สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การสร้างไดนามิกเว็บเพจ
เพื่อการบริหารงาน PORTAL SITE ด้วย JSP
DYNAMIC WEB PAGE FOR
PORTAL SITE MANAGEMENT USING JSP

T 046147

นางสาวพิมพ์รัตน์ อำนวยพรสถิตย์
นายโสรจน์ มากันต์

เลขหมู่.....
เลขทะเบียน 46147
วัน, เดือน, ปี 20 มี.ค. 2546

.b.....
.i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2544

b 1128532

การสร้างไดนามิกเว็บเพจ
เพื่อการบริหารงาน PORTAL SITE ด้วย JSP

DYNAMIC WEB PAGE FOR
PORTAL SITE MANAGEMENT USING JSP

โดย

นางสาวพิมพ์รัตน์ อำนวยพรสถิตย์

นายโสรจน์ มากันต์

อาจารย์ที่ปรึกษา

รศ.บรรจง ปียธำรง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การสร้างไดนามิกเว็บเพจเพื่อการบริหารงาน PORTAL SITE ด้วย JSP

DYNAMIC WEB PAGE FOR PORTAL SITE MANAGEMENT USING JSP

ผู้จัดทำ

1. นางสาวพิมพ์รัตน์ อำนวยพรสถิตย์ รหัสประจำตัว 41014308
2. นายโสรจน์ มากันต์ รหัสประจำตัว 41014496



อาจารย์ที่ปรึกษา

(รศ. บรรจง ปิยธำรง)

การสร้างไดนามิกเว็บเพจเพื่อการบริหารงาน PORTAL SITE ด้วย JSP

นางสาวพิมพ์รัตน์ อำนวยพรสถิตย์
นายโสรจน์ มากันต์
รศ. บรรจง ปิยธำรง อาจารย์ที่ปรึกษา
ปีการศึกษา 2544

บทคัดย่อ

การนำเทคโนโลยีของคอมพิวเตอร์มาใช้งานในเชิงประยุกต์มิได้จำกัดวงอยู่แต่ในห้องทดลองทางวิทยาศาสตร์เท่านั้น หากแต่ได้ก้าวเข้ามาเป็นส่วนหนึ่งของชีวิตประจำวันของคนทั่วไป ปัจจัยหลักมาจากอินเตอร์เน็ตที่ได้รับความนิยมอย่างรวดเร็ว จนทำให้เกิดนวัตกรรมทางธุรกิจบนอินเตอร์เน็ตในรูปแบบต่างๆ อย่างต่อเนื่อง ไม่ว่าจะเป็นธุรกิจดอทคอม, B2B, B2C

พอร์ทอลไซต์ (Portal Site) คืออีกผลพวงหนึ่งที่เกิดขึ้น ซึ่งเกิดจากการรวบรวมเนื้อหาและข้อมูลของเว็บต่างๆ มาไว้บนเว็บเพจเดียวกัน เพื่อให้ง่ายต่อการเข้าถึงข้อมูลของผู้เข้าชมเว็บเพจ ดังนั้นข้อมูลจากเว็บต่างๆ ที่นำมาแสดงบนพอร์ทอลไซต์จึงต้องมีระบบการจัดการที่ดี และสามารถรายงานผลทางสถิติต่างๆ นอกเหนือจากการแสดงข้อมูลเพียงอย่างเดียว

ปริญญานิพนธ์ฉบับนี้ได้สร้างพอร์ทอลไซต์ที่แสดงผลงานทางวิชาการของนักศึกษาระดับปริญญาตรี โดยให้ความสำคัญกับการจัดการข้อมูลเชิงสถิติต่างๆ เพื่อรายงานให้กับผู้ดูแลเว็บเพจต่างๆ ได้รับทราบเพื่อเป็นแนวทางในการพัฒนาและปรับปรุงข้อมูลต่อไปโดยใช้ Oracle9iAS Portal และภาษา JavaServer Pages (JSP) ในการพัฒนาพอร์ทอลไซต์ดังกล่าว

Dynamic Web Page For Portal Site Management Using JSP

Pimrat Umnuaypornsatid

Soroj Margun

Assoc. Prof. Bunjong Piyathumrong Advisor

ABSTRACT

Using computer technology in term of applications is not limited only in laboratary but it's also participate in our modern life style. The main criteria of this phenomena is the rapidly growing of Internet Technology that affect to innovate many type of business online such as business.com , B2B and B2C.

Portal site is one of the result which taken place by collecting content to show in only one website as the central of the others to ensure that it is access efficiently. Therefore, the systematic data management and the statistical reports on portal site are important not only display data.

This thesis uses Oracle9iAS Portal and JavaServer Pages (JSP) to create portal site as Academic Project of Bachelor degree by realize the important of statistical data management to support the monitoring of web administrator to nearing development.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจสำเร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลายฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ อาจารย์บรรจง ปียธำรง อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ และให้คำแนะนำเสมอมา

ขอขอบพระคุณบุคลากรและคณาจารย์ในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้ประสิทธิประสาทความรู้ให้โดยตลอด

ขอบคุณทุกน้ำใจที่เกิดจากเพื่อนร่วมสถาบัน โดยเฉพาะอย่างยิ่งเพื่อนๆ ในภาควิชาวิศวกรรมคอมพิวเตอร์ เพื่อนห้องเน็ตเวิร์ก อีกทั้งเพื่อน พี่ น้องในชุมนุมอิเล็กทรอนิกส์ที่มีกำลังใจคอยเติมเต็มให้เสมอ

ขอขอบคุณบริษัท ORACLE ที่ให้การสนับสนุนในเรื่องซอฟต์แวร์และคำแนะนำที่ดี

สุดท้ายขอขอบพระคุณบุคคลสำคัญที่สุดที่ทำให้ข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดูผู้เขียนมาเป็นอย่างดี พร้อมทั้งให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจ เอาใจใส่เสมอมา ในทุก ๆ ด้านอันหาที่เปรียบมิได้ ข้าพเจ้าขอระลึกในพระคุณอันสุดประมาณ และขอกราบขอบพระคุณมา ณ ที่นี้

นางสาวพิมพ์รัตน์ อำนวยพรสถิตย์

นายโสรจน์ มากันต์

สารบัญ

		หน้าที่
บทคัดย่อภาษาไทย		I
บทคัดย่อภาษาอังกฤษ		II
กิตติกรรมประกาศ		III
สารบัญ		IV
สารบัญตาราง		VIII
สารบัญภาพ		XI
บทที่ 1	บทนำ	1
	1.1 ความสำคัญและที่มา	1
	1.2 วัตถุประสงค์ของโครงงาน	1
	1.3 ขอบเขตของโครงงาน	2
	1.4 วิธีการดำเนินงาน	2
บทที่ 2	หลักการพัฒนาโปรแกรมด้วยวิธีเชิงวัตถุ (Object Oriented Programming)	3
	2.1 หลักการของวัตถุ	3
	2.2 คุณสมบัติของหลักการเชิงวัตถุ	4
	2.2.1 Encapsulation	4
	2.2.2 Inheritance	4
	2.2.3 Polymorphism	4
	2.3 ภาษาในการเขียนโปรแกรมเชิงวัตถุ	4
	2.3.1 จะต้องเป็นภาษาที่ใช้วัตถุมิใช่อัลกอริทึมในการแก้ปัญหาเป็นหลัก	4
	2.3.2 วัตถุที่ปรากฏในการเขียนโปรแกรมต้องเป็นอินสแทนซ์ของคลาส	5
	2.3.3 คลาสจะต้องมีความสามารถในการถ่ายทอดคุณสมบัติ	5
	2.4 ภาษาจาวา	5
	2.5 หลักการของภาษาจาวา	6
	2.6 จาวาสคริปต์ (Javascript)	7
	2.7 จาวาแอพเพล็ต (Java Applet)	7
	2.8 จาวาเซิร์ฟเล็ต (Java Servlet)	8
	2.9 จาวาบีน (Java Bean)	8
บทที่ 3	จาวาเซิร์ฟเวอร์เพจ (Java Server Page)	9
	3.1 เปรียบเทียบ JSP กับเทคโนโลยีอื่นๆ	9
	3.1.1 แอคทีฟเซิร์ฟเวอร์เพจ	9

3.1.2 จาวาเซิร์ฟเล็ต 9

3.1.3 จาวาสคริปต์ 9

3.2 JSP คอนเทนเนอร์ (JSP Container) 10

3.3 JSP Standard Syntax and Semantic 10

3.3.1 Directive 10

3.3.2 Declaration 11

3.3.3 Expression 12

3.3.4 Code Fragment/Scriplets 12

3.3.5 Comment 12

3.3.6 JSP Implicit Objects 12

3.3.7 JSP Action 13

บทที่ 4 การวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุ (Object Oriented Analysis & Design) 15

4.1 วัตถุ (Object) 15

4.1.1 แอททริบิวต์ (Attribute) 16

4.1.2 เมธอดหรือพฤติกรรม 17

4.1.3 เมสเซจ (Message) 17

4.1.4 ความรับผิดชอบ (Responsibility) 18

4.2 ความสัมพันธ์ระหว่างวัตถุ 18

4.2.1 ลิงค์ 18

4.2.2 ความสัมพันธ์แบบเป็นส่วนหนึ่งของ 18

4.3 Unified Modeling Language 19

4.4 แผนภาพของ UML 20

4.4.1 แผนภาพ Use Case 20

4.4.2 แผนภาพ Sequence 21

4.4.3 แผนภาพ Collaboration 22

4.4.4 แผนภาพ Class 23

4.4.5 แผนภาพ Object 24

4.4.6 แผนภาพ State 24

4.4.7 แผนภาพ Component 24

4.4.8 แผนภาพ Deployment 24

4.4.9 แผนภาพ Interaction 24

4.4.10 แผนภาพ Activity 24

4.5 สัญลักษณ์ของ UML 25

บทที่ 5 ระบบเครือข่ายแบบไคลเอ็นต์/เซิร์ฟเวอร์ (Client/Server) 27
5.1 รูปแบบของไคลเอ็นต์/เซิร์ฟเวอร์ที่ใช้งานมีอยู่ 4 ชนิด 28
5.2 การจัดแบ่งประเภทของไคลเอ็นต์/เซิร์ฟเวอร์จะแบ่งตามลำดับชั้น (Tier) ในการทำงาน 29
5.3 การจัดแบ่งประเภทของเซิร์ฟเวอร์จะแบ่งตามประเภทการใช้งาน 30

บทที่ 6 ทฤษฎีและความรู้พื้นฐานเกี่ยวกับระบบฐานข้อมูล 31
6.1 ระบบจัดการฐานข้อมูลเชิงวัตถุสัมพันธ์ 31
6.1.1 ความสามารถของระบบจัดการฐานข้อมูลเชิงสัมพันธ์ 31
6.1.2 ความสามารถของระบบจัดการฐานข้อมูลแบบออบเจ็กต์ 31
6.2 ชนิดข้อมูลมาตรฐาน 32
6.2.1 ชนิดข้อมูลแบบตัวอักษร (Character Datatypes) 32
6.2.2 ชนิดข้อมูลแบบตัวเลข (Number Datatype) 33
6.2.3 ชนิดข้อมูลที่เป็นวันที่ (Date Datatype) 33
6.2.4 ชนิดข้อมูลที่มีขนาดใหญ่ (LOB Datatypes) 33
6.2.5 RAW และ LONG RAW Datatypes 33
6.2.6 ROWID Datatype 34
6.3 ชนิดข้อมูลแบบกำหนดเอง 34
6.3.1 ชนิดข้อมูลแบบออบเจ็กต์ 34
6.3.2 ชนิดข้อมูลแบบกลุ่ม 34
6.4 ฐานข้อมูล (Database) พื้นที่ตาราง (Tablespaces) และไฟล์ข้อมูล (Datafiles) 35
6.5 Java Database Connectivity 35
6.5.1 ความหมายของ JDBC 36
6.5.2 รูปแบบการเชื่อมต่อฐานข้อมูลของ JDBC 36
6.5.3 โครงสร้างของ JDBC 38
6.5.4 รูปแบบของ JDBC ไดรเวอร์ (JDBC Driver) 38

บทที่ 7 หลักการทำงานเบื้องต้นของ Oracle Application Server 41
7.1 ส่วนประกอบของ OAS 41
7.2 หลักการเบื้องต้นของแอพพลิเคชันในมุมมองของ OAS 43
7.2.1 Cartridge และแอพพลิเคชัน 43
7.3 แอพพลิเคชันในลักษณะ Jservlet Cartridge 45
7.4 Oracle9iAS Portal 46
7.4.1 ทำไมต้องใช้ Portal ? 46

7.4.2 คุณสมบัติของ Oracle9iAS Portal 46
7.4.3 Oracle Portal ทำงานอย่างไร 48

บทที่ 8 ระบบบริหารจัดการ Portal site 49
8.1 ประโยคปัญหา (Problem Statement) 49
8.2 Use Case Model 49
8.2.1 กำหนดผู้กระทำ (Actors) 49
8.2.2 กำหนด Use Case 50
8.3 Class Diagram 61
8.4 การออกแบบส่วนติดต่อกับผู้ใช้ (User Interface Design) 62
8.4.1 บุคคลทั่วไป (Public User) 62
8.4.2 ผู้ใช้ (Registered User) 64
8.4.3 ผู้ดูแลระบบ (Administrator) 68

บทที่ 9 บทวิจารณ์และสรุป 80
9.1 บทวิจารณ์และสรุป 80
9.1.1 ส่วนการศึกษาระบบที่มีอยู่เดิม 80
9.1.2 ส่วนการวิเคราะห์และการออกแบบระบบ 80
9.1.3 ส่วนการพัฒนาระบบ 80
9.1.4 ส่วนวิจารณ์และบทสรุปโดยรวมของระบบ 81
9.2 ปัญหาและแนวทางในการพัฒนาระบบ 81
9.2.1 ปัญหาเกี่ยวกับการแสดงผลภาษาไทย 81
9.2.2 ปัญหาเกี่ยวกับแหล่งอ้างอิง 81
9.2.3 การรักษาความปลอดภัยของข้อมูล 81

ภาคผนวก ก ขั้นตอนการลงโปรแกรม Oracle 8i Database (8.1.7) และ โปรแกรม Oracle 9iAS (1.0.2.1) บนระบบปฏิบัติการ Window 2000 Server 82

ภาคผนวก ข การใช้งาน Oracle9iAS Portal 112

บรรณานุกรม 138

สารบัญรูปภาพ

หน้าที่

บทที่ 2 หลักการพัฒนาโปรแกรมด้วยวิธีเชิงวัตถุ (Object Oriented Programming)

2-1 รูปการแปลภาษาอื่นๆ เป็นภาษาเครื่อง 6

2-2 รูปการแปลงภาษาจาวาเป็นภาษาเครื่อง 7

2-3 การทำงานของจาวาแอพเพล็ต 7

2-4 การทำงานของจาวาเซิร์ฟเล็ต 8

บทที่ 3 จาวาเซิร์ฟเวอร์เพจ (Java Server Page)

3-1 การทำงานของ JSP 10

บทที่ 4 การวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุ (Object Oriented Analysis & Design)

4-1 แผนภาพต่างๆ ใน UML 19

4-2 ตัวอย่างแผนภาพ Use Case 20

4-3 ตัวอย่างแผนภาพ Sequence 21

4-4 ตัวอย่างแผนภาพ Collaboration 22

4-5 สัญลักษณ์แสดงความสัมพันธ์แบบแอซโซซิเอชัน 23

4-6 สัญลักษณ์แสดงความสัมพันธ์แบบเป็นส่วนหนึ่งของ 23

4-7 สัญลักษณ์แสดงความสัมพันธ์แบบคอมโพซิชัน 23

4-8 สัญลักษณ์แสดงความสัมพันธ์แบบการสืบทอดคุณสมบัติ 24

4-9 ตัวอย่าง Constrain 25

4-10 รูปแบบการเขียนเมสเซจใน UML 26

บทที่ 5 ระบบเครือข่ายแบบไคลเอ็นต์/เซิร์ฟเวอร์ (Client/Server)

5-1 การทำงานในลักษณะ Host-based System 27

5-2 การทำงานในลักษณะไคลเอ็นต์/เซิร์ฟเวอร์ 28

5-3 การจัดแบ่งไคลเอ็นต์เซิร์ฟเวอร์แบบ 1 Tier 29

5-4 การจัดแบ่งไคลเอ็นต์เซิร์ฟเวอร์แบบ 2 Tiers 29

5-5 การจัดแบ่งไคลเอ็นต์เซิร์ฟเวอร์แบบ 3 Tiers 30

บทที่ 6 ทฤษฎีและความรู้พื้นฐานเกี่ยวกับระบบฐานข้อมูล

6-1 พื้นที่ตารางและไฟล์ข้อมูล 35

6-2 Two – Tiers Model 37

6-3 Three – Tiers Model 37

6-4 ระดับการเชื่อมต่อของ JDBC API 38
6-5 JDBC Driver ประเภทที่ 3 และ 4 ที่ใช้ pure Java JDBC driver 39
6-6 JDBC Driver ประเภทที่ 1 และ 2 39

บทที่ 7 หลักการทำงานเบื้องต้นของ Oracle Application Server
7-1 ส่วนประกอบในเว็บไซต์ของ OAS 41
7-2 ตัวอย่างเส้นทางการร้องขอ 42
7-3 ตัวอย่างแอพพลิเคชันและ Cartridge 44
7-4 แสดงลักษณะ portlet บนเว็บเพจ 47
7-5 แสดง Content Areas, Pages และ Applications ที่เป็นองค์ประกอบหลักๆ ของ Oracle Portal 48

บทที่ 8 ระบบบริหารจัดการ Portal site
8-1Use Case Model ของระบบบริหารจัดการพอร์ทอลไซต์ 50
8-2 Sequence Diagram : Browse Webpage 51
8-3 Sequence Diagram : Register 52
8-4 Sequence Diagram : Create User 53
8-5 Sequence Diagram : Log in 54
8-6 Sequence Diagram :Change User Data 55
8-7 Sequence Diagram : Create Webpage 56
8-8 Sequence Diagram : Browse Webpage Data Report 57
8-9 Sequence Diagram : Log out 58
8-10 Sequence Diagram :Browse Login Data Report 59
8-11 Sequence Diagram : Browse Event Data Report 60
8-12 Class Diagram ของระบบบริหารจัดการพอร์ทอลไซต์ 61
8-13 แสดงหน้าแรกของพอร์ทอลไซต์ 62
8-14 ตัวอย่างสารบัญผลงานทางวิชาการของนักศึกษาปีการศึกษา 2541 63
8-15 แสดงตัวอย่างเนื้อหาผลงานทางวิชาการของนักศึกษากลุ่มหนึ่งที่ได้สร้างไว้บนเว็บไซต์นี้ 63
8-16 แสดงหน้าจอการลงทะเบียน 64
8-17 แสดงวิธีการล็อกอินเพื่อเข้าใช้งานในฐานะผู้ใช้ของเว็บไซต์ 65
8-18 แสดงหน้าตาเว็บเพจ หลังผู้ใช้ล็อกอินแล้ว 66
8-19 แท็บ Personal ทำให้ผู้ใช้สามารถสร้างเว็บเพจของตัวเองได้ 66
8-20 แสดงตัวอย่าง Browse webpage data report 67

8-21 เลือก Accout_Info เพื่อเปลี่ยนแปลงข้อมูลส่วนตัว 67
8-22 หน้าจอสำหรับเปลี่ยนแปลงข้อมูลส่วนตัว รวมถึงการเปลี่ยนรหัสผ่าน 68
8-23 แสดงแท็บMain Page ที่ปรากฏแก่ Portal Administrator ของพอร์ทอลไซต์นี้ 69
8-24 แสดงแท็บMain Page ที่ Portal Administrator เรียกดูรายละเอียดของผู้ลงทะเบียน 69
8-25 แสดงแท็บ Administrator สำหรับ Portal Administrator ของพอร์ทอลไซต์นี้ 70
8-26 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟแท่งตามรายชื่อผู้ใช้ 71
8-27 แสดงผลกราฟแท่งที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้ 72
8-28 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟเส้นตามรายชื่อผู้ใช้ 72
8-29 แสดงผลกราฟเส้นที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้ 73
8-30 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟวงกลมตามรายชื่อผู้ใช้ 73
8-31 แสดงผลกราฟวงกลมที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้ 74
8-32 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟแท่งตามช่วงเวลา 74
8-33 แสดงผลกราฟแท่งที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามช่วงเวลา 75
8-34 แสดงฟอร์มขอดูสถิติในการล๊อกอินของผู้ใช้แบบกราฟแท่ง ตามรายชื่อผู้ใช้ 76
8-35 แสดงผลกราฟแท่งที่บอกสถิติในการล๊อกอินของผู้ใช้ตามรายชื่อผู้ใช้ 76
8-36 แสดงฟอร์มขอดูสถิติในการล๊อกอินของผู้ใช้แบบกราฟเส้นตามกลุ่มผู้ใช้ 77
8-37 แสดงผลกราฟเส้นที่บอกสถิติในการล๊อกอินของผู้ใช้ ตามกลุ่มผู้ใช้ 77
8-38 แสดงฟอร์มขอดูรายงานการกระทำต่างๆ ของผู้ใช้ 78
8-39 แสดงผลตารางที่รายงานการกระทำของผู้ใช้ 79

สารบัญตาราง

หน้าที่

บทที่ 4 การวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุ (Object Oriented Analysis & Design)

4-1 ตัวอย่างของ Stereotype 26

บทที่ 7 หลักการทำงานเบื้องต้นของ Oracle Application Server

7-1 หมายเลขพอร์ตปกติที่ใช้ใน OAS 42

7-2 แสดงลักษณะต่างๆ ของ Cartridge 45

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

วิวัฒนาการของคอมพิวเตอร์ได้ถูกพัฒนาเข้าสู่ระบบ World Wide Web และทำให้วิวัฒนาการในด้านภาษาโปรแกรมมิ่งต้องปรับตัวเข้าสู่ระบบใหม่อย่างรวดเร็ว ดังจะเห็นได้ว่า ภาษาโปรแกรมมิ่งยุคใหม่เริ่มปรับเข้าสู่อินเตอร์เน็ต (Internet) มากขึ้น ดังเช่น ภาษาจาวา (Java) ที่แต่เดิมทีมงานสร้างภาษานี้ขึ้นเพื่อให้ทำงานได้กับในหลาย ๆ แพลตฟอร์ม (Platform) และสามารถที่จะผลิตซอฟต์แวร์ที่สามารถฝังตัวอยู่ในผลิตภัณฑ์อิเล็กทรอนิกส์ของลูกค้าได้ แต่ปัจจุบันภาษาจาวาได้ถูกพัฒนาการโปรแกรมบนเครือข่ายอินเตอร์เน็ตทั้ง Java Servlet, Java applet, Java Script, Java Bean และ JSP เป็นต้น ดังนั้นหลายเว็บไซต์ได้หันมาใช้ภาษาจาวา ซึ่งมีลักษณะเป็น Object Oriented Programming มาใช้ในการพัฒนา

เมื่ออินเตอร์เน็ตเติบโตอย่างรวดเร็ว ปัญหาต่างๆที่เกิดขึ้นกับข้อมูลต่างๆขององค์กรที่เผยแพร่ผ่านทางเว็บไซต์ก็เกิดขึ้น ไม่ว่าจะเป็นปัญหา Information Overload ที่เกิดจากปริมาณข้อมูลมากมายแต่ไม่สามารถจัดสรรหรือแสดงผลให้กับบุคคลได้อย่างเหมาะสม, System management complexities รวมถึงปัญหาการจัดการและบำรุงรักษาเว็บไซต์ให้สามารถดำเนินกิจการอยู่ได้โดยสามารถดึงดูดให้ผู้เข้าเยี่ยมชมหวนกลับมาใช้บริการข้อมูลข่าวสารผ่านทางเว็บไซต์นั้นได้อีก กลายเป็นปัญหาสำคัญของแต่ละเว็บไซต์ที่ลงทุนทางด้าน e-business อย่างมาก

ดังนั้นหากเว็บไซต์แต่ละแห่งสามารถเก็บสถิติของผู้เข้าเยี่ยมชมในด้านต่างๆ ผู้บริหารเว็บไซต์ก็สามารถรู้ได้ว่าผู้เยี่ยมชมส่วนใหญ่ต้องการอะไร แล้วนำข้อมูลเหล่านั้นมาใช้ช่วยตัดสินใจในการวางนโยบายในการปรับปรุงและพัฒนาเว็บไซต์เพื่อสร้างความพึงพอใจให้กับผู้เข้าเยี่ยมชมในระยะยาวได้

งานวิจัยนี้ได้พัฒนาระบบให้บริการจัดเก็บสถิติและแสดงผลสำหรับพอร์ทอลไซต์แบบออนไลน์เพื่อเป็นข้อมูลที่สำคัญที่ใช้ในการประกอบการตัดสินใจในการวางนโยบายสำหรับนักพัฒนาเว็บไซต์ โดยไม่ลืมที่จะคำนึงถึงการใช้งานที่ง่าย สะดวกรวดเร็วสำหรับผู้พัฒนาเว็บไซต์อีกด้วย

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาและนำเสนอโมเดลการพัฒนาโปรแกรมเชิงวัตถุด้วย UML (Unified Modeling Language) โดยใช้โปรแกรม Rational Rose เป็น CASE tool ช่วยในการสร้างโมเดล

1.2.2 เพื่อศึกษาการสร้างฐานข้อมูลรีเลชันแนลจากออบเจ็กต์ที่ได้จากการพัฒนาเชิงวัตถุ

1.2.3 เพื่อศึกษาการทำงานของ JSP (JavaServer Pages)

1.2.4 เพื่อศึกษาการใช้งาน Oracle9iAS Portal

1.2.5 เพื่อให้สามารถออกแบบและพัฒนาเว็บแอพพลิเคชันสำหรับการบริหารงานพอร์ทอลไซต์ ด้วย JSP ได้

1.3 ขอบเขตของโครงงาน

โครงงานนี้จะออกแบบและสร้างโปรแกรมประยุกต์เว็บเพจ (Web-base Application) เพื่อการบริหารงานพอร์ทอลไซต์โดยมีฟังก์ชันการทำงานหลักที่เกี่ยวข้องกับการบริหารและจัดการเว็บไซต์ที่ทำหน้าที่รับฝากไดนามิกเว็บเพจ โดยจะมีระบบเว็บเซิร์ฟเวอร์ของ OAS (Oracle9iAS Application Server) เสริมด้วย Oracle9iAS Portal เพื่อการสร้างพอร์ทอลไซต์ที่ง่ายขึ้น และระบบฐานข้อมูลของออราเคิล (Oracle8iEEDatabase) จะทำงานภายใต้ระบบปฏิบัติการ Window 2000 Server ส่วนไคลเอ็นต์จะติดต่อกับเซิร์ฟเวอร์ผ่านทางเว็บไซต์ทางอินเตอร์เน็ต

1.4 วิธีการดำเนินงาน

เริ่มจากการศึกษาทฤษฎีพื้นฐานต่างๆ ที่เกี่ยวข้อง โดยมีเรื่องหลักๆ คือ ภาษาจาวา, สถาปัตยกรรมของระบบไคลเอ็นต์เซิร์ฟเวอร์, ทฤษฎีการพัฒนาเชิงวัตถุด้วยโมเดล UML จากนั้นติดตั้งเว็บเซิร์ฟเวอร์และดาต้าเบสเซิร์ฟเวอร์ ศึกษาการใช้งาน Oracle9iAS Portal วิเคราะห์ความต้องการและออกแบบระบบนำเสนอด้วยโมเดล UML และทำการออกแบบฐานข้อมูลแบบรีเลชันแนล

บทที่ 2
หลักการพัฒนาโปรแกรมด้วยวิธีเชิงวัตถุ
(Object Oriented Programming)

เทคโนโลยีในการพัฒนาระบบด้วยวิธีเชิงวัตถุเป็นหลักการทางวิศวกรรม ซึ่งบรรจุไว้ด้วยองค์ประกอบต่างๆ ที่เรียกว่าโมเดลของวัตถุ (Object Model) ในโมเดลของวัตถุนั้นแบ่งออกเป็นการแยกแยะเอกลักษณ์ (Abstraction) การซ่อนรายละเอียด (Encapsulation) การรวมกลุ่มความสัมพันธ์ (Modularity) ลำดับชั้น (Hierarchy) ชนิด (Typing) การทำงานพร้อมกัน (Concurrency) และการรักษาสถานะ (Persistence) สำหรับลักษณะการวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุนั้น จะแตกต่างจากการออกแบบระบบด้วยวิธีโครงสร้าง เนื่องจากรูปแบบในการคิดและการมองระบบจะต่างกัน นอกจากนั้นการออกแบบและพัฒนาระบบแบบโครงสร้างจะขึ้นอยู่กับภาษาในการเขียนโปรแกรมแบบโครงสร้างด้วย ส่วนการวิเคราะห์และออกแบบระบบเชิงวัตถุก็จะขึ้นอยู่กับภาษาในการเขียนโปรแกรมเชิงวัตถุเช่นกัน และในภาษาสำหรับการเขียนโปรแกรมเชิงวัตถุแต่ละภาษาก็มีรายละเอียดปลีกย่อยที่ต่างกันอีกด้วย

2.1 หลักการของวัตถุ

การพัฒนาระบบแบบโครงสร้างนั้น จะพยายามให้นักพัฒนาระบบแก้ปัญหาด้วยการแบ่งปัญหาออกเป็นส่วนๆ แล้วแก้ปัญหาในแต่ละส่วนด้วยอัลกอริทึม (Algorithm) และโครงสร้างข้อมูล (Data Structure) เฉพาะ สำหรับการพัฒนาระบบด้วยวิธีเชิงวัตถุก็เช่นเดียวกัน แต่จะมีมุมมองต่อปัญหาเป็นวัตถุประกอบกับการเขียนโปรแกรมเชิงวัตถุ การพัฒนาระบบด้วยวิธีเชิงวัตถุไม่เพียงจะเป็นแนวความคิดใหม่ในการเขียนโปรแกรมเท่านั้นแต่ยังมีข้อดีจากโมเดลของวัตถุ ซึ่งเป็นแนวความคิดใหม่ในวงการคอมพิวเตอร์อีกด้วย สามารถนำหลักการในการพัฒนาระบบด้วยวิธีเชิงวัตถุไปประยุกต์ใช้กับการออกแบบส่วนติดต่อกับผู้ใช้ (User Interface) ระบบฐานข้อมูลหรือแม้แต่การออกแบบสถาปัตยกรรมฮาร์ดแวร์ของคอมพิวเตอร์ เนื่องจากหลักการของการพัฒนาระบบด้วยวิธีเชิงวัตถุนั้นเป็นแนวความคิดในการจัดการกับความซับซ้อนของสิ่งต่างๆ อย่างมีระบบ ซึ่งสามารถนำไปประยุกต์ใช้ในงานต่างๆ ได้อย่างกว้างขวาง

แนวความคิดของการพัฒนาระบบด้วยวิธีเชิงวัตถุเกิดขึ้นครั้งแรกเมื่อประมาณ 20 ปีที่ผ่านมา โดยเริ่มจากการออกแบบสถาปัตยกรรมฮาร์ดแวร์ของคอมพิวเตอร์ นักพัฒนาฮาร์ดแวร์ได้ใช้แนวความคิดของการพัฒนาระบบด้วยวิธีเชิงวัตถุช่วยในการลดช่องว่างระหว่างการแยกแยะเอกลักษณ์ในระดับสูง ซึ่งเป็นพวกการเขียนโปรแกรม และการแยกแยะเอกลักษณ์ในระดับล่างที่เป็นการทำงานของตัวฮาร์ดแวร์ ด้วยการนำหลักการของการพัฒนาระบบด้วยวิธีเชิงวัตถุมาใช้ ทำให้การออกแบบและการพัฒนาฮาร์ดแวร์สามารถตรวจสอบความผิดพลาดได้ง่าย ช่วยเพิ่มประสิทธิภาพในการปฏิบัติคำสั่ง ลดชนิดของคำสั่งของฮาร์ดแวร์ มีการคอมไพล์จากภาษาระดับสูงเป็นภาษาระดับต่ำที่ไม่ซับซ้อน และยังช่วยลดพื้นที่ในการเก็บข้อมูลอีกด้วย

2.2 คุณสมบัติของหลักการเชิงวัตถุ

2.2.1 Encapsulation

Encapsulation หมายถึง การแยกส่วนลักษณะภายนอก (Interface) ที่ติดต่อกับวัตถุอื่นๆ ออกจากส่วนที่ทำงาน (Implement) ภายในวัตถุซึ่งจะถูกใช้เฉพาะตัวภายในวัตถุ เพื่อป้องกันไม่ให้ภายนอกเข้าถึงข้อมูลโดยตรง ให้เข้าถึงโดยผ่านเมธอด (Method) ที่จัดไว้ให้ ทำให้มีความปลอดภัยและง่ายต่อการควบคุมดูแลวัตถุ (Object Maintenance)

2.2.2 Inheritance

Inheritance คือการถ่ายทอดคุณสมบัติและความสามารถของคลาสหนึ่งไปยังอีกคลาสหนึ่ง ลักษณะความสัมพันธ์ของคลาสที่ใช้ข้อมูลและพฤติกรรมร่วมกันโดยมีความสัมพันธ์ระหว่างคลาสในลักษณะความสัมพันธ์แบบ "is-a" ซึ่งมีหลักการในการถ่ายทอดคุณสมบัติของคลาสอันประกอบด้วยแอตทริบิวต์ (Attribute) และเมธอด (Method) จากคลาสที่มีลำดับสูงกว่า (Super Class) มายังคลาสที่มีลำดับต่ำกว่า (Sub Class) ทำให้เมื่อเปลี่ยนแอตทริบิวต์หรือเมธอดที่ Super Class จะมีผลทำให้ค่าใน Sub Class เปลี่ยนไปด้วย ทำให้ซอฟต์แวร์มีความสามารถนำกลับมาใช้ได้ (Reusable)

2.2.3 Polymorphism

Polymorphism คือ คุณสมบัติของคลาสที่แบ่งแยกลักษณะการทำงานวัตถุ ให้มีความสามารถในการทำงานในหลายๆรูปแบบ การส่งแมสเซจเดียวกันให้วัตถุที่ต่างกันทำให้วัตถุสามารถแสดงการทำงานที่ต่างกันออกมาได้ คุณสมบัตินี้ทำให้ซอฟต์แวร์มีความยืดหยุ่น (Flexible)

2.3 ภาษาในการเขียนโปรแกรมเชิงวัตถุ

การเขียนโปรแกรมด้วยภาษาเชิงวัตถุ เป็นวิธีการในการสร้างโปรแกรม โปรแกรมจะถูกจัดให้มีโครงสร้างในการทำงานร่วมกันของวัตถุ ซึ่งวัตถุนั้นจะเป็นตัวตนหรือเป็นอินสแทนซ์ (Instance) ของคลาส และคลาสก็เป็นส่วนหนึ่งของลำดับชั้น (Hierarchy) ของคลาสที่มีความสัมพันธ์ในการสืบทอดคุณสมบัติ (Inheritance) โดยภาษาที่จะเป็นภาษาในการเขียนโปรแกรมเชิงวัตถุได้นั้นจะต้องมีคุณสมบัติ 3 ข้อ คือ

2.3.1 จะต้องเป็นภาษาที่ใช้วัตถุมิใช่อัลกอริทึมในการแก้ปัญหาเป็นหลัก

ถ้าเป็นภาษาในการเขียนโปรแกรมแบบเก่า หากเขียนโปรแกรมติดต่อกับอุปกรณ์สื่อสารโมเด็มก็จะแบ่งข้อมูลส่วนต่างๆติดต่อกับโมเด็มตามฟังก์ชันการทำงานต่างๆ เช่น รับ ส่ง หรือตรวจสอบข้อมูล แต่ถ้าเป็นภาษาในการเขียนโปรแกรมเชิงวัตถุแล้วผู้เขียนโปรแกรมจะต้องมองโมเด็มให้เป็นวัตถุตัวหนึ่ง โดยในวัตถุโมเด็มก็อาจจะประกอบไปด้วยวัตถุย่อยๆ และวัตถุย่อยๆ เหล่านี้ก็จะมีการสื่อสารซึ่งกันและกัน สรุปคือ เขียนโปรแกรมโดยดูที่ตัววัตถุเป็นหลัก

2.3.2 วัตถุที่ปรากฏในการเขียนโปรแกรมต้องเป็นอินสแทนซ์ของคลาส

เนื่องจากว่าหลักการของการเขียนโปรแกรมเชิงวัตถุมองวัตถุเป็นหลักซึ่งวัตถุนี้ไม่สามารถกำหนดขึ้นมาลอยๆ ได้ จำเป็นต้องมีลักษณะคุณสมบัติของตัวมันเองซึ่งจะถูกกำหนดไว้ในคลาส ดังนั้นวัตถุจะต้องเป็นอินสแทนซ์ของคลาส

2.3.3 คลาสจะต้องมีความสามารถในการถ่ายทอดคุณสมบัติ

นอกเหนือจากหน้าที่ในการกำหนดคุณสมบัติของคลาส และมีตัวแทนของคลาสที่เรียกว่าอินสแทนซ์แล้ว แต่ละคลาสจำเป็นต้องมีความสัมพันธ์กันด้วย คุณสมบัติที่สำคัญต่อการกำหนดความสัมพันธ์ของคลาสคือคลาสต้องถ่ายทอดคุณสมบัติได้ หากขาดคุณสมบัติข้อนี้ไปก็ไม่มีประโยชน์อันใดในการพัฒนาระบบเชิงวัตถุขึ้นมา

ถ้าภาษาใดขาดคุณสมบัติหนึ่งในสามข้อนี้ไปก็จะไม่เรียกว่าเป็นภาษาในการเขียนโปรแกรมเชิงวัตถุ ตัวอย่างเช่น ถ้าภาษาใดสนับสนุนการสร้างคลาสและวัตถุ แต่ไม่สนับสนุนการสืบทอดคุณสมบัติของคลาสแล้วเราจะเรียกภาษานั้นว่า ภาษาที่ทำงานบน Abstract Data Type หรือเรียกว่าภาษาในการเขียนโปรแกรมบนพื้นฐานของวัตถุ (Object Based) มิใช่ภาษาในการเขียนโปรแกรมเชิงวัตถุ(Object Oriented)

ตัวอย่างของภาษาในการเขียนโปรแกรมเชิงวัตถุ เช่น Smalltalk, Object Pascal, C++, Eiffel, CLOS และ Java ส่วนภาษาในการเขียนโปรแกรมบนพื้นฐานของวัตถุก็ เช่น Ada เป็นต้น ซึ่งในการที่จะเขียนโปรแกรมด้วยภาษาเชิงวัตถุให้ได้อย่างมีประสิทธิภาพ ผู้เขียนโปรแกรมควรจะต้องมีการออกแบบด้วยหลักการพัฒนาระบบด้วยวิธีเชิงวัตถุเช่นกัน

2.4 ภาษาจาวา

จาวาเป็นภาษาโปรแกรมขั้นสูงที่พัฒนาขึ้นมาโดยบริษัท Sun Microsystems โดยมีวัตถุประสงค์เพื่อให้เป็นภาษาระดับสูงที่ใช้พัฒนาซอฟต์แวร์สำหรับอุปกรณ์ที่ควบคุมโดยไมโครโพรเซสเซอร์ (Embedded System) ที่สามารถนำไปรันได้โดยไม่ขึ้นกับแพลตฟอร์ม (Platform)

ภาษาจาวาได้รับการยอมรับว่าเป็นภาษาที่ดีที่สุดในการพัฒนาซอฟต์แวร์ที่ต้องการความปลอดภัยสูงแบบกระจายผ่านระบบเครือข่าย โดยสามารถใช้พัฒนาได้ตั้งแต่โปรแกรมของอุปกรณ์เน็ตเวิร์กที่ควบคุมโดยไมโครโพรเซสเซอร์ โปรแกรมในเว็บบราวเซอร์ จนถึงแอพพลิเคชันทั่วไป ตัวภาษาจาวามีลักษณะเด่นดังนี้

- เป็นภาษาพัฒนาโปรแกรมเชิงวัตถุ
- สามารถนำไปใช้งานได้โดยไม่ขึ้นกับแพลตฟอร์ม
- สามารถประมวลผลแบบกระจาย โดยดึงคลาสไลบรารีจากที่ต่างๆ ผ่านทาง HTTP และ FTP
- สามารถทำงานได้หลายงานพร้อมกันในเวลาเดียวกัน
- ไม่มีตัวแปรแบบพอยน์เตอร์ (Pointer) เพื่อขจัดปัญหาในการจัดการหน่วยความจำ
- สามารถตรวจสอบและจัดการกับความผิดพลาด (Exception) ที่เกิดขึ้น
- มีการรองรับมาตรฐานความปลอดภัยในการใช้งานรูปแบบต่างๆ

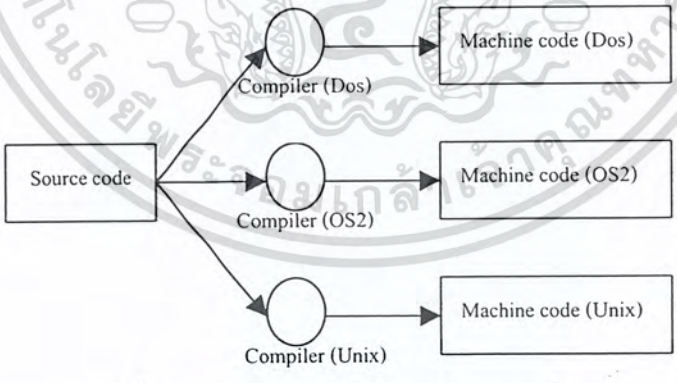
2.5 หลักการทำงานของภาษาจาวา

จาวาเป็นภาษาระดับสูงเชิงวัตถุซึ่งมีกลไกการแปลภาษาและการประมวลผลในลักษณะผสมผสานระหว่างแบบคอมไพเลอร์ (Compiler) และแบบอินเทอร์พรีเตอร์ (Interpreter) เพื่อจุดประสงค์ที่จะรับข้อดีของทั้งสองแบบ

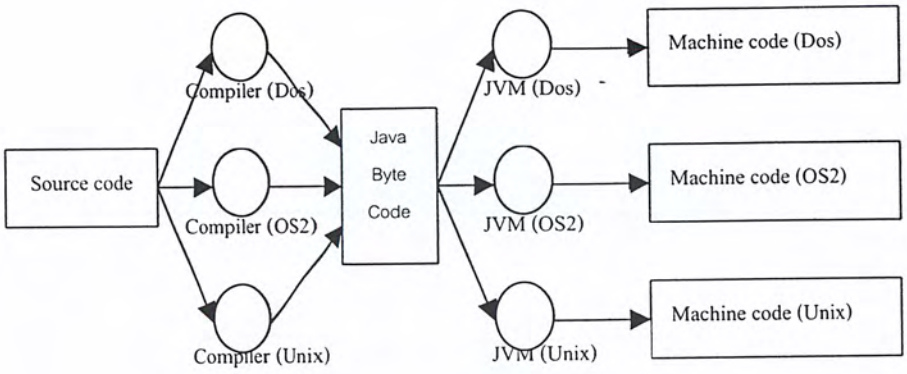
แบบคอมไพเลอร์นั้น ตัวคอมไพเลอร์จะทำหน้าที่วิเคราะห์โค้ดภาษาระดับสูงและแปลเป็นภาษาเครื่อง (Machine code) ซึ่งจะมีข้อดีคือ จะได้โปรแกรมภาษาเครื่องที่มีประสิทธิภาพ เนื่องจากตัวคอมไพเลอร์ได้ทำการวิเคราะห์และปรับแต่งตัวโค้ดให้ได้ภาษาเครื่องที่มีประสิทธิภาพที่สุด แต่ข้อเสียของแบบนี้คือภาษาเครื่องที่ได้จะไม่สามารถนำไปใช้กับแพลตฟอร์มอื่นได้

ส่วนแบบอินเทอร์พรีเตอร์นั้น ตัวอินเทอร์พรีเตอร์จะทำการอ่านโค้ดภาษาระดับสูงทีละบรรทัด แล้วแปลเป็นภาษาเครื่องและทำงานทันที แล้วก็จะกลับไปอ่านโค้ดบรรทัดถัดไป ทำอย่างนี้ไปจนจบโปรแกรม วิธีนี้จะต้องทำการอ่านโค้ดและแปลเป็นภาษาเครื่องสลับกันไป ทำให้ทำงานได้ช้ากว่าแบบคอมไพเลอร์ แต่ก็มีข้อดีคือตัวอินเทอร์พรีเตอร์สามารถสร้างได้ง่ายกว่าตัวคอมไพเลอร์

ภาษาจาวาได้นำข้อดีของทั้งสองแบบมารวมกัน โดยเมื่อทำการแปลโค้ด ตัวจาวาคอมไพเลอร์ (Java Compiler) จะทำการแปลโค้ดภาษาจาวาเป็นไบต์โค้ด (Byte Code) ที่ไม่ขึ้นกับแพลตฟอร์ม และเมื่อจะทำการประมวลผล ตัวจาวาเวอร์ชวลแมชชีน (Java Virtual Machine) จะทำการแปลไบต์โค้ดทีละคำสั่งเป็นภาษาเครื่องของแพลตฟอร์มที่กำลังทำงานแล้วส่งให้ตัวประมวลผลทำงานตามคำสั่ง วิธีการนี้ทำให้โปรแกรมที่เขียนในภาษาจาวาสามารถทำงานได้ในลักษณะ "เขียนครั้งเดียว ใช้งานได้ทุกที่" คือไม่ว่าจะทำการคอมไพล์โค้ดในแพลตฟอร์มใด ก็สามารถนำไบต์โค้ดที่ได้ไปใช้กับแพลตฟอร์มอื่นที่มีตัวจาวาเวอร์ชวลแมชชีนสำหรับใช้ประมวลผล



รูปที่ 2-1 รูปการแปลภาษาอื่นๆ เป็นภาษาเครื่อง



รูปที่ 2-2 รูปการแปลภาษาจาวาเป็นภาษาเครื่อง

2.6 จาวาสคริปต์ (Javascript)

จาวาสคริปต์เป็นภาษาสคริปต์ที่เกิดขึ้นมาจากความนิยมในอินเตอร์เน็ต และเป็นหนึ่งในภาษาสำหรับอินเตอร์เน็ตที่มีการยอมรับและได้รับความนิยมอย่างสูง ด้วยลักษณะของภาษาสคริปต์ที่ง่ายในการศึกษา เรียนรู้และทำความเข้าใจ แต่ให้ศักยภาพและความสามารถเทียบเท่ากับภาษาระดับสูง

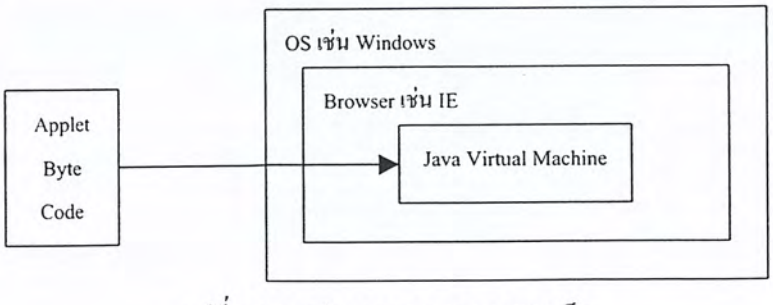
การเขียนจาวาสคริปต์นั้นมีข้อดีที่สามารถใช้บนระบบปฏิบัติการใดๆ ก็ได้และยังสามารถกระโดดไปมาระหว่างระบบปฏิบัติการได้อีกด้วย ซึ่งจะเป็นประโยชน์ในกรณีที่เราต้องการที่จะให้มีการทำงานของสคริปต์ภายใต้บราวเซอร์ในระบบปฏิบัติการต่างๆ โดยจาวาสคริปต์ทำงานได้ทั้งฝั่งไคลเอ็นต์และฝั่งเซิร์ฟเวอร์

ถ้าจะเปรียบเทียบการใช้งานระหว่างจาวาสคริปต์กับภาษาจาวาแบบอื่นแล้วอาจจะพิจารณาได้จากเงื่อนไขต่างๆ เช่น ถ้าต้องการความง่ายและรวดเร็วก็ใช้จาวาสคริปต์ แต่ถ้าต้องการโปรแกรมที่มีความซับซ้อนก็จะใช้ภาษาจาวาแบบอื่นซึ่งแน่นอนกว่า

2.7 จาวาแอพเพล็ต (Java Applet)

จาวาแอพเพล็ตคือโปรแกรมคอมพิวเตอร์ที่เขียนด้วยภาษาจาวาที่ถูกแปลอยู่ในรูปของไบต์โค้ดด้วยการคอมไพล์ และการทำงานของโปรแกรมจะทำงานผ่านจาวาเวอร์ชวลแมชชีนซึ่งอยู่บนบราวเซอร์ที่สนับสนุนการใช้งานจาวาแอพเพล็ต

ข้อแตกต่างอีกประการของการสร้างแอพเพล็ตด้วยภาษาจาวากับภาษาจาวาแบบอื่นคือ ตัวจาวาแอพเพล็ตต้องถูกสืบทอดมาจากคลาส Applet

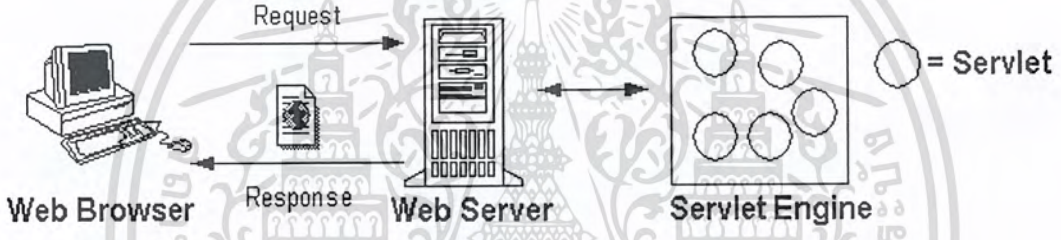


รูปที่ 2-3 การทำงานของจาวาแอพเพล็ต

2.8 จาวาเซิร์ฟเล็ต (Java Servlet)

จาวาเซิร์ฟเล็ตเป็นโปรแกรมที่ทำงานบนเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นเซิร์ฟเวอร์ โดยจาวาเซิร์ฟเล็ตนั้นจะทำการประมวลผลบนเครื่องเซิร์ฟเวอร์โดยจะทำงานควบคู่กับเครื่องของผู้ใช้ และเมื่อผู้ใช้ต้องการประมวลผลก็ส่งคำร้องให้จาวาเซิร์ฟเล็ตที่อยู่บนเซิร์ฟเวอร์ทำงานแล้วส่งผลลัพธ์กลับมาให้เครื่องของผู้ใช้อีกที ทำให้เครื่องของผู้ใช้ไม่จำเป็นต้องเป็นเครื่องที่มีประสิทธิภาพมากนัก

จาวาเซิร์ฟเล็ตนั้นอ้างอิงหลักการของ CGI โดยข้อดีของจาวาเซิร์ฟเล็ตที่อยู่เหนือ CGI อย่างแรกก็คือตัวภาษาที่ใช้เขียนซึ่งก็คือภาษาจาวานั่นเอง นอกจากนี้จาวาเซิร์ฟเล็ตยังมีความเร็วที่เหนือกว่า CGI เพราะเซิร์ฟเล็ตใช้หลักการของเธรดโดยจะทำการสร้าง 1 เธรดต่อหนึ่งคำร้องที่มาจากไคลเอ็นต์ แต่ตัว CGI จะทำการสร้าง 1 โพรเซสต่อหนึ่งคำร้อง ซึ่งจะสิ้นเปลืองทรัพยากรของระบบมากกว่า ถึงแม้ว่าเซิร์ฟเล็ตจะใช้หลักการของ CGI แต่ตัวเว็บเซิร์ฟเวอร์จะไม่สามารถส่งข้อมูลไปให้เซิร์ฟเล็ตโดยตรงได้เหมือนกับ CGI แต่ต้องทำการเพิ่มส่วนที่ใช้เป็นเสมือนตัวห่อหุ้มเซิร์ฟเล็ตต่างๆ ไว้ โดยส่วนที่เพิ่มขึ้นมานี้เราเรียกว่าเซิร์ฟเล็ตเอนจิ้น (Servlet Engine) หรือเซิร์ฟเล็ตคอนเทนเนอร์ (Servlet Container)



รูปที่ 2-4 การทำงานของจาวาเซิร์ฟเล็ต

2.9 จาวาบีน (Java Bean)

จาวาบีนเป็นรูปแบบการใช้งานออปเจ็กต์ที่ถูกสร้างไว้แล้ว และนำมาประกอบกันเป็นซอฟต์แวร์ ถ้าเทียบไปแล้วจะคล้ายกับตัวแอคทีฟคอนโทรล (Active Control) ที่ใช้ในโปรแกรมภาษาวิชวลเบสิก (Visual Basic) แต่จาวาบีนจะสร้างด้วยภาษาจาวาเพื่อให้ใช้งานได้ทุกแพลตฟอร์ม ทำให้จาวาบีนมีการสร้างเพื่อขายเป็นการค้าจากผู้ผลิตรายอื่นๆ (Third Party) ด้วย จาวาบีนจะมีลักษณะดังนี้คือ

- เป็นคลาสในภาษาจาวาเพื่อเรียกใช้ได้ในโปรแกรมพัฒนาจำพวกวิชวล เช่น บีนเกี่ยวกับปุ่ม บีนเกี่ยวกับรูปภาพ บีนเกี่ยวกับเมนู เป็นต้น ถ้าเปรียบเทียบกับวิชวลเบสิกก็คือสิ่งที่เรียกว่าคอนโทรลนั่นเอง
- เป็นส่วนประกอบทางกราฟฟิกที่สามารถนำมาประกอบกันเป็นโปรแกรมที่ติดต่อกับผู้ใช้แบบ GUI ได้ง่ายขึ้นสะดวกขึ้น เหมือนที่ใช้ในโปรแกรมภาษาวิชวลต่างๆ
- เป็นลักษณะคล้ายๆ กล่องที่เราสามารถเปลี่ยนลักษณะการทำงานและคุณสมบัติต่างๆ ข้างในได้โดยเซตที่ตัวพร็อพเพอตี้ของมัน

บทที่ 3
จาวาเซิร์ฟเวอร์เพจ
(JavaServer Pages)

จาวาเซิร์ฟเวอร์เพจ (JavaServer Pages : JSP) เป็นเทคโนโลยีที่เป็นผลรวมของการนำเอาหลักการสร้างเว็บแบบสแตติกด้วย HTML มารวมกับการสร้างเว็บแบบไดนามิกด้วยภาษาจาวา ก่อให้เกิดการแสดงผลเว็บที่มีการเปลี่ยนแปลงข้อมูลได้ตามการใช้งานของผู้ใช้ JSP เป็นเทคโนโลยีคล้ายกับแอคทีฟเซิร์ฟเวอร์เพจ (Active Server Page : ASP) ของบริษัทไมโครซอฟต์ แต่ผิดกันตรงที่หัวใจของ JSP คือภาษาจาวา ซึ่งเป็นภาษาที่มีหัวใจหลักอยู่ที่ออบเจ็กต์ ซึ่งช่วยให้ง่ายต่อการพัฒนาในโปรเจ็กต์ใหญ่ๆ ตลอดจนสามารถนำเอาส่วนประกอบต่างๆกลับมาใช้ได้อีก จุดเด่นที่สำคัญของ JSP คือสามารถทำงานได้โดยไม่ขึ้นกับแพลตฟอร์มและข้อจำกัดของบริษัทซอฟต์แวร์บริษัทใดบริษัทหนึ่งโดยเฉพาะ

3.1 เปรียบเทียบ JSP กับเทคโนโลยีอื่นๆ

JSP ไม่ได้เป็นเทคโนโลยีเดียวในปัจจุบันที่ทำให้เว็บแสดงข้อมูลแบบไดนามิกได้ โดยมีเทคโนโลยีอื่นๆดังนี้

3.1.1 แอคทีฟเซิร์ฟเวอร์เพจ

แอคทีฟเซิร์ฟเวอร์เพจเป็นเทคโนโลยีที่ใช้หลักการคล้ายกับ JSP แต่เป็นเทคโนโลยีที่พัฒนาโดยบริษัทไมโครซอฟต์ โดยจะมีรูปแบบที่แตกต่างกันอย่างเด่นชัด 2 ประการคือ

ประการแรกภาษาที่ใช้ใน JSP คือภาษาจาวา ส่วนภาษาที่ใช้ใน ASP คือภาษา Visual Basic หรือภาษาอื่นๆ ที่กำหนดโดยไมโครซอฟต์

ประการที่สองเป็นผลสืบเนื่องมาจากประการแรก คือ เนื่องจากใช้ภาษาจาวาทำให้ JSP สามารถนำไปใช้งานบนระบบปฏิบัติการใดก็ได้ แต่ ASP ต้องใช้ในระบบปฏิบัติการที่ไมโครซอฟต์กำหนดเท่านั้น

3.1.2 จาวาเซิร์ฟเล็ต

ตัว JSP สามารถทำงานได้เช่นเดียวกันที่เซิร์ฟเล็ตทำได้ แต่จุดเด่นที่สำคัญของ JSP คือ สะดวกในการสร้างและเปลี่ยนแปลงมากกว่า เพราะสามารถแก้ไขที่ไฟล์ได้โดยตรง และไม่จำเป็นต้องคอมไพล์ใหม่ก่อนนำไปใช้งาน

3.1.3 จาวาสคริปต์

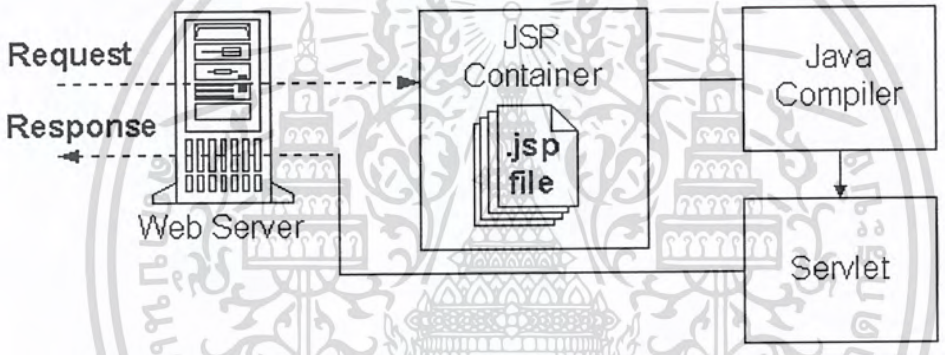
เป็นการทำเนื้อหาไดนามิกให้กับเว็บเหมือนกัน แต่จาวาสคริปต์จะเกิดการประมวลผลที่ไคลเอ็นต์ ถ้าเว็บบราวเซอร์ไม่สนับสนุนจาวาสคริปต์ก็จะทำให้ไม่สามารถทำงานได้ แต่ JSP จะเกิดการประมวลผลที่ตัวเซิร์ฟเวอร์โดยตรงทำให้ไม่มีปัญหากับตัวบราวเซอร์

3.2 JSP คอนเทนเนอร์ (JSP Container)

ไฟล์ JSP ที่เขียนขึ้นมาจะถูกรันโดย JSP คอนเทนเนอร์ ซึ่งมักจะเป็นส่วนประกอบที่อยู่ในเว็บเซิร์ฟเวอร์หรือเป็นตัวปลั๊กอินในแอพพลิเคชันเซิร์ฟเวอร์ (Application Server) โดยทั่วไป JSP คอนเทนเนอร์ จะเป็นตัวรับคำร้องจากไคลเอ็นต์ส่งผ่านไปยังตัว JSP และส่งค่าที่ได้จากการประมวลผลโดยตัว JSP กลับไปยังไคลเอ็นต์

ถ้าดูเผินๆจากทางเว็บบราวเซอร์การเรียกไฟล์ JSP ก็คล้ายกับการเรียกไฟล์ HTML ธรรมดาๆ แต่ความจริงแล้ว หลังจากคำร้องจากไคลเอ็นต์ส่งมาถึงเว็บเซิร์ฟเวอร์ JSP คอนเทนเนอร์จะทำการแปลงไฟล์ JSP ให้กลายเป็นเซิร์ฟเล็ต (ในกรณีที่ไฟล์ JSP นั้นถูกเรียกใช้เป็นครั้งแรก) ซึ่งไฟล์เซิร์ฟเล็ตที่ได้จะถูกคอมไพล์เป็น .class เพื่อใช้ในการประมวลผลคำร้องของไคลเอ็นต์แล้วส่งกลับไปให้เว็บเซิร์ฟเวอร์ในรูปของ OutputStream และทำการส่งต่อไปให้กับตัวไคลเอ็นต์ต่อไป

หลังจากนั้น หากไฟล์ JSP นั้นถูกเรียกใช้อีก JSP คอนเทนเนอร์ก็จะดึงไฟล์ .class ที่ถูกคอมไพล์ไว้แล้วไปประมวลผล โดยไม่ต้องทำการแปลงไฟล์ JSP อีก



รูปที่ 3-1 การทำงานของ JSP

จากข้อความข้างต้น เราสามารถแบ่งระยะเวลาของไฟล์ JSP หนึ่งๆ ออกเป็นสองช่วงคือ

1. Translation Time คือ ช่วงเวลาที่ไฟล์ JSP ถูกแปลงเป็นเซิร์ฟเล็ต และถูกคอมไพล์เป็น .class ซึ่งจะเกิดขึ้นก่อนการรับคำร้องจากไคลเอ็นต์ครั้งแรก
2. Client Request Time คือ ช่วงเวลาที่ไฟล์ .class ที่ได้จากการคอมไพล์ทำการรับคำร้องจากแต่ละไคลเอ็นต์แล้วทำการประมวลผล

3.3 JSP Standard Syntax and Semantic

3.3.1 Directive

JSP Directive เป็นส่วนที่เรียกว่าแมสเซจที่ใช้ส่งไปถึง JSP คอนเทนเนอร์ ในลักษณะของ

<%@ … %>

โดยแมสเซจนี้จะเป็นตัวบอกว่า คอนเทนเนอร์ ควรจะทำอะไรกับ JSP page ต่อไป ซึ่ง Directive ในส่วนนี้จะไม่มีผลต่อหน้าตาที่ออกมาของ JSP page เมื่อถูกแปลงออกมาในรูปของ HTML แล้ว Directive แบ่งออกเป็น 3 ส่วน ส่วนที่ใช้ทั่วๆไปคือ page และ include และ taglib

3.3.1.1 Page Directive

เป็นส่วนที่ปกติเราจะเห็นอยู่บนสุดของ JSP page ซึ่งใช้ในการกำหนดค่าต่างๆ ที่เกี่ยวข้องกับ JSP page นั้นหรือเกี่ยวข้องกับการติดต่อสื่อสารกับ JSP คอนเทนเนอร์ ยกตัวอย่างเช่น ถ้าเราอยากเรียกใช้คลาสที่ชื่อ java.util.Date ก็สามารถที่จะ import คลาสนี้ได้โดยใช้

<%@ page import="java.util.Date" %>

หรือเวลาที่เกิดข้อผิดพลาดใน JSP page ของเรา แล้วเราอยากโชว์ข้อผิดพลาดที่เกิดขึ้นที่ error page ที่ชื่อ errorPage.jsp เราก็ใช้

<%@ page errorPage="errorPage.jsp" %>

หรือแม้กระทั่งเวลาที่เราอยากเก็บสถานะของผู้ใช้ในรูปของ session เราก็สามารถใช้

<%@ page session="true" %>

เพื่อบอก คอนเทนเนอร์ ว่า JSP page นี้สามารถเรียกใช้ HttpSession ในการเปลี่ยนแปลงและเก็บสถานะของผู้ใช้ได้เป็นต้น

3.3.1.2 Include Directive

เป็นส่วนที่ช่วยให้เราสามารถนำไฟล์ JSP อื่นๆ มาเป็นส่วนประกอบของไฟล์ JSP ปัจจุบัน โดยใช้

<%@ include file="otherFile.jsp" %>

ประโยชน์ที่เห็นได้ชัดอย่างหนึ่งของการใช้ include Directive คือการง่ายต่อการเปลี่ยนแปลงและบำรุงรักษา ยกตัวอย่างเช่น สมมุติว่าทุกหน้าของเราต้องการ header หรือ footer ที่เหมือนกัน เราก็แค่ใช้

<%@ include file="header.jsp" %>

ใส่ลงไปในทุกๆ JSP ไฟล์ หลังจากนั้น ทุกครั้งที่เราต้องการเปลี่ยนแปลง header นี้ เราก็แค่เปลี่ยนไฟล์ที่ชื่อ header.jsp ไฟล์อื่นที่ include ไฟล์นี้เข้าไป ก็จะเปลี่ยนแปลงโดยอัตโนมัติ

3.3.2 Declaration

Declaration ใช้ในการประกาศค่าตัวแปร หรือสร้างฟังก์ชันต่างๆเพื่อจะใช้ใน JSP page นั้น Declaration เป็นส่วนที่ถูกประมวลผลในช่วง Translation Time

โดยทั่วไป Declaration จะถูก initialize เมื่อ JSP page ถูก initialize ซึ่งจะทำให้ตัวแปรหรือฟังก์ชันใน Declaration พร้อมใช้งานได้ทันที ยกตัวอย่างเช่น

<%! int i=0; %>

<%! public String s(int i) { if(i<3) return "..."; }%>

3.3.3 Expression

โดยการใช้ Expression สิ่งต่างๆที่อยู่ใน <%= ... %> จะถูกประมวลผลแล้วเปลี่ยนให้อยู่ในรูปของ String และส่วนที่ได้นี้จะถูกรวมเข้าไปอยู่ใน output HTML โดยตรง Expression เป็นส่วนที่ถูกประมวลผลในช่วง Translation Time เช่น

<%= i %>

JSP คอนเทนเนอร์ ก็จะดึงค่าตัวแปร i ออกมาแล้วเปลี่ยนเป็นสตริงโดยใช้ Integer.toString(i)

<%= "Hello" %>

JSP คอนเทนเนอร์ จะนำค่า Hello ใส่เข้าไปใน output HTML

ข้อควรจำอย่างหนึ่งก็คือ ห้ามทำการใส่เซมิโคลอน (;) ลงไปใน <%= ... %> ยกเว้นถ้าเซมิโคลอนนั้นเป็นส่วนหนึ่งของ String เช่น

<%= "Semicolon ;" %>

3.3.4 Code Fragment/Scriplets

เราสามารถใส่โค้ด หรือส่วนหนึ่งของโค้ดข้าไปยัง JSP page ได้โดยใช้ <% ... %> โค้ดที่จะใส่เข้าไปนี้จะไปอยู่ในส่วนของ service() ฟังก์ชันของเซิร์ฟเล็ต ซึ่งจะถูกเรียกใช้เมื่อมีคำร้องจากไคลเอ็นต์ ยกตัวอย่างเช่น

<% int userID=request.getParameter("userID"); %>

3.3.5 Comment

Comment ใน JSP มีอยู่ 2 แบบคือ

1. Comment ซึ่งปกติถูกใช้ใน HTML ไฟล์นั้นจะไม่ถูกละเว้นโดย JSP คอนเทนเนอร์ เมื่อไฟล์ JSP ถูกแปลงเป็น .class แล้ว จะยังคงปรากฏอยู่ที่ output HTML ตัวอย่างเช่น

<!-- This is my comment -->

บางทีเราอาจจะแปลง comment ที่อยู่ข่างใน <!-- ... --> ได้โดยใช้ expression เช่น

<!-- <%= expression %> more comment -->

2. comment ที่ผู้เขียน JSP ใช้สำหรับ comment JSP ไฟล์ที่เขียนอยู่ ซึ่งจะถูกละเว้นโดย JSP คอนเทนเนอร์ ในช่วงของ Translation Time ไม่ไปปรากฏที่ output HTML ตัวอย่างเช่น

<%-- comment for server side only --%>

3.3.6 JSP Implicit Objects

เมื่อไรก็ตามที่เราต้องการเรียกใช้คลาสบางคลาส เราจะต้องทำการ import โดยใช้ page Direction แล้วทำการเรียกใน Scriptlet ยกตัวอย่างเช่น

<%@ page import="java.util.Date" %>

<% Date = new Date() %>

บางครั้งในการเขียน JSP จะมีบางคลาสที่เราใช้บ่อย เช่น HttpSession หรือคลาสบางคลาสที่เราไม่สามารถเรียกใช้ได้โดยตรงจากการ import เช่น HttpServletRequest หรือ HttpServletResponse ดังนั้น JSP จึงช่วยอำนวยความสะดวกโดยให้เราสามารถเรียกออบเจกต์ของคลาสดังกล่าวออกมาใช้ได้เลย โดยไม่ต้องมีการประกาศตัวแปรก่อน (ใช้ได้ใน Scriptlet และ Expression) คลาสพวกนี้เราเรียกว่าคลาส Predefined Variable หรือ Implicit Object โดยมีดังต่อไปนี้

- **request** คือออบเจกต์ที่เป็นอินสแทนซ์ของ ServletRequest ซึ่งถ้าใช้ใน HTTP ก็จะเป็น HttpServletRequest เราสามารถใช้ request ในการอ่านค่าพารามิเตอร์ต่างๆ ที่ส่งมาโดยเมธอด get หรือ post จากไคลเอ็นต์ โดยใช้ request.getParameter() หรือใช้ฟังก์ชันอื่นๆที่มีอยู่ใน ServletRequest
- **response** คือออบเจกต์ที่เป็นอินสแทนซ์ของ ServletResponse ที่ใช้ในการส่งค่าต่างๆ กลับไปที่ไคลเอ็นต์โดยผ่านทาง OutputStream ซึ่งอยู่ในรูปของ PrintWriter หรือ JSPwriter
- **pageContext** คือออบเจกต์ที่เก็บฟังก์ชันที่ใช้ในการเรียกคุณสมบัติต่างๆ ของ JSP คอนเทนเนอร์ ซึ่งจะไม่เหมือนกันในแต่ละตัวมาเชื่อมกับไฟล์ JSP เพื่อเพิ่มประสิทธิภาพในการทำงาน
- **session** คือออบเจ็กต์ที่เป็นอินสแทนซ์ของ HttpSession ของไคลเอ็นต์ที่ส่งคำร้องมา ซึ่งจริงๆแล้วเราสามารถเรียกออบเจ็กนี้มาโดยผ่านทาง request.getSession() ก็ได้
- **application** คือ ServletContext ทีเก็บค่าต่างๆของ environment ที่ไฟล์ JSP นี้อยู่
- **out** คือ PrintWriter หรือ JSPwriter ที่ใช้ในการพิมพ์ text output ออกไปยังไคลเอ็นต์โดยผ่านทาง response อีกทีหนึ่ง
- **config** คือ ServletConfig ที่เก็บค่า setting ต่างๆสำหรับไฟล์ JSP
- **page** คือ this ที่ใช้แทนตัวของไฟล์ JSP เอง
- **exception** คือ Throwable interface ทีใช้ส่งค่า error ต่างๆ ไปยัง error page

สำหรับรายละเอียดในการใช้ Implicit Object แต่ละตัวก็สามารถหาได้จากเอกสารของ API นั้นๆ

3.3.7 JSP Action

JSP action สามารถใช้ในการควบคุมและเปลี่ยนแปลงการทำงานของไฟล์ JSP ที่เราเขียน ซึ่งอาจจะเป็นในลักษณะของ

1. Insert File ต่างๆเข้ามาเป็นส่วนหนึ่งของไฟล์ JSP ปัจจุบัน เช่น

<jsp:include page="otherfile.jsp"/>

2. ใช้สำหรับควบคุมการทำงานของจาวาบีนที่เป็นส่วนประกอบหนึ่งของไฟล์ JSP โดยใช้

<jsp:useBean ... />, <jsp:setProperty ... />, <jsp:getProperty ... />

3. ใช้ในการส่งต่อคำร้องของไคลเอ็นต์ไปยังไฟล์ JSP อื่นๆ โดยใช้

<jsp:forward ... />

4. ใช้เพื่อทำการสร้างออบเจกต์หรือ Embed code ที่ใช้สำหรับเรียก Java Plugin เพื่อทำการรันจาวาแอพเพล็ตหรือจาวาบีนในตัวเว็บบราวเซอร์ที่กำลังเปิดไฟล์ JSP ดังกล่าวอยู่ โดยใช้

<jsp:plugin … />

บทที่ 4
การวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุ
(Object Oriented Analysis & Design)

ก่อนที่จะลงมือสร้างผลงานทางวิศวกรรมใด วิศวกรจำเป็นต้องผ่านกระบวนการที่สำคัญอย่างหนึ่งก่อนคือ ขั้นตอนการวิเคราะห์และออกแบบ ในระบบซอฟต์แวร์ก็เช่นเดียวกัน ผู้ที่จะพัฒนาระบบซอฟต์แวร์ที่ต้องการใช้ภาษาในการเขียนโปรแกรมเชิงวัตถุ (OOP) ก็ควรจะต้องมีการวิเคราะห์และออกแบบระบบซอฟต์แวร์ด้วยกระบวนการพัฒนาเชิงวัตถุเช่นกัน

การวิเคราะห์ระบบด้วยวิธีเชิงวัตถุนั้นจะเน้นการโมเดลระบบในโลกความเป็นจริงโดยใช้มุมของการพัฒนาระบบด้วยวิธีเชิงวัตถุ การวิเคราะห์ด้วยวิธีเชิงวัตถุเป็นวิธีวิเคราะห์และตรวจสอบความต้องการโดยใช้มุมมองจากคลาสและวัตถุจากคำศัพท์ที่อยู่ในโดเมนของปัญหา ส่วนการออกแบบระบบด้วยวิธีเชิงวัตถุเป็นวิธีในการออกแบบกระบวนการในการแยกย่อย และเป็นการใช้สัญลักษณ์แบบการพัฒนาระบบด้วยวิธีเชิงวัตถุสำหรับโมเดลระบบทั้งในทางตรรก (Logic) และทางกายภาพ (Physical) รวมถึงโมเดลการทำงานแบบสแตติก (Static) และไดนามิก (Dynamic) สำหรับระบบที่ต้องการจะออกแบบ ด้วยการแยกย่อยปัญหาด้วยวิธีการเชิงวัตถุ ทำให้การออกแบบระบบด้วยวิธีเชิงวัตถุต่างจากการออกแบบระบบแบบโครงสร้าง เนื่องจากในการออกแบบระบบด้วยวิธีเชิงวัตถุนั้น ผู้ออกแบบระบบจะมีการแยกแยะเอกลักษณ์โดยยึดหลักวัตถุเป็นหลัก ส่วนการออกแบบระบบแบบโครงสร้างจะมีการแยกแยะเอกลักษณ์โดยดูจากอัลกอริทึมการทำงานเป็นหลัก

