การประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิซ

นายจตุรพัชร์ พัฒนทรงศิวิไล

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2556

APPLYING SECURITY PATTERNS AND ASPECT-ORIENTED PROGRAMMING TO WEB SERVICES

Mr. Jaturapat Patanasongsivilai

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Software Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2013

หัวข้อวิทยานิพนธ์	การประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิซ
โดย	นายจตุรพัชร์ พัฒนทรงศิวิไล
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.บัณฑิต เอื้ออาภรณ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(รองศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา)

.....กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร.มฑุปายาส ทองมาก)

จตุรพัชร์ พัฒนทรงศิวิไล : การประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิซ. (APPLYING SECURITY PATTERNS AND ASPECT-ORIENTED PROGRAMMING TO WEB SERVICES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : รศ.ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา, 86 หน้า.

การพัฒนาระบบเว็บเซอร์วิซควรคำนึงถึงความต้องการด้านความมั่นคง เช่น การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง เป็นต้น เมื่อเขียนโค้ดความมั่นคงลงไปในเว็บเซอร์วิซจะเกิดปัญหาคือ ผู้พัฒนาต่างคนต่างเขียนทำให้ไม่มีแบบแผน และโค้ดความมั่นคงจะยุ่งเหยิงและกระจัดกระจายไปทั่ว ทำให้บำรุงรักษายาก และมีสภาพมอดูลาร์ที่ต่ำ งานวิจัยนี้นำเสนอแบบรูปความมั่นคงที่ได้จากการผสมผสานแบบรูปความมั่นคงที่มีอยู่ในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง เพื่อมาประยุกต์ใช้ในการออกแบบฝั่งเซอร์วิซ ร่วมกับการประยุกต์ใช้การโปรแกรมเชิงแง่มุมในฝั่งไคลเอนต์ เพื่อศึกษาผลกระทบจากการประยุกต์ใช้เทคนิคทั้งสองต่อสภาพมอดูลาร์ของระบบเว็บเซอร์วิซ งานวิจัยเสนอการทดลองและวัดสภาพมอดูลาร์ของระบบเว็บเซอร์วิซของบริษัทโทรคมนาคมแห่งหนึ่งในประเทศไทย โดยวัดทั้งก่อนและหลังการนำแบบรูปและวิธีเชิงแง่มุมไปใช้งาน จากผลการทดลองพบว่าแบบรูปความมั่นคงที่เสนอและการโปรแกรมเชิงแง่มุมช่วยเพิ่มสภาพมอดูลาร์ให้กับฝั่งเซอร์วิซและฝั่งไคลเอนต์ได้

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2556

5470914921 : MAJOR SOFTWARE ENGINEERING
KEYWORDS : SECURITY PATTERN / ASPECT-ORIENTED PROGRAMMING / MODULARITY / WEB SERVICES

JATURAPAT PATANASONGSIVILAI : APPLYING SECURITY PATTERNS AND ASPECT-ORIENTED PROGRAMMING TO WEB SERVICES. ADVISOR : ASSOC. PROF. TWITTIE SENIVONGSE, Ph.D., 86 pp.

Security requirements such as authentication, authorization, and security session should be considered during development of Web service systems. Developing security code for Web services can be problematic since there may be several developers writing similar code separately. This makes security code tangled and scattered, reducing maintainability and modularity of the system. This research proposes the integrated security patterns based on a number of authentication, authorization and security session patterns from the literature, and apply them in the design of the service side of the system. In addition, Aspect-Oriented Programming (AOP) is applied to the client side. The research presents an experiment to study the impact of the integrated security patterns and AOP on modularity of the system. We measure modularity of a Web services system of a telecommunication company in Thailand before and after applying the patterns and AOP. The results show that both techniques can together improve modularity on service and client sides.

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

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความเมตตาและความช่วยเหลืออย่างยิ่งจาก รศ. ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา อาจารย์ที่ปรึกษา ที่เสียสละเวลาอันมีค่าที่ให้คำปรึกษาและคำแนะนำต่อการทำวิทยานิพนธ์ ตลอดจนความเอาใจใส่และตรวจงานทั้งในเวลาและนอกเวลาด้วยความเสียสละและอดทนอย่างยิ่ง จนได้งานที่มีคุณภาพ

ขอกราบขอบพระคุณ รศ. ดร.พรศิริ หมื่นไชยศรี ประธานกรรมการสอบวิทยานิพนธ์ และ ผศ. ดร.มฑุปายาส ทองมาก กรรมการสอบวิทยานิพนธ์ ที่ได้กรุณาสละเวลาให้คำแนะนำสำหรับโครงร่างวิทยานิพนธ์ และวิทยานิพนธ์ฉบับนี้ ทำให้งานมีคุณภาพอย่างยิ่ง

ขอขอบพระคุณคณาจารย์ในภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย ทุกท่าน ที่ได้ประสิทธิ์ประสาทวิชาความรู้ให้ข้าพเจ้า ตลอดระยะเวลาศึกษาในระดับปริญญาโท

ขอขอบคุณรุ่นพี่ รุ่นน้อง เพื่อนๆ พี่ๆ ธุรการ และเพื่อนร่วมงานในบริษัททุกท่าน ที่คอยห่วงใยและให้ความช่วยเหลือในทุกๆ ด้านจนข้าพเจ้าทำวิทยานิพนธ์สำเร็จลุล่วง

สุดท้าย ขอบพระคุณบิดา มารดา พี่สาว และครอบครัวอันเป็นที่รัก ที่คอยเป็นกำลังใจและให้การสนับสนุนความช่วยเหลือในการศึกษาและทำวิทยานิพนธ์ในครั้งนี้

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

หน้า

หน้า

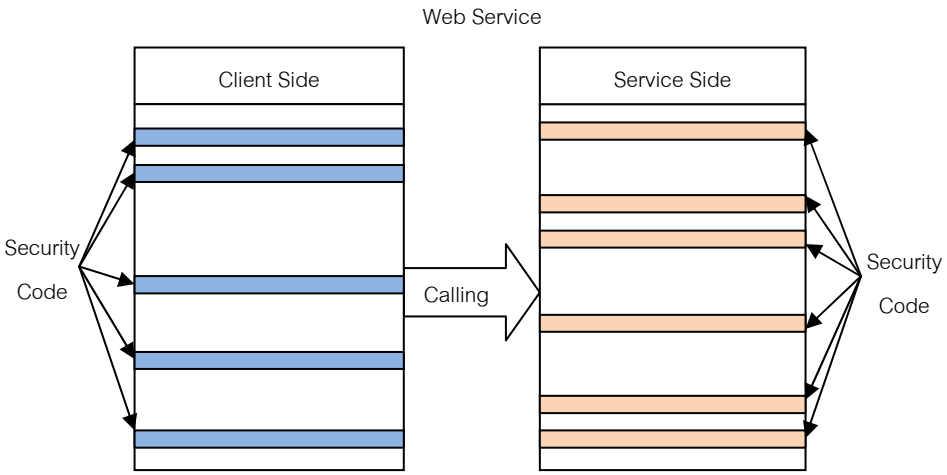
บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันเว็บเซอร์วิซ (Web Services) ได้รับความนิยมและถูกนำไปใช้ในหลาย ๆ องค์กร โดยดับเบิลยูทรีซี (W3C) ได้นิยามเว็บเซอร์วิซว่าเป็น *"ระบบซอฟต์แวร์ที่ถูกออกแบบมาสนับสนุนการทำงานร่วมกันระหว่างเครื่องคอมพิวเตอร์ ซึ่งโต้ตอบกันผ่านทางเครือข่าย เว็บเซอร์วิซจะมีส่วนต่อประสาน (Interface) ซึ่งถูกอธิบายไว้ในรูปแบบที่เครื่องคอมพิวเตอร์สามารถประมวลผลได้ (เป็นภาษาเฉพาะคือวิสเดิล (WSDL)) ระบบอื่น ๆ สามารถทำการโต้ตอบกับเว็บเซอร์วิซตามคำอธิบายที่ได้กำหนดในวิสเดิลด้วยการส่งข้อความโซป (SOAP) ผ่านทางโพรโทคอล เอชทีทีพี (HTTP) พร้อมกับชุดข้อมูลเอกซ์เอ็มแอล (XML) ที่ใช้ร่วมกับมาตรฐานเว็บอื่น ๆ ที่เกี่ยวข้อง"* [1] ตัวอย่างการใช้งาน เช่น เว็บเซอร์วิซจองตั๋วเครื่องบิน เว็บเซอร์วิซจองที่พักในโรงแรม และเว็บเซอร์วิซจองตั๋วภาพยนตร์ เป็นต้น

การนำเว็บเซอร์วิซไปใช้งานดังตัวอย่างข้างต้น ควรคำนึงถึงความมั่นคง (Security) ซึ่งเป็นความต้องการที่ไม่ใช่หน้าที่หลัก (Non-functional Requirement) ที่สำคัญยิ่งในการพัฒนาซอฟต์แวร์ เช่น การพิสูจน์ตัวจริง (Authentication) การพิสูจน์สิทธิ์ (Authorization) และเซสชันด้านความมั่นคง (Security Session) เป็นต้น คุณสมบัติดังกล่าวถือเป็นหนึ่งในความต้องการหลักที่จำเป็น และพบเห็นได้ทั่วไปในระบบซอฟต์แวร์ ตัวอย่างเช่น การระบุชื่อผู้ใช้และรหัสผ่านก่อนใช้งานระบบครั้งแรก เมื่อผู้ใช้ผ่านการพิสูจน์ตัวจริง จะมีสิทธิ์ใช้ทรัพยากรใด ๆ ในระบบได้หรือไม่ขึ้นกับสิทธิ์ที่ได้รับ เป็นต้น

เมื่อเขียนโค้ดความมั่นคงลงไปในเว็บเซอร์วิซจะเกิดปัญหาคือ ผู้พัฒนาต่างคนต่างเขียนทำให้ไม่มีแบบแผน และจากภาพที่ 1.1 จะพบอีกปัญหาคือ โค้ดความมั่นคงจะยุ่งเหยิง (Tangling) และกระจัดกระจาย (Scattering) ไปทั่ว ทำให้โค้ดไม่สามารถแยกย่อยเป็นส่วน ๆ เมื่อแก้ไขส่วนหนึ่งในโค้ดก็จะกระทบส่วนอื่น ทำให้บำรุงรักษายาก ซึ่งลักษณะดังกล่าวถือว่ามีสภาพมอดูลาร์ (Modularity) ที่ต่ำ โดยไอทริปเปิลอี (IEEE) ได้นิยามสภาพมอดูลาร์ว่าเป็น *"ระดับขั้นของโปรแกรมในระบบที่ถูกแยกย่อยออกมา เพื่อให้ได้ส่วนโปรแกรม (Component) ที่แยกเป็นส่วน ๆ ชัดเจน เมื่อมีการเปลี่ยนแปลงส่วนโปรแกรมหนึ่งส่วน ก็จะมีผลกระทบต่อส่วนโปรแกรมอื่น ๆ น้อยที่สุด"* [2]



ภาพที่ 1.1 โค้ดความมั่นคงที่ปรากฏซ้ำ ๆ กระจัดกระจายไปทั่ว และมีความยุ่งเหยิง อยู่ภายในระบบเว็บเซอร์วิซ

อย่างไรก็ตามสามารถนำหลักวิศวกรรมซอฟต์แวร์มาแก้ปัญหาตั้งแต่การออกแบบ โดยนำแบบรูปความมั่นคง (Security Pattern) มาทำให้การเขียนโค้ดมีแบบแผน และใช้การโปรแกรมเชิงแง่มุมหรือเอโอพี (Aspect-Oriented Programming: AOP) มาแก้ปัญหาความยุ่งเหยิงและการกระจัดกระจายของโค้ดความมั่นคง ทำให้โครงสร้างเว็บเซอร์วิซมีสภาพมอดูลาร์ดีขึ้น ยิ่งเมื่อพิจารณาแบบจำลองคุณภาพ (Quality Model) ของ McCall [3] ซึ่งระบุไว้ว่าสภาพมอดูลาร์ ซึ่งเป็นคุณลักษณะเชิงคุณภาพแบบภายใน (Internal Quality Attribute) จะมีอิทธิพลต่อคุณลักษณะเชิงคุณภาพแบบภายนอก (External Quality Attribute) เช่น การบำรุงรักษา และการนำกลับมาใช้ใหม่ เป็นต้น ดังนั้นเมื่อสภาพมอดูลาร์ถูกปรับปรุงให้ดีขึ้น ก็จะทำให้คุณลักษณะเชิงคุณภาพแบบภายนอกที่ยกตัวอย่างมาดีขึ้นตามไปด้วย

จากการศึกษา หนังสือ บทความ และวรรณกรรมต่าง ๆ ผู้วิจัยเห็นว่า แบบรูปความมั่นคงสามารถรองรับความต้องการ พิสูจน์ตัวจริง และการพิสูจน์สิทธิ์ แต่ใช้ในโปรแกรมประยุกต์บนเว็บหรือใช้ในระบบทั่ว ๆ ไป ซึ่งยังมีข้อจำกัดดังต่อไปนี้

1) แบบรูปยังไม่เฉพาะเจาะจงใช้กับเว็บเซอร์วิซ ซึ่งแบบรูปที่ใช้ในเว็บเซอร์วิซ ควรพิจารณาโค้ดในส่วนฝั่งไคลเอนต์กับฝั่งเซอร์วิซ แต่แบบรูปที่มีอยู่จะนำเสนอในฝั่งเซอร์วิซอย่างเดียว
2) บางแบบรูปยังยึดติดกับเทคโนโลยีของบริษัทใดบริษัทหนึ่ง
3) แบบรูปยังมีลักษณะเป็นแนวปฏิบัติด้านความมั่นคง แต่ไม่ได้อยู่ในรูปของแบบจำลองที่จะสามารถนำไปใช้ในการออกแบบระบบได้โดยตรง

ดังนั้นผู้วิจัยจึงนำเสนอแบบรูปความมั่นคงที่ได้จากการผสมผสานแบบรูปความมั่นคงที่มีอยู่ในวรรณกรรมต่าง ๆ ในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง เพื่อนำมาประยุกต์ใช้ในการออกแบบฝั่งเซอร์วิซ ร่วมกับการประยุกต์ใช้เอโอพีในฝั่งไคลเอนต์ แล้วทำการทดลอง และใช้ตัววัด (Metrics) ทำการวัดสภาพมอดูลาร์ ได้แก่ การต่อประกบ (Coupling) ความเชื่อมแน่น (Cohesion) ความยุ่งเหยิง และการกระจัดกระจาย โดยวัดทั้งก่อนและหลังนำแบบรูปและเอโอพีไปใช้งานจริง ซึ่งประโยชน์ที่ได้คือ ทำให้การเขียนโค้ดมีแบบแผน อีกทั้งทำให้การบำรุงรักษาและการนำกลับมาใช้ใหม่ ทำได้ง่ายขึ้น

1.2 วัตถุประสงค์ของการวิจัย

1) เพื่อเสนอแบบรูปความมั่นคงในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง สำหรับเว็บเซอร์วิซ โดยการผสมผสานแบบรูปที่มีอยู่
2) เพื่อประยุกต์ใช้แบบรูปความมั่นคงในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ที่ถูกนำเสนอ ร่วมกับการโปรแกรมเชิงแง่มุม กับเว็บเซอร์วิซ เพื่อปรับปรุงสภาพมอดูลาร์

1.3 ขอบเขตของการวิจัย

1) แบบรูปความมั่นคงที่งานวิจัยนี้เสนอเพื่อนำมาใช้กับเว็บเซอร์วิซ ในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ได้มาจากการผสมผสานแบบรูปจากหนังสือของ Steel [4] และคณะ กับหนังสือของ Schumacher และคณะ [5] จำนวน 6 แบบรูป
2) ประยุกต์แบบรูปในขั้นตอนการออกแบบฝั่งเซอร์วิซ และประยุกต์การโปรแกรมเชิงแง่มุมในฝั่งไคลเอนต์
3) ใช้ระบบเว็บเซอร์วิซของบริษัทโทรคมนาคมแห่งหนึ่งเป็นกรณีศึกษาสำหรับการทดลองประยุกต์แบบรูปและการโปรแกรมเชิงแง่มุม โดยมีบริการฝั่งเซอร์วิซอย่างน้อย 7 เซอร์วิซ และฝั่งไคลเอนต์อย่างน้อย 5 เซอร์วิซ
4) คุณสมบัติของสภาพมอดูลาร์ที่จะทำการปรับปรุงคุณภาพ ได้แก่ การต่อประกบ ความเชื่อมแน่น ความยุ่งเหยิง และการกระจัดกระจาย
5) ทำการปรับปรุงสภาพมอดูลาร์โนโปรแกรมประยุกต์หลักเท่านั้น ไม่ได้ปรับปรุงในโค้ดที่เป็นแบบรูป

6) มาตรฐานความมั่นคงบนเว็บเซอร์วิซที่ใช้ คือ WS-Security 2004
7) เทคโนโลยีโอเพนซอร์ซที่ใช้ ได้แก่ Apache CXF, AspectJ และ Tomcat

1.4 ขั้นตอนการวิจัย

1) ศึกษาวิธีการปรับปรุงคุณภาพซอฟต์แวร์ด้วยแบบรูปความมั่นคง
2) ศึกษาวิธีการปรับปรุงคุณภาพซอฟต์แวร์ด้วยโปรแกรมเชิงแง่มุม
3) ศึกษาตัววัดคุณภาพซอฟต์แวร์ในด้านสภาพมอดูลาร์
4) กำหนดแบบรูปความมั่นคงสำหรับเว็บเซอร์วิซในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง
5) ทำการทดลองตามวิธีวิจัย
6) ประเมินผลวิธีวิจัย
7) จัดทำบทความวิชาการ
8) จัดทำเล่มวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) ได้แบบรูปความมั่นคงด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ซึ่งผสมผสานแบบรูปความมั่นคงที่มีอยู่ สำหรับการทำงานของระบบเว็บเซอร์วิซ ทำให้การเขียนโค้ดมีแบบแผนมากขึ้น
2) แบบรูปความมั่นคงด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ซึ่งมีการประยุกต์ใช้ร่วมกับวิธีเชิงแง่มุมทำให้สามารถปรับปรุงสภาพมอดูลาร์ของระบบเว็บเซอร์วิซโดยรวม ทั้งฝั่งเซอร์วิซและฝั่งไคลเอนต์
3) ช่วยให้การบำรุงรักษาระบบซอฟต์แวร์ที่เป็นเว็บเซอร์วิซในส่วนความต้องการด้านความมั่นคงทำได้ง่าย และสามารถนำโค้ดส่วนความมั่นคงกลับมาใช้งานใหม่ได้ง่าย

1.6 ผลงานตีพิมพ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้รับการตอบรับเพื่อตีพิมพ์เป็นบทความวิจัยในหัวข้อเรื่อง “การประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิส” หน้าที่ 583-589 โดย จตุรพัชร์ พัฒนทรงศิวิไล และ ทวิตีย์ เสนีวงศ์ ณ อยุธยา ในการประชุมวิชาการ The 2013 International Computer Science and Engineering Conference (ICSEC 2013) ซึ่งจัดขึ้นโดย

มหาวิทยาลัยศิลปากร ที่โรงแรม วินเซอร์ สวีทส์ สุขุมวิท 20 กทม ระหว่างวันที่ 4 – 6 กันยายน 2556

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

ทฤษฎีที่เกี่ยวข้องที่กล่าวถึงในงานวิจัยได้แก่ ความต้องการด้านความมั่นคงของเว็บเซอร์วิซ (Web Services Security Requirements) มาตรฐานความมั่นคงของเว็บเซอร์วิซ (Web Services Security Standards) แบบรูปความมั่นคง การโปรแกรมเชิงแง่มุมหรือเอโอพี คำศัพท์ที่สำคัญของการโปรแกรมเชิงแง่มุม แนวคิดการโปรแกรมเชิงแง่มุม และตัววัดสำหรับสภาพมอดูลาร์ ซึ่งมีรายละเอียดดังนี้

2.1.1 ความต้องการด้านความมั่นคงของเว็บเซอร์วิซ

ความต้องการด้านความมั่นคงของเว็บเซอร์วิซในด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชัน มีความหมายดังนี้

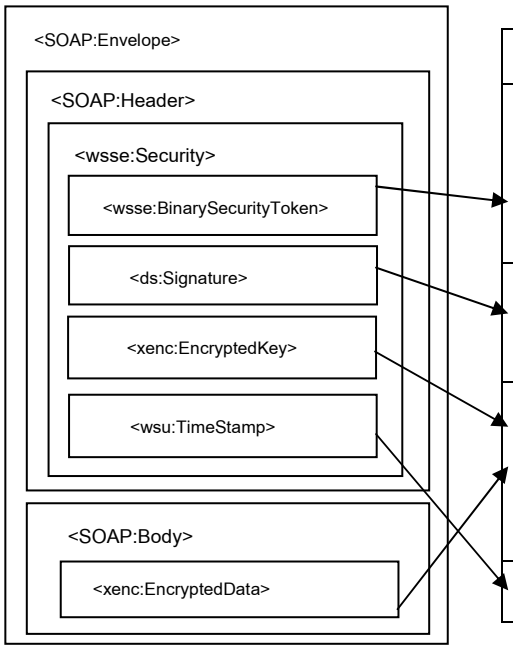
การพิสูจน์ตัวจริง [4] เป็นการบังคับให้มีการทวนสอบ (Verification) และมีการตรวจสอบความสมเหตุสมผล (Validation) ของข้อมูลระบุตัวตน ระหว่างผู้ให้บริการเว็บเซอร์วิซกับผู้เรียกใช้ โดยเริ่มต้น ผู้ร้องขอบริการต้องได้รับการพิสูจน์ตัวจริงด้วยการระบุตัวตน ตามข้อมูลระบุตัวตนที่ส่งไปให้

การพิสูจน์สิทธิ์ [4] และการให้สิทธิ์ เป็นขั้นตอนหลังการพิสูจน์ตัวจริง จะกลายมาเป็นขั้นตอนสำคัญเพื่อควบคุมและเฝ้าสังเกต (Monitor) เมื่อมีการเข้าถึงทรัพยากรที่ผู้ให้บริการเตรียมไว้ให้ สำหรับการพิสูจน์สิทธิ์จะต้องกำหนดกฎและนโยบายที่มีการควบคุมเมื่อเข้าใช้ทรัพยากรที่ต้องการ หลังจากพิสูจน์ตัวจริงสำเร็จ ผู้ร้องขอบริการสามารถเข้าใช้บริการทางธุรกิจตามสิทธิ์ในการเข้าถึงทรัพยากรนั้น ๆ จากนั้นผู้ร้องขอซึ่งได้รับสิทธิ์ควรถูกเฝ้าสังเกตว่า ควรได้รับการอนุญาตหรือปฏิเสธการเข้าถึงทรัพยากรอื่น ๆ ตามความเหมาะสม

เซสชัน [1] หมายถึง การคงไว้ซึ่งช่วงเวลาของการติดต่อสื่อสารระหว่างผู้ใช้งานที่เกี่ยวข้องกับส่วนประกอบต่าง ๆ ภายในระบบคอมพิวเตอร์ ซึ่งต้องมีการจดจำสถานะตลอดช่วงเวลาการติดต่อสื่อสาร (ซึ่งในงานวิจัยนี้จะเป็นเซสชันด้านความมั่นคง กล่าวคือจะจดจำการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ตลอดช่วงเวลาการสื่อสาร)

2.1.2 มาตรฐานความมั่นคงของเว็บเซอร์วิซ

มาตรฐานความมั่นคงของเว็บเซอร์วิซอธิบายได้ดังภาพที่ 2.1 และ ตารางที่ 2.1



ตารางที่ 2.1 องค์ประกอบของโซปที่มีข้อมูลความมั่นคง

ส่วนประกอบ	อธิบาย
Security Token	ใช้ระบุชื่อผู้ใช้ (Username)และรหัสผ่าน (Password) หรือระบุโทเค็น (Token)
XML Signature	ใช้ทำลายเซนต์ดิจิทัล (Digital Signature)
XML Encryption	ทำการเข้ารหัสได้ทั้งส่วนของเฮดเดอร์ (Header) กับบอดี้ (Body) ของโครงสร้างโซป
Timestamp	ใช้ประทับตราเวลา

ภาพที่ 2.1 โครงสร้างความมั่นคงบนโซป

องค์กรโอเอซิส (OASIS) ได้กำหนดมาตรฐานความมั่นคงของเว็บเซอร์วิซ [6] ไว้ในเนื้อข้อความโซปเรียกว่า WS-Security 2004 โดยเพิ่มส่วนประกอบด้านความมั่นคงขึ้นมาในโครงสร้างโซปดังภาพที่ 2.1 และตารางที่ 2.1 ซึ่งการเขียนโค้ดเพื่อพิสูจน์ตัวจริง (ด้วยการระบุชื่อผู้ใช้และรหัสผ่าน) ความต้องการด้านการพิสูจน์สิทธิ์ และการใช้โทเค็นเพื่อจดจำสถานะของไคลเอนต์ตลอดช่วงเวลาสื่อสาร หรือเซสชัน (Session) จะใช้ส่วน Security Token เท่านั้น

2.1.3 แบบรูปความมั่นคง

Schumacher และ Roedig [7] ได้นิยามแบบรูปความมั่นคงว่าเป็น "*การอธิบายปัญหาด้านความมั่นคงที่เกิดขึ้นประจำโดยเฉพาะ ซึ่งเป็นปัญหาที่เกิดขึ้นในบริบทที่เฉพาะเจาะจงลงไป พร้อมทั้งนำเสนอถึงแนวทางการแก้ปัญหาที่ใช้ได้ทั่ว ๆ ไป ซึ่งถูกพิสูจน์มาแล้วว่าใช้ได้ดี*" โดยมีหนังสือที่

รวบรวมและเสนอแบบรูปความมั่นคงตีพิมพ์ออกมาหลายเล่ม แต่งานวิจัยนี้ใช้หนังสือของ Steel [4] และคณะ กับหนังสือของ Schumacher และคณะ [5] สำหรับแบบรูปจากหนังสือสองเล่มดังกล่าว ที่เกี่ยวกับความมั่นคงด้าน การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันความมั่นคง ซึ่งใช้เป็นแนวทางวิจัย จะสรุปไว้ในตารางที่ 2.2

ตารางที่ 2.2 แบบรูปความมั่นคงที่งานวิจัยใช้เป็นแนวทาง

แบบรูป	**คำอธิบายของแบบรูป**
หนังสือ Core Security Patterns: Best Practices and Strategies for J2EE™, Web Services, and Identity Management [4]	
Authentication Enforcer	เบราว์เซอร์ (Browser) ของไคลเอนต์จะเป็นฝ่ายพิสูจน์ตัวจริงกับเซิร์ฟเวอร์ และแบบรูปจะจัดการด้านพิสูจน์ตัวจริงจากคำร้องขอ (ส่งมาทางโพรโทคอลเอชทีทีพี)
Authorization Enforcer	จะจัดการเรื่องตรวจสอบสิทธิ์เพื่ออนุญาตให้เข้าใช้ทรัพยากรต่าง ๆ จากคำร้องขอ (ส่งมาทางโพรโทคอลเอชทีทีพี)
หนังสือ Security patterns: Integrating security and systems engineering [5]	
Single Access Point	การเข้าถึงระบบจากภายนอกต้องถูกป้องกันให้มีช่องทางเข้าถึงได้ทางเดียว
Check Point	ใช้ระบุตัวตนและพิสูจน์ตัวจริง อีกทั้งตรวจสอบสิทธิ์การเข้าใช้โดยไม่ได้รับอนุญาต
Security Session	ต้องการระบุตัวตนและพิสูน์ตัวจริง เพียงครั้งเดียว ตลอดทั้งจดจำสิทธิ์ที่เข้าใช้ทรัพยากรต่าง ๆ ไว้ตลอดช่วงเวลาสื่อสาร หรือเซสชัน
Authorization	ใช้กำหนดบทบาทว่า ใครสามารถเข้าถึงทรัพยากรใด และจะใช้งานอย่างไร

2.1.4 การโปรแกรมเชิงแง่มุมหรือเอโอพี

แนวคิดการโปรแกรมเชิงแง่มุมหรือเอโอพี ถูกนำเสนอครั้งแรกโดย Kiczales และคณะ [8] ที่ศูนย์วิจัย Xerox Palo Alto โดยเป็นเพียงกระบวนทัศน์ในการเขียนโปรแกรม เพื่อแก้ปัญหาการตัดขวางของคอนเซิร์น (Crosscutting Concerns) โดยที่ *คอนเซิร์น* (Concern) ตามคำอธิบายของ ไอทริปเปิลอี 1471 คือสิ่งที่น่าสนใจที่เกี่ยวข้องกับการพัฒนาระบบ และมีความสำคัญ เช่น ประสิทธิภาพ ความน่าเชื่อถือ และความมั่นคง เป็นต้น (สำหรับงานวิจัยนี้ คอนเซิร์นจะอ้างถึงโค้ด

ความมั่นคง) และ *การตัดขวางของคอนเซิร์น* หมายถึง โค้ดของคอนเซิร์นที่กระจัดกระจายไปทั่วทั้งโค้ด และมีความยุ่งเหยิง [8] จากนั้น Kiczales และคณะ [9] ได้สร้างกรอบงาน AspectJ สำหรับการพัฒนาเอโอพี

2.1.5 คำศัพท์ที่สำคัญของการโปรแกรมเชิงแง่มุม

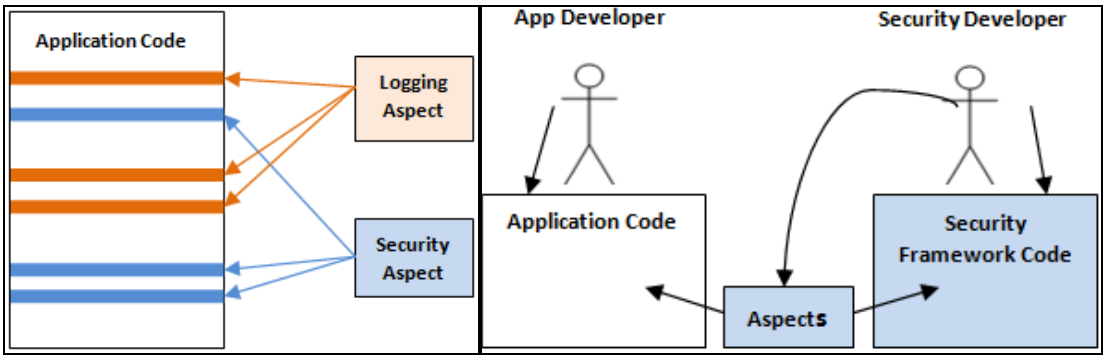
คำศัพท์ที่สำคัญของการโปรแกรมเชิงแง่มุม สามารถอธิบายได้ดังตารางที่ 2.3

ตารางที่ 2.3 คำศัพท์ที่สำคัญของการโปรแกรมเชิงแง่มุม [9]

คำศัพท์	คำอธิบาย
แอสเปกต์ (Aspect)	ส่วนคอนเซิร์นที่มีการตัดขวางแต่พัฒนาให้มีสภาพมอดูลาร์
จอยพอยต์ (Join Points)	จุดต่าง ๆ ในการประมวลผลของโปรแกรม
จุดตัด (Point Cut)	กลุ่มของจอยพอยต์กับค่าที่จอยพอยต์ดังกล่าวซึ่งเป็นที่สนใจและต้องการอ้างถึง
แอดไวซ์ (Advice)	พฤติกรรมที่กำหนดเพิ่มเติมให้กับโปรแกรม ณ จุดตัดที่สนใจ

