สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การพัฒนาซอฟต์แวร์เชิงคอมโพเน็นต์โดยใช้เทคโนโลยีดอทเน็ต

Web Services & Component Software Development Using .Net

TO55083

นายณัฐวุฒิ เนตรภิญโญ
นายเทวินทร์ จิตวัฒนชัย

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2546

เลขหมู่ 55083
เลขทะเบียน
วัน,เดือน,ปี - 8 พ.ย. 2548
b
i

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การพัฒนาซอฟต์แวร์เชิงคอมโพเน็นต์โดยใช้เทคโนโลยีดอทเน็ต

Web Services & Component Software Development Using .Net

โดย

นายณัฐวุฒิ เนตรภิญโญ

นายเทวินทร์ จิตวัฒนชัย

อาจารย์ที่ปรึกษา

ดร. วรวัฒน์ ลิ้มโภคา

ดร. ชุติเมษฎ์ ศรีนิลทา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2546

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การพัฒนาซอฟต์แวร์เชิงคอมโพเน็นต์โดยใช้เทคโนโลยีดอทเน็ต

Web Services & Component Software Development Using .Net

ผู้จัดทำ นายณัฐวุฒิ เนตรภิญโญ รหัส 43010136

นายเทวินทร์ จิตวัฒนชัย รหัส 43010153

อาจารย์ที่ปรึกษา

(ดร.วรวัฒน์ ลิ้มโภคา)

อาจารย์ที่ปรึกษา

(ดร.ชุติเมษฎ์ ศรีนิลทา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การพัฒนาเว็บเซอร์วิสแอพพลิเคชัน โดยใช้ด็อทเน็ตเทคโนโลยี

ณัฐวุฒิ เนตรภิญโญ
เทวินทร์ จิตวัฒนชัย
ดร. วรวัฒน์ ลิ้มโภคา อาจารย์ที่ปรึกษา
ดร. ชุติเมษฎ์ ศรีนิลทา อาจารย์ที่ปรึกษา
ปีการศึกษา 2546

บทคัดย่อ

ในปัจจุบันซอฟต์แวร์ที่พัฒนาขึ้นมามักมีปัญหาทางด้านการแก้ไขข้อบกพร่อง การบำรุงรักษา รวมไปถึงความน่าเชื่อถือในการใช้งาน ถึงแม้ว่าจะมีการปรับเปลี่ยนมาเป็นการพัฒนามาเป็นการพัฒนาเชิงวัตถุแล้วก็ตาม แต่ยังไม่สามารถตอบสนองความต้องการพัฒนาซอฟต์แวร์ที่ต้องการความรวดเร็วในการพัฒนา ซึ่งทำให้เกิดแนวความคิดใหม่ของการนำกลับมาใช้ใหม่ ซึ่งมีประสิทธิภาพสูงกว่า และเชื่อถือได้มากกว่ารูปแบบเดิม เทคโนโลยี .NET เป็นเทคโนโลยีใหม่ที่มีความาสามารถในการรองรับการพัฒนาแอพพลิเคชั่นเชิงคอมโพเน็นต์ ทั้งในรูปแบบของ .NET คอมโพเน็นต์และ เว็บเซอร์วิส ซึ่งเว็บเซอร์วิสช่วยให้ซอฟท์แวร์ที่พัฒนาสามารถที่จะเรียกใช้งานข้ามแพลตฟอร์มได้

ตัวอย่างการพัฒนาระบบงานที่ปริญญานิพนธ์ฉบับนี้นำเสนอคือ ระบบการซื้อขายบ้านสำเร็จรูป และที่ดิน, ระบบการซื้อขายอุปกรณ์ที่ใช้ภายในบ้าน, ระบบการให้บริการเกี่ยวกับการซื้อขายแบบบ้าน และการสร้างบ้าน โดยระบบตรงกลางเป็นระบบซึ่งพัฒนาโดยเทคโนโลยี .NET ทำการเรียกใช้เซอร์วิสจากระบบการซื้อขายบ้านสำเร็จรูป และที่ดิน, ระบบการซื้อขายอุปกรณ์ที่ใช้ภายในบ้าน, ระบบการให้บริการเกี่ยวกับการซื้อขายแบบบ้าน และการสร้างบ้าน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Web Services Application Development by .NET Technology

Mr. Nuttawut Natepinyo
Mr. Tawin Jitwattanachai
Dr. Vorawat Limpoka Advisor
Dr. Shutimet Srinilta Advisor
Academic Year 2003

ABSTRACT

Recently, software development has a problem such as updating application, maintenance and reliability. Object-oriented development cannot solve these problems completely. It also cannot meet requirement of rapid software development. So the concept of "Reusability" that is more efficiency and reliability than legacy system was used.

This thesis is study in developing system by Component Based Development Methodology using .NET Web services. The central application that consist of many called services such as Land and House, HomeForYou., Home and Décor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลาย ๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ อาจารย์ วรวัฒน์ ลิ้มโภคา และอาจารย์ชุติเมษฎ์ ศรีนิลทา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา นอกจากนั้นยังได้ให้ความรู้และแนวคิดในการทำงานรวมถึงการใช้ชีวิต อันมีค่ายิ่งให้แก่ผู้เขียน ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

ขอขอบคุณพี่ ๆ เพื่อนๆ ห้อง OLALA และพี่ๆปริญญาโท ที่ช่วยเป็นกำลังใจเป็นที่ปรึกษาและให้ประสบการณ์ในการใช้ชีวิตในมหาวิทยาลัยที่มีค่ายิ่งใช้กับผู้เขียน

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

ณัฐวุฒิ เนตรภิญโญ
เทวินทร์ จิตวัฒนชัย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้าที่
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญตาราง	XII
สารบัญรูป	XIII
บทที่ 1 บทนำ	
1.1 ความสำคัญและที่มา	1
1.2 วัตถุประสงค์ของโครงงาน	1
1.3 ขอบเขตของโครงงาน	2
1.4 วิธีการดำเนินงาน	2
1.5 ประโยชน์ที่คาดว่าจะได้รับ	3
บทที่ 2 ทฤษฎีคอมโพเน็นต์	
2.1 คอมโพเน็นต์	4
2.1.1 เฟรมเวิร์ก	4
2.1.2 ส่วนประกอบของคอมโพเน็นต์	4
2.1.3 คุณสมบัติของคอมโพเน็นต์	5
2.2 งานที่เหมาะสมกับการพัฒนาเชิงคอมโพเน็นต์	8
2.3 ประโยชน์ของการพัฒนาเชิงคอมโพเน็นต์	8
บทที่ 3 เทคโนโลยี COM+	
3.1 COM+	9
3.2 บริการของคอมโพเน็นต์	10
3.3 รูปจำลองของ COM+	10
3.4 สิ่งที่ต้องคำนึงถึงในการสร้างมิดเดิ้ลเทียร์แอพพลิเคชั่น	12
3.5 Declarative, Attribute-Based Programming	13
3.6 COM+ Catalog	13
3.7 แนวทางของ COM+	15
3.7.1 แนวทางของ COM	15

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7.2 แนวทางของ COM+ 15
3.7.2.1 แอพพลิเคชัน 15
3.7.2.2 คอมโพเน็นต์ 15
3.7.2.3 คอมโพเน็นต์ที่ถูกปรับแต่งและไม่ถูกปรับแต่ง 16
3.7.3 ชนิดของ COM+ Application 16
3.8 สถาปัตยกรรมของ COM+ 16
3.8.1 Context 16
3.8.2 Activation 17
3.8.3 Interception 18
3.8.4 Just-in-Time (JIT) Activation 19
3.9 COM+ Threading Model 20
3.10 ทรานส์แอ็กชันใน COM+ 21
3.11 COM+ และ Scalability 24
3.11.1 Microsoft Clustering Technology 24
3.11.2 Object Pooling 24
3.12 โมเดลความปลอดภัยของ COM+ (COM+ Security Model) 26
3.12.1 เป้าหมายหลักของโมเดลรักษาความปลอดภัยใน COM+ 26
3.12.2 COM+ Security Packages 27
3.12.3 Declarative Security 28
3.12.4 การปรับแต่งคอมโพเน็นต์แต่ละตัว 30
3.12.5 Role 31
3.12.5.1 การสร้าง Role 31
3.12.5.2 การเพิ่มผู้ใช้เข้าไปในโรล 31
3.12.6 การกำหนดความปลอดภัยให้แต่ละส่วนของแอพพลิเคชั่น 31
3.12.7 Programmatic Security 32
บทที่ 4 เว็บเซอร์วิส
4.1 XML 34
4.1.1 แนะนำ XML 34
4.1.2 ไวยากรณ์ของ XML 34
4.1.3 XML Element 36

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.4 XML Attribute 38
4.1.5 XML Validation 40
4.1.6 แนะนำ DTD 40
4.1.6.1 โครงสร้างของ XML 42
4.1.6.2 การกำหนด Element ใน DTD 42
4.1.6.3 การกำหนด Attribute ใน DTD 45
4.1.6.4 การกำหนด Entity ใน DTD 48
4.1.7 XML Namespace 49
4.1.8 XML PCDATA and CDATA 52
4.1.8.1 PCDATA 52
4.1.8.2 CDATA 53
4.1.9 XML Parser 54
4.1.9.1 DOM 54
4.1.9.2 SAX 55
4.1.10 ประโยชน์จาก XML 56
4.1.11 บทสรุปของ XML 57
4.2 SOAP 57
1.4.2แนะนำ SOAP 57
2.4.2ส่วนประกอบของ SOAP 57
4.2.2.1 Envelope 60
4.22..2 Body 60
4..22.3 Header 60
42..3 SOAP Fault Element 61
42..4 SOAP Encoding 62
4.2.4.1 ตัวอย่าง Struct 63
4.2.4.2 ตัวอย่าง Array 64
4.2.5 SOAP ใน HTTP 66
4.2.6 SOAP สำหรับ RPC 67
4.2.7 SOAP Toolkit 67
4.2.8 โครงสร้างภายในของ SOAP Toolkit ของ Microsoft 68
4.2.8.1 การสร้าง SoapClient Object 68

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.8.2 การประมวลผลภายใน SoapClient Object 70
4.2.8.3 การสร้าง SoapServer Object 70
4.2.8.4 การประมวลผลภายใน SoapServer Object 72
4.2.9 การสร้าง SOAP Client และ SOAP Server ด้วย .NET Framework 73
4.2.9.1 การใช้ ASP.NET ทำ SOAP เว็บเซอร์วิส 73
4.2.9.2 การใช้ .NET ทำ SOAP Client 73
4.3 UDDI 74
4.4 WSDL 74
บทที่ 5 ASP .NET
5.1 ASP คืออะไร 77
5.2 การเปลี่ยนแปลงจาก ASP ไปสู่ ASP .NET 77
5.3 เซิร์ฟเวอร์คอนโทรล (Server Control) 79
5.4 วาริเดชั่นคอนโทรล (Validation Controls) 80
5.5 ภาษาที่ใช้เขียน ASP .NET 80
5.6 รู้จักกับ System.Web.UI Namespace 80
5.6.1 คลาส Control 81
5.6.1.1 คุณสมบัติของ Control 81
5.6.1.2 เมธอดในคลาส Control 81
5.6.2 คลาสเพจ 82
5.6.2.1 ตัวอย่างคุณสมบัติและเมธอดของคลาสเพจได้แก่ 82
5.7 การพัฒนาเว็บแอพพลิเคชันด้วย ASP.NET 83
5.7.1 ส่วนประกอบคอมโพเนนต์ของ Web Form 83
5.7.2 วงจรชีวิตของ Web Form 84
5.7.2.1 การกำหนดรูปแบบ 85
5.7.2.2 การถือครองอีเวนต์ 86
5.7.2.3 การจบการทำงาน 86
บทที่ 6 ASP .NET เว็บเซอร์วิส
6.1 องค์ประกอบในการใช้งาน .NET เว็บเซอร์วิส 87
6.1.1รูปแบบการเชื่อมต่อ 87
6.1.2 WSDL 88

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.1.3 Web Services Discovery 88

6.1.4 UDDI (Universal Description, Discovery and Integration) 88

6.2 การสร้าง ASP.NET เว็บเซอร์วิส 88

6.2.1 ขั้นตอนในการสร้างเว็บเซอร์วิส 88

6.2.2 ตัวอย่างเว็บเซอร์วิส 89

6.2.3 การกำหนดค่าแอททริบิวต์ต่างๆให้แก่เว็บเซอร์วิส 89

6.3 การเรียกใช้งานเว็บเซอร์วิส 90

6.4 การจัดการเสตทในเว็บเซอร์วิส 91

6.4.1 แอพพลิเคชันอ็อบเจ็กต์ 91

6.4.2 เซสชั่นอ็อบเจ็กต์ 91

6.4.3 โหมดการจัดการเซสชั่นและการคอนฟิกค่า 92

6.4.4 การเรียกใช้งาน แอพพลิเคชัน และ เซสชั่นเสตทใน เว็บเซอร์วิส 92

6.5 ทรานแซกชัน 93

6.5.1 เหตุผลที่ต้องใช้ทรานแซกชัน 93

6.5.2 ประโยชน์ของทรานแซกชัน 96

6.5.2.1 ACID Properties 97

6.5.3 ทรานแซกชัน โมเดล 98

6.5.4 Transactional Isolation 99

6.5.5 ทรานแซกชันแบบกระจาย 101

6.5.6 ทรานแซกชันใน .NET 101

6.5.6.1 การจัดการทรานแซกชันแบบจัดการด้วยตนเอง (Manual Transaction) 101

6.5.6.2 การจัดการทรานแซกชันแบบการจัดการอัตโนมัติ (Automatic Transaction) 102

6.5.7 การจัดการทรานแซกชันใน เว็บเซอร์วิส 102

6.5.7.1 ตัวอย่างการใช้งาน เว็บเซอร์วิสที่มีการใช้ทรานแซกชัน 103

6.5.7.2 การเรียกใช้เว็บเซอร์วิสที่มีทรานแซกชัน 104

6.5.7.3 ปัญหาในการเรียกใช้ทรานแซกชันเว็บเซอร์วิส 104

6.5.7.4 แนวทางการแก้ไขปัญหา 105

6.6 การเรียกใช้เว็บเซอร์วิสแบบอซิงโครนัส(Asynchronous) 105

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.6.1 การประมวลผลแบบซิงโครนัส(Synchronous Processing) 105
6.6.2 การประมวลผลแบบอซิงโครนัส(Asynchronous Processing) 106
6.6.3 การเรียกใช้แบบอซิงโครนัสใน .NET 106
6.6.4 การเรียกใช้งานเว็บเซอร์วิสแบบอซิงโครนัส 107
6.6.5 ปัญหาที่เกิดขึ้นกับการนำไปใช้งานจริง 108
บทที่ 7 ADO .NET
7.1 สถาปัตยกรรมของ ADO.NET 110
7.2 องค์ประกอบของคอมโพเนนต์ใน ADO.NET 110
7.3 สรุปข้อเปรียบเทียบระหว่าง ADO และ ADO.NET 113
บทที่ 8 Microsoft .NET Technology
8.1 รู้จักกับ Microsoft .NET 114
8.2 .NET Platform 114
8.3 .NET Framework 115
8.3.1 Common Language Runtime (CLR) 115
8.3.1.1 Memory Management 117
8.3.1.2 Common Type System 117
8.3.2 .NET Framework Class Library 118
8.3.3 Presentation Layer 118
8.4 .NET Compact Framework 119
บทที่ 9 แอพพลิเคชั่นระบบ Land and House
9.1 Overview ของระบบ Land and House 120
9.2 Use Case Diagram ระบบ Land And House 120
9.3 Sequence Diagram ระบบ Land And House 121
9.3.1 สมัครสมาชิก 121
9.3.2 การล็อกอิน 122
9.3.3 การค้นหาบ้านและที่ดินตามเงื่อนไข 123
9.3.4 การซื้อสินค้า 124
9.3.5 การดูรายการสั่งซื้อสินค้า 125
9.3.6 การยกเลิกรายการสั่งซื้อ 126
9.3.7 การชำระเงิน 127
9.4 Class Diagram ระบบ Land and House 128

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.5 Database Diagram ของระบบ Land And House 129
9.6 คู่มือการใช้งานระบบ Land And House 130
9.6.1 ขั้นตอนเริ่มการใช้งาน 130
9.6.2 ขั้นตอนการ Signup 130
9.6.3 ขั้นตอนการ Login 132
9.6.4 ขั้นตอนการ Search 135
9.6.5 ขั้นตอนการซื้อสินค้า 136
9.6.6 ขั้นตอนการยืนยันในการสั่งซื้อ 139
9.6.7 ขั้นตอนการดูรายละเอียดของสินค้าที่ได้ซื้อไป 139
9.6.8 ขั้นตอนการ cancel การสั่งซื้อ 140
9.7 ฟังก์ชันที่เปิดให้บริการเป็นเว็บเซอร์วิส 143
9.7.1. การสมัครสมาชิกของลูกค้าใหม่ 143
9.7.2. เป็นการตรวจสอบว่า ชื่อและพาสเวิร์ดได้มีคนสมัครไปแล้วหรือยัง 143
9.7.3. การหารหัสของสินค้าที่ชื่อที่เราต้องการทราบ 144
9.7.4. ทำการสั่งซื้อสินค้าตามรายการ 144
9.7.5. ดูใบเสร็จสินค้าที่ต้องชำระเงิน 144
9.7.6.ดูรายละเอียดของบริษัทสินเชื่อ 144
9.7.7.การตกลงที่จะซื้อสินค้าชนิดนั้น 144
9.7.8.การยกเลิกการสั่งซื้อ 144
9.7.9.การจ่ายเงิน 144
9.7.10.การแสดงรายละเอียดของการจ่ายเงิน 145
บทที่ 10 แอพพลิเคชั่นระบบ Olala Home Services
10.1 Overview ของระบบ Olala Home Services 146
10.2 Use Case Diagram ระบบ Olala Home Services 148
10.3 Sequence Diagram ของระบบ Olala Home Services 149
10.3.1 การสมัครสมาชิก 149
10.3.2 การล็อกอินเข้าสู่ระบบ 150
10.3.3 การค้นหาเฟอร์นิเจอร์ 151
10.3.4 การค้นหาแบบแปลนบ้าน 152
10.3.5 การค้นหาบ้านและที่ดิน 153

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3.6 การสร้างบ้าน 154
10.3.7 การซื้อเฟอร์นิเจอร์ 155
10.3.8 การซื้อแบบแปลนบ้าน 156
10.3.9 การซื้อบ้านและที่ดิน 156
10.3.10 การยกเลิกรายการสั่งซื้อบ้านและที่ดิน 158
10.3.11 การดูบิลของการสั่งซื้อเฟอร์นิเจอร์ 159
10.3.12 การดูรายการการสั่งซื้อบ้านและที่ดิน 160
10.4 คู่มือการใช้งานระบบ Olala Home Services 160
10.4.1 ขั้นตอนเริ่มการใช้งาน 160
10.4.2 ขั้นตอนการ Signup 160
10.4.3 ขั้นตอนการ search décor 163
10.4.4 ขั้นตอนการซื้อ Décor 163
10.4.5 ขั้นตอนการ Search plan 171
10.4.6 ขั้นตอนการสร้างบ้าน 171
10.4.7 ขั้นตอนการ Search บ้านสำเร็จรูปและที่ดิน 174
10.4.8 ขั้นตอนการซื้อบ้านสำเร็จรูปและที่ดิน 174
10.4.9 ขั้นตอนการยืนยันในการสั่งซื้อ 177
10.4.10 ขั้นตอนการดูรายละเอียดของสินค้าบ้านสำเร็จรูปและที่ดิน 178
10.4.11 ขั้นตอนการ cancel การสั่งซื้อ 179
บทที่ 11 บทวิจารณ์และสรุป
11.1 สรุปผลการดำเนินงาน 182
11.2 แนวทางการพัฒนาต่อ 182
ภาคผนวก ก 184
บรรณานุกรม 200

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
ตารางที่ 4-1 ชนิดของ Attribute ใน XML	45
ตารางที่ 4-2 ค่าดีฟอลต์ของ Attribute ใน XML	46
ตารางที่ 4-3 Entity Reference มาตรฐาน	52
ตารางที่ 4-4 XML Parser	56
ตารางที่ 4-5 SOAP Fault Element	62
ตารางที่ 4-6 ค่า faultcode	62
ตารางที่ 4-7 SOAP Toolkit	68
ตารางที่ 5 -1 ตารางเปรียบเทียบความแตกต่างระหว่าง ASP กับ ASP .NET	79
ตารางที่ 6-1ระดับของ Isolation	101

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

หน้า

รูปที่ 2-1 คอมโพเน็นต์และส่วนประกอบต่าง ๆ ของมัน 5
รูปที่ 2-2 ตัวอย่างของอินเทอร์เฟซ 6
รูปที่ 2-3 คอมโพเน็นต์และการแทนที่ 7
รูปที่ 2-4 การเพิ่มเติมคอมโพเน็นต์ 7
รูปที่ 3-1วิวัฒนาการ COM สู่ COM+ 10
รูปที่ 3-2 การทำงานของ COM+ 11
รูปที่ 3-3 COM+ Catalog 14
รูปที่ 3-4 COM+ Explorer 14
รูปที่ 3-5 ความสัมพันธ์ระหว่างอพาร์ตเม็นต์ คอนเท็กซ์ และออบเจ็กซ์ 17
รูปที่ 3-6 เรียกข้าม Context ต้องการ interceptor 18
รูปที่ 3-7 Just-in-time Activation 19
รูปที่ 3-8 ออบเจ็กซ์หลายๆตัวทำงานอยู่ในทรานส์แอ็กชั่นเดียวกัน 23
รูปที่ 3-9 ไดอะล็อก Component Properties 25
รูปที่ 3-10 Object Pooling 25
รูปที่ 3-11 การเปิดเมนูเพื่อกำหนดค่ารักษาความปลอดภัย 28
รูปที่ 3-12 การกำหนดค่าปกติของการยืนยันสิทธิ 29
รูปที่ 3-13 การกำหนดค่าปกติของการเข้าถึง 29
รูปที่ 3-14 การกำหนดค่า Identiry Security 30
รูปที่ 4-1 โครงสร้างของ DOM 54
รูปที่ 4-2 ความสัมพันธ์ของการ parse ด้วย DOM 55
รูปที่ 4-3 โครงสร้างของ SOAP 58
รูปที่ 4-4 การ Client –Side Data Flow 69
รูปที่4-5 โครงสร้างภายในของ SOAP 70
รูปที่ 4-6 Server-Side Data Flow 71
รูปที่ 4-7 โครงสร้างภายใน SOAP Server 72
รูปที่ 4-8 การใช้ ASP.NET ทำ SOAP เว็บเซอร์วิส 73
รูปที่ 4-9 Runtime Callable Wrapper 73
รูปที่4-10 การใช้ .NET ทำ SOAP client 73

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 4-11 COM callable wrapper ที่ทำให้ COM client สามารถเห็น 74
รูปที่ 5-1 ส่วนประกอบของเว็บฟอร์ม 83
รูปที่ 6-1 การติดต่อจากไคลเอนต์ไปยังเว็บเซิร์ฟเวอร์ผ่านพร็อกซี่อ็อบเจ็กต์ 90
รูปที่ 6-2 แอพพลิเคชันที่ทำงานในระบบแบบกระจาย 95
รูปที่ 6-3 การใช้งานฐานข้อมูลพร้อม ๆ กัน 96
รูปที่ 6-4 Flat Transaction 99
รูปที่ 6-5 Nested Transaction 99
รูปที่ 6-6 การติดต่อแบบซิงโครนัส 105
รูปที่ 6-7 การติดต่อแบบอซิงโครนัส 106
รูปที่ 7-1 สถาปัตยกรรม ADO.NET 111
รูปที่ 7-2 เปรียบเทียบระหว่าง SQL Server .NET Data Provider และ OLEDB .NET Data Provider 112
รูปที่ 8-1 องค์ประกอบ .NET Platform 114
รูปที่ 8-2 องค์ประกอบ .NET Framework 115
รูปที่ 8-3 รูปแบบการคอมไพล์โค้ดไปเป็น IL Code 116
รูปที่ 9-1 Use Case Diagram ระบบ Land And House 120
รูปที่ 9-2 Sequence Diagram การสมัครสมาชิกของระบบ Land And House 121
รูปที่ 9-3 Sequence Diagram การล็อกอินของระบบ Land And House 122
รูปที่ 9-4 Sequence Diagram การค้นหาบ้านและที่ดินตามเงื่อนไข ของระบบ Land And House 123
รูปที่ 9-5 Sequence Diagram การซื้อสินค้า ของระบบ Land And House 124
รูปที่ 9-6 Sequence Diagram การดูรายการสั่งซื้อ ของระบบ Land And House 125
รูปที่ 9-7 Sequence Diagram การยกเลิกรายการสั่งซื้อ ของระบบ Land And House 126
รูปที่ 9-8 Sequence Diagram การชำระเงิน ของระบบ Land And House 127
รูปที่ 9-9 Class Diagram ของระบบ Land And House 128
รูปที่ 9-10 Database Diagram ของระบบ Land And House 129
รูปที่ 9-11 การหน้าจอหลัก 130
รูปที่ 9-12 การหน้าจอแสดงการ signup 131
รูปที่ 9-13 หน้ากรอกรายละเอียดของการสมัครสมาชิกที่กรอกไม่สมบูรณ์ 131

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 9-14 หน้ากรอกรายละเอียดของการสมัครสมาชิกที่กรอกไม่สมบูรณ์ แบบที่มีการใช้ usename ที่ใช้งานแล้ว 132
รูปที่ 9-15 ส่วนที่ login เข้าสู่ระบบ 133
รูปที่ 9-16 ส่วนที่ login เข้าสู่ระบบโดยการกดปุ่ม submit 133
รูปที่ 9-17 ถ้า login เข้าสู่ระบบเรียบร้อย 134
รูปที่ 9-18 ส่วนที่ login เข้าสู่ระบบแล้วไม่สมบูรณ์ 134
รูปที่ 9-19 ส่วนที่ค้นหาข้อมูล 135
รูปที่ 9-20 การเลือกการค้นหาบ้านแบบมีสนามหญ้า 136
รูปที่ 9-21 ผลการเลือกการค้นหาบ้านแบบมีสนามหญ้า 136
รูปที่ 9-22 การเลือกดูสินเชื่อของแต่ละธนาคาร 137
รูปที่ 9-23 การเลือกดูสินเชื่อของธนาคาร 138
รูปที่ 9-24 การยืนยันในการสั่งซื้อสินค้ากับธนาคาร 138
รูปที่ 9-25 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย 139
รูปที่ 9-26 หน้าจอรายละเอียดการซื้อสินค้า 140
รูปที่ 9-27 การหน้าจอแสดงการ cancel contact 141
รูปที่ 9-28 การหน้าจอแสดงการ cancel contact เมื่อสมบรูณ์ 141
รูปที่ 9-29 การหน้าจอแสดงการ cancel contact ที่มีการผิดพลาด 142
รูปที่ 9-30 การหน้าจอแสดงการ cancel contact ที่ไม่สมบูรณ์ 143
รูปที่ 10-1 Use Case diagram ของระบบ Olala Home Services 148
รูปที่ 10-2 Sequence Diagram การสมัครสมาชิก ของระบบ Olala Home Services 149
รูปที่ 10-3 Sequence Diagram การล็อกอิน ของระบบ Olala Home Services 150
รูปที่ 10-4 Sequence Diagram การค้นหาเฟอร์นิเจอร์ ของระบบ Olala Home Services 151
รูปที่ 10-5 Sequence Diagram การค้นหาแบบแปลนบ้าน ของระบบ Olala Home Services 152
รูปที่ 10-6 Sequence Diagram การค้นหาบ้านและที่ดิน ของระบบ Olala Home Services 153

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 10-7 Sequence Diagram การสร้างบ้าน ของระบบ Olala Home Services 154
รูปที่ 10-8 Sequence Diagram การซื้อเฟอร์นิเจอร์ ของระบบ Olala Home Services 155
รูปที่ 10-9 Sequence Diagram การซื้อแบบแปลนบ้าน ของระบบ Olala Home Services 156
รูปที่ 10-10 Sequence Diagram การซื้อบ้านและที่ดิน ของระบบ Olala Home Services 157
รูปที่ 10-11 Sequence Diagram การยกเลิกรายการสั่งซื้อบ้านและที่ดิน ของระบบ Olala Home Services 158
รูปที่ 10-12 Sequence Diagram การดูบิลของการสั่งซื้อเฟอร์นิเจอร์ ของระบบ Olala Home Services 159
รูปที่ 10-13 Sequence Diagram การดูรายการการสั่งซื้อบ้านและที่ดิน ของระบบ Olala Home Services 160
รูปที่ 10-14 หน้ากรอกรายละเอียดการสมัครสมาชิก 161
รูปที่ 10-15 หน้าเมื่อสมัครสมาชิกสมบรูณ์ 161
รูปที่ 10-16 หน้าการเข้าสู่ระบบสมัครสมาชิก 162
รูปที่ 10-17 หน้าส่วนที่ค้นหาข้อมูลเกี่ยวกับ Décor 163
รูปที่ 10-18 หน้าการซื้อเตียงนอน 164
รูปที่ 10-19 หน้ารายละเอียดเกี่ยวกับการซื้อสินค้าเกี่ยวกับ Décor 165
รูปที่ 10-20 หน้าการซื้อเตียงนอน อีกครั้ง 165
รูปที่ 10-21 หน้ารายละเอียดเกี่ยวกับการซื้อสินค้าเกี่ยวกับ Decor อีกครั้ง 166
รูปที่ 10-22 หน้าการซื้อโซฟา 167
รูปที่ 10-23 หน้ารายละเอียดเกี่ยวกับการซื้อโซฟา 167
รูปที่ 10-24 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย 168
รูปที่ 10-25 link หน้ารายละเอียดของการสั่งซื้อ 169
รูปที่ 10-26 หน้ารายละเอียดของการสั่งซื้อ 169
รูปที่ 10-27 หน้าส่วนที่ค้นหาข้อมูลเกี่ยวกับ Plan 170
รูปที่ 10-28 หน้าผลส่วนที่ค้นหาข้อมูลเกี่ยวกับ Plan 171

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 10-29 หน้าส่วนที่สั่งซื้อแบบบ้าน 172
รูปที่ 10-30 หน้าส่วนที่ตกลงสั่งซื้อแบบบ้าน 173
รูปที่ 10-31 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย 173
รูปที่ 10-32 ส่วนที่ค้นหาข้อมูล 174
รูปที่ 10-33 ผลการเลือกการค้นหาบ้านแบบมีสนามหญ้า 175
รูปที่ 10-34 การเลือกดูสินเชื่อของแต่ละธนาคาร 175
รูปที่ 10-35 การเลือกดูสินเชื่อของธนาคาร 176
รูปที่ 10-36 การยืนยันในการสั่งซื้อสินค้ากับธนาคาร 177
รูปที่ 10-37 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย 178
รูปที่ 10-38 หน้าจอรายละเอียดการซื้อสินค้า 179
รูปที่ 10-39 การหน้าจอแสดงการ cancel contact ที่มีการผิดพลาด 180
รูปที่ 10-40 การหน้าจอแสดงการ cancel contact เมื่อสมบรูณ์ 181
รูปที่ ก-1 การ NewProject 184
รูปที่ ก-2 การสร้าง ASP.NET 185
รูปที่ ก-3 หน้าการออกแบบ 186
รูปที่ ก-4 การเริ่มการออกแบบ 187
รูปที่ ก-5 การเขียน code ASP.NET 188
รูปที่ ก-6 การกำหนดให้ page นั้นเป็น start page 189
รูปที่ ก-7 ผลลัพธ์ 190
รูปที่ ก-8 การ New Project Web Services 191
รูปที่ ก-9 การเขียน code 191
รูปที่ ก-10 services ที่มีใน project 192
รูปที่ ก-11 การใส่ค่าลงไปใน sevices 193
รูปที่ ก-12 ผลลัพธ์ของ Services 194
รูปที่ ก-13 การออกแบบ Page 195
รูปที่ ก-14 การ Add Web Reference 196
รูปที่ ก-15 การออกแบบ Page 197
รูปที่ ก-16 Code ที่เรียกใช้ services 198
รูปที่ ก-17 ผลลัพธ์ 199

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

เนื่องจากในปัจจุบันนี้ระบบ Internet ได้มีบทบาทในประเทศไทยเป็นอย่างมาก ทั้งในด้านการค้นหาข้อมูลและการซื้อสินค้าต่าง ๆ ผ่านทาง website ทำให้ชีวิตคนเรามีความสะดวกสบายมากขึ้นทำให้เราได้คิดโครงการนี้ขึ้นมา โครงการของเราได้สร้างมาจากสถาปัตยกรรม Microsoft Visual Studio.NET ในการพัฒนา ซึ่งเมื่อเราเขียนด้วยภาษาใดก็ตามในตระกูล Microsoft Visual Studio.NET จะได้ผลที่เหมือนกันในการพัฒนา Application ซึ่งเป็นข้อดีในการพัฒนาด้วยสถาปัตยกรรมนี้

จากแนวคิดของ Web services ทำให้เราสามารถเข้าถึงข้อมูลสารสนเทศจาก Application ที่ต่างกันเป็นไปโดยง่าย โดย Application นั้นๆ สามารถเขียนอยู่คนละสถาปัตยกรรมได้ ดังนั้นเราจึงได้นำมาประยุกต์กับโครงการของเราคือ โครงการระบบการจัดหาจัดซื้อที่อยู่อาศัย โดยที่โครงการของเราจะสามารถให้ความช่วยเหลือกับคนที่ต้องการซื้อบ้านและที่ดิน เพียงแค่ใส่ความต้องการว่าต้องการบ้านราคาประมาณเท่าไหร่ และอยู่แถวใด ระบบของเราก็จะทำการค้นหากับ website ที่ให้ services เกี่ยวกับสิ่งที่เราต้องการและ return ค่าออกมาสำเร็จรูป ทำให้เราทราบว่ามีที่ดินที่เราต้องการว่างหรือไม่และราคาไหร่ และเรายังให้บริการเกี่ยวกับสินค้าตบแต่งบ้านและการสร้างบ้านเข้ามาอยู่ในโครงการเราด้วย คือถ้าลูกค้าตกลงที่จะปลูกบ้านตรงที่ที่ดินของเราแล้ว และต้องการสินค้าตกแต่งบ้านด้วยก็สามารถกรอกรายละเอียดเกี่ยวกับ furniture ที่ต้องการจะซื้อ ทางเราจะทำการ search จาก website ที่ให้ services เกี่ยวกับสินค้าตกแต่งบ้าน และจะ return ค่าออกมาเลยและลูกค้าก็สามารถนำข้อมูลที่ได้มาประกอบการตัดสินใจในการสร้างบ้าน และถ้าต้องการที่จะสร้างบ้านกับเรา เราก็มีบริการการสร้างบ้านโดยที่เราจะให้ลูกค้าเลือกแบบต่าง ๆ กับทางโครงการของเราและเราก็จะไป search จาก site ที่ให้บริการ service นี้ แล้วจะรายงานผลให้ลูกค้าทราบเกี่ยวกับค่าใช้จ่ายสรุปเกี่ยวกับโครงการนี้ จะเห็นได้ว่าโครงการนี้เป็นการรวมเอาการขายที่ดิน สินค้าตกแต่งบ้าน และการปลูกบ้านเข้ามาด้วยกันอย่างสมบูรณ์แบบ

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อเป็นการกระตุ้นเศรษฐกิจของประเทศเรา ที่ซบเทรามานานจากพิษ IMF ในปัจจุบันนี้ธุรกิจอสังหาริมทรัพย์ ได้ซบเทราลงไปมาก
2. เพื่อที่จะอำนวยความสะดวกแก่ผู้คนที่ต้องการที่จะมีบ้านจะได้ไม่ต้องเสียเวลามากในการที่จะค้นหาข้อมูลและสามารถตัดสินใจซื้อได้เลยทันที
3. เพื่อศึกษาการทำงานของ Web services และการนำมาใช้ในระบบธุรกิจอสังหาริมทรัพย์
4. เพื่อเป็นการเพิ่มความสามารถของ E-commerce ให้มีความสามารถมากกว่ามากขึ้น
5. เพื่อเป็นการเข้าใจและสามารถที่จะประยุกต์ระบบธุรกิจของเราให้สามารถนำไปใช้กับธุรกิจแบบอื่น ๆ ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.3 ขอบเขตของของโครงงาน

1. ให้บริการเป็นศูนย์กลางในการเชื่อมต่อกับระบบการซื้อขายที่ดิน สินค้าตกแต่งบ้าน และการสร้างบ้าน โดยผ่านทางระบบ Web services

2. ลูกค้าสามารถเข้ามาเลือกซื้อสินค้าจาก Web ของเราโดยสามารถซื้อได้ครบทุกอย่างโดยผ่านทาง Web services

3. สามารถให้บริการ Service จากค่ายอื่นได้ เช่น Java ไม่จำกัดเฉพาะ .NET เท่านั้น

เนื่องจากเป็นการพัฒนาโครงงานร่วมกัน โดยต้องอาศัยพื้นฐานความรู้เดียวกัน ดังนั้นเนื้อหาและทฤษฎีบางส่วนจะมีความเหมือนกัน การแบ่งหัวข้อเพื่อแยกกันไปศึกษา จะทำให้เสียเวลา เพราะทุกกลุ่มก็ต้องศึกษาเนื้อหาเรื่องเดียวกัน ทำให้ผู้พัฒนาโครงงานได้แบ่งหัวข้อในการศึกษาให้แต่ละกลุ่มรับผิดชอบเพื่อนำความรู้ที่ได้มาใช้ร่วมกัน และ ถ่ายทอดให้กลุ่มที่เหลือได้เข้าใจตรงกัน เพื่อเป็นการประหยัดเวลา โดยหัวข้อที่แบ่ง เป็นดังนี้

กลุ่มการพัฒนาเชิงคอมโพเนนต์ ด้วย .NET กลุ่มที่ 1 (Land and House) ศึกษาเรื่องดังต่อไปนี้

1. เว็บเซอร์วิส (Web Services)
2. ADO.NET
3. .NET Framework

กลุ่มการพัฒนาเชิงคอมโพเนนต์ ด้วย .NET กลุ่มที่ 2 (Home for you) ศึกษาเรื่องดังต่อไปนี้

1. ASP.NET
2. ASP.NET and Web services
3. VRML

กลุ่มการพัฒนาเชิงคอมโพเนนต์ ด้วย EJB กลุ่มที่ 1 (Home and Decor) ศึกษาเรื่องดังต่อไปนี้

1. ทฤษฎีคอมโพเน็นต์ (Component Theory)
2. สถาปัตยกรรม J2EE

1.4 วิธีการดำเนินงาน

แบ่งเป็นขั้นตอนได้ดังนี้คือ

1.4.1 ศึกษาค้นคว้า

- ศึกษาทฤษฎีและความรู้ที่เกี่ยวข้องกับงานวิจัย และแอพพลิเคชั่นที่จะพัฒนา ได้แก่ COM+, Component, XML, SOAP, Web Services, EJB, .Net

- ศึกษา และค้นคว้าถึงความเป็นไปได้ในการนำเอาเทคโนโลยีต่างๆ มาใช้กับแอพพลิเคชั่นที่จะพัฒนา

- ศึกษาการเขียนโปรแกรม และการสร้างเว็บเซอร์วิส ด้วย Microsoft Visual Studio .Net

1.4.2

- ออกแบบ วางแผนโครงสร้างระบบต่าง ๆ ของแอพพลิเคชั่น

- กำหนดรายละเอียดของระบบ รวมทั้งการออกแบบอินเทอร์เฟส (Interface) ของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.4.3 สร้างแอพพลิเคชั่น และพัฒนา

- ทดลองสร้างเว็บเซอร์วิสอย่างง่าย เพื่อเป็นต้นแบบในการพัฒนาระบบ
- สร้างแอพพลิเคชั่นที่จะให้บริการ รวมทั้งทดสอบการเรียกใช้งานแอพพลิเคชั่น
- ศึกษาถึงปัญหาที่เกิดขึ้น พร้อมทั้งหาวิธีแก้ไข

1.5 ประโยชน์ที่คาดว่าจะได้รับ

เพื่อเป็นการศึกษาเทคโนโลยีใหม่ที่จะเป็นแนวโน้มของการพัฒนาซอร์ฟแวร์ในอนาคต เพื่อที่จะเข้าใจถึงข้อดี และข้อเสียของเทคโนโลยีนี้ และนำมาประยุกต์ใช้งานกับระบบอีคอมเมิร์ช (E-Commerce) โดยอาศัยข้อดีของเว็บเซอร์วิสที่สามารถสร้างซอร์ฟแวร์ที่เป็นอิสระต่อกัน รวมทั้งการได้ศึกษาการภาพรวมของทางธุรกิจ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีคอมโพเน็นต์

2.1 คอมโพเน็นต์

คอมโพเน็นต์ คือ ส่วนย่อยของระบบที่ไม่ขึ้นกับส่วนอื่นๆ และซ่อนรายละเอียดการทำงานไว้ภายใน โดยมีส่วนติดต่อกับภายนอกเพื่อให้สามารถเรียกใช้งานได้ ความสามารถนี้ทำให้คอมโพเน็นต์สามารถนำ กลับมาใช้ใหม่ได้ (reuse)

การพัฒนาเชิงคอมโพเน็นต์นั้น มีความหมายกับคนแต่ละกลุ่มแตกต่างกันไปตามความต้องการ ซึ่งมีลักษณะเด่นบางอย่าง ที่นิยามความเป็นระบบคอมโพเน็นต์ดังต่อไปนี้

1. คอมโพเน็นต์มีลักษณะของออบเจ็กต์ คือ การซ่อนรายละเอียด (Encapsulated), โพลีมอร์ฟิค (polymorphic), การกำหนดหน้าที่ และการกำหนดอินเทอร์เฟซ
2. คอมโพเน็นต์ออกแบบภายใต้เฟรมเวิร์ก (Framework) ซึ่งได้สร้างข้อจำกัดบางอย่างไว้ เช่น ต้องไม่มีหลายเธรด, ไม่มีการติดต่อกับภายนอกโดยไม่ผ่านบริการของเฟรมเวิร์ก
3. คอมโพเน็นต์สามารถอยู่ได้ด้วยตัวเอง โดยไม่พึ่งพาคอมโพเน็นต์อื่นๆ ยกเว้น คอมโพเน็นต์ของเฟรมเวิร์ก ที่คอมโพเน็นต์ดังกล่าวใช้อยู่
4. ทุกคอมโพเน็นต์มีอินเทอร์เฟซสามัญที่แน่นอน (Fix and common) และอินเทอร์เฟซนี้เปลี่ยนแปลงไม่ได้
5. คอมโพเน็นต์สามารถอธิบายตนเองได้ โดยอินเทอร์เฟซของคอมโพเน็นต์จะต้องมีข้อมูลมากพอ ที่สามารถทำให้ไคลเอ็นต์สามารถเข้าใจวิธีใช้คอมโพเน็นต์นั้นได้

