การประกอบเว็บเซอร์วิสอย่างอัตโนมัติด้วยข้อกำหนดอาวล์-เอสโพรเซสโมเดล ที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่

นายเติมศักดิ์ วะสิโน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2550

AUTOMATIC WEB SERVICE COMPOSITION USING OWL-S PROCESS MODEL SPECIFICATIONS WITH CONSTRAINTS ON FUNCTIONAL BEHAVIOUR

Mr.Termsak Wasino

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Engineering Program in Computer Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2007

หัวข้อวิทยานิพนธ์	การประกอบเว็บเซอร์วิสอย่างอัตโนมัติด้วยข้อกำหนดอาวล์-เอสโพร เซสโมเดลที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่
โดย	นายเติมศักดิ์ วะสิโน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(ศาสตราจารย์ ดร.ดิเรก ลาวัณย์ศิริ)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(อาจารย์ ดร.ยรรยง เต็งอำนวย)

..... อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา)

..... กรรมการ

(อาจารย์ ดร.วิษณุ โคตรจรัส)

.....กรรมการ

(อาจารย์ ดร.เบญจพร ลิ้มธรรมาภรณ์)

เติมศักดิ์ วะสิโน : การประกอบเว็บเซอร์วิสอย่างอัตโนมัติด้วยข้อกำหนดอาวล์-เอสโพรเซสโมเดลที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่. (AUTOMATIC WEB SERVICE COMPOSITION USING OWL-S PROCESS MODEL SPECIFICATIONS WITH CONSTRAINTS ON FUNCTIONAL BEHAVIOUR) อ. ที่ปรึกษา : ผศ. ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา, 83 หน้า.

การประกอบเว็บเซอร์วิสได้กลายเป็นหัวข้อสำคัญในการค้นคว้าวิจัยในด้านการคำนวณแบบอิงบริการ เนื่องจากความต้องการที่เพิ่มขึ้นในการสร้างบริการที่ตรงตามวัตถุประสงค์การใช้งานของผู้ใช้บริการโดยอาศัยการประกอบกันของบริการอื่นๆ รวมไปถึงการประกอบบริการเข้าด้วยกันยังเป็นกระบวนการที่ซับซ้อนอยู่ งานวิจัยนี้นำเสนอการประกอบเว็บเซอร์วิสอย่างอัตโนมัติด้วยข้อกำหนดอาวล์-เอสโพรเซสโมเดลที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่จากเป้าหมายของการประกอบบริการซึ่งนักออกแบบเว็บเซอร์วิสกำหนดไว้ จะมีการพิจารณาความเข้ากันได้ของเว็บเซอร์วิสที่นำมาประกอบ ซึ่งแต่ละเว็บเซอร์วิสจะมีข้อกำหนดอาวล์-เอสโพรเซสโมเดลซึ่งระบุพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสนั้นอยู่ การพิจารณาความเข้ากันได้จะดูจาก อินพุท เอาท์พุท เงื่อนไขก่อนการทำงาน เอฟเฟ็กต์ และข้อบังคับที่เกี่ยวข้องต่างๆของเว็บเซอร์วิสเหล่านั้น จากนั้นจะคำนวณหาการประกอบที่เหมาะสมที่สุดโดยใช้อัลกอริทึมเส้นทางที่สั้นที่สุดของไดจ์สตรา และ มีการสร้างเว็บเซอร์วิสประกอบในรูปของเอกสารบีเพล

ภาควิชา.....วิศวกรรมคอมพิวเตอร์..... ลายมือชื่อนิสิต.....เติมศักดิ์ วะสิโน.....

สาขาวิชา...วิศวกรรมคอมพิวเตอร์..... ลายมือชื่ออาจารย์ที่ปรึกษา.....ทวิตีย์ เสนีวงศ์ ณ อยุธยา

ปีการศึกษา.....2550.....

4870302921: MAJOR COMPUTER ENGINEERING

KEY WORD: WEB SERVICE COMPOSITION / SEMANTIC WEB SERVICES / OWL-S PROCESS MODEL / ONTOLOGY

TERMSAK WASINO : AUTOMATIC WEB SERVICE COMPOSITION USING OWL-S PROCESS MODEL SPECIFICATIONS WITH CONSTRAINTS ON FUNCTIONAL BEHAVIOUR. THESIS ADVISOR : ASST. PROF. TWITTIE SENIVONGSE, Ph.D., 83 pp.

Web services composition has been a key research issue in service-oriented computing due to the increasing demand for composite services that can serve service consumers' purposes, together with the complexity of the composition process. This thesis proposes an approach to automatic composition of Web services by using OWL-S Process Model with Constraints on Functional Behaviour. Given a composition goal by a Web service designer, the approach examines process specifications of the Web services which are represented by OWL-S process model and determines compatibility between functional behaviour of those Web services, i.e. their inputs, outputs, preconditions, and effects as well as their associated constraints. An optimal composition is identified by using Dijkstra's shortest path algorithm and a composite service can be generated as a BPEL script.

Department.....Computer Engineering.....Student's Signature.....Termsak WASINO

Field of Study.....Computer Engineering.....Advisor's Signature.....Twittie Senivongse

Academic Year.....2007.....

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลงด้วยความกรุณาเป็นอย่างสูงของผู้ช่วยศาสตราจารย์ ดร.ทวิตีย์ เสนีวงศ์ ณ อยุธยา อาจารย์ที่ปรึกษาวิทยานิพนธ์ของข้าพเจ้า ที่ให้คำปรึกษา ให้กำลังใจ และแก้ปัญหาให้ข้าพเจ้ามาตลอด สิ่งต่าง ๆ ที่ข้าพเจ้าทำผิดพลาดไปข้าพเจ้ากราบขออภัย และกราบขอบพระคุณอาจารย์เป็นอย่างสูงไว้ ณ ที่นี้ด้วย

ขอขอบพระคุณ อาจารย์ ดร.ยรรยง เต็งอำนวย ประธานกรรมการสอบ อาจารย์ ดร.วิษณุ โคตรจรัส และ อาจารย์ ดร.เบญจพร ลิ้มธรรมาภรณ์ กรรมการสอบของข้าพเจ้าที่กรุณาให้คำแนะนำทำให้งานวิจัยชิ้นนี้มีความถูกต้องและสมบูรณ์มากขึ้น และขอขอบพระคุณคำชี้แนะดี ๆ ที่จะเป็นประโยชน์ต่อการดำรงชีวิตของข้าพเจ้าต่อไปในภายภาคหน้า

ขอขอบพระคุณอาจารย์ทุกท่านที่ให้ความรู้แก่ข้าพเจ้าเพื่อใช้ในการทำงานวิจัยนี้ รวมถึงคำแนะนำดี ๆ ที่ให้แก่ข้าพเจ้าตลอดมา

ขอขอบคุณพี่ ๆ น้อง ๆ เพื่อนๆ ที่ช่วยให้ชีวิตการทำงานวิจัยมีความสุขสนุกสนาน ขอบคุณสำหรับคำแนะนำดี ๆ และกำลังใจที่ให้มาตลอด

ขอขอบพระคุณ คุณพ่อ และคุณแม่ที่เลี้ยงดูข้าพเจ้า ให้กำลังใจข้าพเจ้า ให้อภัยข้าพเจ้าตลอดมา คุณความดีของวิทยานิพนธ์ฉบับนี้ ขอมอบเป็นเครื่องบูชาคุณบิดา มารดา ไว้ ณ โอกาสนี้

สารบัญ

หน้า

บทที่ หน้า

สารบัญภาพ

ภาพประกอบ หน้า

ภาพประกอบ หน้า

สารบัญตาราง

ตาราง หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เว็บเซอร์วิส (Web Services) [1] เป็นมาตรฐานที่ได้รับความนิยมอย่างมากในปัจจุบัน เนื่องจากเป็นมาตรฐานเปิดในการติดต่อสื่อสารแลกเปลี่ยนข้อมูลและบริการ (Services) ผ่านเครือข่ายอินเทอร์เน็ต (Internet) โดยไม่ขึ้นกับระบบปฏิบัติการ หรือภาษาโปรแกรมที่ใช้ในการพัฒนา ซึ่งเป็นประโยชน์ในการเพิ่มความยืดหยุ่นและลดต้นทุนในการพัฒนาแอพพลิเคชัน (Application)

การประกอบเว็บเซอร์วิส (Web Service Composition) เป็นส่วนสำคัญในการคำนวณแบบอิงบริการ (Service-Oriented Computing) [2] เนื่องจากปริมาณความต้องการเว็บเซอร์วิสที่ซับซ้อนที่ตรงความต้องการของผู้ใช้บริการมีมากขึ้น ซึ่งต้องยอมรับว่าเว็บเซอร์วิสเพียงเว็บเซอร์วิสเดียวไม่สามารถตอบสนองความต้องการเหล่านี้ แต่ถ้าเว็บเซอร์วิสนั้นทำงานร่วมกับเว็บเซอร์วิสตัวอื่นตามลำดับงานที่เหมาะสมแล้ว จะสามารถทำงานตามที่ผู้ใช้บริการต้องการได้อย่างลุล่วง นี่คือประโยชน์ของการประกอบเว็บเซอร์วิส

ด้วยวิธีการประกอบเว็บเซอร์วิสให้ทำงานร่วมกันในปัจจุบันอย่างเช่น บีเพล (BPEL) [3] ผู้ใช้ต้องทำการประกอบเว็บเซอร์วิสด้วยตัวเอง โดยผู้ใช้ต้องทำการค้นหาเว็บเซอร์วิสเสียก่อน เช่น ทำการค้นหาจากยูดีดีไอ (UDDI) [4] จึงทำให้ต้องเสียเวลาส่วนหนึ่ง โดยที่เว็บเซอร์วิสที่ได้มานั้นอาจจะไม่ตรงกับความต้องการ หรือเว็บเซอร์วิสนั้นอาจจะไม่สามารถทำงานร่วมกันได้ การส่งงานให้ตัวแทนการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ (Automatic Web Service Composition Agent) เป็นทางเลือกหนึ่งเพื่อลดเวลาและความยุ่งยากในการประกอบเว็บเซอร์วิส โดยตัวแทนการประกอบเว็บเซอร์วิสอย่างอัตโนมัติสามารถแยกการทำงานออกเป็น 3 ส่วนใหญ่ๆได้ดังนี้ (1) การค้นหาเว็บเซอร์วิสอย่างอัตโนมัติ คือ การค้นหาเว็บเซอร์วิสที่เกี่ยวข้องสำหรับการประกอบเว็บเซอร์วิส (2) การประกอบเว็บเซอร์วิสอย่างอัตโนมัติ คือ การวางแผนงานการทำงานของเว็บเซอร์วิสต่างๆเป็นลำดับขั้นตอน เพื่อบรรลุเป้าหมายที่วางไว้ และ (3) การทำงานอย่างอัตโนมัติ คือขั้นตอนการเรียกใช้งานเว็บเซอร์วิสตามแผนงานที่ได้วางไว้ [5]

การประกอบเว็บเซอร์วิสอย่างอัตโนมัติได้มีการปรับปรุงให้สามารถทำงานในลักษณะเชิงความหมายมากยิ่งขึ้น วิธีหนึ่งที่ใช้อธิบายบริการให้มีลักษณะเชิงความหมายคือ การใช้ภาษาออนโทโลจี (Ontology Language) ซึ่งเป็นวิธีที่ใช้กันอย่างแพร่หลายในงานวิจัยเกี่ยวกับเว็บเซอร์วิส เนื่องจากภาษาออนโทโลจีสามารถอธิบายบริการได้หลายแง่มุม และยังสามารถทำการคิดเหตุผล (Reasoning) ได้ [6] คำอธิบายบริการซึ่งใช้ภาษาออนโทโลจีและเป็นที่รู้จักคือ อาวล์-เอส (OWL-S) [7] ซึ่งประกอบด้วย 3 โพรไฟล์ (Profile) คือ เซอร์วิสโพรไฟล์ (Service Profile) โพรเซสโมเดล

(Process Model) และเซอร์วิสกราวน์ดิง (Service Grounding) เซอร์วิสโพรไฟล์ทำหน้าที่อธิบายลักษณะและพฤติกรรมเชิงหน้าที่ (Functional Behaviour) เช่น อินพุท (Input) เอาท์พุท (Output) เงื่อนไขก่อนการทำงาน (Precondition) เอฟเฟ็กต์ (Effect) โพรเซสโมเดลอธิบายการดำเนินการ (Operation) ของบริการในรูปแบบของพฤติกรรมเชิงหน้าที่ โครงสร้างควบคุม (Control Structure) และโครงสร้างการไหลของข้อมูล (Data Flow Structure) เซอร์วิสกราวน์ดิงทำหน้าที่แมป (Map) โพรเซสโมเดลไปยังดับเบิลยูเอสดีแอล (WSDL)

งานวิจัยที่เกี่ยวข้องกับการประกอบเว็บเซอร์วิสอย่างอัตโนมัติได้มีการพัฒนาในหลายลักษณะโดยอาจแบ่งได้เป็นสองกลุ่ม กลุ่มแรกคือ กลุ่มที่ทราบรูปแบบของการประกอบเว็บเซอร์วิสแล้ว ว่ามีเว็บเซอร์วิสประเภทใดเกี่ยวข้องบ้าง ดังนั้นงานวิจัยจึงเป็นการเลือกเว็บเซอร์วิสของผู้ให้บริการแต่ละประเภทมาประกอบกัน เพื่อสร้างแผนการทำงาน (Plan) โดยอาศัยเกณฑ์บางอย่างซึ่งบ่งบอกว่า เว็บเซอร์วิสตัวใดที่ประกอบกันได้ดีที่สุดตามเกณฑ์นั้น เกณฑ์ที่ใช้โดยมากมักเกี่ยวกับคุณภาพการให้บริการ (Quality of Service) ของเว็บเซอร์วิสแต่ละตัว เช่น เวลาตอบสนอง (Response Time) ปริมาณงานที่ผลิต (Throughput) ราคา (Cost) เป็นต้น ตัวอย่างเช่น งานวิจัย [8] กำหนดฟังก์ชันของคุณภาพของการประกอบเว็บเซอร์วิส โดยพิจารณาจากคุณภาพการให้บริการของเว็บเซอร์วิสที่นำมาประกอบกันนั่นเอง หรือ งานวิจัย [9] ที่ใช้คุณภาพการให้บริการของเว็บเซอร์วิสมาพิจารณาเช่นกัน แต่มีการประยุกต์ใช้อัลกอริทึมทางพันธุกรรม มาช่วยในการประกอบเว็บเซอร์วิสเข้าด้วยกันด้วย งานวิจัย [10] ทำการประกอบเว็บเซอร์วิสโดยพิจารณาจากทั้งคุณภาพการให้บริการและการเข้ากันได้ตามความหมายของพฤติกรรมเชิงหน้าที่ของแต่ละเว็บเซอร์วิส โดยพิจารณาจากข้อกำหนดเชิงความหมายของแต่ละเว็บเซอร์วิส ซึ่งอธิบายด้วยออนโทโลจี

กลุ่มที่สอง เป็นกลุ่มที่ถือว่าไม่ทราบรูปแบบในการประกอบเว็บเซอร์วิสว่ามีเว็บเซอร์วิสประเภทใดเกี่ยวข้องบ้าง แต่จะทราบเพียงข้อมูลเข้าและผลลัพธ์ที่ต้องการ ดังนั้นงานวิจัยจึงเป็นการสร้างแผนงานในการประกอบเว็บเซอร์วิสต่างๆ ในหลากหลายประเภทเข้าด้วยกัน โดยแผนงานนั้นจะสามารถรับข้อมูลเข้าและให้ผลลัพธ์ตามที่ต้องการได้ งานวิจัยในกลุ่มนี้มักใช้ออนโทโลจีเข้ามาช่วยในการอธิบายความสามารถหรือพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิส เพื่อให้แผนงานที่สร้างขึ้นตรงกับความต้องการ ตัวอย่างเช่น งานวิจัย [5], [11] ได้นำความรู้ทางด้านปัญญาประดิษฐ์และตัววางแผน (AI Planner) มาช่วยในการสร้างแผนงานในการประกอบเว็บเซอร์วิส โดยใช้ออนโทโลจีในการอธิบายพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสแต่ละตัว รวมทั้งอธิบายอินพุทของเว็บเซอร์วิสเริ่มต้นและเอาท์พุทของเว็บเซอร์วิสปลายทางที่ต้องการด้วย

งานวิจัยดังกล่าวข้างต้นต่างมีข้อดีข้อเสีย การพิจารณาคุณภาพของบริการทำให้สามารถประกอบเว็บเซอร์วิสที่เมื่อทำงานร่วมกันแล้ว จะให้บริการที่มีคุณภาพดี แต่การเก็บข้อมูลคุณภาพของบริการมาพิจารณาจะทำได้ยาก และค่าข้อมูลจะเปลี่ยนแปลงได้ตลอดเวลา ทำให้แผนงานที่ได้

จะใช้ได้ในช่วงเวลาสั้น งานวิจัยในกลุ่มที่สอง ซึ่งทำการสร้างแผนงานของการประกอบเว็บเซอร์วิสโดยอัตโนมัติ จะทำให้สะดวกต่อผู้ใช้ ในแง่ที่ผู้ใช้ไม่จำเป็นต้องมีความรู้เกี่ยวกับการประกอบเซอร์วิสใดๆเข้าด้วยกันเลย แต่การกำหนดเพียงข้อมูลเข้าและผลลัพธ์ที่ต้องการเป็นการกำหนดความต้องการแบบหลวม ดังนั้นอาจจะให้เกิดการแกว่ง (Fluctuation) ของแผนงานที่สร้างขึ้น เนื่องจากอาจมีเว็บเซอร์วิสหลากหลายประเภท ประกอบกันในหลากหลายรูปแบบที่สามารถรับข้อมูลเข้าและให้ผลลัพธ์ตามที่ระบุ นอกจากนี้งานวิจัยที่ใช้ออนโทโลจีมาช่วยในการอธิบายความสามารถหรือพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิส เพื่อใช้พิจารณาความเข้ากันได้ของเว็บเซอร์วิสที่นำมาประกอบกัน ก็ยังเป็นเพียงการพิจารณาความหมายแบบเบื้องต้น แต่ยังไม่พิจารณาในรายละเอียดที่เกี่ยวกับข้อบังคับ (Constraint) ในการทำงานของเว็บเซอร์วิส ตัวอย่างเช่น ถ้าลูกค้าอยู่ในกรุงเทพฯ เว็บเซอร์วิสจะมีเอฟเฟ็กต์ในการส่งสินค้าที่ซื้อไปให้โดยไม่คิดค่าใช้จ่าย แต่ถ้าไม่เช่นนั้นลูกค้าจะต้องเสียค่านำส่ง ข้อบังคับเช่นนี้จะส่งผลต่อการเลือกเว็บเซอร์วิสที่มาประกอบหากลูกค้าต้องการเว็บเซอร์วิสที่ส่งสินค้าให้โดยไม่คิดค่าใช้จ่าย

งานวิจัยนี้จะทำการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ โดยใช้แนวทางการประกอบแบบกลุ่มที่หนึ่งคือ ทราบรูปแบบการประกอบเว็บเซอร์วิสอยู่แล้ว ว่ามีเว็บเซอร์วิสประเภทใดเกี่ยวข้องบ้าง โดยนักออกแบบเว็บเซอร์วิส ซึ่งมีความรู้เกี่ยวกับเซอร์วิสและการประกอบเซอร์วิสจะเป็นผู้กำหนดรูปแบบ การประกอบเว็บเซอร์วิสจะพิจารณาเฉพาะข้อมูลพฤติกรรมเชิงหน้าที่ที่ผู้ให้บริการเว็บเซอร์วิสแต่ละรายได้ลงทะเบียนไว้ ซึ่งอธิบายด้วยอาวล์–เอสโพรเซสโมเดล และมีข้อบังคับต่างๆ ในการทำงานระบุไว้โดยใช้ภาษากฎ (Rule Language) เช่นภาษาสเวิร์ล (SWRL) [12] ขั้นตอนจะเริ่มจากการรับข้อมูลรูปแบบการประกอบเว็บเซอร์วิสจากผู้ประกอบเว็บเซอร์วิสในลักษณะเป็นกระแสงานซึ่งอธิบายด้วยอาวล์-เอสโพรเซสโมเดล ซึ่งระบุเงื่อนไขกระบวนการภายในของเว็บเซอร์วิสตามที่ต้องการในแต่ละขั้นตอนในกระแสงาน รูปแบบการประกอบเว็บเซอร์วิสนี้จะใช้สำหรับค้นหาเว็บเซอร์วิสที่ตรงตามพฤติกรรมเชิงหน้าที่และเงื่อนไขกระบวนการทำงาน เพื่อสร้างแผนงานที่เป็นไปได้ทั้งหมด จากนั้นจะพิจารณาการเข้ากันได้ของเว็บเซอร์วิสที่มาประกอบกันในแง่ของอินพุทและเอาท์พุทในเชิงออนโทโลจีและเงื่อนไขการทำงาน และใช้ค่าความเข้ากันได้ในการถ่วงน้ำหนักในวิธีการหาแผนงาน เพื่อให้ได้แผนงานที่ดีที่สุดที่จะใช้จริงออกมาแล้วจึงสร้างเป็นเอกสารการประกอบเว็บเซอร์วิสที่สามารถทำงานได้เช่น บีเพล เป็นต้น

1.2 วัตถุประสงค์ของการวิจัย

เพื่อพัฒนาตัวกลางในการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยอาศัยข้อกำหนดกระบวนการในรูปของอาวล์-เอสโพรเซสโมเดล ที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่

1.3 ขอบเขตของการวิจัย

1.3.1 ตัวกลางจะมีความสามารถช่วยผู้ประกอบเว็บเซอร์วิสทำการประกอบเซอร์วิสอย่างอัตโนมัติ จากกระแสงานและข้อบังคับด้านพฤติกรรมเชิงหน้าที่ที่ผู้ประกอบเว็บเซอร์วิสส่งเข้ามายังตัวกลาง

1.3.2 งานวิจัยนี้ใช้ข้อกำหนดกระบวนการและกระแสงานตามอาวล์-เอสโพรเซสโมเดล และรับกระแสงานที่มีโครงสร้าง แบบลำดับ แบบแบ่ง และแบบแบ่ง-รวม เท่านั้น

1.3.3 เงื่อนไขการเข้าคู่จะพิจารณาจากเงื่อนไขทางรูปแบบกระบวนการ ซึ่งพิจารณาจากเงื่อนไขทางออนโทโลจีเทอม เงื่อนไขแบบช่วงตัวเลขซึ่งกำหนดโดยตัวดำเนินการเชิงเปรียบเทียบ (Relational Operator) และเงื่อนไขบนเซตของข้อมูลแจงนับ

1.3.4 ในขั้นตอนการจัดสร้างเอกสารบีเพล จะไม่ตรวจสอบความเข้ากันได้ของซิกเนเชอร์ของแต่ละเว็บเซอร์วิสที่มาประกอบกันโดยจะถือว่าเว็บเซอร์วิสที่นำมาประกอบกันมีซิกเนเชอร์ตรงกันอยู่แล้ว

1.3.5 ทำการทดสอบโดยการเปรียบเทียบผลการประกอบเว็บเซอร์วิสโดยตัวกลางกับผลที่ได้จากการประกอบโดยผู้ประกอบเว็บเซอร์วิส

1.4 ขั้นตอนการวิจัย

1.4.1 ศึกษาการทำงานของอาวล์-เอสโพรเซสโมเดล

1.4.2 ศึกษางานวิจัยเกี่ยวกับการประกอบเว็บเซอร์วิส

1.4.3 ศึกษาเครื่องมือเกี่ยวกับเว็บเซอร์วิส

1.4.4 พัฒนาตัวกลางสำหรับการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

1.4.5 ทดสอบการทำงานของตัวกลางและปรับปรุงตามความเหมาะสม

1.4.6 สรุปผลการวิจัย และจัดทำรายงานวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ได้ตัวกลางที่สามารถใช้ในการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยมีการพิจารณาข้อบังคับด้านพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่นำมาประกอบกันด้วย อันจะเป็นการช่วยลดเวลาในการประกอบเว็บเซอร์วิสได้

1.6 ผลงานตีพิมพ์

ส่วนหนึ่งของวิทยานิพนธ์นี้ได้ตีพิมพ์และนำเสนอในการประชุมวิชาการดังนี้

1. Proceedings of 4th International Joint Conference on Computer Science and Software Engineering, Khon Kaen, Thailand (2007): 194-201 ในบทความเรื่อง Using OWL-S Process Model with Constraints on Functional Behaviour for Automatic Web Services Composition โดยผู้แต่งคือ Termsak Wasino และ Twittie Senivongse

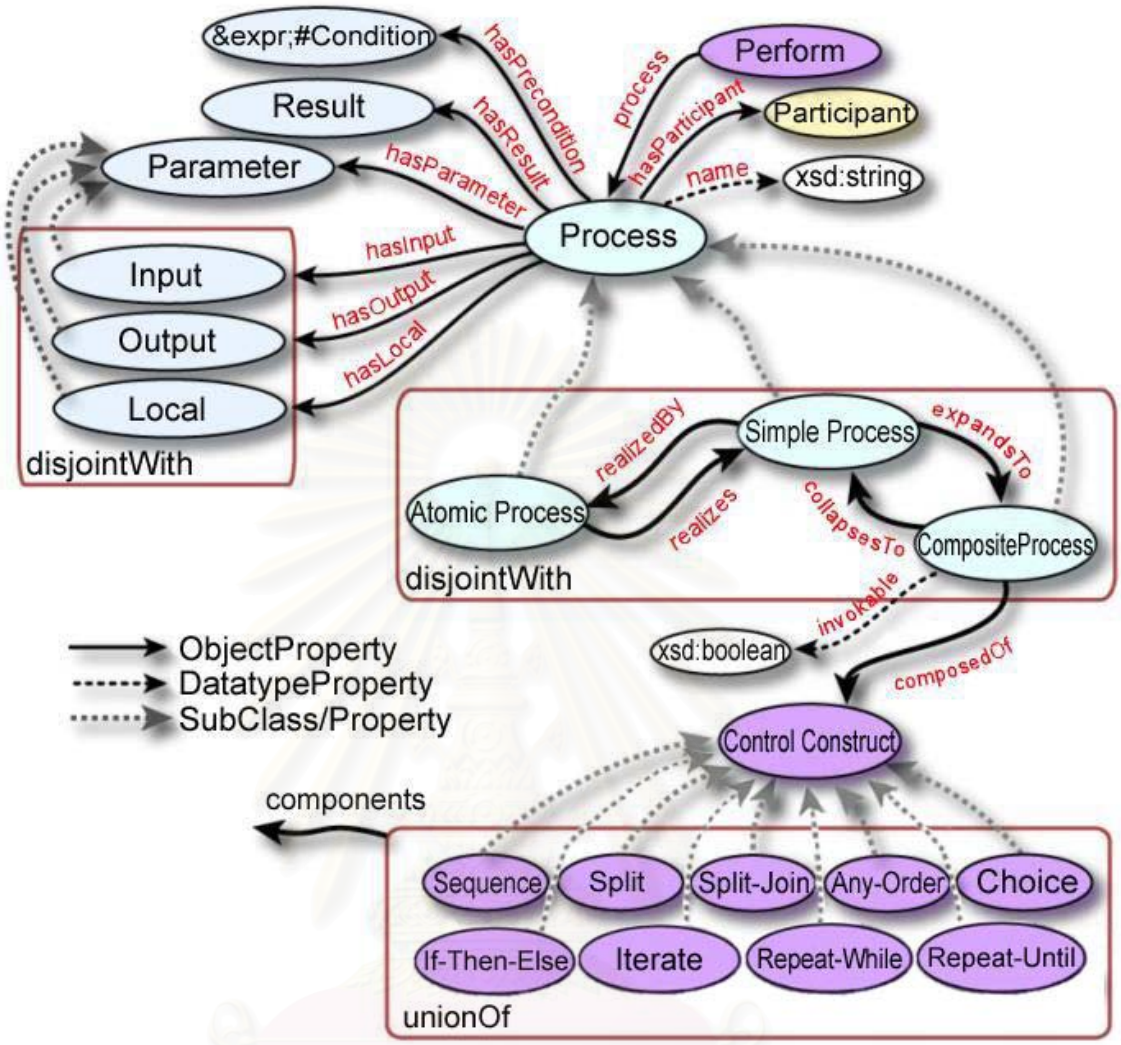
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดและทฤษฎี

2.1.1 อาวล์-เอสโพรเซสโมเดล (OWL-S Process Model)

อาวล์-เอสโพรเซสโมเดลเป็นโพรไฟล์หนึ่งของอาวล์-เอส [7] ทำหน้าที่อธิบายการดำเนินการของบริการในรูปแบบของพฤติกรรมเชิงหน้าที่ โครงสร้างควบคุม และโครงสร้างการไหลของข้อมูล โดยบริการที่ประกาศไว้จะอธิบายด้วยเซอร์วิสโมเดล (Service Model) ซึ่งมีกระบวนการ (Process) เป็นซับคลาส (Subclass) ของเซอร์วิส รูปที่ 2.1 แสดงอัปเปอร์ออนโทโลจี (Upper Ontology) ของอาวล์-เอสโพรเซสโมเดล ซึ่งประกอบด้วยคลาส (Class) และพร็อพเพอร์ตี (Property) ซึ่งร่วมกันอธิบายการทำงานของบริการ กระบวนการจะอธิบายพฤติกรรมเชิงหน้าที่โดยระบุอินพุท เอาท์พุท เงื่อนไขก่อนการทำงาน และเอฟเฟ็กต์ เงื่อนไขก่อนการทำงานคือ นิพจน์ (Expression) ซึ่งต้องเก็บค่าไว้เพื่อกระบวนการจะได้ถูกเรียกใช้งาน โลคัล (Local) เป็นพารามิเตอร์เสริมซึ่งผูกติดกับเงื่อนไขก่อนการทำงาน และมีประโยชน์ในการพิจารณาค่าเชิงตรรกะของเงื่อนไขก่อนการทำงาน ผลลัพธ์ (Result) เกี่ยวข้องกับเอาท์พุทและเอฟเฟ็กต์ ซึ่งอาจถูกกำหนดเงื่อนไขความสัมพันธ์โดยพร็อพเพอร์ตีของเงื่อนไขในการทำงาน (Incondition) ซึ่งระบุเงื่อนไขทางตรรกะของผลลัพธ์ที่จะเกิด ดังนั้นเอาท์พุทและเอฟเฟ็กต์ จะกลายเป็นเอาท์พุทเชิงเงื่อนไข (Conditional Output) และเอฟเฟ็กต์เชิงเงื่อนไข (Conditional Effect)

กระบวนการยังสามารถอธิบายด้วยการประกอบกันของกระบวนการย่อย กระบวนการย่อยอาจเป็นกระบวนการอะตอมมิก (Atomic Process) กระบวนการประกอบ (Composite Process) หรือกระบวนการอย่างง่าย (Simple Process) กระบวนการอะตอมมิกเป็นกระบวนการที่ไม่สามารถมีกระบวนการย่อยได้อีก สามารถเรียกใช้ได้โดยตรง และกระทำการ (Execute) ภายในขั้นตอนเดียว กระบวนการประกอบสามารถแยกได้เป็นกระบวนการที่ไม่ใช่กระบวนการประกอบหรือเป็นกระบวนการประกอบ การแยกกระบวนการสามารถระบุโดยใช้โครงสร้างควบคุม (Control Structure) เช่น ลำดับ (Sequence) ตัวแบ่ง (Split) ตัวแบ่ง-รวม (Split-Join) ลำดับอิสระ (Any-Order) ตัวเลือก (Choice) ทางเลือก (If-Then-Else) อิเทอเรท (Iterate) รีพีท-ไวล์ (Repeat-While) รีพีท-อันทิล (Repeat-Until) กระบวนการอย่างง่ายไม่สามารถเรียกใช้ได้โดยตรง และไม่ได้เชื่อมโยงกับกราวน์ดิง แต่กระทำการภายในขั้นตอนเดียว เช่นเดียวกับกระบวนการอะตอมมิก กระบวนการอย่างง่ายเป็นเพียงการกำหนดนามธรรม (Abstraction) ของการใช้งานกระบวนการอะตอมมิก หรือเตรียมการแสดงตัวอย่างอย่างง่ายของกระบวนการประกอบเพื่อการวางแผนหรือการคิดหาเหตุผล (Reasoning)

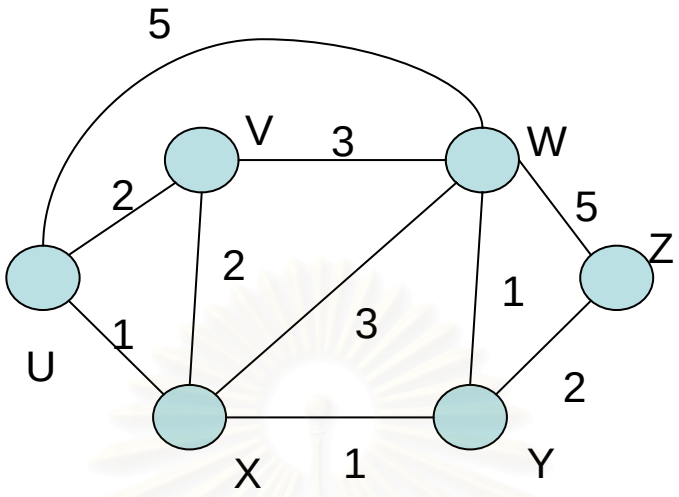


รูปที่ 2.1 อัปเปอร์ออนโทโลจีของอาวล์-เอสโพรเซสโมเดล [7]

2.1.2 อัลกอริทึมไดจ์สตรา (Dijkstra's Algorithm)

อัลกอริทึมไดจ์สตรา (Dijkstra's Algorithm) [13] ได้รับการนำเสนอ ในปี ค.ศ.1959 โดย เอด์สเจอร์ ไวบ์ ไดจ์สตรา (Edsger Wybe Dijkstra) โดยเป็นอัลกอริทึมที่ใช้แก้ปัญหาทางคณิตศาสตร์ในการหาเส้นทางที่สั้นที่สุด (Shortest Path) สำหรับการเดินทางจากโหนด (Node) หนึ่ง ไปยังอีกโหนดหนึ่ง

อัลกอริทึมไดจ์สตราจะใช้ค้นหาเส้นทางที่สั้นที่สุด โดยแต่ละเส้นทาง (Path) จะมีการกำหนดค่าน้ำหนัก (Weight) สำหรับการเดินทางจากโหนดเริ่มต้นไปยังโหนดปลายทาง ดังตัวอย่างดังรูปที่ 2.2



รูปที่ 2.2 ตัวอย่างกราฟของเส้นทาง

กำหนดให้ U เป็นโหนดเริ่มต้น และ Z เป็นโหนดปลายทาง

D(n) = ระยะทางที่สั้นที่สุดจากโหนดเริ่มต้นไปยังโหนด n ใดๆ เช่น D(V) = 2 หมายความว่า ระยะทางที่สั้นที่สุดจากจุดเริ่มต้นมายังโหนด V มีค่าเท่ากับ 2 เป็นต้น

P(n) = โหนดก่อนหน้าที่มีระยะทางสั้นที่สุดจากโหนดเริ่มต้นก่อนที่จะมาถึงโหนด n ใดๆ เช่น P(V) = U หมายความว่า โหนดก่อนหน้าที่มีระยะทางสั้นที่สุดก่อนที่จะมาถึงโหนด V คือ โหนด U เป็นต้น

N' = เซตของโหนดที่ทราบเส้นทางที่เส้นที่สุดจากโหนดเริ่มต้นแล้ว

K = โหนดใดๆที่ไม่อยู่ใน N'

ขั้นตอนการทำงาน :

1. กำหนดให้ N' = {U} ซึ่ง U คือโหนดเริ่มต้น สำหรับโหนด n ใดๆที่เป็นโหนดที่ติดต่อกับโหนด U แล้วให้สำรวจ ค่า D(n) และ P(n) ใส่ในตารางดังเช่นตารางที่ 2.1 ถ้าโหนดใดไม่ได้ติดต่อกับ U ให้ใส่เป็นค่าอนันต์ (∞)

2. ค้นหาโหนด K ที่มีค่า D(n) น้อยที่สุดและไม่อยู่ใน N' แล้วทำการเพิ่ม K ลงใน N' (กรณีมีโหนด K มากกว่าหนึ่งจะเลือกเอากรณีใดกรณีหนึ่งมาก่อน)

3. ทำการปรับปรุงค่า D(n) และ P(n) สำหรับทุกโหนด n ที่ไม่อยู่ใน N' โดยเปรียบเทียบค่าเดิมของ D(n) เก่า กับ D(n) ใหม่ที่บวกระยะทางผ่านโหนดที่เพิ่มลงใน N' ล่าสุด โดยจะเลือกค่าน้อยที่สุด บันทึกลงตารางพร้อมค่า P(n) ล่าสุด การปรับปรุงนี้เป็นการพิจารณาว่าค่า D(n)

จะสั้นลงหรือไม่หากวัดระยะทางผ่านโหนดที่เพิ่มลงใน N' ล่าสุด ถ้าการเพิ่มโหนดดังกล่าวไม่ได้ช่วยให้ค่า D(n) ลดลงจะคงค่า D(n) เดิมไว้