โดยทั้งภาษาการเขียนโปรแกรมเชิงวัตถุ, การการวิเคราะห์ระบบด้วยวิธีเชิงวัตถุและออกแบบระบบด้วยวิธีเชิงวัตถุนั้นจะเกี่ยวข้องกันดังนี้คือ หลังจากที่ได้วิเคราะห์ระบบแล้วก็จะได้โมเดลของระบบเชิงวัตถุ ซึ่งจะใช้เป็นหลักในการออกแบบ และเมื่อออกแบบเสร็จก็จะได้โครงสร้างหลักหรือพิมพ์เขียว (Blueprint) สำหรับการเขียนโปรแกรมด้วยภาษาในการเขียนโปรแกรมเชิงวัตถุต่อไป

4.1 วัตถุ (Object)

การพัฒนาระบบแบบโครงสร้างนั้น จะมองระบบเป็นกลุ่มของข้อมูลและฟังก์ชันการทำงาน โดยแบ่งฟังก์ชันออกเป็นฟังก์ชันย่อยๆ ที่มีการทำงานง่ายๆ ส่วนการพัฒนาระบบด้วยมุมมองของการพัฒนาระบบด้วยวิธีเชิงวัตถุนั้นจะต่างออกไปคือ จะแบ่งแยกระบบออกเป็นวัตถุ โดยวัตถุในที่นี้คือ กลุ่มก้อนของเอนติตี (Entity) ที่ประกอบไปด้วยคุณสมบัติหรือแอตทริบิวต์ (Attribute), พฤติกรรมหรือเมธอด (Behavior or Method) และอาจจะมีสถานะ (State) ด้วย

แต่ถ้ากล่าวให้ครบถ้วน วัตถุเป็นสิ่งที่มีทั้งข้อมูลและพฤติกรรม วัตถุเป็นสิ่งที่ใช้แทนสิ่งที่มีอยู่ในโลกจริง เช่น สุนัข ระบบการไหลของอากาศ เครื่องยนต์ หรืออาจจะแทนสิ่งที่มีตัวตนอยู่ในความคิด เช่น บัญชีธนาคาร รายชื่อนักศึกษา หรือเป็นสิ่งที่มองเห็นได้ เช่น ตัวอักษร ดินสอ ปากกา หนังสือ เป็นต้น โดยวัตถุทั้งหมดนั้นจะมีสิ่งที่ผู้พัฒนาสนใจซึ่งประกอบด้วย

- แอตทริบิวต์ (ข้อมูล)

- เมธอด (การกระทำ)
- สถานะ (หน่วยความจำ)
- สิ่งบ่งชี้ (Identify)
- ความรับผิดชอบ (Responsibility)

แนวคิดหลักของวัตถุคือ การรวบรวมคุณสมบัติที่เกี่ยวข้องไว้ในที่เดียวกัน โดยการพัฒนาระบบแบบโครงสร้างจะออกแบบให้ซอฟต์แวร์มีข้อมูลและฟังก์ชันที่แยกจากกันเช่น ในแผนภาพ Data Flow จะแสดงทั้งการของข้อมูลและกระบวนการที่กระทำกับข้อมูล โดยข้อมูลนั้นสามารถที่จะแบ่งย่อยลงไปได้อีกถ้าจำเป็น และสำหรับผังงานโครงสร้าง (Structure Chart) ก็จะแสดงโครงสร้างต้นไม้ในการแบ่งย่อยฟังก์ชันซึ่งไม่ได้เกี่ยวข้องกับแผนภาพ Data Flow เลย แต่สำหรับวัตถุนั้นจะรวมข้อมูลและฟังก์ชันเข้าไว้ด้วยกัน

วัตถุนั้นเป็นเครื่องมือในการมองสิ่งต่างๆ ที่มีประสิทธิภาพ เพราะว่าในโลกความเป็นจริงต้องติดต่อกับวัตถุอยู่ตลอดเวลา และวัตถุแต่ละวัตถุจะมีคุณสมบัติที่อาจจะสนใจหรือไม่สนใจก็ได้ มีเมธอดหรือไม่มีก็ได้ แต่ส่วนใหญ่วัตถุจะมีเมธอดและจำเป็นต้องมีการดูแล การโมเดลระบบเชิงวัตถุทำให้สามารถมองระบบต่างๆ ที่ติดต่อด้วยได้อย่างเข้าใจ ซึ่งการมองระบบแบบโครงสร้างนั้นไม่สามารถทำได้

โดยทั่วไปวัตถุอาจจะเป็นสิ่งที่สามารถทำงานได้ด้วยตัวเอง ถ้าวัตถุสามารถทำงานได้ด้วยตัวเองจะส่งผลให้วัตถุที่แยกกันมีความผูกพันกันน้อยลง เมธอดของวัตถุจะมีความเป็นพื้นฐานมากกว่าระบบที่พัฒนาแบบ Functional มาตรฐาน ด้วยวิธีการพัฒนาระบบแบบ Functional ผู้ที่เรียกฟังก์ชันจะต้องคอยจนกว่าฟังก์ชันที่ถูกเรียกจะทำงานเสร็จ ผู้เรียกฟังก์ชันจึงจะกลับไปทำงานต่อได้ ซึ่งการเรียกฟังก์ชันแบบนี้เป็นเพียงการเรียกฟังก์ชันแบบหนึ่งในโมเดลของวัตถุเท่านั้น ในโมเดลของวัตถุ ตัววัตถุสามารถเรียกฟังก์ชันแบบซิงโครนัสหรือแบบอะซิงโครนัสก็ได้ ถ้าวัตถุที่เรียกอยู่คนละเธรดหรือบนคนละโพรเซสเซอร์กัน

4.1.1 แอตทริบิวต์ (Attribute)

วัตถุจำเป็นจะต้องมีการเก็บข้อมูลของตัวเอง โดยข้อมูลที่วัตถุเก็บไว้นั้นอาจจะเก็บไว้เฉยๆ เก็บไว้เพื่อที่จะให้ผู้อื่นเรียกไปใช้งาน เก็บไว้ใช้ในการประมวลผลเพื่อตอบสนองเหตุการณ์ภายนอก หรืออาจจะเก็บไว้เพื่อเป็นสถานะของตัววัตถุเอง สรุปว่าวัตถุส่วนใหญ่จะต้องเก็บข้อมูล เพื่อให้วัตถุนั้นสามารถทำหน้าที่รับผิดชอบได้อย่างถูกต้อง

โดยทั่วไปแอตทริบิวต์ของวัตถุจะถูกซ่อนไว้จากผู้ที่จะใช้งานวัตถุ ทั้งนี้เนื่องจากว่าการเข้าไปใช้งานแอตทริบิวต์โดยตรงเปรียบเสมือนเป็นการเข้าถึงโครงสร้างของวัตถุ ดังนั้นการออกแบบวัตถุที่ดีจึงควรซ่อนแอตทริบิวต์ไว้เพื่อไม่ให้ผู้ใช้เห็น

บางครั้งการออกแบบวัตถุที่ไม่ดีก็อาจจะทำให้วัตถุนั้นมีแต่แอตทริบิวต์โดยไม่มีเมธอดได้ ซึ่งในกรณีนี้นักออกแบบจะต้องย้อนกลับไปดูว่าวัตถุนั้นควรจะมีเมธอดอะไร เพราะวัตถุที่มีแต่แอตทริบิวต์นั้นก็ไม่ต่างจากโครงสร้างข้อมูลเลย

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

4.1.2 เมธอดหรือพฤติกรรม

วัตถุที่ทำงานแบบพาสสิฟ (Passive Object) จะมีบริการไว้ให้วัตถุอื่นเรียกใช้ ส่วนวัตถุที่ทำงานแบบแอกทีฟ (Active Object) นั้นจะเป็นวัตถุหลักของเธรดซึ่งจะทำหน้าที่เรียกใช้บริการจากวัตถุที่ทำงานแบบพาสสิฟ โดยเมธอดสามารถแบ่งออกเป็น 3 แบบคือ

- Simple Method
- Automation Method
- Continuous Method

แบบแรก Simple Method วัตถุจะให้บริการกับการร้องขอเข้ามาและไม่ค้างค่าในหน่วยความจำไว้สำหรับให้บริการครั้งต่อไป การกระทำเป็นหนึ่งเดียวและสมบูรณ์ในตัว วัตถุพื้นฐานจะมีข้อมูลชนิดพื้นฐานและโอเปอเรชันที่เกี่ยวข้องกับข้อมูลเหล่านี้ เราจะเรียกวัตถุที่มีเฉพาะเมธอดแบบนี้ว่า Primitive Object แบบที่สอง Automation Method ผู้พัฒนาจะมองวัตถุเสมือนหนึ่งเป็นเครื่องจักรชนิดหนึ่งที่เรียกว่า Automation หรือ Finite State Machine วัตถุชนิดนี้มีกลุ่มของสถานะที่มีขอบเขต วัตถุหนึ่งอาจจะมีสถานะ 1 หรือมากกว่า ณ เวลาใดๆ การที่วัตถุจะอยู่ในสถานะใดๆ นั้นก็จะขึ้นอยู่กับอีเวนต์หรือเหตุการณ์ที่เข้ามาจากสภาวะแวดล้อมภายนอก และเนื่องจากวัตถุชนิดนี้ทำงานแบบ State Machine เพื่อตอบสนองกับอีเวนต์ภายนอกจึงเรียกวัตถุชนิดนี้ว่า Reactive Object และแบบที่สาม Continuous Method วัตถุที่มีเมธอดแบบ Continuous จะมีกลุ่มของสถานะที่ไม่จำกัดและไม่มีขอบเขต การทำงานของวัตถุปัจจุบันจะขึ้นอยู่กับการทำงานของเมธอดและอินพุทก่อนหน้า ซึ่งความเกี่ยวข้องระหว่างสถานะนี้อาจจะเป็นความสัมพันธ์แบบต่อเนื่อง

4.1.3 เมสเซจ (Message)

การสื่อสารระหว่างวัตถุสามารถทำได้โดยการส่งเมสเซจ เมสเซจเป็นการแยกแยะเอกลักษณ์ของข้อมูล (Data) หรือข้อมูลสารสนเทศ (Information) การควบคุมที่ส่งมาจากวัตถุหนึ่งไปยังวัตถุอื่นๆ ในเชิงการเขียนโปรแกรมเมสเซจสามารถทำได้หลายแบบตัวอย่างเช่น การเรียกฟังก์ชัน อินเทอร์รัพต์ เป็นต้น

ในขั้นตอนการวิเคราะห์ระบบจะมีการกำหนดเมสเซจ ส่วนในขั้นตอนของการออกแบบนั้นจึงจะมากำหนดรูปแบบการซิงโครไนเซชันและความต้องการทางด้านเวลาของแต่ละเมสเซจ และเมื่อวัตถุได้รับเมสเซจมา วัตถุจะเปลี่ยนเมสเซจที่เข้ามาให้เป็นโอเปอเรชันการเปลี่ยนสถานะ คำสั่งหรือข้อมูลตามความเหมาะสม

การใช้เมสเซจทำให้แต่ละวัตถุมีความเกี่ยวข้องกันน้อยลง ในขั้นตอนของการวิเคราะห์ผู้พัฒนาระบบจะไม่ได้กำหนดรายละเอียดในการติดต่อของเมสเซจว่า จะมีรายละเอียดของการเรียกฟังก์ชันอย่างไร และเมื่อถึงขั้นตอนของการออกแบบจึงจะจัดการกับรายละเอียดเหล่านี้

ส่วนอินเทอร์เฟซของวัตถุเป็นส่วนที่วัตถุใช้ติดต่อกับโลกภายนอก ซึ่งอินเทอร์เฟซจะทำหน้าที่กำหนดเซ็ตของโพรโตคอลเพื่อสื่อสารกับวัตถุอื่นๆ โพรโตคอลของส่วนอินเทอร์เฟซนั้นประกอบไปด้วย 3 สิ่งดังต่อไปนี้

- Precondition

46147

- Signature
- Postcondition

Precondition เป็นเงื่อนไขที่จะต้องมีค่าเป็นจริงก่อนที่จะส่งหรือรับเมสเซจ โดยปกติ Precondition จะเป็นหน้าที่ของวัตถุที่ผู้ส่งเมสเซจจะต้องตรวจสอบ ส่วน Postcondition เป็นเงื่อนไขที่จะต้องมีค่าจริงหลังจากวัตถุได้ประมวลผลเมสเซจเรียบร้อยแล้ว และเป็นหน้าที่ของวัตถุที่รับเมสเซจ สำหรับ Signature นั้นเป็นรูปแบบในการส่งเมสเซจ

อินเทอร์เฟซเป็นส่วนที่สำคัญในการแสดงคุณลักษณะของวัตถุ และสิ่งที่อยู่ภายนอกก็จะต้องเห็นส่วนอินเทอร์เฟซของวัตถุด้วย วัตถุสามารถซ่อนรายละเอียดที่ไม่สำคัญสำหรับผู้ใช้ได้หรือเรียกว่า Encapsulation ตัวอย่าง เช่น ภาษา Java ก็มีคำสั่งเพื่อใช้ซ่อนแอตทริบิวต์และเมธอดได้

4.1.4 ความรับผิดชอบ

ความรับผิดชอบของวัตถุคือ หน้าที่และบทบาทของวัตถุต่อระบบ โดยส่วนอินเทอร์เฟซและเมธอดของวัตถุนั้นสามารถใช้เพื่อให้วัตถุทำหน้าที่รับผิดชอบได้ ความรับผิดชอบของวัตถุควรจะเป็นสิ่งแรกที่ผู้พัฒนาระบบกำหนดให้กับวัตถุ เพราะถ้าไม่รู้ว่าวัตถุนั้นมีหน้าที่อย่างไร ก็จะไม่สามารถกำหนดคุณลักษณะอื่นๆ ให้กับวัตถุได้อย่างถูกต้องและครบถ้วน

4.2 ความสัมพันธ์ระหว่างวัตถุ

โดยปกตินั้นวัตถุเพียงวัตถุเดียวไม่สามารถที่จะรับผิดชอบงานทั้งหมดของระบบได้ วัตถุจะต้องมีความสัมพันธ์และต้องทำงานร่วมกันเพื่อให้ระบบสามารถทำหน้าที่รับผิดชอบได้ โดยเราแบ่งความสัมพันธ์ระหว่างวัตถุออกเป็น 2 แบบ คือ ลิงค์ (Link) และการเป็นส่วนหนึ่งของ (Aggregation)

4.2.1 ลิงค์

ความสัมพันธ์ระหว่างวัตถุแบบลิงค์ เป็นการเชื่อมต่อระหว่างวัตถุในทางกายภาพหรือในทางตรรกะ การที่วัตถุซึ่งมาประกอบกันเป็นระบบจะทำงานร่วมกันได้นั้น วัตถุเหล่านี้จะต้องมีการสื่อสารระหว่างกัน วัตถุจะสื่อสารกันโดยวัตถุที่เรียกใช้บริการ (service) จากวัตถุอื่นเรียกว่าวัตถุไคลเอ็นต์ (Client) ส่วนวัตถุที่มีบริการให้วัตถุอื่นเรียกนั้นจะเรียกว่าวัตถุเซิร์ฟเวอร์ (Server) หรือ Supplier

4.2.2 ความสัมพันธ์แบบเป็นส่วนหนึ่งของ

ความสัมพันธ์แบบเป็นส่วนหนึ่งของ เป็นความสัมพันธ์ที่วัตถุหนึ่งเป็นส่วนหนึ่งของอีกวัตถุหนึ่ง โดยอาจจะเป็นความสัมพันธ์แบบที่เป็นส่วนหนึ่งแบบธรรมดาหรือคอมโพซิชัน (Composition) ความสัมพันธ์แบบคอมโพซิชันเป็นรูปแบบหนึ่งของความสัมพันธ์แบบเป็นส่วนหนึ่งของ โดยมีความแตกต่างกัน ถ้าเป็นความสัมพันธ์แบบคอมโพซิชันและคลาสที่ประกอบไปด้วยคลาสอื่นจะไม่สามารถแชร์ความเป็นเจ้าของได้ ส่วนคลาสที่เป็นเจ้าของก็จะมีหน้าที่สร้างและทำลายวัตถุของคลาสที่ประกอบอยู่ภายในด้วย ในการวาดโมเดลจะเขียนคลาสที่เป็นส่วนประกอบของคลาสอื่นให้อยู่ในคลาสที่เป็นคลาสเจ้าของ

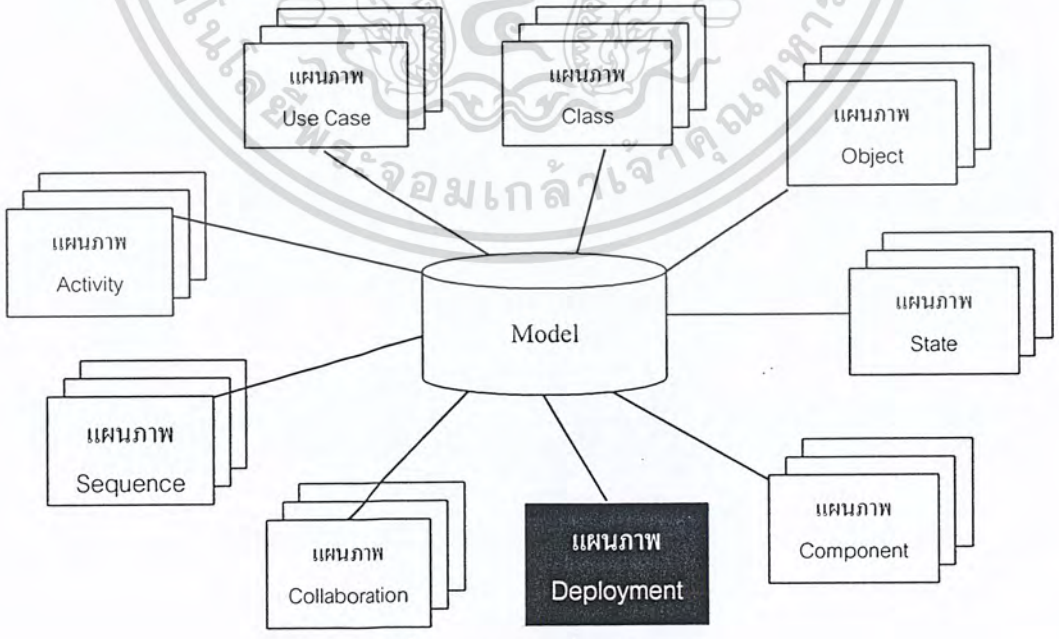
หรือสามารถแทนความสัมพันธ์ของคอมโพซิชันโดยใช้สัญลักษณ์เดียวกับความสัมพันธ์แบบเป็นส่วนหนึ่งของ คือใช้เส้นและหัวลูกศรที่เป็นสี่เหลี่ยมระบายสีทึบ ตัวอย่างการใช้ความสัมพันธ์แบบคอมโพซิชัน คือ วัตถุที่ทำงานแบบแอกทีฟ (ซึ่งเป็นวัตถุหลักของเธรด) ตัววัตถุที่ทำงานแบบแอกทีฟจะทำหน้าที่สร้างเธรดขึ้นมาเพื่อทำงาน โดยวัตถุที่เป็นส่วนประกอบก็จะทำงานอยู่ในเธรด ตัววัตถุซึ่งถูกสร้างจากคลาสที่เป็นเจ้าของจะทำหน้าที่รับเมสเซจจากภายนอก และส่งเมสเซจที่ได้รับมาต่อให้กับวัตถุที่สร้างคลาสที่เป็นส่วนประกอบ

4.3 Unified Modeling Language

Unified Modeling Language หรือ UML เป็นภาษาในการโมเดลมาตรฐาน UML เกิดจากการพัฒนาร่วมกันของผู้นำเทคโนโลยีทางด้านวัตถุ 3 คนคือ Grady Booch, Ivar Jacobson และ Jim Rumbaugh โดยก่อนที่จะมาเป็น UML นั้น ผู้นำทั้ง 3 คนนี้ต่างก็มีโมเดลสำหรับพัฒนาเชิงวัตถุเป็นของตัวเอง ต่อมาทางบริษัท Rational Software จึงได้ร่วมมือกับผู้นำทั้ง 3 คนนี้ในการพัฒนาโมเดลร่วมกัน เพื่อให้เกิดความเป็นหนึ่งเดียวสำหรับการพัฒนาระบบเชิงวัตถุ

UML นั้นได้รวมแนวความคิดของวิธีต่างๆเข้าไว้ด้วยกัน จุดประสงค์ของ UML ก็คือ ต้องการสร้างโมเดลในการพัฒนาที่เข้าใจได้และสร้างได้ง่าย แต่สามารถนำไปใช้ได้กับทุกๆระบบ ตัว UML เองนั้นได้รับการสนับสนุนจากผู้นำในอุตสาหกรรมทางด้านคอมพิวเตอร์หลายๆราย เช่น HP, ICON Computing, IBM, I-Logix, Intellicorp, MCI Systemhouse, Microsoft, ObjecTime, Oracle, Platinum Technology, Ptech, Reich Technologies, Softeam, Sterling Software, Taskon และ Unisys

ใน UML มีโมเดลที่สื่อสารด้วยภาพได้สำหรับระบบหลายๆโมเดล โดยแต่ละโมเดลก็จะแสดงมุมมองต่อระบบที่ไม่เหมือนกัน ประกอบไปด้วย

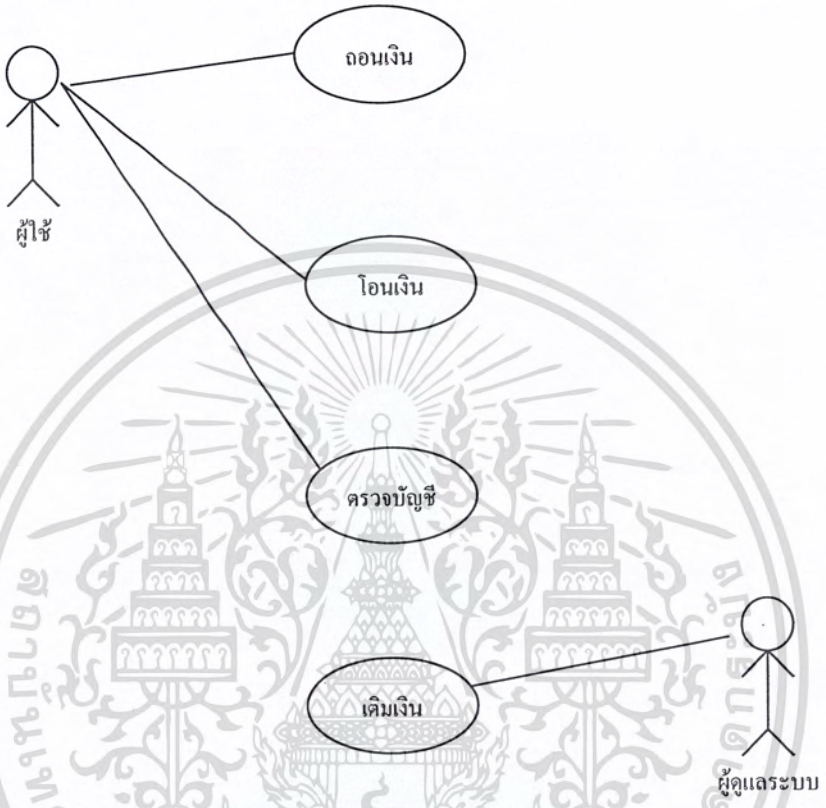


รูปที่ 4-1 แผนภาพต่างๆ ใน UML

4.4 แผนภาพของ UML

4.4.1 แผนภาพ Use Case

แผนภาพ Use Case จะแสดงกรณีทั่วไปในการติดต่อระหว่างระบบกับวัตถุภายนอก โดยแผนภาพ Use Case นั้นจะแสดงฟังก์ชันหลักๆซึ่งผู้ใช้ระดับล่างสามารถทำความเข้าใจได้ง่ายด้วย

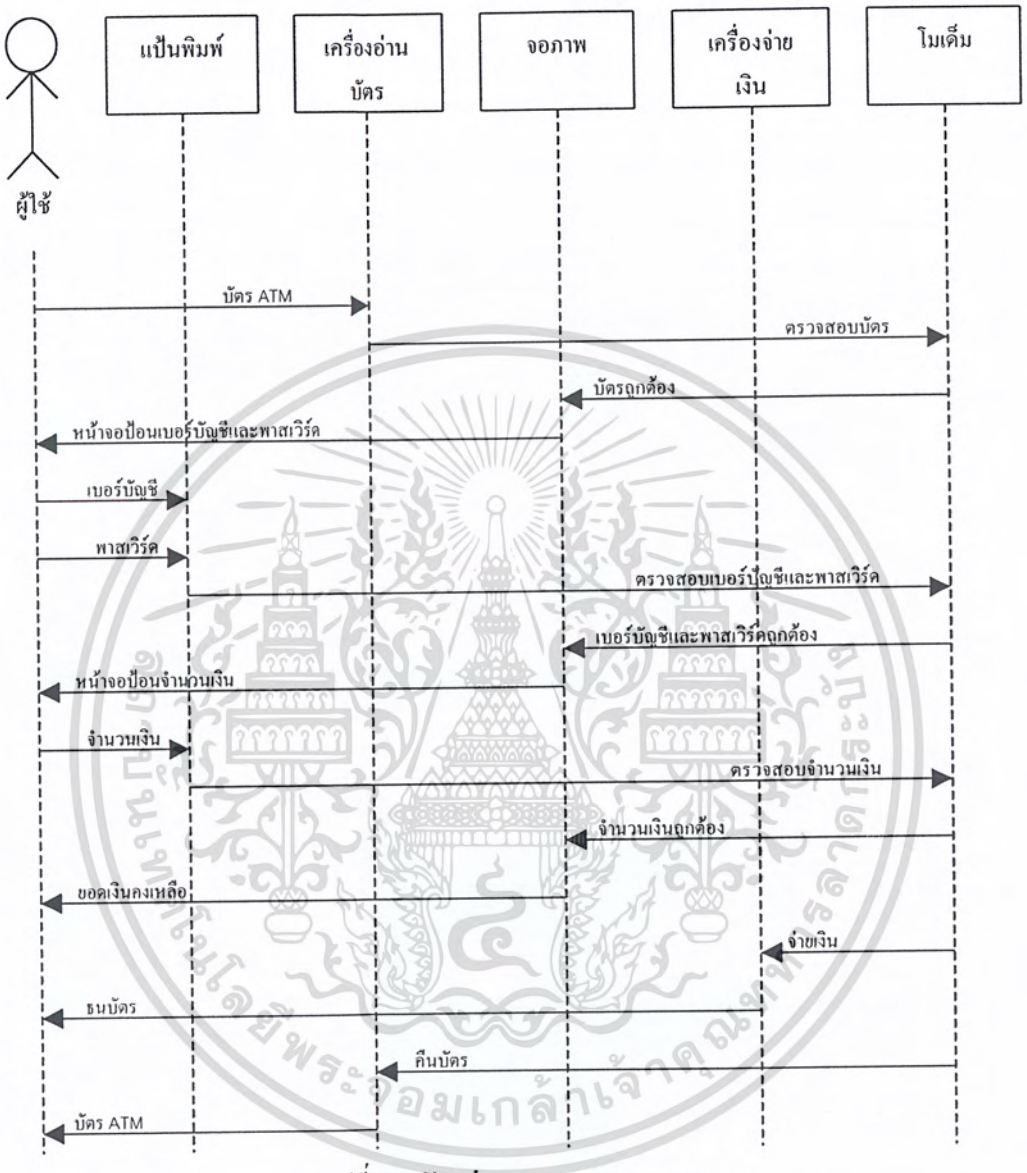


รูปที่ 4-2 ตัวอย่างแผนภาพ Use Case

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

4.4.2 แผนภาพ Sequence

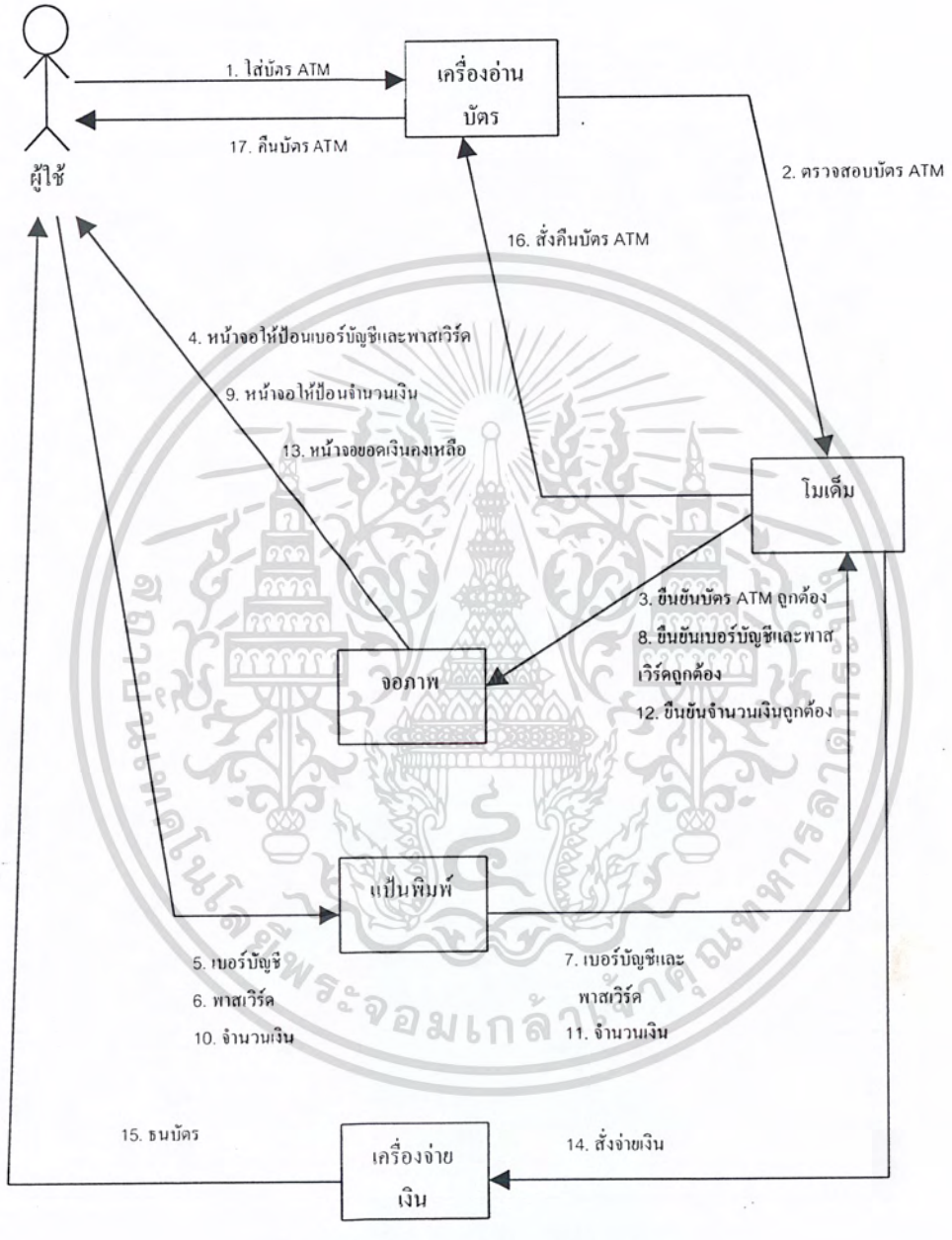
แผนภาพ Sequence จะแสดงลำดับของเมสเซจระหว่างวัตถุ โดยเส้นในแนวตั้งแสดงวัตถุโดยมีชื่อของวัตถุที่อาจจะเขียนอยู่ด้านบนหรือล่างของเส้น ส่วนเส้นในแนวนอนคือเมสเซจ



รูปที่ 4-3 ตัวอย่างแผนภาพ Sequence

4.4.3 แผนภาพ Collaboration

แผนภาพ Collaboration นั้นจะแสดงข้อมูลพื้นฐานเหมือนกับแผนภาพ Sequence แต่จะแตกต่างกันตรงที่ตัวแผนภาพ Sequence นั้นจะเน้นที่ลำดับของเมสเซจ แต่แผนภาพ Collaboration จะเน้นที่โครงสร้างของวัตถุที่ประกอบกันอยู่ในระบบ



รูปที่ 4-4 ตัวอย่างแผนภาพ Collaboration

4.4.4 แผนภาพ Class

แผนภาพของ Class เป็นแผนภาพที่มีความสำคัญมากที่สุดในการวิเคราะห์และออกแบบโดยใช้วิธีเชิงวัตถุ เนื่องจากแผนภาพของคลาสจะแสดงโครงสร้างของวัตถุและคลาสที่มีในระบบรวมทั้งแสดงความสัมพันธ์ด้วย แผนภาพของคลาสจะเป็นโครงสร้างหลักของระบบและยังใช้ในการแยกย่อยรายละเอียด (Decomposite) ด้วยวิธีเชิงวัตถุ

ในระบบที่พัฒนาด้วยวิธีเชิงวัตถุนั้นมีความสัมพันธ์ระหว่างวัตถุอยู่ดังนี้คือ

- **แอซโซซิเอชัน (Association)**

  ความสัมพันธ์แบบแอซโซซิเอชันนั้นเป็นความสัมพันธ์ระหว่างคลาสแบบสองทิศทาง ความสัมพันธ์แบบแอซโซซิเอชันนั้นจะเกิดขึ้นเมื่อวัตถุของคลาสสองวัตถุมีความเกี่ยวข้องกันหรือมีการสื่อสารกัน

รูปที่ 4-5 สัญลักษณ์แสดงความสัมพันธ์แบบแอซโซซิเอชัน

- **ความเป็นส่วนหนึ่งของ (Aggregation)**

  ความสัมพันธ์แบบเป็นส่วนหนึ่งของนั้นเป็นความสัมพันธ์แบบแอซโซซิเอชันชนิดพิเศษ โดยจะหมายถึงว่าคลาสหนึ่งประกอบด้วยคลาสอีกคลาสหนึ่งทั้งในทางตรรก เช่น คลาสทะเบียนนักศึกษาจะประกอบไปด้วยคลาสของวิชานักศึกษาลงทะเบียน และในทางกายภาพ เช่น คลาสระบบตู้เอทีเอ็มประกอบไปด้วยคลาสเครื่องจ่ายเงินเป็นต้น



รูปที่ 4-6 สัญลักษณ์แสดงความสัมพันธ์แบบเป็นส่วนหนึ่งของ

- **คอมโพซิชัน (Composition)**

  ความสัมพันธ์แบบคอมโพซิชันนั้นเป็นชนิดพิเศษของความสัมพันธ์แบบเป็นส่วนหนึ่งของคือนอกจากวัตถุที่เป็นเจ้าของนั้นจะประกอบด้วยวัตถุส่วนประกอบที่อยู่ภายในแล้ววัตถุที่เป็นเจ้าของยังมีหน้าที่รับผิดชอบวัตถุส่วนประกอบที่ตัวเองเป็นเจ้าของอยู่ด้วย เช่น วัตถุเจ้าของนั้นมีหน้าที่ในการสร้างและทำลายวัตถุส่วนประกอบ



รูปที่ 4-7 สัญลักษณ์แสดงความสัมพันธ์แบบคอมโพซิชัน

- **การสืบทอดคุณสมบัติ (Generalization)**

  ความสัมพันธ์แบบบการสืบทอดคุณสมบัติเป็นความสัมพันธ์ระหว่างคลาสที่เกิดขึ้นเมื่อคลาสหนึ่งมีการสืบทอดคุณสมบัติมาจากอีกคลาสหนึ่ง เราสามารถใช้ความสัมพันธ์แบบการสืบทอดคุณสมบัตินี้ในการจัดลำดับขั้นของคลาสได้ โดยคลาสที่อยู่ด้านบนของลำดับชั้นนั้น

เราจะเรียกว่าซูเปอร์คลาสซึ่งเป็นคลาสที่สืบทอดคุณสมบัติและพฤติกรรมกับคลาสที่อยู่ระดับล่างของลำดับชั้นซึ่งเราเรียกว่าสับคลาส



รูปที่ 4-8 สัญลักษณ์แสดงความสัมพันธ์แบบการสืบทอดคุณสมบัติ

4.4.5 แผนภาพ Object

แผนภาพ Object นั้นจะแสดงกลุ่มของวัตถุและความสัมพันธ์ระหว่างวัตถุต่างๆ ณ ช่วงเวลาใดเวลาหนึ่ง

4.4.6 แผนภาพ State

แผนภาพสถานะของ UML นั้นได้รับรูปแบบมาจากแผนภาพสถานะของ Harel ตัวแผนภาพสถานะนั้นมีข้อดีเหนือ FSM แต่ก็ยังใช้ทฤษฎีพื้นฐานเหมือนกับ FSM ตัวแผนภาพสถานะนั้นได้เพิ่มความสามารถของการซ้อนสถานะและการทำงานแบบพร้อมๆ กันและยังเพิ่มความสามารถในการโมเดลแอกชันเข้าไปด้วย โดยแผนภาพสถานะที่ UML สนับสนุนนั้นมี 2 แบบคือ แผนภาพสถานะพื้นฐานและแผนภาพสถานะที่แสดงส่วนประกอบอิสระ แผนภาพนี้สนับสนุนการทำงานพร้อมๆ กันภายในคลาส

4.4.7 แผนภาพ Component

แผนภาพ Component นั้นใช้อธิบายโครงสร้างทางด้านซอฟต์แวร์รวมทั้งยังอาจจะอธิบายโครงสร้างของโค้ดด้วย ในไดอะแกรมนี้จึงประกอบด้วย Package, Task และอาจรวมถึง Class ด้วย

4.4.8 แผนภาพ Deployment

แผนภาพ Deployment นั้นจะเป็นการอธิบายโครงสร้างทางด้านฮาร์ดแวร์ ดังนั้นในไดอะแกรมจึงประกอบด้วยโหนด ที่ใช้สำหรับแทนหน่วยประมวลผลและการเชื่อมต่อระหว่างโนดทางด้านกายภาพ ดังเช่นการเชื่อมต่อผ่านโมเด็ม หรือการเชื่อมต่อผ่านเน็ตเวิร์กเป็นต้น

4.4.9 แผนภาพ Interaction

แผนภาพ Interaction จะแสดงการโต้ตอบ ซึ่งประกอบไปด้วยกลุ่มของวัตถุและความสัมพันธ์ของวัตถุต่างๆ รวมถึงเมสเซจที่ส่งระหว่างวัตถุต่างๆด้วย

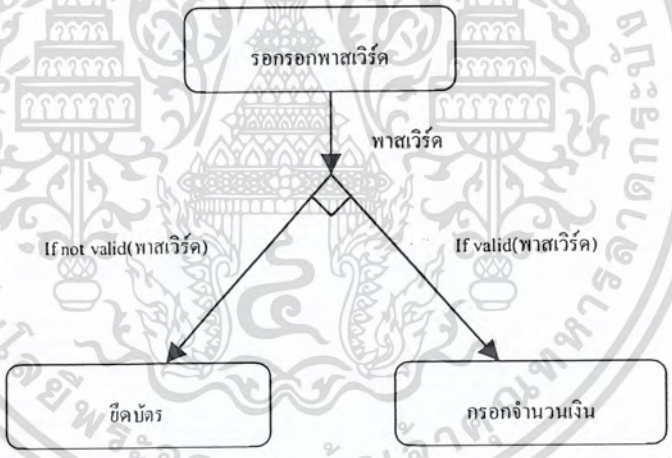
4.4.10 แผนภาพ Activity

แผนภาพ Activity จะแบ่งย่อยระบบให้เป็นกิจกรรมการทำงานย่อยๆ ซึ่งตัวแผนภาพกิจกรรมจะทำหน้าที่โมเดลการทำงานของ Activity นี้ เราอาจจะกล่าวได้ว่าแผนภาพกิจกรรมเป็น Flowchart ชนิดหนึ่งนั่นเอง

4.5 สัญลักษณ์ของ UML

UML นั้นประกอบไปด้วยสัญลักษณ์และเครื่องหมายจำนวนมาก โดยสัญลักษณ์และเครื่องหมายจำนวนมากของ UML เหล่านี้ ทำให้สามารถสร้างโมเดลและแผนภาพได้อย่างยืดหยุ่น และครอบคลุมโดเมนของปัญหาทั้งหมด รายละเอียดสำหรับสัญลักษณ์และเครื่องหมายของ UML ที่มีความสำคัญที่จะต้องทราบคือ Text Note, Constrain และ Stereotype

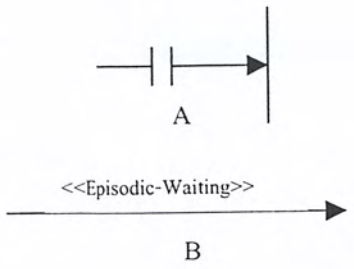
- **Text Note** เป็นส่วนหนึ่งของแผนภาพที่ไม่มีความหมายต่อโมเดลแต่อย่างใด โดยทั่วไปจะใช้ Text Note เพื่ออธิบายสิ่งที่อยู่ในแผนภาพบางสิ่งเพื่อให้ผู้อ่านหรือใช้ไดอะแกรมเข้าใจความหมายได้ดีขึ้น โดยปกติ Text Note จะเป็นตัวอักษรที่อยู่มุมบนขวาขององค์ประกอบในแผนภาพ
- **Constrain** เป็นเงื่อนไขที่เพิ่มเข้ามาสำหรับองค์ประกอบในแผนภาพ ตัวอย่างเช่น ในเงื่อนไขของอินพุทที่อยู่ในแผนภาพสถานะ โดยเงื่อนไขอินพุทนี้จะเป็นข้อกำหนดที่ใช้แสดงว่าระบบจะต้องตอบสนองต่ออินพุทที่มีเข้ามาจากภายนอก โดยปกติ Constrain จะมีอยู่ในจุดแยกหรือ Curly Braces และอาจจะปรากฏอยู่ใน Text Area ก็ได้



รูปที่ 4-9 ตัวอย่าง Constrain

- **Stereotype** เป็น Metamodel ของ UML หรือจะกล่าวง่ายๆว่า Stereotype เป็นเอนติตี ซึ่งเป็นส่วนขยายสำหรับ UML เพราะถึงแม้ว่าใน UML นั้นจะมีสัญลักษณ์และเครื่องหมายจำนวนมาก แต่ก็ยังไม่เพียงพอที่จะใช้แทนทุกๆสิ่งที่มีอยู่ในโลกความเป็นจริง แต่สามารถใช้ Metamodel นี้ในการแทนความหมายของสิ่งเหล่านั้นตามที่ต้องการได้ อย่างไรก็ดีสัญลักษณ์และเครื่องหมายของ UML ทั้งหมดสามารถเขียนอยู่ในรูปของ Stereotype ได้เช่นกัน ตัวอย่างเช่นเมสเซจที่มีลักษณะแบบ Episodic-Waiting ใน UML ได้ตามรูป 4-10A และสามารถเขียนเป็น Stereotype ได้ดังรูป 4-10B นอกจากนั้นยังสามารถสร้าง Metaclass โดย

ขยายข้อกำหนดหรือเพิ่มความเฉพาะเจาะจงให้กับสัญลักษณ์หรือเครื่องหมายของ UML ที่มีอยู่ได้ ซึ่งคุณสมบัตินี้ก็เหมือนกับการสืบทอดคุณสมบัตินั่นเอง



รูปที่ 4-10 รูปแบบการเขียนแมสเซจใน UML

สำหรับสัญลักษณ์ของ Stereotype นั้นจะเขียนชื่อเอนติตี ที่เป็น Stereotype ภายในวงเล็บ <<>> เช่น "<<type>> tree" เป็นการกำหนด Stereotype ของชนิดข้อมูลเพื่อใช้ในการเขียนโปรแกรม เมื่อผู้ดูไดอะแกรมเห็นสัญลักษณ์ของ Stereotype นี้จะเข้าใจทันทีว่าข้อมูลนั้นมีชนิดเป็น "tree" ตัว UML เองก็มีการกำหนดสัญลักษณ์ของ Stereotype พื้นฐานไว้สำหรับ application เฉพาะด้านเช่นกัน เอนติตีในโมเดลของ UML ทุกตัวสามารถเขียนเป็น Stereotype ได้ ตัวอย่างเช่น

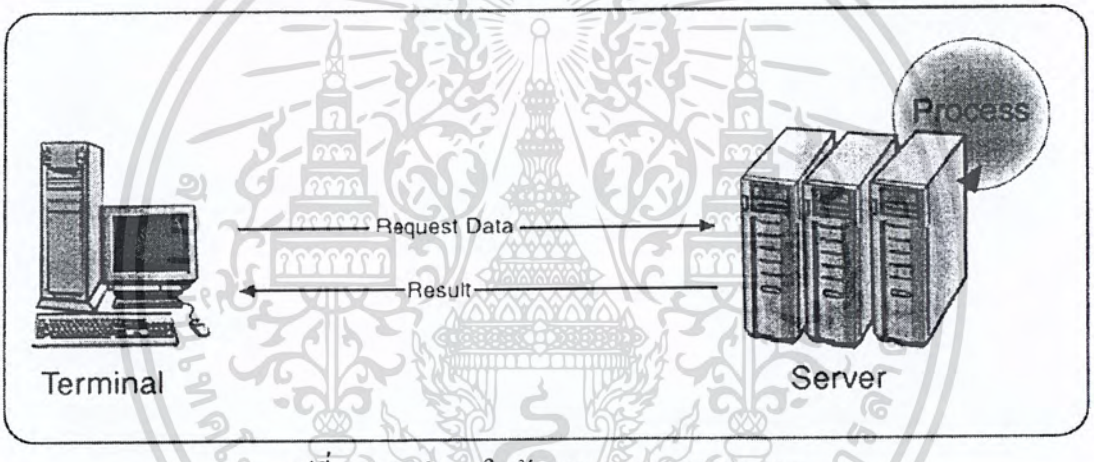
Stereotype	สัญลักษณ์	ใช้กับ
<<Interface>>		คลาส
<<Control>>		คลาส
<<Active Object>>		คลาส
<<Utility>>		คลาส
<<Actor>>		คลาส
<<Import>>		Dependency relation
<<uses>>		Use-case relation
<<parameter>>		Association role
<<signal>>		Event

ตารางที่ 4-1 ตัวอย่างของ Stereotype

บทที่ 5
ระบบเครือข่ายแบบไคลเอ็นต์/เซิร์ฟเวอร์ (Client/Server)

สำหรับการพัฒนาระบบให้ประสบความสำเร็จ นอกเหนือจากการพัฒนาโปรแกรมเพื่อให้สามารถทำงานได้ตามที่ผู้ใช้งานต้องการแล้ว สิ่งหนึ่งที่ควรพิจารณา คือ เวลาที่เครื่องใช้ในการทำงาน ในแต่ละครั้ง (Transaction) ไม่ควรใช้เวลาในการตอบสนองนานจนเกินไป ซึ่งปัจจัยหนึ่งที่มีผลต่อเวลาที่ใช้ก็คือ เรื่องของระบบเครือข่ายนั่นเอง

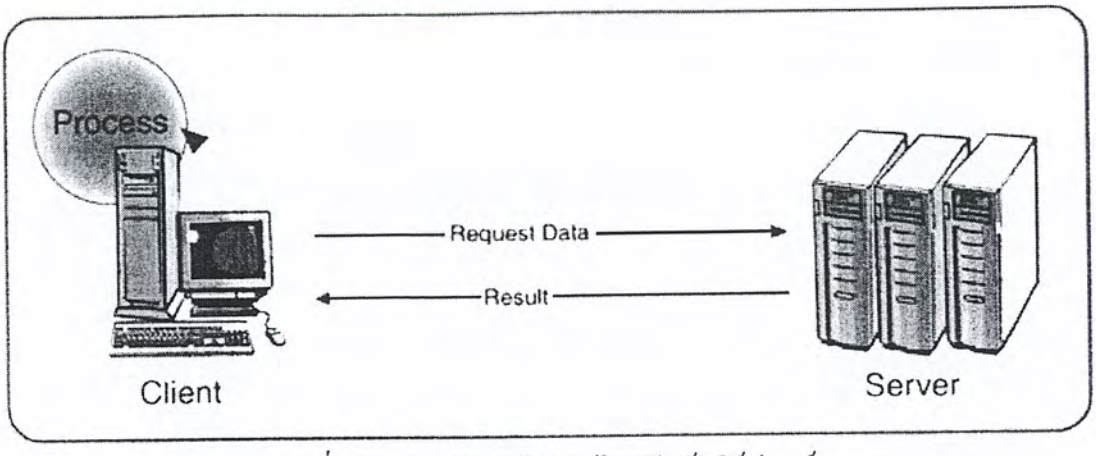
ในอดีต การพัฒนาระบบจะเป็นแบบ Host-based System คือ ในหน่วยงานจะมีเครื่องคอมพิวเตอร์ขนาดใหญ่ ซึ่งใช้เป็นทั้ง Application Server และ Database (File) Server และมีเครื่องคอมพิวเตอร์ขนาดเล็กเชื่อมต่อเข้ากับเครื่องเซิร์ฟเวอร์ทำหน้าที่เป็นเทอร์มินอลในการทำงาน เครื่องเทอร์มินอลจะส่งข้อมูลไปประมวลผลที่เครื่องเซิร์ฟเวอร์ และส่งผลลัพธ์มาแสดงที่เครื่องเทอร์มินอล ดังรูปที่ 5-1



รูปที่ 5-1 การทำงานในลักษณะ Host-based System

แนวความคิดแบบ Host-based System นี้จะมีความสะดวก คือ ไม่จำเป็นต้องใช้เครื่องเทอร์มินอลที่มีประสิทธิภาพสูงมาก และการเปลี่ยนแปลงแก้ไขแอพพลิเคชันจะสามารถทำได้ง่าย แต่ก็มีปัญหา คือ เมื่อระบบมีขนาดใหญ่มากขึ้น มีปริมาณเทอร์มินอลที่เชื่อมต่อมากขึ้น เครื่องเซิร์ฟเวอร์จะต้องรับภาระการทำงานมาก จนอาจมีผลต่อเวลาที่ใช้ในการตอบสนองผู้ใช้งานได้

เพื่อแก้ปัญหาดังกล่าว จึงได้เกิดแนวความคิดแบบไคลเอ็นต์/เซิร์ฟเวอร์โดยการพัฒนาระบบแบบไคลเอ็นต์/เซิร์ฟเวอร์นี้ เครื่องเซิร์ฟเวอร์จะทำหน้าที่เป็นเพียง Database (File) Server โดยการประมวลผลทั้งหมดจะอยู่ที่เครื่องเทอร์มินอล (Client) ดังรูปที่ 5-2_และเนื่องจากเครื่องไคลเอ็นต์ที่ใช้งานในปัจจุบันมีประสิทธิภาพมากขึ้น ทำให้สามารถสนับสนุนการทำงานในลักษณะนี้ได้



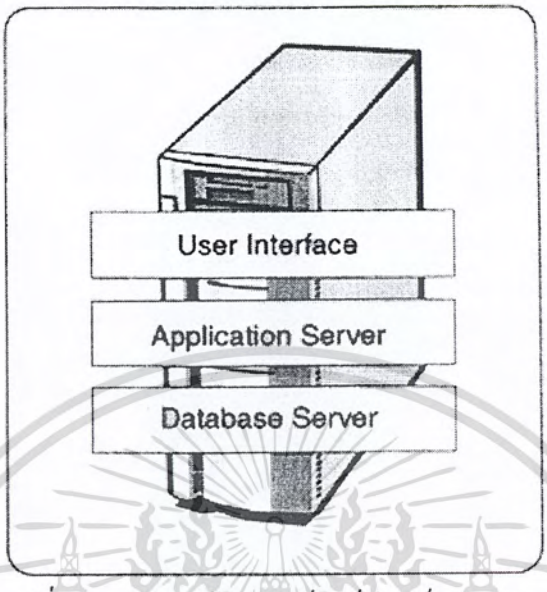
รูปที่ 5-2 การทำงานในลักษณะไคลเอ็นต์/เซิร์ฟเวอร์

5.1 รูปแบบของไคลเอ็นต์/เซิร์ฟเวอร์ที่ใช้งานมีอยู่ 4 ชนิดดังนี้

1. Stand alone Client/Server การทำงานแบบนี้ผู้ให้บริการหรือเซิร์ฟเวอร์จะอยู่บนเครื่องเดียวกับผู้ขอใช้บริการหรือไคลเอ็นต์ ทำให้มีความเร็วในการติดต่อสื่อสารระหว่างผู้ให้บริการและผู้ขอใช้บริการสูงมาก แต่ประสิทธิภาพในการประมวลผลระบบฐานข้อมูลจะลดลงบ้าง ระบบนี้เรียกอีกอย่างว่า Tiny Client/Server
2. Department Client/Server หรือ LAN based single server การทำงานแบบนี้จะมีผู้ให้บริการเกี่ยวกับฐานข้อมูล แอพพลิเคชัน ฯลฯ อยู่บนเครื่องเซิร์ฟเวอร์ และผู้ขอใช้บริการทั้งหลายจะอยู่บนเครื่องไคลเอ็นต์ โดยจะเชื่อมต่อกันด้วยระบบเครือข่ายท้องถิ่น (LAN) และมิดเดิลแวร์ (Middleware) เป็นตัวกลางที่ทำงานอยู่ระหว่างไคลเอ็นต์และเซิร์ฟเวอร์ การติดต่อระหว่างผู้ให้บริการและผู้ขอใช้บริการจะช้ากว่าแบบ Stand alone เพราะจะต้องติดต่อผ่านระบบเครือข่าย ยิ่งถ้ามีผู้ขอใช้บริการเข้ามาดึงข้อมูลกันครั้งละมากๆ หลายๆ เครื่อง ประสิทธิภาพจะลดลงอย่างเห็นได้ชัด ส่วนวิธีเพิ่มประสิทธิภาพ คือ การเพิ่มเครื่องเซิร์ฟเวอร์ขึ้นในระบบ
3. Workgroups Client/Server การทำงานแบบเวิร์กกรุ๊ปนี้จะเป็นกลุ่มของเซิร์ฟเวอร์ที่หลากหลายแพลตฟอร์ม หลายผู้ผลิต มีความแตกต่างกันของเซิร์ฟเวอร์ แต่ทั้งหมดนี้จะเชื่อมต่อกันทางระบบเครือข่าย LAN หรือ WAN และใช้มิดเดิลแวร์มาตรฐานในการทำงาน
4. Enterprise Client/Server การทำงานแบบเอ็นเตอร์ไพรท์หรือระดับองค์กรจะทำให้มีการเชื่อมโยงเครื่องเซิร์ฟเวอร์หรือโฮสต์ต่างแพลตฟอร์มเข้าด้วยกัน ทำให้มีการใช้ทรัพยากรบนระบบได้อย่างมีประสิทธิภาพสูงสุด โดยที่ไคลเอ็นต์สามารถจะเลือกใช้ทรัพยากรฐานข้อมูลจากเซิร์ฟเวอร์เครื่องใดก็ได้ผ่านทางมิดเดิลแวร์

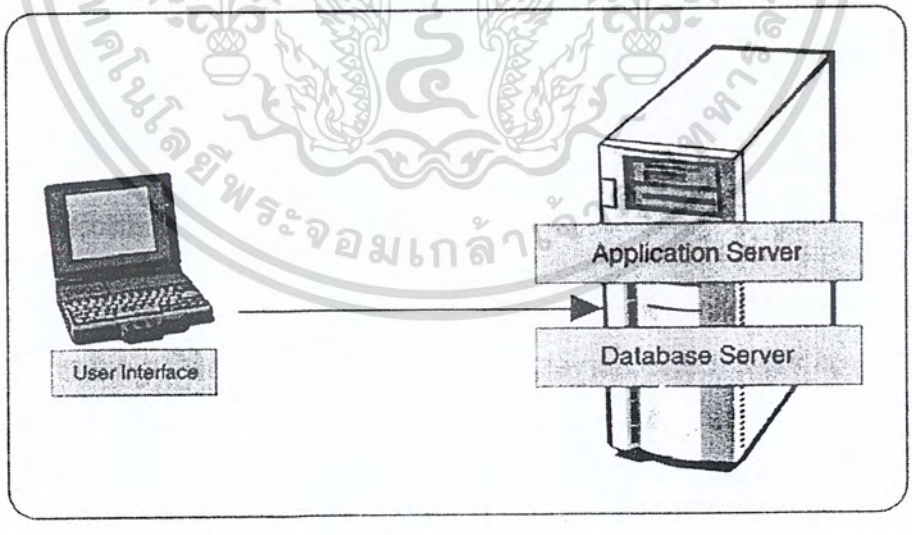
5.2 การจัดแบ่งประเภทของไคลเอ็นต์/เซิร์ฟเวอร์จะแบ่งตามลำดับชั้น (Tier) ในการทำงานดังนี้

- 1 Tier คือ การรวมการทำงานทุกอย่างเบ็ดเสร็จภายในเครื่องเดียว เป็นทั้งไคลเอ็นต์และเซิร์ฟเวอร์ มีลักษณะการทำงานคล้ายกับคอมพิวเตอร์แบบ Stand Alone



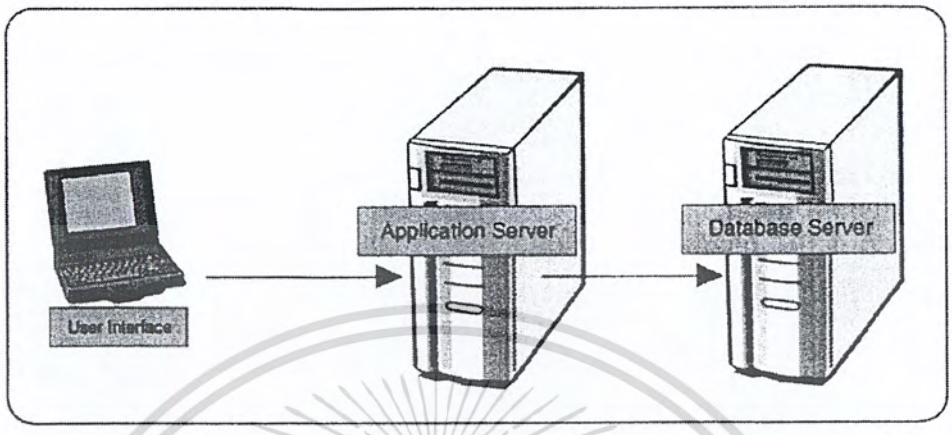
รูปที่ 5-3 การจัดแบ่งไคลเอ็นต์/เซิร์ฟเวอร์แบบ 1 Tier

- 2 Tiers คือ การแบ่งการทำงานเป็นไคลเอ็นต์และเซิร์ฟเวอร์โดยที่ฝั่งไคลเอ็นต์จะมีโปรแกรมเกี่ยวกับยูสเซอร์อินเทอร์เฟซที่สามารถประมวลผลเบื้องต้นได้ ก่อนที่จะสั่งคำสั่งไปขอข้อมูลจากฝั่งเซิร์ฟเวอร์ ซึ่งจะเป็นการลดภาระงานสำหรับเซิร์ฟเวอร์



รูปที่ 5-4 การจัดแบ่งไคลเอ็นต์/เซิร์ฟเวอร์แบบ 2 Tiers

- 3 Tiers คือ การแบ่งการทำงานเป็นไคลเอ็นต์และเซิร์ฟเวอร์ คล้ายกับแบบ 2 Tier แต่ว่าจะกระจายการทำงานของเซิร์ฟเวอร์ออกเป็นอีก 2 Tier โดยที่ Tier หนึ่งทำงานเป็นดาต้าเบสเซิร์ฟเวอร์ และอีก Tier หนึ่งทำงานเป็นแอพพลิเคชันเซิร์ฟเวอร์



รูปที่ 5-5 การจัดแบ่งไคลเอ็นต์/เซิร์ฟเวอร์แบบ 3 Tiers

5.3 การจัดแบ่งประเภทของเซิร์ฟเวอร์จะแบ่งตามประเภทการใช้งาน

File Server คือ เซิร์ฟเวอร์ทำหน้าที่จัดเก็บไฟล์ที่สร้างขึ้น เพื่อให้ไคลเอ็นต์อื่นๆ สามารถเข้ามาใช้ไฟล์ร่วมกันได้ โดยอาจจะกำหนดสิทธิ์ในการใช้ตามกลุ่มของผู้ใช้ เช่น พนักงานในแผนกการเงินสามารถเรียกใช้ข้อมูลได้เฉพาะภายในแผนกเท่านั้น ไม่สามารถเรียกข้อมูลที่สร้างโดยแผนกบุคคลได้ ในขณะที่ผู้บริหารสามารถเรียกใช้ข้อมูลของทุกแผนกได้

Database Server คือ เซิร์ฟเวอร์ที่ทำหน้าที่เป็น DBMS คอยจัดการกับข้อมูลที่จัดเก็บอยู่ในเซิร์ฟเวอร์เดียวกันนี้ ทำให้ผู้ใช้ที่เป็นไคลเอ็นต์สามารถเข้ามาแชร์ใช้ข้อมูลร่วมกัน ทำให้ข้อมูลที่ได้มีความทันสมัย ไม่ซ้ำซ้อน

Application Server คือ เซิร์ฟเวอร์ที่ทำหน้าที่จัดการกับโปรแกรมแอพพลิเคชันในการเชื่อมต่อระหว่าง Database Server กับผู้ใช้ (User) เซิร์ฟเวอร์กับยูสเซอร์ที่เป็นไคลเอ็นต์เพื่อให้ทั้งสองส่วนนี้สามารถเชื่อมต่อกันได้อย่างสมบูรณ์

บทที่ 6
ทฤษฎีและความรู้พื้นฐานเกี่ยวกับระบบฐานข้อมูล

6.1 ระบบจัดการฐานข้อมูลเชิงวัตถุสัมพันธ์

ระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Object-Relational Database Management System : ORDBMS) เป็นระบบจัดการฐานข้อมูลที่มีความสามารถของระบบจัดการฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System : RDBMS) โดยในโครงงานนี้ได้เลือกใช้ระบบจัดการฐานข้อมูลเชิงสัมพันธ์ของบริษัทออราเคิลที่มีชื่อว่า ออราเคิล8i (Oracle8i) เวอร์ชัน 8.1.7

6.1.1 ความสามารถของระบบจัดการฐานข้อมูลเชิงสัมพันธ์

ความสามารถในการเก็บและจัดการข้อมูลที่อยู่ในลักษณะความสัมพันธ์แบบรีเลชันซึ่งตัวอย่างหนึ่งของความสัมพันธ์แบบรีเลชันนี้ คือ ตาราง โดยมีความสัมพันธ์กันระหว่างแถว (Row) และสดมภ์ (Column) ตัวอย่างของระบบจัดการฐานข้อมูลแบบนี้คือ ระบบจัดการฐานข้อมูลที่ใช้ภาษา SQL เป็นหลัก (SQL – Base Database Management System) ซึ่งมีคุณสมบัติดังนี้

1. โครงสร้างข้อมูล (Data Structure หรือ Logical data structure) เป็นในรูปแบบของความสัมพันธ์โดยแสดงออกมาในรูปแบบของตารางซึ่งข้อมูลในตารางจะต้องมีคุณสมบัติดังนี้
   - ข้อมูลในแต่ละแถวไม่ซ้ำกัน
   - ไม่จำเป็นต้องมีการเรียงลำดับของแถวต่างๆ
   - ข้อมูลในแต่ละสดมภ์จะไม่สามารถแบ่งแยกอีกได้
2. กฎบังคับความถูกต้องของข้อมูล (Data Integrity) เป็นกฎบังคับว่าข้อมูลที่ทำการใส่ลงฐานข้อมูลนั้นต้องมีความถูกต้องตามกฎข้อบังคับ (Constraint) ที่ตั้งไว้
3. ภาษาที่ใช้ในการจัดการข้อมูล (Data Manipulation)

6.1.2 ความสามารถของระบบจัดการฐานข้อมูลแบบออบเจ็กต์

มีการเก็บข้อมูล (Data หรือ Property) และการใช้ข้อมูล (Procedure หรือ Method) เข้าด้วยกันอยู่ในลักษณะของออบเจ็กต์ ซึ่งมีคุณสมบัติทางออบเจ็กต์ดังนี้

1. ความสามารถในการซ่อนรายละเอียด (Encapsulation) เป็นการซ่อนการทำงานภายในของออบเจ็กต์ไว้ ซึ่งออบเจ็กต์อื่นจะรู้แค่อินเทอร์เฟซ (Interface) เท่านั้น
2. คุณสมบัติการโพลีมอร์ฟิซึม (Polymorphism)
3. ความสัมพันธ์แบบออบเจ็กต์ คือ
   - ความสัมพันธ์แบบแอคกรีเกชั่น (Aggregation)
   - ความสัมพันธ์แบบสืบทอด (Generalization)

ในระบบจัดการฐานข้อมูลเชิงวัตถุเชิงสัมพันธ์ที่ทำการศึกษาและนำมาใช้งานคือ ระบบจัดการฐานข้อมูลออราเคิล ซึ่งมีความสามารถดังนี้

1. สามารถใช้ชนิดข้อมูลมาตรฐาน (Built – In Datatypes)
2. สามารถใช้ชนิดข้อมูลแบบกำหนดเอง (Users-Defined Datatypes) ซึ่งประกอบด้วย
   2.1 ชนิดข้อมูลแบบออบเจ็กต์ (Object Types)
   2.2 ชนิดข้อมูลแบบกลุ่ม (Collection Types)

และทั้งสองหัวข้อนี้จะถูกนำเสนอถึงรายละเอียดทั้งหมดภายในบทนี้

6.2 ชนิดข้อมูลมาตรฐาน

ใช้ในการระบุชนิดของข้อมูลที่ต้องการเก็บลงฐานข้อมูล ซึ่งมีชื่อและลักษณะที่แตกต่างกันในระบบจัดการฐานข้อมูลที่มีอยู่มากมายในปัจจุบัน โดยชนิดข้อมูลมาตรฐานที่จะนำเสนอนี้จะเป็นชนิดข้อมูลมาตรฐานของระบบจัดการฐานข้อมูลออราเคิล

6.2.1 ชนิดข้อมูลแบบตัวอักษร (Character Datatypes)

ใช้สำหรับการเก็บข้อมูลที่เป็นตัวอักษร ซึ่งอยู่ในรูปแบบของการเข้ารหัสตัวอักษร (Character set) ในลักษณะต่างๆ เช่น ASCII, EBCDIC ฯลฯ ซึ่งมีการแบ่งประเภทของชนิดข้อมูลแบบตัวอักษรนี้ย่อยลงไปอีกได้ดังนี้

6.2.1.1 CHAR Datatype

ใช้เก็บข้อความที่มีการกำหนดความยาวของข้อความไว้แน่นอน ซึ่งมีความยาวข้อมูลอยู่ระหว่าง 1 ถึง 200 ไบต์ และเมื่อทำการเพิ่มหรือเปลี่ยนแปลงข้อมูลที่มีความยาวน้อยกว่าที่กำหนด ข้อมูลชุดนั้นจะถูกแทนที่ด้วยช่องว่างที่เป็นรหัสแอสกีตัวหนึ่ง

6.2.1.2 VARCHAR2 Datatype

ใช้เก็บข้อความที่มีจำนวนของตัวอักษรในข้อความนั้นเปลี่ยนแปลงได้ สามารถกำหนดจำนวนของตัวอักษรมากที่สุดที่อยู่ในหนึ่งข้อความได้ โดยมีค่าได้ระหว่าง 1 ถึง 4000

6.2.1.3 VARCHAR Datatype

เป็นชนิดข้อมูลที่ใช้ในออราเคิลรุ่นเก่าซึ่งยังคงไว้เพื่อสามารถใช้เก็บข้อมูลที่มีอยู่ในฐานข้อมูลออราเคิลรุ่นเก่าได้ โดยมีคุณสมบัติคล้ายกับ VARCHAR2

6.2.1.4 NCHAR และ NVARCHAR2 Datatype

ใช้เก็บข้อมูลที่มีการเข้ารหัสแบบ NLS โดย NCHAR จะมีคุณสมบัติเหมือน CHAR และ NVARCHAR2 จะมีคุณสมบัติเหมือน VARCHAR2

6.2.1.5 LONG Datatype

ใช้เก็บข้อความที่มีจำนวนของตัวอักษรในข้อความนั้นเปลี่ยนแปลงได้ และสามารถเก็บจำนวนตัวอักษรสูงสุดได้มากกว่า VARCHAR2

6.2.2 ชนิดข้อมูลแบบตัวเลข (Number Datatype)

ใช้เก็บข้อมูลที่เป็นตัวเลข ซึ่งประกอบด้วยเลขจำนวนเต็มและจำนวนจริง โดยมีขนาดข้อมูลสูงสุด 21 ไบต์

6.2.3 ชนิดข้อมูลที่เป็นวันที่ (Date Datatype)

ใช้เก็บข้อมูลประเภทเวลา ประกอบด้วยปี, เดือน, วัน, ชั่วโมง, นาที, วินาที ซึ่งมีขนาดคงที่ 7 ไบต์ สำหรับรูปแบบในการเก็บนั้นสามารถกำหนดได้ ซึ่งจะนำเสนอในส่วนของการเปลี่ยนชนิดข้อมูล

6.2.4 ชนิดข้อมูลที่มีขนาดใหญ่ (LOB Datatypes)

ใช้เก็บข้อมูลขนาดใหญ่ที่ไม่มีโครงสร้างข้อมูลที่แน่นอน เช่น ข้อมูลรูปภาพ ข้อมูลมัลติมีเดีย ซึ่งการเก็บข้อมูลด้วยชนิดของการเก็บข้อมูลนี้ทำให้สามารถเข้าถึงบข้อมูลแบบสุ่มได้ (Random access) ในขณะที่การเก็บข้อมูลชนิด LONG จะเข้าถึงข้อมูลแบบเป็นลำดับเท่านั้น (Sequential access) ชนิดข้อมูลประเภทนี้ประกอบด้วย

6.2.4.1 BLOB Datatype

ใช้เก็บข้อมูลไบนารีขนาดใหญ่ที่ไม่มีโครงสร้าง (Unstructured Binary Data)

6.2.4.2 CLOB และ NCLOB Datatype

ใช้เก็บข้อมูลข้อความที่มีจำนวนตัวอักษรมากๆ

6.2.4.3 BFILE Datatype

ใช้เก็บข้อมูลไบนารีขนาดใหญ่โดยทำการเก็บข้อมูลนั้นไว้นอกฐานข้อมูล และเก็บเพียงพอยน์เตอร์นั้นไว้ ซึ่งจะเป็นการเก็บข้อมูลเพื่ออ่านอย่างเดียว

6.2.5 RAW และ LONG RAW Datatypes

ใช้เก็บข้อมูลที่มีขนาดใหญ่โดย RAW มีคุณสมบัติเหมือน VARCHAR2 ส่วน LONG RAW นั้นเหมือน LONG สามารถใช้เก็บข้อมูลที่เป็นมัลติมีเดียได้แต่ประสิทธิภาพจะไม่ดีเท่าใช้ LOB และไม่สามารถทำการ export

6.2.6 ROWID Datatype

ใช้เก็บข้อมูลที่เป็นตำแหน่งจริงของแถวในฐานข้อมูล

6.3 ชนิดข้อมูลแบบกำหนดเอง

เกิดจากการสร้างชนิดข้อมูลขึ้นมาจากชนิดข้อมูลมาตรฐาน และชนิดข้อมูลกำหนดเองอื่นๆ ซึ่งได้ถูกแบ่งออกมาสองลักษณะดังนี้

6.3.1 ชนิดข้อมูลแบบออบเจ็กต์

เป็นชนิดของข้อมูลที่ผู้ใช้สามารถทำการสร้างขึ้นมาเอง โดยทำการกำหนดลักษณะของข้อมูลให้มีความใกล้เคียงกับความเป็นจริงมากที่สุด ซึ่งเป็นการใช้ลักษณะเด่นของข้อมูลนั้นเป็นตัวระบุ ซึ่งชนิดข้อมูลแบบออบเจ็กต์นี้ประกอบด้วยส่วนประกอบ 3 ส่วนดังนี้

1. ชื่อ (Name) ใช้ในการระบุถึงชนิดข้อมูลแบบออบเจ็กต์ ซึ่งต้องไม่ซ้ำกันในโครงสร้างที่เก็บข้อมูลแบบออบเจ็กต์เดียวกัน (Schema)
2. ลักษณะต่างๆ (Attributes) คือข้อมูลที่ออบเจ็กต์ควรมีเพื่อทำให้ออบเจ็กต์นั้นมีความใกล้เคียงกับความเป็นจริงให้มากที่สุด ซึ่งจะเป็นชนิดข้อมูลแบบมาตรฐาน หรือชนิดข้อมูลแบบกำหนดเองก็ได้
3. การกระทำ (Method) เป็นฟังก์ชันหรือโพรซีเจอร์ ที่สร้างจากภาษา PL/SQL เพื่อสร้างการกระทำให้กับออบเจ็กต์

ตัวอย่าง การสร้างข้อมูลชนิดออบเจ็กต์

```
CREATE OR REPLACE TYPE person_object AS OBJECT(
    ssn          NUMBER,
    last_name    VARCHAR2(50),
    first_name   VARCHAR2(50);
```

6.3.2 ชนิดข้อมูลแบบกลุ่ม

เป็นลักษณะของข้อมูลที่มีชนิดของข้อมูลเหมือนกันและมีจำนวนที่ไม่สามารถระบุได้ว่ามีเท่าไร ซึ่งจะแบ่งเป็น ชนิดข้อมูลแบบอาร์เรย์ (Array Types) หรือชนิดข้อมูลแบบ วีอาร์เรย์ (VArray) และชนิดข้อมูลแบบ ตาราง (Table Types) หรือชนิดข้อมูลแบบตารางซ้อนตาราง (Nested Table)

- ชนิดข้อมูลแบบวีอาร์เรย์

เป็นการเก็บข้อมูลที่มีชนิดของข้อมูลเหมือนกัน และข้อมูลที่อยู่ในแต่ละช่องของอาร์เรย์จะถูกระบุด้วยดัชนี (Index) ซึ่งเป็นลักษณะของเซตของข้อมูลแบบมีลำดับ