2.1.1 เฟรมเวิร์ก คือ สภาพแวดล้อมที่มีลักษณะดังต่อไปนี้

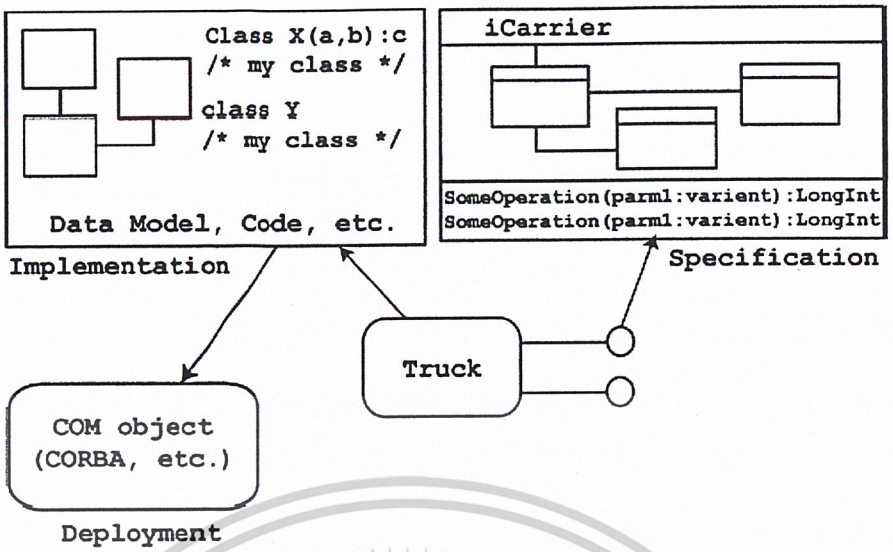
1. สร้างอินสแทนซ์ของคอมโพเน็นต์ตอนรันไทม์
2. ทำให้คอมโพเน็นต์ค้นพบคอมโพเน็นต์อื่นๆ ได้
3. ทำให้คอมโพเน็นต์ต่างๆ สามารถติดต่อกันได้
4. จัดหาบริการสามัญ เช่น เพอร์ซิสเทนซ์ (persistence), ทรานส์แอ็กชัน (transaction), ความไม่ขึ้นกับสถานที่, การรักษาความปลอดภัย, มอนิเตอริง (monitoring)

2.1.2 ส่วนประกอบของคอมโพเน็นต์

คอมโพเน็นต์จะประกอบด้วยส่วนสำคัญ 3 ส่วนดังนี้

- อินเทอร์เฟซ (interface) - คอมโพเน็นต์จะถูกเรียกใช้งานผ่านทางอินเทอร์เฟซ
- อิมพลีเมนต์เตชัน (implementation) - เป็นโค้ดที่กำหนดการทำงานของคอมโพเน็นต์
- ดีพลอยเมนต์ (deployment) – เป็นเอ็กซิคิวต์ไฟล์จะใช้ในการทำให้คอมโพเน็นต์ทำงานได้ ทำหน้าที่จัดหารันไทม์เอ็นไวรอนเมนต์ในการควบคุมการทำงานของคอมโพเน็นต์และจัดหาเซอร์วิสที่จำเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

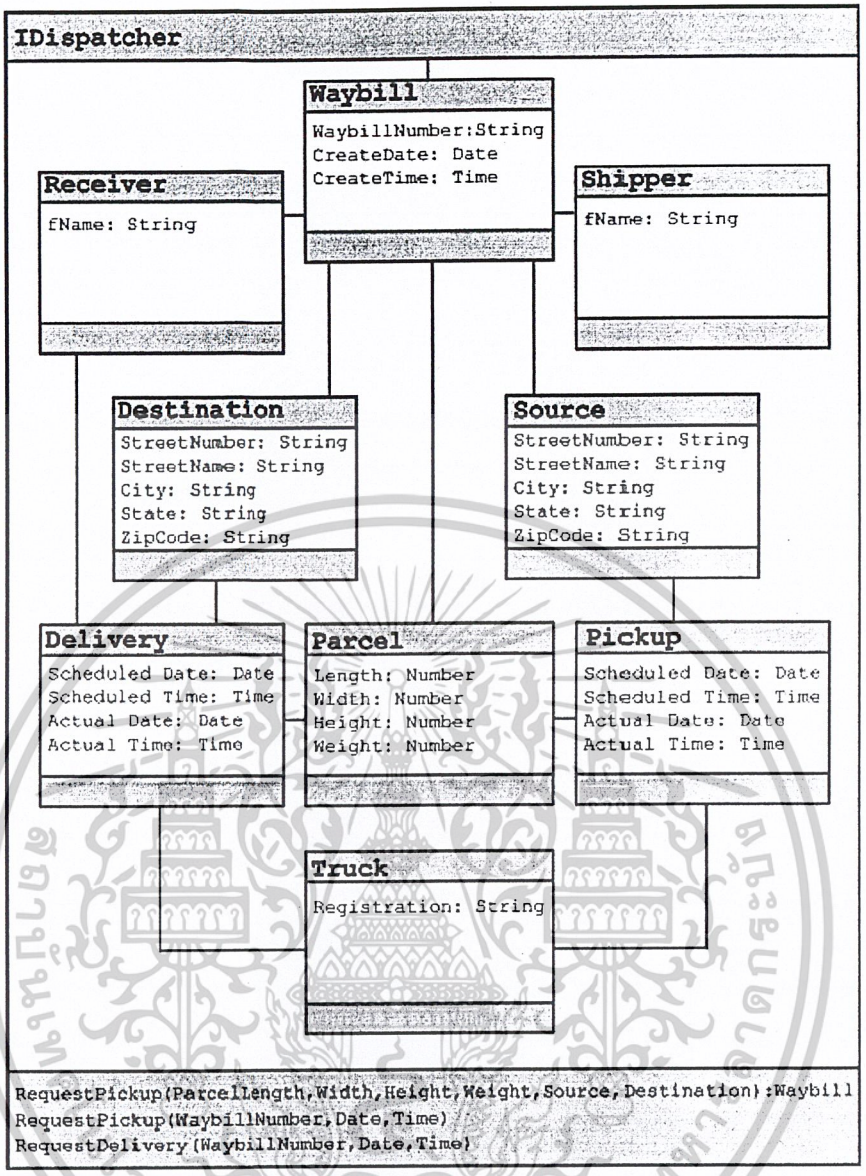


รูปที่ 2-1 คอมโพเน็นต์และส่วนประกอบต่าง ๆ ของมัน

2.1.3 คุณสมบัติของคอมโพเน็นต์

- **Encapsulated** - เป็นกระบวนการในการซ่อนโค้ดหรืออิมพลีเมนต์เตชันของคอมโพเน็นต์ ผู้ใช้จะรู้เพียง อินเทอร์เฟซและจะใช้งานผ่านมัน โดยไม่จำเป็นต้องรู้ถึงอิมพลีเมนต์เตชัน หรือการทำงาน ทำให้ การเปลี่ยนแปลงอิมพลีเมนต์เตชันไม่มีผลกระทบต่อผู้ใช้
- **Descriptive** - เนื่องจากคอมโพเน็นต์จะต้องติดต่อผ่านอินเทอร์เฟซเท่านั้น มันจะต้องมีอินฟอร์เมชันของ มันเอง ที่ผู้ใช้สามารถจะเข้าใจได้ โดยอินฟอร์เมชันนั้น จะต้องอธิบายถึง อินเทอร์เฟซ , อิมพลีเมนต์เตชัน และดีพลอยเมนต์

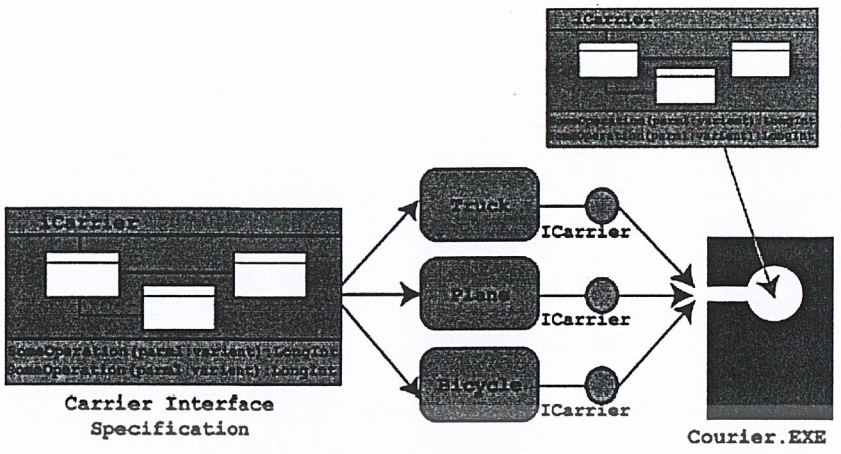
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2-2 ตัวอย่างของอินเทอร์เฟซ

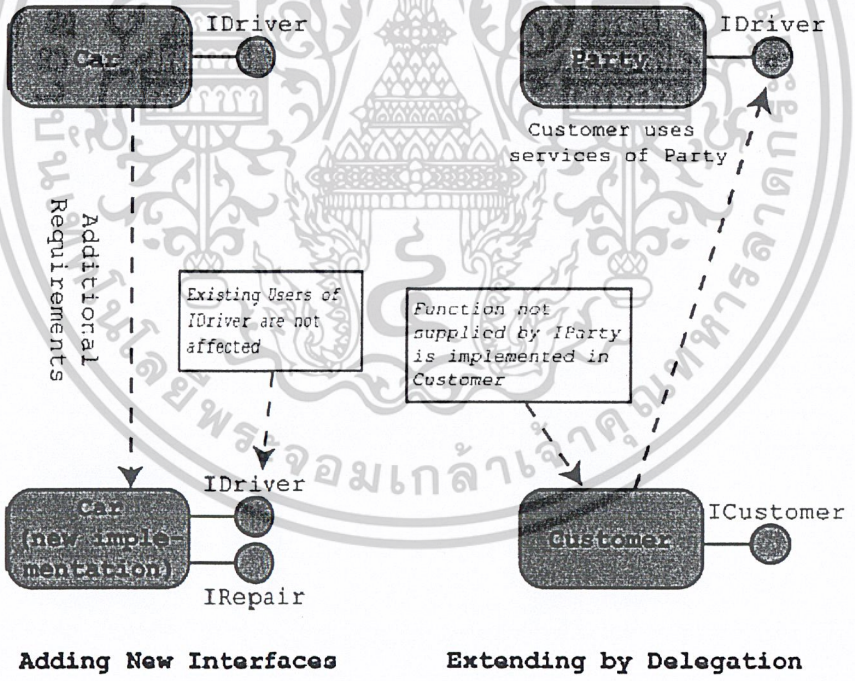
- **Replaceable** – คอมโพเน็นต์มีความสามารถในการเปลี่ยนคอมโพเน็นต์หนึ่งกับคอมโพเน็นต์ ใดๆ ที่มีอินเทอร์เฟซเหมือนกันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2-3 คอมโพเน็นต์และการแทนที่

- **Extensible** - สามารถที่จะเพิ่มความสามารถได้ มี 2 วิธีคือ การเพิ่มอินเทอร์เฟซ และมอบหมายหน้าที่ (Delegating Responsibility) การสร้างคอมโพเน็นต์ใหม่สามารถมอบหมายหน้าที่ให้กับบริการที่มีอยู่ในคอมโพเน็นต์ที่มีอยู่แล้ว



รูปที่ 2-4 การเพิ่มเติมคอมโพเน็นต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 งานที่เหมาะสมกับการพัฒนาเชิง คอมโพเน็นต์

การพัฒนาเชิงคอมโพเน็นต์ เป็นวิธีการพัฒนาคอมโพเน็นต์โดยใช้คอมโพเน็นต์เฟรมเวิร์ก ซึ่งการคงหน้าที่ของคอมโพเน็นต์ไว้อย่างถูกต้อง (Well defined responsibility) กระทำได้โดย ให้คอมโพเน็นต์เชิงธุรกิจ หลีกเลี่ยงโค้ดที่ไม่สามารถทำงานกับอินเทอร์เฟซที่แน่นอนได้

เฟรมเวิร์กคอมโพเน็นต์จะซ่อนบริการต่างๆ ของเอ็นไวรอนเม็นต์ ซึ่งบริการต่างๆ เหล่านี้เป็นมาตรฐานแล้ว เช่น message oriented middle-ware, Transaction Monitors, การรักษาความปลอดภัย และพิสูจน์สิทธิ์

2.3 ประโยชน์ของการพัฒนาเชิงคอมโพเน็นต์

การพัฒนาเชิงคอมโพเน็นต์มีประโยชน์อย่างมากด้านเกี่ยวกับเทคนิค และประโยชน์เหล่านี้ ยังนำไปสู่ประโยชน์ด้านธุรกิจด้วย

ประโยชน์ด้านเทคนิค

1. จัดการความซับซ้อนได้ง่ายขึ้น ทำให้คุณภาพของการแก้ปัญหาซอฟต์แวร์ง่ายขึ้น
2. งานซับซ้อนที่ไม่เกี่ยวกับฟังก์ชันธุรกิจ ถูกรวมไว้ในเฟรมเวิร์ก
3. การออกแบบ, อิมพลีเมนต์, ทดสอบ ส่วนต่างๆอย่างอิสระ ทำให้สามารถพัฒนาส่วนต่างๆ ได้ พร้อมกัน
4. การไม่ขึ้นต่อกันของคอมโพเน็นต์ลดผลกระทบของการเปลี่ยนแปลง requirement ไม่ให้กระจาย ไปทั้งระบบ

ประโยชน์ด้านธุรกิจ

1. ผลิตภัณฑ์มีคุณภาพมากขึ้น
2. เร่งเวลาออกสู่ตลาด
3. ใช้ทรัพยากรบุคคลได้คุ้มค่าขึ้น
4. ลดค่าใช้จ่าย
5. เพิ่มปริมาณการนำกลับมาใช้ ในโครงการต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
เทคโนโลยี COM+

3.1 COM+

COM+ รวมหน้าที่สองอย่างไว้ด้วยกันคือ

- สถาปัตยกรรมพื้นฐานการโปรแกรมมิ่ง สำหรับการสร้างซอฟต์แวร์คอมโพเน็นต์ซึ่งสถาปัตยกรรมนี้ถูกนิยามโดยข้อกำหนดของ COM
- บริการของคอมโพเน็นต์ซึ่งเป็นส่วนของรันไทม์เอ็นไวรอนเมนต์ ทำให้สามารถสร้างซอฟต์แวร์ที่มีความสามารถมากขึ้น

คอมโพเน็นต์ที่ทำงานในเอ็นไวรอนเมนต์ข้างต้นเรียกว่า คอนฟิกคอมโพเน็นต์ (Configured Component) ส่วนคอมโพเน็นต์อื่นๆ ที่ไม่ได้ใช้ความสามารถดังกล่าวเรียกว่า อันคอนฟิกคอมโพเน็นต์ (Unconfigured Component) โดยทำงานภายใต้กฎมาตรฐานของ COM

สถาปัตยกรรมพื้นฐานการโปรแกรมมิ่งของ COM+ จัดเตรียมรูปแบบการสร้าง คอมโพเน็นต์ซอฟต์แวร์ไว้ให้ แต่โดยมุมมองของนักพัฒนานั้นยังไม่เพียงพอต่อการพัฒนางานในองค์กร เพราะนักพัฒนาต้องสร้าง คอมโพเน็นต์ทางธุรกิจที่เป็นส่วนหนึ่งของระบบที่มีขนาดใหญ่ ซึ่งมักใช้วิธี การทำงานแบบไคลเอ็นต์/เซิร์ฟเวอร์ หรือ Three-tier ทำให้ต้องใช้ความพยายามมากแม้เป็นแค่การสร้างคอมโพเน็นต์อย่างง่าย เพราะจุดประสงค์ของคอมโพเน็นต์ทางธุรกิจนั้นต้องการต้องการความแข็งแกร่งและปลอดภัยถ้าไคลเอ็นต์หลายตัวเชื่อมต่อกับคอมโพเน็นต์พร้อมๆกันนั้น นักพัฒนาต้องมั่นใจว่าเฉพาะไคลเอ็นต์ที่มีสิทธิเท่านั้นที่สามารถใช้งานคอมโพเน็นต์ได้ ปัญหาอีกประการคือ scalability ถ้าคอมโพเน็นต์ถูกใช้งานพร้อมกันโดยผู้ใช้ที่มีปริมาณมาก

ระบบขนาดใหญ่ต้องรับมือกับความล้มเหลวในระหว่างการทำงานที่ซับซ้อนและเกี่ยวข้องกับเซิร์ฟเวอร์ฐานข้อมูล โดยไคลเอ็นต์อาจต้องการส่งข้อมูลจากเครื่องตนเองไปเก็บที่เซิร์ฟเวอร์ฐานข้อมูลถ้ามีการล้มเหลวระหว่างการทำงานดังกล่าว จะไม่สามารถมั่นใจในความถูกต้องของฐานข้อมูลได้ถ้าไม่ได้ใช้ทรานส์แอ็กชันโพรโตคอล (Transaction protocol) จะเห็นได้ว่าปัญหาหลักของคอมโพเน็นต์ซอฟต์แวร์คือ รูปแบบพื้นฐาน ที่นักพัฒนาต้องรับมือกับการอิมพลีเมนต์การทำงานมากมายเหล่านี้ ซึ่งมีความเกี่ยวข้องน้อยมากกับจุดประสงค์ของตัวแอพพลิเคชันเอง

ไมโครซอฟท์ตระหนักว่าการสร้างระบบ Server-side ที่แข็งแกร่งต้องใช้ความพยายามมาก เช่นงานที่เกี่ยวกับเธรด, การทำงานพร้อมกัน, การรักษาความปลอดภัย, ความแข็งแกร่ง เป็นสิ่งที่สำคัญต่อระบบแบบกระจาย (distributed system) และการพัฒนาซอฟต์แวร์ที่มีลักษณะแบบนี้ต้องใช้ความพยายามอย่างมากมาย

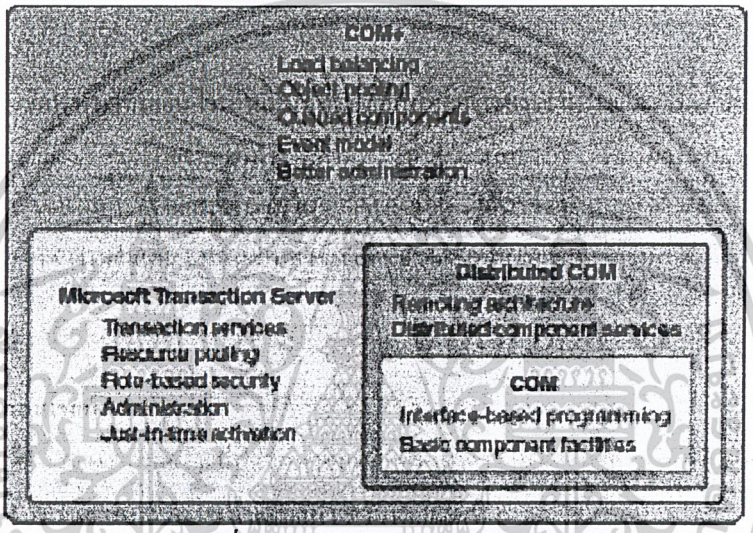
ไมโครซอฟท์จึงได้สร้าง Microsoft Transaction Server (MTS) ซึ่งเป็น รันไทม์ เอ็นไวรอนเมนต์ตัวแรกของวินโดวส์ที่จัดหาบริการเหล่านี้แก่คอมโพเน็นต์ ต่อมารันไทม์เอ็นไวรอนเมนต์ และบริการของคอมโพเน็นต์ของ MTS ได้พัฒนาเป็น COM+

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COM+ จัดหารันไทม์เอ็นไวรอนเมนต์ที่รับมือกับปัญหาของระบบ server-side ทำให้นักพัฒนาสามารถสร้าง in-process คอมโพเนนต์ที่ทำงานตามหน้าที่ที่เกี่ยวกับธุรกิจจริงๆ ได้ง่าย อย่างไรก็ตาม inprocessคอมโพเน็นต์ใดๆ ก็สามารถทำงานใน COM+ รันไทม์เอ็นไวรอนเมนต์ได้ แต่คอมโพเน็นต์ที่ไม่ได้ถูกออกแบบมาให้ทำงานในเอ็นไวรอนเมนต์ เช่น executable คอมโพเน็นต์จะไม่สามารถใช้ประโยชน์จาก COM+

3.2 บริการของคอมโพเน็นต์

บริการของ COM+ คอมโพเน็นต์จัดหาอิมพลีเมนต์มาตรฐานของบริการที่นักพัฒนา คอมโพเน็นต์ใช้บ่อยครั้ง ทำให้สามารถใส่ใจในปัญหาทางธุรกิจมากขึ้น



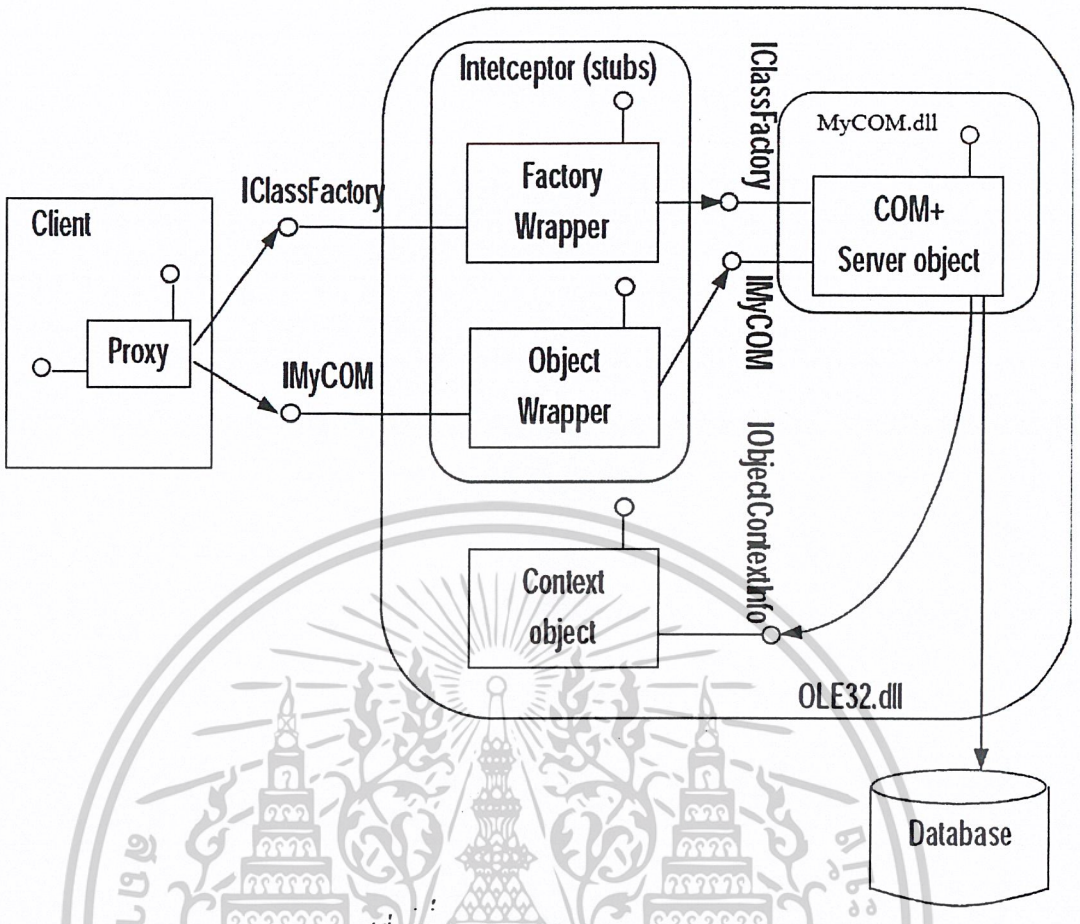
รูปที่ 3-1วิวัฒนาการ COM สู่ COM+

3.3 รูปจำลองของ COM+

รูปจำลองของ COM+ แสดงดังรูปที่ 3-2 ซึ่งประกอบด้วย

1. COM+ surrogate โพรเซส (DLLHOST.EXE)
2. COM+ คลาส โหลดดิง และ รีโมต เฟรมเวิร์ก (OLE32.DLL)
3. คอนเท็กซ์ออบเจ็กต์ และ wrapper proxies สำหรับแต่ละคอมโพเน็นต์
4. COM+ เซิร์ฟเวอร์คอมโพเน็นต์
5. COM+ ไคลเอ็นต์
6. ระบบสนับสนุนต่างๆ ได้แก่ service control manager (SCM), Microsoft Transaction Coordinator (MS-DTC), Microsoft Message Queue (MSMQ), COM-Transaction Integrator (COM-TI)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-2 การทำงานของ COM+

คอมโพเน็นต์ที่ทำงานภายใต้การควบคุมของ COM+ surrogate เรียกว่า COM+ คอมโพเน็นต์ซึ่งคอมโพเน็นต์เหล่านี้ถูกดีพลอย โดยทำงานใน COM+ surrogate (DLLHOST.EXE) ซึ่งถูกจัดการโดย COM+ class loading และรีโมตเฟรมเวิร์ก (OLE32.DLL)

เช่นเดียวกับ COM คอมโพเน็นต์ที่ออบเจ็กต์ต้องอิมพลีเมนต์ IClassFactory ที่ทำหน้าที่เป็นแหล่งผลิตอินสแทนซ์ใหม่ของคอมโพเน็นต์ COM+ แทรก interceptor (ประกอบด้วย Factory Wrapper และ Object rapper) ระหว่างCOM+ คอมโพเน็นต์จริงซึ่งถูกจัดการโดยเฟรมเวิร์ก กับไคลเอ็นต์ของคอมโพเน็นต์นั้น เมื่อไคลเอ็นต์เรียกใช้งาน COM+ คอมโพเน็นต์แล้ว wrapper (Factory และ Object) ก็จะเข้ามาแทรก และใช้วิธีการจัดการอินสแทนซ์ที่เรียกว่า Just In Time Activation (JITA) กับการเรียกใช้งานนั้น จากนั้น

wrapper จะเรียกใช้งาน COM+ คอมโพเน็นต์จริงๆอีกทอดหนึ่ง นอกจากนี้แล้วข้อมูลคุณสมบัติของการดีพลอยคอมโพเน็นต์ได้แก่ รูปแบบทรานส์แอ็กชันและการรักษาความปลอดภัย จะดำเนินการในwrapper ออบเจ็กต์ นี้ด้วย

สำหรับทุก COM+ คอมโพเน็นต์จะมีคอนเท็กซ์ออบเจ็กต์ที่อิมพลีเมนต์อินเทอร์เฟซ IObjectContextInfo ซึ่งคอนเท็กซ์ออบเจ็กต์มีหน้าที่เก็บรักษาข้อมูลของคอมโพเน็นต์ เช่น ข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

transaction, การรักษาความปลอดภัย และข้อมูลของการดีพลอย โดย COM+ คอมโพเน็นต์เรียกใช้งานคอนเท็กซ์ออบเจ็กต์ โดยผ่านอินเทอร์เฟซ IobjectContextInfo

3.4 สิ่งที่ต้องคำนึงถึงในการสร้างมิดเดิ้ลเทียร์แอพพลิเคชั่น

การสร้าง Middle-tier application จะมีสิ่งที่ต้องคำนึงถึงได้แก่

1. Scalability

ปัจจัยที่ทำให้ scalability เป็นหัวข้อสำคัญที่ควรจะพิจารณาเป็นเพราะความนิยมในการใช้งานอินเตอร์เน็ต ซึ่งหนึ่งในจุดประสงค์หลัก COM+ ก็จัดการกับ scalability นั่นเองรูปแบบการใช้งานของอินเตอร์เน็ตจะต่างกับ enterprise application ขนาดใหญ่ตรงที่ เราไม่สามารถทราบจำนวนผู้ใช้งานได้ล่วงหน้า เมื่อเว็บไซต์ได้รับความนิยมมากก็มีโอกาสที่เว็บไซต์จะไม่สามารถรับมือกับโหลดที่มีขนาดมากในช่วงเวลาหนึ่งได้

นอกจากระยะเวลาที่ไซต์ไม่สามารถทำงานจะทำความเสียหายแล้ว ระยะเวลาในการตอบสนองต่อผู้ใช้งานก็เป็นปัญหาที่สำคัญอีกด้วย ทางแก้หนึ่งคือการเพิ่มจำนวนของเมนเฟรม แต่ก็เกิดคำถามว่าควรจะใช้จำนวนเท่าไรเพราะสิ่งเหล่านี้มีต้นทุนสูง และปัญหาก็ไม่ได้แก้ด้วยฮาร์ดแวร์เท่านั้น ทางแก้ที่มีระสิทธิภาพประกอบด้วย software ที่เหมาะสมที่ทำงานบนฮาร์ดแวร์ที่ถูกที่สุด ซึ่งคำตอบที่ชัดเจนมากก็คือ PC นั่นเอง

COM+ ช่วยการ scalability ด้วย just-in-time activation, object pooling, load balancing

2. Reliability

แอพพลิเคชันที่ให้บริการแก่หลายผู้ใช้งานไม่ว่าจะเป็นเว็บหรือเอ็นเตอไพรส์ขนาดใหญ่มักจะเป็นงานที่มีความสำคัญและต้องการความเชื่อถือได้ในการทำงาน ซึ่งความล้มเหลวของระบบการทำงานแบบกระจาย เช่นฮาร์ดแวร์เสีย เป็นสิ่งที่หลีกเลี่ยงไม่ได้ ซึ่งทางแก้ไขคือทำอย่างไรเมื่อมีการเสียของจุดหนึ่งในระบบจะไม่ทำให้ระบบทั้งหมดล้มเหลวและทำให้ข้อมูลที่สำคัญหายไป

อีกมุมมองของความน่าเชื่อถือคือ การลดข้อผิดพลาดของซอฟต์แวร์เพราะค่าเสียหายของ distributed enterprise นั้นสูงกว่า desktop application มาก COM+ สนับสนุนการพัฒนาระบบที่น่าเชื่อถือโดยเตรียมบริการที่จำเป็นต่างๆ, transaction,component load balancing

3. Complexity

ความซับซ้อนของระบบทำให้เสียค่าใช้จ่ายและเวลามาก ซึ่งการใช้งานเทคโนโลยีซอฟท์แวร์ใหม่ก็ทำให้ต้องเรียนรู้ API มากมาย

COM+ ทำให้การเขียนโปรแกรมง่ายขึ้น เช่น COM+ สนับสนุน automatic transaction ช่วยการทำงานแบบ distributed transaction โดยไม่ต้องเพิ่มโค้ดพิเศษเลย, COM+ มี security ที่ง่าย ซึ่งสนับสนุนการทำ synchronization ที่ง่ายขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.5 Declarative, Attribute-Based Programming

หัวใจของการเขียนโปรแกรมของ COM+ คือรูปแบบของการประกาศความต้องการ โดยใช้แอตทริบิวต์ซึ่ง COM+ runtimeจะจัดการกับความต้องการนี้เองข้อดีคือความสะดวกของโปรแกรมเมอร์ที่จะประกาศแอตทริบิวต์ซึ่งง่ายกว่าการเขียนโปรแกรมที่ซับซ้อนซึ่งง่ายต่อการเกิดความผิดพลาด ในทางกลับกันกับโค้ดที่ทำโดยระบบปฏิบัติการซึ่งมีความน่าเชื่อถือมากกว่า

บริการพื้นฐานของ COM+ นั้นยากที่จะอิมพลีเมนต์ให้ถูกต้องด้วยตนเอง และเรายังสามารถปรับแต่งค่าแอตทริบิวต์โดยไม่ต้องแก้ไขโค้ดเลย

1. อพาร์ตเมนต์ ในรูปแบบของเขียนโปรแกรมแบบใช้การประกาศ

ตัวอย่างของการใช้แอตทริบิวต์ คือ การใช้งานฟังชันของ COM apartment ที่ช่วยให้เขียนโค้ดที่ทำงานพร้อมๆ กันโดยที่ไม่ต้องเขียนโค้ดจัดการ synchronization เองเลย

SINGLE-THREAD APARTMENT

สามารถใช้งาน synchronization แบบอัตโนมัติแก่คลาสโดยประกาศให้คลาสมี apartment เธรดโมเดลโดยกำหนดค่า "Apartment" แก่ เธรดโมเดล ในรีจิสตรีของ CLSID หมายความว่าเมื่อสร้างออบเจ็กต์ของคลาส มันจะอยู่ใน STA ของตนเอง ถ้าผู้ใช้ทำงานในเธรดเดียวกับออบเจ็กต์ก็จะสามารถใช้งานอินเทอร์เฟซได้ตรงๆ โดยถ้าเธรดอื่นพยายามที่จะเข้ามาใช้งานในขณะที่เจ้าของ อพาร์ตเมนต์ กำลังใช้งานอยู่ เธรดอื่นๆจะต้องทำงานผ่านพร็อกซี่ตามกฎการทำงานของ COM

พร็อกซี่ดังกล่าวใช้แมสเสจคิวของวินโดวส์ในการทำ synchronization แบบอัตโนมัติ ซึ่งเธรดที่สองจะต้องรอจนกว่าเธรดแรกจะทำงานเสร็จ จุดสำคัญคือพร็อกซี่ทำให้ COM สามารถ "แทรก" การใช้งานเมธอดเพื่อการทำsynchronization โครงสร้างของเธรด synchronization ดังกล่าวกระทำโดยไม่ต้องเขียนโค้ดในแอพพลิเคชั่นชันเลย

2 การประกาศความต้องการทรานส์แอ็กชัน

COM+ ขยายความคิดของการโปรแกรมแบบแอตทริบิวต์ไปอีกมาก ซึ่งทรานส์แอ็กชันก็เป็นตัวอย่างในการใช้งานแอตทริบิวต์ของCOM+ ถ้าเกิดความต้องการในการสร้างคอมโพเน็นต์ที่ทำงานกับบัญชีในธนาคารที่เก็บข้อมูลใน SQL Server ซึ่งการจัดการทรานส์แอ็กชัน ทำโดยรีซอร์สแมเนเจอร์ เราสามารถประกาศ "Transaction support" แก่คอมโพเน็นต์ของเราแบบ "Required" เมื่อคอมโพเน็นต์ทำงาน COM+ runtime จะรับทราบว่าคอมโพเน็นต์ของเราต้องการทรานส์แอ็กชันและจะแทรกโค้ดที่เรียกว่า "interceptor" ที่จะทำรายการทรานส์แอ็กชันให้

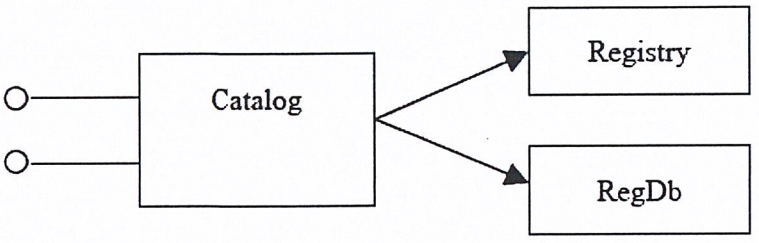
ผลลัพธ์ที่เกิดขึ้นคือการรับประกันว่าการทำงานจะเสร็จสมบูรณ์หรือไม่ก็ไม่เกิดขึ้นเลย โดยกระบวนการทั้งหมดไม่ต้องเขียนโค้ดเลย

3.6 COM+ Catalog

COM มาตรฐานจะเก็บข้อมูลลงในรีจิสเตอร์ แต่ COM+ ต้องการข้อมูลการปรับแต่งที่มากกว่า ข้อมูลเหล่านี้จะถูกแยกเก็บในฐานข้อมูลที่เรียกว่า RegDb ซึ่งฐานข้อมูลนี้ไม่สามารถเข้าถึงได้โดยตรง การ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เข้าถึงข้อมูลเหล่านี้ต้องกระทำผ่าน ระบบออบเจ็กต์ที่เรียกว่า Catalog Manager หรือเรียกสั้นๆ ว่า Catalog ที่จัดบริการการเข้าใช้ RegDb และรีจิสตรีดังรูปด้านล่าง

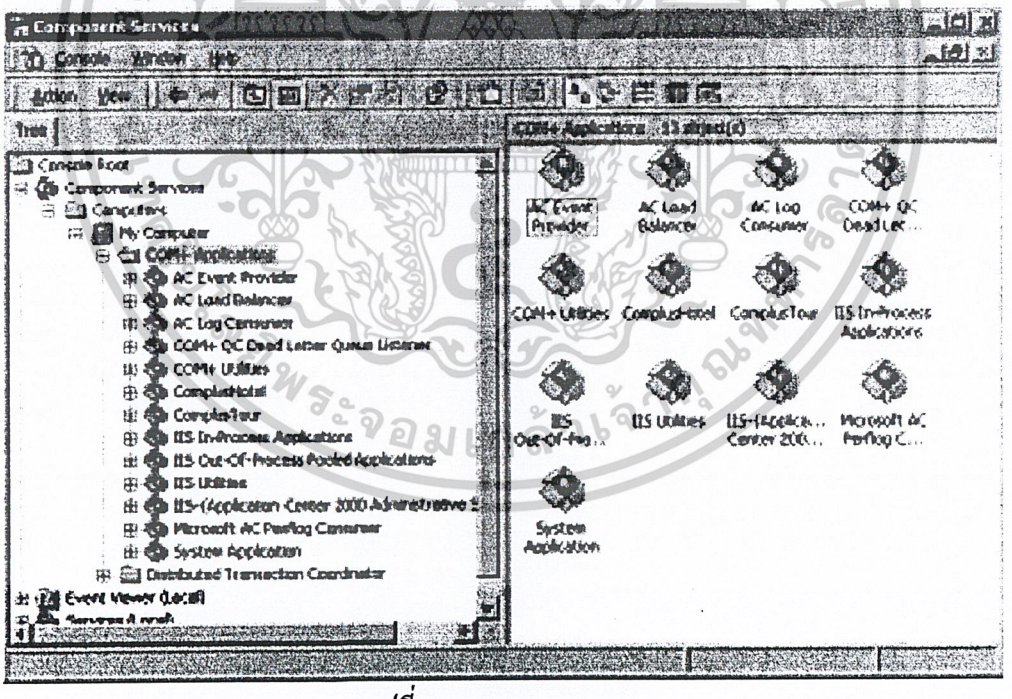


รูปที่ 3-3 COM+ Catalog

Catalog Administration object สามารถเข้าถึงได้โดยโปรแกรมและสคริปต์ ซึ่งเปิดโอกาสให้ควบคุมการทำงานได้อย่างอัตโนมัติ

1. Component service Snap-in

วินโดวส์ 2000 มีเครื่องมือ จัดการแบบ Snap-in ซึ่งทำงานอยู่บน Catalog บางครั้งเรียกว่า COM+ Explorer โดยสามารถใช้งานเครื่องมือดังกล่าวผ่านทางเมนู Start | Programs | Administrative Tools |Component services ดังรูป



รูปที่ 3-4 COM+ Explorer

เราสามารถปรับค่าแอตทริบิวต์ของคอมโพเน็นต์ด้วย COM+ Explorer โดยเลือกคอมโพเน็นต์ที่ต้องการและคลิกขวาเลือกแถบคุณสมบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7 แนวทางของ COM+

บริการต่างๆของ COM+ เกิดขึ้นได้เพราะโครงสร้างพื้นฐานของ COM+ ที่สนับสนุนให้บริการต่างๆ

3.7.1 แนวทางของ COM

COM จัดลำดับชั้นในรูปแบบ server/ class/ interface/ method

Class ของ COM คือ Concrete mplementation ของ abstract data type แบบ interface

Interface มีอย่างน้อย 1 เมธอดซึ่งแสดงถึงความสามารถของอินเทอร์เฟซ และมี signature ที่อธิบายพารามิเตอร์และการคืนค่า ใช้เป็นข้อตกลงในการใช้งานอินเทอร์เฟซ

Object เป็นอินแสตนซ์ของคลาสและอาจจะมีข้อมูลภายในตัวเอง ซึ่งข้อมูลนี้เป็นไพรเวตสามารถเข้าถึงโดยเมธอดของมันเองเท่านั้น

อพาร์ตเมนต์ แบ่งออบเจ็กต์เป็นกลุ่ม โดยเป็นกลุ่มที่ใช้งาน concurrency แบบเดียวกัน ซึ่งอพาร์ตเมนต์ที่ออบเจ็กต์อาศัยอยู่เป็นเป็นส่วนหนึ่งที่ใช้เป็น ID ของออบเจ็กต์

ในขณะที่ class, interface, method เป็นตัวแทนของโครงสร้างทางโลจิคอล ในส่วนบนสุดของลำดับชั้น COM คือ server ซึ่งเป็นส่วน physical จริงๆ ลักษณะเป็นไบนารีโมดูลที่บรรจุอย่างน้อย 1 คลาส และอาจ อยู่ในรูปแบบ EXE หรือ DLL

3.7.2 แนวทางของ COM+

COM+ ยังคงรักษาแนวคิดของ class/ interface/ method และ อพาร์ตเมนต์ แต่แนวคิดของserver ไม่สำคัญอีกต่อไป เพราะคลาสใน COM+ ต้องเป็นแบบ DLL ถ้าต้องการใช้งานเป็น EXE server แล้ว COM+ จะใช้งานคอมโพเน็นต์ใน Surrogate มาตรฐานเรียกว่า DLLHOST.EXE

3.7.2.1 แอพพลิเคชัน

COM+ เริ่มแนวคิดที่เรียกว่า application ซึ่งเป็นกลุ่มของคลาสที่ใช้งานคุณสมบัติต่างๆ ร่วมกันเช่น ความปลอดภัย, การระบุตัวตน ดังนั้นลำดับชั้นของ COM+ คือ application/class/interface