4. ทำซ้ำข้อ 2 จนกระทั่ง N' มีโหนดที่เกี่ยวข้องครบทุกโหนด

ตารางที่ 2.1 การคำนวณเส้นทางของรูปที่ 2.2 ด้วยวิธีไดจ์สตรา

ลำดับงาน	N'	D(V),P(V)	D(W),P(W)	D(X),P(X)	D(Y),P(Y)	D(Z),P(Z)
0	U	2,U	5,U	1,U	∞	∞
1	UX	2,U	4,X		2,X	∞
2	UXY	2,U	3,Y			4,Y
3	UXYV		3,Y			4,Y
4	UXYVW					4,Y
5	UXYVWZ					

จะพบว่า โหนด V, W และ X ติดต่อกับโหนด U ด้วยระยะทาง 2, 5 และ 1 ตามลำดับ จึงบันทึกค่าได้ดังลำดับงานที่ 0 ในตารางที่ 2.1 และ โหนด Y และ Z ไม่ได้ติดต่อกับ U จึงใส่เป็นค่าอนันต์ ระยะทางจาก U ไป X นั้นมีค่าน้อยที่สุดจึงเลือก X เข้าสู่เซต N' ระยะทางจาก U ไป V ค่าใหม่โดยผ่านโหนด X มีค่าเท่ากับ 3 ซึ่งค่าเดิมของระยะทางจาก U ไป V มีค่าเท่ากับ 2 จึงน้อยกว่าเพราะฉะนั้น ค่า D(V) และ P(V) จึงไม่เปลี่ยนแปลง ส่วนค่าอื่นมีค่าน้อยลง จึงบันทึกลงตารางในลำดับงานที่ 1 และทำการเลือกค่าที่น้อยที่สุดเข้าสู่เซต N' ตามลำดับ เมื่อใน N' มีโหนดที่เกี่ยวข้องครบทุกโหนดแล้วเราจะทราบระยะทางที่สั้นที่สุดไปยังโหนดต่างๆ เช่น ระยะทางที่สั้นที่สุดจากโหนดเริ่มต้นไปยังโหนด Z มีค่าเท่ากับ 4 จากตารางจะพบว่าโหนด Z มีค่า P(Z) เป็น Y เพราะฉะนั้นโหนดก่อนหน้ามาถึงโหนด Z คือ โหนด Y จากนั้นพิจารณาที่โหนด Y ก็จะพบว่าโหนดก่อนหน้าคือโหนด X ซึ่งโหนด X ก็มีโหนด U เป็นโหนดก่อนหน้า เพราะฉะนั้นการเดินทางจึงผ่านโหนด U, X และ Y ตามลำดับ

งานวิจัยนี้จะนำอัลกอริทึมไดจ์สตรามาใช้ในการหาแผนงานที่ดีที่สุด โดยจะแทนโหนดด้วยแต่ละเซอร์วิสของเว็บเซอร์วิสแต่ละประเภท และระยะทางจากโหนดหนึ่งไปยังอีกโหนดหนึ่งจะแทนด้วยระดับความเข้ากันได้ของแต่ละเซอร์วิสที่ประกอบกันซึ่งจะเป็นค่าที่ตัวกลางในการประกอบเว็บเซอร์วิสคำนวณให้ ถ้าสองเซอร์วิสที่ประกอบกันเข้ากันได้ยิ่งมากเท่าไหร่ค่าก็จะยิ่งน้อยเพื่อให้สอดคล้องกับอัลกอริทึมไดจ์สตรา

2.2 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.2.1 การประกอบเว็บเซอร์วิสอย่างอัตโนมัติ (Automated Web Service Composition) โดย Do'nal Murtagh

งานวิจัย [5] นำเสนอการประกอบเว็บเซอร์วิสโดยนำความรู้ทางด้านปัญญาประดิษฐ์และตัววางแผน มาช่วยในการสร้างแผนงานในการประกอบเว็บเซอร์วิส โดยจะนำเว็บเซอร์วิสที่เกี่ยวข้องซึ่งใช้ อาวล์-เอส ในการอธิบายพฤติกรรมเชิงหน้าที่และนำอินพุทของเว็บเซอร์วิสเริ่มต้นและเอาท์พุทของเว็บเซอร์วิสปลายทางที่ต้องการมาแปลงเป็นภาษาพีดีดีแอล (PDDL) ซึ่งเป็นภาษาที่ตัววางแผนที่เรียกว่า เจชอป (JSHOP) สามารถเข้าใจได้ จากนั้นจะส่งรายละเอียดข้างต้นไปยังตัววางแผน โดยตัววางแผนจะมีการติดต่อกับเครื่องมือคิดหาเหตุผลเพื่อพิจารณาเงื่อนไขก่อนการทำงาน และ เงื่อนไขหลังการทำงาน ของเว็บเซอร์วิสที่เกี่ยวข้อง เพื่อสร้างแผนงานที่เป็นไปได้ออกมา

งานวิจัยนี้ในขณะทำการประกอบเว็บเซอร์วิสจะไม่พิจารณาถึงเงื่อนไขการเข้าคู่กันของอินพุท เอาท์พุท และกรณีที่ เอฟเฟ็กต์ หรือ เอาท์พุท มีเงื่อนไขที่ต้องพิจารณา

2.2.2 การประกอบเว็บเซอร์วิสโดยพิจารณาคุณภาพ (Quality Driven Web Service Composition) โดย L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam

งานวิจัย [8] จะพิจารณาคุณภาพการให้บริการของเว็บเซอร์วิสอันได้แก่

1. ราคาในการดำเนินงาน (Execution Price) คือ จำนวนเงินที่จะต้องจ่ายเมื่อมีการร้องขอใช้บริการ

2. ความยาวนานในการดำเนินงาน (Execution Duration) คือ ระยะเวลาในการดำเนินงาน

3. ความน่าเชื่อถือ (Reliability) คือ ความน่าจะเป็นที่การทำงานของเว็บเซอร์วิสที่ถูกร้องขอจะทำการตอบสนองต่อผู้ร้องขอตามเวลาที่กำหนด

4. ความสามารถในการใช้งานได้ (Availability) คือ ความน่าจะเป็นที่เว็บเซอร์วิสนั้นๆสามารถติดต่อเพื่อขอใช้บริการได้

5. ความมีชื่อเสียงของบริการ (Reputation) คือ ความไว้วางใจในการให้บริการของเว็บเซอร์วิสนั้นๆ

เมื่อทำการวัดผลดังกล่าวแล้วจะนำมาคำนวณเป็นค่าคะแนนสำหรับเว็บเซอร์วิสนั้นๆเพื่อใช้ในการประกอบการตัดสินใจในการเลือกเว็บเซอร์วิสเพื่อการประกอบเว็บเซอร์วิส โดยจะเลือกแผนการดำเนินงานที่มีค่าคะแนนโดยรวมดีที่สุดมาใช้

งานวิจัยนี้จะไม่ได้พิจารณาเว็บเซอร์วิสในเชิงความหมายและไม่ได้พิจารณาในด้านพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิส แต่จะเน้นไปทางด้านคุณภาพการให้บริการของเว็บเซอร์วิสที่จะนำมาใช้ในการประกอบเว็บเซอร์วิส

2.2.3 การประกอบเว็บเซอร์วิสด้วยข้อบังคับในเมทีออร์-เอส (Constraint Driven Web Service Composition in METEOR-S) โดย Rohit Aggarwal, Kunal Verma

งานวิจัย [10] นำเสนอการสร้างรูปแบบการประกอบเว็บเซอร์วิสในลักษณะของกระแสงาน โดยใช้เอกสารบีเพล และกำหนดลักษณะของเว็บเซอร์วิสที่ต้องการ โดยใช้แผ่นแบบการบริการ (Service Templates) โดยให้ผู้ออกแบบกำหนดรายละเอียดของเว็บเซอร์วิสที่ต้องการซึ่งจะประกอบไปด้วย สถานที่ตั้งของเซอร์วิส (Geographic Location of the Service) ความหมายเกี่ยวกับกระบวนการทำงานของเว็บเซอร์วิส (Functional Semantics) อินพุท เอาท์พุท เงื่อนไขก่อนการทำงาน และ เงื่อนไขภายหลังการทำงาน (Post Condition) ในรูปออนโทโลจีเทอม และได้นำคุณภาพการให้บริการของเว็บเซอร์วิสเข้ามาช่วยในการพิจารณาประกอบเว็บเซอร์วิสด้วย

งานวิจัยนี้ในขณะทำการประกอบเว็บเซอร์วิสจะไม่พิจารณาถึงเงื่อนไขการเข้าคู่กันของอินพุท เอาท์พุท และกรณีที่ เอฟเฟ็กต์ หรือ เอาท์พุท มีเงื่อนไขที่ต้องพิจารณา

2.2.4 การประกอบเว็บเซอร์วิสอย่างอัตโนมัติของเว็บเซอร์วิสเชิงความหมาย (Automatic Composition of Semantic Web Service) โดย Ruoyan Zhang, I. Budak Arpinar, Boanerges Aleman-Meza

งานวิจัย [14] นำ ดีเอเอ็มเอล-เอส (DAML-S) มาใช้ในการอธิบายออนโทโลจีที่จะใช้ในการทำการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ โดยใช้เทคนิคการประกอบเว็บเซอร์วิสชื่อว่า ไอเอ็มเอ (IMA) โดยการประกอบเว็บเซอร์วิสจะใช้การเข้ากันได้ของอินพุทของเว็บเซอร์วิสหนึ่ง กับเอาท์พุทของอีกเว็บเซอร์วิสหนึ่ง ซึ่งจะมีระดับความเข้ากันได้ตามออนโทโลจีที่กำหนด มาสร้างแผนงานที่เป็นไปได้ทั้งหมดขึ้นมา จากนั้นนำค่าเวลาการดำเนินงานของแต่ละเว็บเซอร์วิสและความเข้ากันได้ในระดับออนโทโลจีของอินพุทและเอาท์พุทมาคิดเป็นค่าน้ำหนักที่ใช้ในการเลือกประกอบเว็บเซอร์วิสที่เหมาะสม โดยใช้วิธีค้นหาเส้นทางที่สั้นที่สุด (Shortest Path) เพื่อเลือกแผนงานที่ดีที่สุดในการดำเนินงาน

งานวิจัยนี้ใช้เพียงการเข้ากันได้ของอินพุทและเอาท์พุทในด้านออนโทโลจีเท่านั้น แต่ไม่ได้พิจารณาในด้านอื่นๆซึ่งมีอยู่ในเว็บเซอร์วิสเช่น เงื่อนไขเริ่มต้นการทำงาน เอฟเฟ็กต์ เป็นต้น

2.2.5 การเข้าคู่และการจัดลำดับเว็บเซอร์วิสเชิงความหมายโดยใช้เซอร์วิสโพรไฟล์รวม (Matchmaking and Ranking of Semantic Web Services Using Integrated Service Profile) โดย Natenapa Sriharee, Twittie Senivongse

งานวิจัย [15] นำเสนอโมเดลการอธิบายเว็บเซอร์วิสที่เรียกว่าเซอร์วิสโพรไฟล์รวม (Integrated Service Profile) ซึ่งอธิบายความสามารถของเว็บเซอร์วิสในหลายแง่มุม และนำไปใช้ในการค้นหาเว็บเซอร์วิสได้ ผลการค้นหาเว็บเซอร์วิสยังถูกนำมาจัดลำดับการตรงกับความต้องการอีกด้วย เซอร์วิสโพรไฟล์รวมประกอบด้วยโพรไฟล์ย่อย 2 โพรไฟล์ คือ

1. โพรไฟล์แอททริบิวท์อย่างง่าย (Simple Attribute Profile) เป็นโพรไฟล์ที่ใช้สำหรับประกาศเว็บเซอร์วิสในยูดีดีไอแบบปกติ

2. โพรไฟล์ความสามารถ (Capability Profile) ถูกจัดการโดยระบบลงทะเบียนเชิงความหมาย (Semantic Registry) ตัวโพรไฟล์ความสามารถเองไม่ได้แสดงถึงความสามารถของเว็บเซอร์วิสแต่อย่างใด แต่จะใช้โพรไฟล์ย่อยในการแสดงความสามารถ ซึ่งประกอบด้วยโพรไฟล์ย่อย 4 โพรไฟล์

2.1 โพรไฟล์แอททริบิวท์เชิงความหมาย (Semantic Attribute Profile) บรรจุแอททริบิวท์เชิงความหมาย ซึ่งมีค่าเชิงออนโทโลจีอยู่ เพื่อประโยชน์ในการคิดหาเหตุผล

2.2 โพรไฟล์โครงสร้าง (Structural Profile) เป็นโครงสร้างของความรู้เชิงความหมาย (Semantic Knowledge) เกี่ยวกับบริการ ซึ่งผู้ใช้บริการคาดหวังที่จะรู้ก่อนการตัดสินใจใช้บริการ เช่น ผลิตภัณฑ์ของบริการ รายละเอียดการขาย การส่งของ

2.3 โพรไฟล์พฤติกรรม (Behavioural Profile) เป็นความสามารถทางหน้าที่ของบริการด้านการปฏิบัติงาน อินพุท เอาท์พุท เอฟเฟ็กต์ ซึ่งบางครั้งจะต้องมีเงื่อนไขเข้ามาเกี่ยวข้อง เงื่อนไขนี้จะเกี่ยวข้องโดยตรงกับโพรไฟล์กฎ

2.4 โพรไฟล์กฎ (Rule Profile) แบ่งเป็น 2 ชนิดคือ เงื่อนไขเชิงพฤติกรรม (Behavioural Constraint) ซึ่งจะเกี่ยวข้องกับโพรไฟล์พฤติกรรม และเงื่อนไขในการปฏิบัติงาน (Operational Constraint) ซึ่งไม่เกี่ยวข้องกับโพรไฟล์พฤติกรรม

ทั้งโพรไฟล์ความสามารถ และโพรไฟล์ย่อยทั้ง 4 โพรไฟล์สืบทอดมาจากอัปเปอร์ออนโทโลจีของแต่ละโพรไฟล์

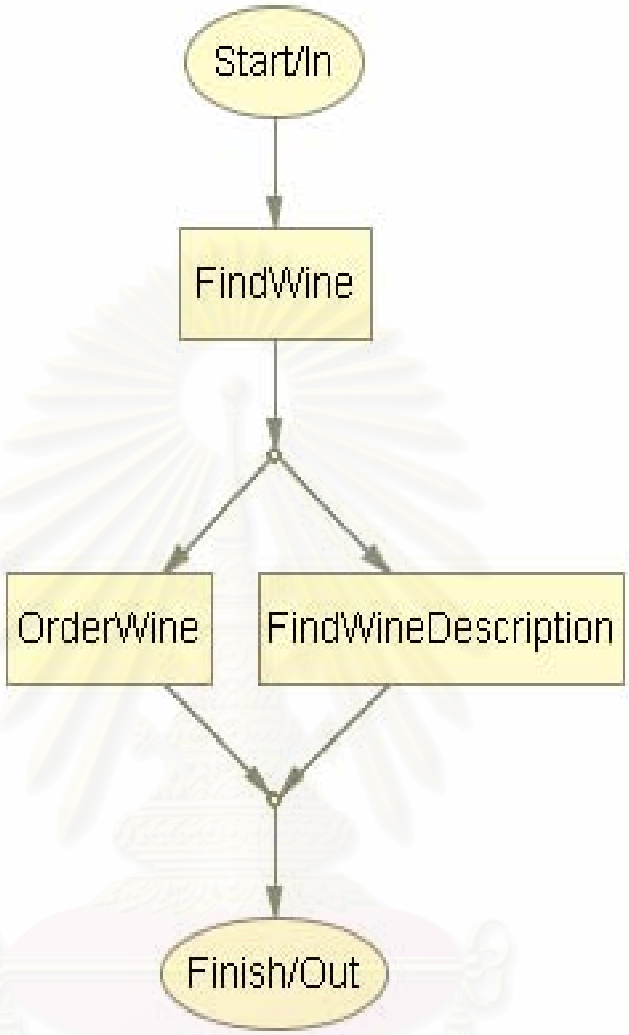
งานวิจัยนี้ได้นำเสนอหลักการในการค้นหาเว็บเซอร์วิสโดยพิจารณาจากทุกโพรไฟล์ข้างต้น โดยแต่ละโพรไฟล์จะมีเงื่อนไขในการค้นหาที่แตกต่างกัน ซึ่งเงื่อนไขเหล่านี้ได้นำมาใช้ในการจัดลำดับการตรงความต้องการการค้นหา งานวิจัยนี้ได้ยกตัวอย่างการค้นหาเว็บเซอร์วิส และยังได้นำเสนอเฟรมเวิร์กสำหรับการประกาศและการค้นหาเว็บเซอร์วิส ผู้วิจัยจะนำแนวคิดการเข้าคู่ของออนโทโลจีเทอมและเงื่อนไขเกี่ยวกับพฤติกรรมของเว็บเซอร์วิสจากงานวิจัยนี้มาใช้

บทที่ 3
การประกอบเว็บเซอร์วิสอย่างอัตโนมัติด้วยข้อกำหนดอาวล์-เอสโพรเซสโมเดลที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่

งานวิจัยนี้เสนอแนวทางการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยการพิจารณาพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่แสดงด้วยอาวล์-เอสโพรเซสโมเดลโดยคำนึงถึงเงื่อนไขทางพฤติกรรม การอธิบายแนวคิดของการวิจัยนี้ จะใช้กรณีตัวอย่าง ได้แก่ “ผู้ใช้บริการซึ่งเป็นนักออกแบบเว็บเซอร์วิสต้องการประกอบเว็บเซอร์วิสที่มีบริการการสั่งซื้อไวน์ที่เหมาะกับรายการอาหารที่กำหนด โดยมีข้อบังคับว่าบริการนี้จะรับบัตรเครดิตของบริษัท เอเม็กซ์ (Amex) และต้องมีบริการส่งถึงบ้านภายใน 3 วัน โดยที่อยู่ของลูกค้าอยู่ในกรุงเทพฯ พร้อมทั้งแสดงรายละเอียดของไวน์ที่สั่งซื้อด้วย”

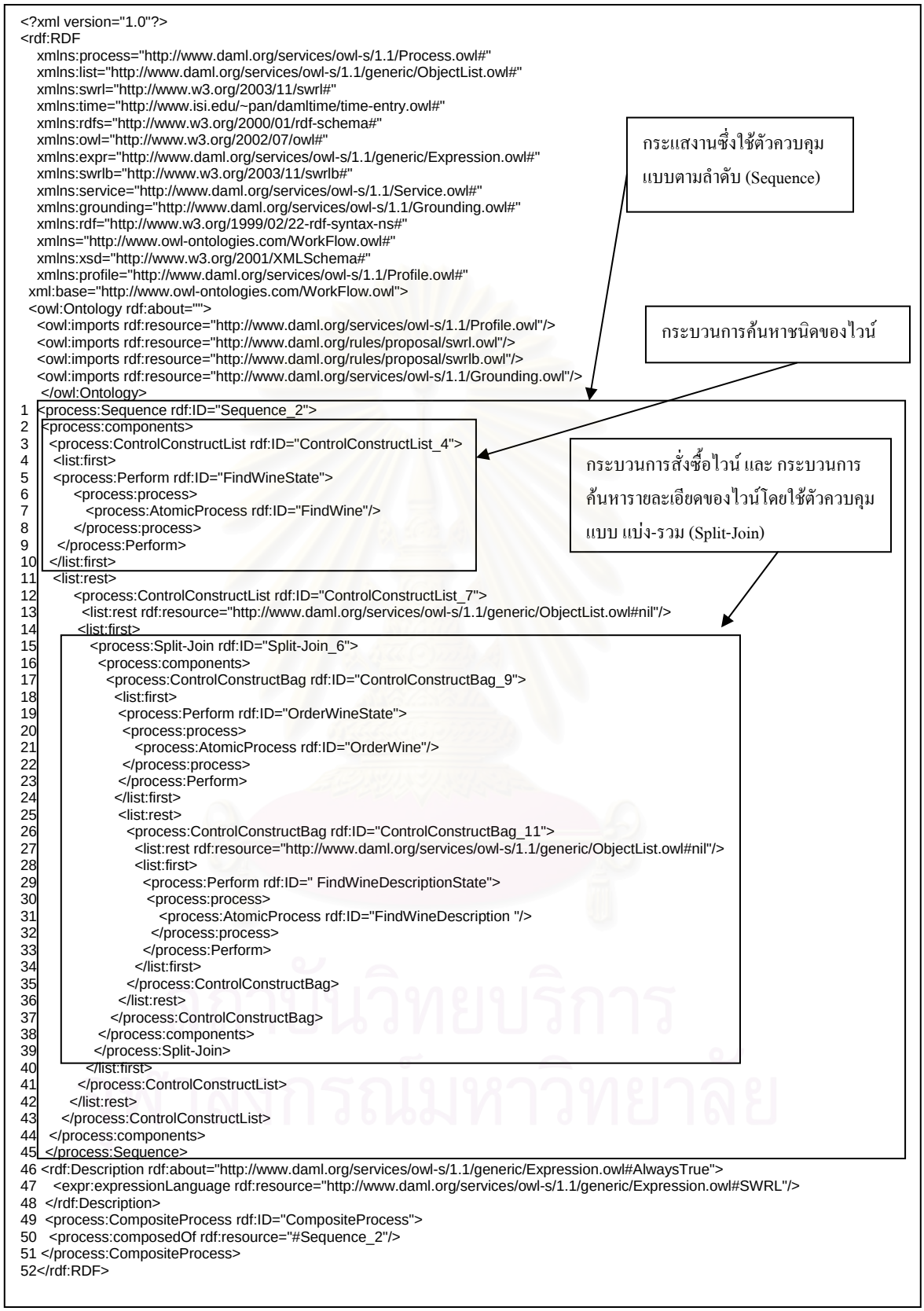
3.1 กระแสงานและข้อบังคับในกระบวนการ

ผู้ใช้บริการสามารถสร้างกระแสงานที่กำหนดรูปแบบการประกอบเว็บเซอร์วิสโดยใช้เครื่องมือสร้างออนโทโลจี เช่น โพรทีเจ (Protégé) [16] รูปที่ 3.1 แสดงกระแสงานของบริการของเว็บเซอร์วิสประกอบที่ให้บริการค้นหาไวน์พร้อมทั้งสั่งซื้อและแสดงรายละเอียดของไวน์



รูปที่ 3.1 กระแสงานของบริการของเว็บเซอร์วิสประกอบที่ให้บริการค้นหาไวน์พร้อมทั้งสั่งซื้อและแสดงรายละเอียดของไวน์

กระแสงานนี้มีการทำงานคือ เมื่อรับรายการอาหารเข้ามาเป็นอินพุทแล้วจะทำการค้นหาชนิดไวน์จากเซอร์วิสค้นหาไวน์ จากนั้นจะทำกระบวนการสั่งซื้อไวน์ และกระบวนการค้นหารายละเอียดของไวน์ โดยเซอร์วิสสั่งซื้อไวน์และเซอร์วิสค้นหารายละเอียดของไวน์ และทำทั้งสองกระบวนการจนเสร็จจึงจบกระแสงาน อาวล์-เอสโพรเซสโมเดลซึ่งอธิบายกระแสงานนี้แสดงได้ดังรูปที่ 3.2

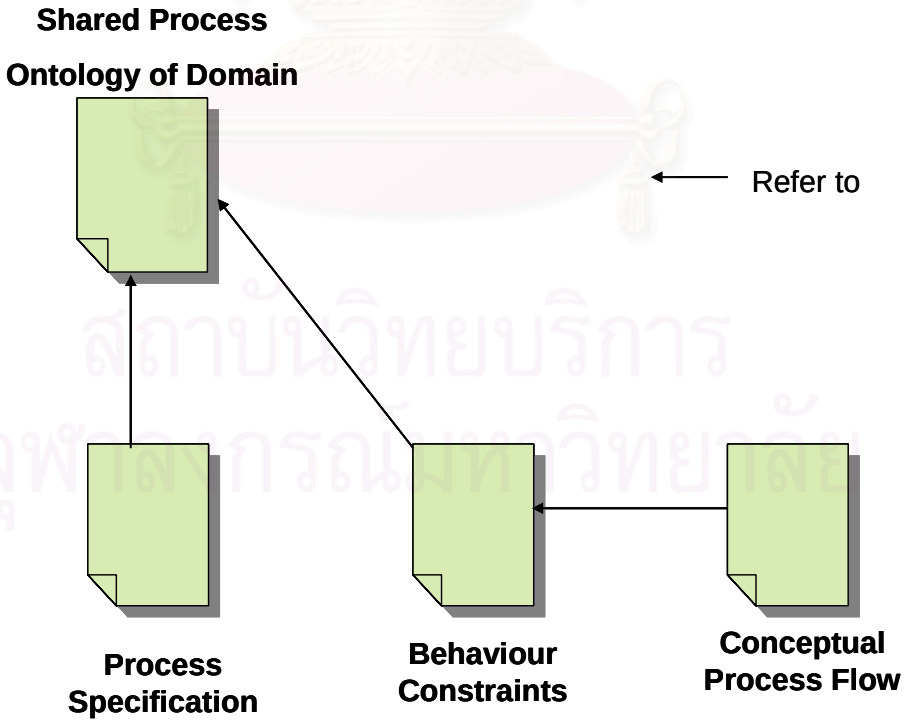



```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
   xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
   xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns="http://www.owl-ontologies.com/WorkFlow.owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
 xml:base="http://www.owl-ontologies.com/WorkFlow.owl">
 <owl:Ontology rdf:about="">
   <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Profile.owl"/>
   <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrl.owl"/>
   <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrlb.owl"/>
   <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Grounding.owl"/>
  </owl:Ontology>
1  <process:Sequence rdf:ID="Sequence_2">
2   <process:components>
3    <process:ControlConstructList rdf:ID="ControlConstructList_4">
4     <list:first>
5      <process:Perform rdf:ID="FindWineState">
6       <process:process>
7        <process:AtomicProcess rdf:ID="FindWine"/>
8       </process:process>
9      </process:Perform>
10    </list:first>
11    <list:rest>
12     <process:ControlConstructList rdf:ID="ControlConstructList_7">
13      <list:rest rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#nil"/>
14      <list:first>
15       <process:Split-Join rdf:ID="Split-Join_6">
16        <process:components>
17         <process:ControlConstructBag rdf:ID="ControlConstructBag_9">
18          <list:first>
19           <process:Perform rdf:ID="OrderWineState">
20            <process:process>
21             <process:AtomicProcess rdf:ID="OrderWine"/>
22            </process:process>
23           </process:Perform>
24          </list:first>
25          <list:rest>
26           <process:ControlConstructBag rdf:ID="ControlConstructBag_11">
27            <list:rest rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#nil"/>
28            <list:first>
29             <process:Perform rdf:ID=" FindWineDescriptionState">
30              <process:process>
31               <process:AtomicProcess rdf:ID="FindWineDescription "/>
32              </process:process>
33             </process:Perform>
34            </list:first>
35           </process:ControlConstructBag>
36          </list:rest>
37         </process:ControlConstructBag>
38        </process:components>
39       </process:Split-Join>
40      </list:first>
41     </process:ControlConstructList>
42    </list:rest>
43   </process:ControlConstructList>
44  </process:components>
45 </process:Sequence>
46 <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
47   <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
48 </rdf:Description>
49 <process:CompositeProcess rdf:ID="CompositeProcess">
50  <process:composedOf rdf:resource="#Sequence_2"/>
51 </process:CompositeProcess>
52</rdf:RDF>
```

รูปที่ 3.2 อาวล์-เอสโพรเซสโมเดลที่ใช้อธิบายกระแสงาน

ในกระแสงานจะใช้ตัวควบคุม (Control Construct) เป็นแบบตามลำดับ (Sequence) (บรรทัดที่ 1) โดยเริ่มต้นด้วย กระบวนการค้นหาชนิดของไวน์ (บรรทัดที่ 5-9) ตามด้วยตัวควบคุมแบบ แบ่ง-รวม (Split-Join) สำหรับสองกระบวนการ คือ กระบวนการสั่งซื้อไวน์ และ กระบวนการค้นหารายละเอียดของไวน์ (บรรทัดที่ 15-39) เมื่อทั้งสองกระบวนการทำเสร็จเป็นอันจบกระแสงาน

รูปที่ 3.3 แสดงภาพรวมการเชื่อมโยงระหว่างกระแสงานและเว็บเซอร์วิสที่มีคุณสมบัติตามที่ต้องการ เราจะทำการค้นหาโดยพิจารณาข้อกำหนดกระบวนการ (Process Specification) ของเว็บเซอร์วิสที่มีการลงทะเบียนไว้ซึ่งข้อกำหนดกระบวนการนี้ ผู้ให้บริการเว็บเซอร์วิสจะเป็นผู้กำหนดโดยอ้างอิงกับออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน (Shared Process Ontology of Domain) ซึ่งจะสร้างโดยผู้เชี่ยวชาญของโดเมนนั้น ตัวอย่างเช่น บางส่วนของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนร้านขายไวน์ แสดงในรูปที่ 3.4 และ บางส่วนของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนค้นหาไวน์ แสดงในรูปที่ 3.5 ส่วนผู้ประกอบเว็บเซอร์วิสเองนอกเหนือจากการกำหนดกระแสงานแล้วยังต้องใช้ออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนเพื่ออ้างอิงในการระบุข้อบังคับลักษณะพฤติกรรมเชิงหน้าที่ (Behaviour Constraints) ของเว็บเซอร์วิสที่ต้องการ เช่น ลักษณะอินพุท ลักษณะเอาท์พุท เงื่อนไขก่อนการทำงาน เอฟเฟ็กต์ ที่ต้องการ เป็นต้น กระแสงานจะอาศัยเอกสารข้อกำหนดพฤติกรรมเชิงหน้าที่ในการบ่งบอกว่าแต่ละเซอร์วิสที่ระบุในกระแสงานมีลักษณะอย่างไร



รูปที่ 3.3 ภาพรวมการเชื่อมโยงระหว่างกระแสงานและลักษณะของเว็บเซอร์วิสที่ต้องการ

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.owl-ontologies.com/WineShopDomain.owl#"
    xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.owl-ontologies.com/WineShopDomain.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrlb.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Grounding.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrl.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Profile.owl"/>
  </owl:Ontology>
  <owl:Class rdf:ID="DeliveryDayCondition">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#Condition"/>
  </owl:Class>
  <owl:Class rdf:ID="OrderConfirmed">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Result"/>
  </owl:Class>
  <owl:Class rdf:ID="CustomerLocation">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#ResultVar"/>
  </owl:Class>
  <owl:Class rdf:ID="LocationCondition">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#Condition"/>
  </owl:Class>
<owl:Class rdf:ID="AcceptedCreditCard">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#Condition"/>
  </owl:Class>
  <owl:Class rdf:ID="CustomerCreditcardType">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Local"/>
  </owl:Class>
  <owl:Class rdf:ID="WineName">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Input"/>
  </owl:Class>
  <owl:Class rdf:ID="CustomerInfo">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Input"/>
  </owl:Class>
  <owl:Class rdf:ID="DeliveryDay">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#ResultVar"/>
  </owl:Class>
  <owl:Class rdf:ID="OrderedWine">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Output"/>
  </owl:Class>
<owl:Class rdf:ID="SellWineProcess">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#AtomicProcess"/>
  </owl:Class>
<rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
    <expr:expressionBody rdf:parseType="Literal"><swrl:AtomList xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:swrl="http://www.w3.org/2003/11/swrl#" rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"></swrl:AtomList>
  </expr:expressionBody>
    <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
  </rdf:Description>
</rdf:RDF>
```

รูปที่ 3.4 บางส่วนของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนร้านขายไวน์

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.owl-ontologies.com/WineAgentDomain.owl#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.owl-ontologies.com/WineAgentDomain.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrl.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Profile.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/services/owl-s/1.1/Grounding.owl"/>
    <owl:imports rdf:resource="http://www.daml.org/rules/proposal/swrlb.owl"/>
  </owl:Ontology>
  <owl:Class rdf:ID="FoodName">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Input"/>
  </owl:Class>
  <owl:Class rdf:ID="WineName">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Output"/>
  </owl:Class>
  <owl:Class rdf:ID="WinePrice">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Output"/>
  </owl:Class>
  <owl:Class rdf:ID="WineDescription">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#Output"/>
  </owl:Class>
 <owl:Class rdf:ID="WineAgentProcess">
    <rdfs:subClassOf rdf:resource="http://www.daml.org/services/owl-s/1.1/Process.owl#AtomicProcess"/>
  </owl:Class>
  <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
    <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
  </rdf:Description>
</rdf:RDF>
```

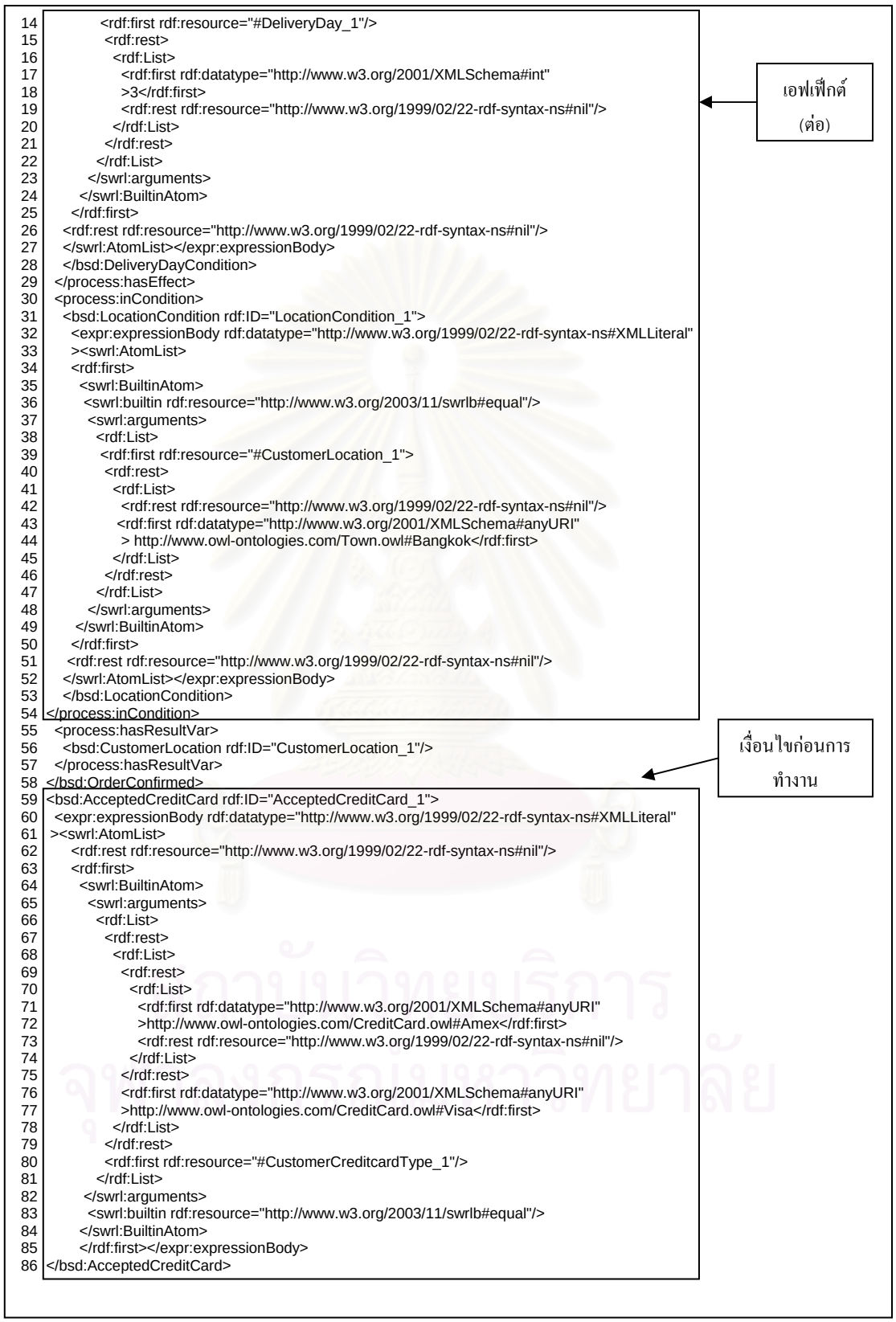
รูปที่ 3.5 บางส่วนของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนค้นหาไวน์

ผู้ให้บริการเว็บเซอร์วิสจะกำหนดพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสของตนซึ่งเกี่ยวกับร้านขายไวน์โดยการนำเข้าออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนร้านขายไวน์นี้เข้ามายังส่วนแสดงพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิส ดังตัวอย่าง แสดงในรูปที่ 3.6

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rss="http://purl.org/rss/1.0/"
   xmlns:cus="http://www.owl-ontologies.com/Customerinfo.owl#"
   xmlns:fd="http://www.w3.org/TR/2003/PR-owl-guide-20031209/food#"
   xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
   xmlns:bsd="http://www.owl-ontologies.com/WineShopDomain.owl.owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:p1="http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine#"
   xmlns:vin="http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#"
   xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
   xmlns="http://www.owl-ontologies.com/WineShop1.owl#"
   xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
   xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:pi="http://a.com/ontology#"
   xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
   xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
   xmlns:daml=http://www.daml.org/2001/03/daml+oil#
   xmlns:to=http://www.owl-ontologies.com/Town.owl#
   xmlns:cd=http://www.owl-ontologies.com/CreditCard.owl#
   xml:base="http://www.owl-ontologies.com/WineShop1.owl">
 <owl:Ontology rdf:about="">
   <owl:imports rdf:resource="http://localhost:8080/ontology/customerinfo.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/Food.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/pip.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/WineShopDomain.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/CreditCard.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/Town.owl"/>
 </owl:Ontology>

