การออกแบบการปฏิสัมพันธ์ระหว่างเว็บเซอร์วิสบนช่องทางการสื่อสารที่มีเสถียรภาพ

นางสาวรพีพร จินะ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาวิทยาการคอมพิวเตอร์และสารสนเทศ
บัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2549

ชื่อ : นางสาวรพีพร จินะ
ชื่อวิทยานิพนธ์ : การออกแบบการปฏิสัมพันธ์ระหว่างเว็บเซอร์วิสบนช่องทางการสื่อสารที่มีเสถียรภาพ
สาขาวิชา : วิทยาการคอมพิวเตอร์
สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ที่ปรึกษาวิทยานิพนธ์ : อาจารย์ ดร. เบญจพร ลิ้มธรรมาภรณ์
ปีการศึกษา : 2549

บทคัดย่อ

งานวิจัยนี้ได้ทำการศึกษาและวิเคราะห์การจัดการเซสชัน (Session) บนโพรโทคอลการสื่อสารแบบต่าง ๆ สำหรับการติดต่อและทำธุรกรรมร่วมกันของเว็บเซอร์วิสโดยคำนึงถึงการจัดการเซสชันที่มีเสถียรภาพ ซึ่งงานวิจัยนี้ได้ออกแบบการปฏิสัมพันธ์ระหว่างเว็บเซอร์วิสบนช่องทางการสื่อสารที่มีเสถียรภาพโดยอาศัย Session Initiation Protocol (SIP) และกลไกของฮาร์ทบีท (Heartbeat) มาทำงานร่วมกัน ในกรณีที่มีการตรวจจับได้ว่าเซิร์ฟเวอร์หลักล้มเหลว กลไกการเลือกเซิร์ฟเวอร์หลักตัวใหม่ (Leader Election) จะทำหน้าที่เลือกเซิร์ฟเวอร์ที่มีลำดับความสำคัญ (Priority) สูงที่สุดในระบบมาทำงานแทน ซึ่งกลไกนี้อาศัยพื้นฐานการทำงานของริงอัลกอริธึม (Ring Algorithm)

(วิทยานิพนธ์มีจำนวนทั้งสิ้น 48 หน้า)

คำสำคัญ : SIP, เซสชัน, การจัดการเซสชัน, เว็บเซอร์วิส

_____อาจารย์ที่ปรึกษาวิทยานิพนธ์

Name : Miss Rapeeporn Jina
Thesis Title : A Design of Multi-party Web Services on Reliability Session
Major Field : Computer Science
King Mongkut's Institute of Technology North Bangkok
Thesis Advisor : Dr. Benchaphon Limthanmaphon
Academic Year : 2006

Abstract

This thesis studies and analyzes session management of communication among web services. The focus is on reliability of session management. In this thesis a model of reliable interaction between web services via Session Initiation Protocol (SIP) and Heartbeat mechanism is proposed. When a leader server fails, a new leader server with the highest priority is elected to replace the failed node by using Ring algorithm.

(Total 48 pages)

Keywords : SIP, session, session management, web service

_____ Advisor

สารบัญ

สารบัญ (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การทำธุรกรรมผ่านเว็บเซอร์วิสที่มีการติดต่อกับเว็บเซอร์วิสมากกว่าหนึ่งตัว ทำให้เกิดการไหลเวียนทางธุรกิจ (Business Flow) ในที่นี้หมายถึงกระบวนการทางธุรกิจที่ถูกดำเนินการแบบอัตโนมัติโดยระบบคอมพิวเตอร์ ซึ่งประกอบด้วยเซสชันหรือช่องทางการแลกเปลี่ยนข้อมูลระหว่างผู้ให้บริการและผู้เรียกใช้บริการจำนวนมาก โดยเมื่อมีผู้ร้องขอติดต่อเว็บเซอร์วิสเพื่อขอใช้บริการเซสชันจะถูกกำหนดขึ้นมาเพื่อให้การดำเนินธุรกรรมสามารถดำเนินได้อย่างมีเสถียรภาพ จึงมีความจำเป็นต้องมีการดูแลเซสชันให้มีเสถียรภาพตั้งแต่เซสชันถูกกำหนดขึ้นมาจนกระทั่งเสร็จสิ้นกระบวนการในการดำเนินธุรกรรม

ด้วยเหตุข้างต้น จึงได้นำ Session Initiation Protocol (SIP) ซึ่งเป็นโพรโทคอลมาตรฐานการควบคุมการสื่อสารระดับแอพพลิเคชัน (Standard Application Layer Control Protocol) ที่ได้รับการพัฒนาโดย IETF (Internet Engineering Task Force) มาทำหน้าที่จัดการการเริ่มตั้งเซสชัน (Initiate Session) การแก้ไขเซสชัน (Modify Session) และการสิ้นสุดเซสชัน (Termination) ที่ทำงานระหว่างผู้ใช้ซึ่งในที่นี้คือไคลเอนท์และเว็บเซอร์วิส

เนื่องจากการจัดการเซสชันของเว็บเซอร์วิสโดยมากจะมีการจัดการแบบศูนย์กลาง ดังนั้นหากว่าเซิร์ฟเวอร์ซึ่งเป็นตัวจัดการกลางหลักเกิดล่มลง จะทำให้ระบบทั้งหมดไม่สามารถทำงานได้เลย ในการติดต่อกันระหว่างเว็บเซอร์วิส จึงควรที่จะมีเซสชันที่มีเสถียรภาพเพื่อไม่ให้เกิดปัญหาในการดำเนินธุรกรรม ดังนั้นการที่จะรักษาแต่ละเซสชันให้ดำเนินการต่อไปได้อย่างต่อเนื่องจะต้องมีเซิร์ฟเวอร์มาทำงานแทน ในงานวิจัยนี้จึงมีแนวทางในการนำริงอัลกอริธึมมาประยุกต์ใช้ในการเลือกเซิร์ฟเวอร์ตัวใหม่มาแทนที่ในการให้บริการหลัก

1.2. วัตถุประสงค์

งานวิจัยนี้มีวัตถุประสงค์เพื่อออกแบบระบบการปฏิสัมพันธ์ระหว่างเว็บเซอร์วิสบนช่องทางการสื่อสารที่มีเสถียรภาพ

1.3. ขอบเขตของการวิจัย

1.3.1 ออกแบบเว็บเซอร์วิสที่มีการติดต่อสื่อสารกันในลักษณะมัลติเว็บเซอร์วิส

1.3.2 ออกแบบเซิร์ฟเวอร์จำนวนหลายตัวให้ทำงานร่วมกันโดยใช้หลักการของโพรโทคอล SIP

1.3.3 ออกแบบการทำงานของริงอัลกอริธึมเพื่อใช้ในการเลือกเซิร์ฟเวอร์ที่ให้บริการหลัก

1.3.4 ออกแบบวิธีการกระจายข้อมูล Session ID ให้แก่ SIP เซิร์ฟเวอร์ทั้งหมด

1.4. วิธีการวิจัย

วิจัยเชิงเปรียบเทียบ และออกแบบโมเดลของระบบให้ตรงตามวัตถุประสงค์ และขอบเขตการวิจัย

1.5. ประโยชน์ที่ได้จากการวิจัย

สามารถออกแบบเครือข่ายการติดต่อกันระหว่างเว็บเซอร์วิสที่มีเสถียรภาพ

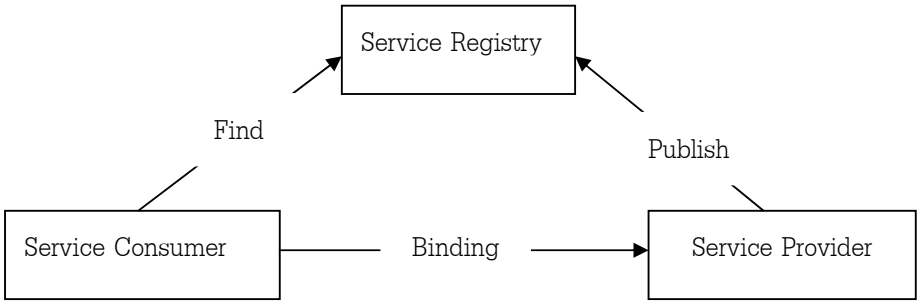
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

บทนี้จะกล่าวถึงองค์ประกอบและการทำงานของเว็บเซอร์วิส องค์ประกอบของ Simple Object Access Protocol (SOAP) และระบบพื้นฐานของ SOAP จากนั้น จะกล่าวถึงรูปแบบมาตรฐานและหลักการทำงานของ Session Initial Protocol (SIP) และสุดท้ายจะกล่าวถึงการจัดการเซสชันบนโพรโทคอลแบบต่าง ๆ ได้แก่ การจัดการเซสชันใน TCP/IP และ HTTP ประกอบด้วยการจัดการเซสชันใน Servlet การจัดการเซสชันโดยใช้ SOAP วิธีการในการจัดการเซสชันโดยใช้โพลโทคอล SIP การจัดการเซสชันในงานด้านบริการทางธุรกิจในรูปแบบของ Service Oriented Architecture (SOA)

ต่อจากนั้นจะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องในด้านของการสร้างระบบเครือข่าย และการจัดการข้อมูลที่มีเสถียรภาพบนระบบกระจายศูนย์ (Distributed System) โดยอาศัยวิธีการ และเทคนิคต่าง ๆ เช่นโพรโทคอลการจัดการเซสชันแบบมัลติคาสท์ (Multicast) และการใช้ฮาร์ทบีท (Heartbeat) รวมถึงการให้บริการทำซ้ำข้อมูลแบบกระจายในระบบธุรกิจต่อธุรกิจ (Business to Business) และอัลกอริธึมในการเลือกเซิร์ฟเวอร์หลักมาทำงานแทนเซิร์ฟเวอร์ที่ล่มไป เช่นริงอัลกอริธึม (Ring Algorithm) และบูลลีย์อัลกอริธึม (Bully Algorithm)

2.1 เทคโนโลยีเว็บเซอร์วิส

เว็บเซอร์วิส (Web Service) เป็นระบบซอฟต์แวร์ที่รองรับการทำงานร่วมกันระหว่างเครื่องคอมพิวเตอร์ที่แตกต่างกันในเครือข่าย โดยเว็บเซอร์วิสจะใช้ Web Service Description Language (WSDL) ในการอธิบายการใช้งานหรืออินเทอร์เฟส ในขณะที่ระบบอื่น ๆ สามารถเรียกใช้เว็บเซอร์วิสได้โดยอาศัยการรับส่งข้อความในรูปของ SOAP (SOAP Message) ซึ่งทำงานบนโพรโทคอล HTTP (Hypertext Transfer Protocol) [1] ในตัวของเว็บเซอร์วิสเองมีหลักการทำงานพื้นฐานมาจาก SOA [2] ซึ่งระบบซอฟต์แวร์ทั้งหมดจะถูกกระจายเสมือนเป็นชุดของบริการ ในการที่จะอนุญาตให้ระบบอื่นได้ใช้เว็บเซอร์วิสเหล่านี้จะต้องมีกลไกที่เป็นทางการในการอธิบายบริการ การค้นหา และการเรียกใช้บริการ จากภาพที่ 2-1 แสดงถึงการทำงานพื้นฐาน 3 ส่วน และการทำงานร่วมกันระหว่างส่วนต่าง ๆ เหล่านี้ ซึ่งประกอบด้วยผู้ให้บริการ (Service Provider) ผู้บริการจดทะเบียน (Service Registry) และผู้เรียกใช้บริการ (Service Consumer)

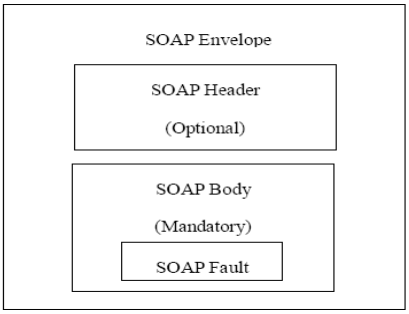


ภาพที่ 2-1 รูปแบบการจัดการร่วมกันในสถาปัตยกรรมแบบ Service Oriented Architecture

ในส่วนของการทำงานหลักของผู้ให้บริการ จะต้องอธิบายการอินเทอร์เฟสหรือฟังก์ชันในการให้บริการโดยมีการจดทะเบียนการให้บริการที่ผู้บริการจดทะเบียน ผู้เรียกใช้บริการสามารถทำการค้นหาบริการต่าง ๆ จากผู้บริการจดทะเบียนได้ [4] โดยที่จะมองผู้บริการจดทะเบียนเสมือนเป็น Web Services Library ซึ่งผู้บริการจดทะเบียนประกาศ (Publish) เว็บเซอร์วิสทั้งหมดที่ขึ้นทะเบียนไว้ ผู้เรียกใช้บริการสามารถค้นหาบริการ (Find) เว็บเซอร์วิสที่ต้องการ และจะทำการติดต่อ (Binding) และเรียกใช้เว็บเซอร์วิสที่ต้องการได้จาก WSDL ผู้เรียกใช้บริการจะทราบถึงชื่อเมธอด ชนิดของข้อมูล พารามิเตอร์ และทรานสปอร์ตโพรโทคอลในการเรียกใช้เมธอด ต่อจากนั้นจะใช้ข้อมูลนี้ในการเรียกใช้เว็บเซอร์วิส เทคโนโลยีที่รองรับการทำงานของส่วนพื้นฐานสามข้อนี้ได้แก่ UDDI WSDL และ SOAP

2.1.1 Simple Object Access Protocol (SOAP)

SOAP เป็นโพรโทคอลที่มีพื้นฐานอยู่ในรูป XML ที่นำมาใช้ในการแลกเปลี่ยนข้อมูลในการอธิบายเมธอด และค่าพารามิเตอร์เพื่อสร้างการเรียกใช้ในระยะไกลข้ามเครือข่ายผ่านทางโพรโทคอล HTTP ดังนั้นแอพพลิเคชันต่าง ๆ ที่ทำงานบนระบบปฏิบัติการและโปรแกรมภาษาที่แตกต่างกัน จึงสามารถทำงานร่วมกันได้โดยอาศัยการแลกเปลี่ยนข้อมูลในรูปของ SOAP ซึ่งโครงสร้างของ SOAP Message แสดงดังภาพที่ 2-2



ภาพที่ 2-2 โครงสร้างของ SOAP Message

2.1.1.1 SOAP Envelope

เป็นตัวกำหนดโครงร่างทั้งหมดของ SOAP Message และสร้างองค์ประกอบต่าง ๆ ที่เป็นรากฐานของแพ็กเก็ต (Packet) ที่ถูกส่งออกไป SOAP Envelope ยังทำหน้าที่ห่อหุ้ม SOAP Header และ SOAP Body ของ SOAP Message ด้วย

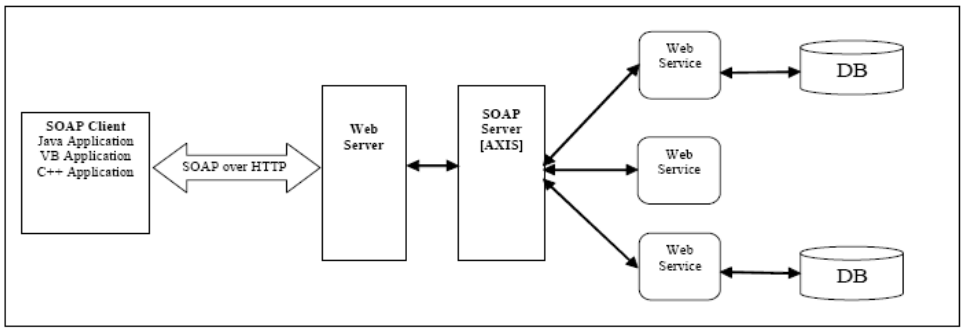
2.1.1.2 SOAP Header

ส่วนประกอบต่าง ๆ ของ SOAP Header จะมีความสัมพันธ์กับส่วนประกอบต่าง ๆ ใน SOAP Body โดยทำการรองรับข้อมูลที่อยู่ใน SOAP Body และเนื่องจาก SOAP ไม่มีกฎตายตัวในเรื่องของชนิดข้อมูลที่ถูกใส่ไว้ใน SOAP Header ซึ่งเป็นข้อดีที่สามารถใส่ข้อมูลที่ต้องการได้เช่น Session ID, User Name และ รหัสผ่านของผู้ส่ง ข้อมูลที่นำมาใช้ในการพิสูจน์ตน (Authentication) หรือข้อมูลสำหรับจัดการทรานแซคชัน (Transaction) เป็นต้น

2.1.1.3 SOAP Body

เป็นที่บรรจุการกระทำทั้งหมด รวมถึงข้อมูลข่าวสารถูกส่งผ่านไปยังเว็บเซอร์วิส กล่าวคือเป็นส่วนที่ทำการแลกเปลี่ยนข้อมูลกับผู้รับ SOAP Message นั่นเอง และบางครั้งอาจจะมีส่วนของ SOAP Fault ที่ถูกนำมาใช้ในการนำส่งข้อผิดพลาดที่เกิดขึ้นและข้อมูลเกี่ยวกับสถานะ

ภาพที่ 2-3 แสดงระบบพื้นฐานที่สำคัญของ SOAP ซึ่งมีทั้งหมดอยู่สี่ส่วนหลัก ๆ ได้แก่ SOAP Client , SOAP Server, เว็บเซิร์ฟเวอร์ (Web Server) และเว็บเซอร์วิส (Web Service)



ภาพที่ 2-3 ระบบพื้นฐานของ SOAP (อ้างอิงจาก [24])

สามารถอธิบายการทำงานของแต่ละส่วนได้ดังต่อไปนี้

ก) SOAP Client

ทำหน้าที่ส่ง SOAP Request และรับ SOAP Response ผ่านโพรโทคอล HTTP เว็บเซิร์ฟเวอร์ทำหน้าที่รับ SOAP Request และตอบกลับไปยัง SOAP Client

ข) SOAP Server

ทำหน้าที่วิเคราะห์ SOAP Request Message และเรียกเว็บเซอร์วิสที่ตรงตามความต้องการพร้อมทั้งสร้าง SOAP Response Message ไปยังไคลเอนท์

ค) Web Service

เว็บเซอร์วิส [3, 5] ทำหน้าที่ให้บริการแก่ไคลเอนท์ซึ่งหน้าที่ของบริการขึ้นอยู่กับความต้องการทางธุรกิจ เว็บเซอร์วิสอาจจะมีการติดต่อเรียกใช้ฐานข้อมูลไว้รองรับการบริการเพื่อเพิ่มศักยภาพในการให้บริการ

2.1.2 Universal Description, Discovery and Integration (UDDI)

ทำงานเป็นผู้บริการจดทะเบียนในสถาปัตยกรรม SOA โดยจะระบุถึงบริการที่ผู้ให้บริการเว็บเซอร์วิสได้ลงทะเบียนข้อมูลการให้บริการ เพื่อให้ผู้เรียกใช้บริการสามารถค้นหาบริการที่ได้ลงทะเบียนไว้แล้ว และยอมให้ใช้บริการนั้นได้ [4] หรือกล่าวได้ว่า UDDI มีกลไกในการทำการลงทะเบียนแก่เว็บเซอร์วิสที่กระจายทั่วไปในอินเทอร์เน็ต และให้บริการในการค้นหาแก่ผู้เรียกใช้บริการ ซึ่งจะเห็นได้ว่าหน้าที่ของ UDDI จะคล้ายกับการทำ Naming Services ของ CORBA และ RMI (Remote Method Invocation) [5,18]

2.1.3 Web Services Description Language (WSDL)

มีไว้สำหรับอธิบายการเรียกใช้เว็บเซอร์วิส ซึ่งอยู่ในรูปของ XML ในการอธิบายเว็บเซอร์วิส

ที่เป็นข้อมูลต่าง ๆ ของการสื่อสารระหว่างผู้สื่อสารปลายทาง (End Points) ที่ซึ่งสามารถแลกเปลี่ยนข้อความที่แน่นอนได้ [21]

