เว็บสนทนาผ่านโปรโตคอลแจปเปอร์

JABBER INSTANT MESSAGING WEB CLIENT

นายธีรพงษ์ บุญเกี๋ยงวงค์ รหัส 47380023

นายนพดล โตบันลือภพ รหัส 47380024

นายพุทธิพล พันธุ์สืบ รหัส 47380035

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2550

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	เว็บสนทนาผ่านโปรโตคอลแจปเปอร์	
ผู้ดำเนินโครงงาน	นายธีรพงษ์ บุญเกี๋ยงวงค์	รหัส 47380023
	นายนพดล โตบันลือภพ	รหัส 47380024
	นายพุทธิพล พันธุ์สืบ	รหัส 47380035
อาจารย์ที่ปรึกษา	อาจารย์เศรษฐา ตั้งค้าวานิช	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์	
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์	
ปีการศึกษา	2550	

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงการฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

เศรษฐา ตั้งค้าวานิช.....ประธานกรรมการ

(อาจารย์เศรษฐา ตั้งค้าวานิช)

.....กรรมการ

(ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สุรเชษฐ์ กานต์ประชา)

หัวข้อโครงงาน เว็บสนทนาผ่านโปรโตคอลแจปเปอร์

ผู้ดำเนินโครงงาน นายธีรพงษ์ บุญเกี๋ยงวงค์ รหัส 47380023

นายนพดล โตบันลือภพ รหัส 47380024

นายพุทธิพล พันธุ์สีบ รหัส 47380035

อาจารย์ที่ปรึกษา อาจารย์เศรษฐา ตั้งค้าวานิช

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2550

บทคัดย่อ

โครงงานเว็บสนทนาผ่านโปรโตคอลแจปเปอร์ เป็นเว็บสนทนาที่สามารถติดต่อได้หลายโปรโตคอล (MSN, GTALK) และสามารถติดต่อสื่อสารผ่านทางเครือข่ายเน็ตเวิร์คเพื่อใช้ภายในองค์กร ซึ่งทำให้เกิดความสะดวกในการติดต่อและลดปริมาณการใช้ Internet ภายในองค์กรลง ทำให้นำไปใช้งานในส่วนอื่นได้ โครงงานนี้พัฒนาด้วย JSP ผลที่ได้รับในโครงงานนี้ คือ เว็บแอปพลิเคชั่น ที่สามารถสนทนาได้หลายโปรโตคอลและสามารถติดต่อสื่อสารผ่านทางเครือข่ายเน็ตเวิร์ค

Project	Jabber Instant Messaging Web Client
Name	Mr.Teerapong Bonkengwong ID. 47380023
	Mr.Noppadon Tobunluepob ID. 47380024
	Mr.Puttipon Punsuib ID. 47380035
Project Advisor	Mr.Settha Tungkawanit
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic Year	2007

ABSTRACT

This project is chat webpage which is use to communicate in many protocols (MSN, GTALK) and communicate through network in order to use in organization. Besides, it is convenient to communicate and decrease bandwidth in organization. This project is developed by JSP. The results of this project are web applications which can chat with many protocols and communicate through network.

กิตติกรรมประกาศ

ในการจัดทำโครงงานในครั้งนี้ได้สำเร็จลุล่วงไปได้ด้วยดี เพราะได้รับการสนับสนุนและความช่วยเหลือจากหลายฝ่าย อันได้แก่ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ที่ได้มอบทุนในการศึกษา ค้นคว้า และจัดทำโครงงานนี้

คณะผู้จัดทำขอขอบพระคุณอาจารย์ที่ปรึกษา อาจารย์เศรษฐา ตั้งค้าวานิช ที่ได้กรุณาเสียสละเวลาและคำปรึกษาที่มีประโยชน์ ทำให้คณะผู้จัดทำโครงงานได้รับประโยชน์และความรู้ใหม่ๆ

และขอขอบพระคุณอาจารย์ท่านอื่นๆ รวมไปถึงเจ้าหน้าที่ประจำคณะวิศวกรรมศาสตร์ ที่ไม่ได้กล่าวนามทุกท่านที่คอยให้คำแนะนำและความช่วยเหลือในการจัดทำโครงงานนี้ คณะผู้จัดทำขอกราบขอบพระคุณไว้ ณ ที่นี้

นายธีรพงษ์ บุญเกี้ยงวงค์
นายนพดล โตบันลือภพ
นายพุทธิพล พันธุ์สืบ

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ในปัจจุบันนี้การติดต่อสื่อสารถือเป็นหัวใจสำคัญอย่างหนึ่งในโลกที่ต้องมีการรับส่งข้อมูลข่าวสาร การทำให้ระบบการติดต่อสื่อสารมีประสิทธิภาพและมีความรวดเร็ว การใช้งานระบบรับส่งข้อความแบบทันที (Instant Messaging) กลายเป็นสิ่งที่จำเป็นสำหรับคอมพิวเตอร์ที่เชื่อมต่อเครือข่ายเน็ตเวิร์คหรือ internet จะเห็นได้จากการใช้งานโปรแกรม IM (Instant Massager) ของค่ายต่างๆ เป็นที่นิยมอย่างมากเช่น MSN, YAHOO, ICQ, QQ เป็นต้น แต่ข้อจำกัดของโปรแกรม IM ที่มีใช้ในปัจจุบันโดยทั่วไป คือ ต้องติดตั้งตัวโปรแกรมลงบนเครื่องของผู้ใช้ ซึ่งโดยทั่วไปสามารถติดต่อได้เพียงโปรโตคอลเดียว ในบางครั้งเมื่อจำเป็นต้องทำการติดต่อผ่านทางโปรแกรม IM แต่ก็ไม่มีโปรแกรม IM ติดตั้งอยู่บนเครื่องหรือมีโปรแกรม IM ที่ติดตั้งแล้วแต่ก็เป็นโปรแกรม IM คนละโปรโตคอลกับที่เราต้องการจะใช้ ทำให้ต้องเสียเวลาในการหาโปรแกรมมาติดตั้งใหม่อีก

ด้วยเหตุนี้ จึงได้มีการจัดทำ Web IM ที่สามารถติดต่อได้หลายโปรโตคอล (Multiprotocol instant messaging application) โปรแกรมนี้จะสามารถเลือกได้ว่าจะติดต่อกับโปรโตคอลไหนโดยจะทำออกมาในรูปแบบของ เว็บไซต์ ซึ่งสามารถที่จะใช้งานได้เลยโดยที่ไม่ต้องมีการ ติดตั้งโปรแกรมลงเครื่อง ทำให้ลดปัญหาดังกล่าวลงไปได้ รวมทั้งมีความสามารถในการสื่อสารผ่านทางเครือข่ายเน็ตเวิร์คเพื่อใช้ภายในองค์กรโดยไม่ต้องผ่าน Internet

1.2 วัตถุประสงค์

1.2.1 เพื่อพัฒนาระบบรับส่งข้อความแบบทันทีผ่านเว็บแอพพลิเคชั่น (Web Instant Messaging)

1.2.2 เพื่อพัฒนาระบบการสนทนาที่รองรับหลายโปรโตคอล (Multiprotocol instant messaging application)

1.2.3 เพื่อพัฒนาระบบการสนทนาที่ใช้ภายในองค์กร (Local Area Network Instant Messaging)

1.3 ขอบข่ายของโครงงาน

1.3.1 เขียนเว็บIMที่สามารถติดต่อกับ MSN, YAHOOและภายในองค์กรโดยใช้โปรโตคอล Jabber

1.3.2 การใช้งานโปรโตคอลอื่นจะต้องมี Account ของโปรโตคอลนั้นๆที่มีอยู่แล้ว

1.3.3 การใช้งานภายในองค์กรจะสามารถสร้างใหม่เพื่อใช้งานได้เลย

1.3.4 โปรแกรมสามารถรับ-ส่งข้อความที่เป็นตัวอักษรระหว่างเครื่องได้

1.4 ผลที่คาดว่าจะได้รับ

1.4.1 ได้ระบบรับส่งข้อความที่ลดเวลาในการติดตั้งโปรแกรม IM เพื่อที่จะใช้ในการติดต่อสื่อสาร

1.4.2 ได้ระบบรับส่งข้อความที่สามารถติดต่อได้หลายโปรโตคอล

1.4.3 ได้ระบบรับส่งข้อความไว้ใช้ภายในองค์กรเพื่อเป็นการเพิ่มช่องทางและเพิ่มประสิทธิภาพการสื่อสารภายในองค์กร

1.5 แผนการดำเนินงาน

ตารางที่ 1.1 ขั้นตอนการดำเนินงาน

กิจกรรม	ระยะเวลาดำเนินงาน (เดือน)						
	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
	พ.ศ. 2550			พ.ศ. 2551			
1. ศึกษาและเก็บรวบรวมข้อมูล	⟷	⟷	⟷				
2. ออกแบบโปรแกรม			⟷				
3. เขียนโปรแกรม			⟷	⟷	⟷		
4. ทดสอบโปรแกรม						⟷	
5. ตรวจสอบและปรับปรุงแก้ไขให้สมบูรณ์						⟷	⟷
6. จัดทำรายงานเป็นรูปเล่ม						⟷	⟷

1.6 งบประมาณที่ใช้

1.6.1 ค่าใช้จ่ายในการซื้อหนังสือ	1,000	บาท
1.6.2 ค่าใช้จ่ายในการทำรายงาน	500	บาท
1.6.3 ค่าใช้จ่ายเบ็ดเตล็ด	500	บาท
รวมทั้งสิ้น	2,000	บาท

(สองพันบาทถ้วน)

บทที่ 2
หลักการและทฤษฎี

โครงงาน เว็บสนทนาผ่านโปรโตคอลแจปเปอร์ เป็นโครงงานเกี่ยวกับการติดต่อสื่อสาร ระหว่างClient และ Server ดังนั้นจึงจำเป็นต้องใช้ทฤษฎีต่อไปนี้ในการทำโครงงาน

2.1 INSTANT MESSAGING (ที่มา : http://th.wikipedia.org/wiki/Instant_messaging)

เมสเซนเจอร์ หรือ อินสแตนท์ เมสเซจจิง (instant messaging, IM) คือระบบการส่งข้อความทันที ระหว่างสองคน หรือกลุ่มคนใน เน็ตเวิร์ก เดียวกัน เช่น การส่งข้อความผ่านทางอินเทอร์เน็ต

การทำงานของเมสเซนเจอร์จำเป็นต้องใช้ไคลเอนท์ซอฟต์แวร์ โดยซอฟต์แวร์ทำการเชื่อมต่อระบบที่บริการเมสเซนเจอร์ การส่งข้อความผ่านเมสเซนเจอร์ในยุคแรก ตัวอักษรแต่ละตัวที่ทำการพิมพ์จะปรากฏทางหน้าจอของผู้ที่ส่งข้อความด้วยทันที ในขณะเดียวกัน การลบตัวอักษรแต่ละตัว จะลบข้อความทันที ซึ่งแตกต่างกับระบบเมสเซนเจอร์ในปัจจุบัน โดยข้อมูลที่ปรากฏจะเกิดขึ้นหลังจากที่มีตกลงยอมรับส่งข้อความแล้ว ในปัจจุบันเมสเซนเจอร์ที่ได้รับความนิยมได้แก่ MSN Messenger AOL Instant Messenger Yahoo! Messenger Google Talk .NET Messenger Serviee Jabber และ ICQ

2.2 Jabber (ที่มา : http://www.jabber.org/)

2.2.1 Jabber เป็นเทคโนโลยีซึ่งเกี่ยวข้องกับ IM (Instant Messaging) หรือระบบรับส่งข้อความที่ออกมาอย่างแพร่หลายในปัจจุบันซึ่งมีจุดเด่นคือ สามารถติดต่อกับ IM ตัวอื่นๆ ได้ไม่จำกัดซึ่ง โดยปกติ IM ไม่เหมือนกันจะสามารถติดต่อกันได้ เช่น MSN ไม่สามารถติดต่อกับ Yahoo ได้เป็นต้น

2.2.2 Jabber เป็นซอฟต์แวร์ Open Source ที่สนับสนุนโปรโตคอลการรับส่งข้อความแบบ Streaming XML Protocol, XMPP Protocol ที่ IETF รองรับ นอกจากนี้ยังสามารถนำไปพัฒนาให้สามารถติดต่อกันภายในเครือข่ายของสำนักงานได้ โดยไม่ต้องใช้อินเตอร์เน็ต

2.2.3 Jabber แบ่งออกเป็นสองส่วนคือ Client กับ Server และใช้ Port 5222 ในการติดต่อ เราสามารถที่จะสร้างเครือข่าย IM ของเรา Server ของ Jabber แต่ละตัวสามารถสื่อสารกับ Server ตัวอื่นๆ ที่อยู่กันคนละเครือข่ายได้

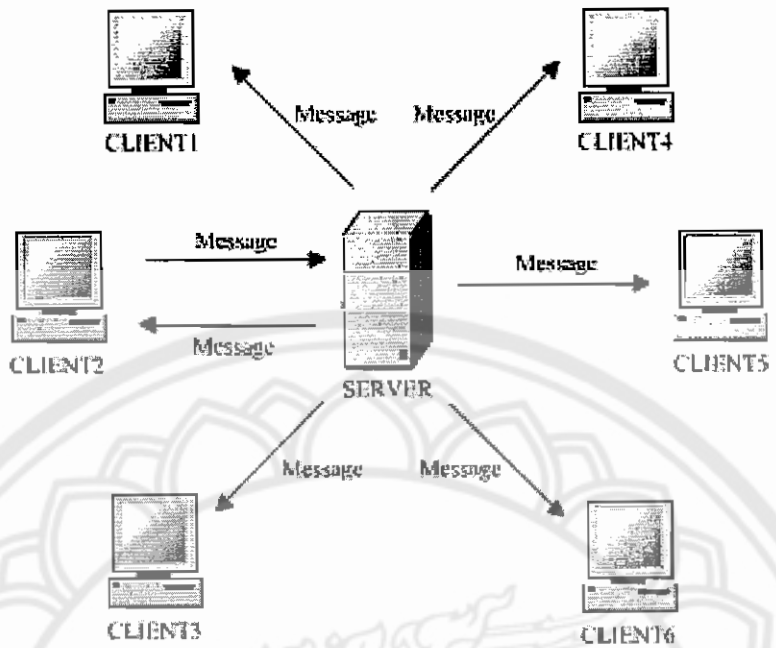
2.3 Client-server (ที่มา : www.bcoms.net/dictionnary)

2.3.1 Client เป็นโปรแกรมหรือผู้ใช้ที่ขอ ในความสัมพันธ์ Client/Server เช่น ผู้ใช้ Web Browser เป็นผู้สร้างการขอ Client สำหรับเพจ จากเครื่องแม่ข่าย Browser โดยตัวเอง เป็น Client ของความสัมพันธ์กับคอมพิวเตอร์ที่นำและส่งกลับคำขอไฟล์ HTML คอมพิวเตอร์ที่รับคำขอและส่งกลับไฟล์ HTML คือ Server