1<bsd:OrderConfirmed rdf:ID="OrderConfirmed_1">
2 <process:hasResultVar>
3    <bsd:DeliveryDay rdf:ID="DeliveryDay_1"/>
4   </process:hasResultVar>
5   <process:hasEffect>
6    <bsd:DeliveryDayCondition rdf:ID="DeliveryDayCondition_1">
7     <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
8     ><swrl:AtomList>
9    <rdf:first>
10       <swrl:BuiltinAtom>
11        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual/>
12         <swrl:arguments>
13          <rdf:List>
```

เอฟเฟ็กต์

รูปที่ 3.6 อาวล์-เอสโพรเซสโมเดลของส่วนแสดงพฤติกรรมเชิงหน้าที่ของร้านขายไวน์




```
14          <rdf:first rdf:resource="#DeliveryDay_1"/>
15           <rdf:rest>
16            <rdf:List>
17             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
18              >3</rdf:first>
19             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
20            </rdf:List>
21           </rdf:rest>
22          </rdf:List>
23         </swrl:arguments>
24       </swrl:BuiltinAtom>
25     </rdf:first>
26    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
27    </swrl:AtomList></expr:expressionBody>
28    </bsd:DeliveryDayCondition>
29  </process:hasEffect>
30  <process:inCondition>
31    <bsd:LocationCondition rdf:ID="LocationCondition_1">
32     <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
33      ><swrl:AtomList>
34     <rdf:first>
35       <swrl:BuiltinAtom>
36        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#equal"/>
37         <swrl:arguments>
38          <rdf:List>
39           <rdf:first rdf:resource="#CustomerLocation_1">
40            <rdf:rest>
41             <rdf:List>
42              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
43              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
44               > http://www.owl-ontologies.com/Town.owl#Bangkok</rdf:first>
45             </rdf:List>
46            </rdf:rest>
47           </rdf:List>
48         </swrl:arguments>
49       </swrl:BuiltinAtom>
50     </rdf:first>
51    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
52    </swrl:AtomList></expr:expressionBody>
53    </bsd:LocationCondition>
54 </process:inCondition>
55   <process:hasResultVar>
56    <bsd:CustomerLocation rdf:ID="CustomerLocation_1"/>
57   </process:hasResultVar>
58 </bsd:OrderConfirmed>
59 <bsd:AcceptedCreditCard rdf:ID="AcceptedCreditCard_1">
60  <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
61 ><swrl:AtomList>
62     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
63     <rdf:first>
64      <swrl:BuiltinAtom>
65       <swrl:arguments>
66        <rdf:List>
67         <rdf:rest>
68          <rdf:List>
69           <rdf:rest>
70            <rdf:List>
71             <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
72              >http://www.owl-ontologies.com/CreditCard.owl#Amex</rdf:first>
73             <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
74            </rdf:List>
75           </rdf:rest>
76           <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
77            >http://www.owl-ontologies.com/CreditCard.owl#Visa</rdf:first>
78          </rdf:List>
79         </rdf:rest>
80         <rdf:first rdf:resource="#CustomerCreditcardType_1"/>
81        </rdf:List>
82       </swrl:arguments>
83       <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#equal"/>
84      </swrl:BuiltinAtom>
85      </rdf:first></expr:expressionBody>
86 </bsd:AcceptedCreditCard>
```

รูปที่ 3.6 อาวล์-เอสโพรเซสโมเดลของส่วนแสดงพฤติกรรมเชิงหน้าที่ของร้านขายไวน์ (ต่อ)

```
87<bsd:CustomerCreditcardType rdf:ID="CustomerCreditcardType_1"/>
88 <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
89   <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
90  </rdf:Description>

91 <bsd:OrderedWine rdf:ID="OrderedWine_1">
92   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
93   >http://a.com/ontology#PurchaseOrderConfirmation</process:parameterType>
94  </bsd:OrderedWine>

95 <bsd:CustomerInfo rdf:ID="CustomerInfo_1">
96   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
97    > http://localhost/ontology/domain/Customerinfo.owl#Customerinfo</process:parameterType>
98 </bsd:CustomerInfo>

99 <bsd:WineName rdf:ID="WineName_1">
100   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
101   >http://localhost/ontology/domain/food.owl#Wine</process:parameterType>
102</bsd:WineName>
103       <to:Town rdf:ID = “Town”>
104       <cd:CreditCard rdf:ID="CreditCard">
105 <bsd:SellWineProcess rdf:ID="AlcoholShopAtomicProcess">
106     <process:hasInput rdf:resource="#CustomerInfo_1"/>
107     <process:hasInput rdf:resource="#WineName_1"/>
108       <process:hasPrecondition rdf:resource="#AcceptedCreditCard_1"/>
109       <process:hasLocal rdf:resource="#CustomerCreditcardType_1"/>
110       <process:hasResult rdf:resource="#OrderConfirmed_1"/>
111     <process:hasOutput rdf:resource="#OrderedWine_1"/>
112 </bsd:SellWineProcess >
</rdf:RDF>
```

อินพุท

เอาท์พุท

รูปที่ 3.6 อาวล์-เอสโพรเซสโมเดลของส่วนแสดงพฤติกรรมเชิงหน้าที่ของร้านขายไวน์ (ต่อ)

จากรูปที่ 3.6 เว็บเซอร์วิสนี้ต้องการข้อมูลของลูกค้า (Customer Information) และ ชนิดของไวน์ (Wine Type) เพื่อเป็นอินพุท (บรรทัดที่ 106, 107) และได้ไวน์ที่สั่งแล้วเป็นเอาท์พุท (บรรทัดที่ 111) บริการมีเงื่อนไขก่อนการทำงาน คือจะรับบัตรเครดิตประเภท เอเม็กซ์ (Amex) และวีซ่า (Visa) (บรรทัดที่ 59-86) เอฟเฟ็กต์ของบริการนี้เป็นเอฟเฟ็กต์ที่มีเงื่อนไข ซึ่งขึ้นอยู่กับที่อยู่ของลูกค้าคือ ถ้าลูกค้าอยู่ในกรุงเทพฯจะส่งของให้ภายใน 3 วัน (บรรทัดที่ 5-54) โดยเงื่อนไขทั้งหมดอธิบายด้วยภาษากฎสเวิร์ล และมีการนำเข้าออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนร้านขายไวน์ (บรรทัดที่ 1, 3, 6, 31, 56, 59, 87, 91, 95, 99, 105) สำหรับร้านขายไวน์อื่นๆอาจมีรายละเอียดส่วนแสดงพฤติกรรมเชิงหน้าที่ที่คล้ายคลึงกับร้านขายไวน์ร้านนี้แต่อาจจะแตกต่างในแง่ของเอฟเฟ็กต์ หรือ ชนิดพารามิเตอร์ (ParameterType) ของอินพุทหรือเอาท์พุทหรือเงื่อนไขก่อนการทำงาน เป็นต้น และเว็บเซอร์วิสที่ให้บริการประเภทอื่นๆที่เกี่ยวข้องก็จะมีส่วนแสดงพฤติกรรมเชิงหน้าที่แตกต่างกันไปขึ้นกับว่าอยู่ในโดเมนใดและมีออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนเป็นอย่างไร นอกจากนี้ยังมีส่วนนำเข้าออนโทโลจีที่เกี่ยวข้องเช่น ออนโทโลจีเกี่ยวกับเมือง (บรรทัดที่ 103) และ ออนโทโลจีเกี่ยวกับบัตรเครดิต (บรรทัดที่ 104) ดังแสดงในรูปที่ 3.7

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.owl-ontologies.com/CreditCard.owl#"
  xml:base="http://www.owl-ontologies.com/CreditCard.owl">

 <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Visa">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="CreditCard"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Amex">
    <rdfs:subClassOf rdf:resource="#CreditCard"/>
  </owl:Class>
  <owl:Class rdf:ID="Master">
    <rdfs:subClassOf rdf:resource="#CreditCard"/>
  </owl:Class>
   ......
</rdf:RDF>
```

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.owl-ontologies.com/Town.owl#"
  xml:base="http://www.owl-ontologies.com/Town.owl">

 <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Chantaburi">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Town"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Bangkok">
    <rdfs:subClassOf rdf:resource="#Town"/>
  </owl:Class>
  <owl:Class rdf:ID="Chiangmai">
    <rdfs:subClassOf rdf:resource="#Town"/>
  </owl:Class>
  <owl:Class rdf:ID="Chonburi">
    <rdfs:subClassOf rdf:resource="#Town"/>
  </owl:Class>
   ......
</rdf:RDF>
```

รูปที่ 3.7 บางส่วนของออนโทโลจีเกี่ยวกับบัตรเครดิตและออนโทโลจีเกี่ยวกับเมือง

ในการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ ผู้ประกอบเว็บเซอร์วิสจะระบุข้อบังคับ (Constraint) สำหรับแต่ละกระบวนการทำงานในกระแสงานพร้อมทั้งอินพุทเพื่อให้ระบบทำการค้นหาเว็บเซอร์วิสที่เกี่ยวข้องที่ตรงตามข้อบังคับ เพื่อใช้ในการประกอบกันของเว็บเซอร์วิส ตัวอย่างอินพุทและข้อบังคับของกระบวนการทำงานแสดงไว้ในรูปที่ 3.8

```
1 input :   http://localhost/ontology/domain/food.owl#MealCourse
2           http://localhost/ontology/domain/customerinfo.owl#Customerinfo

3 #FindWine{
4  Q =  {
5        HasInput(Process,FoodName)I
6        HasOutput(Process,WineName)O
7        }
8                  }

9 #OrderWine{
10 Q = {
11          HasInput(Process,WineName)I
12          HasInput(Process,CustomerInfo)I
13          HasOutput(Process,OrderedWine)O
14          HasCustomerCreditcardType(CreditCardType,Amex)P
15          HasCustomerLocation(CustomerLocation,Bangkok)R
16          HasDeliveryDay(DeliveryDay,LessThanOrEqual,3)R
17      }
18
19                  }
20
21 #FindWineDescription{
22 Q = {
          HasInput(Process,WineName)I
23        HasOutput(Process,WineDescription)O
24       }
25                          }
```

รูปที่ 3.8 อินพุทและข้อบังคับแต่ละส่วนกระบวนการในกระแสงานซึ่งต้องการค้นหา

จากรูปที่ 3.8 ให้ชื่อเซอร์วิสที่สอดคล้องกับกระแสงานแต่ละเซอร์วิสที่ต้องการค้นหาแสดงอยู่หลังสัญลักษณ์ # เช่น บรรทัดที่ 3 เป็นการบอกว่าเริ่มการกำหนดข้อบังคับของเซอร์วิสค้นหาไวน์และข้อความลักษณะพฤติกรรมของเว็บเซอร์วิสนั้นซึ่งต้องการค้นหา ($\mathbb{Q}$) เป็นเซตของนิพจน์โดยอยู่ในรูปแบบ property (subject, object) ซึ่งสอดคล้องกับรูปแบบของอาร์ดีเอฟ (RDF) <subject, property, object> สำหรับเงื่อนไขแบบช่วงตัวเลขของแต่ละส่วนกระบวนการจะอยู่ในรูปแบบ property(argument, relationaloperator, literalvalue1 [, literalvalue2]) แต่ละนิพจน์จะมีสัญลักษณ์กำกับว่าเป็นเงื่อนไขเกี่ยวกับอะไร เช่น P, I, O, R หมายถึง เงื่อนไขก่อนการทำงาน อินพุท เอาท์พุท เงื่อนไขของผลลัพธ์ ตามลำดับ

ในการพิจารณาว่าเว็บเซอร์วิสหนึ่งๆ ตรงกับความต้องการหรือไม่ ข้อกำหนดกระบวนการของเว็บเซอร์วิสจะถูกแปลงไปเป็นนิพจน์ในทำนองเดียวกับข้อความค้นหา การแปลงข้อกำหนดกระบวนการไปเป็นนิพจน์ทำได้โดยการใช้ตัวแจง (Parser) ซึ่งแบ่งได้เป็นสามส่วนที่แตกต่างกัน ส่วนแรกคือส่วนกระบวนการภายในจะได้รับการแปลงโดยใช้เอพีไอสำหรับตัวแจงของอาวล์-เอส (OWL-S Parser API) ส่วนถัดมาคือส่วนที่เกี่ยวข้องกับออนโทโลจีของโดเมนต่างๆ ส่วนนี้จะได้รับการแปลงโดยตัวแจงสำหรับออนโทโลจีของจีนา (Jena) [17] และส่วนสุดท้ายคือส่วนที่เป็นภาษากฎสเวิร์ล จะได้รับการแปลงโดยตัวแจงภาษาสเวิร์ล จากนั้นนำนิพจน์เหล่านี้มาคิดหาเหตุผลการเข้าคู่กันโดยใช้เครื่องมือคิดหาเหตุผลได้แก่ บอสซาม (Bossam) [18] ซึ่งตรงส่วนนี้จะทำในลักษณะเดียวกับงานวิจัย [19]

3.2 เงื่อนไขในการเข้าคู่ (Matching Criteria)

การค้นหาเว็บเซอร์วิสที่มีกระบวนการเป็นไปตามที่ต้องการ จะพิจารณาองค์ประกอบต่างๆ ภายในข้อกำหนดอาวล์-เอสโพรเซสโมเดลว่าเข้าคู่กับข้อความค้นหาหรือไม่ โดยการพิจารณาจะอาศัยเงื่อนไขการเข้าคู่ต่างๆ ที่ประยุกต์ใช้จากงานวิจัย [15] ได้แก่ การเข้าคู่ของออนโทโลจีเทอม (หัวข้อที่ 3.2.1) การเข้าคู่ของช่วงตัวเลข (หัวข้อที่ 3.2.2) และการเข้าคู่ของเซตข้อมูลแจงนับ (หัวข้อที่ 3.2.3)

3.2.1 การเข้าคู่ของออนโทโลจีเทอม

เงื่อนไขการเข้าคู่นี้ใช้กับออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน รวมทั้งโดเมนออนโทโลจีอื่นๆ ที่เกี่ยวข้อง โดยมีพื้นฐานจากความสัมพันธ์แบบคลาส-ซับคลาสของออนโทโลจี ซึ่งมีใช้ในงานวิจัย [15] กำหนดให้ C_Q เป็นคอนเซ็ปต์ (Concept) ซึ่งระบุในการค้นหา และ C_P เป็นคอนเซ็ปต์ในข้อกำหนดกระบวนการ

1) ถ้า $C_Q \equiv C_P$ แล้ว C_P จะเข้าคู่อย่างถูกต้อง (Exact Match) สำหรับ C_Q เมื่อ $\equiv$ หมายถึง เท่าเทียมกับ โดยจะให้น้ำหนักเป็น 1
2) ถ้า $C_P \sqsubseteq C_Q$ แล้ว C_P จะเข้าคู่อย่างเจาะจง (Specialised Match) สำหรับ C_Q เมื่อ $\sqsubseteq$ หมายถึง ครอบคลุมโดย (Is Subsumed By) โดยจะให้น้ำหนักเป็น 2
3) ถ้า $C_Q \sqsubseteq C_P$ แล้ว C_P จะเข้าคู่อย่างกว้าง (Generalised Match) สำหรับ C_Q ซึ่งหมายความว่าคอนเซ็ปต์ของการค้นหาเจาะจงมากกว่าคอนเซ็ปต์ของข้อกำหนดกระบวนการ โดยจะให้น้ำหนักเป็น 3
4) ถ้า $(C_Q \not\sqsubseteq C_P) \wedge (C_P \not\sqsubseteq C_Q) \wedge (C_Q \sqsubseteq C_C) \wedge (C_P \sqsubseteq C_C)$ แล้ว C_P จะเข้าคู่บางส่วน (Partial Match) สำหรับ C_Q เมื่อ $\not\sqsubseteq$ หมายถึง ไม่ครอบคลุมโดย และ C_C เป็นโหนดในออนโทโลจีเดียวกัน โดยจะให้น้ำหนักเป็น 4
5) กรณีอื่นนอกเหนือจากสี่กรณีข้างต้น C_P จะไม่เข้าคู่ (Fail Match) สำหรับ C_Q โดยจะให้น้ำหนักเป็น ∞

ซึ่งสามารถสรุปโดยรวมได้ดังตารางที่ 3.1

ตารางที่ 3.1 สรุปค่าน้ำหนักของลักษณะการเข้าคู่ทางออนโทโลจีเทอม

ลักษณะการเข้าคู่	น้ำหนัก
เข้าคู่อย่างถูกต้อง	1
เข้าคู่อย่างเจาะจง	2
เข้าคู่อย่างกว้าง	3
เข้าคู่บางส่วน	4
ไม่เข้าคู่	∞

3.2.2 การเข้าคู่ของช่วงตัวเลข (Numerical Range)

ให้ N_Q เป็นเซตไม่ว่าง (Nonempty Set) ของค่าช่วงตัวเลขของนิพจน์ในการค้นหา (S_Q) และ N_P เป็นเซตไม่ว่างของค่าช่วงตัวเลขของนิพจน์ในข้อกำหนดกระบวนการ (S_P)

1) ถ้า $N_P \subseteq N_Q$ แล้ว S_P จะเข้าคู่อย่างถูกต้องสำหรับ S_Q โดยจะให้น้ำหนักเป็น 1
2) ถ้า $N_Q \subseteq N_P$ แล้ว S_P จะเข้าคู่แบบเสียบเข้า (Plug-In Match) สำหรับ S_Q โดยจะให้น้ำหนักเป็น 2
3) ถ้า $(N_P \cap N_Q \neq \phi) \wedge (N_P \not\subseteq N_Q) \wedge (N_Q \not\subseteq N_P)$ แล้ว S_P จะเข้าคู่อย่างอ่อน (Weak Match) สำหรับ S_Q โดยจะให้น้ำหนักเป็น 3
4) ถ้า $N_P \cap N_Q = \phi$ แล้ว S_P จะไม่เข้าคู่สำหรับ S_Q โดยจะให้น้ำหนักเป็น ∞

ซึ่งสามารถสรุปโดยรวมได้ดังตารางที่ 3.2

ตารางที่ 3.2 สรุปค่าน้ำหนักของลักษณะการเข้าคู่ของช่วงตัวเลข

ลักษณะการเข้าคู่	น้ำหนัก
เข้าคู่อย่างถูกต้อง	1
เข้าคู่อย่างเสียบเข้า	2
เข้าคู่อย่างอ่อน	3
ไม่เข้าคู่	∞

3.2.3 การเข้าคู่ของเซตข้อมูลแจงนับ (Enumeration Values)

ให้ E_Q เป็นเซตไม่ว่างของค่าข้อมูลแจงนับ i ของนิพจน์ในการค้นหา และ E_P เป็นเซตไม่ว่างของค่าข้อมูลแจงนับ j ของนิพจน์ในข้อกำหนดกระบวนการ โดยค่าข้อมูลแจงนับ อาจเป็นค่าข้อมูลพื้นฐานของเอกซ์เอ็มแอลสกีมา (XML Schema Data Type) หรือเป็นออนโทโลจีเทอม (Concept)

หากค่าข้อมูลแจงนับเป็นค่าพื้นฐานของเอกซ์เอ็มแอลสกีมา :

1.) $\forall i, \exists j : (i \in E_Q) \wedge (j \in E_P) \wedge (i = j)$ แล้ว E_P จะเข้าคู่อย่างถูกต้อง (Exact Match) สำหรับ E_Q โดยจะให้น้ำหนักรวมเท่ากับผลบวกของน้ำหนักการเข้าคู่สำหรับแต่ละคู่ของ i และ j ซึ่งมีค่าเท่ากับ 1

2.) กรณีอื่นนอกเหนือจากนี้ E_P จะไม่เข้าคู่ (Fail Match) สำหรับ E_Q โดยจะให้น้ำหนักเป็น ∞

หากค่าข้อมูลแจงนับเป็นออนโทโลจีเทอม :

1.) $\forall i, \exists j : (i \in E_Q) \wedge (j \in E_P) \wedge (i \otimes j)$ โดยที่ $\otimes$ หมายถึงการเข้าคู่ของออนโทโลจีเทอมตามหัวข้อที่ 3.1 แล้ว E_P จะเข้าคู่สำหรับ E_Q โดยจะให้น้ำหนักรวมเท่ากับผลบวกของน้ำหนักการเข้าคู่สำหรับแต่ละคู่ของ i และ j ซึ่งมีค่าตั้งแต่ 1-4 แล้วแต่กรณี

2.) กรณีอื่นนอกเหนือจากนี้ E_P จะไม่เข้าคู่ (Fail Match) สำหรับ E_Q โดยจะให้น้ำหนักเป็น ∞

ซึ่งสามารถสรุปโดยรวมได้ดังตารางที่ 3.3

ตารางที่ 3.3 สรุปค่าน้ำหนักของลักษณะการเข้าคู่ของเซตข้อมูลแจงนับ

ลักษณะข้อมูลแจงนับ	ลักษณะการเข้าคู่	น้ำหนัก
ค่าพื้นฐานทางเอกซ์เอ็มแอลสกีมา	เข้าคู่อย่างถูกต้อง	1
	ไม่เข้าคู่	∞
ค่าข้อมูลแจงนับเป็นออนโทโลจีเทอม	เข้าคู่อย่างถูกต้อง	1
	เข้าคู่อย่างเจาะจง	2
	เข้าคู่อย่างกว้าง	3
	เข้าคู่บางส่วน	4
	ไม่เข้าคู่	∞

ในการนำเว็บเซอร์วิสมาพิจารณาเงื่อนไขการเข้าคู่ตามหัวข้อที่ 3.2.1 – 3.2.3 นั้น หากเว็บเซอร์วิสที่นำมาพิจารณามีการเข้าคู่ที่มีค่าน้ำหนักเป็น ∞ จะไม่นำเว็บเซอร์วิสนั้นมาพิจารณาในการประกอบเว็บเซอร์วิส

3.3 ตัวอย่างการพิจารณาการประกอบกันของเว็บเซอร์วิสตามกระแสงาน

สมมติว่าส่วนทำการค้นหาเว็บเซอร์วิสพบสองเว็บเซอร์วิสที่มีพฤติกรรมเชิงหน้าที่ตรงตามที่กำหนดในส่วนของกระบวนการสั่งซื้อไวน์ในกระแสงาน โดยรูปแบบอย่างย่อที่อธิบายพฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสทั้งสองแสดงในรูปที่ 3.9 และ 3.10 ตามลำดับ

```
1 Service = WineShop1

2 prefix
3 cus =http://www.owl-ontologies.com/CustomerInfo.owl#
4 fd = http://www.w3.org/TR/2003/PR-owl-guide-20031209/Food#
5 pi = http://a.com/ontology#
6 bsd=http://www.owl-ontologies.com/WineShopDomain.owl#

7 Input {
8 <bsd:WineName>WineName_1.ParameterType =<fd:Wine>
9 <bsd:CustomerInfo>CustomerInfo_1.ParameterType = <cus:CustomerInfo>
10 }

11  Precondition{
12 <bsd:AcceptedCreditCard>
13 CreditCardType = Amex
14 }

15 Effect{
16 Constraint : if <bsd:CustomerLocation>CustomerLocation = Bangkok
17                 Then <bsd:DeliveryDay>DeliveryDay<=3
18 }

19 Output{
20 <bsd:OrderedWine>OrderedBeverage_1.ParameterType =
21 <pi:PurchaseOrderConfirmation>

22}
```

รูปที่ 3.9 รายละเอียดอย่างย่อของเว็บเซอร์วิสแรกที่เกี่ยวกับการสั่งซื้อไวน์

```
1 Service = WineShop2

2 prefix
3 cus =http://www.owl-ontologies.com/CustomerInfo.owl#
4 fd = http://www.w3.org/TR/2003/PR-owl-guide-20031209/Food#
5 pi = http://a.com/ontology#
6 bsd=http://www.owl-ontologies.com/WineShopDomain.owl#

7 Input {
8 <bsd:WineName>WineName_2.ParameterType = <fd:Bordeaux>
9 <bsd:CustomerInfo>Customerinfo_2.ParameterType = <cus:CustomerInfo>
10 }

11  Precondition{
12 <bsd:AcceptedCreditCard>
13 CreditCardType = Amex,Visa
14 }

15 Effect{
16 Constraint : if <bsd:CustomerLocation>CustomerLocation = Bangkok
17                 Then <bsd:DeliveryDay>DeliveryDay<=5
18 }

19 Output{
20 <bsd:OrderedWine>OrderedBeverage_2.ParameterType =
21 <pi:PurchaseOrderConfirmation>

22}
```

รูปที่ 3.10 รายละเอียดอย่างย่อของเว็บเซอร์วิสที่สองที่เกี่ยวกับการสั่งซื้อไวน์

รูปแบบเต็มจะมีลักษณะคล้ายคลึงกับรูปที่ 3.6 โดยในรูปแบบย่อมีการอธิบายว่าแต่ละส่วนมาจากออนโทโลจีใด อาทิเช่น บรรทัดที่ 7-10 ของรูปที่ 3.9 ในส่วนข้อมูลอินพุทที่รับชื่อของไวน์ (บรรทัดที่ 8) จะบอกว่าพารามิเตอร์นี้นำเข้ามาจากออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนร้านขายไวน์ (ในที่นี้ใช้ตัวแปรแทนคือ <bsd:WineName>) และมีชนิดพารามิเตอร์ที่นำเข้ามาจากออนโทโลจีอาหารในส่วนของไวน์ (ในที่นี้ใช้ตัวแทนคือ <fd:Wine>) เป็นต้น ในทำนองเดียวกันเราสามารถอธิบายรูปแบบอย่างย่อที่อธิบายพฤติกรรมเชิงหน้าที่ของสองเว็บเซอร์วิสที่เกี่ยวกับการค้นหาไวน์ซึ่งมีอินพุทตรงตามที่ผู้ประกอบเว็บเซอร์วิสกำหนด โดยรูปที่ 3.11 แสดงรายละเอียดอย่างย่อของเว็บเซอร์วิสแรก และ รูปที่ 3.12 แสดงรายละเอียดอย่างย่อของเว็บเซอร์วิสที่สอง

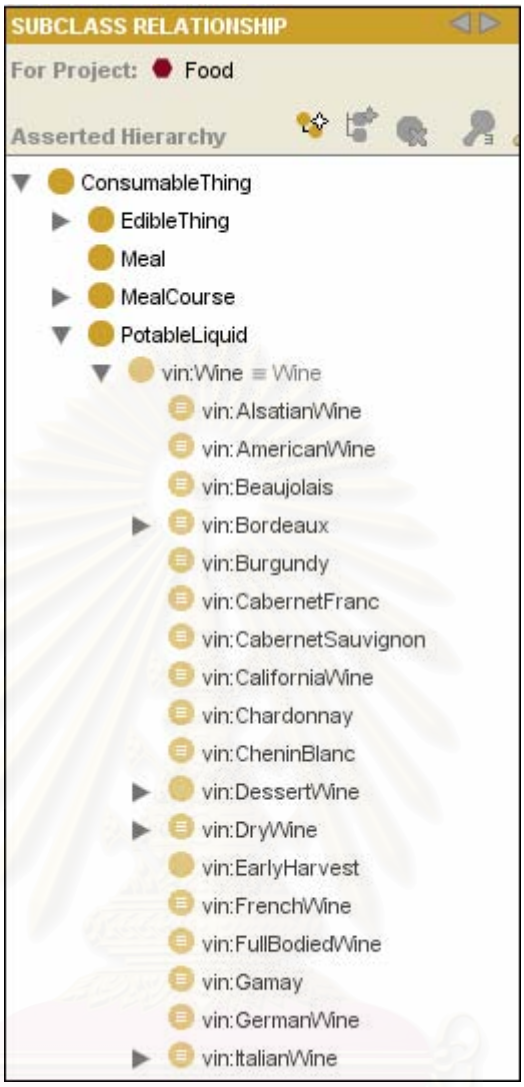
```
1 Service = WineAgent1

2 prefix
3 fd = http://www.w3.org/TR/2003/PR-owl-guide-20031209/Food#
4 wad=http://www.owl-ontologies.com/WineAgentDomain.owl#

5 Input {
6 <wad:FoodName>FoodName_1.ParameterType =<fd:MealCourse>

7 }

8 Output{
9 <wad:WineName>WineName_1.ParameterType =<fd:DryWine>

10}
```

รูปที่ 3.11 รายละเอียดอย่างย่อของเว็บเซอร์วิสแรกที่เกี่ยวกับการค้นหาไวน์

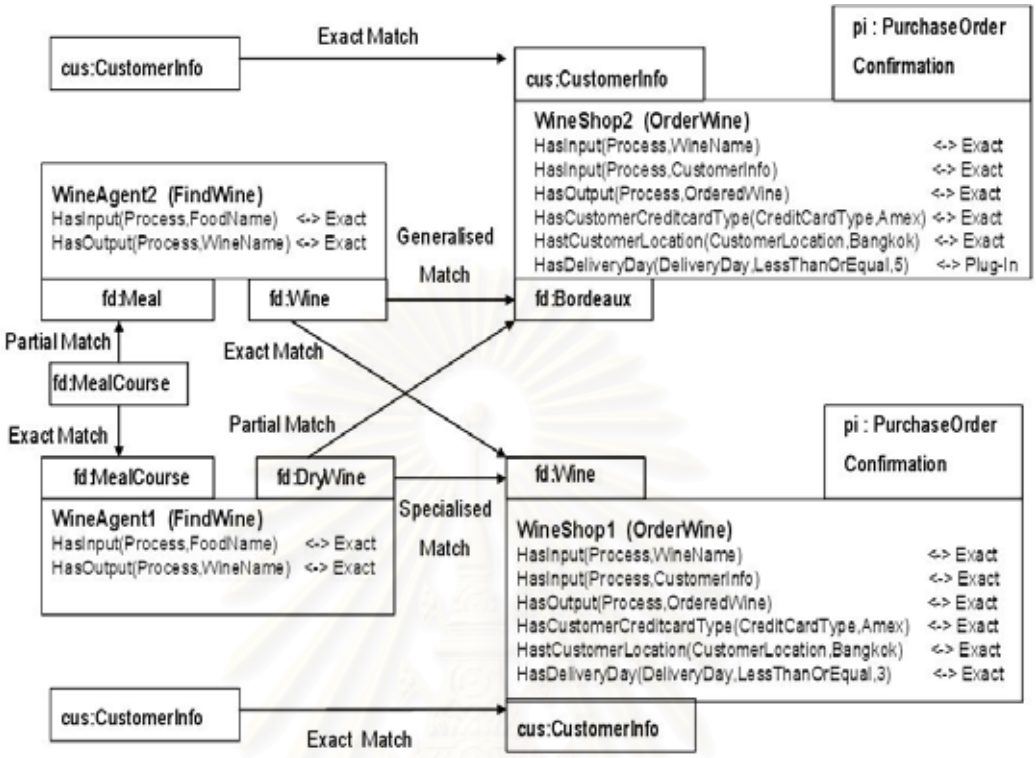
```
1 Service = WineAgent2

2 prefix
3 fd = http://www.w3.org/TR/2003/PR-owl-guide-20031209/Food#
4 wad=http://www.owl-ontologies.com/WineAgentDomain.owl#