3.7.2.2 คอมโพเน็นต์

ในไมโครซอฟท์ แพลตฟอร์มเอสดีเค (Platform SDK) ได้อธิบายความหมายของคอมโพเน็นต์ว่าเป็นหน่วยของไบนารี่ โค้ดที่สามารถสร้างไบนารี่ออบเจ็กต์ รวมถึงมีความสามารถในการลงทะเบียนตนเองด้วย จุดสำคัญ คือคอมโพเน็นต์มีชื่อเอกลักษณ์ด้วย CLSID, ProgID และลงถูกทะเบียนในรีจิสตรี

คุณสมบัติในการเป็น COM+ คอมโพเน็นต์คือ

1. คอมโพเน็นต์เป็น COM class ที่อิมพลีเม็นต์ IUnknown
2. คอมโพเน็นต์เป็น DLL และมีโครงสร้างสำหรับการสร้าง object, CLSID, class factory และ DLL ต้องมีฟังชัน DllGetClassObject, DllCanUnloadNow
3. คอมโพเน็นต์มีความสามารถในการลงทะเบียนตนเองด้วยการสนับสนุน llRegisterServerและ DllUnegisterServer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. คอมโพเน็นต์ต้องมีไทป์ไลบรารีที่บรรจุใน DLL

5. คอมโพเน็นต์ต้องสามารถมาแชลอินเตอร์เฟซได้ โดยดูอัลอินเตอร์เฟซ (Dual interface) จะมาแชลโดย Automation และต้องจัดหาพร็อกซี่/สตับสำหรับ custom interfaceสำหรับทุกๆ คลาสที่สร้างด้วย ActiveX DLL โดย Visual Basic และ ATL จะมีคุณสมบัติดังกล่าวครบถ้วน

3.7.2.3 คอมโพเน็นต์ที่ถูกปรับแต่งและ ไม่ถูกปรับแต่ง

คอมโพเน็นต์ที่มีคุณสมบัติครบถ้วนตามด้านบนสามารถใช้งาน COM+ service ได้แต่จะไม่เป็นไปในทันทีจนกว่าจะมีการติดตั้งบน COM+ applicationคอมโพเน็นต์สามารถเพิ่มโดยวิธีการใช้เครื่องมือและการโปรแกรม เมื่อคอมโพเน็นต์ถูกติดตั้งจะเรียกว่าคอมโพเน็นต์ที่ถูกปรับแต่ง (configured component) ส่วนคอมโพเน็นต์ที่ไม่ถูกติดตั้งจะเรียกว่าคอมโพเน็นต์ที่ไม่ถูกปรับแต่ง (unconfigured component)

3.7.3 ชนิดของ COM+ Application

พื้นฐานของ COM+ application คือ server application ซึ่งอยู่ในโพรเซสของตนเอง คือ DLLHOST.EXE เมื่อเซิร์เวอร์ทำงานผิดพลาดจะไม่ทำให้ไคลเอ็นต์โพรเซสเสียหาย เพราะอยู่กันคนละโพรเซส และไคลเอ็นต์สามารถใช้งานผ่านทาง DCOM โดย server application มีความสามารถในการแทรกบริการอย่างเต็มรูปแบบ

library application ทำงานในโพรเซสของไคลเอ็นต์ ซึ่งไม่มีการป้องกันการทำงานผิดพลาด และจำกัดความสามารถในการแทรกบริการเท่านั้น จุดประสงค์หลักของ library application คือการนำโค้ดกลับมาใช้สำหรับ COM+ application มากกว่าที่จะบริการไคลเอ็นต์ตรงๆ

proxy application ประกอบด้วยข้อมูลที่จะติดตั้งบนไคลเอ็นต์ เพื่อให้มีความสามารถในการเรียกใช้งานจากระยะไกล

preinstalled COM+ application เป็น application ของ COM+ และ IIS ซึ่งไม่สามารถแก้ไขเปลี่ยนแปลงได้

3.8 สถาปัตยกรรมของ COM+

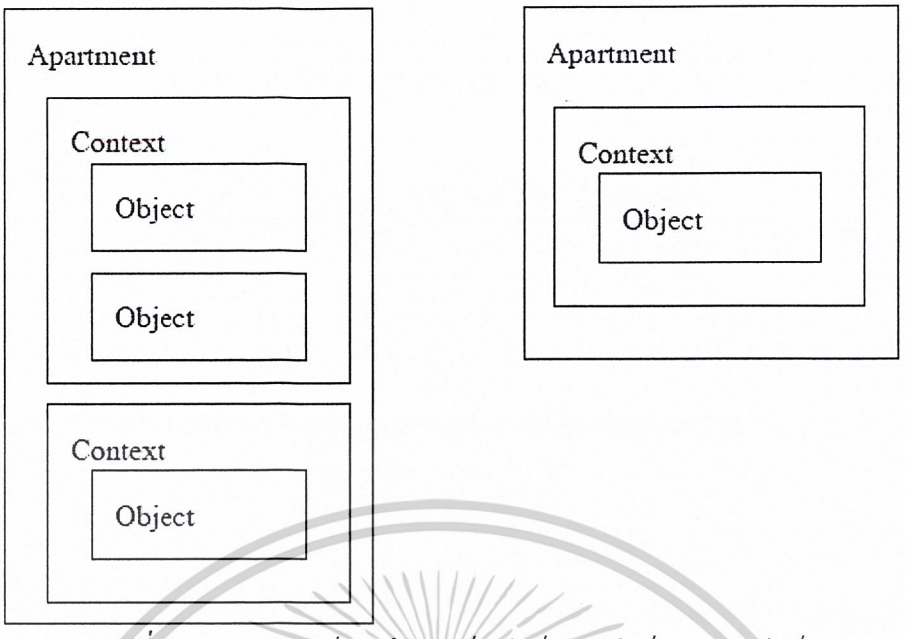
COM+ ตั้งอยู่บน 3 แนวคิดคือ context, activation และ interception

3.8.1 Context

คอนเท็กซ์เป็นกลุ่มของออบเจ็กต์ในอพาร์เมนต์ที่มีความต้องการตอนรันไทม์เหมือนกัน ทุกๆ COM object ที่ถูกแอกทิเวตจะต้องสัมพันธ์กับ 1 คอนเท็กซ์เท่านั้น และคอนเท็กซ์จะอาศัยใน 1 อพาร์ตเมนต์เท่านั้น 1 อพาร์ตเมนต์ สามารถมีได้หลายคอนเท็กซ์ และคอนเท็กซ์สามารถมีได้หลายออบเจ็กต์ ดังนั้น ออบเจ็กต์ที่อยู่คนละอพาร์ตเมนต์ก็ต้องอยู่คนละคอนเท็กซ์ด้วย ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง



รูปที่ 3-5 ความสัมพันธ์ระหว่างอพาร์ตเม็นต์ คอนเท็กซ์ และออบเจ็กซ์

ทุกอพาร์ตเมนต์จะที่ default context เมื่อคอมโพเน็นต์ที่ไม่ถูกปรับแต่งถูกแอกทิเวต จะถูกนำเข้าไปอยู่ใน default context โดยทั่วไปแล้ว คอนเท็กซ์นี้จะไม่ได้ใช้งาน

สำหรับคอมโพเน็นต์ที่ถูกปรับแต่ง คุณสมบัติของคอนเท็กซ์ถูกกำหนดโดยแอตทริบิวต์ของคอมโพเน็นต์ที่เก็บไว้ใน catalog ซึ่งคอนเท็กซ์นี้เป็นกุญแจสำคัญของการใช้งาน COM+ service

1. Context Object

คอนเท็กซ์เป็นสิ่งที่ใช้อ้างอิงถึงกลุ่มของ COM object ซึ่งการเชื่อมต่อกับคอนเท็กซ์นั้นจริงๆ แล้วก็เป็น COM object เช่นกัน เรียกว่า "Context Object" ซึ่งจะเก็บข้อมูลของคอนเท็กซ์นั้นๆ

โค้ดในเมธอดสามารถเข้าถึงคอนเท็กซ์ออบเจ็กต์โดยใช้ API CoGetObjectContext (หรือ GetObjectContext ใน visual basic) ซึ่งคืนค่าเป็นพอยเตอร์ไปที่ interface IObjectContext จากนั้นสามารถใช้เมธอดของอินเทอร์เฟซดังกล่าวเพื่อแก้ไขคุณสมบัติของคอนเท็กซ์ออบเจ็กต์

2. Call Context

บางส่วนของคอนเท็กซ์ใช้ควบคุมเรื่องความปลอดภัย ซึ่งจะใช้ API CoGetCallContext ซึ่งจะคืนค่าเป็น ISecurityCallContext ซึ่งมีข้อมูลเกี่ยวกับความปลอดภัยในการใช้งานเมธอด

3.8.2 Activation

จุดสำคัญของการใช้งานคอนเท็กซ์นั่นคือ คอนเท็กซ์ใช้อธิบายสภาพแวดล้อมของออบเจ็กต์ไม่ใช่โครงสร้างข้อมูลภายในออบเจ็กต์

ออบเจ็กต์โดยตัวเองแล้วไม่มีคอนเท็กซ์ ซึ่งกระบวนการในการเชื่อมออบเจ็กต์กับคอนเท็กซ์เรียกว่า "Activation"

สภาพแวดล้อมของออบเจ็กต์คือคอนเท็กซ์และคอนเท็กซ์นี้มีความสำคัญมากในการพิจารณาพฤติกรรมของออบเจ็กต์ขณะรันไทม์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
55083

การนำออบเจ็กต์ให้มาอยู่ในสถานะพร้อมต้องใช้ 2 ขั้นตอนใน COM+

1. การสร้าง เมื่อออบเจ็กต์ถูกสร้างด้วย class factory เป็น COM ธรรมดา

2. activation เมื่อออบเจ็กต์ถูกเชื่อมเข้ากับคอนเท็กซ์แล้วก็พร้อมใช้งานได้

การทำลายออบเจ็กต์ก็มี 2 ขั้นตอนเช่นกัน

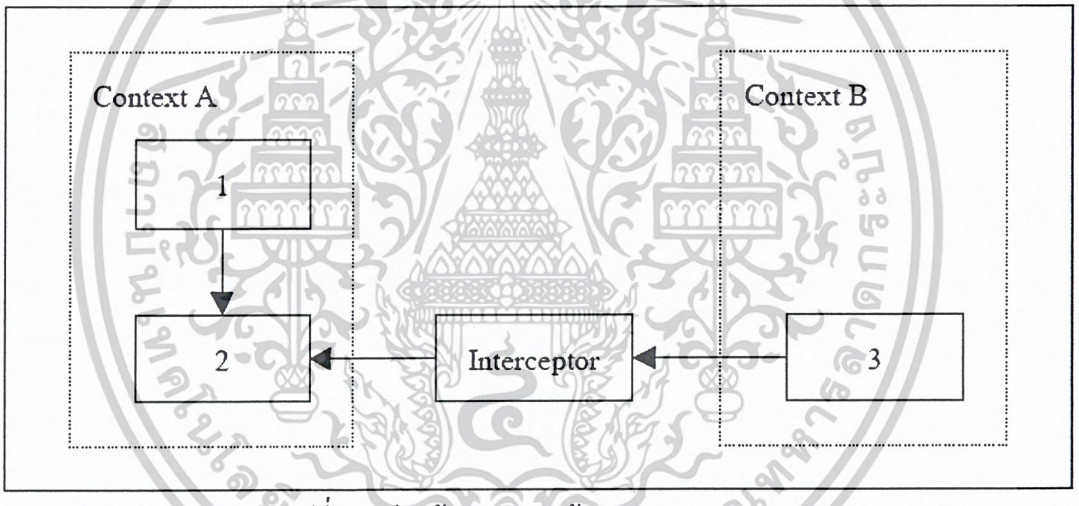
1. deactivate เป็นการกำจัดคอนเท็กซ์ออกจากออบเจ็กต์

2. ทำลายออบเจ็กต์

โดยปกติแล้ว activation จะเกิดขึ้นหลังจากการสร้าง และ deactivate จะเกิดก่อนการทำลาย แต่ COM+ มีความสามารถที่เรียกว่า object pooling ซึ่งสามารถเก็บออบเจ็กต์ที่ถูกสร้างไว้แล้วได้ เมื่อไคลเอ็นต์ต้องการ activate object ก็จะดึงออบเจ็กต์ออกมาจาก pool และเมื่อใช้งานออบเจ็กต์เสร็จก็จะ deactivate และใส่กลับเข้าไปใน pool

3.8.3 Interception

เมื่อออบเจ็กต์ถูก activate เข้ากับคอนเท็กซ์ การเรียกใช้งานออบเจ็กต์หนึ่งไปยังอีกออบเจ็กต์ที่อยู่ในคอนเท็กซ์เดียวกันจะไม่ต้องการการแทรกแซงจาก



รูปที่ 3-6 เรียกข้าม Context ต้องการ interceptor

1. Interceptor

Interceptor คือพร็อกซีขนาดเล็กที่จัดการโดย COM+ รันไทม์เพื่อแก้ปัญหาการไม่เข้ากันของ context Interceptor จะมีอินเทอร์เฟซที่เหมือนกับออบเจ็กต์เป้าหมาย ซึ่งในมุมมองของไคลเอ็นต์ จะไม่ทราบว่าทำงานกับ Interceptor แทนที่จะเป็นออบเจ็กต์จริงๆ

Interceptor สามารถทำงานก่อนการเรียกออบเจ็กต์ และสามารถทำงานหลังการทำงานของออบเจ็กต์ก่อนที่จะคืนค่ากลับไปยังไคลเอ็นต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Interceptor ถือว่าเป็นขนาดเล็กเพราะไม่จำเป็นต้องใช้ thread switch ซึ่งต่างจากพร็อกซี่ทั่วไปที่ต้องการ thread switch หรือแม้กระทั่ง process switch

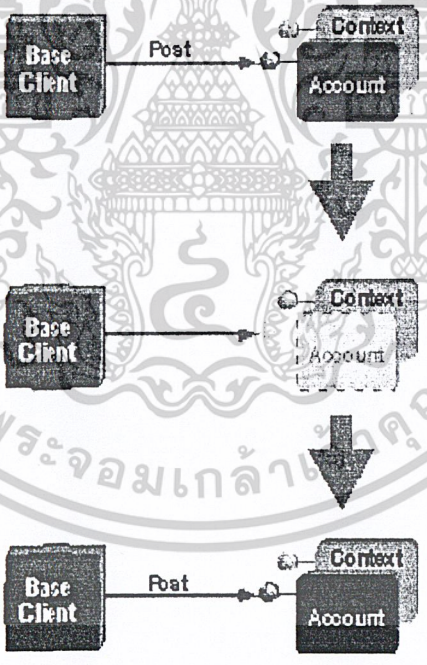
2. interface pointer marshalling

ถ้าเราผ่าน interface pointer เป็นพารามิเตอร์ของการเรียกเมธอดที่ข้ามคอนเท็กซ์ แล้ว COM+ จะแปลงพอยน์เตอร์เป็นพร็อกซี่อย่างอัตโนมัติ และเมื่อมีการ call back มาจากข้างนอกก็จะต้องผ่าน interceptor ถ้าต้องการแบ่งกันใช้งาน interface pointer ในลักษณะของตัวแปร global เราจะต้องรับผิดชอบในการ marshalling interface ด้วยตนเอง

3.8.4 Just-in-Time (JIT) Activation

ลักษณะที่สำคัญของ COM+ คือความสามารถในการขยาย middle-tier คอมโพเน็นต์เพื่อการรองรับการใช้งานไคลเอ็นต์พร้อมกันหลายพันคน โดยไคลเอ็นต์ที่พยายามจะสร้าง COM+ ออบเจ็กต์ จะได้รับreference ของ wrapper ออบเจ็กต์ ซึ่งไม่ใช้ reference ของออบเจ็กต์ของคอมโพเน็นต์จริงๆ

COM+ ออบเจ็กต์จะเริ่มต้นทำงานที่สถานะ deactivated เมื่อไคลเอ็นต์เรียกเมธอดของออบเจ็กต์ที่มีสถานะ deactivated แล้ว COM+ จะ activate ออบเจ็กต์ให้อย่างอัตโนมัติ วิธีการนี้เรียกว่า just-in-time activation ทำให้ไคลเอ็นต์ได้ รับ reference ไปยัง ออบเจ็กต์ ได้โดยที่อาจจะยังไม่ใช้งานในทันทีทันใด



รูปที่ 3-7 Just-in-time Activation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COM+ สามารถ deactivate ออบเจ็กต์ ได้แม้ว่าไคลเอ็นต์ยังมี reference ไปยังคอมโพเน็นต์นั้นอยู่ ซึ่งการ deactivate สามารถทำได้โดย release ทุก reference ของออบเจ็กต์ ทำให้ COM+ คอมโพเน็นต์ถูกทำลายไปเมื่อ reference count ลดลงถึงศูนย์

เมื่อไคลเอ็นต์ เรียกใช้งานออบเจ็กต์ที่อยู่ในสถานะ deactrivated แล้ว COM+ จะ reactivate ให้อย่างอัตโนมัติ ดังนั้นขณะที่ไคลเอ็นต์รู้สึกว่าทำงานกับออบเจ็กต์เดียวในความเป็นจริงอาจจะทำงานกับหลายอินสแทนซ์ของ คลาส เดียวกันก็ได้

just-in-time activation ช่วยให้ COM+ คอมโพเน็นต์สามารถแบ่งปันทรัพยากรของเซิร์ฟเวอร์ใช้ได้อย่างเหมาะสม ยกตัวอย่าง ไคลเอ็นต์ ที่ใช้งานเพียง 10% ของช่วงชีวิตของออบเจ็กต์ เวลาที่เหลืออีก 90% นั้นด้วย just-in-time activation จะทำให้เซิร์ฟเวอร์สามารถใช้ทรัพยากรรองรับงานได้มากขึ้น

3.9 COM+ Threading Model

อพาร์ตเมนต์ จะรวมกลุ่มออบเจ็กต์ที่มีความต้องการ Concurrency เหมือนกัน ในวินโดวส์เอ็นที 4.0 มีอพาร์ตเมนต์ อยู่ 2 แบบคือ STA และ MTA

STA(Single-Threaded Apartments)

STAจะจัดลำดับการเข้าใช้งานออบเจ็กต์โดยใช้วินโดวส์แมสเสจคิวซึ่งเป็นบริการของวินโดวส์ที่สร้างและซ่อนไว้โดย STA ทำให้ไม่มีการเข้าใช้งานเมธอดของออบเจ็กต์ที่อยู่ใน STA พร้อมกัน ทำให้ไม่ต้องมีการปกป้องข้อมูลของอินแสตนซ์ในกรณีที่มีการเข้าถึงพร้อมกัน ซึ่งช่วยให้การเขียนโปรแกรมง่ายขึ้น

MTA(Multithreaded Apartment)

MTA ไม่มีกลไกการจัดลำดับการเข้าถึง ดังนั้นหลาย ๆ เธรดในMTAสามารถเข้าถึงออบเจ็กต์เดียวกันได้พร้อมๆ กัน เธรดที่ทำงานใน MTA บางครั้งเรียกว่า free thread ดังนั้นเมธอดของคลาสที่เรียกโดย free thread จะต้องเป็น thread safe ซึ่งต้องการทำซิงโครไนเซชันเพื่อปกป้องข้อมูลในส่วนคริติคอลเซ็กชัน แต่ประสิทธิภาพที่ได้จะมากขึ้น

ในแต่ละ โพรเซสสามารถมีได้หลาย STA แต่มีได้เพียง MTA เดียวเท่านั้น DLL server ไม่สามารถเรียก CoInitializeExโดยตัวเองจึงต้องประกาศเธรดดิงโมเดลในรีจิสตรีซึ่งค่าที่เป็นไปได้ของโมเดลเธรด มีทั้งหมด 5 ค่าได้แก่

1) Single มีชนิดของ อพาร์ตเมนท์ เป็น STA ออบเจ็กต์ประเภทนี้จะไม่เป็น Thread-Safe ไม่ใช่แค่ตัวออบเจ็กต์เท่านั้นแต่ยังรวมถึง Class Factory สำหรับคลาส และฟังก์ชัน DllGetClassObject และ DllCanUnloadNow ก็เช่นกัน Single-Thread สามารถทำการเรียกใช้ทุก ๆ เมธอดที่อยู่ในเซิร์ฟเวอร์

2) อพาร์ตเมนต์ มีชนิดของอพาร์ตเมนท์เป็น STA ออบเจ็กต์ประเภทนี้จะสามารถป้องกันปัญหาของระบบ Multi-Thread ได้เฉพาะในระบบดับคลาสเท่านั้น เนื่องจาก Class Factory และ ฟังก์ชัน DllGetClassObject และ DllCanUnloadNow ในคลาสของมันเป็น Thread-Safe แต่ในระดับ อินแสตนซ์จะถูกจำกัด เนื่องจากการเข้าถึงแต่ละอินแสตนซ์จะถูกกำหนดให้เป็น Single-Thread เท่านั้น

3) Free มีชนิดของ อพาร์ตเมนท์ เป็น MTA ออบเจ็กต์ประเภทนี้สามารถป้องกันปัญหาของระบบ มัลติเธรด ทั้งสถานะต่ออินแสตนซ์ และสถานะต่อคลาสโดยใช้ Thread Synchronization

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Primitives เช่น Critical Section หรือ Mutex ออบเจ็กต์เหล่านี้สามารถทำงานในสถานะที่เป็น มัลติเธรดได้อย่างปลอดภัย

4) Both สามารถมีชนิดของอพาร์ตเมนท์ ได้ทั้ง STA และ MTA ออบเจ็กต์ประเภทนี้จะสร้างตัวเองอยู่ในอพาร์ตเมนท์เดียวกับไคลเอนต์ของมันซึ่งทำให้มีประสิทธิภาพสูงสุด และทำงานในเธรดเดียวกับผู้ที่เรียกมัน และเนื่องจากมันสามารถทำงานได้ในระบบที่เป็นมัลติเธรดมันจึงป้องกันปัญหาที่จะเกิดในระดับคลาส และ ระดับอินแสตนซ์

5) Neutral มีชนิดของอพาร์ตเมนท์เป็น TNA (Thread Neutral Apartment) ออบเจ็กต์ประเภทนี้จะทำงานในเธรดของผู้เรียก แต่อยู่ในอพาร์ตเมนท์ของมันเองแทนที่จะเป็นอพาร์ตเมนท์ของผู้เรียก เนื่องจาก เธรดที่อยู่ในอพาร์ตเมนท์ประเภทอื่นในโพรเซสเดียวกันสามารถเข้าถึง TNA ได้ทุกเวลา และออกจาก อพาร์ตเมนท์เมื่อมันทำงานเมื่อการทำงานของเมธอดนั้นเสร็จ เนื่องจากอพาร์ตเมนท์ประเภทนี้สามารถทำการ Thread-Synchronization primitives ได้ภายในตัวของมันเอง หรือสามารถ Thread-Synchronize ได้ด้วยบริการของ COM+

3.10 ทรานส์แอ็กชันใน COM+

COM+ สนับสนุนการทำทรานส์แอ็กชันแบบกระจายสำหรับ COM คอมโพเน็นต์ โดยเราไม่ต้องเขียนโค้ดที่ซับซ้อน กระบวนการต่างๆ เป็นดังนี้

1. ปรับแต่งออบเจ็กต์ที่ทำงานภายใต้ COM+
2. ตั้งค่าแอตทริบิวต์ของออบเจ็กต์ว่าจะให้เป็นแบบ Requires a Transaction, Requires a New Transaction หรือ Supports Transactions
3. ใช้เมธอด SetComplete ของออบเจ็กต์คอนเท็กซ์ (Object Context) เมื่อพร้อมที่จะคอมมิททรานส์แอ็กชัน
4. ใช้เมธอด SetAbort ของออบเจ็กต์คอนเท็กซ์ (Object Context) เมื่อพร้อมที่จะโรลแบ็กทรานส์แอ็กชัน

1. สรุปความหมายของค่าต่างๆ ของทรานส์แอ็กชันแอตทริบิวต์ทั้งหมดของ COM+ ออบเจ็กต์ดังนี้

Requires a Transaction

ออบเจ็กต์นี้จะต้องทำงานภายใต้ทรานส์แอ็กชัน COM+ จะเริ่มต้นทรานส์แอ็กชันใหม่สำหรับออบเจ็กต์นี้ก็ต่อเมื่อ ไม่มีออบเจ็กต์อื่น ๆ ที่อยู่ในคอนเท็กซ์เดียวกัน เริ่มต้นทรานส์แอ็กชันเลย

Requires a New Transaction

ออบเจ็กต์นี้จะต้องทำงานภายใต้ทรานส์แอ็กชัน COM+ จะเริ่มต้นทรานส์แอ็กชันใหม่ให้เมื่อออบเจ็กต์นี้ถูกสร้างขึ้น

Supports Transaction

ออบเจ็กต์นี้จะทำงานภายใต้ทรานส์แอ็กชันก็ต่อเมื่อ ผู้สร้าง (Creator) ได้สร้างออบเจ็กต์ที่เริ่มต้นทรานส์แอ็กชันไว้ก่อนแล้ว ไม่เช่นนั้นออบเจ็กต์จะทำงานโดยปราศจากทรานส์แอ็กชัน

Does Not Support Transactions

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ออปเจ็กต์นี้ไม่ต้องทำงานแบบทรานส์แอ็กชัน และ COM+ จะไม่รับออบเจ็กต์นี้ภายใต้ทรานส์แอ็กชันใหม่ (เป็นค่าดีฟอลต์)

Disabled

ออบเจ็กต์ไม่สนใจที่จะเลือกค่าใดๆ ในทรานส์แอ็กชันแอตทริบิวต์ ตัวเลือกนี้ทำให้คอมโพเน็นต์มีลักษณะเป็นคอมโพเน็นต์ที่ยังไม่ได้ตั้งค่าอะไรเลย

2. การเริ่มต้นทรานส์แอ็กชัน

หลังจากที่ตั้งค่าทรานส์แอ็กชันให้กับคอมโพเน็นต์ COM+ รันไทม์จะอ่านค่าทรานส์แอ็กชันแอตทริบิวต์เมื่อเริ่มสร้างออบเจ็กต์ หลังจากนั้นจะติดต่อกับทรานส์แอ็กชันโคออร์ดิเนเตอร์ที่เรียกว่า Distribute Transaction Coordinator (DTC) โดยจะบอกให้ DTC เริ่มต้นทรานส์แอ็กชันสำหรับออบเจ็กต์นี้ DTC จะเริ่มต้นทรานส์แอ็กชันให้กับออบเจ็กต์แล้วส่งค่า GUID ของทรานส์แอ็กชันกลับมา (DTC จะกำหนด GUID ให้กับทุกทรานส์แอ็กชัน) GUID นี้จะเก็บอยู่ในคอนเท็กซ์ของออบเจ็กต์นั้น เราสามารถใช้เมธอด IsInTransaction และ GetTransactionId ของอินเทอร์เฟซ IObjectContextInfo เพื่อดูว่าออบเจ็กต์นี้ทำงานอยู่ภายใต้ทรานส์แอ็กชันหรือไม่ และมี GUID ทรานส์แอ็กชันอะไร

ออบเจ็กต์ที่เป็นตัวเริ่มต้นทรานส์แอ็กชันจะเรียกว่า ออบเจ็กต์ราก (Root Object) ของทรานส์แอ็กชัน ออบเจ็กต์รากมีความสำคัญมากเนื่องจากว่า COM+ รันไทม์จะไม่คอมมิททรานส์แอ็กชัน จนกว่าออบเจ็กต์รากจะถูกดีแอกติเวท (Deactived) ออบเจ็กต์อื่น ๆ ที่อยู่ภายใต้ทรานส์แอ็กชันเดียวกันจะเป็นเพียงตัวลงความเห็น ว่าควรจะคอมมิททรานส์แอ็กชันหรือไม่ แต่ออบเจ็กต์รากเป็นตัวกำหนดว่าเมื่อไหร่ที่จะคอมมิททรานส์แอ็กชัน

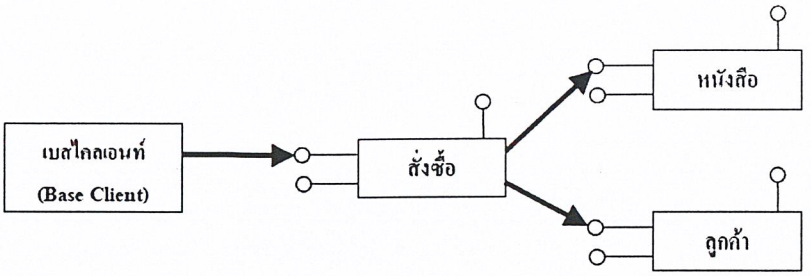
เมื่อ COM+ ออบเจ็กต์ที่อยู่ภายใต้ทรานส์แอ็กชันทำการติดต่อกับรีซอร์สเมเนเจอร์ รีซอร์สเมเนเจอร์นั้นก็จะอยู่ในทรานส์แอ็กชันด้วย รีซอร์สเมเนเจอร์จะติดต่อกับ DTC เพื่อบอกว่าขณะนี้มันเป็นส่วนหนึ่งของทรานส์แอ็กชัน ถ้าออบเจ็กต์ติดต่อกับรีซอร์สเมเนเจอร์หลายตัว รีซอร์สเมเนเจอร์เหล่านั้นทั้งหมดก็จะอยู่ใน ทรานส์แอ็กชันเช่นกัน

เมธอด SetComplete เป็นการโหวตเพื่อคอมมิททรานส์แอ็กชัน ออบเจ็กต์รากเป็นตัวบ่งบอกว่าขณะนี้พร้อมคอมมิทแล้ว DTC จะทำงานโดยใช้โปรโตคอลทู-เฟสคอมมิท เพื่อคอมมิทหรือโรลแบ็กทรานแอ็กชั่นตามขั้นตอนที่ได้กล่าวมาแล้วขั้นต้น

3. ทรานส์แอ็กชันที่เกี่ยวข้องกับออบเจ็กต์มากกว่า 1 ตัว

ก่อนหน้านี้เราได้กล่าวถึง COM+ ออบเจ็กต์ตัวเดียวที่ทำงานในลักษณะที่เป็นทรานส์แอ็กชันและมีการติดต่อกับรีซอร์ทเมเนเจอร์ตั้งแต่สองตัวขึ้นไป แต่ในหลาย ๆ กรณีทรานส์แอ็กชันมักจะเกี่ยวข้องหรือใช้ออบเจ็กต์หลาย ๆ ตัว ตัวอย่างเช่นการสั่งหนังสือออนไลน์ สมมติว่าให้ออบเจ็กต์การสั่งซื้อเป็นออบเจ็กต์รากของทรานส์แอ็กชัน ออบเจ็กต์การสั่งซื้อมีการสร้างและเรียกใช้ออบเจ็กต์ลูกค้ากับออบเจ็กต์หนังสือดังรูปที่ 6-8

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-8 ออบเจ็กซ์หลายๆตัวทำงานอยู่ในทรานส์แอ็กชั่นเดียวกัน

เมื่อ COM+ ออบเจ็กต์ถูกสร้างโดย COM+ ออบเจ็กต์ด้วยกัน ออบเจ็กที่ถูกสร้างขึ้นใหม่จะมีลักษณะการทำงานแบ่งได้เป็น 3 ทาง คือ (1) อยู่ในทรานส์แอ็กชันเดียวกับออบเจ็กที่สร้างมัน (2) สร้างทรานส์แอ็กชันของตนเองขึ้นมาใหม่ (3) ไม่ทำงานเป็นทรานส์แอ็กชันเลย กรณีทั้งสามจะเกิดขึ้นอยู่กับการตั้งค่าทรานส์แอ็กชันแอตทริบิวต์ให้กับออบเจ็กต์ต่าง ๆ

ถ้าออบเจ็กต์สั่งซื้อและออบเจ็กต์ลูกค้าต่างมีทรานส์แอ็กชันเป็นของตัวเอง ออบเจ็กต์สั่งซื้ออาจคอม มิทงานของตน โดยที่ออบเจ็กต์ลูกค้าอาจทำงานแล้วเกิดปัญหาทำให้ต้องโรลแบ็กงานของตน จะเห็นได้ว่าออบเจ็กต์ทั้งสองอาจทำงานขัดแย้งกันได้ซึ่งเป็นเหตุการณ์ที่เราไม่ต้องการ หากออบเจ็กต์ทั้งคู่ทำงานอยู่ในทรานส์แอ็กชันเดียวกัน การคอมมิทหรือโรลแบ็กทรานส์แอ็กชันจะขึ้นอยู่กับออบเจ็กต์ทั้งหมดที่อยู่ในทรานส์แอ็กชันนั้น ออบเจ็กต์แต่ละตัวจะเสนอการคอมมิทโดยเรียกใช้เมธอด SetComplete ของอินเทอร์เฟส IObjectContext หรืออาจเสนอให้โรลแบ็กโดยใช้เมธอด SetAbort ของอินเทอร์เฟส IObjectContext เช่นเดียวกัน ถ้าออบเจ็กต์ทุกตัวเสนอให้คอมมิท ทรานส์แอ็กชันจึงจะคอมมิท หากมีตัวใดตัวหนึ่งเสนอให้ โรลแบ็ก ทรานส์แอ็กชันจะต้องโรลแบ็ก สิ่งสำคัญประการหนึ่งคือ การเรียกใช้เมธอด SetComplete เป็นเพียงการเสนอว่าออบเจ็กต์นั้นยอมรับและพร้อมคอมมิท ทรานส์แอ็กชัน ออบเจ็กต์จะยังไม่ทราบว่าทรานส์แอ็กชันคอมมิทแล้วจริง ๆ หรือไม่ ดังนั้น COM+ ออบเจ็กต์ที่ทำงานในลักษณะที่เป็นทรานส์แอ็กชันไม่ควรสรุปว่าสถานะของตนหลังจากใช้เมธอด SetComplete ไปแล้วจะเป็นสถานะสุดท้ายที่ถูกต้องเสมอไปเมื่อทรานส์แอ็กชันคอมมิท ตัวอย่างเช่น ถ้าออบเจ็กต์ลูกค้าทำงานโดยหักเงินของบัญชีลูกค้าในฐานข้อมูล ออบเจ็กต์ลูกค้ายังไม่ควรเปลี่ยนแปลงสถานะยอดเงินจนกว่าทรานส์แอ็กชันจะคอมมิท เพราะหากเปลี่ยนแปลงแล้วออบเจ็กต์อื่น ๆ ที่ทำงานในทรานส์แอ็กชันเดียวเสนอให้โรลแบ็กทรานส์แอ็กชัน จะทำให้ออบเจ็กต์ลูกค้าอยู่ในสถานะที่ไม่ถูกต้องโดยเงินในบัญชีลูกค้าถูกหักออกไปโดยที่การสั่งหนังสือทำงานไม่สำเร็จ ในการป้องกันปัญหาดังกล่าวCOM+ จะดีแอกติเวทออบเจ็กต์ทั้งหมดที่เกี่ยวข้องกับทรานส์แอ็กชันเมื่อ ทรานส์แอ็กชันคอมมิท การดีแอกติเวทออบเจ็กต์เป็นการเพิ่ม scalability ของระบบแต่เหตุผลของในเรื่องของ ทรานส์แอ็กชันคือเพื่อความถูกต้องของงานที่ปฏิบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.11 COM+ และ Scalability

3.11.1 Microsoft Clustering Technology

ทางแก้ปัญหาโดยทั่วไปเมื่อมีการเพิ่มโหลดการทำงานคือการเพิ่มฮาร์ดแวร์โดยการซื้อระบบที่ใหญ่ขึ้นมาแทนที่ระบบเดิม แต่วิธีนี้ใช้ค่าใช้จ่ายมากและยังไปติดอยู่ที่ขนาดที่ใหญ่ที่สุดของระบบเดี่ยวด้วยทางแก้อีกทาง คือมัลติโพรเซสเซอร์ ซึ่งมีอยู่ 2 แบบคือ

1. Tightly couple คือ Symmetric, shared-memory multiprocessor (SMP) ซึ่งระบบจะใช้งานระบบปฏิบัติการเพียงสำเนาเดียว และมีการจัดการการทำงานของเธรดต่างๆอย่างอัตโนมัติแต่ SMP มีข้อจำกัดเนื่องจากการแบ่งกันใช้ช่องทางเดินของหน่วยความจำ (shared-memorybus)

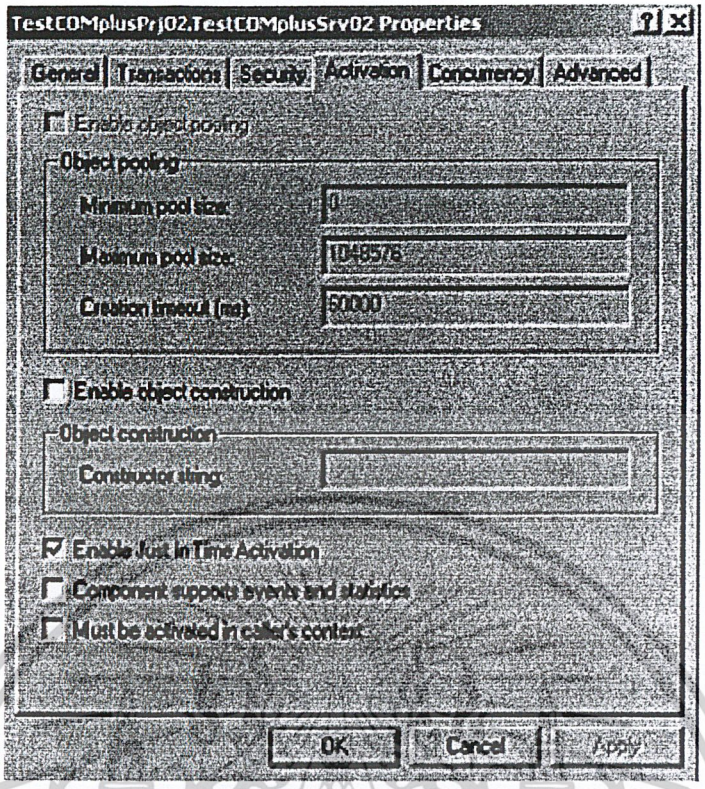
2. Loosely couple คือ clusters ซึ่งมีความสเกลมากกว่าและยังทนทานต่อการเสียหายอีกด้วย เพราะclustersประกอบด้วยหลายโหนดที่อิสระต่อกัน จึงไม่มีการแบ่งกันใช้ช่องทางเดินของหน่วยความจำและเมื่อโหลดใดเสียหายก็ไม่ทำให้ทั้งระบบเสียหายไปด้วย สำหรับ clusterแบบนี้เป็นแบบที่ใช้ในโครงงานนี้ ามารถ อ่านรายละเอียดที่บท 8

3.11.2 Object Pooling

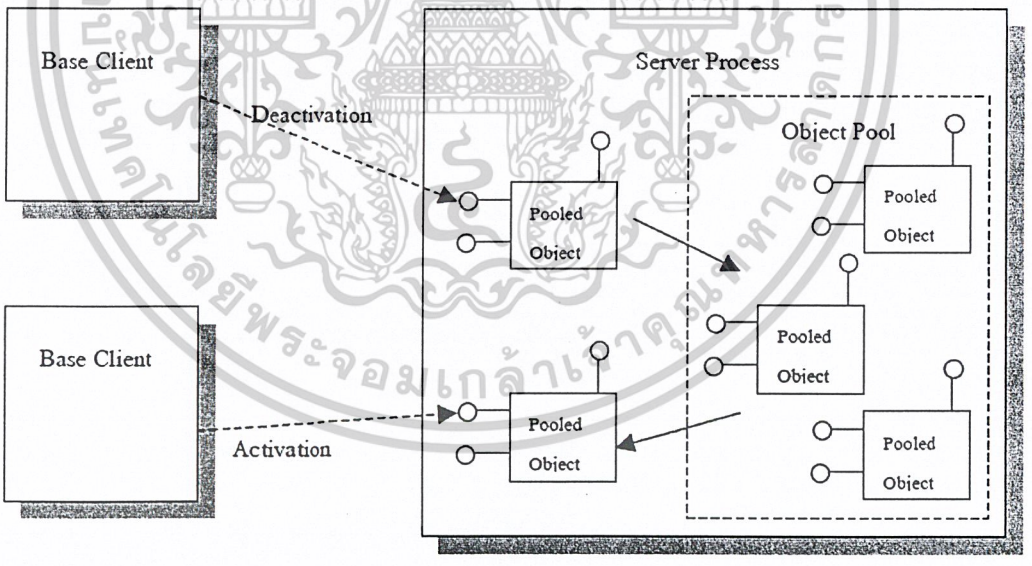
สำหรับแอพพลิเคชั่นที่ใช้ JIT Activation ในการ Activate และ Deactivate ออบเจ็กต์เป็นจำนวนมาก ๆ นั้นค่อนข้างจะเป็นวิธีที่สิ้นเปลืองที่จะต้องสร้างออบเจ็กต์ใหม่ ทุกๆ ครั้งที่มีการเริ่มต้นทรานแซคชั่น Object Pool เป็นวิธีการที่ถูกนำมาใช้ในการแก้ปัญหาดังกล่าวโดยเมื่อไคลเอนต์จะทำการ Activate ออบเจ็กต์นั้น COM+ Runtime จะส่งอินแสตนซ์จาก Object Pool มาให้ถ้าหากมีออบเจ็กต์ชนิดนั้นอยู่ใน Object Pool แต่หากไม่มีก็จะทำการสร้างอินแสตนซ์ใหม่ขึ้นมา ไคลเอนต์สามารถใช้ออบเจ็กต์ได้นานเท่าที่มันต้องการและเมื่อไคลเอนต์ปล่อยออบเจ็กต์นั้น ออบเจ็กต์ก็จะไม่ถูกทำลายแต่จะถูกนำกลับไปไว้ที่ Object Pool เพื่อรอการเรียกใช้ครั้งต่อไป ซึ่งจะมีลักษณะดังรูปที่ 3-9 สำหรับการที่จะทำให้ออบเจ็กต์มีความสามารถที่เป็นออบเจ็กต์ที่สามารถใช้ใน Object Pool ได้นั้นก็สามารถจะเซตแอททริบิวต์ได้เหมือนกับบริการอื่น ๆ ของ COM+ Runtime โดยมีขั้นตอนดังนี้

1. คลิกขวาที่คอมโพเน็นต์ในโฟลเดอร์ Components ของแอพพลิเคชั่น
2. เลือก Properties... จากเมนู Context จะปรากฏไดอะลอก Component Properties
3. เลือกหน้า Activation
4. เซตเชคบ็อกซ์ Enable object pooling
5. กด OK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-9 ไดอะล็อก Component Properties



