การออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์

นาย บรรจง ห่วงรัตน์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2546

ISBN 974-17-5569-4

THE DESIGN AND DEVELOPMENT OF A CHANGE NOTIFICATION SYSTEM OF REMOTE SERVICES VIA TRADER FEDERATION

Mr.Banjong Huongrat

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2003

ISBN 974-17-5569-4

หัวข้อวิทยานิพนธ์	การออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์
โดย	นาย บรรจง ห่วงรัตน์
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

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

คณะกรรมการสอบวิทยานิพนธ์

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

.....อาจารย์ที่ปรึกษา
(ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา)

.....กรรมการ
(อาจารย์ จารุมาตร ปิ่นทอง)

.....กรรมการ
(อาจารย์ ดร.ณัฐวุฒิ หนูไพโรจน์)

บรรจง ห่วงรัตน์ : การออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ (THE DESIGN AND DEVELOPMENT OF A CHANGE NOTIFICATION SYSTEM OF REMOTE SERVICES VIA TRADER FEDERATION) อ.ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา, 125 หน้า. ISBN 974-17-5569-4.

งานวิจัยนี้ได้ทำการออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ โดยสร้างส่วนขยายเพิ่มเติมให้กับบริการเทรดเดอร์ เพื่อสนับสนุนการทำงานของระบบแจ้งการเปลี่ยนแปลงบริการเอสซีเอ็นเดิม ให้สามารถรองรับการแจ้งการเปลี่ยนแปลงของบริการระยะไกลได้ โดยให้ระบบแจ้งการเปลี่ยนแปลงแต่ละระบบ ซึ่งต่างก็เชื่อมต่อกับเทรดเดอร์คนละตัว สามารถส่งต่อรายการเปลี่ยนแปลงที่เกิดขึ้นในระบบของตนไปให้ระบบอื่นๆ ได้ โดยอาศัยกลไกการเชื่อมต่อของเทรดเดอร์เหล่านั้น การส่งต่อจะพิจารณาจากสัญญาข้อตกลงที่แต่ละเทรดเดอร์ทำร่วมกัน โดยหากบริการใดมีการเปลี่ยนแปลงและบริการนั้นได้รับการส่งออกไปยังเทรดเดอร์อื่น การเปลี่ยนแปลงนี้จะถูกส่งต่อไปแจ้งยังผู้รับบริการในระบบอื่นนั้นได้

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษา .....

ปีการศึกษา 2546

4371440521 : MAJOR COMPUTER SCIENCE

KEY WORD : CHANGE NOTIFICATION / TRADER FEDERATION / FEDERATION CONTRACT

BANJONG HUONGRAT: THE DESIGN AND DEVELOPMENT OF A CHANGE NOTIFICATION SYSTEM OF REMOTE SERVICES VIA TRADER FEDERATION. THESIS ADVISOR: ASSIST. PROF DR. TWITTIE SENIVONGSE, 125 pp. ISBN 974-17-5569-4.

This research proposes the design and development of a Service Change Notification Service (SCN) for remote services via trader federation which extends from the existing Service Change Notification Service to support change notification of remote services. In the system, each of instance of the SCN will connect to a local trader and will send notification of change of local services to remote sites that are federated with the local trader. Remote notification is performed according to federation contracts among the federated traders. That is, if a local service is changed and that service is exported to other traders, change notification will be delivered to remote clients.

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

Field of study Computer Science Advisor's signature .....

Academic year 2003

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความช่วยเหลืออย่างดียิ่งจาก ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งท่านได้ให้คำแนะนำต่างๆ รวมทั้งให้ข้อคิดเห็นในงานวิจัยด้วยดีเสมอ

ขอขอบคุณ อาจารย์ ดร.ยรรยง เต็งอำนวย อาจารย์ ดร. ณัฐวุฒิ หนุไพโรจน์ และอาจารย์ จารุมาตร ปิ่นทอง กรรมการวิทยานิพนธ์ ที่กรุณาให้คำแนะนำและตรวจสอบแก้ไขต้นฉบับวิทยานิพนธ์นี้

ขอขอบคุณ คุณภาสิน สุริเยนทรากร ที่ให้ความช่วยเหลือในการอธิบายระบบแจ้งการเปลี่ยนแปลง [6] และช่วยแก้ปัญหาที่เกิดจากข้อผิดพลาดของโปรแกรม

ขอขอบคุณ เพื่อนๆ พี่ๆ และน้องๆ ทั้งหลาย ที่คอยให้กำลังใจ และให้ความช่วยเหลือ รวมทั้งให้ความสนุกสนาน และความบันเทิงมาโดยตลอด

สุดท้ายนี้ ผู้วิจัยใคร่ขอกราบขอบพระคุณ บิดา มารดา รวมถึงทุกคนในครอบครัว ที่ให้การสนับสนุนในด้านต่างๆ และให้กำลังใจแก่ผู้วิจัยเสมอมาจนสำเร็จการศึกษา

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

สารบัญรูปภาพ (ต่อ)

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในระบบกระจายคอร์บานั้นเมื่อระบบมีขนาดใหญ่ขึ้นและมีบริการชนิดต่างๆ ปรากฏในบริการเทรดเดอร์มากขึ้น ทำให้การค้นหาบริการไม่ค่อยมีประสิทธิภาพ จึงมีความจำเป็นต้องแยกชนิดของบริการไปไว้ในเทรดเดอร์ตัวอื่น โดยแบ่งเทรดเดอร์ออกเป็นกลุ่มตามหน่วยงานหรือชนิดของบริการ ดังนั้นจำเป็นต้องมีการเชื่อมต่อเทรดเดอร์หลายๆ ตัวเข้าด้วยกัน (Trader Federation[3]) ทำให้ผู้รับบริการไม่จำเป็นต้องติดต่อกับเทรดเดอร์มากกว่าหนึ่งตัวเพื่อค้นหาบริการ แต่สามารถค้นหาบริการที่ต้องการได้โดยอ้อมโดยการร้องขอบริการผ่านเทรดเดอร์ที่กำลังติดต่อไปยังเทรดเดอร์ปลายทางที่มีการเชื่อมต่อกัน

ในการเชื่อมต่อเทรดเดอร์เพื่อแลกเปลี่ยนข้อมูลระหว่างกันนั้น ผู้ดูแลเทรดเดอร์นำเข้า (Importing Trader) และผู้ดูแลเทรดเดอร์ส่งออก (Exporting Trader) จะมีการสร้างสัญญาข้อตกลง (Contract) ระหว่างกันขึ้นเพื่อใช้ในการค้นหาบริการ [5] โดยผู้ดูแลเทรดเดอร์ส่งออกจะทำการสร้างบัญชีรายชื่อ (Catalogue) ของบริการที่ต้องการโฆษณาขึ้น แล้วจัดส่งบัญชีรายชื่อดังกล่าวไปให้ผู้ดูแลเทรดเดอร์นำเข้า ผู้ดูแลเทรดเดอร์นำเข้าจะทำการเลือกบริการที่ตนเองต้องการ แล้วส่งกลับไปยังผู้ดูแลเทรดเดอร์ส่งออกเพื่อพิจารณา จากนั้นจะมีการสร้างสัญญาข้อตกลงระหว่างสองฝ่ายเพื่อทำการแลกเปลี่ยนข้อมูลระหว่างกัน ได้แก่ สัญญานำเข้า (Import-Contract) ซึ่งจะเก็บไว้ที่เทรดเดอร์นำเข้า และสัญญาส่งออก (Export-Contract) ซึ่งจะเก็บไว้ที่เทรดเดอร์ส่งออก สัญญานี้จะใช้เป็นข้อกำหนดในการค้นหาบริการ ซึ่งประกาศไว้ที่เทรดเดอร์ตัวอื่น และเมื่อมีการปรับปรุงหรือเปลี่ยนแปลงรายการในสัญญาข้อตกลงหรือมีการยกเลิกสัญญาข้อตกลงเกิดขึ้น จำเป็นต้องมีการแจ้งการเปลี่ยนแปลงให้ผู้รับบริการที่ต้องการทราบด้วย

จากงานวิจัย [6] ได้สร้างส่วนขยายบริการเทรดเดอร์ในระบบคอร์บาเพื่อทำการตรวจจับการเปลี่ยนแปลงของบริการที่เกิดขึ้น และแจ้งการเปลี่ยนแปลงบริการแบบต่างๆ ไปยังผู้รับบริการเมื่อเกิดการเปลี่ยนแปลงขึ้น หรือแจ้งล่วงหน้าเพื่อให้ผู้รับบริการรับทราบการเปลี่ยนแปลงที่จะเกิดขึ้นในอนาคต รูปแบบการเปลี่ยนแปลงที่รองรับ ได้แก่ การเปลี่ยนแปลงชนิดของบริการ การเปลี่ยนแปลงค่าคุณสมบัติของบริการ การยกเลิกการใช้งานบริการ และการเกิดบริการขึ้นใหม่ ระบบดังกล่าวได้ถูกพัฒนาขึ้นมาเพื่อใช้งานกับเทรดเดอร์เพียงตัวเดียวเท่านั้น ไม่สามารถนำไปใช้เพื่อแจ้งการเปลี่ยนแปลงกับระบบที่มีบริการเทรดเดอร์มากกว่าหนึ่งตัวที่มีการเชื่อมต่อกันได้ เนื่องจากผู้รับบริการต้องทำการลงทะเบียนกับระบบแจ้งเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์ตัวหนึ่ง เพื่อขอทราบการเปลี่ยนแปลงบริการที่อยู่ในเทรดเดอร์ตัวนั้น หากผู้รับบริการได้เคยใช้บริการของเทรดเดอร์

อื่นที่ได้มาจากการค้นหาผ่านสัญญาข้อตกลงและต้องการรับทราบการเปลี่ยนแปลงของบริการนี้ จะยังไม่สามารถทำได้ เนื่องจากระบบแจ้งเหตุการณ์ยังไม่มีกลไกในการรับการส่งต่อการแจ้งเตือนจากระบบแจ้งเหตุการณ์อื่นที่ทำงานร่วมกับเทรดเดอร์อื่น

จากปัญหาดังกล่าวผู้วิจัยจึงมีแนวคิดที่จะออกแบบโครงสร้างและพัฒนาต้นแบบสำหรับการเชื่อมต่อกันของระบบแจ้งการเปลี่ยนแปลงบริการผ่านการเชื่อมต่อระหว่างเทรดเดอร์ขึ้น โดยอาศัยสัญญาข้อตกลงในการแลกเปลี่ยนข้อมูลระหว่างเทรดเดอร์ เพื่อเป็นข้อมูลช่วยในการตรวจสอบการแจ้งการเปลี่ยนแปลงของบริการ

1.2 วัตถุประสงค์ของการวิจัย

เพื่อออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ เพื่อแจ้งเตือนเหตุการณ์ไปยังผู้รับบริการข้ามระบบได้

1.3 ขอบเขตของการวิจัย

1. ต้นแบบที่พัฒนาและทดสอบจะเป็นการทำงานร่วมกันของระบบแจ้งการเปลี่ยนแปลง 2 ระบบ โดยแต่ละระบบทำงานร่วมกับเทรดเดอร์ 1 ตัว ซึ่งเทรดเดอร์จะเชื่อมต่อกัน
2. บริการเทรดเดอร์ที่จะทำการเพิ่มขยายและบริการแจ้งเหตุการณ์ที่จะนำมาใช้ จะอยู่ภายใต้ข้อกำหนดของคอร์บารุ่นปรับปรุงที่ 2.2 เป็นอย่างน้อย

1.4 ขั้นตอนในการดำเนินงาน

1. ศึกษาทฤษฎีของระบบกระจายคอร์บาและบริการต่างๆ ได้แก่ บริการเทรดเดอร์ บริการแจ้งเหตุการณ์
2. ศึกษารูปแบบการเปลี่ยนแปลงต่างๆ ของบริการภายในเทรดเดอร์ที่มีการเชื่อมต่อกัน
3. ศึกษาระบบแจ้งเตือนการเปลี่ยนแปลงในงานวิจัย [3]
4. ศึกษาและทดลองเครื่องมือต่างๆ ที่จะนำมาใช้ในการพัฒนา
5. ออกแบบการแจ้งการเปลี่ยนแปลงของบริการระยะไกลโดยอาศัยเทรดเดอร์ที่มีการเชื่อมต่อกัน
6. พัฒนากลไกเพื่อการแจ้งการเปลี่ยนแปลงของบริการระยะไกลโดยอาศัยเทรดเดอร์ที่มีการเชื่อมต่อกัน
7. ทดสอบและปรับปรุงแก้ไขระบบ
8. วิเคราะห์และสรุปผล พร้อมข้อเสนอแนะ
9. จัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ได้ระบบแจ้งการเปลี่ยนแปลงบริการที่สามารถส่งต่อการแจ้งเตือนระหว่างกันได้ โดยผ่านการเชื่อมต่อกันของเทรดเดอร์

1.6 ผลงานตีพิมพ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตอบรับให้นำเสนอในงานประชุมวิชาการระดับนานาชาติดังนี้

1. The 3rd International Symposium on Information and Communication Technology (ISICT'04) June 16 - 18, 2004, Las Vegas, USA ในบทความเรื่อง Cross-Domain Service Change Notification via Trading Federation

โดยผู้แต่งคือ Banjong Huongrat and Twittie Senivongse

1.7 โครงสร้างวิทยานิพนธ์

ในบทต่อไปของวิทยานิพนธ์นี้จะกล่าวถึงทฤษฎีที่นำมาประยุกต์ใช้และงานวิจัยที่เกี่ยวข้อง ส่วนในบทที่ 3 จะกล่าวถึงแนวคิดในการออกแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ ในบทที่ 4 จะกล่าวถึงต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ที่ได้พัฒนาขึ้น ในบทที่ 5 จะกล่าวถึงรายละเอียดต่างๆ ในการทดสอบและผลการทดสอบการจำลองการทำงานของการเปลี่ยนแปลงบริการผ่านสัญญาข้อตกลง ด้วยการสร้าง/แก้ไข/ยกเลิก ชนิดบริการหรือข้อเสนอบริการในสัญญาข้อตกลง และในบทสุดท้ายจะเป็นการสรุปผลของงานวิทยานิพนธ์และข้อเสนอแนะในการพัฒนาต่อไป

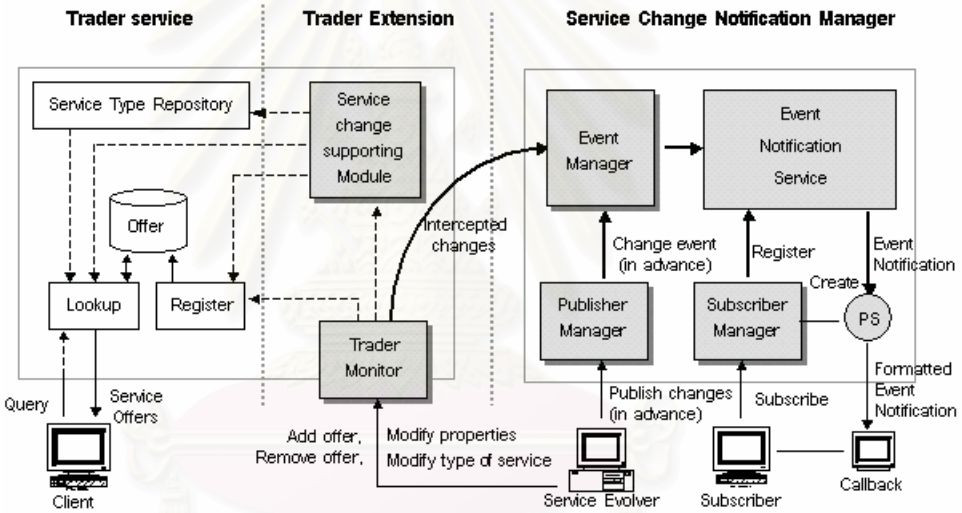
บทที่ 2

งานวิจัยและทฤษฎีที่เกี่ยวข้อง

2.1 งานวิจัยที่เกี่ยวข้อง

2.1.1. Design and Development of A Change Notification System of Distributed Service [6][7]

งานวิจัยนี้ได้พัฒนาต้นแบบบริการสำหรับแจ้งการเปลี่ยนแปลงบริการในระบบกระจายคอร์บา (รูปที่ 2.1) เพื่ออำนวยความสะดวกแก่ผู้รับบริการในกรณีที่เกิดเหตุการณ์การเปลี่ยนแปลงของบริการในกรณีต่างๆ ซึ่งผู้วิจัยจะได้ใช้ต้นแบบนี้เป็นองค์ประกอบในการแจ้งเตือนการเปลี่ยนแปลงบริการระยะไกลด้วย รายละเอียดของต้นแบบนี้มีดังนี้



รูปที่ 2.1 สถาปัตยกรรมระบบแจ้งการเปลี่ยนแปลงบริการ

2.1.1.1. เหตุการณ์การเปลี่ยนแปลงบริการที่พิจารณา

เหตุการณ์การเปลี่ยนแปลงบริการที่ได้รับการพิจารณา มีด้วยกัน 4 ประเภท คือ

<u>เหตุการณ์ที่ 1 บริการได้รับการเปลี่ยนแปลงชนิดของบริการ</u>

บริการจะได้รับการเปลี่ยนแปลงชนิดของบริการก็ต่อเมื่อบริการนั้นได้รับการเปลี่ยนแปลงส่วนต่อประสานของบริการ เปลี่ยนแปลงรายการคุณสมบัติของบริการ และ (หรือ) เปลี่ยนแปลงรายการชนิดของบริการที่สืบทอด

<u>เหตุการณ์ที่ 2 บริการได้รับการเปลี่ยนแปลงค่าคุณสมบัติของบริการ</u>

การเปลี่ยนแปลงค่าคุณสมบัติของบริการ อาจไม่ส่งผลกระทบต่อการทำงานของผู้รับบริการหรืออาจทำให้ผู้รับบริการทำงานต่อไม่ได้ก็เป็นได้ ทั้งนี้ขึ้นอยู่กับวิธีการค้นหาและการใช้งานบริการของผู้รับบริการแต่ละราย

เหตุการณ์ที่ 3 บริการถูกยกเลิกการใช้งาน

การยกเลิกการใช้งานบริการส่งผลกระทบโดยตรงต่อรับบริการอย่างหลีกเลี่ยงไม่ได้ การแจ้งเตือนการยกเลิกบริการ โดยเฉพาะการแจ้งเตือนล่วงหน้าจะทำให้ผู้รับบริการสามารถที่จะเตรียมการรองรับได้ทัน

เหตุการณ์ที่ 4 เกิดบริการใหม่ขึ้นในระบบ

การเกิดบริการใหม่ขึ้นในระบบไม่ส่งผลกระทบต่อการใช้งานบริการเดิมแต่อย่างไร แต่เป็นการแจ้งเพื่อให้ผู้รับบริการทราบถึงทางเลือกใหม่ที่เพิ่มขึ้นของการให้บริการ

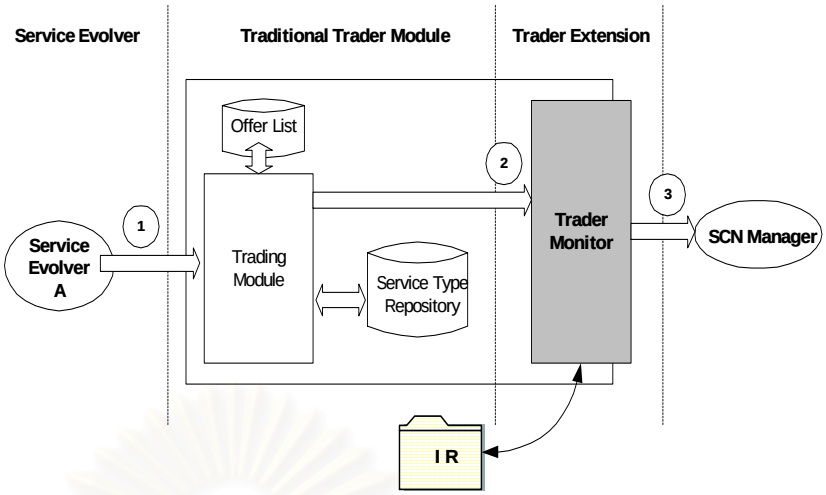
ระบบแจ้งการเปลี่ยนแปลงจากรูปที่ 2.1 เรียกว่า บริการเอสซีเอ็น (SCN Service: Service Change Notification Service) โดยบริการเอสซีเอ็นนี้จะรองรับการแจ้งการเปลี่ยนแปลงบริการทั้งการแจ้งการเปลี่ยนแปลงล่วงหน้า และการแจ้งการเปลี่ยนแปลงในขณะที่บริการได้รับการเปลี่ยนแปลงดังได้กล่าวไปแล้ว บริการเอสซีเอ็นสนับสนุนการแจ้งการเปลี่ยนแปลงบริการในรูปแบบที่แตกต่างกันตามความต้องการของผู้รับบริการที่ต้องการรับทราบการเปลี่ยนแปลง ทั้งรูปแบบของการจัดส่งและชนิดของข้อมูลบริการ บริการเอสซีเอ็น ประกอบด้วย 2 ส่วนที่สำคัญ คือ ส่วนเพิ่มขยายบริการเทรดเดอร์ (Trader Extension) และตัวจัดการการแจ้งการเปลี่ยนแปลงบริการ (SCN Manager: Service Change Notification Manager) โดยอธิบายได้ดังนี้

2.1.1.2. **ส่วนเพิ่มขยายบริการเทรดเดอร์** (Trader Extension)

ส่วนเพิ่มขยายบริการเทรดเดอร์ทำหน้าที่หลักในการตรวจสอบการเปลี่ยนแปลงของบริการโดยอัตโนมัติในขณะที่เกิดการเปลี่ยนแปลงบริการ โดยแบ่งได้เป็น 2 ส่วนด้วยกัน คือ

2.1.1.2.1.**ตัวตรวจบริการเทรดเดอร์** (Trader Monitor)

ตัวตรวจบริการเทรดเดอร์ทำงานร่วมกับบริการเทรดเดอร์ในลักษณะของคอร์บาอินเทอร์เซ็พเตอร์เพื่อใช้ในการตรวจสอบการใช้งานตัวกระทำการ (Operation) ต่างๆ ที่เกี่ยวข้องกับการเปลี่ยนแปลงข้อมูลของบริการ ผลที่ได้จากการตรวจสอบคือ ข้อมูลการเปลี่ยนแปลงบริการในขณะที่บริการได้รับการเปลี่ยนแปลง



รูปที่ 2.2 โครงสร้างการทำงานของตัวตรวจบริการเทรดเดอร์

จากรูปที่ 2.2 อธิบายการทำงานของตัวตรวจบริการได้ดังนี้

ขั้นตอนที่ 1 ผู้เปลี่ยนแปลงบริการเรียกใช้งานตัวกระทำการต่างๆ ที่เกี่ยวข้องกับการเปลี่ยนแปลงข้อมูลของบริการบนบริการเทรดเดอร์

ขั้นตอนที่ 2 ข้อมูลการเรียกใช้งานตัวกระทำการต่างๆ เหล่านั้นจะถูกตรวจจับโดยตัวตรวจบริการเทรดเดอร์ ซึ่งคอยเฝ้ามองการเรียกใช้งานตัวกระทำการเหล่านั้นอยู่

ขั้นตอนที่ 3 ตัวตรวจบริการเทรดเดอร์ทำการส่งข้อมูลการเปลี่ยนแปลงบริการที่ตรวจจับได้ไปยังตัวจัดการการแจ้งการเปลี่ยนแปลงบริการ เพื่อทำการแจ้งไปยังผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการต่อไป

ตัวตรวจบริการเทรดเดอร์จะทำการตรวจจับการเรียกใช้งานตัวกระทำการของบริการเทรดเดอร์ (ตามข้อกำหนดของ [3]) เฉพาะที่เกี่ยวข้องกับการเปลี่ยนแปลงข้อมูลของบริการเท่านั้น ตัวกระทำการเหล่านั้น ได้แก่

1. <u>ตัวกระทำการเพิ่มบริการ</u> มีส่วนต่อประสานดังนี้

```
OfferId export {
        in Object reference,
        in ServiceTypeName type,
        in PropertySeq properites
};
```

2. <u>ตัวกระทำการยกเลิกบริการ</u> มีส่วนต่อประสานดังนี้

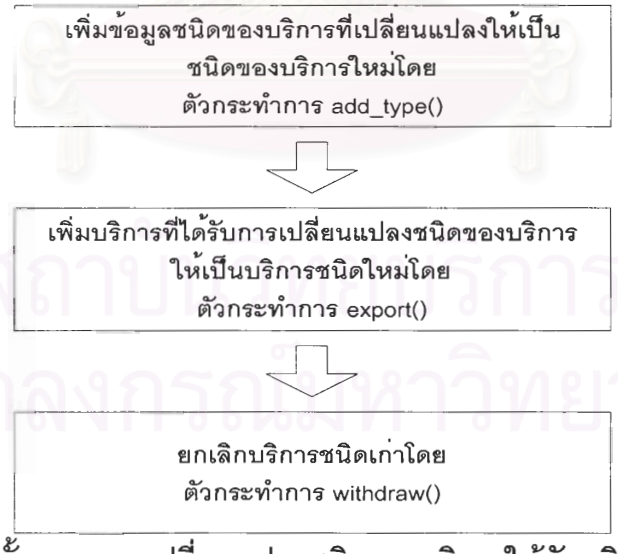
```
void withdraw { in OfferId id };
```

3. ตัวกระทำการเปลี่ยนแปลงค่าคุณสมบัติของบริการ มีส่วนต่อประสานดังนี้

```
void modify {
        in OfferId id,
        in PropertyNameSeq del_list,
        in PropertySeq modify_list
};
```

2.1.1.2.2. **มอดูลสนับสนุนการเปลี่ยนแปลงบริการ** (Service Change Supporting Module)

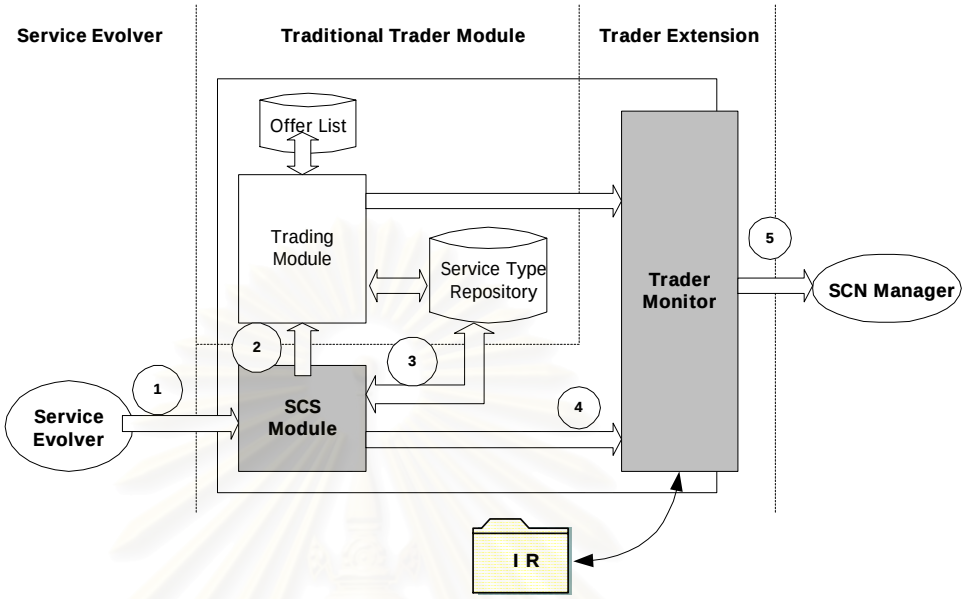
ตัวตรวจบริการเทรดเดอร์สามารถตรวจจับเหตุการณ์การเปลี่ยนแปลงบริการในหัวข้อที่ 2.1.1.1 ได้จากการดักการเรียกใช้ตัวกระทำการข้างต้นของบริการเทรดเดอร์ แต่จะไม่สามารถตรวจจับเหตุการณ์การเปลี่ยนแปลงชนิดของบริการได้ ทั้งนี้เนื่องจากการเปลี่ยนแปลงข้อมูลชนิดของบริการให้กับบริการหนึ่งๆ นั้นไม่สามารถกระทำได้ด้วยตัวกระทำการเดียวโดยตรง แต่จะประกอบด้วยชุดของตัวกระทำการของบริการเทรดเดอร์ดังแสดงในรูปที่ 2.3 ดังนั้นตัวตรวจบริการเทรดเดอร์จึงไม่สามารถแยกแยะได้ว่าการกระทำต่างๆ ดังกล่าวเป็นขั้นตอนของการเปลี่ยนชนิดของบริการให้กับบริการหนึ่งๆ หรือเป็นการเพิ่มและยกเลิกบริการที่ไม่เกี่ยวข้องกัน และเพื่อให้ตัวตรวจบริการเทรดเดอร์สามารถแยกแยะได้ ในงานวิจัย [6] จึงทำการเพิ่มมอดูลสนับสนุนการเปลี่ยนแปลงบริการเข้าไปในบริการเทรดเดอร์



รูปที่ 2.3 ขั้นตอนการเปลี่ยนแปลงชนิดของบริการให้กับบริการหนึ่งๆ

มอดูลสนับสนุนการเปลี่ยนแปลงบริการมีวัตถุประสงค์หลัก เพื่อเพิ่มตัวกระทำการสำหรับการเปลี่ยนชนิดของบริการ ซึ่งจะทำให้ตัวตรวจบริการเทรดเดอร์สามารถตรวจจับการ

เปลี่ยนแปลงชนิดของบริการหนึ่งๆ ได้จากการดักการเรียกใช้ตัวกระทำการที่เพิ่มขึ้นนี้ การทำงานของตัวกระทำการเปลี่ยนชนิดของบริการสามารถแสดงได้ดังรูปที่ 2.4 โดยอธิบายเป็นขั้นตอนได้ดังนี้



รูปที่ 2.4 โครงสร้างการทำงานของส่วนมอดูลสนับสนุนการเปลี่ยนแปลงบริการ

ขั้นตอนที่ 1 ผู้เปลี่ยนแปลงบริการจะทำการเรียกใช้งานตัวกระทำการ เพื่อขอเปลี่ยนแปลงชนิดของบริการให้กับบริการหนึ่งๆ

ขั้นตอนที่ 2,3 เป็นการทำงานภายในของตัวกระทำการที่มีการติดต่อขอเรียกใช้งานตัวกระทำการในมอดูลมาตรฐานของบริการเทรดเดอร์

ขั้นตอนที่ 4 ตัวตรวจบริการเทรดเดอร์จะตรวจพบการเรียกใช้งานตัวกระทำการ

ขั้นตอนที่ 5 ตัวตรวจบริการเทรดเดอร์จะทำการส่งข้อมูลการเปลี่ยนแปลงบริการที่ตรวจสอบได้ไปยังตัวจัดการการแจ้งการเปลี่ยนแปลงบริการเพื่อทำการแจ้งไปยังผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการต่อไป

นอกจากตัวกระทำการสำหรับเปลี่ยนแปลงชนิดของบริการแล้ว มอดูลสนับสนุนการเปลี่ยนแปลงบริการยังประกอบด้วยส่วนต่อประสานสำหรับผู้เปลี่ยนแปลงบริการ (ผู้พัฒนาบริการ) และผู้ต้องการรับทราบการเปลี่ยนแปลงบริการในการเริ่มต้นเข้าใช้งานบริการเอสซีเอ็นโดยทั้งนี้มีจุดประสงค์เพียงเพื่อความสะดวกของผู้ใช้งานเท่านั้น

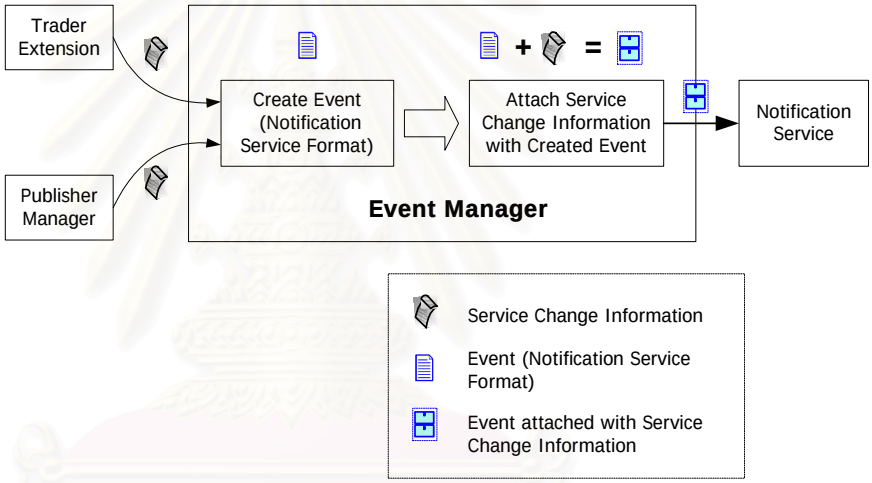
2.1.1.3. **ตัวจัดการการแจ้งการเปลี่ยนแปลงบริการ** (SCN Manager: Service Change Notification Manager)

ตัวจัดการการแจ้งการเปลี่ยนแปลงบริการจะทำการรับข้อมูลความต้องการจากผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ (Subscriber) และส่งข้อมูลแจ้งการเปลี่ยนแปลงบริการกลับไปยังผู้ที่ต้องการรับทราบการเปลี่ยนแปลงเมื่อเกิดการเปลี่ยนแปลงบริการในรูปแบบต่างๆ ขึ้นอยู่กับความต้องการของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการแต่ละราย โครงสร้างภายในของตัวจัดการการแจ้งการเปลี่ยนแปลง ประกอบด้วย 5 ส่วนหลักๆ ซึ่งมีหน้าที่ที่แตกต่างกันดังนี้ คือ

2.1.1.3.1.**ตัวจัดการผู้แจ้งการเปลี่ยนแปลง** (Publisher Manager)

ทำหน้าที่รับการลงทะเบียนและรับข้อมูลแจ้งการเปลี่ยนแปลงบริการล่วงหน้าจากผู้เปลี่ยนแปลงบริการ ข้อมูลที่ได้รับมาจะถูกส่งต่อไปยังตัวจัดการเหตุการณ์

2.1.1.3.2.**ตัวจัดการเหตุการณ์** (Event Manager)



รูปที่ 2.5 โครงสร้างการทำงานของตัวจัดการเหตุการณ์

จากรูปที่ 2.5 ตัวจัดการเหตุการณ์จะทำหน้าที่รับข้อมูลการเปลี่ยนแปลงบริการจากทั้งตัวจัดการผู้แจ้งการเปลี่ยนแปลงสำหรับข้อมูลแจ้งการเปลี่ยนแปลงล่วงหน้า และรับข้อมูลจากตัวตรวจบริการเทรดเดอร์สำหรับข้อมูลแจ้งการเปลี่ยนแปลงในขณะที่บริการได้รับการเปลี่ยนแปลง จากนั้นจึงนำข้อมูลที่ได้รับมาสร้างเป็นเหตุการณ์ในรูปแบบที่บริการแจ้งเหตุการณ์กำหนดไว้ พร้อมกับแนบท้ายด้วยข้อมูลการเปลี่ยนแปลงบริการที่ได้รับ จากนั้นจึงนำเหตุการณ์ดังกล่าวส่งต่อไปยังบริการแจ้งเหตุการณ์เพื่อทำการจัดส่งข้อมูลการเปลี่ยนแปลงบริการ ให้ตรงกับความต้องการของผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการต่อไป

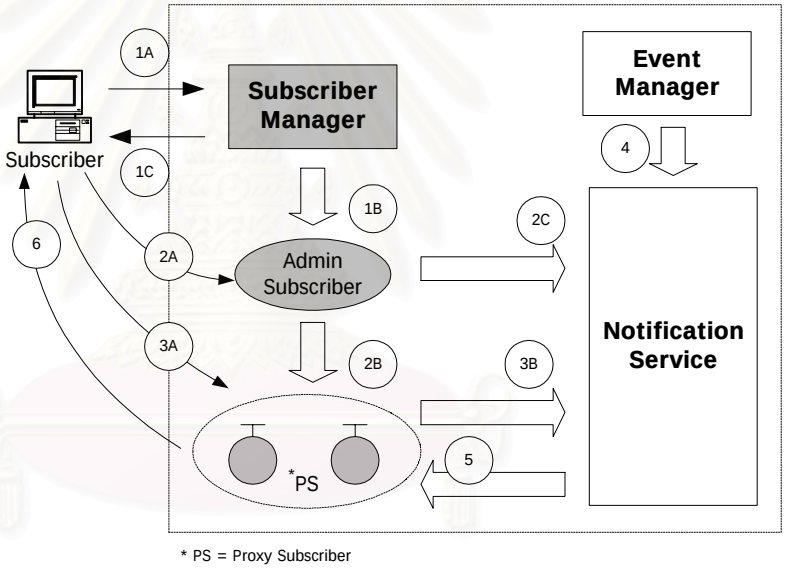
2.1.1.3.3.**บริการแจ้งเหตุการณ์** (Notification Service) [4]

บริการแจ้งเหตุการณ์เป็นบริการพื้นฐานในระบบกระจายคอร์บา ซึ่งภายในงานวิจัย [6] นี้นำเข้ามาใช้งาน จากรูปที่ 2.1 บริการแจ้งเหตุการณ์ได้รับข้อมูลเหตุการณ์การเปลี่ยนแปลงบริการมา

จากตัวจัดการเหตุการณ์ บริการแจ้งเหตุการณ์จะทำการตรวจสอบเหตุการณ์ที่ได้รับมาว่าตรงกับความต้องการของผู้รับบริการรายไหนบ้าง (Filtering) จากนั้นจึงส่งเหตุการณ์ดังกล่าวไปให้แก่ผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการนั้นๆ โดยผ่านตัวแทนผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (Proxy Subscriber) นอกจากนี้บริการแจ้งเหตุการณ์ยังดูแลในส่วนของรูปแบบการจัดส่ง (แบบพุช และ พูล) การจัดลำดับการจัดส่ง (Order Policy) การคัดทิ้ง (Discard Policy) รวมทั้งการจัดเก็บเหตุการณ์การเปลี่ยนแปลงในกรณีที่ผู้ต้องการรับทราบการเปลี่ยนแปลงไม่พร้อมที่จะรับข้อมูลแจ้งการเปลี่ยนแปลงบริการ

2.1.1.3.4.**ตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ** (Subscriber Manager)

ทำหน้าที่รับการลงทะเบียนและดูแลข้อมูลของผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ รูปที่ 2.6 แสดงขั้นตอนการเข้าใช้งานตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ ซึ่งสามารถสรุปเป็นขั้นตอนได้ดังต่อไปนี้



รูปที่ 2.6 การใช้งานของผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ

ขั้นตอนที่ 1 ผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการทำการลงทะเบียน เพื่อขอเข้าใช้งานระบบ (1A) เมื่อผ่านการตรวจสอบความถูกต้องของข้อมูลเรียบร้อยแล้ว ตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการจะทำการสร้างวัตถุผู้ดูแล (Admin Subscriber) ขึ้น (1B) ข้อมูลอ้างถึงวัตถุ (Object Reference) ของวัตถุผู้ดูแลจะถูกส่งให้กับผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ (1C)

ขั้นตอนที่ 2 ผู้ใช้งานติดต่อกับวัตถุผู้ดูแล เพื่อสร้างตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ (Proxy Subscriber) ขึ้น (2A) ตัวแทนจะถูกสร้างขึ้น (2B) และได้รับการลงทะเบียนไว้กับบริการแจ้งเหตุการณ์ (2C) เพื่อขอรับเหตุการณ์

การเปลี่ยนแปลงบริการ (รายละเอียดของตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการจะกล่าวในส่วนถัดไป)

ขั้นตอนที่ 3 ผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการทำการกำหนดหรือแก้ไข รายการบริการที่ตนต้องการรับทราบการเปลี่ยนแปลง พร้อมทั้งกำหนดค่าคุณสมบัติต่างๆ (Subscriber QoS Properties) (3A) เมื่อตัวแทนได้รับข้อมูลก็จะไปทำการบันทึกข้อมูลการกรอง (Filter) และข้อมูลคุณสมบัติต่างๆ ที่บริการแจ้งเหตุการณ์ (3B)

ขั้นตอนที่ 4,5,6 บริการแจ้งเหตุการณ์ได้รับเหตุการณ์การเปลี่ยนแปลงบริการจากตัวจัดการเหตุการณ์ (4) จากนั้นบริการแจ้งเหตุการณ์ตรวจสอบว่าเหตุการณ์การเปลี่ยนแปลงบริการที่ได้รับมา ตรงกับความต้องการของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงรายใดบ้าง การจัดส่งจะกระทำผ่านตัวแทนผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (5) โดยรูปแบบข้อมูลที่จัดส่งจะเป็นไปตามชนิดของตัวแทนที่ถูกสร้างขึ้นมา (6)

2.1.1.3.5.**ตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ** (PS: Proxy Subscriber)

เป็นส่วนที่สร้างขึ้นโดยคำสั่งของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ เพื่อทำหน้าที่หลัก ในการส่งผ่านข้อมูลแจ้งการเปลี่ยนแปลงบริการที่ได้รับมาจากบริการแจ้งเหตุการณ์ไปยังวัตถุเรียกกลับ (Callback Object) ของผู้ที่ต้องการรับทราบการเปลี่ยนแปลง ตามรูปแบบการจัดส่งข้อมูลและรูปแบบของข้อมูลซึ่งขึ้นอยู่กับประเภทของตัวแทนที่ได้รับการสร้างขึ้น

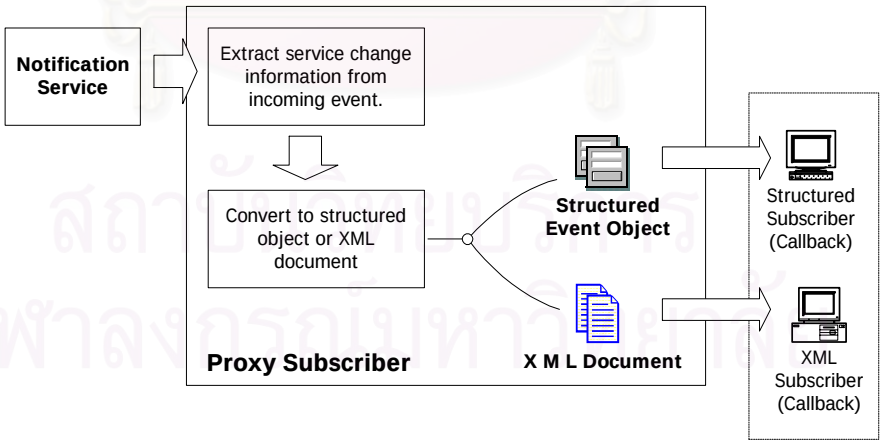
รูปแบบการจัดส่งข้อมูลที่จะสนับสนุน ได้แก่ การจัดส่งแบบพุชและพูล ส่วนรูปแบบของข้อมูลประกอบด้วย วัตถุเหตุการณ์แบบโครงสร้าง ชุดของวัตถุเหตุการณ์แบบโครงสร้าง และเอกสารเอ็กซ์เอ็มแอล ในส่วนของรูปแบบข้อมูลที่เป็นเอกสารเอ็กซ์เอ็มแอลนั้นมีวัตถุประสงค์เพื่อรองรับผู้ใช้งานบริการที่อยู่ในสภาวะแวดล้อมที่ต่างไปจากระบบกระจายคอร์บา รวมทั้งการส่งผ่านข้อมูลการเปลี่ยนแปลงบริการไปให้แก่ระบบอินเทอร์เน็ตด้วย

ตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ แบ่งเป็น 6 ประเภท คือ

1. StructuredPushProxy ตัวแทนจัดส่งข้อมูลแบบพุช โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบวัตถุเหตุการณ์แบบโครงสร้าง (Structured Event Object)
2. XMLPushProxy ตัวแทนจัดส่งข้อมูลแบบพุช โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบของเอกสารเอ็กซ์เอ็มแอล

3. SequenceStructuredPushProxy ตัวแทนจัดส่งข้อมูลแบบพุช โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบของชุดของวัตถุเหตุการณ์แบบโครงสร้าง (Sequence of Structured Event Object)
4. StructuredPullProxy ตัวแทนจัดส่งข้อมูลแบบพูล โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบของวัตถุเหตุการณ์แบบโครงสร้าง
5. XMLPullProxy ตัวแทนจัดส่งข้อมูลแบบพูล โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบของเอกสารเอ็กซ์เอ็มแอล
6. SequenceStructuredPullProxy ตัวแทนจัดส่งข้อมูลแบบพูล โดยข้อมูลที่ทำการจัดส่งอยู่ในรูปแบบของชุดของวัตถุเหตุการณ์แบบโครงสร้าง

จากรูปที่ 2.7 เป็นการทำงานของตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงที่มีการจัดส่งข้อมูลการเปลี่ยนแปลงบริการแบบพุช (StructuredPushProxy, XMLPushProxy และ SequenceStructuredPushProxy) โดยการทำงานเริ่มต้นจากบริการแจ้งเหตุการณ์ทำการส่งเหตุการณ์การเปลี่ยนแปลงบริการให้กับตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ จากนั้นตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงจะทำการแยกเอาแต่ข้อมูลการเปลี่ยนแปลงออกมาจากเหตุการณ์การเปลี่ยนแปลงบริการ ข้อมูลการเปลี่ยนแปลงบริการจะถูกจัดให้อยู่ในรูปแบบของวัตถุเหตุการร์แบบโครงสร้าง หรือ เอกสารเอ็กซ์เอ็มแอล ขึ้นอยู่กับประเภทของตัวแทน จากนั้นทำการส่งให้กับวัตถุเรียกกลับ (Callback) ของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการต่อไป