2.2 Session Initial Protocol (SIP)

อินเทอร์เน็ตมีแอพพลิเคชันจำนวนมากที่ต้องการสร้างและจัดการเซสชัน ในที่นี้เซสชันหมายถึงช่องทางการแลกเปลี่ยนข้อมูลระหว่างผู้ให้บริการและผู้เรียกใช้บริการ การนำแอพพลิเคชันเหล่านี้มาใช้เป็นเรื่องยากในทางปฏิบัติ เนื่องจากผู้ใช้บริการมีการเคลื่อนที่ไปมาอยู่เสมอ และมีข้อมูลการสื่อสารในหลายรูปแบบ และในบางครั้งการสร้างเซสชันอาจเกิดขึ้นพร้อมกันได้ โพรโทคอล SIP สามารถจัดการเซสชันในการส่งข้อมูลที่เป็นสื่อประสม (Multimedia) แบบ Real Time ได้หลายรูปแบบเช่น เสียง ภาพ วิดีโอ หรือ ข้อความพิมพ์ [7, 8, 16]

โพรโทคอล SIP ได้รับการสร้างและพัฒนาโดย Internet Engineering Task Force (IETF) ซึ่งสามารถทำงานได้สอดคล้องกับโพรโทคอลต่าง ๆ ได้โดยการให้ Internet Endpoints (ต่อไปนี้จะเรียกว่า User Agent) ค้นหาตำแหน่งของผู้ติดต่อ และยินยอมให้มีการแสดงคุณลักษณะเฉพาะของเซสชันที่ User Agent เหล่านั้นต้องการร่วมใช้ ในส่วนของการกำหนดผู้เข้าร่วมเซสชัน และหน้าที่อื่น ๆ ของ SIP คือจะสร้างแม่ข่ายของเครือข่าย (Network Hosts) ต่อไปนี้จะเรียกว่าพร็อกซีเซิร์ฟเวอร์ (Proxy Servers) ซึ่ง User Agents สามารถทำการลงทะเบียน (Registrations) การเชิญ (Invitations) และตอบรับ Requests อื่น ๆ ได้

โพรโทคอล SIP เป็นโพรโทคอลที่มีความรวดเร็วในการสร้าง การปรับเปลี่ยน และจบการทำเซสชัน โดยมีความเป็นอิสระตามโพรโทคอลของการส่งข้อมูล และไม่ขึ้นกับชนิดของเซสชันที่ได้เริ่มสร้างขึ้น ทั้งนี้ โพรโทคอล SIP ยังสามารถทำงานร่วมกันกับ IPv4 และ IPv6 ได้ [8, 10,11]

2.2.1 ลักษณะโดยทั่วไปในการทำงานของ SIP

โพรโทคอล SIP ทำงานในระดับแอพพลิเคชันของชั้นการสื่อสารตามมาตรฐานเครือข่าย ISO-OSI ที่ควบคุมโพรโทคอลให้เริ่มสร้าง ปรับเปลี่ยน และ จบเซสชัน ตัวอย่างการใช้งานเช่นการทำ Internet Telephony Calls เป็นต้น โพรโทคอล SIP ยังสามารถทำ Multicast Conferences และยังสามารถเพิ่ม และลบสื่อ (Media) ในระบบได้

โพรโทคอล SIP มีหน้าที่การทำงานหลักสำหรับการสื่อสารระหว่างผู้ใช้ปลายทางสองฝั่งอยู่ 5 ประการได้แก่

1. ค้นหาตำแหน่งของผู้เรียกและผู้ถูกเรียกในการสื่อสาร
2. SIP สามารถกำหนดค่าที่เหมาะสมในการสื่อสารของ User Agent

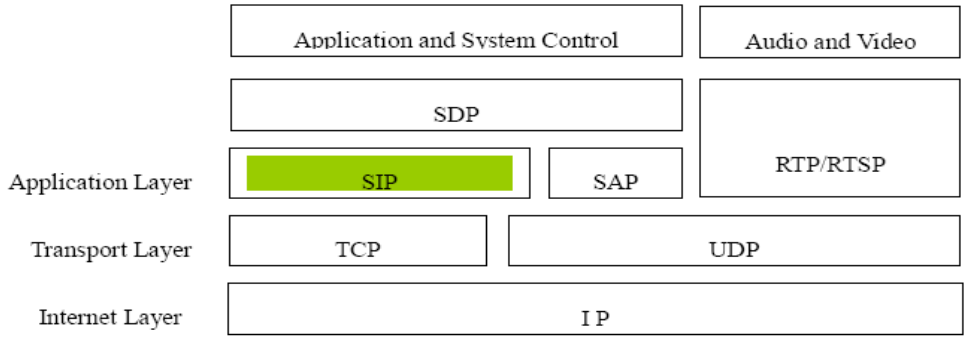
3. SIP สามารถค้นหาการสื่อสารในรูปแบบสื่อและค่าพารามิเตอร์ของสื่อนั้นได้

4. ตั้งค่าพารามิเตอร์ของเซสชันให้แก่ผู้เรียก และผู้ถูกเรียก

5. จัดการเซสชันรวมถึงการส่ง (Transfer) การสิ้นสุด (Termination) เซสชัน และการปรับเปลี่ยน (Modifying) ค่าพารามิเตอร์ของเซสชันและการเรียกใช้บริการ

2.2.2 SIP Protocol Stack

ภาพที่ 2-4 แสดงถึงลำดับชั้นโพรโทคอลการสื่อสารของ SIP (SIP Protocol Stack) โดยปกติถ้าหากเป็นการสื่อสารแบบสื่อประสม SIP จะใส่ Session Description Protocol (SDP) ไว้ใน Body ของตัวมันเองและใช้ TCP หรือ UDP (User Datagram Protocol) [9] เพื่อส่งข้อความผ่านอินเทอร์เน็ต ในปัจจุบัน SIP ยังสามารถใช้ TLS โดยผ่าน TCP ในการเข้ารหัสการนำส่ง (Encrypted Transport) พร้อมด้วยความสามารถในการทำการพิสูจน์ตน (Authentication)



ภาพที่ 2-4 SIP Protocol Stack (อ้างอิงจาก [24])

2.2.3 สถาปัตยกรรมของ SIP

มีสองส่วนพื้นฐานหลัก ๆ ดังต่อไปนี้

2.2.3.1 SIP User Agent (UA)

เป็นผู้สื่อสารปลายทาง ซึ่งอาจเป็นอุปกรณ์ฮาร์ดแวร์หรือซอฟต์แวร์ที่นำมาใช้ร่วมกันกับโพรโทคอล SIP (ตัวอย่างเช่น IP Phone) ซึ่งประกอบด้วยสองส่วนหลักดังต่อไปนี้

1. User Agent Client (UAC) เป็นแอพพลิเคชันบนฝั่งไคลเอนท์ โดยจะเป็นผู้สร้าง SIP Request

2. User Agent Server (UAS) เป็นแอพพลิเคชันบนฝั่งเซิร์ฟเวอร์ที่ตอบสนองต่อ SIP Request จาก UAC

2.2.3.2 Network Server

เป็นส่วนจับสัญญาณที่มีการเรียกจากคู่สายจำนวนมาก และมาจากที่ต่าง ๆ กัน (Multiple Calls) การจำแนกชื่อ (Name Resolution) และการกำหนดตำแหน่งของผู้เรียกและผู้ถูกเรียก (User Location) ซึ่งประกอบด้วยสามส่วนหลัก ๆ ดังต่อไปนี้

ก) SIP Register Server

รับข้อความการลงทะเบียนจากผู้สื่อสารปลายทาง (End Point) โดยยึดกับตำแหน่งของผู้เรียกและผู้ถูกเรียกในขณะนั้น และจับคู่ที่ตรงกันของ SIP Address ในโดเมนของผู้สื่อสารปลายทาง ข้อมูลการจับคู่เหล่านั้นจะจัดเก็บไว้ในฐานข้อมูลซึ่งอาจจะอยู่ในเครื่องเดียวกันหรืออยู่ต่างเครื่องที่ไกลกันก็ได้

ข) SIP Proxy Server

ทำหน้าที่ส่ง SIP Message ต่อไปยังพร็อกซีเซิร์ฟเวอร์ต่าง ๆ โดยสร้างในลักษณะของทรี เพื่อให้ SIP Message ถึงจุดหมายที่ต้องการ โดยมีโหมดของการทำงานที่ต่างกันอยู่สองอย่างสำหรับเซิร์ฟเวอร์ชนิดนี้ได้แก่ Stateless คือเซิร์ฟเวอร์จะไม่จดจำข้อมูลทั้งหมดในแต่ละ Request ที่มีการติดต่อกับเซิร์ฟเวอร์ และ Stateful คือเซิร์ฟเวอร์จะเก็บข้อมูลการค้นหาเส้นทางก่อนหน้าและสามารถใช้ข้อมูลเหล่านี้สำหรับพัฒนาการส่งข้อมูลได้

ค) SIP Redirect Server

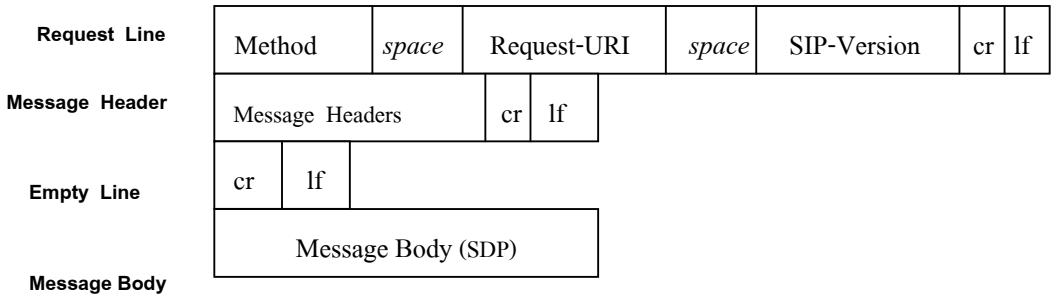
เป็นส่วนที่ยอมให้ SIP พร็อกซีเซิร์ฟเวอร์ กำหนดตำแหน่งของผู้ถูกเรียกซึ่งอยู่ต่างโดเมนได้

2.2.4 SIP Messages

SIP เป็นโพรโทคอลที่มีพื้นฐานเป็นข้อความซึ่งหมายถึงผู้ใช้สามารถอ่าน SIP Message ได้โดยตรง และสามารถเพิ่มเติมคุณลักษณะใหม่ ๆ ได้ง่าย [10] โดยแท้จริงแล้ว SIP มีพื้นฐานมาจาก HTTP Request และ HTTP Response ดังนั้นข้อความที่ส่งจึงแบ่งได้เป็น SIP Request และ SIP Response [8]

2.2.4.1 SIP Request Messages

ประกอบด้วย Request Line, Message Header, Empty Line และ Message Body ซึ่งรูปแบบของ SIP Request Messages แสดงไว้ในภาพที่ 2-5



ภาพที่ 2-5 รูปแบบของ SIP request message (อ้างอิงจาก [24])

ในส่วนของเมธอดซึ่งผู้เรียกจะต้องใช้ในการส่งข้อความโดยพื้นฐานแล้วมีอยู่ 6 เมธอด โดยหน้าที่ และชนิดต่าง ๆ ของเมธอดได้แสดงไว้ในตารางที่ 2-1

ตารางที่ 2-1 ตัวอย่างของ Request Method

INVITE	เมธอดนี้หมายถึงผู้ใช้ (User) หรือบริการ (Service) ได้รับเชิญให้เข้าร่วมในเซสชัน ในส่วนของ Message Body จะมีการอธิบายเซสชันให้กับผู้ถูกเรียก ซึ่งเมธอดนี้นำมาใช้ร่วมกับ Proxy Server, Redirect Server, User Agent Servers (UAS) และ User Agent Clients (UAC)
ACK	ตอบรับว่าไคลเอนท์ได้รับข้อความล่าสุดที่ตอบรับการส่ง INVITE แล้ว สามารถใส่ข้อความเพิ่มใน Message Body ด้วย เป็นการอธิบายเซสชันล่าสุดให้กับผู้ถูกเรียกได้นำข้อมูลไปใช้ เมธอดนี้นำไปใช้ได้กับ SIP Proxy Server , Redirect Server, UAS และ UAC
BYE	UAC จะใช้เมธอดนี้เพื่อแสดงให้เซิร์ฟเวอร์ทราบว่าต้องการยกเลิกการเรียก ซึ่งอาจจะต้องส่งต่อไปเหมือนกับ INVITE และผู้เรียกหรือผู้ถูกเรียกอาจจะเป็นผู้ใช้เมธอดนี้ซึ่งทั้ง Proxy Server , Redirect Server และ UAS รองรับเมธอดนี้
CANCEL	เมธอดนี้ ถูกนำมาใช้เพื่อยกเลิก Request ที่กำลังกระทำอยู่ โดย ยังคงใช้ค่า Call-ID, To, From และค่า Cseq Header Fields เดิมแต่จะไม่มีผลต่อ Request ที่เสร็จสิ้นแล้ว หรือ Request ที่ได้ทำไปแล้ว หรือ Response ที่ได้ส่งตอบกลับมา เมธอดนี้ใช้กับ Proxy Servers และ SIP server ทุกชนิด
REGISTER	เมธอดนี้ใช้โดยไคลเอนท์ในการลงทะเบียนตำแหน่ง (Address) ที่ทำรายการไว้ใน Header Field ใน SIP Server
OPTIONS	เมธอดนี้เซิร์ฟเวอร์ จะถูก Query ตามความสามารถของเซิร์ฟเวอร์และนำไป ใช้กับ SIP Proxy, Redirect Server , UAC และ UAS

อ้างอิงจาก [8]

จาก SIP Request Method ดังกล่าวข้างต้น จะมี Requesting-URI ที่แสดงถึงเป้าหมาย URL ของ Request นี้ ซึ่ง URI Syntax ประกอบด้วย ";", ":", "@", "?", และ "/" ตัวอย่างเช่น <Method.>@<Host>:<Port><SIP Version> ดังนั้น Sip Request Message ที่จะส่งออกไปจะเป็นดังนี้

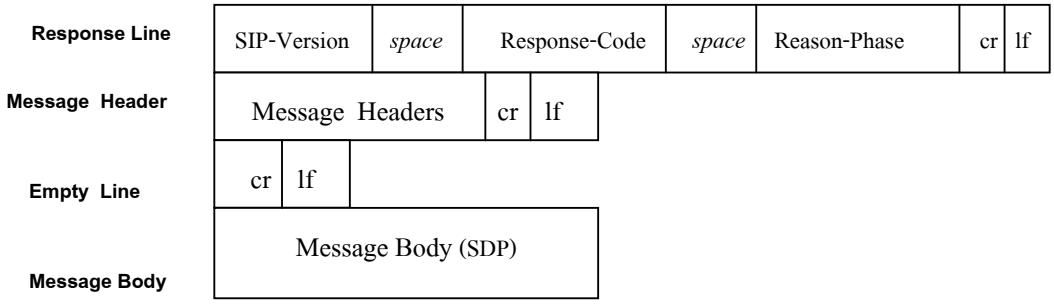
Method** space **@ UAC IP Address** space **SIP Version

SIP-Version จะต้องใส่ทุกครั้งเพื่อจะได้ตรวจสอบว่าผู้ส่งและผู้รับใช้เวอร์ชันเดียวกันอยู่ และหากผู้ส่งและผู้รับใช้ต่างเวอร์ชันกันก็จะไม่สามารถสื่อสารกันได้

ในส่วนของ Message Body ซึ่งได้แสดงในรูปแบบของ Session Description Protocol (SDP) [22] ใช้อธิบายชนิดของสื่อประสมในการทำเซสชัน ในกรณีที่เป็นแค่การส่งข้อความอาจไม่ต้องใส่ Message Body ก็ได้

2.2.4.2 SIP Response Messages

เป็นข้อความที่เซิร์ฟเวอร์ตอบกลับมายังฝั่งไคลเอนท์ ซึ่งรูปแบบของ SIP Response Message แสดงดังภาพที่ 2-6



ภาพที่ 2-6 รูปแบบของ SIP Response Message (อ้างอิงจาก [24])

Response Line ประกอบด้วย SIP-Version, Response-Code และ Reason-Phase จากนั้นจะเป็นส่วนของ Message Header และ Message Body รูปแบบข้อความของ Response Line เป็นดังนี้

SIP-Version space **Response-Code** space **Reason-Phrase** CRLF

Reason-Phrase จะอธิบาย Response-Code (บางครั้งอาจเรียกว่า Status-Code) โดยใช้คำอธิบายสั้นๆ ซึ่งอาจจะไม่ใส่ก็ได้ ในส่วนของ Response-Code นั้นเป็นตัวเลข 3 หลักที่แสดงถึงผลลัพธ์ของการทำความเข้าใจและความแน่ใจใน Request ที่เข้ามา ตัวเลขแรกสุดของ Response-

Code เป็นการกำหนดคลาสของ Response ใน SIP ซึ่งภาพที่ 2-7 แสดงถึงชนิดของ SIP Response Message

SIP Response ทั้งหมดแบ่งได้เป็น 6 ชนิด ซึ่งจะอธิบายความหมายของ Response-Code ทั้ง 6 ชนิดได้ดังต่อไปนี้

1xx: Informational หมายถึงได้รับ Request แล้วและกำลังจะดำเนินการ

2xx: Success การดำเนินการสำเร็จ ได้ทำความเข้าใจและยอมรับแล้ว

3xx: Redirection ต้องการการกระทำนอกเหนือจากนี้เพื่อให้ Request สมบูรณ์

4xx: Client Error Request มี Bad Syntax หรือไม่สามารถทำให้สำเร็จได้ที่เซิร์ฟเวอร์นี้

ภาพที่ 2-7 ชนิดของ SIP Response Message (อ้างอิงจาก [34])

5xx: Server Error เซิร์ฟเวอร์ล้มเหลวในการทำ Valid Request ให้สำเร็จ

6xx: Global Failure ไม่สามารถทำ Request ให้สำเร็จไม่ว่าจะเป็นที่เซิร์ฟเวอร์ตัวใดในระบบ

SIP Response Message ขึ้นอยู่กับชนิดของ Response-Code ทั้ง 6 ชนิดนี้ นอกจากนี้ยังสามารถเพิ่มเติมได้อีกตามความจำเป็น แต่แอพพลิเคชันจะต้องเข้าใจคลาสของResponse-Codeเหล่านี้

2.2.5 HEADER FIELDS

เป็นส่วนที่จะต้องมีใน SIP Requrest Message และ SIP Response Message โดยปกติที่มักใช้ประจำคือ CALL-ID, CSEQ, FROM, TO, VIA, CONTENT-TYPE และ CONTENT-Length

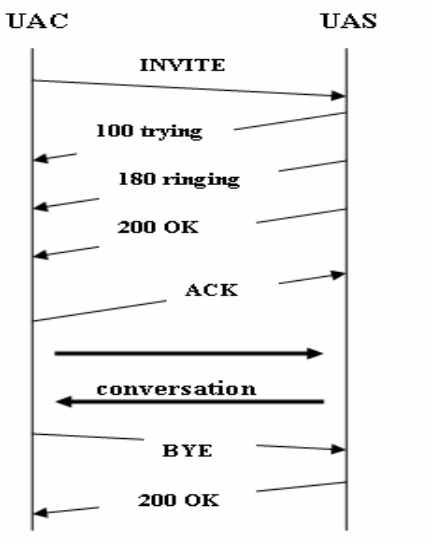
ใน SIP Proxy Server สามารถเพิ่ม Via field แต่ไม่สามารถจัดลำดับ Header ใหม่ (Reorder Headers) ที่มีอยู่ได้ SIP Message Headers [8] ตารางที่ 2-2 แสดงเซ็ตของ Headers ดังกล่าวพร้อมการอธิบายในแต่ละ Header

ตารางที่ 2-2 ตัวอย่างของ Header Fields

Headers	Descriptions
CALL-ID	Uniquely identify a call between two user agents
CSEQ	Contain a decimal number which increase for each request
FROM	Indicate the originator of the request
TO	Indicate the recipient of the request
VIA	Record the SIP request and response routes
CONTENT-TYPE	Specify the Internet media type in the SIP message body
CONTENT-Length	Indicate the number of octets in the SIP message body

อ้างอิงจาก [24]

ตัวอย่างการติดต่อสื่อสารและการสนทนาระหว่าง UAC และ UAS ได้แสดงดังภาพที่ 2-8



ภาพที่ 2-8 รูปแบบการติดต่อของ Peer To Peer SIP Session (อ้างอิงจาก [7])

2.2.6 SIP MESSAGE BODY

Requests ทั้งหมดที่เป็นการทำสื่อประสม (Multimedia) อาจจะมี Message Body ยกเว้น BYE Request สำหรับ INVITE, ACK และ OPTIONS Request จำเป็นต้องอธิบายเซสชันเสมอ ส่วนมากแล้วจะตามด้วย Session Description Protocol (SDP) ซึ่งทั้ง SIP และ SDP เป็นตระกูลโพรโทคอลเดียวกัน ความยาวของ SIP Message Body ควรใส่ไว้ใน *Content-Length* Header Field และ *Content-Type* Header Field ต้องใส่ชนิดของข้อความด้วย หากว่าข้อความได้รับการเข้ารหัสก็จะต้องใส่ไว้ใน *Content-Encoding* Header Field

2.2.7 Session Description Protocol (SDP) [22]

SDP เป็นโพรโทคอลที่ใช้สำหรับทำการตกลงการติดต่อด้วยสื่อประสม ในตารางที่ 2-3 แสดงชนิดของฟิลด์ของ SDP โดย SDP จะถูกใช้ก่อนขั้นตอนการทำ Media & Control Channel การทำงานของ SIP ในระบบ IP สำหรับ Telephony และ SDP มีรายละเอียดคร่าว ๆ โดยสรุปได้ดังนี้

1. ใช้กำหนดการติดต่อด้วยสื่อประสม
2. ถูกบรรจุอยู่ใน Body Message ของ SIP Message
3. มีลักษณะเป็น Text-based
4. กำหนดโดย RFC 232

ตารางที่ 2-3 ชนิดของฟิลด์ใน SDP

Field	Description	Mandatory
v=	Protocol version	Yes
o=	Owner/creator and session identifier	Yes
s=	Session name	Yes
i=	Session information	No
u=	URI of description	No
e=	Email address	No
p=	Phone number	No
c=	Connection information	No
b=	Bandwidth information	No
z=	Time zone adjustments	No
k=	Encryption key	No
a=	Attribute lines	No
t=	Time the session is active	Yes
r=	Zero or more repeat times	No
m=	Media information	Yes
a=	Media attributes	No

อ้างอิงจาก [24]

2.3 การจัดการเซสชัน

นิยามของเซสชันในที่นี้หมายถึง การติดต่อกันแบบมีสถานะระหว่างสองฝั่งในช่วงของหนึ่งการสื่อสาร [30] ซึ่งการจัดการเซสชันควรรวมถึงขั้นตอนการเริ่มต้น (Starting) การสื่อสารกัน (Joining) การสิ้นสุดการสื่อสาร (Terminating) และการอ่านสถานะ (Browsing) ของเซสชันบนเครือข่าย [31] โดยจะมีการพิจารณาวิธีหลักอยู่สองอย่าง ที่จะนำมาใช้ในการแก้ปัญหาเซสชันนี้ อย่างแรกคือ วิธีการจัดการเซสชันจำนวนมากบนฝั่งเซิร์ฟเวอร์ อีกวิธีหนึ่งคือ วิธีการเก็บสถานะระหว่างไคลเอนท์และเซิร์ฟเวอร์ในช่วงของการสื่อสาร

การเก็บสถานะเซสชันนี้เป็นส่วนสำคัญสำหรับการสื่อสารในเครือข่ายแบบ OSI TCP ซึ่งอยู่ในชั้นของทรานสปอร์ต จะจัดตั้งการจัดการเซสชันระหว่างคอมพิวเตอร์สองเครื่องผ่านพอร์ตที่ได้กำหนดไว้ทั้งสองเครื่อง [31] ซึ่งในความเป็นจริง TCP ใช้คู่ของซ็อคเก็ตในการเริ่มการติดต่อระหว่างสองเครื่องในเครือข่าย หลังจากที่ซ็อคเก็ตของทั้งสองเครื่องนั้นได้จัดตั้งการสื่อสารแล้ว กระบวนการโต้ตอบกันจะเป็นการรักษาสถานะคู่การสื่อสารของคอมพิวเตอร์ทั้งสองเครื่อง [15]

หัวข้อต่อไปนี้จะกล่าวถึงวิธีการจัดการเซสชันแบบดั้งเดิมได้แก่การจัดการเซสชันใน TCP/IP HTTP Servlet และการจัดการเซสชันใน HTML นอกจากนั้นจะกล่าวถึงการจัดการเซสชันวิธีใหม่โดยจะจัดการบน Soap Message ได้แก่ การรวมการจัดการเซสชันไว้ในเว็บเซิร์ฟเวอร์ การเพิ่ม State Context ไว้ใน SOAP Header การจัดการเซสชันใน HTML การจัดการเซสชันโดยใช้โพรโทคอล SIP และสุดท้ายจะกล่าวถึงโพรโทคอลการจัดการเซสชันแบบมัลติคาสท์

2.3.1 การจัดการเซสชันบนโพรโทคอล TCP/IP และ HTTP

ในลำดับชั้นของเครือข่ายมาตรฐาน (Networking Stack) HTTP อยู่ในชั้นของแอพพลิเคชันและอยู่ด้านบนของ TCP แต่เนื่องจาก HTTP เป็นโพรโทคอลแบบไม่มีสถานะ เว็บเซิร์ฟเวอร์จะไม่เก็บสถานะของไคลเอนท์ หรือข้อมูลส่วนตัวของผู้ใช้ ตัวอย่างเช่น เมื่อเว็บเซิร์ฟเวอร์รับ HTTP Request จะไม่รู้ว่า Request นี้ได้ส่งมาโดยผู้ใช้คนใหม่หรือคนก่อนหน้า ซึ่งแอพพลิเคชันของอิเล็กทรอนิกส์พาณิชย์ธุรกรรมถือว่า การเก็บรักษาข้อมูลของไคลเอนท์เป็นสิ่งสำคัญ จึงได้มีการนำ CGI (Common Gateway Interface) [9] และ Servlet มาใช้เพื่อเก็บรักษาข้อมูลเซสชันสำหรับผู้ใช้ในช่วงของการทำ Request

ในช่วงแรกของการพัฒนา World Wide Web เว็บเซิร์ฟเวอร์จะแยกการจัดการ Request ต่าง ๆ จากบราวเซอร์ที่ต่างกันด้วยวิธีการสร้าง Response Pages แบบไดนามิกส์ (Dynamic) ซึ่ง CGI ก็ใช้วิธีการแยกการจัดการ request เช่นเดียวกัน โดยจะอ่านข้อมูลจาก HTTP Request และเขียนข้อมูลไปยัง HTTP Response [6] CGI จะสร้างไฟล์เพื่อเก็บข้อมูลเซสชัน และ ID Number ที่ตอบสนองกับผู้ใช้เซสชันนั้น ซึ่งข้อมูลนี้จะเป็นส่วนหนึ่งของ URL (Uniform Resource Locator) [18] แต่อย่างไรก็

ตามวิธีการดังกล่าวนี้ขึ้นอยู่กับชนิดของแพลตฟอร์ม และยังมีค่าใช้จ่ายสูงในส่วนของโพรเซสเซอร์และหน่วยความจำ [9]

2.3.1.1 การจัดการเซสชันบน Servlet

เป็นเทคโนโลยีหนึ่งที่สามารถนำมาใช้ประโยชน์สำหรับการสร้างเว็บเพจชนิดไดนามิกส์ โดยใช้ HTTP Request การจัดการเซสชันก็จะทำในรูปของกลุ่มของคลาสซึ่งจะเรียกว่า Session Tracking API [19] เมื่อบราวเซอร์ส่ง Request ไปยังเว็บเซิร์ฟเวอร์เป็นการเรียกใช้ Servlet เพื่อสร้าง HttpSession Object และ Cookie Object โดยที่ Cookie Object จะถูกส่งไปยังบราวเซอร์ โดยผ่าน HTTP Header และจัดเก็บไว้บนฝั่งของไคลเอนท์ และสามารถใช้ Cookie Object เพื่อจัดเก็บสถานะและข้อมูลของผู้ใช้ เช่นข้อมูลของการซื้อ (Shopping Cart) เป็นต้น HttpSession Object ได้รับการจัดเก็บไว้ทางฝั่งไคลเอนท์ และนำมาใช้สำหรับเก็บและดึงข้อมูลจากฝั่งไคลเอนท์ Cookie Object ทุกตัวจะมี Session Identifier ซึ่งเชื่อมโยงกับ HttpSession Object โดยที่ Servlet ใช้ Cookie Identifier เพื่อติดตามผู้ใช้เว็บ และเพื่อค้นหาการตอบสนอง HttpSession Objects บนเซิร์ฟเวอร์ ซึ่งวิธีนี้ Servlet จะสามารถจัดการเซสชันให้กับเว็บแอพพลิเคชันได้

เทคนิคการทำ Cookie [11, 14] คือ โครงสร้างข้อมูลขนาดเล็กที่ถูกส่งมาจากเว็บเซิร์ฟเวอร์ไปยังบราวเซอร์และได้จัดเก็บไว้บนฮาร์ดดิสก์ในรูปของไฟล์ข้อความ ซึ่งประกอบด้วยตัวอักษรและตัวเลข ซึ่งเป็นข้อมูลของผู้ใช้นั้น ๆ เนื่องจากเป็นวิธีที่สะดวกในการนำไปใช้ในการติดตามเซสชัน ดังนั้นการใช้ Cookie จึงถูกนำมาใช้อย่างกว้างขวาง

อย่างไรก็ตาม ถึงแม้ว่าเทคนิคการใช้ Cookie จะได้รับการรับรองจากบราวเซอร์ส่วนมาก แต่พบว่ายังมีข้อจำกัดที่สำคัญอยู่ อันดับแรกคือ Cookie จะไม่สร้างความปลอดภัยให้กับผู้ใช้กรณีเมื่อมีการซื้อสินค้าทางอินเทอร์เน็ต ตัวอย่างเช่น เมื่อลูกค้าซื้อสินค้าผ่านอินเทอร์เน็ตก็จะมี Unique Session ระหว่างบราวเซอร์ และ เว็บเซิร์ฟเวอร์ ซึ่งได้มีการบันทึกไว้โดยเว็บเซสชัน จากข้อมูล Session ID และ Session Object นี้เว็บมาสเตอร์จะทราบว่าลูกค้าได้ซื้ออะไรไปและข้อมูลส่วนตัวของผู้ซื้อจะถูกนำมาไว้ใน Credit Number ด้วยหากว่าเว็บไซต์ได้ทิ้ง Cookie ไว้บนคอมพิวเตอร์ของลูกค้าก็จะสามารถพบข้อมูล Session ID ได้ในไฟล์ Cookie เนื่องจาก Cookie เป็นแค่ไฟล์ข้อความจึงสามารถอ่านได้ง่ายเพียงแค่ใช้ Text Editor ดังนั้นการจารกรรมข้อมูลจากไฟล์ Cookie จึงเป็นไปได้โดยง่าย นอกจากนั้นผู้บุกรุกยังสามารถเปลี่ยนแปลงข้อมูลภายในไฟล์ Cookie ได้หากว่าผู้บุกรุกใช้พร็อกซีเซิร์ฟเวอร์เป็นตัวกลางระหว่างเว็บเซิร์ฟเวอร์และบราวเซอร์ ก็จะสามารถปรับเปลี่ยนข้อมูลได้ ซึ่งในความจริงแล้ว Hyperlink ใด ๆ บนเว็บเพจอาจนำมาต่อกับพร็อกซีเซิร์ฟเวอร์ของผู้บุกรุกได้ ซึ่งผู้ใช้ไม่สามารถแยกได้ว่าอันไหนเป็นเว็บไซต์จริงหรือเป็นพร็อกซีเซิร์ฟเวอร์ของผู้บุกรุก ข้อเสียที่สองคือผู้ใช้จะไม่ทราบว่า Cookie มาจากไหน มีการเก็บ Cookie ไว้ในคอมพิวเตอร์ของผู้ใช้เมื่อไร และจะ

ทำงานกับคอมพิวเตอร์เหล่านั้นอย่างไร สิ่งเหล่านี้จะทำให้ผู้ใช้เป็นกังวลถึงความเป็นส่วนตัวของพวกเขา ดังนั้นบางไคลเอนท์จึงเลือกที่จะหลีกเลี่ยงที่จะมี Cookie ไว้ในบราวเซอร์ของไคลเอนท์

จากปัญหาข้างต้นจึงมีความต้องการวิธีการใหม่ ที่จะใช้ในการจัดการเซสชันให้กับ Servlet ในแบบที่ไม่ใช้ Cookie ซึ่งวิธีการใหม่นี้คือการทำ URL Rewriting คือแทนที่จะเพิ่มข้อมูลใน HTTP Header ก็จะใช้การฝัง Session Identifier ไว้ใน URL ที่ถูกเรียกใช้ โดยจะส่ง URL กลับและส่งต่อไประหว่างไคลเอนท์และเซิร์ฟเวอร์ [10] แต่ข้อเสียหลักของวิธีการนี้คือการทำการพิสูจน์ (Identifier) ซึ่งโดยปกติแล้วก็จะเป็นค่าตัวเลข ที่นำมาใส่ไว้ใน Page Links ถ้าผู้ใช้ทำการ Bookmark หน้าที่ใช้ แล้ว Bookmark จะรวมเอา Session ID ไว้ด้วย ดังนั้น ถ้าผู้ใช้พยายามที่จะเข้ามาดูอีกรอบหนึ่ง ก็จะพบคำว่า “page not found error” เนื่องจากเซสชันได้หมดเวลาไป อย่างไรก็ดี วิธีการนี้อาจจะไม่เหมาะสมสำหรับแอพพลิเคชันที่ซับซ้อน และที่สำคัญ Session ID อาจถูกปรับเปลี่ยนได้ง่ายโดยผู้เจาะระบบเครือข่ายคอมพิวเตอร์ (Hackers) ดังนั้นในส่วนของความปลอดภัย วิธีการนี้จึงไม่เหมาะสม

เทคนิควิธีการ Cookie ขึ้นอยู่กับการเพิ่มเซสชันใน HTTP Layer และได้ให้การทำโปรแกรมในระดับชั้นของแอพพลิเคชัน (Application Layer Programming Languages) เช่น Servlet ในส่วนของเนื้อหา (Content) ผ่านกลไกที่ได้กำหนดไว้ใน HTML [12] แต่ SOAP ไม่ได้ขึ้นอยู่กับโพรโทคอล HTTP และไม่ได้เป็นไฟล์ HTML นั่นก็คือวิธีของ Cookie จึงไม่เหมาะสมสำหรับใช้กับการจัดการ SOAP Session ซึ่งก็คล้ายกับ URL Rewriting ที่ให้แต่ละผู้ใช้ระบุ URL ในการคุยกับเว็บเซิร์ฟเวอร์ แต่ข้อมูล URL นี้ไม่สามารถส่งไปยัง SOAP Messages เพื่อที่จะให้ SOAP จัดเก็บข้อมูลเซสชันได้ [12]

2.3.1.2 การจัดการเซสชันบนโพรโทคอล HTML

วิธีการจัดการเซสชันบนโพรโทคอล HTML อย่างง่ายคือ การซ่อนข้อมูลเซสชันไว้ในฟอร์มของ HTML เมื่อผู้ใช้ออนไลน์ส่งข้อมูลผ่านเว็บเพจ และใส่ข้อมูลในฟอร์มของเพจนั้น ข้อมูล Session ID จะถูกซ่อนในฟอร์มและส่งจากไคลเอนท์ไปยังเซิร์ฟเวอร์ ซึ่ง Session ID นี้อาจถูกพบได้ง่ายโดยการค้นหาจากซอร์สโค๊ดของเว็บเพจนั้น และเนื่องจากว่า การทำงานของเว็บเซอร์วิสจะต้องติดต่อกันโดยใช้โพรโทคอล SOAP แต่ SOAP ไม่รองรับการทำงานร่วมกันกับฟอร์มของ HTML ดังนั้นการจัดการเซสชันโดยใช้ฟอร์มของ HTML ไม่สามารถทำงานร่วมกับ SOAP Message

2.3.2 การจัดการเซสชันบนโพรโทคอล SOAP

SOAP ให้การอินเทอร์เฟสที่มีประสิทธิภาพสำหรับเว็บเซอร์วิส และยอมให้ผู้ใช้สร้างเมธอดที่มีความซับซ้อนในการเรียกระหว่างเว็บแอพพลิเคชันที่แตกต่างกัน และยังทำงานบนแพลตฟอร์มที่ต่างกันได้ ดังนั้นจึงมีนักพัฒนาจำนวนมากต้องการใช้ข้อดีเหล่านี้ และนำ SOAP มาใช้ในการขยาย

การทำงานในเว็บแอพพลิเคชันของพวกเขา แต่อย่างไรก็ตาม เทคนิคการจัดการเซสชันตามวิธีการดั้งเดิมที่ได้กล่าวมาแล้วนั้นไม่สามารถรองรับการจัดการเซสชันบน SOAP ได้ ในการที่จะใช้ SOAP ในเว็บเซอร์วิสต่าง ๆ จะมีเทคนิคหลักอยู่สองวิธีที่ออกแบบมาเพื่อเพิ่มข้อมูลเซสชันให้กับ SOAP Calls ดังจะกล่าวต่อไปนี้

2.3.2.1 การรวมการจัดการเซสชัน ไว้ในเว็บเซิร์ฟเวอร์

ในโปรเจคที่ชื่อ REGNET [13] ได้ใช้ Zap ซอฟต์แวร์เพื่อการจัดการเซสชันให้แก่ SOAP Message จุดประสงค์ของโปรเจคนี้คือต้องการสร้างการทำงานของเครือข่ายที่ชื่อ Cultural Service Centers ทั่วทั้งยุโรป ซึ่งแสดงให้เห็นถึงการใช้โพรโทคอล SOAP ในระบบธุรกรรมทางอิเล็กทรอนิกส์ ซึ่ง Zap เป็นซอฟต์แวร์ที่เป็นของ Apache เว็บเซิร์ฟเวอร์ ซึ่งจะสามารถซ่อนการจัดการเซสชันที่ซับซ้อนไว้ได้ โดยโปรเจคนี้ได้นำ Apache มาใช้ในการให้บริการ HTTP Service สำหรับ SOAP Calls การรวบรวมการให้บริการเซสชันไว้ภายในเว็บเซิร์ฟเวอร์มีทั้งข้อดี และข้อเสีย ข้อดีคือ สามารถเรียกใช้ซอฟต์แวร์ได้ง่าย และสามารถพัฒนาระบบโดยรวมได้ง่าย นอกจากนี้คือช่วยลดภาระของการดูแลระบบและยังขยายประสิทธิภาพของระบบได้อีกด้วย [14] แต่ข้อเสียคือ Zap นี้เป็นส่วนหนึ่งของ Apache เซิร์ฟเวอร์ซึ่งจะไม่ทำงานร่วมกับเว็บเซิร์ฟเวอร์อื่นในการจัดการเซสชัน

2.3.2.2 การเพิ่ม State Context ใน SOAP Header

อีกวิธีการหนึ่งในการเพิ่มการจัดการเซสชัน บน SOAP Messages คือการใช้ SOAP Header เพื่อให้ได้การบริการแบบ Stateful ในงานวิจัย [2] ที่ใช้ SOAP 1.2 [17] ในส่วนของ SOAP Header จะมีลักษณะเฉพาะอยู่สี่แบบดังนี้

1. SOAP Header เป็นข้อมูลดิบใส่ใน SOAP Messages
2. SOAP Users สามารถกำหนด Header ด้วยตัวเองได้
3. SOAP Header จะมี Namespace ในตัวเองซึ่งจะแตกต่างจาก SOAP Namespace
4. SOAP Nodes เป็นตัวกลางระหว่างผู้ส่ง SOAP และ ผู้รับปลายทางซึ่งสามารถจัดการการส่ง SOAP Message ตลอดเส้นทาง และยังสามารถจัดการกับข้อมูลใน SOAP Header ได้

ในงานวิจัยของ Jeckle [14] กล่าวว่า SOAP Node ยังมีคุณสมบัติอื่น ๆ อีกเช่น SOAP Node สามารถจัดการ SOAP Message ได้ตลอดการติดต่อสื่อสารของข้อความนั้น SOAP Node สามารถแยก SOAP Header ออกได้เมื่อ SOAP Node นั้นตรงกับผู้รับ SOAP Message ที่ได้มีการกำหนดไว้ใน SOAP Header และผู้ส่งสามารถระบุตำแหน่งของ SOAP Node ได้จาก SOAP Header จากการใช้ URI (Uniform Resource Identifier)

จากคุณสมบัติเหล่านี้ Jeckle ได้กำหนดข้อมูล Context ใน SOAP Header ข้อมูลใน Header มีทั้ง Context ID การกำหนดเวลา ผู้สั่งและอีกหลายอย่าง Jeckle ยืนยันว่าวิธีการนี้มีโครงสร้างและภาษาโปรแกรมที่เป็นอิสระ เนื่องจาก SOAP Header จะใช้การขยายเพิ่มเติมภายในฟอร์ม SOAP

จากข้อดีดังกล่าว SOAP จึงไม่ขึ้นอยู่กับ HTTP ในการส่งข้อความ แต่สามารถใช้โพรโทคอลการส่ง เช่น SMTP ได้อีกด้วย นอกจากนี้ SOAP ยังสามารถทำงานร่วมกับโปรแกรมภาษาได้หลายภาษานอกเหนือจากภาษา Java แต่อย่างไรก็ตาม Jeckle ไม่ได้นำเสนอการประยุกต์การจัดการเซสชันให้กับผู้พัฒนาเว็บนั่นคือ ผู้พัฒนาแต่ละคนจะต้องเลือกวิธีการของตัวเองในการจัดการข้อมูลเซสชันบน SOAP

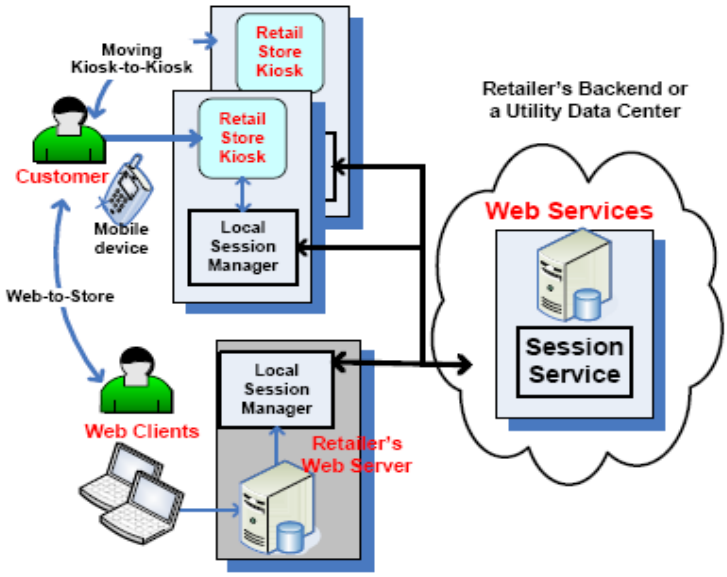
ไมโครซอฟท์ได้ใช้ SOAP ในการจัดการ SOAP เซสชันโดยใช้ SOAP Header Elements 3 แบบดังต่อไปนี้ <BeginSession> <Session> <EndSession> ซึ่งข้อมูลที่จะใช้ในการสร้างเซสชันก็จะสามารถใส่ไว้ใน SOAP [20] ได้ แต่อย่างไรก็ตามพบว่าเวอร์ชันของ SOAP เซสชัน นี้ยังคงเป็นวิธีการแบบ Cookie โดยมีการใส่ Cookie ไว้ใน <Session> Tag ซึ่งรวมไว้ใน SOAP Header [17] นอกจากนั้น <Session> Element ก็ยังถูกส่งระหว่างไคลเอนท์และเซิร์ฟเวอร์เช่นเดียวกับ HTTP Cookie นักพัฒนาจะต้องแน่ใจว่าว่าไคลเอนท์แอพพลิเคชันยอมรับ <Session> Tags ที่เป็น HTTP Cookie และส่งกลับไปยังเซิร์ฟเวอร์ในแต่ละ Request นอกจากนี้วิธีการนี้ใช้สมมติฐานว่า SOAP ได้ถูกส่งไปบน HTTP ซึ่งเป็นการจำกัดการทำงานร่วมกันระหว่าง SOAP และโพรโทคอลการส่งชนิดอื่น

2.4 การจัดการเซสชันในงานด้านบริการทางธุรกิจ

ในงานวิจัย [23] ได้ออกแบบระบบงานโดยให้การจัดการเซสชันที่อยู่ห่างไกลเป็นการจัดการเซสชันแบบกระจายศูนย์ โดยใช้หลักการของ SOA หรือ Service Oriented Architecture ซึ่งมีหลักการการออกแบบเพื่อการเชื่อมโยงธุรกิจและการคำนวณทรัพยากร ซึ่งโดยทั่วไปแล้วคือ องค์กรธุรกิจ แอพพลิเคชัน และข้อมูล ซึ่งต้องอยู่บนพื้นฐานของความต้องการของผู้ใช้บริการ (ซึ่งอาจจะเป็น ผู้ใช้ หรือ บริการอื่นที่มาใช้บริการนี้)

2.4.1 รูปแบบการจัดการเซสชันของระบบ

จากภาพที่ 2.9 ประกอบด้วยส่วนที่เป็นหน่วยงานท้องถิ่น ซึ่งจะต้องมีส่วนที่จัดการเซสชันหลักของพื้นที่นั้น ๆ เนื่องจากจุดประสงค์หลักคือต้องการลดปัญหาขององค์กรที่มีระบบกระจายอยู่หลายแห่งซึ่งแต่ละที่มีความแตกต่างของระบบและอุปกรณ์พื้นฐานที่มีอยู่แต่เดิม และอยู่ห่างไกลจากกัน โดยการเพิ่มการจัดการเซสชันที่เป็นโลคอลให้แก่หน่วยงานพื้นที่นั้น ๆ และแต่ละแห่งจะติดต่อ



ภาพที่ 2-9 การออกแบบการจัดการเซสชันและองค์ประกอบของระบบ (อ้างอิงจาก [23])

กับส่วนที่ให้บริการเซสชันที่อยู่ห่างไกล ซึ่งเป็นศูนย์กลางการจัดการรวมขององค์กร สามารถอธิบายการทำงานของแต่ละส่วนได้ดังนี้

2.4.1.1 อุปกรณ์การจัดการเซสชันในพื้นที่

ทำงานร่วมกับอุปกรณ์และสิ่งแวดล้อมเดิมเช่น เครื่องคอมพิวเตอร์ตั้งโต๊ะ เว็บ และ อุปกรณ์แบบเคลื่อนที่ เพื่อทำให้ส่วนต่าง ๆ เหล่านี้สามารถทำการจัดลำดับชุดข้อมูลพิเศษของผู้ใช้ (Session State) ทำการจัดเก็บข้อมูลในพื้นที่ให้รวดเร็ว จับเวลาเซสชัน ตรวจจับเหตุการณ์ต่าง ๆ ของเซสชัน และสุดท้ายคือสื่อสารกับหน่วยบริการเซสชันที่อยู่ไกลซึ่งเป็นเว็บเซอร์วิสของระบบ

2.4.1.2 อุปกรณ์การจัดการเซสชันในระยะไกล

ส่วนนี้จะทำหน้าที่สร้างการทำงานร่วมกันของหน่วยงานต่าง ๆ ขององค์กรตามพื้นที่ที่กระจายห่างออกไป จัดเก็บข้อมูลขององค์กรรวมในระยะยาว ให้บริการช่องทางการสื่อสารจำนวนมาก และทำดาต้าไมนิ่งข้อมูลของผู้ใช้ใน Session State เพื่อให้บริการแก่ลูกค้า ซึ่งส่วนนี้จะเป็นกลุ่มของเว็บเซอร์วิสที่ทำงานร่วมกันเพื่อให้เกิดการบริการตามวัตถุประสงค์

2.4.2 รายละเอียดการจัดการเซสชัน

2.4.2.1 อุปกรณ์การจัดการเซสชันในพื้นที่

ในส่วนของการจัดการเซสชันในพื้นที่ จะมีคลาสที่สร้างขึ้นมาเมื่อแอพพลิเคชันในพื้นที่นั้นได้เริ่มขึ้นหรือเว็บเซิร์ฟเวอร์ได้รับการร้องขอครั้งแรกระบบการจัดการเซสชันนี้จะตรวจสอบว่ามีเซสชันก่อนหน้าอยู่หรือไม่เพื่อให้ได้ User ID หรือ Session ID หากว่ายังมีอยู่ ก็จะทำการเรียกเซสชันนี้จากแหล่งข้อมูลในพื้นที่เช่น ไฟล์ หรือ ฐานข้อมูล หรือไม่ก็เรียกมาจากหน่วยจัดการเซสชันที่อยู่ไกลได้ แต่หากว่าไม่พบเซสชันก่อนหน้านี้ จะต้องสร้างเซสชันตัวใหม่ นอกจากนั้นจะต้องสร้าง SessionManager Class ที่ทำหน้าที่ให้บริการเสมือนเป็นคอนเทนเนอร์สำหรับสถานะของแอพพลิเคชัน ตัวอย่างเช่น เก็บข้อมูลการซื้อ เป็นต้น ซึ่งจะคล้ายกับเว็บเซสชัน แต่กรณีนี้สามารถเก็บเซสชันและโหลดไปไว้ที่แชนเนลใดก็ได้ ซึ่งคลาสตัวนี้จะรู้ถึงตำแหน่ง วิธีการ และเวลาในการจัดเก็บข้อมูล ตัวอย่างเช่น ทราบถึงตำแหน่งในการจัดเก็บสถานะไว้ในหน่วยความจำชั่วคราว ในไฟล์ ในฐานข้อมูล หรือ ในเซสชันคอนเท็กส์ของเซิร์ฟเวอร์ ทราบถึงวิธีการใช้ XML Binary หรือ ชนิดของลำดับข้อมูล และสุดท้าย ทราบถึง เวลาที่ทำซิงค์โครไนส์สถานะกับตัวจัดการเซสชันที่อยู่ไกลได้ทันที หรือต้องรอ หรือไม่ทำการใด ๆ ทั้งนี้ขึ้นอยู่กับกฎที่ได้ตั้งไว้

2.4.2.2 อุปกรณ์การจัดการเซสชันที่อยู่ไกล

ส่วนนี้จะมีเมธอดพื้นฐานอยู่สองเมธอดได้แก่ ExportSession และ ImportSession ซึ่งให้ตัวจัดการเซสชันในพื้นที่สามารถเรียกใช้ได้ นอกจากนี้ยังติดต่อเว็บเซอร์วิสตัวอื่นเพื่อยืนยันเซสชันและทำการเรียกข้อมูลกลับมาดู

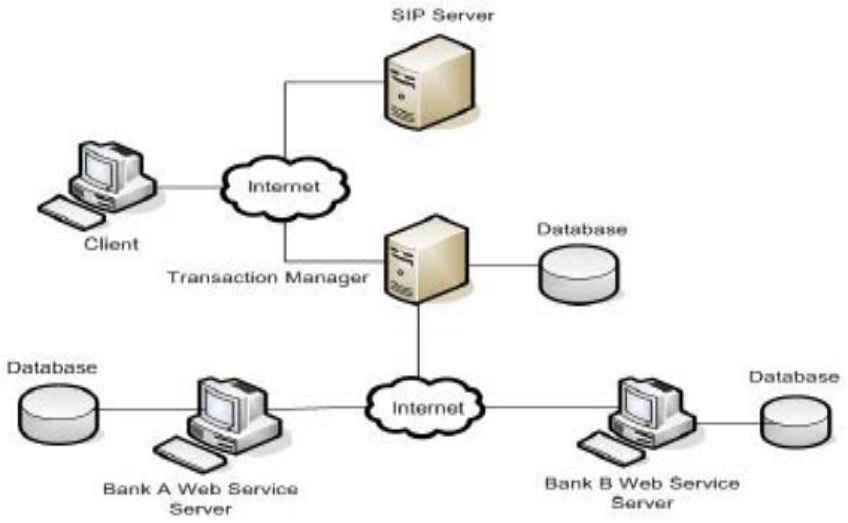
2.4.3 การนำไปใช้

การจัดการเซสชันจะต้องมีความหลากหลายของเซสชันของผู้ใช้ อันดับแรก ความต่อเนื่องของแอพพลิเคชันที่มีหลายแชนเนล สามารถเก็บสถานะของแอพพลิเคชันที่เป็นไปตามลำดับ และทำการดึงกลับมาได้จากแชนเนลเดียวกันหรือต่างกันก็ได้ [23]

2.5 การจัดการเซสชัน (Session) และทรานแซคชัน (Transaction) ให้กับเว็บเซอร์วิสโดยการใช้ SIP

ในงานวิจัย [24] ได้ใช้โพรโทคอล SIP ในการจัดการเซสชัน โดยจะนำ Session ID ใส่ใน SOAP Messages ซึ่งจะทำหน้าที่ในการเรียกและใช้บริการของเว็บเซอร์วิส โดยได้สร้างระบบ Video Shopping Web Services และระบบ E-banking Web Services Transaction ซึ่งจะให้การบริการในเรื่องของการจัดการทรานแซคชันที่เกิดจากการใช้บริการ Video Shopping Web Services ซึ่งเป็นตัวสร้าง Session ID หลักของระบบในการที่จะให้การจัดการทั้งเซสชัน และทรานแซคชัน

ระบบที่งานวิจัยนี้สร้างขึ้นอยู่ในสมมติฐานขั้นตอนของการเริ่มสร้างเซสชัน ซึ่งจะเกิดภายในโดเมนเดียวกันและกำหนดให้ผู้ใช้ได้รู้ IP address ของ UAS แล้ว ดังนั้นระบบนี้จึงไม่ต้องมีทั้ง SIP Register Server และ SIP Proxy Server ภาพที่ 2-10 แสดงโมเดลของระบบ

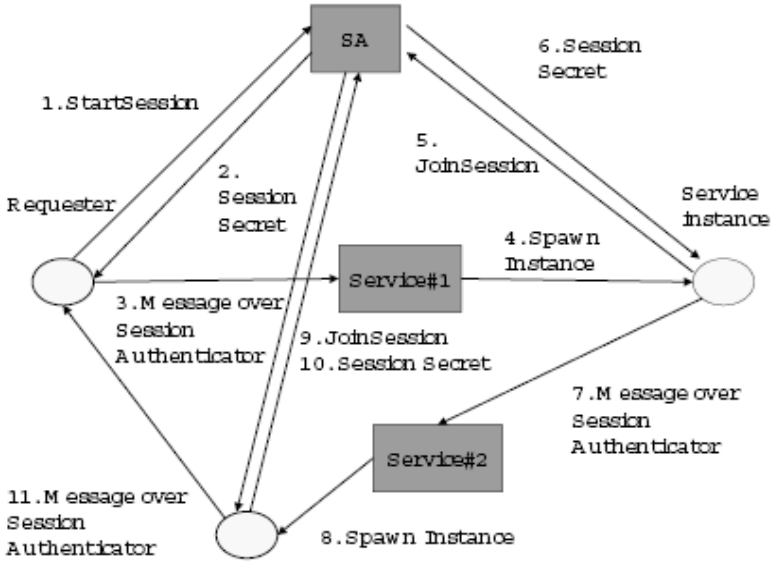


ภาพที่ 2-10 สถาปัตยกรรมของระบบ อี-แบงค์ (อ้างอิงจาก [24])

2.6 การจัดการเซสชันบน SOAP

ในงานวิจัย [25] ได้สร้าง E-business ที่ซึ่งประกอบด้วยสมาชิกหลาย ๆ ส่วน เช่นไคลเอนท์และเว็บเซอร์วิสมากกว่าหนึ่งตัว ที่ต้องการทำงานร่วมกันเพื่อสร้าง Business Flow โดยจะนำเสนอโพรโทคอลซึ่งทำให้สมาชิกของการทำ Business Flow สามารถมั่นใจว่ากำลังสื่อสารกับ Service Instances ของเว็บเซอร์วิสซึ่งเป็นส่วนหนึ่งของสมาชิกที่อยู่ใน Business Flow เดียวกัน ซึ่งโพรโทคอลนี้ประกอบด้วยสองส่วนหลัก ๆ ได้แก่ Message Authentication Protocol ซึ่งทำหน้าที่ในการพิสูจน์ตัวตนของการร่วมเซสชันเดียวกัน ซึ่งในกรณีนี้มีการร่วมใช้ Session Secret ระหว่างสมาชิกใน Business Flow อีกส่วนคือโพรโทคอลการจัดการเซสชันซึ่งเป็นการจัดการเซสชันของระบบ

งานวิจัยนี้ใช้เทคโนโลยีพื้นฐานที่เป็นมาตรฐานเว็บเซอร์วิส เช่น SOAP, XML Signature, Encryption และ SOAP-DSIG นอกจากนี้ยังแสดงให้เห็นถึงขั้นตอนของการเริ่มเซสชันและการทำงานร่วมกันของไคลเอนท์ การทำการพิสูจน์ตนของเซสชัน (Session Authentication) และเว็บเซอร์วิสสองตัว ภาพที่ 2-11 แสดงโมเดลของระบบ



ภาพที่ 2-11 การจัดการเซสชันบน SOAP (อ้างอิงจาก [25])

2.7 ระบบกระจายศูนย์

การให้บริการในระบบกระจาย เป็นการทำงานในลักษณะชุดของการทำงานร่วมกันในการดำเนินการของเซิร์ฟเวอร์ที่ได้กระจายอยู่ระหว่างเครื่องคอมพิวเตอร์จำนวนมาก ซึ่งการดำเนินการของเซิร์ฟเวอร์เหล่านี้จะรับ Request ที่เกิดขึ้นจากการดำเนินการของผู้ใช้บริการและร่วมกันจัดการข้อมูลและประมวลผลการให้บริการ โดยปกติแล้วการดำเนินการจะมีชั้นของการสื่อสารในการส่งข้อมูล และชั้นของการสื่อสารนี้จะให้การรับรองการสื่อสารที่แตกต่างกัน ตัวอย่างเช่น การจัดลำดับข้อความแบบ FIFO (FIFO Message Ordering) ซึ่งเป็นการรับและส่งข้อความตามลำดับ และโพรโทคอลการสื่อสารแบบกลุ่ม (Group Communication Protocols [33])

ในปัจจุบัน ระบบปฏิบัติการจะให้บริการในการสื่อสารเพื่อการแลกเปลี่ยนข้อมูลระหว่างการดำเนินการ (Processes) โดยใช้ TCP/IP Protocol Stack ซึ่งไม่ขึ้นกับเทคโนโลยีด้านเครือข่าย ซึ่งการบริการของระบบปฏิบัติการ จะต้องตอบสนองในการส่งและรับข้อความที่ถูกส่งไปเพื่อตอบสนองต่อการดำเนินการนั้น ๆ ดังนั้น การดำเนินการจึงต้องมีความไว้วางใจระบบการปฏิบัติการในการติดตามการสื่อสาร แต่การให้บริการในการสื่อสารต้องการเฉพาะการวิเคราะห์เป้าหมายเพื่อการจัดส่งข้อมูลด้วยวิธีการนี้ การดำเนินการสามารถสื่อสารซึ่งกันและกันกับการดำเนินการชนิดอื่นในระบบหากว่าสามารถพิสูจน์ทราบตำแหน่งของเป้าหมายที่ต้องการได้ ดังนั้นเซ็ตของการดำเนินการที่มีการ

แลกเปลี่ยนข้อความโดยใช้ TCP/IP Protocol Stack และได้ทราบตำแหน่งการสื่อสารของกันและกันแล้ว อาจพิจารณาได้ว่าเป็นระบบกระจายที่ซึ่งการดำเนินการนั้นสามารถสร้างเครือข่ายที่สมบูรณ์ได้

ในมุมมองของการการควบคุมหลาย ๆ อัลกอริธึมที่เรียกว่าเป็นระบบกระจาย ในความจริงยังถือเป็นระบบรวมศูนย์อยู่ โดยจะมีผู้ทำงานหลักที่จะให้บริการแก่สมาชิกอื่นเมื่อได้รับการร้องขอ ซึ่งอัลกอริธึมเหล่านี้จึงเป็นเสมือนระบบศูนย์กลาง ซึ่งระบบแบบนี้มีผลเสียอันเกิดจากการล้มเหลวของตัวกลางที่ทำหน้าที่หลักได้ แต่สามารถแก้ไขปัญหานี้ได้ เพื่อให้ระบบยังคงดำเนินการได้อย่างต่อเนื่องโดยใช้วิธีการคัดเลือกหนึ่งในสมาชิกเพื่อทำงานหลักแทนที่ตัวที่ล้มไป ซึ่งเรียกอัลกอริธึมชนิดนี้ว่า Leader Election ซึ่งมีอยู่หลายอัลกอริธึม เช่น อัลกอริธึมบูลลีย์ (Bully Algorithm) และ ริงอัลกอริธึม (Ring Algorithm) เป็นต้น

อัลกอริธึมนี้สามารถนำมาเลือกเซิร์ฟเวอร์ที่เหมาะสมที่จะให้บริการเหลัก เช่นให้บริการด้านการลงทะเบียนในฐานข้อมูล หรือ URL เป็นต้น ซึ่งเซิร์ฟเวอร์เหล่านี้อาจอยู่ในรูปแบบการทำคลัสเตอร์ (Cluster) หรือเซ็ตของเซิร์ฟเวอร์ที่กระจายอยู่ในอินเทอร์เน็ต

2.8 ความเสถียรภาพของระบบ (Reliability)

หมายถึงคุณสมบัติที่ระบบสามารถทำงานได้อย่างต่อเนื่องโดยปราศจากการล้มเหลว ซึ่งกำหนดในรูปของช่วงเวลา (Time Interval) ระบบที่มีเสถียรภาพสูงควรจะทำงานได้อย่างต่อเนื่องโดยไม่มีการอินเตอร์รัพ และทำงานได้อย่างยาวนาน [26] การที่จะสร้างระบบให้มีเสถียรภาพจะต้องประกอบด้วยการส่งข้อมูลได้อย่างทั่วถึง ผู้รับสามารถรับข้อมูลได้ ผู้ส่งแน่ใจได้ว่าผู้รับได้รับข้อมูลแน่นอน มีการสำรองข้อมูล หรือระบบมีฟังค์ชันที่จะสามารถเลือกตัวทำงานหลักตัวใหม่ทำงานแทนกันได้

2.8.1 การทำ Reliable Multicasting

เป็นการส่งข้อมูลหนึ่งไปยังกลุ่มของผู้รับที่เป็นเป้าหมายในการส่งข้อมูล โดยที่ผู้รับแต่ละเครื่องเมื่อได้รับข้อมูลแล้วจะต้องมีการตอบรับข้อมูลนั้น ซึ่งการตอบกลับอาจเป็นได้ทั้งเป็น Positive Acknowledgement (ACK) หรือ Non-Acknowlegement (NAK) ขึ้นอยู่กับความเหมาะสมของระบบการตอบกลับเหล่านี้จะทำให้เกิดความเสถียรภาพของระบบ ซึ่งการทำ Reliable Multicast เหมาะสมสำหรับเครือข่ายชนิด LAN หรือ WAN ซึ่งช่วยลดค่าใช้จ่ายที่เกิดขึ้นในระบบ [27]

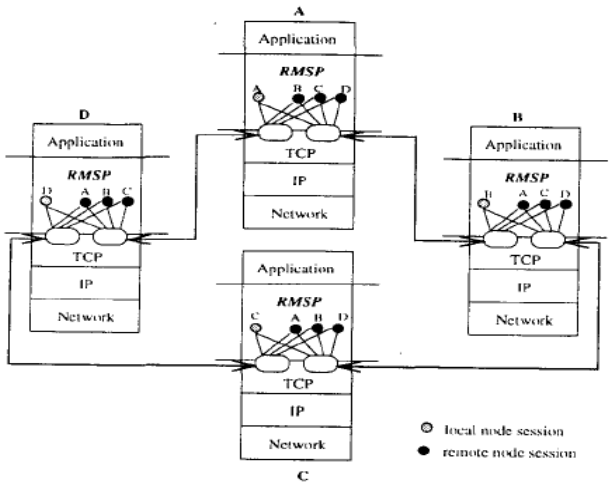
2.8.2 กลไกฮาร์ทบีทโดยการระบุช่วงเวลา

สามารถสร้างระบบให้มีเสถียรภาพได้ โดยการเพิ่มส่วนของการตรวจจับการส่งแพ็คเก็ตข้อมูลโดยใช้การกำหนดช่วงเวลาให้แก่เซิร์ฟเวอร์แต่ละตัวในระบบจะมีตัวจับเวลา (Timer) ซึ่งเป็นตัวกำหนดเวลาของการส่งแพ็คเก็ต โดยแต่ละเซิร์ฟเวอร์จะส่งสถานะของตัวเองเป็นช่วงเวลาให้กับ

เซิร์ฟเวอร์ตัวต่อไปจนครบ ช่วงเวลาของแต่ละเซิร์ฟเวอร์จะต้องกำหนดให้สอดคล้องกัน ในกรณีที่กำหนดให้เครือข่ายเป็นแบบ LAN และติดต่อกันแบบเครือข่าย Virtual Ring ดังนั้นก็จะส่งให้กันในลักษณะวงแหวน เมื่อแพ็คเก็ตวนกลับมายังผู้ส่งเริ่มแรกตามเวลาที่กำหนด เป็นการยืนยันได้ว่าไม่มีเซิร์ฟเวอร์ตัวใดล่ม และแต่ละเซิร์ฟเวอร์ที่มีการติดต่อกันในแต่ละด้านของวงก็จะสามารถตรวจสอบสถานะภาพการทำงานซึ่งกันและกันได้

2.9 โพรโทคอลการจัดการเซสชันแบบมัลติคาสท์ (Multicast)

งานวิจัย [28] ได้นำเสนอโพรโทคอลที่ให้การนำส่งข้อมูลจำนวนมากให้กับการสื่อสารที่มีลักษณะ NxN Multicast หรือแบบวงแหวนเสมือน (Virtual Ring) โดยจะทำการมัลติแคสท์ข้อมูลแอพพิเคชันไปยังสมาชิกในเครือข่ายวงแหวนเสมือน ที่ระดับ Session Layer และสามารถทำงานบนโพรโทคอล TCP โพรโทคอลนี้จะทำมัลติเพล็กซ์ Session Messages หลาย ๆ ตัวต่อหนึ่ง Transport Level Packet เพื่อลด Overhead ในชั้นของทรานสปอร์ตต่อหนึ่งข้อความ แต่ละ Node จะสื่อสารกับ Node ข้างเคียงโดยการใช้การติดต่อแบบ Two Transport Level เท่านั้น ภาพที่ 2-12 แสดงให้เห็นถึงโมเดลของงานวิจัยนี้



ภาพที่ 2-12 RMSP Multiplex Session Layer (อ้างอิงจาก [28])

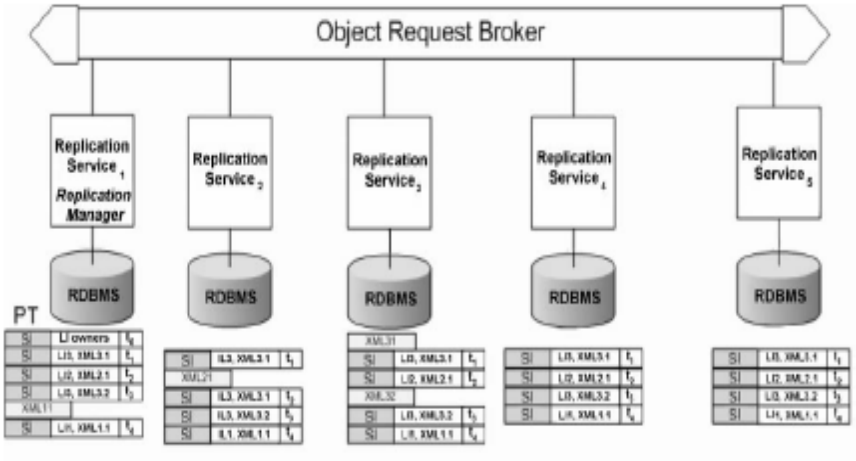
ซึ่งการทำงานของแต่ละ Node ในการควบคุมการไหลของแพ็คเก็ตได้ใช้หลักการของกลไกฮาร์ทบีท (Heartbeat Mechanism) เพื่อทำการตรวจสอบการล้มเหลวของ Node

2.10 การออกแบบและการใช้ฮาร์ทบีท (Heartbeat) ในสภาพแวดล้อมแบบมัลติแมชชีน (Multi-Machine)

งานวิจัย [32] นำการใช้กลไกฮาร์ทบีท ซึ่งเป็นเทคนิคการใช้การจับเวลามาควบคุมระบบ ทำให้ระบบมีความน่าเชื่อถือ โดยจะตรวจสอบความล้มเหลวของ Node และ มีการส่ง Transport Level Packets อย่างต่อเนื่อง แต่ละ Node ในระบบจะมีการจับเวลาในตัวอง ระบบนี้จะกำหนดให้แต่ละ Node ในระบบต้องรับรู้และเข้าใจใน Finite State Machine ซึ่งประกอบด้วย Election State ทำหน้าที่ค้นหาโทโปโลยีของเครือข่าย (Network Topology) ส่วนที่สองคือ Standby state ทำหน้าที่ Queries และ คอยตรวจจับ (Monitor) การทำงานของ Node ที่เป็น Master State ในส่วนของ Not-Active จะไม่ทำอะไรยกเว้นจะตอบสนองกับการ "Sweeping" จาก Master และคำสั่งการควบคุม "Standby" จาก Console และ Master หนึ่งใน Node ทั้งหมดจะเป็น Master Node และสถานะของมันจะเป็น Master ซึ่งทำหน้าที่หลักในการให้บริการในระบบ กรณีที่มีตรวจสอบว่าส่วนที่เหลือจะเป็น Standby Nodes และสถานะจะเป็น Standby ในกรณีที่ต้องการที่จะให้ตัวใดตัวหนึ่งทำงานหลักแทนตัวเองก็จะมีการถ่ายโอนข้อมูลให้กับตัวที่ได้เลือกไว้ แล้วเปลี่ยนสถานะตัวเองเป็น Not Active แต่ถ้าตัวหลักที่ทำงานเกิดล่มลง ระบบจะใช้อัลกอริธึมบูลลีย์ ในการเลือกตัวหลักจากที่เหลือในระบบมาทำงานแทน

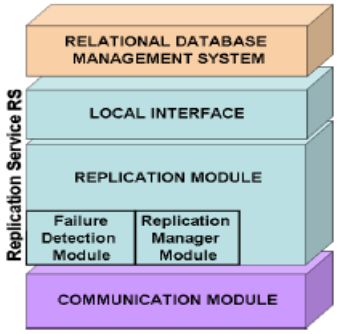
2.11 การให้บริการการทำซ้ำข้อมูลแบบกระจายในระบบ Business to Business (B2B)

งานวิจัย [29] แสดงให้เห็นถึงการให้บริการการทำซ้ำข้อมูลแบบกระจาย สำหรับจัดการจัดเก็บข้อมูลถาวรซึ่งเป็นข้อมูลทางการค้าที่ได้สร้างอยู่ในรูปแบบ XML โดย B2B แอพพลิเคชันที่ใช้งานกับเว็บเซอร์วิส แสดงดังภาพที่ 2-13 ซึ่งแสดงผังการทำงานของระบบนี้



ภาพที่ 2-13 ผังการทำงานของการเรียกการทำซ้ำข้อมูล (อ้างอิงจาก [29])

เว็บเซิร์ฟเวอร์จะถูกเชื่อมเข้ากับ Replicate Nodes (RN) โดยจะมีค่าใช้จ่ายในส่วนของการทำการทำซ้ำข้อมูลทางการค้า การเรียกร้องการทำซ้ำข้อมูล (Replication) ได้ถูกจัดการโดยส่วนจัดการการทำซ้ำข้อมูลแบบกระจาย Replication Manager Module (RMM) เพื่อให้แน่ใจว่ามีการควบคุมการเกิดขึ้นพร้อมกันที่ถูกต้อง และยังให้บริการทำ Deadlock Free ที่ได้จากการรวบรวมและจัดลำดับคำสั่งที่ต้องการทำซ้ำข้อมูล นอกจากนี้ยังมีการทำ Fault-Tolerance ของ Replication Manager (RM) โดยการใช้ Leader Election Algorithm ในการเลือกตัวจัดการตัวใหม่ถ้าหากตัวจัดการหลักล่มลง RM ยังทำการจัดภาระการทำงานในระบบให้สมดุล (Load Balancing) สำหรับทุก ๆ การร้องขอการทำซ้ำ ดังนั้นในแต่ละ Replicaton Node (RN) จะต้องประกอบด้วยส่วนสำคัญดังภาพที่ 2-14



ภาพที่ 2-14 โครงสร้างของ Replication Node (อ้างอิงจาก [29])

โดยในโมเดลนี้ได้ใช้ Leader Election Algorithm for Complete Networks ในการเลือกตัวหลักใหม่ ซึ่งจะไม่ใช้การค้นหาเส้นทาง แต่จะใช้ค่าลำดับความสำคัญที่พบได้ในแต่ละเซิร์ฟเวอร์ที่ได้กำหนดไว้

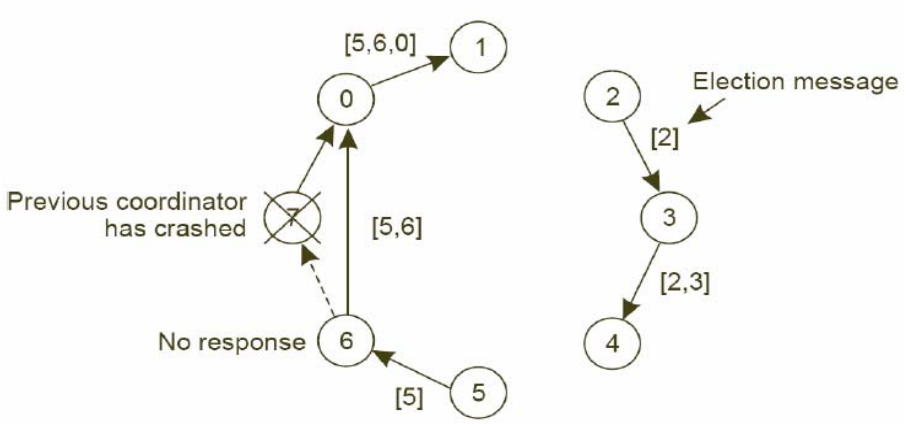
2.12 การทำมัลติคาสท์ที่มีเสถียรภาพบนอีเธอเน็ต

ในงานวิจัย [27] ได้อ้างถึงว่าในอินเทอร์เน็ตทั่วไปนั้นการทำมัลติคาสท์ที่มีเสถียรภาพ (Reliable Multicast) ค่อนข้างยากเนื่องจากเหตุผลสามประการ อันดับแรกคือ การที่จะให้ระบบมีความเสถียรภาพจะต้องมีการตอบรับกลับมาจากผู้รับ อย่างที่สองคือ การดูแลสมาชิกในระบบ และการทำให้ระบบมีเสถียรภาพในกรณีที่ต้องทำ Dynamic Multicast Group ซึ่งสมาชิกในเครือข่ายอาจจะเข้าหรือออกจากระบบกลุ่มได้ตลอดเวลา และเหตุผลที่สามได้แก่การควบคุมการไหลของแพ็คเก็ตข้อมูลที่จะให้แก่ผู้รับ ค่อนข้างยุ่งยาก เนื่องจากว่าผู้รับมีความแตกต่างกัน แต่ในการทำคลัสเตอร์ร่วมกับการทำมัลติคาสท์ ซึ่งเป็นการสื่อสารกันแบบ Multicast Group ได้แสดงให้เห็นว่าในรูปแบบการสื่อสารทั้งหมดการทำมัลติคาสท์นั้นจะมีความคงที่สูงและจำนวนของสมาชิกจะไม่มีผลกับระบบ

ในการทำมัลติคาสท์จะใช้ ACK-Based Protocol และ NAK-Based Protocol ในการทดสอบ ACK นั้นผู้ส่งจะทำการมัลติคาสท์ข้อมูลไปให้กับกลุ่มผู้รับ ผู้รับจะทำการ Unicast ACK กลับมาและผู้ส่งจะปล่อยบัฟเฟอร์ให้กับแพ็คเก็ตข้อมูลหลังจากมี ACK ตอบกลับมาจากทุกผู้รับแล้ว ข้อเสียของวิธี ACK คือผู้ส่งต้องรับ ACK แพ็คเก็ตทั้งหมดจากผู้รับ ซึ่งจะทำให้จำกัดความเป็น Scalability ได้ ในการทดสอบโดยใช้ NAK จะแตกต่างกับวิธีแรก ผู้รับจะส่ง NAK กลับไปยังผู้ส่งในกรณีเดียวคือต้องการทำการส่งใหม่อีกครั้ง (Retransmission) ซึ่งจะลดจำนวนแพ็คเก็ตที่มากเกินไป แต่มีข้อเสียคือสมาชิกในระบบจะต้องสามารถเข้าใจการทำมัลติคาสท์แบบ NAK เพื่อสามารถสร้าง Reliable Multicast Service ซึ่งวิธีการนี้จะเปลืองหน่วยความจำมาก

2.13 Ring algorithm

เป็นอัลกอริธึม [26] ที่ใช้ในการเลือก Leader ของเซิร์ฟเวอร์ที่มีจัดการในลักษณะวงแหวน (Ring) ซึ่งวิธีการนี้จะไม่ใช้ Token อัลกอริธึมนี้มีสมมติฐานว่าสมาชิกแต่ละเครื่องจะติดต่อกันในลักษณะเรียงตามลำดับตามวงแหวน โดยสมาชิกแต่ละเครื่องจะรู้ว่าใครเป็นสมาชิกที่อยู่ถัดไป เมื่อสมาชิกใดพบว่า Leader ไม่ทำงาน มันจะสร้าง ELECTION Message ซึ่งประกอบด้วย ค่าลำดับความสำคัญของผู้ส่ง และจะส่งข้อความนี้ไปยังสมาชิกที่อยู่ถัดไป หากว่าตัวนี้เกิดล่มลง ผู้ส่งจะข้ามตัวนี้ไป และจะทำการส่งต่อยังตัวถัดไป หรือตัวก่อนหน้านี้ จนกระทั่งพบสมาชิกที่กำลังทำงานอยู่ ซึ่งในแต่ละขั้นตอนนั้น ผู้ส่งจะเพิ่มค่าลำดับความสำคัญของตัวเองลงบนรายการ (List) ในข้อความที่ส่งนั้น ส่งผลให้เกิดการการสร้าง Candidate ในตัวเองที่จะถูกคัดเลือกให้เป็น Leader ของระบบ ผลที่สุดแล้วข้อความก็จะกลับมายังผู้ส่งที่เป็นผู้ตั้งต้นข้อความนี้ หลังจากนั้นก็จะมีการวิเคราะห์ข้อความที่รับกลับมาซึ่งมีค่าลำดับความสำคัญของตัวเองอยู่ด้วยมันก็จะทราบว่าใครเป็น Leader ของระบบ หากว่าตัวมันเองมีค่าลำดับความสำคัญสูงสุดมันก็จะเปลี่ยนสถานะตัวเองเป็น Leader ณ จุดนี้ Leader ก็จะส่งข้อความที่บอกถึงสถานะตัวเองไปยังสมาชิกในระบบให้รับทราบว่าใครเป็น Leader ใหม่ เมื่อข้อความได้วนกลับมาอีกครั้งก็จะถูกเอาออกจากระบบ และสมาชิกทั้งหมดก็จะกลับมาทำงานตามสถานะตัวเองอีกครั้ง ภาพที่ 2-15 แสดงการทำงานของริงอัลกอริธึม



ภาพที่ 2-15 Ring Algorithm (อ้างอิงจาก [26])

ในกรณีที่มีสมาชิกสองตัวจากภาพคือสมาชิกที่มีค่าลำดับความสำคัญเป็น 2 และ 5 ค้นพบพร้อมกันว่า Leader ซึ่งในที่นี้คือ 7 ได้ล้มลง แต่ละตัวก็จะสร้าง ELECTION Message และแต่ละตัวก็จะเริ่มส่งข้อความไปยังสมาชิกถัดไปเรื่อย ๆ โดยไม่ส่งผลกระทบซึ่งกันและแน่นอนว่าแต่ละการส่งต่อข้อความของแต่ละสมาชิกจะต้องใส่ค่าลำดับความสำคัญไว้ในข้อความที่จะส่งทุกครั้ง ซึ่งในที่สุดแล้วข้อความทั้งสองก็จะเดินทางจนครบวงและทั้ง 2 และ 5 ก็จะพิจารณารายการค่าลำดับความสำคัญของตัวเองเพื่อหาว่าใครสมควรเป็น Leader ซึ่งในที่สุดแล้วก็จะได้ค่าลำดับความสำคัญที่ตรงกันซึ่งเป็นค่าสูงสุดของสมาชิกในระบบที่ยังคงทำงานอยู่

บทที่ 3
การวิเคราะห์ระบบ

เนื่องจากในบทที่สองได้กล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง ซึ่งได้กล่าวถึงการจัดการเซสชันในแบบต่าง ๆ และ งานวิจัยที่กล่าวถึงการสร้างระบบเครือข่ายที่มีเสถียรภาพ ดังนั้นในบทนี้จะทำการวิเคราะห์และเปรียบเทียบ โมเดลหรือหลักการการทำงานของแต่ละงานวิจัย ในแง่ของข้อดีและข้อเสียของแต่ละงานวิจัย ทั้งนี้จะแบ่งเป็นการเปรียบเทียบงานวิจัยในส่วนของการจัดการเซสชัน สุดท้ายจะกล่าวถึงงานวิจัยในส่วนของระบบเครือข่ายที่มีเสถียรภาพ

3.1 การเปรียบเทียบงานวิจัยในส่วนของการจัดการเซสชัน

3.1.1 การจัดการ เซสชัน ใน TCP/IP และ HTTP

การนำ Servlet มาสร้างเซสชัน เป็นการนำเทคนิค Cookie มาจัดการเซสชันโดยได้รวมไว้ในเว็บเพจแบบไดนามิกส์ สร้างกลุ่มของคลาสเพื่อสร้าง Cookie Object โดย Cookie นี้จะส่งไปกับบราวเซอร์ และมีการจัดเก็บไว้ในเครื่องของไคลเอนท์ด้วย ตารางที่ 3-1 แสดงการเปรียบเทียบข้อดีข้อเสียของการจัดการเซสชันใน TCP/IP และ HTTP

ตารางที่ 3-1 การเปรียบเทียบข้อดีข้อเสียของการจัดการเซสชันใน TCP/IP และ HTTP

ข้อดี	ข้อเสีย
เนื่องจากเป็นข้อมูลขนาดเล็กและอยู่ในรูปแบบข้อความ (Plain Text) เพื่อความสะดวกต่อการนำไปใช้งาน เนื่องจากวิธีนี้ไม่เปลืองทรัพยากรมากจึงเป็นที่นิยมในการใช้กับทุกบราวเซอร์	เนื่องจากข้อมูลของการทำธุรกิจบางอย่างการต้องการความเป็นส่วนตัว เช่นเครดิตการ์ดของผู้ใช้ ซึ่งไม่สมควรจะให้ผู้อื่นทราบหรือสามารถเปิดดูได้โดยง่าย แต่เทคนิคนี้มีการบันทึกข้อมูลการทำ Cookie ไว้ในเครื่องไคลเอนท์ด้วย จึงไม่เหมาะสมสำหรับความปลอดภัยในพาณิชย์ธุรกิจ

3.1.2 การจัดการเซสชันบน SOAP

SOAP ให้การอินเทอร์เฟสที่มีประสิทธิภาพสำหรับเว็บเซอร์วิส ยอมให้ผู้ใช้สร้างเมธอดที่มีความซับซ้อนในการเรียกระหว่างเว็บแอพพลิเคชันที่มีความหลากหลาย และทำงานบนแพลตฟอร์มที่ต่างกัน จากโมเดล [25] ได้นำเสนอโมเดลการจัดการเซสชัน และการพิสูจน์ตน (Authentication) โดยมีโมเดลที่ใช้ในการจัดการเซสชันแบบรวมศูนย์ที่เรียกว่า Session Authority (SA) เป็นส่วนจัดกลางเซสชันของระบบ โดยใช้ SOAP เป็นตัวส่งข้อมูลการทำเซสชัน และใช้ในการเพิ่มการพิสูจน์ตนเพื่อสร้างความปลอดภัย ในตารางที่ 3-2 จะแสดงถึงการเปรียบเทียบข้อดีข้อเสียของการจัดการเซสชันบน SOAP

ตารางที่ 3-2 การเปรียบเทียบข้อดีข้อเสียของการจัดการเซสชันบน SOAP

ข้อดี	ข้อเสีย
มีการสร้างการจัดการเซสชันและความปลอดภัยที่สะดวก ซึ่งสามารถเพิ่มชนิดของความปลอดภัยโดยการเพิ่มไว้ใน SOAP Message ได้	มีส่วนที่เรียกว่า Session Authority (SA) ซึ่งจะทำหน้าที่สร้าง Session ID และให้ Secret Key แก่ระบบ ดังนั้นทุกสมาชิกในระบบจะต้องติดต่อกับส่วนนี้ก่อนเสมอจึงจะทำงานร่วมกันได้ หากเซิร์ฟเวอร์หรือ SA หลักของระบบนี้ล้มเหลวลง การดำเนินการของระบบก็จะต้องหยุดชะงักลงด้วย

3.1.3 การจัดการเซสชันและทรานแซคชันให้กับเว็บเซอร์วิสโดยการใช้ SIP

เป็นการนำโพรโทคอล SIP มาจัดการเซสชันของระบบ โดยจะมีเซิร์ฟเวอร์ที่เรียกว่า User Agent Server (UAS) เป็นตัวสร้าง Session ID ให้กับ User Agent Client (UAC) และให้บริการการตรวจสอบสถานะการลงทะเบียน Session ID ของ UAC ให้แก่เว็บเซอร์วิส และนำ Session ID มาจัดการทรานแซคชันของระบบ สามารถวิเคราะห์ข้อดีข้อเสียของโมเดลได้ในตารางที่ 3-3

ตารางที่ 3-3 การเปรียบเทียบข้อดีข้อเสียของการจัดการเซสชันและทรานแซคชันให้กับเว็บเซอร์วิส โดยใช้ SIP

ข้อดี	ข้อเสีย
การจัดการเซสชันของระบบจะใช้งานร่วมกับ SOAP โดยการใส่ Session ID ไว้ใน SOAP Header ของทุก ๆ SOAP Message ดังนั้นจึงไม่มีปัญหาในส่วนของความไม่เข้ากันของแพลตฟอร์ม และนอกจากนั้นยังสามารถเพิ่มความปลอดภัยเช่นการทำการพิสูจน์ตนโดยจะเพิ่มลงใน SOAP Message	โมเดลนี้ มีการจัดกลางแบบศูนย์กลางทุกสมาชิกในระบบจะต้องใช้บริการกับ UAS ดังนั้น จึงอาจพบปัญหาการล้มเหลวของตัวจัดการหลัก หาก UAS ล้มเหลวก็จะส่งผลต่อการทำธุรกรรมทั้งหมดได้เช่นกัน

3.1.4 การจัดการเซสชันในงานด้านบริการทางธุรกิจในระบบ SOA

จากรายละเอียดของโครงสร้างและการทำงานของระบบจะพบว่า การจัดการเซสชันจะแบ่งเป็นสองส่วนได้แก่ การจัดการเซสชันในพื้นที่ซึ่งเป็นการจัดการในลักษณะศูนย์กลาง และการจัดการเซสชันในระยะไกลซึ่งถือได้ว่าเป็นระบบการจัดการแบบกระจายศูนย์ ระบบนี้จึงมีการจัดการทั้งสองแบบคือแบบศูนย์กลาง และแบบกระจายศูนย์ โมเดลดังกล่าวนั้นมีระบบการจัดการระยะไกลซึ่งถือว่าเป็นตัวจัดการหลักขององค์กร ซึ่งตัวจัดการเซสชันในพื้นที่จะต้องติดต่อด้วยในกรณีที่ต้องการเรียกดูข้อมูลจากระบบส่วนกลางนี้ และระบบส่วนกลางนี้สร้างมาจากเว็บเซอร์วิสหลาย ๆ ตัวที่ทำงานร่วมกันเพื่อสร้างการให้บริการ สามารถแสดงการวิเคราะห์โดยการเปรียบเทียบข้อดี ข้อเสียได้ในตารางที่ 3-4

ตารางที่ 3-4 การเปรียบเทียบข้อดีข้อเสียของการให้การจัดการเซสชันในงานด้านบริการทางธุรกิจ

ข้อดี	ข้อเสีย
การออกแบบระบบส่วนกลางจะเป็นลักษณะการกระจายศูนย์ ดังนั้นจะไม่เกิดปัญหาในเรื่องของการทำงานหนัก หรือเกิดข้อมูลคับคั่งได้ ทั้งนี้ยังลดปัญหาในเรื่องของความไม่ต่อเนื่องของการทำงานอันเกิดจากความล้มเหลวของระบบกลาง	การจัดการเซสชันในแต่ละเซิร์ฟเวอร์ขององค์กรย่อยที่เป็นส่วนประกอบหนึ่งขององค์กรหลัก ยังมีลักษณะการจัดการแบบรวมศูนย์ ดังนั้นจึงอาจเกิดปัญหาการล่มของเซิร์ฟเวอร์หลักซึ่งทำหน้าที่เชื่อมต่อระหว่างองค์กรย่อยกับเซิร์ฟเวอร์ขององค์กรหลัก

3.2 เปรียบเทียบงานวิจัยในส่วนของการสร้างความมีเสถียรภาพของระบบ

3.2.1 โพรโทคอลการจัดการเซสชันแบบมัลติคาสท์

เป็นการออกแบบระบบเพื่อจำลองการสร้างการมัลติคาสท์ข้อมูลที่เป็นแอพพลิเคชันที่มีจำนวนมากไปยังเซิร์ฟเวอร์ในระบบ โดยจะมีการทำมัลติเพล็กส์ และดีมัลติเพล็กส์ข้อมูลในขณะส่งและในขณะรับตามลำดับ ในที่นี้ได้แสดงไว้จำนวนสี่ตัว โดยที่จะมีการติดต่อกันในลักษณะของวงแหวนเสมือน (Virtual Ring) ดังนั้น ข้อมูลจะถูกส่งต่อ ๆ ไปยังเซิร์ฟเวอร์ข้างเคียงของตัวส่งจนกระทั่งวนกลับมาที่ผู้ส่ง โดยใช้หลักการของ Reliable Multicast แบบตอบกลับ ACK Based Protocol และใช้ฮาร์ทบีทอัลกอริธึมในการตรวจจับการรับส่งข้อมูลภายในเวลาที่กำหนด สามารถวิเคราะห์ข้อดีข้อเสียของระบบได้ในตารางที่ 3-5

ตารางที่ 3-5 การเปรียบเทียบข้อดีข้อเสียของโพรโทคอลการจัดการเซสชันแบบมัลติคาสท์

ข้อดี	ข้อเสีย
เป็นการสร้างระบบแบบกระจายศูนย์ แสดงให้เห็นถึงการกระจายข้อมูลแบบมัลติคาสท์โดยกำหนดตำแหน่งของผู้รับในลักษณะวงแหวน ซึ่งมีข้อดีในด้านของการลด Transport Layer Overhead เนื่องจากการทำ มัลติเพล็กส์ และดีมัลติเพล็กส์ข้อมูล มีความเป็น Scalability สูง และให้ความสะดวกไม่ยุ่งยากในการส่งข้อมูลและการสร้างระบบ	ไม่เหมาะสมกับการนำไปใช้กับองค์กรที่มีขนาดใหญ่ และอยู่ห่างไกล เนื่องจากเป็นระบบที่ออกแบบมาสำหรับใช้งานระบบเครือข่ายเฉพาะในพื้นที่ เช่น ระบบเครือข่าย LAN หรือ WAN

3.2.2 การออกแบบและการใช้ฮาร์ทบีทในสภาพแวดล้อมแบบมัลติแมชชีน (Multi-Machine)

เป็นการนำฮาร์ทบีทอัลกอริธึมมาใช้ตรวจจับการทำงานของ Node หรือเซิร์ฟเวอร์ที่มีอยู่ในระบบ หากเซิร์ฟเวอร์ตัวใดที่ทำหน้าที่หลักของระบบเกิดล่มไป จะใช้บูลลีย์อัลกอริธึมในการเลือกเซิร์ฟเวอร์หลักตัวต่อไป ซึ่งสามารถวิเคราะห์ข้อดีข้อเสียได้ในตารางที่ 3-6

ตารางที่ 3-6 การเปรียบเทียบข้อดีข้อเสียของการใช้ ฮาร์ทบีทในสภาพแวดล้อมแบบมัลติแมชชีน (Multi-machine)

ข้อดี	ข้อเสีย
ทำให้ได้ระบบที่มี Scalability และขั้นตอนการตรวจจับโดยใช้ฮาร์ทบีทนี้ต้องการเพียงไม่กี่ขั้นตอน ซึ่งช่วยลด Overhead และการจัดการระบบจะไม่ซับซ้อน	การใช้ฮาร์ทบีทเพื่อประสานเวลาในการส่งและรับข้อมูลกันระหว่างเซิร์ฟเวอร์ต่าง ๆ ในระบบจำเป็นต้องมีการซิงโครไนซ์กันในเรื่องของเวลา แต่เซิร์ฟเวอร์ต่างๆ นั้นไม่มี Clock ที่ใช้ร่วมกัน จำเป็นต้องอาศัยโพรโทคอลอื่นมาจัดการ และอาจต้องจัดการในเรื่องของการหน่วงเวลา (Delay) และปัญหาการคับคั่งในเครือข่าย

3.2.3 การให้บริการการทำซ้ำข้อมูลแบบกระจายในระบบ Business to Business (B2B)

โมเดลนี้ประกอบด้วย Replication Nodes จำนวนหลายตัวที่มีหน้าที่ให้การทำซ้ำข้อมูล โดยจะมีเซิร์ฟเวอร์หลักทำงานตามการร้องขอการทำซ้ำ แต่เมื่อตัวหลักไม่สามารถทำงานได้ก็จะเลือกตัวใหม่ทำงานแทนโดยใช้อัลกอริธึมในการคัดเลือกตัวเซิร์ฟเวอร์หลัก เพื่อให้เครือข่ายติดต่อกันได้อย่างสมบูรณ์ โดยข้อดีและข้อเสียของโมเดลนี้ได้อธิบายไว้ในตารางที่ 3-7

ตารางที่ 3-7 การเปรียบเทียบข้อดีข้อเสียของการให้บริการการทำซ้ำข้อมูลแบบกระจายในระบบ Business to Business (B2B)

ข้อดี	ข้อเสีย
มีการกระจายข้อมูลแบบ Load Balancing สามารถเข้าถึงข้อมูลได้รวดเร็ว ถ้าตัวใดตัวหนึ่งล่มไปจะสามารถกู้ระบบได้ทันที ทำให้มั่นใจได้ว่าข้อมูลทุกข้อมูลไม่สูญหาย เนื่องจากจะทำการเก็บข้อมูลไว้ในเซิร์ฟเวอร์หลัก	การสำรองข้อมูลไว้ในเซิร์ฟเวอร์มากกว่าหนึ่งตัวเป็นการเปลืองแบนด์วิธอย่างมาก ในกรณีที่มีการเปลี่ยนแปลงข้อมูลในฐานข้อมูลในเซิร์ฟเวอร์หลักก็จะต้องทำการเปลี่ยนแปลงข้อมูลในเซิร์ฟเวอร์อื่น ๆ นอกจากนั้นยังเปลืองพื้นที่ในการจัดเก็บด้วย

บทที่ 4
การออกแบบระบบ

จากบทที่ 3 ได้กล่าวถึงการวิเคราะห์ข้อดีข้อเสียของระบบต่าง ๆ จะเห็นได้ว่าสามารถนำมาสร้างระบบที่มีการจัดการเซสชันที่มีเสถียรภาพได้ โดยใช้ข้อดีของงานวิจัยต่าง ๆ ที่ได้วิเคราะห์มาสร้างระบบให้ได้ตามวัตถุประสงค์ ดังนั้นในบทที่ 4 นี้จะกล่าวถึงการออกแบบระบบที่นำเอาทฤษฎีและการวิเคราะห์ดังกล่าวมาใช้

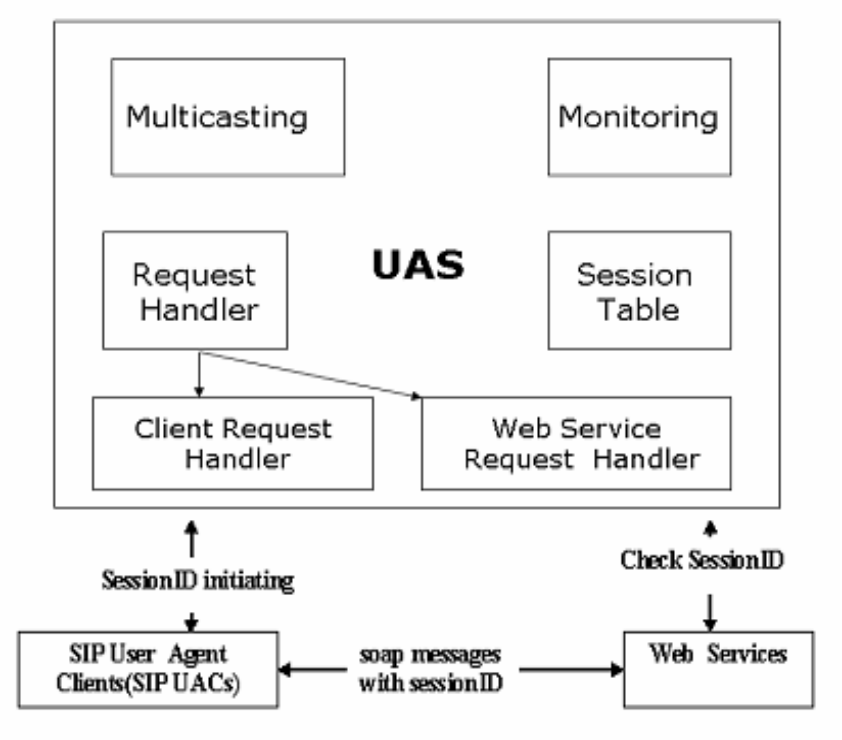
เนื่องจากวัตถุประสงค์ของงานวิจัยนี้ ต้องการออกแบบการจัดการเซสชันที่มีเสถียรภาพให้กับไคลเอนท์ที่จะเรียกใช้บริการของเว็บเซอร์วิสที่มีการติดต่อกันมากกว่าหนึ่งเว็บเซอร์วิส การออกแบบระบบของงานวิจัยจะประกอบด้วย User Agent Server (UAS) จำนวน 4 ตัว แต่จะมีหนึ่งตัวที่ทำหน้าที่หลักในการกำหนด Session ID สร้างการติดต่อระหว่าง User Agent Client (UAC) และเว็บเซอร์วิส ทำการตรวจสอบและให้การรับรอง Session ID ต่อเว็บเซอร์วิสเมื่อมีการสอบถาม โดยที่ Session ID ที่เกิดขึ้นนี้ UAC จะนำไปใช้ในการติดต่อกับเว็บเซอร์วิสโดยใส่ลงไปในทุก ๆ SOAP Message ที่จะติดต่อกับเว็บเซอร์วิส และเว็บเซอร์วิสทุกตัวจะนำ Session ID นี้ไปตรวจสอบกับ UAS ด้วย ทั้งนี้ก็เพื่อให้แน่ใจได้ว่าไคลเอนท์หรือ UAC นี้เป็นผู้ร่วมเซสชันเดียวกัน

เนื่องจากระบบจะต้องมีความเสถียรภาพ ดังนั้น ในการที่จะทำให้ระบบมีการทำงานได้อย่างต่อเนื่อง จะต้องมีระบบตรวจสอบการทำงาน (Monitoring) ของ UAS อยู่ตลอดเวลา แต่หากการทำงานของ UAS ตัวหลักล้มเหลวลงจะต้องใช้ ริงอัลกอริธึมมาใช้ในการเลือก UAS ที่ทำหน้าที่หลักตัวใหม่ขึ้นมาแทน

ในบทนี้จะเริ่มจากการกล่าวถึงการออกแบบระบบที่เรียกว่า การจัดการเซสชันแบบกระจายโดยใช้โพรโทคอล SIP จากนั้นจะกล่าวถึงสถาปัตยกรรมรวมของระบบ และการแสดงการทำงานของระบบในส่วนต่าง ๆ

4.1 กระบวนการในการสร้างการจัดการเซสชันแบบกระจายโดยใช้โพรโทคอล SIP

ระบบการจัดการเซสชันแบบกระจายโดยใช้โพรโทคอล SIP นี้จะประกอบด้วย UAS จำนวน 4 ตัวซึ่งทำหน้าที่หลักในการจัดการเซสชัน และมี UAC ซึ่งเป็นไคลเอนท์ของ UAS ที่ต้องการใช้บริการจากเว็บเซอร์วิส โดยระบบนี้จะมีเว็บเซอร์วิสสองตัวที่ทำงานแบบมัลติเว็บเซอร์วิสดังแสดงในภาพที่4-1



ภาพที่ 4-1 สถาปัตยกรรมรวมของระบบการจัดการเซสชันแบบกระจายโดยใช้โพรโทคอล SIP

4.1.1 SIP User Agent Server (UAS)

เป็นเซิร์ฟเวอร์ที่ส่ง SIP Response Message ตอบกลับไปยัง UAS โดยมีหน้าที่หลักคือสร้าง Unique Session (Session ID) ให้กับ UAC เมื่อมีการร้องขอใช้บริการจากเว็บเซอร์วิส โดยที่ Session ID ได้มาจากการนำข้อความของ Request Message ของ UAC มาสร้าง Session ID ข้อมูลที่ใช้ในการสร้าง Session ID ได้แก่ “@ IP address” ของไคลเอนท์ซึ่งส่งมากับ SIP Request Message และ UAS ยังต้องบันทึกเวลาที่ UAC ได้ติดต่อขอ Session ID แล้วนำข้อมูลทั้งสองนี้มาใช้ในการสร้าง Session ID และเมื่อมีการสร้างเซสชันแล้ว เซสชันที่ถูกสร้างขึ้นทุกตัวจะถูกบันทึกไว้ใน Session Table ของ UAS ด้วย

UAS จะยืนยัน Session ID เมื่อเว็บเซอร์วิสร้องขอการตรวจสอบ Session ID ของ UAC เนื่องจากการที่เว็บเซอร์วิสจะอนุญาตให้ UAC ใช้บริการของเว็บเซอร์วิสนั้น จะต้องมีการตรวจสอบ Session ID ที่ได้จาก SOAP Header ของ SOAP Message ที่ UAC ได้ส่งมาเพื่อนำไปตรวจสอบกับ UAS เทียบกับ Session ID ที่ได้อยู่ใน Session Table หากว่าตรงกัน เว็บเซอร์วิสนั้นจึงจะอนุญาตให้ใช้บริการได้

UAS แต่ละตัวในระบบจะมีองค์ประกอบหลักของการทำงานอยู่ 4 ส่วนดังนี้ Session Table, Request Handler, Monitoring และ Multicasting สามารถอธิบายแต่ละองค์ประกอบได้ดังต่อไปนี้

4.1.1.1 Request Handler

แบ่งเป็นสองส่วนคือ

ก) Client Request Handler

ทำหน้าที่ตอบรับการร้องขอ Session ID จาก UAC ซึ่งตัวอย่างของ SIP Request ที่ไคลเอนท์ส่งมา และ SIP Response ได้ตอบกลับจะมีรูปแบบดังนี้

Request Message ประกอบด้วยชื่อเมธอด (ตามตารางที่ 2-1) ตามด้วยเครื่องหมาย @ ตำแหน่งไอพีแอดเดรสของ UAC และสุดท้ายคือ SIP version ซึ่งรูปแบบการส่งข้อความเป็นดังนี้

INVITE@UAC's IP Address SIP version

Response Message ประกอบด้วย SIP Version ตามด้วยเครื่องหมายขีดล่าง และ Response Code (ตามภาพที่ 2-7) ซึ่งมีรูปแบบดังนี้ "SIP Version_180" หมายถึงได้รับ Request แล้วและกำลังจะดำเนินการอยู่ อีกตัวอย่างเช่น "SIP Version_200" หมายถึง ดำเนินการเสร็จแล้วพร้อมสำหรับ Request Message ต่อไป

ข) Web Service Request Handler

ตอบรับการติดต่อจากเว็บเซอร์วิสและทำหน้าที่ให้การยืนยันการตรวจสอบ Session ID ต่อเว็บเซอร์วิส

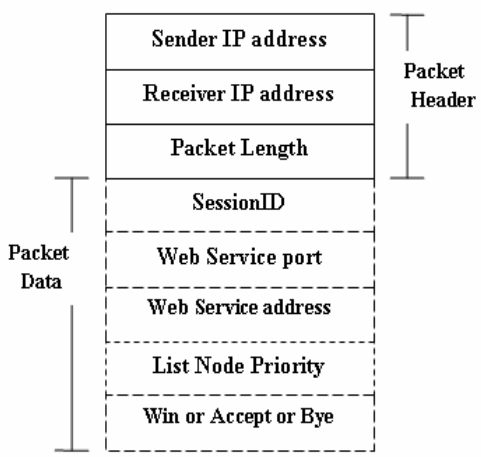
4.1.1.2 Session Table

เป็นฐานข้อมูลที่บันทึก Session ID และ IP Address ของเว็บเซอร์วิส รวมถึงหมายเลขพอร์ตของเว็บเซอร์วิสที่เปิดให้บริการ เมื่อเซสชันเกิดขึ้นหลังการติดต่อกันระหว่าง UAS และ UAC แล้ว UAS จะบันทึก Session ID ไว้ใน Session Table เมื่อเว็บเซอร์วิสติดต่อตรวจสอบ Session ID ของไคลเอนท์กับ UAS แล้ว UAS จะจัดเก็บ IP Address ของเว็บเซอร์วิส และพอร์ตของเว็บเซอร์วิสไว้ในฐานข้อมูลด้วย ทั้งนี้เพื่อใช้ในการติดต่อจนกระทั่งถึงในการยกเลิกเซสชัน

4.1.1.3 Multicasting

ในระบบนี้ได้กำหนดให้ UAS ทั้งหมดติดต่อกันแบบวงแหวนเสมือน UAS แต่ละตัวจะติดต่อกับ UAS ตัวถัดไปในวงแหวน และจะทำการมัลติคาสท์แพ็คเก็ตแบบทิศทางเดียว (Unidirection) ไปยังตัวถัดไป โดยที่ Multicast Packet จะประกอบด้วย Packet Header และ Packet Data ซึ่งจะมี IP Address ของผู้รับ และขนาดหรือความยาวของแพ็คเก็ต (Packet Length) ในส่วนของ Packet Data จะมี Session ID ไอพีแอดเดรสของเว็บเซอร์วิส หมายเลขพอร์ตที่เว็บเซอร์

ให้บริการติดต่อ และข้อความบางส่วนที่ต้องการส่งเพิ่มเติมเช่น (Win, Accept หรือ Bye) ดังแสดงในภาพที่ 4-2



ภาพที่ 4-2 Multicast Packet

4.1.1.4 Monitoring

ทำหน้าที่ ตรวจสอบการทำงานของ UAS และกำหนดเวลาในการส่งแพ็คเก็ตซึ่งจะใช้ฮาร์ทบีทอัลกอริธึมในการควบคุมการทำงานของแต่ละ UAS

4.1.2 UAC

ทำหน้าที่เป็นไคลเอนท์ของ UAS และเว็บเซอร์วิส โดยจะส่ง Request Messages ซึ่งประกอบด้วย INVITE ในกรณีที่ต้องการร่วมเซสชันไปยัง UAS และจะส่งข้อความ ACK เพื่อบอกกับ UAS ในกรณีที่ได้รับ Session ID แล้ว

4.1.3 Web Service

ในระบบที่ออกแบบนี้ ได้กำหนดให้เป็นเว็บเซอร์วิสอย่างง่ายที่ทำงานร่วมกันระหว่างเว็บเซอร์วิส และให้บริการแก่ไคลเอนท์บนเครือข่าย ซึ่งเว็บเซอร์วิสจะใช้ WSDL ในการอธิบายการอินเทอร์เฟส และใช้ SOAP ในการการสื่อสารให้กับ UAC และก่อนที่ UAC จะใช้บริการเว็บเซอร์วิสจะต้องติดต่อขอ Session ID จาก UAS ก่อน

4.2 การออกแบบการทำงานของระบบ

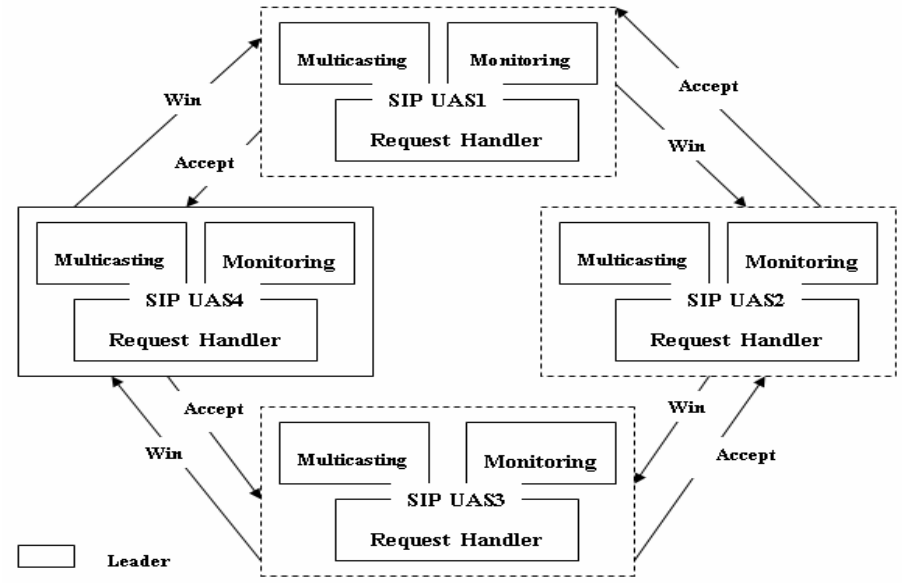
สามารถแยกการทำงานของแต่ละส่วนในระบบในได้ดังต่อไปนี้

4.2.1 การทำ Monitoring

โครงสร้างเครือข่ายของระบบนี้จะเป็นลักษณะวงแหวนเสมือน โดยจะใช้กลไกฮาร์ทบีทเพื่อตรวจสอบการล้มเหลวของ UAS หลัก ในงานวิจัยนี้กำหนดให้ใช้ฮาร์ทบีทไทม์ (Heartbeat Time) เพื่อควบคุมการไหลของแพ็คเก็ต (Packet Flow) โดยจะมีการกำหนดค่าฮาร์ทบีทไทม์ในแต่ละ UAS ไว้ก่อนแล้ว แต่ละ UAS จะส่งสถานะของตัวเองเป็นช่วงจังหวะตามเวลาที่กำหนดไว้ไปยัง UAS ถัดไป หากว่า UAS ตัวใดตรวจพบการล่มของตัวใดตัวหนึ่งจากการตรวจสอบจากเวลาจากนั้นจะบอกไปยัง UAS ทางซ้ายและขวา ให้หยุดการติดต่อไปยัง UAS ที่มีปัญหา และเริ่มสร้างการติดต่อระหว่างตัวใหม่นี้ ในกรณีที่ตัวที่ล้มไปเป็น Leader UAS ก็จะมีการคัดเลือกเซิร์ฟเวอร์หลักขึ้นมาแทนที่โดยได้นำหลักการของริงอัลกอริธึมมาใช้ในการเลือก UAS ตัวใหม่

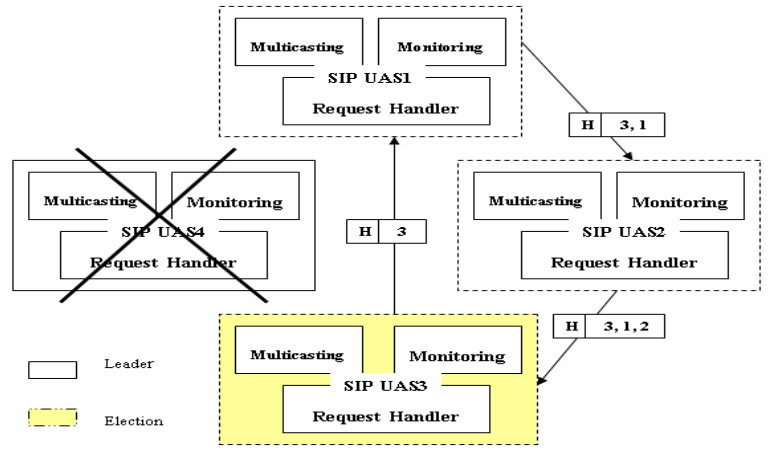
4.2.2 การทดสอบ Leader Algorithm

กำหนดให้ UAS แต่ละตัวได้รับการกำหนดค่าลำดับความสำคัญ (Node Priority) ของ UAS ไว้ก่อนแล้วโดย Leader จะมีค่าลำดับความสำคัญสูงสุด แต่ละ UAS จะทำการมัลติคาสท์แพ็คเก็ตเป็นวงแหวนในลักษณะทิศทางเดียว (Unidirection) ไปยัง UAS ถัดไปและตอบกลับ Positive Acknowledge (Accept) ไปยัง UAS ที่เป็นผู้ส่ง จากภาพที่ 4-3 แสดงให้เห็นว่า UAS 4 เป็น Leader ก็จะมัลติคาสท์ข้อความ “Win” เป็นช่วงจังหวะ เพื่อบอกสถานะตัวเองไปยัง UAS 1



ภาพที่ 4-3 รูปแบบการทำมัลติคาสท์

เมื่อ UAS 1 ได้รับแพ็คเก็ตนี้ตามเวลาแล้วจะส่งข้อความตอบกลับ (Accept) ไปยังผู้ส่งเพื่อให้ผู้ส่งมั่นใจได้ว่าผู้รับได้รับข้อความแล้ว แต่หาก UAS 3 ไม่ได้รับข้อความ “Win” ตามกำหนดเวลาก็จะตั้งสมมติฐานว่า UAS 4 ล้มเหลว จากนั้น UAS 3 จะเริ่มการทำ Election เพื่อหา Leader ตัวใหม่ โดย UAS 3 จะตั้งตัวเองเป็นคู่แข่งเซิร์ฟเวอร์หลัก (Candidate New Leader) ตามภาพที่ 4-4



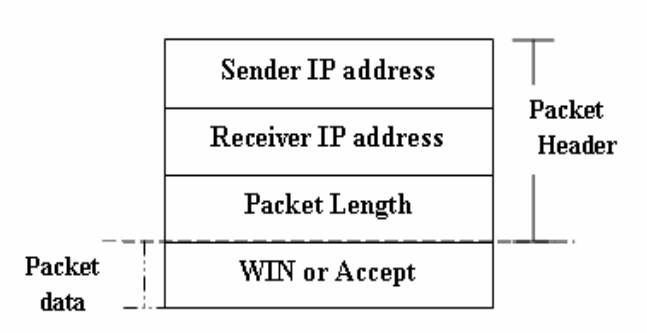
ภาพที่ 4-4 จำลองการล้มเหลวของ UAS4 ซึ่งเป็น Leader ระบบ

UAS 3 จะมัลติคาสท์ “Elect” Message Packet ซึ่งประกอบด้วยค่าลำดับความสำคัญของ UAS ใน Data Packet ไปยัง UAS 1 จากนั้น UAS 1 จะแทรกค่าลำดับความสำคัญของตัวองไว้บน “Elect” แพ็คเก็ต ดังนั้นรายการลำดับความสำคัญ (List Node Priority) จึงมีค่า <3, 1> UAS 1 มัลติคาสท์ Packet นี้ไปยัง UAS 2 เมื่อ UAS 2 ได้รับแพ็คเก็ตที่ส่งมาก็จะแทรกค่าลำดับความสำคัญของตัวเองเช่นเดียวกัน ดังนั้นรายการลำดับความสำคัญจึงเป็น <3,1,2> จากนั้น UAS 2 ก็จะมัลติคาสท์แพ็คเก็ตต่อไปยัง UAS 3 ซึ่งเป็นผู้ตั้งต้นส่งข้อความ “Elect” นี้ ดังนั้นเมื่อ UAS 3 ได้รับแพ็คเก็ตที่ส่งกลับมาที่ตน ก็จะรู้ค่าลำดับความสำคัญของทั้งหมดและของตัวเองจากแพ็คเก็ตนี้ และสามารถสรุปได้ว่า ขณะนี้ UAS 3 มีค่าลำดับความสูงสุด ดังนั้น UAS 3 จึงกลายเป็น Leader ซึ่งก็จะมัลติคาสท์ข้อความ “Win” บอกไปยัง UAS ตัวอื่นต่อไป

4.2.3 การทำมัลติคาสท์

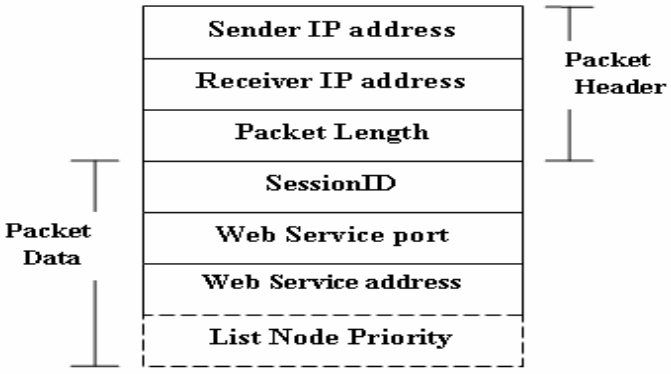
ในระบบนี้จะมีการมัลติคาสท์ข้อความอยู่สองแบบ อันดับแรกคือ Leader UAS จะมัลติคาสท์ “Win” ไปยังตัว UAS ถัดไป เมื่อผู้รับได้รับแล้วก็จะตอบกลับ โดยทำการมัลติคาสท์ “Accept” กลับมายังผู้ส่ง ภาพที่ 4-5 แสดงถึงการมัลติคาสท์แพ็คเก็ต ซึ่งประกอบด้วยไอพีแอดเดรส

ของผู้ส่ง (Sender IP address) ไอพีแอดเดรสของผู้รับ (Receiver IP address) ขนาดหรือความยาวของแพ็คเก็ต (Packet length) และ Packet data (ซึ่งอาจจะเป็น Win หรือ Accept)



ภาพที่ 4-5 รูปแบบข้อความในการทำ Multicast State Packet

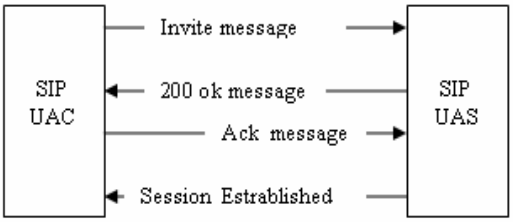
แบบที่สองคือ UAS ตัวหลักจะมัลติคาสท์ข้อมูลจาก Session Table ของตัวเองไปยัง UAS ถัดไป เมื่อ UAS ตัวใด (ยกเว้น Leader) ได้รับข้อมูลนี้ก็จะเพิ่มข้อมูลนี้ไว้ใน Session Table ของตัวเอง และหากกว่า UAC สิ้นสุดการกระทำก็จะทำการมัลติคาสท์ข้อความ “BYE” ไปด้วยซึ่งก็จะเป็นการสั่งให้ปรับปรุงข้อมูลใน Session Table ภาพที่ 4-6 แสดงถึงรูปแบบของการทำ Session ID Multicasting



ภาพที่ 4-6 รูปแบบข้อความในการทำ Session ID Multicasting

4.2.4 ขั้นตอนการเริ่มสร้างเซสชัน

แต่ละ UAC จะมี Client Request Handler เพื่อสร้างการสื่อสารระหว่าง UAC และ Leader UAS โดย UAS เปิด Server Socket เพื่อรับข้อความของ UAC จากภาพที่ 4-7

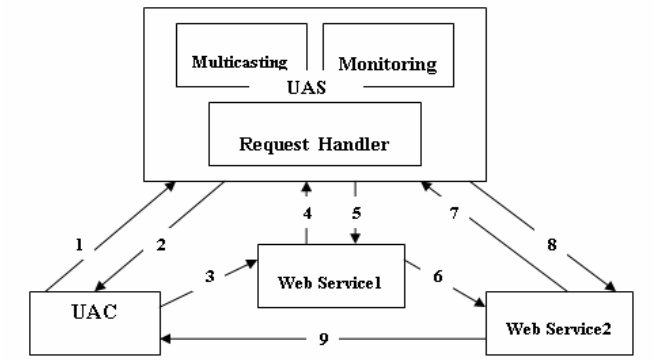


ภาพที่ 4-7 การเริ่มสร้าง Session ID

UAC จะได้รับ Session ID ซึ่งก็จะมีการบันทึกไว้ใน UAS ตัวอย่างเช่น สมมติให้ IP address ของ UAC คือ 202.44.40.13 ทำการจัดตั้ง (Setup) เซสชันกับ UAS ที่เวลา 5:06:30 am ในวันที่ 19 December 2006 ดังนั้น Session ID คือ 20061219050630@202.44.40.13 เมื่อไคลเอนท์ได้ใช้ UAC แล้ว UAC จะส่งข้อความ “INVITE” (INVITE@UAC IP address) ไปยัง UAS 4 ซึ่งเป็น Leader ดังนั้น UAS 4 จะส่งข้อความ 200 OK กลับมายัง UAC จากนั้น UAC ส่งข้อความ ACK ไปยัง UAS ดังนั้นเซสชันจึงเกิดขึ้นระหว่าง UAC และ UAS 4

4.2.5 การติดต่อสื่อสารของมัลติเว็บเซอร์วิสและ UAS หลักของระบบ

เป็นการจำลองการทำงานร่วมกันของ UAS UAC และ เว็บเซอร์วิส ซึ่งในภาพที่ 4-8 จะเป็นการจำลองเว็บเซอร์วิส 1 ให้เป็น Book Supplier และเว็บเซอร์วิส 2 เป็น Book Shipping จากภาพที่ 4-8 ขั้นตอนที่ 1 ถึง 2 ได้อธิบายไว้แล้วในหัวข้อที่ 4.2.4 ซึ่งเป็นการเริ่มสร้างเซสชันหรือที่เรียกว่า Session ID

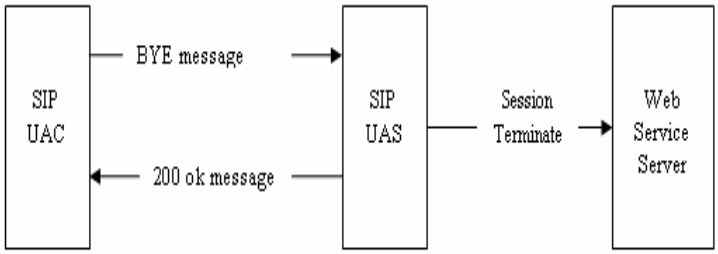


ภาพที่ 4-8 การติดต่อสื่อสารของมัลติเว็บเซอร์วิสและ UAS หลักของระบบ

หลังจากที่ได้รับ Session ID จาก UAS แล้ว (ตามลูกศร 1-2) UAC สามารถส่งแอพพลิเคชัน Request (Application Request) ซึ่งอยู่ในรูปข้อความ SOAP Message ที่ใส่ Session ID ไว้ใน Soap Header Message ไปยังเว็บเซอร์วิส 1 (ขั้นตอนที่ 3) เมื่อเว็บเซอร์วิส 1ได้รับ Request มันจะใช้ข้อมูลใน Soap Header เพื่อตรวจสอบไปยัง UAS โดยการส่งข้อความ “Check SessionID” ซึ่งจะมี Session ID ของ UAC ไปตรวจสอบ (ขั้นตอนที่ 4) ถ้า Session ID ตรงกันกับที่ UAS บันทึกไว้ใน Session Table แล้ว UAS จะส่งข้อความ “Session Has Setup” (ขั้นตอนที่ 5) กลับไปยัง เว็บเซอร์วิส 1 จากนั้นเว็บเซอร์วิส 1 จะบันทึก Session ID ไว้ด้วย ถ้าเว็บเซอร์วิส 1 ต้องการจะติดต่อกับเว็บเซอร์วิส 2 เพื่อเรียกใช้บริการต่อ ก็จะส่ง Soap Message (พร้อมกับ Session ID ไว้ใน SOAP Header) ไปยังเว็บเซอร์วิส 2 (ขั้นตอนที่ 6) เว็บเซอร์วิส 2 จะตรวจสอบ Session ID นี้กับ UAS (ขั้นตอนที่ 7-8) ถ้าตรงกันก็จะอนุมัติให้ใช้ และ เว็บเซอร์วิส 2 จะบันทึกไว้เช่นกัน จากนั้นเว็บเซอร์วิส 2 ก็จะสามารถติดต่อกับ UAC ได้เลย (ขั้นตอนที่ 9) ซึ่งทั้งเว็บเซอร์วิส 2 และ UAC จะมั่นใจได้ว่าต่างฝ่ายต่างก็เป็นผู้ทำธุรกรรมร่วมกัน

4.2.6 การทำการสิ้นสุดเซสชัน

เมื่อ UAC เสร็จสิ้นการทำงานกับเว็บเซอร์วิสแล้ว จะต้องส่งข้อความ Bye ทุกครั้งเพื่อทำการสิ้นสุดเซสชัน จากภาพที่ 4-9



ภาพที่ 4-9 การสิ้นสุดเซสชัน

จะเห็นได้ว่า นอกจากจะต้องลบ Session ID ใน UAS แล้วยังต้องทำให้เว็บเซอร์วิสแต่ละตัวทำการลบข้อมูล Session ID ที่ได้บันทึกไว้ด้วย ซึ่งทำได้จากการที่ UAS ส่งข้อความบอกแก่เว็บเซอร์วิสเซิร์ฟเวอร์เพื่อให้ทำการลบข้อมูลนี้ออกจากระบบ

บทที่ 5
สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลการวิจัย

งานวิจัยนี้ได้นำเสนอแนวคิดในเรื่องของการจัดการเซสชัน (Session Management) ซึ่งเป็นส่วนสำคัญในการจัดการธุรกรรมทางอิเล็กทรอนิกส์ โดยเริ่มจากการนำแนวคิดและวิธีในการจัดการเซสชันที่มีอยู่ในปัจจุบัน เช่นการจัดการเซสชันหลายอย่างมาเปรียบเทียบข้อดีข้อเสียของแต่ละโมเดล สามารถอธิบายได้ว่า ใน TCP/IP และ HTTP ซึ่งมีข้อเสียในด้านความปลอดภัยของข้อมูล จึงไม่เหมาะสมสำหรับการสร้างความเชื่อมั่นในระบบพาณิชย์ธุรกิจ [4, 5, 9] การจัดการเซสชันบน SOAP [25] ซึ่งมีข้อเสียคือเป็นการจัดการแบบศูนย์กลาง ซึ่งอาจพบปัญหา Leader Failure ได้ การจัดการเซสชันและทรานแซคชันให้กับเว็บเซอร์วิสโดยการใช้ SIP [24] ซึ่งก็มีข้อเสียในด้านการจัดการเซสชันแบบศูนย์กลาง ซึ่งอาจพบปัญหา Leader Failure ได้ การจัดการเซสชันในงานด้านบริการทางธุรกิจในระบบ SOA อาจเกิดปัญหา Leader Failure เช่นกัน จากข้อจำกัดการจัดการเซสชันของโมเดลต่าง ๆ ที่มีดังกล่าว จึงได้นำเสนอโมเดลการจัดการเซสชันแบบกระจายโดยใช้โพรโทคอล SIP ซึ่งมีการจัดการเซสชันแบบกระจายศูนย์ และมีคุณสมบัติของการจัดการเซสชันของโพรโทคอล SIP

แต่เนื่องจากการจัดการเซสชันดังกล่าวควรทำงานได้อย่างมีเสถียรภาพ การออกแบบโมเดลของงานวิจัยนี้จึงได้นำหลักการออกแบบระบบที่มีเสถียรภาพโดยการใช้ฮาร์ทบีทอัลกอริธึม [32] มาติดตามและตรวจจับการทำงานของ UAS และใช้เทคนิคการทำมัลติคาสท์ [27, 28] ข้อมูลเพื่อทำการสำรองข้อมูลที่สำคัญไว้ กรณีที่ UAS ตัวใดล้มไปจะใช้ริงอัลกอริธึม [26] ในการเลือก UAS หลักมาทำงานแทน

5.2 ข้อเสนอแนะ

เนื่องจากการจัดการเซสชันโดยใช้โพรโทคอล SIP นั้น มีรูปแบบของ Session ID ที่เกิดขึ้นเป็นรูปแบบที่ไม่ยุ่งยาก หากจะนำไปใช้ในด้านความปลอดภัยให้เกิดประโยชน์จริง ควรมีการเพิ่มการสร้างข้อมูล Session ID ที่ไม่สามารถคาดเดาได้ นอกจากนี้ควรมีการทำเพิ่มความปลอดภัยในด้านของการทำ Authentication และ Authorization ด้วย

เอกสารอ้างอิง

1. Booth D., Haas H., McCabe F., Newcomer E., Champion M., Ferris C. and Orchard D. "Web Services Architecture". W3C Working Group .11 February 2004.
2. Jeckle M.C. Adhere to Session-Oriented Communication Principles. [Online]. Available from: http://www.jeckle.de/files/ssgrr2002.pdf[2003,Oct 5].
3. Ray J. Randy, Kulchenko. P Programming Web Services with Perl O'Reilly & Association Inc., c2001.
4. UDDI, UDDI technical white paper. [Online]. Available from: http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf[2004,Oct 22].
5. Newmarch J. Web services. [Online]. Available from: http://jan.netcomp.monash.edu.au/webservices/tutorial.html[2004,Oct 22].
6. Fielding R., Gettys J., Mogul J., Frystyk H., Masinter L., Leach P. and Berners-Lee T. "Hypertext Transfer Protocol -- HTTP/1.1" RFC2068 (Jan 1999).
7. Cai H. L., W. Yang B. and Tang L. H. "Session Initiation Protocol and Web Services for Next Generation Multimedia Applications, Multimedia Software Engineering," Proceedings of the IEEE Fourth International. (2002) : 70- 80.
8. Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J.,Sparks R., Handley M. and Schooler. E. "SIP: Session Initiation Protocol." RFC 3261. 2002.
9. Boutell T. CGI Programming in C& Perl. Kim Fryer, c1996.
10. Camarillo G. SIP Demystified. McGraw-Hill United States of America, c2002.
11. Peng W. H. and Clisna J. "HTTP cookies- a Promising Technology." Online Information Review. Vol.24, Iss.2, 150-153, 2000.
12. Gudgin M., Hadley M., Mendelsohn N., Moreau J.-J. and Nielsen H.F. SOAP Version 1.2 Part 2: Adjuncts. [Online] Available form: http://www.w3.org/TR/2003/REC-soap12-part2-20030624[2004,Sept 30].
13. Tsenov M. "Application of SOAP Protocol in E-commerce Solution." First International IEEE Symposium. (20 Oct., 2004) Vol.3 : 59-62.

14. Berghel H. "Hijacking the Web-Cookies revisited: Continue the dialog on personal security and underling privacy issue." Communication of the ACM vol. 45 (2002) : 23-27.

15. Newmarch J. A Critique of Web Services. [Online] Available from: http://jan.netcomp.monash.edu.au [2004,Oct 26].

16. Lennox J., Rosenberg J. and Schulzrinne H. "Common Gateway Interface for SIP." Internet Draft, Internet Engineering Task Force RFC3050, (May 1999).

17. Gudgin M., Hadley M., Mendelsohn N., Moreau J.-J. and Nielsen H.F. SOAP Version 1.2. Part 2: Adjuncts. [Online] Available from: http://www.w3.org/TR/2003/REC-soap12-part2-20030624 [2004,Sept 30].

18. Ts'o T. and Chiu A. ONC Remote Procedure Call. [Online] Available from: http://www.ietf.org/html.charters/oncrpc-charter.html[2004,Oct 20].

19. Chan H., Lee R., Dillon T. and Chang E. E-Commerce Fundamentals and Applications, John Wiley & Sons john Wiley & Sons Ltd Baffins Lane Chichester West Succex PO19 IUD England, c2001.

20. Looker N. and Xu J. "Assessing the Dependability of SOAP RPC-based Web Services by Fault Injection." Object-Oriented Real-Time Dependable Systems, 2003. Proceedings of 9th IEEE International Workshop. (1-3 Oct. 2003) : 163- 170

21. Christensen E., Curbera F., Meredith G. and Weerawarana S. Web Services Description Language (WSDL) 1.1. [Online] Available from: http://www.w3.org/TR/wsdl[2003,Oct 22].

22. Handley M., and Jacobson V. RFC 2327 - SDP: Session Description Protocol. [Online] Available from: http://www.faqs.org/rfcs/rfc2327.html[2004,Oct 20].

23. Ismail Ari, Jun Li, Riddhiman Ghosh and Mohamed Dekhil. Providing Session Management as Core Business Service. ACM, (8-12 May 2007) : 1263 – 1264.

24. Dong W. and Newmarch J. Adding Session and Transaction Management to XML Web Services by Using SIP. Monarch University, Submitted February, c2005.

25. Hada S. and Maruyama H. "Session Authentication Protocol for Web Services." IEEE (Jan.2002) 26 : 158-165.

26. Andrew S. Tanenbaum and Marten Van Steen. Distributed System Principles and Paradigms. Prentice Hall Inc. Upper Saddle River, New Jersey 07458 . c2002.

27. Ryan G. Lane Scott Daniels Xin Yuan. An Empirical Study of Reliable Multicast Protocols over Ethernet Connected Networks. IEEE (3-7 Sept. 2001) : 553- 560.

28. Mostafa M., Singhal M. "A Reliable Multicast Session Protocol for Collaborative Continuous Feed Applications." ACM. (1997) : 35 – 39.

29. Astrain J. J., Cordoba A. and Villadangos J. "A B2B Distributed Replication Service." IEEE. (15-17 Feb. 2006) : 141 – 144.

30. Patterson J. F., Hill R. D., Rohall S. L., and Meeks S. W. "An Architecture for Synchronous Multi-user Applications." Computer Supported Cooperative Work, Proceedings of the 1990 ACM Conference on Computer Supported Cooperative Work. (1990) : 317-328.

31. Hughes M., Shoffner M. and Hamner D. Java Network Programming. 2nd, Special Sales Department Publication Co. 32, Lafyence Place Greenwich, CT 6830. c1999.

32. Zonghao Hou, Yongxiang Huang, Shouqi Zheng, Xiaoshe Dong and Bingkang Wang. "Design and Implementation of Heartbeat in Multi-machine Environment" IEEE (27-29 March 2003) : 583- 586.

33. Shimamura, K,Tanaka, K, Takizawa, M, "Group communication protocol for multimedia application." IEEE (16-19 Oct. 2001) : 303 – 308.

34. Mora. J. Francisco and Kantola. R. "An Implementation of the Internet Call Waiting Service using SIP." Helsinki University of Technology. December.1999.

ประวัติผู้วิจัย

ชื่อ : นางสาว รพีพร จินะ

ชื่อวิทยานิพนธ์ : การออกแบบการปฏิสัมพันธ์ระหว่างเว็บเซอร์วิสบนช่องทางการสื่อสารที่มีเสถียรภาพ

สาขาวิชา : วิทยาการคอมพิวเตอร์

ประวัติ

ประวัติการศึกษา
- ปริญญาตรีอุตสาหกรรมศาสตรบัณฑิตสาขาอิเล็กทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
- กำลังศึกษาปริญญาโท สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาวิทยาการคอมพิวเตอร์ และสารสนเทศ บัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ประวัติการทำงาน และผลงานวิชาการ
- พ.ศ. 2537–2542 เจ้าหน้าที่ฝ่ายเทคนิค ศูนย์ปฏิบัติการกลาง คณะสาธารณสุขศาสตร มหาวิทยาลัยมหิดล
- พ.ศ. 2542-2549 เจ้าหน้าที่วิจัย ศูนย์ปฏิบัติการกลาง คณะสาธารณสุขศาสตร มหาวิทยาลัย มหิดล
- พ.ศ. 2550 บทความได้รับการตีพิมพ์ “Reliable Session Management for Web Services by Using SIP” , JCSSE 2007, (http://www.jcsse.org) Khon Kaen University Thailand

สถานที่ติดต่อ
- 17 ถนน จรัญสนิทวงศ์ ซอย จรัญสนิทวงศ์ 62 เขต บางพลัด แขวงบางยี่ขัน กรุงเทพฯ
- Email Address: nyret3200@yahoo.com, nyret3200@hotmail.com