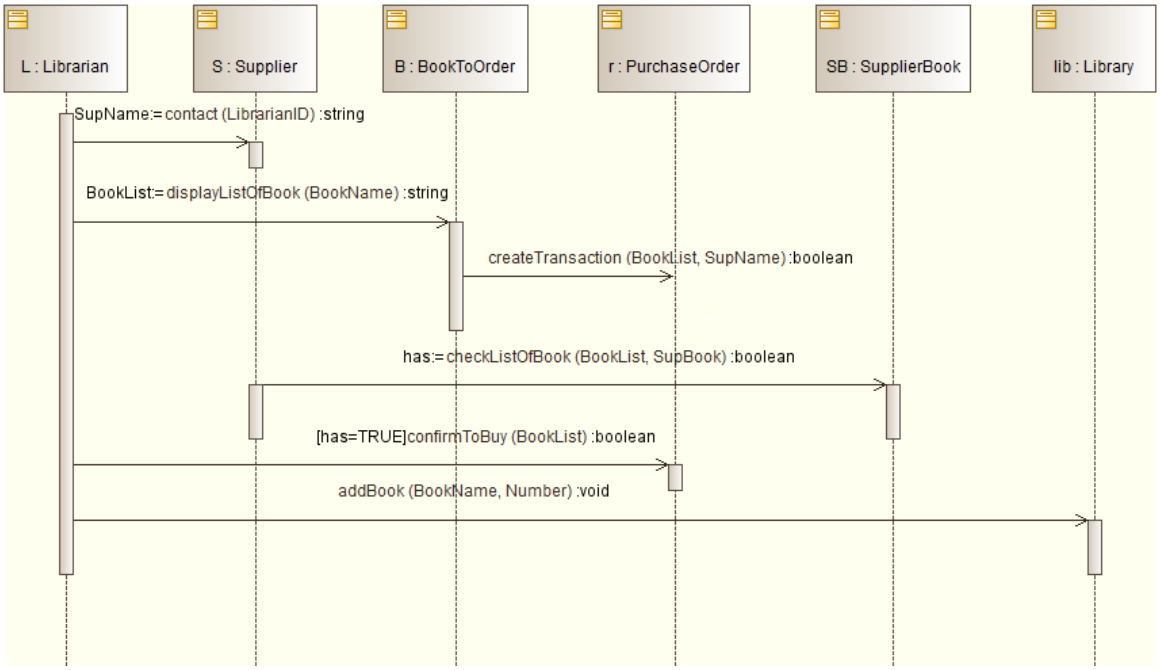
1. ตัดคลาส Student และ Instructor ทิ้ง เนื่องจากไม่มีการเรียกใช้ ดังรูปที่ 5.13
2. ในแผนภาพลำดับเหตุการณ์ยืมหนังสือที่จองไว้ เปลี่ยนให้ Librarian ซึ่งมีความสัมพันธ์กับ reservationTransaction เป็นผู้ส่งข้อความ insertList แทน ดังรูปที่ 5.14
3. ในแผนภาพลำดับเหตุการณ์สั่งซื้อหนังสือ ให้ BookToOrder ส่งข้อความ creatTransaction แบบธรรมดาแทนที่จะเป็นแบบสร้าง ดังรูปที่ 5.15



รูปที่ 5.13 แผนภาพคลาสของระบบห้องสมุดที่ลบคลาสที่ไม่ได้ใช้ออกแล้ว



รูปที่ 5.14 แผนภาพลำดับเหตุการณ์ยืมหนังสือที่จองไว้ที่แก้ไขแล้ว



รูปที่ 5.15 แผนภาพลำดับเหตุการณ์สั่งซื้อหนังสือที่แก้ไขแล้ว

5.3.2 กรณีศึกษาที่ 2 ระบบเอทีเอ็ม

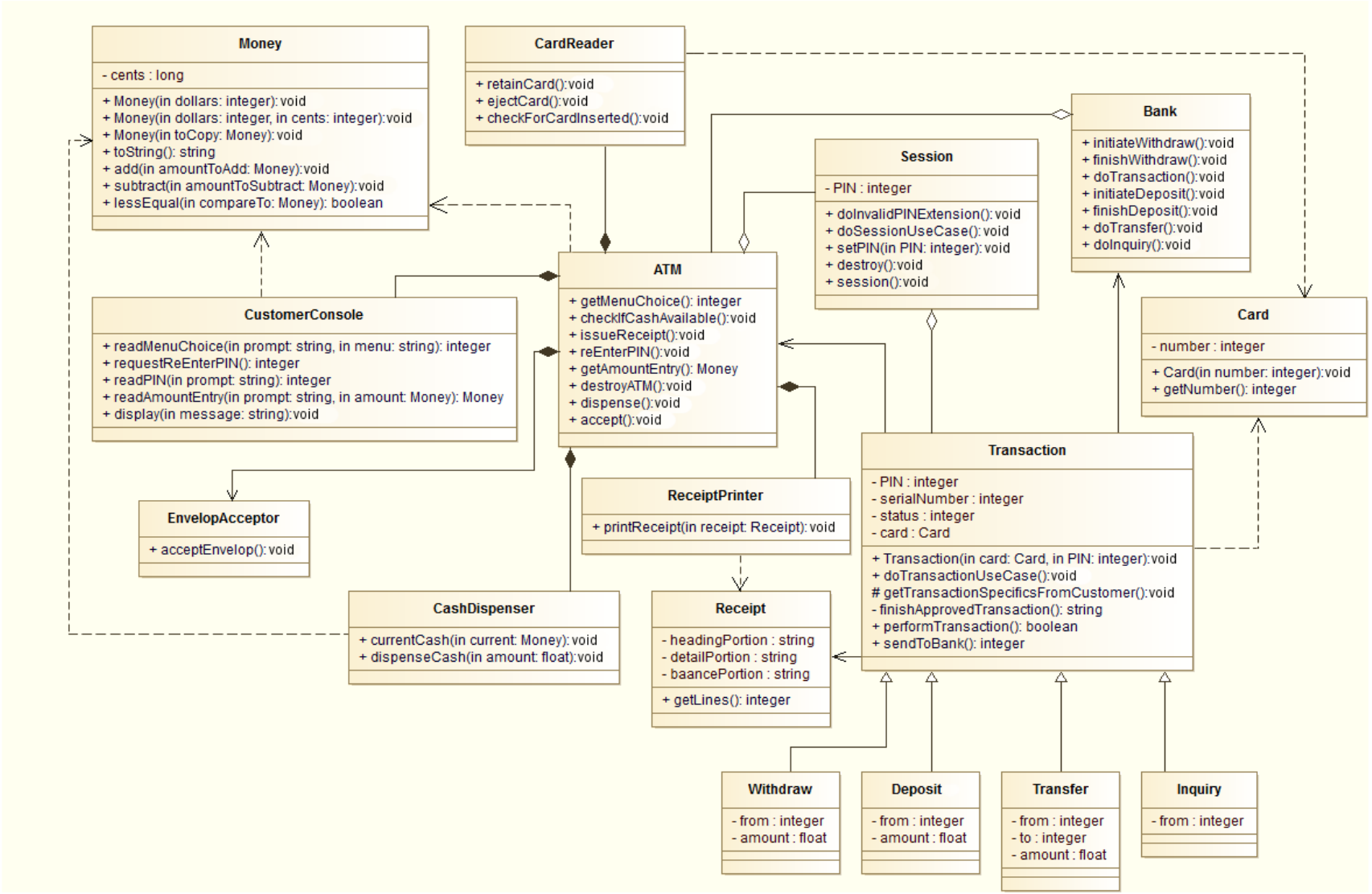
เป็นระบบสำหรับให้ผู้ใช้ทำรายการถอนเงิน ฝากเงิน โอนเงิน และตรวจสอบยอดเงินคงเหลือที่เครื่องเอทีเอ็มได้โดยไม่จำเป็นต้องไปทำรายการที่ธนาคาร โดยธนาคารหนึ่ง ๆ สามารถมีตู้เอทีเอ็มได้หลายตู้ ส่วนตู้เอทีเอ็มหนึ่งตู้ประกอบไปด้วย ช่องใส่บัตร ส่วนติดต่อผู้ใช้หรือแป้นควบคุมการทำรายการ ช่องจ่ายเงิน ช่องฝากเงิน และช่องพิมพ์ใบรายการ

เริ่มต้นการทำงานโดยผู้ใช้สอดบัตรเอทีเอ็มในช่อง จากนั้นกดรหัส ถ้ารหัสไม่ถูกต้องจะต้องใส่ใหม่ได้ 3 ครั้ง มิฉะนั้นจะถูกยึดบัตร เมื่อรหัสถูกต้องก็จะสามารถทำรายการต่าง ๆ ได้ ได้แก่ ถอนเงิน ฝากเงิน โอนเงิน และดูยอดเงินคงเหลือ หลังจากทำรายการเสร็จก็จะพิมพ์ใบรายการที่ทำไว้ จากนั้นก็จะคืนบัตรและออกจากระบบตามลำดับ

5.3.2.1 แผนภาพคลาสและแผนภาพลำดับของระบบเอทีเอ็ม

1. แผนภาพคลาส

แผนภาพคลาสของระบบเอทีเอ็มประกอบด้วยคลาส 16 คลาส มีคลาสที่มีบทบาทหลัก ๆ เช่น ATM Bank Transaction CardReader และ Receipt ดังรูปที่ 5.16

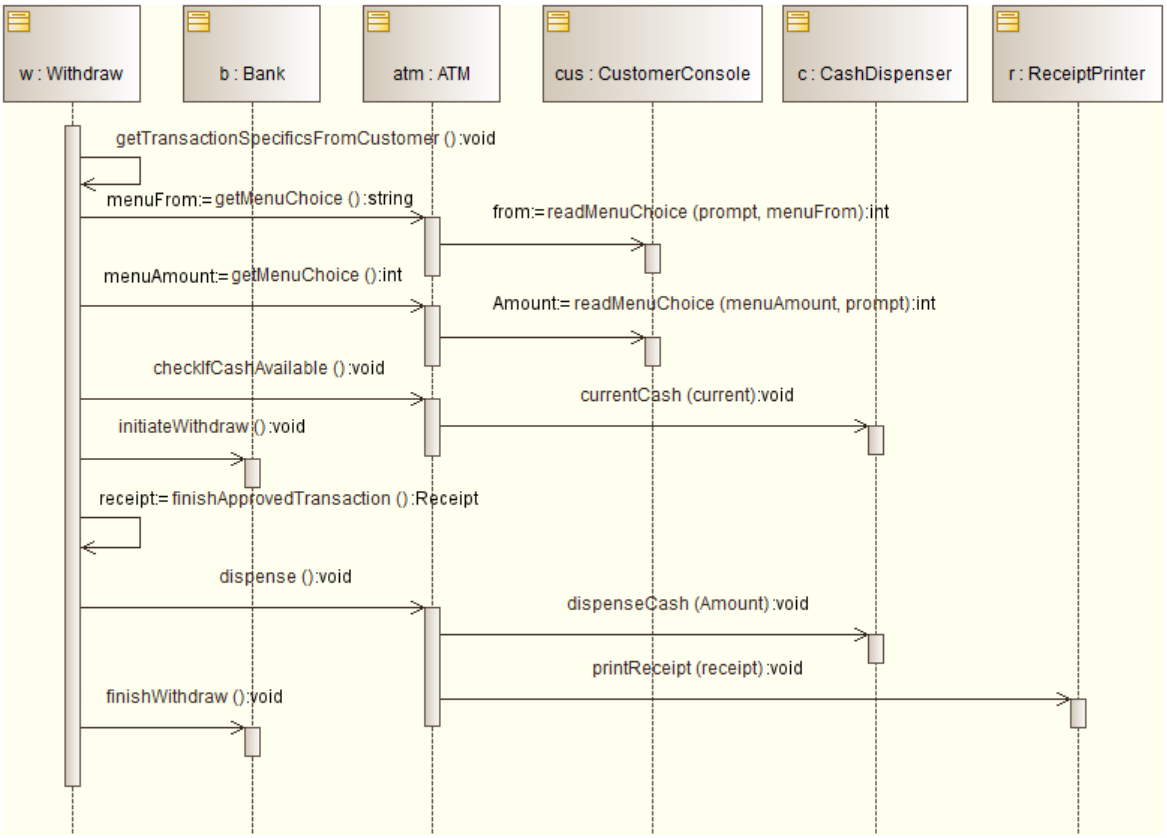


รูปที่ 5.16 แผนภาพคลาสของระบบเอทีเอ็ม

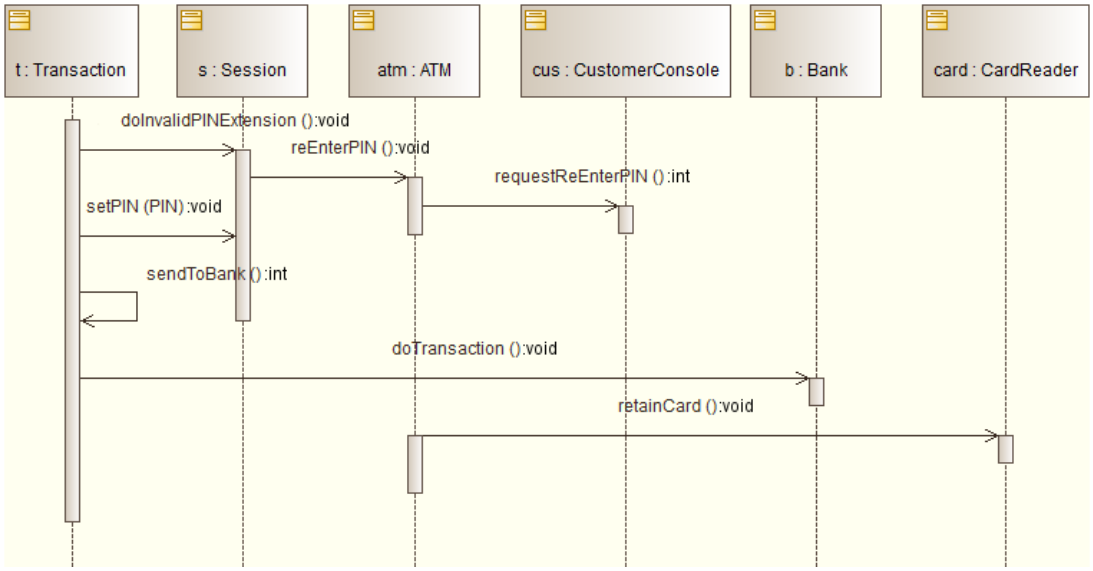
2. แผนภาพลำดับ

แผนภาพลำดับแสดงเหตุการณ์ที่เกิดขึ้นในระบบเอทีเอ็ม 6 เหตุการณ์ แสดงดังรูปที่ 5.17 ถึง 5.22 ตามลำดับดังนี้