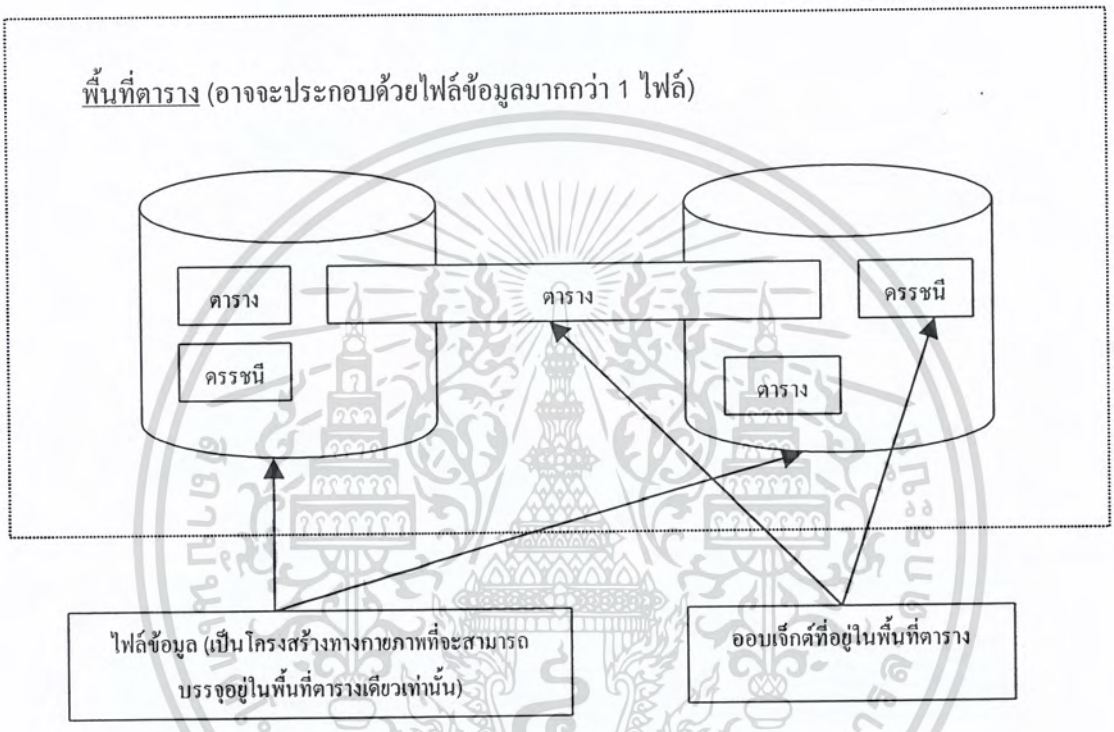
ตัวอย่าง การสร้างชนิดข้อมูลแบบวีอาร์เรย์

```
CREATE TYPE Project AS OBJECT(
    project_no      NUMBER(2),
```

```
    title           VARCHAR2(35),
    cost            NUMBER(7,2))
CREATE TYPE ProjectList AS VARRAY(50) OF Project
```

6.4 ฐานข้อมูล (Database) พื้นที่ตาราง (Tablespaces) และไฟล์ข้อมูล (Datafiles)

ข้อมูลของออราเคิลที่ทำการเก็บลงในฐานข้อมูลนั้นสามารถพิจารณาใน 2 มุมมอง คือ ในทางด้านความคิดหรือทางลอจิกนั้นจะเก็บข้อมูลลงในพื้นที่ตาราง (Tablespaces) และในมุมมองทางด้านกายภาพนั้นจะทำการเก็บข้อมูลลงในไฟล์ข้อมูลที่กำหนดในพื้นที่ตารางดังแสดงในรูปที่ 6-1



รูปที่ 6-1 พื้นที่ตารางและไฟล์ข้อมูล

แม้ว่าฐานข้อมูล พื้นที่ตารางและไฟล์ข้อมูลจะมีความสัมพันธ์กันแต่ก็มีความแตกต่างกันอยู่บ้าง คือ ในฐานข้อมูลของออราเคิลจะมีหน่วยทางตรรกที่จัดเก็บข้อมูลทั้งหมดคือพื้นที่ตารางและในพื้นที่ตารางนั้นจะมีไฟล์ที่ทำการเก็บข้อมูลในทางกายภาพเรียกว่าไฟล์ข้อมูล

เมื่อเราสามารถสร้างพื้นที่ตารางแล้วเราสามารถสร้างตารางขึ้นมาใช้งานได้ โดยไม่เกี่ยวข้องกับพื้นที่ตารางระบบของทางออราเคิลที่ได้สร้างขึ้นมาเมื่อเราได้ติดตั้งระบบ

6.5 Java Database Connectivity

ภาษาจาวาเป็นภาษาคอมพิวเตอร์ที่ได้รับการพัฒนาอย่างรวดเร็วและต่อเนื่อง นอกเหนือจากการสร้างกราฟฟิกหรือองค์ประกอบที่ใช้ในการประดับโฮมเพจ (Homepage) ยังมีการพัฒนาองค์ประกอบใหม่ๆ ในการปฏิบัติงานในด้านต่างๆได้ถูกเพิ่มเติมประสิทธิภาพให้แก่จาวา คือ JDBC (Java Database

Connectivity) ซึ่งทำหน้าที่อำนวยความสะดวกในการเชื่อมต่อและสื่อสารกับระบบฐานข้อมูล โดยพื้นฐานของภาษาจาวาที่ได้รับการออกแบบมาเพื่อปฏิบัติงานกับระบบเครือข่ายคอมพิวเตอร์โดยตรง องค์ประกอบนี้จึงทำให้จาวาเหมาะสมเป็นอย่างยิ่งในการเป็นเครื่องมือสำหรับสร้างโปรแกรมประยุกต์เพื่อเข้าถึงข้อมูลแบบไคลเอ็นต์เซิร์ฟเวอร์ โดยมี JDBC เป็นองค์ประกอบในการเชื่อมการถ่ายทอดคำสั่งการเรียกค้นข้อมูลจากไคลเอ็นต์ไปสู่เซิร์ฟเวอร์ และรับข้อมูลผลลัพธ์จากเซิร์ฟเวอร์กลับคืนไปสู่ไคลเอ็นต์

6.5.1 ความหมายของ JDBC

JDBC (Java Database Connectivity) ถูกพัฒนาโดย JavaSoft Department ของบริษัท Sun Microsystems ซึ่งก็คือฟังก์ชันมาตรฐานหรือ Java Application Programming Interface (API) สำหรับการเชื่อมต่อกับระบบฐานข้อมูล, นักพัฒนาสามารถใช้ JDBC API และยังประกอบด้วย packages อื่นๆด้วย ซึ่งนำเสนอในรูปแบบฟังก์ชันพิเศษหรือ higher level API เพื่อเขียนโปรแกรมอิสระในการกำหนด DBMS หรือ database connectivity mechanism ในการเริ่มใช้งาน, โดยทั่วไปการใช้ SQL database ในการติดต่อ (interface) กับ framework เพื่อที่จะจัดมาตรฐานในการติดต่อในส่วนบนสุดของชนิดต่างๆ ของ database connectivity modules ซึ่งก็คือมาตรฐานของ ANSI SQL-2 Entry level database เพราะว่า Relational databases เกือบจะทั้งหมดในปัจจุบันใช้มาตรฐานของ SQL-2 Entry level

JDBC สร้างระดับการเชื่อมต่อเพื่อสื่อสารกับฐานข้อมูลในรูปแบบที่คล้ายคลึงกับ ODBC Database Connectivity ของบริษัทไมโครซอฟต์ ซึ่งในปัจจุบันได้ถือว่าเป็นมาตรฐานของการเชื่อมต่อกับฐานข้อมูลสำหรับเครื่องคอมพิวเตอร์ส่วนบุคคล (PC) และระบบเครือข่ายท้องถิ่น (LAN) แต่ตามหลักการทำงานของทั้ง JDBC และ ODBC ตั้งอยู่บนมาตรฐานเดียวกันคือ X/Open SQL Call-level Interface ของระบบ X-Windows และ JDBC driver ต้องเข้ากันได้กับระดับมาตรฐานในการเข้าถึง SQL (ANSI SQL Entry Level Standard) และต้องผ่าน Conformance test ซึ่ง JavaSoft เป็นผู้กำหนดขึ้น

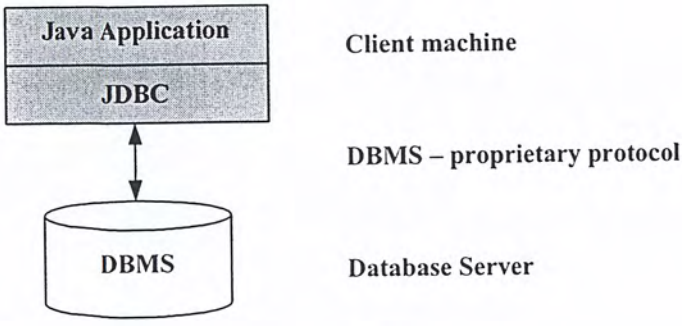
6.5.2 รูปแบบการเชื่อมต่อฐานข้อมูลของ JDBC

JDBC API สนับสนุนรูปแบบการเชื่อมต่อฐานข้อมูลทั้งแบบ Two-Tiers model และ Three-Tiers model

- Two – Tiers model

จาวาแอพเพล็ต (Java applet) หรือจาวาแอพพลิเคชัน (Application) จะติดต่อกับฐานข้อมูลโดยตรง จึงมีความจำเป็นที่โปรแกรมจาวาต้องการ JDBC ไดรเวอร์พิเศษที่สามารถสื่อสารกับระบบจัดการฐานข้อมูลชนิดนั้นได้ รูปแบบการเชื่อมต่อแสดงดังรูปที่ 6-2 คำสั่งในการค้นคืนข้อมูลในรูปของภาษา SQL (Structured Query Language) จะถูกส่งจากผู้ใช้ไปสู่ฐานข้อมูล หลังจากนั้นผลจากการประมวลผลของระบบจัดการฐานข้อมูล ก็จะถูกส่งกลับมาสู่ผู้ใช้ฐานข้อมูลดังกล่าวนี้ส่วนมากจะติดตั้งอยู่ต่างเครื่องคอมพิวเตอร์ โดยผู้ใช้สามารถเชื่อมต่อผ่านระบบเน็ตเวิร์ก (Network) รูปแบบ Two – Tiers นี้ใช้หลักการทำงานเช่นเดียวกันกับรูปแบบไคลเอ็นต์/เซิร์ฟเวอร์ ที่เรารู้จักกันดี โดยที่เครื่องคอมพิวเตอร์ของผู้ใช้คือ

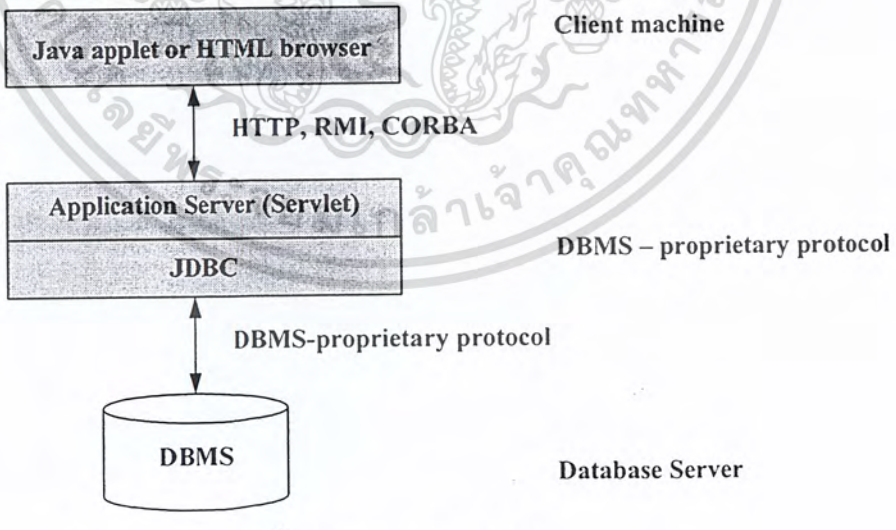
ไคลเอ็นต์และเครื่องคอมพิวเตอร์ ที่ให้บริการฐานข้อมูลคือเซิร์ฟเวอร์ เครือข่ายคอมพิวเตอร์ที่นิยมใช้รูปแบบ Two – Tiers มักเป็นเครือข่ายอินทราเน็ต (Intranet) สำหรับดำเนินธุรกรรมภายในองค์กร



รูปที่ 6-2 Two – Tiers Model

- **Three – Tiers Model**

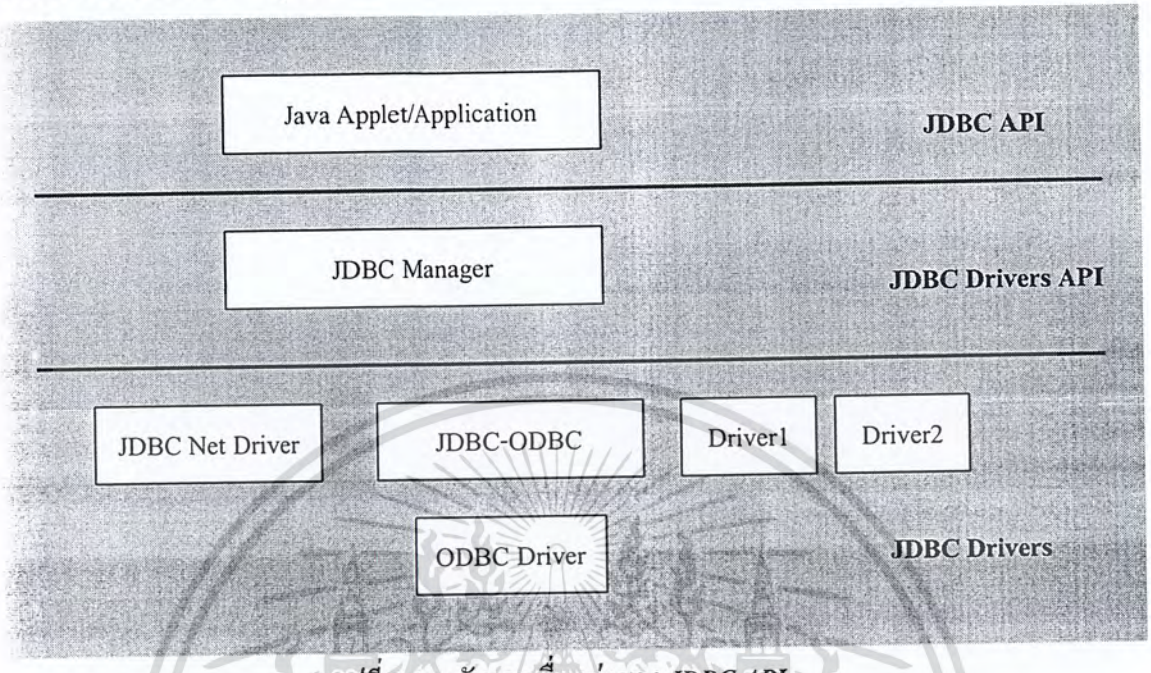
คำสั่งค้นคืนต่างๆจากผู้ใช้จะถูกส่งไปให้กับ Middle Tier หรือส่วนกลางของการบริการเสียก่อน หลังจากนั้น Middle Tier จะแปลงคำสั่งเหล่านี้ให้เป็นภาษา SQL เพื่อส่งไปให้กับระบบจัดการฐานข้อมูล เพื่อทำการประมวลผล ข้อมูลผลลัพธ์ที่ได้ก็จะถูกส่งกลับคืนไปให้กับ Middle Tier และส่งต่อไปให้ผู้ใช้ในที่สุด หลักการทำงานเช่นนี้มักจะพบในเครือข่ายอินเทอร์เน็ตซึ่งประกอบด้วยเครื่องคอมพิวเตอร์หลากหลายชนิด และเว็บเซิร์ฟเวอร์ ซึ่งทำหน้าที่เป็น Middle Tier ก็จะเป็นตัวกลางในการจัดการให้คอมพิวเตอร์ทั้งไคลเอ็นต์และเซิร์ฟเวอร์ฐานข้อมูลสามารถพูดคุยกันได้ การปรับเปลี่ยนระบบคอมพิวเตอร์ของไคลเอ็นต์และเซิร์ฟเวอร์ หรือแม้กระทั่งการเปลี่ยนฐานข้อมูลตัวใหม่จะไม่มีผลกระทบซึ่งกันและกันเกิดขึ้น



รูปที่ 6-3 Three – Tiers Model

6.5.3 โครงสร้างของ JDBC

โครงสร้างการเชื่อมต่อภายใน JDBC ประกอบด้วย 3 ระดับหลักคือ JDBC API, JDBC Driver API และ JDBC Driver ดังรูปที่ 6-4



รูปที่ 6-4 ระดับการเชื่อมต่อของ JDBC API

JDBC API ประกอบด้วยส่วนติดต่อ (Interface) 2 ส่วนที่สำคัญ ส่วนแรกคือ JDBC API เป็นระดับของฟังก์ชัน API ที่อำนวยความสะดวกให้แก่ผู้เขียนโปรแกรมประยุกต์(application writers) ส่วนที่สอง คือ JDBC driver API ในระดับ lower-level สำหรับผู้เขียนไดรเวอร์ (driver writers)

6.5.4 รูปแบบของ JDBC ไดรเวอร์ (JDBC drivers)

JDBC ไดรเวอร์หรือส่วนที่อยู่เบื้องหลังการทำงานของฟังก์ชัน API ต่างๆของ JDBC โดยเทคโนโลยี JDBC drivers ถูกแบ่งออกเป็น 4 ประเภทด้วยกันคือ

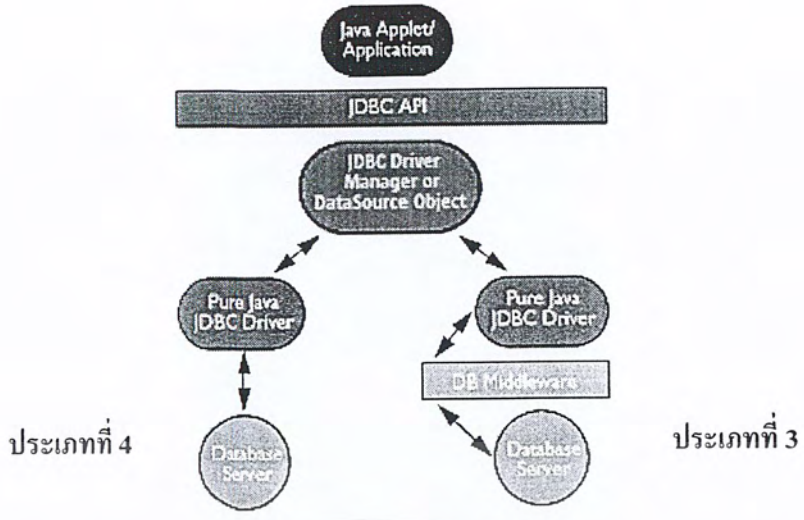
ประเภทที่ 1: JDBC-ODBC Bridge plus ODBC Driver

ประเภทที่ 2: Native-API (partly Java) Driver

ประเภทที่ 3: Pure Java Driver สำหรับ Database Middleware

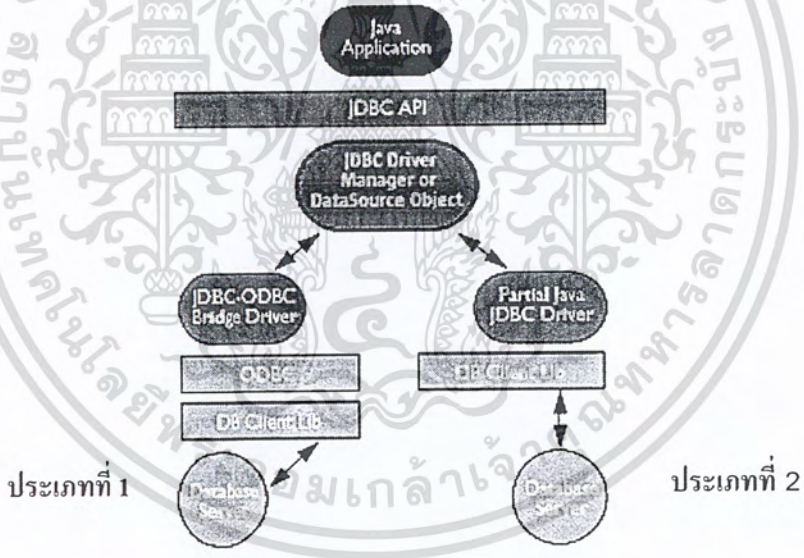
ประเภทที่ 4 : Direct-to-Database Pure Java Driver

แอพพลิเคชันและแอพเพล็ตสามารถเข้าถึงฐานข้อมูลผ่าน JDBC API โดยใช้เทคโนโลยี Pure Java JDBC driver ดังแสดงในรูปที่ 6-5



รูปที่ 6-5 JDBC driver ประเภทที่ 3 และ 4 ที่ใช้ pure Java JDBC driver

JDBC Connectivity โดยใช้ ODBC ไดรเวอร์ (ODBC drivers) และ libraries ของฐานข้อมูลในเครื่องไคลเอ็นต์ที่มีอยู่แล้ว แสดงในรูปที่ 6-6



รูปที่ 6-6 JDBC driver ประเภทที่ 1 และ 2

6.5.4.1 ประเภท 1 (Type1: JDBC-ODBC Bridge)

มักเรียกกันสั้นๆว่า JDBC-ODBC บริดจ์ หมายถึง JDBC ที่ทำงานอยู่บนชั้นการสื่อสารมิดเดิลแวร์ที่ชื่อ ODBC อีกทีหนึ่ง ถูกใช้สำหรับการศึกษาวิธีการเขียนโปรแกรมจาวา เพื่อติดต่อกับ JDBC เหมาะกับระบบงานที่มี ODBC ในการใช้งานอยู่แล้ว

- สำหรับระบบงานที่ทั้งหมดทำงานอยู่บนพื้นฐานของแพลตฟอร์มวินโดว์

- ไม่เหมาะกับการนำไปใช้งานบนระบบงานที่มีขนาดใหญ่ๆ เนื่องจากทำให้เกิดความช้าในการทำงาน และประสิทธิภาพในการทำงานที่ไม่ดี
- มีข้อมูลในส่วนโอเวอร์เฮด(Overhead) สูง เนื่องจากต้องมีส่วนในการติดต่อระหว่าง JDBC และ ODBC เพิ่มเติม
- ไม่สนับสนุนความสามารถทั้งหมดของมาตรฐาน JDBC เนื่องจากข้อจำกัดของ ODBC ที่มีรูปแบบการทำงานน้อยกว่า JDBC

6.5.4.2 ประเภท 2 (Type2: Native-API Driver)

สำหรับ JDBC ประเภทนี้ ตัวไดรเวอร์ (มิดเดิลแวร์ของแต่ละผลิตภัณฑ์มักจะใช้คำว่าไดรเวอร์แทนมิดเดิลแวร์ตัวนั้นๆ) จะถูกโหลดไว้ที่ไคลเอ็นต์หรือ Tier ส่วนติดต่อกับผู้ใช้

- มีประสิทธิภาพในการดีกว่าประเภท 1 เมื่อเปรียบเทียบกัน
- คำสั่งในการติดต่อกับเซิร์ฟเวอร์ จะเป็นคำสั่งที่ดีที่สุดสำหรับเซิร์ฟเวอร์นั้นๆ โดยเฉพาะ ทำการทำงานโดยรวมดีกว่า

ข้อด้อยของการนำไปใช้งาน

- ผู้ใช้โปรแกรมในส่วนไคลเอ็นต์ยังต้องการไดรเวอร์ สำหรับเซิร์ฟเวอร์โดยเฉพาะ
- เมื่อมีการเปลี่ยนแปลงเซิร์ฟเวอร์เป็นผลิตภัณฑ์ตัวอื่น โปรแกรมในส่วนไคลเอ็นต์ต้องมีการเปลี่ยนแปลงและคอมไพล์ใหม่เสมอ

6.5.4.3 ประเภท 3 (Type 3: Pure Java Driver)

ต้องมีแอพพลิเคชันเซิร์ฟเวอร์ทำงานร่วมอยู่ในระบบ

- เป็นระบบที่มีประสิทธิภาพในการทำงานที่ดีกว่าประเภทที่ 1 และ 2 เมื่อเปรียบเทียบกัน
- เหมาะสำหรับองค์กรที่มีเซิร์ฟเวอร์ทางด้านฐานข้อมูลที่หลากหลายรูปแบบ
- การทำงานของไคลเอ็นต์ไม่จำเป็นต้องติดตั้ง JDBC ไดรเวอร์ไว้ในทุกตัว
- การติดตั้งและการดูแลระบบไคลเอ็นต์ทำได้ง่าย และสะดวก
- ยังต้องการไดรเวอร์สำหรับแต่ละผลิตภัณฑ์ เพื่อติดตั้งไว้ที่แอพพลิเคชันเซิร์ฟเวอร์

6.5.4.4 ประเภท 4 (Type 4: Direct-to-DB)

ลักษณะการทำงานของ JDBC ในประเภทที่ 4 มีการทำงานโดยการส่งคำสั่ง SQL จากไคลเอ็นต์ไปที่เซิร์ฟเวอร์ฐานข้อมูลโดยตรงผ่านระบบเครือข่าย

- มีประสิทธิภาพในการทำงานดีที่สุด เมื่อเทียบกับทุกประเภทที่กล่าวมา
- มีความยุ่งยากในการพัฒนาโปรแกรมเพื่อทำงานในฝั่งไคลเอ็นต์ เพราะผู้พัฒนาต้องเรียนรู้การทำงานของเซิร์ฟเวอร์แต่ละตัวก่อน

บทที่ 7
หลักการทำงานเบื้องต้นของ Oracle Application Server

ในบทนี้จะขอกล่าวถึงส่วนประกอบของ OAS (Oracle Application Server) ของบริษัทออราเคิล ซึ่งทำหน้าที่เป็นแอพพลิเคชันเซิร์ฟเวอร์ และหน้าที่การทำงานของ Oracle Application Server Manager และ Oracle Application Server Utilities

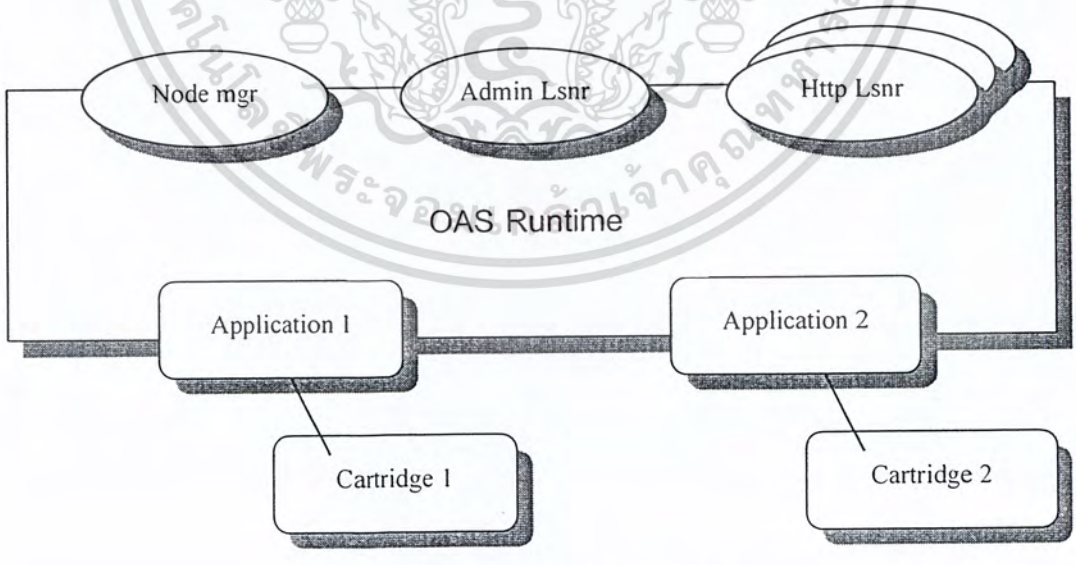
7.1 ส่วนประกอบของ OAS

แนวความคิดของเว็บไซต์คือ กลุ่มของเครื่องคอมพิวเตอร์ที่นำมาเชื่อมต่อกันและมีการใช้งาน Web Request Broker (WRB) ร่วมกัน โดย Web Request Broker คือโพรเซสของออราเคิลแอพพลิเคชันเซิร์ฟเวอร์ที่ทำงานเพื่อทำการจัดการกับทรัพยากรของระบบ

ในแต่ละโหนดในเว็บไซต์นั้นประกอบด้วยส่วนต่างๆดังนี้

- listeners ทำหน้าที่รับการร้องขอที่มาจากไคลเอ็นต์
- runtime processes เป็นส่วนที่กำหนดค่าเริ่มต้นที่จำเป็นสำหรับการร้องขอ
- Object Request Broker (ORB) ทำหน้าที่ในการจัดการกับการเชื่อมต่อระหว่างโหนดต่างๆ
- แอพพลิเคชันมีหน้าที่เตรียมส่วนประกอบในการทำงานให้กับ Cartridge เพื่อที่จะนำไปประมวลผล
- Cartridge เป็นโปรแกรมที่จะนำไปประมวลผล

ส่วนประกอบต่างที่ได้กล่าวมานี้สามารถทำงานอยู่ในเครื่องคอมพิวเตอร์เพียงเครื่องเดียวหรือกระจายไปอยู่บนเครื่องต่างๆ ซึ่งอยู่ที่การกำหนดองค์ประกอบการทำงานให้ระบบ



รูปที่ 7-1 ส่วนประกอบในเว็บไซต์ของ OAS

HTTP Listener เป็นส่วนประกอบภายในแอพพลิเคชันเซิร์ฟเวอร์ที่ทำหน้าที่รับการร้องขอการใช้งานแอพพลิเคชัน เมื่อได้รับการร้องขอ HTTP Listener จะทำการส่งการร้องขอไปให้กับโพรเซสที่ทำงานอยู่ ขั้นตอนต่อไปโพรเซสนั้นจะทำการกำหนดค่าเริ่มต้นให้กับแพพลิเคชันและอาจมีการสร้าง Cartridge ขึ้นมา จากนั้นแอพพลิเคชันจะถูกนำไปประมวลผลซึ่งได้มีการเตรียมส่วนประกอบในการทำงานให้กับ Cartridge

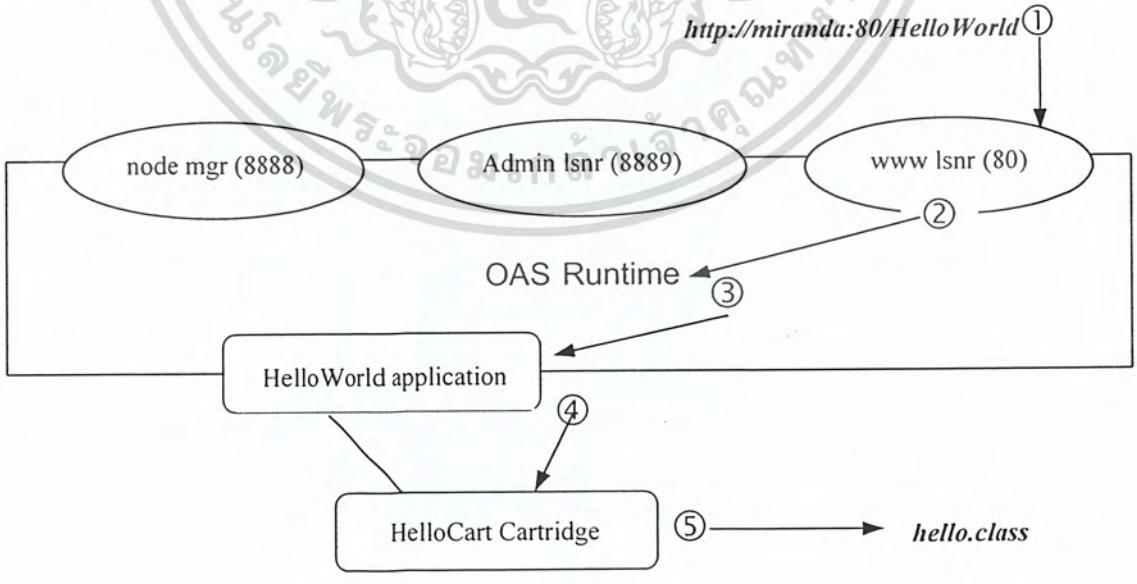
- ตัวอย่างของการร้องขอใช้บริการ

ในรูปที่ 7-2 ได้นำเสนอลำดับเส้นทางการร้องขอใช้งานแอพพลิเคชัน ในตัวอย่างนี้เว็บไซต์นั้นทำงานอยู่บนเครื่องที่มีชื่อว่า miranda และได้มีการใช้หมายเลขพอร์ตปกติสำหรับตัว listener ดังได้แสดงในตารางที่ 7-1

Listener	Port
Node manager	8888
Admin	8889
www	80

ตารางที่ 7-1 หมายเลขพอร์ตปกติที่ใช้ใน OAS

ในเว็บไซต์นี้ได้มีแอพพลิเคชันที่ชื่อว่า Hello World ซึ่งทำการประมวลผล Cartridge ที่เป็น Jservlet (เป็นชื่อที่ทางออราเคิลเรียก Cartridge Servlet) ที่มีชื่อว่า HelloCart และภายใน Cartridge นี้ได้มีการบรรจุคลาสไฟล์จาวา (Servlet) ที่มีชื่อว่า hello.class



รูปที่ 7-2 ตัวอย่างเส้นทางการร้องขอ

จากรูปที่ 7-2 ได้แสดงตัวอย่างการร้องขอโดยมีลำดับดังต่อไปนี้

1. ไคลเอ็นต์ได้เรียกใช้บริการทาง http://miranda:80/HelloWorld และ www Listener มีหน้าที่รับการร้องขอผ่านทางพอร์ตหมายเลข 80
2. Listener จะส่งค่าการร้องขอไปให้กับโพรเซส runtime และโพรเซสจะทำหน้าที่หาว่าใน HelloWorld แอพพลิเคชันมี Cartridge ที่มีชื่อว่า HelloCart จากนั้นโพรเซสจะหาว่ามีออบเจ็กต์ของ HelloCart ที่ว่างอยู่หรือไม่ ถ้าไม่พบก็จะทำการสร้างออบเจ็กต์นั้นขึ้นมา
3. OAS จะเริ่มต้นการทำงานให้กับแอพพลิเคชัน HelloWorld โดยแอพพลิเคชันจะต้องเตรียมส่วนประกอบในการทำงานให้กับ Cartridge ที่มีชื่อว่า HelloCart เนื่องจาก HelloCart นี้เป็น Cartridge ชนิด Jservlet ดังนั้นจึงต้องทำการเริ่มต้นการทำงานให้กับจาวาเวอร์ชวลแมชชีน
4. ออบเจ็กต์ของ HelloCart ถูกประมวลผลด้วยแอพพลิเคชัน HelloWorld
5. ออบเจ็กต์ HelloCart ประมวลผลคำสั่งภายในโปรแกรมภายในคลาสที่ชื่อ hello.class และทำการคืนค่าผลลัพธ์ผ่านตัวโพรเซส runtime และ listener ให้กับผู้ใช้งาน

7.2 หลักการเบื้องต้นของแอพพลิเคชันในมุมมองของ OAS

การทำงานหลักของ OAS คือให้บริการแอพพลิเคชันที่ทำงานอยู่ในฝั่งเซิร์ฟเวอร์ ซึ่งเราสามารถทำการพัฒนาแอพพลิเคชันด้วยการเขียนโปรแกรมด้วยภาษาต่างๆ และนอกจากนั้นเราสามารถเลือกโพรโตคอลที่ใช้ในการติดต่อสื่อสาร ระหว่างไคลเอ็นต์กับแอพพลิเคชันที่เราได้ทำการสร้างขึ้นมาซึ่งประกอบไปด้วย HTTP หรือ CORBA/IIOP ในลักษณะของไคลเอ็นต์ของแอพพลิเคชันนั้นอาจจะเป็นเว็บบราวเซอร์ เช่น Netscape Navigator หรือ Microsoft Internet Explorer หรือจาวาแอพเพล็ตที่ทำงานอยู่ในเว็บบราวเซอร์หรือแม้กระทั่งแอพพลิเคชันที่ทำงานด้วยตัวเอง (Stand alone Application) ในหัวข้อนี้จะได้กล่าวถึงหลักการเบื้องต้นของแอพพลิเคชันและ Cartridge รูปแบบการให้บริการของแอพพลิเคชันทั่วไป

7.2.1 Cartridge และแอพพลิเคชัน

ในการพัฒนาแอพพลิเคชันในฝั่งเซิร์ฟเวอร์ของออราเคิลเว็บเซิร์ฟเวอร์นั้น สามารถแบ่งออกได้เป็น 2 ลักษณะคือ

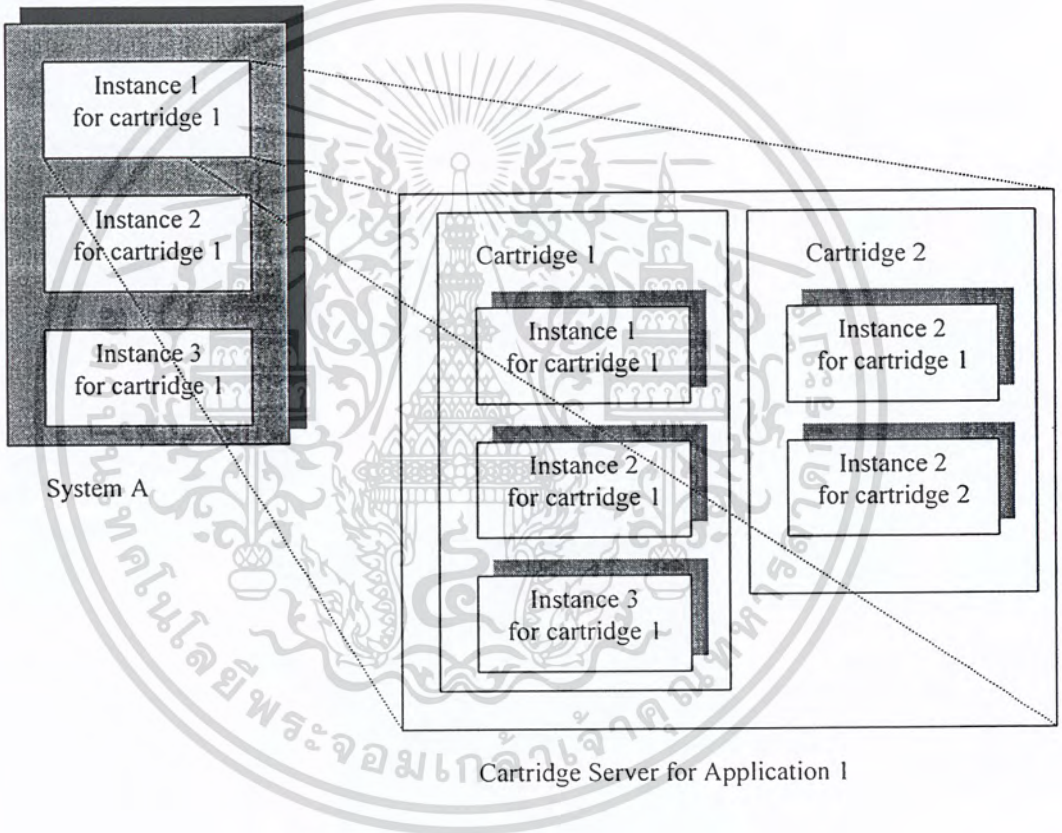
- แอพพลิเคชันที่ทำงานด้วย Cartridge (Cartridge – Based Applications)
- แอพพลิเคชันที่ทำงานด้วย IIOP (IIOP Based Applications)
- แอพพลิเคชัน ที่ทำงานด้วย Cartridge

OAS นั้นประกอบไปด้วย Cartridge เซิร์ฟเวอร์ที่สามารถให้บริการแอพพลิเคชันที่ทำงานฝั่งเซิร์ฟเวอร์ โดย Cartridge เซิร์ฟเวอร์นั้นก็คือ โพรเซสที่ให้ Cartridge ที่สร้างขึ้นมาทำการประมวลผลกลุ่มคำสั่งและ Instance Cartridge ประกอบไปด้วยกลุ่มของคำสั่งและข้อมูลที่นำไปใช้ในการกำหนดพื้นที่ให้กับ Cartridge และการควบคุม

เมื่อเราสร้างแอพพลิเคชันเราสามารถที่จะกำหนดองค์ประกอบโดยใช้เครื่องมือที่มีชื่อว่า ออราเคิลแอพพลิเคชันเซิร์ฟเวอร์แมเนเจอร์ (Oracle Application Server Manager) โดยเราสามารถที่จะกำหนด

องค์ประกอบในสองระดับคือ ในระดับของแอพพลิเคชันเป็นการกำหนดค่าพื้นฐานให้กับกลุ่มของ Cartridge ที่เป็นประเภทเดียวกันในแต่ละแอพพลิเคชัน (แสดงในตารางที่ 7-2) และในระดับของ Cartridge เป็นการกำหนดองค์ประกอบเฉพาะสำหรับแต่ละ Cartridge ซึ่งจะนำไปใช้กับ Instance Cartridge ที่สร้างมาจาก Cartridge ชนิดนั้น

ตัวอย่างในรูปที่ 7-3 ในระบบ A ประกอบไปด้วย Cartridge เซิร์ฟเวอร์ 3 ตัวซึ่งเป็นของแอพพลิเคชัน 1 สองตัวและสำหรับแอพพลิเคชัน 2 หนึ่งตัว Cartridge เซิร์ฟเวอร์คือโพรเซสที่ถูกควบคุมด้วยแอพพลิเคชันเซิร์ฟเวอร์ที่ทำงานอยู่ในระบบ A เมื่อพิจารณาที่ Cartridge เซิร์ฟเวอร์ที่อยู่ในแอพพลิเคชัน 1 นั้นมีอยู่สอง Cartridge ที่ทำงานอยู่ ใน Cartridge แรกมีอินสแทนซ์ ของ Cartridge1 สามอินสแทนซ์และอินสแทนซ์ของ Cartridge2 สองอินสแทนซ์



รูปที่ 7-3 ตัวอย่างแอพพลิเคชันและ Cartridge

ในตารางที่ 7-2 ได้นำเสนอ Cartridge ที่ OAS ได้เตรียมให้ โดยแอพพลิเคชันที่ทำงานด้วย Cartridge นั้นจะใช้โพรโตคอล HTTP ในการติดต่อสื่อสารเพื่อรับการร้องขอจากไคลเอ็นต์

ชนิดของ Cartridge	ลักษณะการทำงานของโปรแกรม
C Cartridge	มีการใช้ Library (UNIX) หรือ DLL (NT) ร่วมกัน
JServlet, Jweb Cartridge	Java Class file
JSP Cartridge	มีการนำ directive และ tag ไปรวมในไฟล์ JSP
LiveHTML Cartridge	มีการนำ Script อยู่ในไฟล์ HTML
Perl Cartridge	Perl Scripts
PL/SQL	Stored Procedure ในฐานข้อมูลหรือไฟล์ SQL script

ตารางที่ 7-2 แสดงลักษณะต่างของ Cartridge

- **แอพพลิเคชันที่ทำงานด้วย IIOP**

ใน OAS สามารถให้เราทำการพัฒนาแอพพลิเคชันที่ใช้หลักการกระจายออบเจ็กต์ด้วยมาตรฐานของ CORBA ในการติดต่อสื่อสารด้วยโพรโตคอล IIOP และแอพพลิเคชันนั้นสามารถรองรับรูปแบบของแอพพลิเคชันได้ดังนี้

- แอพพลิเคชันในรูปแบบ ECO/Java
- แอพพลิเคชันในรูปแบบ Enterprise JavaBean (EJB)
- แอพพลิเคชันในรูปแบบ C++ CORBA Cartridge

ECO/Java และ EJB คือการสร้างออบเจ็กต์ของ CORBA ที่สามารถเข้าใช้งานได้จากไคลเอ็นต์ชนิดต่างๆ เช่น จาวาแอพเพล็ต, จาวาแอพพลิเคชันและ CORBA Application ส่วน C++, CORBA Cartridge นั้นเป็นการกำหนดสถาปัตยกรรมสำหรับสร้างแอพพลิเคชัน ในลักษณะของออบเจ็กต์แบบกระจายในทางธุรกิจ ด้วยโปรแกรมภาษา C++

7.3 แอพพลิเคชันในลักษณะ JServlet Cartridge

ภายใน JServlet Cartridge คือ จาวาเวอร์ชวลแมชีน และจาวาคลาส Library มีหน้าที่ในการเตรียมส่วนประกอบในการทำงานของแอพพลิเคชันที่เขียนด้วยข้อกำหนดทางภาษาจาวา Servlet และทำงานอยู่ในฝั่งเซิร์ฟเวอร์ ข้อดีของการใช้งาน JServlet Cartridge พอที่จะสรุปได้ดังนี้

- มีประสิทธิภาพที่ดีกว่าซีจีไอ (CGI) เพราะว่าไม่ต้องมีการเริ่มต้นและหยุดการทำงานของจาวาเวอร์ชวลแมชชีนเมื่อมีการร้องขอเข้ามาในแต่ละครั้ง ตัวอย่างเช่น การเชื่อมต่อกับฐานข้อมูลสามารถสร้างการเชื่อมต่อเพียงครั้งเดียวสำหรับในแต่ละการเชื่อมต่อ (session) ทำให้สามารถเชื่อมต่อฐานข้อมูลทุกครั้งที่มีการร้องขอจากไคลเอ็นต์
- สามารถใช้ข้อดีของออราเคิลแอพพลิเคชันเซิร์ฟเวอร์ในการจัดการด้านความสมดุลย์ การขยายขอบเขต, การตรวจสอบ, การติดตามการใช้งานและความสามารถในการบริหารการติดต่อ

- มีการใช้งานทรัพยากรของระบบน้อยเพราะมีการทำงานของ JServlet Cartridge หลายๆตัว สามารถทำงานอยู่บนจาวาเวอร์ชวลแมชชีนเดียวกันได้ และนอกจากนั้นอินสแทนซ์ที่ว่างยังนำกลับมาใช้ได้ใหม่อีกด้วย
- สามารถใช้ชุดเครื่องมือของทาง JServlet ซึ่งประกอบด้วยชุดของคลาสไฟล์ ที่สามารถสร้างเอกสาร HTML และเชื่อมต่อกับฐานข้อมูล

การใช้งาน JServlet นั้นจะทำงานอยู่ภายใต้ฝั่งเซิร์ฟเวอร์ที่แตกต่างกับแอพเพล็ตที่ต้องโหลดและไปทำงานอยู่ในฝั่งไคลเอ็นต์

7.4 Oracle9iAS Portal

Oracle9iAS Portal จัดเป็น software framework สำหรับสร้าง, จัดการและบำรุงรักษา e-business portal web sites ที่มีลักษณะส่วนบุคคล (Personalized web sites) ด้วยเครื่องมือต่างๆที่ใช้งานง่าย อีกทั้งสามารถจัดการกับข้อมูลแบบไดนามิกได้ และเลือกเปิดเผยข้อมูลทางธุรกิจบนเว็บเพจแตกต่างกันตามลักษณะของผู้เข้าเยี่ยมชมได้อีกด้วย

7.4.1 ทำไมต้องใช้พอร์ทอล (Portal) ?

ผลของการเติบโตของเทคโนโลยีด้านเว็บและอินเตอร์เน็ตอย่างรวดเร็ว ทำให้องค์กรธุรกิจหลายแห่งต้องหันมาให้ความสำคัญกับข้อมูลข่าวสารที่จะเผยแพร่ทางอินเตอร์เน็ต และบางแห่งนำช่องทางนี้มาสร้างโอกาสที่ดีให้กับธุรกิจของตน

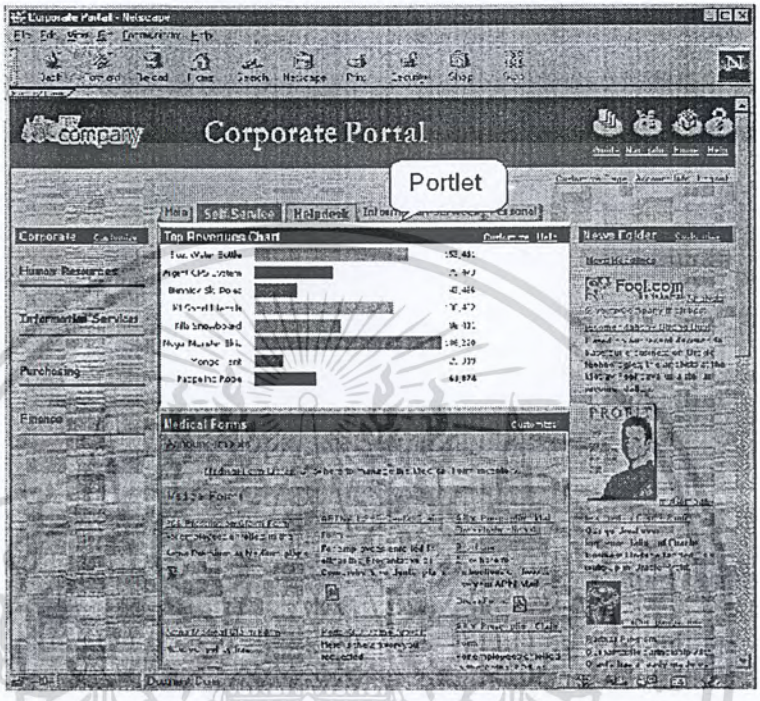
แต่กลับเกิดปัญหาบางอย่างตามมาต่อองค์กรธุรกิจนั้น เพราะข้อมูลขององค์กรที่แสดงบนเว็บเพจไม่สามารถจำแนกข้อมูลตามลักษณะตำแหน่งและบทบาทของผู้เข้าเยี่ยมชมบนเครือข่ายทั้งแบบ Intranet และ Internet ได้ ส่งผลให้เกิดปัญหา Information Overload, ปัญหาความซ้ำซ้อนในการจัดการระบบ และข้อมูลสับสนผิดพลาดตามมา

ดังนั้นองค์กรต่างๆจึงหันมาใช้รูปแบบเว็บเพจที่นำเสนอผ่านพอร์ทอลไซต์ ถือเป็นกุญแจหนึ่งในกลยุทธ์ทางธุรกิจ ซึ่งโดยความเป็นจริงแล้ว e-business portal web sites เป็นการนำข้อมูลทั้งหมดขององค์กรมารวมไว้ที่แหล่งเดียว เพื่อให้เข้าถึงบริการต่างๆได้ง่าย และพอร์ทอลได้เพิ่มคุณสมบัติที่โดดเด่นเข้าด้วยกัน ได้แก่ personalized (การจัดหน้าตาแสดงผลเว็บเพจตามที่ผู้ใช้แต่ละคนต้องการ), Job-based Information (การแสดงข้อมูลข่าวสารทางเว็บเพจตามตำแหน่งหน้าที่ของผู้ใช้ เช่น ผู้จัดการบริษัทจะเห็นข่าวสารแตกต่างจากลูกจ้าง หรือลูกค้า เป็นต้น)

7.4.2 คุณสมบัติของ Oracle9iAS Portal

Oracle9iAS Portal เป็น browser-base software framework สำหรับพัฒนา, บริหารจัดการและแก้ไขดัดแปลง e-business portals ที่มีคุณสมบัติต่างๆที่ช่วยให้การสร้าง e-business portals ยุ่งยากซับซ้อนน้อยลง, ปลอดภัยมากขึ้น, เชื่อถือได้ และรองรับการขยายตัวทางธุรกิจได้ดี โดยมีคุณสมบัติสำคัญได้แก่

- ความสามารถในการจัดการข้อมูลข่าวสารได้อย่างถูกต้องและเหมาะสมกับบทบาทหน้าที่ของผู้เข้าเยี่ยมชมเว็บเพจแต่ละคน เช่นในส่วนแสดงเนื้อหาในเว็บเพจ หรือที่เรียกว่า พอร์ตเล็ต (portlet) ดังรูป 7.4 เราสามารถกำหนดให้แสดงรายงานสรุปผลประกอบการประจำปีสำหรับผู้จัดการ ส่วนลูกจ้างอาจเห็นข้อมูลการทำงานประวัติการขาดลามาสายของตนเอง ในขณะที่ถ้าเป็นลูกค้าของบริษัท ข้อมูลผลิตภัณฑ์และราคาจะนำมาแสดงผลแทน



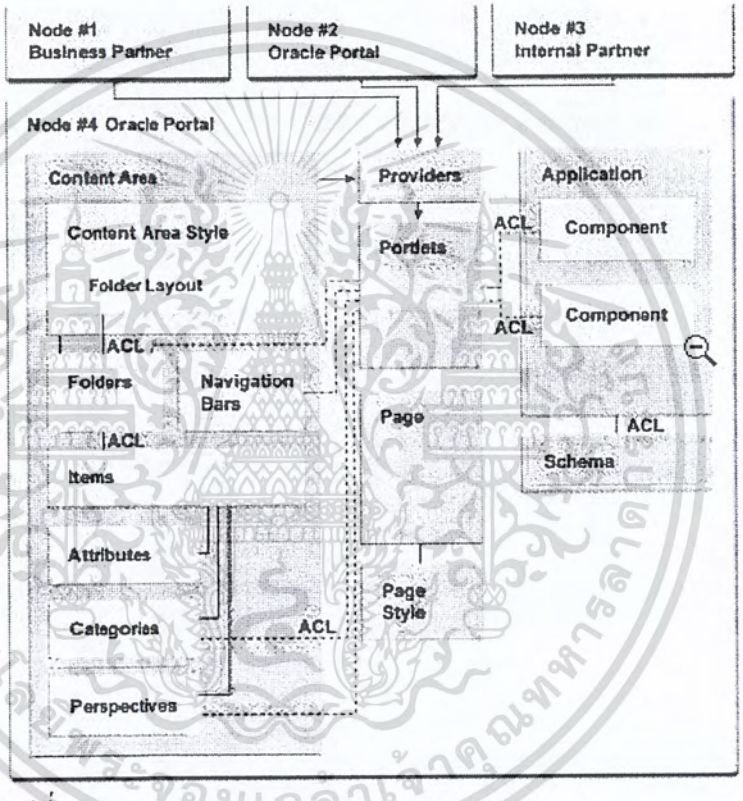
รูปที่ 7-4 แสดงลักษณะ portlet บนเว็บเพจ

- Single-sign on ที่เอื้อประโยชน์ต่อผู้ใช้ในการ log in เพียงครั้งเดียว แล้วสามารถเข้าใช้งาน application ต่างๆได้ โดยไม่ต้อง log in อีก ยกตัวอย่างเช่น เมื่อเรา log in portal แล้ว เราจะเข้าสู่หน้าเว็บเพจส่วนตัว (personalized) และในหน้าเพจส่วนตัวนั้น อาจมี link ให้เข้าไปขอใช้บริการ mail ของ Hotmail.com, Yahoo.com ฯลฯ โดยเราไม่จำเป็นต้อง log in อีกต่อไป
- สามารถดัดแปลงแก้ไขหน้าตาเว็บเพจได้ตามต้องการ เช่น สามารถแก้ไขสี, background, รูปแบบตัวอักษรได้ อีกทั้งยังสามารถแก้ไขเนื้อหาของพอร์ตเล็ตได้ตามสิทธิที่ผู้ใช้แต่ละคนได้รับ โดยผู้ที่มีหน้าที่ดูแลและกำหนดสิทธิต่างๆของผู้ใช้ เรียกว่า Portal Administrator
- สามารถนำข้อมูลจาก Oracle Database มาใช้แสดงผลในรูปแบบต่างๆ ไม่ว่าจะเป็น HTML bar charts, report, MS Excel format ฯลฯ ผ่านทาง Oracle9iAS Portal ได้อย่างง่ายดายด้วย wizard

7.4.3 Oracle Portal ทำงานอย่างไร

Oracle Portal ช่วยจัดการการเข้าถึง และโต้ตอบข้อมูล(interact) โดยมีองค์ประกอบที่สำคัญ 3ส่วน คือ

- content areas เป็นส่วนโครงสร้างสำหรับจัดการ, แบ่งแยกประเภทสำหรับองค์ประกอบย่อย (item) ในเว็บไซต์
- applications เป็นส่วนสำหรับเพิ่มเติม, แก้ไขปรับปรุง, และควบคุมการแสดงผลข้อมูลของ Oracle8I Database
- pages เป็นส่วนจัดการข้อมูลจาก content areas, application รวมถึงแหล่งข้อมูลจากภายนอกให้สามารถเข้าถึงได้ภายในเว็บเพจ



รูปที่ 7-5 แสดง Content Areas, Pages และ Applications ที่เป็นองค์ประกอบหลักๆของ Oracle Portal

จากรูปที่ 7-5 pages, portlets และ portlet providers จัดเป็นส่วนที่สำคัญที่สุดในการสร้างพอร์ทอลไซต์ เพราะถ้ามี content area หรือ application แต่ขาด pages ก็ไม่สามารถแสดงผลให้เห็นเป็นพอร์ทอลเว็บไซต์ได้

พอร์ทอลจึงเป็น Web-based application ที่ทำให้ผู้ใช้เข้าถึง content area, external web sites, แอพพลิเคชันอื่น และข่าวสารที่เป็นประโยชน์ได้ โดยผ่านทาง page นั่นเอง

บทที่ 8
การออกแบบระบบบริหารจัดการพอร์ทอลไซต์

8.1 ประโยคปัญหา

ระบบบริหารจัดการพอร์ทอลไซต์เป็นโปรแกรมที่ช่วยให้ผู้ดูแลระบบสามารถบริหารพอร์ทอลไซต์ได้ง่ายขึ้น โดยตัวโปรแกรมช่วยจัดการกับข้อมูลที่ผู้ใช้ป้อนเข้ามา และรวบรวมข้อมูลสถิติต่างๆ มานำเสนอให้ผู้ใช้หรือผู้ดูแลระบบ ฟังก์ชันการทำงานของโปรแกรมประกอบไปด้วย ระบบจัดการกับข้อมูลต่างๆ ที่ผู้ใช้ป้อนเข้ามาในระบบ และฟังก์ชันการรวบรวมสถิติต่างๆ มาแสดงให้ผู้ใช้หรือผู้ดูแลระบบทราบโดยนำเสนอในรูปของกราฟวงกลม กราฟแท่งและกราฟเส้น โดยสามารถกำหนดเงื่อนไขในการแสดงข้อมูลเพื่อให้ได้ข้อมูลที่ตรงกับความต้องการมากที่สุด

การเรียกใช้ฟังก์ชันของระบบบริหารจัดการพอร์ทอลไซต์นั้น จะต้องเรียกใช้ผ่านเว็บบราวเซอร์เท่านั้น สำหรับบุคคลทั่วไปจะสามารถทำได้เพียงเข้าชมเว็บเพจเท่านั้น หากต้องการใช้บริการของระบบจะต้องทำการลงทะเบียนขอสิทธิใช้งานระบบ ผู้ดูแลระบบก็จะทำการเพิ่มข้อมูลผู้ใช้เข้าไปในรายชื่อผู้ใช้งานระบบ และกำหนดสิทธิการใช้งานให้ เมื่อมีชื่อในรายชื่อผู้ใช้แล้ว ผู้ใช้ก็สามารถล็อกอิน (Log in) เข้าในระบบเพื่อเรียกใช้บริการต่างๆ ที่ระบบจัดไว้ให้ได้ เช่น บริการสร้างเว็บเพจ บริการเปลี่ยนแปลงข้อมูลของตัวผู้ใช้ และบริการเรียกดูสถิติการเข้าชมเว็บเพจที่ผู้ใช้สร้างขึ้นมาเป็นต้น เมื่อต้องการออกจากระบบก็จะต้องทำการล็อกเอาท์ (Log out) ออกจากระบบ ซึ่งกิจกรรมต่างๆ ที่ผู้ใช้กระทำจะถูกบันทึกไว้ในฐานข้อมูลเพื่อใช้เป็นข้อมูลสถิติต่อไป

สำหรับผู้ดูแลระบบ นอกจากจะสามารถทำกิจกรรมต่างๆ ที่ผู้ใช้สามารถกระทำได้แล้ว ยังสามารถเพิ่มผู้ใช้เข้าไปในรายชื่อผู้ใช้งานระบบ กำหนดสิทธิการใช้งานให้กับผู้ใช้ และสามารถเรียกดูข้อมูลสถิติต่างๆ ที่เก็บไว้ในระบบ เช่น ข้อมูลการล็อกอินเข้าสู่ระบบของผู้ใช้ ข้อมูลการเข้าชมเว็บเพจของผู้ใช้ และข้อมูลกิจกรรมต่างๆ ของผู้ใช้เป็นต้น โดยข้อมูลสถิติต่างๆ จะถูกนำเสนอในรูปกราฟเพื่อให้เข้าใจได้ง่าย

8.2 Use Case Model

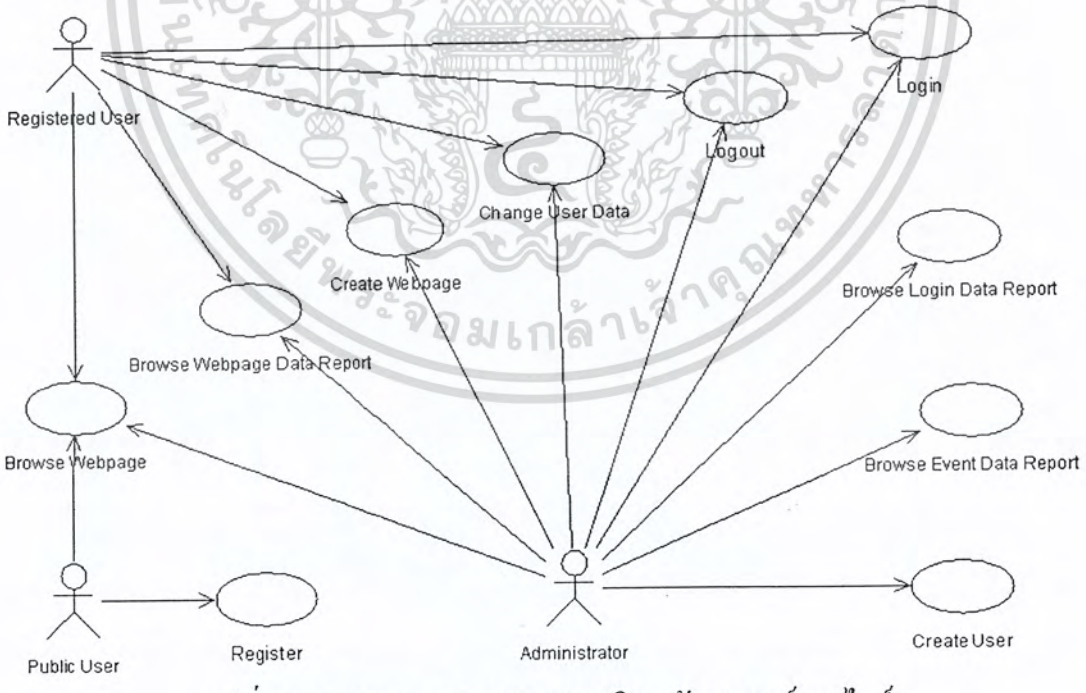
จากประโยคปัญหาข้างต้น เราสามารถนำมาเขียน Use Case Model เพื่อช่วยให้สามารถทำความเข้าใจกับปัญหา และช่วยในการออกแบบระบบได้ดังนี้

8.2.1 กำหนดผู้กระทำ (Actor)

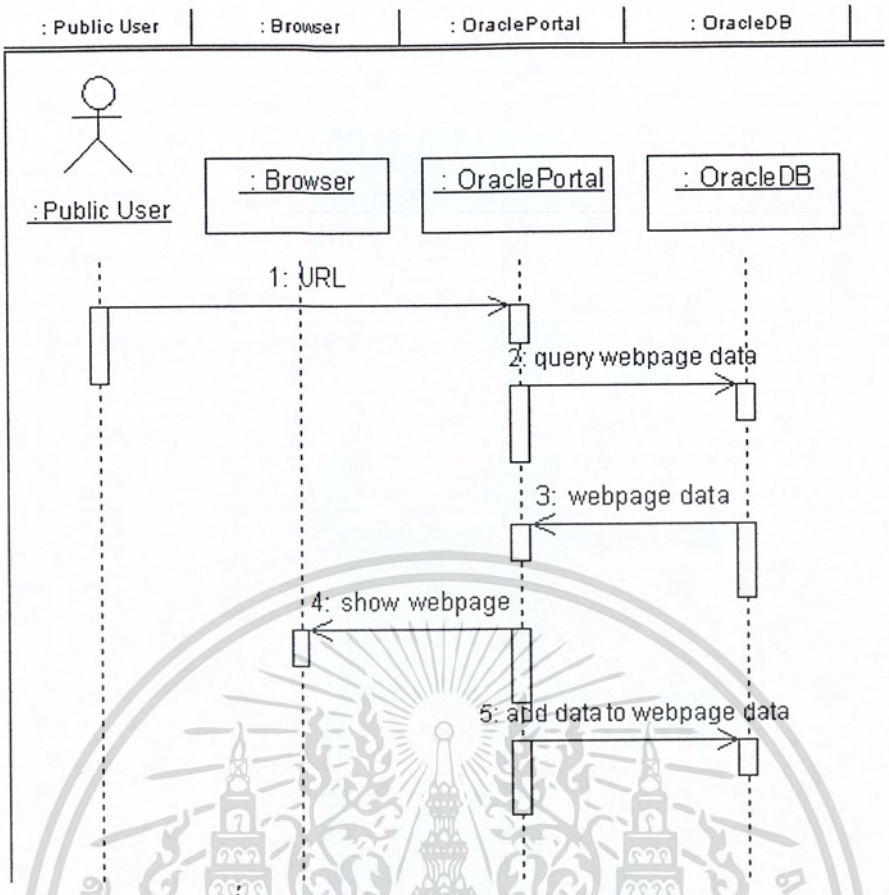
ผู้กระทำของระบบนี้ คือ บุคคลทั่วไป ผู้ใช้ และผู้ดูแลระบบ

8.2.2 กำหนด Use Case

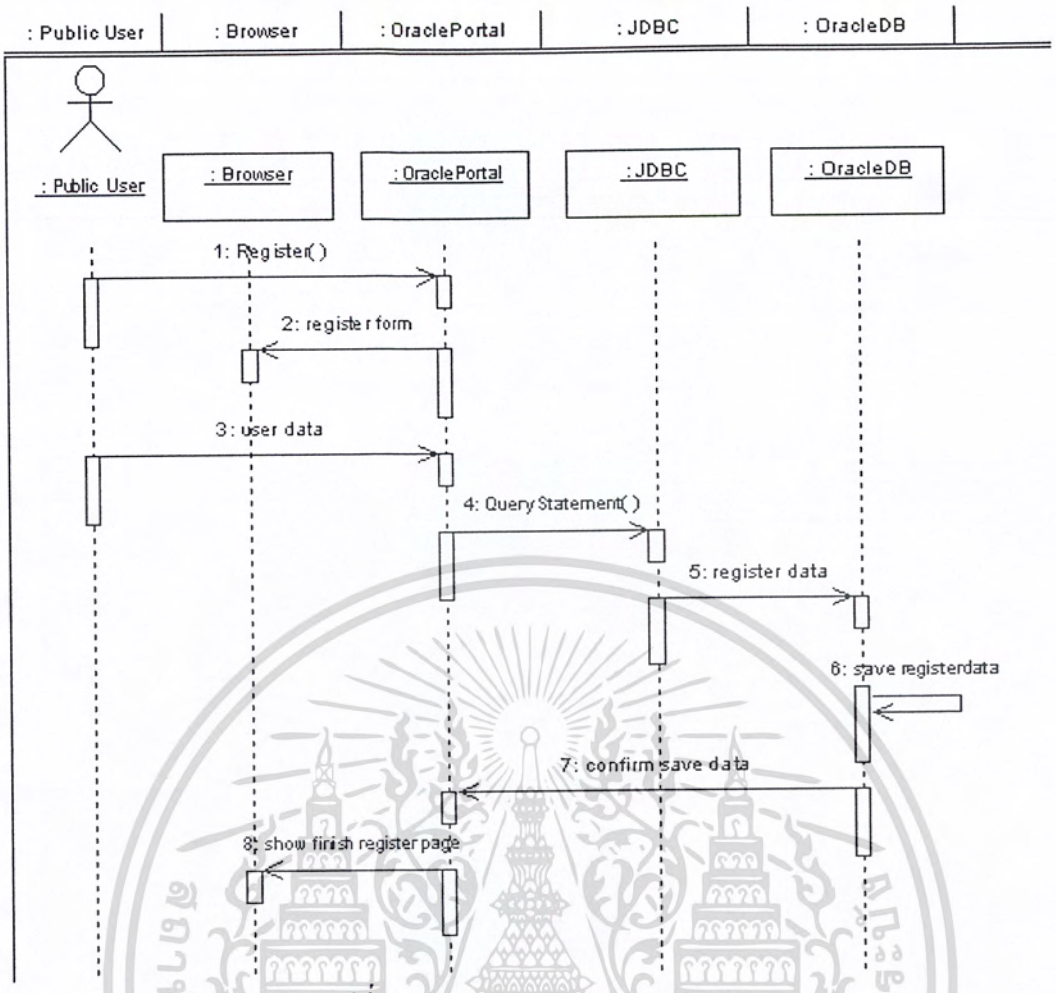
- **Browse Webpage** ทุกคนสามารถเข้าชมเว็บเพจที่ถูกสร้างขึ้นมาได้
- **Register** บุคคลทั่วไปจะต้องมาลงทะเบียนขอสิทธิการใช้งานระบบก่อน จึงจะสามารถเข้าใช้บริการของระบบได้
- **Create User** ผู้ดูแลระบบสามารถเพิ่มข้อมูลผู้ใช้เข้าไปในรายชื่อผู้ใช้ และกำหนดสิทธิการเข้าใช้บริการของระบบให้แก่ผู้ใช้ได้
- **Log in** ผู้ใช้ที่ลงทะเบียนแล้ว และผู้ดูแลระบบจะต้องล๊อกอินเข้าสู่ระบบก่อน จึงจะสามารถเรียกใช้บริการของระบบได้
- **Change User Data** ผู้ใช้บริการระบบสามารถเปลี่ยนแปลงข้อมูลของตัวเองได้
- **Create Webpage** ผู้ใช้บริการระบบสามารถสร้างเว็บเพจได้
- **Browse Webpage Data Report** ผู้ใช้บริการระบบสามารถเรียกดูสถิติการเข้าชมเว็บเพจที่ตัวเองสร้างขึ้นมาได้
- **Log out** ผู้ใช้บริการระบบจะต้องล๊อกเอาท์ออกจากระบบ เมื่อเลิกใช้งานระบบ
- **Browse Login Data Report** ผู้ดูแลระบบสามารถเรียกดูข้อมูลสถิติการล๊อกอินเข้าสู่ระบบของผู้ใช้ได้
- **Browse Event Data Report** ผู้ดูแลระบบสามารถเรียกดูข้อมูลสถิติกิจกรรมต่าง ๆ ที่ผู้ใช้กระทำได้



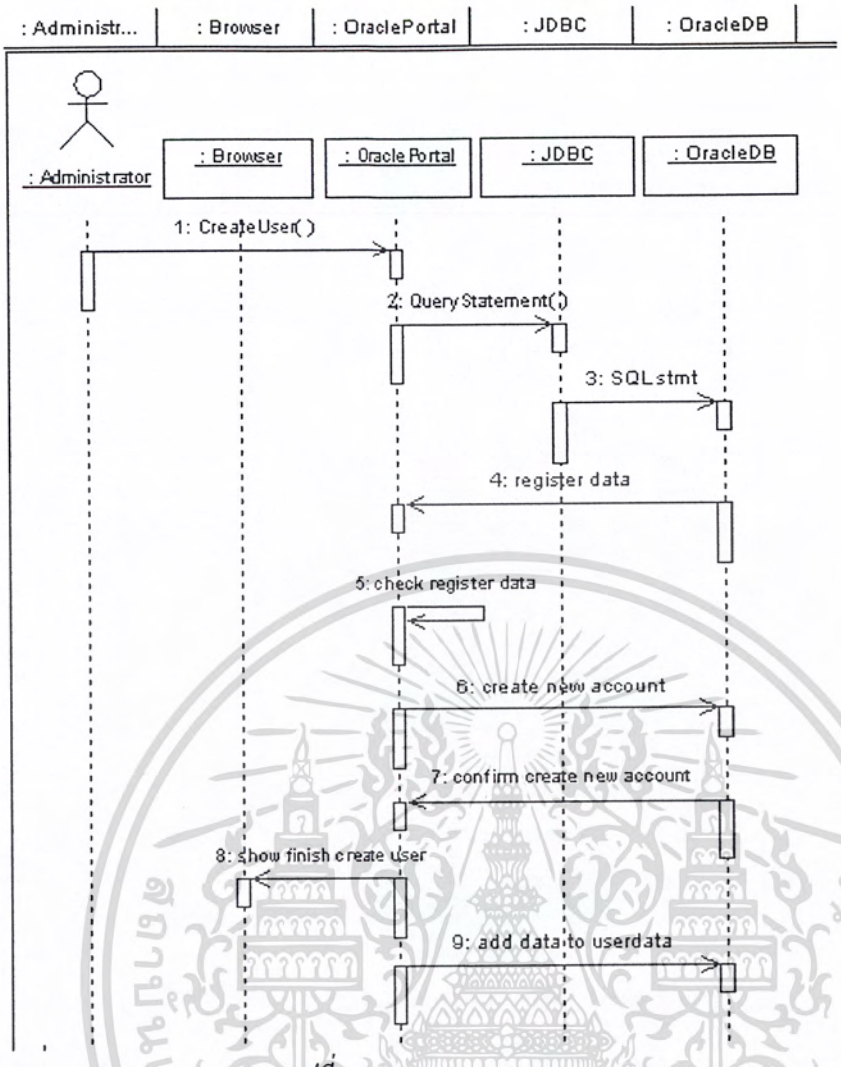
รูปที่ 8-1 Use Case Model ของระบบบริหารจัดการพอร์ทอลไซต์