5 Input {
6 <wad:FoodName>FoodName_2.ParameterType = <fd:Meal>

7 }

8 Output{
9 <wad:WineName>WineName_2.ParameterType = <fd:Wine>

10}
```

รูปที่ 3.12 รายละเอียดอย่างย่อของเว็บเซอร์วิสที่สองที่เกี่ยวกับการค้นหาไวน์

ในขั้นตอนการสร้างแผนงานที่เป็นไปได้ตามกระแสงานที่นำเข้ามาทั้งหมด ระบบจะทำการตรวจสอบการเข้าคู่ของเอาท์พุทของเซอร์วิสการค้นหาไวน์กับอินพุทของเซอร์วิสที่ให้บริการสั่งซื้อไวน์ และ เซอร์วิสค้นหาคำอธิบายรายละเอียดของไวน์ โดยใช้ออนโทโลจีเกี่ยวกับอาหาร ดังแสดงในรูปที่ 3.13 ในตัวอย่างนี้เราจะให้ความสำคัญกับเซอร์วิสการให้บริการสั่งซื้อไวน์ มากกว่าเซอร์วิสการค้นหาคำอธิบายรายละเอียดของไวน์ ดังนั้น ระบบจะทำการค้นหาคู่เซอร์วิสที่เหมาะสมกันระหว่างเซอร์วิสค้นหาไวน์กับเซอร์วิสให้บริการสั่งซื้อไวน์ก่อนดังแสดงในรูปที่ 3.14 โดยการเข้าคู่นั้นจะดูทั้งการเข้าคู่โดยเงื่อนไขทางออนโทโลจีเทอมของพฤติกรรมเชิงหน้าที่และเงื่อนไขทางช่วงตัวเลขของข้อบังคับในกระบวนการ จากนั้น ระบบจะทำการตรวจสอบเซอร์วิสเกี่ยวกับการค้นหาคำอธิบายไวน์ที่เหมาะสมกับเซอร์วิสการค้นหาไวน์ซึ่งได้จากการเข้าคู่ที่เหมาะสมแล้วกับเว็บเซอร์วิสการสั่งซื้อไวน์อีกทีหนึ่ง



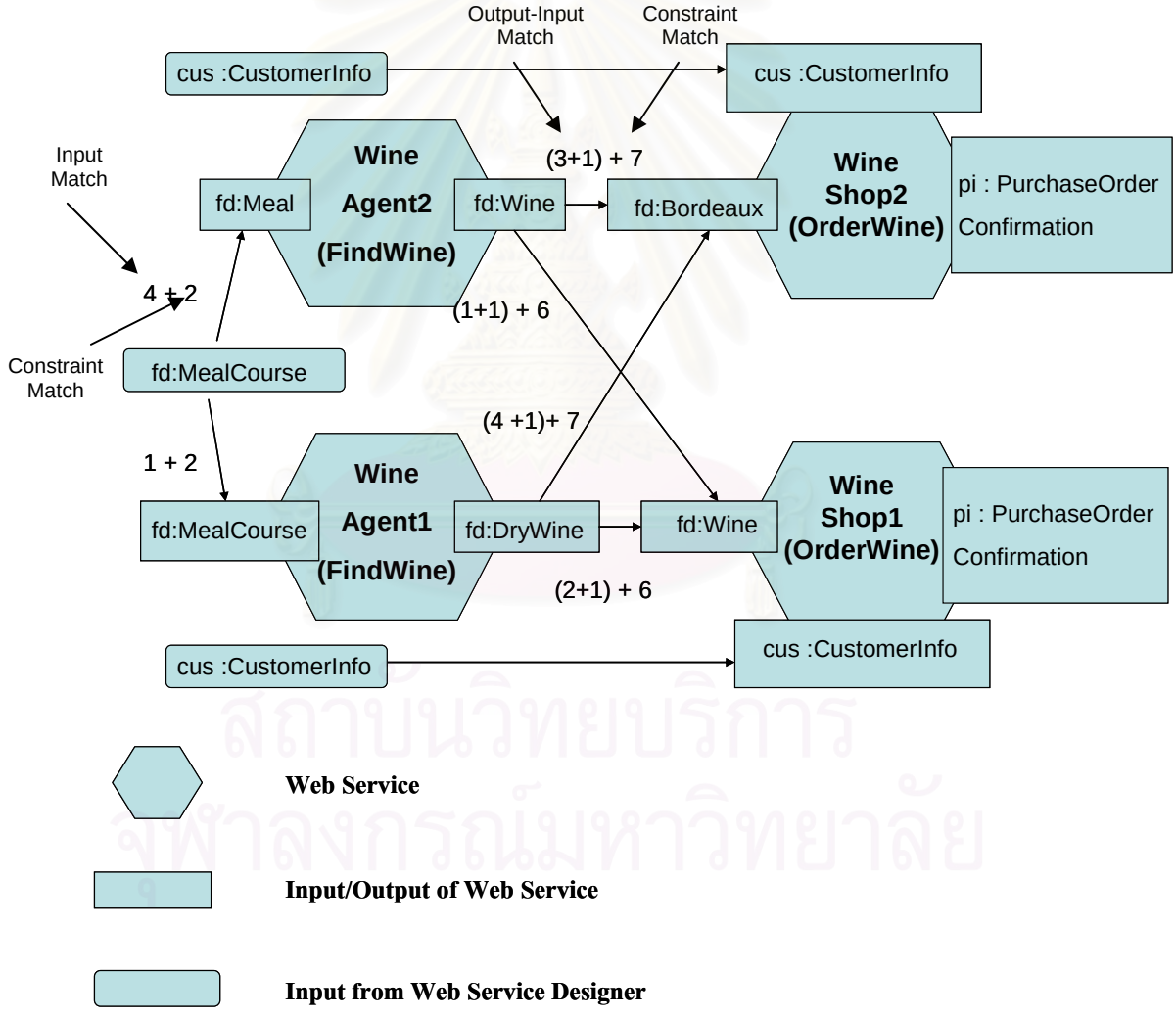
รูปที่ 3.13 ออนโทโลจีอาหาร



รูปที่ 3.14 การเข้าคู่กันระหว่างเซอร์วิสค้นหาไวน์และเซอร์วิสสั่งซื้อไวน์

การพิจารณาเงื่อนไขการเข้าคู่ระหว่างอินพุทและเอาท์พุทของสองเว็บเซอร์วิสที่จะพิจารณาตามหัวข้อที่ 3.2 จะถูกเปลี่ยนเป็นค่าน้ำหนักดังที่กำหนดเพื่อใช้ในการหาแผนงานที่ดีที่สุดโดยอาศัยวิธีการหาเส้นทางที่สั้นที่สุดโดยอัลกอริทึมของไดจ์สตรา โดยค่าน้ำหนักที่จะแทนในแต่ละเส้นของกราฟจะเท่ากับ น้ำหนักการเข้าคู่กันระหว่างอินพุทและเอาท์พุทของเว็บเซอร์วิสที่เกี่ยวข้องบวกกับผลรวมของการเข้าคู่ของข้อบังคับในกระบวนการ ตัวอย่างเช่น ผู้ประกอบเว็บเซอร์วิสระบุพฤติกรรมเชิงหน้าที่และข้อบังคับสำหรับเว็บเซอร์วิสที่ให้บริการสั่งซื้อไวน์ทั้งหมดหกข้อ (ดังรูปที่ 3.8) สมมติให้เว็บเซอร์วิส WineShop1 ที่ให้บริการสั่งซื้อไวน์ (ดังรูปที่ 3.9 และ 3.14) มีพฤติกรรมเชิงหน้าที่และข้อบังคับในกระบวนการที่เข้าคู่กับที่ผู้ประกอบเว็บเซอร์วิสกำหนดทั้งหกข้อและทุกข้อเป็นแบบเข้าคู่อย่างถูกต้อง (Exact Match) ซึ่งการเข้าคู่ลักษณะนี้เราให้น้ำหนักเท่ากับ 1 เพราะฉะนั้นผลรวมของการเข้าคู่ของข้อกำหนดกระบวนการจึงเท่ากับ 6 ในขณะที่เว็บเซอร์วิส WineShop2 ที่ให้บริการสั่งซื้อไวน์เช่นกัน (ดังรูปที่ 3.10 และ 3.14) เข้าคู่อย่างถูกต้อง 5 ข้อและเข้าคู่แบบเสียบเข้า (Plug-In Match) 1 ข้อในเรื่องเกี่ยวกับจำนวนวันในการส่งของ ซึ่งใช้เวลาน้อยกว่า 5 วัน แต่ผู้ประกอบเว็บเซอร์วิสต้องการน้อยกว่า 3 วัน จึงมีน้ำหนักเท่ากับ 2 ในข้อนี้ ดังนั้นผลรวมของการเข้าคู่ของข้อกำหนดกระบวนการจึงเท่ากับ 7 จากนั้นจึงพิจารณาน้ำหนักการเข้าคู่กันระหว่างอินพุทและเอาท์พุท เช่น เซอร์วิสการค้นหาไวน์ WineAgent2 (ดังรูปที่ 3.12) มีชนิดของเอาท์พุทที่

เข้าคู่อย่างถูกต้องตามเงื่อนไขทางออนโทโลจีเทอมกับชนิดอินพุทของเซอร์วิสการสั่งซื้อไวน์ WineShop1 จึงมีน้ำหนักเป็น 1 แต่ในขณะเดียวกันเซอร์วิสการค้นหาไวน์ WineAgent2 มีเงื่อนไขการเข้าคู่กันทางออนโทโลจีกับเซอร์วิสสั่งซื้อไวน์ WineShop2 แบบเข้าคู่อย่างกว้าง (Generalised Match) จึงมีน้ำหนักเป็น 3 นอกจากนี้เซอร์วิสสั่งซื้อไวน์ WineShop1 และ WineShop2 ต่างก็มีการเข้าคู่อย่างถูกต้องตามเงื่อนไขทางออนโทโลจีเทอมกับชนิดอินพุทข้อมูลลูกค้าที่ผู้ประกอบเว็บเซอร์วิสกำหนดมา จึงมีน้ำหนักเป็น 1 เป็นต้น โดยสรุปแล้วน้ำหนักการเข้าคู่กันระหว่างอินพุทและเอาท์พุทของเว็บเซอร์วิสที่เกี่ยวข้อง กับ ผลรวมของการเข้าคู่ของข้อบังคับกระบวนการระหว่างเซอร์วิสค้นหาไวน์และเซอร์วิสสั่งซื้อไวน์แสดงได้ดังรูปที่ 3.15โดยนำรูปแบบสัญลักษณ์การนำเสนอมาจากงานวิจัย [14] และปรับรูปแบบเพื่อให้ง่ายต่อการเข้าใจ



รูปที่ 3.15 น้ำหนักของแต่ละเส้นเชื่อมต่อระหว่างเซอร์วิสค้นหาไวน์และเซอร์วิสสั่งซื้อไวน์

ระบบจะรับค่าความพึงพอใจจากผู้ประกอบเว็บเซอร์วิส ซึ่งแทนด้วยสัญลักษณ์ λ และมีค่าไม่เกิน 1 ซึ่งจะบ่งบอกว่าผู้ประกอบเว็บเซอร์วิสให้ความสำคัญกับการเข้าคู่ของอินพุทและเอาท์พุทของคู่เว็บเซอร์วิสหรือผลรวมของการเข้าคู่ของข้อบังคับในกระบวนการมากกว่ากัน ดังนั้นค่าน้ำหนักบนแต่ละเส้นเชื่อมต่อจะกลายเป็น

น้ำหนัก = λ*น้ำหนักการเข้าคู่กันระหว่างอินพุทและเอาท์พุทของคู่เว็บเซอร์วิส + (1-λ) ผลรวมของการเข้าคู่ของข้อบังคับในกระบวนการของเว็บเซอร์วิส

จากสูตร เราจะสามารถคำนวณเส้นทางที่สั้นที่สุดซึ่งหมายถึง แผนงานการประกอบเว็บเซอร์วิสที่ตรงกับความต้องการของผู้ประกอบเว็บเซอร์วิสมากที่สุดได้ ตารางที่ 3.4 แสดงเส้นทางที่สั้นที่สุดที่คำนวณได้สำหรับกราฟรูปที่ 3.15 เมื่อกำหนดค่า λ เป็นค่าต่างๆ

ตารางที่ 3.4 เส้นทางที่สั้นที่สุดที่คำนวณได้ภายใต้ค่าความพอใจต่างๆ

λ	เส้นทางที่สั้นที่สุด	ค่าน้ำหนักรวม
λ =0	WineAgent1->WineShop1 WineAgent2->WineShop1	8
λ =0.5	WineAgent1->WineShop1	6
λ =1	WineAgent1->WineShop1	4

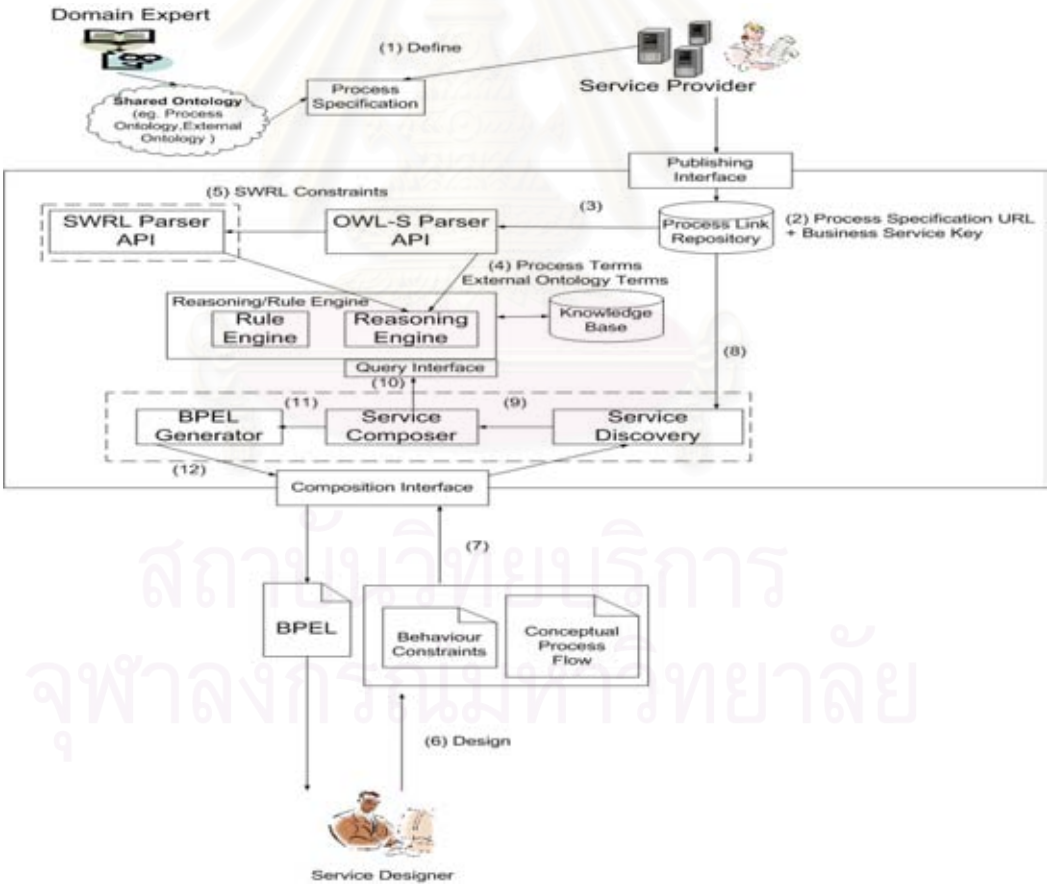
ถ้าเส้นทางที่สั้นที่สุดที่หาได้มีมากกว่าหนึ่งทางเลือก (ดังเช่นกรณี λ = 0 ข้างต้น) ระบบจะสุ่มเลือกมาเส้นทางหนึ่ง เมื่อระบบได้เว็บเซอร์วิสที่เกี่ยวข้องทั้งหมดที่จะใช้ในกระแสงานแล้ว ระบบจะนำข้อมูลเหล่านี้ไปสร้างเป็นเอกสารบีเพลต่อไป

บทที่ 4
สถาปัตยกรรมการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยอาศัยข้อกำหนดกระบวนการในรูปอาวล์-เอสโพรเซสโมเดล

ในบทนี้จะกล่าวถึงการออกแบบและพัฒนาสถาปัตยกรรมการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยอาศัยข้อกำหนดกระบวนการในรูปอาวล์-เอสโพรเซสโมเดล โดยสถาปัตยกรรมประกอบด้วยตัวกลาง (Mediator) ที่ทำหน้าที่รับการประกาศบริการและให้บริการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

4.1 การออกแบบการทำงานของตัวกลาง

การทำงานของตัวกลางสามารถแสดงเป็นเฟรมเวิร์กได้ดังรูปที่ 4.1



รูปที่ 4.1 เฟรมเวิร์กสำหรับการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

ส่วนประกอบภายในตัวกลางมีดังนี้

1. **ส่วนของตัวกลางที่ทำหน้าที่รับการประกาศจากผู้ให้บริการ (Publishing Interface)** ทำหน้าที่รับการลงทะเบียนจากผู้ให้บริการและส่งรายละเอียดการลงทะเบียนที่เกี่ยวข้องไปเก็บไว้ที่ฐานข้อมูลเชื่อมโยงข้อกำหนดกระบวนการ

2. **ส่วนของตัวกลางที่ทำหน้าที่รับข้อมูลการประกอบเว็บเซอร์วิส (Composition Interface)** ทำหน้าที่รับกระแสงานอาวล์-เอส และ ข้อมูลพฤติกรรมเชิงหน้าที่จากนักออกแบบเว็บเซอร์วิสเพื่อเข้ามาประมวลผลยังตัวกลาง

3. **ฐานข้อมูลเชื่อมโยงข้อกำหนดกระบวนการ (Process Link Repository)** ทำหน้าที่เก็บยูอาร์แอลข้อกำหนดกระบวนการและรหัสบิสิเนสเซอร์วิสที่ได้จากการลงทะเบียนเว็บเซอร์วิสไว้กับยูดีดีไอ [4] โดยฐานข้อมูลนี้จะรับข้อมูลมาจากส่วนของตัวกลางที่ทำหน้าที่รับการประกาศจากผู้ใช้บริการ

4. **ตัวแจงอาวล์-เอส (OWL-S Paser API)** ทำหน้าที่ในการแยกส่วน เทอมของกระบวนการ เทอมของออนโทโลจีภายนอก และ ในส่วนข้อบังคับที่เป็นภาษาสเวิร์ล จากข้อกำหนดกระบวนการ

5. **ตัวแจงสเวิร์ล (SWRL Paser API)** ทำหน้าที่แปลงภาษาสเวิร์ล ให้เป็นลักษณะคำสั่งเพื่อให้เครื่องมืออนุมานสามารถเข้าใจและอนุมานได้เพื่อเก็บไว้ในฐานข้อมูลความรู้ (Knowledge Base)

6. **เครื่องมืออนุมาน (Reasoning/Rule Engine)** ทำหน้าที่อนุมานเทอมของกระบวนการ เทอมของออนโทโลจีภายนอกและข้อบังคับที่เป็นภาษาสเวิร์ล และเก็บไว้ในฐานข้อมูลความรู้

7. **ส่วนค้นหาเซอร์วิส (Service Discovery)** ทำหน้าที่อ่านเอกสารข้อบังคับกระบวนการทำงานและทำการค้นหาเว็บเซอร์วิสที่ตรงตามข้อบังคับกระบวนการทำงานและส่งยูอาร์เอลของเว็บซอร์วิสที่เกี่ยวข้องที่ได้จากฐานข้อมูลเชื่อมโยงข้อกำหนดกระบวนการให้ส่วนประกอบเว็บเซอร์วิส

8. **ส่วนประกอบเซอร์วิส (Service Composer)** ทำหน้าที่หาแผนงานการประกอบเว็บเซอร์วิสที่ดีที่สุดโดยอาศัยอัลกอริทึมไดจ์สตรา

9. **ส่วนสร้างเอกสารบีเพล (BPEL Genrator)** ทำหน้าที่สร้างเอกสารบีเพลจากแผนงานที่ดีที่สุดที่ประมวลได้จากส่วนประกอบเว็บเซอร์วิส

ทั้งนี้ผู้วิจัยได้พัฒนา ตัวแจงสเวิร์ล ส่วนค้นหาเซอร์วิส ส่วนประกอบเซอร์วิส และส่วนสร้างเอกสารบีเพล ขึ้นเอง (ส่วนที่ล้อมรอบด้วยเส้นประในรูปที่ 4.1)

4.2 ขั้นตอนการทำงาน

จากรูปที่ 4.1 ขั้นตอนการทำงานเพื่อการประกอบเว็บเซอร์วิสมีดังนี้

(1) ผู้ให้บริการสร้างข้อกำหนดกระบวนการอาวล์-เอสโพรเซสโมเดลสำหรับเว็บเซอร์วิสของตนเอง โดยข้อกำหนดกระบวนการจะอยู่บนพื้นฐานของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน (Shared Process Ontology of Domain) และ ออนโทโลจีภายนอก (External Ontology) ซึ่งจะสร้างโดยผู้เชี่ยวชาญของโดเมนนั้น (Domain Expert) การสร้างข้อกำหนดกระบวนการของผู้ให้บริการสามารถกระทำได้โดยใช้เครื่องมือสร้างออนโทโลจีโพรทีเจ (Protégé)

(2) ทุกๆเซอร์วิสจะทำการลงทะเบียนกับยูดีดีไอ (UDDI) และจากนั้นจะมาทำการลงทะเบียนกับส่วนของตัวกลางที่ทำหน้าที่รับการประกาศจากผู้ให้บริการ (Publishing Interface) โดย ยูอาร์แอลข้อกำหนดกระบวนการและรหัสบิสิเนสเซอร์วิสที่ได้จากการลงทะเบียนกับยูดีดีไอจะถูกจัดเก็บไว้ที่ฐานข้อมูลเชื่อมโยงข้อกำหนดกระบวนการ (Process Link Repository)

(3) ตัวแจงอาวล์-เอส (OWL-S Pasrer API) จะถูกใช้ในการแยกส่วน เทอมของกระบวนการ เทอมของออนโทโลจีภายนอก และ ในส่วนข้อบังคับที่เป็นภาษาสเวิร์ล จากข้อกำหนดกระบวนการ

(4) เทอมของกระบวนการและเทอมของออนโทโลจีภายนอก จะถูกอนุมานโดยเครื่องมืออนุมาน แล้วข้อมูลทุกอย่างที่อนุมานได้จะถูกนำมาเก็บไว้เป็นองค์ความรู้ในฐานความรู้ (Knowledge Base)

(5) ข้อบังคับที่เป็นภาษาสเวิร์ลจะถูกแจงโดยตัวแจงสเวิร์ล (SWRL Paser API) และถูกอนุมานโดยเครื่องมืออนุมาน แล้วข้อมูลทุกอย่างที่อนุมานได้จะถูกนำมาเก็บไว้เป็นองค์ความรู้ในฐานความรู้

(6) ในการประกอบเว็บเซอร์วิส นักออกแบบเว็บเซอร์วิสจะทำการออกแบบโดยอ้างอิงออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน และ ออนโทโลจีภายนอก จากนั้นจะทำการออกแบบกระแสงานอาวล์-เอส และ ข้อมูลพฤติกรรมเชิงหน้าที่ โดยใช้เครื่องมือโพรทีเจ

(7) ทำการส่งกระแสงานอาวล์-เอส และ ข้อมูลพฤติกรรมเชิงหน้าที่ให้ส่วนของตัวกลางที่ทำหน้าที่รับข้อมูลการประกอบเว็บเซอร์วิส (Composition Interface)

(8) กระแสงานอาวล์-เอส และ ข้อมูลพฤติกรรมเชิงหน้าที่ จะถูกใช้ในการทำการค้นหาบริการที่เกี่ยวข้องจากฐานข้อมูลเชื่อมโยงข้อกำหนดกระบวนการ

(9) บริการที่เกี่ยวข้องทั้งหมดจะถูกนำมาเปรียบเทียบโดยส่วนของการประกอบเว็บเซอร์วิส (Service Composer) เพื่อหาบริการที่ตรงกับความต้องการของนักออกแบบเว็บเซอร์วิส

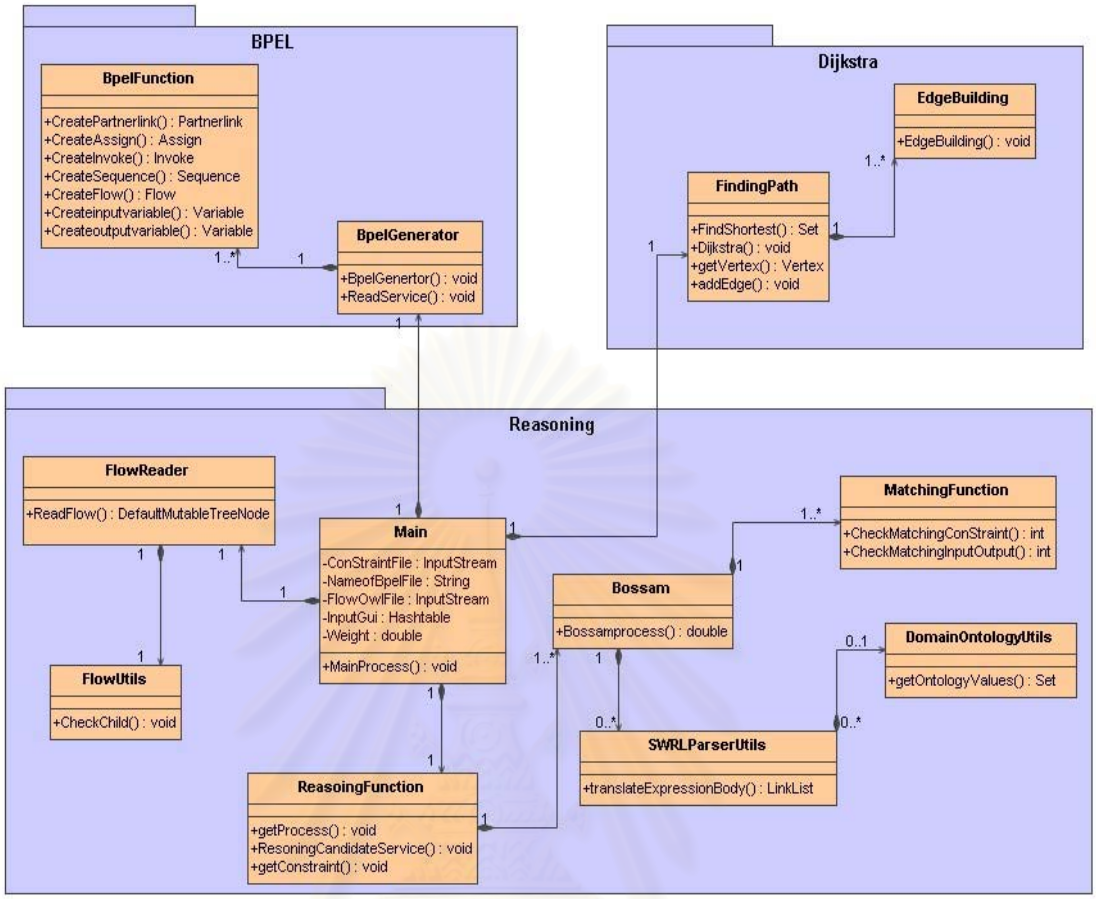
(10) ส่วนของการประกอบเว็บเซอร์วิสจะทำการติดต่อกับเครื่องมืออนุมานเพื่อตรวจสอบข้อเท็จจริง (Fact) จากฐานความรู้และคำนวณการเข้าคู่ของข้อมูลพฤติกรรมเชิงหน้าที่ กับ เครื่องมือตรวจสอบกฎ (Rule Engine)

(11) เมื่อได้การประกอบเว็บเซอร์วิสที่ดีที่สุดแล้วจะทำการสร้างเอกสารบีเพล ในส่วนของการสร้างเอกสารบีเพล (BPEL Generator)

(12) ทำการส่งเอกสารบีเพล และ เอกสารดับเบิลยูเอสดีแอลที่เกี่ยวข้องให้กับนักออกแบบเว็บเซอร์วิส

4.3 การออกแบบตัวกลาง

การออกแบบตัวกลางสามารถแสดงได้ด้วยแผนภาพคลาส โดยแบ่งออกเป็น 3 แพ็คเกจคือ แพ็คเกจการอนุมานออนโทโลจี แพ็คเกจการค้นหาแผนงานที่ดีที่สุด และ แพ็คเกจการสร้างเอกสารบีเพล ดังรูปที่ 4.2 โดยแพ็คเกจการอนุมานออนโทโลจีมีหน้าที่รับข้อมูลเกี่ยวกับกระแสงานอาวล์-เอส และ ข้อมูลพฤติกรรมเชิงหน้าที่และทำการอนุมานตรวจสอบโดยใช้เครื่องมือการอนุมาน แพ็คเกจการค้นหาแผนงานที่ดีที่สุดมีหน้าที่ค้นหาแผนงานที่ดีที่สุดโดยอาศัยอัลกอริทึมไดจ์สตราและแพ็คเกจการสร้างเอกสารบีเพลทำการรับแผนงานที่ดีที่สุดจากแพ็คเกจการค้นหาแผนงานที่ดีที่สุดมาทำการสร้างเอกสารบีเพล



รูปที่ 4.2 แผนภาพคลาสของตัวกลางสำหรับการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

4.4 เครื่องมือที่ใช้ในการพัฒนา

ในการพัฒนาตัวกลางในการประกอบเว็บเซอร์วิสนั้น ผู้วิจัยได้ใช้เครื่องมือในการพัฒนาดังนี้

1. จาวาเอสดีเค (Java SDK) รุ่น 1.5 เพื่อใช้เป็นตัวแปลโปรแกรม
2. อิคลิพซ์ (Eclipse) รุ่น 3.2 สำหรับเป็นเครื่องมือและสภาพแวดล้อมสำหรับการพัฒนาตัวกลาง
