กรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส

นายพิเชษฐ์ อำนวยกาญจนสิน

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

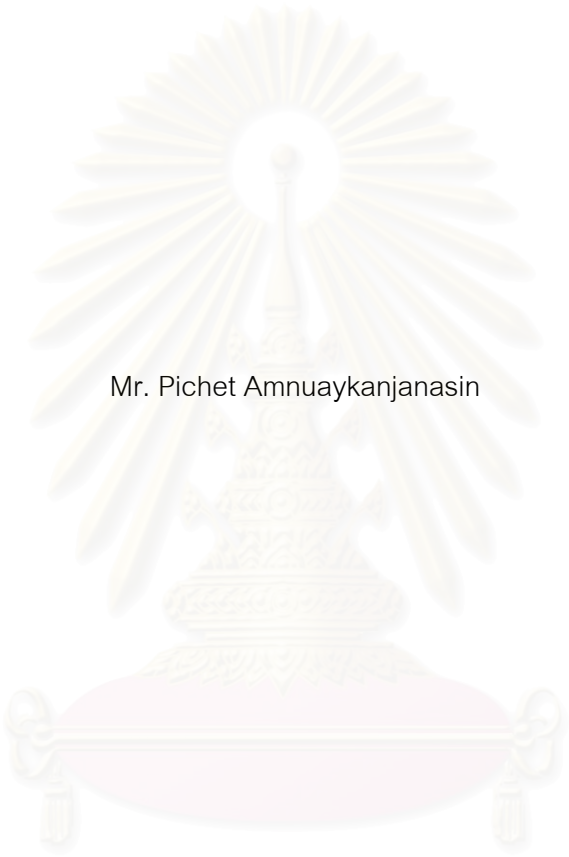
วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2547

ISBN 974-17-7000-6

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

BPEL-ORCHESTRATED FRAMEWORK FOR GRID SERVICES

Mr. Pichet Amnuaykanjanasin



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Computer Engineering

Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University

Academic Year 2004

ISBN 974-17-7000-6

หัวข้อวิทยานิพนธ์	กรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส
โดย	นายพิเชษฐ์ อำนวยกาญจนสิน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา	อาจารย์ ดร.ณัฐวุฒิ หนูไพโรจน์

คณะวิศวกรรมศาตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

……..คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.ดิเรก ลาวัณย์ศิริ)

คณะกรรมการสอบวิทยานิพนธ์

……..ประธานกรรมการ
(อาจารย์ ดร.ยรรยง เต็งอำนวย)

……..อาจารย์ที่ปรึกษา
(อาจารย์ ดร.ณัฐวุฒิ หนูไพโรจน์)

……..กรรมการ
(อาจารย์ ดร.วีระ เหมืองสิน)

……..กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.ภุชงค์ อุทโยภาศ)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

พิเชษฐ์ อำนวยกาญจนสิน : กรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส. (BPEL-ORCHESTRATED FRAMEWORK FOR GRID SERVICES) อ. ที่ปรึกษา: อาจารย์ ดร.ณัฐวุฒิ หนูไพโรจน์, จำนวนหน้า 97 หน้า. ISBN 974-17-7000-6.

ระบบจัดการกระแสงานสำหรับระบบกริดในปัจจุบันนั้นล้วนแต่ถูกสร้างบนพื้นฐานเทคโนโลยีและระบบที่ต่างกันไป ทำให้ระบบเหล่านี้ไม่สามารถทำงานร่วมกัน รวมไปถึงเครื่องมือต่างๆที่ใช้ในการจัดการก็มีอยู่อย่างจำกัด นอกจากนั้นยังไม่มีข้อกำหนดการจัดการกระแสงานที่เป็นมาตรฐานในระบบกริดเกิดขึ้นเลย เพื่อแก้ปัญหาดังกล่าวจึงได้เกิดแนวคิดในการนำมาตรฐานการจัดการกระแสงานรูปแบบหนึ่งของระบบกระจายที่ได้รับการยอมรับกันอย่างกว้างขวางในปัจจุบัน คือ บีเพล มาใช้ในการจัดการกระแสงานในระบบกริด โดยจะนำมาตรฐานบีเพลมาใช้ประสานการทำงานร่วมกันระหว่างกริดเซอร์วิสต่างๆ ที่สร้างขึ้นภายใต้ข้อกำหนดโอจีเอสไอ ซึ่งเป็นข้อกำหนดสำหรับสถาปัตยกรรมเชิงเซอร์วิสสำหรับระบบกริด แต่การนำบีเพลมาใช้กับกริดเซอร์วิสนั้นไม่สามารถทำได้โดยตรง เนื่องมาจากบีเพลไม่สามารถเรียกใช้งานกลไกและคุณสมบัติของกริดเซอร์วิสที่กำหนดขึ้นตามข้อกำหนดโอจีเอสไอ รวมไปถึงการใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยตามข้อกำหนดจีเอสไอ

วิทยานิพนธ์นี้นำเสนอกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส ซึ่งทำหน้าที่เป็นชั้นประสานการทำงานระหว่างกริดเซอร์วิสกับบีเพล โดยนำเสนอส่วนประกอบสำคัญ 3 ส่วน ได้แก่ เว็บเซอร์วิสผู้ช่วย เครื่องมือปรับปรุงดับบลิวเอสดีแอล และตัวแทนความปลอดภัย การใช้งานส่วนประกอบทั้ง 3 จะทำให้เราสามารถใช้บีเพลประสานการทำงานและเรียกใช้งานกลไกและคุณสมบัติของกริดเซอร์วิสทั้งที่มีและไม่มีการรักษาความปลอดภัยได้เป็นอย่างดี โดยไม่ต้องมีการแก้ไขเพิ่มเติมข้อกำหนดบีเพลใดๆทั้งสิ้น วิทยานิพนธ์นี้ยังได้นำกรอบการประสานการทำงานที่สร้างขึ้น ไปใช้ในการสร้างกระบวนการบีเพลเพื่อเรียกใช้แกรมกริดเซอร์วิส ซึ่งเป็นกริดเซอร์วิสที่สร้างขึ้นโดยโกลบัสทูลคิทสำหรับการจัดการทรัพยากรกริด ซึ่งทำให้ผู้ใช้สามารถเข้าใช้งานทรัพยากรกริดต่างๆผ่านทางกระบวนการบีเพลได้โดยตรง เป็นการเพิ่มทางเลือกให้แก่ผู้ที่ต้องการใช้งานทรัพยากรกริดได้อีกทางหนึ่ง

ภาควิชาวิศวกรรมคอมพิวเตอร์ .....ลายมือชื่อนิสิต.....
สาขาวิชาวิศวกรรมคอมพิวเตอร์..... ลายมือชื่ออาจารย์ที่ปรึกษา.....
ปีการศึกษา 2547

4670413721 : MAJOR COMPUTER ENGINEERING

KEY WORD: BPEL / GRID SERVICES / ORCHESTRATION / FRAMEWORK / WORKFLOW

PICHET AMNUAYKANJANASIN : BPEL-ORCHESTRATED FRAMEWORK FOR GRID SERVICES. THESIS ADVISOR: NATAWUT NUPAIROJ, Ph.D., 97 pp. ISBN 974-17-7000-6.

Nowadays, the workflow management systems of Grid systems are developed and implemented based on proprietary technologies and systems. These systems are not interoperable and have limited management tools. Furthermore, there is no standard workflow specification of Grid systems. To resolve these problems, this thesis proposes the use of BPEL specification to manage Grid workflows. BPEL is currently a de-facto workflow standard in distributed systems. This thesis use BPEL to orchestrate OGSI-based Grid services. In facts, the use of BPEL orchestrating Grid services has many problems. BPEL processes are not able to use OGSI-based Grid services' mechanisms and features. They also can not orchestrate secure Grid services.

This thesis proposes BPEL-Orchestrated Framework for Grid Services. Our framework act as convergence layer between Grid services and BPEL processes. This framework proposes three important components; helper Web services, WSDL modification tools and security proxy. By using these components, we can use BPEL process to orchestrate and use all features and mechanisms of both non-secure and secure Grid services without modifying standard BPEL specification. For testing the framework, this thesis developed BPEL processes to orchestrate secured sample Grid services and Globus's GRAM Grid services. Using GRAM Grid service via BPEL process encourages the use of Grid systems for non-scientific domain as well as simplifies the development of Grid applications.

Department of Computer Engineering Student's signature.....

Field of study Computer Engineering Advisor's signature…….....

Academic year 2004

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยความกรุณาและความช่วยเหลืออย่างดียิ่งจาก อ.ดร.ณัฐวุฒิ หนูไพโรจน์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้สละเวลาในการให้คำปรึกษา คำแนะนำอันมีค่า ตลอดจนช่วยตรวจทานแก้ไขวิทยานิพนธ์และบทความตีพิมพ์ต่างๆ จนวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี

ขอขอบคุณ อ.ดร.ยรรยง เต็งอำนวย อ.ดร.วีระ เหมืองสิน และ ผศ.ดร.ภุชงค์ อุทโยภาศ ซึ่งเป็นกรรมการสอบวิทยานิพนธ์นี้ รวมไปถึง ผศ.ดร.ทวิตีย์ เสนีย์วงศ์ ณ อยุธยา ที่ช่วยทำให้ห้องปฏิบัติการวิจัยระบบสารสนเทศ เป็นห้องวิจัยที่เพียบพร้อมที่สุด

ขอขอบคุณพี่และเพื่อนกลุ่มวิจัยกริดของอ.ดร.ณัฐวุฒิ หนูไพโรจน์ ได้แก่ นายณัฐกฤต สงวนดีกุล นายปฏิภาณ คงศิริกุล และนายธีรยุทธ หิรัญทราภรณ์ ที่ช่วยกันศึกษาหาความรู้ในแขนงงานวิจัยด้านนี้และให้คำปรึกษาในทุกๆเรื่องมาโดยตลอด

ขอขอบคุณพี่ๆ เพื่อนๆ ในห้องปฏิบัติการวิจัยระบบสารสนเทศทุกคน ที่ช่วยมอบบรรยากาศอันดีและแสนอบอุ่น รวมไปถึงเสบียง ในการทำงานวิจัย

ขอขอบคุณ อ.ดร.วิษณุ โคตรจรัส และ อ.เชษฐ์ พัฒโนทัย และน้องๆ หลักสูตรพัฒนาซอฟต์แวร์ทุกคน ที่ช่วยให้การเรียนในระดับปริญญาโทของข้าพเจ้า ไม่เงียบเหงาและน่าเบื่อ

ขอขอบคุณเพื่อนๆ พี่ๆ ระดับบัณฑิตศึกษา ภาควิชาวิศวกรรมคอมพิวเตอร์ทุกท่าน สำหรับความสนุกสนาน ครื้นเครง ตลอด 2 ปีสำหรับสถานะนิสิตปริญญาโท

สุดท้ายนี้ ขอกราบขอขอบพระคุณน้าพล น้าแต๋ว ที่ช่วยเลี้ยงดู ให้การสนับสนุนในทุกๆด้าน จนข้าพเจ้าสำเร็จการศึกษาวิศวกรรมศาสตรมหาบัณฑิตในวันนี้ และขอขอบคุณแก็ป น้องชายสุดที่รักของข้าพเจ้า สำหรับความเป็นน้องและเพื่อนที่ดีที่สุดของข้าพเจ้า

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง
บทคัดย่อภาษาอังกฤษ ..... จ
กิตติกรรมประกาศ ..... ฉ
สารบัญ ..... ช
สารบัญตาราง ..... ฎ
สารบัญรูปภาพ ..... ฎ
บทที่ 1 บทนำ 1
1.1 ความเป็นมาและความสำคัญของปัญหา 1
1.2 วัตถุประสงค์ของการวิจัย 3
1.3 ขอบเขตของการวิจัย 3
1.4 ประโยชน์ที่คาดว่าจะได้รับ 4
1.5 วิธีดำเนินการวิจัย 4
1.6 ผลงานตีพิมพ์ 5
บทที่ 2 เอกสารและงานวิจัยที่เกี่ยวข้อง 6
2.1 แนวคิดและทฤษฎี 6
2.1.1 กริด 6
2.1.2 กริดเซอร์วิส 7
2.1.3 โครงสร้างพื้นฐานความปลอดภัยกริด (Grid Security Infrastructure – GSI) 9
2.1.4 ระบบจัดการกระแสงาน (Workflow Management System) 10
2.1.4.1 ข้อกำหนดกระแสงาน (Workflow Specification) 10
2.1.4.2 ระบบทำงานกระแสงาน (Workflow Engine) 11
2.1.4.3 ตัวสร้างกระแสงาน (Workflow Editor) 11
2.1.4.4 เครื่องมือตรวจสอบกระแสงาน (Workflow Monitoring Tools) 11
2.1.5 บีเพล (BPEL) 12
2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง 13
2.2.1 Orchestration of an OGSI-enabled scientific application using the Business Process Execution Language 13

สารบัญ(ต่อ)

2.2.2 On Using BPEL Extensibility to Implement OGSI and WSRF Grid Workflows 14
บทที่ 3 การออกแบบและการพัฒนากรอบการประสานการทำงานร่วมกันของบีเพล สำหรับกริดเซอร์วิส 16
3.1 ชั้นประสานกริดเซอร์วิส (Grid Services Convergence Layer) 18
3.2 ปัญหาที่พบในการใช้บีเพลประสานการทำงานระหว่างกริดเซอร์วิส 19
3.2.1 ปัญหาในการเรียกใช้คุณสมบัติและกลไกต่างๆของกริดเซอร์วิส 19
3.2.2 ปัญหาในการใช้งานกลไกความปลอดภัยของกริดเซอร์วิส 21
3.3 ส่วนประกอบต่างๆของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส 22
3.3.1 เว็บเซอร์วิสผู้ช่วย (Helper Web Services) 23
3.3.1.1 โอเปอร์เรชั่น createGridServiceInstance 24
3.3.1.2 โอเปอร์เรชั่น subscribeToHelper 25
3.3.2 เครื่องมือปรับปรุงดับบลิวเอสดีแอล (WSDL Modification Tools) 27
3.3.2.1 เครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ (Automatic GWSDL Generator) 28
3.3.2.2 เครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอล (GWSDL-to-WSDL Transformation Tool) 29
3.3.2.3 เครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานกับบีเพล (WSDL-for-BPEL modification Tool) 31
3.3.3 ตัวแทนความปลอดภัย (Security Proxy) 33
3.3.3.1 ไฟล์โครงแบบความปลอดภัย (Security Configuration File) 38
3.3.3.2 ข้อจำกัดของตัวแทนความปลอดภัยที่สร้างจากตัวแทนความปลอดภัย 42
3.3.3.3 การติดตั้งตัวแทนความปลอดภัยที่สร้างจากตัวสร้างตัวแทนอัตโนมัติ 43
3.4 แนวทางในการเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสผ่านกรอบการทำงานที่สร้างขึ้น 45

สารบัญ(ต่อ)

3.4.1 การเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้กลไกของกริดเซอร์วิสโดยทั่วไป 45
3.4.2 การเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้กลไกการแจ้งข่าว 46
บทที่ 4 ขั้นตอนการพัฒนากระบวนการบีเพลในการเรียกใช้งานกริดเซอร์วิสและตัวอย่าง 48
4.1 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานเว็บเซอร์วิส 48
4.2 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย 48
4.3 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย 50
4.4 ตัวอย่างการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย 51
4.4.1 กริดเซอร์วิสที่มีการรักษาความปลอดภัยตัวอย่าง 51
4.4.2 การสร้างตัวแทนความปลอดภัยของกริดเซอร์วิส 53
4.4.3 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้กริดเซอร์วิสตัวอย่าง 55
บทที่ 5 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรม 63
5.1 การจัดการทรัพยากรในโกลบัสทูลคิท เวอร์ชั่น 3 63
5.2 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิส (GRAM Grid Services) 65
5.2.1 การสร้างตัวแทนความปลอดภัยสำหรับแกรมกริดเซอร์วิส 66
5.2.2 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิส 69
บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ 77
6.1 สรุปผลการวิจัย 77
6.2 อภิปรายผลการวิจัย 78
6.3 ข้อเสนอแนะ 79
รายการอ้างอิง 80
ภาคผนวก 83
ภาคผนวก ก 84
ภาคผนวก ข 86

สารบัญ(ต่อ)

ภาคผนวก ค 91
ภาคผนวก ง 94
ประวัติผู้เขียนวิทยานิพนธ์ 97

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญตาราง

ตาราง หน้า

ตารางที่ 1 แสดงคุณสมบัติและกลไกต่างๆของกริดเซอร์วิสที่สามารถเรียกใช้งานได้โดยตรงจากบีเพล 20

ตารางที่ 2 อีเลเมนต์ต่างๆของไฟล์โครงแบบความปลอดภัย 41

ตารางที่ 3 ค่าโดยปริยายที่ตัวสร้างตัวแทนอัตโนมัติจะกำหนดให้กับตัวแทนความปลอดภัย 42

ตารางที่ 4 แสดงการเรียกใช้งานกลไกและคุณสมบัติของกริดเซอร์วิสผ่านกรอบการทำงานที่สร้างขึ้น 44

ตารางที่ 5 แสดงความสามารถและปัญหาที่ส่วนประกอบต่างๆของกรอบการทำงานสามารถแก้ไขได้ 47

ตารางที่ 6 ปัญหาที่ส่วนประกอบต่างๆของกรอบการประสานการทำงานร่วมกันสามารถแก้ไขได้ 78

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

สารบัญรูปภาพ

ภาพประกอบ หน้า

รูปที่ 1 ส่วนประกอบของระบบจัดการกระแสงาน 12

รูปที่ 2 ข้อกำหนดและมาตรฐานต่างๆที่เกี่ยวข้องกับกับระบบกระแสงานบนสถาปัตยกรรมเว็บเซอร์วิส 16

รูปที่ 3 ข้อกำหนดและมาตรฐานต่างๆที่เกี่ยวข้องกับกับระบบกระแสงานบนสถาปัตยกรรมกริดเซอร์วิส 17

รูปที่ 4 แบบจำลองการทำงานร่วมกันระหว่างบีเพลกับกริดเซอร์วิส 18

รูปที่ 5 ภาพรวมของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส 23

รูปที่ 6 การทำงานของโอเปอร์เรชั่น createGridServiceInstance ของเว็บเซอร์วิสผู้ช่วย 24

รูปที่ 7 การทำงานของโอเปอร์เรชั่น subscribeToHelper 25

รูปที่ 8 ซีเควนซ์ไดอะแกรมของการเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยจากกระบวนการบีเพล 26

รูปที่ 9 การใช้งานเว็บเซอร์วิสผู้ช่วยร่วมกับการเรียกใช้งานกริดเซอร์วิสโดยตรงจากกระบวนการบีเพล 27

รูปที่ 10 การทำงานของเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ 28

รูปที่ 11 การทำงานของเครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอล 30

รูปที่ 12 การทำงานของเครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานกับบีเพล 32

รูปที่ 13 การใช้งานตัวแทนความปลอดภัยในการเรียกใช้กริดเซอร์วิสจากกระบวนการบีเพล 34

รูปที่ 14 ภาพรวมการใช้งานตัวแทนความปลอดภัยโดยใช้ตัวสร้างตัวแทนอัตโนมัติ 35

รูปที่ 15 ซีเควนซ์ไดอะแกรมของการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยจากกระบวนการบีเพล 37

รูปที่ 16 เปรียบเทียบการเรียกใช้งานกริดเซอร์วิสทั้งสองแบบผ่านกรอบการทำงานที่นำเสนอขึ้น 45

รูปที่ 17 ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย 49

รูปที่ 18 ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอ 51

รูปที่ 19 ไฟล์โครงแบบความปลอดภัยสำหรับกริดเซอร์วิส TransientAutodockService 53

สารบัญรูปภาพ(ต่อ)

รูปที่ 20 สิ่งที่เพิ่มเติมลงในไฟล์โครงแบบความปลอดภัยสำหรับใช้ในการสร้างตัวแทนความปลอดภัย 54
รูปที่ 21 ไฟล์โครงแบบความปลอดภัยสำหรับโรงงานเซอร์วิส 55
รูปที่ 22 โค้ดเทียมแสดงการทำงานของกระบวนบีเพลที่สร้างขึ้น 56
รูปที่ 23 สิ่งที่เพิ่มเติมในดับบลิวเอสดีแอลของตัวแทนความปลอดภัยเพื่อใช้งานกับกระบวนการบีเพล 57
รูปที่ 24 ส่วนประกาศ partnerLink ของกระบวนการบีเพลที่สร้างขึ้น 57
รูปที่ 25 ส่วนประกาศตัวแปรของกระบวนการบีเพลที่สร้างขึ้น 58
รูปที่ 26 ส่วนประกาศตัวระบุความสัมพันธ์ของกระบวนการบีเพลที่สร้างขึ้น 58
รูปที่ 27 ส่วนประกาศการทำงานของกระบวนการบีเพลที่สร้างขึ้น 60
รูปที่ 28 กริดเซอร์วิสที่เกี่ยวข้องในการให้บริการแกรม 64
รูปที่ 29 ภาพรวมการรักษาความปลอดภัยในการเรียกใช้งานแกรม 65
รูปที่ 30 ไฟล์โครงแบบความปลอดภัยของ MJS สำหรับการสร้างตัวแทนความปลอดภัย 67
รูปที่ 31 ไฟล์โครงแบบความปลอดภัยสำหรับ MMJFS 68
รูปที่ 32 ตัวอย่างโค้ดกระบวนการบีเพลการเรียกใช้โอเปอร์เรชั่น createServiceInstance สำหรับแกรมกริดเซอร์วิส 70
รูปที่ 33 โค้ดเทียมของกระบวนการบีเพลสำหรับการเรียกใช้แกรมกริดเซอร์วิสแบบที่ 1 71
รูปที่ 34 โค้ดบีเพลสำหรับการบอกรับการแจ้งข่าวจากแกรมกริดเซอร์วิส 72
รูปที่ 35 ดับบลิวเอสดีแอลของโอเปอร์เรชั่น retreiveManagedJobState 74
รูปที่ 36 โค้ดเทียมสำหรับกระบวนการบีเพลที่มีการเรียกใช้งานแกรมกริดเซอร์วิสที่สามารถทำงานได้จริง 75
รูปที่ 37 โค้ดบีเพลสำหรับการสั่งให้แกรมกริดเซอร์วิสอินสแตนท์เริ่มทำงาน การวนลูปตวจสอบสถานะงานและการทำลายแกรมกริดเซอร์วิสอินสแตนท์ 76
รูปที่ 38 สิ่งที่ต้องเพิ่มเติมลงในดับบลิวเอสดีแอลของกระบวนการบีเพล 87
รูปที่ 39 ตัวอย่างการประกาศใช้ตัวระบุความสัมพันธ์ในเอกสารกระบวนการบีเพล 88
รูปที่ 40 ตัวอย่างเอกสารกระบวนการบีเพลที่ต้องการรับข้อความการแจ้งข่าวโดยใช้อีเลเมนต์ pick 88

สารบัญรูปภาพ(ต่อ)

รูปที่ 41 ตัวอย่างเอกสารกระบวนการบีเพลที่ต้องการรอรับข้อความโดยใช้อีเลเมนต์ eventHandlers 90
รูปที่ 42 ตัวอย่างการใช้งานเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติสำหรับกริดเซอร์วิสแบบถาวร 92
รูปที่ 43 ตัวอย่างการใช้งานเครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอลและเครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับบีเพล 92
รูปที่ 44 ตัวอย่างการเรียกใช้งานตัวสร้างตัวแทนอัตโนมัติสำหรับกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริง 95

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ระบบกริด [1] (Grid System) เกิดจากความต้องการในการใช้ทรัพยากร (resource) ร่วมกันในงานทางวิทยาศาสตร์ งานต่างๆที่ทำงานผ่านระบบกริดอาจจะทำงานอยู่บนทรัพยากรของระบบที่อยู่ห่างไกลออกไป โดยนำเสนอแนวความคิดองค์กรเสมือน (Virtual Organization) ที่อนุญาตให้ผู้ใช้ (User) ที่อยู่ในองค์กรเสมือนสามารถใช้งานทรัพยากรที่กำหนดไว้ภายในองค์กรเสมือนนั้นได้ ดังนั้นงานในระบบกริดงานหนึ่ง อาจจะเกิดจากการทำงานย่อยหลายๆงานร่วมกัน งานย่อยต่างๆ เหล่านี้อาจทำงานอยู่บนระบบที่แตกต่างกันไปภายในระบบกริด เช่น งานย่อยบางงานอาจต้องการทำงานบนซูเปอร์คอมพิวเตอร์ และใช้เวลาในการทำงานเป็นเวลานาน ในขณะที่งานย่อยบางงาน อาจทำเสร็จในเวลาเพียงไม่กี่ชั่วโมงบนเครื่องคลัสเตอร์ขนาด 16 โหนด

ด้วยลักษณะงานในระบบกริดดังที่กล่าวมาแล้ว จึงนำไปสู่การนำแนวคิดการนำระบบการจัดการกระแสงาน (Workflow Management System) มาใช้กับระบบกริด การนำแนวคิดดังกล่าวมาใช้จะช่วยลดความยุ่งยากซับซ้อนในการสร้างและพัฒนาแอพพลิเคชั่นในระบบกริดเป็นอย่างมาก จากเดิมที่ผู้ใช้จะต้องจัดการการทำงานงานย่อยต่างๆ ด้วยตนเอง แต่เมื่อนำแนวคิดระบบการจัดการกระแสงานมาใช้ในระบบกริด ผู้พัฒนาแอพพลิเคชั่น เพียงแต่เขียนเอกสารกระแสงาน (Workflow Document) ตามรูปแบบที่กำหนดในข้อกำหนดกระแสงาน (Workflow Specification) แล้วจึงส่งไปให้แก่ระบบทำงานกระแสงาน (Workflow Engine) เพื่อทำงานต่างๆ ตามที่ระบุในเอกสารกระแสงานต่อไป

การนำแนวความคิดของระบบจัดการกระแสงานมาใช้ในระบบกริดมีมาพร้อมกับการเริ่มนำกริดมาใช้ในวงการวิทยาศาสตร์ ผู้พัฒนาระบบกริดและผู้พัฒนาแอพพลิเคชั่นบนระบบกริดล้วนได้นำแนวความคิดกระแสงานมาใช้กับซอฟต์แวร์พื้นฐานหรือแอพพลิเคชั่น และได้ถูกพัฒนาเรื่อยมาจนกระทั่งพัฒนาขึ้นเป็นระบบจัดการกระแสงานที่มีความสมบูรณ์ เช่น กริดแอนท์ (GridAnt) [2] คาราจัน (Karajan) [3] กริดโฟลว์ (GridFlow) [4] พีกาซัส (Pegasus) [5] และ จีเอสเอฟแอล (GSFL) [6] เป็นต้น แต่ระบบเหล่านี้ล้วนถูกพัฒนาขึ้นบนพื้นฐานเทคโนโลยีและระบบที่ต่างกันไปตามโครงการกริดที่เป็นผู้พัฒนาระบบขึ้น ทำให้ระบบจัดการกระแสงานกริดที่

พัฒนาขึ้นเหล่านี้ไม่สามารถทำงานร่วมกันได้ เอกสารกระแสงานกระแสงานที่สร้างจากระบบจัดการกระแสงานกริดอันหนึ่งไม่สามารถทำงานได้บนระบบทำงานกริดอื่น หรือข้อกำหนดกระแสงานกริดของแต่ละระบบก็มีความแตกต่างกัน งานที่เหมือนกันแต่ทำงานภายใต้ระบบกระแสงานกริดต่างกัน ก็ต้องเขียนเอกสารกระแสงานกริดในรูปแบบที่ต่างกัน

ในปัจจุบัน มีมาตรฐานกระแสงานหนึ่งที่ได้รับความนิยมและได้รับการยอมรับกันอย่างกว้างขวาง คือ บีเพล (BPEL – Business Process Execution Language) [7] มาตรฐานบีเพลถูกออกแบบมาให้เป็นมาตรฐานกระแสงานสำหรับเว็บเซอร์วิส (Web Services) ประโยชน์ที่สำคัญของบีเพล คือ ใช้ในการบรรยายและกำหนดการทำงานของเว็บเซอร์วิสต่างๆ ที่ทำงานร่วมกัน เพื่อก่อให้เกิดเป็นเว็บเซอร์วิสใหม่ขึ้นมา (Service Composition) โดยมีตัวกลาง (Agent) ทำหน้าที่ประสานการทำงานร่วมกัน (Orchestration) ของเว็บเซอร์วิสเหล่านี้ นอกจากนั้นได้มีผู้พัฒนาเครื่องมือ (Tools) ที่มีประสิทธิภาพสำหรับการใช้งานกับบีเพลเป็นจำนวนมาก จึงเกิดแนวความคิดที่จะนำมาตรฐานบีเพลมาใช้ในการจัดการกระแสงานในระบบกริด และจากการศึกษาเบื้องต้นพบว่ามีความเป็นไปได้ โดยเฉพาะการนำมาตรฐานบีเพลมาใช้ในการประสานการทำงานร่วมกันระหว่างกริดเซอร์วิส (Grid Services) [8] ที่ทำงานภายใต้ข้อกำหนดโอจีเอสไอ (OGSI Specification) ซึ่งย่อมาจากข้อกำหนดโครงสร้างพื้นฐานกริดเซอร์วิสแบบเปิด (Open Grid Services Infrastructure) [9] ข้อกำหนดโอจีเอสไอถูกพัฒนาเพิ่มเติมขึ้นจากเทคโนโลยีเว็บเซอร์วิส เพื่อให้เหมาะสมกับการทำงานในระบบกริด ดังนั้นจึงมีความเหมาะสมที่จะนำมาตรฐานบีเพลมาใช้ในการจัดการกระแสงานในระบบกริด แต่ปัญหาที่สำคัญในการนำมาตรฐานบีเพลมาใช้กับข้อกำหนดโอจีเอสไอ คือ กลไกการทำงานต่างๆ (Features) ที่ข้อกำหนดโอจีเอสไอได้กำหนดขึ้นนั้น บางกลไกไม่สามารถทำงานร่วมกับเครื่องมือเว็บเซอร์วิสทั่วไป (Generic Web Services Tools) ซึ่งในที่นี้รวมถึงเครื่องมือบีเพล (BPEL Tools) ได้โดยตรง ขณะที่บางกลไกสามารถทำงานได้แต่ต้องอาศัยความเข้าใจในกลไกการทำงานของกริดเซอร์วิสเป็นอย่างดี

ในวิทยานิพนธ์ฉบับนี้จะทำการศึกษาถึงกรอบการทำงานที่จะทำให้บีเพลสามารถประสานการทำงานร่วมกับกริดเซอร์วิสได้ โดยจะทำการออกแบบและพัฒนากรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส (BPEL-Orchestrated Framework for Grid Services) เพื่อให้บีเพลสามารถประสานการทำงานของกริดเซอร์วิส พร้อมทั้งเรียกใช้กลไกการทำงานของกริดเซอร์วิสได้อย่างมีประสิทธิภาพ กรอบการทำงาน (Framework) ที่นำเสนอนี้จะไม่มีการเปลี่ยนแปลงหรือเพิ่มเติมข้อกำหนดบีเพลใดๆ รวมถึงจะไม่มีการสร้างหรือแก้ไขระบบทำงานบีเพล (BPEL Engine) เนื่องจาก ต้องการให้ผู้พัฒนากริดแอพพลิเคชั่นหรือผู้ใช้งานกริดเซอร์วิส

สามารถใช้เครื่องมือบีเพลมาตรฐานได้โดยตรง โดยไม่ต้องใช้ระบบทำงานที่ถูกพัฒนาขึ้นมาพิเศษ ซึ่งจะทำให้เกิดความสะดวกและสามารถทำงานร่วมกับระบบบีเพลได้กว้างขวางทั้งที่มีอยู่ในปัจจุบันและในอนาคต

1.2 วัตถุประสงค์ของการวิจัย

สร้างกรอบการทำงาน (Framework) เพื่อให้กระบวนการบีเพลที่เขียนขึ้นตามข้อกำหนดบีเพลมาตรฐาน สามารถประสานการทำงาน (Orchestrate) ร่วมกับกริดเซอร์วิสต่างๆ และเรียกใช้กลไกต่างๆของกริดเซอร์วิสที่กำหนดขึ้นตามข้อกำหนดโอจีเอสไอและมีการใช้งานโครงสร้างพื้นฐานความปลอดภัยกริดได้

1.3 ขอบเขตของการวิจัย

1. สร้างกรอบการประสานการทำงานร่วมกัน ที่ผู้ใช้สามารถสร้างกระบวนการบีเพลตามข้อกำหนดบีเพลฟอร์ดับบลิวเอสเวอร์ชั่น 1.1 เพื่อประสานการทำงานระหว่างกริดเซอร์วิสต่างๆ ได้
2. กรอบการประสานการทำงานร่วมกัน จะทำให้ผู้ใช้สามารถเรียกใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสที่สร้างขึ้นตามข้อกำหนดโอจีเอสไอเวอร์ชั่น 1.0 ได้แก่
   - โรงงานเซอร์วิส
   - การอ้างอิงเซอร์วิส
   - การจัดการสถานะ
   - ข้อมูลเซอร์วิส
   - การแจ้งข่าว
3. กริดเซอร์วิสที่สามารถทำงานได้กับกรอบการประสานการทำงานร่วมกันนี้จะต้องไม่มีการเปลี่ยนแปลงจุดอ้างอิงปลายทาง (Endpoint Reference)
4. ตัวแทนความปลอดภัยจะรองรับวิธีการพิสูจน์ยืนยันตัวตน (Authentication) ของกริดเซอร์วิสใดๆ ได้ 3 รูปแบบ คือ
   - None
   - GSISecureMessage
   - GSISecureConversation

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

5. กริดเซอร์วิสที่ใช้อ้างอิงในการพัฒนากรอบการประสานการทำงานร่วมกันนี้จะเป็นกริดเซอร์วิสที่สร้างขึ้นและทำงานด้วยซอฟต์แวร์โกลบัสทูลคิท เวอร์ชั่น 3.2
6. ระบบทำงานบีเพล (BPEL Engine) และเครื่องมือสร้างเอกสารบีเพล (BPEL Editor) ที่ใช้อ้างอิงในการพัฒนากรอบการประสานการทำงานร่วมกัน คือ ออราเคิล บีเพลโพรเซสเมเนเจอร์ เวอร์ชั่น 0.95

1.4 ประโยชน์ที่คาดว่าจะได้รับ

เกิดกรอบการประสานการทำงานร่วมกันร่วมกันระหว่างบีเพลกับกริดเซอร์วิส เพื่อใช้ในการพัฒนาแอพพลิเคชั่นกริดในรูปแบบของกระบวนการบีเพล ร่วมกับเครื่องมือบีเพลทั่วไป การใช้งานในรูปแบบนี้จะทำให้การพัฒนาแอพพลิเคชั่นกริดที่มีการเรียกใช้กริดเซอร์วิสเป็นไปได้ด้วยความสะดวกรวดเร็วยิ่งขึ้น และสามารถใช้เครื่องมือบีเพลใดก็ได้ในการพัฒนาไม่จำเป็นต้องเป็นเครื่องมือบีเพลที่ได้รับการเปลี่ยนแปลงให้เข้ากับกริดเซอร์วิส

1.5 วิธีดำเนินการวิจัย

1. ศึกษาเทคโนโลยีกริดและงานวิจัยที่เกี่ยวข้องกับเบื้องต้นเกี่ยวกับกริด
2. ศึกษาเทคโนโลยีกริดเซอร์วิส และศึกษาถึงความแตกต่างระหว่างกริดเซอร์วิสและเว็บเซอร์วิสในแง่มุมต่างๆ
3. ศึกษาเทคโนโลยีเกี่ยวกับการจัดการกระแสงานของเว็บเซอร์วิสและระบบกริดที่มีอยู่ในปัจจุบัน
4. ศึกษางานวิจัยที่เกี่ยวข้องในการนำบีเพลมาใช้งานกับระบบกริด
5. ศึกษาข้อกำหนดบีเพลและข้อกำหนดโอจีเอสไอ
6. ศึกษาถึงรายละเอียดและขั้นตอนการพัฒนากริดเซอร์วิสโดยใช้ซอฟต์แวร์โกลบัสทูลคิท เวอร์ชั่น 3.2.1
7. ศึกษาวิธีการสร้างและพัฒนากระบวนการบีเพล พร้อมทั้งวิธีการใช้งานระบบทำงานบีเพล
8. เขียนระบบทดสอบความเป็นไปได้ในการเรียกใช้งานกริดเซอร์วิสผ่านทางกระบวนการบีเพล
9. ออกแบบและกำหนดรายละเอียดกรอบการประสานงาน

10. พัฒนากรอบการประสานงานที่ออกแบบโดยพัฒนาบนพื้นฐานของซอฟต์แวร์โกลบัสทูลคิทเวอร์ชั่น 3.2 และอาเปเช่แอ็กซิส
11. ทดสอบการทำงานของกรอบการทำงานกับกระบวนการบีเพลที่ทำงานบนระบบทำงานออราเคิลบีเพลโพรเซสเมเนเจอร์
12. พัฒนาตัวอย่างการนำกรอบการประสานการทำงานไปใช้งานจริง โดยใช้แอพพลิเคชั่นการทำงานจริงของระบบกริดมาเป็นตัวทดสอบ

1.6 ผลงานตีพิมพ์

- P. Amnuaykanjanasin and N. Nupairoj, "Orchestrating Grid Applications with BPEL," presented at 8th Annual Symposium on Computational Science and Engineering, Nakorn Ratchasima, Thailand, 2004.
- P. Amnuaykanjanasin and N. Nupairoj, "The BPEL Orchestrating Framework for Secured Grid Services," presented at International Conference on Information Technology Coding and Computation, Las Vegas, NV, USA, 2005.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดและทฤษฎี

2.1.1 กริด [1]