รูปที่ 3-10 Object Pooling

จากวิธีที่ได้กล่าวมานั้นเป็นวิธีที่จะเซตให้ออบเจ็กต์สามารถนำมาใช้ร่วมกัน (Pool) ได้แต่ก่อนที่เซต ค่านั้นได้ตัวออบเจ็กต์จะต้องมีลักษณะดังนี้ด้วยคือ

1. ต้องอยู่ในอพารต์เมนต์ชนิด Thread Neutral หรือ Free-Thread

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ต้องไม่มีสถานะ

3. ต้องไม่มี Thread-affinity

4. ต้องสามารถรวมกันได้ (aggregatable)

นอกจากจะเซตให้ออบเจ็กต์นั้นสามารถใช้งานร่วมกันได้แล้วยังสามารถกำหนดจำนวนอินแสตนซ์ของออบเจ็กต์ที่ต้องมีอย่างน้อยที่สุด และมากที่สุดเท่าที่จะมีได้ โดยหากออบเจ็กต์ที่มีอยู่ใน Object Poolยัง น้อยกว่าจำนวนที่ต้องมีอย่างน้อยที่สุด เมื่อมีการเรียกใช้จากไคลเอนต์ตัวใหม่ COM+ Runtime ก็จะสร้างอินแสตนซ์ใหม่ให้ทันที แต่หากมีอยู่เกินกว่าจำนวนที่น้อยที่สุดแล้วแต่ถูกใช้อยู่ทุกตัว และยังมีจำนวนที่น้อยกว่าจำนวนที่มากที่สุดเท่าที่จะมีได้แล้ว ก็จะสร้างอินแสตนซ์ใหม่ให้กับไคลเอนต์นั้น แต่ถ้าเกินขอบเขตที่มากที่สุดแล้ว ไคลเอนต์จะต้องรอจนกว่าจะมีออบเจ็กต์ว่าง แล้วจึงนำเอาออบเจ็กต์นั้นไปใช้งานได้

สำหรับ COM+ ออบเจ็กต์ที่ต้องการแสดงสถานะว่าสามารถที่จะถูกนำกลับไปที่ Pool ได้ หรือให้ทำการ Activate และ Deactivate เองต้องทำการอิมพลีเมนต์อินเตอร์เฟส IObjectControl โดยมีเมธอด Activate และ Deactivate ที่ใช้ในการ Activate และ Deactivate ออบเจ็กต์ตามลำดับ และ เมธอด CanBePooled ที่ใช้ในการบอก COM+ Runtime ว่าสามารถนำกลับไปใช้ได้หรือไม่ในขณะนั้น

3.12 โมเดลความปลอดภัยของ COM+ (COM+ Security Model)

COM+ มีกลไกที่ช่วยรักษาความปลอดภัย เรียกว่า COM+ Security ซึ่งช่วยให้ผู้พัฒนาไม่ยุ่งยากในการจัดการกับกลไกการรักษาความปลอดภัยของวินโดวส์หรือ RPC ซึ่งคอมโพเน็นต์ต่างๆ จะสามารถสืบทอดข้อดีจากบริการรักษาความปลอดภัยตัวใหม่ได้อย่างอัตโนมัติ โมเดลรักษาความปลอดภัยนี้ถูกสร้างขึ้นมาจาก Security Support Provider Interface

ไมโครซอฟต์วินโดวส์ 2000 สนับสนุนโมเดลรักษาความปลอดภัยแบบกระจาย โดยมุ่งประเด็นไปใน2 เรื่องหลัก คือ

- การพิสูจน์ตน (Authentication) โดยปกติการพิสูจน์ตนจะถูกเรียกใช้เมื่อผู้ใช้ล็อกอินเข้าสู่ระบบเน็ตเวิร์กหรือเรียกใช้บริการจากเครื่องอื่นและระบบต้องการรู้ว่าผู้ใช้งานนั้นมีสิทธิใช้งานจริง

- การควบคุมการเข้าถึง (Access control) ใช้จำกัดสิทธิผู้ใช้แต่ละคนว่าอนุญาตให้ใช้บริการนั้นๆ หรือไม่

3.12.1 เป้าหมายหลักของโมเดลรักษาความปลอดภัยใน COM+

มี 4 ประการ คือ

1. Activation Control

2. Access Control

3. Authentication Control

4. Identify Control

Activation Control ใช้ควบคุมว่าใครสามารถ launch คอมโพเน็นต์ได้ ส่วน *Access Control* เป็นการจำกัดการเข้าถึงออบเจ็กต์ของคอมโพเน็นต์ (คือ ในบางกรณีผู้ใช้อาจได้รับอนุญาตให้เข้าถึงฟังก์ชัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บางส่วนของคอมโพเน็นต์เท่านั้น) ตัวอย่างเช่น ผู้ใช้ทั่วไปทำการเรียกใช้ผ่านทางเว็ปจะได้รับอนุญาตให้เรียกใช้แค่บางฟังก์ชั่นของคอมโพเน็นต์เท่านั้น แต่ถ้าเป็นผู้ดูแลระบบจะสามารถเรียกใช้ได้ทุกฟังก์ชั่นเป็นต้น

Authentication Control นั้นถูกใช้เพื่อให้แน่ใจได้ว่า ข้อมูลที่ส่งในระบบเน็ตเวิร์กนั้นจะสามารถเห็นได้โดยผู้ใช้ที่มีสิทธิ์เท่านั้น นอกจากนี้เรายังสามารถกำหนดระดับของ Authentication Control เพื่อเข้ารหัส ข้อมูลก่อนที่จะส่งผ่านเครือข่าย

ส่วน *Identify Control* ใช้กำหนดหลักฐานด้านความปลอดภัยของคอมโพเน็นต์ที่ทำงาน ซึ่งอาจได้มาจากการปรับแต่งที่บัญชีผู้ใช้หรือจากไคลเอ็นต์แอพพลิเคชั่นการกำหนดความปลอดภัยของ COM+ คอมโพเน็นต์ สามารถทำได้ 2 วิธีคือ

1. Declarative Security
2. Programmatic Security

Declarative Security เป็นการกำหนดความปลอดภัยใน COM+ catalog ภายนอกคอมโพเน็นต์ซึ่งกระทำโดยผู้ดูแลระบบ ส่วน Programmatic Security เป็นการเขียนโปรแกรมเพื่อควบคุมความปลอดภัยทำโดยผู้พัฒนาคอมโพเน็นต์

โมเดลความปลอดภัยทั้ง 4 แบบด้านบนสามารถใช้ Declarative Security กำหนดได้ทั้งหมดผ่านทาง COM+ catalog โดยใช้ Component Services administrative tools หรือ Distributed COM Configuration utility (dcomcnfg.exe) (ในปัจจุบันใช้ในการจัดการกับ unconfigured components) แต่ Programmatic Security ไม่สามารถใช้กับโมเดลแบบ Activation Security และ Identity Security ได้ เพราะการกำหนดค่าเหล่านี้ต้องถูกกำหนดก่อนที่คอมโพเน็นต์จะถูก launch

3.12.2 COM+ Security Packages

COM+ จำเป็นต้องมีโมเดลรักษาความปลอดภัยที่สามารถใช้ได้ทุกๆ แพล็ตฟอร์มที่มีการใช้ COM+ Services จึงไม่ควรใช้ระบบรักษาความปลอดภัยของวินโดวส์ ดังนั้น COM+ ได้กำหนดโมเดลรักษาความปลอดภัยระดับสูงขึ้นมาโดยใช้กลไกรักษาความปลอดภัยของพื้นฐานของวินโดวส์และ RPC โมเดลรักษาความปลอดภัยที่ยืดหยุ่นนี้ถูกสร้างบน Security Support Provider Interface (SSPI) ซึ่งเป็น API พื้นฐานที่ออกแบบมาเพื่อสนับสนุนความปลอดภัยในสภาวะแวดล้อมแบบกระจายการทำงาน

การออกแบบของ SSPI คล้ายกับสถาปัตยกรรมของ ODBC โดยการอิมพลีเม็นต์ SSPI API ใน DLL (เรียกว่า Security Service Provider (SSP)) ทำให้แอพพลิเคชั่นที่เขียนให้ทำงานกับ security provider หนึ่ง สามารถปรับเปลี่ยนไปใช้กับ security provider อื่นได้โดยการปรับแต่งเพียงเล็กน้อย ทุก security providers ต้องอิมพลีเม็นต์ SSPI ซึ่งภายในอาจใช้กลไกรักษาความปลอดภัยใดก็ได้

วินโดวส์ 2000 สนับสนุนโมเดลรักษาความปลอดภัยทั้งแบบ NTLMSSP ซึ่งเป็นการกลไกการยืนยันสิทธิโดยไม่ต้องส่งรหัสผ่านเข้าสู่ระบบเน็ตเวิร์กเลย และเคอบิรอส (Kerberos)

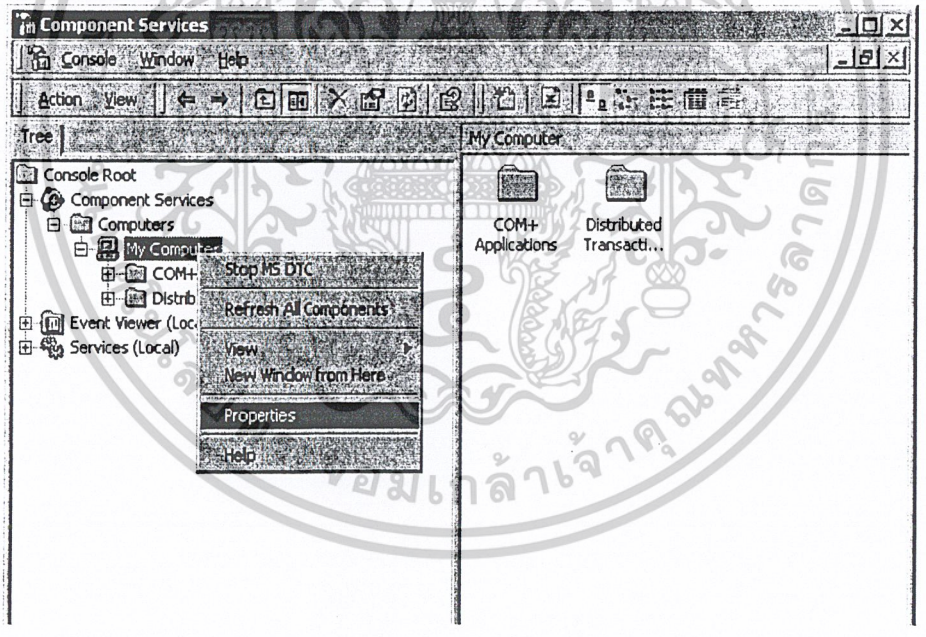
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.12.3 Declarative Security

COM+ catalog ใช้ในการปรับแต่งข้อมูลเกี่ยวกับโมเดลรักษาความปลอดภัยของ COM+ ซึ่งเรียกใช้ได้ผ่านทาง Component Services administration tool

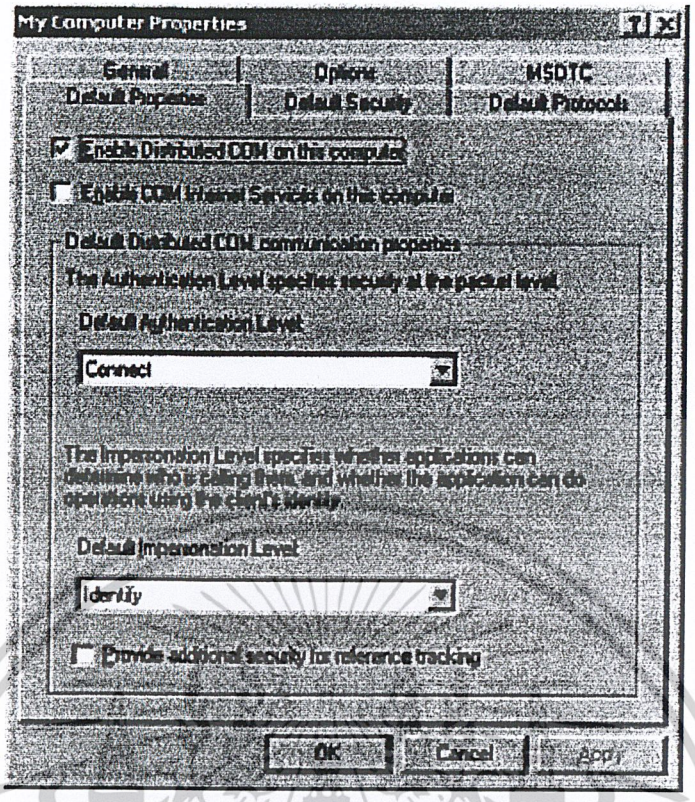
ข้อมูลของ Declarative security แบ่งออกเป็น default security และ component security โดย default security ใช้กำหนดค่าปกติในการ launch และการเข้าถึงทุกๆ คอมโพเน็นต์ที่กำลังใช้งานบนเครื่องนั้นๆ ส่วน component security ใช้กำหนดความปลอดภัยให้แก่คอมโพเน็นต์แต่ละตัว ซึ่งการกำหนดค่าที่นี้จะทับค่าที่ได้กำหนดไว้ใน default security (คือ override) และถ้ามีการกำหนดค่าที่ Programmatic security ค่าที่กำหนดไว้ที่ Programmatic security ก็จะทับทั้งค่าที่กำหนดไว้ใน default security และ component security

เปิด Component Services administration tool คลิ๊กขวาที่ My Computer จากนั้น เลือก DefaultProperties tab ดังรูปที่ 3-11 จะได้ผลดังรูปที่ 3-12 จากรูปตัวเลือก “Enable Distributed COM on this computer” หมายถึงว่าให้สามารถใช้ DCOM บนเครื่องนี้ได้ ซึ่งถ้าตัวเลือกนี้ไม่ถูกเลือกแล้ว จะไม่สามารถทำการเรียกข้ามเครื่องได้เลย ส่วนตัวเลือก "Enable COM Internet Services on this computer" นั้นใช้เลือกให้ COM+ Internet Service นั้นสามารถใช้งานได้หรือไม่ (ค่าปกติ คือ ไม่เลือก) เราสามารถกำหนดค่าระดับรักษาความปลอดภัยในระบบได้ในแท็ปต่างๆ เหล่านี้

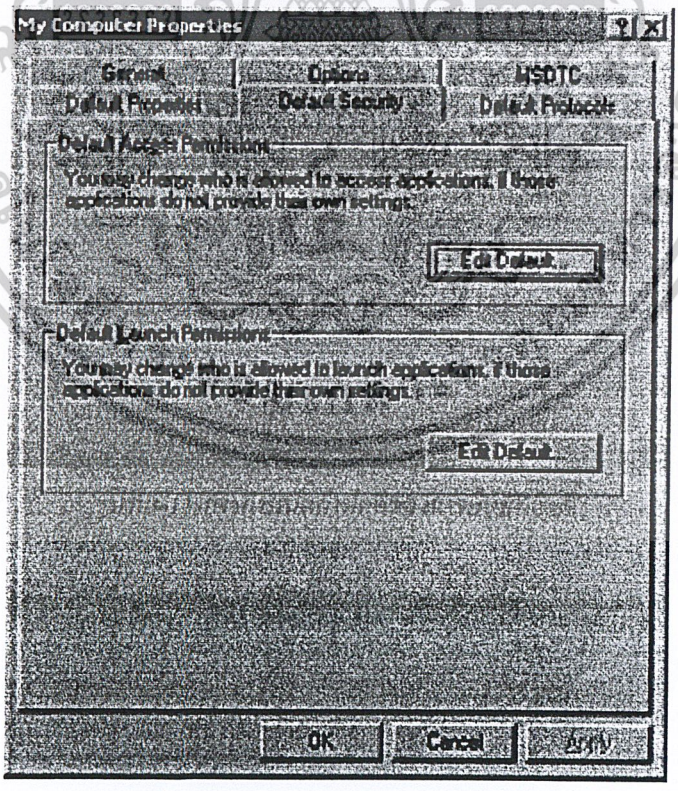


รูปที่ 3-11 การเปิดเมนูเพื่อกำหนดค่ารักษาความปลอดภัย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3-12 การกำหนดค่าปกติของการยืนยันสิทธิ



รูปที่ 3-13 การกำหนดค่าปกติของการเข้าถึง

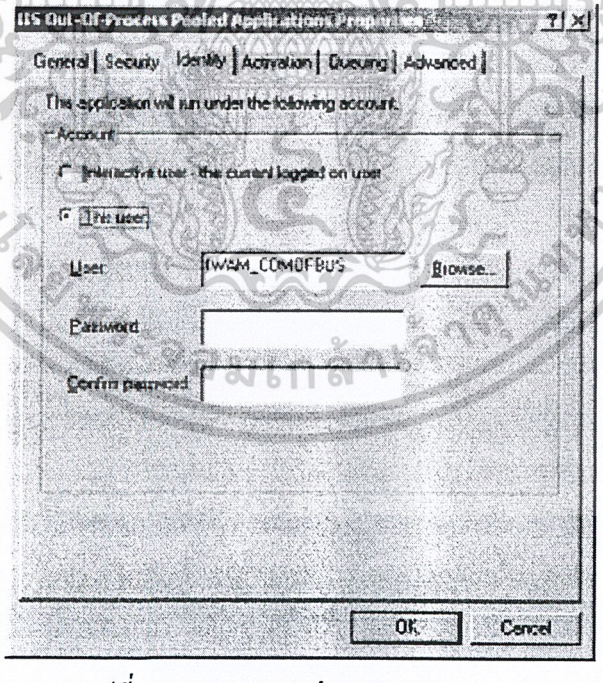
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ก่อนหน้าที่จะมี COM+ เราสามารถกำหนดความปลอดภัยแบบ Declarative Security โดยใช้แอพพลิเคชั่น dcomcnfg ปัญหาหลักของ dcomcnfg คือค่าความปลอดภัยจะครอบคลุมทั้งหมดของ COM เซิร์ฟเวอร์, COM คลาสทั้งหมดที่อยู่ใน COM เซิร์ฟเวอร์เดียวกันมีค่าความปลอดภัยเหมือนกันหมด ซึ่งไม่เหมาะสมหาก COM เซิร์ฟเวอร์ประกอบด้วยคลาสหรืออินเตอร์เฟสที่ต้องการค่าความปลอดภัยแตกต่างกัน ตัวอย่างเช่น เรามีคอมโพเน็นต์ที่ประกอบด้วย 2 อินเตอร์เพส อินเตอร์เฟสแรกประกอบด้วยเมธอดต่างๆ ที่จัดเตรียมไว้ให้ผู้ใช้ทั่วไปใช้งาน ส่วนอินเตอร์เฟสที่สองเป็นของผู้ดูแลระบบเพื่อใช้ในการจัดการควบคุมคอมโพเน็นต์โดยไม่อนุญาตให้ผู้ใช้ทั่วไปใช้อินเตอร์เฟสนี้ ในกรณีตัวอย่างที่กล่าวมา dcomcnfg ไม่สามารถทำได้ วิธีเดียวที่ทำได้คือใช้ API ของ COM Security อย่างไรก็ดีในการใช้ COM+ รันไทม์ ก็ยังสามารถใช้ประโยชน์จากวิธีนี้ได้เมื่อต้องการให้ Role ของผู้ใช้เป็นเงื่อนไขสำหรับการทำงานของเมธอดในคอมโพเน็นต์ เช่นให้บริการที่แตกต่างกันจากเมธอดเดียวกัน เมื่อผู้ใช้อยู่ใน Role ที่แตกต่างกันเป็นต้น

สำหรับ COM+ เราสามารถควบคุมการใช้เซิร์ฟเวอร์ได้ตั้งแต่ระดับคลาส อินเตอร์เฟส และเมธอดซึ่ง เรียกว่า Fine-Grained Security โดยที่ไม่ต้องเขียนโปรแกรมเพิ่มเติม การใช้ Fine-Grained Security เราต้องทำความเข้าใจเกี่ยวกับโรล (Role) ซึ่งเป็นสิ่งสำคัญในการกำหนดความปลอดภัยบน COM+

3.12.4 การปรับแต่งคอมโพเน็นต์แต่ละตัว

การปรับแต่งคอมโพเน็นต์แต่ละตัว ทำได้โดยคลิ๊กขวาที่คอมโพเน็นต์ที่ต้องการใน Component Services administration tool จะได้ดังรูปที่ 3-14



รูปที่ 3-14 การกำหนดค่า Identiry Security

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.12.5 Role

COM+ ใช้โมเดลความปลอดภัยแบบใหม่โดยแสดงอยู่ในรูปแบบของโรล โรลคือกลุ่มของผู้ใช้ COM+ แอพพลิเคชั่นที่มีค่าความปลดภัยเหมือนกัน การกำหนดโรลมีลักษณะเหมือนชีวิตจริงที่มีการกำหนดบทบาทให้บุคคลว่ามีบทบาทอะไรในสังคม สามารถทำหรือไม่อนุญาตให้ทำอะไรได้บ้าง ตัวอย่างเช่น COM+ แอพพลิเคชั่นที่ออกแบบสำหรับธนาคารแบ่งโรลออกเป็น Teller role, Manager role และ Loan Office role ผู้ใช้ที่อยู่ในโรลเดียวกันมีค่าความปลอดภัยเหมือนกัน แต่ละโรลอนุญาตและไม่อนุญาตให้ทำอะไรเหมือนกัน เมื่อกำหนดโรลให้กับ COM+ แอพพลิเคชั่น เราสามารถเพิ่มผู้ใช้หรือกลุ่มของผู้ใช้เข้าไปในโรลได้ 2 วิธี คือใช้คอมโพเน็นต์เซอร์วิสเอ็กซพลอเรอร์ (Declarative Security) หรือเขียนโปรแกรมโดยเรียกข้อมูลจากคอนเท็กซ์เพื่อดูว่าผู้ใช้มีโรลอะไร หากเป็นโรลที่ไม่อนุญาตก็สามารถหยุดการใช้งานหรือแจ้งความผิดพลาดที่เกิดขึ้น (Programmatic Security)

3.12.5.1 การสร้าง Role

เราสามารถเพิ่มโรลให้ COM+ Application โดยมีขั้นตอนดังนี้

1. คลิกขวาที่โฟลเดอร์ Roles ของแอพพลิเคชั่นที่ต้องการเพิ่มโรล
2. เลือก New > Role จะปรากฏไดอะลอก Role
3. ตั้งชื่อโรลที่ต้องการแล้วคลิก OK

เมื่อสร้างโรลแล้วต่อมาคือเพิ่มผู้ใช้เข้าไปในโรลและกำหนดส่วนต่าง ๆ ของแอพพลิเคชั่น (คอมโพเน็นต์, อินเตอร์เฟส, เมธอด) ว่าต้องการให้โรลใดเรียกใช้ได้

3.12.5.2 การเพิ่มผู้ใช้เข้าไปในโรล

มีขั้นตอนดังนี้

1. คลิกขวาที่โฟลเดอร์ Users ที่อยู่ในไดเรกทอรีของโรลที่ต้องการเพิ่มผู้ใช้
2. เลือก New > Users จะปรากฏไดอะลอกให้เลือกผู้ใช้ (ไดอะลอกที่ปรากฏขึ้นอยู่ว่ามีใช้Active Directory หรือไม่)
3. เลือกผู้ใช้ที่ต้องการแล้วคลิก OK

3.12.6 การกำหนดความปลอดภัยให้แต่ละส่วนของแอพพลิเคชั่น

ก่อนอื่นเราต้องเลือกออบชั่น Access Control จึงจะใช้งานได้ โดยค่าดีฟอลต์แล้ว Access Control จะไม่ได้เลือกไว้ การเลือกออบชั่น Access Control มีขั้นตอนดังนี้

1. เปิดคอมโพเน็นต์เซอร์วิสเอ็กซพลอเรอร์
2. คลิกขวาที่แอพพลิเคชั่นที่ต้องการ เลือก Properties
3. คลิกที่แท็บ Security ของไดอะลอกคอมโพเน็นต์ที่ปรากฏขึ้นมา
4. เลือกเช็คบ็อกซ์ Enforce access checks for this application แล้วคลิก Apply หรือ OK การเปลี่ยนแปลงค่าความปลอดภัยของ COM+ จะยังไม่เกิดผลทันทีจนกว่าเราจะรีสตาร์ทแอพพลิเคชันใหม่

การกำหนดความปลอดให้แต่ละส่วนของแอพพลิเคชั่นมีขั้นตอนดังนี้

1. คลิกขวาที่คอมโพเน็นต์ อินเตอร์เฟส หรือเมธอดที่ต้องการ เลือก Properties
2. คลิกที่แท็บ Security ของไดอะลอกที่ปรากฏขึ้นมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. เลือกโรลที่ต้องการที่อยู่ Role explicit set for selected item(s):

4. คลิก Appy หรือ OK

3.12.7 Programmatic Security

เราใช้ declarative security สำหรับกำหนดความปลอดภัยเป็นส่วนใหญ่ และใช้ programmatic security สำหรับงานประเภทพิเศษที่ต้องการควบคุมสูง เนื่องจาก Declarative security ไม่สามารถปรับแต่ง role ภายในเมธอดเดียวได้ แต่เราสามารถทำได้ในส่วน programmatic นี้ โดยการใช้ context object ที่ COM+ มีให้ เพื่อเรียกเมธอด IObjectContext::IsSecurityEnabled และ IObjectContext::IsCallerInRole

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
เว็บเซอร์วิส

Web Services คือ แอพพลิเคชั่น หรือ โปรแกรมที่ทำงานอย่างใดอย่างหนึ่ง ในลักษณะให้บริการ โดยจะถูกเรียกใช้งานจาก แอพพลิเคชั่น อื่นๆ ในรูปแบบ RPC (Remote Procedure Call) ซึ่งการเปลี่ยนคือ XML ทำให้เราสามารถเรียกใช้คอมโพเน็นต์ ใดๆ ก็ได้ ใน platform ใดๆ ก็ได้ บนโพรโตคอล HTTP ซึ่งเป็นโพรโตคอลสำหรับ World Wide Web อันเป็นช่องทางที่ได้รับการยอมรับทั่วโลกในการติดต่อสื่อสารกันระหว่างแอพพลิเคชั่นกับแอพพลิเคชั่นในปัจจุบัน

เทคโนโลยีในการกระจายข่าวสารข้อมูลทาง Internet ในปัจจุบันก็คือ เว็บเพจ แต่จากการที่มันมีความสามารถที่จะทำงานได้ด้วยการรวมภาษาทั้ง Client และ Server Side Script ไว้ในตัวเองเช่นภาษา VBScript, Java Script หรือ ASP, PHP, JSP นั้นทำให้เว็บเพจมีลักษณะคล้าย Application จึงถูกเรียกรวมกันว่า Web Application

Web Application สามารถตอบสนองความคิด Distributed Processing ได้ในระดับหนึ่งซึ่งก็คือการแบ่งการประมวลผลไว้ที่ฝั่ง Client และฝั่ง Server และมักจะมีการใช้ database ควบคู่กับการทำ Web Application ไปด้วยตามความต้องการในการทำ E-Business และ E-Commerce ที่กำลังเป็นที่นิยมในปัจจุบัน และเกิดปัญหาที่ตามาคือ เรื่องของการจ่ายเงินหรือที่เรียกว่า E-payment หรือ Payment-Gateway ซึ่ง Web Application ที่ทำ E-Commerce ต้องใช้บริการจากธนาคาร Online ในการจัดเก็บเงินกับลูกค้า เพราะด้วยเทคโนโลยีนี้การใช้บริการเก็บเงินจากธนาคาร Online จำเป็นที่ผู้ค้าต้องไปทำการตกลงกับธนาคารและเขียนโปรแกรมให้ตรงตามมาตรฐานที่ธนาคาร Online กำหนดไว้

ด้วยปัญหายุ่งยากในการค้นหา ติดต่อและตกลงในการขอใช้บริการเก็บเงินจากธนาคาร Online แนวคิด Web Serviceจึงดูเหมือนเป็นทางออกของปัญหานี้ ความเด่นของเทคโนโลยี Web Service นี้ก็คือการทำให้ Web กับ Web สามารถติต่อสื่อสารกันได้ด้วยเอกสาร XML ที่ทั้งคนและคอมพิวเตอร์เข้าใจ และคอมพิวเตอร์ยังสามารถนำข้อมูลนั้นไปประมวลผลต่อได้ ด้วยเอกสาร XML นี้เองทำให้ Web สามารถส่งข้อมูลที่จำเป็นไปให้อีก Web หนึ่งทำงานบางอย่างให้หรือใช้บริการนั่นเอง ทำให้เป็นการง่ายที่จะเขียนโปรแกรมที่จะติดต่อสื่อสารหรือขอใช้บริการเก็บเงินจากธนาคาร Online แต่สำหรับ Web Application ที่ใช้การส่งข้อมูลเป็น html ทำให้ข้อมูลนั้นไม่สามารถนำไปใช้ต่อได้ การเขียนโปรแกรมจึงยุ่งยากตามที่กล่าวด้านบน

แนวคิดของ Web Service ก็คือ Web ที่สามารถทำงานอะไรบางอย่างหรือก็คือให้บริการบางอย่างจากการร้องขอจากต่าง Server ด้วยเหตุนี้ทำให้เทคโนโลยี Web Service เอื้อต่อแนวคิด Distributed Processing มากกว่า Web Application และเมื่อประกอบกับการที่ Web Service มี UDDI ทำให้ Web Service สามารถค้นหาบริการต่างๆ ที่ต้องการได้จากทั่วทุกมุมโลก

ในอนาคตอาจเป็นไปได้ว่า Application ของก็อาจเป็นเพียงแค่การรวม Service ที่แต่ละ Web Service มีบริการมาให้ใช้เท่านั้นเอง

องค์ประกอบในการใช้งาน .NET เว็บเซอร์วิส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1 XML

4.1.1 แนะนำ XML

XML ย่อมาจาก eXtensible Markup Language XML ถูกออกแบบมาเพื่อใช้อธิบายข้อมูล ใน XML จะไม่มีแท็ก (tag) ที่กำหนดไว้ล่วงหน้า เราจะต้องสร้างแท็กของเราขึ้นมาเอง และจะใช้ Document Type Definition (DTD) ในการอธิบายรูปแบบของข้อมูล

ข้อแตกต่างที่สำคัญระหว่าง XML และ HTML คือ XML ถูกออกแบบมาเพื่อใช้แลกเปลี่ยนข้อมูล ไม่ใช่มาแทนที่ HTML XML กับ HTML ถูกออกแบบมาเพื่อจุดประสงค์ที่ต่างกัน HTML จะเกี่ยวข้องกับการแสดงข้อมูล XML จะเกี่ยวข้องกับการอธิบายข้อมูล แท็กของ HTML เช่น <H1></H1> เป็นแท็กสำหรับบอกรูปแบบการแสดงผล ในขณะที่แท็กของ XML ต้องสร้างขึ้นมาเองเช่น

```
<BOOK>
   <NAME>Beginning XML</NAME>
   <ISDN>1-861003-41-2</ISDN>
   <PAGE>800</PAGE>
<BOOK>
```

จะใช้แสดงข้อมูลของหนังสือที่ประกอบด้วย element BOOK , NAME , ISDN และ PAGE ข้อมูลที่อยู่ในรูปแท็กซ์นี้เป็นอิสระต่อซอฟต์แวร์และฮาร์ดแวร์ จึงเหมาะที่จะนำมาใช้ในการแลกเปลี่ยนข้อมูลระหว่างระบบบนอินเทอร์เน็ต การเปลี่ยนข้อมูลให้อยู่ในรูป XML เป็นการลดความซับซ้อนและสามารถสร้างข้อมูลที่ถูกอ่านโดยแอพพลิเคชันใด ๆ ก็ได้ นอกจากนั้นการเพิ่มเติม element จะไม่ทำให้แอพพลิเคชันเสียหาย เนื่องจากการอ่านข้อมูลโดย XML parser จะใช้วิธีแยกข้อมูลจาก element ที่ต้องการ จาก XML ข้างต้นนี้จะแยกข้อมูล element NAME , ISDN และ PAGE หากเพิ่ม element AUTHOR เข้าไป แอพพลิเคชันก็ยังคงดึงข้อมูลใน element เดิมโดยจะมองข้าม element ที่ไม่ได้กำหนดให้แยกข้อมูลออกมาในตอนพัฒนาแอพพลิเคชัน

4.1.2 ไวยากรณ์ของ XML

กฎไวยากรณ์ของ XML นั้นง่ายและเคร่งครัดมาก กฎของมันง่ายต่อการเรียนรู้และง่ายต่อการใช้ ด้วยเหตุนี้การสร้างซอฟต์แวร์ในการอ่านและจัดการกับข้อมูลจึงเป็นเรื่องง่าย ตัวอย่างของเอกสาร XML ดังนี้

```
<?XML version="1.0"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

ในบรรทัดแรกคือการประกาศเอกสาร XML ที่บอกถึง XML เวอร์ชัน

บรรทัดต่อมาคือ root element (note) อีก 4 บรรทัดถัดมาเป็นการอธิบายถึง 4 child element ของ root (to , from , heading , body) และบรรทัดสุดท้ายเป็นการกำหนดจุดสิ้นสุดของ root

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กฎของเอกสาร XML มีดังนี้

- **XML Element ทุกตัวต้องมีการปิดแท็ก**

ใน HTML บาง element ไม่จำเป็นต้องมีการปิดแท็ก แต่ใน XML ทุก element จำเป็นที่จะต้องมีการปิดแท็ก

This is a paragraph	X
This is a paragraph	√

- **XML แท็กมีความแตกต่างระหว่างตัวพิมพ์เล็กและตัวพิมพ์ใหญ่ (case sensitive)**

ใน XML แท็ก <Letter> นั้นต่างจาก <letter> เนื่องจาก XML นั้นมีคุณสมบัติที่แยกความแตกต่างระหว่างตัวพิมพ์เล็กและตัวพิมพ์ใหญ่

<Message>This is incorrect</message>	X
<message>This is correct</message>	√

- **XML Element ทุกตัวต้องมีการซ้อนกันอย่างเหมาะสม**

ใน HTML บาง element สามารถเหลื่อมกันได้ แต่ใน XML นั้นต้องซ้อนกันเป็นขั้นโดยสมบูรณ์

This text is bold and italic	X
This text is bold and italic	√

- **ทุกเอกสาร XML ต้องมี root แท็ก**

แท็กแรกในเอกสาร XML นั้นจะเป็น root แท็กและจะมีได้เพียงหนึ่งเดียวเท่านั้น element อื่น ๆ ทั้งหมดจะต้องถูกซ้อนอยู่ภายใน root element เท่านั้น และ element เหล่านั้นก็สามารถมี element ย่อย (child element) ได้อีก ดังนี้

```
<root>
   <child>
      <subchild>.....</subchild>
   </child>
</root>
```

- **ค่าของ Attribute ต้องมีอยู่ภายในเครื่องหมายคำพูด (“”)**

XML element สามารถมี attributeได้ โดยที่ค่าของ attribute จะต้องอยู่ภายในเครื่องหมายคำพูด (“) เอกสาร XML ข้างล่างนี้ อันแรกผิด อันที่สองถูก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<note date=12/11/99>	X
<note date="12/11/99">	√

4.1.3 XML Element

XML element มีคุณสมบัติดังนี้

- **XML Element สามารถขยายได้**

ตามตัวอย่างนี้

```
<note>
    <to>Tove</to>
    <from>Jani</from>
    <body>Don't forget me this weekend!</body>
</note>
```

ถ้าเราจะสร้างแอพพลิเคชันที่สามารถแยก element to , from และ body จาก เอกสาร XML เพื่อแสดงผลลัพธ์ดังนี้

MESSAGE

To: Tove

From: Jani

Don't forget me this weekend!

และ ถ้าเพิ่มข้อมูลเพิ่มเติมลงไปดังนี้

```
<note>
    <date>1999-08-01</date>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

เมื่อเอกสาร XML เป็นดังนี้แล้ว แอพพลิเคชันที่สร้างไว้จะไม่เสียหาย เนื่องจาก แอพพลิเคชันจะยังคงค้นหา element to , from และ heading ในเอกสาร XML และยังคงให้ผลลัพธ์เหมือนเดิม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **XML Element มีความสัมพันธ์กัน**

ในการที่จะเข้าใจถึงภาพรวมของ XML จะต้องเข้าใจถึงความสัมพันธ์ระหว่างชื่อของ XML element ที่ตั้งและ content ของ element

คำอธิบายของหนังสือดังนี้

```
Book Title: My First XML
Chapter 1: Introduction to XML
    What is HTML
    What is XML
Chapter 2: XML Syntax
    Elements must have a closing tag
    Elements must be correctly nested
```

สามารถมีเอกสาร XML ที่อธิบายหนังสือนั้น ได้ดังนี้

```
<book>
    <title>My First XML</title>
    <prod id="33-657" media="paper"></prod>
    <chapter>Introduction to XML
        <para>What is HTML</para>
        <para>What is XML</para>
    </chapter>
    <chapter>XML Syntax
        <para>Elements must have a closing tag</para>
        <para>Elements must be properly nested</para>
    </chapter>
</book>
```

book เป็น root element title และ chapter เป็น child element ของ book book เป็น parent element ของทั้ง title และ chapter title และ chapter จึงเป็น sibling กันเนื่องจากมี parent เดียวกัน

- **Element มี Content**

XML element คือทั้งหมดตั้งแต่เปิดแท็กจนถึงปิดแท็ก element หนึ่งสามารถมี element content , mixed content , simple content หรือ empty content และยังมี attribute ได้ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากตัวอย่างข้างบน book มี element content เพราะมันประกอบด้วย element อื่น chapter เป็น mixed content เพราะมันประกอบด้วยเท็กซ์และ element อื่น ๆ para เป็น simple content (text content) เพราะมันประกอบด้วยเท็กซ์เท่านั้น prod เป็น empty element เนื่องจากมันไม่มีข้อมูลใดๆ และเฉพาะ prod element เท่านั้นที่มี attribute attribute ที่ชื่อ id มีค่าเท่ากับ "33-657" attribute ที่ชื่อ medis มีค่าเท่ากับ "paper"

- **การตั้งชื่อ Element**

XML element มีกฎการตั้งชื่อดังนี้

- ชื่อประกอบด้วย ตัวอักษร , ตัวเลข และอักขระอื่นๆ
- ชื่อต้องไม่ขึ้นต้นด้วยตัวเลขหรือ "_ "(underscore)
- ชื่อต้องไม่ขึ้นด้วยคำว่า XML (หรือ Xml หรือ xml)
- ชื่อไม่สามารถมีช่องว่างได้ (space)
- ชื่อต้องไม่ควรประกอบด้วย ":" เพราะมันถูกสงวนไว้กับการใช้ namespace

ทุกๆชื่อสามารถใช้ได้ ไม่มีคำใดเป็นคำสงวน แต่จะเป็นการดีในการที่จะหลีกเลี่ยงการใช้ "–" หรือ "." แล้วใช้เฉพาะ "_" เท่านั้น เนื่องจากอาจเกิดการสับสนของซอฟต์แวร์ในกรณีพยายามที่จะตัดคำ (first-name) หรือคิดว่า "name" เป็นคุณสมบัติของ object "first" (first.name)

4.1.4 XML Attribute

XML attribute มีคุณสมบัติดังนี้

- **รูปแบบของ Quote , "female" or 'female'**

ค่าของ attribute ต้องอยู่ในเครื่องหมายคำพูดเสมอ แต่อาจเป็น ' (single quote) หรือ " (double quote) ก็ได้

<person sex="female"> √
หรือ
<person sex='female'> √

Double quote จะเป็นที่ใช้โดยทั่วไปมากกว่า แต่ในบางครั้ง (ถ้าค่าของ attribute จำเป็นที่จะต้องมี quote) จึงจำเป็นที่จะต้องใช้ single quote ดังนี้

```
<gangster name='George "Shotgun" Ziegler'>
```

- **การใช้ Element และ Attribute**

ข้อมูลสามารถที่จะถูกเก็บใน child element หรือ attribute ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<person sex="female">
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

หรือ

```
<person>
  <sex>female</sex>
  <firstname>Anna</firstname>
  <lastname>Smith</lastname>
</person>
```

ตัวอย่างแรก sex เป็น attribute ตัวอย่างที่ 2 sex เป็น child element ซึ่งทั้งสองตัวอย่างให้ความหมายเดียวกัน มันไม่มีกฎว่าจะใช้ attribute เมื่อไร หรือ child element เมื่อไร แต่แนะนำว่าควรจะหลีกเลี่ยงการใช้attribute ควรจะใช้ child element ถ้าอินฟอร์เมชันนั้นดูเหมือนจะเป็นข้อมูล (data)

- **หลีกเลี่ยงการใช้ Attribute**

  บางปัญหาของการใช้ attribute

  - attribute ไม่สามารถมีหลายค่า (child element สามารถมีได้)
  - attribute ไม่ง่ายที่จะขยาย (ในอนาคต)
  - attribute ไม่สามารถอธิบายโครงสร้าง (child element สามารถทำได้)
  - attribute ยากที่จะจัดการด้วยโค้ดของโปรแกรม
  - ค่าของ attribute ไม่ง่ายในการจะตรวจสอบกับ DTD ที่มีอยู่

  แต่อย่างไรก็ตาม จะสามารถใช้ attribute ก็ต่อเมื่อ อินฟอร์เมชันนั้นไม่สัมพันธ์กับข้อมูล ตัวอย่างเช่น ในบางครั้ง ID reference สามารถถูกใช้ในการเข้าถึง XML element

```
<messages>
  <note ID="501">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>
  <note ID="502">
    <to>Jani</to>
    <from>Tove</from>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    <heading>Re: Reminder</heading>
    <body>I will not!</body>
  </note>
</messages>
```

ID ในตัวอย่างนี้เป็น counter หรือ unique identifier ในการอ้างถึง note ที่ต่างกันของไฟล์ XML ไม่ใช่ส่วนหนึ่งของข้อมูล note

4.1.5 XML Validation

เอกสาร XML แบ่งการตรวจสอบความถูกต้องของเอกสารเป็น 2 ชนิดคือ

1) **"Well Formed" XML Document**

"Well Formed" XML document เป็นเอกสารที่เป็นไปตามกฎไวยากรณ์ XML ที่ได้กล่าวมาข้างต้น