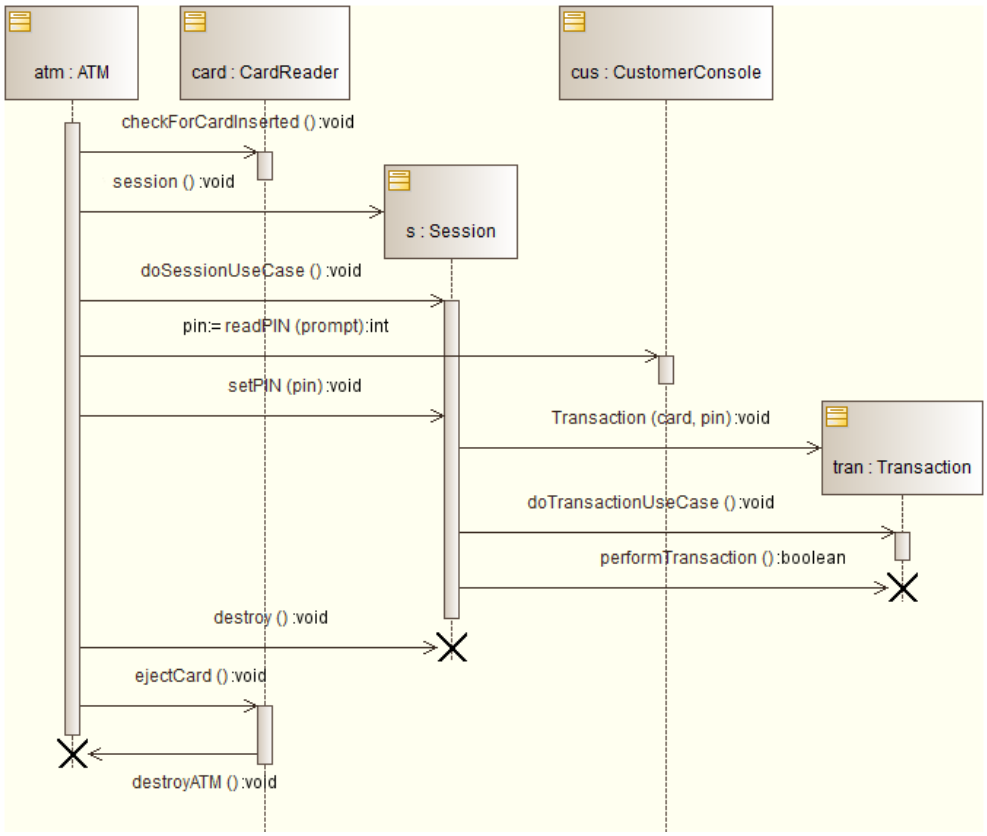
- แผนภาพลำดับแสดงเหตุการณ์ถอนเงิน
- แผนภาพลำดับแสดงเหตุการณ์ใส่รหัสบัตรผิด
- แผนภาพลำดับแสดงเหตุการณ์ทำงานปกติ
- แผนภาพลำดับแสดงเหตุการณ์ฝากเงิน
- แผนภาพลำดับแสดงเหตุการณ์โอนเงิน
- แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบยอดเงิน



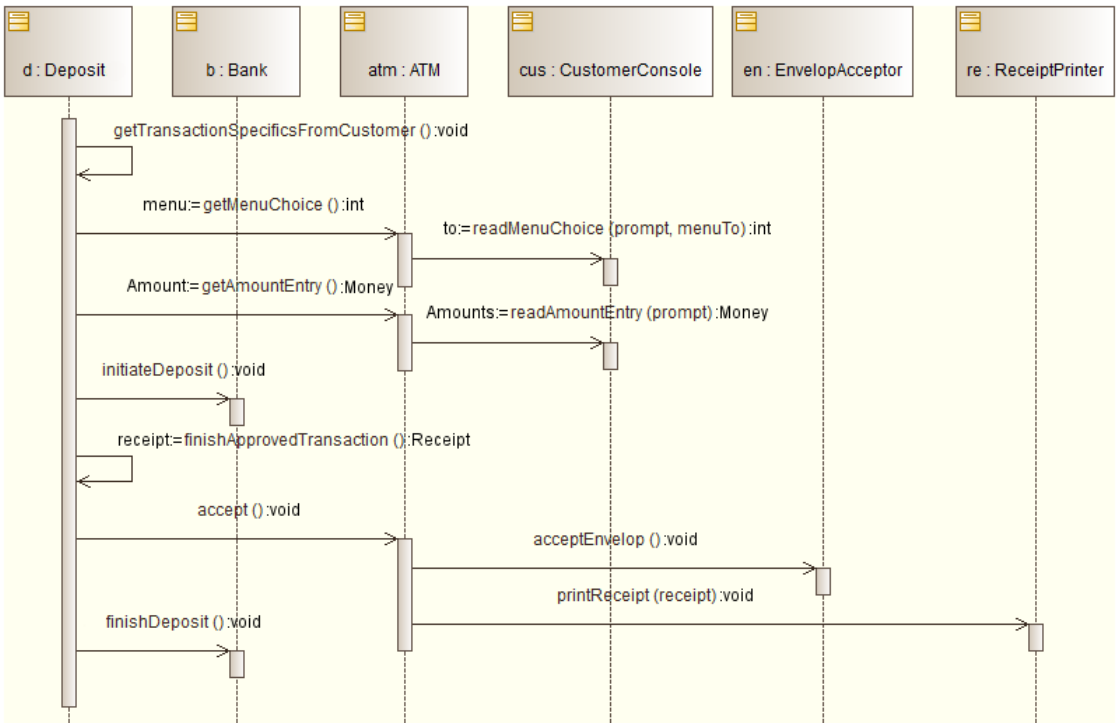
รูปที่ 5.17 แผนภาพลำดับแสดงเหตุการณ์ถอนเงิน



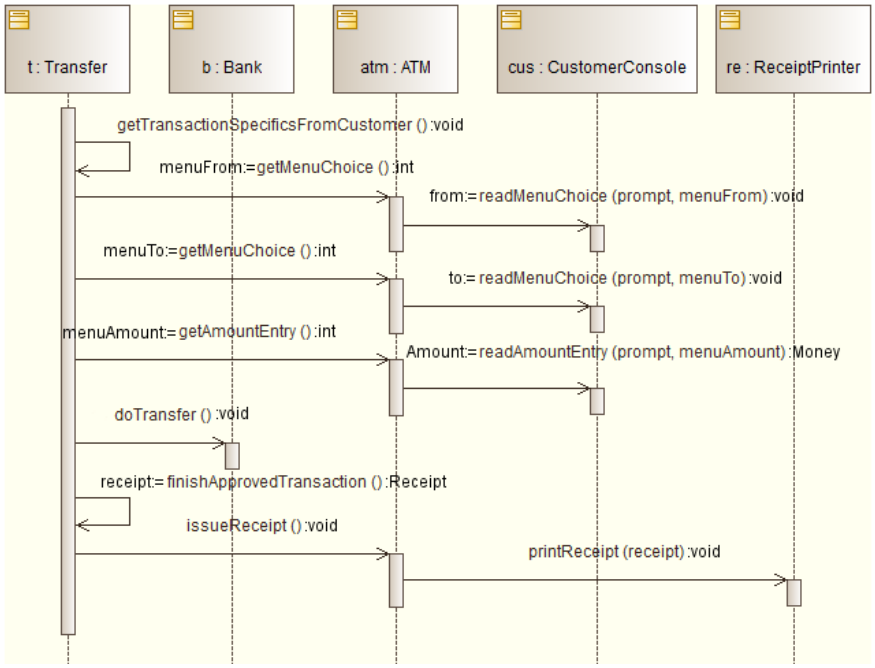
รูปที่ 5.18 แผนภาพลำดับแสดงเหตุการณ์ใส่รหัสบัตรผิด



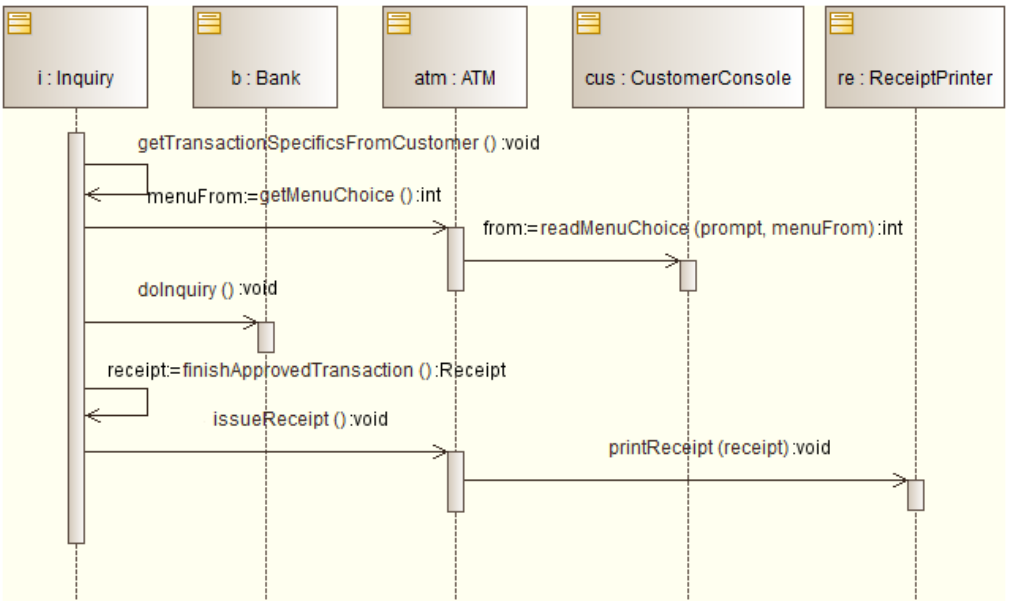
รูปที่ 5.19 แผนภาพลำดับแสดงเหตุการณ์ทำงานปกติ



รูปที่ 5.20 แผนภาพลำดับแสดงเหตุการณ์ฝากเงิน



รูปที่ 5.21 แผนภาพลำดับแสดงเหตุการณ์โอนเงิน



รูปที่ 5.22 แผนภาพลำดับแสดงเหตุการณ์ตรวจสอบยอดเงิน

5.3.2.2 ผลการตรวจสอบความพร้อมของข้อมูล

เครื่องมือตรวจไม่พบความไม่พร้อมของข้อมูล เครื่องมือตรวจสอบจึงสรุปรายละเอียดของแผนภาพคลาสและแผนภาพลำดับ ดังรูปที่ 5.23 ดังนี้

- มีคลาสทั้งหมด 16 คลาส
- มีคุณลักษณะทั้งหมด 18 คุณลักษณะ
- มีเมธอดทั้งหมด 48 เมธอด
- มีเส้นความสัมพันธ์กันทั้งหมด 21 เส้น
- มีประเภทของตัวแปรทั้งหมด 8 ประเภท
- มีแผนภาพลำดับทั้งหมด 6 แผนภาพ
- มีส่วนประกอบของระบบทั้งหมด 33 ส่วนประกอบ
- มีข้อความทั้งหมด 60 ข้อความ

```
>>> ================================ RESTART ================================
>>>
Please insert file name (*.xmi): Simple ATM.xmi
Entering Readieness Checking Module
.
..
...
....
.....
Checking result...

### Readiness checking is completed. Your diagrams are ready for checking Consistency. ###

You have 16 Classes, 18 Attributes, 48 Methods, 21 Relations, 68 Parameters and 8 Parameter
 Types

You have 6 Sequence Diagrams, 33 Participants and 60 Message
```

รูปที่ 5.23 เครื่องมือรายงานผลการตรวจสอบความพร้อมของข้อมูลและสรุปรายละเอียดของแผนภาพ

5.3.2.3 ผลการตรวจสอบความสอดคล้อง

ตรวจสอบด้วยเครื่องมือแล้ว รายงานผลแจ้งว่าเกิดความไม่สอดคล้องขึ้น 8 จุด ซึ่งขัดกับกฎข้อ 2 7 10 และ 11 ดังรูปที่ 5.24 สามารถสรุปความไม่สอดคล้องที่เกิดขึ้นได้ดังนี้