กริดเป็นระบบกระจาย (Distributed Systems) รูปแบบหนึ่ง ที่เกิดจากความต้องการรวบรวมเอาทรัพยากรต่างๆ ที่มีอยู่อย่างกระจัดกระจาย ภายในองค์กรต่างๆ ให้อยู่ร่วมกันในลักษณะขององค์กรเสมือน (Virtual Organization - VO) ทำให้ทรัพยากรเหล่านั้นสามารถถูกใช้ร่วมกันได้ จากบุคคลที่อยู่ในองค์กรเสมือนเดียวกัน หรือ องค์การเสมือนที่มีการเชื่อถือกัน (Trusted Virtual Organization) เพื่อร่วมกันทำงานให้บรรลุวัตถุประสงค์หนึ่งๆโดยไม่ขึ้นกับโครงสร้างการเชื่อมต่อจริงของแต่ละองค์กร

ระบบกริดได้กำหนดโครงสร้างพื้นฐานต่างๆ ที่จำเป็นในการใช้งานทรัพยากรร่วมกัน ไม่ว่าจะเป็น ระบบจัดสรรและจัดการทรัพยากร (Grid Resource Allocation and Management - GRAM) [10] โครงสร้างพื้นฐานทางด้านความปลอดภัย (Grid Security Infrastructure – GSI) [11] หรือ ระบบบริการข้อมูลกริด (Grid Information Services) [12] โดยหัวใจหลักของระบบกริดก็คือ กลไกความปลอดภัย (Security Mechanisms) กลไกนี้ช่วยให้การใช้ทรัพยากรร่วมกันในระบบกริดมีความสะดวกและปลอดภัย โดยอาศัยหลักการต่างๆ ได้แก่ โครงสร้างการเข้ารหัสสาธารณะ (Public-Key Infrastructure - PKI) การใช้งานใบรับรอง (Certificate) ที่ออกจากหน่วยงานออกใบรับรองที่เชื่อถือกัน (Trusted Certificate Authorities) การมอบอำนาจผ่านทางใบรับรอง (Credential Delegation) การลงทะเบียนเข้าระบบเพียงครั้งเดียว (Single Sign-on) และอื่นๆ

ระบบกริดได้ถูกนำไปประยุกต์ใช้ในงานต่างๆ มากมาย ซึ่งอาจแบ่งออกได้เป็น

- งานที่เน้นการคำนวณ (Computation-Intensive Applications)
- งานที่มีการเก็บข้อมูลจำนวนมาก (Data-Intensive Applications)
- งานที่ต้องอาศัยการทำงานร่วมกัน (Distributed-Collaboration Applications)

ในปัจจุบันได้มีโครงการจำนวนมากพัฒนาซอฟต์แวร์สำหรับระบบกริดและแอพพลิเคชั่นต่างๆบนระบบกริด แต่ซอฟต์แวร์ที่ได้รับการยอมรับและเป็นพื้นฐานสำหรับการพัฒนาระบบกริดอื่นๆเป็นจำนวนมาก คือ โกลบัสทูลคิท (Globus Toolkit) [13] ที่พัฒนาขึ้นโดยโกลบัสอัลลิแอนซ์ (Globus Alliance) โกลบัสทูลคิทได้ถูกออกแบบมาให้ตอบสนองความต้องการในระบบกริดต่างๆ เช่น ความปลอดภัย การจัดการทรัพยากร การค้นหาสารสนเทศ การจัดการข้อมูล การสื่อสาร และการจัดการความผิดพลาด เป็นต้น

2.1.2 กริดเซอร์วิส

กริดเซอร์วิส [8] เกิดจากความต้องการในการสร้างมาตรฐานการสร้างและให้บริการต่างๆ ที่อยู่ในระบบกริดให้เป็นรูปแบบเดียวกัน จึงได้มีการออกแบบของสถาปัตกรรมกริดเซอร์วิสแบบเปิดหรือโอจีเอสเอ (OGSA – Open Grid Services Architecture) เพื่อเป็นสถาปัตยกรรมพื้นฐานสำหรับการทำงานร่วมกันของเซอร์วิสต่างๆ ภายในระบบกริด โดยมีพื้นฐานอยู่บนสถาปัตยกรรมเว็บเซอร์วิส (Web Services Architecture) และมีการกำหนดขึ้นมาเป็นข้อกำหนดมาตรฐานที่เรียกว่า โครงสร้างพื้นฐานกริดเซอร์วิสแบบเปิด หรือ โอจีเอสไอ (OGSI - Open Grid Services Infrastructure) [9] ดังนั้นจึงอาจกล่าวได้ว่า กริดเซอร์วิสหมายถึงเซอร์วิสที่มีคุณสมบัติตามที่ระบุในข้อกำหนดโครงสร้างพื้นฐานกริดเซอร์วิสแบบเปิด

จากที่กล่าวมาแล้วมาว่ากริดเซอร์วิสถูกพัฒนาขึ้นภายใต้สถาปัตยกรรมเว็บเซอร์วิส จึงมีการนำมาตรฐานของเว็บเซอร์วิสมาพัฒนาปรับปรุงเพิ่มเติม อันได้แก่ การนำมาตรฐานดับบลิวเอสดีแอล (WSDL – Web Services Description Language) [14] มาใช้ในการบรรยายกริดเซอร์วิส โดยทำการปรับปรุงจนเป็นมาตรฐานจีดับบลิวเอสดีแอล (GWSDL – Grid Web Services Description Language) ที่อนุญาตให้มีการสืบทอดคุณสมบัติ (Inheritance) จากกริดเซอร์วิสหนึ่งไปยังอีกกริดเซอร์วิสหนึ่ง และทำการส่งข้อมูลโดยใช้โปรโตคอลโซป (SOAP – Service-Oriented Architecture Protocol) [15] นอกจากนั้นข้อกำหนดโอจีเอสไอยังกำหนดพอร์ทไทป์ (PortType) มาตรฐานสำหรับรอ งรับการทำงานของกลไกต่างๆ เพื่อรองรับการทำงานที่เหมาะสมกับระบบกริด กลไกและลักษณะพิเศษที่ข้อกำหนดโอจีเอสไอกำหนดขึ้น ได้แก่

- โรงงานเซอร์วิส (Service Factory) อนุญาตให้ผู้ใช้สร้างกริดเซอร์วิสอินสแตนท์ (Grid Service Instance) สำหรับทำงานตามที่ต้องการได้เอง และติดต่อกับกริดเซอร์วิสอินสแตนท์ที่สร้างขึ้นแบบจำสถานะได้ (Stateful Interaction) จนกว่ากริดเซอร์วิส

อินสแตนท์ที่สร้างขึ้นจะตายหรือถูกทำลาย การใช้งานกริดเซอร์วิสอินสแตนท์นี้ลักษณะคล้ายกับการโปรแกรมเชิงวัตถุ (Object-Oriented Programming) ซึ่งไม่ปรากฏในการใช้งานเว็บเซอร์วิสทั่วไป

- การจัดการสถานะ (State Management) เมื่อผู้ใช้สร้างกริดเซอร์วิสอินสแตนท์ขึ้นนั้น ผู้ใช้สามารถกำหนดเวลาเลิก (Termination Time) ของกริดเซอร์วิสอินสแตนท์ เมื่อถึงเวลากริดเซอร์วิสอินแสตนท์ก็จะมีการทำลายตัวเอง (Self-Destroy) นอกจากนั้น ผู้ใช้ยังสามารถทำลาย (Destroy) กริดเซอร์วิสอินสแตนท์ด้วยตนเองได้อีกด้วย
- การอ้างอิงเซอร์วิส (Service References) กริดเซอร์วิสต่างๆจะมีสิ่งที่เรียกว่า จีเอสเอช (GSH – Grid Service Handler) และ จีเอสอาร์ (GSR – Grid Service Reference) จีเอสเอช เป็นสิ่งที่กริดเซอร์วิสใช้ในการอ้างอิงถึงกริดเซอร์วิสอินสแตนท์ที่เฉพาะเจาะจงในรูปแบยูอาร์ไอ (URI) ส่วนจีเอสอาร์ถูกกำหนดให้เป็นข้อมูลที่จำเป็นสำหรับไคลเอนท์ในการเรียกใช้กริดเซอร์วิสอินสแตนท์ ดังนั้น เมื่อต้องการเรียกใช้งานกริดเซอร์วิสใดๆ ไคลเอนท์จะต้องทราบจีเอสเอชของกริดเซอร์วิสอินสแตนท์นั้น จากนั้นจึงใช้จีเอสเอชที่มีอยู่เรียกข้อมูลจีเอสอาร์ซึ่งจำเป็นในการเรียกใช้กริดเซอร์วิสต่อไป โดยปรกติแล้วข้อมูลในจีเอสอาร์จะอยู่ในรูปของจีดับบลิวเอสดีแอล สิ่งสำคัญอย่างหนึ่งก็คือ ข้อมูลจีเอสอาร์อาจมีการเปลี่ยนแปลงได้เสมอ โดยทั่วไปแล้วสิ่งที่เปลี่ยนแปลงก็คือข้อมูลการเชื่อมต่อ (Binding Information) ซึ่งหมายถึงว่าขณะที่กริดเซอร์วิสหนึ่งทำงานอยู่อาจมีการเปลี่ยนแปลงจุดอ้างอิงปลายทาง (Endpoint Reference) ที่ใช้ในการติดต่อกับกริดเซอร์วิสนั้นได้ตลอดเวลา
- ข้อมูลเซอร์วิส (Service Data) ผู้สร้างกริดเซอร์วิสสามารถกำหนดข้อมูลต่างๆ ที่เกี่ยวข้องกับกริดเซอร์วิสอินสแตนท์นั้นไว้อย่างชัดแจ้งภายในเอกสารจีดับบลิวเอสดีแอล เมื่อผู้ใช้ใช้งานกริดเซอร์วิสอินสแตนท์ก็สามารถเรียกดูข้อมูลที่กำหนดไว้ได้ผ่านทางกลไกมาตรฐานที่กำหนดไว้ของกริดเซอร์วิส ข้อมูลเซอร์วิสมักจะเป็นข้อมูลที่เกี่ยวข้องกับกริดเซอร์วิสอินสแตนท์นั้นโดยเฉพาะ และมักจะมีการเปลี่ยนแปลงอยู่เสมอ
- กลไกการแจ้งข่าว (Notification Mechanisms) กริดเซอร์วิสอินสแตนท์ที่ทำหน้าที่เป็นแหล่งต้นทางการแจ้งข่าว (Notification Source) สามารถใช้กลไกนี้เพื่อส่งข้อมูลที่สำคัญไปให้แก่กริดเซอร์วิสที่ทำหน้าที่เป็นแหล่งปลายทางการแจ้งข่าว (Notification Sink) ที่แจ้งความจำนงบอกรับ (Subscribe) ไว้ ในรูปแบบของการสื่อสารแบบอสม

วาร (Asynchronous Communication) แหล่งปลายทางการแจ้งข่าวสามารถกำหนดรูปแบบของข้อมูลที่ต้องการให้แจ้งข่าวกลับในขั้นตอนการแจ้งความจำนงได้

ลักษณะพิเศษ (Features) และกลไกต่างที่เพิ่มขึ้นมาเหล่านี้ทำให้กริดเซอร์วิสเหมาะกับการนำไปใช้สร้างเซอร์วิสที่มีความเหมาะสมในการทำงานในระบบกริด เช่น มีความคงทนต่อความผิดพลาด หรือมีการทำงานเป็นเวลานาน เป็นต้น แต่ข้อสำคัญอย่างหนึ่งก็คือข้อกำหนดโอจีเอสไอ ไม่ได้มีการกล่าวถึงกลไกทางด้านความปลอดภัย (Security Mechanisms) ใดๆ ไว้ในข้อกำหนดโอจีเอสไอเลย เพื่อให้ผู้พัฒนาซอฟต์แวร์กริดเซอร์วิสหรือผู้สร้างกริดเซอร์วิสสามารถเลือกใช้กลไกความปลอดภัยได้ตามต้องการ ตัวอย่างของกลไกคลวามปลอดภัยสำหรับกริดเซอร์วิส ก็คือ การใช้งานข้อกำหนดดับบลิวเอส-ซีเคียวริตี้ (WS-Security Specification) เป็นต้น สำหรับซอฟต์แวร์ที่ถูกพัฒนาขึ้นมาสำหรับสร้างและใช้งานกริดเซอร์วิสที่ใช้เป็นซอฟต์แวร์อ้างอิง ก็คือ โกลบัสทูลคิทเวอร์ชั่น 3

2.1.3 โครงสร้างพื้นฐานความปลอดภัยกริด (Grid Security Infrastructure – GSI) [11]

จีเอสไอถูกออกแบบให้อยู่บนพื้นฐานของการเข้ารหัสกุญแจสาธารณะ (Public Key Cryptography) เพื่อเป็นโครงสร้างพื้นฐานความปลอดภัยให้แก่การทำงานบนระบบกริด ทีมีการทำงานร่วมกันระหว่างทรัพยากรต่างๆ ซึ่งทำงานอยู่ในองค์กรจริงที่ต่างกัน จีเอสไอยังมีจุดมุ่งหมายในการสร้างรูปแบบการเข้าใช้ระบบเพียงครั้งเดียว (Single Sign-On) และการส่งผ่านสิทธิ (Credential Delegation) เมื่อมีการใช้งานทรัพยากรจากหลายแหล่งพร้อมกัน โดยการทำงานของจีเอสไอนั้นจะอาศัยสิ่งที่เรียกว่า ใบรับรอง (Certificate) เพื่อทำให้เกิดการยืนยันตัวตนร่วมกัน (Mutual Authentication) ระหว่างทั้งสองฝ่ายที่เกี่ยวข้องกับการใช้ทรัพยากรนั้น

จีเอสไอที่ใช้บนโกลบัสทูลคิทเวอร์ชั่น 2 และ 3 นั้นจะมีความแตกต่างกันในด้านของเทคโนโลยีพื้นฐานที่ใช้ คือ โกลบัสทูลคิทเวอร์ชั่น 2 จะทำงานบนเทคโนโลยีความปลอดภัยบนชั้นทรานสปอร์ตหรือทีแอลเอส (Transport Layer Security – TLS) สำหรับโกลบัสทูลคิทเวอร์ชั่น 3 ที่การทำงานต่างๆ จะอยู่ในรูปของการสร้างและใช้งานกริดเซอร์วิส ทำให้มีการนำเทคโนโลยีความปลอดภัยของเว็บเซอร์วิสและเอ็กซ์เอ็มแอลมาใช้เป็นจำนวนมาก ได้แก่ ดับบลิวเอส-ซีเคียรริตี้ (WS-Security) [16] เอ็กซ์เอ็มแอล เอ็นคริพชั่น (XML Encryption) [16] และ เอ็กซ์เอ็มแอล ซิกเนเจอร์ (XML Signature) [17] ซึ่งเทคโนโลยีเหล่านี้เน้นความปลอดภัยในระดับข้อความ (Message

Level Security) ทำให้สามารถนำกลไกความปลอดภัยเหล่านี้ไปใช้กับโปรโตคอลการติดต่อแบบใดก็ได้ ไม่ผูกติดกับเทคโนโลยีเหมือนกับเทคโนโลยีทีแอลเอส

2.1.4 ระบบจัดการกระแสงาน (Workflow Management System)

กระแสงาน (Workflow) สามารถนิยามได้ว่าเป็นการกำหนดการทำงานร่วมกันของกิจกรรมต่างๆ เพื่อบรรลุเป้าหมายที่ใหญ่ขึ้นและมีความสลับซับซ้อนมากกว่า [18] ดังนั้นระบบจัดการกระแสงาน (Workflow Management System) จึงหมายถึง ระบบที่ช่วยในการจัดการการทำงานร่วมกันของกิจกรรมต่างๆที่ประกอบกันขึ้นเป็นกระแสงานหนึ่ง ระบบจัดการกระแสงานเริ่มมาจากความต้องการในการอธิบายกระบวนการทำงานของระบบสารสนเทศในสำนักงาน (Office Information System) ด้วยโมเดลทางคณิตศาสตร์ แต่ก็ไม่ได้รับการยอมรับหรือประสบผลสำเร็จมากนักในตอนต้น แต่ก็ได้มีการพัฒนาต่อมาจนกลายเป็นระบบจัดการกระแสงานในที่สุด

ตั้งแต่เริ่มต้นจนถึงปัจจุบัน การพัฒนาระบบจัดการกระแสงาน ได้เกิดปัญหาขึ้นต่างๆอย่างมาก ทั้งปัญหาทางด้านเทคนิคและปัญหาทางด้านแนวคิด (Concept) แต่ปัญหาสำคัญที่สุดก็คือ การขาดมาตรฐานในการอธิบายขั้นตอนต่างๆ (Process Modeling) ในระบบกระแสงาน ระบบจัดการกระแสงานแต่ละตัวล้วนถูกพัฒนาขึ้นภายใต้พื้นฐานเทคโนโลยีและมาตรฐานที่แตกต่างกัน และระบบจัดการกระแสงานที่สร้างขึ้นนั้นมักจะถูกสร้างขึ้นมาสำหรับงานเฉพาะด้าน เช่น งานทางด้านธนาคารหรือประกันภัย

หน้าที่หลักของระบบจัดการกระแสงาน คือ ช่วยอำนวยความสะดวกในการประสานการทำงานระหว่างงานย่อยต่างๆ ที่ทำงานอยู่บนเครื่องต่างๆกัน ระบบจะต้องทำหน้าที่ติดต่อกับทั้งงานย่อยต่างๆ และกับผู้ใช้ เราสามารถแบ่งส่วนประกอบของระบบจัดการกระแสงานได้เป็นส่วนต่างๆ ดังต่อไปนี้

2.1.4.1 ข้อกำหนดกระแสงาน (Workflow Specification)

ข้อกำหนดกระแสงานเป็นมาตรฐานสำหรับการกำหนดรูปแบบการทำงานร่วมกันระหว่างงานย่อยต่างๆ ข้อกำหนดนี้ ระบบจัดการกระแสงานแต่ละระบบจะทำการกำหนดรูปแบบของข้อกำหนดกระแสงานขึ้นมา จากนั้นผู้ใช้จะทำการเขียนเอกสารกระแสงาน (Workflow Document) ตามข้อกำหนดกระแสงานเพื่อให้ระบบจัดการกระแสงานทำงานตามที่ผู้ใช้ต้องการ

คุณสมบัติต่างๆที่ระบบจัดการกระแสงานจะรองรับนั้นขึ้นอยู่กับข้อกำหนดกระแสงานของแต่ละระบบ เช่น การจัดการความผิดพลาด การควบคุมการไหลของข้อมูล หรือการทำงานแบบเงื่อนไข เป็นต้น นอกจากนั้น ยังได้มีการกำหนดมาตรฐานของข้อกำหนดกระแสงานที่เป็นมาตรฐานขึ้น เพื่อให้ผู้พัฒนาระบบกระแสงานนั้นไปใช้เป็นมาตรฐาน ดังนั้น เมื่อผู้ใช้พัฒนาเอกสารกระแสงานที่ตรงตามมาตรฐาน ก็จะสามารถนำเอกสารกระแสงานนั้นไปทำงานบนระบบจัดการกระแสงานใดก็ได้ ที่รองรับมาตรฐานข้อกำหนดกระแสงานนี้

2.1.4.2 ระบบทำงานกระแสงาน (Workflow Engine)

ระบบทำงานกระแสงานทำหน้าที่ควบคุมการทำงานที่กำหนดในเอกสารกระแสงาน โดยเมื่อผู้ใช้ทำการส่งเอกสารกระแสงานเข้ามายังระบบทำงานกระแสงาน ระบบจะทำงานตามที่ผู้ใช้กำหนดเข้ามาในเอกสารกระแสงาน เมื่อระบบทำงานกระแสงานทำงานเสร็จสิ้นหรือเกิดความผิดพลาดขึ้นในการทำงาน ระบบจะทำการส่งสถานการณ์ทำงานที่เกิดขึ้นกับมายังผู้ใช้ หรือตำแหน่งตามที่ผู้ใช้กำหนด ระบบทำงานกระแสงานสามารถทำงานร่วมกับเซิร์ฟเวอร์ที่มีอยู่แล้ว หรือทำงานเป็นเซิร์ฟเวอร์ที่ทำงานโดยเฉพาะก็ได้

2.1.4.3 ตัวสร้างกระแสงาน (Workflow Editor)

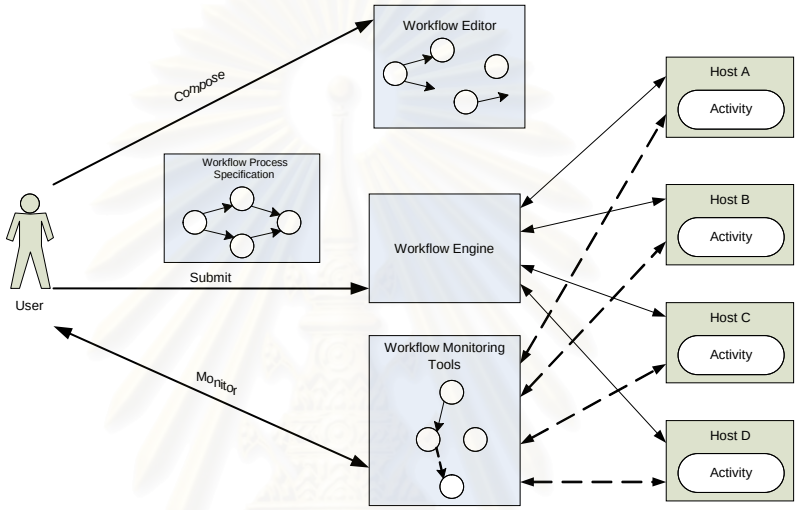
ผู้ใช้สามารถสร้างและแก้ไขเอกสารกระแสงานได้ผ่านทางตัวสร้างกระแสงาน ซึ่งจะช่วยอำนวยความสะดวกในการพัฒนาระบบกระแสงาน บริษัทต่างๆได้ทำการพัฒนาตัวสร้างกระแสงานต่างๆกันไปตามข้อกำหนดกระแสงานของแต่ละระบบ มีตั้งแต่ ตัวสร้างกระแสงานในรูปแบบวิชวล (Visual Workflow Editor) ที่ผู้ใช้สามารถออกแบบกระบวนการกระแสงานแบบลากแล้วปล่อย (Drag and Drop) จนถึงตัวสร้างกระแสงานที่เป็นในรูปแบบตัวอักษรธรรมดา นอกจากนั้นตัวสร้างกระแสงานบางตัวยังมีความสามารถในการดีพลอย (Deploy) กระบวนการกระแสงานที่สร้างขึ้นไปยังระบบทำงานกระแสงานได้อย่างอัตโนมัติ

2.1.4.4 เครื่องมือตรวจสอบกระแสงาน (Workflow Monitoring Tools)

เครื่องมือตรวจสอบกระแสงานใช้ในการตรวจสอบการสถานะการทำงานของกระบวนการกระแสงานต่างๆที่กำลังทำงานอยู่ โดยเครื่องมือนี้จะแสดงถึงขั้นที่กระบวนการกระแสงานนั้นกำลังทำงานอยู่ และถ้ามีความผิดพลาดใดๆเกิดขึ้นเครื่องมือก็จะแสดงถึงปัญหาและต้นตอ

ของปัญหาที่เกิดขึ้น นอกจากนั้นเครื่องมือตรวจสอบกระแสงานยังสามารถใช้แสดงและวิเคราะห์ความสัมพันธ์ของงานย่อยต่างๆ เพื่อที่ผู้ใช้จะสามารถทราบปัญหาและแก้ไขกระบวนการกระแสงานที่จะสร้างขึ้นได้ในอนาคต

เราสามารถแสดงภาพรวมของระบบจัดการกระแสงานที่ประกอบจากส่วนประกอบย่อยทั้ง 4 ส่วน ได้ตามรูปที่ 1



รูปที่ 1 ส่วนประกอบของระบบจัดการกระแสงาน

2.1.5 บีเพล (BPEL) [7]

บีเพล (BPEL – Business Process Execution Language) ถูกออกแบบมาสำหรับเป็นภาษาที่ใช้ในการประกอบเซอร์วิสเข้าด้วยกันในรูปแบบกระแสงาน (Workflow-based Service Composition Language) เป้าหมายเริ่มต้นของการออกแบบบีเพล คือ การสร้างภาษาสำหรับบรรยายการทำงานร่วมกันระหว่างเว็บเซอร์วิส (Coordination) บีเพลสามารถใช้กำหนดกระบวนการเว็บเซอร์วิส (Web Services Process) ได้ทั้งกระบวนการนามธรรม (Abstract Process) ที่ใช้ในการสร้างข้อกำหนดของกระบวนการต่างๆ (Process Specification) และกระบวนการที่สามารถทำงานได้จริง (Executable Process) ที่ใช้ในงานสร้างเซอร์วิสใหม่จากเซอร์วิสที่มีอยู่เดิมทำงานร่วมกัน (Composite Web Services)

บีเพลเป็นมาตรฐานที่เกิดจากการรวมข้อดีของภาษากระแสงาน 2 ภาษาที่มีอยู่เดิมในเว็บเซอร์วิส ได้แก่ ดับบลิวเอสเอฟแอล (WSFL – Web Services Flow Language) [19] และ

เอ็กซ์แลง (XLANG) [20] โดยดับบลิวเอสเอฟแอลเป็นภาษากระแสงานในเชิงกราฟ ที่มองในรูปการไหลของข้อมูลเชื่อมต่อจากเว็บเซอร์วิสหนึ่งไปยังอีกเว็บเซอร์วิสหนึ่ง ส่วนเอ็กซ์แลงเป็นภาษากระแสงานในเชิงโครงสร้าง ดังนั้นบีเพลจึงกลายเป็นภาษาที่มีความสามารถในการบรรยายการทำงานของเว็บเซอร์วิสต่างๆในเชิงกราฟหรือการไหลได้ดี พร้อมทั้งข้อดีของภาษาที่มีความสามารถคล้ายกับภาษาโปรแกรมเชิงโครงสร้าง (Structured Programming Language) ทำให้บีเพลได้รับความนิยมในการใช้เป็นภาษากระแสงาน (Workflow Language) ของเว็บเซอร์วิสอย่างรวดเร็ว

บีเพลถูกสร้างขึ้นบนมาตรฐานดับบลิวเอสดีแอล และข้อกำหนดเอ็กซ์เอ็มแอลที่สำคัญอื่นๆ ได้แก่ เอ็กซ์เอ็มแอลสกีมา (XML-Schema) [21] เอ็กซ์-พาธ (X-Path) [22] และ ดับบลิวเอส-แอดเดรสซิ่ง (WS-Addressing) [23] แนวคิดที่สำคัญของบีเพล ก็คือ การแบ่งเว็บเซอร์วิสที่เกี่ยวข้องออกเป็นพาร์ทเนอร์ (Partner) โดยแต่ละพาร์ทเนอร์ที่ทำงานประสานกันนั้น จะต้องมีเอกสารดับบลิวเอสดีแอลที่บรรยายการทำงานของพาร์ทเนอร์นั้น จากนั้นจึงทำการกำหนดชื่อที่แน่นอนให้แก่พาร์ทเนอร์ไว้ใช้ในการบรรยายการทำงานของกระบวนการที่จะสร้างขึ้นต่อไป

นอกจากการกำหนดพาร์ทเนอร์ที่เกี่ยวข้องในการทำงานของกระบวนการนั้นแล้ว ผู้พัฒนากระบวนการ (Process Developer) ยังสามารถกำหนดตัวแปรของข้อมูลต่างๆ (Variables) ที่ใช้ในการทำงานได้ ในส่วนของขั้นตอนการทำงานของกระบวนการทำงานนั้น บีเพลมีความสามารถในการบรรยายการทำงานในหลากหลายรูปแบบทั้งการทำงานตามลำดับ (Sequential) การทำงานแบบขนาน (Parallel) การทำงานแบบลูป การทำงานแบบตัวเลือก (Selection) นอกจากนั้นยังสามารถกำหนดตัวจัดการ (Handler) ไว้ใช้ในการจัดการความผิดพลาด การทำงานตามเวลาหรือการทำงานตามข้อความได้อีกด้วย

2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.2.1 Orchestration of an OGSI-enabled scientific application using the Business Process Execution Language [24]

งานวิจัยนี้นำเสนอการวิเคราะห์ข้อบกพร่องของบีเพลที่ทำให้ไม่เหมาะกับการใช้งานในการสร้างแอพพลิเคชั่นกริด ซึ่งได้นำเสนอข้อบกพร่องต่างๆ ของบีเพล คือ

- สามารถกำหนดข้อมูลการเชื่อมต่อ (Binding Information) ได้ในขณะการสร้างกระบวนการเท่านั้น (Deployment-time Binding)
- ไม่สนับสนุนคุณสมบัติการสืบทอดและการบรรยายข้อมูลเซอร์วิสในจีดับบลิวเอสดีแอล
- ขาดกลไกในการรับรู้ข้อมูลการเชื่อมต่อจากแหล่งต่างๆ เช่น ยูดีดีไอ (UDDI – Universal Description, Discovery, and Integration) [25] เป็นต้น
- ไม่มีกลไกในการกำหนดการสื่อสารแบบเซอร์วิสถึงเซอร์วิส (Service-to-Service Communication)
- กระบวนการทำงานไม่สามารถเปลี่ยนแปลงได้ในระหว่างการทำงาน

ข้อสรุปของงานวิจัยนี้ ก็คือ เราสามารถใช้บีเพลในการประสานการทำงานร่วมกันระหว่างกริดเซอร์วิสที่พัฒนาขึ้นตามข้อกำหนดโอจีเอสไอได้ โดยไม่ต้องมีการเพิ่มเติมจากข้อกำหนดมาตรฐานเลย เพียงแต่จะต้องมีการพัฒนาระบบทำงานบีเพล (BPEL Engine) ให้สามารถรองรับการทำงานกับกริดเซอร์วิสได้ นอกจากนั้นงานวิจัยนี้ยังได้สร้างระบบทำงานบีเพลต้นแบบที่สามารถประสานการทำงานกริดเซอร์วิสที่ห่อหุ้มโปรแกรมที่พัฒนาขึ้นด้วยภาษาฟอร์ทรานได้

ข้อสังเกตของงานวิจัยชิ้นนี้ก็คือ ไม่มีการกล่าวถึงกลไกและลักษณะพิเศษของกริดเซอร์วิสที่แตกต่างจากเว็บเซอร์วิสทั่วไป ซึ่งถือว่าเป็นจุดเด่นเลย กล่าวถึงเพียงแต่การแก้ปัญหากลไกการอ้างอิงชื่อเซอร์วิส (Service Reference) เท่านั้น ซึ่งงานวิจัยนี้แก้ปัญหาโดยการสร้างส่วนประกอบเพิ่มเติม (Component) ขึ้นมาทำหน้าที่ในการติดต่อระหว่างบีเพลกับกริดเซอร์วิสเท่านั้น

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

2.2.2 On Using BPEL Extensibility to Implement OGSI and WSRF Grid Workflows [26]

งานวิจัยชิ้นนี้มุ่งเน้นในการนำข้อกำหนดบีเพลมาใช้งานกับระบบบกริดโดยมีจุดมุ่งหมายอยู่ที่ระบบกริดที่สร้างขึ้นภายใต้ข้อกำหนดโอจีเอสไอ และข้อกำหนดดับบลิวเอสอาร์เอฟ สาเหตุที่เลือกระบบกริดทั้ง 2 ระบบนี้เนื่องมาจากเป็นระบบกริดที่ทำงานอยู่บนสถาปัตยกรรมเชิงเซอร์วิส (Service-Oriented Architecture) ที่เป็นมาตรฐานที่สุดของระบบกริด ดังนั้นงานวิจัย

ชิ้นนี้จึงถูกแบ่งออกเป็น 2 ส่วน คือ การใช้งานบีเพลร่วมกับกริดเซอร์วิสตามมาตรฐานโอจีเอสไอ และการใช้งานบีเพลร่วมกับดับบลิวเอสอาร์เอฟ

ในส่วนแรกนั้นงานวิจัยนี้ได้เสนอแนะถึงการพัฒนาระบบทำงานบีเพล (BPEL Engine) ในรูปของกริดเซอร์วิส สาเหตุที่เราสามารถทำการสร้างระบบทำงานบีเพลเป็นกริดเซอร์วิสได้เนื่องมาจากข้อกำหนดบีเพลไม่ได้กล่าวถึงรายละเอียดในการพัฒนาระบบทำงาน (Engine) เพียงแต่กล่าวถึงว่ากระบวนการบีเพลจะต้องสามารถแสดงอยู่ในรูปดับบลิวเอสดีแอลได้เท่านั้น สาเหตุหลักของการออกแบบให้ระบบทำงานบีเพลอยู่ในรูปแบบของกริดเซอร์วิสเกิดจากความจำเป็นที่จะให้บีเพลสามารถทำงานร่วมกันกริดเซอร์วิสไคเอนท์ได้อย่างสมบูรณ์ สิ่งที่ตามมาจากออกแบบระบบในรูปแบบนี้ก็คือ การจัดการอินสแตนท์ของกระบวนการบีเพลจะสามารถถูกจัดการในรูปของกริดเซอร์วิสอินสแตนท์ได้ แทนที่รูปแบบเดิมที่การจัดการอินสแตนท์ของกระบวนการบีเพลจะขึ้นอยู่กับสิ่งที่เรียกว่า ตัวระบุความสัมพันธ์ (Correlation Set) เท่านั้น ทำให้เราต้องเพิ่มกลไกการสร้างกระบวนการบีเพลใหม่ เนื่องมาจากกริดเซอร์วิสอนุญาตให้สร้างอินสแตนท์ใหม่ได้จากโรงงานเซอร์วิสเท่านั้น แต่ตามข้อกำหนดบีเพลจะมีการสร้างอินสแตนท์ใหม่ผ่านทางกลไกอื่นๆ ดังนั้น งานวิจัยนี้จึงนำเสนอการออกแบบให้ระบบทำงานบีเพลมีกลไกการสร้างบีเพลอินสแตนท์ใหม่ผ่าน 2 ขั้นตอนด้วยกัน สำหรับการใช้บีเพลเรียกใช้งานกริดเซอร์วิสนั้น งานวิจัยนี้ชี้ให้เห็นว่า ระบบทำงานบีเพลที่สนับสนุนกริดเซอร์วิสจะต้องรองรับแนวคิดการอ้างอิงโดยใช้จีเอสเอชและจีเอสอาร์

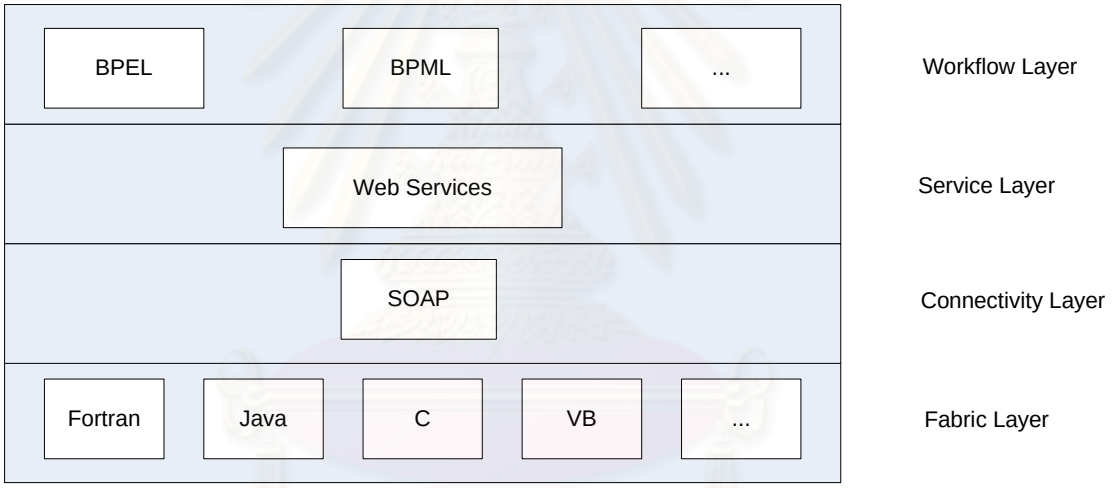
ในส่วนของการใช้งานบีเพลร่วมกับข้อกำหนดดับบลิวเอสอาร์เอฟนั้น งานวิจัยชิ้นนี้ชี้ให้เห็นว่า เนื่องจากข้อกำหนดดับบลิวเอสอาร์เอฟนั้นถูกออกแบบให้สามารถใช้งานและทำงานร่วมกับมาตรฐานและข้อกำหนดต่างๆในวงการเว็บเซอร์วิสตั้งแต่ต้น ทำให้ไม่มีความจำเป็นในการใช้งานเซอร์วิสแทนหรือการพัฒนาให้ระบบทำงานบีเพลอยู่ในรูปของกริดเซอร์วิส ดังที่ได้กล่าวในส่วนแรก

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 3

การออกแบบและพัฒนากรอบการประสานการทำงานร่วมกันของบีเพล สำหรับกริดเซอร์วิส

การทำงานของระบบกระแสงานในระบบกระจายโดยทั่วไปนั้นจะเกี่ยวข้องกับเทคโนโลยีต่างๆเป็นจำนวนมาก เช่น โปรโตคอลการสื่อสารในระดับต่างๆ ไปจนถึง มิดเดิลแวร์ (Middleware) เป็นต้น จากการศึกษาและวิเคราะห์ระบบกระแสงานบนสถาปัตยกรรมเว็บเซอร์วิส เราสามารถแสดงแบบจำลองสถาปัตยกรรมสำหรับระบบกระแสงานบนสถาปัตยกรรมเว็บเซอร์วิสได้ดังรูปที่ 2