2.3.2 Server ในแบบจำลองระบบโปรแกรม Client/Server Server เป็นโปรแกรมที่รอและตอบสนองให้กับโปรแกรม Client ในเครื่องคอมพิวเตอร์เดียวกันหรือคนละเครื่อง คอมพิวเตอร์ที่ได้รับการประยุกต์จะทำงานในฐานะ Client โดยขอรับการบริการจากโปรแกรมอื่น และ Server เป็นผู้รับคำขอจากโปรแกรมอื่น ถ้าเจาะจงที่เว็บ Web Server เป็นโปรแกรมคอมพิวเตอร์ที่บริการการขอเพจ HTML ของผู้ใช้ Web Browser เป็น Client ที่ใช้ขอไฟล์ HTML จาก Web Server

2.4 Socket (ที่มา : http://www.kku.ac.th/)

XML Socket Class เป็น Built-In Class ที่ Action Script ได้จัดเตรียมไว้ให้ Client กับ Scrver หรือคอมพิวเตอร์ 2 เครื่องที่อยู่บนเครือข่ายเดียวกันสามารถติดต่อสื่อสารกันได้ผ่านทางSocket โดยใช้โปรแกรม Flash และข้อมูลที่ใช้ในการติดต่อสื่อสารจะอยู่ในรูปแบบของXML โดยจะนำ XML Socket Class ไป Implement กับ Client Socket ทำให้เครื่องที่เป็น Client สามารถติดต่อกับ Server ได้ โดยการระบุ IP Address หรือ Domain Nameหลักการทำงานของ XML Socket Class การติดต่อสื่อสารกันระหว่าง Client และ Server ด้วยวิธี XML Socket นั้น ฝั่ง Server จะตรวจสอบข้อมูลขาเข้าของตัวเองอยู่ตลอดเวลา เพื่อรอฝั่ง Client ส่งข้อมูลที่อยู่ในรูปแบบของ XML มาที่ฝั่ง Servcr ผ่านทาง Socket และเมื่อ Server ต้องการส่งข้อมูลให้กับ Client ก็จะใช้วิธีการกระจาย (Broadcast) ข้อมูลไปยังทุกๆ Clicnt ที่อยู่ในระบบทำให้ทุกๆ Client ได้รับข้อมูลที่เหมือนกัน โดย Client จะรอรับข้อมูลจาก Server อยู่ตลอดเวลา เมื่อฝั่ง Client ได้รับข้อมูลแล้ว ก็จะนำข้อมูลดังกล่าวมาตรวจสอบและนำไปใช้งานต่อไป ให้พิจารณาภาพการส่งข้อมูลระหว่าง Client และ Server ดังนี้



รูปที่ 2.1 แสดงภาพการรับส่งข้อมูลของ XML Socket (ที่มา : http://www.kku.ac.th/)

รูปที่ 2.1 แสดงการรับส่งข้อมูลระหว่าง Server กับแต่ละ Client จะเห็นว่าเมื่อ "CLIENT2" ส่ง Message ซึ่งเป็นข้อมูลให้กับ Server แล้ว เครื่อง Server จะนำข้อมูลดังกล่าวไปประมวลผลและกระจาย (Broadcast) ข้อมูลที่เป็นผลลัพธ์ไปยังทุก Client

2.5 Openfire Server (ที่มา : http://www.narisa.com)

2.5.1 Openfire เป็น Open source XMPP server มีชื่อเดิมว่า wildfire ถูกพัฒนาขึ้นโดยภาษา java ใช้ jabber เป็น protocol สำหรับทำเป็น IM (Instant Massager) Server

2.5.2 Openfire ทำหน้าที่เป็น Server สำหรับให้บริหารจัดการ IM การจัดการ User และ การจัดการ Gateway โดยจะสามารถทำงานได้โดยไม่ต้องผ่าน Internet ทำให้สามารถนำมาทำเป็น Local Area Network Instant Messaging Server ที่ใช้ภายในองค์กรได้ สำหรับการติดตั้ง Openfire server จะสามารถติดตั้งบน windows หรือ Linux ก็ได้ โดยภายในเครื่องจะต้องมี MySQL version 4 ขึ้นไป เป็นโปรแกรม Database สำหรับเก็บข้อมูลของระบบ และโปรแกรม Java Version Jdk 1.5 ขึ้นไปเนื่องจาก Openfire ถูกพัฒนามาจากภาษา java การจะทำให้ Openfire สามารถติดต่อกับโปรโตคอลอื่นๆได้เช่น AIM, Yahoo, MSN, ICQ, หรือ IRC จะต้องมี Plug-in สำหรับ Open fire เรียกว่า Gateway Plug-in เพื่อใช้ในการติดต่อhttp://www.jivesoftware.com/products/openfire /features/gateways.jsp

2.6 IM Gateway

IM Gateway เป็นตัวที่อนุญาตให้ User สามารถติดต่อสื่อสารกับ User คนอื่นๆที่ใช้โปรโตคอลต่างกัน เช่น AIM, ICQ, MSN, Yahoo เป็นต้น

2.7 Smack

Smack เป็นซอฟต์แวร์ open source ที่เป็น elient library ที่ถูกพัฒนาโดยภาษา Java โดยเฉพาะสำหรับ instant messaging รันบนโปรโตคอล XMPP ซึ่งเป็นโปรโตคอลเดียวกับ Jabber สามารถนำมาใช้สร้าง XMPP elient instant messaging ที่สามารถส่งการแจ้งเตือนข้อความได้

2.8 JSP (ที่มา : http://kateep.com)

2.8.1 JSP ย่อมาจาก Java Server Pages เทคโนโลยีที่คิดค้นโดยบริษัท Sun Microsystems (ผู้ผลิตคอมพิวเตอร์Sun และผู้พัฒนาจาวา)โดยพัฒนาบนพื้นฐาน ของภาษาจาวาเพื่อเพิ่มประสิทธิภาพให้หน้าเว็บเพจมีความยืดหยุ่นสูงขึ้น โครงสร้างของ JSP นั้นเป็นลักษณะของแท็ก (tag) ชนิดพิเศษที่แทรกเข้าไปใน เอกสาร HTML และเปลี่ยนนามสกุลของเอกสารเป็น. JSP แทนที่จะเป็น .HTM หรือ .HTML JSP มีการทำงานอยู่บนฝั่ง Server หรือ อาจเรียกได้ว่าเป็นการทำงานแบบ Server side โดยแท็ก เหล่านี้เว็บบราวเซอร์จะไม่สามารถตีความหมายได้ จะต้องนำไปประมวลผลก่อนที่เว็บเซิร์ฟเวอร์เท่านั้น (หรือที่เราเรียกว่าการทำงานแบบ Server Side) แล้วนำผลลัพธ์ทั้งหมดส่งกลับมายังเว็บบราวเซอร์ในลักษณะของเอกสาร HTML ซึ่งเว็บบราวเซอร์สามารถตีความหมายและนำมาแสดงผลได้ ขั้นตอนการทำงานจะเริ่มตั้งแต่การร้องขอ หรือ เกิด Request จาก Browser หรือ Client ที่มีเอกสารนามสกุลเป็น JSP ไปยังเว็บเซิร์ฟเวอร์ผ่านทางโปรโตคอล HTTP เว็บเซิร์ฟเวอร์ก็จะนำเอกสาร JSP ที่ได้รับมานั้นส่งต่อไปให้ JSP Engine (JSP Engine คือ แอพพลิเคชั่นที่ถูกโหลดสู่หน่วยความจำและทำงานอยู่บนเว็บเซิร์ฟเวอร์ หน้าที่หลักคือแปลความหมายและประมวลผลเอกสาร JSP) จากนั้น JSP Engine ก็จะประมวลผล และส่งผลลัพธ์กลับมายังเว็บเซิร์ฟเวอร์ แล้วเว็บเซิร์ฟเวอร์ก็จะส่งผลลัพธ์กลับมายังบราวเซอร์ (HTTP Reponses) อีกที ในลักษณะของ เอกสาร HTML บราวเซอร์ก็จะสามารถแสดงผลได้

2.8.2 การใช้ JSP ร่วมกับ Technology อื่น นอกจาก JSP จะถูกนำมาประมวลผลในรูปแบบข่างต้นแล้ว JSP ยังอาจนำไปใช้ร่วมกับ Component หรือ เทคโนโลยีอื่นๆ ได้ เช่น นำไปใช้ร่วมกับ Servlet, Class Bean หรือ EJB เป็นต้น ซึ่งจะทำให้การทำงานของระบบมีประสิทธิภาพสูงขึ้น ในขณะเดียวกันก็

มีความซับซ้อนมากตามไปด้วย เช่นกันและไฟล์ที่เขียนด้วยเทคโนโลยีนี้ จะต้องกำหนดให้ไฟล์มีนามสกุลเป็น *.Jsp เสมอ เพื่อบอกให้ Server ทราบว่าจะต้องทำการประมวลผลกับไฟล์ที่ถูกร้องขอย่างไรการทำงานเมื่อทำการประมวลผล Jsp ด้วย Browser ๆ จะยังไม่สามารถทำการแสดงผลได้ในทันที แต่ Jsp จะถูกทำการประมวลผลเป็น Servlet ก่อน จากนั้น Server จะทำการส่งผลลัพธ์ออกมาให้ Browser เพื่อทำการแสดงผลอีกทีหนึ่ง (จะเห็นว่าการทำงานจะแตกต่างกับ HTML คือ JSP จะไม่สามารถทำการแสดงผลได้ทันที แต่จะต้องมีการประมวลผลบน Server ก่อน)

2.8.3 ข้อดีของ JSP เมื่อเปรียบกับภาษาอื่นๆ ที่ใช้ในการสร้าง Web ด้วยสาเหตุที่ JSP สามารถเขียน Tag Html และ Java แทรกอยู่ปนกันได้ และไม่ต้องทำการ Compile เป็น *.Class ก่อน จึงทำให้ Jsp สามารถใช้งานได้สะดวก รวดเร็วมาก เพราะผู้พัฒนาโปรแกรมสามารถนำเอาไฟล์ Html มาทำการตกแต่ง ให้สวยงามก่อนแล้วจึงแทรก Tag Java เข้าไปทีหลังได้ นอกจากนั้นการทดสอบโปรแกรมก็ไม่ต้องทำการ Compile เป็น *.Class ซึ่งสามารถลดเวลาการทำงานได้เป็นอันมาก และเนื่องจากการที่ Jsp มีพื้นฐานการทำงานมาจาก java จึงทำให้ Jsp มีคุณสมบัติเด่นๆ ของ java ติดมาด้วย เช่น Write once run anywhere คุณสมบัติการใช้งานร่วมกับ Object ต่างๆ ของ java ได้เป็นอย่างดี คุณสมบัติการทำงานแบบ Multithrcad , component reuse able , มีการเปลี่ยนแปลงข้อมูลได้ตลอดเวลาตามserver ไม่แสดงข้อมูลเก่าซ้ำๆเหมือนHTML จากคุณสมบัติเด่นๆ ของ java เหล่านี้เองทำให้ Jsp มีความโดดเด่นมากกว่าภาษาโปรแกรมอื่นๆ

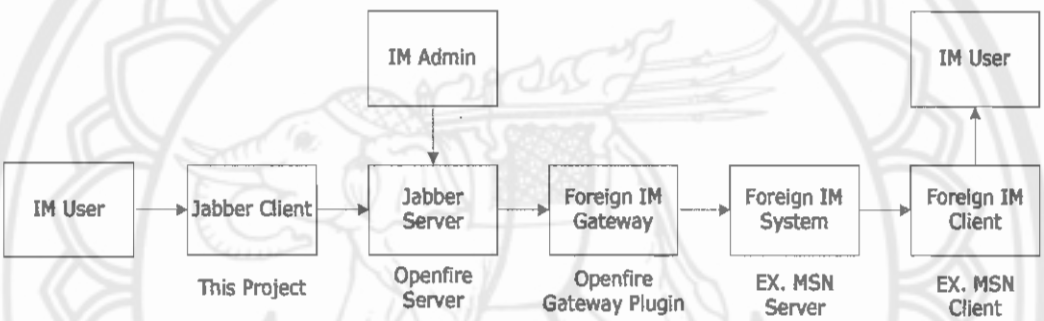
จากการที่ได้ศึกษาทฤษฎี ช่วยให้เข้าใจหลักการและมีส่วนช่วยในการพัฒนาโปรแกรมดังนี้ Instant Messaging ทำให้ทราบถึงระบบการส่งข้อความทันที ระหว่างจุดสองจุดบนเครือข่าย, Jabber ทำให้ทราบถึง หลักการของมัลติโปรโตคอลที่นำมาใช้ในการติดต่อ , Client-Server ทำให้ทราบการทำงานในรูปแบบของClientและServer , Soeket ทำให้ทราบถึงวิธีการติดต่อสื่อสารกันระหว่างจุดสองจุดบนเครือข่าย, Openfire ทำให้ทราบถึงหลักการทำงานของ Server ที่นำมาใช้, IM Gateway ทำให้ทราบถึงวิธีการเชื่อมต่อระหว่างจุดสองจุดผ่านโปรแกรมประยุกต์, Smack ทำให้ทราบถึงคำสั่งของAPI ที่นำมาใช้ในการพัฒนาโปรแกรม , JSP ทำให้ทราบถึงเครื่องมือที่ใช้ในการพัฒนาเว็บแอพพลิเคชั่นของภาษาจาวา ซึ่งจากการศึกษาข้อมูลเหล่านี้ทำให้เป็นประโยชน์ในการออกแบบและพัฒนาโปรแกรมต่อไป

บทที่ 3
การดำเนินการ

ในการสร้างเว็บสนทนาผ่านโปรโตคอลแจปเปอร์จะแบ่งขั้นตอนการดำเนินการงานออกเป็นส่วนๆดังนี้

3.1 การออกแบบโปรแกรม

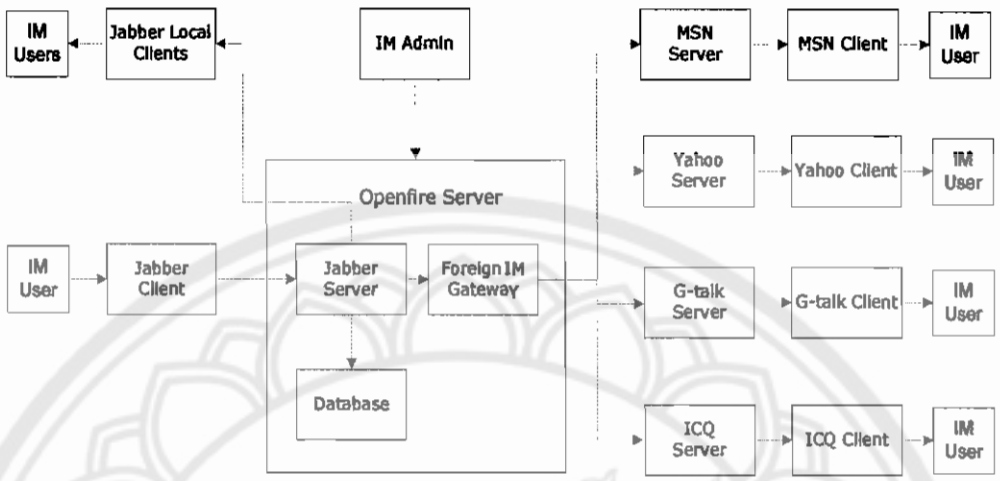
3.1.1 Block Diagram ของเว็บสนทนาผ่านโปรโตคอลแจปเปอร์