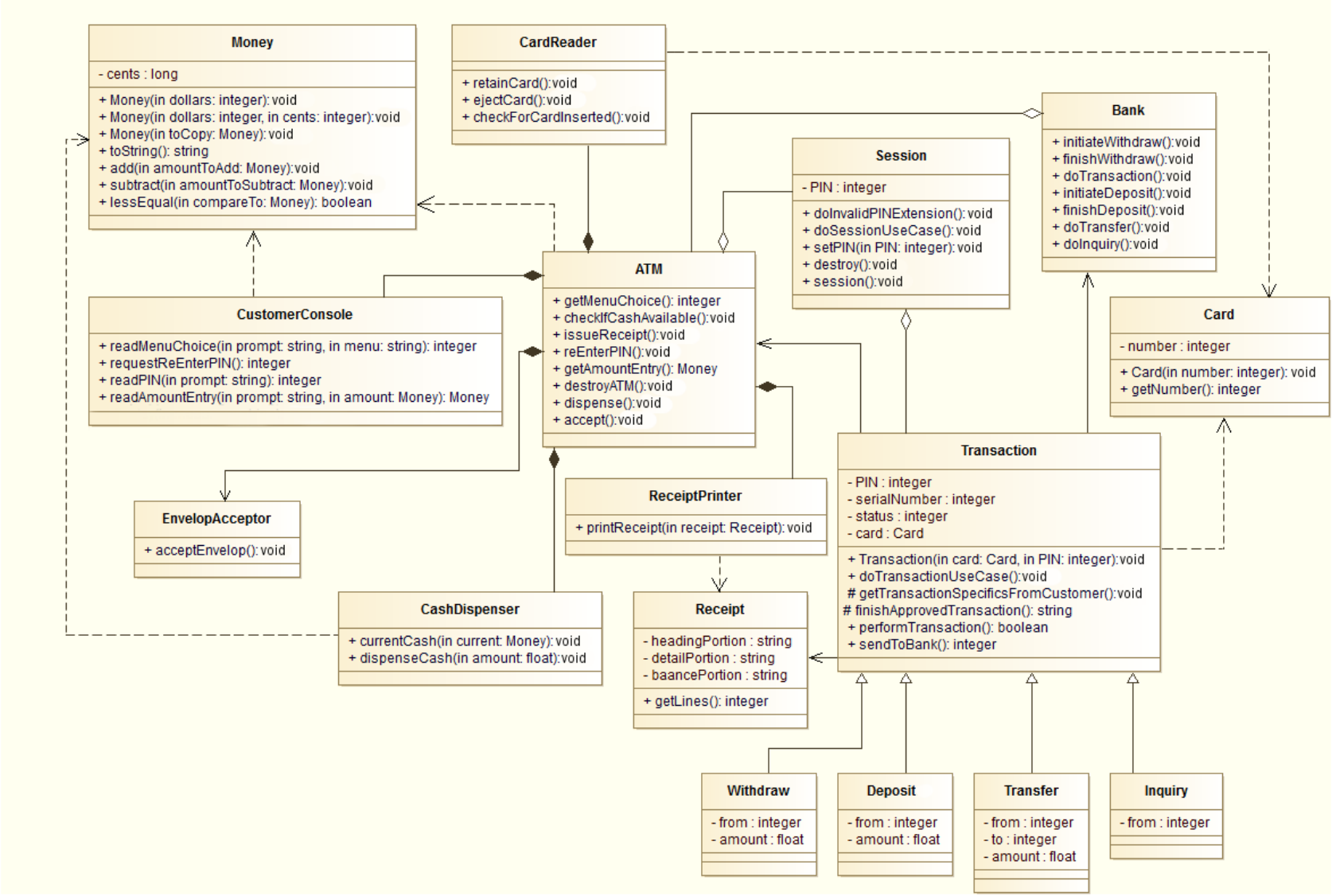
1. เมธอด display ของคลาส CustomerConsole ไม่ถูกเรียกใช้ในแผนภาพลำดับใด ๆ เลย
2. เมธอด finishApprovedTransaction ถูกคลาสผู้ใช้บริการเรียกใช้ทั้ง ๆ ที่ประกาศการเข้าถึงเป็นส่วนตัว
3. คลาส CardReader ซึ่งมีความสัมพันธ์แบบคอมโพซิชันกับคลาส ATM ส่งข้อความแบบทำลายไปยังคลาสแม่ในแผนภาพลำดับแสดงเหตุการณ์การทำงานปกติ
4. ข้อความ getMenuChoice และ finishApprovedTransaction ในแผนภาพลำดับแสดงเหตุการณ์ถอนเงินคืนค่าตัวแปรผิดชนิด

```
---Entering Consistency Checking Module---
.
..
...
....
.....
Rule 1: all participants in Sequence Diagrams must be valid as class in Class Diagram
---pass---
Rule 2: all methods in Class Diagram must be used
--not pass: There're Method(s) that don't be used.
<--error code: Class Name: < CustomerConsole > Method ID: < _L9Y-bIp2EeKz5IIZyd7PAQ > Metho
d Name: < display >
Rule 3: all classes in Class Diagram must be used as participants in Sequence Diagrams
---pass---
Rule 4: relation in Sequence Diagrams must be valid in Class Diagram
---pass---
Rule 5: subclass must not create motherclass if their relationship is aggregation
---pass---
Rule 6: subclass must not create motherclass if their relationship is composition
---pass---
Rule 7: subclass must not destroy motherclass if their relationship is composition
--not pass: There're subclass(es) sending 'DeleteMessage' to motherclass(es). [Composition]
<--error code: SQD Name: < Working > Message ID: < _L9cCAYp2EeKz5IIZyd7PAQ > Message Name:
< destroyATM >
Rule 8: after provider was destroyed, user cannot use it's services
---pass---
Rule 9: after motherclass was destroyed, subclass must be destroyed too. If their relations
hip is composition
---pass---
Rule 10: relationship between participants in Sequence Diagram must be valid and have right
in Class Diagram
---pass--- : all messages is valid
---pass--- : all messages send in right direction
--not pass: There're message(s) that call those have no permission.
<--error code: error type: < private > SQD Name: < Transfer > Message ID: < _L9dP1Yp2EeKz5I
IZyd7PAQ > Message name: < finishApprovedTransaction >
<--error code: error type: < private > SQD Name: < Deposite > Message ID: < _L9co6op2EeKz5I
IZyd7PAQ > Message name: < finishApprovedTransaction >
<--error code: error type: < private > SQD Name: < Check Balance > Message ID: < _L9dQG4p2E
eKz5IIZyd7PAQ > Message name: < finishApprovedTransaction >
<--error code: error type: < private > SQD Name: < Withdraw > Message ID: < _L9bauYp2EeKz5I
IZyd7PAQ > Message name: < finishApprovedTransaction >
Rule 11: variables in messages must match in methods
--not pass: There're variable(s) in message mismatch in Class Method.
<--error code: error type: < return > SQD Name: < Withdraw > Message ID: < _L9basIp2EeKz5II
Zyd7PAQ > Message name: < getMenuChoice > Parameter type: < String > Parameter Name: < menu
From >
<--error code: error type: < return > SQD Name: < Withdraw > Message ID: < _L9bauYp2EeKz5II
Zyd7PAQ > Message name: < finishApprovedTransaction > Parameter type: < Receipt > Parameter
Name: < receipt >
Rule 12: variables in guard condition must match in attributes or grobal variables
---pass---
Rule 13: there must not be recursion between Sequence Diagrams
---pass---
You still have 8 consistency error(s) to correct.
>>>
```

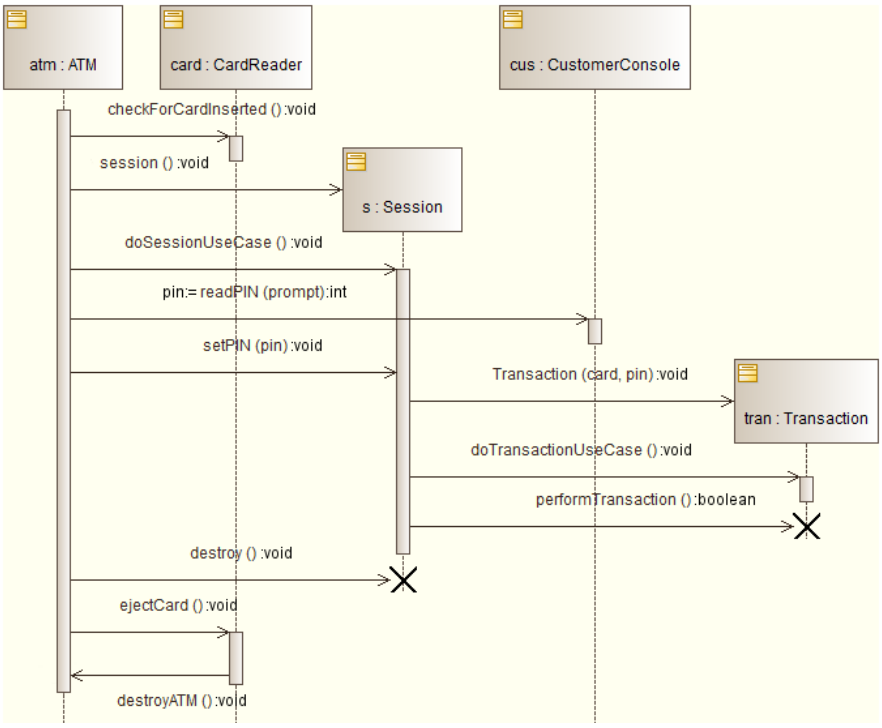
รูปที่ 5.24 รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบเอทีเอ็ม

แก้ไขความไม่สอดคล้องทั้ง 8 จุดดังนี้

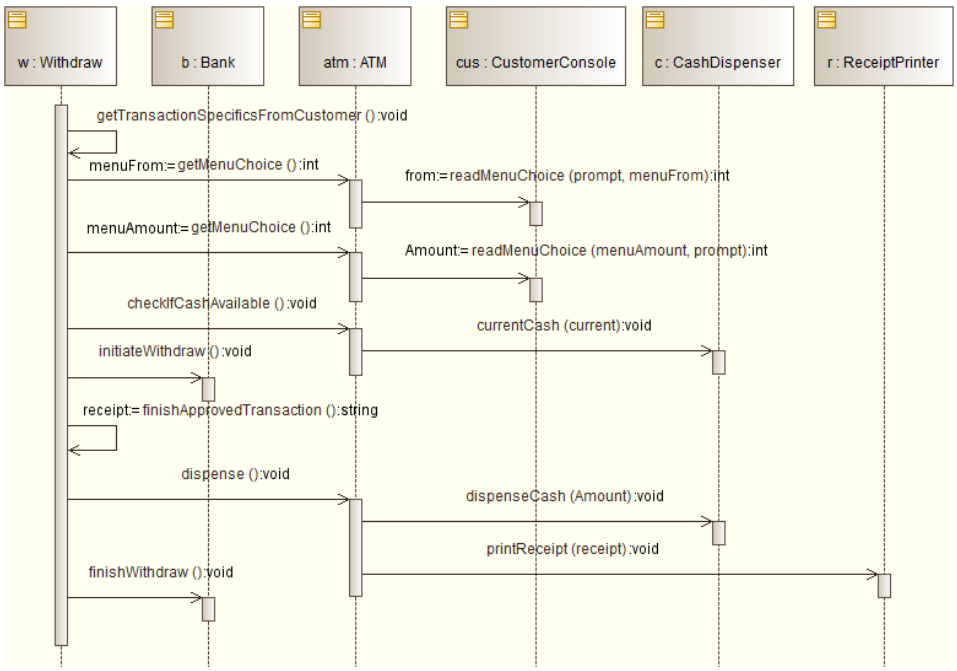
1. ตัดเมธอด display ของคลาส CustomerConsole ออก
2. เปลี่ยนการเข้าถึงเมธอด finishApprovedTransaction เป็นแบบป้องกัน ดังรูปที่ 5.25
3. เปลี่ยนประเภทการส่งข้อความ destroyATM เป็นการส่งข้อความธรรมดาแทนการส่งข้อความแบบทำลาย ดังรูปที่ 5.26
4. เปลี่ยนประเภทตัวแปรคืนค่าของข้อความ getMenuChoice เป็น Integer และ finishApprovedTransaction เป็น String ดังรูปที่ 5.27