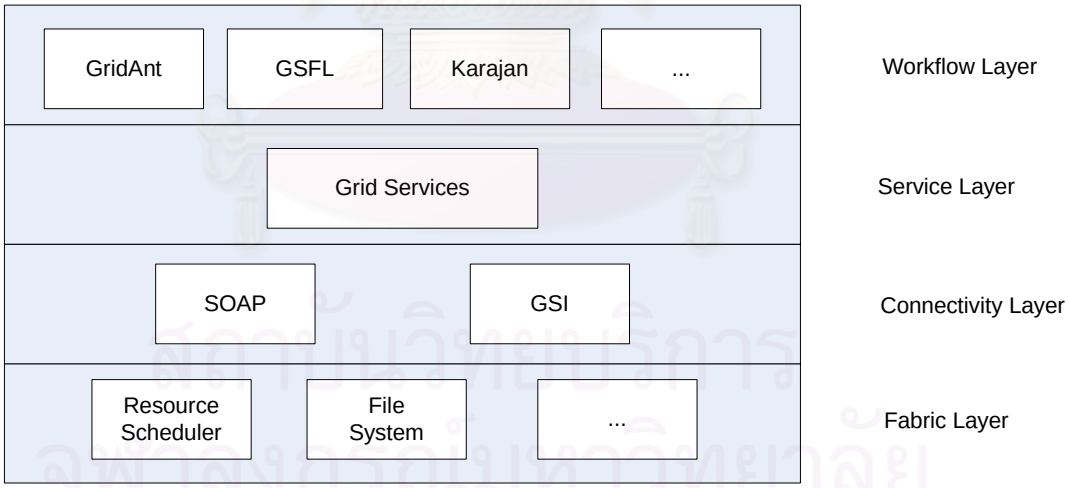


รูปที่ 2 ข้อกำหนดและมาตรฐานต่างๆที่เกี่ยวข้องกับกับระบบกระแสงานบนสถาปัตยกรรมเว็บเซอร์วิส

จากรูปที่ 2 ภายในแต่ละชั้นของแบบจำลองจะแสดงถึงมาตรฐานหรือข้อกำหนดต่างๆที่เกี่ยวข้องกับการทำงานของชั้นนั้นๆ และเนื่องจากสถาปัตยกรรมเว็บเซอร์วิสนั้นได้ถูกออกแบบมาให้เป็นสถาปัตยกรรมเชิงเซอร์วิส (Service-Oriented Architecture) อย่างสมบูรณ์ ทำให้เราสามารถแสดงสถาปัตยกรรมชั้นล่างสุดได้เป็นชั้นแฟบริค (Fabric Layer) ซึ่งเป็นชั้นที่แสดงถึงการทำงานต่างๆของเว็บเซอร์วิสที่เปิดให้บริการ สำหรับการทำงานของเว็บเซอร์วิสนั้นอาจเป็นได้ตั้งแต่โค้ดภาษาจาวาไปจนถึงโปรแกรมขององค์กรที่มีอยู่เดิมก็เป็นได้ แล้วจึงถูกซ้อนทับด้วยชั้นการเชื่อมต่อ (Connectivity Layer) ซึ่งในชั้นนี้จะแสดงถึงเทคโนโลยีที่เกี่ยวข้องในการติดต่อและเรียกใช้เว็บเซอร์วิส ในชั้นต่อมา คือ ชั้นเซอร์วิส (Service Layer) ในชั้นนี้จะแสดงถึงว่าใน

สถาปัตยกรรมเว็บเซอร์วิสนั้น เซอร์วิสทุกอย่างจะต้องอยู่ในรูปของเว็บเซอร์วิส และชั้นบนสุดก็คือชั้นกระแสงานที่มีการใช้แนวคิดกระแสงานประสานการทำงานต่างๆของเว็บเซอร์วิส

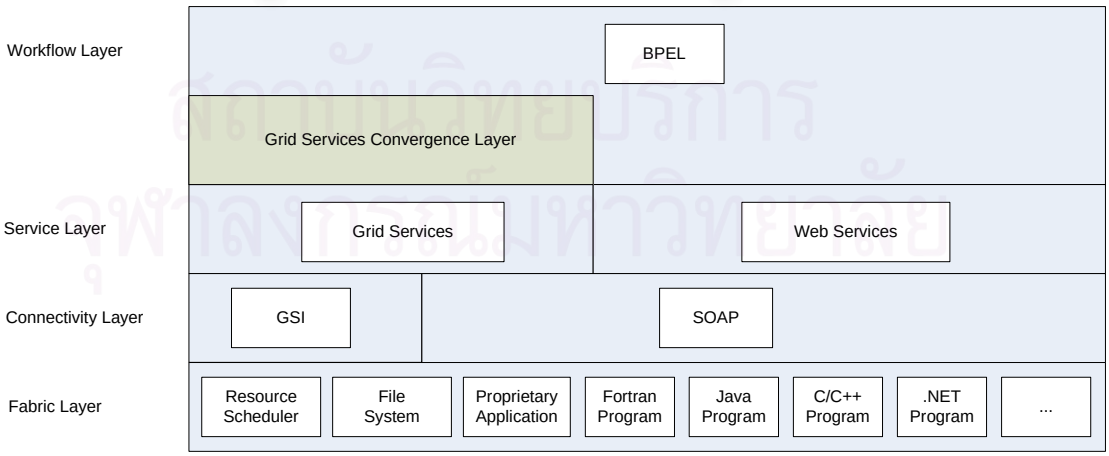
เมื่อเปรียบเทียบกับสถาปัตยกรรมระบบกระแสงานบนสถาปัตยกรรมกริดเซอร์วิส ซึ่งแม้ว่ากริดเซอร์วิสจะมีพื้นฐานอยู่บนเทคโนโลยีเว็บเซอร์วิส แต่จุดมุ่งหมายในการออกแบบก็เพื่อสร้างส่วนประกอบต่างๆในรูปแบบของเซอร์วิสเพื่อตอบสนองต่อความต้องการต่างๆในระบบกริด ซึ่งความปลอดภัยนั้นถือได้ว่าเป็นส่วนสำคัญมากที่สุดของระบบกริด ทำให้แบบจำลองจะมีความแตกต่างออกไป ซึ่งจุดที่สำคัญก็คือ ในชั้นการเชื่อมต่อ (Connectivity Layer) นั้นจะมีเทคโนโลยีที่เกี่ยวข้องและมีความสำคัญเพิ่มเข้ามา ก็คือ เทคโนโลยีจีเอสไอ ซึ่งเป็นเทคโนโลยีความปลอดภัยของระบบกริดนั่นเอง สำหรับในส่วนของชั้นล่างสุด ก็คือ ชั้นแฟบริคนั้น โดยทั่วไปสำหรับกริดเซอร์วิสจะเป็นชั้นที่เกี่ยวข้องกับการติดต่อกับทรัพยากรหรืองานต่างๆที่กริดเซอร์วิสนั้นรับผิดชอบอยู่ ในความจริงแล้วกริดเซอร์วิสก็อาจจะมีการทำงานเหมือนกับเว็บเซอร์วิส ซึ่งอาจเป็นโปรแกรมภาษาต่างๆที่เขียนขึ้นใหม่หรือเป็นโปรแกรมที่มีอยู่เดิมก็เป็นได้ ทำให้ชั้นแฟบริคของแบบจำลองนี้ก็จะมีลักษณะเหมือนกับชั้นแฟบริคของเว็บเซอร์วิสได้อีกด้วย รูปที่ 3 แสดงถึงข้อกำหนดและมาตรฐานต่างๆที่เกี่ยวข้องกับระบบกระแสงานที่ทำงานบนสถาปัตยกรรมกริดเซอร์วิส



รูปที่ 3 ข้อกำหนดและมาตรฐานต่างๆที่เกี่ยวข้องกับกับระบบกระแสงานบนสถาปัตยกรรมกริดเซอร์วิส

3.1 ชั้นประสานกริดเซอร์วิส (Grid Services Convergence Layer)

เมื่อพิจารณาจากแบบจำลองสถาปัตกรรมระบบกระแสงานของระบบกระจายทั้งสอง พบว่า มาตรฐานและข้อกำหนดที่เกี่ยวข้องในชั้นการเชื่อมต่อ (Connectivity Layer) ของระบบกระแสงานทั้งสองมีความแตกต่างกัน กล่าวคือ ชั้นการเชื่อมต่อของสถาปัตยกรรมของกริดเซอร์วิสจะมีการใช้เทคโนโลยีจีเอสไอ เพื่อการรักษาความปลอดภัยให้กับกริดเซอร์วิส แต่สำหรับสถาปัตยกรรมเว็บเซอร์วิสแล้วจะไม่รองรับเทคโนโลยีดังกล่าว ดังนั้นการที่จะนำบีเพลมาใช้เป็นระบบจัดการกระแสงานของสถาปัตยกรรมกริดเซอร์วิสทำให้เราจำเป็นจะต้องสร้างชั้นที่ทำหน้าที่ติดต่อระหว่างชั้นกระแสงานของระบบกระแสงานบนสถาปัตยกรรมเว็บเซอร์วิส กับชั้นเซอร์วิสของระบบกระแสงานบนสถาปัตยกรรมกริดเซอร์วิส โดยเราจะเรียกชั้นที่เพิ่มเติมขึ้นมานี้ว่า ชั้นประสานกริดเซอร์วิส (Grid Services Convergence Layer) หน้าที่ของชั้นนี้คือ การทำให้ระบบกระแสงานบีเพล สามารถประสานการทำงานระหว่างกริดเซอร์วิสต่างๆได้ ซึ่งจะทำให้เราสามารถใช้มาตรฐานบีเพลในการประสานการทำงานร่วมกันของทั้งกริดเซอร์วิสและเว็บเซอร์วิสได้เป็นอย่างดี หรืออาจกล่าวได้ว่า เป็นการทำให้กริดเซอร์วิสสามารถถูกมองเห็นได้ในรูปแบบของเว็บเซอร์วิสนั่นเอง ผลคือ เราสามารถนำบีเพลมาประสานการทำงานของกริดเซอร์วิสที่ถูกแปลงให้อยู่ในรูปเว็บเซอร์วิสได้ เราสามารถแสดงแบบจำลองระบบกระแสงานที่ใช้มาตรฐานบีเพลทำงานร่วมกับกริดเซอร์วิสและเว็บเซอร์วิสได้ดังรูปที่ 4



รูปที่ 4 แบบจำลองการทำงานร่วมกันระหว่างบีเพลกับกริดเซอร์วิส

วิทยานิพนธ์นี้จึงนำเสนอกรอบประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส ซึ่งจะทำหน้าที่เป็นชั้นประสานกริดเซอร์วิส โดยรายละเอียดของกรอบการทำงานที่ออกแบบและพัฒนาขึ้นจะกล่าวในส่วนต่อไป

3.2 ปัญหาที่พบในการใช้บีเพลประสานการทำงานระหว่างกริดเซอร์วิส

ในการออกแบบกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสนั้น เราจำเป็นจะต้องทราบปัญหาต่างๆที่จะเกิดขึ้น เมื่อมีการนำบีเพลมาใช้ประสานการทำงานระหว่างกริดเซอร์วิส สำหรับการนำบีเพลมาประสานการทำงานระหว่างกริดเซอร์วิสนั้นมีความเป็นไปได้เนื่องจากกริดเซอร์วิสได้รับการพัฒนาบนพื้นฐานของเทคโนโลยีเว็บเซอร์วิส โดยเฉพาะการเรียกใช้โอเปอร์เรชั่นต่างๆที่ผู้ใช้สร้างขึ้นเอง (User-Defined Operations) ที่พัฒนาขึ้นของกริดเซอร์วิสต่างๆ ซึ่งรวมไปถึงการเรียกใช้โอเปอร์เรชั่นเหล่านี้โดยใช้เครื่องมือเว็บเซอร์วิสทั่วไปก็สามารถทำได้เช่นกัน เนื่องมาจากการเรียกใช้โอเปอร์เรชั่นเหล่านี้สามารถถูกมองได้ว่าเป็นเหมือนกับโอเปอร์เรชั่นหนึ่งของเว็บเซอร์วิสอันหนึ่ง นอกจากนั้นกลไกการส่งข้อมูลของเว็บเซอร์วิสกับกริดเซอร์วิสก็ทำผ่านโปรโตคอลโซปเช่นเดียวกัน จึงไม่มีปัญหาเกิดขึ้นในการนำเครื่องมือบีเพลมาเรียกใช้โอเปอร์เรชั่นเหล่านี้ แต่สิ่งสำคัญที่ทำให้กริดเซอร์วิสไม่สามารถถูกใช้งานได้จากเครื่องมือบีเพลทั่วไปได้โดยตรง ก็คือ คุณสมบัติและกลไกต่างๆ ที่สร้างขึ้นตามข้อกำหนดโอจีเอสไอที่มีในกริดเซอร์วิสทุกตัว รวมไปถึงการใช้งานโครงสร้างพื้นฐานความปลอดภัยกริดหรือจีเอสไอในการรักษาความปลอดภัยให้แก่กริดเซอร์วิส ซึ่งจะได้กล่าวในรายละเอียดต่อไป

3.2.1 ปัญหาในการเรียกใช้คุณสมบัติและกลไกต่างๆของกริดเซอร์วิส

จากการศึกษาคุณสมบัติและกลไกต่างๆของกริดเซอร์วิสที่สร้างตามข้อกำหนดโอจีเอสไอ เราจะพบว่าคุณสมบัติและกลไกต่างๆเหล่านี้บางส่วน สามารถถูกเรียกใช้งานได้โดยตรงจากบีเพล ขณะที่อีกบางส่วนไม่สามารถถูกเรียกใช้ได้โดยตรงจากบีเพล เราสามารถสรุปได้ดังตารางที่ 1

ตารางที่ 1 แสดงคุณสมบัติและกลไกต่างๆของกริดเซอร์วิสที่สามารถเรียกใช้งานได้โดยตรงจากบีเพล

กลไกและคุณสมบัติของกริดเซอร์วิส	คุณสมบัติ	
	สามารถเรียกใช้ได้โดยตรงจากบีเพล	ไม่สามารถเรียกใช้ได้โดยตรงจากบีเพล
การอ้างอิงเซอร์วิส		✓
โรงงานเซอร์วิส		✓
การจัดการสถานะ	✓	
ข้อมูลเซอร์วิส	✓	
กลไกการแจ้งข่าว		✓
การสืบทอดพอร์ทไทป์		✓

จากตารางที่ 1 เราจะพบว่ากลไกการจัดการสถานะและข้อมูลเซอร์วิสนั้น สามารถถูกเรียกใช้ได้โดยตรงก็เนื่องมาจาก กลไกทั้งสองถูกสร้างขึ้นเป็นโอเปอร์เรชั่นหนึ่งของกริดเซอร์วิส เราจึงสามารถเรียกใช้กลไกเหล่านี้เหมือนกับการเรียกใช้โอเปอร์เรชั่นหนึ่งของเว็บเซอร์วิสทั่วไป สำหรับกลไกและคุณสมบัติที่ไม่สามารถเรียกใช้ได้โดยตรงจากบีเพลนั้น เราสามารถสรุปได้เป็นดังหัวข้อต่อไปนี้

- *การอ้างอิงอินสแตนท์ที่แตกต่างกัน* กริดเซอร์วิสได้นำแนวความคิดในการใช้งานจีเอสเอชและจีเอสอาร์มาใช้งานในการอ้างอิงถึงกริดเซอร์วิสอินสแตนท์แต่ละตัว ดังนั้นในการอ้างอิงถึงกริดเซอร์วิสอินสแตนท์แต่ละตัว เราจำเป็นจะต้องทราบจีเอสเอชของกริดเซอร์วิสอินสแตนท์ที่ต้องการนั้นก่อน แต่การอ้างอิงอินสแตนท์หรือกระบวนการแต่ละตัวของบีเพล จะใช้สิ่งที่เรียกว่า คอรีเลชั่นเซ็ท (Correlation Set) ในการแยกแยะอินสแตนท์ของกระบวนการบีเพลออกจากกัน ทำให้การส่งข้อความอสมวาร (Asynchronous Message) ระหว่างกระบวนการบีเพลกับกริดเซอร์วิสอินสแตนท์จะเกิดปัญหาขึ้น เนื่องจากแต่ละฝ่ายไม่สามารถรับรู้การอ้างอิงอินสแตนท์ที่แตกต่างกันได้ ซึ่งปัญหานี้จะเกิดขึ้นในกรณีที่เราต้องการให้กระบวนการบีเพลทำหน้าที่เป็นแหล่ง

ปลายทางการแจ้งข่าว (Notification Sink) เพื่อรอรับข้อมูลที่ส่งมาจากแหล่งต้นทางการแจ้งข่าว (Notification Source) ที่เป็นกริดเซอร์วิส

- *การเชื่อมต่อแบบพลวัฒน์ (Dynamic Binding)* แนวคิดในการใช้จีเอสเอชและจีเอสอาร์ทำให้ในระหว่างทำงานกริดเซอร์วิสสามารถเปลี่ยนการเชื่อมต่อ (Binding) หรือเปลี่ยนจุดอ้างอิงปลายทาง (Endpoint Reference) ทำให้ข้อมูลที่จำเป็นในการเรียกใช้งานกริดเซอร์วิสจะทราบได้ก็ต่อเมื่อมีการนำจีเอเอชไปพิสูจน์ทราบข้อมูลจีเอสอาร์เท่านั้น ซึ่งเราจะทราบจีเอสอาร์ได้ก็ต่อเมื่อกริดเซอร์วิสอินสแตนท์นั้นถูกสร้างและใช้จีเอสเอชในการร้องขอข้อมูลจีเอสอาร์ผ่านทางพอร์ทไทป์ที่ชื่อแฮนเดิลรีโซฟเวอร์ (HandleResolver portType) เท่านั้น และข้อมูลในจีเอสอาร์จะเป็นข้อมูลที่มีความสำคัญ โดยเฉพาะข้อมูลการเชื่อมต่อ (Binding Information) แต่ตามข้อกำหนดบีเพลนั้นเราจำเป็นจะต้องทราบข้อมูลการเชื่อมต่อของเซอร์วิสที่จะเรียกใช้ตั้งแต่ขั้นตอนการสร้างกระบวนการ หรืออีกนัยหนึ่งคือบีเพลใช้การเชื่อมต่อแบบสถิตเท่านั้น (Static Binding) ซึ่งไม่สามารถทำได้เมื่อเราใช้งานกริดเซอร์วิสมาตรฐาน นอกจากนั้นข้อมูลในจีเอสอาร์ยังสามารถเปลี่ยนแปลงได้ตามที่กล่าวใน ซึ่งบีเพลไม่สามารถรับรู้การเปลี่ยนแปลงเหล่านี้ได้เลย
- *การใช้งานกริดเซอร์วิสอินสแตนท์* โดยการใช้กลไกโรงงานเซอร์วิสทำให้ผู้ใช้สามารถสร้างกริดเซอร์วิสอินสแตนท์ใหม่ขึ้นใช้งานได้ตลอดเวลา กริดเซอร์วิสอินสแตนท์เหล่านี้ล้วนแต่มีข้อมูลการเชื่อมต่อที่ระบุอยู่ในดับบลิวเอสดีแอลที่แตกต่างกันไป และจากข้อกำหนดบีเพลที่อนุญาตให้กำหนดข้อมูลการเชื่อมต่อได้ตอนสร้างกระบวนการเท่านั้น หมายถึงเราจะต้องรู้ดับบลิวเอสดีแอลของเซอร์วิสที่ต้องการใช้งานตั้งแต่ตอนสร้างกระบวนการ ทำให้บีเพลไม่สามารถใช้กลไกโรงงานเซอร์วิสได้โดยตรง เนื่องจากไม่สามารถทราบดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์ที่ยังไม่ได้สร้างขึ้นจริงเหล่านี้ได้เลย
- *คุณสมบัติการสืบทอดของพอร์ทไทป์ (PortType Inheritance)* เนื่องจากภาษาบรรยายของกริดเซอร์วิสคือจีดับบลิวเอสดีแอล อนุญาตให้มีการสืบทอดทั้งโอเปอร์เรชั่นและข้อมูลเซอร์วิส แต่บีเพลไม่สามารถรับรู้คุณสมบัติเหล่านี้ได้เลยเนื่องจากใช้เพียงภาษาบรรยายมาตรฐานสำหรับเว็บเซอร์วิส ซึ่งก็คือ ดับบลิวเอสดีแอล ที่ไม่อนุญาตให้มีการสืบทอดคุณสมบัติของพอร์ทไทป์ใดๆทั้งสิ้น

3.2.2 ปัญหาในการใช้งานกลไกความปลอดภัยของกริดเซอร์วิส

เนื่องจากกลไกความปลอดภัยจีเอสไอของโกลบัสทูลคิทเวอร์ชั่น 3 อยู่บนพื้นฐานของดับบลิวเอส-ซีเคียวริตี้ และเทคโนโลยีความปลอดภัยอื่นๆของเว็บเซอร์วิสนั้น ซึ่งหลักการพื้นฐานที่สำคัญของดับบลิวเอส-ซีเคียวริตี้ก็คือ การใส่โทเคนความปลอดภัย (Security Token) เข้าไปในส่วนหัว (Header) ของข้อความโซป (SOAP Message) เพื่อระบุถึงข้อมูลต่างๆที่เกี่ยวข้องกับในการรักษาความปลอดภัยข้อมูลที่อยู่ในส่วนตัวของข้อความโซป (Body) แต่เนื่องจากข้อกำหนดบีเพลมาตรฐานไม่อนุญาตให้มีการจัดการใดกับส่วนหัวของข้อความโซปได้เลย ทำให้เราไม่สามารถใช้บีเพลติดต่อกับกริดเซอร์วิสที่มีการรักษาความปลอดภัยได้ เพราะไม่สามารถจัดการกับโทเคนความปลอดภัยเหล่านี้ได้โดยตรง ไม่ว่าจะเป็นในการส่งหรือรับข้อความที่มีความปลอดภัย (Secured Message) นอกจากนี้สำหรับการรักษาความปลอดภัยบางรูปแบบข้อมูลการเรียกใช้เว็บเซอร์วิสที่อยู่ในส่วนตัว (Body) ของข้อความโซปนั้นยังถูกเข้ารหัสไว้อีกด้วย ซึ่งตามข้อกำหนดบีเพลมาตรฐานนั้นไม่สนับสนุนการทำงานกับข้อความโซปที่มีการเข้ารหัส ทำให้กระบวนการบีเพลไม่สามารถเข้าใจข้อมูลที่อยู่ในข้อความโซปในกรณีที่ข้อความนั้นถูกเข้ารหัสไว้ได้เลย ปัญหาอีกข้อหนึ่งก็คือการเข้าใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยจะเกี่ยวข้องกับเซอร์วิสภายนอกต่างๆเป็นจำนวนมากทำให้ถ้าเราต้องการเรียกใช้กริดเซอร์วิสเหล่านี้โดยตรงจากกระบวนการบีเพลจะทำให้กระบวนการบีเพลที่ได้มีความสลับซับซ้อนและเข้าใจได้ยาก ซึ่งจะเป็นการขัดแย้งกับแนวคิดของบีเพลประการหนึ่งที่ต้องการให้กระบวนการบีเพลทำงานเฉพาะในส่วนที่เกี่ยวข้องกับกระบวนการทำงาน (Business Logic) เท่านั้น

3.3 ส่วนประกอบต่าง ๆของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส

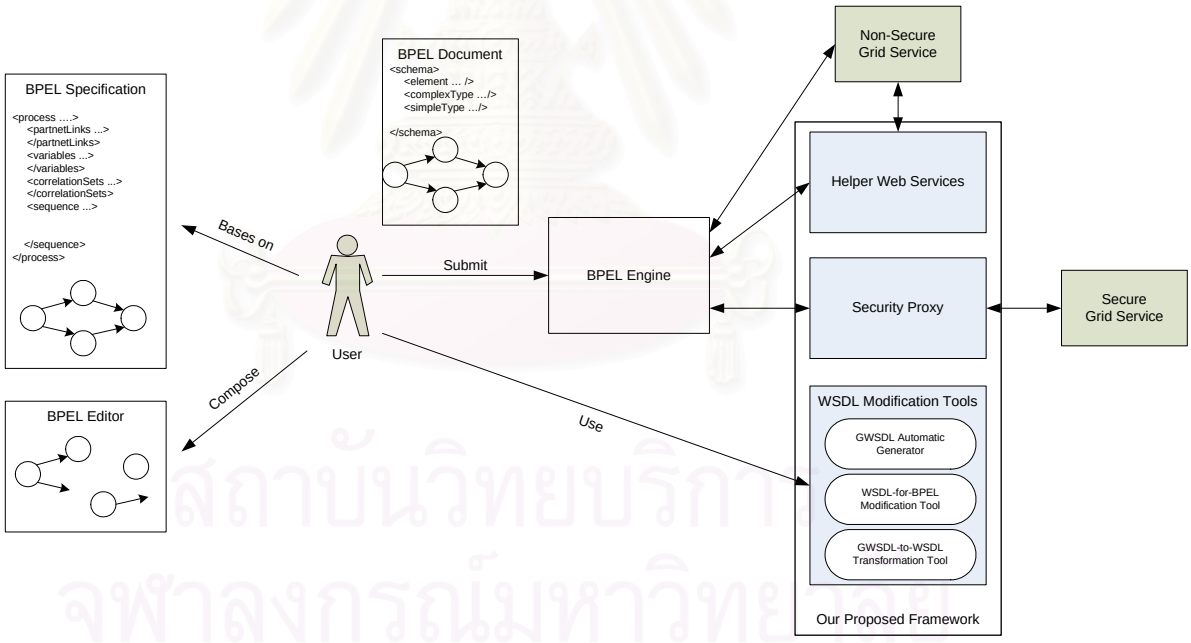
จากแนวคิดชั้นประสานกริดเซอร์วิส (Grid Services Convergence Layer) และปัญหาต่างๆที่พบในการใช้บีเพลประสานการทำงานของกริดเซอร์วิส วิทยานิพนธ์นี้จึงนำเสนอกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส (BPEL-Orchestrated Framework for Grid Services) ขึ้นเพื่อแก้ปัญหาเหล่านี้ โดบกรอบการทำงาน (Framework) ที่นำเสนอจะประกอบไปด้วยส่วนสำคัญ 3 ส่วน ได้แก่

1. *เว็บเซอร์วิสผู้ช่วย (Helper Web Services)* สร้างขึ้นแก้ปัญหาการใช้งานกริดเซอร์วิสอินสแตนท์และปัญหาการอ้างอิงอินสแตนท์ที่ต่างกันที่เกิดขึ้นในการเรียกใช้กลไกการแจ้งข่าว (Notification mechanisms) ทำให้กระบวนการบีเพล

สามารถรับข้อมูลจากกริดเซอร์วิสที่ทำหน้าที่เป็นแหล่งต้นทางการแจ้งข่าวได้ (Notification Source)

2. *เครื่องมือปรับปรุงดับบลิวเอสดีแอล (WSDL Modification Tools)* ออกแบบสำหรับการสร้างและปรับปรุงดับบลิวเอสดีแอลของกริดเซอร์วิสให้เหมาะสมกับการใช้งานกับกระบวนการบีเพล
3. *ตัวแทนความปลอดภัย (Security Proxy)* สร้างขึ้นเพื่อให้กระบวนการบีเพลสามารถเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยในรูปแบบต่างๆได้ โดยทำหน้าที่เป็นเสมือนตัวกลางในการติดต่อระหว่างกระบวนการบีเพลกับกริดเซอร์วิสที่มีการรักษาความปลอดภัย ทำให้กระบวนการบีเพลสามารถเรียกใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสได้อย่างครบถ้วย

เราสามารถแสดงภาพรวมของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส ได้ดังรูปที่ 5



รูปที่ 5 ภาพรวมของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส

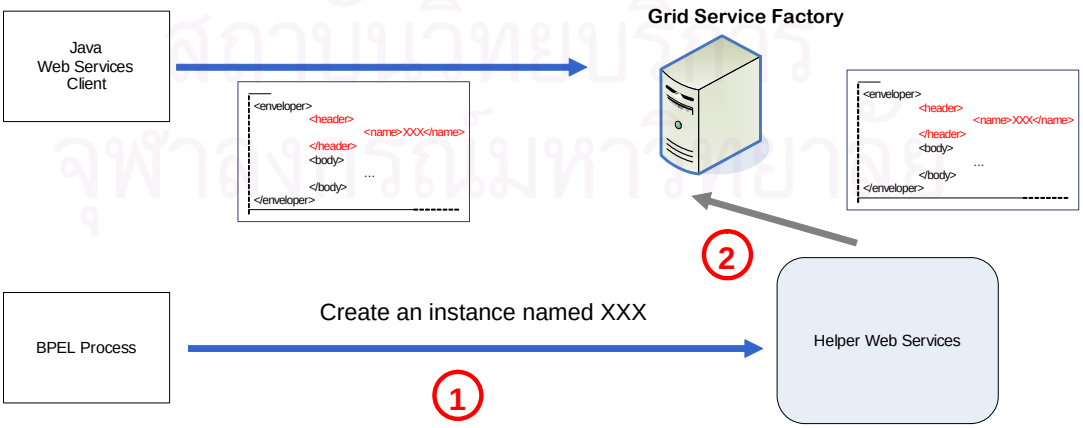
3.3.1 เว็บเซอร์วิสผู้ช่วย (Helper Web Services)

จุดมุ่งหมายของการสร้างเว็บเซอร์วิสผู้ช่วยก็เพื่อแก้ปัญหาข้อจำกัดของบีเพลที่ไม่อนุญาตให้มีการกำหนดการเชื่อมต่อแบบพลวัฒน์ (Dynamic Binding) ซึ่งทำให้บีเพลไม่สามารถ

เรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่จะถูกสร้างในขณะที่กระบวนการบีเพลทำงานอยู่ได้เลย หรือก็เพื่อตอบสนองการใช้งานกลไกโรงงานเซอร์วิสนั่นเอง นอกจากนั้นเว็บเซอร์วิสผู้ช่วยนี้ยังสามารถตอบสนองการใช้งานกลไกการแจ้งข่าวของกริดเซอร์วิสจากกระบวนการบีเพลได้โดยทำหน้าที่เป็นตัวกลางการติดต่อระหว่างกระบวนการบีเพลกับกริดเซอร์วิสที่มีรูปแบบการอ้างอิงอินสแตนท์ที่แตกต่างกันด้วย จากที่กล่าวมาเว็บเซอร์วิสผู้ช่วยที่สร้างขึ้นจึงประกอบไปด้วยโอเปอร์เรชั่นที่สำคัญ 2 โอเปอร์เรชั่น ได้แก่ โอเปอร์เรชั่น createGridServiceInstance และโอเปอร์เรชั่น subscribeToHelper ดังจะได้กล่าวต่อไป

3.3.1.1 โอเปอร์เรชั่น createGridServiceInstance

โอเปอร์เรชั่น createGridServiceInstance เป็นโอเปอร์เรชั่นที่ช่วยให้บีเพลสามารถสร้างกริดเซอร์วิสอินสแตนท์ใหม่โดยสามารถระบุชื่อของกริดเซอร์วิสอินสแตนท์ที่แน่นอนได้ การใช้งานโรงงานเซอร์วิสโดยตรงจากบีเพลจะไม่สามารถตั้งชื่อกริดเซอร์วิสอินสแตนท์ตามที่ต้องการได้ จึงมีความจำเป็นที่จะต้องอาศัยโอเปอร์เรชั่นนี้ในการตั้งชื่อกริดเซอร์วิสอินสแตนท์ตามที่ต้องการ ซึ่งจะเป็นกลไกหนึ่งที่จะทำให้เราสามารถทราบข้อมูลการเชื่อมต่อ (Binding Information) ได้แม้จะไม่มีกริดเซอร์วิสนั้นอยู่จริงในขณะนั้น และเมื่อเรานำชื่ออินสแตนท์ที่จะใช้งานจริงโดยผ่านโอเปอร์เรชั่นนี้ไปใช้ในการสร้างภาษาบรรยาย (Description Language) ล่วงหน้าโดยใช้เครื่องมือที่จะกล่าวในส่วนต่อไปนั้น จะช่วยให้บีเพลสามารถรองรับการเชื่อมต่อแบบพลวัฒน์ซึ่งเป็นการสร้างและใช้งานกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริงขณะตอนสร้างกระบวนการบีเพลได้ เราสามารถแสดงรูปอธิบายการทำงานของโอเปอร์เรชั่นได้ดังรูปที่ 6

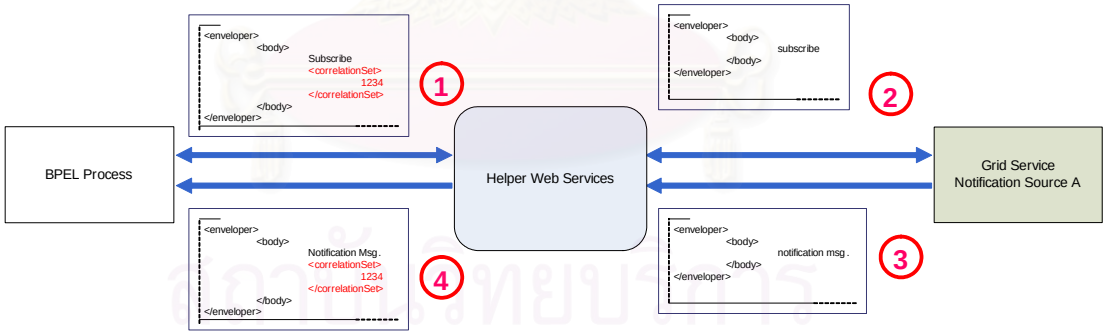


รูปที่ 6 การทำงานของโอเปอร์เรชั่น createGridServiceInstance ของเว็บเซอร์วิสผู้ช่วย

จากรูปที่ 6 เราสามารถเปรียบเทียบการใช้งานสร้างกริดเซอร์วิสอินสแตนท์ใหม่จากไคลเอนท์ที่เขียนด้วยภาษาจาวากับเว็บเซอร์วิสผู้ช่วย กล่าวคือ ไคลเอนท์ที่เขียนด้วยภาษาจาวานั้น สามารถตั้งชื่อของกริดเซอร์วิสอินสแตนท์ที่จะสร้างขึ้นใหม่ได้โดยตรง โดยแทรกมาชื่อมาในส่วนหัวของข้อความโซป แต่ในกรณีของกระบวนการบีเพลไม่สามารถทำได้จึงต้องอาศัยโอเปอร์เรชั่นนี้ของเว็บเซอร์วิสผู้ช่วย โดยเรียกใช้โอเปอร์เรชั่นนี้ จากนั้นโอเปอร์เรชั่นนี้จึงทำการติดต่อกับโรงงานกริดเซอร์วิสต่อไป ตามลำดับ

3.3.1.2 โอเปอร์เรชั่น subscribeToHelper

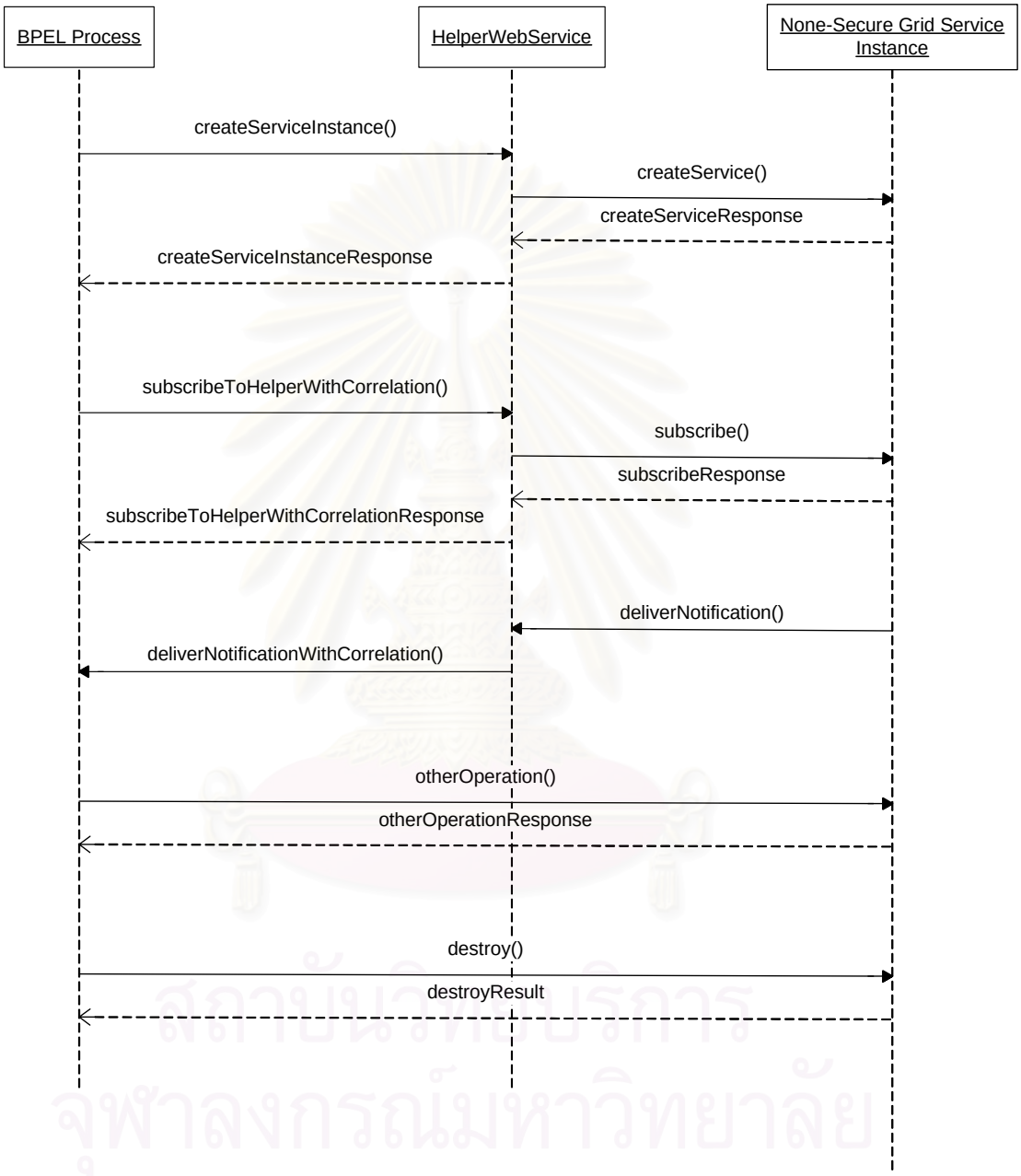
โอเปอร์เรชั่น subscribeToHelper สร้างขึ้นเพื่อให้กริดเซอร์วิสที่เกี่ยวข้องสามารถส่งการแจ้งข่าวกลับไปยังกระบวนการบีเพลที่ได้ทำการลงทะเบียนบอกรับการแจ้งข่าวไว้ การเรียกใช้โอเปอร์เรชั่นนี้จะต้องระบุจีเอสเอชของกริดเซอร์วิสเป้าหมาย ข้อมูลเซอร์วิสที่ต้องการจะบอกรับและจุดอ้างอิงปลายทาง (Endpoint Reference) ของกระบวนการบีเพลที่ทำหน้าที่รับการแจ้งข่าว และจะต้องระบุค่าของตัวระบุความสัมพันธ์ (Correlation Set) ของกระบวนการบีเพลเพื่อให้เว็บเซอร์วิสผู้ช่วยสามารถส่งข่าวที่ได้รับกลับไปยังกระบวนการบีเพลที่ถูกต้องได้ เราสามารถแสดงการทำงานของโอเปอร์เรชั่นนี้ได้ดังรูปที่ 7