รูปที่ 2.7 การทำงานของตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการแบบพุช

สำหรับการทำงานของตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงที่มีการจัดส่งข้อมูลการเปลี่ยนแปลงบริการแบบพูล (StructuredPullProxy, XMLPullProxy และ SequenceStructuredPullProxy) ก็จะมีลักษณะที่คล้ายคลึงกันต่างกันแต่เพียงการทำงานจะเริ่มต้น

จากผู้ที่ต้องการรับทราบการเปลี่ยนแปลงต้องมาทำการร้องขอ เพื่อรับทราบข้อมูลการเปลี่ยนแปลงบริการจากตัวแทน จากนั้นตัวแทนก็จะไปทำการดึงเหตุการณ์การเปลี่ยนแปลงบริการจากบริการแจ้งเหตุการณ์ โดยเหตุการณ์ที่ดึงมาได้ก็จะได้รับการจัดการในลักษณะเดียวกันกับการทำงานโดยตัวแทนแบบพุช

ผู้ที่ต้องการรับทราบการเปลี่ยนแปลงแต่ละรายสามารถสร้างตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงได้ประเภทละหนึ่งตัวเท่านั้น เพื่อใช้ในจัดรูปแบบข้อมูลการเปลี่ยนแปลงบริการให้ตรงกับความต้องการ

งานวิจัย [6] นี้ ได้ทดสอบระบบแจ้งเหตุการณ์หนึ่งระบบที่ทำงานร่วมกับเทรดเดอร์หนึ่งตัว หากมีเทรดเดอร์มากกว่าหนึ่งตัวที่เชื่อมต่อกัน ระบบแจ้งเหตุการณ์จะไม่สามารถแจ้งการเปลี่ยนแปลงบริการไปยังผู้รับบริการของเทรดเดอร์ตัวอื่นที่เชื่อมต่อกันอยู่ได้ ในกรณีที่บริการที่ลงทะเบียนเพื่อขอทราบการเปลี่ยนแปลงนั้นได้มาจากการค้นหาจากเทรดเดอร์ตัวอื่น

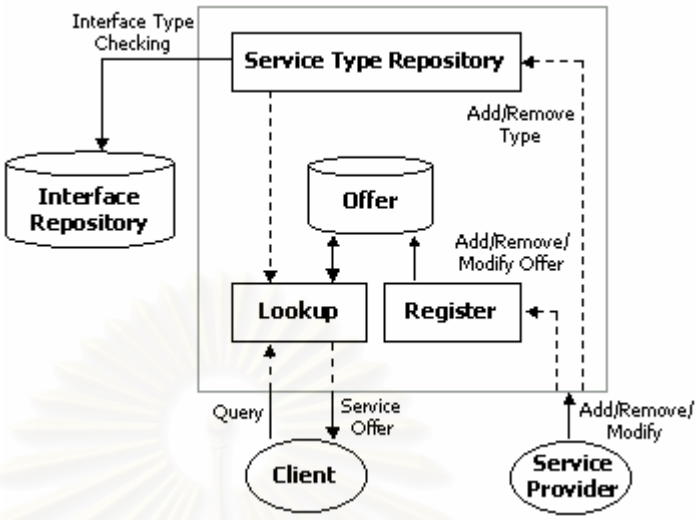
จากปัญหาดังกล่าวผู้วิจัยจึงมีแนวคิดที่จะสร้างส่วนขยายระบบแจ้งการเปลี่ยนแปลงบริการขึ้นให้สามารถรองรับการแจ้งการเปลี่ยนแปลงของบริการระยะไกลได้ โดยให้ระบบแจ้งการเปลี่ยนแปลงแต่ละระบบ ซึ่งต่างก็เชื่อมต่อกับเทรดเดอร์คนละตัว สามารถส่งต่อรายการเปลี่ยนแปลงที่เกิดขึ้นในระบบของตนไปให้ระบบอื่นๆ ได้ โดยอาศัยกลไกการเชื่อมต่อของเทรดเดอร์เหล่านั้น การส่งต่อจะพิจารณาจากสัญญาข้อตกลงที่แต่ละเทรดเดอร์ทำร่วมกัน โดยหากบริการใดมีการเปลี่ยนแปลงและบริการนั้นได้รับการส่งออกไปยังเทรดเดอร์อื่น การเปลี่ยนแปลงนี้จะถูกส่งต่อไปแจ้งยังผู้รับบริการในระบบอื่นนั้นได้

2.1.2. Federation Management for the OMG-Trader [8]

งานวิจัยนี้ได้พัฒนาระบบตามแบบจำลองการเชื่อมต่อระหว่างเทรดเดอร์ เพื่อใช้จัดการกับเทรดเดอร์ที่มีการเชื่อมต่อกัน โดยอาศัยบัญชีรายชื่อบริการเป็นข้อมูลที่ใช้ในการโฆษณาบริการของเทรดเดอร์ส่งออก และมีการแลกเปลี่ยนข้อมูลระหว่างผู้ดูแลเทรดเดอร์นำเข้าและผู้ดูแลเทรดเดอร์ส่งออก เพื่อสร้างสัญญาข้อตกลงระหว่างกัน ซึ่งในส่วนของสัญญาส่งออกของเทรดเดอร์ส่งออกและสัญญานำเข้าของเทรดเดอร์นำเข้าจะมีข้อมูลบริการที่สอดคล้องกันทั้งสองฝ่ายเก็บอยู่ เมื่อผู้ดูแลเทรดเดอร์ทั้งสองฝ่ายมีการเปลี่ยนแปลงข้อมูลบริการในสัญญาข้อตกลง หรือยกเลิกสัญญาข้อตกลง ก็จะต้องคอยดูแลให้สัญญาทั้งสองฝ่ายสอดคล้องกันเสมอ

2.2 ทฤษฎีที่เกี่ยวข้อง

2.2.1. บริการเทรดเดอร์ [3]



รูปที่ 2.8 โครงสร้างของบริการเทรดเดอร์

บริการเทรดเดอร์เป็นบริการพื้นฐานตัวหนึ่งที่กำหนดขึ้นโดยโอเอ็มจี ทำหน้าที่คล้ายกับสมุดหน้าเหลืองสำหรับการค้นหาบริการในระบบกระจายคอร์บา การค้นหาบริการจะอาศัยข้อมูลค่าคุณสมบัติต่างๆ ของบริการเป็นคีย์สำคัญในการค้นหาซึ่งต่างจากบริการชื่อ (Naming Service) [2] ที่ใช้ชื่อเป็นคีย์ในการค้นหาบริการ

บริการเทรดเดอร์ประกอบด้วยองค์ประกอบที่สำคัญ 4 ส่วน (รูปที่ 2.8) คือ

2.2.1.1. คลังชนิดของบริการ (Service Type Repository)

มีลักษณะเป็นฐานข้อมูลสำหรับเก็บข้อมูลชนิดของบริการ (รูปที่ 2.9) โดยมีการติดต่อกับคลังส่วนต่อประสาน (Interface Repository) [1] เพื่อตรวจสอบความถูกต้องของชื่อชนิดของส่วนต่อประสาน (Interface Type Name)

```
Service <ServiceTypeName>[:<BaseServiceTypeName> [,
                    <BaseServiceTypeName>]*] {
        interface <InterfaceTypeName>;
        {[mandatory][read-only][Normal] property <IDL Type> <PropertyName>;]*
};
```

รูปที่ 2.9 BNF แสดงโครงสร้างของชนิดบริการ

2.2.1.2. **ฐานข้อมูลข้อเสนอบริการ** (Offer List)

เป็นฐานข้อมูลสำหรับเก็บข้อเสนอของบริการ (Offer) ซึ่งประกอบด้วยข้อมูลดังต่อไปนี้

1. ข้อมูลอ้างถึงวัตถุ (Object Reference) ที่แสดงตำแหน่งที่อยู่ของตัวบริการ
2. ชื่อชนิดของบริการ (Service Type Name)
3. รายการคู่ลำดับของชื่อคุณสมบัติและค่าคุณสมบัติ (Service Properties)

2.2.1.3. **ส่วนการลงทะเบียน** (Register Component)

เป็นส่วนที่ผู้พัฒนาบริการใช้ในการติดต่อกับบริการเทรดเดอร์เพื่อขอโฆษณาบริการ โดยปกติแล้วการโฆษณาบริการประกอบด้วยขั้นตอนที่สำคัญ 2 ขั้นตอนด้วยกัน คือ

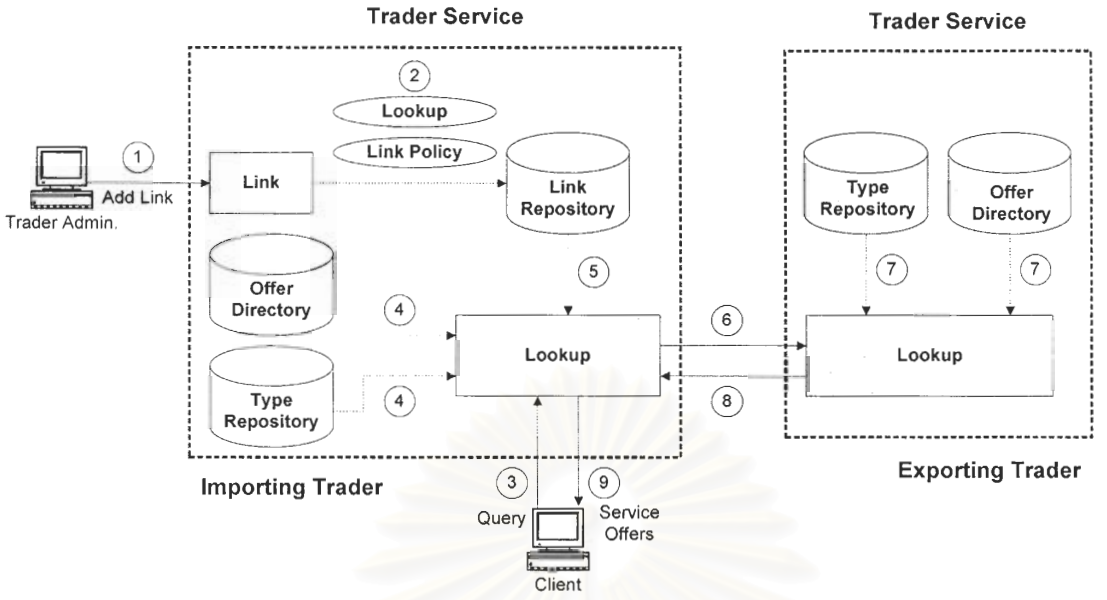
ขั้นตอนที่ 1 ผู้พัฒนาบริการให้ข้อมูลชนิดของบริการแก่บริการเทรดเดอร์ในกรณีที่ชนิดของบริการนั้นยังไม่มีอยู่ในบริการเทรดเดอร์ซึ่งข้อมูลชนิดของบริการนี้จะได้รับการจัดเก็บยังคลังชนิดของบริการ

ขั้นตอนที่ 2 ผู้พัฒนาบริการให้ข้อมูลข้อเสนอบริการแก่บริการเทรดเดอร์ โดยข้อมูลนี้จะได้รับการจัดเก็บยังฐานข้อมูลข้อเสนอบริการ

2.2.1.4. **ส่วนค้นหาบริการ** (Lookup Component)

เป็นส่วนที่ผู้รับบริการใช้ในการติดต่อเพื่อทำการสืบค้นบริการ ส่วนค้นหาบริการนี้มีการติดต่อกับคลังชนิดของบริการและฐานข้อมูลข้อเสนอบริการ สำหรับทำการค้นหาบริการที่ตรงกับความต้องการของผู้ร้องขอมากที่สุด

ในข้อกำหนด [3] กล่าวถึงการค้นหาบริการที่เกิดจากการเชื่อมต่อกันของเทรดเดอร์หลาย ๆ เทรดเดอร์ (Trader Federation) ไว้ด้วย การค้นหาบริการข้ามระบบที่มีเทรดเดอร์เชื่อมต่อกัน จะอาศัยส่วนต่อประสาน Link เพื่อใช้ในการกำหนดพฤติกรรมในการค้นหาบริการของเทรดเดอร์ดังรูปที่ 2.10 โดยการเชื่อมต่อจะเป็นการเชื่อมเทรดเดอร์นำเข้า (Importing Trader) เข้ากับเทรดเดอร์ส่งออก (Exporting Trader) ซึ่งในที่นี้หมายถึงว่า เทรดเดอร์ส่งออกได้ส่งออกบริการของตนให้กับเทรดเดอร์นำเข้า ทำให้เทรดเดอร์นำเข้าสามารถค้นหาบริการที่ประกาศอยู่ที่เทรดเดอร์ส่งออกได้เสมือนกับว่าบริการเหล่านั้นประกาศอยู่ที่เทรดเดอร์นำเข้าเอง ขั้นตอนการทำงานของการเชื่อมต่อเทรดเดอร์เป็นดังนี้



รูปที่ 2.10 การค้นหาบริการจากเทรดเดอร์ที่มีการเชื่อมต่อกัน

ขั้นตอนที่ 1,2 ก่อนการค้นหาบริการจากเทรดเดอร์ส่งออก ผู้ดูแลเทรดเดอร์นำเข้าจะต้องสร้างการเชื่อมต่อกับเทรดเดอร์ส่งออกเสียก่อน โดยผ่านส่วนต่อประสาน Link ซึ่งข้อมูลการเชื่อมต่อ ได้แก่ ส่วนต่อประสาน Lookup ของเทรดเดอร์ส่งออก และนโยบายการเชื่อมต่อ ซึ่งข้อมูลเหล่านี้จะถูกบันทึกลงคลังข้อมูลการเชื่อมต่อ (Link Repository)

ขั้นตอนที่ 3,4,5 เมื่อผู้รับบริการต้องการค้นหาบริการผ่านส่วนต่อประสาน Lookup แต่เทรดเดอร์ไม่พบบริการที่สอดคล้องภายในคลังชนิดของบริการ (Type Repository) และฐานข้อมูลข้อเสนอบริการ (Offer Directory) ของตน เทรดเดอร์จะค้นหาส่วนต่อประสาน Lookup และนโยบายการเชื่อมต่อกับเทรดเดอร์ส่งออก จากคลังข้อมูลการเชื่อมต่อ

ขั้นตอนที่ 6 เทรดเดอร์นำเข้าส่งต่อคำร้องขอของผู้รับบริการผ่านส่วนต่อประสาน Lookup ของเทรดเดอร์ส่งออก

ขั้นตอนที่ 7 เทรดเดอร์ส่งออกทำการค้นหาบริการจากคลังชนิดบริการและฐานข้อมูลข้อเสนอบริการของตน

ขั้นตอนที่ 8,9 เทรดเดอร์ส่งออกส่งบริการที่สอดคล้องกลับไปยังเทรดเดอร์นำเข้า ซึ่งจะส่งต่อกลับไปยังผู้รับบริการ

2.2.2. Federating Traders: an ODP Adventure [2]

งานวิจัยนี้ได้นำเสนอรูปแบบของการทำงานร่วมกันระหว่างเทรดเดอร์ของอาร์เอ็ม-โอดีพี (Reference Model – Open Distributed Processing : RM-ODP) ออกเป็น 5 มุมมองด้วยกัน คือ

2.2.2.1. **มุมมองเชิงองค์กร** (Enterprise Viewpoint)

มีการกำหนดกฎเกณฑ์ต่างๆ ในการร้องขอและโฆษณาบริการระหว่างเทรดเดอร์ โดยเทรดเดอร์หนึ่งสามารถร้องขอบริการได้จากเทรดเดอร์มากกว่าหนึ่งตัว และสามารถโฆษณาบริการไปยังเทรดเดอร์ได้มากกว่าหนึ่งตัวเช่นเดียวกัน โดยไม่จำเป็นต้องมีตัวกลางเพื่อเป็นศูนย์รวมระหว่างเทรดเดอร์ ผู้รับบริการสามารถจะร้องขอบริการหรือโฆษณาบริการได้ในทางอ้อมโดยผ่านเทรดเดอร์ที่ผู้รับบริการกำลังใช้อยู่

2.2.2.2. **มุมมองเชิงสารสนเทศ** (Information Viewpoint)

อธิบายลักษณะเฉพาะของเทรดเดอร์ บัญชีรายชื่อสำหรับโฆษณาบริการ สัญญาข้อตกลงในการแลกเปลี่ยนข้อมูล ชนิดของบริการที่เทรดเดอร์จะทำการร้องขอบริการระหว่างกันได้ และข้อจำกัดในการรับข้อมูลของผู้โฆษณาบริการ

2.2.2.3. **มุมมองเชิงคำนวณ** (Computational Viewpoint)

การติดต่อระหว่างเทรดเดอร์จะใช้ส่วนต่อประสาน ซึ่งในการติดต่อกันนั้นจะมีการทำสัญญาข้อตกลงขึ้นมาก่อนว่าบริการใดที่ส่งออกหรือนำเข้าได้บ้าง ดังนั้นวิธีการติดต่อระหว่างเทรดเดอร์ในกลุ่มนั้นสามารถแบ่งออกได้เป็น 2 วิธี

2.2.2.3.1. การจัดการสัญญา

จะมีตัวดำเนินการ (Operation) สำหรับบัญชีรายชื่อบริการ สัญญานำเข้า สัญญาส่งออก เพื่อกระจายบัญชีรายชื่อบริการ (Distributing Catalogue) ร้องขอบัญชีรายชื่อบริการ (Requesting Catalogue) สร้าง/แสดงรายการ/แก้ไข/ยกเลิกสัญญาการเชื่อมต่อ (Establishing/Listing/Modifying/Terminating Federation Contracts)

2.2.2.3.2. การจัดการการติดต่อกันในกลุ่ม (Federation Operation)

สามารถแบ่งออกเป็นกลุ่มตามการใช้งานได้ดังนี้

2.2.2.3.2.1. ตัวดำเนินการสำหรับผู้นำเข้าบริการ ได้แก่ F-List, F-Search, F-Select

2.2.1.3.2.2. ตัวดำเนินการสำหรับส่งออกบริการ ได้แก่ F-Export, F-Modify, F-Withdraw

2.2.2.3.2.3. ตัวดำเนินการสำหรับจัดการชนิดของบริการ ได้แก่ F-Add-Type, F-Delete-Type, F-List-Type

2.2.2.3.2.4. ตัวดำเนินการสำหรับจัดการไดเร็กทอรี ได้แก่ F-Add-Directory, F-Delete-Directory, F-List-Directory

2.2.2.4. **มุมมองเชิงวิศวกรรม** (Engineering Viewpoint)

รายละเอียดข้อตกลงการส่งออกและการนำเข้าบริการจะถูกเก็บและสามารถดึงข้อมูลได้ตามความต้องการเพื่อที่จะใช้ในมุมมองเชิงคำนวณต่อไป

2.2.2.5. **มุมมองเชิงเทคโนโลยี** (Technology Viewpoint)

จะพิจารณารายละเอียดทางเทคโนโลยี เช่น การใส่รหัสพื้นฐานเพื่อใช้ในการเปลี่ยนแปลงข้อมูลก่อนนำเสนอ หรือการเพิ่มประสิทธิภาพในการค้นหาสำหรับการร้องขอบริการจากเทรดเดอร์ตัวอื่นนั้น อาจมีการสร้างดัชนีอ้างอิง (Import Indexes) เพื่อให้การค้นหาง่ายขึ้น ดัชนีอ้างอิงนี้จะเป็นการรวบรวมชนิดของบริการทั้งหมดที่สามารถร้องขอได้จากเทรดเดอร์ที่ส่งออก

บทที่ 3

การออกแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ในบทนี้กล่าวถึงรายละเอียดของการออกแบบการเชื่อมต่อกันของระบบแจ้งการเปลี่ยนแปลงของบริการในสถาปัตยกรรมคอร์บาให้รองรับการแจ้งการเปลี่ยนแปลงของบริการระยะไกลได้ โดยให้ระบบแจ้งการเปลี่ยนแปลงแต่ละระบบ ซึ่งต่างก็เชื่อมต่อกับเทรดเดอร์คนละตัวนั้น สามารถส่งต่อรายการเปลี่ยนแปลงที่เกิดขึ้นในระบบของตนไปให้ระบบอื่นๆ ได้ ทั้งนี้เป็นผลเนื่องมาจากการเชื่อมต่อกันของเทรดเดอร์เหล่านั้น ที่ทำให้ผู้รับบริการของเทรดเดอร์ตัวหนึ่งสามารถเรียกใช้บริการของเทรดเดอร์อีกตัวหนึ่ง อาจมีความประสงค์ที่จะรับทราบการเปลี่ยนแปลงของบริการนั้นด้วย การพัฒนาการเชื่อมต่อกันของระบบแจ้งการเปลี่ยนแปลง จะทำโดยอาศัยกลไกการเชื่อมต่อกันของเทรดเดอร์เหล่านั้น กล่าวคือ การส่งต่อข้อมูลแจ้งการเปลี่ยนแปลง จะพิจารณาจากสัญญาข้อตกลงที่แต่ละเทรดเดอร์ทำงานร่วมกัน โดยหากบริการใดมีการเปลี่ยนแปลงและบริการนั้นได้รับการส่งออกไปยังเทรดเดอร์อื่น ข้อมูลแจ้งการเปลี่ยนแปลงนี้จะถูกส่งต่อไปแจ้งยังผู้รับบริการในระบบอื่นได้

ในสัญญาข้อตกลงระหว่างสองเทรดเดอร์ การส่งออกบริการทำได้ 2 ลักษณะดังนี้

1. <u>การส่งออกในระดับชนิดบริการ</u> ตัวอย่างเช่น การส่งออกชนิดบริการ Bank จะหมายถึงข้อเสนอบริการที่มีชนิด Bank ทั้งหมดถูกส่งออก (เช่น ข้อเสนอบริการ Bank ที่มี Bank name = “TFB” และที่มี Bank name = “SCB” ถูกส่งออกทั้งหมด) ข้อมูลที่ระบุในสัญญาข้อตกลง ได้แก่ ชนิดบริการที่จะส่งออก
2. <u>การส่งออกในระดับข้อเสนอบริการ</u> ตัวอย่างเช่น การส่งออกเฉพาะข้อเสนอบริการ Bank ที่มี Bank name = “TFB” เท่านั้น ข้อมูลที่ระบุในสัญญาข้อตกลง ได้แก่ ชนิดบริการของข้อเสนอ ค่าคุณสมบัติต่างๆ ของข้อเสนอ และวัตถุอ้างอิงบริการที่จะส่งออก

สัญญานำเข้าและสัญญาส่งออกระหว่างสองเทรดเดอร์จะต้องสอดคล้องกันเสมอ นั่นคือหากฝ่ายหนึ่งฝ่ายใดต้องการแก้ไขสัญญา จะต้องแจ้งให้ผู้ดูแลเทรดเดอร์คู่สัญญารับทราบ เพื่อแก้ไขสัญญาให้ตรงกัน

การแจ้งการเปลี่ยนแปลงโดยอาศัยสัญญาข้อตกลง จะเกิดขึ้นใน 2 ลักษณะดังนี้

1. เมื่อบริการในระบบหนึ่งได้รับการเปลี่ยนแปลง หากบริการนั้นถูกส่งออกไปยังระบบอื่นตามสัญญาข้อตกลงการเชื่อมต่อกันของเทรดเดอร์ ข้อมูลการเปลี่ยนแปลงของบริการนั้น จะถูกส่งต่อไปยังระบบแจ้งการเปลี่ยนแปลงของระบบอื่น แต่หากบริการนั้นไม่ได้ถูกส่งออกไปยังระบบอื่น การแจ้งการเปลี่ยนแปลงก็จะเกิดขึ้นเฉพาะในระบบนั้นตามปกติ
2. เมื่อเกิดสัญญาข้อตกลงใหม่ในระบบ หรือสัญญาข้อตกลงที่มีอยู่ได้รับการยกเลิก หรือแก้ไข (โดยการเพิ่ม|ลบ|แก้ไขชนิดบริการ และการเพิ่ม|ลบ|แก้ไขข้อเสนอบริการ) จะเสมือนกับมีบริการเกิดขึ้น

ใหม่|ยกเลิก|เปลี่ยนแปลงในระบบของเทรดเดอร์นำเข้า ดังนั้นจะมีการแจ้งการเปลี่ยนแปลงเกิดขึ้นในระบบนั้น ทั้งนี้การแก้ไขสัญญาข้อตกลงนี้ อาจเป็นผลมาจากการที่บริการส่งออกเกิดการเปลี่ยนแปลงจริงๆ หรือเป็นเพียงการเปลี่ยนแปลงข้อตกลงระหว่างเทรดเดอร์นำเข้ากับเทรดเดอร์ส่งออก โดยไม่ได้เกิดการเปลี่ยนแปลงขึ้นกับบริการจริงๆ ก็ได้

3.1 การแจ้งการเปลี่ยนแปลงระยะไกลโดยอาศัยสัญญาข้อตกลง

จากลักษณะการส่งออกบริการและลักษณะการแจ้งการเปลี่ยนแปลงข้างต้น สามารถสรุปเป็นกรณีต่างๆ ที่ต้องดำเนินการดังนี้

3.1.1 บริการได้รับการเปลี่ยนแปลง

จะแจ้งเตือนภายในเทรดเดอร์ตัวนั้น พร้อมทั้งตรวจสอบ

3.1.1.1 หากบริการไม่ได้ถูกส่งออก

ไม่ต้องส่งต่อการแจ้งเตือน

3.1.1.2 หากบริการถูกส่งออก

3.1.1.2.1 ส่งออกระดับชนิดบริการ

- การเปลี่ยนแปลงเกิดกับชนิดบริการ จะส่งต่อการแจ้งเตือนไปยังเทรดเดอร์นำเข้า
- การเปลี่ยนแปลงเกิดกับข้อเสนอบริการ จะส่งต่อการแจ้งเตือนไปยังเทรดเดอร์นำเข้า

3.1.1.2.2 ส่งออกระดับข้อเสนอบริการ

- การเปลี่ยนแปลงเกิดกับชนิดบริการ จะส่งต่อการแจ้งเตือนไปยังเทรดเดอร์นำเข้า โดยเป็นผลสืบเนื่องจากการเปลี่ยนแปลงสัญญาข้อตกลง เมื่อชนิดของข้อเสนอบริการที่ส่งออกเปลี่ยนไป
- การเปลี่ยนแปลงเกิดกับข้อเสนอบริการ จะส่งต่อการแจ้งเตือนไปยังเทรดเดอร์นำเข้าเฉพาะกรณีที่ข้อเสนอบริการที่เปลี่ยนแปลงคือข้อเสนอบริการที่ส่งออก

3.1.2 บริการไม่ได้รับการเปลี่ยนแปลง แต่มีการเปลี่ยนแปลงสัญญาข้อตกลง

3.1.2.1 การแก้ไขสัญญาข้อตกลง โดยการเพิ่ม|ลบ|แก้ไขชนิดบริการ และเพิ่ม|ลบ|แก้ไขข้อเสนอบริการ

จะเกิดการแจ้งเตือนในระบบของเทรดเดอร์นำเข้า

3.1.2.2 การยกเลิกสัญญาข้อตกลง

เป็นการลบชนิดบริการ/ข้อเสนอบริการทั้งหมดดังนั้นจะเกิดการแจ้งเตือนในระบบของเทรดเดอร์นำเข้า

3.1.2.3 **การสร้างสัญญาข้อตกลงใหม่**

เป็นการเพิ่มชนิดบริการ/ข้อเสนอบริการใหม่ทั้งหมด ดังนั้นจะเกิดการแจ้งเตือนในระบบของเทรดเดอร์นำเข้า

3.1.3 **บริการได้รับการเปลี่ยนแปลง แล้วส่งผลให้เกิดการเปลี่ยนแปลงสัญญาข้อตกลงด้วย**

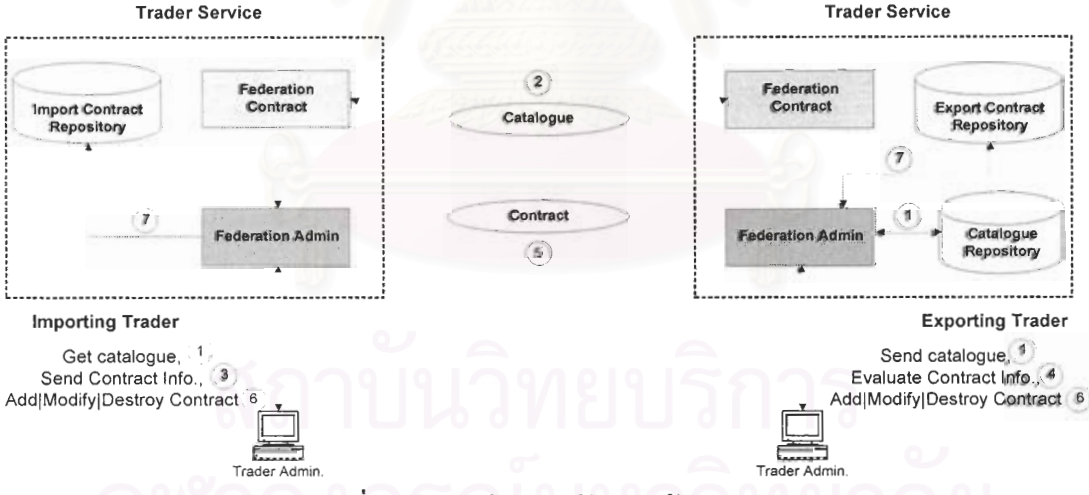
จะเกิดการแจ้งเตือนดังในหัวข้อที่ 3.1.1 และ 3.1.2 รวมกัน

3.2 สถาปัตยกรรมของระบบ

จากการออกแบบ การแจ้งการเปลี่ยนแปลงของบริการระยะไกลจะกระทำผ่านสัญญาข้อตกลงของการเชื่อมต่อกันของเทรดเดอร์ แต่กลไกการเชื่อมต่อของเทรดเดอร์ในคอร์บานั้น ยังไม่รองรับการจัดการสัญญาข้อตกลง ดังนั้นงานวิจัยนี้ จะได้ออกแบบและพัฒนาส่วนจัดการสัญญาข้อตกลงเพิ่มเติมให้กับเทรดเดอร์ก่อนที่จะทำการพัฒนาการแจ้งการเปลี่ยนแปลงระยะไกลได้

3.2.1 การเพิ่มส่วนจัดการสัญญาข้อตกลง

รูปที่ 3.1 แสดงส่วนจัดการสัญญาข้อตกลง ซึ่งอธิบายขั้นตอนการทำงานได้ดังนี้



รูปที่ 3.1 ส่วนจัดการสัญญาข้อตกลง

ขั้นตอนที่ 1 ผู้ดูแลเทรดเดอร์ส่งออกนำข้อมูลบัญชีรายชื่อบริการจากคลังบัญชีรายชื่อบริการ (Catalogue Repository) หรือสร้างบัญชีรายชื่อบริการใหม่ที่ยอมให้ผู้รับบริการของเทรดเดอร์นำเข้าเรียกใช้ได้ แล้วส่งไปยังเทรดเดอร์นำเข้า หรือผู้ดูแลเทรดเดอร์นำเข้าสามารถขอให้ผู้ดูแลเทรดเดอร์ส่งออกส่งบัญชีรายชื่อบริการมาให้ตนได้

ขั้นตอนที่ 2 ส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก ส่งบัญชีรายชื่อบริการไปยังส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า โดยผ่านส่วนต่อประสาน FederationContract ของเทรดเดอร์นำเข้า

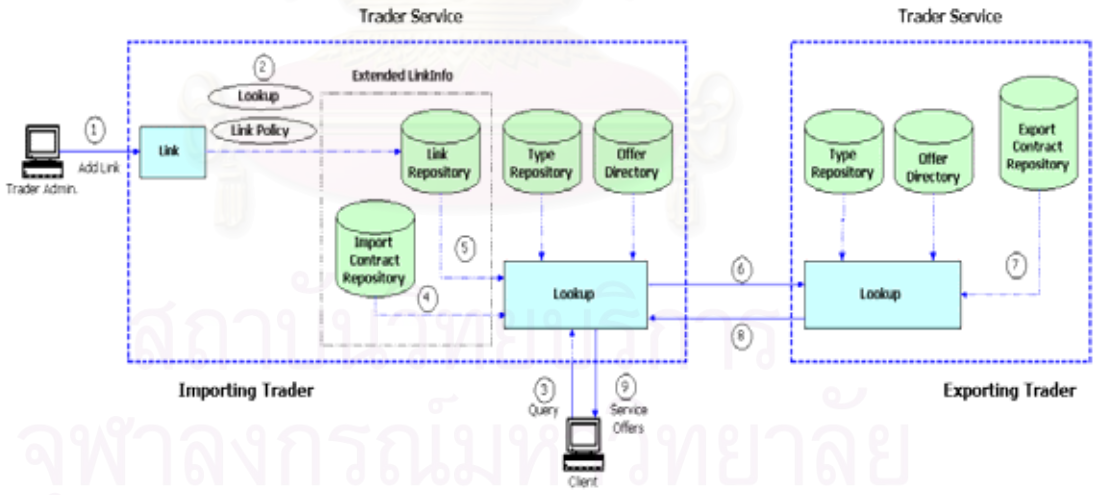
ขั้นตอนที่ 3 ผู้ดูแลเทรดเดอร์นำเข้าเลือกบริการที่ต้องการนำเข้า แล้วเพิ่มรูปแบบสัญญาที่ตนต้องการไว้ในคลังสัญญานำเข้า และส่งรูปแบบสัญญาดังกล่าวไปยังผู้ดูแลเทรดเดอร์ส่งออก

ขั้นตอนที่ 4 ผู้ดูแลเทรดเดอร์ส่งออกพิจารณารูปแบบสัญญาที่ผู้ดูแลเทรดเดอร์นำเข้าเสนอว่าเหมาะสมหรือไม่ เพื่อสร้างสัญญาฉบับสมบูรณ์ที่ผ่านการพิจารณาแล้ว

ขั้นตอนที่ 5 ผู้ดูแลเทรดเดอร์ส่งออกเพิ่มสัญญาที่ผ่านการพิจารณาแล้วไว้ในคลังสัญญาส่งออก แล้วส่งไปยังผู้ดูแลเทรดเดอร์นำเข้า

ขั้นตอนที่ 6,7 ผู้ดูแลเทรดเดอร์นำเข้านำสัญญาที่ผ่านการพิจารณาแล้วไปแก้ไขสัญญาที่ร้องขอในคลังสัญญานำเข้าเพื่อให้สัญญาทั้งสองฝ่ายสอดคล้องกัน โดยทั้งสองฝ่ายสามารถทำการแก้ไขหรือยกเลิกสัญญานี้ได้ในภายหลัง แต่จะต้องแจ้งให้คู่สัญญาทราบ เพื่อให้ทั้งสองฝ่ายมีข้อมูลสัญญาข้อตกลงที่สอดคล้องกันเสมอ

ภายหลังการสร้างสัญญาข้อตกลง การค้นหาบริการระยะไกลจะเป็นดังขั้นตอนในรูปที่ 3.2 ซึ่งอธิบายได้ดังนี้



รูปที่ 3.2 การค้นหาบริการระยะไกลผ่านสัญญาข้อตกลง

ขั้นตอนที่ 1,2 ก่อนการค้นหาบริการระยะไกล ผู้ดูแลเทรดเดอร์นำเข้าจะต้องสร้างการเชื่อมต่อกับเทรดเดอร์ส่งออกเสียก่อน โดยผ่านส่วนต่อประสาน Link โดยที่ข้อมูลการเชื่อมต่ออัน ได้แก่ ส่วนต่อประสาน Lookup ของเทรดเดอร์ส่งออก และนโยบายการเชื่อมต่อจะถูกบันทึกลงคลังข้อมูลการเชื่อมต่อ (Link

Repository) ขั้นตอนนี้เป็นไปตามข้อกำหนดของเทรดเดอร์ของคอร์บาในปัจจุบัน และจะกระทำครั้งเดียว เมื่อติดต่อกับเทรดเดอร์ส่งออกเป็นครั้งแรก

ขั้นตอนที่ 3,4,5 เมื่อผู้รับบริการต้องการค้นหาบริการผ่านส่วนต่อประสาน Lookup แต่เทรดเดอร์ไม่พบบริการที่สอดคล้องภายในคลังชนิดบริการ (Type Repository) และคลังข้อเสนอบริการ (Offer Directory) ของตน เทรดเดอร์จะค้นหาจากคลังสัญญานำเข้าว่าบริการดังกล่าว สามารถนำเข้าจากเทรดเดอร์ตัวอื่นหรือไม่ และจะค้นหาส่วนต่อประสาน Lookup และนโยบายการเชื่อมต่อกับเทรดเดอร์ตัวนั้นจากคลังข้อมูลการเชื่อมต่อ

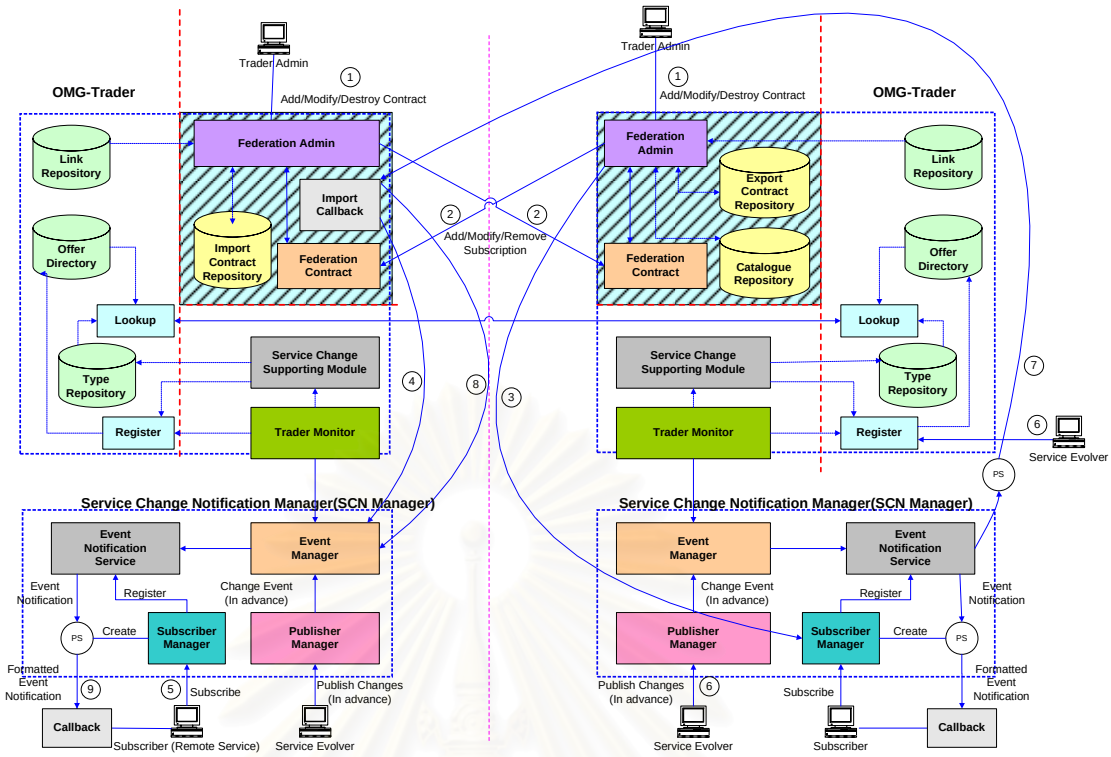
ขั้นตอนที่ 6 เทรดเดอร์นำเข้าส่งต่อคำร้องขอของผู้รับบริการผ่านส่วนต่อประสาน Lookup ของเทรดเดอร์ส่งออก

ขั้นตอนที่ 7 เทรดเดอร์ส่งออกตรวจสอบกับสัญญาส่งออกว่าคำร้องขอจากเทรดเดอร์นำเข้าเป็นไปอย่างถูกต้อง แล้วทำการค้นหาบริการจากคลังชนิดบริการและคลังข้อเสนอบริการของตน

ขั้นตอนที่ 8,9 เทรดเดอร์ส่งออกส่งบริการที่สอดคล้องกลับไปยังเทรดเดอร์นำเข้า ซึ่งจะส่งต่อกลับไปยังผู้รับบริการ

3.2.2 การแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์

งานวิจัยนี้จะได้นำการเชื่อมต่อกันของเทรดเดอร์โดยใช้สัญญานำเข้ามาเป็นพื้นฐานที่ช่วยในการแจ้งการเปลี่ยนแปลงของบริการระยะไกล โดยอาศัยตัวจัดการแจ้งการเปลี่ยนแปลงบริการของงานวิจัย [5] ดังรูปที่ 3.3 (ส่วนที่เพิ่มขยายอยู่ในกรอบสี่เหลี่ยมแรเงา) โดยอธิบายได้ดังนี้



รูปที่ 3.3 การแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ขั้นตอนที่ 1,2 เมื่อมีการสร้างสัญญานำเข้าใหม่ เทรดเดอร์นำเข้าจะลงทะเบียนขอรับแจ้งการเปลี่ยนแปลงบริการที่นำเข้าทั้งหมดผ่านส่วนต่อประสาน FederationContract และส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก โดยระบุตัวเรียกกลับนำเข้า (Import Callback) สำหรับรอรับการแจ้งเหตุการณ์ ในกรณีที่ฝ่ายใดทำการเปลี่ยนแปลงแก้ไขสัญญาข้อตกลง ซึ่งส่งผลให้สัญญานำเข้าได้รับการเปลี่ยนแปลง เทรดเดอร์นำเข้าจะทำการเปลี่ยนแปลงการลงทะเบียนรับแจ้งการเปลี่ยนแปลงให้สอดคล้องกัน

ขั้นตอนที่ 3 คำขอรับแจ้งการเปลี่ยนแปลงบริการ จะถูกส่งไปลงทะเบียนกับตัวจัดการกับผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (Subscriber Manager) ซึ่งอยู่ภายในตัวจัดการแจ้งการเปลี่ยนแปลงบริการของเทรดเดอร์ส่งออก และจะส่งผลให้บริการแจ้งการเปลี่ยนแปลงเหตุการณ์ (Event Notification Service) ทำการสร้างตัวแทนรับทราบการเปลี่ยนแปลงบริการ (Proxy Subscriber) สำหรับเทรดเดอร์นำเข้า

ขั้นตอนที่ 4 เมื่อมีการสร้างสัญญาใหม่เกิดขึ้น หรือมีการเปลี่ยนแปลงสัญญา ตัวเรียกกลับนำเข้าของเทรดเดอร์นำเข้า จะทำการแจ้งไปยังตัวจัดการเหตุการณ์ของระบบแจ้งการเปลี่ยนแปลงบริการภายในเทรดเดอร์นำเข้า และเหตุการณ์จะได้รับการส่งต่อไปยังตัวแทนรับทราบการเปลี่ยนแปลงบริการของผู้รับบริการที่สนใจ

ขั้นตอนที่ 5 ผู้รับบริการภายในเทรดเดอร์นำเข้าสามารถลงทะเบียนรับแจ้งการเปลี่ยนแปลงบริการระยะไกลที่สนใจผ่านตัวจัดการกับผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของระบบแจ้งการเปลี่ยนแปลงของเทรดเดอร์นำเข้า โดยสามารถขอรับแจ้งได้ทั้งแบบพุช (Push) และพูล (Pull)

ขั้นตอนที่ 6 เมื่อบริการภายในเทรดเดอร์ส่งออกได้รับการเปลี่ยนแปลงไม่ว่าจะเป็นการแจ้งผ่านตัวจัดการกับผู้ที่ต้องการแจ้งการเปลี่ยนแปลงบริการ (Publisher Manager) หรือตัวตรวจบริการเทรดเดอร์ตรวจจับเหตุการณ์การเปลี่ยนแปลงที่เกิดขึ้น ณ บริการเทรดเดอร์ได้ หากการเปลี่ยนแปลงเหล่านั้นเป็นของบริการที่เทรดเดอร์นำเข้าทำการลงทะเบียนไว้ บริการแจ้งการเปลี่ยนแปลงเหตุการณ์ของเทรดเดอร์ส่งออกก็จะทำการแจ้งไปยังตัวแทนรับทราบการเปลี่ยนแปลงบริการสำหรับเทรดเดอร์นำเข้า

ขั้นตอนที่ 7 ตัวแทนรับทราบการเปลี่ยนแปลงบริการสำหรับเทรดเดอร์นำเข้าจะแจ้งเหตุการณ์ไปยังตัวเรียกกลับนำเข้าของเทรดเดอร์นำเข้า

ขั้นตอนที่ 8 ตัวเรียกกลับนำเข้าจะจัดรูปแบบเหตุการณ์ เพื่อส่งไปยังตัวจัดการเหตุการณ์ของระบบแจ้งการเปลี่ยนแปลงบริการภายในเทรดเดอร์นำเข้าเอง และเหตุการณ์จะได้รับการส่งต่อไปยังตัวแทนรับทราบการเปลี่ยนแปลงบริการของผู้รับบริการที่สนใจ

ขั้นตอนที่ 9 ตัวแทนรับทราบการเปลี่ยนแปลงบริการแจ้งเหตุการณ์ไปยังตัวเรียกกลับของผู้รับบริการ