2) **"Valid" XML Document**

"Valid" XML document เป็นเอกสาร XML ที่มีรูปแบบที่เป็นไปตามกฎของ DTD โดยจะต้องตรงตามโครงสร้างข้อมูลของเอกสารที่ได้ประกาศโดย DTD

- **XML DTD**

วัตถุประสงค์ของ DTD คือกำหนดโครงสร้างของเอกสาร XML เพื่อใช้ในการตรวจสอบว่าเอกสาร XML นั้นถูกต้องตามรูปแบบหรือไม่ เช่น element BOOK จะต้องมี element NAME เสมอ อาจจะมี element PAGE หรือไม่ก็ได้ เป็นต้น

- **XML Schema**

XML schema มีวัตถุประสงค์เหมือนกับ DTD จะถูกนำมาใช้แทน DTD เนื่องจาก

- ง่ายในการเรียนรู้มากกว่า DTD
- ใช้ได้ดีกว่า DTD
- schema เขียนด้วย XML
- รองรับ datatype
- รองรับ namespace

4.1.6 แนะนำ DTD

1) **การประกาศ DOCTYPE ภายใน**

ถ้า DTD ถูกรวมอยู่ในไฟล์ XML นั้น มันจะต้องอยู่ภายในการให้คำจำกัดความ DOCTYPE ตามไวยากรณ์ นี้

```
<!DOCTYPE root-element [element-declarations]>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างดังนี้

```
<?XML version="1.0"?>
<!DOCTYPE note [
   <!ELEMENT note (to,from,heading,body)>
   <!ELEMENT to     (#PCDATA)>
   <!ELEMENT from   (#PCDATA)>
   <!ELEMENT heading (#PCDATA)>
   <!ELEMENT body   (#PCDATA)>
]>
<note>
   <to>Tove</to>
   <from>Jani</from>
   <heading>Reminder</heading>
   <body>Don't forget me this weekend</body>
</note>
```

DTD ข้างบนนี้สามารถแปลความหมายได้ดังนี้

!DOCTYPE note (บรรทัดที่ 2) กำหนดว่าเป็นเอกสารของ type note

!ELEMENT note (บรรทัดที่ 3) กำหนดว่า note element มี 4 element คือ to , from , heading และ body

!ELEMENT to (บรรทัดที่ 4) กำหนดว่า to element เป็นข้อมูลชนิด #PCDATA

!ELEMENT from (บรรทัดที่ 5) กำหนดว่า from element เป็นข้อมูลชนิด #PCDATA

2) การประกาศ DOCTYPE ภายนอก

ถ้า DTD อยู่ภายนอกไฟล์ XML มันจะต้องอยู่ภายในการให้คำจำกัดความ DOCTYPE ตามไวยากรณ์นี้

```
<!DOCTYPE root-element SYSTEM "filename">
```

ตัวอย่างไฟล์ XML เป็นดังนี้

```
<?XML version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไฟล์ของ DTD ชื่อ note.dtd เป็นดังนี้

note.dtd

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

4.1.6.1 โครงสร้างของ XML

เอกสาร XML จะถูกสร้างจากส่วนต่างๆ ดังนี้

- Element – เป็นโครงสร้างหลักของ XML
- Tag - ใช้ในการกำหนด markup element
- Attribute - ให้ข้อมูลเพิ่มเติมกับ element
- Entity – เป็นตัวแปรที่ใช้ในการกำหนดเท็กซ์ทั่ว ๆ ไป entity reference เป็นการอ้างถึง entity เช่น entity ชื่อ domain มีค่าเท่ากับ www.w3.org entity reference คือ &domain หากภายในเอกสาร XML มี &domain ค่าของมันจะมีค่าเท่ากับ www.w3.org
- PCDATA -หมายถึง parse character data PCDATA เป็นเท็กซ์ที่จะถูก parse โดย parser
- CDATA -หมายถึง character data เป็นเท็กซ์ที่จะไม่ถูก parse โดย parser

4.1.6.2 การกำหนด Element ใน DTD

วิธีการในการประกาศ element จะเป็นดังนี้

- **การประกาศ Element**

ประกาศตามไวยากรณ์นี้

```
<!ELEMENT element-name category>
```

หรือ

```
<!ELEMENT element-name (element-content)>
```

-
- **Element ว่างเปล่า (Empty)**

ถูกประกาศโดยคีย์เวิร์ด EMPTY

```
<!ELEMENT element-name EMPTY>
```

DTD example:

```
<!ELEMENT br EMPTY>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
XML example:
<br />
```

- **Element ที่เป็นข้อมูลชนิดตัวอักษร**

  element ที่เป็นตัวอักษรเท่านั้นจะถูกประกาศด้วยคีย์เวิร์ด PCDATA ที่เปิดและปิดด้วย "(" ")"

```
<!ELEMENT element-name (#PCDATA)>
```

```
DTD example:
<!ELEMENT note (#PCDATA)>
```

- **Element ที่เป็นข้อมูลใด ๆ**

  ประกาศกับคีย์เวิร์ด ANY content จะสามารถเป็นข้อมูลชนิดใดก็ได้

```
<!ELEMENT element-name ANY>
```

```
DTD example:
<!ELEMENT note ANY>
```

- **Element และ Children (Sequences)**

  ประกาศโดยใช้ "," ในการแยก child element แล้วเปิดและปิดด้วย "( )"

```
<!ELEMENT element-name (child-element-name)>
```

หรือ

```
<!ELEMENT element-name (child-element-name,child-element-name,.....)>
```

```
DTD example:
<!ELEMENT note (to,from,heading,body)>
```

ถ้ามีการประกาศ child element จะต้องประกาศ child element ให้สมบูรณ์ด้วยดังตัวอย่างข้างล่างนี้

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to     (#PCDATA)>
<!ELEMENT from   (#PCDATA)>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body    (#PCDATA)>
```

- **การประกาศเพียงครั้งเดียวของ Element เดียวกัน**

```
<!ELEMENT element-name (child-name)>
```

```
DTD example:
<!ELEMENT note (message)>
```

จากตัวอย่างหมายความว่า note element จะประกอบด้วย child element message เท่านั้น

- **การประกาศอย่างน้อย 1 ครั้งของ Element เดียวกัน**

```
<!ELEMENT element-name (child-name+)>
```

```
DTD example:
<!ELEMENT note (message+)>
```

เครื่องหมาย + ถูกใช้ในความหมายหนึ่งหรือมากกว่า จึงหมายความว่า note element จะประกอบด้วย child element message ตั้งแต่ 1 ค่าขึ้นไป

- **การไม่ประกาศหรือประกาศเท่าไรก็ได้ของ Element เดียวกัน**

```
<!ELEMENT element-name (child-name*)>
```

```
DTD example:
<!ELEMENT note (message*)>
```

เครื่องหมาย * ถูกใช้ในความหมายว่าไม่มีหรือมีเท่าไรก็ได้ จึงหมายความว่า note element จะประกอบด้วย child element message กี่ค่าก็ได้ หรือไม่มีเลยก็ได้

- **การไม่ประกาศหรือประกาศเพียงครั้งเดียวของ Element เดียวกัน**

```
<!ELEMENT element-name (child-name?)>
```

```
DTD example:
<!ELEMENT note (message?)>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องหมาย ? ถูกใช้ในความหมาย ไม่มีหรือมีเพียงหนึ่ง จึงหมายความว่า note element จะประกอบด้วย child element message หนึ่งค่า หรือไม่มีก็ได้

- **การประกาศ Content แบบให้เลือก**

  จะใช้เครื่องหมาย | แทนความหมายหรือ

DTD example:

<!ELEMENT note (to,from,header,message|body)>

ดังนั้นตัวอย่างนี้จึงหมายถึง note element จะต้องประกอบด้วย to element , from element , header element แล้วก็ message element หรือ body element อย่างใดอย่างหนึ่งเท่านั้น

- **การประกาศ Content แบบผสม**

  เป็นการผสมการประกาศหลาย ๆ อย่างเข้าด้วยกัน

DTD example:

<!ELEMENT note (#PCDATA|to|from|header|message)*>

ตัวอย่างนี้หมายถึง note element จะประกอบด้วย character data หรือ element to , from , header , และ message จำนวนเท่าไรก็ได้

4.1.6.3 การกำหนด Attribute ใน DTD

วิธีการประกาศ attribute จะเป็นดังนี้

- **การประกาศ Attribute**

  attribute ต้องถูกประกาศตามไวยากรณ์ นี้

<!ATTLIST element-name attribute-name attribute-type default-value>

DTD example:

<!ATTLIST payment type CDATA "check">

XML example:

<payment type="check">

ชนิดของ attribute สามารถมีค่าได้ดังตารางที่ 4-1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่า	คำอธิบาย
CDATA	ค่าต้องเป็น character ที่จะไม่ถูก parse โดย parser
(en1|en2|..)	ค่าต้องเป็นอันใดอันหนึ่งในรายการนั้น
ID	ค่าต้องเป็น unique id
IDREF	ค่าต้องเป็น id ของ element อื่น
IDREFS	ค่าต้องเป็นรายการของ id อื่น โดยจะเว้นแต่ละรายการด้วยช่องว่าง
NMTOKEN	ค่าต้องเป็นชื่อที่ประกอบด้วยตัวอักษร , ตัวเลข , เครื่องหมายจุด (.) , เครื่องหมายขีดกลาง (-) , เครื่องหมายขีดล่าง (_) และสามารถมีเครื่องหมายโคลอน (:) ยกเว้นในตำแหน่งแรกได้
NMTOKENS	ค่าต้องเป็นรายการของ NMTOKEN โดยจะเว้นแต่ละรายการด้วยช่องว่าง
ENTITY	ค่าต้องเป็น entity
ENTITIES	ค่าต้องเป็นรายการของ entity
NOTATION	ค่าต้องเป็นชื่อของ notation (notation คือชนิดของข้อมูลภายนอกที่กำหนดขึ้นเอง เช่น GIF เพื่อบอกว่าเป็นไฟล์รูปภาพ)
XML:	ค่าต้องขึ้นต้นด้วย XML

ตารางที่ 4-1 ชนิดของ Attribute ใน XML

default-value สามารถมีค่าได้ดังตารางที่ 4-2

ค่า	คำอธิบาย
#DEFAULT value	ค่าดีฟอลต์ของ attribute
#REQUIRED	element ต้องมี attribute
#IMPLIED	element จะมี attribute หรือไม่ก็ได้
#FIXED value	attribute ต้องเหมือนกับที่กำหนดไว้

ตารางที่ 4-2 ค่าดีฟอลต์ของ Attribute ใน XML

- **ตัวอย่างของการประกาศ Attribute**

DTD example:

```
<!ELEMENT square EMPTY>
<!ATTLIST square width CDATA "0">
```

XML example:

```
<square width="100"></square>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างนี้หมายความว่า square element เป็น empty element กับ attribute width ซึ่งเป็นชนิด CDATA ถ้าไม่ระบุจะยึดค่า default เป็น 0

- **ค่าของ Attribute แบบดีฟอลต์**

```
<!ATTLIST element-name attribute-name attribute-type "default-value">
```

DTD example:

```
<!ATTLIST payment type CDATA "check">
```

XML example:

```
<payment type="check">
```

ถ้าในเอกสาร XML ไม่มีการระบุจะใช้ค่าดีฟอลต์ แต่ถ้าระบุก็จะใช้ตามที่ระบุไว้

- **Attribute โดยนัย**

```
<!ATTLIST element-name attribute-name attribute-type #IMPLIED>
```

DTD example:

```
<!ATTLIST contact fax CDATA #IMPLIED>
```

XML example:

```
<contact fax="555-667788">
```

จะใช้ #IMPLIED ในกรณีจะระบุ attribute หรือไม่ก็ได้ ถ้าไม่ระบุก็จะไม่มีค่า default ให้

- **Attribute ที่ต้องระบุเสมอ**

```
<!ATTLIST element-name attribute_name attribute-type #REQUIRED>
```

DTD example:

```
<!ATTLIST person number CDATA #REQUIRED>
```

XML example:

```
<person number="5677">
```

ใช้ในการบังคับให้ต้องระบุ attribute ถ้าในเอกสาร XML ไม่ระบุ attribute เอกสารนั้นจะไม่ถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **ค่าของ Attribute คงที่**

```
<!ATTLIST element-name attribute-name attribute-type #FIXED "value">
```

DTD example:

```
<!ATTLIST sender company CDATA #FIXED "Microsoft">
```

XML example:

```
<sender company="Microsoft">
```

ในแบบนี้ถ้าไม่ได้ระบุค่า attribute จะใช้ค่าดีฟอลต์ แต่ถ้าจะระบุต้องระบุให้ตรงกับค่าดีฟอลต์ มิฉะนั้นจะถือว่าผิด

- **ค่าของ Attribute กำหนดเอง**

```
<!ATTLIST element-name attribute-name (en1|en2|..) default-value>
```

DTD example:

```
<!ATTLIST payment type (check|cash) "cash">
```

XML example:

```
<payment type="check">
```

หรือ

```
<payment type="cash">
```

จะใช้เมื่อเราต้องการกำหนดค่าของ attribute เอง

4.1.6.4 การกำหนด Entity ใน DTD

- **การประกาศ Entity ภายใน**

```
<!ENTITY entity-name "entity-value">
```

DTD Example:

```
<!ENTITY writer "Donald Duck.">
<!ENTITY copyright "Copyright W3Schools.">
```

XML example:

```
<author>&writer;&copyright;</author>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **การประกาศ Entity ภายนอก**

```
<!ENTITY entity-name SYSTEM "URI/URL">
```

DTD Example:

```
<!ENTITY writer
 SYSTEM "http://www.w3schools.com/entities/entities.XML">
<!ENTITY copyright
 SYSTEM "http://www.w3schools.com/entities/entities.dtd">
```

XML example:

```
<author>&writer;&copyright;</author>
```

4.1.7 XML Namespace

XML namespace จะหาวิธีหลีกเลี่ยงความสับสนที่เกิดจากชื่อ element (Name Conflicts)

- *ความสับสนที่เกิดจากชื่อ*

ความสับสนที่เกิดจากชื่อ element จะเกิดขึ้นบ่อยเมื่อเอกสารที่แตกต่างกัน ใช้ชื่อเดียวกันในการอธิบายชนิดของ element ที่แตกต่างกัน เช่น table ใน 2 ตัวอย่างต่อไปนี้

```
<table>
  <tr>
  <td>Apples</td>
  <td>Bananas</td>
  </tr>
</table>
```

และ

```
<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

ถ้านำเอกสาร XML ทั้ง 2 มารวมกัน จะเกิดความสับสนจากชื่อขึ้นเพราะทั้ง 2 มี element (table) ซึ่งมีใช้ในความหมายที่ต่างกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ***การแก้ปัญหาความสับสนที่เกิดจากชื่อโดยใช้ Prefix***

เอกสาร XML ทั้ง 2 คือ

```
<h:table>
  <h:tr>
  <h:td>Apples</h:td>
  <h:td>Bananas</h:td>
  </h:tr>
</h:table>
```

และ

```
<f:table>
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

ความสับสนที่เกิดจากชื่อจะถูกกำจัดไปโดยการเพิ่ม prefix ทำให้ได้ element ที่ต่างกัน 2 ชนิด (<h:table> และ <f:table>)

- ***การใช้ Namespaces***

จากเอกสาร XML ทั้ง2 คือ

```
<h:table xmlns:h="http://www.w3.org/TR/html4/">
  <h:tr>
  <h:td>Apples</h:td>
  <h:td>Bananas</h:td>
  </h:tr>
</h:table>
```

และ

```
<f:table xmlns:f="http://www.w3schools.com/furniture">
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถเพิ่ม attribute xmlns ลงในแท็ก เพื่อบอกชื่อที่ระบุไว้ ที่เกี่ยวข้องกับ namespace

- ***Attribute Namespace***

attribute namespace จะวางในแท็กเริ่มต้น (start tag) ตามไวยากรณ์ ดังนี้

```
Xmlns:namespace-prefix="namespace"
```

จากตัวอย่างข้างบน namespace จะสามารถกำหนดด้วย Internet address

```
Xmlns:f="http://www.w3schools.com/furniture">
```

จาก namespace specification ของ W3C กล่าวไว้ว่า namespace สามารถเป็น Uniform Resource Identifier (URI)

ถ้า namespace ถูกกำหนดในแท็กเริ่มต้นแล้ว child element ทุกตัวที่มี prefix เหมือนกันจะต้อง มีความสัมพันธ์กับ namespace เดียวกันด้วย

และ address ที่ใช้บอก namespace ใช้เพื่อให้ชื่อมีความเป็นเอกลักษณ์ไม่ได้ใช้เป็น address เพื่ออ้างอิงหาข้อมูล แต่หลายบริษัทใช้ namespace เพื่อเป็นตัวชี้ไปยัง web page ที่มีอยู่จริงซึ่งเก็บข้อมูลเกี่ยวกับ namespaceไว้

- ***Default Namespaces***

การกำหนด default namespace จะป้องกันการใช้ prefix ใน child element ทุกตัว มีไวยากรณ์ดังนี้

```
<element xmlns="namespace">
```

ซึ่งจากตัวอย่างที่ผ่านมาจะได้ดังนี้

```
<table xmlns="http://www.w3.org/TR/html4/">
  <tr>
  <td>Apples<td>
  <td>Bananas<td>
  <tr>
</table>
```

และ

```
<table xmlns="http://www.w3schools.com/furniture">
  <name>African Coffee Table</name>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  <width>80</width>
  <length>120</length>
</table>
```

4.1.8 XML PCDATA and CDATA

Parsable Character Data (PCDATA) คือข้อมูลแบบเท็กซ์ที่จะถูก parse โดย parser

Character Data (CDATA) คือข้อมูลแบบเท็กซ์ที่ไม่ถูก parse โดย parser

4.1.8.1 PCDATA

- **XML Parser จะมองทุกเท็กซ์เป็น Parsable Characters (PCDATA)**

เมื่อ XML ถูก parse เท็กซ์ที่อยู่ระหว่าง XML แท็กจะถูก parse ไปด้วย

```
<message>This text is also parsed</message>
```

parser ทำเช่นนี้เนื่องจาก XML element สามารถมีหลาย element อยู่ภายใน ดังในตัวอย่าง

```
<name><first>Bill</first><last>Gates</last></name>
```

และ parser จะแบ่งเป็น sub-element ดังนี้

```
<name>
  <first>Bill</first>
  <last>Gates</last>
</name>
```

- ***Escape Character***

XML character ที่ไม่สามารถใช้ได้จะถูกแทนที่ด้วย entity reference

ถ้าใส่ตัวอักษร "<" ใน XML element จะทำให้เกิดความผิดพลาด เพราะ parser จะแปลเป็นจุดเริ่มต้นของ element ใหม่ ดังนั้นจึงไม่สามารถเขียน อย่างนี้ได้

```
<message>if salary < 1000 then</message>
```

การหลีกเลี่ยง สามารถทำได้โดย แทนที่ "<" ด้วย entity reference ดังนี้

```
<message>if salary &lt; 1000 then</message>
```

entity reference มาตรฐานที่ได้กำหนดมาให้ก่อนแล้วใน XML มีทั้งหมด 5 ตัว ดังตารางที่ 4-3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Entity Reference	ค่า	ความหมาย
<	<	less than
>	>	greater than
&	&	ampersand
'	'	apostrophe
"	"	quotation mark

ตารางที่ 4-3 Entity Reference มาตรฐาน

entity reference จะขึ้นต้นด้วย & และลงท้ายด้วย ;

เฉพาะ “<” และ “&” เท่านั้นที่ไม่สามารถใช้ได้ใน XML ส่วนตัวที่เหลือควรทำให้เป็นนิสัย

4.1.8.2 CDATA

parser จะไม่ parse ทุกอย่างที่อยู่ในส่วนของ CDATA

ถ้าเท็กซ์มี “<” หรือ “&” อยู่ภายใน XML element นั้นจะถูกกำหนดเป็นส่วนของ CDATA ส่วนของ CDATA จะเริ่มต้นด้วย “<![CDATA[“ และลงท้ายด้วย “]]>”

```
<script>
<![CDATA[
function matchwo(a,b)
{
if (a < b && a < 0) then
  {
  return 1
  }
else
  {
  return 0
  }
}
]]>
</script>
```

จากตัวอย่างข้างต้น ทุกอย่างที่อยู่ในส่วนของ CDATA จะถูกเพิกเฉยโดย parser

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.9 XML Parser

XML parser คือโปรแกรมที่ใช้ในการอ่าน สร้าง และจัดการกับเอกสาร XML รวมถึงตรวจความถูกต้องของเอกสาร XML parser สามารถเขียนได้ด้วยภาษาใด ๆ ก็ได้ ในบราวเซอร์ IE 5.0 ขึ้นไปก็มี Microsoft XML parser ทำให้ IE สามารถแสดงผลเอกสาร XML ได้ XML parser ของภาษาจาวา อย่างเช่น XML parser ของซัน (http://java.sun.com/xml) หากเราต้องการสร้างแอพพลิเคชันที่สามารถจัดการกับ XML เราก็ต้องมี parser ของภาษานั้นและเขียนโค้ดติดต่อกับ API ของมัน

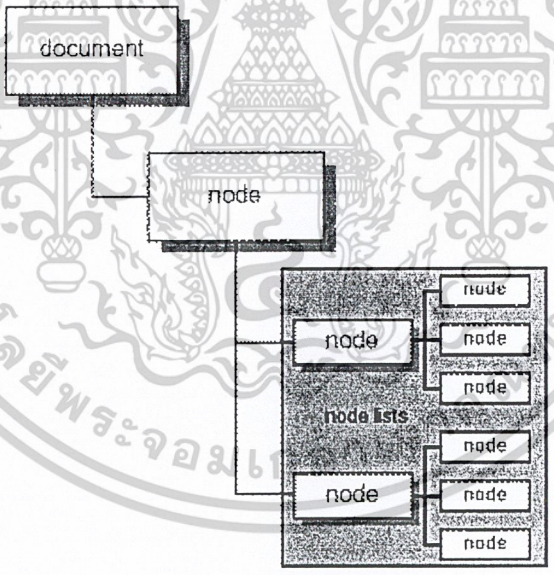
XML parser มี 2 ชนิดคือ

- DOM (Document Object Model)
- SAX (Simple API for XML)

4.1.9.1 DOM

DOM จะใช้วิธี parse เอกสาร XML เป็นลำดับชั้น (hierachy) ของออบเจ็กต์ดังนี้ (ดังรูปที่ 4-1)

1) Document Object – แหล่งข้อมูล XML
2) Node Object – Parent node ของ child node
3) Node List Object – list ของ sibling node
4) Parse Error – ใช้ในการรับค่าผิดพลาดที่เกิดจากการ parse



รูปที่ 4-1 โครงสร้างของ DOM

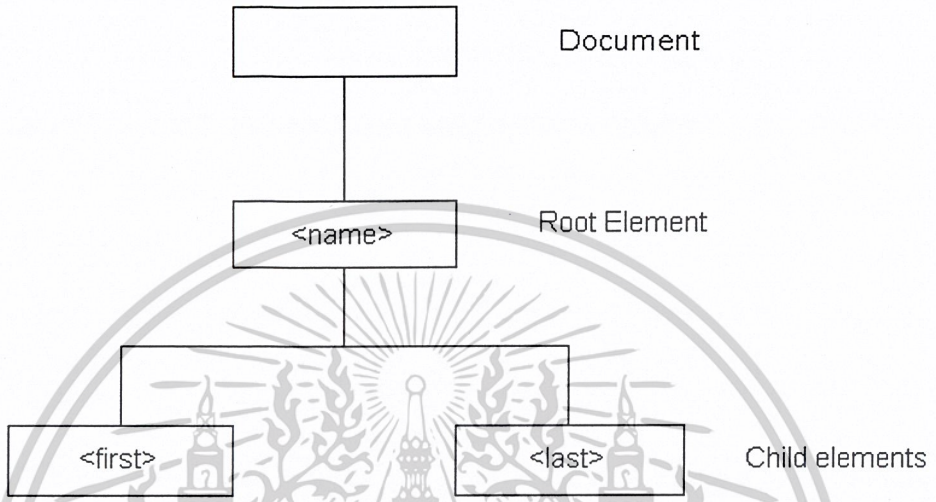
หากเอกสาร XML เป็นดังนี้

```
<name>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    <first>Jim(/first>
    <last>Jones</last>
</name>
```

จะได้ความสัมพันธ์ดังรูปที่ 4-2



รูปที่ 4-2 ความสัมพันธ์ของการ parse ด้วย DOM

4.1.9.2 SAX

SAX (Simple API for XML) วิธีนี้จะเป็นแบบ event-driven โดย event จะประกอบด้วย

1) Start Element – จะตอบสนองกับ event นี้เมื่อพบจุดเริ่มต้นของแท็ก XML
2) End element – จะตอบสนองกับ event นี้เมื่อพบจุดสิ้นสุดของแท็ก
3) Character – จะตอบสนองกับ event นี้เมื่อพบกับตัวอักษร
4) Start Document – จะตอบสนองกับ event นี้ในตอนเริ่ม parse
5) End Document - จะตอบสนองกับ event นี้ในเมื่อจบการ parse

ดังนั้นจากโค้ดข้างบนจะเกิด event ดังนี้

```
                                    →  Start Document
<name>                              →  Start Element name
    <first>                         →  Start Element first
        Jim                         →  Character Jim
    </first>                        →  End Element first
    <last>                          →  Start Element last
        Jones                       →  Character Jones
    </last>                         →  End Element last
</name>                             →  Start Element name
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

→ End Document

ทั้ง 2 วิธีมีข้อดีข้อเสียต่างกันคือ DOM จะใช้เมโมรีและมีการประมวลผลมาก แต่สามารถจัดการได้ง่าย สามารถที่จะแก้ไขค่าได้ ไม่เหมาะกับเอกสาร XML ที่มีขนาดใหญ่ ถูกนำไปใช้ในบราวเซอร์ ในขณะที่ SAX ไม่ต้อง parse เอกสาร XML ทั้งหมดจึงใช้โมเมรีและมีการประมวลผลน้อยกว่า เหมาะกับการสร้างและรับค่าจากเอกสาร จะถูกใช้ในการประมวลผลทางฝั่งเซิร์ฟเวอร์ ตัวอย่าง XML parser ดังตารางที่ 4-4

ชื่อ	แพล็ตฟอร์ม	คำบรรยาย
ActiveDOM	Microsoft Window	http://www.vivid-creations.com
ActiveSAX	Microsoft Window	http://www.vivid-creations.com
JavaTM API for XML	Java	http://java.sun.com/xml
MS XML	Microsoft Window	http://www.microsoft.com
Oracle XML Developer's Kit	Java , C++ , PL/SQL	http://technet.oracle.com/tech/xml
Xerces	Java , C++	http://xml.apache.org
XP	Java	http://www.jclark.com/xml

ตารางที่ 4-4 XML Parser

4.1.10 ประโยชน์จาก XML

สำหรับประโยชน์ของ XML นั้น เป็นด้านความยืดหยุ่นในการใช้งานสำหรับแอพพลิเคชันที่อิงกับ Web Base ที่ใช้ง่ายในการค้นหาข้อมูล มีความยืดหยุ่นในการพัฒนาเว็บ สามารถผสมผสานข้อมูลจากหลายแหล่ง จากแอพพลิเคชันที่ต่างกัน สามารถแสดงข้อมูลแบบต่างๆ และสามารถ update ข้อมูลให้ทันสมัยเสมอ และคาดว่าจะเป็นมาตรฐานใหม่ของระบบเปิด ซึ่งนับเป็น format ใหม่สำหรับการส่งข้อมูลบนเว็บที่มากด้วยข้อมูลหลายแบบ แต่ส่งผ่านด้วยเทคโนโลยีที่บีบอัดข้อมูลที่ให้ความเร็วได้รับการสนับสนุนจากผลิตภัณฑ์ค่ายไมโครซอฟท์

สถาปัตยกรรม XML

ภาษา XML ได้รับการสนับสนุนจาก W3C ให้นักพัฒนาเว็บแอพพลิเคชันได้หันมาใช้เป็นส่วนประกอบของการพัฒนาเว็บไซต์ เพราะ XML มีประสิทธิภาพและมีความน่าเชื่อสูงในการแปลงข้อมูลและโครงสร้างข้อมูลให้สามารถนำไปใช้งาน สถาปัตยกรรม 3 Tier ที่ XML สามารถสร้างขึ้นจากระบบข้อมูลที่ใช้โมเดลของ 3-Tier โครงสร้าง ของข้อมูลต่างๆ สามารถนำมาแสดงตามข้อกำหนด หรือรูปแบบที่ต้องการตามการใช้งานได้

XML เป็นการทำงานในระดับกลาง middle tier ที่สามารถเรียกใช้ฐานข้อมูลได้หลากหลายระบบฐานข้อมูลและโอนข้อมูลให้อยู่ในรูปแบบของ XML และมีการให้รายละเอียดเกี่ยวกับตัวข้อมูลโครงสร้างต่างๆ ของระบบฐานข้อมูลได้ XML เป็นระบบเปิดที่นำเสนอข้อมูลในรูปแบบ text ผ่านทาง HTTP เหมือนกับ HTML แต่จะมีคุณสมบัติในการให้ข้อมูลแบบ real time อัพเดทหรือเปลี่ยนแปลงได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตามข้อกำหนด การแสดงข้อมูลจาก XML ใน HTML จะเป็นการเพิ่มในส่วนของรายละเอียดข้อมูล ที่มีการเรียกใช้จากแหล่งหรือฐานข้อมูลที่เชื่อมโยงกันในหลายแหล่งเพื่อให้HTMLมีความสมบูรณ์มากขึ้น ในอนาคตการพัฒนาเว็บหรือการเขียน และสร้าง HTML ไม่จำเป็นต้องมีการเขียนชุดคำสั่งที่ยุ่งยากซับซ้อนมากก็สามารถทำงานร่วมกับระบบข้อมูลได้อย่างมีประสิทธิภาพ XML จะทำการกำหนดค่าสำหรับโครงสร้างข้อมูลที่จะนำไปแสดงใน HTML นอกจากนั้นยังสามารถนำไปสนับสนุนระบบการแลกเปลี่ยนข้อมูลข่าวสารทาง Electronic ได้อย่างดีอีกด้วย

4.1.11 บทสรุปของ XML

XML มีความยืดหยุ่นพอในการนำเสนอข้อมูลแบบต่างๆ ที่สามารถให้รายละเอียดโครงสร้างข้อมูลตามระดับและความต้องการในการนำไปใช้งาน

4.2 SOAP

4.2.1 แนะนำ SOAP

SOAP (Simple Object Access Protocol) เป็นโพรโตคอลที่ใช้ XML เป็นพื้นฐาน เพื่อให้ซอฟต์แวร์คอมโพเนนต์ และแอพพลิเคชันสามารถติดต่อกันผ่าน HTTP ซึ่งเป็นมาตรฐานอินเตอร์เน็ตโพรโตคอลได้

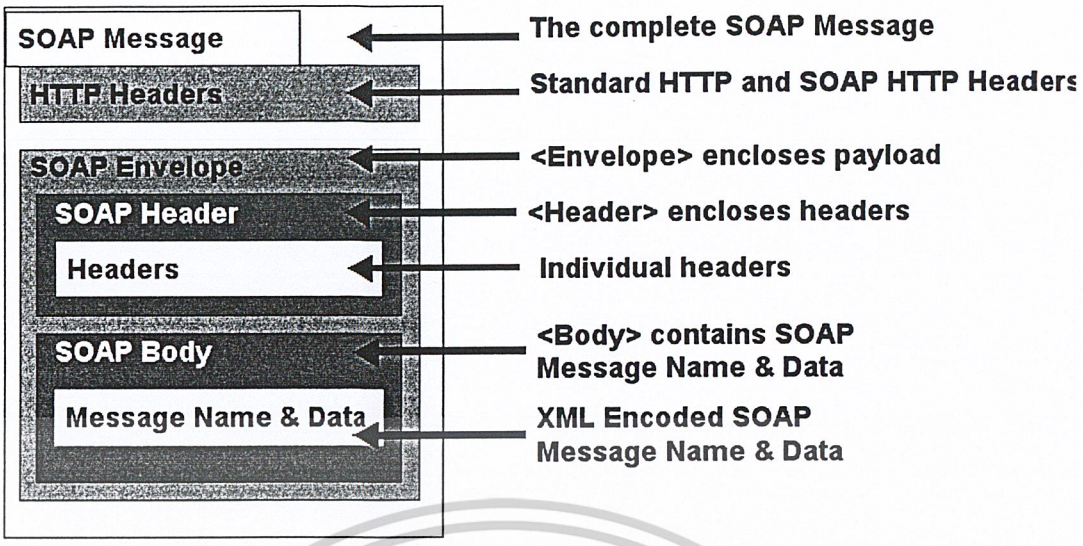
เนื่องจากการสื่อสารระหว่างแอพพลิเคชันจำเป็นต่อการพัฒนาแอพพลิเคชันบนอินเตอร์เน็ต เป็นอย่างมาก แต่แอพพลิเคชันแบบกระจายกันทำงาน (distributed application) ในปัจจุบันใช้ Remote Procedure Call (RPC) สื่อสารกันระหว่างออบเจ็กต์ เช่น DCOM และ CORBA ซึ่งไม่ได้ใช้พอร์ตเดียวกับ HTTP ที่เป็นพอร์ตมาตรฐานสำหรับให้บริการเว็บ ดังนั้น RPC จึงนำมาปรับใช้กับอินเตอร์เน็ตได้ยาก และมีปัญหาทางด้านความปลอดภัย ไฟร์วอลล์และพร็อกซีเซิร์ฟเวอร์จะไม่ยอมให้ส่งข้อมูลชนิดนี้ได้ตามปกติ วิธีที่ดีกว่าคือใช้ HTTP เพราะเป็นที่ยอมรับโดยอินเตอร์เน็ตบราวเซอร์ และเซิร์ฟเวอร์ทุกชนิด ซึ่ง SOAP ถูกสร้างมาเพื่อใช้ในกรณีนี้

ข้อดีของ SOAP ก็คือ SOAP ไม่ขึ้นกับคอมโพเนนต์เทคโนโลยี และภาษาการเขียนโปรแกรมใดๆ สามารถเขียนได้ง่ายและขยายเพิ่มเติมได้

4.2.2 ส่วนประกอบของ SOAP

SOAP เมสเซจใช้ไวยากรณ์ของ XML ในการสร้าง ประกอบด้วย 3 อีลีเมนต์มาตรฐาน คือ SOAP Envelope, SOAP Header และ SOAP Body

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4-3 โครงสร้างของ SOAP

SOAP เมสเซจ จะต้องเป็นไปตามกฎนี้

- Envelope เป็นอีลีเมนต์ที่อยู่บนสุด ต้องมีอีลีเมนต์นี้เสมอ
- Header อาจจะมีหรือไม่มีก็ได้ แต่ถ้ามีต้องเป็น child element แรกของ Envelope
- Body ต้องเป็น child element แรกของ envelope หรือ header

ตัวอย่างของ SOAP เมสเซจ ตัวอย่างนี้คือ GetLastTradePrice SOAP Request ที่ส่งไปยัง StockQuote service ประกอบด้วยสตริงพารามิเตอร์ symbol และคืนค่า float กลับมากับ SOAP response SOAP envelope element อยู่ที่จุดบนสุดของเอกสาร XML XML เนมสเปซ ใช้เพื่อป้องกันการสับสนของ SOAP identifier

SOAP request จะเป็นดังนี้

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"

<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    <m:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP response จะเป็นดังนี้

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  <SOAP-ENV:Body>
    <m:GetLastTradePriceResponse xmlns:m="Some-URI">
      <Price>34.5</Price>
    </m:GetLastTradePriceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

นอกจากนั้นใน SOAP เมสเซจ จะมีการใช้ XML เนมสเปซ ทุกๆ อีลีเมนต์ในเอกสารจะขึ้นต้นด้วย เนมสเปซ เนมสเปซ จะถูกกำหนดโดยใช้ xmlns attribute ดังนี้

```
<SOAP-ENV:Envelope
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
```

แต่หากไม่ต้องการให้ขึ้นต้นด้วย เนมสเปซ ก็สามารถทำได้ ดังนี้

```
<Envelope
        xmlns="http://schemas.xmlsoap.org/soap/envelope/">
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2.1 Envelope

envelope element เป็นอีลีเมนต์บนสุดของเอกสาร XML ที่ใช้แสดงเมสเซจ อาจประกอบด้วยแอตทริบิวต์ คือ envelope เนมสเปซ และ encodingStyle

1. Envelope เนมสเปซ – SOAP เมสเซจจะแสดงเวอร์ชันโดยใช้ เนมสเปซ ของ envelope element เนมสเปซ จะถูกนำไปใช้ขึ้นต้น envelope, header และ body element
2. EncodingStyle Attribute – ใช้แสดงวิธี serialization ของ SOAP เมสเซจ แอตทริบิวต์นี้สามารถมีได้ในทุกอีลีเมนต์และจะมีผลกับ content ของอีลีเมนต์นั้นและ child element ทั้งหมดของมันที่ไม่ได้ประกาศแอตทริบิวต์

4.2.2.2 Body

เป็นส่วนของข้อมูลที่ใช้ในการแลกเปลี่ยน โดยทั่ว ๆ ไปข้อมูลจะเป็นการ marshall RPC call และ error report child element ของ body element จะถูกเรียกว่า Body entry

body entry จะต้องเป็นไปตามกฎนี้

- body entry จะต้องแสดงด้วยชื่อเต็มที่ประกอบด้วย เนมสเปซ URI และ ชื่อของมัน (local name)
- SOAP encodingStyle attribute อาจจะมีได้ เพื่อแสดงถึงวิธี encode ของ body entry นั้น

4.2.2.3 Header

เป็นส่วนเพิ่มเติมของเมสเซจสามารถประกอบด้วยอินฟอร์เมชันที่ระบุไปยังแอพพลิเคชัน ส่วนเพิ่มเติมนี้อาจนำไปใช้อิมพลีเม็นต์เป็น authentication, transaction management เป็นต้น ทุก ๆ child element ของ header element จะถูกเรียกว่า Header entry

header entry จะต้องเป็นไปตามกฎดังนี้

- header entry จะต้องแสดงด้วยชื่อเต็มที่ประกอบด้วย เนมสเปซ URI และ ชื่อของมัน (local name)
- อาจมี SOAP encodingStyle attribute ที่ใช้สำหรับ header entry
- SOAP mustUnderStand attribute และ SOAP actor attribute จะมีหรือไม่ก็ได้ เพื่อใช้บอกว่าจะจัดการกับ entry นั้นอย่างไรและโดยใคร

```
<soap:Header>
   <m:local xmlns:m="http://www.w3schools.com/local/"
                  soap:actor="http://www.w3schools.com/appml" />
      <m:language>en</m:language>
      <m:currency>USD</m:currency>
    </m:local>
</soap:Header>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1) SOAP Actor Attribute – SOAP เมสเซจสามารถถูกส่งไปยังปลายทางผ่านตัวกลางของ SOAP (SOAP intermediary) ตัวกลางของ SOAP คือแอพพลิเคชันที่มีความสามารถทั้งรับและส่งต่อ SOAP เมสเซจ ในการส่งต่อตัวกลางจะต้องไม่ส่งต่อ header element ไปให้กับแอพพลิเคชัน โดยตัวกลางนี้จะถูกกำหนดเป็น URI ถ้าไม่มี attribute นี้หมายถึงผู้รับ SOAP เมสเซจเป็นปลายทางสุดท้าย
2) SOAP MustUnderStand Attribute – ใช้แสดงว่า header entry นี้จำเป็นหรือเป็นเพียงอ็อปชันสำหรับผู้รับที่จะนำไปประมวลผล ค่าของมันคือ “1” หรือ “0” ถ้าไม่มีการระบุจะมีค่าเท่ากับ “0” โดย “0” หมายถึงเป็นอ็อปชัน และ “1” หมายถึงจำเป็น โดยจะต้องประมวลผลให้ถูกต้องตาม semantic หรือต้อง fail เช่น เมสเซจเป็นส่วนหนึ่งของ transaction ถ้าปลายทางไม่รองรับ transaction จะต้องไม่ประมวลผลและส่งผลผิดพลาดกลับไปในกรณีที่ mustUnderStand attribute เป็น “1” แต่ถ้าเป็น “0” จะยังคงประมวลผลต่อไป

4.2.3 SOAP Fault Element

ข้อความแสดงความผิดพลาดจากแอพพลิเคชันของ SOAP จะเก็บอยู่ใน fault element ซึ่งถ้ามีจะต้องปรากฏใน body element เพียงครั้งเดียวใน SOAP เมสเซจ ตัวอย่างอาจเป็นดังนี้

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
        soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:MustUnderstand</faultcode>
      <faultstring>Mandatory Header error.</faultstring>
      <faultactor>http://www.wrox.com/heroes/endpoint.asp</faultactor>
      <detail>
        <w:source xmlns:w="http://www.wrox.com/">
          <module>endpoint.asp</module>
          <line>203</line>
        </w:source>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

SOAP fault element มี element ย่อย ๆ ดังตารางที่ 4-5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Sub Element	Description
<faultcode>	โค้ดที่ระบุถึงการ error
<faultstring>	ข้อความการ error
<faultactor>	ใครเป็นสาเหตุของการ error
<detail>	ระบุอินฟอร์เมชันของการ error

ตารางที่ 4-5 SOAP Fault Element

ค่าของ faultcode สามารถมีค่าได้ดังตารางที่ 4-6

Error	Description
VersionMismatch	เนมสเปซ ภายใน SOAP Envelope element ไม่ถูกต้อง
MustUnderstand	child element ของ Header element กับ mustUnderstand attribute ที่มีค่า "1" ผู้รับไม่รองรับ
Client	เมสเซจมีรูปแบบไม่ถูกต้อง หรือมีอินฟอร์เมชันที่ไม่ถูกต้อง
Server	เกิดปัญหากับเซิร์ฟเวอร์ ไม่สามารถประมวลผลได้

ตารางที่ 4-6 ค่า faultcode

4.2.4 SOAP Encoding

SOAP encoding มีวิธีการ map จาก type ของโปรแกรมมิ่งไปเป็น XML 2 วิธี คือจากภายนอกโดยใช้ WSDL (Web Services Description Language) ที่บอกถึง type ของข้อมูลที่รับหรือส่ง หรือใช้ xsi:type attribute ในกรณีที่ภาษาที่ใช้ไม่รองรับ WSDL โดยทั้ง 2 วิธีจะใช้ XML schema ในการระบุ type ตัวอย่างของการใช้ xsi จะเป็นดังนี้