รูปที่ 7 การทำงานของโอเปอร์เรชั่น subscribeToHelper

จากรูปที่ 7 เมื่อผู้ใช้ทำการเรียกใช้โอเปอร์เรชั่นนี้ (หมายเลข 1) โอเปอร์เรชั่นนี้จะทำหน้าที่ไปลงทะเบียนบอกรับกับกริดเซอร์วิสเป้าหมายจริงๆ (หมายเลข 2) จากนั้นเมื่อมีการส่งการแจ้งข่าวกลับมายังเว็บเซอร์วิสผู้ช่วย (หมายเลข 3) โอเปอร์เรชั่นนี้ก็จะทำการส่งข่าวที่ได้รับกลับไปยังกระบวนการบีเพล โดยทำการแทรกค่าคอร์รีเลชั่นเซ็ทที่ผู้ใช้กรอกเข้ามาตอนเรียกใช้งานโอเปอร์เรชั่น (หมายเลข 4) เพื่อจะช่วยให้ข่าวที่ได้รับถูกส่งกลับไปยังอินสแตนท์ของกระบวนการบีเพลที่ถูกต้อง

เราสามารถแสดงซีเควนซ์ไดอะแกรมของการใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยจากกระบวนการบีเพลซึ่งมีการใช้งานเว็บเซอร์วิสผู้ช่วยได้ดังรูปที่ 8

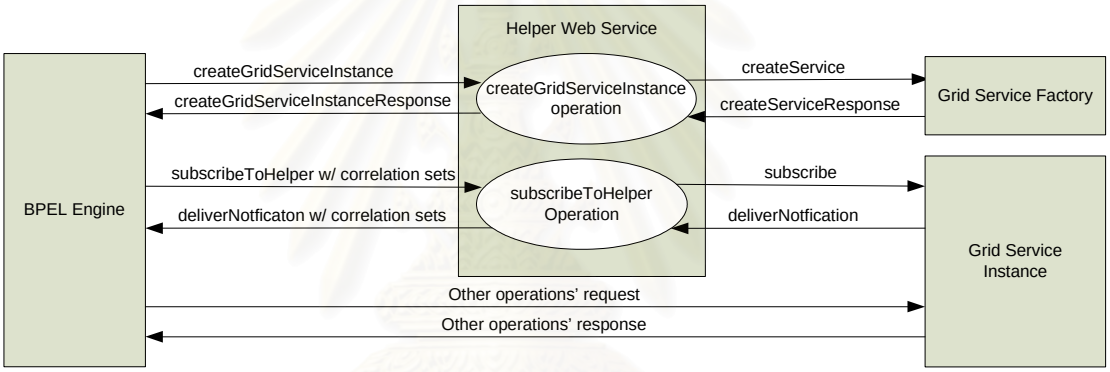


รูปที่ 8 ซีเควนซ์ไดอะแกรมของการเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยจากกระบวนการบีเพล

จากรูปที่ 8 จะพบว่า ถ้าเป็นการเรียกใช้งานกลไกและคุณสมบัติอื่นๆของกริดเซอร์วิสอินสแตนท์ รวมไปถึงโอเปอร์เรชั่นของกริดเซอร์วิสที่ผู้ใช้กำหนดขึ้นเอง สามารถถูกเรียกใช้งานได้โดยตรงจากกระบวนการบีเพลโดยไม่ต้องผ่านตัวกลางใดๆ

ในการพัฒนาโอเปอร์เรชั่นนี้เพื่อการใช้งานจริงนั้น เราได้ทำการแบ่งออกเป็น 2 โอเปอร์เรชั่นย่อย คือ subscribeToHelper และ subscribeToHelperByName โดยสำหรับโอเปอร์เรชั่น subscribeToHelper ผู้ใช้สามารถทำการระบุถึงข้อมูลเซอร์วิสที่ต้องการบอกรับในรูปเอ็กซ์เอ็มแอลแบบใดก็ได้ ซึ่งเป็นไปตามข้อกำหนดโอจีเอสไอ ส่วนโอเปอร์เรชั่น subscribeToHelperByName ผู้ใช้เพียงแต่ระบุชื่อของข้อมูลเซอร์วิสเพียงอย่างเดียว เว็บเซอร์วิสผู้ช่วยจะทำการสร้างเอ็กซ์เอ็มแอลที่เหมาะสมสำหรับการบอกรับการแจ้งข่าวกับกริดเซอร์วิสเป้าหมายจริงๆโดยอัตโนมัติ

จากโอเปอร์เรชั่นทั้ง 2 ที่กล่าวมาแล้ว เราสามารถแสดงส่วนประกอบต่างๆ เมื่อมีการใช้งานเว็บเซอร์วิสผู้ช่วยร่วมกับการใช้งานกริดเซอร์วิสโดยตรงจากกระบวนการบีเพล ได้ดังรูปที่ 9



รูปที่ 9 การใช้งานเว็บเซอร์วิสผู้ช่วยร่วมกับการเรียกใช้งานกริดเซอร์วิสโดยตรงจากกระบวนการบีเพล

ข้อสำคัญในการใช้งานเว็บเซอร์วิสผู้ช่วยก็คือ เว็บเซอร์วิสผู้ช่วยนี้ทำหน้าที่ตอบสนองการใช้งานกลไกโรงงานเซอร์วิสและกลไกการแจ้งข่าว สำหรับกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยเท่านั้น เนื่องมาจากในการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น การเรียกใช้จะทำผ่านตัวแทนความปลอดภัยที่จะได้กล่าวต่อไปทั้งหมด ทำให้ความสามารถของเว็บเซอร์วิสผู้ช่วยนี้จะถูกรวมอยู่ในตัวแทนความปลอดภัยทั้งหมดแล้ว

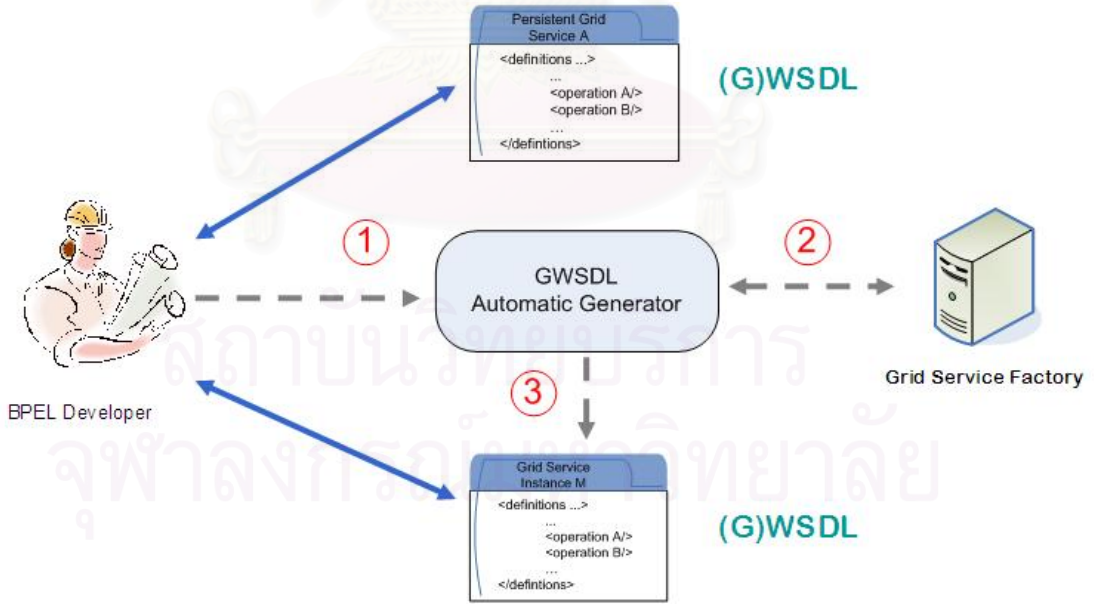
3.3.2 เครื่องมือปรับปรุงดับบลิวเอสดีแอล (WSDL Modification Tools)

เครื่องมือปรับปรุงดับบลิวเอสดีแอลถูกสร้างขึ้นเพื่อแก้ปัญหาการใช้งานดับบลิวเอสดีแอลในขั้นตอนการสร้างกระบวนการบีเพล เช่น บีเพลไม่รองรับคุณสมบัติการสืบทอดพอร์ทไทป์ หรือผู้ใช้จำเป็นจะต้องทราบดับบลิวเอสดีแอลของเซอร์วิสที่จะเรียกใช้ตั้งแต่ขั้นตอนการสร้าง

เอกสารบีเพล ซึ่งจะต่างจากการเรียกใช้กริดเซอร์วิสอินสแตนท์ ที่จะทำการสร้างกริดเซอร์วิสอินสแตนท์ผ่านทางกลไกโรงงานเซอร์วิส ในขณะที่กระบวนการกำลังทำงานอยู่ เป็นต้น โดยเครื่องมือเหล่านี้จะแบ่งออกเป็น 3 ส่วน คือ

3.3.2.1 เครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ (Automatic GWSDL Generator)

เครื่องมือนี้ทำหน้าที่สร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริงขึ้นมา เพื่อให้ผู้ใช้สามารถนำจีดับบลิวเอสดีแอลที่ได้ไปใช้ในการสร้างกระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริงได้ โดยการเรียกใช้เครื่องมือนี้ผู้ใช้จะต้องระบุจีเอสเอชของกริดเซอร์วิสที่ทำหน้าที่เป็นโรงงานเซอร์วิส และชื่อของกริดเซอร์วิสอินสแตนท์ที่ต้องการการใช้งานเครื่องมือนี้ควบคู่กับโอเปอร์เรชั่นสำหรับสร้างกริดเซอร์วิสอินสแตนท์ใหม่ของเว็บเซอร์วิสผู้ช่วย จะทำให้เราสามารถสร้างกระบวนการบีเพลที่มีการสร้างกริดเซอร์วิสอินสแตนท์และเรียกใช้อินสแตนท์นั้นได้ภายในกระบวนการบีเพลเดียวกัน เราสามารถแสดงการทำงานของเครื่องมือนี้ได้ดังรูปที่ 10



รูปที่ 10 การทำงานของเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ

จากรูปที่ 10 เมื่อผู้พัฒนากระบวนการบีเพล (BPEL Developer) ต้องการสร้างเอกสารกระบวนการบีเพล ในกรณีที่เป็นการเรียกใช้งานกริดเซอร์วิสแบบถาวร (Persistent Grid

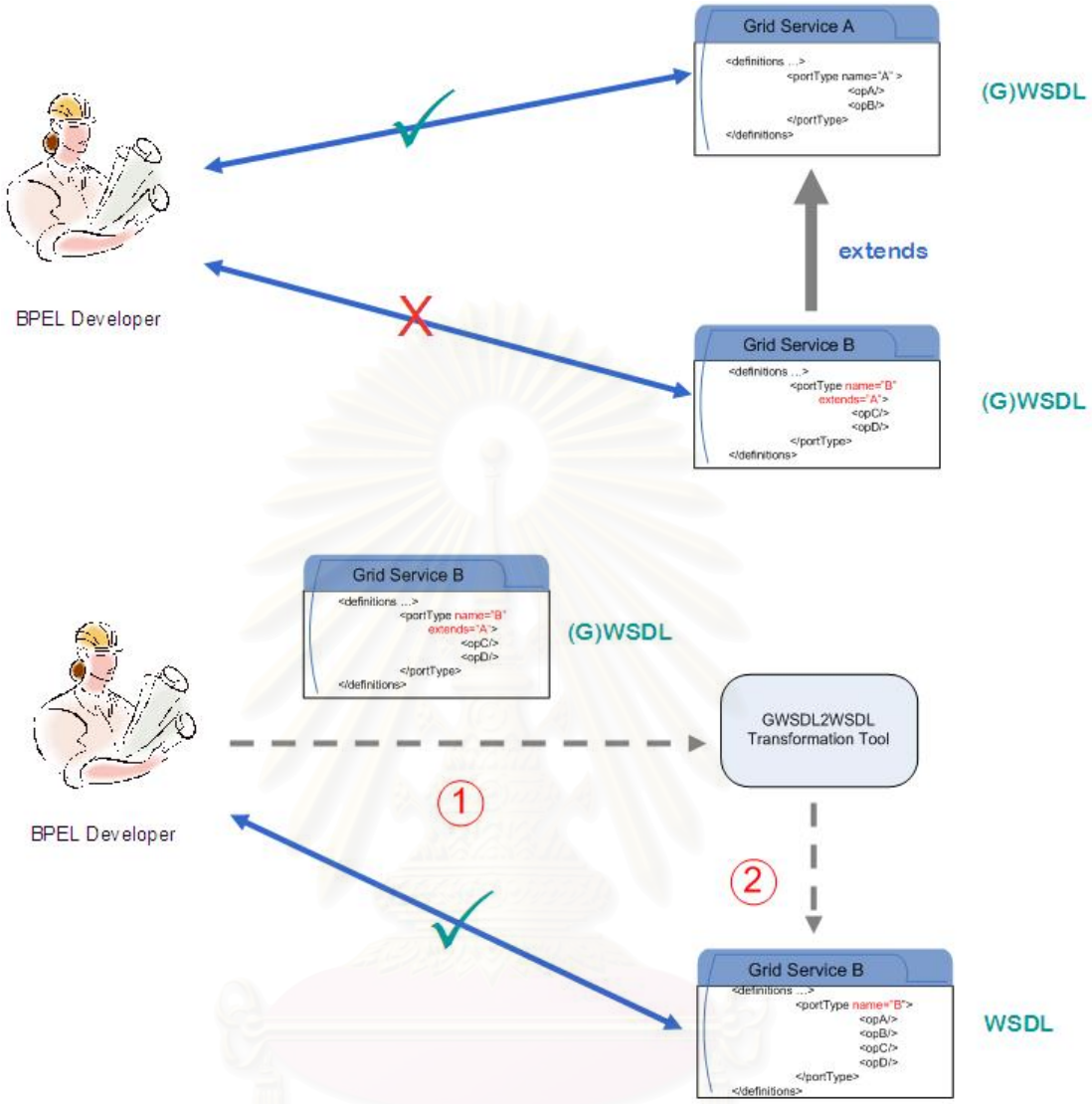
Service) ซึ่งเป็นกริดเซอร์วิสอินสแตนท์ที่มีอยู่จริงแล้วนั้น ผู้พัฒนาสามารถเรียกดูจีดับบลิวเอสดีแอลของกริดเซอร์วิสแบบถาวรนั้นเพื่อใช้ในการสร้างเอกสารกระบวนการบีเพลได้ทันที ตัวอย่างเช่น จีดับบลิวเอสดีแอลของกริดเซอร์วิสแบบถาวรเอ (Persistent Grid Service A) เป็นต้น แต่ในกรณีที่ผู้พัฒนาต้องการเรียกใช้กริดเซอร์วิสที่ยังไม่มีอยู่จริงในขณะนั้น ผู้พัฒนาจะต้องสร้างดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์นั้นโดยใช้เครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ (ขั้นตอนที่ 1) ซึ่งเครื่องมือนี้จะทำหน้าที่ในการติดต่อกับโรงงานกริดเซอร์วิส (ขั้นตอนที่ 2) จากนั้นเครื่องมือนี้จะสร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์ตามที่ผู้พัฒนาต้องการ (ขั้นตอนที่ 3) ผู้พัฒนาก็จะสามารถนำจีดับบลิวเอสดีแอลที่ได้นั้นไปใช้ในการสร้างเอกสารกระบวนการบีเพลได้ต่อไป

งานวิจัยนี้ได้สร้างคลาสที่ชื่อ GWSDLAutomaticGenerator ที่อยู่ในแพ็กเกจ com.zeus.ws.wsdl.modification.generator.automatic ทำหน้าที่เป็นเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ

3.3.2.2 เครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอล (GWSDL-to-WSDL Transformation Tool)

เครื่องมือนี้จะทำการแปลงจีดับบลิวเอสดีแอลที่มีการใช้คุณสมบัติการสืบทอดพอร์ทไทป์ ให้กลับเป็นดับบลิวเอสดีแอลมาตรฐานที่มีการแสดงทุกโอเปอร์เรชั่นอย่างชัดเจน (Explicit) การใช้เครื่องมือนี้จะทำให้เครื่องมือบีเพลสามารถรับรู้และเข้าใจกริดเซอร์วิสที่ถูกบรรยายในรูปของจีดับบลิวเอสดีแอลเพื่อใช้ในการสร้างกระบวนการบีเพลต่อไป เราสามารถแสดงการทำงานของเครื่องมือนี้ได้ดังรูปที่ 11

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 11 การทำงานของเครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอล

จากรูปที่ 11 รูปด้านบนแสดงให้เห็นว่าในกรณีที่ผู้พัฒนากระบวนการบีเพล (BPEL Developer) ต้องการใช้งานกริดเซอร์วิส โดยกริดเซอร์วิสนั้นไม่มีการขยายหรือสืบทอดพอร์ทไทป์มาจากกริดเซอร์วิสอื่น ตัวอย่างเช่น กริดเซอร์วิสเอ (Grid Service A) ผู้พัฒนาสามารถนำจีดับบลิวเอสดีแอลของกริดเซอร์วิสนั้นไปใช้ในการพัฒนาได้ทันที (แสดงด้วยเครื่องหมายถูก) แต่ถ้ากริดเซอร์วิสที่ต้องการเรียกใช้มีการขยายหรือสืบทอดพอร์ทไทป์มาจากกริดเซอร์วิส ตัวอย่างเช่น กริดเซอร์วิสบี (Grid Service B) ขยายมาจากกริดเซอร์วิสเอ ผู้พัฒนาจะไม่สามารถนำจีดับบลิวเอสดีแอลของกริดเซอร์วิสนั้นไปใช้ได้ทันที (แสดงด้วยเครื่องหมายผิด) ดังนั้นผู้พัฒนาจะต้องใช้เครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอลดังในรูปด้านล่าง เพื่อทำการแปลงจีดับบลิวเอสดีแอลนั้นให้เป็นดับบลิวเอสดีแอลมาตรฐาน จากนั้นเครื่องมือนี้ก็จะทำการแปลงจี

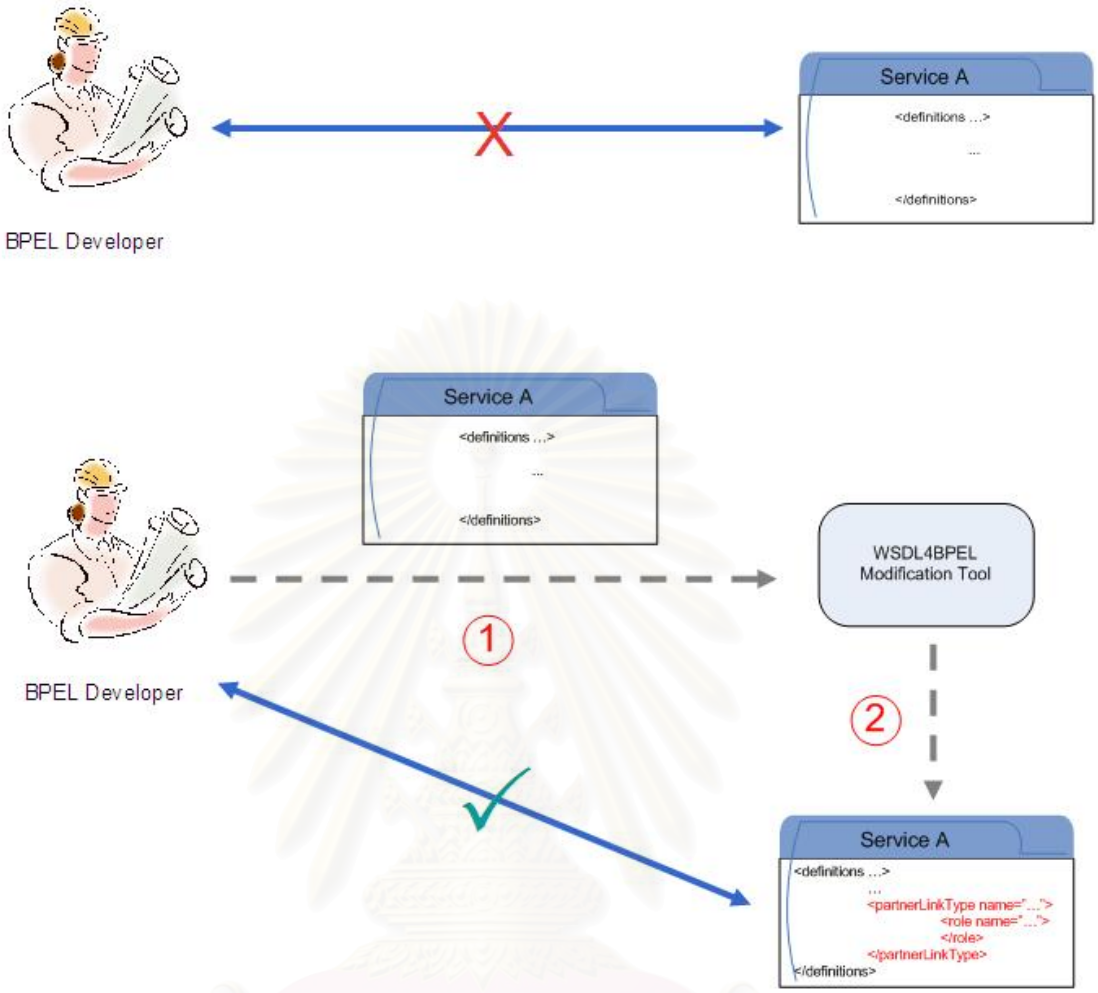
ดับบลิวเอสดีแอลที่ได้รับให้เป็นดับบลิวเอสดีแอลมาตรฐาน เพื่อให้ผู้พัฒนาสามารถนำดับบลิวเอสดีแอลของกริดเซอร์วิสบีที่ได้ใหม่นี้ ไปใช้ในการสร้างเอกสารกระบวนการบีเพลได้ต่อไป (แสดงด้วยเครื่องหมายถูก)

งานวิจัยนี้ได้สร้างคลาสที่ชื่อ GWSDL2WSDL ที่อยู่ในแพ็กเกจ com.zeus.ws.wsdl.modification.gwsdl2wsdl สำหรับทำหน้าที่เป็นเครื่องมือในการแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอล

3.3.2.3 เครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานกับบีเพล (WSDL-for-BPEL modification Tool)

เนื่องจากเว็บเซอร์วิสและกริดเซอร์วิสที่จะนำมาใช้งานกับบีเพลนั้น จะต้องมีการระบุความสัมพันธ์ในการเรียกใช้งานเซอร์วิสในรูปของประเภทความสัมพันธ์พาร์ทเนอร์ (PartnerLinkType) ไว้ภายในดับบลิวเอสดีแอลอย่างชัดแจ้ง ดังนั้น เราจะต้องมีเครื่องมือในการสร้างการบรรยายประเภทความสัมพันธ์พาร์ทเนอร์ลงในดับบลิวเอสดีแอลของเซอร์วิสที่ไม่มีการระบุไว้ให้โดยอัตโนมัติ เพื่อให้ผู้ใช้สามารถนำดับบลิวเอสดีแอลไปใช้ในกับเครื่องมือบีเพลต่างๆได้อย่างสะดวก โดยเครื่องมือนี้จะทำการสร้างประเภทความสัมพันธ์พาร์ทเนอร์ให้อัตโนมัติตามจำนวนพอร์ทไทป์ที่มีอยู่ดับบลิวเอสดีแอลที่ต้องการปรับปรุง เราสามารถแสดงการทำงานของเครื่องมือนี้ได้รูปที่ 12

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 12 การทำงานของเครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานกับบีเพล

จากรูปที่ 12 รูปด้านบนแสดงให้เห็นว่าเมื่อผู้พัฒนากระบวนการบีเพลต้องการนำดับบลิวเอสดีแอลของกริดเซอร์วิสหรือเว็บเซอร์วิสแต่ละตัวไปใช้ในการสร้างเอกสารนั้น ถ้าดับบลิวเอสดีแอลนั้นยังไม่มีการระบุประเภทความสัมพันธ์พาร์ทเนอร์ ตัวอย่างเช่น ดับบลิวเอสดีแอลของเซอร์วิสเอ (Service A) เป็นต้น เราจะไม่สามารถนำดับบลิวเอสดีแอลนั้นไปใช้ในการสร้างเอกสารกระบวนการบีเพลได้ (แสดงด้วยเครื่องหมายผิด) ดังนั้นผู้พัฒนาจะต้องใช้เครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานบีเพลในการปรับปรุงดับบลิวเอสดีแอลดังในรูปด้านล่าง ซึ่งเครื่องมือนี้จะทำการปรับปรุงดับบลิวเอสดีแอลโดยการแทรกการบรรยายประเภทความสัมพันธ์พาร์ทเนอร์ลงไป ผู้พัฒนาก็จะสามารถนำดับบลิวเอสดีแอลที่ได้ไปใช้ในการสร้างเอกสารบีเพลได้ต่อไป (แสดงด้วยเครื่องมือหมายถูก) งานวิจัยนี้ได้สร้างคลาสที่ชื่อ WSDL4BPEL ที่อยู่ในแพ็กเกจ com.zeus.ws.wsdl.modification.wsdl4bpel ทำหน้าที่เป็นเครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับการใช้งานกับบีเพล

จากที่กล่าวมาทั้งหมดเกี่ยวกับเครื่องมือปรับปรุงดับบลิวเอสดีแอลนั้น มีข้อสำคัญที่ควรคำนึงถึง คือ การใช้งานเครื่องมือปรับปรุงดับบลิวเอสดีแอลนั้น มีความจำเป็นเฉพาะในกรณีที่มีการใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยเท่านั้น เนื่องมาจากในกรณีที่มีการใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น จากที่กล่าวมาแล้วว่า จะมีการทำงานผ่านตัวแทนความปลอดภัยทั้งหมด ซึ่งตัวแทนความปลอดภัยนี้จะสร้างขึ้นในรูปเว็บเซอร์วิสอยู่แล้ว ทำให้ภาษาบรรยาย (Description Language) ของตัวแทนความปลอดภัยเหล่านี้ อยู่ในรูปของดับบลิวเอสดีแอลที่เป็นมาตรฐานอยู่แล้ว นอกจากนั้นในกรณีที่เป็นการใช้งานกริดเซอร์วิสถาวร (Persistent Grid Service) เราไม่มีความจำเป็นจะต้องใช้เครื่องมือสร้างจีดับบลิวเอสดีแอลอัตโนมัติ เนื่องจากกริดเซอร์วิสเหล่านี้จะมีจีดับบลิวเอสดีแอลอยู่แล้ว

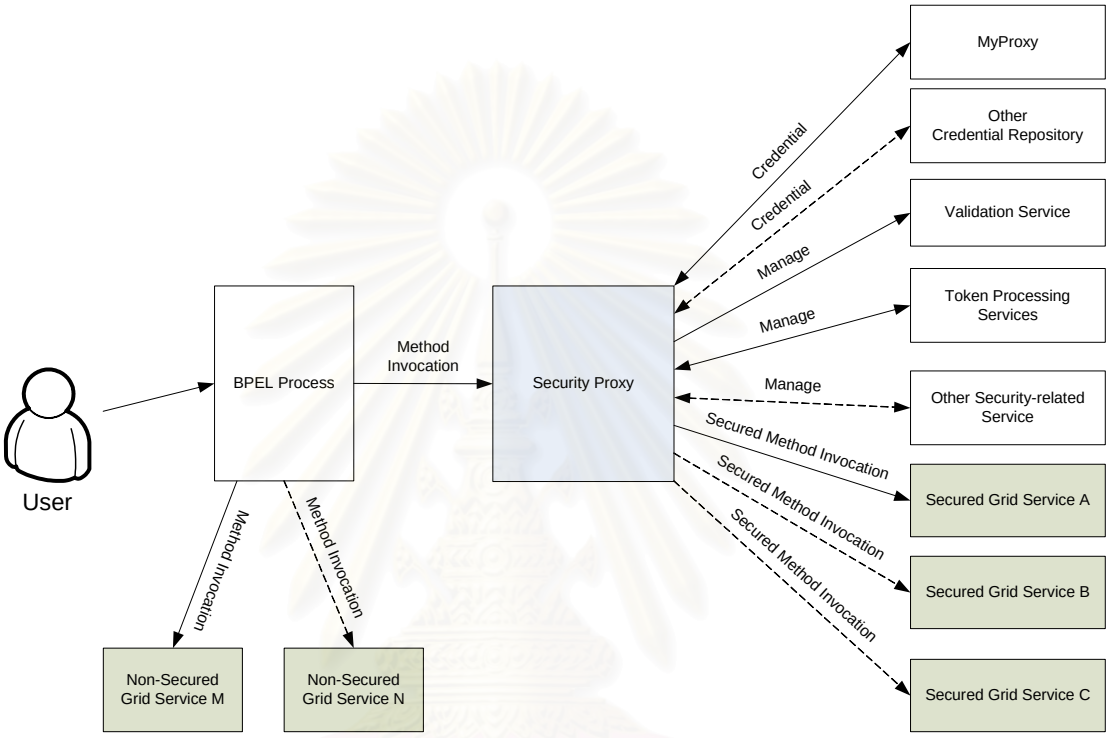
3.3.3 ตัวแทนความปลอดภัย (Security Proxy)

ในการแก้ปัญหาข้อจำกัดของบีเพลที่ไม่อนุญาตให้มีการจัดการกับส่วนหัวของข้อความโซปได้นั้น จำเป็นจะต้องอาศัยตัวกลางในการติดต่อระหว่างกระบวนการบีเพลและกริดเซอร์วิส โดยตัวกลางการติดต่อนี้จะทำหน้าที่ในการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยแทนกระบวนการบีเพล ทำให้กระบวนการบีเพลไม่ต้องจัดการกับโทเคนความปลอดภัยด้วยตัวเอง พร้อมทั้งติดต่อกับเซอร์วิสต่างๆที่เกี่ยวข้องในการเรียกใช้กริดเซอร์วิสที่มีการรักษาความปลอดภัย นอกจากนั้นตัวกลางเหล่านี้ยังทำหน้าที่ในการเข้าและถอดรหัสข้อมูลที่อยู่ในข้อความโซปในกรณีที่มีการเข้ารหัสไว้ด้วย ซึ่งจะทำให้บีเพลสามารถรับรู้ข้อมูลที่ส่งกลับมาจากกริดเซอร์วิสในกรณีที่ข้อความนั้นถูกเข้ารหัส ดังนั้นข้อความที่ส่งระหว่างกระบวนการบีเพลกับตัวกลางเหล่านี้จะเป็นข้อความเอ็กซ์เอ็มแอลธรรมดาที่กระบวนการบีเพลสามารถเข้าใจได้ ด้วยแนวคิดการใช้ตัวกลางนี้ เราจะนำเสนอการใช้งานตัวแทนความปลอดภัย (Security Proxy) เพื่อทำงานเป็นตัวกลางตามที่ได้กล่าวมาแล้ว

ในการเรียกใช้งานกริดเซอร์วิสที่มีความปลอดภัย (Secured Grid Services) ที่สร้างบนพื้นฐานของจีเอสไอนั้น ไคลเอนท์ที่ทำการติดต่อกับกริดเซอร์วิสนั้นจำเป็นจะต้องมีเครเดนเชี่ยล (Credential) ทำให้ตัวแทนความปลอดภัยจำเป็นจะต้องมีเครเดนเชี่ยล (Credential) เพื่อใช้ในการติดต่อกับกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น ดังนั้นเราจึงออกแบบตัวแทนความปลอดภัยให้ผู้ใช้สามารถใช้เครเดนเชี่ยลจากแหล่งเก็บเครเดนเชี่ยล (Credential Repository) แบบใดก็ได้มาใช้งานเป็นเครเดนเชี่ยลของตัวแทนคลวามปลอดภัย ซึ่งแหล่งเก็บเครเดนเชี่ยลที่รู้จักกันดี ก็

คือ มายพร็อกซี่ (MyProxy) [27] หลังจากนั้นตัวแทนความปลอดภัยก็จะสามารถทำงานภายใต้สิทธิที่ได้รับจากเครเดนเชี่ยลนั้นได้ต่อไป

เราสามารถแสดงภาพรวมเมื่อมีการใช้งานตัวแทนความปลอดภัยได้ดังรูปที่ 13

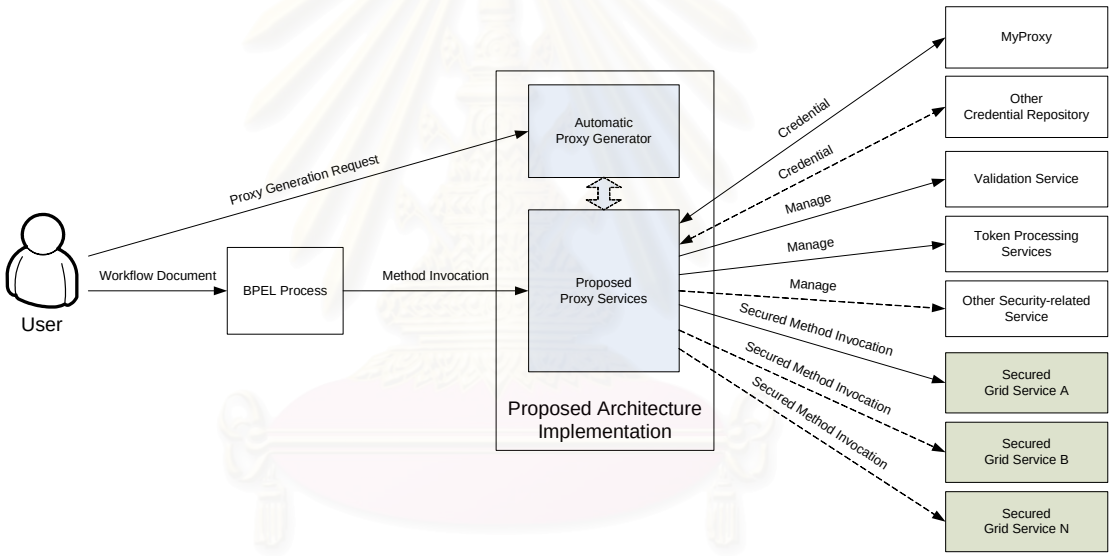


รูปที่ 13 การใช้งานตัวแทนความปลอดภัยในการเรียกใช้กริดเซอร์วิสจากกระบวนการบีเพล

จากรูปที่ 13 เราจะพบว่าการใช้งานตัวแทนความปลอดภัยจะใช้เมื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยไว้เท่านั้น ส่วนกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยจะสามารถเรียกใช้งานได้โดยตรงจากกระบวนการบีเพลโดยอาศัยความช่วยเหลือจากเว็บเซอร์วิสผู้ช่วย สำหรับตัวแทนความปลอดภัยที่เราสร้างขึ้นนั้น จะต้องทำหน้าที่ในการสร้างตัวแปรกริดเซอร์วิสใหม่และบอกรับการแจ้งข่าวในกรณีที่กริดเซอร์วิสที่เป็นแหล่งต้นทางการแจ้งข่าวมีการรักษาความปลอดภัยด้วย เพื่อให้สามารถทำหน้าที่แทนเว็บเซอร์วิสผู้ช่วยได้ ทำให้ในกรณีที่เป็นการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย เราจะทำงานผ่านตัวแทนความปลอดภัยทั้งหมด โดยไม่ต้องอาศัยเว็บเซอร์วิสผู้ช่วยอีกต่อไป แต่ผู้ใช้ก็ยังจำเป็นจะต้องใช้งานรูปแบบการเขียนบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสและเครื่องมือปรับปรุงดับบลิวเอสดีแอลประกอบในการสร้างเอกสารบีเพล นอกจากนี้ ตัวแทนความปลอดภัยยังต้องมีการติดต่อกับเซอร์วิสอื่นที่ทำหน้าที่ในการจัดการความปลอดภัย เช่น เซอร์วิสที่ทำหน้าที่แลกเปลี่ยนโทเคนความปลอดภัย เป็นต้น

ดังนั้นการใช้งานตัวแทนความปลอดภัย ยังช่วยลดความยุ่งยากในการสร้างกระบวนการบีเพล โดยทำหน้าที่ติดต่อกับเซอร์วิสเหล่านี้ให้เองโดยอัตโนมัติ โดยที่ผู้ใช้ไม่ต้องรู้รายละเอียดของการทำงานใดๆทั้งสิ้น

ในการนำตัวแทนความปลอดภัยมาใช้งานจริงนั้น เราจำเป็นจะต้องสร้างตัวแทนความปลอดภัยสำหรับกริดเซอร์วิสที่มีการรักษาความปลอดภัยต่างๆ ในลักษณะ 1 ต่อ 1 มี ซึ่งทำหน้าที่ในการ ดังนั้นในกรอบการทำงานของเราจะมีส่วนประกอบที่สำคัญขึ้นมา ก็คือ ตัวสร้างตัวแทนอัตโนมัติ (Automatic Proxy Generator) ซึ่งจะช่วยสร้างตัวแทนความปลอดภัยสำหรับกริดเซอร์วิสอินสแตนท์แต่ละตัว เรานำเสนอภาพรวมของการใช้งานตัวแทนความปลอดภัย ได้ตามรูปที่ 14