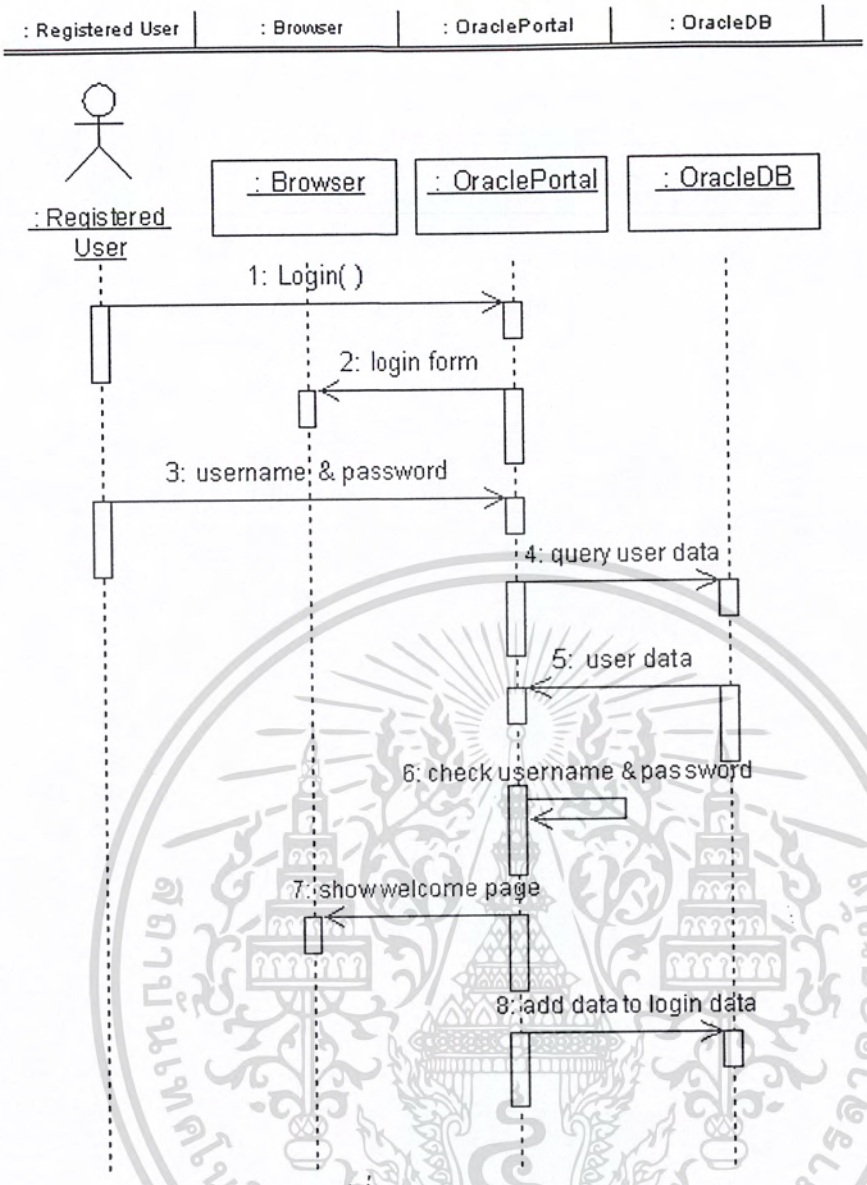
รูปที่ 8-2 Sequence Diagram : Browse Webpage



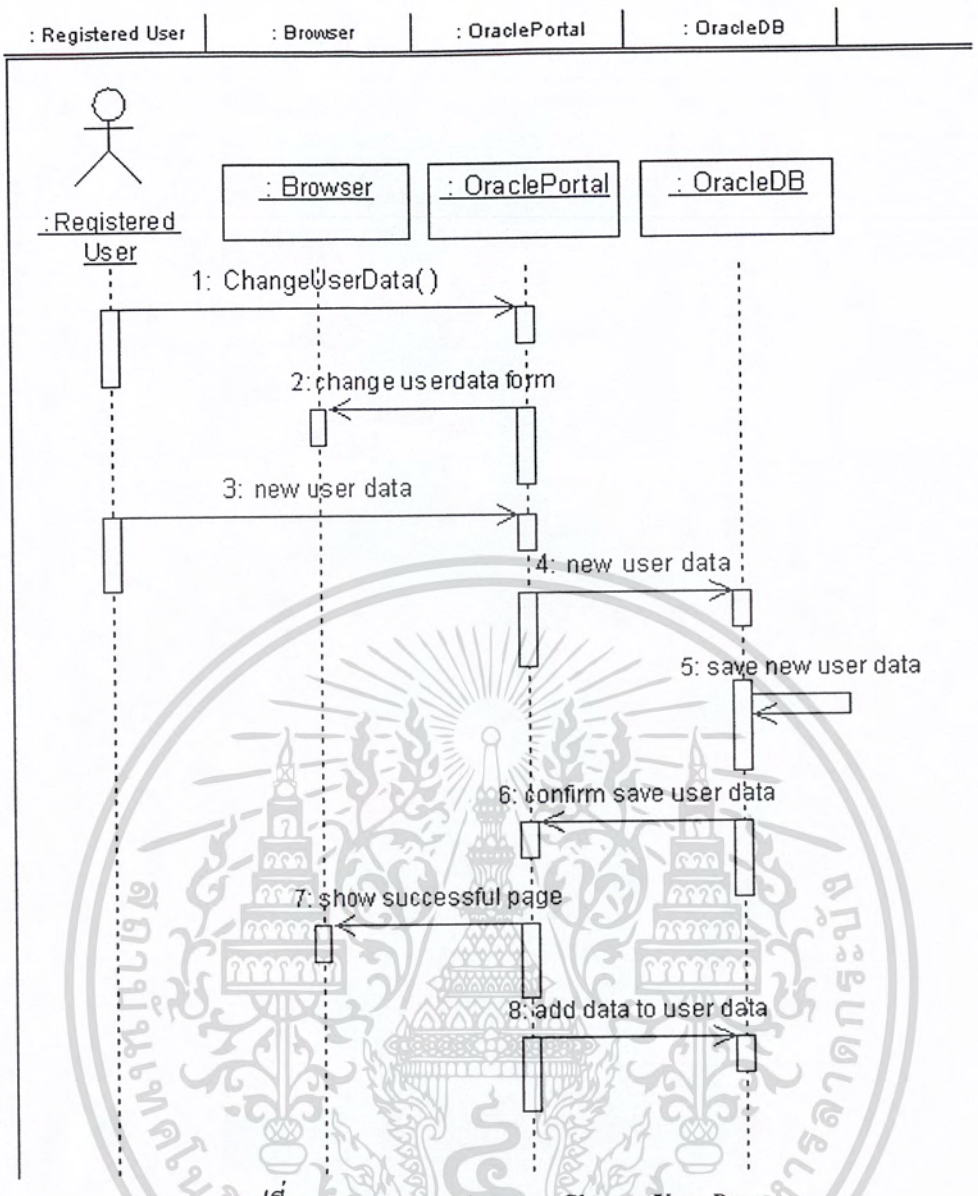
รูปที่ 8-3 Sequence Diagram : Register



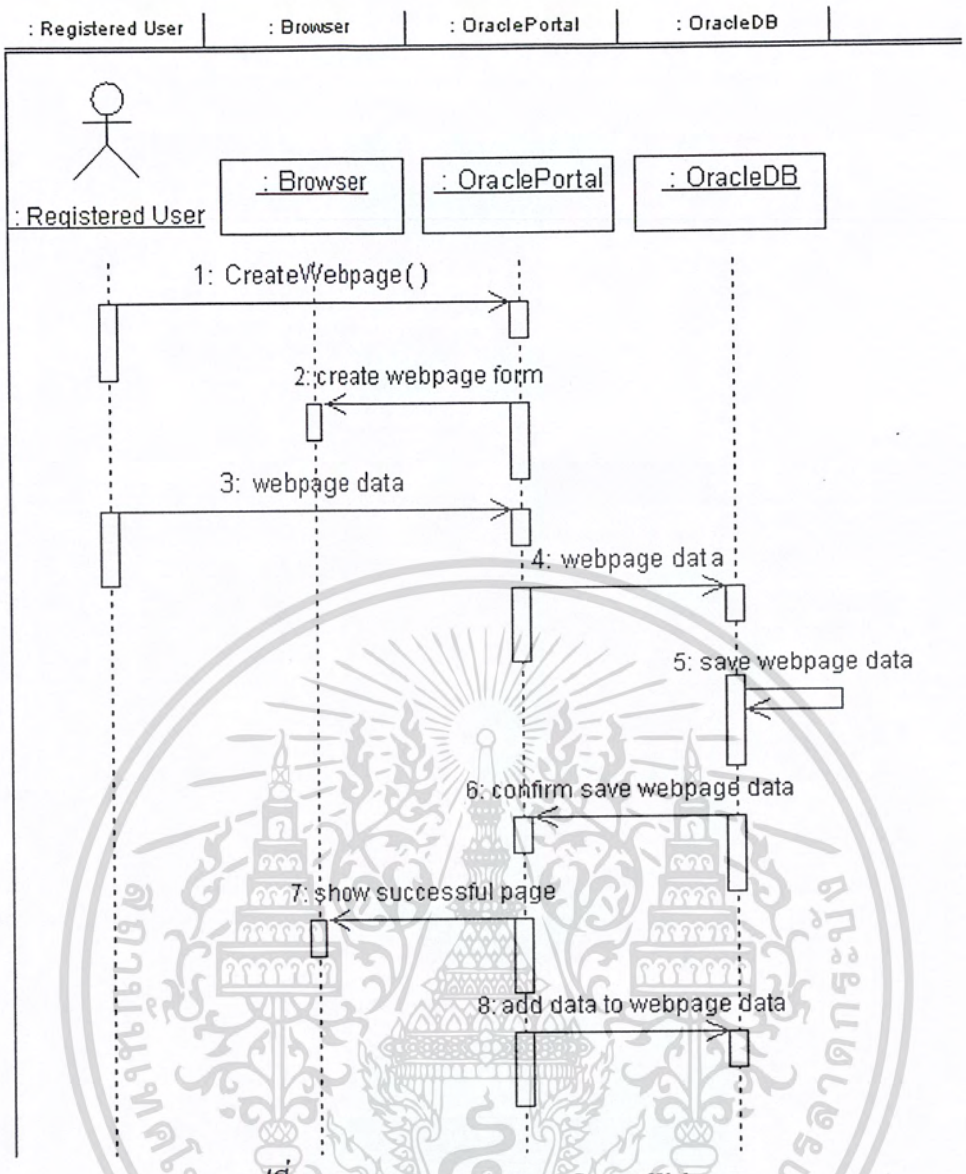
รูปที่ 8-4 Sequence Diagram : Create User



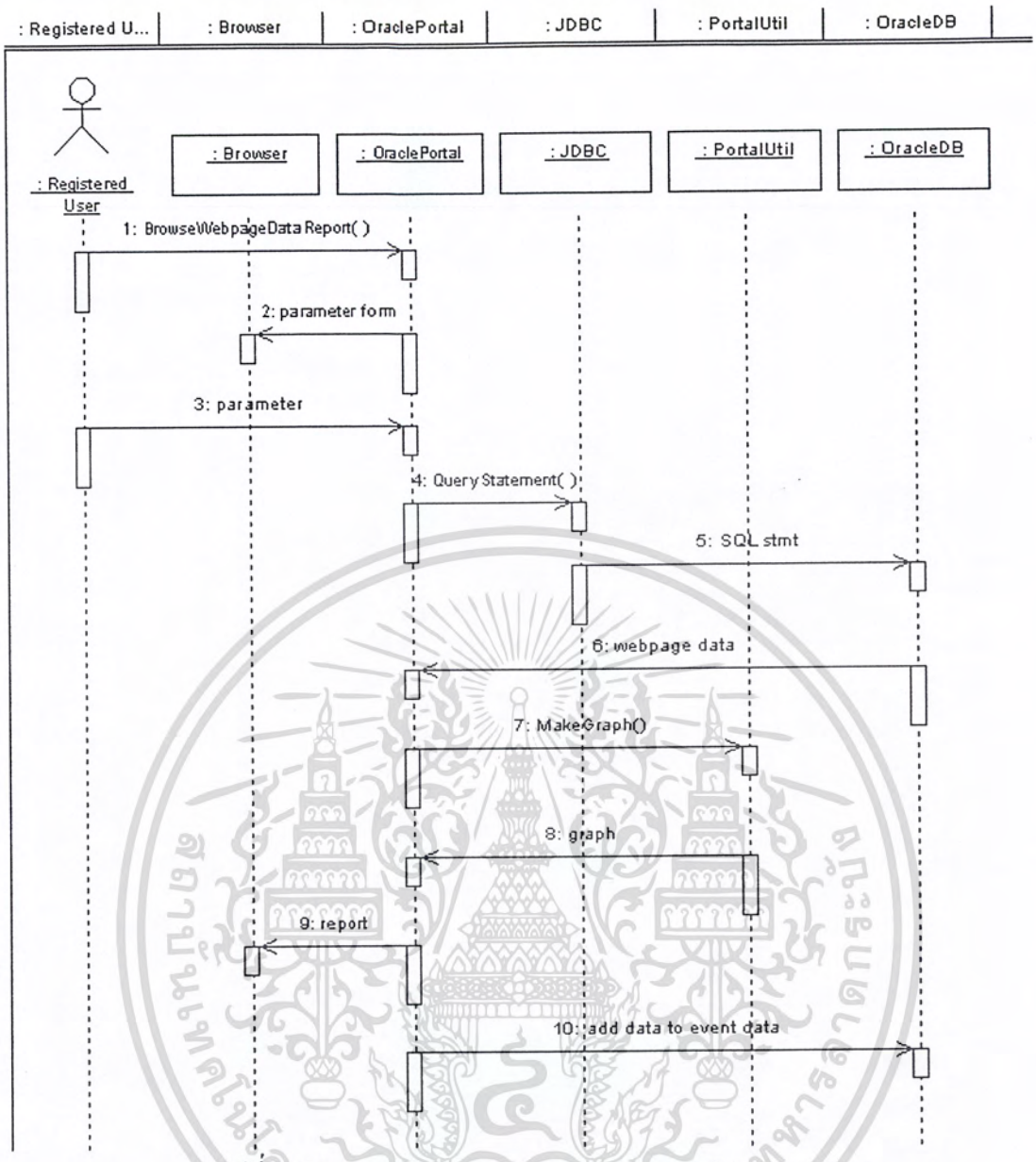
รูปที่ 8-5 Sequence Diagram : Log in



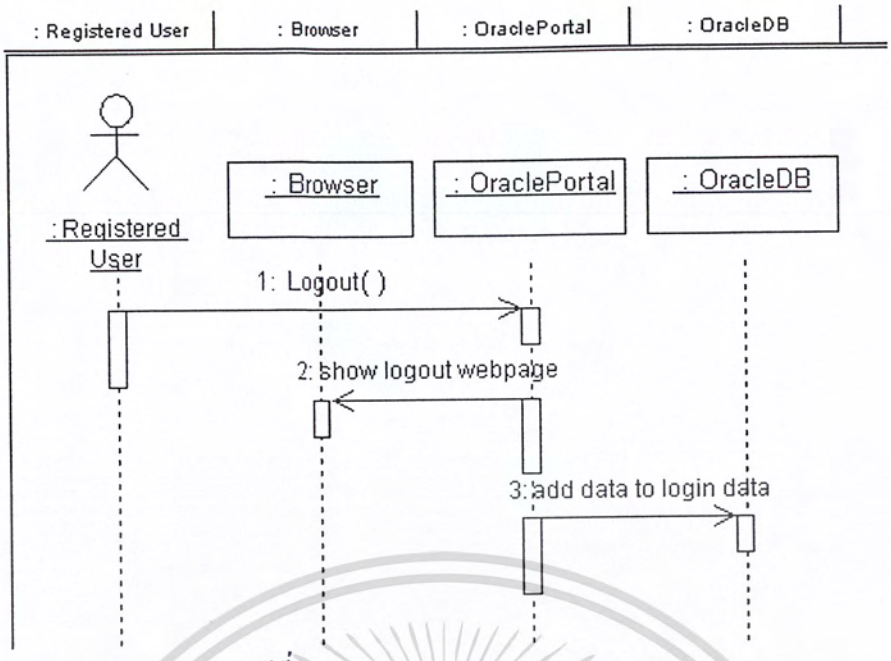
รูปที่ 8-6 Sequence Diagram :Change User Data



รูปที่ 8-7 Sequence Diagram : Create Webpage

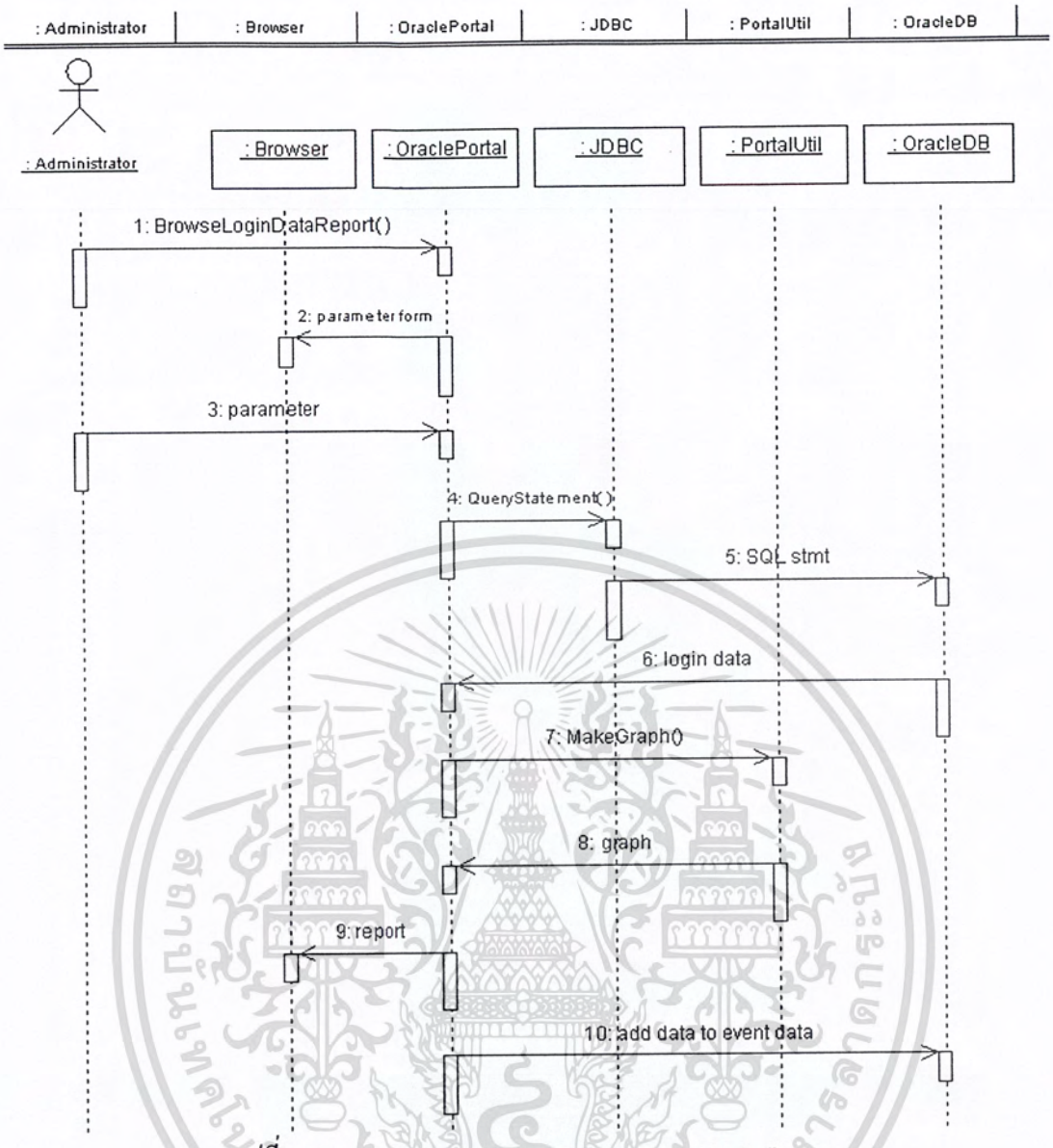


รูปที่ 8-8 Sequence Diagram : Browse Webpage Data Report

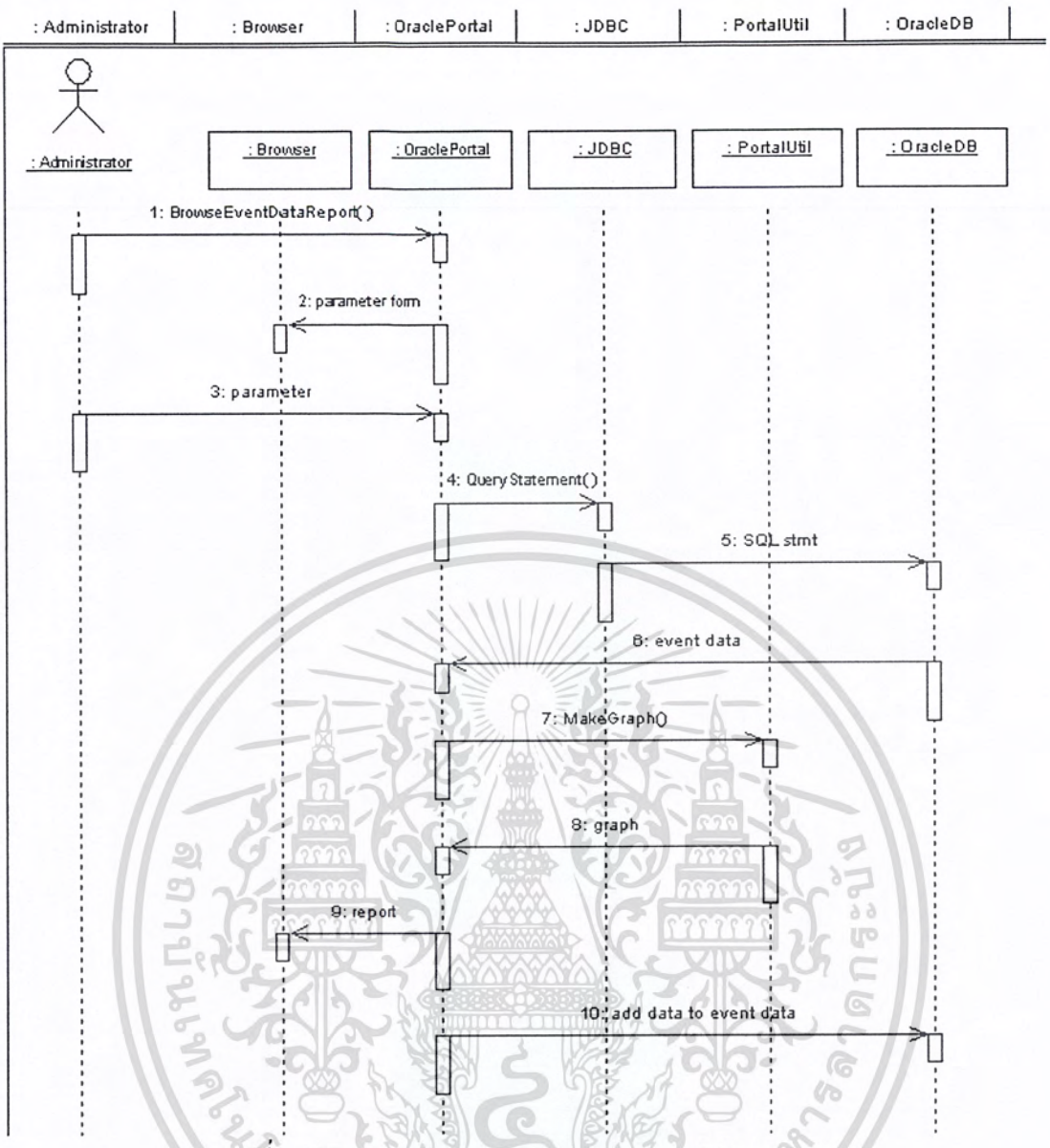


รูปที่ 8-9 Sequence Diagram : Log out

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

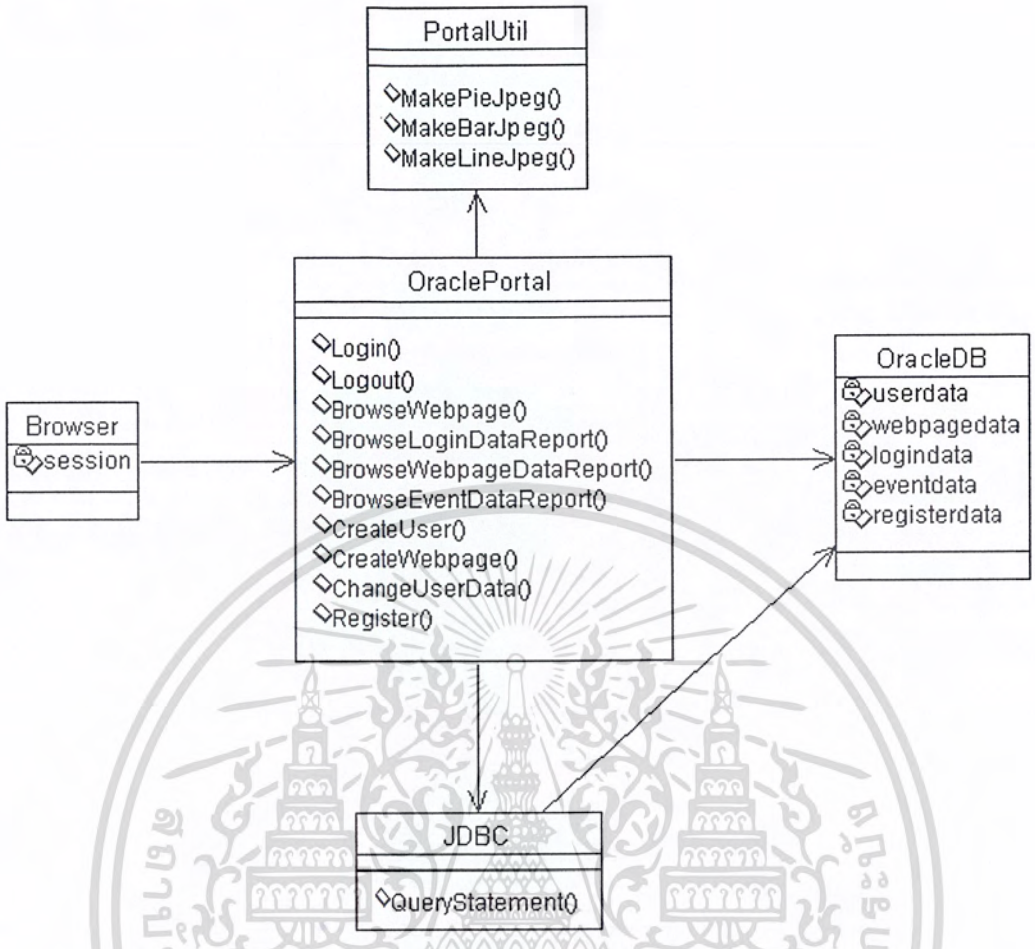


รูปที่ 8-10 Sequence Diagram :Browse Login Data Report



รูปที่ 8-11 Sequence Diagram : Browse Event Data Report

8.3 Class Diagram



รูปที่ 8-12 Class Diagram ของระบบบริหารจัดการพอร์ทอลไซต์

8.4 การออกแบบส่วนติดต่อกับผู้ใช้ (User Interface Design)

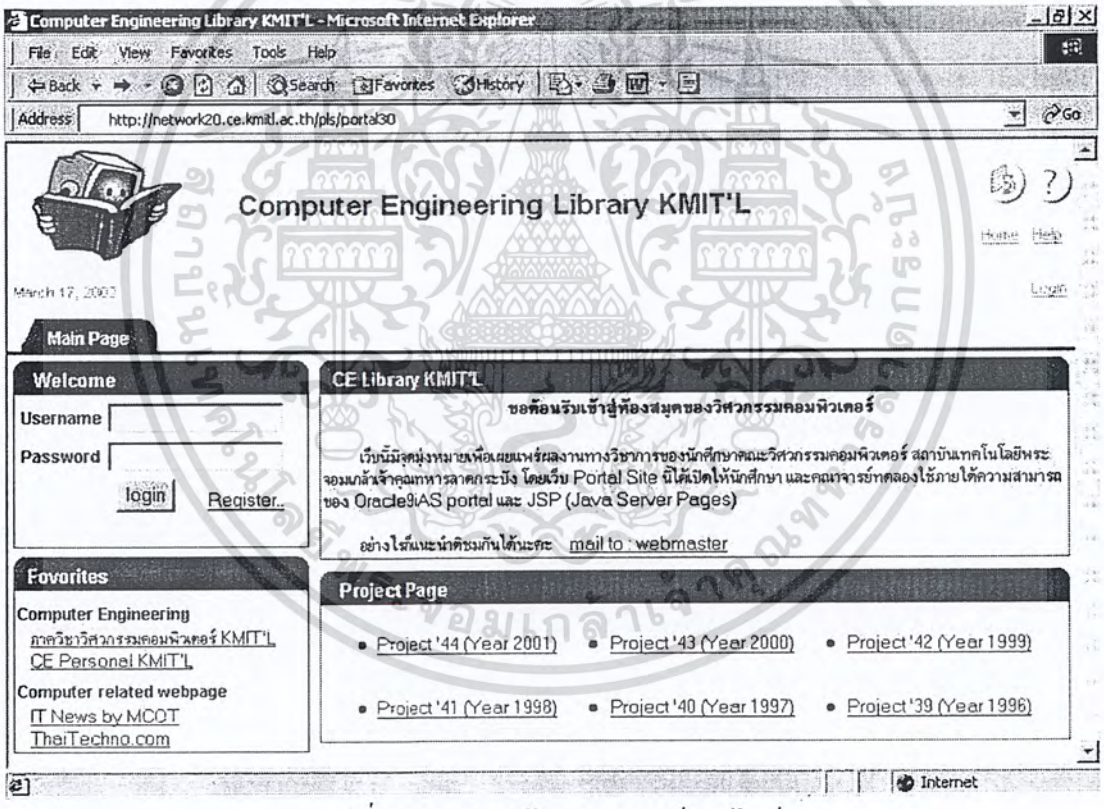
สำหรับการออกแบบส่วนติดต่อกับผู้ใช้ ในที่นี้คือ หน้าตาของเว็บพอร์ทอลไซต์ที่มีความสำคัญมาก นอกเหนือจากการออกแบบให้มีความน่าสนใจแล้ว ยังต้องออกแบบให้มีความสะดวกกับผู้ใช้ (User Friendly) มากที่สุดอีกด้วย

สำหรับการออกแบบส่วนติดต่อกับผู้ใช้ เป็นส่วนที่ติดต่อกับผู้กระทำ (Actors) ต่างๆที่มีในระบบ ซึ่งประกอบด้วย

8.4.1 บุคคลทั่วไป (Public User)

บุคคลทั่วไปที่เข้ามาเยี่ยมชมเว็บเพจ จาก http://network20.ce.kmitl.ac.th/pls/portal30 จะเห็นหน้าตาเว็บไซต์ต่างๆ ดังนี้

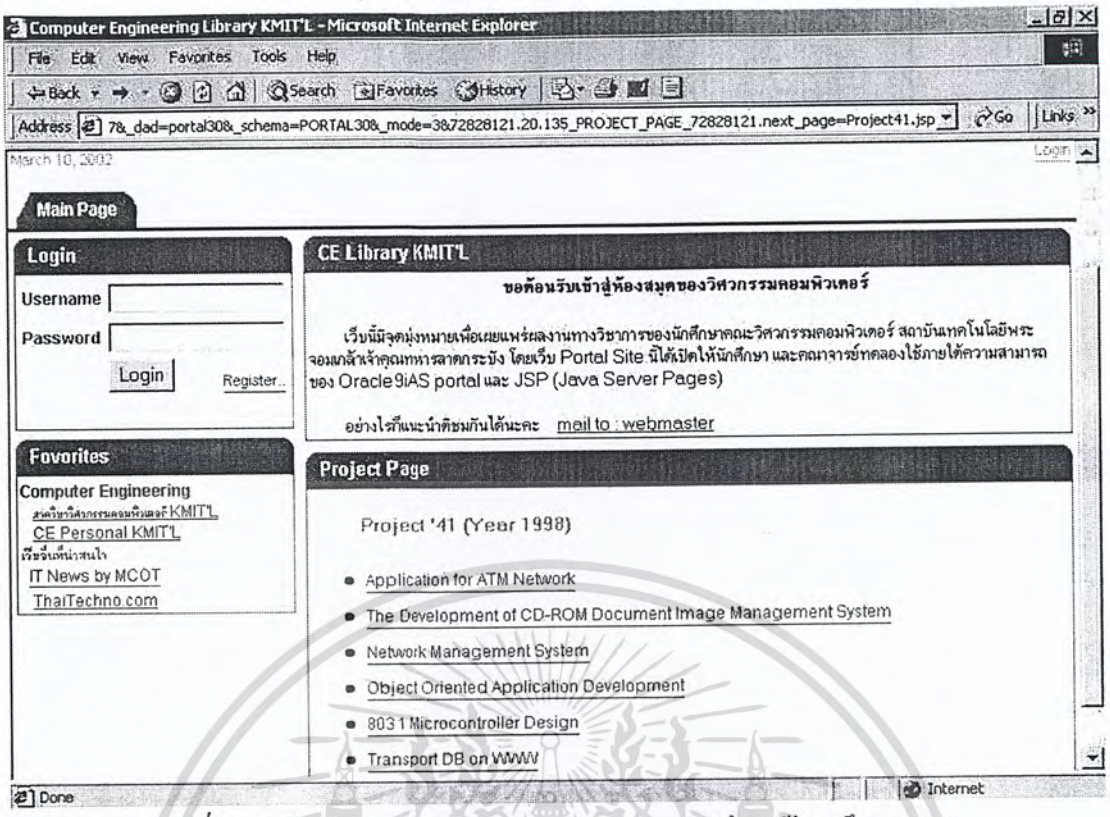
8.4.1.1 หน้าแรกของพอร์ทอลเว็บไซต์ (Public Page)



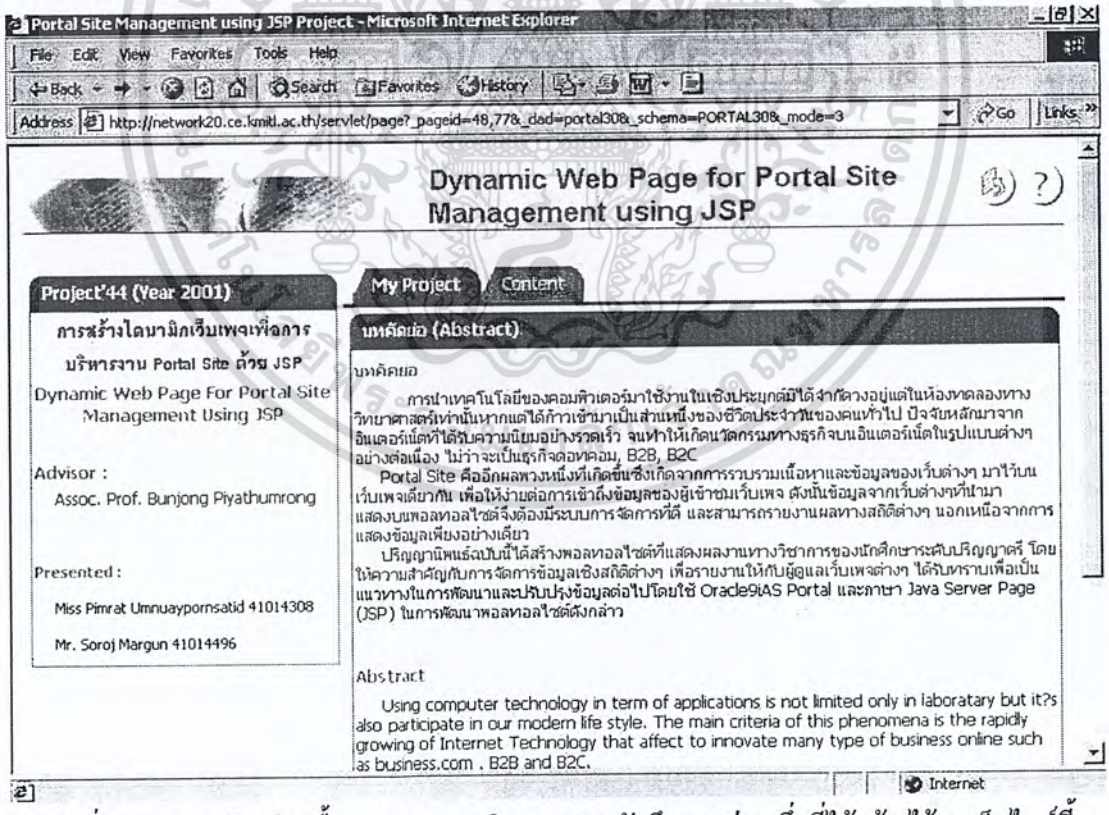
รูปที่ 8-13 แสดงหน้าแรกของพอร์ทอลไซต์

8.4.1.2 เนื้อหาข้อมูลในพอร์ทอลไซต์

พอร์ทอลไซต์ที่สร้างขึ้นนี้มีจุดมุ่งหมายให้เป็นห้องสมุดที่รวบรวมผลงานทางวิชาการของนักศึกษาในแต่ละปีของนักศึกษาคณะวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังเผยแพร่บนเว็บไซต์ โดยจำแนกผลงานทางวิชาการตามปีการศึกษา



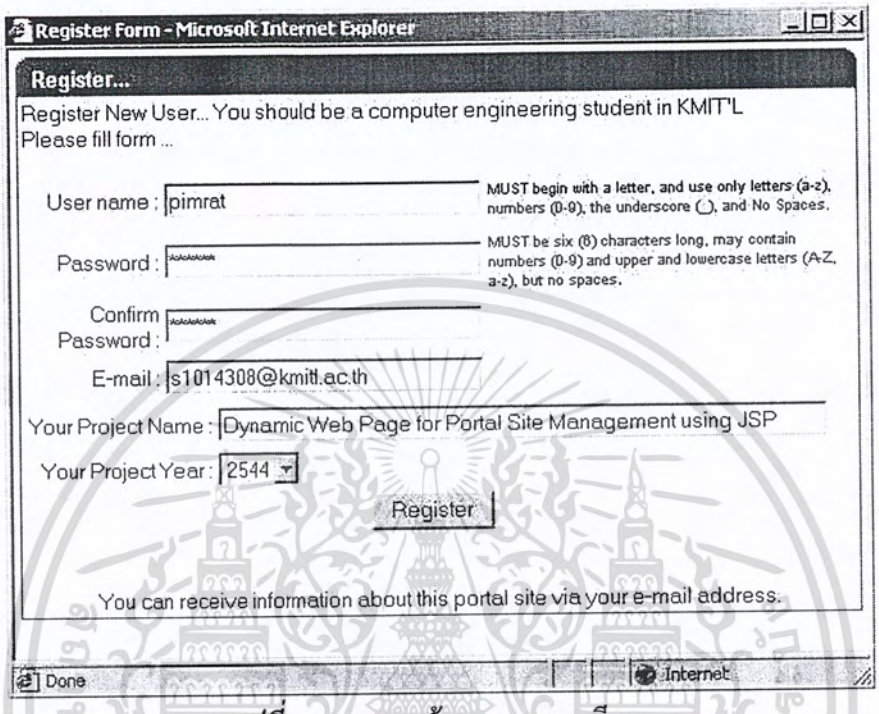
รูปที่ 8-14 ตัวอย่างสารบัญผลงานทางวิชาการของนักศึกษาปีการศึกษา 2541



รูปที่ 8-15 แสดงตัวอย่างเนื้อหาผลงานทางวิชาการของนักศึกษากลุ่มหนึ่งที่ได้สร้างไว้บนเว็บไซต์นี้

8.4.1.3 การลงทะเบียน (Register)

สำหรับนักศึกษาที่ต้องการสร้างเว็บเพจของตัวเองเพื่อแสดงผลงานไว้บนพอร์ทอลไซต์นี้ ต้องมีล๊อกอินเพื่อเข้าใช้งานก่อน ดังนั้นจึงต้องทำการลงทะเบียนขอรหัสผู้ใช้จากผู้ดูแลระบบ โดยคลิกที่ลิงก์ Register ในพอร์ตเล็ต Log in จากหน้าหลักพอร์ทอลไซต์นั่นเอง

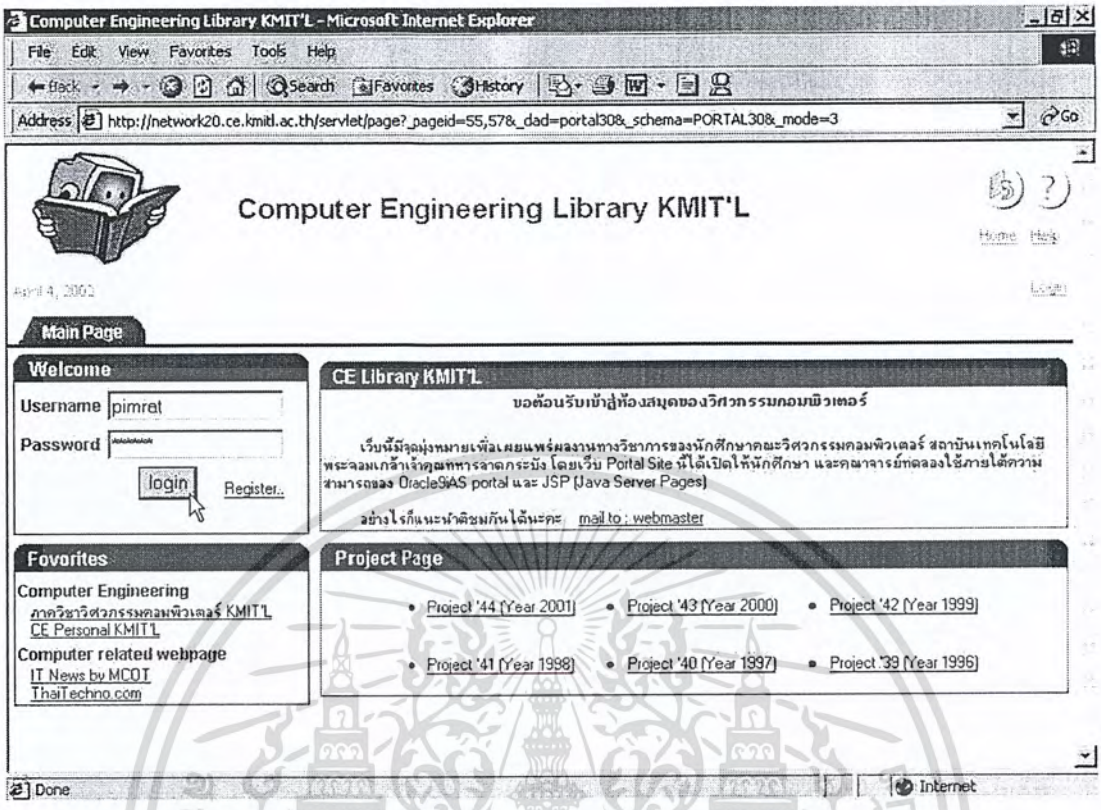


รูปที่ 8-16 แสดงหน้าจอการลงทะเบียน

8.4.2 ผู้ใช้ (Registered User)

ผู้ใช้ของพอร์ทอลไซต์มีสิทธิที่มากกว่าบุคคลทั่วไป นั่นคือ สามารถสร้างเว็บเพจของตัวเองได้ผ่านพอร์ทอลไซต์แห่งนี้ และสามารถเรียกดูสถิติที่เกี่ยวข้องกับเว็บไซต์ของตัวเองได้อีกด้วย ดังนี้

8.4.2.1 การล็อกอิน (Log in)

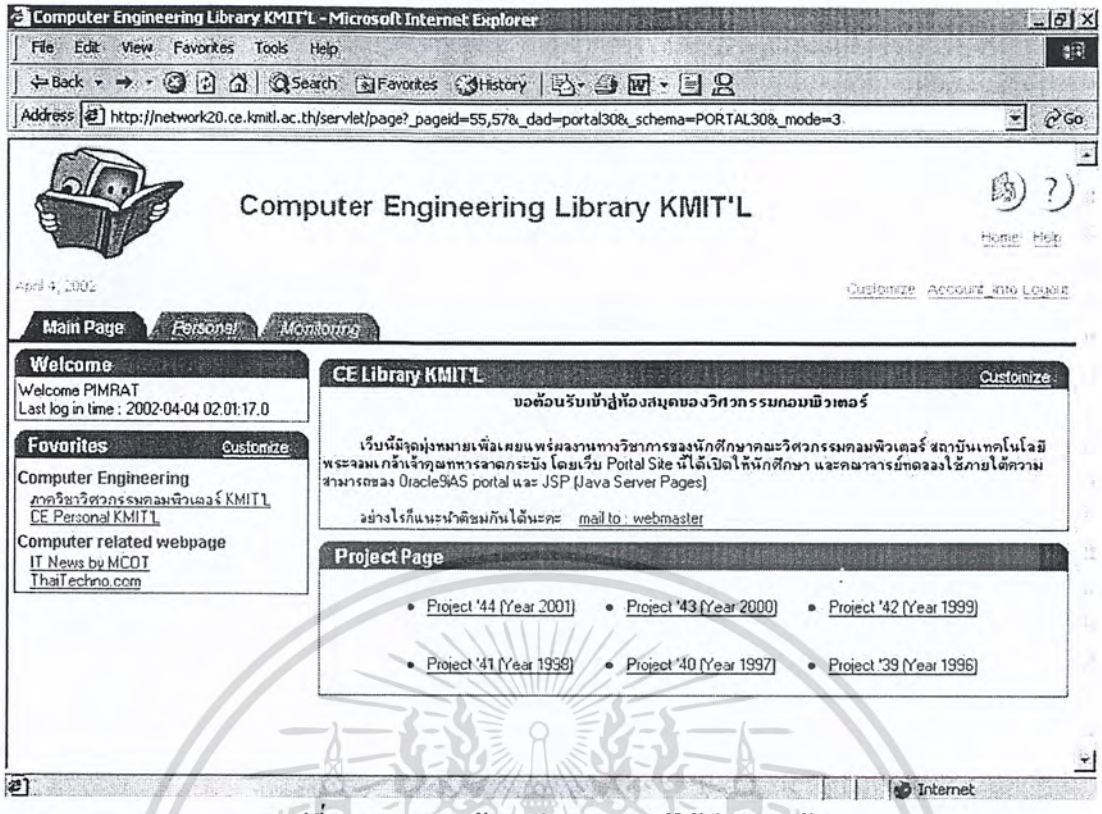


รูปที่ 8-17 แสดงวิธีการล็อกอินเพื่อเข้าใช้งานในฐานะผู้ใช้ของเว็บไซต์

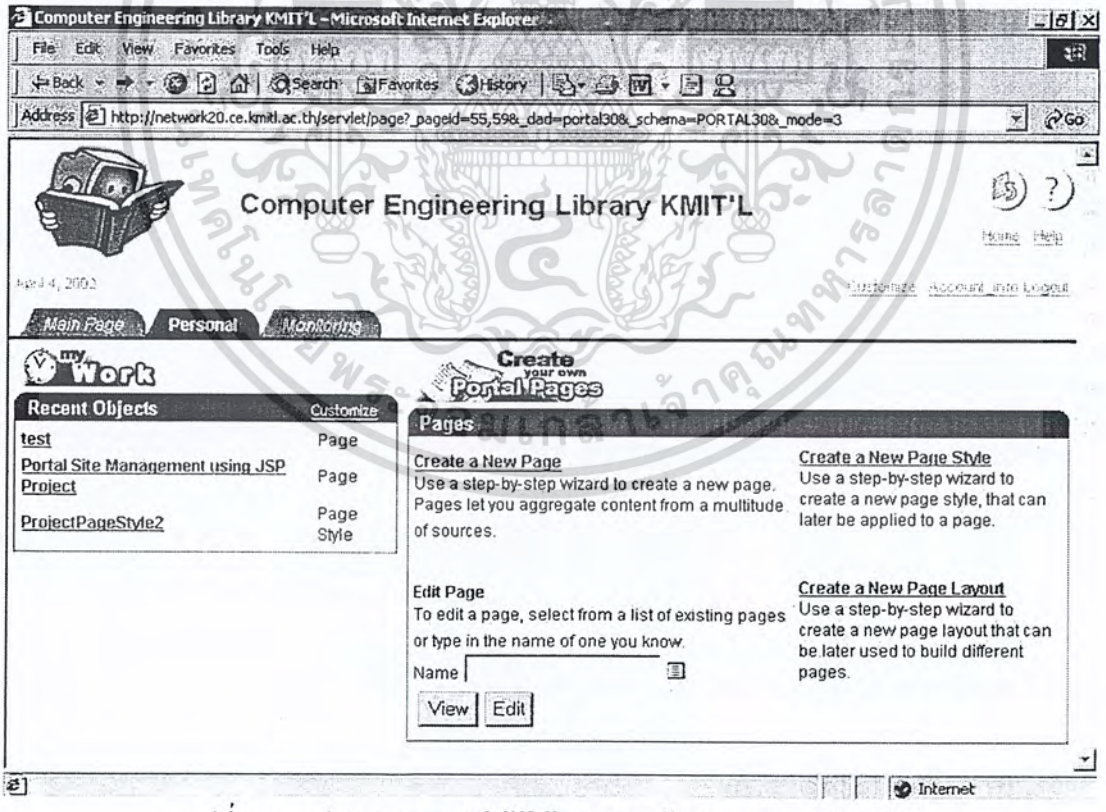
8.4.2.2 เว็บไซต์ส่วนตัว (Personal Website)

หลังจากผู้ใช้ล็อกอินแล้ว หน้าตาของเว็บไซต์จะเปลี่ยนไปเป็นเว็บไซต์ส่วนตัว และสามารถแก้ไขเปลี่ยนแปลงได้โดยคลิก Customize โดยไม่มีผลกระทบต่อส่วนรวม มีรายละเอียดเพิ่มขึ้นมาดังนี้

- แท็บ Main Page เหมือนเว็บเพจหลักของพอร์ทอลไซต์ โดยสามารถเปลี่ยนแปลงแก้ไข Portlet ได้ตามต้องการ ดังรูป 8-18
- แท็บ Personal มีไว้บริการผู้ใช้ที่จะสร้างเว็บเพจ, เพจสไตล์ และเลย์เอาต์บนพอร์ทอลไซต์ ดังรูป 8-19 โดยมีวิซาร์ดช่วยให้การสร้างเพจเป็นไปอย่างสะดวก รวดเร็วขึ้น (สำหรับวิธีการสร้างเพจสามารถดูได้จากภาคผนวก ข.)
- แท็บ Monitoring แสดงรายงานเชิงสถิติในการเข้าชมเว็บเพจที่ผู้ใช้สร้างไว้บนพอร์ทอลไซต์ ดังรูป 8-20 (ดูตัวอย่างผลกราฟรูปแบบต่างๆ ได้จากรูปที่ 8-27, 8-29 และ 8-31)

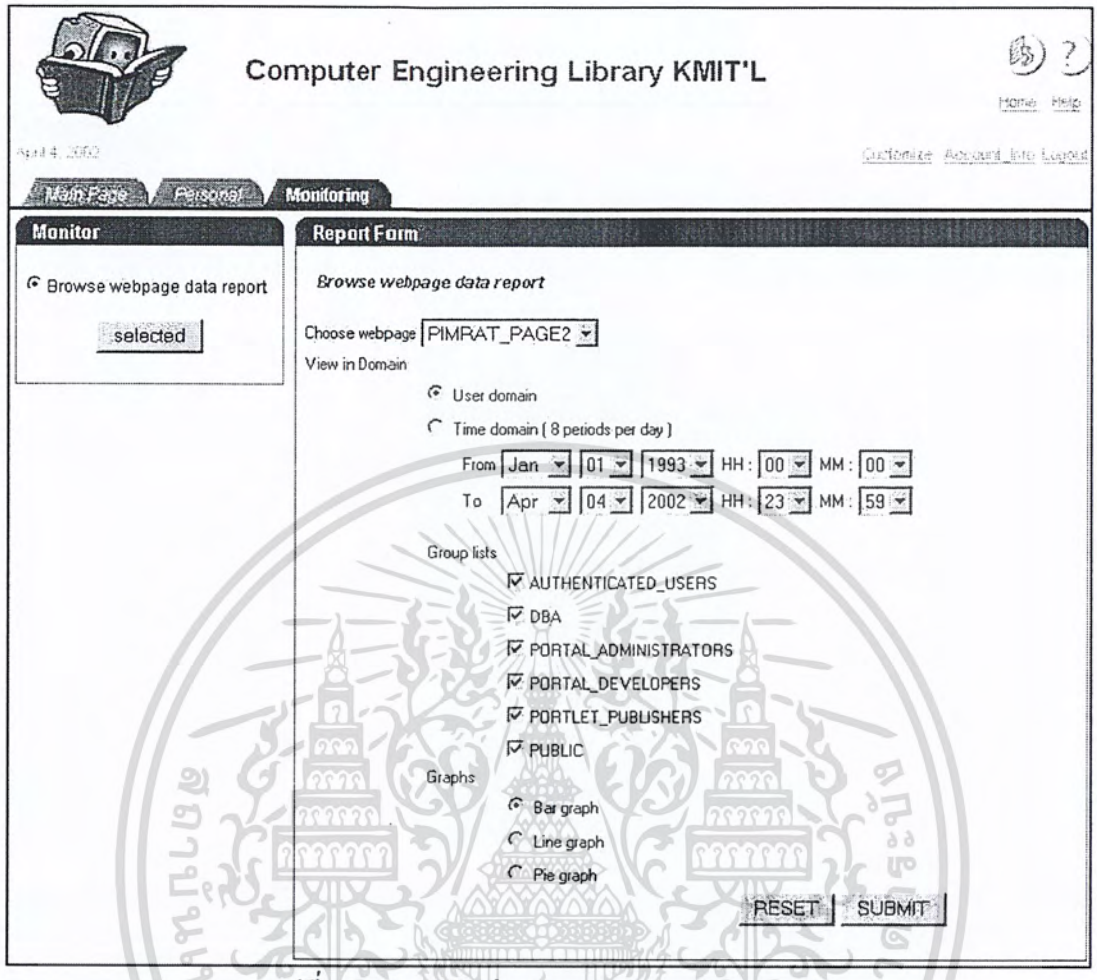


รูปที่ 8-18 แสดงหน้าตาเว็บเพจ หลังผู้ใช้ล็อกอินแล้ว



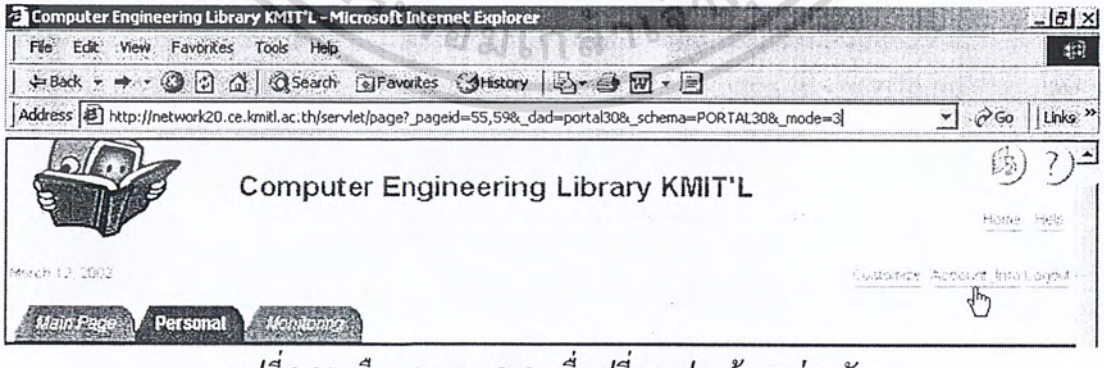
รูปที่ 8-19 แท็บ Personal ทำให้ผู้ใช้สามารถสร้างเว็บเพจของตัวเองได้

8.4.2.3 แท็บ Monitoring

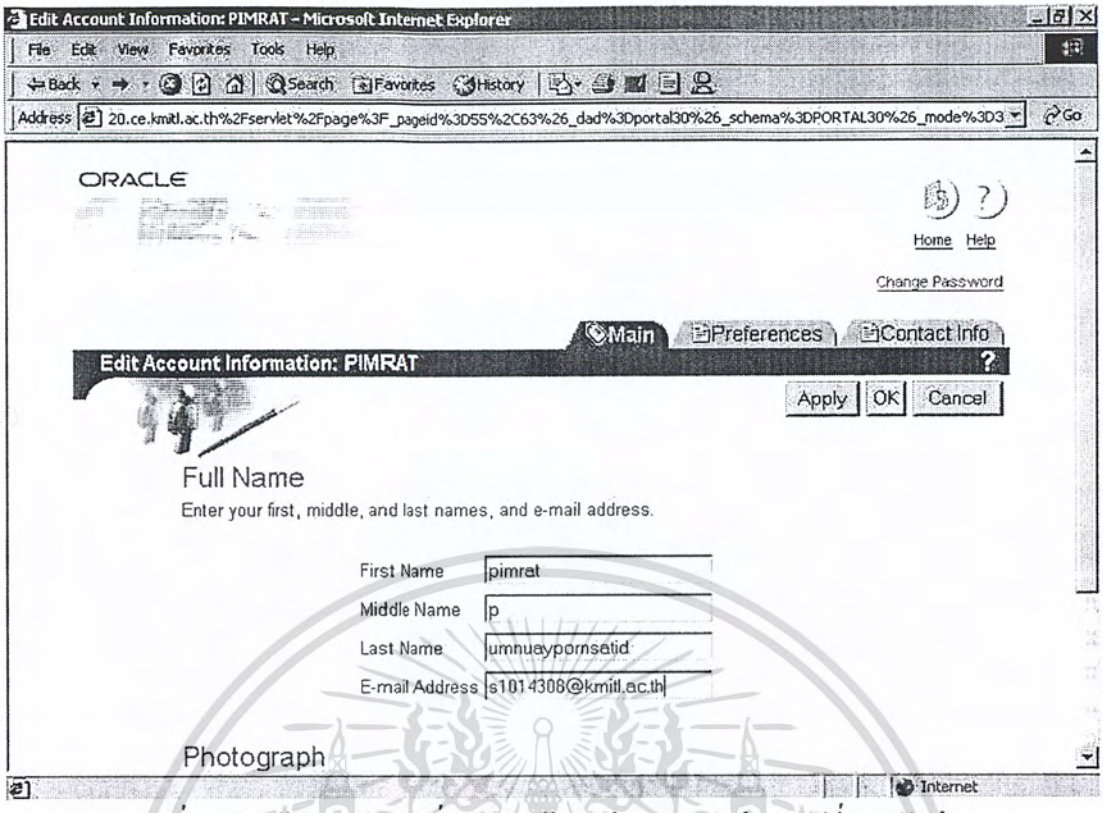


รูปที่ 8-20 แสดงตัวอย่าง Browse webpage data report

8.4.2.4 การเปลี่ยนข้อมูลส่วนบุคคลของผู้ใช้ (Change Account Info)



รูปที่ 8-21 เลือก Accout_Info เพื่อเปลี่ยนแปลงข้อมูลส่วนตัว



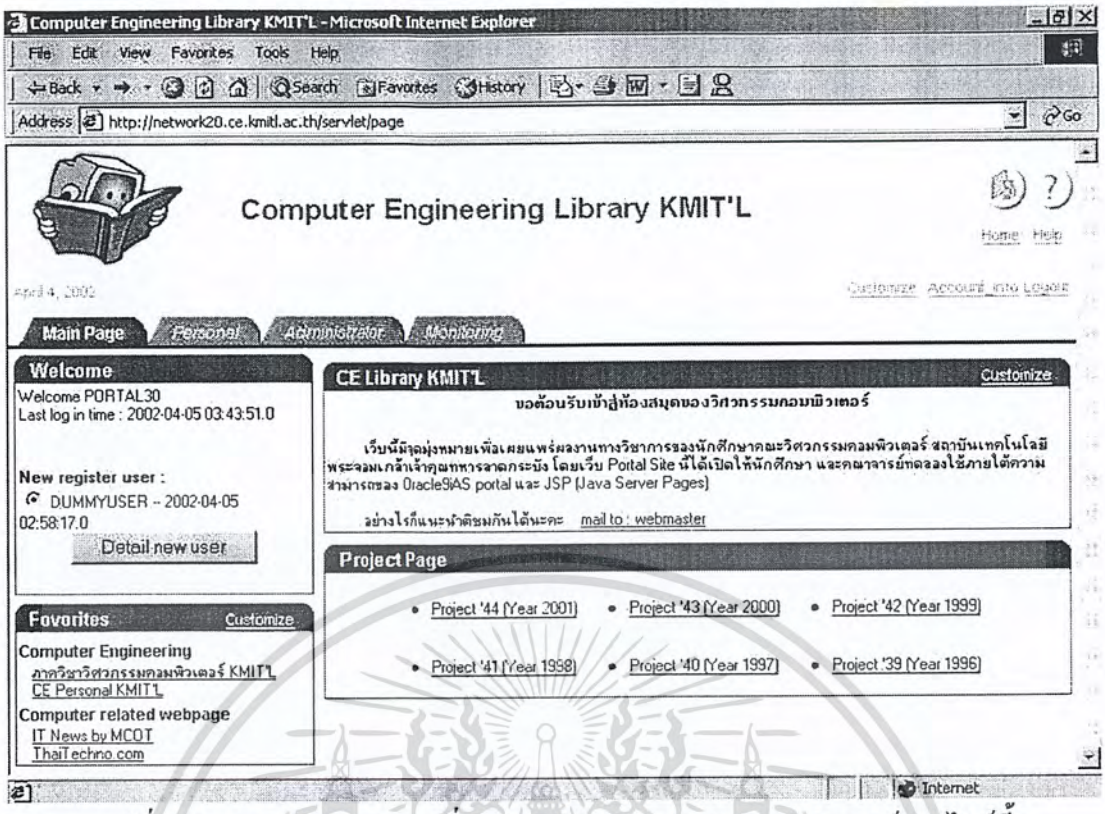
รูปที่ 8-22 หน้าจอสำหรับเปลี่ยนแปลงข้อมูลส่วนตัว รวมถึงการเปลี่ยนรหัสผ่าน

8.4.3 ผู้ดูแลระบบ (Administrator)

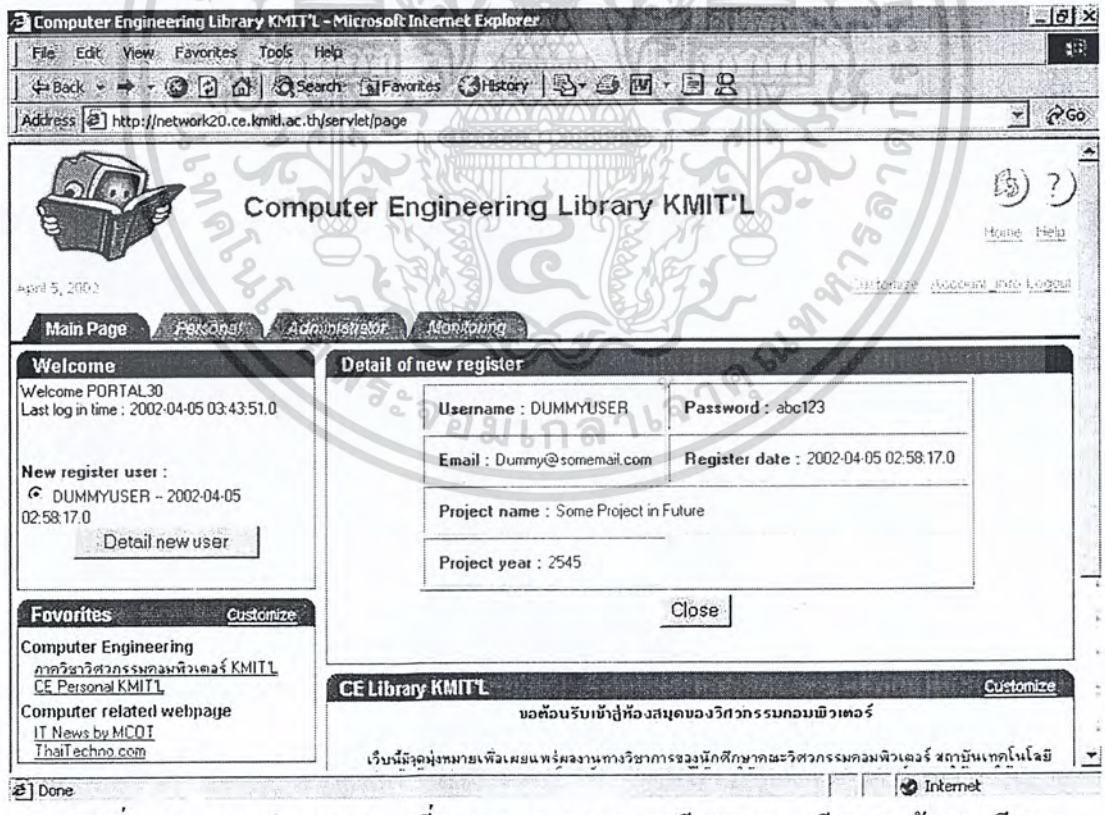
ผู้ดูแลระบบ มีหน้าที่ดูแลระบบให้ทำงานได้อย่างถูกต้อง ในฐานะ Portal Administrator จะมีรายละเอียดเพิ่มเติมดังนี้

8.4.3.1 แท็บ Main Page

เหมือนเว็บเพจหลักของพอร์ทอลไซต์ โดยผู้ดูแลระบบจะเห็นข้อมูลของผู้ลงทะเบียนใหม่และการล็อกอินเข้าใช้งานครั้งก่อน ดังรูป 8-23, 8-24



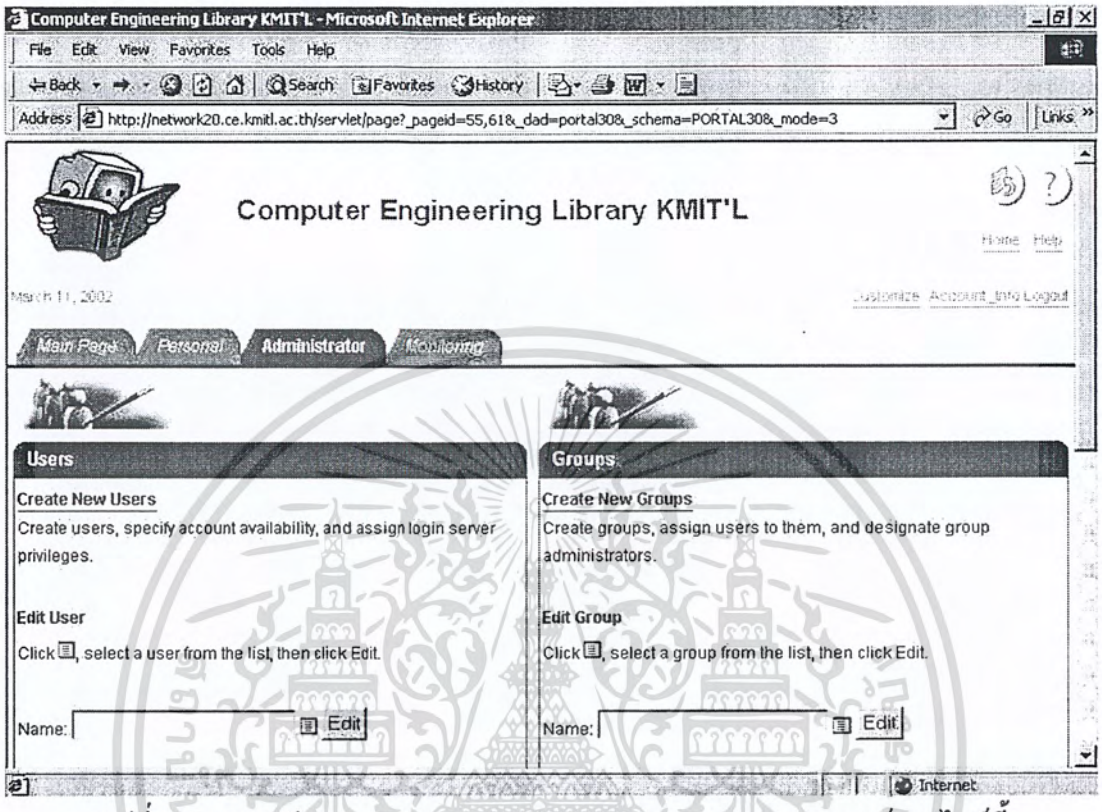
รูปที่ 8-23 แสดงแท็บMain Page ที่ปรากฏแก่ Portal Administrator ของพอร์ทอลไซต์นี้



รูปที่ 8-24 แสดงแท็บMain Page ที่ Portal Administrator เรียกดูรายละเอียดของผู้ลงทะเบียน

8.4.3.2 แท็บ Administrator

ทำให้ผู้ดูแลระบบมีสิทธิ์เพิ่มรายชื่อผู้ใช้หรือแก้ไขข้อมูลผู้ใช้ และเพิ่มกลุ่มของผู้ใช้เพื่อกำหนดสิทธิต่างๆได้ ดังรูป 8-25



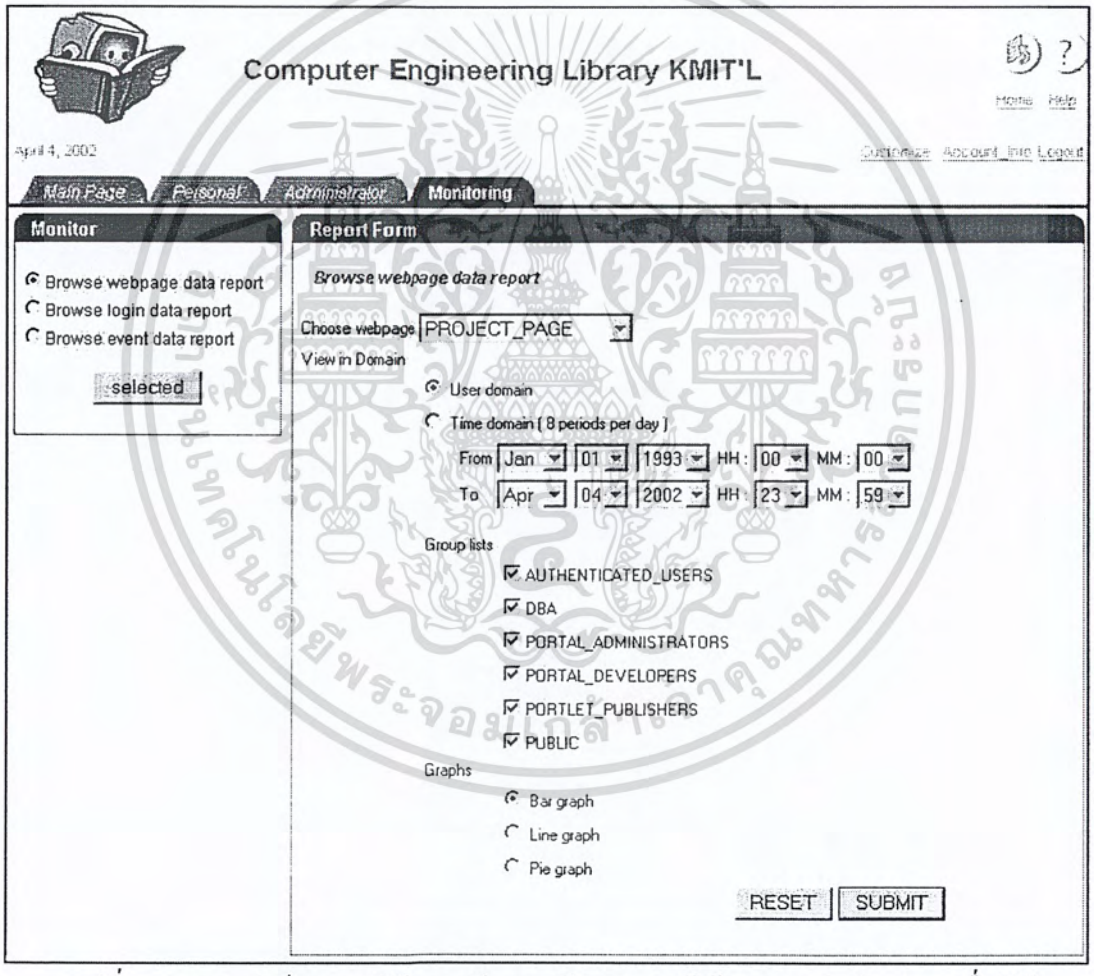
รูปที่ 8-25 แสดงแท็บ Administrator สำหรับ Portal Administrator ของพอร์ทอลไซต์นี้

8.4.3.3 แท็บ Monitoring

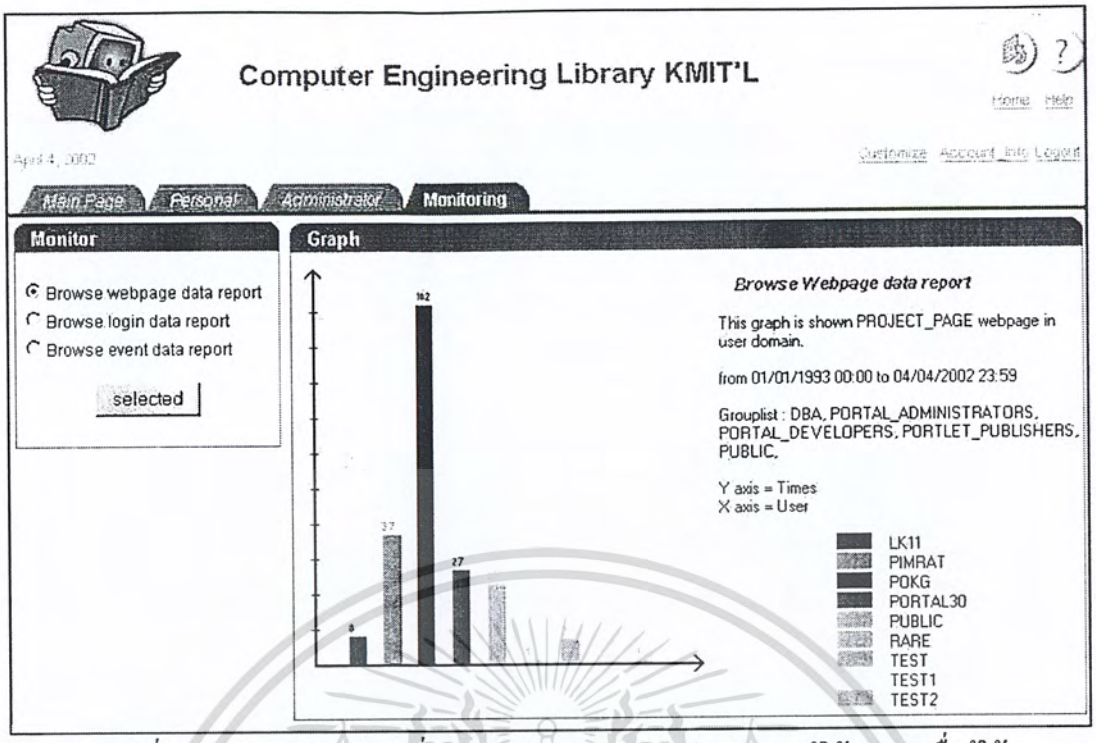
เป็นการแสดงรายงานเชิงสถิติต่างๆ เพื่อประโยชน์ในการวิเคราะห์ผลการดำเนินงานของพอร์ทอลไซต์ โดยมีรายงานสถิติต่างๆ ดังนี้