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
        soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <m:MixedMessage xmlns:m="http://www.wrox.com/mix/">
      <param1 xsi:type="xsd:string">OU812</param1>
      <param2 xsi:type="xsd:integer">2001</param2>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    <param3 xsi:type="xsd:double">3.14159</param3>
  </m:MixedMessage>
 </soap:Body>
</soap:Envelope>
```

SOAP รองรับ type ของข้อมูลได้ทั้ง simple type และ compound type เช่น struct และ array

4.2.4.1 ตัวอย่าง Struct

struct book ที่เขียนด้วยภาษา c++ จะเป็นดังนี้

```
struct Book
{
   string author;
   string name;
   int page;
};
```

Soap เมสเซจที่ไม่ใช้ WSDL แต่ใช้ xsi จะเป็นดังนี้

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">
<SOAP-ENV:Body>
<ns1:searchBook xmlns:ns1="http://soapinterop.org/" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<bookResult xmlns:ns2="http://soapinterop.org/xsd" xsi:type="ns2:Book">
 <author xsi:type="xsd:string">Ed Roman</author>
 <name xsi:type="xsd:string">Mastering EJB</intro>
 <page xsi:type="xsd:int">500</page>
 </bookResult>
 </ns1:searchBook>
</SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

แต่ถ้าใช้ WSDL ในส่วนของการประกาศ schema จะต้องประกอบด้วย schema ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<types>
  <schema targetเนมสเปซ='http://soapinterop.org/xsd'
    xmlns='http://www.w3.org/2001/XMLSchema'
    xmlns:SOAP-ENC='http://schemas.xmlsoap.org/soap/encoding/'
    xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
    elementFormDefault='qualified'>
      <complexType name="Book">
        <sequence>
          <element name='author' type='string'/>
          <element name='name' type='string'/>
          <element name='page' type='int'/>
        </sequence>
      </complexType>
  </schema>
</types>
```

4.2.4.2 ตัวอย่าง Array

ตัวอย่างนี้เป็น array ของ struct book ในตัวอย่างก่อน Soap เมสเซจที่ไม่ใช้ WSDL แต่ใช้ xsi จะเป็นดังนี้

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/1999/XMLSchema">
<SOAP-ENV:Body>
    <ns1:BookArrayResult xmlns:ns1="http://soapinterop.org/" SOAP-
    ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <BookArray xmlns:ns2="http://schemas.xmlsoap.org/soap/encoding/"
        xsi:type="ns2:Array" xmlns:ns3="http://soapinterop.org/xsd"
        ns2:arrayType="ns3:Book[3]">
        <item xsi:type="ns3:Book">
          <author xsi:type="xsd:string">Ed Roman</author>
          <name xsi:type="xsd:string">Mastering EJB</name>
          <page xsi:type="xsd:int">500</page>
        </item>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            <item xsi:type="ns3:Book">
                <author xsi:type="xsd:string">Roger Session</author>
                <name xsi:type="xsd:string">The Battle of the  middleware</name>
                <page xsi:type="xsd:int">350</page>
            </item>
            <item xsi:type="ns3:Book">
                <author xsi:type="xsd:string">Chris Dix</author>
                <name xsi:type="xsd:string">Professional XML Web Service</name>
                <page xsi:type="xsd:int">550</page>
            </item>
        </BookArray>
    </ns1:BookArrayResult>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

แต่ถ้าใช้ WSDL ในส่วนของการประกาศ schema จะต้องประกอบด้วย schema ดังนี้

```
<types>
  <schema targetNamespace='http://soapinterop.org/xsd'
    xmlns='http://www.w3.org/2001/XMLSchema'
    xmlns:SOAP-ENC='http://schemas.xmlsoap.org/soap/encoding/'
    xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
    elementFormDefault='qualified'>
      <complexType name="Book">
        <sequence>
          <element name='author' type='string'/>
          <element name='name' type='string'/>
          <element name='page' type='int'/>
        </sequence>
      </complexType>
      <complexType  name ='ArrayOfBook'>
      <complexContent>
        <restriction base='SOAP-ENC:Array'>
          <attribute ref='SOAP-ENC:arrayType' wsdl:arrayType='typens:Book[]'/>
        </restriction>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    </complexContent>
   </complexType>
  </schema>
</types>
```

4.2.5 SOAP ใน HTTP

ในการส่ง SOAP ผ่านทาง HTTP จะต้องใช้ content-type เป็น text/xml แต่ใน SOAP request จะต้องมี header SOAPAction ภายใน HTTP header SOAPAction จะเป็นตัวบอกให้เซิร์ฟเวอร์รู้ว่า HTTP Post นั้นเป็น SOAP เมสเซจ และค่าของ header คือ URI ที่แสดงถึงจุดหมายของ SOAP เมสเซจ ส่วน SOAP response จะต้องมี status code ตามมาตรฐานของ HTTP โดย 200-299 แสดงว่าสำเร็จ แต่ถ้า response เมสเซจ เป็นการ fault แล้ว status code จะต้องเป็น 500 ซึ่งแสดงถึง internal server error ตัวอย่างของ response ที่มี status code เป็น 500 อาจเป็นดังนี้

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml
Content-Length: ###
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
        soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:VersionMismatch</faultcode>
        <faultstring>The SOAP เนมสเปซ is incorrect.</faultstring>
      <faultactor>http://www.wrox.com/endpoint.asp</faultactor>
      <detail>
        <w:errorinfo xmlns:w="http://www.wrox.com/">
          <desc>The SOAP เนมสเปซ was blank.</desc>
        </w:errorinfo>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.6 SOAP สำหรับ RPC

จุดประสงค์ของการออกแบบ SOAP คือทำ RPC โดยใช้ XML การเรียก RPC นั้นจะ map เข้ากับ HTTP request และ RPC response จะ map กับ HTTP response

ในการทำ mechod call จำเป็นที่จะต้องมีอินฟอร์เมชันดังนี้

1. URI ของออบเจ็กต์ปลายทาง (target object)
2. ชื่อของเมธอด
3. คำอธิบายของเมธอด (method signature) ส่วนนี้เป็นอ็อปชัน
4. พารามิเตอร์สำหรับเมธอด
5. ข้อมูลส่วนหัว (header data) เป็นอ็อปชัน

พิจารณาเมธอดดังนี้

```
double GetStockQuote ( [in] stirng sSymbol);
```

ถ้า เนมสเปซ ของเมธอดคือ "http://www.worxstox.com/" แล้วการเรียกเมธอดโดย request stock quote ใช้ symbol OU812 จะเป็นดังนี้

```
<q:GetStockQuote xmlns:q="http://www.wroxstox.com/">
  <q:sSymbol xsi:type="xsd:string">OU812</q:sSymbol>
</q:GetStockQuote>
```

ชื่อเมธอดและชื่อของอีลีเมนต์จะต้อง match กันเช่นเดียวกับพารามิเตอร์

สำหรับ response จะต้องชื่ออีลีเมนต์จะต้องเป็นชื่อเมธอดตามด้วย response ดังนี้

```
<q:GetStockQuoteResponse xmlns:q="http://www.wroxstox.com/"
   <q:ret xsi:type="xsd:double">100.0</q:ret>
</q:GetStockQuoteResponse>
```

ชื่ออีลีเมนต์ของค่าที่คืนมาสามารถเปลี่ยนได้ โดยจะกำหนดได้จากโปรแกรมที่เขียนหรือจาก WSDL

4.2.7 SOAP Toolkit

SOAP toolkit คือเครื่องมือที่จะทำหน้าที่ในการประมวลผล SOAP request และส่ง response กลับไปให้ไคลเอนต์ โดย SOAP toolkit แต่ละตัวใช้ได้บนแพลตฟอร์มที่ต่างกัน รองรับเทคโนโลยี เว็บเซอร์วิส ต่างกัน บางตัวอาจจะรองรับ WSDL หรือ UDDI ในขณะที่บางตัวรองรับอย่างใดอย่างหนึ่งหรือไม่รองรับทั้งสองอย่าง และถึงแม้ว่า SOAP จะเป็นอิสระต่อแพลตฟอร์ม แต่ใน SOAP toolkit บางตัวยังไม่สามารถทำงานข้ามผลิตภัณฑ์ (interoperability) ได้ เนื่องจากรองรับเทคโนโลยีของ เว็บเซอร์วิส ไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เหมือนกัน เช่น xsi ในบางผลิตภัณฑ์จะบังคับให้ SOAP เมสเซจจะต้องระบุ type ด้วย xsi หากไม่มีก็จะ falut ซึ่งในปัจจุบันแต่ละผลิตภัณฑ์ก็พยายามพัฒนาให้สามารถทำงานด้วยกันได้

ชื่อ	แพลตฟอร์ม	แหล่งที่มา
4s4c	COM	http://www.4s4c.com
A SOAP for RPC NT Service	COM	http://www.whitemesa.com
Apache Axis	Java	http://xml.apache.org/axis
Apache SOAP	Java	http://xml.apache.org/soap
CapeConnect	Java	http://www.capeclear.com
Glue	Java	http://www.themindelectric.com
IBM Web Services Toolkit	Java	http://www.alphaworks.ibm.com
IdooXoap for Java and C++	Java , C++	http://www.zvon.org
Microsoft SOAP Toolkit	COM	http://www.microsoft.com
PocketSOAP	COM	http://www.pocketsoap.com
SOAP::Lite	Perl	http://www.soaplite.com
Visual Studio .NET	COM	http://www.microsoft.com

ตารางที่ 4-7 SOAP Toolkit

4.2.8 โครงสร้างภายในของ SOAP Toolkit ของ Microsoft

The Web Services Description Language (WSDL) เป็นรูปแบบหนึ่งของ XML เพื่อใช้อธิบายว่าเซิร์ฟเวอร์นั้นมีเซอร์วิส (Services) และ operation ภายในเซอร์วิสอะไรให้ใช้บ้าง ซึ่ง operation ภายในเซอร์วิสนั้นจะอธิบายรูปแบบที่ไคลเอนต์ (client) ต้องใช้ในการเรียก operation

The Web Services Meta Language (WSML) file เป็นส่วนที่เพิ่มขึ้นมาจากไฟล์ WSDL เพื่อใช้ในการเชื่อมโยงแต่ละ operation ในเซอร์วิสกับ COM object โดย WSML file จะพิจารณาว่า COM object ตัวใดที่จะต้องถูกโหลด (load) ขึ้นมาเมื่อมีการเรียกใช้ operation ในเซอร์วิส

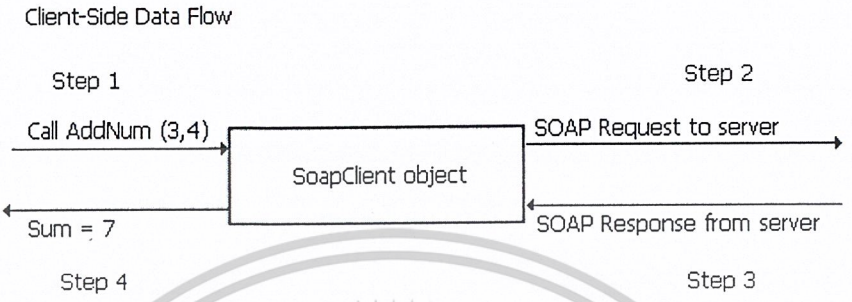
ทั้งไฟล์ WSDL และ WSML จะต้องอยู่ที่เซิร์ฟเวอร์ โดยไคลเอนต์ที่ต้องการที่จะส่ง SOAP request ไปที่เซิร์ฟเวอร์ จะต้องได้รับ WSDL จากเซิร์ฟเวอร์ก่อน เพื่อหารู้รูปแบบของ SOAP request ในการติดต่อ

4.2.8.1 การสร้าง SoapClient Object

ทางฝั่งไคลเอนต์ เมื่อแอพพลิเคชั่นส่ง message ไปที่ Soap Client object เพื่อเรียกใช้ operation หนึ่งๆ นั้น(ตามรูปนี้คือการบวกเลข 2 ตัว) SoapClient object จะทำการสร้าง SOAP request ส่งไปที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เซิร์ฟเวอร์ เมื่อเซิร์ฟเวอร์ทำการประมวลผลเสร็จเรียบร้อยแล้ว ก็จะส่งผลลัพธ์ในรูปแบบของ SOAP response กลับมาให้ไคลเอนต์ หลังจากนั้น SoapClient object จึงจะทำการแปลและส่ง message ที่เก็บผลลัพธ์ของ operation นั้นกลับไปให้แอพพลิเคชั่น ตามไดอะแกรมด้านล่าง



รูปที่ 4-4 การ Client –Side Data Flow

แอพพลิเคชั่นต้อง instantiate Soap object ก่อนที่จะทำการเรียกใช้ operation ใดๆ (SoapClient object นี้ถูกสร้างเก็บไว้ในไฟล์ MSSOAP1.dll) โดยใช้เมธอด SoapClient.mssoapinit (ชื่อไฟล์ WSDL, ชื่อเซอร์วิส, ข้อมูลพอร์ต) โดยข้อมูลพอร์ตจะใช้ในการแบ่งแยกช่องทางที่ใช้ในการส่ง SOAP message

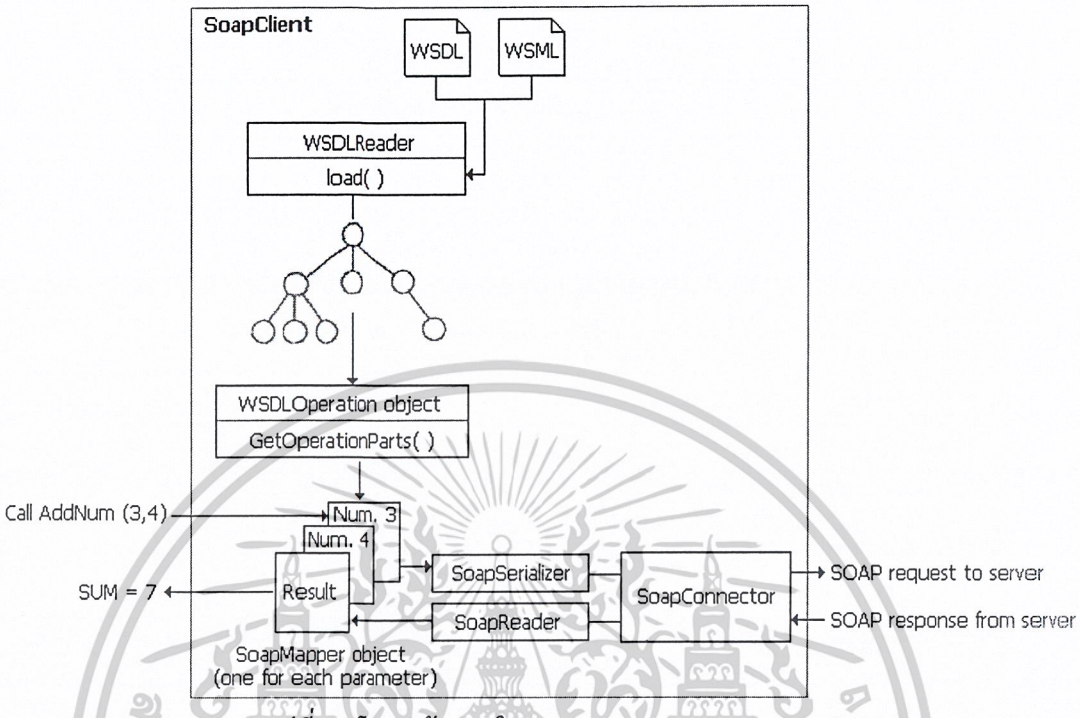
ในระหว่าง initialize SoapClient object จะตรวจสอบเมธอดทั้งหมดที่อยู่ในไฟล์ WSDL ด้วยการทำงานแบบไดนามิคนี้ทำให้เราสามารถที่จะเรียกใช้เมธอดใดก็ได้ที่ได้อธิบายไว้ในไฟล์ WSDL ดังแสดงตาม Microsoft Visual Basic Scripting Edition (VBScript) code ด้านล่างนี้

```
SET soapclient = CreateObject("MSSOAP.SoapClient")
Call soapclient.mssoapinit("http://localhost/MyWSDL.wsdl", "TestService", "TestPort")
' Now the client can call an operation listed in the portType element
' specified when calling mssoapinit().
wscript.echo soapclient.AddNumbers(2, 3)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.8.2 การประมวลผลภายใน SoapClient Object

การทำงานภายใน SoapClient object แสดงได้ดังรูปที่ 4-5



รูปที่4-5 โครงสร้างภายในของ SOAP Client

WSDL Reader object จะทำการโหลดไฟล์ WSDL และไฟล์ WSML เข้าไปไว้ใน DOM แล้วทำการวิเคราะห์ โดย WSDLReader object จะสร้าง WSDLOperation object เพื่อทำการเรียกใช้งาน operation (ตามรูปนี้คือ AddNum(3, 4)) หลังจากนั้น WSDLOperation จะเรียกเมธอด GetOperationParts เพื่อดึงรูปแบบ message ของอินพุทและเอาท์พุทของ operation ออกมา แล้ว SoapClient จะสร้าง Soap Mapper objects ให้กับแต่ละส่วนที่ได้จากเมธอด GetOperationParts และโหลดค่าที่ได้มาจากการเรียกใช้ operation เข้าไว้ใน objects เหล่านี้ พอถึงขั้นตอนนี้ SoapSerializer objects จะสร้าง SOAP request message จาก SoapMapper objects ที่เหมาะสมและส่งไปที่เซิร์ฟเวอร์

เมื่อเซิร์ฟเวอร์ประมวลผลเสร็จแล้วก็จะส่ง SOAP response กลับไปที่ไคลเอนต์ SoapClient object จะทำการโหลดผลลัพธ์ของ operation ไปเก็บใน SoapMapper object ที่เหมาะสมแล้วส่งผลลัพธ์กลับไปให้แอพพลิเคชั่นของผู้ใช้

4.2.8.3 การสร้าง SoapServer Object

ส่วนทางฝั่งเซิร์ฟเวอร์ เราสามารถกำหนดรูปแบบการรับ SOAP request (SOAP listener) ได้ 2 รูปแบบ คือ Internet Server API (ISAPI) server และActive Server Pages (ASP) server

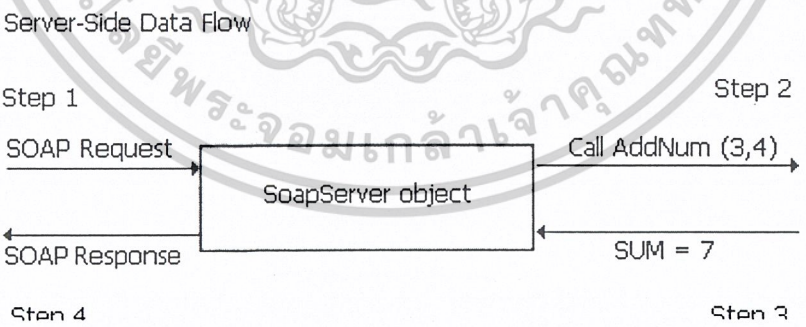
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในไฟล์ WSDL จะมี URL ที่ระบุว่าจะใช้ไฟล์ใดบนเซิร์ฟเวอร์ในการรับ SOAP request เพื่อใช้ในการพิจารณาว่าจะรับ SOAP request อย่างไรบนเซิร์ฟเวอร์ ตามโค้ดด้านล่างนี้ ตัวพิมพ์หนาจะเป็นส่วนที่ใช้ในการเรียกไปที่ ISAPI listener

```
<definitions>
...
 <service name='DocSample1' >
  <port name='DocSample1PortType' binding='tns:DocSample1Binding' >
   <soap:address
     location='http://localhost/DocSample1Test/DocSample1.wsdl' />
  </port>
 </service>
...
</definitions>
```

การทำงานทางฝั่งเซิร์ฟเวอร์

ไม่ว่า SOAP request จะเรียกไปยัง ISAPI หรือ ASP listener ก็ตาม ข้อมูลที่รับส่งก็ยังเหมือนกัน เมื่อ SoapServer object ได้รับ SOAP request จากไคลเอนต์ ก็จะทำการแปลและเรียกไปยังเมธอด COM ที่ตรงกับ operation ที่ต้องการ (ตามไดอะแกรมนี้คือเมธอด AddNum) หลังจากนั้นเมธอดที่ถูกเรียกใช้งานก็จะส่งผลลัพธ์กลับมาให้ SoapServer object ซึ่งมันก็จะแปลงผลลัพธ์ให้อยู่ในรูปของ SOAP response แล้วจึงส่งกลับไปที่ไคลเอนต์

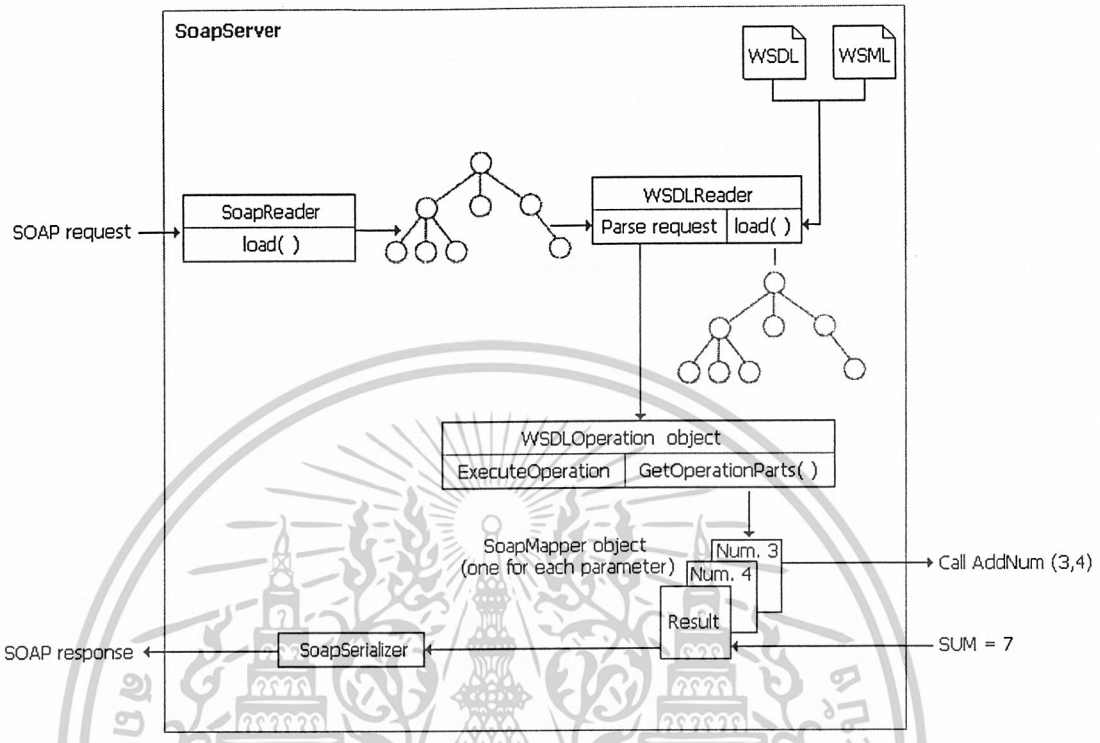


รูปที่ 4-6 Server-Side Data Flow

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.8.4 การประมวลผลภายใน SoapServer Object

การทำงานภายใน SoapServer object แสดงได้ดังรูปด้านล่าง



รูปที่ 4-7 โครงสร้างภายใน SOAP Server

เมื่อได้รับ SOAP request จากไคลเอนต์แล้ว SoapReader object ก็จะโหลด request message ไปไว้ใน DOM structure หนึ่ง ในขณะที่ WSDLReader จะโหลดไฟล์ WSDL และไฟล์ WSML ไปไว้ในอีก DOM structure หนึ่ง ตัวWSDLReader จะแปล request และสร้าง WSDLOperation object ขึ้นเพื่อใช้ใน requested operation หลังจากนั้น WSDLOperation object จะเรียกใช้เมธอด GetOperationParts เพื่อดึงรูปแบบ message ของอินพุทและเอาท์พุทของ operation ออกมา แล้ว SoapServer จะสร้าง Soap Mapper objects ให้กับแต่ละส่วนที่ได้จากเมธอด GetOperationParts และโหลดค่าที่ได้มาจากการเรียกใช้ operation เข้าไว้ใน objects เหล่านี้ พอถึงขั้นตอนนี้ SoapServer object จะทำการเรียกไปที่ COM method ที่ตรงกับที่ร้องขอเข้ามา

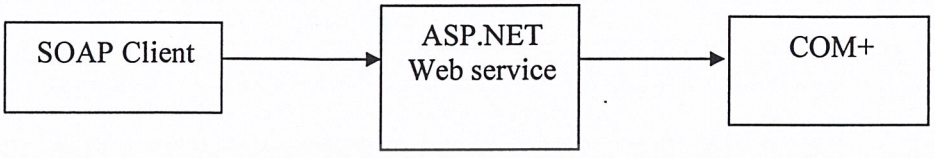
เมื่อ COM method ประมวลผลเสร็จแล้วก็จะส่งผลลัพธ์กลับไปที่ SoapServer object ซึ่ง SoapSErver object ก็จะทำการโหลดผลลัพธ์ของ operation ไปเก็บใน SoapMapper object ที่เหมาะสม แล้วจึงใช้ SoapSerializer object ส่ง SOAP response message กลับไปให้ไคลเอนต์แอพพลิเคชั่นของผู้ใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.9 การสร้าง SOAP Client และ SOAP Server ด้วย .NET Framework

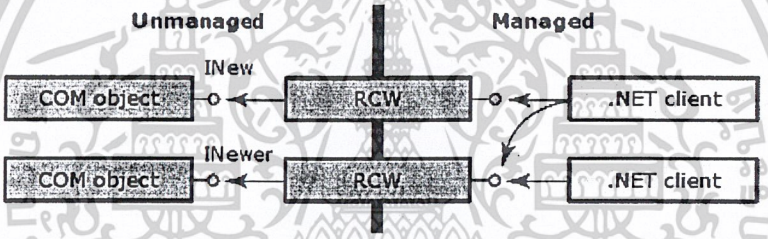
4.2.9.1 การใช้ ASP.NET ทำ SOAP เว็บเซอร์วิส

ใช้ความสามารถในการทำ COM+ Interop โดยสร้างคลาสของ ASP.NET มาเรียกใช้งาน business logic ที่สร้างด้วย Visual Basic 6



รูปที่ 4-8 การใช้ ASP.NET ทำ SOAP เว็บเซอร์วิส

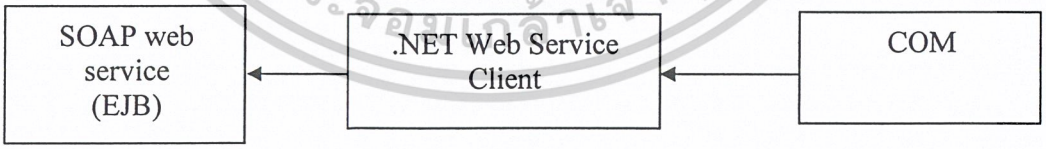
อาศัยความสามารถในการทำ marshall / unmarshall object ให้อยู่ในรูปแบบ SOAP ของ ASP.NET โดย ASP.NET ต้องทำ Interop โดยอาศัย Runtime Callable Wrapper ช่วยจำลอง COM object ให้ .NET มองเห็นเป็น .NET object ธรรมดา



รูปที่ 4-9 Runtime Callable Wrapper

4.2.9.2 การใช้ .NET ทำ SOAP Client

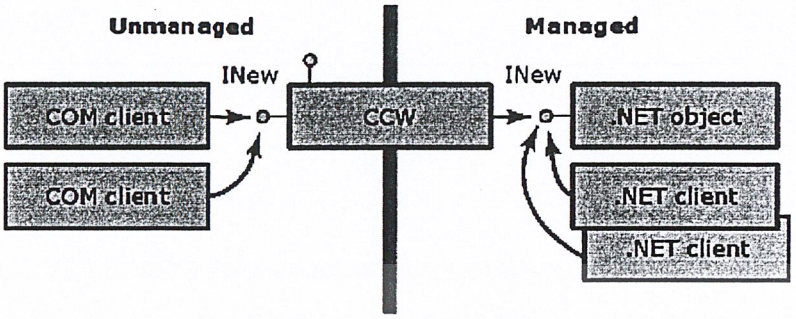
ในทางกลับกันที่ COM ต้องการใช้บริการจาก SOAP ก็กระทำผ่าน .NET เว็บเซอร์วิส client



รูปที่4-10 การใช้ .NET ทำ SOAP client

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่การที่จะทำให้ COM ติดต่อกับ .NET class ได้นั้นต้องผ่านกระบวนการ COM Interop โดยการสร้าง COM callable wrapper ซึ่งจะทำให้ COM สามารถเห็น .NET class เป็น COM object ธรรมดา



รูปที่ 4-11 COM callable wrapper ที่ทำให้ COM client สามารถเห็น

4.3 UDDI

UDDI (Universal Description, Discovery and Integration) เป็นมาตรฐานที่จัดตั้งขึ้น โดยบริษัท IBM, Microsoft และบริษัทยักษ์ใหญ่ทางธุรกิจ B2B (Business-to-Business) อื่น ๆ UDDI ถูกสร้างขึ้นมาเป็นมาตรฐาน ในการค้นหาบริการ เว็บเซอร์วิส สำหรับคู่ค้าทางธุรกิจ ซึ่งเปรียบได้กับฐานข้อมูล ขนาดใหญ่ ซึ่งมีข้อมูลของ เว็บเซอร์วิส ที่เปิดให้บริการ โดยที่ web site สำหรับค้นหา เว็บเซอร์วิส มีอยู่หลายที่ อย่างเช่น

- http://uddi.microsoft.com/search.aspx
- http://www-3.ibm.com/services/uddi/testregistry/find
- http://uddi.org

4.4 WSDL

WSDL (Web Services Description Language) เกิดจากความร่วมมือระหว่าง IBM และ Microsoft WSDL เป็นภาษาที่ใช้อธิบายคุณลักษณะของ เว็บเซอร์วิส และวิธีการติดต่อกับ เว็บเซอร์วิส นั้น ๆ โดยใช้ไวยากรณ์ของภาษา XML ซึ่ง WSDL อยู่ในความดูแลของ W3C version ล่าสุด คือ WSDL 1.1 รายละเอียด สามารถหาอ่านเพิ่มเติมได้จากเว็บไซต์ของ W3C ที่ http://www.w3.org/TR/wsdl ส่วนในทางปฏิบัติ หากเราต้องการ สร้าง เว็บเซอร์วิส ขึ้นมาเป็นของตนเอง ก็สามารถสร้างเอกสาร WSDL ได้โดยอัตโนมัติ เราจึงไม่ต้องไปกังวลในรายละเอียด ในข้อกำหนดใน WSDL มากนักโดยตัวอย่างเอกสาร WSDL เช่น

```
<?xml version="1.0"?>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<definitions name="StockQuote"target namespace="http://example.com/stockquote.wsdl"
      xmlns:tns="http://example.com/stockquote.wsdl"
      xmlns:xsd1="http://example.com/stockquote.xsd"
      xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns="http://schemas.xmlsoap.org/wsdl/">
 <types>
   <schema target namespace="http://example.com/stockquote.xsd"
        xmlns="http://www.w3.org/2000/10/XMLSchema">
     <element name="TradePriceRequest">
       <complexType>
         <all>
           <element name="tickerSymbol" type="string"/>
         </all>
       </complexType>
     </element>
     <element name="TradePrice">
       <complexType>
         <all>
           <element name="price" type="float"/>
         </all>
       </complexType>
     </element>
   </schema>
 </types>
 <message name="GetLastTradePriceInput">
   <part name="body" element="xsd1:TradePriceRequest"/>
 </message>
 <message name="GetLastTradePriceOutput">
   <part name="body" element="xsd1:TradePrice"/>
 </message>
 <portType name="StockQuotePortType">
   <operation name="GetLastTradePrice">
     <input message="tns:GetLastTradePriceInput"/>
     <output message="tns:GetLastTradePriceOutput"/>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    </operation>
  </portType>
  <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetLastTradePrice">
      <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
  <service name="StockQuoteService">
    <documentation>My first service</documentation>
    <port name="StockQuotePort" binding="tns:StockQuoteBinding">
      <soap:address location="http://example.com/stockquote"/>
    </port>
  </service>
</definitions>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

ASP .NET

5.1 ASP คืออะไร

ASP (Active Service Pages) เป็นการเขียนโปรแกรมเพื่อประมวลผลคำสั่งบนเว็บเซิร์ฟเวอร์ ก่อนจะส่งผลลัพธ์กลับมายังเบราเซอร์ที่ร้องขอข้อมูลไป

เหตุผลที่ต้องมีการประมวลผลข้อมูลบนเว็บเซิร์ฟเวอร์ ก็คือ แม้ว่าเราสามารถเขียนโปรแกรมเพื่อทำตามเงื่อนไขบางอย่างได้ทางฝั่งไคลเอนต์ หรือฝั่งของผู้ใช้ โดยใช้ภาษาสคริปต์ เช่น JavaScript, VBScript แต่ก็สามารถทำได้ภายในขอบเขตที่จำกัดเท่านั้น มูลเหตุสำคัญประการหนึ่งของการพัฒนาเว็บไซต์ในปัจจุบัน คือจำเป็นต้องใช้ข้อมูลนการพัฒนา โดยที่ข้อมูลเหล่านี้จะมีปริมาณค่อนข้างมากตามขนาดของเว็บไซต์ ซึ่งการที่จะเคลื่อนย้ายข้อมูลเหล่านี้มายังไคลเอนต์แล้วประมวลที่นี่ย่อมเป็นไปได้ยาก นอกจากนี้แล้วความต้องการข้อมูลของผู้ใช้แต่ละคนจะมีเงื่อนไขที่ต่างกันออกไป ดังนั้นจึงจำเป็นต้องมีกระบวนการในการคัดเลือก และประมวลผลข้อมูลเหล่านั้นก่อนที่จะส่งไปให้กับผู้ใช้ ซึ่งการสร้างเว็บเพจแบบไดนามิก (Dynamic) นี้จะช่วยให้เราสามารถพัฒนาเว็บไซต์ในรูปแบบที่หลากหลาย และสามารถตอบสนองต่อผู้ใช้ได้อย่างสมบูรณ์มากยิ่งขึ้น

5.2 การเปลี่ยนแปลงจาก ASP ไปสู่ ASP .NET

เนื่องจากอินเทอร์เน็ตนั้นเติบโตขึ้นทุกวัน และไม่ได้จำกัดอยู่เฉพาะบนเครื่องพีซีเท่านั้น แต่ยังรวมไปถึงอุปกรณ์พกพาอย่าง มือถือ, PDA, PocketPC เป็นต้น ด้วยเหตุนี้ไมโครซอรฟต์จึงต้องการให้ ASP สามารถรองรับการทำงานร่วมกับอุปกรณ์เหล่านี้ได้ด้วย

ก่อนหน้าที่จะมาเป็น ASP .NET ไมโครซอฟต์ได้พัฒนา ASP มาแล้ว 3 เวอร์ชั่น ซึ่งแต่ละเวอร์ชั่นก็ได้รับความนิยมอย่างดีจากผู้ใช้ทั่วโลก แต่อย่างไรก็ตาม ASP เวอร์ชั่นก่อนๆ นั้นยังมีข้อจำกัดอยู่ เช่น ต้องอาศัยภาษา HTML เป็นหลักในการเขียนโปรแกรม แล้วแทรกด้วยสคริปต์ของ VBScript ทำให้มีข้อจำกัดในการพัฒนาโปรแกรมเพราะเราไม่สามารถดึงความสามารถของภาษามาใช้ได้อย่างเต็มที่ ในเวอร์ชั่น .NET จึงได้เปลี่ยนจากการใช้ VBScript มาเป็นภาษาที่สามารถขยายขีดความสามารถได้อย่างเต็มที่อย่าง VB .NET หรือ ซี-ชาร์ป (C#) ให้ผู้ใช้เลือกตามความถนัด

นอกจากนี้แล้ว ASP .NET ยังสร้างสิ่งอำนวยความสะดวกมากมายมาช่วยให้เราสามารถเขียนโปรแกรมได้ง่าย สะดวกรวดเร็ว และช่วยลดข้อผิดพลาดลงได้มาก ซึ่งต่อไปแม้ว่าผู้เขียนโปรแกรมจะไม่มีพื้นความรู้ทางด้าน Programming มากนักก็สามารถสร้างเว็บไซต์ของตนเองได้แล้ว

การเปลี่ยนแปลงที่สำคัญอีกอย่างหนึ่ง คือเปลี่ยนจากการแปลภาษาสคริปต์ทีละบรรทัด (interpret) ใน ASP แบบเดิม มาเป็นการคอมไพล์โค้ด ASP .NET ให้เป็นรูปแบบ IL (Intermediate Language) ตรงนี้เองที่สนับสนุนความจริงที่ว่า ASP .NET จะเขียนด้วยภาษาอะไรก็ได้ที่สามารถคอมไพล์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นรูปแบบ IL (นั่นคือ CLR ยอมรับ) ด้วยเหตุนี้ เราสามารถกล่าวได้ว่า ASP .NET ก็คือ ASP เวอร์ชั่นที่ถูกคอมไพล์นั่นเอง

นักพัฒนาที่ใช้ Visual Basic มักจะคุ้นเคยกับความสะดวกสบายในการเขียนโปรแกรมโดยใช้ฟอร์ม และคอนโทรลซึ่งมีรูปแบบที่สะดวกโดยลากสิ่งที่ต้องการลงมาไว้บนฟอร์มและเขียนโค้ดโดยใช้การเขียนโปรแกรมแบบ event-handling เนื่องจากวิธีนี้เป็นวิธีที่ไม่ยุ่งยาก จึงได้ถูกนำมาใช้ใน ASP.NET

ASP.NET ทำให้การพัฒนาหน้าเว็บสะดวกขึ้นด้วยการเขียนโปรแกรมโดยใช้ฟอร์ม ใน ASP.NET เราเรียกฟอร์มเหล่านี้ว่า เว็บฟอร์ม (Web Forms) ซึ่งนำมาแทนที่หน้า ASP การเขียนโปรแกรมแบบเว็บฟอร์มนั้นเป็นแบบ event base เช่นเดียวกับใน visual basic นอกจากนี้ ASP.NET ยังแยกส่วนระหว่างส่วนแอพพลิเคชันและส่วนในการแสดงผลอีกด้วย

ASP.NET เกี่ยวข้องกับรูปแบบการเขียนโปรแกรมASP โดยมีคุณสมบัติพิเศษเพิ่มเติมขึ้นมา ดังนี้

- มีการแยกส่วนเด็ดขาดระหว่างส่วนโค้ดการประมวลผลของแอพพลิเคชันและส่วนแสดงผล (HTML markup)จึงไม่มีความสับสนในโค้ดทั้งสองส่วนอีก
- มีกลุ่มของเซิร์ฟเวอร์คอนโทรลจำนวนมากให้เลือกใช้ ซึ่งจะถูกแปลงเป็น HTML โดยอัตโนมัติ
- มีการจัดการสถานะโดยใช้อ็อบเจ็กต์เซสชั่น
- เป็นรูปแบบการเขียนโปรแกรมแบบ event-base ทางฝั่งเซิร์ฟเวอร์ ทำให้ง่ายขึ้น
- สามารถเขียนโค้ดส่วนแอพพลิเคชันโดยใช้ภาษาใดใน Microsoft.NETก็ได้ เช่นภาษา VB, C# เป็นต้น โค้ดของแอพพลิเคชันบนฝั่งเซิร์ฟเวอร์จะถูกคอมไพล์เพื่อประสิทธิภาพการทำงานที่ดีกว่า
- มีVisual Studio.NET เป็นเครื่องมืออำนวยความสะดวกในการพัฒนาเว็บฟอร์ม

สรุปข้อดีของ ASP .NET

- เขียนได้ง่าย และดูแลรักษาได้ง่าย เนื่องจาก ASP .NET มีโครงสร้างที่จะช่วยทำให้การนำโค้ดกลับมาใช้ใหม่ และการใช้งานโค้ดร่วมกันทำได้ง่ายขึ้น
- การส่งมอบ (Deployment) แอพพลิเคชั่นทำได้สะดวกมากขึ้น ซึ่งจากในเวอร์ชั่นเดิมของ ASP จะมีความยุ่งยากมาก สำหรับระบบที่มีคอมโพเนนต์อยู่มากมาย แต่สำหรับ ASP .NET นี้การรีจิสเตอร์คอมโพเนนต์จะไม่มีอีกต่อไป และจะไม่มีปัญหาการล็อกไฟล์ DLL เกิดขึ้น ซึ่งในอดีตนั้น ถ้าเราจำเป็นต้องส่งมอบแอพพลิเคชั่นสำหรับเว็บไซต์ขนาดใหญ่ เราจำเป็นจะต้อง Shut down เซิร์ฟเวอร์ก่อนที่จะติดตั้งเวอร์ชั่นใหม่ลงไป ซึ่งสำหรับเว็บไซต์ขนาดใหญ่นั้น เรื่องนี้จะมีผลกระทบเป็นอย่างมาก แต่สำหรับ ASP .NET นั้นจะใช้งานไฟล์ XML เพื่อเก็บข้อมูลต่างๆ ซึ่งจะทำให้เราส่งมอบแอพพลิเคชั่นได้ง่ายขึ้น โดยก็อปปี้ทั้งไดเรคทอรี่เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

นอกจากนี้ ASP .NET ยังสนับสนุนบราวเซอร์หลายๆ ตัวด้วยกัน ทำให้ปัญหาเรื่องเข้ากันได้ (Compatible) นั้นหมดลงไป และ ASP .NET ยังสนับสนุนการคอมไพล์โค้ดด้วย ทำให้เราไม่จำเป็นต้องห่วงในเรื่องของประสิทธิภาพการทำงานซึ่งเกิดขึ้นในเวอร์ชั่นก่อน ที่ใช้ภาษาสคริปต์วิ่งจะต้องมีการแปลความตอนรันแอพพลิเคชั่น นอกจากนี้เรายังสามารถคอมไพล์โค้ดคำสั่งได้ด้วย ซึ่งการคอมไพล์โค้ดจะช่วยซ่อนซอร์สโค้ดที่เราได้เขียนขึ้นมา ทำให้เมื่อเราต้องส่งมอบโปรแกรม เราก็ไม่ต้องกังวลในเรื่องของการเปิดเผยซอร์สโค้ดให้คนอื่นเห็นอีกต่อไป

สรุปตารางเปรียบเทียบระหว่าง ASP กับ ASP .NET

คุณลักษณะ	ASP	ASP .NET
นามสกุลของไฟล์	.asp	.aspx, .ascx, .asmx
การทำงานของโปรแกรม	คอมไพล์โค้ดด้วยตัว Interpreter	คอมไพล์โค้ดด้วยตัว JIT Complier
การจัดการเรื่องสถานะของโปรแกรม	- มีการใช้คุกกี้ (Cookies) ที่ไคลเอ็นต์ - ไม่สามารถทำงานร่วมกันผ่านแพลตฟอร์มได้	- สามารถฝังลงใน URL ได้เลยโดยไม่ต้องใช้คุกกี้ (Cookies) - สามารถบันทึกข้อมูลที่ใช้ร่วมกันลงในฐานข้อมูล SQL ภายนอกได้
อัพเดตไฟล์ DLL Library	ต้องปิดเครื่องเว็บเซิร์ฟเวอร์ก่อน	สามารถคัดลอกไฟล์ DLL ลงในไดเร็กทอรี่ bin ได้ทันที (ไม่ต้องมีการลงทะเบียน)

ตารางที่ 5 -1 ตารางเปรียบเทียบความแตกต่างระหว่าง ASP กับ ASP .NET

5.3 เซิร์ฟเวอร์คอนโทรล (Server Control)

การสร้างกราฟฟิกอินเทอร์เฟส (Graphic Interface) ใน ASP เวอร์ชั่นก่อนๆ ต้องอาศัยแท็ก <input …> ของ HTML เป็นหลักแต่เนื่องจาก่ข้อจำกัดหลายๆ ประการของแท็กดังกล่าวทำให้เราไม่สามารถสร้างกราฟฟิกอินเทอร์เฟสในรูปแบบที่แปลกๆ ใหม่ๆ และขั้นตอนการเขียนโปรแกรมก็ค่อนข้างยุ่งยาก ดังนั้นในเวอร์ชั่น .NET นี้ไมโครซอฟต์จึงได้นำเอาคอนโทรลที่เราคุ้นเคยกันดีในการสร้าง วินโดว์แอพพลิเคชั่น เช่น TextBox, Lable, Button เป็นต้น มาทำเป็นกราฟฟิกอินเทอร์เฟสบนเว็บฟอร์ม (แต่รูปแบบเดิมนั้นก็ยังสามารถใช้ได้อยู่) และสามารถเขียนโปรแกรมเพื่อจัดการบางอีเวนต์ที่สำคัญๆ ของคอนโทรลตัวนั้นๆ ได้ (แต่ทำได้เพียงบางอีเวนต์เท่านั้น ไม่ครบทุกอีเวนต์ที่มีในวินโดว์แอพพลิเคชั่น) ตัวอย่างโค้ดของคอนโทรลพื้นฐานที่ทำงานบนเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<asp : textbox id = text1 runat=server/>
text1.text = "Hello World"

<asp : checkbox id = check1 runat=server/>
check1.checked = True

<asp : button id = button1 runat=server/>
button1_onClick()

<asp : DropDownList  id = DropDownList1 runat=server/>
DropDownList1.SelectedItem.Text = "Hello"
```

5.4 วาริเดชั่นคอนโทรล (Validation Controls)

โดยทั่วไปการกรอกข้อมูลของผู้ใช้มักมีความผิดพลาดอยู่เสมอ ทั้งแบบที่ตั้งใจ และไม่ได้ตั้งใจ ในบางกรณีหากผู้ใช้กรอกข้อมูลที่ผิดพลาด อาจส่งผลให้โปรแกรมไม่สามารถทำงานต่อได้เลย วิธีการเดิมที่ใช้กันคือ การนำข้อมูลที่ผู้ใช้กรอกเข้าไปตรวจสอบด้วยการเขียนโปรแกรมเองทั้งหมด แต่ใน ASP .NET นี้มีคอนโทรลที่มาทำหน้าที่ในการตรวจสอบข้อมูลที่ผู้ใช้กรอกเข้ามาโดยเฉพาะ ว่าเป็นไปตามรูปแบบที่ต้องการหรือไม่ โดยที่เราไม่ต้องเขียนโปรแกรมให้ยุ่งยาก เพียงแต่กำหนดรูปแบบการตรวจสอบให้กับคอนโทรลเท่านั้น ก็สามารถใช้งานได้แล้ว

ตัวอย่างโค้ดของคอนโทรลวาริเดชั่น

```
<asp : RequiredFieldValidator ControlToValidate = "txtName"
ErrorMessage = "Please Enter Your Name" runat = "server"/>
```

5.5 ภาษาที่ใช้เขียน ASP .NET

ภาษาที่ใช้ในการพัฒนา ASP .NET นั้นนับได้ว่าเป็นจุดเปลี่ยนแปลงที่สำคัญที่สุดจุดหนึ่ง ของ ASP .NET จากเดิมที่เราเคยใช้กันแต่ VBScript มาถึงวันนี้ VBScript ได้ถูกแทนที่ด้วย VB .NET, C# และ Jscript ทั้งนี้เพื่อให้เราสามารถดึงความสามารถจากภาษาเหล่านี้มาใช้ได้อย่างเต็มที่

5.6 รู้จักกับ System.Web.UI Namespace

ในส่วนนี้จะอธิบายถึงคลาส ที่สำคัญที่อยู่ในเนมสเปซชื่อ System.Web.UI ในเฟรมเวิร์คของ ASP.NET เนมสเปซ System.Web.UI นั้นระบุคลาสและอินเทอร์เฟสที่ใช้ในการสร้างสิ่งต่างๆในคลาสเว็บฟอร์ม ที่สำคัญที่สุดในSystem.Web.UI คือคลาส Control ซึ่งบอกคุณสมบัติ เมธอด และ event ของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่ละเซิร์ฟเวอร์คอนโทรลในเฟรมเวิร์คของเว็บฟอร์ม อีกคลาสที่สำคัญคือ เพจ ซึ่งจะกล่าวถึงในลำดับต่อไป

5.6.1 คลาส Control

คลาส Control เป็นรากฐานของคอนโทรลทั้งหมด ตัวอย่างเช่น textbox หรือ button ก็คือคอนโทรลโดยทั่วไปคลาส Control จะมีฟังก์ชั่นและคุณสมบัติของส่วนติดต่อกับผู้ใช้ทุกตัว ทุกอย่างที่เห็นใน ASP.NET ก็เป็นคลาสที่สืบทอดมาจากแหล่งใดแหล่งหนึ่งทั้งสิ้น

5.6.1.1 คุณสมบัติของ Control

คลาส Control มีคุณสมบัติที่สำคัญดังต่อไปนี้ Control, ID, Parent, EnableViewState, Visible, Context และ ViewState คุณสมบัติของคอนโทรลเหล่านี้แทนลูกของ instace ของคอนโทรลคุณสมบัติของคลาสแม่จะบอกคลาส แม่ของคอนโทรล คุณสมบัติทั้งสองนี้ ทำให้มีลำดับชั้น(hierarchy)ของคอนโทรลในหน้าเว็บคุณสมบัติ ID ทำให้คอนโทรลถูกเข้าถึงจากการเขียนโปรแกรมได้โดยใช้ ID ตามด้วยจุดและชื่อคุณสมบัติหรือเมธอดอื่นๆที่เราต้องการจะเข้าถึงของคอนโทรลนั้น เช่น MyObjectId.propertyname และยังทำให้เราสามารถเขียนตัวจัดการกับเหตุการณ์ที่เกิดขึ้นกับคอนโทรลได้ด้วย

EnableViewState เป็น flag ที่บอกว่าคอนโทรลจะยังคงเก็บview ของสถานะมันไว้หรือไม่ เช่นเดียวกับ view สถานะทั้งหมดของคอนโทรลลูกของมัน คุณสมบัติของ Context ทำให้เราสามารถดึงข้อมูลของ HTTP request ได้

ViewState เป็นอินสแตนท์ของคลาส StateBag ซึ่งใช้ในการเก็บชื่อและค่าของข้อมูลที่สามารถสร้างให้เข้าถึงได้โดยหลาย request สำหรับหน้าเพจเดียว ชื่อและค่าเหล่านี้คืออินสแตนท์ของคลาส StateItem

5.6.1.2 เมธอดในคลาส Control

เมธอดของคลาส Control นั้นมีเป็นจำนวนมาก อย่างไรก็ดีเราจะกล่าวถึงเพียงเมธอดที่มีความสำคัญ

เมธอด DataBind

เป็นการโยงคอนโทรลเข้ากับแหล่งข้อมูล เมื่อเมธอด นี้ถูกเรียก แท็กของการผูกข้อมูล <%#%> จะถูกนำมาพิจารณาใหม่เพื่อทำให้ข้อมูลใหม่ถูกโยงเข้ากับตำแหน่งtagที่เหมาะสม

เมธอด CreateChildControls

จะถูกเรียกก่อนที่คอนโทรล compositional custom จะถูกจัดแสดงคอนโทรล compositional custom นั้นคล้ายกับคอนโทรล ActiveX คือมันจะจัดการสร้างคอนโทรลอื่นๆ ในการพัฒนาคอนโทรล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

custom เมธอดนี้สามารถถูก overide เพื่อให้ผู้พัฒนาคอนโทรล custom สามารถสร้าง คอนโทรลลูก ก่อนทำการแสดง คอนโทรลไม่ว่าในการทำครั้งแรกหรือครั้งต่อๆไปก็ตาม

เมธอด Render

คล้ายกับ CreateChildControls เริ่มต้นใช้สำหรับพัฒนาคอนโทรล custom ผู้พัฒนาคอนโทรลจะ overrideเมธอดนี้ เพื่อแสดงตัวเนื้อหาในคอนโทรลผ่านทางพารามิเตอร์ HtmlTextWriter ที่เตรียมไว้

เมธอด SaveViewState และ LoadViewState

เก็บบันทึกและเรียกซ้ำสถานะของคอนโทรล คอนโทรลของเซิร์ฟเวอร์ คงสถานะของมันระหว่างการrequest ผ่านทางเมธอด นี้

5.6.2 คลาสเพจ

คลาสเพจนี้สืบทอดมาจากคลาส Control ซึ่งหมายความว่ามันสืบทอดมาทั้งคุณสมบัติเมธอด และ event ที่แสดงโดยคลาส Control นอกจากการสืบทอดแล้วคลาส เพจ ระบุคุณสมบัติเมธอด และ event ที่เจาะจงมากขึ้น สำหรับหน้าเว็บในเฟรมเวิร์คของ ASP.NET

ใน ASP.NET ทั้ง Application , Request , Respond , Server และ Session เป็นคุณสมบัติของคลาส เพจ นอกเหนือจากอ๊อบเจ็กต์เหล่านี้คลาส เพจ ยังมีคุณสมบัติอื่นๆอีก เช่น Cache, ErrorPage, Postback, IsValid, Trace และ Validators

5.6.2.1 ตัวอย่างคุณสมบัติและเมธอดของคลาสเพจได้แก่

คุณสมบัติ Cache

ชี้ไปยังอ๊อบเจ็กต์ cache ของขอบเขตของหน้าปัจจุบัน ในที่นี้ ทรัพยากรเช่นการติดต่อฐานข้อมูล (database connection)จะถูกเก็บเพื่อนำมาใช้ใหม่โดยไม่ต้องสร้างการติดต่อใหม่ หากว่า cache item นั้นยังไม่หมดอายุ

คุณสมบัติ ErrorPage

บอกหน้าที่จะถูกเรียกให้แสดงเมื่อมีerror เกิดขึ้น อาจใช้ @Pageแทนได้

คุณสมบัติ IsPostBack

บอกว่าการเรียกหน้าเว็บเป็นการโหลดครั้งแรกหรือไม่ หาก IsPostBack มีค่าเป็น true แสดงว่าไม่ใช่การโหลดครั้งแรก จึงไม่ควรทำการ initialize หน้าเพจซ้ำ เพื่อทำให้ประสิทธิภาพดีขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติ Validator

รวมกลุ่มของเซิร์ฟเวอร์คอนโทรลซึ่งสามารถกำหนดระยะการทำงานของตัวเองลงในคุณสมบัติ Validator ของหน้าเว็บ เมื่อหน้าเว็บใดต้องมีการกำหนดอายุของตัวเอง ก็จะกำหนดอายุของคอนโทรลทั้งหมดในหน้านั้นแล้วกำหนดค่าที่เหมาะสมให้แก่คุณสมบัติ IsValid

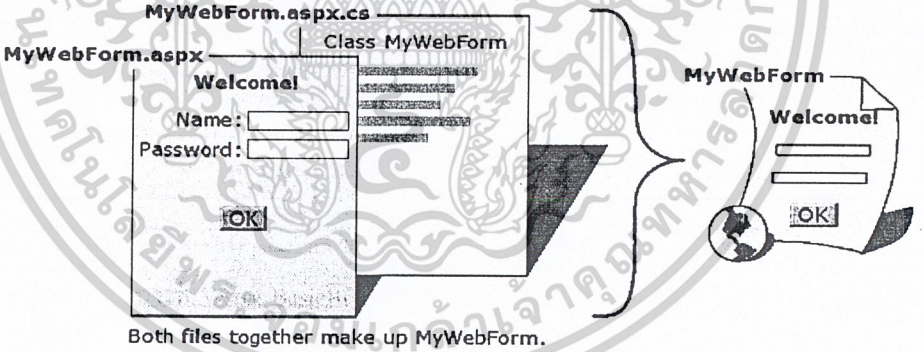
คุณสมบัติ Trace

อ้างอิงจากอ๊อบเจ็ค TraceContext ซึ่งใช้บอกคำเตือนหรือข้อความบอกความผิดพลาดที่เกิดขึ้น สามารถกำหนดให้ทำการตรวจสอบหรือไม่ทำก็ได้โดยกำหนดใน web.config ซึ่งเป็นเท็กซ์ไฟล์ที่เก็บค่าขณะรันไทม์ของแอพพลิเคชันของASP.NET

5.7 การพัฒนาเว็บแอพพลิเคชันด้วย ASP.NET

5.7.1 ส่วนประกอบคอมโพเนนต์ของ Web Form

ใน Web Forms pagesนั้น การเขียน program ของ user interface แบ่งเป็นสองส่วนคือ visual Component และ logic ซึ่งก็คล้ายกับการที่ Visual Basicมีการแบ่งระหว่างส่วนของ form และส่วน code เบื้องหลังที่ formนั้นทำงานด้วยนั่นเอง ส่วนแสดงผลที่มองเห็นได้นั้นก็คอ Web Form page ซึ่งประกอบด้วยไฟล์ ที่มี HTML หรอ ASP.NET server control. Web Form page ทำงานเหมือนเป็น container สำหรับ static text และส่วนควบคุมที่เราต้องการแสดงผล ในส่วน logic ของ Web Form page ประกอบด้วย โค้ด ที่เรา สร้างขึ้นเพื่อสื่อสารกับ form logic การเขียนโปรแกรม จะอยู่แยกกับส่วน file ของ user interface ซึ่งไฟล์นี้จะเปรียบเสมือน โค้ดที่อยู่เบื้องหลังการทำงาน สามารถเขียนด้วยภาษา Visual Basic หรือ Visual C#



รูปที่ 5-1 ส่วนประกอบของเว็บฟอร์ม

ไฟล์ที่ประกอบด้วยคลาสต่างๆและ เบื้องหลังการทำงาน สำหรับ Web Form ทั้งหมดจะถูกคอมไพล์เป็น ไฟล์ dynamic-link library(.dll) page ที่เป็น .aspx ก็จะถูก compile เช่นกัน แต่มีกระบวนการที่ต่างกัน ในครั้งแรกที่ ผู้ใช้ เรียก file หน้า .aspx ASP.NET จะสร้างไฟล์ .NET class มาเป็นตัวแทนของ page โดยอัตโนมัติ แล้ว คอมไพล์ เป็น ไฟล์ .dll ตัวที่สอง class ที่ถูกสร้างสำหรับ .aspx page สืบทอดคุณสมบัติมาจาก class ที่เป็น โค้ดการทำงานเบื้องหลังที่ถูก compile ไปแล้วก่อนหน้านี้ เมื่อผู้ใช้ร้องขอ URLของ web page ไฟล์ .dll จะ run บน server และสร้างผลเป็น HTMLแบบไดนามิคสำหรับ page ที่เรียก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.7.2 วงจรชีวิตของ Web Form

ใน ASP เว็บเพจจะเริ่มต้นวงจรชีวิตเมื่อไคลเอนท์เริ่มทำการร้องขอเพจที่เจาะจงมา IIS ทำการกระจายคำและทำการใช้งานสคริปต์บน ASP เพจ เพื่อทำให้เป็นเนื้อความ HTML ในที่สุดการทำการแปลงก็จะสิ้นสุดอายุของเพจนั้นก็จะสิ้นสุด ถ้าคุณมีฟอร์มที่ส่งข้องมูลกลับไปที่ ASP เพจก็จะเริ่มกระบวนการใหม่เหมือนกับ ASP กระทำคำร้องใหม่ โดยไม่ต้องรู้เกี่ยวกับขั้นตอนก่อนหน้าเลย การส่งผ่านข้อมูลกลับไปที่เพจเริ่มต้นสำหรับการประมวลผลก็เป็นการถูกอ้างแบบโพสต์แบ็ค (postback) ด้วย

ใน ASP.NET มีหลายสิ่งที่แตกต่างกันเล็กน้อย เพจยังคงเริ่มที่การร้องขอของไคลเอนท์อย่างไรก็ตามมันจะยังคงอยู่เรื่อยๆเท่าที่ไคลเอนท์ยังคงติดต่อกับเพจอยู่ เพื่อทำให้ง่ายขึ้นอาจจะกล่าวได้ว่าเพจยังคงอยู่ไปเรื่อยแต่ในความเป็นจริงมีเพียงขั้นตอนการดู(view)ของเพจระหว่างการร้องขอเท่านั้น ขั้นตอนการดูเท่านั้นที่อนุญาตให้ควบคุมบนเซิร์ฟเวอร์ที่จะปรากฏว่า ถ้ามันยังคงครองอีเวนต์ของเซิร์ฟเวอร์ เราสามารถค้นหาขั้นตอนการโพสต์แบ็คของเพจผ่านคุณสมบัติ IsPostBack ของเพจอ็อบเจ็กต์และการกำหนดค่าเริ่มต้นใหม่ การครองอีเวนต์ระหว่างการโพสต์แบ็คมันก็คือการทำ ASP.NET แตกต่างกว่าการพัฒนา ASP ตามแบบธรรมดาอย่างมาก

จากตัวอย่างเราทำการขยายส่วนต้น เช่นการครองโพสต์แบ็ค เมื่อทำการโหลดอีเวนต์ที่ถูกถือครองครั้งแรก เราจะทำการย้ายข้อมูลลงใน drop-down list box หลังจากนั้นเราจะระบุเวลาของอีเวนต์ที่เพิ่มขึ้นโดยไม่มีการโหลดข้อมูลเข้ามาใหม่เท่านั้น ตัวอย่างนี้แสดงให้เห็นว่าเซิร์ฟเวอร์อีเวนต์แฮนด์เลอร์ (handler)ที่ชื่อ handlerButtonClick เป็นขอบเขตของอีเวนต์ที่ชื่อ ServerClick ของปุ่ม

```
<html>
    <head><title>Testing Page Event</title></head>
    <body>
        <script language="C#" runat=server>
            void Page_Init(Object oSender,EventArgs oEvent){
                labelInit.Text = DataTime.Now.ToString( );
            }
            void Page_Load (Object oSender, EventArgs oEvent) {
                labelInit.Text = DataTime.Now.ToString( );
                if (!IsPostBack){
                    selectCtrl.Items.Add("Acrua");
                    selectCtrl.Items.Add("BMW");
                    selectCtrl.Items.Add("Cadillac");
                    selectCtrl.Items.Add("Mercedes");
                    selectCtrl.Items.Add("Porche");
                }else{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    labelLoad.text += "(Postback)";
                }
            }
            void handleButtonClick(Object oSender, EventArgs oEvent){
                labelOutput.Text = "You've selected: "+ selectCtrl.Value;
                labelEvent.Text = DateTime.Now.ToString ( );
            }
        </script>
        <form runat = server>
            Init Time: <asp:Label id=labelInit runat=server/><br/>
            Load Time: <asp:Label id=labelLoad runat=server/><br/>
            Event Time: <asp.Label id=labelEvent runat=server/><br/>
            Choice: <select id=selectCtrl runat=server/><br/>
            <asp:Label id=labelOutput runat=server></select><br/>
            <input type=button value=update
                onServerClick="handleButtonClick" runat=server />
        </form>
    </body>
</html>
```

วงจรชีวิตของเว็บฟอร์ม ประกอบด้วยสามขั้นตอนหลัก การกำหนดรูปแบบ การถือครองอีเวนต์ และการสิ้นสุด โดยขั้นตอนเหล่านี้จะข้ามผ่านคำร้องขอที่เหมือนกันซึ่งแย้งกับนโยบาย serving-one-page-at-a-time ที่พบใน ASP

5.7.2.1 การกำหนดรูปแบบ

ขั้นตอนการกำหนดรูปแบบนี้ โหลดอีเวนต์ของเพจจะถูกยกขึ้นมันเป็นงานของคุณในการถือครองอีเวนต์นี้เพื่อติดตั้งเพจคุณ เพราะว่าโหลดอีเวนต์จะถูยกขึ้นเมื่อการควบคุมพร้อมแล้วงานคุณตอนนี้คือการอ่านและทำการอัพเดตคุณสมบัติการควบคุมเหมือนเป็นส่วนของการติดตั้งเพจ ในตัวอย่างโค้ดข้างต้น เราจะถือครอง โหลดอีเวนต์เพื่อย้ายข้อมูลบางอย่างลง drop-down list เราอัพเดตข้อความของการควบคุม labelLoad เพื่อแสดง เวลาโหลดอีเวนต์ที่เกิดขึ้น ในแอพลิเคชันของคุณคุณจะต้องข้อมูลที่เป็นไปได้จาก ฐานข้อมูลและทำการหนดค่าเริ่มต้นฟอร์มฟิลด์ด้วยค่าดีฟอลต์

คุณสมบัติ IsPostBack ของเพจแสดงว่าเพจแรกได้ถูกโหลดหรือมันเป็นการโพสต์แบคเช่นถ้าคุณควบคุมเนื้อหารายการของข้อมูลคุณจะต้องโหลดการควบคุมเพจแรกที่ถูกโหลดโดยการตรวจสอบคุณสมบัติ IsPostBackของเพจนี้เท่านั้นเมื่อ IsPostBack มีค่าเป็นจริงคุณจะรู้ว่ารายการควบคุมอ็อบเจ็กต์ได้ถูกโหลดด้วยข้อมูลเรียบร้อยแล้ว ไม่จำเป็นที่ต้องย้ายข้อมูลในรายการใหม่ตัวอย่างโค้ดข้างต้นเราจะข้ามการการย้ายข้อมูลของ drop-down และแสดงเพียง สตริงว่า "(postback)"

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณอาจจะต้องกระทำการการผูก (bind) ข้อมูลและกำหนดค่าใหม่ data-binding expressions ในตอนแรก และลำดับต่อมาของเพจนี้

5.7.2.2 การถือครองอีเวนต์

ในขั้นตอนกลางนี้ ฟังก์ชัน server event-handlingของเพจจะถูกเรียกเป็นผลของบางอีเวนต์เป็นการถูกเรียกให้ทำงานจากฝั่งไคลเอนต์ อีเวนต์เหล่านี้มาจากการควบคุมที่คุณแทนที่บนเว็บฟอร์ม

5.7.2.3 การจบการทำงาน

ที่ขั้นตอนนี้เพจจะสิ้นสุดการแปลงและถูกกำจัดทิ้งแล้วสิ่งที่คุณต้องรับผิดชอบคือการทำความสะอาดไฟล์แฮนด์เลอร์ ยกเลิกการติดต่อกับฐานข้อมูลและปล่อยอ็อบเจ็กต์ถึงแม้คุณจะเชื่อมั่นการทำงานของตัวกำจัดขยะ(garbage collection)ของ CLR ก็ตามแต่เรายังแนะนำอย่างยิ่งยวดว่าควรจำทำความสะอาดด้วยตัวเองเพราะตัวกำจัดขยะจำทำงานเป็นระยะๆเท่านั้น บนการทำงานหนักๆถ้าตัวกำจัดขยะไม่เหมาะสมจะมีทรัพยากรที่ไม่ได้คืนให้ระบบทำให้หน่วยความจำหมดไปได้และทำให้ระบบคุณค้างไป

เราสามารถทำความสะอาดตัวอย่างข้างต้นได้ด้วยการอันโหลดอีเวนต์แฮนด์เลอร์ดังที่จะแสดงต่อไปนี้เพราะว่ าไม่มีแสดงในตัวอย่างเราจะแสดงเฉพาะฟังก์ชันเป็น เทมเพลต

```
Void Page_Unload(Object oSender,Eventargs oEvent){
    //cleaning up code here
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
ASP .NET เว็บเซอร์วิส

" เว็บเซอร์วิส คือ แอพพลิเคชั่น หรือโปรแกรมซึ่งทำงานอย่างใดอย่างหนึ่งในลักษณะให้บริการ โดยจะถูกเรียกใช้งานจากแอพพลิเคชั่น หรือโปรแกรมอื่นๆ ผ่านเว็บ ซึ่งเว็บเซอร์วิสได้อาศัยโปรโตคอลที่ชื่อว่า SOAP (Simple Object Access Protocol) ด้วยโปรโตคอลนี้เองที่ทำให้เราสามารถเรียกใช้แอพพลิเคชั่นใดๆ ในแพลตฟอร์มใดๆ ก็ได้ การให้บริการของเว็บเซอร์วิส จะมีเอกสารที่อธิบายคุณสมบัติของบริการกำกับไว้ (WSDL : Web Service Description Language) และมีการนำเสนอให้สาธารณชนรับทราบ ผู้ใช้บริการจึงสามารถค้นหาเว็บเซอร์วิส ได้จาก UDDI (Universal Description Discovery and Integration) โดยที่ผู้ใช้บริการไม่จำเป็นต้องรู้ที่อยู่จริงของแอพพลิเคชั่นหรือโปรแกรมนั้น "

6.1 องค์ประกอบในการใช้งาน .NET เว็บเซอร์วิส

6.1.1รูปแบบการเชื่อมต่อ

Microsoft .NET Web Services ตอนนี้รองรับ 3 โปรโตคอล ได้แก่ HTTP GET, HTTP POST และ SOAP (Simple Object Access Protocol) เนื่องจากโปรโตคอลเหล่านี้เป็นมาตรฐานของเว็บอยู่แล้ว จึงง่ายที่ไคลเอนต์จะเรียกใช้

HTTP GET

query string จะรวมอยู่ใน URL ของเซิร์ฟเวอร์นั้น

HTTP POST

จะทำการรวม ชื่อ ค่าตัวแปรต่างๆ ไว้ในส่วนของบอดี้ ของ HTTP request message

SOAP

SOAP ถูกสร้างบน XML เป็นการทำให้แอพพลิเคชั่นสามารถติดต่อกันได้ตามมาตรฐาน และง่ายในการเชื่อมต่อ ทำให้สามารถสร้างแอพพลิเคชั่นส่งไปตามอินเทอร์เน็ตได้ ดังรูปที่ 1

ถ้าเราคิดเกี่ยวกับการโต้ตอบกันระหว่าง XML Web Service เช่น การโทรศัพท์ XML จะอธิบายสิ่งที่แอพพลิเคชั่นพูดกันโดยการใช้ภาษาของแอพพลิเคชั่น ส่วน SOAP จะอธิบายวิธีที่ติดต่อกันโดยใช้โทรศัพท์

SOAP เหมือน XML คือเป็นส่วนหนึ่งของมาตรฐาน W3C และไมโครซอฟท์เป็นผู้ร่วมออกมาตรฐานด้วย พร้อมกับบริษัทชั้นนำอื่น เช่น IBM

SOAP จะแตกต่างจากทั้งสองวิธีข้างต้น มันจะใช้ XML ในการจัดวางรูปแบบเมสเสจที่ส่งไปจะมีโครงสร้างที่ดีกว่า และสามารถส่งผ่านข้อมูลที่ซับซ้อนได้ ข้อแตกต่างอีกอย่างคือ SOAP สามารถใช้บนทรานสปอร์ต โปรโตคอล (transport protocols) อื่นๆได้ เช่น SMTP นอกเหนือจาก HTTP

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.1.2 WSDL

WSDL (Web Services Description Language) มีวิธีที่เป็นมาตรฐานในการอธิบายฟังก์ชันที่ใช้ได้จาก XML Web Services เฉพาะ และอาร์กิวเมนต์ที่จะต้องส่งผ่านเพื่อติดต่อกัน WSDL เป็นเช่นเดียวกับ SOAP คือได้รับการดูแลโดย W3C เพื่อกำหนดเป็นมาตรฐาน

6.1.3 Web Services Discovery

เป็นกระบวนการในการค้นหาเซอร์วิสและทำการตรวจสอบ WSDL ของเว็บเซอร์วิสว่ามีอยู่จริงไหม แบ่งเป็น 2 แบบ

Static discovery

เราต้องทำการบอกชัดเจนในไฟล์ .disco ว่า wsdl อยู่ที่ไหน

```
<?xml version="1.0" ?>
<disco:discovery xmlns:disco="http://schemas.xmlsoap.org/disco">
</disco:discovery>
```

Dynamic discovery

เราสามารถตั้งค่าให้เป็นแบบ dynamic discovery ได้ ซึ่งจะทำให้เว็บเซอร์วิสทั้งหมดที่อยู่ภายใต้ URL ที่เรากำหนดจะถูกลิสต์ขึ้นมาโดยอัตโนมัติ

6.1.4 UDDI (Universal Description, Discovery and Integration)

ถ้า XML เป็นเหมือนการสนทนา และ SOAP เหมือนเป็นหลักสำหรับวิธีการติดต่อผู้อื่น UDDI ก็คือสมุดโทรศัพท์ UDDI จะจัดเตรียมไดเร็กทอรีของ XML Web Services ช่วยให้สามารถค้นหาธุรกิจที่ใช้ XML Web Services ได้

6.2 การสร้าง ASP.NET เว็บเซอร์วิส

6.2.1 ขั้นตอนในการสร้างเว็บเซอร์วิส

1. สร้างไฟล์ .asmx สำหรับเว็บเซอร์วิสที่สร้างด้วย ASP.NET ต้องประกอบไปด้วย directive <% webservice . . . %> การเขียนก็คล้ายกับการเขียนคลาส
2. เว็บเซอร์วิสคลาส (Web Services Class) นั้นต้องทำการ inherite มาจาก System.Web.Services namespace รวมทั้งทำการกำหนด namespace ของเว็บเซอร์วิสของคุณด้วยเนื่องจาก defalut ของมันคือ http://tempuri.org/ ซึ่งจะไม่ unique
3. เมธอดที่ต้องการให้เป็น เมธอดสาธารณะที่คนอื่นๆเข้ามาเรียกใช้ได้ ต้องทำการกำหนดให้เป็น <Web Method> ด้วย
4. นำไฟล์ .asmx ที่สร้างไปไว้ในไดเร็กทอรีที่ต้องการใน IIS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2.2 ตัวอย่างเว็บเซอร์วิส

```
<%@ WebService Language="VB" Class="Math"%>
Imports System.Web.Services
Imports System
    <WebService(Namespace:="http://www.olala.com/")> _
    Public Class Math
     <WebMethod()> _
     Public Function Add(num1 As Integer, num2 As Integer) _
            As Integer
       Return num1 + num2
     End Function
    <WebMethod()> _
     Public Function Hello() _
            As String
       Return "Hello World"
     End Function
   End Class
```

6.2.3 การกำหนดค่าแอททริบิวต์ต่างๆให้แก่เว็บเซอร์วิส

BufferResponse (boolean)

ควบคุมให้มีบัฟเฟอร์ (buffer) ของผลตอบรับของเมธอดหรือไม่

CacheDuration

กำหนดค่าช่วงเวลา (วินาที) ที่ทำการเก็บ method response ใน cache ค่าปกติคือ 0 วินาที

Description

กำหนดคำอธิบายของเมธอดนี้

EnableSession (boolean)

ให้มีการใช้ เซสชั่นสำหรับเมธอดนี้หรือไม่ โดยค่าปกติเป็น true ถ้าเราไม่ได้ ใช้งานควรกำหนดเป็น false เพื่อเพิ่มประสิทธิภาพ

MessageName

เพื่อใช้ในการแย่งแยกเว็บเมธอดในกรณีที่มี ชื่อของเมธอดเหมือนกัน

TransactionOption

เป็นการกำหนด transaction ของเมธอดนั้น มีทั้งหมด 5 แบบ ดังที่ได้กล่าวไปแล้วในเรื่อง Transaction ได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. Disabled - object ที่สร้างขึ้นจะไม่มี transaction ใน COM+ transaction แต่สามารถติดต่อโดยตรงกับ DTC ในกรณีที่ต้องการการสนับสนุน
2. NotSupported - ไม่มีการทำ transaction เลย
3. Supported - ถ้า object นั้นจะถูกรันใน context transaction ของตัวที่สร้างมันขึ้นซึ่งมามีการทำ transaction , แต่ถ้าตัวมันเอง เป็นตัวเริ่ม(root - ไม่ได้ถูกใครสร้างมาอีกที) หรือ context ที่สร้างมัน ไม่ทำ transaction มันก็จะไม่ทำ transaction
4. Required - ถ้า object นั้นจะถูกรันใน context transaction ของตัวที่สร้างมันขึ้นซึ่งมามีการทำ transaction , แต่ถ้าตัวมันเอง เป็นตัวเริ่ม(root - ไม่ได้ถูกใครสร้างมาอีกที) หรือ context ที่สร้างมัน ไม่ทำ transaction มันก็จะทำ transactionใหม่
5. RequiresNew - บอกว่า object ต้องทำ transaction ใหม่

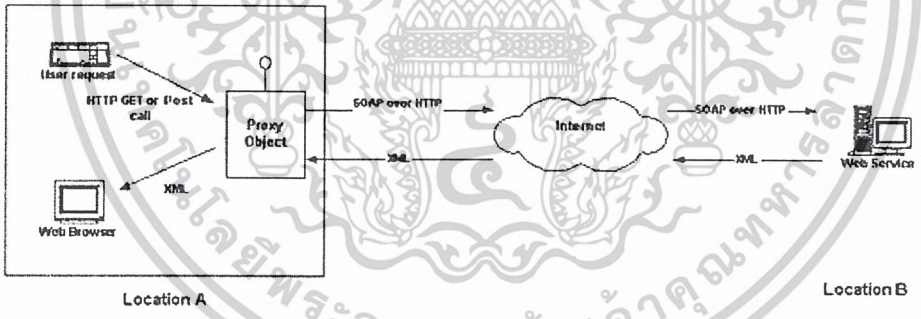
6.3 การเรียกใช้งานเว็บเซอร์วิส

ในการเรียกใช้งานเว็บเซอร์วิสนั้นเราสามารถที่จะเรียกใช้ได้ทั้ง 3 แบบ คือ HTTP GET, HTTP POST , SOAP

HTTP GET นั้นเราสามารถเรียกใช้ Web Service ได้โดยตรงผ่านทาง URL โดยใส่ parameter รวมไปด้วยเลย เช่น

http://localhost/Xtest.asmx/Add?num1=3&num2=6

แต่ในส่วนของ HTTP POST และ SOAP นั้นเราต้องทำการสร้าง Proxy Class ขึ้นมาก่อนเพื่อทำการติดต่อกับ Web Service



รูปที่ 6-1 การติดต่อจากไคลเอนต์ไปยังเว็บเซิร์ฟเวอร์ผ่านพร็อกซี่อ็อบเจ็กต์

ขั้นตอนการสร้าง Proxy Class

1. ใช้ wsdl.exe ในการคอมไพล์โดยใส่พารามิเตอร์เป็น ชนิดโปรโตคอล, ตำแหน่งของเอกสาร wsdl ,ภาษาที่ใช้ และอื่นๆ
2. จะได้ไฟล์ออกมา .cs ซึ่งเป็นไฟล์ของ C# เราใช้ตัวคอมไพล์ของ C# ก็จะได้ไฟล์ .dll ออกมา
3. นำไฟล์ .dll นี้ไป add reference ในโปรเจ็กที่จะเรียกใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.4 การจัดการเสตทในเว็บเซอร์วิส

ในส่วนของ ASP.NET มีฟีเจอร์ในการจัดการเสตท ทั้งในส่วนของระดับแอพพลิเคชัน และระดับของเซสชั่นของผู้ใช้ ในส่วนนี้เราจะกล่าวถึงการใช้งานเสตทของ ASP.NET เพื่อนำมาพัฒนาเสตทฟูลเว็บเซอร์วิส

คำว่า เสตท นั้น เราอาจจะนิยามได้ว่า มันคือความสามารถของอ็อบเจ็กต์, ตัวแปร หรือคอนเทนเนอร์อื่นๆ ที่สามารถจดจำค่าของมันได้ ซึ่งเรามีอยู่สองระดับคือ แอพพลิเคชัน และเซสชั่น ในระดับแอพพลิเคชันนั้น เปรียบเหมือนร้านอาหาร บริกรต้องจำได้ว่า วันนี้ร้านมีเมนูพิเศษอะไรบ้าง ซึ่งจะนำไปบอกลูกค้าทุกโต๊ะเหมือนกันหมด แต่ระดับเซสชั่น บริกรจะจำได้ว่า ลูกค้าคนนี้สั่งอะไร แต่ละคนสั่งคนละอย่างไม่เกี่ยวกัน

ทั้งสองระดับที่กล่าวมานั้นใน ASP.NET เรียกว่า แอพพลิเคชันอ็อบเจ็กต์ ซึ่งให้บริการกระบวนการในกาจัดเก็บข้อมูล ซึ่งสามารถเข้าถึงได้จากทุกโค้ด ที่อยู่ในเว็บแอพพลิเคชันนั้น และเซสชั่นอ็อบเจ็กต์ ซึ่งจัดเก็บข้อมูลตามเซสชั่นของไคลเอนท์แต่ละคนที่เข้ามา

6.4.1 แอพพลิเคชันอ็อบเจ็กต์

แอพพลิเคชัน อ็อบเจ็กต์จะถูกสร้างขึ้น หลังจากเกิดการเรียกใช้เว็บหรือเว็บเซอร์วิสนั้นขึ้นเป็นครั้งแรกจนกระทั่ง ปิดระบบเว็บเซอร์วิสลงไป ซึ่งแอพพลิเคชันอ็อบเจ็กต์นั้นให้การข้อดีต่างๆดังเช่น

- ง่ายต่อการเรียกใช้เสตท และสามารถใช้จากภาษาใดๆก็ได้ที่ .NET รองรับ
- มีฟังก์ชั่นในการจัดการ ซิงโครไนเซชั่น ทำให้นักพัฒนาง่ายต่อการจัดการการเข้าถึง ตัวแปรแบบโกบอลใน แอพพลิเคชันเสตท
- แอพพลิเคชันเสตท เข้าถึงได้เพียงโค้ดในส่วนของแอพพลิเคชันนั้นๆเท่านั้น ไม่สามารถที่แอพพลิเคชันอื่นจะเข้าถึงได้

ตัวอย่างการเรียกใช้งาน แอพพลิเคชัน อ็อบเจ็กต์

```
Application["DatabaseName"] = "DataStore" ;
```

6.4.2 เซสชั่นอ็อบเจ็กต์

เซสชั่นหมายถึง ช่วงเวลาที่ผู้ใช้คนหนึ่งๆทำการติดต่อกับเว็บแอพพลิเคชันหนึ่ง ซึ่งใน ASP.NET มีการทำงานที่ดีขึ้นกว่า ASP เวอร์ชั่นก่อนหน้าดังนี้

- เซสชั่น อ็อบเจ็กต์โปรเซสจะถูกแยกออกมา ไม่ได้อยู่ใน โปรเซสของ ASP เหมือนของเดิม ดังนั้นถ้าโปรเซสของ ASP ล้มเหลว ตัวเซสชั่นก็ยังอยู่
- เซสชั่น อ็อบเจ็กต์ใน ASP.NET ให้การสนับสนุนการใช้ เซิร์ฟเวอร์หลายๆตัว(server farm) จากแนวคิดของ out-of-process ข้างต้น เราจึงสามารถทำให้ เซิร์ฟเวอร์หลายๆใช้ เซสชั่นเสตทร่วมกันได้
- เซสชั่นอ็อบเจ็กต์ สามารถเรียกใช้โดยไม่ผ่าน คุกกี้ ได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การจัดการของเซสชั่นนั้น จะบอกได้จาก เลขเซสชั่นไอดี ซึ่งเป็นเลข 128 บิต หรือเรารู้จักกันดีว่า GUID's(Globally Unique Identifier) ซึ่งถูกสร้างขึ้นจากอัลกอริทึมที่มั่นใจว่าจะไม่เกิดการซ้ำกัน เลขเซสชั่นนี้จะถูกจัดเก็บในสองรูปแบบ คือ

1. ผ่านคุกกี้ ซึ่งคุกกี้ นั้นสามารถบ่งบอกถึงเซสชั่นของไคลเอนต์
2. จากการรวมค่าของไอดี เข้าไปยัง URL โดยตรง

เว็บเซอร์วิสนั้นก็ต้องการตัวในการจัดการเสตทกับไคลเอนต์ เครื่องไคลเอนต์ที่จะเรียกใช้เว็บเซอร์วิสต้องสนับสนุนการใช้งานคุกกี้ด้วย เพื่อที่จะเก็บ เซสชั่นไอดี

ตัวอย่างการใช้งาน เซสชั่นอ็อบเจ็กต์ใน ASP.NET

```
Session["UserName"] = "John Smith" ;
```

6.4.3 โหมดการจัดการเซสชั่นและการคอนฟิกค่า

การคอนฟิกค่าต่างๆใน ASP.NET แบ่งออกได้เป็นสองส่วนด้วยกัน คือ machine.config และ web.config ไฟล์แรกนั้นจะส่งผลกับทุกเว็บแอพพลิเคชันในเครื่อง และ ไฟล์หลังจะเป็นการคอนฟิกเฉพาะแอพพลิเคชันซึ่งจะทับแมชชีนคอนฟิกอีกที