รูปที่ 14 ภาพรวมการใช้งานตัวแทนความปลอดภัยโดยใช้ตัวสร้างตัวแทนอัตโนมัติ

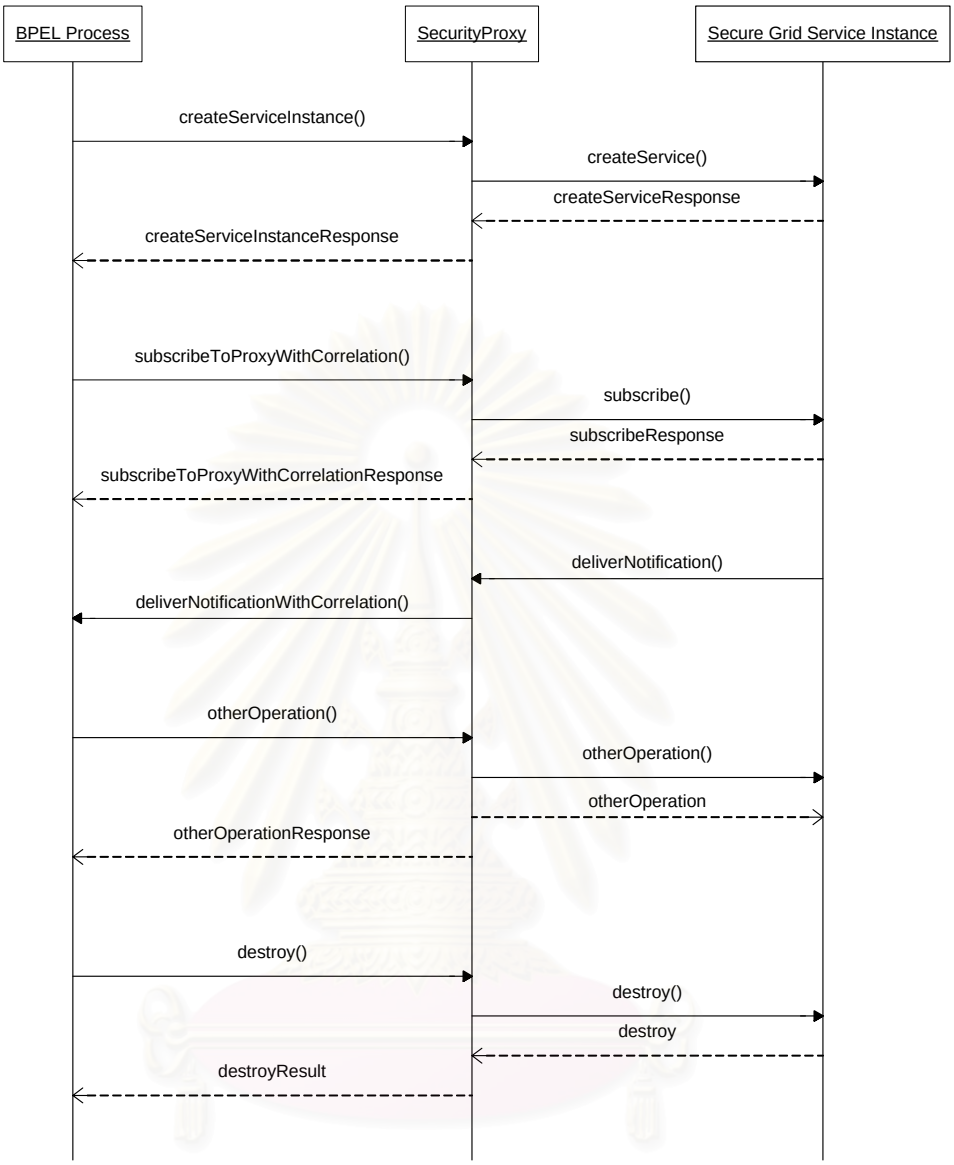
สำหรับตัวแทนความปลอดภัยที่สร้างขึ้นจากตัวสร้างตัวแทนอัตโนมัตินั้น จะประกอบไปด้วยโอเปอร์เรชั่นต่างๆ ดังต่อไปนี้

- โอเปอร์เรชั่นต่างๆของกริดเซอร์วิสเป้าหมายทั้งหมด ให้ผู้ใช้สามารถเรียกใช้งานโอเปอร์เรชั่นเหล่านี้ได้เหมือนกับการเรียกใช้โดยตรงจากกริดเซอร์วิสนั้น โดยไม่ต้องคำนึงถึงการรักษาความปลอดภัยใดๆทั้งสิ้น หลังจากนั้นตัวแทนความปลอดภัยจะทำหน้าที่ในการจัดการความปลอดภัยในการเรียกใช้โอเปอร์เรชั่นเหล่านั้นได้โดยอัตโนมัติ

- *createServiceInstance operation* ในกรณีที่เป็นการสร้างตัวแทนของกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง สาเหตุที่เราออกแบบให้มีโอเปอร์เรชั่นนี้ เนื่องมาจากเนื่องจากผู้ใช้สามารถสร้างกริดเซอร์วิสอินสแตนท์ในเวลาใดๆ ก็ได้ โดยไม่ต้องมีการสร้างกริดเซอร์วิสอินสแตนท์ไว้ล่วงหน้าตั้งแต่ขั้นตอนของการสร้างตัวแทนความปลอดภัย ซึ่งจะเป็นการช่วยลดภาระของเครื่องที่ให้บริการกริดเซอร์วิสได้ และตรงตามแนวคิดของการใช้งานกริดเซอร์วิสอินสแตนท์เป็นอย่างดี
- *subscribeToProxy operation และ subscribeToProxyByName operation* ซึ่งโอเปอร์เรชั่นทั้งสองนี้จะทำหน้าที่เหมือนกับโอเปอร์เรชั่นเดียวกันนี้ของเว็บเซอร์วิสผู้ช่วย

การที่เราออกแบบให้ตัวแทนความปลอดภัยมีโอเปอร์เรชั่นที่ทำหน้าที่แทนเว็บเซอร์วิสผู้ช่วยอยู่ภายในตัวมันเอง เนื่องมาจากการที่เราออกแบบให้ตัวแทนความปลอดภัยทำหน้าที่เป็นตัวแทนของกริดเซอร์วิสใดๆในลักษณะ 1 ต่อ 1 อยู่แล้ว ดังนั้นการให้ตัวแทนเหล่านี้ทำหน้าที่ในการสร้างกริดเซอร์วิสอินสแตนท์ใหม่และบอกรับการแจ้งข่าวในตัวมันเองด้วย จึงมีความเหมาะสม เราสามารถแสดงซีเควนซ์ไดอะแกรมของการใช้งานตัวแทนความปลอดภัยได้ดังรูปที่ 15

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 15 ซีเควนซ์ไดอะแกรมของการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยจากกระบวนการบีเพล

จากรูปที่ 15 จะพบว่าในการติดต่อกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น กระบวยการบีเพลจะเรียกใช้งานโอเปอร์เรชั่นต่างๆของกริดเซอร์วิสผ่านตัวแทนความปลอดภัยทั้งหมด โดยจะไม่มีการเรียกใช้งานโดยตรงไปยังกริดเซอร์วิสที่มีการรักษาความปลอดภัยเลย นอกจากนี้ในรูปไม่มีการแสดงการติดต่อระหว่างตัวแทนความปลอดภัยกับกริดเซอร์วิสที่เกี่ยวข้องในการรักษาความปลอดภัย ซึ่งตัวแทนความปลอดภัยจะต้องเรียกใช้ควบคู่ไปกับการเรียกใช้กริดเซอร์วิสที่มีการรักษาความปลอดภัย

งานวิจัยนี้ได้สร้างตัวสร้างตัวแทนอัตโนมัติออกเป็น 2 รูปแบบ คือ ตัวสร้างตัวแทนอัตโนมัติสำหรับกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่มีอยู่แล้ว คลาสที่ทำหน้าที่เป็นตัวสร้างตัวแทนอัตโนมัติในรูปแบบนี้ คือ คลาส ProxyGeneratorForPersistentGridService ที่อยู่ในแพ็กเกจ com.zeus.ws.proxy.generator.tools และ ตัวสร้างตัวแทนอัตโนมัติสำหรับกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง สำหรับคลาสที่ทำหน้าที่เป็นตัวสร้างตัวแทนอัตโนมัติแบบนี้ คือ คลาส ProxyGeneratorForAGridServiceInstance ที่อยู่ในแพ็กเกจ com.zeus.ws.proxy.generator.tools เช่นเดียวกัน

ในการใช้งานตัวสร้างตัวแทนอัตโนมัตินั้น ผู้ใช้จะต้องกำหนดค่าจีเอสเอชของกริดเซอร์วิสที่เราต้องการจะสร้างตัวแทนความปลอดภัยด้วย โดยในกรณีที่เราต้องการจะสร้างตัวแทนความปลอดภัยของกริดเซอร์วิสแบบถาวร ผู้ใช้จะต้องกำหนดค่าจีเอสเอชของกริดเซอร์วิสถาวรนั้น แต่ถ้าต้องการสร้างตัวแทนความปลอดภัยของกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง ในกรณีนี้ผู้ใช้จะต้องกำหนดค่าจีเอสเอชของกริดเซอร์วิสทีทำหน้าที่เป็นโรงงานเซอร์วิสของกริดเซอร์วิสอินสแตนท์ที่ต้องการด้วย

3.3.3.1 ไฟล์โครงแบบความปลอดภัย (Security Configuration File)

สิ่งสำคัญในการใช้งานตัวสร้างตัวแทนความปลอดภัยแบบอัตโนมัติอีกสิ่งหนึ่ง ก็คือผู้ใช้จำเป็นจะต้องมีหรือสร้างไฟล์โครงแบบความปลอดภัย (Security Configuration File) ให้แก่ตัวสร้างตัวแทนอัตโนมัติด้วย เพื่อให้ตัวแทนความปลอดภัยสามารถติดต่อกับโอเปอร์เรชั่นต่างๆของกริดเซอร์วิสเป้าหมาย โดยใช้วิธีการรักษาความปลอดภัยที่เหมาะสมตามที่โอเปอร์เรชั่นนั้นต้องการได้ และเป็นไปตามรูปแบบของโครงสร้างพื้นฐานความปลอดภัยกริดหรือจีเอสไอและตามข้อกำหนดดับบลิวเอส-ซีเคียวริตี้ได้อย่างเหมาะสม ซึ่งจากการศึกษาและทดลองสร้างไคลเอนท์ภาษาจาวาเพื่อติดต่อกับกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น จะพบว่ามีสิ่งที่ต้องคำนึงถึงในการสร้างไคลเอนท์เพื่อทำการติดต่อกับกริดเซอร์วิสที่มีการรักษาความปลอดภัยทั้งหมด 4 ประการ ได้แก่

- วิธีการยืนยันตัวตน (Authentication Method) ตามรูปแบบของจีเอสไอที่ใช้ในโกลบัสทูลคิทเวอร์ชั่น 3 ผู้พัฒนากริดเซอร์วิสสามารถกำหนดวิธีการยืนยันตัวตนของกริดเซอร์วิสได้เป็น 3 รูปแบบ คือ

- *บทสนทนาที่มีความปลอดภัย (Secured Conversation)* ลักษณะพิเศษของการยืนยันตัวตนในรูปแบบนี้ คือ ก่อนที่จะมีการส่งข้อมูลในการเรียกใช้งานโอเปอร์เรชั่นต่างๆจริงนั้น จะต้องมีการสร้างบริบทความปลอดภัย (Security Context Establishment) เพื่อใช้ในการจัดการความปลอดภัยระหว่างข้อความที่จะส่งระหว่างกันเสียก่อน โดยการรักษาความปลอดภัยนี้ยังสามารถแบ่งออกได้เป็น อีก 2 รูปแบบย่อย คือ การรักษาความลับส่วนตัว (Privacy) และการรักษาความสมบูรณ์ของข้อมูล (Integrity)
- *ข้อความที่มีความปลอดภัย (Secure Message)* การยืนยันตัวตนในรูปแบบนี้ไม่จำเป็นจะต้องมีการสร้างบริบทความปลอดภัยก่อนการเรียกใช้งานโอเปอร์เรชั่นต่างๆ สิ่งสำคัญก็คือ การยืนยันตัวตนในรูปแบบนี้จะเน้นในการรักษาความสมบูรณ์ของข้อมูลเพียงเท่านั้น
- ไม่มีการยืนยันตัวตนใดๆทั้งสิ้น (None)

ดังนั้นผู้ที่ต้องการเรียกใช้งานกริดเซอร์วิสจะต้องสามารถกำหนดได้ว่า จะใช้วิธีการยืนยันตัวตนแบบใดๆในการติดต่อกับกริดเซอร์วิสที่ต้องการ

- วิธีการพิสูจน์สิทธิ (Authorization Method) ตามรูปแบบของจีเอสไอที่ใช้กับโกลบัสทูลคิทเวอร์ชั่น 3 การพิสูจน์สิทธิสามารถทำได้ทั้งที่ฝั่งไคลเอนท์และฝั่งเซอร์วิส โดยการพิสูจน์สิทธิที่ฝั่งเซอร์วิสนั้น ผู้พัฒนากริดเซอร์วิสจะเป็นผู้กำหนดรูปแบบการพิสูจน์สิทธิ โดยอาศัยเอกลักษณ์ที่ได้จากการยืนยันตัวตนมาพิสูจน์สิทธิว่าจะอนุญาตให้ผู้เรียกเข้าใช้งานกริดเซอร์วิสได้หรือไม่
- การจัดการนโยบายกริม (GRIM Policy Handling) กริมถูกนำมาใช้ในจีเอสไอสำหรับโกลบัสทูลคิทเวอร์ชั่น 3 สำหรับการจัดการสิทธิการเข้าใช้ทรัพยากรต่างๆ โดย กริมจะทำหน้าที่ในการออกเครเดนเชี่ยลตัวแทน (Proxy Credential) สำหรับใช้เป็นเครเดนเชี่ยลของทรัพยากรที่ต้องการเข้าใช้ นอกจากนั้นภายในยังมีการบรรจุนโยบายสำหรับการเข้าใช้ทรัพยากรเหล่านั้น ทำให้ผู้ใช้ทรัพยากรหรือในที่นี้คือผู้เรียกใช้เซอร์วิสสามารถทำการตรวจสอบได้ว่า ตัวเองมีสิทธิในการเข้าใช้ทรัพยากรนั้นหรือไม่ และทรัพยากรหรือเซอร์วิสที่กำลังติดต่อด้วยในขณะนั้นใช้ทรัพยากรที่ต้องการหรือไม่
- การมอบอำนาจเครเดนเชี่ยล (Credential Delegation) ไคลเอนท์ที่ทำการเรียกใช้กริดเซอร์วิสสามารถทำการกำหนดได้ว่ากริดเซอร์วิสเป้าหมายนั้น จะสามารถนำเครเดนเชี่ยลที่ได้รับจากการประมวลผลการรักษาความปลอดภัยนั้นไปใช้ต่อได้เพียงไร โดยทั่วไปไคลเอนท์สามารถกำหนดการมอบอำนาจเครเดนเชี่ยล ได้ 3 รูปแบบ คือ

- *การมอบอำนาจเต็มที่ (Full Credential Delegation)* กริดเซอร์วิสสามารถนำเครเดนเชี่ยลที่ได้รับ ไปใช้ในการจัดการความปลอดภัยสำหรับกริดเซอร์วิสได้อย่างเต็มที่
- *การมอบอำนาจแบบจำกัด (Limited Credential Delegation)* กริดเซอร์วิสสามารถนำเครเดนเชี่ยลที่ได้รับ ไปใช้ในการจัดการความปลอดภัยได้อย่างจำกัด
- *ไม่มีการมอบอำนาจ (No Credential Delegation)* กริดเซอร์วิสไม่สามารถนำเครเดนเชี่ยลที่ได้รับไปใช้ในการรักษาความปลอดภัยต่อได้เลย

- เครเดนเชี่ยล (Credential) ไคลเอนท์จะต้องสามารถกำหนดได้ว่า จะนำเครเดนเชี่ยลจากที่ใดหรือแหล่งใดมาใช้ในการเรียกใช้งานกริดเซอร์วิส

จากที่กล่าวมา ทำให้มีความจำเป็นที่ผู้ใช้งานตัวสร้างตัวแทนความปลอดภัยแบบอัตโนมัติ จะต้องสามารถกำหนดสิ่งต่างๆ เหล่านี้ได้ เพื่อให้ตัวแทนความปลอดภัยที่จะถูกสร้างขึ้นนั้นสามารถทำงานได้อย่างถูกต้อง และเนื่องจากเพื่อความสะดวกในการใช้งานตัวสร้างตัวแทนอัตโนมัติ เราจึงกำหนดมีผู้ใช้งานสามารถกำหนดไฟล์โครงแบบความปลอดภัย เพื่อให้ตัวสร้างตัวแทนอัตโนมัติ สามารถอ่านและสร้างตัวแทนความปลอดภัยได้ตามที่กำหนด โดยรูปแบบของไฟล์โครงแบบความปลอดภัยนี้จะอยู่ในรูปเอ็กซ์เอ็มแอล โดยยึดต้นแบบจากไฟล์โครงแบบความปลอดภัยที่ใช้ในการตั้งค่าความปลอดภัยต่างๆให้กับกริดเซอร์วิสของโกลบัสทูลคิทเวอร์ชั่น 3 แต่เนื่องจากไฟล์โครงแบบความปลอดภัยที่ใช้ในโกลบัสทูลคิทเวอร์ชั่น 3 นั้น จะมีการกำหนดแค่รูปแบบของวิธีพิสูจน์ตัวตนเท่านั้น ทำให้ไม่เพียงพอกับการใช้งานร่วมกับตัวแทนความปลอดภัยของเรา ทำให้จะต้องมีการกำหนดรูปแบบของเอ็กซ์เอ็มแอลที่จะใช้เพิ่มเติมให้กับไฟล์โครงแบบความปลอดภัยด้วย ซึ่งจะทำให้ไฟล์โครงแบบความปลอดภัยที่ใช้กับตัวสร้างตัวแทนอัตโนมัติของเราประกอบไปด้วยส่วนสำคัญดังต่อไปนี้

- auth-method ใช้กำหนดรูปแบบวิธีการพิสูจน์ตัวตน ซึ่งสามารถกำหนดรูปแบบได้อีก 3 รูปแบบย่อย คือ none gsi และ pkey โดยเมื่อกำหนดค่าเป็น gsi จะหมายถึงกำหนดให้ใช้วิธีการพิสูจน์ตัวตนแบบบทสนทนาที่มีความปลอดภัย ซึ่งยังสามารถกำหนดระดับการป้องกัน (Protection Level) ได้เป็นอีก 2 รูปแบบ คือ privacy และ integrity ส่วนเมื่อกำหนดค่าเป็น pkey หมายถึงกำหนดวิธีการพิสูจน์ตัวตนแบบข้อความที่มีความปลอดภัย
- grim-policy-handling ใช้กำหนดรูปแบบการจัดการนโยบายกริม โดยสามารถกำหนดได้ 2 รูปแบบ คือ ignore และ grim โดยเมื่อกำหนดค่าเป็น ignore หมายถึง

ไม่มีการใช้งานนโยบายกริม ส่วนเมื่อกำหนดค่าเป็น grim หมายถึง มีการใช้งานนโยบายกริม โดยเราจะต้องกำหนดเอกลักษณ์ที่จะใช้ในการตรวจสอบด้วย

- credential-delegation ใช้กำหนดรูปแบบของการมอบอำนาจเครเดนเชี่ยล โดยเราสามารถกำหนดได้ 3 รูปแบบ คือ no limited และ full
- credential ใช้กำหนดเครเดนเชี่ยลที่จะใช้ในการประมวลผลความปลอดภัยในการติดต่อกับกริดเซอร์วิสเป้าหมาย โดยเราจะต้องกำหนดคุณสมบัติของแหล่งเก็บเครเดนเชี่ยล (Credential Repository) ที่เราต้องการจะติดต่อด้วย ได้แก่
  - type คือชนิดของแหล่งเก็บเครเดนเชี่ยลที่ต้องการ ซึ่งในงานวิจัยชิ้นนี้สามารถกำหนดได้เพียงชนิดเดียว คือ MyProxy
  - host คือ ที่อยู่ของแหล่งเก็บเครเดนเชี่ยลที่ต้องการติดต่อ
  - username คือ ชื่อผู้ใช้ที่สัมพันธ์กับแหล่งเก็บเครเดนเชี่ยลที่ต้องการติดต่อ
  - password คือ รหัสผ่านที่สัมพันธ์กับแหล่งเก็บเครเดนเชี่ยลที่ต้องการติดต่อ

เราสามารถสรุปอีเลเมนต์ต่างๆของไฟล์โครงแบบความปลอดภัยได้ดังตารางที่ 2

ตารางที่ 2 อีเลเมนต์ต่างๆของไฟล์โครงแบบความปลอดภัย

ชื่ออีเลเมนต์	คำอธิบาย	อีเลเมนต์ลูกที่อนุญาตให้ใช้ได้
auth-method	กำหนดรูปแบบวิธีการพิสูจน์ตัวตน	none หรือ gsi หรือ pkey
grim-policy-handling	กำหนดรูปแบบการจัดการนโยบายกริม	ignore หรือgrim
credential-delegation	กำหนดรูปแบบของการมอบอำนาจเครเดนเชี่ยล	no หรือ limited หรือ full
credential	กำหนดเครเดนเชี่ยลที่จะใช้ในการประมวลผลความปลอดภัยในการติดต่อกับกริดเซอร์วิสเป้าหมาย	type, host, username และ password ตามลำดับ

สำหรับสกีมาไฟล์ (Schema File) ของไฟล์โครงแบบความปลอดภัยที่ใช้กับตัวสร้างตัวแทนอัตโนมัติ อยู่ในภาคผนวก ก

ในกรณีไฟล์โครงแบบความปลอดภัย ที่ป้อนให้กับตัวสร้างตัวแทนความปลอดภัย ไม่มีการกำหนดค่าในส่วนใด ตัวสร้างตัวแทนอัตโนมัติจะทำการกำหนดค่าโดยปริยาย ให้กับตัวแทนความปลอดภัยที่สร้างขึ้น ดังต่อไปนี้

- วิธีพิสูจน์ตัวตน ตั้งเป็นไม่มีการพิสูจน์ตัวตน หรือเท่ากับตั้งค่า auth-method เป็น none ไว้ในไฟล์โครงแบบความปลอดภัย
- การจัดการนโยบายกริม ตั้งเป็นปฏิเสธนโยบายกริม หรือเท่ากับการตั้งค่า grim-policy-handling เป็น ignore ไว้ในไฟล์โครงแบบความปลอดภัย
- การมอบอำนาจเครเดนเชี่ยล ตั้งเป็นมอบอำนาจเต็มที่ หรือเท่ากับการตั้งค่า credential-delegation เป็น full ไว้ในไฟล์โครงแบบความปลอดภัย

เราสามารถสรุปค่าโดยปริยายที่ตัวสร้างตัวแทนอัตโนมัติจะกำหนดให้กับตัวแทนความปลอดภัยได้ดังตารางที่ 3

ตารางที่ 3 ค่าโดยปริยายที่ตัวสร้างตัวแทนอัตโนมัติจะกำหนดให้กับตัวแทนความปลอดภัย

ชื่ออีเลเมนต์	ค่าโดยปริยาย
auth-method	none
grim-policy-handling	ignore
credential-delegation	full

สำหรับในส่วนการตั้งค่าเครเดนเชี่ยลนั้น ถ้าไม่มีการกำหนดค่าในส่วนนี้ ตัวสร้างตัวแทนอัตโนมัติ จะไม่มีการตั้งค่าใดๆให้เนื่องจาก ไม่มีข้อมูลพียงพอในการทำสิ่งใดได้

3.3.3.2 ข้อจำกัดของตัวแทนความปลอดภัยที่สร้างจากตัวแทนความปลอดภัย

ในงานวิจัยนี้ได้สร้างตัวสร้างตัวแทนอัตโนมัติสำหรับการสร้างตัวแทนความปลอดภัยจากจีเอสเอชและไฟล์โครงแบบความปลอดภัยที่กำหนดนั้น แต่ตัวแทนความปลอดภัยที่สร้างได้นั้นมีข้อจำกัดในบางส่วน ซึ่งผู้ใช้อาจจะต้องนำไปปรับปรุงให้เหมาะสมกับการใช้งานจริงอีกครั้ง ข้อจำกัดเหล่านี้ได้แก่

- ไม่รองรับการกำหนดรูปแบบการจัดการนโยบายกริม โดยจะตั้งค่าให้ปฏิเสธนโยบายกริมสำหรับการเรียกใช้โอเปอร์เรชั่นต่างๆ เสมอ
- ไม่รองรับการกำหนดรูปแบบการมอบอำนาจเครเดนเชี่ยล โดยจะตั้งค่าให้มีการมอบอำนาจเครเดนเชี่ยลแบบเต็มที่สำหรับการเรียกใช้โอเปอร์เรชั่นต่างๆเสมอ

ตัวแทนความปลอดภัยที่สร้างขึ้น จะมีการตั้งค่าในการเรียกใช้โอเปอร์เรชั่นต่างๆ เป็นตามทั้ง 3 ข้อที่กล่าวมาแล้วเสมอ ถึงแม้ว่าจะมีการตั้งค่าผ่านทางไฟล์โครงแบบความปลอดภัยก็

ตาม ดังนั้นในกรณีที่การเรียกใช้โอเปอร์เรชั่นนั้นต้องการการตั้งค่าความปลอดภัยนอกเหนือจากที่ตัวสร้างตัวแทนความปลอดภัยสร้างให้ ผู้ใช้จะต้องทำการแก้ไขจาวาคลาสที่ได้จากตัวสร้างตัวแทนความปลอดภัยด้วยตนเอง

3.3.3.3 การติดตั้งตัวแทนความปลอดภัยที่สร้างจากตัวสร้างตัวแทนอัตโนมัติ

หลังจากผู้ใช้ทำการสร้างตัวแทนความปลอดภัยจากตัวสร้างตัวแทนอัตโนมัตินั้น สิ่งที่ได้จากตัวแทนความปลอดภัยจะประกอบไปด้วย

- จาวาคลาสที่ทำหน้าที่เป็นคลาสทำงาน (Operation Class) ให้แก่เว็บเซอร์วิสของตัวแทนความปลอดภัย โดยคลาสเหล่านี้จะรวมอยู่ภายในแพ็กเกจเดียวกันทั้งหมด
- ไฟล์ deploy4AutomaticImpl.wsdd ใช้สำหรับการประกาศสร้างเว็บเซอร์วิสที่ทำหน้าเป็นตัวแทนความปลอดภัย
- ไฟล์ undeploy4AutomaticImpl.wsdd ใช้สำหรับการประกาศยกเลิกเว็บเซอร์วิสที่ทำหน้าที่เป็นตัวแทนความปลอดภัย

ไฟล์ที่ได้จากตัวสร้างตัวแทนอัตโนมัตินี้ ใช้สำหรับการสร้างเว็บเซอร์วิสใหม่ซึ่งทำหน้าที่เป็นตัวแทนความปลอดภัย และเนื่องจากงานวิจัยชิ้นนี้ใช้ Apache Axis [28] เป็นเว็บเซอร์วิสแพลตฟอร์ม ดังนั้นหลังจากที่ตัวสร้างตัวแทนอัตโนมัติทำการสร้างไฟล์เหล่านี้แล้ว ผู้ใช้จะต้องทำการสร้างเว็บเซอร์วิสใหม่เพื่อทำหน้าที่เป็นตัวแทนความปลอดภัยโดยอาศัยไฟล์เหล่านี้ โดยมีขั้นตอนดังต่อไปนี้

- รวมจาวาคลาสที่ตัวสร้างตัวแทนอัตโนมัติสร้างขึ้นไว้ในจาร์ไฟล์ (JAR file) แล้วจึงทำการคัดลอกจาร์ไฟล์ที่ได้ไปยังโฟลเดอร์ lib ของ Apache Axis และเนื่องจากงานวิจัยชินนี้ใช้โกลบัสทูลคิท [13] ซึ่งพัฒนาบนพื้นฐานของ Apache Axis ทำหน้าที่เป็นเว็บเซอร์วิสแพลตฟอร์มในตัวด้วย ดังนั้น ให้ทำการคัดลอกจาร์ไฟล์ไปยังโฟลเดอร์ <GLOBUS_LOCATION>/lib
- ประกาศสร้างเว็บเซอร์วิสใหม่ โดยใช้คำสั่งต่อไปนี้

```
java org.apache.axis.client.AdminClient deploy4AutomaticImpl.wsdd
```

หรืออีกทางหนึ่งผู้ใช้สามารถคัดลอกเนื้อหาที่อยู่ในไฟล์ deploy4AutomaticImpl.wsdd ไปใส่ในไฟล์ server-config.wsdd ของ Apache

Axis ได้โดยตรง ในกรณีที่ใช้โกลบัสทูลคิทเป็นเว็บเซอร์วิสแพลตฟอร์มให้คัดลอกไปใส่ในไฟล์ <GLOBUS_LOCATION>/server-config.wsdd ได้โดยตรง

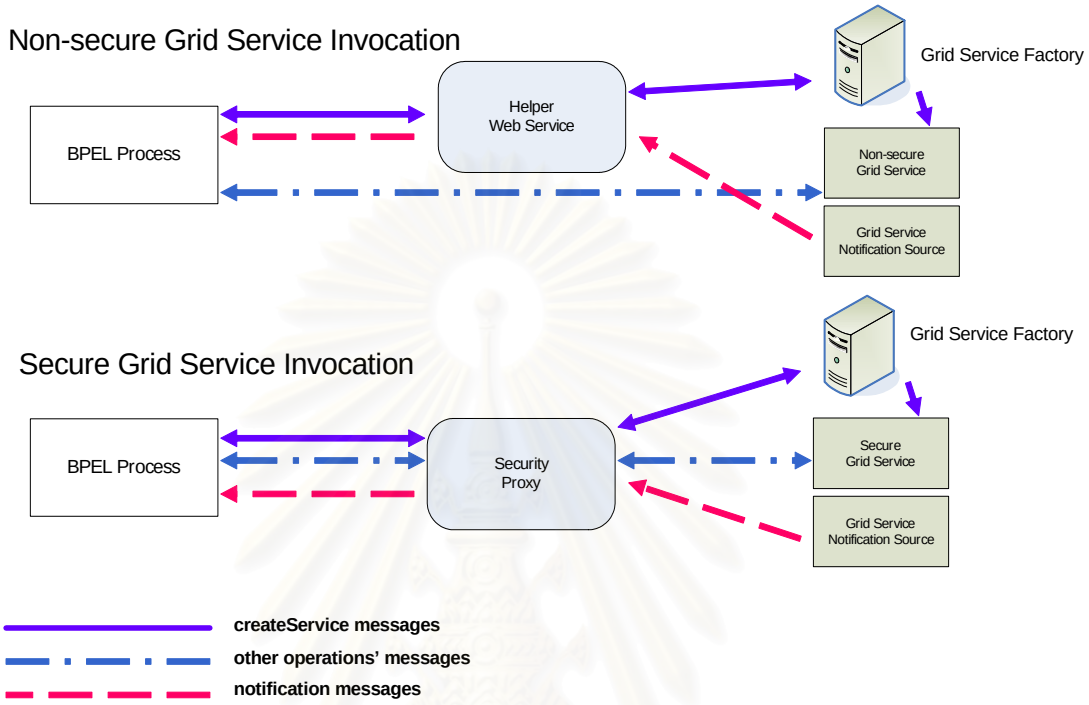
จากส่วนประกอบต่างๆ ของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสที่เราได้ออกแบบไว้แล้วนั้น เราสามารถสรุปถึงการเรียกใช้งานกลไกและคุณสมบัติต่างๆ ของกริดเซอร์วิสทั้งที่มีและไม่มีการรักษาความปลอดภัยได้ดังตารางที่ 4

ตารางที่ 4 แสดงการเรียกใช้งานกลไกและคุณสมบัติของกริดเซอร์วิสผ่านกรอบการทำงานที่สร้างขึ้น

กลไกหรือคุณสมบัติ	ชื่อพอร์ทไทป์::ชื่อโอเปอร์เรชั่น (ตามข้อกำหนดโอจีเอสไอ)	กริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย	กริดเซอร์วิสที่มีการรักษาความปลอดภัย
โรงงานเซอร์วิสและกริดเซอร์วิสอินสแตนท์	Factory::createService	เว็บเซอร์วิสผู้ช่วย	ตัวแทนความปลอดภัย
ข้อมูลเซอร์วิส	GridService::findServiceData GridService::setServiceData	เรียกใช้โดยตรง	ตัวแทนความปลอดภัย
การจัดการสถานะ	GridService::requestTerminationTimeAfter GridService::requestTerminationTimeBefore	เรียกใช้โดยตรง	ตัวแทนความปลอดภัย
การอ้างอิงเซอร์วิส	HandleResolver::findByHandle	เรียกใช้โดยตรง	ตัวแทนความปลอดภัย
กลไกการแจ้งข่าว	NotificationSource::subscribe	เว็บเซอร์วิสผู้ช่วย	ตัวแทนความปลอดภัย

ตารางที่ 4 ครอบคลุมถึงกลไกและคุณสมบัติของกริดเซอร์วิสตามข้อกำหนดโอจีเอสไอที่มีการใช้จริงในปัจจุบันทั้งหมด ในคอลัมน์ที่ 2 นั้นเป็นการระบุถึงพอร์ทไทป์และโอเปอร์เรชั่นที่ข้อกำหนดโอจีเอสไอกำหนดไว้ เพื่อใช้ในการทำงานเพื่อตอบสนองกลไกและคุณสมบัติต่างๆของกริดเซอร์วิส นอกจากนั้นยังช่วยให้ผู้ใช้สามารถเรียกใช้โอเปอร์เรชั่นได้ถูกต้องในกรณีที่มีการระบุให้มีการเรียกใช้งานโดยตรงจากกริดเซอร์วิส นอกจากนี้เราจะสังเกตได้ว่า ในกรณีของการเรียกใช้กริดเซอร์วิสที่มีการรักษาความปลอดภัยจะทำได้โดยอาศัยตัวแทนความปลอดภัยทั้งหมด ส่วนในกรณีของการเรียกใช้กริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย จะมีเพียงการใช้งานกลไกโรงงานเซอร์วิสและกลไกการแจ้งข่าวเท่านั้นที่ต้องเรียกใช้ผ่านเว็บเซอร์วิสผู้ช่วย นอกจากนั้นให้เรียกใช้

งานจากกริดเซอร์วิสได้โดยตรง เราสามารถแสดงภาพเปรียบเทียบการเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยกับที่มีการรักษาความปลอดภัยได้ดังรูปที่ 16



รูปที่ 16 เปรียบเทียบการเรียกใช้งานกริดเซอร์วิสทั้งสองแบบผ่านกรอบการทำงานที่นำเสนอขึ้น

3.4 แนวทางในการเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสผ่านกรอบการทำงานที่สร้างขึ้น

3.4.1 การเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้กลไกของกริดเซอร์วิสโดยทั่วไป

โดยทั่วไปแล้ว การสร้างเอกสารกระบวนการบีเพล (BPEL process document) ที่มีการเรียกใช้กริดเซอร์วิสนั้น ทำได้เหมือนกับการสร้างเอกสารกระบวนการบีเพลโดยทั่วไป กล่าวคือ เป็นการเขียนเพื่อเรียกใช้โอเปอร์เรชั่นต่างๆของเซอร์วิสที่เปิดให้บริการอยู่ ดังนั้นในกรณีของการสร้างเอกสารบีเพลสำหรับกริดเซอร์วิส ก็คือ การเขียนเอกสารบีเพลเพื่อเรียกใช้งานโอเปอร์เรชั่นต่างๆของกริดเซอร์วิส ทั้งการเรียกโดยตรงไปยังกริดเซอร์วิสเป้าหมาย หรือการเรียกใช้โอเปอร์เรชั่นต่างๆของเว็บเซอร์วิสผู้ช่วย รวมไปถึงการเรียกใช้งานโอเปอร์เรชั่นต่างๆของกริดเซอร์วิส ผ่านทางตัวแทนความปลอดภัย ซึ่งการเรียกใช้โอเปอร์เรชั่นของเซอร์วิสต่างๆนี้ สามารถเขียนได้โดยการใช้

อีเลเมนต์ที่ชื่อ invoke ของบีเพล ซึ่งเป็นการเรียกใช้โอเปอร์เรชั่นแบบสมวารเป็นหลัก (Synchronous Method Invocation)

3.4.2 การเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้กลไกการแจ้งข่าว

นอกจากการเรียกใช้งานโอเปอร์เรชั่นต่างๆผ่านอีเลเมนต์ invoke เพื่อตอบสนองการใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสแล้ว ยังมีกลไกหนึ่งของกริดเซอร์วิสที่เราจะต้องทราบรูปแบบการเขียนเอกสารบีเพลที่เหมาะสมจึงจะสามารถใช้งานกลไกนี้ได้อย่างมีประสิทธิภาพ ก็คือ กลไกการแจ้งข่าว จุดเด่นของกลไกนี้คือจะมีการส่งข้อความแจ้งเตือน (Notification Message) จากแหล่งต้นทางการแจ้งข่าว (Notification Source) กลับมายังแหล่งปลายทางการแจ้งข่าวที่ได้ลงทะเบียนไว้ (Subscribed Notification Sink) ดังนั้นกระบวนการบีเพลยังต้องทำหน้าที่เป็นแหล่งปลายทางการแจ้งข่าว ในกรณีที่ต้องการใช้งานกลไกการแจ้งข่าวนี้ด้วย สำหรับการสร้างเอกสารกระบวนการบีเพลเพื่อทำหน้าที่นี้จะมีหลักการที่สำคัญ คือ กระบวนการบีเพลจะต้องมีการรอรับข้อความที่ส่งมาจากภายนอก ซึ่งจากข้อกำหนดของบีเพลสามารถทำได้ 2 วิธี คือ pick และ onMessage นอกจากนั้นจะต้องกำหนดรายละเอียดอื่นๆที่เกี่ยวข้องไว้ภายในเอกสารด้วย ในส่วนของรายละเอียดสามารถอ่านได้จากภาคผนวก ข