รูปที่ 3.1 ระบบโดยรวม

จากรูปที่ 3.1 แสดงการทำงานของระบบ โดยเริ่มจากการที่ Client1 ส่งข้อความไปหา Client2 ผ่านทาง Jabber Client โดยข้อความที่ถูกส่งมานั้นจะส่งเข้ามาที่ server โดย Server จะมีหน้าที่ในการสร้าง Connection ในการส่งข้อความไปหา Client2 ภายใน Server นี้จะมี IM Gateway เพื่อทำการเชื่อมต่อกับ Client2

3.1.2 Block Diagram การทำงานของระบบแบบละเอียด



รูปที่ 3.2 ระบบแบบละเอียด

จากรูปที่ 3.2 เป็นการแสดงการเมื่อผู้ใช้ Jabber Client ต้องการจะติดต่อกับผู้ใช้ Msn, Yahoo, Gtalk หรือ ICQ โดยการติดต่อผ่านOpenfire Server ซึ่งภายใน Openfire server นี้จะมี IM Gateway ที่จะเป็นตัวทำการสร้างการเชื่อมต่อกับโปรโตคอลอื่นๆ โดยภายในIM Gateway นี้จะมีชื่อHost และ port ของ Protocol ที่ต้องการติดต่อจึงทำให้สามารถติดต่อกับโปรโตคอลอื่นๆได้

3.2 ออกแบบหน้าต่างการใช้งานของโปรแกรม

รูปที่ 3.3 หน้าต่างของเว็บสนทนาผ่านโปรโตคอลแจปเปอร์

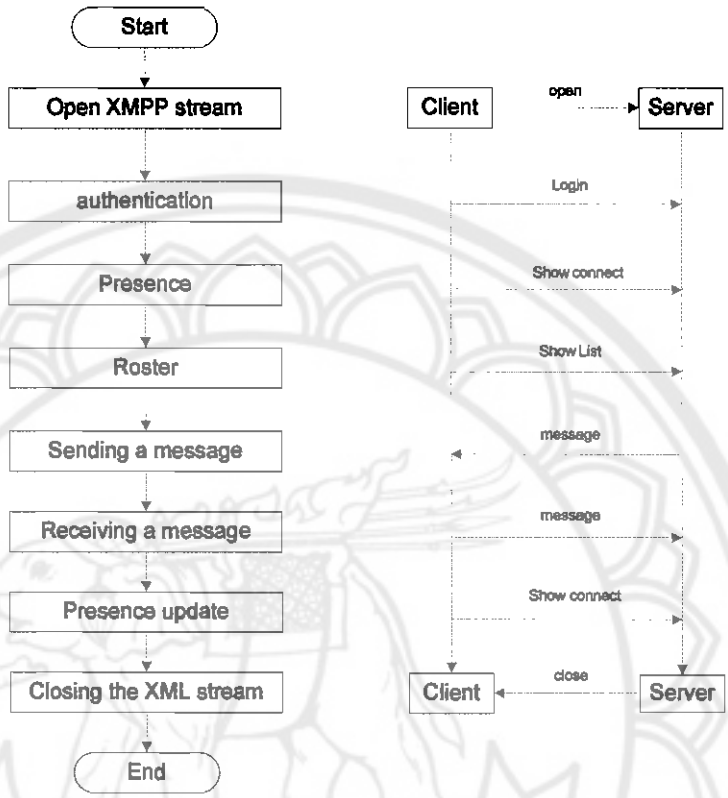
จากรูปที่ 3.3 หน้าต่างของเว็บสนทนานี้จะมี 3 ส่วนด้วยกัน

3.2.1 ส่วนของฟังก์ชั่นต่างๆ Add contact, Add group, Preference, Logout

3.2.2 ส่วนของหน้าต่างในการสนทนา

3.2.3 ส่วนของรายชื่อของผู้ติดต่อ

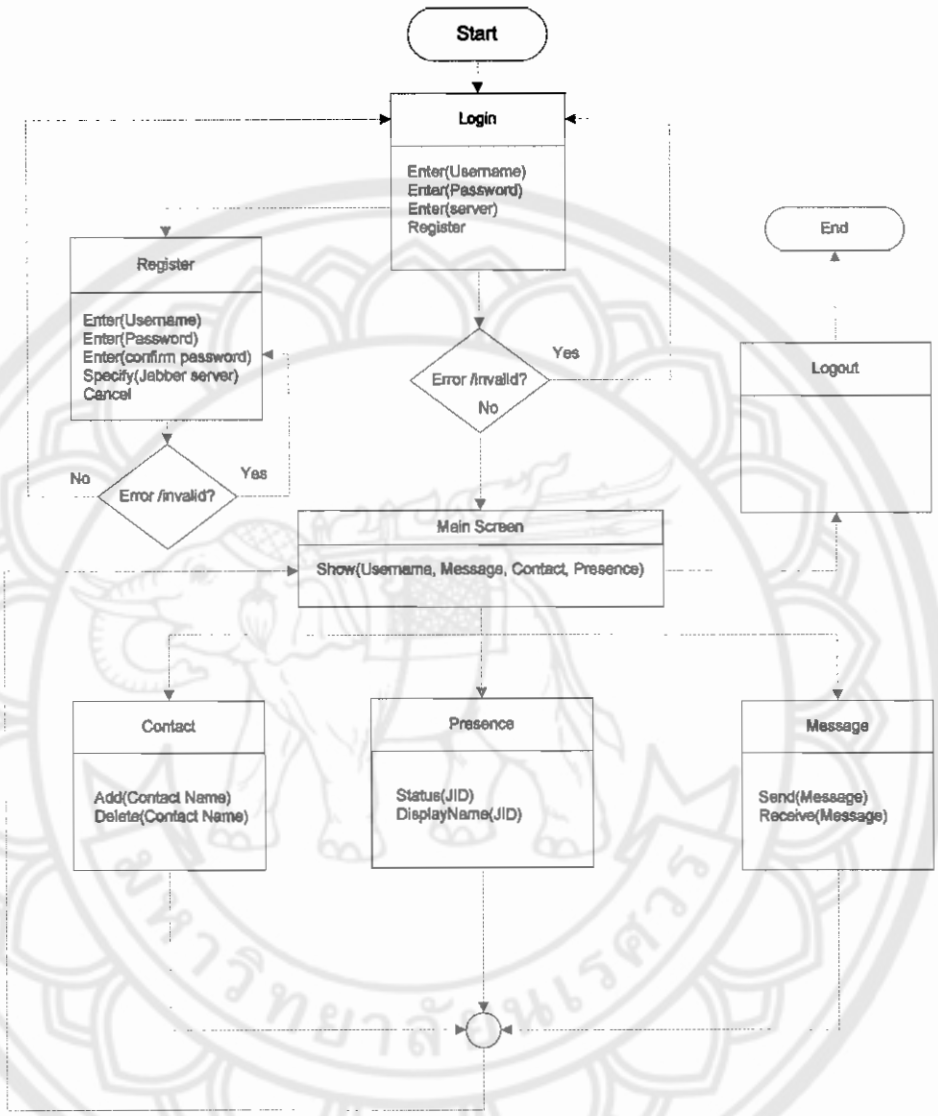
3.3 ขั้นตอนในการทำงานของโปรแกรม



รูปที่ 3. 4 ขั้นตอนการทำงานของโปรแกรม

1. เปิดใช้ Open XMPP Stream
2. ตรวจสอบผู้ที่มาใช้บริการ
3. แสดงสถานะของผู้ใช้ เช่น Online, Offline
4. แสดงรายชื่อผู้ติดต่อของผู้ใช้
5. ส่งข้อความ
6. รับข้อความ
7. Update สถานะของผู้ใช้
8. ปิดการทำงานของ XML Stream

3.4 แผนภาพขั้นตอนการทำงานของโปรแกรม



รูปที่ 3.5แผนภาพการทำงานของโปรแกรม

รูปที่ 3.5 เริ่มจากผู้ใช้ล็อกอิน ถ้าหากกรอกชื่อ, พาสเวิร์ดผิด หรือไม่ได้กรอก ระบบจะเปิดหน้าลงทะเบียนขึ้นมา เมื่อผู้ใช้ลงทะเบียนเสร็จก็จะย้อนกลับไปหน้าล็อกอินอีกครั้ง แต่ถ้าหากผู้ใช้กรอกชื่อและพาสเวิร์ดถูก จะเข้ามาที่หน้าต่างสนทนา ผู้ใช้สามารถใช้หน้าต่างนี้พูดคุยกับผู้ใช้อื่นๆได้ และเมื่อคลิ๊กปุ่มLog out จะเป็นการออกจากระบบ

3.5 เครื่องมือที่ใช้ในการพัฒนาโปรแกรม

1. คอมพิวเตอร์
2. My Eclipse 5.1.1 GA
3. Macromedia Dreamweaver 8
4. ApacheTomcat5.5
5. JRE 1.5
6. JDK 1.5
7. Openfire 3.5
8. Smack 3_0_4
9. IM Gateway

จากการออกแบบการทำงานของโปรแกรม ทำให้สามารถพัฒนาโปรแกรมตรงตามวัตถุประสงค์ง่ายต่อการพัฒนา และสามารถหาเครื่องมือที่จะนำมาพัฒนาโปรแกรมได้อย่างเหมาะสมตามความต้องการในการพัฒนาโปรแกรม ซึ่งการออกแบบเว็บสนทนาผ่านโปรโตคอลแจปเปอร์จะต้องสร้างในรูปแบบของเว็บแอพพลิเคชั่น และมี API ที่รองรับกับภาษาจาวาจึงเลือกเครื่องมือที่ใช้ในการพัฒนาเป็นเครื่องมือพัฒนาเว็บแอพพลิเคชั่นของภาษาจาวา คือ Java Servlet และ JSP

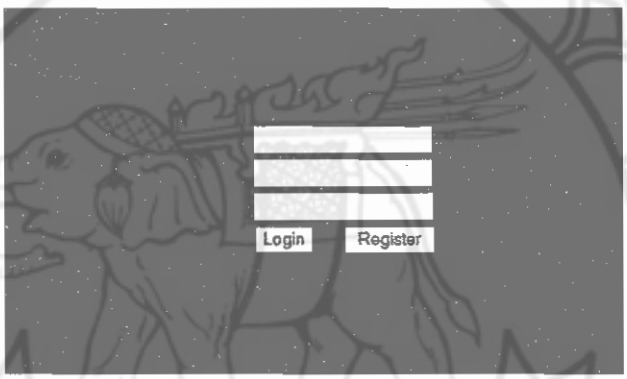
บทที่ 4

การทดลองและผลการทดลอง

จากการทำการทดลองใช้งานเว็บสนทนาผ่านโปรโตคอลแจปเปอร์ โดยทำการลงทะเบียนใหม่และล็อกอินเข้าไปใช้งาน เพื่อทดสอบฟังก์ชั่นการใช้งานและวิธีการใช้งานเว็บสนทนาผ่านโปรโตคอลแจปเปอร์ โดยจะยกตัวอย่างการทดสอบการติดต่อกับโปรโตคอลของMSN

4.1 การใช้งาน

4.1.1 หน้า Login

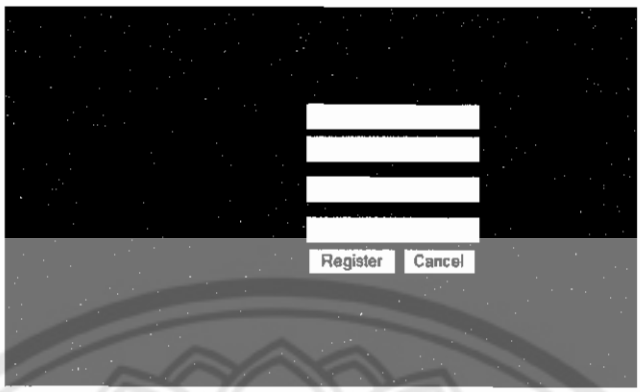


รูปที่ 4.1 หน้า Login

หน้า Login ประกอบด้วย

- Username ใส่ JID ของผู้ใช้
- Password ใส่รหัสผ่านของผู้ใช้
- Server ใส่ชื่อ Server ที่ต้องการติดต่อ
- Login คลิ๊กที่ปุ่มนี้เพื่อทำการ ยืนยันข้อมูลที่ใส่ลงไปเพื่อ เข้าสู่ระบบ
- Register คลิ๊กปุ่มนี้ถ้าต้องการไปที่หน้าลงทะเบียนเข้าใช้

4.1.2 หน้า Register

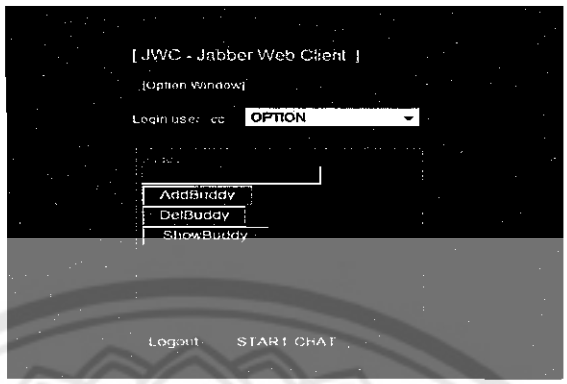


รูปที่ 4.2 หน้าRegister

หน้าRegister ประกอบด้วย

- Username ใส่ชื่อ JID ที่ต้องการ
- Password ใส่รหัสผ่านตามที่ต้องการ
- Confirm Password ยืนยันรหัสผ่าน
- Server ใส่ชื่อServer
- Register คลิ๊กเพื่อยืนยันการลงทะเบียน
- Cancel คลิ๊กเพื่อยกเลิกการลงทะเบียน

4.1.3 หน้าหลัก

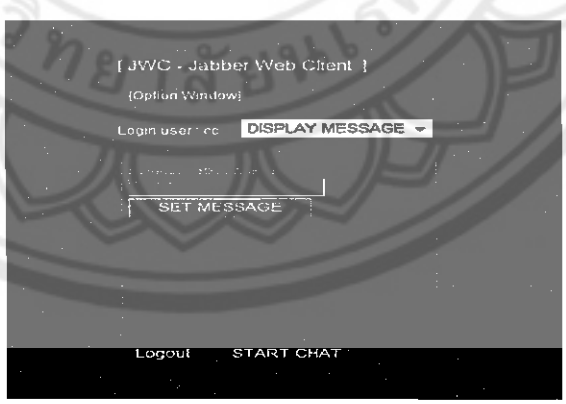


รูปที่ 4.3 หน้าหลัก

หน้าหลักประกอบด้วย

- Option list เลือกโหมดการใช้งาน
- Enter buddy to Chat with ใส่ชื่อที่ต้องการจะสนทนาด้วย
- Start Chat เริ่มการสนทนา
- Login user แสดงชื่อที่ผู้ใช้ Login
- Logout ออกจากระบบ

4.1.4 Display Message



รูปที่ 4.4 แสดงOption ส่วนที่เป็น Display message

- **Display message คือ ข้อความที่ผู้ใช้ต้องการแสดงต่อท้ายชื่อ**

4.1.5 หน้า Start Chat

รูปที่ 4.5 แสดงการ Start Chat

รูปที่ 4.5 หน้า Start Chat จะออกมาเมื่อกดที่ปุ่ม STRAT CHAT ตรงกรอบสี่เหลี่ยมสีแดง