8.4.3.3.1 รายงานสถิติในการเข้าชมเว็บเพจ (Browse webpage data report)

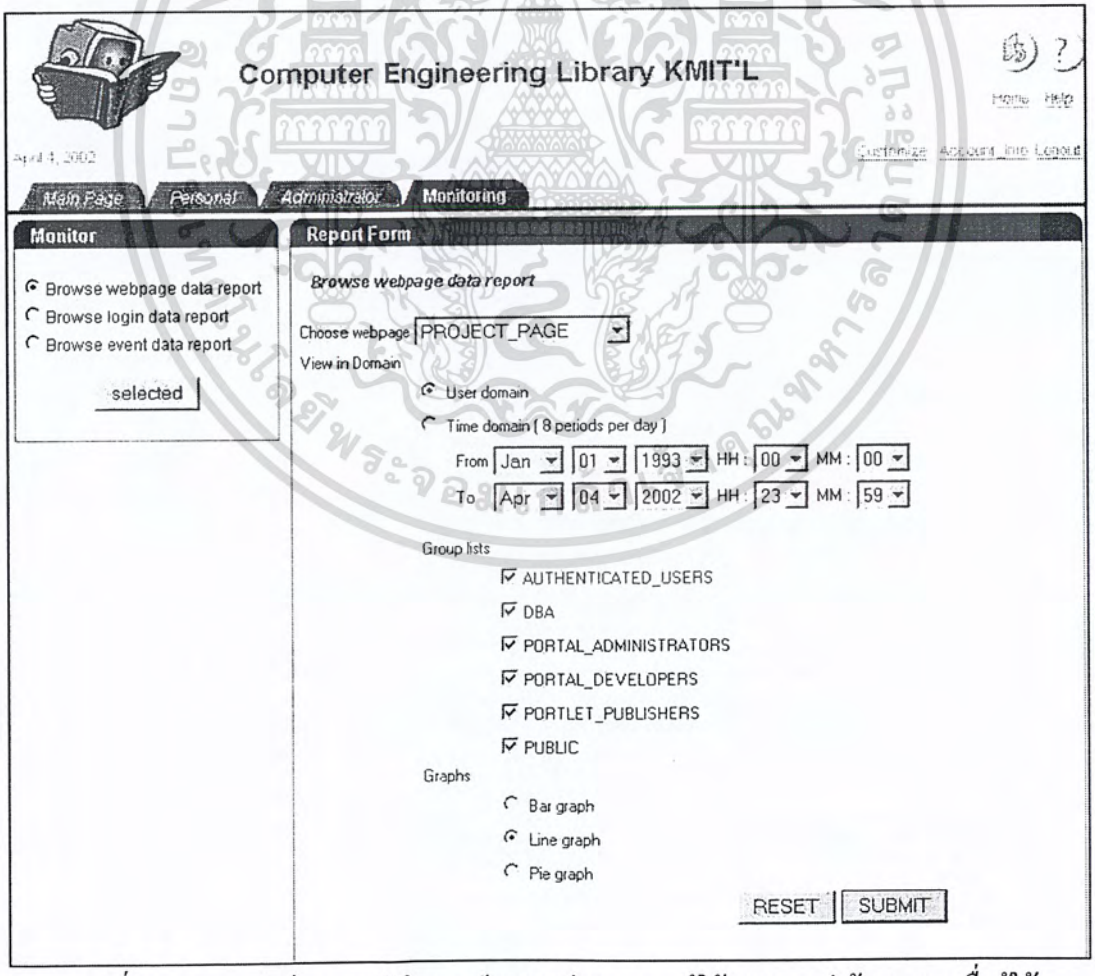
รายงานสถิติในการเข้าชมเว็บเพจของผู้ใช้แต่ละคนหรือทุกเว็บเพจที่สร้างไว้บนพอร์ทอลไซต์ สามารถเลือกว่าจะแสดงผลตามรายชื่อผู้ใช้ (User domain) หรือ ตามช่วงเวลา (Time domain) แบ่งเป็น 8 ช่วงเวลาต่อวัน คือ 0:00 น. – 2:59 น., 3:00 น. – 5:59 น., 6:00 น. – 8:59 น., 9:00 น. – 11:59 น., 12:00 น. – 14:59 น., 15:00 น. – 17:59 น., 18:00 น. – 20:59 น., 21:00 น.– 23:59 น. อีกทั้งสามารถเลือกกราฟการแสดงผลได้ 3 แบบคือ กราฟแท่ง กราฟเส้น และกราฟวงกลม ดังรูปที่ 8-26 ถึง 8-33



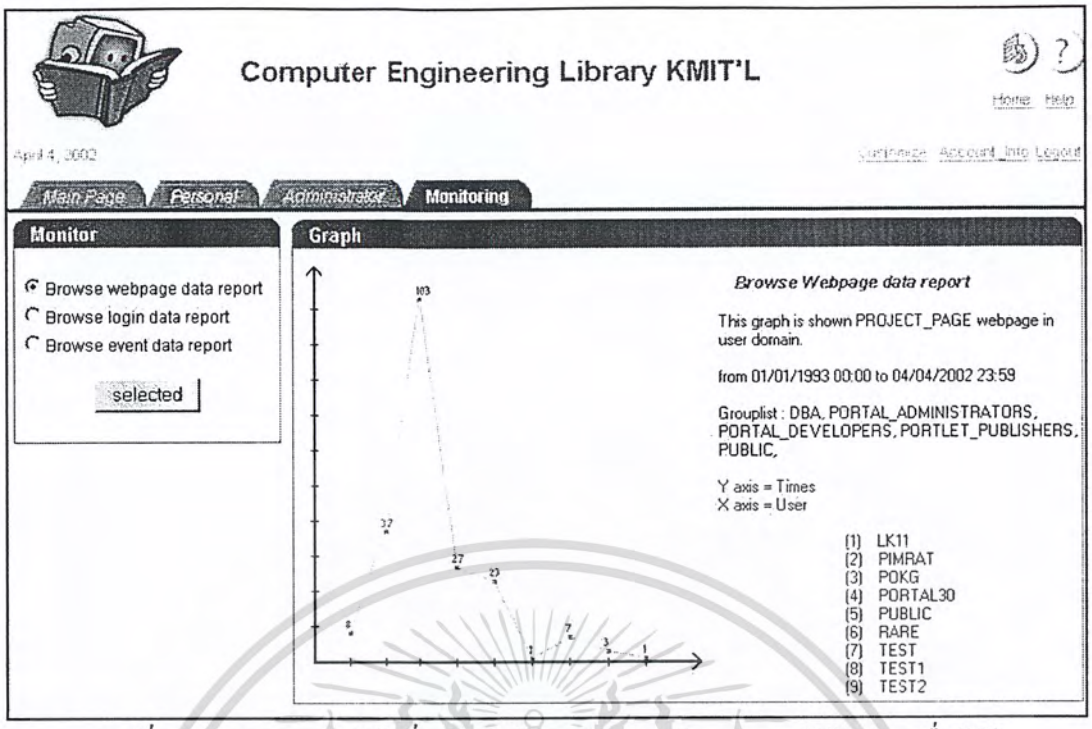
รูปที่ 8-26 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟแท่งตามรายชื่อผู้ใช้



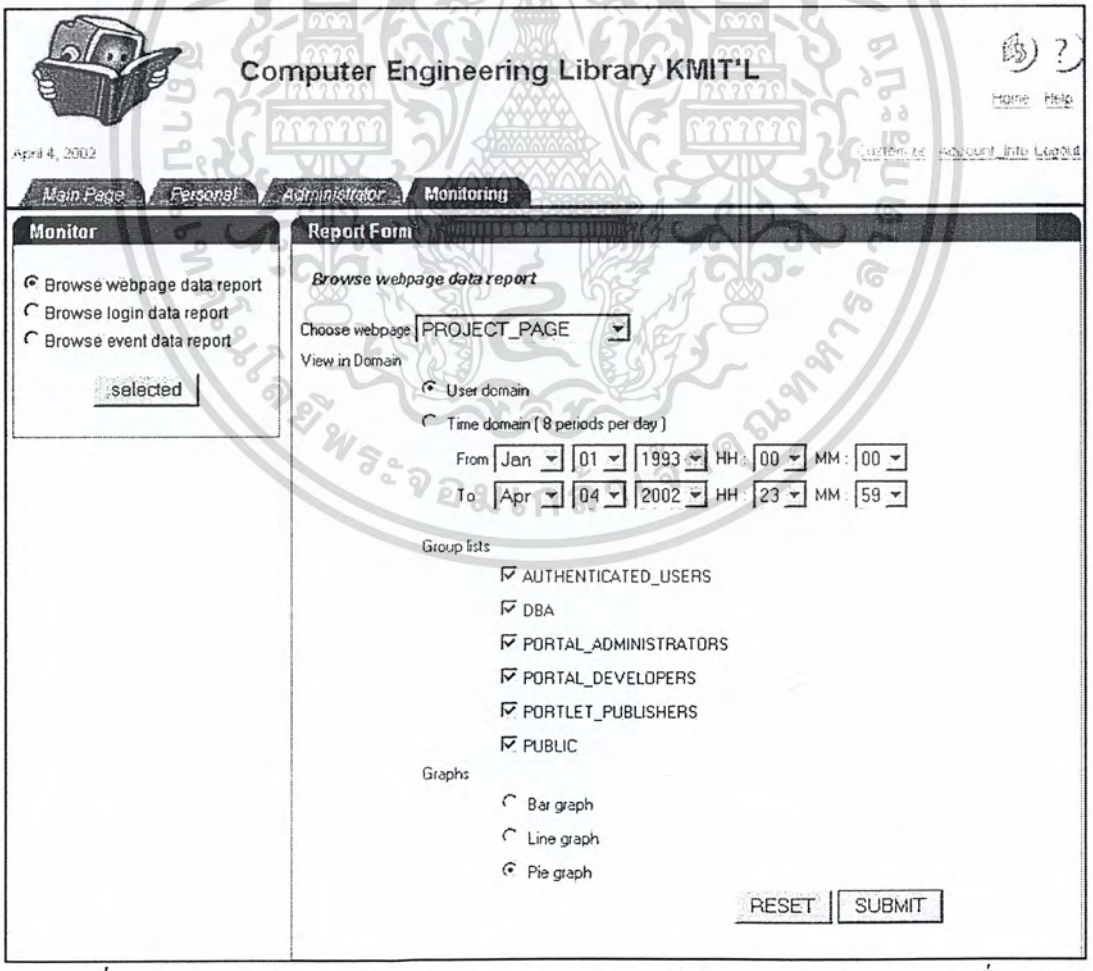
รูปที่ 8-27 แสดงผลกราฟแท่งที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้



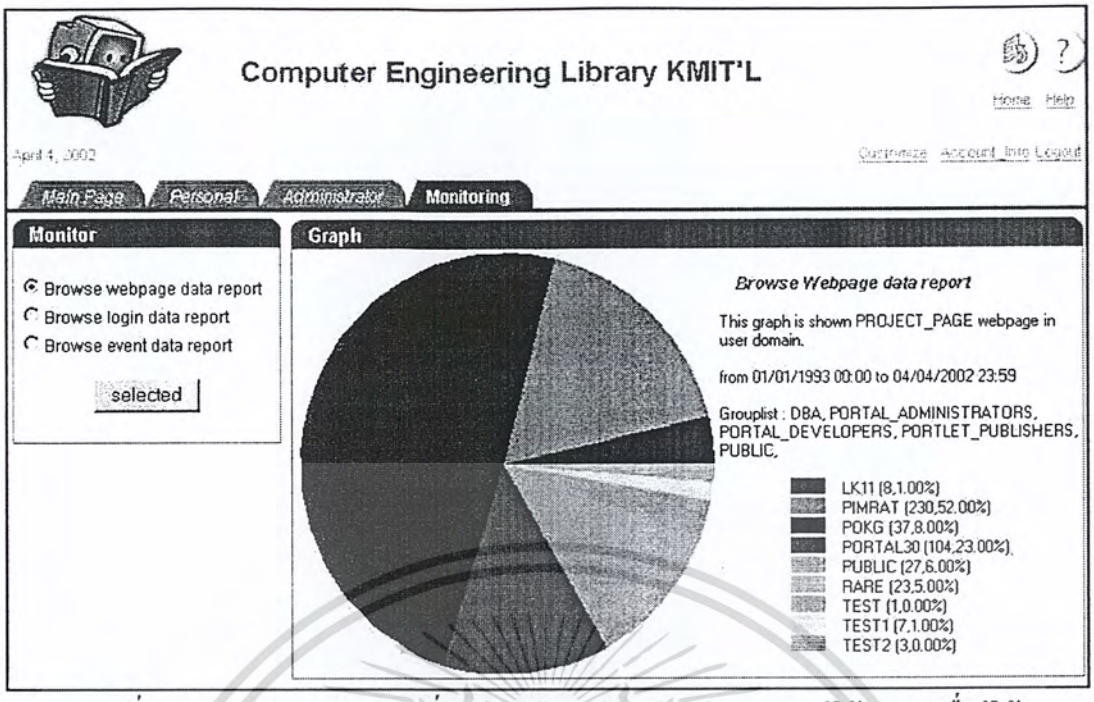
รูปที่ 8-28 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟเส้นตามรายชื่อผู้ใช้



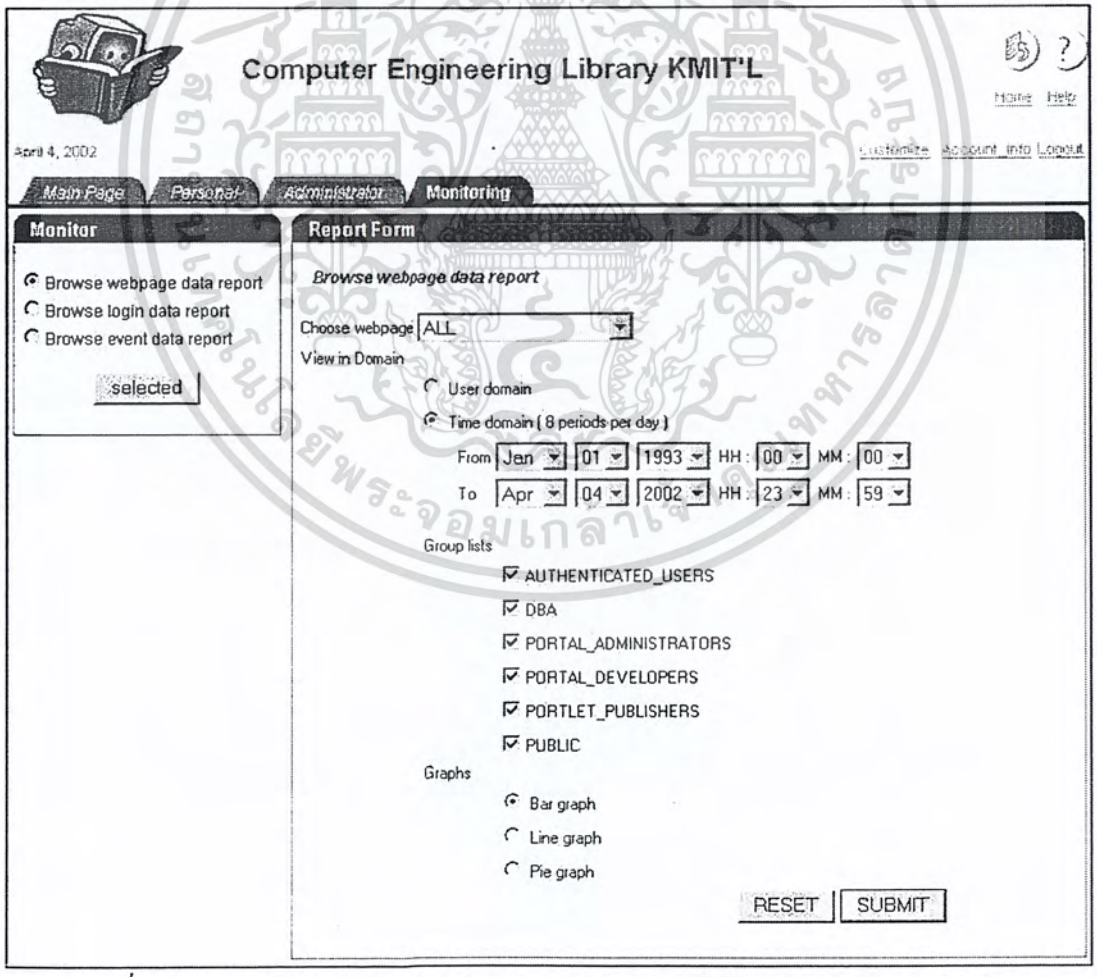
รูปที่ 8-29 แสดงผลกราฟเส้นที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้



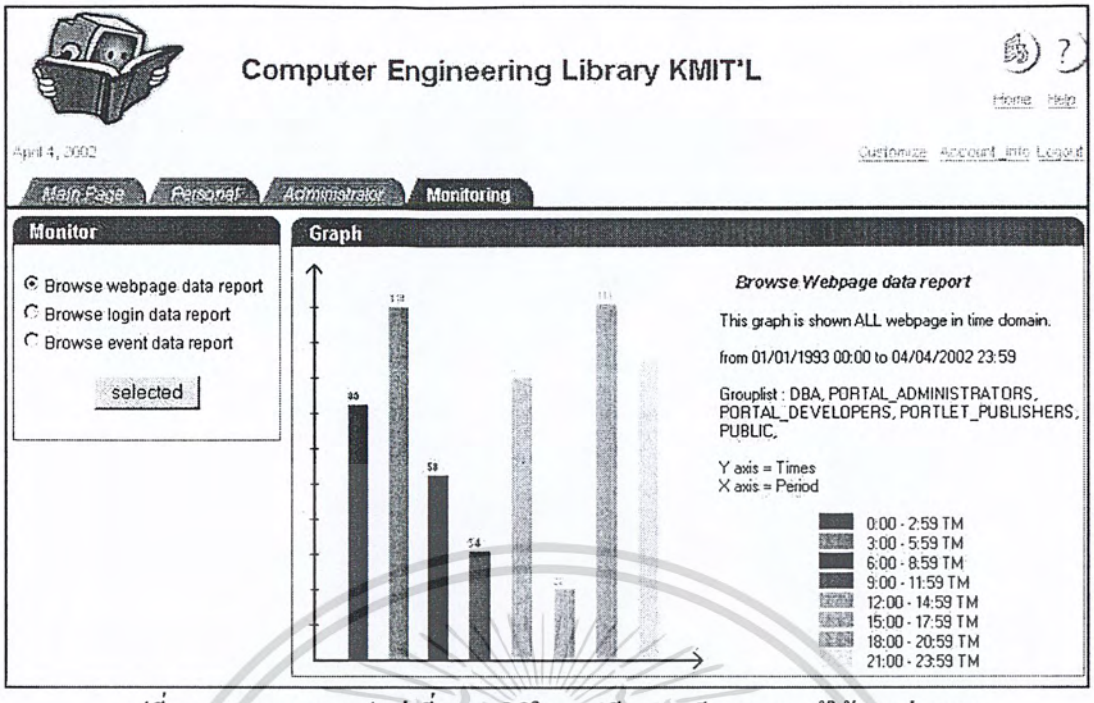
รูปที่ 8-30 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟวงกลมตามรายชื่อผู้ใช้



รูปที่ 8-31 แสดงผลกราฟวงกลมที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามรายชื่อผู้ใช้



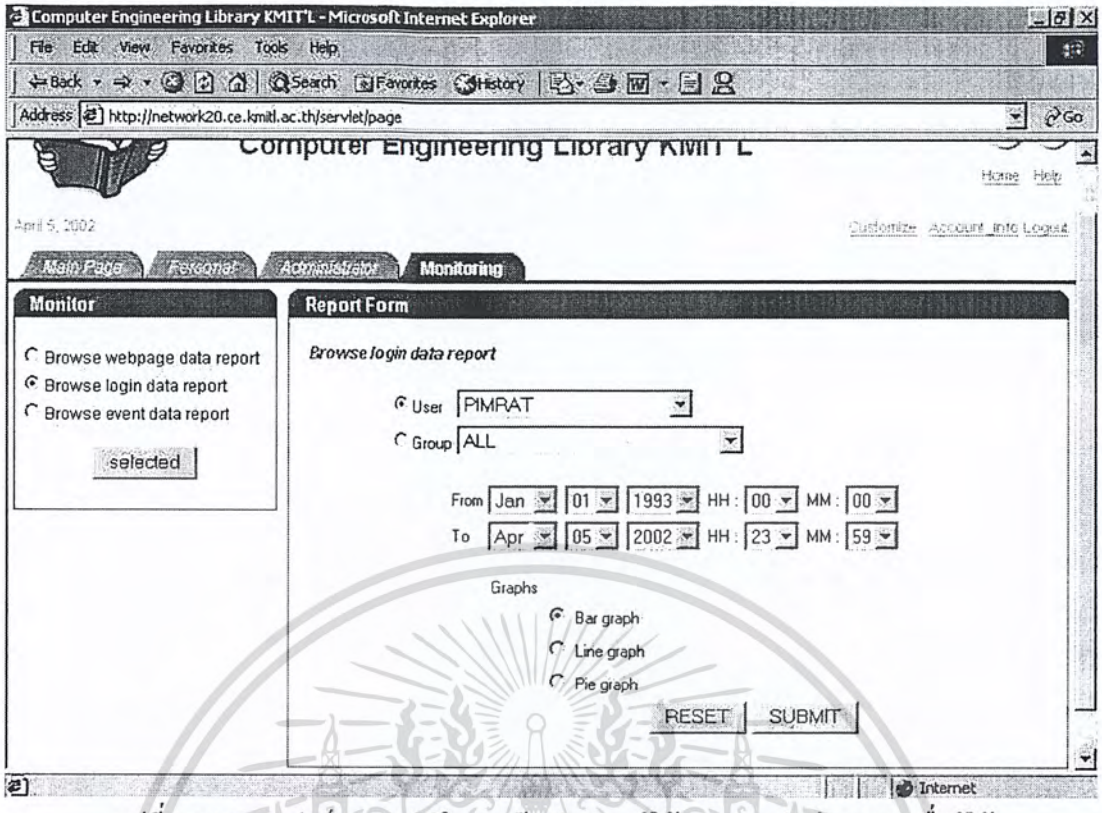
รูปที่ 8-32 แสดงฟอร์มขอดูสถิติในการเรียกชมเว็บเพจของผู้ใช้แบบกราฟแท่งตามช่วงเวลา



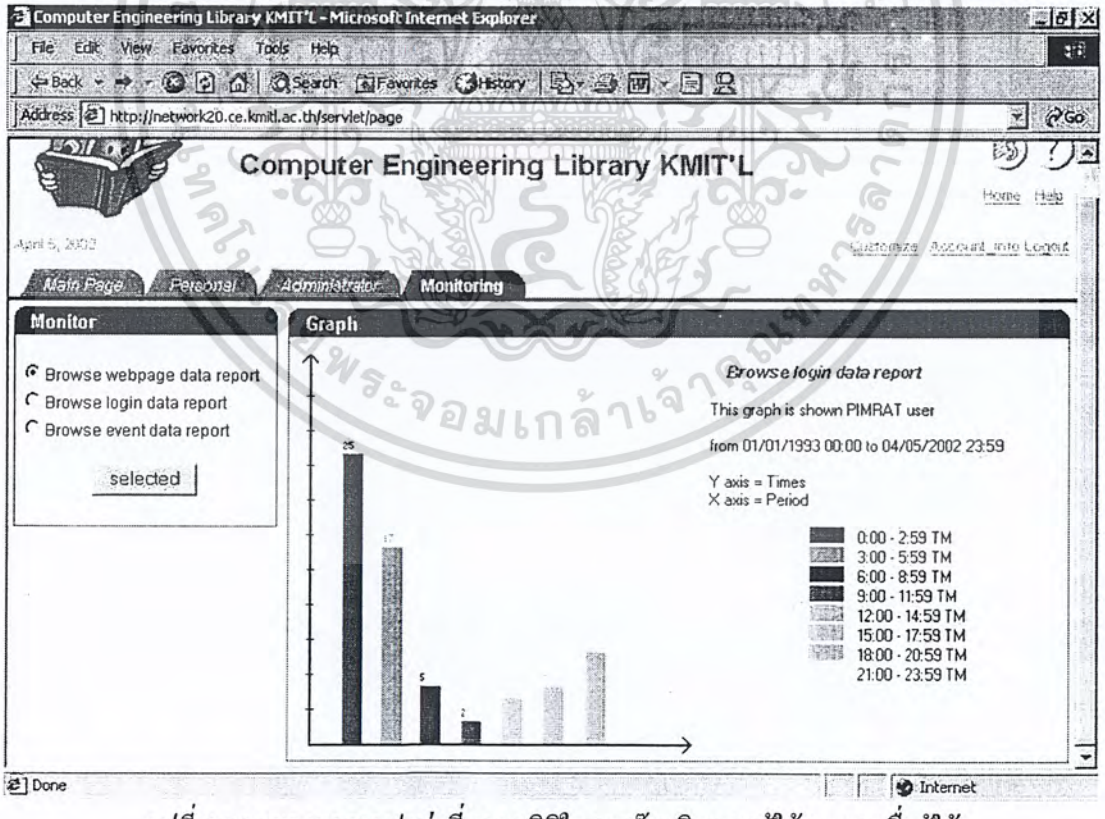
รูปที่ 8-33 แสดงผลกราฟแท่งที่บอกสถิติในการเรียกชมเว็บเพจของผู้ใช้ตามช่วงเวลา

8.4.3.3.2 รายงานสถิติในการล๊อกอินของผู้ใช้ (Browse login data report)

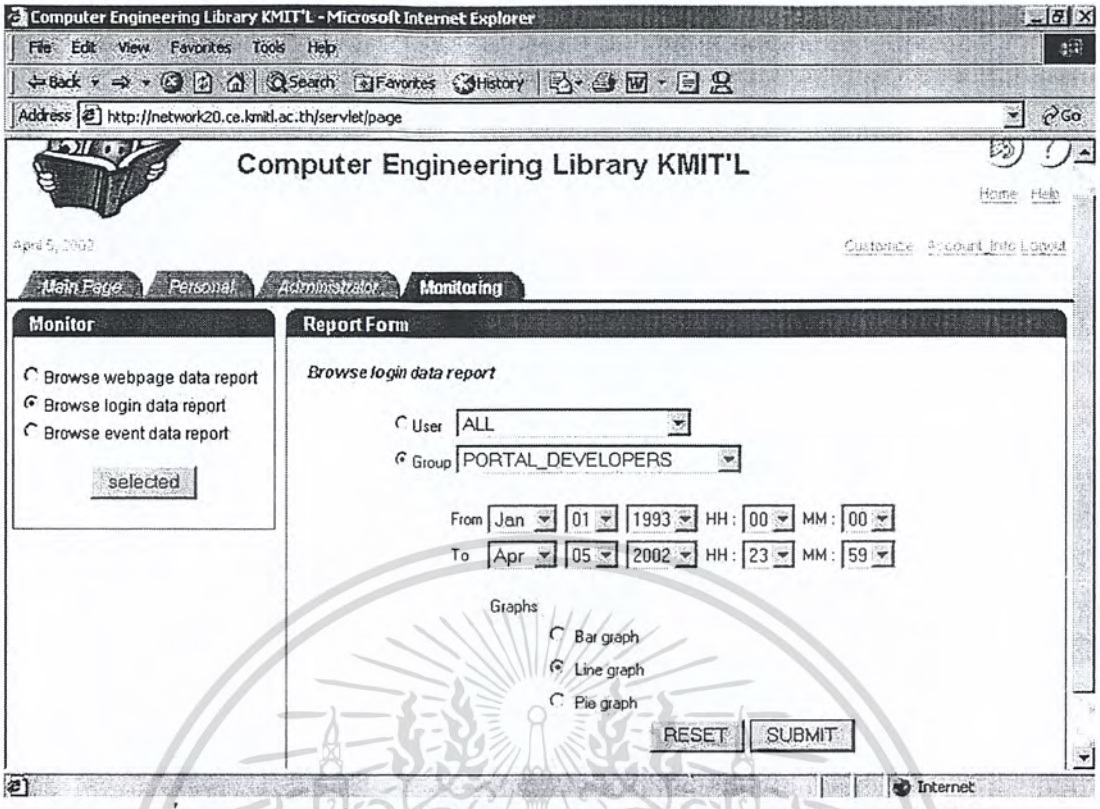
รายงานสถิติในการล๊อกอินของผู้ใช้ สามารถเลือกว่าจะแสดงผลจากรายชื่อผู้ใช้หรือกลุ่มผู้ใช้ (Group) ตามช่วงเวลา ซึ่งแบ่งเป็น 8 ช่วงเวลาต่อวัน คือ 0:00 น. – 2:59 น., 3:00 น. – 5:59 น., 6:00 น. – 8:59 น., 9:00 น. – 11:59 น., 12:00 น. – 14:59 น., 15:00 น. – 17:59 น., 18:00 น. – 20:59 น., 21:00 น.– 23:59 น. อีกทั้งสามารถเลือกกราฟการแสดงผลได้ 3 แบบคือ กราฟแท่ง กราฟเส้น และกราฟวงกลม ดังรูปที่ 8-34 ถึง 8-37



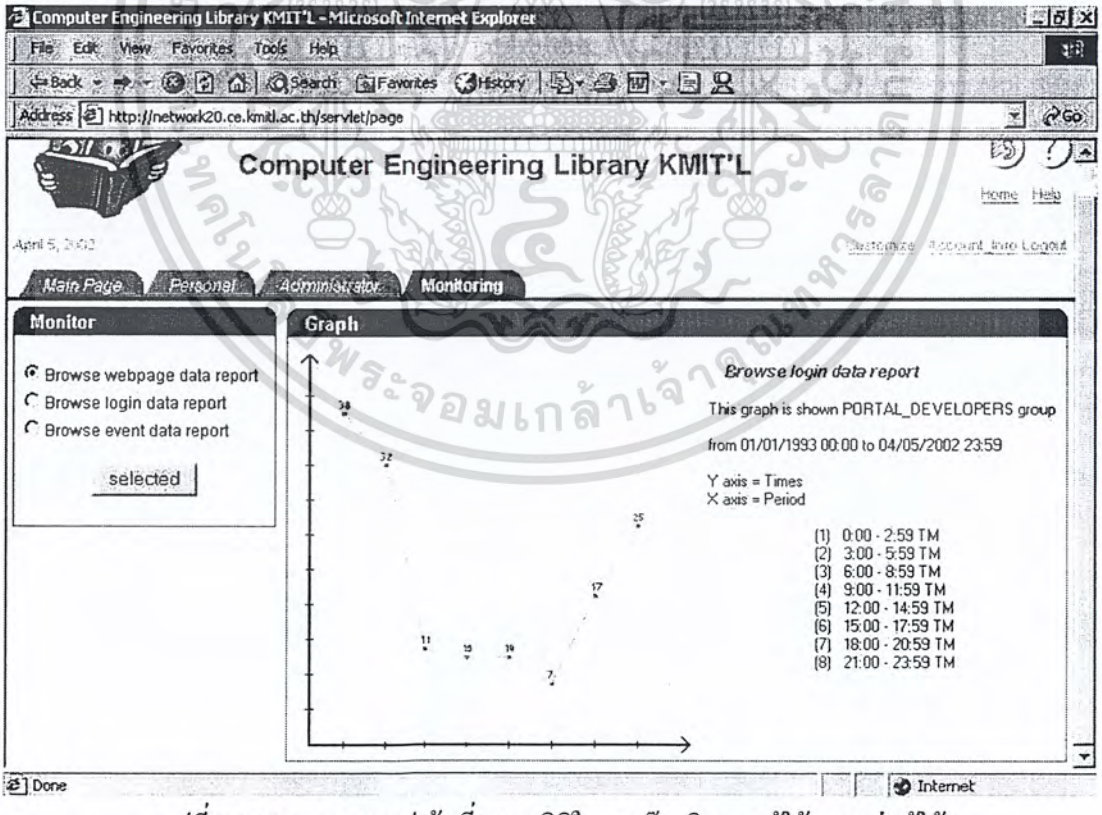
รูปที่ 8-34 แสดงฟอร์มขอดูสถิติในการล็อกอินของผู้ใช้แบบกราฟแท่งตามรายชื่อผู้ใช้



รูปที่ 8-35 แสดงผลกราฟแท่งที่บอกสถิติในการล็อกอินของผู้ใช้ตามรายชื่อผู้ใช้



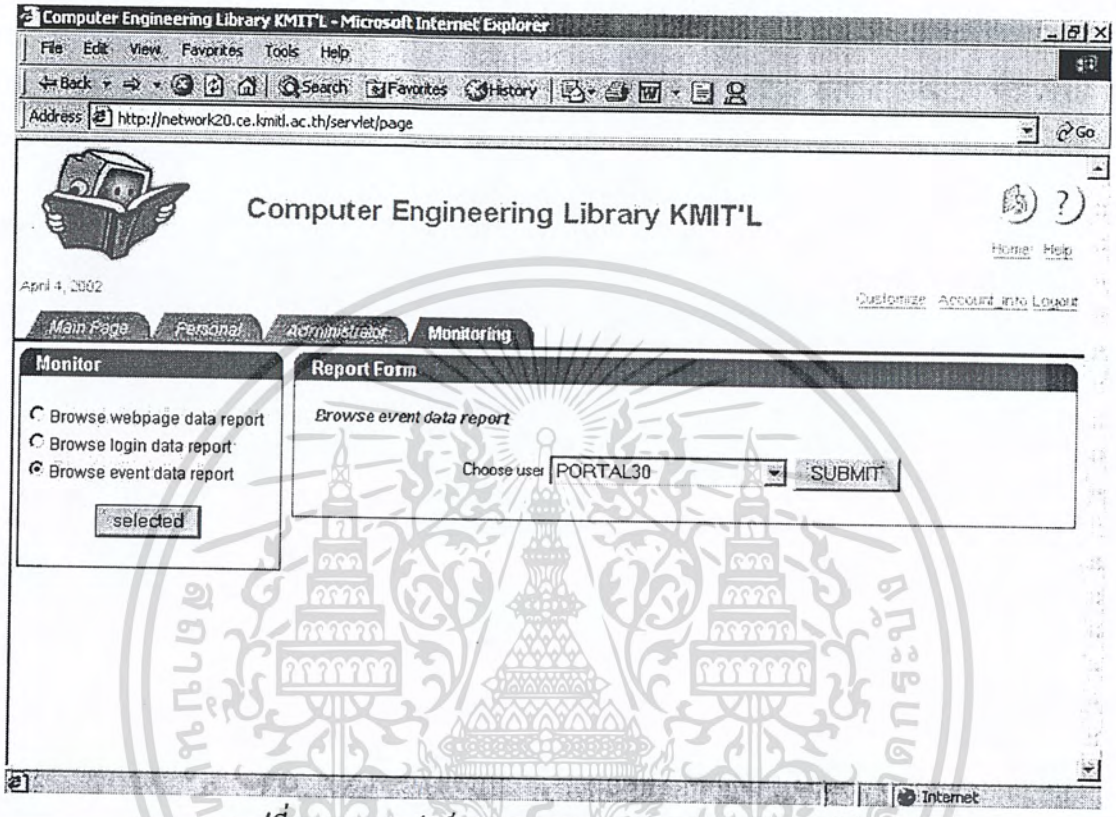
รูปที่ 8-36 แสดงฟอร์มขอดูสถิติในการล็อกอินของผู้ใช้แบบกราฟเส้นตามกลุ่มผู้ใช้



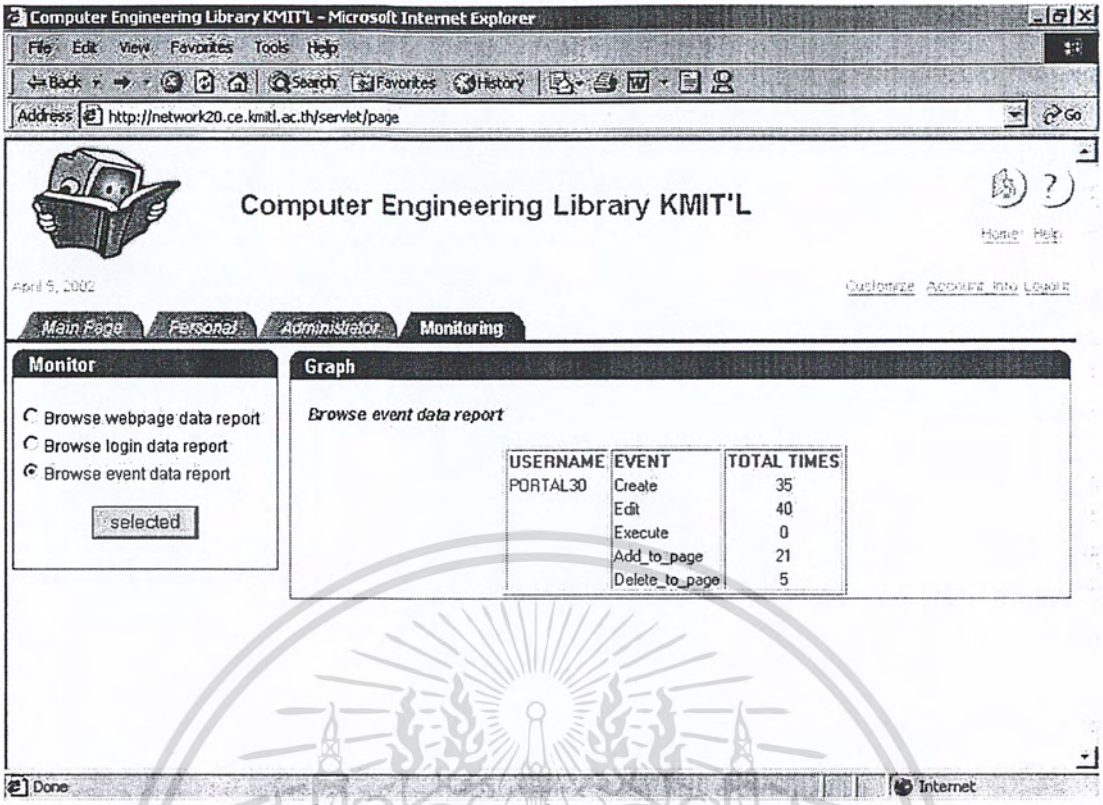
รูปที่ 8-37 แสดงผลกราฟเส้นที่บอกสถิติในการล็อกอินของผู้ใช้ตามกลุ่มผู้ใช้

8.4.3.3.3 รายงานการกระทำต่างๆของผู้ใช้ (Browse event data report)

รายงานการกระทำต่างๆ ของผู้ใช้บนพอร์ทอลไซต์ ได้แก่ การสร้างพอร์ตเล็ต(Create), การแก้ไขพอร์ตเล็ต(Edit), การเรียกพอร์ตเล็ต (Execute), การเพิ่มพอร์ตเล็ตในหน้าเพจ(Add_to_page), การลบพอร์ตเล็ตในหน้าเพจ(Delete_to_page) ดังรูป 8-38 และ 8-39



รูปที่ 8-38 แสดงฟอร์มขอดูรายงานการกระทำต่างๆ ของผู้ใช้



รูปที่ 8-39 แสดงผลตารางที่รายงานการกระทำของผู้ใช้

บทที่ 9
บทวิจารณ์และสรุป

9.1 บทวิจารณ์และสรุป

สำหรับบทวิจารณ์และสรุป จะเป็นการสรุปการทำงานในขั้นตอนต่าง ๆ โดยกล่าวเป็นส่วน ๆ พร้อมทั้งกล่าวถึงปัญหาและวิธีการแก้ไข รวมทั้งได้สรุปการทำงานโดยรวมของระบบไว้ด้วย

9.1.1 ส่วนการศึกษาระบบที่มีอยู่เดิม

ในขั้นต้น เป็นขั้นตอนของการศึกษาระบบเก่าที่มีอยู่เดิม เพื่อเป็นแนวทางในการวิเคราะห์และออกแบบสร้างระบบต่อไป โดยจากการศึกษาระบบการทำงานของพอร์ทอลไซต์ต่าง ๆ เช่น เว็บไซต์ของ IBM, เว็บไซต์ของ Oracle, เว็บไซต์ของ Yahoo เป็นต้น พบว่าพอร์ทอลไซต์แบ่งออกเป็นหลายแบบ เช่น พอร์ทอลไซต์ที่มีลักษณะเป็นที่รวมลิงค์ไปยังเว็บไซต์ต่าง ๆ เช่น เว็บไซต์ของ Yahoo หรือพอร์ทอลไซต์ที่ทำหน้าที่รวบรวมข่าวสารจากที่ต่าง ๆ กันมาแสดงผลอยู่ด้วยกัน เป็นต้น โดยพอร์ทอลไซต์ที่เราสนใจ มีลักษณะการทำงานดังนี้ คือ เป็นเว็บไซต์ที่รวบรวมข่าวสารมาอยู่ที่เกี่ยวข้องมาอยู่ในหน้าเดียวกันเพื่อให้สามารถเข้าถึงได้ง่าย และสามารถให้ผู้ใช้ที่ลงทะเบียนกับระบบแล้วสามารถเลือกรูปแบบของการแสดงผลได้ตามแบบที่ตัวเองชอบ โดยไม่จำเป็นต้องเหมือนกับผู้ใช้คนอื่น

9.1.2 ส่วนการวิเคราะห์และการออกแบบระบบ

ในส่วนของการวิเคราะห์และออกแบบระบบนั้น ได้เลือกใช้หลักการของออบเจ็กต์โอเรียนเท็ด (Object Oriented) เป็นหลักในการออกแบบ เนื่องจากมีข้อดีคือ ทำให้ผู้พัฒนาสามารถวิเคราะห์ และออกแบบระบบได้อย่างสะดวก และมีประสิทธิภาพ โดยใช้ภาษา UML ช่วยในการออกแบบระบบ เนื่องจากเป็นภาษาที่มีมาตรฐาน และมีการใช้งานอย่างกว้างขวาง

9.1.3 ส่วนการพัฒนาระบบ

ในส่วนของการพัฒนาระบบนั้น ได้เลือกใช้ซอฟต์แวร์ Oracle 8i Database (8.1.7) เป็นตัวจัดการฐานข้อมูลของระบบ ใช้ซอฟต์แวร์ Oracle 9iAS Portal (3.0.8) เป็นเว็บเซิร์ฟเวอร์ และเป็นตัวจัดการพอร์ทอลไซต์ทั้งหมด และใช้วินโดว์ 2000 เซิร์ฟเวอร์ (Windows 2000 Server) เป็นระบบปฏิบัติการ

ส่วนตัวโปรแกรมบริหารจัดการพอร์ทอลไซต์นั้น พัฒนาโดยใช้เทคโนโลยี JSP เป็นตัวพัฒนา เนื่องจากได้รับข้อดีของภาษาจาวาคือ เป็นภาษาที่ใช้หลักการของออบเจ็กต์โอเรียนเท็ด ทำให้ง่ายต่อการทำความเข้าใจ และเป็นภาษาที่ไม่ขึ้นกับแพลตฟอร์ม ทำให้ไม่มีปัญหาในการนำไปใช้งานบนเครื่องต่าง ๆ โดยใช้ JDBC เป็นตัวกลางในการติดต่อกับตัวฐานข้อมูล

9.1.4 ส่วนวิจารณ์และบทสรุปโดยรวมของระบบ

ระบบบริหารจัดการพอร์ทอลไซต์ที่ได้พัฒนาขึ้นนั้น มีความสามารถครบตามที่ได้ออกแบบมาและสามารถตอบสนองความต้องการพื้นฐานในการบริหารจัดการพอร์ทอลไซต์ได้ แต่ก็ยังถือว่ามีฟังก์ชันในการบริหารจัดการน้อยมากหากจะนำไปใช้งานจริง และเนื่องจากพอร์ทอลไซต์มีหลายแบบ โปรแกรมระบบบริหารจัดการพอร์ทอลไซต์ที่พัฒนาขึ้นมานี้อาจไม่เหมาะสำหรับพอร์ทอลไซต์แบบอื่น ๆ

9.2 ปัญหาและแนวทางในการพัฒนาระบบ

9.2.1 ปัญหาเกี่ยวกับการแสดงผลภาษาไทย

เนื่องจากซอฟต์แวร์ Oracle 9iAS Portal ไม่สนับสนุนภาษาไทย 100% ทำให้ ในบางพอร์ตเล็ทต์ (portlet) ไม่สามารถแสดงผลภาษาไทยได้ ดังนั้นควรจะมีการพัฒนาให้สามารถแสดงผลภาษาไทยได้ 100% บน Oracle 9iAS Portal

9.2.2 ปัญหาเกี่ยวกับแหล่งอ้างอิง

เอกสารอ้างอิงเกี่ยวกับข้อมูลฟังก์ชันต่าง ๆ ของ Oracle 9iAS ยังมีน้อยอยู่เนื่องจากเป็นเทคโนโลยีที่ค่อนข้างใหม่ และเนื่องจากซอฟต์แวร์ Oracle 9iAS เป็น Commercial ware ทำให้ไม่สามารถหาข้อมูลสนับสนุนในบางเรื่องได้ เนื่องจากไม่สามารถเข้าถึงข้อมูลบางส่วนที่มีไว้บริการเฉพาะลูกค้าของบริษัท Oracle เท่านั้น ดังนั้นหากจะมีการพัฒนาต่อ ควรมีการจัดทำเอกสารติดต่อขอความร่วมมือกับทางบริษัท Oracle อย่างเป็นทางการ เพื่อให้ผู้ที่มาพัฒนาต่อสามารถเข้าถึงข้อมูลส่วนนั้น ๆ ได้

9.2.3 การรักษาความปลอดภัยของข้อมูล

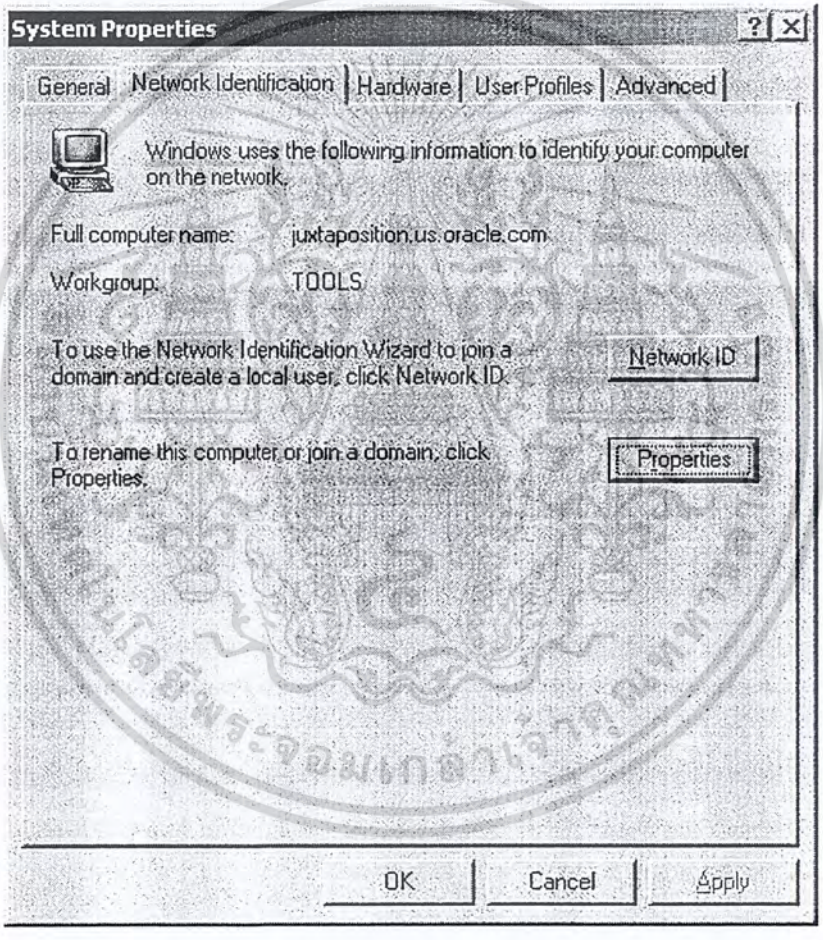
เนื่องจากเป็นการเรียกใช้บริการผ่านเว็บบราวเซอร์ ทำให้ระบบรักษาความปลอดภัยของข้อมูลผู้ใช้มีเพียงแค่การใช้เซสชั่นเท่านั้น ซึ่งหากนำไปใช้งานจริงอาจไม่ปลอดภัยเพียงพอ ดังนั้นหากมีการพัฒนาต่อ ควรจะเพิ่มระบบรักษาความปลอดภัยให้มากขึ้น เช่น ใช้ HTTPS ในการส่งข้อมูลที่มีความสำคัญแทนที่ HTTP เป็นต้น

ภาคผนวก ก

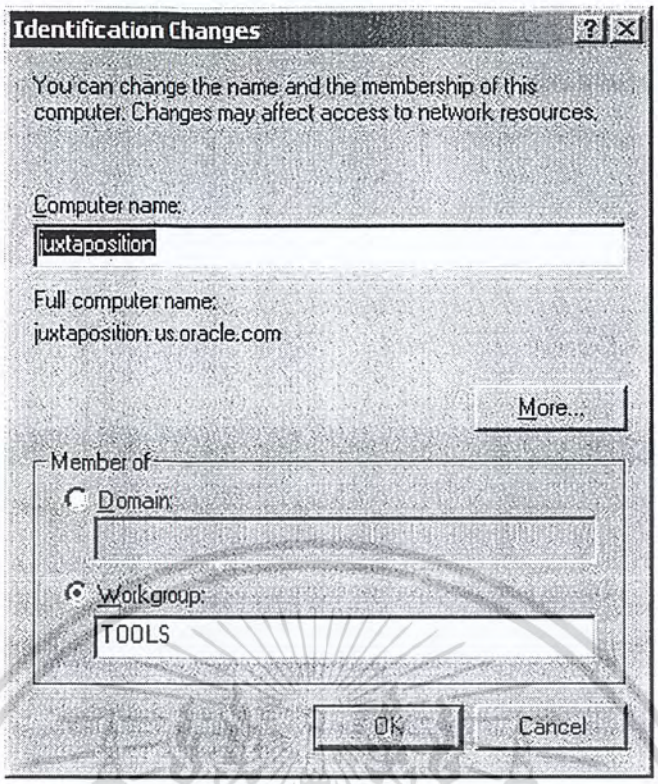
ขั้นตอนการลงโปรแกรม Oracle 8i Database (8.1.7) และ โปรแกรม Oracle 9iAS (1.0.2.1) บนระบบปฏิบัติการ Windows 2000 Server

โครงงานนี้ส่วนหนึ่งเป็นการศึกษาและใช้งานโปรแกรม Oracle9iAS Portal โดยจะต้องลงโปรแกรม Oracle DB และ Oracle Application Server ทางผู้จัดทำเลือกที่จะใช้ Oracle8i Database เวอร์ชัน 8.1.7 และ Oracle9iAS เวอร์ชัน 1.0.2.1 บนระบบปฏิบัติการวินโดวส์ 2000 เซิร์ฟเวอร์ โดยโปรแกรมเหล่านี้สามารถหาดาวน์โหลดได้จาก http://www.oracle.com

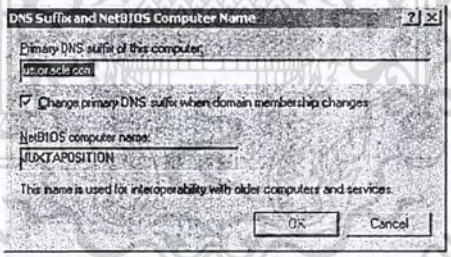
ขั้นตอนในการติดตั้งโปรแกรมมีดังนี้



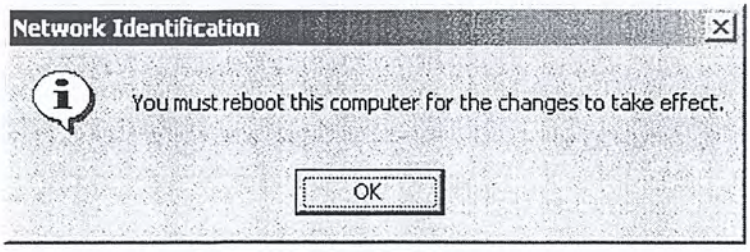
1. ตรวจสอบว่าเครื่อง Server มีการตั้งค่า DNS ไว้เรียบร้อยหรือไม่ โดยเข้าไปที่ Start > Settings > Control Panel > System > Network Identification



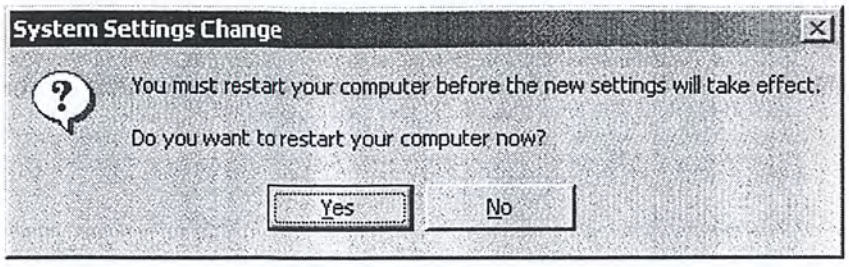
2. เลือก Properties แล้วใส่ชื่อเครื่องในช่อง Computer name เป็นตัวพิมพ์เล็กทั้งหมด และป้อนสมาชิกของ Workgroup ของท่าน แล้วเลือก More



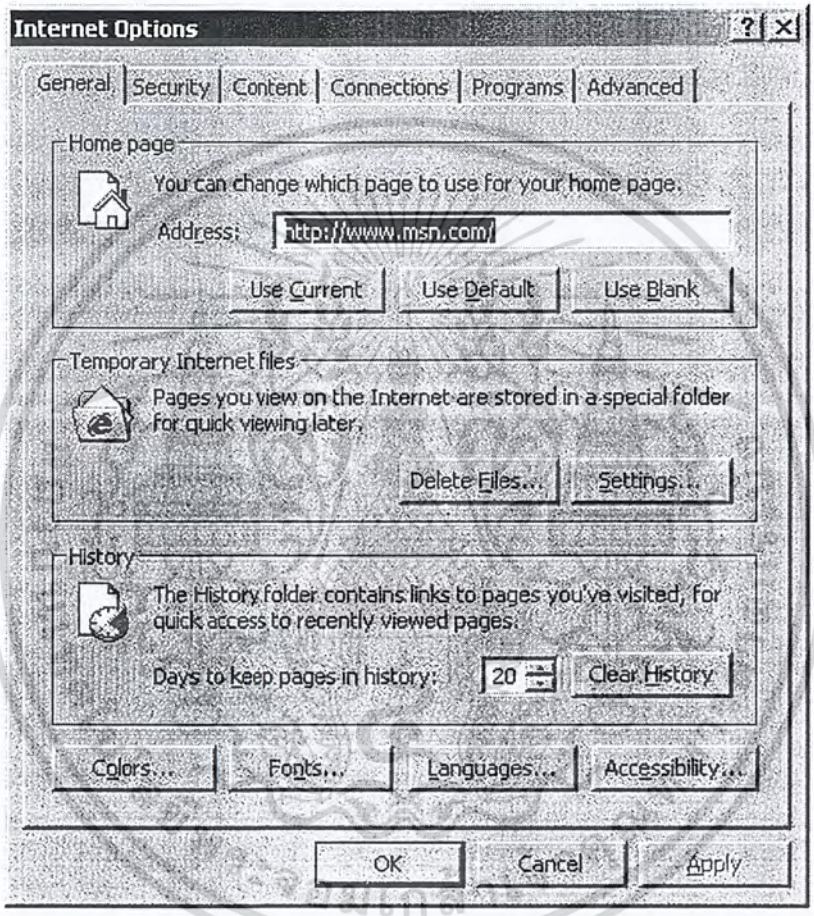
3. DNS suffix ของเครือข่าย “ce.kmitl.ac.th” เป็นตัวพิมพ์เล็กทั้งหมด แล้วเลือก OK

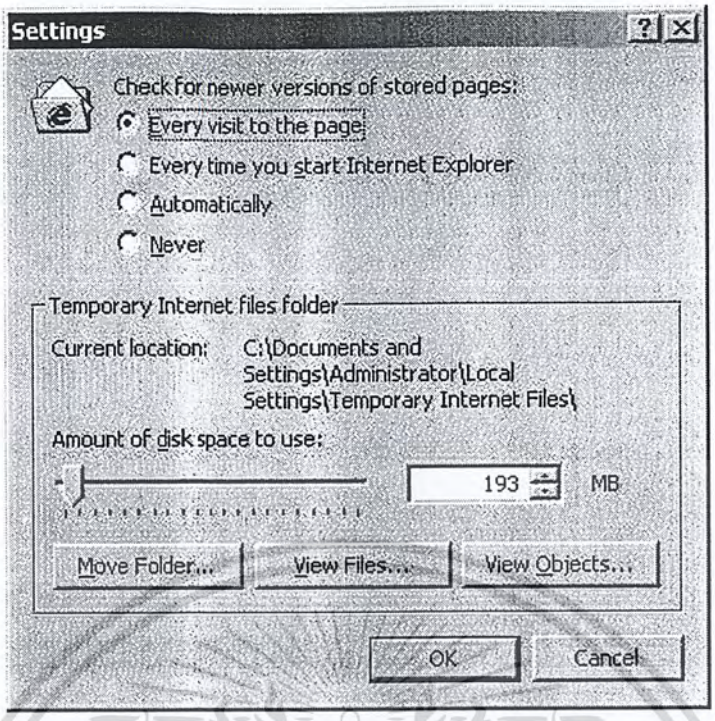


4. เลือก OK จะมีข้อความเตือนให้ restart เครื่อง

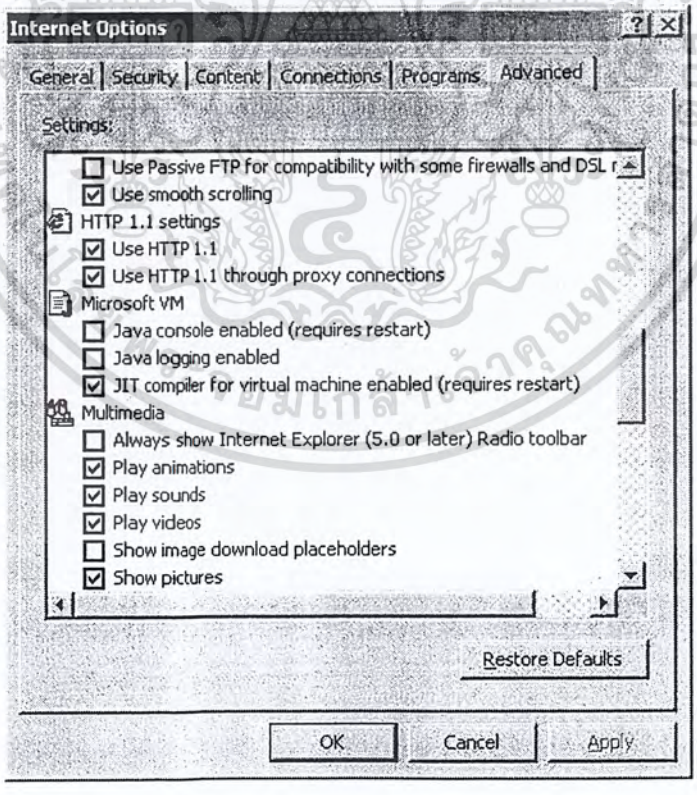


5. เลือก Yes เพื่อ restart เครื่อง

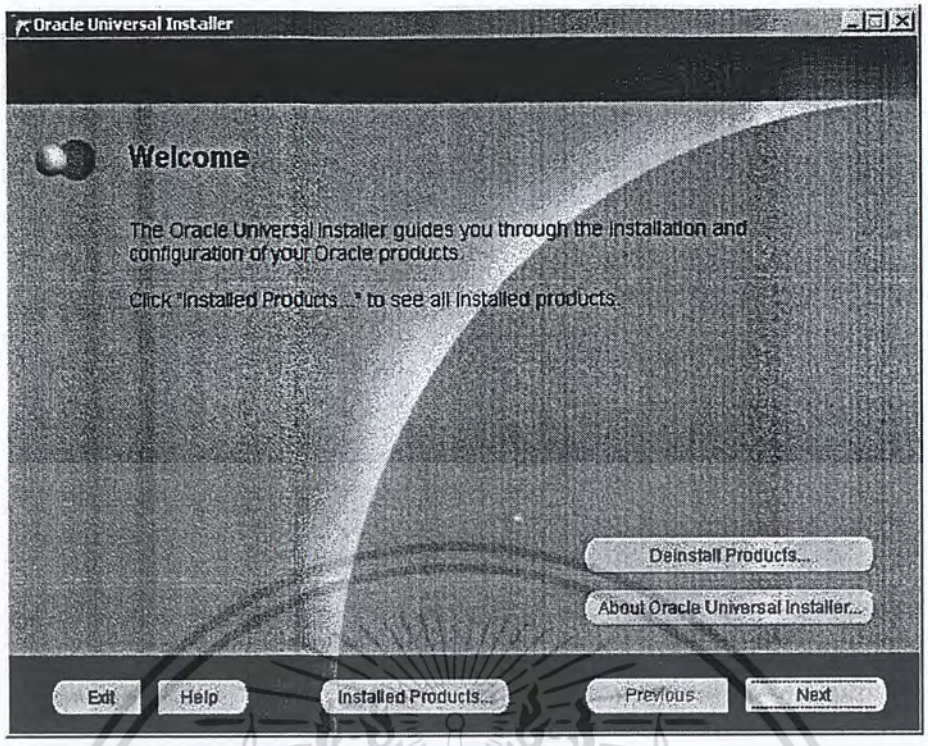




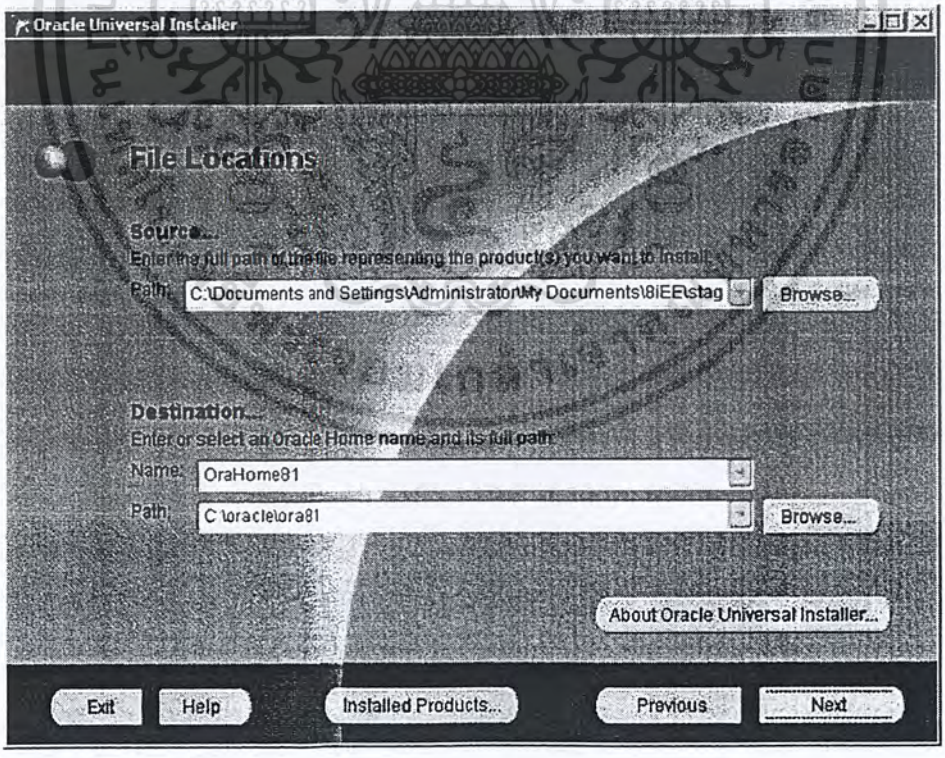
6. สั่งให้เว็บบราวเซอร์ตรวจสอบหน้าเวปเพจทุกครั้งที่เปิด (Every visit to the page) โดยเข้าไปตั้งค่าที่ Start > Control Panel > Internet Options > General > Temporary Internet files Settings



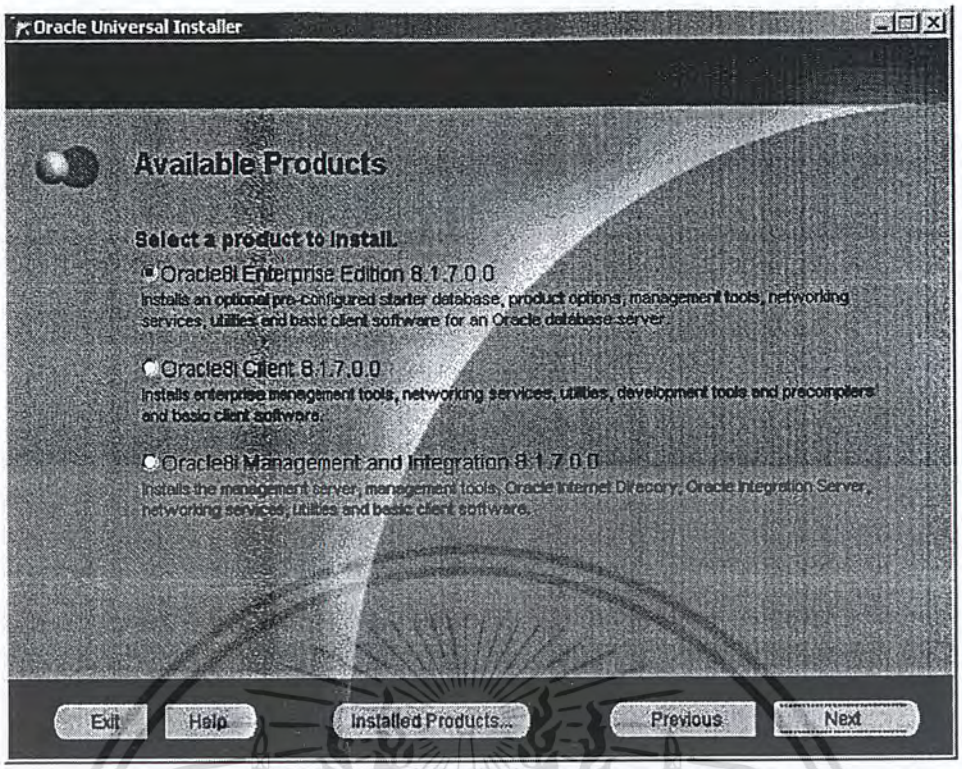
7. ตั้งค่าให้ browser ใช้โพรโตคอล HTTP 1.1 ผ่านทาง proxy (Use HTTP 1.1 through proxy connections) โดยเข้าไปตั้งค่าที่ Start > Control Panel > Internet Options > Advanced > HTTP 1.1 settings



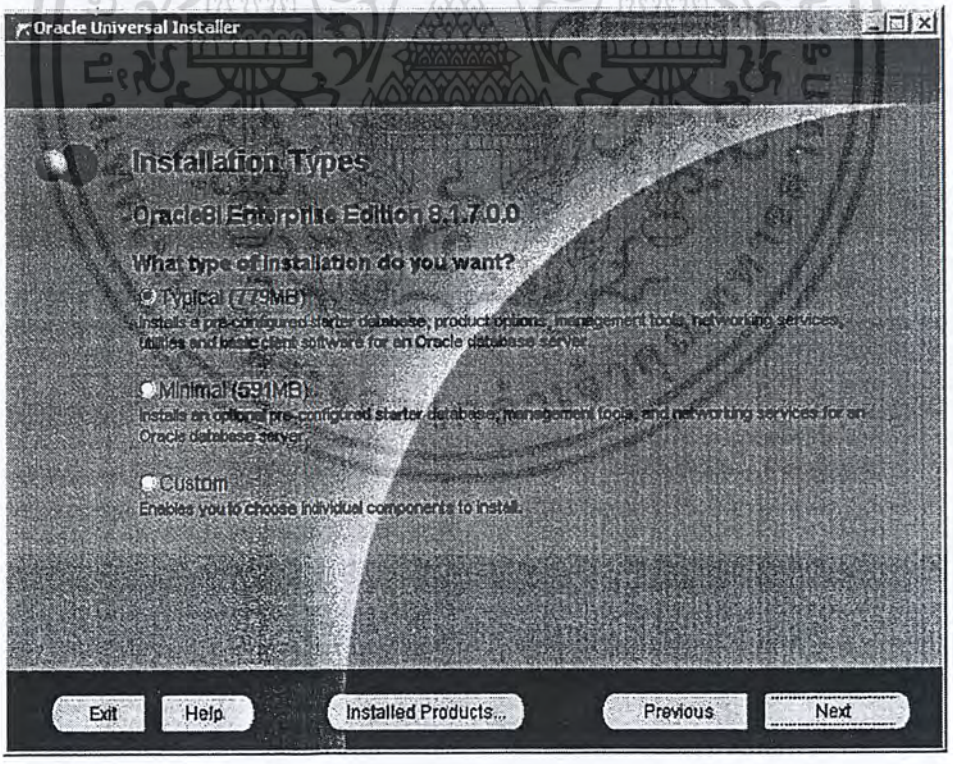
8. เรียกโปรแกรม Oracle Universal Installer ขึ้นมาเพื่อเริ่มลงโปรแกรม Oracle 8i โดยเรียกไฟล์ setup.exe จะเห็นหน้าจอต้อนรับ เลือก Next



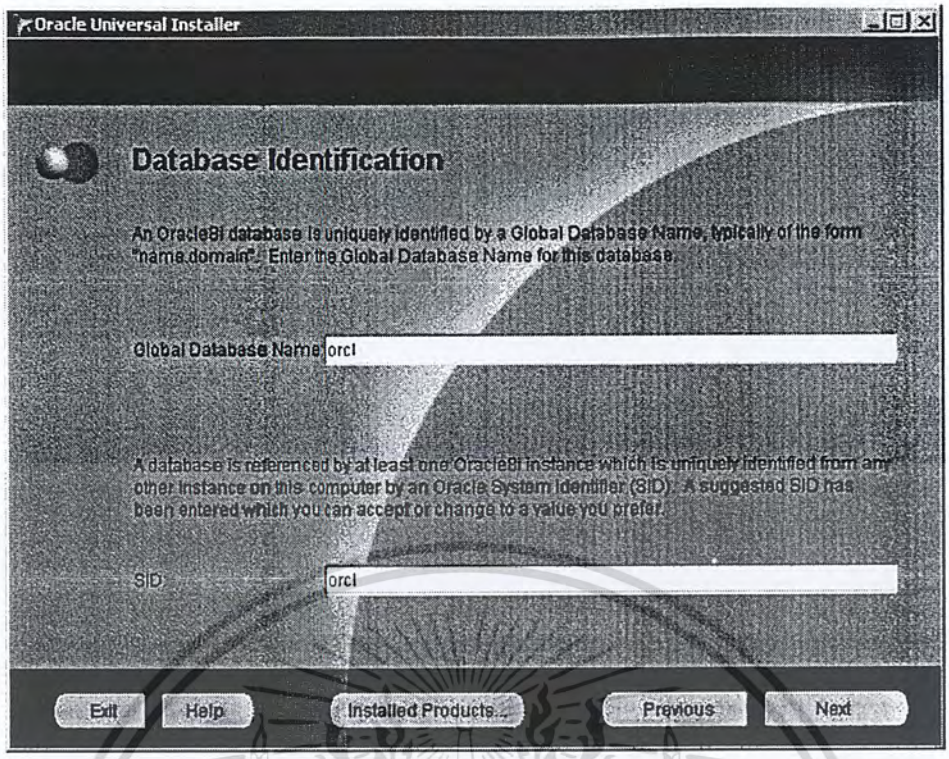
9. ตั้งชื่อ Oracle Home ว่า "OraHome81" และตั้งค่าไดเรกทอรีปลายทางที่จะเก็บโปรแกรมว่า "C:\Oracle\ora81" แล้วเลือก Next