จากส่วนประกอบต่างๆ ที่เรากล่าวมาทั้งหมดแล้วนั้น เราสามารถสรุปปัญหาที่ส่วนประกอบต่างๆของกรอบการทำงานที่นำเสนอได้ดังตารางที่ 5

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 5 แสดงความสามารถและปัญหาที่ส่วนประกอบต่างๆของกรอบการทำงานสามารถแก้ไขได้

ส่วนประกอบ	ทำงานร่วมกับกริดเซอร์วิส		เพื่อแก้ไขปัญหา				คุณสมบัติอื่นๆ
	ไม่มีการรักษาความปลอดภัย	มีการรักษาความปลอดภัย	การอ้างอิงอินสแตนท์ที่ต่างกัน	การเชื่อมต่อแบบพลวัฒน์	การใช้งานกริดเซอร์วิสอินสแตนท์	คุณสมบัติการสืบทอดพอร์ทไทป์	ปรับปรุงดับบลิวเอสดีแอลให้ใช้กับบีเพลได้
เว็บเซอร์วิสผู้ช่วย	✓		✓	✓	✓		
เครื่องมือปรับปรุงดับบลิวเอสดีแอล	✓				✓	✓	✓
ตัวแทนความปลอดภัย		✓	✓	✓	✓	✓	✓

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 4

ขั้นตอนการพัฒนากระบวนการบีเพลในการเรียกใช้กริดเซอร์วิสและตัวอย่าง

4.1 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานเว็บเซอร์วิส

การพัฒนากระบวนการบีเพลก็คือการพัฒนาเอกสารกระแสงานตามข้อกำหนดบีเพล เพื่อให้กระบวนการนั้นทำงานตามที่ต้องการ จากนั้นจึงนำเอกสารกระแสงานที่ได้ไปทำงานบนระบบทำงานกระแสงาน เนื่องจากบีเพลถูกออกแบบมาให้ใช้งานกับเว็บเซอร์วิสอยู่แล้ว ดังนั้น การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานเว็บเซอร์วิสจึงสามารถทำได้อย่างปรกติ ซึ่งสามารถสรุปได้เป็นขั้นตอนดังต่อไปนี้

1. ค้นหาข้อมูลของเว็บเซอร์วิสที่เราต้องการจะเรียกใช้ สิ่งสำคัญในขั้นตอนนี้ คือ เราจะต้องทราบดับบลิวเอสดีแอลของเว็บเซอร์วิสที่เราจะเรียกใช้
2. เขียนเอกสารกระบวนการบีเพลโดยอาศัยดับบลิวเอสดีแอลของเว็บเซอร์วิสที่ค้นหาได้ นอกจากนั้น เราจะต้องทำการพัฒนาดับบลิวเอสดีแอลของกระบวนการบีเพลที่เราสร้างขึ้นด้วย เพื่อให้ผู้ใช้สามารถเรียกใช้งานกระบวนการบีเพลได้อย่างถูกต้อง

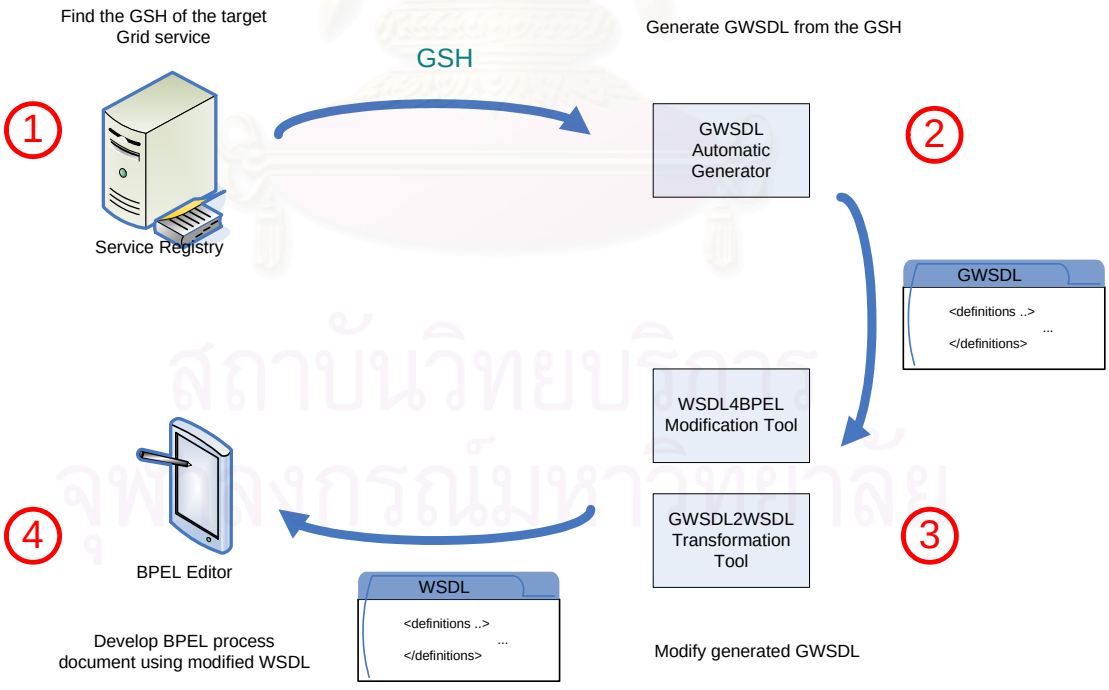
จากที่กล่าวมาจะพบว่า การเขียนเอกสารกระบวนการบีเพลสำหรับเรียกใช้เว็บเซอร์วิสนั้น สามารถทำได้โดยง่ายและไม่มีความสลับซับซ้อนมากนัก ต่างจากการเขียนหรือพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสดังที่จะได้กล่าวในหัวข้อต่อไป

4.2 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย

จากที่กล่าวมาแล้วว่า ในการใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย จะต้องอาศัยส่วนประกอบที่สำคัญ 2 ส่วนของกรอบการประสานการทำงานร่วมกันที่สร้างขึ้น คือ เว็บเซอร์วิสผู้ช่วย และเครื่องมือปรับปรุงดับบลิวเอสดีแอล ควบคู่ไปกับการเรียกใช้งานไปยังกริดเซอร์วิสโดยตรง จะทำให้เราสามารถเรียกใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสเหล่านั้นได้อย่างเต็มที่นั้น เราสามารถสรุปเป็นขั้นตอนต่างๆ ในการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยได้ดังต่อไปนี้

1. ค้นหาข้อมูลของกริดเซอร์วิสที่ต้องการจะเรียกใช้งาน ข้อมูลที่ได้จากขั้นนี้ ก็คือ จีเอสเอช (GSH – Grid Service Handler) ของกริดเซอร์วิสที่เราต้องการจะเรียกใช้
2. ทำการสร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสที่ต้องการใช้งาน โดยใช้เครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ
3. นำจีดับบลิวเอสดีแอลของกริดเซอร์วิสที่เราต้องการเรียกใช้ที่สร้างขึ้นจากเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ มาผ่านเครื่องมือในการแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอลและเครื่องมือปรับปรุงบีเพลให้เหมาะสมกับการใช้งานกับบีเพล เพื่อให้ได้ดับบลิวเอสดีแอลของกริดเซอร์วิสที่สามารถใช้ในการพัฒนาเอกสารกระแสงานบีเพลได้
4. พัฒนาเอกสารกระบวนบีเพลเพื่อเรียกใช้งานโอเปอร์เรชั่นต่างๆของกริดเซอร์วิสที่ต้องการโดยใช้ดับบลิวเอสดีแอลที่ผ่านการปรับปรุงจากเครื่องมือต่างๆแล้ว

เราสามารถสรุปขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยโดยใช้ภาพได้ดังรูปที่ 17 และสามารถอ่านรายละเอียดเพิ่มเติมได้จากภาคผนวก ค



รูปที่ 17 ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย

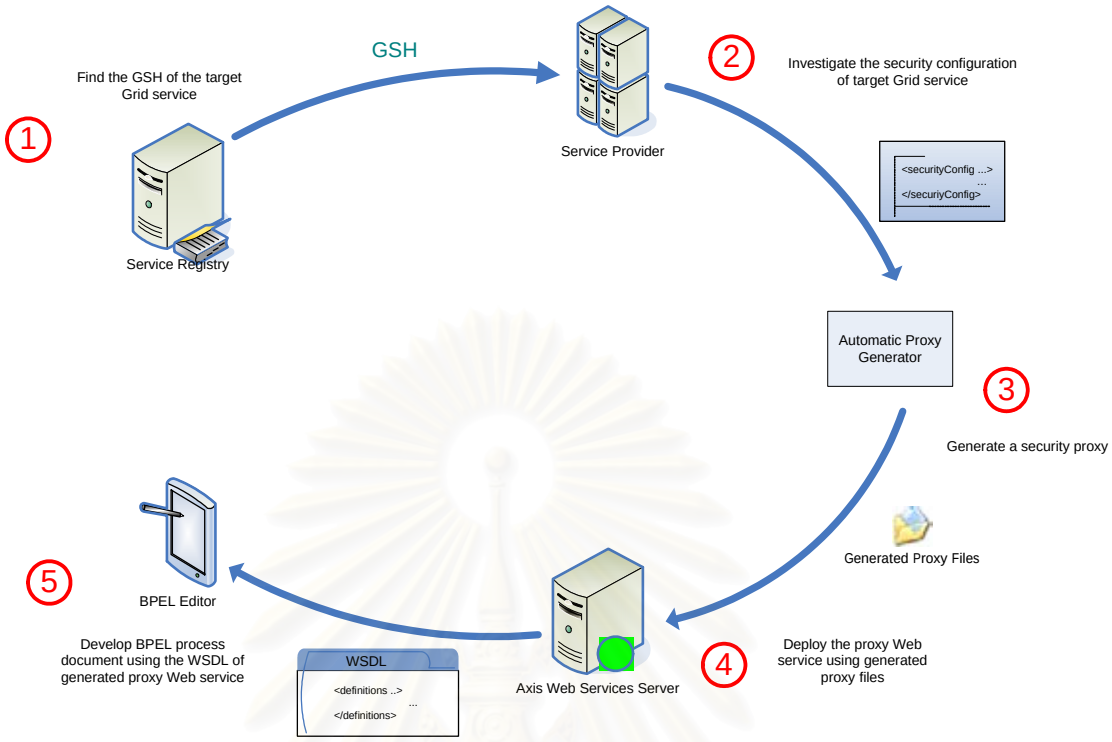
4.3 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย

สำหรับการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้นจะมีข้อแตกต่างที่สำคัญจากการพัฒนาเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย คือ การเรียกใช้งานกลไก คุณสมบัติ และโอเปอร์เรชั่นต่างๆของกริดเซอร์วิสที่มีการรักษาความปลอดภัย จะทำงานผ่านตัวแทนความปลอดภัยทั้งหมด โดยไม่ต้องมีการใช้เว็บเซอร์วิสผู้ช่วยและจะไม่มีการติดต่อโดยตรงไปยังกริดเซอร์วิสเป้าหมาย โดยสามารถสรุปเป็นขั้นตอนต่างๆได้ดังต่อไปนี้

1. ค้นหาข้อมูลของกริดเซอร์วิสที่ต้องการจะเรียกใช้งาน โดยจะต้องทราบจีเอสเอชของกริดเซอร์วิสที่ต้องการจะใช้งาน
2. ศึกษาถึงรูปแบบการรักษาความปลอดภัยของกริดเซอร์วิสที่เราต้องการจะเรียกใช้ จากนั้นจึงสร้างไฟล์โครงแบบความปลอดภัยที่อธิบายถึงความปลอดภัยที่เกี่ยวข้องกับกริดเซอร์วิสทั้งหมด
3. ใช้ตัวสร้างตัวแทนอัตโนมัติเพื่อสร้างตัวแทนความปลอดภัยเพื่อทำหน้าที่ติดต่อกับกริดเซอร์วิสที่ต้องการ
4. ติดตั้งตัวแทนความปลอดภัยจากไฟล์ที่ได้จากตัวสร้างตัวแทนอัตโนมัติ
5. พัฒนาเอกสารกระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย โดยใช้ดับบลิวเอสดีแอลของเว็บเซอร์วิสที่ทำหน้าที่เป็นตัวแทนความปลอดภัยที่สร้างขึ้น

เราสามารถสรุปขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยได้ดังรูปที่ 18 และสามารถอ่านรายละเอียดเพิ่มเติมได้จากภาคผนวก ง

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 18 ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย

4.4 ตัวอย่างการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย

4.4.1 กริดเซอร์วิสที่มีการรักษาความปลอดภัยตัวอย่าง

ในการทดสอบการทำงานของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสนั้น เราจะสร้างกริดเซอร์วิสที่มีการรักษาความปลอดภัยขึ้นมาสำหรับให้กระบวนการบีเพลเรียกใช้ โดยกริดเซอร์วิสนี้จำลองมาจากกริดเซอร์วิสที่สร้างขึ้นสำหรับการคำนวณทางการจับตัวระหว่างโมเลกุล (Molecular Screening) และใช้ซอฟต์แวร์ที่ชื่อ Autodock [29] และให้ชื่อว่า TransientAutodockInstanceService หน้าที่หลักก็คือ ให้ผู้ใช้สามารถส่งงานไฟล์อินพุทสำหรับโปรแกรม Autodock เข้ามาทำงานยังเครื่องที่เซอร์วิสนี้เปิดให้บริการอยู่ ลักษณะสำคัญของเซอร์วิสนี้ก็คือ มีการใช้งานโรงงานเซอร์วิส สำหรับสร้างกริดเซอร์วิสอินสแตนท์ขึ้นมาเพื่อใช้ในการทำงาน กริดเซอร์วิสอินสแตนท์ที่สร้างขึ้นจะประกอบไปด้วยโอเปอร์เรชั่นที่กำหนดขึ้นเอง 2 โอเปอร์เรชั่น ได้แก่

- submitAutodock ใช้สำหรับให้ผู้ใช้ส่งงานเข้ามาทำงานยังเครื่องที่เซอร์วิสเปิดให้บริการอยู่
- setAutodockJobStatus ใช้สำหรับให้ตัวจัดการทรัพยากร (Resource Scheduler) ที่รับผิดชอบการทำงานนี้ แจ้งสถานะของการทำงานกลับมายังกริดเซอร์วิสอินสแตนท์

นอกจากนั้น กริดเซอร์วิสอินสแตนท์นี้จะมีการระบุข้อมูลเซอร์วิสที่ชื่อ AutodockJobData ซึ่งระบุข้อมูลต่างๆ เกี่ยวกับงานที่กำลังทำอยู่นี้ กริดเซอร์วิสนี้ยังอนุญาตให้ผู้ใช้ใช้กลไกการบอกรับการแจ้งข่าว โดยกริดเซอร์วิสอินสแตนท์จะมีการแจ้งข่าวกลับไปยังผู้ลงทะเบียนบอกรับ เมื่อสถานะของงานเปลี่ยนไปอันเนื่องมาจากมีการเรียกใช้โอเปอร์เรชั่น setAutodockJobStatus

สำหรับการรักษาความปลอดภัยนั้น เนื่องจากกริดเซอร์วิสนี้เป็นการจำลองขึ้นมา และไม่มีการใช้งานจริง เราจะสร้างให้โอเปอร์เรชั่นต่างๆ ของกริดเซอร์วิสนี้ มีรูปแบบการรักษาความปลอดภัยที่ต่างกัน ดังต่อไปนี้

- โอเปอร์เรชั่น submitAutodock ใช้วิธีการยืนยันตัวตนแบบ GSISecureConversation โดยตั้งระดับการป้องกันเป็น privacy
- โอเปอร์เรชั่น setAutodockJobStatus ใช้วิธีการยืนยันตัวตนแบบ GSISecureMessage
- โอเปอร์เรชั่น destroy ใช้วิธีการยืนยันตัวตนแบบ GSISecureConveration โดยรองรับระดับการป้องกันทั้ง 2 รูปแบบ
- โอเปอร์เรชั่น subscribe ไม่มีการยืนยันตัวตน

ในส่วนของการพิสูจน์สิทธิฝั่งเซอร์วิสนั้น กำหนดให้ไม่มีการพิสูจน์สิทธิใดๆทั้งสิ้น เราสามารถแสดงไฟล์โครงแบบความปลอดภัยที่ใช้ในการสร้างกริดเซอร์วิสนี้ ได้ดังรูปที่ 19

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

```
<securityConfig xmlns="http://www.globus.org""
xmlns:autodock="http://chemistry.thaigrid.org/portal/gridservices/autodock/transient/instance"
xmlns:ogsi="http://www.gridforum.org/namespaces/2003/03/OGSI" >
<method name="autodock:submitAutodock">
 <auth-method>
  <gsi>
   <protection-level>
    <privacy/>
   </protection-level>
  </gsi>
 </auth-method>
</method>
<method name="autodock:setAutodockJobStatus">
 <auth-method>
  <pkey/>
 </auth-method>
</method>
<method name="destroy">
 <auth-method>
  <gsi/>
 </auth-method>
</method>
<method name="subscribe">
 <auth-method>
  <none/>
 </auth-method>
</method>
</securityConfig>
```

รูปที่ 19 ไฟล์โครงแบบความปลอดภัยสำหรับกริดเซอร์วิส TransientAutodockService

4.4.2 การสร้างตัวแทนความปลอดภัยของกริดเซอร์วิส

ในการใช้งานกริดเซอร์วิสตัวอย่างผ่านกระบวนการบีเพลนั้น เราจำเป็นจะต้องสร้างตัวแทนความปลอดภัยขึ้น โดยใช้ตัวสร้างตัวแทนความปลอดภัย ซึ่งในการสร้างตัวแทนความปลอดภัยจำเป็นจะต้องสร้างไฟล์โครงแบบความปลอดภัยสำหรับให้ตัวสร้างตัวแทนความ

ปลอดภัยใช้ในการสร้างตัวแทน โดยเราจะทำการเพิ่มเติมไฟล์โครงแบบความปลอดภัยในรูปที่ 19 โดยกำหนดข้อมูลเพิ่มเติมสำหรับการสร้างตัวแทนความปลอดภัย ดังนี้

- ปฏิเสธการจัดการนโยบายกริม
- ตั้งค่าการมอบอำนาจแบบเต็มที่
- เครเดนเชี่ยลให้เครเดนเชี่ยลที่เก็บอยู่ใน MyProxy ที่เครื่อง apollo1.cp.eng.chula.ac.th โดยมีชื่อผู้ใช้คือ g46pan พร้อมทั้งรหัสผ่าน

เนื่องด้วยการตั้งค่าทั้งสาม เป็นการตั้งค่าให้ใช้กับทั้งเซอร์วิส โดยไม่เจาะจงกับโอเปอร์เรชั่นใด ดังนั้น เราจะตั้งค่าเป็นแบบปริยาย (default) ซึ่งเราสามารถแสดงสิ่งที่เพิ่มเติมในไฟล์โครงแบบความปลอดภัยเดิมได้ดังรูปที่ 20

```
<securityConfig xmlns="http://www.globus.org" …
 >
…
        <grim-policy-handling>
                <ignore/>
        </grim-policy-handling>

        <credential-delegation>
                <full/>
        </credential-delegation>

        <credential type="MyProxy">
                <host>apollo1.cp.eng.chula.ac.th</host>
                <username>g46pan</username>
                <password>5a44e3e1</password>
        </credential>

</securityConfig>
```

รูปที่ 20 สิ่งที่เพิ่มเติมลงในไฟล์โครงแบบความปลอดภัยสำหรับใช้ในการสร้างตัวแทนความปลอดภัย

เนื่องจากกริดเซอร์วิสที่เราจะสร้างตัวแทนความปลอดภัย มีการใช้งานกลไกโรงงานเซอร์วิส ทำให้เราจำเป็นจะต้องสร้างไฟล์โครงแบบความปลอดภัยสำหรับโรงงานเซอร์วิสด้วย ซึ่งโรงงานเซอร์วิสของกริดเซอร์วิสของเรา มีการใช้รูปแบบการยืนยันตัวตนแบบ

GSISecureMessage ทำให้เราสามารถสร้างไฟล์โครงแบบความปลอดภัยสำหรับโรงงานกริดเซอร์วิสได้รูปที่ 21

```
<securityConfig xmlns="http://www.globus.org"
 xmlns:ogsi="http://www.gridforum.org/namespaces/2003/03/OGSI">
        <method name="ogsi:createService">
                <auth-method>
                        <pkey/>
                </auth-method>
        </method>
        <method name="ogsi:destroy">
                <auth-method>
                        <pkey/>
                </auth-method>
        </method>
        <credential type="MyProxy">
                <host>apollo1.cp.eng.chula.ac.th</host>
                <username>g46pan</username>
                <password>5a44e3e1</password>
        </credential>
</securityConfig>
```

รูปที่ 21 ไฟล์โครงแบบความปลอดภัยสำหรับโรงงานเซอร์วิส

จากนั้น เมื่อเราทำการสร้างตัวแทนความปลอดภัยโดยใช้ตัวสร้างตัวแทนอัตโนมัติ เราจะได้ไฟล์ที่เกี่ยวข้องทั้งหมด แล้วจึงทำตามขั้นตอนที่กำหนดไว้ในหัวข้อ 3.3.3.3 เราก็จะได้ตัวแทนความปลอดภัยสำหรับการเรียกใช้โดยกระบวนการบีเพลต่อไป

4.4.3 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้กริดเซอร์วิสตัวอย่าง

เพื่อทดสอบการใช้งานกรอบการประสารการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสที่เราสร้างขึ้นนั้น เราได้สร้างกระบวนการบีเพลหนึ่งที่มีการเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัยตัวอย่างที่เราได้กล่าวมาแล้ว สำหรับการทำงานของกระบวนการบีเพลนี้ คือ กระบวนการบีเพลนี้จะเรียกใช้โอเปอร์เรชั่น createServiceInstance เพื่อทำการสร้างกริดเซอร์วิสอินสแตนท์ให้สามารถเรียกใช้งานได้จริง จากนั้นจะเรียกใช้โอเปอร์เรชั่น setAutodockJobStatus หลังจากนั้นกริดเซอร์วิสนี้ จะรอรับการแจ้งข่าวจากกริดเซอร์วิสอินสแตนท์ และจะจบการทำงาน

เมื่อมีข้อความการแจ้งข่าวส่งมาถึง เราสามารถแสดงโค้ดเทียม (Pseudo Code) ของกระบวนการบีเพลนี้ได้ดังรูปที่ 22

```
Start
        proxy.createServiceInstance
        proxy.setAutodockJobStatus
        proxy.subscribeToProxyByName
        pick a deliverNotificationFromProxy message
End
```

รูปที่ 22 โค้ดเทียมแสดงการทำงานของกระบวนบีเพลที่สร้างขึ้น

จากโค้ดเทียมในรูปที่ 22นั้น จะพบว่าในการทำงานของกระบวนการบีเพลนี้ จะเรียกใช้โอเปอร์เรชั่นต่างๆของตัวแทนความปลอดภัยแต่เพียงอย่างเดียวโดยไม่ต้องอาศัยเซอร์วิสอื่นๆเข้ามาเกี่ยวข้องเลย

สำหรับการเขียนเอกสารกระบวนการบีเพลนี้ ในขั้นตอนแรกเราจะต้องทราบดับบลิวเอสดีแอลของตัวแทนความปลอดภัย สิ่งที่เราจะต้องเพิ่มเติมในดับบลิวเอสดีแอลของกริดเซอร์วิสก็คือ การประกาศตัวระบุความสัมพันธ์ เนื่องมาจากโดยปรกติแล้ว ดับบลิวเอสดีแอลจะไม่มีการประกาศถึงข้อมูลตัวระบุความสัมพันธ์ ดังนั้นเราจะต้องเพิ่มเติมดับบลิวเอสดีแอลให้มีการประกาศตัวระบุความสัมพันธ์อย่างชัดเจน และเนื่องจากตัวแทนความปลอดภัยสามารถทำหน้าที่บอกรับการแจ้งข่าวได้ผ่านทางโอเปอร์เรชั่น subscribeToProxy และ subscribeToProxyByName ซึ่งทั้งสองโอเปอร์เรชั่นนี้รับอินพุท 1 ตัวที่ชื่อ correlationInteger ซึ่งตัวแทนความปลอดภัยจะใช้เป็นตัวระบุความสัมพันธ์ในการส่งข้อมูลกลับมายังกระบวนการบีเพลอยู่แล้ว เราจึงกำหนดให้ใช้อินพุทนี้ของทั้งสองโอเปอร์เรชั่นเป็นตัวระบุความสัมพันธ์ เราสามารถแสดงสิ่งที่เพิ่มเติมในดับบลิวเอสดีแอลของตัวแทนความปลอดภัยได้ดังรูปที่ 23

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<definitions …
        xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        …>
        <bpws:property name="correlationInteger" type="xsd:int"/>
        <bpws:propertyAlias propertyName="tns:correlationInteger"
        messageType="tns:subscribeToProxyByNameRequest" part="parameters"
        query="/tns8:subscribeToProxyByName/tns8:correlationInteger"/>
        …
</definitions>
```

รูปที่ 23 สิ่งที่เพิ่มเติมในดับบลิวเอสดีแอลของตัวแทนความปลอดภัยเพื่อใช้งานกับกระบวนการบีเพล

เอกสารกระบวนการบีเพลที่เราสร้างขึ้นนั้นจะประกอบไปด้วยส่วนสำคัญ 4 ส่วน ได้แก่

- ส่วนประกาศ partnerLink ใช้ในการประกาศถึงพาร์ทเนอร์ต่างๆที่เกี่ยวข้องกับกระบวนการบีเพลนี้ทั้งหมด ซึ่งในกระบวนการนี้จะเกี่ยวข้องกับพาร์ทเนอร์ 2 ประเภท คือ ไคลเอนท์ที่เป็นผู้เรียกใช้กระบวนการบีเพลนี้ และ AutodockProvider ซึ่งเป็นผู้ให้บริการกริดเซอร์วิส TransientAutodockService นี้ โดยการกำหนดพาร์ทเนอร์นี้จะมีการอ้างอิงถึงประเภทความสัมพันธ์พาร์ทเนอร์ของพาร์ทเนอร์ที่เกี่ยวข้องนี้ด้วย เราสามารถแสดงเอกสารกระบวนการบีเพลในส่วนนี้ได้ดังรูปที่ 24

```
<partnerLinks>
        <partnerLink name="client" partnerLinkType="tns:DemoAutodockBPELProcessForThesis"
        myRole="DemoAutodockBPELProcessForThesisProvider"
        partnerRole="DemoAutodockBPELProcessForThesisRequester"/>
        <partnerLink name="AutodockProvider"
        partnerLinkType="autodock:TransientAutodockInstancePortType0Link"
        partnerRole="TransientAutodockInstancePortType0Provider"/>
</partnerLinks>
```

รูปที่ 24 ส่วนประกาศ partnerLink ของกระบวนการบีเพลที่สร้างขึ้น

- ส่วนประกาศตัวแปร ใช้ประกาศถึงตัวแปรต่างๆในกระบวนการของเรา ตัวแปรเหล่านี้คือตัวแปรของข้อความที่เกิดจากการเรียกใช้โอเปอร์เรชั่นต่างๆของเซอร์วิสที่เกี่ยวข้อง

รวมไปถึงข้อความที่ส่งและรับเข้ามาในกระบวนการบีเพลด้วย เราสามารถแสดงส่วนประกาศตัวแปรของเอกสารกระบวนการบีเพลนี้ได้รูปที่ 25

```
<variables>
        <variable name="input"  messageType="tns:DemoAutodockBPELProcessForThesisRequestMessage"/>->
        <variable name="output" messageType="tns:DemoAutodockBPELProcessForThesisResponseMessage"/>
        <variable name="input4SetAutodockJobStatus" messageType="autodock:setAutodockJobStatusRequest"/>
        <variable name="output4SetAutodockJobStatus"
        messageType="autodock:setAutodockJobStatusResponse"/>
        <variable name="input4CreateServiceInstance" messageType="autodock:createServiceInstanceRequest"/>
        <variable name="output4CreateServiceInstance"
        messageType="autodock:createServiceInstanceResponse"/>
        <variable name="input4Destroy0" messageType="autodock:destroy0Request"/>
        <variable name="output4Destroy0" messageType="autodock:destroy0Response"/>
        <variable name="input4SubscribeToProxyByName"
        messageType="autodock:subscribeToProxyByNameRequest"/>
        <variable name="output4SubscribeToProxyByName"
        messageType="autodock:subscribeToProxyByNameResponse"/>
        <variable name="inputFromNotificaitonProxy"
        messageType="tns:DeliverNotificationFromProxyInputMessage"/>
</variables>
```

รูปที่ 25 ส่วนประกาศตัวแปรของกระบวนการบีเพลที่สร้างขึ้น

- ส่วนประกาศตัวระบุความสัมพันธ์ ใช้ประกาศตัวระบุความสัมพันธ์ที่ใช้ในกระบวนการนี้ ซึ่งในกระบวนการของเรานี้มีการใช้ตัวระบุความสัมพันธ์เพียงตัวเดียว โดยให้ชื่อว่า correlationInteger สามารถแสดงส่วนประกาศตัวระบุความสัมพันธ์ได้รูปที่ 26

```
<correlationSets>
        <correlationSet name="correlationInteger" properties="cor:correlationInteger"/>
</correlationSets>
```

รูปที่ 26 ส่วนประกาศตัวระบุความสัมพันธ์ของกระบวนการบีเพลที่สร้างขึ้น

- ส่วนควบคุมการทำงาน ในส่วนนี้จะเป็นส่วนที่บ่งบอกว่ากระบวนการบีเพลนี้มีการทำงานอย่างไรบ้าง มีการเรียกใช้โอเปอร์เรชั่นใดตามลำดับใดบ้าง สามารถแสดงส่วนควบคุมการทำงานของกระบวนการบีเพลที่สร้างขึ้นได้ดังรูปที่ 27

```
<sequence name="main">
    <receive name="receiveInput" partnerLink="client" portType="tns:DemoAutodockBPELProcessForThesis" operation="initiate"
    variable="input" createInstance="yes"/>
```

```
<invoke name="createServiceInstance" partnerLink="AutodockProvider"
portType="autodock:TransientAutodockInstancePortType0" operation="createServiceInstance"
inputVariable="input4CreateServiceInstance" outputVariable="output4CreateServiceInstance"/>
<assign name="assignInput4SetAutodockJobStatus">
        <copy>
                <from variable="input" part="payload"
query="/tns:DemoAutodockBPELProcessForThesisRequest/tns:autodockJobNumber"/>
                <to variable="input4SetAutodockJobStatus" part="parameters"
query="/package:setAutodockJobStatus/package:autodockJobNumber"/>
        </copy>
        <copy>
                <from variable="input" part="payload"
query="/tns:DemoAutodockBPELProcessForThesisRequest/tns:autodockJobStatus"/>
                <to variable="input4SetAutodockJobStatus" part="parameters"
query="/package:setAutodockJobStatus/tns1:jobStatus"/>
        </copy>
</assign>
<invoke name="setAutodockJobStatus" partnerLink="AutodockProvider"
portType="autodock:TransientAutodockInstancePortType0" operation="setAutodockJobStatus"
inputVariable="input4SetAutodockJobStatus" outputVariable="output4SetAutodockJobStatus"/>
<assign name="assign4SubscriptionByName">
        <copy>
                <from variable="input" part="payload"
query="/tns:DemoAutodockBPELProcessForThesisRequest/tns:correlationInteger"/>
                <to variable="input4SubscribeToProxyByName" part="parameters"
query="/package:subscribeToProxyByName/package:correlationInteger"/>
        </copy>
        <copy>
                <from>
                        <serviceDataName xmlns="http://g1108706624806.generated.proxy.ws.zeus.com">
                                TransientAutodockData
                        </serviceDataName>
                </from>
                <to variable="input4SubscribeToProxyByName" part="parameters"
query="/package:subscribeToProxyByName/package:serviceDataName"/>
        </copy>
        <copy>
                <from>
                        <sink xmlns="http://g1108706624806.generated.proxy.ws.zeus.com">
                                <handle>
                                http://localhost:9700/orabpel/default/DemoAutodockBPELProcessForThesis
                                </handle>
                        </sink>
                </from>
                <to variable="input4SubscribeToProxyByName" part="parameters"
query="/package:subscribeToProxyByName/package:sink"/>
        </copy>
```

