การสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิส

Syntax-based Test Case Generation for Web Services Testing

จุฑาพร อินทะสะระ

Jutarporn Intasara

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญา
วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science
Prince of Songkla University
2554

ชื่อวิทยานิพนธ์ การสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิส
ผู้เขียน นางสาวจุฑาพร อินทะสะระ
สาขาวิชา วิทยาการคอมพิวเตอร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	**คณะกรรมการสอบ**
.....	.....ประธานกรรมการ
(ดร.สุภาภรณ์ กานต์สมเกียรติ)	(ผู้ช่วยศาสตราจารย์ ดร.กฤดาภัทร สีหารี)
	.....กรรมการ
	(ดร.สุภาภรณ์ กานต์สมเกียรติ)
	.....กรรมการ
	(ผู้ช่วยศาสตราจารย์ ดร.อำนาจ เปาะทอง)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

.....
(ศาสตราจารย์ ดร.อมรรัตน์ พงศ์ดารา)
คณบดีบัณฑิตวิทยาลัย

ชื่อวิทยานิพนธ์ การสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิส
ผู้เขียน นางสาวจุฑาพร อินทะสะระ
สาขาวิชา วิทยาการคอมพิวเตอร์
ปีการศึกษา 2553

บทคัดย่อ

ปัจจุบันเทคโนโลยีเว็บเซอร์วิสได้ถูกใช้กันอย่างแพร่หลาย ทั้งนี้เนื่องจากความยืดหยุ่นในการพัฒนาระบบและความอิสระจากรูปแบบของซอฟต์แวร์ การทดสอบเป็นกระบวนการที่ใช้ประเมินความถูกต้องและช่วยตัดสินใจในการเลือกใช้บริการ แต่เนื่องจากองค์ประกอบต่างๆ ของเว็บเซอร์วิสมักถูกสงวนไว้เป็นของผู้ขาย ดังนั้นโปรแกรมต้นฉบับจึงมิได้ถูกจัดเตรียมไว้ให้เพื่อใช้ในการออกแบบและสร้างกรณีทดสอบ ผู้ใช้มีเพียงเอกสาร WSDL ที่อธิบายถึงการเรียกใช้บริการเท่านั้น วิทยานิพนธ์นี้จึงได้เสนอการออกแบบและสร้างกรณีทดสอบโดยใช้เอกสาร WSDL โดยที่เอกสาร WSDL จะถูกแปลงให้อยู่ในรูปของแบบจำลองเชิงไวยากรณ์ซึ่งจะใช้เพื่อสร้างกรณีทดสอบ ผลลัพธ์ที่ได้จากกรณีศึกษาแสดงให้เห็นว่ากรณีทดสอบที่สร้างตามหลักการที่ได้นำเสนอสามารถตรวจสอบการทำงานที่ผิดพลาดของเว็บเซอร์วิสได้

Thesis Title	Syntax-based Test Case Generation for Web Services Testing
Author	Miss. Jutarporn Intasara
Major Program	Computer Science
Academic Year	2010

ABSTRACT

Nowadays, web service technologies are becoming more widely used because of the flexibility of developing the systems and the ability to have platform-independent software. Testing evaluates the correctness and also helps to decide how to use web service. Because web service components often belong to third party vendors, source programs are seldom available to use in test case design and generation. Users have only WSDL documents, which describe how to invoke the web service. Thus, this thesis describes the application of WSDL documents to design and generate test cases. The document is converted into a syntax-based model, which is used to create test cases. The results from an experiment exhibited how test cases generated by this method detected several web service errors.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยความช่วยเหลือและสนับสนุนจากบุคคลหลายฝ่าย ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณอย่างสูง คือ

ดร.สุภาภรณ์ กานต์สมเกียรติ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่กรุณาให้คำปรึกษาแนะนำ และช่วยเหลือในการแก้ปัญหาต่างๆ ให้แก่ผู้วิจัยเสมอมา พร้อมทั้งตรวจทานและแก้ไขวิทยานิพนธ์ให้แก่ผู้วิจัย

ผู้ช่วยศาสตราจารย์ ดรกฤ.ดาภัทร สีหารี ประธานกรรมการสอบวิทยานิพนธ์ ที่กรุณาช่วยตรวจทานและแก้ไขวิทยานิพนธ์ให้มีความสมบูรณ์

ผู้ช่วยศาสตราจารย์ ดร.อำนาจ เปาะทอง กรรมการในการสอบวิทยานิพนธ์ ที่กรุณาให้คำปรึกษาแนะนำในการทำวิจัย พร้อมทั้งตรวจทานและแก้ไขวิทยานิพนธ์ให้มีความสมบูรณ์

อาจารย์ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ทุกท่านที่ให้ความรู้ทางด้านวิชาการ ซึ่งสามารถนำมาใช้ในการทำวิทยานิพนธ์ได้อย่างดียิ่ง

เจ้าหน้าที่ภาควิชาวิทยาการคอมพิวเตอร์ และเจ้าหน้าที่บัณฑิตวิทยาลัยทุกท่านที่ให้ความช่วยเหลือ และอำนวยความสะดวกเกี่ยวกับเอกสารต่างๆ

เพื่อนๆ พี่ๆ และน้องๆ ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ ที่ให้คำปรึกษา และช่วยเหลือในการทำวิทยานิพนธ์

คุณพ่อ คุณแม่ และครอบครัว ที่ให้การสนับสนุนคอยเป็นห่วงสุขภาพและให้กำลังใจแก่ผู้วิจัยมาโดยตลอด

ผู้วิจัยขอขอบคุณทุกท่านเป็นอย่างสูงมา ณ โอกาสนี้

จุฑาพร อินทะสะระ

สารบัญ

หน้า

สารบัญ ..... (6)
รายการตาราง ..... (9)
รายการภาพประกอบ ..... (10)
บทที่
1 บทนำ 1
1.1 ความสำคัญและที่มาของงานวิจัย 1
1.2 งานวิจัยที่เกี่ยวข้อง 2
1.3 วัตถุประสงค์ของงานวิจัย 2
1.4 ขอบเขตการดำเนินงานของการวิจัย 3
1.5 ขั้นตอนและระยะเวลาการดำเนินงาน 3
1.5.1 ขั้นตอนการดำเนินงาน 3
1.5.2 ระยะเวลาการดำเนินงาน 3
1.5.3 แผนการดำเนินการวิจัย 3
1.6 สถานที่และเครื่องมือที่ใช้ 4
1.6.1 สถานที่ 4
1.6.2 เครื่องมือที่ใช้ 5
1.7 ประโยชน์ที่คาดว่าจะได้รับ 5
2 ทฤษฎีที่เกี่ยวข้อง 6
2.1 สถาปัตยกรรมเชิงบริการ 6
2.1.1 องค์ประกอบพื้นฐานของสถาปัตยกรรมเชิงบริการ 6
2.2 เทคโนโลยีเว็บเซอร์วิส 7
2.2.1 มาตรฐานที่เกี่ยวข้องกับเว็บเซอร์วิส 7
2.3 XML Schema 10
2.3.1 ประเภทของอิลิเมนต์ใน XML Schema 10
2.4 BNF 14
2.4.1 ไวยากรณ์ของ BNF 15
2.5 การทดสอบซอฟต์แวร์ 16
2.5.1 เทคนิคการทดสอบซอฟต์แวร์ 16
2.5.2 การครอบคลุมการทดสอบ 16

สารบัญ (ต่อ)

หน้า

3 ขั้นตอนการดำเนินการวิจัย 18
3.1 กรอบแนวคิดการสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสจากเอกสาร WSDL 18
3.2 ขั้นตอนการสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสจากเอกสาร WSDL 19
4 การออกแบบและพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบ 33
4.1 การออกแบบเครื่องมือสำหรับสร้างกรณีทดสอบ 33
4.2 การพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบ 36
4.2.1 ขั้นตอนการนำไฟล์เข้าสู่ระบบ 36
4.2.2 ขั้นตอนการสร้างแบบจำลอง 36
4.2.3 ขั้นตอนการปรับปรุงแบบจำลอง 41
4.2.4 ขั้นตอนการสร้างกรณีทดสอบ 43
4.3 การออกแบบส่วนติดต่อผู้ใช้ 45
4.3.1 ส่วนการเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นเข้าสู่ระบบ 45
4.3.2 ส่วนการสร้างแบบจำลองเชิงไวยากรณ์และกรณีทดสอบ 46
4.3.3 ส่วนการออกจากระบบ 48
5 การทดสอบเว็บเซอร์วิส 50
5.1 การกำหนดค่าที่ใช้ทดสอบของกรณีทดสอบ 50
5.1.1 การแทนค่าที่ถูกต้อง 50
5.1.2 การแทนค่าที่ไม่ถูกต้อง 52
5.2 เว็บเซอร์วิสที่ใช้ในการทดสอบ 52
5.2.1 เว็บเซอร์วิส ConvertTemperature 52
5.2.2 เว็บเซอร์วิส MathService 53
5.3 แนวทางการทดสอบ 53
5.4 การสร้างกรณีทดสอบตามวิธีการที่นำเสนอ 53
5.5 การสร้างกรณีทดสอบตามวิธีการของ Samer 59
5.5 ผลการทดสอบ 60
6 บทสรุปและข้อเสนอแนะ 62
6.1 บทสรุป 62

สารบัญ (ต่อ)

หน้า

6.2 ปัญหาและอุปสรรค 62
6.3 ข้อเสนอแนะและงานในอนาคต 63
บรรณานุกรม 64
ภาคผนวก 66
ก ผลงานตีพิมพ์ในการประชุมวิชาการ JCSSE 2010 67
ข ผลงานตีพิมพ์ในการประชุมวิชาการ NCCIT 2010 74
ประวัติผู้เขียน 81

รายการตาราง

ตาราง **หน้า**

1.1 ระยะเวลาการดำเนินการวิจัย 4
2.1 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย sequence และเอกสาร XML ที่สอดคล้องกับ XML Schema 11
2.2 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย all และเอกสาร XML ที่สอดคล้องกับ XML Schema 12
2.3 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย choice และเอกสาร XML ที่สอดคล้องกับ XML Schema 12
2.4 XML Schema ที่กำหนดจำนวนครั้งของการเกิดอิลิเมนต์ด้วยแอตทริบิวต์ minOccurs และแอตทริบิวต์ maxOccurs และเอกสาร XML ที่สอดคล้องกับ XML Schema 13
2.5 ชื่อและความหมายของเงื่อนไขจำกัด 14
2.6 ตัวอย่างการนิยามสัญลักษณ์ไม่สิ้นสุดชื่อ number 15
3.1 สัญลักษณ์และความหมายของสัญลักษณ์สำหรับแบบจำลองเชิงไวยากรณ์ 24
3.2 ตารางเงื่อนไขจำกัดของอิลิเมนต์ชื่อ year ของการดำเนินการ booksRequest 25
4.1 คำอธิบายยูสเคสนำไฟล์เข้าสู่ระบบ 34
4.2 คำอธิบายยูสเคสสร้างแบบจำลอง 35
4.3 คำอธิบายยูสเคสปรับปรุงแบบจำลอง 35
4.4 คำอธิบายยูสเคสสร้างกรณีทดสอบ 35
5.1 ค่าสูงสุดและค่าต่ำสุดของชนิดข้อมูลแบบตัวเลข 51
5.2 ตารางเงื่อนไขจำกัดซึ่งเป็นเงื่อนไขจำกัดกลุ่มของค่า 51
5.3 เงื่อนไขจำกัดของการดำเนินการ ConvertTemp 55
5.4 ค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของการดำเนินการ ConvertTemp 56
5.5 ค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของการดำเนินการ Divide 58
5.6 ค่าขอบเขตของแต่ละพารามิเตอร์ของการดำเนินการ ConvertTemp ตามวิธีการของ Samer 59
5.7 ค่าขอบเขตของแต่ละพารามิเตอร์ของการดำเนินการ Divide ตามวิธีการของ Samer 60
5.8 จำนวนกรณีทดสอบทั้งหมดที่ถูกสร้างขึ้นและจำนวนกรณีทดสอบที่สามารถจับข้อผิดพลาดของเว็บเซอร์วิสด้วยกรณีทดสอบจากวิธีที่นำเสนอและวิธีของ Samer 61

รายการภาพประกอบ

ภาพประกอบ **หน้า**

2.1 องค์ประกอบของเว็บเซอร์วิส 7
2.2 เอกสาร XML ของ book 8
2.3 โครงสร้างของ SOAP Message 9
2.4 โครงสร้างของเอกสาร WSDL 9
2.5 ตัวอย่าง BNF 15
3.1 ขั้นตอนการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิสจากเอกสาร WSDL 18
3.2 โครงสร้างของอินเตอร์มีเดียทโอเปอร์เรชั่น 19
3.3 แผนภาพกิจกรรมการสร้างอินเตอร์มีเดียทโอเปอร์เรชั่น 20
3.4 เอกสาร WSDL ของเว็บเซอร์วิส MathService 22
3.5 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Add 23
3.6 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ booksRequest 25
3.7 แผนภาพกิจกรรมของการสร้างแบบจำลองเชิงไวยากรณ์ 27
3.8 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Add 29
3.9 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Add ที่ปรับปรุงแล้ว 29
3.10 การสร้างกรณีทดสอบของการดำเนินการ Add ด้วยแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้ว 30
3.11 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน * 31
3.12 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน ? 31
3.13 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน + ... 32
3.14 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการสลับลำดับ 32
4.1 แผนภาพการทำงานของเครื่องมือสำหรับสร้างกรณีทดสอบ 34
4.2 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ persons 36
4.3 โครงสร้างต้นไม้ของอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ persons 37
4.4 โครงสร้างอาเรย์ที่ใช้เก็บข้อมูลเงื่อนไขของแบบจำลองเชิงไวยากรณ์ 37
4.5 ขั้นตอนการเก็บข้อมูลเงื่อนไขของแบบจำลองเชิงไวยากรณ์ 38
4.6 ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์ 39
4.7 โครงสร้างลิงค์ลิสต์ที่ใช้เก็บข้อมูลแบบจำลองเชิงไวยากรณ์ 41
4.8 ขั้นตอนการปรับปรุงแบบจำลองเชิงไวยากรณ์ 42
4.9 ขั้นตอนการสร้างกรณีทดสอบ 44

รายการภาพประกอบ (ต่อ)

ภาพประกอบ **หน้า**

4.10 โครงสร้างลิงค์ลิสต์ที่ใช้เก็บกรณีทดสอบ 44
4.11 หน้าจอเริ่มต้นของระบบ 45
4.12 หน้าต่างเลือกอินเตอร์มีเดียทโอเปอร์เรชั่น 46
4.13 แท็บ Select File แสดงตำแหน่งของอินเตอร์มีเดียทโอเปอร์เรชั่นที่เลือกเข้าสู่ระบบ 46
4.14 แท็บ Generate Model แสดงแบบจำลองเชิงไวยากรณ์ 47
4.15 แท็บ Optimize Model แสดงแบบจำลองเชิงไวยากรณ์ที่ผ่านการปรับปรุงแล้ว 47
4.16 แท็บ Generate Test Case แสดงกรณีทดสอบและเงื่อนไขของกรณีทดสอบ 48
4.17 หน้าจอการออกจากระบบ 49
4.18 หน้าจอยืนยันการออกจากระบบ 49
5.1 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ ConvertTemp 54
5.2 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp 55
5.3 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp ที่ปรับปรุงแล้ว 55
5.4 การสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp ที่ปรับปรุงแล้ว 56
5.5 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Divide 57
5.6 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide 57
5.7 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide ที่ปรับปรุงแล้ว 58
5.8 การสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide ที่ปรับปรุงแล้ว 58

บทที่ 1

บทนำ

1.1 ความสำคัญและที่มาของงานวิจัย

ในปัจจุบันการดำเนินกิจกรรมต่างๆ ขององค์กรย่อมอาศัยเทคโนโลยีใหม่ๆ เข้ามาเป็นตัวช่วยขับเคลื่อนให้การดำเนินงานเป็นไปอย่างรวดเร็ว โดยการนำเทคโนโลยีที่มีอยู่มาใช้พัฒนาระบบขององค์กรให้มีความทันสมัยมากยิ่งขึ้น ดังนั้นยิ่งเทคโนโลยีมีความทันสมัยมากเท่าไรความแตกต่างของระบบก็ยิ่งมีมากขึ้นเท่านั้น ด้วยเหตุนี้จึงเป็นเรื่องยากหากในอนาคตต้องการนำระบบขององค์กรมารวมกันเป็นระบบเดียวกัน จากปัญหาที่เกิดขึ้นทำให้เกิดแนวคิดที่เรียกว่าสถาปัตยกรรมเชิงบริการ (Service-Oriented Architecture: SOA) เข้ามาช่วยในการแก้ไขปัญหาดังกล่าว โดยในแนวคิดของสถาปัตยกรรมเชิงบริการจะมองว่าระบบซอฟต์แวร์ดำเนินการบนพื้นฐานของบริการ

เว็บเซอร์วิส (Web Services) เป็นเทคโนโลยีหนึ่งที่นำมาใช้ในการพัฒนาตามแนวคิดสถาปัตยกรรมเชิงบริการโดยใช้มาตรฐานต่างๆ ที่สอดคล้องกับแนวคิดดังกล่าว และในปัจจุบันองค์กรต่างๆ มีการนำเว็บเซอร์วิสมาใช้กันเป็นจำนวนมากซึ่งมีทั้งการเรียกใช้บริการจากภายในและภายนอกองค์กร ดังนั้นความถูกต้องของผลลัพธ์จากการทำงานของบริการจึงเป็นสิ่งสำคัญ แต่ในการเรียกใช้บริการจะไม่มีโปรแกรมต้นฉบับ (Source Program) ของบริการมาให้ มีเพียงเอกสาร WSDL (Web Services Description Language) ที่ได้รับมาจากการค้นหาบริการซึ่งบอกเฉพาะรายละเอียดของบริการและวิธีการเชื่อมต่อเท่านั้น ทำให้ไม่สามารถสร้างกรณีทดสอบโดยวิธีการทั่วไป ซึ่งนิยมสร้างกรณีทดสอบจากโปรแกรมต้นฉบับหรือสร้างกรณีทดสอบจากเอกสารข้อกำหนดความต้องการ (Requirements Specification) ดังนั้นในงานวิจัยนี้จึงได้เสนอวิธีการสร้างกรณีทดสอบสำหรับทดสอบเว็บเซอร์วิสโดยใช้เอกสาร WSDL โดยสร้างแบบจำลองเชิงไวยากรณ์แสดงลักษณะของข้อมูลที่ใช้ในการทดสอบ จากนั้นกรณีทดสอบจะถูกสร้างจากแบบจำลองเชิงไวยากรณ์ที่สร้างขึ้น

1.2 งานวิจัยที่เกี่ยวข้อง

WSDL-Based Automatic Test Case Generation for Web Services Testing (Xiaoying, *et al.*, 2005)

งานวิจัยนี้เสนอวิธีการสร้างกรณีทดสอบจากเอกสาร WSDL แบบอัตโนมัติโดยการแปลง XML Schema ให้อยู่ในรูปโครงสร้างต้นไม้ และสร้างกรณีทดสอบจากชนิดข้อมูลใน XML Schema และการเรียงลำดับของอิลิเมนต์ จากนั้นวิเคราะห์การขึ้นต่อกันของข้อมูล (Data Dependency) ของตัวดำเนินการในเอกสาร WSDL กรณีทดสอบที่ได้อยู่ในรูปของเอกสาร XML ที่เรียกว่า Service Test Specification (STS)

An Approach for Specification-based Test Case Generation for Web Services (Samer and Malcolm, 2007)

งานวิจัยนี้นำเสนอวิธีการสร้างกรณีทดสอบสำหรับทดสอบเว็บเซอร์วิสแบบอัตโนมัติจาก XML Schema ของเอกสาร WSDL โดยแปลงให้อยู่ในรูปของโครงสร้างต้นไม้ จากนั้นทำการสร้างกรณีทดสอบโดยใช้การวิเคราะห์ค่าขอบเขต (Boundary Value Analysis: BVA) ของโหนดต่างๆ ในโครงสร้างต้นไม้ และกรณีทดสอบที่ได้อยู่ในรูปของเอกสาร XML

WSDL-Based Automated Test Data Generation for Web Services (Chunyan, *et al.*, 2008)

งานวิจัยนี้เสนอวิธีการสร้างกรณีทดสอบสำหรับทดสอบเว็บเซอร์วิสแบบอัตโนมัติจากโครงสร้างต้นไม้ที่แปลงจาก XML Schema ในเอกสาร WSDL และสร้างกรณีทดสอบโดยพิจารณาจาก (1) ค่าสูงสุดและค่าต่ำสุดของชนิดข้อมูลต่างๆ (2) จำนวนครั้งของการเกิดอิลิเมนต์ที่น้อยที่สุด (minOccurs) และจำนวนครั้งของการเกิดอิลิเมนต์ที่มากที่สุด (maxOccurs) แล้วสร้างเป็นชุดทดสอบซึ่งจัดลำดับของ อิลิเมนต์ย่อยตามคำสำคัญ (Keyword) คือ sequence choice และ all

1.3 วัตถุประสงค์ของงานวิจัย

1.3.1 เพื่อเสนอวิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส

1.3.2 เพื่อพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบโดยใช้แบบจำลองเชิงไวยากรณ์