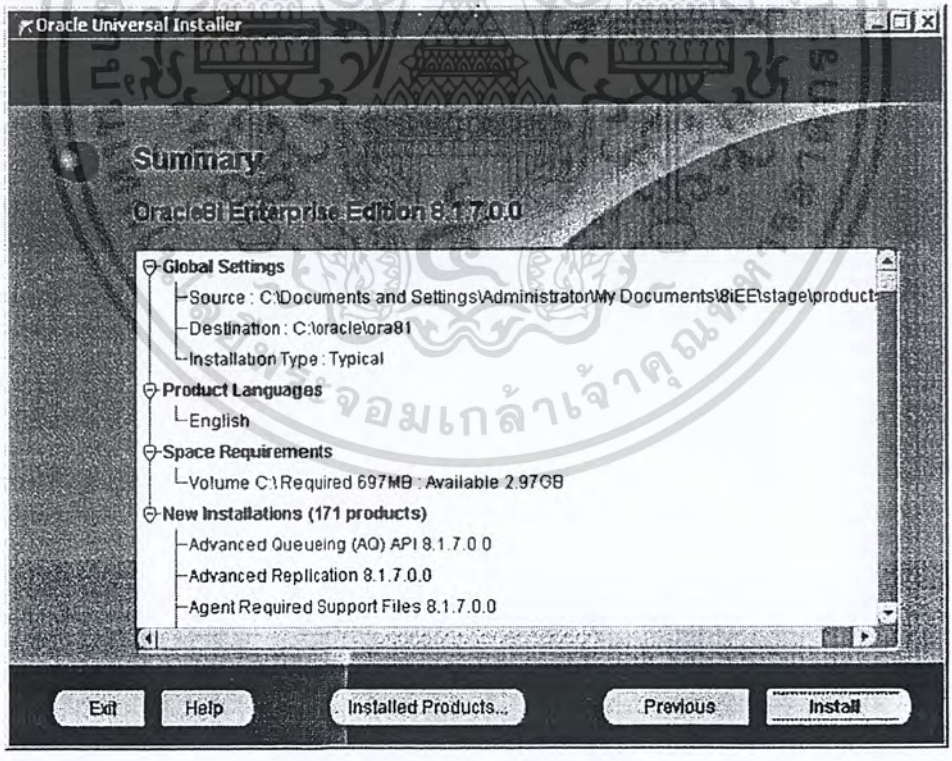
10. เลือกติดตั้งโปรแกรมแบบ Enterprise Edition แล้วเลือก Next



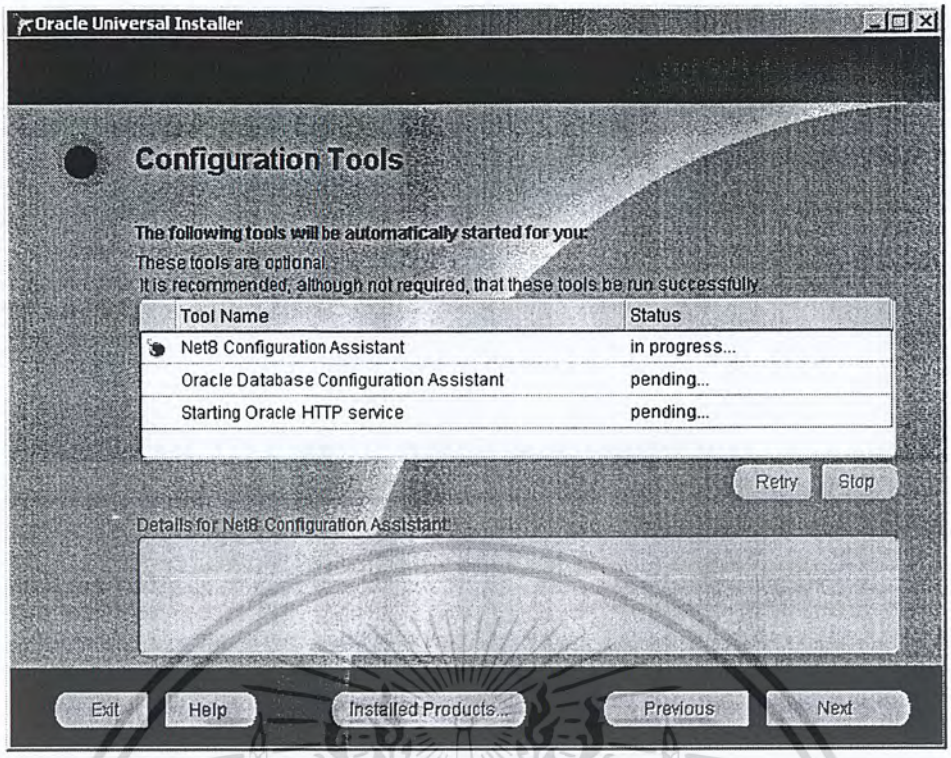
11. เลือกชนิดการติดตั้งแบบ Typical แล้วเลือก Next



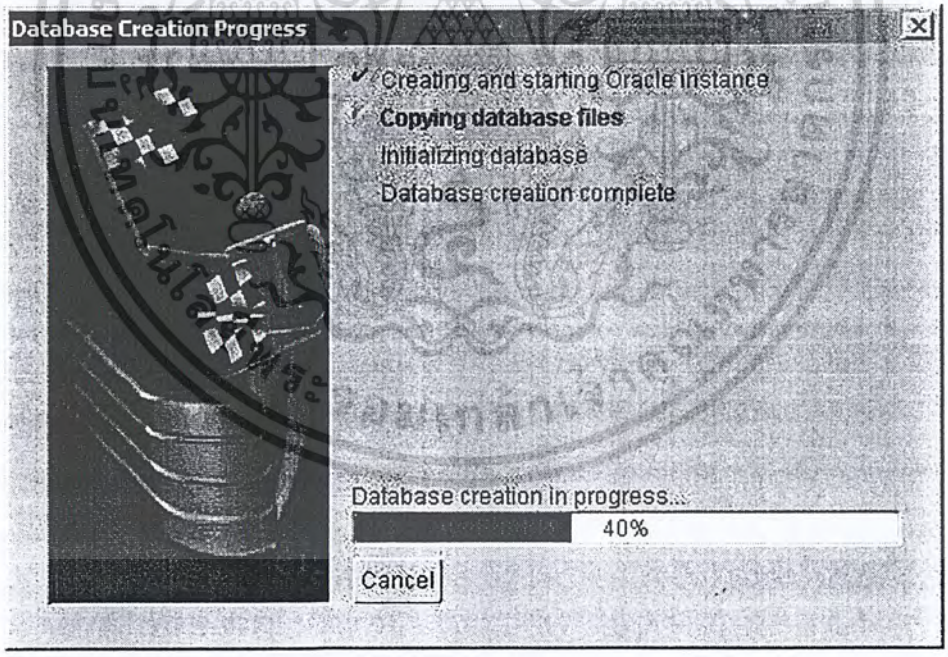
12. ตั้งชื่อฐานข้อมูลทั้ง Global Database Name และ SID ว่า "orcl" แล้วเลือก Next

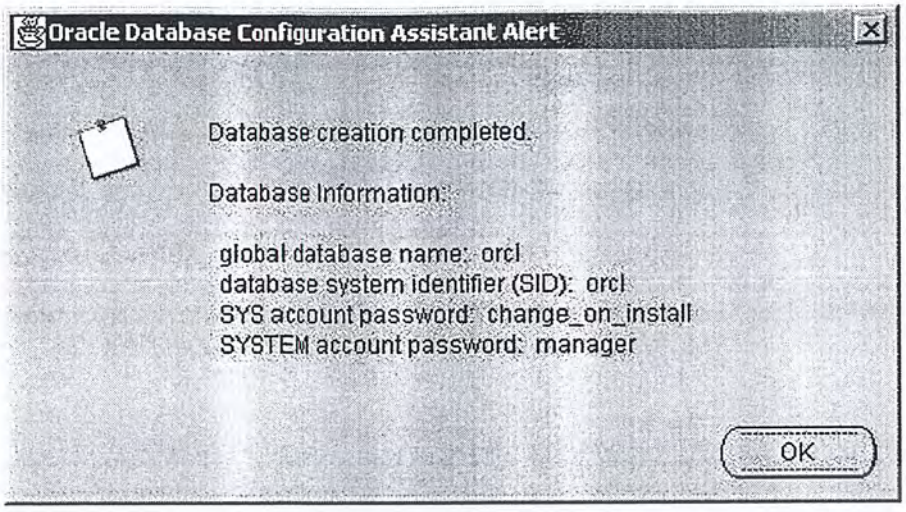


13. จะเห็นหน้าจอสรุปโปรแกรมที่จะต้องติดตั้ง เลือก Install

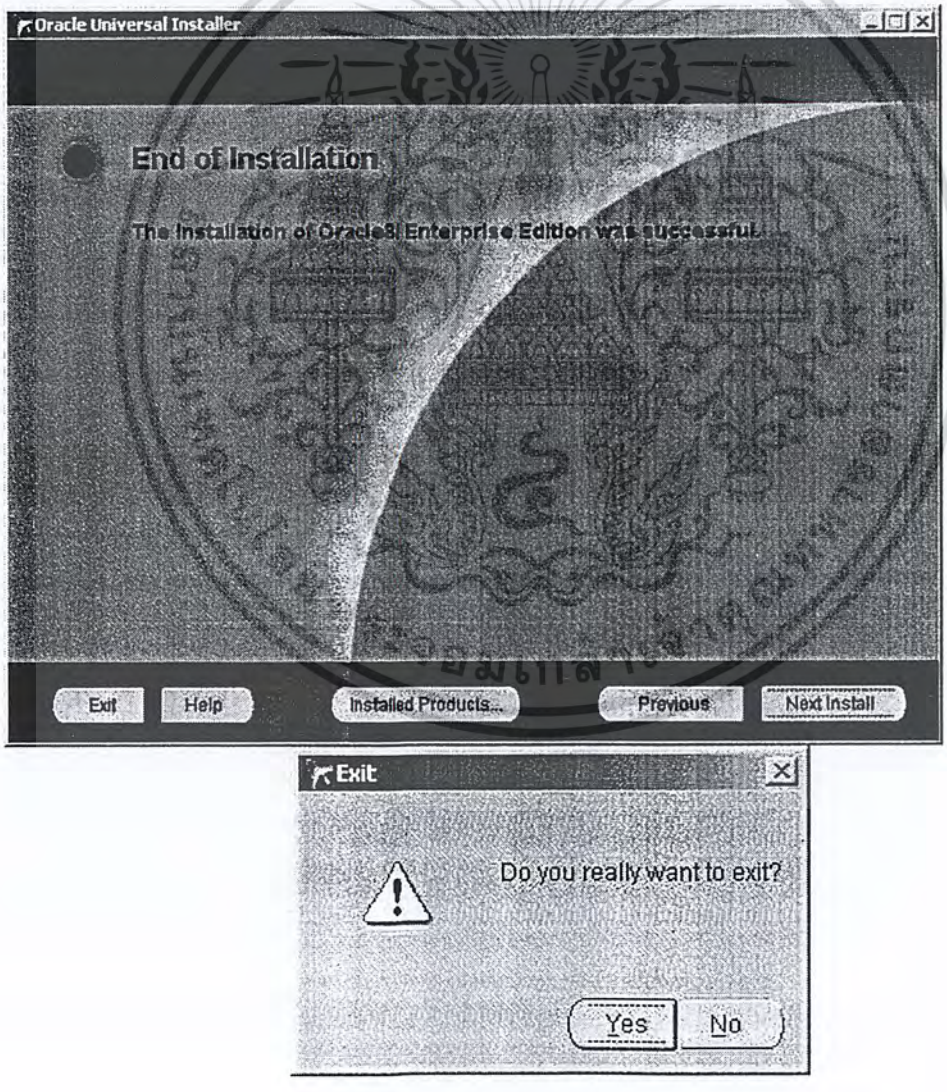


14. เมื่อขึ้นหน้าจอ Configuration Tools โปรดรอสักครู่

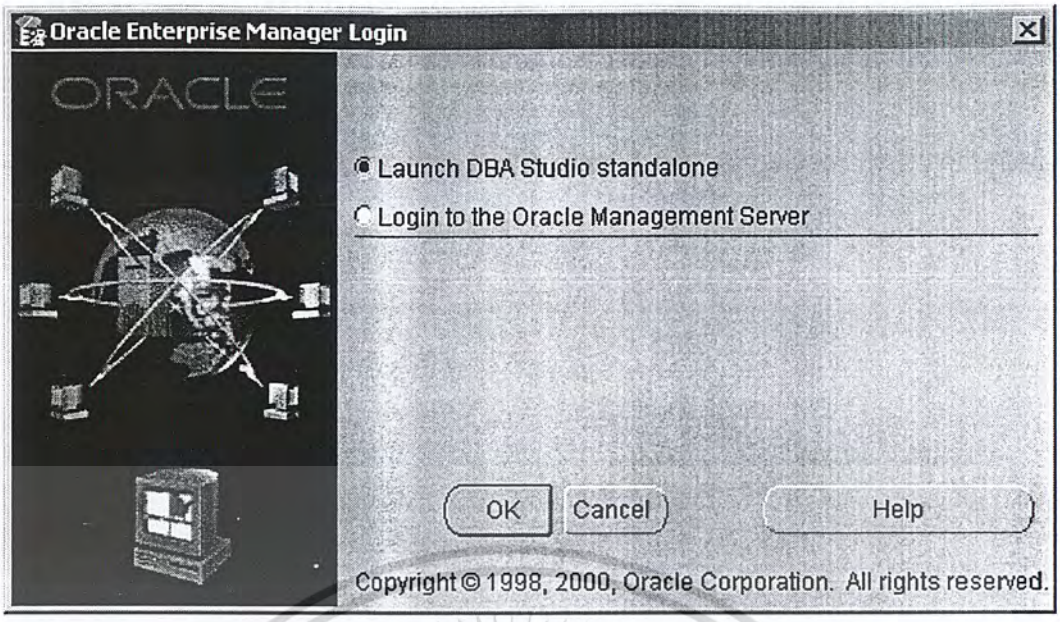




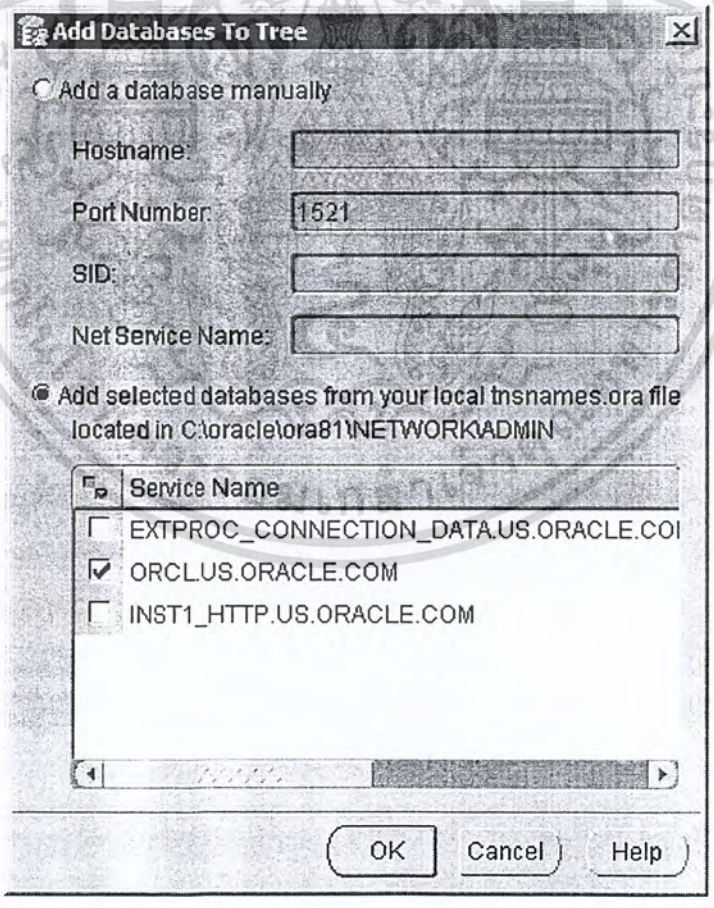
15. หน้าจอยืนยันรหัสผ่านในการเข้าใช้ฐานข้อมูล เลือก OK



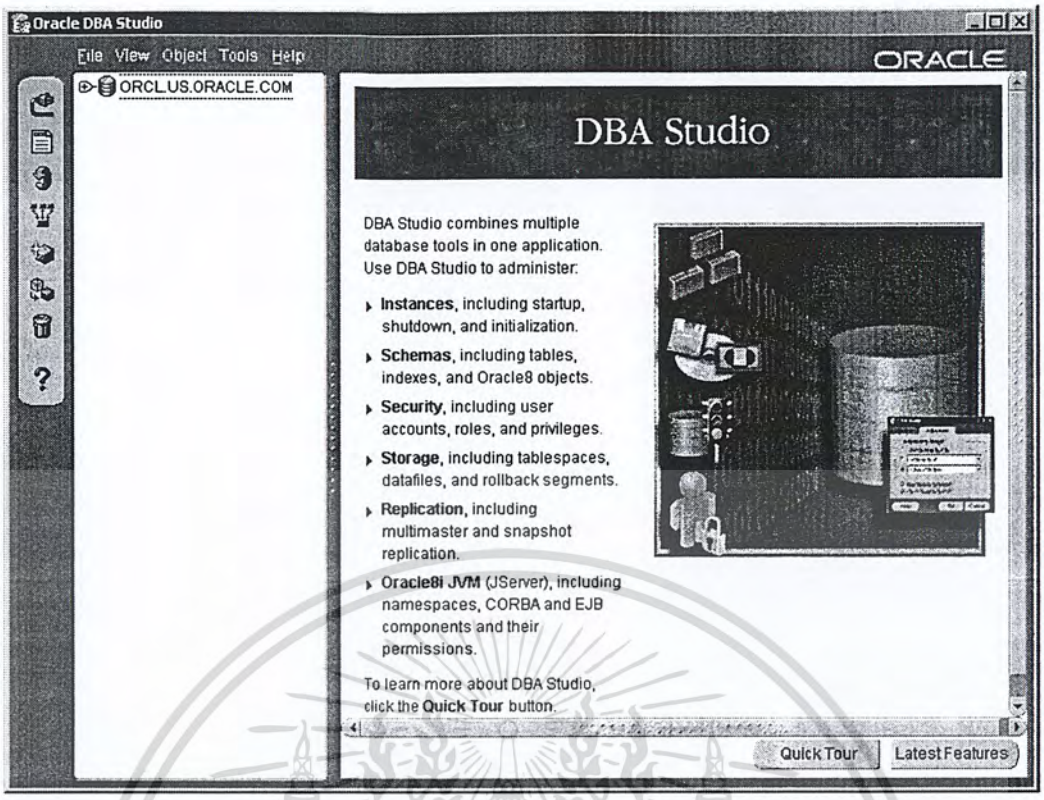
16. หน้าจอสิ้นสุดการลงโปรแกรม เลือก Yes



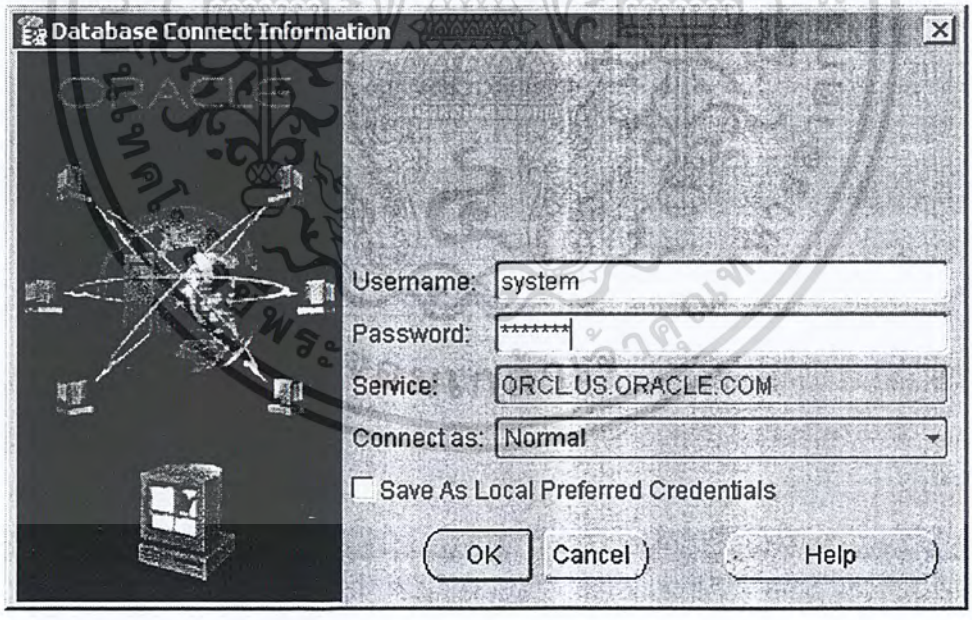
17. เปิดโปรแกรม Oracle DBA Studio โดยเลือกที่ Start > Programs > Oracle > OraHome81 > Database Administration > DBA Studio



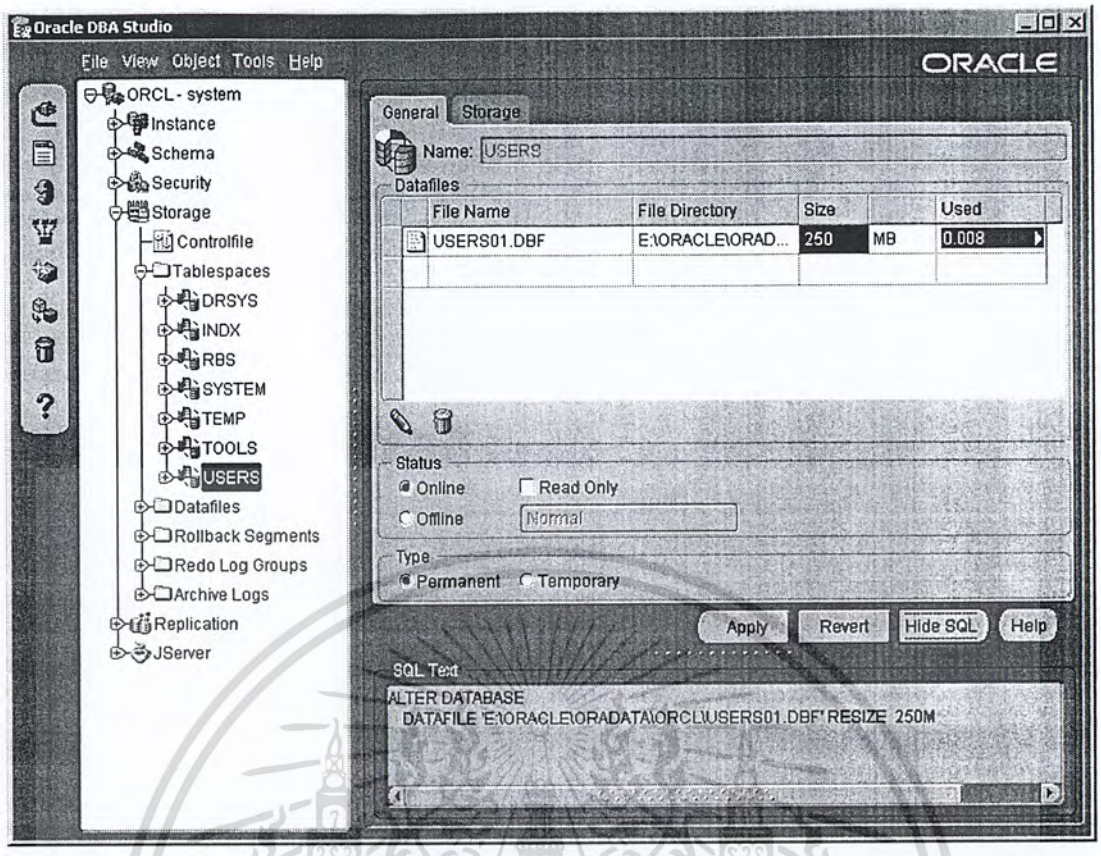
18. เลือก database "orcl" แล้วเลือก OK



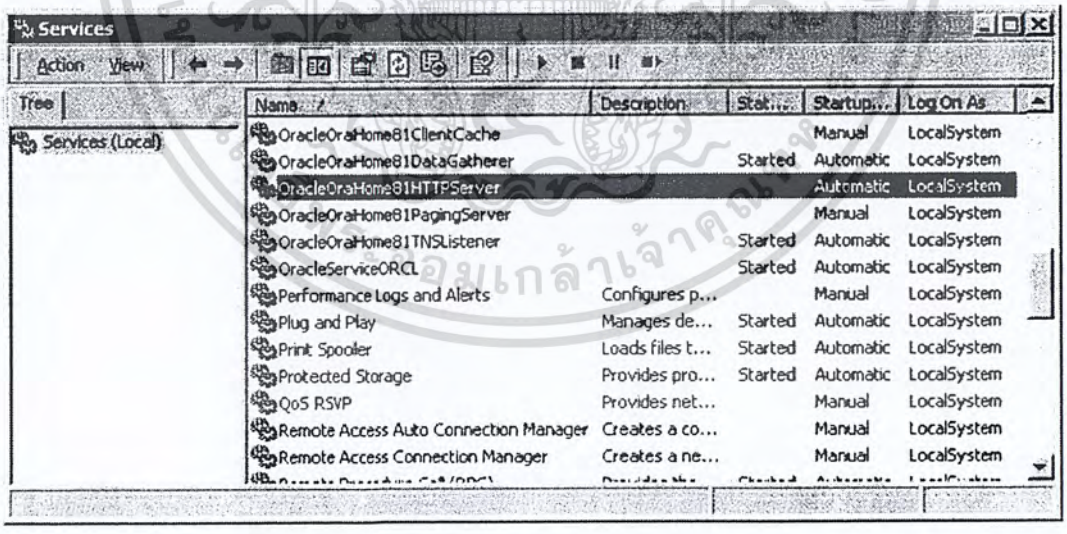
19. เลือก + หน้า orcl database



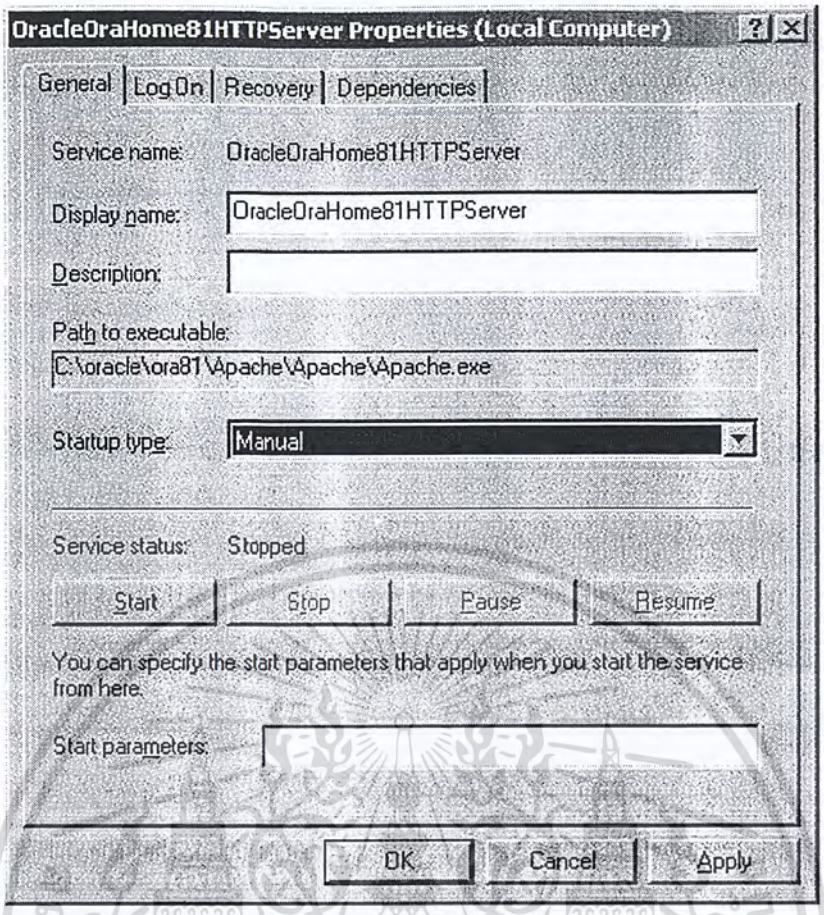
20. ใส่ username "system" และ password "manager" แล้วเลือก OK



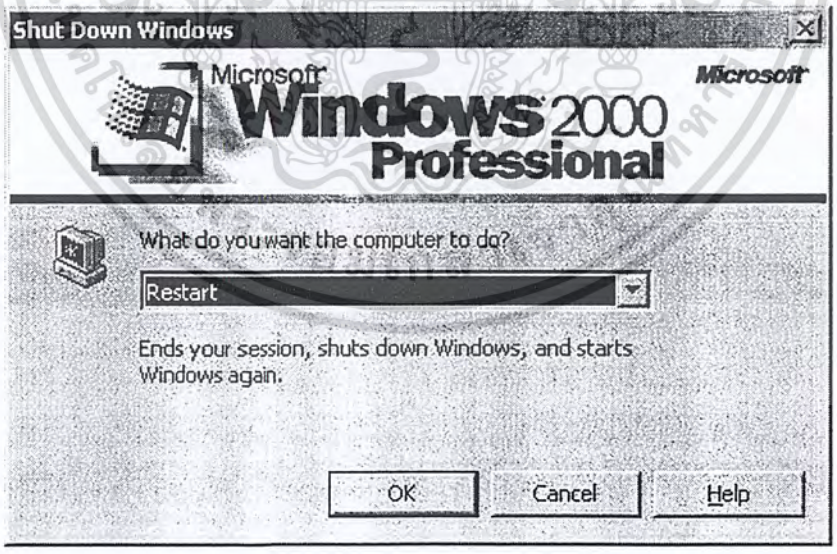
21. เลือก Storage > Tablespaces > User แล้วปรับขนาดเป็น 250 MB แล้วเลือก Apply เสร็จแล้วปิดโปรแกรม Oracle DBA Studio



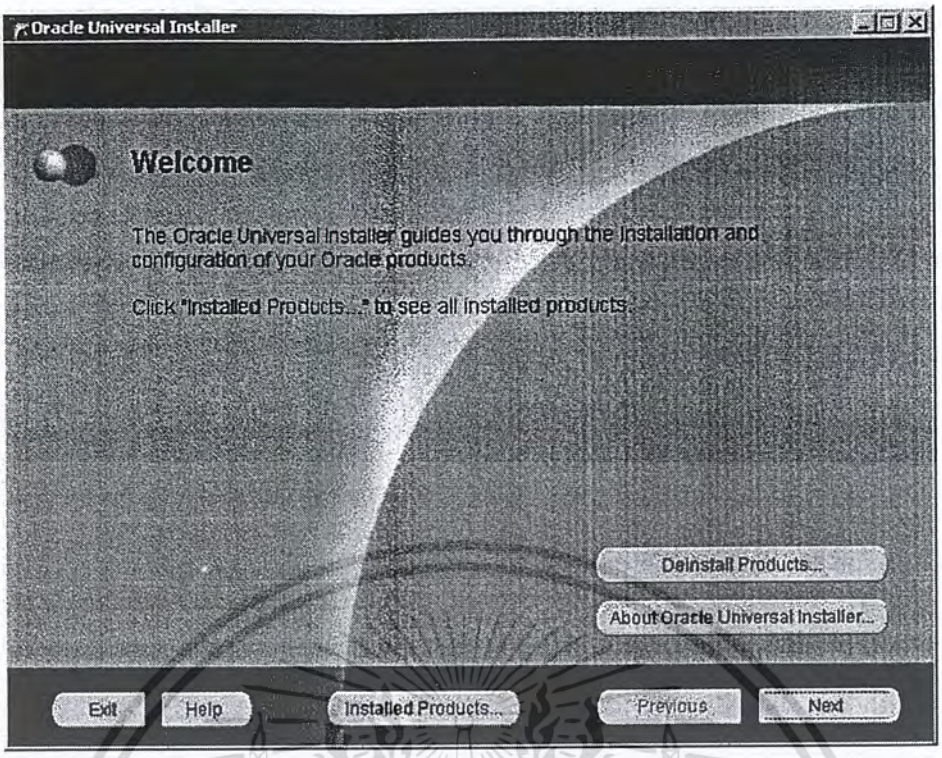
22. ตั้งค่า service HTTP ของ Oracle 8i เป็น Manual โดยเข้าไปที่ Start > Settings > Control Panel > Administrative Tools > Services แล้ว double click ที่ service OracleOraHome81HTTPServer



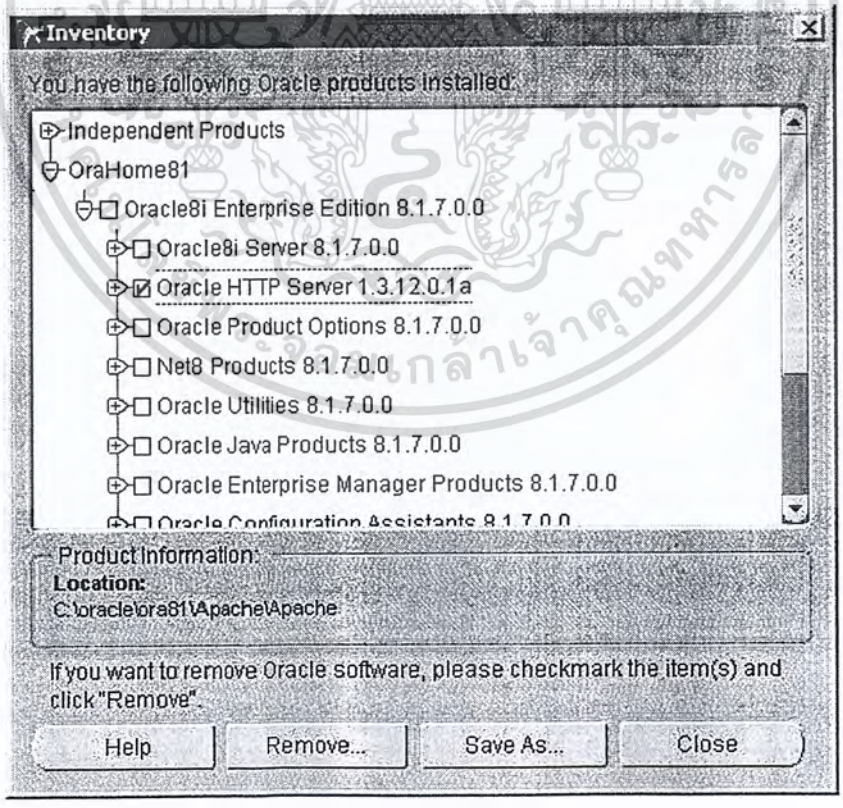
23. เปลี่ยน Startup type จาก Automatic เป็น Manual แล้วเลือก OK



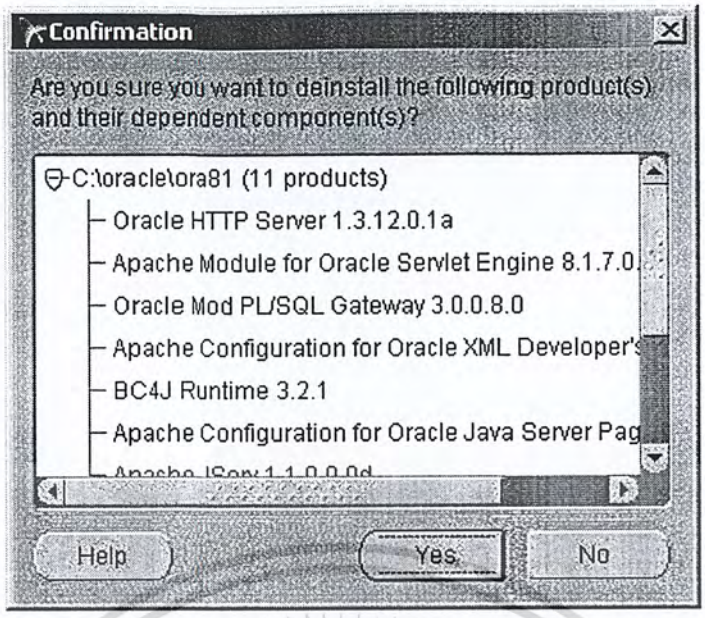
24. เลือก Start > Shut Down แล้วเลือก Restart เครื่องเพื่อให้ค่าที่ตั้งไว้ทำงาน



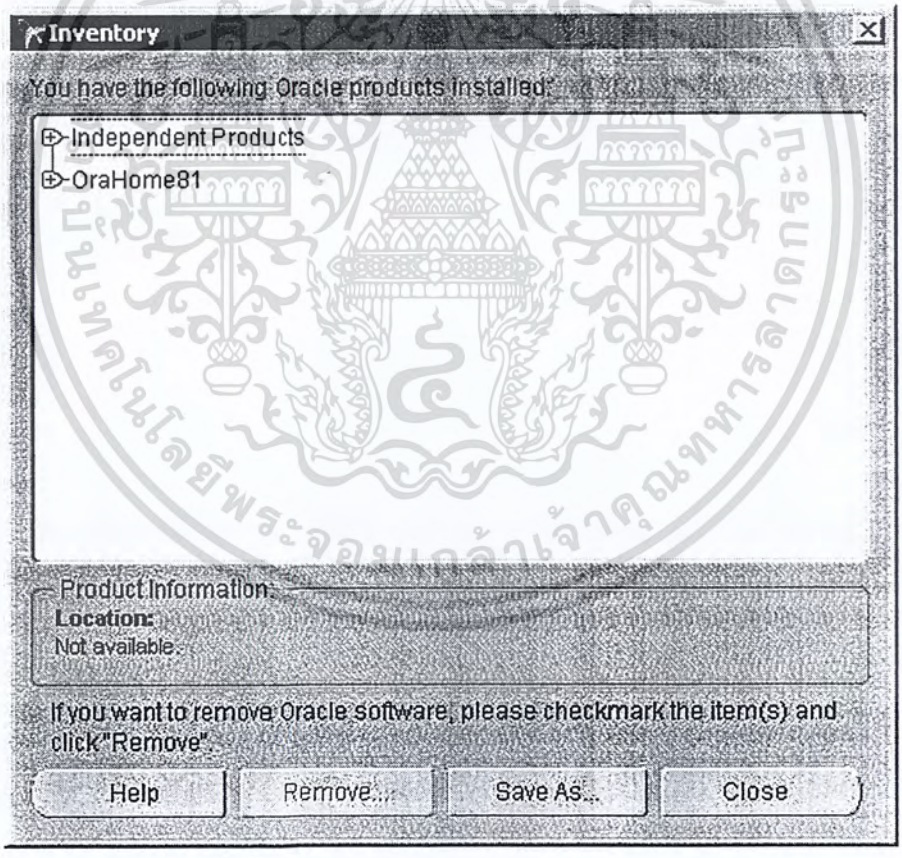
25. เข้าไปที่ Start > Programs > Oracle Installation Products > Universal Installer เรียกโปรแกรม Oracle Universal Installer แล้วเลือก Deinstall Products



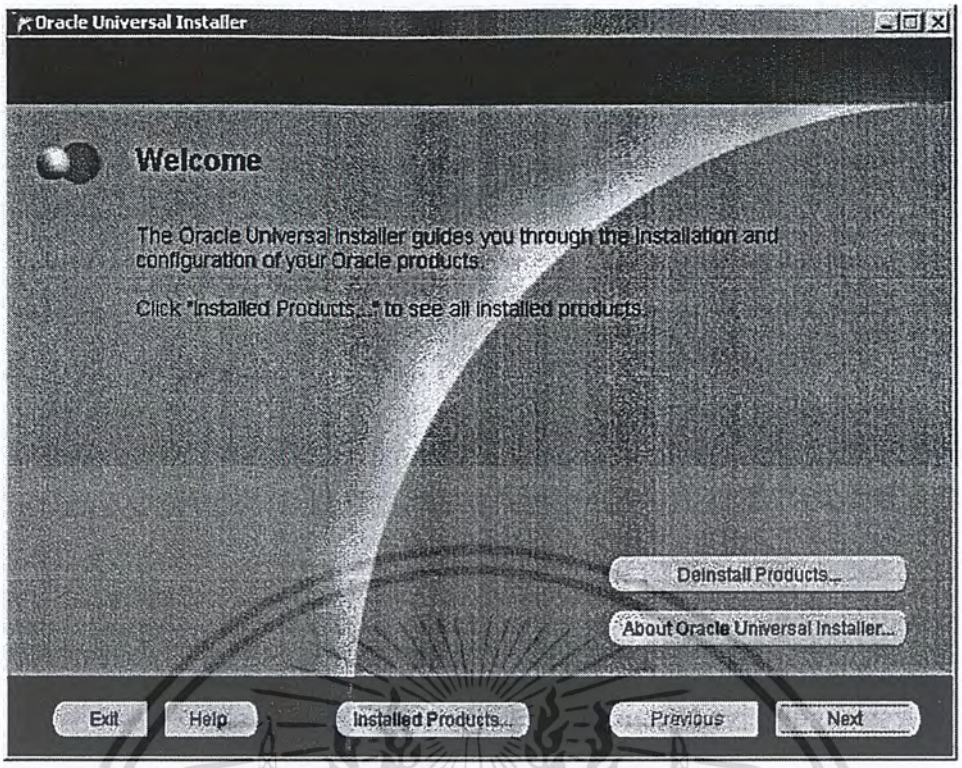
26. เลือก Oracle HTTP Server 1.3.12.0.1a แล้วเลือก Remove



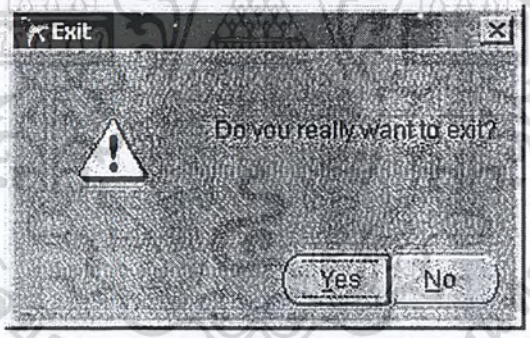
27. เลือก Yes



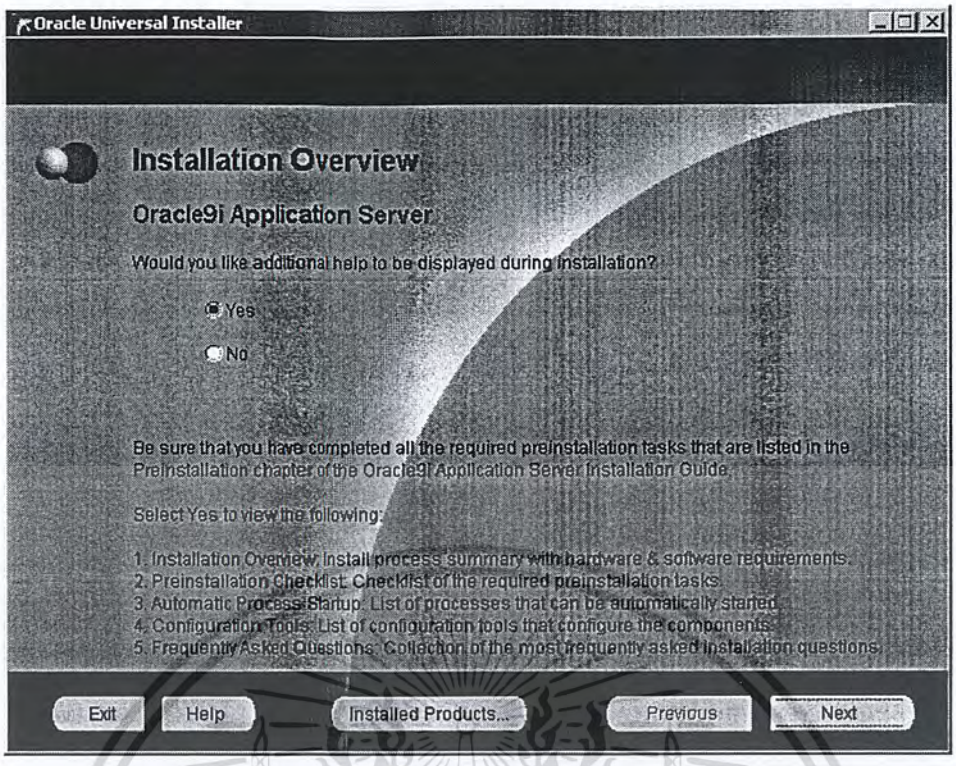
28. เลือก Close



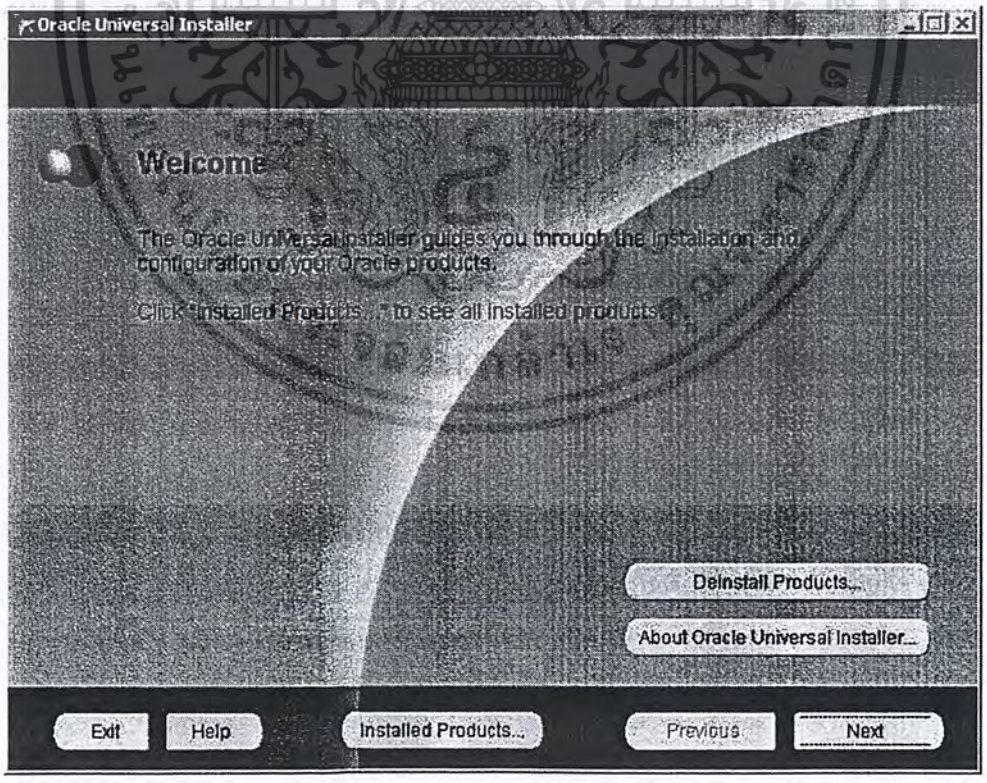
29. เลือก Exit



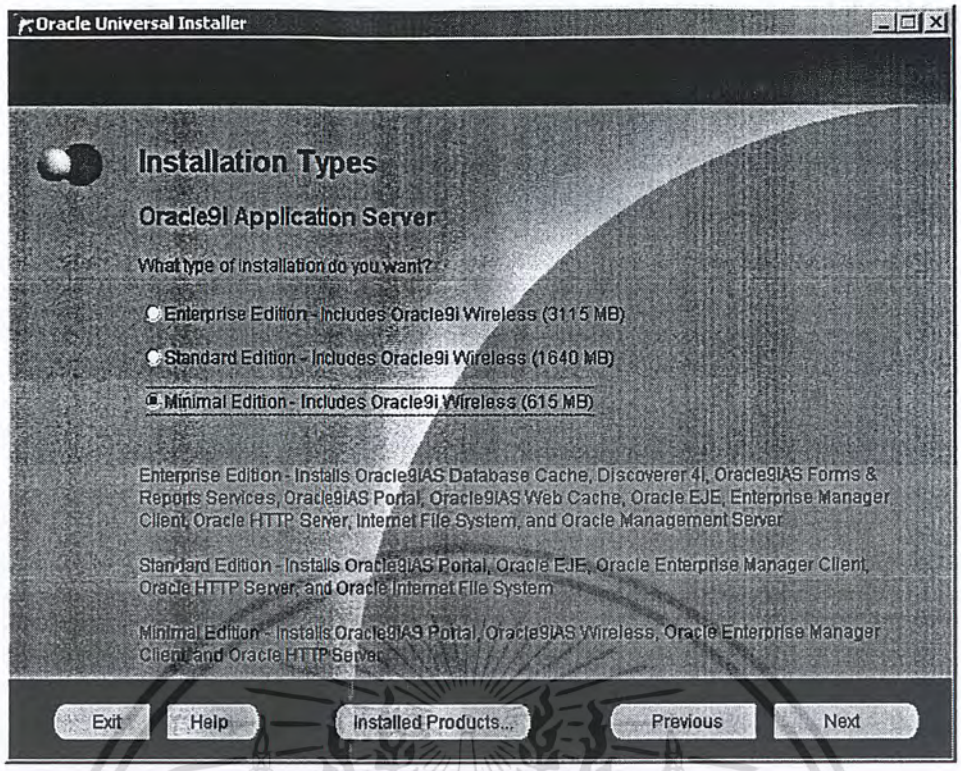
30. เลือก Yes



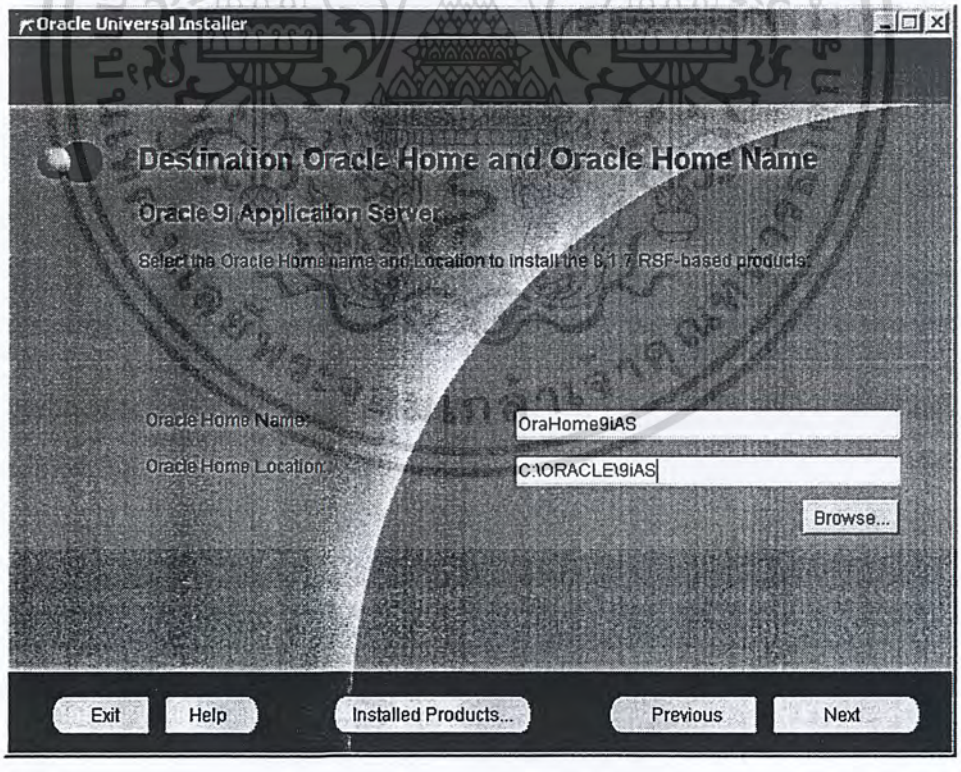
31. เรียกโปรแกรม Oracle Universal Installer ขึ้นมาเพื่อเริ่มลงโปรแกรม Oracle 9iAS โดยเรียกไฟล์ setup.exe จะเห็นหน้าจอต้อนรับ เลือก No แล้วเลือก Next



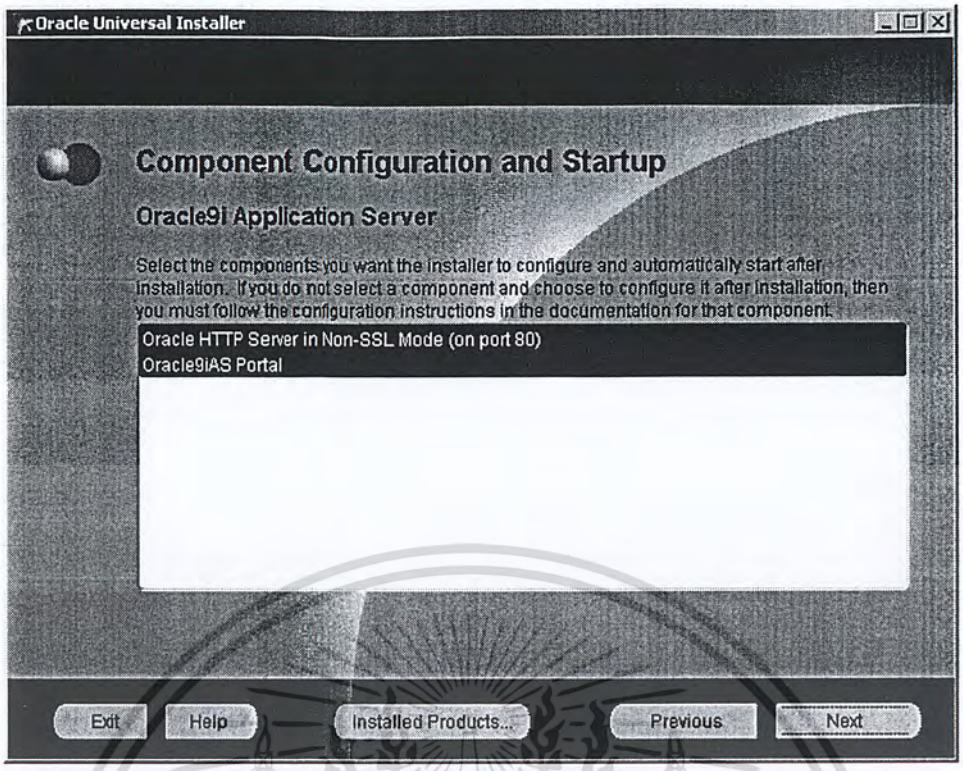
32. เลือก Next



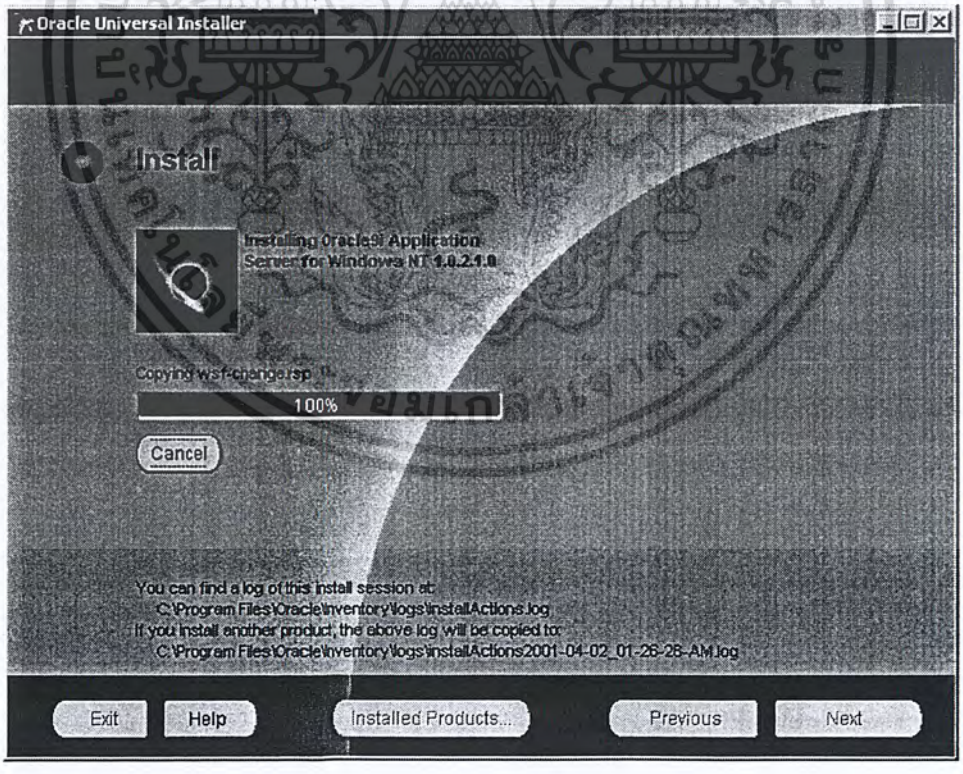
33. เลือกติดตั้งแบบ Minimal Edition แล้วเลือก Next



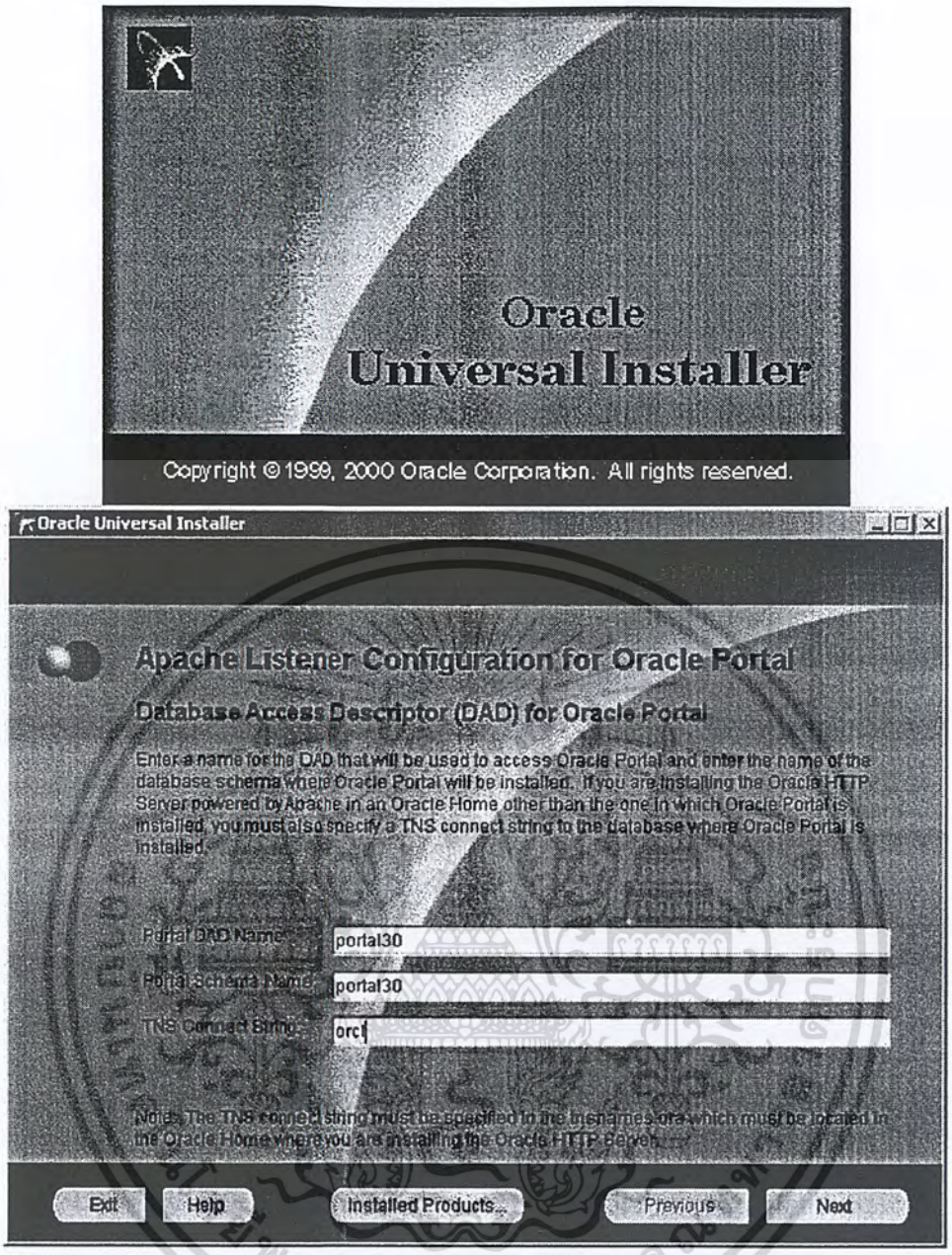
34. ตั้งค่า Oracle Home Name ว่า "OraHome9iAS" และตั้งค่า Oracle Home Location ว่า "C:\Oracle\9iAS"



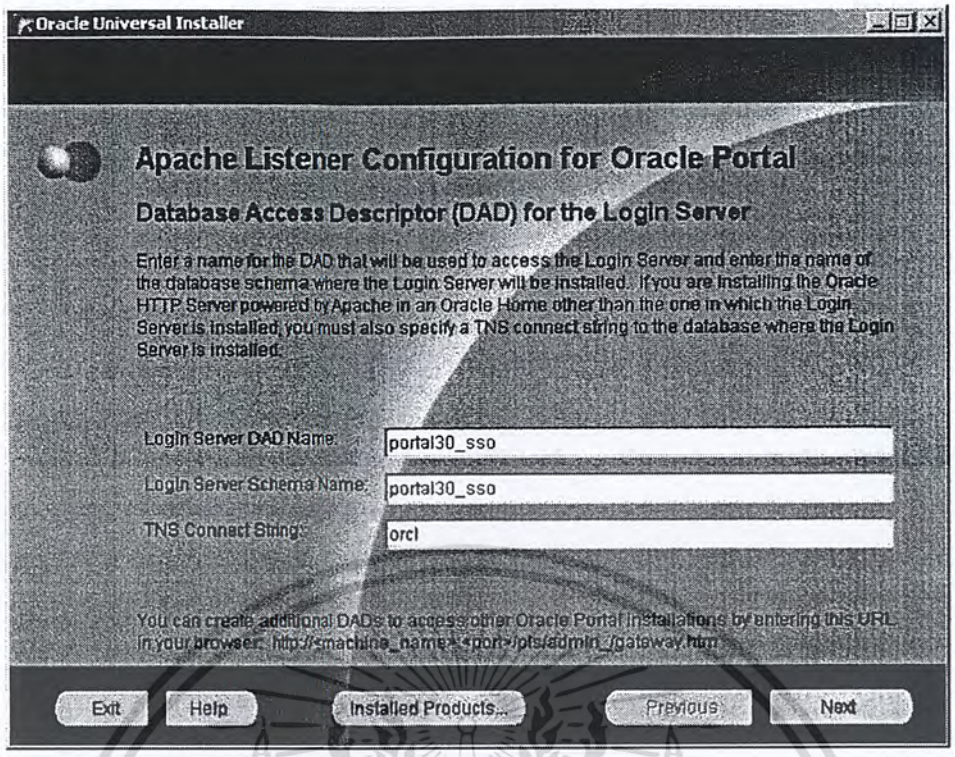
35. เลือก Oracle HTTP Server และ Oracle 9iAS Portal แล้วเลือก Next



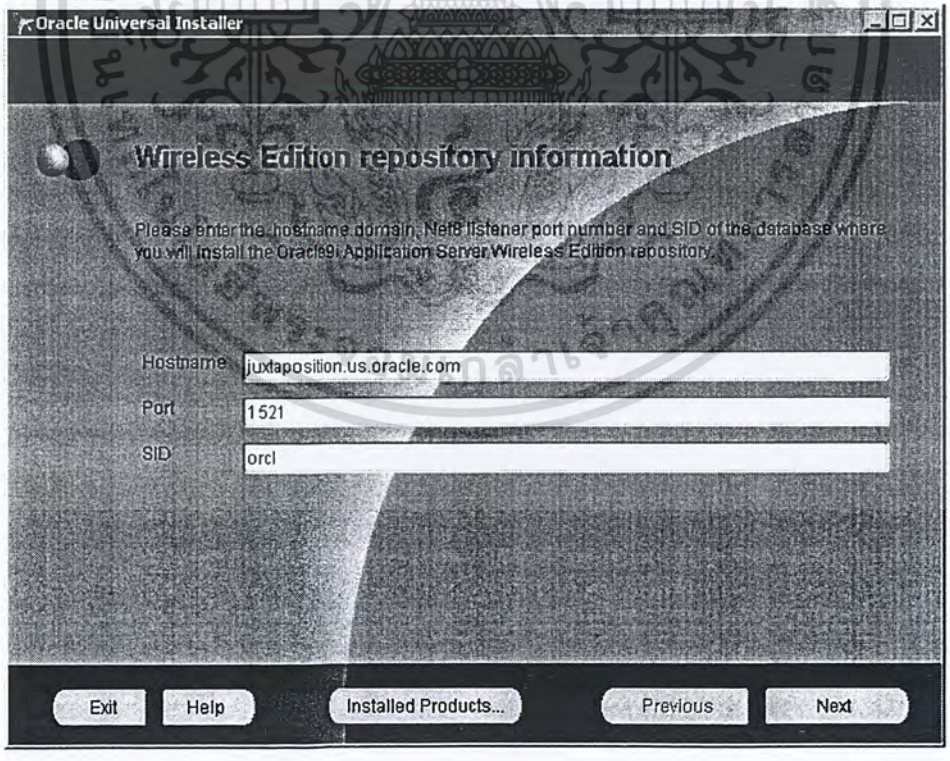
36. เลือก Next



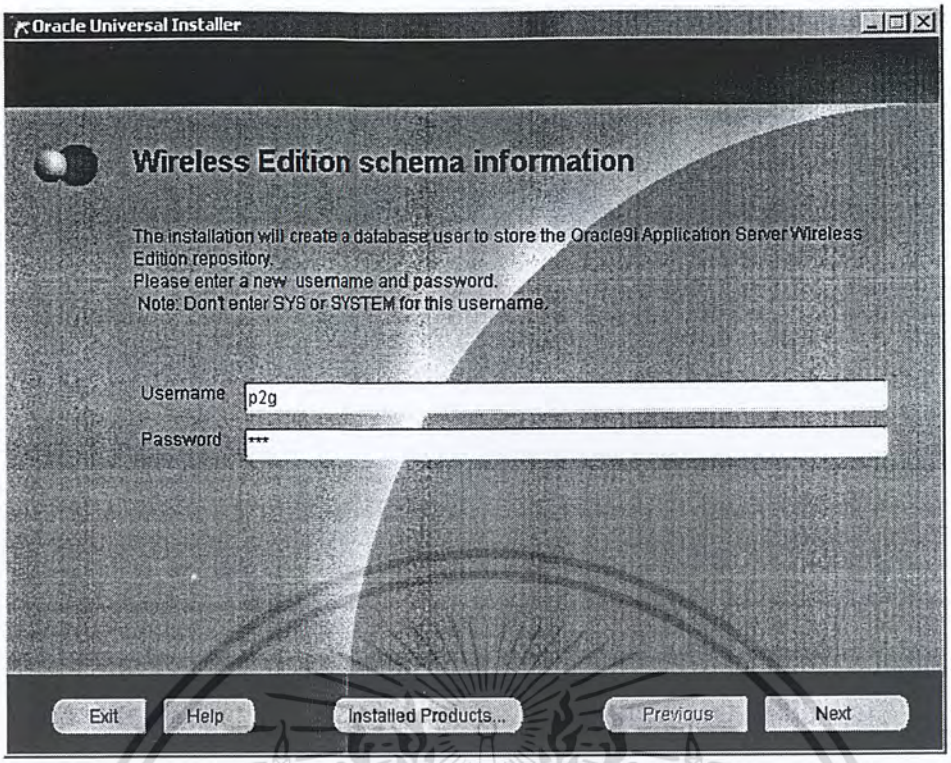
37. ตั้งค่า Portal DAD Name และ Portal Schema Name ว่า “portal30” และตั้งค่า TNS Connect String ว่า “orcl” แล้วเลือก Next



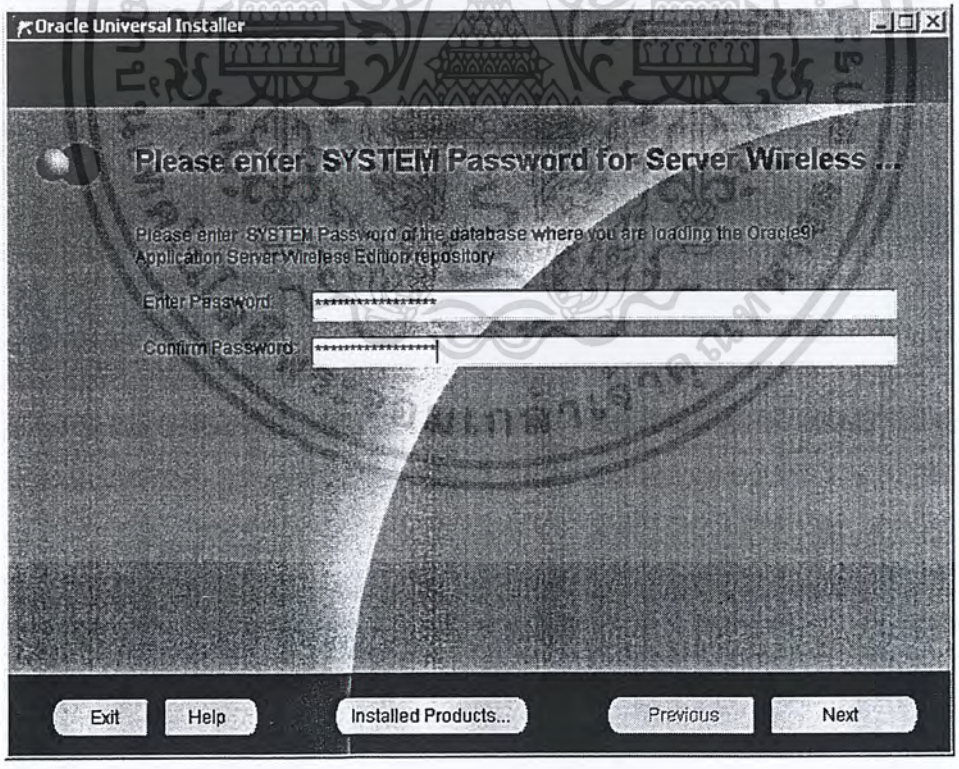
38. ตั้งค่า Login Server DAD Name และ Login Server Schema Name ว่า "portal30_sso" และตั้งค่า TNS Connect String ว่า "orcl" แล้วเลือก Next



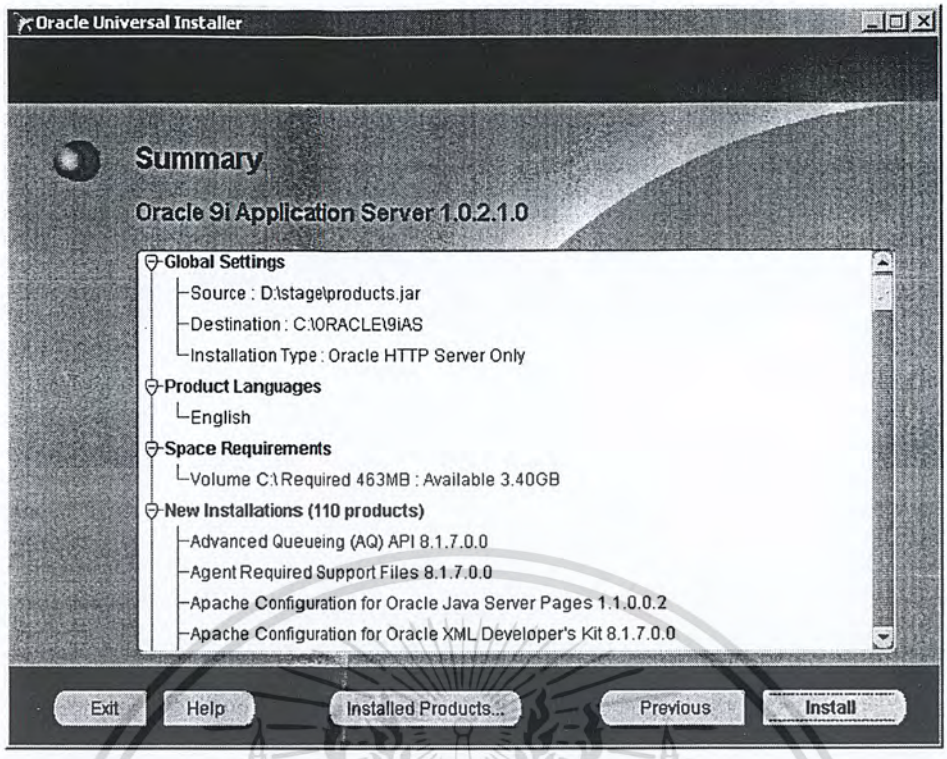
39. ตั้งค่า Hostname ว่า "network20.ce.kmitl.ac.th" และตั้งค่า Port ว่า "1521" และตั้งค่า SID ว่า "orcl" แล้วเลือก Next



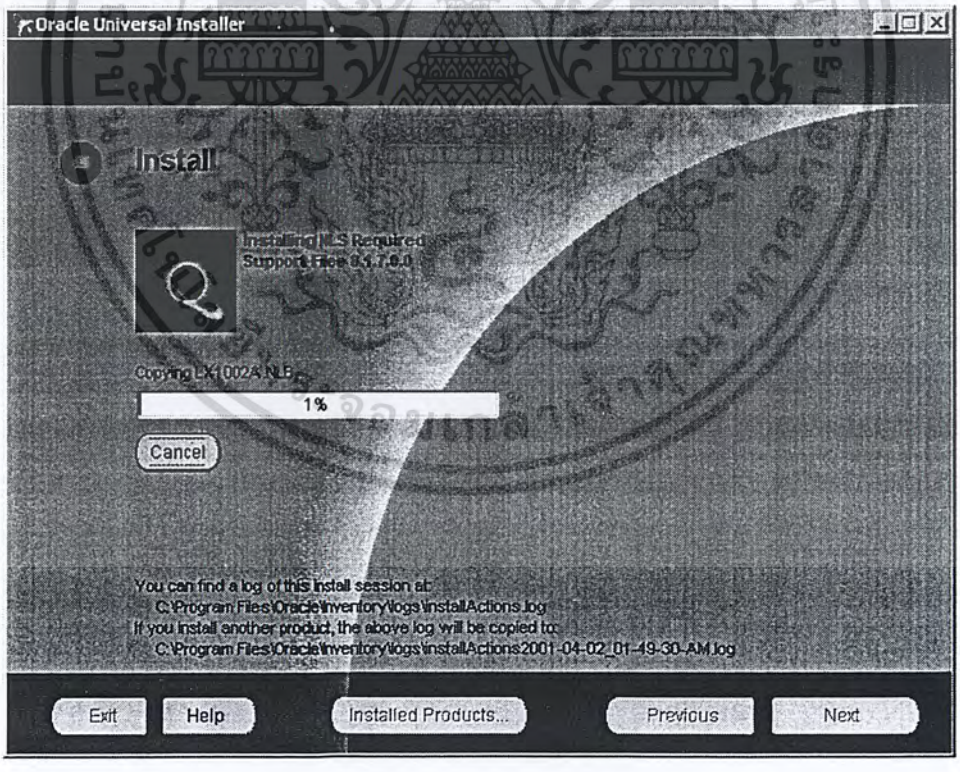
40. ตั้งค่า Username และ Password ว่า "p2g" แล้วเลือก Next