```
            <copy>
                <from>
                    <expirationTime
            xmlns="http://g1108706624806.generated.proxy.ws.zeus.com">2547-04-05T11:05:46.051Z</expirationTime>
                </from>
                <to variable="input4SubscribeToProxyByName" part="parameters"
        query="/package:subscribeToProxyByName/package:expirationTime"/>
            </copy>
        </assign>
        <invoke name="subscribeToProxyByName" partnerLink="AutodockProvider"
        portType="autodock:TransientAutodockInstancePortType0" operation="subscribeToProxyByName"
        inputVariable="input4SubscribeToProxyByName" outputVariable="output4SubscribeToProxyByName">
            <correlations>
                <correlation set="correlationInteger" initiate="no" pattern="out"/>
            </correlations>
        </invoke>
        <pick name="pick-1">
            <onMessage partnerLink="client" portType="tns:DemoAutodockBPELProcessForThesis"
        operation="deliverNotificationFromProxy" variable="inputFromNotificaitonProxy">
                <correlations>
                    <correlation set="correlationInteger" initiate="no"/>
                </correlations>
                <sequence>
                    <empty name="emptyAtOnMessage"/>
                </sequence>
            </onMessage>
        </pick>
        <invoke name="destroy0" partnerLink="AutodockProvider"
        portType="autodock:TransientAutodockInstancePortType0" operation="destroy0" inputVariable="input4Destroy0"
        outputVariable="output4Destroy0"/>
        <invoke name="callbackClient" partnerLink="client" portType="tns:DemoAutodockBPELProcessForThesisCallback"
        operation="onResult" inputVariable="output"/>
    </sequence>
```

รูปที่ 27 ส่วนประกาศการทำงานของกระบวนการบีเพลที่สร้างขึ้น

จากรูปที่ 27 เราสามารถแบ่งการทำงานออกเป็น 9 ขั้นตอน ได้แก่

1. รับข้อมูลการเรียกใช้กระบวนการบีเพลนี้จากไคลเอนท์ โดยใช้คำสั่ง receive
2. เรียกใช้โอเปอร์เรชั่น createServiceInstance เพื่อสร้างกริดเซอร์วิสอินสแตนท์ให้สามารถเรียกใช้งานได้ โดยใช้คำสั่ง invoke
3. คัดลอกข้อมูลจากข้อความที่ส่งมาจากไคลเอนท์ให้แก่ข้อความที่เป็นอินพุทสำหรับเรียกใช้โอเปอร์เรชั่น setAutodockJobStatus โดยใช้คำสั่ง assign

4. เรียกใช้โอเปอร์เรชั่น setAutodockJobStatus ซึ่งเป็นการเรียกใช้โอเปอร์เรชั่นของกริดเซอร์วิสที่มีการรักษาความปลอดภัย โดยใช้คำส่ง invoke
5. ตั้งค่าต่างๆที่จำเป็นในการบอกรับการแจ้งข่าวผ่านการเรียกใช้โอเปอร์เรชั่น subscribeToProxyByName โดยใช้คำสั่ง assign ซึ่งจะเห็นว่าเป็นคำสั่งเดียวกับการคัดลอกข้อมูลในขั้นตอนที่ 3 สิ่งสำคัญในขั้นตอนนี้ก็คือ จะต้องทำการตั้งค่าที่ใช้เป็นตัวระบุความสัมพันธ์ของข้อความที่ใช้เป็นอินพุทของการเรียกใช้โอเปอร์เรชั่น subscribeToProxyByName ซึ่งคืออีเลเมนต์ที่ชื่อ correlationInteger นอกจากนั้นจะต้องทำการระบุถึงตำแหน่งปลายทาง (Endpoint Location) ของกระบวนการบีเพลนี้ลงในอีเลเมนต์ sink ด้วย เพื่อให้ตัวแทนความปลอดภัยสามารถส่งข้อความแจ้งข่าวกลับมายังกระบวนการบีเพลได้อย่างถูกต้อง
6. เรียกใช้โอเปอร์เรชั่น subscribeToProxyByName เพื่อทำการบอกรับการแจ้งข่าว โดยใช้คำสั่ง invoke สิ่งสำคัญที่ทำให้การเรียกใช้โอเปอร์เรชั่นนี้ต่างจากโอเปอร์เรชั่นอื่นๆ ก็คือ ภายในอีเลเมนต์ Invoke นั้น จะมีอีเลเมนต์ภายในที่ชื่อ correlations เพื่อทำการระบุตัวระบุความสัมพันธ์ที่ส่งไปพร้อมกับข้อความนี้ด้วย
7. รอรับข้อความการแจ้งข่าว ที่จะส่งมาจากตัวแทนความปลอดภัย โดยใช้คำสั่ง pick สำหรับภายในอีเลเมนต์ pick นั้นจะประกอบด้วยอีเลเมนต์ onMessage ซึ่งเป็นตัวระบุว่าจะทำการรอรับข้อความที่ชื่อ inputFromNotificationProxy โดยได้รับข้อความนี้เข้ามาแล้ว กระบวนการก็จะทำขั้นตอนต่อไปทันที สิ่งสำคัญสำหรับอีเลเมนต์ onMessage นี้ก็คือ มีการกำหนดตัวระบุความสัมพันธ์ของข้อความที่จะรอรับด้วย
8. เรียกใช้โอเปอร์เรชั่น destroy0 เพื่อทำลายกริดเซอร์วิสอินสแตนท์ที่ใช้อยู่นี้
9. ส่งผลลัพธ์ที่ได้จากการทำงานของกระบวนการบีเพลนี้ กลับไปยังไคลเอนท์ที่เรียกใช้

นอกจากส่วนประกอบทั้ง 4 ที่ได้กล่าวมาแล้ว เอกสารกระบวนการบีเพลโดยทั่วไปยังประกอบไปด้วยส่วนสำคัญอีกส่วนหนึ่ง คือ ส่วนจัดการ (Handlers) ซึ่งประกอบไปด้วย ส่วนจัดการความผิดพลาด (Fault Handlers) ส่วนจัดการการทดแทน (Compensation Handlers) และส่วนจัดการเหตุการณ์ (Event Handlers) แต่เนื่องจากกระบวนการบีเพลนี้สร้างขึ้นมาทดสอบกรอบการประสานการทำงานร่วมกันที่เราสร้างขึ้นเท่านั้น จึงไม่ส่วนประกอบนี้อยู่ในกระบวนการบีเพลของเรา

ลักษณะสำคัญของเอกสารกระบวนการบีเพลที่สร้างขึ้นนี้ คือ มีการกำหนดตัวระบุความสัมพันธ์และใช้ตัวระบุความสัมพันธ์นี้ในการติดต่อกับตัวแทนความปลอดภัย เพื่อให้ตัวแทนความปลอดภัยใช้ตัวระบุความสัมพันธ์นี้ในการส่งข้อความกลับมายังกระบวนการบีเพลที่ถูกต้องได้ ซึ่งกลไกการจัดการตัวระบุความสัมพันธ์นี้ไม่สามารถทำได้เลยในกรณีที่เราติดต่อกับกริดเซอร์วิสโดยตรง นอกจากนั้น จากเอกสารกระบวนการบีเพลจะพบว่าเราสามารถอ่านและทำความเข้าใจกับกระบวนการบีเพลนี้ได้โดยง่าย เนื่องจากการเรียกโอเปอร์เรชั่นต่างๆของตัวแทนความปลอดภัยมีชื่อเหมือนกับการเรียกกริดเซอร์วิสโดยตรง

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 5

การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรม

5.1 การจัดการทรัพยากรในโกลบัสทูลคิท เวอร์ชั่น 3

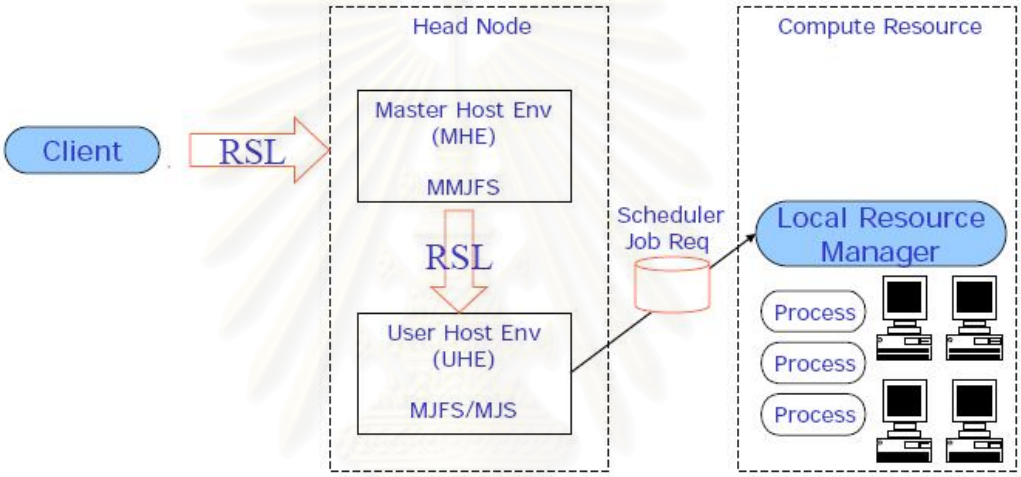
จุดมุ่งหมายที่สำคัญอย่างหนึ่งของวิทยานิพนธ์นี้ คือ การนำข้อกำหนดบีเพลมาใช้ในในการจัดการทรัพยากรกริดที่ถูกจัดการโดยโกลบัสทูลคิท หรือกล่าวอีกนัยหนึ่งก็คือ การใช้กระบวนการบีเพลมาประสานการทำงานของแกรม ซึ่งเป็นชื่อของส่วนประกอบของโกลบัสทูลคิท เวอร์ชั่น 3 [13] ที่ใช้ในการจัดการทรัพยากรกริดโดยเฉพาะ ดังนั้นเราจะต้องศึกษาองค์ประกอบต่างๆของแกรมเพื่อจะทำให้สามารถใช้กระบวนการบีเพลเรียกใช้งานได้อย่างถูกต้อง

แกรมเป็นระบบจัดการทรัพยากรที่ให้บริการเซอร์วิสสำหรับจัดการกับงาน (Job) ที่ต้องการส่งไปทำงานบนเครื่องที่แกรมเปิดให้บริการอยู่ [10] หน้าที่ของแกรม ประกอบไปด้วย

- การสร้างสภาวะแวดล้อม (Environment) ขึ้นมาเพื่อให้งานนั้นทำงาน
- จัดเตรียมไฟล์ต่างๆที่จำเป็นสำหรับงานนั้นมาไว้ในสภาวะแวดล้อมที่สร้างขึ้น
- ส่งงานไปทำงานยังตัวจัดการทรัพยากรท้องถิ่น (Local Scheduler)
- ตรวจสอบสภาพของงานที่กำลังทำงานอยู่
- แจ้งข่าวเมื่องานมีการเปลี่ยนแปลงสถานะ
- จัดการกับข้อมูลผลลัพธ์และความผิดพลาดที่เกิดจากการทำงานนี้

สำหรับแกรมในโกลบัสทูลคิทเวอร์ชั่น 3 นั้น มีความแตกต่างจากแกรมในโกลบัสทูลคิทเวอร์ชั่น 2 กล่าวคือ แกรมในโกลบัสทูลคิทเวอร์ชั่น 3 นั้น ถูกสร้างขึ้นในรูปแบบของกริดเซอร์วิสทั้งหมด กล่าวคือ ผู้ให้บริการแกรมจะเปิดให้บริการกริดเซอร์วิสที่ชื่อ Master Managed Job Factory Service หรือ MMJFS ซึ่งกริดเซอร์วิสนี้ทำหน้าที่เป็นโรงงานเซอร์วิสของแกรมกริดเซอร์วิสทั้งหมดเมื่อผู้ใช้ต้องการจะส่งงานมาทำงานยังเครื่องที่เปิดให้บริการแกรมอยู่นั้น จะต้องทำการสร้างกริดเซอร์วิสอินสแตนท์ใหม่จากกริดเซอร์วิสที่ได้กล่าวมาแล้ว โดยการระบุรูปแบบของงานที่ต้องการส่งมาทำงานในรูปแบบของอาร์เอสแอล (RSL – Resource Specification Language) จากนั้นกริดเซอร์วิสนี้จะสร้างกริดเซอร์วิสอินสแตนท์ใหม่ที่เรียกว่า Manage Job Service หรือ MJS เพื่อให้ผู้ใช้ทำการติดต่อกับกริดเซอร์วิสอินสแตนท์นี้ในการสั่งทำงานจริงต่อไป

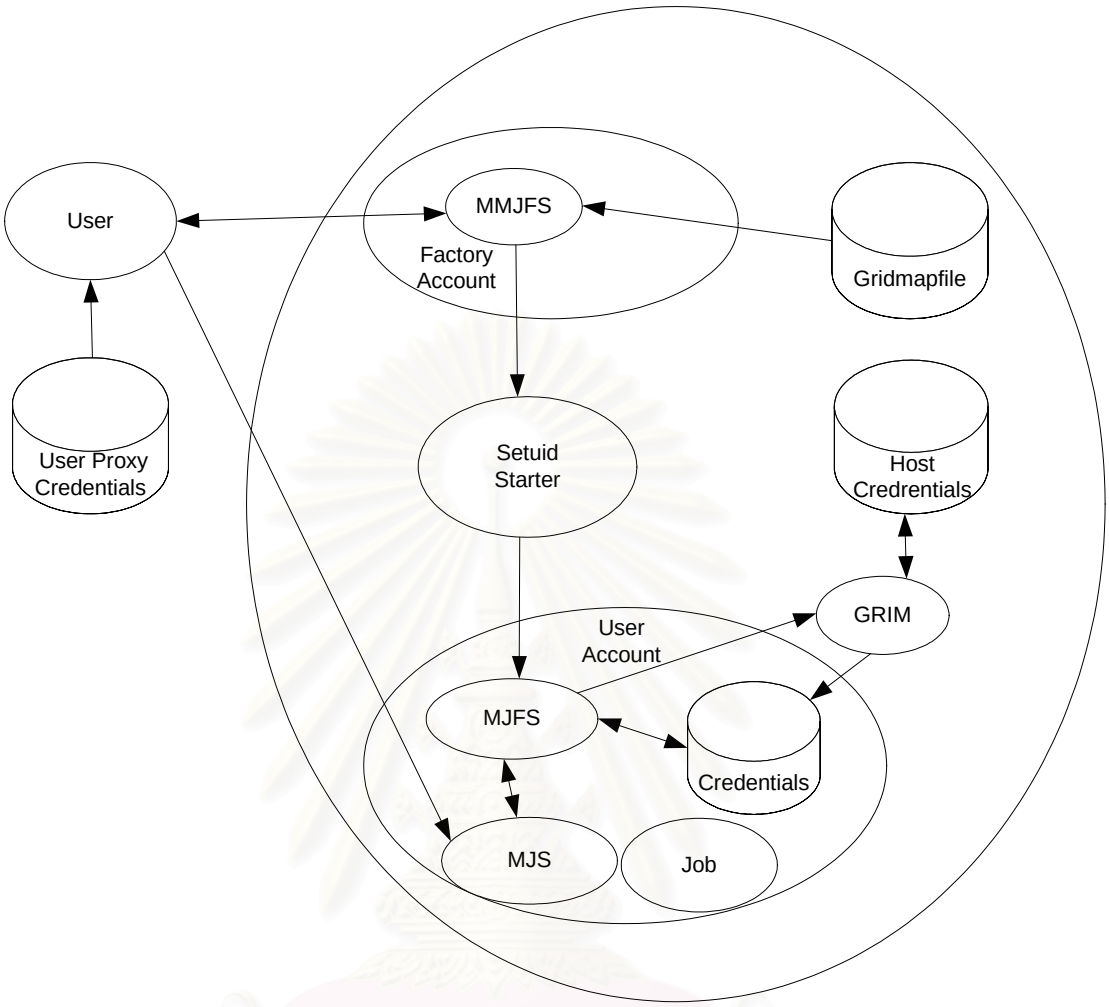
ในการทำงานของ Master Managed Job Factory Service จริงนั้น จะมีการสร้างโรงงานเซอร์วิสย่อยที่ชื่อ Managed Job Factory Service สำหรับผู้ใช้แต่ละคน กล่าวคือ เมื่อมีการร้องขอการสร้างกริดเซอร์วิสอินสแตนท์ใหม่ Master Managed Job Factory Service จะส่งการร้องขอนั้นมายัง Managed Job Factory Service ของผู้ใช้แต่ละคน จากนั้น Managed Job Factory Service จึงเป็นผู้สร้าง Managed Job Service ขึ้นสำหรับให้บริการงานนั้น แล้วส่งจีเอสเอชของกริดเซอร์วิสอินสแตนท์ที่สร้างขึ้นใหม่กลับไปยังผู้ใช้อีกทีหนึ่ง เราสามารถแสดงกริดเซอร์วิสต่างๆที่เกี่ยวข้องตามที่กล่าวมาแล้วได้รูปที่ 28



รูปที่ 28 กริดเซอร์วิสที่เกี่ยวข้องในการให้บริการแกรม

ในส่วนของการรักษาความปลอดภัยของแกรมนั้น จะอาศัยกลไกความปลอดภัยของกริดเซอร์วิสที่มีพื้นฐานอยู่บนโครงสร้างจีเอสไอในการจัดการทั้งหมด เราสามารถแสดงภาพรวมกลไกการรักษาความปลอดภัยได้ดังรูปที่ 29

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 29 ภาพรวมการรักษาความปลอดภัยในการเรียกใช้งานแกรม

จากรูปที่ 29 เราจะพบว่า ในการเรียกใช้งานแกรมกริดเซอร์วิสนั้น ผู้ใช้จะอาศัยเครเดนเชี่ยลตัวแทนของผู้ใช้ (User Proxy Credential) ในการเข้ารหัสข้อความที่จะส่งมายังแกรมกริดเซอร์วิสในรูปแบบเดียวกันกับการเรียกใช้กริดเซอร์วิสอื่นๆ เมื่อข้อความการเรียกใช้ถูกส่งมาถึง MMJFS แล้วจะมีการตรวจสอบสิทธิในรูปแบบของกริดแมปไฟล์ (Gridmapfile) ถ้าผู้ใช้ได้รับการอนุญาตให้เข้าใช้แกรมได้ ก็จะส่งการร้องขอต่อไปยัง MJFS ของผู้ใช้แต่ละคนต่อไป นอกจากนั้นจากรูปยังพบว่า MJFS มีการใช้งานกริมในกระบวนการรักษาความปลอดภัยด้วย

5.2 **การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิส** (GRAM Grid Services)

การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิส [10] นั้น สามารถทำได้โดยใช้กรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสโดยตรง โดยไม่ต้องมีการเพิ่มเติมส่วนประกอบใดๆลงในกรอบการประสานการทำงานที่มีอยู่เลย แต่อาจจะต้องมีการเพิ่มเติมความสามารถให้กับตัวสร้างตัวแทนอัตโนมัติในบางจุด ดังที่จะได้กล่าวต่อไป

ในการทดสอบการใช้กระบวนการบีเพลเรียกใช้แกรมกริดเซอร์วิสนี้ เราจะทำการเรียกใช้งานแกรมแบบฟอร์ก (Fork) ดังนั้นในส่วนของแกรมกริดเซอร์วิสที่ทำหน้าที่เป็นโรงงานเซอร์วิสจะมีชื่อว่า Master Fork Managed Job Factory Service และแกรมกริดเซอร์วิสอินสแตนท์ที่รับผิดชอบในการจัดการงานแต่ละงานจะชื่อว่า Fork Managed Job Service

5.2.1 การสร้างตัวแทนความปลอดภัยสำหรับแกรมกริดเซอร์วิส

ขั้นตอนแรกในการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิส ก็คือ การสร้างตัวแทนความปลอดภัยเช่นเดียวกับการเรียกใช้งานกริดเซอร์วิสที่มีความปลอดภัยอื่น ในการสร้างตัวแทนความปลอดภัยนั้น เราจำเป็นต้องสร้างโครงแบบความปลอดภัยของ MJS ขึ้น เพื่อใช้ในการตั้งค่าความปลอดภัยของตัวแทนความปลอดภัยที่จะสร้างขึ้น เราสามารถแสดงไฟล์โครงแบบความปลอดภัยที่สร้างขึ้นสำหรับ Fork Managed Job Service ได้ดังรูปที่ 30

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<securityConfig xmlns="http://www.globus.org">
        <method name="start">
                <run-as>
                        <caller-identity/>
                </run-as>
                <auth-method>
                        <gsi/>
                </auth-method>
        </method>
        <method name="subscribe">
                <auth-method>
                        <none/>
                </auth-method>
        </method>
        <!-- defaults -->
        <auth-method>
                <gsi/>
                <pkey/>
        </auth-method>
        <grim-policy-handling>
                <ignore/>
        </grim-policy-handling>
        <credential-delegation>
                <full/>
        </credential-delegation>
        <credential type="MyProxy">
                <host>apollo1.cp.eng.chula.ac.th</host>
                <username>g46pan</username>
                <password>5a44e3e1</password>
        </credential>
</securityConfig>
```

รูปที่ 30 ไฟล์โครงแบบความปลอดภัยของ MJS สำหรับการสร้างตัวแทนความปลอดภัย

ไฟล์โครงแบบความปลอดภัยในรูปที่ 30 นั้น สร้างขึ้นโดยใช้ไฟล์โครงแบบความปลอดภัยที่ใช้จริงในการทำงานของ MJS เป็นพื้นฐานโดยสิ่งที่เพิ่มเติมลงไปได้แก่ อีเลเมนต์ client-

authz-method grim-policy-handling credential-delegation และ credential โดยมีรายละเอียดดังต่อไปนี้

- ไม่มีการพิสูจน์สิทธิฝั่งไคลเอนท์ เนื่องจากเป็นการทำงานที่ฝั่งไคลเอนท์เท่านั้น โดยไม่เกี่ยวข้องกับฝั่งเซอร์วิส ดังนั้นเราสามารถไม่ทำการพิสูจน์สิทธิได้
- ปฏิเสธการจัดการนโยบายกริม ซึ่งจากรูปที่ 29 ถึงแม้ว่าแกรมจะมีการใช้งาน GRIM ก็ตาม แต่เราก็สามารถไม่สนใจกับนโยบายต่างๆ ที่ระบุในกริมเครเดนเชี่ยลได้
- ตั้งค่าการมอบอำนาจแบบเต็มที่ เนื่องจาก MJS มีข้อกำหนดว่าเครเดนเชี่ยลที่ได้รับจะต้องมีการมอบอำนาจอย่างเต็มที่
- เครเดนเชี่ยลให้เครเดนเชี่ยลที่เก็บอยู่ใน MyProxy ที่เครื่อง apollo1.cp.eng.chula.ac.th โดยมีชื่อผู้ใช้คือ g46pan พร้อมทั้งรหัสผ่าน

นอกจากไฟล์โครงแบบความปลอดภัยของ MJS ที่จะต้องใช้ในการสร้างตัวแทนความปลอดภัยแล้ว เนื่องจากแกรมมีการใช้งานกลไกโรงงานของกริดเซอร์วิส ผ่านทาง MMJFS ดังนั้นเราจะต้องสร้างไฟล์โครงแบบความปลอดภัยของ MMJFS ด้วย ซึ่งจากการทดลองพบว่า MMJFS มีการรักษาความปลอดภัย โดยการใช้การยืนยันตัวตนแบบ GSISecureMessgae ดังนั้นเราสามารถสร้างไฟล์โครงแบบความปลอดภัยของ MMJFS ได้ดังรูปที่ 31

```
<securityConfig xmlns="http://www.globus.org"
 xmlns:ogsi="http://www.gridforum.org/namespaces/2003/03/OGSI">
        <method name="ogsi:createService">
                <auth-method>
                        <pkey/>
                </auth-method>
        </method>
        <credential type="MyProxy">
                <host>apollo1.cp.eng.chula.ac.th</host>
                <username>g46pan</username>
                <password>5a44e3e1</password>
        </credential>
</securityConfig>
```

รูปที่ 31 ไฟล์โครงแบบความปลอดภัยสำหรับ MMJFS

ในขั้นตอนต่อไปก็คือ การสร้างตัวแทนความปลอดภัยจากตัวสร้างตัวแทนอัตโนมัติโดยใช้ไฟล์โครงแบบความปลอดภัยที่สร้างขึ้นทั้งสองไฟล์ สำหรับตัวแทนความปลอดภัยที่สร้างขึ้น

เพื่อใช้งานกับแกรมกริดเซอร์วิสนั้น จะมีความพิเศษการตัวแทนความปลอดภัยสำหรับการเรียกใช้กริดเซอร์วิสที่มีการรักษาความปลอดภัยทั่วไป ซึ่งจะได้กล่าวในหัวข้อต่อไป

5.2.2 การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิส

การพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิสนั้น สามารถทำได้เหมือนกับการเรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่มีการรักษาความปลอดภัยอื่นๆ คือ ก่อนเริ่มใช้งานจะต้องมีการเรียกใช้โอเปอร์เรชั่น createServiceInstance เช่นเดียวกับ แต่มีสิ่งที่แตกต่างก็คือเราจะต้องบรรยายลักษณะของงานที่เราต้องการจะทำในรูปแบบของอาร์เอสแอลเข้าเป็นอินพุทของการเรียกใช้โอเปอร์เรชั่น createServiceInstance นี้ด้วย เราสามารถแสดงตัวอย่างของการเรียกใช้งานโอเปอร์เรชั่น createServiceInstance สำหรับแกรมกริดเซอร์วิสได้รูปที่ 32

```
<assign name="assignRsl2Ext">
    <copy>
        <from>
            <createService xmlns="http://www.gridforum.org/namespaces/2003/03/OGSI"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
                <creationParameters>
                    <rsl:rsl xmlns:enum="http://www.globus.org/namespaces/2004/02/rsl/enum"
                    xmlns:gram="http://www.globus.org/namespaces/2004/02/rsl/gram"
                    xmlns:rsl="http://www.globus.org/namespaces/2004/02/rsl">
                        <gram:job>
                            <gram:executable>
                                ...
                            </gram:executable>
                            <gram:directory>
                                ...
                            </gram:directory>
                            <gram:arguments>
                                ...
                            </gram:arguments>
                            <gram:environment>
                                ...
                            </gram:environment>
                            <gram:stdin>
                                ...
                            </gram:stdin>
                            <gram:stdout>
                                ...
                            </gram:stdout>
                            <gram:stderr>
                                ...
                            </gram:stderr>
```

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
                                        <gram:count>
                                            ...
                                        </gram:count>
                                        <gram:jobType>
                                            ...
                                        </gram:jobType>
                                        <gram:gramMyJobType>
                                            ...
                                        </gram:gramMyJobType>
                                        <gram:dryRun>
                                            ...
                                        </gram:dryRun>
                                    </gram:job>
                                </rsl:rsl>
                            </creationParameters>
                        </createService>
                    </from>
                    <to variable="input4CreateService" part="parameters" query="/k1001:createServiceInstance0"/>
                </copy>
            </assign>
            <invoke name="createService" partnerLink="GramProxy" portType="service:ManagedJobPortType" operation="createServiceInstance0"
            inputVariable="input4CreateService" outputVariable="output4CreateService"/>
```

รูปที่ 32 ตัวอย่างโค้ดกระบวนการบีเพลการเรียกใช้โอเปอร์เรชั่น createServiceInstance สำหรับแกรมกริดเซอร์วิส

จากรูปที่ 32 จะพบว่าเราทำการคัดลอกข้อมูลของงานที่ต้องการจะทำซึ่งอยู่ในรูปของอาร์เอสแอลนั้น ให้แก่ตัวแปรที่ชื่อ input4CreateService ซึ่งเป็นตัวแปรที่เป็นอินพุทในการเรียกใช้โอเปอร์เรชั่น createServiceInstance นั่นเอง

นอกจากนั้นในการใช้งานแกรมกริดเซอร์วิสโดยทั่วไปนั้น มีสิ่งสำคัญอีกอย่างหนึ่ง ก็คือ เราสามารถตรวจสอบสถานะของงานที่ส่งมาทำงานได้โดยอาศัยกลไกการแจ้งข่าวของ MJS ที่เราติดต่อด้วยอยู่ เราสามารถเขียนโค้ดเทียมสำหรับการเรียกใช้งานแกรมกริดเซอร์วิสได้ดังรูปที่ 33

```
scope{
        ext=RSL;
        proxy.createServiceInstance(ext);
        proxy.start();
        proxy.subscribeToProxyByName();
        wait(INFINITY);
}
OnMessage( notificationMessageFromProxy){
        jobStatus = query job status from notificationMessageFromProxy;
        If( jobStatus == .”Done” ) {
                proxy.destroy();
                terminate();
        }
}
```

รูปที่ 33 โค้ดเทียมของกระบวนการบีเพลสำหรับการเรียกใช้แกรมกริดเซอร์วิสแบบที่ 1

จากรูปที่ 33 กระบวนการบีเพลนี้ได้ทำการบอกรับการแจ้งข่าวไปยังตัวแทนความปลอดภัย จากนั้นจึงใช้กลไกการจัดการข้อความ (Message Handler) ในการรอรับข่าวที่จะส่งกลับมาจากตัวแทนความปลอดภัย ซึ่งในที่นี้ข่าวที่ได้รับก็คือ สถานะของงานที่กำลังทำอยู่ เมื่อข้อความแจ้งข่าว (Notification Message) ส่งกลับมายังกระบวนการบีเพลแล้ว กระบวนการบีเพลจะทำการเรียกข้อมูลสถานะงานออกมาจากข้อความที่ได้รับโดยใช้เอ็กซ์พาธ แล้วทำการเปรียบเทียบว่าสถานะงานนั้นมีค่าเท่ากับ Done หรือไม่ถ้าเท่ากัน ก็จะเรียกใช้โอเปอร์เรชั่น destroy เพื่อทำลายแกรมกริดเซอร์วิสอินสแตนท์ที่ใช้อยู่ แล้วจึงจบการทำงานของกระบวนการบีเพลนี้ เราสามารถแสดงตัวอย่างโค้ดบีเพลสำหรับการบอกรับการแจ้งข่าวไปยังตัวแทนความปลอดภัยได้ดังรูปที่ 34

```
<assign name="assign4SubscriptionByName">
    <copy>
        <from>
            <subscriberEndpointLocation xmlns="http://ws.zeus.com/proxy/notification/subscribe/generic">
                http://localhost:8082/ogsa/services/org/thaigrid/chemistry/portal/gridservices/gaussian/persistent
            </subscriberEndpointLocation>
        </from>
        <to variable="input4GenericSubscriptionByName" part="parameters"
        query="/generic:subscribeToProxyByName/generic:subscriberEndpointLocation"/>
    </copy>
    <copy>
        <from>
            <serviceDataName
            xmlns="http://ws.zeus.com/proxy/notification/subscribe/generic">PersistentGaussianData</serviceDataName>
        </from>
        <to variable="input4GenericSubscriptionByName" part="parameters"
        query="/generic:subscribeToProxyByName/generic:serviceDataName"/>
    </copy>
    <copy>
        <from>
            <sink xmlns="http://ws.zeus.com/proxy/notification/subscribe/generic">
                <handle>http://localhost:9700/orabpel/default/TestOrchestratingGeneratedGramProxyAtOGSA</handle>
            </sink>
        </from>
        <to variable="input4GenericSubscriptionByName" part="parameters" query="/generic:subscribeToProxyByName/generic:sink"/>
    </copy>
    <copy>
        <from>
            <expirationTime xmlns="http://ws.zeus.com/proxy/notification/subscribe/generic">
                2547-04-05T11:05:46.051Z
            </expirationTime>
        </from>
        <to variable="input4GenericSubscriptionByName" part="parameters"
        query="/generic:subscribeToProxyByName/generic:expirationTime"/>
    </copy>
</assign>
<invoke name="invokeGenericSubscriptionProxyByName" partnerLink="GenericNotificationProxy"
portType="generic:NotificationProxyPortType" operation="subscribeToProxyByName" inputVariable="input4GenericSubscriptionByName"
outputVariable="output4GenericSubscriptionByName">
    <correlations>
        <correlation set="correlationInteger" initiate="no" pattern="out"/>
    </correlations>
</invoke>
```

รูปที่ 34 โค้ดบีเพลสำหรับการบอกรับการแจ้งข่าวจากแกรมกริดเซอร์วิส

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

จากรูปที่ 34 มีจุดที่น่าสนใจ คือ เราทำการคัดลอกข้อมูลลงในอีเลเมนต์ที่ชื่อ sink ซึ่งเป็นส่วนหนึ่งของข้อความที่จะใช้ในการบอกรับการแจ้งข่าว โดยภายในอีเลเมนต์ sink นี้ เราจะต้องทำการระบุจุดหมายปลายทาง (Endpoint Location) ของกระบวนการบีเพลนี้ด้วย ตัวแทนความปลอดภัยของแกรมกริดเซอร์วิสจะจึงสามารถทำการส่งข้อความการแจ้งข่าวกลับมาได้อย่างถูกต้อง

แนวคิดในการเขียนกระบวนการบีเพลนี้ เมื่อได้ทำการสร้างและทดลองใช้งานจริงแล้วปรากฏว่าไม่สามารถทำงานได้อย่างถูกต้อง เนื่องมาจากกระบวนการบีเพลเรียกข้อมูลจากข้อความแจ้งข่าวที่ได้รับโดยใช้เอ็กซ์พาธได้อย่างถูกต้อง ทำให้ไม่สามารถทราบสถานะของงานที่กำลังทำอยู่ได้เลย ซึ่งคาดว่าอาจเกิดจากระบบทำงานบีเพลที่ใช้ในงานวิจัยนี้ คือ Oracle BPEL Process Manager ไม่รองรับการเรียกข้อมูลโดยใช้เอ็กซ์พาธ ในกรณีที่ไม่มีเอ็กซ์เอ็มแอลสกีมา (XML Schema) ของข้อความที่ส่งกลับมาจากตัวแทนความปลอดภัย

ทางแก้ปัญหาหนึ่งที่สามารถใช้ในการแก้ไขได้ คือ การเรียกดูสถานะของงานที่ส่งมาจากการเรียกใช้โอเปอร์เรชั่น findServiceData โดยการสั่งให้ค้นหาข้อมูลเซอร์วิสที่ชื่อ ManagedJobState ซึ่งในการทำงานกลไกการแจ้งข่าวของแกรมกริดเซอร์วิสนั้น จะมีการแจ้งข่าวกลับไปยังผู้ลงทะเบียนได้ก็ต่อเมื่อข้อมูลเซอร์วิสที่ชื่อ ManagedJobSatate มีการเปลี่ยนแปลงเช่นเดียวกัน ดังนั้นการเรียกดูข้อมูลเซอร์วิสที่ชื่อ ManagedJobState ก็จะได้สถานะของงานเช่นเดียวกับการรอรับข้อมูลแจ้งข่าว แต่เนื่องจากผลลัพธ์ที่ได้กลับมาจากโอเปอร์เรชั่น findServiceData จะอยู่ในรูปเอ็กซ์เอ็มแอลที่ซับซ้อนและไม่มีสกีมาแน่นอนเช่นเดียวกับเอ็กซ์เอ็มแอลที่ได้จากข้อความการแจ้งข่าว ทำให้อาจเกิดปัญหาในการใช้งานได้เช่นเดียวกัน ดังนั้น เราจะเสนอทางแก้ปัญหาโดยให้ผู้ใช้ทำการสร้างโอเปอร์เรชั่นเพิ่มเติมให้แก่ตัวแทนความปลอดภัยที่ตัวสร้างตัวแทนอัตโนมัติสร้างให้ที่ชื่อ retrieveManagedJobState โดยเมื่อมีผู้เรียกใช้โอเปอร์เรชั่นนี้จะทำการค้นหาข้อมูลเซอร์วิสที่ชื่อ ManagedJobState ให้โดยอัตโนมัติจากนั้นจะทำการคืนค่าสถานะของงานส่งกลับมาแต่เพียงอย่างเดียว ทำให้สามารถนำผลลัพธ์ที่ได้ไปจัดการต่อได้โดยง่ายโดยไม่ต้องคำนึงถึงปัญหาการเรียกข้อมูลเอ็กซ์เอ็มแอลโดยใช้เอ็กซ์พาธตามที่ได้กล่าวมาแล้ว เราสามารถแสดงดับบลิวเอสดีแอลเฉพาะของโอเปอร์เรชั่น retrieveManagedJobState ได้รูปที่ 35

```
<wsdl:definitions …>
    <wsdl:types ..>
        <schema …>
        …
            <element name="retrieveManagedJobState">
                <complexType />
            </element>
            <element name="retrieveManagedJobStateResponse">
                <complexType>
                    <sequence>
                        <element name="retrieveManagedJobStateReturn" type="xsd:string" />
                    </sequence>
                </complexType>
            </element>
        </schema>
    </wsdl:types>
    …
    <wsdl:message name="retrieveManagedJobStateResponse">
        <wsdl:part element="tns1:retrieveManagedJobStateResponse" name="parameters" />
    </wsdl:message>
    <wsdl:message name="retrieveManagedJobStateRequest">
        <wsdl:part element="tns1:retrieveManagedJobState" name="parameters" />
    </wsdl:message>
    …
    <wsdl:portType name="ManagedJobPortType">
        <wsdl:operation name="retrieveManagedJobState">
            <wsdl:input message="impl:retrieveManagedJobStateRequest" name="retrieveManagedJobStateRequest" />
            <wsdl:output message="impl:retrieveManagedJobStateResponse" name="retrieveManagedJobStateResponse" />
            <wsdl:fault message="impl:ExtensibilityTypeFaultType" name="ExtensibilityTypeFaultType" />
            <wsdl:fault message="impl:GSSException" name="GSSException" />
            <wsdl:fault message="impl:TargetInvalidFaultType" name="TargetInvalidFaultType" />
            <wsdl:fault message="impl:ExtensibilityNotSupportedFaultType" name="ExtensibilityNotSupportedFaultType" />
            <wsdl:fault message="impl:FaultType1" name="FaultType" />
            <wsdl:fault message="impl:MyProxyException" name="MyProxyException" />
        </wsdl:operation>
    <wsdl:portType>
    …
<wsdl:definitions>
```

รูปที่ 35 ดับบลิวเอสดีแอลของโอเปอร์เรชั่น retrieveManagedJobState

ผลจากการสร้างโอเปอร์เรชั่น retrieveManagedJobStatus ในการเรียกดูสถานะของงานแทนที่การใช้กลไกการแจ้งข่าว ทำให้การเขียนโค้ดบีเพลเพื่อเรียกใช้แกรมกริดเซอร์วิสจะต้องมีการเปลี่ยนแปลง โดยเราจะต้องใช้การวนลูปเพื่อคอยเรียกดูสถานะของงานจนกว่าสถานะของงาน

จะเป็น Done แทนที่การใช้กลไกการจัดการข้อความของบีเพล ซึ่งในการวนลูปแต่ละรอบจะต้องมีการรอเป็นเวลาที่เหมาะสมเพื่อไม่ให้กระบวนการบีเพลทำงานตลอดเวลาอันจะเป็นการสิ้นเปลืองทรัพยากรของเครื่องที่ระบบทำงานบีเพลทำงานอยู่ เราสามารถแสดงโค้ดเทียมของกระบวนการบีเพลเพื่อเรียกใช้งานแกรมกริดเซอร์วิสเมื่อมีการใช้โอเปอร์เรชั่น retrieveManagedJobStatus ได้ดังรูปที่ 36

```
scope{
        ext=RSL;
        proxy.createServiceInstance(ext);
        proxy.start();
        while(true){
                wait(XXX);
                jobStatus=proxy.retrieveManagedJobStatus;
                If( jobStatus == “Done”){
                        proxy.destroy;
                        client.callback();
                        Terminate();
                }
        }
}
```

รูปที่ 36 โค้ดเทียมสำหรับกระบวนการบีเพลที่มีการเรียกใช้งานแกรมกริดเซอร์วิสที่สามารถทำงานได้จริง

เราสามารถแสดงโค้ดบีเพลสำหรับการสั่งให้แกรมกริดเซอร์วิสอินสแตนท์เริ่มทำงานจริง การวนลูปเพื่อเรียกดูสถานะของงาน การทำลายแกรมกริดเซอร์วิสอินสแตนท์และการจบการทำงานของกระบวนการบีเพลได้ดังรูปที่ 37

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<invoke name="start" partnerLink="GramProxy" portType="service:ManagedJobPortType" operation="start0" inputVariable="input4Start"
outputVariable="output4Start"/>
        <while condition="true()">
    <scope name="while-scope">
        <sequence>
            <wait for="PT5S" name="wait-1"/>
            <invoke name="retrieveManagedJobState" partnerLink="GramProxy" portType="service:ManagedJobPortType"
            operation="retrieveManagedJobState0" inputVariable="input4RetrieveJobState" outputVariable="output4RetrieveJobState"/>
            <switch>
                <case condition
="bpws:getVariableData('output4RetrieveJobState','parameters','/k1001:retrieveManagedJobState0Response/k1001:retrieveManagedJobState0Return')
='Done'">
                    <sequence name="switch-scope">
                        <invoke name="destroy" partnerLink="GramProxy" portType="service:ManagedJobPortType" operation="destroy0"
                            inputVariable="input4Destroy" outputVariable="output4Destroy"/>
                        <invoke name="callbackClient" partnerLink="client"
                        portType="tns:TestOrchestratingGeneratedGramProxyAtOGSACallback" operation="onResult"
                        inputVariable="output"/>
                        <terminate/>
                    </sequence>
                </case>
            </switch>
        </sequence>
    </scope>
</while>
```

รูปที่ 37 โค้ดบีเพลสำหรับการสั่งให้แกรมกริดเซอร์วิสอินสแตนท์เริ่มทำงาน การวนลูปตรวจสอบสถานะงานและการทำลายแกรมกริดเซอร์วิสอินสแตนท์

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 6

สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

วิทยานิพนธ์ชิ้นนี้ได้นำเสนอกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส โดยนำเสนอแนวคิดของชั้นประสานการทำงานกริดเซอร์วิส ที่ทำหน้าที่เป็นตัวเชื่อมโยงระหว่างกริดเซอร์วิสกับระบบกระแสงานบีเพล ทำให้สามารถนำระบบกระแสงานบีเพลที่แต่เดิมถูกออกแบบมาให้ใช้กับเทคโนโลยีเว็บเซอร์วิส มาใช้ในการประสานการทำงานระหว่างกริดเซอร์วิสต่างๆ ได้อย่างง่ายดาย กรอบการประสานการทำงานที่สร้างขึ้นนั้นสามารถรองรับการทำงานได้ทั้งกับกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยและกริดเซอร์วิสที่มีการรักษาความปลอดภัย

กรอบการประสานการทำงานที่นำเสนอนี้ประกอบไปด้วยส่วนประกอบสำคัญ 3 ส่วน ได้แก่ เว็บเซอร์วิสผู้ช่วย เครื่องมือปรับปรุงดับบลิวเอสดีแอล และตัวแทนความปลอดภัย เพื่อแก้ไขปัญหาต่างๆที่เกิดขึ้น ในการนำบีเพลมาใช้ประสานการทำงานร่วมกันของกริดเซอร์วิส อันได้แก่

- การอ้างอิงอินสแตนท์ที่ต่างกัน
- การเชื่อมต่อแบบพลวัฒน์
- การใช้งานกริดเซอร์วิสอินสแตนท์
- คุณสมบัติการสืบทอดพอร์ทไทป์

โดยเราสามารถแสดงตารางสรุปปัญหาที่ส่วนประกอบต่างๆของกรอบการประสานการทำงานร่วมกันที่เราสร้างขึ้นได้ดังตารางที่ 6

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ตารางที่ 6 ปัญหาที่ส่วนประกอบต่างๆของกรอบการประสานการทำงานร่วมกันสามารถแก้ไขได้

ส่วนประกอบ	ทำงานร่วมกับกริดเซอร์วิส		ปัญหาที่แก้ไขได้			
	ไม่มีการรักษาความปลอดภัย	มีการรักษาความปลอดภัย	การอ้างอิงอินสแตนท์ที่ต่างกัน	การเชื่อมต่อแบบพลวัฒน์	การใช้งานกริดเซอร์วิสอินสแตนท์	คุณสมบัติการสืบทอดพอร์ทไทป์
เว็บเซอร์วิสผู้ช่วย	✓		✓	✓	✓	
เครื่องมือปรับปรุงดับบลิวเอสดีแอล	✓				✓	✓
ตัวแทนความปลอดภัย		✓	✓	✓	✓	✓

การใช้งานส่วนประกอบทั้งสามร่วมกับการสร้างเอกสารกระบวนการบีเพลตามแนวทางที่กำหนดในวิทยานิพนธ์นี้ จะทำให้เราสามารถใช้กระบวนการบีเพลประสานการทำงานของกริดเซอร์วิส โดยการเรียกใช้กลไกและคุณสมบัติต่างๆของกริดเซอร์วิสได้อย่างครบถ้วน และสามารถเรียกใช้กริดเซอร์วิสได้ทั้งกริดเซอร์วิสที่ผู้ใช้สร้างขึ้นเองและกริดเซอร์วิสที่กำหนดขึ้นในโกลบัสทูลคิท เช่น แกรมกริดเซอร์วิส ที่ใช้ในการจัดการการเข้าใช้ทรัพยากร เป็นต้น

6.2 อภิปรายผลการวิจัย

กรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสที่นำเสนอในวิทยานิพนธ์นี้นั้น สามารถประสานการทำงานระหว่างกริดเซอร์วิสต่างๆ ได้เป็นอย่างดี แต่จากการออกแบบทำให้กรอบการทำงานที่เราสร้างขึ้นมีข้อที่ควรคำนึงในการใช้งานบางประการ ได้แก่

- ผู้ใช้ยังจำเป็นจะต้องทำการสร้างเว็บเซอร์วิสของตัวแทนความปลอดภัยด้วยตนเอง
- ตัวแทนความปลอดภัยจะทำงานในลักษณะ 1 ต่อ 1 เท่านั้น คือ กริดเซอร์วิสอินสแตนท์หรือกริดเซอร์วิสแบบถาวร 1 ตัว จะต้องมีตัวแทนความปลอดภัยที่ทำหน้าที่เป็นตัวกลางในการติดต่อ 1 เซอร์วิสเสมอ ทำให้อาจเกิดปัญหาได้เมื่อมีต้องการสร้างตัวแทนความปลอดภัยของกริดเซอร์วิสเป็นจำนวนมาก

นอกจากนั้น การนำข้อกำหนดบีเพลมาใช้งานร่วมกับกริดเซอร์วิสนั้น ยังแสดงให้เห็นถึงความไม่สมมูลกันระหว่างกริดเซอร์วิสกับเทคโนโลยีเว็บเซอร์วิส ในหลายด้านที่สำคัญ ได้แก่

- ลักษณะการอ้างอิงอินสแตนท์ ซึ่งกริดเซอร์วิสใช้กลไกจีเอสเอชในการอ้างอิงถึงอินสแตนท์ที่ต่างกัน แต่เว็บเซอร์วิสจะมีการแทรกสิ่งที่ใช้ในการระบุอินสแตนท์ที่ต้องการติดต่อลงในข้อความหรือในส่วนหัวของข้อความโซป
- กลไกความปลอดภัยที่ใช้ในกริดเซอร์วิส มีการทำงานในรูปแบบที่เป็นของตนเอง ซึ่งยังไม่มีการนำมาตรฐานทางด้านความปลอดภัยของเว็บเซอร์วิสมาใช้หรือใช้เพียงบางส่วน ซึ่งอาจเกิดจากการยังไม่ได้รับการยอมรับของมาตรฐานทางด้านความปลอดภัยของเว็บเซอร์วิสก็เป็นได้

6.3 ข้อเสนอแนะ

- พัฒนาตัวสร้างตัวแทนอัตโนมัติ ให้สามารถสร้างตัวแทนความปลอดภัยที่มีคุณสมบัติครบถ้วนตามที่ระบุในไฟล์โครงแบบความปลอดภัยที่กำหนดขึ้นในวิทยานิพนธ์นี้ โดยผู้ใช้ไม่ต้องทำการแก้ไขไฟล์ตัวแทนความปลอดภัยใดๆทั้งสิ้น
- พัฒนาตัวสร้างตัวแทนอัตโนมัติ ให้สามารถสร้างเว็บเซอร์วิสตัวแทนความปลอดภัยได้โดยอัตโนมัติ โดยผู้ใช้ไม่ต้องทำการสร้างเว็บเซอร์วิสจากไฟล์ที่ตัวสร้างตัวแทนอัตโนมัติด้วยตนเองอีกต่อไป
- ปรับปรุงรูปแบบของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส ในกรณีที่อาจจะมีระบบทำงานบีเพลที่รองรับมาตรฐาน WS-Security ได้โดยตรง
- ศึกษาแนวทางในการใช้งานกริดเซอร์วิสจากกระบวนการบีเพลในกรณีที่กริดเซอร์วิสรองรับข้อกำหนด WS-Addressing

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

รายการอ้างอิง

[1] I. Foster, "The anatomy of the grid: enabling scalable virtual organizations," presented at Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on, 2001.