3. เจเอสพี (JSP) รุ่น 2.4 สำหรับเป็นภาษาในการเขียนสคริปต์
4. อาพาเช ทอมแคท (Apache Tomcat) รุ่น 5.0 สำหรับเป็นเว็บคอนเทนเนอร์
5. มายเอสคิวแอล (Mysql) รุ่น 5.0.24 สำหรับเป็นฐานข้อมูล
6. โพรทีเจ รุ่น 3.1 สำหรับสร้างแฟ้มข้อมูลอาวล์ของผู้ให้บริการ และโดเมนออนโทโลจี
7. จีนา รุ่น 2.1 สำหรับประมวลผลออนโทโลจี
8. บอสซาม รุ่น 0.8 สำหรับประมวลผลการเข้าคู่ของบริการ

9. ตัวแจงอาวล์-เอส เอพีไอ รุ่น 1.0 สำหรับประมวลผลแฟ้มข้อมูลอาวล์-เอส

10. ไอบีเอ็ม บีพีดับเบิลยูเอสฟอร์เจ (BPWS4J) [20] รุ่น 2.1 สำหรับทดสอบเอกสารบีเพล

11. เน็ตบีนส์ (NetBeans) [21] รุ่น 5.5 ใช้สำหรับสร้างเอกสารบีเพลแบบปกติเพื่อนำมาเปรียบเทียบ

บทที่ 5
การทดสอบตัวกลางในการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยอาศัยข้อกำหนดกระบวนการในรูปอาวล์-เอสโพรเซสโมเดล

การทดสอบด้วยกรณีศึกษานั้นจะทดสอบด้วยกรณีศึกษาที่มีโครงสร้างกระแสงานแบบลำดับร่วมกับแบบแบ่ง-ร่วม กรณีศึกษาที่มีโครงสร้างกระแสงานแบบลำดับร่วมกับแบบแบ่ง และเปรียบเทียบผลลัพธ์ที่ได้จากตัวกลางกับผลลัพธ์ที่ได้จากวิธีปกติ โดยจะแสดงการใช้งานตัวกลางที่พัฒนาขึ้นโดยละเอียด

5.1 กรณีศึกษาที่มีโครงสร้างกระแสงานแบบลำดับร่วมกับแบบแบ่ง-ร่วม

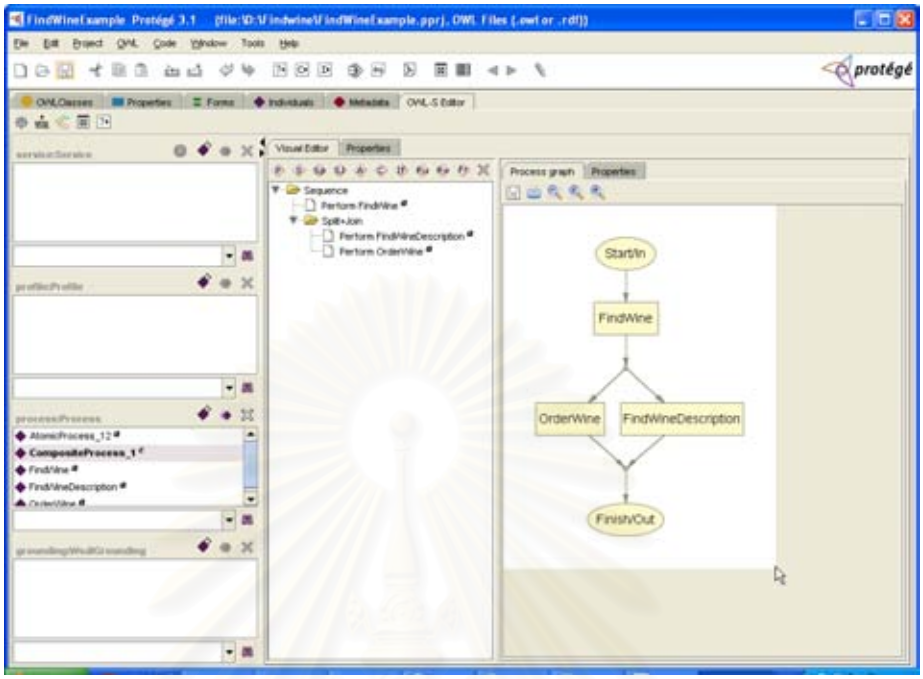
กรณีศึกษานี้จะใช้ตัวอย่างในหัวข้อที่ 3.1 ซึ่งได้แก่ “ผู้ใช้บริการซึ่งเป็นนักออกแบบเว็บเซอร์วิสต้องการประกอบเว็บเซอร์วิสที่มีบริการการสั่งซื้อไวน์ที่เหมาะกับรายการอาหารที่กำหนด โดยมีข้อบังคับว่าบริการนี้จะรับบัตรเครดิตของบริษัทเอเม็กซ์ (Amex) และต้องมีบริการส่งถึงบ้านภายใน 3 วัน โดยที่อยู่ของลูกค้าอยู่ในกรุงเทพฯ พร้อมทั้งแสดงรายละเอียดของไวน์ที่สั่งซื้อด้วย” โดยนักออกแบบเว็บเซอร์วิสออกแบบกระแสงานโดยใช้โปรแกรมโพรทีเจ ดังรูปที่ 5.1 และออกแบบข้อกำหนดกระบวนการของแต่ละกระบวนการโดยอาศัยออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของกระบวนการนั้นๆดังในรูปที่ 5.2 ซึ่งจะใช้เว็บเซอร์วิสที่มีพฤติกรรมเชิงหน้าที่ต่างๆกันในการทดลอง ดังตารางที่ 5.1 และออนโทโลจีที่ใช้ในกรณีศึกษานี้ดังรูปที่ 5.3

ตารางที่ 5.1 พฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่ใช้ในกรณีศึกษาแรก

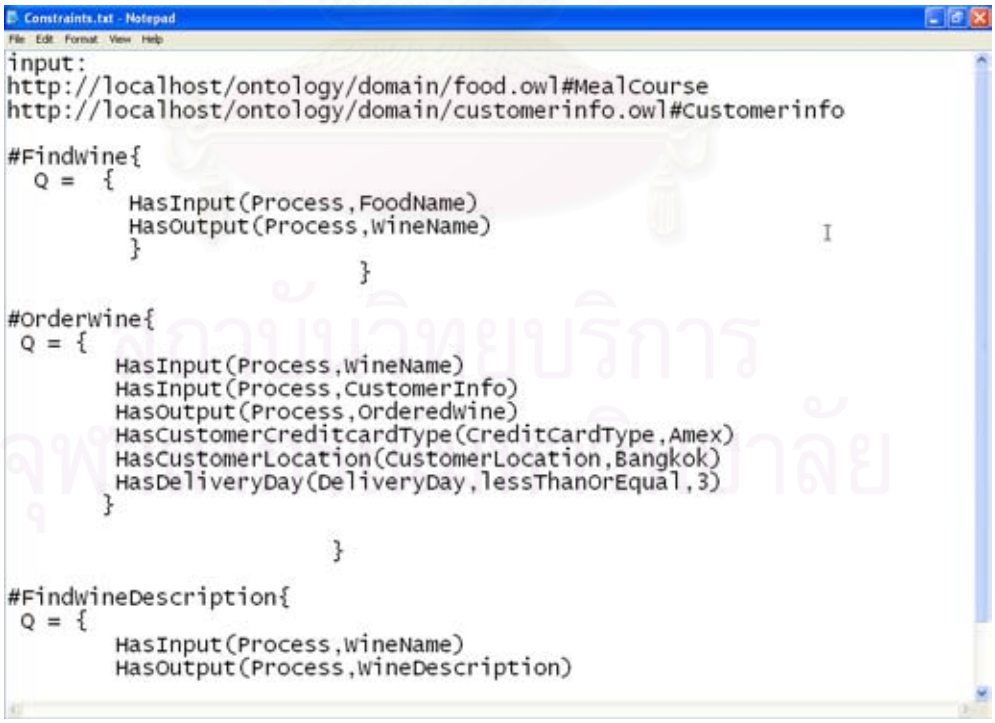
ชื่อเว็บเซอร์วิส	ลักษณะ
FindWine1	Input=MealCourse Output = RedWine
FindWine2	Input = RedMeatCourse Output = Wine
FindWine3	Input = FishCourse Output = DryWine

ตารางที่ 5.1 พฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่ใช้ในกรณีศึกษาแรก (ต่อ)

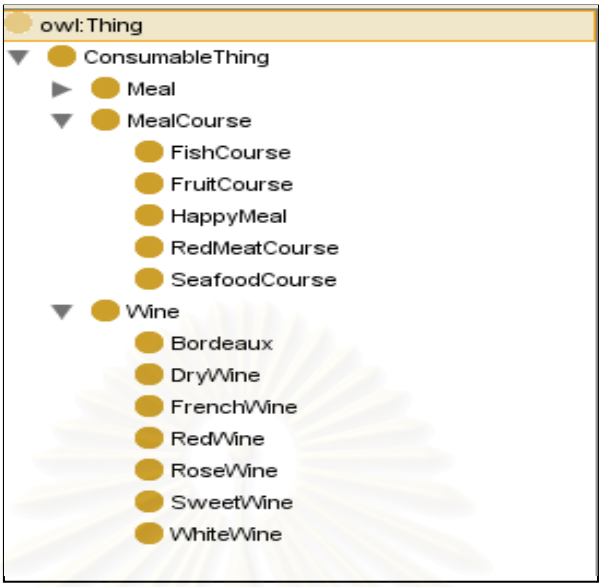
OrderWine1	Input = Wine Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=3 Town = Bangkok CreditType = Amex
OrderWine2	Input = DryWine Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=5 Town = Bangkok CreditType = Visa
OrderWine3	Input = Bordeaux Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=4 Town = Bangkok CreditType = Amex,Visa
FindWineDescription1	Input = RedWine Output = WineDescription
FindWineDescription2	Input = Wine Output = WineDescription
FindWineDescription3	Input = Bordeaux Output = WineDescription



รูปที่ 5.1 การออกแบบกระแสงานโดยโปรแกรมโพรทีเจของกรณีศึกษาแรก

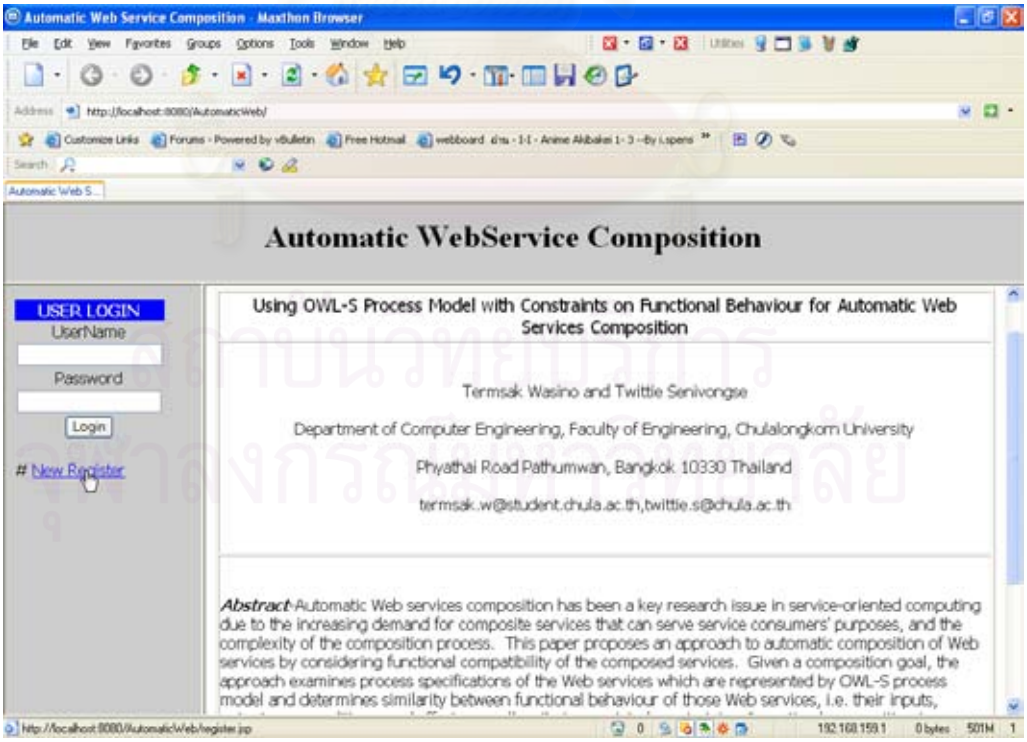


รูปที่ 5.2 การกำหนดข้อกำหนดกระบวนการของแต่ละกระบวนการของกรณีศึกษาแรก

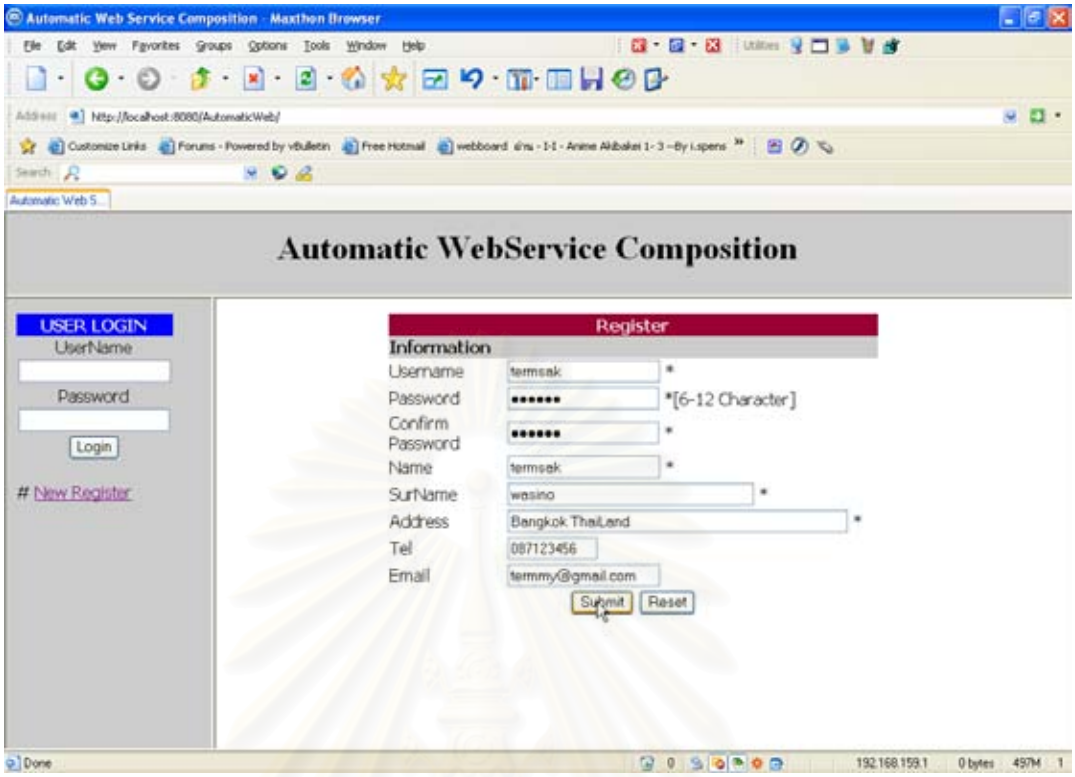


รูปที่ 5.3 ออนโทโลจีที่ใช้ในกรณีศึกษาแรก

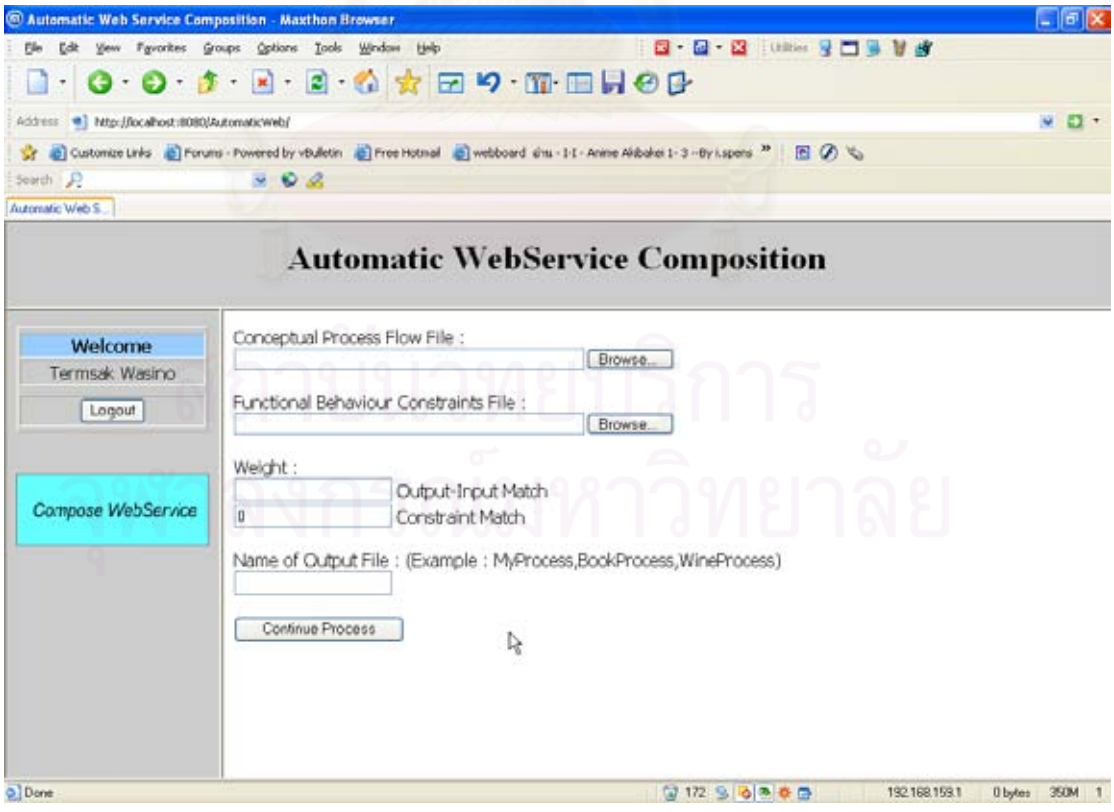
หลังจากจัดเตรียมไฟล์อาวล์-เอสจากการสร้างด้วยโปรแกรมโพรทีเจ และ ไฟล์ที่ทำการกำหนดข้อกำหนดกระบวนการต่างๆแล้ว ให้เข้าไปยังเว็บไซต์ของตัวกลางดังรูปที่ 5.4 จากนั้นทำการสมัครสมาชิกเพื่อขอใช้บริการ ดังรูปที่ 5.5 และทำการล็อกอิน เพื่อเข้าไปใช้งานดังรูปที่ 5.6



รูปที่ 5.4 หน้าเว็บของตัวกลาง

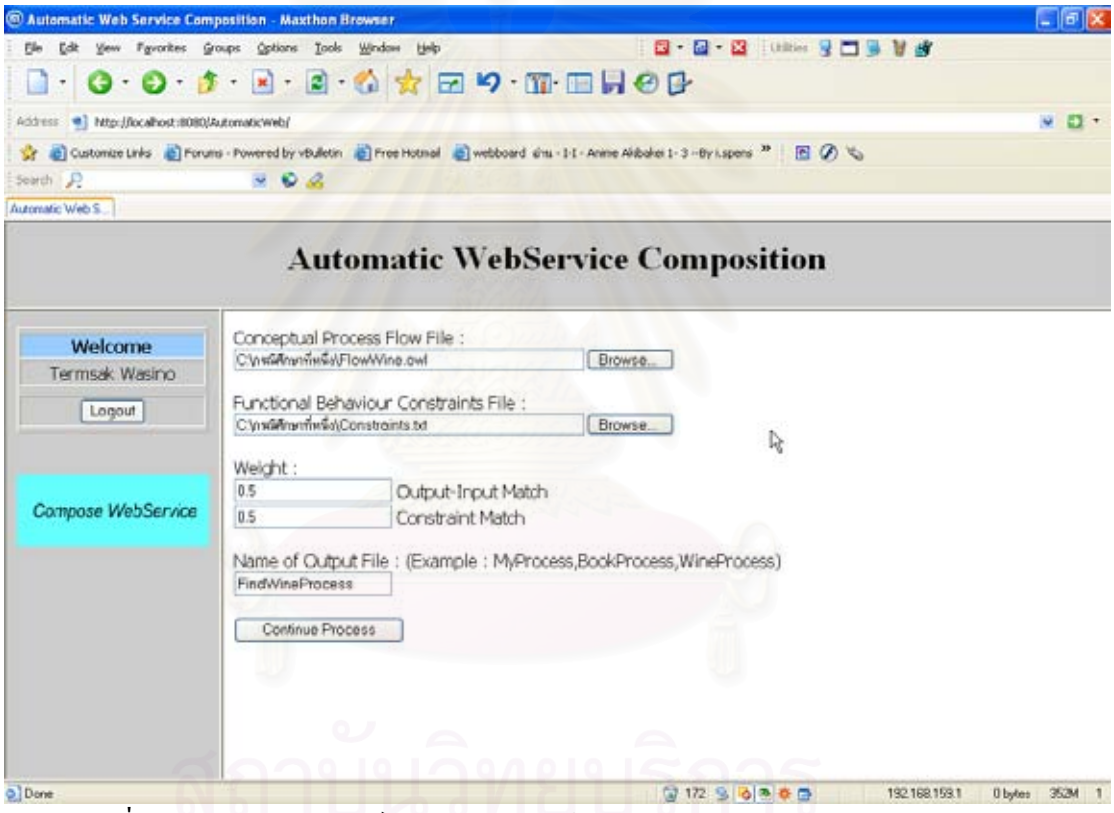


รูปที่ 5.5 การสมัครสมาชิกเพื่อขอใช้บริการ

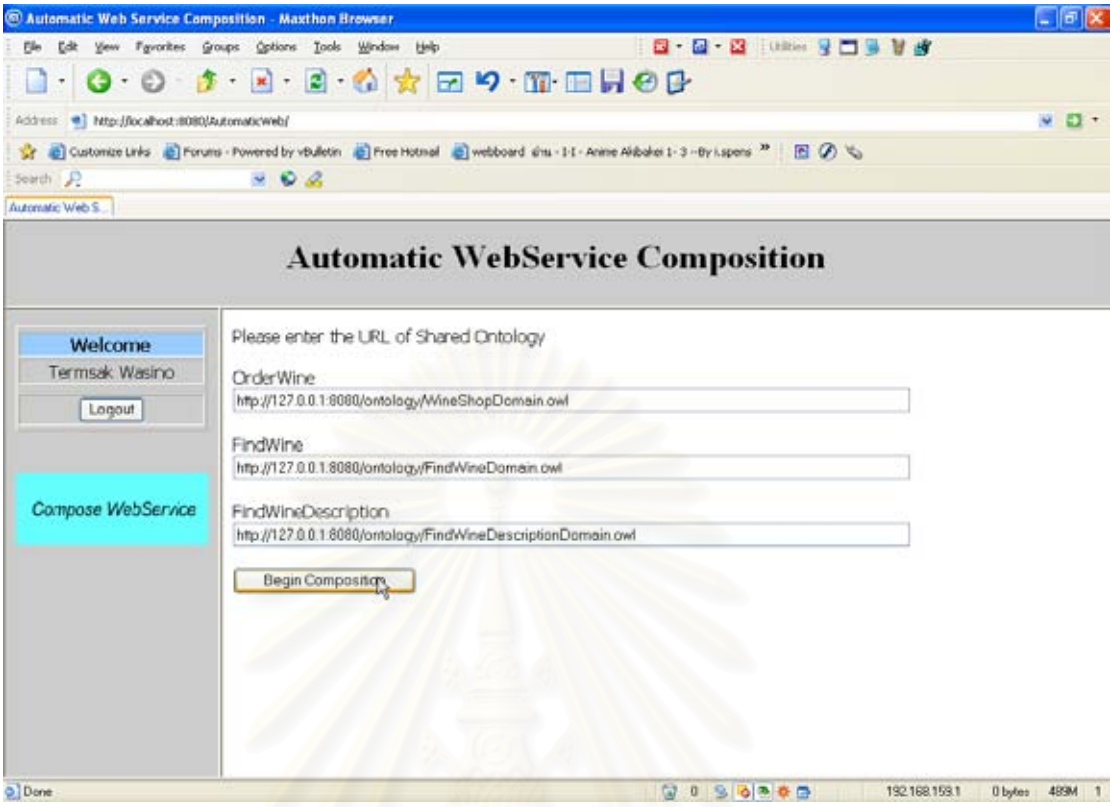


รูปที่ 5.6 หน้าเว็บหลังจากการล็อกอินเพื่อเข้าใช้บริการจากตัวกลาง

จากนั้นจึงทำการอัพโหลดไฟล์อาวล์-เอส จากการสร้างด้วยโปรแกรมโพรทีเจ และ ไฟล์ที่ทำการกำหนดข้อกำหนดกระบวนการต่างๆ ขึ้นไปยังตัวกลางพร้อมกับกำหนดค่าน้ำหนัก **λ** ในช่องการเข้าคู่ของข้อมูลขาเข้าและข้อมูลขาออกระหว่างเว็บเซอร์วิส จากนั้นกำหนดชื่อเอกสารของผลลัพธ์ ดังรูปที่ 5.7 เมื่อคลิกปุ่มสั่งให้เริ่มทำงานขั้นต่อไป ตัวกลางจะทำการอ่านเอกสารที่อัพโหลดขึ้นไปพร้อมกับถามถึงออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน ให้ทำการใส่ยูอาร์แอลของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของแต่ละกระบวนการลงไปดังรูปที่ 5.8 และกดปุ่มเพื่อให้ตัวกลางเริ่มทำการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

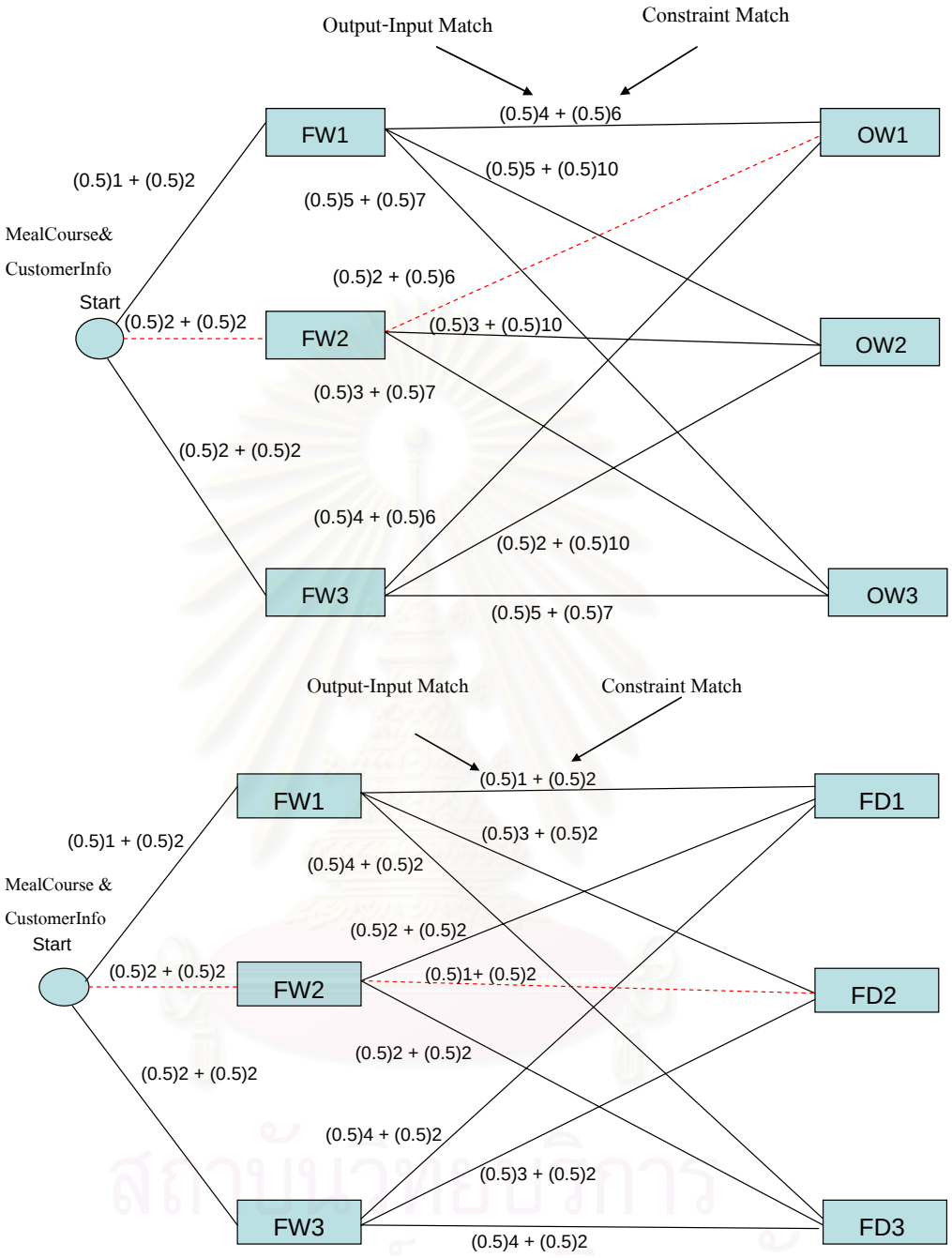


รูปที่ 5.7 การอัพโหลดไฟล์กระแสงานและข้อกำหนดกระบวนการของกรณีศึกษาแรก

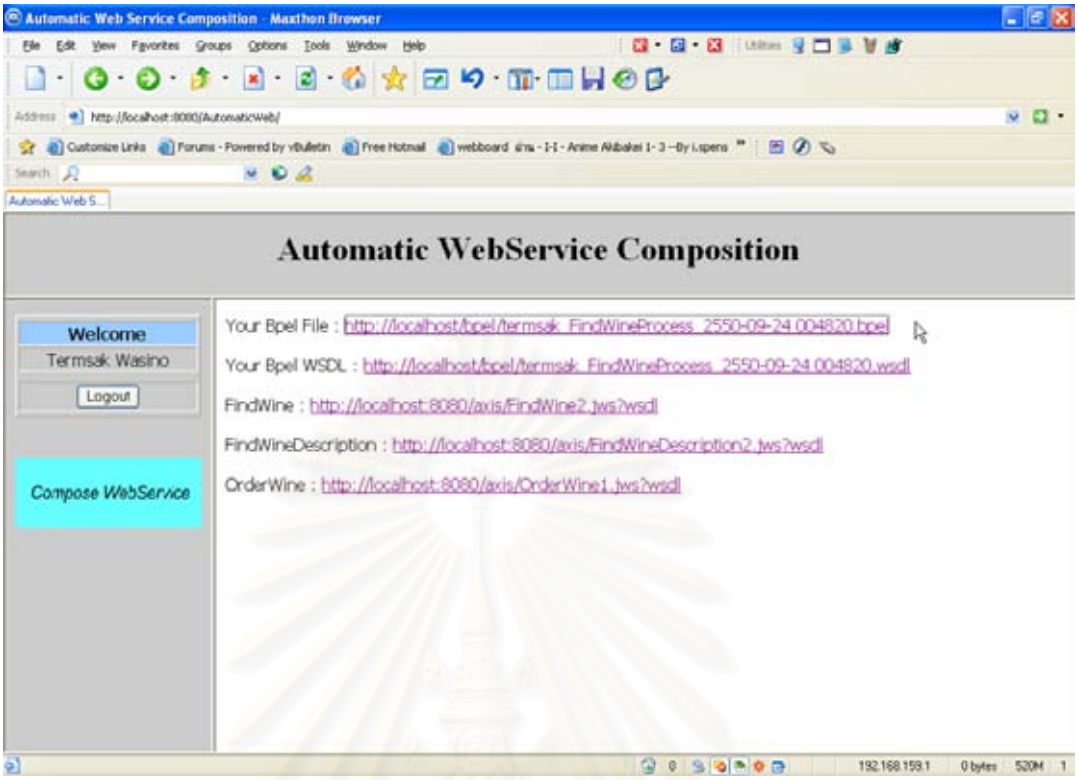


รูปที่ 5.8 การใส่ค่ายูอาร์แอลของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของกรณีศึกษาแรก

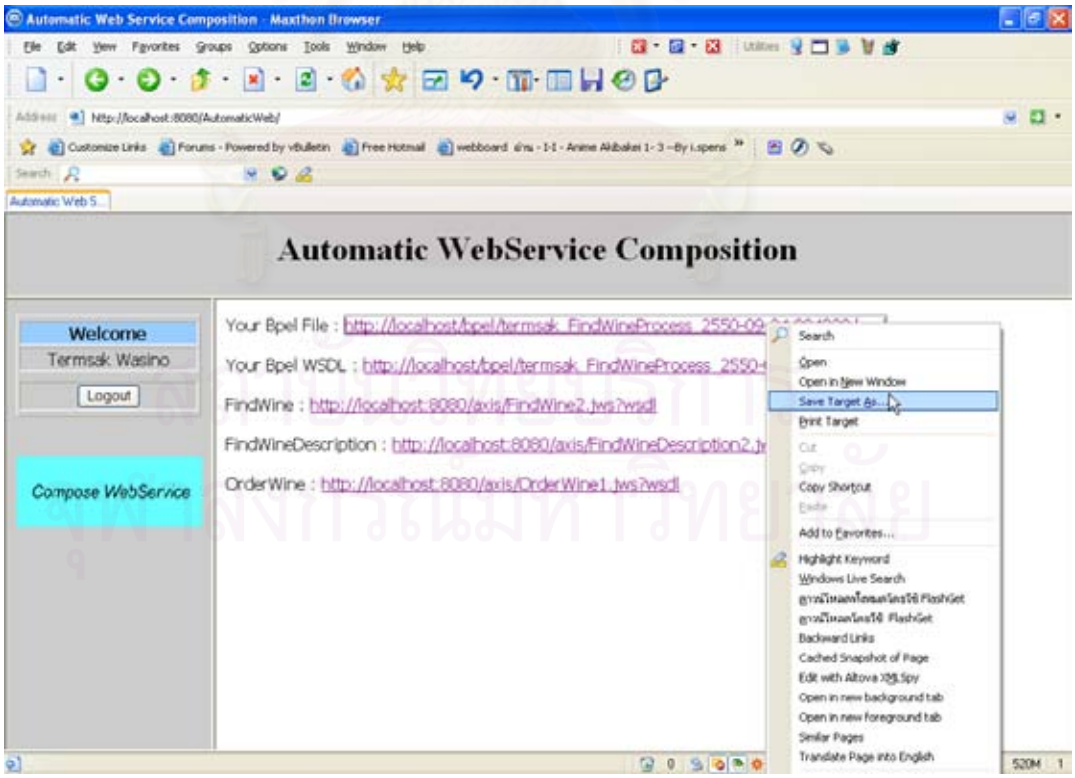
ภายในขั้นตอนการประกอบเว็บเซอร์วิส ตัวกลางจะพิจารณาผลลัพธ์ที่ได้จากการใช้วิธีของไดจ์สตราซึ่งแสดงได้ดังรูปที่ 5.9 และเลือกแผนงานที่ดีที่สุดซึ่งแสดงไว้ด้วยเส้นเชื่อมประ แล้วสร้างไฟล์บีเพลที่มีชื่อตามชื่อของล็อกอินผู้ใช้รวมกับชื่อที่ผู้ใช้กำหนดขึ้นและวันเวลาที่สร้างไฟล์ขึ้นมาพร้อมทั้งไฟล์ดับเบิลยูเอสดีแอลที่เกี่ยวข้องทั้งหมดดังรูปที่ 5.10 จากนั้นทำการจัดเก็บไฟล์ที่เกี่ยวข้องทั้งหมดไปยังเครื่องของผู้ใช้บริการดังรูปที่ 5.11



รูปที่ 5.9 การใช้วิธีของไดจ์สตราเพื่อหาแผนงานที่ดีที่สุดในกรณีศึกษาแรก

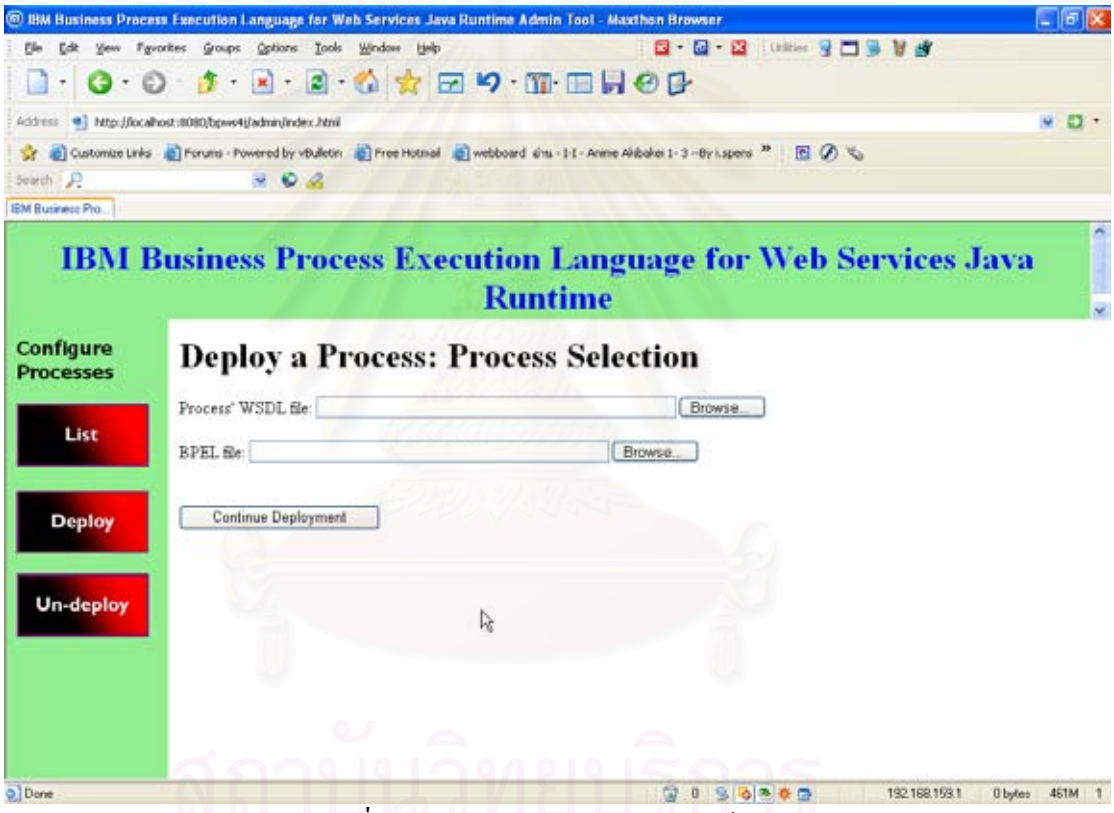


รูปที่ 5.10 ผลลัพธ์ของการประกอบเว็บเซอร์วิสอย่างอัตโนมัติของกรณีศึกษาแรก

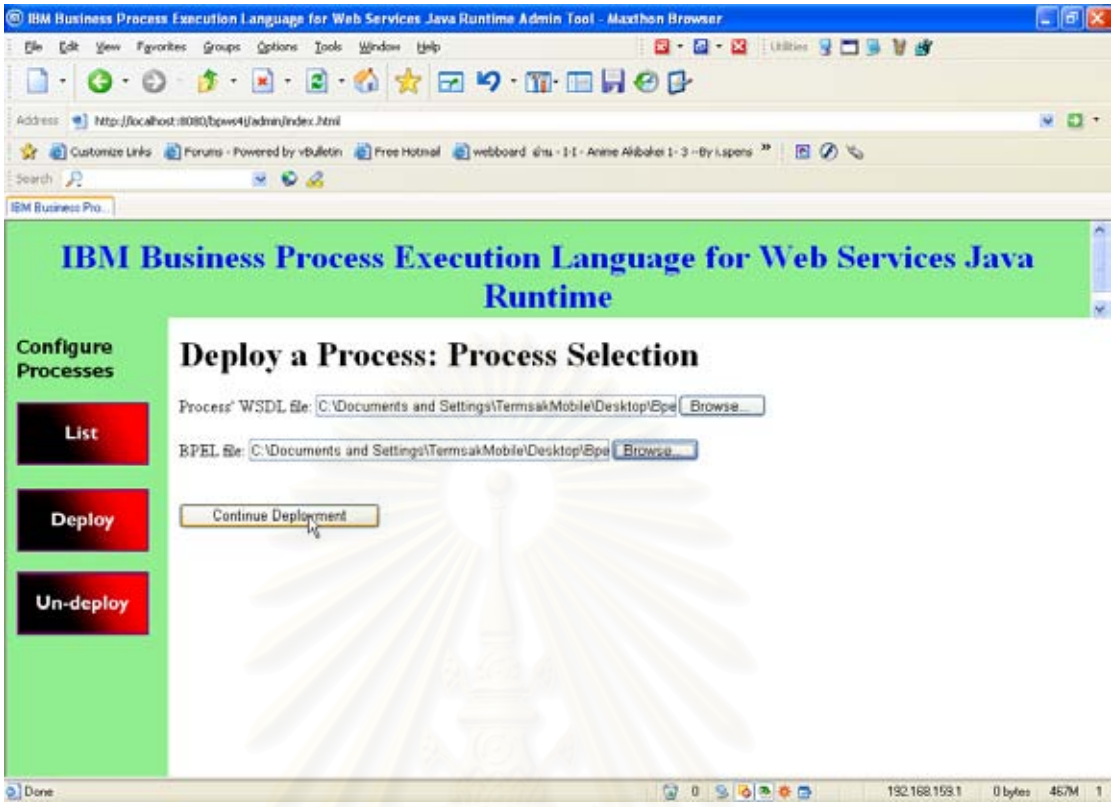


รูปที่ 5.11 การจัดเก็บไฟล์ผลลัพธ์ที่ได้ไปใช้งานของกรณีศึกษาแรก

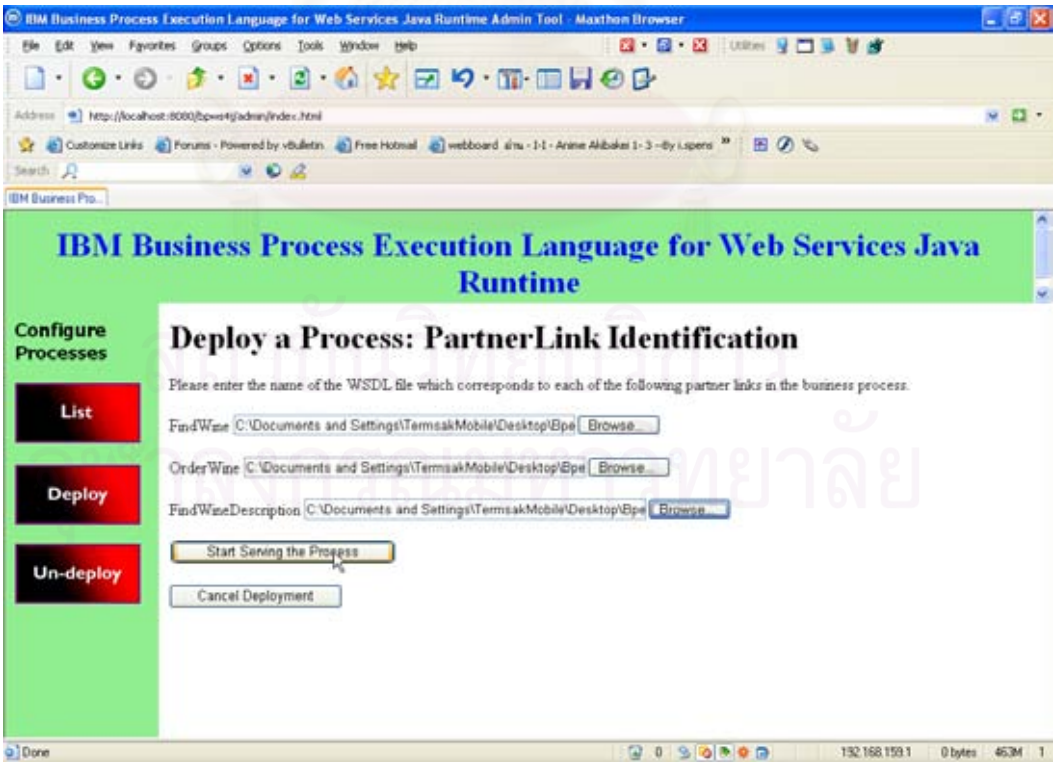
เมื่อทำการจัดเก็บไฟล์ที่เป็นผลลัพธ์ที่ได้ทั้งหมด ผู้ใช้สามารถนำไปใช้ได้โดยการดีพลอยไปยังเครื่องมือที่ใช้ในการทำงานเกี่ยวกับเอกสารบีเพล อาทิเช่น ไอบีเอ็มบีพีดับเบิลยูเอสฟอร์เจ (BPWS4J) รุ่น 2.1 ดังรูปที่ 5.12 จากนั้นทำการอัพโหลดไฟล์บีเพล และ ดับเบิลยูแอสดีแอลสำหรับไฟล์บีเพล ที่ได้จากการทำงานของตัวกลาง ดังแสดงในรูปที่ 5.13 เมื่อคลิกทำงานลำดับถัดไป เครื่องมือจะทำการถามถึงดับเบิลยูเอสดีแอลของกระบวนการที่เกี่ยวข้อง ให้ทำการอัพโหลดดับเบิลยูเอสดีแอลของกระบวนการที่เกี่ยวข้องซึ่งได้มาพร้อมกับผลลัพธ์จากตัวกลางประกอบเว็บเซอร์วิสอย่างอัตโนมัติ ดังรูปที่ 5.14 จากนั้นทำการดีพลอยเซอร์วิส เครื่องมือจะทำการแจ้งรายละเอียดการดีพลอยเพื่อนำไปใช้งานดังรูปที่ 5.15



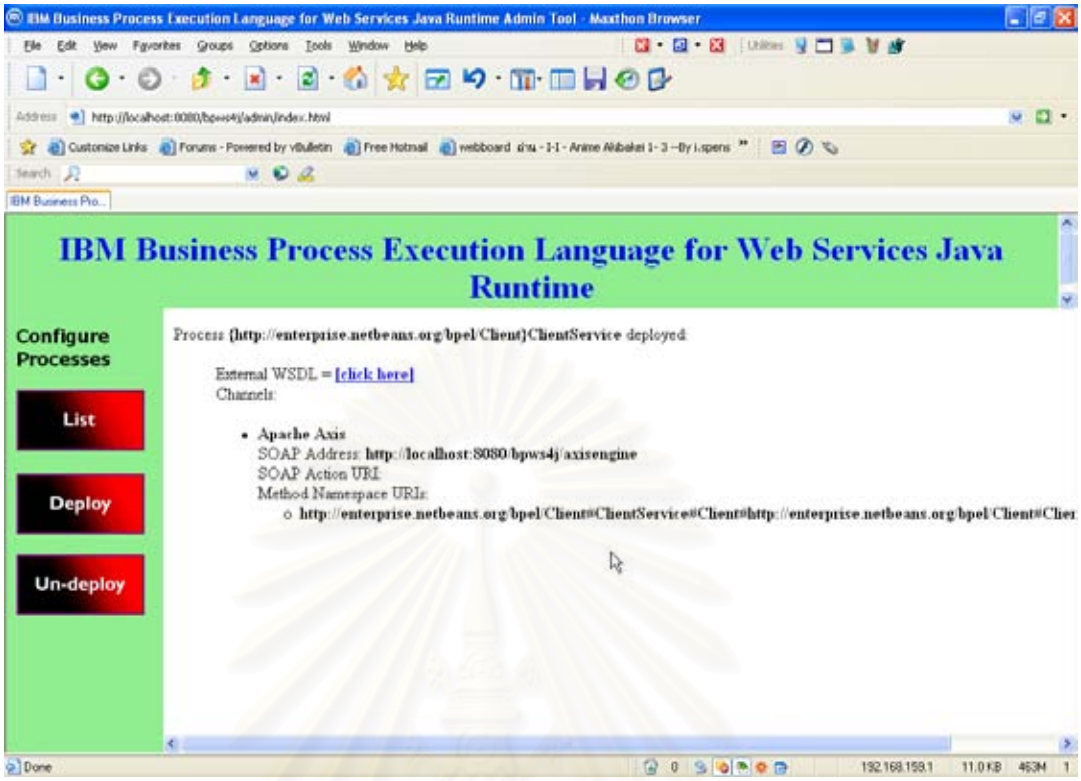
รูปที่ 5.12 หน้าจอรับการดีพลอยไฟล์บีเพล



รูปที่ 5.13 การอัพโหลดไฟล์บีเพลและดับเบิลยูเอสแอลของบีเพล



รูปที่ 5.14 การอัพโหลดไฟล์ดับเบิลยูเอสดีแอลของกระบวนการที่เกี่ยวข้อง



รูปที่ 5.15 รายละเอียดการดีพลอยเพื่อนำไปใช้งาน

5.2 กรณีศึกษาที่มีโครงสร้างกระแสงานแบบลำดับร่วมกับแบบแบ่ง

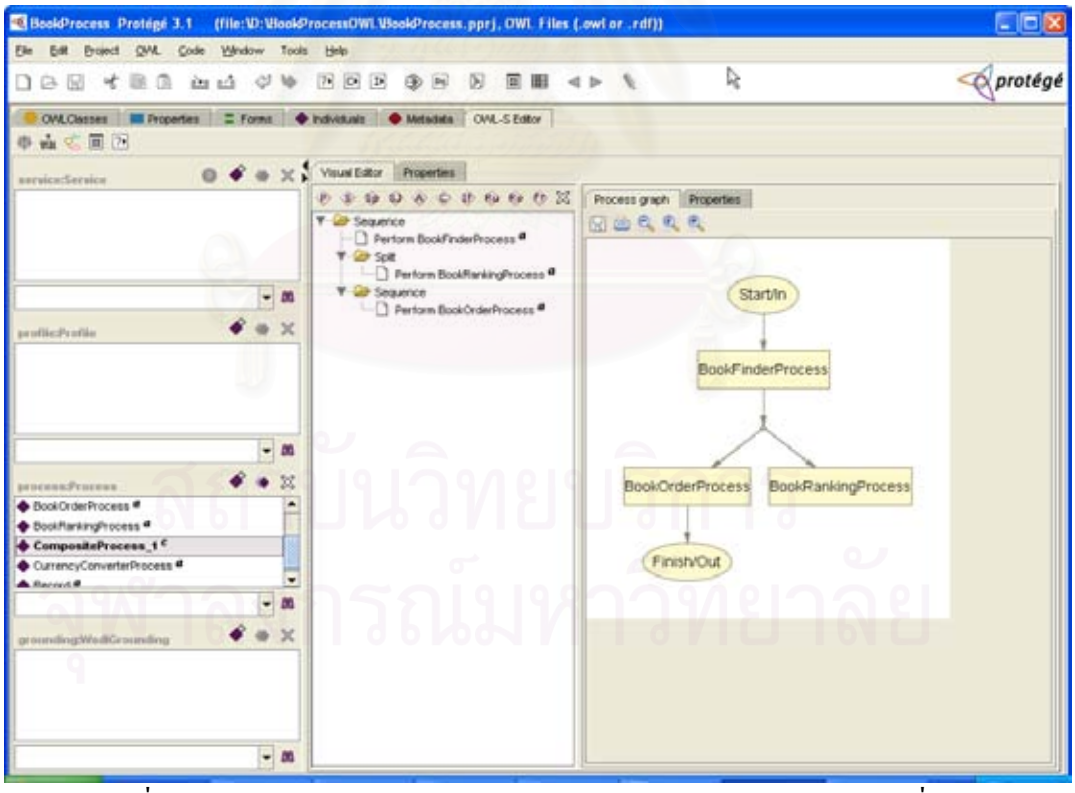
กรณีศึกษานี้จะใช้ตัวอย่างได้แก่ “ผู้ใช้บริการซึ่งเป็นนักออกแบบเว็บเซอร์วิสต้องการประกอบเว็บเซอร์วิสที่มีบริการการสั่งซื้อหนังสือออนไลน์ โดยมีข้อบังคับว่าบริการนี้จะรับบัตรเครดิตของบริษัทเอเม็กซ์ (Amex) และต้องมีบริการส่งถึงบ้านภายใน 3 วัน โดยที่อยู่ของลูกค้าอยู่ในกรุงเทพฯ พร้อมทั้งส่งข้อมูลการสั่งซื้อไปยังเว็บที่ทำการจัดลำดับหนังสือยอดนิยมด้วย” โดยนักออกแบบเว็บเซอร์วิสออกแบบกระแสงานโดยใช้โปรแกรมโพรทีเจ ดังรูปที่ 5.16 และออกแบบข้อกำหนดกระบวนการของแต่ละกระบวนการโดยอาศัยออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของกระบวนการนั้นๆดังในรูปที่ 5.17 ซึ่งจะใช้เว็บเซอร์วิสที่มีพฤติกรรมเชิงหน้าที่ต่างๆกันในการทดลอง ดังตารางที่ 5.2 และออนโทโลจีที่ใช้ในการทดลองดังรูป ที่ 5.18

ตารางที่ 5.2 พฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่ใช้ในกรณีศึกษาที่สอง

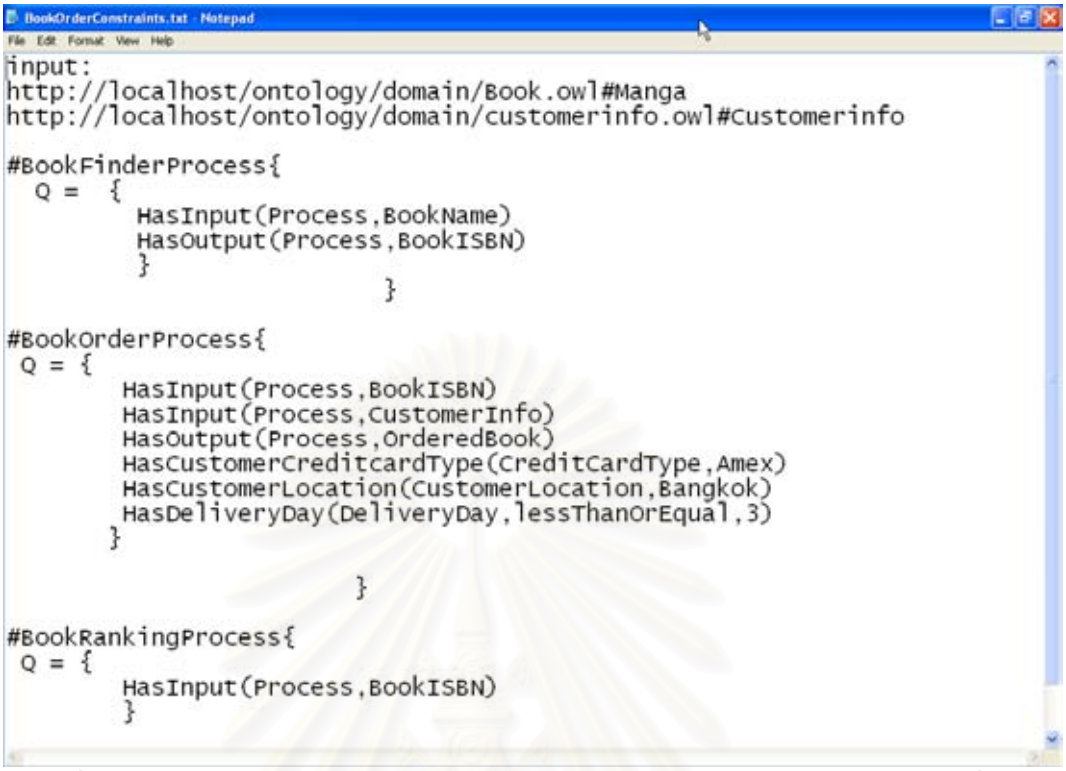
ชื่อเว็บเซอร์วิส	ลักษณะ
BookFinder1	Input = Manga Output = BookISBN
BookFinder2	Input = Teens Output = ISBN
BookFinder3	Input = School Output = ISBN
BookOrder1	Input = BookISBN Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=3 Town = Bangkok CreditType = Amex
BookOrder2	Input = ISBNBook Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=5 Town = Bangkok CreditType = Visa
BookOrder3	Input = ISBN Input = CustomerInfo Output = PurchaseOrderConfirmation DeliveryDay <=4 Town = Bangkok CreditType = Amex,Visa

ตารางที่ 5.2 พฤติกรรมเชิงหน้าที่ของเว็บเซอร์วิสที่ใช้ในกรณีศึกษาที่สอง (ต่อ)

BookRanking1	Input = BookISBN Output = int
BookRanking2	Input = ISBN Output = int
BookRanking3	Input = ISBNBook Output = int



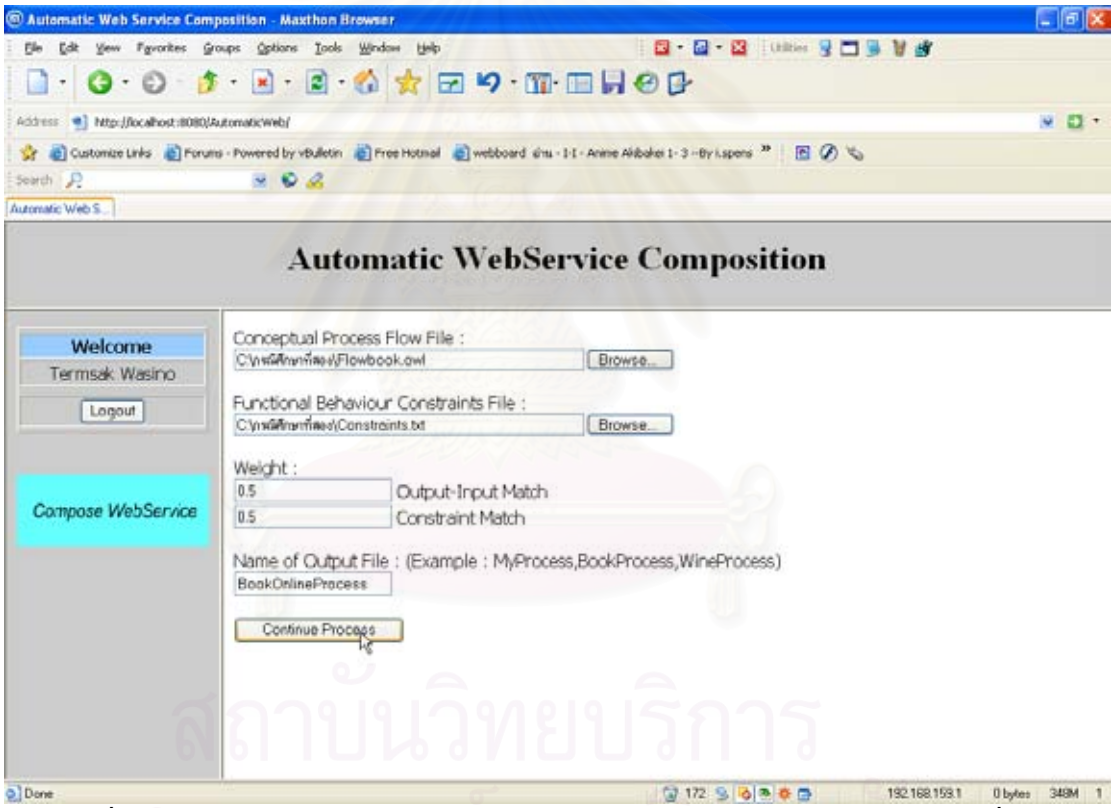
รูปที่ 5.16 การออกแบบกระแสงานโดยโปรแกรมโพรทีเจของกรณีศึกษาที่สอง



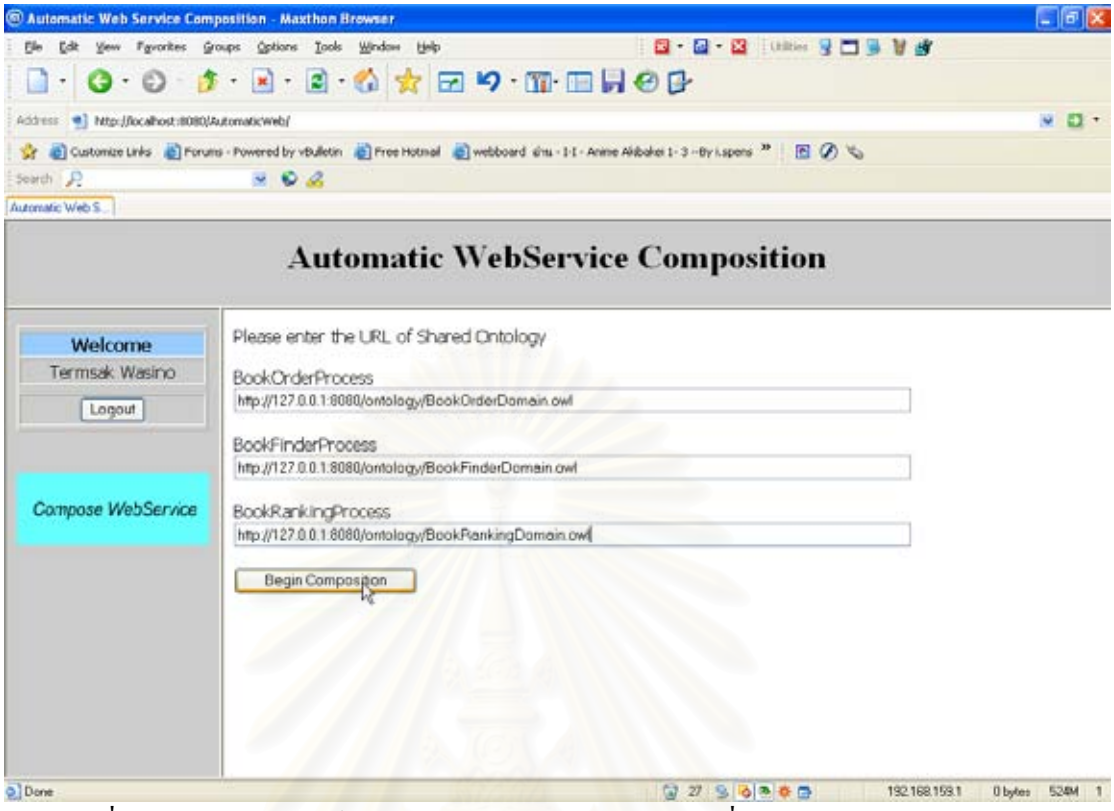
รูปที่ 5.17 การกำหนดข้อกำหนดกระบวนการของแต่ละกระบวนการของกรณีศึกษาที่สอง

รูปที่ 5.18 ออนโทโลจีที่ใช้ในกรณีศึกษาที่สอง

หลังจากจัดเตรียมไฟล์อาวล์-เอสจากการสร้างด้วยโปรแกรมโพรทีเจ และ ไฟล์ที่ทำการกำหนดข้อกำหนดกระบวนการต่างๆแล้ว ให้เข้าไปยังเว็บไซต์ของตัวกลางดังรูปที่ 5.4 จากนั้นทำการสมัครสมาชิกเพื่อขอใช้บริการ ดังรูปที่ 5.5 และทำการล็อกอิน เพื่อเข้าไปใช้งานดังรูปที่ 5.6 จากนั้นจึงทำการอัพโหลดไฟล์อาวล์-เอส จากการสร้างด้วยโปรแกรมโพรทีเจ และ ไฟล์ที่ทำการกำหนดข้อกำหนดกระบวนการต่างๆ ขึ้นไปยังตัวกลางพร้อมกับกำหนดค่าน้ำหนัก λ ดังรูปที่ 5.19 เมื่อคลิกปุ่มสั่งให้เริ่มทำงานขั้นต่อไป ตัวกลางจะทำการอ่านไฟล์ที่อัพโหลดขึ้นไปพร้อมกับถามถึงออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมน ให้ทำการใส่ยูอาร์แอลของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของแต่ละกระบวนการลงไปดังรูปที่ 5.20 และกดปุ่มเพื่อให้ตัวกลางเริ่มทำการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ

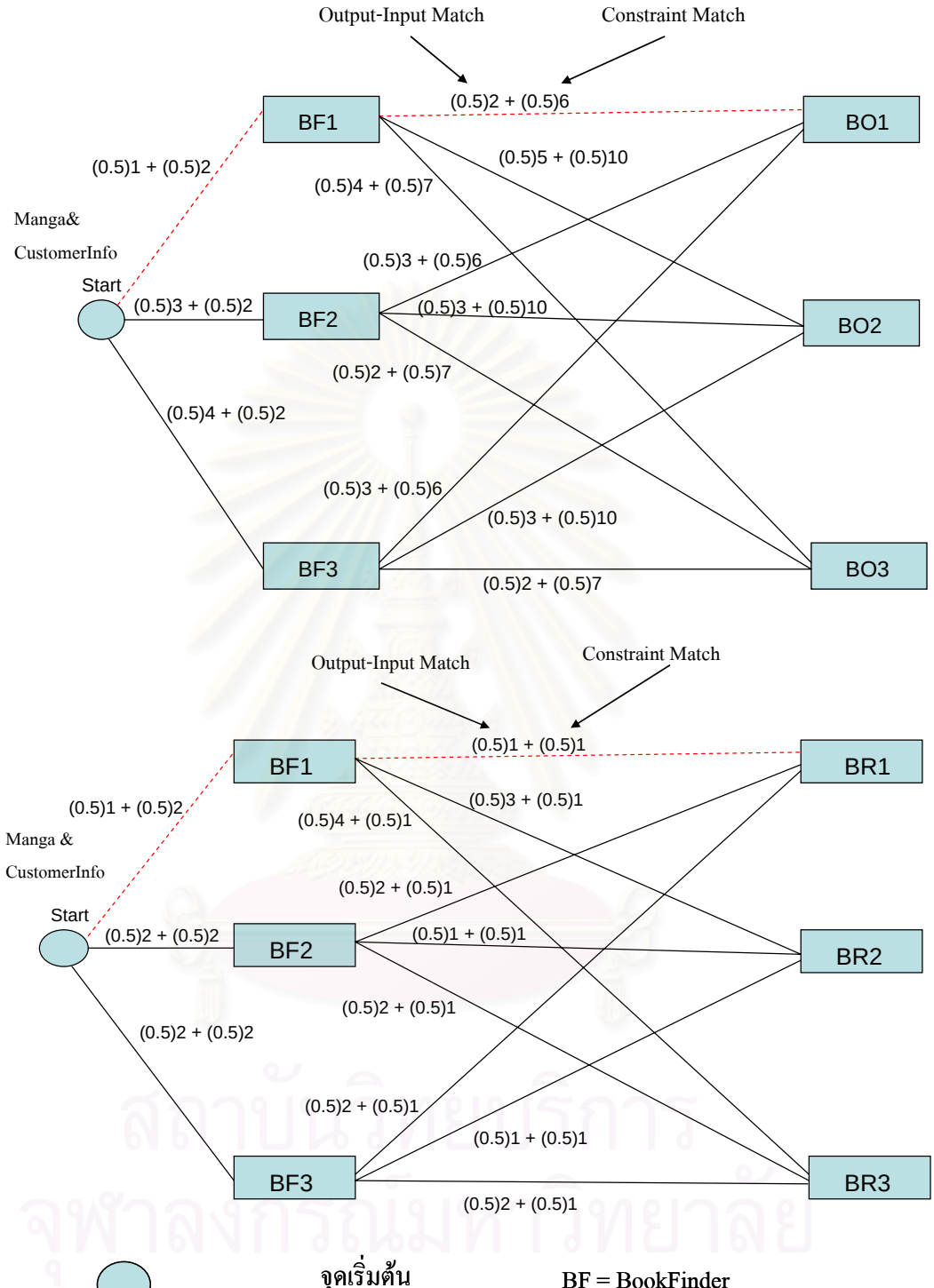


รูปที่ 5.19 การอัพโหลดไฟล์กระแสงานและข้อกำหนดกระบวนการของกรณีศึกษาที่สอง

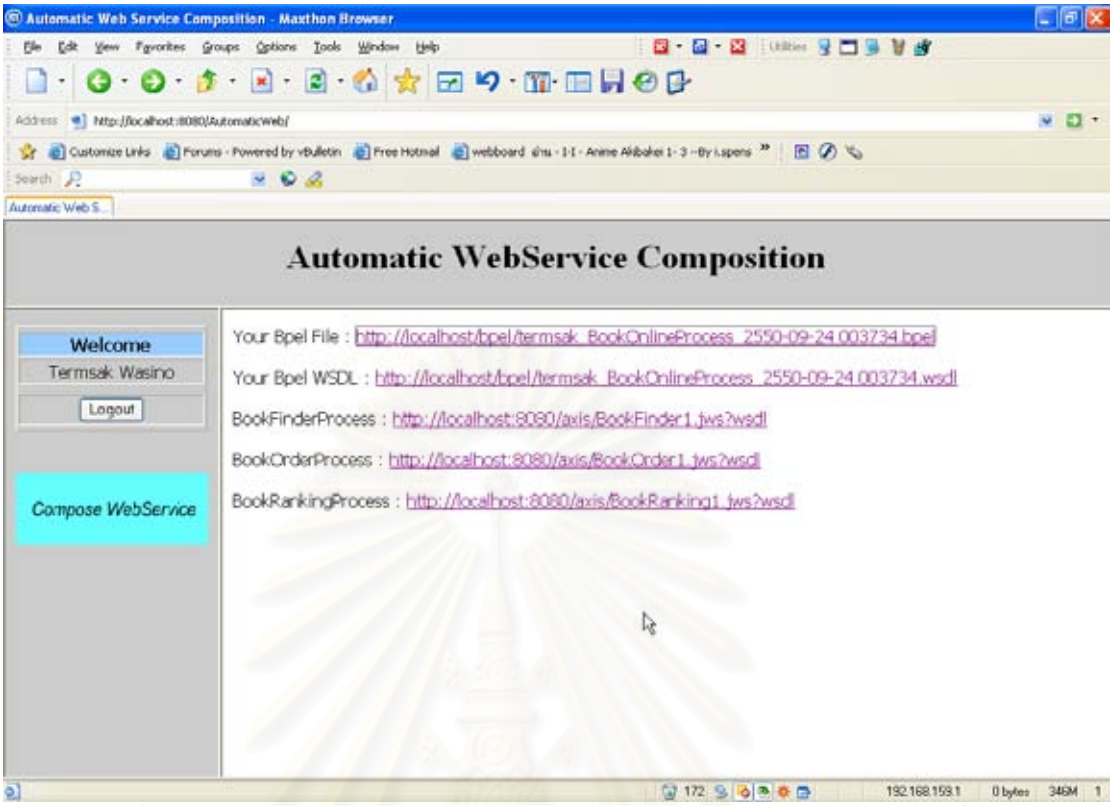


รูปที่ 5.20 การใส่ค่ายูอาร์แอลของออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนของกรณีศึกษาที่สอง

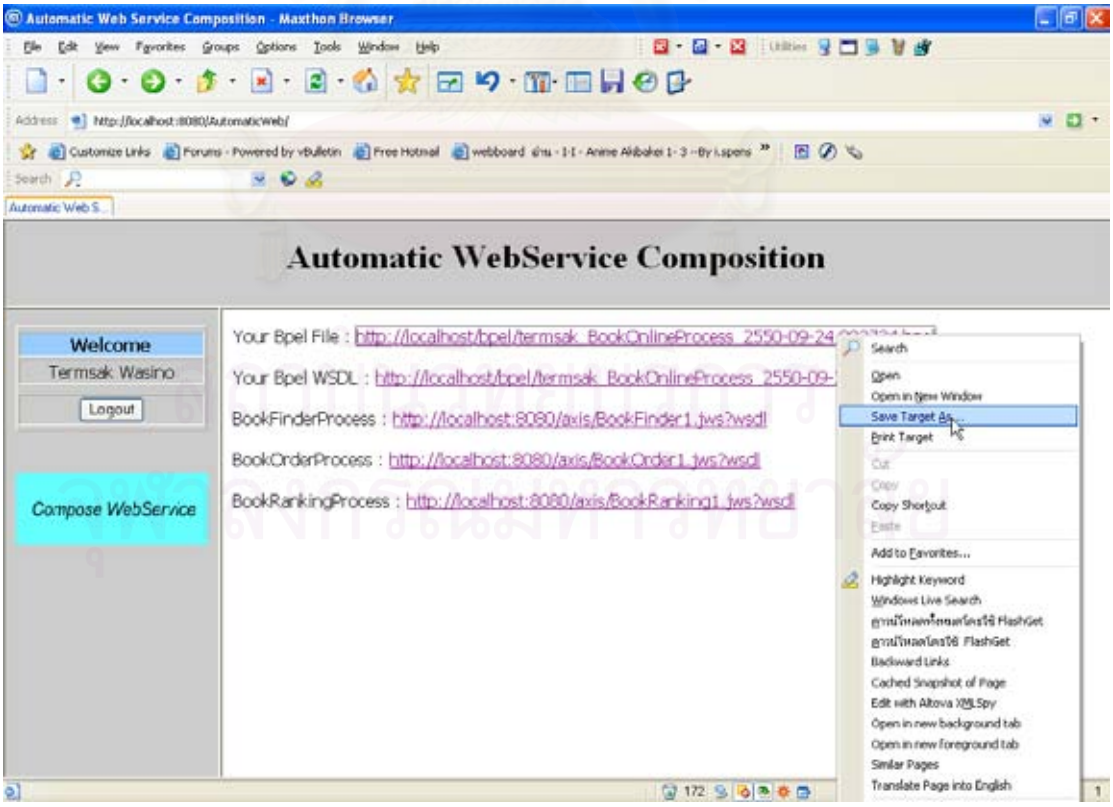
ภายในขั้นตอนการประกอบเว็บเซอร์วิส ตัวกลางจะพิจารณาผลลัพธ์ที่ได้จากการใช้วิธีของไดจ์สตราซึ่งแสดงได้ดังรูปที่ 5.21 และเลือกแผนงานที่ดีที่สุดซึ่งแสดงไว้ด้วยเส้นเชื่อมประ แล้วสร้างไฟล์บีเพลที่มีชื่อตามชื่อของล็อกอินผู้ใช้รวมกับชื่อที่ผู้ใช้กำหนดขึ้นและวันเวลาที่สร้างไฟล์ขึ้นมาพร้อมทั้งไฟล์ดับเบิลยูเอสดีแอลที่เกี่ยวข้องทั้งหมดดังรูปที่ 5.22 จากนั้นทำการจัดเก็บไฟล์ที่เกี่ยวข้องทั้งหมดไปยังเครื่องของผู้ใช้บริการดังรูปที่ 5.23



รูปที่ 5.21 การใช้วิธีของไดจ์สตราเพื่อหาแผนงานที่ดีที่สุดในกรณีศึกษาที่สอง



รูปที่ 5.22 ผลลัพธ์ของการประกอบเว็บเซอร์วิสอย่างอัตโนมัติของกรณีศึกษาที่สอง

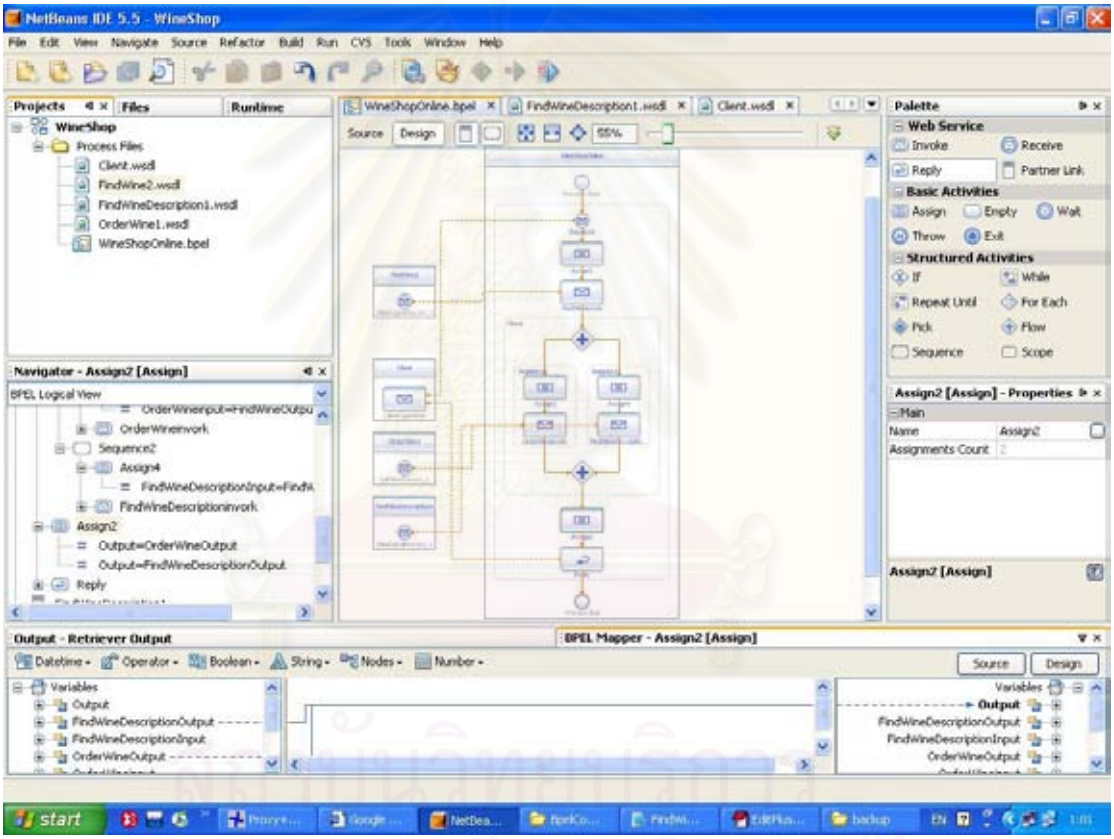


รูปที่ 5.23 การจัดเก็บไฟล์ผลลัพธ์ที่ได้ไปใช้งานของกรณีศึกษาที่สอง

เมื่อทำการจัดเก็บไฟล์ที่เป็นผลลัพธ์ที่ได้ทั้งหมด ผู้ใช้สามารถนำไปใช้ได้โดยการดีพลอยไปยังเครื่องมือที่ใช้ในการทำงานเกี่ยวกับเอกสารบีเพล ดังเช่นกรณีศึกษาแรก

5.3 การเปรียบเทียบผลลัพธ์ที่ได้จากตัวกลางกับการประกอบเว็บเซอร์วิสตามปกติ

รูปที่ 5.24 แสดงการใช้โปรแกรมเน็ตบีนส์ทำการประกอบเว็บเซอร์วิสด้วยวิธีปกติ รูปที่ 5.25 แสดงไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสตามปกติด้วยโปรแกรมเน็ตบีนส์ และ รูปที่ 5.26 แสดงไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสจากตัวกลาง



รูปที่ 5.24 การใช้โปรแกรมเน็ตบีนส์ประกอบเว็บเซอร์วิสด้วยวิธีปกติ