1.4 ขอบเขตการดำเนินงานของการวิจัย

1.4.1 สร้างแบบจำลองเชิงไวยากรณ์จากเอกสาร WSDL โดยพิจารณาส่วนของ XML Schema

1.4.2 เสนอวิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส

1.4.3 พัฒนาเครื่องมือสร้างกรณีทดสอบโดยใช้แบบจำลองเชิงไวยากรณ์ที่นำเสนอ

1.5 ขั้นตอนและระยะเวลาการดำเนินงาน

1.5.1 ขั้นตอนการดำเนินงาน

1) ศึกษางานวิจัยและเอกสารที่เกี่ยวข้อง
2) วิเคราะห์และออกแบบแบบจำลองเชิงไวยากรณ์จากเอกสาร WSDL
3) พัฒนาและปรับปรุงเครื่องมือสำหรับสร้างกรณีทดสอบ
4) ทดสอบเพื่อประเมินประสิทธิภาพของกรณีทดสอบที่ได้จากแบบจำลองเชิงไวยากรณ์
5) เขียนบทความวิจัยและเผยแพร่
6) จัดทำรายงานการวิจัย

1.5.2 ระยะเวลาการดำเนินงาน

มิถุนายน 2552 – พฤษภาคม 2554

1.5.3 แผนการดำเนินการวิจัย

ระยะเวลาการดำเนินการวิจัยแสดงดังตารางที่ 1.1

ตารางที่ 1.1 ระยะเวลาการดำเนินการวิจัย

ขั้นตอนการดำเนินงาน	เดือน											
	2552	2553						2554				
	6-12	1-2	3-4	5-6	7-8	9-10	11-12	1	2	3	4	5
1) ศึกษางานวิจัยและเอกสารที่เกี่ยวข้อง	←	—	—	—	—	—	→					
2) วิเคราะห์และออกแบบแบบจำลองเชิงไวยากรณ์จากเอกสาร WSDL	←	→										
3) พัฒนาและปรับปรุงเครื่องมือสำหรับสร้างกรณีทดสอบ				←	—	—	—	—	—	→		
4) ทดสอบเพื่อประเมินประสิทธิภาพของกรณีทดสอบที่ได้จากแบบจำลองเชิงไวยากรณ์							←	—	—	→		
5) เขียนบทความวิจัยและเผยแพร่		←	—	—	→							
6) จัดทำรายงานการวิจัย		←	—	—	—	—	—	—	—	—	—	→

1.6 สถานที่และเครื่องมือที่ใช้

1.6.1 สถานที่

ห้องวิจัยกลุ่มวิศวกรรมซอฟต์แวร์และงานประยุกต์ ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์

1.6.2 เครื่องมือที่ใช้

1) ด้านฮาร์ดแวร์

เครื่องคอมพิวเตอร์ส่วนบุคคลที่มีคุณลักษณะ ได้แก่ หน่วยประมวลผล Intel Core2Duo 3.0 GHz หน่วยความจำขนาด 2 GB และฮาร์ดดิสก์ความจุ 320 GB สำหรับพัฒนาและทดสอบระบบ

2) ด้านซอฟต์แวร์

2.1) ระบบปฏิบัติการ Microsoft Windows 7

2.2) Java Development Kit Standard Edition 1.6 เป็นภาษาในการพัฒนาระบบ

2.3) เครื่องมือสนับสนุนการทำงาน Netbeans 6.8

1.7 ประโยชน์ที่คาดว่าจะได้รับ

1.7.1 ได้วิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส

1.7.2 ได้เครื่องมือเพื่อใช้ในการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ที่นำเสนอ

บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

เนื้อหาในบทนี้กล่าวถึงทฤษฎีและหลักการต่าง ๆ ในส่วนแรกกล่าวถึงสถาปัตยกรรมเชิงบริการ เทคโนโลยีเว็บเซอร์วิส และ XML Schema ส่วนที่สองกล่าวถึงไวยากรณ์ BNF การทดสอบซอฟต์แวร์ และการครอบคลุมการทดสอบ ซึ่งมีรายละเอียดดังต่อไปนี้

2.1 สถาปัตยกรรมเชิงบริการ (Service-Oriented Architecture: SOA)

สถาปัตยกรรมเชิงบริการ (Mustafa, *et al*., 2010) เป็นรูปแบบของการสร้างโปรแกรมประยุกต์เชิงบริการ (Service-Oriented Applications) ซึ่งมีวัตถุประสงค์เพื่อจัดเตรียมบริการ (Service) ซึ่งสามารถถูกเรียกใช้โดยบริการอื่น บริการในแง่ของสถาปัตยกรรมเชิงบริการจะเป็นฟังก์ชันการทำงานที่สามารถทำงานอย่างอิสระ (Autonomy) ตามขอบเขตหน้าที่การทำงานของบริการที่ถูกกำหนดไว้ โดยมีการประกาศออกสู่ภายนอกและค้นหาผ่านกลไกการค้นหา นอกจากนั้นสถาปัตยกรรมเชิงบริการยังสนับสนุนบริการให้สามารถทำงานร่วมกัน (Interoperability) โดยอาศัยมาตรฐานเปิด (Open Standard) บริการมีการผูกติดกันแบบหลวม (Loose Coupling) ทำให้การปรับเปลี่ยนบริการเป็นไปอย่างอิสระ และสนับสนุนการนำบริการที่มีอยู่กลับมาใช้ใหม่ (Reuse) ได้

2.1.1 องค์ประกอบพื้นฐานของสถาปัตยกรรมเชิงบริการ

สถาปัตยกรรมเชิงบริการมีองค์ประกอบหลัก 3 ส่วน คือ ส่วนผู้ให้บริการ (Service Provider) ส่วนผู้ใช้บริการ (Service User) และส่วนตัวแทนบริการ (Service Broker)

1) ผู้ให้บริการ เป็นเจ้าของบริการ โดยผู้ให้บริการจะทำการประกาศ (Publish) บริการของตนออกสู่ภายนอกโดยการลงทะเบียนบริการนั้นไว้ที่ส่วนตัวแทนบริการ

2) ผู้ใช้บริการ จัดเป็นส่วนที่สำคัญเนื่องจากเป็นผู้เริ่มต้นที่ทำให้เกิดการดำเนินการหลัก 2 อย่างคือค้นหา (Find) และเรียกใช้ (Invoke) โดยหลังจากที่ผู้ใช้บริการได้ทำการค้นหาบริการที่ต้องการแล้ว ผู้ใช้บริการจะเรียกใช้บริการนั้นตามข้อมูลการเรียกใช้บริการที่ได้จากตัวแทนบริการ ซึ่งข้อมูลการเรียกใช้จะบอกให้ทราบถึงที่อยู่ของบริการ วิธีการเรียกใช้บริการ และบริการที่มีให้ใช้

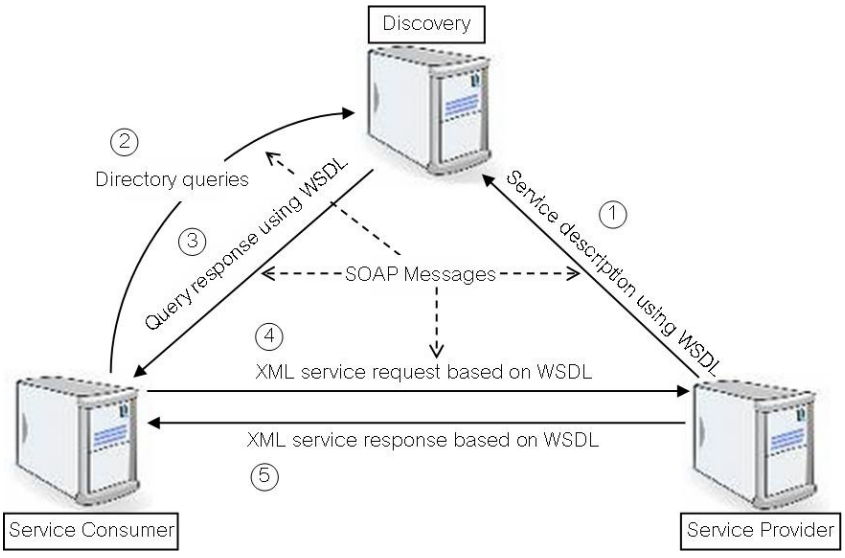
3) ตัวแทนบริการ เปรียบเสมือนกลไกในการค้นหาบริการ โดยอนุญาตให้ผู้ใช้บริการทำการค้นหาบริการที่ตรงตามความต้องการและจัดเตรียมข้อมูลในการเรียกใช้ (Bind) บริการเหล่านั้น

2.2 เทคโนโลยีเว็บเซอร์วิส

เทคโนโลยีที่สนับสนุนแนวคิดสถาปัตยกรรมเชิงบริการมีหลายเทคโนโลยี เช่น เทคโนโลยี CORBA (Common Object Request Broker Adapter) (CORBA, 2011) ซึ่งทำให้ระบบทำงานร่วมกันโดยใช้โปรโตคอลเฉพาะ เทคโนโลยี DCOM (Distributed Component Object Model) (DCOM, 2011) สนับสนุนการเชื่อมต่อระบบที่ใช้ผลิตภัณฑ์ของบริษัทไมโครซอฟต์ และเทคโนโลยี Java RMI (Java Remote Method Invocation) (RMI, 2011) สนันสนุนการเชื่อมต่อระบบที่พัฒนาโดยใช้ภาษาจาวาเท่านั้น เป็นต้น เว็บเซอร์วิสเป็นเทคโนโลยีที่สนับสนุนแนวคิดสถาปัตยกรรมเชิงบริการที่นิยมใช้กันในปัจจุบัน

2.2.1 มาตรฐานที่เกี่ยวข้องกับเว็บเซอร์วิส

เว็บเซอร์วิสใช้มาตรฐานหลายอย่าง ได้แก่ XML (Extensible Markup Language), SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) และ UDDI (Universal Description, Discovery and Integration) ซึ่งมาตรฐานดังกล่าวสอดคล้องกับสถาปัตยกรรมเชิงบริการ แสดงได้ดังภาพประกอบที่ 2.1



ภาพประกอบที่ 2.1 องค์ประกอบของเว็บเซอร์วิส (Douglas, 2003)

จากภาพประกอบผู้ให้บริการทำการประกาศบริการที่ส่วนการลงทะเบียนที่เรียกว่า UDDI โดยใช้เอกสาร WSDL เมื่อผู้ใช้บริการต้องการเรียกใช้บริการใดจะทำการค้นหาบริการนั้นที่ส่วนการลงทะเบียน ผลจากการค้นหาบริการจะได้เอกสาร WSDL ที่บอกรายละเอียดเกี่ยวกับบริการกลับมายังผู้ใช้บริการ จากนั้นผู้ใช้บริการจะเรียกใช้งานบริการโดยทำการติดต่อผู้ให้บริการโดยตรงตามเอกสาร WSDL และผลลัพธ์จากการเรียกใช้บริการจะถูกส่งกลับมายังผู้ใช้บริการซึ่งมีรูปแบบตามที่กำหนดในเอกสาร WSDL เช่นเดียวกัน สำหรับการสื่อสารระหว่างผู้ใช้บริการ ผู้ให้บริการ และส่วนการลงทะเบียนจะอยู่ในรูปของการส่ง SOAP Message ถึงกัน

มาตรฐานที่เกี่ยวข้องกับเว็บเซอร์วิสมีรายละเอียดดังต่อไปนี้

1) XML (W3C, 2008) เป็นมาตรฐานที่กำหนดขึ้นโดย W3C (World Wide Web Consortium) เป็นภาษาที่ถูกออกแบบมาเพื่อใช้ในการอธิบายข้อมูล โดยการกำหนดแท็ก (Tag) ซึ่งมีลักษณะคล้ายกับแท็กในภาษา HTML (Hypertext Markup Language) แต่แท็กของ XML สามารถกำหนดชื่อแท็กได้ตามความต้องการของผู้ใช้ เช่น <book>SOA</book> เป็นต้น เว็บเซอร์วิสกำหนดให้ XML เป็นภาษาที่ใช้ในการแลกเปลี่ยนข้อมูลเนื่องจาก XML เป็นภาษาที่ไม่ขึ้นอยู่กับแพลตฟอร์ม (Platform) ตัวอย่างเอกสาร XML ของ book แสดงดังภาพประกอบที่ 2.2

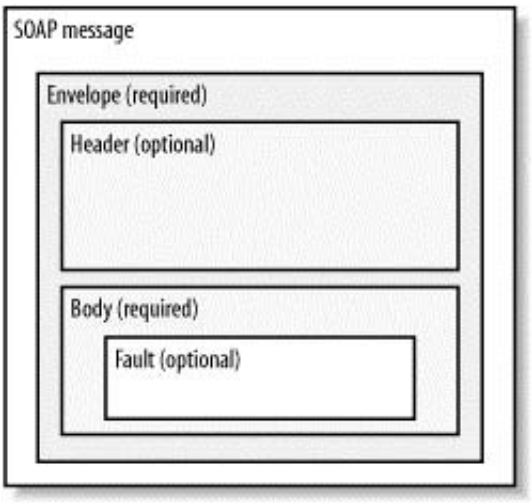
```
<book>
   <ISBN>0-672-32374-5</ISBN>
   <price>59.99</price>
   <year>2002</year>
</book>
```

ภาพประกอบที่ 2.2 เอกสาร XML ของ book

2) SOAP (W3C, 2007) เป็นโปรโตคอลที่ใช้ในการสื่อสารของเว็บเซอร์วิสซึ่งมีคุณสมบัติที่สำคัญคือสามารถสื่อสารผ่านโปรโตคอลสื่อสารใดก็ได้ โปรโตคอลสื่อสารที่ใช้อยู่ในปัจจุบัน เช่น HTTP (Hypertext Transfer Protocol) หรือ SMTP (Simple Mail Transfer Protocol) เป็นต้น แต่ในเว็บเซอร์วิสนิยมใช้ SOAP ร่วมกับ HTTP เนื่องจาก HTTP เป็นโปรโตคอลสำหรับอินเตอร์เน็ตที่ใช้กันอย่างแพร่หลาย

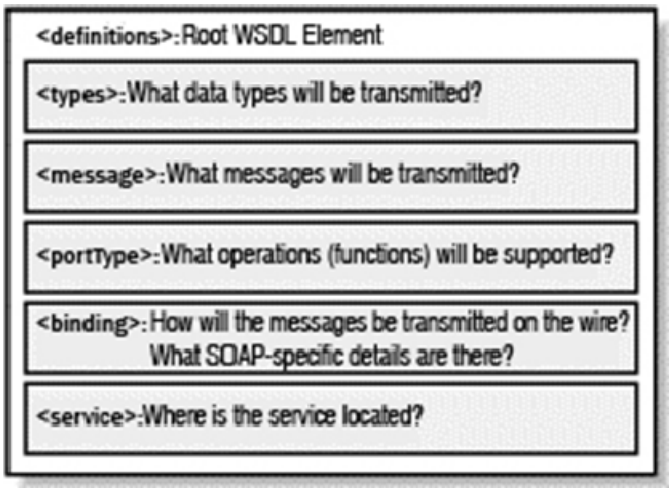
SOAP มีลักษณะเป็นเอกสารคล้ายจดหมายเรียกว่า SOAP Message ซึ่งมีโครงสร้างดังภาพประกอบที่ 2.3 โดยข้อมูลที่ใช้สื่อสารกันอยู่ในรูปของ XML ซึ่งอยู่ภายในส่วนของ SOAP Envelope

SOAP Envelope แบ่งออกเป็น 2 ส่วนคือ (1) ส่วนของ SOAP Header เป็นส่วนที่จะมีหรือไม่มีก็ได้ ใช้สำหรับบรรจุข้อมูลต่างๆ เช่น ข้อมูลเกี่ยวกับผู้ส่งสำหรับใช้ในการตรวจสอบสิทธิ์การเรียกใช้บริการ เป็นต้น และ (2) ส่วนของ SOAP Body ใช้สำหรับบรรจุเนื้อหาของ SOAP Message ซึ่งถือว่าเป็นส่วนสำคัญและต้องเป็นส่วนที่มีเสมอใน SOAP Message



ภาพประกอบที่ 2.3 โครงสร้างของ SOAP Message (Ethan, 2002)

3) WSDL (W3C, 2001) เป็นเอกสารที่เขียนอยู่ในรูปแบบของ XML ใช้สำหรับอธิบายรายละเอียดของเว็บเซอร์วิสว่ามีบริการอะไรให้เรียกใช้บ้างและสามารถเรียกใช้บริการนั้นได้อย่างไร โครงสร้างของเอกสาร WSDL แสดงได้ดังภาพประกอบที่ 2.4 และมีรายละเอียดดังนี้



ภาพประกอบที่ 2.4 โครงสร้างของเอกสาร WSDL (Cerami, 2002)

(1) อิลิเมนต์ <types> เป็นอิลิเมนต์ที่ใช้กำหนดชนิดข้อมูล (Datatype) ของข้อความ (Message) ที่บริการใช้แลกเปลี่ยนระหว่างกัน โดยการกำหนดชนิดข้อมูลจะใช้ XML Schema อธิบายชนิดข้อมูล

(2) อิลิเมนต์ <message> เป็นอิลิเมนต์ที่ใช้อธิบายข้อความในการติดต่อสื่อสารระหว่างบริการ ภายในบรรจุอิลิเมนต์ <part> ซึ่งใช้ระบุข้อมูลนำเข้าซึ่งมีความสัมพันธ์กับข้อมูลที่ถูกกำหนดในอิลิเมนต์ <types>

(3) อิลิเมนต์ <portType> เป็นอิลิเมนต์ที่ใช้กำหนดการดำเนินการ (operation) ต่างๆ โดยการกำหนดอิลิเมนต์ <operation> ไว้ภายในเพื่อบอกรายละเอียดของข้อความที่บริการใช้ในการรับและส่งข้อมูล

(4) อิลิเมนต์ <binding> เป็นอิลิเมนต์ที่ใช้ระบุรายละเอียดของโปรโตคอลและรูปแบบข้อมูลของการดำเนินการและข้อความที่ระบุในอิลิเมนต์ <portType>

(5) อิลิเมนต์ <service> เป็นอิลิเมนต์ที่ใช้รวบรวมกลุ่มของอิลิเมนต์ <port> ที่ระบุตำแหน่งของการ binding ไปยังเว็บเซอร์วิส

4) UDDI (สุธี, 2550) เป็นตัวกลางในการติดต่อสื่อสารระหว่างผู้ให้บริการและผู้ใช้บริการ โดยจัดเตรียมข้อมูลที่จำเป็นต้องใช้ในการติดต่อสื่อสารของทั้งสองฝ่าย โดย UDDI เป็นข้อกำหนดที่ผู้ให้บริการใช้สำหรับลงทะเบียนบริการเพื่อประกาศรายละเอียดของบริการให้ผู้อื่นได้ทราบ ส่วนผู้ใช้บริการจะเข้ามาค้นหาบริการและรายละเอียดต่างๆ ของบริการจาก UDDI

2.3 XML Schema

XML Schema (Carey, 2007) ใช้สำหรับกำหนดโครงสร้างของเอกสาร XML เช่นเดียวกับ DTD (Document Type Definition) แต่นิยมใช้ XML Schema มากกว่า DTD เนื่องจาก XML Schema สามารถรองรับชนิดข้อมูลได้มากกว่า DTD และสามารถสร้างชนิดข้อมูลใหม่จากชนิดข้อมูลที่มีอยู่เดิม และเขียนในรูปไวยากรณ์ของ XML ทำให้ง่ายต่อการเข้าใจ

2.3.1 ประเภทของอิลิเมนต์ใน XML Schema

XML Schema แบ่งอิลิเมนต์ออกเป็น 2 แบบคือ

1) อิลิเมนต์แบบชนิดข้อมูลแบบซับซ้อน (Complex Type Element) เป็นอิลิเมนต์ที่ประกอบด้วยอิลิเมนต์ย่อยและ/หรือแอตทริบิวต์ (Attribute) อิลิเมนต์ย่อยแต่ละตัว

สามารถกำหนดลำดับ (Order) และ/หรือจำนวนครั้งของการเกิด (Occurrence) อิลิเมนต์ได้ โดยลำดับของอิลิเมนต์แบ่งออกเป็น 3 แบบคือ

(1) sequence หมายถึง ทุกอิลิเมนต์ย่อยจะต้องปรากฏและมีการเรียงลำดับตามที่กำหนด ตัวอย่าง XML Schema และเอกสาร XML ที่สอดคล้องกันแสดงดังตารางที่ 2.1

ตารางที่ 2.1 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย sequence และเอกสาร XML ที่สอดคล้องกับ XML Schema

XML Schema	เอกสาร XML
<xs:element name= “address”> <xs:complexType> <xs:sequence> <xs:element name= “street” type= “xs:string”/> <xs:element name= “city” type= “xs:string” /> <xs:element name= “state” type= “xs:string” /> <xs:element name= “country” type= “xs:string” /> </xs:sequence> </xs:complexType> </xs:element>	<address> <street>Punnakan</street> <city>Hadyai</city> <state>Songkhla</state> <country>Thailand</country> </address>

(2) all หมายถึง ทุกอิลิเมนต์ย่อยจะต้องปรากฏแต่สามารถสลับลำดับของอิลิเมนต์ได้ ตัวอย่าง XML Schema และเอกสาร XML ที่สอดคล้องกันแสดงดังตารางที่ 2.2

(3) choice หมายถึง ต้องมีอิลิเมนต์ย่อยตัวใดตัวหนึ่งปรากฏจาก อิลิเมนต์ย่อยทั้งหมดที่กำหนดขึ้น ตัวอย่าง XML Schema และเอกสาร XML ที่สอดคล้องกันแสดงดังตารางที่ 2.3

สำหรับจำนวนครั้งของการเกิดอิลิเมนต์สามารถระบุด้วยแอตทริบิวต์ดังต่อไปนี้คือ (1) แอตทริบิวต์ minOccurs ใช้ระบุจำนวนครั้งของการเกิดอิลิเมนต์ที่น้อยที่สุด และ (2) แอตทริบิวต์ maxOccurs ใช้ระบุจำนวนครั้งของการเกิดอิลิเมนต์ที่มากที่สุด ในการกำหนดแอตทริบิวต์ดังกล่าวให้กับอิลิเมนต์สามารถใช้ทั้งสองแอตทริบิวต์ร่วมกัน หรือใช้แอตทริบิวต์ตัวใดตัวหนึ่งก็ได้ หากอิลิเมนต์ใดไม่มีการกำหนดแอตทริบิวต์ทั้งสองให้ถือว่าค่าของแอต-