บทที่ 4

ต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลในวิทยานิพนธ์นี้ได้พัฒนาขึ้นภายใต้ข้อกำหนดของสถาปัตยกรรมคอร์บารุ่นที่ 2.2 โดยใช้ออร์บของวิสิโบรกเกอร์ (Visibroker) สำหรับภาษาจาวา รุ่นที่ 3.4 [9] บริการเทรดเดอร์ของจาคอร์บ (JacORB) รุ่นที่ 1.3.30 [10] บริการแจ้งเหตุการณ์ของดีคอน (dCon) รุ่น 3.0 [11] และใช้ภาษาจาวา (Java) รุ่นที่ 1.4.2 ในการพัฒนา

4.1 ส่วนต่อประสานของระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ในหัวข้อนี้จะกล่าวถึง ส่วนต่อประสานที่เพิ่มขยายบริการเทรดเดอร์ทั้งหมด ซึ่งในงานวิทยานิพนธ์นี้ ได้ใช้ส่วนต่อประสานดังกล่าวเพื่อช่วยสนับสนุนการทำงานของบริการเอสซีเอ็น [6] ให้สามารถแจ้งการเปลี่ยนแปลงของบริการไปยังผู้รับบริการระยะไกลได้ โดยแบ่งออกได้เป็น 2 ส่วน คือ ส่วนต่อประสานของการจัดการสัญญาข้อตกลง และส่วนต่อประสานของตัวเรียกกลับนำเข้า ซึ่งมีรายละเอียดดังนี้

4.1.1 ส่วนต่อประสานของการจัดการสัญญาข้อตกลง

```
module CosTrading {
  ...
  #use for sending the Contract
 interface FederationAdmin;
 interface FederationContract;
  ...
  interface TraderComponents {
       readonly   attribute    Lookup              lookup_if;
       readonly   attribute   Register              register_if;
      readonly   attribute    Link                 link_if;
      readonly   attribute    Proxy                proxy_if;
      readonly   attribute    Admin                admin_if;
      readonly   attribute   FederationAdmin     federationAdmin_if;
      readonly   attribute   FederationContract  federationContract_if;
  };
```

ส่วนต่อประสานของการจัดการสัญญาข้อตกลงประกอบด้วย 2 ส่วน ได้แก่ ส่วนต่อประสานของผู้ดูแลการเชื่อมต่อ (FederationAdmin) และส่วนต่อประสานของการทำสัญญา (FederationContract) ส่วนต่อประสานทั้งสองได้เพิ่มเข้าไปในมอดูล CosTrading ของคอร์บาเทรดเดอร์ เพื่อใช้จัดการเกี่ยวกับสัญญาข้อตกลง และใช้ข้อมูลบริการที่ร้องขอจากสัญญาข้อตกลง

เพื่อลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก รวมทั้งสร้างตัวเรียกกลับนำเข้า (ImportCallback) เพื่อรอรับแจ้งการเปลี่ยนแปลงบริการจากตัวแจ้งเหตุการณ์ (EventManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก โดยมีการประกาศชนิดของข้อมูลและเอ็กซ์เซ็พชันบางส่วนที่มีการใช้งานร่วมกันในทั้งสองส่วนต่อประสาน ดังนี้

```
module CosTrading {
```

<u>การประกาศข้อมูล</u> ประกอบด้วย

```
    typedef Istring TraderNM; //trader name
    enum ExportLevel {  //export level
        TYPE,
        OFFER
    };

    struct TypeDesc {
        ServiceTypeName                                          name;
        string                                                   if_name;
        CosTradingRepos::ServiceTypeRepository::PropStructSeq      props;
        CosTradingRepos::ServiceTypeRepository::ServiceTypeNameSeq  super_types;
        boolean                                                  masked;
        CosTradingRepos::ServiceTypeRepository::IncarnationNumber   incarnation;
        CORBA::InterfaceDef                                      ifdef;
        ExportLevel                                              level;
    };
    typedef sequence<TypeDesc> TypeDescSeq;

    struct OfferDesc {
        OfferId         offerID;
        Istring         reference; //ior
        ServiceTypeName  type;
        PropertySeq     properties;
    };
    typedef sequence<OfferDesc> OfferDescSeq;

    typedef Istring CatalogueID;
    typedef sequence<CatalogueID> CatalogueIDSeq;
```

```
struct Catalogue {
    CatalogueID  catalogueID;
    TypeDescSeq  typeList;    //list of service types
    OfferDescSeq offerList;   //list of service offers
};
typedef sequence<Catalogue> CatalogueSeq;

typedef Istring UserID;
typedef Istring Password;
typedef Istring ContractID;
typedef sequence<ContractID> ContractIDSeq;

struct Contract {
    ContractID   expContractID;  //reference id of exporting contract
    ContractID   impContractID;  //reference id of importing contract
    Catalogue    catalogue;
     TypeDescSeq  typeReq;       //list of service types request
     OfferDescSeq offerReq;      //list of service offers request
     TraderNM     expTrader;     //exporting trader name
};
typedef sequence<Contract> ContractSeq;

typedef Istring InboxID;
typedef sequence<InboxID> InboxIDSeq;
typedef Istring MesgType;
typedef Istring DateTime;
typedef Istring Mesg;
typedef Istring DataType;

struct DataObj {
    DataType   dataType; //'MSG'=Message, 'CTL'=Catalogue, 'CTR'=Contract
    Mesg         mesg;
    Catalogue  catalogue;
    Contract    contract;
};

 struct Inbox {
     InboxID  inboxID;
     MesgType mesgType; //'M'=Message, 'R'=Request Catalogue, 'D'=Distribute Catalogue,
                        //'E'=Establish Contract, 'A'=Approve Contract
     TraderNM trader;   //exporting trader name
```

```
    UserID   sender;
    DateTime sendDt;
    DateTime receiveDt;
    DataObj  dataObj;
    boolean  isReady;
    Mesg     action;
    DateTime modifyDt;
};
typedef sequence<Inbox> InboxSeq;

struct ErrorMessage {
    string mesg;
};
typedef sequence<ErrorMessage> ErrorMessageSeq;

struct ResultStatus {
    boolean              result; //Indicates the success or failure of a request
    ErrorMessageSeq  errMesgList;
};
```

<u>การประกาศเอ็กซ์เซ็พชัน</u> ประกอบด้วย

```
exception InvalidUserID      { UserID userID; };
exception InvalidPassword    { Password password; };
exception InvalidTraderName  { TraderNM trader; };
exception InvalidCatalogueID  { CatalogueID catalogueID; };
exception NotFoundCatalogueID { CatalogueID catalogueID; };
exception InvalidCatalogue   { Catalogue catalogue; };
exception InvalidTypeDescSeq  { TypeDescSeq typeDescSeq; };
exception InvalidOfferDescSeq { OfferDescSeq offerDescSeq; };
exception InvalidContractID  { ContractID contractID; };
exception NotFoundContractID  { ContractID contractID; };
exception InvalidContract    { Contract contract; };
exception InvalidLinkName    { TraderNM linkName; };
exception InvalidInboxID     { InboxID inboxID; };
exception NotFoundInboxID    { InboxID inboxID; };
exception InvalidInbox       { Inbox inbox; };
exception InvalidMesgType    { MesgType mesgType; };
exception InvalidDateTime    { DateTime datetime; };
exception InvalidObject      { DataObj dataObj; };
```

1.ส่วนต่อประสานของผู้ดูแลการเชื่อมต่อ

interface FederationAdmin : TraderComponents {

เป็นส่วนต่อประสานที่สืบทอดมาจากส่วนต่อประสาน TraderComponents เพื่อใช้ในการจัดการบัญชีรายชื่อบริการ และแลกเปลี่ยนสัญญาข้อตกลง ซึ่งอธิบายได้ดังนี้

ตัวกระทำการของส่วนต่อประสาน

- addCatalogue()

เป็นตัวกระทำการที่ใช้สำหรับเพิ่มข้อมูลบัญชีรายชื่อบริการ ซึ่งประกอบด้วย ข้อมูลรหัสบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการ และอาร์เรย์ของข้อมูลข้อเสนอบริการที่จะทำการส่งออกโฆษณา ดังรายละเอียดต่อไปนี้

```
CatalogueID addCatalogue (in TypeDescSeq  typeList,
                          in OfferDescSeq offerList);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- typeList คือ อาร์เรย์ของข้อมูลชนิดบริการที่จะทำการโฆษณา
- offerList คือ อาร์เรย์ของข้อมูลข้อเสนอบริการที่จะทำการโฆษณา

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- CatalogueID คือ ข้อมูลรหัสบัญชีรายชื่อบริการ ซึ่งมีรูปแบบ CTL#9999999

- destroyCatalogue()

เป็นตัวกระทำการที่ใช้สำหรับยกเลิกข้อมูลบัญชีรายชื่อบริการออกจากระบบ ดังรายละเอียดต่อไปนี้

```
void destroyCatalogue (in CatalogueID catalogueID)
    raises (InvalidCatalogueID,
            NotFoundCatalogueID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- catalogueID คือ ข้อมูลรหัสบัญชีรายชื่อบริการ

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสบัญชีรายชื่อบริการไม่ถูกต้อง
- NotFoundCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบข้อมูลรหัสบัญชีรายชื่อบริการ

- describeCatalogue()

เป็นตัวกระทำการที่ใช้สำหรับแสดงข้อมูลบัญชีรายชื่อบริการที่มีในระบบ ดังรายละเอียดต่อไปนี้

```
Catalogue describeCatalogue (in CatalogueID catalogueID)
        raise (InvalidCatalogueID,
            NotFoundCatalogueID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- catalogueID คือ ข้อมูลรหัสบัญชีรายชื่อบริการ

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสบัญชีรายชื่อบริการไม่ถูกต้อง
- NotFoundCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบข้อมูลรหัสบัญชีรายชื่อบริการ

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- Catalogue คือ ข้อมูลบัญชีรายชื่อบริการ ซึ่งประกอบด้วย ข้อมูลรหัสบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการ และอาร์เรย์ของข้อมูลข้อเสนอบริการ

- modifyCatalogue()

เป็นตัวกระทำการที่ใช้สำหรับแก้ไขข้อมูลบัญชีรายชื่อบริการที่ได้ทำการโฆษณาไว้แล้ว ดังรายละเอียดต่อไปนี้

```
void modifyCatalogue (in CatalogueID catalogueID,
                      in Catalogue   catalogue)
  raises (InvalidCatalogueID,
        NotFoundCatalogueID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- catalogueID คือ ข้อมูลรหัสบัญชีรายชื่อบริการ
- catalogue คือ ข้อมูลบัญชีรายชื่อบริการที่จะทำการแก้ไข

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสบัญชีรายชื่อบริการไม่ถูกต้อง
- NotFoundCatalogueID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบข้อมูลรหัสบัญชีรายชื่อบริการ

- getLocalCatalogueID()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของรหัสบัญชีรายชื่อบริการในระบบ ดังรายละเอียดต่อไปนี้

```
CatalogueIDSeq getLocalCatalogueID();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- CatalogueIDSeq คือ อาร์เรย์ของข้อมูลรหัสบัญชีรายชื่อบริการ

- getLocalCatalogue()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลบัญชีรายชื่อบริการ ดังรายละเอียดต่อไปนี้

```
CatalogueSeq getLocalCatalogue();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- CatalogueSeq คือ อาร์เรย์ของข้อมูลรายการบัญชีรายชื่อบริการ

- addExportContract()

เป็นตัวกระทำการที่ใช้สำหรับเพิ่มข้อมูลสัญญาข้อตกลง ซึ่งประกอบด้วย ข้อมูลรหัสส่งออกสัญญา ข้อมูลรหัสนำเข้าสัญญา ข้อมูลบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ และชื่อของเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
ContractID addExportContract (in ContractID      impContractID,
                              in Catalogue       catalogue,
                              in TypeDescSeq  typeReq,
                              in OfferDescSeq offerReq,
                              in TraderNM        expTrader)
                raises (InvalidContractID,
                        InvalidCatalogue,
                        InvalidTypeDescSeq,
                        InvalidOfferDescSeq,
                           InvalidTraderName);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- impContractID คือ ข้อมูลรหัสนำเข้าสัญญาข้อตกลง ซึ่งถูกสร้างโดยเทรดเดอร์นำเข้า มีรูปแบบ IMP#9999999
- catalogue คือ ข้อมูลบัญชีรายชื่อบริการที่ส่งออก
- typeReq คือ อาร์เรย์รายการข้อมูลชนิดบริการที่ร้องขอ
- offerReq คือ อาร์เรย์รายการข้อมูลข้อเสนอบริการที่ร้องขอ
- expTrader คือ ชื่อเทรดเดอร์ที่ทำการส่งออกบริการ

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสนำเข้าสัญญาข้อตกลงไม่ถูกต้อง
- InvalidCatalogue เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบข้อมูลบัญชีรายชื่อบริการไม่ถูกต้อง
- InvalidTypeDescSeq เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบรายการข้อมูลชนิดบริการไม่ถูกต้อง
- InvalidOfferDescSeq เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรายการข้อมูลข้อเสนอบริการไม่ถูกต้อง
- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- ContractID คือ ข้อมูลรหัสส่งออกสัญญาข้อตกลง ซึ่งจะถูกสร้างโดยเทรดเดอร์ส่งออก มีรูปแบบ EXP#9999999

■ destroyExportContract()

เป็นตัวกระทำการที่ใช้สำหรับยกเลิกข้อมูลสัญญาข้อตกลงออกจากระบบ ดังรายละเอียดต่อไปนี้

```
void destroyExportContract (in ContractID expContractID,
                            in ContractID impContractID)
    raises (InvalidContractID,
            NotFoundContractID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- expContractID คือ ข้อมูลรหัสส่งออกสัญญาข้อตกลง
- impContractID คือ ข้อมูลรหัสนำเข้าสัญญาข้อตกลง

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

- describeExportContract()

เป็นตัวกระทำการที่ใช้สำหรับแสดงข้อมูลสัญญาข้อตกลงที่มีในระบบ ดังรายละเอียดต่อไปนี้

```
Contract describeExportContract (in ContractID expContractID,
                                 in ContractID impContractID)
        raises (InvalidContractID,
                NotFoundContractID);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- expContractID คือ ข้อมูลรหัสส่งออกสัญญาข้อตกลง
- impContractID คือ ข้อมูลรหัสนำเข้าสัญญาข้อตกลง

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

<u>ค่าคืนกลับของตัวกระทำการ</u> ประกอบด้วย

- Contract คือ ข้อมูลสัญญาข้อตกลง ซึ่งประกอบด้วย ข้อมูลรหัสส่งออกสัญญา ข้อมูลรหัสนำเข้าสัญญา ข้อมูลบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ และชื่อของเทรดเดอร์ส่งออก

- modifyExportContract()

เป็นตัวกระทำการที่ใช้แก้ไขข้อมูลสัญญาข้อตกลงที่ได้ทำสัญญาไว้แล้ว ดังรายละเอียดต่อไปนี้

```
void modifyExportContract (in Contract contract)
    raises (InvalidContract,
            NotFoundContractID);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- contract คือ ข้อมูลส่งออกสัญญาข้อตกลง ซึ่งประกอบไปด้วย ข้อมูลรหัสส่งออกสัญญา ข้อมูลรหัสนำเข้าสัญญา ข้อมูลบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ และชื่อของเทรดเดอร์ส่งออก

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

- getLocalExportContractID()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลรหัสส่งออกสัญญาข้อตกลง ดังรายละเอียดต่อไปนี้

```
ContractIDSeq getLocalExportContractID();
```

<u>ค่าคืนกลับของตัวกระทำการ</u> ประกอบด้วย

- ContractIDSeq คือ อาร์เรย์ของข้อมูลรหัสส่งออกสัญญาข้อตกลง

- getLocalExportContract()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลส่งออกสัญญาข้อตกลง ดังรายละเอียดต่อไปนี้

```
ContractSeq getLocalExportContract();
```

<u>ค่าคืนกลับของตัวกระทำการ</u> ประกอบด้วย

- ContractSeq คือ อาร์เรย์ของข้อมูลส่งออกสัญญาข้อตกลง

- addImportContract()

เป็นตัวกระทำการที่ใช้สำหรับเพิ่มข้อมูลสัญญาข้อตกลง ซึ่งประกอบด้วย ข้อมูลรหัสสัญญาส่งออก ข้อมูลรหัสสัญญานำเข้า ข้อมูลบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ และชื่อเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
ContractID addImportContract (in Catalogue      catalogue,
                              in TypeDescSeq  typeReq,
                              in OfferDescSeq offerReq,
                              in TraderNM     expTrader)
```

```
raises (InvalidCatalogue,
        InvalidTypeDescSeq,
        InvalidOfferDescSeq,
        InvalidTraderName);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- catalogue คือ ข้อมูลบัญชีรายชื่อบริการที่ส่งออก
- typeReq คือ อาร์เรย์ของข้อมูลรายการชนิดบริการที่ร้องขอ
- offerReq คือ อาร์เรย์ของข้อมูลรายการข้อเสนอบริการที่ร้องขอ
- expTrader คือ ชื่อของเทรดเดอร์ส่งออก

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidCatalogue เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลบัญชีรายชื่อบริการไม่ถูกต้อง
- InvalidTypeDescSeq เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรายการข้อมูลชนิดบริการไม่ถูกต้อง
- InvalidOfferDescSeq เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรายการข้อมูลข้อเสนอบริการไม่ถูกต้อง
- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- ContractID คือ ข้อมูลรหัสนำเข้าสัญญา ซึ่งจะถูกสร้างโดยเทรดเดอร์นำเข้า โดยมีรูปแบบ IMP#9999999

หมายเหตุ รหัสส่งออกสัญญาที่ถูกเก็บไว้ในสัญญานำเข้า จะให้ค่าโดยปริยายเป็น EXP#? ก่อน เนื่องจากจะต้องรอผลการอนุมัติจากผู้ดูแลเทรดเดอร์ส่งออกเสียก่อน แล้วจึงทำการเรียกใช้ตัวกระทำการ applyImportContract() เพื่อแก้ไขรหัสส่งออกสัญญาในสัญญานำเข้าให้สอดคล้องกับในสัญญาส่งออกที่เก็บไว้ในเทรดเดอร์ส่งออก

- destroyImportContract()

เป็นตัวกระทำการที่ใช้สำหรับยกเลิกข้อมูลสัญญาข้อตกลงออกจากระบบ ดังรายละเอียดต่อไปนี้

```
void destroyImportContract (in ContractID impContractID,
                            in ContractID expContractID,
    raises (InvalidContractID,
            NotFoundContractID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- expContractID คือ ข้อมูลรหัสส่งออกสัญญาข้อตกลง
- impContractID คือ ข้อมูลรหัสนำเข้าสัญญาข้อตกลง

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบข้อมูลรหัสสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

- describeImportContract()

เป็นตัวกระทำการที่ใช้สำหรับแสดงข้อมูลสัญญาข้อตกลงที่มีในระบบ ดังรายละเอียดต่อไปนี้

```
Contract describeImportContract (in ContractID impContractID,
                                 in ContractID expContractID)
    Raises (InvalidContractID,
            NotFoundContractID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- impContractID คือ ข้อมูลรหัสนำเข้าสัญญาข้อตกลง
- expContractID คือ ข้อมูลรหัสส่งออกสัญญาข้อตกลง

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- Contract คือ ข้อมูลสัญญาข้อตกลง ซึ่งประกอบด้วย ข้อมูลรหัสส่งออกสัญญา ข้อมูลรหัสนำเข้าสัญญา ข้อมูลบัญชีรายชื่อบริการ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ และชื่อของเทรดเดอร์ส่งออก

- modifyImportContract()

เป็นตัวกระทำการที่ใช้แก้ไขข้อมูลสัญญาข้อตกลงที่ได้ทำสัญญาไว้แล้ว ดังรายละเอียดต่อไปนี้

```
void modifyImportContract (in Contract contract)
    raises (InvalidContract,
            NotFoundContractID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- contract คือ ข้อมูลสัญญาข้อตกลง

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

- applyImportContract()

เป็นตัวกระทำการที่ใช้ในการแก้ไขข้อมูลรหัสสัญญาส่งออกในสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก ซึ่งตัวกระทำการนี้จะมีการเรียกใช้ตัวกระทำการ register() เพื่อทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
void applyImportContract (in Contract contract)
    raises (InvalidContract,
            NotFoundContractID);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- contract คือ ข้อมูลสัญญาข้อตกลง

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลสัญญาข้อตกลงไม่ถูกต้อง
- NotFoundContractID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสสัญญาข้อตกลง

- getLocalImportContractID()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลรหัสนำเข้าสัญญาข้อตกลง ดังรายละเอียดต่อไปนี้

```
ContractIDSeq getLocalImportContractID();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- ContractIDSeq คือ อาร์เรย์ของข้อมูลรหัสนำเข้าสัญญาข้อตกลง

- getLocalImportContract()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลนำเข้าสัญญาข้อตกลง ดังรายละเอียดต่อไปนี้

```
ContractSeq getLocalImportContract();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- ContractSeq คือ อาร์เรย์ของข้อมูลนำเข้าสัญญาข้อตกลง

- addInbox()

เป็นตัวกระทำการที่ใช้สำหรับเพิ่มข้อความแจ้งรายละเอียดในกล่องข้อความขาเข้าที่เกิดจากการแลกเปลี่ยนสัญญาที่ผู้ดูแลเทรดเดอร์ส่งออกและผู้ดูแลเทรดเดอร์นำเข้าได้ทำการเรียกใช้ ดังรายละเอียดต่อไปนี้

```
InboxID addInbox (in MesgType mesgType,
                  in TraderNM trader,
                  in UserID   sender,
                  in DateTime sendDt,
                   in DataObj  dataObj,
                   in Mesg     action)
        raises (InvalidMesgType,
                InvalidTraderName,
                InvalidUserID,
                InvalidDateTime,
                InvalidObject);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- trader คือ ชื่อเทรดเดอร์ที่ทำการส่งข้อความ
- sender คือ ชื่อผู้ส่งข้อความหรือผู้ดูแลเทรดเดอร์ที่เรียกใช้ตัวกระทำการ
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- dataObj คือ วัตถุที่ทำการส่งมาพร้อมกับข้อความที่แจ้ง ซึ่งประกอบด้วย ประเภทข้อมูล ข้อความที่แจ้ง บัญชีรายชื่อบริการ และสัญญาข้อตกลง โดยที่
  1. ถ้าประเภทข้อมูลเป็น MSG (Message) ข้อความที่แจ้งจะมีค่า ส่วนบัญชีรายชื่อบริการจะมีค่าเป็น null และสัญญาข้อตกลงก็มีค่าเป็น null

2. ถ้าประเภทข้อมูลเป็น CTL (Catalogue) ข้อความที่แจ้งจะมีค่า บัญชีรายชื่อบริการจะมีค่า ส่วนสัญญาข้อตกลงจะมีค่าเป็น null
3. ถ้าประเภทข้อมูลเป็น CTR (Contract) ข้อความแจ้งจะมีค่า บัญชีรายชื่อบริการจะมีค่าเป็น null ส่วนสัญญาข้อตกลงจะมีค่า

- action คือ สิ่งที่จะดำเนินการหรือกระทำ เมื่อได้รับข้อความแจ้ง

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความไม่ถูกต้อง
- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นชื่อของเทรดเดอร์ที่ส่งข้อความไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อผู้ส่งข้อความหรือผู้ทำการเรียกใช้ตัวกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidObject เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวัตถุที่ทำการส่งไม่ถูกต้อง

<u>ค่าคืนกลับของตัวกระทำการ</u> ประกอบด้วย

- InboxID คือ ข้อมูลรหัสข้อความขาเข้า

- destroyInbox()

เป็นตัวกระทำการที่ใช้สำหรับลบข้อมูลข้อความที่แจ้งรายละเอียดในกล่องข้อความขาเข้า ดังรายละเอียดต่อไปนี้

```
void destroyInbox (in InboxID inboxID)
    raises (InvalidInboxID,
        NotFoundInboxID);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- inboxID คือ ข้อมูลรหัสข้อความขาเข้า

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสข้อความขาเข้าไม่ถูกต้อง
- NotFoundInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสข้อความขาเข้า

- describeInbox()

เป็นตัวกระทำการที่ใช้สำหรับแสดงข้อมูลข้อความที่แจ้งรายละเอียดในกล่องข้อความขาเข้า ดังรายละเอียดต่อไปนี้

```
Inbox describeInbox (in InboxID inboxID)
        raises (InvalidInboxID,
                NotFoundInboxID);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- inboxID คือ ข้อมูลรหัสข้อความขาเข้า

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสข้อความขาเข้าไม่ถูกต้อง
- NotFoundInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสข้อความขาเข้า

- modifyInbox()

เป็นตัวกระทำการที่ใช้สำหรับแก้ไขข้อมูลข้อความแจ้งรายละเอียดในกล่องข้อความขาเข้า ดังรายละเอียดต่อไปนี้

```
void modifyInbox (in InboxID inboxID,
                  In Mesg    action)
    raises (InvalidInboxID,
            NotFoundInboxID);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- inboxID คือ ข้อมูลรหัสข้อความนำเข้า
- action คือ ข้อความบอกวิธีการที่จะดำเนินการหรือกระทำต่อไป เมื่อได้รับข้อความ

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสข้อความนำเข้าไม่ถูกต้อง
- NotFoundInboxID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสข้อความนำเข้า

- getLocalInboxID()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลรหัสข้อความขาเข้า ดังรายละเอียดต่อไปนี้

```
InboxIDSeq getLocalInboxID();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- InboxIDSeq คือ อาร์เรย์ของข้อมูลรหัสข้อความขาเข้า

- getLocalInbox()

เป็นตัวกระทำการที่ใช้สำหรับแสดงอาร์เรย์ของข้อมูลข้อความแจ้งทั้งหมดในกล่องข้อความขาเข้า ดังรายละเอียดต่อไปนี้

```
InboxSeq getLocalInbox();
```

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- InboxSeq คือ อาร์เรย์ของข้อมูลข้อความแจ้งในกล่องข้อความขาเข้า

- register()

เป็นตัวกระทำการที่ใช้ข้อมูลบริการที่ร้องขอจากสัญญาข้อตกลง เพื่อใช้ในการลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการ จากตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก และทำการสร้างตัวเรียกกลับนำเข้า (ImportCallback) เพื่อรอรับแจ้งเหตุการณ์การเปลี่ยนแปลงบริการจากตัวแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
void register (in   TraderNM                  expTrader,
               in   scncomm::SubscriberID subscriberID,
               in   scncomm::Password      password,
               out ResultStatus               resultStatus)
    raises (scncomm::DuplicateSubscriberID,
            scncomm::InvalidPassword);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- subscriberID คือ ข้อมูลรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- passwd คือ ข้อมูลรหัสผ่านของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- resultStatus คือ พารามิเตอร์ที่ส่งไปเพื่อรับค่าคืนกลับ

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- DuplicateSubscriberID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการซ้ำกับรหัสของผู้อื่นที่ลงทะเบียนไว้ก่อนหน้านี้
- InvalidPassword เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อข้อมูลรหัสผ่านไม่ถูกต้อง

- unregister()

เป็นตัวกระทำการที่ใช้ในการยกเลิกการขอรับทราบการเปลี่ยนแปลงบริการจากตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออกและทำลายตัวเรียกกลับนำเข้า ดังรายละเอียดต่อไปนี้

```
void unregister (in   TraderNM                expTrader,
                 in   scncomm::SubscriberID subscriberID,
                 in   scncomm::Password     password,
                 out ResultStatus           resultStatus);
    raises (scncomm::NotFoundSubscriberID,
            scncomm::InvalidPassword);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- subscriberID คือ ข้อมูลรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- passwd คือ ข้อมูลรหัสผ่านของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- resultStatus คือ พารามิเตอร์ที่ส่งไปเพื่อรับค่าคืนกลับ

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- NotFoundSubscriberID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- InvalidPassword เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสผ่านไม่ถูกต้อง

- distributeCatalogue()

เป็นตัวกระทำการสำหรับผู้ดูแลเทรดเดอร์ส่งออกเรียกใช้ เพื่อส่งออกบริการไปให้กับผู้ดูแลเทรดเดอร์นำเข้า ให้ทำการเลือกบริการจากบัญชีรายชื่อบริการ และทำการสร้างสัญญานำเข้า โดยตัวกระทำการ distributeCatalogue() จะทำการเรียกใช้ตัวกระทำการ sendCatalogue() ของส่วนต่อประสาน FederationContract ของเทรดเดอร์นำเข้า ซึ่งส่วนต่อประสาน FederationContract จะทำการเรียกใช้ตัวกระทำการ addInbox() จากส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำ

เข้า เพื่อแจ้งข้อความรายละเอียดของตัวกระทำการไปยังกล่องข้อความขาเข้าให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ ดังรายละเอียดต่อไปนี้

```
void distributeCatalogue (in  TraderNM     impTrader,
                          in  UserID       userID,
                          in  MesgType     mesgType,
                          in  DateTime     sendDt,
                          in  Mesg         mesg,
                          in  Catalogue    catalogue,
                          out ResultStatus resultStatus)
raises (InvalidTraderName,
        InvalidUserID,
        InvalidMesgType,
        InvalidDateTime,
        InvalidCatalogue);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- impTrader คือ ชื่อของเทรดเดอร์นำเข้า
- userID คือ ชื่อผู้ดูแลเทรดเดอร์ส่งออก
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความแจ้ง
- mesg คือ ข้อความที่อธิบายตัวกระทำการ
- catalogue คือ บัญชีรายชื่อบริการที่ส่งออก
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นออบเจ็กต์โครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความแจ้งไม่ถูกต้อง
- InvalidCatalogue เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อบัญชีรายชื่อบริการไม่ถูกต้อง

- requestCatalogue()

เป็นตัวกระทำการสำหรับผู้ดูแลเทรดเดอร์นำเข้าร้องขอให้ผู้ดูแลเทรดเดอร์ส่งออกทำการส่งบัญชีรายชื่อบริการมาให้ โดยตัวกระทำการ requestCatalogue() จะทำการเรียกใช้ตัวกระทำการ getCatalogue() ของส่วนต่อประสาน FederationContract ของเทรดเดอร์ส่งออก ซึ่งส่วนต่อประสาน FederationContract จะทำการเรียกใช้ตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก เพื่อแจ้งข้อความรายละเอียดของตัวกระทำการไปยังกล่องข้อความขาเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ ดังรายละเอียดต่อไปนี้

```
void requestCatalogue (in  TraderNM     expTrader,
                       in  UserID       userID,
                       in  MesgType     mesgType,
                       in  DateTime     sendDt,
                       in  Mesg         mesg,
                       out ResultStatus resultStatus)
    raises (InvalidTraderName,
            InvalidUserID,
            InvalidMesgType,
            InvalidDateTime);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- userID คือ ชื่อผู้ดูแลเทรดเดอร์นำเข้า
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์นำเข้าไม่ถูกต้อง

- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง

- establishFed()

เป็นตัวกระทำการสำหรับผู้ดูแลเทรดเดอร์นำเข้าร้องขอให้ผู้ดูแลเทรดเดอร์ส่งออกทำการสร้างหลักฐานเกี่ยวกับสัญญาข้อตกลงขึ้น โดยตัวกระทำการ establishFed() จะทำการเรียกใช้ตัวกระทำการ exchangeContract() ของส่วนต่อประสาน FederationContract ของเทรดเดอร์ส่งออก ซึ่งส่วนต่อประสาน FederationContract จะทำการเรียกใช้ตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก เพื่อแจ้งข้อความรายละเอียดของตัวกระทำการไปยังกล่องข้อความขาเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ ดังรายละเอียดต่อไปนี้

```
void establishFed (in  TraderNM     expTrader,
                   in  UserID       userID,
                   in  MesgType     mesgType,
                   in  DateTIme     sendDt,
                   in  Mesg         mesg,
                   in  Contract     contract,
                   out ResultStatus resultStatus)
    raises (InvalidTraderName,
            InvalidUserID,
            InvalidMesgType,
            InvalidDateTime,
            InvalidContract);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- userID คือ ชื่อผู้ดูแลเทรดเดอร์นำเข้า
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- contract คือ สัญญาข้อตกลงที่ผู้ดูแลเทรดเดอร์นำเข้าร้องขอให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติ

- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบสัญญาข้อตกลงไม่ถูกต้อง

- evaluateContract()

เป็นตัวกระทำการสำหรับผู้ดูแลเทรดเดอร์ส่งออกเรียกใช้ในการอนุมัติ หรือปฏิเสธคำร้องขอบริการในสัญญาข้อตกลงที่ส่งมาจากผู้ดูแลเทรดเดอร์นำเข้า และส่งกลับไปให้กับผู้ดูแลเทรดเดอร์นำเข้าทราบ เพื่อสร้างสัญญาฉบับสมบูรณ์ต่อไป โดยตัวกระทำการ evaluateContract() จะทำการเรียกใช้ตัวกระทำการ evaluateResult() ของส่วนต่อประสาน FederationContract ของเทรดเดอร์นำเข้า ซึ่งส่วนต่อประสาน FederationContract จะทำการเรียกใช้ตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin เพื่อแจ้งข้อความรายละเอียดของตัวกระทำการไปยังกล่องข้อความขาเข้าให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ ดังรายละเอียดต่อไปนี้

```
void evaluateContract (in  TraderNM     impTrader,
                       in  UserID       userID,
                       in  MesgType     mesgType,
                       in  DateTime     sendDt,
                       in  Mesg         mesg,
                       in  Contract     contract,
                       out ResultStatus resultStatus)
    raises (InvalidTraderName,
            InvalidUserID,
            InvalidMesgType,
            InvalidDateTime,
            InvalidContract);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- impTrader คือ ชื่อของเทรดเดอร์นำเข้า
- userID คือ ชื่อผู้ดูแลเทรดเดอร์ส่งออก

- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- contract คือ สัญญาข้อตกลงที่ผู้ดูแลเทรดเดอร์ส่งออกส่งไปให้ผู้ดูแลเทรดเดอร์นำเข้าทราบผลการพิจารณาอนุมัติ
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นออบเจ็กต์โครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบสัญญาข้อตกลงไม่ถูกต้อง

2. ส่วนต่อประสานของการทำสัญญา

```
interface FederationContract : TraderComponents {
```

เป็นส่วนต่อประสานของการทำสัญญาระหว่างเทรดเดอร์ สืบทอดมาจากส่วนต่อประสาน TraderComponents โดยจะถูกเรียกใช้จากส่วนต่อประสานผู้ดูแลการเชื่อมต่อเทรดเดอร์ระยะไกล โดยอธิบายได้ดังนี้

ตัวกระทำการของส่วนต่อประสาน

- register()

เป็นตัวกระทำการที่ถูกเรียกใช้จากตัวกระทำการ register() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า เพื่อขอลงทะเบียนรับทราบการเปลี่ยนแปลงจากตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
scncomm::AdminSubscriber register(in  scncomm::SubscriberID  subscriberID,
                                  in  scncomm::Password      password,
                                  out ResultStatus           resultStatus)
    raises (scncomm::InvalidPassword,
            scncomm::ExceedSubscriberLimit,
            scncomm::ExceedConnectSubscriberLimit,
            scncomm::DuplicateSubscriberID,
            scncomm::UnsupportedQos);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- subscriberID คือ ข้อมูลรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- passwd คือ ข้อมูลรหัสผ่านของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidPassword เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสผ่านไม่ถูกต้อง
- ExceedSubscriberLimit เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อจำนวนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการเกินจำนวนที่ตั้งไว้
- ExceedConnectSubscriberLimit เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อจำนวนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการเชื่อมต่อเกินกำหนดที่ตั้งไว้
- DuplicateSubscriberID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการซ้ำกับรหัสของผู้อื่นที่ลงทะเบียนไว้ก่อนหน้านี้
- UnsupportedQos เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อ**ชื่อของคุณสมบัติและชนิดของค่าข้อมูลที่ระบุให้กับคุณสมบัตินั้นๆ** ไม่ถูกต้องตามคุณภาพของบริการ

<u>ค่าคืนกลับของตัวกระทำการ</u> ประกอบด้วย

- AdminSubscriber คือ วัตถุผู้ดูแล (Admin Subscriber) ที่ใช้ในการสร้างและดูแลตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงสำหรับรายละเอียดส่วนต่อประสานของ AdminSubscriber

- unregister()

เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ unregister() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า เพื่อขอยกเลิกการขอรับทราบการเปลี่ยนแปลงจากตัว

จัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก ดังรายละเอียดต่อไปนี้

```
void unregister (in  scncomm::SubscriberID  subscriberID,
                 in  scncomm::Password      password,
                 out ResultStatus           resultStatus)
    raises (scncomm::NotFoundSubscriberID,
            scncomm::InvalidPassword);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- subscriberID คือ ข้อมูลรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- passwd คือ ข้อมูลรหัสผ่านของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- NotFoundSubscriberID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- InvalidPassword เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสผ่านไม่ถูกต้อง

- getAdminSubscriber()

เป็นตัวกระทำการที่ใช้สำหรับผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ ใช้ในการขอข้อมูลอ้างอิงวัตถุของผู้ดูแล (Admin Subscriber) ดังรายละเอียดต่อไปนี้

```
scncomm::AdminSubscriber getAdminSubscriber (in  scncomm::SubscriberID  subscriberID,
                                              in  scncomm::Password      password,
                                              out ResultStatus           resultStatus)
    raises (scncomm::NotFoundSubscriberID,
            scncomm::InvalidPassword);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- subscriberID คือ ข้อมูลรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- passwd คือ ข้อมูลรหัสผ่านของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- NotFoundSubscriberID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อไม่พบรหัสของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
- InvalidPassword เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรหัสผ่านไม่ถูกต้อง

ค่าคืนกลับของตัวกระทำการ ประกอบด้วย

- AdminSubscriber คือ ออบเจ็กต์ตัวแทน (Admin Subscriber) ที่สร้างขึ้นจากการลงทะเบียนของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ (ตัวกระทำการ registerSubscriber())

- sendCatalogue()

เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ distributeCatalogue() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก สำหรับใช้ในการส่งออกบัญชีรายชื่อบริการไปให้กับผู้ดูแลเทรดเดอร์นำเข้า เพื่อเลือกบริการและสร้างสัญญาข้อตกลง ดังรายละเอียดต่อไปนี้

```
void sendCatalogue (in  TraderNM     expTrader,
                    in  UserID       userID,
                    in  MesgType     mesgType,
                    in  DateTime     sendDt,
                    in  Mesg         mesg,
                    in  Catalogue    catalogue,
                    out ResultStatus resultStatus)
   raises (InvalidTraderName,
           InvalidUserID,
           InvalidMesgType,
           InvalidDateTime,
           InvalidCatalogue);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- userID คือ ชื่อผู้ดูแลเทรดเดอร์ส่งออก
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ

- mesg คือ ข้อความที่อธิบายตัวกระทำการ distributeCatalogue
- catalogue คือ บัญชีรายชื่อบริการที่ถูกส่งมาจากตัวกระทำการ distributeCatalogue()
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidCatalogue เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อบัญชีรายชื่อบริการไม่ถูกต้อง

- getCatalogue()

เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ requestCatalogue() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า สำหรับใช้ในการส่งต่อคำร้องขอบัญชีรายชื่อบริการไปให้กับผู้ดูแลเทรดเดอร์ส่งออก เพื่อทำการส่งบัญชีรายชื่อบริการกลับไปให้ผู้ดูแลเทรดเดอร์นำเข้า ดังรายละเอียดต่อไปนี้

```
void getCatalogue (in  TraderNM     impTrader,
                   in  UserID       userID,
                   in  MesgType     mesgType,
                   in  DateTime     sendDt,
                   in  Mesg         mesg,
                   out ResultStatus resultStatus)
    raise (InvalidTraderName,
           InvalidUserID,
           InvalidMesgType,
           InvalidDateTime);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- impTrader คือ ชื่อของเทรดเดอร์นำเข้า
- userID คือ ชื่อผู้ดูแลเทรดเดอร์นำเข้า
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่

  M = Message, R = Request Catalogue, D = Distribute Catalogue,

E = Establish Contract, A = Approve Contract

- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง

- exchangeContract()

เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ establishFed() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า สำหรับใช้ในการส่งคำร้องขอบัญชีรายชื่อบริการให้กับผู้ดูแลเทรดเดอร์ส่งออก เพื่อทำการส่งบัญชีรายชื่อบริการกลับไปให้ผู้ดูแลเทรดเดอร์นำเข้า ดังรายละเอียดต่อไปนี้

```
void exchangeContract (in  TraderNM     impTrader,
                       in  UserID       userID,
                       in  MesgType     mesgType,
                       in  DateTime     sendDt,
                       in  Mesg         mesg,
                       in  Contract     contract,
                       out ResultStatus resultStatus)
  raises (InvalidTraderName,
          InvalidUserID,
          InvalidMesgType,
          InvalidDateTime,
          InvalidContract);
```

<u>พารามิเตอร์ของตัวกระทำการ</u> ประกอบด้วย

- impTrader คือ ชื่อของเทรดเดอร์นำเข้า
- userID คือ ชื่อผู้ดูแลเทรดเดอร์นำเข้า

- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- contract คือ สัญญาข้อตกลงที่ผู้ดูแลเทรดเดอร์นำเข้าร้องขอให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติต่อไป
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นออบเจ็กต์โครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u> ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์นำเข้าไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบสัญญาข้อตกลงไม่ถูกต้อง

- evaluateResult()

เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ evaluateContract ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก สำหรับใช้ในการส่งสัญญาข้อตกลงที่ได้ทำการอนุมัติโดยผู้ดูแลเทรดเดอร์ส่งออกไปให้กับผู้ดูแลเทรดเดอร์นำเข้า เพื่อทำการแก้ไขรหัสสัญญาส่งออกในสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก ดังรายละเอียดต่อไปนี้

```
void evaluateResult (in  TraderNM   expTrader,
                     in  UserID     userID,
                     in  MesgType   mesgType,
                     in  DateTIme   sendDt,
                     in  Mesg       mesg,
                     in  Contract   contract,
                     out ResultStatus resultStatus)
    raises (InvalidTraderName,
            InvalidUserID,
            InvalidMesgType,
            InvalidDateTime,
            InvalidContract);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- expTrader คือ ชื่อของเทรดเดอร์ส่งออก
- userID คือ ชื่อผู้ดูแลเทรดเดอร์ส่งออก
- mesgType คือ ตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการ ได้แก่
  M = Message, R = Request Catalogue, D = Distribute Catalogue,
  E = Establish Contract, A = Approve Contract
- sendDt คือ วันที่และเวลาที่ส่งข้อความ
- mesg คือ ข้อความที่อธิบายตัวกระทำการตัวนี้
- contract คือ สัญญาข้อตกลงที่ผู้ดูแลเทรดเดอร์ส่งออกส่งไปให้ผู้ดูแลเทรดเดอร์นำเข้าทราบผลการพิจารณาอนุมัติ
- resultStatus คือ พารามิเตอร์แบบค่าคืนกลับ ค่าที่คืนกลับมาจะมีชนิดข้อมูลเป็นวัตถุโครงสร้าง สามารถดูรายละเอียดได้จากการประกาศข้อมูลในหัวข้อที่ 4.1.1

เอ็กซ์เซ็พชันของตัวกระทำการ ประกอบด้วย

- InvalidTraderName เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อของเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidUserID เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อชื่อผู้ดูแลเทรดเดอร์ส่งออกไม่ถูกต้อง
- InvalidMesgType เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อตัวอักษรย่อชนิดของข้อความที่ถูกกระทำการไม่ถูกต้อง
- InvalidDateTime เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อวันที่และเวลาที่ส่งข้อความไม่ถูกต้อง
- InvalidContract เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อรูปแบบสัญญาข้อตกลงไม่ถูกต้อง

4.1.2 ส่วนต่อประสานของตัวเรียกกลับนำเข้า

ส่วนต่อประสานของตัวเรียกกลับนำเข้า (ImportCallback) ได้เพิ่มเข้าไปในมอดูล CosTrading ของคอร์บาเทรดเดอร์ เพื่อใช้ในการรอรับข้อมูลเหตุการณ์การเปลี่ยนแปลงบริการจากตัวแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก ซึ่งอธิบายได้ดังนี้

ตัวกระทำการของส่วนต่อประสาน

- ตัวกระทำการ sendNewEvent()

เป็นตัวกระทำการที่ใช้แจ้งเหตุการณ์ให้ผู้รับบริการทราบว่ามีบริการเกิดขึ้นใหม่ โดยใช้ข้อมูลการร้องขอบริการจากสัญญานำเข้าที่ได้รับการอนุมัติแล้วในการแจ้ง ดังรายละเอียดต่อไปนี้

```
void sendNewEvent (in TypeDescSeq typeReq,
                   in OfferDescSeq offerReq);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- typeReq คือ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอ
- offerReq คือ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอ

- ตัวกระทำการ sendChangeEvent()