หน้าStart Chat ประกอบไปด้วย

- Chat with ใส่ชื่อของผู้ที่ต้องการจะติดต่อ
- Chat window แสดงข้อความของผู้ใช้ที่ได้สนทนาออกไป
- Message ใช้พิมพ์ข้อความที่จะสนทนา
- ปุ่มSend ส่งข้อความ

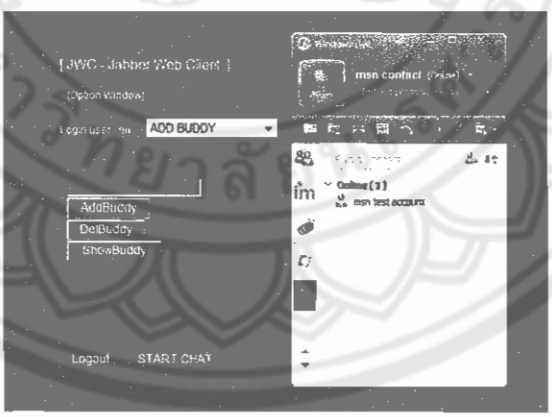
4.2 การทดสอบการติดต่อกับ MSN

4.2.1ทดสอบการ Login เข้าสู่ระบบ

รูปที่ 4.6 การ Login เข้าสู่ระบบ

- ใส่ Username, Password และ Server ของ JID ที่ได้ทำการตั้งค่าให้ติดต่อกับ ID ของ MSN ลงไป ดังรูปที่ 4.6

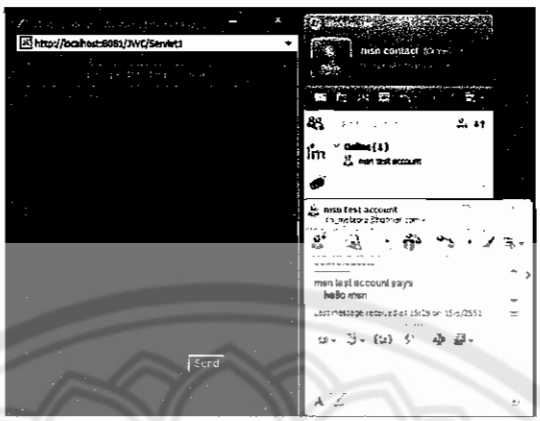
เมื่อ Login สำเร็จ



รูปที่ 4.7 แสดงหน้าหลัก

- เมื่อทำการ Loginสำเร็จ จะเข้ามาสู่หน้าหลักและ จะเห็น ID MSN ที่ใช้ทดสอบมีสถานะออนไลน์ ดังรูปที่ 4.7

4.2.2 ทดสอบการสนทนากับ MSN



รูปที่ 4.8 การสนทนากับ MSN

- เมื่อส่งข้อความไปหา ID MSN ในหน้าต่างของ MSN ก็จะแสดงข้อความที่ส่งไปดังรูปที่ 4.8

4.3 ขั้นตอนการใช้งาน

4.3.1 ใส่ Username, Passwordและ Server แล้วคลิ๊กLogin

4.3.2 เมื่อเข้ามาหน้าหลักแล้วให้ใส่ชื่อ เพื่อนที่ต้องการสนทนาลงในช่อง "Enter buddy to chat with" แล้วคลิ๊กปุ่ม "Start chat"

4.3.3 จะมีหน้าต่างใหม่ขึ้นมาดังรูปที่ 4.9 หน้านี้จะเป็นหน้าที่ใช้สนทนา ให้ใส่ชื่อของผู้ที่ต้องการจะติดต่อที่ Chat with แล้วเริ่มการสนทนา โดยพิมพ์ข้อความที่ต้องการลงในไป แล้วคลิ๊ก "Send" เพื่อส่งข้อความไปยังผู้ที่กำลังสนทนา

4.3.4 เมื่อต้องเลิกการใช้งานให้ คลิ๊กที่ "Logout "

รูปที่ 4.9แสดงหน้าต่างที่ใช้ในการสนทน

จากการที่ได้ทำการทดลองใช้งานเว็บสนทนาผ่านโปรโตคอลแชปเปอร์แล้ว ผลที่ออกมาเป็นดังต่อไปนี้

- สามารถเข้าสู่ระบบสำหรับผู้ใช้ที่สมัครแล้วได้
- สามารถลงทะเบียนสำหรับผู้ใช้ใหม่ได้
- สามารถเพิ่มรายชื่อผู้ติดต่อได้
- สามารถลบรายชื่อผู้ติดต่อได้
- สามารถตั้งค่าการแสดงข้อความส่วนตัวได้
- สามารถแสดงสถานะของผู้ติดต่อได้
- สามารถแสดงผลการสนทนากับคู่สนทนาที่ใช้โปรแกรมMSNและGtalkในการสนทนาได้ โดยข้อความของคู่สนทนาจะแสดงทางConsole

บทที่ 5
บทสรุป

จากการทำโครงงาน เว็บสนทนาผ่านโปรโตคอลแจปเปอร์ คณะผู้จัดทำโครงงานได้บทสรุปดังนี้

5.1 สรุปผลการทำโครงงาน

5.1.1 ได้เว็บสนทนาผ่านโปรโตคอลแจปเปอร์ ที่สามารถสนทนาได้หลายโปรโตคอล ทำให้มีความสะดวกในการสนทนากันมากขึ้น

5.1.2 เว็บสนทนาผ่านโปรโตคอลแจปเปอร์ สามารถนำไปใช้ภายในองค์กรได้ โดยการติดตั้ง เซิร์ฟเวอร์ภายในองค์กรเอง ซึ่งจะช่วยลดช่องทางการติดต่อโดยการใช้ Internet ลดลง เพราะการติดต่อภายในองค์กรจะเป็นแบบ LocalHost

5.1.3 เว็บสนทนาผ่านโปรโตคอลแจปเปอร์ สามารถใช้ได้โดยไม่ต้อง ติดตั้งลงเครื่อง จึงทำให้สามารถใช้ได้ทุกที่ และไม่ต้องเสียเวลาในการติดตั้ง

5.2 ปัญหาและการแก้ปัญหา

เนื่องจากการเขียนเว็บสนทนาผ่านโปรโตคอลแจปเปอร์ ต้องใช้ความรู้เกี่ยวกับ Ajax ซึ่งทางผู้จัดทำโครงงานไม่เคยศึกษามาก่อนหน้านี้จึงทำให้ เกิดปัญหาในการเขียนโปรแกรม การแก้ปัญหาก็คือ ศึกษาและถามผู้มีความรู้ในเรื่องการเขียน Ajax

5.3 ข้อเสนอแนะ

5.3.1 ในการจัดทำเว็บสนทนาผ่านโปรโตคอลแจปเปอร์นั้น ผู้พัฒนาจำเป็นต้องมีความรู้พื้นฐาน และความถนัดในการเขียนภาษาที่จะนำมาใช้เขียนด้วย จึงจะทำเว็บที่มีประสิทธิภาพและสามารถนำไปใช้งานได้จริง

5.3.2 การออกแบบหน้าต่างการใช้งานโปรแกรม ผู้พัฒนาควรคำนึงถึงความง่ายในการใช้งานของผู้ที่นำเว็บนี้ไปใช้จริง ไม่ควรออกแบบยุ่งยากซับซ้อนเกินไป

5.3.3 โครงงานเว็บสนทนาผ่านโปรโตคอลแจปเปอร์นี้ยังไม่สามารถสื่อสารกันโดยใช้ภาษาไทยหากนำไปพัฒนาต่อควรทำให้สามารถสนทนากันโดยใช้ภาษาไทยได้

5.3.4 แนวทางการแก้ไขให้สามารถรับข้อความที่ส่งกลับมาแสดงผลบนหน้าเว็บ อาจจะต้องใช้หลักการของโมเดลที่สองมาช่วยในการแก้ไข

เอกสารอ้างอิง

[1] อรพิน ประวัติบริสุทธิ์,คู่มือเรียน JSP(JavaServer Pages),บริษัท โปรวิชั่นจำกัด กรุงเทพมหานคร:บริษัท โปรวิชั่น จำกัด ,2550

[2] Jabber.org Team. "Open instant messaging and presence."[ออนไลน์]. เข้าถึงได้จาก:http://www.jabber.org/.
สืบค้น 9 มกราคม 2551.

[3] Jive software. "openfire chat server." [ออนไลน์].
เข้าถึงได้จาก:http://www.jivesoftware.com.
สืบค้น 9 มกราคม 2551.

[4] Wikipedia, the free encyclopedia. "Eclipse (software)." [ออนไลน์].เข้าถึงได้จาก:
http://en.wikipedia.org/wiki/Eclipse_(software).
สืบค้น 10 มกราคม 2551.

[5] The Apache Software Foundation. "Apache Tomcat." [ออนไลน์].เข้าถึงได้จาก:
http://tomcat.apache.org.
สืบค้น 10 มกราคม 2551.

[6] วิกิพีเดีย สารานุกรมเสรี. "เมสเซนเจอร์." [ออนไลน์].
เข้าถึงได้จาก: http://th.wikipedia.org/wiki/Instant_messaging.
สืบค้น 8 มกราคม 2551.

[7] วิกิพีเดีย สารานุกรมเสรี. "จาวา." [ออนไลน์].
เข้าถึงได้จาก: http://th.wikipedia.org/wiki/Java.
สืบค้น 8 มกราคม 2551.

ภาคผนวก ก

Code Jabber Instant Messaging Web Client

Model code 1 : Servlet and JSP

Servlet : - ChatClient

- Servlet1

JSP : - Buddybox

- Chat
- Disbox
- Login
- Register
- Wchat

Model code 2: Pure Servlet

Servlet :- Login

- Chat

ไฟล์ ChatClient

```
package org.jivesoftware.smack;

import java.util.*;
import javax.servlet.http.HttpServlet;
import org.jivesoftware.smack.Chat;
import org.jivesoftware.smack.filter.*;
import org.jivesoftware.smack.packet.*;
import org.jivesoftware.smack.ConnectionConfiguration;
import org.jivesoftware.smack.MessageListener;
import org.jivesoftware.smack.Roster;
import org.jivesoftware.smack.RosterEntry;
import org.jivesoftware.smack.XMPPConnection;

public class ChatClient extends HttpServlet implements MessageListener {
XMPPConnection connection;

public boolean login(String Server, String username, String password) {

boolean loginFlag = false;
try {
ConnectionConfiguration config = new ConnectionConfiguration(
Server, 5222);
connection = new XMPPConnection(config);
connection.connect();
connection.login(username, password);

loginFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
loginFlag = false;
}
```

```
return loginFlag;
}

public boolean Register(String r_server, String r_username,
String r_password) {
boolean RegisFlag = false;
try {
ConnectionConfiguration Conn = new ConnectionConfiguration(
r_server, 5222);
connection = new XMPPConnection(Conn);
connection.connect();
AccountManager acc = new AccountManager(connection);
acc.createAccount(r_username, r_password);
RegisFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
RegisFlag = false;
}
return RegisFlag;
}

public boolean sendMessage(String message, String Contract) {
boolean SendFlag = false;
try {
final Chat chat = connection.getChatManager().createChat(Contract,
this);

chat.sendMessage(message);
System.out.println("<" + connection.getUser() + ">" + " says: "
+ message);

// Accept only messages from Contract
```

```
PacketFilter filter = new AndFilter(new PacketTypeFilter(
Message.class), new FromContainsFilter(chat
.getParticipant()));

PacketListener myListener = new PacketListener() {
public void processPacket(Packet packet) {
try {
if (packet instanceof Message) {
Message msg = (Message) packet;
// Process message
if (msg.getType() == Message.Type.chat) {
String remessage = "<" + chat.getParticipant()
+ ">" + " says: " + msg.getBody();
System.out.println(remessage);
}
}
} catch (Exception ex) {
ex.printStackTrace();
}
}
};
// Register the listener.
connection.addPacketListener(myListener, filter);
SendFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
SendFlag = false;
}
return SendFlag;
}

public boolean AddBuddyList(String buddy) {
```

```
boolean AddFlag = false;
try {
Roster roster = connection.getRoster();
roster.createEntry(buddy, buddy, null);
AddFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
AddFlag = false;
}
return AddFlag;
}

public boolean DeleteBuddyList(String buddy) {
boolean DelFlag = false;
try {
Roster roster = connection.getRoster();
roster.removeEntry(roster.getEntry(buddy));
DelFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
DelFlag = false;
}
return DelFlag;
}

public boolean display(String display) {
boolean ChkFlag = false;
try {
Presence presence = new Presence(Presence.Type.available);
presence.setStatus(display);
connection.sendPacket(presence);
ChkFlag = true;
```

```
} catch (Exception ex) {
ex.printStackTrace();
ChkFlag = false;
}
return ChkFlag;
}

public boolean displayBuddyList() {
boolean ShowFlag = false;
try {
Roster roster = connection.getRoster();
Collection<RosterEntry> entries = roster.getEntries();
System.out.println("\n\n" + entries.size() + " buddy(entries):");
int count1 = 1, count2 = 1, num1 = 0, num2 = 0;
String list1 = null, list2 = null;
for (RosterEntry r : entries) {
Presence presence = roster.getPresence(r.getUser());
if (presence.getType() == Presence.Type.available) {
if (count1 == 1) {
list1 = r.getUser() + "\n";
count1 = 2;
num1++;
} else {
list1 += r.getUser() + "\n";
num1++;
}
} else {
if (count2 == 1) {
list2 = r.getUser() + "\n";
count2 = 2;
num2++;
} else {
```

```
list2 += r.getUser() + "\n";
num2++;
}

}
}
System.out.print("[ ONLINE " + num1 + " USER ]\n");
System.out.print(list1);
System.out.print("\n\n[ OFFLINE " + num2 + " USER ]\n");
System.out.print(list2);
ShowFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
ShowFlag = false;
}
return ShowFlag;
}

public boolean logout() {
boolean logoutFlag = false;
try {
connection.disconnect();
logoutFlag = true;
} catch (Exception ex) {
ex.printStackTrace();
logoutFlag = false;
}
return logoutFlag;
}
public void processMessage(Chat chat, Message msg) {
}
}
```

ไฟล์ Servlet1

```
package org.servlet;

import java.io.IOException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.jivesoftware.smack.ChatClient;

public class Servlet1 extends HttpServlet {

private ChatClient tmp = new ChatClient();

/**
* Constructor of the object.
*/
public Servlet1() {
super();
}

/**
* Destruction of the servlet. <br>
*/
public void destroy() {
super.destroy(); // Just puts "destroy" string in log
// Put your code here
}

public void doDelete(final HttpServletRequest request,
final HttpServletResponse response) throws ServletException,
```

```
IOException {

// Put your code here
}

public void doGet(final HttpServletRequest request,
final HttpServletResponse response) throws ServletException,
IOException {
doPost(request, response);
}

public void doPost(final HttpServletRequest request,
final HttpServletResponse response) throws ServletException,
IOException {

String action = request.getParameter("action") != null ? request
.getParameter("action") : "";
String path = "";

// login variable
String server = request.getParameter("server");
String username = request.getParameter("username");
String password = request.getParameter("password");
// register variable
String r_server = request.getParameter("r_server");
String r_username = request.getParameter("r_username");
String r_password = request.getParameter("r_password");
// chat variable
String message = request.getParameter("message");
String contract = request.getParameter("contract");
String buddy = request.getParameter("buddy");
String display = request.getParameter("display");
```

```
if (action.equals("login")) {
boolean loginFlag = tmp.login(server, username, password);

// Java to Java
request.getSession().setAttribute("serverr", server);
request.getSession().setAttribute("usernamee", username);
request.getSession().setAttribute("passWordd", password);

// Java to Jsp
request.setAttribute("user", username);

if (loginFlag) {
System.out.println("\nLogin Success !!! ");
path = "/chat.jsp";
} else {
System.out.println("\nLogin Failed !!! ");
path = "/register.jsp";
}
} else if (action.equals("logout")) {

boolean logoutFlag = tmp.logout();
if (logoutFlag) {
System.out.println("\nLogout Success!!! ");
path = "/login.jsp";
} else {
System.out.println("\nLogout Failed !!! ");
}
} else if (action.equals("regis")) {
path = "/register.jsp";
} else if (action.equals("register")) {
```

```
boolean regisFlag = tmp.Register(r_server, r_username, r_password);

if (regisFlag) {
System.out.println("\nRegister Success !!! ");
path = "/login.jsp";
} else {
System.out.println("\nRegister Failed !!! ");
path = "/register.jsp";
}
} else if (action.equals("cancel")) {
path = "/login.jsp";
} else if (action.equals("send")) {

String usernamee = (String) request.getSession().getAttribute(
"usernamee");
request.setAttribute("user", usernamee);

boolean SendFlag = tmp.sendMessage(message, contract);

if (SendFlag) {
// System.out.println("Send Success !!! ");
path = "/wchat.jsp";
} else {
System.out.println("\nSend Failed !!! ");
path = "/wchat.jsp";
}
} else if (action.equals("ShowBuddy")) {

String usernamee = (String) request.getSession().getAttribute(
"usernamee");
request.setAttribute("user", usernamee);
```

```
boolean ShowFlag = tmp.displayBuddyList();
if (ShowFlag) {
System.out.println("\nShow Success !!! ");
path = "/chat.jsp";
} else {
System.out.println("\nShow Failed !!! ");
path = "/chat.jsp";
}
} else if (action.equals("AddBuddy")) {

String usernamee = (String) request.getSession().getAttribute(
"usernamee");
request.setAttribute("user", usernamee);

boolean AddFlag = tmp.AddBuddyList(buddy);
if (AddFlag) {
System.out.println("\nAdd Success !!! ");
path = "/chat.jsp";
} else {
System.out.println("\nAdd Failed !!! ");
path = "/chat.jsp";
}
} else if (action.equals("DelBuddy")) {

String usernamee = (String) request.getSession().getAttribute(
"usernamee");
request.setAttribute("user", usernamee);

boolean DelFlag = tmp.DeleteBuddyList(buddy);
if (DelFlag) {
System.out.println("\nDel Success !!! ");
path = "/chat.jsp";
```

```
} else {
System.out.println("\nDel Failed !!! ");
path = "/chat.jsp";
}
} else if (action.equals("display")) {

String usernamee = (String) request.getSession().getAttribute(
"usernamee");
request.setAttribute("user", usernamee);

boolean DisFlag = tmp.display(display);
if (DisFlag) {
System.out.println("\nSet Success !!! ");
path = "/chat.jsp";
} else {
System.out.println("\nSet Failed !!! ");
path = "/chat.jsp";
}
}
final RequestDispatcher dispatch = getServletContext()
.getRequestDispatcher("" + path);
dispatch.forward(request, response);
}

public void init() throws ServletException {
// Put your code here

}
}
```

ไฟล์ Buddybox

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<form name="chat" action="Servlet1" method="POST">
<input type="hidden" name="action" />
Buddy :
<br>
<input type="text" name="buddy">
<br>
<input type="submit" value="AddBuddy"
onclick="javascript:document.chat.action.value = 'AddBuddy'">
<br>
<input type="submit" value="DelBuddy"
onclick="javascript:doeument.chat.action.value = 'DelBuddy'">
<br>
<input type="submit" value="ShowBuddy"
onclick="javascript:document.chat.action.value = 'ShowBuddy'">
</form>
</body>
</html>
```

ไฟล์ Chat

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<html>
<head>
<title>JWC - Jabber Web Client</title>
<script language="JavaScript" type="text/javascript">
/* Settings you might want to define */
//var waittime=800;

//String user = (String)request.getAttribute("user");

/* Internal Variables & Stuff */
chatmsg.focus();
document.getElementById("chatwindow").innerHTML = "";
var xmlhttp = false;
var xmlhttp2 = false;
var check = true;

function startChat() {
//Set the focus to the Message Box.
document.getElementById('chatmsg').focus();
//Start Recieving Messages.
getChatText();
}

/* Request for Reading the Chat Content (receive ajax)*/
function ajax_read(url) {
if(window.XMLHttpRequest){
xmlhttp=new XMLHttpRequest();
if(xmlhttp.overrideMimeType){
xmlhttp.overrideMimeType('text/xml');
```

```
}
} else if(window.ActiveXObject){
try{
xmlhttp=new ActiveXObject("Msxml2.XMLHTTP");
} catch(e) {
try{
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
} catch(e){
}
}
}
if(!xmlhttp) {
alert('Giving up :( Cannot create an XMLHTTP instance');
return false;
}

/*AJAX state change*/
xmlhttp.onreadystatechange = function() {
if (xmlhttp.readyState==4) {
//Chat section
if (check == false){
document.getElementById("div_list").innerHTML = xmlhttp.responseText;
time = new Date();
ms = (time.getHours() * 24 * 60 * 1000) + (time.getMinutes() * 60 * 1000) + (time.getSeconds()
* 1000) + time.getMilliseconds();
intUpdate = setTimeout("ajax_read('chat.txt?x=" + ms + "')", waittime);

}
else (check == true)
{ document.getElementById("div_option").innerHTML = xmlhttp.responseText;

time = new Date();
```

```
ms = (time.getHours() * 24 * 60 * 1000) + (time.getMinutes() * 60 * 1000) + (time.getSeconds()
* 1000) + time.getMilliseconds();
intUpdate = setTimeout("ajax_read('chat.txt?x=" + ms + "')", waittime);}
}
}
xmlhttp.open('GET',url,true);
xmlhttp.send(null);
}

/* Request for Writing the Message (send ajax)*/
function ajax_write(url){
if(window.XMLHttpRequest){
xmlhttp2=new XMLHttpRequest();
if(xmlhttp2.overrideMimeType){
xmlhttp2.overrideMimeType('text/xml');
}
} else if(window.ActiveXObject){
try{
xmlhttp2=new ActiveXObject("Msxml2.XMLHTTP");
} catch(e) {
try{
xmlhttp2=new ActiveXObject("Microsoft.XMLHTTP");
} catch(e){
}
}
}
if(!xmlhttp2) {
alert('Giving up :( Cannot create an XMLHTTP instance');
return false;
}
xmlhttp2.open('GET',url,true);
xmlhttp2.send(null);
```

```
}

function changediv(text){
check = true;
div_option.innerHTML = text;
}

function Linkup(){
var option = document.FSL.SL.selectedIndex;
location.href = document.FSL.SL.options[option].value;
}

function addbuddy(){
check = true;
ajax_read("buddybox.jsp");
xmlhttp.open('GET',"buddybox.jsp",true);
xmlhttp.send(null);
}

function display(){
check = true;
ajax_read("disbox.jsp");
xmlhttp.open('GET',"disbox.jsp",true);
xmlhttp.send(null);
}

function showlist(){
check = false;
ajax_read("listbox.jsp");
xmlhttp.open('GET',"listbox.jsp",true);
xmlhttp.send(null);
}
```

```
function conchat(){
check = false;
ajax_read("JWC.jsp");
xmlhttp.open('GET',"JWC.jsp",true);
xmlhttp.send(null);
}

/* Submit the Message */
function sendmsg(){
Contract = document.getElementById("chatbuddy").value;
// check null buddy
if (Contract == "") {
check = prompt("please enter buddy:");
if (check == null) return 0;
if (check == "") check = "anonymous";
document.getElementById("chatbuddy").value = check;
Contract = check;
}
ajax_read("wchat.jsp");
xmlhttp.open('GET',"wchat.jsp",true);
xmlhttp.send(null);

}

/* Check if Enter is pressed */
function keyup(arg1) {
if (arg1 == 13){
sendmsg();
}

}
```

```
function newwindows() {
newwindow = window.open ('wchat.jsp', 'newwindow','width=370,
height=500,resizable=no,scrollbars=no,toolbar=no,location=no,directories=no,status=no,menubar
=no,copyhistory=no');

}
/* Start the Requests! ;) */
var intUpdate = setTimeout("ajax_read('chat.txt')", waittime);
alert("3");
</script>
</head>

<style type="text/css">
body                    { padding-left:300px; background:#57767F; font-family:arial;}
input, textarea { font-family: arial; color:white; background:#57767F; font-size: 14px; }
#content                { width:800px; text-align:left; margin-left:60px; }
#chatwindow     { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:550px; height:auto; font-family:courier new;}
#div_buddy          { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:200px; height:auto; font-family:courier new;}
#div_list           { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:200px; height:auto; font-family:courier new;}
#div_option     { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:200px; height:auto; font-family:courier new;}
#chatmsg                { border:1px solid #aaaaaa; border-bottom:1px solid #aaaaaa;
padding:4px; background:#232D2F; }
#info                   { text-align:left; padding-left:100px; font-family:arial; }
#info td            { font-size:12px; padding-right:10px; color:#DFDFDF;  }
#info .small        { font-size:12px; padding-left:10px; padding-right:0px;}
#info a             { text-decoration:none; color:white; }
#info a:hover       { text-decoration:underline; color:#CF9700;}
</style>
```

```
<body onLoad="javascript:addbuddy();">
<div align="center">
</div>
<div id="info">
<div align="right">
</div>
<table border="0" width="531" height="505">
<tr>
<td colspan="1" align="center">
<div align="right">
</div>
<div align="center">
<div align="center">
<div align="center">
<font style="font-size: 16px;">         
</font>
<br>
</div>
<font style="font-size: 16px;">     <br>
<br>
<br>
<br>
</font>
<div align="left">
<font style="font-size: 16px;"> </font><font
style="font-size: 16px;"> [ JWC - Jabber Web
Client  ]</font>
<br>

</div>
</div>
```

```
<div align="center">
<div align="left">
<br>
</div>
</div>
<font style="font-size: 16px;"> </font>
</div>
</td>
</tr>
<tr>
<td valign="top">
   [Option Window]
<form name="FSL">
<%
String user = (String) request.getAttribute("user");
out.print("Login user : " + user);
%>
    
<select name="SL" onChange="Linkup(this.form)">
<br>
<br>
<br>
<option value="javascript:changediv('<br>');">
OPTION
</option>
<option value="javascript:addbuddy();">
ADD BUDDY
</option>
<option value="javascript:display();">
DISPLAY MESSAGE
</option>
</select>
```

```
</form>
</td>
</td>
</tr>
<tr>
<td valign="top">
<table width="296" height="319" border="0">
<tr>
<td valign="top">
<div id="div_option"
style="height: 200px; width: 250px; overflow: auto;">
</div>
<div align="left">
<form name="logout" action="Servlet1" method="post">
<input type="hidden" name="action">
<input name="submit" type="submit"
style="border: 1px solid gray; cursor: pointer;"
onClick="javascript:document.logout.action.value = 'logout'"
value="Logout">
<input name="button" type="button"
style="border: 1px solid gray; cursor: pointer;"
onClick="newwindows();" value="START CHAT">
<input name="text" type="hidden" id="hidd"
style="cursor: pointer;" onKeyUp="keyup(event.keyCode);"
size="20" maxlength="20";>
</form>
</div>
</td>
</table>
</td>
</table></form></body></html>
```

ไฟล์ Disbox

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<form name="chat" action="Servlet1" method="POST">
<input type="hidden" name="action" />
DISPLAY MESSAGE :
<br>
<input type="text" name="display">
<br>
<input type="submit" value="SET MESSAGE"
onclick="javascript:document.chat.action.value = 'display'">
<br>
</form>
</body>
</html>
```

ไฟล์ Login

```
<HTML>
<HEAD>
<style type="text/css">
body                    { padding-left:40px; background:#57767F; font-family:arial;}
</style>
<TITLE>Login page</TITLE>
<meta http-equiv="Content-Type" content="text/html; charset=tis-620">
</HEAD>
<BODY>
<form name="login" action="Servlet1" method="POST">
<input type="hidden" name="action" />
<br><br><br><br><br><br><br><br><br>
<table align="center" width="286" height="152" border="0"
background="picture/bg_log2.bmp">
<tbody>
<tr>
<td>
<table width="286" border="0" align="center" cellpadding="1"
cellspacing="1">
<tr>
<td colspan="2" align="center">
<strong>Login</strong>
</td>
</tr>
<tr>
<td>
<strong>Username</strong>
</td>
<td>
<input type="text" name="username" size="20" maxlength="20">
</td>
```

```
</tr>
<tr>
<td>
<strong>Password</strong>
</td>
<td>
<input type="password" name="password" size="20"
maxlength="20">
</td>
</tr>
<tr>
<td>
<strong>Server</strong>
</td>
<td>
<input type="text" name="server" size="20" maxlength="20">
</td>
</tr>
<tr>
<td>
 
</td>
<td>
<input type="submit" value="Login"
onclick="javascript:document.login.action.value = 'login'">

    
<input type="submit" value="Register"
onclick="javascript:document.login.action.value = 'regis'">
</td>
</tr>
</table>
```

```
</td>
</tr>
</tbody>
</table>
</BODY>
</form>
</HTML>
```

ไฟล์ Register

```
<HTML>
<HEAD>
<style type="text/css">
body                    { padding-left:40px; background:#57767F; font-family:arial;}
</style>
<TITLE>Register page</TITLE>
<meta http-equiv="Content-Type" content="text/html; charset=tis-620">
</HEAD>
<BODY>
<form name="Register" action="Servlet1" method="POST">
<input type="hidden" name="action" />
<br><br><br><br><br><br><br><br><BR>

<table align="center" width="286" height="152" border="0"
background="picture/bg_log2.bmp">
<tbody>
<tr>
<td>
<table width="286" border="0" align="center" cellpadding="1"
cellspacing="1">
<tr>
<td colspan="2" align="center">
```

```
<strong>Register</strong>
</td>
</tr>
<tr>
<td>
<strong>Username</strong>
</td>
<td>
<input type="text" name="r_username" size="20" maxlength="20">
</td>
</tr>
<tr>
<td>
<strong>Password</strong>
</td>
<td>
<input type="password" name="r_password" size="20"
maxlength="20">
</td>
</tr>
<tr>
<td>
<strong>Server</strong>
</td>
<td>
<input type="text" name="r_server" size="20" maxlength="20">
</td>
</tr>
<tr>
<td>
 
</td>
```

```
<td>
<input type="submit" value="Register"
onclick="javascript:document.Register.action.value = 'register'">
<input type="submit" value="Cancel"
onclick="javascript:document.Register.action.value = 'cancel'">
</td>
</tr>
</table>
</td>
</tr>
</tbody>
</table>
</BODY>
</form>
</HTML>
```

ไฟล์ Wchat

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="TIS-620"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1" />
<title>Chat Windows</title>

<style type="text/css">
body                    { background:#57767F; font-family:arial;}
input, textarea { font-family: arial; color:white; background:#57767F; font-size: 14px; }
#content                { width:800px; text-align:left; margin-left:60px; }
```

```
#chatwindow     { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:550px; height:auto; font-family:courier new;}
#chatpage       { border:1px solid #aaaaaa; padding:4px; background:#232D2F; color:white;
width:340px; height:auto; font-family:courier new;}
#message        { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:200px; height:auto; font-family:courier new;}
#chatmsg        { border:1px solid #aaaaaa; border-bottom:1px solid #aaaaaa;
padding:4px; background:#232D2F; }
#contract       { border:1px solid #aaaaaa; border-bottom:1px solid #aaaaaa;
padding:4px; background:#232D2F; }
#info           { text-align:left; padding-left:100px; font-family:arial; }
#info td        { font-size:12px; padding-right:10px; color:#DFDFDF;  }
#info .small    { font-size:12px; padding-left:10px; padding-right:0px;}
#info a         { text-decoration:none; color:white; }
#info a:hover   { text-decoration:underline; color:#CF9700;}
.style1 {color: #FFFFFF}
</style>
</head>
<body>
<%
String userCont = request.getParameter("contract");
%>
<form name="chat" action="Servlet1" method="POST">
<input type="hidden" name="action" />
Chat with :
<input type="text" value="<%=userCont%>" name="contract" id="contract" >
<td valign="bottom">
<textarea name="chatpage" id="chatpage" rows="20" readonly
style="width:350; overflow: auto;"><%
String msg = request.getParameter("message");
String remsg = (String) application.getAttribute("remsg");
String chat = (String) application.getAttribute("chat");
```

```
String chat2 = (String) application.getAttribute("chat2");

String user = (String) request.getAttribute("user");

if (msg != null) {
//print conversation message
String conversation = "[" + user + "] Say: " + msg;
if(chat == null){
chat = conversation + "\n";
}else{
chat += conversation + "\n";
}
application.setAttribute("chat", chat);
out.print(chat);
if (remsg != null) {
String buddy = "[" + "Contact" + "] Say:" + remsg;
if(chat2 == null){
chat2 = buddy + "\n";
}else{
chat2 += buddy + "\n";
}
application.setAttribute("chat2", chat2);
out.print(chat2);
}
}
%>
</textarea></td><br>
<input typc="text" name="message" id="message">
<input type="submit" name="Submit" value="Send"
onclick="javascript:document.chat.action.value = 'send'">
  
</form></body></html>
```

Model code 2: Pure Servlet

Servlet :- Login

- Chat

ไฟล์ Login

```
package org.jivesoftware.smack;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.Servlet;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class login extends HttpServlet implements Servlet {

protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

response.setStatus(HttpServletResponse.SC_OK);
response.setContentType("text/html");
PrintWriter out = response.getWriter();

out.println("<HTML>");
out.println("<style type=\"text/css\">");
out.println("body          { padding-left:40px; background:#57767F; font-family:arial;}");
out.println("input      { font-family: arial; color:white; background:#57767F; font-size: 14px; }");
out.println("#txtId     { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
```

```
out.println("#txtPass  { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtServ      { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("</style>");
out.println("</HEAD>");
out.println(" <meta http-equiv=\"content-type\" content=\"text/html; charset=tis-620\">");
out.println("<BODY>");
out.println("<BODY>");
out.println("<br><br><br><br><br><br><br><br><br><br>");
out.println("<table align=\"center\" width=\"286\" height=\"152\" border=\"0\">");
out.println("<tr>");
out.println("<td>");
out.println("<table width=\"28\" border=\"0\" align=\"center\" cellpadding=\"1\"
cellspacing=\"1\">");
out.println("<tr>");
out.println("<td colspan=\"2\" align=\"center\">");
out.println("<strong>LOGIN</strong>");
out.println("<br>");
out.println("</td>");
out.println("</tr>");
out.println("<form name=myForm action=chat method=Post>");
out.println("<tr>");
out.println("<td>");
out.println("<strong>Username</strong>");
out.println("</td>");
out.println("<td>");
out.println("<input type=text name=txtId id=txtId size=20 maxlength=20>");
out.println("</td>");
out.println("</tr>");
out.println("<tr>");
out.println("<td>");
```

```
out.println("<strong>Password</strong>");
out.println("</td>");
out.println("<td>");
out.println("<input type=password name=txtPass id=txtPass size=20 maxlength=20>");
out.println("</td>");
out.println("</tr>");
out.println("<tr>");
out.println("<td>");
out.println("<strong>Server</strong>");
out.println("</td>");
out.println("<td>");
out.println("<input type=text name=txtServ id=txtServ size=20 maxlength=20>");
out.println("</td>");
out.println("</tr>");
out.println("<tr>");
out.println("<td>");
out.println(" ");
out.println("</td>");
out.println("<td align=\"center\">");
out.println("<input type=submit value=Login name=myButton>");
out.println("</td>");
out.println("</form>");
out.println("</tr>");
out.println("</table>");
out.println("</td>");
out.println("</tr>");
out.println("</table>");
out.println("</BODY></HTML>");
}

protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
```

```
doGet(request, response);
}
}
```

ไฟล์ Chat

```
package org.jivesoftware.smack;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Collection;
import javax.servlet.Servlet;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.jivesoftware.smack.Chat;
import org.jivesoftware.smack.filter.*;
import org.jivesoftware.smack.packet.*;
import org.jivesoftware.smack.ConnectionConfiguration;
import org.jivesoftware.smack.MessageListener;
import org.jivesoftware.smack.Roster;
import org.jivesoftware.smack.RosterEntry;
import org.jivesoftware.smack.XMPPConnection;

public class chat extends HttpServlet implements Servlet ,MessageListener {
String chat = null , remsg = null;

public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
```

```
response.setContentType("text/html");
PrintWriter out = response.getWriter();

/********** Request variable from JSP page **********/
// Login variable
String userName = request.getParameter("txtId");
String userPass = request.getParameter("txtPass");
String userServ = request.getParameter("txtServ");

// Contact variable
String userCont = request.getParameter("txtCont");

// Message variable
String userMsg = request.getParameter("txtmsg");

//*****************************************************************//
//                          Function                               //
//*****************************************************************//

// XMPP Connection
XMPPConnection connection = null;
try {
ConnectionConfiguration config = new ConnectionConfiguration(userServ, 5222);
connection = new XMPPConnection(config);
connection.connect();
connection.login(userName, userPass);

} catch (XMPPException ex) {
System.out.println(ex);
}
```

```
// Show contact list
Roster roster = connection.getRoster();
Collection<RosterEntry> entries = roster.getEntries();
int count1=1 ,count2=1, num1=0, num2=0;
String list1=null, list2=null;
for(RosterEntry r:entries)
{
Presence presence = roster.getPresence(r.getUser());
if (presence.getType() == Presence.Type.available){
if (count1 == 1){
list1 = r.getUser() +"\n" ;
count1 = 2;
num1++;
}
else
{
list1 += r.getUser() +"\n" ;
num1++;
}
}else{
if (count2 == 1){
list2 = r.getUser() +"\n" ;
count2 = 2;
num2++;
}
else
{
list2 += r.getUser() +"\n" ;
num2++;
}

}
```

```
}

//*********************************************************************//
//                          show page                                  //
//*********************************************************************//
out.println("<HTML>");
out.println("<HEAD><TITLE>Jabber Instant Messaging Web Client</TITLE>");
// style
out.println("<style type=\"text/css\">");
out.println("body                    { padding-left:40px; background:#57767F; font-
family:arial;}");
out.println("input, textarea { font-family: arial; color:white; background:#57767F; font-size:
14px; }");
out.println("#txtDis          { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtCont         { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtChat         { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtList         { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtmsg          { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#txtBuddy        { border:1px solid #aaaaaa; padding:4px; background:#232D2F;
color:white;  width:auto; height:auto; font-family:courier new;}");
out.println("#info                   { text-align:left; padding-left:100px; font-family:arial; }");
out.println("#info td                { font-size:12px; padding-right:10px; color:#DFDFDF;  }");
out.println("#info .small   { font-size:12px; padding-left:10px; padding-right:0px;}");
out.println("#info a                 { text-decoration:none; color:white; }");
out.println("#info a:hover           { text-decoration:underline; color:#CF9700;}");
out.println(".style1       {color: #FFFFFF}");
out.println("</style>");
```

```
out.println("</HEAD>");
out.println(" <meta http-equiv=\"content-type\" content=\"text/html; charset=tis-620\">");
out.println("<BODY>");
out.println("<br><br>");
out.println("<strong>Jabber Instant Messaging Web Client</strong>");
out.println("<br><br>");
out.println("<textarea name=txtChat id=txtChat rows=20 cols=70 style=\"overflow: auto;\"
readonly>");
//Send and Receive Message
try{
final Chat chat = connection.getChatManager().createChat(userCont, this);
if(userMsg != null){
chat.sendMessage(userMsg);
}
// Accept only messages from Contract
PacketFilter filter = new AndFilter(new PacketTypeFilter(Message.class),new
FromContainsFilter(chat.getParticipant()));

PacketListener myListener = new PacketListener() {
public void processPacket(Packet packet) {
try {
if (packet instanceof Message) {
Message msg = (Message) packet;
// Process message
if(msg.getType() == Message.Type.chat){
remsg = msg.getBody();
}
}
}catch(Exception ex){
ex.printStackTrace();
}
}
```

```
};
// Register the listener.
connection.addPacketListener(myListener, filter);
} catch (Exception ex) {
ex.printStackTrace();
}

if (userMsg != null) {
//print conversation message
String conversation = "[" + userName + "] Say: " + userMsg + "\n";
if(chat == null){
chat = conversation;
}else{
chat =chat + conversation;
}
}
if (remsg != null) {
String buddy = "[" + userCont + "] Say:" + remsg + "\n";
if(chat == null){
chat = buddy;
}else{
chat =chat + buddy;
}
}
if(chat != ""){
out.println(chat);
}
out.println("</textarea>");
//List box
out.println("<textarea name=txtList id=txtList rows=20 cols=30 readonly>");
out.println("have " + entries.size() + " buddy in list:");
if(list1 != null){
```

```
out.println(list1);
}
out.println(list2);
out.println("</textarea>");
out.println("<br><br>");
out.println("<form name=myForm action=chat method=GET>");
out.println("<input type=hidden value="+userName+" name=txtId>");
out.println("<input type=hidden value="+userPass+" name=txtPass>");
out.println("<input type=hidden value="+userServ+" name=txtServ>");
//Message box
out.println("<strong>Message : </strong>");
out.println("<input type=text name=txtmsg id=txtmsg >");
out.println("<strong> To : </strong>");
out.println("<input type=text value="+userCont+" name=txtCont id=txtCont >");
//Send message button
out.println("<input type=submit value=Send name=myButton>");
out.print("</FORM>");
//Logout button
out.println("<form name=myForm2 action=login method=GET>");
out.println("<input name=button type=submit value= Logout");
out.println("</from>");
out.println("</BODY>");
out.println("</HTML>");
}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
doGet(request, response);
}
public void processMessage(Chat arg0, Message arg1) {
// TODO Auto-generated method stub }
}
```

ภาคผนวก ข

การติดตั้ง Tomcat 5.X บน Windows XP/2000/2003

1. เซต สภาพแวดล้อมการทำงานของ Java 5

เมื่อได้ Install JAVA เสร็จ ถัดไปจะต้องกำหนด Environment Variables โดยตัวแปรที่เราต้องเซตมีดังนี้

1.1 JAVA_HOME = C:\Program Files\Java\jdk1.5.0_07

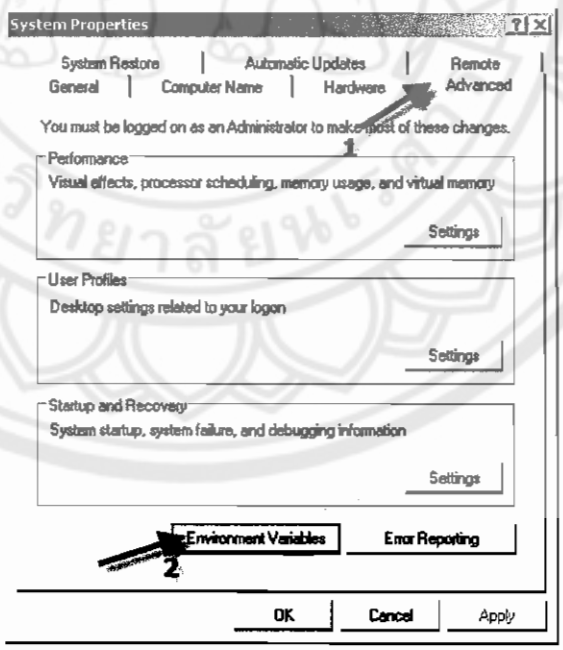
1.2 JRE_HOME = C:\Program Files\Java\jre1.5.0_07

1.3 CLASSPATH = %JRE_HOME%\jre\lib;

1.4 PATH = %JAVA_HOME%\bin

2. วิธีกำหนด Environment Variables

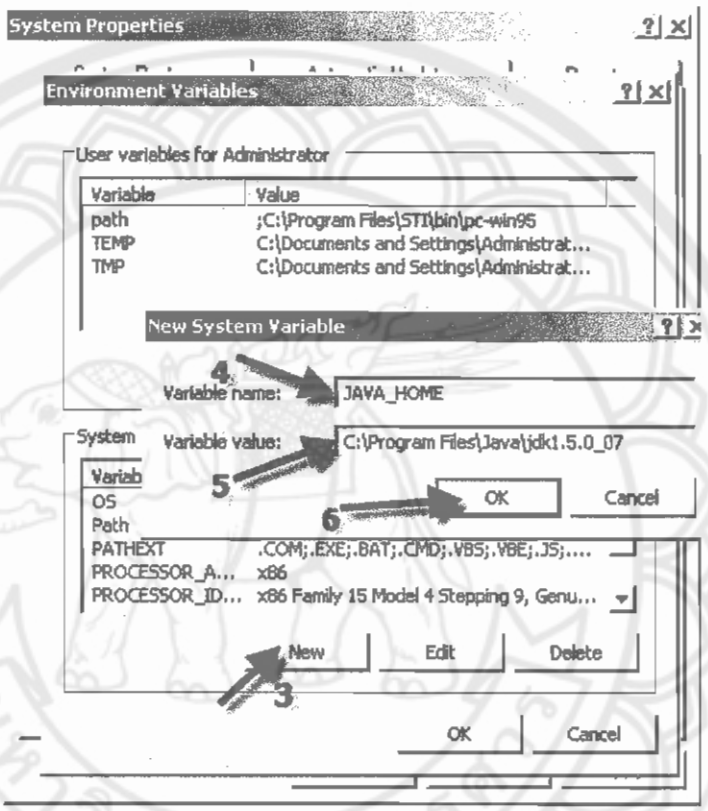
2.1 คลิ๊กขวาบน My Computer แล้วเลือก Properties จะปรากฏ System Properties ขึ้นมา จากนั้นให้กดที่แท็บ **Advanced (1)** ตามรูปภาพภาพ



รูปที่ ข.1 การเข้าไปเซตตัวแปร Environment Variables

2.2 เมื่อคลิกที่แท็บ Advanced แล้ว ให้คลิกบนปุ่ม **Environment Variables (2)** ตามรูปภาพ แล้วจะปรากฏหน้าต่าง Environment Variables ขึ้นมาจากนั้นให้สร้างตัวแปรโดยคลิกที่ปุ่ม **New**

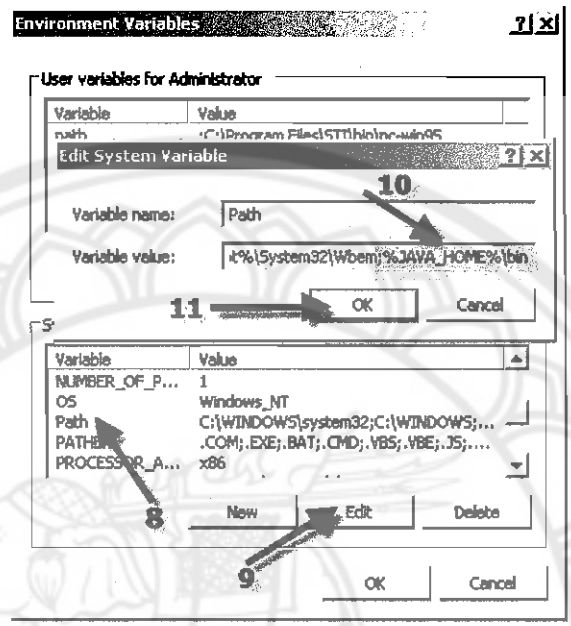
(3) แล้วใส่**ชื่อตัวแปร (4)** และ **ค่าของตัวแปร (5)** เมื่อเสร็จแล้วให้คลิกปุ่ม **OK (6)**



รูปที่ ข.2 การสร้างและใส่ค่าตัวแปร

สร้างตัวแปรทีละตัว ให้ครบทั้ง 3 ตัวแปร คือ JAVA_HOME, JRE_HOME, CLASSPATH

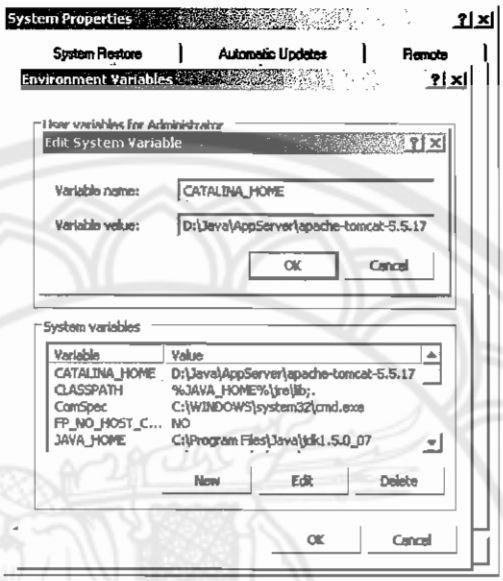
ส่วนตัวแปร PATH เราจะไม่สร้างขึ้นใหม่ เนื่องจาก มีอยู่แล้วจึงไม่ต้องสร้างใหม่ ให้ Edit แล้ว กำหนดค่าตามรูปภาพที่ ข.3



รูปที่ ข.3 การเข้าไป Edit ค่าในPath

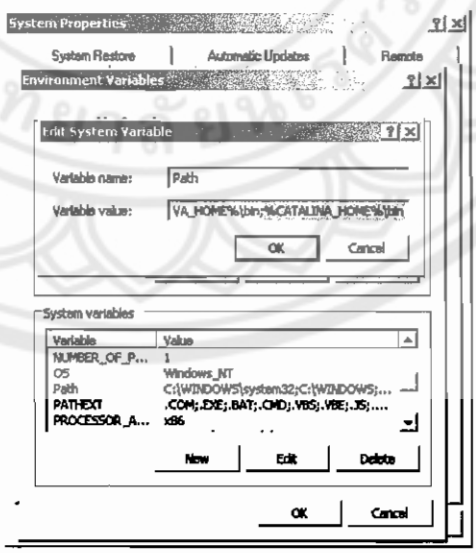
3. การเซต Environment Variables ของ Tomcat จะเซตตัวแปร 2 ตัวคือ

3.1 CATALINA_HOME = D:\Java\AppServer\apache-tomcat-5.5



รูปที่ ข.4 การเซต Catalina_home

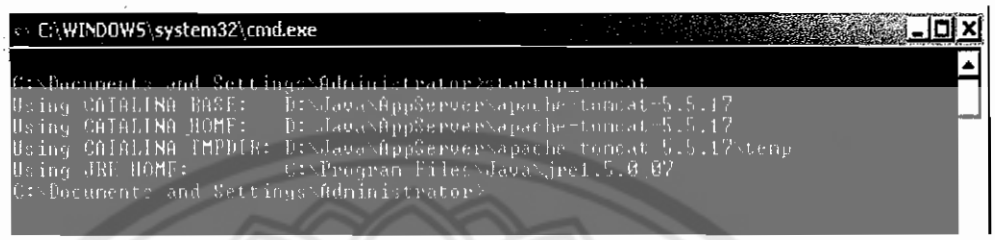
3.2 PATH = %CATALINA_HOME%\bin



รูปที่ ข.5 การเซต Path

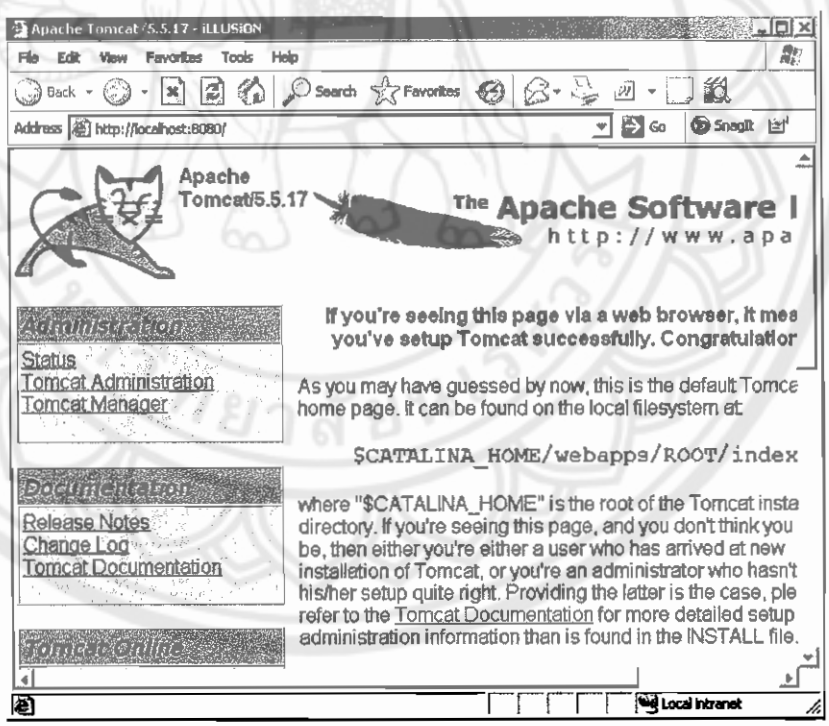
4. การ Start Tomcat

4.1 ให้ไปที่ command prompt แล้วพิมพ์ startup_tomcat



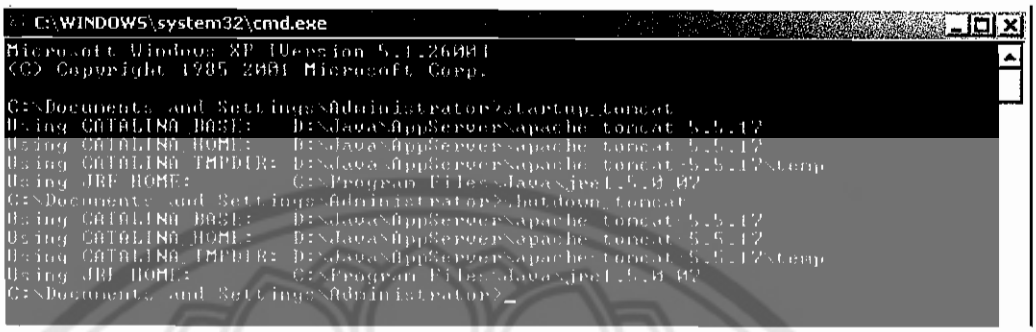
รูปที่ ข.6 การ Start Tomcat

4.2 Tomcat ทำงานหรือไม่ ให้เปิด Browser ขึ้นมา แล้วลองเรียกไปที่ http://localhost:8080 ถ้าไม่มีข้อผิดพลาดจะได้ผลลัพธ์ตามรูปภาพที่ ข.7



รูปที่ ข.7 Tomcat

5. การ Shutdown Tomcat

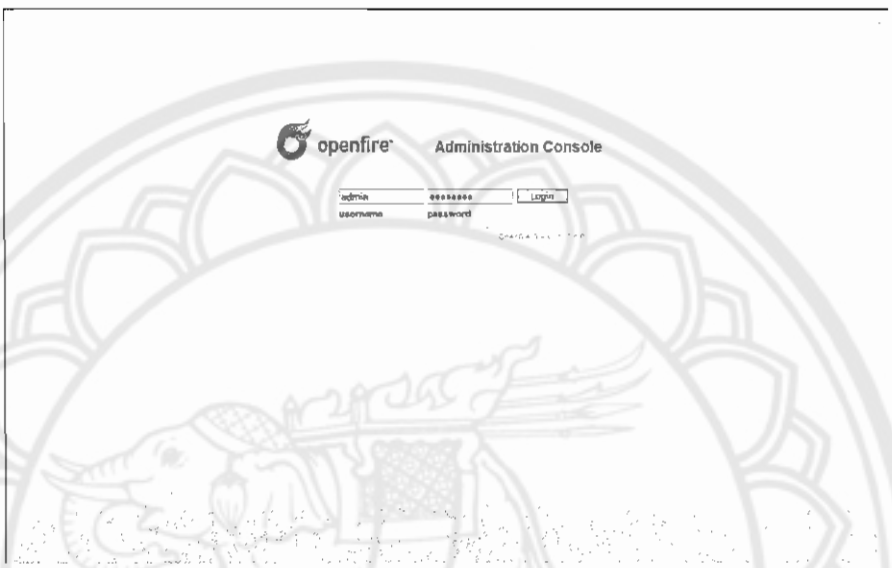


รูปที่ ข.8 การ Shutdown Tomcat

ภาคผนวก ค

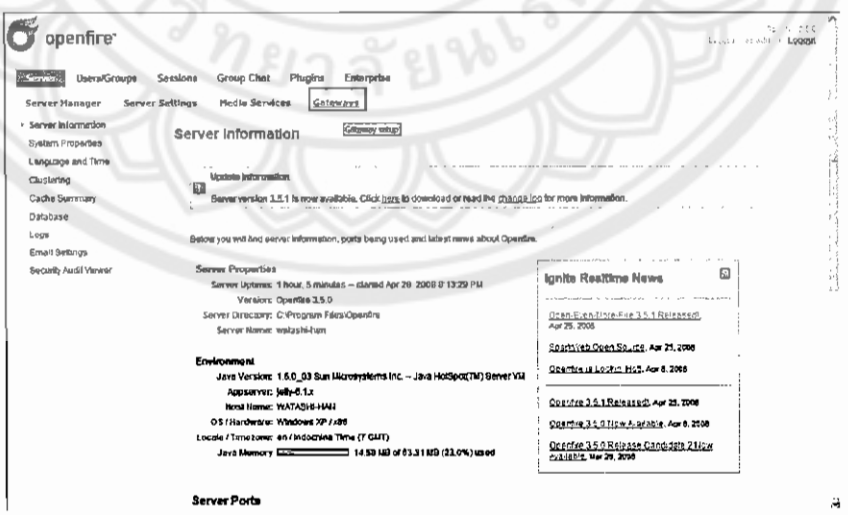
การใช้ IM Gateway เพื่อติดต่อกับ โปรโตคอลอื่น

1. Login เข้าไปที่ Openfire Server



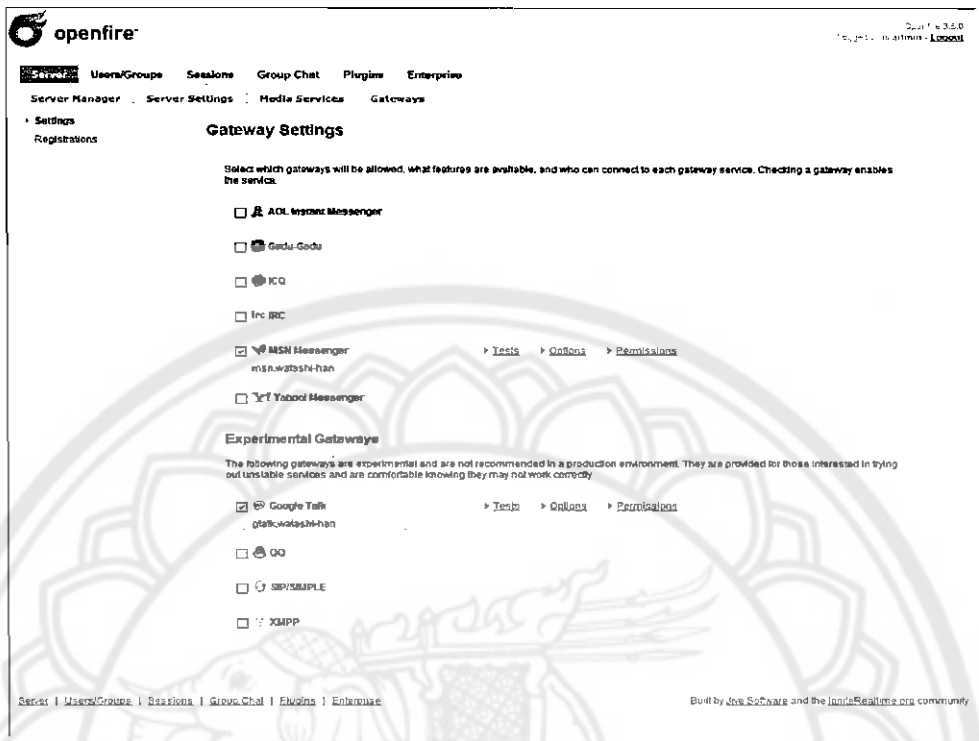
รูปที่ ค.1 หน้าLogin ของ Openfire Server