ทริบิวต์ทั้งสองเท่ากับหนึ่ง ตัวอย่าง XML Schema และเอกสาร XML ที่สอดคล้องกันแสดงดังตารางที่ 2.4

ตารางที่ 2.2 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย all และเอกสาร XML ที่สอดคล้องกับ XML Schema

XML Schema	เอกสาร XML
<xs:element name= “Family”> <xs:complexType> <xs:all> <xs:element name= “Father” type= “xs:string” /> <xs:element name= “Mother” type= “xs:string” /> </xs:all> </xs:complexType> </xs:element>	<Family> <Mother>Marry</Mother> <Father>John</Father> </Family> หรือ <Family> <Father>David</Father> <Mother>Ann</Mother> </Family>

ตารางที่ 2.3 XML Schema ที่กำหนดลำดับของอิลิเมนต์ด้วย choice และเอกสาร XML ที่สอดคล้องกับ XML Schema

XML Schema	เอกสาร XML
<xs:element name= “sponsor”> <xs:complexType> <xs:choice> <xs:element name= “parent” type= “xs:string” /> <xs:element name= “guardian” type= “xs:string” /> </xs:choice> </xs:complexType> </element>	<sponsor> <parent>Thomson</parent> </sponsor> หรือ <sponsor> <guardian>Thomson</guardian> </sponsor>

ตารางที่ 2.4 XML Schema ที่กำหนดจำนวนครั้งของการเกิดอิลิเมนต์ด้วยแอตทริบิวต์ minOccurs และแอตทริบิวต์ maxOccurs และเอกสาร XML ที่สอดคล้องกับ XML Schema

XML Schema	เอกสาร XML
<xs:element name= “person”> <xs:complexType> <xs:sequence> <xs:element name= “full_name” type= “xs:string” /> <xs:element name= “child_name” type= “xs:string” maxOccurs= “10” minOccurs= “0” /> </xs:sequence> </xs:complexType> </xs:element>	<person> <full_name>Brook</full_name> <child_name>Robert</child_name> <child_name>Joe</child_name> <child_name>John</child_name> <child_name>Jane</child_name> </person>

2) อิลิเมนต์แบบชนิดข้อมูลแบบง่าย (Simple Type Element) เป็นอิลิเมนต์ที่มีข้อมูลภายในเป็นชนิดข้อมูลพื้นฐานต่างๆ ชนิดข้อมูลพื้นฐานของอิลิเมนต์แบบชนิดข้อมูลแบบง่ายแบ่งออกเป็น 2 ประเภทคือ

2.1) ชนิดข้อมูลแบบกำหนดมาให้แล้ว (Built-in Type) เป็นชนิดข้อมูลพื้นฐานทั่วไป เช่น string

2.2) ชนิดข้อมูลแบบผู้ใช้กำหนดเอง (User-derived Type) เป็นชนิดข้อมูลที่เกิดจากการกำหนดเงื่อนไขจำกัด (Constrain) ให้กับชนิดข้อมูลแบบกำหนดมาให้แล้ว เงื่อนไขจำกัดแสดงดังตารางที่ 2.5 และสามารถแบ่งออกเป็นกลุ่ม (w3school, 2011) ได้ดังนี้

(1) เงื่อนไขจำกัดค่า (Restriction on Values) ได้แก่ minInclusive minExclusive maxInclusive maxExclusive fractionDigits และ totalDigits

(2) เงื่อนไขจำกัดกลุ่มของค่า (Restriction on a Set of Values) ได้แก่ enumeration

(3) เงื่อนไขจำกัดรูปแบบของค่า (Restriction on a Series of Values) ได้แก่ pattern

(4) เงื่อนไขจำกัดของช่องว่าง (Restriction on Whitespace Characters) ได้แก่ whitespace

(5) เงื่อนไขจำกัดความยาว (Restriction on Length) ได้แก่ length minLength และ maxLength

ตารางที่ 2.5 ชื่อและความหมายของเงื่อนไขจำกัด (w3school, 2011)

เงื่อนไข	คำอธิบาย
enumeration	กำหนดรายการข้อมูลที่เป็นไปได้
fractionDigits	กำหนดจำนวนทศนิยม
length	กำหนดความยาวของตัวอักษรซึ่งมีความยาวตั้งแต่ศูนย์ขึ้นไป
maxExclusive	กำหนดค่าของตัวเลขที่มากที่สุด โดยค่าข้อมูลที่ใช้ต้องน้อยกว่าค่าที่กำหนดให้
maxInclusive	กำหนดค่าของตัวเลขที่มากที่สุด โดยค่าข้อมูลที่ใช้ต้องน้อยกว่าหรือเท่ากับค่าที่กำหนดให้
maxLength	กำหนดความยาวที่มากที่สุดของตัวอักษรซึ่งมีความยาวตั้งแต่ศูนย์ขึ้นไป
minExclusive	กำหนดค่าของตัวเลขที่น้อยที่สุด โดยค่าข้อมูลที่ใช้ต้องมากกว่าค่าที่กำหนดให้
minInclusive	กำหนดค่าของตัวเลขที่น้อยที่สุด โดยค่าข้อมูลที่ใช้ต้องมากกว่าหรือเท่ากับค่าที่กำหนดให้
minLength	กำหนดความยาวที่น้อยที่สุดของตัวอักษรซึ่งมีความยาวตั้งแต่ศูนย์ขึ้นไป
pattern	กำหนดรูปแบบของข้อความ
totalDigits	กำหนดจำนวนที่แน่นอนของจุดทศนิยม ซึ่งต้องมีจำนวนตั้งแต่ศูนย์ขึ้นไป
whiteSpaces	กำหนดวิธีการจัดการกับช่องว่าง (whitespace)

2.4 BNF (Backus-Naur Form)

BNF (Sebesta, 2005) เป็นลักษณะของการกำหนดไวยากรณ์ในภาษาการโปรแกรมเพื่อใช้ในการตรวจสอบไวยากรณ์ของภาษาการโปรแกรมนั้นๆ นอกจากนี้ยังสามารถใช้กำหนดไวยากรณ์รูปแบบข้อมูลนำเข้าของโปรแกรมได้อีกด้วย

2.4.1 ไวยากรณ์ของ BNF

ไวยากรณ์ของ BNF ประกอบด้วยชุดของสัญลักษณ์ไม่สิ้นสุด (Non-Terminal Symbol) ซึ่งเป็นสัญลักษณ์ที่อยู่ภายในเครื่องหมาย “<” และ “>” ชุดของสัญลักษณ์สิ้นสุด (Terminal Symbol) และชุดของกฎที่เรียกแต่ละกฎว่าโปรดักชั่น (Production)

โปรดักชั่นประกอบด้วยสัญลักษณ์ที่อยู่ทางด้านซ้าย (Left-Hand Side: LHS) และสัญลักษณ์ที่อยู่ทางด้านขวา (Right-Hand Side: RHS) ของเครื่องหมาย ::= นั่นคือ สัญลักษณ์ทางด้านซ้ายถูกสร้างเป็นสัญลักษณ์ทางด้านขวา โดยที่สัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุด และสัญลักษณ์ทางด้านขวาเป็นลำดับของสัญลักษณ์ที่อาจจะเป็นสัญลักษณ์สิ้นสุดหรือสัญลักษณ์ไม่สิ้นสุดก็ได้ สัญลักษณ์ทางด้านขวาที่เป็นสัญลักษณ์ไม่สิ้นสุดจะต้องนำไปสร้างเป็นกฎใหม่จนกระทั่งสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์สิ้นสุดทั้งหมด และหากสัญลักษณ์ไม่สิ้นสุดนั้นสามารถสร้างได้มากกว่า 1 รูปแบบจะใช้เครื่องหมาย “|” คั่นระหว่างรูปแบบ ตัวอย่างของ BNF แสดงดังภาพประกอบที่ 2.5

```
<number>   ::=   <digit> | <number><digit>
<digit>    ::=   0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

ภาพประกอบที่ 2.5 ตัวอย่าง BNF

จากภาพประกอบที่ 2.6 BNF ประกอบด้วยโปรดักชั่นแรกคือสัญลักษณ์ไม่สิ้นสุดชื่อ number นิยามเป็นสัญลักษณ์ไม่สิ้นสุดชื่อ digit หรือสัญลักษณ์ไม่สิ้นสุดชื่อ number ตามด้วยสัญลักษณ์ไม่สิ้นสุดชื่อ digit และโปรดักชั่นที่ 2 คือสัญลักษณ์ไม่สิ้นสุดชื่อ digit นิยามเป็น 0 หรือ 1 หรือ 2 หรือ 3 หรือ 4 หรือ 5 หรือ 6 หรือ 7 หรือ 8 หรือ 9 ตัวอย่างการนิยามสัญลักษณ์ไม่สิ้นสุดชื่อ number แสดงดังตารางที่ 2.6

ตารางที่ 2.6 ตัวอย่างการนิยามสัญลักษณ์ไม่สิ้นสุดชื่อ number

ตัวอย่างที่ 1	ตัวอย่างที่ 2
<number> ::= <digit> ::= 6	<number> ::= <number><digit> ::= <digit><digit> ::= 18

2.5 การทดสอบซอฟต์แวร์ (Software Testing)

การทดสอบซอฟต์แวร์ (Ammann and Offutt, 2008) เป็นกิจกรรมหนึ่งที่ทำควบคู่ไปกับการพัฒนาซอฟต์แวร์ซึ่งมีจุดมุ่งหมายเพื่อประเมินคุณภาพและแก้ไขข้อบกพร่องของซอฟต์แวร์ด้วยการค้นหาข้อผิดพลาด (Error) ทั้งที่ปรากฏให้เห็นและไม่ปรากฏให้เห็น

2.5.1 เทคนิคการทดสอบซอฟต์แวร์

เทคนิคในการทดสอบซอฟต์แวร์โดยทั่วไปจะแบ่งออกเป็น 2 แบบคือการทดสอบแบบกล่องดำ (Black-box Testing) และการทดสอบแบบกล่องขาว (White-box Testing) กล่าวคือ วิธีการทดสอบแบบกล่องดำใช้ลักษณะของ input และ output จากข้อกำหนดของความต้องการ (Requirements) มาใช้ในการสร้างกรณีทดสอบ และมักจะทำการทดสอบโดยผู้ที่ไม่ได้เป็นผู้พัฒนาโปรแกรม เช่น การวิเคราะห์ค่าขอบเขต (Boundary Value Analysis) การทดสอบแบบสุ่ม (Random Testing) เป็นต้น ส่วนวิธีการทดสอบแบบกล่องขาวมักใช้โครงสร้างของโปรแกรมมาใช้ในการสร้างกรณีทดสอบ และดำเนินการทดสอบโดยผู้พัฒนาโปรแกรม เช่น การทดสอบโดยใช้เงื่อนไข (Predicate Testing) การทดสอบการไหลของข้อมูล (Data Flow Testing) เป็นต้น

2.5.2 การครอบคลุมการทดสอบ (Coverage Criteria)

การครอบคลุมการทดสอบ (Ammann and Offutt, 2008) เป็นกฎหรือกลุ่มของกฎที่ใช้กำหนดความต้องการในการทดสอบ (Test Requirement) บนชุดทดสอบ (Test Set) ในการทดสอบซอฟต์แวร์ใดๆ นั้นกรณีทดสอบที่จะนำมาใช้ทดสอบจะต้องครอบคลุมทุกๆ กรณี

การครอบคลุม BNF (BNF Coverage Criteria) เป็นหลักการครอบคลุมเชิงไวยากรณ์ (Syntax-based Coverage Criteria) โดยใช้ไวยากรณ์ที่อยู่ในรูปของ BNF มาใช้ในการสร้างกรณีทดสอบ โดยกรณีทดสอบจะได้จากการนิยามรูปแบบของสัญลักษณ์สิ้นสุดด้วยการแทนที่สัญลักษณ์ไม่สิ้นสุดด้วยโปรดักชั่น การแทนที่จะถูกกระทำจนกระทั่งสัญลักษณ์ไม่สิ้นสุดทุกตัวถูกแทนที่ด้วยสัญลักษณ์สิ้นสุดทั้งหมด การสร้างกรณีทดสอบโดยหลักการครอบคลุม BNF มี 2 ลักษณะต่อไปนี้คือ

1) การครอบคลุมสัญลักษณ์สิ้นสุด (Terminal Symbol Coverage: TSC) กล่าวคือ กรณีทดสอบที่ได้ต้องครอบคลุมทุกสัญลักษณ์สิ้นสุด โดยแต่ละสัญลักษณ์สิ้นสุดจะต้อง

ถูกใช้อย่างน้อยหนึ่งครั้ง เช่น จากภาพประกอบที่ 2.5 มีสัญลักษณ์สิ้นสุด 10 ตัว คือ 0 1 2 3 4 5 6 7 8 และ 9 ดังนั้นกรณีทดสอบที่ได้จะต้องครอบคลุมสัญลักษณ์สิ้นสุดทั้ง 11 ตัว

2) การครอบคลุมโปรดักชั่น (Production Coverage: PDC) กล่าวคือ กรณีทดสอบที่ได้ครอบคลุมทุกโปรดักชั่น โดยแต่ละโปรดักชั่นจะต้องถูกใช้อย่างน้อยหนึ่งครั้ง เช่น จากภาพประกอบที่ 2.5 ที่โปรดักชั่นแรกสามารถแบ่งเป็นโปรดักชั่นย่อยได้ 2 โปรดักชั่น และโปรดักชั่นที่ 2 สามารถแบ่งเป็นโปรดักชั่นย่อยได้ 10 โปรดักชั่น รวมโปรดักชั่นย่อยทั้งหมด 12 โปรดักชั่น ดังนั้นกรณีทดสอบที่ได้จะต้องครอบคลุมทั้ง 12 โปรดักชั่น

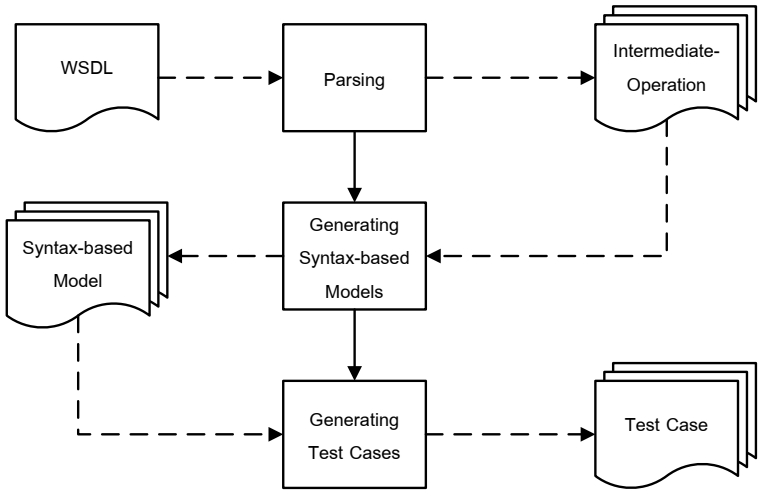
บทที่ 3

ขั้นตอนการดำเนินการวิจัย

เนื้อหาในบทนี้กล่าวถึงวิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสจากเอกสาร WSDL ประกอบด้วยกรอบแนวคิดที่นำเสนอและขั้นตอนต่างๆ ในการสร้างกรณีทดสอบ

3.1 กรอบแนวคิดการสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสจากเอกสาร WSDL

งานวิจัยนี้นำเสนอวิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส แบบจำลองเชิงไวยากรณ์สร้างจากเอกสาร WSDL โดยพิจารณาในส่วนที่อธิบายรูปแบบข้อมูลนำเข้าของการดำเนินการ กรอบแนวคิดของวิธีการที่นำเสนอนี้เริ่มจากขั้นตอนการสกัดเอกสาร WSDL ให้เหลือเฉพาะส่วนที่เป็นรูปแบบข้อมูลนำเข้าของแต่ละการดำเนินการ โดยในงานวิจัยนี้จะเรียกส่วนที่สกัดได้ว่า “อินเตอร์มีเดียทโอเปอร์เรชั่น (Intermediate-Operation)” ขั้นตอนถัดไปเป็นขั้นตอนที่นำแต่ละอินเตอร์มีเดียทโอเปอร์เรชั่นมาสร้างเป็นแบบจำลองเชิงไวยากรณ์ สุดท้ายทำการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ดังภาพประกอบที่ 3.1



ภาพประกอบที่ 3.1 ขั้นตอนการสร้างกรณีทดสอบสำหรับเว็บเซอร์วิสจากเอกสาร WSDL

3.2 ขั้นตอนการสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสจากเอกสาร WSDL

การสร้างกรณีทดสอบเชิงไวยากรณ์ประกอบด้วย 3 ขั้นตอนหลักคือ 1) ขั้นตอนการสกัดข้อมูล 2) ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์ และ 3) ขั้นตอนการสร้างกรณีทดสอบ โดยมีรายละเอียดของแต่ละขั้นตอนดังต่อไปนี้

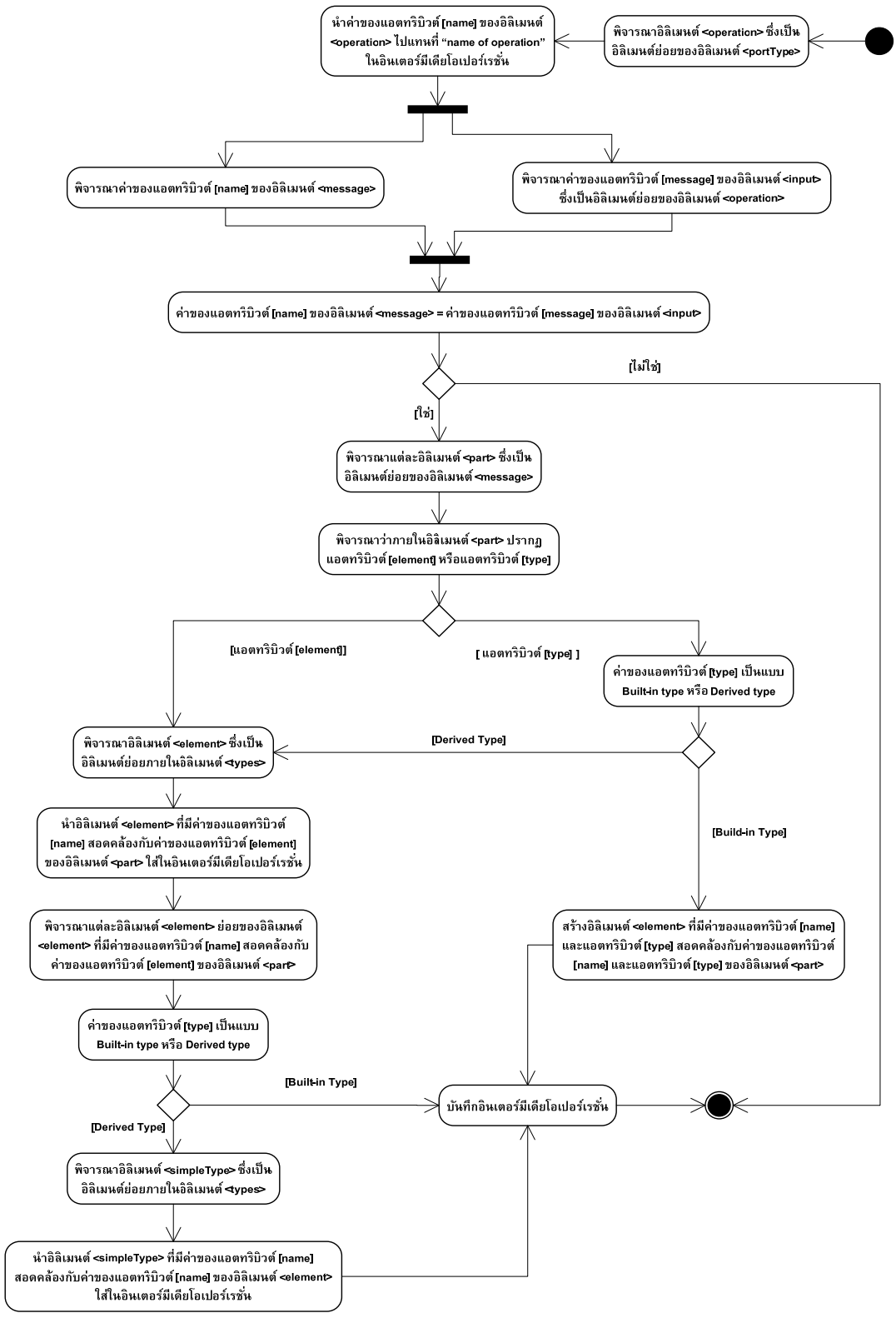
ขั้นตอนที่ 1 การสกัดข้อมูล (Parsing)

ขั้นตอนการสกัดข้อมูลเป็นขั้นตอนการสกัดการดำเนินการต่างๆ ของเว็บเซอร์วิสซึ่งพิจารณาจากความสัมพันธ์ของอิลิเมนต์ 3 อิลิเมนต์ในเอกสาร WSDL ได้แก่ อิลิเมนต์ <types> อิลิเมนต์ <message> และอิลิเมนต์ <portType> และเรียกสิ่งที่สกัดได้ว่า อินเตอร์มีเดียทโอเปอร์เรชั่น อินเตอร์มีเดียทโอเปอร์เรชั่นที่สกัดได้จะบอกให้ทราบถึงชื่อการดำเนินการ จำนวนพารามิเตอร์ และชนิดข้อมูลของแต่ละพารามิเตอร์ของการดำเนินการ โดยในหนึ่งอินเตอร์มีเดียทโอเปอร์เรชั่นจะเก็บการดำเนินการเพียง 1 การดำเนินการเท่านั้น ดังนั้นจำนวนอินเตอร์มีเดียทโอเปอร์เรชั่นที่สกัดได้จะเท่ากับจำนวนของการดำเนินการที่มีอยู่ในเอกสาร WSDL อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการมีโครงสร้างแสดงดังภาพประกอบที่ 3.2