2.1.6 แนวคิดการโปรแกรมเชิงแง่มุม

หลักการโปรแกรมเชิงแง่มุม สามารถอธิบายได้ดังภาพที่ 2.2



ภาพที่ 2.2 หลักการโปรแกรมเชิงแง่มุม

ในภาพที่ 2.2 ทางซ้ายมือ เป็นตัวอย่างของ ความต้องการสำหรับการลงบันทึก (Logging) กับความมั่นคง ซึ่งถือเป็นคอนเซิร์นที่มีลักษณะการตัดขวาง กล่าวคือ การเขียนโค้ดเพื่อการลง

บันทึก และโค้ดเพื่อจัดการความมั่นคง จะปรากฏซ้ำ ๆ มีการกระจัดกระจายไปทั่วทั้งโค้ด และมีความยุ่งเหยิง

ส่วนรูปทางขวามือ จะเป็นตัวอย่างแนวคิดการโปรแกรมเชิงแง่มุม ที่แยกโค้ดความมั่นคงออกจากส่วนของโปรแกรมประยุกต์หลัก ทำให้โค้ดหลักเป็นระเบียบขึ้น เมื่อโปรแกรมถูกทำงานจริง โค้ดในส่วนความมั่นคงที่เป็นแอดไวซ์จะประสานเข้ากับโค้ดหลักที่จุดตัด (หรือเรียกว่าวีฟ (Weave)) หลังจากนั้นจึงประมวลผลตามปกติ

ประโยชน์ที่ได้คือ เดิมรูปทางซ้าย โค้ดจะมีความยุ่งเหยิง และกระจัดกระจาย ยากต่อการพัฒนา และบำรุงรักษา [8] แต่เมื่อนำแนวคิดนี้มาใช้ โค้ดความมั่นคงจะรวมเป็นมอดูลมากขึ้น สภาพมอดูลาร์ดีขึ้น ส่วนประโยชน์อื่นตัวอย่างเช่น แยกหน้าที่ความรับผิดชอบของผู้พัฒนาโปรแกรมหลัก กับส่วนความมั่นคงออกจากกัน [10] เป็นต้น

2.1.7 ตัววัดสำหรับสภาพมอดูลาร์

ตัววัดสำหรับสภาพมอดูลาร์ที่ใช้ในงานวิจัยนี้นำมาจาก [11] ได้แก่ CBM, LCO, CDO, CDC และ LOCC ดังในตารางที่ 2.4 ซึ่งถ้าตัวเลขเหล่านี้มีค่าน้อยลงแสดงว่าสภาพมอดูลาร์ดีขึ้น

ตารางที่ 2.4 ตัววัดสำหรับสภาพมอดูลาร์ที่ใช้ในงานวิจัยนี้

ตัววัด	ความหมาย
การต่อประกบระหว่างมอดูล (Coupling Between Modules: CBM)	วัดจากจำนวนมอดูล (คลาสหรือแอสเปกต์) ซึ่งต่อประกบกับมอดูลอื่น โดยมอดูลอื่นทำการอ้างถึงมอดูลนั้น ซึ่งตัววัดนี้ใช้วัดการต่อประกบ
การขาดความเชื่อมแน่น (Lack of Cohesion: LCO)	วัดจากจำนวนคู่ของตัวดำเนินการ (เมท็อดหรือแอดไวซ์) ภายในมอดูล (คลาสหรือแอสเปกต์) ซึ่งไม่ได้เข้าถึงตัวแปรอินสแตนซ์ (Instance Variable) ตัวเดียวกัน ซึ่งตัววัดนี้วัดความเชื่อมแน่น
การแพร่ของคอนเซิร์นบนตัวดำเนินการ Concern Diffusion over Operations (CDO)	วัดจากจำนวนตัวดำเนินการที่มีจุดประสงค์เพื่อพัฒนาคอนเซิร์น และจำนวนตัวดำเนินการ (เมท็อดหรือแอดไวซ์) อื่น ๆ ที่ใช้งานมัน ซึ่งตัววัดนี้ใช้วัดความยุ่งเหยิงและการกระจัดกระจาย

ตารางที่ 2.4 ตัววัดสำหรับสภาพมอดูลาร์ที่ใช้ในงานวิจัยนี้ (ต่อ)

ตัววัด	ความหมาย
การแพร่ของคอนเซิร์นบนส่วนโปรแกรม (Concern Diffusion over Components: CDC)	วัดจากจำนวนมอดูลที่มีจุดประสงค์เพื่อพัฒนาคอนเซิร์น และจำนวนมอดูลอื่น ๆ ที่ใช้งานมัน ซึ่งตัววัดนี้ใช้วัดความยุ่งเหยิงและการกระจัดกระจาย
จำนวนบรรทัดของโค้ดภายในคลาส Lines of Class Code (LOCC)	วัดจากจำนวนบรรทัดของโค้ดภายในคลาส

2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง

จากการศึกษางานวิจัยที่เกี่ยวข้อง เพื่อใช้เป็นแนวทางประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิซ สามารถแบ่งกลุ่มงานวิจัยออกเป็นสามกลุ่มหลักดังต่อไปนี้

2.2.1 ประโยชน์ของการใช้แบบรูปความมั่นคง

แบบรูปเป็นแนวคิดที่สำคัญในงานวิศวกรรมซอฟต์แวร์ตั้งแต่ขั้นตอนการออกแบบ จากการศึกษางานวิจัยที่เกี่ยวข้อง มีการกล่าวถึงข้อดีการใช้แบบรูปความมั่นคง เช่น งานวิจัยของ Halkidis และคณะ [12] ได้แสดงว่าการนำแบบรูปความมั่นคงไปใช้จะมีผลให้ระบบทนทาน (Robust) มากขึ้น เมื่อเปรียบเทียบกับระบบที่ไม่ใช้แบบรูป และงานวิจัยของ Chawla และ Mehta [13] ได้อธิบายการนำแบบรูปความมั่นคงมาใช้ในวงจรการพัฒนาระบบ (System Development Life Cycle: SDLC) ทำให้ช่วยลดความผิดพลาดในการพัฒนา

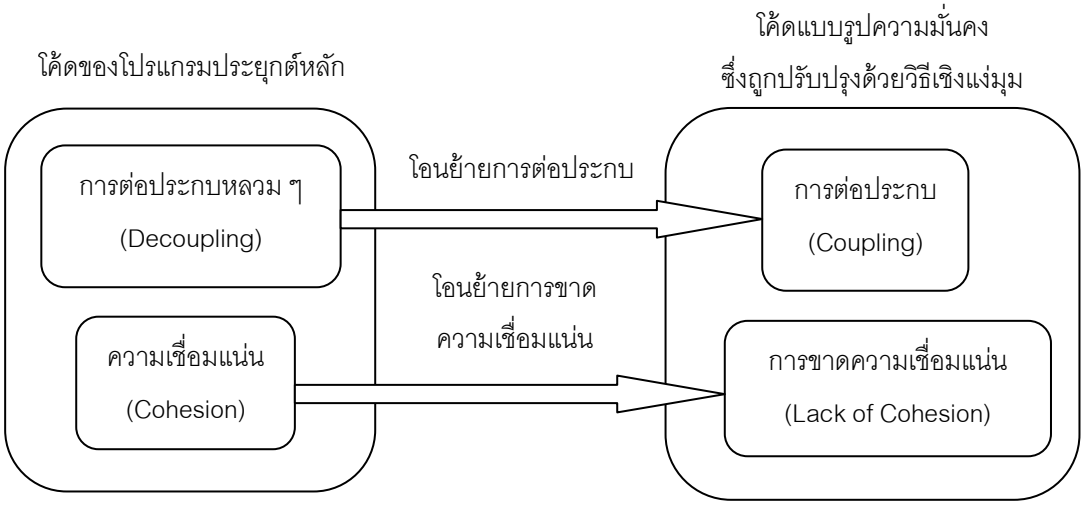
อย่างไรก็ตามแบบรูปยังมีจุดอ่อนตามการอธิบายของ Hachani และ Bardou [14] กล่าวคือ แบบรูปความมั่นคงเหมือนกับแบบรูปทั่ว ๆ ไป จะมีจุดอ่อนเรื่องความยุ่งเหยิงและการกระจัดกระจายของโค้ด เนื่องมาจากการตัดขวางของคอนเซิร์นของโค้ดความมั่นคง ซึ่งเกิดจากการขึ้นต่อกันระหว่างวัตถุ (Object) ที่เพิ่มขึ้น การขาดการตามรอย (Traceability) ของแบบรูปความมั่นคงในโค้ด และความซับซ้อนในโค้ดที่เพิ่มขึ้น

2.2.2 การปรับปรุงคุณภาพซอฟต์แวร์ด้วยการโปรแกรมเชิงแง่มุม

ตั้งแต่ Kiczales และคณะ [8] นำเสนอแนวคิดการโปรแกรมเชิงแง่มุมครั้งแรก เพื่อแก้ปัญหาโค้ดที่ยุ่งเหยิงและกระจัดกระจายไปทั่ว จากนั้นแนวคิดนี้ถูกนำไปใช้ในงานด้านความมั่นคง ตัวอย่างเช่น งานวิจัยของ Georg และคณะ [15] นำเสนอและแสดงถึงเทคนิคการออกแบบเชิงแง่มุม สำหรับงานออกแบบระบบความมั่นคง ซึ่งมองคอนเซิร์นของความมั่นคงในระดับการออกแบบที่ตัดขวางไปทั่วแผนภาพยูเอ็มแอล (UML) แต่งานนี้ยังไม่ได้ระบุชัดเจนว่าแนวคิดเชิงแง่มุมสามารถปรับปรุงคุณภาพในด้านสภาพมอดูลาร์

อย่างไรก็ตามมีการนำวิธีเชิงแง่มุมมาปรับปรุงคุณภาพ เช่น งานวิจัยของ Sonchaiwanich และคณะ [10] แสดงถึงแนวคิดเชิงแง่มุมช่วยแยกตรรกะทางธุรกิจ (Business Logic) กับคอนเซิร์นด้านความมั่นคง (Security Concern) ออกจากกันในขั้นตอนการออกแบบและพัฒนาซอฟต์แวร์ระบบเอสโอเอ (SOA) ซึ่งส่งผลให้เกิดความคล่องตัวในการดำเนินการทางธุรกิจ และเพิ่มสภาพมอดูลาร์ แต่งานวิจัยนี้เป็นการเขียนโค้ดความมั่นคงโดยไม่ใช้แบบรูปใด ๆ และในงานวิจัยของ Garcia และคณะ [16] กับงานวิจัยของ Hannemann และ Kiczales [17] ได้แสดงว่าการปรับปรุงสภาพมอดูลาร์ของระบบที่เป็นเชิงแง่มุมจะเหนือกว่าที่เป็นเชิงวัตถุ (Object-Oriented) ของฟังก์ชันงานแบบเดียวกัน แต่งานดังกล่าวไม่ได้ใช้ในโดเมนความมั่นคง

ต่อมามีงานวิจัยที่นำแนวคิดเชิงแง่มุมมาปรับปรุงสภาพมอดูลาร์ในโค้ดที่มีแบบรูปความมั่นคง โดย Edge และ Mitropoulos [11] ได้ศึกษาสองวิธี ได้แก่ วิธีพัฒนาแบบรูปความมั่นคงด้วยวิธีเชิงวัตถุอย่างเดียว กับผสมผสานวิธีเชิงแง่มุมเข้าไป แล้วทดลองวัดสภาพมอดูลาร์ ทั้งในโค้ดโปรแกรมประยุกต์บนเว็บ กับโค้ดแบบรูปความมั่นคง เพื่อวิเคราะห์คุณภาพเมื่อนำวิธีเชิงแง่มุมมาใช้ และต่อมาพวกเขา [18] ได้พัฒนายุทธวิธีเชิงแง่มุม (Aspect-Oriented Strategy) ให้มีรูปแบบที่เป็นทางการ ซึ่งแต่ละยุทธวิธีของงานชิ้นนี้ได้นำแนวคิดเชิงแง่มุมไปใช้ผสมผสานร่วมกับแบบรูปที่เขียนด้วยวิธีเชิงวัตถุเพียงบางส่วนของแบบรูปเท่านั้น และผลการทดลองในงานวิจัย [11,18] ยืนยันตรงกันว่า การผสมผสานวิธีเชิงแง่มุมเข้ากับแบบรูปที่เขียนด้วยวิธีเชิงวัตถุ ช่วยปรับปรุงสภาพมอดูลาร์ในโค้ดหลักดีกว่าการใช้วิธีเชิงวัตถุอย่างเดียว โดยสาเหตุที่ช่วยปรับปรุงให้ดีขึ้น ตามมุมมองของผู้วิจัยอธิบายดังภาพที่ 2.3



ภาพที่ 2.3 การปรับปรุงคุณภาพซอฟต์แวร์ด้วยการโปรแกรมเชิงแง่มุม

ในภาพที่ 2.3 แสดงการนำวิธีเชิงแง่มุมมาใช้กับแบบรูปความมั่นคง โดยภาพซ้ายมือเป็นส่วนโค้ดโปรแกรมประยุกต์หลัก จะเห็นว่าการต่อประกบจะหลวมมากขึ้นและความเชื่อมแน่นจะเพิ่มมากขึ้น เนื่องจากการต่อประกบถูกย้ายไปอยู่ในฝั่งรูปทางขวามือ ซึ่งคือโค้ดแบบรูปความมั่นคงที่ถูกเขียนด้วยวิธีเชิงแง่มุม เช่นเดียวกันความเชื่อมแน่นจากโค้ดโปรแกรมประยุกต์หลักเพิ่มขึ้นในฝั่งซ้ายมือ เพราะการขาดความเชื่อมแน่นถูกโอนย้ายไปอยู่ฝั่งขวามือแทน

สำหรับข้อจำกัดของวิธีนี้คือ ไม่สามารถปรับปรุงสภาพมอดูลาร์ทั้งโปรแกรมประยุกต์ เพราะมันจะปรับปรุงคุณภาพในโค้ดหลักอย่างเดียว แล้วไปลดสภาพมอดูลาร์ในแบบรูปความมั่นคงแทน ดังนั้นผู้วิจัยจึงอ้างอิงงานวิจัย [11, 18] มาพัฒนาแบบรูปความมั่นคงสำหรับเว็บเซอร์วิซ เพื่อช่วยปรับปรุงสภาพมอดูลาร์ในโค้ดหลักเพียงอย่างเดียว ยังไม่พิจารณาการปรับปรุงในส่วนแบบรูปความมั่นคง

จากงานวิจัยที่กล่าวมาสามารถเปรียบเทียบกับงานของผู้วิจัยว่ามีความแตกต่างกันอย่างไร ดังในตารางที่ 2.5

ตารางที่ 2.5 การเปรียบเทียบงานวิจัยก่อนหน้ากับงานของผู้วิจัย

ชื่องานวิจัยก่อนหน้า	**เปรียบเทียบงานวิจัยก่อนหน้ากับงานของผู้วิจัย**	
	สิ่งที่เหมือนกัน	**สิ่งที่แตกต่างกัน**
Using Aspects to Design a Secure System [15]	- ใช้วิธีเชิงแง่มุม - ใช้กับความมั่นคงด้านการพิสูจน์ตัวจริง	- ไม่ใช้แบบรูปความมั่นคง - ไม่ใช้กับเว็บเซอร์วิซ
Using AOP to Separate SOA Security Concerns from Application Implementation [10]	- ใช้วิธีเชิงแง่มุม - ใช้กับความมั่นคง - ปรับปรุงสภาพมอดูลาร์ - ใช้กับเว็บเซอร์วิซ	- ไม่ใช้แบบรูปความมั่นคง
Modularizing Design Patterns with Aspects: A Quantitative Study [16]	- ใช้วิธีเชิงแง่มุม - ปรับปรุงสภาพมอดูลาร์	- ไม่ใช้ในโดเมนความมั่นคง - ใช้กับแบบรูป Gang-of-Four (GOF) - ไม่ใช้กับเว็บเซอร์วิซ
Design Pattern Implementation in Java and AspectJ [17]	- ใช้วิธีเชิงแง่มุม - ปรับปรุงสภาพมอดูลาร์	- ไม่ใช้ในโดเมนความมั่นคง - ใช้กับแบบรูป GOF - ไม่ใช้กับเว็บเซอร์วิซ

ตารางที่ 2.5 การเปรียบเทียบงานวิจัยก่อนหน้ากับงานของผู้วิจัย (ต่อ)

ชื่องานวิจัยก่อนหน้า	เปรียบเทียบงานวิจัยก่อนหน้ากับงานของผู้วิจัย	
	สิ่งที่เหมือนกัน	**สิ่งที่แตกต่างกัน**
Quantitative Analysis of Modularity Tradeoffs with AspectJ Web-Tier Security Patterns [11]	- ใช้วิธีเชิงแง่มุม - ใช้กับความมั่นคงด้านพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ - ใช้แบบรูปในหนังสือของ Steel [4] และคณะ ได้แก่ Authentication Enforcer และ Authorization Enforcer - ปรับปรุงสภาพมอดูลาร์	- ไม่ใช้แบบรูปในหนังสือของ Schumacher และคณะ [7] - ไม่ใช้กับเว็บเซอร์วิซ
Improving Security Design Patterns with Aspect-Oriented Strategies [18]	- ใช้วิธีเชิงแง่มุม - ใช้กับความมั่นคงด้านพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ - ใช้แบบรูปในหนังสือของ Steel [4] และคณะ ได้แก่ Authentication Enforcer และ Authorization Enforcer - ปรับปรุงสภาพมอดูลาร์	- ไม่ใช้แบบรูปในหนังสือของ Schumacher และคณะ [7] - ไม่ใช้กับเว็บเซอร์วิซ

2.2.3 ตัววัดคุณภาพซอฟต์แวร์ในด้านสภาพมอดูลาร์

Greenwood และคณะ [19] ได้ออกแบบแพลตฟอร์มเพื่อการทดสอบสำหรับโครงการที่พัฒนาซอฟต์แวร์ด้วยวิธีเชิงแง่มุม (A Testbed for Aspect Oriented Software Development: TAO) ซึ่งจะเปรียบเทียบระบบที่พัฒนาด้วยวิธีเชิงแง่มุม กับเทคนิคการพัฒนาด้วยวิธีมอดูลอื่น ๆ โดยมีโปรแกรมประยุกต์ที่มีการวัดเปรียบเทียบสมรรถนะ (Benchmark)) และชุดตัววัดสภาพมอดูลาร์ ซึ่งถูกเตรียมไว้ให้ จากงานดังกล่าว Edge และ Mitropoulos ได้นำตัววัดไปใช้ในงานของเขา [11] ด้วยการนำโปรแกรมประยุกต์บนเว็บซึ่งมีแบบรูปความมั่นคงอยู่ภายใน แต่เขียนด้วยวิธีเชิงวัตถุ มาวัดค่าสภาพมอดูลาร์ทั้งก่อนและหลังการนำวิธีเชิงแง่มุมมาใช้กับแบบรูป ซึ่งใช้ 15 ตัววัด

จากนั้นพวกเขาทำการวิจัยต่อ [18] เพื่อพัฒนายุทธวิธีเชิงคุณลักษณะดังที่กล่าวมาในหัวข้อที่ 2.2.2 โดยใช้โปรแกรมประยุกต์บนเว็บด้วยตัวอย่างเดียวกัน แล้วนำมาวัดค่าก่อนและหลังใช้แบบรูป แต่ได้ลดตัววัดเหลือ 12 ตัว

บทที่ 3
การกำหนดแบบรูปความมั่นคงสำหรับเว็บเซอร์วิซ

ในบทนี้จะกล่าวถึงแนวคิดการประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิซ เพื่อที่จะได้แบบรูปความมั่นคงสำหรับเว็บเซอร์วิซ สำหรับแก้ปัญหาที่ผู้พัฒนาต่างคนต่างเขียนโค้ดความมั่นคงทำให้ไม่มีแบบแผน และปัญหาที่โค้ดความมั่นคงยุ่งเหยิง และกระจัดกระจายไปทั่ว ซึ่งมีแนวคิดและวิธีดำเนินการวิจัยดังต่อไปนี้

3.1 กำหนดองค์ประกอบของแต่ละแบบรูป

แม้ว่าแบบรูปความมั่นคงมีข้อจำกัดในการปรับปรุงสภาพมอดูลาร์ แต่ก็มีประโยชน์ในงานวิศวกรรมซอฟต์แวร์ตามที่กล่าวมาในข้อที่ 2.2.1 และผู้วิจัยใช้แบบรูปมาทำให้การเขียนโค้ดความมั่นคงมีแบบแผน โดยเริ่มต้นจะกำหนดองค์ประกอบของแบบรูปความมั่นคงสำหรับเว็บเซอร์วิซ ซึ่งองค์ประกอบบางส่วนอิงจาก [4, 5] ดังตารางที่ 3.1

ตารางที่ 3.1 องค์ประกอบของแต่ละแบบรูปความมั่นคง

องค์ประกอบแบบรูป	รายละเอียด
ชื่อแบบรูป (Pattern Name)	ชื่อแบบรูปใช้คำศัพท์ที่สื่อความหมายชัดเจน
จุดประสงค์ (Intent)	อธิบายจุดประสงค์และเหตุผลของการใช้แบบรูปนี้
บริบท (Context) [5]	อธิบายสถานการณ์ที่จะนำแบบรูปนี้ไปใช้งาน
ปัญหา (Problem) [5]	ระบุปัญหาที่ต้องใช้แบบรูปนี้มาแก้ไข
การแก้ปัญหา (Solution) [5]	วิธีการแก้ปัญหาของแบบรูปนี้
โครงสร้าง (Structure) [5]	อธิบายถึงโครงสร้างของระบบเว็บเซอร์วิซ ที่จะใช้แก้ปัญหา
ผลที่ตามมา (Consequences) [4]	ประโยชน์หรืออาจเป็นผลกระทบที่ตามมาหลังจากใช้แบบรูปนี้
แบบรูปที่เกี่ยวข้อง (Related Pattern) [4]	ระบุถึงแบบรูปอื่น ๆ ที่เกี่ยวข้องกัน

3.2 เลือกแบบรูป

จากการศึกษาหนังสือของ Steel และคณะ [4] กับหนังสือของ Schumacher และคณะ [5] มีแบบรูปที่ใช้กับเว็บเซอร์วิซ และแบบรูปที่เกี่ยวกับความมั่นคงด้าน การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ดังแสดงในตารางที่ 3.2 อีกทั้งในตารางนี้ยังแสดงแบบรูปที่ถูกคัดเลือกเพื่อใช้เป็นแนวทางวิจัย

ตารางที่ 3.2 ลักษณะของแบบรูปความมั่นคงที่เกี่ยวข้องและการเลือกแบบรูปความมั่นคงที่ใช้ในงานวิจัยนี้

แบบรูป	**สอดคล้องกับ การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง**	**ใช้กับเว็บเซอร์วิซโดยเฉพาะ**	**เลือก**
หนังสือ *Core Security Patterns: Best Practices and Strategies for J2EE™, Web Services, and Identity Management* [4]			
Interceptor Gateway	-	✓	-
Message Inspector	-	✓	-
Secure Message Router	-	✓	-
Authentication Enforcer	✓	-	✓
Authorization Enforcer	✓	-	✓
Secure Session Object	✓	-	-
Secure Session Manager	✓	-	-
หนังสือ *Security Patterns: Integrating Security and Systems Engineering* [5]			
Single Access Point	✓	-	✓
Check Point	✓	-	✓
Authorization	✓	-	✓
Security Session	✓	-	✓

จากตารางที่ 3.2 ในหนังสือ Steel และคณะ [4] ได้ระบุแบบรูปที่ใช้กับเว็บเซอร์วิซโดยเฉพาะ ได้แก่ Interceptor Gateway, Message Inspector และ Secure Message Router แต่แบบรูปเหล่านี้ไม่ได้ใช้กับความมั่นคงด้านการการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ ผู้วิจัยจึงไม่

เลือกแบบรูปชุดนี้ ส่วนแบบรูป Secure Session Object กับ Secure Session Manager ยึดติดกับเทคโนโลยีของเจทูอีอี (J2EE) มากเกินไป จึงไม่เลือกแบบรูปชุดนี้ สำหรับแบบรูป Authentication Enforcer กับ Authorization Enforcer ใช้ได้กับการพิสูจน์สิทธิ์ และการพิสูจน์ตัวจริง ตามลำดับ ถึงจะไม่ใช้กับเว็บเซอร์วิซโดยเฉพาะ แต่ผู้วิจัยเห็นว่าสามารถใช้เป็นแนวทางวิจัยได้ จึงเลือกแบบรูปชุดนี้

ส่วนในหนังสือของ Schumacher และคณะ [5] ถึงแม้ไม่มีแบบรูปใดเกี่ยวกับเว็บเซอร์วิซโดยเฉพาะ แต่แบบรูป Single Access Point, Check Point, Authorization และ Security Session เกี่ยวข้องกับการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ซึ่งผู้วิจัยเห็นว่าสามารถใช้เป็นแนวทางวิจัยได้ จึงเลือกแบบรูปชุดนี้ อย่างไรก็ตามแบบรูปเหล่านี้ถูกกำหนดในลักษณะของแนวปฏิบัติด้านความมั่นคง แต่ไม่ได้อยู่ในรูปของแบบจำลองที่จะสามารถนำไปใช้ในการออกแบบระบบได้โดยตรง ดังนั้นผู้วิจัยจะเสนอเป็นแบบจำลองสำหรับการประยุกต์ใช้แบบรูปเหล่านี้

โดยสรุปผู้วิจัยเลือกแบบรูป ได้แก่ Authentication Enforcer, Authorization Enforcer, Single Access Point, Check Point, Authorization และ Security Session มาพัฒนา

3.3 จัดหมวดหมู่แบบรูป

แบบรูปความมั่นคงที่เลือกมาในข้อที่ 3.2 สามารถนำมาจัดเป็นหมวดหมู่ได้สามหมวดหมู่ ดังตารางที่ 3.3

ตารางที่ 3.3 การแบ่งหมวดหมู่ของแบบรูปความมั่นคง

หมวดหมู่	แบบรูปความมั่นคง
การพิสูจน์ตัวจริง	Check Point [5], Single Access Point [5] และ Authentication Enforcer [4]
การพิสูจน์สิทธิ์	Check Point [5], Single Access Point [5], Authorization [5] และ Authorization Enforcer [4]
เซสชันด้านความมั่นคง	Check Point [5], Single Access Point [5] และ Security Session [5]

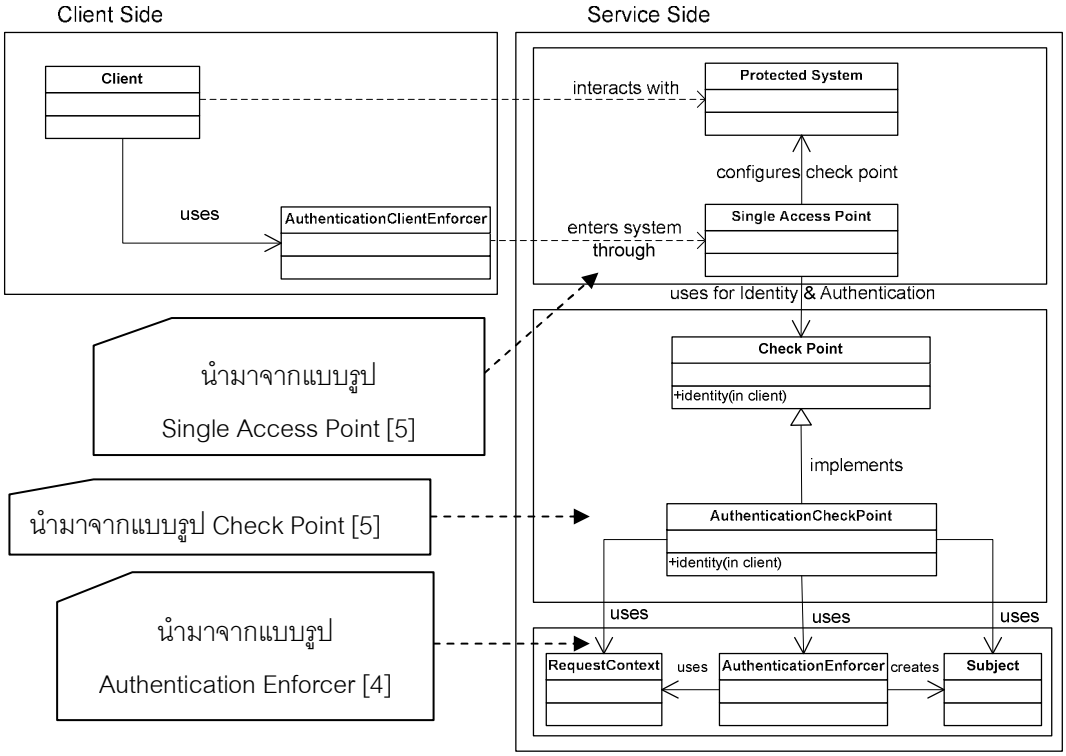
จากตารางที่ 3.3 จะเห็นว่าแบบรูป Single Access Point สามารถใช้ได้ทั้งการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และการจัดการเซสชันด้านความมั่นคง จึงจัดวางไว้อยู่ทั้งสามหมวดหมู่ ส่วน Check Point จะถูกแนะนำให้ใช้คู่กับ Single Access Point จึงจัดอยู่ทั้งสามหมวดหมู่เช่นเดียวกัน

3.4 ผสมผสานแบบรูปในหมวดหมู่เดียวกัน

จากตารางที่ 3.3 แบบรูปที่อยู่ในหมวดหมู่เดียวกัน จะนำมาผสมผสาน เพื่อจุดประสงค์ในการนำไปใช้กับความมั่นคงสำหรับเว็บเซอร์วิซ ซึ่งจะเป็นการผสมผสานในฝั่งเว็บเซอร์วิซ โดยมีรายละเอียดดังนี้

3.4.1 การผสมผสานแบบรูปในหมวดหมู่การพิสูจน์ตัวจริง

จากตารางที่ 3.3 ทำการผสมผสานแบบรูปในหมวดหมู่การพิสูจน์ตัวจริงซึ่งจะได้ภาพที่ 3.1



ภาพที่ 3.1 แผนภาพคลาสของการพิสูจน์ตัวจริง หลังจากผสมผสานแบบรูป

ในหนังสือของ Schumacher และคณะ [5] มีแบบรูป Single Access Point กับ Check Point ซึ่งใช้ได้กับการพิสูจน์ตัวจริง ร่วมกับการพิสูจน์สิทธิ์พร้อมกัน ขณะเดียวกันหนังสือของ Steel

และคณะ [4] มีแบบรูปคือ Authentication Enforcer ซึ่งใช้กับการพิสูจน์ตัวจริงอย่างเดียว แต่ใช้ในโปรแกรมประยุกต์บนเว็บ ไม่ใช้กับเว็บเซอร์วิซ ผู้วิจัยเห็นว่าสามารถนำแบบรูปของทั้งสองเล่มมาประยุกต์ร่วมกันได้ จึงจะแสดงว่าสามารถนำหลาย ๆ แบบรูปมาผสมผสานกัน เพื่อจุดประสงค์ในการพิสูจน์ตัวจริงในเว็บเซอร์วิซ

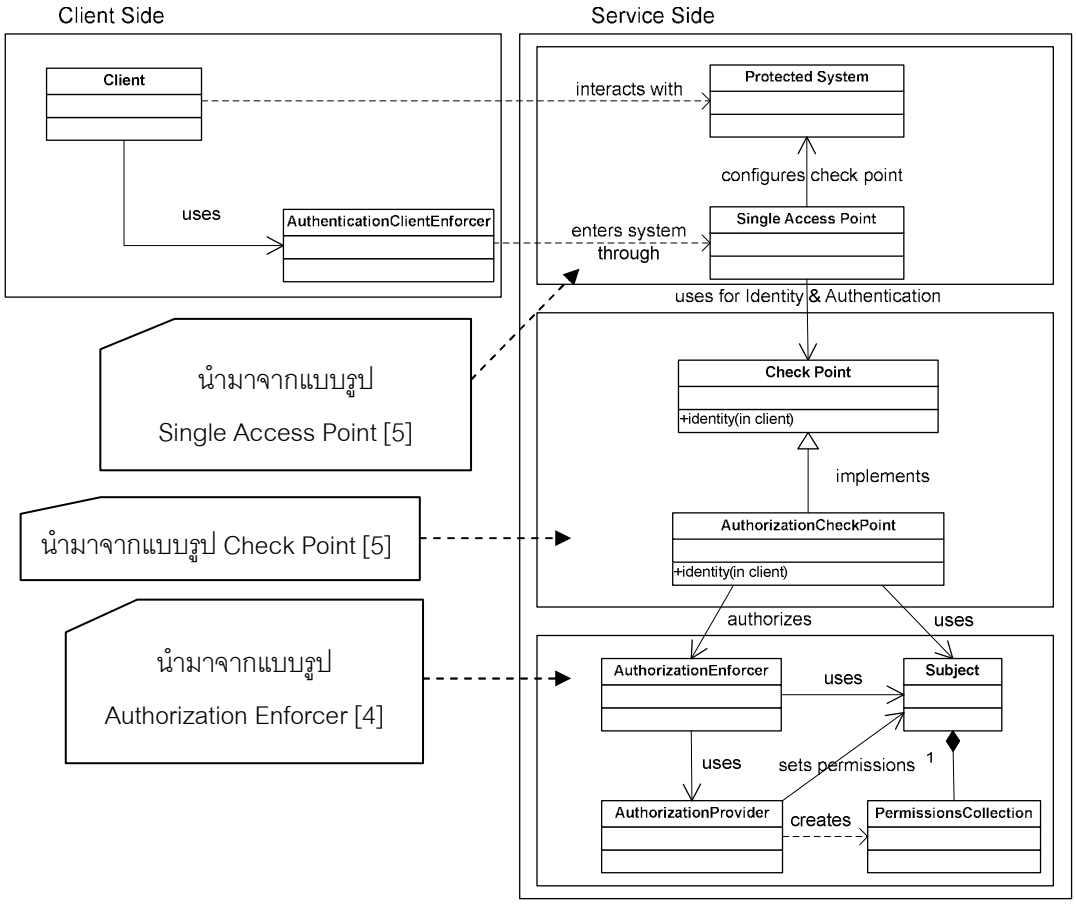
ในภาพที่ 3.1 แสดงแบบรูป Authentication ซึ่งแบ่งเป็น 2 ฝั่งคือ ฝั่งไคลเอนต์ (Client Side) กับฝั่งเซอร์วิซ (Service Side) โดย Protected System [5] เป็นระบบภายในเว็บเซอร์วิซซึ่งถูกป้องกันไม่ให้เข้าถึงโดยตรง แต่ให้เข้ามาทาง Single Access Point [5] ซึ่งเป็นทางเข้าเดียวเท่านั้น โดยสามารถนำไฟร์วอลล์มาประยุกต์ใช้งานจริงได้

สำหรับคำร้องขอพิสูจน์ตัวจริงของไคลเอนต์ต้องผ่าน Check Point [5] จุดนี้จุดเดียว ซึ่งเป็นจุดศูนย์รวมการพิสูจน์ตัวจริงทั้งหมด และมี AuthenticationCheckPoint มาอิมพลีเมนต์ใช้งานจริงซึ่งจะไปเรียกใช้คลาส RequestContext [4], AuthenticationEnforcer [4] และ Subject [4] โดยที่ RequestContext คือตัวแทนคำร้องขอที่มีข้อมูลระบุตัวตน ได้แก่ ชื่อผู้ใช้ และรหัสผ่าน ส่วน AuthenticationEnforcer คือกลไกการพิสูจน์ตัวจริงตามมาตรฐาน WS-Security 2004 (ในข้อที่ 2.1.2) และ Subject เป็นตัวแทนของผู้ใช้งานที่ผ่านการพิสูจน์ตัวจริง

ส่วนฝั่งไคลเอนต์ มีคลาส AuthenticationClientEnforcer ซึ่งจะใช้มาตรฐานของ WS-Security 2004 สำหรับส่งคำร้องขอพิสูจน์ตัวจริงไปยังฝั่งเซอร์วิซ

3.4.2 การผสมผสานแบบรูปในหมวดหมู่การพิสูจน์สิทธิ์

จากตารางที่ 3.3 ทำการผสมผสานแบบรูปในหมวดหมู่การพิสูจน์สิทธิ์ซึ่งจะได้ภาพที่ 3.2



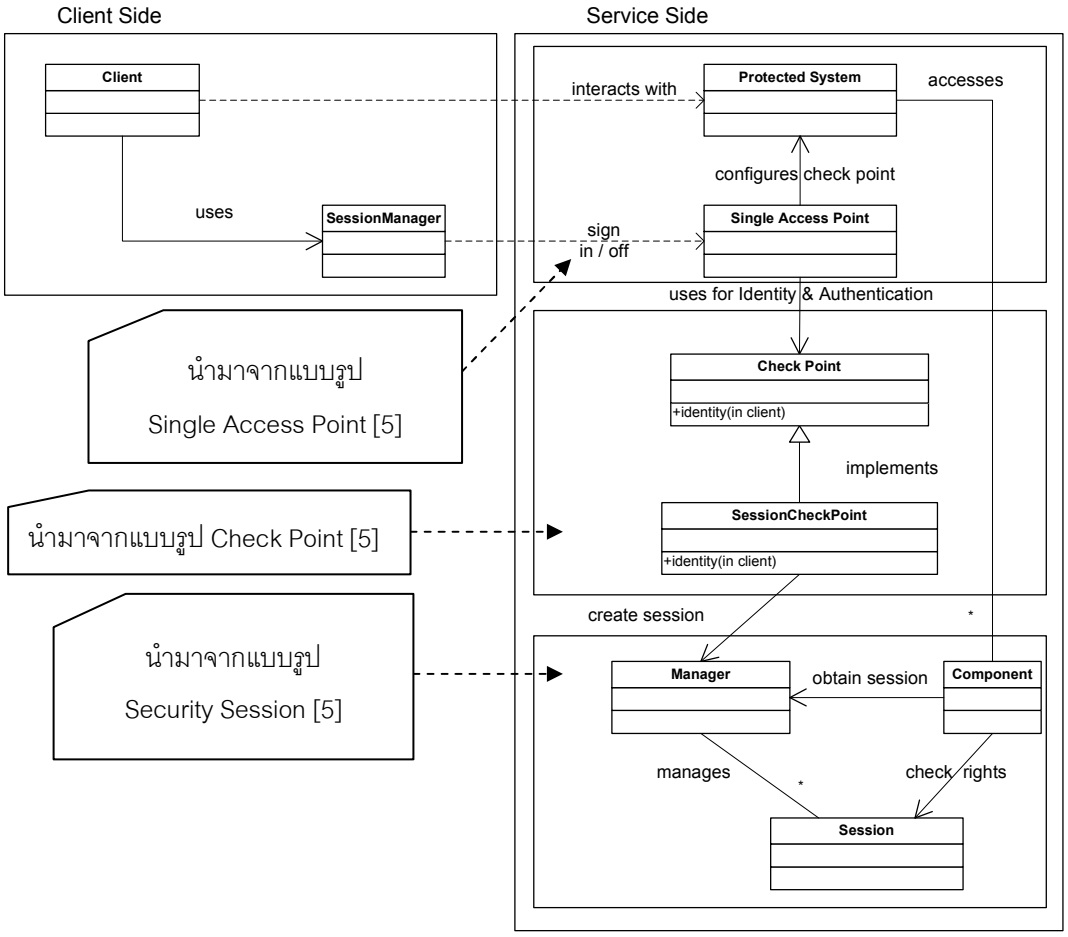
ภาพที่ 3.2 แผนภาพคลาสของการพิสูจน์สิทธิ์ หลังจากผสมผสานแบบรูป

ในภาพที่ 3.2 แนวคิดจะคล้ายกับการผสมผสานแบบรูปในหมวดหมู่การพิสูจน์ตัวจริง โดยจุดที่แตกต่างกันคือใช้ AuthorizationCheckPoint มาอิมพลีเมนต์ใช้งานจริงซึ่งจะไปเรียกใช้คลาส AuthorizationEnforcer [4], และ Subject [4] โดยที่ AuthorizationEnforcer [4] จะเป็นโค้ดส่วนการพิสูจน์สิทธิ์ ส่วน AuthorizationProvider [4] ทำหน้าที่ในการกำหนดสิทธิ์ในการเข้าใช้ทรัพยากรและบันทึกสิทธิ์นั้นใน PermissionsCollection [4]

ส่วนฝั่งไคลเอนต์ มีคลาส AuthenticationClientEnforcer แบบเดียวกับภาพที่ 3.1 เพราะไคลเอนต์ต้องผ่านการพิสูจน์ตัวจริงก่อนได้รับการตรวจสอบสิทธิ์เพื่อใช้งานภายในระบบ

3.4.3 การผสมผสานแบบรูปในหมวดหมู่เซสชันด้านความมั่นคง

จากตารางที่ 3.3 ทำการผสมผสานแบบรูปในหมวดหมู่เซสชันด้านความมั่นคงซึ่งจะได้ภาพที่ 3.3



ภาพที่ 3.3 แผนภาพคลาสของเซสชันด้านความมั่นคง หลังจากผสมผสานแบบรูป

ในภาพที่ 3.3 แนวคิดจะเหมือนกับการผสมผสานแบบรูปในหมวดหมู่การพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ โดยจุดที่แตกต่างกันคือใช้ SessionCheckPoint มาอิมพลีเมนต์ใช้งานจริงซึ่งจะไปสร้างคลาส Manager [4] ซึ่งจะทำหน้าที่จัดการ Session [4] ที่ทำหน้าที่เก็บข้อมูลผู้ใช้งาน สิทธิ์ในการใช้ทรัพยากร และแบ่งปันข้อมูลร่วมกันภายในทรัพยากรระบบ โดยที่ Component [5] จะหมายถึงทรัพยากรใด ๆ ภายในระบบที่จะมาเรียกใช้ Session ผ่านทาง Manager และตรวจสอบสิทธิ์และใช้งานต่าง ๆ ผ่านทาง Session ส่วนฝั่งไคลเอนต์ มีคลาส SessionManager เพื่อทำหน้าที่จัดการเซสชันด้านความมั่นคง

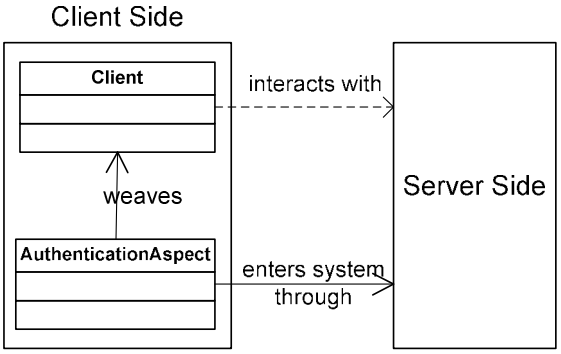
3.4.4 ข้อดีของการผสมผสานแบบรูปในฝั่งเซอร์วิซ

สำหรับการผสมสานแบบรูปจากหนังสือสองเล่มเข้าด้วยกัน มีข้อดีดังต่อไปนี้

1) เนื่องจากระบบเว็บเซอร์วิซ จะประกอบด้วยเซอร์วิซย่อย ๆ อยู่ภายใน การที่ไคลเอนต์ต้องทำการติดต่อเข้ามาทางช่องทางเดียว ทำให้ง่ายต่อการควบคุม ส่งผลให้กลไกพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และการจัดการเซสชันด้านความมั่นคง จะมีอยู่เพียงที่เดียว ทำให้สภาพมอดูลาร์ฝั่งเซอร์วิซดีขึ้น ง่ายต่อการแก้ไขโค้ด การบำรุงรักษาทำได้ง่ายขึ้น และสามารถนำโค้ดกลับมาใช้ในหลายเซอร์วิซย่อยภายในซึ่งช่วยลดการเขียนโค้ดที่ซ้ำซ้อน
2) ส่วนโค้ดฝั่งไคลเอนต์ ผู้วิจัยได้นำวิธีเชิงแง่มุมมาใช้ ทำให้มีสภาพมอดูลาร์ที่ดีขึ้น ซึ่งจะต่างกับงานวิจัย ของ Edge และ Mitropoulos [11, 18] กับหนังสือทั้งสองเล่ม ที่ไม่ได้กล่าวถึงแบบรูปในฝั่งไคลเอนต์
3) ในงานวิจัยใช้ Check Point เป็นอินเทอร์เฟซ และจะมีคลาส AuthenticationCheckPoint AuthorizationEnforcer และ SessionCheckPoint มาอิมพลีเมนต์ไปใช้งานจริง ดังนั้นในอนาคตสามารถมีคลาสมาอิมพลีเมนต์ไปใช้กับความมั่นคงด้านอื่น ๆ ตัวอย่างเช่น การเข้ารหัสข้อมูล เป็นต้น

3.5 นำวิธีเชิงแง่มุมมาใช้ร่วมกับแบบรูป

จากแบบรูปทั้งสามหมวดหมู่ที่เสนอในข้อที่ 3.4 จะนำวิธีเชิงแง่มุมมาใช้ เพื่อปรับปรุงสภาพมอดูลาร์ในโค้ดความมั่นคงบนฝั่งไคลเอนต์ สุดท้ายจึงได้แบบรูปใหม่ขึ้นมา ซึ่งมีรายละเอียดดังนี้

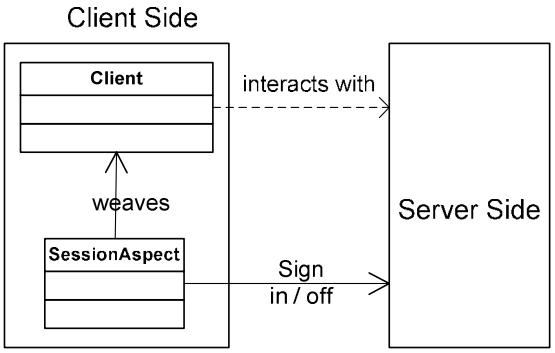


ภาพที่ 3.4 แผนภาพคลาสของการพิสูจน์ตัวจริงกับการพิสูจน์สิทธิ์ในฝั่งไคลเอนต์ หลังใช้วิธีเชิงแง่มุม

จากงานวิจัยของ Edge และ Mitropoulos [11, 18] ได้ใช้แบบรูป Authentication Enforcer บนโปรแกรมประยุกต์บนเว็บโดยจะวีฟแอสเปกต์ในฝั่งเซิร์ฟเวอร์ เนื่องจากมีแบบรูป FrontController หรือ ApplicationController อยู่ในฝั่งเซิร์ฟเวอร์ทำหน้าที่เป็นไคลเอนต์ จึงทำการ

วีฟในฝั่งนี้ แต่ในระบบเว็บเซอร์วิซไม่ได้มีแบบรูปชุดนี้ในฝั่งไคลเอนต์ จึงได้วีฟ AuthenticationAspect เข้าไปยังฝั่งไคลเอนต์แทน

ในภาพที่ 3.4 แสดงการนำวิธีเชิงแง่มุมมาใช้ ด้วยการนำเสนอคลาส AuthenticationAspect ซึ่งเป็นเพียงคลาสเดียวที่นำวิธีเชิงแง่มุมมาใช้ในแบบรูป โดยมันจะโอนย้ายการต่อประกบที่สูง ๆ และความเชื่อมแน่นต่ำ ๆ ที่เกิดขึ้นภายในโปรแกรมประยุกต์หลักมาไว้ที่ตัวโค้ดของมันเอง ทำให้โค้ดฝั่งไคลเอนต์มีสภาพมอดูลาร์ดีขึ้น ซึ่งแผนภาพคลาสนี้จะใช้ทั้งการพิสูจน์ตัวจริงกับการพิสูจน์สิทธิ์ ในฝั่งไคลเอนต์



ภาพที่ 3.5 แผนภาพคลาสของเซสชันด้านความมั่นคงในฝั่งไคลเอนต์ หลังใช้วิธีเชิงแง่มุม

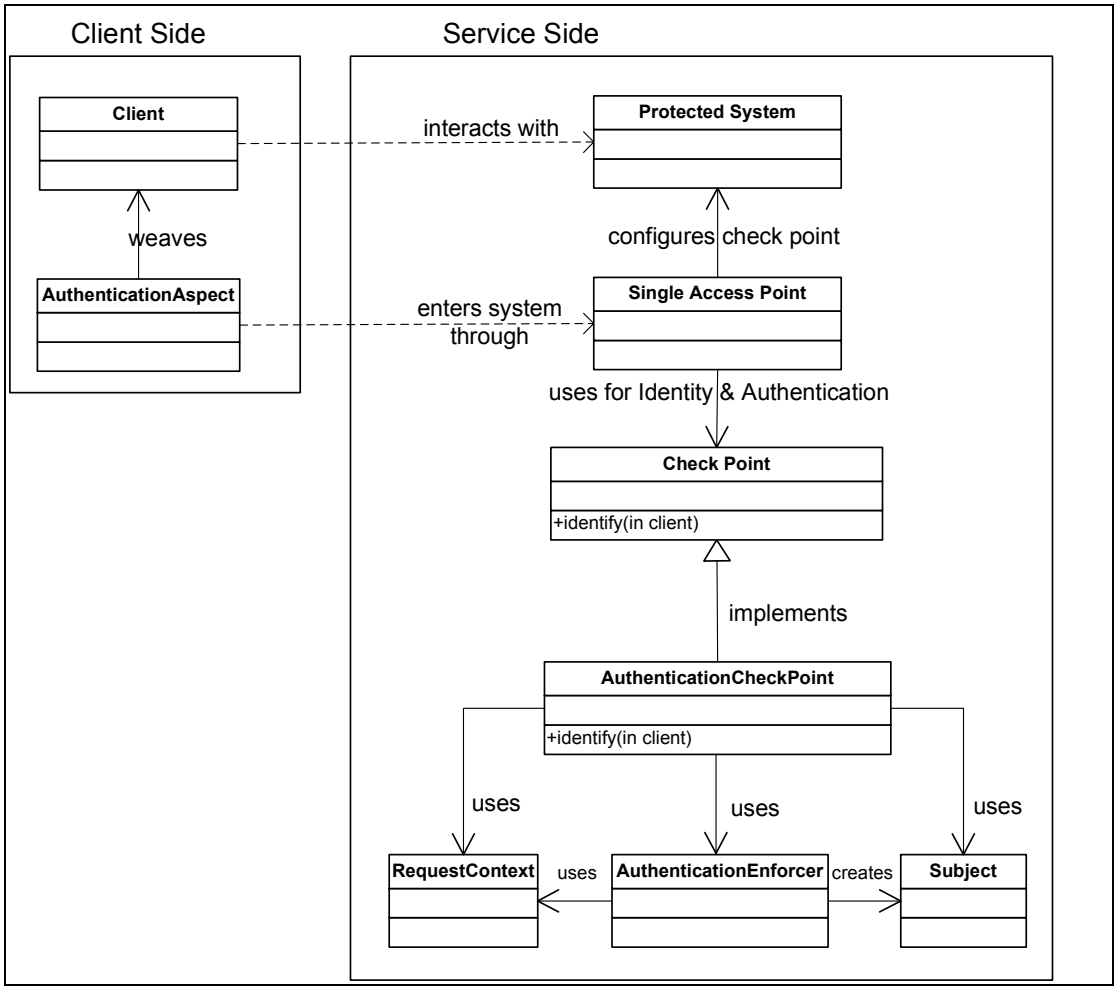
ในภาพที่ 3.5 จะแสดงแผนภาพคลาสของการทำเซสชันด้านความมั่นคง ในฝั่งไคลเอนต์ ซึ่งใช้หลักการเดียวกับภาพที่ 3.4 โดยนำเสนอคลาส SessionAspect ซึ่งใช้วิธีเชิงแง่มุม มาทำการโอนย้ายการต่อประกบที่สูง ๆ และความเชื่อมแน่นต่ำ ๆ ที่เกิดขึ้นภายในโปรแกรมประยุกต์หลักมาไว้ที่ตัวโค้ดของมันเอง ทำให้โค้ดฝั่งไคลเอนต์มีสภาพมอดูลาร์ดีขึ้น

3.6 สรุป

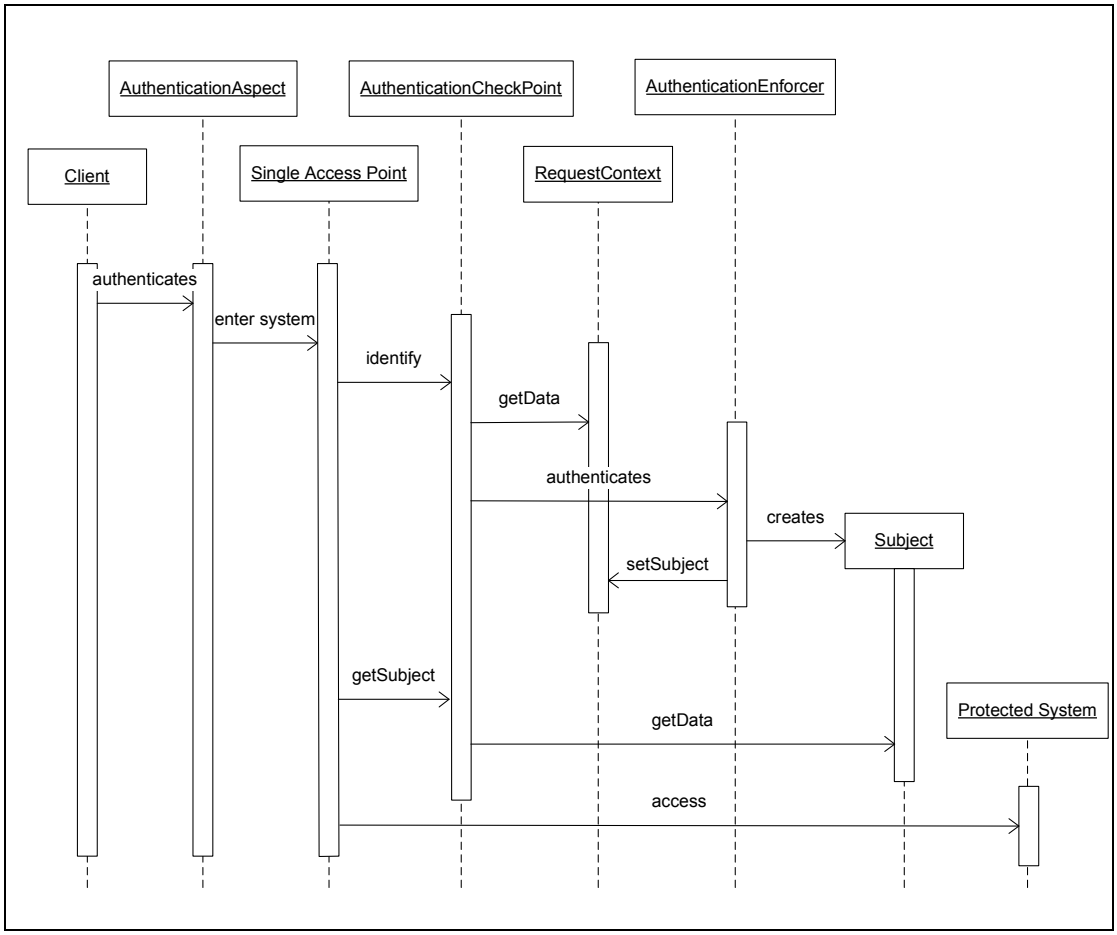
จากแนวคิดและขั้นตอนวิจัยที่กล่าวมาทั้งหมด สามารถสรุปออกมาเป็นแบบรูปที่งานวิจัยจะนำเสนอ คือ Authentication, Authorization และ Security Session พร้อมทั้งอธิบายองค์ประกอบของแต่ละแบบรูป ซึ่งได้มาจากการวิเคราะห์และสรุปมาจากหนังสือสองเล่ม และใช้แนวคิดเชิงแง่มุมมาปรับปรุงเฉพาะในส่วนโค้ดหลัก เช่นเดียวกับงานวิจัย [11, 18] ตามที่กล่าวมาในข้อที่ 2.2.2 โดยมีรายละเอียดดังนี้

3.6.1 **แบบรูป** Authentication

ในภาพที่ 3.6 ได้แสดงแบบรูป Authentication และในภาพที่ 3.7 ได้แสดงลำดับการทำงานของการพิสูจน์ตัวจริง ซึ่งเป็นผลจากแนวคิดและขั้นตอนวิจัย โดยจะอธิบายองค์ประกอบของแบบรูปในตารางที่ 3.4



ภาพที่ 3.6 แผนภาพคลาสของแบบรูป Authentication



ภาพที่ 3.7 แผนภาพลำดับของแบบรูป Authentication

ตารางที่ 3.4 องค์ประกอบของแบบรูป Authentication

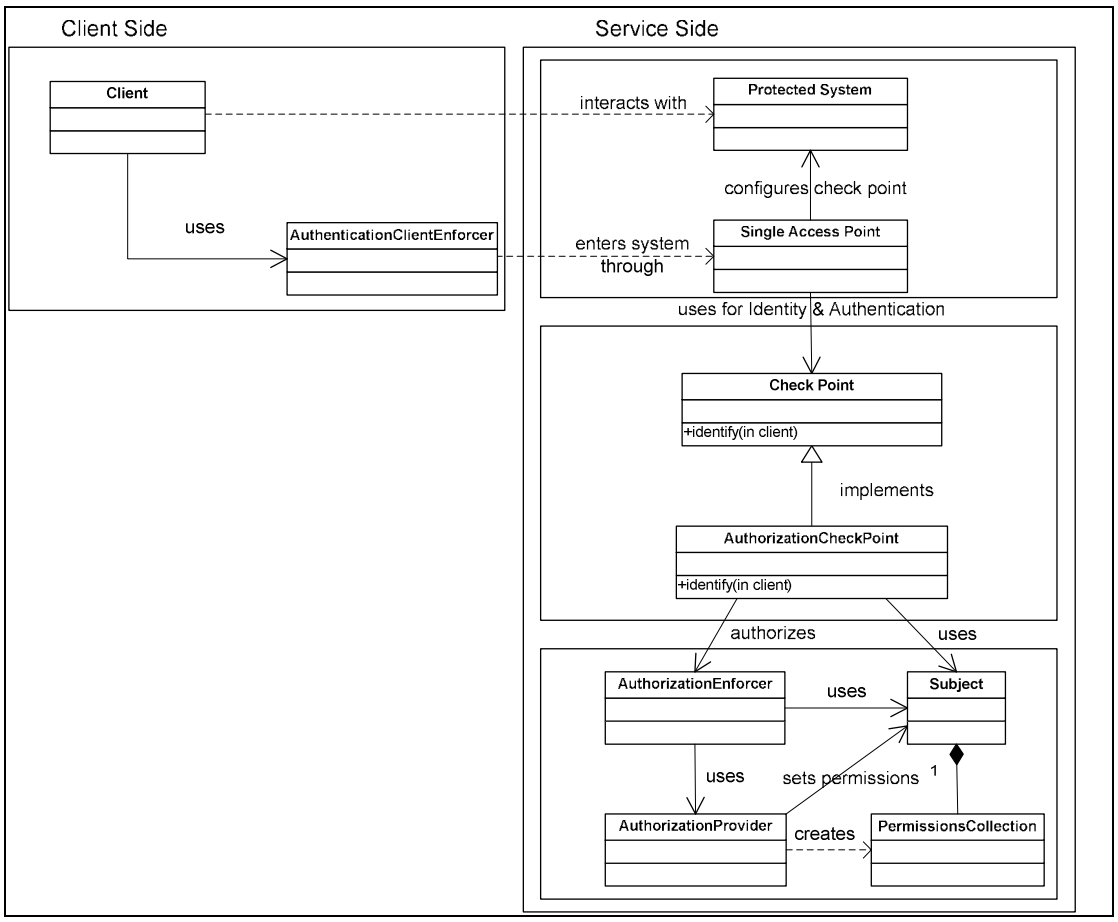
องค์ประกอบแบบรูป	**รายละเอียด**
ชื่อแบบรูป	Authentication
จุดประสงค์	ใช้พิสูจน์ตัวจริงของไคลเอนต์ ก่อนใช้งานระบบเว็บเซอร์วิซ
บริบท	ใช้กับระบบเว็บเซอร์วิซที่ต้องการให้พิสูจน์ตัวจริงของไคลเอนต์ ก่อนใช้งานระบบครั้งแรก
ปัญหา	การจัดการแต่ละคำร้องขอพิสูจน์ตัวจริงของไคลเอนต์ จะมีความแตกต่างกันไป ส่งผลให้โค้ดเขียนซ้ำ ๆ กัน ในหลาย ๆ ส่วน จึงทำให้กลไกการพิสูจน์ตัวจริงยากต่อการเปลี่ยนแปลงแก้ไข และบำรุงรักษายาก ทั้งในฝั่งเซอร์วิซกับไคลเอนต์

ตารางที่ 3.4 องค์ประกอบของแบบรูป Authentication (ต่อ)

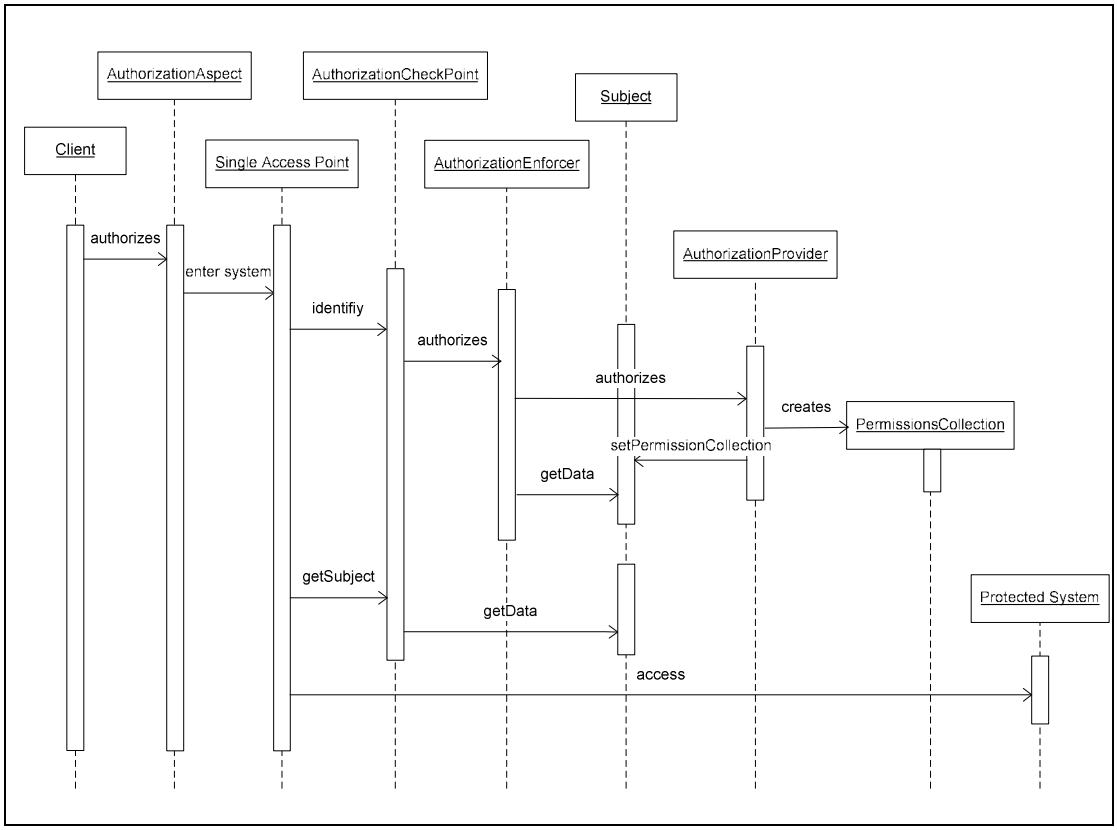
องค์ประกอบแบบรูป	**รายละเอียด**
การแก้ปัญหา	แนะนำแนวทางแก้ปัญหา โดยสร้างศูนย์กลางพิสูจน์ตัวจริงอยู่ที่เดียว เพื่อลดจำนวนโค้ดพิสูจน์ตัวจริงในฝั่งเซอร์วิซ พร้อมทั้งเสนอการโปรแกรมเชิงแง่มุมมาใช้ในฝั่งไคลเอนต์
โครงสร้าง	อธิบายตามภาพที่ 3.6 และภาพที่ 3.7
ผลที่ตามมา	การรวมกลไกพิสูจน์ตัวจริงไว้ที่ศูนย์กลางจุดเดียว ทำให้ง่ายต่อการทดสอบและพัฒนา พร้อมทั้งยืดหยุ่นในการปรับเปลี่ยนโค้ด นอกจากนั้นยังสามารถเพิ่มหน้าที่ความมั่นคงอื่น ๆ ได้ เช่น ลงบันทึก (Log) หรือ การตรวจสอบการบุกรุก เป็นต้น
แบบรูปที่เกี่ยวข้อง	Authorization และ Security Session

3.6.2 แบบรูป Authorization

ในภาพที่ 3.8 ได้แสดงแบบรูป Authorization และในภาพที่ 3.9 ได้แสดงลำดับการทำงานของการพิสูจน์สิทธิ์ ซึ่งเป็นผลจากแนวคิดและขั้นตอนวิจัย โดยจะอธิบายองค์ประกอบของแบบรูปในตารางที่ 3.5



ภาพที่ 3.8 แผนภาพคลาสของแบบรูป Authorization



ภาพที่ 3.9 แผนภาพลำดับของแบบรูป Authorization

ตารางที่ 3.5 องค์ประกอบของแบบรูป Authorization

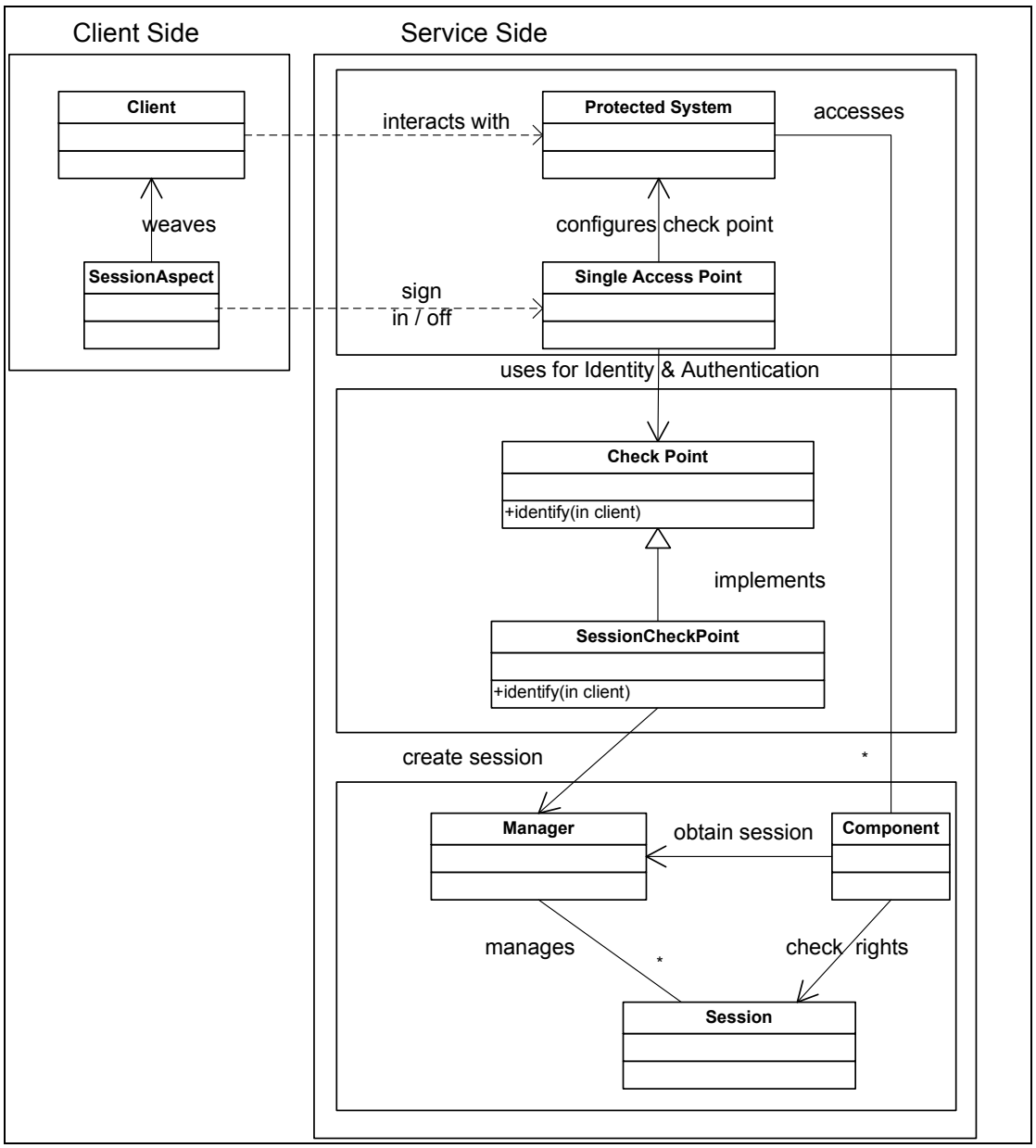
องค์ประกอบแบบรูป	**รายละเอียด**
ชื่อแบบรูป	Authorization
จุดประสงค์	ใช้ตรวสอบสิทธิ์ของไคลเอนต์เพื่ออนุญาตให้เข้าใช้ทรัพยากรภายในระบบเว็บเซอร์วิซ
บริบท	ใช้กับระบบเว็บเซอร์วิซที่ต้องการป้องกันการเข้าใช้ทรัพยากรภายในระบบจากคำร้องขอที่ไม่มีสิทธิ์
ปัญหา	ไคลเอนต์ที่แตกต่างกัน จะมีสิทธิ์ในการเข้าใช้ทรัพยากรภายในระบบแตกต่างกันขึ้นกับสิทธิ์ที่ได้รับ ส่งผลให้โค้ดเขียนซ้ำ ๆ กัน ในหลาย ๆ ส่วนของฝั่งเซอร์วิซ จึงทำให้กลไกการพิสูจน์สิทธิ์ใช้งานยากต่อการเปลี่ยนแปลงแก้ไข และบำรุงรักษายาก ในฝั่งเซอร์วิซ

ตารางที่ 3.5 องค์ประกอบของแบบรูป Authorization (ต่อ)

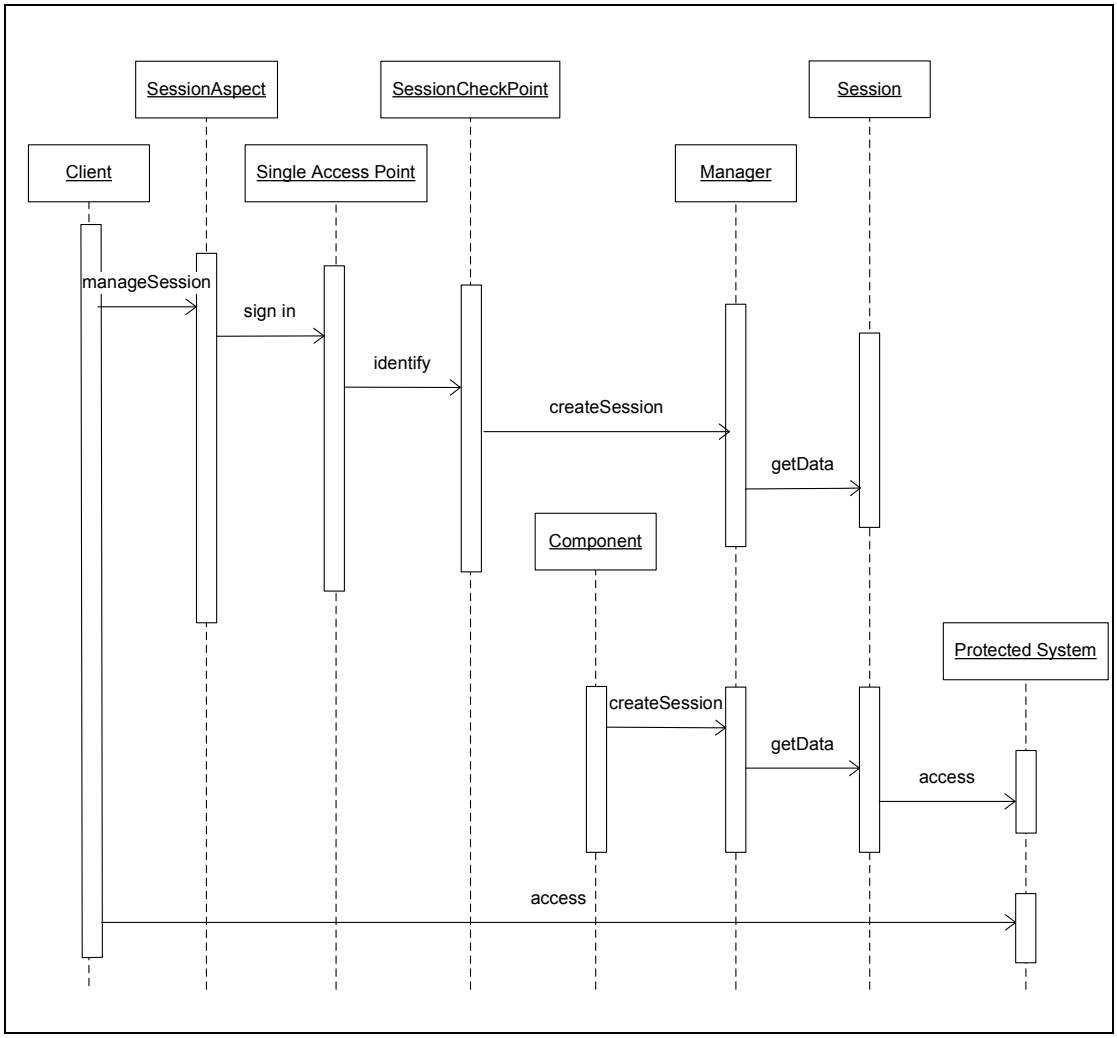
องค์ประกอบแบบรูป	**รายละเอียด**
การแก้ปัญหา	แนะนำแนวทางแก้ปัญหา โดยสร้างศูนย์กลางในการตรวจสอบสิทธิ์และการอนุญาตเข้าใช้งานทรัพยากรภายในระบบ ให้มีโค้ดอยู่ที่เดียว เพื่อลดจำนวนโค้ดในฝั่งเซอร์วิซ
โครงสร้าง	อธิบายตามภาพที่ 3.8 และภาพที่ 3.9
ผลที่ตามมา	การรวมกลไกการตรวจสอบสิทธิ์และการอนุญาติไว้ที่ศูนย์กลางจุดเดียว ทำให้ง่ายต่อการทดสอบและพัฒนา พร้อมทั้งยืดหยุ่นในการปรับเปลี่ยนโค้ด นอกจากนั้นยังสามารถเพิ่มหน้าที่ความมั่นคงอื่น ๆ ได้ เช่น ลงบันทึก (Log) หรือ การตรวจสอบการบุกรุก เป็นต้น
แบบรูปที่เกี่ยวข้อง	Authentication และ Security Session

3.6.3 แบบรูป Security Session

ในภาพที่ 3.10 ได้แสดงแบบรูป Security Session และในภาพที่ 3.11 ได้แสดงลำดับการทำงานของการจัดการเซสชันด้านความมั่นคง ซึ่งเป็นผลจากแนวคิดและขั้นตอนวิจัย โดยจะอธิบายองค์ประกอบของแบบรูปในตารางที่ 3.6



ภาพที่ 3.10 แผนภาพคลาสของแบบรูป Security Session



ภาพที่ 3.11 แผนภาพลำดับของแบบรูป Security Session

ตารางที่ 3.6 องค์ประกอบของแบบรูป Security Session

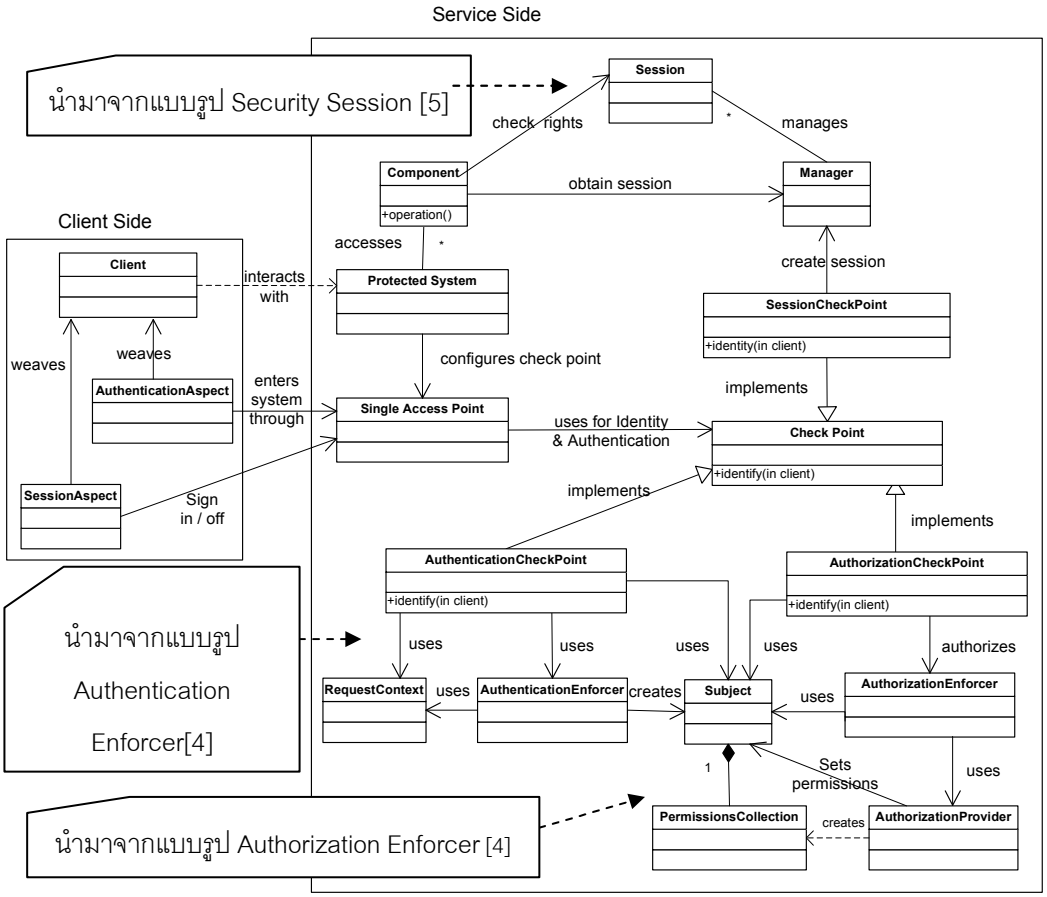
องค์ประกอบแบบรูป	**รายละเอียด**
ชื่อแบบรูป	Security Session
จุดประสงค์	ต้องการจดจำการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์เพื่อใช้งานทรัพยากรภายในระบบตลอดช่วงเวลาที่ไคลเอนต์ติดต่อสื่อสารเข้ามา
บริบท	ใช้กับระบบเว็บเซอร์วิซที่ต้องคงไว้ซึ่งช่วงเวลาของการติดต่อสื่อสารระหว่างไคลเอนต์กับระบบ

ตารางที่ 3.6 องค์ประกอบของแบบรูป Security Session (ต่อ)

องค์ประกอบแบบรูป	**รายละเอียด**
ปัญหา	การที่ไคลเอนต์ต้องยืนยันและทำการพิสูจน์ตัวจริงทุกครั้ง และฝั่งเซอร์วิซเองตรวจสอบสิทธิ์ในการเข้าถึงทุกครั้งเมื่อมีคำร้องขอจากไคลเอนต์เข้ามา จะสร้างความรำคาญต่อผู้ใช้งานและผู้พัฒนาระบบ อีกทั้งยังมีปัญหาเรื่องการมีการแบ่งปันข้อมูลความต้องการความมั่นคงภายในระบบ
การแก้ปัญหา	แนะนำแนวทางแก้ปัญหา โดยสร้างศูนย์กลางการจัดการเซสชันให้อยู่ที่เดียว เพื่อลดจำนวนโค้ดร้องขอพิสูจน์ตัวจริงในฝั่งไคลเอนต์ พร้อมทั้งเสนอการโปรแกรมเชิงแง่มุมมาใช้ในฝั่งไคลเอนต์
โครงสร้าง	อธิบายตามภาพที่ 3.10 และภาพที่ 3.11
ผลที่ตามมา	มีระบบจดจำการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์เพื่อใช้งานทรัพยากรภายในระบบตลอดช่วงเวลาที่ไคลเอนต์ติดต่อสื่อสารเข้ามา อีกทั้งสามารถแบ่งปันข้อมูลความต้องการความมั่นคงภายในระบบ
แบบรูปที่เกี่ยวข้อง	Authentication และ Authorization

3.6.4 แผนภาพรวมในฝั่งเซอร์วิซ

เนื่องจากการผสมสานแบบรูปในฝั่งเซอร์วิซ ได้ใช้หลายแบบรูปร่วมกัน ได้แก่ Single Access Point [5] กับ Check Point [5] ดังนั้นเพื่อให้ง่ายและเห็นภาพชัดเจน จึงแสดงแผนภาพรวมในฝั่งเซอร์วิซของทั้ง 3 แบบรูป ดังแสดงในภาพที่ 3.12



ภาพที่ 3.12 แผนภาพรวมทั้งของแบบรูป Authentication, Authorization และ Security Session ในฝั่งเซอร์วิซ

บทที่ 4
การดำเนินการทดลอง

ในบทนี้จะกล่าวถึงการนำแบบรูปที่ได้จากบทที่ 3 มาทำการทดลอง โดยจะกล่าวถึงการเลือกมาตรฐานความมั่นคงของเว็บเซอร์วิซและเทคโนโลยีโอเพนซอร์ซที่จะมาใช้พัฒนาระบบเว็บเซอร์วิซ การเตรียมตัวอย่างการทดลอง การเขียนโค้ดเพิ่มเติม การนำแบบรูปที่ได้ไปประยุกต์ใช้งานจริง สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา ซึ่งมีรายละเอียดดังนี้

4.1 เลือกมาตรฐานความมั่นคงของเว็บเซอร์วิซและเทคโนโลยีโอเพนซอร์ซ

เนื่องด้วยความมั่นคงด้านการพิสูจน์ตัวจริงของเว็บเซอร์วิซมีได้หลายมาตรฐาน ซึ่งในหนังสือ Steel [4] และคณะ ได้แบ่งมาตรฐานความมั่นคงของเว็บเซอร์วิซเป็น 9 มาตรฐาน แต่ผู้วิจัยเลือกมาตรฐาน WS-Security 2004 เพื่อใช้เป็นมาตรฐานสำหรับพัฒนาการพิสูจน์ตัวจริงของแบบรูป Authentication ที่จะนำเสนอ พร้อมทั้งใช้เป็นมาตรฐานในการจดจำตัวตนของผู้ร้องขอและสิทธิ์ในการอนุญาตเข้าใช้ทรัพยากรต่าง ๆ ตลอดช่วงเวลาสื่อสารของแบบรูป Security Session ที่จะนำเสนอ

สำหรับโอเพนซอร์ซที่สนับสนุนมาตรฐานความมั่นคงดังกล่าวมีหลายตัว เช่น JAX-WS, Axis2, WSS4J, Apache CXF และ Spring-WS เป็นต้น ซึ่งรองรับการทำงานกับภาษาจาวา ซึ่งผู้วิจัยเลือก Axis2

ส่วนกรอบงานในการโปรแกรมเชิงแง่มุม มีโอเพนซอร์ซหลายตัวที่สนับสนุนในภาษาจาวา เช่น AspectJ, Spring AOP, JBoss AOP, JAsCo และ PROSE ซึ่งผู้วิจัยเลือก AspectJ

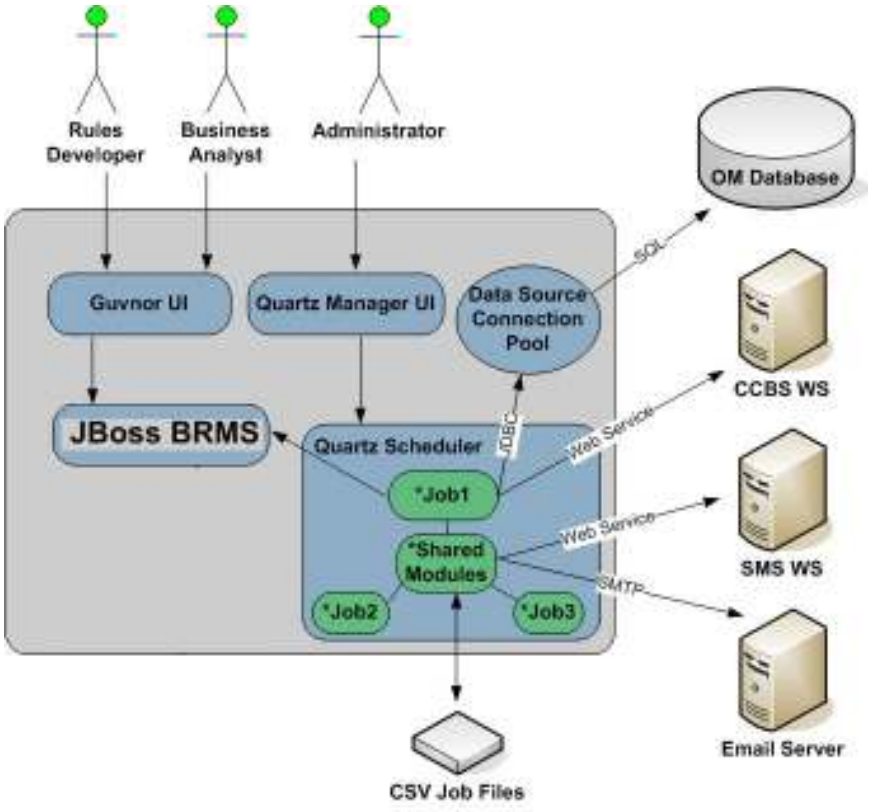
4.2 ตัวอย่างการทดลอง

ผู้วิจัยทำการเลือกระบบเว็บเซอร์วิซของบริษัทโทรคมนาคมแห่งหนึ่งในประเทศไทยมาเป็นหน่วยทดลอง เนื่องด้วยสามารถใช้กับมาตรฐานความมั่นคงของเว็บเซอร์วิซ และเทคโนโลยีโอเพนซอร์ซ ที่เลือกมาจากข้อที่ 4.1 อีกทั้งระบบนี้เป็นระบบที่ใช้งานจริง ผู้วิจัยจึงเลือกใช้ระบบนี้

ระบบเว็บเซอร์วิซที่ใช้เป็นหน่วยทดลองมีชื่อว่า “Service Bundling Job” ซึ่งเป็นส่วนหนึ่งในระบบจัดการสั่งซื้อสินค้าและบริการ (Order Management Sytem) โดยหน้าที่หลักของระบบจะทำการสมัครโปรโมชันให้กับลูกค้ามือถือที่สมัครเข้ามา และจะมีอายุการใช้งานตามโปรโมชันที่สมัคร ตัวอย่างโปรโมชันเช่น โปรโมชันให้ลูกค้าสมัครเข้ามาภายใน 1 เดือน ถ้าสมัครเข้ามาช่วงเวลานี้จะได้รับส่วนลดค่าโทรศัพท์ 15% และจะมีอายุการใช้งานตั้งแต่วันที่เริ่มสมัครไปจนครบอายุ 3 เดือน เป็นต้น

สำหรับระบบนี้จะถูกพัฒนาด้วยระบบเว็บเซอร์วิซ ซึ่งมีความต้องการความมั่นคงได้แก่ การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และจดจำการพิสูจน์ตัวจริงกับการพิสูจน์สิทธิ์ ตลอดช่วงเวลาสื่อสาร แต่ยังไม่ได้ใช้แบบรูปความมั่นคงใด ๆ ทั้งสิ้น

เนื่องด้วยระบบจะมีการเปลี่ยนแปลงความต้องการ (Change Requirement) ค่อนข้างสูง ซึ่งเป็นไปตามลักษณะธุรกิจที่มีการแข่งขันสูงในปัจจุบัน ดังนั้นระบบสามารถเพิ่มและลดจำนวนเซอร์วิซ หรือเปลี่ยนความต้องการได้ตลอดเวลา แต่ช่วงระยะเวลาที่ทำวิจัยขณะนี้ ระบบจะมีการให้บริการฝั่งเซอร์วิซอย่างน้อย 8 เซอร์วิซ และฝั่งไคลเอนต์ 10 เซอร์วิซ โดยมีจำนวนคลาสฝั่งเซอร์วิซ 140 คลาส และคลาสฝั่งไคลเอนต์ 174 คลาส ซึ่งถูกพัฒนาด้วยภาษาจาวา ซึ่งภาพรวมของระบบอธิบายได้ด้วยรูป ภาพที่ 4.1



ภาพที่ 4.1 แผนภาพสถาปัตยกรรมของระบบโดยรวม

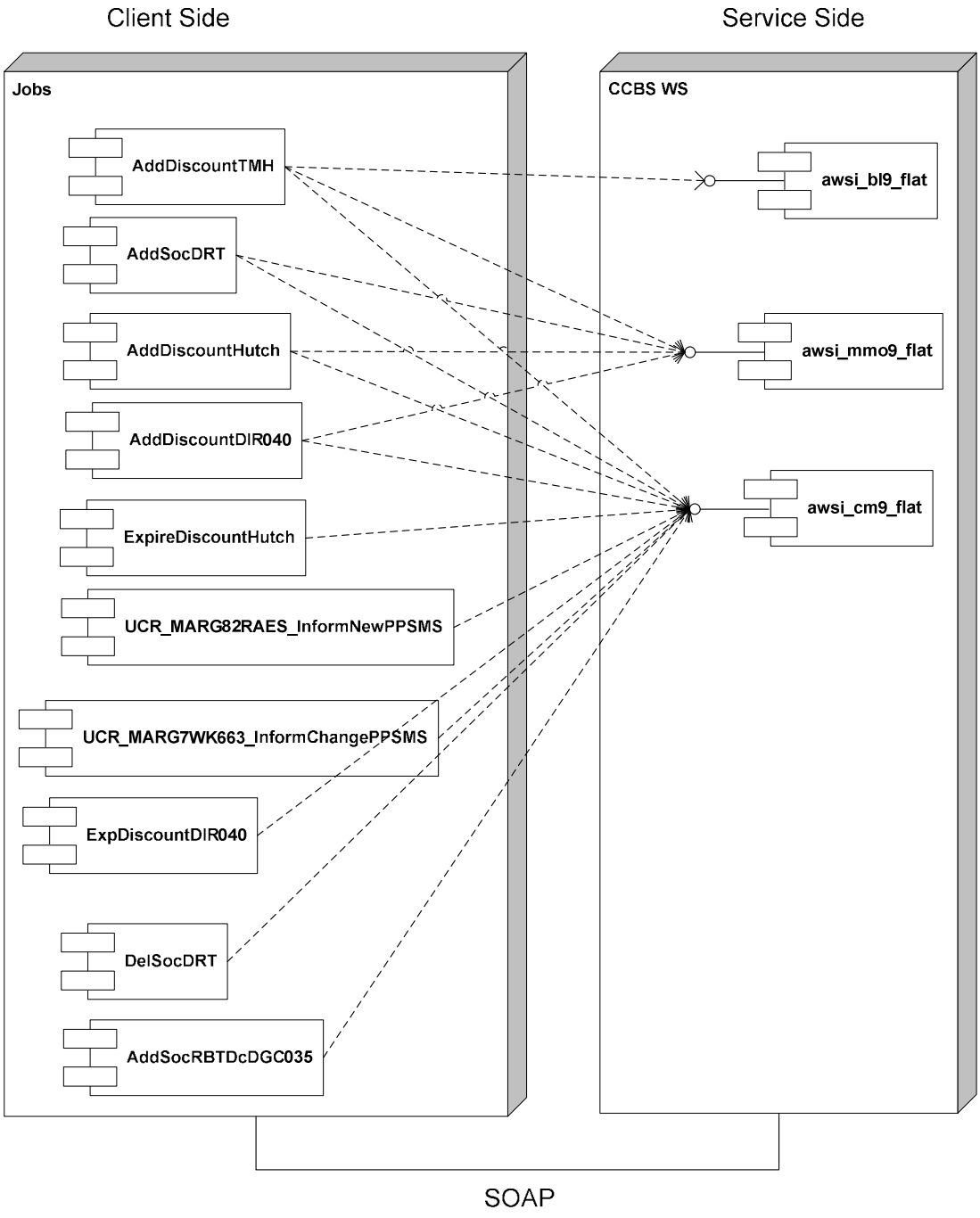
จากภาพที่ 4.1 เป็นแผนภาพสถาปัตยกรรมของระบบโดยรวม ซึ่งสามารถอธิบายแต่ละองค์ประกอบโดยย่อได้ดังนี้

1) JBoss BRMS เป็นผลิตภัณฑ์ของทางบริษัท JBoss ซึ่งเป็นระบบจัดการเงื่อนไขทางธุรกิจ (Business Rule Management System) ซึ่งในกรณีนี้เงื่อนไขตรรกะของโปรโมชันมือถือจะถูกจัดการที่ส่วนนี้
2) Guvnor UI จะเป็นหน้าจอยูสเซอร์อินเทอร์เฟซ (User Interface) เพื่อติดต่อใช้งาน JBoss BRMS
3) Data Source Connection Pool ทำหน้าที่คอยดูแลและจัดการคอนเนคชันสำหรับเชื่อมต่อกับฐานข้อมูล
4) Job คือเว็บเซอร์วิซที่เป็นไคลเอนต์
5) Quartz Scheduler เป็นระบบตั้งเวลาสำหรับการรัน Job (ข้อที่ 4) ให้ทำงานตามวันเวลาที่กำหนด
6) CSV Job Files จะเป็น Input Files ที่อ่านเข้ามาให้แต่ละ Job ได้ทำงาน
7) CCBS WS จะเป็นระบบเว็บเซอร์วิซที่ Job ในข้อที่ 4 จะไปเรียกใช้งาน
8) SMS WS จะเป็นระบบเว็บเซอร์วิซที่จัดการส่ง SMS
9) Email Server จะเป็นระบบจัดการส่งเมล

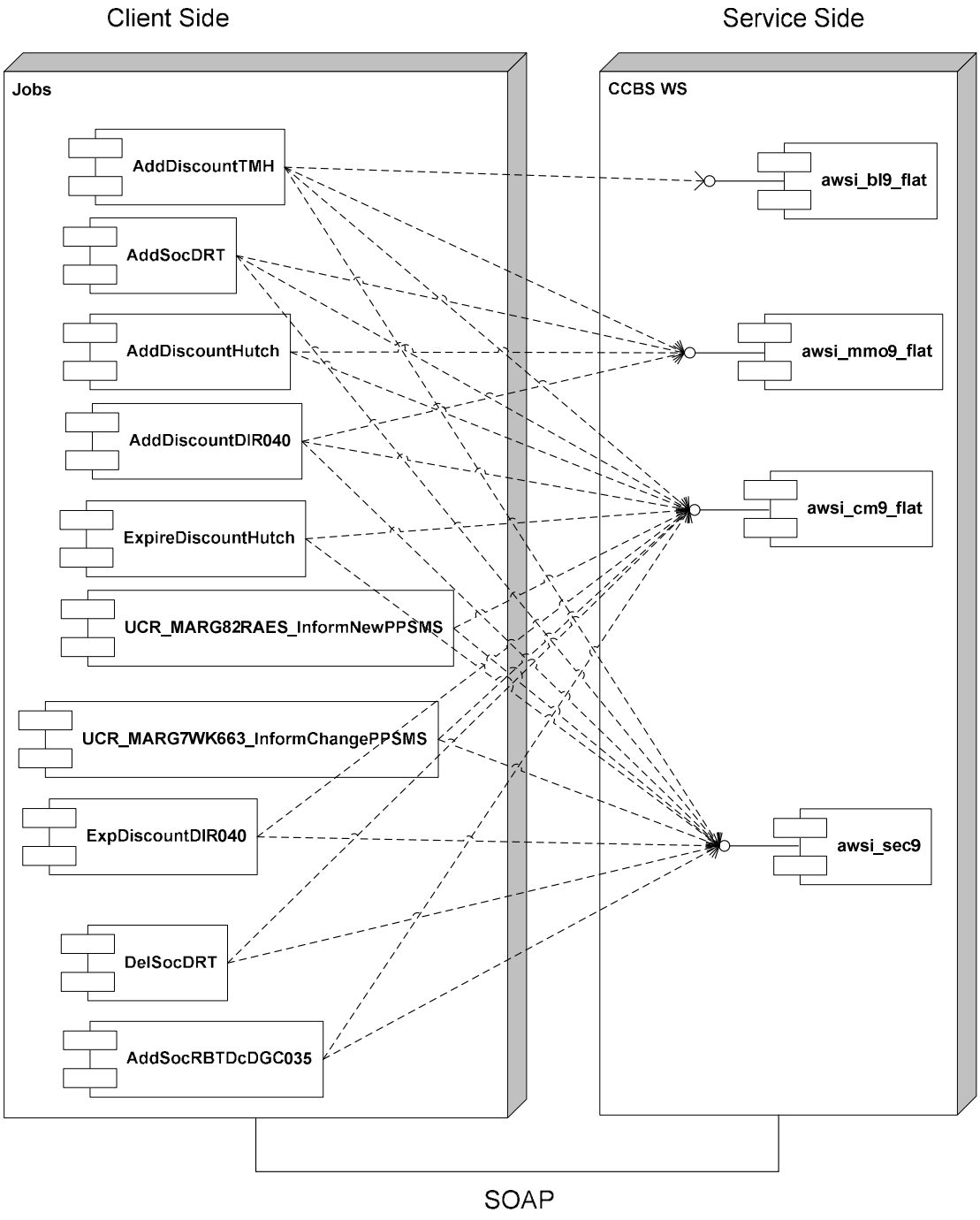
จากองค์ประกอบที่อธิบายมาข้างต้น ส่วนที่เกี่ยวข้องกับงานวิจัยคือในข้อ 4 กับข้อ 7 กล่าวคือ Job (ข้อที่ 4) จะเป็นส่วนเว็บเซอร์วิซในฝั่งไคลเอนต์ที่ไปเรียกใช้ CCBS WS ซึ่งเป็นส่วนเซอร์วิซ (ข้อที่ 7) โดยการติดต่อสื่อสารกันตรงส่วนนี้ระบบต้องการโค้ดความมั่นคง สำหรับรายละเอียดโค้ดบริเวณนี้สามารถตัดออกมาแสดงเป็นแผนภาพดีพลอยเมนต์ (Deployment Diagram) ได้ดังภาพที่ 4.2 และ ภาพที่ 4.3 ซึ่งจะเป็นระบบที่ถูกนำมาทดลองในงานวิจัยนี้ โดยสิ่งที่ 2 ภาพนี้แตกต่างกันคือ ภาพที่ 4.2 ใช้ในกรณีความต้องการความมั่นคงด้านการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์ซึ่งต้องการคอมโพเนนต์เพียง 3 ตัวคือ awsi_bl9_flat, awsi_cm9_flat และ awsi_mmo9 เท่านั้น ซึ่งคอมโพเนนต์ดังกล่าวมีเซอร์วิซแยกย่อยอยู่ภายในดังภาพที่ 4.4 และภาพที่ 4.5 รวมทั้งเซอร์วิซทั้งสิ้น 8 เซอร์วิซ แต่ในภาพที่ 4.3 ใช้ในกรณีความต้องการความมั่นคงด้านจัดการเซสชัน ซึ่งต้องการคอมโพเนนต์เพิ่มมา 1 ตัวคือ awsi_sec9

เมื่อพิจารณาภาพที่ 4.2 และ ภาพที่ 4.3 จะเห็นว่าคอมโพเนนต์ต่าง ๆ ที่ปรากฏในฝั่ง Jobs หรือไคลเอนต์ เมื่อติดต่อสื่อสารไปยัง CCBS WS ผ่านทางโซปจะมีความยุ่งเหยิง เมื่อทำการเขียนโค้ดความมั่นคงในระบบจึงมีโอกาสที่ผู้พัฒนาต่างคนต่างเขียน ทำให้โค้ดไม่มีแบบแผน รวมทั้งโอกาสโค้ดจะยุ่งเหยิง และกระจายไปทั่ว ดังนั้นผู้วิจัยจึงสนใจนำโค้ดส่วนนี้มาทดลองกับแบบรูปที่ได้ในบทที่ 3 เพื่อทำให้คอมโพเนนต์ต่าง ๆ ในฝั่งเซอร์วิซ สามารถรวมกันเป็นมอดูล

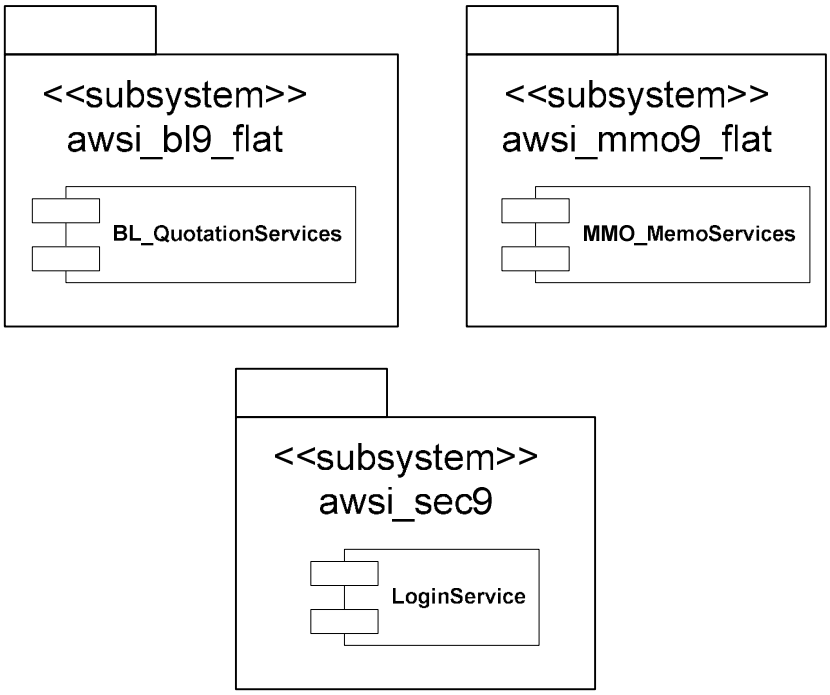
เดียวกัน พร้อมกับปรับปรุงโค้ดฝั่งไคลเอนต์ด้วยวิธีเอโอพี โดยแต่ละไคลเอนต์หรือ Job สามารถแสดงรายละเอียดการทำงานได้ด้วยแผนภาพลำดับ (Sequence Diagram) ซึ่งจะอยู่ในภาคผนวก ก



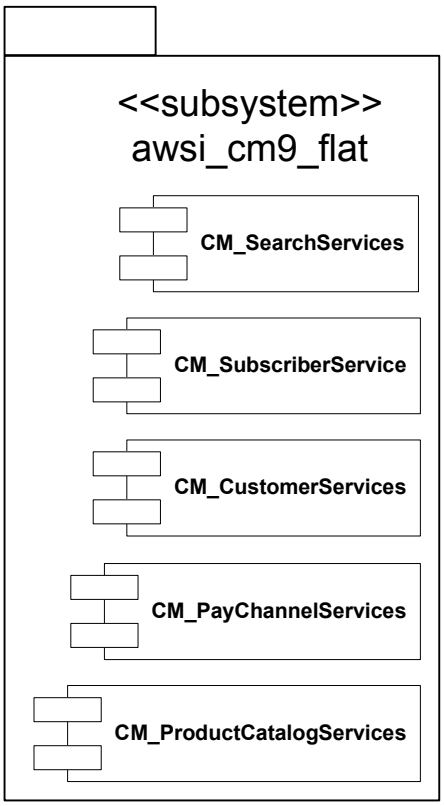
ภาพที่ 4.2 แผนภาพดีพลอยเมนต์ของระบบเว็บเซอร์วิซตัวอย่าง (ใช้ในกรณีความต้องการความมั่นคงด้านการพิสูจน์ตัวจริงและการพิสูจน์สิทธิ์)



ภาพที่ 4.3 แผนภาพดีพลอยเมนต์ของระบบเว็บเซอร์วิซตัวอย่าง (ใช้ในกรณีความต้องการความมั่นคงด้านการจัดการเซสชัน)



ภาพที่ 4.4 แผนภาพคอมโพเนนต์แสดงเซอร์วิซภายในของ awsi_bl9_flat, awsi_mmo9_flat และ awsi_sec9 ตามลำดับ



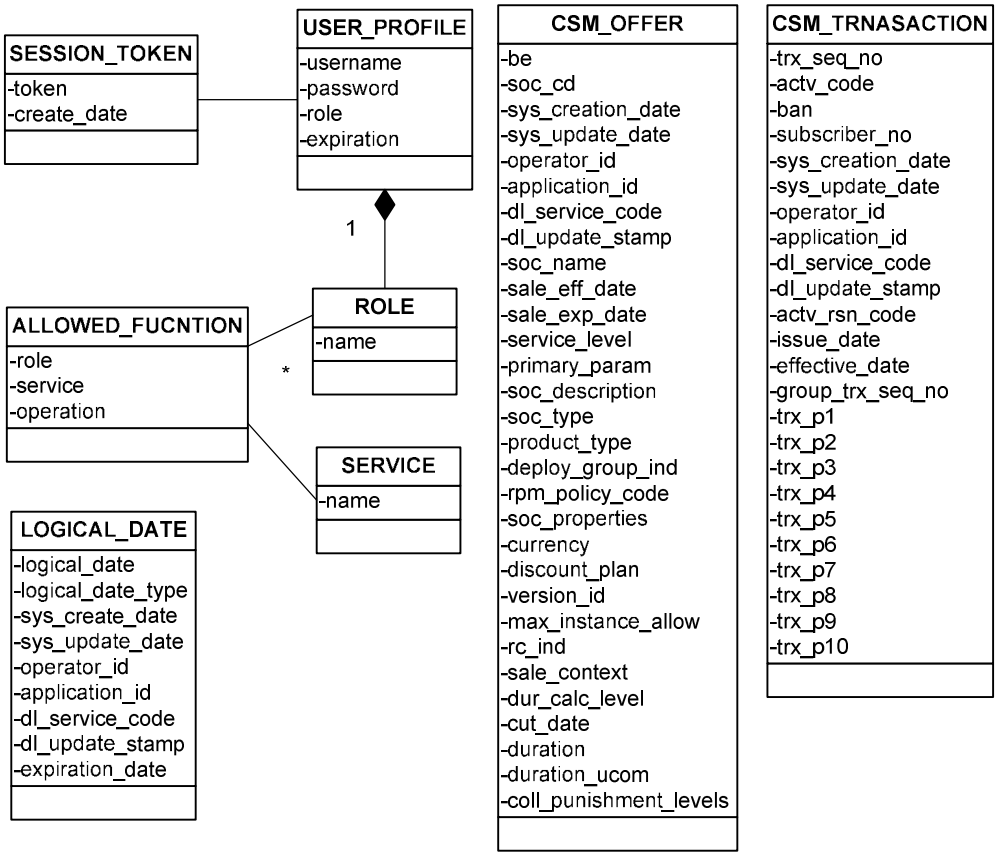
ภาพที่ 4.5 แผนภาพคอมโพเนนต์แสดงเซอร์วิซภายในของ awsi_cm9_flat

4.3 เขียนโค้ดเพิ่มเติม

เนื่องด้วยระบบตัวอย่าง เป็นระบบที่ใหญ่และซับซ้อน อีกทั้งโค้ดในฝั่งเซอร์วิซก็ยังติดปัญหาที่ไม่สามารถนำมาใช้ทดลองได้ นำมาได้แค่เพียงบางส่วน เนื่องด้วยนโยบายความมั่นคงที่ต้องรักษาความลับของบริษัท ผู้วิจัยจึงต้องนำโค้ดฝั่งเซอร์วิซมาพัฒนาต่อและออกแบบฐานข้อมูลเพิ่มเติมเพื่อให้ใช้ทดลองได้ ดังต่อไปนี้

4.3.1 การออกแบบฐานข้อมูล

ผู้วิจัยได้ทำการออกแบบฐานข้อมูลเพิ่มเติมดังแสดงแผนภาพคลาสในภาพที่ 4.6



ภาพที่ 4.6 แผนภาพคลาสของข้อมูลตัวอย่างฝั่งเซอร์วิซ

สำหรับแต่ละคลาสในภาพที่ 4.6 สามารถอธิบายได้โดยย่อดังนี้

รายละเอียดคลาสที่ผู้วิจัยได้ออกแบบเพิ่มมาใหม่ ได้แก่

1) ALLOWED_FUNCTION เก็บฟังก์ชันของแต่ละเซอร์วิซและสิทธิ์ที่จะให้ใช้งาน

2) LOGICAL_DATE เก็บข้อมูลวันที่ของระบบ

3) ROLE เก็บข้อมูลที่เป็นสิทธิ์การเข้าถึงและใช้งานทรัพยากรภายในระบบ

4) SERVICE เก็บข้อมูลที่เป็นรายชื่อเซอร์วิซต่าง ๆ ที่จะให้ไคลเอนต์มาใช้งาน

5) SESSION_TOKEN เก็บข้อมูลด้านเซสชัน

6) USER_PROFILE เก็บข้อมูลผู้ใช้งานได้แก่ ชื่อผู้ใช้งาน รหัสผ่าน และสิทธิ์ในการใช้งาน

รายละเอียดตารางที่ระบบมีอยู่แล้ว ผู้วิจัยไม่ได้ออกแบบเพิ่มขึ้นมา ได้แก่

1) CSM_OFFER เก็บข้อมูลรายละเอียดโปรโมชัน

2) CSM_TRANSACTIONS เก็บข้อมูลการทำทรานแซกชัน (Transaction) ของลูกค้า

4.4 นำแบบรูปไปใช้งานจริงกับตัวอย่างทดลอง

นำแบบรูปที่ได้ในบทที่ 3 นำมาประยุกต์ใช้กับตัวอย่างการทดลองในหัวข้อที่ 4.2 โดยในหัวข้อนี้จะแสดงตัวอย่างการเขียนโค้ดในภาษาจาวา ซึ่งมีตัวอย่างดังนี้

ในภาพที่ 4.7 เป็นการนำแบบรูป Authentication มาใช้งานฝั่งเซอร์วิซ โดยยกตัวอย่างการเขียนโค้ดของคลาส AuthenticationCheckPoint ซึ่งจะอิมพลีเมนต์มาจาก CheckPoint อีกที ทำให้โค้ดความมั่นคงในแง่การพิสูจน์ตัวจริงมีจุดศูนย์กลางอยู่ที่คลาส AuthenticationCheckPoint ที่เดียว ส่วนในภาพที่ 4.8 เป็นตัวอย่างการนำแบบรูป Authentication มาเขียนด้วย AspectJ ได้เป็นคลาสชื่อ AuthenticationAspect และได้ทำการวีฟโค้ดการพิสูจน์ตัวจริงในฝั่งไคลเอนต์

```java
package amdocs.main.wsse;

import org.apache.log4j.Logger;

import amdocs.main.model.Subject;

public class AuthenticationCheckPoint  implements CheckPoint {
    final static Logger logger = Logger.getLogger(AuthenticationCheckPoint.class);
    private AuthenticationEnforcer authenticationEnforcer = new AuthenticationEnforcer();

//  @Override
    public boolean identify(RequestContext requestContext) throws Exception {
        return authenticationEnforcer.authenticate(requestContext);
    }

    public Subject getSubject() {
        return authenticationEnforcer.getSubject();
    }

}

```

ภาพที่ 4.7 ตัวอย่างโค้ดของคลาส AuthenticationCheckPoint

```java
package it.ccbsoa.svcbundling.aop;

import it.ccbomesb.svcbundling.conf.Config;

public aspect AuthenticationAspect {

    pointcut serviceStubMethod() : execution(public * amdocs..*ServicesSoapBindingStub.*(..));

    Object around() : serviceStubMethod() {
        Object instance = thisJoinPoint.getThis();
        if (!(instance instanceof Stub)) {
            throw new IllegalArgumentException("Invalid instance. it isn't "
                    + Stub.class.getName());
        }
        System.out.println(thisJoinPoint.getThis().getClass().getName() + "."
                + thisJoinPoint.getSignature().getName());
        Stub clientStub = (Stub) instance;
        clientStub.setTimeout(EnvControl.WEBSERVICE_TIMEOUT);
        try {
            clientStub = setHeader(clientStub);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return proceed();
    }

    public Stub setHeader(Stub clientStub) throws Exception{
        clientStub.clearHeaders();
        clientStub.setHeader(this.getSOAPHeaderSecurity());
        return clientStub;
    }

```

ภาพที่ 4.8 ตัวอย่างโค้ด AuthenticationAspect ของแบบรูป Authentication

ภาพที่ 4.9 เป็นการนำแบบรูป Authorization มาใช้งานฝั่งเซอร์วิซ โดยยกตัวอย่างการเขียนโค้ดของคลาส AuthorizationCheckPoint ซึ่งจะอิมพลีเมนต์มาจาก CheckPoint อีกทีทำให้โค้ดความมั่นคงในแง่การพิสูจน์สิทธิ์มีจุดศูนย์กลางอยู่ที่คลาส AuthorizationCheckPoint ทีเดียว ส่วนในภาพที่ 4.10 เป็นตัวอย่างการนำคลาสชื่อ AuthenticationAspect ซึ่งเขียนขึ้นด้วย

AspectJ มาทำการวีฟโค้ดการพิสูจน์ตัวจริงในฝั่งไคลเอนต์ เนื่องจากไคลเอนต์ต้องทำการพิสูจน์ตัวจริง เมื่อผ่านการพิสูจน์ตัวจริงสำเร็จ ก็จะไปพิสูจน์สิทธิ์ในฝั่งเซอร์วิซลำดับต่อมา

```java
package amdocs.main.wsse;


import org.apache.log4j.Logger;

import amdocs.main.model.Subject;

public class AuthorizationCheckPoint implements CheckPoint {
    final static Logger logger = Logger.getLogger(AuthorizationCheckPoint.class);
    private AuthorizationEnforcer authorizationEnforcer = new AuthorizationEnforcer();

//  @Override
    public boolean identify(RequestContext requestContext) throws Exception {
        authorizationEnforcer.authorize(requestContext);
        return authorizationEnforcer.isAuthorized(requestContext);
    }

//  @Override
    public Subject getSubject() {
        return authorizationEnforcer.getSubject();
    }

}
```

ภาพที่ 4.9 ตัวอย่างโค้ดของคลาส AuthorizationCheckPoint

```java
package it.ccbsoa.svcbundling.aop;

import it.ccbomesb.svcbundling.conf.Config;

public aspect AuthenticationAspect {

    pointcut serviceStubMethod() : execution(public * amdocs..*ServicesSoapBindingStub.*(..));

    Object around() : serviceStubMethod() {
        Object instance = thisJoinPoint.getThis();
        if (!(instance instanceof Stub)) {
            throw new IllegalArgumentException("Invalid instance. it isn't "
                    + Stub.class.getName());
        }
        System.out.println(thisJoinPoint.getThis().getClass().getName() + "."
                + thisJoinPoint.getSignature().getName());
        Stub clientStub = (Stub) instance;
        clientStub.setTimeout(EnvControl.WEBSERVICE_TIMEOUT);
        try {
            clientStub = setHeader(clientStub);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return proceed();
    }

    public Stub setHeader(Stub clientStub) throws Exception{
        clientStub.clearHeaders();
        clientStub.setHeader(this.getSOAPHeaderSecurity());
        return clientStub;
    }
```

ภาพที่ 4.10 ตัวอย่างโค้ด AuthenticationAspect ของแบบรูป Authorization

ในภาพที่ 4.11 เป็นการนำแบบรูป Security Session มาใช้งานฝั่งเซอร์วิซ โดยยกตัวอย่างการเขียนโค้ดของคลาส SessionCheckPoint ซึ่งจะอิมพลีเมนต์มาจาก CheckPoint อีกที ทำให้โค้ดความมั่นคงในแง่เซสชันด้านความมั่นคงมีจุดศูนย์กลางอยู่ที่คลาส SessionCheckPoint ที่เดียว ส่วนในภาพที่ 4.12 เป็นตัวอย่างการนำแบบรูป Security Session มาเขียนด้วย AspectJ ได้เป็นคลาสชื่อ SessionAspect และได้ทำการวีฟโค้ดเซสชันด้านความมั่นคงในฝั่งไคลเอนต์

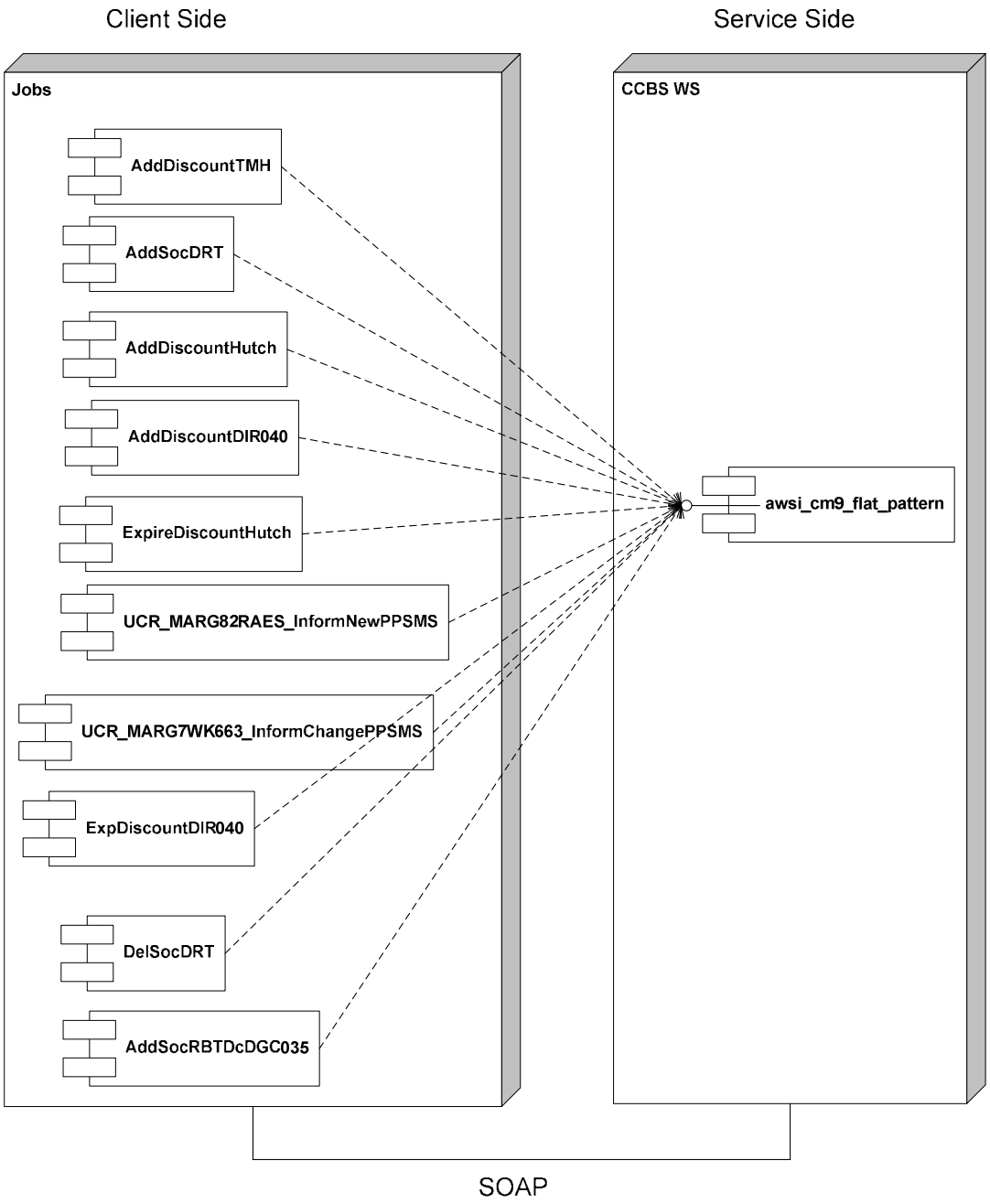
```
1  package amdocs.main.wsse;
2
3  import java.util.List;
4
5  import org.apache.axis.AxisFault;
6
7  import amdocs.main.dao.AllowedFunctionDAO;
8  import amdocs.main.model.AllowedFunction;
9  import amdocs.main.model.Session;
10 import amdocs.main.model.UserProfile;
11 import amdocs.main.util.AppUtils;
12
13 public class SessionCheckPoint implements CheckPoint {
14
15 //  @Override
16     public boolean identify(String sessionId, String inputPassword)
17             throws Exception {
18         Session session = SessionManager.get(sessionId);
19         if(session!=null){
20             return true;
21         }
22
23         UserProfile userProfile = Authentication.authenticate(sessionId, inputPassword);
24         if(userProfile==null){
25             throw new AxisFault(sessionId + " user login failed, user name or password is wrong");
26         }
27
28         List<AllowedFunction> allowedFunctionList = AllowedFunctionDAO.searchAllowedOperation(sessionId);
29         String sessionToken = AppUtils.generateSessionToken();
30
31         SessionManager.set(sessionToken,userProfile,allowedFunctionList);
32         return false;
33     }
34 }
```

ภาพที่ 4.11 ตัวอย่างโค้ดของคลาส SessionCheckPoint

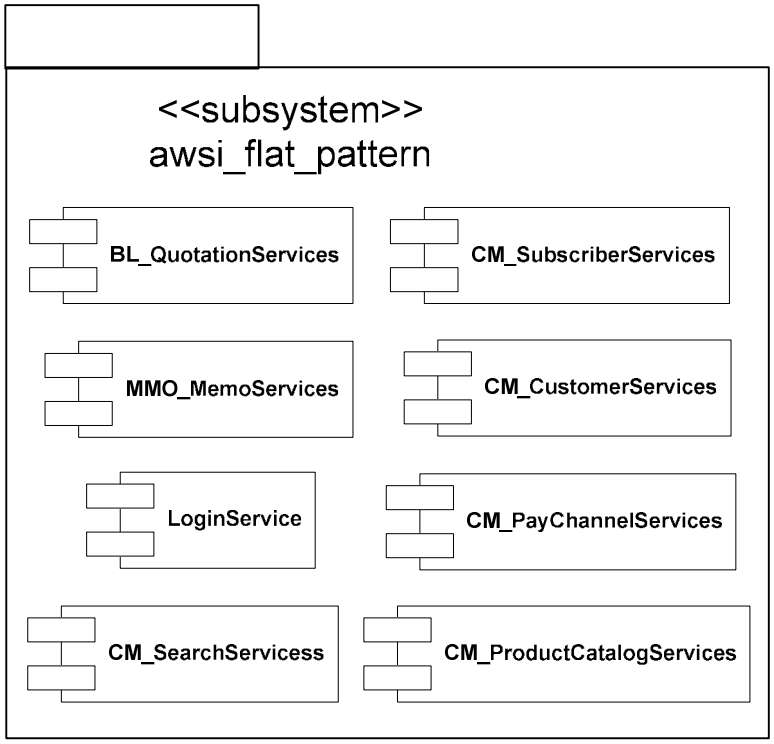
```
21  public aspect SessionAspect {
22      private String sessionToken = "";
23
24      pointcut serviceStubMethod() : execution(public * amdocs..*ServicesSoapBindingStub.*(..));
25
26      Object around() throws RemoteException, CMException: serviceStubMethod() {
27          Object instance = thisJoinPoint.getThis();
28          if (!(instance instanceof Stub)) {
29              throw new IllegalArgumentException("Invalid instance. it isn't "
30                      + Stub.class.getName());
31          }
32
33          Object result = null;
34          Stub clientStub = (Stub) instance;
35          try {
36              clientStub.setTimeout(EnvControl.WEBSERVICE_TIMEOUT);
37              clientStub = setHeader(clientStub);
38              result = proceed();
39          } catch (Exception e) {
40              if (isSessionExpired(e)) {// if session is expire
41                  try {
42                      this.authen();
43                      clientStub.setTimeout(EnvControl.WEBSERVICE_TIMEOUT);
44                      clientStub = setHeader(clientStub);
45                  } catch (Exception e1) {
46                      e1.printStackTrace();
47                  }
48                  result = proceed(); // retry WS call
49              } else {
50                  e.printStackTrace();
51              }
52          }
53          return result;
54      }
```

ภาพที่ 4.12 ตัวอย่างโค้ด SessionAspect ของแบบรูป Security Session

สำหรับคอมโพเนนต์ต่าง ๆ ของระบบตัวอย่างหลังจากใช้แบบรูปในฝั่งเซอร์วิซและวิธีเอโอพีในฝั่งไคเลอนต์จะแสดงให้เห็นดังภาพที่ 4.13 ซึ่งคอมโพเนนต์ต่าง ๆ ในฝั่งเซอร์วิซจะถูกรวมเป็นคอมโพเนนต์เดียวชื่อ awsi_flat_pattern โดยที่จำนวนเซอร์วิซในฝั่งเซอร์วิซยังคงมีอยู่เท่าเดิมดังแสดงในภาพที่ 4.14 ซึ่งได้แสดงคอมโพเนนต์ต่าง ๆ ภายในของ awsi_flat_pattern เอาไว้ สำหรับส่วนของคอมโพเนนต์ฝั่งไคลเอนต์ยังคงมีจำนวนเท่าเดิม

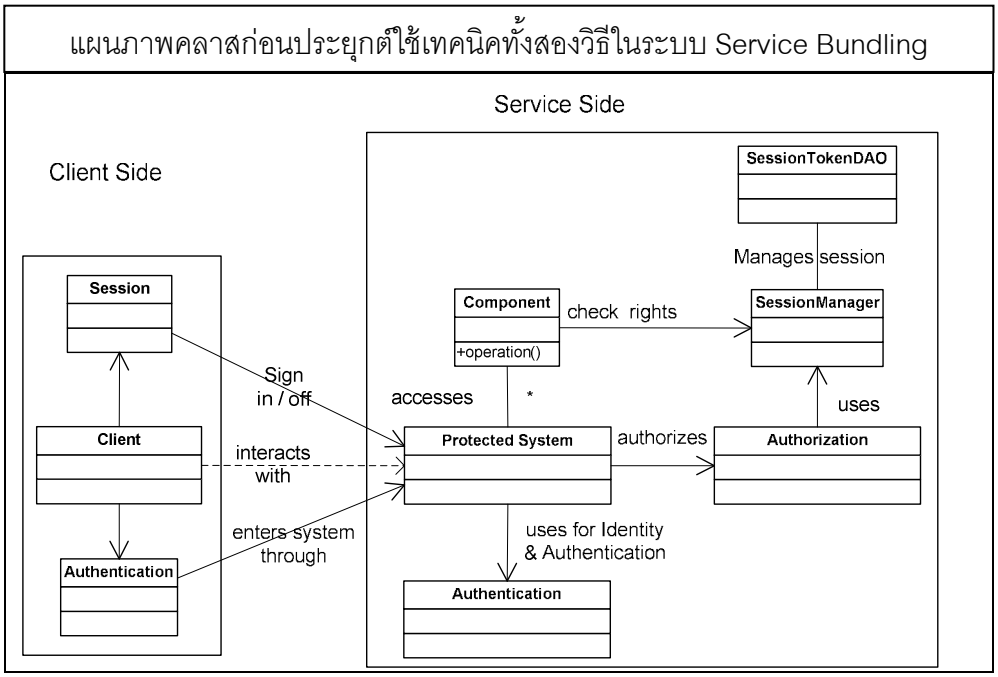


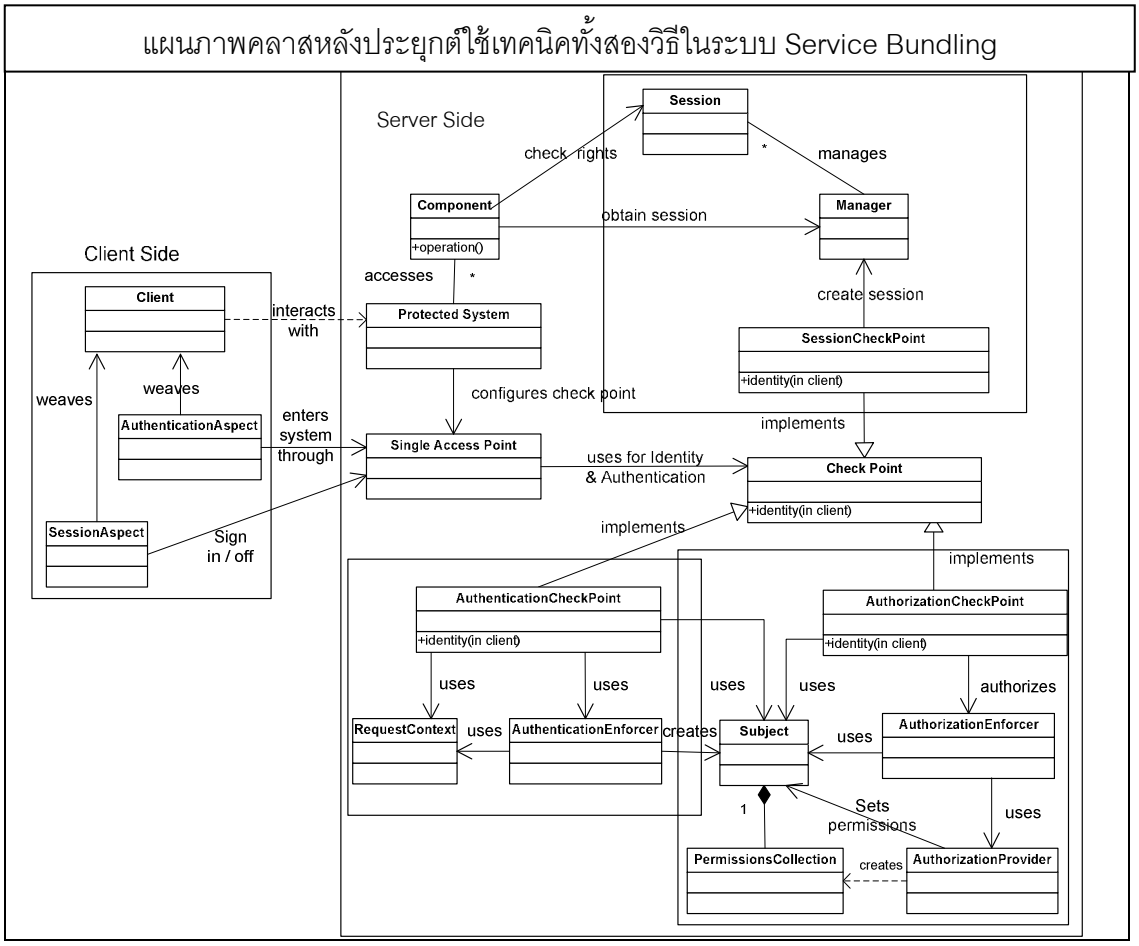
ภาพที่ 4.13 แผนภาพคอมโพเนนต์แสดงระบบตัวอย่างหลังใช้แบบรูปและเอโอพี



ภาพที่ 4.14 แผนภาพคอมโพเนนต์แสดงเซอร์วิซภายในของ awsi_flat_pattern

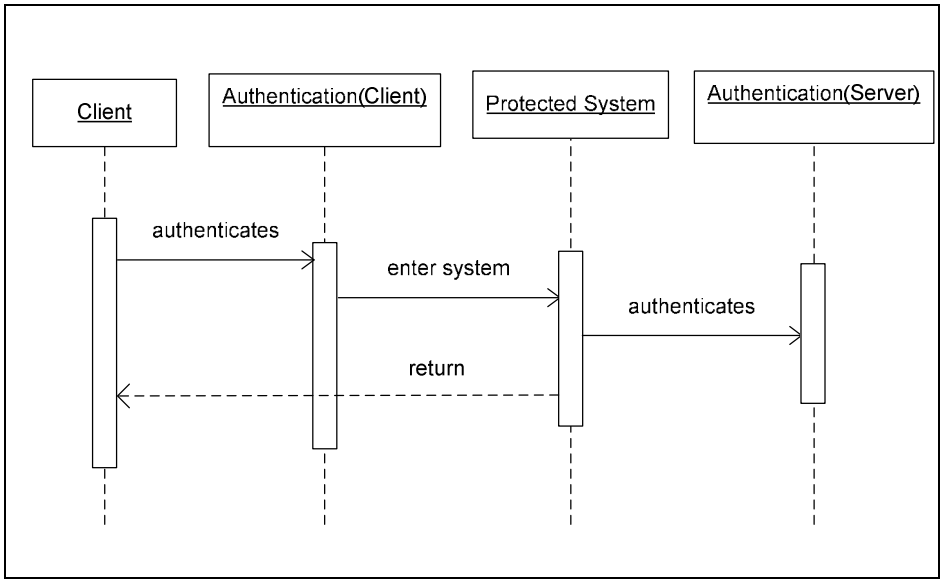
สำหรับภาพที่ 4.15 เป็นแผนภาพคลาสโดยรวมของระบบตัวอย่าง Service Bundling ซึ่งจะแสดงภาพก่อนที่จะใช้แบบรูปในฝั่งเซอร์วิซและนำวิธีเอโอพีมาใช้ในฝั่งไคลเอนต์ ขณะเดียวกันก็จะแสดงเปรียบเทียบหลังจากนำเทคนิคทั้งสองวิธีมาใช้ในฝั่งเซอร์วิซและไคลเอนต์ ซึ่งจะทำให้สภาพมอดูลาร์ในระบบตัวอย่างโดยรวมดีขึ้น



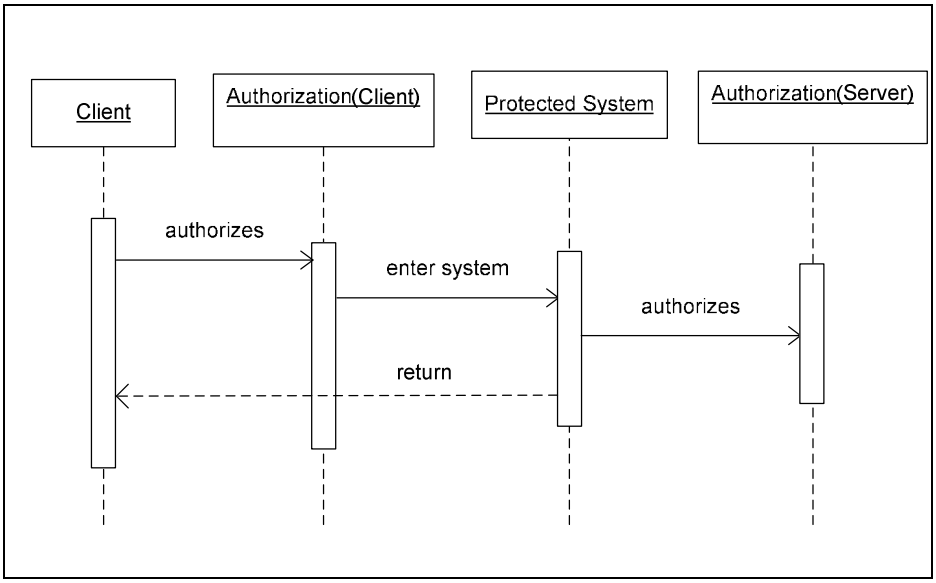


ภาพที่ 4.15 ภาพของระบบ Service Bundling ก่อนและหลังนำแบบรูปและเอโอพีมาประยุกต์ใช้

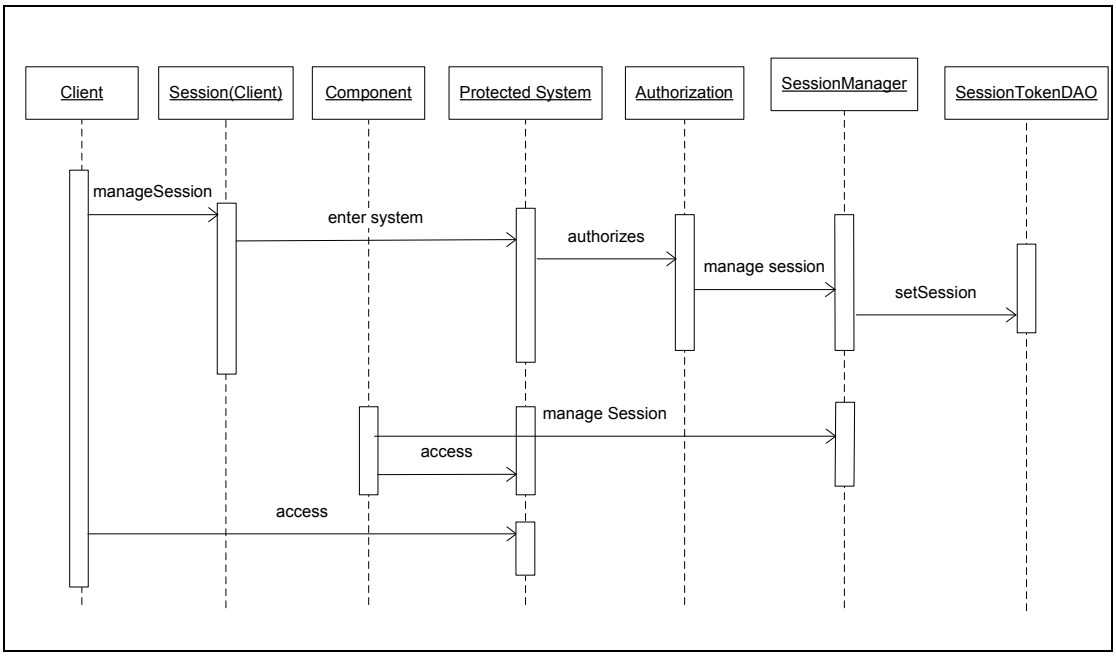
ภาพที่ 4.16 ภาพที่ 4.17 และภาพที่ 4.18 เป็นแผนภาพลำดับก่อนนำแบบรูปมาใช้ฝั่งเซอร์วิซ และก่อนนำวิธีเอโอพีมาใช้ฝั่งไคลเอนต์ สำหรับความมั่นคงด้านการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และการจัดการเซสชันด้านความมั่นคง ตามลำดับ ซึ่งหลังจากนำเทคนิคทั้งสองวิธีมาใช้ ก็ยังคงมีความมั่นคงทั้งสามอย่างอยู่ครบทุกประการ ตามที่ได้แสดงลำดับการทำงานของการพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และการจัดการเซสชันด้านความมั่นคง ดังแสดงในภาพที่ 3.7 ภาพที่ 3.9 และภาพที่ 3.11 (ในหัวข้อที่ 3.6) อีกทั้งการใช้แบบรูปที่นำเสนอ ยังเพิ่มความมั่นคงในแง่การป้องกันการถูกบุกรุกจากภายนอก เนื่องจากระบบก่อนที่จะใช้เทคนิคทั้งสองวิธีนั้น การเข้าถึงระบบจากภาพที่ 4.16 ภาพที่ 4.17 และภาพที่ 4.18 จะเข้าถึงส่วน Protected System ได้โดยตรง ทำให้เสี่ยงต่อการถูกบุกรุก แต่หลังจากนำแบบรูปมาใช้งาน การเข้าถึงจะกระทำได้เพียงช่องทางเดียวคือ Single Access Point เท่านั้น ทำให้ป้องกันการถูกบุกรุกได้มีประสิทธิภาพมากยิ่งขึ้น



ภาพที่ 4.16 แผนภาพลำดับของระบบตัวอย่างสำหรับความมั่นคงด้านพิสูจน์ตัวจริงก่อนใช้แบบรูป



ภาพที่ 4.17 แผนภาพลำดับของระบบตัวอย่างสำหรับความมั่นคงด้านการพิสูจน์สิทธิ์ก่อนใช้แบบรูป



ภาพที่ 4.18 แผนภาพลำดับของระบบตัวอย่างสำหรับการจัดการเซสชันด้านความมั่นคงก่อนใช้แบบรูป

4.5 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา

สภาพแวดล้อมในการพัฒนาเครื่องมือ ทั้งฮาร์ดแวร์ (Hardware) และซอฟต์แวร์ (Software) มีรายละเอียดดังนี้

1) เครื่องคอมพิวเตอร์พกพา (Notebook)
2) หน่วยประมวลผล Intel(R) Core(TM) i5-2450 ความเร็ว 2.5 กิกะเฮิรตซ์ (GHz)
3) หน่วยความจำหลัก DDR3 ขนาด 4096 เมกะไบต์ (MB)
4) จานบันทึกแบบแข็ง (Hard Disk) ความจุ 500 กิกะไบต์ (GB)
5) ระบบปฏิบัติการวินโดวส์เซเวนอัลติเมท (Microsoft Windows 7 Ultimate)
6) เครื่องมือพัฒนาโปรแกรมได้แก่ อีคลิปส์ เวอร์ชัน จูโน (Eclipse Version Juno)
7) เครื่องมือสำหรับออกแบบและจัดทำเอกสารของการพัฒนาระบบ ได้แก่ ไมโครซอฟท์ออฟฟิซ 2007 (Microsoft Office 2007)
8) เครื่องมือพัฒนาเอโอพีได้แก่ ชุดเสริม (Plugin) สำหรับการพัฒนาเอโอพีบนอีคลิปส์ (AspectJ Development Tools or AJDT) http://www.eclipse.org/ajdt/
9) เครื่องมือสำหรับวัดสภาพมอดูลาร์ ได้แก่ คอนเซิร์นแทกเกอร์ (ConcernTagger) ซึ่งเป็นชุดเสริม (Plugin) บนอีคลิปส์ ใช้สำหรับวัดค่า CDC และ CDO และเอโอพีเมตริกส์ เวอร์ชัน 3.0 (aopmetrics version 3.0) ใช้สำหรับวัดค่า LOCC, CBM และ LCO
10) ฐานข้อมูลเป็นออราเคิล 11 g เอกซ์เพรสเอดิชัน (Oracle Database 11g Express Edition)
11) เซิร์ฟเวอร์เพื่อรองรับบริการเว็บเซอร์วิซ ได้แก่ Tomcat

บทที่ 5
ผลการทดลองและวิเคราะห์ผล

ในบทนี้จะกล่าวถึงผลการทดลองและการวิเคราะห์ผล จากการนำแบบรูปที่เสนอในบทที่ 3 ไปใช้กับตัวอย่างการทดลองในบทที่ 4 ซึ่งมีรายละเอียดดังนี้

งานวิจัยได้ทำการวัดสภาพมอดูลาร์ของระบบ Service Bundling Job ทั้ง 2 เวอร์ชัน คือ เวอร์ชันที่มีการพัฒนาความต้องการด้านความมั่นคงโดยยังไม่ใช้แบบรูปและเอโอพี กับเวอร์ชันที่มีการประยุกต์แบบรูปด้านความมั่นคงและเอโอพีแล้ว ในการวัดสภาพมอดูลาร์จะแบ่งตัววัดเป็น 2 ชุด ชุดแรกได้แก่ LOCC, CBM และ LCO ใช้วัดโค้ดฝั่งเซอร์วิซ ส่วนตัววัดชุดที่สองได้แก่ LOCC, CBM, LCO, CDC และ CDO ใช้วัดโค้ดฝั่งไคลเอนต์ ซึ่งตัววัดทั้ง 2 ชุดให้ผลดังต่อไปนี้

5.1 ผลการทดลองวัดสภาพมอดูลาร์ในฝั่งเซอร์วิซ

งานวิจัยนี้ได้ทำการวัดค่า LOCC, CBM และ LCO ก่อนใช้แบบรูปและหลังใช้แบบรูปในฝั่งเซอร์วิซ ซึ่งให้ผลดังนี้

ตารางที่ 5.1 ผลการทดลองวัดสภาพมอดูลาร์ในฝั่งเซอร์วิซ ก่อนและหลังใช้แบบรูปความมั่นคง

	จำนวนคลาส	LOCC		CBM		LCO	
		รวมทั้งระบบ	เฉลี่ยต่อคลาส	รวมทั้งระบบ	เฉลี่ยต่อคลาส	รวมทั้งระบบ	เฉลี่ยต่อคลาส
ผลการวัดแบบรูป Authentication							
ไม่ใช้แบบรูป	34	1722	50.65	101	2.97	42	1.24
ใช้แบบรูป	22	1013	46.05	65	2.95	68	3.09
ผลการวัดแบบรูป Authorization (ซึ่งมีผลการวัด Authentication มาผสมรวมด้วย)							
ไม่ใช้แบบรูป	43	2116	49.21	132	3.07	48	1.12
ใช้แบบรูป	28	1158	41.36	82	2.93	70	2.50

ตารางที่ 5.1ผลการทดลองวัดสภาพมอดูลาร์ในฝั่งเซอร์วิซก่อนและหลังใช้แบบรูปความมั่นคง(ต่อ)

	จำนวนคลาส	LOCC		CBM		LCO	
		รวมทั้งระบบ	เฉลี่ยต่อคลาส	รวมทั้งระบบ	เฉลี่ยต่อคลาส	รวมทั้งระบบ	เฉลี่ยต่อคลาส
ผลการวัดแบบรูป Authorization (ซึ่งไม่มีผลการวัด Authentication มาผสมรวมด้วย)							
ไม่ใช้แบบรูป	37	1813	49.00	108	2.92	48	1.30
ใช้แบบรูป	24	1014	42.25	69	2.88	16	0.67
ผลการวัดแบบรูป Security Session							
ไม่ใช้แบบรูป	63	3193	50.68	189	3.00	436	6.92
ใช้แบบรูป	27	1324	49.04	93	3.44	69	2.56

จากตารางที่ 5.1 จะพบว่าค่า CBM ของแบบรูป Security Session หลังใช้แบบรูปเพิ่มขึ้นเมื่อพิจารณาเฉลี่ยต่อคลาส โดยสาเหตุที่ทำให้สูงเนื่องด้วยแบบรูปที่นำเสนอได้เพิ่มคลาส Manager กับ Session เข้ามา ซึ่งทั้ง 2 คลาสดังกล่าวมีโอกาสถูกเรียกใช้จากคอมโพเนนต์อื่น ๆ ของระบบด้วย จึงมีโอกาสทำให้การต่อประกบเพิ่มสูงขึ้น โดยจะต่างกับแบบรูป Authentication ซึ่งคลาสต่าง ๆ ที่นำเสนอ เช่น RequestContext และ AuthenticationEnforcer จะไม่มีโอกาสต่อประกบกับคอมโพเนนต์อื่น ๆ ในระบบ และเช่นเดียวกัน แบบรูป Authorization ซึ่งคลาสต่าง ๆ ที่นำเสนอ เช่น AuthorizationEnforcer และ AuthorizationProvider ไม่มีโอกาสต่อประกบกับคอมโพเนนต์อื่น ๆ ในระบบ ผลการวัดค่า CMB จึงดีกว่ากรณีของแบบรูป Security Session

จากตารางที่ 5.1 เช่นเดียวกันจะพบว่าแบบรูป Authentication กับแบบรูป Authorization (ซึ่งมีแบบรูป Authentication รวมอยู่ด้วย) มีค่า LCO ที่สูงขึ้นหลังจากใช้แบบรูป ผู้วิจัยจึงทำการวิเคราะห์ 2 แบบรูปดังกล่าวถึงสาเหตุที่ทำให้ค่า LCO สูงขึ้น โดยมีรายละเอียดดังนี้

1) การวิเคราะห์ค่า LCO ของแบบรูป Authentication

ระบบตัวอย่างซึ่งมีโค้ดความมั่นคงของการพิสูจน์ตัวจริง แต่ยังไม่มีการใช้แบบรูป Authentication ในฝั่งเซอร์วิซ ระบบจะถูกแยกย่อยเป็นคอมโพเนนต์ต่าง ๆ ได้แก่ awsi_bl9_flat, awsi_cm9_flat และ awsi_mmo9 (อ้างอิงได้จากภาพที่ 4.2) โดยในตารางที่ 5.2 จะแสดงผลการวัดค่า LOCC, CBM และ LCO ของแต่ละคอมโพเนนต์ออกมา พร้อมทั้งแสดงผลการวัดเมื่อนำแบบรูป Authentication มาใช้งาน ทำให้คอมโพเนนต์ต่าง ๆ ที่ถูกแยกย่อยถูกรวมกันเป็นมอดูลเดียว ส่วนในตารางที่ 5.3 จะสรุปผลรวมของการวัดทั้งก่อนและหลังใช้แบบรูป

ตารางที่ 5.2 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
1	StringUtils	39	0	0	39	0	0	39	0	0	39	0	0
2	AppUtils	20	4	0	20	4	0	20	4	0	20	4	0
3	DBUtils	65	1	0	65	1	0	65	1	0	65	1	0
4	UserProfileDAO	59	2	0	59	2	0	59	2	0	59	2	0
5	SOAPHandler	18	5	0	18	5	0	18	5	0	18	5	0
6	SecurityHandler	32	2	0	32	2	0	32	2	0			
7	UserProfile	101	0	14	101	0	14	101	0	14	101	0	14
8	SecurityEncoding	10	1	0	10	1	0	10	1	0	10	1	0
9	Authentication	42	6	0	42	6	0	42	6	0			

ตารางที่ 5.2 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ (ต่อ)

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
10	BL_QuotationServicesSoapBindingImpl	15	4	0							14	3	0
11	CM_ProductCatalogServicesSoapBindingImpl				69	3	0				56	2	0
12	CM_SubscriberServicesSoapBindingImpl				214	11	0				179	10	0
13	CM_CustomerServicesSoapBindingImpl				70	5	0				57	4	0
14	CM_SearchServicesSoapBindingImpl				119	7	0				108	6	0

ตารางที่ 5.2 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ (ต่อ)

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
15	CM_PayChannelServicesSoapBindingImpl				51	6	0				43	5	0
16	MMO_MemoServicesSoapBindingImpl							26	2	0	21	1	0
17	SingleAccessPoint										47	7	0
18	Subject										37	1	0
19	AuthenticationCheckPoint										10	2	0
20	RequestContext										52	3	54
21	CheckPointFactory										40	1	0

ตารางที่ 5.2 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ (ต่อ)

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
22	AuthenticationEnforcer										23	6	0
23	CheckPoint										4	0	0
24	NullCheckPoint										10	1	0
	ผลรวม	401	25	14	909	53	14	412	23	14	1013	65	68

ตารางที่ 5.3 สรุปผลรวมการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ

ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ			
ชื่อคอมโพเนนต์	LOCC	CBM	LCO
awsi_bl9_flat	401	25	14
awsi_cm9_flat	909	53	14
awsi_mmo9	412	23	14
ผลรวม	**1722**	**101**	**42**
ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ			
คอมโพเนนต์รวมหลังใช้แบบรูป	LOCC	CBM	LCO
ผลรวม	**1013**	**65**	**68**

จากตารางที่ 5.3 พบว่าแบบรูปได้นำคลาส RequestContext [4] มาใช้งาน ซึ่งคลาสนี้มีค่า LCO ที่สูงมาก อีกทั้งเมื่อพิจารณาระบบก่อนนำแบบรูปมาใช้ ค่าความเชื่อมแน่นของระบบค่อนข้างดีอยู่แล้ว เมื่อนำแบบรูปมาใช้จึงส่งผลให้การวัดโดยรวมมีค่า LCO ที่เพิ่มขึ้นหรือทำให้แย่ลงกว่าเดิม และเมื่อพิจารณาโค้ดของ Request Context ดังภาพที่ 5.1 มันเป็นเพียงคลาสที่เก็บข้อมูล เช่น ชื่อผู้ใช้งาน รหัสผ่าน และข้อมูลผู้ใช้ เป็นต้น ซึ่งมีแต่เมท็อด Getter และ Setter ที่ใช้สำหรับดึงค่าตัวแปรและแก้ไขค่าตัวแปร ซึ่งจะไม่ยุ่งเกี่ยวกับตัวแปรอื่น ๆ ภายในคลาส จะยุ่งเฉพาะกับตัวแปรของมันอย่างเดียว จึงเป็นเหตุให้ค่า LCO เฉพาะคลาสนี้สูงขึ้น

```
public class RequestContext {
	private String username;
	private String password;
	private MessageContext messageContext;
	private String operationName ;
	private String serviceName ;
	private Subject subject;

	public RequestContext(WSPasswordCallback pc){
		this.username = pc.getIdentifier();
		this.password = pc.getPassword();
		this.messageContext =MessageContext.getCurrentContext();
		this.operationName = messageContext.getOperation().getName();
		this.serviceName = messageContext.getTargetService();
	}


	public RequestContext(String username,String password){
		this.username = username;
		this.password = password;
		this.messageContext =MessageContext.getCurrentContext();
		this.operationName = messageContext.getOperation().getName();
		this.serviceName = messageContext.getTargetService();
	}


public String getUsername() { return username; }

public void setUsername(String username) { this.username = username; }

	/// To do Getter / Setter Method
	// ……
	// ……
```

ภาพที่ 5.1 โค้ดบางส่วนของ RequestContext

2) การวิเคราะห์ค่า LCO ของแบบรูป Authorization

ในตารางที่ 5.4 จะแสดงผลการวัดค่าคอมโพเนนต์ในระบบได้แก่ awsi_bl9_flat, awsi_cm9_flat และ awsi_mmo9 ซึ่งยังไม่ใช้แบบรูป Authorization พร้อมทั้งแสดงผลการวัดหลังจากที่นำแบบรูปมาใช้งาน ทำให้คอมโพเนนต์ต่าง ๆ ที่แยกย่อยรวมกันเป็นมอดูลเดียวกัน ส่วนในตารางที่ 5.5 จะแสดงผลรวมของการวัดทั้งก่อนและหลังใช้แบบรูป

ตารางที่ 5.4 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
1	StringUtils	39	0	0	39	0	0	39	0	0	39	0	0
2	AppUtils	20	4	0	20	4	0	20	4	0	20	4	0
3	DBUtils	65	1	0	65	1	0	65	1	0	65	1	0
4	UserProfileDAO	59	2	0	59	2	0	59	2	0	59	2	0
5	AllowedFunctionDAO	41	2	0	41	2	0	41	2	0	41	2	0
6	SOAPHandler	18	5	0	18	5	0	18	5	0	18	5	0
7	SecurityHandler	32	2	0	32	2	0	32	2	0			
8	AllowedFunction	24	0	2	24	0	2	24	0	2	24	0	2
9	UserProfile	101	0	14	101	0	14	101	0	14	101	0	14

ตารางที่ 5.4 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ (ต่อ)

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
10	Authorization	38	6	0	38	6	0	38	6	0			
11	SecurityEncoding	10	1	0	10	1	0	10	1	0	10	1	0
12	Authentication	42	6	0	42	6	0	42	6	0			
13	BL_QuotationServicesSoapBindingImpl	16	5	0							14	3	0
14	CM_ProductCatalogServicesSoapBindingImpl				82	4	0				56	2	0
15	CM_SubscriberServicesSoapBindingImpl				249	12	0				179	10	0

ตารางที่ 5.4 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ (ต่อ)

		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
16	CM_CustomerServicesSoapBindingImpl				83	6	0				57	4	0
17	CM_SearchServicesSoapBindingImpl				129	8	0				107	6	0
18	CM_PayChannelServicesSoapBindingImpl				59	7	0				43	5	0
19	MMO_MemoServicesSoapBindingImpl							31	3	0	21	1	0
20	SingleAccessPoint										52	7	0
21	Subject										37	1	0
22	PermissionsCollection										30	5	0

ตารางที่ 5.4 ผลการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ (ต่อ)

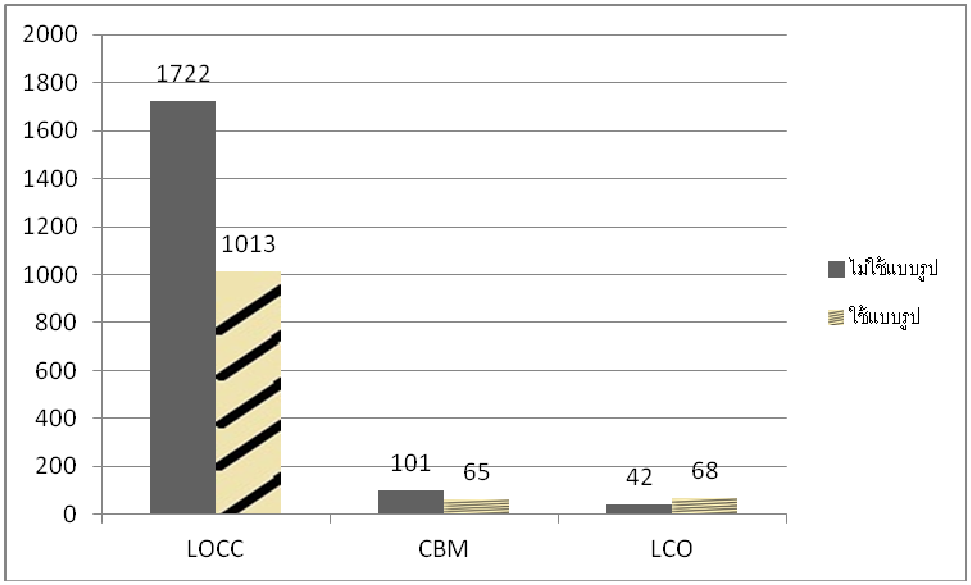
		ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ									ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ		
		คอมโพเนนต์ชื่อ awsi_bl9_flat			คอมโพเนนต์ชื่อ awsi_cm9_flat			คอมโพเนนต์ชื่อ awsi_mmo9			รวมทั้ง 3 คอมโพเนนต์		
	ชื่อคลาส	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO	LOCC	CBM	LCO
23	AuthorizationCheckPoint										11	2	0
24	AuthenticationCheckPoint										10	2	0
25	RequestContext										52	3	54
26	CheckPointFactory										40	1	0
27	AuthenticationEnforcer										23	6	0
28	AuthorizationProvider										16	4	0
29	CheckPoint										4	0	0
30	AuthorizationEnforcer										19	4	0
31	NullCheckPoint										10	1	0
	ผลรวม	505	34	16	1091	66	16	520	32	16	1158	82	70

ตารางที่ 5.5 ผลรวมการวัดสภาพมอดูลาร์ก่อนและหลังนำแบบรูป Authorization มาใช้ในฝั่งเซอร์วิซ

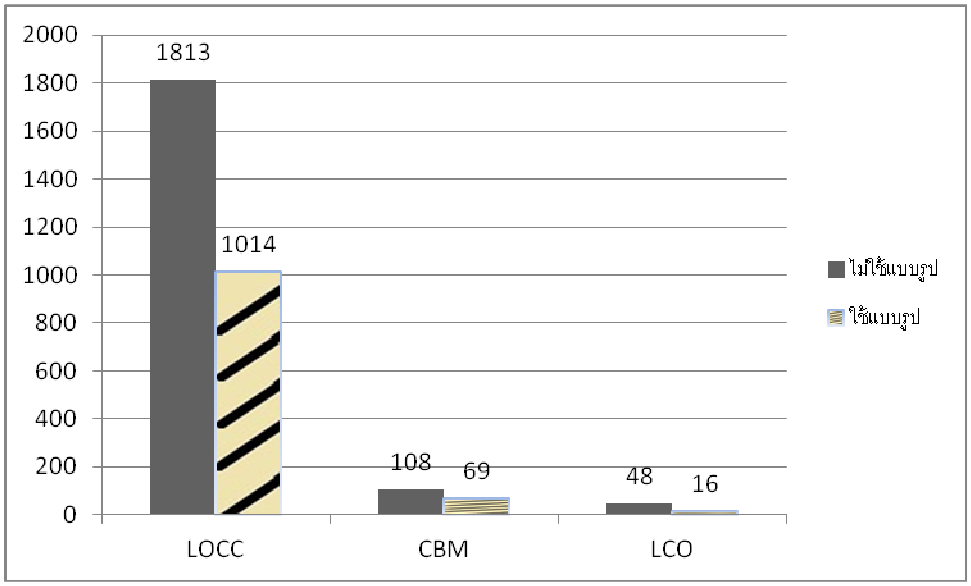
ผลการวัดสภาพมอดูลาร์ก่อนนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ			
ชื่อคอมโพเนนต์	LOCC	CBM	LCO
awsi_bl9_flat	505	34	16
awsi_cm9_flat	1091	66	16
awsi_mmo9	520	32	16
ผลรวมทั้งหมด	**2116**	**132**	**48**
กรณีไม่รวมผลการวัดจากแบบรูป Authentication	**1813**	**108**	**48**
ผลการวัดสภาพมอดูลาร์หลังนำแบบรูป Authentication มาใช้ในฝั่งเซอร์วิซ			
คอมโพเนนต์หลังใช้แบบรูป	LOCC	CBM	LCO
ผลรวม	**1158**	**82**	**70**
กรณีไม่รวมผลการวัดจากแบบรูป Authentication	**1014**	**69**	**16**

เมื่อพิจารณาความมั่นคงด้านการพิสูจน์สิทธิ์ ทั้งก่อนและหลังใช้แบบรูป จำเป็นต้องมีโค้ดด้านการพิสูจน์ตัวจริงผสมร่วมด้วย เพราะความมั่นคงด้านพิสูจน์สิทธิ์จะเกิดขึ้นตามหลังการพิสูจน์ตัวจริงเสมอ ดังนั้นในตารางที่ 5.4 จะในตารางที่ 5.5 จึงมีคลาสของแบบรูปการพิสูจน์ตัวจริงร่วมด้วยได้แก่ คลาส UserProfileDAO, Authentication, AuthenticationCheckPoint, RequestContext และ AuthenticationEnforcer แต่ถ้าผู้วิจัยไม่รวมผลการวัดที่เกี่ยวกับการพิสูจน์ตัวจริงทั้งก่อนและหลังการใช้แบบรูป จะให้ผลการวัดแบบรูป Authorization ที่ไม่มีผลการวัด แบบรูป Authenttication มาผสมร่วมด้วย ดังในตารางที่ 5.5 ซึ่งจะพบว่าตัวเลขค่า LCO ของแบบรูป Authorization ลดลงเนื่องจากไม่ได้นำผลการวัดค่าของคลาส RequestContext จากแบบรูป Authentication ซึ่งจะมีค่า LCO ที่มาก มาพิจารณานั่นเอง

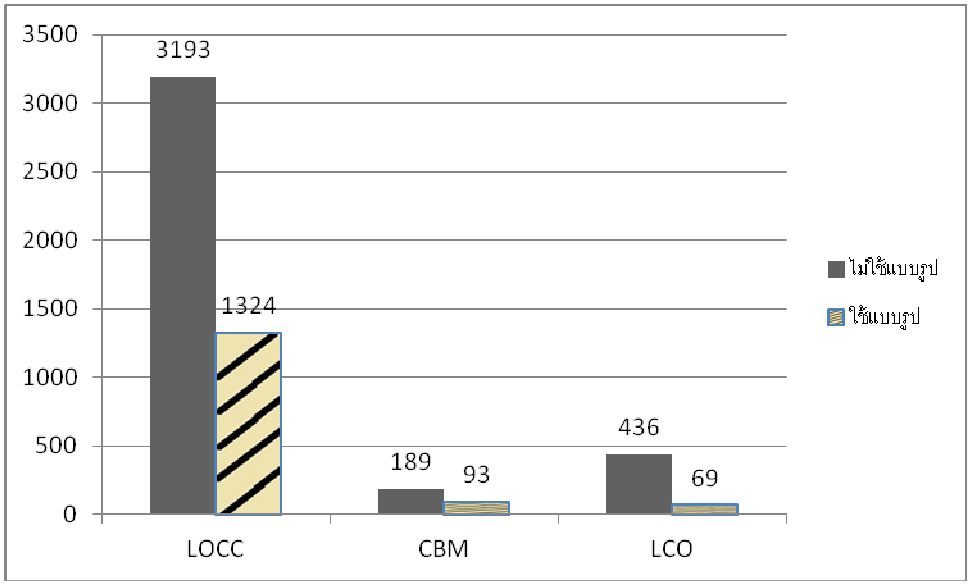
ผลจากการวัดสภาพมอดูลาร์ในฝั่งเซอร์วิซที่แสดงไว้ในตารางที่ 5.5 สามารถนำมาวาดเป็นกราฟเพื่อให้ง่ายต่อการแปลความได้ดังภาพที่ 5.2 ภาพที่ 5.3 และภาพที่ 5.4 (ซึ่งผลการวัดของแบบรูป Authorization ในภาพที่ 5.3 จะไม่ได้รวมผลของการวัดแบบรูป Authentication เข้ามาวาดเป็นกราฟ)



ภาพที่ 5.2 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Authentication ในฝั่งเซอร์วิซ



ภาพที่ 5.3 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Authorization ในฝั่งเซอร์วิซ



ภาพที่ 5.4 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Security Session ในฝั่งเซอร์วิซ

สรุป จากกราฟในภาพที่ 5.2 ภาพที่ 5.3 และภาพที่ 5.4 ซึ่งเป็นผลการวัดของแบบรูป Authentication, Authorization และ Security Session ตามลำดับ ผลการทดลองจะให้ค่า LOCC, CBM และ LCO ที่มีค่าลดลงเมื่อใช้แบบรูป แสดงว่าการใช้แบบรูปที่งานวิจัยนำเสนอส่งผลดีขึ้นในแง่จำนวนบรรทัดของโค้ดและการต่อประกบระหว่างมอดูลซึ่งมีค่าลดลง และมีการเชื่อมแน่นในมอดูลดีขึ้น ยกเว้นกรณีของแบบรูปด้านการพิสูจน์ตัวจริงซึ่งค่า LCO กลับเพิ่มขึ้น

5.2 ผลการทดลองวัดสภาพมอดูลาร์ในฝั่งไคลเอนต์

งานวิจัยนี้ใช้ตัววัดได้แก่ LOCC, CBM, LCO, CDC และ CDO ทำการวัดสภาพมอดูลาร์ก่อนและหลังนำวิธีเอโอพีมาใช้ในฝั่งไคลเอนต์ ซึ่งได้ผลดังตารางที่ 5.6

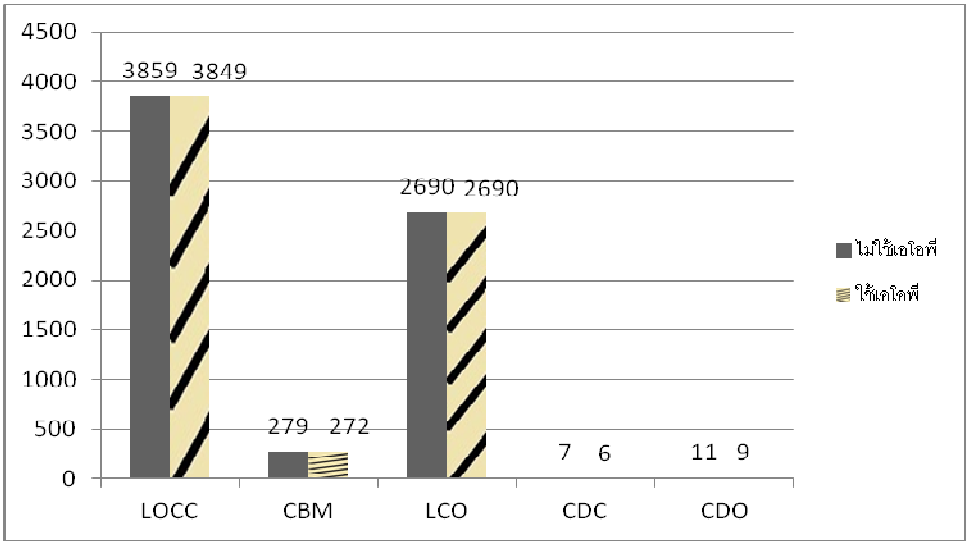
ตารางที่ 5.6 ผลการทดลองวัดสภาพมอดูลาร์ในฝั่งไคลเอนต์ก่อนและหลังนำเอโอพีมาใช้

	จำนวนคลาส	LOCC	CBM	LCO	CDC	CDO
แบบรูป Authentication						
ไม่ใช้เอโอพี	58	3859	279	2690	7	11
ใช้เอโอพี	58	3849	272	2690	6	9
แบบรูป Authorization						
ไม่ใช้เอโอพี	58	3859	279	2690	7	11
ใช้เอโอพี	58	3849	272	2690	6	9

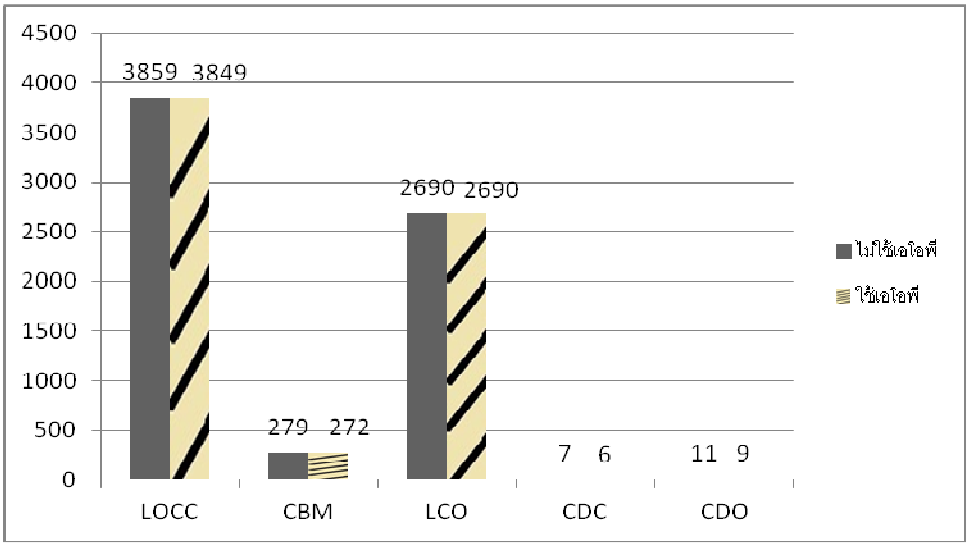
ตารางที่ 5.7 ผลการทดลองวัดแบบรูปในฝั่งไคลเอนต์ก่อนและหลังนำเอโอพีมาใช้ (ต่อ)

	จำนวนคลาส	LOCC	CBM	LCO	CDC	CDO
แบบรูป Security Session						
ไม่ใช้เอโอพี	58	4047	282	2690	7	13
ใช้เอโอพี	58	3872	275	2690	6	9

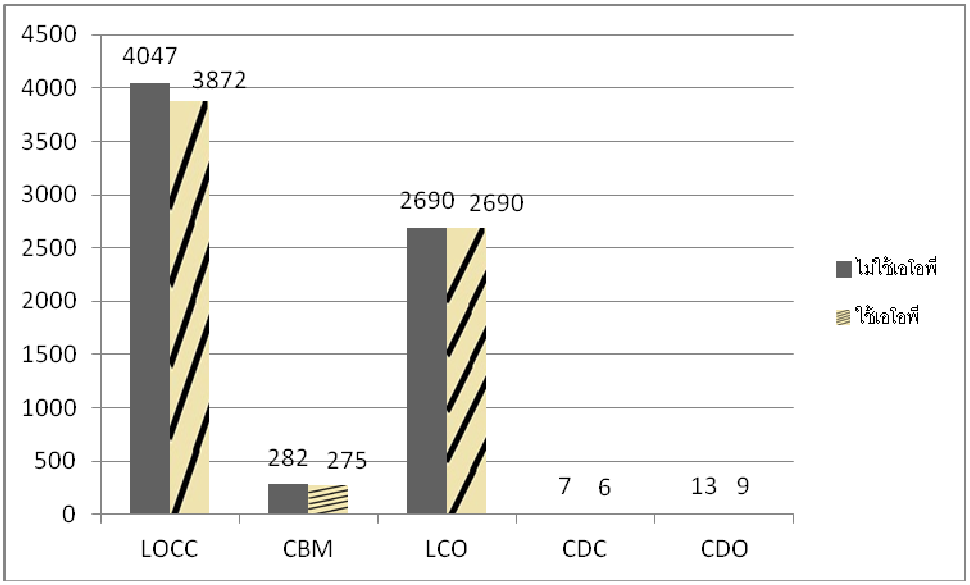
จากผลจากการวัดสภาพมอดูลาร์ในฝั่งไคลเอนต์ที่แสดงไว้ในตารางที่ 5.6 สามารถนำมาวาดเป็นกราฟเพื่อให้ง่ายต่อการแปลความได้ดังภาพที่ 5.5 ภาพที่ 5.6 และภาพที่ 5.7



ภาพที่ 5.5 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Authentication ในฝั่งไคลเอนต์



ภาพที่ 5.6 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Authorization ในฝั่งไคลเอนต์



ภาพที่ 5.7 กราฟแสดงผลการวัดสภาพมอดูลาร์สำหรับแบบรูป Security Session ในฝั่งไคลเอนต์

สรุป จากกราฟในภาพที่ 5.5 ภาพที่ 5.6 และภาพที่ 5.7 ซึ่งเป็นผลการวัดของแบบรูป Authentication, Authorization และ Security Session ตามลำดับ โดยผลการทดลองตัวเลขที่ได้จากค่า LOCC, CBM, CDC, CDO ลดลง แสดงว่าการโปรแกรมเชิงแง่มุมส่งผลดีขึ้นในแง่ของจำนวนบรรทัด การต่อประกบระหว่างมอดูล และการแพร่กระจายของคอนเซิร์น ซึ่งมีค่าลดลง ส่วนค่า LCO ค่าเท่าเดิม แสดงว่าวิธีเชิงแง่มุมยังไม่ส่งผลในการปรับปรุงความเชื่อมแน่นภายในมอดูล

บทที่ 6
บทสรุป

6.1 สรุปผลของวิทยานิพนธ์

แบบรูปที่งานวิจัยนำเสนอเกิดจากการผสมผสานข้อดีจากแบบรูปที่มีอยู่ เพื่อให้ได้ความต้องการด้านความมั่นคงได้แก่ การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง ครบทุกประการ พร้อมทั้งสนับสนุนการออกแบบเชิงวัตถุสำหรับการพัฒนาเว็บเซอร์วิซ อีกทั้งการประยุกต์ใช้วิธีเชิงแง่มุมยังช่วยปรับปรุงคุณภาพโค้ดฝั่งไคลเอนต์ด้วย โดยงานวิจัยได้นำเสนอ 3 แบบรูปด้วยกันได้แก่ Authentication, Aurthorization และ Security Session ซึ่งจากผลการทดลองผู้วิจัยเห็นว่าแม้ว่าเทคนิคทั้งสองจะไม่ส่งผลกระทบในเชิงบวกต่อสภาพมอดูลาร์ในทุกด้านตามตัววัดที่ใช้ โดยเฉพาะในด้านความเชื่อมแน่นที่ให้ผลการวัดไม่ค่อยดี แต่อย่างไรก็ตามค่าจากตัววัดส่วนใหญ่บ่งบอกว่าสามารถช่วยปรับปรุงสภาพมอดูลาร์ของระบบเว็บเซอร์วิซโดยรวมให้ดีขึ้นได้

6.2 ปัญหาและข้อจำกัดของงานวิจัย

จากการดำเนินงาน พบปัญหาและข้อจำกัดในการทดลอง ดังต่อไปนี้

1) เนื่องจากจำนวนตัวอย่างการทดลองกับระบบเว็บเซอร์วิซมีเพียงตัวอย่างเดียว ยังไม่ได้ทดลองกับตัวแทนของระบบเว็บเซอร์วิซอื่น ๆ ที่หลากหลาย จึงยังไม่มีผลการเปรียบเทียบกับระบบอื่น ๆ
2) เนื่องจากหน่วยทดลองที่ใช้ในฝั่งเซอร์วิซ เป็นระบบค่อนข้างใหญ่และซับซ้อน อีกทั้งผู้วิจัยไม่สามารถนำโค้ดทั้งหมดหรือนำระบบทั้งหมดมาทดลองได้ นำมาได้เพียงบางส่วน อีกทั้งยังติดขัดเรื่องข้อจำกัดด้านนโยบายความมั่นคงของบริษัท ที่โค้ดและข้อมูลบ้างอย่างไม่อาจเปิดเผยได้ ทำให้ผู้พัฒนาเสียเวลาไปกับขั้นตอนเพื่อพัฒนาโค้ดเพิ่มเติม และออกแบบฐานข้อมูลเพิ่มเติมอยู่พอสมควร เพื่อให้ได้โค้ดของระบบเว็บเซอร์วิซฝั่งไคลเอนต์และเซอร์วิซ ที่สามารถทดลองและวัดผลได้
3) จำนวนหน่วยทดลองที่ใช้ยังน้อยพอสมควรมีเพียง 8 เซอร์วิซ และฝั่งไคลเอนต์ 10 เซอร์วิซ โดยมีจำนวนคลาสฝั่งเซอร์วิซ 140 คลาส และคลาสฝั่งไคลเอนต์ 174 คลาส ทำให้ตัวเลขการวัดผล โดยเฉพาะฝั่งไคลเอนต์ก่อนใช้กับหลังใช้แบบรูป ให้ผลไม่แตกต่างกันมากนัก
4) เนื่องด้วยระบบที่นำมาทดลองมีค่าความเชื่อมแน่นค่อนข้างดีอยู่ก่อนแล้ว จึงอาจทำให้ผลการวัดตัวเลขค่าความเชื่อมแน่นไม่ได้ดีขึ้นอย่างมีนัยสำคัญ

6.3 ข้อเสนอแนะ

ประเด็นที่งานวิจัยนี้ยังไม่ได้ศึกษา และสามารถวิจัยเพิ่มเติมได้ในอนาคตมีดังต่อไปนี้

1) เนื่องจากงานวิจัยได้ประยุกต์ใช้กับความมั่นคงด้านได้แก่ การพิสูจน์ตัวจริง การพิสูจน์สิทธิ์ และเซสชันด้านความมั่นคง แต่ยังมีความมั่นคงแบบอื่นที่ยังไม่ได้พิจาณา เมื่อพิจารณาจากหนังสือของ Steel [4] ยังมีความมั่นคงอื่น ๆ อีกมาก ได้แก่ การสอบบัญชีและการตามรอย (Auditability and Traceability) บูรณภาพของข้อมูล (Data Integrity) การรักษาความลับของข้อมูล (Data Confidentiality) การห้ามปฏิเสธความรับผิดชอบ (Non-repudiation) สภาพพร้อมใช้งานและการบริการอย่างต่อเนื่อง (Availability and Service Continuity) การยืนยันตัวตนเพียงครั้งเดียวและการมอบหมายอำนาจ (Single Sign-on and Delegation) การจัดการเอกลักษณ์และนโยบาย (Identity and Policy Management) และ ความสามารถในการทำงานร่วมกันของความมั่นคง (Security Interoperability) [4] ซึ่งในอนาคตสามารถเสนอแบบรูปเพิ่มเติมเพื่อรองรับความต้องการเหล่านี้ได้
2) ในอนาคตงานวิจัยสามารถทำการทดสอบทางสถิติว่าแบบรูปที่เสนอและการโปรแกรมเชิงแง่มุมส่งผลดีต่อสภาพมอดูลาร์อย่างมีนัยสำคัญหรือไม่ ขณะเดียวกันอาจเพิ่มตัววัดในการโปรแกรมเชิงวัตถุและเชิงแง่มุมอื่น ๆ เข้ามาใช้วิเคราะห์ทางสถิติเพิ่มเติม
3) ในอนาคตสามารถนำไปทดลองกับตัวอย่างเว็บเซอร์วิซระบบอื่น ๆ ที่หลากหลาย ด้วยขนาดจำนวนเซอร์วิซ และจำนวนคลาส ที่ใช้ทดลองที่มากขึ้น จะทำให้สามารถเปรียบเทียบผลการทดลองได้ดียิ่งขึ้น

รายการอ้างอิง

[1] Haas, H., and Brown, A. Web Services Glossary [Online]. 2004. Available from: http://www.w3.org/TR/ws-gloss/ [2012, October]

[2] IEEE Std 610.12-1990. The IEEE Standard Glossary of Software Engineering Terminology. 1990.

[3] Fenton, N. E., and Pfleeger, S. L. Software Metrics: A Rigorous and Practical Approach. PWS Publishing Company, 1997.

[4] Steel, C., Nagappan, R., and Lai, R. Core Security Patterns: Best Practices and Strategies for J2EE™, Web Services, and Identity Management. Prentice Hall PTR / Sun Micros, 2005.

[5] Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., and Sommerlad, P. Security Patterns: Integrating Security and Systems Engineering. John Wiley and Sons, 2005.

[6] OASIS Standard 200401. Web Services Security: SOAP Message Security 1.0 (WS-Security 2004) [Online]. 2004. Available from: http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf [2012, October].

[7] Schumacher, M., and Roedig, U. Security Engineering with Patterns. Proceedings of the 8th Conference on Pattern Languages of Programs (PLoP 2001), 2001.

[8] Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M., and Irwin, J. Aspect-Oriented Programming. Proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP 1997), pp. 220 -242. Finland: LNCS 1241, 1997.

[9] Kiczales, G., Hilsdale, E., Hugunin, J., Kersten, M., Palm, J., and Griswold, W. G. An Overview of AspectJ. Proceedings of the 15th European Conference on Object-Oriented Programming (ECOOP 2001), pp. 327-353. 2001.

[10] Sonchaiwanich, E., Zhao, J., Dowin, C., and McRoberts, M. Using AOP to Separate SOA Security Concerns from Application Implementation. Proceedings of the 2010 Military Communications Conference (MILCOM 2010), pp. 470-474. 2010.

[11] Edge, C., and Mitropoulos, F. Quantitative Analysis of Modularity Tradeoffs with AspectJ Web- Tier Security Patterns. Proceedings of 3rd Workshop on Empirical Evaluation of Software Composition Techniques (ESCOT 2010), pp.1-8. 2010.

[12] Halkidis, S. T., Chatzigeorgiou, A., and Stephanides, G. Quantitative Evaluation of Systems with Security Patterns Using a Fuzzy Approach. Proceedings of OTM Workshops 2006, pp. 554-564. LNCS 4277, 2006.

[13] Chawla, R., and Mehta, N. Software Security Patterns in Security Engeering. International Journal of Research in IT & Management (IJRIM) 2 (February 2012): pp. 327-332.

[14] Hachani, O., and Bardou, D. Using Aspect-Oriented Programming for Design Patterns Implementation. Proceedings of Workshop on Reuse in Object-Oriented Information Systems Design, 2002.

[15] Georg, G., Ray, I., and France, R. Using Aspects to Design a Secure System. Proceedings of the Eighth International Conference on Engineering of Complex Computer Systems (ICECCS 2002), pp. 117-126. 2002.

[16] Garcia, A., Sant'Anna, C., Figueiredo, E., Kulesza, U., Lucena, C., and von Staa, A. Modularizing Design Patterns with Aspects: A Quantitative Study. Proceedings of the 4th International Conference on Aspect-Oriented Software Development (AOSD 2005), pp. 3-14. 2005.

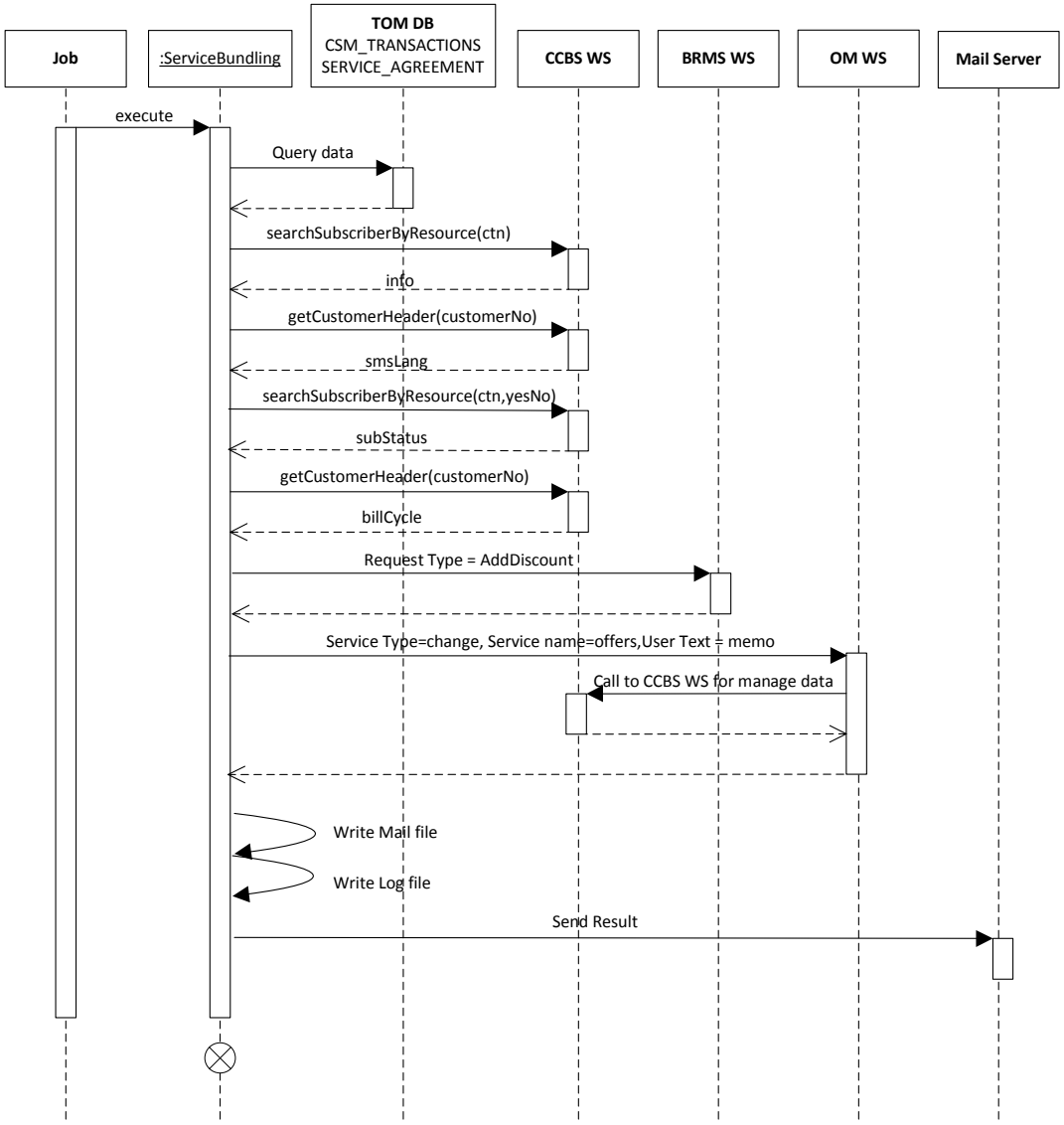
[17] Hannemann, J., and Kiczales, G. Design Pattern Implementation in Java and AspectJ. Proceedings of the 17th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA 2002), pp. 162-173. ACM, 2002.

[18] Edge, C., and Mitropoulos, F. Improving Security Design Patterns with Aspect-Oriented Strategies. Proceedings of the 50th Annual Southeast Regional Conference (ACMSE 2012), Tuscaloosa, Alabama: ACM, 2012.

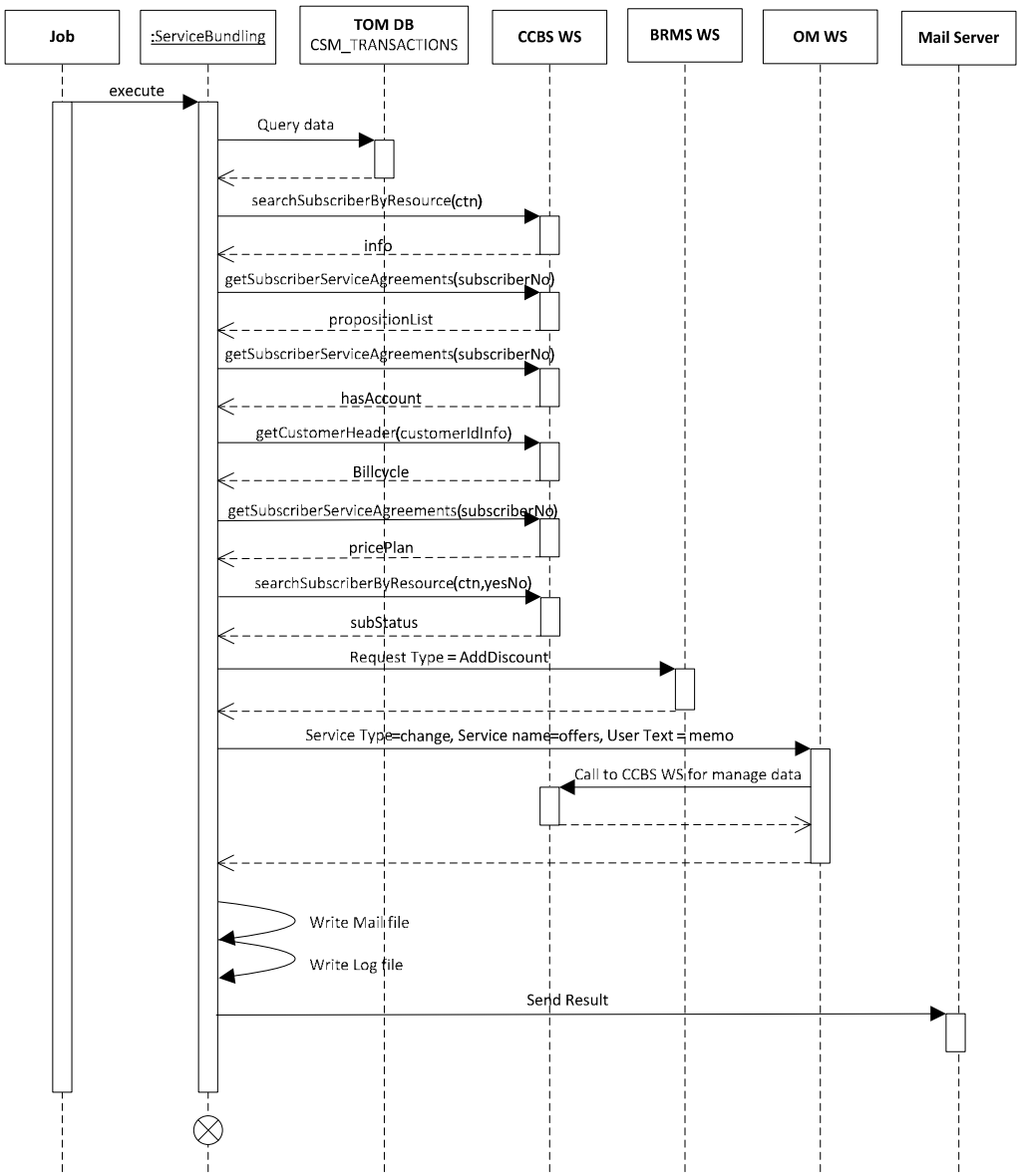
[19] Greenwood, P., Garcia, A., Rashid, A., Figueiredo, E., Sant' Anna, C., Cacho, N., et al. On the Contributions of an End-to-End AOSD Testbed. Proceedings of Early Aspects at ICSE: Workshops in Aspect-Oriented Requirements Engineering and Architecture Design (EARLYASPECTS 2007), 2007.

ภาคผนวก

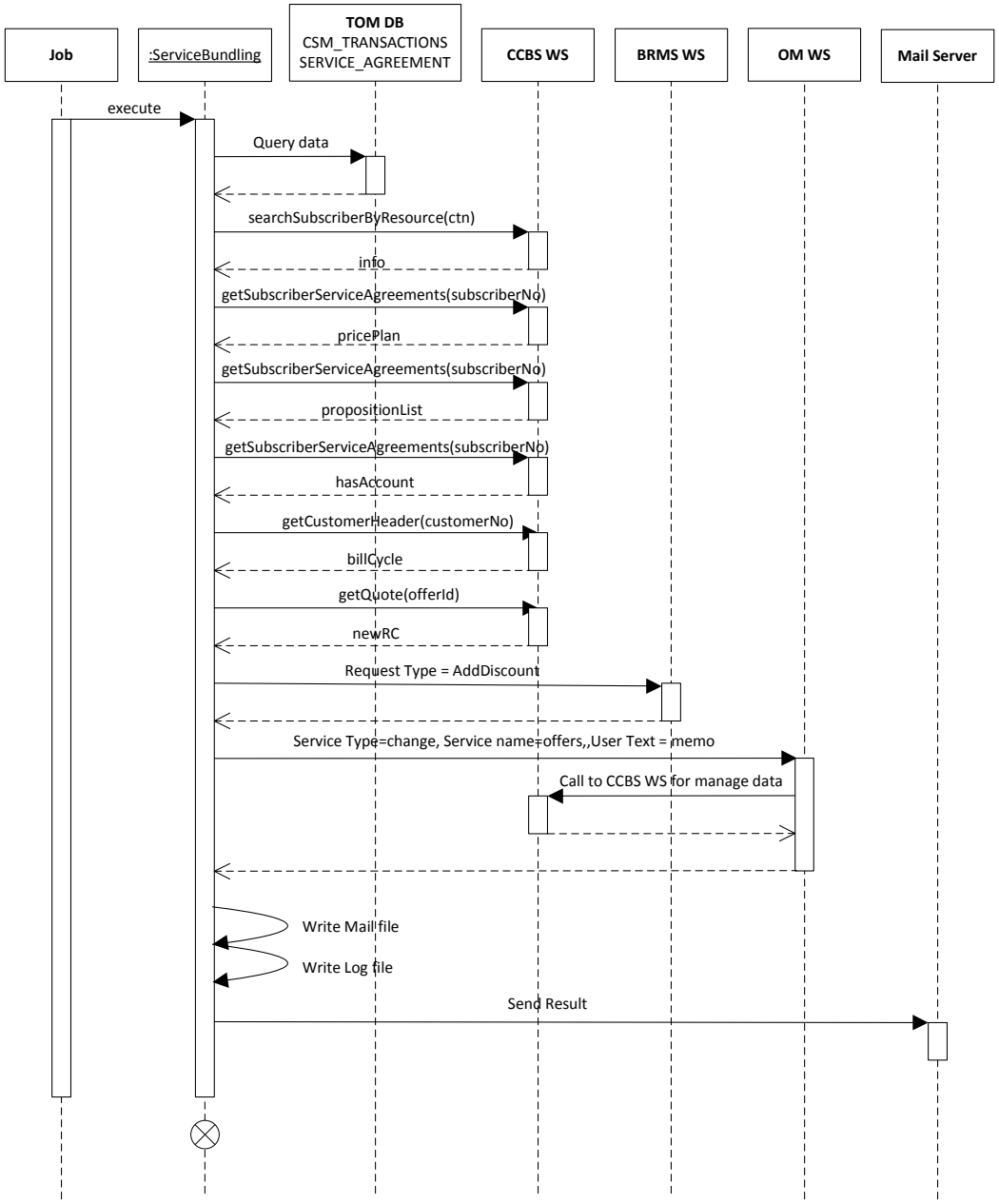
ภาคผนวก ก
แผนภาพลำดับของระบบตัวอย่างทดลอง



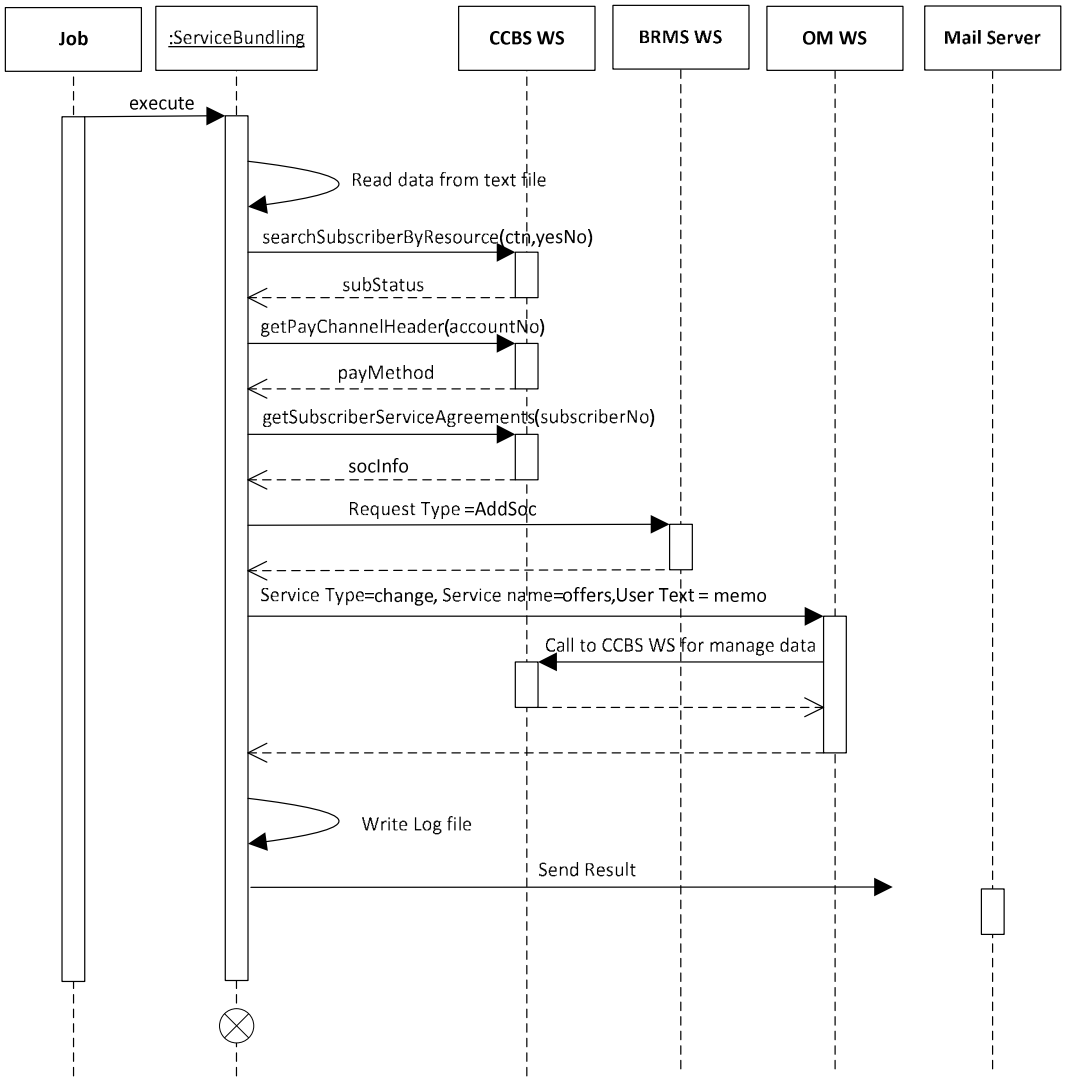
ภาพที่ ก.1 แผนภาพลำดับของ Job ชื่อ “AddDiscountHutch”



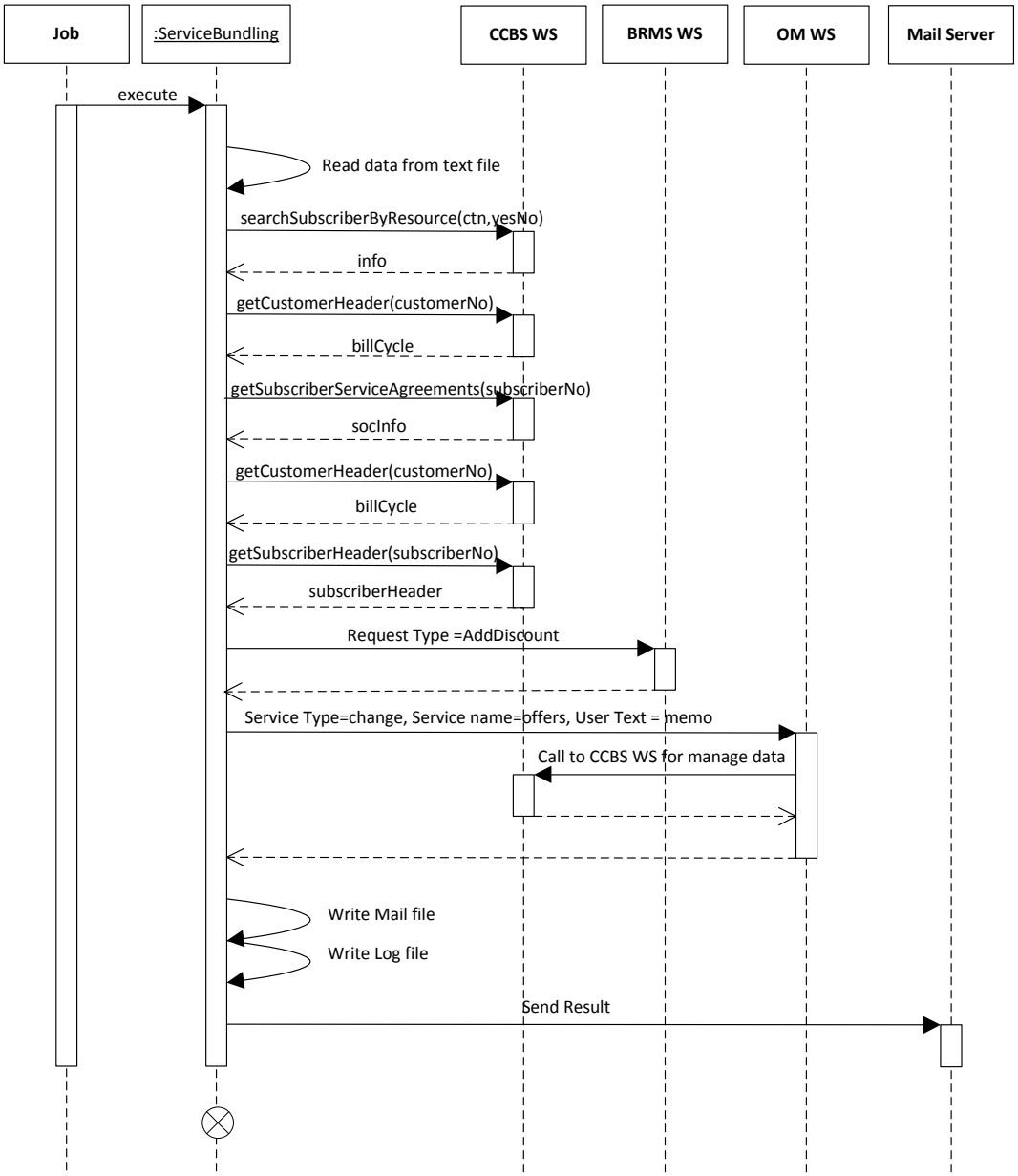
ภาพที่ ก.2 แผนภาพลำดับของ Job ชื่อ “ AddDiscountDIR040”



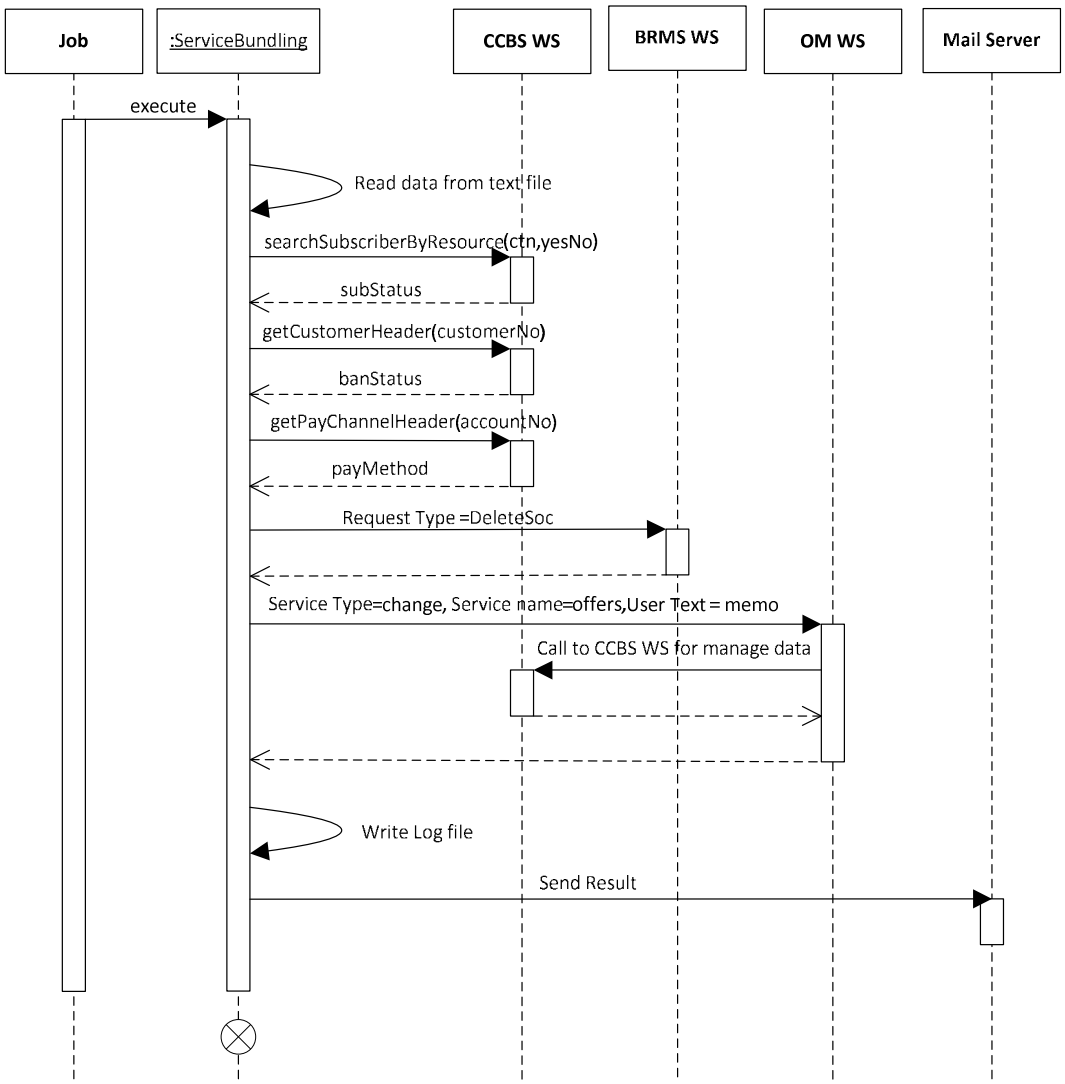
ภาพที่ ก.3 แผนภาพลำดับของ Job ชื่อ “ AddDiscountTMH”



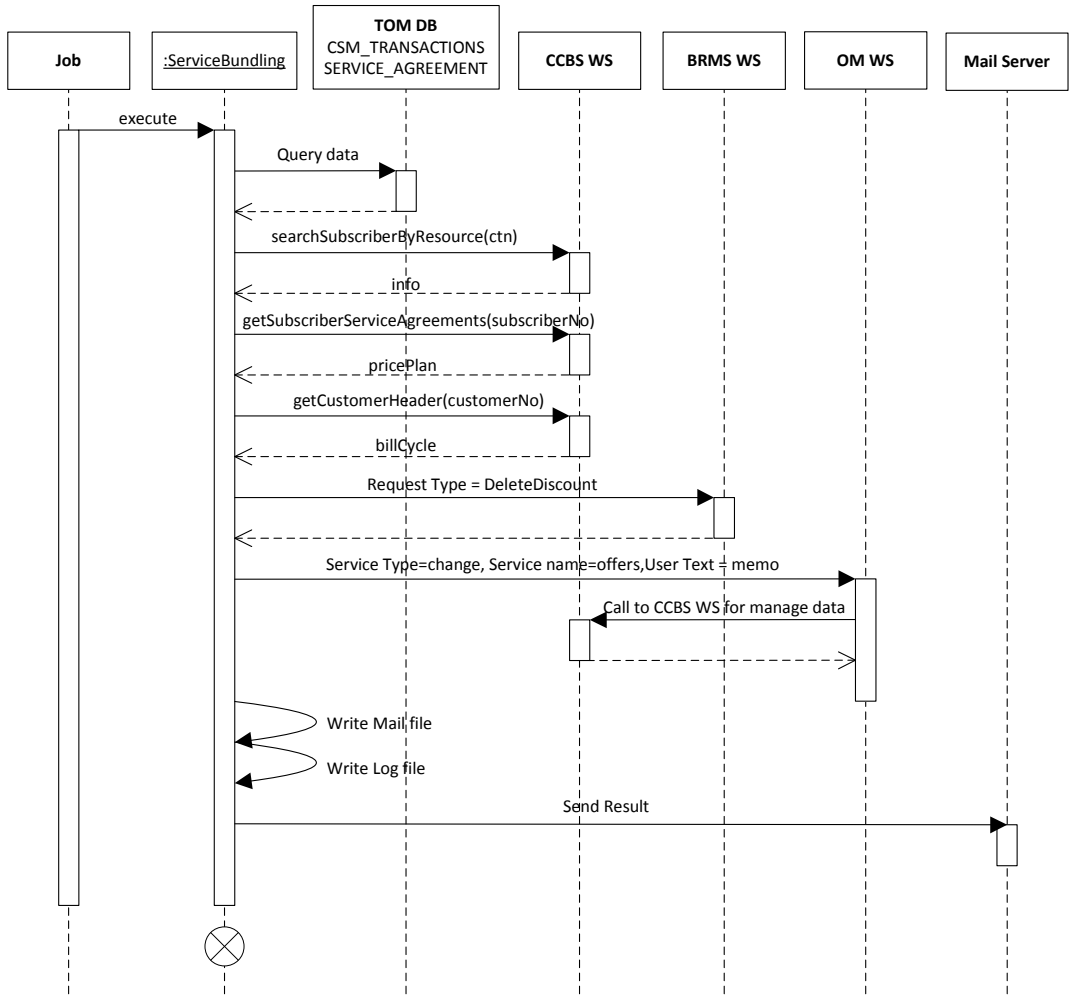
ภาพที่ ก.4 แผนภาพลำดับของ Job ชื่อ “AddSocDRT”



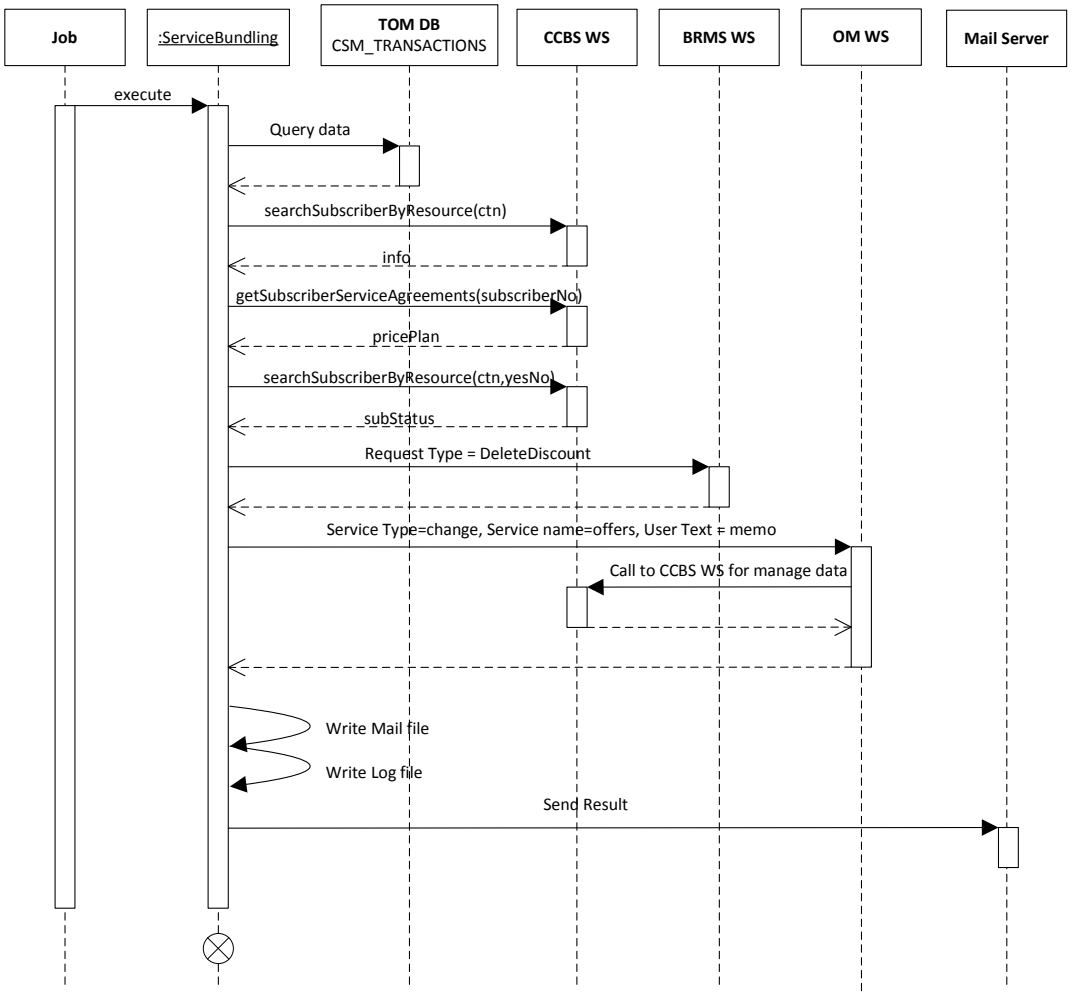
ภาพที่ ก.5 แผนภาพลำดับของ Job ชื่อ “ AddSocRBTDcDGC035”



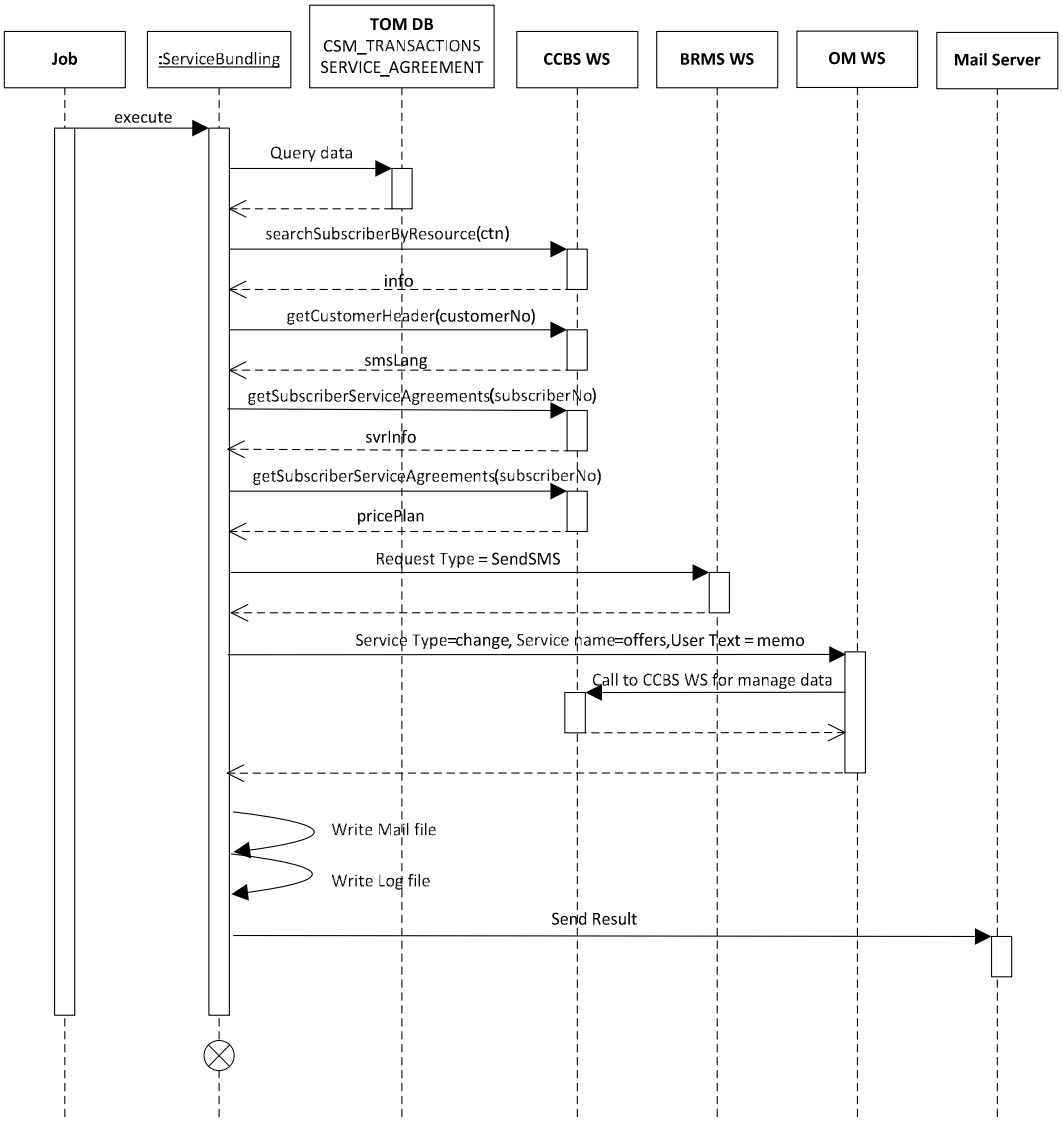
ภาพที่ ก.6 แผนภาพลำดับของ Job ชื่อ “ DelSocDRT”



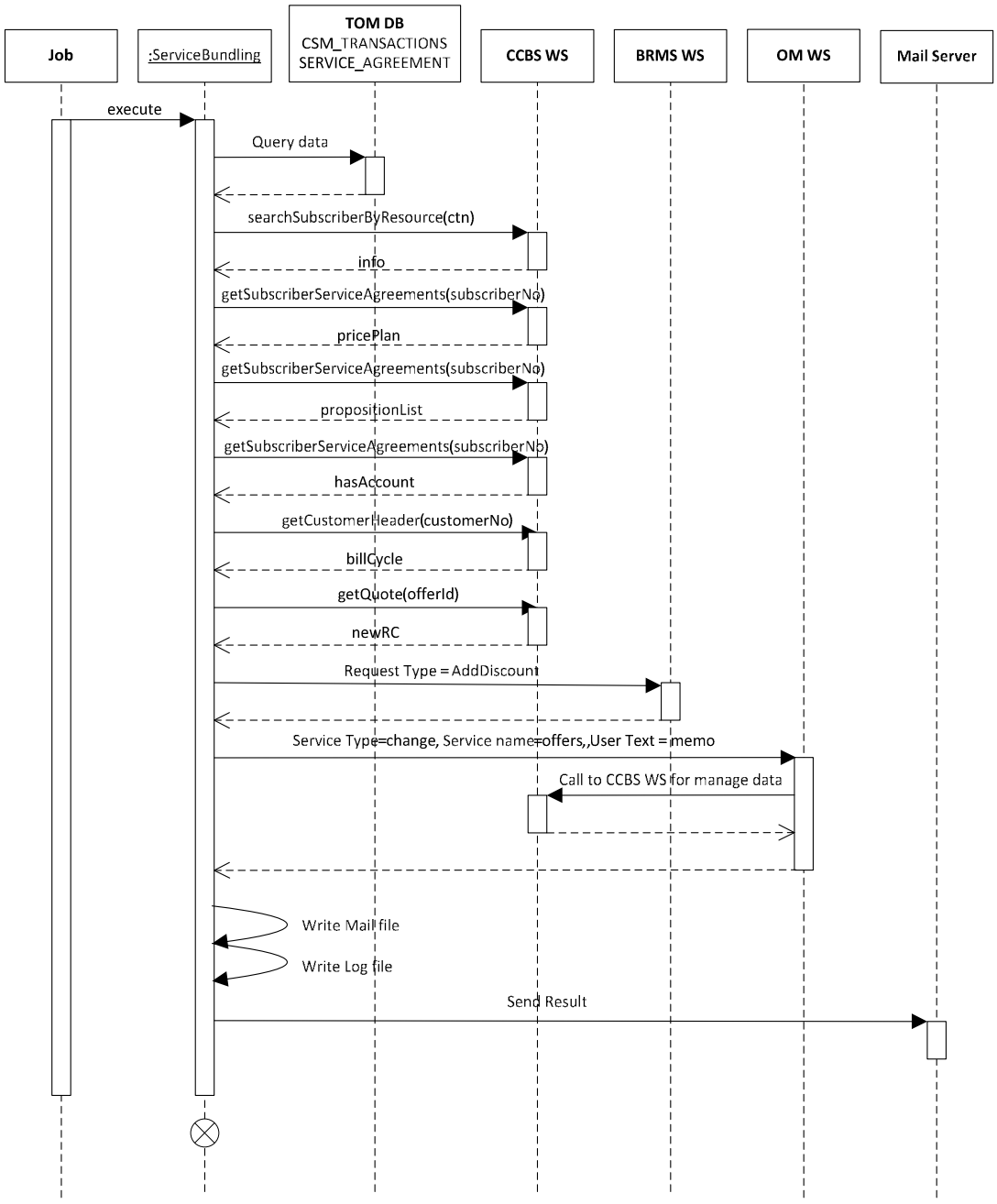
ภาพที่ ก.7 แผนภาพลำดับของ Job ชื่อ “ ExpireDiscountHutch”



ภาพที่ ก.8 แผนภาพลำดับของ Job ชื่อ “ ExpDiscountDIR040”



ภาพที่ ก.9 แผนภาพลำดับของ Job ชื่อ “ UCR_MARG7WK663_InformChangePPSMS”



ภาพที่ ก.10 แผนภาพลำดับของ Job ชื่อ “UCR_MARG82RAES_InformNewPPSMS”

ประวัติผู้เขียนวิทยานิพนธ์

นายจตุรพัชร์ พัฒนทรงศิวิไล เกิดเมื่อวันที่ 1 กันยายน พ.ศ. 2524 ที่จังหวัดนครราชสีมา สำเร็จการศึกษาระดับปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น ในปีการศึกษา 2546 และได้เข้าศึกษาต่อในหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมซอฟแวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2554 และมีผลงานวิจัยตีพิมพ์ในบทความวิชาการร่วมกับอาจารยที่ปรึกษาวิทยานิพนธ์ ในการประชุมวิชาการ The 2013 International Computer Science and Engineering Conference (ICSEC 2013) โดยบทความชื่อ “การประยุกต์แบบรูปความมั่นคงและการโปรแกรมเชิงแง่มุมกับเว็บเซอร์วิส”