[2] K. Amin, G. von Laszewski, M. Hategan, N. J. Zaluzec, S. Hampton, and A. Rossi, "GridAnt: a client-controllable grid workflow system," presented at Proceedings of the 37th Annual Hawaii International Conference on System Sciences, 2004.

[3] "Java CoG Kit User Manual Version 4.0-pre-alpha," Argonne National Laboratory, Manual 2004.

[4] A. Adiga, "GridFlow & GridSteer : Globus-Based Tools for Scientific Visualisation," Presentation 2004.

[5] Y. Gil, E. Deelman, J. Blythe, C. Kesselman, and H. Tangmunarunkit, "Artificial intelligence and grids: workflow planning and beyond," Intelligent Systems, IEEE, vol. 19, pp. 26-33, 2004.

[6] S. Krishnan, P. Wagstrom, and G. v. Laszewski, "GSFL: A Workflow Framework for Grid Services," Argonne National Labaratory ANL/MCS-P980-0802, 2002.

[7] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Levmann, K. Liu, D. Roller, D. Smith, S. Thattle, I. Trickovic, and S. Weerawarana, "Business Process Execution Language for Web Services V. 1.1," 2003.

[8] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "The Physiology of the Grid : An Open Grid Services Architecture for Distributed Systems Integration," 6/22/2002 2002.

[9] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, and P. Vanderbilt, "Open Grid Services Infrastructure Version 1.0," 2004.

[10] "WS GRAM Documentation", available from: http://www-unix.globus.org/toolkit/docs/3.2/gram/ws/index.html.

[11] "GSI: Grid Security Infrastructure", available from: http://www-unix.globus.org/toolkit/docs/3.2/gsi/index.html.

[12] "Monitoring & Discovery System (MDS)", available from: http://www-unix.globus.org/toolkit/mds/.

[13] "The Globus Toolkit", available from: http://www-unix.globus.org/toolkit/.

[14] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, "Web Services Description Language (WSDL) 1.1", available from: http://www.w3.org/TR/wsdl.

[15] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. E. Nielsen, S. Thatte, and D. Winer, "Simple Object Access Protocol (SOAP) 1.1", available from: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[16] B. Atkinson, G. Della-Libera, S. Hada, M. Hondo, P. Hallam-Baker, J. Klein, B. LaMacchia, P. Leach, J. Manferdelli, H. Maruyama, A. Nadalin, N. Nagaratnam, H. Prafullchandra, J. Shewchuk, and D. Simon, "Web Services Security (WS-Security)", available from: http://www-106.ibm.com/developerworks/webservices/library/ws-secure/.

[17] "XML Signature", available from: http://www.w3.org/TR/xmldsig-core/.

[18] B. Wilkinson and M. Allen, Parallel Programming: Alan Apt, 1999.

[19] F. Leymann, "Web Services Flow Language (WSFL 1.0)," 2004.

[20] "XLANG", available from: http://www.ebpml.org/xlang.htm.

[21] "XML Schema", available from: http://www.w3.org/XML/Schema.

[22] J. Clark and S. DeRose, "XML Path Language (XPath) version 1.0," 1999.

[23] D. Box, E. Christensen, F. Curbera, D. Ferguson, J. Frey, M. Hardley, C. Kaler, D. Langworthy, F. Leymann, B. Lovering, S. Lucco, S. Millet, N. Mukhi, M.

Nottingham, D. Orchard, J. Shewchuk, E. Sindambiwe, T. Storey, S. Weerawarana, and S. Winkler, "Web Services Addressing (WS-Addressing)," 2004.

[24] B. Butchart and W. Emmerich, "Orchestration of an OGSI-enabled scientific application using the Business Process Execution Language," University College London, Presentation 2004.

[25] T. Bellwood, L. Clement, and C. v. Riegen, "UDDI Version 3.0.1 UDDI Spec Technical Committee Specification", available from: http://uddi.org/pubs/uddi_v3.htm.

[26] A. Slomiski, "On Using BPEL Extensibility to Implement OGSI and WSRF Grid Workflows," Technical Report 2004.

[27] J. Novotny, S. Tuecke, and V. Welch, "An online credential repository for the Grid: MyProxy," presented at High Performance Distributed Computing, 2001. Proceedings. 10th IEEE International Symposium on, 2001.

[28] "Apache Axis", available from: http://ws.apache.org/axis/.

[29] "Autodock - Automated Docking of Flexible Ligands to Molecules," 2005.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ก

สกีมาไฟล์ของไฟล์โครงแบบความปลอดภัยที่ใช้กับตัวสร้างตัวแทนอัตโนมัติ

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://gs.zeus.com/security/config" xmlns:tns="http://gs.zeus.com/security/config">
	<complexType name="tProtection-level">
		<choice>
			<element name="privacy"/>
			<element name="integrity"/>
		</choice>
	</complexType>
	<complexType name="tGsi">
		<sequence>
			<element name="protection-level" type="tns:tProtection-level"/>
		</sequence>
	</complexType>
	<complexType name="tPkey">
		<sequence>
			<element name="protection-level" type="tns:tProtection-level"/>
		</sequence>
	</complexType>
	<complexType name="tAuth-method">
		<sequence>
			<choice>
				<element name="gsi" type="tns:tGsi"/>
				<element name="pkey" type="tns:tPkey"/>
				<element name="none"/>
			</choice>
		</sequence>
	</complexType>
	<complexType name="tMethod">
		<sequence>
			<element name="auth-method" type="tns:tAuth-method"/>
		</sequence>
		<attribute name="name" type="xs:QName"/>
	</complexType>
	<complexType name="tGrim">
```

```
                <attribute name="idenity" type="string"/>
        </complexType>
        <complexType name="tGrim-policy-handling">
                <sequence>
                        <choice>
                                <element name="grim" type="tns:tGrim"/>
                                <element name="ignore"/>
                        </choice>
                </sequence>
        </complexType>
        <complexType name="tCredential-delegation">
                <choice>
                        <element name="no"/>
                        <element name="limited"/>
                        <element name="full"/>
                </choice>
        </complexType>
        <complexType name="tCredential">
                <sequence>
                        <element name="host" type="string"/>
                        <element name="username" type="string"/>
                        <element name="password" type="string"/>
                </sequence>
                <attribute name="type" type="string"/>
        </complexType>
        <complexType name="tSecurityConfig">
        <sequence>
                <element name="method" type="tns:tMethod" minOccurs="0" maxOccurs="unbounded"/>
                <element name="auth-method" type="tns:tAuth-method" minOccurs="0" maxOccurs="1"/>
                <element name="grim-policy-handling" type="tns:tGrim-policy-handling" minOccurs="0"
maxOccurs="1"/>
                <element name="credential-delegation" type="tns:tCredential-delegation"/>
                <element name="credential" type="tns:tCredential"/>
        </sequence>
        </complexType>
        <element name="securityConfig" type="tns:tSecurityConfig"/>
</xs:schema>
```

ภาคผนวก ข

การเขียนเอกสารกระบวนการบีเพลเพื่อเรียกใช้กลไกการแจ้งข่าว

การเขียนเอกสารกระบวนการบีเพลเพื่อทำให้กระบวนการบีเพลสามารถทำหน้าที่เป็นแหล่งปลายทางการแจ้งข่าวในกลไกการแจ้งข่าวนั้น มีขั้นตอนดังนี้

1. สร้างดับบลิวเอสดีแอลของกระบวนการบีเพลที่ต้องการทำหน้าที่เป็นแหล่งปลายทางการแจ้งข่าวให้มีส่วนประกอบดังต่อไปนี้

```
<definitions …
    xmlns:defBpelNotif=”http://bpel.zeus.com/notification/sink/default”
    xmlns:cor="http://bpel.zeus.com/test/notification/ProcessCorrelation"
    xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    >
    <types>
    …
        <schema targetNamespace="http://bpel.zeus.com/notification/sink/default"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:defBpelNotif="http://bpel.zeus.com/notification/sink/default"
            xmlns="http://www.w3.org/2001/XMLSchema">
            <complexType name="BPELExtensibilityType">
                <sequence>
                    <any namespace="##any"/>
                </sequence>
            </complexType>
            <element name="deliverNotificationFromProxy">
                <complexType>
                    <sequence>
                        <element name="correlationInterger" type="xsd:int"/>
                        <element name="message"
            type="defBpelNotif:BPELExtensibilityType"/>
                    </sequence>
                </complexType>
            </element>
        </schema>
    </types>
    …
    <bpws:property name="correlationInteger" type="xsd:int"/>
    …
    <message name="DeliverNotificationFromProxyInputMessage">
        <part name="parameters" element="defBpelNotif:deliverNotificationFromProxy"/>
    </message>
    …
```

```
    <bpws:propertyAlias propertyName="cor:correlationInteger"  messageType="tns:DeliverNotificationFromProxyInputMessage"
    part="parameters"        query="/defBpelNotif:deliverNotificationFromProxy/defBpelNotif:correlationInteger"/>
    …
    <portType name="DefaultNotificationSinkPortType">
          <operation name="deliverNotificationFromProxy">
               <input message="tns:DeliverNotificationFromProxyInputMessage"/>
          </operation>
     </portType>
    …
</definitions>
```

รูปที่ 38 สิ่งที่ต้องเพิ่มเติมลงในดับบลิวเอสดีแอลของกระบวนการบีเพล

จากรูปที่ 38 จะพบว่า สิ่งที่เราจะต้องทำการเพิ่มลงไปในดับบลิวเอสดีแอลจะประกอบไปด้วย

- ส่วนประกาศสกีมา (schema) ทำหน้าที่ประกาศสกีมาของข้อความ (Message) ที่เราจะใช้ในการรับข้อความที่ส่งมาจากแหล่งต้นทางการแจ้งข่าว ในที่นี้คือ อีเลเมนต์ที่ชื่อ deliverNotificationFromProxy
- ส่วนประกาศข้อความ (message) ทำหน้าที่ประกาศชนิดของข้อความที่จะใช้ในการรับข้อความที่ส่งมาจากแหล่งต้นทางการแจ้งข่าว ในที่นี้คือ ข้อความที่ชื่อ DeliverNotificationFromProxyInputMessage
- ส่วนประกาศพอร์ทไทป์และโอเปอร์เรชั่น (portType & operation) ทำหน้าที่ประกาศโอเปอร์เรชั่นและพอร์ทไทป์ใหม่ที่จะใช้เป็นโอเปอร์เรชั่นสำหรับเว็บเซอร์วิสผู้ช่วยและตัวแทนความปลอดภัยในการส่งข้อความการแจ้งข่าวกลับมายังกระบวนการบีเพล ในที่นี้เราทำการประกาศโอเปอร์เรชั่นที่ชื่อ deliverNotificationFromProxy ของพอร์ทไทป์ DefaultNotificationSinkPortType
- ส่วนประกาศคุณสมบัติ (property) ทำหน้าที่ประกาศคุณสมบัติที่จะใช้เป็น correlationSet ในการรับข้อความการแจ้งข่าว ซึ่งจะถูกนำไปใช้ในการกำหนด correlationSet ภายในเอกสารกระบวนการบีเพลต่อไป ในที่นี้เราทำการประกาศคุณสมบัติที่ชื่อ correlationInteger
- ส่วนประกาศนามแฝงคุณสมบัติ (propertyAlias) ทำหน้าที่ประกาศความเกี่ยวพันระหว่างคุณสมบัติกับอีเลเมนต์ของข้อความ เพื่อที่จะใช้อีเลเมนต์นั้นเป็น correlationSet ในที่นี้เราประกาศให้ใช้อีเลเมนต์ที่ชื่อ correlationInteger ของ

ข้อความที่ชื่อ deliverNotificationFromProxy มีค่าเท่ากับคุณสมบัติที่ชื่อ correlationInteger

2. สร้างเอกสารกระบวนการบีเพลตามปรกติ โดยจะต้องมีการประกาศตัวระบุความสัมพันธ์ (Correlation Sets) ที่ชื่อ correlationInteger ด้วย ให้มีคุณสมบัติเป็น correlationInteger ซึ่งคุณสมบัตินี้เป็นสิ่งที่เรากำหนดไว้ในดับบลิวเอสดีแอลของกระบวนการนี้นั่นเอง เราสามารถแสดงการประกาศตัวระบุความสัมพันธ์ได้ดังรูปที่ 39

```
<process …>
    …
    <correlationSets>
        <correlationSet name="correlationInteger" properties="cor:correlationInteger"/>
    </correlationSets>
    …
</scope>
```

รูปที่ 39 ตัวอย่างการประกาศใช้ตัวระบุความสัมพันธ์ในเอกสารกระบวนการบีเพล

จากนั้นเมื่อต้องการรับข้อความการแจ้งข่าวจากแหล่งต้นทางการแจ้งข่าวสามารถทำได้ 2 รูปแบบ คือ

- การรอรับข้อความโดยใช้อีเลเมนต์ pick การเขียนเอกสารกระบวนการบีเพลแบบนี้เมื่อกระบวนการบีเพลทำงานมาถึงอีเลเมนต์ pick กระบวนการจะหยุดรอจนกว่ามีข้อความตามที่กำหนดไว้ในอีเลเมนต์ pick เข้ามาถึง เราสามารถแสดงตัวอย่างของเอกสารกระบวนการบีเพลตามรูปแบบนี้ได้ดังรูปที่ 40

```
<sequence>
    …
    <pick name="pick-1">
        <onMessage partnerLink="client" portType="tns: DefaultNotificationSinkPortType "
    operation="deliverNotificationFromProxy" variable="inputFromNotificaitonProxy">
            <correlations>
                <correlation set="correlationInteger" initiate="no"/>
            </correlations>
            <sequence>
                …
            </sequence>
        </onMessage>
    </pick>
    …
</sequence>
```

รูปที่ 40 ตัวอย่างเอกสารกระบวนการบีเพลที่ต้องการรับข้อความการแจ้งข่าวโดยใช้อีเลเมนต์ pick

จากรูปที่ 40 นั้นจะสังเกตได้ว่าภายในอีเลเมนต์ pick จะประกอบไปด้วยอีเลเมนต์ภายใน (Nested Element) คือ อีเลเมนต์ onMessage ซึ่งเป็นการกำหนดว่าอีเลเมนต์ pick นี้จะทำงานเมื่อมีข้อความใดเข้ามา ในที่นี้เรากำหนดให้ รอรับข้อความของโอเปอร์เรชั่นที่ชื่อ deliverNotificationFromProxy ของพอร์ทไทป์ที่ชื่อ DefaultNotificationSinkPortType ซึ่งเราได้กำหนดไว้ในดับบลิวเอสดีแอลของกระบวนการนี้ในขั้นตอนที่ผ่านมานั่นเอง

สิ่งที่น่าสนใจก็คือ ภายในอีเลเมนต์ onMesage นั้นจะมี element ภายในที่น่าสนใจ คือ อีเลเมนต์ที่ชื่อ correlations และอีเลเมนต์ sequence สำหรับอีเลเมนต์ correlations นั้นเป็นการกำหนดว่าในการรับข้อความนั้น จะใช้คุณสมบัติตัวใดทำหน้าที่เป็นตัวระบุความสัมพันธ์ (Correlation Set) ซึ่งตัวระบุความสัมพันธ์นี้ยังใช้เป็นตัวอ้างอิงในการระบุอินสแตนท์ของกระบวนการบีเพลที่จะติดต่อด้วย ดังที่ได้กล่าวมาแล้ว ในที่นี้เรากำหนดให้ใช้คุณสมบัติที่ชื่อ correlationInteger ซึ่งเป็นคุณสมบัติที่เราได้กำหนดไว้ในขั้นตอนที่การสร้างดับบลิวเอสดีแอลของกระบวนการนี้นั่นเอง สำหรับอีเลเมนต์ sequence นั้น ใช้สำหรับการกำหนดกิจกรรมที่จะทำเมื่อกระบวนการได้รับข้อความนั้น ซึ่งขึ้นอยู่กับผู้ออกแบบกระบวนการบีเพล

- การรอรับข้อความโดยใช้อีเลเมนต์ eventHandlers การทำงานแบบนี้จะทำงานในรูปแบบคล้ายกับตัวดักฟังเหตุการณ์ (Event Listener) ในภาษาเชิงวัตถุที่นิยม โดยอีเลเมนต์ eventHandlers นี้จะต้องอยู่ในภายในอีเลเมนต์ที่ชื่อ scope ซึ่งใช้สำหรับกำหนดขอบเขตของการทำงานใดๆ ของกระบวนการบีเพลนั่นเอง เราสามารถแสดงตัวอย่างของการรอรับข้อความโดยใช้อีเลเมนต์ eventHandlers ได้ดังรูปที่ 41

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<scope name="mainScope">
    ...
    <eventHandlers>
        <onMessage partnerLink="client" portType="tns: DefaultNotificationSinkPortType "
    operation="deliverNotificationFromProxy" variable="inputFromNotificaitonProxy">
            <correlations>
                <correlation set="correlationInteger" initiate="no"/>
            </correlations>
            <sequence>
                ...
            </sequence>
        </onMessage>
    </eventHandlers>
    ...
</sequence>
```

รูปที่ 41 ตัวอย่างเอกสารกระบวนการบีเพลที่ต้องการรอรับข้อความโดยใช้อีเลเมนต์ eventHandlers

จากรูปที่ 41 เราจะพบว่าภายในอีเลเมนต์ eventHandlers นั้นจะมีอีเลเมนต์ภายในที่เหมือนกับอีเลเมนต์ภายในของอีเลเมนต์ pick คือ มีอีเลเมนต์ onMessage และอีเลเมนต์ sequence เนื่องมาจากทั้งคู่มีหลักการทำงานที่เหมือนกัน แต่สิ่งที่ต่างกันก็คือ การใช้งานในรูปแบบอีเลเมนต์ eventHandlers นั้น จะเป็นอิสระจากกิจกรรมที่กระบวนการบีเพลทำอยู่ หมายถึงกระบวนการบีเพลไม่ต้องหยุดรอรับข้อความเหมือนกับในกรณีของการใช้อีเลเมนต์ pick กล่าวคือเมื่อกระบวนการได้รับข้อความที่ต้องการแล้วก็จะทำงานตามที่กำหนดไว้ในอีเลเมนต์ eventHandlers ได้ไม่สนใจว่ากระบวนการหลักกำลังทำงานใดอยู่

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ค

ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย

ในการพัฒนากระบวนการบีเพลเพื่อเรียกใช้กริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัยนั้น เพื่อที่จะสามารถเรียกใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสเหล่านั้นได้อย่างเต็มที่นั้น จำเป็นจะต้องอาศัยส่วนต่างๆของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิส อันได้แก่ เว็บเซอร์วิสผู้ช่วย และเครื่องมือปรับปรุงดับบลิวเอสดีแอล โดยสามารถสรุปเป็นขั้นตอนต่างๆ ได้ดังต่อไปนี้

1. ค้นหาข้อมูลของกริดเซอร์วิสที่ต้องการจะเรียกใช้งาน ซึ่งในขั้นนี้จะมีความแตกต่างที่สำคัญจากการเรียกใช้งานเว็บเซอร์วิส คือ ข้อมูลของกริดเซอร์วิสที่จะเรียกใช้งานนั้น จะอยู่ในรูปของจีเอสเอช ไม่ใช่ดับบลิวเอสดีแอลอย่างในกรณีของเว็บเซอร์วิส สำหรับจีเอสเอชที่เราจะต้องทราบนั้น แบ่งออกเป็น 2 กรณี ย่อย คือ
   - ถ้าเราต้องการจะเรียกใช้กริดเซอร์วิสที่เป็นชนิดถาวร หรือ กริดเซอร์วิสอินสแตนท์ที่มีการสร้างไว้แล้ว เราจะต้องทราบจีเอสเอชของกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่สร้างขึ้นแล้วนั้น
   - ถ้าเราต้องการจะเรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง และต้องการจะสร้างขึ้นใหม่ เราจำเป็นต้องทราบจีเอสเอชของกริดเซอร์วิสที่ทำหน้าที่เป็นโรงงานเซอร์วิสสำหรับสร้างกริดเซอร์วิสอินสแตนท์ที่ต้องการนั้น
2. ทำการสร้างดับบลิวเอสดีแอลของกริดเซอร์วิสที่ต้องการใช้งาน โดยใช้เครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ โดยระบุจีเอสเอชตามกรณีย่อยที่ได้กล่าวในหัวข้อที่แล้ว พร้อมทั้งจะต้องระบุชื่อของกริดเซอร์วิสอินสแตนท์ที่เราต้องการด้วย ซึ่งถ้าเป็นในกรณีของการสร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่มีอยู่แล้วให้ระบุชื่อของกริดเซอร์วิสอินสแตนท์เป็น persistent เราสามารถแสดงตัวอย่างการสร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริงและเราต้องการสร้างขึ้นใหม่ โดยค่าจีเอสเอชของโรงงานเซอร์วิสสำหรับกริดเซอร์วิสอินสแตนท์นี้ คือ
   http://127.0.0.1:8082/ogsa/services/org/thaigrid/chemistry/portal/gridservices/autodock/transient/instance ได้ดังรูปที่ 42

```
>GWSDLAutomaticGenerator
http://127.0.0.1:8082/ogsa/services/org/thaigrid/chemistry/portal/gridservices/autodock/transient/instance
autodockX AutodockX.gwsdl
```

รูปที่ 42 ตัวอย่างการใช้งานเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติสำหรับกริดเซอร์วิสแบบถาวร

จากตัวอย่างเป็นเราสร้างจีดับบลิวเอสดีแอลของกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริง โดยให้ชื่อของกริดเซอร์วิสอินสแตนท์ที่จะเกิดขึ้นภายหลังว่า autodockX โดยจีดับบลิวเอสดีแอลที่ได้จะอยู่ในไฟล์ที่ชื่อ AutodockX.gwsdl

3. นำจีดับบลิวเอสดีแอลของกริดเซอร์วิสที่เราต้องการเรียกใช้ที่สร้างขึ้นจากเครื่องมือสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติ มาผ่านเครื่องมือในการแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอลและเครื่องมือปรับปรุงบีเพลให้เหมาะสมกับการใช้งานกับบีเพล เพื่อให้ได้ดับบลิวเอสดีแอลของกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่สามารถใช้ในการพัฒนาเอกสารกระแสงานบีเพลได้ เราสามารถแสดงตัวอย่างการใช้เครื่องมือทั้งสองได้ดังรูปที่ 43

```
>GWSDL2WSDL AutodockX.gwsdl AutodockX.wsdl
>WSDL4BPEL AutodockX.wsdl AutodockX-ready.wsdl
```

รูปที่ 43ตัวอย่างการใช้งานเครื่องมือแปลงจีดับบลิวเอสดีแอลเป็นดับบลิวเอสดีแอลและเครื่องมือปรับปรุงดับบลิวเอสดีแอลให้เหมาะสมกับบีเพล

จากรูปที่ 43 ในบรรทัดแรกเราจะแปลงจีดับบลิวเอสดีแอลที่อยู่ในไฟล์ AutodockX.gwsdl ไปเป็นดับบลิวเอสดีแอล โดยนำผลลัพธ์ที่ได้ไปไว้ในไฟล์ AutodockX.wsdl จากนั้นจึงปรับปรุงดับบลิวเอสดีแอลในไฟล์ AutodockX.wsdl ให้เหมาะสมกับบีเพล โดยผลลัพธ์ที่ได้จะอยู่ในไฟล์ AutodockX-ready,wsdl

4. พัฒนาเอกสารกระบวนบีเพลเพื่อเรียกใช้งานโอเปอร์เรชั่นต่างๆของกริดเซอร์วิสที่ต้องการโดยใช้ดับบลิวเอสดีแอลที่ผ่านการปรับปรุงจากเครื่องมือต่างๆแล้ว โดยมีข้อพิจารณาในการพัฒนาดังนี้
   - ถ้าเป็นการใช้งานกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง จะต้องเรียกใช้โอเปอร์เรชั่น createGridServiceInstance ของเว็บเซอร์วิสผู้ช่วยเพื่อทำการ

สร้างกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริงนั้นขึ้น โดยจะต้องระบุชื่อของกริดเซอร์วิสอินสแตนท์ให้ตรงกับที่ได้ระบุในขั้นตอนการสร้างจีดับบลิวเอสดีแอลแบบอัตโนมัติก่อนเสมอ แต่ถ้าเป็นกรณีการเรียกใช้

- ในการเรียกใช้โอเปอร์เรชั่นต่างๆที่ผู้ใช้สร้างขึ้นเองรวมไปถึงโอเปอร์เรชั่นมาตรฐานของกริดเซอร์วิสตามข้อกำหนดโอจีเอสไอ ยกเว้นโอเปอร์เรชั่น subscribe ซึ่งเป็นโอเปอร์เรชั่นมาตรฐานสำหรับการบอกรับการแจ้งข่าวตามข้อกำหนดโอจีเอสไอ สามารถเรียกใช้งานได้โดยตรง

ในการใช้งานกลไกการแจ้งข่าวให้เรียกใช้โอเปอร์เรชั่นสำหรับการบอกรับการแจ้งข่าวของเว็บเซอร์วิสผู้ช่วย อันได้แก่ โอเปอร์เรชั่น subscribeToHelper และ subscribeToHelperByName เพื่อบอกรับการแจ้งข่าวจากกริดเซอร์วิสที่ต้องการ และพัฒนากระบวนการบีเพลตามแนวทางการเขียนกระบวนการบีเพลเพื่อเรียกใช้งานกลไกการแจ้งข่าวที่ได้กล่าวมาแล้ว

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาคผนวก ง

ขั้นตอนการพัฒนากระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย

ในการพัฒนากระบวนการบีเพลเพื่อเรียกใช้กริดเซอร์วิสที่มีการรักษาความปลอดภัยนั้น เพื่อที่จะสามารถเรียกใช้งานกลไกและคุณสมบัติต่างๆของกริดเซอร์วิสเหล่านั้นได้อย่างเต็มที่นั้น เราจะต้องอาศัยส่วนประกอบที่สำคัญของกรอบการประสานการทำงานร่วมกันของบีเพลสำหรับกริดเซอร์วิสสำหรับการทำงานร่วมกับกริดเซอร์วิสที่มีการรักษาความปลอดภัย คือ ตัวแทนความปลอดภัย โดยสามารถสรุปเป็นขั้นตอนต่างๆ ได้ดังต่อไปนี้

1. ค้นหาข้อมูลของกริดเซอร์วิสที่ต้องการจะเรียกใช้งาน โดยจะต้องทราบจีเอสเอชของกริดเซอร์วิสที่ต้องการจะใช้งาน ซึ่งสามารถแบ่งออกได้เป็น 2 กรณีย่อย เช่นเดียวกับการเรียกใช้งานกริดเซอร์วิสที่ไม่มีการรักษาความปลอดภัย คือ
   - ถ้าเราต้องการจะเรียกใช้กริดเซอร์วิสที่เป็นชนิดถาวร หรือ กริดเซอร์วิสอินสแตนท์ที่มีการสร้างไว้แล้ว เราจะต้องทราบจีเอสเอชของกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่สร้างขึ้นแล้วนั้น
   - ถ้าเราต้องการจะเรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง และต้องการจะสร้างขึ้นใหม่ เราจำเป็นต้องทราบจีเอสเอชของกริดเซอร์วิสที่ทำหน้าที่เป็นโรงงานเซอร์วิสสำหรับสร้างกริดเซอร์วิสอินสแตนท์ที่ต้องการนั้น
2. ศึกษาถึงรูปแบบการรักษาความปลอดภัยของกริดเซอร์วิสที่เราต้องกาจะเรียกใช้ จากนั้นจึงสร้างไฟล์โครงแบบความปลอดภัยที่อธิบายถึงความปลอดภัยที่เกี่ยวข้องกับกริดเซอร์วิสทั้งหมดสำหรับกริดเซอร์วิสนั้น ตามรูปแบบไฟล์โครงแบบความปลอดภัยที่ได้กล่าวมาแล้ว ซึ่งสามารถแบ่งได้เป็น 2 กรณีย่อย คือ
   - ถ้าเราต้องการเรียกใช้งานกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์ที่มีการสร้างไว้ เราจะต้องสร้างไฟล์โครงแบบความปลอดภัยที่ระบุถึงความปลอดภัยของกริดเซอร์วิสแบบถาวรหรือกริดเซอร์วิสอินสแตนท์นั้น
   - ถ้าเราต้องการเรียกใช้งานกริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง เราจะต้องสร้างไฟล์โครงแบบความปลอดภัยที่ระบุถึงความปลอดภัยของโรงงานเซอร์วิสของกริดเซอร์วิสอินสแตนท์ที่เราต้องการจะสร้าง นอกจากนั้นจะต้อง

สร้างไฟล์โครงแบบความปลอดภัยที่ระบุถึงความปลอดภัยของกริดเซอร์วิสอินสแตนท์ที่เราต้องการจะสร้างด้วย

3. ใช้ตัวสร้างตัวแทนอัตโนมัติเพื่อสร้างตัวแทนความปลอดภัยเพื่อทำหน้าที่ติดต่อกับกริดเซอร์วิสที่ต้องการ โดยเลือกใช้ตัวสร้างตัวแทนอัตโนมัติให้เหมาะสมกับชนิดของกริดเซอร์วิสที่ต้องการจะสร้างตัวแทน เราสามารถแสดงตัวอย่างการใช้งานตัวสร้างตัวแทนอัตโนมัติได้ดังรูปที่ 44

```
>ProxyGeneratorForAGridServiceInstance
http://localhost:8082/ogsa/services/org/thaigrid/chemistry/portal/gridservices/autodock/transient/instance
TransientAutodockInstance-factory-security-config.xml TransientAutodockInstance-instance-security-
config.xml
```

รูปที่ 44 ตัวอย่างการเรียกใช้งานตัวสร้างตัวแทนอัตโนมัติสำหรับกริดเซอร์วิสอินสแตนท์ที่ไม่มีอยู่จริง

จากรูปที่ 44 เราทำการสร้างตัวแทนอัตโนมัติสำหรับกริดเซอร์วิสอินสแตนท์ ที่มีจีเอสเอชของโรงงานเซอร์วิสสำหรับกริดเซอร์วิสอินสแตนท์นี้ คือ http://localhost:8082/ogsa/services/org/thaigrid/chemistry/portal/gridservices/autodock/transient/instance โดยจะเห็นว่าเราจะต้องระบุถึงไฟล์เอ็กซ์เอ็มแอล 2 ไฟล์ซึ่งเป็นไฟล์โครงแบบความปลอดภัยของโรงงงานเซอร์วิสและของกริดเซอร์วิสอินสแตนท์นั้นเอง ตามลำดับ

4. ติดตั้งตัวแทนความปลอดภัยจากไฟล์ที่ได้จากตัวสร้างตัวแทนอัตโนมัติ ตามขั้นตอนที่ระบุในหัวข้อ 3.3.3.3 จะทำให้เราได้เว็บเซอร์วิสใหม่ที่ทำหน้าที่เป็นตัวแทนความปลอดภัยในการติดต่อกับกริดเซอร์วิสที่ต้องการนี้
5. พัฒนาเอกสารกระบวนการบีเพลเพื่อเรียกใช้งานกริดเซอร์วิสที่มีการรักษาความปลอดภัย โดยใช้ดับบลิวเอสดีแอลของเว็บเซอร์วิสที่ทำหน้าที่เป็นตัวแทนความปลอดภัยที่สร้างขึ้น โดยมีข้อพิจารณาในการพัฒนาดังนี้
   - ในกรณีที่เป็นการเรียกใช้กริดเซอร์วิสอินสแตนท์ที่ยังไม่มีอยู่จริง ผู้ใช้จะต้องเรียกโอเปอร์เรชั่นสำหรับสร้างกริดเซอร์วิสอินสแตนท์ (createServiceInstance operation) ที่มีอยู่ในตัวแทนความปลอดภัยก่อนเพื่อสร้างให้เกิดกริดเซอร์วิสอินสแตนท์นั้นขึ้นจริงก่อนเสมอ

- การเรียกใช้โอเปอร์เรชั่นต่างๆ ของกริดเซอร์วิสที่มีการรักษาความปลอดภัย ทั้งโอเปอร์เรชั่นที่กำหนดขึ้นเอง และโอเปอร์เรชั่นที่มีตามข้อกำหนดโอจีเอส ไอ ให้เรียกผ่านตัวแทนความปลอดภัยทั้งหมด

ในการใช้งานกลไกการแจ้งข่าวให้เรียกใช้โอเปอร์เรชั่นสำหรับการบอกรับการแจ้งข่าวของตัวแทนความปลอดภัย อันได้แก่ โอเปอร์เรชั่น subscribeToProxy และ subscribeToProxyByName เพื่อบอกรับการแจ้งข่าวจากกริดเซอร์วิสที่ต้องการ และพัฒนากระบวนการบีเพลตามแนวทางการเขียนกระบวนการบีเพลเพื่อเรียกใช้งานกลไกการแจ้งข่าวที่ได้กล่าวมาแล้ว

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ประวัติผู้เขียนวิทยานิพนธ์

นายพิเชษฐ์ อำนวยกาญจนสิน เกิดเมื่อวันที่ 23 ธันวาคม 2524 ที่จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2545 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาตรมหาบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2546

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย