สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การพัฒนาโปรแกรมประยุกต์เชิงวัตถุโดยใช้ภาษาจาวา

Object Oriented Application using Java

จัดทำโดย

นาย บุญช่วย ชื่นชม

นาย อดิศักดิ์ พงศ์ทิพากร

อาจารย์ที่ปรึกษา

ดร. วรวัฒน์ ลิ้มโภคา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2542

เลขหมู่.....
เลขทะเบียน 37076
วัน, เดือน, ปี 3 0 ธ.ค. 2543

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2542

ภาควิชา วิศวกรรมคอมพิวเตอร์

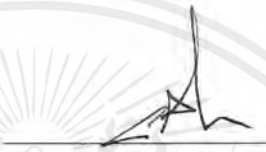
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การพัฒนาโปรแกรมประยุกต์เชิงวัตถุโดยใช้ภาษาจาวา

Object Oriented Application using Java

ผู้จัดทำ

1. นาย บุญช่วย ชื่นชม รหัสประจำตัว 40013255
2. นาย อดิศักดิ์ พงศ์ทิพากร รหัสประจำตัว 40013277



อาจารย์ที่ปรึกษา

(ดร. วรวัฒน์ ลิ้มโภคา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การพัฒนาโปรแกรมประยุกต์เชิงวัตถุโดยใช้ภาษาจาวา

นาย บุญช่วย ชื่นชม 40013255
นาย อดิศักดิ์ พงศ์ทิพากร 40013277
ดร. วรวัฒน์ ลิ้มโภคา อาจารย์ที่ปรึกษา
ปีการศึกษา 2542

บทคัดย่อ

เนื่องจากในปัจจุบันคอมพิวเตอร์ได้เข้ามามีบทบาทสำคัญกับชีวิตประจำวันของมนุษย์มากขึ้น และรูปแบบในการพัฒนาซอฟท์แวร์ในปัจจุบันก็มีความแตกต่างจากเมื่อก่อนมาก ซึ่งการพัฒนาซอฟท์แวร์ในปัจจุบันนี้มีลักษณะเป็นแบบเชิงวัตถุซึ่งทำให้การบำรุงรักษาและแก้ไขซอฟท์แวร์ทำได้ง่าย และภาษาที่ใช้ในการพัฒนาซอฟท์แวร์เชิงวัตถุก็มีให้เลือกใช้หลายภาษา และภาษาจาวาก็เป็นภาษาหนึ่งซึ่งมีคุณสมบัติโดดเด่นกว่าภาษาอื่นคือความสามารถในการใช้งานข้ามแพล็ตฟอร์มได้

ปริญญานิพนธ์นี้เป็นการทดลองใช้รูปแบบการพัฒนาแอบพลิเคชั่นเชิงวัตถุโดยใช้ภาษาจาวาโดยเลือกพัฒนาระบบเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์ ซึ่งผู้ใช้สามารถเรียกใช้งานผ่านเวบเบราเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Object Oriented Application using Java

Bhoonchoy Chuenchom
Adisak pongtipakorn
Dr. Worawat Limpoka Advisor

ABSTRACT

Now a day, Computers were more important parts of people and software development process was more different from previously. At times, software development is used object-oriented concept which easy to maintained and modified. It have more languages that support. JAVA is one of these language which distinguished in working over platforms

The thesis was tried to develop object-oriented application by JAVA language which implement to web server and mail server system. User can run application via web browser

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดีหากไม่ได้รับคำปรึกษาชี้แนะจากอาจารย์ วรวัฒน์ ลิ้มโภคา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก และต้องขอขอบคุณคุณพ่อคุณแม่พี่ๆและเพื่อนๆที่คอยให้กำลังใจ ให้คำปรึกษา และเสนอแนะแนวคิดในการแก้ปัญหาต่างที่เกิดขึ้นในการทำวิทยานิพนธ์ครั้งนี้ตลอดมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้าที่
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญตาราง	VI
สารบัญภาพ	VII
บทที่ 1 บทนำ	1
1.1 ความสำคัญและที่มา	1
1.2 วัตถุประสงค์ของงานวิจัย	1
1.3 ขอบเขตของงานวิจัย	2
1.4 วิธีการดำเนินงาน	2
บทที่ 2 ทฤษฎีและหลักการ	3
2.1 คุณสมบัติของออบเจ็กต์โอเรียนท์เต็ด	3
2.1.1 การแยกแยะเอกลักษณ์ (Abstraction)	3
2.1.2 การซ่อนรายละเอียด (Encapsulation)	3
2.1.3 การสืบทอดคุณสมบัติ (Inheritance)	3
2.1.4 โพลิมอปริซึม (Polymorphism)	4
2.2 คุณสมบัติของจาวา	4
2.3 ทฤษฎีเบื้องต้นเกี่ยวกับเว็บเซิร์ฟเวอร์	5
2.3.1 รูปแบบการเรียกติดต่อจากเบราเซอร์ไปยังเว็บเซิร์ฟเวอร์	5
2.3.2 การทำงานของ HTTP	6
2.3.2.1 วิธีการร้องขอจากเว็บเบราเซอร์ไปยังเว็บเซิร์ฟเวอร์ (Request Method)	7
2.3.3 การเข้ารหัสข้อมูลก่อนส่ง	10
2.3.4 เลือกใช้วิธี GET หรือ POST ดี	11
2.3.5 วิธีการตอบสนองจากเว็บเซิร์ฟเวอร์ไปยังเว็บเบราเซอร์	12
2.3.6 รหัสแสดงสถานะ (Status Code)	13
2.3.7 รูปแบบของเอกสาร HTML	16
2.3.7.1 Static HTML	16
2.3.7.2 Dynamic HTML	17

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.8 ทำความรู้จักกับซีจีไอ (CGI) 18
2.4 ทฤษฎีเบื้องต้นเกี่ยวกับเมล์เซิร์ฟเวอร์ 18
2.4.1 แบบจำลองของ SMTP 20
2.4.2 การเข้ารหัสและไมน์ (MINE) 23
2.5 ทฤษฎีเบื้องต้นเกี่ยวกับดีเอ็นเอส (DNS) 25
2.5.1 ข้อมูลที่เก็บใน DNS 26
2.6 ทฤษฎีเบื้องต้นเกี่ยวกับออราเคิล (ORACLE) 28
2.6.1 Java Database connectivity และการติดต่อกับฐานข้อมูล 28

บทที่ 3 ทฤษฎีและการออกแบบ 34
3.1 การทดลองการติดต่อระหว่างไคลเอนต์กับเซิร์ฟเวอร์ 35
3.1.1 ทดลองโพรโตคอล HTTP ด้วยเทลเน็ท (Telnet) 35
3.1.2 การทดลองการร้องขอจากไคลเอนต์ไปยังเซิร์ฟเวอร์ด้วยเมธอด GET 36
3.1.3 ทดลองโปรโตคอล SMTP ด้วยเทลเน็ท 37
3.2 ออกแบบเว็บเซิร์ฟเวอร์และเมล์เซิร์เวอร์โดยใช้ยูเอ็มเอล (UML) 39
3.2.1 ส่วนของคลาสและไฟล์ซีจีไอที่ถูกเรียกใช้ 42
3.3 ตารางที่ใช้เก็บข้อมูล 55

บทที่ 4 การทำงานของโปรแกรม 56

บทที่ 5 บทวิจารณ์และสรุป 64
5.1 บทสรุปและวิจารณ์ 64
5.2 สิ่งที่ต้องปรับปรุงแก้ไขและพัฒนาต่อ 64
ภาคผนวก ก 66
ภาคผนวก ข 67
บรรณานุกรม 68

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตารางที่		หน้าที่
1.	ตารางที่2-1 ขั้นตอนการติดต่อระหว่างเว็บเบราเซอร์และเว็บเซิร์ฟเวอร์	7
2.	ตารางที่ 2-2 แสดงการเข้ารหัส URL-encoded	11
3.	ตารางที่ 2-3 แสดงวิธีเลือกใช้เมธอด GET กับ POST	12
4.	ตารางที่ 2-4 รหัสแสดงสถานะ	15
5.	ตารางที่ 2-5 แสดงรายละเอียดในคำสั่งต่างๆของ SMTP ที่ใช้งาน	21
6.	ตารางที่ 2-6 แสดงตารางการเข้ารหัสแบบ base64	24
7.	ตารางที่ 2-7 แสดงรายละเอียดของข้อมูลในฟิลด์ Type	27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ

รูปที่	หน้าที่
1. รูปที่ 2-1 แสดงการขั้นตอนการเชื่อมต่อระหว่างไคลเอนต์กับเซิร์ฟเวอร์	6
2. รูปที่ 2-2 แสดง Static HTML	16
3. รูปที่ 2-3 แสดง dynamic HTML	17
4. รูปที่ 2-4 แสดง dynamic HTML (มีการติดต่อกับฐานข้อมูล)	17
5. รูปที่ 2-5 แสดงการรับและส่งไปรษณีย์อิเล็กทรอนิกส์	19
6. รูปที่ 2-6 แสดงโมเดลของ SMTP	21
7. รูปที่2-7 แสดงแผนภูมิแบบลำดับชั้นอย่างง่ายของโดเมนเนมในอินเตอร์เน็ต	25
8. รูปที่ 2-8 แสดงฟิลด์ต่างๆของ resource record ใน DNS	26
9. รูปที่ 2-9 แสดงระดับการเชื่อมต่อของ JDBC API	29
10. รูปที่ 3-1 แสดงการป้อนข้อมูลสำหรับการเชื่อมต่อ	35
11. รูปที่ 3-2 แสดงการกำหนดค่าต่างๆในโปรแกรม Telnet	36
12. รูปที่ 3-3 แสดงเฮดเดอร์ HTTP ของข้อความตอบสนอง	37
13. รูปที่ 3-4 แสดงการใช้โปรแกรมเทลเน็ทในการส่งเมล์	38
14. รูปที่ 3-5 แสดง Use Case Diagramของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์	39
15. รูปที่ 3-6 แสดง Object Diagramของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์	40
16. รูปที่ 3-7 แสดง Class Diagramของเว็บเซิร์ฟเวอร์	41
17. รูปที่ 3-8 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์(ให้บริการ static html)	44
18. รูปที่ 3-9 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์ (ให้บริการ dynamic html)	45
19. รูปที่ 3-10 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์ในการส่งเมล์	46
20. รูปที่ 3-11 แสดง Class Diagramของเมล์เซิร์ฟเวอร์	47
21. รูปที่ 3-12 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์ (กรณีเริ่มทำงานของเมล์เซิร์ฟเวอร์)	50
22. รูปที่ 3-13 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์ (กรณีที่มีการตรวจเช็คอีเมล์แอดเดรส)	51
23. รูปที่ 3-14 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์ (กรณีที่ข้อความมีการเข้ารหัสแบบQuoted Printable)	52
24. รูปที่ 3-15 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์ (กรณีที่ข้อความของเมล์มีการเข้ารหัสแบบ base64)	53

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

25. รูปที่ 3-16 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์
(กรณีที่อีเมล์แอดเดรสของผู้รับเป็นของเซิร์ฟเวอร์ตัวอื่น) 54
26. รูปที่ 4-1 แสดงหน้าจอหลัก 56
27. รูปที่ 4-2 แสดงหน้าจอแบบฟอร์มการลงทะเบียน 57
28. รูปที่ 4-3 แสดงหน้าจอการล็อกอิน 57
29. รูปที่ 4-4 แสดงหน้าจอการใส่รหัสผ่านผิดพลาด 58
30. รูปที่ 4-5 แสดงหน้าจอการเข้ามาอ่านจดหมาย 58
31. รูปที่ 4-6 แสดงหน้าจอการเข้ามาดูจดหมายทั้งหมด 59
32. รูปที่ 4-7 แสดงหน้าจอการลบจดหมาย 59
33. รูปที่ 4-8 แสดงหน้าจอการเข้ามาดูจดหมายทั้งหมด 60
34. รูปที่ 4-9 แสดงหน้าจอการเข้ามาอ่านจดหมาย 60
35. รูปที่ 4-10 แสดงหน้าจอการตอบจดหมาย 61
36. รูปที่ 4-11 แสดงหน้าจอการตอบจดหมาย 61
37. รูปที่ 4-12 แสดงหน้าจอการเข้ามาอ่านจดหมาย 62
38. รูปที่ 4-13 แสดงหน้าจอการเลือกอัปโหลดไฟล์ 62
39. รูปที่ 4-14 แสดงหน้าจอการอัปโหลดไฟล์ 63

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ในปัจจุบันนี้อินเทอร์เน็ตได้เข้ามามีบทบาทเป็นอย่างมากในชีวิตประจำวัน ไม่ว่าจะในสถานศึกษาหรือในบริษัทต่างๆ ผู้ที่เคยเล่นอินเทอร์เน็ตคงจะคุ้นเคยกับคำว่าเวิลด์ไวด์เว็บ (World Wild Web) ซึ่งแต่เดิมเราสามารถแสดงได้เฉพาะส่วนที่เป็นข้อความล้วนๆ โดยใช้ภาษา HTML (Hyper Text Markup Language) ในการสร้างไฟล์เอกสารเหล่านี้ ต่อมาจึงได้มีการพัฒนาให้สามารถนำเสนอภาพนิ่งภาพเคลื่อนไหวและลูกเล่นต่างๆ ทำให้เว็บมีความน่าสนใจเพิ่มขึ้น และทำให้เกิดความรวดเร็วในการติดต่อสื่อสาร ดังจะเห็นได้จากในอดีตเราติดต่อสื่อสารกันโดยใช้จดหมายซึ่งทำให้เกิดความล่าช้า และสิ้นเปลืองค่าใช้จ่ายสูง แต่หลังจากอินเตอร์เน็ตได้เข้ามามีบทบาทในชีวิตประจำวันเพิ่มขึ้นทำให้เราสามารถติดต่อสื่อสารกันโดยใช้ไปรษณีอิเล็กทรอนิกส์ (Electronic Mail) ส่งข้อความถึงกันได้ทั่วโลก โดยใช้เวลาไม่นานก็ถึงปลายทาง ซึ่งการรับส่งจดหมายโดยใช้ ไปรษณีอิเล็กทรอนิกส์นี้ได้รับความนิยมเป็นอย่างมาก และมีเว็บไซต์หลายเว็บไซต์ที่ให้บริการฟรีอีเมล์ให้เราเลือกใช้บริการ เช่น http://www.thaimail.com และ http://www.hotmail.com เป็นต้น ซึ่งผู้ใช้แต่ละคนจะมีอีเมล์แอดเดรสเป็นของตัวเอง นั่นคือเราต้องทำการลงทะเบียนเพื่อขออีเมล์แอดเดรสก่อน จากนั้นจึงจะสามารถใช้บริการต่างๆที่จัดเตรียมไว้ให้ได้ เช่น สามารถส่งจดหมายไปหาเพื่อนที่อยู่ต่างประเทศ หรือ เรียกดูจดหมายที่มีผู้อื่นส่งมาให้เราได้ รวมไปถึงบริการฟรีโฮมเพจที่เราสามารถทำการอัฟโหลดไฟล์เอกสาร HTML จากเครื่องเราไปไว้ที่เครื่องเซิร์ฟเวอร์ทำให้สามารถเรียกดูโฮมเพจที่เราสร้างขึ้นจากเครื่องเซิร์ฟเวอร์ได้ ซึ่งฟรีอีเมล์และฟรีโฮมเพจที่เป็นของคนไทยเองนั้นมีจำนวนน้อยมาก จากจุดนี้เองจึงทำให้เกิดความสนใจที่จะศึกษาการทำงานในส่วนของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์ว่ามีการทำงานอย่างไร มีการติดต่อสื่อสารกันได้อย่างไร โดยได้เลือกใช้ภาษาจาวา (JAVA) ในการพัฒนาโปรแกรมเพราะภาษาจาวามีข้อดีต่างๆดังที่จะได้กล่าวถึงต่อไป

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 เพื่อศึกษาการออกแบบและการเขียนโปรแกรมในรูปแบบของออบเจ็กต์โอเรียนท์เต็ดโดยภาษาที่เลือกใช้ในการพัฒนาคือภาษาจาวา

1.2.2 เพื่อศึกษาการติดต่อสื่อสารระหว่างเครื่องไคลเอนต์กับเครื่องเซิร์ฟเวอร์ว่ามีการติดต่อสื่อสารข้อมูลกันอย่างไร

1.2.3 เพื่อศึกษาการทำงานของเว็บเซิร์ฟเวอร์ว่าเมื่อมีเครื่องไคลเอนต์ร้องขอบริการเข้ามาเครื่องเซิร์ฟเวอร์จะมีการตอบสนองอย่างไรบ้าง

1.2.4 เพื่อศึกษาการทำงานของเมล์เซิร์ฟเวอร์ว่ามีการรับและการส่งเมล์นั้นต้องมีขั้นตอนการติดต่อสื่อสารอย่างไร

1.2.5 เพื่อศึกษาการนำข้อมูลเก็บลงในดาต้าเบสและการเรียกข้อมูลออกมาจากดาต้าเบส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.3 ขอบเขตของงานวิจัย

ศึกษาในส่วนของเว็บเซิร์ฟเวอร์ว่ามีการให้บริการเอกสาร HTML ในรูปแบบ Static html และ Dynamic html แก่เครื่องไคลเอนต์ได้อย่างไร และในส่วนของเมล์เซิร์ฟเวอร์ว่ามีการรับเมล์ที่ส่งมาจากภายนอกได้อย่างไร และถ้าเราจะส่งเมล์ออกไปให้เพื่อนที่อยู่ต่างประเทศจะสามารถทำได้อย่างไร

1.4 วิธีการดำเนินงาน

งานวิจัยในโครงงานนี้จะเริ่มด้วยการศึกษาทฤษฎีพื้นฐานต่าง ๆ ที่เกี่ยวข้องกับงานวิจัย ซึ่งก็มีเรื่องหลัก ๆอยู่ 4 ส่วนคือ

- ส่วนของเว็บเซิร์ฟเวอร์
- ส่วนของเมล์เซิร์ฟเวอร์
- ส่วนที่ใช้ติดต่อกับผู้ใช้ ซึ่งสามารถติดต่อผ่านทางเว็บเบราเซอร์
- ส่วนที่ใช้ติดต่อกับฐานข้อมูล

ซึ่งในส่วนของเนื้อหาต่างๆที่ได้ทำการศึกษาจะมีดังต่อไปนี้

บทที่สองอธิบายถึงทฤษฎีและหลักการต่างๆของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์

บทที่สามอธิบายถึงทฤษฎีและการออกแบบโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการ

2.1 คุณสมบัติของออบเจ็กต์โอเรียนท์เต็ด

แนวความคิดของออบเจ็กต์นั้นเกิดขึ้นครั้งแรกเมื่อประมาณ 20 ปีที่ผ่านมา ออบเจ็กต์โอเรียนท์เต็ดเกิดขึ้นจากแนวความคิดที่จะออกแบบซอฟแวร์ ให้มีลักษณะเลียนแบบวัตถุต่างๆ ในชีวิตจริงคือ มองส่วนประกอบทุกอย่างในโปรแกรมเป็นวัตถุ นอกจากนั้นยังมีการเลียนแบบพฤติกรรม และคุณสมบัติของวัตถุด้วย ซึ่งการเขียนโปรแกรมแบบออบเจ็กต์นั้นจะมีความใกล้ชิดหรือการขึ้นต่อกันน้อย (loose couple) ทำให้การดูแลและแก้ไขส่วนต่างๆของระบบสามารถทำได้ง่ายและมีผลกระทบต่อส่วนอื่นๆของระบบน้อยที่สุด

2.1.1 การแยกแยะเอกลักษณ์ (Abstraction)

การแยกแยะเอกลักษณ์เป็นวิธีการหนึ่งที่มนุษย์ใช้ในการมองสิ่งใดๆเพื่อลดความซับซ้อนของสิ่งนั้นๆ โดยแยกเอาเฉพาะสิ่งที่ผู้มองสนใจออกมา ซึ่งแน่นอนว่าถ้าผู้มองระบบเป็นคนละคนกันก็จะแยกแยะออกมาได้ต่างกัน การแยกแยะเอกลักษณ์เป็นการแสดงคุณลักษณะที่สำคัญของออบเจ็กต์ ซึ่งเป็นคุณลักษณะที่ทำให้ออบเจ็กต์นั้นๆแตกต่างจากออบเจ็กต์อื่นๆ เช่น ออบเจ็กต์รถยนต์ ถ้าเป็นการแยกแยะเอกลักษณ์ของนักเดินทางก็จะมองว่ารถยนต์นั้นต้องพาผู้โดยสารไปยังปลายทางได้สะดวกรวดเร็ว แต่ถ้าเป็นการแยกแยะเอกลักษณ์ของช่างเครื่องก็จะมองว่ารถยนต์ประกอบด้วยเครื่องยนต์ต่างๆมากมายที่ทำงานร่วมกัน

2.1.2 การซ่อนรายละเอียด (Encapsulation)

ถ้าพิจารณาจากศัพท์แล้วคงแปลได้ว่า การนำมารวมกันโดยมีเปลือกห่อหุ้ม ซึ่งความหมายที่แท้จริงของการซ่อนรายละเอียด ก็คือ การรวมกันของโครงสร้างข้อมูล (Data Structure) กับฟังก์ชันที่เกี่ยวข้องหรือเรียกใช้ข้อมูลนั้น (เรียกว่า Methods หรือ Actions) เกิดเป็นวัตถุตัวใหม่ที่สามารถทำการซ่อนข้อมูลจากภายนอกได้ ทำให้ข้อมูลไม่ถูกเข้าถึง หรือเปลี่ยนแปลงจากภายนอก การห่อหุ้มข้อมูล (data encapsulation) หรือการซ่อนข้อมูล (data hiding) นี้เป็นจุดเด่นของการเขียนโปรแกรมเชิงวัตถุ (เหมือนเป็นยาเม็ดแคปซูลที่เราทานแก้หวัดกัน ซึ่งจะมองไม่เห็นยาภายในรู้แต่ว่ายาเม็ดนี้รักษาโรคอะไรเท่านั้น)

2.1.3 การสืบทอดคุณสมบัติ (Inheritance)

การสืบทอดคุณสมบัติเป็นความสัมพันธ์แบบ "is - a " เป็นการกำหนดความสัมพันธ์คลาสและถือว่าเป็นคุณสมบัติหนึ่งที่สำคัญที่สุดของออบเจ็กต์โอเรียนเต็ด การสืบทอดคุณสมบัติทำให้คลาสสามารถแชร์โครงสร้างและเมธอดระหว่างคลาสได้ โดยเราสามารถเขียนคลาสที่มีการสืบทอดคุณสมบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้เป็นลำดับขั้นได้โดยสับคลาสจะสืบทอดคุณสมบัติทุกอย่างมาจากซูเปอร์คลาส ตัวอย่างเช่น หมีเป็นชนิดหนึ่งของสัตว์เลี้ยงลูกด้วยนม หมีก็จะมีคุณสมบัติและ Behavior เหมือนกับสัตว์เลี้ยงลูกด้วยนมหรือบ้านเป็นชนิดหนึ่งของที่อยู่อาศัย บ้านก็จะมีคุณสมบัติและ Behavior เหมือนกับที่อยู่อาศัยเป็นต้น

2.1.4 โพลิมอปริซึม (Polymorphism)

คำว่า Polymorphism แปลว่ารูปแบบหลายๆ รูปแบบ ในทางการออกแบบโปรแกรมแบบออบเจ็กต์โอเรียนท์เต็ดจะให้ความหมายของคำว่า โพลีมอร์ฟิซึม คือ ความสามารถที่แต่ละออบเจ็กต์อาจใช้ชื่อเหมือนกันได้แต่จะใช้ตัวพารามิเตอร์ที่ส่งมาด้วย เป็นตัวบอกว่าจะใช้ได้กับเมื่อใดทำให้มีคุณสมบัติ ที่เมื่อออบเจ็กต์ต่างๆ ได้รับคำสั่งเดียวกันจากโปรแกรมแล้วแต่ละออบเจ็กต์จะทำงานตามแบบของตัวเอง ซึ่งทำให้ได้ผลลัพธ์แตกต่างกันซึ่งเป็นคลาสที่ได้รับการถ่ายทอดคุณสมบัติจากคลาสเดียวกัน การที่คลาสในกลุ่มความสัมพันธ์นี้ จะทำงานด้วยวิธีที่ต่างกันด้วยเมสเสจเดียวกันประโยชน์ของ Polymorphism คือสามารถจัดการกับอินซแตน (Instance) ต่างคลาสกันด้วยวิธีเดียวกัน เป็นการสนับสนุนการออกแบบโปรแกรมเพื่อให้ใช้งานร่วมกันหรือการนำกลับมาใช้ได้อีก

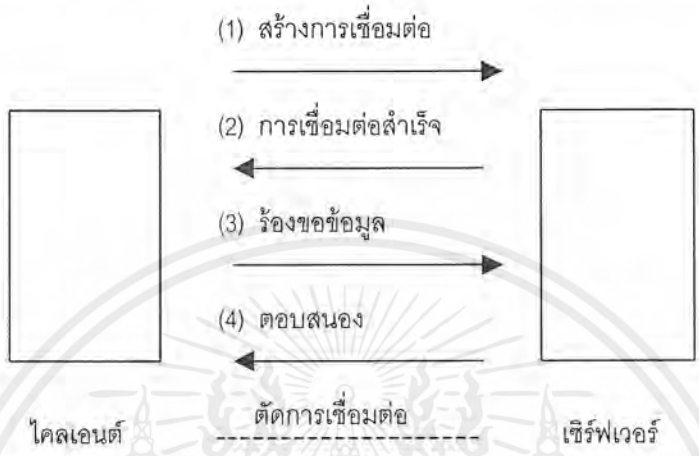
2.2 คุณสมบัติของจาวา

- เป็นออบเจ็กต์โอเรียนท์เต็ด โดยที่ออบเจ็กต์โอเรียนท์เต็ดเกิดขึ้นจากแนวความคิดที่จะออกแบบซอฟแวร์ ให้มีลักษณะเลียนแบบวัตถุต่างๆ ในชีวิตจริงคือ มองส่วนประกอบทุกอย่างในโปรแกรมเป็นวัตถุ นอกจากนั้นยังมีการเลียนแบบพฤติกรรม และคุณสมบัติของวัตถุด้วย ซึ่งการออกแบบโปรแกรมโดยใช้หลักการออบเจ็กต์โอเรียนท์เต็ททำให้เราการเขียนโปรแกรมได้ง่าย
- เป็นภาษาที่ไม่ขึ้นกับแพลตฟอร์มคือโปรแกรมที่เขียนด้วยภาษาจาวาสามารถนำรันบนเครื่องที่ต่างแพลตฟอร์มได้ ดังคำกล่าวที่ว่า "write one run anywhere"
- เป็นภาษาที่ง่าย (simple) หมายถึงมีกลไกของภาษาจำนวนไม่มาก และไม่ซับซ้อน เพื่อให้ผู้ใช้เรียนรู้ได้อย่างรวดเร็ว ภาษาจาวานำไวยากรณ์ส่วนใหญ่มาจากภาษา C++ แต่ก็กลไกของภาษา C++ ออกไปหลายอย่าง โดยไปเพิ่มความสามารถให้คอมไพเลอร์จัดการให้แทน เช่นภาษาจาวาไม่มี preprocessor commands , function prototype และ header files
- มีความคงทน (robust) โปรแกรมที่ถูกสร้างขึ้นด้วยภาษาจาวาจะต้องมีความผิดพลาดที่ไม่คาดคิดมาก่อนน้อยที่สุดและโปรแกรมจะต้องไม่ล้มเหลวง่ายๆ โดยที่ภาษาจาวาเป็นภาษาที่เน้นการใช้กลไก exception handling ซึ่งช่วยให้โปรแกรมจัดการกับความผิดปกติบางอย่างที่เกิดขึ้นขณะโปรแกรมทำงาน โดยที่โปรแกรมไม่หยุดลง
- ภาษาจาวาตัดการอ้างถึงแอดเดรสของตัวแปรและการใช้พอยท์เตอร์ในการอ่านหรือเขียนค่าหน่วยความจำโดยตรงออก เพื่อให้โปรแกรมไม่มีโอกาสใช้พอยท์เตอร์อ้างถึงหน่วยความจำที่โปรแกรมไม่มีสิทธิ์อ้างถึงนอกจากนี้ภาษาจาวายังกำหนดให้ ผู้โปรแกรมไม่สามารถคืนหน่วยความจำขณะโปรแกรมทำงาน แต่อาศัย automatic garbage collector ทำหน้าที่เก็บหน่วยความจำที่ไม่อาจอ้างถึงได้แล้ว และนำกลับมาใช้งานอีกโดยอัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.2 การทำงานของ HTTP

โปรโตคอล HTTP เป็นโปรโตคอลแบบไคลเอนต์และเซิร์ฟเวอร์ (client/server) ดังนั้นการติดต่อสื่อสารใดๆ ผ่านโปรโตคอลนี้จำเป็นต้องมีเครื่องคอมพิวเตอร์ตัวลูกกับตัวแม่ การสื่อสารจึงจะสมบูรณ์ได้ เรามาดูว่าการติดต่อกันระหว่างไคลเอนต์ไปยังเซิร์ฟเวอร์ผ่านโปรโตคอล HTTP (เช่นการเปิดเว็บเพจผ่านเว็บเบราเซอร์) จะมีขั้นตอนอย่างไรบ้าง



รูปที่ 2-1 แสดงการขั้นตอนการเชื่อมต่อระหว่างไคลเอนต์กับเซิร์ฟเวอร์

จากรูปขั้นแรกคือเครื่องไคลเอนต์ (ในที่นี้คือเว็บเบราเซอร์) จะสร้างการเชื่อมต่อ (connection) กับเครื่องเซิร์ฟเวอร์ผ่านสิ่งที่เรียกว่า "ซ็อกเก็ต" (socket) เมื่อซ็อกเก็ตทั้งสองฝั่งเชื่อมต่อกันได้สำเร็จแล้ว เครื่องไคลเอนต์จะส่งคำร้องขอข้อมูล (request) ไปยังเครื่องเซิร์ฟเวอร์ จากนั้นเซิร์ฟเวอร์จะไปหาข้อมูลที่ไคลเอนต์ต้องการ ซึ่งไม่ว่าจะมีหรือไม่มีข้อมูลตามที่ไคลเอนต์ร้องขอ เซิร์ฟเวอร์ก็จะต้องส่งข้อมูลตอบสนอง (response) กลับมายังไคลเอนต์เสมอ สุดท้ายการเชื่อมต่อจะถูกตัดขาดหรือปลดการเชื่อมต่อของซ็อกเก็ตทั้งสองฝั่งออก ด้วยการทำงานของโปรโตคอล HTTP ที่มีการเชื่อมต่อในระยะเวลาเพียงสั้นๆ หรือที่เรียกกันว่าเป็นโปรโตคอลแบบ connectionless ทำให้ในช่วงเวลาหนึ่งเซิร์ฟเวอร์ที่ให้บริการ www สามารถรองรับไคลเอนต์ได้จำนวนมากพร้อมๆกัน เพราะไม่มีใครได้ทำการเชื่อมต่ออย่างถาวร เช่น สมมุติว่าไฟล์โฮมเพจของเซิร์ฟเวอร์ kmitl.ac.th ถูกกำหนดไว้ให้เป็นชื่อ index.html และมีเนื้อหาในไฟล์ดังต่อไปนี้

```
<html>
<head><title> kmitl.ac.th</title></head>
<body background="linethai.gif">
<img src="logo.gif">
</body>
</html>
```

ถ้าเราสั่งให้เว็บเบราเซอร์ติดต่อไปยัง kmitl.ac.th เว็บเบราเซอร์จะต้องค้นหาเซิร์ฟเวอร์ kmitl.ac.th ให้ได้เสียก่อน เพื่อทำการเชื่อมต่อ เมื่อเชื่อมต่อได้แล้ว เว็บเบราเซอร์จะส่งข้อความร้องขอไปยังเซิร์ฟเวอร์ แต่เนื่องจากเราไม่ได้บอกว่าเป็นไฟล์ชื่ออะไร (เพราะระบุเพียงแค่ kmitl.ac.th) เมื่อเซิร์ฟเวอร์รับทราบการร้องขอแล้วพบว่าไม่มีการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บอกระบุชื่อไฟล์ ดังนั้นเว็บเบราเซอร์จะส่งไฟล์ที่เป็นไฟล์โฮมเพจกลับมาให้แทน ซึ่งก็คือไฟล์ index.html เมื่อเว็บเบราเซอร์ได้รับไฟล์มาแล้ว เซิร์ฟเวอร์จะตัดการเชื่อมต่อทันที

จากนั้นเมื่อเว็บเบราเซอร์เริ่มอ่านและตีความหมายของแท็กในไฟล์ index.html จึงพบว่ารูปพื้นแบ็กกราวด์ของเว็บเพจระบุให้ใช้ไฟล์ linethai.gif ดังนั้นเว็บเบราเซอร์ต้องสร้างการเชื่อมต่อไปยัง kmitl.ac.th เป็นครั้งที่สอง เพื่อขอไฟล์ linethai.gif จากเซิร์ฟเวอร์อีก เมื่อเซิร์ฟเวอร์ส่งไฟล์ linethai.gif ให้เว็บเบราเซอร์แล้ว การเชื่อมต่อก็ถูกตัดขาดลงอีกครั้งหนึ่ง จากนั้นเมื่อเว็บเบราเซอร์อ่านพบในบรรทัดต่อมาว่าต้องแสดงรูปที่ชื่อ logo.gif ในเว็บเพจ เว็บเบราเซอร์ก็ต้องสร้างการเชื่อมต่อเพื่อขอข้อมูลอีก การทำงานของเว็บเบราเซอร์จะเป็นอย่างนี้ไปเรื่อยๆ ดังนั้นการเปิดเว็บเพจหนึ่งๆ จึงอาจมีการเชื่อมต่อกับเซิร์ฟเวอร์เพื่อขอข้อมูลหลายครั้ง

เว็บเบราเซอร์	ทิศทาง	เว็บเซิร์ฟเวอร์
(1) ผู้ใช้พิมพ์ค่า URL แล้วกด Enter หรือคลิกที่ไฮเปอร์ลิงค์เช่น http://www.ce.kmitl.ac.th (2) เว็บเบราเซอร์จะสร้าง Request Header ส่งไปยังเว็บเซิร์ฟเวอร์	(3) Request Header ถูกส่งไปยังเว็บเซิร์ฟเวอร์ →	(4) ตรวจสอบความต้องการของเบราเซอร์จาก Request Header ว่าสามารถตอบสนองให้ได้หรือไม่
(7) เว็บเบราเซอร์อ่านค่าจาก Res - ponseHeader และทำการแสดงผลเว็บเพจที่ได้รับมา	(6) ResponseHeader และข้อมูลที่ต้องการถูกส่งกลับไปยังเว็บเบราเซอร์ ←	(5) สร้าง ResponseHeader ส่งกลับไปยังเว็บเบราเซอร์หากมีข้อมูลใดจะส่งกลับไปด้วยก็จะส่งต่อท้าย ResponseHeader ไป

ตารางที่ 2-1 ขั้นตอนการติดต่อระหว่างเว็บเบราเซอร์และเว็บเซิร์ฟเวอร์

2.3.2.1 วิธีการร้องขอจากเว็บเบราเซอร์ไปยังเว็บเซิร์ฟเวอร์ (Request Method)

ใน HTTP/1.0 และ 1.1 นั้น เราสามารถกำหนด Request Method ได้หลายวิธี ได้แก่ Get, Head, Post , Put , Delete , Link , และ Unlink แต่โดยปกติเราจะใช้อยู่ 2 วิธีคือ Get และ Post

- **Get**

กรณีที่เว็บเบราเซอร์ขอเปิดเอกสาร HTML จากเซิร์ฟเวอร์จะร้องขอแบบ GET โดยเมื่อเว็บเบราเซอร์สร้างซ็อกเก็ตเชื่อมต่อกับเซิร์ฟเวอร์ได้แล้วจะสร้างข้อความร้องขอขึ้นมาเพื่อส่งไปยังเซิร์ฟเวอร์ ข้อความในบรรทัดแรกสุดของเฮดเดอร์ HTTP จะประกอบไปด้วย 3 ส่วนคือ วิธีการร้องขอ หรือที่เรียกว่าเมธอด (method) , ไดเรกทอรีกับชื่อไฟล์ที่ร้องขอ และเวอร์ชั่นของ HTTP ที่ผู้ร้องขอใช้อยู่แต่ละส่วนของข้อความจะถูกคั่นด้วยช่วงว่าง (space) ฉะนั้นรูปแบบในการเขียนข้อความบรรทัดแรกของเฮดเดอร์ HTTP จึงเป็นดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

GET /path/file HTTP/x.x

- ***GET*** บอกว่าใช้เมธอด GET ในการร้องขอข้อมูลจากเซิร์ฟเวอร์ (ต้องเขียนเป็นตัวพิมพ์ใหญ่หมด)
- ***/path/file*** ไดเรกทอรีและชื่อไฟล์ที่ต้องการจากเซิร์ฟเวอร์
- ***HTTP/x.x*** *x.x* คือเวอร์ชั่นของ HTTP (คำว่า HTTP เขียนด้วยตัวพิมพ์ใหญ่หมด) บอกแก่เซิร์ฟเวอร์ให้รู้ว่าทางไคลเอนต์ใช้ HTTP เวอร์ชั่นไหนอยู่

ตัวอย่างเช่น ถ้าเราป้อน URL ให้กับเว็บเบราเซอร์เป็น http://venus09.ce.kmitl.ac.th/news/index.html เว็บเบราเซอร์ต้องสร้างการเชื่อมต่อไปยัง venus09.ce.kmitl.ac.th ก่อน แล้วจึงสร้างข้อความร้องขอส่งไปยังเซิร์ฟเวอร์ ดังนั้นโครงสร้างของข้อความร้องขอจะเป็นดังนี้

GET /news/index.html HTTP/1.0

User-Agent:Mozilla/4.03(en) (win95;I)

Accept:image/gif;image/jpg,*/*

Accept-Language:en

Accept-Charset:ISO-8859-1,*,utf-8

Cache-Control:max-age=259200

เว้นหนึ่งบรรทัด

ความหมายจากเฮดเดอร์ HTTP จะได้ว่า เว็บเบราเซอร์ร้องขอเปิดไฟล์ index.html ในไดเรกทอรี่ /news และเว็บเบราเซอร์ใช้โปรโตคอล HTTP เวอร์ชัน 1.0 จะเห็นว่านอกจากคำร้องขอที่อยู่ในบรรทัดแรกแล้ว ยังมีข้อมูลอื่นๆที่เว็บเบราเซอร์ส่งไปให้เซิร์ฟเวอร์ด้วย ข้อมูลในส่วนที่สองนี้เรียกว่าเฮดเดอร์ ซึ่งข้อมูลในส่วนเฮดเดอร์ของเว็บเบราเซอร์แต่ละโปรแกรมแต่ละเวอร์ชันอาจไม่เหมือนกัน

- **Post**

สำหรับการร้องขอด้วยเมธอด POST จะใช้ในกรณีที่ต้องการส่งข้อมูลจากไคลเอนต์ไปยังเซิร์ฟเวอร์ คือไม่ได้ขอไฟล์จากเซิร์ฟเวอร์ แต่จะส่งข้อมูลไปให้เซิร์ฟเวอร์รับไปทำงาน เมื่อเซิร์ฟเวอร์รับข้อความร้องขอแบบ POST แล้วก็จะทราบว่าไคลเอนต์ต้องการส่งข้อมูลมาให้ รูปแบบข้อความร้องขอในบรรทัดแรกจะเขียนคล้ายกับเมธอด GET ดังนี้

POST /path/file HTTP/1.0

การร้องขอด้วยเมธอด POST จะมีเงื่อนไขต่อไปนี้

เอกสารนี้ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ข้อมูลที่ส่งไปให้เซิร์ฟเวอร์จะอยู่ภายในบล็อกข้อมูล ดังนั้นจึงต้องมีเฮดเดอร์เพื่อบอกรายละเอียดของข้อมูลในบล็อกข้อมูลแนบไปด้วย เช่น ข้อมูลเป็นแบบไหน (Content-Type:) และข้อมูลมีกี่ไบต์ (Content-Length:) เสมอ
- /path/file คือชื่อโปรแกรม CGI ในเซิร์ฟเวอร์ที่จะทำหน้าที่รับข้อมูลไปประมวลผล
- ข้อความตอบสนองที่เซิร์ฟเวอร์จะส่งกลับให้ไคลเอนต์ จะได้มาจากการทำงานของโปรแกรมหรือ CGI ในเซิร์ฟเวอร์ ดังนั้น CGI ที่รับข้อมูลไปจึงต้องทำหน้าที่ส่งข้อความตอบกลับไปให้ไคลเอนต์

โดยส่วนใหญ่แล้ว การส่งข้อมูลจากฟอร์มในเว็บเพจไปประมวลผลด้วย CGI ในเซิร์ฟเวอร์ จะใช้เมธอด POST มากที่สุด ความจริงแล้วการส่งข้อมูลไปยังเซิร์ฟเวอร์สามารถใช้เมธอด GET ก็ได้ แต่มีข้อจำกัดเรื่องความยาวของข้อมูลที่จะส่งไปให้เซิร์ฟเวอร์ ถ้าใช้เมธอด POST เพื่อร้องขอส่งข้อมูลไปยังเซิร์ฟเวอร์ เฮดเดอร์ที่ชื่อ Content-Type จะถูกกำหนดให้เป็น application/x-www-form-urlencoded เพื่อบอกแก่เซิร์ฟเวอร์ว่าข้อมูลที่ส่งไปให้มีการเข้ารหัส และเฮดเดอร์ Content Length จะใช้สำหรับบอกความยาวของข้อมูลที่เข้ารหัสแล้ว เพราะข้อมูลที่กรอก ผ่านอินพุตในฟอร์มจะถูกเว็บเบราเซอร์เข้ารหัสก่อนส่งเสมอ การเข้ารหัสนี้เรียกว่า URL-encode ซึ่งมีรายละเอียดดังนี้

- แปลงตัวอักษรหรือเครื่องหมายบางตัวให้อยู่ในรูป %xx โดยที่ xx จะเป็นค่ารหัสแอสกีของอักษรตัวนั้น ตัวอักษรที่ต้องมีการแปลง เช่น = , & , % และ + เพราะเป็นเครื่องหมายที่ใช้เป็นตัวแบ่งแยกข้อมูลที่จะส่งไปให้เซิร์ฟเวอร์
- เปลี่ยนช่องว่าง (space) ทุกตัวเป็นเครื่องหมายบวก (+)
- รวมชื่อตัวแปรและค่าตัวแปรเข้าด้วยกัน โดยคั่นกลางด้วยเครื่องหมาย = และคั่นระหว่างตัวแปรด้วยเครื่องหมาย & เช่น number=152-401099&from=Adisak&message=kmitl+computer

ตัวอย่างข้อความร้องขอด้วยเมธอด POST ที่เว็บเบราเซอร์สร้างขึ้นเพื่อส่งข้อมูลจากฟอร์มในเว็บเพจไปให้แก่ CGI ที่ชื่อ test.cgi ในไดเรกทอรี /cgi-bin ของเซิร์ฟเวอร์ http://203.146.1.214

```
POST /cgi-bin/test.cgi HTTP/1.0
-----------------------------------------------------------------
.....
Content-Type: application/x-www-form-urlencoded
Content-Length:64
.....
-----------------------------------------------------------------
namefrom=Adisak&subject=test+mailserver&message=amazing+thailand
```

2.3.3 การเข้ารหัสข้อมูลก่อนส่ง

เราจะเห็นว่าในส่วนของข้อความ (message) ที่เรากรอกในฟอร์มคือ amazing thailand แต่ข้อความในบล็อกข้อมูลที่เว็บเบราเซอร์ส่งไปให้เซิร์ฟเวอร์จะถูกเข้ารหัสเป็น amazing+thailand นั่นคือช่อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ว่างในข้อความจะถูกเปลี่ยนเป็นเครื่องหมาย + หมดทุกตัว ที่ต้องทำการเข้ารหัสก่อน เราสามารถอธิบายด้วยการส่งด้วยเมธอด GET ได้ดังนี้ ในการส่งข้อมูลจากฟอร์มด้วยเมธอด GET นั้นเราทราบว่าเว็บเบราเซอร์จะเอาข้อมูลจากฟอร์มไปต่อท้ายชื่อไดเรกทอรีและชื่อไฟล์ CGI ที่ร้องขอบริการ โดยที่รูปแบบของโปรโตคอล HTTP ได้กำหนดไว้ว่า ในบรรทัดแรกของข้อความร้องขอจะประกอบด้วย 3 ส่วนคือ เมธอด , URI และเวอร์ชันของ HTTP คั่นแต่ละส่วนออกจากกันด้วยช่องว่าง ดังนั้นหาก URI มีช่องว่างปนมาด้วยจะทำให้เว็บเซิร์ฟเวอร์ที่รับข้อความร้องขอ ตีความหมายของข้อมูลที่รับมาผิดได้

ตัวอย่างเช่น สมมุติว่าภายในฟอร์มมีอินพุตรับข้อมูลตัวเดียวที่ชื่อ key เมื่อคลิกปุ่ม send จะส่งข้อมูลจากฟอร์มด้วยเมธอด GET ไปให้ CGI ที่ชื่อ test.cgi ในไดเรกทอรี /cgi-bin ที่อยู่ในเซิร์ฟเวอร์ venus09.ce.kmitl.ac.th รับไปทำงาน ซึ่งซอร์ซโค้ด HTML ในส่วนของฟอร์มมีลักษณะดังต่อไปนี้

```
<FORM  ACTION= "http://venus09.ce.kmitl.ac.th/cgi-bin/test.cgi " METHOD="GET" >
<INPUT TYPE="text" NAME = "key" SIZE="20" MAXLENGTH="40">
<INPUT TYPE="submit" VALUE = "Send">
<INPUT TYPE="reset" VALUE = "Clear">
</FORM>
```

สมมุติว่าผู้ใช้ป้อนคำว่า "amazing thailand" ให้แก่อินพุต key แล้วคลิกปุ่ม Send เพื่อส่งข้อมูล ถ้าเว็บเบราเซอร์ไม่มีการเข้ารหัสข้อมูลก่อนส่งบรรทัดแรกสุดของข้อความร้องขอ จะได้ดังรูป

```
GET  /cgi-bin/test.cgi?key=amazing thailand  HTTP/1.0
--------------------------------------------------------------------------
User-Agent:Mozilla/4.03(en) (Win95;I)
Accept:image/gif,Image/jpeg,*/*
.............
.............
--------------------------------------------------------------------------
```

เว้นหนึ่งบรรทัด

ซึ่งเมื่อทางเว็บเซิร์ฟเวอร์ได้รับข้อความร้องขอนี้ จะต้องทำการแยกเมธอด , URI และเวอร์ชันของ HTTP จากข้อความร้องขอบรรทัดแรก โดยจะได้แยกได้ดังนี้

เมธอด	GET
URI	/cgi-bin/test.cgi&key=amazing
เวอร์ชัน HTTP	thailand

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นว่าเกิดความผิดพลาดขึ้น ซึ่งเริ่มตั้งแต่การแยก URI โดยได้ URI ไม่ครบตามที่คาดไว้ เพราะมีที่ว่างหลังคำว่า thailand ก็กลายเป็นเวอร์ชันของ HTTP

ดังนั้นจึงต้องมีการเข้ารหัสข้อมูลในฟอร์มก่อนที่จะส่งทุกครั้ง เพื่อกำจัดช่องว่างและอักขระพิเศษต่างๆ ที่ผู้ใช้อาจป้อนเข้ามา โดยจะแปลงให้อยู่ในรูปของ %xx เมื่อ xx คือเลขฐานสิบหกที่ใช้แทนอักขระนั้น การเข้ารหัสนี้เรียกว่า URL-encode ตามตารางนี้

อักขระ	รหัส %xx
%	%25
&	%26
'	%27
+	%2B
=	%3D
?	%3F

ตารางที่ 2-2 แสดงการเข้ารหัส URL-encoded

ตัวอย่างเช่น ถ้ามีการป้อนข้อความว่า "Amazing + Thailand = ?" เว็บเบราเซอร์จะ ต้องเข้ารหัสข้อความให้กลายเป็น "Amazing+%2B+Thailand+%3D+%3F" เสียก่อน และไม่ว่าจะเป็นการส่งข้อมูลด้วยเมธอด GET หรือ POST ก็จะต้องมีการเข้ารหัสข้อความเสมอ ความยาวของข้อมูลหลังจากการเข้ารหัสแล้วอาจจะมากกว่า หรือเท่ากับข้อมูลต้นฉบับในฟอร์มก็ได้ และเมื่อทาง CGI รับข้อมูลไปแล้วก็ต้องถอดรหัสข้อความกลับเสียก่อนที่จะนำข้อความนั้นไปใช้งานต่อไป

2.3.4 เลือกใช้วิธี GET หรือ POST ดี

ไม่ว่าการส่งข้อมูลจากฟอร์มไปให้ CGI จะกระทำด้วยเมธอด GET หรือ POST ก็ตาม CGI ก็สามารถอ่านไปใช้งานได้เหมือนกัน หากเราทราบว่า CGI รอรับข้อมูลแบบ GET เราไม่จำเป็นต้องกรอกข้อมูลในฟอร์มแล้วคลิกปุ่มเพื่อส่งข้อมูล แต่สามารถป้อนข้อมูลที่ต้องการส่งต่อท้าย URL ได้เลยส่วนกรณีที่ CGI รอรับข้อมูลแบบ POST ทางเลือกคือ ต้องส่งข้อมูลผ่านฟอร์มของเว็บเบราเซอร์หรือไม่ก็ต้องเขียน CGI ส่งข้อมูลของเราไปให้แก่ CGI ในเซิร์ฟเวอร์

GET	POST
1. ส่งข้อมูลด้วยการป้อนต่อท้าย URL โดยไม่ต้องใช้ฟอร์ม หรืออาจเขียน CGI เพื่อส่งข้อมูลก็ได้	1. ส่งข้อมูลไปยังเซิร์ฟเวอร์ด้วยการนำข้อมูลไปใส่ลงในบล็อกของข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. สามารถทำ bookmark หรือทำลิงก์ให้กับข้อมูลที่ส่งได้ ทำให้ส่งข้อมูลได้หลายครั้ง โดยไม่ต้องป้อนข้อมูลทุกครั้ง	2. ไม่สามารถทำ bookmark หรือลิงก์ให้กับข้อมูลที่ส่ง
3. เว็บเซิร์ฟเวอร์สามารถเก็บบันทึกข้อมูลที่ส่งเข้ามาไว้ใน log file ได้จึงเหมาะสำหรับข้อมูลที่ไม่เป็นความลับ	3. ข้อมูลที่ส่งเข้ายังเซิร์ฟเวอร์จะไม่นำไปบันทึกลงใน log file ของเว็บเซิร์ฟเวอร์ จึงเหมาะสำหรับข้อมูลที่ต้องการให้เป็นความลับ
4. มีข้อจำกัดเรื่องความยาวของข้อมูลที่จะส่ง คือได้เพียงครั้งละ 256 ตัวอักษร(นับจากที่เข้ารหัสแล้ว) ความยาวนี้ขึ้นอยู่กับระบบปฏิบัติการอีกทีหนึ่ง	4. ไม่มีข้อจำกัดเรื่องความยาวของข้อมูลที่จะส่ง จึงสามารถใช้ส่งไฟล์ขนาดใหญ่ๆ ได้
5. ข้อมูลที่ไคลเอนต์ส่งมาให้เซิร์ฟเวอร์จะถูกเก็บไว้ในตัวแปรแวดล้อมชื่อ QUERY_STRING ถ้า CGI อยากได้ข้อมูล ก็ต้องไปอ่านจากตัวแปรนี้	5. ถ้า CGI อยากได้ข้อมูลที่ไคลเอนต์ส่งมาให้ ต้องไปอ่านจาก Standard Input(STDIN)

ตารางที่ 2-3 แสดงวิธีเลือกใช้เมธอด GET กับ POST

2.3.5 วิธีการตอบสนองจากเว็บเซิร์ฟเวอร์ไปยังเว็บเบราเซอร์ (HTTP Response Header)

หลังจากที่เซิร์ฟเวอร์ได้รับ Request Header แล้ว เซิร์ฟเวอร์จะทำการตรวจสอบสิ่งที่เบราเซอร์ต้องการ และจากนั้นจะสร้าง Response Header เพื่อตอบกลับไปยังเบราเซอร์ โดยหากเป็นวิธี Get จะเริ่มดูจากค่า URI

รูปแบบของ Response Header เป็นดังนี้

PROTOCOL/VERSION

Status_Code Status_Description

{ResponseHeaderField}

.....

{เว้นหนึ่งบรรทัดเพื่อบอกให้ทราบว่าจบส่วนของเฮดเดอร์}

เอกสารหรือไฟล์ที่ส่งไปยังเบราเซอร์

บรรทัดแรกจะเป็น status line ค่า status_code และ status_Description เป็นค่าที่ใช้บอกสถานะว่าสามารถตอบสนองความต้องการของเบราเซอร์ได้หรือไม่อย่างไร ถัดจาก status line ในบรรทัดแรก ก็จะเป็นส่วนของเฮดเดอร์ซึ่งจะมีรายละเอียดต่างๆ เกี่ยวกับเอกสารหรือข้อมูลที่ส่งกลับไปยังเบราเซอร์ ตัวอย่างของ Response Header เป็นดังนี้

HTTP/1.0 200 OK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Date:Thu,31 Dec 1998 23:30:00 GMT

Server:Apache/1.2.5

MIME-version:1.0

LastModified:Mon,7 Dec 1998 12:00:00 GMT

Content-type:text/html

Content-length:4567

{เว้นหนึ่งบรรทัดเพื่อบอกให้ทราบว่าจบส่วนของเฮดเดอร์แล้ว}

<HTML><HEAD><TITLE>.....

ความหมายของ Response Header ข้างต้นก็คือใช้ HTTP เวอร์ชัน 1.0 เซิร์ฟเวอร์สามารถตอบสนองสิ่งที่ต้องการได้ เฮดเดอร์นี้สร้างขึ้นเมื่อเวลา 23:30 น. วันพฤหัสบดีที่ 31 ธันวาคม ค.ศ. 1998 ตามเวลามาตรฐาน (Greenwich) โดยเซิร์ฟเวอร์ใช้โปรแกรมอะพาเช่ (Apache) เวอร์ชัน 1.2.5 เป็น httpd และ MIME เวอร์ชัน 1.0 เอกสารที่ส่งมาเป็นเอกสารแบบ text/html ซึ่งก็คือเอกสาร HTML ที่ใช้บนเว็บเพจนี่เอง ได้รับการแก้ไขครั้งสุดท้ายเมื่อวันเวลา 12:00 น. ของวันจันทร์ที่ 7 ธันวาคม ค.ศ. 1998 มีขนาด 4567 ไบต์ กรณีที่เป็นเอกสาร HTML ตามตัวอย่างข้างต้น ในบรรทัดล่างสุดและถัดไป ก็จะเป็นข้อมูลที่เป็นภาษา HTML ที่เราเขียนขึ้นนั่นเอง

รายละเอียดในส่วนของเฮดเดอร์ที่สำคัญมี 2 ส่วนคือ

- Content-Type : ใช้บอกว่าข้อมูลที่อยู่ในบล็อกข้อมูลเป็นข้อมูลประเภทไหนเช่น หากเป็นเอกสารแบบ HTML จะต้องระบุเป็น text/html ถ้าเป็นไฟล์รูปกราฟิกแบบ gif ก็ต้องระบุเป็น image/gif เป็นต้น แต่สำหรับเว็บเบราเซอร์รุ่นใหม่ๆแล้ว หากข้อมูลที่ได้รับไม่มีการระบุว่าเป็นประเภทไหนแล้ว เว็บเบราเซอร์จะถือว่าเป็นประเภท text/html เสมอ
- Content-Length : ใช้ระบุความยาวของข้อมูลในบล็อกข้อมูลมีหน่วยเป็นไบต์ (byte) เพื่อที่ผู้รับจะได้ทราบว่ามีข้อมูลส่งมาให้กี่ไบต์

สิ่งที่ควรทราบอีกอย่างคือ รายการเฮดเดอร์ในการร้องขอและการตอบสนองนั้น อาจมีจำนวนรายการและลำดับการเรียงไม่เหมือนกัน ขึ้นอยู่กับโปรแกรมที่ใช้ เช่น เบราเซอร์รุ่นต่างกันจะมีรายการเฮดเดอร์ที่แตกต่างกัน ในทำนองเดียวกัน โปรแกรมเว็บเซิร์ฟเวอร์ต่างโปรแกรมกัน ก็อาจมีรายการเฮดเดอร์ตอบสนองที่ไม่เหมือนกัน

2.3.6 รหัสแสดงสถานะ (Status Code)

เป็นรหัสที่เป็นตัวเลข 3 หลัก เซิร์ฟเวอร์ใช้ในการบอกให้เบราเซอร์ทราบถึงสถานะของความต้องการที่ร้องขอมาว่าสำเร็จหรือไม่ เพียงใด พร้อมทั้งเหตุผลรายละเอียดของ status code

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รหัส	ความหมาย
1XX Informational(1.1)	**เริ่มใช้งานใน HTTP 1.1**
100 Continue (1.1)	ใช้ในกรณีที่เบราเซอร์อยู่ในระหว่างส่ง Request แต่ยังส่งไม่หมด แต่เซิร์ฟเวอร์ต้องการให้ทราบว่าได้รับ Request แล้วให้ส่งส่วนที่เหลือต่อไป
101 Switching Protocol (1.1)	ใช้ร่วมกับเฮดเดอร์ Upgrade กรณีที่ต้องการเปลี่ยนไปใช้โปรโตคอลอื่นที่ความสามารถสูงกว่าเช่น HTTP/2.0 ซึ่งอาจจะมีในอนาคต
2XX Success	**การร้องขอปรสพความสำเร็จ**
200 OK	การร้องขอประสพความสำเร็จ เอกสารที่ต้องการถูกส่งมาด้วย
201 Created	เอกสารหรือสิ่งที่ต้องการถูกส่งขึ้นเรียบร้อยแล้ว
202 Accepted	ได้รับคำร้องขอเรียบร้อยแล้ว แต่สิ่งที่ต้องการยังไม่เรียบร้อย
203 Non-Authoritative(1.1)	การร้องขอปรสพความสำเร็จ
204 No Content	การร้องขอประสพความสำเร็จ แต่ไม่มีข้อมูลจะส่งกลับ
205 Reset Content(1.1)	เซิร์ฟเวอร์ได้รับข้อมูลเรียบร้อยแล้ว และบอกให้เบราเซอร์ลบข้อความที่กรอกในแบบฟอร์มเดิมออก เพื่อสะดวกในการกรอกข้อมูลถัดไป
206 Patial Content(1.1)	เซิร์ฟเวอร์ได้รับข้อมูลบางส่วนเรียบร้อยแล้ว
3XX Redirection	**มีการเปลี่ยนทิศทางหรือชี้ไปยังตำแหน่งอื่น**
300 Multiple Choice	ข้อมูลหรือเอกสารที่ต้องการมีอยู่ในหลายแห่ง
301 Moved Permanently	เอกสารที่ต้องการถูกย้ายไปที่อื่นอย่างถาวรแล้ว
302 Moved Temporarily	เอกสารที่ต้องการถูกย้ายไปชั่วคราว
303 See Other (1.1)	ใช้ในกรณีที่ต้องการบอกให้ทราบว่าสิ่งที่ต้องการอยู่ใน URI อื่น ซึ่งเบราเซอร์สามารถใช้ GET เพื่อไปเรียกดูเอกสารนั้นๆ ได้
304 Not Modified	กรณีที่เบราเซอร์ติดต่อโดยระบุ if-Modified-Sence ไว้ด้วย แต่เอกสารนั้นยังไม่มีการเปลี่นแปลงใดๆ เซิร์ฟเวอร์จะไม่มีการส่งเอกสารนั้นๆ มาให้
305 Use Proxy (1.1)	บอกให้เบราเซอร์ทราบว่าเอกสารที่ต้องการมีอยู่ใน Proxy ซึ่ง URL ของ Proxy จะกำหนดใน Location
4XX Client Error	**มีข้อผิดพลาดที่เกิดขึ้นจากเบราเซอร์**
400 Bad Request	เซิร์ฟเวอร์ไม่เข้าใจสิ่งที่ต้องการ
401 Unauthorized	ไม่ได้รับอนุญาตให้เข้าถึงเอกสารนั้นๆ
402 Payment Required	ยังไม่มีการใช้งาน แต่สำรองไว้ใช้ในอนาคต
403 Forbidden	มักใช้ในกรณีที่เซิร์ฟเวอร์ต้องการปฏิเสธการร้องขอ แต่ไม่เข้าข่ายอื่นๆ หรือไม่ต้องการบอกเหตุผล
404 Not Found	ไม่พบเอกสารที่ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รหัส	ความหมาย
405 Method Not Allowed(1.1)	Method ที่ใช้ ไม่ได้รับอนุญาต กรณีนี้เซิร์ฟเวอร์จะระบุ Allow เพื่อบอกให้ทราบว่าอนุญาตให้ใช้ Method ใดบ้าง
406 Not Acceptable(1.1)	ข้อมูลที่ต้องการเป็นข้อมูลที่เบราเซอร์ไม่สามารถเข้าใจได้ เนื่องจากไม่อยู่ในรายการ Accept ที่ระบุใน Request Header
407 Proxy Authentication (Unauthorized) Request(1.1)	เหมือนกับ 401 แต่ต้องได้รับอนุญาตจาก Proxy จะระบุเฮดเดอร์ Proxy-Authenticate ให้ทราบ ซึ่งเบราเซอร์สามารถจะส่ง Request ใหม่โดยระบุเฮดเดอร์ proxy-Authorization ด้วย
408 Request Timeout(1.1)	เบราเซอร์ไม่ส่ง Request ตามเวลาที่เซิร์ฟเวอร์รอได้
409 Conflict(1.1)	เบราเซอร์ส่งข้อมูลที่มีความหมายขัดแย้งกันเอง
410 Gone(1.1)	เอกสารที่ต้องการไม่ได้อยู่บนเซิร์ฟเวอร์แล้ว
411 Length Required(1.1)	เซิร์ฟเวอร์ต้องการให้ระบุ Content-Length ด้วย
412 Precondition Failed(1.1)	เงื่อนไขบางอย่างที่กำหนดใน Request Header ตกไป
413 Request Entity Too Large (1.1)	ข้อมูลที่ส่งมามีขนาดใหญ่เกินกว่าที่เซิร์ฟเวอร์จะรองรับได้
414 Request URI Too Long (1.1)	ค่า URL ที่ระบุ ยาวเกินไป
415 Unsupport Media Type (1.1)	ไม่รองรับการทำงานของ Media Type
5XX Server Error	**มีข้อผิดพลาดที่เกิดจากเซิร์ฟเวอร์**
500 Internal Server Error	มีข้อผิดพลาดที่เกิดจากการทำงานของเซิร์ฟเวอร์เอง
501 Not Implemented	เซิร์ฟเวอร์ไม่รองรับการทำงานที่ต้องการ
502 Bad Gateway	ในขณะทำหน้าที่เป็น Geteway หรือ Proxy ได้รับข้อมูลที่ผิดพลาดจากเซิร์ฟเวอร์ปลายทาง
503 Service Unavailable	เซิร์ฟเวอร์ไม่สามารถให้บริการในขณะนั้นได้ อาจเนื่องจากมีผู้ใช้บริการมากเกินความสามารถที่จะให้บริการได้ หรืออยู่ระหว่างการปรับปรุงเซิร์ฟเวอร์
504 Gateway Timeout(1.1)	ในขณะทำหน้าที่เป็น Gateway หรือ Proxy ไม่ได้รับข้อมูลตอบจากเซิร์ฟเวอร์ปลางทางในเวลาที่กำหนด
505 HTTP Version not Supported(1.1)	เซิร์ฟเวอร์ไม่รองรับการทำงานของ HTTP เวอร์ชั่นนั้นๆ

ตารางที่ 2-4 รหัสแสดงสถานะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความหมายหลักๆ ของรหัสแสดงสถานะ

1XX Informational(1.1)	ข้อความข่าวสารอย่างเดียว
2XX Success	การร้องขอประสพความสำเร็จ
3XX Redirection	มีการเปลี่ยนทิศทางหรือชี้ไปยังตำแหน่งอื่น
4XX Client Error	มีข้อผิดพลาดที่เกิดขึ้นจากเบราเซอร์
5XX Server Error	มีข้อผิดพลาดที่เกิดจากเซิร์ฟเวอร์

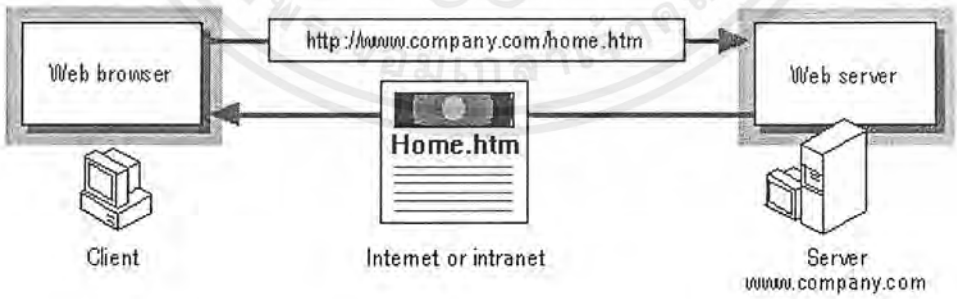
ถึงแม้ว่าเว็บเพจที่สร้างจะได้รับการออกแบบให้เป็นที่น่าดึงดูดความสนใจเต็มไปด้วยสีสัน รูปภาพ การสร้าง Link เชื่อมโยงข้อมูล แต่สิ่งหนึ่งที่จะทำให้เว็บเพจยิ่งน่าสนใจมากขึ้นอีกคือ การเปิดโอกาสให้มีการโต้ตอบจากผู้อ่านทั่วไปซึ่งทำได้โดยการสร้างแบบฟอร์ม ที่มีลักษณะเหมือนกับแบบสอบถามทั่วๆไป ลงในเว็บเพจเพื่อให้ผู้อ่านกรอกข้อมูลลงในแบบฟอร์มแล้วส่งกลับมายังเครื่องบริการที่เป็นเครื่องเซิร์ฟเวอร์ ในกรณีนี้จำเป็นต้องมีการเขียนโปรแกรมย่อยที่เรียกว่าสคริปท์ (Script) เก็บไว้ในเครื่องเซิร์ฟเวอร์ เพื่อทำหน้าที่รวบรวมข้อมูลที่ส่งมาจากแบบฟอร์ม และนำไปประมวลผลเก็บไว้ หรืออาจส่งผลลัพธ์กลับไปยังผู้อ่านอีกทีหนึ่ง โดยที่สคริปท์ ที่ถูกเขียนขึ้นนี้อาจสร้างจากภาษาใดภาษาหนึ่ง เช่น ภาษา c,perl ,java, pascal

2.3.7 รูปแบบของเอกสาร HTML

รูปแบบของเอกสาร HTML โดยทั่วไปจะมีอยู่ 2 ลักษณะคือ

2.3.7.1 Static HTML

จะเป็นเอกสาร HTML ที่ถูกเขียนขึ้นและเก็บอยู่ในเครื่องเซิร์ฟเวอร์ เมื่อมีการเรียกใช้งานจากผู้ใช้ผ่านทาง URL เครื่องเซิร์ฟเวอร์ก็จะส่งเอกสารนี้ไปให้กับผู้ใช้ ซึ่งเอกสารชนิดนี้จะถูกเขียนขึ้นโดยใช้ HTML tag ธรรมดา ไม่จำเป็นต้องมีการสร้างสคริปท์ใดๆ



รูปที่ 2-2 แสดง Static HTML

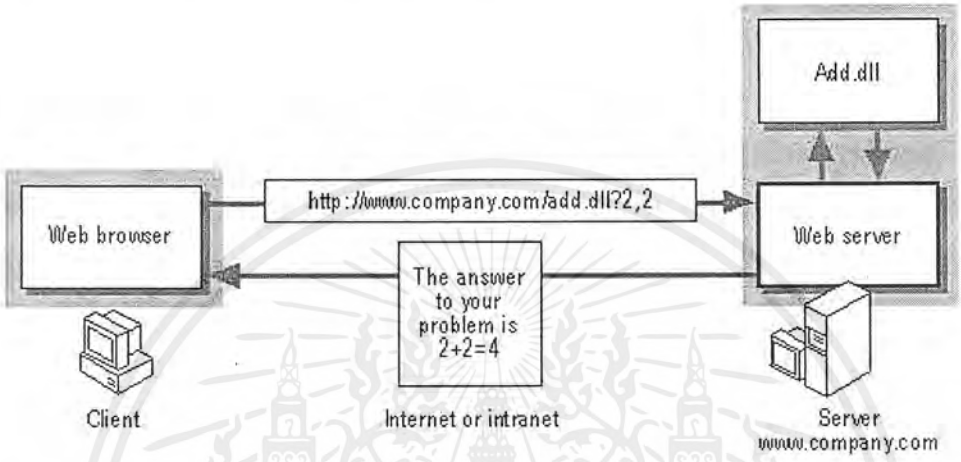
จากรูปเมื่อพิมพ์ http://www.company.com/home.htm การร้องขอ URL จะถูกส่งไปที่ server จากนั้น server จะตอบสนองโดยการส่ง static HTML page กลับมาที่ browser

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

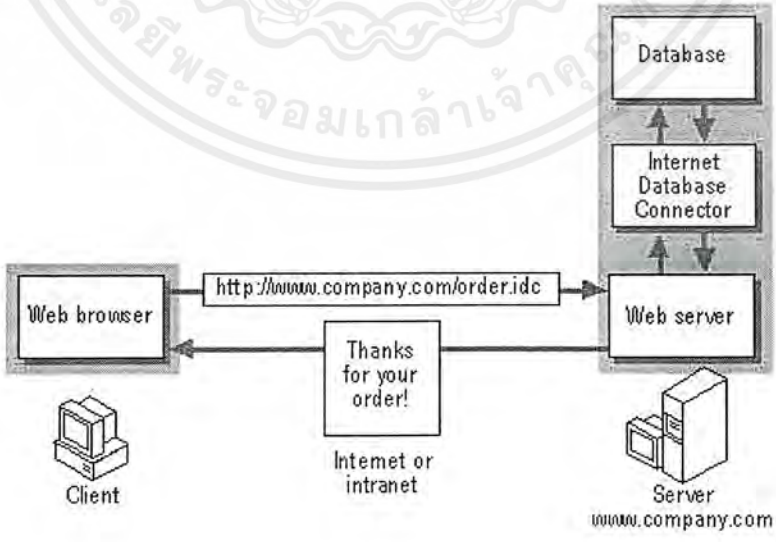
2.3.7.2 Dynamic HTML

สามารถเรียกได้อีกอย่างว่า " HTML on – the – fly " เอกสารประเภทนี้จำเป็นต้องมีการเขียน Script เก็บไว้ที่เครื่องเซิร์ฟเวอร์ เซิร์ฟเวอร์จะทำการประมวลผล Script นั้นและ Script จะทำการสร้างผลลัพธ์ที่เป็นเอกสาร HTML ที่มีรูปแบบของผลลัพธ์ขึ้นอยู่กับข้อมูลที่ผู้ใช้ป้อนจากแบบฟอร์ม จึงเรียกเอกสารประเภทนี้ว่า "Dynamic HTML" เพื่อให้เซิร์ฟเวอร์ส่งผลลัพธ์นั้นกลับไปยังผู้ใช้อีกที



รูปที่ 2-3 แสดง dynamic HTML

จากรูปที่ 2 เมื่อ user ป้อนจำนวนสองจำนวนที่ต้องการบวกกัน ในที่นี้คือ 2 กับ 2 ต่อจากนั้นคลิก OK จำนวน 2 จำนวนนั้นจะถูกส่งไปคำนวนที่ server โดยเรียกการทำงานไปที่ Add.dll จากนั้นผลลัพธ์คือ 4 จะถูกส่งกลับไปที่ HTML page ของ user



รูปที่ 2-4 แสดง dynamic HTML (มีการติดต่อกับฐานข้อมูล)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
37076

จากรูปที่ 2.4 แสดงการติดต่อกับฐานข้อมูล โดยใช้ Internet Database Connector เมื่อผู้ใช้กรอกข้อมูลในฟอร์มเสร็จแล้วคลิก "ตกลง" ข้อมูลจะถูกส่งไปที่เซิร์ฟเวอร์จากนั้น เครื่องเซิร์ฟเวอร์จะนำข้อมูลไปเก็บที่ดาต้าเบส ต่อจากนั้นจะส่ง HTML page กลับไปที่ผู้ใช้เพื่อเป็นการยืนยันความถูกต้อง

จะเห็นได้ว่าในส่วนของ Dynamic Html นั้นจะมีการทำงานในลักษณะ ไคลเอ็นต์/เซิร์ฟเวอร์ คือ มีการสร้างฟอร์ม HTML ในเว็บเพจเพื่อให้ผู้ใช้ดาวน์โหลดไปที่เครื่องไคลเอ็นต์เพื่อกรอกข้อมูลต่างๆ แล้วส่งกลับมาประมวลผลที่เครื่องเซิร์ฟเวอร์ได้

2.3.8 ทำความรู้จักกับซีจีไอ (CGI)

การส่งข้อมูลจากแบบฟอร์มไปยังเครื่องเซิร์ฟเวอร์ การประมวลข้อมูลที่เครื่องเซิร์ฟเวอร์ และการส่งข้อมูลกลับไปยังเบราเซอร์ของผู้ใช้ ถือว่าเป็นการสื่อสารแบบ 2 ทิศทาง (Two way communications) จำเป็นต้องมีโปรโตคอลช่วยควบคุมให้ทั้งสองฝ่ายสามารถสื่อสารกันได้อย่างเข้าใจ โดยที่โปรโตคอลที่ใช้ควบคุมการส่งข้อมูลระหว่างเว็บเบราเซอร์ และ เซิร์ฟเวอร์ และสคริปท์จะถูกเรียกว่า CGI (Common Gateway Interface)

CGI ถูกสร้างขึ้นมาเนื่องจากโปรโตคอลพื้นฐานของเว็บคือ HTTP ไม่เปิดโอกาสให้เว็บไคลเอนต์และเว็บเซิร์ฟเวอร์โต้ตอบกันในลักษณะ 2 ทิศทางได้ เว็บไคลเอนต์จะสามารถเรียกดูข้อมูลจากเว็บเซิร์ฟเวอร์ได้เพียงอย่างเดียว ดังนั้น CGI เป็นการเพิ่มเติมความสามารถของโปรโตคอล HTTP

ปัญหาที่มักพบในการสร้างโปรแกรมสำหรับการใช้ CGI คือ โปรแกรมที่สร้างขึ้นต้องการกำลังเครื่องมากในการทำงาน ทั้งนี้เพราะ CGI เหล่านั้นจะแตกตัวเป็นโพรเซสย่อยๆ ตามจำนวนไคลเอ็นต์ที่ติดต่อมายังเซิร์ฟเวอร์ ปัญหานี้ไม่ได้เป็นปัญหาใหม่ หากแต่ว่าในอดีตเหล่าเว็บเซิร์ฟเวอร์นั้นมักใช้ระบบปฏิบัติการยูนิกซ์ซึ่งเก่งในเรื่องจัดการโพรเซสและเธรดเป็นอย่างดี แต่ในปัจจุบันเรามีการใช้งานวินโดวส์เอ็นทีเป็นเว็บเซิร์ฟเวอร์มากขึ้น ทำให้นักพัฒนาต้องหาวิธีการใหม่ๆ มาช่วยเพื่อลดโพรเซสในการทำไดนามิกคอนเท็นต์ ไม่ว่าจะเป็น ISAPI, NSAPI ,ASP หรือ Fast-CGI

2.4 ทฤษฎีเบื้องต้นเกี่ยวกับเมล์เซิร์ฟเวอร์

อีเมล์ (E-mail) หรือจดหมายอิเล็กทรอนิคส์ได้เริ่มใช้งานกันมานานแล้ว ตั้งแต่ยุคของเครื่องเมนเฟรมหรือมินิคอมพิวเตอร์ ซึ่งไอบีเอ็มได้พัฒนาระบบอีเมล์ที่เรียกว่า PROFS(Professional Office System) ออกมาใช้งาน นอกจากนี้ก็มีบนระบบยูนิกส์ ต่อมาหลายๆค่ายก็ได้พัฒนาระบบอีเมล์ของตนขึ้นมา โดยส่วนใหญ่จะเป็นองค์ประกอบในแอพพลิเคชั่นที่ทำงานบนระบบเน็ตเวิร์ก เช่น ไมโครซอฟท์เมล์ ของไมโครซอฟท์ และ cc:Mail ของโลตัส เป็นต้น ซึ่งต่างก็ใช้เทคโนโลยีของตนเองและเป็นระบบปิด ดังนั้นการส่งเมล์ไปยังอีกผู้ใช้ที่ใช้ระบบเมล์คนละค่ายกันจึงเป็นเรื่องที่ยุ่งยาก

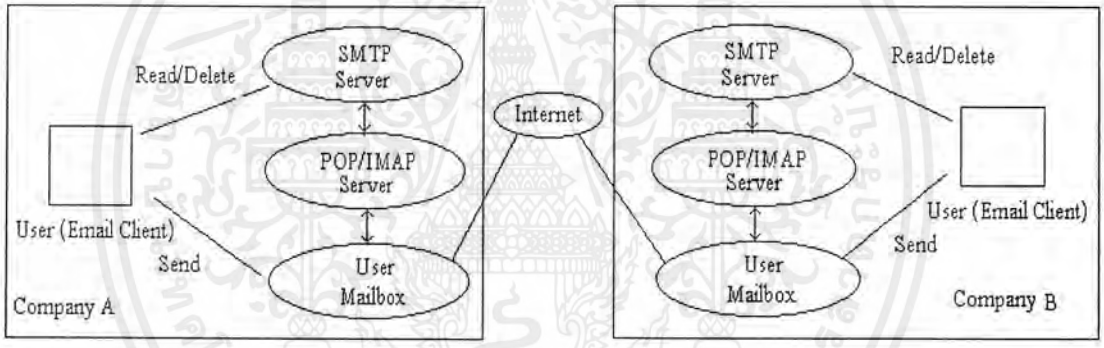
ในยุคต่อมาที่ระบบเน็ตเวิร์กทั้งแลน (Lan) และแวน (Wan) ต่างมีมาตรฐาน และเป็นระบบเปิด (Open System) มากขึ้น ก็ได้ปรับเปลี่ยนการทำงานของอีเมล์มาเป็นแบบไคลเอนต์เซิร์ฟเวอร์ที่เป็นพื้นฐานแบบที่ใช้กันในระบบยูนิกส์และมีการพัฒนาอีเมล์เซิร์ฟเวอร์ขึ้นมา โดยเฉพาะ เช่น เอ็กเช้งเซิร์ฟเวอร์ (Exchange Server) ของไมโครซอฟท์ หรือ โน้ตเซิร์ฟเวอร์ (Note Server) ของโลตัสเป็นต้น ซึ่งผู้ใช้จะติด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อเข้าสู่อีเมล์เซิร์ฟเวอร์ได้ทั้งโดยการผ่านระบบแลนหรือใช้โมเด็มเข้ามาจากแวนทำให้ผู้ใช้จะไม่เห็นไฟล์ในฮาร์ดดิสก็บนเซิร์ฟเวอร์เลย ดังนั้นความปลอดภัยของระบบจึงมีมากขึ้น

การรับและการส่งจดหมายอิเล็กทรอนิกส์ในเครือข่ายอินเทอร์เน็ตนั้น จะใช้โปรโตคอล SMTP (Simple Mail Transport Protocol) ซึ่งจะทำงานที่ Port 25 เป็นมาตรฐาน ซึ่งจะแสดงดังรูปที่ 2.5 ซึ่งแสดงถึงการรับส่งจดหมายอิเล็กทรอนิกส์ระหว่างสองหน่วยงาน แต่ละหน่วยงานจะมีป๊อบเซิร์ฟเวอร์ (POP Server ใช้ในการอ่านจดหมายอิเล็กทรอนิกส์) SMTP server (ใช้ในการรับและส่งจดหมายอิเล็กทรอนิกส์) ซึ่งอาจจะอยู่บนคอมพิวเตอร์เครื่องเดียวกันก็ได้ และจะมีส่วนเมล์บ็อกของผู้ใช้ (User Mailbox) ซึ่งเป็นพื้นที่ดิสก์ที่เก็บจดหมายอิเล็กทรอนิกส์ของผู้ใช้แต่ละคน

สมมุติว่า ผู้ใช้ในหน่วยงาน A ต้องการส่งจดหมายอิเล็กทรอนิกส์ ไปยังเพื่อนที่อยู่ในหน่วยงาน B ก็ต้องเรียกโปรแกรม E-mail client เช่น Netscape Mail,Eudora , Microsoft Outlook แต่งจดหมายให้เรียบร้อย จากนั้นเมื่อผู้ใช้กด Send mail โปรแกรมก็จะทำการส่งจดหมายฉบับนั้นไปยัง SMTP Server ของหน่วยงาน A ซึ่งต้องระบุไว้ที่ตัว E - mail Client ในหัวข้อ SMTP/Outgoing Mail Server นั่นเอง



รูปที่ 2-5 แสดงการรับและส่งไปรษณีย์อิเล็กทรอนิกส์

เมื่อเซิร์ฟเวอร์ SMTP ของหน่วยงาน A ได้รับจดหมายจากผู้ใช้ ก็จะทำการตรวจดูว่า ต้องการส่งไปถึงใคร และทำการตรวจสอบ DNS MX record ของโดเมนนั้นว่าระบุให้ไปที่เซิร์ฟเวอร์ SMTP ปลายทางตัวใด ในกรณีที่เซิร์ฟเวอร์ SMTP ของผู้ส่งไม่สามารถติดต่อเซิร์ฟเวอร์ SMTP ปลางทางได้ หรือติดต่อได้แต่ถูกปฏิเสธมา (เช่น User not found , quota exceed) จดหมายฉบับนั้นก็จะเด้งกลับไปยังผู้ส่ง โดยจะบอกถึงเหตุผลและปัญหาที่เกิดขึ้น

ลองพิจารณาถึงฝั่งรับกันบ้าง จากรูปที่ 2.5 เซิร์ฟเวอร์ SMTP ของหน่วยงาน A นอกจากจะทำหน้าที่ส่งต่อจดหมายจากผู้ใช้ภายในของหน่วยงาน A ไปยัง SMTP ปลายทางที่เหมาะสมแล้ว ยังต้องรับจดหมายจากเซิร์ฟเวอร์ SMTP ของหน่วยงานอื่นๆ ที่จ่าหน้ามาถึง ผู้ใช้ภายในหน่วยงาน A เองด้วย เมื่อมีการติดต่อเข้ามาจากเซิร์ฟเวอร์ SMTP อื่น เซิร์ฟเวอร์ SMTP ก็จะตรวจสอบจดหมายที่เข้ามาว่า จ่าหน้าถึงผู้ใด เป็นผู้ใช้ของหน่วยงานเราจริงหรือป่าว ตัวอย่างเช่น ในกรณีเครื่อง user.school.net.th ทำหน้าที่เป็น SMTP server ของโดเมน school.net.th ทั้งโดเมน (ต้องระบุใน configuration file) ก็ต้องดูว่าจดหมายที่เข้ามา ส่งมายัง <user>@school.net.th หรือป่าว ถ้าใช่ก็รับไว้ ถ้าไม่ใช่ก็ตอบปฏิเสธไป ว่าไม่มีผู้ใช้ชื่อนี้ เซิร์ฟเวอร์ SMTP ที่ส่งมาก็จะเด้งกลับจดหมายฉบับนั้นไปยังผู้ส่งต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.1 แบบจำลองของ SMTP

เมล์เซิร์ฟเวอร์เป็น SMTP เซิร์ฟเวอร์ที่ทำหน้าที่ในการรับส่งเมล์ ซึ่งจะทำการรับส่งเมล์ ผ่านพอร์ท 25 และใช้ RFC821 เป็นมาตราฐานในการติดต่อสื่อสารกันโดย SMTP ที่ออกแบบเป็นพื้นฐานในรูปแบบของการติดต่อกันระหว่าง SMTPผู้ส่ง กับ SMTPผู้รับ โดยก่อนที่เราจะกล่าวถึงโมเดลของ SMTP จะขอกล่าวถึงรูปแบบในการพูดคุยกันระหว่างSMTPผู้ส่ง กับ SMTPผู้รับ ก่อน โดยเมื่อเริ่มต้นการติดต่อผู้ส่งต้องส่งคำสั่ง "HELO" พร้อมกับรายละเอียดด้านผู้ส่งออกไป จากนั้นจะส่งคำสั่ง "MAIL" เพื่อแจ้งให้เซิร์ฟเวอร์เตรียมรับอีเมล์ ในส่วนของเซิร์ฟเวอร์เมื่อพร้อมที่จะรับอีเมล์ก็จะตอบรับกลับมาด้วยคำสั่ง "OK" จากนั้นที่ด้านส่งก็จะเริ่มส่งโดยใช้คำสั่ง "RCPT" เพื่อกำหนดอีเมล์แต่ละฉบับที่ส่งไป ซึ่งการส่งข้อมูลของอีเมล์จะถูกระบุด้วยคำสั่ง "DATA"

การส่งอีเมล์ของโปรโตคอล SMTP ได้จัดเตรียมคำสั่งอื่นๆไว้เพื่ออำนวนความสะดวกและคล่องตัวในการทำงาน ซึ่งประกอบด้วยคำสั่ง "VRFY" เพื่อให้ด้านที่ส่งตรวจสอบรายชื่อว่าผู้ใช้รายนี้มีสิทธิ์ใช้งานอีเมล์บ็อกซ์นั้นๆหรือไม่ , คำสั่ง "EXPN" ใช้จัดการและตรวจสอบรายชื่อจากลิสต์รายชื่อ และคำสั่ง "TURN" ใช้สลับให้ไคลเอนต์ของผู้ส่งทำหน้าที่รับข้อมูลจากเซิร์ฟเวอร์แทน

เมื่อได้รับคำสั่งต่างๆของผู้ส่งแล้ว เซิร์ฟเวอร์จะมีหน้าที่ตรวจสอบความถูกต้องของคำสั่ง จากนั้นจึงทำงานตามคำสั่งและส่งผลตอบกลับมา ส่วนลักษณะของข้อมูลที่ตอบกลับ (reply message) นั้นจะเป็นข้อมูลที่อยู่ในรูปของเท็กท์ (text) ที่เป็นรหัสแอสกีโดยจะประกอบด้วยตัวเลขนำหน้าข้อความ 3 หลัก ทำหน้าที่แสดงสถานะการทำงานของเซิร์ฟเวอร์ และเปลี่ยนสถานะการทำงานของโปรโตคอล SMTP ด้วย ถัดจากตัวเลขจะคั่นด้วยช่องว่างแล้วตามด้วยข้อความ ซึ่งปิดท้ายด้วยเครื่องหมายคาร์เรียรีเทอน (Carriage Return) และไลน์ฟีด (Line Feed) ตัวอย่างเช่น "500 Syntax error , command unrecognized"หมายถึงคำสั่งที่ส่งไปไม่ถูกต้องหรือ "503 Bad sequence of commands"หมายถึงลำดับการส่งคำสั่งไม่ถูกต้อง เป็นต้น

คำสั่ง	รายละเอียด
HELLO	ใช้เมื่อไคลเอนต์ของอีเมล์ต้องการเริ่มติดต่อกับเซิร์ฟเวอร์
MAIL	เริ่มเข้าสู่สถานะการส่งอีเมล์
RCPT	เป็นคำสั่งเพื่อระบุอีเมล์ที่จะส่งทีละฉบับ โดยเป็นคำสั่งที่ใช้ต่อจาก MAIL
DATA	จะเป็นคำสั่งที่ใช้ต่อจาก RCPT เพื่อส่งข้อมูลของอีเมล์
SEND	ทำหน้าที่เหมือนคำสั่ง DATA แต่ไม่ค่อยมีผู้ใช้งาน
SOML	ทำหน้าที่เหมือนคำสั่ง DATA แต่ไม่ค่อยมีผู้ใช้งาน
SAML	ทำหน้าที่เหมือนคำสั่ง DATA แต่ไม่ค่อยมีผู้ใช้งาน
VRFY	เป็นคำสั่งที่ใช้เพื่อตรวจสอบความถูกต้องของชื่อและเมล์บ็อกซ์
EXPN	เป็นคำสั่งเพื่อตรวจสอบรายละเอียดของสิสต์รายชื่อ
HELP	ใช้ตรวจสอบคำสั่งที่สามารถใช้งานได้กับเซิร์ฟเวอร์
NOOP	เป็นคำสั่ง No Operation เมื่อเซิร์ฟเวอร์ได้รับคำสั่งนี้ จะตอบ OK กลับมา
QUIT	สิ้นสุดการติดต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่ง	รายละเอียด
RSET	ยกเลิกการส่งข้อมูลในขณะนี้
TURN	เป็นคำสั่งที่สลับหน้าที่ของผู้ส่งข้อมูลมาทำหน้าที่รับข้อมูลแทน

ตารางที่ 2-5 แสดงรายละเอียดในคำสั่งต่างๆของ SMTP ที่ใช้งาน

รูปที่ 2-6 แสดงโมเดลของ SMTP

จากรูปที่ 2.6 เป็นรูปโมเดลของ SMTP ที่ประกอบไปด้วยส่วนต่างๆ 5 ส่วนคือ

1. ส่วนผู้ใช้จะทำการติดต่อกับเมล์เซิร์ฟเวอร์ที่ให้บริการส่งเมล์ (Sender-SMTP) แล้วทำการส่งข้อมูลตามมาตรฐานต่างๆให้ แก่เซิร์ฟเวอร์ได้แก่

- คำสั่ง MAIL<SP>FROM:<reverse-path><CRLF>

จะเป็นการบอกว่าผู้ส่งเมล์คือ reverse-path ซึ่งเป็น e-mail address ของผู้ส่ง เมื่อทำการส่งและได้รับการยอมรับจากเซิร์ฟเวอร์ ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 250 กลับมา เราก็จะทำการส่งคำสั่งต่อไปคือ

- คำสั่ง RCPT<SP>TO:<forward-path><CRLF>

จะเป็นการบอกว่าผู้ส่งต้องการส่งเมล์ให้กับ forward-path ซึ่งเป็นอีเมล์แอดเดรสของผู้รับเมื่อทำการส่งและได้รับการยอมรับจากเซิร์ฟเวอร์ ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 250 กลับมา เราก็จะทำการส่งคำสั่งต่อไป ซึ่งอาจเป็นคำสั่ง RCPT<SP>TO:<forward-path><CRLF> จะเป็นการบอกว่าผู้ส่งต้องการส่งเมล์ ให้กับ forward-path ซึ่งเป็นอีเมล์แอดเดรส ของผู้รับอีกคนหนึ่งเป็นการส่งเมล์ ให้กับผู้รับหลายคนแต่ข้อความใน mail เป็นข้อความเดียวกันเมื่อทำการส่งและได้รับการยอมรับจากเซิร์ฟเวอร์ ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 250 กลับมาเราก็ทำคำสั่งต่อไป

- คำสั่ง DATA<CRLF>

ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 250 กลับมา เราก็จะทำการส่งข้อความในเมล์ ไปให้แก่เซิร์ฟเวอร์ แล้วจบด้วย <CRLF>.<CRLF> ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 250 กลับมา เราก็จะทำการส่งคำสั่งต่อไป

- คำสั่ง QUIT<CRLF>

เป็นการบอกเลิกการติดต่อกับเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ส่วนเมล์เซิร์ฟเวอร์ที่ให้บริการส่งเมล์ ทำหน้าที่เป็นตัวส่ง SMTP แล้วทำการรับข้อมูลตามมาตรฐานต่างๆจากผู้ใช้ ตามที่ได้กล่าวมาแล้วในส่วนของผู้ใช้ทางพอร์ท 25 แล้วทำการส่งสตรีมให้กับ ผู้ใช้ตามมาตรฐาน RFC281 หากข้อมูลต่างๆ ถูกต้องเซิร์ฟเวอร์ ก็จะตรวจว่า forward-path เป็นของเซิร์ฟเวอร์เองหรือไม่ ถ้าใช่ก็จะทำการเก็บเมล์นั้นไว้ในไฟล์ซีสเต็มของเซิร์ฟเวอร์เองแต่ถ้าไม่ใช่ก็จะทำการส่งข้อมูลตามมาตรฐานต่างๆให้แก่เซิร์ฟเวอร์ของ forward-path อีกที่ ซึ่งมีขั้นตอนดังนี้

- ทำการติดต่อกับพอร์ท 25 ของเมล์เซิร์ฟเวอร์ของ forward-path (ผู้ส่งSMTP) เมื่อทำการส่งและได้รับการยอมรับจากเซิร์ฟเวอร์ ตัวเซิร์ฟเวอร์ก็จะส่งสตรีม 220 กลับมา เราก็จะทำการส่งคำสั่งต่อไปคือ

- คำสั่ง HELO<SP><domain>

จะเป็นการบอกโดเมนของผู้ส่ง SMTP เมื่อทำการส่งและได้รับการยอมรับจากผู้รับSMTP ตัวผู้รับSMTP ก็จะส่งสตรีม 250 กลับมา ผู้ส่งSMTP ก็จะทำการส่งคำสั่งต่อไปคือ

- คำสั่ง MAIL<SP>FROM:<reverse-path><CRLF>

จะเป็นการบอกว่าผู้ส่ง mail คือ reverse-path ซึ่งเป็นอีเมล์แอดเดรสของผู้ส่ง เมื่อทำการส่งและได้รับการยอมรับจากSMTPผู้รับ ตัวผู้รับSMTP ก็จะส่งสตรีม 250 กลับมา SMTPผู้ส่งก็จะทำการส่งคำสั่งต่อไปคือ

- คำสั่ง RCPT<SP>TO:<forward-path><CRLF>

จะเป็นการบอกว่าผู้ส่งต้องการส่งเมล์ให้กับ forward-path ซึ่งเป็นอีเมล์แอดเดรสของผู้รับเมื่อทำการส่งและได้รับการยอมรับจากSMTPผู้รับ ตัวSMTPผู้รับก็จะส่งสตรีม 250 กลับมา ก็จะทำการส่งคำสั่งต่อไป ซึ่งอาจเป็นคำสั่ง RCPT<SP>TO:<forward-path><CRLF> SMTPผู้ส่งจะเป็นการบอกว่าผู้ส่งต้องการส่งเมล์ให้กับ forward-path ซึ่งเป็นอีเมล์แอดเดรสของผู้รับในSMTPผู้รับ และได้รับการยอมรับจากSMTPผู้รับ ตัวSMTPผู้รับ ก็จะส่งสตรีม 250 กลับมา เราก็ทำคำสั่งต่อไป

- คำสั่ง DATA<CRLF>

ตัวSMTPผู้รับ ก็จะส่งสตรีม 250 กลับมา SMTPผู้ส่งก็จะทำการส่งข้อความในเมล์ไปให้ แล้วจบด้วย <CRLF>.<CRLF> ตัวSMTPผู้รับก็จะส่งสตรีม 250 กลับมา SMTPผู้ส่ง ก็จะทำการส่งคำสั่งต่อไป

- คำสั่งQUIT

เป็นการบอกเลิกการติดต่อกับSMTPผู้รับ

3. ส่วนเมล์เซิร์ฟเวอร์ในส่วนSMTPผู้รับจะทำการรับข้อมูลตามมาตรฐานต่างๆจากSMTPผู้ส่ง ตามที่ได้กล่าวมาแล้ว ในส่วนของผู้ใช้ที่ติดต่อทางพอร์ท 25 แล้วทำการเก็บเมล์ไว้ในไฟล์ซีสเต็มในฐานะเมล์ของผู้ใช้ ที่เป็นผู้ที่สิทธิในเมล์เซิร์ฟเวอร์นี้ แล้วทำการรอให้ ผู้ใช้มารับเมล์ของตัวเองอีกที่ และในส่วนที่ 4. และส่วนที่ 5. เป็นไฟล์ซิสเต็มของเมล์เซิร์ฟเวอร์ทำหน้าที่เก็บเมล์ของ ผู้ที่มีสิทธิของแต่ละเมล์เซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.2 การเข้ารหัสและไมน์ (MINE)

ในการรับส่งอีเมล์ผ่านเครือข่ายนั้น คอมพิวเตอร์ที่เชื่อมต่ออยู่ในเครือข่ายมักจะมีหลากหลายชนิด ดังนั้นข้อมูลที่ส่งผ่านจึงจะต้องเป็นข้อมูลที่อยู่ในรูปแบบกลางๆ ซึ่งคอมพิวเตอร์จะรับรู้และเข้าใจเหมือนกัน เพื่อไม่ให้ข้อมูลที่รับหรือส่งเหล่านั้นผิดเพี้ยนไปจากความเป็นจริง และสามารถส่งข้อมูลทั้งที่เป็นข้อความและไม่เป็นข้อความ (เช่น ข้อมูลที่เป็นรูปภาพและเสียง) รวมกันไปในอีเมล์ฉบับเดียวกันได้ ดังนั้นจึงได้นำเทคนิคการเข้ารหัสที่เรียกว่า MINE มาใช้เพื่อเข้ารหัสและถอดรหัสในการรับส่งอีเมล์โดยทั่วไป

ลักษณะข้อมูลของ MINE ประกอบด้วย 2 ส่วนคือ ส่วนหัว หรือเรียกว่าคอนเท็นทรานสเฟอร์เอ็นโค้ดดิ้ง (Content Transfer Encoding) ซึ่งจะเก็บรายละเอียดของไฟล์ที่เข้ารหัสไว้ เช่น ประเภทของไฟล์เป็นต้น ส่วนที่สองเป็นส่วนของข้อมูลที่เข้ารหัสแล้ว การเข้ารหัสและถอดรหัสของ MINE นี้จะถูกระบุไว้ในส่วนหัวเพื่อให้ผู้รับและผู้ส่งเข้าใจตรงกันว่าอีเมล์นี้เข้ารหัสและถอดรหัสด้วยวิธีใด ซึ่งมีอยู่ด้วยกัน 6 วิธี

Quoted – Printable

เทคนิคการเข้ารหัสวิธีนี้จะแปลงข้อมูลให้อยู่ในลักษณะที่อ่านได้เสมอ ซึ่งหากข้อมูลเป็นแอสกี้ 7 บิตอยู่แล้วก็จะไม่มีการแปลงข้อมูล แต่ถ้าเป็นค่าบิตที่ศูนย์มีค่าเป็น 1 ข้อมูลจะถูกแปลงให้มาอยู่ในรูปค่าของเลขฐาน 16 (01234567890ABCDEF) และนำหน้าด้วยเครื่องหมายเท่ากับ (=) ตัวอย่างเช่น ข้อมูลที่เข้ารหัสแล้วมีค่าเป็น =A1 หมายถึงข้อมูลที่ค่าแอสกี้เป็น 161 (10100001) ซึ่งตรงกับภาษาไทยคือ ‘ก’ หรือค่า Hex เป็น A1 นั่นเองและถ้ารหัสเป็น =A2 ก็จะหมายถึง (10100010) ซึ่งมีค่าเท่ากับ 162 ในเลขฐานสิบซึ่งตรงกับภาษาไทยคือ ‘ข’ เป็นต้น

Base64

เป็นเทคนิคการเข้ารหัสโดยจะแปลงข้อมูลจำนวน 24 บิต (ข้อมูล 8 บิตจำนวน 3 ไบต์) ออกเป็นข้อมูล 6 บิตจำนวน 4 ชุด โดยหลังจากที่เข้ารหัสแล้ว ข้อมูลจะถูกแปลงให้อยู่ในรูปของตัวอักษร 64 ตัว มีค่าตามตาราง Base64 Alphabet แต่ข้อมูลดังกล่าวจะไม่เปลี่ยนแปลงค่าของ Carriage Return และ Line Feed และปิดท้ายข้อมูลด้วยเครื่องหมาย = ซึ่งเรียกว่าแพด (PAD)

ตัวเลข	รหัส	ตัวเลข	รหัส	ตัวเลข	รหัส	ตัวเลข	รหัส
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	A	43	r	60	8
10	K	27	B	44	s	61	9
11	L	28	C	45	T	62	+
12	M	29	D	46	U	63	/
13	N	30	E	47	V		
14	O	31	F	48	W	(PAD)	=
15	P	32	G	49	X		
16	Q	33	H	50	Y		

ตารางที่ 2-6 แสดงตารางการเข้ารหัสแบบ base64

ตัวอย่างของการถอดรหัสโดยวิธี base64 เช่นสมมุติว่าเพื่อนเราต้องการส่งข้อมูลมาให้เราโดยส่งมาจากไทยเมล์ โดยที่เวลาไทยเมล์ เวลาส่งเมล์ไปที่อื่นจะเรารหัสด้วยวิธี base64 ฉนั้นฝ่ายรับต้องทำการถอดรหัสข้อมูลนั้นก่อน ถ้าข้อมูลที่ส่งมาให้เราเป็น oaGh เราต้องจำข้อมูลที่ได้รับมาทำการเปรียบเทียบกับค่าในตารางก่อนจากตัวอย่างคือ o = 40 , a = 26 , G = 6 , h = 33 เมื่อเทียบเสร็จให้เรานำค่าตัวเลขที่ได้แปลงเป็นฐาน 2 ขนาด 6 บิตจะได้เป็น

o = 101000

a = 011010

G = 000110

H = 100001

จากนั้นให้เรานำค่าเลขฐาน 2 ขนาด 6 บิตที่แปลงได้นำมาต่อกันจะได้เป็นค่า

101000 011010 000110 100001

ให้เราทำการจัดกลุ่มจากด้านซ้ายมือกลุ่มละ 8 บิต จะได้เป็น

10100001 10100001 10100001

จะสามารถแบ่งข้อมูลออกมาได้เป็น 3 กลุ่ม ซึ่งแต่ละกลุ่มมีค่าเท่ากับ 161 ซึ่งมีค่าเท่ากับ “ก” ในตารางแอสกี ทำให้เรารู้ว่าข้อมูลที่เพื่อนเราส่งมาให้เราเป็น “กกก”

Binary

เป็นข้อมูลที่ต่อเนื่องกันเป็นค่าไบนารี ไม่แบ่งออกเป็นบรรทัด ซึ่งข้อมูลประเภทนี้จะส่งโดยไม่มีการเข้ารหัสข้อมูล

เอกสารนี้มีไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Seven-Bit

เป็นข้อมูลที่มีค่าแอสกี 8 บิต ซึ่งข้อมูลประเภทนี้จะส่งโดยไม่มีการเข้ารหัสข้อมูล

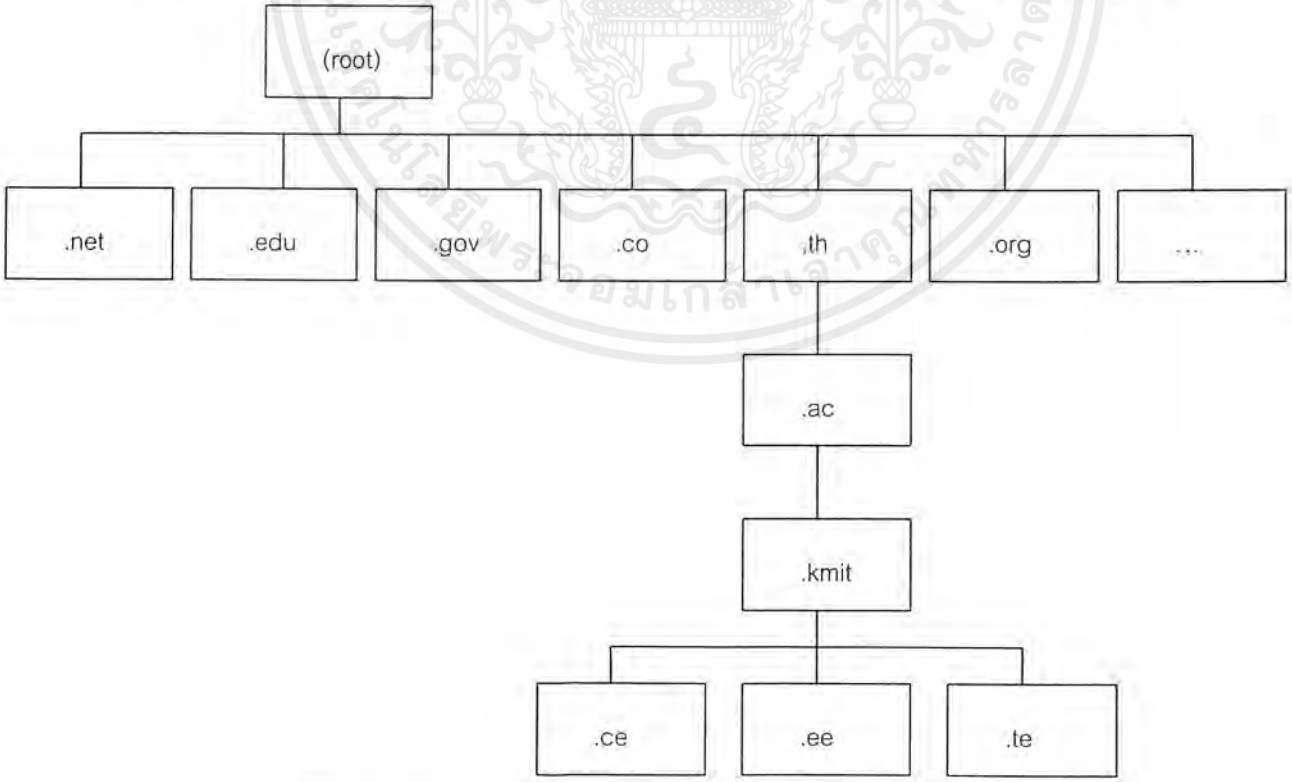
X-Token

เป็นเทคนิคการเข้ารหัสที่ต้องมีการติดต่อและตกลงกันระหว่างด้านผู้ส่งและผู้รับของ SMTP เซิร์ฟเวอร์ก่อน

2.5 ทฤษฎีเบื้องต้นเกี่ยวกับดีเอ็นเอส (DNS)

เครื่องคอมพิวเตอร์และอุปกรณ์สื่อสารต่างๆ ที่อยู่บนเครือข่ายอินเทอร์เน็ตนั้น แต่ละตัวจะต้องมีเลขไอพีแอดเดรส (IP Address ที่ต้องไม่ซ้ำกัน) การติดต่อสื่อสารระหว่างเครื่องต่างๆ จะใช้ IP Address เป็นหลัก แต่เราคงอาจจะสงสัยกันว่าเหตุใดเราจึงสามารถดูเว็บไซต์ต่างๆ โดยการพิมพ์ชื่อ เช่น www.kmitl.ac.th ก็สามารถเข้าไปในเว็บไซต์ของลาดกระบังได้ โดยที่ไม่ต้องใส่ IP Address ที่เป็นเช่นนี้เพราะเรามีระบบ DNS (Domain Name System) เป็นระบบของการแปลงกลับไปกลับมาระหว่างชื่อโฮสต์ให้เป็น IP Address เช่น www.kmitl.ac.th มี IP Address คือ 161.246.10.21 เป็นต้น

การแปลงจากชื่อโฮสต์ให้เป็น IP Address เรียกว่า Forward mapping คือแปลงจากชื่อ www.kmitl.ac.th ให้เป็น IP Addrsss 161.246.10.21 และการแปลงจาก IP Address ให้เป็นชื่อโฮสต์เรียกว่า Reverse mapping เช่นแปลงจาก IP Address 161.246.10.21 ให้เป็นชื่อโฮสต์คือ www.kmitl.ac.th



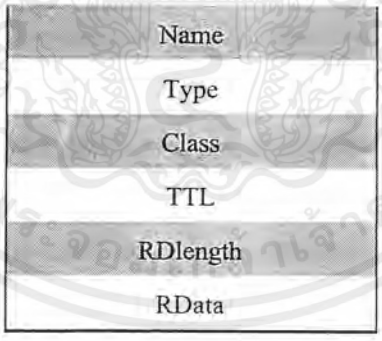
รูปที่2-7 แสดงแผนภูมิแบบลำดับชั้นอย่างง่ายของโดเมนเนมในอินเตอร์เน็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงสร้างของการจัดการ DNS เป็นลำดับขั้น (hierarchical) โดยมองได้ในลักษณะของแผนภูมิต้นไม้ จะเห็นว่าลำดับขั้นบนสุดของแผนภูมิเรียกว่ารูท (root) และชั้นถัดลงมาแยกย่อยเป็นลำดับ ในแต่ละลำดับของ DNS เรียกว่าโหนด (node) ซึ่งแต่ละโหนดจะมีชื่อที่ถูกกำหนดไว้หรือที่เราเรียกว่าโดเมนเนมจากตัวอย่างในลำดับล่างสุดจะมีอยู่ 3 โนด คือ ce , ee , te ซึ่งอยู่ภายใต้ลำดับชั้นของ .kmitl และอยู่ภายใต้ .ac และ .th ตามลำดับ ซึ่งในลักษณะนี้ถ้าเรามองที่ลำดับชั้นล่างสุด ก็จะได้ชื่อแบบเต็มๆของแต่ละโหนดว่า ce.kmitl.ac.th , ee.kmitl.ac.th , te.kmitl.ac.th โดยมีข้อสังเกตคือ ในการอ่านชื่อโหนดเต็มๆนี้ให้อ่านไล่จากชื่อลำดับชั้นล่างสุด คั่นด้วยจุด แล้วตามด้วยชื่อลำดับชั้นถัดไปเรื่อยๆจนถึงชั้นบนสุดที่เป็นรูท (ใช้เครื่องหมายจุดแทน) ซึ่งไม่ต้องนำมาใส่ เท่านี้เราก็จะได้ชื่อของแต่ละโหนด แบบเต็มๆที่เรียกว่า fully qualified name และในโหนดแต่ละชื่อสามารถกำหนดให้มีความยาวได้ไม่เกิน 63 ไบต์ และรวมจำนวนชื่อได้ไม่เกิน 256 ตัวอักษร โดยอักษรตัวใหญ่และเล็กจะมีความหมายแตกต่างกัน

2.5.1 ข้อมูลที่เก็บใน DNS

การทำงานของ DNS เป็นการทำงานแบบไคลเอนต์เซิร์ฟเวอร์ โดยใช้ฐานข้อมูลแบบกระจาย ซึ่งข้อมูลที่ตัวฐานข้อมูลจะถูกลำดับไว้เป็นเรคอร์ด เรียกว่า resource record ในแต่ละเรคอร์ดจะมีข้อมูลที่จำเป็นในการทำงานเช่น หมายเลข IP address คู่กับชื่อเครื่อง , ชื่อเครื่องที่จะทำหน้าที่รับส่งอีเมล์ของโดเมนเนมนั้น เป็นต้น ส่วนจำนวนเรคอร์ดจะมีมากหรือน้อยขึ้นอยู่กับจำนวนโดเมนเนมที่อยู่ในความรับผิดชอบของเซิร์ฟเวอร์นั้น ซึ่งลักษณะของ resource record จะมีข้อมูลในหนึ่งเรคอร์ดเป็นดังรูป



รูปที่ 2-8 แสดงฟิลด์ต่างๆของ resource record ใน DNS

ความหมายของฟิลด์ต่างๆใน resource record มีดังนี้

Name เป็นชื่อโดเมนของข้อมูลเรคอร์ดนี้

Type แสดงชนิดของ resource record มีขนาด 16 บิต ข้อมูลในเรคอร์ดส่วนใหญ่จะเป็น Type A หรือข้อมูล IP Address มากที่สุด

Class แสดงโปรโตคอลของ resource record ในคลาสมีขนาด 16 บิตฟิลด์จะมีค่าเป็น IN หรือ "อินเตอร์เน็ต" เสมอ

TTL กำหนดค่าเวลาการปรับปรุงข้อมูลเป็นวินาที มีขนาด 32 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RDlength กำหนดขนาดความยาวของฟิลด์ Rdata

RData มีขนาดไม่แน่นอนตามค่าของ RDlength

Type	คำอธิบาย
A	ข้อมูลชนิด A คือ Address record จะระบุชื่อ domain name และ หมายเลข IP Address ที่คู่กัน
NS	ข้อมูลชนิด NS คือ Name Server record เป็นข้อมูลระบุชื่อ DNS server ที่รับผิดชอบโดเมน (domain) นี้
MD	Mail Destination ไม่มีการใช้งานเนื่องจากใช้ค่า MX แทน
MF	Mail Forwarder ไม่มีการใช้งานเนื่องจากใช้ค่า MX แทน
CNAME	ข้อมูลชนิด CNAME คือ Canonical Name record ใช้ระบุชื่อโดเมนเนมอื่นที่ตั้งเอาไว้คู่กับโดเมนเนมหลัก ใช้ในกรณีที่มีการตั้งชื่อโดเมน อื่นตรงกับโดเมนหลัก (Alias name) เช่นมีชื่อ abccompany.com เป็น domain name หลักและกำหนดให้มีชื่อเดียวกับ abccompany.net เป็นต้น เรียกว่าชื่อทั้งสอง aliasกัน
SOA	ข้อมูลชนิด SOA (Start Of Authority record) จะระบุรายละเอียดของหน่วยความจำแคชที่ใช้เก็บข้อมูลคำตอบที่ได้และระบุชื่อ Secondary DNS Server
MB	Mailbox ใช้กำหนดชื่อตู้จดหมาย (mailbox) ของ domain นั้น แต่ยังไม่มีการใช้งาน
MG	Mail Group member ยังไม่มีการใช้งาน
WKS	Well- Know Service description
PTR	ข้อมูลชนิด PTR คือ pointer record จะระบุหมายเลข IP address และชื่อโดเมนที่คู่กันเพื่อใช้กรณีที่ทราบ IP address แล้วต้องการทราบชื่อโดเมนเนม ด้วย
HINFO	Host information ใช้แสดงข้อมูลรายละเอียดของโฮสท์ นั้น เช่น ระบบปฏิบัติการ,ซีพียู
MINFO	Mailbox หรือ mail list แสดงข้อมูลจากเมล์บล็อก
MX	Mail exchanger ใช้ระบุชื่อเครื่องเมล์เซิร์ฟเวอร์ที่รับผิดชอบโดเมนนั้น

ตารางที่ 2-7 แสดงรายละเอียดของข้อมูลในฟิลด์ Type

ข้อมูลส่วนใหญ่ในฐานข้อมูลของดีเอ็นเอสเซิร์ฟเวอร์ จะเป็น Type A คือเป็นเรคอร์ดที่ระบุชื่อเครื่อง (โดเมนเนม) และหมายเลข IP address คู่กัน สำหรับข้อมูลที่มี Type เป็น MX หมายถึงการระบุให้เครื่องนี้ทำหน้าที่เป็นเมล์เซิร์ฟเวอร์ (Mail Exchanger) ของโดเมนเนมนั้นๆ ซึ่งผู้ใช้สามารถส่งอีเมล์ถึงผู้ใช้ในโดเมนที่กำหนดได้โดยไม่ต้องระบุชื่อเครื่องเป็น smtp_gw.abccompany.com ของโดเมน abccompany.com ส่วนในเครื่องเมล์เซิร์ฟเวอร์ก็มีผู้ใช้ชื่อ somsak อยู่ และเมื่อมีการส่งอีเมล์มาให้ somsak จะอ้างอิงชื่อผู้รับและชื่อโดเมนได้เลยคือ somsak@abccompany.com ไม่ต้องระบุเต็มว่า somsak@smtp_gw.abccompany.com ให้ยุ่งยาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 ทฤษฎีเบื้องต้นเกี่ยวกับออราเคิล (ORACLE)

ดาต้าเบสเซิร์ฟเวอร์เป็นส่วนที่สำคัญในการจัดการปัญหาที่เกี่ยวกับการจัดการข้อมูลต่างๆ ซึ่งตัวเซิร์ฟเวอร์จะต้องมีความสามารถในการจัดการข้อมูลที่มีจำนวนมากแม้ในสภาวะที่มีผู้ใช้ กำลังใช้ข้อมูลชุดเดียวกัน ก็ต้องจัดการให้ผู้ใช้แต่ละคนทำงานกับข้อมูลนั้นได้ (concurrency control) และยังต้องมีความสามารถในการป้องกันการบุกรุกเข้ามาใช้ข้อมูลจากผู้ใช้ที่ไม่มีสิทธิในระบบนั้นๆ รวมถึงการทำการกู้ข้อมูล (recovery) กับคืนมาในกรณีที่ เกิดการเสีย (failure) ของงานเกิดขึ้นได้

Oracle 8 เป็นฐานข้อมูลเชิงวัตถุสัมพันธ์(object-relational database management system: ORDBMS) ซึ่งเป็นระบบฐานข้อมูลที่พัฒนาขึ้นมาจากระบบฐานข้อมูลแบบเดิมคือระบบฐานข้อมูลเชิงสัมพันธ์ (relational database management system: RDBMS) ซึ่งมีความสามารถที่เพิ่มขึ้นในการจัดเก็บข้อมูลที่มีความซับซ้อนมาก ๆ เช่น ข้อมูลมัลติมีเดีย, รูปภาพ, เสียง สามารถที่จะจัดการกับข้อมูลเหล่านี้ได้อย่างมีประสิทธิภาพ โดยมีการเพิ่มชนิดข้อมูล (DataType) ใหม่ขึ้นมาเพื่อรองรับการจัดเก็บข้อมูล และมีชนิดของข้อมูลที่ผู้ใช้สามารถที่จะกำหนดขึ้นมาเอง (User-Defined DataType) ข้อมูลชนิดนี้จะมีคุณสมบัติของออบเจ็กต์โอเรียนท์เด็ดเหมือนกับออบเจ็กต์ที่เขียนขึ้นจากภาษาที่เป็นแบบโปรแกรมเชิงวัตถุแบบต่าง ๆ แต่ยังไม่เทียบเท่าภาษาเหล่านั้น

2.6.1 Java Database Connectivity และการติดต่อกับฐานข้อมูล

- **ความหมายของ JDBC**

JDBC (Java Database Connectivity) ถูกพัฒนาโดย JavaSoft Department ของบริษัท Sun MicroSystems ซึ่ง JDBC ก็คือฟังก์ชั่นมาตรฐานหรือ Java Application Programming Interface(API) สำหรับการเชื่อมต่อกับฐานข้อมูล มีการสร้างระดับการเชื่อมต่อในการสื่อสารกับฐานข้อมูลลักษณะเดียวกับ ODBC ของบริษัทไมโครซอฟท์ ซึ่งในปัจจุบันถือว่าเป็นมาตราฐานของการเชื่อมต่อกับฐานข้อมูลของเครื่องคอมพิวเตอร์ส่วนบุคคล และระบบเครือข่ายท้องถิ่น แต่ตามหลักการทำงานของ JDBC และ ODBC นั้นตั้งอยู่บนมาตราฐานเดียวกันคือ X/Open SQL Call-Level Interface ของระบบ X-Windows และ JDBC Driver และต้องเข้ากันได้กับมาตราฐานในการเข้าถึง SQL (ANSI SQL Entry Level Standard) และต้องผ่าน Conformance test ซึ่งจาวาซอฟท์เป็นผู้กำหนดขึ้น

- **โครงสร้างของ JDBC**

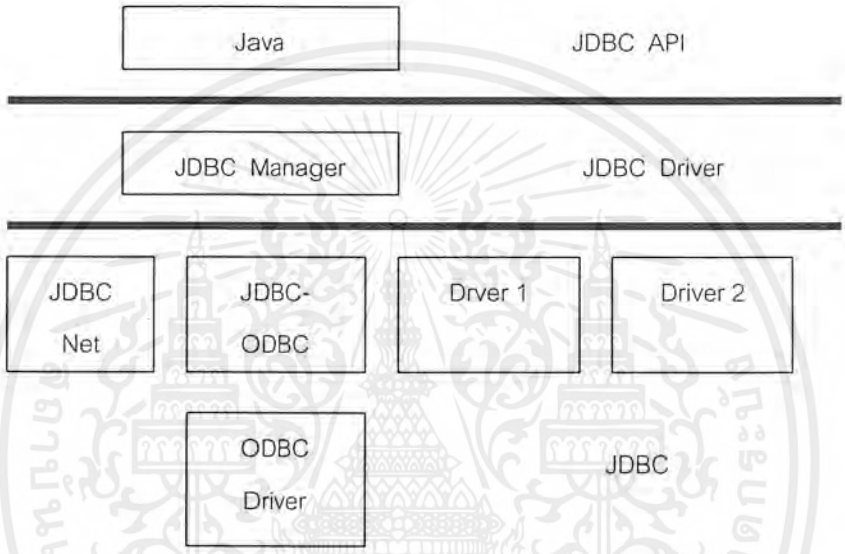
โครงสร้างการเชื่อมต่อภายใน JDBC ประกอบด้วย 3 ระดับหลักคือ JDBC API,JDBC Driver API และ JDBC Driver ดังรูปที่ 2 ระดับบนสุด JDBC API เป็นระดับของฟังก์ชั่น API ที่อำนวยความสะดวกให้แก่โปรแกรมประยุกต์ ระดับกลาง JDBC Driver (มีไดร์ฟเวอร์ที่ต่างกันอยู่ 4 ชนิด) ที่เหมาะสม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **รูปแบบของ JDBC ไดร์ฟเวอร์**

JDBC ไดร์ฟเวอร์(Drivers) หรือส่วนที่อยู่เบื้องหลังการทำงานของฟังก์ชั่น API ต่าง ๆ ของ JDBC ถูกจัดแบ่งออกเป็นทั้งหมด 4 ชนิด คือ

1. JDBC – OCBC Bridge
2. Native – API (Partly – Java) Drivers
3. Net – Protocol (All – Java) Drivers
4. Native – Protocol (All – Java) Drivers



รูปที่ 2-9 แสดงระดับการเชื่อมต่อของ JDBC API

- **ไดร์เวอร์ JDBC ของออราเคิล**

บริษัทออราเคิลได้ทำการพัฒนาตัวไดร์เวอร์ JDBC สำหรับใช้ในการติดต่อระหว่างแอบพลิเคชั่นและตัวฐานข้อมูลที่พัฒนาขึ้นมาจากภาษาจาวาโดยสามารถที่จะแยกออกเป็นไดร์เวอร์สำหรับไคล์เอ็นท์และไดร์เวอร์สำหรับเซิฟเวอร์ ซึ่งไดร์เวอร์สำหรับไคล์เอ็นท์จะมีด้วยกัน 2 ชนิด คือ JDBC Thin Driver และ JDBC OCI Driver ซึ่งสามารถใช้เขียนโปรแกรมในรูปของจาวาแอปพริเคชั่น หรือจาวาแอปเพล็ต ก็ได้ ส่วนไดร์เวอร์สำหรับเซิร์ฟเวอร์นั้นจะมีเซิร์ฟเวอร์ไซด์ JDBC เป็นตัวจัดการให้จาวาเวอร์ชวล แมชชีน (JVM) ติดต่อกับเอสคิวแอลเอ็นจิ้น (SQL engine) ซึ่งสามารถแบ่งตัวไดร์เวอร์ออกได้เป็น 3 แบบคือ

1. JDBC Thin Driver จะใช้จาวาซ็อกเก็ทในการติดต่อกับดาต้าเบสเซิร์ฟเวอร์ (Database Server) ซึ่งเหมาะสำหรับจาวาแอปเพล็ท โดยเมื่อเราทำการเลือก URL จาก HTML page ที่มี Java Applet tag อยู่เว็ปเซิร์ฟเวอร์ จะทำการโหลดจาวาแอปเพล็ท และ JDBC Thin Driver ไปยังไคล์เอ็นท์

2. JDBC OCI Driver จะทำการเชื่อมต่อ JDBC กับดาต้าเบสเซิฟเวอร์โดยใช้ Oracle Call Interface (OCI) เรียกว่า Call native methods ซึ่งถูกเขียนขึ้นมาด้วยโค้ดภาษาซี ทำให้ไม่เหมาะที่จะใช้ในจาวาแอปเพล็ท แต่จะเหมาะกับจาวาแอปพริเคชั่น หรือ Java Middle tiers

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.JDBC Server Driver จะใช้สำหรับ เซิร์ฟเวอร์เท่านั้นซึ่งเซิร์ฟเวอร์ไดร์เวอร์นี้จะมี JDBC สำหรับโปรแกรมภาษาจาวา ที่ใช้ในดาต้าเบสเซิฟเวอร์ เช่น Java Stored Procedure, Enterprise Java Beans (EJB) , SQL และ PL/SQL

ลำดับในการติดต่อดาต้าเบสเซิร์ฟเวอร์ ในการสืบค้น (Query) ข้อมูลจากตัวไคลเอนท์

- Importing Packages
- Registering the JDBC Drivers
- Opening a Connection to a Database
- Creating a Statement Object
- Executing a Query and Returning a Result Set Object
- Processing the Result Set
- Closing the Result Set and Statement Objects
- Closing the Connection

ขั้นตอนที่ 1 การ Importing Packages

ทำการ Importing Packages ต่าง ๆ ที่ต้องการใช้ในการเขียนโปรแกรม เช่น

Import java.sql.*

Import java.math.*

ขั้นตอนที่ 2 ลงทะเบียนไดรเวอร์ JDBC (Registering the JDBC Drivers)

ขั้นตอนนี้จะต้องทำการเรียกใช้ สเตติกเมธอด (Static Methods) registerDriver( ) ของคลาส JDBC DriverManager ซึ่งเป็นคลาสสำหรับจัดการกับ JDBC Driver ต่าง ๆ

DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver");

ขั้นตอนที่ 3 เปิดการติดต่อกับฐานข้อมูล (Opening a Connection to a Database)

ขั้นตอนนี้จะทำการเรียกใช้สเตติกเมธอด getConnection( ) ของคลาส JDBC DriverManager ซึ่งสามารถใช้ JDBC OCI Driver หรือ JDBC Thin Driver

เปิดการติดต่อกับฐานข้อมูลโดยใช้ JDBC OCI Driver

ในการใช้ JDBC OCI Driverเพื่อทำการติดต่อกับฐานข้อมูล จำเป็นจะต้องมีTNSNAME ซึ่งจะถูกเก็บไว้ในไฟล์tnsnames.oraบนวินโดว์ ไฟล์นี้จะถูกเก็บใน[ORACLE_HOME]\NETWORK\ADMIN ส่วนบนยูนิกส์จะอยู่ใน /var/opt/oracle

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่าง ถ้าคุณต้องการติดต่อกับฐานข้อมูลบนโฮสต์ชื่อ MyHostString โดยใช้ผู้ใช้เป็น scott และรหัสผ่านเป็น tiger ส่วน TNSNAME ก็คือ MyHostString สามารถเขียนการติดต่อได้ดังนี้

```
Connection conn =
DriverManager.getConnection ("jdbc:oracle:oci8:@MyHostString","scott",tiger" );
```

เปิดการติดต่อกับฐานข้อมูลโดยใช้ JDBC Thin Driver

ในการใช้ JDBC Thin Driver นั้นใช้ในแอปเพล็ทดังนั้นจึงไม่ต้องใช้ TNSNAMES ในการติดต่อกับฐานข้อมูล แต่จะมีวิธีการติดต่อ 2 แบบคือ

- ใช้ host name , TCP/IP พอร์ตและ Oracle SID (System identifier) เช่น ต้องการติดต่อกับฐานข้อมูลบนเครื่องชื่อ myhost ที่ใช้ TCP/IP บนพอร์ต 1521 สำหรับ Database SID ก็คือ ORCL โดยมีผู้ใช้เป็น scott และรหัสผ่านเป็น tiger สามารถเขียนการติดต่อได้ดังนี้

```
Connection conn =
DriverManager.getConnection("jdbc:oracle:thin:@myhost:1521:orcl","scott","tiger");
```

- ใช้ *keyword – value* จากตัวอย่างเดิมสามารถเขียนได้ดังนี้

```
Connection conn =
DriverManager.getConnection ("jdbc:oracle:thin:@(description=(address=(host=myhost)
(protocol=tcp)(port=1521))(connect_data=(sid=orcl)))","scott","tiger");
```

ขั้นตอนที่ 4 สร้างออฟเจ็กต์เงื่อนไข (Creating a Statement Object)

เมื่อคุณต้องการติดต่อกับฐานข้อมูลเพื่อประมวลผล ขั้นแรกจะต้องสร้างออฟเจ็กต์การเชื่อมต่อ (Connection object) หลังจากนั้นจะต้องสร้างออฟเจ็กต์เงื่อนไข (Statement object) โดยใช้เมธอด CreateStatement() ของออฟเจ็กต์ JDBC Connection ซึ่งจะทำการรีเทิร์น (return) ออฟเจ็กต์ของคลาส JDBC Statement ออกมาที่ stmt สามารถเขียนได้ดังนี้

```
Statement stmt = conn.createStatement();
```

ขั้นตอนที่ 5 การประมวลผลการคิวรีและการรีเทิร์นออฟเจ็กต์ Result Set (Executing a Query and Returning a Result Set Object)

ในการใช้คิวรีเรียกไปที่ฐานข้อมูลจะใช้ เมธอด executeQuery() ของออปเจ็กต์ Statement ซึ่งเมธอดนี้จะทำการรันคำสั่ง SQL ที่ป้อนเข้าไปและรีเทิร์นออฟเจ็กต์ของคลาส JDBC ResultSet

ตัวอย่าง ต้องการเลือกคอลัมน์ ENAME (employee name) จากตาราง EMP โดยผลลัพธ์จะไปเก็บไว้ใน ออปเจ็กต์ Resultset ชื่อ rset สามารถเขียนได้ดังนี้

```
ResultSet rset = stmt.executeQuery("SELECT ename FROM emp");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นตอนที่ 6 การประมวลผล ResultSet (Processing the ResultSet)

ขั้นตอนนี้จะทำการประมวลผล ResultSet โดยใช้เมธอด next() ของออฟเจ็กต์ ResultSet ในการไล่ลำดับของผลลัพธ์ โดยการวนลูปไล่ผลลัพธ์ที่ละ row จนกระทั่งถึง row สุดท้าย ซึ่งเราอาจจะใช้เมธอด getXXX() ของออฟเจ็กต์ ResultSet ในการดึงค่าของผลลัพธ์ไปใช้งานได้ โดยที่ XXX คือชนิดของข้อมูลในภาษาจาวาดังนั้นสามารถที่จะเขียนได้ดังนี้

```
while(rset.next())
        System.out.println(rset.getString(1));
```

ขั้นตอนที่ 7 การปิดออฟเจ็กต์ ResultSet และ Statement (Closing the Result Set and Statement Objects)

เมื่อใช้งานออฟเจ็กต์ ResultSet และ Statement เสร็จแล้วจะต้องทำการปิดด้วย เนื่องจากออฟเจ็กต์ ResultSet และ Statement ใช้ Oracle JDBC Driver ซึ่งไดร์เวอร์เหล่านี้ไม่มี เมธอด finalizer ซึ่งทำหน้าที่กำจัดขยะออกจากหน่วยความจำ ดังนั้นเราจึงต้องใช้เมธอด close() ของคลาส ResultSet และ Statement ในการกำจัดขยะออกจากหน่วยความจำแทน ซึ่งถ้าเราไม่ทำการปิดออฟเจ็กต์ ResultSet และ Statement แล้วอาจจะทำให้เมโมรี่ไม่เพียงพอได้

```
rset.close();
stmt.close();
```

ขั้นตอนที่ 8 การปิดการติดต่อกับฐานข้อมูล (Closing the Connection)

เมื่อการใช้งานในการติดต่อดึงข้อมูลจากฐานข้อมูลเสร็จแล้ว จะต้องทำการปิดการติดต่อด้วยโดยใช้ เมธอด close() ของคลาส Connection ดังนั้น ถ้าออฟเจ็กต์ Connection คือ conn จะทำการปิดได้ดังนี้

```
conn.close();
```

ตัวอย่าง การติดต่อ , คิวรี่ และประมวลผลผลลัพธ์

```
import java.sql.*;
import java.math.*;
import java.io.*;
import java.awt.*;

class JdbcTest{
    public static void main (String args[]) throws SQLException {
    // Load Oracle driver
    DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//Connect to the local database
Connection conn =
   DriverManager.getConnection("jdbc:oracle"thin"@myhost:1521:ORCL",
         "scott","tiger");
//Query the employee names
Statement stmt = conn.createStatement();
ResultSet rset = stmt.executeQuery ("SELECT account FROM infomail");
//Print the name out
while (rset.next())
         System.out.println(rset.getString(1));
//close the result set,statement, and the connection
rset.close();
stmt.close();
conn.close();
}
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
ทฤษฎีและการออกแบบ

การออกแบบเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์นั้นเราได้พยายามให้โปรแกรมมีลักษณะเป็นออบเจ็กต์โอเรียนท์เต็ดเพื่อให้ง่ายในการปรับปรุงแก้ไขโปรแกรมโดยไม่กระทบต่อการทำงานของออบเจ็กต์อื่น และมีการใช้คุณสมบัติต่างๆของออบเจ็กต์เช่น การสืบทอดคุณสมบัติ , การซ่อนรายละเอียด เป็นต้น และทำการออกแบบการทำงานของโปรแกรมให้มีการทำงานเป็นเธรดเพื่อให้สามารถรองรับการเชื่อมต่อจากผู้ใช้ได้เป็นจำนวนมาก โดยผู้ใช้สามารถเรียกใช้บริการเซิร์ฟเวอร์ผ่านทางเบราเซอร์ การออกแบบเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์นั้น เราได้ออกแบบความต้องการออกเป็น 4 ส่วนหลักๆคือ

1 ส่วนที่ใช้ในการติดต่อกับผู้ใช้
2 ส่วนของเว็บเซิร์ฟเวอร์
3 ส่วนของเมล์เซิร์ฟเวอร์
4 ส่วนที่ติดต่อกับดาต้าเบส

ซึ่งในส่วนที่ใช้ติดต่อกับผู้ใช้เราได้ออกแบบให้ผู้ใช้ที่ใช้งานเครื่องไคลเอนต์ (client) ติดต่อมายังเครื่องเซิร์ฟเวอร์ (server) ผ่านทางเบราเซอร์ โดยเมื่อผู้ใช้พิมพ์ URL แล้วกดเอ็นเตอร์ จากนั้น URL จะถูกส่งมาที่เครื่องเซิร์ฟเวอร์ ซึ่งส่วนของเว็บเซิร์ฟเวอร์นี้จะทำการส่งไฟล์เอกสาร html หรือไฟล์รูปภาพที่เครื่องไคลเอนท์ร้องขอมากลับไปให้เครื่องไคลเอนต์ ซึ่งเอกสารจะแสดงที่เบราเซอร์ เป็นหน้าจอให้ผู้ใช้เข้ามาล็อกอิน หรือทำการลงทะเบียนก่อน ในกรณีที่ผู้ใช้ยังไม่ได้ลงทะเบียนเป็นสมาชิก ซึ่งจะมีการทำงานคล้ายกับไทยเมล์ เมื่อผู้ใช้เลือกที่จะลงทะเบียน เครื่องเซิร์ฟเวอร์ จะส่งแบบฟอร์มเอกสารมาให้เพื่อให้ผู้ใช้ทำการกรอกข้อมูลต่างๆ เช่น ชื่อที่จะใช้ในการล็อกอิน , รหัสผ่าน และข้อมูลต่างๆเป็นต้น และเมื่อผู้ใช้กรอกข้อมูลต่างๆลงในแบบฟอร์มเสร็จเรียบร้อยแล้ว แบบฟอร์มนั้นจะถูกส่งมาที่เครื่องเซิร์ฟเวอร์เพื่อทำการเพิ่มจำนวนผู้ใช้ลงในส่วนของดาต้าเบส จากนั้นเซิร์ฟเวอร์จะส่งไฟล์เอกสารหน้าแรกไปให้เครื่องไคลเอนต์อีกครั้ง ซึ่งจะแสดงหน้าจอให้ผู้ใช้ล็อกอิน เป็นอันเสร็จในส่วนของการลงทะเบียนเป็นสมาชิก จากนั้นผู้ใช้สามารถเข้าไปเช็คเมล์ได้โดยทำการพิมพ์ชื่อล็อกอินและรหัสผ่านลงไปในแบบฟอร์ม ชื่อล็อกอินและรหัสผ่านจะถูกส่งไปที่เครื่องเซิร์ฟเวอร์เครื่องเซิร์ฟเวอร์จะทำการเรียกไฟล์ซีจีไอที่เขียนขึ้นเพื่อใช้ในการเช็คชื่อที่ล็อกอินกับรหัสผ่านว่าถูกต้องหรือไม่ ถ้าไม่ถูกต้องจะส่งข้อความแสดงความผิดพลาดในการล็อกอินกลับไปแสดงที่เครื่องไคลเอนต์ แต่ถ้าถูกต้องเครื่องเซิร์ฟเวอร์ก็จะส่งเอกสารที่แสดงว่าผู้ใช้มีจดหมายอยู่จำนวนกี่ฉบับ และมีจำนวนกี่ฉบับที่ผู้ใช้ยังไม่ได้อ่านจดหมาย ซึ่งตอนนี้เรายังไม่มีจดหมายเพราะว่าเราเพิ่งทำการลงทะเบียน ในกรณีที่มีเพื่อนส่งจดหมายมาให้เรา เบราเซอร์จะแสดงในส่วนของหัวข้อ (subject) เป็นตัวอักษรสีน้ำเงินเข้ม ผู้ใช้สามารถคลิกที่ตัวอักษรนั้น เครื่องเซิร์ฟเวอร์จะทำการติดต่อกับดาต้าเบสแล้วทำการส่งส่วนของเนื้อความในจดหมายมาแสดงที่เครื่องไคลเอนต์ และเมื่อผู้ใช้อ่านจดหมายแล้วสามารถที่จะทำการตอบจดหมายกลับได้ โดยคลิกที่อักษรคำว่า Reply ซึ่งเป็นตัวอักษรสีน้ำเงินเข้ม เครื่องเซิร์ฟเวอร์จะทำการส่งแบบฟอร์มให้ผู้ใช้กรอกจดหมาย และเมื่อกรอกเสร็จแล้วก็ทำการส่งจด

เอกสารนี้ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายออกไป ซึ่งส่วนนี้จะเป็นส่วนของเมล์เซิร์ฟเวอร์ และเมื่อมีเพื่อนเราส่งจดหมายมาให้เรา เราต้องทำการเก็บข้อมูลต่างๆไว้ในตัวแปรเช่น อีเมล์แอดเดรสของผู้ส่ง , ข้อความจดหมาย , หัวข้อจดหมาย เป็นต้น จากนั้นจึงทำการเรียกไฟล์ซีจีไออีกครั้งเพื่อทำการจัดเก็บข้อมูลต่างๆลงในดาต้าเบส

* ในการอ่านจดหมายเราอ่านผ่านเบราเซอร์จึงไม่จำเป็นต้องใช้ POP3 หรือ IMAP4

3.1 การทดลองการติดต่อระหว่างไคลเอนต์กับเซิร์ฟเวอร์

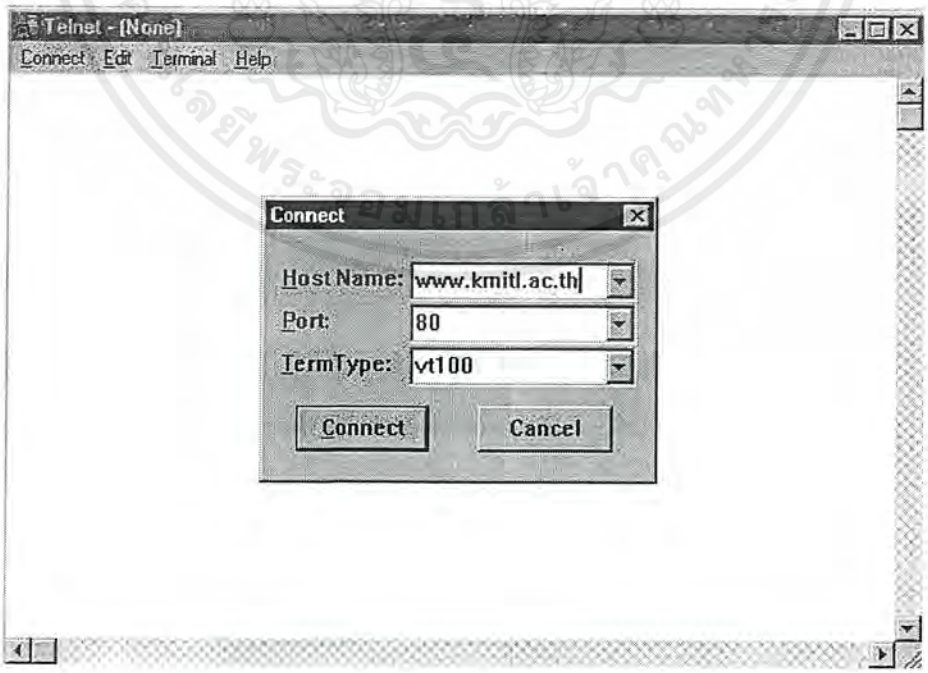
เมื่อเราได้ออกแบบความต้องการแล้ว เราก็ทำการทดลองว่าสิ่งที่เราออกแบบไว้มีความเป็นไปได้มากน้อยแค่ไหนโดยทดลองติดต่อระหว่างเครื่องไคลเอนท์กับเครื่องเซิร์ฟเวอร์เพื่อดูว่ามีการร้องขอและการตอบสนองกันอย่างไรบ้าง

3.1.1 ทดลองโพรโตคอล HTTP ด้วยเทลเน็ท (Telnet)

เพื่อให้เข้าใจโปรโตคอล HTTP มากยิ่งขึ้น เราจะใช้โปรแกรมเทลเน็ท (Telnet) เพื่อทดลองส่งข้อความร้องขอไปยังเซิร์ฟเวอร์ และดูข้อความตอบสนองที่ส่งกลับจากเซิร์ฟเวอร์กันบ้าง

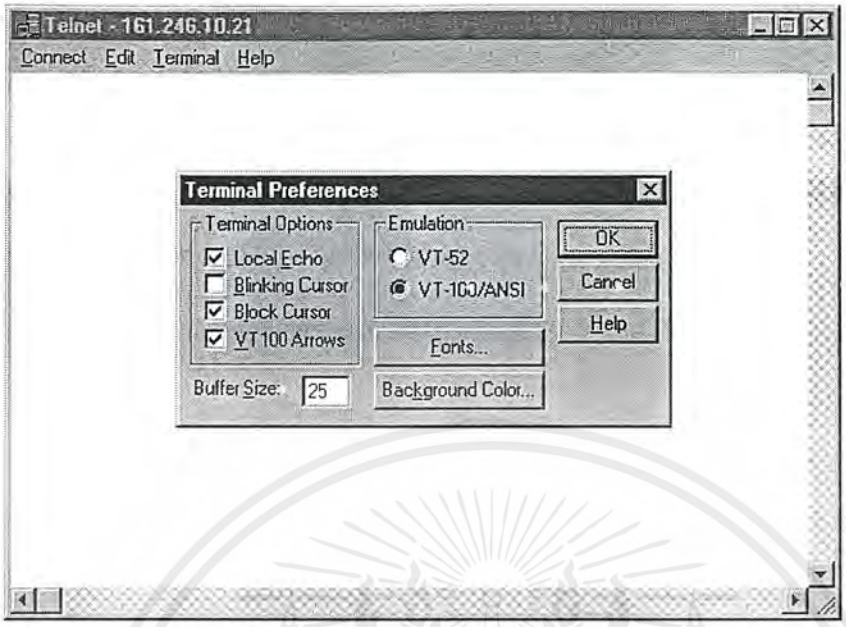
โปรแกรม Telnet เป็นโปรแกรมที่มีการทำงานที่ยืดหยุ่นมาก เราสามารถใช้โปรแกรม Telnet เชื่อมต่อไปยังเซิร์ฟเวอร์ที่ให้บริการต่างๆได้ เพียงแต่ระบุชื่อเซิร์ฟเวอร์และหมายเลยพอร์ตให้ถูกต้อง

ในการทดลองนี้จะทำกับเซิร์ฟเวอร์ใดก็ได้ ข้อสำคัญคือ เซิร์ฟเวอร์นั้นจะต้องเปิดบริการ www ด้วย เริ่มต้นให้เราเรียกโปรแกรมเทลเน็ทพร้อมทั้งป้อนชื่อเซิร์ฟเวอร์ที่ต้องการติดต่อ และหมายเลขพอร์ต 80 การป้อนชื่อเซิร์ฟเวอร์จะป้อนเป็นชื่อโฮสต์และโดเมนเนม หรือระบุเป็นเลข IP โดยตรงก็ได้ เช่น telnet www.kmitl.ac.th 80 หรือ telnet 161.246.10.21 80 หลังจากนั้นวินโดว์โปรแกรมเทลเน็ทจะปรากฏบนหน้าจอดังรูปที่ 3.1



รูปที่ 3-1 แสดงการป้อนข้อมูลสำหรับการเชื่อมต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-2 แสดงการกำหนดค่าต่างๆในโปรแกรม Telnet

ขั้นต่อไปให้เลือกคำสั่ง Preferences จากเมนู Terminal จะปรากฏไดอะล็อกบ็อกซ์ Terminal Preferences ดังรูปที่ 3.2 จากนั้นให้กาเครื่องหมายถูกที่เช็กบ็อกซ์ Local Echo เพื่อให้โปรแกรมเทลเน็ทแสดงข้อความที่เราพิมพ์ออกทางจอเทอร์มินัลด้วย มิฉะนั้นจะไม่เห็นข้อความที่เราพิมพ์เข้าไป ต่อจากนั้นจึงคลิกปุ่ม "OK"

ถึงขั้นตอนนี้ถ้าในวินโดว์โปรแกรมเทลเน็ท มีเคอร์เซอร์รอการป้อนข้อมูลนั่นแสดงว่าโปรแกรมเทลเน็ทสามารถเปิดการเชื่อมต่อกับเซิร์ฟเวอร์ได้แล้ว สิ่งที่เซิร์ฟเวอร์ทำก็คือ รอว่าไคลเอนต์จะร้องขออะไรเข้ามา แต่หากเซิร์ฟเวอร์รอนาน จนถึงระยะเวลาที่กำหนดไว้ และพบว่ายังไม่มีการส่งข้อความร้องขอเข้ามาเซิร์ฟเวอร์จะปลดการเชื่อมต่อกับไคลเอนต์นั้นออกทันที เมื่อโปรแกรมเทลเน็ทรับทราบว่าถูกเซิร์ฟเวอร์ปลดการเชื่อมต่อแล้ว จะแสดงไดอะล็อกบ็อกซ์แจ้งว่า Connection to host lost.

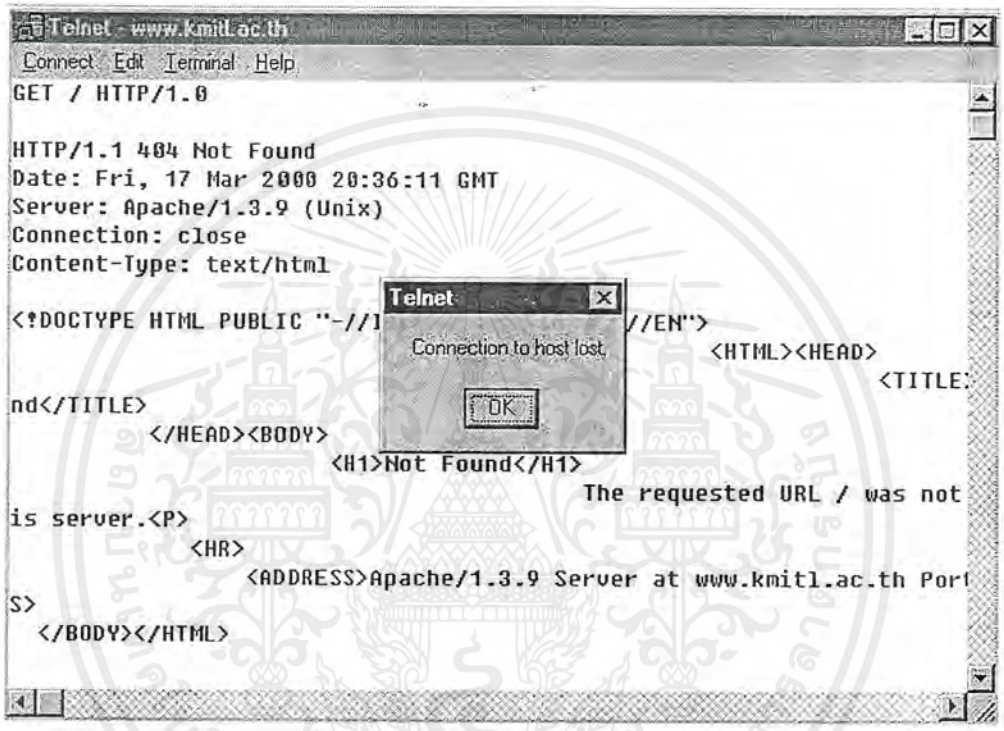
3.1.2 การทดลองการร้องขอจากเครื่องไคลเอนต์ไปยังเซิร์ฟเวอร์ด้วยเมธอด GET

ถ้าเชื่อมต่อกับเซิร์ฟเวอร์ได้แล้ว เราจะทดลองส่งข้อความร้องขอไฟล์ไปยังเซิร์ฟเวอร์ โดยพิมพ์ GET /HTTP/1.0 แล้วกดคีย์เอ็นเตอร์บนคีย์บอร์ด 2 ครั้ง การกด Enter ครั้งแรกเป็นการลงบรรทัดใหม่ และกดเอ็นเตอร์ อีกครั้งเป็นการสร้างบรรทัดว่างๆ ตามกฎเกณฑ์ของเฮดเดอร์ HTTP

ข้อความร้องขอเราใช้เมธอด GET แต่ระบุไดเรกทอรีและไฟล์เป็นเพียงเครื่องหมาย / ดังนั้นเมื่อโปรแกรมเว็บเซิร์ฟเวอร์ได้รับข้อความร้องขอนี้แล้ว จะส่งไฟล์ HTML ที่เป็นไฟล์โฮมเพจมาให้ ซึ่งอาจได้แก่ index.html , index.htm, default.htm หรืออาจเป็นไฟล์อื่นก็ได้ ขึ้นอยู่กับการเซ็ตที่เซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อความตอบสนองในส่วนของเฮดเดอร์ HTTP บรรทัดแรกบอกไว้ว่า HTTP/1.0 200 OK แสดงว่าร้องขอสำเร็จ จากนั้นต่อด้วยเฮดเดอร์รายละเอียดต่างๆ ของเซิร์ฟเวอร์และไฟล์ เมื่อหมดข้อความส่วนเฮดเดอร์ก็จะเป็นบรรทัดว่าง 1 บรรทัด แล้วตามด้วยบล็อกข้อมูล เมื่อโปรแกรมเทลเน็ท ได้รับข้อมูลของไฟล์หมดแล้ว เซิร์ฟเวอร์จะตัดการเชื่อมต่อกับเทลเน็ทไปทันที โปรแกรมเทลเน็ทจึงแสดงไดอะล็อกบ็อกซ์แจ้งให้ผู้ใช้ทราบว่า Connection to host lost.



รูปที่ 3-3 แสดงเฮดเดอร์ HTTP ของข้อความตอบสนอง

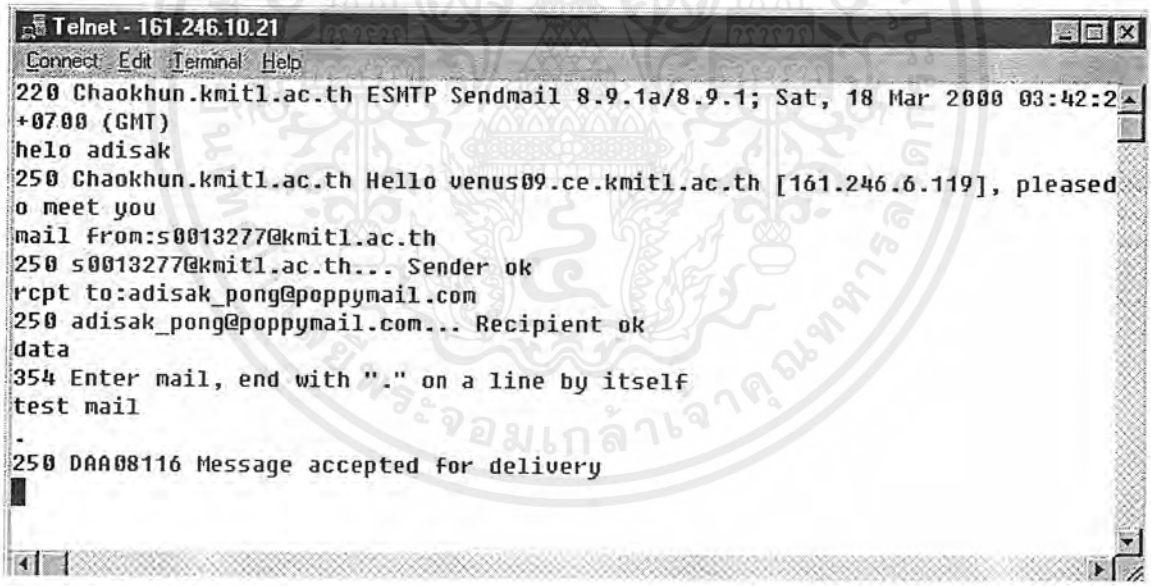
จากรูปที่ 3.3 แสดงการร้องขอจากเครื่องไคลเอนต์ด้วยโปรแกรมเทลเน็ท โดยใช้คำสั่ง GET / HTTP/1.0 แต่เนื่องจากในการพิมพ์ครั้งแรกเราใช้ตัวอักษรตัวพิมพ์เล็กแล้วจึงแบล็กสเปส (BackSpace) เพื่อแก้ไขเป็นตัวพิมพ์ใหญ่(ทำให้เครื่องเซิร์ฟเวอร์ไม่เข้าใจ) เป็นผลให้เครื่องเซิร์ฟเวอร์ตอบสนองกลับมาด้วยส่วนของเฮดเดอร์ที่แสดงความผิดพลาด โดยที่มีสถานะโค้ด 400 คือ เซิร์ฟเวอร์ไม่เข้าใจสิ่งที่ต้องการ และในส่วนของเอกสาร HTML ที่ส่งต่อท้ายมากับส่วนเฮดเดอร์จะเป็นข้อความแสดงความผิดพลาด และถ้าเราเปลี่ยนจากโปรแกรมเทลเน็ทเป็นโปรแกรมเบราเซอร์ เบราเซอร์จะทำการแสดงข้อมูลต่างๆที่ส่งมาตามรูปแบบของเอกสาร HTML

3.1.3 ทดลองโปรโตคอล SMTP ด้วยเทลเน็ท

ในการทดลองโปรโตคอล SMTP นี้เราจะใช้โปรแกรม telnet เช่นเดียวกับการทดลอง HTTP โดยการเชื่อมต่อไปที่ www.kmitl.ac.th เหมือนเดิม เพียงเปลี่ยนเป็นพอร์ต 25 เท่านั้น จากนั้นเครื่องเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

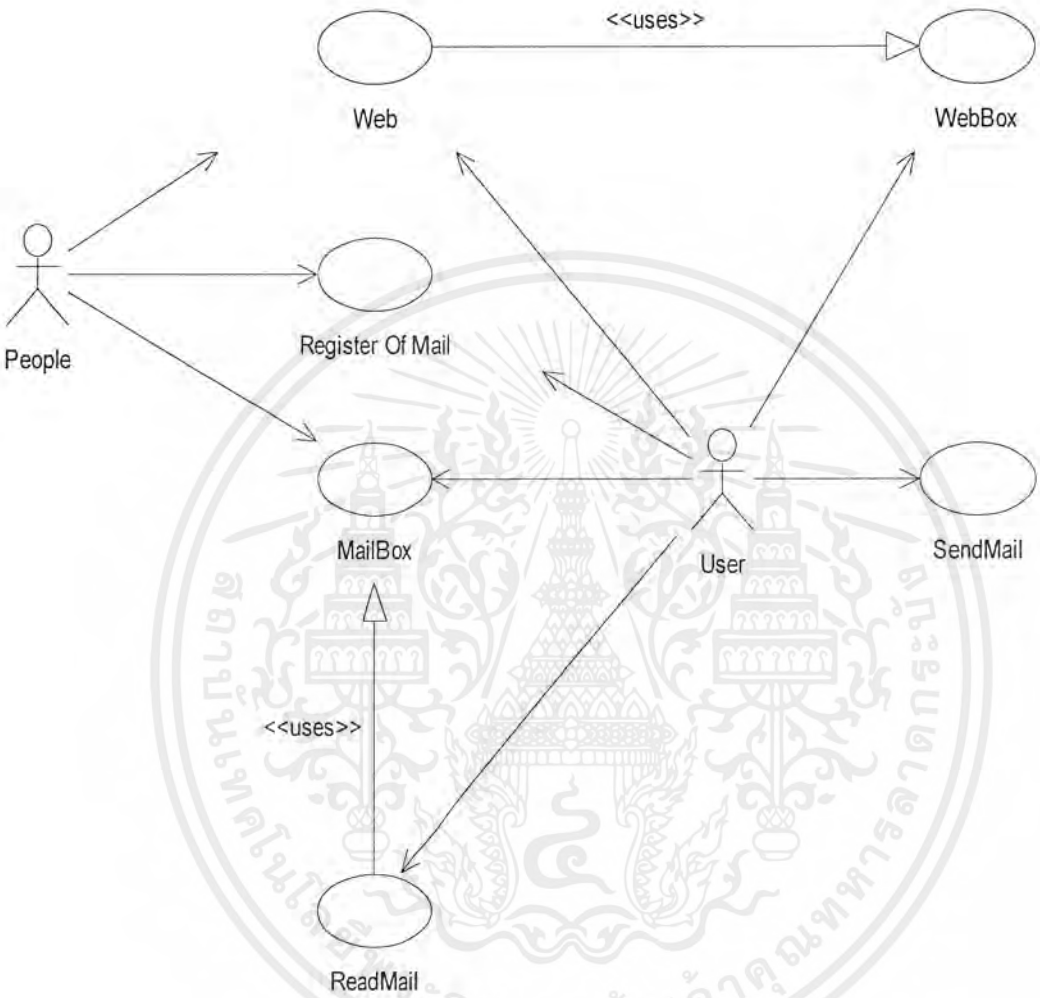
จะตอบกับมาด้วยรหัส 220 จากนั้นให้เราพิมพ์ helo เพื่อบอกว่าต้องการเริ่มการติดต่อกับเซิร์ฟเวอร์จาก นั้นเครื่องเซิร์ฟเวอร์จะตอบกับมาด้วยรหัส 250 จากนั้นให้เราพิมพ์คำสั่ง mail from:<reverse-path> ซึ่ง "reverse-path" คืออีเมล์แอดเดรสของผู้ส่ง เครื่องเซิร์ฟเวอร์จะตอบกลับมาด้วยรหัส 250 เพื่อบอกว่าอีเมล์แอดเดรสของผู้ส่งถูกต้อง (Sender ok) จากนั้นให้เราพิมพ์คำสั่ง rcpt to:<forword-path> ซึ่ง "forword-path" คืออีเมล์แอดเดรสของผู้รับ เครื่องเซิร์ฟเวอร์จะตอบกลับมาด้วยรหัส 250 เพื่อบอกว่าอีเมล์แอดเดรสของผู้รับถูกต้อง (Recipient ok) จากนั้นให้เราพิมพ์คำสั่ง "data" เพื่อบอกว่าส่วนถัดไปจะเป็นส่วนของข้อความที่จะส่ง เครื่องเซิร์ฟเวอร์จะตอบกับมาด้วยรหัส 354 เพื่อบอกว่าพร้อมที่จะรับข้อมูลที่จะส่ง ในการจบส่วนของข้อความให้จบด้วยเครื่องหมายดอด (.) จากนั้นให้เราพิมพ์ข้อความที่ต้องการส่ง และเมื่อต้องการจบส่วนของข้อความให้เรากดเอ็นเตอร์ เพื่อขึ้นบรรทัดใหม่จากนั้นให้เราพิมพ์ " . " เพื่อบอกให้รู้ว่าจบส่วนของข้อความที่จะส่งแล้ว จากนั้นเครื่องเซิร์ฟเวอร์จะตอบกลับมาด้วยรหัส 250 เพื่อบอกว่าการส่งจดหมายเสร็จเรียบร้อยแล้ว เราจะเห็นว่าจะมีการติดต่อกันระหว่างเครื่องไคลเอนต์กับเครื่องเซิร์ฟเวอร์ในการส่งเมล์นั้นเราต้องระบุว่าใครเป็นผู้ส่ง , ใครเป็นผู้รับ และส่วนของข้อมูลที่จะส่ง ซึ่งจะสิ้นสุดส่วนของข้อมูลด้วยเครื่องหมายดอด (.)



รูปที่ 3-4 แสดงการใช้โปรแกรมเทลเน็ทในการส่งเมล์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

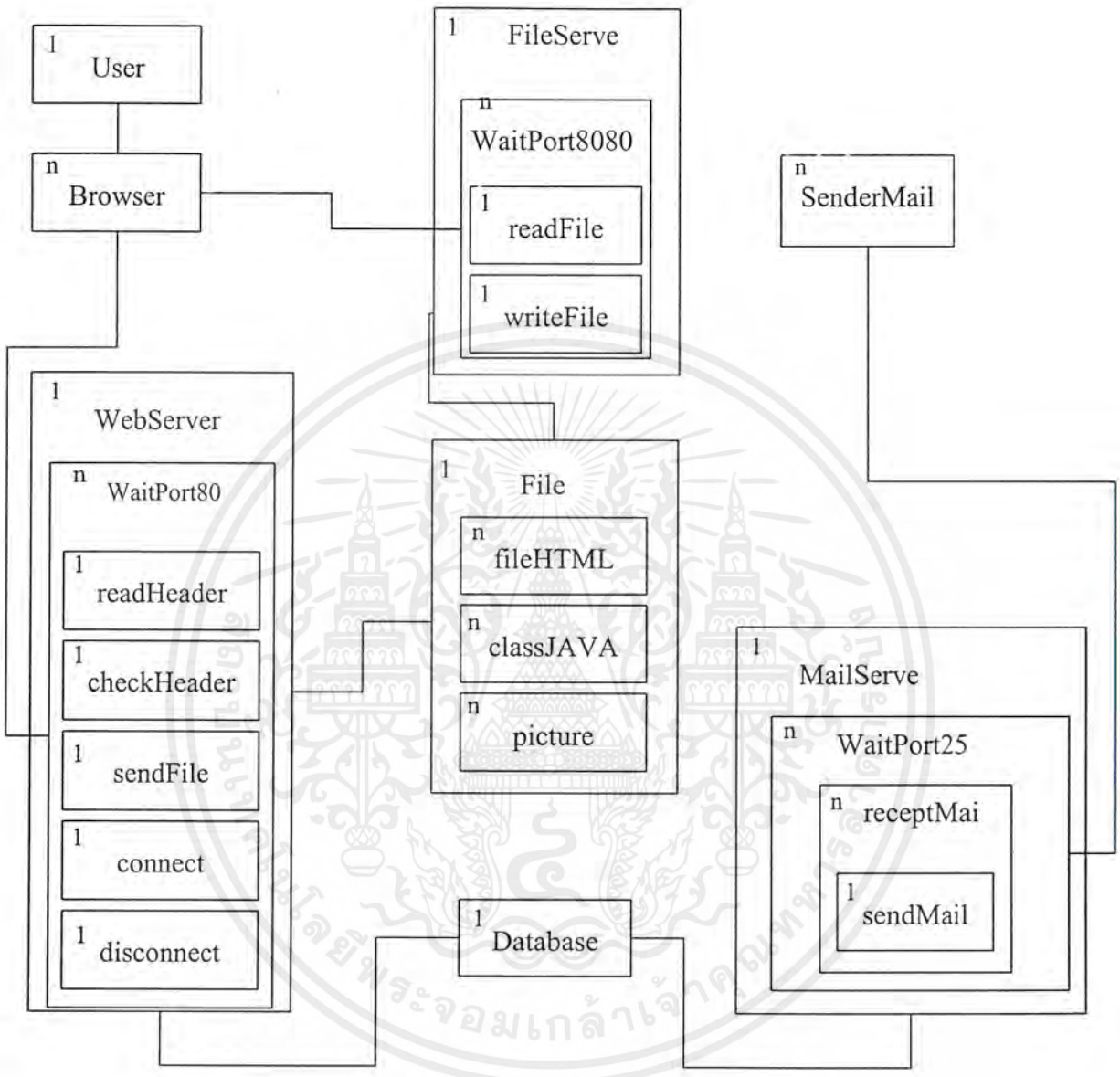
3.2 ออกแบบเว็บเซิร์ฟเวอร์และเมล์เซิร์เวอร์โดยใช้ยูเอ็มเอล (UML)



รูปที่ 3-5 แสดง Use Case Diagramของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์

จากรูปที่ 3-5 เป็นการแสดงการใช้งานของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์ โดยมีผู้ใช้หลักๆอยู่ 2 กลุ่มคือกลุ่มของผู้ที่ยังไม่ได้ลงทะเบียน (People) จะเข้ามาใช้งานโดยอาจจะเข้ามาใช้บริการเว็บเซิร์ฟเวอร์ หรืออาจจะส่งเมล์มาให้แก่สมาชิกที่ได้ทำการลงทะเบียนไว้แล้ว และอาจจะเข้ามาสมัครเป็นสมาชิกได้ ส่วนกลุ่มที่สองคือกลุ่มของผู้ที่ได้ลงทะเบียนเป็นสมาชิกแล้ว โดยผู้ใช้กลุ่มนี้สามารถล็อกอินเข้ามาใช้บริการส่งเมล์อ่านเมล์หรือนำโฮมเพจมาเก็บไว้ที่เครื่องเซิร์ฟเวอร์ได้

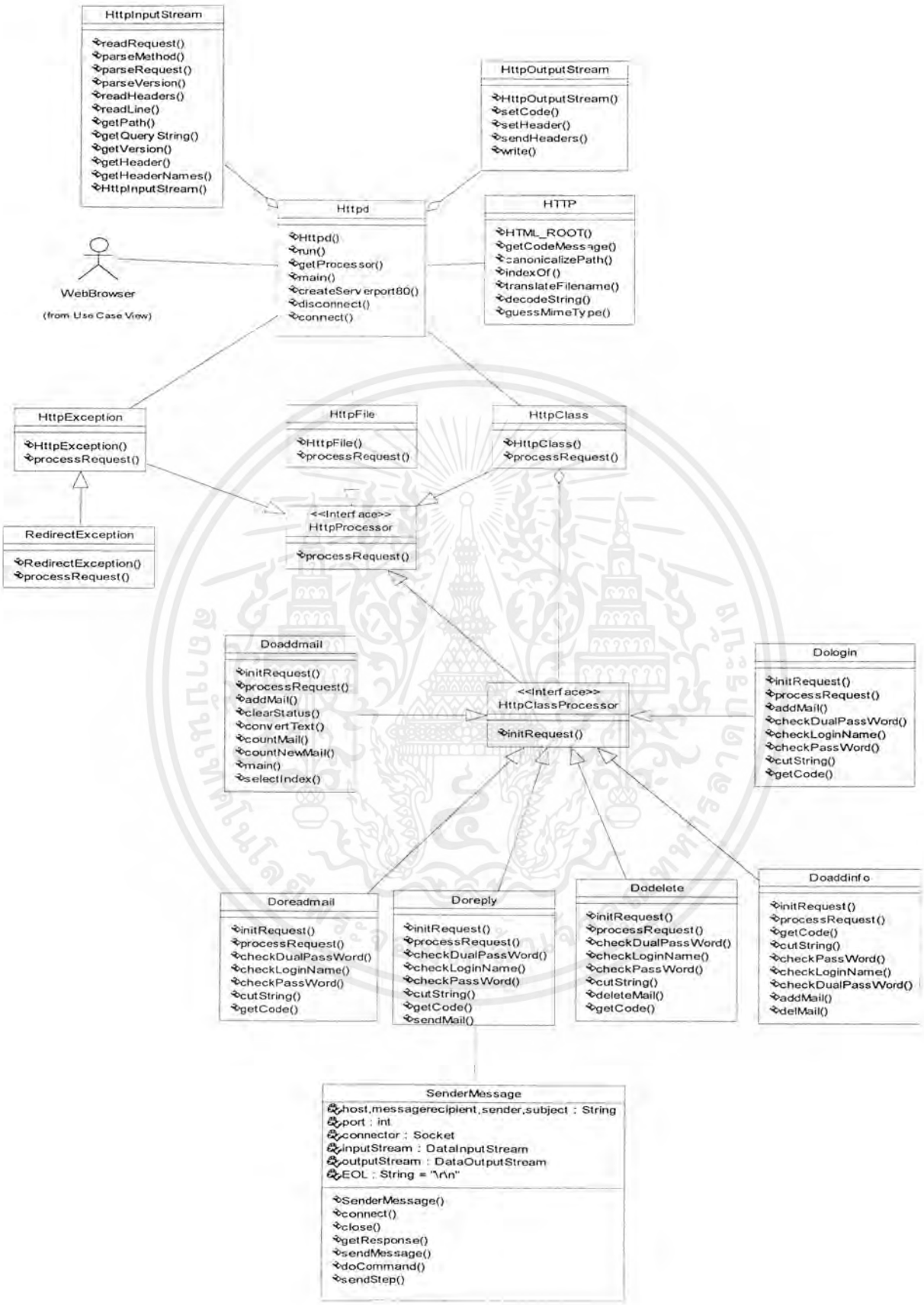
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-6 แสดง Object Diagramของเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์

จากโครงสร้างของออบเจ็กไดอะแกรมของระบบประกอบด้วยเซิร์ฟเวอร์ 3 ตัวได้แก่เว็บเซิร์ฟเวอร์ที่ให้บริการเกี่ยวกับเว็บจะเปิดให้บริการที่พอร์ต 80 และจะมีการทำงานเป็นเธรด เพื่อรองรับการติดต่อจากผู้ใช้บริการหลายคนในเวลาเดียวกัน ส่วนที่สองคือเมล์เซิร์ฟเวอร์ที่ให้บริการเกี่ยวกับเว็บจะเปิดให้บริการที่พอร์ต 25 และจะมีการทำงานเป็นเธรด เพื่อรองรับการติดต่อจากผู้ใช้บริการหลายคนในเวลาเดียวกัน และส่วนสุดท้ายคือไฟล์เซิร์ฟเวอร์ที่ให้บริการเกี่ยวกับเว็บจะเปิดให้บริการที่พอร์ต 8080 เพื่อรอรับไฟล์เอกสารและไฟล์รูปภาพที่ผู้ใช้ต้องการนำมาเก็บไว้ที่เซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-7 แสดง Class Diagram ของเว็บเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.1 ส่วนของคลาสและไฟล์ซีจีไอที่ถูกเรียกใช้

การออกแบบเว็บเซิร์ฟเวอร์จะมีการออกแบบดังรูปที่ 3-7

Httpd เป็นส่วนสำคัญในการออกแบบ webserver โดยเป็นส่วนที่มีการรอรับการร้องขอจากไคลเอ็นท์ และเมื่อมีการร้องขอเข้ามาจากไคลเอ็นท์จะมีการทำงานเป็น thread เพื่อตอบสนองการร้องรอนั้น และสุดท้ายจะทำการตัดการเชื่อมต่อ

interface HttpProcessor การตอบสนองการร้องขอจากเครื่องไคลเอ็นท์จะถูกอ้างมาที่ interface HttpProcessor ในส่วนของ interface นี้จะทำการเรียก processor ที่จะตอบสนองไปแก่ไคลเอ็นต์อีกทีหนึ่ง

HttpInputStream ในส่วนของ class นี้จะทำหน้าที่ในการอ่านการร้องขอจากเครื่องไคลเอ็นท์รวมถึงการ parsing ด้วย

HttpOutputStream ในส่วนของ class นี้จะทำหน้าที่ในการตอบสนองกลับไปยังเครื่องไคลเอ็นท์ และทำการสร้างส่วนของ header เพื่อส่งกลับไปยังไคลเอ็นท์

HttpFile จากรูปจะเห็นว่า HttpFile เป็น class ที่สนับสนุนการทำงานของส่วน HttpProcessor interface ทำหน้าที่ในการให้บริการเอกสารชนิด static html แก่เครื่องไคลเอ็นท์ที่ร้องขอเข้ามา

HttpClass เป็น class ที่สนับสนุนการทำงานของส่วน HttpProcessor interface เช่นกัน เป็นส่วนที่อำนวนความสะดวกในการที่จะติดต่อกับ cgi ที่เขียนขึ้นด้วยภาษา java

HttpClassProcessor เป็นส่วนที่ทำหน้าที่ในการให้บริการการทำงานของ CGI ที่เขียนขึ้นด้วยภาษา java ที่เป็นไฟล์นามสกุลดอต class ทำให้เว็บเซิร์ฟเวอร์ที่เราสร้างขึ้นสามารถตอบสนองเอกสารชนิด dynamic html ได้

HttpException เป็นส่วนที่จะแจ้งความผิดพลาดที่เกิดขึ้นส่งไปให้ส่วนของไคลเอ็นท์

RedirectException คลาส RedirectException เป็นเป็นคลาสที่ขยายมาจากคลาส HttpException เป็นคลาสที่ทำหน้าที่ redirect ข้อความกลับไปยังเครื่องไคลเอ็นท์

HTTP เป็น class ที่ทำหน้าที่กำหนดค่าคงที่ต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไฟล์ซีจีไอที่เขียนขึ้นด้วยภาษาจาวา

doaddinfo.java เป็นไฟล์ cgi ที่ทำหน้าที่ในการเพิ่มผู้ใช้เข้าไปในดาต้าเบส เพราะในส่วนของเมล์เซิร์ฟเวอร์นั้นผู้ที่จะเข้ามาใช้งานต้องทำการลงทะเบียนก่อนเมื่อผู้ใช้เลือกที่จะลงทะเบียนจะมีแบบฟอร์มให้ผู้ใช้กรอกข้อมูล เช่นชื่อล็อกอิน , รหัสผ่าน , ที่อยู่ เป็นต้น เมื่อผู้ใช้กรอกแบบฟอร์มเสร็จแล้วและทำการส่งแบบฟอร์ม ข้อมูลต่างๆที่ผู้ใช้กรอกจะถูกส่งมาที่ไฟล์นี้ ซึ่งต้องทำการแยกข้อมูลต่างๆที่ส่งมาเก็บไว้ในตัวแปรก่อนจากนั้นจึงทำการติดต่อกับดาต้าเบสเพื่อนำข้อมูลที่ได้มาเก็บลงในดาต้าเบส

dodelete.java เป็นไฟล์ cgi ที่ทำหน้าที่ในการลบไฟล์ที่ผู้ใช้ต้องการลบทิ้งไป โดยถ้าผู้ใช้ต้องการลบไฟล์ใดทิ้ง ผู้ใช้สามารถทำได้โดยการเช็คเครื่องหมายถูกที่ check box และเมื่อผู้ใช้กดปุ่มลบจดหมายข้อมูลจะถูกส่งมาที่ไฟล์นี้ โดยข้อมูลที่ส่งมาคือชื่อล็อกอินและอินเด็กซ์ที่ต้องการลบซึ่งสามารถมีได้หลายอินเด็กซ์ เราต้องทำการวนลูปลบข้อมูลในดาต้าเบสโดยเราจะให้ชื่อล็อกอินและอินเด็กซ์ที่ส่งมาเป็นตัวกำหนดว่าจะลบจดหมายฉบับใดออกจากดาต้าเบส เมื่อลบจดหมายทิ้งไปแล้วจะทำการแสดงผลจำนวนจดหมายทั้งหมดอีกครั้งที่มีภายหลังจากที่ได้ลบไปแล้ว

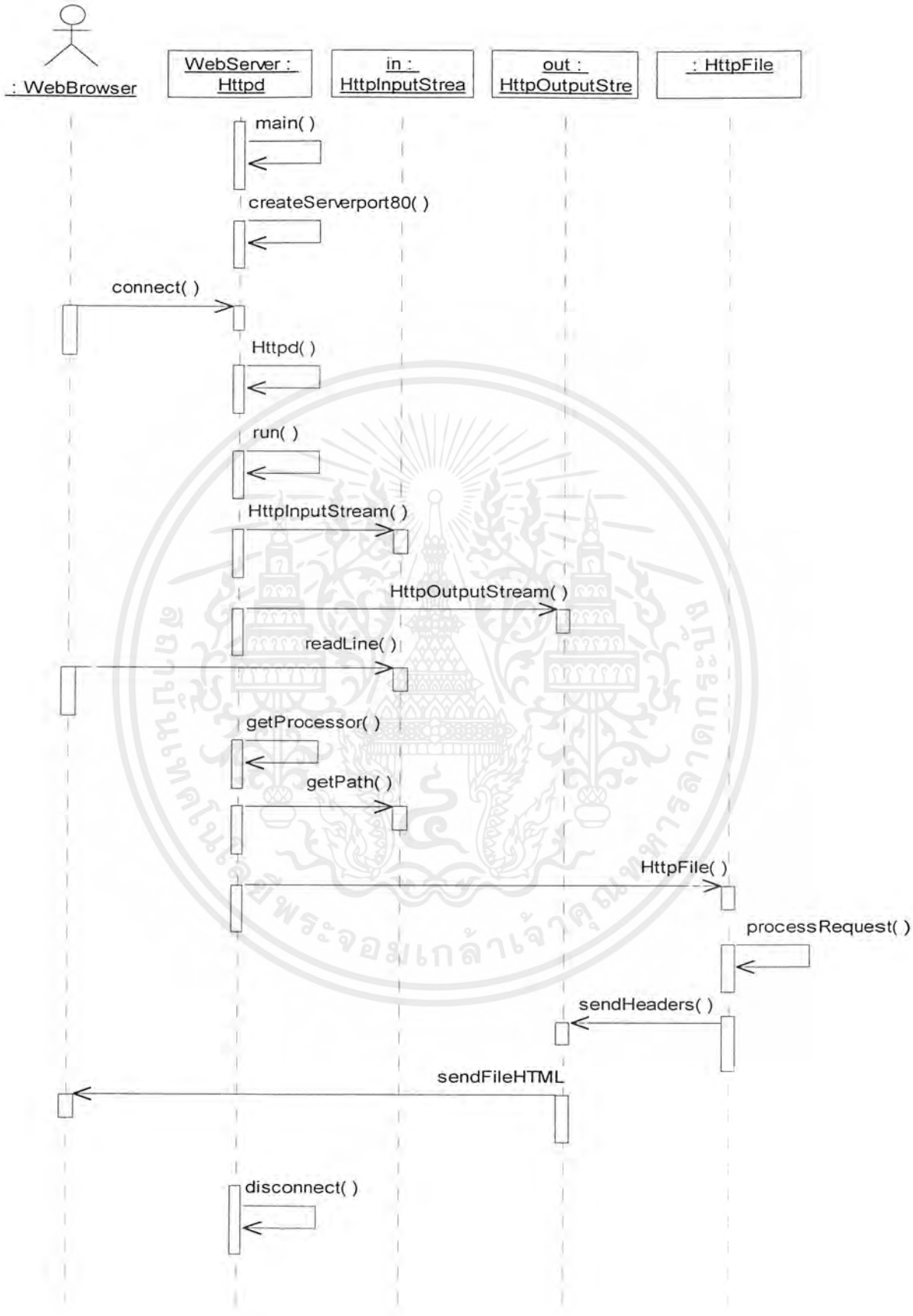
dologin.java เป็นไฟล์ cgi ที่จะถูกเรียกใช้เมื่อผู้ใช้ป้อนชื่อล็อกอินและรหัสผ่านจากนั้นกดปุ่ม Login ข้อมูลจะถูกส่งมาที่ไฟล์นี้แต่เราจะไม่เห็นข้อมูลปรากฏอยู่ที่ URL เพราะเราส่งด้วยวิธี POST ซึ่งเราต้องทำการแยกชื่อล็อกอินและรหัสผ่านเก็บไว้ในตัวแปรก่อน จากนั้นเราจะทำการติดต่อกับดาต้าเบสเพื่อค้นหาว่าชื่อล็อกอินที่ผู้ใช้พิมพ์เข้ามามีในดาต้าเบสหรือป่าว ถ้าไม่มีก็จะส่งข้อความแสดงการผิดพลาดกลับไปให้แก่ผู้ใช้ที่ล็อกอินเข้ามาทราบ ถ้ามีเราจะทำการอ่านค่ารหัสผ่านที่เก็บอยู่ในดาต้าเบสมาทำการเปรียบเทียบกับรหัสผ่านที่เรารับเข้ามาถ้าไม่เหมือนกันเราจะส่งข้อความแสดงการผิดพลาดกลับไปให้แก่ผู้ใช้ แต่ถ้ารหัสผ่านเหมือนกันเราจะทำการติดต่อกับดาต้าเบสอีกครั้งเพื่อทำการอ่านรายชื่อผู้ที่ส่งจดหมายมาหาเราและส่วนของหัวข้อเพื่อมาแสดงให้ผู้ใช้รู้ว่ามีจดหมายที่ผู้อื่นส่งมาให้ทั้งหมดกี่ฉบับ และผู้ใช้สามารถเข้าไปอ่านจดหมายได้โดยกดคลิกที่หัวข้อจดหมายที่ต้องการอ่าน

doreadmail.java เป็นไฟล์ cgi ที่ทำหน้าที่ในการอ่านเมล์ที่มีผู้ใช้ส่งมาให้เรา โดยมีการทำงานต่อเนื่องมาจากส่วนของ dologin.java โดยเมื่อผู้ใช้คลิกที่หัวข้อจดหมายที่มีผู้อื่นส่งมาให้ (ส่วนของ dologin.java) การทำงานจะถูกส่งต่อให้กับ doreadmail.java โดยพารามิเตอร์ที่ส่งมาด้วยก็คือชื่อล็อกอินและอินเด็กซ์ของจดหมายที่จะอ่าน โดยการทำงานในส่วนของไฟล์นี้จะทำการนำชื่อล็อกอินและอินเด็กซ์เป็นตัวเลือกจดหมายที่อยู่ในดาต้าเบสมาแสดงแก่ผู้ใช้

doreply.java เป็นไฟล์ที่ทำหน้าที่ในการส่งจดหมายออกไปให้ผู้อื่น โดยจะมีฟอร์มให้ผู้ใช้กรอกและเมื่อกรอกเสร็จแล้ว ทำการกดปุ่มส่งจดหมาย จดหมายจะถูกส่งออกไปจากนั้นจะแสดงข้อความว่าการส่งจดหมายเสร็จเรียบร้อยแล้ว

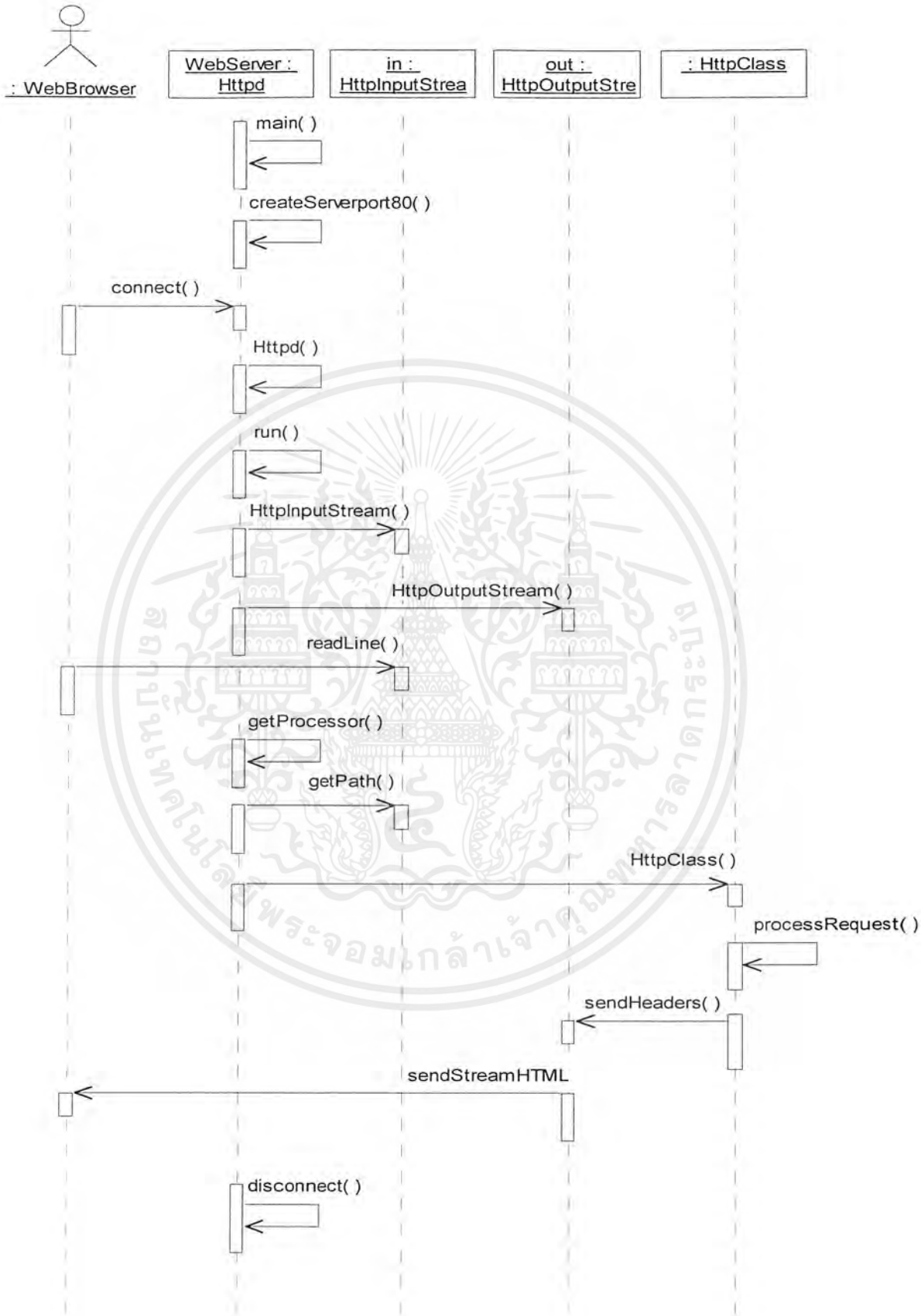
doaddmail.java เป็นส่วนที่ใช้ติดต่อกับฐานข้อมูล

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



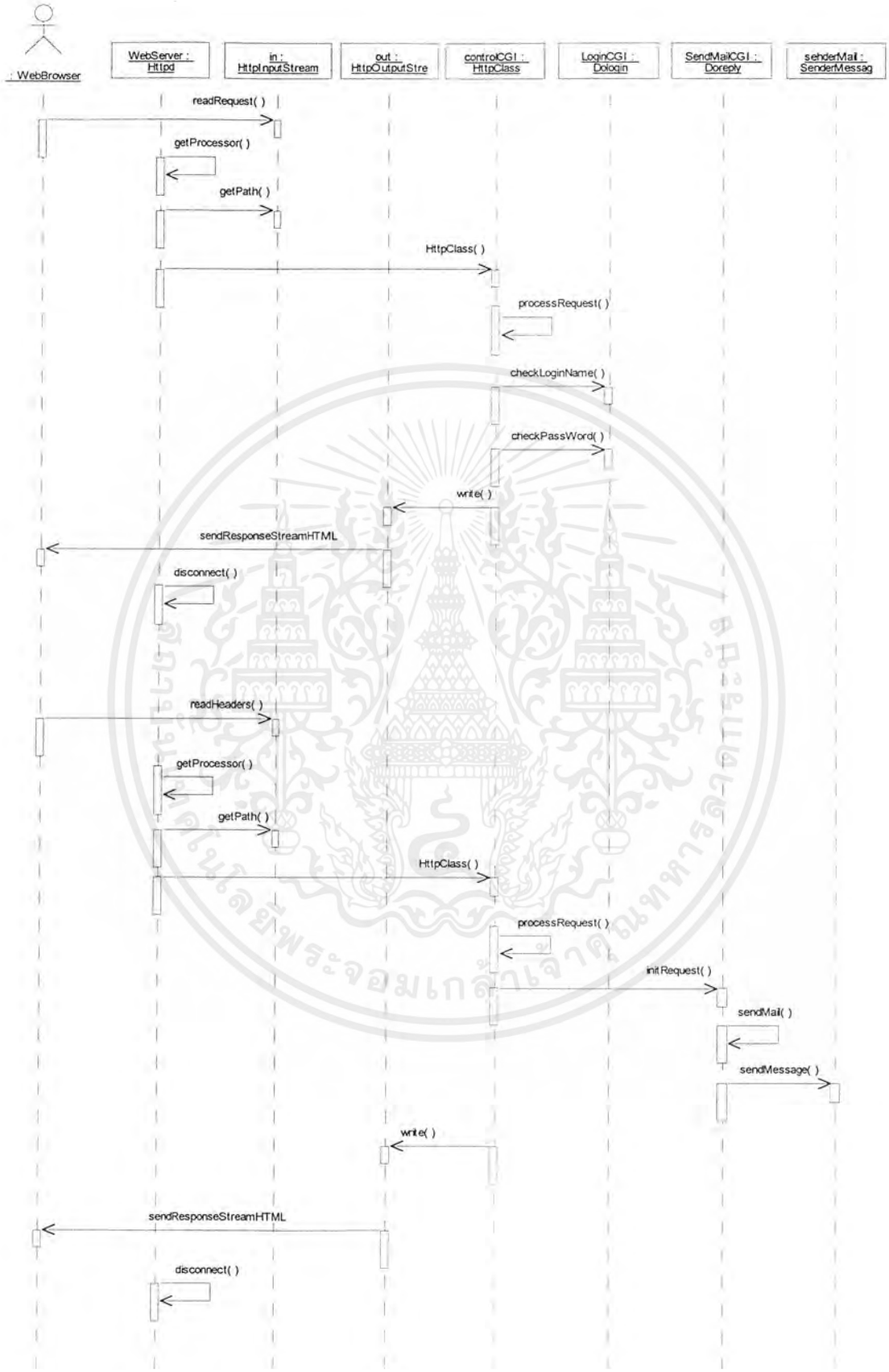
รูปที่ 3-8 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์(ให้บริการ static html)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



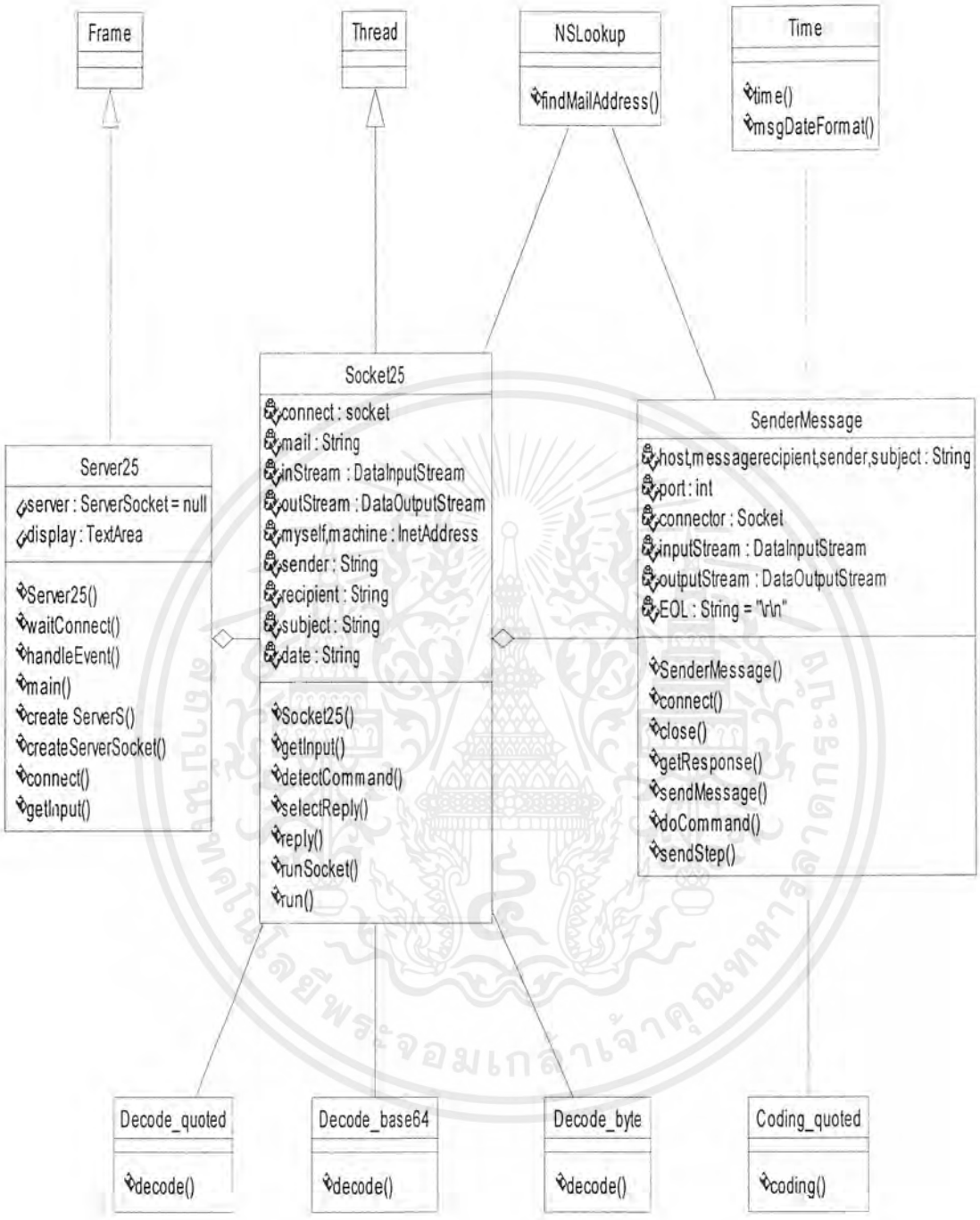
รูปที่ 3-9 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์ (ให้บริการ dynamic html)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-10 แสดง Sequence Diagram ของเว็บเซิร์ฟเวอร์ในการส่งเมล์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-11 แสดง Class Diagramของเมล์เซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากการขั้นตอนการทำงานของโปรโตคอล SMTP ที่ได้กล่าวถึงในข้างต้นได้นำมาใช้ในการออกแบบโปรแกรมของเมล์เซิร์ฟเวอร์ ซึ่งเขียนเป็นโครงสร้างของโปรแกรมได้ดังรูปที่ 3-11

Server25

การทำงานของโปรแกรมจะมีคลาสเซิร์ฟเวอร์25เป็นจุดศูนย์กลางเป็นจุดเริ่มต้นของโปรแกรมเมื่อรันโปรแกรม จะทำการสร้างเซิร์ฟเวอร์ซ็อคเก็ตที่ พอร์ต 25 ซึ่งเป็น พอร์ต ที่ใช้เปิดรับ Mail ตามมาตราฐานของโปรโตคอล SMTP และคลาสเซิร์ฟเวอร์25 จะทำการดูแลการเข้ามาขอเชื่อมต่อจากเมล์เซิร์ฟเวอร์ อื่นๆที่จะทำการส่งจดหมายอิเล็กทรอนิกส์มาให้ผู้ที่มีตู้รับจดหมาย เมื่อมีการเชื่อมต่อกับเมล์เซิร์ฟเวอร์อื่นคลาสเซิร์ฟเวอร์25 จะทำการสร้างซ็อคเก็ต ที่ใช้ในการเชื่อมต่อขึ้นมาเพื่อเป็นตัวกลางรับส่งข้อความซึ่งในเวลาหนึ่งอาจมีการเชื่อมกับเมล์เซิร์ฟเวอร์ อื่นๆหลายๆเครื่องพร้อมๆกัน จึงทำการออกแบบการจัดการโต้ตอบกับเมล์เซิร์ฟเวอร์ ต่างๆให้เป็นการทำงานที่เป็นเธรด(Thread) ซึ่งการทำงานดังกล่าวได้ทำการออกแบบเป็นคลาสซ็อคเก็ต25 ดังนั้นเซิร์ฟเวอร์25 จะทำการสร้างออบเจ็คของคลาสซ็อคเก็ต25 โดยส่งซ็อคเก็ตที่สร้างมาให้กับออบเจ็คของคลาสซ็อคเก็ต25 ทำการรับช่วงในการทำการโต้ตอบกับคำสั่งต่างๆในขั้นตอนการรับจดหมายอิเล็กทรอนิกส์ของโปรโตคอล SMTP ดังที่กล่าวไปแล้วในช่วงต้น และจะทำการ วนกลับไปรอรับการเชื่อมต่อจากเมล์เซิร์ฟเวอร์อื่นๆต่อไป

Socket25

เมื่อออบเจ็คของคลาสซ็อคเก็ต25 ที่ถูกสร้างโดยเซิร์ฟเวอร์25 โดยมีพารามิเตอร์เป็นซ็อคเก็ตที่เมล์เซิร์ฟเวอร์ สร้างขึ้นเช่นกัน เมื่อเริ่มต้นโปรแกรมจะทำการสร้าง DataInputStream ใช้ในการรับข้อมูลเข้ามาจากเมล์เซิร์ฟเวอร์ ที่มาเชื่อมต่อ และสร้าง DataOutputStream ใช้ในการส่งข้อมูลออกไปให้กับเมล์เซิร์ฟเวอร์ ที่มาเชื่อมต่อ จากนั้นก็จะทำการโต้ตอบกันกันดังขั้นตอนต่างๆ ของมาตราฐาน SMTP มี Methrod getInput() เป็นตัวจัดการในการวนลูปอ่านข้อความที่ส่งมาจาก MailServer ที่ติดเชื่อมต่ออยู่ผ่าน DataInputStream เข้ามาเป็น Integer ทีละตัวแล้วเช็คว่าเป็นอักษรภาษาไทยหรือป่าวถ้าใช่ก็แปลงเป็นอักษรภาษาไทยถ้าไม่ก็แปลงเป็นตัวอักษร โดยคลาส decode_byte ซึ่งออกจากลูปเมื่ออ่านเข้ามาครบบรรทัดแล้วส่งขอมูลหนึ่งบรรทัดดังกล่าวไปทำการตีความหมายและจะได้ข้อมูลโต้ตอบกับให้เมล์เซิร์ฟเวอร์ที่ติดต่ออยู่แล้วจึงส่งข้อมูลดังกล่าวโดย Methrod reply(String) ซึ่งส่งออกทาง DataOutputStream อีกที

declose_quoted

จะเป็นคลาสที่ถูกเรียกใช้ในกรณีที่เมล์เซิร์ฟเวอร์ที่มาทำการเชื่อมต่อเพื่อส่งจดหมายอิเล็กทรอนิกส์มีการเข้ารหัสในส่วนของข้อความของจดหมายแบบ quoted-printable เมื่อซ็อคเก็ต25 ได้รับข้อความครบแล้วก็จะทำการส่งข้อความในส่วนของเนื้อความจดหมาย ไปทำการถอดรหัสเป็นเนื้อความปกติ

Decode_base64

จะเป็นคลาสที่ถูกเรียกใช้ในกรณีที่เมล์เซิร์ฟเวอร์ที่มาทำการเชื่อมต่อเพื่อส่งจดหมายอิเล็กทรอนิกส์มีการเข้ารหัสในส่วนของข้อความของจดหมายแบบ base64 เมื่อซ็อคเก็ต25 ได้รับข้อความครบแล้วก็จะทำการส่งข้อความในส่วนของเนื้อความจดหมาย ไปทำการถอดรหัสเป็นเนื้อความปกติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

NSLookup

จะเป็นคลาสที่ถูกเรียกใช้ในส่วนของการทำการตรวจสอบข้อมูลที่เป็นอีเมล์แอดเดรสไม่ว่าจะเป็นของผู้รับหรือส่งว่ามีโฮสท์ ที่ทำหน้าที่เป็นเมล์เซิร์ฟเวอร์ คือโฮสท์ ที่มีชื่อว่าอะไรมีอยู่จริงหรือไม่ เพื่อประโยชน์ในด้านต่างๆ เช่นอีเมล์แอดเดสรของผู้ส่งก็เป็นต้งบอกให้ผู้รับทราบว่าใครเป็นเขียนจดหมายส่งมาให้และเป็นตัวบอกที่อยู่ที่จะทำการส่งจดหมายไปถึงผู้ส่งได้ ส่วนอีเมล์แอดเดรสของผู้รับก็จะเป็นตัวบอกว่าจดหมายนี้ส่งถึงใครถ้าหากโฮสท์ของอีเมล์แอดเดรสของผู้รับไม่ใช่โฮสท์ของเราก็แสดงว่าซ๊อคเก็ต25จะต้องทำการส่งจดหมายดังกล่าวไปให้กับโฮสท์ ที่เป็นเมล์เซิร์ฟเวอร์ของอีเมล์แอดเดรสของผู้รับนั้นผ่านพอร์ต 25 ของโฮสท์ที่เป็นเมล์เซิร์ฟเวอร์ของอีเมล์แอดเดรสของผู้รับนั้น

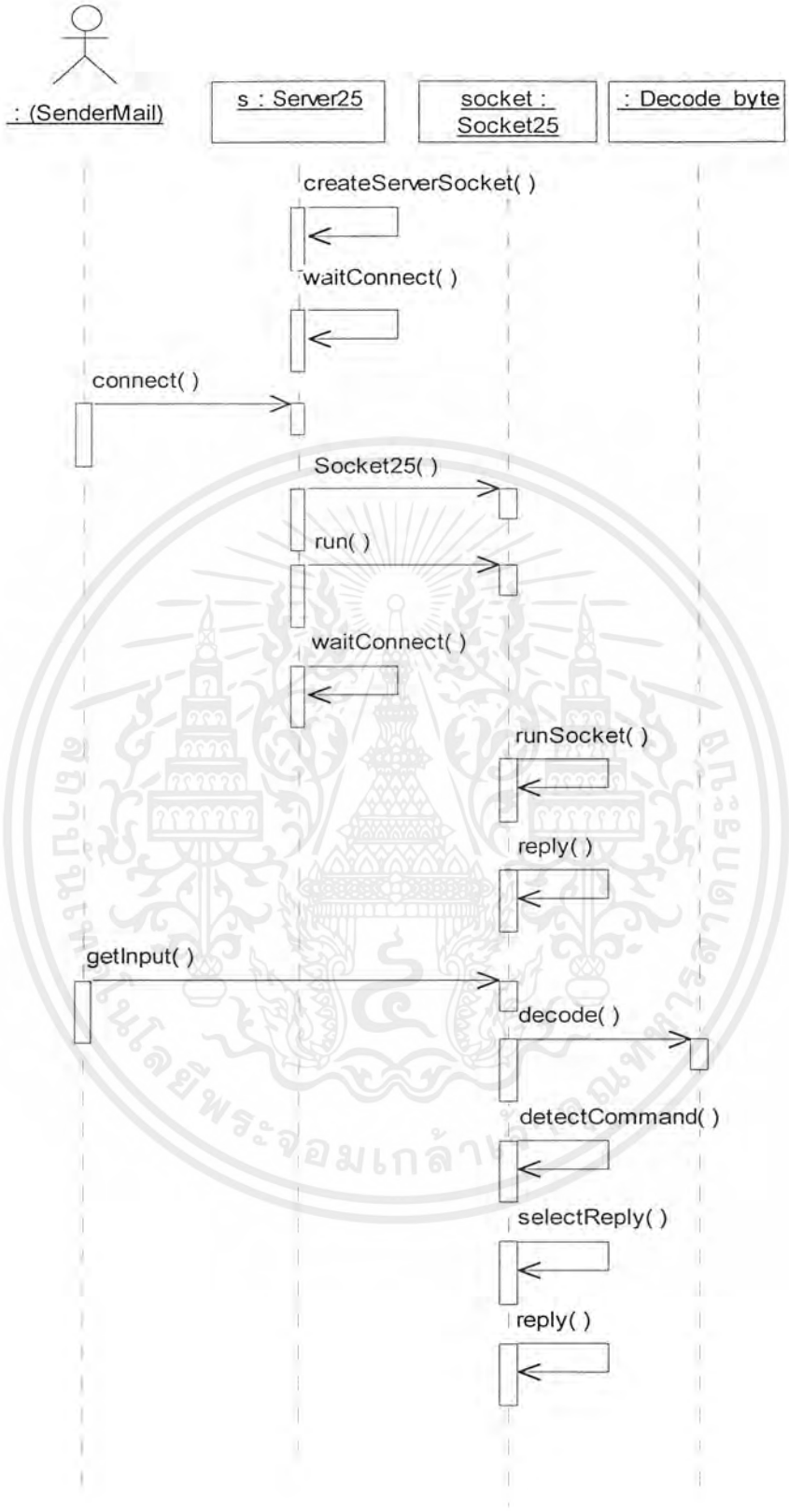
SenderMessage

จะถูกสร้างออบเจ็คขึ้นในคลาสซ๊อคเก็ต25 ในกรณีที่มีการเชื่อมต่อกับเมล์เซิร์ฟเวอร์จากภายนอกแล้วมีการส่งจดหมายอิเล็กทรอนิกส์มาแต่โฮสท์ของเมล์เซิร์ฟเวอร์ไม่ใช่โฮสท์ของเมล์เซิร์ฟเวอร์ที่ทำงานอยู่ซ๊อคเก็ต25 ก็จะส่งข้อมูลในจดหมายได้แก่อีเมล์แอดเดรสของผู้รับและผู้ส่งชื่อหัวข้อจดหมาย และตัวเนื้อจดหมายซึ่งผ่านการเข้ารหัสแบบ quoted-printable ซึ่งการทำงานจะสัมพันธ์กับการทำงานของคลาสซ๊อคเก็ต25 เพราะเปรียบเป็นตัวเมล์เซิร์ฟเวอร์ ที่เข้ามาเชื่อมต่อเพื่อทำการส่งจดหมายให้ผู้รับที่มีอีเมล์แอดเดรสอยู่ที่เครื่องดังกล่าว เริ่มตั้งแต่ทำการสร้างซ๊อคเก็ตเชื่อมต่อกับพอร์ตเมล์เซิร์ฟเวอร์ปลายทาง สร้างDataInputStream และDataOutputStream เพื่อเป็นตัวรับข้อมูลเข้าและส่งข้อมูลออกตามลำดับแล้วทำการส่งข้อมูลโต้ตอบกับเมล์เซิร์ฟเวอร์

coding_quoted

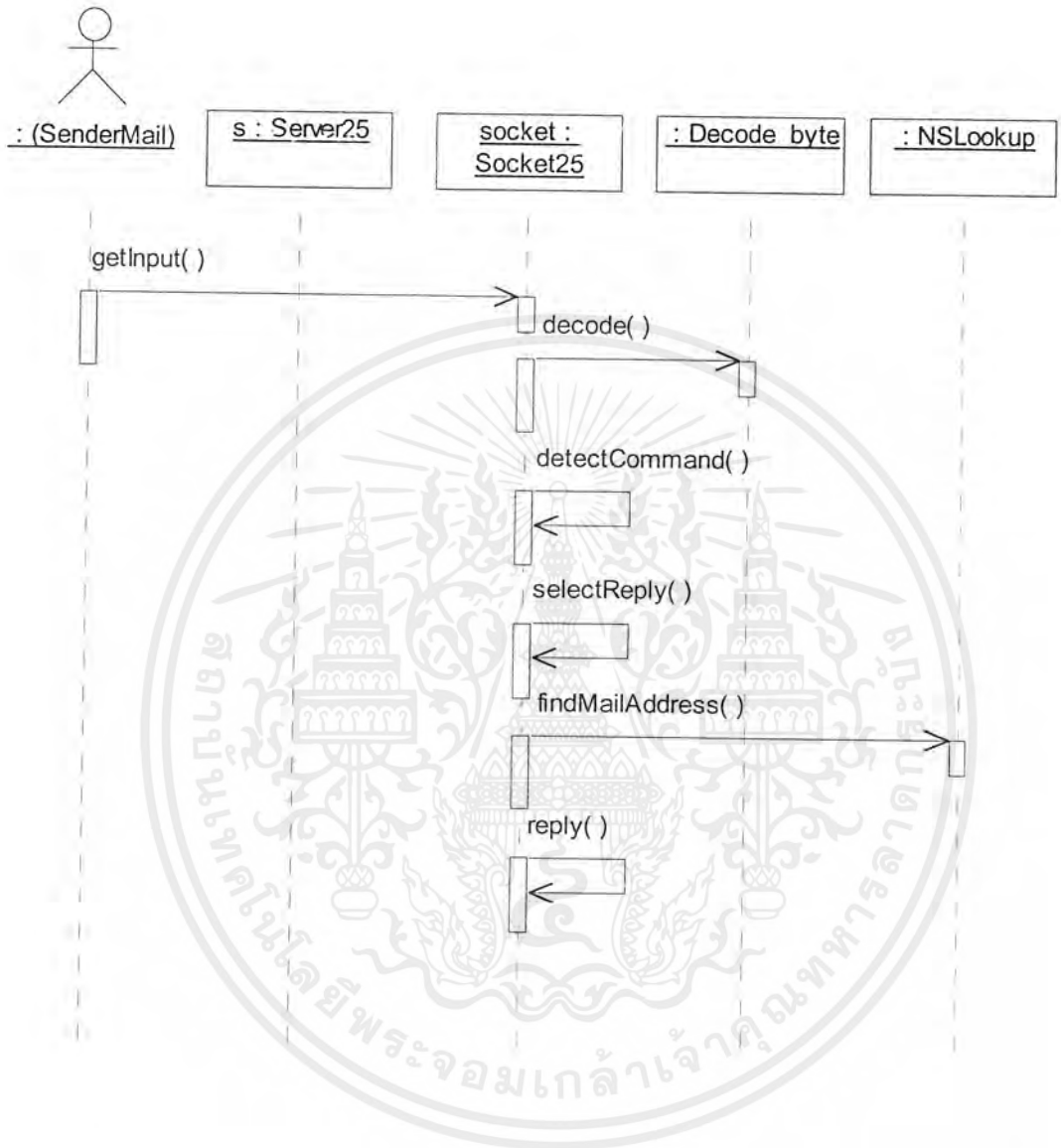
จะถูกเรียกจากคลาส SenderMessage ในการทำการเข้ารหัสเนื้อความในจดหมายเป็นแบบ quoted-printable จะมีการทำงานตรงกันข้ามกับคลาส decode_quoted

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



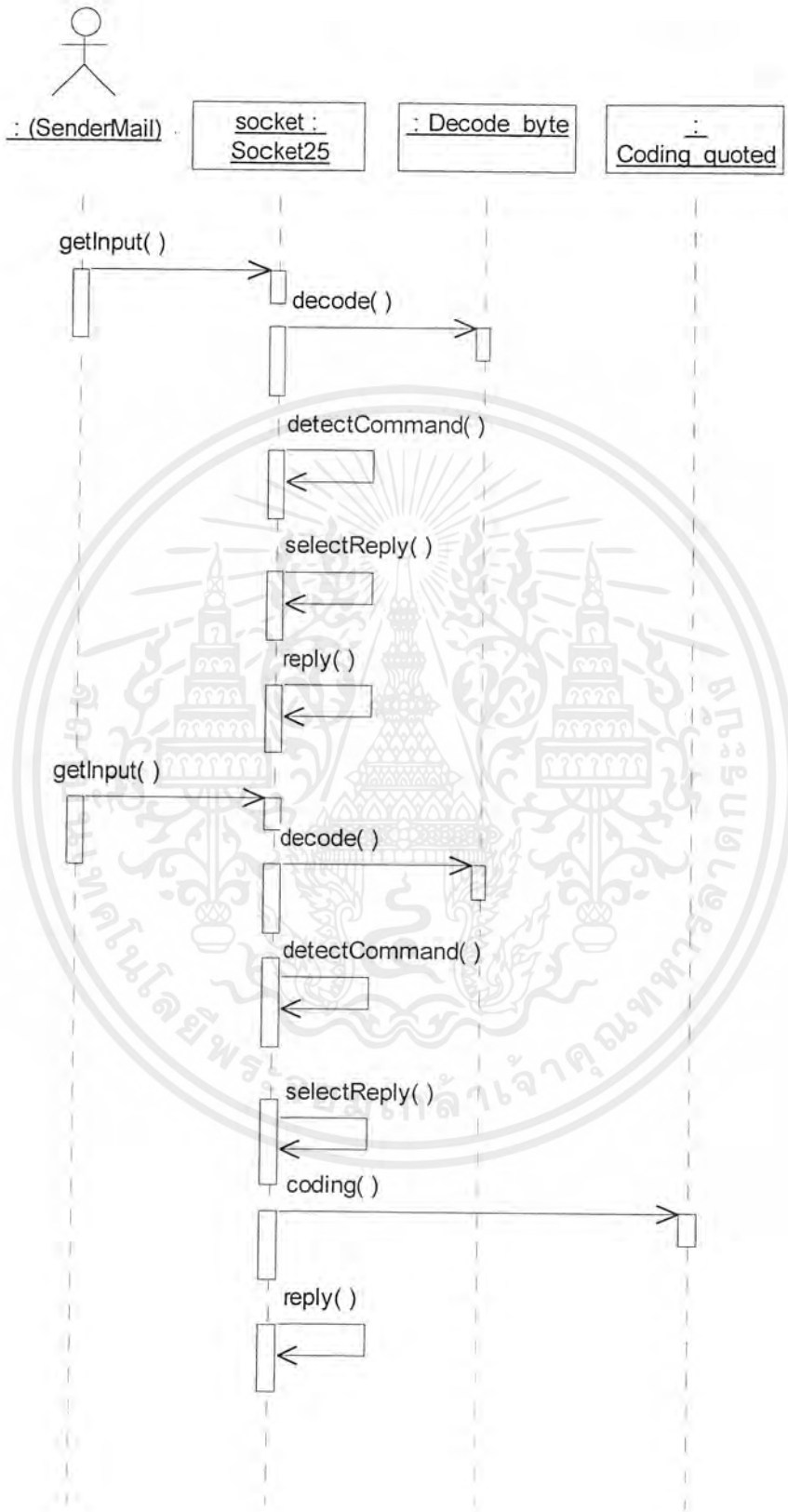
รูปที่ 3-12 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์(กรณีเริ่มทำงานของเมล์เซิร์ฟเวอร์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



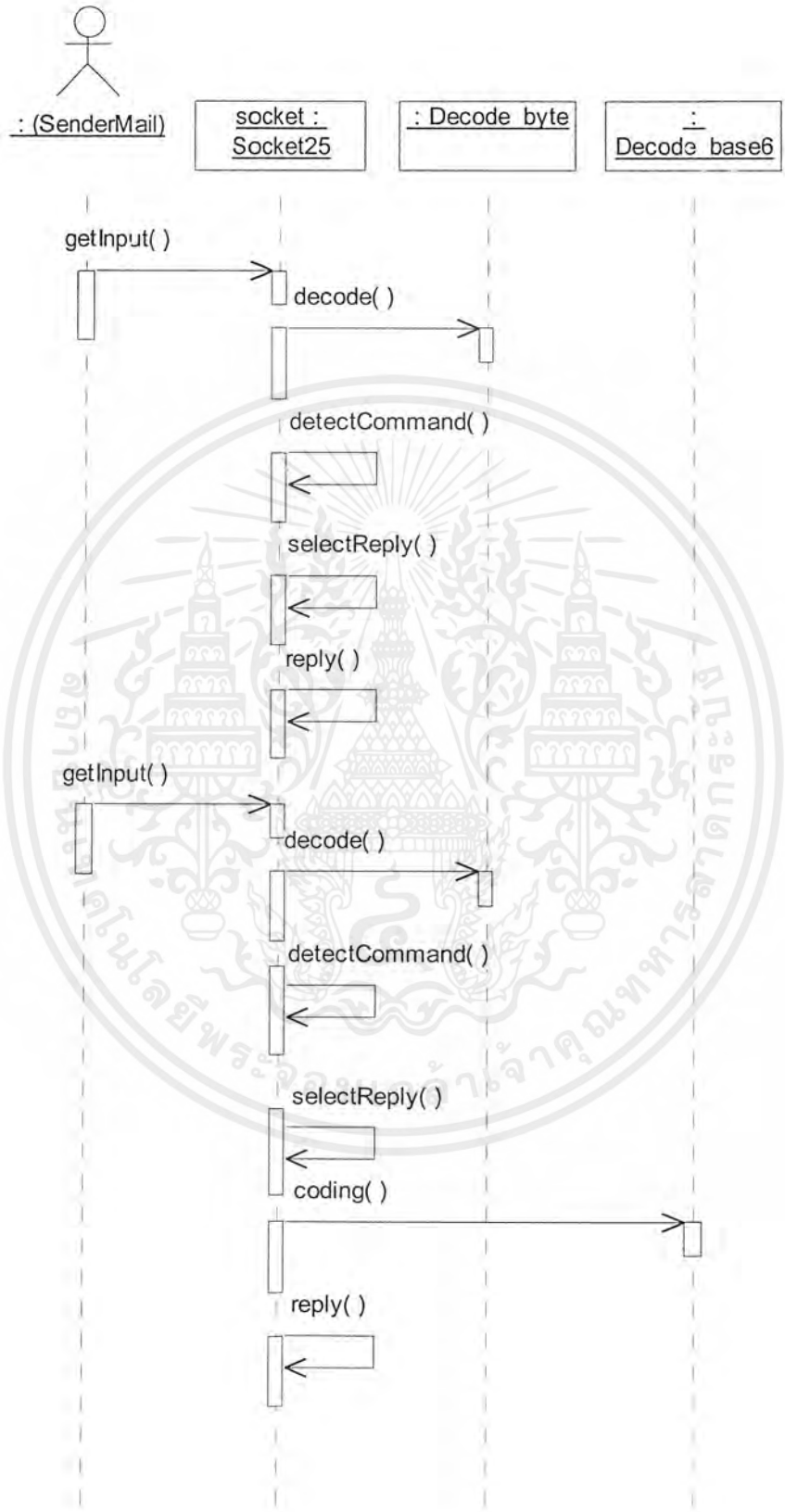
รูปที่ 3-13 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์(กรณีที่มีการตรวจเช็คอีเมล์แอดเดรส)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



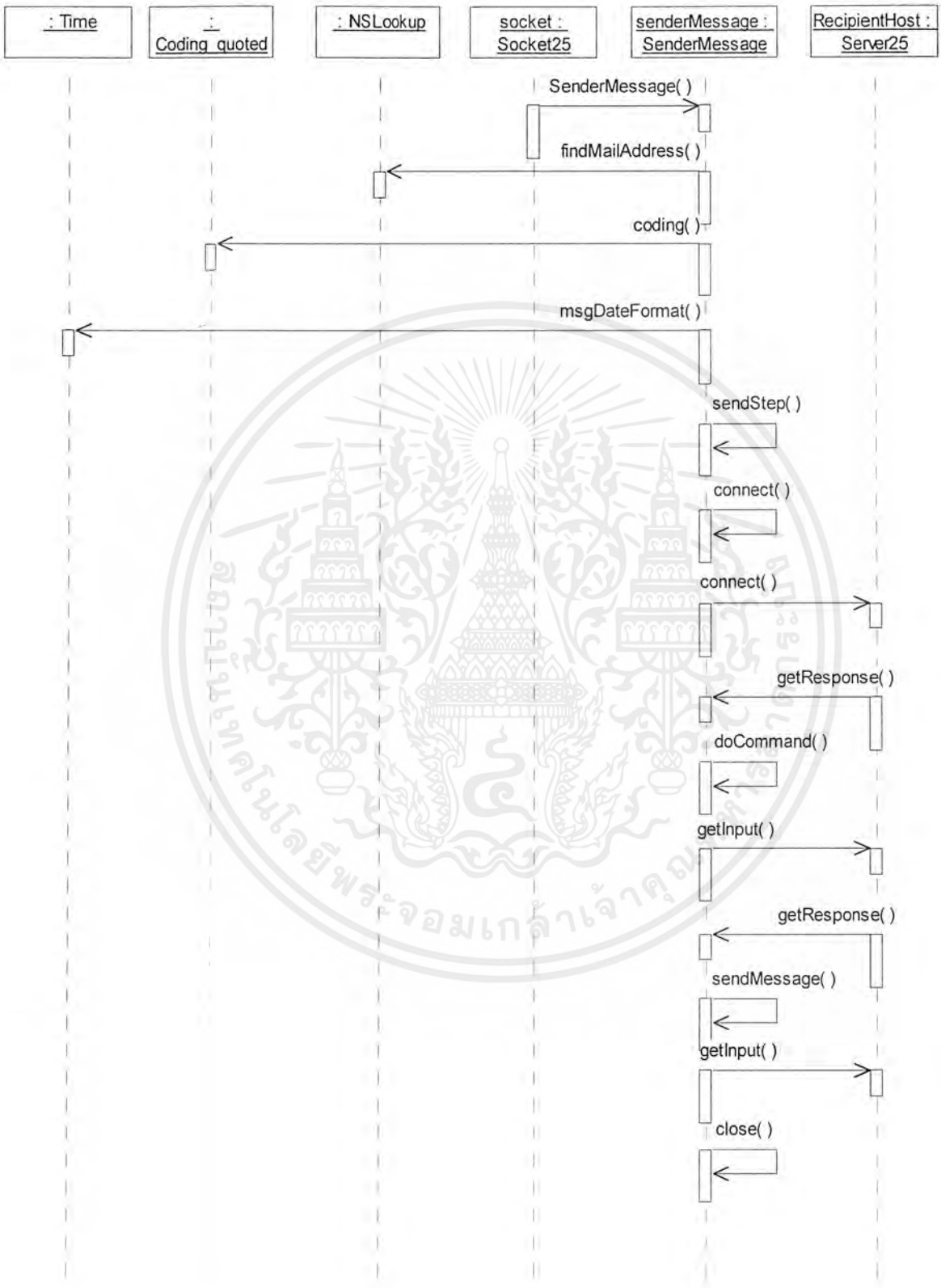
รูปที่ 3-14 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์(กรณีที่ข้อความมีการเข้ารหัสแบบQuoted Printable)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-15 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์(กรณีที่ข้อความของเมล์มีการเข้ารหัสแบบ base64)

เอกสารนี้ใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-16 แสดง Sequence Diagram ของเมล์เซิร์ฟเวอร์(กรณีที่อีเมล์แอดเดรสของผู้รับเป็นของเซิร์ฟเวอร์ตัวอื่น)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 ตารางที่ใช้เก็บข้อมูล

infomail

Account	Name	Password	address	telephone	sex	age	work	part	Icq

datamail

Account	Indexs	sender	datas	subject	message	Status

ในการเก็บข้อมูลเราออกแบบตารางเป็นสองตารางคือ

- ตาราง infomail เป็นตารางที่เก็บข้อมูลเกี่ยวกับประวัติของผู้ใช้ที่ทำการลงทะเบียน

account เก็บชื่อติดต่อบริการของผู้ใช้

name เก็บชื่อและนามสกุลของผู้ใช้

password เก็บรหัสผ่านของผู้ใช้

address เก็บที่อยู่ของผู้ใช้

telephone เก็บหมายเลขโทรศัพท์ของผู้ใช้

sex เก็บเพศของผู้ใช้ว่าเป็นชายหรือหญิง ถ้าเป็นชายจะเก็บ 1 ถ้าเป็นหญิงจะเก็บ 0

age เก็บอายุของผู้ใช้

work เก็บอาชีพของผู้ใช้

part เก็บชื่อองค์กรของผู้ใช้

icq เก็บหมายเลขไอซีคิวของผู้ใช้

- ตาราง datamail เป็นตารางที่เก็บข้อมูลเกี่ยวกับชื่อผู้ส่งจดหมายและข้อความต่างๆ เป็นต้น

account เก็บชื่อติดต่อบริการ

indexs เป็นตัวชี้ไปที่จดหมายแต่ละฉบับ

sender เก็บชื่ออีเมล์แอดเดรสของผู้ส่งจดหมาย

datas เก็บวันที่ที่ผู้ส่งส่งจดหมายมาให้เรา

subject เก็บหัวข้อจดหมาย

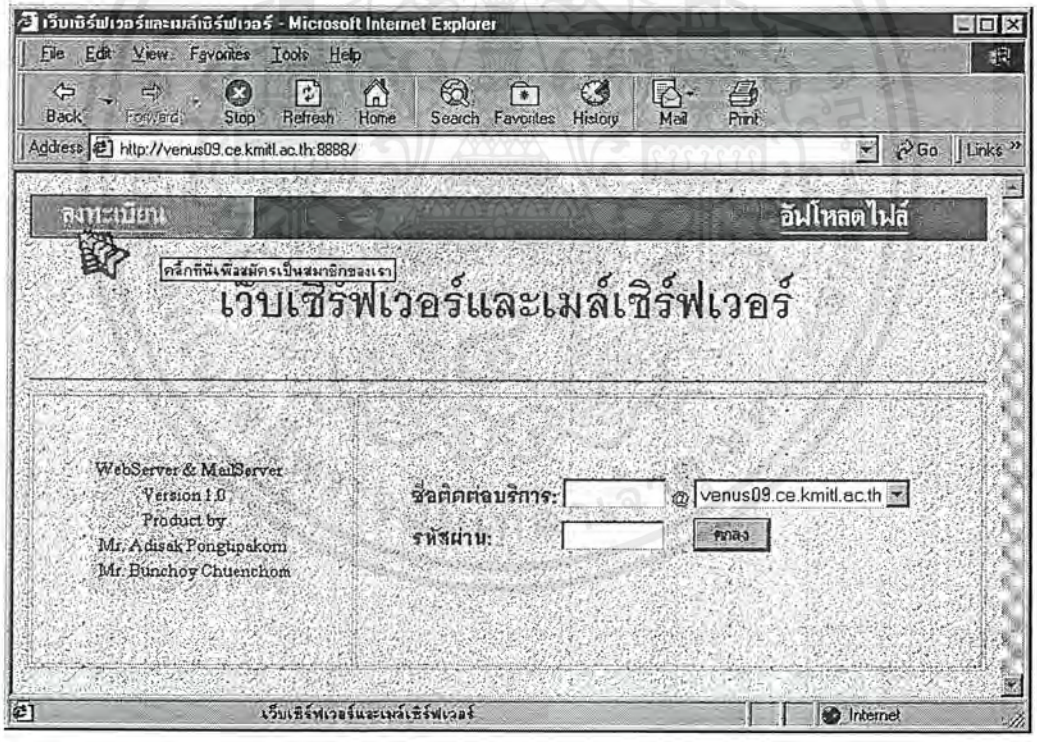
message เก็บข้อความที่ต้องการจะส่ง

status เก็บสถานะของจดหมายว่าอ่านแล้วหรือยังไม่อ่าน ถ้าจดหมายยังไม่ได้อ่านจะเก็บ 1 ถ้าจดหมายถูกอ่านแล้วจะเก็บ 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การทำงานของโปรแกรม

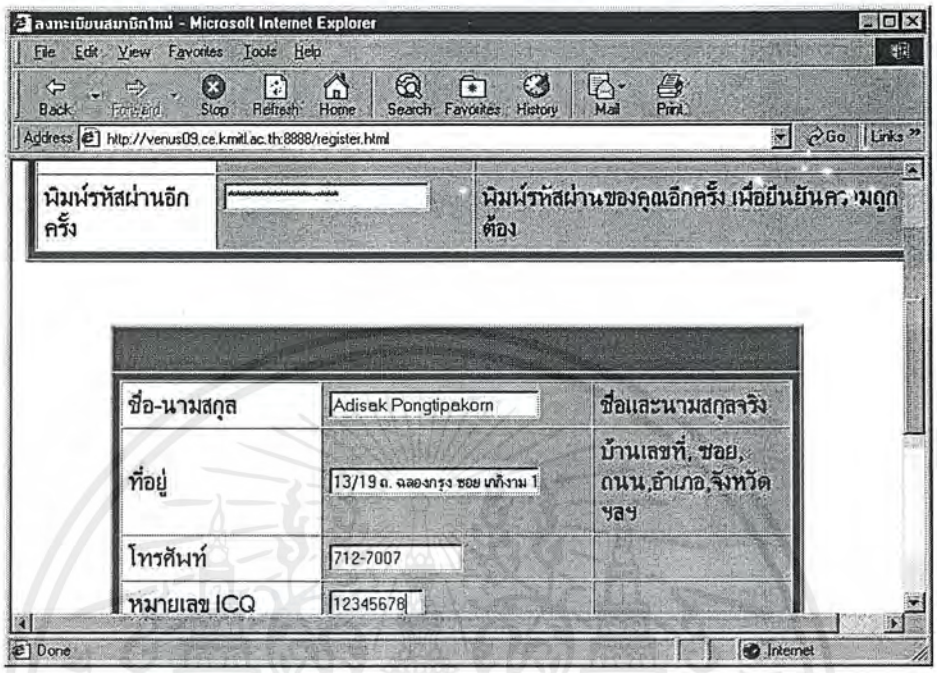
ในส่วนนี้จะเป็นส่วนของการทดลองและทดสอบโปรแกรมที่เขียนขึ้นมา ตามที่เราได้ออกแบบไว้ โดยเราต้องทำการรันโปรแกรมที่เขียนขึ้นเพื่อทำหน้าที่เป็นเซิร์ฟเวอร์ ผู้ใช้สามารถเข้ามาใช้งานเครื่องเซิร์ฟเวอร์ได้โดยพิมพ์ URL ที่เบราเซอร์ดังนี้ http://venus09.ce.kmitl.ac.th:8888 หรือสามารถใช้เป็นไอพีแอดเดรสก็ได้ http://161.246.6.119:8888 จากนั้นเครื่องเซิร์ฟเวอร์จะส่งเอกสาร HTML กลับไปยังเครื่องไคลเอนท์แล้วจึงทำการตัดการเชื่อมต่อและหากเครื่องไคลเอนท์ต้องการไฟล์อื่นเพิ่มเช่นไฟล์รูปภาพก็จะทำการร้องขอมาที่เซิร์ฟเวอร์อีกครั้ง และเมื่อเซิร์ฟเวอร์ตอบสนองไคลเอนท์เสร็จแล้วก็จะตัดการเชื่อมต่ออีกครั้ง ซึ่งแสดงว่ามีการเชื่อมต่อมาที่เครื่องเซิร์ฟเวอร์หลายครั้ง และเมื่อร้องขอครบทุกไฟล์แล้วหน้าจอที่แสดงที่เบราเซอร์จะเป็นดังรูปที่ 4-1



รูปที่ 4-1 แสดงหน้าจอหลัก

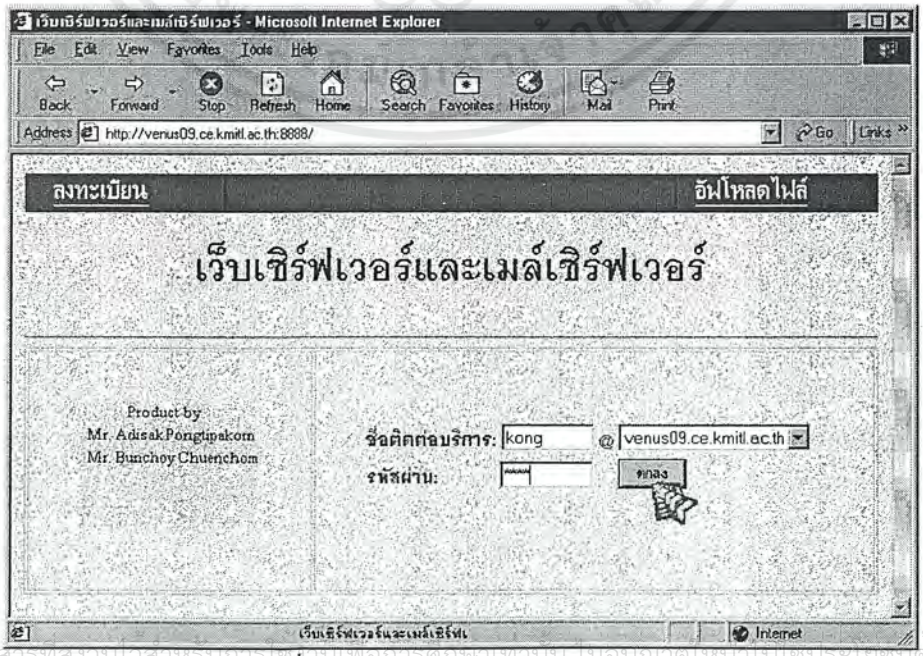
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมมุติว่าเรายังไม่เคยเข้ามาใช้งานเซิร์ฟเวอร์ เราต้องทำการลงทะเบียนก่อนถึงจะสามารถเข้ามาใช้งานเซิร์ฟเวอร์ได้ โดยการคลิกที่ "ลงทะเบียน" จะปรากฏหน้าจอให้เรากรอกแบบฟอร์ม ดังรูปที่4-2



รูปที่ 4-2 แสดงหน้าจอแบบฟอร์มการลงทะเบียน

หลังจากที่เราได้ทำการลงทะเบียนแล้วเครื่องเซิร์ฟเวอร์จะทำการเพิ่มชื่อล็อกอินลงในดาต้าเบสและทำการสร้างไดเร็กทอรี่ชื่อเดียวกันกับชื่อล็อกอินไว้สำหรับเก็บโฮมเพจของผู้ใช้ ซึ่งเป็นเอกสาร HTML และไฟล์รูปภาพต่างๆ หลักจากนั้นถ้าเราต้องการเข้ามาเช็คเมล์ก็สามารถทำได้ โดยการพิมพ์ชื่อติดต่อบริการที่และรหัสผ่านให้ถูกต้องแล้วกดปุ่ม "ตกลง" ดังรูป



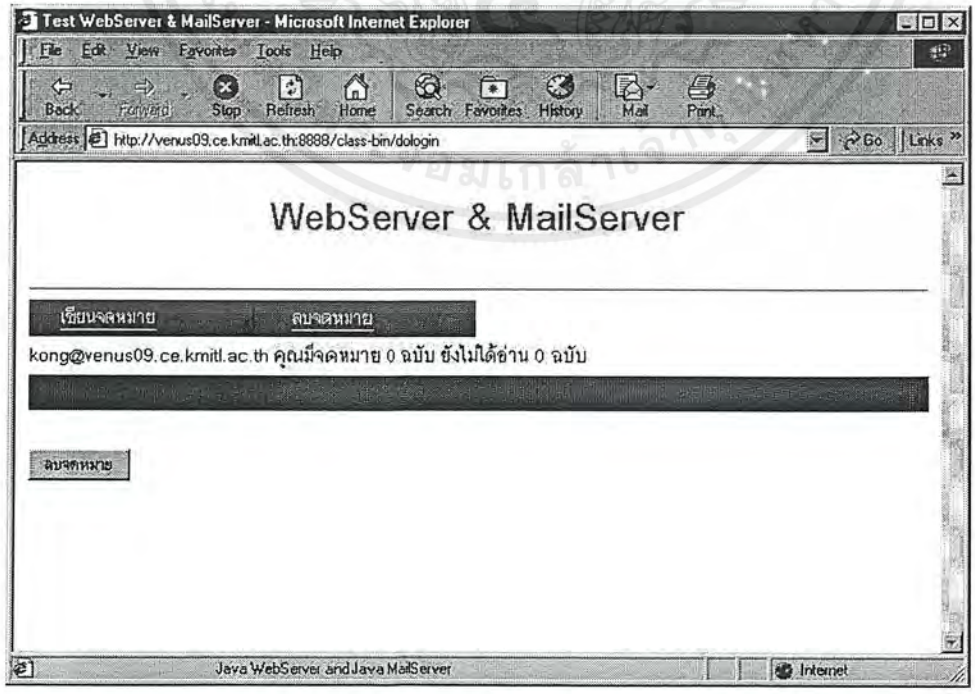
รูปที่ 4-3 แสดงหน้าจอการล็อกอิน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าเราพิมพ์ชื่อติดต่อบริการผิดหรือว่าพิมพ์รหัสผ่านผิดเครื่องเซิร์ฟเวอร์จะส่งข้อความแสดงความผิดพลาดขึ้นดังรูปที่ 4-4

รูปที่ 4-4 แสดงหน้าจอการใส่รหัสผ่านผิดพลาด

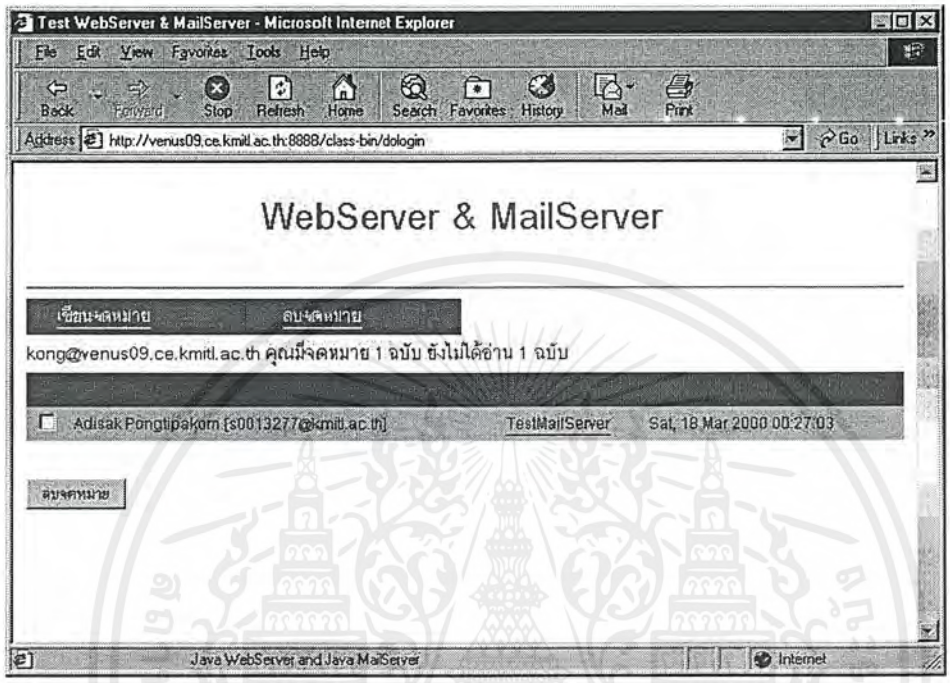
ให้เราคลิกที่ "กลับเมนูหลัก" แล้วทำการพิมพ์ชื่อติดต่อบริการและรหัสผ่านให้ถูกต้อง เมื่อถูกต้องแล้วเครื่องเซิร์ฟเวอร์จะส่งเอกสาร HTML มาแสดงแก่ผู้ใช้ดังรูปที่ 4.5



รูปที่ 4-5 แสดงหน้าจอการเข้ามาอ่านจดหมาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในขณะนี้เราเพิ่งทำการลงทะเบียนจึงยังไม่มีใครส่งจดหมายมาหาเราจะเห็นว่ามีจดหมายอยู่ 0 ฉบับ ถ้าสมมุติว่ามีเพื่อนส่งจดหมายมาให้เรา หน้าจอการแสดงผลจะยังไม่เปลี่ยนแปลงให้เรากดที่ปุ่ม "Refresh" ที่เบราเซอร์ หน้าจอการแสดงผลจะเป็นดังรูป 4-6



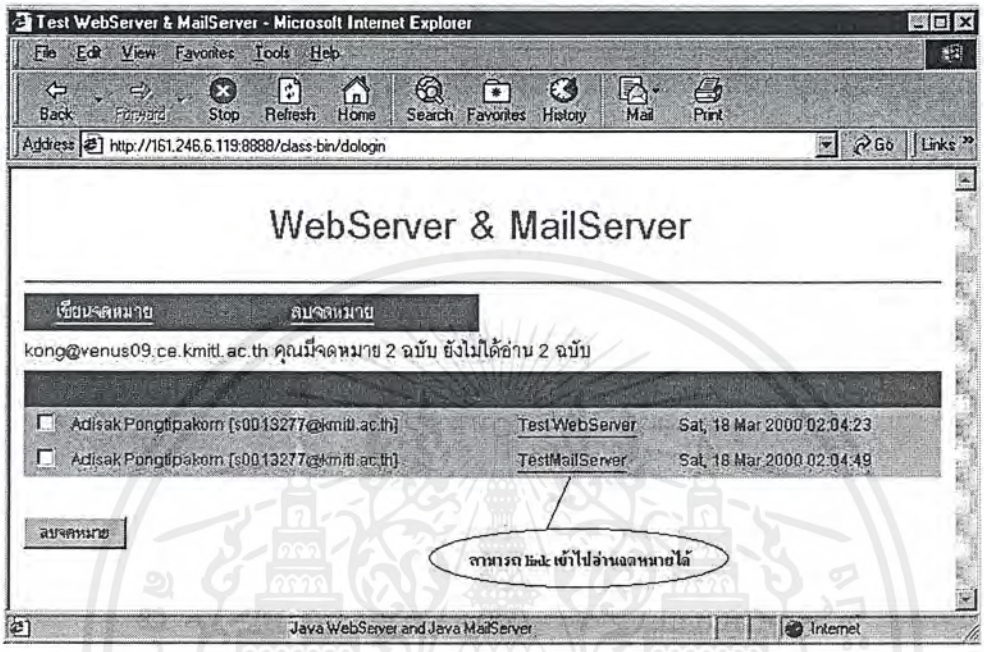
รูปที่ 4-6 แสดงหน้าจอการเข้ามาดูจดหมายทั้งหมด

เราสามารถทำการลบไฟล์ที่เพื่อนเราส่งมาให้ได้โดยการ คลิกที่เช็คบล็อกดังรูปที่ 4-7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

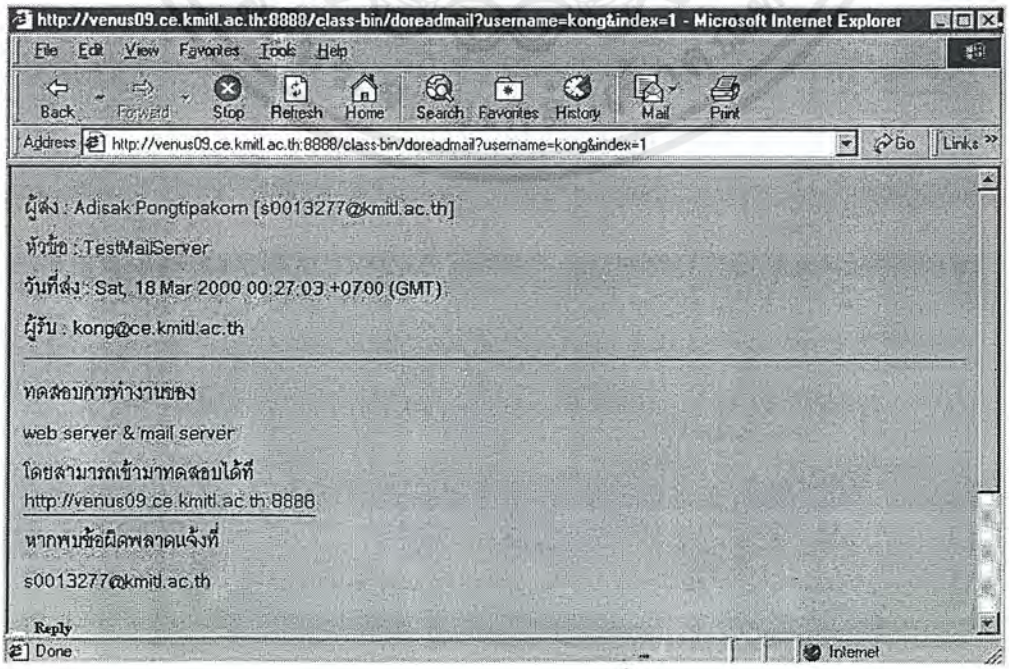
รูปที่ 4-7 แสดงหน้าจอการลบจดหมาย

จากนั้นทำการกดปุ่ม "ลบหมาย" เครื่องเซิร์ฟเวอร์จะทำการลบจดหมายที่เราได้เช็คบล็อกไว้ แล้วกลับมาแสดงหน้าจอเหมือนตอนยังไม่มีใครส่งจดหมายมาให้เรา และถ้ามีเพื่อนส่งจดหมายมาให้เราสองฉบับ หน้าจอในการแสดงผลจะเป็นดังรูปที่ 4-8



รูปที่ 4-8 แสดงหน้าจอการเข้ามาดูจดหมายทั้งหมด

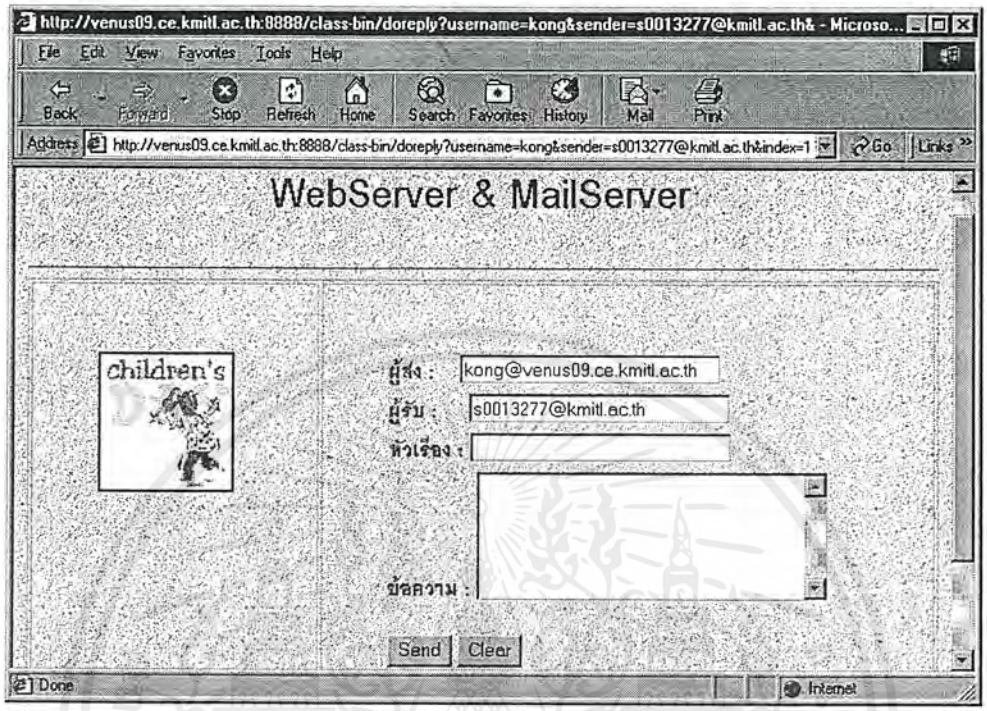
ถ้าเราต้องการเข้าไปอ่านจดหมายที่เพื่อนเราส่งมาให้ให้เราคลิกที่หัวข้อจดหมายที่ต้องการอ่านในที่นี้เราคลิกที่หัวข้อ "TestMailServer"



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

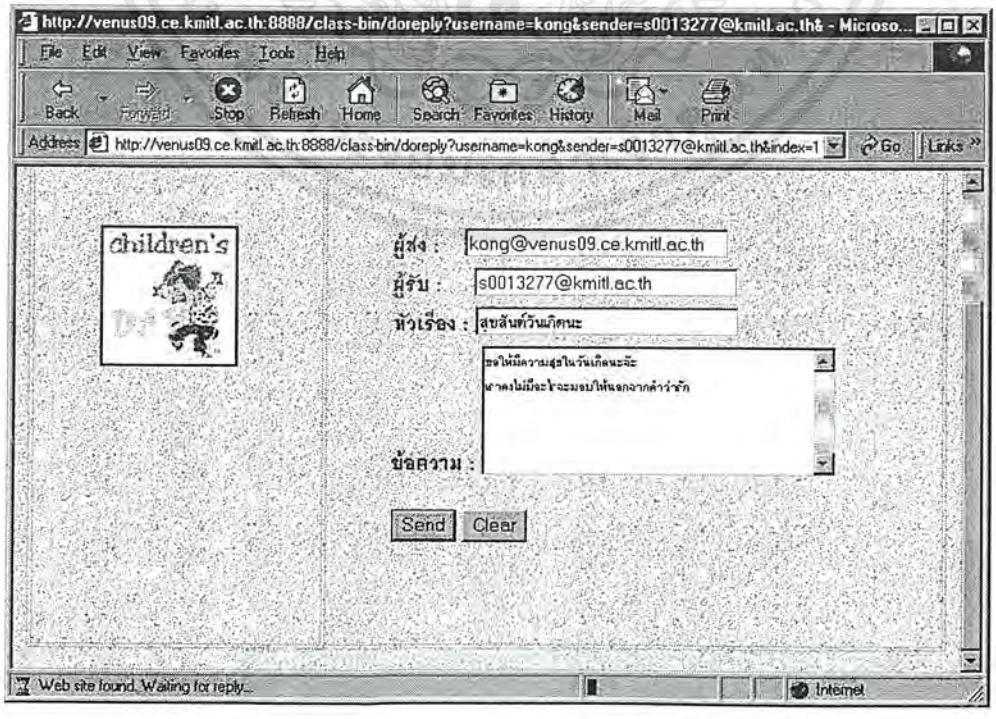
รูปที่ 4-9 แสดงหน้าจอการเข้ามาอ่านจดหมาย

เมื่อเราอ่านจดหมายเสร็จเรียบร้อยแล้วเราสามารถส่งจดหมายตอบกลับได้โดยการคลิกที่ “Reply” จะแสดงหน้าจอดังรูปที่ 4-10



รูปที่ 4-10 แสดงหน้าจอการตอบจดหมาย

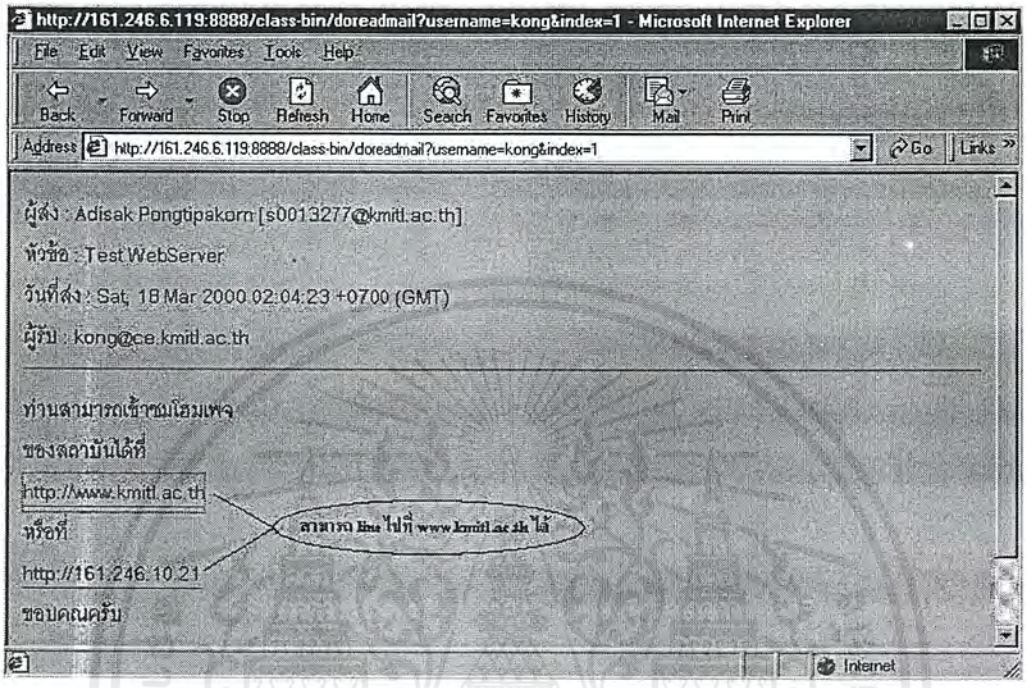
จะเห็นว่าในส่วนของผู้รับกับผู้ส่งเครื่องเซิร์ฟเวอร์จะจัดการใส่ข้อมูลให้เราแล้ว ให้เรากรอกเพียงหัวข้อและข้อความที่ต้องการ จากนั้นคลิกที่ปุ่ม “Send” จะปรากฏหน้าจอดังรูปที่ 4-11



รูปที่ 4-11 แสดงหน้าจอการตอบจดหมาย

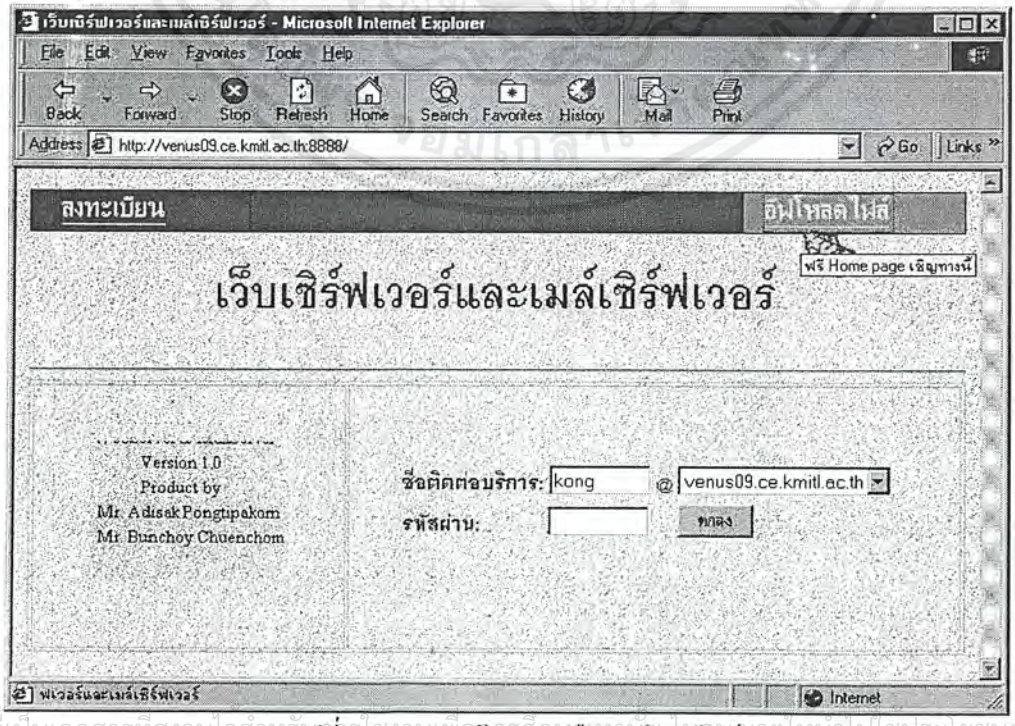
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และเมื่อส่งจดหมายเสร็จแล้วจะแสดงหน้าจอบอกให้ทราบว่าจดหมายถูกส่งเรียบร้อยแล้ว และในการอ่านจดหมายนั้นถ้าในส่วนของข้อความ มีส่วนที่เป็น URL ผู้ใช้สามารถคลิกเพื่อลิ้งค์ไปยัง URL ที่ต้องการได้ ดังรูปที่ 4-12



รูปที่ 4-12 แสดงหน้าจอการเข้ามาอ่านจดหมาย

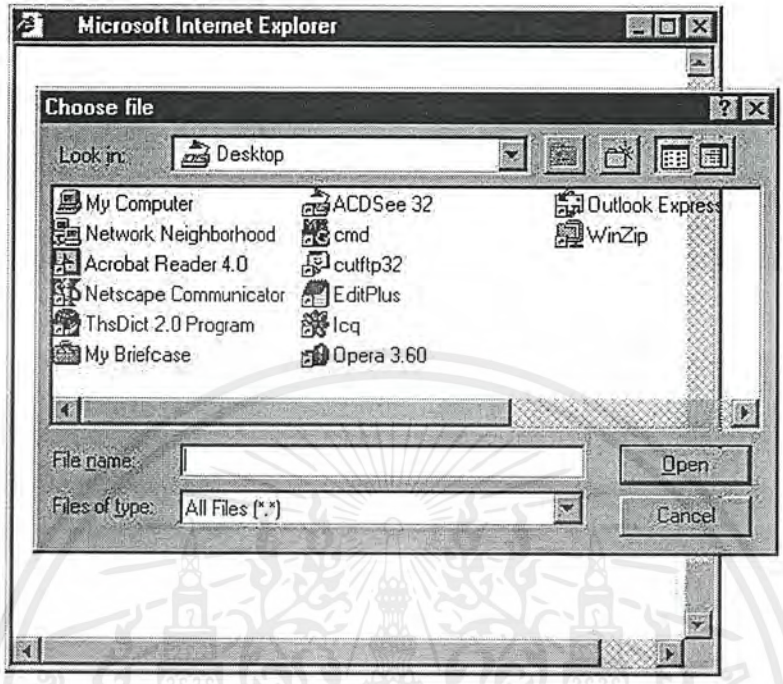
และในกรณีที่ผู้ใช้ต้องการนำโฮมเพจเข้ามาเก็บไว้ในเครื่องเซิร์ฟเวอร์สามารถทำได้โดยคลิกที่"อัปโหลดไฟล์" ดังรูปที่ 4-13



รูปที่ 4-13 แสดงหน้าจอการเลือกอัปโหลดไฟล์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากนั้นจะแสดงหน้าจอให้ผู้ใช้เลือกไฟล์ที่ต้องการจะนำมาเก็บไว้ที่เครื่องเซิร์ฟเวอร์ ซึ่งจะแสดงหน้าจอดังรูปที่ 4-14



รูปที่ 4-14 แสดงหน้าจอการอัปโหลดไฟล์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
บทสรุปและวิจารณ์

5.1 สรุปและบทวิจารณ์

โปรโตคอล HTTP เป็นโปรโตคอลที่ใช้งานในการกระจายข้อมูลข่าวสารบนอินเทอร์เน็ต โดยเน้นความรวดเร็วในการรับส่งข้อมูลเป็นสำคัญ ผู้ใช้งานโปรโตคอลนี้ก็คือโปรแกรมที่ทำหน้าที่เป็นเว็บเบราเซอร์และเว็บเซิร์ฟเวอร์ โดยเบราเซอร์จะส่ง Request Header ไปให้เว็บเซิร์ฟเวอร์ จากนั้นเว็บเซิร์ฟเวอร์จะส่ง Response Header และข้อมูล (ถ้ามี) กลับมายังเว็บเบราเซอร์ เว็บเบราเซอร์จะอ่านค่า Response Header และแสดงผลเว็บเพจนั้น

SMTP เป็นมาตราฐานที่ใช้ในการรับและส่งจดหมายอิเล็กทรอนิกส์ในเครือข่ายอินเทอร์เน็ตซึ่งทำงานที่ Port 25 เป็นมาตรฐาน โดยเซิร์ฟเวอร์จะรอรับจดหมายและถ้ามีจดหมายถูกส่งมาเซิร์ฟเวอร์จะทำการจัดเก็บลงในดาต้าเบส และในการอ่านจดหมายนั้นจะทำการอ่านผ่านเบราเซอร์ซึ่งไม่จำเป็นต้องใช้ POP3 หรือ IMAP4 การอ่านข้อมูล เซิร์ฟเวอร์จะทำการติดต่อกับดาต้าเบสจากนั้นจึงสงข้อมูลที่อ่านได้กลับไปยังเบราเซอร์เพื่อแสดงผลต่อไป

ส่วนของโครงงานที่พัฒนาคือเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์นั้น ขอบเขตของการให้บริการต่างๆนั้นมีมาก เราได้เลือกศึกษาเพียงบางส่วนเท่านั้น ยังมีส่วนที่น่าสนใจที่ควรจะพัฒนาต่อไปอีกเช่น ส่วนของ POP3 และ IMAP4 และการพูดคุยกันด้วยเสียงผ่านเบราเซอร์ และในส่วนของการให้บริการ Dynamic HTML นั้นเราไม่สามารถติดต่อกับ Jrun ได้ทำให้เราไม่สามารถใช้งาน servlet ได้ เพราะว่าเว็บเซิร์ฟเวอร์ที่เราเขียนขึ้นไม่ซับพอร์ท

ส่วนของภาษาที่เลือกใช้เราได้พิจารณาจากความต้องการที่จะใช้ความสามารถของภาษาจาวาคือมีการทำงานที่ไม่ขึ้นกับแพล็ตฟอร์มเราจึงเลือกที่จะใช้จาวา jdk เพราะถ้าเรานำโปรแกรมที่เขียนขึ้นบนระบบปฏิบัติการ windows ไปใช้งานในระบบปฏิบัติการ unix เราเพียงนำโปรแกรมนั้นไปทำการรันบนระบบปฏิบัติการ unix ก็สามารถใช้งานได้ และโปรแกรมที่เราสร้างขึ้นไม่มีส่วนของ GUI เราจึงเลือกเขียนด้วยจาวา jdk แทนที่จะเป็นจาวาคาเฟ่หรือจาวา Jbuilder

5.2 แนวทางในการพัฒนาต่อ

เนื่องจากระบบที่เป็นเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์เป็นการทำงานของระบบที่เป็นมาตราฐานมีอ้างอิงกับ RFC ที่เป็นมาตราฐานอยู่หลาย RFC แต่จากโครงงานเป็นการศึกษาการทำงานการพัฒนาแอพพลิเคชั่นเชิงวัตถุโดยใช้ภาษาจาวาหลักๆของระบบเราจึงทำการเลือกพัฒนาระบบในส่วนการทำงานหลักๆของระบบดังนั้นการทำงานของระบบจึงยังไม่ได้รองรับทุกกรณีเท่ากับระบบที่ใช้งานจริง เช่น ระบบเมล์เซิร์ฟเวอร์มีการรองรับเพียงคำสั่งที่ใช้อยู่เป็นประจำยังไม่รองรับทุกคำสั่งที่มีใน RFC821 และรูปแบบการส่งเมล์แบบ TEXT/PLAIN มีการรองรับการเข้ารหัสในแบบ Base64 กับแบบ Quoted- printable ดังนั้นการส่งจากโปรแกรมส่งเมล์ เช่น โปรแกรม Outlook อาจมีปัญหาเพราะเป็นการใช้รูปแบบการส่งเมล์แบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช

multipart/alternative ซึ่งเป็นรูปแบบการส่งเมล์อีกแบบหนึ่งที่ระบบที่ทำการพัฒนานี้ยังไม่ได้รองรับไว้ดังนั้นแนวทางในการพัฒนาระบบต่อจึงเป็นการพัฒนาระบบให้มีการรองรับกรณีต่าง ๆได้สมบูรณ์และการเพิ่มความสามารถของระบบซึ่งยกตัวอย่างเป็นข้อๆดังนี้

- การเพิ่มความสามารถในการรับและส่งไฟล์ที่แนบไปกับจดหมายได้
- พัฒนาให้มีความทันสมัยโดยสามารถรับจดหมายจากเมล์เซิร์ฟเวอร์ที่ทำการเข้ารหัสข้อมูลในรูปแบบต่างๆให้ครบทุกรูปแบบ
- การเพิ่มความสามารถให้สามารถส่งจดหมายให้แก่ผู้รับหลายๆคนในการส่งครั้งเดียวได้
- การเพิ่มความสามารถของแอพพลิเคชั่นบนเว็บให้สามารถทำการคุยกันผ่านเว็บเบราเซอร์ได้

ข้อมูลที่ใช้ในการทำโครงงานนี้และข้อมูลที่จะใช้ในการศึกษาเพื่อพัฒนาโครงงานนี้ต่อจะสามารถหาได้จากหนังสือที่กล่าวในส่วนของบรรณานุกรมจะเป็นส่วนของข้อมูลที่เกี่ยวกับการศึกษาภาษาจาวาที่ใช้ในการพัฒนาโครงงานซึ่งเป็นทั้งในส่วนของการใช้ภาษาจาวาและโค้ดตัวอย่างของแอพพลิเคชั้นต่างๆซึ่งมีการทำงานแบบต่างๆจึงสามารถนำมาศึกษาและประยุกต์ใช้ในส่วนของแอพพลิเคชั้นที่เป็นโครงงานและมีบางส่วนที่สามารถนำมาใช้ได้เลยซึ่งเป็นข้อดีของแอพพลิเคชั้นที่พัฒนาโดยวิธีเชิงวัตถุ ส่วนข้อมูลที่เกี่ยวกับการทำงานหลักของระบบเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์เป็นข้อมูลที่ได้มาจากเอกสารทางวิชาการที่เกี่ยวกับ Internet Requests for Comments (RFC) สามาะถหาข้อมูลได้จากในอินเตอร์เน็ต เช่น ในเว็บ http://www.cis.ohio-state.edu/hypertext/information/rfc.html จะให้ข้อมูลของ RFC เบอร์ต่างๆโดยละเอียด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

Java Developer Kit (JDK)

ในการสร้างโปรแกรมภาษาจาวานั้นต้องมีคอมไพเลอร์และอินเตอร์ปรีเตอร์ ของภาษาจาวาซึ่งปัจจุบันมีบริษัทผลิตชุดโปรแกรมสำหรับพัฒนาโปรแกรมภาษาจาวาออกมาหลายบริษัทแต่ในปริญญานิพนธ์เล่มนี้เราใช้ JDK ของบริษัทจาวาซอฟต์(JavaSoft) เนื่องจากบริษัทจาวาซอฟต์ เป็นผู้พัฒนาภาษาจาวา ดังนั้นคอมไพเลอร์ที่มีมากับ JDK จึงเป็นมาตราฐานของภาษา อีกทั้งภาษาจาวาถูกพัฒนาอย่างรวดเร็ว มีสิ่งใหม่ๆออกมาอย่างต่อเนื่องบริษัทอื่นๆ จึงพัฒนาได้ช้ากว่าไม่ทันสมัยเท่ากับคอมไพเลอร์ที่มากับ JDK

เมื่อจะทำการศึกษาภาษาจาวาเริ่มแรกต้องหาโปรแกรมสำหรับติดตั้ง JDK ก่อน โดยอาจได้มาจากการ down load ที่ http://www.javasoft.com หรือจากแผ่นซีดีรวมคอมไพเลอร์ที่หาซื้อได้ตามร้านขายซอฟต์แวร์ต่างๆ

เมื่อได้โปรแกรมสำหรับติดตั้ง JDK มาแล้วสมมติว่าเป็นไฟล์ JDK1.1.8.exe ให้ทำการรันโปรแกรม มันจะถูกขยายออก พร้อมการติดตั้งโปรแกรมและไฟล์ลงในไดเรกทอรีที่ให้เราระบุได้ สมมุติว่าชื่อ jdk.1.1.8 ในไดร์ว C: โปรแกรมที่ติดตั้งนั้นจะสร้างไดเรกทอรีชื่อ bin, demo, include, lib,และ src อยู่ภายใต้ไดเรกทอรี c:\jdk1.1.8

เมื่อติดตั้ง JDK ถึงขั้นนี้ คอมไพเลอร์(javac.exe) และ อินเตอร์ปรีเตอร์(java.exe) จะถูกเรียกใช้ได้จากไดเรกทอรี C:\jdk1.1.8\bin เท่านั้น ดังนั้นเพื่อให้สามารถเรียกใช้ javac.exe และ java.exe ได้จากในไดเรกทอรีใดๆ เราจึงควรตั้งค่า directory path ไปที่ C:\jdk1.1.8\bin โดยเพิ่มคำสั่งข้างล่างนี้ลงในไฟล์ autoexec.bat

```
SET PATH=C:\JDK1.1.8\BIN
SET CLASSPATH=.;C:\JDK1.1.8\LIB\CLASSES.ZIP
```

ซึ่งการใช้งาน JDK จะใช้เป็นคำสั่งบนดอสมีโปรแกรมหลักๆ อยู่ 2 โปรแกรมที่ใช้ในการพัฒนาโปรแกรมแอพพลิเคชั่นคือ

- javac.exe เป็นโปรแกรมคอมไพเลอร์ภาษาจาวา
- java.exe เป็นโปรแกรมอินเตอร์ปรีเตอร์ภาษาจาวา

ตัวอย่าง ถ้าเราเขียนไฟล์แอพพลิเคชั่นโดยภาษาจาวาชื่อ Hello.java แล้วทำการคอมไพล์เป็นไฟล์คลาส ในภาษาจาวา โดยใช้คำสั่งดังนี้ (สมมุติเก็บโปรแกรมไว้ใน C:\MyJava)

```
C:\MyJava\javac Hello.java
```

จะได้คลาส Hello.class ใน C:\MyJava เวลาต้องการรันโปรแกรมก็ทำได้โดยใช้คำสั่ง

```
C:\MyJava\java Hello
```

อินเทอร์ปรีนเตอร์ภาษาจาวาก็จะทำการรันคลาสดังกล่าว เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

การลงโปรแกรม

ในการลงโปรแกรมเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์ที่เราเขียนขึ้นนั้น เราได้เตรียมไฟล์ทั้งหมดที่ต้องใช้ซิปไว้แล้ว ผู้ใช้เพียงทำการเรียกไฟล์ Web&Mail.zip เพื่อทำการขยายไฟล์ที่ได้ซิปไว้ ซึ่งมีไฟล์หลักๆ คือ

- Server25.java เป็นไฟล์ที่ใช้ในการรับส่งเมล์
- Httpd.java เป็นไฟล์ที่ให้บริการเว็บเซิร์ฟเวอร์
- FileServer.java เป็นไฟล์ที่ทำหน้าที่รอรับไฟล์ที่ส่งมาจากไคลเอนท์

ให้เราทำการคอมไพล์ไฟล์ทั้งสามโดยใช้คำสั่ง

javac Server25.java

javac Httpd.java

javac FileServer.java

เมื่อเราทำการคอมไพล์ไฟล์ทั้งสามแล้ว เราจะได้ไฟล์นามสกุลคลาสขึ้นมา ให้เราทำการรันโปรแกรมโดยใช้คำสั่ง

java Server25

java Httpd

java FileServer

เป็นอันเสร็จขั้นตอน ขณะนี้เครื่องเซิร์ฟเวอร์ของเราสามารถให้บริการเว็บเซิร์ฟเวอร์และเมล์เซิร์ฟเวอร์ได้แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] Weerasak Sungthawon (1998) : "*Fundamental of JAVA programming (Volume I)* " ,SUM Publishing Department, SUM System Company Limitted Bangkok Thailand

[2] Weerasak Sungthawon (2000) : "*Fundamental of JAVA programming (Volume 2)* " ,SE-EDUCATION PUBLIC COMPANY LIMITED

[3] Mark Wutka (1997): "*HACKING JAVA(The Java Professional's Resource Kit)*" ,Que Corporation

[4] Merlin Hughes, Michael Shoffner, Derek Hamner (1999) : "*Java Network Programming*",Manning Publications Co. All rights reserved.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้