รูปที่ 5.25 แผนภาพคลาสของระบบเอทีเอ็มที่แก้ไขตามข้อ 1-3 แล้ว



รูปที่ 5.26 แผนภาพลำดับแสดงเหตุการณ์ทำงานปกติที่แก้ไขแล้ว



รูปที่ 5.27 แผนภาพลำดับแสดงเหตุการณ์ถอนเงินที่แก้ไขแล้ว

5.3.3 กรณีศึกษาที่ 3 ระบบลงทะเบียน

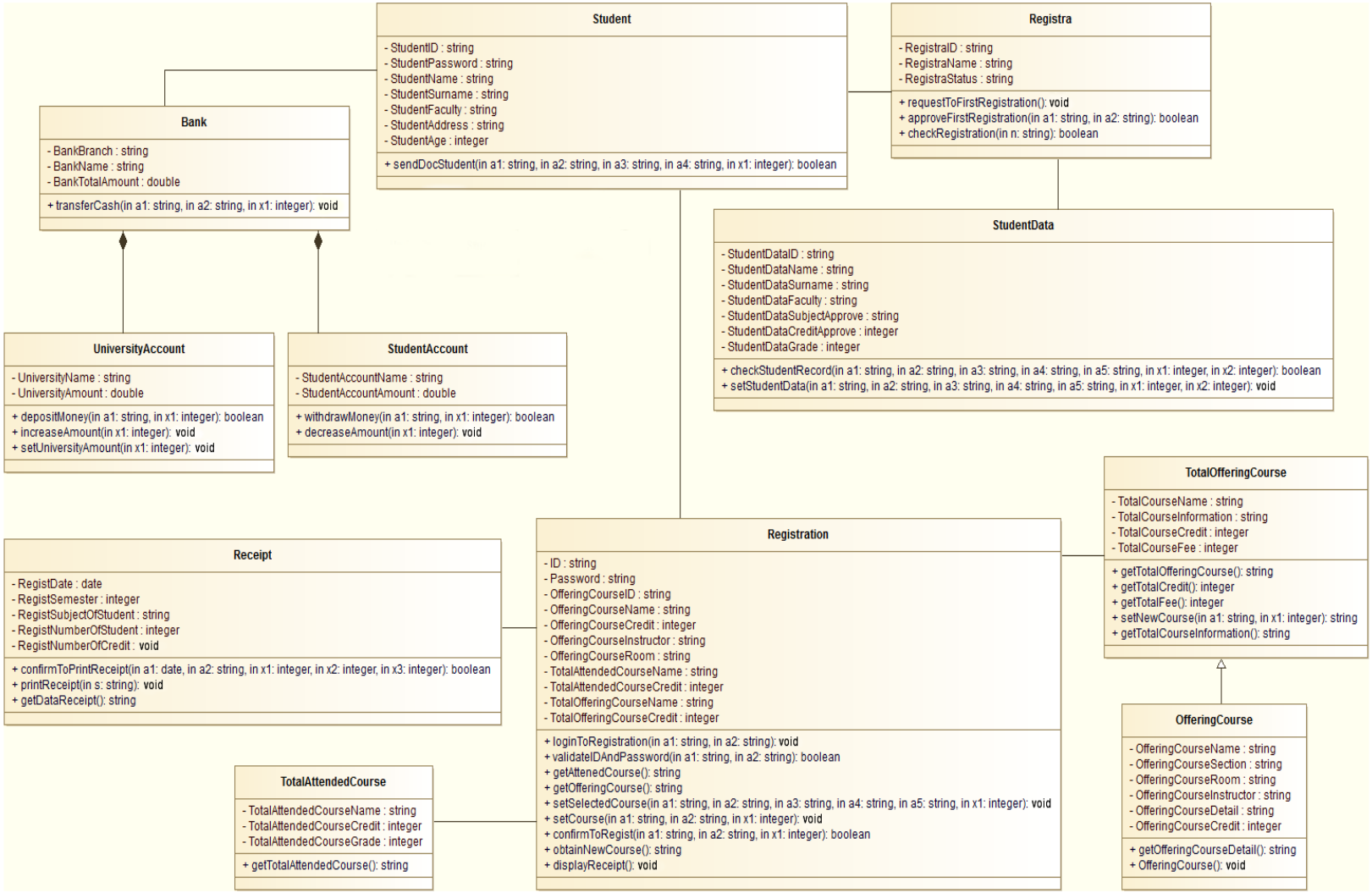
เป็นระบบที่ใช้ในการจัดการเกี่ยวกับการลงทะเบียนในมหาวิทยาลัย นิสิตสามารถลงทะเบียนโดยเลือกดูรายวิชาจากใบรายการ เมื่อลงทะเบียนแล้วระบบจะออกใบเสร็จรับเงิน นิสิตสามารถแจ้งความจำนงชำระค่าลงทะเบียนผ่านทางธนาคารได้ โดยธนาคารหนึ่งธนาคารจะประกอบด้วยบัญชีเงินฝากของนิสิตและบัญชีเงินฝากของมหาวิทยาลัยตั้งแต่หนึ่งบัญชีขึ้นไป

การลงทะเบียนยังแบ่งออกเป็นสองแบบคือการลงทะเบียนแรกเข้าซึ่งเป็นการลงทะเบียนข้อมูลของนิสิตเข้าสู่ระบบ และการลงทะเบียนเรียนตามปกติอีกด้วย

5.3.3.1 แผนภาพคลาสและแผนภาพลำดับของระบบลงทะเบียน

1. แผนภาพคลาส

แผนภาพคลาสของระบบลงทะเบียนประกอบด้วยคลาส 11 คลาส มีคลาสที่มีบทบาทหลัก ๆ เช่น Student, StudentData, Registration, Receipt, และ Bank ดังรูปที่ 5.28

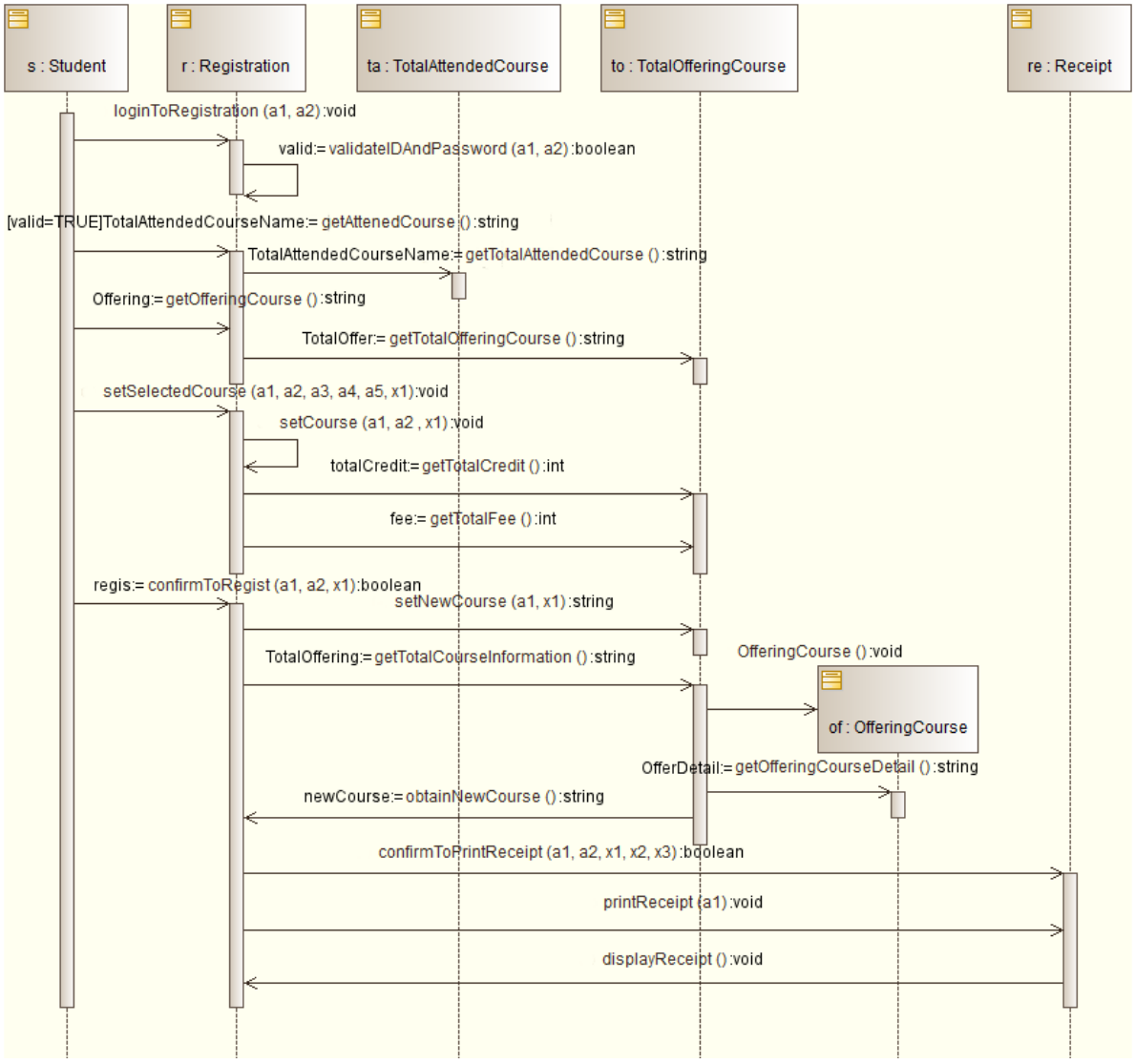


รูปที่ 5.28 แผนภาพคลาสของระบบลงทะเบียน

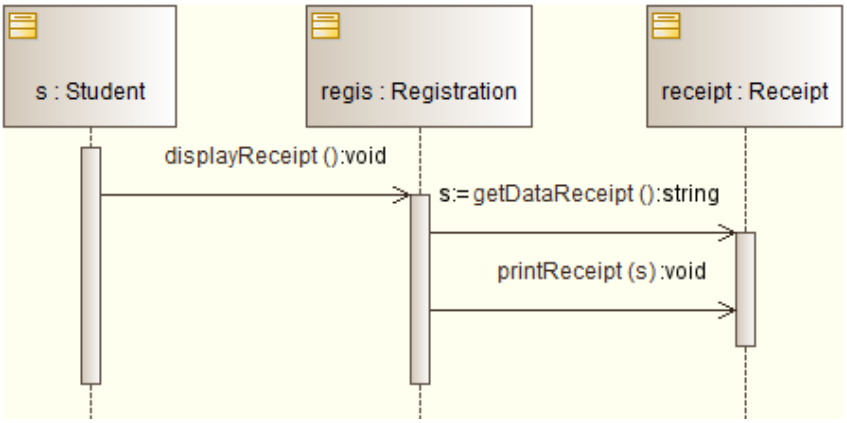
2. แผนภาพลำดับ

แผนภาพลำดับแสดงเหตุการณ์ที่เกิดขึ้นในระบบลงทะเบียน 4 เหตุการณ์ แสดงดังรูปที่ 5.29 ถึง 5.32 ตามลำดับดังนี้

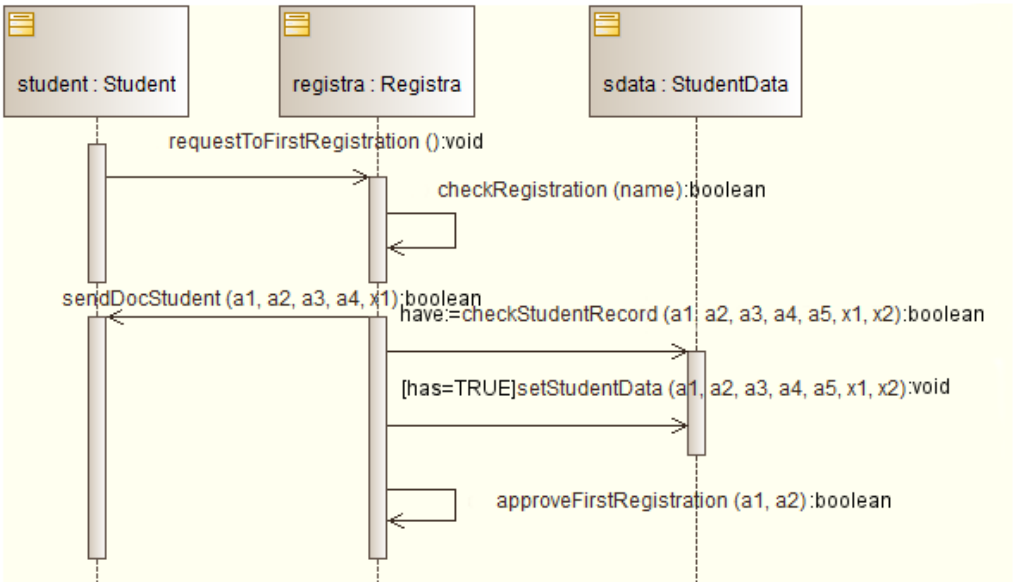
- แผนภาพลำดับแสดงเหตุการณ์ขั้นตอนการลงทะเบียน
- แผนภาพลำดับแสดงเหตุการณ์การแสดงใบเสร็จค่าลงทะเบียน
- แผนภาพลำดับแสดงเหตุการณ์การลงทะเบียนแรกเข้า
- แผนภาพลำดับแสดงเหตุการณ์การชำระเงินผ่านธนาคาร



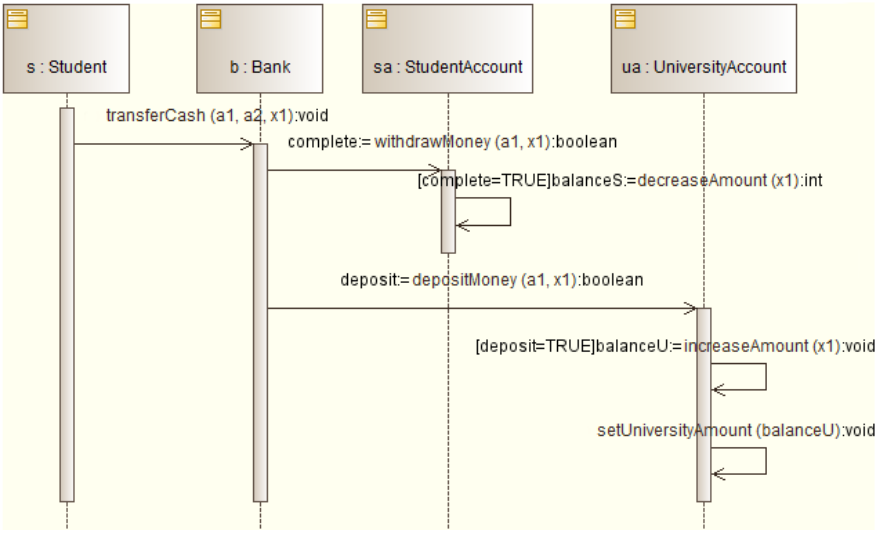
รูปที่ 5.29 แผนภาพลำดับแสดงเหตุการณ์ขั้นตอนการลงทะเบียน



รูปที่ 5.30 แผนภาพลำดับแสดงเหตุการณ์การแสดงใบเสร็จค่าลงทะเบียน



รูปที่ 5.31 แผนภาพลำดับแสดงเหตุการณ์การลงทะเบียนแรกเข้า



รูปที่ 5.32 แผนภาพลำดับแสดงเหตุการณ์การชำระเงินผ่านธนาคาร

5.3.3.2 ผลการตรวจสอบความพร้อมของข้อมูล

เครื่องมือตรวจไม่พบความไม่พร้อมของข้อมูล เครื่องมือตรวจสอบจึงสรุปรายละเอียดของแผนภาพคลาสและแผนภาพลำดับ ดังรูปที่ 5.33 ดังนี้

- มีคลาสทั้งหมด 11 คลาส
- มีคุณลักษณะทั้งหมด 53 คุณลักษณะ
- มีเมธอดทั้งหมด 32 เมธอด
- มีเส้นความสัมพันธ์กันทั้งหมด 10 เส้น
- มีประเภทของตัวแปรทั้งหมด 6 ประเภท
- มีแผนภาพลำดับทั้งหมด 4 แผนภาพ
- มีส่วนประกอบของระบบทั้งหมด 16 ส่วนประกอบ
- มีข้อความทั้งหมด 34 ข้อความ

```
>>> ================================ RESTART ================================
>>>
Please insert file name (*.xmi): Simple Registration.xmi
Entering Readieness Checking Module
.
..
...
....
.....
Checking result...

### Readiness checking is completed. Your diagrams are ready for checking Consistency. ###

You have 11 Classes, 53 Attributes, 32 Methods, 10 Relations, 88 Parameters and 6 Parameter
Types

You have 4 Sequence Diagrams, 16 Participants and 34 Message
```

รูปที่ 5.33 เครื่องมือรายงานผลการตรวจสอบความพร้อมของข้อมูลและสรุปรายละเอียดของแผนภาพ

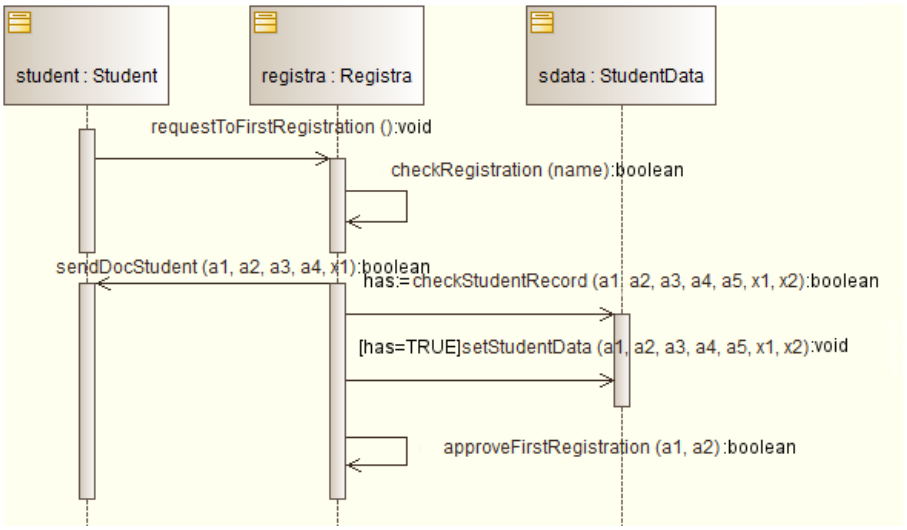
5.3.3.3 ผลการตรวจสอบความสอดคล้อง

ตรวจสอบด้วยเครื่องมือแล้ว รายงานผลแจ้งว่าเกิดความไม่สอดคล้องขึ้นจุดเดียว ซึ่งขัดกับกฎข้อ 12 ดังรูปที่ 5.34 คือตัวแปรเงื่อนไขที่ชื่อ have บนข้อความ setStudentData ของแผนภาพลำดับแสดงเหตุการณ์การลงทะเบียนแรกเข้าไม่ตรงกับที่ประกาศไว้ในแผนภาพคลาสหรือตัวแปรท้องถิ่น

```
---Entering Consistency Checking Module---
.
..
...
....
.....
Rule 1: all participants in Sequence Diagrams must be valid as class in Class Diagram
---pass---
Rule 2: all methods in Class Diagram must be used
---pass---
Rule 3: all classes in Class Diagram must be used as participants in Sequence Diagrams
---pass---
Rule 4: relation in Sequence Diagrams must be valid in Class Diagram
---pass---
Rule 5: subclass must not create motherclass if their relationship is aggregation
---pass---
Rule 6: subclass must not create motherclass if their relationship is composition
---pass---
Rule 7: subclass must not destroy motherclass if their relationship is composition
---pass---
Rule 8: after provider was destroyed, user cannot use it's services
---pass---
Rule 9: after motherclass was destroyed, subclass must be destroyed too. If their relationship is
composition
---pass---
Rule 10: relationship between participants in Sequence Diagram must be valid and have right in Cl
ass Diagram
---pass--- : all messages is valid
---pass--- : all messages send in right direction
---pass--- : all messages sent have right to do so
Rule 11: variables in messages must match in methods
---pass---
Rule 12: variables in guard condition must match in attributes or grobal variables
--not pass: There're variable(s) guard in message mismatch in Class Attribute or grobal variable.
<--error code: SQD Name: < First Registration > Message Name: < setStudentData > Guard name: < ha
ve > Guard type: < Boolean >
Rule 13: there must not be recursion between Sequence Diagrams
---pass---
You still have 1 consistency error(s) to correct.
```

รูปที่ 5.34 รายงานผลความไม่สอดคล้องของแผนภาพลำดับและแผนภาพคลาสของระบบลงทะเบียน

แก้ไขชื่อตัวแปรเงื่อนไขบนข้อความ setStudentData จาก have เป็น has ดังรูปที่ 5.35



รูปที่ 5.35 แผนภาพลำดับเหตุการณ์การลงทะเบียนแรกเข้าที่แก้ไขแล้ว

บทที่ 6

สรุปผลการวิจัยและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงการสรุปผลของงานวิจัยในการสร้างเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ รวมทั้งประโยชน์ของเครื่องมือ ข้อจำกัดของเครื่องมือ ปัญหาและอุปสรรค และข้อเสนอแนะ

6.1 สรุปผลการวิจัย

งานวิจัยได้รวบรวมและนำเสนอกฎในการตรวจสอบความพร้อมของข้อมูลและความสอดคล้องของแผนภาพคลาสและแผนภาพลำดับดังต่อไปนี้

6.1.1 สรุปการตรวจสอบความพร้อมของข้อมูลของแผนภาพทั้งสองต้องพิจารณาสิ่งต่อไปนี้

แผนภาพคลาส

- แผนภาพคลาสจะต้องมีชื่อของระบบที่อธิบายอยู่
- คลาสทุกคลาสที่ปรากฏในแผนภาพต้องมีชื่อคลาส
- คลาสทุกคลาสที่ปรากฏในแผนภาพต้องมีเมธอดและการเข้าถึงของเมธอดระบุไว้
- คลาสทุกคลาสต้องมีความสัมพันธ์กับคลาสอื่น

แผนภาพลำดับ

- แผนภาพลำดับทุกแผนภาพจะต้องมีชื่อแผนภาพ
- แอคเตอร์และส่วนประกอบของระบบที่ปรากฏในแผนภาพต้องมีชื่อ
- ข้อความที่ส่งหากันทุกข้อความจะต้องมีชื่อเมธอด
- ส่วนประกอบของต้องมีความสัมพันธ์กับส่วนอื่น

6.1.2 สรุปกฎการตรวจสอบความสอดคล้อง

- กฎข้อที่ 1 แอคเตอร์และส่วนของระบบในแผนภาพลำดับจะต้องปรากฏเป็นคลาสในแผนภาพคลาส
- กฎข้อที่ 2 เมธอดทุกเมธอดในแผนภาพคลาสจะต้องถูกเรียกใช้ในแผนภาพลำดับ
- กฎข้อที่ 3 คลาสทุกคลาสในแผนภาพคลาสจะต้องปรากฏเป็นแอ็คเตอร์หรือส่วนของระบบในแผนภาพลำดับ
- กฎข้อที่ 4 ความสัมพันธ์ของแอคเตอร์และส่วนของระบบในแผนภาพลำดับจะต้องมีอยู่จริงในแผนภาพคลาส

- กฎข้อที่ 5 คลาสย่อยไม่สามารถส่งข้อความแบบสร้างในแผนภาพลำดับไปสร้างคลาสหลักได้ ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบแอกกรีเกชันในแผนภาพคลาส

- กฎข้อที่ 6 คลาสย่อยไม่สามารถส่งข้อความแบบสร้างในแผนภาพลำดับไปสร้างคลาสหลักได้ ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส

- กฎข้อที่ 7 คลาสย่อยไม่สามารถส่งข้อความแบบทำลายในแผนภาพลำดับไปทำลายคลาสหลักได้ ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส

- กฎข้อที่ 8 ไม่สามารถเรียกคลาสผู้ให้บริการผ่านคลาสผู้ใช้บริการได้ ถ้าคลาสผู้ให้บริการถูกทำลายไปแล้ว

- กฎข้อที่ 9 ถ้าคลาสย่อยและคลาสหลักมีความสัมพันธ์แบบคอมโพซิชันชันในแผนภาพคลาส เมื่อคลาสหลักถูกทำลายในแผนภาพลำดับ คลาสย่อยจะถูกทำลายด้วย

- กฎข้อที่ 10.1 ทิศทางของความสัมพันธ์ ถ้าเป็นความสัมพันธ์ทางเดียว คลาสด้านหัวลูกศรจะเรียกคลาสอีกด้านไม่ได้

- กฎข้อที่ 10.2 การเข้าถึงของเมธอด ถ้าการเข้าถึงเป็นแบบสาธารณะคลาสที่สัมพันธ์กันจะมีสิทธิ์เข้าถึง ถ้าเป็นการเข้าถึงแบบป้องกัน คลาสลูกและตัวเองเท่านั้นที่มีสิทธิ์เข้าถึง ถ้าเป็นการเข้าถึงแบบส่วนตัว คลาสตัวเองเท่านั้นที่มีสิทธิ์เข้าถึง

- กฎข้อที่ 11 จำนวน ลำดับ และชนิดของพารามิเตอร์ของเมธอดบนข้อความในแผนภาพลำดับ จะต้องเหมือนกับที่ประกาศไว้ในคลาสของแผนภาพคลาส

- กฎข้อที่ 12 ตัวแปร และชนิดของตัวแปรที่เป็นตัวกำหนดเงื่อนไขในเครื่องหมาย '[]' บนข้อความในแผนภาพลำดับ จะต้องประกาศเป็นคุณลักษณะในคลาสที่ถูกเรียก หรือตัวแปรจากข้อความก่อนหน้า

- กฎข้อที่ 13 ข้อความที่ส่งต่อกันระหว่างส่วนของระบบสามส่วนในแผนภาพลำดับ ที่มีลักษณะเป็นข้อความหลักและข้อความย่อยแล้ว ถ้ามีแผนภาพลำดับอื่นที่มีข้อความตัวเดียวกันกับแผนภาพลำดับแรก ข้อความย่อยจะต้องไม่เรียกข้อความหลัก

6.2 ประโยชน์ของเครื่องมือ

ช่วยตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ เพื่อลดเวลาที่ต้องใช้ในช่วงการออกแบบลง

6.3 ข้อจำกัดของเครื่องมือ

6.3.1 เครื่องมือตรวจสอบได้เฉพาะความครบถ้วนสมบูรณ์เชิงโครงสร้างของการออกแบบที่ชั้นนามธรรมเพื่อตรวจสอบความพร้อมของข้อมูลเท่านั้น ไม่สามารถตรวจสอบความครบถ้วนสมบูรณ์เชิงความหมายได้

6.3.2 เครื่องมือตรวจสอบได้เฉพาะความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับที่นำเข้าจากไฟล์เอ็กซ์เอ็มไอได้เท่านั้น ไม่สามารถตรวจสอบความสอดคล้องกับแผนภาพอื่น ๆ และความต้องการซอฟต์แวร์ได้

6.4 ปัญหาและอุปสรรค

6.4.1 มาตรฐานกลางเอ็กซ์เอ็มไอใช้เป็นตัวกลางในการแลกเปลี่ยนข้อมูลระหว่างเครื่องมือที่ใช้สร้างแผนภาพยูเอ็มแอล แต่ในทางปฏิบัติ ไฟล์เอ็กซ์เอ็มไอเหล่านั้นกลับใช้ไม่ได้ผลจริง มีทั้งแลกเปลี่ยนได้บางส่วนและแลกเปลี่ยนกันไม่ได้เลย ทำให้การพัฒนาอยู่บนข้อจำกัดว่า จะต้องเป็นแฟ้มข้อมูลเอ็กซ์เอ็มไอที่นำออกจากเครื่องมือสร้างแผนภาพยูเอ็มแอลเดียวกัน นั่นคือจากโปรแกรม Modelio เท่านั้น