เป็นตัวกระทำการที่ใช้แจ้งเหตุการณ์ให้ผู้รับบริการทราบว่ามีบริการเกิดขึ้นใหม่หรือบริการที่เคยใช้ได้รับการยกเลิก โดยอาศัยข้อมูลการร้องขอบริการจากสัญญานำเข้าเดิมและสัญญานำเข้าใหม่ที่ได้รับการแก้ไข เพื่อเป็นข้อมูลในการเปรียบเทียบว่าบริการตัวใดได้รับการร้องขอใหม่หรือบริการตัวใดได้รับการยกเลิกแล้ว

```
void sendChangeEvent (in TypeDescSeq  oldTypeReq,
                      in OfferDescSeq oldOfferReq,
                      in TypeDescSeq newTypeReq,
                      in OfferDescSeq newOfferReq);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

- oldTypeReq คือ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอเดิม
- oldOfferReq คือ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอเดิม
- newTypeReq คือ อาร์เรย์ของข้อมูลชนิดบริการที่ร้องขอใหม่
- newOfferReq คือ อาร์เรย์ของข้อมูลข้อเสนอบริการที่ร้องขอใหม่

- ตัวกระทำการ push_structured_event()

เป็นตัวกระทำการที่บริการแจ้งเหตุการณ์ใช้ในการส่งเหตุการณ์การเปลี่ยนแปลงบริการมาให้แก่ตัวแทนผู้ต้องการรับทราบการเปลี่ยนแปลงบริการ ข้อมูลการเปลี่ยนแปลงบริการที่อยู่ภายในเหตุการณ์จะถูกส่งต่อไปให้แก่ผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการต่อไป

```
void push_structured_event (in scns::scnmesg::StructuredEventMessage mesg)
    raise (scns::scncomm::Disconnected);
```

พารามิเตอร์ของตัวกระทำการ ประกอบด้วย

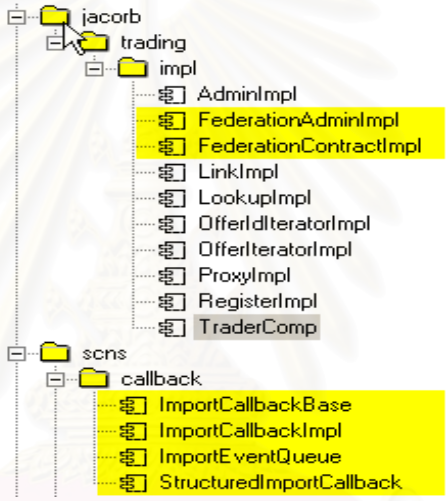
- mesg คือ รูปแบบของเหตุการณ์การเปลี่ยนแปลงบริการที่ได้รับแจ้งจากตัวแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก

<u>เอ็กซ์เซ็พชันของตัวกระทำการ</u>

- Disconnected เอ็กซ์เซ็พชันนี้เกิดขึ้นเมื่อมีการเรียกใช้งานตัวกระทำการในขณะที่ไม่ได้ติดต่ออยู่กับตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ

4.2 ต้นแบบส่วนเพิ่มขยายบริการเทรดเดอร์

ส่วนเพิ่มขยายบริการเทรดเดอร์ประกอบด้วย 2 ส่วน คือ ตัวจัดการสัญญาข้อตกลง และมอดูลตัวเรียกกลับนำเข้า โดยมีแบบจำลองแสดงดังรูปที่ 4.1 ตัวจัดการสัญญาข้อตกลงได้รับการพัฒนาขึ้นในแพคเกจ jacorb.trading.impl ส่วนมอดูลตัวเรียกกลับนำเข้านั้นได้รับการพัฒนาขึ้นในแพคเกจ scns.callback



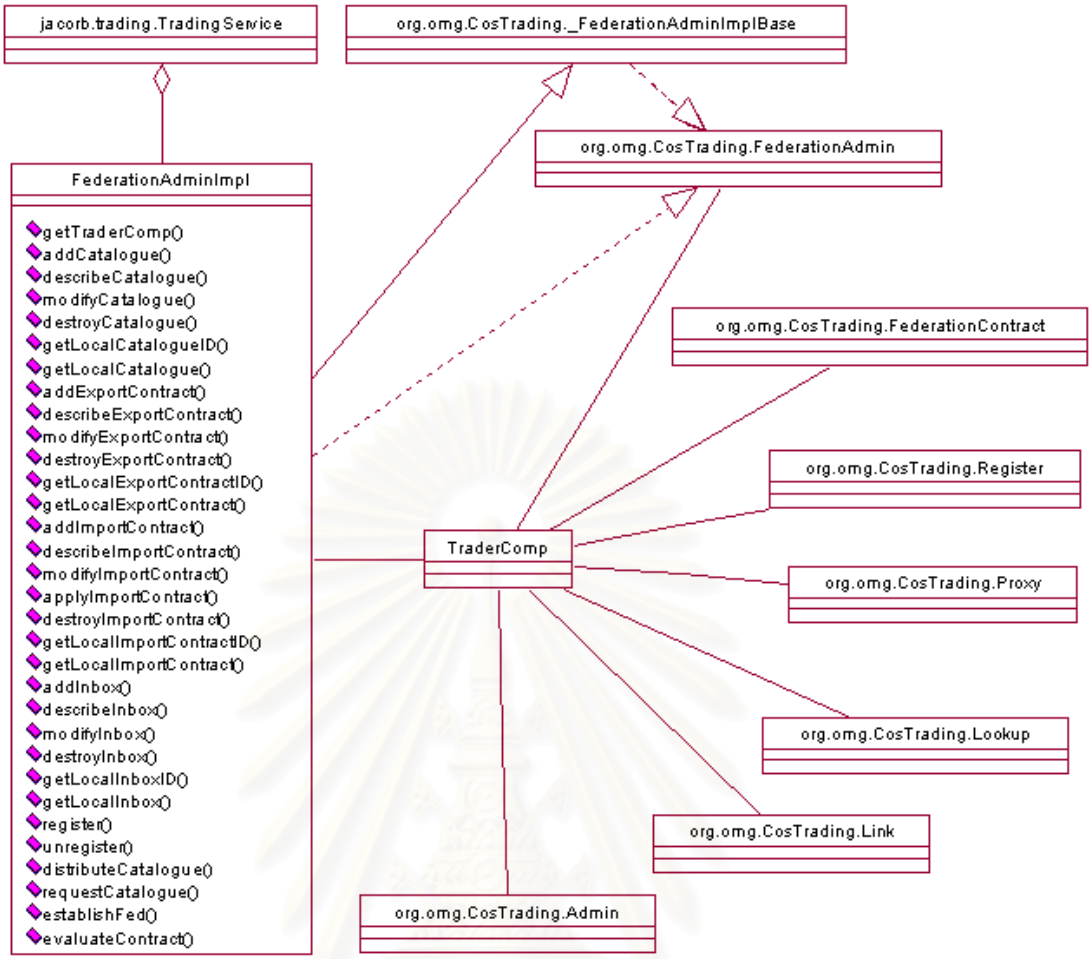
รูปที่ 4.1 แบบจำลองแพคเกจของส่วนเพิ่มขยายบริการเทรดเดอร์

4.2.1 ตัวจัดการสัญญาข้อตกลง

วิทยานิพนธ์นี้ได้พัฒนาส่วนจัดการสัญญาข้อตกลง เพื่อรองรับการจัดการสัญญาตามข้อกำหนดของโอเอ็มจี [8] ซึ่งใช้ออร์บของวิสิโบรกเกอร์ โดยผูกเข้ากับบริการเทรดเดอร์ของจาคอร์บ จากรูปที่ 4.1 ตัวจัดการสัญญาข้อตกลง (ภายในแพคเกจ jacorb.trading.impl) ประกอบด้วยคลาส 2 คลาส ได้แก่ คลาส FederationAdminImpl และคลาส FederationContractImpl ซึ่งทั้ง 2 คลาสจะทำงานร่วมกัน เพื่อจัดการกับบัญชีรายชื่อบริการ และสัญญาข้อตกลงที่เกิดจากการแลกเปลี่ยนสัญญาระหว่างผู้ดูแลเทรดเดอร์ส่งออกและผู้ดูแลเทรดเดอร์นำเข้า โดยอธิบายรายละเอียดได้ดังนี้

1. <u>คลาส FederationAdminImpl</u>

เป็นคลาสพัฒนาของผู้ดูแลการเชื่อมต่อ โดยมีแบบจำลองคลาสดังรูปที่ 4.2



รูปที่ 4.2 แบบจำลองของคลาสผู้ดูแลการเชื่อมต่อ

คลาส FederationAdminImpl สืบทอดมาจากคลาส _FederationAdminImplBase (ในแพคเกจ org.omg.CosTrading) ซึ่งเป็นเซิร์ฟเวอร์สเกเลตันของส่วนต่อประสาน FederationAdmin (ในแพคเกจ org.omg.CosTrading ตามที่กล่าวไว้ในหัวข้อที่ 4.1.1) และยังเป็นคลาสส่วนประกอบของคลาส TradingService (ในแพคเกจ jacorb.trading) โดยทำหน้าที่เกี่ยวกับการจัดการบัญชีรายชื่อบริการ และการจัดการสัญญาข้อตกลง โดยตัวกระทำการต่างๆ ของคลาส FederationAdminImpl สามารถอธิบายได้ดังนี้

ตัวกระทำการประกอบด้วย

- getTraderComp() เป็นตัวกระทำการที่ใช้ในการเรียกดูข้อมูลอ้างถึงวัตถุที่เป็นส่วนประกอบของคลาส TraderComp ได้แก่ คลาสส่วนต่อประสาน Admin คลาสส่วนต่อประสาน

FederationAdmin คลาสส่วนต่อประสาน FederationContract คลาสส่วนต่อประสาน Link คลาสส่วนต่อประสาน Lookup คลาสส่วนต่อประสาน Proxy และคลาสส่วนต่อประสาน Register

- addCatalogue() เป็นตัวกระทำการที่ใช้ในการเพิ่มข้อมูลบัญชีรายชื่อบริการที่ต้องการโฆษณา ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- describeCatalogue() เป็นตัวกระทำการที่ใช้ในการแสดงข้อมูลบัญชีรายชื่อบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- modifyCatalogue() เป็นตัวกระทำการที่ใช้ในการแก้ไขบริการในบัญชีรายชื่อบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- destroyCatalogue() เป็นตัวกระทำการที่ใช้ในการยกเลิกบัญชีรายชื่อบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- getLocalCatalogueID เป็นตัวกระทำการที่ใช้เรียกดูอาร์เรย์ของข้อมูลรหัสบัญชีรายชื่อบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- getLocalCatalogue เป็นตัวกระทำการที่ใช้เรียกดูอาร์เรย์ของข้อมูลบัญชีรายชื่อบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- addExportContract() เป็นตัวกระทำการที่ใช้เพิ่มข้อมูลสัญญาส่งออกบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- describeExportContract() เป็นตัวกระทำการที่ใช้ในการแสดงข้อมูลสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- modifyExportContract() เป็นตัวกระทำการที่ใช้ในการแก้ไขข้อมูลสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- destroyExportContract() เป็นตัวกระทำการที่ใช้ในการยกเลิกสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

- getLocalExportContractID() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อมูลรหัสสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ getLocalExportContract() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อมูลสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ addImportContract() เป็นตัวกระทำการที่ใช้เพิ่มข้อมูลสัญญานำเข้าบริการ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ describeImportContract() เป็นตัวกระทำการที่ใช้ในการแสดงข้อมูลสัญญานำเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ modifyImportContract() เป็นตัวกระทำการที่ใช้ในการแก้ไขข้อมูลสัญญานำเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ applyImportContract() เป็นตัวกระทำการที่ใช้ในการแก้ไขข้อมูลรหัสสัญญาส่งออกในสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ destroyImportContract() เป็นตัวกระทำการที่ใช้ในการยกเลิกสัญญานำเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ getLocalImportContractID() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อมูลรหัสสัญญานำเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ getLocalImportContract() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อมูลสัญญานำเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ addInbox() เป็นตัวกระทำการที่ใช้ในการส่งข้อความ เพื่อแจ้งรายละเอียดให้ผู้ดูแลเทรดเดอร์ระยะไกลทราบ ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ describeInbox() เป็นตัวกระทำการที่ใช้ในการแสดงข้อความที่อยู่ในกล่องขาเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ destroyInbox() เป็นตัวกระทำการที่ใช้ในการลบข้อความในกล่องขาเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ getLocalInboxID() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อมูลรหัสข้อความในกล่องขาเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ getLocalInbox() เป็นตัวกระทำการที่ใช้ในการเรียกดูอาร์เรย์ของข้อความในกล่องขาเข้า ดังได้กล่าวไว้แล้วในหัวข้อที่ 4.1.1

■ register() เป็นตัวกระทำการที่ถูกเรียกโดยตัวกระทำการ applyImportContract() เพื่อทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก โดยเรียกใช้ผ่านตัวกระทำการ register() ของคลาส FederationContractImpl ของเทรดเดอร์ส่งออก เพื่อให้ดำเนินการลงทะเบียนแทน หลังจากนั้นจะทำการสร้างตัวเรียกกลับนำเข้า (ImportCallback) เพื่อรอรับแจ้งเหตุการณ์การเปลี่ยนแปลงบริการจากตัวแจ้งเหตุการณ์ (EventManager) ของบริการเอสซีเอ็นที่ทำงานงานร่วมกับเทรดเดอร์ส่งออก และทำการแจ้งการเกิดบริการขึ้นใหม่ไปยังตัวจัดการแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์นำเข้าด้วย

■ unregister() เป็นตัวกระทำการที่ถูกเรียกโดยตัวกระทำการ destroyImportContract() เพื่อทำการยกเลิกการขอรับทราบการเปลี่ยนแปลงกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก โดยเรียกใช้ผ่านตัวกระทำการ unregister() ของคลาส FederationContractImpl ของเทรดเดอร์ส่งออก เพื่อให้ดำเนินการยกเลิกการลงทะเบียนแทน และทำการแจ้งการยกเลิกบริการทั้งหมดไปยังผู้รับบริการที่ต้องการรับทราบการเปลี่ยนแปลงจากตัวแจ้งเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์นำเข้า หลังจากนั้นก็จะทำลายวัตถุตัวเรียกกลับนำเข้า

■ distributeCatalogue() เป็นตัวกระทำการที่ผู้ดูแลเทรดเดอร์ส่งออกใช้ในการส่งบัญชีรายชื่อบริการไปให้ผู้ดูแลเทรดเดอร์นำเข้า เพื่อทำการเลือกบริการที่ต้องการใช้บริการ แล้วส่งสัญญาข้อตกลงกลับไปให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติต่อไป โดยตัวกระทำการ distributeCatalogue จะทำการเรียกใช้ตัวกระทำการ addInbox() ของเทรดเดอร์นำเข้า เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยเรียกผ่านตัวกระทำการ sendCatalogue() ของคลาส FederationContractImpl ของเทรดเดอร์นำเข้า

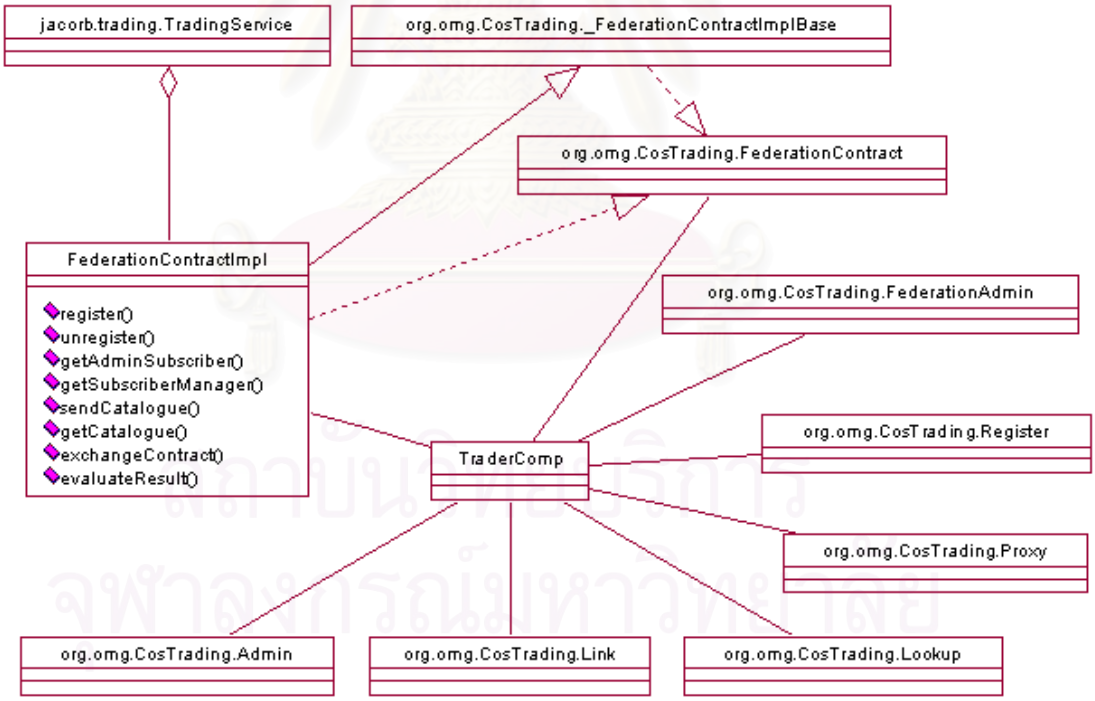
■ requestCatalogue() เป็นตัวกระทำการที่ผู้ดูแลเทรดเดอร์นำเข้าใช้ในการร้องขอให้ผู้ดูแลเทรดเดอร์ส่งออกส่งบัญชีรายชื่อบริการมาให้ตน เพื่อทำการเลือกบริการที่ต้องการใช้บริการ แล้วส่งสัญญาข้อตกลงกลับไปให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติต่อไป โดยตัวกระทำการ requestCatalogue() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของเทรดเดอร์ส่งออก เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยเรียกผ่านตัวกระทำการ getCatalogue() ของคลาส FederationContractImpl ของเทรดเดอร์ส่งออก

■ establishFed() เป็นตัวกระทำการที่ผู้ดูแลเทรดเดอร์นำเข้าใช้ในการส่งสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์ส่งออกทำการอนุมัติหรือปฏิเสธคำร้องขอบริการ โดยตัวกระทำการ establishFed() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของเทรดเดอร์ส่งออก เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยเรียกผ่านตัวกระทำการ exchangeContract() ของคลาส FederationContractImpl ของเทรดเดอร์ส่งออก

■evaluateContract() เป็นตัวกระทำการที่ผู้ดูแลเทรดเดอร์ส่งออกใช้ในการแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบผลว่าสัญญาได้รับการอนุมัติหรือไม่ โดยตัวกระทำการ evaluateContract() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของเทรดเดอร์นำเข้า เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยเรียกผ่านตัวกระทำการ evaluateResult() ของคลาส FederationContractImpl ของเทรดเดอร์นำเข้า

2. <u>คลาส FederationContractImpl</u>

เป็นคลาสพัฒนาของตัวจัดการสัญญาข้อตกลง โดยมีแบบจำลองคลาสดังรูปที่ 4.3



รูปที่ 4.3 แบบจำลองของคลาสการทำสัญญา

คลาส FederationContractImpl สืบทอดมาจากคลาส _FederationContractImplBase (ในแพคเกจ org.omg.CosTrading) ซึ่งเป็นส่วนประกอบหนึ่งของคลาส TradingService (ในแพคเกจ jacorb.trading) และได้พัฒนาตามส่วนต่อประสานของคลาส FederationContract (ในแพค

เกจ org.omg.CosTrading) โดยมีการทำงานร่วมกับคลาส FederationAdminImpl เพื่อทำหน้าที่ในการแลกเปลี่ยนสัญญาข้อตกลง โดยตัวกระทำการต่างๆ ของคลาส FederationContractmpl สามารถอธิบายได้ดังนี้

ตัวกระทำการประกอบด้วย

- register() เป็นตัวกระทำการที่ถูกเรียกโดยตัวกระทำการ register() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก แล้วส่ง**ข้อมูลอ้างถึงวัตถุของวัตถุผู้ดูแล** (AdminSubscriber) คืนกลับไปให้

- unregister() เป็นตัวกระทำการที่ถูกเรียกโดยตัวกระทำการ unregister() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อทำการยกเลิกการขอรับทราบการเปลี่ยนแปลงกับตัวจัดการผู้ที่ต้องการทราบการเปลี่ยนแปลงบริการ (SubscriberManager) ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก

- getAdminSubscriber() เป็นตัวกระทำการที่ผู้ต้องการรับทราบการเปลี่ยนแปลงบริการใช้ในการขอข้อมูลอ้างถึงวัตถุของวัตถุผู้ดูแลตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลง

- getSubscriberManager() เป็นตัวกระทำการที่ใช้ติดต่อกับข้อมูลอ้างถึงวัตถุของตัวจัดการผู้ต้องการรับทราบการเปลี่ยนแปลงบริการ

- sendCatalogue() เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ distributeCatalogue() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อทำการเลือกบริการที่ต้องการใช้บริการแล้วส่งสัญญาข้อตกลงกลับไปให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติต่อไป โดยตัวกระทำการ sendCatalogue จะทำการเรียกใช้ตัวกระทำการ addInbox() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ

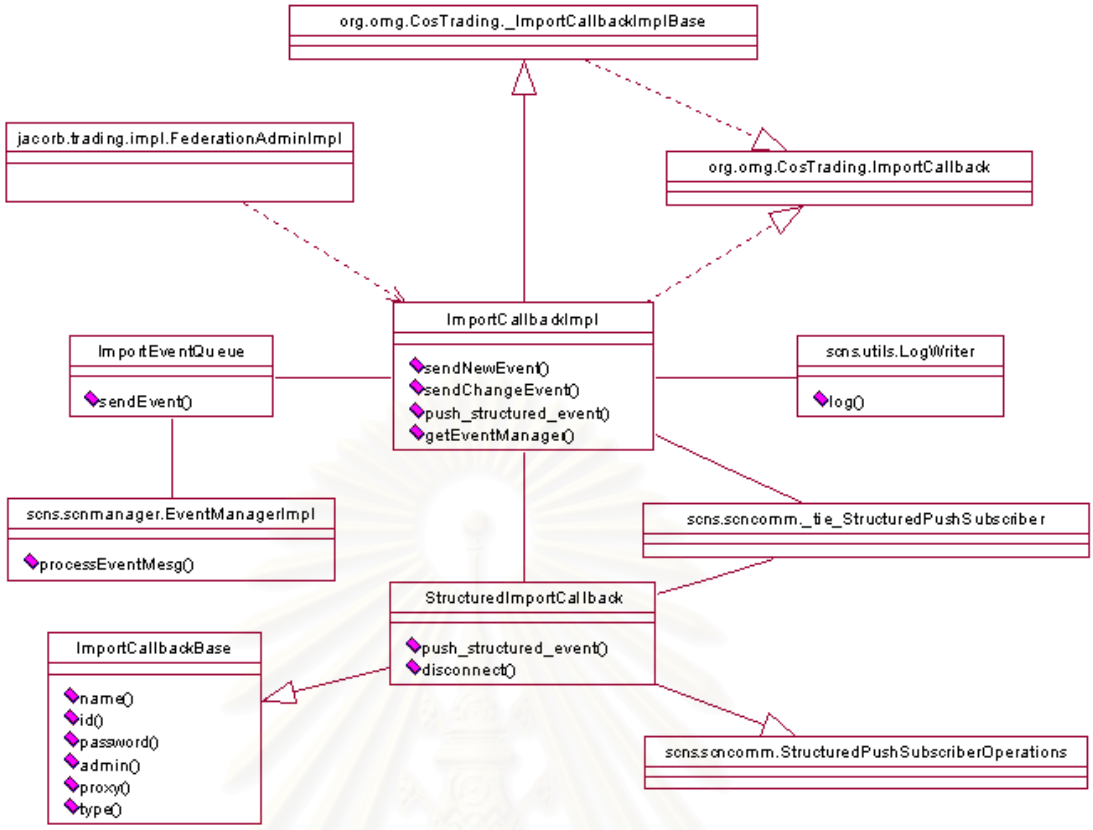
- getCatalogue() เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ requestCatalogue() ของคลาส FederationAdminImpl ของเทรดเดอร์ส่งออก เพื่อให้ผู้ดูแลเทรดเดอร์ส่งออก ทำการส่งบัญชีรายชื่อบริการกลับไปให้ผู้ดูแลเทรดเดอร์นำเข้า โดยตัวกระทำการ getCatalogue() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของคลาส FederationAdminImpl ของเทรดเดอร์ส่งออก เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ

■ exchangeContract() เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ establishFed() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อส่งสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติ โดยตัวกระทำการ exchangeContract() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของคลาส FederationAdminImpl ของเทรดเดอร์ส่งออก เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ

■ evaluateResult() เป็นตัวกระทำการที่ถูกเรียกจากตัวกระทำการ evaluateContract() ของคลาส FederationAdminImpl ของเทรดเดอร์ส่งออก เพื่ออนุมัติหรือปฏิเสธสัญญา หากสัญญานำเข้าได้รับการอนุมัติ ผู้ดูแลเทรดเดอร์ส่งออกจะจัดส่งสัญญาส่งออกที่สอดคล้องกับสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์นำเข้า โดยตัวกระทำการ evaluateResult() จะทำการเรียกใช้ตัวกระทำการ addInbox() ของคลาส FederationAdminImpl ของเทรดเดอร์นำเข้า เพื่อส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ

4.2.2 มอดูลตัวเรียกกลับนำเข้า

มอดูลตัวเรียกกลับนำเข้า (ImportCallback) ได้พัฒนามาจากตัวเรียกกลับ (Callback) ของงานวิจัย [6] โดยทำการปรับปรุงให้สามารถรับข้อมูลแจ้งเหตุการณ์การเปลี่ยนแปลงบริการจากตัวแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก โดยมอดูลตัวเรียกกลับนำเข้าประกอบด้วยคลาส 4 คลาส ได้แก่ คลาส ImportCallbackBase คลาส ImportCallbackImpl คลาส ImportEventQueue และคลาส StructuredImportCallback โดยมีแบบจำลองของคลาสดังรูปที่ 4.4



รูปที่ 4.4 แบบจำลองของคลาสตัวเรียกกลับนำเข้า

1. <u>คลาส ImportCallbackBase</u>

เป็นคลาสพื้นฐานที่เก็บข้อมูลที่จำเป็นสำหรับการลงทะเบียนและข้อมูลอ้างถึงวัตถุผู้ดูแล รวมทั้งรูปแบบของข้อมูลแจ้งการเปลี่ยนแปลง โดยอธิบายตัวกระทำการต่างๆ ได้ดังนี้

- ตัวกระทำการ name() เป็นตัวกระทำการที่เรียกดูข้อมูลชื่อของตัวเรียกกลับนำเข้า
- ตัวกระทำการ id() เป็นตัวกระทำการที่เรียกดูข้อมูลรหัสผู้ขอลงทะเบียนรับทราบการเปลี่ยนแปลง
- ตัวกระทำการ password() เป็นตัวกระทำการที่เรียกดูข้อมูลรหัสผ่านของผู้ขอลงทะเบียนรับทราบการเปลี่ยนแปลง
- ตัวกระทำการ admin() เป็นตัวกระทำการที่เรียกดูข้อมูลอ้างถึงวัตถุผู้ดูแลตัวแทน (AdminSubscriber) ผู้ที่ต้องการรับทราบการเปลี่ยนแปลง
- ตัวกระทำการ proxy() เป็นตัวกระทำการที่เรียกดูข้อมูลวัตถุตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลง
- ตัวกระทำการ type() เป็นตัวกระทำการที่เรียกดูข้อมูลประเภทของวัตถุตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลง

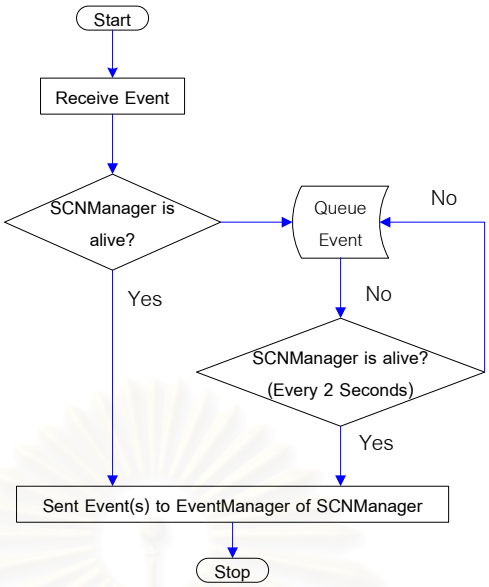
2. <u>คลาส ImportCallbackImpl</u>

คลาส ImportCallbackImpl สืบทอดมาจากคลาส _ImportCallbackImplBase (ในแพคเกจ org.omg.CosTrading) เพื่อทำหน้าที่สร้างตัวแทนผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ ซึ่งออบเจ็กต์ของคลาสนี้จะถูกสร้างและถูกทำลายโดยการเรียกใช้งานตัวกระทำการ register() และ unregister() ของคลาส FederationAdminImpl (ในแพคเกจ jacorb.trading.impl) นอกจากนี้คลาส ImportCallbackImpl ยังมีความสัมพันธ์กับคลาสต่างๆ ได้แก่

- คลาส StructuredImportCallback เป็นวัตถุตัวแทนประเภทโครงสร้างของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการที่ส่งเป็นพารามิเตอร์ไปให้คลาส _tieStructuredPushSubscriber (ในแพคเกจ scns.scncomm)
- คลาส _tie_StructuredPushSubscriber (ในแพคเกจ scns.scncomm) สืบทอดมาจากคลาส _StructuredPushSubscriberImplBase ซึ่งเป็นเซิร์ฟเวอร์สเกเลตันของส่วนต่อประสาน StructuredPushSubscriber เพื่อทำหน้ารอรับข้อมูลการเปลี่ยนแปลงจากตัวแจ้งเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก
- คลาส scns.utils.LogWriter เพื่อทำการบันทึกการทำงานและข้อผิดพลาดต่างๆ

3. <u>คลาส ImportEventQueue</u>

เป็นคลาสที่ได้รับการปรับปรุงจากคลาส EventQueue ของงานวิจัย [6] เพื่อทำหน้าที่ส่งผ่านข้อมูลการเปลี่ยนแปลงบริการที่ถูกเรียกจากตัวกระทำการ push_structured_event ของคลาส FederationAdminImpl ไปให้แก่ตัวจัดการเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์นำเข้า นอกจากการส่งผ่านข้อมูลการเปลี่ยนแปลงบริการตามปกติแล้ว คลาส ImportEventQueue ยังทำหน้าที่เก็บรักษาข้อมูลการเปลี่ยนแปลงบริการเอาไว้ในกรณีที่ตรวจพบว่าตัวจัดการการแจ้งการเปลี่ยนแปลงไม่สามารถทำงานได้ ณ เวลาหนึ่งๆ แผนภาพการทำงานเป็นดังรูปที่ 4.5

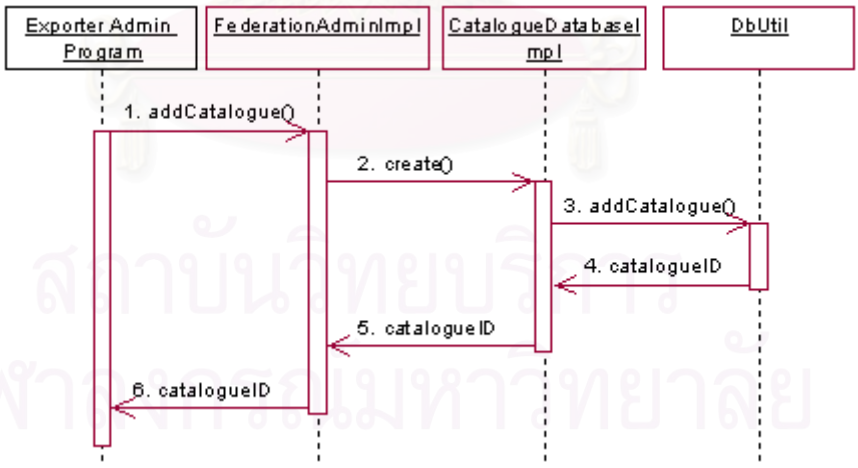


รูปที่ 4.5 แผนภาพแสดงการทำงานของคลาส ImportEventQueue

4. คลาส StructuredImportCallback

เป็นวัตถุตัวแทนประเภทโครงสร้างของผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการที่ส่งเป็นพารามิเตอร์ไปให้คลาส _tie_StructuredPushSubscriber (ในแพคเกจ scns.scncomm) เพื่อรอรับแจ้งการเปลี่ยนแปลงจากตัวแจ้งเหตุการณ์

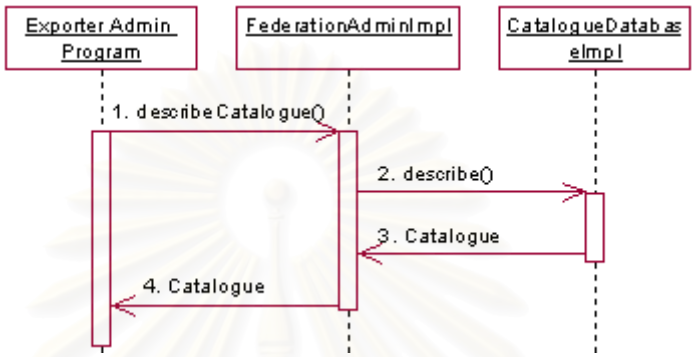
ลำดับเหตุการณ์ต่างๆ ของตัวจัดการสัญญาและตัวเรียกกลับนำเข้าสามารถแสดงได้ดังนี้



รูปที่ 4.6 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ addCatalogue()

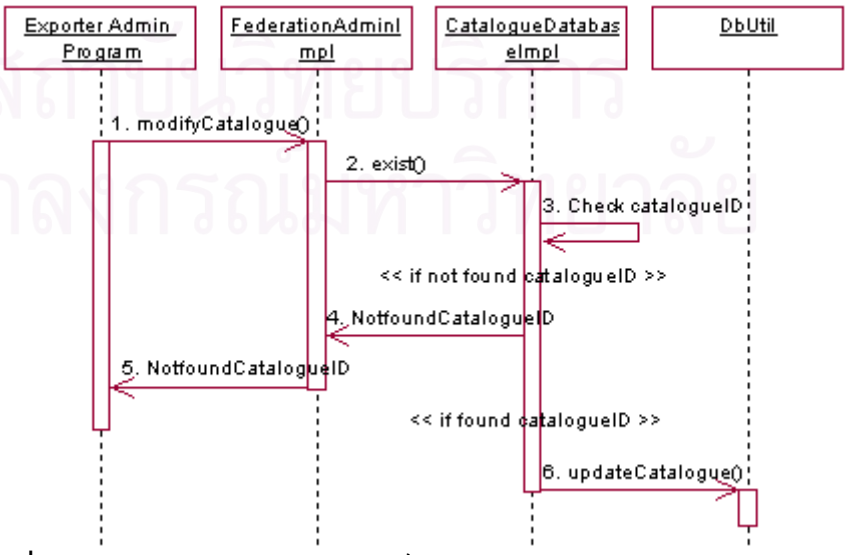
รูปที่ 4.6 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ addCatalogue() เพื่อทำการสร้างบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ addCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ addCatalogue()

จะทำการเรียกใช้ตัวกระทำการ create() (2) ของวัตถุ CatalogueDatabaseImpl ซึ่งเป็นตัวจัดการเกี่ยวกับบัญชีรายชื่อบริการ และตัวกระทำการ create() จะทำการเพิ่มบัญชีรายชื่อบริการเข้าไปในตารางแฮช (Hashtable) และจะทำการเรียกใช้ตัวกระทำการ addCatalogue() (3) ของวัตถุ DbUtil เพื่อทำการเก็บบัญชีรายชื่อบริการลงในฐานข้อมูล จากนั้นตัวกระทำการจะทำการส่งรหัสบัญชีรายชื่อบริการ catalogueID คืนกลับให้แก่ผู้ดูแลเทรดเดอร์ส่งออกเพื่อใช้งานต่อไป (4, 5, 6)



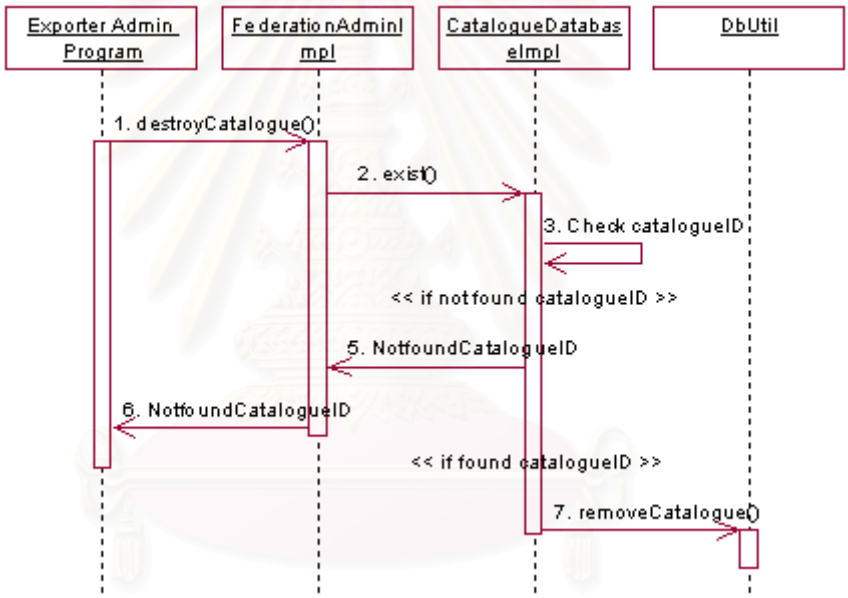
รูปที่ 4.7 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ describeCatalogue()

รูปที่ 4.7 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ describeCatalogue() เพื่อทำการแสดงบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ describeCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ describeCatalogue() จะทำการเรียกใช้ตัวกระทำการ describe() (2) ของวัตถุ CatalogueDatabaseImpl เพื่อทำการเรียกดูบัญชีรายชื่อบริการจากตารางแฮช (Hashtable) จากนั้นตัวกระทำการจะทำการส่งบัญชีรายชื่อบริการ Catalogue คืนกลับให้แก่ผู้ดูแลเทรดเดอร์ส่งออกเพื่อใช้งานต่อไป (3, 4)



รูปที่ 4.8 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ modifyCatalogue()

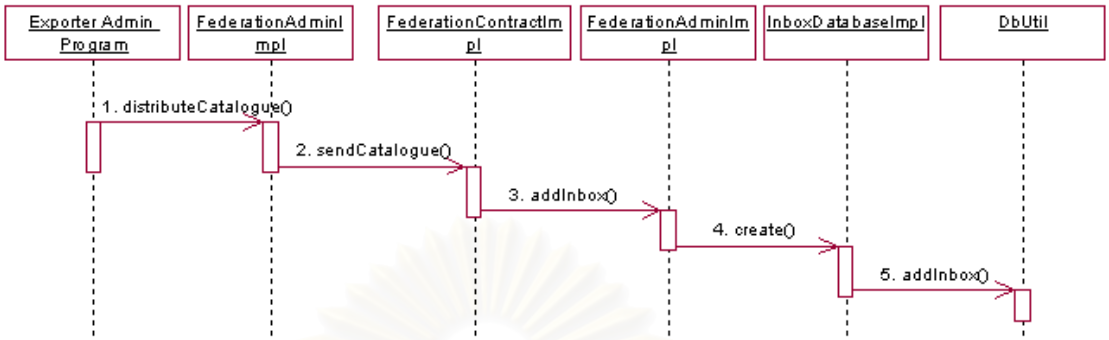
รูปที่ 4.8 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ modifyCatalogue() เพื่อทำการแก้ไขบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ modifyCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ modifyCatalogue() จะทำการเรียกใช้ตัวกระทำการ exist() (2) ของวัตถุ CatalogueDatabaseImpl จากนั้นตัวกระทำการ exist() จะทำการตรวจสอบว่ามีรหัสบัญชีรายชื่อบริการอยู่ในตารางแฮชหรือไม่ (3) หากไม่พบรหัสบัญชีรายชื่อบริการในตารางแฮช ตัวกระทำการจะส่งเอ็กซ์เซ็พชัน NotFoundCatalogueID คืนให้กับผู้ดูแลเทรดเดอร์ส่งออก (4, 5) หากพบรหัสบัญชีรายชื่อบริการ ตัวกระทำการก็จะทำการแก้ไขบัญชีรายชื่อบริการในตารางแฮช หลังจากนั้นตัวกระทำการ exist() ก็จะเรียกใช้ตัวกระทำการ updateCatalogue() (6) ของวัตถุ DbUtil เพื่อทำการแก้ไขบัญชีรายชื่อบริการในฐานข้อมูลให้ถูกต้อง



รูปที่ 4.9 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ destroyCatalogue()

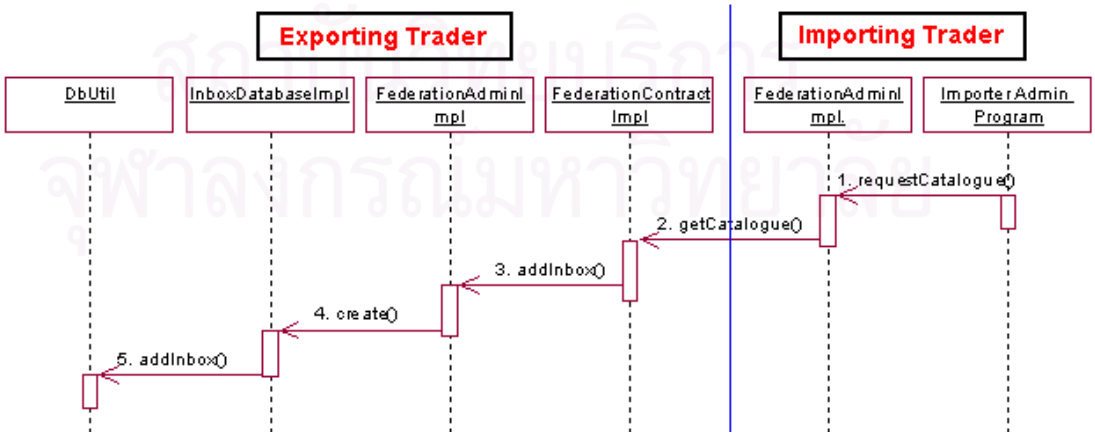
รูปที่ 4.9 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ destroyCatalogue() เพื่อทำการลบบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ destroyCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ destroyCatalogue() จะทำการเรียกใช้ตัวกระทำการ exist() (2) ของวัตถุ CatalogueDatabaseImpl จากนั้นตัวกระทำการ exist() จะทำการตรวจสอบว่ามีรหัสบัญชีรายชื่อบริการอยู่ในตารางแฮชหรือไม่ (3) หากไม่พบรหัสบัญชีรายชื่อบริการในตารางแฮช ตัวกระทำการจะส่งเอ็กซ์เซ็พชัน NotFoundCatalogueID คืนให้กับผู้ดูแลเทรดเดอร์ส่งออก (4, 5) หากพบรหัสบัญชีรายชื่อบริการ ตัวกระทำการก็จะทำการลบบัญชีรายชื่อบริการในตารางแฮช หลังจากนั้นตัวกระทำ

การ exist() ก็จะเรียกใช้ตัวกระทำการ removeCatalogue() (7) ของวัตถุ DbUtil เพื่อทำการลบบัญชีรายชื่อบริการในฐานข้อมูลให้ถูกต้อง



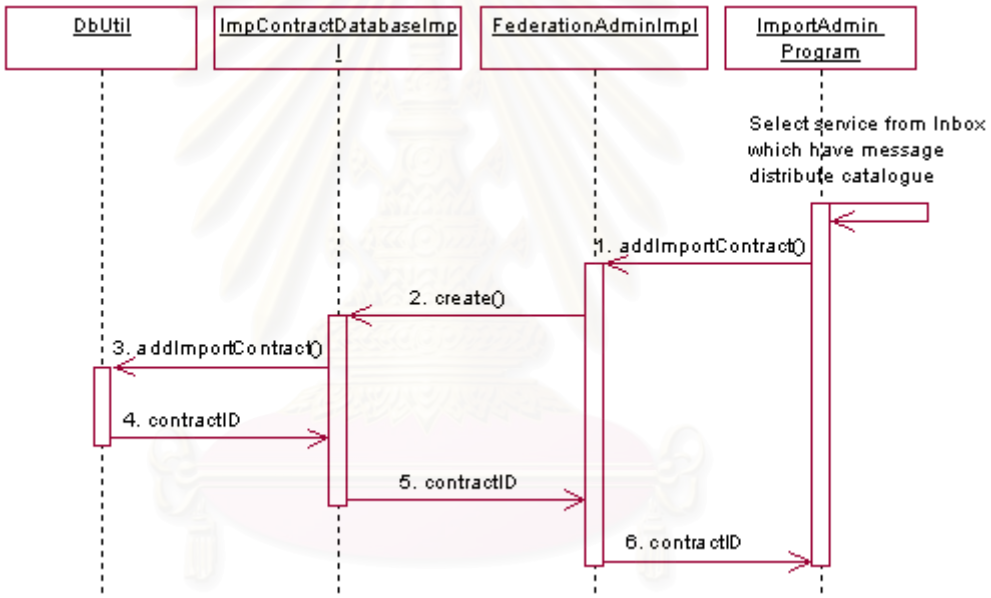
รูปที่ 4.10 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ distributeCatalogue()

รูปที่ 4.10 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ distributeCatalogue() เพื่อทำการส่งบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ distributeCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ distributeCatalogue() จะทำการเรียกใช้ตัวกระทำการ sendCatalogue() (2) ของวัตถุ FederationContractImpl เพื่อทำการส่งบัญชีรายชื่อบริการ หลังจากนั้นตัวกระทำการ sendCatalogue() ก็จะทำการส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยการเรียกใช้ตัวกระทำการ addInbox() (3) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ addInbox() ก็จะทำการเรียกใช้ตัวกระทำการ create() (4) ของวัตถุ InboxDatabaseImpl แล้วทำการเพิ่มข้อความแจ้งเข้าไปในตารางแฮช (Hashtable) จากนั้นตัวกระทำการ create() ก็จะทำการเรียกใช้ตัวกระทำการ addInbox() (5) ของวัตถุ DbUtil เพื่อทำการเพิ่มข้อความแจ้งลงในฐานข้อมูลต่อไป



รูปที่ 4.11 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ requestCatalogue()

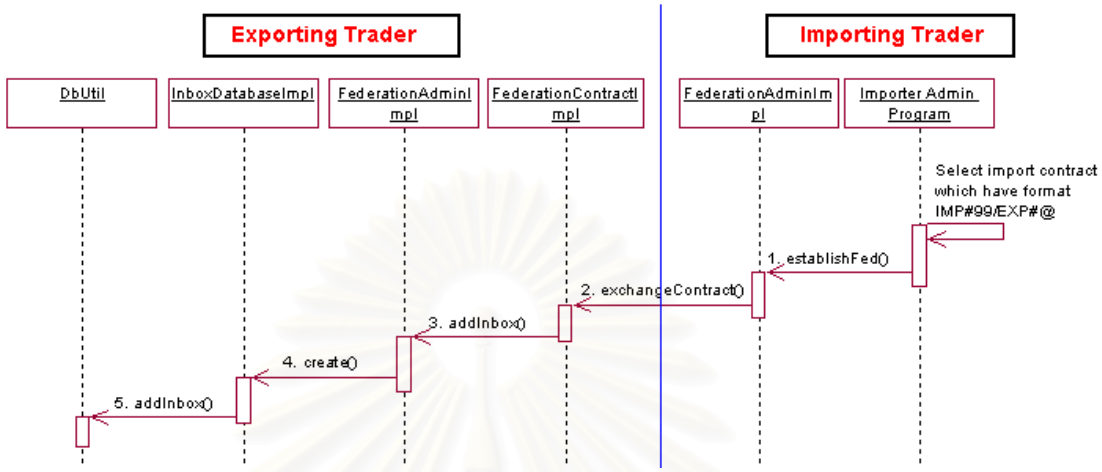
รูปที่ 4.11 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ requestCatalogue() เพื่อทำการร้องขอบัญชีรายชื่อบริการ เริ่มต้นจากผู้ดูแลเทรดเดอร์นำเข้าทำการเรียกใช้งานตัวกระทำการ requestCatalogue() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ requestCatalogue() จะทำการเรียกใช้ตัวกระทำการ getCatalogue() (2) ของวัตถุ FederationContractImpl เพื่อทำการร้องขอบัญชีรายชื่อบริการ หลังจากนั้นตัวกระทำการ getCatalogue() ก็จะทำการส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยการเรียกใช้ตัวกระทำการ addInbox() (3) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ addInbox() ก็จะทำการเรียกใช้ตัวกระทำการ create() (4) ของวัตถุ InboxDatabaseImpl แล้วทำการเพิ่มข้อความแจ้งเข้าไปในตารางแฮช (Hashtable) จากนั้นตัวกระทำการ create() ก็จะทำการเรียกใช้ตัวกระทำการ addInbox() (5) ของวัตถุ DbUtil เพื่อทำการเพิ่มข้อความแจ้งลงในฐานข้อมูลต่อไป



รูปที่ 4.12 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ addImportContract()

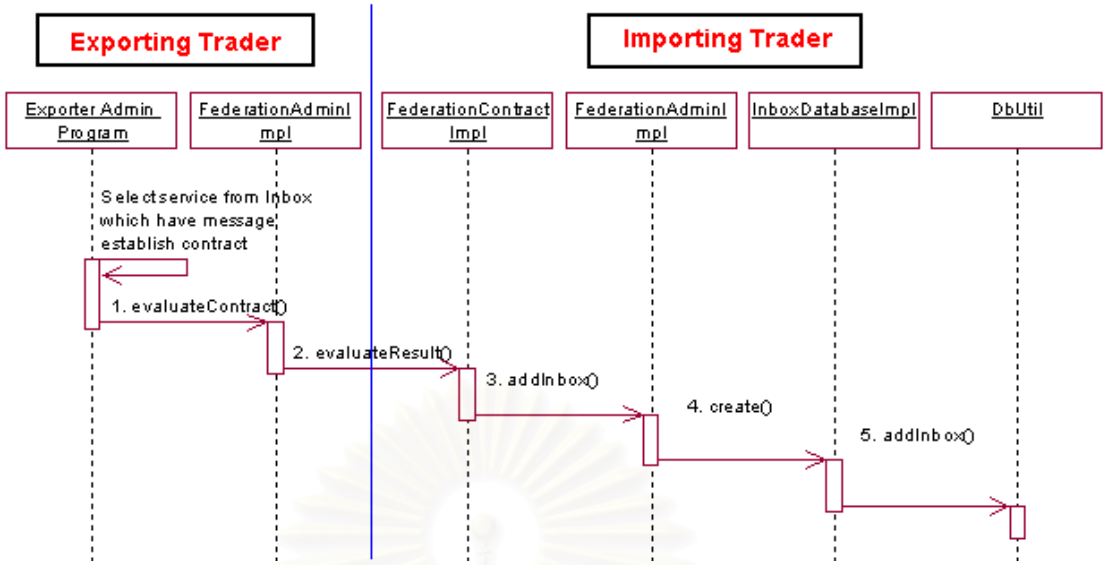
รูปที่ 4.12 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ addImportContract() เพื่อทำการสร้างสัญญานำเข้า เริ่มต้นจากผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกข้อความในกล่องข้อความขาเข้าที่มีข้อความแจ้งว่า “Distribute Catalogue” แล้วทำการเรียกใช้ตัวกระทำการ addImportContract() (1) ของวัตถุ FederationAdminI จากนั้นตัวกระทำการ addImportContract() จะทำการเรียกใช้ตัวกระทำการ create() (2) ของวัตถุ ImpContractDatabaseImpl ซึ่งเป็นตัวจัดการเกี่ยวกับสัญญานำเข้า และตัวกระทำการ create() จะทำการเพิ่มสัญญานำเข้าเข้าไปในตารางแฮช (Hashtable) และจะทำการเรียกใช้ตัวกระทำการ addImportContract() (3) ของวัตถุ DbUtil เพื่อทำการเก็บสัญญานำเข้าลงในฐานข้อมูล ซึ่งสัญญา

นำเข้าดังกล่าวจะมีรหัสสัญญาส่งออกเริ่มต้นเป็น EXP#@ เนื่องจากยังไม่ได้รับการอนุมัติจากผู้ดูแลเทรดเดอร์ส่งออก จากนั้นตัวกระทำการจะทำการส่งรหัสสัญญาข้อตกลง contractID คืนกลับให้แก่ผู้ดูแลเทรดเดอร์นำเข้าเพื่อใช้งานต่อไป (4, 5, 6)



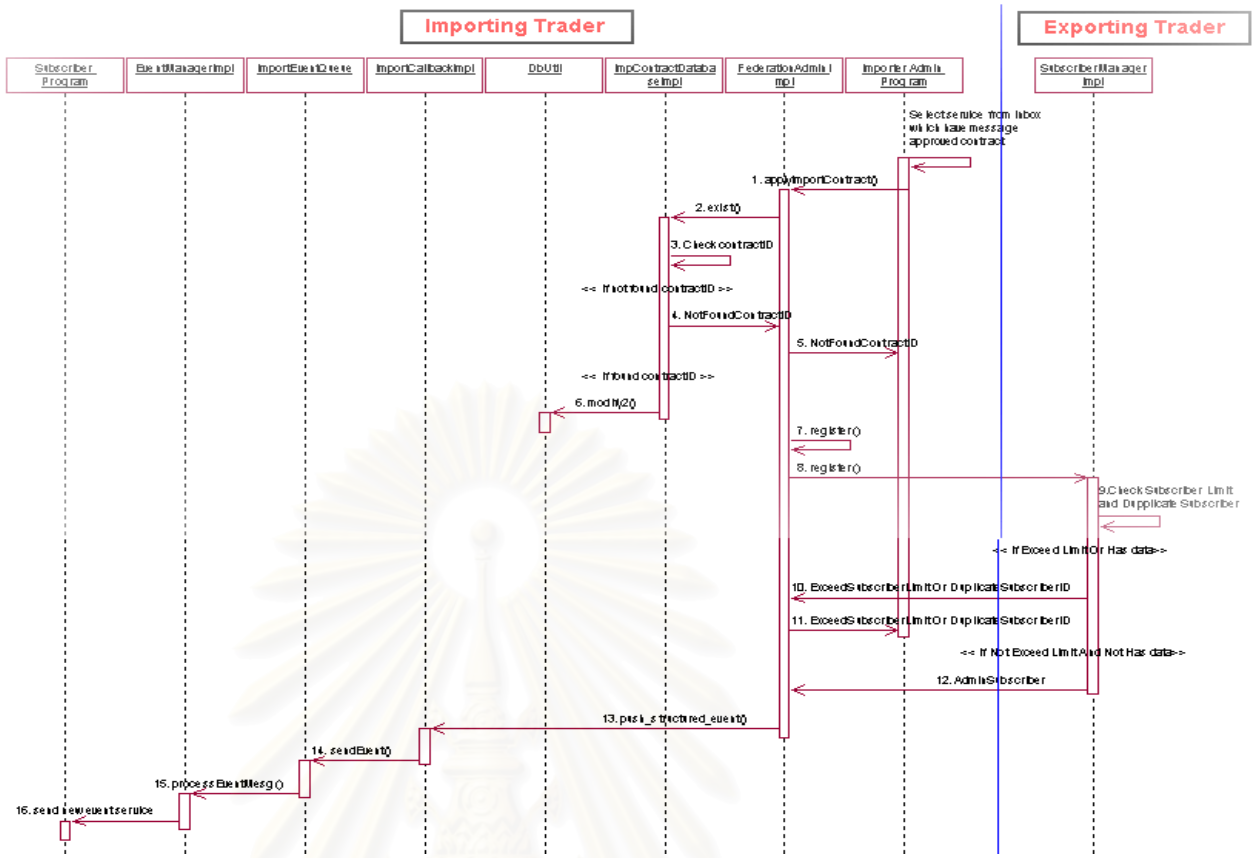
รูปที่ 4.13 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ establishFed()

รูปที่ 4.13 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ establishFed() เพื่อทำการแลกเปลี่ยนสัญญา เริ่มต้นจากผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกสัญญาข้อตกลงที่มีข้อมูลรหัสสัญญานำเข้ารูปแบบเป็น IMP#99/EXP#@ จากนั้นทำการเรียกใช้งานตัวกระทำการ establishFed() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ establishFed() จะทำการเรียกใช้ตัวกระทำการ exchangeContract() (2) ของวัตถุ FederationContractImpl เพื่อทำการส่งสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์ส่งออกทำการอนุมัติ หลังจากนั้นตัวกระทำการ exchangeContract() ก็จะทำการส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยการเรียกใช้ตัวกระทำการ addInbox() (3) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ addInbox() ก็จะทำการเรียกใช้ตัวกระทำการ create() (4) ของวัตถุ InboxDatabaseImpl แล้วทำการเพิ่มข้อความแจ้งเข้าไปในตารางแฮช (Hashtable) จากนั้นตัวกระทำการ create() ก็จะทำการเรียกใช้ตัวกระทำการ addInbox() (5) ของวัตถุ DbUtil เพื่อทำการเพิ่มข้อความแจ้งลงในฐานข้อมูลต่อไป



รูปที่ 4.14 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ evaluateContract()

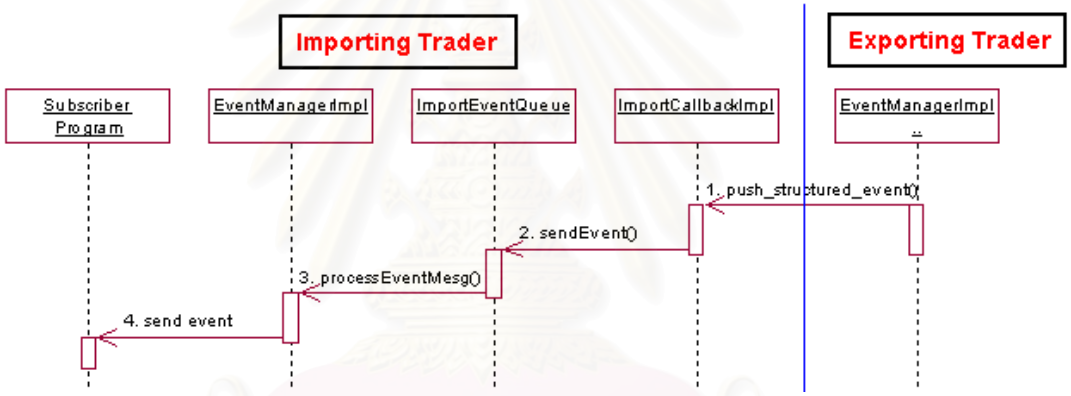
รูปที่ 4.14 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ evaluateContract() เพื่อทำการอนุมัติสัญญาข้อตกลง เริ่มต้นจากผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกข้อความในกล่องข้อความขาเข้าที่มีข้อความแจ้งว่า “Establish Contract” แล้วทำการเรียกใช้งานตัวกระทำการ evaluateContract() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ evaluateContract() จะทำการเรียกใช้ตัวกระทำการ evaluateResult() (2) ของวัตถุ FederationContractImpl เพื่อแจ้งผลการอนุมัติสัญญาไปให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ หลังจากนั้นตัวกระทำการ evaluateResult() ก็จะทำการส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยการเรียกใช้ตัวกระทำการ addInbox() (3) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ addInbox() ก็จะทำการเรียกใช้ตัวกระทำการ create() (4) ของวัตถุ InboxDatabaseImpl แล้วทำการเพิ่มข้อความแจ้งเข้าไปในตารางแฮช (Hashtable) จากนั้นตัวกระทำการ create() ก็จะทำการเรียกใช้ตัวกระทำการ addInbox() (5) ของวัตถุ DbUtil เพื่อทำการเพิ่มข้อความแจ้งลงในฐานข้อมูลต่อไป



รูปที่ 4.15 แผนภาพลำดับเหตุการณ์ของตัวกระทำการ applyImportContract()

รูปที่ 4.15 แสดงลำดับเหตุการณ์การเรียกใช้ตัวกระทำการ applyImportContract() เพื่อทำการแก้ไขรหัสสัญญาส่งออกในสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก เริ่มต้นจากผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกข้อความในกล่องข้อความขาเข้าที่มีข้อความแจ้งว่า “Approve Contract” แล้วทำการเรียกใช้งานตัวกระทำการ applyImportContract() (1) ของวัตถุ FederationAdminImpl จากนั้นตัวกระทำการ applyImportContract() จะทำการเรียกใช้ตัวกระทำการ exist() (2) ซึ่งตัวกระทำการ exist() จะทำการตรวจสอบว่ามีข้อมูลรหัสสัญญาส่งออกที่มีรูปแบบ EXP#@ และมีรหัสสัญญานำเข้าที่เหมือนกันอยู่ในตารางแฮชหรือไม่ (3) หากไม่พบรหัสสัญญาส่งออกและรหัสสัญญานำเข้าที่ตรงกัน ตัวกระทำการจะส่งเอ็กซ์เซ็พชัน NotFoundContractID คืนให้กับผู้ดูแลเทรดเดอร์นำเข้า (4, 5) หากพบรหัสสัญญาส่งออกและรหัสสัญญานำเข้าที่ตรงกัน ตัวกระทำการก็จะทำการแก้ไขข้อมูลรหัสสัญญาส่งออกในตารางแฮชของสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก หลังจากนั้นตัวกระทำการ exist() ก็จะเรียกใช้ตัวกระทำการ modify2() (6) ของวัตถุ DbUtil เพื่อทำการแก้ไขสัญญานำเข้าในฐานข้อมูลให้ถูกต้อง หลังจากนั้นตัวกระทำการ register() (7) ของวัตถุ FederationAdminImpl ก็จะทำการเรียกใช้ตัวกระทำการ register() (8) ของวัตถุ SubscriberManagerImpl เพื่อทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการ โดยจะทำการตรวจสอบว่าจำนวนผู้ขอรับทราบการเปลี่ยนแปลงเกินจำนวนที่กำหนดหรือมีการลงทะเบียนซ้ำ

หรือไม่ (9) หากพบมีจำนวนเกินกำหนดหรือมีการลงทะเบียนซ้ำ ก็จะส่งเอ็กซ์เซ็พชัน ExceedSubscriberLimit หรือ DuplicateSubscriberID ให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ (10, 11) และหากพบมีจำนวนไม่เกินกำหนดและไม่เคยลงทะเบียนมาก่อน ก็จะส่งวัตถุผู้ดูแลตัวแทนผู้ต้องการรับทราบการเปลี่ยนแปลง (AdminSubscriber) (12) คืนไปให้วัตถุ FederationAdminImpl ซึ่งก็จะทำการเรียกใช้ตัวกระทำการ push_structured_event() (13) ของวัตถุ ImportCallbackImpl เพื่อทำการแจ้งการเกิดบริการใหม่ โดยตัวกระทำการ push_structured_event() จะทำการเรียกใช้ตัวกระทำการ sendEvent() (14) ของวัตถุ ImportEventQueue เพื่อเข้าคิวรอส่งเหตุการณ์ให้กับตัวแจ้งเหตุการณ์ (EventManager) จากนั้นตัวกระทำการ sendEvent() จะทำการเรียกใช้ตัวกระทำการ processEventMesg() (15) ของวัตถุ EventManagerImpl เพื่อส่งข้อมูลการเปลี่ยนแปลงบริการและตัวกระทำการ processEventMesg() ก็จะทำการแจ้งการเกิดบริการใหม่ให้กับผู้รับบริการที่ได้ทำการลงทะเบียนขอรับทราบเหตุการณ์ไว้แล้ว



รูปที่ 4.16 แผนภาพลำดับเหตุการณ์ของการแจ้งการเปลี่ยนแปลงระยะไกลภายหลังจากที่ตัวเรียกกลับนำเข้าได้ทำการลงทะเบียนไว้

รูปที่ 4.16 แสดงลำดับเหตุการณ์ของการแจ้งการเปลี่ยนแปลงระยะไกล เริ่มจากตัวแจ้งเหตุการณ์ (EventManagerImpl) ของบริการแจ้งเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์ส่งออกทำการเรียกใช้ตัวกระทำการ push_structured_event() (1) ของวัตถุ ImportCallbackImpl จากนั้นตัวกระทำการ push_structured_event() จะเรียกใช้ตัวกระทำการ sendEvent() (2) ของวัตถุ ImportEventQueue เพื่อรอคิวส่งเหตุการณ์ไปให้กับตัวแจ้งเหตุการณ์ของบริการแจ้งเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์นำเข้า หลังจากนั้นตัวกระทำการ sendEvent() จะเรียกใช้ตัวกระทำการ processEventMesg() (3) ของวัตถุ EventManagerImpl ของบริการแจ้งเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์นำเข้า ซึ่งตัวกระทำการจะส่งข้อมูลแจ้งการเปลี่ยนแปลงไปให้ผู้รับบริการที่ทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงต่อไป

บทที่ 5

การทดสอบการใช้งานต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ในบทนี้จะกล่าวถึงรายละเอียดการทดสอบการทำงานของต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลที่ได้รับการพัฒนา เพื่อสนับสนุนการทำงานของบริการเอสซีเอ็น [6] ให้สามารถแจ้งการเปลี่ยนแปลงข้ามระบบได้ ซึ่งการทดสอบประกอบด้วยการทดสอบการแลกเปลี่ยนสัญญาระหว่างเทรดเดอร์ส่งออกและเทรดเดอร์นำเข้า การทดสอบการรับข้อมูลแจ้งการเปลี่ยนแปลงของตัวเรียกกลับนำเข้าจากตัวแจ้งเหตุการณ์ที่อยู่ต่างระบบ โดยรายละเอียดการทดสอบมีดังนี้

5.1 สภาวะที่ใช้ในการทดสอบ

1. เครื่อง Intel Pentium IV 2.4 กิกะเฮิร์ตซ์ หน่วยความจำขนาด 512 เมกะไบต์ ระบบปฏิบัติการวินโดวส์ รุ่น 2000 สำหรับรันบริการเอสซีเอ็น บริการเทรดเดอร์ และบริการแจ้งเหตุการณ์ 1 ตัว
2. เครื่อง Intel Pentium IV 2.0 กิกะเฮิร์ตซ์ หน่วยความจำขนาด 256 เมกะไบต์ ระบบปฏิบัติการวินโดวส์ รุ่น 2003 สำหรับรันบริการเอสซีเอ็น บริการเทรดเดอร์ และบริการแจ้งเหตุการณ์ 1 ตัว
3. เครื่อง Intel Pentium III 1.0 กิกะเฮิร์ตซ์ หน่วยความจำขนาด 256 เมกะไบต์ ระบบปฏิบัติการวินโดวส์ รุ่น 2000 สำหรับการจำลองผู้แจ้งการเปลี่ยนแปลงบริการ
4. เครื่อง Intel Pentium IV 1.6 กิกะเฮิร์ตซ์ หน่วยความจำขนาด 128 เมกะไบต์ ระบบปฏิบัติการวินโดวส์ รุ่น 2000 สำหรับการจำลองผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ
5. เครื่องมือพัฒนาภาษาจาวา เจดีเค 1.4.2
6. วิสิโบรกเกอร์รุ่น 3.4 สำหรับภาษาจาวา
7. บริการเทรดเดอร์ของจาคอร์บ รุ่น 1.3.30 จำนวน 2 ตัว
8. บริการแจ้งเหตุการณ์ของดีคอน รุ่น 3.0 จำนวน 2 ตัว
9. บริการเอสซีเอ็นจำนวน 2 ตัว
10. ระบบฐานข้อมูลมายเอสคิวแอล รุ่น 4.0.18 จำนวน 2 ตัว

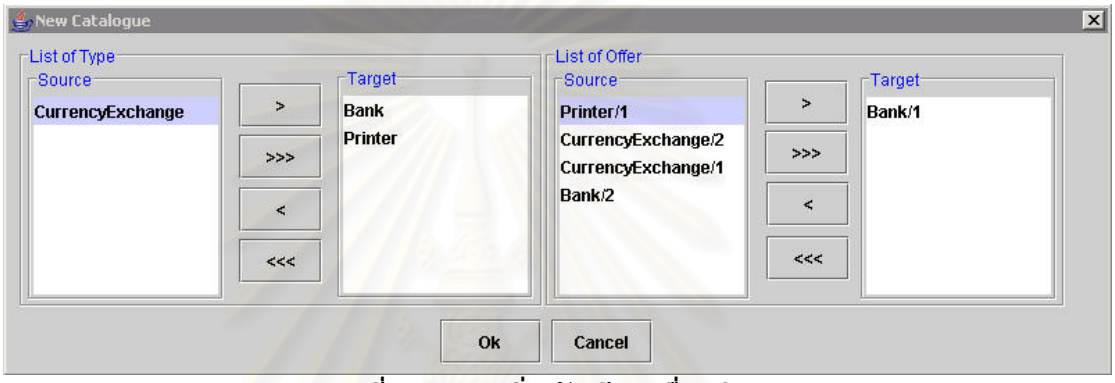
5.2 การทดสอบการทำงานของต้นแบบระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกล

ในส่วนนี้เป็นการทดสอบการทำงานของส่วนต่อประสานต่างๆ ที่ได้เพิ่มขยายให้กับบริการเทรดเดอร์ โดยการทดสอบจะอาศัยโปรแกรมที่พัฒนาขึ้นสำหรับการทดสอบโดยเฉพาะ โดยภายใน

โปรแกรมจะมีการติดต่อกับบริการเทรดเดอร์เพื่อเรียกใช้งานส่วนต่อประสานต่างๆ การทดสอบแบ่งออกเป็นส่วนๆ ดังนี้

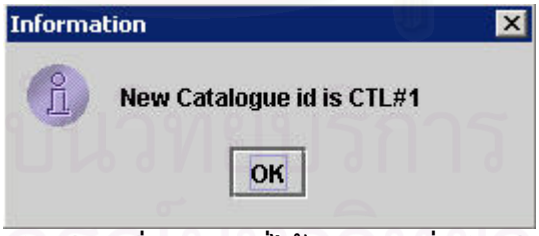
5.2.1 การทดสอบการสร้างบัญชีรายชื่อบริการ

การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำนวน 1 ตัว โดยจำลองเป็นผู้ดูแลเทรดเดอร์ส่งออก ทำการสร้างบัญชีรายชื่อบริการจากชนิดบริการและข้อเสนอบริการที่มีอยู่ในระบบ เพื่อทดสอบการทำงานของตัวกระทำการ addCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก และเก็บไว้ในคลังบัญชีรายชื่อบริการ ซึ่งอธิบายได้ดังรูปที่ 5.1-5.3



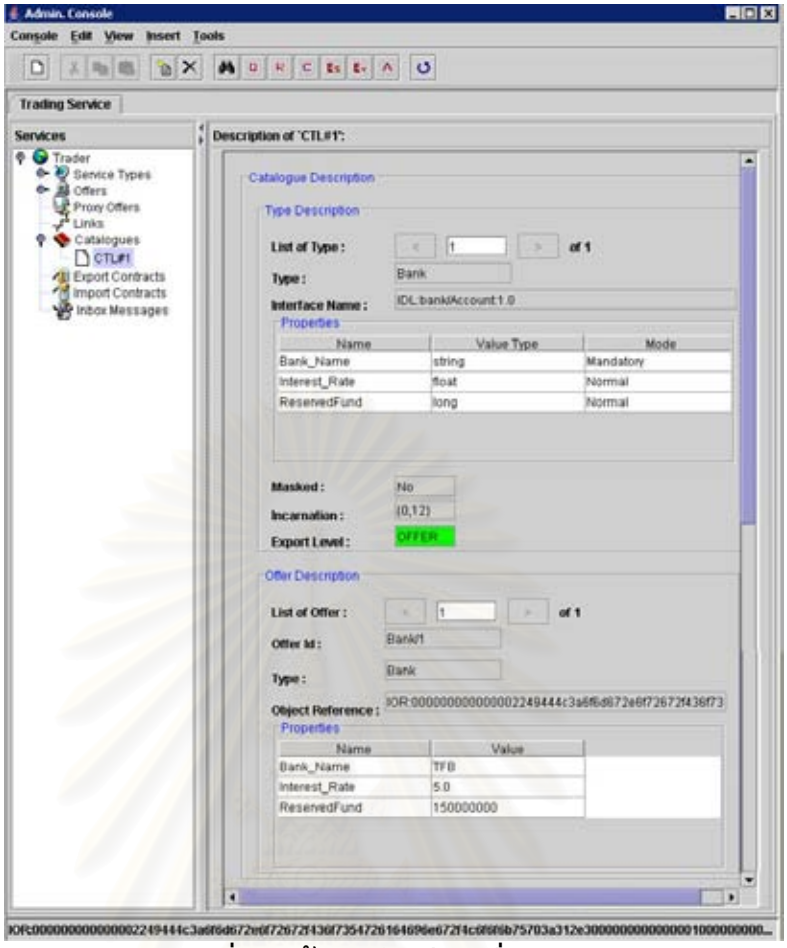
รูปที่ 5.1 การเพิ่มบัญชีรายชื่อบริการ

รูปที่ 5.1 ผู้ดูแลเทรดเดอร์ส่งออกทำการเลือกข้อมูลบริการที่มีชนิดบริการ คือ Bank กับ Printer และข้อเสนอบริการที่มีรหัสข้อเสนอบริการ คือ Bank/1 (Bank_Name="TFB") แล้วทำการสร้างบัญชีรายชื่อบริการ โดยกดปุ่ม Ok เพื่อเรียกใช้งานตัวกระทำการ addCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก



รูปที่ 5.2 รหัสบัญชีรายชื่อบริการที่ได้จากการเพิ่มบัญชีรายชื่อบริการ

รูปที่ 5.2 แสดงผลลัพธ์เมื่อระบบทำการเพิ่มบัญชีรายชื่อบริการเข้าไปเก็บในคลังบัญชีรายชื่อบริการเรียบร้อยแล้ว และคืนค่ารหัสบัญชีรายชื่อบริการ คือ CTL#1 กลับมาให้ผู้ดูแลเทรดเดอร์ส่งออก



รูปที่ 5.3 ข้อมูลบัญชีรายชื่อบริการ

รูปที่ 5.3 แสดงรายละเอียดข้อมูลบัญชีรายชื่อบริการ ภายหลังจากระบบทำการเพิ่มข้อมูลบัญชีรายชื่อบริการเรียบร้อยแล้ว

จากผลการทำงานในรูปที่ 5.1-5.3 สามารถอธิบายได้ว่าการทำงานของตัวกระทำการ addCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก สามารถเพิ่มบัญชีรายชื่อบริการและเก็บไว้ในคลังบัญชีรายชื่อบริการได้อย่างถูกต้อง โดยสังเกตจากรูปที่ 5.2 จะเห็นว่ามีข้อความ “New Catalogue id is CTL#1” และสังเกตจากรูปที่ 5.3 มีรหัสบัญชีรายชื่อบริการอยู่ใต้ไอคอน “Catalogues” ในฝั่งซ้าย และรายละเอียดข้อมูลบัญชีรายชื่อบริการในฝั่งขวา

5.2.2 การทดสอบการร้องขอบัญชีรายชื่อบริการ

การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำนวน 1 ตัว โดยจำลองเป็นผู้ดูแลเทรดเดอร์นำเข้า ทำการร้องขอบัญชีรายชื่อบริการจากผู้ดูแลเทรดเดอร์ส่งออก เพื่อทดสอบการทำงานของตัวกระทำการ requestCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า และทดสอบการทำงานของตัวกระทำการ getCatalogue() ของส่วนต่อประสาน FederationContract() ในฝั่งเทรดเดอร์ส่งออก จากนั้นตัวกระทำการ getCatalogue() จะทำการ

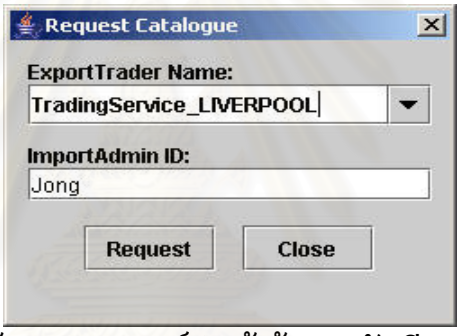
เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก เพื่อแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยมีลำดับขั้นดังต่อไปนี้

1. เรียกใช้งานตัวกระทำการ requestCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อร้องขอบัญชีรายชื่อบริการจากผู้ดูแลเทรดเดอร์ส่งออก ดังรูปที่ 5.4

2. ตัวกระทำการ requestCatalogue() ทำการเรียกใช้งานตัวกระทำการ getCatalogue() ของส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์ส่งออก เพื่อแจ้งข้อมูลการร้องขอบัญชีรายชื่อบริการให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ

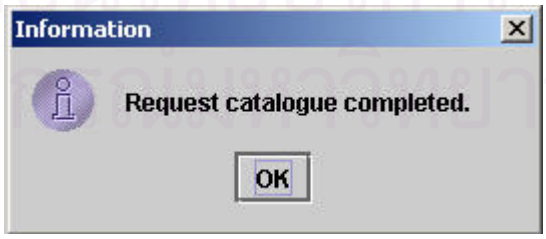
3. ตัวกระทำการ getCatalogue() เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก เพื่อทำการเก็บข้อมูลการร้องขอบัญชีรายชื่อบริการไว้ในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออก

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.4 - 5.6



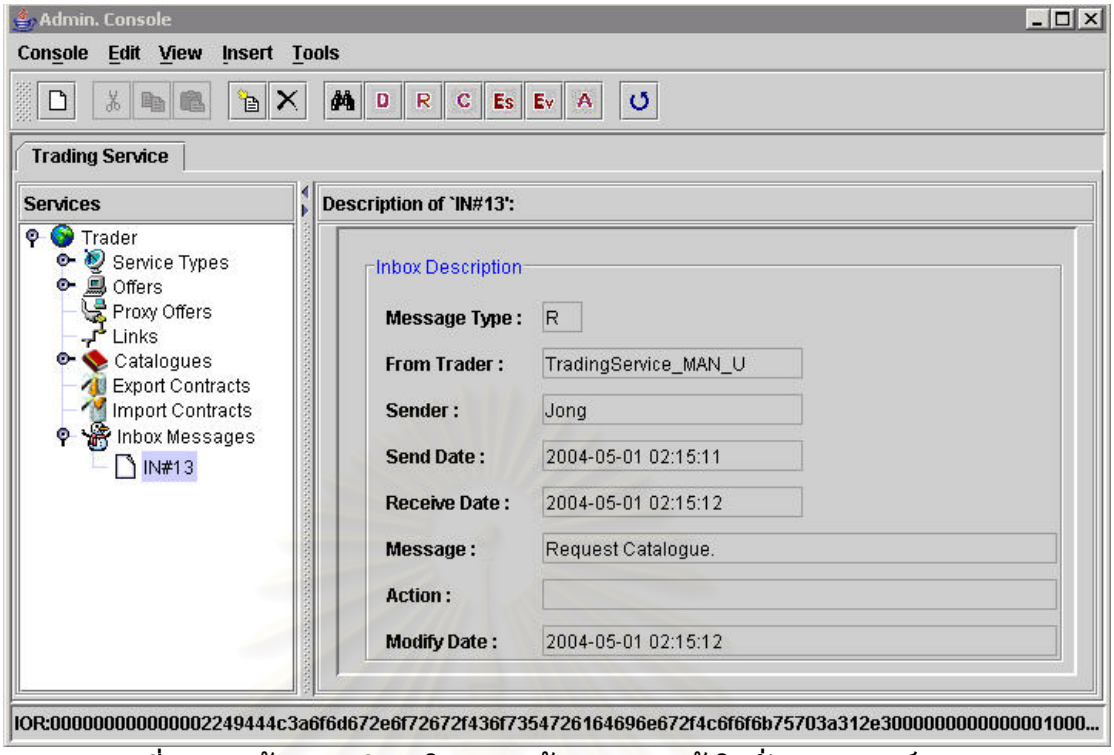
รูปที่ 5.4 ผู้ดูแลเทรดเดอร์นำเข้าร้องขอบัญชีรายชื่อบริการ

รูปที่ 5.4 ผู้ดูแลเทรดเดอร์นำเข้าทำการระบุชื่อเทรดเดอร์ส่งออก ชื่อของผู้ดูแลเทรดเดอร์นำเข้า เพื่อส่งข้อความร้องขอบัญชีรายชื่อบริการจากผู้ดูแลเทรดเดอร์ส่งออก โดยกดปุ่ม Request เพื่อเรียกใช้งานตัวกระทำการ requestCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า



รูปที่ 5.5 ผลลัพธ์จากการร้องขอบัญชีรายชื่อบริการ

รูปที่ 5.5 แสดงผลลัพธ์เมื่อข้อความร้องขอบริการได้เพิ่มเข้าไปในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออกเรียบร้อยแล้ว



รูปที่ 5.6 คำร้องขอบริการในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออก

รูปที่ 5.6 แสดงรายละเอียดข้อความร้องขอบัญชีรายชื่อบริการของผู้ดูเทรดเดอร์นำเข้าภายหลังจากเทรดเดอร์ส่งออกทำการเพิ่มข้อความร้องขอบัญชีรายชื่อบริการเรียบร้อยแล้ว

จากผลการทำงานในรูปที่ 5.4 - 5.6 สามารถอธิบายได้ว่าการทำงานของตัวกระทำการ requestCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า สามารถส่งข้อความร้องขอบัญชีรายชื่อบริการไปยังผู้ดูแลเทรดเดอร์ส่งออกได้อย่างถูกต้อง โดยสังเกตจากรูปที่ 5.5 จะเห็นว่ามีข้อความ “Request catalogue completed.” และสังเกตจากรูปที่ 5.6 มีรหัสข้อความนำเข้า IN#13 อยู่ใต้ไอคอน “Inbox Messages” ในฝั่งซ้าย และรายละเอียดข้อความคำร้องขอบริการในฝั่งขวา โดยมีข้อความว่า “Request Catalogue.”

5.2.3 การทดสอบการส่งออกบัญชีรายชื่อบริการ

การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำนวน 1 ตัว โดยจำลองเป็นผู้ดูแลเทรดเดอร์ส่งออก ทำการส่งบัญชีรายชื่อบริการไปให้ผู้ดูแลเทรดเดอร์นำเข้า โดยโปรแกรมจะทำการสร้างข้อมูลบัญชีรายชื่อบริการ โดยมีรายละเอียดของข้อมูลตามโครงสร้างข้างล่างดังนี้

```
struct Catalogue {
   CatalogueID  catalogueID;
   TypeDescSeq  typeList;    //list of service types
   OfferDescSeq offerList;   //list of service offers
};
```

ข้อมูลที่ 1 ข้อมูลบัญชีรายชื่อบริการ (จากหัวข้อ 5.2.1) มีรายละเอียดดังนี้

บัญชีรายชื่อบริการ

```
CatalogueID :CTL#1
```

ชนิดของบริการ

```
Service Type Name : Bank

Property Definition :
  Name          [string]
  ReservedFund  [long]
  Interest_Rate [float]

 interface Bank {
  float   balance (in string accno);
  boolean deposit (in string accno, in float amount);
  boolean withdraw(in string accno, in float amount);
};

Export Level : OFFER
```

ข้อเสนอบริการ

```
Name          = TFB
ReservedFund  = 150000000
Interest_Rate = 5.0
```

ข้อเสนอบริการ

```
Name          = SCB
ReservedFund  = 120000000
Interest_Rate = 4.75
```

ชนิดของบริการ

```
Service Type Name : Printer

Property Definition :
  Type  [string]
  Name  [string]
  Speed [long]

interface PrinterService {
  boolean print(in string url);
};

Export Level : TYPE
```

ข้อเสนอบริการ

ไม่มี เนื่องจาก Type Printer ส่งออกระดับชนิดบริการ

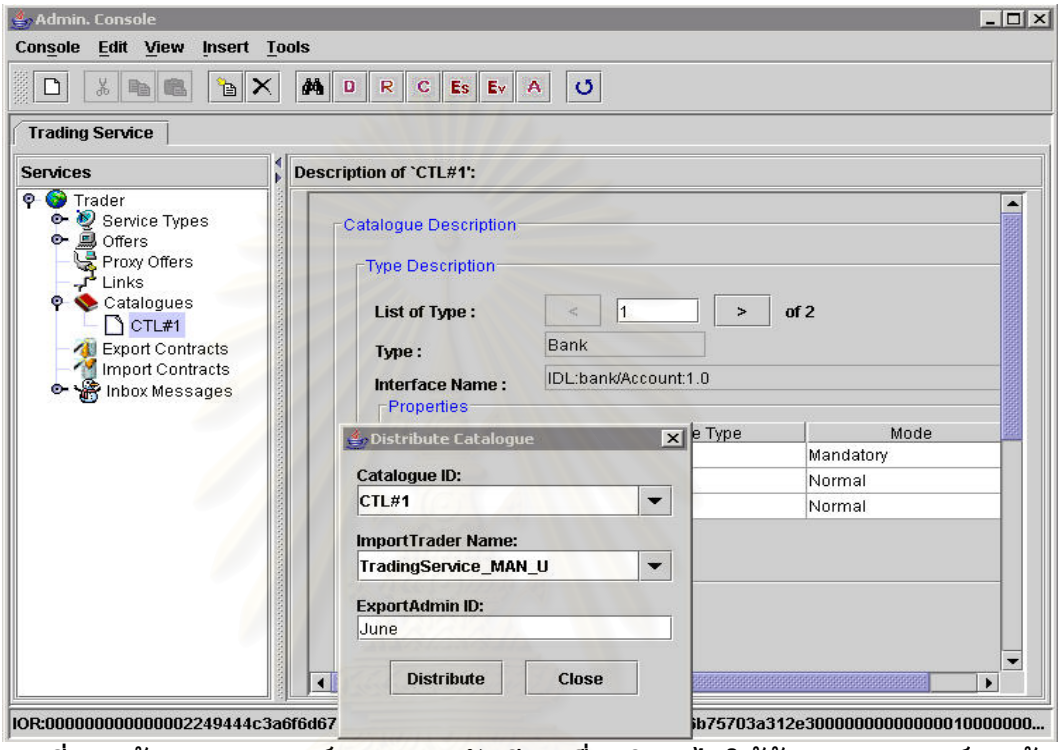
การทำงานของโปรแกรมที่จำลองเป็นผู้ดูแลเทรดเดอร์ส่งออกประกอบด้วย

1. ผู้ดูแลเทรดเดอร์ส่งออกทำการส่งข้อมูลบัญชีรายชื่อบริการ (ข้อมูลที่ 1) ซึ่งมีข้อมูลรหัสบัญชีรายชื่อบริการ คือ CTL#1 ไปให้กับผู้ดูแลเทรดเดอร์นำเข้า โดยเรียกใช้งานตัวกระทำการ distributeCatalogue() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก ดังรูปที่ 5.7

2. ตัวกระทำการ distributeCatalogue() เรียกใช้งานตัวกระทำการ sendCatalogue() ของส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์นำเข้า เพื่อแจ้งข้อมูลการส่งบัญชีรายชื่อบริการให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ ดังรูปที่ 5.8

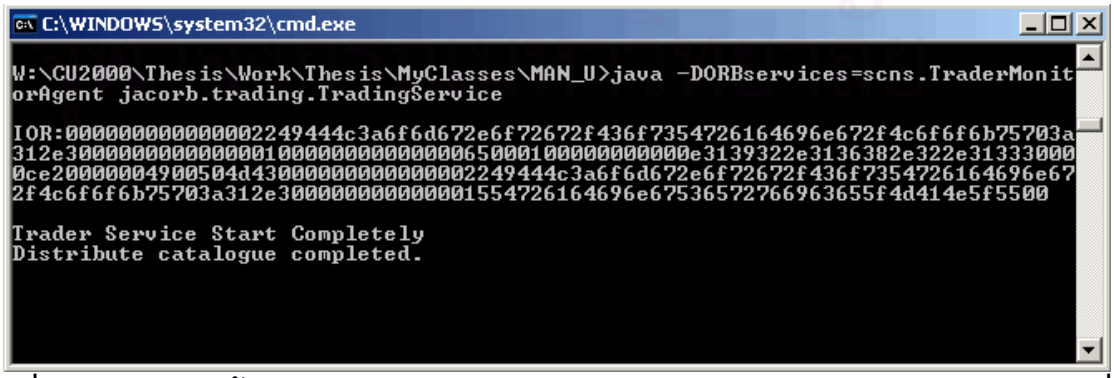
3. ตัวกระทำการ sendCatalogue() เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อทำการเก็บข้อมูลการส่งบัญชีรายชื่อบริการไว้ในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์นำเข้า ดังรูปที่ 5.10

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.7 - 5.10



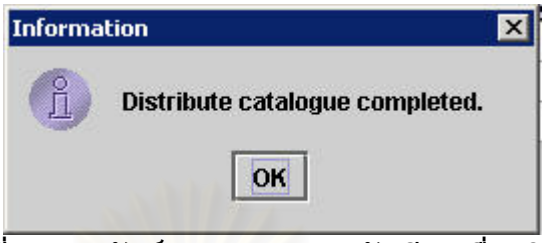
รูปที่ 5.7 ผู้ดูแลเทรดเดอร์ส่งออกส่งบัญชีรายชื่อบริการไปให้ผู้ดูแลเทรดเดอร์นำเข้า

รูปที่ 5.7 ผู้ดูแลเทรดเดอร์ส่งออกทำการระบุชื่อเทรดเดอร์นำเข้า ชื่อของผู้ดูแลเทรดเดอร์ส่งออก เพื่อส่งข้อความส่งออกบัญชีรายชื่อบริการไปยังเทรดเดอร์นำเข้า โดยกดปุ่ม Distribute เพื่อเรียกใช้งานตัวกระทำการ distributeCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก



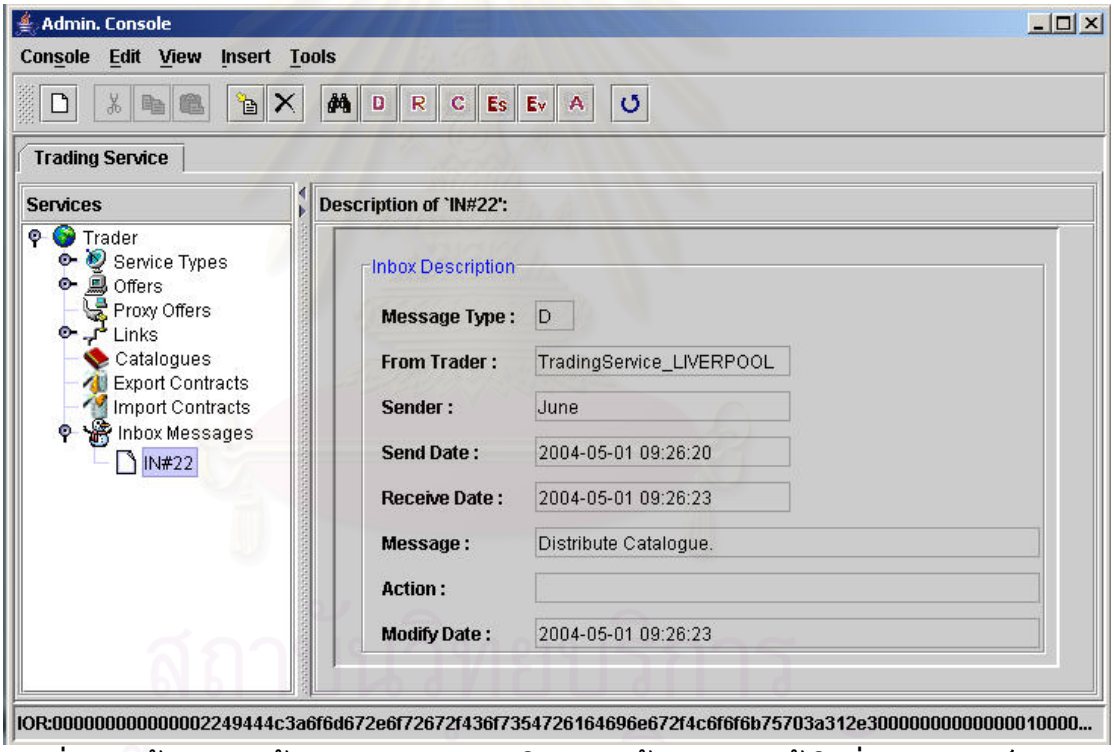
รูปที่ 5.8 การเรียกใช้งานตัวกระทำการ sendCatalogue() และตัวกระทำการ addInbox() ในฝั่งเทรดเดอร์นำเข้า

รูปที่ 5.8 แสดงการเรียกใช้งานตัวกระทำการ sendCatalogue() และตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์นำเข้า เพื่อทำการส่งข้อความแจ้งเรื่องส่งออกบัญชีรายชื่อบริการ



รูปที่ 5.9 ผลลัพธ์จากการส่งออกบัญชีรายชื่อบริการ

รูปที่ 5.9 แสดงผลลัพธ์เมื่อข้อความส่งออกบริการได้เพิ่มเข้าไปในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์นำเข้าเรียบร้อยแล้ว



รูปที่ 5.10 ข้อความแจ้งการส่งออกบริการในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออก

รูปที่ 5.10 แสดงรายละเอียดข้อความการส่งออกบัญชีรายชื่อบริการของผู้ดูเทรดเดอร์ส่งออกภายหลังจากเทรดเดอร์นำเข้าทำการเพิ่มข้อความส่งออกบัญชีรายชื่อบริการเรียบร้อยแล้ว

จากผลการทำงานในรูปที่ 5.7 – 5.10 สามารถอธิบายได้ว่าการทำงานของตัวกระทำการ distributeCatalogue() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก สามารถส่งข้อความแจ้งเรื่องการส่งบัญชีรายชื่อบริการให้ผู้ดูแลเทรดเดอร์นำเข้าทราบได้อย่างถูกต้อง โดย

สังเกตจากรูปที่ 5.8 - 5.9 จะเห็นว่ามีข้อความ “Distribute catalogue completed.” และสังเกตจากรูปที่ 5.10 มีรหัสข้อความนำเข้า IN#22 อยู่ใต้ไอคอน ‘Inbox Messages’ ในฝั่งซ้าย และรายละเอียดข้อความคำร้องขอบริการในฝั่งขวา โดยมีข้อความว่า “Distribute Catalogue.”

5.2.4 การทดสอบการสร้างสัญญานำเข้า

การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำลองเป็นผู้ดูแลเทรดเดอร์นำเข้าจำนวน 1 ตัว ทำการสร้างสัญญานำเข้า โดยโปรแกรมจะทำการเลือกข้อมูลชนิดบริการและข้อเสนอบริการจากบัญชีรายชื่อบริการ (ข้อมูลที่ 1 หัวข้อที่ 5.2.3) โดยมีรายละเอียดของข้อมูลดังต่อไปนี้

ข้อมูลที่ 2 ข้อมูลรายชื่อชนิดบริการและข้อเสนอบริการที่ร้องขอ มีรายละเอียดดังนี้

บัญชีรายชื่อบริการ จากข้อมูลที่ 1 หัวข้อ 5.2.3 CatalogueID :CTL#1
รายการชนิดของบริการที่ร้องขอ Service Type Name : Bank, Printer
รายการข้อเสนอบริการที่ร้องขอ Name = TFB

การทำงานของโปรแกรมที่จำลองเป็นผู้ดูแลเทรดเดอร์นำเข้าประกอบด้วย

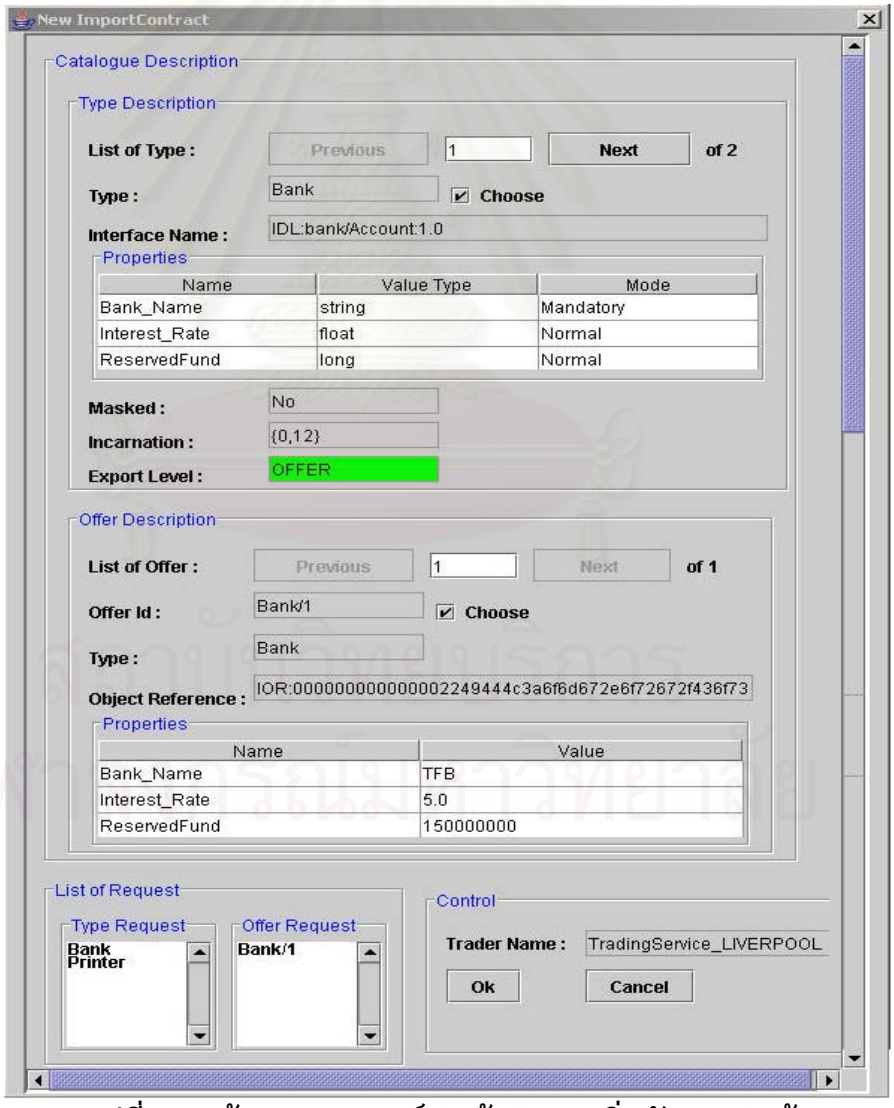
1. ผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกชนิดบริการและข้อเสนอบริการ (ข้อมูลที่ 2) โดยเรียกใช้งานตัวกระทำการ addImportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อสร้างสัญญานำเข้า โดยข้อมูลสัญญานำเข้าที่ถูกสร้างขึ้นจะมีรายละเอียดตามโครงสร้างข้างล่าง (ข้อมูลที่ 3) ดังนี้

```
struct Contract {
   ContractID   expContractID;  //reference id of exporting contract
   ContractID   impContractID;  //reference id of importing contract
   Catalogue    catalogue;
    TypeDescSeq  typeReq;       //list of service types request
    OfferDescSeq offerReq;      //list of service offers request
    TraderNM     expTrader;     //exporting trader name
};
```

ข้อมูลที่ 3 ข้อมูลสัญญานำเข้า มีรายละเอียดดังนี้

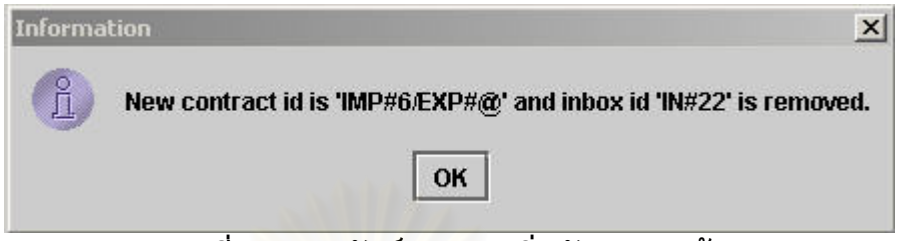
สัญญาข้อตกลง ExpContractID :EXP#@ (รหัสสัญญาส่งออก) ImpContractID :IMP#6 (รหัสสัญญานำเข้า)
บัญชีรายชื่อบริการ จากข้อมูลที่ 2 CatalogueID :CTL#1
รายการชนิดของบริการที่ร้องขอ Service Type Name :Bank, Printer
รายการข้อเสนอบริการที่ร้องขอ Name = TFB
เทรดเดอร์ที่ส่งออกบริการ Trader Name : TradingService_LIVERPOOL

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.11 – 5.13



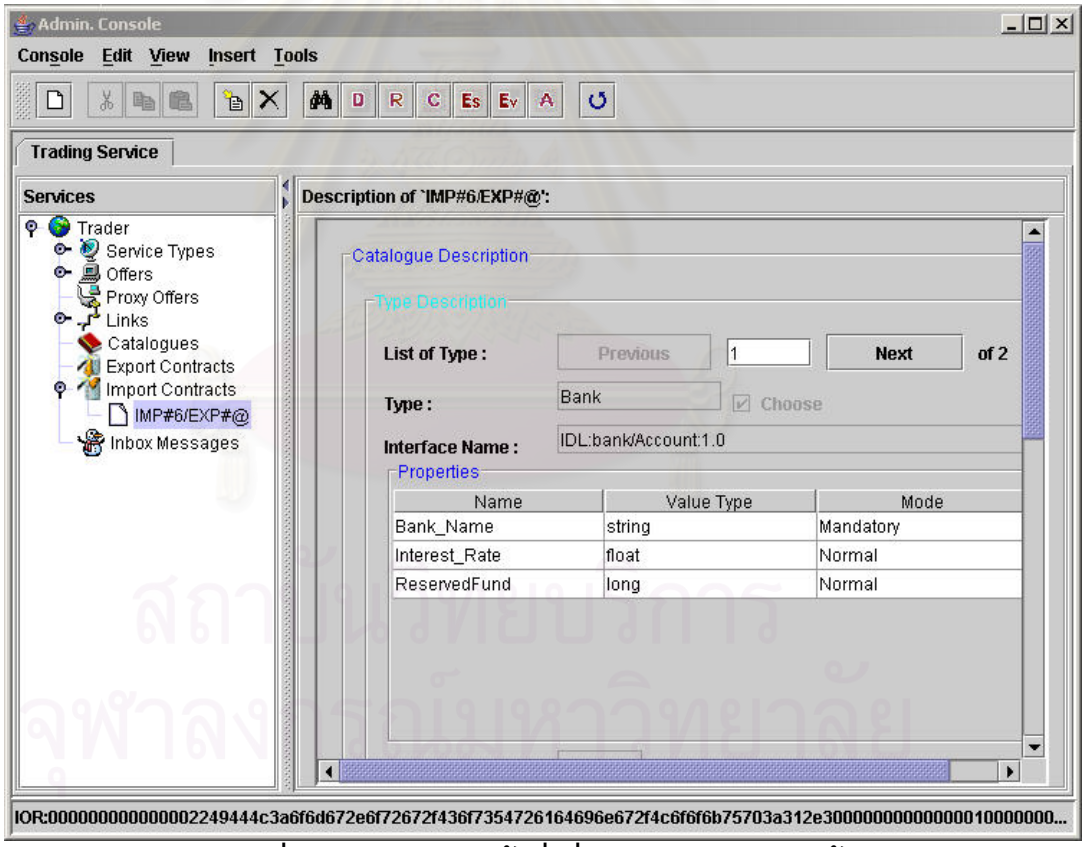
รูปที่ 5.11 ผู้ดูแลเทรดเดอร์นำเข้าทำการเพิ่มสัญญานำเข้า

รูปที่ 5.11 ผู้ดูแลเทรดเดอร์นำเข้าทำการเลือกชนิดบริการและข้อเสนอบริการจากบัญชีรายชื่อบริการ และสร้างสัญญานำเข้า โดยกดปุ่ม Ok เพื่อเรียกใช้งานตัวกระทำการ addImportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า



รูปที่ 5.12 ผลลัพธ์จากการเพิ่มสัญญานำเข้า

รูปที่ 5.12 แสดงผลลัพธ์เมื่อเทรดเดอร์นำเข้าทำการเพิ่มสัญญานำเข้าเก็บไว้ในคลังสัญญานำเข้า และคืนค่ารหัสสัญญานำเข้าคือ IMP#6 และรหัสสัญญาส่งออกคือ EXP#@ และทำการลบข้อความรหัส IN#22 ออกจากกล่องข้อความขาเข้าในฝั่งเทรดเดอร์นำเข้า



รูปที่ 5.13 สัญญานำเข้าที่เพิ่มในคลังสัญญานำเข้า

รูปที่ 5.13 แสดงรายละเอียดสัญญานำเข้าที่สร้างขึ้นโดยผู้ดูเทรดเดอร์นำเข้า ภายหลังจากเทรดเดอร์นำเข้าทำการเพิ่มสัญญานำเข้าไว้ในคลังสัญญานำเข้าเรียบร้อยแล้ว

จากผลการทำงานในรูปที่ 5.11 – 5.13 สามารถอธิบายได้ว่าการทำงานของตัวกระทำการ addImportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า สามารถสร้างสัญญาข้อตกลงเก็บไว้ในคลังสัญญานำเข้า โดยมีรหัสสัญญานำเข้า คือ IMP#6 และรหัสสัญญาส่งออก คือ EXP#@ (เครื่องหมาย @ เป็นค่าโดยปริยาย เนื่องจากต้องรอการอนุมัติจากผู้ดูแลเทรดเดอร์ส่งออกก่อน) ซึ่งจะเห็นว่าตัวกระทำการดังกล่าวสามารถทำงานได้อย่างถูกต้อง โดยสังเกตจากรูปที่ 5.12 จะเห็นข้อความว่า “New contract id is ‘IMP#6/EXP#@’ and inbox id IN#22 is removed.” และสังเกตจากรูปที่ 5.13 มีรหัสสัญญาข้อตกลง IMP#6/EXP#@ อยู่ใต้ไอคอน “Import Contracts” ในฝั่งซ้าย และมีรายละเอียดสัญญานำเข้าในฝั่งขวา

5.2.5 การทดสอบการส่งสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติ

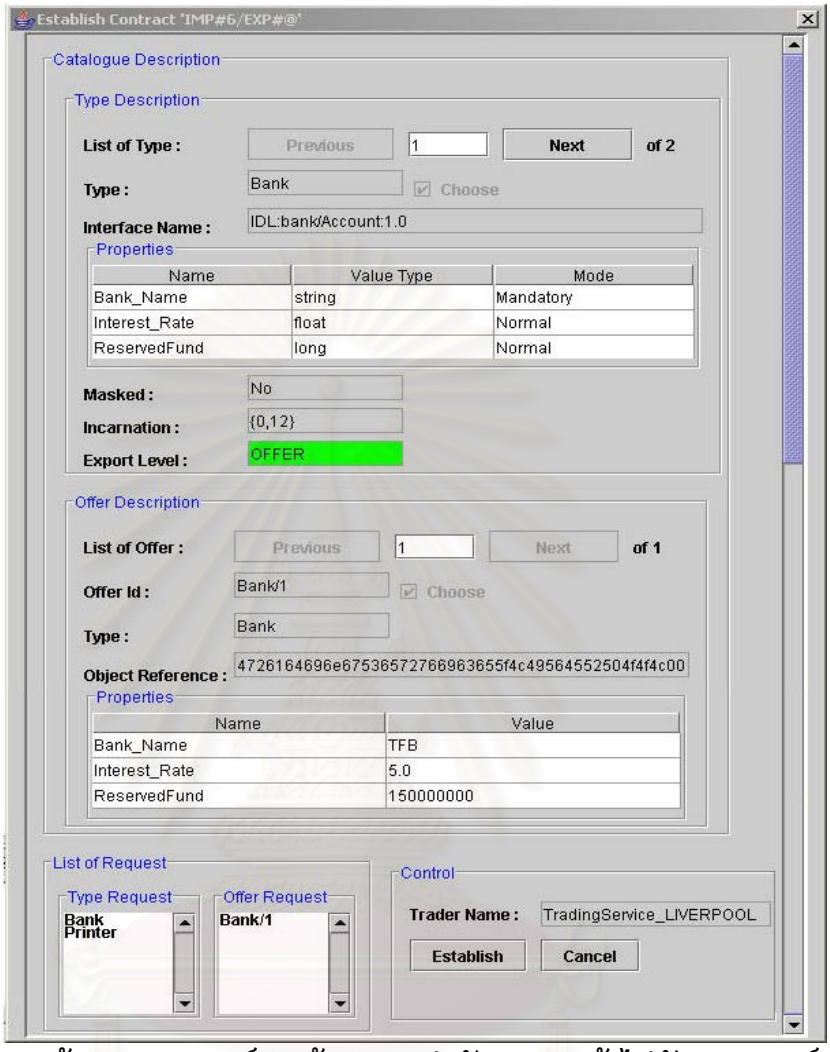
การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำลองเป็นผู้ดูแลเทรดเดอร์นำเข้าจำนวน 1 ตัว ทำการส่งสัญญานำเข้า (ข้อมูลที่ 3 หัวข้อ 5.2.4) ไปให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติ เพื่อทดสอบการทำงานของตัวกระทำการ establishFed() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า และทดสอบการทำงานของตัวกระทำการ exchangeContract() ของส่วนต่อประสาน FederationContract() ในฝั่งเทรดเดอร์ส่งออก จากนั้นตัวกระทำการ exchangeContract() จะทำการเรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก เพื่อแจ้งให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ โดยมีลำดับขั้นดังต่อไปนี้

1. เรียกใช้งานตัวกระทำการ establishFed() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า เพื่อส่งสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณา ดังรูปที่ 5.14

2.เรียกใช้งานตัวกระทำการ exchangeContract() ของส่วนต่อประสาน FederationContract ของเทรดเดอร์ส่งออก เพื่อแจ้งข้อมูลการพิจารณาสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ

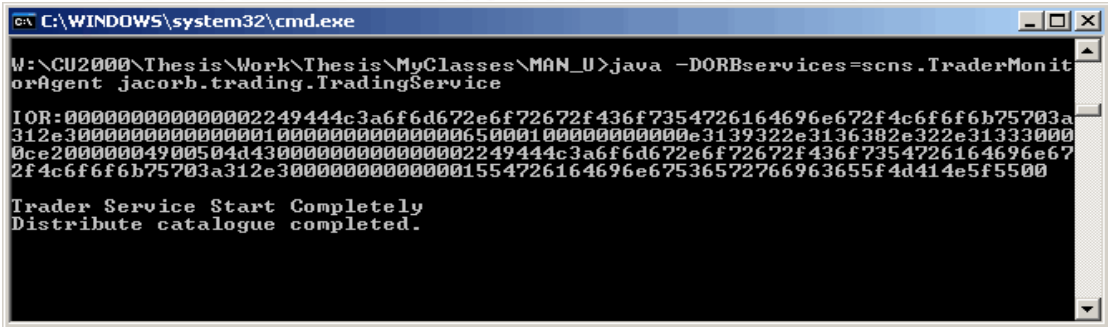
3. เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก เพื่อทำการเก็บข้อมูลการพิจารณาสัญญานำเข้าไว้ในกล่องข้อความขาเข้าของเทรดเดอร์ส่งออก

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.14 - 5.18



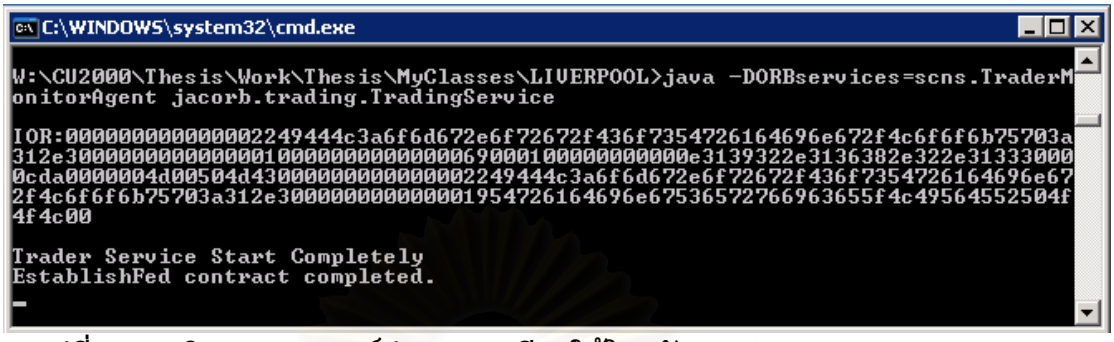
รูปที่ 5.14 ผู้ดูแลเทรดเดอร์นำเข้าทำการส่งสัญญานำเข้าไปยังเทรดเดอร์ส่งออก

รูปที่ 5.14 ผู้ดูแลเทรดเดอร์นำเข้าทำการส่งสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์ส่งออก โดยกดปุ่ม Ok เพื่อเรียกใช้งานตัวกระทำการ establishFed() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า



รูปที่ 5.15 การเรียกใช้ตัวกระทำการ establishFed() ในฝั่งเทรดเดอร์นำเข้า

รูปที่ 5.15 แสดงการทำงานของเทรดเดอร์นำเข้า เมื่อเรียกใช้ตัวกระทำการ establishFed() เพื่อส่งสัญญานำเข้าไปให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติ

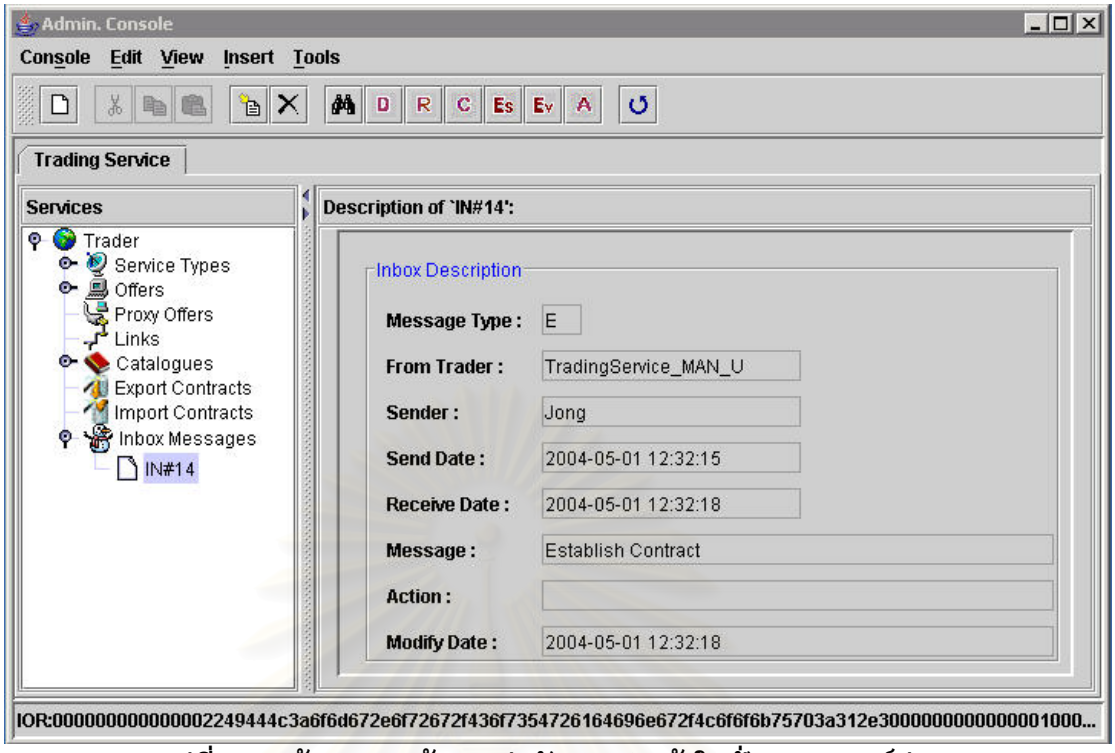


รูปที่ 5.16 บริการเทรดเดอร์ส่งออกถูกเรียกใช้โดยตัวกระทำการ exchangeContract()

รูปที่ 5.16 แสดงการทำงานของเทรดเดอร์ส่งออก เมื่อเรียกใช้ตัวกระทำการ exchangeContract() เพื่อแลกเปลี่ยนสัญญาข้อตกลง

รูปที่ 5.17 ผลลัพธ์จากการส่งสัญญานำเข้าไปให้ผู้ดูเทรดเดอร์พิจารณา

รูปที่ 5.17 แสดงผลลัพธ์เมื่อเทรดเดอร์ส่งออกทำการเพิ่มข้อความแจ้งการส่งสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติ และเก็บไว้ในกล่องในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออกเรียบร้อยแล้ว



รูปที่ 5.18 ข้อความแจ้งการส่งสัญญานำเข้าในฝั่งเทรดเดอร์ส่งออก

รูปที่ 5.18 แสดงข้อความแจ้งการส่งสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติ โดยถูกเก็บไว้ในคลังสัญญาส่งออกแล้ว

จากผลการทำงานในรูปที่ 5.14 – 5.18 สามารถอธิบายได้ว่าการทำงานของตัวกระทำการ establishFed() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้าสามารถส่งข้อความแจ้งเรื่องส่งสัญญานำเข้าให้ผู้ดูแลเทรดเดอร์ส่งออกทราบ เพื่อทำการพิจารณาอนุมัติได้อย่างถูกต้อง โดยสังเกตจากรูปที่ 5.16 - 5.17 จะเห็นว่ามีข้อความ “EstablishFed contract completed.” และสังเกตจากรูปที่ 5.18 มีรหัสข้อความนำเข้า IN#14 อยู่ใต้ไอคอน “Inbox Messages” ในฝั่งซ้าย และมีรายละเอียดข้อความแจ้งการส่งสัญญานำเข้าเพื่อให้ผู้ดูแลเทรดเดอร์ส่งออกพิจารณาอนุมัติในฝั่งขวา

5.2.6 การทดสอบการอนุมัติสัญญาข้อตกลงและสร้างสัญญาส่งออก

การทดสอบกระทำโดยใช้โปรแกรม GUI ที่พัฒนาขึ้น จำลองเป็นผู้ดูแลเทรดเดอร์ส่งออกจำนวน 1 ตัว ทำการอนุมัติและสร้างสัญญาส่งออก โดยโปรแกรมจะทำการอนุมัติสัญญานำเข้า (ข้อมูลที่ 3 หัวข้อที่ 5.2.5) ที่ส่งมาจากผู้ดูแลเทรดเดอร์นำเข้า โดยมีรายละเอียดของข้อมูลดังต่อไปนี้

ข้อมูลที่ 4 ข้อมูลสัญญาส่งออก มีรายละเอียดดังนี้

สัญญาข้อตกลง ExpContractID :EXP#5 (รหัสสัญญาส่งออกที่ทับค่า EXP#0 ไปแล้ว) ImpContractID :IMP#6 (รหัสสัญญานำเข้า)
บัญชีรายชื่อบริการ จากข้อมูลที่ 3 หัวข้อ 5.2.4 CatalogueID :CTL#1
รายการชนิดของบริการที่ร้องขอ Service Type Name :Bank, Printer
รายการข้อเสนอบริการที่ร้องขอ Name = TFB
เทรดเดอร์ที่ส่งออกบริการ Trader Name : TradingService_LIVERPOOL

การทำงานของโปรแกรมที่จำลองเป็นผู้ดูแลเทรดเดอร์ส่งออกประกอบด้วย

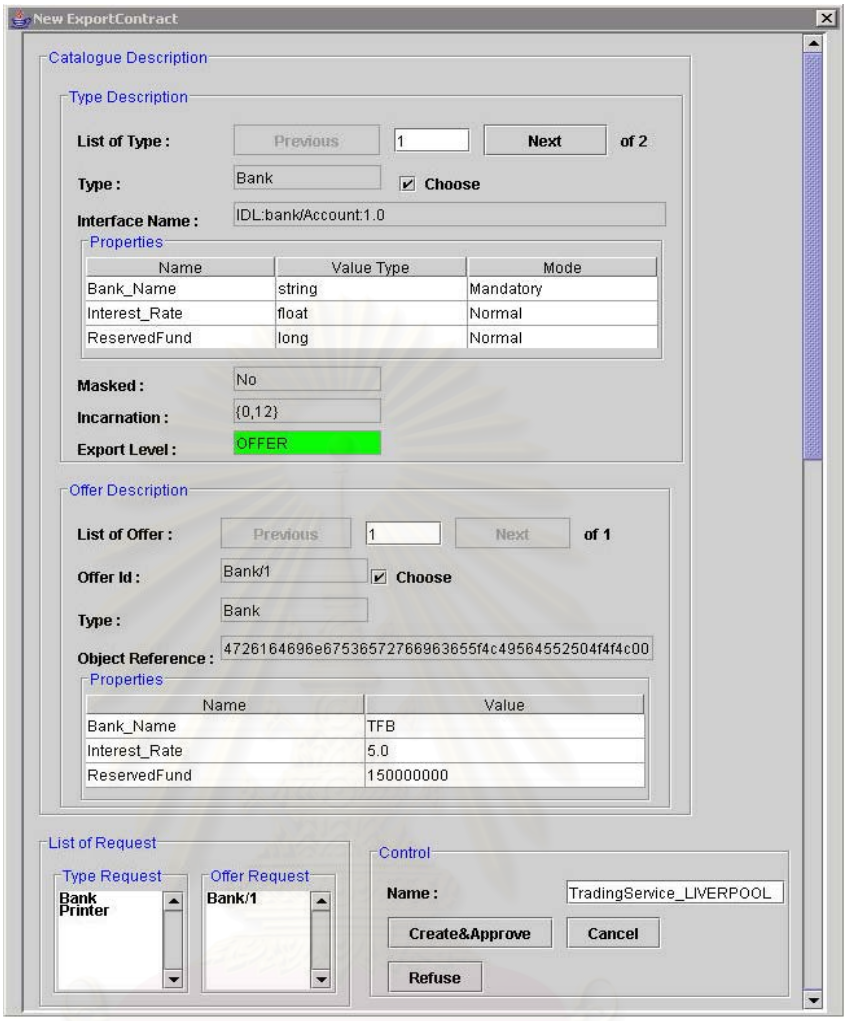
1. ผู้ดูแลเทรดเดอร์ส่งออกทำการเรียกใช้งานตัวกระทำการ addExportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก เพื่อสร้างสัญญาส่งออก ซึ่งจะได้รหัสสัญญาส่งออก คือ EXP#5 ดังข้อมูลที่ 4 คืนกลับมา ดังรูปที่ 5.19

2. ตัวกระทำการ addExportContract() เรียกใช้งานตัวกระทำการ evaluateContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อส่งสัญญาส่งออกที่มีรหัสสัญญาส่งออกแล้วไปให้ผู้ดูแลเทรดเดอร์นำเข้าทำการแก้ไขสัญญานำเข้าให้สอดคล้องกัน ดังรูปที่ 5.20

3. ตัวกระทำการ evaluateContract() เรียกใช้งานตัวกระทำการ evaluateResult() ของส่วนต่อประสาน FederationContract เพื่อแจ้งข้อมูลผลการอนุมัติสัญญาข้อตกลงให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ ดังรูปที่ 5.21

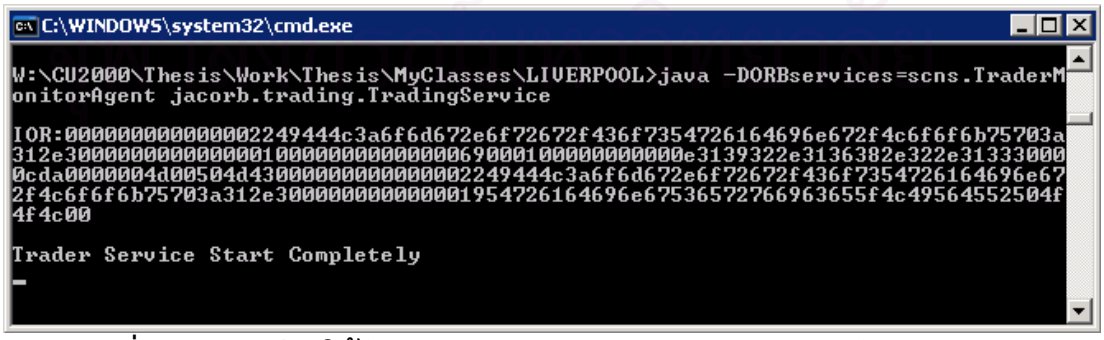
4. ตัวกระทำการ evaluateResult() เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า เพื่อทำการเก็บข้อมูลผลการอนุมัติสัญญาข้อตกลงไว้ในกล่องข้อความขาข้าวของเทรดเดอร์นำเข้า ดังรูปที่ 5.21

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.19 - 5.25



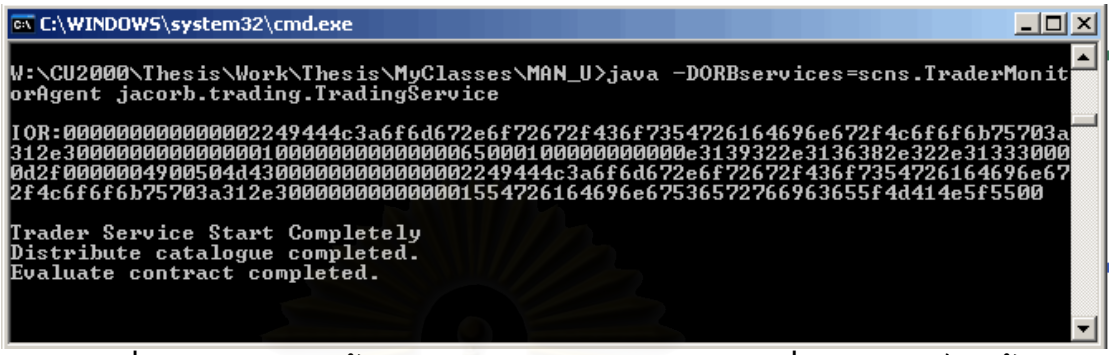
รูปที่ 5.19 ผู้ดูแลเทรดเดอร์ส่งออกทำการพิจารณาอนุมัติสัญญา

รูปที่ 5.19 ผู้ดูแลเทรดเดอร์ส่งออกทำการอนุมัติหรือปฏิเสธสัญญานำเข้า และส่งข้อความแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยกดปุ่ม Create&Approve เพื่อเรียกใช้งานตัวกระทำการ addExportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก



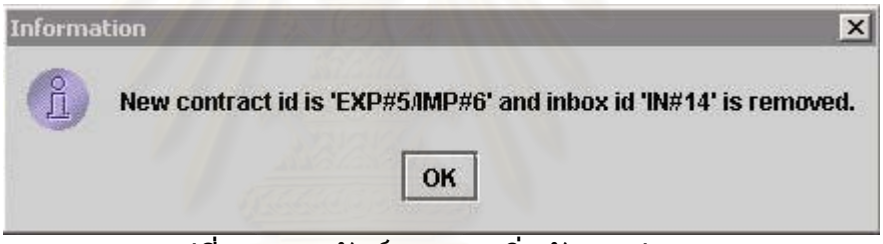
รูปที่ 5.20 การเรียกใช้ตัวกระทำการ addExportContract() และตัวกระทำการ evaluateContract() ในฝั่งเทรดเดอร์ส่งออก

รูปที่ 5.20 แสดงการเรียกใช้ตัวกระทำการ addExportContract() ในฝั่งเทรดเดอร์ส่งออก เพื่อทำการสร้างสัญญาส่งออก และเรียกใช้ตัวกระทำการ evaluateContract() เพื่อส่งสัญญาที่ผ่านการพิจารณาแล้วไปให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ



รูปที่ 5.21 การเรียกใช้ตัวกระทำการ evaluateResult() ในฝั่งเทรดเดอร์นำเข้า

รูปที่ 5.21 แสดงการเรียกใช้ตัวกระทำการ evaluateResult() ในฝั่งเทรดเดอร์นำเข้า เพื่อแจ้งให้ผู้ดูแลเทรดเดอร์นำเข้าทราบผลการอนุมัติสัญญาข้อตกลง

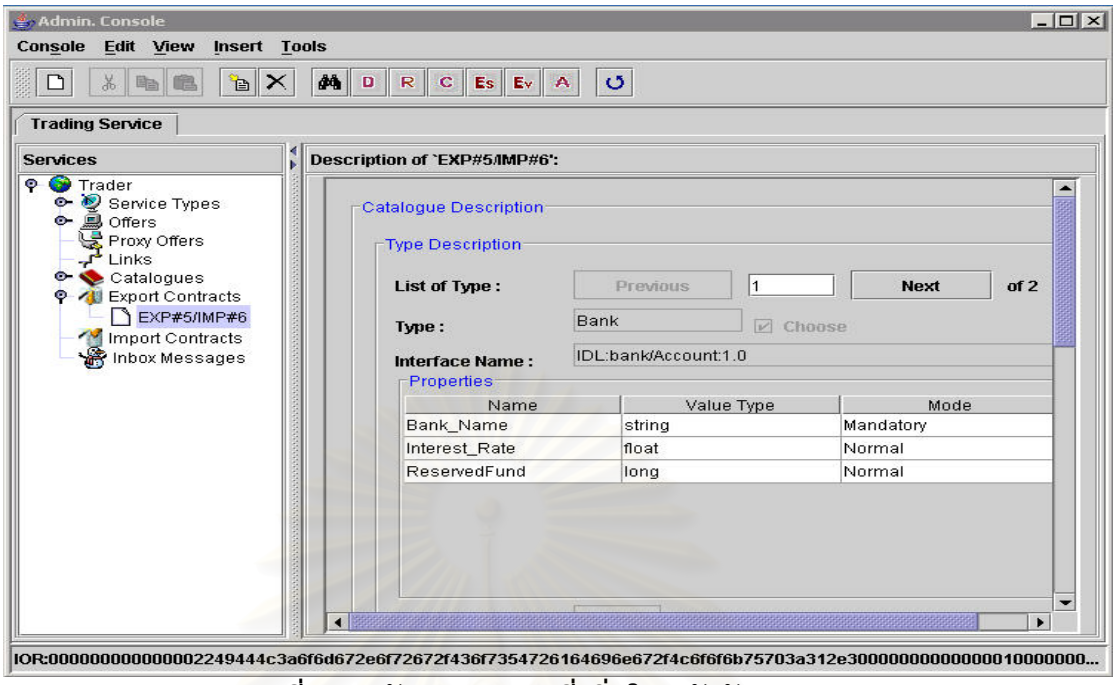


รูปที่ 5.22 ผลลัพธ์จากการเพิ่มสัญญาส่งออก

รูปที่ 5.22 แสดงผลลัพธ์เมื่อเทรดเดอร์ส่งออกทำการเพิ่มสัญญาส่งออกและเก็บไว้ในคลังสัญญาส่งออก และคืนค่ารหัสสัญญานำเข้า คือ IMP#6 และรหัสสัญญาส่งออกคือ EXP#5 และทำการลบข้อความรหัส IN#14 ออกจากกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออก

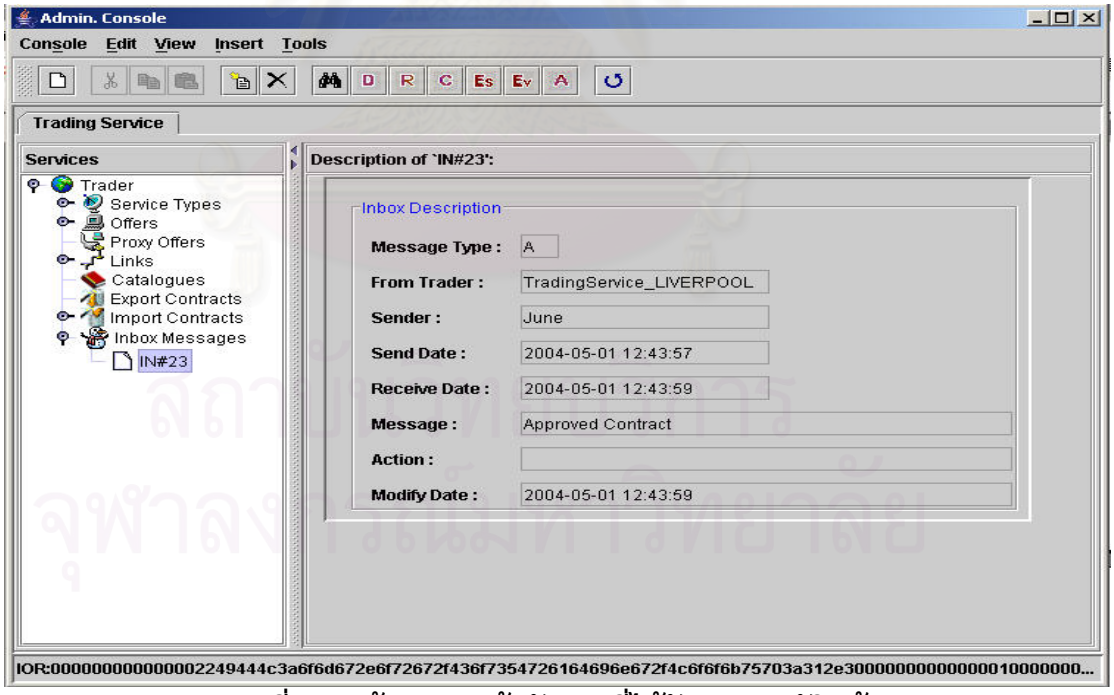
รูปที่ 5.23 ผลลัพธ์จากการส่งสัญญานำเข้าไปให้ผู้ดูเทรดเดอร์พิจารณาอนุมัติ

รูปที่ 5.23 แสดงผลลัพธ์เมื่อเทรดเดอร์นำเข้าทำการเพิ่มข้อความแจ้งผลการอนุมัติสัญญาข้อตกลงให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ และเก็บไว้ในกล่องในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์นำเข้าเรียบร้อยแล้ว



รูปที่ 5.24 สัญญาส่งออกที่เพิ่มในคลังสัญญาส่งออก

รูปที่ 5.24 แสดงรายละเอียดสัญญาส่งออกที่สร้างขึ้นโดยผู้ดูแลเทรดเดอร์ส่งออก ภายหลังจากเทรดเดอร์ส่งออกทำการเพิ่มสัญญาส่งออกไว้ในคลังสัญญาส่งออกเรียบร้อยแล้ว



รูปที่ 5.25 ข้อความแจ้งสัญญาที่ได้รับการอนุมัติแล้ว

รูปที่ 5.25 แสดงข้อความแจ้งผลการอนุมัติสัญญาให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ โดยถูกเก็บไว้ในกล่องข้อความขาเข้าในฝั่งเทรดเดอร์นำเข้า

จากผลการทำงานในรูปที่ 5.19 – 5.25 สามารถอธิบายได้ดังนี้

1.การทำงานของตัวกระทำการ addExportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์ส่งออก สามารถสร้างสัญญาข้อตกลงเก็บไว้ในคลังสัญญาส่งออก โดยมีรหัสสัญญาส่งออก คือ EXP#5 และรหัสสัญญานำเข้า คือ IMP#6 และจะเห็นว่าค่าตัวเลข 5 ของรหัสส่งออกได้แทนค่า @ ไปแล้ว อันเป็นผลจากการอนุมัติจากผู้ดูแลเทรดเดอร์ส่งออก ซึ่งจะเห็นว่าตัวกระทำการสามารถทำงานได้อย่างถูกต้อง โดยสังเกตได้จากรูปที่ 5.22 จะเห็นว่ามีข้อความ “New contract id is ‘EXP#5/IMP#6’ and inbox id IN#14 is removed.” และสังเกตจากรูปที่ 5.24 มีรหัสสัญญาข้อตกลง EXP#5/IMP#6 อยู่ใต้ไอคอน “Export Contracts” ในฝั่งซ้าย และมีรายละเอียดสัญญาส่งออกในฝั่งขวา

2. การทำงานของตัวกระทำการ evaluateContract() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์ส่งออก สามารถส่งข้อความแจ้งผลการอนุมัติสัญญาข้อตกลงให้ผู้ดูแลเทรดเดอร์นำเข้าทราบ เพื่อทำการแก้ไขสัญญานำเข้าให้สอดคล้องกัน ซึ่งจะเห็นว่าได้ว่าตัวกระทำการสามารถทำงานได้อย่างถูกต้อง โดยสังเกตได้จากรูปที่ 5.21 กับ 5.23 จะเห็นว่ามีข้อความ “Evaluate contract completed.” และสังเกตจากรูปที่ 5.25 มีรหัสข้อความ IN#23 อยู่ใต้ไอคอน “Inbox Messages” ในฝั่งซ้าย และมีรายละเอียดข้อความแจ้งผลการอนุมัติสัญญาข้อตกลงในฝั่งขวา

5.2.7 การทดสอบการแก้ไขสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออกและการทดสอบตัวเรียกกลับนำเข้าเพื่อแจ้งการเกิดบริการ

การทดสอบกระทำโดยใช้โปรแกรม ประกอบด้วย

1.โปรแกรมจำลองเป็นผู้ดูแลเทรดเดอร์นำเข้า จำนวน 1 ตัว ทำการแก้ไขสัญญานำเข้าและสร้างวัตถุตัวเรียกกลับนำเข้า โดยโปรแกรมจะทำการแก้ไขรหัสสัญญาส่งออกที่มีรหัส คือ EXP#@ ในสัญญานำเข้า (ข้อมูลที่ 3 หัวข้อที่ 5.2.4) ให้มีค่าเป็น EXP#5 เพื่อให้สอดคล้องกับสัญญาส่งออก ดังข้อมูลที่ 5

ข้อมูลที่ 5 ข้อมูลสัญญานำเข้าภายหลังจากการแก้ไขรหัสสัญญาส่งออก มีรายละเอียดดังนี้

สัญญานำเข้า ExpContractID :EXP#5 (รหัสสัญญาส่งออกที่ทับค่า EXP#@ ไปแล้ว) ImpContractID :IMP#6 (รหัสสัญญานำเข้า)
บัญชีรายชื่อบริการ จากข้อมูลที่ 3 หัวข้อ 5.2.4 CatalogueID :CTL#1
รายการชนิดของบริการที่ร้องขอ Service Type Name :Bank, Printer
รายการข้อเสนอบริการที่ร้องขอ Bank Name = TFB
เทรดเดอร์ที่ส่งออกบริการ Trader Name : TradingService_LIVERPOOL

การทำงานของโปรแกรมที่จำลองเป็นผู้ดูแลเทรดเดอร์นำเข้าประกอบด้วย

1. ผู้ดูแลเทรดเดอร์นำเข้าทำการเรียกใช้งานตัวกระทำการ applyImportContract() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อทำการแก้ไขรหัสสัญญาส่งออกของสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออกดังข้อมูลที่ 5 แล้วเก็บลงในคลังสัญญานำเข้า
2. ตัวกระทำการ applyImportContract() เรียกใช้งานตัวกระทำการ register() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการกับตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลง (SubscriberManager) ของบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออก ภายหลังจากลงทะเบียนเสร็จก็จะทำการสร้างวัตถุตัวเรียกกลับนำเข้าเพื่อแจ้งเหตุการณ์การเกิดบริการใหม่ไปยังผู้รับบริการที่ลงทะเบียนขอรับทราบการเปลี่ยนแปลงกับบริการเอสซีเอ็นในฝั่งเทรดเดอร์นำเข้า และรอรับแจ้งเหตุการณ์การเปลี่ยนแปลงจากบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออกอีกด้วย
3. ตัวกระทำการ register() จากข้อ 2 จะเรียกใช้งานตัวกระทำการ register() ของส่วนต่อประสาน FederationContract เพื่อให้ทำการลงทะเบียนแทน ซึ่งจะคืนค่าวัตถุอ้างถึงวัตถุผู้ดูแลตัวแทนไปให้
4. ตัวกระทำการ register() ของส่วนต่อประสาน FederationContract() เรียกใช้งานตัวกระทำการ addInbox() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เพื่อทำการเก็บข้อมูลผลการอนุมัติสัญญาข้อตกลงไว้ในกล่องข้อความขาเข้าของเทรดเดอร์นำเข้า

2.โปรแกรมจำลองเป็นผู้ต้องการรับทราบการเปลี่ยนแปลง จำนวน 1 ตัว สำหรับรับข้อมูลการเกิดบริการขึ้นใหม่ ซึ่งเป็นผลมาจากการสร้างสัญญาข้อตกลงใหม่และอยู่ในฝั่งเทรดเดอร์นำเข้า โดยทำการลงทะเบียนโดยใช้ชื่อ user1 มีรหัสผ่านคือ iamuser1 ทำการสร้างตัวแทนประเภท StructuredProxyPushSubscriber และมีรายละเอียดของข้อมูลบริการที่ต้องการรับทราบการเปลี่ยนแปลง ดังข้อมูลที่ 6

ข้อมูลที่ 6 ข้อมูลบริการที่ต้องการทราบการเปลี่ยนแปลง (Subscription Information) มีรายละเอียดดังนี้

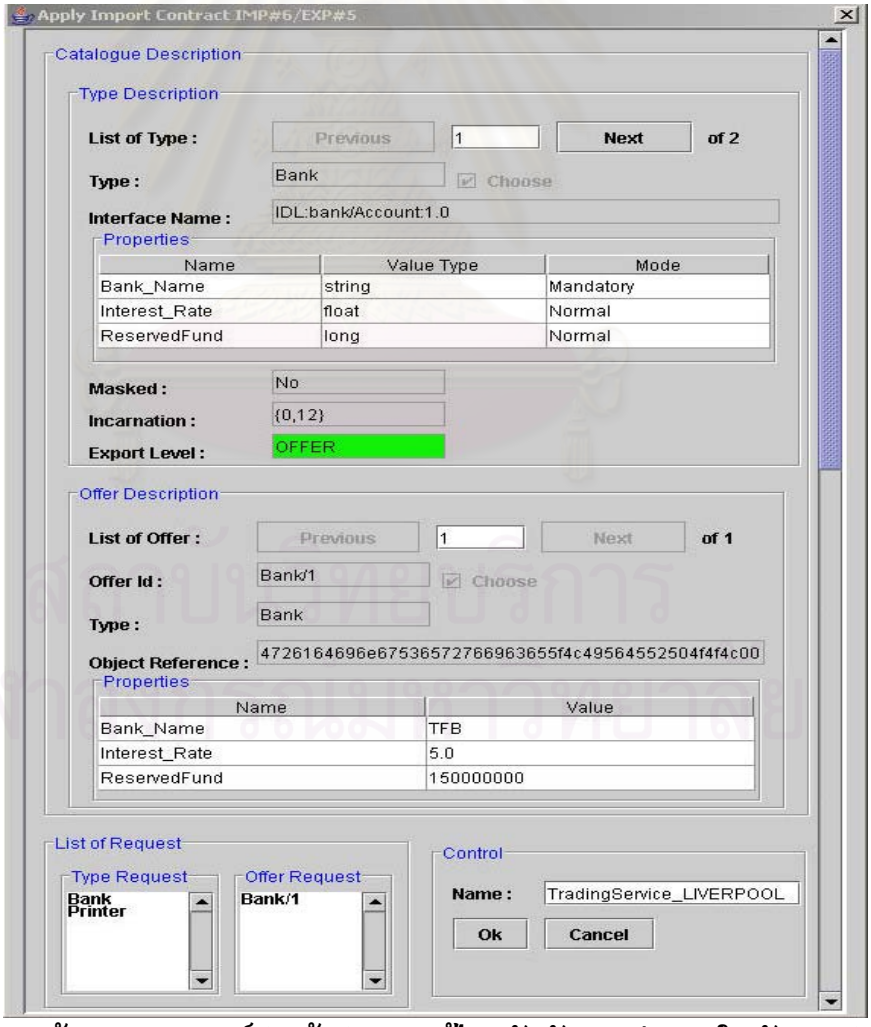
```
Kind Of Event = Instantiation of Service
Service Identification Type = Service not specified
```

การทำงานของโปรแกรมผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการประกอบด้วย

1. ลงทะเบียนเข้ากับบริการเอสซีเอ็นในฝั่งเทรดเดอร์นำเข้าและสร้างวัตถุตัวแทน โดยขอทราบเหตุการณ์การเกิดบริการใหม่ (Instantiation of Service) ดังข้อมูลที่ 6 โดยไม่ระบุชนิดบริการและข้อเสนอบริการ
2. เรียกใช้ตัวกระทำการ connect() ของส่วนต่อประสานวัตถุตัวแทนเพื่อติดต่อเข้ากับวัตถุตัวแทนสำหรับการรอรับข้อมูลการเปลี่ยนแปลงบริการ
3. รอรับข้อมูลแจ้งการเกิดบริการใหม่ ซึ่งเป็นผลมาจากการสร้างสัญญาข้อตกลงใหม่สำหรับผู้ต้องการรับทราบการเปลี่ยนแปลงแบบพุช

จากนั้นทำการรันโปรแกรมผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการ แล้วรันโปรแกรมผู้จำลองเป็นผู้ดูแลเทรดเดอร์นำเข้า

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.26 - 5.33

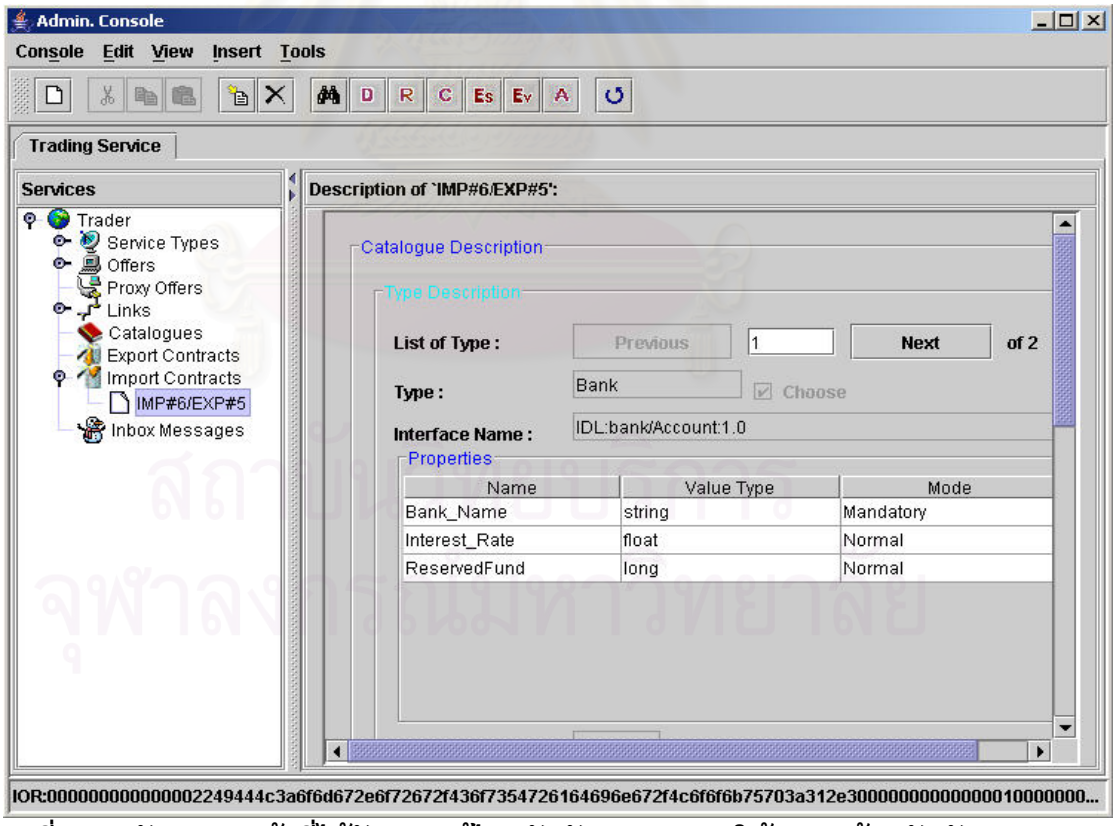


รูปที่ 5.26 ผู้ดูแลเทรดเดอร์นำเข้าทำการแก้ไขรหัสสัญญาส่งออกในสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก

รูปที่ 5.26 ผู้ดูแลเทรดเดอร์ทำการแก้ไขสัญญานำเข้าให้สอดคล้องกับสัญญาส่งออก โดยกดปุ่ม Ok ซึ่งจะทำการเรียกใช้งานตัวกระทำการ applyImportContract ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า แล้วทำการเรียกใช้ตัวกระทำการ register() เพื่อทำการลงทะเบียนกับบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออก โดยใช้ข้อมูลที่ร้องขอจากสัญญาข้อตกลง

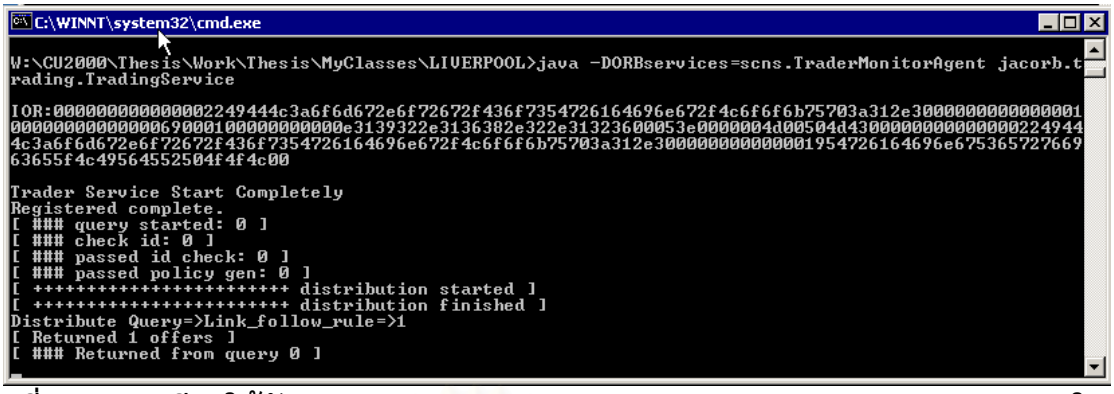
รูปที่ 5.27 ผลลัพธ์จากการแก้ไขรหัสสัญญาส่งออกในสัญญานำเข้า

รูปที่ 5.27 แสดงผลลัพธ์เมื่อเทรดเดอร์นำเข้าทำการแก้ไขรหัสสัญญาส่งออกในสัญญานำเข้าและเก็บไว้ในคลังสัญญานำเข้า และแจ้งให้ทราบว่ารหัสสัญญานำเข้า คือ IMP#6 และรหัสสัญญาส่งออกที่แก้ไขคือ EXP#5 และทำการลบข้อความรหัส IN#23 ออกจากกล่องข้อความขาเข้าในฝั่งเทรดเดอร์ส่งออก



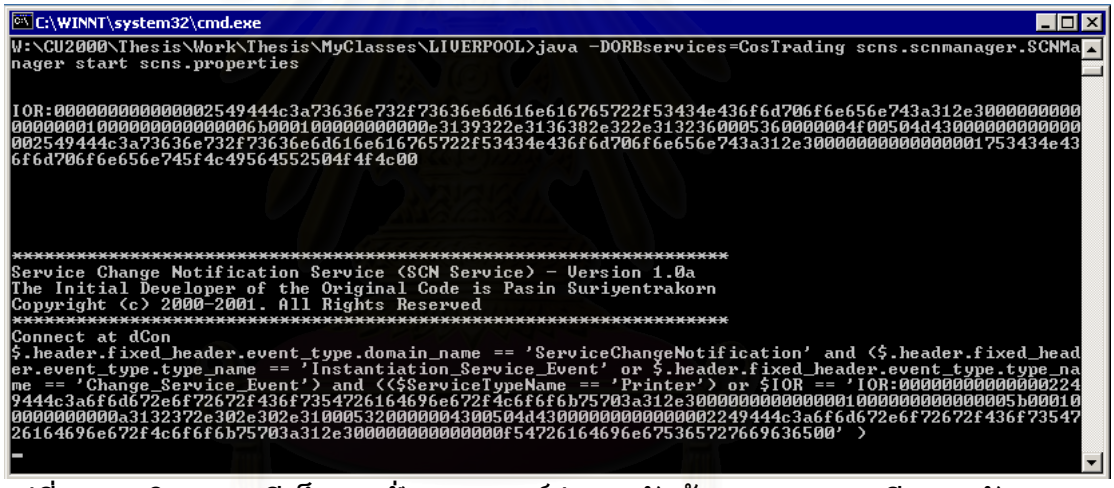
รูปที่ 5.28 สัญญานำเข้าที่ได้รับการแก้ไขรหัสสัญญาส่งออกให้สอดคล้องกับสัญญาส่งออก

รูปที่ 5.28 แสดงรายละเอียดสัญญานำเข้าที่ได้รับการแก้ไขรหัสสัญญาส่งออก โดยผู้ดูเทรดเดอร์นำเข้า เพื่อให้สอดคล้องกับสัญญาส่งออกในฝั่งเทรดเดอร์ส่งออก



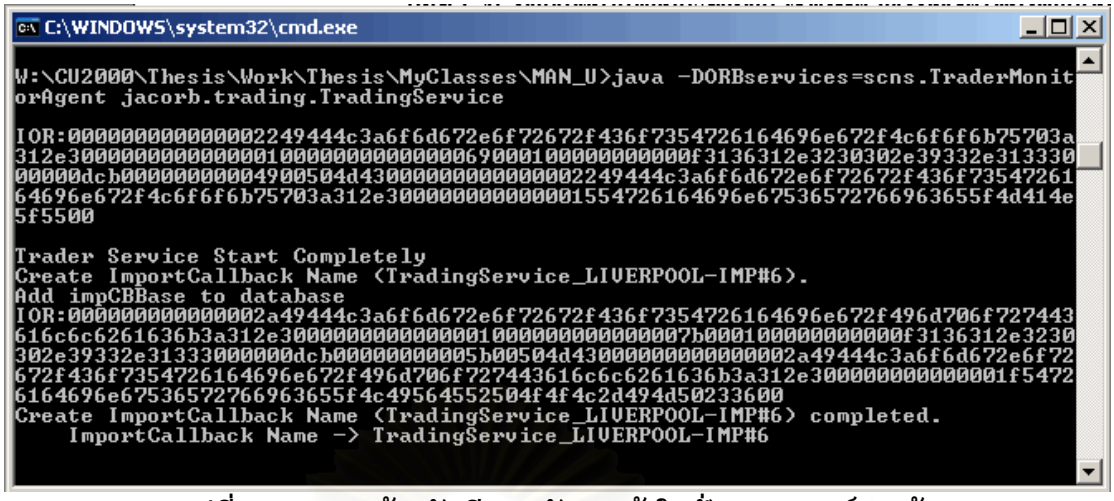
รูปที่ 5.29 การเรียกใช้ตัวกระทำการ register() ผ่านส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์ส่งออก

รูปที่ 5.29 แสดงการเรียกใช้ตัวกระทำการ register() ของส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์ส่งออก เพื่อขอลงทะเบียนกับบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออก



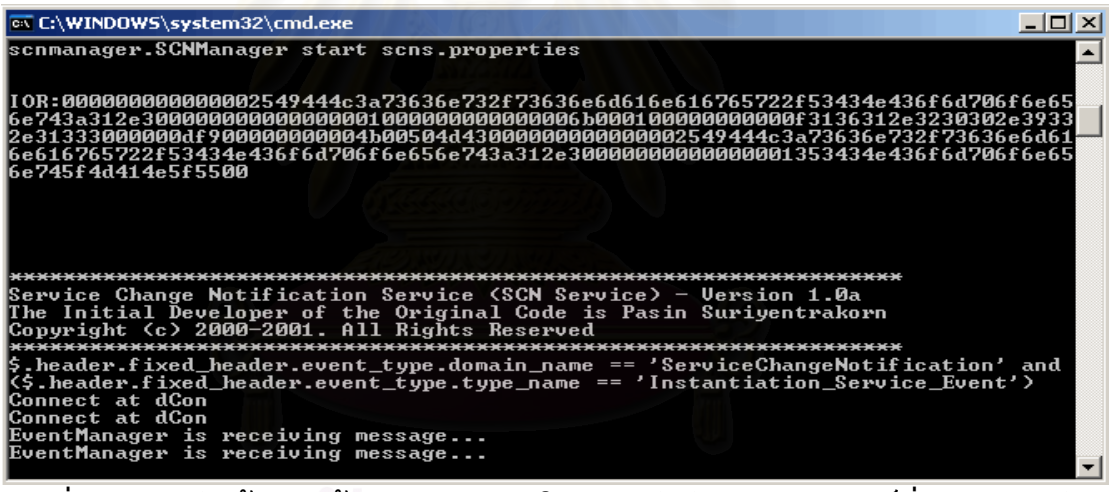
รูปที่ 5.30 บริการเอสซีเอ็นของฝั่งเทรดเดอร์ส่งออกรับข้อมูลการลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการ

รูปที่ 5.30 ตัวจัดการกับผู้ที่ขอทราบการเปลี่ยนแปลงของบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออก ทำการรับข้อมูลการลงทะเบียนขอทราบการเปลี่ยนแปลงและทำการลงทะเบียนไว้กับบริการแจ้งเหตุการณ์ (Notification Service)



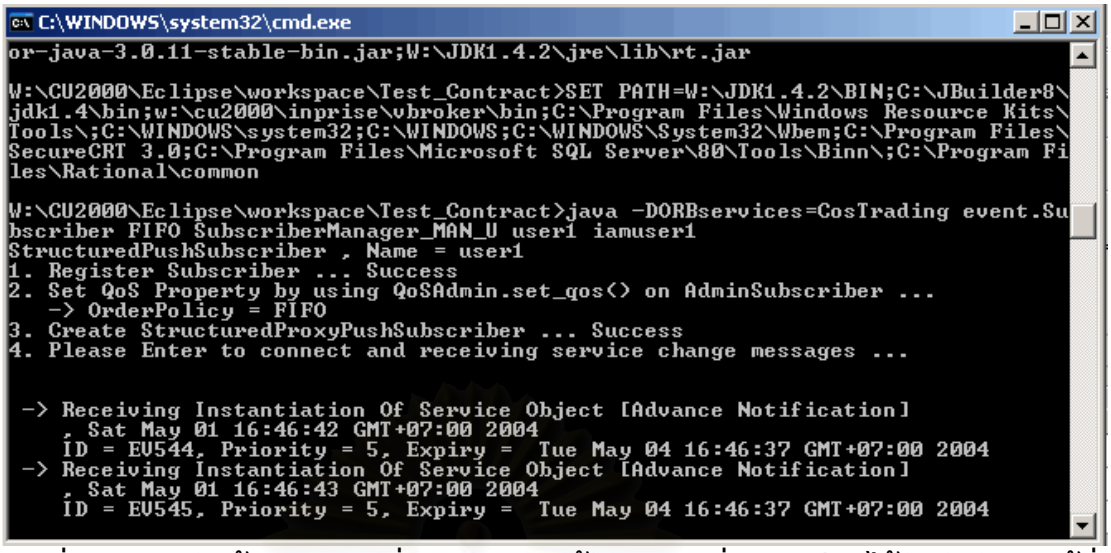
รูปที่ 5.31 การสร้างตัวเรียกกลับนำเข้าในฝั่งเทรดเดอร์นำเข้า

รูปที่ 5.31 เทรดเดอร์นำเข้าทำการสร้างตัวเรียกกลับนำเข้า ภายหลังจากการลงทะเบียนเพื่อขอทราบการเปลี่ยนแปลงกับบริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออกเรียบร้อยแล้ว และรอรับแจ้งการเปลี่ยนแปลงบริการจากเทรดเดอร์ฝั่งส่งออก



รูปที่ 5.32 การรับข้อมูลแจ้งการเกิดบริการใหม่ของตัวจัดการเหตุการณ์ที่ทำงานร่วมกับเทรดเดอร์นำเข้า

รูปที่ 5.32 ตัวจัดการเหตุการณ์ได้รับข้อมูลการเกิดบริการใหม่ ซึ่งเป็นผลมาจากคำร้องขอบริการในสัญญานำเข้าที่ได้ทำการแก้ไขให้สอดคล้องกับสัญญาส่งออก



รูปที่ 5.33 การรับข้อมูลการเปลี่ยนแปลงของผู้รับบริการที่ลงทะเบียนไว้กับตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงในฝั่งเทรดเดอร์นำเข้า

จากผลการทำงานในรูปที่ 5.26 – 5.33 สามารถอธิบายได้ดังนี้

1. การทำงานของตัวกระทำการ applyImportContract() ของส่วนต่อประสาน FederationAdmin ของเทรดเดอร์นำเข้า สามารถแก้ไขสัญญานำเข้าที่เก็บอยู่ในคลังสัญญานำเข้า โดยมีรหัสสัญญานำเข้า คือ IMP#6 และรหัสสัญญาส่งออก คือ EXP#5 โดยค่าตัวเลข 5 ของรหัสส่งออกได้แทนค่า @ ไปแล้ว (ข้อมูลที่ 5) ซึ่งจะเห็นว่าตัวกระทำการสามารถทำงานได้อย่างถูกต้อง โดยสังเกตได้จากรูปที่ 5.27 จะเห็นว่ามีข้อความ “Apply contract id is ‘IMP#6/EXP#5’ and inbox id IN#23 is removed.” และสังเกตจากรูปที่ 5.24 มีรหัสสัญญาข้อตกลง IMP#6/EXP#5 อยู่ใต้ไอคอน “Import Contracts” รูปฐานข้อมูลในฝั่งซ้าย และมีรายละเอียดสัญญานำเข้าในฝั่งขวา

2. การทำงานของตัวกระทำการ register() ของส่วนต่อประสาน FederationAdmin ในฝั่งเทรดเดอร์นำเข้า เรียกใช้งานตัวกระทำการ register() ของส่วนต่อประสาน FederationContract ในฝั่งเทรดเดอร์ส่งออก สามารถทำงานได้อย่างถูกต้อง โดยสังเกตได้จากด้านล่างรูปที่ 5.30 จะเห็นว่ามีข้อมูลคำร้องขอบริการจากสัญญาข้อตกลงที่ใช้ทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลง

3. วัตถุของตัวเรียกกลับนำเข้าถูกสร้างขึ้นทำการส่งข้อความแจ้งการเกิดบริการขึ้นใหม่โดยอาศัยข้อมูลร้องขอบริการจากสัญญานำเข้าเป็นเงื่อนไขและผู้รับบริการที่ทำการลงทะเบียนเพื่อขอรับการเปลี่ยนแปลงกรณีเกิดบริการขึ้นใหม่สามารถงานได้อย่างถูกต้อง โดยสังเกตได้จากรูปที่ 5.33 จะเห็นว่าผู้รับบริการที่ลงทะเบียนขอทราบการเกิดบริการขึ้นใหม่ได้รับแจ้งเตือนเหตุการณ์ครบ ซึ่งบริการที่เกิดขึ้นใหม่ ได้แก่ ชนิดบริการ Bank ที่มีข้อเสนอบริการ คือ “TFB” และชนิดบริการ Printer ที่มีข้อเสนอบริการ คือ “HP LaserJet 1100”

5.2.8 การทดสอบการจัดส่งและรับข้อมูลการเปลี่ยนแปลงบริการจากเทรดเดอร์ส่งออก

วิทยานิพนธ์นี้ได้นำข้อมูลตัวอย่างบางส่วนของงานวิจัย [6] มาใช้เป็นข้อมูลในการทดสอบ โดยการทดสอบกระทำโดยใช้โปรแกรม ประกอบด้วย

1. โปรแกรมจำลองเป็นผู้แจ้งการเปลี่ยนแปลงบริการล่วงหน้า จำนวน 1 ตัว ทำการส่งข้อมูลการเปลี่ยนแปลงบริการไปยังฝั่งเทรดเดอร์ส่งออก โดยโปรแกรมจะทำการสร้างข้อมูลการเปลี่ยนแปลงที่แตกต่างกันจำนวน 4 ข้อมูล โดยจัดส่งตามลำดับจากข้อมูลที่ 7 ถึงข้อมูลที่ 10 โดยข้อมูลแต่ละตัวมีรายละเอียดดังต่อไปนี้

ข้อมูลที่ 7 ข้อมูลการเพิ่มบริการใหม่ มีรายละเอียดดังนี้

ค่าความสำคัญ = 5

ค่าวัน-เวลาหมดอายุ = 3 วัน

ชนิดของบริการ

```
Service Type Name :Bank

Property Definition :
  Name           [string]
  ReservedFund   [long]
  Interest_Rate  [float]

interface Bank {
   float balance (in string accno);
   boolean deposit (in string accno, in float amount);
   boolean withdraw(in string accno, in float amount);
};
```

รายการคุณสมบัติ

```
Name          = BBL
ReservedFund  = 100000000
Interest_Rate = 5.0
```

ข้อมูลที่ 8 ข้อมูลการลบบริการ มีรายละเอียดดังนี้

ค่าความสำคัญ = 10

ค่าวัน-เวลาหมดอายุ = 7 วัน

ชนิดของบริการ

```
Service Type Name :Printer

Property Definition :
  Type   [string]
  Name   [string]
  Speed  [long]

interface PrinterService {
  boolean print(in string url);
};
```

รายการคุณสมบัติ

```
Type  =Laser Printer
Name  =HP LaserJet 1100
Speed =20
```

ข้อมูลที่ 9 ข้อมูลการเปลี่ยนแปลงค่าคุณสมบัติของบริการ มีรายละเอียดดังนี้

ค่าความสำคัญ = 8

ค่าวัน-เวลาหมดอายุ = 5 วัน

ชนิดของบริการ

```
Service Type Name :Printer

Property Definition :
  Type   [string]
  Name   [string]
  Speed [long]

interface PrinterService {
  boolean print(in string url);
};
```

รายการคุณสมบัติ (ก่อนการเปลี่ยนแปลง)

```
Type   = Laser Printer
Name   = HP LaserJet 1100
Speed = 20
```

รายการคุณสมบัติ (หลังการเปลี่ยนแปลง)

```
Type   = Laser Printer
Name   = HP LaserJet Super Turbo
Speed = 30
```

ข้อมูลที่ 10 ข้อมูลการเปลี่ยนแปลงชนิดของบริการ มีรายละเอียดดังนี้

ค่าความสำคัญ = 10

ค่าวัน-เวลาหมดอายุ = 7 วัน

ชนิดของบริการ (ก่อนการเปลี่ยนแปลง)

```
Service Type Name :Bank

Property Definition :
  Name           [string]
  ReservedFund   [long]
  Interest_Rate [float]

 interface Bank {
  float balance (in string accno);
  boolean deposit (in string accno, in float amount);
  boolean withdraw(in string accno, in float amount);
};
```

รายการคุณสมบัติ (ก่อนการเปลี่ยนแปลง)

```
Name          = TFB
ReservedFund  = 150000000
Interest_Rate = 5.0
```

ชนิดของบริการ (หลังการเปลี่ยนแปลง)

```
Service Type Name : Bank_2

Property Definition :
  Name           [string]
  ReservedFund   [long]
  Interest_Rate [float]

 interface Bank_2 {
   float balance   (in string accno);
```

```
    boolean deposit (in string accno,
                     in long type,
                     in float amount);
    boolean withdraw(in string accno,
                     in long type
                     in float amount);
    boolean transfer(in string accno_src,
                     in string accno_dest,
                                in float amount);
};
```

รายการคุณสมบัติ (หลังการเปลี่ยนแปลง)

```
Name           = TFB
ReservedFund   = 150000000
Interest_Rate = 5.0
```

2. โปรแกรมจำลองเป็นผู้ต้องการรับทราบการเปลี่ยนแปลง จำนวน 1 ตัว สำหรับรับข้อมูลการเปลี่ยนแปลงบริการและอยู่ในฝั่งเทรดเดอร์นำเข้า โดยทำการลงทะเบียนโดยใช้ชื่อ user2 มีรหัสผ่านคือ iamuser2 ทำการสร้างวัตถุตัวแทนประเภท StructuredProxyPushSubscriber และมีรายละเอียดของข้อมูลบริการที่ต้องการรับทราบการเปลี่ยนแปลง ดังข้อมูลที่ 11

ข้อมูลที่ 11 ข้อมูลบริการที่ต้องการทราบการเปลี่ยนแปลง (Subscription Information) มีรายละเอียดดังนี้

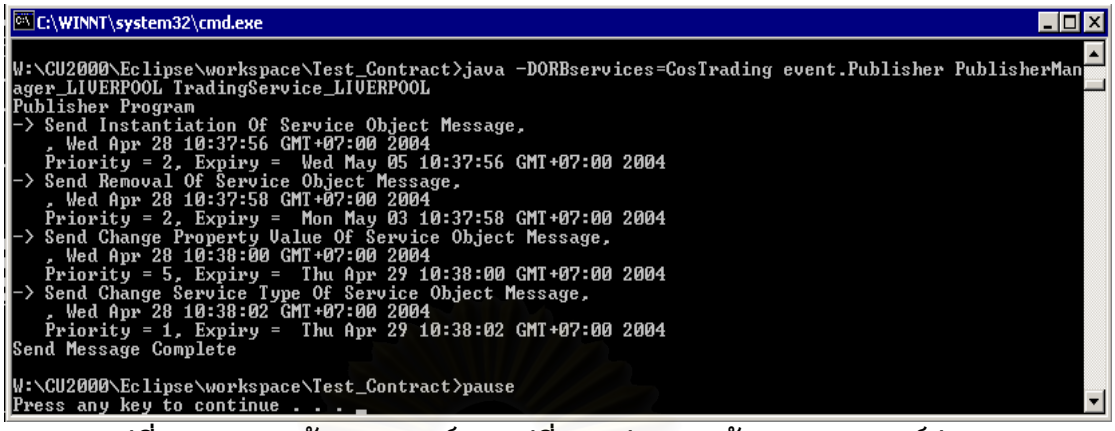
```
Kind Of Event = Change of Service
Service Identification Type = TraderConstraint
ServiceTypeName = Bank, Printer
Property Name = Name   Value = TFB
```

การทำงานของโปรแกรมผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการประกอบด้วย

1. ลงทะเบียนเข้ากับบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์นำเข้าและสร้างวัตถุตัวแทนโดยขอทราบเหตุการณ์การเปลี่ยนแปลงบริการ (Change Service) ดังข้อมูลที่ 11 โดยระบุชนิดบริการ คือ “Bank” และข้อเสนอบริการที่มีชื่อ คือ “TFB”
2. เรียกตัวกระทำการ connect () ของส่วนต่อประสานวัตถุตัวแทนแต่ละตัวเพื่อติดต่อเข้ากับวัตถุตัวแทนสำหรับการรอรับข้อมูลการเปลี่ยนแปลงบริการ
3. รอรับข้อมูลการเปลี่ยนแปลงบริการสำหรับผู้ต้องการรับทราบการเปลี่ยนแปลงที่สร้างวัตถุตัวแทนแบบพุช

จากนั้นทำการรันโปรแกรมผู้ต้องการรับทราบการเปลี่ยนแปลงบริการ แล้วรันโปรแกรมผู้แจ้งการเปลี่ยนแปลง

ผลการทำงานของโปรแกรมสามารถแสดงได้ดังรูปที่ 5.34 - 5.38



รูปที่ 5.34 การแจ้งเหตุการณ์การเปลี่ยนแปลงของผู้ดูแลเทรดเดอร์ส่งออก

จากรูปที่ 5.34 ผู้แจ้งเหตุการณ์จะทำการส่งข้อมูลการเปลี่ยนแปลง (ข้อมูลที่ 7-10) โดยแจ้งการเปลี่ยนแปลงบริการไปยังบริการเอสซีเอ็นของเทรดเดอร์ส่งออกตามลำดับข้อมูลดังนี้

1. การเพิ่มบริการใหม่
2. การยกเลิกบริการ
3. การเปลี่ยนแปลงค่าคุณสมบัติของบริการ
4. การเปลี่ยนแปลงชนิดบริการ

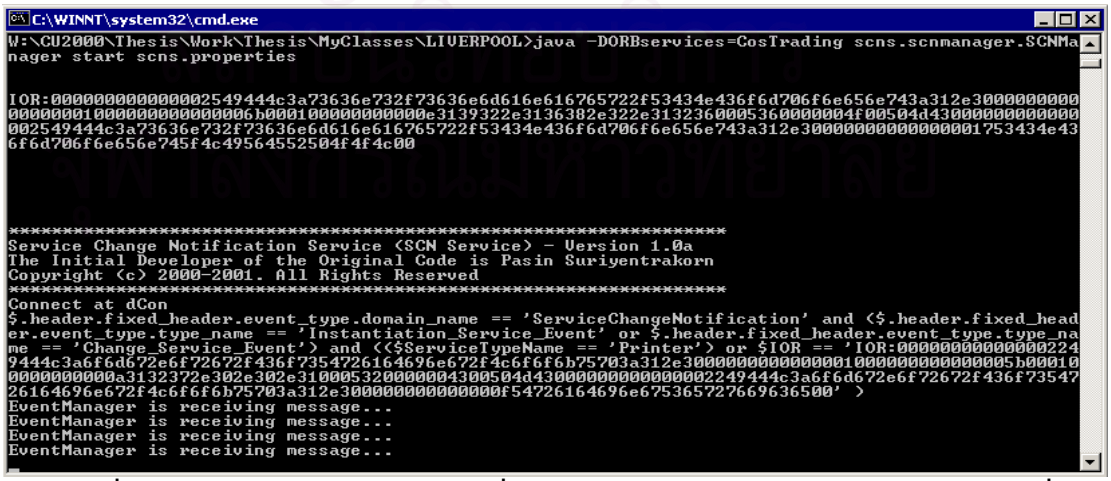
ซึ่งในรูปที่ 5.34 จะแสดงข้อความที่แจ้งดังนี้

-> Send Instantiation of Service Object Message

-> Send Removal of Service Object Message

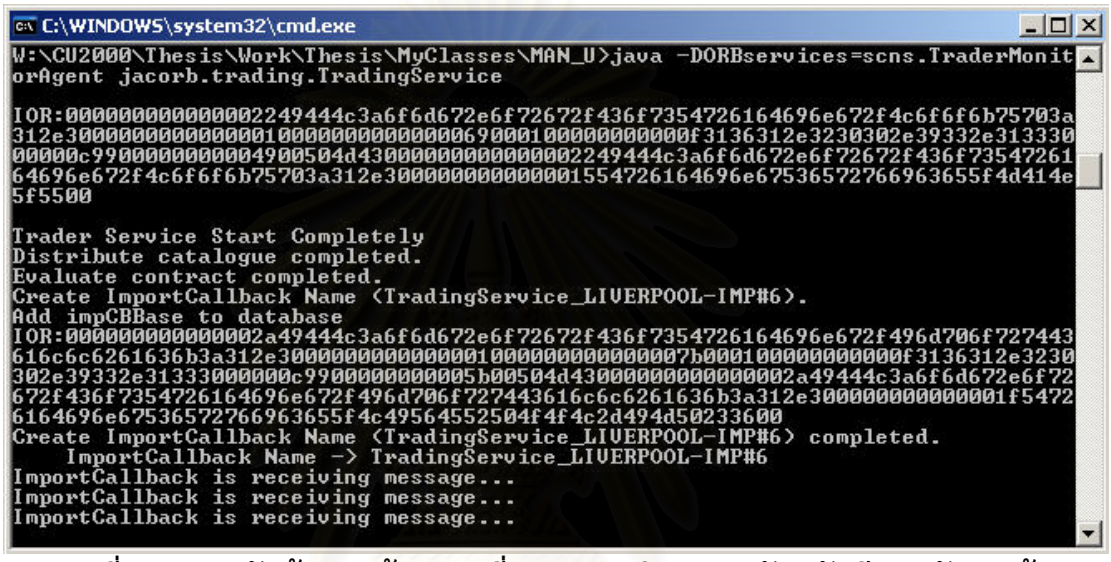
-> Send Change Property Value of Service Object Message

-> Send Change Service Type of Service Object Message



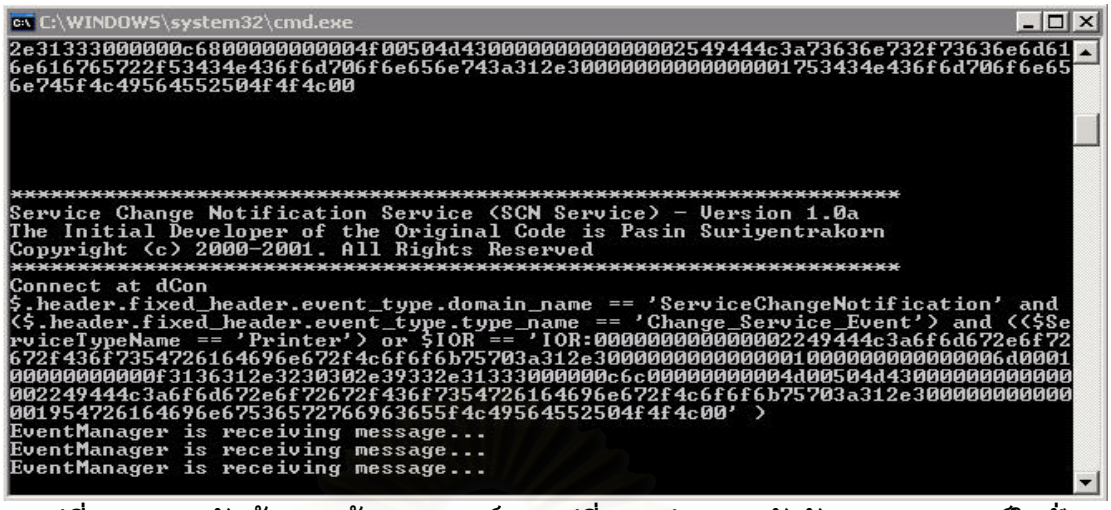
รูปที่ 5.35 การรับข้อมูลแจ้งการเปลี่ยนแปลงบริการของตัวจัดการเหตุการณ์ในฝั่งเทรดเดอร์ส่งออก

จากรูปที่ 5.35 บริการเอสซีเอ็นในฝั่งเทรดเดอร์ส่งออกได้รับข้อมูลแจ้งการเปลี่ยนแปลงจากผู้แจ้งเหตุการณ์ ซึ่งสังเกตข้อความ “EventManager is receiving message...” 4 ข้อความ โดยที่ข้อความแรกจะเป็นข้อมูลแจ้งการเพิ่มบริการ และข้อความสุดท้ายจะเป็นข้อมูลแจ้งการเปลี่ยนแปลงชนิดบริการ ซึ่งข้อมูลแจ้งการเปลี่ยนแปลงทั้ง 4 เหตุการณ์จะถูกส่งไปยังผู้รับบริการฝั่งเทรดเดอร์ส่งออกที่ทำการลงทะเบียนขอรับทราบการเปลี่ยนแปลงไว้แล้ว ขณะเดียวกันก็จะทำการแจ้งการเปลี่ยนแปลงไปให้ตัวเรียกกลับนำเข้า ซึ่งเป็นตัวแทนของผู้รับบริการฝั่งเทรดเดอร์นำเข้าอีกด้วย



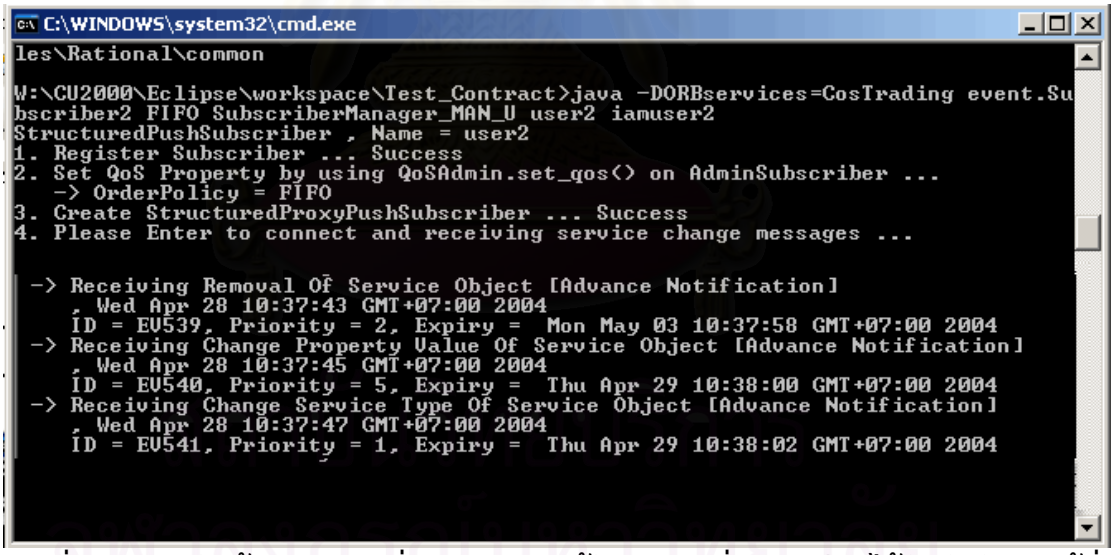
รูปที่ 5.36 การรับข้อมูลแจ้งการเปลี่ยนแปลงบริการของวัตถุตัวเรียกกลับนำเข้า

จากรูปที่ 5.36 ตัวเรียกกลับนำเข้าในฝั่งเทรดเดอร์นำเข้าจะได้รับข้อมูลแจ้งการเปลี่ยนแปลงที่มาจากบริการเอสซีเอ็นฝั่งเทรดเดอร์ส่งออก ซึ่งสังเกตได้จากข้อความ “ImportCallback is receiving message...” 3 เหตุการณ์ คือได้รับข้อมูลที่ 8-10 และทำการส่งต่อไปยังตัวจัดการเหตุการณ์ฝั่งเทรดเดอร์นำเข้าต่อไป



รูปที่ 5.37 การรับข้อมูลแจ้งเหตุการณ์การเปลี่ยนแปลงของตัวจัดการเหตุการณ์ในฝั่งเทรดเดอร์นำเข้า

รูปที่ 5.37 บริการเอสซีเอ็นฝั่งเทรดเดอร์นำเข้าได้รับข้อมูลแจ้งการเปลี่ยนแปลงจากตัวเรียกกลับนำเข้า ซึ่งสังเกตได้จากข้อความ “EventManager is receiving message...” 3 ข้อความ และจะส่งต่อการแจ้งเตือนไปยังผู้รับบริการในฝั่งเทรดเดอร์นำเข้าที่ลงทะเบียนขอรับทราบการเปลี่ยนแปลงไว้



รูปที่ 5.38 การรับข้อมูลการเปลี่ยนแปลงของผู้รับบริการที่ลงทะเบียนไว้กับตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงที่ทำงานร่วมกับเทรดเดอร์นำเข้า

รูปที่ 5.38 ผู้รับบริการฝั่งเทรดเดอร์นำเข้าที่ได้ทำการลงทะเบียนขอทราบการเปลี่ยนแปลงได้รับแจ้งการเปลี่ยนแปลงบริการจากบริการเอสซีเอ็นของฝั่งเทรดเดอร์นำเข้าทั้งหมด 3 เหตุการณ์ตามข้อมูลที่ 8-10

จากผลการทำงานในรูปที่ 5.34 – 5.38 สามารถอธิบายได้ดังนี้

1. จากรูปที่ 5.34 ผู้แจ้งเหตุการณ์ทำการแจ้งการเปลี่ยนแปลงบริการไปยังตัวจัดการเหตุการณ์ในฝั่งเทรดเดอร์ส่งออกดังรูปที่ 5.35 ทั้งหมด 4 เหตุการณ์ (ข้อมูลที่ 7–10) และตัวจัดการเหตุการณ์ในฝั่งเทรดเดอร์ส่งออกก็ทำการแจ้งไปยังตัวเรียกกลับนำเข้าดังรูปที่ 5.36 ทั้งหมด 3 เหตุการณ์ (ข้อมูลที่ 8-10) และตัวเรียกกลับนำเข้าก็ส่งต่อไปให้ตัวจัดการเหตุการณ์ในฝั่งเทรดเดอร์นำเข้าดังรูปที่ 5.37 ทั้งหมด 3 เหตุการณ์เช่นกัน หลังจากนั้นตัวจัดการเหตุการณ์ก็ส่งต่อการแจ้งเตือนไปยังผู้รับบริการในฝั่งเทรดเดอร์นำเข้าดังรูปที่ 5.38 และผู้รับบริการได้รับแจ้งเตือนการเปลี่ยนแปลง 3 เหตุการณ์เช่นกัน ดังนั้นจะเห็นว่าผู้แจ้งเหตุการณ์สามารถแจ้งการเปลี่ยนแปลงไปยังตัวเรียกกลับนำเข้าได้อย่างถูกต้อง และในทำนองเดียวกันตัวเรียกกลับนำเข้าก็แจ้งการเปลี่ยนแปลงบริการผ่านตัวจัดการเหตุการณ์ไปยังผู้รับบริการที่ขอทราบการเปลี่ยนแปลงได้อย่างถูกต้องเช่นเดียวกัน

บทที่ 6

สรุปผลการวิจัยและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงผลสรุปของงานวิจัย ปัญหาและข้อจำกัดต่างๆ ของระบบ รวมทั้งข้อเสนอแนะที่สามารถนำไปปรับปรุงและพัฒนาได้ต่อไปในอนาคต โดยมีรายละเอียดดังต่อไปนี้

6.1 สรุปผลการวิจัย

งานวิทยานิพนธ์นี้ ได้ทำการออกแบบและพัฒนาระบบแจ้งการเปลี่ยนแปลงของบริการระยะไกลผ่านการเชื่อมต่อกันของเทรดเดอร์ไปยังผู้ใช้บริการในระบบกระจายคอร์บา โดยได้ทำการปรับรูปแบบการทำงานในส่วนของการเชื่อมต่อเทรดเดอร์ของคอร์บาที่ยังไม่สนับสนุนการใช้สัญญาข้อตกลง ให้สามารถใช้สัญญาข้อตกลงตามข้อกำหนดของอาร์เอ็มโอดีพี [5] เพื่อประโยชน์สำหรับการควบคุมสิทธิ์ในการใช้งานบริการของผู้รับบริการระยะไกล และสัญญาข้อตกลงดังกล่าวยังถูกใช้ประโยชน์สำหรับการแจ้งเตือนการเปลี่ยนแปลงบริการระยะไกลอีกด้วย โดยผลที่ได้รับคือ โครงสร้างและต้นแบบของส่วนขยายเทรดเดอร์ที่มีการเชื่อมต่อกัน และระบบแจ้งเหตุการณ์เอสซีเอ็น [6] สามารถแจ้งการเปลี่ยนแปลงข้ามระบบได้ โดยอาศัยกลไกในการกรองข้อมูลชนิดบริการและข้อเสนอบริการที่นำเข้าจากบัญชีรายชื่อบริการ เพื่อใช้เป็นข้อมูลเงื่อนไขในการลงทะเบียนขอรับทราบการเปลี่ยนแปลงจากตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก โดยเทรดเดอร์นำเข้าจะทำการสร้างตัวเรียกกลับนำเข้า เพื่อรอรับแจ้งการเปลี่ยนแปลงที่มาจากบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออก เมื่อเกิดการเปลี่ยนแปลงบริการเกิดขึ้นที่เทรดเดอร์ส่งออก ตัวจัดการเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออกจะทำการแจ้งการเปลี่ยนแปลงมายังตัวเรียกกลับนำเข้าของเทรดเดอร์นำเข้า หลังจากนั้นตัวเรียกกลับนำเข้าก็จะทำการส่งต่อการแจ้งเตือนเหตุการณ์การเปลี่ยนแปลงไปยังตัวจัดการเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์นำเข้า เพื่อทำการแจ้งการเปลี่ยนแปลงไปยังผู้รับบริการต่อไป ซึ่งจะเห็นได้ว่าการแจ้งการเปลี่ยนแปลงมีความโปร่งใส เนื่องจากผู้รับบริการไม่จำเป็นต้องลงทะเบียนเพื่อขอรับทราบการเปลี่ยนแปลงบริการจากตัวจัดการผู้ที่ต้องการรับทราบการเปลี่ยนแปลงบริการของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออกโดยตรง แต่ให้ตัวเรียกกลับนำเข้าทำหน้าที่ลงทะเบียนเพื่อขอรับทราบการเปลี่ยนแปลงบริการระยะไกลแทน ซึ่งจะเห็นได้ว่าการใช้กลไกของการกรองข้อมูลชนิดบริการและข้อเสนอบริการจากสัญญาข้อตกลงดังกล่าวส่งผลให้ข้อความแจ้งเตือนการเปลี่ยนแปลงจากบริการเอสซีเอ็นระยะไกลลดลง เมื่อทำการเปรียบเทียบกับการที่ไม่มีการใช้สัญญาข้อตกลง ซึ่งจะทำการส่งข้อความแจ้งเตือนการเปลี่ยนแปลงของบริการ

ทั้งหมดในเทรดเดอร์ส่งออกไปยังฝั่งเทรดเดอร์นำเข้า และทำให้ประสิทธิภาพการทำงานของระบบช้าลงอีกด้วย

จากการทดสอบการทำงานของส่วนขยายเทรดเดอร์และตัวเรียกกลับนำเข้าที่รอรับแจ้งการเปลี่ยนแปลงจากตัวจัดการเหตุการณ์ของบริการเอสซีเอ็นที่ทำงานร่วมกับเทรดเดอร์ส่งออกนั้น พบได้ว่าบริการเอสซีเอ็นระยะไกลสามารถทำงานได้อย่างถูกต้อง

6.2 ปัญหาและข้อจำกัดที่ได้พบจากการวิจัย

1. ถึงแม้ระบบจะมีการควบคุมสิทธิ์ในการให้บริการและใช้วิธีการกรองข้อมูลจากคำร้องขอบริการผ่านทางการกำหนดสัญญาข้อตกลง แต่ก็ยังพบว่าแม้เมื่อไม่มีผู้รับบริการในฝั่งเทรดเดอร์นำเข้าลงทะเบียนขอรับทราบการเปลี่ยนแปลงบริการที่เกิดขึ้นในฝั่งเทรดเดอร์ส่งออก ตัวเรียกกลับนำเข้าของฝั่งเทรดเดอร์นำเข้าก็จะยังได้รับแจ้งเหตุการณ์การเปลี่ยนแปลงที่เกิดขึ้นในฝั่งเทรดเดอร์ส่งออกอยู่ดี จึงอาจเป็นการส่งข้อความแจ้งเตือนที่ไม่จำเป็น

2. ส่วนเพิ่มขยายบริการเทรดเดอร์ที่พัฒนาขึ้นยังไม่รองรับการให้สิทธิ์ผู้ดูแลเทรดเดอร์นำเข้าในการส่งออกแบบลูกโซ่ ซึ่งก็คือการส่งออกบริการจากบัญชีรายชื่อบริการที่ได้นำเข้าไปให้เทรดเดอร์ตัวอื่นอีกต่อหนึ่ง

6.3 ข้อเสนอแนะ

1. ควรปรับปรุงให้ระบบแจ้งการเปลี่ยนแปลงระยะไกลมีกลไกที่มีประสิทธิภาพมากขึ้นกว่านี้ เนื่องจากตัวเรียกกลับนำเข้ายังได้รับแจ้งเตือนเหตุการณ์การเปลี่ยนแปลงจากบริการเอสซีเอ็นระยะไกลอยู่ ถึงแม้จะไม่มีผู้รับบริการทางฝั่งเทรดเดอร์นำเข้าที่ลงทะเบียนขอรับทราบการเปลี่ยนแปลงในฝั่งเทรดเดอร์ส่งออกก็ตาม

2. ควรปรับปรุงความสามารถในการส่งออกบริการ โดยให้ผู้ดูแลเทรดเดอร์นำเข้าสามารถส่งออกบริการไปยังเทรดเดอร์ตัวอื่นได้ในลักษณะแบบลูกโซ่ตามข้อกำหนดมาตรฐาน ทำให้ไม่ถูกจำกัดด้วยการส่งออกบริการระหว่างเทรดเดอร์เพียง 2 ตัว ซึ่งกรณีดังกล่าวข้างต้น เทรดเดอร์นำเข้าจะต้องได้รับอนุญาตจากเทรดเดอร์ส่งออกเสียก่อน จึงจะสามารถส่งออกบริการที่นำเข้าไปยังเทรดเดอร์ตัวอื่นได้

รายการอ้างอิง

[1] Object Management Group. The Common Object Request Broker: Architecture and Specification Revision 3.0.2., 2002.

[2] Object Management Group. CORBAservices : Common Object Services Specification., 2002.

[3] Object Management Group. Trading Object Service Specification Revision 1.0, 2000.

[4] Object Management Group. Notification Service Specification Revision 1.0.1, 2002.

[5] Bearman, M. and Raymond, K. Federating Traders: an ODP Adventure. Proceedings of the IFIP TC6/WG6.4 International Workshop on Open Distributed Processing, Berlin, Germany, (October 1991)

[6] Suriyentrakorn, P., Design and Development of A Change Notification System of Distributed Service, Master Thesis, Dept. of Computer Engineering, Chulalongkorn University, Bangkok, Thailand, 2001.

[7] Senivongse, T. and Suriyentrakorn, P. A CORBA-Based Architecture for Service Change Notification. The 5th International Enterprise Distributed Object Conference (EDOC 2001), Seattle, Washington, USA, September 2001

[8] Ercan, G., Aghoutane, N. , Tschammer, V. and Herzog, H. Federation Management for the OMG-Trader. Proceeding First International Enterprise Distributed Object Computing Workshop (EDOC 1997), Marriott Resort, Gold Coast, Australia, October 1997

[9] Inprise VisiBroker for Java, Available from: http://www.inprise.com

[10] JacORB– a free Java ORB, Available from: http://jacorb.inf.fu-berlin.de

[11] DSTC dCon, Available from: http://www.dstc.com

[12] MySQL, Available from: http://www.mysql.com

ภาคผนวก

ภาคผนวก ก
ลำดับการรันงานของระบบและการเรียกใช้ส่วนต่าง ๆ ของบริการแจ้งเหตุการณ์ระยะไกล

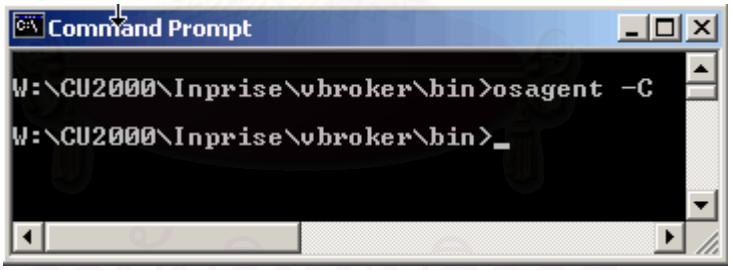
ลำดับการเริ่มต้นทำงานของบริการแจ้งเหตุการณ์ระยะไกลนั้น จะต้องทำการรันโปรแกรมบริการเอสซีเอ็นและบริการอื่นๆ ที่เกี่ยวข้องบนเซิร์ฟเวอร์อย่างน้อย 2 เครื่อง โดยให้เครื่องแรกเป็นบริการของเทรดเดอร์ส่งออก (ชื่อ LIVERPOOL) และเครื่องที่สองเป็นบริการของเทรดเดอร์นำเข้า (ชื่อ MAN_U) ซึ่งแสดงดังนี้

1. เริ่มโปรแกรม osagent ของวิสิโบรกเกอร์ ด้วยคำสั่ง

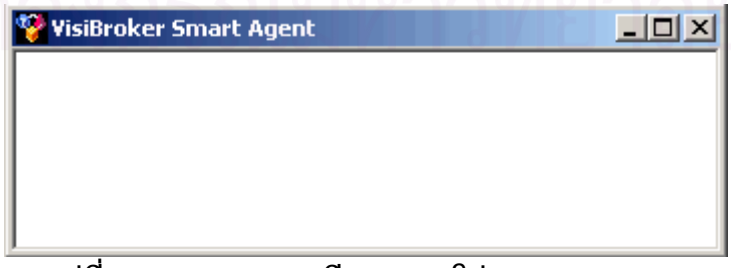
```
osagent [-options]
```

คำสั่ง osagent สามารถระบุพอร์ตด้วยการระบุค่าในตัวแปรสภาวะ (Environment Variable) OSAGENT_PORT หรือทางอาร์กิวเมนต์ของคำสั่ง เช่น osagent –p=14001 ได้ ทั้งนี้เพื่อให้สามารถแยกการทำงานของระบบต่างๆ ออกเป็นโดเมนได้ โดยหากไม่ระบุโปรแกรมจะใช้ค่าโดยปริยาย คือ 14000 (รูปที่ ก1 และ ก2)

คำสั่งนี้จะทำการรันทั้งบนเครื่อง LIVERPOOL และ MAN_U



รูปที่ ก1 จอภาพแสดงการเริ่มโปรแกรม osagent



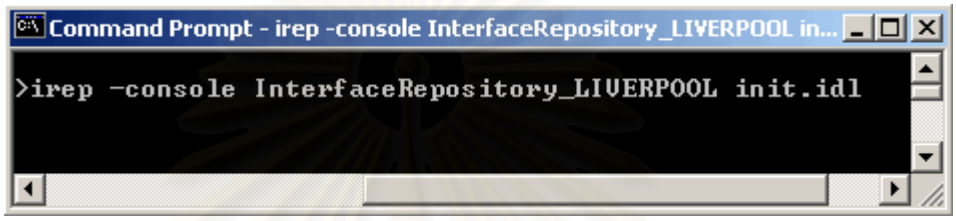
รูปที่ ก2 จอภาพแสดงดีมอนของโปรแกรม osagent

2. เริ่มโปรแกรมคลังส่วนต่อประสาน ด้วยคำสั่ง

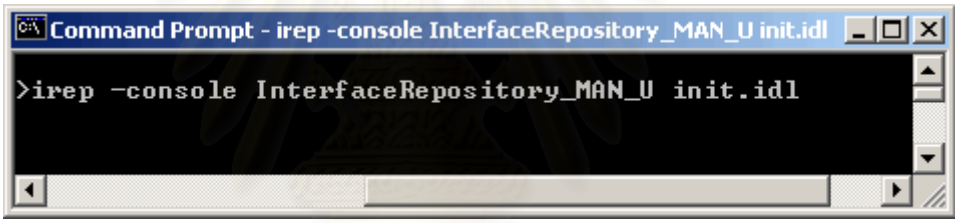
```
irep [-options] <irep name> [<idl storage file name>]
```

ผู้ใช้สามารถใช้ทางเลือก –console เพื่อให้โปรแกรมทำงานในโหมดอักษร (Text Mode) ได้ และสามารถระบุชื่อคลังส่วนต่อประสานเพื่อเป็นชื่ออ้างอิงในการเรียกใช้ และสามารถกำหนดชื่อไฟล์ที่เก็บข้อมูลส่วนต่อประสานได้ (รูปที่ ก3 และ ก4)

คำสั่งนี้จะทำการรันทั้งบนเครื่อง LIVERPOOL และ MAN_U



รูปที่ ก3 จอภาพแสดงการเริ่มโปรแกรมคลังส่วนต่อประสานชื่อ LIVERPOOL



รูปที่ ก4 จอภาพแสดงการเริ่มโปรแกรมคลังส่วนต่อประสานชื่อ MAN_U

3. เริ่มบริการแจ้งเหตุการณ์ของ dCon ด้วยคำสั่ง

```
cosnitification start [property file]
```

ผู้ใช้สามารถระบุไฟล์คุณสมบัติเพื่อกำหนดการทำงานเริ่มต้นให้กับบริการแจ้งเหตุการณ์ได้ โดยหากไม่ระบุบริการก็จะทำงานด้วยค่าโดยปริยายที่บริการกำหนด (รูปที่ ก5 และ ก6)

คำสั่งนี้จะทำการรันทั้งบนเครื่อง LIVERPOOL และ MAN_U

รูปที่ ก5 จอภาพแสดงการเริ่มโปรแกรมบริการแจ้งเหตุการณ์ชื่อ LIVERPOOL

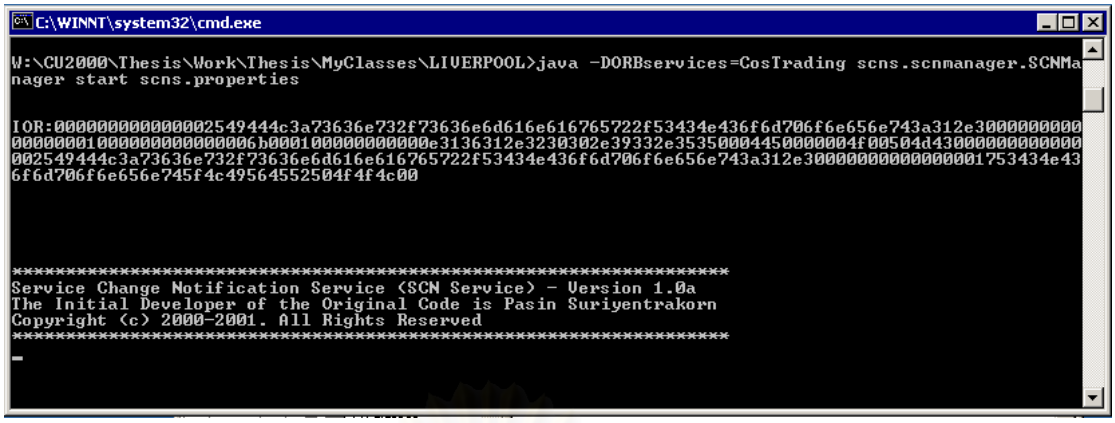
รูปที่ ก6 จอภาพแสดงการเริ่มโปรแกรมบริการแจ้งเหตุการณ์ชื่อ MAN_U

4. เริ่มบริการเอสซีเอ็น ด้วยคำสั่ง

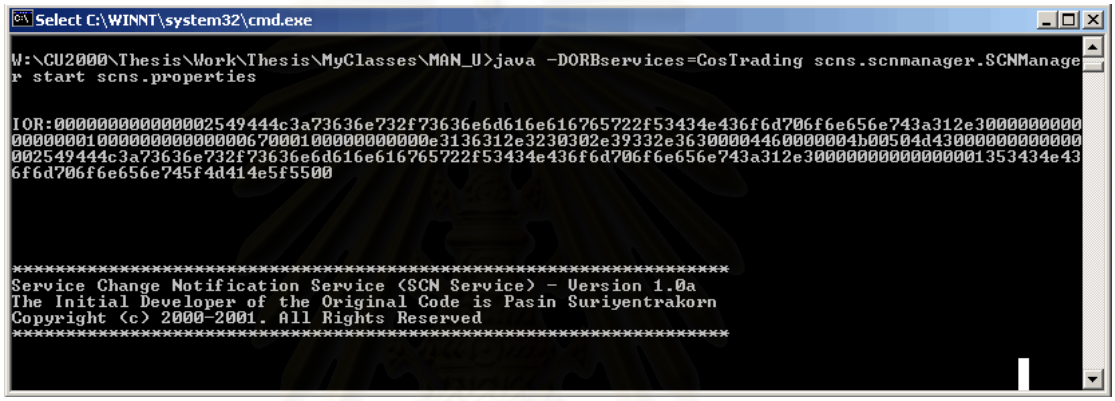
java -DORBservices=CosTrading scns.scnmanager.SCNManager [reset start stop] [propertie file]

อาร์กิวเมนต์ของการเริ่มบริการเอสซีเอ็นมี ได้แก่ -DORBservices=CosTrading เป็นการระบุว่าต้องการใช้งานบริการเทรดเดอร์ โดยที่scns.scnmanager.SCNManager เป็นไฟล์เริ่มการทำงานของบริการเอสซีเอ็น ส่วน[reset start stop] เป็นทางเลือกให้กับผู้ใช้ โดย ตัวเลือก reset เป็นการสั่งเริ่มต้นบริการโดยมีการเริ่มต้นค่าต่างๆ ใหม่ทั้งหมด ตัวเลือก start เป็นการสั่งเริ่มต้นบริการโดยไม่มีการเริ่มต้นค่าใหม่ และตัวเลือก stop เป็นการสั่งหยุดการทำงานของบริการเอสซีเอ็น และผู้ใช้ยังสามารถระบุไฟล์คุณสมบัติเพื่อกำหนดการทำงานเริ่มต้นให้กับบริการเอสซีเอ็นได้ โดยหากไม่ระบุบริการก็จะทำงานด้วยค่าโดยปริยายที่บริการกำหนด (รูปที่ ก7 และ ก8)

คำสั่งนี้จะทำการรันทั้งบนเครื่อง LIVERPOOL และ MAN_U



รูปที่ ก7 จอภาพแสดงการเริ่มโปรแกรมบริการเอสซีเอ็นชื่อ LIVERPOOL



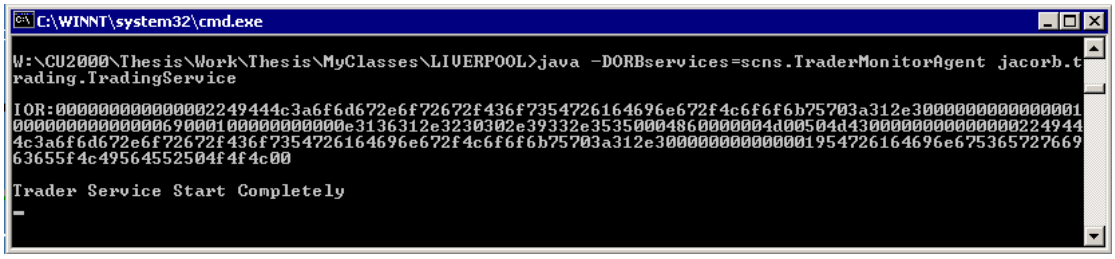
รูปที่ ก8 จอภาพแสดงการเริ่มโปรแกรมบริการเอสซีเอ็นชื่อ MAN_U

5. เริ่มบริการเทรดเดอร์ ด้วยคำสั่ง

```
java -DORBservices=scns.TraderMonitorAgent
     jacorb.trading.TradingService
```

อาร์กิวเมนต์ของการเริ่มบริการเทรดเดอร์ ได้แก่ -DORBservices=scns.Trader MonitorAgent เป็นการระบุการใช้งานบริการเทรดเดอร์ร่วมกับตัวตรวจบริการเทรดเดอร์ ด้วยการระบุชื่อเพกเกจของตัวตรวจบริการเทรดเดอร์ ส่วน jacorb.trading. TradingService เป็นไฟล์เริ่มการทำงานของบริการเทรดเดอร์ (รูปที่ ก9 และ ก10)

คำสั่งนี้จะทำการรันทั้งบนเครื่อง LIVERPOOL และ MAN_U



รูปที่ ก9 จอภาพแสดงการเริ่มโปรแกรมบริการเทรดเดอร์ชื่อ LIVERPOOL

รูปที่ ก10 จอภาพแสดงการเริ่มโปรแกรมบริการเทรดเดอร์ชื่อ MAN_U

ภาคผนวก ข
ผลงานตีพิมพ์

1. The 3rd International Symposium on Information and Communication Technology (ISICT'04) June 16 - 18, 2004, Las Vegas, USA ในบทความเรื่อง Cross-Domain Service Change Notification via Trading Federation โดยผู้แต่ง คือ Banjong Huongrat and Twittie Senivongsen

Cross-Domain Service Change Notification via Trader Federation[1]

Banjong Huongrat and Twittie Senivongse
Department of Computer Engineering, Chulalongkorn University
Phyathai Road, Pathumwan, Bangkok 10330 Thailand
Tel: +66 2 2186996 Fax: +66 2 2186955
Email: banjong.h@student.chula.ac.th, twittie.s@chula.ac.th

Abstract

The nature of distribution and autonomy in an open environment makes it difficult to maintain a consistent view of service provision among distributed groups of clients. In the case that services are changed, their clients should be informed properly in order to reduce unexpected outcome of using the changed services. This paper presents an extension to our previous CORBA-based change notification system that can notify service changes of several kinds to subscribing clients in a single trader domain. Now the notification is extended for remote clients in a trader federation, meaning that clients who obtain remote services via federated traders can now be informed of remote service changes. The concept of service contracts in RM-ODP traders is adopted since the contract can be seen as a subscription of the importing trader to be notified of changes in those remote services listed in the contract.

Key Words: Service changes, CORBA notification service, federation contract

1. Introduction

Standard distributed architectures normally define a basic directory service as a place for service providers to export (i.e. publish) service advertisements and for clients to import (i.e. request for) services that fit their requirements. For the Common Object Request Broker Architecture (CORBA), such a directory is called Trading Object Service or Trader [1]. Service advertisements that are published with a trader will list down the characteristics of the services and how to access service objects. When there are changes in such descriptions or in the service objects themselves, there could be impacts on the operations of the clients if they are not properly informed. In a trader federation in which multiple traders are connected to cooperate, a link will be set up between each pair of connecting traders. This is to allow one trader to forward local clients' requests to other remote traders in order to enlarge search space and increase the chance to discover the required services. Such forwarding of requests will be transparent, meaning that the clients will be unaware that the requests are sent further and results gathered for them by the initiating local trader that they contact. In this scenario, clients who are affected by a service change may be either local clients within a single trader domain or remote clients via federation. In the previous work [2], we propose a CORBA-based notification architecture that supports notification of service changes to interested clients in a particular trader domain. Clients will subscribe their interest with the system, and when service providers publish any service changes or when the trader can detect some changes, the system will generate notification messages to subscribing clients. This paper extends such architecture to support trader federation so that cross-domain notification is possible, allowing service changes in one trader domain to be notified to remote clients in another trader domain. We also adopt the idea in [3] that extends CORBA trader federation

[1] This work is supported by Chulalongkorn University-Industry Linkage Research Grant and Thailand-Japan Technology Transfer Project

with the concept of federation by service contracts, which is defined in the Reference Model of Open Distributed Processing (RM-ODP) [4]. A service contract is a contract for cross-domain service access, containing a list of services that a particular trader imports from or exports to another trader. Since only those listed services can be discovered and accessed via a particular remote trader for which the contract has been established, a federation contract can be seen as a specification of service change interests that an importing trader subscribes with the exporting trader.

In the rest of this paper, the previous change notification system is revisited in Section 2. Section 3 describes the extension to support cross-domain service change notification service via service contract and gives an example use of this service. The paper is concluded in Section 4.

2. Revisiting Service Change Notification Service

For the single-domain service change notification service or SCNS [2], service change events are classified according to the model of service type and service offer of the trader. A service type description specifies service type name, base service type from which the type inherits the interface signature and properties, Interface Definition Language (IDL) file in which operation signatures are defined, and property names and their types. A service offer description specifies the characteristics of a service instance according to its type. Figure 1 (a) shows an example of a service type Bank. Under this model, clients need to know the type definition of the required service when importing with the trader, e.g. an import query might be to find service offers of type Bank whose Interest_Rate is more than 0.05. Or the clients can save the references to the service objects that are returned by the trader from previous import for future invocation. This places a requirement to notify those clients when type and offer descriptions or service objects themselves are changed. The classification of service change events is as follows:

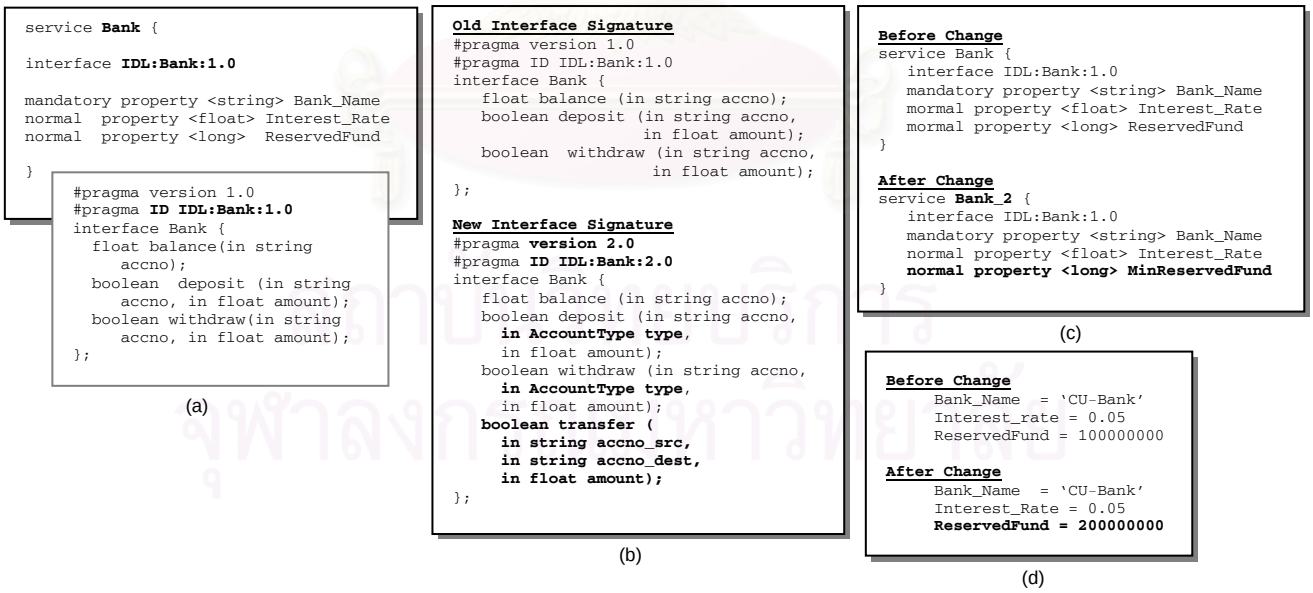


Figure 1. Example of service changes (a) Service type and interface before change (b) Change of interface signature (c) Change of property definition (d) Change of property value

- Change of service type or offer description can either be
  - Change of interface signature (e.g. Figure 1(b))
  - Change of property definition (e.g. Figure 1(c))

  - Change of base service type. This has a similar effect to change of inherited interface signature and property definition.
  - Change of service type name. This may refer to change to a new service type and reflect change in other parts of the type definition.
  - Change of property value (e.g. Figure 1(d))
- Change to service instance can either be
  - Removal of object. This results in the removal of the corresponding service offer description.
  - Instantiation of new object. This results in the addition of the corresponding service offer description.

Figure 2 shows the architecture of the SCNS. The Service Change Notification Manager (SCN Manager) is the core component of the SCNS. It receives, via Subscriber Manager, the subscription information from subscribers who are those clients interested in the changes of particular service types or service instances. Change events are either published in advance by service providers or evolvers via Publisher Manager, or detected automatically by Trader Monitor and Service Change Supporting (SCS) Module. Trader Monitor and SCS Module work together to intercept the requests, which are sent to the trader by service evolvers, to change service type/offer definitions and remove/instantiate service objects. Event Manager will transform change events into an appropriate format for CORBA Notification Service [5] which will then generate change notifications to the subscribers. PS or Proxy Subscribers, created by Subscriber Manager at the time the subscribers subscribe their interest, will obtain change notification messages and take care of their representation formats, i.e. structured event object or XML document format. Notifications are delivered, either by push or pull style and by predefined notification policies and QoS, to the corresponding subscribers via their subscribed callback objects. For more information, see [2].

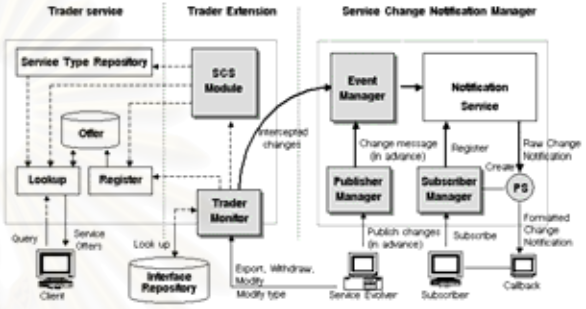


Figure 2. Architecture of single-domain SCNS

3. Cross-Domain Service Change Notification

We extend SCNS to support cross-domain service change notification via trader federation. Since federation for CORBA traders only means a setup of links to identify the remote traders that a particular trader can forward import requests, we enhance it with RM-ODP contract management so that constraints on services that will be imported or exported can be specified. For example, an exporting trader may allow only some, not all, of its services to be imported, discovered, and used by clients of another importing trader. When a client trades for a service offer with the trader that imports under a contract, the (lookup component of) the trader can choose to forward the request only to the (lookup component of) those traders from which it imports such a service type or offer. The contract is set up by an exporting trader distributing the list of its services (i.e. catalogue) to an importing trader, or by an importing trader requesting for the catalogue. The importing trader will propose an import contract containing the list of the services to be imported. Approving of the import contract, the exporting trader will create a corresponding export contract. Consistency between the two counterpart contracts must be maintained.

The exporting trader may export services at service type or service offer levels. Exporting a service type Bank at service type level is similar to exporting all published offers of type Bank. On the other hand, the exporting trader may export some offers of a service type only by specifying object references or constraints that will identify specific offers, e.g. exporting an offer of type Bank whose Bank_Name is CU-Bank only.

3.1 Contract-Based Service Change Notification Rules

With the contracts, service change notification will now follow these rules:

1. Assume a service Bank is changed by any change category in Section 2, the change will be notified to local subscribing clients within the domain by the SCNS in a usual way. Moreover more checking is done below:
   - 1.1 If Bank is not exported, no more notification is needed.
   - 1.2 If Bank is exported to an importing trader at service type level, the change will be notified to importing traders.
   - 1.3 If Bank is exported to an importing trader at service offer level,
     - 1.3.1 If the change is on service type definition of Bank, the change will be notified to the importing trader since the imported Bank offer will be affected by the changed definition.
     - 1.3.2 If the change is on a service offer of Bank, the change will be notified to the importing trader only when the changed offer is the imported offer.
2. When no service is changed but the contract is modified,
   - 2.1 If a service type description or service offer description is added to, removed from, or modified in the contract, the importing trader will be notified since the action reflects change to the services available in the importing trader domain.
   - 2.2 If the contract is destroyed, this results in the removal of all imported service types and offers, so the importing trader will be notified.
   - 2.3 If a new contract is created, this reflects new imported service types and offers, so the importing trader will be notified.
3. When a service is changed and that results in the change of the contract, the importing trader will be notified as in case 1 and 2 above.

3.2 Architecture for Cross-Domain Service Change Notification

Figure 3 depicts the collaboration between two trading domains for cross-domain service change notification. The left part of the figure is the importing domain and the right part is the exporting domain. Both domains have the same architecture, which consists of the standard CORBA trader components, single-domain SCNS components, and federation management extension (shown in shaded box). The Federation Admin module of the exporting trader will distribute a service catalogue from its Catalogue Repository to the Federation Contract Module of the importing trader, and finally the import contract and export contract will be established and stored in the corresponding Import/Export Contract Repository (1). The importing trader subscribes with the Federation Admin of the exporting trader its interest for change notification (2). The subscription will be according to the import contract and specifies the Import Callback object that will listen to incoming notifications. The Federation Admin will forward the subscription to the Subscriber Manager of the local SCNS (3), resulting in the Proxy Subscriber object of the importing trader to be created. When the import contract is modified or destroyed, the subscription information will be changed accordingly.

The creation/modification/removal of the import contract also reflects the change in the set of available services within the importing trader domain, and so the event will be reported to the Event Manager of the SCNS (4) so that notification messages will be sent to interested subscribers. Clients in the importing domain can subscribe their interest in remote services, as if they were local services, via the local Subscriber Manager (5). And at exporting side, when service changes are published or automatically detected (6), the changes will be notified to the Proxy Subscriber of the subscribing importing trader according to the rules in Section 3.1. The notification will be forwarded to the associated Import Callback in the importing domain (7) and then to the Event Manager (8). The messages will be further notified to the Proxy Subscriber and Callback object of individual subscribers respectively (9).

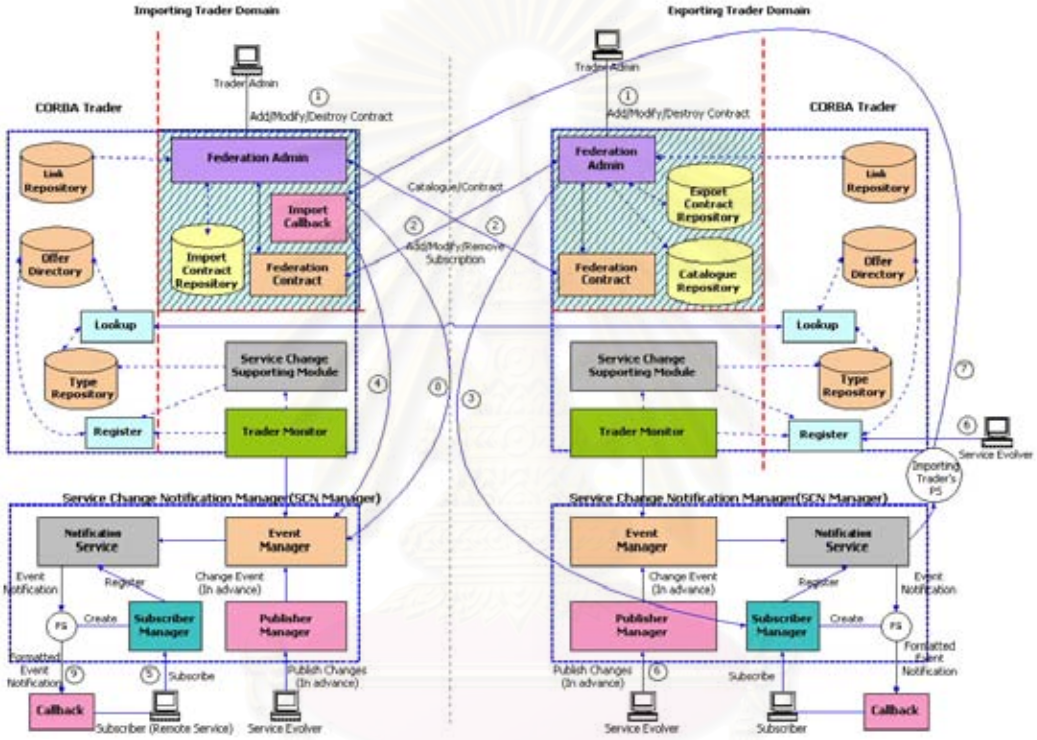
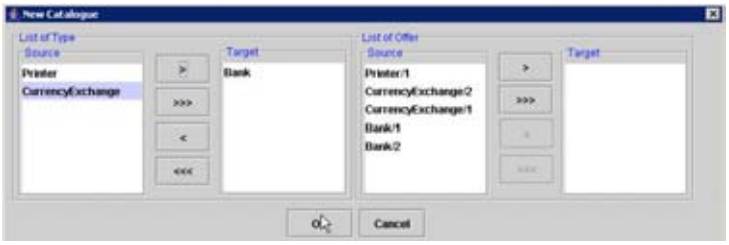


Figure 3. Architecture for cross-domain service change notification

The prototype architecture is developed on VisiBroker v.3.4 for Java with a Trading service from JacORB v.1.3.30 for Java. We use DSTC's dCon v.3.0 for Java for the Notification Service component of the SCNS.

Let us walk through an example. Suppose that an exporting trader, selecting from its list of registered service types and offers, creates a service catalogue in Figure 4 (a) and distributes it to other federated traders. In this example, only the service type named Bank is exported at service type level. An importing trader proposes an import contract to import this Bank service type (Figure 4 (b)), and eventually the import and export contracts are agreed. Later via federation, a client of the importing trader transparently imports and uses the service of an offer of type Bank, and becomes interested in any kinds of changes to this service. Thus, the client subscribes with the local SCNS (Figure 4 (c)). When a new instance of service type Bank, named BBL, is instantiated later on in the exporting domain, the client is notified of this remote change, i.e. the notification informs that a new Bank instance is now available for the clients of the importing domain (Figure 4(d)).



(a)

```
Kind Of Event = Change of Service
Service Identification type= TraderConstraint
ServiceTypeName =Bank
Property Name=""    Value=""
…
```

(c)

(b)

```
<?xml version="1.0" ?>
<!DOCTYPE ServiceChangeNotification (View Source for full doctype…)>
<ServiceChangeNotification>
    <NotificationMessageHeader>
        <MessageID>EV101</MessageID>
        <MessageType value="Instantiation_Of_Service"/>
        <TimeIndicator value="Immediate_Notification"/>
        <Priority>2</Priority>
        <ExpiryTime>Fri Mar 05 08:00:00 GMT+07:00 2004</ExpiryTime>
    </NotificationMessageHeader>
    <NotificationMessageBody>
        <InstantiationOfService>
            <ServiceDescription type="CHANGED_SERVICE">
                <ServiceTypeDescription>
                    <TraderServiceType Id="IDL:Bank:1.0" Name="Bank" Masked="NO">
                        <Property Name="Bank_Name" Type="string" Mode="MANDATORY"/>
                        …
                    </TraderServiceType>
                    <Interface Id="IDL:Bank:1.0" Name="Bank" Version="1.0">
                        <Operation Id="IDL:Bank/balance:1.0" Name="balance" Version="1.0"
                            Type="float" Mode="NORMAL"/>
                        …
                    </Interface>
                </ServiceTypeDescription>
                <Property Name="Bank_Name" Value="BBL"/>
                <Property Name="Interest_Rate" Value="0.04"/>
                …
                <ObjectReference>IOR:000000000000000154944c3a…</ObjectReference>
            </ServiceDescription>
            <Objective>…</Objective>
            <Time>Wed Feb 25 08:00:04 GMT+07:00 2004</Time>
        </InstantiationOfService>
    </NotificationMessageBody>
    <Additional>…</Additional>
</ServiceChangeNotification>
```

(d)

Figure 4. Example (a) Catalogue (b) Import contract (c) Subscription info (d) Notification msg

4. Conclusion

This paper shows an extension to a service change notification service to allow notifications across federated trading domains. The architecture adds to standard CORBA trader federation a portion of contract management by which the change notification service can utilise service information in the contract to arrange change notification appropriately. We plan to test the performance of this architecture in various loads of changes, delivery styles, and notification policies. The architecture can also be enhanced to support notification of changes in those services that are exported in chain.

References

[1] OMG, "Trading Object Service v1.0: formal/00-06-27", June 2000.

[2] Senivongse, T., and Suriyentrakorn, P., "A CORBA-Based Architecture for Service Change Notification", Proceedings of 5th IEEE International Enterprise Distributed Object Conference (EDOC 2001), Seattle, Washington, USA, September 2001.

[3] Ercan, G. et al., "Federation Management for the OMG-Trader", Proceedings of 1st International Enterprise Distributed Object Computing Workshop (EDOC'97), Gold Coast, Australia, October 1997.

[4] Bearman, M., and Raymond, K., "Federating Traders: an ODP adventure", Proceedings of IFIP TC6/WG6.4 International Workshop on Open Distributed Processing", Berlin, Germany, October 1991.

[5] OMG, "Notification Service v1.0.1: formal/02-08-04", August 2002.

ประวัติผู้เขียนวิทยานิพนธ์

นายบรรจง ห่วงรัตน์ เกิดเมื่อวันที่ 12 พฤษภาคม พ.ศ. 2514 สำเร็จการศึกษาหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ จากคณะวิทยาศาสตร์ มหาวิทยาลัยรามคำแหง เมื่อปีการศึกษา 2535 ต่อมาในเดือนธันวาคม ปี พ.ศ.2535 ได้เข้าทำงานกับบริษัท Norsk Data Thai Ltd. ในตำแหน่ง Programmer Analyst หลังจากนั้นได้เข้าทำงานกับบริษัททรู จำกัด (มหาชน) (เดิมชื่อบริษัทเทเลคอมเอเชีย) ในตำแหน่ง Senior System Analyst ในเดือนมกราคม ปี พ.ศ. 2538 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2543