```
<“name of operation”>
    <element />
    …
    <element />
</“name of operation”>
```

ภาพประกอบที่ 3.2 โครงสร้างของอินเตอร์มีเดียทโอเปอร์เรชั่น

ขั้นตอนการสกัดอินเตอร์มีเดียทโอเปอร์เรชั่นของแต่ละการดำเนินการที่มีอยู่ในเอกสาร WSDL แสดงดังภาพประกอบที่ 3.3 และมีรายละเอียดดังต่อไปนี้



ภาพประกอบที่ 3.3 แผนภาพกิจกรรมการสร้างอินเตอร์มีเดียทโอเปอร์เรชั่น

1) พิจารณาอิลิเมนต์ <operation> ภายในอิลิเมนต์ <portType> โดยนำค่าของแอตทริบิวต์ [name] ของอิลิเมนต์ <operation> ไปแทนที่ “name of operation” ในอินเตอร์มีเดียทโอเปอร์เรชั่น

2) พิจารณาอิลิเมนต์ <message> ที่มีค่าของแอตทริบิวต์ [name] สอดคล้องกับค่าของแอตทริบิวต์ [message] ของอิลิเมนต์ <input> ภายในอิลิเมนต์ <operation> จากนั้นพิจารณาแต่ละอิลิเมนต์ <part> ภายในอิลิเมนต์ <message> ดังนี้

2.1) ถ้าอิลิเมนต์ <part> ปรากฏแอตทริบิวต์ [element] ให้นำอิลิเมนต์ <element> ภายในอิลิเมนต์ <types> ที่มีค่าของแอตทริบิวต์ [name] สอดคล้องกับค่าของแอตทริบิวต์ [element] ของอิลิเมนต์ <part> ใส่ในอินเตอร์มีเดียทโอเปอร์เรชั่น

2.2) ถ้าอิลิเมนต์ <part> ปรากฏแอตทริบิวต์ [type] ให้พิจารณาค่าของแอตทริบิวต์ [type] ตามกรณีต่อไปนี้

2.2.1) กรณีค่าของแอตทริบิวต์ [type] เป็นชนิดข้อมูลแบบกำหนดมาให้แล้ว ให้สร้างอิลิเมนต์ <element> ที่มีค่าของแอตทริบิวต์ [name] และแอตทริบิวต์ [type] สอดคล้องกับค่าของแอตทริบิวต์ [name] และแอตทริบิวต์ [type] ของ อิลิเมนต์ <part> ใส่ในอินเตอร์มีเดียทโอเปอร์เรชั่น

2.2.2) กรณีค่าของแอตทริบิวต์ [type] เป็นชนิดข้อมูลแบบผู้ใช้กำหนดเอง ให้นำอิลิเมนต์ <element> ภายในอิลิเมนต์ <types> ที่มีค่าของแอตทริบิวต์ [name] สอดคล้องกับค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <part> ใส่ในอินเตอร์มีเดียทโอเปอร์เรชั่น

3) พิจารณาค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element> ของอินเตอร์-มีเดียทโอเปอร์เรชั่น ถ้าค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element> เป็นชนิดข้อมูลแบบผู้ใช้กำหนดเอง ให้นำอิลิเมนต์ <simpleType> ภายในอิลิเมนต์ <types> ที่มีค่าของแอตทริบิวต์ [name] สอดคล้องกับค่าของแอตทริบิวต์ [type] ใส่ในอินเตอร์มีเดียทโอเปอร์เรชั่น

ตัวอย่างเอกสาร WSDL ดังแสดงในภาพประกอบที่ 3.4 เป็นเอกสาร WSDL ของเว็บเซอร์วิส MathService ที่ให้บริการเกี่ยวกับการหาผลบวกซึ่งมีการดำเนินการ 1 การดำเนินการคือการดำเนินการ Add และอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Add แสดงดังภาพประกอบที่ 3.5

จากภาพประกอบที่ 3.4 การดำเนินการ Add คืออิลิเมนต์ <operation> ในบรรทัดที่ 17 ซึ่งมีค่าของแอตทริบิวต์ [message] ของอิลิเมนต์ <input> เป็น AddSoapIn เริ่มสร้างอินเตอร์มีเดียทโอเปอร์เรชั่นโดยนำค่าของแอตทริบิวต์ [name] ของอิลิเมนต์ <operation> ไปแทนที่ “name of operation” ในอินเตอร์มีเดียทโอเปอร์เรชั่นดังแสดงในบรรทัดที่ 1 และ 10 ของภาพประกอบที่ 3.5 จากนั้นพิจารณาอิลิเมนต์ <message> ของภาพประกอบที่ 3.4 ที่มีค่าของแอตทริบิวต์ [name] เป็น AddSoapIn ซึ่งมีอิลิเมนต์ <part> ในบรรทัดที่ 14 ปรากฏแอต-ทริบิวต์ [element] ชื่อ Add ดังนั้นภายในอิลิเมนต์ <types> นำอิลิเมนต์ <element> ตั้งแต่

บรรทัดที่ 4-11 ของภาพประกอบที่ 3.4 ไปใส่อินเตอร์มีเดียทโอเปอร์เรชั่นดังแสดงในบรรทัดที่ 2-9 ของภาพประกอบที่ 3.5 จากนั้นพิจารณาอิลิเมนต์ <element> ย่อยของอิลิเมนต์ <element> ชื่อ Add ในบรรทัดที่ 5-6 ของภาพประกอบที่ 3.5 พบว่าอิลิเมนต์ย่อยทั้งหมดมีชนิดข้อมูลแบบกำหนดมาให้แล้ว

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <wsdl:definitions>
3  <wsdl:types>
4      <s:element name="Add">
5        <s:complexType>
6          <s:sequence>
7            <s:element minOccurs="1" maxOccurs="1" name="A" type="s:float" />
8            <s:element minOccurs="1" maxOccurs="1" name="B" type="s:float" />
9          </s:sequence>
10       </s:complexType>
11     </s:element>
12   </wsdl:types>
13   <wsdl:message name="AddSoapIn">
14     <wsdl:part name="parameters" element="tns:Add" />
15   </wsdl:message>
16   <wsdl:portType name="MathServiceSoap">
17     <wsdl:operation name="Add">
18       <wsdl:input message="tns:AddSoapIn" />
19       <wsdl:output message="tns:AddSoapOut" />
20     </wsdl:operation>
21   </wsdl:portType>
22   <wsdl:binding>…</wsdl:binding>
23   <wsdl:service>…</wsdl:service>
24 </wsdl:definitions>
```

ภาพประกอบที่ 3.4 เอกสาร WSDL ของเว็บเซอร์วิส MathService

```
1  <Add>
2      <element name="Add">
3        <complexType>
4          <sequence>
5            <element minOccurs="1" maxOccurs="1" name="A" type="float" />
6            <element minOccurs="1" maxOccurs="1" name="B" type="float" />
7          </sequence>
8        </complexType>
9      </element>
10 </Add>
```

ภาพประกอบที่ 3.5 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Add

ขั้นตอนที่ 2 การสร้างแบบจำลองเชิงไวยากรณ์ (Generating Syntax-based Models)

เมื่อได้อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการจากขั้นตอนที่ 1 แล้ว แต่ละอินเตอร์มีเดียทโอเปอร์เรชั่นจะถูกนำมาสร้างเป็นแบบจำลองเชิงไวยากรณ์ และแบบจำลองเชิงไวยากรณ์ที่สร้างได้จะต้องทำการปรับปรุงก่อนนำไปใช้ในการสร้างกรณีทดสอบ

แบบจำลองเชิงไวยากรณ์ประกอบด้วยหลายๆ โปรดักชั่นซึ่งสร้างจากอิลิเมนต์ต่างๆ ในอินเตอร์มีเดียทโอเปอร์เรชั่น แต่ละโปรดักชั่นประกอบด้วยสัญลักษณ์ทางด้านซ้ายและสัญลักษณ์ทางด้านขวาของเครื่องหมายเท่ากับ (=) นั่นคือสัญลักษณ์ทางด้านซ้ายจะถูกสร้างเป็นสัญลักษณ์ทางด้านขวา โดยที่สัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุด และสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์หรือกลุ่มของสัญลักษณ์ที่อาจเป็นสัญลักษณ์สิ้นสุด หรือสัญลักษณ์ไม่สิ้นสุด หรือทั้งสองสัญลักษณ์ประกอบกัน โปรดักชั่นจะถูกสร้างต่อไปจนกระทั่งสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์ไม่สิ้นสุดทั้งหมด ในการสร้างแบบจำลองเชิงไวยากรณ์ได้กำหนดสัญลักษณ์และความหมายของสัญลักษณ์แสดงดังตารางที่ 3.1

ตารางที่ 3.1 สัญลักษณ์และความหมายของสัญลักษณ์สำหรับแบบจำลองเชิงไวยากรณ์

สัญลักษณ์	ความหมาย
< >	สัญลักษณ์ไม่สิ้นสุด
“ ”	สัญลักษณ์สิ้นสุดที่เป็นค่าคงที่
datatype	สัญลักษณ์สิ้นสุดที่เป็นชนิดข้อมูล
[ ]	สัญลักษณ์แสดงการมีเงื่อนไขจำกัดในตาราง
( )	สัญลักษณ์แสดงการสลับลำดับ
|	สัญลักษณ์แสดงการเลือก
?	สัญลักษณ์แสดงการแทนหรือไม่แทน
+	สัญลักษณ์แสดงการแทนอย่างน้อย 1 ครั้ง
*	สัญลักษณ์แสดงการแทนหรือไม่แทน การแทนสามารถแทนจำนวนเท่าไรก็ได้

พารามิเตอร์บางตัวในอินเตอร์มีเดียทโอเปอร์เรชั่นอาจมีเงื่อนไขจำกัดของชนิดข้อมูลแบบกำหนดเอง ซึ่งเงื่อนไขจำกัดดังกล่าวอธิบายด้วยอิลิเมนต์ <simpleType> ในอินเตอร์มีเดียทโอเปอร์เรชั่น แบบจำลองเชิงไวยากรณ์ที่สร้างจากอินเตอร์มีเดียทโอเปอร์เรชั่นลักษณะนี้จะต้องเก็บข้อมูลเงื่อนไขจำกัดลงในตารางเงื่อนไขจำกัด ซึ่งในตารางประกอบด้วย 2 ส่วนดังต่อไปนี้

1) ชนิดข้อมูล เป็นชนิดข้อมูลแบบกำหนดมาให้แล้วซึ่งได้จากค่าของแอตทริบิวต์ [base] ของอิลิเมนต์ <restriction> ซึ่งเป็นอิลิเมนต์ภายในของอิลิเมนต์ <simpleType> ในอินเตอร์มีเดียทโอเปอร์เรชั่น

2) ชื่อและค่าของเงื่อนไขจำกัด โดยชื่อเงื่อนไขจำกัดได้จากอิลิเมนต์เงื่อนไขจำกัดภายในอิลิเมนต์ <restriction> และค่าของเงื่อนไขจำกัดได้จากค่าของแอตทริบิวต์ [value] ของอิลิเมนต์เงื่อนไขจำกัด

ตัวอย่างอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ booksRequest ที่มีเงื่อนไขจำกัดของพารามิเตอร์แสดงดังภาพประกอบที่ 3.6

```
1  <booksRequest>
2    <xs:element name="book">
3      <xs:complexType>
4        <xs:sequence>
5          <xs:element name="ISBN" type="xs:string" />
6          <xs:element name="price" type="xs:double"/>
7          <xs:element name="year" type="xs:yearType"/>
8        </xs:sequence>
9      </xs:complexType>
10   </xs:element>
11   <xs:simpleType name= "yearType">
12     <xs:restriction base = "xs:integer">
13     <xs:totalDigits value = "4" />
14    </xs:restriction>
15   </xs:simpleType>
16 </booksRequest>
```

ภาพประกอบที่ 3.6 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ booksRequest

จากภาพประกอบที่ 3.6 อิลิเมนต์ชื่อ year ในบรรทัดที่ 7 มีชนิดข้อมูลเป็น yearType ซึ่งเป็นชนิดข้อมูลแบบผู้ใช้กำหนดเองซึ่งอธิบายเงื่อนไขจำกัดด้วยอิลิเมนต์ <simpleType> ในบรรทัดที่ 11 โดยชนิดข้อมูลถูกกำหนดด้วยอิลิเมนต์ <restriction> ในบรรทัดที่ 12 ซึ่งเป็นชนิดข้อมูลแบบ integer และกำหนดเงื่อนไขจำกัดด้วยอิลิเมนต์เงื่อนไขจำกัดในบรรทัดที่ 13 ซึ่งมีชื่อเงื่อนไขเป็น totalDigits และค่าของเงื่อนไขจำกัดมีค่าเป็น 4 ตามค่าของแอตทริบิวต์ [value] ตารางเงื่อนไขจำกัดของการดำเนินการ booksRequest แสดงดังตารางที่ 3.2

ตารางที่ 3.2 ตารางเงื่อนไขจำกัดของอิลิเมนต์ชื่อ year ของการดำเนินการ booksRequest

ชนิดข้อมูล	เงื่อนไขจำกัด
integer	totalDigits = 4

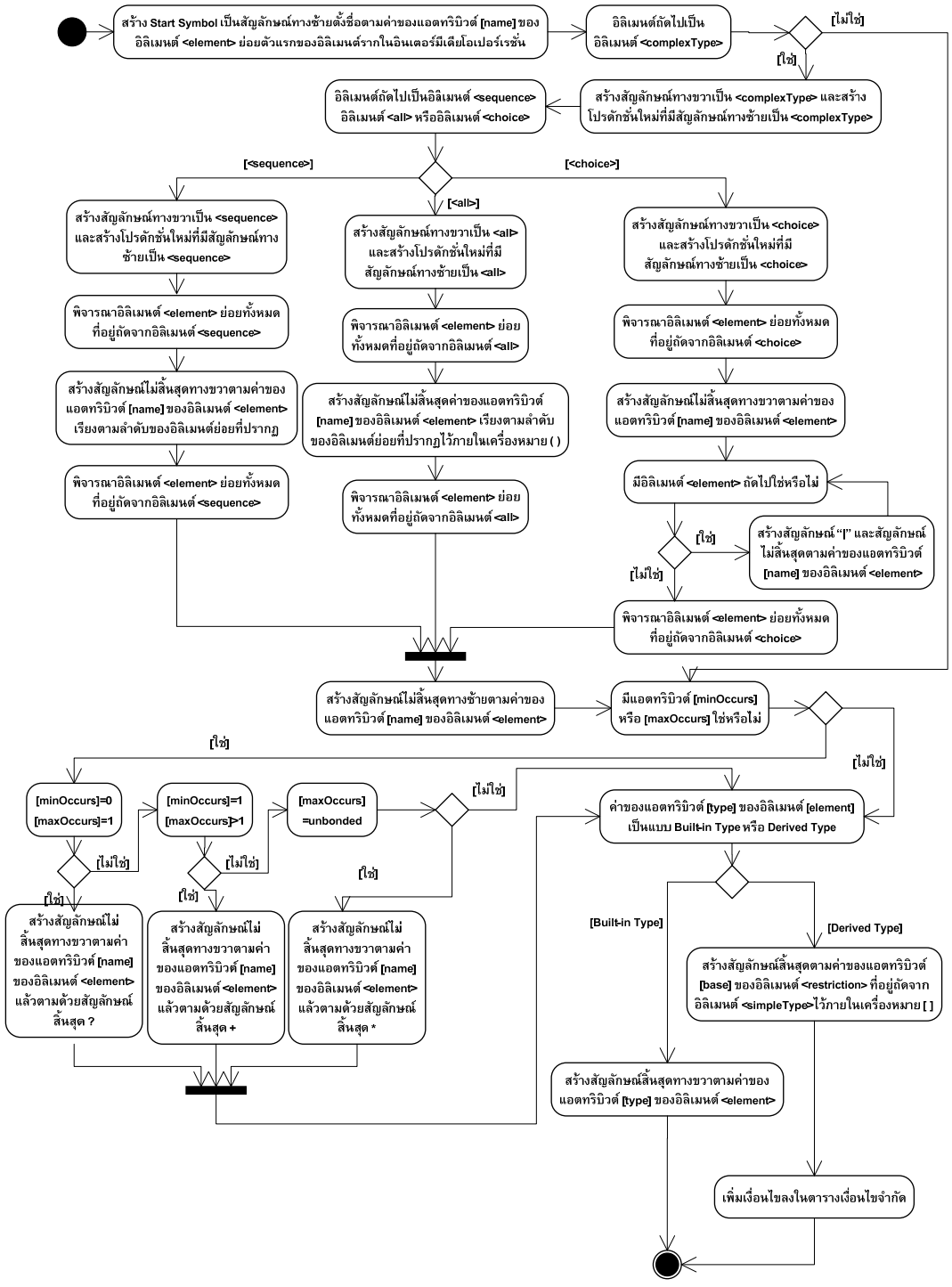
ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์จากอินเตอร์มีเดียทโอเปอร์เรชั่น แสดงดังภาพประกอบที่ 3.7 และมีรายละเอียดดังต่อไปนี้

1) สร้างสัญลักษณ์เริ่มต้น (Start Symbol) ของแบบจำลองเชิงไวยากรณ์เป็นสัญลักษณ์ไม่สิ้นสุดทางด้านซ้ายซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] ของอิลิเมนต์ <element> ย่อยตัวแรกของอิลิเมนต์รากในอินเตอร์มีเดียทโอเปอร์เรชั่น จากนั้นสร้างสัญลักษณ์ทางด้านขวาโดยพิจารณาที่อิลิเมนต์ถัดไป กรณีที่ตามด้วยอิลิเมนต์ <complexType> ให้ดำเนินการในขั้นตอน 1.1) มิเช่นนั้นให้ดำเนินการในขั้นตอน 2)

1.1) กรณีที่ตามด้วยอิลิเมนต์ <complexType> ให้สร้างสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเป็น <complexType> และสร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็น <complexType> และพิจารณาต่อว่าอิลิเมนต์ถัดไปมีลักษณะใดต่อไปนี้

1.1.1) ถ้าอิลิเมนต์ถัดไปเป็นอิลิเมนต์ <sequence> ให้สร้างสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเป็น <sequence> และสร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุด <sequence> จากนั้นพิจารณาอิลิเมนต์ <element> ย่อยทั้งหมดที่อยู่ถัดจากอิลิเมนต์ <sequence> เพื่อสร้างเป็นลำดับของสัญลักษณ์ไม่สิ้นสุดทางด้านขวาซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] ของแต่ละอิลิเมนต์ <element> โดยลำดับของสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเรียงตามลำดับของอิลิเมนต์ <element> ย่อยที่อยู่ถัดจากอิลิเมนต์ <sequence>

1.1.2) ถ้าอิลิเมนต์ถัดไปเป็นอิลิเมนต์ <all> ให้สร้างสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเป็น <all> และสร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุด <all> จากนั้นพิจารณาอิลิเมนต์ <element> ย่อยทั้งหมดที่อยู่ถัดจากอิลิเมนต์ <all> เพื่อสร้างเป็นลำดับของสัญลักษณ์ไม่สิ้นสุดทางด้านขวาซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] ของแต่ละอิลิเมนต์ <element> เรียงตามลำดับของอิลิเมนต์ <element> ย่อยที่อยู่ถัดจากอิลิเมนต์ <all> โดยสัญลักษณ์ไม่สิ้นสุดทางด้านขวาทั้งหมดอยู่ภายในเครื่องหมาย “(” และ “)”



ภาพประกอบที่ 3.7 แผนภาพกิจกรรมของการสร้างแบบจำลองเชิงไวยากรณ์

1.1.3) ถ้าอิลิเมนต์ถัดไปเป็นอิลิเมนต์ <choice> ให้สร้างสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเป็น <choice> และสร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุด <choice> จากนั้นพิจารณาอิลิเมนต์ <element> ย่อยทั้งหมดที่อยู่ถัดจากอิลิเมนต์ <choice> เพื่อสร้างเป็นลำดับของสัญลักษณ์ไม่สิ้นสุดทางด้านขวาซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] ของแต่ละอิลิเมนต์ <element> ซึ่งลำดับของสัญลักษณ์ไม่สิ้นสุดทางด้านขวาเรียงตามลำดับของอิลิเมนต์ <element> ย่อยที่อยู่ถัดจากอิลิเมนต์ <choice> และแยกสัญลักษณ์ไม่สิ้นสุดทางด้านขวาแต่ละตัวด้วยสัญลักษณ์แสดงการเลือก (|)

1.2) พิจารณาแต่ละอิลิเมนต์ <element> ย่อยของอิลิเมนต์ <sequence> อิลิเมนต์ <all> และอิลิเมนต์ <choice> ตามขั้นตอนในข้อ 2)

2) กรณีที่มีแอตทริบิวต์ [minOccurs] และแอตทริบิวต์ [maxOccurs] ให้พิจารณาค่าของแอตทริบิวต์ [minOccurs] และแอตทริบิวต์ [maxOccurs] ตามขั้นตอนในข้อ 2.1) ดังนี้

2.1) ถ้าค่าของแอตทริบิวต์ [minOccurs] = 0 และค่าของแอตทริบิวต์ [maxOccurs] = 1 ให้สร้างสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] และสร้างสัญลักษณ์แสดงการแทนหรือไม่แทน (?) ตามลำดับ จากนั้นดำเนินการต่อในข้อ 2.2)

2.2) ถ้าค่าของแอตทริบิวต์ [minOccurs] = 1 และค่าของแอตทริบิวต์ [maxOccurs] มีค่ามากกว่า 1 ให้สร้างสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] และสร้างสัญลักษณ์แสดงการแทนอย่างน้อยหนึ่งครั้ง (+) ตามลำดับ จากนั้นดำเนินการต่อในข้อ 2.3)

2.3) ถ้าค่าของแอตทริบิวต์ [maxOccurs] = unbounded ให้สร้างสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] และสร้างสัญลักษณ์แสดงการแทนหรือไม่แทน (*) ตามลำดับ จากนั้นดำเนินการต่อในข้อ 3)

3) พิจารณาค่าของแอตทริบิวต์ [type] ตามกรณีต่อไปนี้

3.1) ถ้าค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element> เป็นชนิดข้อมูลแบบกำหนดมาให้แล้ว ให้สร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] และสร้างสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์สิ้นสุดตามค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element>

3.2) ถ้าค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element> เป็นชนิดข้อมูลแบบผู้ใช้กำหนดเอง ให้พิจารณาอิลิเมนต์ <simpleType> ที่มีค่าของแอตทริบิวต์ [name] สอดคล้องกับค่าของแอตทริบิวต์ [type] ของอิลิเมนต์ <element> จากนั้นสร้างโปรดักชั่นใหม่ที่มีสัญลักษณ์ทางด้านซ้ายเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตามค่าของแอตทริบิวต์ [name] ของอิลิเมนต์ <simpleType> และสร้างสัญลักษณ์ทางด้านขวาเป็นสัญลักษณ์ไม่สิ้นสุดซึ่งตั้งชื่อตาม

ค่าของแอตทริบิวต์ [base] ของอิลิเมนต์ <restriction> ซึ่งอยู่ภายในเครื่องหมาย “[” และ “]” จากนั้นเพิ่มเงื่อนไขจำกัดลงในตารางเงื่อนไขจำกัดดังตัวอย่างในตารางที่ 3.2 หากมีเงื่อนไขจำกัดมากกว่า 1 เงื่อนไขให้แยกแต่ละเงื่อนไขด้วยเครื่องหมาย “,”

จากอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Add ในภาพประกอบที่ 3.5 สามารถสร้างแบบจำลองเชิงไวยากรณ์ดังภาพประกอบที่ 3.8

```
<Add> = <complexType>
<complexType> = <sequence>
<sequence> = <A><B>
<A> = float
<B> = float
```

ภาพประกอบที่ 3.8 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Add

แบบจำลองเชิงไวยากรณ์ที่สร้างจากขั้นตอนข้างต้นจะถูกทำการปรับปรุง (optimize) ก่อนนำไปใช้ในการสร้างกรณีทดสอบในขั้นตอนที่ 3 การปรับปรุงแบบจำลองเชิงไวยากรณ์จะเริ่มตั้งแต่โปรดักชั่นแรก โดยพิจารณาจากสัญลักษณ์ไม่สิ้นสุดทางด้านขวาและสัญลักษณ์ไม่สิ้นสุดทางด้านซ้ายของโปรดักชั่นถัดไป ถ้าเป็นสัญลักษณ์เดียวกันให้ตัดสัญลักษณ์ดังกล่าวออกเพื่อทำการรวมโปรดักชั่น

แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Add โปรดักชั่นที่ 1 มีสัญลักษณ์ไม่สิ้นสุดทางขวา <complexType> ซึ่งเหมือนกับสัญลักษณ์ไม่สิ้นสุดทางซ้าย <complexType> ของโปรดักชั่นที่ 2 และในโปรดักชั่นที่ 2 มีสัญลักษณ์ไม่สิ้นสุดทางขวา <sequence> ซึ่งเหมือนกับสัญลักษณ์ไม่สิ้นสุดทางซ้าย <sequence> ของโปรดักชั่นที่ 3 ดังนั้นจึงสามารถรวมโปรดักชั่นที่ 1, 2 และ 3 เข้าด้วยกันได้ทำให้แบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วแสดงดังภาพประกอบที่ 3.9

```
1 <Add> = <A><B>
2 <A> = float
3 <B> = float
```

ภาพประกอบที่ 3.9 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Add ที่ปรับปรุงแล้ว

ขั้นตอนที่ 3 การสร้างกรณีทดสอบ (Generating Test Case)

การสร้างกรณีทดสอบใช้หลักการครอบคลุมโปรดักชั่น (Ammann and Offutt, 2008) ซึ่งแต่ละโปรดักชั่นจะต้องถูกใช้อย่างน้อย 1 ครั้ง โดยขั้นตอนการสร้างกรณีทดสอบจะใช้แบบจำลองเชิงไวยากรณ์ที่ผ่านการปรับปรุงจากขั้นตอนที่ 2 มาใช้ โดยจะทำการแทนที่สัญลักษณ์ไม่สิ้นสุดด้วยโปรดักชั่นจากแบบจำลองเชิงไวยากรณ์ โดยใช้เครื่องหมายลูกศร (-->) แทนการแทนที่ในแต่ละครั้งจนได้สัญลักษณ์สิ้นสุดทั้งหมด ซึ่งกรณีทดสอบจะได้จากการแทนที่จนกระทั่งได้สัญลักษณ์ทุกตัวเป็นสัญลักษณ์สิ้นสุด

การแทนที่เริ่มต้นจากสัญลักษณ์เริ่มต้นของแบบจำลองเชิงไวยากรณ์ซึ่งเป็นสัญลักษณ์ไม่สิ้นสุด และทำการแทนที่สัญลักษณ์เริ่มต้นด้วยโปรดักชั่นแรกในแบบจำลองเชิงไวยากรณ์ จากนั้นทำการแทนที่สัญลักษณ์ไม่สิ้นสุดแต่ละตัวที่อยู่ทางขวาของเครื่องหมายลูกศรด้วยโปรดักชั่นครั้งละ 1 โปรดักชั่น โดยเลือกโปรดักชั่นที่มีสัญลักษณ์ไม่สิ้นสุดทางซ้ายตรงกับสัญลักษณ์ไม่สิ้นสุดที่อยู่ทางขวาของเครื่องหมายลูกศรซึ่งกำลังถูกพิจารณา ในการแทนที่สัญลักษณ์ไม่สิ้นสุดนั้น แต่ละโปรดักชั่นในแบบจำลองเชิงไวยากรณ์จะต้องถูกใช้อย่างน้อย 1 ครั้งและต้องทำการแทนที่สัญลักษณ์ไม่สิ้นสุดจนกระทั่งสัญลักษณ์ทางขวาของลูกศรเป็นสัญลักษณ์สิ้นสุดทั้งหมด

จากแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วในภาพประกอบที่ 3.9 สัญลักษณ์เริ่มต้นของแบบจำลองเชิงไวยากรณ์คือ <Add> ทำการแทนที่สัญลักษณ์ไม่สิ้นสุด <Add> ด้วยโปรดักชั่นของภาพประกอบที่ 3.9 บรรทัดที่ 1 ได้ผลลัพธ์เป็นสัญลักษณ์ไม่สิ้นสุด <A> ตามด้วยสัญลักษณ์ไม่สิ้นสุด <B> ดังภาพประกอบที่ 3.10 บรรทัดที่ 1 จากนั้นทำการแทนที่สัญลักษณ์ไม่สิ้นสุดซึ่งอยู่ทางขวาของเครื่องหมายลูกศร นั่นคือ แทนที่สัญลักษณ์ไม่สิ้นสุด <A> ของภาพประกอบที่ 3.10 ด้วยโปรดักชั่นที่ 2 ของภาพประกอบที่ 3.9 ได้ผลลัพธ์เป็นสัญลักษณ์สิ้นสุด float ตามด้วยสัญลักษณ์ไม่สิ้นสุด <B> ดังภาพประกอบที่ 3.10 บรรทัดที่ 2 และทำการแทนที่สัญลักษณ์ไม่สิ้นสุด <B> ของภาพประกอบที่ 3.10 ด้วยโปรดักชั่นที่ 3 ของภาพประกอบที่ 3.9 ได้ผลลัพธ์เป็นสัญลักษณ์สิ้นสุด float ตามด้วยสัญลักษณ์สิ้นสุด float แสดงดังภาพประกอบที่ 3.10 บรรทัดที่ 3 เป็นกรณีทดสอบ

```
1  <Add> ---> <A><B>
2          ---> float <B>
3          ---> float float
```

ภาพประกอบที่ 3.10 การสร้างกรณีทดสอบของการดำเนินการ Add ด้วยแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้ว

หากสัญลักษณ์ไม่สิ้นสุดที่กำลังทำการแทนที่มีสัญลักษณ์แสดงการแทน หรือสัญลักษณ์แสดงการสลับลำดับปรากฏอยู่ ให้พิจารณาสัญลักษณ์เหล่านั้นตามกรณีต่อไปนี้ก่อนทำการแทนที่ด้วยโปรดักชั่น

1) กรณีสัญลักษณ์ไม่สิ้นสุดมีสัญลักษณ์แสดงการแทน * ให้สร้างสัญลักษณ์ไม่สิ้นสุดนั้นใน 3 ลักษณะคือ (1) สร้างด้วยสัญลักษณ์สิ้นสุด null 1 ครั้ง (2) สร้างด้วยสัญลักษณ์ไม่สิ้นสุดที่กำลังทำการแทนที่อยู่ 1 ครั้ง และ (3) สร้างด้วยสัญลักษณ์ไม่สิ้นสุดที่กำลังทำการแทนที่อยู่ 2 ครั้ง โดยแยกลักษณะทั้งสามด้วยสัญลักษณ์แสดงการเลือก (|) ตัวอย่างการสร้างสัญลักษณ์ไม่สิ้นสุดซึ่งมีสัญลักษณ์แสดงการแทน * แสดงดังภาพประกอบที่ 3.11

```
<books>      ---> <book>*
             ---> null | <book> | <book><book>
```

ภาพประกอบที่ 3.11 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน *

2) กรณีสัญลักษณ์ไม่สิ้นสุดมีสัญลักษณ์แสดงการแทน ? ให้สร้างสัญลักษณ์ไม่สิ้นสุดนั้นใน 2 ลักษณะคือ (1) สร้างด้วยสัญลักษณ์สิ้นสุด null 1 ครั้ง และ (2) สร้างด้วยสัญลักษณ์ไม่สิ้นสุดที่กำลังทำการแทนที่อยู่ 1 ครั้ง โดยแยกลักษณะทั้งสองด้วยสัญลักษณ์แสดงการเลือก ตัวอย่างการสร้างสัญลักษณ์ไม่สิ้นสุดซึ่งมีสัญลักษณ์แสดงการแทน ? แสดงดังภาพประกอบที่ 3.12

```
<addresses>  ---> <address>?
             ---> null | <address>
```

ภาพประกอบที่ 3.12 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน ?

3) กรณีสัญลักษณ์ไม่สิ้นสุดมีสัญลักษณ์แสดงการแทน + ให้สร้างสัญลักษณ์ไม่สิ้นสุดที่กำลังทำการ derive อยู่ 1 ครั้ง ตัวอย่างการสร้างสัญลักษณ์ไม่สิ้นสุดซึ่งมีสัญลักษณ์แสดงการแทน + แสดงดังภาพประกอบที่ 3.13

```
<contact>      ---> <telephone>+
               ---> <telephone>
```

ภาพประกอบที่ 3.13 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการแทน +

4) กรณีสัญลักษณ์ไม่สิ้นสุดมีสัญลักษณ์แสดงการสลับลำดับ ให้สร้างสัญลักษณ์ใน 2 ลักษณะคือ (1) สร้างชุดของสัญลักษณ์ไม่สิ้นสุดตามลำดับที่ปรากฏ และ (2) สร้างชุดของสัญลักษณ์ไม่สิ้นสุดที่สลับลำดับระหว่างสัญลักษณ์ไม่สิ้นสุดตัวแรกและตัวสุดท้ายโดยแยกลักษณะทั้งสองด้วยสัญลักษณ์แสดงการเลือก ตัวอย่างการสร้างสัญลักษณ์ไม่สิ้นสุดซึ่งมีสัญลักษณ์แสดงการสลับลำดับแสดงดังภาพประกอบที่ 3.14

```
<symbol>       ---> (<AA><BB><CC><DD>)
               ---> <AA><BB><CC><DD> | <DD><BB><CC><AA>
```

ภาพประกอบที่ 3.14 การสร้างสัญลักษณ์ไม่สิ้นสุดที่มีสัญลักษณ์แสดงการสลับลำดับ

บทที่ 4

การออกแบบและพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบ

เนื้อหาในบทนี้กล่าวถึงการออกแบบและพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิสเพื่อสนับสนุนแนวคิดที่นำเสนอในงานวิจัย โดยใช้ภาษาจาวาและเครื่องมือสนับสนุนการทำงาน Netbeans 6.8 ในการพัฒนา

4.1 การออกแบบเครื่องมือสำหรับสร้างกรณีทดสอบ

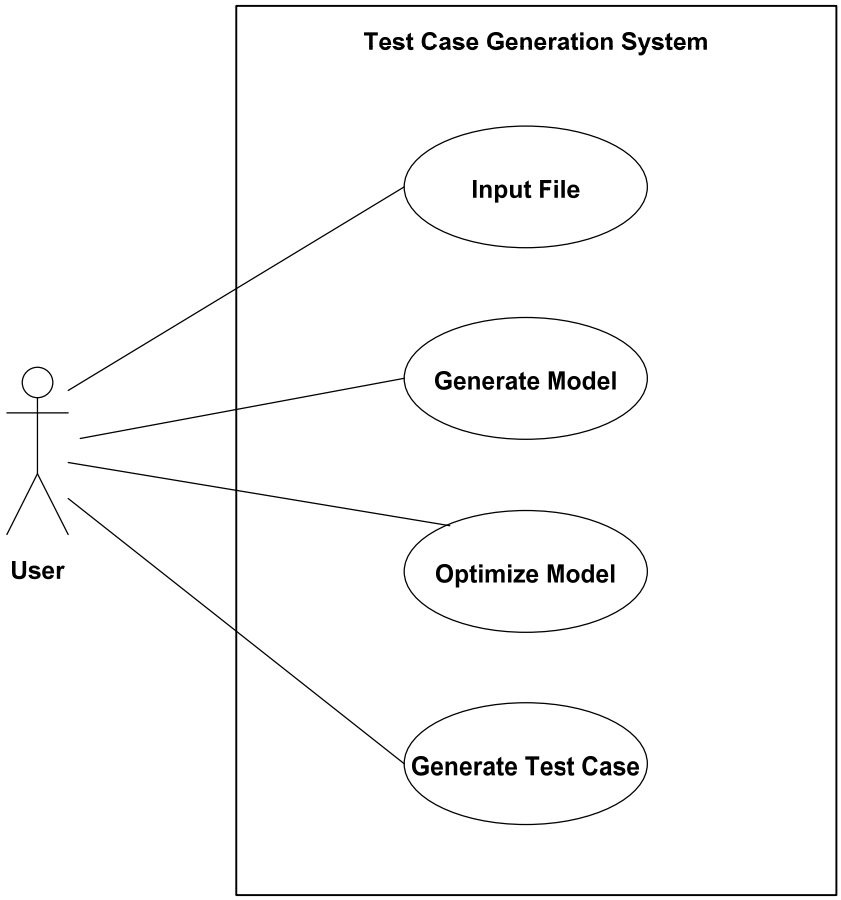
เครื่องมือสร้างกรณีทดสอบสำหรับทดสอบเว็บเซอร์วิสมีส่วนประกอบที่เกี่ยวข้อง 4 ส่วนคือ 1) ส่วนนำไฟล์เข้าสู่ระบบ (Input File) 2) ส่วนสร้างแบบจำลอง (Generate Model) 3) ส่วนปรับปรุงแบบจำลอง (Optimize Model) และ 4) ส่วนสร้างกรณีทดสอบ (Generate Test Case) ภาพประกอบที่ 4.1 แสดงแผนภาพยูสเคส (Use Case Diagram) การทำงานของเครื่องมือสำหรับสร้างกรณีทดสอบสำหรับเว็บเซอร์วิส โดยมีรายละเอียดต่างๆ ดังนี้

1) ยูสเคสนำไฟล์เข้าสู่ระบบ เป็นยูสเคสที่เลือกอินเตอร์มีเดียทโอเปอร์เรชั่นซึ่งเป็นเอกสาร XML เข้าสู่ระบบ โดยเอกสารอินเตอร์มีเดียทโอเปอร์เรชั่นจะถูกจัดเตรียมด้วยมือตามขั้นตอนการสกัดข้อมูลในหัวข้อ 3.2 คำอธิบายยูสเคสนำไฟล์เข้าสู่ระบบแสดงดังตารางที่ 4.1

2) ยูสเคสสร้างแบบจำลอง เป็นยูสเคสที่ทำหน้าที่อ่านเอกสาร XML ให้อยู่ในรูปโครงสร้างต้นไม้ และสร้างเป็นแบบจำลองเชิงไวยากรณ์ คำอธิบายยูสเคสสร้างแบบจำลองแสดงดังตารางที่ 4.2

3) ยูสเคสปรับปรุงแบบจำลอง เป็นยูสเคสที่ทำหน้าที่นำแบบจำลองเชิงไวยากรณ์ที่สร้างขึ้นมาทำการปรับปรุง โดยกำจัดส่วนที่ไม่จำเป็นออกไป คำอธิบายยูสเคสปรับปรุงแบบจำลองแสดงดังตารางที่ 4.3

4) ยูสเคสสร้างกรณีทดสอบ เป็นยูสเคสที่ทำหน้าที่นำแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วมาสร้างกรณีทดสอบ คำอธิบายยูสเคสสร้างกรณีทดสอบแสดงดังตารางที่ 4.4



ภาพประกอบที่ 4.1 แผนภาพการทำงานของเครื่องมือสำหรับสร้างกรณีทดสอบ

ตารางที่ 4.1 คำอธิบายยูสเคสนำไฟล์เข้าสู่ระบบ

ชื่อ	Input File
เป้าหมาย	เพื่อให้ผู้ใช้งานเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นที่ต้องการสร้างแบบจำลองเชิงไวยากรณ์
ผู้ดำเนินการ	ผู้ใช้งาน (user)
การดำเนินการ	ผู้ใช้งานเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นที่ต้องการสร้างแบบจำลองเชิงไวยากรณ์
เงื่อนไขหรือข้อยกเว้น	อินเตอร์มีเดียทโอเปอร์เรชั่นต้องเป็นเอกสาร XML เท่านั้น
ผลลัพธ์	ที่อยู่ของอินเตอร์มีเดียทโอเปอร์เรชั่นที่เลือกเข้าสู่ระบบ

ตารางที่ 4.2 คำอธิบายยูสเคสสร้างแบบจำลอง

ชื่อ	Generate Model
เป้าหมาย	เพื่อให้ผู้ใช้งานสั่งให้ระบบสร้างแบบจำลองเชิงไวยากรณ์
ผู้ดำเนินการ	ผู้ใช้งาน (user)
การดำเนินการ	ผู้ใช้งานสั่งให้ระบบสร้างแบบจำลองเชิงไวยากรณ์
เงื่อนไขหรือข้อยกเว้น	มีการนำเข้าอินเตอร์มีเดียทโอเปอร์เรชั่น
ผลลัพธ์	แบบจำลองเชิงไวยากรณ์

ตารางที่ 4.3 คำอธิบายยูสเคสปรับปรุงแบบจำลอง

ชื่อ	Optimize Model
เป้าหมาย	เพื่อให้ผู้ใช้งานสั่งให้ระบบปรับปรุงแบบจำลองเชิงไวยากรณ์
ผู้ดำเนินการ	ผู้ใช้งาน (user)
การดำเนินการ	ผู้ใช้งานสั่งให้ระบบปรับปรุงแบบจำลองเชิงไวยากรณ์
เงื่อนไขหรือข้อยกเว้น	มีการสั่งให้ระบบสร้างแบบจำลองเชิงไวยากรณ์
ผลลัพธ์	แบบจำลองเชิงไวยากรณ์ที่ผ่านการปรับปรุงแล้ว

ตารางที่ 4.4 คำอธิบายยูสเคสสร้างกรณีทดสอบ

ชื่อ	Generate Model
เป้าหมาย	เพื่อให้ผู้ใช้งานสั่งให้ระบบสร้างกรณีทดสอบ
ผู้ดำเนินการ	ผู้ใช้งาน (user)
การดำเนินการ	ผู้ใช้งานสั่งให้ระบบสร้างกรณีทดสอบ
เงื่อนไขหรือข้อยกเว้น	มีการสั่งให้ระบบปรับปรุงแบบจำลองเชิงไวยากรณ์
ผลลัพธ์	กรณีทดสอบ

4.2 การพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบ

การพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบสามารถอธิบายส่วนของขั้นตอนการทำงานของเครื่องมือซึ่งแบ่งออกเป็น 4 ขั้นตอนหลักดังนี้

4.2.1 ขั้นตอนการนำไฟล์เข้าสู่ระบบ

เลือกอินเตอร์มีเดียทโอเปอร์เรชั่นซึ่งเป็นเอกสาร XML เข้าสู่ระบบ จากนั้นระบบจะแสดงตำแหน่งที่อยู่ของอินเตอร์มีเดียทโอเปอร์เรชั่นที่ได้เลือกเข้าสู่ระบบ

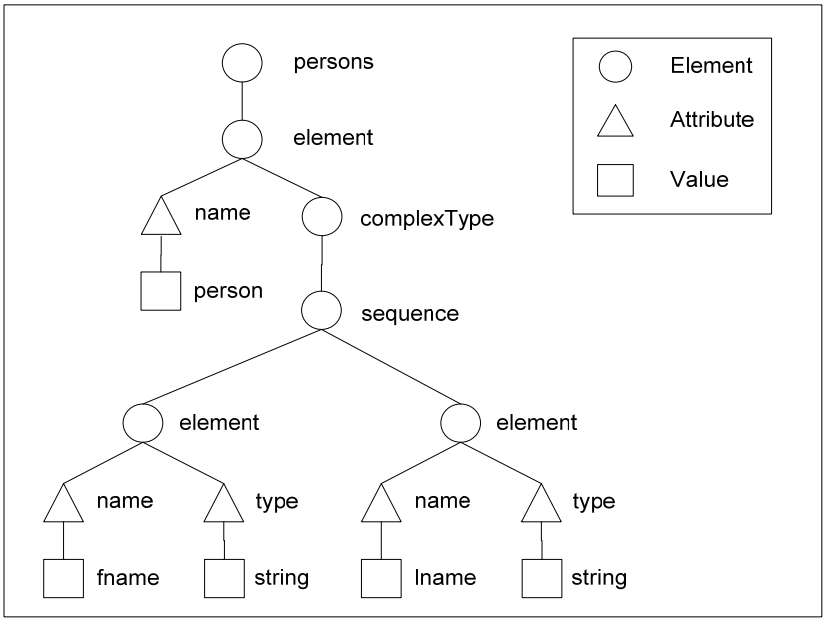
4.2.2 ขั้นตอนการสร้างแบบจำลอง

ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์แบ่งขั้นตอนการทำงานออกเป็น 3 ขั้นตอนย่อยดังนี้

1) ระบบอ่านอินเตอร์มีเดียทโอเปอร์เรชั่นที่เลือกเข้าสู่ระบบโดยใช้ XML Parser ประเภท Document Object Model (DOM) ซึ่งอินเตอร์มีเดียทโอเปอร์เรชั่นจะถูกแปลงให้อยู่ในรูปโครงสร้างต้นไม้ (XML Tree) ตัวอย่างอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ persons ในภาพประกอบที่ 4.2 ถูกแปลงให้อยู่ในรูปโครงสร้างต้นไม้แสดงดังภาพประกอบที่ 4.3

```
<persons>
    <element name =“person”>
      <complexType>
        <sequence>
          <element name = “fname” type = “string” />
          <element name = “lname” type = “string” />
        </sequence>
      </complexType>
    </element>
</persons>
```

ภาพประกอบที่ 4.2 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ persons



ภาพประกอบที่ 4.3 โครงสร้างต้นไม้ของอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ persons

2) ระบบท่องโครงสร้างต้นไม้ (Traverse) ของอินเตอร์มีเดียทโอเปอร์-เรชั่นเพื่อตรวจสอบว่าอินเตอร์มีเดียทโอเปอร์เรชั่นมีข้อมูลเงื่อนไขหรือไม่ ถ้ามีข้อมูลเงื่อนไข ระบบจะทำการจัดเก็บข้อมูลเงื่อนไขลงในโครงสร้างอาเรย์ดังภาพประกอบที่ 4.4 และขั้นตอนในการเก็บข้อมูลเงื่อนไขแสดงดังภาพประกอบที่ 4.5 หากไม่มีข้อมูลเงื่อนไขระบบจะทำงานต่อในขั้นตอนย่อยถัดไป

```
FACET[1..n]
{
        NAME
        BASE
        REST[1..m]
}
```

ภาพประกอบที่ 4.4 โครงสร้างอาเรย์ที่ใช้เก็บข้อมูลเงื่อนไขของแบบจำลองเชิงไวยากรณ์

```
getFacet (node rootNode)
  child  <---  get first childnode of rootNode
  while child is not empty
    if child is simpleType node
      nameVal  <---  get Value of Attribute name of child
      FACET[n].NAME  =  nameVal
      base_list  <---  get list childnode of child
      length  <---  count amount of base_list
      for i=0 to length-1
        nodeBase  <---  get node at position i
        baseVal  <---  get Value of Attribute base of nodeBase
        FACET[i].BASE  =  baseVal
        length_rest  <--- count amount of rest_list
        for j=0 to length_rest-1
          nodeRest  <---  get node at position j
          nameRest  <---  get name of nodeRest
          restVal  <---  get Value of Attribute base of nodeRest
          strRest  <---  concatenate  nameRest, "=" and restVal
          FACETt[n].REST[m]  =  strRest
          m++
        end for j=0 to length_rest-1
      end for i=0 to length-1
    end if child is simpleType node
    m  =  0
    n++
  end while child is not empty
end
```

ภาพประกอบที่ 4.5 ขั้นตอนการเก็บข้อมูลเงื่อนไขของแบบจำลองเชิงไวยากรณ์

3) ระบบท่องโครงสร้างต้นไม้ของอินเตอร์มีเดียทโอเปอร์เรชั่นเพื่อสร้างแบบจำลองเชิงไวยากรณ์ซึ่งมีขั้นตอนดังภาพประกอบที่ 4.6 โดยเก็บแบบจำลองเชิงไวยากรณ์ลงในโครงสร้างข้อมูลลิงค์ลิสต์ (Linked List) ดังภาพประกอบที่ 4.7 ซึ่งประกอบด้วยฟิลด์ข้อมูล 4 ฟิลด์ ดังนี้

- HEADERNODE คือ ค่าที่บอกว่าสัญลักษณ์ดังกล่าวเป็นสัญลักษณ์ทางซ้ายหรือสัญลักษณ์ทางขวา โดยที่ 0 หมายถึง สัญลักษณ์ทางขวา
1 หมายถึง สัญลักษณ์ทางซ้าย
- NAME คือ ชื่อสัญลักษณ์
- NON_TERM คือ ค่าที่บอกว่าสัญลักษณ์ดังกล่าวเป็นสัญลักษณ์สิ้นสุดหรือสัญลักษณ์ไม่สิ้นสุด โดยที่ 0 หมายถึง สัญลักษณ์สิ้นสุด
1 หมายถึง สัญลักษณ์ไม่สิ้นสุด
- NEXT คือ ตำแหน่งของลิงค์ลิสต์ถัดไป

```
genModel(node Node)
  get nodeName of Node
    if type of node is ELEMENT
      if node has Attribute
        get value of Attribute name, type, minOccurs and maxOccurs
        if count=0
          create LinkedList with HEADERNODE=1, NAME=nodeName and
          NON_TERM=1
        else
          if parentNode of node is choice
            if position of node is even number
```

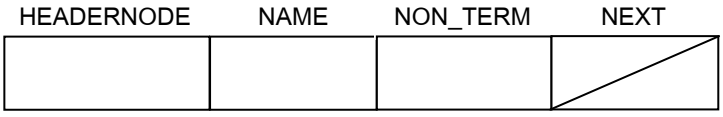
ภาพประกอบที่ 4.6 ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์

```
                create LinkedList with HEADERNODE=0, NAME=nodeName and
                NON_TERM=1
            else
                create LinkedList with HEADERNODE=0, NAME= “|” and
                NON_TERM=1
                create LinkedList with HEADERNODE=0, NAME=nodeName and
                NON_TERM=1
            create LinkedList with HEADERNODE=1, NAME=nodeName and
            NON_TERM=1
        else
            create LinkedList with HEADERNODE=0, NAME=nodeName and
            NON_TERM=1
            If Attribute minOccurs equal 0 and maxOccurs equals 1
                create LinkedList with HEADERNODE=0, NAME= “?” and
                NON_TERM=1
            else If Attribute minOccurs equal 1
                create LinkedList with HEADERNODE=0, NAME= “+” and
                NON_TERM=1
            else If Attribute maxOccurs equals unbounded
                create LinkedList with HEADERNODE=0, NAME= “*” and
                NON_TERM=1
            create LinkedList with HEADERNODE=1, NAME=nodeName and
            NON_TERM=1
    else
        create LinkedList with HEADERNODE=0, NAME=nodeName and
        NON_TERM=1
        create LinkedList with HEADERNODE=1, NAME=nodeName and
        NON_TERM=1
```

ภาพประกอบที่ 4.6 ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์ (ต่อ)

```
            If nodeName equal all
               create LinkedList with HEADERNODE=0, NAME= “(” and
               NON_TERM=0
          If value of Attribute type is primitive type
            create LinkedList with HEADERNODE=0, NAME= value of Attribute type
and
            NON_TERM=0
          else
            find value of NAME of Facet that equal value of Attribute type
            create LinkedList with HEADERNODE=0, NAME= [value of Attribute
type]
            and NON_TERM=0
       end if
       get amount of childNode of Node
       if node has childNode
         for each childNode
           if childNode not equal #text
             genModel()
```

ภาพประกอบที่ 4.6 ขั้นตอนการสร้างแบบจำลองเชิงไวยากรณ์ (ต่อ)



ภาพประกอบที่ 4.7 โครงสร้างลิงค์ลิสต์ที่ใช้เก็บข้อมูลแบบจำลองเชิงไวยากรณ์

4.2.3 ขั้นตอนการปรับปรุงแบบจำลอง

ระบบทำการปรับปรุงแบบจำลองเชิงไวยากรณ์ที่ได้จากขั้นตอนย่อยที่ 3 ของขั้นตอนที่ 4.2.2 โดยมีขั้นตอนการปรับปรุงดังภาพประกอบที่ 4.8

```
optModel (headPtr)
  first, second, tmp1, tmp2  =  null
  if LinkedList is not empty
    first  <---  headPtr
    second  <---  first.next
    while first and second are not empty
      head1  <---  get HEADERNODE value of first
      head2  <---  get HEADERNODE value of second
      if head1 equal head2
        tmp1  <---  first
        first  <---  second
        second  <---  second.next
      else
        name1  <---  get NAME value of first
        name2  <---  get NAME value of second
        if name1 equal name2
          tmp2  <---  second.next
          tmp1.setNext(tmp2)
          tmp2  <---  null
          first  <---  tmp1
          second  <--- first.next
        else
          tmp1  <---  first
          first  <---  second
          second  <---  second.next
    end while first and second are not empty
end
```

ภาพประกอบที่ 4.8 ขั้นตอนการปรับปรุงแบบจำลองเชิงไวยากรณ์

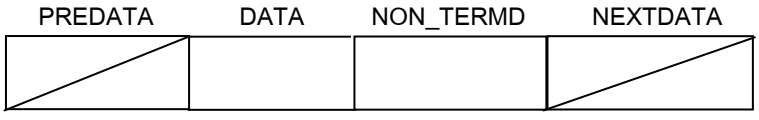
4.2.4 ขั้นตอนการสร้างกรณีทดสอบ

ระบบทำการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ซึ่งมีขั้นตอนการการสร้างกรณีทดสอบดังภาพประกอบที่ 4.9 โดยเก็บกรณีทดสอบลงในโครงสร้างข้อมูลลิงค์ลิสต์ดังภาพประกอบที่ 4.10 ซึ่งประกอบด้วยฟิลด์ข้อมูล 4 ฟิลด์ ดังนี้

- PREDATA คือ ตำแหน่งของลิงค์ลิสต์ก่อนหน้า
- DATA คือ ชื่อสัญลักษณ์
- NON_TERMD คือ ค่าที่บอกว่าสัญลักษณ์ดังกล่าวเป็นสัญลักษณ์สิ้นสุดหรือสัญลักษณ์ไม่สิ้นสุด โดยที่ 0 หมายถึง สัญลักษณ์สิ้นสุด
  1 หมายถึง สัญลักษณ์ไม่สิ้นสุด
- NEXTDATA คือ ตำแหน่งของลิงค์ลิสต์ถัดไป

```
genTC(Rule nodeRule)
   head, current  =  null
   NAMED  <---  get NAME value of first LinkedList of nodeRule
   NON_TERMD  <---  get NON_TERM value of first LinkedList of nodeRule
   create TestCase with NAMED and NON_TERMD value
   head  =  current  =  TestCase
   while current NON_TERMD equal 1
      if current NON_TERMD equal 1
         find LinkedList of nodeRule
         if nodeRule.NAME equal current.NAMED
            nodeRule  <---  nodeRule.next
            while nodeRule.HADERNODE equal 0
               NAMED  <---  get NAME value of LinkedList of nodeRule
               NON_TERMD  <---  get NON_TERM value of LinkedList of nodeRule
               create TestCase with NAMED and NON_TERMD value
            end while nodeRule.HEADERNODE equal 0
         end if nodeRule.NAME equal current.NAMED
      else
         current  <---  current.next
   end while current NON_TERMD equal 1
end
```

ภาพประกอบที่ 4.9 ขั้นตอนการสร้างกรณีทดสอบ



ภาพประกอบที่ 4.10 โครงสร้างลิงค์ลิสต์ที่ใช้เก็บกรณีทดสอบ

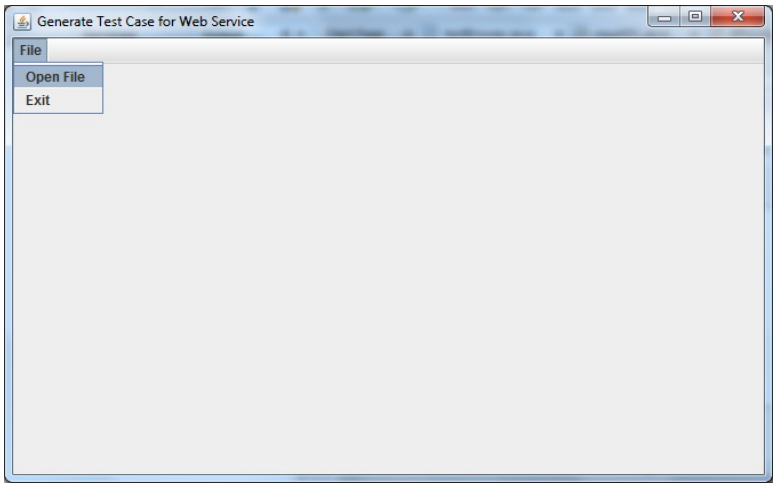
4.3 การออกแบบส่วนติดต่อผู้ใช้

เครื่องมือสำหรับสร้างกรณีทดสอบประกอบด้วยส่วนติดต่อผู้ใช้ 3 ส่วนคือ 1) ส่วนการเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นเข้าสู่ระบบ 2) ส่วนการสร้างแบบจำลองเชิงไวยากรณ์ และ 3) ส่วนการออกจากระบบ โดยมีรายละเอียดของแต่ละส่วนดังต่อไปนี้

4.3.1 ส่วนการเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นเข้าสู่ระบบ

ส่วนการเลือกอินเตอร์มีเดียทโอเปอร์เรชั่นเข้าสู่ระบบมีรายละเอียดดังนี้

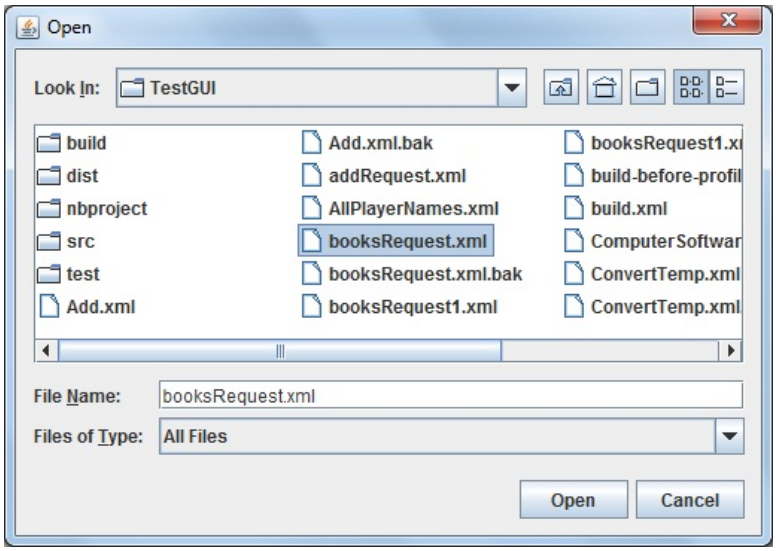
1) หน้าจอเริ่มต้นระบบแสดงแถบเมนู File เมื่อต้องการเลือกอินเตอร์มีเดีย-โอเปอร์เรชั่นเข้าสู่ระบบ ให้เลือกเมนู Open File ดังภาพประกอบที่ 4.11



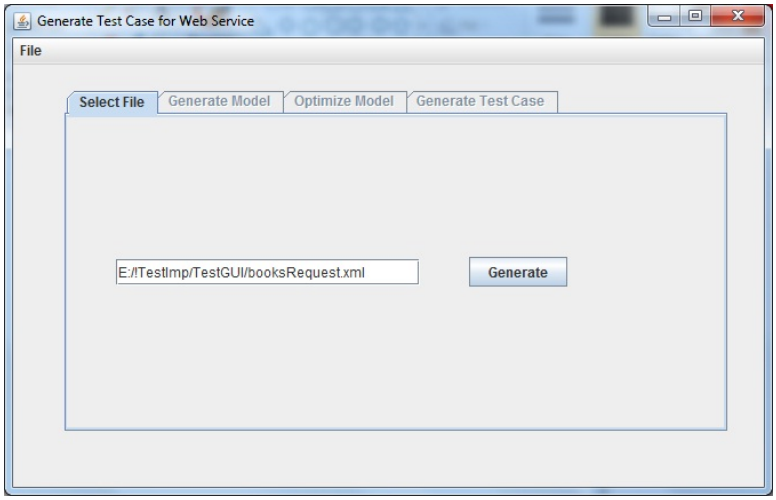
ภาพประกอบที่ 4.11 หน้าจอเริ่มต้นของระบบ

2) เมื่อเลือกเมนู Open File จะปรากฏหน้าต่างเลือกอินเตอร์มีเดียโอ-เปอร์เรชั่น โดยอินเตอร์มีเดียทโอเปอร์เรชั่นที่ถูกเลือกจะปรากฏชื่อไฟล์ในช่อง File Name ดังภาพประกอบที่ 4.12 ผู้ใช้สามารถกดปุ่ม Open เพื่อยืนยันการเลือกอินเตอร์มีเดียทโอเปอร์เรชั่น

3) เมื่อกดปุ่ม Open ในภาพประกอบที่ 4.12 จะปรากฏแท็บ 4 แท็บ คือ (1) แท็บ Select File (2) แท็บ Generate Model (3) แท็บ Optimize Model และ (4) แท็บ Generate Test Case โดยเริ่มต้นผู้ใช้จะใช้งานได้เฉพาะแท็บ Select File เท่านั้น ส่วนแท็บอื่นๆ จะสามารถใช้งานได้ในขั้นตอนถัดๆ ไป ภายในแท็บ Select File จะแสดงตำแหน่งของอินเตอร์มีเดียทโอเปอร์เรชั่นที่ถูกเลือกเข้าสู่ระบบดังภาพประกอบที่ 4.13 ผู้ใช้สามารถกดปุ่ม Generate เพื่อสร้างแบบจำลองเชิงไวยากรณ์



ภาพประกอบที่ 4.12 หน้าต่างเลือกอินเตอร์มีเดียทโอเปอร์เรชั่น



ภาพประกอบที่ 4.13 แท็บ Select File แสดงตำแหน่งของอินเตอร์มีเดียทโอเปอร์เรชั่นที่เลือกเข้าสู่ระบบ

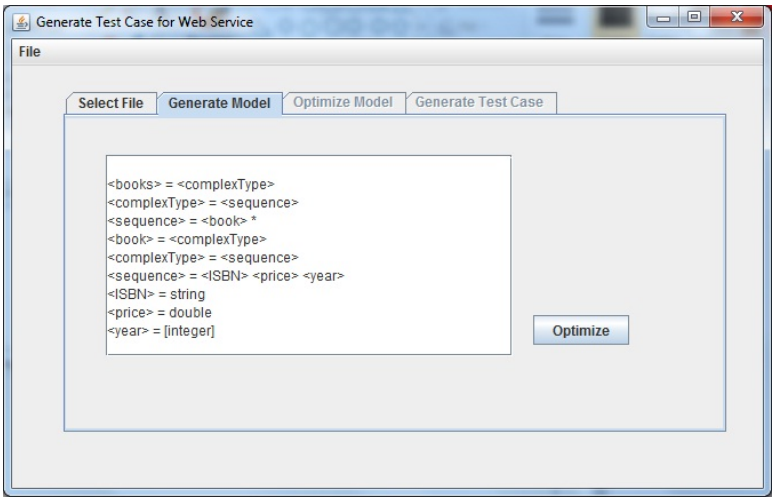
4.3.2 ส่วนการสร้างแบบจำลองเชิงไวยากรณ์และกรณีทดสอบ

ส่วนการสร้างแบบจำลองเชิงไวยากรณ์และกรณีทดสอบมีรายละเอียดดังนี้

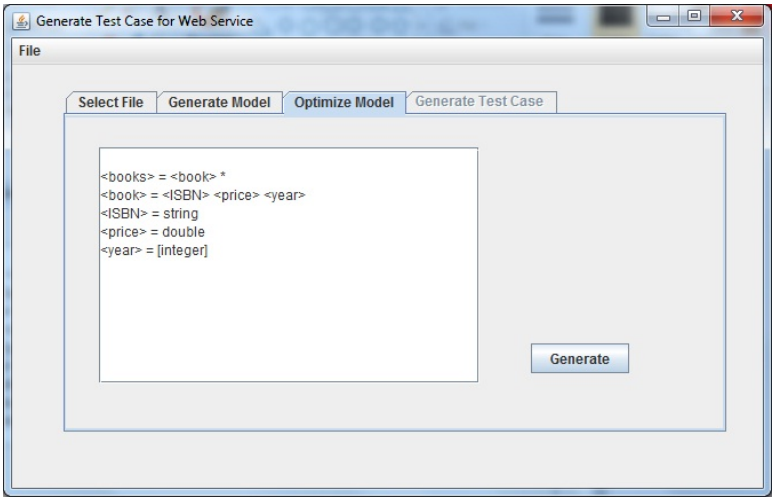
1) เมื่อกดปุ่ม Generate ในภาพประกอบที่ 4.13 ระบบจะเข้าสู่ส่วนการสร้างแบบจำลองเชิงไวยากรณ์ โดยจะปรากฏแท็บ Generate Model ซึ่งแสดงแบบจำลองเชิงไวยากรณ์ที่สร้างจากอินเตอร์มีเดียทโอเปอร์เรชั่นที่ถูกเลือกในขั้นตอน 4.3.1 ดังภาพประกอบที่

4.14 สำหรับแท็บ Optimize Model และแท็บ Generate Test Case ยังไม่สามารถทำงานได้ในขั้นตอนนี้ ผู้ใช้สามารถกดปุ่ม Optimize เพื่อปรับปรุงแบบจำลองเชิงไวยากรณ์

2) เมื่อกดปุ่ม Optimize ในภาพประกอบที่ 4.14 ระบบจะทำการปรับปรุงแบบจำลองเชิงไวยากรณ์ และปรากฏแท็บ Optimize Model ซึ่งแสดงแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วดังภาพประกอบที่ 4.15 สำหรับแท็บ Generate Test Case ยังไม่สามารถทำงานได้ในขั้นตอนนี้ ผู้ใช้สามารถกดปุ่ม Generate เพื่อสร้างกรณีทดสอบ

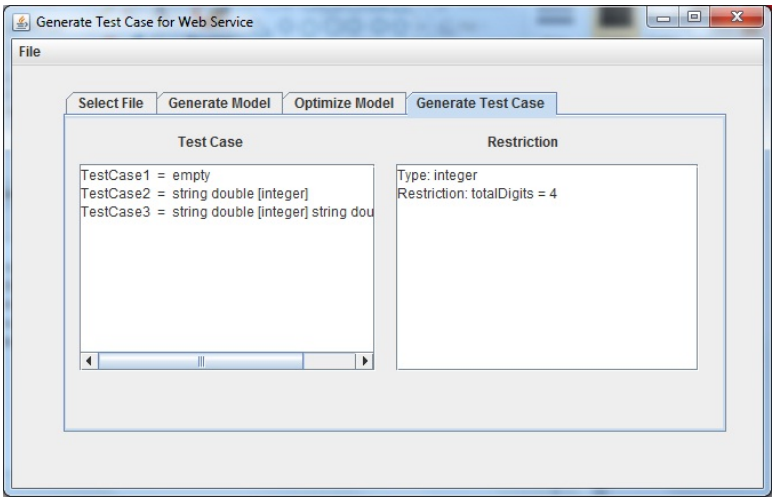


ภาพประกอบที่ 4.14 แท็บ Generate Model แสดงแบบจำลองเชิงไวยากรณ์



ภาพประกอบที่ 4.15 แท็บ Optimize Model แสดงแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้ว

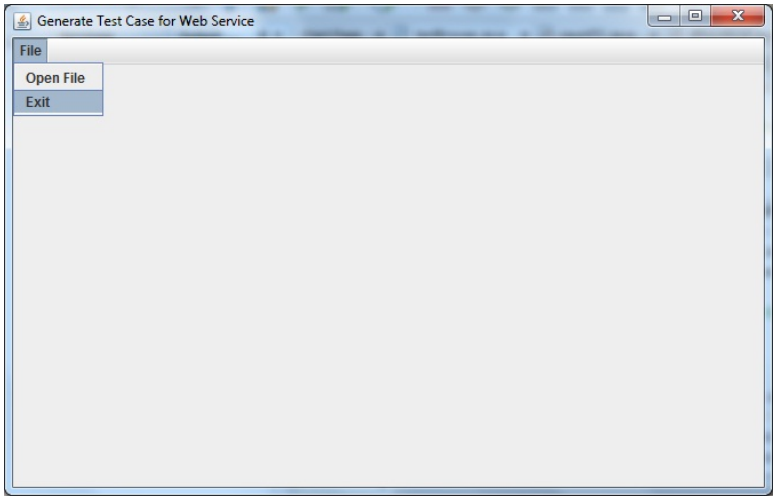
3) เมื่อกดปุ่ม Generate ในภาพประกอบที่ 4.15 ระบบจะทำการสร้างกรณีทดสอบ และปรากฏแท็บ Generate Test Case ดังแสดงในภาพประกอบที่ 4.16 ซึ่งแสดงกรณีทดสอบในช่อง Test Case ทางด้านซ้ายมือ หากมีเงื่อนไขจะปรากฏข้อมูลของเงื่อนไขในช่อง Restriction ทางด้านขวามือ



ภาพประกอบที่ 4.16 แท็บ Generate Test Case แสดงกรณีทดสอบและเงื่อนไขของกรณีทดสอบ

4.3.3 ส่วนการออกจากระบบ

เมื่อผู้ใช้ต้องการออกจากระบบให้เลือกแถบเมนู File และเลือกเมนู Exit ดังภาพประกอบที่ 4.17 จากนั้นจะปรากฏหน้าจอเพื่อยืนยันการออกจากระบบดังภาพประกอบที่ 4.18 ผู้ใช้สามารถกดปุ่ม Exit เพื่อยืนยันการออกจากระบบ และกดปุ่ม No เพื่อยกเลิกการออกจากระบบ



ภาพประกอบที่ 4.17 หน้าจอการออกจากระบบ

ภาพประกอบที่ 4.18 หน้าจอยืนยันการออกจากระบบ

บทที่ 5

การทดสอบเว็บเซอร์วิส

เนื้อหาในบทนี้กล่าวถึงการทดสอบเว็บเซอร์วิสจากกรณีทดสอบที่สร้างขึ้นตามวิธีการที่ได้นำเสนอ โดยกล่าวถึงการกำหนดค่าที่ใช้ทดสอบซึ่งพิจารณาจากสัญลักษณ์สิ้นสุดทุกตัวในกรณีทดสอบ เว็บเซอร์วิสที่ใช้ในการทดสอบ แนวทางการทดสอบ รายละเอียดของการสร้างกรณีทดสอบด้วยวิธีการที่นำเสนอและวิธีการของ Samer และทำการประเมินความสามารถในการจับข้อผิดพลาดของกรณีทดสอบที่สร้างด้วยวิธีการที่ได้นำเสนอเทียบกับวิธีการของ Samer

5.1 การกำหนดค่าที่ใช้ทดสอบของกรณีทดสอบ

จากกรณีทดสอบที่ได้จากขั้นตอนที่ 3 ในหัวข้อ 3.2 จะถูกนำมากำหนดค่าที่ใช้สำหรับทดสอบเว็บเซอร์วิส ซึ่งพิจารณาจากสัญลักษณ์สิ้นสุดทุกตัวที่ได้จากการแทนที่สัญลักษณ์ไม่สิ้นสุดด้วยโปรดักชั่นจากแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้ว โดยสัญลักษณ์สิ้นสุดแต่ละตัวจะถูกแทนด้วยค่าที่ถูกต้องและค่าที่ไม่ถูกต้องตามสัญลักษณ์สิ้นสุดนั้น ยกเว้นสัญลักษณ์สิ้นสุด null ซึ่งมีเพียงค่าเดียวคือค่าที่ถูกต้อง

5.1.1 การแทนค่าที่ถูกต้อง

การแทนค่าที่ถูกต้องจะพิจารณาตามสัญลักษณ์สิ้นสุดดังนี้

1) กรณีสัญลักษณ์สิ้นสุดอยู่ภายในเครื่องหมาย “ ” ให้ใช้ค่าที่อยู่ภายในเครื่องหมาย “ ” เป็นค่าที่ถูกต้อง เช่น สัญลักษณ์สิ้นสุดคือ “red” ค่าที่ถูกต้องคือ red เป็นต้น

2) กรณีสัญลักษณ์สิ้นสุดเป็นชนิดข้อมูลที่กำหนดมาให้แล้วและไม่มีเงื่อนไขจำกัดในตารางเงื่อนไขจำกัด ให้พิจารณาดังนี้

2.1) ถ้าเป็นชนิดข้อมูลแบบตัวเลขดังตารางที่ 5.1 ให้ใช้ค่าศูนย์ ค่าสูงสุด และค่าต่ำสุดของชนิดข้อมูลนั้นๆ เป็นค่าที่ถูกต้อง เช่น สัญลักษณ์สิ้นสุดคือ float ค่าที่ถูกต้องคือ 0, 3.4028235E38 และ -3.4028235E38 เป็นต้น

2.2) ถ้าเป็นชนิดข้อมูลนอกเหนือจากข้อ 2.1 ให้สร้างค่าที่ถูกต้องโดยการสุ่มค่าตามชนิดข้อมูลนั้น เช่น สัญลักษณ์สิ้นสุดคือ string ค่าที่ถูกต้องคือ abcdef เป็นต้น

ตารางที่ 5.1 ค่าสูงสุดและค่าต่ำสุดของชนิดข้อมูลแบบตัวเลข (Dan Vint, 2010)

ชนิดข้อมูล	ค่าสูงสุด (Maximum Value)	ค่าต่ำสุด (Minimum Value)
byte	127	-128
double	1.7976931348623157E308	-1.7976931348623157E308
float	3.4028235E38	-3.4028235E38
int	2147483647	-2147483648
long	9223372036854775807	-9223372036854775808
short	32767	-32768
unsignedByte	255	0
unsignedInt	4294967295	0
unsignedLong	18446744073709551615	0
unsignedShort	65535	0

3) กรณีสัญลักษณ์สิ้นสุดเป็นชนิดข้อมูลที่กำหนดมาให้แล้วและมีเงื่อนไขจำกัดในตารางเงื่อนไขจำกัด ให้พิจารณาดังนี้

3.1) ถ้าเป็นเงื่อนไขจำกัดกลุ่มของค่า ได้แก่ enumeration ให้สร้างค่าที่ถูกต้องโดยใช้ค่าที่เป็นไปได้ทุกค่าของเงื่อนไขจำกัดกลุ่มของค่า ตัวอย่างตารางเงื่อนไขจำกัดซึ่งเป็นเงื่อนไขจำกัดกลุ่มของค่าแสดงดังตารางที่ 5.2

ตารางที่ 5.2 ตารางเงื่อนไขจำกัดซึ่งเป็นเงื่อนไขจำกัดกลุ่มของค่า

ชนิดข้อมูล	เงื่อนไขจำกัด
string	enumeration = Bit, enumeration = Byte, enumeration = Kilobyte

จากตารางที่ 5.2 สัญลักษณ์สิ้นสุดมีชนิดข้อมูล string และมีค่าที่เป็นไปได้ของเงื่อนไขจำกัดกลุ่มของค่า 3 ค่าคือ Bit Byte และ Kilobyte ดังนั้นค่าที่ถูกต้องคือ Bit Byte และ Kilobyte

3.2) ถ้าเป็นเงื่อนไขจำกัดอื่นๆ ให้สร้างค่าที่ถูกต้องตามชนิดข้อมูลที่กำหนดมาให้และเงื่อนไขจำกัดนั้น เช่น จากเงื่อนไขจำกัดในตารางที่ 3.2 สัญลักษณ์สิ้นสุดมีชนิดข้อมูลเป็น integer และมีเงื่อนไขจำกัด totalDigits = 4 ดังนั้นค่าที่ถูกต้องคือ 1234 เป็นต้น

5.1.2 การแทนค่าที่ไม่ถูกต้อง

การแทนค่าที่ไม่ถูกต้องจะพิจารณาตามสัญลักษณ์สิ้นสุดดังนี้

1) กรณีสัญลักษณ์สิ้นสุดอยู่ภายในเครื่องหมาย “ ” ให้แทนด้วยค่าที่ไม่ใช่ค่าที่อยู่ภายในเครื่องหมาย “ ” เช่น สัญลักษณ์สิ้นสุดคือ “red” ค่าที่ไม่ถูกต้องคือ “der” เป็นต้น

2) กรณีสัญลักษณ์สิ้นสุดเป็นชนิดข้อมูลที่กำหนดมาให้แล้วให้แทนค่าที่ไม่ถูกต้องดังนี้

2.1) ถ้าเป็นชนิดข้อมูลแบบตัวเลข ให้แทนค่าที่ไม่ถูกต้องเป็นลำดับของตัวอักษร เช่น สัญลักษณ์สิ้นสุดคือ float ค่าที่ไม่ถูกต้องคือ “qsefth” เป็นต้น

2.2) ถ้าเป็นชนิดข้อมูลเป็น string ให้แทนค่าที่ไม่ถูกต้องเป็นลำดับของตัวเลข เช่น สัญลักษณ์สิ้นสุดคือ string ค่าที่ไม่ถูกต้องคือ 1283 เป็นต้น

เมื่อแทนค่าที่ถูกต้องและไม่ถูกต้องให้กับสัญลักษณ์สิ้นสุดแต่ละตัวแล้วจำนวนกรณีทดสอบที่ได้จะเท่ากับผลคูณคาร์ทิเชียนของจำนวนกรณีทั้งหมดของสัญลักษณ์สิ้นสุด เช่น มีสัญลักษณ์สิ้นสุด 2 ตัว ซึ่งมีจำนวนค่าที่ถูกต้องและไม่ถูกต้องเท่ากับ 4 และ 2 ตามลำดับ ดังนั้นจำนวนกรณีทดสอบจะเท่ากับ 4x2 = 8 กรณี เป็นต้น

5.2 เว็บเซอร์วิสที่ใช้ในการทดสอบ

ปัจจุบันมีเว็บเซอร์วิสที่ให้บริการการทำงานต่างๆ มากมาย และจากการศึกษาเอกสาร WSDL ของแต่ละเว็บเซอร์วิสพบว่า ในส่วนของอิลิเมนต์ <types> มักจะอธิบายอิลิเมนต์ที่เป็นข้อมูลนำเข้าของการดำเนินการด้วยอิลิเมนต์แบบชนิดข้อมูลแบบซับซ้อน มีการระบุลำดับของอิลิเมนต์เป็นแบบ sequence และจำนวนครั้งของการเกิดอิลิเมนต์มากที่สุดและน้อยที่สุดจะเท่ากับหนึ่ง ดังนั้นผู้วิจัยจึงได้เลือกเว็บเซอร์วิส ConvertTemperature (ConvertTemp, 2011) และเว็บเซอร์วิส MathService (Divide, 2011) มาใช้ในการทดสอบ โดยมีรายละเอียดของแต่ละเว็บเซอร์วิสดังนี้

5.2.1 เว็บเซอร์วิส ConvertTemperature

เว็บเซอร์วิส ConvertTemperature มีการดำเนินการชื่อ ConvertTemp เป็นเว็บเซอร์วิสที่ให้บริการเกี่ยวกับการแปลงค่าอุณหภูมิจากหน่วยหนึ่งไปเป็นอีกหน่วยหนึ่ง เช่น แปลงจากหน่วยองศาเซลเซียสเป็นหน่วยองศาฟาเรนไฮต์ เป็นต้น โดยรับค่าของอุณหภูมิ หน่วยของอุณหภูมิก่อนแปลง และหน่วยของอุณหภูมิหลังแปลง จากนั้นเว็บเซอร์วิสจะทำการคำนวณและส่งผลลัพธ์ค่าอุณหภูมิซึ่งแปลงค่าตามหน่วยของอุณหภูมิที่ต้องการกลับมายังผู้ใช้

5.2.2 เว็บเซอร์วิส MathService

เว็บเซอร์วิส MathService เป็นเว็บเซอร์วิสที่ให้บริการเกี่ยวกับการบวก ลบ คูณ และหาร โดยรับค่าของจำนวน 2 จำนวนเพื่อทำการคำนวณค่า จากนั้นเว็บเซอร์วิสจะทำการคำนวณและส่งผลลัพธ์กลับมายังผู้ใช้ โดยในการทดสอบผู้วิจัยได้เลือกการดำเนินการ Divide มาใช้ในการทดสอบ

5.3 แนวทางการทดสอบ

การทดสอบเว็บเซอร์วิสทำขึ้นเพื่อเปรียบเทียบประสิทธิภาพของกรณีทดสอบที่สร้างจากวิธีการที่นำเสนอกับวิธีการสร้างกรณีทดสอบของ Samer ซึ่งทำการสร้างกรณีทดสอบโดยใช้การวิเคราะห์ค่าขอบเขต และจำนวนกรณีทดสอบทั้งหมดหาได้จากผลคูณคาร์ทีเชียนของจำนวนกรณีทดสอบในแต่ละพารามิเตอร์

แนวทางในการทดสอบเว็บเซอร์วิสในงานวิจัยนี้มีขั้นตอนการดำเนินงานดังต่อไปนี้

1) นำเว็บเซอร์วิสในหัวข้อ 5.2 มาทำการสร้างกรณีทดสอบโดยใช้เครื่องมือที่พัฒนาขึ้น จากนั้นสร้างค่าที่ใช้ทดสอบจากกรณีทดสอบ

2) นำเว็บเซอร์วิสในหัวข้อ 5.2 มาทำการสร้างกรณีทดสอบตามวิธีการของ Samer

3) ทำการทดสอบเว็บเซอร์วิส ConvertTemperature และเว็บเซอร์วิส MathService ด้วยกรณีทดสอบที่สร้างขึ้นจากวิธีการที่นำเสนอและวิธีการของ Samer จากนั้นทำการเปรียบเทียบจำนวนกรณีทดสอบที่สามารถจับข้อผิดพลาดจากการทำงานของเว็บเซอร์วิสทั้งสอง

5.4 การสร้างกรณีทดสอบตามวิธีการที่นำเสนอ

การสร้างกรณีทดสอบตามวิธีการที่นำเสนอสำหรับเว็บเซอร์วิส ConvertTemperature และเว็บเซอร์วิส MathService มีรายละเอียดดังนี้

1) เว็บเซอร์วิส ConvertTemperature

ขั้นตอนในการสร้างกรณีทดสอบของการดำเนินการ ConvertTemp ของเว็บเซอร์วิส ConvertTemperature ตามวิธีการที่นำเสนอมีรายละเอียดในแต่ละขั้นตอนดังนี้

(1) จากการวิเคราะห์เอกสาร WSDL ของเว็บเซอร์วิส ConvertTemperature สามารถสกัดอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ ConvertTemp ซึ่งมีพารามิเตอร์ 3 ตัวคือ Temperature FromUnit และ ToUnit แสดงดังภาพประกอบที่ 5.1

```
<ConvertTemp>
  <element name="ConvertTemp">
    <complexType>
      <sequence>
        <element minOccurs="1" maxOccurs="1" name="Temperature"
                 type="double" />
        <element minOccurs="1" maxOccurs="1" name="FromUnit"
                 type="TemperatureUnit" />
        <element minOccurs="1" maxOccurs="1" name="ToUnit"
                 type="TemperatureUnit" />
      </sequence>
    </complexType>
  </element>
  <simpleType name="TemperatureUnit">
    <restriction base="string">
      <enumeration value="degreeCelsius" />
      <enumeration value="degreeFahrenheit" />
      <enumeration value="degreeRankine" />
      <enumeration value="degreeReaumur" />
      <enumeration value="kelvin" />
    </restriction>
  </simpleType>
</ConvertTemp>
```

ภาพประกอบที่ 5.1 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ ConvertTemp

(2) สร้างแบบจำลองเชิงไวยากรณ์จากอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ ConvertTemp แสดงดังภาพประกอบที่ 5.2 และมีตารางเงื่อนไขจำกัดแสดงดังตารางที่ 5.3

<ConvertTemp> = <complexType>
<complexType> = <sequence>
<sequence> = <Temperature><FromUnit><ToUnit>
<Temperature> = double
<FromUnit> = [string]
<ToUnit> = [string]

ภาพประกอบที่ 5.2 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp

ตารางที่ 5.3 เงื่อนไขจำกัดของการดำเนินการ ConvertTemp

ชื่อ	เงื่อนไขจำกัด
string	enumeration = degreeCelsius, enumeration = degreeFahrenheit, enumeration = degreeRankine, enumeration = degreeReaumur, enumeration = kelvin

(3) นำแบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp มาทำการปรับปรุง จะได้แบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วแสดงดังภาพประกอบที่ 5.3

<ConvertTemp> = <Temperature><FromUnit><ToUnit>
<Temperature> = double
<FromUnit> = [string]
<ToUnit> = [string]

ภาพประกอบที่ 5.3 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp ที่ปรับปรุงแล้ว

(4) สร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp ที่ปรับปรุงแล้ว แสดงดังภาพประกอบที่ 5.4

<ConvertTemp> ---> <Temperature><FromUnit><ToUnit>
---> double <FromUnit><ToUnit>
---> double [string] <ToUnit>
---> double [string] [string]

ภาพประกอบที่ 5.4 การสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ ConvertTemp ที่ปรับปรุงแล้ว

(5) กำหนดค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของแต่ละกรณีทดสอบของการดำเนินการ ConvertTemp แสดงดังตารางที่ 5.4

ตารางที่ 5.4 ค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของการดำเนินการ ConvertTemp

อิลิเมนต์	ค่าที่ถูกต้องตามสัญลักษณ์สิ้นสุด	ค่าที่ไม่ถูกต้องตามสัญลักษณ์สิ้นสุด
<Temperature>	0, 1.7976931348623157E308, -1.7976931348623157E308	a3d5g7j
<FromUnit>	degreeCelsius, degreeFahrenheit, degreeRankine, degreeReaumur, kelvin	-
<ToUnit>	degreeCelsius, degreeFahrenheit, degreeRankine, degreeReaumur, kelvin	-

ดังนั้นจำนวนกรณีทดสอบทั้งหมดของการดำเนินการ ConvertTemp ของเว็บเซอร์วิส ConvertTemperature จะมีค่าเท่ากับ

จำนวนกรณีทดสอบ = จำนวนค่าที่ถูกต้องและไม่ถูกต้องของ <Temperature>
X จำนวนค่าที่ถูกต้องและไม่ถูกต้องของ <FromUnit>
x จำนวนค่าที่ถูกต้องและไม่ถูกต้องของ <ToUnit>
= 4x5x5
= 100 กรณี

2) เว็บเซอร์วิส MathService

ขั้นตอนในการสร้างกรณีทดสอบของการดำเนินการ Divide ของเว็บเซอร์วิส MathService ตามวิธีการที่นำเสนอมีรายละเอียดในแต่ละขั้นตอนดังนี้

(1) จากการวิเคราะห์เอกสาร WSDL ของเว็บเซอร์วิส MathService สามารถสกัดอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Divide ซึ่งมีพารามิเตอร์ 2 ตัว คือ A และ B แสดงดังภาพประกอบที่ 5.5

```
<Divide>
  <element name="Divide">
    <complexType>
      <sequence>
        <element minOccurs="1" maxOccurs="1" name="A" type="float" />
        <element minOccurs="1" maxOccurs="1" name="B" type="float" />
      </sequence>
    </complexType>
  </element>
</Divide>
```

ภาพประกอบที่ 5.5 อินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Divide

(2) สร้างแบบจำลองเชิงไวยากรณ์จากอินเตอร์มีเดียทโอเปอร์เรชั่นของการดำเนินการ Divide แสดงดังภาพประกอบที่ 5.6

```
<Divide> = <complexType>
<complexType> = <sequence>
<sequence> = <A><B>
<A> = float
<B> = float
```

ภาพประกอบที่ 5.6 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide

(3) นำแบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide มาทำการปรับปรุง จะได้แบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้วแสดงดังภาพประกอบที่ 5.7

<Divide> = <A><B>
<A> = float
<B> = float

ภาพประกอบที่ 5.7 แบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide ที่ปรับปรุงแล้ว

(4) สร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide ที่ปรับปรุงแล้วแสดงดังภาพประกอบที่ 5.8

<Divide> ---> <A><B>
---> float <B>
---> float float

ภาพประกอบที่ 5.8 การสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ของการดำเนินการ Divide ที่ปรับปรุงแล้ว

(5) กำหนดค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของแต่ละกรณีทดสอบของการดำเนินการ Divide แสดงดังตารางที่ 5.5

ตารางที่ 5.5 ค่าที่ถูกต้องและไม่ถูกต้องตามสัญลักษณ์สิ้นสุดของการดำเนินการ Divide

อิลิเมนต์	ค่าที่ถูกต้องตามสัญลักษณ์สิ้นสุด	ค่าที่ไม่ถูกต้องตามสัญลักษณ์สิ้นสุด
<A>	0, 3.4028235E38, -3.4028235E38	rsjgtyhtujrk
<B>	0, 3.4028235E38, -3.4028235E38	dfietiepuytti

ดังนั้นจำนวนกรณีทดสอบทั้งหมดของการดำเนินการ Divide ของเว็บเซอร์วิส MathService จะมีค่าเท่ากับ

จำนวนกรณีทดสอบ = จำนวนค่าที่ถูกต้องและไม่ถูกต้องของ <A>
x จำนวนค่าที่ถูกต้องและไม่ถูกต้องของ <B>
= 4x4
= 16 กรณี

5.5 การสร้างกรณีทดสอบตามวิธีการของ Samer

การสร้างกรณีทดสอบตามวิธีการของ Samer จะใช้วิธีการวิเคราะห์ค่าขอบเขตตามชนิดข้อมูลนั้น เช่น ถ้าเป็นชนิดข้อมูลที่เป็นตัวเลขจะใช้ค่าขอบดังนี้ (1) ค่าสูงสุด (2) ค่าสูงสุด+1 (3) ค่าสูงสุด-1 (4) ค่าต่ำสุด (5) ค่าต่ำสุด-1 และ (6) ค่าต่ำสุด+1 หรือหากเป็นค่าของเงื่อนไข เช่น Enumeration จะใช้ค่าที่เป็นไปได้ของเงื่อนไขดังนี้ (1) ค่าที่เป็นไปได้ของเงื่อนไขตัวที่ 1 (2) ค่าที่เป็นไปได้ของเงื่อนไขตัวที่ 2 (3) ค่าที่เป็นไปได้ของเงื่อนไขตัวที่ n-1 และ (4) ค่าที่เป็นไปได้ของเงื่อนไขตัวที่ n เป็นต้น และจำนวนกรณีทดสอบที่ได้จะเท่ากับผลคูณคาร์ทีเชียนของจำนวนค่าขอบเขตของแต่ละพารามิเตอร์ เช่น มีพารามิเตอร์ 2 ตัว ซึ่งมีจำนวนค่าขอบเขตเท่ากับ 3 และ 2 ตามลำดับ ดังนั้นจำนวนกรณีทดสอบจะเท่ากับ 3x2 = 6 กรณี เป็นต้น

การสร้างกรณีทดสอบตามวิธีการของ Samer สำหรับเว็บเซอร์วิส ConvertTemperature และเว็บเซอร์วิส MathService มีรายละเอียดดังนี้

1) เว็บเซอร์วิส ConvertTemperature

เว็บเซอร์วิส ConvertTemperature มีพารามิเตอร์ทั้งหมด 3 ตัวคือ Temperature FromUnit และ ToUnit ตามลำดับ โดยที่พารามิเตอร์ Temperature มีชนิดเป็น double ส่วนพารามิเตอร์ FromUnit และพารามิเตอร์ ToUnit เป็น Enumeration โดยทั้งสองพารามิเตอร์มีค่าที่เป็นไปได้ของเงื่อนไขทั้งหมด 5 ค่าคือ degreeCelsius degreeFahrenheit degreeRankine degreeReaumur และ kelvin ค่าขอบเขตซึ่งพิจารณาตามวิธีการของ Samer ของแต่ละพารามิเตอร์ของการดำเนินการ ConvertTemp ของเว็บเซอร์วิส ConvertTemperature แสดงดังตารางที่ 5.6

ตารางที่ 5.6 ค่าขอบเขตของแต่ละพารามิเตอร์ของการดำเนินการ ConvertTemp ตามวิธีการของ Samer

พารามิเตอร์	ชนิด	ค่าขอบเขต
Temperature	double	ค่าสูงสุด ค่าสูงสุด+1 ค่าสูงสุด-1 ค่าต่ำสุด ค่าต่ำสุด-1 และค่าต่ำสุด+1
FromUnit	Enumeration	degreeCelsius degreeFahrenheit degreeReaumur และ kelvin
ToUnit	Enumeration	degreeCelsius degreeFahrenheit degreeReaumur และ kelvin

ดังนั้นจำนวนกรณีทดสอบทั้งหมดของเว็บเซอร์วิส ConvertTemperature จะมีค่าเท่ากับ

จำนวนกรณีทดสอบ = จำนวนค่าขอบของ Temperature x จำนวนค่าขอบของ FromUnit x จำนวนค่าขอบของ ToUnit
= 6x4x4
= 96 กรณี

2) เว็บเซอร์วิส MathService

เว็บเซอร์วิส MathService มีพารามิเตอร์ทั้งหมด 2 ตัวคือ A และ B ตามลำดับ โดยพารามิเตอร์ทั้งสองมีชนิดเป็น float ค่าขอบเขตซึ่งพิจารณาตามวิธีการของ Samer ของแต่ละพารามิเตอร์ของการดำเนินการ Divide ของเว็บเซอร์วิส MathService แสดงดังตารางที่ 5.7

ตารางที่ 5.7 ค่าขอบเขตของแต่ละพารามิเตอร์ของการดำเนินการ Divide ตามวิธีการของ Samer

พารามิเตอร์	ชนิดข้อมูล	กรณีทดสอบ
A	float	ค่าสูงสุด ค่าสูงสุด+1 ค่าสูงสุด-1 ค่าต่ำสุด ค่าต่ำสุด-1 และค่าต่ำสุด+1
B	float	ค่าสูงสุด ค่าสูงสุด+1 ค่าสูงสุด-1 ค่าต่ำสุด ค่าต่ำสุด-1 และค่าต่ำสุด+1

ดังนั้นจำนวนกรณีทดสอบทั้งหมดของเว็บเซอร์วิส MathService จะมีค่าเท่ากับ

จำนวนกรณีทดสอบ = จำนวนค่าขอบของ A x จำนวนค่าขอบของ B
= 6x6
= 36 กรณี

5.6 ผลการทดสอบ

จากการสร้างกรณีทดสอบสำหรับทดสอบเว็บเซอร์วิส ConvertTemperature และเว็บเซอร์วิส MathService ด้วยวิธีการที่นำเสนอในงานวิจัย ซึ่งทำการสกัดเอกสาร WSDL ให้ได้อินเตอร์มีเดียทโอเปอร์เรชั่น จากนั้นสร้างและปรับปรุงแบบจำลองเชิงไวยากรณ์ สุดท้ายสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ และวิธีการของ Samer เมื่อนำกรณีทดสอบที่

สร้างจากวิธีการทั้งสองไปทดสอบกับเว็บเซอร์วิส สามารถสรุปจำนวนกรณีทดสอบทั้งหมดที่ถูกสร้างขึ้นและจำนวนกรณีทดสอบที่สามารถจับข้อผิดพลาดได้แสดงดังตารางที่ 5.8

ตารางที่ 5.8 จำนวนกรณีทดสอบทั้งหมดและจำนวนกรณีทดสอบที่สามารถจับข้อผิดพลาดของเว็บเซอร์วิสด้วยกรณีทดสอบจากวิธีที่นำเสนอและวิธีของ Samer

เว็บเซอร์วิส	วิธีการที่ใช้	จำนวนกรณีทดสอบทั้งหมด (กรณี)	จำนวนกรณีทดสอบที่สามารถจับข้อผิดพลาดได้	
			จำนวน (กรณี)	เปอร์เซ็นต์ (%)
ConvertTemprature	Samer	96	76	79.17
	งานวิจัย	100	90	90
MathService	Samer	36	24	66.67
	งานวิจัย	16	11	68.75

จากตารางที่ 5.8 เว็บเซอร์วิส ConvertTemprature สามารถสร้างกรณีทดสอบตามวิธีการของ Samer ได้ทั้งหมด 96 กรณี สามารถจับข้อผิดพลาดได้ 76 กรณี คิดเป็น 79.17 เปอร์เซ็นต์ และสร้างกรณีทดสอบตามวิธีการที่นำเสนอได้กรณีทดสอบทั้งหมด 100 กรณี สามารถจับข้อผิดพลาดได้ 90 กรณี คิดเป็น 90 เปอร์เซ็นต์ ส่วนเว็บเซอร์วิส MathService สามารถสร้างกรณีทดสอบตามวิธีการของ Samer ได้ทั้งหมด 36 กรณี สามารถจับข้อผิดพลาดได้ 24 กรณี คิดเป็น 66.67 เปอร์เซ็นต์ และสร้างกรณีทดสอบตามวิธีการที่นำเสนอได้กรณีทดสอบทั้งหมด 16 กรณี สามารถจับข้อผิดพลาดได้ 11 กรณี คิดเป็น 68.75 เปอร์เซ็นต์ จากผลการทดลองจะเห็นว่ากรณีทดสอบที่สร้างตามวิธีการที่นำเสนอสามารถจับการทำงานที่ผิดพลาดของเว็บเซอร์วิสทั้งสองได้มากกว่าวิธีการของ Samer

เมื่อพิจารณาจำนวนกรณีทดสอบทั้งหมดของวิธีการที่นำเสนอพบว่าจำนวนกรณีทดสอบที่ได้จะมีทั้งที่มากกว่าและน้อยกว่าวิธีการของ Samer เมื่อพิจารณาจะพบว่าเว็บเซอร์วิส ConvertTemprature มีจำนวนกรณีทดสอบที่สร้างตามวิธีการที่นำเสนอมากกว่าจำนวนกรณีทดสอบที่สร้างตามวิธีการของ Samer ทั้งนี้เนื่องจากสัญลักษณ์สิ้นสุดบางตัวมีเงื่อนไขจำกัดในตารางเป็นแบบเงื่อนไขจำกัดกลุ่มของค่า ทำให้ต้องใช้ค่าเงื่อนไขจำกัดทุกค่าที่อยู่ในตารางเงื่อนไขจำกัดมาใช้ในการทดสอบจึงทำให้มีจำนวนกรณีทดสอบเพิ่มมากขึ้น แต่จากการทดลองพบว่ากรณีทดสอบที่สร้างตามวิธีการที่นำเสนอมีจำนวนเพิ่มขึ้น 4 กรณีทดสอบ คิดเป็น 4.17 เปอร์เซ็นต์ แต่จากการเพิ่มกรณีทดสอบนี้ทำให้สามารถจับข้อผิดพลาดของเว็บเซอร์วิสได้เพิ่มขึ้น 14 กรณี ซึ่งคิดเป็น 18.42 เปอร์เซ็นต์

บทที่ 6

บทสรุปและข้อเสนอแนะ

จากการนำเสนอแนวคิดในการสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิสจากเอกสาร WSDL และการพัฒนาเครื่องมือซึ่งสนับสนุนแนวคิดที่นำเสนอ สามารถสรุปผล ปัญหาและอุปสรรคในการวิจัย รวมไปถึงข้อเสนอแนะและงานในอนาคตได้ดังนี้

6.1 บทสรุป

วิทยานิพนธ์นี้นำเสนอวิธีการสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส ขั้นตอนการสร้างกรณีทดสอบเริ่มจากการวิเคราะห์เอกสาร WSDL โดยพิจารณาจากความสัมพันธ์ของอิลิเมนต์ <types> อิลิเมนต์ <message> และอิลิเมนต์ <portType> เพื่อสกัดรูปแบบการดำเนินการต่าง ๆ ของเว็บเซอร์วิสให้อยู่ในรูปของอินเตอร์-มีเดียทโอเปอร์เรชั่น จากนั้นสร้างและปรับปรุงแบบจำลองเชิงไวยากรณ์จากอินเตอร์มีเดียทโอ-เปอร์เรชั่น สุดท้ายสร้างกรณีทดสอบจากแบบจำลองเชิงไวยากรณ์ที่ปรับปรุงแล้ว

นอกจากนี้วิทยานิพนธ์นี้ยังได้พัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบเพื่อสนับสนุนแนวคิดดังกล่าว และใช้เครื่องมือดังกล่าวทดลองสร้างกรณีทดสอบสำหรับเว็บเซอร์วิส 2 เว็บเซอร์วิส ได้แก่ เว็บเซอร์วิส ConvertTemperature ที่ให้บริการแปลงหน่วยของอุณหภูมิ และเว็บเซอร์วิส MathService ที่ให้บริการคำนวณทางคณิตศาสตร์ ผลการทดลองแสดงให้เห็นว่าเครื่องมือสามารถสร้างแบบจำลอง ปรับปรุงแบบจำลอง และสร้างกรณีทดสอบได้ถูกต้องตามแนวคิดที่นำเสนอ และกรณีทดสอบสร้างขึ้นสามารถจับการทำงานที่ผิดพลาดของเว็บเซอร์วิสได้

6.2 ปัญหาและอุปสรรค

6.2.1 เนื่องจากผู้วิจัยไม่มีความรู้ในเรื่องของ XML Parser ประเภท DOM ทำให้ต้องใช้เวลาในการศึกษาค่อนข้างนาน

6.2.2 เนื่องจากเว็บเซอร์วิสที่มีอยู่บนอินเตอร์เน็ตมีจำนวนมากทำให้ต้องใช้เวลาในการรวบรวมลักษณะส่วนใหญ่ของอิลิเมนต์ในเอกสาร WSDL ของเว็บเซอร์วิสที่จะนำมาใช้ในการพัฒนาเครื่องมือสำหรับสร้างกรณีทดสอบค่อนข้างนาน

6.3 ข้อเสนอแนะและงานในอนาคต

ข้อเสนอแนะและงานในอนาคตสำหรับงานวิจัยมีดังนี้

6.3.1 งานวิจัยนี้ครอบคลุมเอกสาร WSDL ซึ่งอธิบายรูปแบบข้อมูลนำเข้าของการดำเนินการด้วยอิลิเมนต์แบบชนิดข้อมูลแบบซับซ้อน ดังนั้นในอนาคตควรปรับปรุงวิธีการให้ครอบคลุมเอกสาร WSDL ที่มีความหลากหลายมากขึ้น

6.3.2 เครื่องมือที่สร้างขึ้นสนับสนุนการสร้างกรณีทดสอบโดยอัตโนมัติ แต่ไม่สนับสนุนการนำกรณีทดสอบที่สร้างขึ้นไปทดสอบกับเว็บเซอร์วิสโดยอัตโนมัติ ดังนั้นในอนาคตควรพัฒนาเครื่องมือให้สามารถทำงานได้โดยอัตโนมัติ

6.3.3 การสร้างข้อมูลทดสอบในส่วนที่เป็นการกำหนดค่าที่ไม่ถูกต้องจะกำหนดเฉพาะชนิดข้อมูลที่ไม่มีเงื่อนไขจำกัดเท่านั้น ดังนั้นในอนาคตมีการกำหนดค่าที่ไม่ถูกต้องให้กับชนิดข้อมูลที่มีเงื่อนไขจำกัดด้วย

6.3.4 การทดสอบเปรียบเทียบประสิทธิภาพของวิธีการที่นำเสนอ ควรเพิ่มโดยใช้วิธีการทดลองเชิงประจักษ์เพื่อหาข้อสรุปได้ดียิ่งขึ้น

บรรณานุกรม

สุธี พงศาสกุลชัย. 2550. การพัฒนาระบบด้วยสถาปัตยกรรมเชิงบริการบนเทคโนโลยีของ Web Service. พิมพ์ครั้งที่ 1. เคทีพี คอมพ์ แอนด์ คอนซอลท์: กรุงเทพฯ.

Chunyan Ma, Chenglie Du, Tao Zhang, Fei Hu and Xiaobin Cai. 2008. WSDL-Based Automated Test Data Generation for Web Service. 2008 International Conference on Computer Science and Software Engineering.December 2008. Vol. 2. pp.731-737.

ConvertTemp. ConvertTemperature. http://www.webservicex.net/ConvertTemperature.asmx/ (accessed 17/11/2010)

CORBA. Common Object Request Broker Architecture (CORBA/IIOP). http://www.omg.org/teghnology/documents/corba_spec_catalog.htm/ (accessed 17/8/2010)

Dan Vint. 2003. XML Schema - Structures Quick Reference Card. http://www.xml.dvint.com/docs/SchemaDataTypesQR-2.pdf/ (accessed 08/08/2010)

DCOM. Distributed Component Object Model (DCOM). http://msdn.microsoft.com/th-th/library/cc201989(en-us,PROT.10).aspx/ (accessed 17/8/2010)

Divide. MathService. http://aspalliance.com/quickstart/aspplus/samples/services/MathService/VB/MathService.asmx/ (accessed 17/11/2010)

Douglas K. Barry. 2003. Web Services and Service-Oriented Architectures: The Savvy Manager's Guide. Morgan Kaufmann.

Ethan Cerami. 2002. Web services essentials: Distributed applications with XML-RPC, SOAP, UDDI & WSDL. 1st edition. O'Reilly.

Mustafa Bozkurt, Mark Harman and Youssef Hassoun. 2010. Testing Web Services: A Survey. Technical report TR-10-01. Department of Computer Science King's College London.

Patrick Carey. 2007. New Perspectives on XML. 2nd edition. Thomson Course Technology.

Paul Ammann and Jeff Offutt. 2008. Introduction to Software Testing. Cambridge University Press.

RMI. Remote Method Invocation (RMI). http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp/ (accessed 17/8/2010)

Samer Hanna and Malcolm Munro. 2007. An Approach for Specification-based Test Case Generation for Web Services. IEEE/ACS International Conference on Computer Systems and Applications. May 2007. pp. 16-23.

Sebesta, Robert W. 2005. Concepts of Programming Languages. 7th edition. Pearson Addison Wesley.

W3C. Extensible Markup Language (XML). 2008. http://www.w3.org/TR/REC-xml/ (accessed: 9/7/2010).

W3C. Simple Object Access Protocol (SOAP). 2007. http://www.w3.org/TR/soap12-part1/ (accessed 19/6/2010).

W3C. Web Services Description Language (WSDL). 2001. http://www.w3.org/TR/wsdl/ (accessed 19/6/2010).

W3schools. XML Schema. http://www.w3schools.com/schema/default.asp/ (accessed 11/1/2011).

Xiaoying Bai, Wenli Dong, Wei-Tek Tsai and Yinong Chen. 2005. WSDL-Based Automatic Test Case Generation for Web Services Testing. Proceedings of the 2005 IEEE International Workshop on Service-Oriented System Engineering (SOSE'05). IEEE Computer Society Press. pp. 215-220.

ภาคผนวก

ภาคผนวก ก.

ผลงานตีพิมพ์

เรื่อง	การสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิส
งานประชุมวิชาการ	The 7th International Joint Conference on Computer Science and Software Engineering (JCSSE 2010)
สถานที่	กรุงเทพมหานคร ประเทศไทย
วันที่	12 – 14 พฤษภาคม 2553

ประวัติผู้เขียน

ชื่อ สกุล นางสาวจุฑาพร อินทะสะระ

รหัสประจำตัวนักศึกษา 5110220011

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วท.บ. (วิทยาการคอมพิวเตอร์)	มหาวิทยาลัยสงขลานครินทร์	2550

การตีพิมพ์เผยแพร่ผลงาน

จุฑาพร อินทะสะระ และสุภาภรณ์ กานต์สมเกียรติ. การสร้างกรณีทดสอบเชิงไวยากรณ์สำหรับการทดสอบเว็บเซอร์วิส. The 7th International Joint Conference on Computer Science and Software Engineering (JCSSE 2010). กรุงเทพมหานคร ประเทศไทย. 12-14 พฤษภาคม 2553

จุฑาพร อินทะสะระ และสุภาภรณ์ กานต์สมเกียรติ. แบบจำลองเชิงไวยากรณ์สำหรับทดสอบเว็บเซอร์วิส. การประชุมทางวิชาการระดับชาติด้านคอมพิวเตอร์และเทคโนโลยีสารสนเทศ (NCCIT 2010) ครั้งที่ 6. กรุงเทพมหานคร ประเทศไทย. 3-5 มิถุนายน 2553