6.4.2 แม้จะมีกฎการตรวจสอบความครบถ้วนสมบูรณ์เพื่อบังคับให้แผนภาพมีความละเอียดถึงระดับที่นำไปตรวจสอบความสอดคล้องได้ แต่กฎดังกล่าวก็ไม่สามารถบังคับให้แผนภาพมีข้อมูลครบทุกอย่างได้หมด เช่นพารามิเตอร์หรือการคืนค่าอาจไม่จำเป็นต้องมีก็ได้ ดังนั้นจึงยังเป็นข้อจำกัดในบางอย่างที่ไม่สามารถตรวจสอบได้ ทำให้ผลการตรวจสอบอาจไม่ถูกต้องเสมอไป

6.5 ข้อเสนอแนะ

6.5.1 เครื่องมือนี้อาจสามารถพัฒนาให้ยืดหยุ่นกับการตรวจสอบด้วยกฎมากขึ้น กล่าวคือสามารถตรวจสอบได้ว่าแผนภาพที่จะทำการตรวจสอบมีความละเอียดให้ตรวจสอบถึงขั้นไหน แล้วจึงเลือกตรวจสอบเฉพาะกฎที่มีข้อมูลเพียงพอ

6.5.2 อาจพิจารณาให้ครอบคลุมถึงความหลากหลาย (multiplicity) ด้วย

6.5.3 เครื่องมือนี้อาจสามารถพัฒนาให้แก้ไขส่วนที่ไม่สอดคล้องได้อย่างอัตโนมัติ โดยที่ผู้ใช้ไม่จำเป็นต้องมาแก้ไขเองเพื่อลดเวลาที่ใช้ในการแก้ไขลงได้อีก

6.5.4 งานวิจัยนี้เครื่องมือสามารถตรวจสอบความสอดคล้องเฉพาะแผนภาพคลาสและแผนภาพลำดับ จึงสามารถพัฒนาให้รองรับแผนภาพอื่น ๆ เพิ่มเติมได้

รายการอ้างอิง

[1] Object Management Group. OMG Unified Modeling Language 2.3 Superstructure [Online]. 2010. Available from: http://www.omg.org/spec/UML/2.3/ 2010. [2010, Oct 15]

[2] Miles, R. and Hamilton, K. Learning UML 2.0. California: O'Reilly Media, 2006.

[3] Kotb, Y. and Katayama, T. Consistency checking of UML model diagrams using the XML semantics approach. In Special interest tracks and posters of the 14th international conference on World Wide Web, 982-983. New York, USA, 2005.

[4] Egyed, A. Instant consistency checking for the UML. In Proceedings of the 28th international conference on Software engineering, 381-390. New York, USA, 2006.

[5] Li, X., Qiu, X., Wang, L., Lei, B., and Wong, W. E. UML state machine diagram driven runtime verification of Java programs for message interaction consistency. In Proceedings of the 2008 ACM symposium on Applied computing, 384-389. New York, USA, 2008.

[6] Rational Software Corporation. Guidelines: Class Diagram [Online]. 2002. Available from: http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/modguide/md_clsdm.htm 2002. [2012, Jun 20]

[7] Scott, W. UML 2 Class Diagram Guidelines [Online]. 2002. Available from: http://www.agilemodeling.com/style/classDiagram.htm [2012, Jun 20]

[8] Bell, D. UML basics: The class diagram [Online]. 2004. Available from: http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/ [2012, Jun 20]

[9] Rational Software Corporation. Guidelines: Sequence Diagram [Online]. 2002. Available from: http://www.ts.mah.se/RUP/RationalUnifiedProcess/process/modguide/md_seqdm.htm [2012, Jun 20]

[10] Scott, W. UML 2 Sequence Diagramming Guidelines [Online]. 2002. Available from: http://www.agilemodeling.com/style/sequenceDiagram.htm [2012, Jun 20]

[11] Object Management Group. metamodel, in UML 2.4.1, for the XMI-specific model elements [Online]. 2011. Available from: http://www.omg.org/spec/XMI/20110701/XMI-model.xmi 2011. [2012, Jun 20]

[12] Object Management Group. Documents Associated With MOF/XMI Mapping, Version 2.4.1 [Online]. 2011. Available from: http://www.omg.org/spec/XMI/ 2.4.1/PDF. [2012, Jun 20]

[13] IEEE-SA. IEEE Recommended Practice for Software Requirements Specifications [Online]. 1998. Available from: http://standards.ieee.org [2010, Oct 15]

[14] Wang, H., Feng, T., Zhang, J., and Zhang, K. Consistency check between behavior Models. In IEEE International Symposium on Communications and Information Technology, 486-489. Beijing, China, 2005.

[15] Dubauskaite, R. and Vasilecas, O. The approach of ensuring consistency of UML model based on rules. In Proceedings of the 11th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing on International Conference on Computer Systems and Technologies, 71-76. New York, USA, 2010.

[16] Simmonds, J., Van Der Straeten, R., Jonckers, V., and Mens, T. Maintaining consistency between UML models using description logic. Série L'objet-logiciel, base de données, réseaux (October 2004): 231-244.

[17] Alanazi, M. N. Basic Rules to Build Correct UML Diagrams. In Proceedings of the 2009 International Conference on New Trends in Information and Service Science, 72–76. Beijing, China, 2009.

[18] Briand, L. C., Labiche, Y., and O'Sullivan, L. Impact Analysis and Change Management of UML Models. In Proceedings of the International Conference on Software Maintenance, 256-265. Washington, DC, USA, 2003.

[19] กณิษฐา บุญคุ้ม. การออกแบบและพัฒนาเครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาส แผนภาพซีเควนซ์ และแผนภาพสเตทชาร์ท. วิทยานิพนธ์ปริญญามหาบัณฑิต, สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2550.

ภาคผนวก

ภาคผนวก ก

การใช้งานเครื่องมือ

ในภาคผนวกนี้จะกล่าวถึงการเตรียมสภาพแวดล้อมและขั้นตอนการใช้งาน ของการใช้เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับ

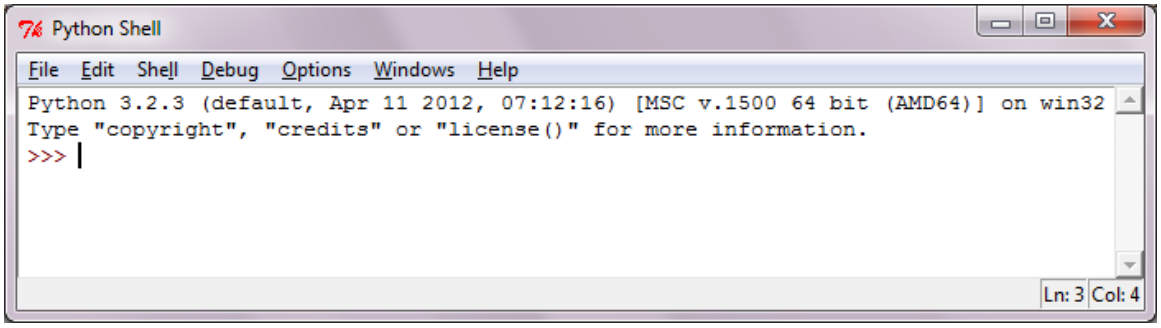
1. สภาพแวดล้อมในการใช้งาน

เครื่องมือตรวจสอบความสอดคล้องระหว่างแผนภาพคลาสและแผนภาพลำดับที่พัฒนาขึ้นมาในงานวิจัยนี้ สร้างออกมาเป็น Python file ดังนั้นเครื่องคอมพิวเตอร์ที่ใช้จะต้องติดตั้ง Python เสียก่อน โดยใช้ Python รุ่น 3.2.3 ซึ่งสามารถดาวน์โหลดได้ฟรีที่ http://www.python.org/download/ โดยรองรับทั้งระบบปฏิบัติการ Windows Mac OSX และ Linux

เมื่อติดตั้งเสร็จแล้ว ให้นำไฟล์ Checking.py ซึ่งเป็นไฟล์เครื่องมือที่งานวิจัยนี้พัฒนาขึ้น ไปวางไว้ในโฟลเดอร์ของ Python ที่ติดตั้งลงไป เช่นเดียวกับไฟล์เอ็กซ์เอ็มไอที่ต้องการนำมาตรวจสอบความสอดคล้อง หรืออาจวางไว้ในโฟลเดอร์ใดก็ได้ แต่ต้องวางไว้ในโฟลเดอร์เดียวกัน

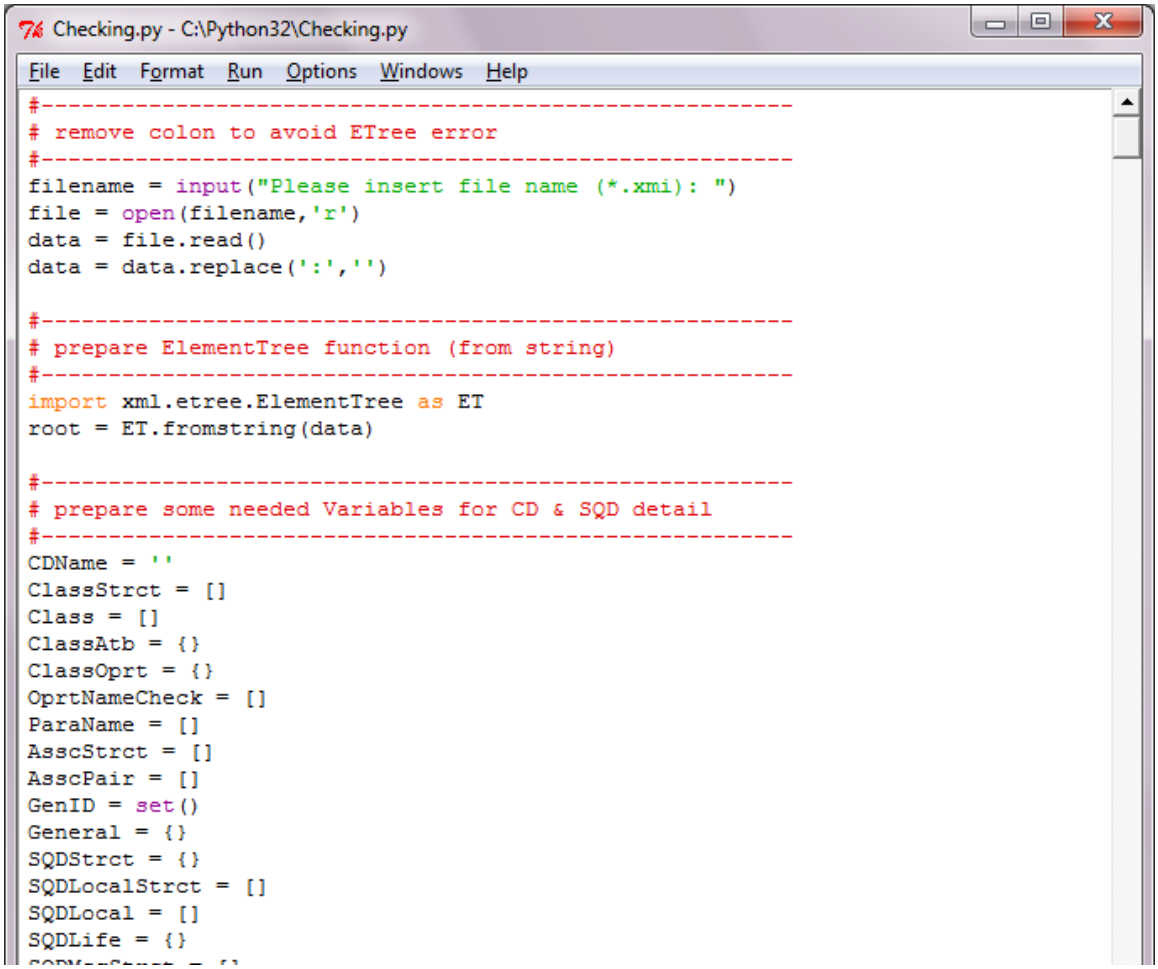
2. ขั้นตอนการใช้งาน

2.1 เปิด Python Shell ที่ติดตั้งไว้ขึ้นมา ดังรูปที่ ก.1



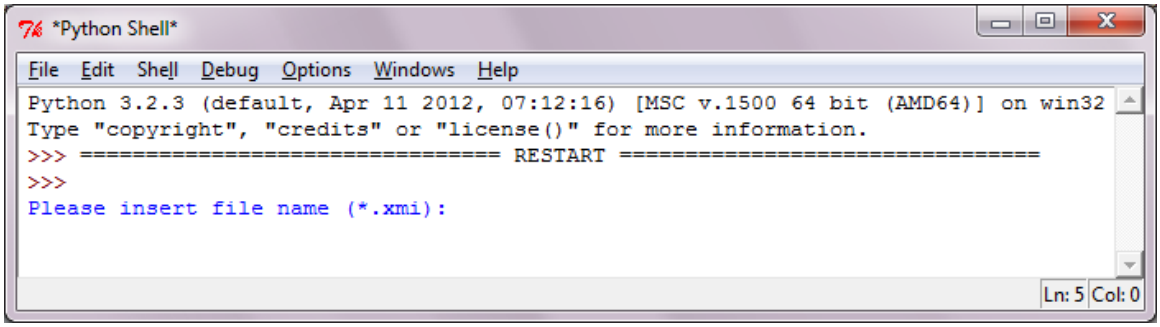
รูปที่ ก.1 Python Shell เมื่อเริ่มเปิดใช้

2.2 เปิดไฟล์ Checking.py ด้วยการกด Ctrl + o แล้วค้นหาไฟล์ที่เก็บไว้ในโฟลเดอร์ จะทำให้เปิดหน้าต่าง Checking.py ขึ้นมาใหม่ดังรูปที่ ก.2



รูปที่ ก.2 หน้าต่าง Checking.py โค้ดของเครื่องมือตรวจสอบ

2.3 สั่งรันโปรแกรมด้วยการกด F5 โปรแกรมจะเริ่มทำงานในหน้าต่าง Python Shell ดังรูป ก.3



รูปที่ ก.3 เครื่องมือเริ่มทำงาน โดยถามชื่อไฟล์เอ็กซ์เอ็มไอที่จะตรวจสอบ

2.4 โปรแกรมจะถามชื่อไฟล์เอ็กซ์เอ็มไอที่จะตรวจสอบ ให้กรอกชื่อไฟล์ลงไปและกด Enter โปรแกรมจะทำงานและรายงานผลทันที ดังตัวอย่างในรูปที่ ก.4

```
>>> ================================ RESTART ================================
>>>
Please insert file name (*.xmi): Simple Registration_r.xmi
Entering Readieness Checking Module
.
..
...
....
.....
Checking result...

### Readiness checking is completed. Your diagrams are ready for checking Consistency. ###

You have 11 Classes, 53 Attributes, 32 Methods, 10 Relations, 88 Parameters and 6 Paramete
r Types

You have 4 Sequence Diagrams, 16 Participants and 34 Message

---Entering Consistency Checking Module---
.
..
...
....
.....
Rule 1: all participants in Sequence Diagrams must be valid as class in Class Diagram
---pass---
Rule 2: all methods in Class Diagram must be used
---pass---
Rule 3: all classes in Class Diagram must be used as participants in Sequence Diagrams
---pass---
Rule 4: relation in Sequence Diagrams must be valid in Class Diagram
---pass---
Rule 5: subclass must not create motherclass if their relationship is aggregation
---pass---
Rule 6: subclass must not create motherclass if their relationship is composition
---pass---
Rule 7: subclass must not destroy motherclass if their relationship is composition
---pass---
Rule 8: after provider was destroyed, user cannot use it's services
---pass---
Rule 9: after motherclass was destroyed, subclass must be destroyed too. If their relation
ship is composition
---pass---
Rule 10: relationship between participants in Sequence Diagram must be valid and have righ
t in Class Diagram
---pass--- : all messages is valid
---pass--- : all messages send in right direction
---pass--- : all messages sent have right to do so
Rule 11: variables in messages must match in methods
---pass---
Rule 12: variables in guard condition must match in attributes or grobal variables
---pass---
Rule 13: there must not be recursion between Sequence Diagrams
---pass---
Congraulations! Your diagrams pass all consistency rules.
```

รูปที่ ก.4 เครื่องมือทำงานและแสดงรายงานการตรวจสอบ

ภาคผนวก ข

รูปแบบของรายการที่แสดงผลของการตรวจสอบ

การแสดงผลของการตรวจสอบของเครื่องมือ จะแบ่งออกเป็น 2 ส่วนใหญ่ ๆ ได้แก่รายงานการตรวจสอบความพร้อมของข้อมูล และรายงานการตรวจสอบความสอดคล้อง โดยมีรายละเอียดดังนี้

1. รายงานการตรวจสอบความพร้อมของข้อมูล

1.1 เมื่อตรวจสอบผ่านความพร้อมของข้อมูลทั้งสองแผนภาพ เครื่องมือจะรายงานว่า “### Readiness checking is completed. Your diagrams are ready for checking Consistency. ###”

1.2 เมื่อตรวจสอบพบความไม่พร้อมของข้อมูลของทั้งสองแผนภาพหรือแผนภาพใดแผนภาพหนึ่ง เครื่องมือจะรายงานความไม่พร้อมของข้อมูลที่ตรวจพบนั้น โดยรายงานตามแผนภาพ และแต่ละแผนภาพแบ่งย่อยแผนภาพละ 4 หัวข้อดังนี้

1.2.1 แผนภาพคลาส

- ชื่อระบบ จะรายงานผลดังนี้ "<--Class Diagram wasn't given any name."

- ชื่อคลาส จะรายงานผล 2 ส่วนคือ

- มีคลาสทั้งหมดกี่คลาสที่ไม่ได้กำหนดชื่อดังนี้ "<--There's <จำนวนคลาส> Class(es) that have no given name."

- รายงานรหัสของคลาสที่ไม่ได้กำหนดชื่อดังนี้ "----Class ID: <รหัสคลาส> doesn't have name."

- เมธอดของคลาสและการเข้าถึงของเมธอด จะรายงานผล 3 ส่วนคือ

- รายงานว่ามีปัญหากี่จุดดังนี้ "<--There's <จำนวนปัญหา> Class Methods errors which have no given Method or Visibility."

- รายงานรหัสและชื่อของคลาสที่ไม่มีเมธอดดังนี้ "----Class ID: <รหัสคลาส> Class name: <ชื่อคลาส> doesn't have operation."

- รายงานรหัสและชื่อคลาสและเมธอด ดังนี้ "----Class ID: <รหัสคลาส> Class name: <ชื่อคลาส> Meth ID: <รหัสเมธอด> Meth Name: <ชื่อเมธอด> doesn't have visibility."

- ความสัมพันธ์ของคลาส จะรายงานผล 2 ส่วนคือ

- รายงานว่ามีคลาสกี่คลาสที่ไม่มีความสัมพันธ์กับคลาสอื่นเลยดังนี้ "<--There's <จำนวนคลาส> Class(s) that have no relation to other Class."

- รายงานรหัสและชื่อคลาสที่ไม่มีความสัมพันธ์กับคลาสอื่นดังนี้ "----Class ID: <รหัสคลาส> Class name: <ชื่อคลาส> doesn't have any relation to others."

1.2.2 แผนภาพลำดับ

- ชื่อแผนภาพ จะรายงานผล 2 ส่วนคือ

- รายงานว่ามีแผนภาพลำดับกี่แผนภาพที่ไม่ได้กำหนดชื่อแผนภาพดังนี้ "<--There's <จำนวนแผนภาพ> SQD Diagram have no given name."

- รายงานรหัสแผนภาพลำดับที่ไม่ได้กำหนดชื่อดังนี้ "----SQD ID: <รหัสแผนภาพ> doesn't have name."

- ชื่อแอคเตอร์และส่วนประกอบของระบบจะรายงานผล 2 ส่วนคือ

- รายงานว่ามีแอคเตอร์หรือส่วนประกอบของระบบกี่ส่วนที่ไม่ได้กำหนดชื่อดังนี้ "<--There's <จำนวนแอคเตอร์หรือส่วนประกอบของระบบ> Actor(s) or Participant(s) that have no given name."

- รายงานรหัสแอคเตอร์หรือส่วนประกอบของระบบที่ไม่ได้กำหนดชื่อและชื่อแผนภาพลำดับนั้นดังนี้ "----Participant ID: <รหัสแอคเตอร์หรือส่วนประกอบของระบบ> in SQD name: <ชื่อแผนภาพลำดับ> doesn't have name."

- ชื่อเมธอดของข้อความจะรายงานผล 2 ส่วนคือ

- รายงานว่ามีข้อความกี่ข้อความที่ไม่ได้กำหนดชื่อดังนี้ "<--There's <จำนวนข้อความ> Message(s) that have no given name."

- รายงานรหัสข้อความที่ไม่ได้กำหนดชื่อและชื่อแผนภาพลำดับนั้นดังนี้ "----Message ID: <รหัสข้อความ> in SQD name: <ชื่อแผนภาพลำดับ> doesn't have name."

- ความสัมพันธ์ของส่วนประกอบของระบบจะรายงานผล 2 ส่วนคือ

- รายงานว่ามีแอคเตอร์หรือส่วนประกอบของระบบกี่ส่วนที่ไม่มีความสัมพันธ์กับส่วนอื่นเลยดังนี้ "<--There's <จำนวนแอคเตอร์หรือส่วนประกอบของระบบ> Actor(s) or Participant(s) that have no relation to others."

- รายงานรหัสแอคเตอร์หรือส่วนประกอบของระบบที่ไม่มีวความสัมพันธ์กับส่วนอื่นและชื่อแผนภาพลำดับนั้นดังนี้ "----Participant ID: <รหัสแอคเตอร์หรือส่วนประกอบของระบบ> in SQD name: <ชื่อแผนภาพลำดับ> doesn't have any relation to others."

1.3 หลังจากรายงานความไม่พร้อมของข้อมูลที่พบแล้วเครื่องมือจะรายงานว่า "### Readiness checking not pass. Please fix problem before checking consistency between diagrams. ###" และสรุปปัญหาที่พบทั้งหมดอีกครั้งดังนี้ ">> You have <จำนวนปัญหาของแผนภาพคลาส> Class diagram Readiness problems and <จำนวนปัญหาของแผนภาพลำดับ> Sequence diagram Readiness problems to fix."

1.4 นอกจากนี้เครื่องมือยังรายงานภาพรวมของแผนภาพทั้งหมดดังนี้

- แผนภาพคลาส “You have <จำนวนคลาส> Classes <จำนวนคุณลักษณะ> Attributes <จำนวนเมธอด> Methods <จำนวนเส้นความสัมพันธ์> Relations <จำนวนพารามิเตอร์> Parameters and <จำนวนชนิดของพารามิเตอร์> Parameter Types”

- แผนภาพลำดับ “You have <จำนวนแผนภาพลำดับ> Sequence Diagrams <จำนวนแอคเตอร์หรือส่วนประกอบของระบบ> Participants and <จำวนวข้อความ> Message”

2. รายงานการตรวจสอบความสอดคล้อง

เครื่องมือจะรายงานการตรวจสอบความสอดคล้องออกมาตามกฎเป็นข้อ ๆ ไป ซึ่งถ้าตรวจสอบกฎข้อไหนผ่าน จะรายงานว่า “---pass---” ส่วนข้อที่ไม่ผ่านจะรายงานดังนี้

2.1 รายงานว่า "--not pass: There're Actor(s) or Paticipant(s) don't appear in Class Diagram." และบอกรายละเอียดดังนี้ "<--error code: SQD ID: <รหัสแผนภาพลำดับ> Participant ID: <รหัสส่วนประกอบของระบบ>"

2.2 รายงานว่า "--not pass: There're Method(s) that don't be used." และบอกรายละเอียดดังนี้ "<--error code: Class Name: <ชื่อคลาส> Method ID: <รหัสเมธอด> Method Name: <ชื่อเมธอด>"

2.3 รายงานว่า "--not pass: There're Class(es) that don't be used." และบอกรายละเอียดดังนี้ "<--error code: Class ID: <รหัสคลาส> Class name: <ชื่อคลาส>"

2.4 รายงานว่า "--not pass: There're Relation(s) between Actor or Paticipant that don't appear in Class Diagram." และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message name: <ชื่อข้อความ>"

2.5 รายงานว่า "--not pass: There're subclass(es) sending 'CreateMessage' to motherclass(es). [Aggregation]" และบอกรายละเอียดดังนี้ <--error code: Sequence Diagram Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message Name: <ชื่อข้อความ>"

2.6 รายงานว่า "--not pass: There're subclass(es) sending 'CreateMessage' to motherclass(es). [Composition]" และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ"> Message ID: <หมายเลขข้อความ> Message Name: <ชื่อข้อความ>"

2.7 รายงานว่า "--not pass: There're subclass(es) sending 'DeleteMessage' to motherclass(es). [Composition]" และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message Name: <ชื่อข้อความ>"

2.8 รายงานว่า "--not pass: There're message(s) calling motherclass through subclass after it was destroyed. [Dependency]" และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message Name: <ชื่อข้อความ>"

2.9 รายงานว่า "--not pass: There're message(s) calling motherclass or subclass after it was destroyed. [Composition]" และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message Name: <ชื่อข้อความ>"

2.10 มี 3 ส่วนย่อยดังนี้

- รายงานว่า "--not pass: There're message(s) that don't appear in Class Method." และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message name: <ชื่อข้อความ>"

- รายงานว่า "--not pass: There're message(s) that call to worng direction." และบอกรายละเอียดดังนี้ "<--error code: SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message name: <ชื่อข้อความ>"

- รายงานว่า "--not pass: There're message(s) that call those have no permission." และบอกรายละเอียดดังนี้ "<--error code: error type: <", Error[0], "> SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message name: <ชื่อข้อความ>"

2.11 รายงานว่า "--not pass: There're variable(s) in message mismatch in Class Method." และบอกรายละเอียดดังนี้ "<--error code: error type: <ประเภทของข้อผิดพลาด> SQD Name: <ชื่อแผนภาพลำดับ> Message ID: <หมายเลขข้อความ> Message name: <ชื่อข้อความ> Parameter type: <ประเภทพารามิเตอร์> Parameter Name: <ชื่อพารามิเตอร์>"

2.12 รายงานว่า "--not pass: There're variable(s) guard in message mismatch in Class Attribute or grobal variable." และบอกรายละเอียดดังนี้ "<--error code: SQD Name:

<ชื่อแผนภาพลำดับ> Message Name: <ชื่อข้อความ> Guard name: <ชื่อเงื่อนไข> Guard type: <ประเภทเงื่อนไข>"

2.13 รายงานว่า "---not pass: There're message(s) recursion between two Sequence Diagrams." และบอกรายละเอียดดังนี้ "<--error code: SQD 1 Name: <ชื่อแผนภาพลำดับ> Message 1 ID: <หมายเลขข้อความ> Message 1 Name: <ชื่อข้อความ> SQD 2 Name: <ชื่อแผนภาพลำดับ> Message 1 ID: <หมายเลขข้อความ> Message 2 Name: <ชื่อข้อความ>"

ภาคผนวก ค

ตัวอย่างของรูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอ

1. รูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพคลาส

รูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพคลาสที่ส่งออกจากโปรแกรมวาดแผนภาพยูเอ็มแอล Modelio มีลักษณะเป็นกลุ่มแท็กของคลาส (แท็ก packageElement ที่มี คุณลักษณะ xmi:type = "uml:Class") และความสัมพันธ์ระหว่างคลาสต่าง ๆ (แท็ก packageElement ที่มีคุณลักษณะ xmi:type = "uml:Association") โดยกลุ่มแท็กของคลาสจะมีกลุ่มแท็กย่อยเป็นกลุ่มแท็กคุณลักษณะ (แท็ก ownedAttribute) และกลุ่มแท็กเมธอด (แท็ก ownedOperation) ส่วนกลุ่มแท็กความสัมพันธ์จะมีกลุ่มแท็กย่อยบอกประเภทความสัมพันธ์และรหัสคลาสที่มีความสัมพันธ์กัน ดังรูปที่ ค.1

```
<packagedElement xmi:type="uml:Class" xmi:id="รหัสคลาส" name="ชื่อคลาส">
  <ownedAttribute xmi:type="uml:Property" xmi:id="รหัสคุณลักษณะ" name="ชื่อคุณลักษณะ" visibility="การเข้าถึง">
    <type xmi:type="uml:PrimitiveType" href="อ้างอิงประเภทพารามิเตอร์จากโอเอ็มจี"/>
  </ownedAttribute>
  <ownedOperation xmi:type="uml:Operation" xmi:id="รหัสเมธอด" name="ชื่อเมธอด" visibility="การเข้าถึง">
    <ownedParameter xmi:id="รหัสพารามิเตอร์" name="ชื่อพารามิเตอร์">
      <type xmi:type="uml:PrimitiveType" href="อ้างอิงประเภทพารามิเตอร์จากโอเอ็มจี"/>
    </ownedParameter>
    <ownedParameter xmi:id="รหัสพารามิเตอร์" type="ประเภทพารามิเตอร์แบบกำหนดเอง" direction="return">
    </ownedParameter>
  </ownedOperation>
</packagedElement>
<packagedElement xmi:type="uml:Association" xmi:id="รหัสความสัมพันธ์" memberEnd="รหัสความสัมพันธ์ต้น
รหัสความสัมพันธ์ปลาย">
  <ownedEnd xmi:type="uml:Property" xmi:id="รหัสความสัมพันธ์ต้น" type="รหัสคลาส1" aggregation=
  "ประเภทความสัมพันธ์" association="รหัสความสัมพันธ์"/>
  <ownedEnd xmi:type="uml:Property" xmi:id="รหัสความสัมพันธ์ต้น" type="รหัสคลาส2" navigable="yes"
  association="รหัสความสัมพันธ์"/>
</packagedElement>
```

รูปที่ ค.1 โครงสร้างไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพคลาส

2. รูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพลำดับ

รูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพลำดับที่ส่งออกจากโปรแกรมวาดแผนภาพยูเอ็มแอล Modelio มีลักษณะเป็นแท็กใหญ่ 1 กลุ่ม (แท็ก packageElement ที่มี คุณลักษณะxmi:type = “uml:Interaction”) มีกลุ่มแท็กย่อย แบ่งเป็น 4 กลุ่มใหญ่ ๆ ได้แก่

2.1 กลุ่มแท็ก locals (แท็ก nestedClassifier) เป็นกลุ่มแท็กอธิบายคลาสจากแผนภาพคลาสที่ถูกดึงมาใช้เป็นแอคเตอร์หรือส่วนประกอบของระบบในแผนภาพลำดับ

2.2 กลุ่มแท็กเส้นชีวิต (แท็ก lifeline) เป็นกลุ่มแท็กอธิบายแคดเตอร์หรือส่วนประกอบของระบบ ว่ามีชื่ออะไร อ้างอิงจาก local ตัวไหน และมีรหัสเหตุการณ์อะไรเกิดขึ้นบ้างในเส้นชีวิตเรียงตามลำดับ

2.3 กลุ่มแท็กเหตุการณ์ (แท็ก fragment) เป็นกลุ่มแท็กอธิบายประเภทเหตุการณ์ และบอกว่าเหตุการณ์นั่นอยู่บนเส้นชีวิตใด

2.4 กลุ่มแท็กข้อความ (แท็ก message) เป็นกลุ่มแท็กอธิบายรายละเอียดของข้อความ บอกรหัสข้อความ ชื่อข้อความ ประเภทข้อความ รหัสเหตุการณ์ส่ง รหัสเหตุการณ์รับ และรายละเอียดพารามิเตอร์ต่าง ๆ รวมถึงเงื่อนไขอีกด้วย

รูปแบบไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพลำดับเป็นดังรูปที่ ค.2

```
<packagedElement xmi:type="uml:Interaction" xmi:id="รหัสแผนภาพลำดับ" name="ชื่อแผนภาพลำดับ">
  <nestedClassifier xmi:type="uml:Collaboration" xmi:id="รหัสท้องถิ่น" name="locals">
    <ownedAttribute xmi:type="uml:Property" xmi:id="รหัสส่วนประกอบท้องถิ่น" name="ชื่อส่วนประกอบท้องถิ่น" type=
    "รหัสคลาส"/>
  </nestedClassifier>
  <lifeline xmi:id="รหัสเส้นชีวิต" name="ชื่อส่วนประกอบท้องถิ่น" represents="รหัสส่วนประกอบท้องถิ่น" coveredBy=
  "รหัสเหตุการณ์1 รหัสเหตุการณ์2 รหัสเหตุการณ์3"/>
  <fragment xmi:type="uml:ExecutionOccurrenceSpecification" xmi:id="รหัสเหตุการณ์" name=
  "ExecutionOccurenceSpecification" covered="รหัสเส้นชีวิต"/>
  <fragment xmi:type="uml:BehaviorExecutionSpecification" xmi:id="รหัสเหตุการณ์" name=
  "ExecutionSpecification" covered="รหัสเส้นชีวิต" start="รหัสเส้นชีวิต1" finish="รหัสเส้นชีวิต2"/>
  <fragment xmi:type="uml:MessageOccurrenceSpecification" xmi:id="รหัสเหตุการณ์" name=
  "ExecutionOccurenceSpecification" covered="รหัสเส้นชีวิต" message="รหัสข้อความ"/>
  <message xmi:type="uml:Message" xmi:id="รหัสข้อความ" name="ชื่อข้อความ" messageSort="asynchCall"
  receiveEvent="รหัสเหตุการณ์1" sendEvent="รหัสเหตุการณ์2">
    <argument>
      <value type="ประเภทพารามิเตอร์แบบกำหนดเอง" direction="return"/>
    </argument>
  </message>
  <message xmi:type="uml:Message" xmi:id="รหัสข้อความ" name="ชื่อข้อความ" messageSort="asynchCall"
  receiveEvent="รหัสเหตุการณ์1" sendEvent="รหัสเหตุการณ์2">
    <argument>
      <value name="ชื่อพารามิเตอร์" type="อ้างอิงประเภทพารามิเตอร์จากโอเอ็มจี" direction="return"/>
    </argument>
  </message>
  <message xmi:type="uml:Message" xmi:id="รหัสข้อความ" name="ชื่อข้อความ" messageSort="asynchCall"
  receiveEvent="รหัสเหตุการณ์1" sendEvent="รหัสเหตุการณ์2">
    <argument>
      <value name="ชื่อพารามิเตอร์1" type="อ้างอิงประเภทพารามิเตอร์จากโอเอ็มจี"/>
      <value name="ชื่อพารามิเตอร์2" type="ประเภทพารามิเตอร์แบบกำหนดเอง"/>
      <value name="ชื่อตัวแปรคืนค่า" type="อ้างอิงประเภทพารามิเตอร์จากโอเอ็มจี" direction="return"/>
    </argument>
  </message>
</packagedElement>
```

รูปที่ ค.2 โครงสร้างไฟล์ข้อมูลเอ็กซ์เอ็มไอของแผนภาพลำดับ

3. รูปแบบข้อมูลเอ็กซ์เอ็มไอของตัวแปรกำหนดเอง

โอเอ็มจีกำหนดประเภทของพารามิเตอร์ไว้ 5 ประเภทได้แก่ Boolean, Interger, Real, String, UnlimitedNatural ดังรูปที่ ค.3 หากผู้ใช้ต้องการใช้พารามิเตอร์ที่นอกเหนือจากนี้ จะต้องกำหนดข้อมูลพารามิเตอร์กำหนดเองในตอนท้ายของไฟล์ เพื่อให้แผนภาพต่าง ๆ ใช้ร่วมกันดังมีรูปแบบดังรูปที่ ค.4

```
<xmi:XMI xmlns:xmi="http://www.omg.org/spec/XMI/20110701" xmlns:uml=
"http://www.omg.org/spec/UML/20110701" xmlns:mofext="http://www.omg.org/spec/MOF/20110701">
  <uml:Package xmi:type="uml:Package" xmi:id="_0" name="PrimitiveTypes" URI=
  "http://www.omg.org/spec/PrimitiveTypes/20110701">
    <packagedElement xmi:type="uml:PrimitiveType" xmi:id="Boolean" name="Boolean">
      <ownedComment xmi:type="uml:Comment" xmi:id="Boolean-_ownedComment.0" annotatedElement=
      "Boolean">
        <body>A Boolean type is used for logical expression, consisting of the predefined values
        true and false.</body>
      </ownedComment>
    </packagedElement>
    <packagedElement xmi:type="uml:PrimitiveType" xmi:id="Integer" name="Integer">
      <ownedComment xmi:type="uml:Comment" xmi:id="Integer-_ownedComment.0" annotatedElement=
      "Integer">
        <body>An integer is a primitive type representing integer values.</body>
      </ownedComment>
    </packagedElement>
    <packagedElement xmi:type="uml:PrimitiveType" xmi:id="Real" name="Real">
      <ownedComment xmi:type="uml:Comment" xmi:id="Real-_ownedComment.0" annotatedElement="Real">
        <body>A real is a primitive type representing the mathematical concept of real.</body>
      </ownedComment>
    </packagedElement>
    <packagedElement xmi:type="uml:PrimitiveType" xmi:id="String" name="String">
      <ownedComment xmi:type="uml:Comment" xmi:id="String-_ownedComment.0" annotatedElement=
      "String">
        <body>A string is a sequence of characters in some suitable character set used to
        display information about the model. Character sets may include non-Roman alphabets and
        characters.</body>
      </ownedComment>
    </packagedElement>
    <packagedElement xmi:type="uml:PrimitiveType" xmi:id="UnlimitedNatural" name=
    "UnlimitedNatural">
      <ownedComment xmi:type="uml:Comment" xmi:id="UnlimitedNatural-_ownedComment.0"
      annotatedElement="UnlimitedNatural">
        <body>An unlimited natural is a primitive type representing unlimited natural values.
        </body>
      </ownedComment>
    </packagedElement>
  </uml:Package>
  <mofext:Tag xmi:type="mofext:Tag" xmi:id="_2" name="org.omg.xmi.nsPrefix" value="primitives"
  element="_0"/>
  <mofext:Tag xmi:type="mofext:Tag" xmi:id="_3" name="org.omg.xmi.schemaType" value=
  "http://www.w3.org/2001/XMLSchema#boolean" element="Boolean"/>
  <mofext:Tag xmi:type="mofext:Tag" xmi:id="_4" name="org.omg.xmi.schemaType" value=
  "http://www.w3.org/2001/XMLSchema#integer" element="Integer"/>
  <mofext:Tag xmi:type="mofext:Tag" xmi:id="_5" name="org.omg.xmi.schemaType" value=
  "http://www.w3.org/2001/XMLSchema#double" element="Real"/>
</xmi:XMI>
```

รูปที่ ค.3 พารามิเตอร์ทั้ง 5 ประเภทที่โอเอ็มจีกำหนดไว้

```
<packagedElement xmi:type="uml:PrimitiveType" xmi:id="รหัสพารามิเตอร์แบบกำหนดเอง" name="Void"/>
<packagedElement xmi:type="uml:PrimitiveType" xmi:id="รหัสพารามิเตอร์แบบกำหนดเอง" name="Float"/>
```

รูปที่ ค.4 รูปแบบการกำหนดประเภทพารามิเตอร์กำหนดเอง

ประวัติผู้เขียนวิทยานิพนธ์

นายวรวุฒิ ประสิทธิวุฒิศักดิ์ เกิดเมื่อวันอาทิตย์ที่ 20 มิถุนายน พ.ศ. 2525 ที่จ.น่าน สำเร็จการศึกษาระดับประถมศึกษาจากโรงเรียนบ้านดอน (ศรีเสริมกสิกร) จ.น่าน ในปีการศึกษา 2537 สำเร็จการศึกษาระดับมัธยมศึกษาจากโรงเรียนศรีสวัสดิ์วิทยาคาร จ.น่าน ในปีการศึกษา 2543 สำเร็จการศึกษาระดับวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมสำรวจ จากคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2548 แล้วได้เข้าทำงานกับบริษัทสเปเชียลไดเมนชันโซลูชัน จำกัดเป็นเวลา 3 ปี แล้วจึงได้เข้ามาศึกษาต่อหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2552