```
<?xml version="1.0" encoding="UTF-8"?>
<process
    name="WineShopOnline"
    targetNamespace="http://enterprise.netbeans.org/bpel/WineShopOnline"
    xmlns="http://schemas.xmlsoap.org/ws/2004/03/business-process/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:bpws="http://schemas.xmlsoap.org/ws/2004/03/business-process/"
    xmlns:wsdlNS="http://enterprise.netbeans.org/bpel/WineShopOnline" xmlns:ns1="http://enterprise.netbeans.org/bpel/Client"
xmlns:ns2="http://localhost:8080/axis/FindWine2.jws" xmlns:ns3="http://localhost:8080/axis/OrderWine1.jws"
xmlns:ns4="http://localhost:8080/axis/FindWineDescription1.jws">
   <import namespace="http://enterprise.netbeans.org/bpel/Client" location="Client.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://localhost:8080/axis/FindWine2.jws" location="FindWine2.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://localhost:8080/axis/OrderWine1.jws" location="OrderWine1.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://localhost:8080/axis/FindWineDescription1.jws" location="FindWineDescription1.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <partnerLinks>
     <partnerLink name="FindWineDescription" partnerLinkType="ns4:FindWineDescription1LinkType"
partnerRole="FindWineDescription1Role"/>
     <partnerLink name="OrderWine" partnerLinkType="ns3:OrderWine1LinkType" partnerRole="OrderWine1Role"/>
     <partnerLink name="FindWine" partnerLinkType="ns2:FindWine2LinkType" partnerRole="FindWine2Role"/>
     <partnerLink name="Client" partnerLinkType="ns1:ClientPartner" myRole="ClientPortTypeRole"/>
   </partnerLinks>
   <variables>
     <variable name="Output" messageType="ns1:ClientOperationReply"/>
     <variable name="FindWineDescriptionOutput" messageType="ns4:WineDetailProcess_1Response"/>
     <variable name="FindWineDescriptionInput" messageType="ns4:WineDetailProcess_1Request"/>
     <variable name="OrderWineOutput" messageType="ns3:SellWineProcess_1Response"/>
     <variable name="OrderWineinput" messageType="ns3:SellWineProcess_1Request"/>
     <variable name="FindWineOutput" messageType="ns2:WineAgentProcess_1Response"/>
     <variable name="FindWineInput" messageType="ns2:WineAgentProcess_1Request"/>
     <variable name="Input" messageType="ns1:ClientOperationRequest"/>
   </variables>
   <sequence>
     <receive name="Receive1" createInstance="yes" partnerLink="Client" operation="ClientOperation" portType="ns1:ClientPortType"
variable="Input"/>
     <assign name="Assign1">
       <copy>
         <from variable="Input" part="in0"/>
         <to variable="FindWineInput" part="in0"/>
       </copy>
     </assign>
     <invoke name="FindWineinvork" partnerLink="FindWine" operation="WineAgentProcess_1" portType="ns2:FindWine2"
inputVariable="FindWineInput" outputVariable="FindWineOutput"/>
     <flow name="Flow1">
       <sequence name="Sequence1">
         <assign name="Assign3">
           <copy>
             <from variable="FindWineOutput" part="WineAgentProcess_1Return"/>
             <to variable="OrderWineinput" part="in0"/>
           </copy>
         </assign>
         <invoke name="OrderWineinvork" partnerLink="OrderWine" operation="SellWineProcess_1" portType="ns3:OrderWine1"
inputVariable="OrderWineinput" outputVariable="OrderWineOutput"/>
       </sequence>
       <sequence name="Sequence2">
         <assign name="Assign4">
           <copy>
             <from variable="FindWineOutput" part="WineAgentProcess_1Return"/>
             <to variable="FindWineDescriptionInput" part="in0"/>
           </copy>
         </assign>
         <invoke name="FindWineDescriptioninvork" partnerLink="FindWineDescription" operation="WineDetailProcess_1"
portType="ns4:FindWineDescription1" inputVariable="FindWineDescriptionInput" outputVariable="FindWineDescriptionOutput"/>
       </sequence>
     </flow>
```

ส่วนประกาศเนมสเปซ
และการนำเข้าไฟล์

รูปที่ 5.25 ไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสตามปกติด้วยโปรแกรมเน็ตบีนส์

```
<assign name="Assign2">
        <copy>
          <from variable="OrderWineOutput" part="SellWineProcess_1Return"/>
          <to variable="Output" part="SellWineProcess_1Return"/>
        </copy>
        <copy>
          <from variable="FindWineDescriptionOutput" part="WineDetailProcess_1Return"/>
          <to variable="Output" part="WineDetailProcess_1Return"/>
        </copy>
      </assign>
      <reply name="Reply" partnerLink="Client" operation="ClientOperation" portType="ns1:ClientPortType" variable="Output"/>
    </sequence>
    </process>
```

รูปที่ 5.25 ไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสตามปกติด้วยโปรแกรมเน็ตบีนส์ (ต่อ)

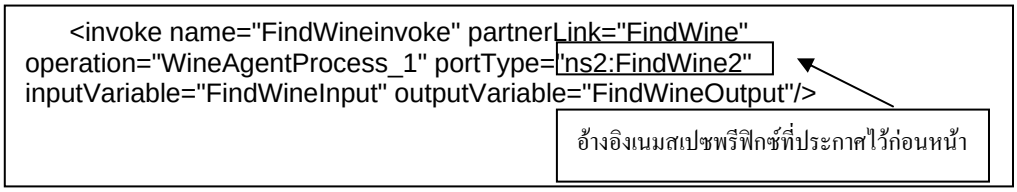
```
<process xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        name="BpelProcess"
        targetNamespace="http://xmlns.oracle.com/Bpel"
        xmlns:tns="http://xmlns.oracle.com/Bpel">
<partnerLinks>
    <partnerLink name="Client" xmlns:ns1="http://enterprise.netbeans.org/bpel/Client" partnerLinkType="ns1:ClientPartner"
myRole="ClientPortTypeRole"/>
    <partnerLink name="FindWine" xmlns:ns2="http://localhost:8080/axis/FindWine2.jws" partnerLinkType="ns2:FindWineLinkType"
partnerRole="FindWineRole"/>
    <partnerLink name="FindWineDescription" xmlns:ns3="http://localhost:8080/axis/FindWineDescription1.jws"
partnerLinkType="ns3:FindWineDescriptionLinkType" partnerRole="FindWineDescriptionRole"/>
    <partnerLink name="OrderWine" xmlns:ns4="http://localhost:8080/axis/OrderWine1.jws" partnerLinkType="ns4:OrderWineLinkType"
partnerRole="OrderWineRole"/>
  </partnerLinks>
  <variables>
    <variable name="Input" xmlns:ns5="http://enterprise.netbeans.org/bpel/Client" messageType="ns5:ClientOperationRequest"/>
    <variable name="Output" xmlns:ns6="http://enterprise.netbeans.org/bpel/Client" messageType="ns6:ClientOperationReply"/>
    <variable name="FindWineinput" xmlns:ns7="http://localhost:8080/axis/FindWine2.jws"
messageType="ns7:WineAgentProcess_1Request"/>
    <variable name="FindWineoutput" xmlns:ns8="http://localhost:8080/axis/FindWine2.jws"
messageType="ns8:WineAgentProcess_1Response"/>
    <variable name="FindWineDescriptioninput" xmlns:ns9="http://localhost:8080/axis/FindWineDescription1.jws"
messageType="ns9:WineDetailProcess_1Request"/>
    <variable name="FindWineDescriptionoutput" xmlns:ns10="http://localhost:8080/axis/FindWineDescription1.jws"
messageType="ns10:WineDetailProcess_1Response"/>
    <variable name="OrderWineinput" xmlns:ns11="http://localhost:8080/axis/OrderWine1.jws"
messageType="ns11:SellWineProcess_1Request"/>
    <variable name="OrderWineoutput" xmlns:ns12="http://localhost:8080/axis/OrderWine1.jws"
messageType="ns12:SellWineProcess_1Response"/>
  </variables>
 <sequence>
    <receive name="Receive"
           partnerLink="Client" xmlns:ns13="http://enterprise.netbeans.org/bpel/Client" portType="ns13:ClientPortType"
operation="ClientOperation"
          variable="Input" createInstance="yes">
    </receive>
    <assign >
      <copy>
        <from variable="Input" part="in0"/>
        <to variable="FindWineinput" part="in0"/>
      </copy>
    </assign>
    <invoke name="FindWineinvork"
          partnerLink="FindWine" xmlns:ns14="http://localhost:8080/axis/FindWine2.jws" portType="ns14:FindWine2"
operation="WineAgentProcess_1"
          inputVariable="FindWineinput" outputVariable="FindWineoutput">
    </invoke>
    <flow>
      <sequence>
        <assign >
          <copy>
            <from variable="FindWineoutput" part="WineAgentProcess_1Return"/>
            <to variable="FindWineDescriptioninput" part="in0"/>
          </copy>
        </assign>
```

รูปที่ 5.26 ไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสด้วยตัวกลาง

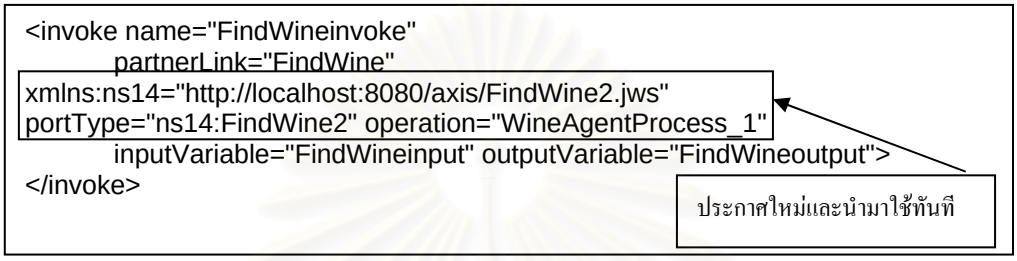
```
<invoke name="FindWineDescriptioninvork"
            partnerLink="FindWineDescription" xmlns:ns15="http://localhost:8080/axis/FindWineDescription1.jws"
portType="ns15:FindWineDescription1" operation="WineDetailProcess_1"
            inputVariable="FindWineDescriptioninput" outputVariable="FindWineDescriptionoutput">
        </invoke>
      </sequence>
      <sequence>
        <assign >
          <copy>
            <from variable="FindWineoutput" part="WineAgentProcess_1Return"/>
            <to variable="OrderWineinput" part="in0"/>
          </copy>
        </assign>
        <invoke name="OrderWineinvork"
            partnerLink="OrderWine" xmlns:ns16="http://localhost:8080/axis/OrderWine1.jws" portType="ns16:OrderWine1"
operation="SellWineProcess_1"
            inputVariable="OrderWineinput" outputVariable="OrderWineoutput">
        </invoke>
      </sequence>
    </flow>
    <assign >
      <copy>
        <from variable="OrderWineoutput" part="SellWineProcess_1Return"/>
        <to variable="Output" part="SellWineProcess_1Return"/>
      </copy>
    </assign>
    <assign >
      <copy>
        <from variable="FindWineDescriptionoutput" part="WineDetailProcess_1Return"/>
        <to variable="Output" part="WineDetailProcess_1Return"/>
      </copy>
    </assign>
    <reply name="Reply"
          partnerLink="Client" xmlns:ns17="http://enterprise.netbeans.org/bpel/Client" portType="ns17:ClientPortType"
operation="ClientOperation"
          variable="Output">
    </reply>
  </sequence>
</process>
```

รูปที่ 5.26 ไฟล์บีเพลของกรณีศึกษาแรกที่ได้จากการประกอบเว็บเซอร์วิสด้วยตัวกลาง (ต่อ)

ไฟล์เอกสารบีเพลที่ได้จากการประกอบแบบปกติจะมีการประกาศเนมสเปซแอตทริบิวต์ (Namespace Attribute) และกำหนดพรีฟิกซ์ (Prefix) ในการอ้างอิงถึงเนมสเปซยูอาร์แอล (Namespace URL) ของพาร์ตเนอร์ลิงค์ (PartnerLink) ก่อนเพียงครั้งเดียวโดยประกาศไว้ภายในรูตแท็ก (Root Tag) ของเอกสารและทุกครั้งที่มีการเรียกใช้จะอ้างอิงพรีฟิกซ์ที่ประกาศไว้ และมีการประกาศนำเข้าไฟล์ดับเบิลยูเอสดีแอลของเว็บเซอร์วิสต่างๆในโปรเจคการทำงานของเน็ตบีนส์สำหรับการสร้างไฟล์เอกสารบีเพล แต่ไฟล์เอกสารบีเพลที่ได้จากตัวกลางจะไม่มีการประกาศเนมสเปซแอตทริบิวต์ไว้ที่รูตแท็กและไม่มีการประกาศนำเข้าไฟล์ดับเบิลยูเอสดีแอลของเว็บเซอร์วิสต่างๆแต่จะอ้างอิงเป็นยูอาร์แอลแทน และจะประกาศเนมสเปซภายในแต่ละแท็กที่มีการอ้างอิงโดยเปลี่ยนพรีฟิกซ์ไปเรื่อยๆจึงต่างจากการประกอบแบบปกติ แต่การทำงานจะให้ผลเหมือนกัน ยกตัวอย่างเช่น ในการร้องขอบริการ (Invoke) เซอร์วิสค้นหาไวน์ของไฟล์เอกสารบีเพลที่ทำการประกอบปกติ ดังรูปที่ 5.27 จะมีการอ้างอิงพรีฟิกซ์ที่ทำการประกาศไว้ก่อนที่รูตแท็ก แต่การร้องขอบริการเซอร์วิสค้นหาไวน์ของไฟล์เอกสารบีเพลที่ทำการประกอบโดยตัวกลางจะอ้างอิงถึงพรีฟิกซ์ที่ทำการประกาศใหม่ไว้ในแท็กนั้น ดังรูปที่ 5.28



รูปที่ 5.27 แท็กการร้องขอบริการเซอร์วิสค้นหาไวน์ของไฟล์บีเพลที่ทำการประกอบแบบปกติ



รูปที่ 5.28 แท็กการร้องขอบริการเซอร์วิสค้นหาไวน์ของไฟล์บีเพลที่ทำการประกอบโดยตัวกลาง

เพราะฉะนั้นความแตกต่างในระหว่างการประกอบแบบปกติและการใช้ตัวกลางจะมีเพียงเล็กน้อยในด้านการประกาศเนมสเปซและพรีฟิกซ์ แต่ในด้านกระบวนการทำงานภายในจะเหมือนกันและให้ผลการทำงานที่เหมือนกัน แต่ตัวกลางนั้นจะช่วยลดขั้นตอนการค้นหาเว็บเซอร์วิสที่นำมาใช้ในการประกอบเว็บเซอร์วิสและระยะเวลาในการประกอบเว็บเซอร์วิส พร้อมทั้งยังสามารถพิจารณาในเชิงความหมายได้อีกด้วย

บทที่ 6
สรุปผลการวิจัย

6.1 สรุปผลการวิจัย

ในงานวิจัยนี้ได้นำเสนอการประกอบเว็บเซอร์วิสอย่างอัตโนมัติโดยอาศัยข้อกำหนดกระบวนการในรูปของอาวล์-เอสโพรเซสโมเดลที่มีข้อบังคับด้านพฤติกรรมเชิงหน้าที่ ซึ่งงานวิจัยนี้อธิบายข้อกำหนดกระบวนการด้วยอาวล์-เอสโพรเซสโมเดลเนื่องจากมีการใช้กันอย่างแพร่หลาย และ อธิบายเงื่อนไขกระบวนการด้วยภาษากฎสเวิร์ล รูปแบบกระแสงานที่งานวิจัยนี้สนใจได้แก่ กระแสงานที่มีโครงสร้างเป็นแบบตามลำดับ แบบแบ่ง และแบบแบ่ง-ร่วม

ผู้วิจัยได้พัฒนาตัวกลางเพื่อช่วยในการประกอบเว็บเซอร์วิสอย่างอัตโนมัติ ซึ่งช่วยเพิ่มประสิทธิภาพในการประกอบเว็บเซอร์วิส โดยได้ใช้กรณีศึกษาการประกอบเว็บเซอร์วิสสั่งซื้อไวน์ และ การประกอบเว็บเซอร์วิสสั่งซื้อหนังสือออนไลน์ เพื่อทดสอบตัวกลาง พร้อมทั้งเปรียบเทียบผลที่ได้กับวิธีการประกอบเว็บเซอร์วิสแบบปกติ โดยผลที่ได้นั้นเป็นไปตามความต้องการของผู้ใช้งานที่ต้องการประกอบเว็บเซอร์วิส

6.2 ปัญหาและอุปสรรค

1. ถึงแม้ว่าแฟ้มข้อมูลอาวล์-เอสจะเป็นที่แพร่หลายในงานวิจัยเกี่ยวกับเว็บเซอร์ซิส และมีเครื่องมือตัวแจงอาวล์-เอสให้ใช้ แต่การใช้งานยังมีข้อจำกัดคือตัวแจงอาวล์-เอสยังไม่สามารถอ่านออนโทโลจีที่ซับซ้อนมากๆ ดังนั้นออนโทโลจีที่นำมาใช้จึงต้องไม่ซับซ้อนจนตัวแจงอาวล์-เอสอ่านไม่ได้

2. ภาษากฎสเวิร์ลซึ่งเป็นภาษากฎที่แพร่หลายในงานวิจัยยังไม่มีตัวแจงที่ใช้งานได้ยืดหยุ่นเพียงพอ เนื่องจากตัวแจงภาษากฎสเวิร์ลที่มีอยู่ได้ถูกออกแบบเพื่อการประมวลผลโปรแกรมเชิงตรรกะเท่านั้น ไม่ได้ถูกออกแบบให้ใช้ร่วมกับอาวล์-เอสโพรเซสโมเดล ทำให้ในการพัฒนาต้องทำการสร้างตัวแจงภาษากฎสเวิร์ลขึ้นเอง

3. ตัวอนุมานสเวิร์ลของเครื่องมืออนุมานที่ใช้คือ บอสซาม สามารถแจงได้เพียงบางส่วนของมาตรฐานของภาษาสเวิร์ลเท่านั้น

6.3 แนวทางการวิจัยต่อไป

ประเด็นที่งานวิจัยนี้ยังไม่ได้ศึกษา และสามารถทำการวิจัยเพิ่มเติมได้ในอนาคตได้แก่

1. สามารถพัฒนาตัวกลางของงานวิจัยนี้ให้มีประสิทธิภาพยิ่งขึ้นในเรื่องของเวลาในการประกอบเว็บเซอร์วิสโดยเปลี่ยนแปลงอัลกอริทึมของการค้นหาแผนงานที่ดีที่สุดให้มีประสิทธิภาพมากขึ้น และ ลดภาระการคำนวณเกี่ยวกับข้อกำหนดกระบวนการของแต่ละกระบวนการ โดยออกแบบระบบให้ทำการคำนวณเหล่านี้ไว้ล่วงหน้าและเก็บไว้ในฐานข้อมูลเมื่อถึงเวลาก็นำมาใช้ได้ หรือ เก็บแผนที่เคยมีผู้ใช้บริการก่อนหน้าเคยทำไว้แล้วและตัวงานมีลักษณะคล้ายกัน เพราะปัจจุบันตัวกลางต้องทำการคำนวณใหม่ทุกครั้งที่มีการประกอบเว็บเซอร์วิสใหม่ในแต่ละครั้ง
2. ในการพิจารณาเงื่อนไขการเข้าคู่ของช่วงตัวเลขในกรณีที่เว็บเซอร์วิสที่ค้นพบมีข้อกำหนดเชิงพฤติกรรมเป็นซับเซตของข้อกำหนดเชิงพฤติกรรมที่กำหนด จะถือว่าเว็บเซอร์วิสนั้นเข้ากันได้กับข้อกำหนดของนักออกแบบเว็บเซอร์วิสได้ดีที่สุดโดยเรียกว่าเป็นการเข้าคู่อย่างถูกต้อง ตัวอย่างเช่นกรณีศึกษาแรกในหัวข้อที่ 5.1 นักออกแบบเว็บเซอร์วิสต้องการเว็บเซอร์วิสที่มีระยะเวลาส่งไวน์ภายใน 3 วัน แต่ถ้าเว็บเซอร์วิสที่ค้นพบสามารถส่งไวน์ให้ภายใน 2 วันจะเข้ากรณีเข้าคู่อย่างถูกต้อง การกำหนดการเข้าคู่ในลักษณะซับเซตนี้เป็นการรับรองว่า เงื่อนไขของเว็บเซอร์วิสที่ค้นพบจะทำให้เงื่อนไขตามข้อกำหนดของนักออกแบบเว็บเซอร์วิสเป็นจริงด้วย [15] อย่างไรก็ตามในบางกรณีค่าช่วงตัวเลขแบบซูเปอร์เซตจะให้ผลลัพธ์ที่ดีกว่าแบบเป็นซับเซต ยกตัวอย่างเช่น นักออกเว็บเซอร์วิสต้องการส่วนลดไวน์ 10% แต่เว็บเซอร์วิสที่ค้นพบให้ส่วนลดไวน์ 15% ซึ่งกรณีนี้งานวิจัยจะถือว่าเป็นการเข้าคู่แบบเสียบเข้า แต่ในความเป็นจริงแล้วเว็บเซอร์วิสที่ให้ส่วนลดไวน์ 15% ย่อมเป็นที่ต้องการของนักออกแบบเว็บเซอร์วิสมากกว่าเว็บเซอร์วิสที่ให้ส่วนลดไวน์ที่น้อยกว่า 10% ซึ่งงานวิจัยถือว่าเป็นการเข้าคู่อย่างถูกต้อง ดังนั้นในกรณีนี้การเข้าคู่แบบเสียบเข้าจะเข้ากันได้กับข้อกำหนดของนักออกแบบเว็บเซอร์วิสได้ดีกว่าการเข้าคู่แบบถูกต้อง ลักษณะเช่นนี้จะถือว่าเป็นข้อจำกัดของการพิจารณา จึงควรมีการขยายกฎการพิจารณาให้สามารถรองรับความหมายของเงื่อนไขได้ว่าการเป็นซับเซตหรือซูเปอร์เซตถึงจะเหมาะสมกว่ากัน
3. ตัวกลางยังไม่รองรับโครงสร้างของไฟล์อาวล์-เอสที่เป็นแบบอื่นนอกเหนือจากโครงสร้าง แบบตามลำดับ แบบแบ่ง และแบบแบ่งร่วม จึงควรทำให้ครบคลุมทุกกรณี

4. งานวิจัยนี้สามารถพัฒนาต่อไปให้สมบูรณ์ขึ้นโดยการพิจารณาค่าคุณภาพการให้บริการ (Quality of Service) ร่วมด้วยในการพิจารณาประกอบเว็บเซอร์วิส
5. เนื่องจากออนโทโลจีกระบวนการที่ใช้ร่วมกันภายในโดเมนยังไม่มีมาตรฐานตายตัว จึงควรติดต่อร่วมมือกับกลุ่มนักวิจัยทางด้านออนโทโลจีเพื่อผลักดันให้มีการกำหนดออนโทโลจีมาตรฐานสำหรับโดเมนงานต่างๆเพื่อนำไปใช้ต่อไป

รายการอ้างอิง

[1] Cerami, E. Web Services Essentials. First Edition. O'Reilly, 2002.

[2] Huhns, M. N., Singh, M. P. Service-Oriented Computing: Key Concepts and Principles. IEEE Internet Computing. (January-February 2005): 75-81.

[3] BPEL. Business Process Execution Language for Web Services (Online). (2003). http://www-106.ibm.com/developerworks/webservice/library/ws-bpel

[4] uddi.org. UDDI: Universal Description, Discovery, and Integration of Web Services (online). (2002). http://www.uddi.org.

[5] Murtagh D. Automatic Web Service Composition. Master's Thesis. Department of Computer Science, University of Dublin, 2004.

[6] Burstein, M. et al. Semantic Web Services Architecture. IEEE Internet Computing (2005): 72-81.

[7] OWL-S Coalition. OWL-S 1.1 Release (online). http://www.daml.org/services/owl-s/1.1

[8] Zeng, L.; Benatallah, B.; Dumas, M.; Kalagnanam J. Quality Driven Web Service Composition. Processdings of the 12 th International World Wide Web Conference, Budapest, Hungary (2003): 411-421

[9] Sounsri, S.; Wichadakul, D.; Savamipak, D. A Genetic Algorithm for an Automated Web Service Composition. The 1 st National Conference on Computing and Information Technology, Bangkok, Thailand (2005): 322-328

[10] Aggarwal, R.; Verma, K. Constraint Driven Web Service Composition in METEOR-S. IEEE International Conference on Services Computing, Shanghai, China (2004): 23-30.

[11] Klusch, M.; Gerber, A.; Schmidt, M. Semantic Web Service Composition Planning with OWLS-XPlan. Proceedings of the 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, United States of America (2005): 77-84

[12] Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M. SWRL: A Semantic Web Rule Language combining OWL and RuleML. (online). (2003). http://daml.org/2003/11/swrl/

[13] Cormen, H. T.; Leiserson, E. C.; Rivest, L. R.; Stein, C. Introduction to Algorithms, Second Edition, pp. 595–601. United States of America: MIT Press and McGraw-Hill, 2001.

[14] Zhang, R.; Arpinar, B. I.; Aleman-Meza B. Automatic Composition of Web Service. Proceedings of the International Conference on Web Services, Las Vegas, Nevada, United States of America (2003): 38-41.

[15] Sriharee, N., Senivongse, T. Matchmaking and Ranking of Semantic Web Services Using Integrated Service Profile. International Journal of Metadata, Semantics and Ontologies. 1,2 (2006): 100-118.

[16] Protégé. (online). http://protege.stanford.edu/

[17] Jena Semantic Web Framework: Jena. (online). http://jena.sourceforge.net/index.html

[18] Bossam Rule/Owl Reasoner. (online). http://mknows.etri.re.kr/bossam/

[19] Suwannopas, P.; Senivongse, T. Discovering Semantic Web Service with Process Specifications. International Conference on Distributed Applications and Interoperable Systems, LNCS 4025, Bologna, Italy (2006): 113-127.

[20] BPWS4J. The IBM Business Process Execution Language for Web Services Java Run Time (online). http://www.alphaworks.ibm.com/tech/bpws4j

[21] NetBeans. (online). http://www.netbeans.org/

ภาคผนวก

ภาคผนวก ก

ตัวอย่างอาวล์-เอสโพรเซสโมเดลที่ใช้ในการทดสอบและผลลัพธ์ที่ได้จากตัวกลาง

ในภาคผนวกนี้จะแสดงตัวอย่างอาวล์-เอสโพรเซสโมเดลที่ใช้ในการทดสอบและไฟล์บีเพลผลลัพธ์ที่ได้จากตัวกลาง ในกรณีศึกษานี้จะแสดงเฉพาะอาวล์-เอสโพรเซสโมเดลของบริการแรกของแต่ละกระบวนการเท่านั้น

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rss="http://purl.org/rss/1.0/"
   xmlns:cus="http://www.owl-ontologies.com/Customerinfo.owl#"
   xmlns:fd="http://www.w3.org/TR/2003/PR-owl-guide-20031209/food#"
   xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
   xmlns:bsd="http://www.owl-ontologies.com/WineShopDomain.owl.owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:p1="http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine#"
   xmlns:vin="http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine#"
   xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
   xmlns="http://www.owl-ontologies.com/AlcoholShop.owl#"
   xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
   xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:pi="http://a.com/ontology#"
   xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
   xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
   xmlns:daml=http://www.daml.org/2001/03/daml+oil#
   xmlns:to=http://www.owl-ontologies.com/Town.owl#
   xmlns:cd=http://www.owl-ontologies.com/CreditCard.owl#
   xml:base="http://www.owl-ontologies.com/AlcoholShop.owl">
  <owl:Ontology rdf:about="">
   <owl:imports rdf:resource="http://localhost:8080/ontology/customerinfo.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/Food.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/pip.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/WineShopDomain.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/CreditCard.owl"/>
   <owl:imports rdf:resource="http://localhost:8080/ontology/Town.owl"/>
  </owl:Ontology>

1<bsd:OrderConfirmed rdf:ID="OrderConfirmed_1">
2 <process:hasResultVar>
3    <bsd:DeliveryDay rdf:ID="DeliveryDay_1"/>
4   </process:hasResultVar>
5   <process:hasEffect>
6    <bsd:DeliveryDayCondition rdf:ID="DeliveryDayCondition_1">
7     <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
8     ><swrl:AtomList>
9    <rdf:first>
10       <swrl:BuiltinAtom>
11        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#lessThanOrEqual/>
12         <swrl:arguments>
13          <rdf:List>
```

รูปที่ ก.1 อาวล์-เอสโพรเซสโมเดลของร้านขายไวน์แรก

```
14            <rdf:first rdf:resource="#DeliveryDay_1"/>
15             <rdf:rest>
16               <rdf:List>
17                 <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
18                 >3</rdf:first>
19                 <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
20               </rdf:List>
21             </rdf:rest>
22           </rdf:List>
23         </swrl:arguments>
24       </swrl:BuiltinAtom>
25     </rdf:first>
26    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
27    </swrl:AtomList></expr:expressionBody>
28    </bsd:DeliveryDayCondition>
29   </process:hasEffect>
30   <process:inCondition>
31    <bsd:LocationCondition rdf:ID="LocationCondition_1">
32     <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
33     ><swrl:AtomList>
34     <rdf:first>
35       <swrl:BuiltinAtom>
36        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#equal"/>
37         <swrl:arguments>
38           <rdf:List>
39            <rdf:first rdf:resource="#CustomerLocation_1">
40             <rdf:rest>
41               <rdf:List>
42                 <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
43                <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
44                 > http://www.owl-ontologies.com/Town.owl#Bangkok</rdf:first>
45               </rdf:List>
46             </rdf:rest>
47           </rdf:List>
48         </swrl:arguments>
49       </swrl:BuiltinAtom>
50     </rdf:first>
51     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
52    </swrl:AtomList></expr:expressionBody>
53    </bsd:LocationCondition>
54  </process:inCondition>
55   <process:hasResultVar>
56    <bsd:CustomerLocation rdf:ID="CustomerLocation_1"/>
57   </process:hasResultVar>
58  </bsd:OrderConfirmed>
59  <bsd:AcceptedCreditCard rdf:ID="AcceptedCreditCard_1">
60   <expr:expressionBody rdf:datatype="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"
61   ><swrl:AtomList>
62      <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
63      <rdf:first>
64       <swrl:BuiltinAtom>
65         <swrl:arguments>
66           <rdf:List>
67             <rdf:rest>
68               <rdf:List>
69                 <rdf:rest>
70                   <rdf:List>
71                     <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
72                     >http://www.owl-ontologies.com/CreditCard.owl#Amex</rdf:first>
73                     <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
74                   </rdf:List>
75                 </rdf:rest>
76                 <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
77                 >http://www.owl-ontologies.com/CreditCard.owl#Visa</rdf:first>
78               </rdf:List>
79             </rdf:rest>
80             <rdf:first rdf:resource="#CustomerCreditcardType_1"/>
81           </rdf:List>
82         </swrl:arguments>
83         <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#equal"/>
84       </swrl:BuiltinAtom>
85       </rdf:first></expr:expressionBody>
86  </bsd:AcceptedCreditCard>
```

รูปที่ ก.1 อาวล์-เอสโพรเซสโมเดลของร้านขายไวน์แรก (ต่อ)

```
87<bsd:CustomerCreditcardType rdf:ID="CustomerCreditcardType_1"/>
88 <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
89   <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
90  </rdf:Description>

91 <bsd:OrderedWine rdf:ID="OrderedWine_1">
92   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
93   >http://a.com/ontology#PurchaseOrderConfirmation</process:parameterType>
94  </bsd:OrderedWine>

95 <bsd:CustomerInfo rdf:ID="CustomerInfo_1">
96   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
97    > http://localhost/ontology/domain/Customerinfo.owl#Customerinfo</process:parameterType>
98 </bsd:CustomerInfo>

99 <bsd:WineName rdf:ID="WineName_1">
100    <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
101    >http://localhost/ontology/domain/food.owl#Wine</process:parameterType>
102</bsd:WineName>
103        <to:Town rdf:ID = “Town”>
104        <cd:CreditCard rdf:ID=”CreditCard”>
105 <bsd:SellWineProcess rdf:ID="AlcoholShopAtomicProcess">
106        <process:hasInput rdf:resource="#CustomerInfo_1"/>
107        <process:hasInput rdf:resource="#WineName_1"/>
108        <process:hasPrecondition rdf:resource="#AcceptedCreditCard_1"/>
109        <process:hasLocal rdf:resource="#CustomerCreditcardType_1"/>
110        <process:hasResult rdf:resource="#OrderConfirmed_1"/>
111        <process:hasOutput rdf:resource="#OrderedWine_1"/>
112 </bsd:SellWineProcess >
</rdf:RDF>
```

รูปที่ ก.1 อาวล์-เอสโพรเซสโมเดลของร้านขายไวน์แรก (ต่อ)

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:fo="http://www.w3.org/TR/2003/PR-owl-guide-20031209/food#"
    xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns="http://www.owl-ontologies.com/FindWine1.owl#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:fwd="http://www.owl-ontologies.com/FindWineDomain.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.owl-ontologies.com/FindWine1.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://127.0.0.1:8080/ontology/Food.owl"/>
    <owl:imports rdf:resource="http://127.0.0.1:8080/ontology/FindWineDomain.owl"/>
  </owl:Ontology>
  <fwd:FoodName rdf:ID="FoodName_1">
    <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://localhost/ontology/domain/food.owl#MealCourse</process:parameterType>
  </fwd:FoodName>
  <fwd:WineAgentProcess rdf:ID="WineAgentProcess_1">
    <process:hasOutput>
      <fwd:WineName rdf:ID="WineName_1">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://localhost/ontology/domain/food.owl#RedWine</process:parameterType>
      </fwd:WineName>
    </process:hasOutput>
    <process:hasInput rdf:resource="#FoodName_1"/>
  </fwd:WineAgentProcess>
  <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
    <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
    <expr:expressionBody rdf:parseType="Literal"><swrl:AtomList xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:swrl="http://www.w3.org/2003/11/swrl#" rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"></swrl:AtomList>
  </expr:expressionBody>
  </rdf:Description>
</rdf:RDF>
```

รูปที่ ก.2 อาวล์-เอสโพรเซสโมเดลของบริการค้นหาไวน์บริการแรก

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:process="http://www.daml.org/services/owl-s/1.1/Process.owl#"
    xmlns:list="http://www.daml.org/services/owl-s/1.1/generic/ObjectList.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:time="http://www.isi.edu/~pan/damltime/time-entry.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:fo="http://www.w3.org/TR/2003/PR-owl-guide-20031209/food#"
    xmlns:expr="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns="http://www.owl-ontologies.com/FindWineDescription1.owl#"
    xmlns:service="http://www.daml.org/services/owl-s/1.1/Service.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:grounding="http://www.daml.org/services/owl-s/1.1/Grounding.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:fdd="http://www.owl-ontologies.com/FindWineDescriptionDomain.owl#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:profile="http://www.daml.org/services/owl-s/1.1/Profile.owl#"
  xml:base="http://www.owl-ontologies.com/FindWineDescription1.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://127.0.0.1:8080/ontology/Food.owl"/>
    <owl:imports rdf:resource="http://127.0.0.1:8080/ontology/FindWineDescriptionDomain.owl"/>
  </owl:Ontology>
  <fdd:WineDescription rdf:ID="WineDescription_1">
    <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
    >http://www.w3.org/2001/XMLSchema#string</process:parameterType>
  </fdd:WineDescription>
  <rdf:Description rdf:about="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#AlwaysTrue">
    <expr:expressionLanguage rdf:resource="http://www.daml.org/services/owl-s/1.1/generic/Expression.owl#SWRL"/>
    <expr:expressionBody rdf:parseType="Literal"><swrl:AtomList xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:swrl="http://www.w3.org/2003/11/swrl#" rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"></swrl:AtomList>
  </expr:expressionBody>
  </rdf:Description>
  <fdd:WineDetailProcess rdf:ID="WineDetailProcess_1">
    <process:hasOutput rdf:resource="#WineDescription_1"/>
    <process:hasInput>
      <fdd:WineName rdf:ID="WineName_2">
        <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI"
        >http://localhost/ontology/domain/food.owl#RedWine</process:parameterType>
      </fdd:WineName>
    </process:hasInput>
  </fdd:WineDetailProcess>
</rdf:RDF>
```

รูปที่ ก.3 อาวล์-เอสโพรเซสโมเดลของบริการแสดงรายละเอียดของไวน์บริการแรก

```
<process xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        name="BpelProcess"
        targetNamespace="http://xmlns.oracle.com/Bpel"
        xmlns:tns="http://xmlns.oracle.com/Bpel">
  <partnerLinks>
    <partnerLink name="Client" xmlns:ns1="http://enterprise.netbeans.org/bpel/Client" partnerLinkType="ns1:ClientPartner"
myRole="ClientPortTypeRole"/>
    <partnerLink name="FindWine" xmlns:ns2="http://localhost:8080/axis/FindWine2.jws"
partnerLinkType="ns2:FindWineLinkType" partnerRole="FindWineRole"/>
    <partnerLink name="FindWineDescription" xmlns:ns3="http://localhost:8080/axis/FindWineDescription1.jws"
partnerLinkType="ns3:FindWineDescriptionLinkType" partnerRole="FindWineDescriptionRole"/>
    <partnerLink name="OrderWine" xmlns:ns4="http://localhost:8080/axis/OrderWine1.jws"
partnerLinkType="ns4:OrderWineLinkType" partnerRole="OrderWineRole"/>
  </partnerLinks>
  <variables>
    <variable name="Input" xmlns:ns5="http://enterprise.netbeans.org/bpel/Client" messageType="ns5:ClientOperationRequest"/>
    <variable name="Output" xmlns:ns6="http://enterprise.netbeans.org/bpel/Client" messageType="ns6:ClientOperationReply"/>
    <variable name="FindWineinput" xmlns:ns7="http://localhost:8080/axis/FindWine2.jws"
messageType="ns7:WineAgentProcess_1Request"/>
    <variable name="FindWineoutput" xmlns:ns8="http://localhost:8080/axis/FindWine2.jws"
messageType="ns8:WineAgentProcess_1Response"/>
    <variable name="FindWineDescriptioninput" xmlns:ns9="http://localhost:8080/axis/FindWineDescription1.jws"
messageType="ns9:WineDetailProcess_1Request"/>
    <variable name="FindWineDescriptionoutput" xmlns:ns10="http://localhost:8080/axis/FindWineDescription1.jws"
messageType="ns10:WineDetailProcess_1Response"/>
    <variable name="OrderWineinput" xmlns:ns11="http://localhost:8080/axis/OrderWine1.jws"
messageType="ns11:SellWineProcess_1Request"/>
    <variable name="OrderWineoutput" xmlns:ns12="http://localhost:8080/axis/OrderWine1.jws"
messageType="ns12:SellWineProcess_1Response"/>
  </variables>
 <sequence>
    <receive name="Receive"
            partnerLink="Client" xmlns:ns13="http://enterprise.netbeans.org/bpel/Client" portType="ns13:ClientPortType"
operation="ClientOperation"
           variable="Input" createInstance="yes">
    </receive>
    <assign >
      <copy>
        <from variable="Input" part="in0"/>
        <to variable="FindWineinput" part="in0"/>
      </copy>
    </assign>
    <invoke name="FindWineinvork"
          partnerLink="FindWine" xmlns:ns14="http://localhost:8080/axis/FindWine2.jws" portType="ns14:FindWine2"
operation="WineAgentProcess_1"
          inputVariable="FindWineinput" outputVariable="FindWineoutput">
    </invoke>
    <flow>
      <sequence>
        <assign >
          <copy>
            <from variable="FindWineoutput" part="WineAgentProcess_1Return"/>
            <to variable="FindWineDescriptioninput" part="in0"/>
          </copy>
        </assign>
        <invoke name="FindWineDescriptioninvork"
              partnerLink="FindWineDescription" xmlns:ns15="http://localhost:8080/axis/FindWineDescription1.jws"
portType="ns15:FindWineDescription1" operation="WineDetailProcess_1"
              inputVariable="FindWineDescriptioninput" outputVariable="FindWineDescriptionoutput">
        </invoke>
      </sequence>
      <sequence>
        <assign >
          <copy>
            <from variable="FindWineoutput" part="WineAgentProcess_1Return"/>
            <to variable="OrderWineinput" part="in0"/>
          </copy>
        </assign>
        <invoke name="OrderWineinvork"
              partnerLink="OrderWine" xmlns:ns16="http://localhost:8080/axis/OrderWine1.jws" portType="ns16:OrderWine1"
operation="SellWineProcess_1"
              inputVariable="OrderWineinput" outputVariable="OrderWineoutput">
        </invoke>
      </sequence>
    </flow>
```

รูปที่ ก.4 เอกสารบีเพลที่เป็นผลลัพธ์จากตัวกลาง

```
<assign >
     <copy>
       <from variable="OrderWineoutput" part="SellWineProcess_1Return"/>
       <to variable="Output" part="SellWineProcess_1Return"/>
     </copy>
   </assign>
   <assign >
     <copy>
       <from variable="FindWineDescriptionoutput" part="WineDetailProcess_1Return"/>
       <to variable="Output" part="WineDetailProcess_1Return"/>
     </copy>
   </assign>
   <reply name="Reply"
          partnerLink="Client" xmlns:ns17="http://enterprise.netbeans.org/bpel/Client" portType="ns17:ClientPortType"
operation="ClientOperation"
          variable="Output">
   </reply>
  </sequence>
</process>
```

รูปที่ ก.4 เอกสารบีเพลที่เป็นผลลัพธ์จากตัวกลาง (ต่อ)

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="WineService" targetNamespace="http://enterprise.netbeans.org/bpel/Client"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://enterprise.netbeans.org/bpel/Client"
xmlns:plink="http://schemas.xmlsoap.org/ws/2004/03/partner-link/">
    <types/>

    <message name="ClientOperationRequest">
        <part name="in0" type="xsd:string"/>
    </message>
    <message name="ClientOperationReply">
        <part name="SellWineProcess_1Return" type="xsd:string"/>
                <part name="WineDetailProcess_1Return" type="xsd:string"/>
    </message>
    <portType name="ClientPortType">
        <operation name="ClientOperation">
            <input name="input1" message="tns:ClientOperationRequest"/>
            <output name="output1" message="tns:ClientOperationReply"/>
        </operation>
    </portType>
    <binding name="ClientBinding" type="tns:ClientPortType">
        <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="ClientOperation">
            <soap:operation/>
            <input name="input1">
                <soap:body use="literal" namespace="http://enterprise.netbeans.org/bpel/Client"/>
            </input>
            <output name="output1">
                <soap:body use="literal" namespace="http://enterprise.netbeans.org/bpel/Client"/>
            </output>
        </operation>
    </binding>
    <service name="ClientService">
        <port name="ClientPort" binding="tns:ClientBinding">
            <soap:address location="http://localhost:18181/ClientService/ClientPort"/>
        </port>
    </service>
    <plink:partnerLinkType name="ClientPartner">
        <!-- partnerLinkType are automatically generated when a new portType is added. partnerLinkType are used by BPEL
processes.
In a BPEL process, a partner link represents the interaction between the BPEL process and a partner service. Each partner link
is associated with a partner link type.
A partner link type characterizes the conversational relationship between two services. The partner link type can have one or two
roles.-->
        <plink:role name="ClientPortTypeRole" portType="tns:ClientPortType"/>
    </plink:partnerLinkType>
</definitions>
```

รูปที่ ก.5 เอกสารดับเบิลยูเอสดีแอลของบีเพล

ภาคผนวก ข

ผลงานตีพิมพ์

ผลงานตีพิมพ์ซึ่งเป็นส่วนหนึ่งของงานวิจัยมีดังนี้

1. Proceedings of 4th International Joint Conference on Computer Science and Software Engineering, Khon Kaen, Thailand (2007): 194-201 ในบทความเรื่อง Using OWL-S Process Model with Constraints on Functional Behaviour for Automatic Web Services Composition โดยผู้แต่งคือ Termsak Wasino และ Twittie Senivongse

Using OWL-S Process Model with Constraints on Functional Behaviour for Automatic Web Services Composition

Termsak Wasino and Twittie Senivongse
Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University
Phyathai Road, Pathumwan, Bangkok 10330 Thailand
termsak.w@student.chula.ac.th, twittie.s@chula.ac.th

***Abstract*-Automatic Web services composition has been a key research issue in service-oriented computing due to the increasing demand for composite services that can serve service consumers' purposes, together with the complexity of the composition process. This paper proposes an approach to automatic composition of Web services by considering functional compatibility of the composed services. Given a composition goal, the approach examines process specifications of the Web services which are represented by OWL-S process model and determines similarity between functional behaviour of those Web services, i.e. their inputs, outputs, preconditions, and effects as well as their associated constraints. An optimal composition is identified by using Dijkstra's shortest path algorithm and a composite service can be generated as a BPEL script. A service composition framework for service designers is also presented.**

I. INTRODUCTION

Web services technology has been widely accepted as a technology for building loosely-coupled software components as services which can be composed into larger services with aggregated functionalities. This is due to the fact that service consumers' requirements cannot be fulfilled by a single Web service and hence several Web services are aggregated into a certain composition in order to answer the requirements. Web services technology supports a Business Process Execution Language (BPEL) as a de-facto standard for describing a workflow of collaborating Web services [1]. With BPEL, the workflow can be seen as an internal process of a composite Web service which executes by the orchestration of an execution engine. A service designer designs a composite Web service by defining Web service instances that will participate in particular tasks within the workflow. This manual composition is not an easy task as the service designer will need prior knowledge about each Web service to be composed and will need to ensure that those Web services are compatible and can really collaborate.

Automatic Web services composition hence comes into the picture to aid service designers. The composition comprises 1) automated discovery of candidate services to be composed 2) automated composition of candidate services that altogether can fulfill the composition goal and 3) automated execution of the resulting composite service [2]. In many cases, automatic Web services composition makes use of semantic technology, i.e. ontology, as it enables the composition process to also consider semantic compatibility of the candidate services.

Research in automatic Web services composition can be viewed in two approaches: composition with a conceptual process flow and composition without a conceptual process flow. The former requires a service designer's knowledge about what tasks to be composed and how they are composed, and the designer defines a conceptual plan that specifies the collaboration of the tasks as a workflow. Therefore composition becomes the process that assigns a suitable Web service for each particular task. For the latter, a service designer does not have prior knowledge about the conceptual process flow. All that is known is the input and the expected goal or output; the designer does not know what tasks are to be in the workflow to produce the expected output, given the input. Therefore composition becomes the process that finds a chain of Web services that can consume the given input and produces the specified output from its flow. Although this second approach is convenient, the given input and output are considered very loose requirements and the chains of composed services may vary considerably.

This paper proposes a technique for automatic Web services composition that belongs to the first approach. The technique aims at helping service designers to select an optimal set of composed services that fit together into a conceptual process flow according to their functional behaviour, i.e. inputs, outputs, preconditions, and effects. The functional behaviour of a Web service is defined in an ontology-based OWL-S process specification [3]. The contribution of this paper is that the functional behaviour may have associated constraints. Automated composition will also evaluate these constraints to determine the degree of composability between any pair of candidate services to be composed. The result of composition will be a number of concrete flows that represent composite services. We identify an optimal composite service by using Dijkstra's shortest path algorithm [4] and have a corresponding BPEL script generated for the service.

The rest of the paper is organised as follows. Section II discusses related work. A case study of the conceptual process flow to compose a Wine composite service is introduced in Section III. Section IV presents shared process ontologies and external ontologies that are the basis for the composition, and Section V gives OWL-S process specifications of Web services for the case study. Section VI lists matching criteria that are used to determine composable candidate services. Section VII shows how to determine concrete process flow and how an optimal composition is

identified. Section VIII presents a composition framework, and finally a discussion and conclusion with future research directions will be in Section IX.

II. Related work

As mentioned earlier, automatic Web services composition can be viewed in two approaches: composition with a conceptual process flow [5], [6], [7] and composition without a conceptual process flow [2], [8], [9]. Also, most research considers either quality of service (QoS) attributes or behavioural semantics of the Web services as the criteria to determine the quality of the composition. Ref. [5] assigns Web services to the conceptual process flow by considering their quality (i.e. execution duration, reliability, availability, price, reputation) so that the resulting process flow will give the best quality all round. Ref. [6] considers QoS attributes but applies a genetic algorithm to find a composite service with an optimal quality. Ref. [7] considers behavioural semantics together with QoS attributes, and BPEL is generated for the resulting process flow. Refs. [2], [8] use OWL-S process model to define the input and output of the required composite service and have an AI planner generate possible plans, i.e. possible process flows. Ref. [9] uses DAML-S (the predecessor of OWL-S) to define the input and expected output, and uses a shortest path algorithm to find the best process flow according to semantic compatibility and execution time between pairs of services within the flow.

This paper takes the approach of composition with a predefined process flow. It is close to [7] and [9] in that behavioural semantics of the Web services are considered and a shortest path algorithm is used to find an optimal composition. However, QoS attributes are not taken into account here as it is difficult in practice to obtain such attributes whose values are always changing, and none of the research above considers functional behaviour of the services with associated constraints.

III. Conceptual Process Flow and Constraints

This section introduces a case study, adapted from [9], that will be used throughout the paper. A service designer would like "a Wine composite service that can find a matching wine for a particular meal and also order the wine. The payment can be made by a certain credit card (e.g. Amex) and the wine must be delivered to the customer's address (e.g. Bangkok) within a specified period (e.g. 3 days). The service should also give the details of the ordered wine." Knowing about these required tasks, the service designer defines a conceptual process flow as in Fig. 1. The flow involves 1) the FindWine task which gives a matching wine name for a given food name 2) the OrderWine task which takes an order for a wine from a customer and 3) the FindWineDescription which gives a description of a particular wine. The service designer may use an ontology editor tool such as Protégé [10] to design the graphical flow and have the tool generate a corresponding ontology-based OWL-S process specification (Fig. 2). The specification shows a flow of tasks (i.e. AtomicProcess) that are chained by control constructs (e.g. Sequence, Split-Join).

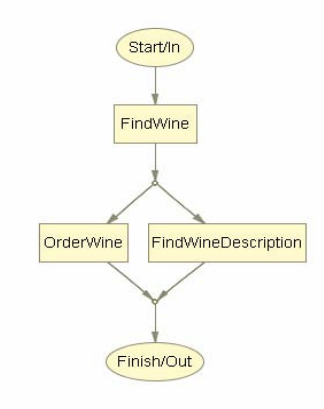


Figure 1. Conceptual process flow of wine composite service.

```
…
<process:Sequence rdf:ID="Sequence_2">
 <process:components>
  <process:ControlConstructList rdf:ID="ControlConstructList_4">
   <list:first>
    <process:Perform rdf:ID="FindWineState">
      <process:process>
        <process:AtomicProcess rdf:ID="FindWine"/>
      </process:process>
    </process:Perform>
 </list:first>
 <list:rest>
     <process:ControlConstructList
          rdf:ID="ControlConstructList_7">
      <list:rest rdf:resource="http://www.daml.org/services/
          owl-s/1.1/generic/ObjectList.owl#nil"/>
     <list:first>
       <process:Split-Join rdf:ID="Split-Join_6">
        <process:components>
         <process:ControlConstructBag
               rdf:ID="ControlConstructBag_9">
          <list:first>
           <process:Perform rdf:ID="OrderWineState">
            <process:process>
              <process:AtomicProcess rdf:ID="OrderWine"/>
            </process:process>
           </process:Perform>
          </list:first>
          <list:rest>
           <process:ControlConstructBag
                 rdf:ID="ControlConstructBag_11">
            <list:rest
                  rdf:resource="http://www.daml.org/services/
                      owl-s/1.1/generic/ObjectList.owl#nil"/>
           <list:first>
            <process:Perform
                  rdf:ID=" FindWineDescriptionState">
             <process:process>
               <process:AtomicProcess
                     rdf:ID="FindWineDescription "/>
             </process:process>
            </process:Perform>
           </list:first>
          </process:ControlConstructBag>
         </list:rest>
        </process:ControlConstructBag>
       </process:components>
      </process:Split-Join>
     </list:first>
    </process:ControlConstructList>
   </list:rest>
  </process:ControlConstructList>
 </process:components>
</process:Sequence>
  …
```

Figure 2. OWL-S process specification for conceptual process flow.

Additionally, the service designer defines necessary information as well as the constraints related to the functional behaviour of each task within the conceptual process flow (Fig. 3). Regarding the functional behaviour of the tasks, outputs from a task should essentially supply appropriate inputs to subsequent tasks in the flow. In the example, given the food name, the FindWine task should give a matching wine name which is then passed on as an input for the subsequent OrderWine and FindWineDescription tasks. However, some tasks may have additional behavioural constraints. In the example, the OrderWine task requires that any Web services that can perform this task must be able to accept Amex credit card and deliver the ordered wine within 3 days to the customer's address in Bangkok. These requirements can be present in the OWL-S conceptual process flow, but for simplicity, we represent them here as in Fig. 3 as a set of relation expressions of the requirements Q. Each expression is in the form of *property(subject, object)* which corresponds to an ontology-based RDF statement <*subject, property, object*>. For an expression that relates to a numerical range (e.g. the expression on the number of days for delivery), the expression is represented as *property(argument, relational operator, literal value 1 [, literal value 2])*. Each relation expression is superscripted by any of the symbols *I*, *O*, *P* or *R*. The symbol respectively refers to the functional behaviour on which the relation expression applies, i.e. input, output, precondition, and result. (Note that result is OWL-S collective construct for output and effect.)

The service designer will define necessary information (i.e. food name, address, credit card) and constraints on functional behaviour of each task by using ontological terms defined in the shared ontologies of the relevant application domains (see Section IV). These shared domain ontologies help to bridge the semantics of the service designer's requirements to the semantics of the Web services to be composed.

```
input :
http://www.w3.org/TR/2003/PR-owl-guide-20031209/Food#
        MealCourse
http://www.owl-ontologies.com/CustomerInfo.owl#CustomerInfo

 #FindWine{
 Q =  { HasInput(Process,FoodName)^I
         HasOutput(Process,WineName)^O
        }
               }
#OrderWine{
 Q = { HasInput(Process,WineName)^I
        HasInput(Process,CustomerInfo)^I
        HasOutput(Process,OrderedWine)^O
        HasCustomerCreditcardType(CreditCardType,Amex)^P
        HasCustomerLocation(CustomerLocation,Bangkok)^R
        HasDeliveryDay(DeliveryDay,LessThanOrEqual,3)^R
     }
               }
#FindWineDescription{
Q = { HasInput(Process,WineName)^I
       HasOutput(Process,WineDescription)^O
     }
                        }
```

Figure 3. Necessary information and constraints on functional behaviour.

IV. Shared Ontologies

The composition algorithm assumes the shared ontologies of the application domain are adopted by service providers who offer Web services and by service designers.

A. Process Ontology

Process ontology refers to an ontology that describes common process of Web services within a particular domain. The ontology is defined by domain experts and adopted by service providers within the domain. We represent a process ontology with an OWL-S process model; it defines common vocabularies (i.e. ontological classes) for the process, inputs, output, conditions, and related information. Fig. 4 shows part of the process ontology for FindWine; it defines a FindWineProcess as a subclass of OWL-S atomic process with FoodName as an input, and WineName, WinePrice, and WineDescription as outputs. Fig. 5 shows part of the process ontology for OrderWine; it defines an OrderWineProcess as a subclass of OWL-S atomic process with WineName and CustomerInfo as inputs, and OrderedWine as an output. AcceptedCreditCard is the condition for the process; it will depend on the CustomerCreditcardType local variable (i.e. the process will accept an order only from the customer who holds an accepted credit card). OrderConfirmed is the result of the process and it will be associated with a DeliveryDayCondition constraint. This constraint is related to the LocationCondition which depends on the CustomerLocation result variable (i.e. the process will confirm the order and make a delivery within a specified period if the customer is located in a certain area).

B. External Ontology

External ontology refers to auxiliary ontologies, the vocabularies of which are used by service designers to specify necessary information for the composition, and by service providers in defining constraints in the process specifications of Web services. Fig. 6 shows part of the food ontology (presented by the Protégé tool). In our example, we also use the external ontology for customer information which relates to customer's credit card and location.

```
…
 <owl:Class rdf:ID="FoodName">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#Input"/>
 </owl:Class>
 <owl:Class rdf:ID="WineName">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#Output"/>
 </owl:Class>
 <owl:Class rdf:ID="WinePrice">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#Output"/>
 </owl:Class>
 <owl:Class rdf:ID="WineDescription">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#Output"/>
 </owl:Class>
<owl:Class rdf:ID="FindWineProcess">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#AtomicProcess"/>
 </owl:Class>
…
```

Figure 4. Part of shared process ontology for FindWine.

```
…
 <owl:Class rdf:ID="DeliveryDayCondition">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/generic/Expression.owl#Condition"/>
 </owl:Class>
 <owl:Class rdf:ID="OrderConfirmed">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#Result"/>
 </owl:Class>
 <owl:Class rdf:ID="CustomerLocation">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#ResultVar"/>
 </owl:Class>
 <owl:Class rdf:ID="LocationCondition">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/generic/Expression.owl#Condition"/>
 </owl:Class>
 <owl:Class rdf:ID="AcceptedCreditCard">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/generic/Expression.owl#Condition"/>
 </owl:Class>
 <owl:Class rdf:ID="CustomerCreditcardType">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#Local"/>
 </owl:Class>
 <owl:Class rdf:ID="WineName">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#Input"/>
 </owl:Class>
 <owl:Class rdf:ID="CustomerInfo">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#Input"/>
 </owl:Class>
 <owl:Class rdf:ID="DeliveryDay">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
     owl-s/1.1/Process.owl#ResultVar"/>
 </owl:Class>
 <owl:Class rdf:ID="OrderedWine">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#Output"/>
 </owl:Class>
<owl:Class rdf:ID="OrderWineProcess">
  <rdfs:subClassOf rdf:resource="http://www.daml.org/services/
      owl-s/1.1/Process.owl#AtomicProcess"/>
 </owl:Class>
…
```

Figure 5. Part of shared process ontology for OrderWine.

Figure 6. Part of external ontology for Food.

V. Process Specifications of Web Services

Service providers will publish process specifications of their Web services by using OWL-S process model. Fig. 7 shows part of the process specification of a wine shop that sells wine (i.e. a Web service for OrderWine). The process specification is based on the process ontology in Fig. 5 and the external ontologies for city and credit card. This wine shop receives wine name (line 99-102, 107) and customer info (line 95-98, 106) as inputs, and gives an ordered wine as an output (line 91-94, 111). The precondition of the process is that the accepted credit card must be evaluated to true (line 108) and this depends on the credit card type that the customer holds (line 87, 109). The accepted cards are Amex and Visa cards (line 59-86). The result of the process is that the order is confirmed (line 110) with a constrained effect on wine delivery within 3 days (line 5-18). But this guarantee is only when the customer is located in Bangkok (line 30-44). All the constraints about credit card types, customer's location, and the number of days for delivery are described by the Semantic Web Rule Language (SWRL) [11].

```
…
1<bsd:OrderConfirmed rdf:ID="OrderConfirmed_1">
2 <process:hasResultVar>
3    <bsd:DeliveryDay rdf:ID="DeliveryDay_1"/>
4   </process:hasResultVar>
5   <process:hasEffect>
6    <bsd:DeliveryDayCondition rdf:ID="DeliveryDayCondition_1">
7      <expr:expressionBody rdf:datatype="http://www.w3.org/1999/
            02/22-rdf-syntax-ns#XMLLiteral"
8      ><swrl:AtomList>
9    <rdf:first>
10       <swrl:BuiltinAtom>
11        <swrl:builtin rdf:resource="http://www.w3.org/2003/11/
                swrlb#lessThanOrEqual/>
12         <swrl:arguments>
13          <rdf:List>
14           <rdf:first rdf:resource="#DeliveryDay_1"/>
15            <rdf:rest>
16             <rdf:List>
17              <rdf:first rdf:datatype="http://www.w3.org/2001/
                      XMLSchema#int"
18              >3</rdf:first>
…
30  <process:inCondition>
31   <bsd:LocationCondition rdf:ID="LocationCondition_1">
32    <expr:expressionBody rdf:datatype="http://www.w3.org/1999/
            02/22-rdf-syntax-ns#XMLLiteral"
33    ><swrl:AtomList>
34    <rdf:first>
35     <swrl:BuiltinAtom>
36      <swrl:builtin rdf:resource="http://www.w3.org/2003/11/
              swrlb#equal"/>
37       <swrl:arguments>
38        <rdf:List>
39         <rdf:first rdf:resource="#CustomerLocation_1">
40          <rdf:rest>
41           <rdf:List>
42            <rdf:rest rdf:resource="http://www.w3.org/1999/
                    02/22-rdf-syntax-ns#nil"/>
43             <rdf:first rdf:datatype="http://www.w3.org/2001/
                    XMLSchema#anyURI"
44              > http://www.owl-ontologies.com/
                  City.owl#Bangkok</rdf:first>
…
```

Figure 7. Part of process specification of an OrderWine Web service.

```
55   <process:hasResultVar>
56     <bsd:CustomerLocation rdf:ID="CustomerLocation_1"/>
57   </process:hasResultVar>
…
59 <bsd:AcceptedCreditCard rdf:ID="AcceptedCreditCard_1">
60   <expr:expressionBody rdf:datatype="http://www.w3.org/1999/
          02/22-rdf-syntax-ns#XMLLiteral">
61   <swrl:AtomList>
62       <rdf:rest rdf:resource="http://www.w3.org/1999/
            02/22-rdf-syntax-ns#nil"/>
…
71                  <rdf:first rdf:datatype="http://www.w3.org/2001/
                        XMLSchema#anyURI"
72                  >http://www.owl-ontologies.com/
                      CreditCard.owl#Amex</rdf:first>
73                  <rdf:rest rdf:resource="http://www.w3.org/1999/
                        02/22-rdf-syntax-ns#nil"/>
74                </rdf:List>
75              </rdf:rest>
76              <rdf:first rdf:datatype="http://www.w3.org/2001/
                    XMLSchema#anyURI"
77              >http://www.owl-ontologies.com/
                  CreditCard.owl#Visa</rdf:first>
78            </rdf:List>
79          </rdf:rest>
80          <rdf:first rdf:resource=
                "#CustomerCreditcardType_1"/>
81        </rdf:List>
82      </swrl:arguments>
83       <swrl:builtin rdf:resource="http://www.w3.org/2003/
              11/swrlb#equal"/>
84      </swrl:BuiltinAtom>
85      </rdf:first></expr:expressionBody>
86 </bsd:AcceptedCreditCard>
87<bsd:CustomerCreditcardType rdf:ID=
        "CustomerCreditcardType_1"/>
88 <rdf:Description rdf:about="http://www.daml.org/services/
       owl-s/1.1/generic/Expression.owl#AlwaysTrue">
89   <expr:expressionLanguage rdf:resource=
           "http://www.daml.org/services/owl-s/1.1/generic/
              Expression.owl#SWRL"/>
90 </rdf:Description>
91 <bsd:OrderedWine rdf:ID="OrderedWine_1">
92   <process:parameterType rdf:datatype="http://www.w3.org/
              2001/XMLSchema#anyURI"
93   >http://a.com/ontology
       #PurchaseOrderConfirmation</process:parameterType>
94 </bsd:OrderedWine>
95 <bsd:CustomerInfo rdf:ID="CustomerInfo_1">
96   <process:parameterType rdf:datatype=
            "http://www.w3.org/2001/XMLSchema#anyURI"
97   >http://www.owl-ontologies.com/Customerinfo.owl
        #Customerinfo</process:parameterType>
98 </bsd:CustomerInfo>
99 <bsd:WineName rdf:ID="WineName_1">
100   <process:parameterType rdf:datatype="http://www.w3.org/
              2001/XMLSchema#anyURI"
101   >http://www.w3.org/TR/2003/PR-owl-guide-20031209/
         Food#Wine</process:parameterType>
102</bsd:WineName>
103<ci:City rdf:ID = "City"/>
104<cd:CreditCard rdf:ID="CreditCard"/>
105 <bsd:OrderWineProcess rdf:ID="WineShopAtomicProcess">
106       <process:hasInput rdf:resource="#CustomerInfo_1"/>
107       <process:hasInput rdf:resource="#WineName_1"/>
108       <process:hasPrecondition rdf:resource=
                  "#AcceptedCreditCard_1"/>
109       <process:hasLocal rdf:resource=
                  "#CustomerCreditcardType_1"/>
110       <process:hasResult rdf:resource="#OrderConfirmed_1"/>
111       <process:hasOutput rdf:resource="#OrderedWine_1"/>
112 </bsd:OrderWineProcess >
...
```

Figure 7. Part of process specification of an OrderWine Web service (continued).

VI. MATCHING CRITERIA

Matching the process specifications of Web services to the conceptual process flow of the service designer is based on matching of ontological terms that describe the functional behaviour of the Web services and on the evaluation of the constraints associated to such functional behaviour. We adopt the criteria from [12] on matching of ontological concepts, matching of numerical range (e.g. time duration of wine delivery), and matching of enumeration values (e.g. accepted credit card types). We also define a match score for each kind of match for later use in Section VII.

A. Matching of Ontological Concepts

Let C_Q be the ontological concept specified in the conceptual process flow and C_P be the ontological concept in the process specification:

(i) If $C_Q \equiv C_P$, then C_P is an exact match for C_Q where $\equiv$ means is equivalent to. The match score for this case is 1.

(ii) If $C_P \sqsubseteq C_Q$, then C_P is a specialised match for C_Q where $\sqsubseteq$ means is subsumed by (i.e. C_P is more specific than C_Q). The match score for this case is 2.

(iii) If $C_Q \sqsubseteq C_P$, then C_P is a generalised match for C_Q. This means the concept in the conceptual process flow is more specific than, and is subsumed by, the one in the process specification. The match score for this case is 3.

(iv) If $(C_Q \not\sqsubseteq C_P) \wedge (C_P \not\sqsubseteq C_Q) \wedge (C_Q \sqsubseteq C_C) \wedge (C_P \sqsubseteq C_C)$, then C_P is a partial match for C_Q, where $\not\sqsubseteq$ means is not subsumed by and C_C is a node in the same IS-A taxonomy. This means it is acceptable for the concept in the process specification to be a match for the concept in the conceptual process flow provided that the two concepts have common characteristics through a common parent concept. The match score for this case is 4.

(v) If none of the above relationships exist, then C_P is a failed match for C_Q. No match score is defined for this case.

B. Matching of Numerical Ranges

Let N_Q be a nonempty set of numerical range values of the expression in the conceptual process flow (S_Q), and N_P be a nonempty set of numerical range values of the expression in the process specification (S_P):

(i) If $N_P \subseteq N_Q$, then S_P is an exact match for S_Q. The match score for this case is 1.

(ii) If $N_Q \subseteq N_P$, then S_P is a plug-in match for S_Q. The match score for this case is 2.

(iii) If $(N_P \cap N_Q \neq \phi) \wedge (N_P \not\subseteq N_Q) \wedge (N_Q \not\subseteq N_P)$, then S_P is a weak match for S_Q. The match score for this case is 3.

(iv) If $N_P \cap N_Q = \phi$, then S_P is a failed match for S_Q. No match score is defined for this case.

C. Matching of Enumeration Values

Let E_Q be a nonempty set of enumeration values *i* of the expression in the conceptual process flow and E_P be a nonempty set of enumeration values *j* in the process specification. The enumeration values may be either XML schema data type values or ontological values.

If enumeration values are XML schema data type values:

(i) If $\forall i, \exists j: (i \in E_Q) \wedge (j \in E_P) \wedge (i = j)$, then E_P is an exact match for E_Q. The match score for this case is the summation of the scores of all *i* and *j* pairs that are matched, where each match pair has a score of 1.

(ii) If none of the above relationship exists, then E_P is a failed match for E_Q. No match score is defined for this case.

If enumeration values are ontological values:

(i) If $\forall i, \exists j: (i \in E_Q) \wedge (j \in E_P) \wedge (i \otimes j)$, then E_P is an exact match for E_Q where $\otimes$ means ontological matching as in Subsection A above. The match score for this case is the summation of the scores of all *i* and *j* pairs that are matched, where each match pair has a score of 1-4 depending on the kind of ontological match.

(ii) If none of the above relationship exists, then E_P is a failed match for E_Q. No match score is defined for this case.

VII. Finding an Optimal Composite Service

To find an optimal composite service that can best answer the conceptual process flow, we will first identify possible concrete compositions and then apply the shortest path algorithm of Dijkstra to determine the optimal composition.

To identify possible concrete compositions, process specifications of the Web services are compared against the conceptual process flow and its associated constraints. The process specification of a Web service will be a match to a particular task in the conceptual process flow if, based on the matching criteria, its functional behaviour and constraints match to all the relation expressions that the service designer requires for that task. For example, with the service designer's requirement as in Fig. 1-3, the outputs of the Web services of a particular task must match to the inputs of a subsequent task in the conceptual process flow. That is, the outputs of FindWine Web services must match to the inputs of the OrderWine and FindWineDescription Web services. (Note that, matching of outputs and inputs should consider both signature and semantics of the Web services, but this paper will demonstrate semantic matching only.) Also, each Web service must match according to the behavioural constraints, e.g. the OrderWine Web services must accept Amex credit card.

Assume that there are two wine agents (i.e. WineAgent1 and WineAgent2 Web services) that support the FindWine task, and two wine shops (i.e. WineShop1 and WineShop2 Web services) that support the OrderWine task. Fig. 8 shows the functional behaviour and associated constraints of these Web services as well as the kinds of matching the relation expressions of each Web service have with the service designer's requirement of Fig. 1-3. (Note that the functional behaviour and constraints of WineShop1 are the same as what have been shown in Fig. 7.) By matching of ontological concepts, most relation expressions of these Web services are an exact match to the corresponding expressions from the service designer. By matching of enumeration values, the expressions about the credit card type of WineShop1 and WineShop2 are an exact match. By matching of numerical ranges, the expression about delivery day of WineShop1 is an exact match while that of WineShop2 is a plug-in match.

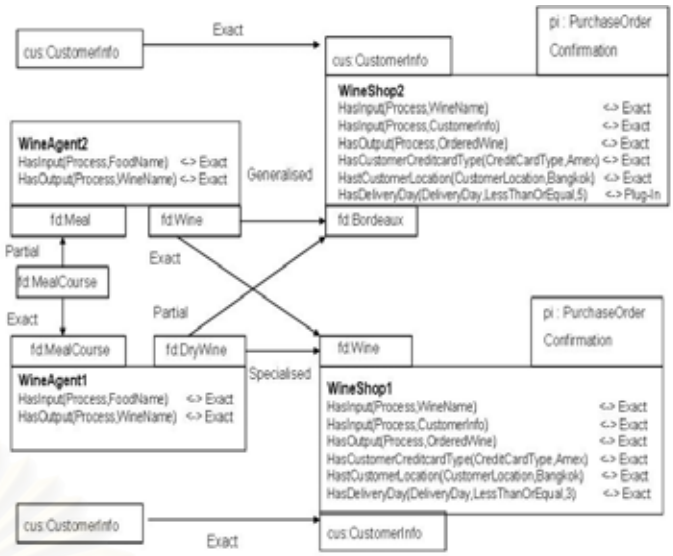


Figure 8. Matching between FindWine and OrderWine Web services.

Moreover, Fig. 8 also shows that the FoodName input of WineAgent1 is of class MealCourse and the WineName output is of class DryWine, whereas the FoodName input of WineAgent2 is of class Meal and the WineName output is of class Wine. For WineShop1, the WineName and CustomerInfo inputs are of classes Wine and CustomerInfo respectively, and the OrderedWine output is of class PurchaseOrderConfirmation. For WineShop2, the WineName and CustomerInfo inputs are of classes Bordeaux and CustomerInfo respectively, and the OrderedWine output is of class PurchaseOrderConfirmation. Thus, provided that the service designer specifies an input of class MealCourse, the FoodName input of WineAgent1 is an exact match while the FoodName input of WineAgent2 is a partial match according to ontological matching. Consequently, the WineName output of WineAgent1 can match to the WineName input of WineShop1 as a specialised match, whereas it is a partial match to the WineName input of WineShop2. But for WineAgent2, its WineName output is an exact match to the WineName input of WineShop1 and a generalised match to the WineName input of WineShop2. Also, the service designer's input of class CustomerInfo is an exact match to the CustomerInfo input of both WineShop1 and WineShop2.

We can transform Fig. 8 into a graph that, starting with the service designer's inputs, represents possible paths of composition as in Fig. 9. Each edge of the graph is labelled with the match score that represents the degree of similarity between the two connecting nodes. Each match score is the summation of the score for input/output matching and the score for constraint matching. For example, the match score between the service designer's input of class MealCourse and WineAgent2 is equal to 4+2 because of the partial match (score 4) of their input classes, and the exact match (score 1) of the two input and output constraints that WineAgent2 exhibits (hence, score 2 in total). For the edge between WineAgent2 and WineShop2, the match score is (3+1)+7. The output-input match is determined by a generalised match of WineName (score 3) and an exact match of CustomerInfo (score 1). The constraint match is determined by the

constraints that WineShop2 exhibits; all are exact matches (score 1 each) except for the plug-in match (score 2) of the constraint on delivery day (hence, score 7 in total).

However, our approach also allows the service designer to specify a preference on either input/output matching or constraint matching. In other words, the service designer may give a priority to similarity of input/output semantics over similarity of constraints, and vice versa. Equation (1) is used to calculate a weighted match score for each edge where $\lambda \in [0,1]$.

Weighted match score =
λ * match score for input/output matching +
(1-λ) * match score for constraint matching. (1)

For example, if the service designer gives equal weight for input/output matching and constraint matching (i.e. $\lambda = 0.5$), the match score on the edge between WineAgent2 and WineShop2 will be 0.5*(3+1) + 0.5*7 = 5.5. Fig. 10 shows the graph of Fig. 9 with weighted match scores where $\lambda = 0.5$. With match scores, Dijkstra's shortest path algorithm can be applied to find an optimal path that represents the optimal composition. From Fig. 10, the optimal composition of FindWine and OrderWine tasks is one that composes WineAgent1 to WineShop1 with the path score equal to 6.

It is possible that there may be more than one path that gives the shortest path score. If that is the case, our approach will randomly pick one to generate a corresponding BPEL.

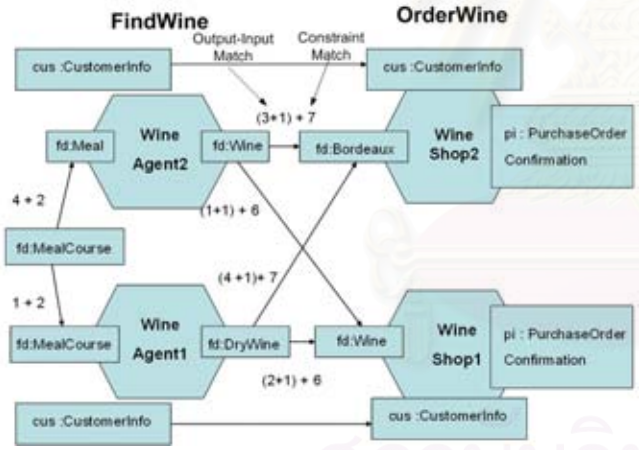


Figure 9. Graph with match scores for FindWine and OrderWine.

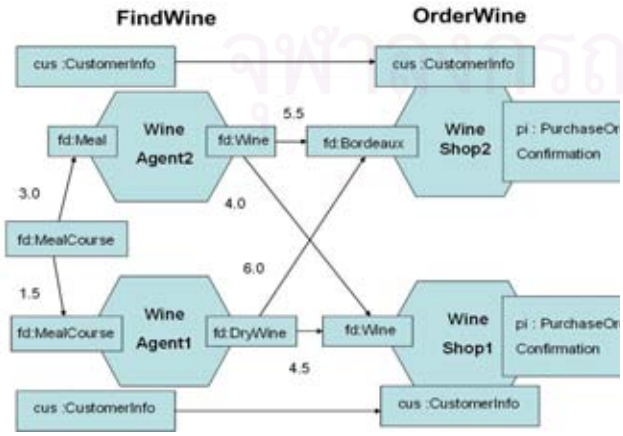


Figure 10. Graph with weighted match scores for FindWine and OrderWine.

Apart from the OrderWine task, the FindWine task in the example is also followed by a parallel FindWineDescription task. Therefore, after an optimal composition of FindWine and OrderWine Web services is found, the composition process repeats to find a FindWineDescription Web service that can be best composed to the FindWine Web service that is part of the optimal FindWine-OrderWine composition.

VIII. Service Composition Framework

A Web services composition framework provides a mediator to support Web services composition (Fig. 11). Service providers will create process specifications for their Web services by using an ontology editor such as Protégé (1). Process specifications will be based on shared process ontologies and external ontologies that are defined by domain experts. Any service provider will publish a Web service with a UDDI directory [13] as usual and additionally register through the publishing proxy the URL of the process specification and the business service key of the corresponding service entry in the UDDI; the process specification and the business service key will be stored in a process link repository (2). An OWL-S parser API is then used to extract process terms, external ontology terms, and SWRL constraints from the process specification stored in the process link repository (3). The process terms and external ontology terms are inferred by a reasoning engine and the obtained facts are stored in a knowledge base (4). SWRL constraints are parsed by a SWRL parser API and relevant terms are also inferred and stored in the knowledge base (5).

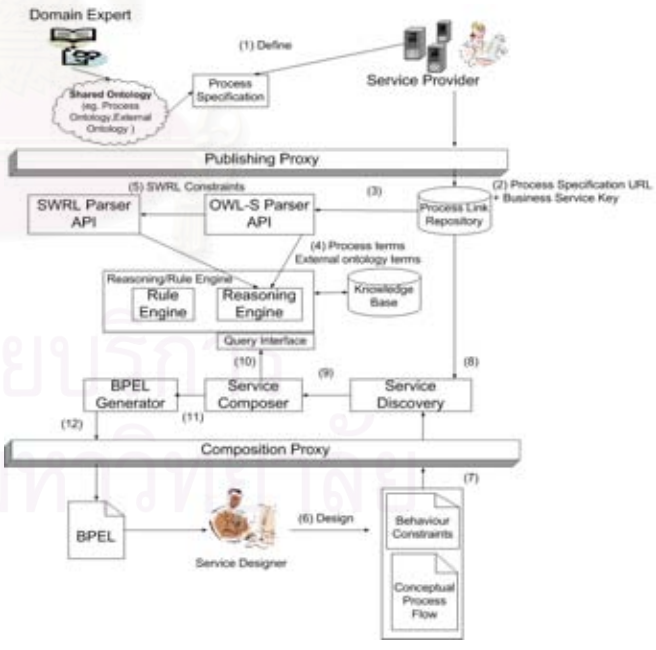


Figure 11. Service composition framework.

To compose a composite Web service, a service designer will, based on the shared process ontologies and external ontologies, design an OWL-S based conceptual process flow with associated behaviour constraints by using Protégé (6) and submits to the mediator through the composition proxy

(7). The conceptual process flow and constraints will be parsed to determine relevant service categories that will implement the required tasks, and process specifications of Web services under such categories will be discovered from the process link repository (8). Those candidate process specifications will be compared, by the service composer, with the service designer's requirement (9). The service composer will query the reasoning/rule engine which will then consult facts from the knowledge base and evaluate behaviour constraints with the rule engine component (10). Once an optimal composition is identified, the BPEL generator will create a corresponding BPEL (11) and return it back to the service designer (12). The service designer can retrieve WSDLs of the constituent Web services in the composition from the UDDI and has the BPEL script run by a BPEL execution engine.

In the current implementation, OWL-S API [14] is used to extract ontological terms and SWRL constraints from the process specifications. We develop our own SWRL parser to extract details contained in the SWRL constraints. Bossam [15] is used as the reasoning/rule engine; it incorporates both a reasoning engine and a rule engine which can make inference on ontological terms as well as evaluate logical rules. The library of BPWS4J [16] is used in constructing BPEL scripts. In the implementation, UDDI is not a mandatory component of the framework since no information from UDDI is needed for the composition; nevertheless, the framework provides a process link repository as a means to associate process specifications of the Web services to other of their information in the UDDI (if available).

IX. Discussion and Conclusion

The contribution of this paper is a Web services composition approach that is based on semantic analysis of Web service processes which characterise both functional behaviour and relevant constraints of the Web services. The approach is suitable for service designers who have knowledge about the constituent tasks within the composition. A framework is provided to support the composition by using Dijkstra's shortest path algorithm to identify the best composition and can generate a BPEL for such composition.

Similarly to other research work in semantic Web services, this work incurs overheads in publishing extra semantics-based process specifications and in learning about shared process ontologies and external ontologies. However, these overheads come with the power to automatically and semantically analyse and compose Web services. We believe that it is worth, considering standard processes and standard ontologies for vertical service domains are emerging in the near future.

The approach mainly supports service composition at design time. This is because the behaviour and process constraints of the Web services are published prior to the composition. The approach hence focuses on static behaviour of the Web services. Nevertheless, it is possible to use the proposed technique at run time. We can, for example, analyse service behaviour to find a replacement Web services when part of a business process or a composite service fails.

Our approach does not yet consider other important issues in the composition such as QoS attributes and trust between composed services. QoS attributes of Web services are static and dynamic information that represents the quality of service provision. Dynamic information is mostly related to service performance, and hence contributes to run-time composition. However, such quality attributes are still difficult to monitor, measure, and apply to service composition in practice. Trust for the composed services is also important. Two Web services with compatible behaviour may not be able to work together because they do not have mutual trust or the service designer does not trust them.

For future research, we plan to evaluate the performance of the composition framework and explore the QoS and trust issues. The approach will be applied to a fault tolerant composition framework that supports substitution and recovery of business processes of Web services.

Acknowledgment

This research is part of the Engineering New Paradigm Software for Enterprises with Service-Oriented Architecture Project, supported by Thailand's Software Industry Promotion Agency (Public Organisation).

References

[1] IBM, *BPEL: Business Process Execution Language for Web Services*, 2003, http://www.106.ibm.com/developerworks/webservice/library/ws-bpel

[2] D. Murtagh, *Automatic Web Service Composition*. Master's Thesis, Department of Computer Science, University of Dublin, 2004.

[3] OWL-S Coalition, *OWL-S 1.1 Release*, http://www.daml.org/services/owl-s/1.1/

[4] H.T. Cormen, E.C. Leiserson, L.R. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. USA: MIT Press and McGraw-Hill, 2001, pp. 595–601.

[5] L. Zeng, B. Benatallah, M. Dumas, and J. Kalagnanam, "Quality driven web service composition," in *Proc. 12th Intl. World Wide Web Conf.*, Budapest, Hungary, 2003, pp. 411-421.

[6] S. Sounsri, D. Wichadakul, and D. Savamipak, "A genetic algorithm for an automated web service composition," in *Proc. 1st National Conf. on Computing and Information Technology*, Bangkok, Thailand, 2005, pp. 322-328.

[7] R. Aggarwal and K. Verma, "Constraint driven web service composition in METEOR-S," in *Proc. IEEE International Conf. on Services Computing*, Shanghai, China, 2004, pp. 23-30.

[8] M. Klusch, A. Gerber, and M. Schmidt, "Semantic web service composition planning with OWLS-XPlan," in *Proc. 1st Intl. AAAI Fall Symp. on Agents and the Semantic Web*, Arlington, VA, USA, 2005, pp. 77-84.

[9] R. Zhang, I.B. Arpinar, and B. Aleman-Meza, "Automatic Composition of Web Service," in *Proc. Intl. Conf. on Web Services*, Las Vegas, Nevada, USA, 2003, pp. 38-41.

[10] The Protégé Ontology Editor and Knowledge Acquistion System, http://protege.stanford.edu/

[11] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*, 2003, http://daml.org/2003/11/swrl/

[12] N. Sriharee and T. Senivongse, "Matchmaking and ranking of semantic web services using integrated service profile," *Intl. J. of Metadata, Semantics and Ontologies*, vol. 1, no. 2, pp. 100-118, 2006.

[13] uddi.org, *UDDI: Universal Description, Discovery, and Integration of Web Services*, 2002, http://www.uddi.org

[14] OWL-S API, http://projects.semwebcentral.org/projects/owl-s-api/

[15] Bossam Rule/OWL Reasoner, http://mknows.etri.re.kr/bossam/

[16] BPWS4J, http://www.alphaworks.ibm.com/tech/bpws4j

ประวัติผู้เขียนวิทยานิพนธ์

นายเติมศักดิ์ วะสิโน เกิดเมื่อวันที่ 21 ธันวาคม พ.ศ. 2525 ที่จังหวัดชลบุรี สำเร็จการศึกษาระดับปริญญาบัณฑิต หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากมหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตศรีราชา เมื่อ พ.ศ. 2548 และได้เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชา วิศวกรรมคอมพิวเตอร์ ณ จุฬาลงกรณ์มหาวิทยาลัย เมื่อปี พ.ศ. 2548