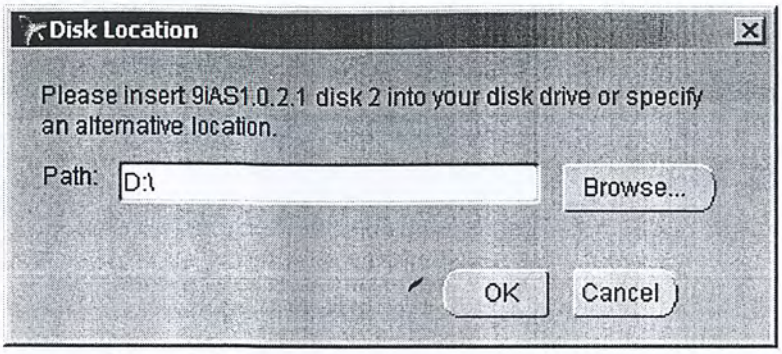


41. ใส่ password ทั้งสองช่องว่า "manager" แล้วเลือก Next

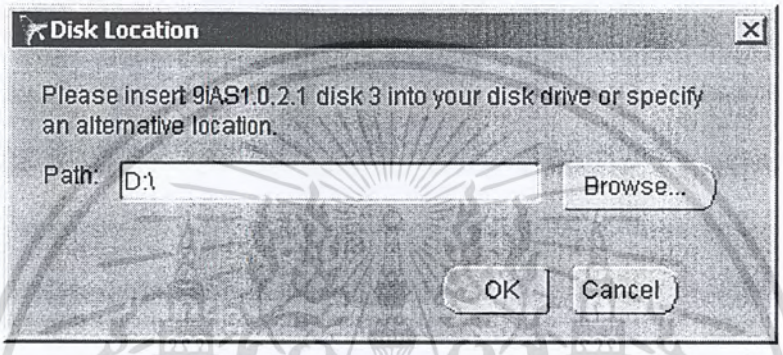


42. เลือก Install

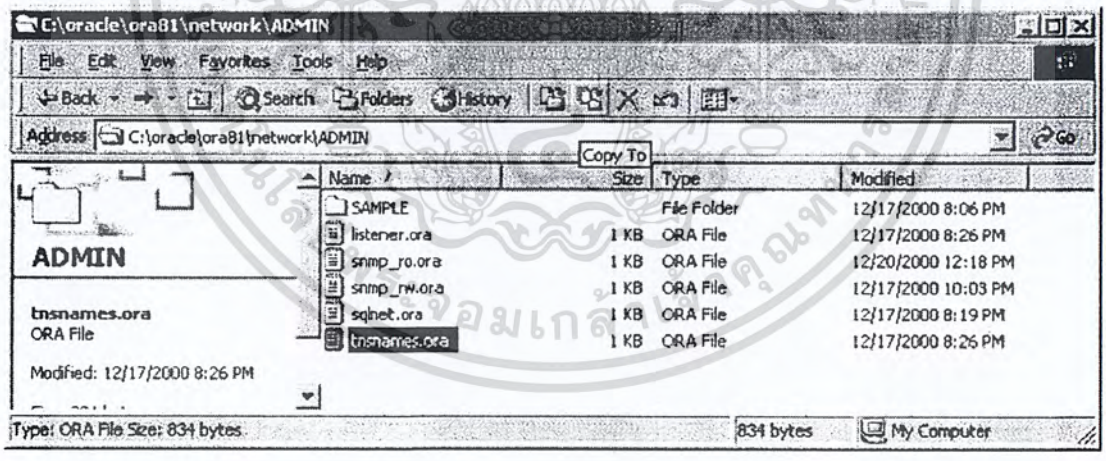


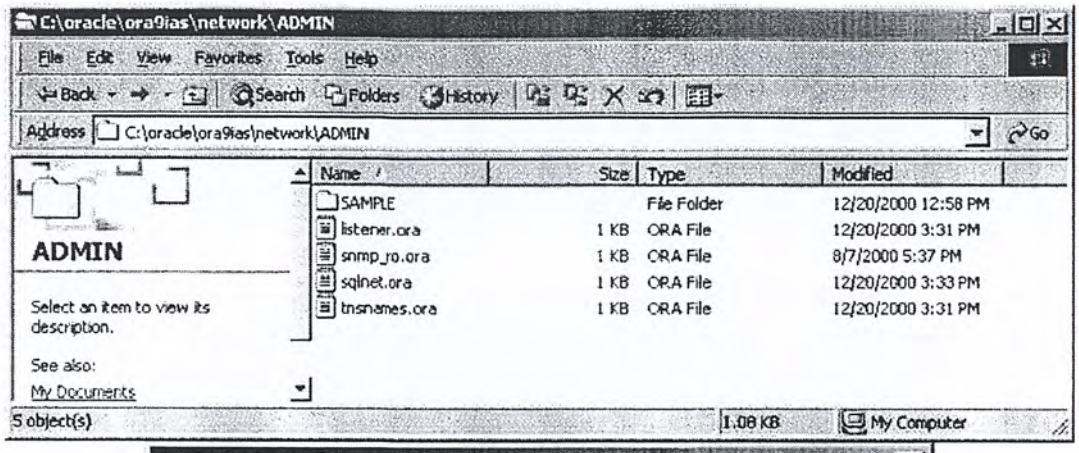


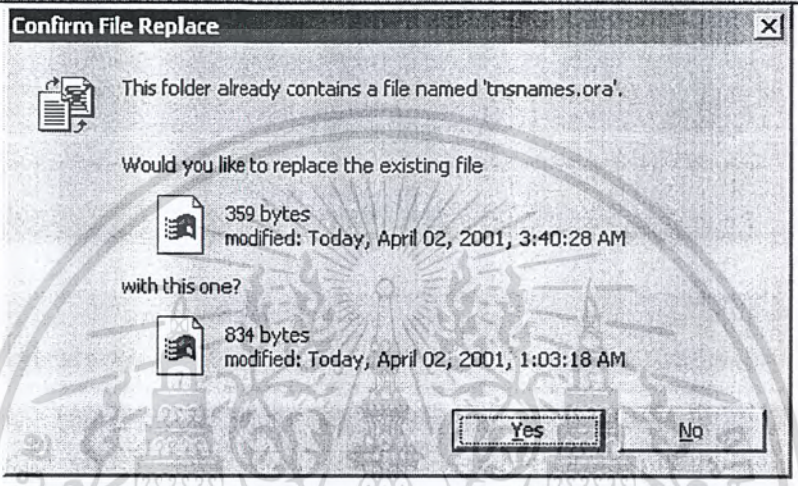
43. ใส่ค่าไดเรกทอรีที่เก็บตัวโปรแกรมแผ่นที่ 2 แล้วเลือก OK



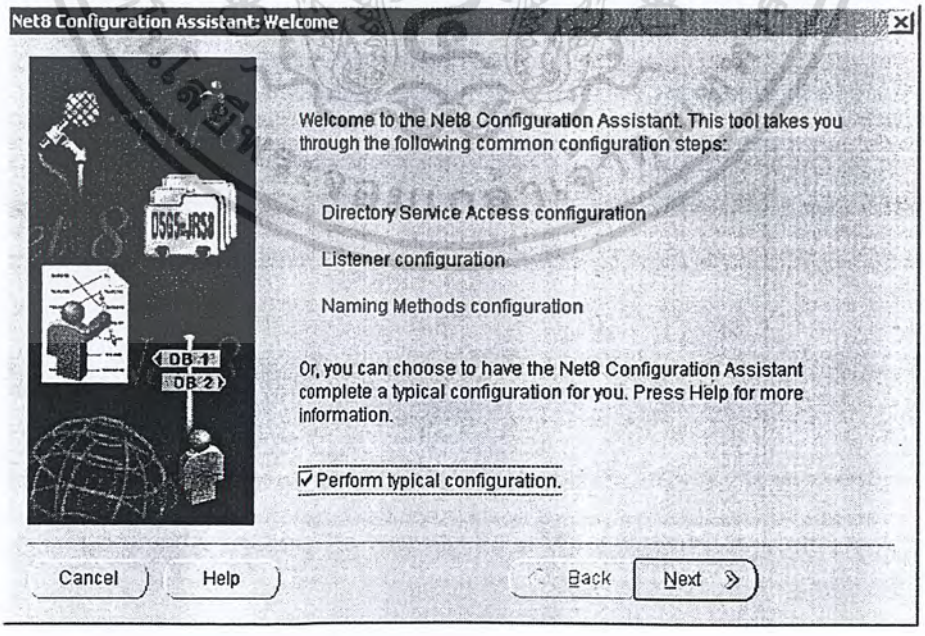
44. ใส่ค่าไดเรกทอรีที่เก็บตัวโปรแกรมแผ่นที่ 3 แล้วเลือก OK



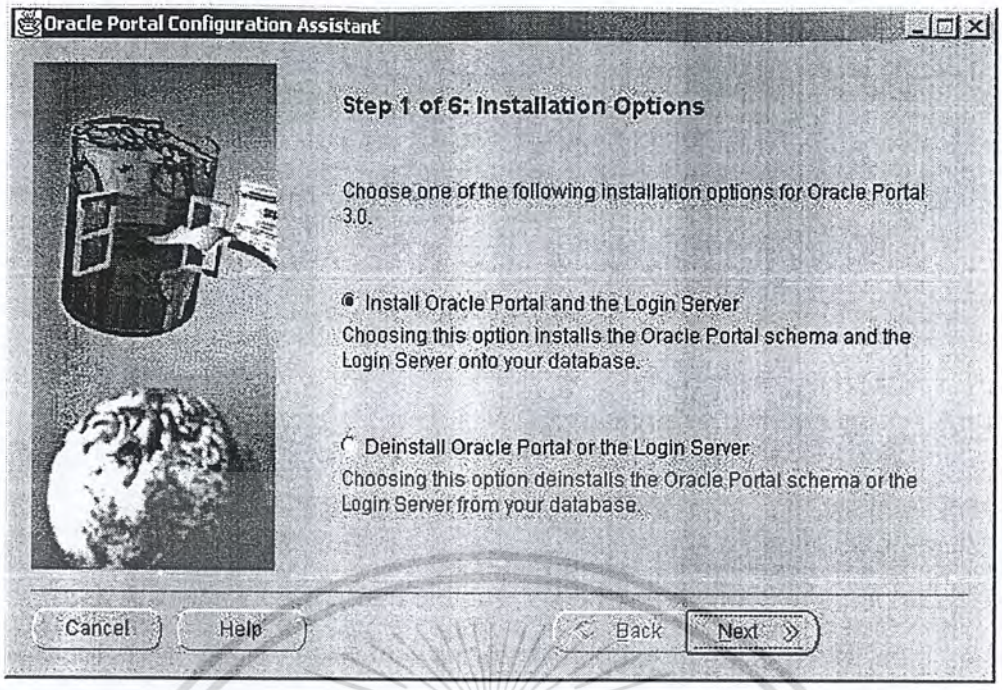




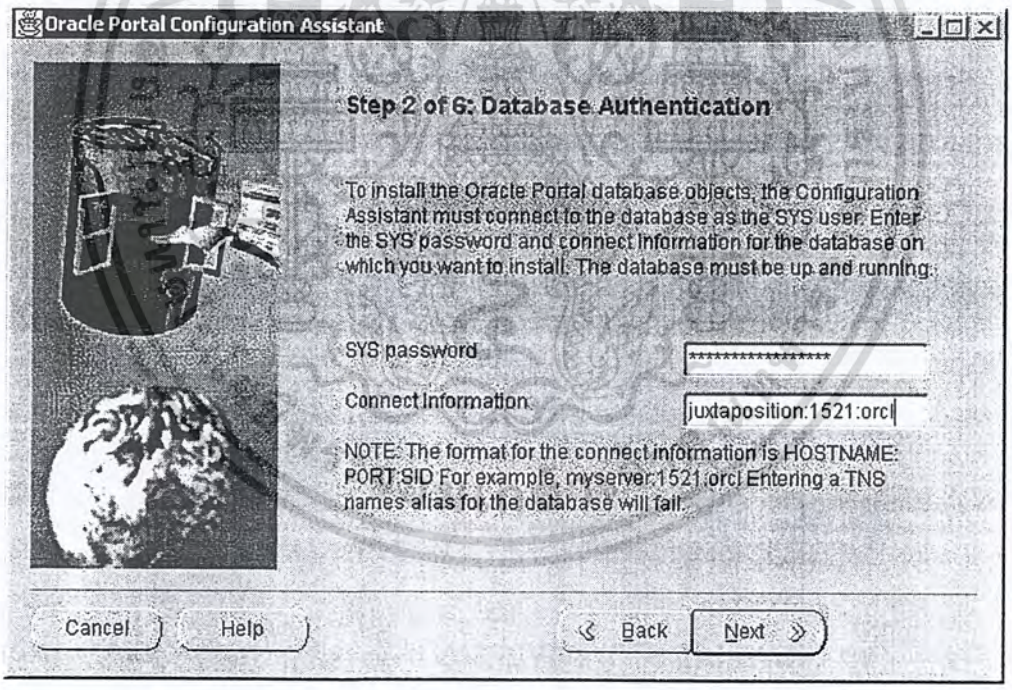
45. Copy ไฟล์ tnsnames.ora จากไดเรกทอรี C:\Oracle\Ora81\Network\Admin ไปที่ไดเรกทอรี C:\Oracle\Ora9iAS\Network\Admin



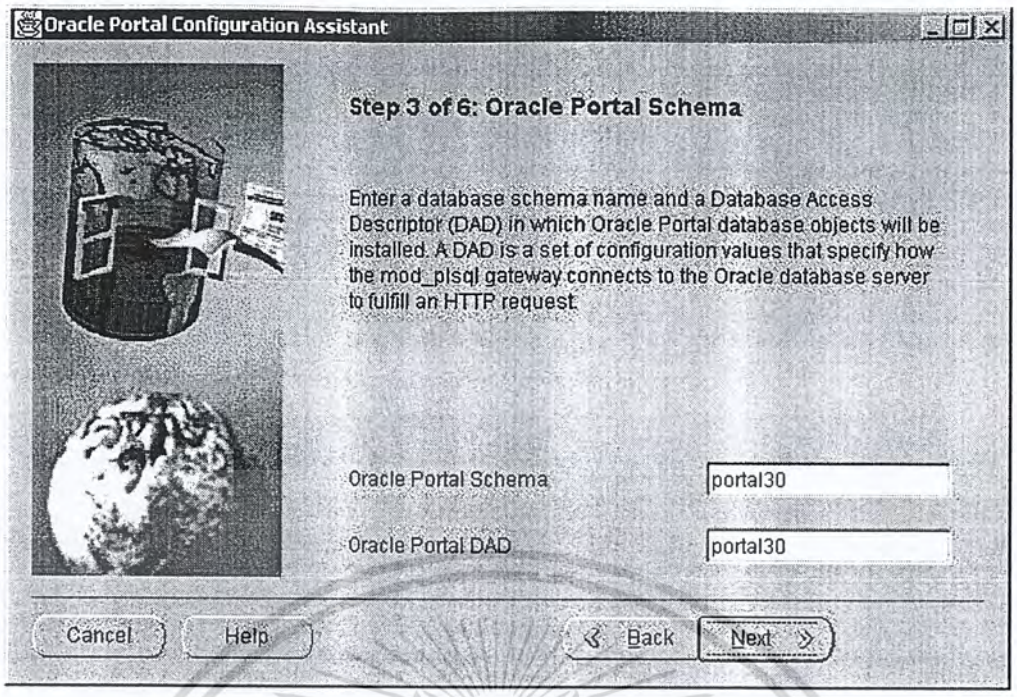
46. เลือก Perform typical configuration แล้วเลือก Next



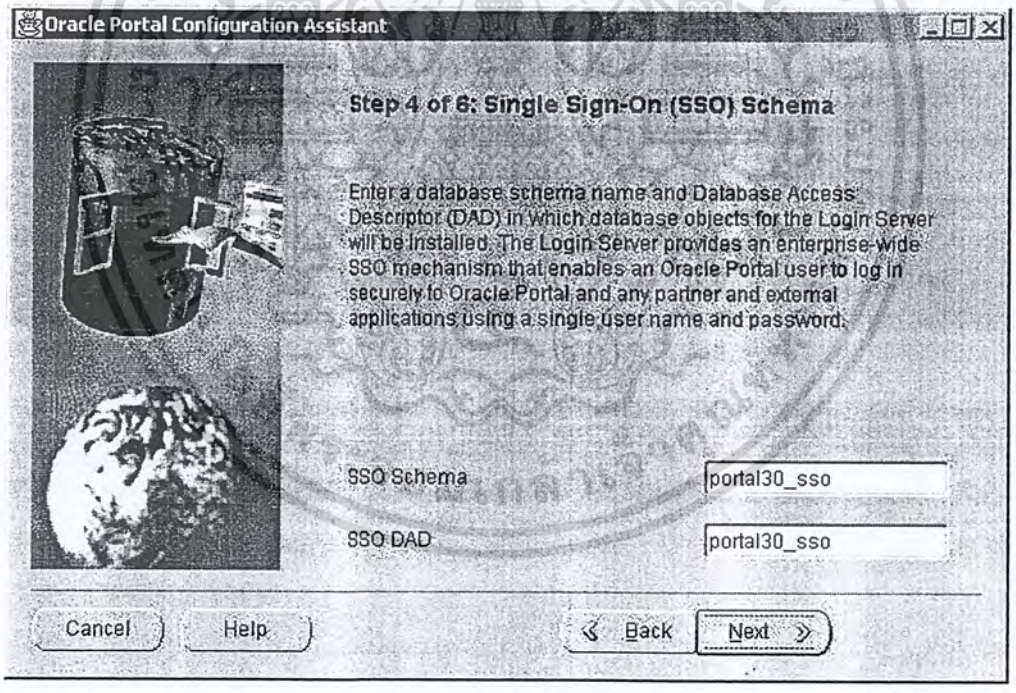
47. เลือก Install Oracle Portal and the Login Server แล้วเลือก Next



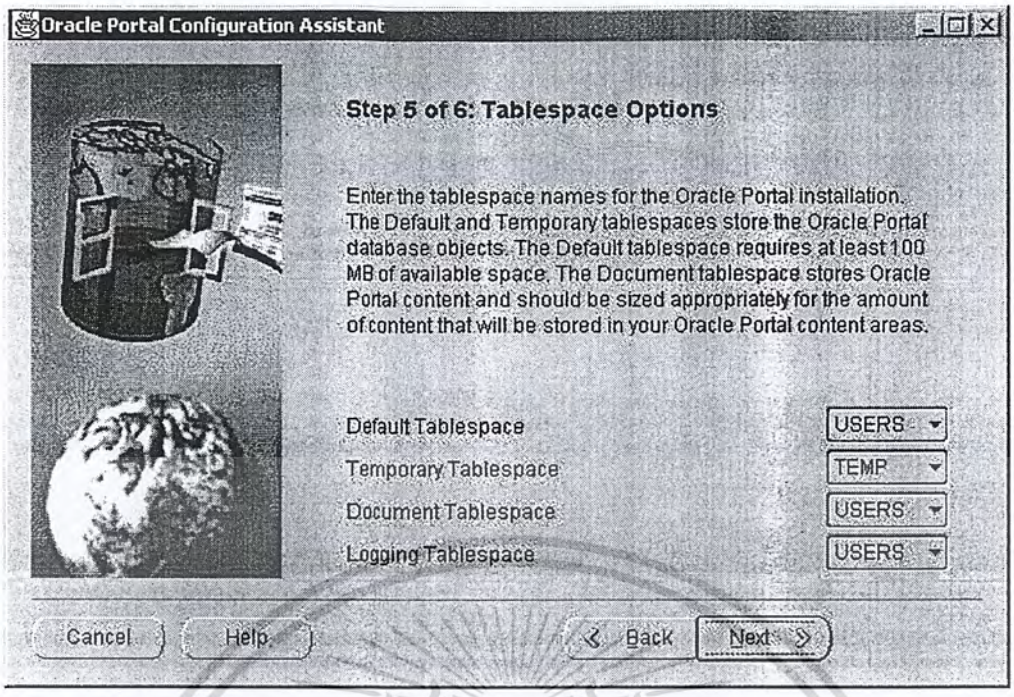
48. ใส่ค่า SYS Password ว่า "change_on_install" และใส่ค่า Connect Information ว่า "network20:1521:orcl" แล้วเลือก Next



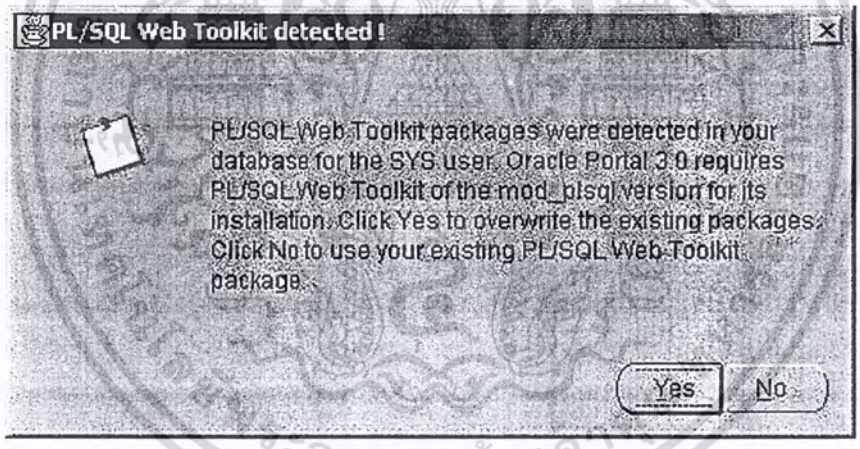
49. ใส่ค่า Oracle Portal Schema และ Oracle Portal DAD ว่า "portal30" แล้วเลือก Next



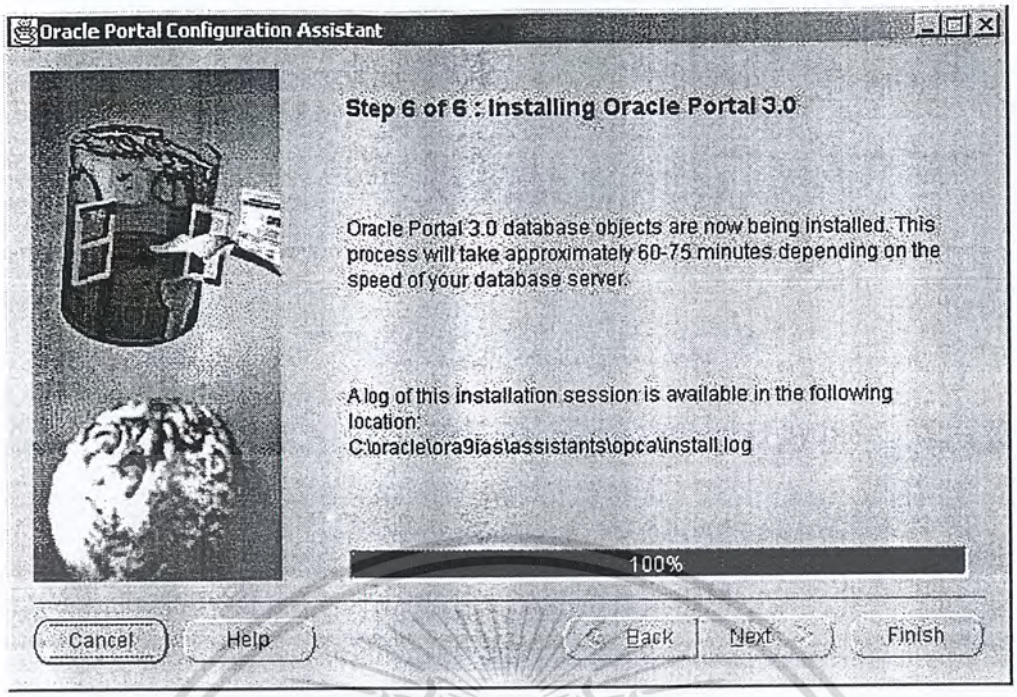
50. ใส่ค่า SSO Schema และ SSO DAD ว่า "portal30_sso" แล้วเลือก Next



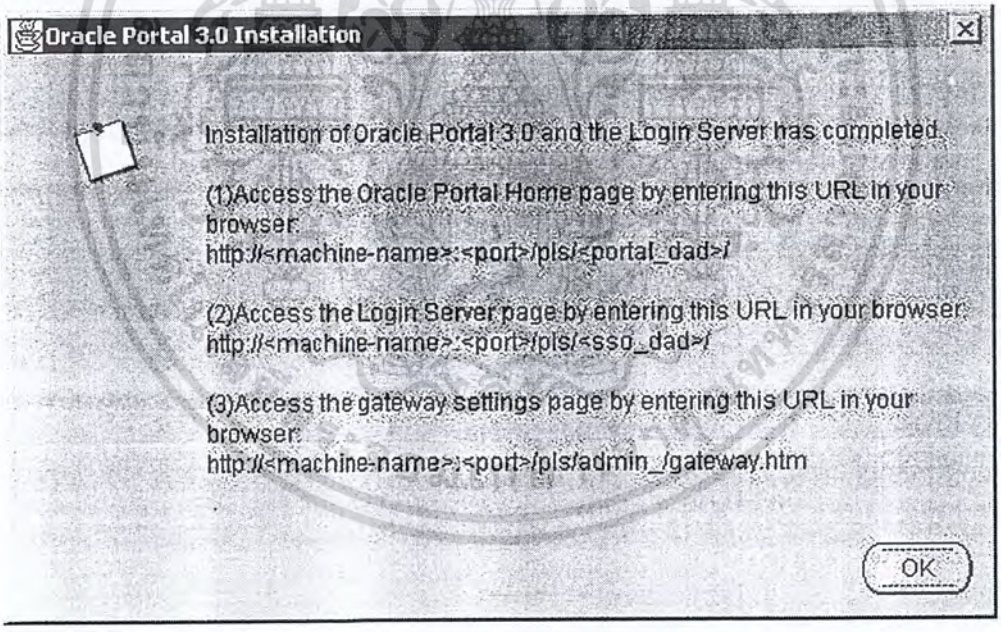
51. เลือก Next



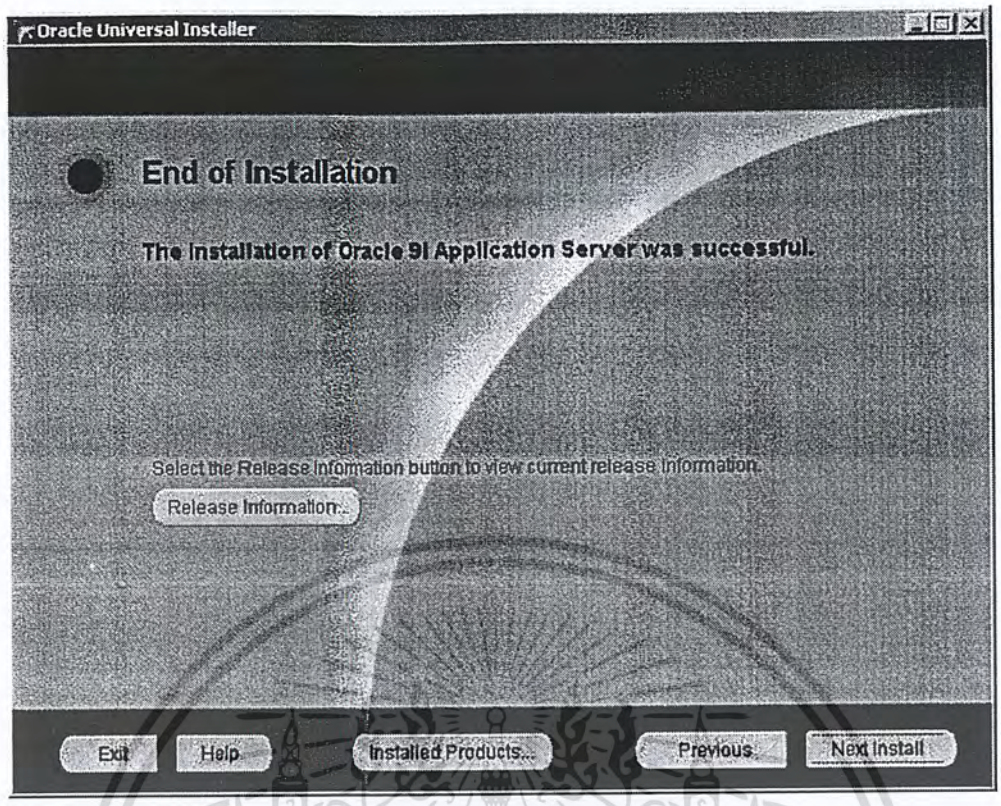
52. เลือก Yes



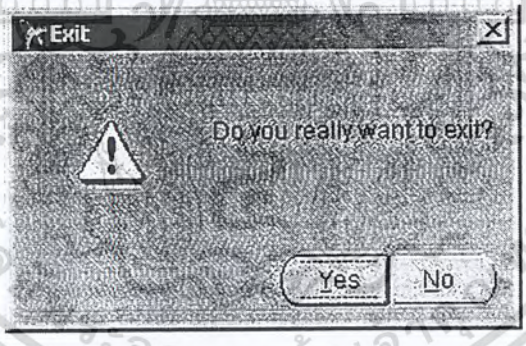
53. เลือก Finish



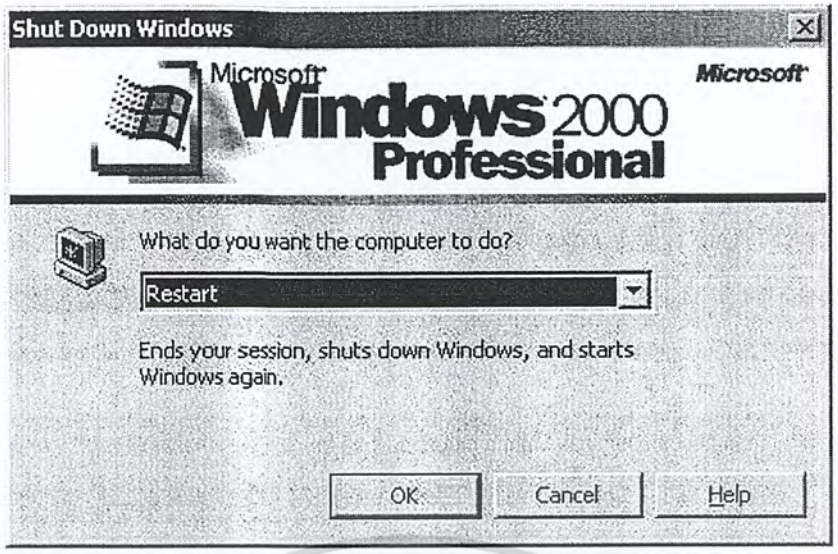
54. เลือก OK



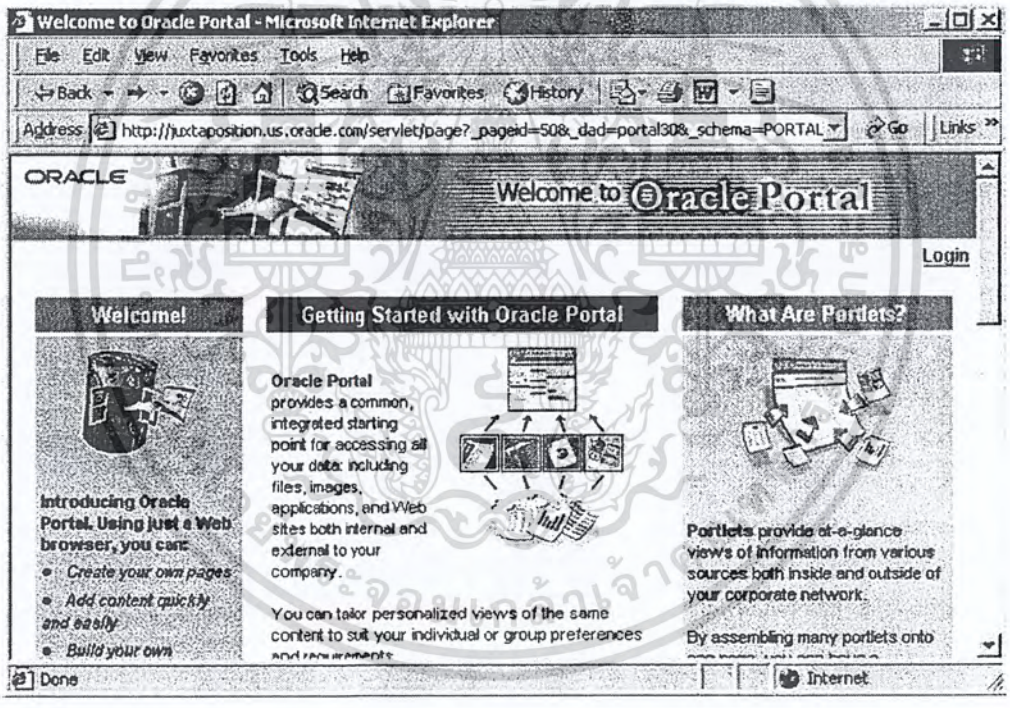
55. เลือก Exit



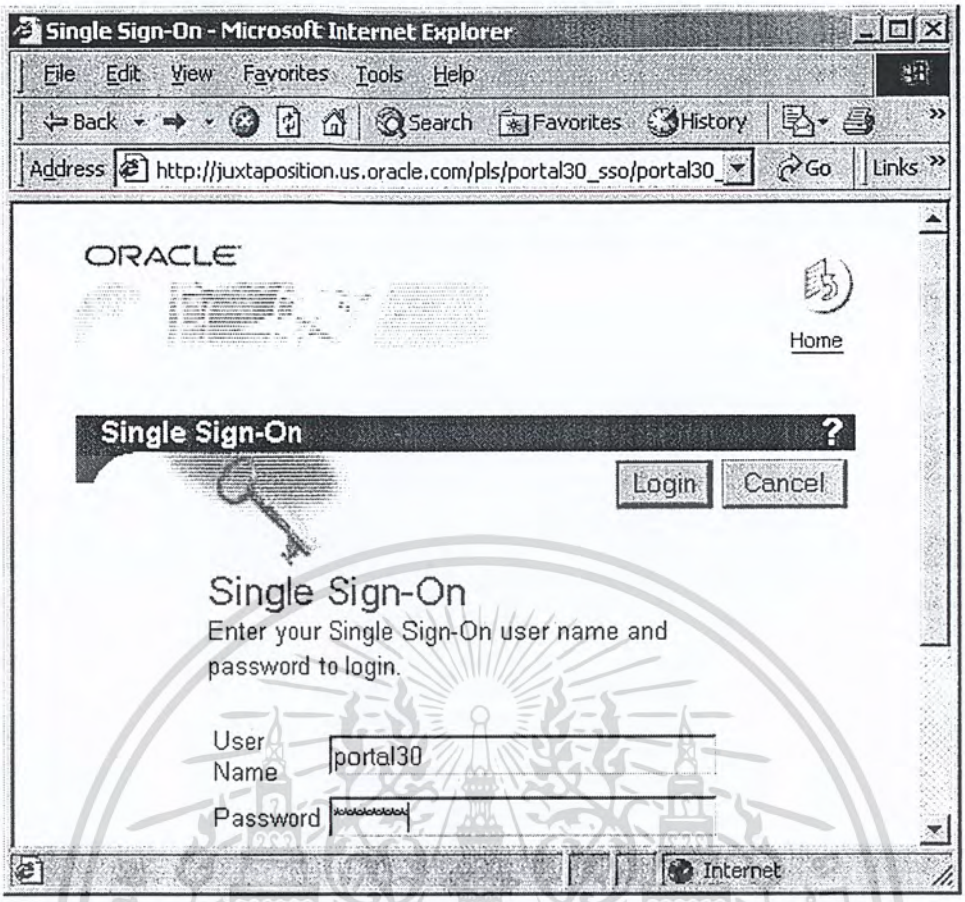
56. เลือก Yes



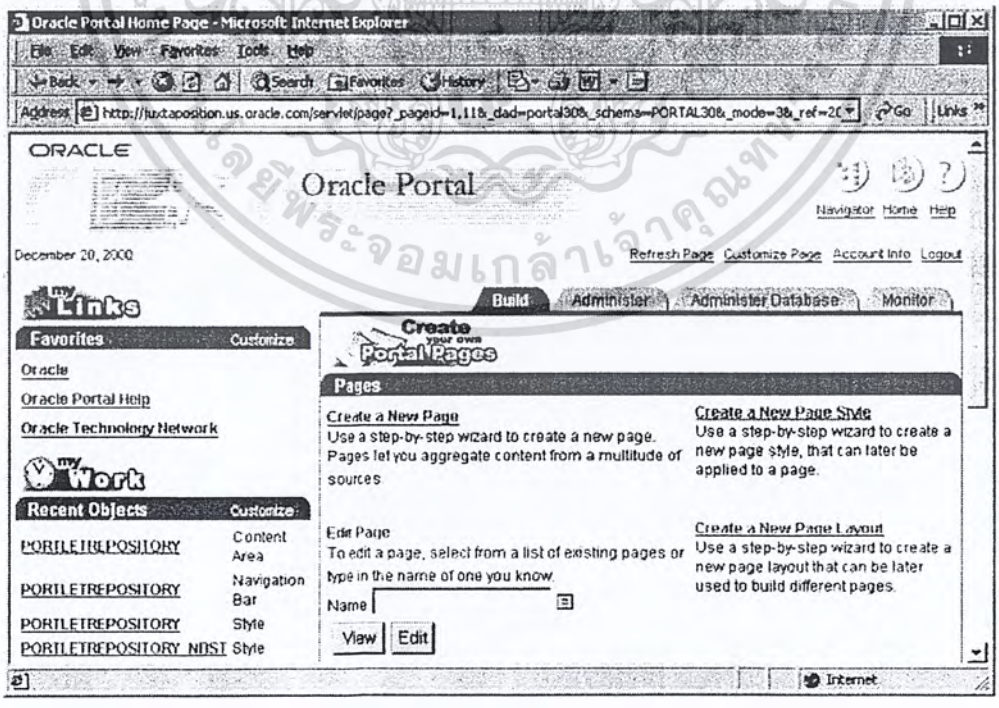
57. เลือก Start > Shut Down แล้วเลือก Restart เครื่องเพื่อให้ค่าที่ตั้งไว้ทำงาน



58. ตรวจสอบการลงโปรแกรมโดยการเปิด browser แล้วพิมพ์ URL ว่า "http://network20.ce.kmitl.ac.th/pls/portal30" แล้วเลือก GO จะแสดงหน้าจอ Welcome to Oracle Portal แล้วเลือก Login



59. ใส่ Username และ Password ว่า "portal30" แล้วเลือก Login

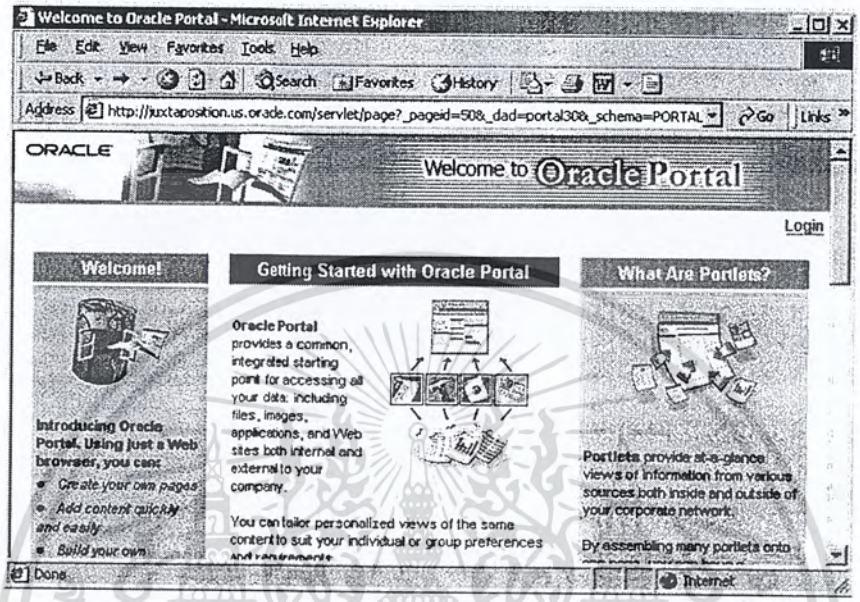


60. จะแสดงหน้าจอ Oracle Portal เป็นอันเสร็จสิ้นกระบวนการ Install

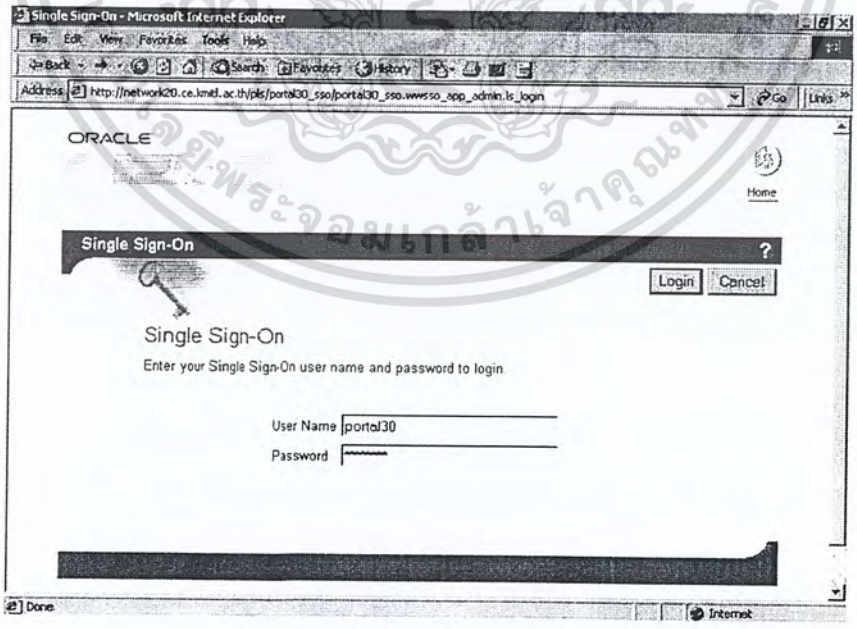
ภาคผนวก ข.

การใช้งาน Oracle9iAS Portal

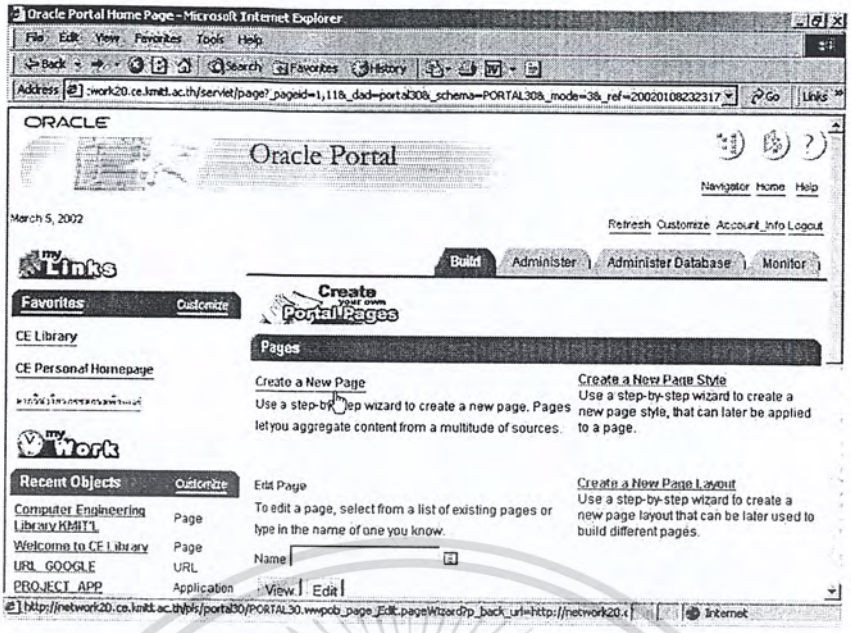
1. การสร้าง Page



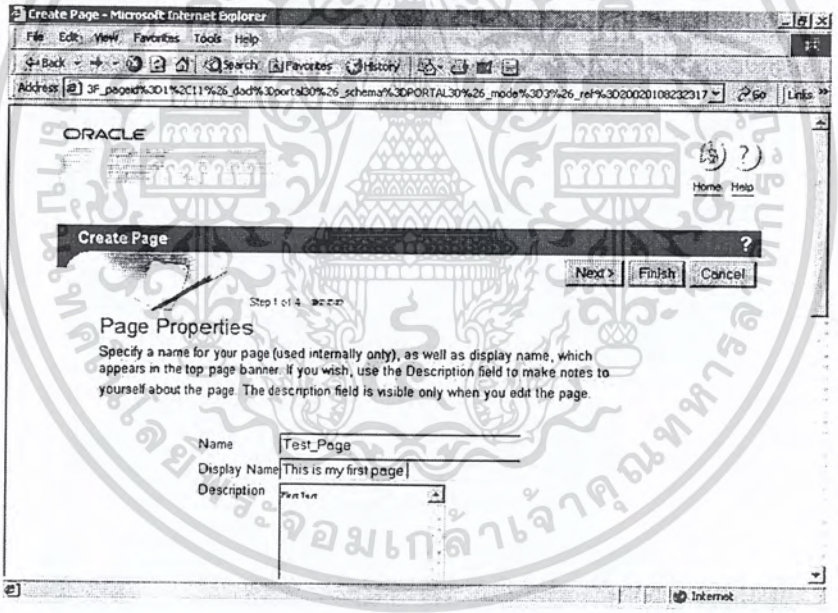
1. เมื่อเราเปิดเว็บบราวเซอร์ พิมพ์ URL “http://<ComputerName>.ce.kmitl.ac.th/pls/portal30” แล้วเลือก GO จะแสดงหน้าจอ Welcome to Oracle Portal แล้วเลือก Login ที่ปรากฏตรงมุมขวาด้านบน



2. ทำการล๊อกอินในฐานะ Portal Administrator ในที่นี้คือ User name : portal30

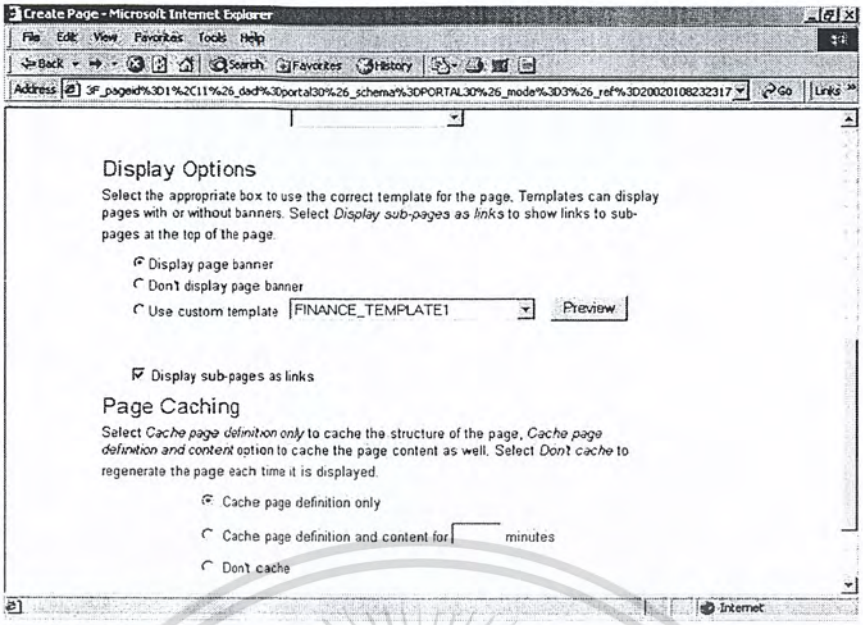


3. เมื่อเข้ามาหน้า Default Home Page เลือก Create a New Page ใน Build Tab เพื่อสร้างเว็บเพจใหม่

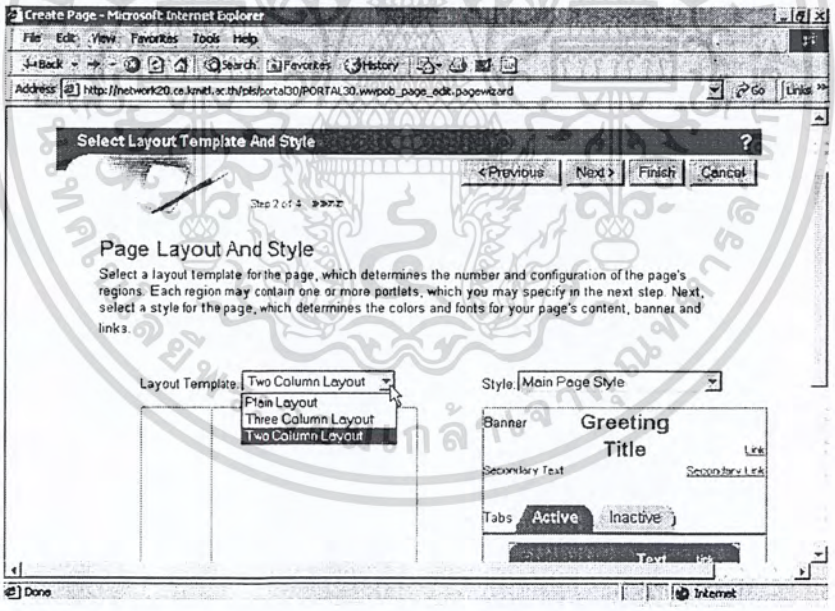


4. วิซาร์ดสำหรับสร้างเว็บเพจจะปรากฏขึ้นมา โดยมี 4 ขั้นตอน ในขั้นตอนแรกนี้

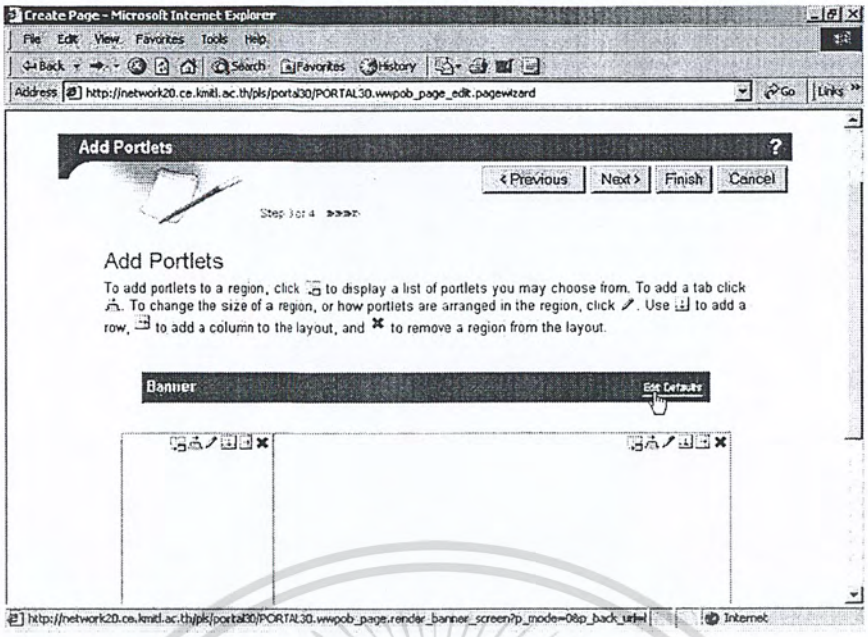
- ป้อนชื่อเว็บเพจในช่อง Name โดยต้องไม่มีช่องว่างและไม่ซ้ำ เช่น Test_Page
- ช่อง Display Name ป้อนชื่อที่ต้องการให้ปรากฏบนส่วนหัวของเว็บเพจ เทียบได้กับ <TITLE>This is my first page.</TITLE> ใน HTML นั่นเอง
- Description เป็นการระบุข้อมูลเว็บเพจ จะระบุหรือไม่ก็ได้



5. เลื่อนหน้าจอลง ในหัวข้อ Display Options สามารถเลือกได้ว่าจะให้ปรากฏแบนเนอร์หรือไม่ หรือเลือกให้แทนที่แบนเนอร์ด้วยเท็มเพลทก็ได้ (วิธีการสร้างเท็มเพลทจะกล่าวถึงในหัวข้อต่อไป) และเลือก Default Page Caching จากนั้น Next ไปยังขั้นตอนต่อไป



6. เลือก Layout ว่าจะแบ่งสัดส่วนของเว็บเพจอย่างไร และเลือก style ของเว็บเพจ ได้แก่ สี, ชนิดตัวอักษร, และหน้าตาของเว็บเพจ (การสร้าง Layout Template และ Page Style จะกล่าวในหัวข้อต่อไป) ในที่นี้เลือก Two Column Layout และ Main Page Style แล้วกด Next เพื่อไปขั้นตอนถัดไป



7. ในขั้นตอนนี้เป็นการตั้งค่า Banners และเพิ่ม Portlet ลงในเว็บเพจ

- เลือก Edit Defaults เพื่อเปลี่ยนแปลงแบนเนอร์ โดยสามารถเปลี่ยนแปลงรูปภาพแบนเนอร์และข้อความบนแบนเนอร์ได้

- การเพิ่ม Portlet มีไอคอนที่สำคัญดังนี้

 เพิ่ม Portlet

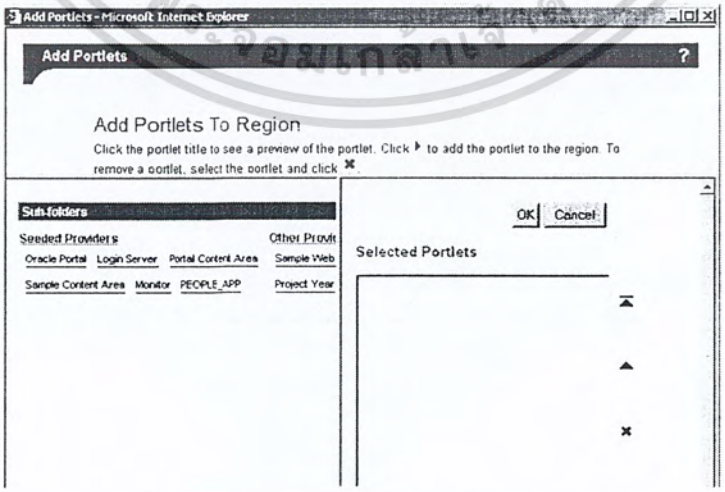
 เพิ่ม Tab

 ตั้งค่าคุณสมบัติของ Region

 เพิ่มแถวใน Region

 เพิ่มคอลัมน์ใน Region

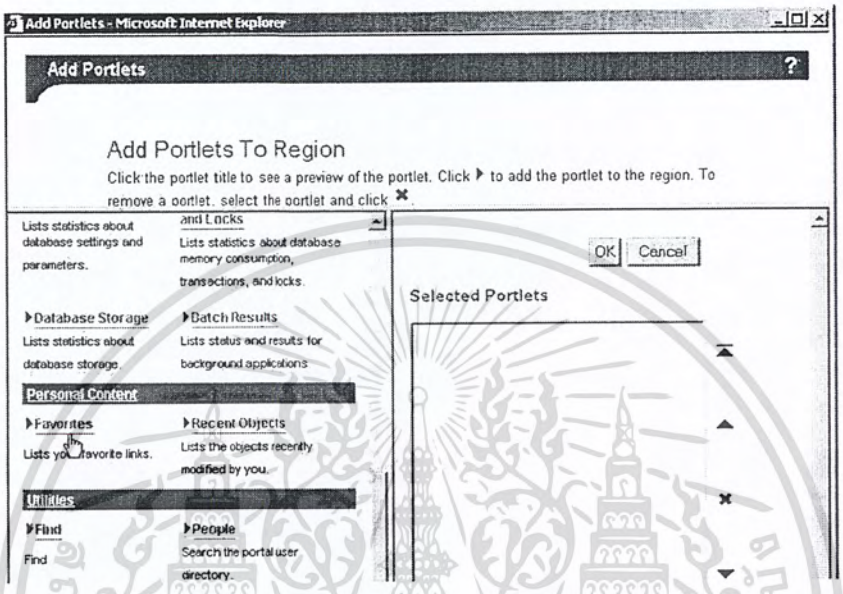
 ลบ Region



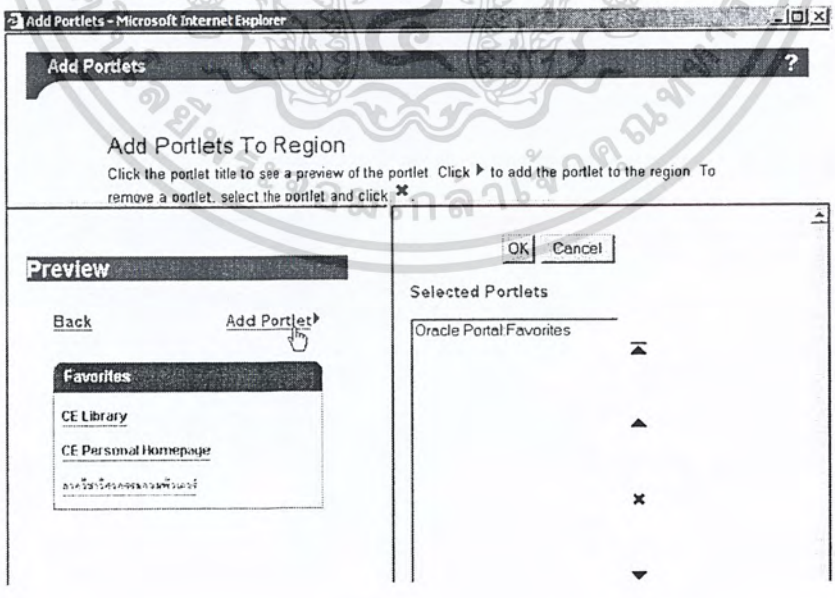
8. เลือก เพื่อเพิ่ม Portlet จะปรากฏหน้าต่างของ Portlet ให้เลือก

- เฟรมด้านซ้าย คือ Portlet Repository ที่รวบรวม Portlet ทั้งหมด โดยผู้ใช้สามารถเลือกได้ว่าจะนำ Portlet ไหนไปปรากฏบนเว็บเพจ และสามารถ Customize ภายหลังได้ สำหรับ Portlet เหล่านี้นำเข้ามาผ่านทาง Provider ที่เปรียบเสมือนตัวติดต่อระหว่าง Portlet และ Oracle Portal

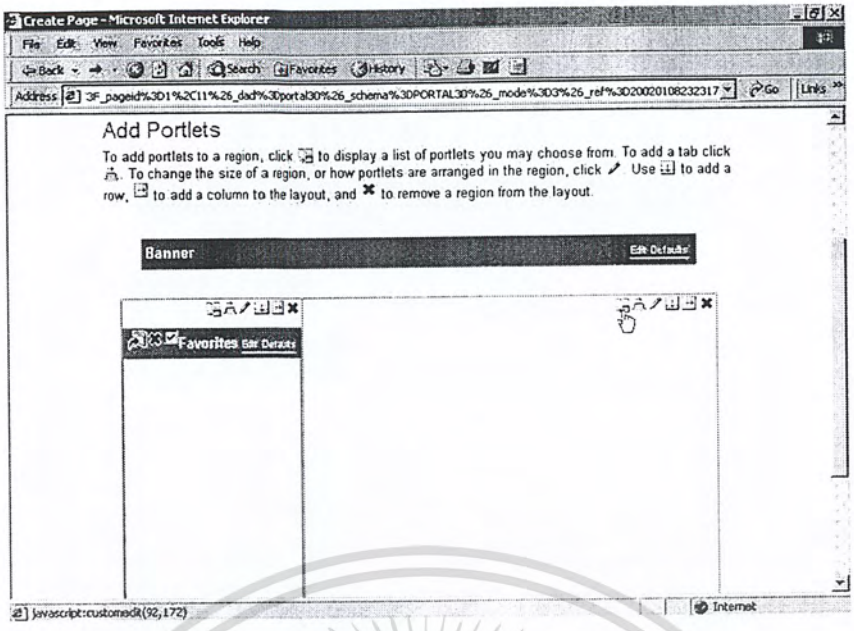
- เฟรมด้านขวาแสดง Portlet ที่เราเลือกเพื่อไปแสดงใน Region ของเว็บเพจนั่นเอง แต่เนื่องจากยังไม่ได้เลือก Portlet เฟรมนี้จึงยังไม่มีอะไรปรากฏขึ้นมา



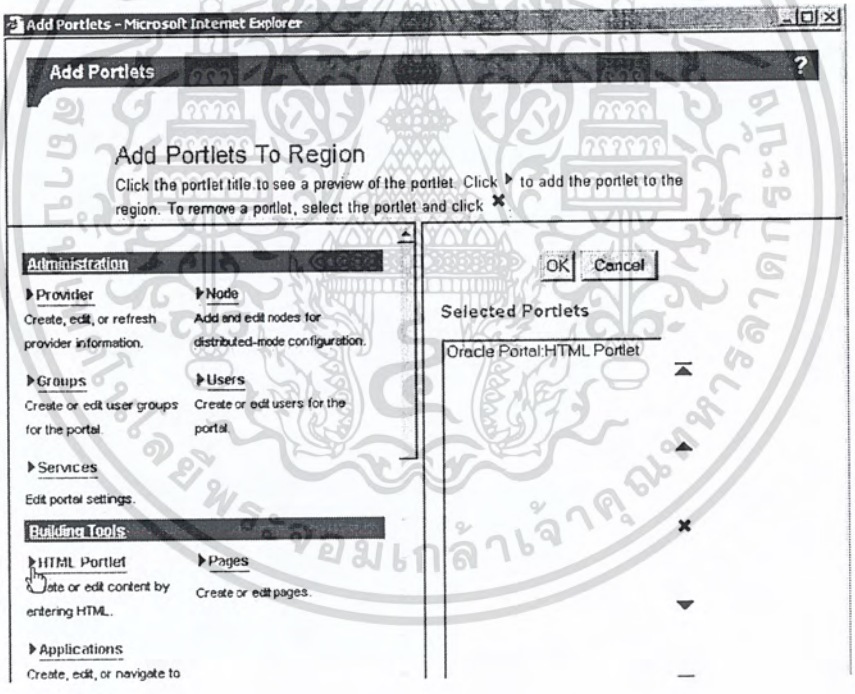
9. เลือก Oracle Portal > Favorites เพื่อเพิ่ม Portlet ลงบนเว็บเพจ (ถ้าคลิกที่ ▶ หน้า Favorites จะเป็นการเพิ่ม Portlet โดยไม่ต้อง Preview)



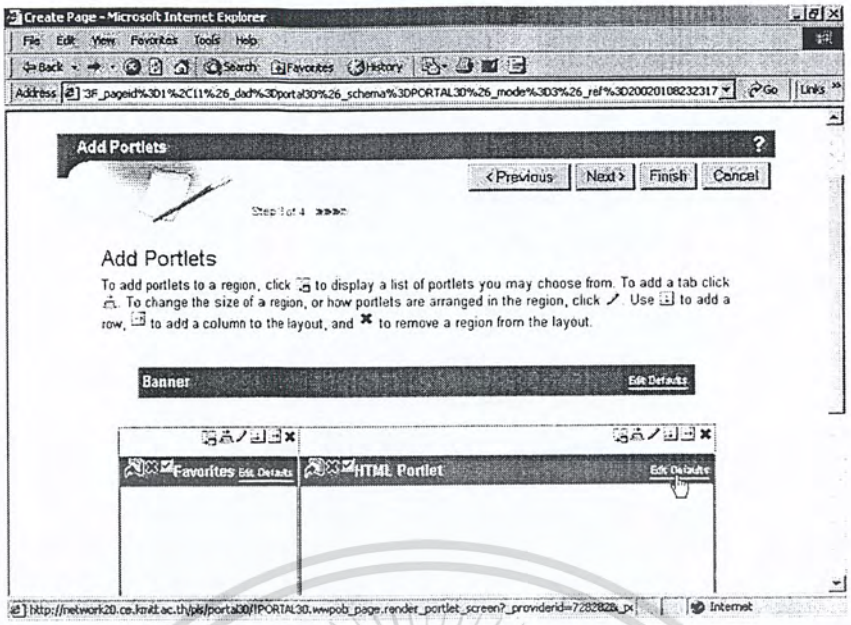
10. เมื่อเลือก Portlet ครบแล้ว คลิก OK



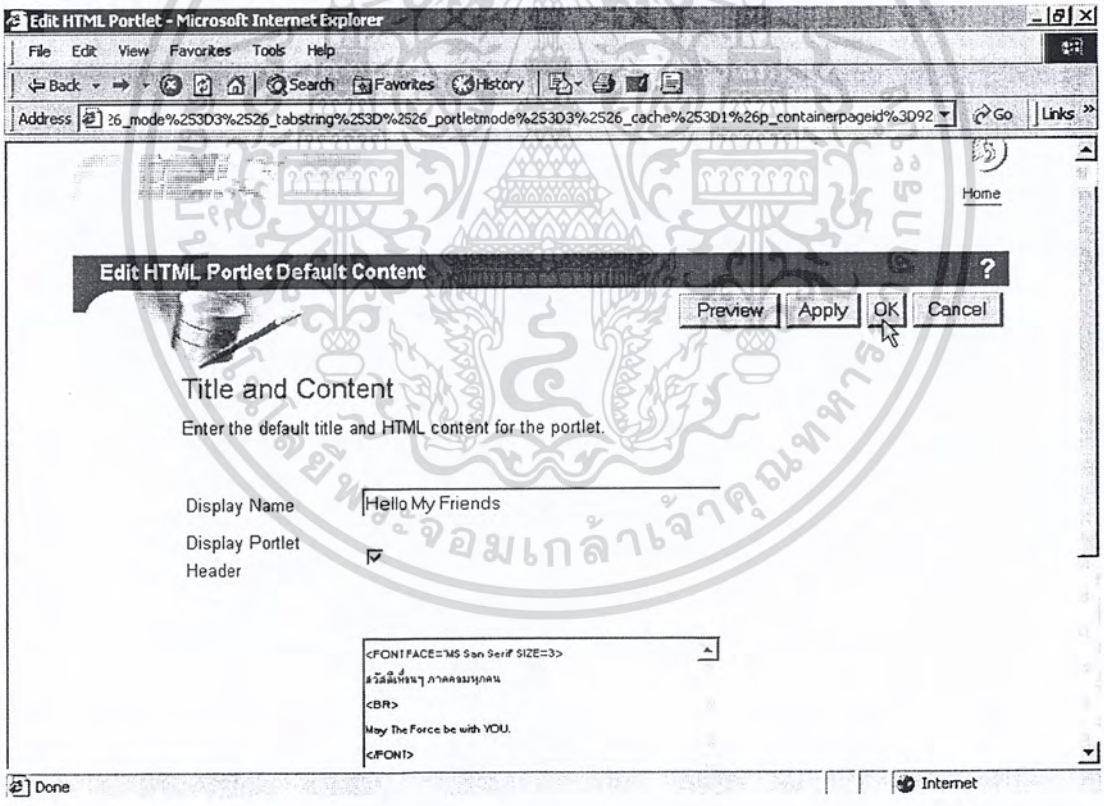
11. จะเห็นว่ามี Favorites เพิ่มขึ้นมาบนพื้นที่ด้านซ้ายแล้ว จากนั้นลองเพิ่ม Portlet อื่นในพื้นที่ด้านขวา



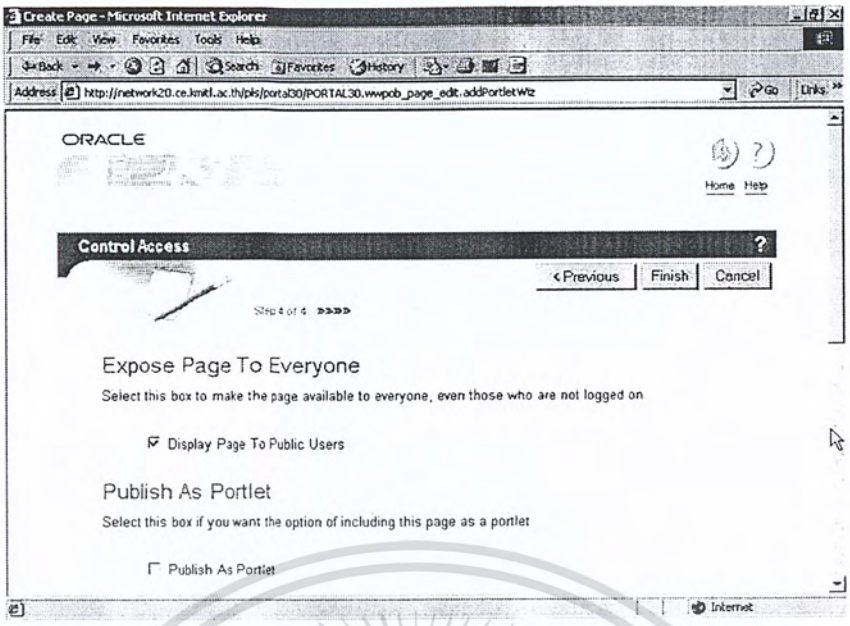
12. เพิ่ม HTML Portlet แล้วคลิก OK



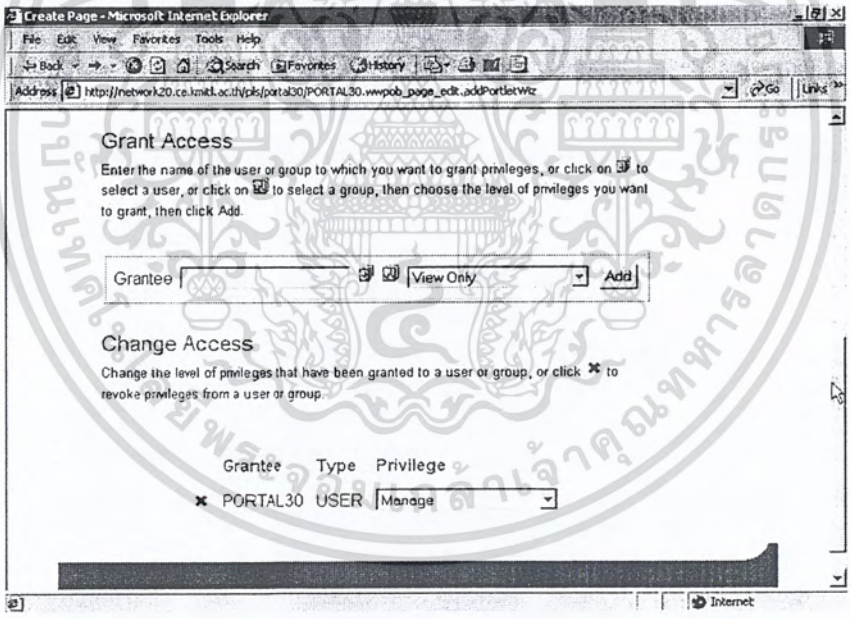
13. ทำการแก้ไข HTML Portlet โดยคลิกที่ Edit Default link



14. แก้ไข Display Name ที่จะปรากฏส่วนหัวของ Portlet และแก้ไข Content ตามต้องการด้วยโค้ด HTML แล้วคลิก OK (สามารถคลิก Preview เพื่อดูการแสดงผลที่ได้จากโค้ดโปรแกรมก่อนได้)



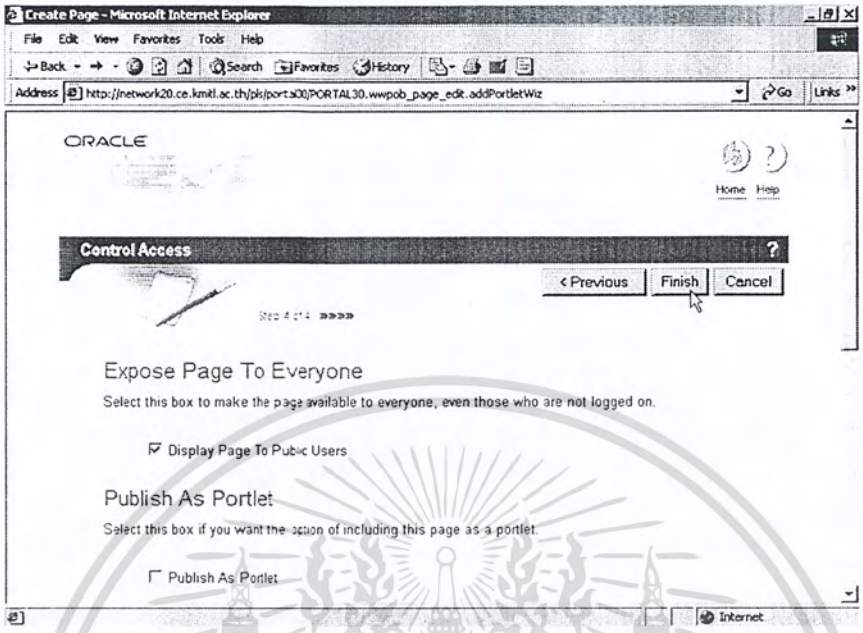
15. ขั้นตอนสุดท้ายนี้เป็นการกำหนดว่าจะแสดงผลให้ Public Users เห็นหรือไม่ และให้ Publish เหมือน Portlet หรือไม่ ในที่นี้ให้ทำตามรูป จากนั้นเลื่อนหน้าจอลง



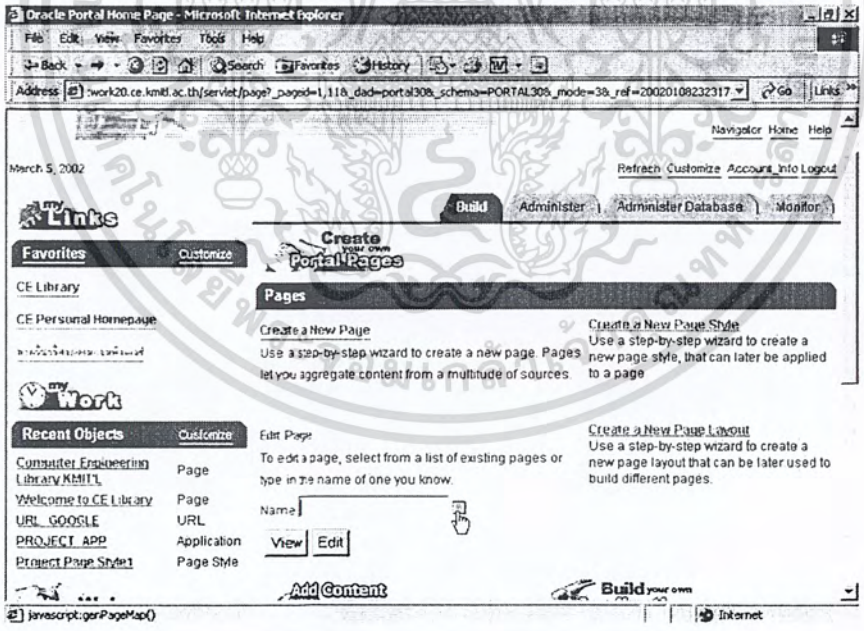
16. กำหนดความลำดับความสำคัญของบุคคลที่สามารถเข้ามาแก้ไขหรือจัดการเว็บเพจนี้ได้ มี Privilege ให้เลือกดังนี้

- Manage
- Edit Contents
- Manage Style
- Customization (Full)

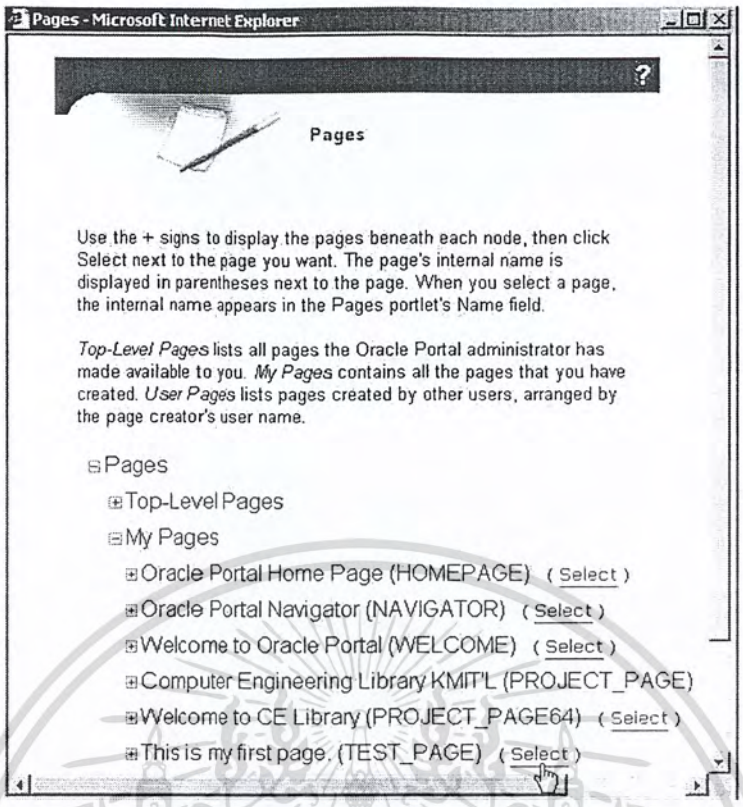
- Customization (Add-only)
- View Only



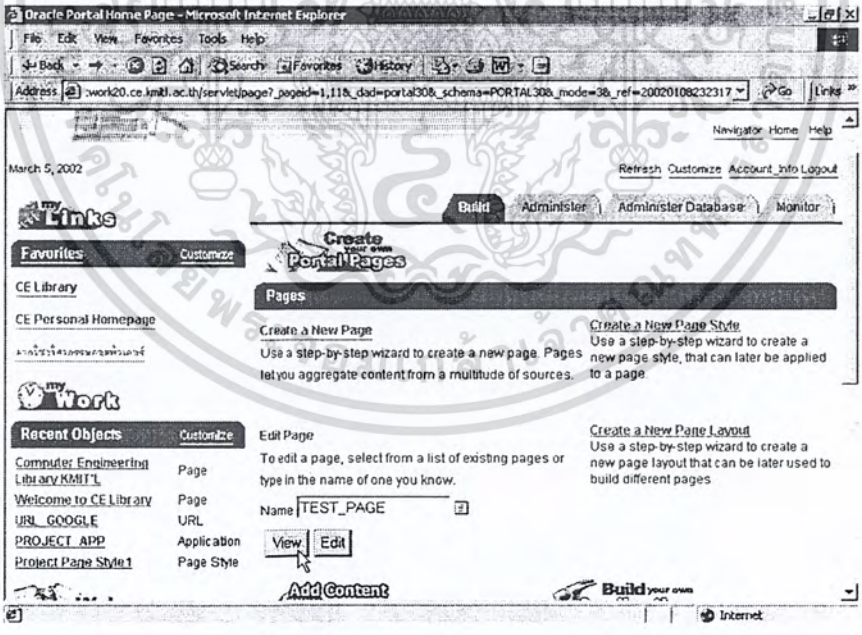
17. คลิก Finish เป็นการจบการสร้างเว็บเพจนี้ จากนั้นเว็บเพจหน้าหลักจะแสดงผลแทน



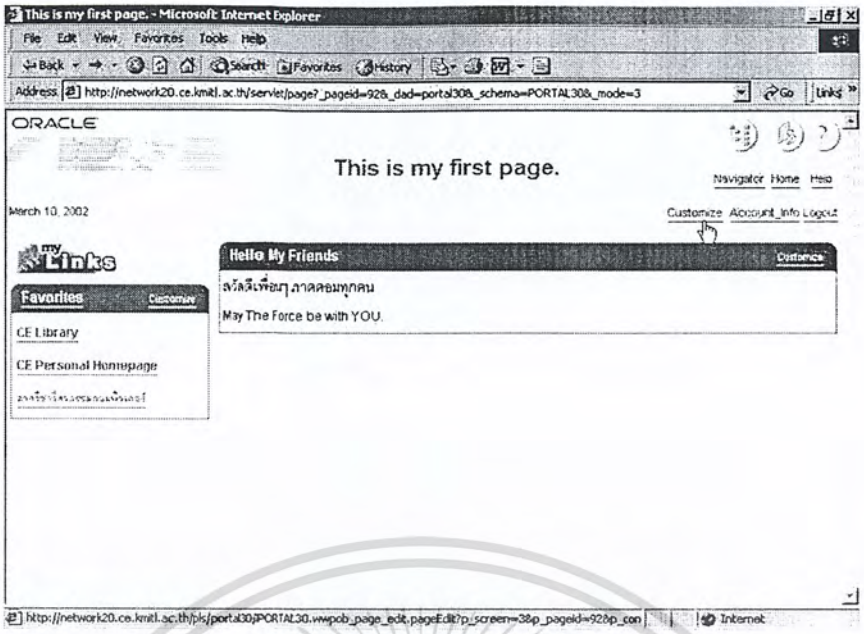
18. เลือก 📄 เพื่อเลือกเว็บเพจ



19. คลิก Select ที่ This is my first page (TEST_PAGE) ที่เพิ่งสร้างเสร็จ



20. ชื่อ TEST_PAGE ปรากฏในช่อง Name ทำการ View ขึ้นมาดูเว็บเพจที่ได้สร้างขึ้น

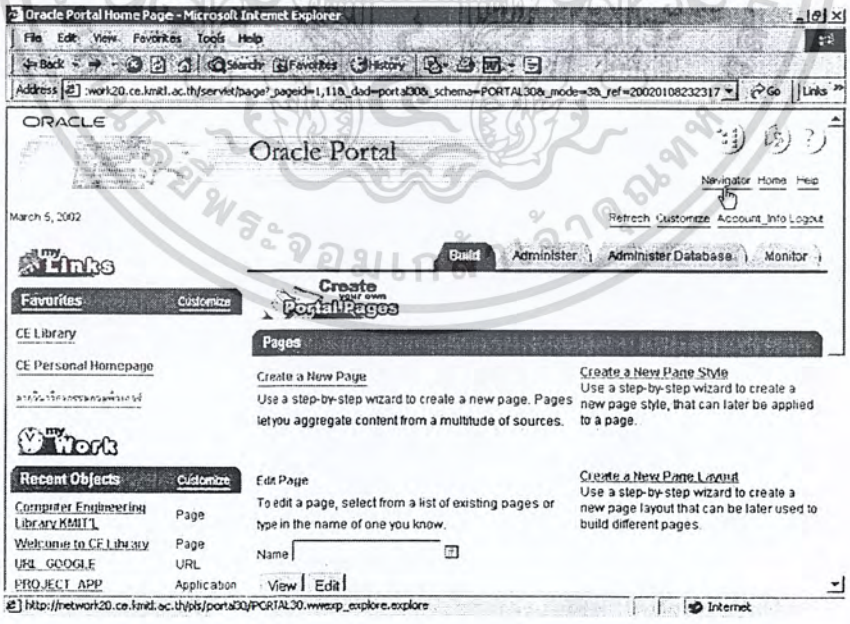


21. เว็บเพจที่สร้างขึ้นเป็นดังรูป เราสามารถแก้ไขเปลี่ยนแปลงเว็บเพจใหม่โดยเลือก Customize หรือเลือก Home เพื่อกลับไปยังหน้าหลัก

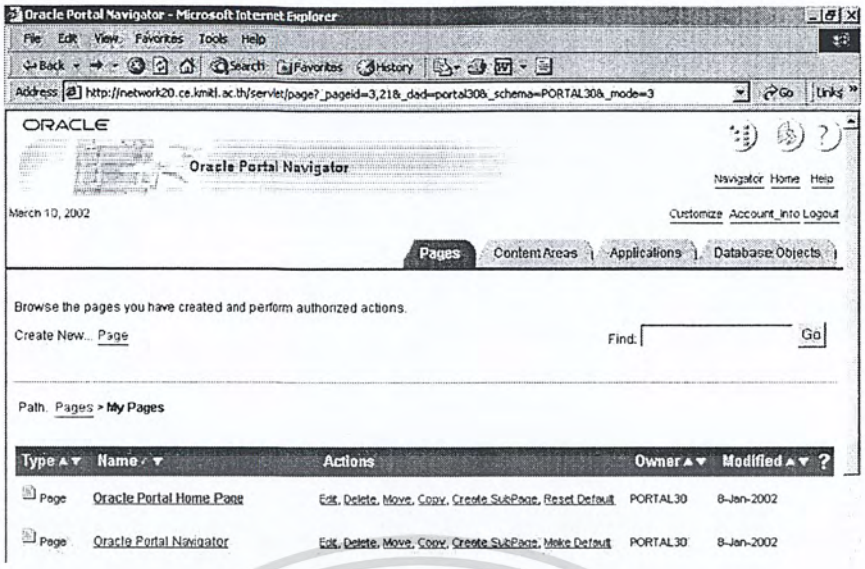
หมายเหตุ อีกวิธีในการเข้าถึงเว็บเพจที่สร้างจากเว็บบราวเซอร์โดยตรง สามารถพิมพ์ URL ดังนี้ http://<Computer_Name>/pls/portal30/url/page/<Name_Page>

จากตัวอย่างนี้ คือ http://network20.ce.kmitl.ac.th/pls/portal30/url/page/test_page

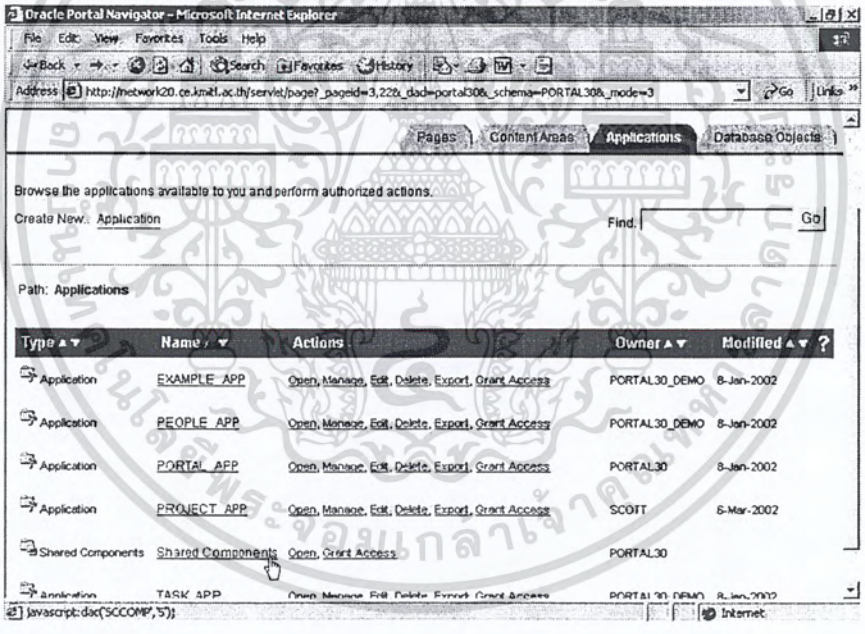
2. การสร้างเท็มเพลต และการใช้ Navigator



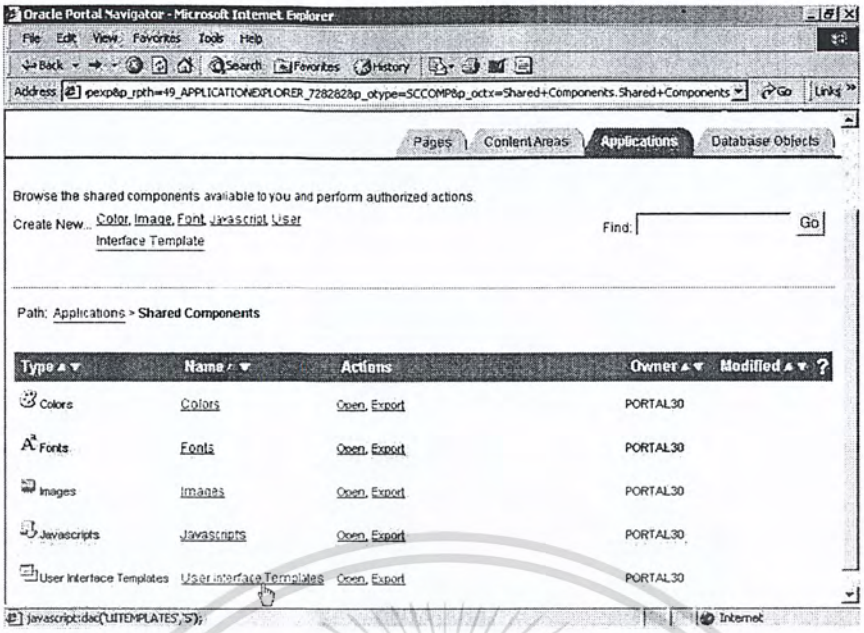
1. จากหน้าเว็บเพจหลักเลือก Navigator เปรียบเสมือนแสดงโครงสร้างเป็น Tree แทนการแสดงด้วยรูปแบบเว็บเพจหลัก



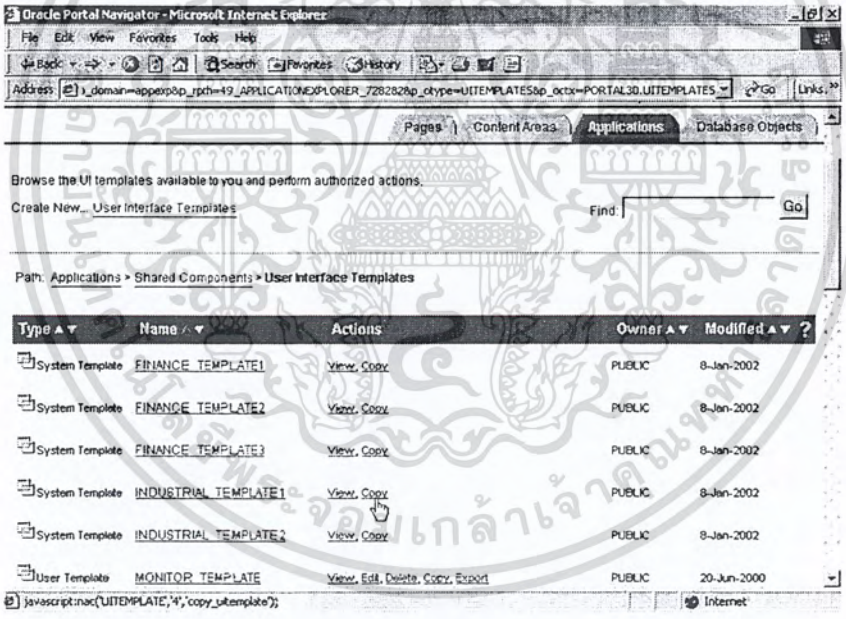
2. ในหน้า Navigator มีแท็บ 4 แท็บดังรูป เราสามารถจัดการกับเว็บเพจ, Content Areas, แอพพลิเคชั่น และ Database Objects ได้โดยง่ายผ่านทาง Actions ต่างๆ

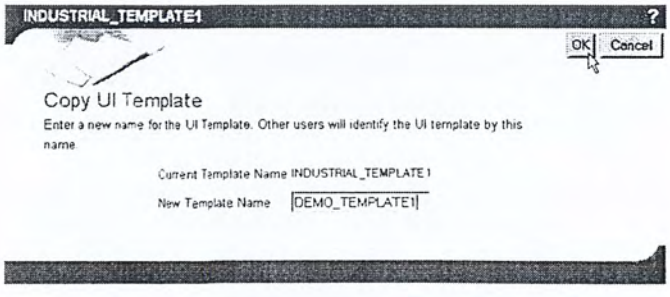


3. การจัดการเท็มเพลต เลือก Shared Component ใน Application Tab

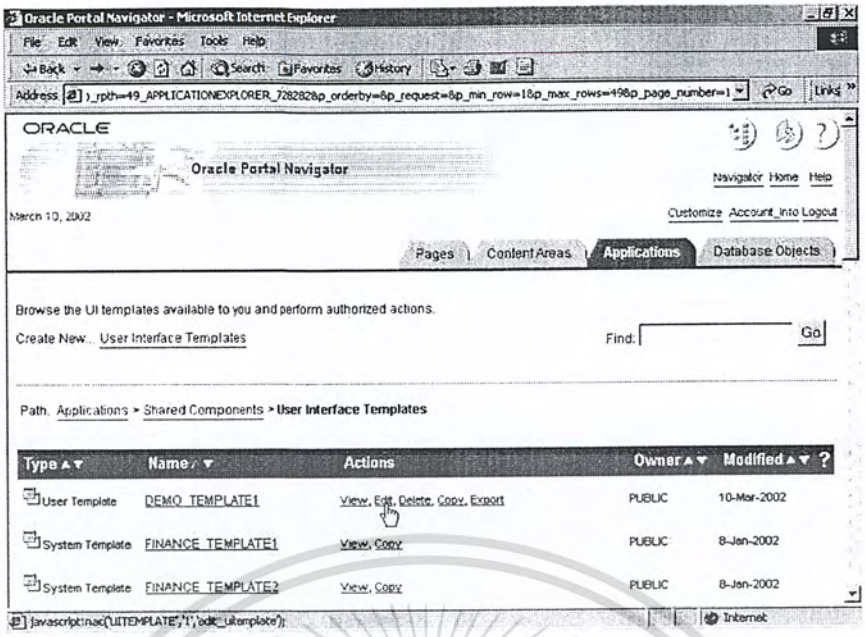


4. เลือก User Interface Templates

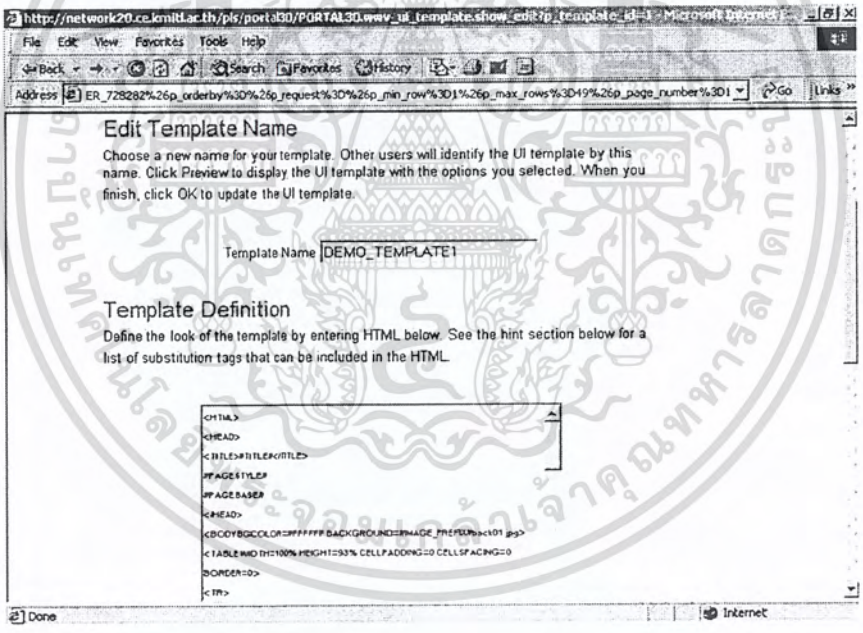




5. คัดลอกเท็มเพลตที่มีอยู่ และป้อนชื่อเท็มเพลตใหม่ที่ต้องการ คลิก OK

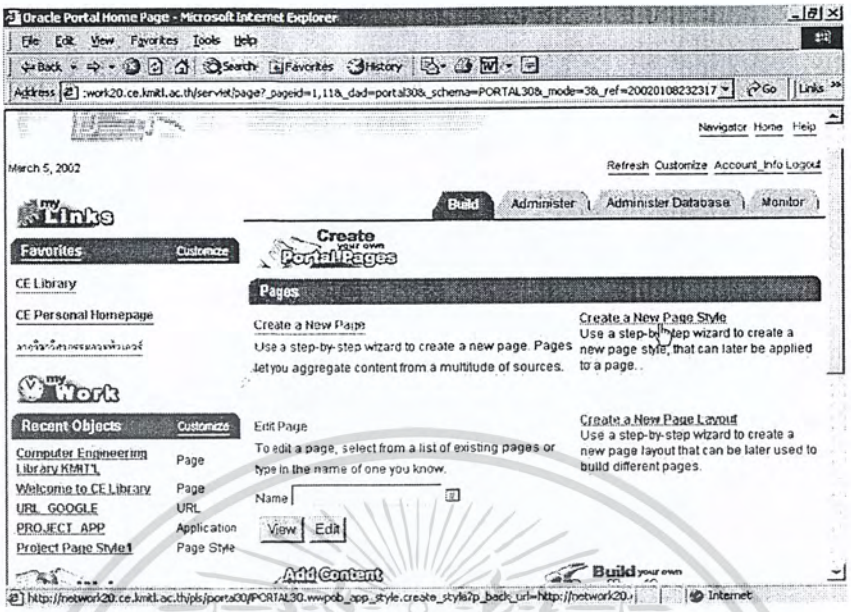


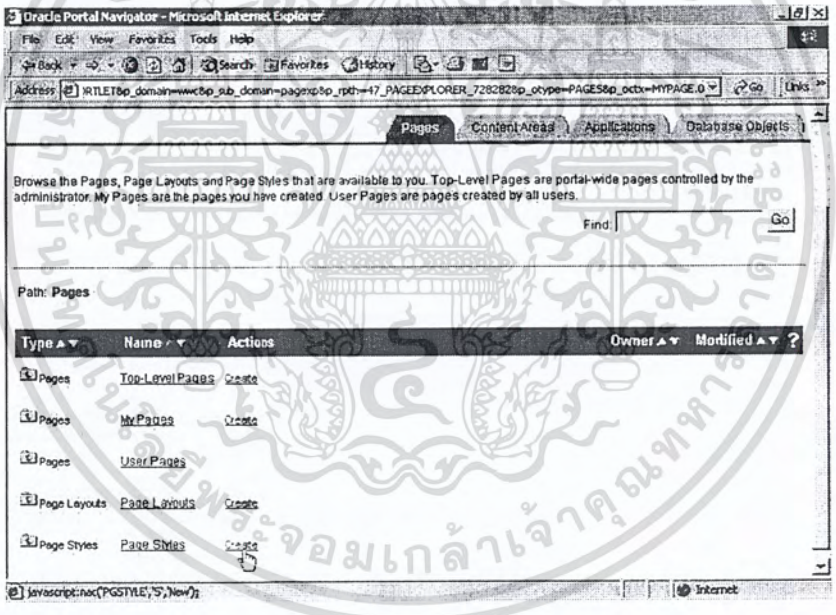
6. แก้ไข DEMO_TEMPLATE1 ที่สร้างขึ้นใหม่ คลิก Edit



7. เท็มเพลตสร้างจากโครงสร้างภาษา HTML, <ORACLE> Tags และภาษาอื่นเช่น JavaScript

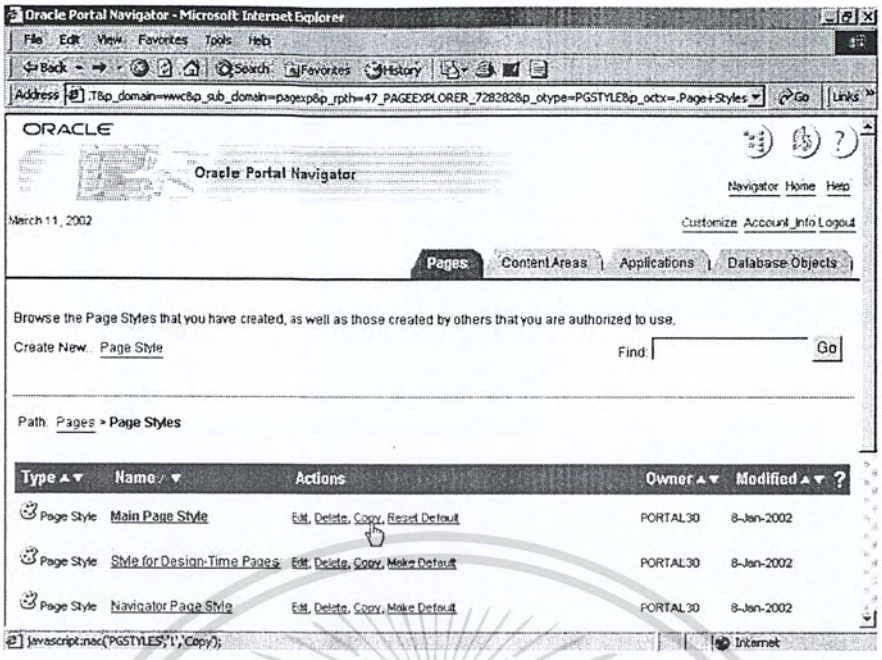
3. การสร้างและเปลี่ยนแปลง Page Style



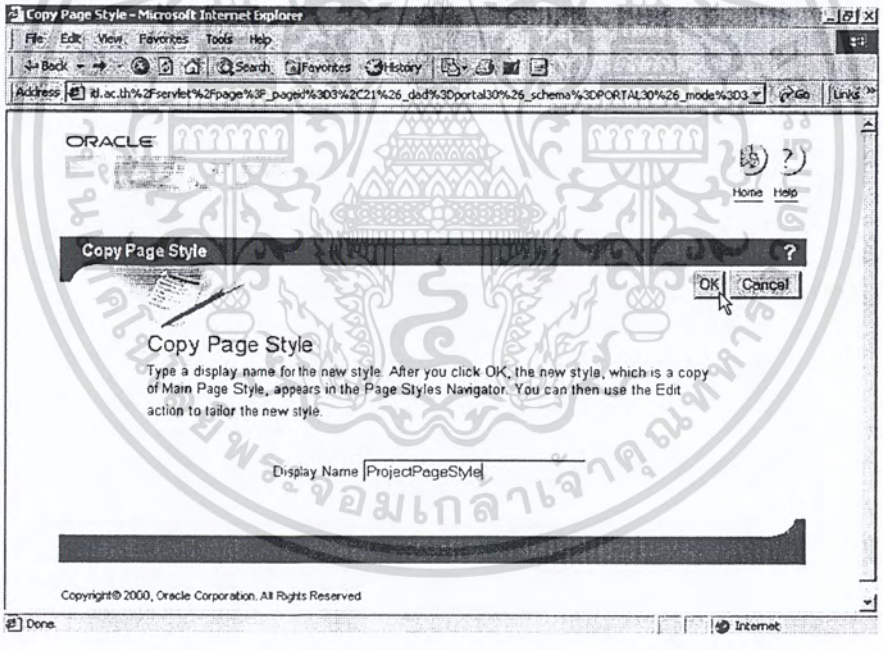


1. การสร้าง Page Style สามารถเรียกได้ 2 วิธี

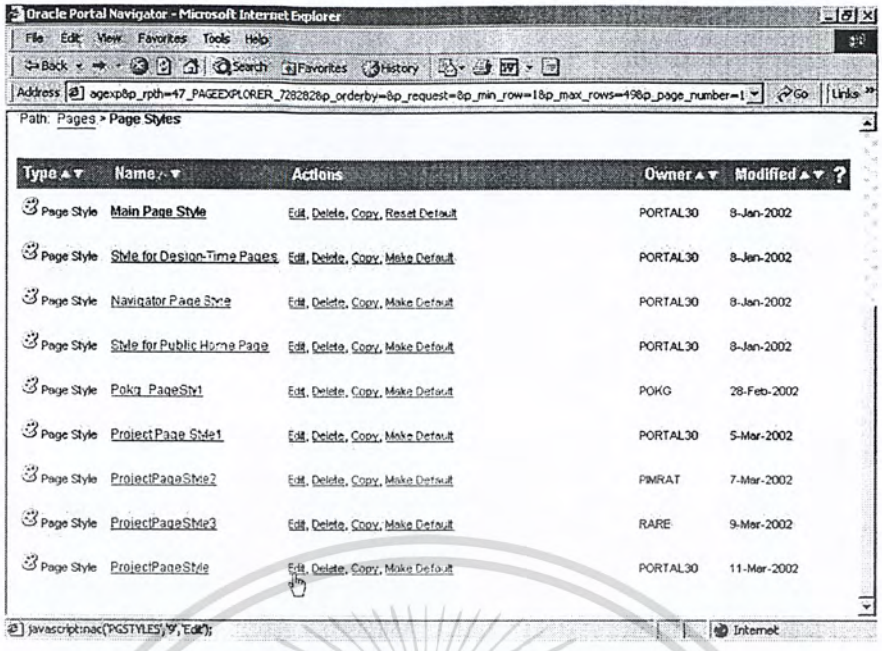
- วิธีแรก เลือก Create a New Page Style จากหน้าหลัก
- วิธีที่ 2 เลือก Navigator > Pages Tab และเลือก Creates Page Styles



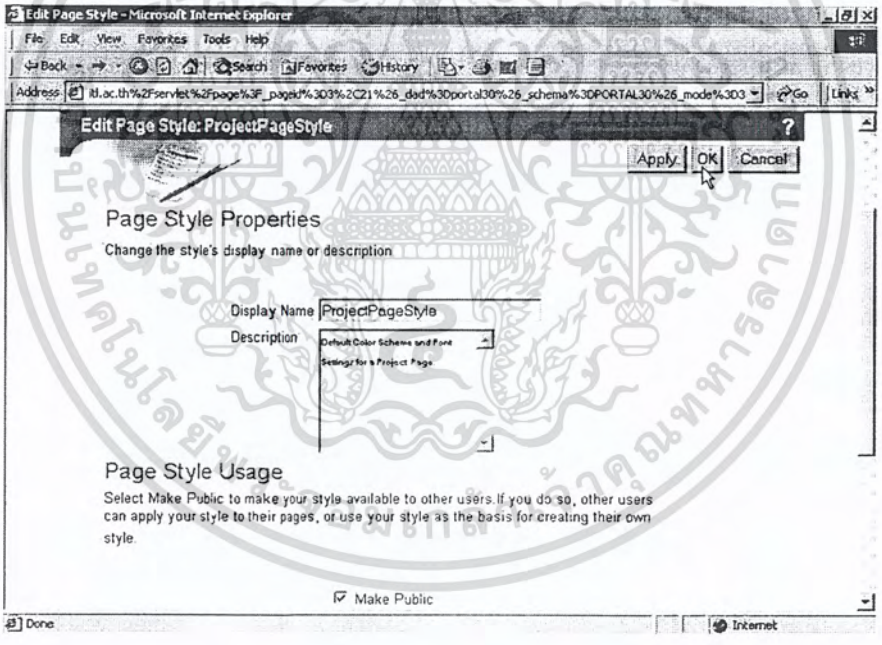
2. เพื่อความสะดวกรวดเร็ว ให้เลือกคัดลอก Page Style ในที่นี้เลือกคัดลอกจาก Main Page Styles



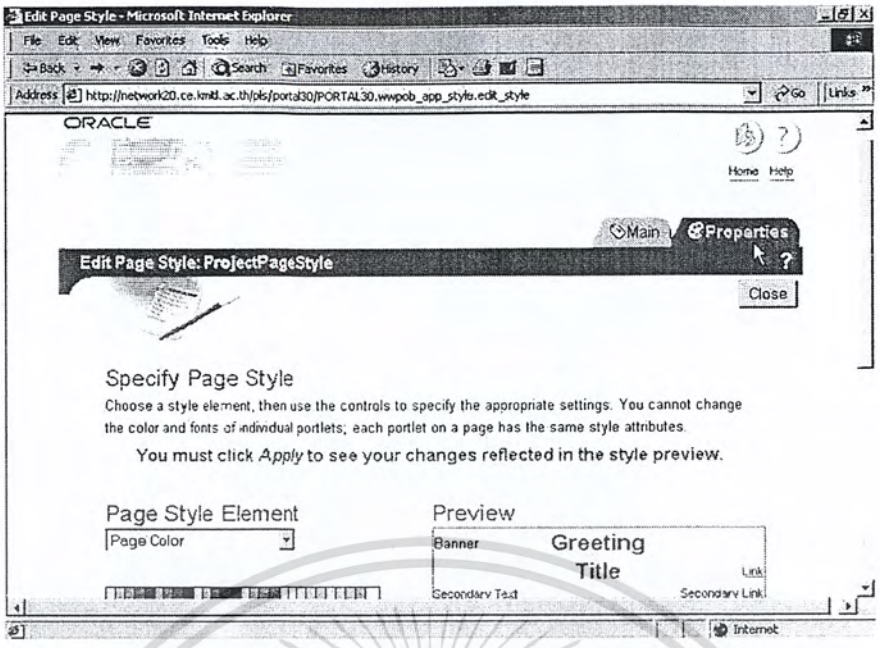
3. ป้อนชื่อในช่อง Display Name สำหรับ Page Style ใหม่



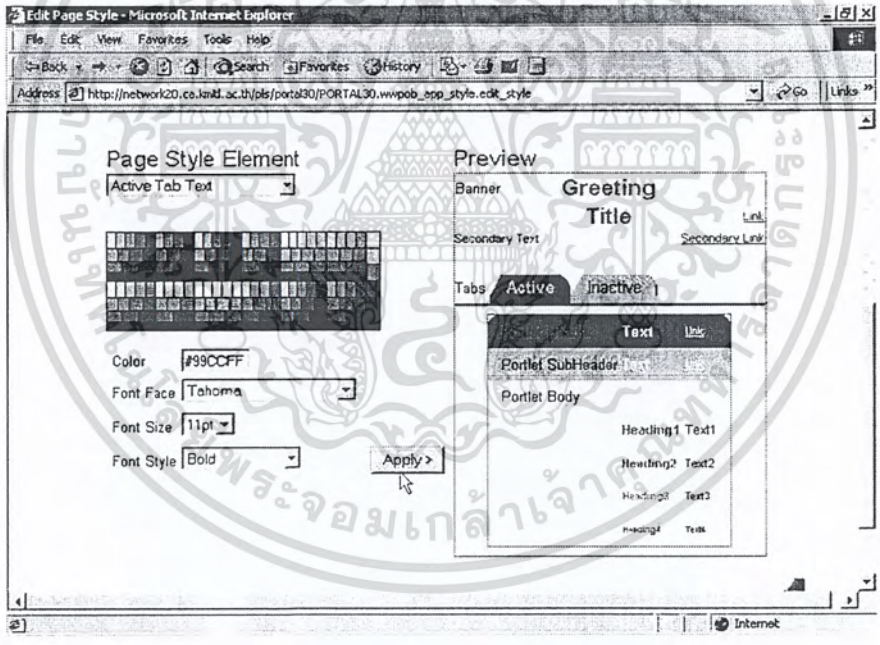
4. ทำการ Edit Page Style โดยแก้ไข ProjectPageStyle ตามต้องการ



5. เปลี่ยนแปลงชื่อและรายละเอียดได้จาก Display Name และ Description อีกทั้งสามารถกำหนดได้ว่าจะอนุญาตให้ผู้ใช้ทั่วไปสามารถนำ Page Style นี้ไปใช้ได้หรือไม่ที่ช่อง Make Public แล้วคลิก Apply

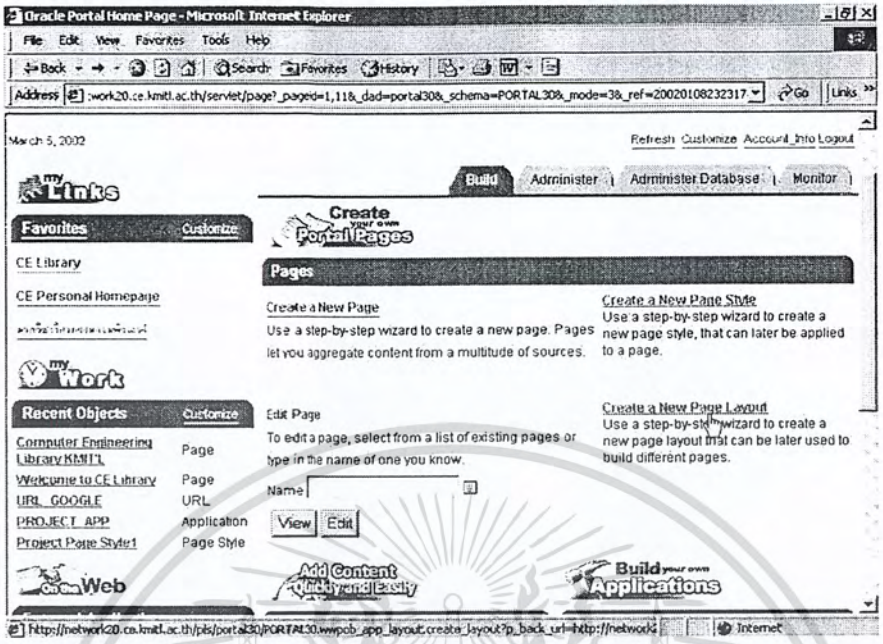


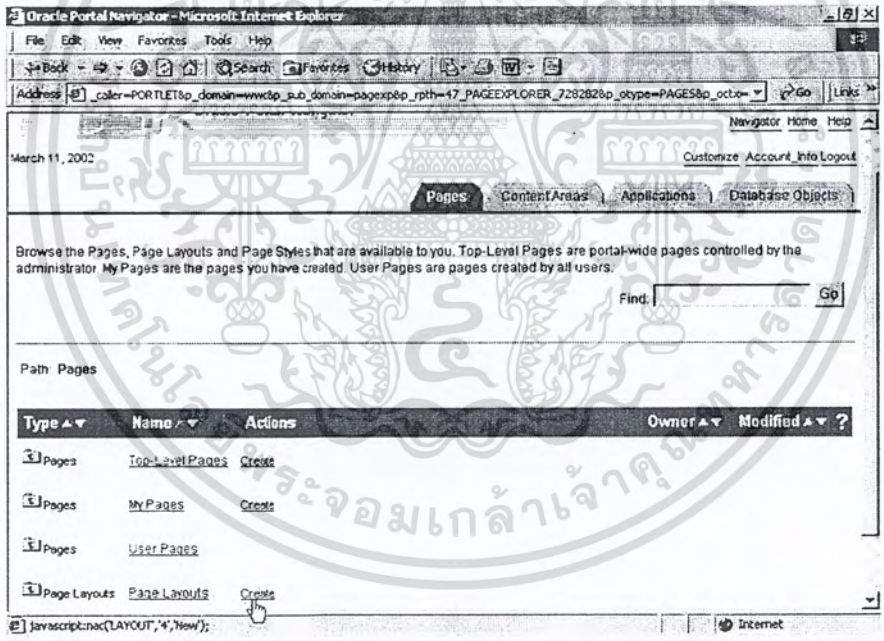
6. ที่แท็บ Properties สามารถกำหนดลักษณะต่างๆของหน้าเว็บเพจได้ตามต้องการ



7. ใน Page Style Element ทำการเลือกส่วนประกอบย่อยในเว็บเพจที่ต้องการเปลี่ยนแปลง โดยแก้ไขได้ทั้ง สี, ชนิดของตัวอักษร, ขนาดตัวอักษร และลักษณะตัวอักษร เมื่อคลิก Apply หน้าจอจะ Refresh ใหม่และแสดงผลที่ส่วน Preview ให้อัตโนมัติ

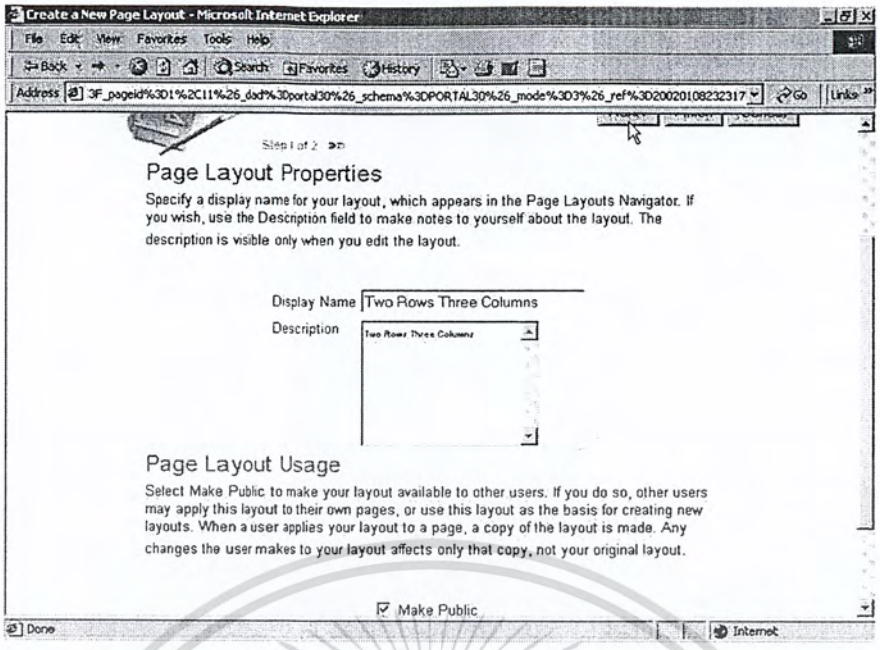
4. การสร้าง Layout



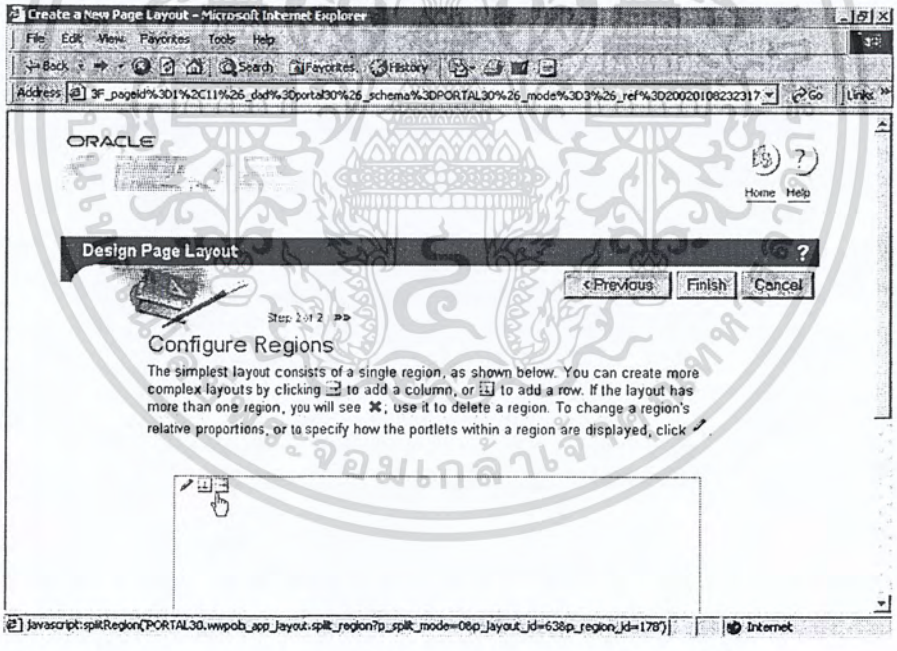


1. การสร้าง Layout สามารถทำได้ 2 วิธี คือ

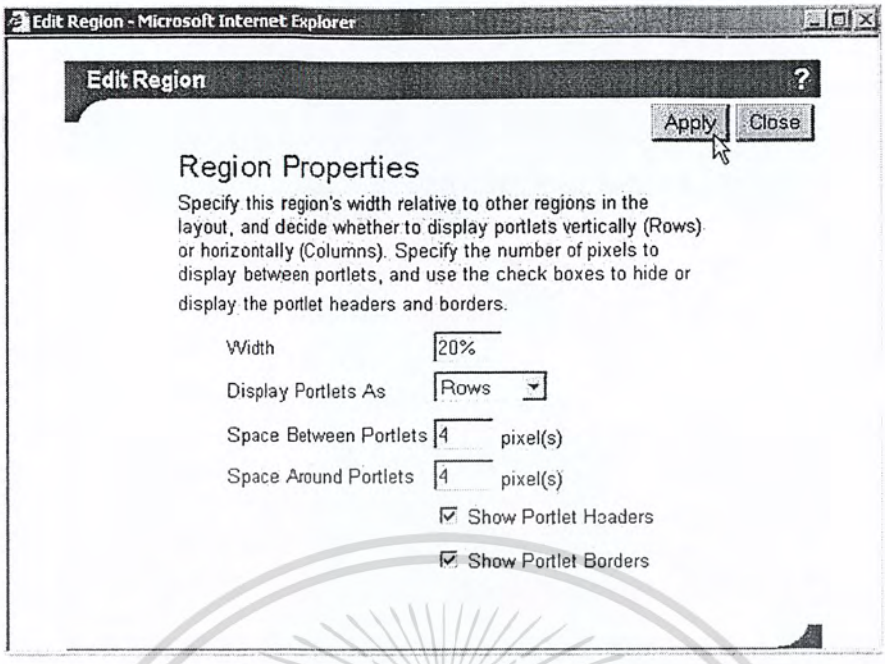
   - คลิก Create a new Page Layout จากหน้าหลัก
   - คลิก Create Page Layouts จากหน้า Navigator



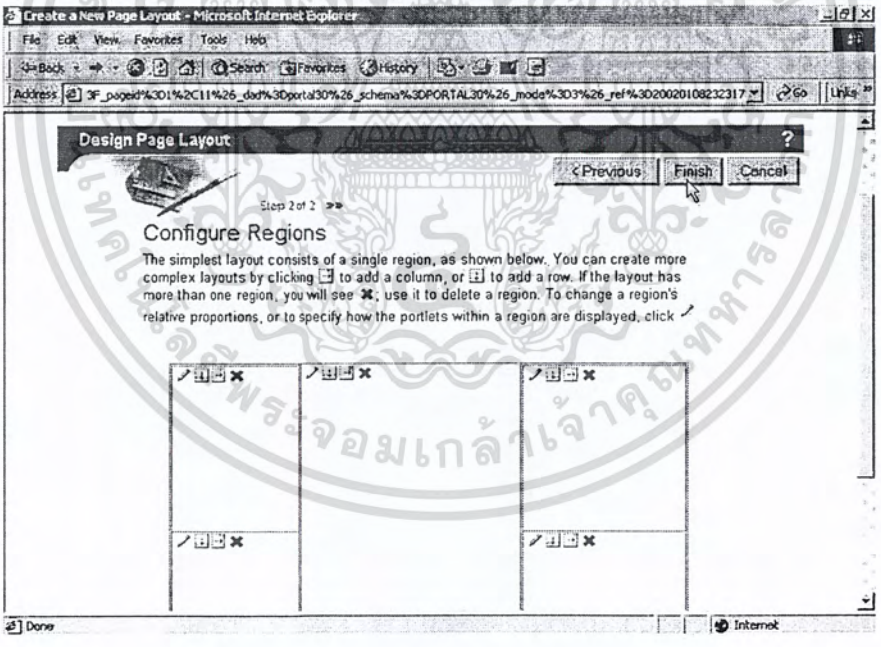
2. กำหนดชื่อในช่อง Display Name ส่วน Description จะระบุหรือไม่ก็ได้ และถ้าต้องการให้ผู้ใช้คนอื่นนำ Page Layout นี้ไปใช้งานได้ ให้ทำเครื่องหมายหน้า Make Public แล้วคลิก Next ไปยังขั้นตอนถัดไป



3. ใช้ ✎ ⊞ ⊟ เพื่อสร้าง Layout ตามต้องการ



4. เมื่อคลิก จะปรากฏหน้าจอดังรูป เพื่อให้แก้ไขความกว้างของพื้นที่ที่เลือก และกำหนดระยะห่างระหว่าง Portlet ได้ รวมถึงเลือกให้แสดงหรือไม่แสดงขอบและส่วนหัวของ Portlet ได้อีกด้วย



5. เมื่อได้ Layout ตามต้องการ คลิก Finish ก็เสร็จขั้นตอนในการสร้าง Layout

5. การนำไฟล์ JSP เพิ่มเข้าเป็น Portlet ใน Oracle9iAS Portal ผ่าน Provider

EditPlus - [D:\oracle\ora9ias\Apache\Apache\htdocs\jpdk\Management\provider.xml *]

File Edit View Search Document Project Tools Window Help

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<?providerDefinition version="2.0"?>
<!DOCTYPE provider [
<!ENTITY virtualRoot "/jpdk/">
<!ENTITY physicalRoot "D:\oracle\ora9ias\Apache\Apache\htdocs\jpdk\">
]>
<provider class="oracle.portal.provider.v1.http.DefaultProvider">
  <session>true</session>

  <portlet class="oracle.portal.provider.v1.http.DefaultPortlet">
    <id>21</id>
    <name>main_browsing</name>
    <title>Reports</title>
    <description>Main radio browsing portlet.</description>
    <timeout>100</timeout>
    <timeoutMessage>Browsing Report Portlet timeout.</timeoutMessage>
    <acceptContentType>text/html</acceptContentType>
    <renderer class="oracle.portal.provider.v1.RenderManager">
        <appPath>&virtualRoot;Management</appPath>
        <appRoot>&physicalRoot;Management</appRoot>
        <renderContainer>true</renderContainer>
        <showPage class="oracle.portal.provider.v1.http.JspRenderer">
        <name>main_browses.jsp</name>
        </showPage>
    </renderer>
  </portlet>
</provider>
```

1. ทำการสร้างไฟล์ provider.xml ตามในรูป โดยแก้ไขรายละเอียดให้สอดคล้องกับไฟล์ .jsp

EditPlus - [zone.properties]

File Edit View Search Document Project Tools Window Help

```
##############################################################################
#                  Servlet Zone Configuration File                           #
##############################################################################

################################WARNING#######################################
#
# Unlike normal Java properties, JServ configurations have some important
# extentions:
#
#   1) commas are used as token separators
#   2) multiple definitions of the same key are concatenated in a
#      comma-separated list.
#
##############################################################################

# List of Repositories
########################

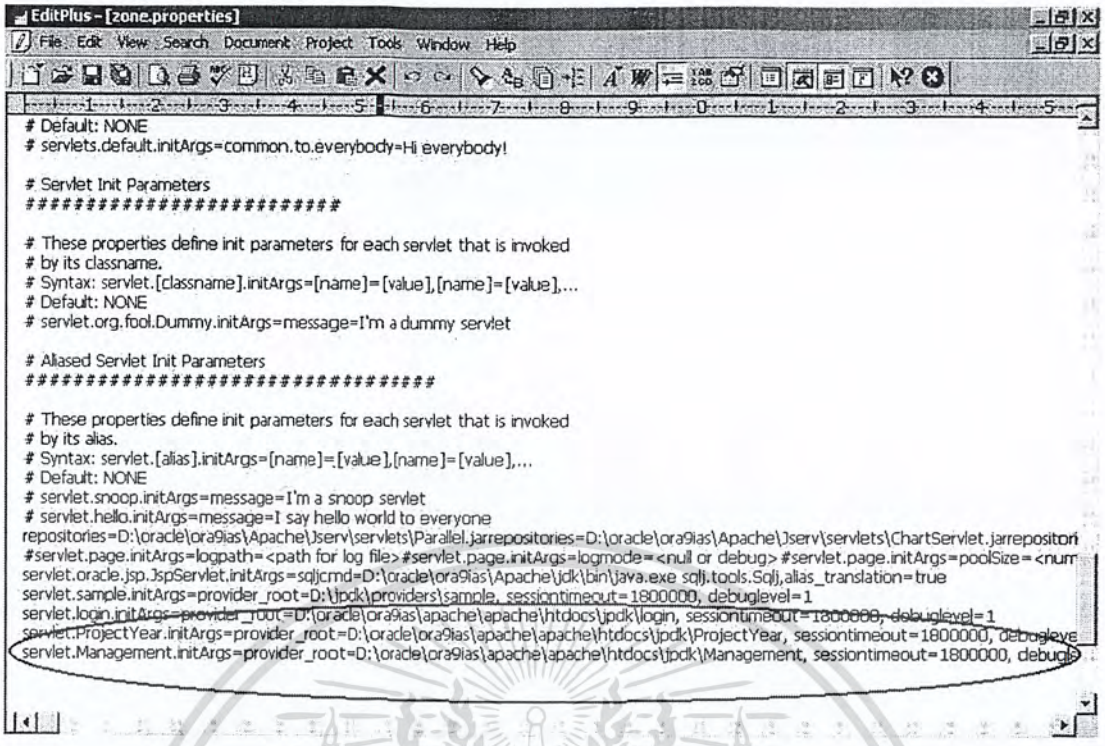
# The list of servlet repositories controlled by this servlet zone
# Syntax: repositories=[repository],[repository]...
# Default: NONE
# Note: The classes you want to be reloaded upon modification should be put
#       here.

repositories=D:\oracle\ora9ias\Apache\Jserv\servlets

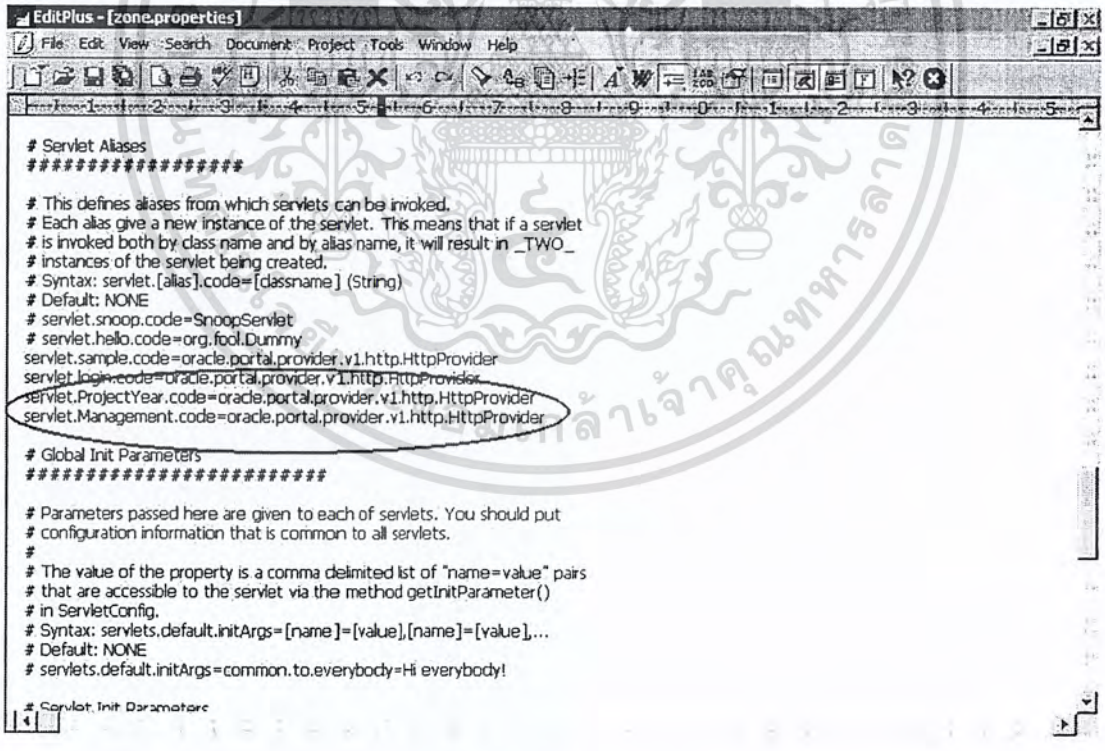
# Classloader parameters
########################

# Enable servlet class autoreloading.
# Syntax: autoreload.classes=[true,false] (boolean)
```

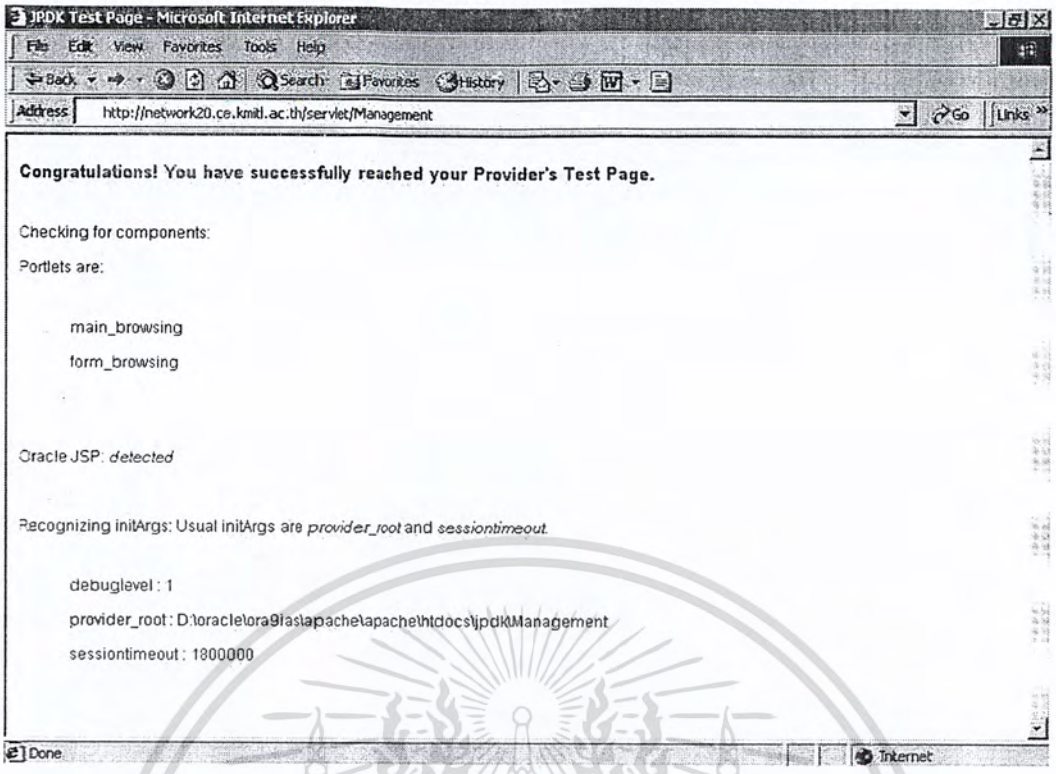
2. เปิดไฟล์ zone.properties ใน Path Oracle\ora9ias\Apache\Jserv\servlets



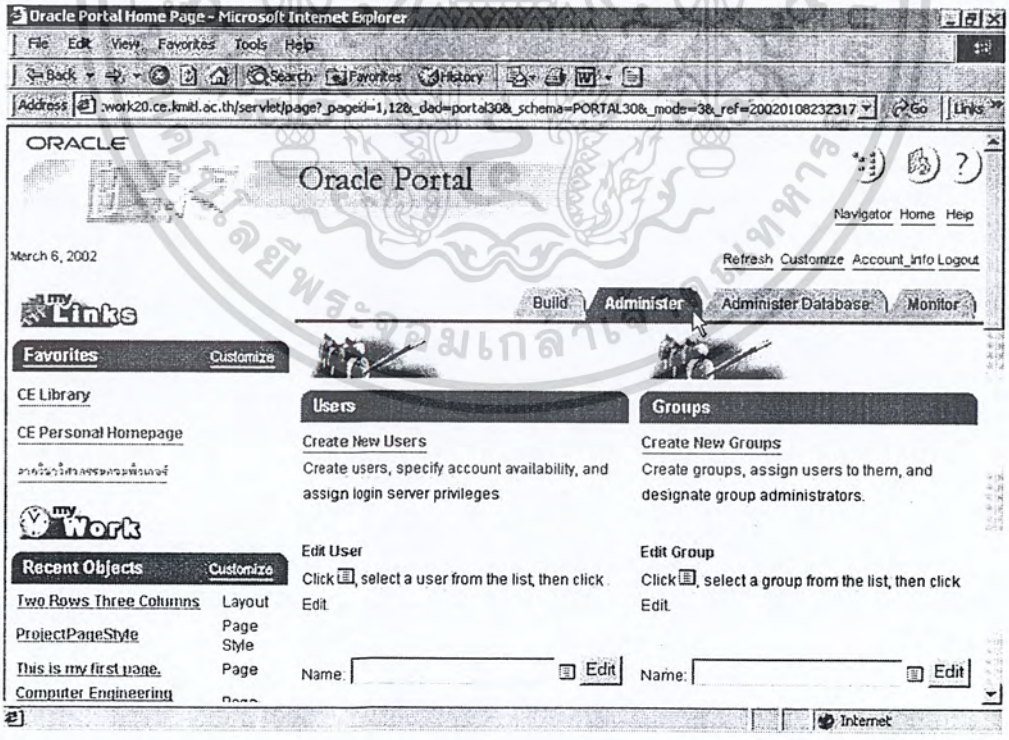
3. ส่วน # Aliased Servlet Init Parameters กำหนด Path ของไดเร็กทอรี่ที่เก็บไฟล์ provider.xml ตามรูป



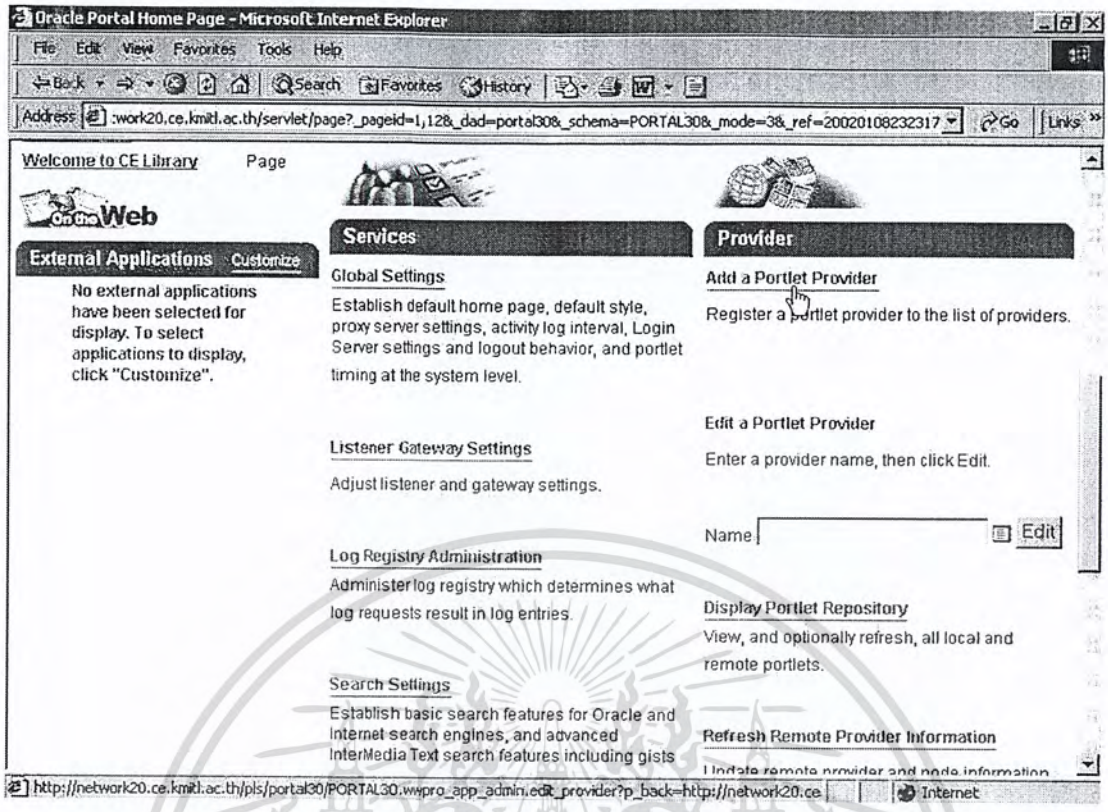
3. ส่วน # Servlet Aliases แทรก Code ดังในรูป จากนั้น Save



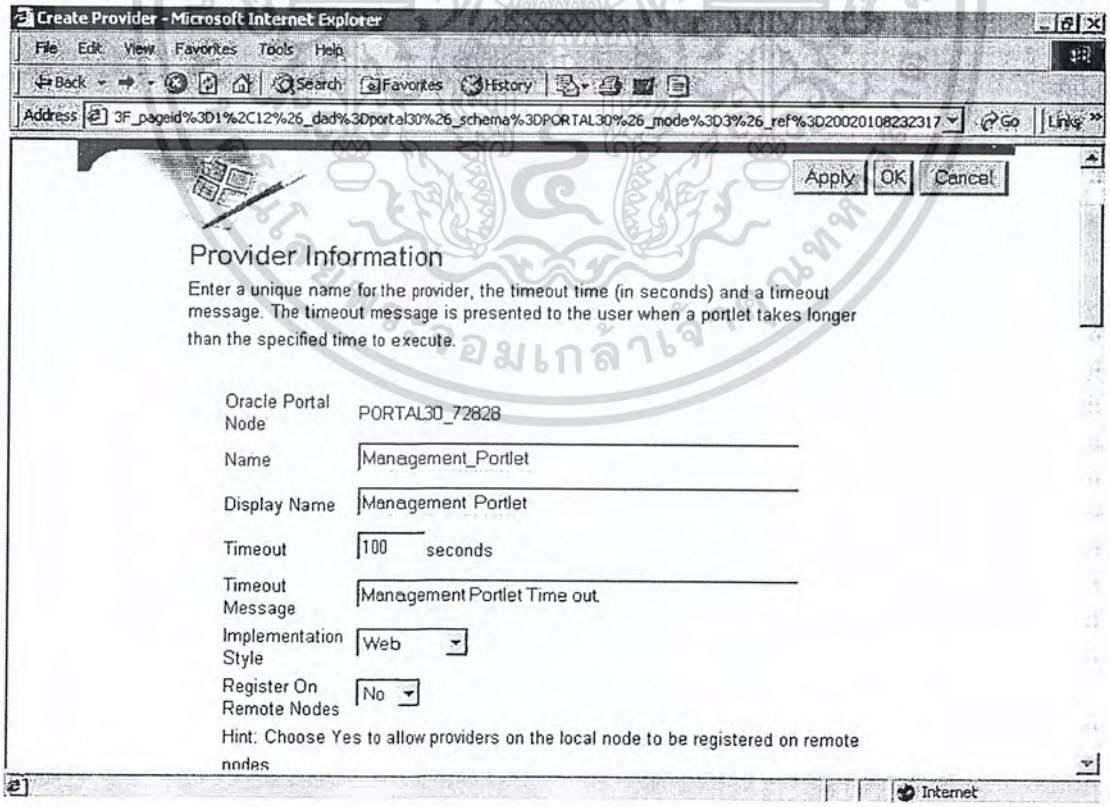
4. ทดสอบว่ากำหนดค่าถูกต้อง โดยเปิดเว็บเบราเซอร์และพิมพ์ http://<computer_name>/servlet/<your_provider_directory> ได้ผลลัพธ์ดังรูปถือว่าถูกต้อง



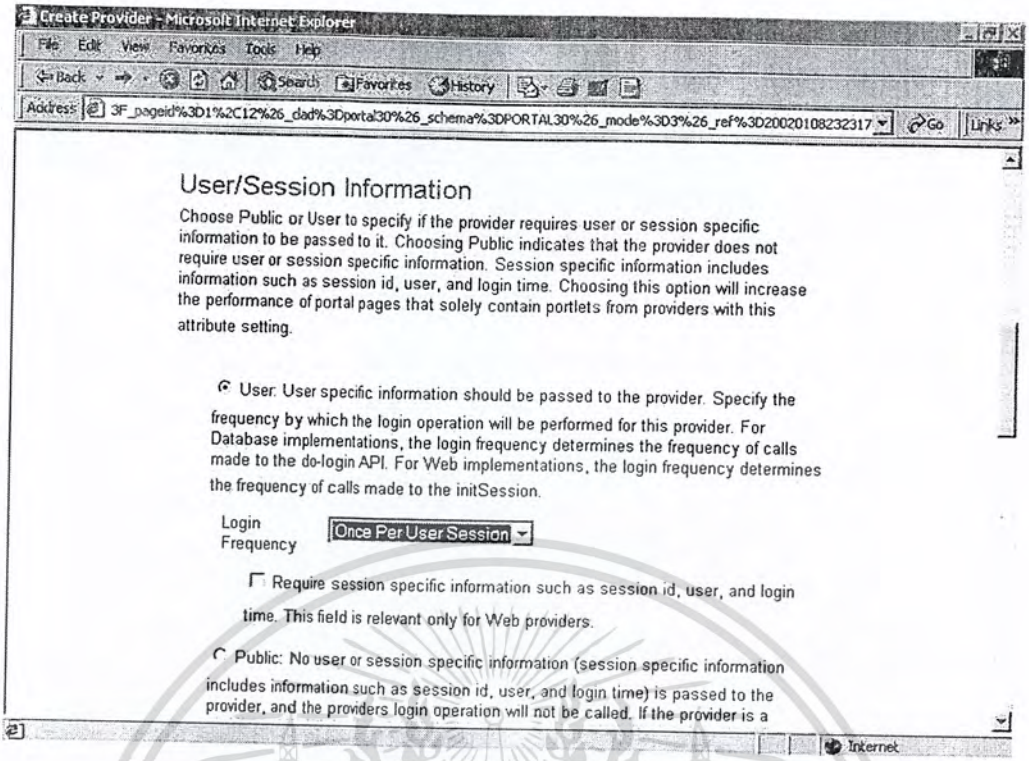
5. ไปที่หน้าหลักของเว็บเพจ เลือกแท็บ Administrator



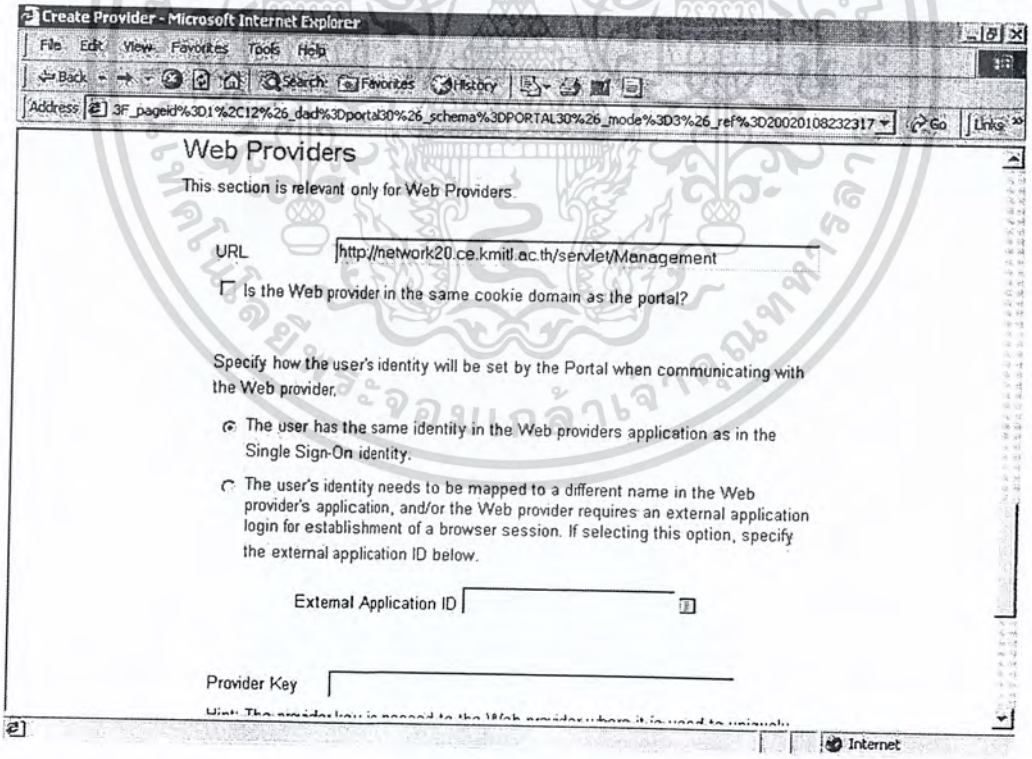
6. เลือก Add a Portlet Provider



7. ใส่ชื่อและ Display Name ที่จะให้แสดงบน Portlet Repository และกรอกข้อมูลดังรูป



8. ตั้งค่า Login Frequency เป็น Once Per User Session



9. ป้อน URL http://<computer_name>/servlet/<your_provider_directory> ที่ผ่านการทดสอบก่อนหน้า เมื่อเสร็จขั้นตอนการสร้าง Provider แล้ว ให้ลองเพิ่ม Portlet ในเว็บเพจที่สร้างขึ้น ดูวิธีได้จากเรื่อง การสร้าง Page ในตอนต้นของภาคผนวกนี้

บรรณานุกรม

หนังสืออ้างอิง

[1] Hans Bergsten (2001) : "*JavaServer Pages™, First Edition*" , O'Reilly & Associates, Inc., 2001

[2] Timothy T.Gottleber and Timothy N.Trainor (2000) : "*More Excellent HTML with an Introduction to Javascript, International Editions 2000*", McGraw-Hill, 1976

[3] Marty Hall (2000) : "*Core Servlets and JavaServer Pages*", Prentice Hall PTR, 2001

[4] ทรงพล บูรณะโอสถ, สุรชาติ พงศ์สุธนะ, "*Oracle*", บริษัท XLNC จำกัด, 2544

[5] อภิเนตร อูนากูล, "*Object-Oriented Analysis and Design*", ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2543

โฮมเพจอ้างอิง

[1] http://www.java.sun.com

[2] http://www.oracle.com

[3] http://portalstudio.oracle.com

[4] http://portalcatalog.oracle.com

[5] http://otn.oracle.com

[6] http://www.javacentrix.com

[7] http://www.openlinksw.com/licenses/appmt.htm#arc3

[8] http://article.exzilla.net/exDocs/HowtoOracleJDBC01.html

[9] http://www.computerworld.com