2. คลิ๊กเข้าไปที่ Gateway ดังรูปภาพที่ ค.2



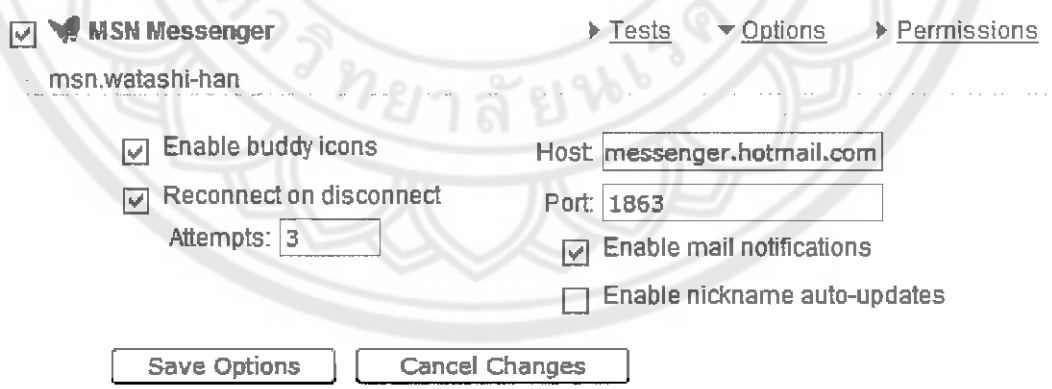
รูปที่ ค.2 หน้าแรกของ Openfire Server

3. เลือกโปรโตคอลที่ต้องการจะติดต่อ โดยเช็คเครื่องหมายถูก ในช่องสี่เหลี่ยม



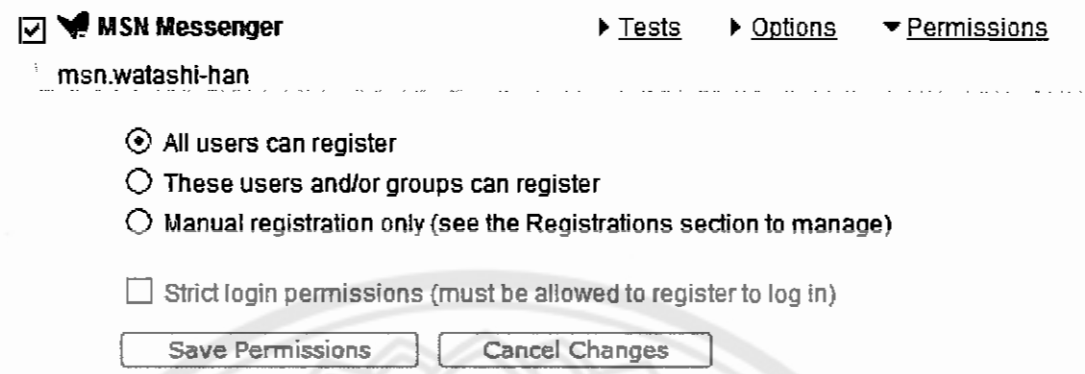
รูปที่ ค.3 การเลือกโปรโตคอลที่จะติดต่อ

4. ใส่ข้อมูลลงในช่องตามรูปที่ ค.4



รูปที่ ค.4 วิธีการใส่ข้อมูลเพื่อสร้างการติดต่อกับ MSN

5. กำหนดสิทธิการใช้ให้กับ โปรโตคอล MSN

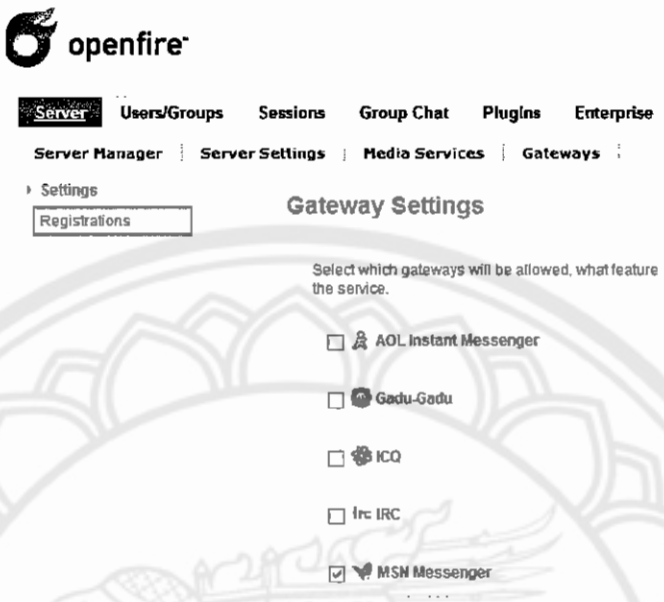


รูปที่ ค.5 การกำหนดสิทธิให้กับโปรโตคอลMSN

6. ทดสอบการติดต่อว่าสามารถติดต่อได้หรือไม่ โดยการคลิ๊กที่ Test Connection

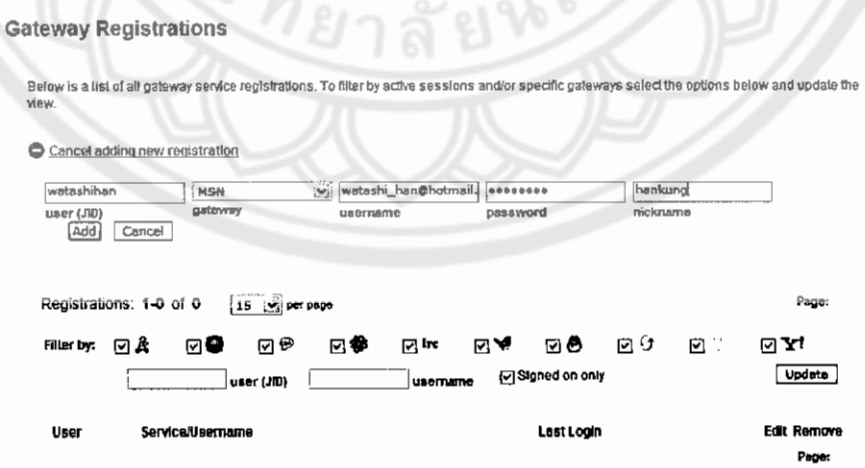
รูปที่ ค.6 การทดสอบการติดต่อ

7. คลิ๊กที่ Registrations



รูปที่ ค.7 การเข้าไปเพิ่มการเชื่อมต่อ

8. เมื่อเข้ามาแล้วก็คลิ๊กที่ Add new registrations จะขึ้นหน้าต่างดังรูปภาพ ค.8 ขึ้นมา ให้เราใส่ข้อมูลลงไป โดยช่องแรกให้ใส่ ID ที่เราสมัครไว้กับ Openfire Server แล้วก็เลือก Gateway ว่าจะใช้เชื่อมต่อกับ โปรโตคอลไหน จากนั้นในช่อง username ก็ให้ใส่ ชื่อ E-mail ของ โปรโตคอลที่เลือก เช่น XXX@hotmail.com แล้วก็พาสเวิร์ดให้ถูกต้อง แล้วก็คลิ๊กที่ Add

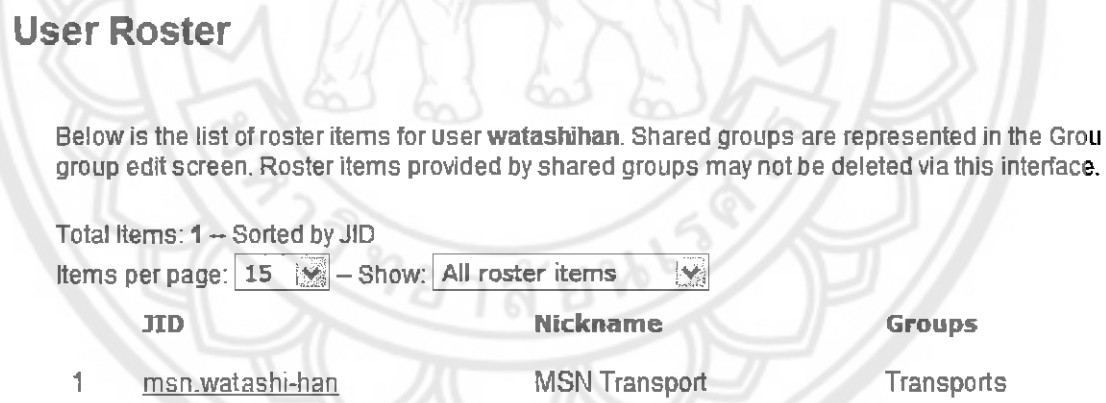


รูปที่ ค.8 การกำหนดให้ JID ของ Openfire ให้สามารถติดต่อกับ MSN

9. เข้าไปที่ Roster โดยคลิ๊กที่ Roster ดังรูปภาพที่ ค.9

รูปที่ ค.9 การเข้าไปหน้าของ Roster

10. เมื่อเข้ามาแล้วก็จะเห็นว่ามี JID ที่สามารถติดต่อกับ โปรโตคอลของ MSN ได้แล้ว

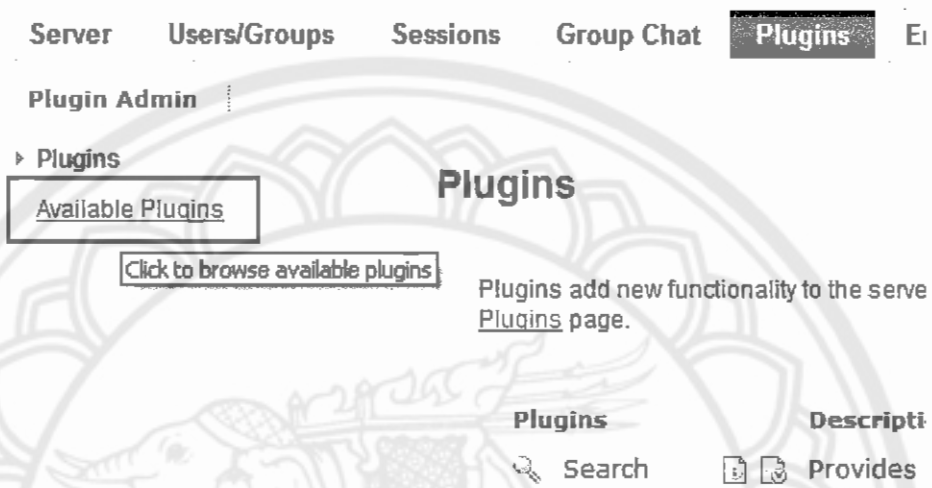


รูปที่ ค.10 แสดง JID ที่สามารถติดต่อได้กับ โปรโตคอลของ MSN

ภาคผนวก ง

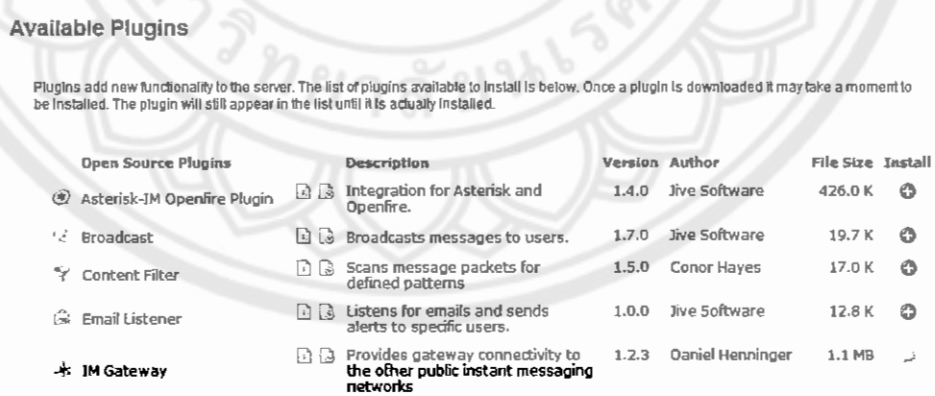
การนำ Gateway Plugin มาใช้

1. เปิด Openfire ไปที่หน้าของ Plugin แล้วคลิ๊กที่ Available Pulgins



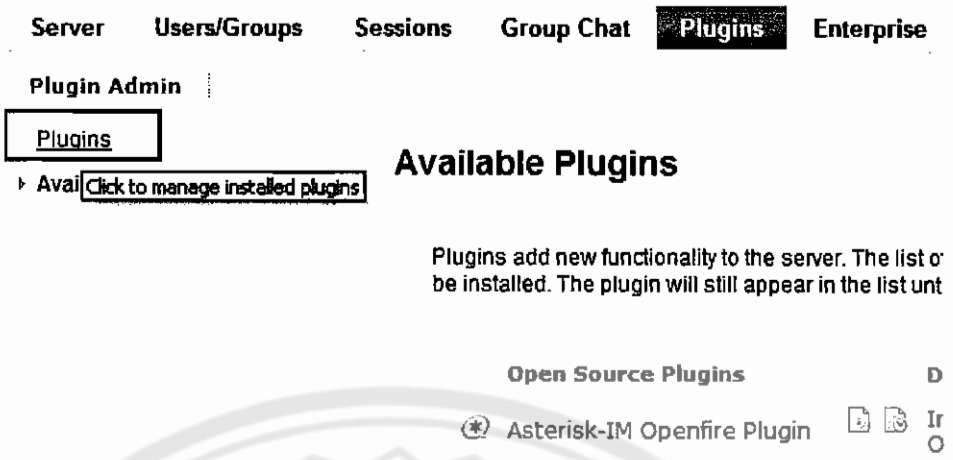
รูปที่ ง.1 การเข้าไปเลือก Gateway Plugin มาใช้

2. เลือกPlugin ที่ชื่อว่า IM Gateway แล้วคลิ๊กที่เครื่องหมาย +



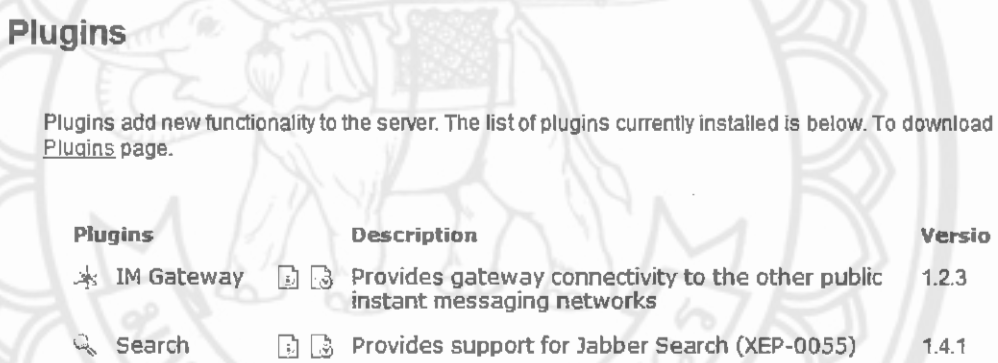
รูปที่ ง.2 การ Install IM Gateway

3. เมื่อ Install เสร็จแล้วก็เข้าไปตรวจสอบดูว่าติดตั้ง IM Gateway Plugin ไปแล้วหรือไม่โดยการคลิ๊กเข้าไปที่ Plugins ดังรูปภาพ ง.3



รูปที่ ง.3 แสดงการเข้าไปตรวจสอบดูว่า ติดตั้ง IM Gateway ไปแล้วหรือไม่

4. เมื่อเข้ามาในหน้า Plugins ก็จะเห็น Plugin ที่ชื่อว่า IM Gateway เพิ่มเข้ามาใหม่



รูปที่ ง.4 แสดง IM Gateway ที่ได้ติดตั้งเพิ่มเข้ามา