โหมดในการใช้งานเซสชั่นนั้นมีด้วยกัน 3 โหมด ได้แก่

1. In Proc เป็นค่าเริ่มต้นที่ติดตั้ง คือจะจัดเก็บเซสชั่นในโปรเซสเดียวกับเว็บเซิร์ฟเวอร์
2. State Server ทำการจัดเก็บลงในแบบ out-of-process ลงในแมมโมรี่ โดยเครื่องที่จะทำเซิร์ฟเวอร์นี้สามารถเป็นเครื่องเดียวกับ เว็บเซิร์ฟเวอร์ได้

   ```
   stateConnectionString="tcpip=127.0.0.1:42424"
   ```

3. SqlServer ทำการจัดเก็บลงในแบบ out-of-process เช่นกัน แต่จัดเก็บลงในดาต้าเบส

   ```
   sqlConnectionString="data source=127.0.0.1;user id=sa;password="
   ```

ค่าอื่นๆที่สามารถปรับแต่งได้ก็คือ โหมดของการใช้ คุกกี้ ว่าจะเป็นแบบ ใช้คุกกี้ หรือว่า ผ่าน url และการกำหนดค่าช่วงเวลาของคุกกี้

```
cookieless="true"
timeout="40"
```

ในเรื่องของประสิทธิภาพ โหมด In Proc มีประสิทธิภาพมากกว่า State Server และ SqlServer ตามลำดับ แต่ สองโหมดหลังจะสามารถใช้กับแบบ เซิร์ฟเวอร์ฟาร์มได้ โหมด SqlServer ก็จะช้าที่สุดแต่ก็มีความน่าเชื่อถือที่สุดเช่นกัน

6.4.4 การเรียกใช้งาน แอพพลิเคชัน และ เซสชั่นเสตทใน เว็บเซอร์วิส

เว็บเซอร์วิสเป็นส่วนหนึ่งของ ASP.NET จึงสามารถใช้งานในส่วนเสตทนี้ทำให้เป็น เสตทฟูล เว็บเซอร์วิสได้ ซึ่งการใช้งานนั้นต้องทำการเซ็ตค่าของเว็บเมธอดเพื่อให้จำค่าเซสชั่นได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
[WebMethod(EnableSession:=True)]
```

ในกรณีที่เมธอดนั้นมีการใช้งาน แอพพลิเคชันอ็อบเจ็กต์เท่านั้น ไม่จำเป็นที่ต้องกำหนดค่าดังกล่าว

การเขียนแอพพลิเคชัน เรียกใช้งานเว็บเซอร์วิส

การเรียกใช้งานนั้นอย่างที่บอกไปแล้ว ว่าต้องใช้คุกกี้เพื่อให้จำเลขเซสชั่น ไอดีได้

1. ทำการสร้างเว็บเซอร์วิสอ็อบเจ็กต์ขึ้นมา

```
localhost.StatefulService objService = new localhost.StatefulService();
```

2. สร้าง CookieContainer ซึ่งต้อง using System.Net ด้วย

```
CookieContainer objCookie = new CookieContainer();
```

3. ทำการกำหนดค่าให้แก่ เว็บเซอร์วิสอ็อบเจ็กต์

```
objService.CookieContainer = objCookie;
```

ถ้าเป็นในส่วนของเว็บฟอร์มนั้น ต้องมีการกำหนดค่า

```
if(Session["Container"] == null)
{
    objContainer = new CookieContainer(); // first call
    Session["Container"]=objContainer;
}
else
    objContainer = (CookieContainer)Session["Container"]; // all subsequent

callwb.CookieContainer = objContainer;
```

เพื่อให้ทุกครั้งที่มีการโหลดเพจใหม่ขึ้นมา ยังจำค่าคุกกี้เดิมได้

6.5 ทรานแซกชัน

6.5.1 เหตุผลที่ต้องใช้ทรานแซกชัน

ตัวอย่างปัญหาที่ต้องใช้ทรานแซกชันในการแก้

- **Atomic Operation**

กรณีที่ต้องการทำคำสั่งหลาย ๆ คำสั่ง โดยมองทั้งหมดเป็นเหมือนคำสั่งเดียว (atomic operation) เช่น กรณีบัญชีธนาคาร เมื่อต้องการโอนเงินจากบัญชีหนึ่งไปยังอีกบัญชีหนึ่ง ต้องถอนเงินออกจากบัญชีหนึ่ง จากนั้นจึงฝากเงินเข้าอีกบัญชีหนึ่ง ถ้ามีคำสั่งใดคำสั่งหนึ่งล้มเหลว ต้องให้คำสั่งทั้งคู่ล้มเหลว ไม่เช่นนั้นยอดเงินคงเหลือในบัญชีจะไม่ถูกต้องจึงต้องไม่เกิดสถานการณ์ที่คำสั่งหนึ่งทำงานสำเร็จ ในขณะที่อีกคำสั่งทำงานล้มเหลว เนื่องจากทั้งสองคำสั่งนั้นเป็นทรานแซกชันเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การแก้ปัญหานี้วิธีหนึ่งคือการใช้ exception ดังตัวอย่างต่อไปนี้

```
try {
// Withdraw funds from account 1
}
catch (Exception e) {
// If an error occurred, do not proceed.
return;
}
try {
// Otherwise, deposit funds into account 2
}
catch (Exception e) {
// If an error occurred, do not proceed,
// and redeposit the funds back into account 1.
return;
}
```

ตัวอย่างโค้ดข้างต้น ถอนเงินจากบัญชีที่ 1 ก่อน หากเกิดข้อผิดพลาดขึ้น แอพพลิเคชันจะจบการทำงาน และไม่มีผลของการทำงานใด ๆ เกิดขึ้น หากไม่เกิดข้อผิดพลาดก็จะฝากเงินเข้าไปยังบัญชีที่ 2 ในขณะฝากเงินนั้น หากเกิดข้อผิดพลาด จะต้องทำการฝากเงินกลับเข้าไปในบัญชีที่ 1 ใหม่ จากนั้นจึงจบการทำงาน

ปัญหาที่เกิดขึ้นจากการแก้ปัญหาด้วยวิธีนี้คือ

- โปรแกรมมีความซับซ้อน
- ต้องตรวจสอบทุกความเป็นไปได้ที่จะทำให้เกิดการทำงานผิดพลาดในทุก ๆ ขั้นตอน และต้องเขียนโค้ดเพื่อยกเลิกการเปลี่ยนแปลงที่ได้ทำไปก่อนที่จะเกิดความผิดพลาดนั้นเอง
- ในการตรวจสอบความผิดพลาด แทบจะไม่สามารถควบคุมได้หากกระบวนการทั้งหมดมีความซับซ้อน เช่นการปรับปรุงข้อมูลทางการเงิน ที่ต้องอาศัยการทำงานทั้งหมด 10 ขั้นตอนด้วยกัน จะต้องหาวิธีการในการรองรับการทำงานที่ผิดพลาดในทุกขั้นตอนที่เป็นไปได้
- การตรวจสอบการทำงานได้ยาก เพราะต้องทำการจำลองการทำงานที่ผิดพลาดที่เกิดขึ้นได้ในทุกขั้นตอน

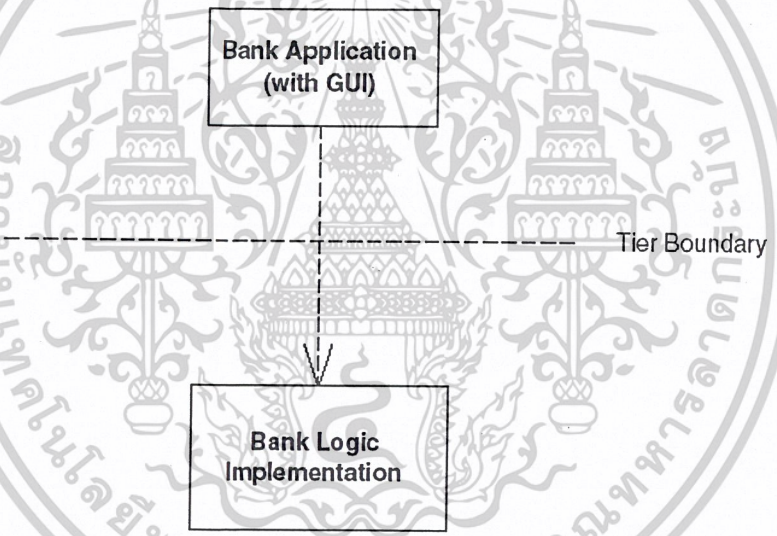
ในทางอุดมคตินั้นต้องการวิธีการที่สามารถทำคำสั่งทั้งสองได้ โดยมองเป็นคำสั่งใหญ่เพียงคำสั่งเดียว และรับประกันว่าคำสั่งทั้งสองจะต้องทำงานสำเร็จทั้งคู่ หรือทำงานผิดพลาดทั้งคู่เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **การทำงานผิดพลาดของเครื่องคอมพิวเตอร์ หรือระบบเครือข่าย**

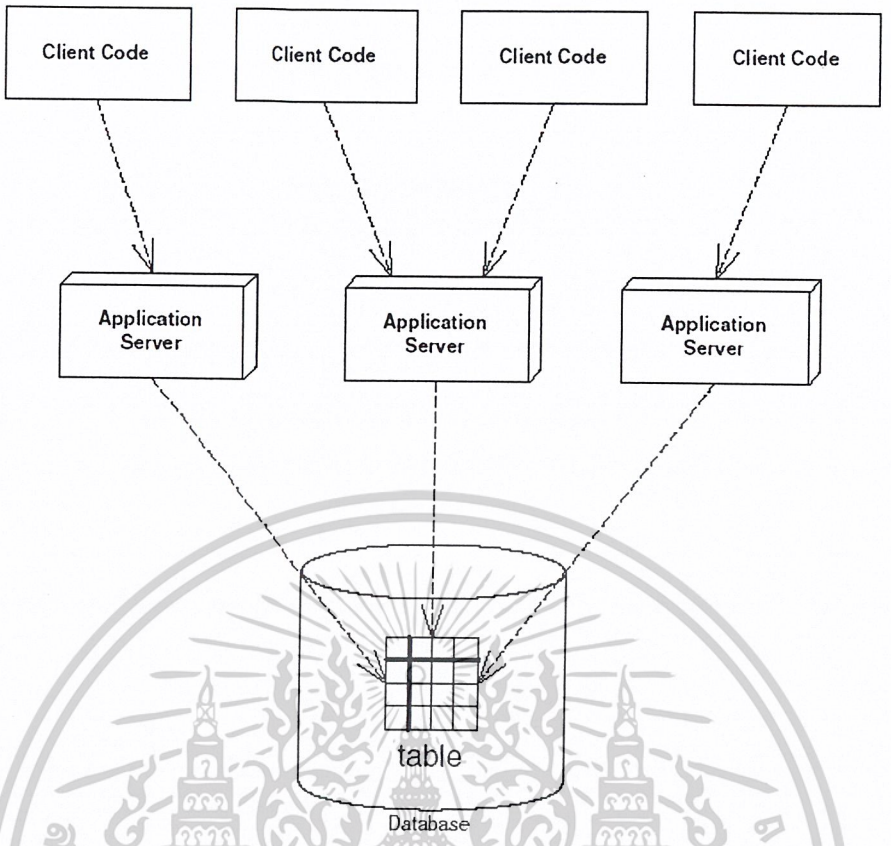
จากตัวอย่างข้างต้น ถ้าลอจิกต่าง ๆ ที่ทำงานกับบัญชีธนาคารถูกกระจายบนเครือข่ายโดยดีพลอยแบบมัลติเทียร์ เพื่อเหตุผลทางด้านความปลอดภัย , การเพิ่มขยาย หรือประโยชน์จากการแบ่งโปรแกรมออกเป็นโมดูล ในการดีพลอยแบบมัลติเทียร์ ไคลเอนต์ที่ต้องการใช้งานแอพพลิเคชันบัญชีธนาคารนี้จะต้องกระทำผ่านทาง remote method invocation แสดงในรูปที่ 6-2 ทำให้ต้องคำนึงถึงเรื่องของการทำงานผิดพลาดและความน่าเชื่อถือ เช่น ระบบเครือข่ายแครชระหว่างที่โปรแกรมบัญชีธนาคารกำลังทำคำสั่งใด ๆ อยู่ แม้ว่าเมื่อระบบเครือข่ายแครชจะมีการสร้าง exception ขึ้นมา แต่การแครชของระบบเครือข่ายนั้นอาจจะเกิดขึ้นก่อนหน้าการถอนเงิน หรืออาจเกิดขึ้นภายหลังการถอนเงินก็ได้ ซึ่งไม่อาจแยกความแตกต่างระหว่าง exception ที่เกิดขึ้นจากกรณีทั้งสองนี้ได้เลย

ในความเป็นจริง ระบบเครือข่ายไม่ได้เป็นจุดเดียวที่ทำให้เกิดปัญหานี้ขึ้น เนื่องจากข้อมูลบัญชีธนาคารก็คือข้อมูลแบบ persistent ที่อยู่ในฐานข้อมูล มีความเป็นไปได้เสมอที่ระบบฐานข้อมูลจะแครชหรือเครื่องคอมพิวเตอร์ที่ดีพลอยระบบฐานข้อมูลไว้จะเกิดแครช ถ้าหากการแครชนั้นเกิดขึ้นในระหว่างที่กำลังเขียนข้อมูลลงไปในฐานข้อมูล อาจทำให้ฐานข้อมูลอยู่ในสถานะที่ไม่ consistent ได้



รูปที่ 6-2 แอพพลิเคชันที่ทำงานในระบบแบบกระจาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-3 การใช้งานฐานข้อมูลพร้อม ๆ กัน

- **การที่ผู้ใช้หลายคนสามารถใช้งานข้อมูลเดียวกันพร้อมกันได้**

ในระบบแบบ enterprise-level distributed system เป็นรูปแบบที่ไคลเอนต์หลาย ๆ ตัวติดต่อกับแอพพลิเคชันเซิร์ฟเวอร์หลาย ๆ ตัว ถ้าแอพพลิเคชันเซิร์ฟเวอร์ทั้งหมดใช้ฐานข้อมูลเดียวกัน (ดังรูปที่ 6-3) เซิร์ฟเวอร์สามารถที่จะแก้ไขข้อมูลตัวเดียวกันที่อยู่ในฐานข้อมูลได้

ถ้ามีไคลเอนต์สองตัวพยายามแก้ไขข้อมูลเดียวกันที่อยู่ในฐานข้อมูลในเวลาเดียวกันแล้ว คำสั่งของไคลเอนต์ทั้งสองอาจทำงานซ้อนกัน และได้ผลลัพธ์สุดท้ายผิดพลาด ซึ่งการมีข้อมูลที่ไม่ถูกต้องในฐานข้อมูล ในระบบทั่วไปถือว่าเป็นสิ่งที่ยอมรับไม่ได้ (unacceptable) และอาจสร้างความเสียหายอย่างมากให้กับองค์กรดังนั้นจึงจำเป็นต้องหาวิธีการที่จะจัดการกับการแก้ไขข้อมูลพร้อมกันของไคลเอนต์หลาย ๆ ตัว โดยจะต้องไม่ทำให้ผลลัพธ์สุดท้ายที่ได้นั้นเป็นผลลัพธ์ที่ไม่ถูกต้อง

6.5.2 ประโยชน์ของทรานแซกชัน

ทรานแซกชัน คือเซ็ตของคำสั่งที่ถูกเอ็กซิคิวต์โดยมองว่าเป็นคำสั่งใหญ่เพียงอันเดียว ทรานแซกชันรับประกันว่าคำสั่งทั้งหมดทำงานสำเร็จ มิฉะนั้นจะต้องไม่มีผลการทำงานจากคำสั่งใดเกิดขึ้นเลย ทรานแซกชันสามารถแก้ปัญหาที่เกิดจากการทำงานผิดพลาดของระบบเครือข่ายได้ อนุญาตให้ไคลเอนต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลาย ๆ ตัวใช้งานข้อมูลเดียวกันพร้อมกันได้ และรับประกันว่าข้อมูลทั้งหมดที่คำสั่งเหล่านี้ทำการแก้ไขจะถูกต้องสมบูรณ์ โดยไม่ถูกรบกวนจากการแก้ไขข้อมูลโดยไคลเอนต์อื่น ๆ

การใช้ทรานแซกชันอย่างถูกต้องทำให้สามารถมีหลายไคลเอนต์ทำงานกับฐานข้อมูลได้โดยไม่ขึ้นต่อกัน เช่น ถ้ามีไคลเอนต์สองตัวอ่านและเขียนข้อมูลลงในฐานข้อมูล ไคลเอนต์ทั้งสองจะ mutual exclusiveกันระบบจัดการฐานข้อมูลจะจัดการเกี่ยวกับการควบคุมการทำงานพร้อมกันที่จำเป็นบนฐานข้อมูล เช่น การล็อคข้อมูล เพื่อไม่ให้การทำงานของแต่ละเธรดมีผลกระทบซึ่งกันและกัน

6.5.2.1 ACID Properties

เมื่อใช้งานทรานแซกชันอย่างถูกต้อง คำสั่งจะถูกเอ็กซิคิวต์ โดยมีการรับประกันผลการทำงานทั้งสิ้น 4 อย่างด้วยกัน เรียกว่า ACID properties โดยคำว่า ACID ย่อมาจาก Atomicity , Consistency , Isolation และ Durability

- **Atomicity** - รับประกันว่าคำสั่งทั้งหมดจะถูกรวมเข้าไว้ด้วยกัน และมองเป็นหน่วยของการทำงานเพียงอันเดียว การทำงานของทรานแซกชันจะเป็นไปในลักษณะของ All-or-Nothing คือ การอัพเดตข้อมูลในฐานข้อมูลทั้งหมดจะต้องเสร็จสิ้น หรือถ้าเกิดข้อผิดพลาดใด ๆ ขึ้นระหว่างการทำงาน การอัพเดตฐานข้อมูลทั้งหมดนั้นจะต้องไม่เกิดขึ้นเลย โดยการอัพเดตฐานข้อมูลที่เกิดขึ้นก่อนแล้ว จะต้องถูกยกเลิก
- **Consistency** - รับประกันว่าทรานแซกชันจะต้องทำให้ระบบอยู่ในสถานะที่ consistent เมื่อทรานแซกชันทำงานเสร็จเรียบร้อย เช่น การโอนเงินจะต้องไม่มีเงินหายไปจากระบบ แต่ในระหว่างการทำงานของทรานแซกชันนี้ ขณะที่เงินถูกถอนออกมาจากบัญชีแรก ข้อมูลอยู่ในสถานะที่ไม่ consistent แต่เนื่องจากทรานแซกชันจะมองเห็นคำสั่งทั้งหมดเปรียบเสมือนเป็นคำสั่งเดียว ดังนั้นถ้าทรานแซกชันสำเร็จก็คือคำสั่งทั้งหมดจะต้องสำเร็จ แต่ถ้าทรานแซกชันทำงานไม่สำเร็จคำสั่งทั้งหมดก็จะต้องไม่สำเร็จ และการเปลี่ยนแปลงใด ๆ ที่ได้กระทำไประหว่างนั้นจะต้องถูกยกเลิก ตามคุณสมบัติข้อแรกของทรานแซกชัน ดังนั้นข้อมูลจะอยู่ในสถานะที่ consistent เสมอ
- **Isolation** - เป็นการป้องกันทรานแซกชันที่ทำงานพร้อมกัน ไม่ให้เห็นผลลัพธ์ที่ยังไม่สมบูรณ์ที่เกิดจากทรานแซกชันอื่น isolation อนุญาตให้หลายทรานแซกชันสามารถอ่านหรือเขียนฐานข้อมูลโดยไม่รู้ถึงการมีอยู่ของทรานแซกชันอื่น ถ้าไม่มีคุณสมบัติ isolation สถานะของข้อมูลอาจจะอยู่ในสถานะที่ไม่ consistent ได้ ซึ่งมีประโยชน์มากในกรณีไคลเอนต์หลายตัวแก้ไขฐานข้อมูลในเวลาเดียวกัน โดยแต่ละไคลเอนต์จะทำงานโดยเปรียบเสมือนว่ามีเพียงตัวมันเองเท่านั้นที่กำลังใช้งานฐานข้อมูลอยู่ ทรานแซกชันสร้างคุณสมบัติ isolation โดยการใช้ synchronization โพรโตคอลต่าง ๆ กับข้อมูลในฐานข้อมูล เช่น ระหว่างการทำงานของทรานแซกชัน ข้อมูลจะถูกล็อคและปลดล็อคโดยอัตโนมัติตามความจำเป็น และการอัพเดตข้อมูลของทรานแซกชันอื่น จะไม่มีผลกับข้อมูลที่ทรานแซกชันนี้กำลังทำงานอยู่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **Durability** - รับประกันว่าการเปลี่ยนแปลงข้อมูลใด ๆ ในฐานข้อมูลที่เสร็จสิ้น (commit) แล้ว จะต้องมีความถาวร คือต้องไม่สูญหายเมื่อเกิดความผิดพลาดใด ๆ ขึ้น เช่น เครื่องคอมพิวเตอร์แครช ,ระบบเครือข่ายแครช , ฮาร์ดดิสก์แครช หรือไฟฟ้าดับ โดยต้องมีการเก็บ log file ของการอัพเดตข้อมูลใด ๆ ในทรานแซกชัน เมื่อระบบเกิดแครชขึ้น ข้อมูลสามารถสร้างขึ้นมาใหม่โดยการทำงานตามลำดับใน log file อีกครั้งหนึ่ง

6.5.3 ทรานแซกชันโมเดล

ทรานแซกชันโมเดลเป็นวิธีการที่ต่างกันในการทำทรานแซกชัน มีหลายโมเดลที่ใช้จัดการทรานแซกชัน แต่ละโมเดลจะมีความซับซ้อนและฟีเจอร์ที่ต่างกัน 2 โมเดลที่ได้รับความนิยมมากที่สุดคือ flat transaction และ nested transaction

- **Flat Transactions**

flat transaction เป็นทรานแซกชันโมเดลที่เข้าใจได้ง่าย flat transaction เป็นกลุ่มของ operation ที่ทำงานเป็นหน่วยเดียว ถ้า operation ไม่สำเร็จทั้งหมดจะเกิด rollback ขึ้น แต่ถ้าสำเร็จจะเกิด commit กระบวนการของ flat transaction ดังรูปที่ 6-4

- **Nested Transactions**

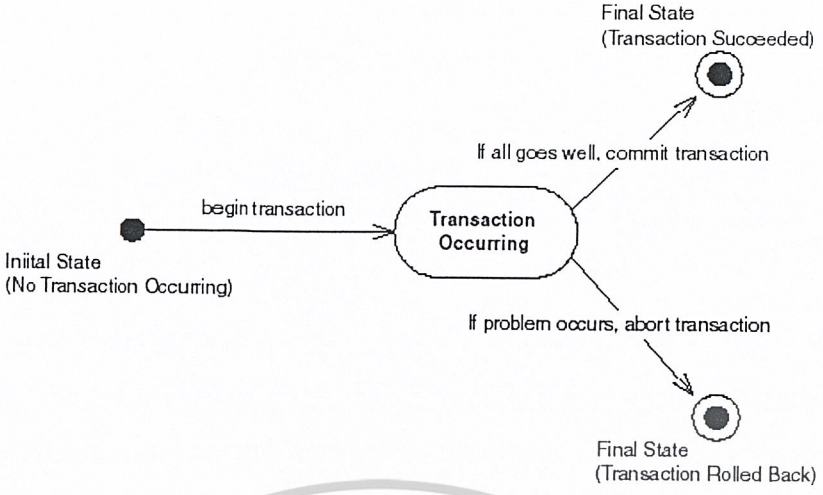
ตัวอย่างของงานที่ต้องใช้ nested transaction คือปัญหา trip-planning โดยแอพพลิเคชันจะมี operation ดังนี้

1. ซื้อตั๋วรถไฟจากบอสตันไปยังนิวยอร์ค
2. ซื้อตั๋วเครื่องบินจากนิวยอร์คไปยังลอนดอน
3. ซื้อตั๋วบอลลูนจากลอนดอนไปยังปารีส
4. แอพพลิเคชันพบว่าไม่มีไฟลท์ไปยังปารีส

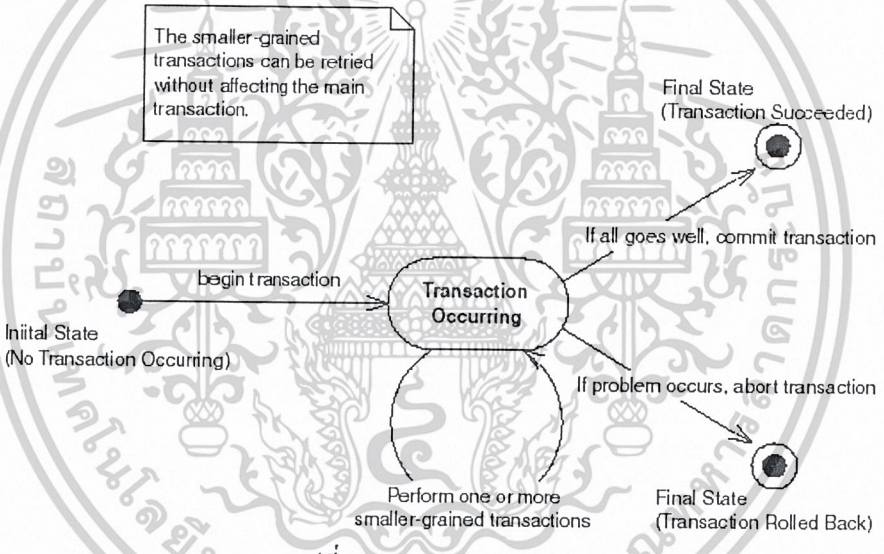
ถ้าลำดับของโอเปอร์เรชั่น (operation) นี้ทำกับ flat transaction จะเกิดโรเบ็ก (rollback) เท่านั้น เนื่องจากไม่มีไฟลท์ไปยังปารีส แอพพลิเคชันจะต้องไม่เก็บการจองใด ๆ ไว้เลย แต่ตรงจุดนี้อาจมีทางในการเปลี่ยนการเดินทางด้วยบอลลูนไปเป็นทางอื่น ทำให้เก็บข้อมูลการจองไว้ได้

nested transaction ช่วยในการแก้ปัญหานี้ มันจะยอมให้เกิดทรานแซกชันซ้อนกันได้ รูปที่ 6-5 แสดงแนวคิดของ nested transaction

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6-4 Flat Transaction



รูปที่ 6-5 Nested Transaction

6.5.4 Transactional Isolation

คุณสมบัติ isolation รับประกันว่า ผู้ใช้ที่ทำงานอยู่พร้อม ๆ กันจะไม่รู้สึกและได้รับผลกระทบจากผู้ใช้คนอื่น ๆ แม้ว่าผู้ใช้ทั้งหมดนั้นกำลังทำงานกับข้อมูลเดียวกันในฐานข้อมูลก็ตาม โดยสามารถระบุระดับของ isolation ที่ต้องการได้

การระบุระดับของ isolation สามารถควบคุมได้ว่าจะให้แต่ละทรานแซกชัน isolate จากกันมากน้อยเพียงใด ถ้าตั้งระดับของ isolation ที่สูง ก็มั่นใจได้ว่าแต่ละทรานแซกชันนั้น จะถูก isolate ออกจากทรานแซกชันอื่น ๆ ทั้งหมด แต่การตั้งระดับของ isolation สูงก็ทำให้ประสิทธิภาพในการทำงานที่ได้ลดลง เนื่องจากในการทำ isolation นั้นใช้วิธีการล็อคค่าในฐานข้อมูลเป็นหลัก แต่ถ้าตั้งระดับของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

isolation ไว้ต่ำ ทรานแซกชันก็จะไม่ isolate แต่ประสิทธิภาพในการทำงานร่วมกันของหลาย ๆ ทรานแซกชันจะดีขึ้น

ดังนั้น ต้องพิจารณาให้รอบคอบว่า ต้องการความเป็น isolation สูงต่ำเพียงใดในการทำงาน การกำหนด transaction isolation มีอยู่ทั้งหมด 4 ระดับด้วยกัน ดังนี้

- **READ UNCOMMITTED**

เป็นระดับการทำงานที่ต่ำที่สุด ไม่มีการรับประกันคุณสมบัติ isolation เลย แต่จะให้ประสิทธิภาพในการทำงานสูงสุด การทำงานในระดับนี้อาจเกิดปัญหา dirty read ขึ้นได้ โดยหากทรานแซกชันหนึ่งทำงานพร้อมกันไปกับทรานแซกชันอื่น แล้วอีกทรานแซกชันนั้นเขียนข้อมูลโดยยังไม่ได้ commit ทรานแซกชันเดิมจะอ่านค่านั้นเข้ามาได้ทำให้ได้ค่าที่ผิดพลาดไป โดยไม่ขึ้นกับระดับของ isolation ที่อีกทรานแซกชันหนึ่งใช้งานอยู่ นอกจากนี้ยังเกิดปัญหา unrepeatable read และ phantom phenomenon ได้อีกด้วย

- **READ COMMITTED**

การทำงานในระดับนี้คล้ายกับระดับ READ UNCOMMITTED แตกต่างกันที่ในระดับนี้โค้ดจะอ่านค่าข้อมูลที่ commit แล้วเท่านั้น จะไม่อ่านค่าที่ถูกเขียนลงไปยังฐานข้อมูลแล้วแต่ยังไม่ commit ดังนั้นระดับของ isolation ระดับนี้จึงสามารถแก้ปัญหา dirty read ได้ แต่ยังไม่สามารถแก้ปัญหาที่ซับซ้อนเช่นปัญหา unrepeatable read และ phantom phenomenon

- **REPEATABLE READ**

เป็นการรับประกันที่สูงขึ้นไปจาก READ COMMITTED เมื่อใดก็ตามที่อ่านค่าข้อมูลที่คอมมิท (commit) แล้วจากฐานข้อมูล สามารถจะอ่านค่าข้อมูลนั้นซ้ำได้ และค่าที่ได้จะเป็นค่าที่เหมือนกับค่าที่เราอ่านขึ้นมาได้ครั้งแรก ซึ่งระดับของ isolation ที่ READ COMMITTED หรือต่ำกว่า ทรานแซกชันอื่นที่ทำงานอยู่พร้อมกันอาจจะคอมมิทข้อมูลในระหว่างการอ่านข้อมูลแต่ละครั้งได้ ทำให้ค่าของข้อมูลเดียวกันที่อ่านได้ในแต่ละครั้งไม่ตรงกัน การทำงานในโหมดนี้แก้ปัญหาที่เรียกว่า unrepeatable read

- **SERIALIZABLE**

สามารถแก้ปัญหา phantom phenomenon คือการที่เซตของข้อมูลใหม่ถูกเพิ่มเข้ามาในฐานข้อมูลระหว่างการอ่านข้อมูลของทรานแซกชัน รวมถึงแก้ปัญหาอื่น ๆ ที่กล่าวมาก่อนหน้านี้ทั้งหมด รับประกันว่าทุก ๆ ทรานแซกชันจะทำงานเป็นลำดับกับทรานแซกชันอื่น ๆ และมีคุณสมบัติ ACID properties ของฐานข้อมูลครบถ้วน หมายความว่าแต่ละทรานแซกชันจะทำงานแยกกับทรานแซกชันอื่น ๆ อย่างแท้จริง ตารางที่ 5-1 จะแสดงถึงปัญหาที่อาจเกิดในแต่ละระดับของ isolation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ISOLATION LEVEL	DIRTY READ?	UNREPEATABLE READS?	PHANTOM READ?
READ UNCOMMITTED	Yes	Yes	Yes
READ COMMITTED	No	Yes	Yes
REPEATABLE READ	No	No	Yes
SERIALIZABLE	No	No	No

ตารางที่ 6-1ระดับของ Isolation

การกำหนด isolation ใน EJB ในกรณีของ bean-managed persistent จะกำหนด isolation โดยใช้ resource manager API เช่น JDBC โดยการเรียก *java.sql.Connection.SetTransactionIsolation(...)* ส่วนในกรณีของ container-managed persistent จะไม่สามารถกำหนดระดับของ isolation ใน deployment descriptor ได้ การกำหนดระดับของ isolation จะขึ้นอยู่กับผลิตภัณฑ์ที่ใช้เป็นคอนเทนเนอร์หรือฐานข้อมูล

6.5.5 ทรานแซกชันแบบกระจาย

นอกจากทรานแซกชันพื้นฐาน คือแอพพลิเคชันเซิร์ฟเวอร์หนึ่งตัวเชื่อมต่อกับฐานข้อมูลหนึ่งตัวแล้ว ยังสามารถที่จะทำ distributed transaction ได้เช่นเดียวกัน โดยขึ้นอยู่กับการรองรับของแอพพลิเคชันเซิร์ฟเวอร์ ซึ่ง distributed transaction ยึดหลักการเดียวกับทรานแซกชันธรรมดา โดยที่หากมีคอมโพเน็นต์หนึ่งบนเครื่องคอมพิวเตอร์เครื่องหนึ่ง abort ทรานแซกชัน ทรานแซกชันทั้งหมดจะต้องถูก abort ด้วยเช่นกัน distributed transaction สามารถทำงานได้กับหลายแอพพลิเคชันเซิร์ฟเวอร์และหลายฐานข้อมูล หมายความว่าคอมโพเน็นต์ที่ดีพลอยในแอพพลิเคชันเซิร์ฟเวอร์หนึ่ง ทำงานในทรานแซกชันเดียวกันร่วมกับอีกคอมโพเน็นต์ซึ่งถูกดีพลอยในอีกแอพพลิเคชันเซิร์ฟเวอร์ ในการทำงานบิซิเนสเมธอดหนึ่ง ๆ เครื่องหลายเครื่องอาจจำเป็นต้องทำงานร่วมกัน distributed transaction จึงอนุญาตให้หลายแอพพลิเคชันเซิร์ฟเวอร์ซึ่งอาจเป็นซอฟต์แวร์จากผู้พัฒนาคนละรายกันทำงานร่วมกันในทรานแซกชันได้

6.5.6 ทรานแซกชันใน .NET

ทรานแซกชัน มีทั้งแบบโลคอล ซึ่งเป็นทรานแซกชัน ซึ่งมีขอบเขตการจัดการภายใต้แหล่งข้อมูลตัวเดียวกัน (single- transaction-aware data resource) หรือเรียกว่า internal transaction และอีกแบบ เป็นทรานแซกชันแบบกระจาย (Distributed transaction) เป็นการจัดการทรานแซกชันจากหลากหลายแหล่งข้อมูลร่วมกัน เช่น SQL Server , MSMQ

6.5.6.1 การจัดการทรานแซกชันแบบจัดการด้วยตนเอง (Manual Transaction)

Manual transaction ให้เราสามารถควบคุมขอบเขตการทำ transaction ได้อย่างชัดแจ้ง (explicit) แต่ผลเสียก็คือ เราต้องจัดการขอบเขตของ transaction ให้ดีและ ที่สำคัญคือ เราต้องจัดการควบคุม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

distributed transaction เอง จึงมีงานมากมายให้เราทำ เราต้องงควบคุมทุกๆ connection และ resource เพื่อให้เป็นไปตาม ACID ของ transaction ตัวอย่างของ transaction หนึ่งคือ ADO.NET สามารถ กำหนด จุดเริ่มต้น, commit, rollback ของ transaction ได้เอง

6.5.6.2 การจัดการทรานแซกชันแบบการจัดการอัตโนมัติ (Automatic Transaction)

.Net มี MTS/COM+ ในการจัดการทรานแซกชันแบบการจัดการอัตโนมัติ ซึ่ง COM+ ใช้ Microsoft Distributed Transaction Coordinator (DTC) เป็นตัวจัดการทรานแซกชันทำให้ .NET แอพพลิเคชัน สามารถจัดการการใช้ทรัพยากรจาก ที่ต่างๆร่วมกันได้ เช่นจาก SQL SERVER , MSMQ , ORACLE ประสิทธิภาพของมันก็ขึ้นอยู่กับ DTC

ข้อดีคือ ง่ายในการเขียนโปรแกรม ไม่ต้องมากำหนดเองตรงๆ แต่ข้อเสียคือ ยากในการดูแลจัดการและเรามักจะละเลยไม่สังเกตถึงว่ามีการทำทรานแซกชันทั้งหมดฟังก์ชัน และในแบบแรกนั้น เราสามารถที่จะจัดการเกี่ยวกับ exception ที่ได้รับได้โดยตรง

ในแต่ละวิธีการจัดการทำทรานแซกชัน นั้นเป็นการแลกเปลี่ยนระหว่างประสิทธิภาพของแอพพลิเคชัน และ ความสามารถในการดูแลแอพพลิเคชัน

โดยสรุปรูปแบบการใช้งานทรานแซกชัน ใช้ Database Transaction ดูแลในเรื่องการจัดการทรานแซกชันใน สโตร์โพรซีเยอร์ (Stored procedure) Manual Transaction โดยการใช้ ADO.NET จะง่ายในการเขียน โค้ด และมีความยืดหยุ่นในการใช้ควบคุมทรานแซกชัน โดยทำได้โดยใช้คำสั่งภายนอกกำหนด begin transaction และ end transaction ส่วน Automatic Transaction จะมีประโยชน์มากกับงานประเภทที่มีการทำงานเกี่ยวกับทรานแซกชันจากหลายๆ แหล่งข้อมูลร่วมกัน

6.5.7 การจัดการทรานแซกชันใน เว็บเซอร์วิส

มีการจัดการทรานแซกชันในเว็บเซอร์วิสอยู่สองส่วนที่เราต้องจัดการ อันแรก คือ เราต้องการที่จะควบคุมทรานแซกชันเป็นส่วนหนึ่งของเว็บเซอร์วิสที่เราให้บริการ เนื่องจากเรามีการควบคุมระดับเน็ตเวิร์กเลเยอร์ภายใน ASP.NET แอพพลิเคชัน เราสามารถใช้ COM+ เข้ามาช่วยได้ ส่วนที่สอง เรามองไปยังการรวมไปถึงทรานแซกชันของแอพพลิเคชันที่เรียกใช้ เว็บเซอร์วิสนั้น ซึ่งไม่มีการควบคุมในระดับของเน็ตเวิร์กเลเยอร์ ทำให้เป็นปัญหาในเรื่องของ การควบคุมทรานแซกชันแบบกระจายขึ้น (Distributed Transaction Coordinator)

การเรียกใช้งานในเว็บเซอร์วิสนั้น ต้องทำการกำหนดค่า เว็บเมธอดแอททริบิวต์

```
[WebMethod(TransactionOption=TransactionOption.RequiresNew)]
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คลาสนั้นต้องทำการ inherit มาจาก System.EnterpriseServices.ServicedComponent เป็นการสั่งให้ทำงานใน COM+ แล้ว COM+ จะทำการติดต่อกับ DTC ในการสร้าง distributed transaction และจัดการ resource ทั้งหมดที่ติดต่อ โดยมีการกำหนดตัวเลือก ไว้ 5 แบบคือ

1. Disabled - อ็อบเจ็กต์ที่สร้างขึ้นจะไม่มีทรานแซกชัน ใน COM+ ทรานแซกชัน แต่สามารถติดต่อโดยตรงกับ DTC ในกรณีที่ต้องการการสนับสนุน
2. NotSupported - ไม่มีการทำทรานแซกชันเลย
3. Supported - ถ้าอ็อบเจ็กต์นั้นจะถูกรันในทรานแซกชัน context ของตัวที่สร้างมันขึ้นมาซึ่งมีการทำทรานแซกชัน , แต่ถ้าตัวมันเอง เป็นตัวเริ่ม (root - ไม่ได้ถูกใครสร้างมาอีกที) หรือ context ที่สร้างมัน ไม่ทำทรานแซกชัน มันก็จะไม่ทำทรานแซกชัน
4. Required - ถ้าอ็อบเจ็กต์นั้นจะถูกรันในทรานแซกชัน context ของตัวที่สร้างมันขึ้นมามีการทำทราแซกชัน , แต่ถ้าตัวมันเอง เป็นตัวเริ่ม (root - ไม่ได้ถูกใครสร้างมาอีกที) หรือ context ที่สร้างมัน ไม่ทำทรานแซกชัน มันก็จะทำทรานแซกชันใหม่
5. RequiresNew - บอกว่าอ็อบเจ็กต์ ต้องทำทรานแซกชัน ใหม่

ในการบ่งบอกถึง การ commit และ rollback นั้น เราทำได้โดยการ try /catch เพื่อตรวจจับ exception แต่เราก็สามารถให้มันทำการจัดการ commit ,abort ให้เองได้ด้วย การเรียกใช้ System.EnterpriseServices.AutoComplete

การกำหนดค่านี้ จะเป็นการกำหนดให้ไปเรียกใช้เซอร์วิสของคอมพลัส โดยอัตโนมัติ โดยมีโหมดให้เรียกใช้ TransactionOption เหมือนกับ การที่เขียนคอมโพเนนต์ที่ สืบทอดมาจาก EnterpriseServices ถ้ามี exception ขึ้นทรานแซกชันจะทำการ roll back โดยอัตโนมัติ แต่ถ้าการทำงานนั้นอยู่ภายใต้ try…catch ทรานแซกชันจะคอมมิท ยกเว้นแต่ เราจะทำการ SetAbort ด้วยตัวเอง โดยเรียกใช้ผ่านอ็อบเจ็กต์ ContextUtil

6.5.7.1 ตัวอย่างการใช้งาน เว็บเซอร์วิสที่มีการใช้ทรานแซกชัน

มี 2 ฟังก์ชันคือ UpdateDoctor ทำการอัพเดทสถานะของหมอ และ InsertLog เพื่อทำการเก็บบันทึกการเปลี่ยนแปลง และมีฟังก์ชันที่เป็นเว็บเมธอด ให้ใช้งานซึ่งจะไปเรียก 2 ฟังก์ชันดังกล่าว

โดยหลังจากทำการอัพเดทสถานะแล้วก็จะบันทึกการเปลี่ยนแปลงเก็บไว้ด้วย แต่จะมีการโยน exception ออกมาทำให้เกิดการ roll back ผลลัพธ์ที่ได้นั้นจะส่งผลให้ ฟังก์ชัน UpdateDoctor เกิดการ roll back ด้วยเช่นกัน

```
[WebMethod(TransactionOption=TransactionOption.RequiresNew)]
public void UpdateStatus(int DoctorID, int StatusID)
{
        UpdateDoctor(DoctorID, StatusID);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        InsertLog(DoctorID, StatusID);
    }
```

6.5.7.2 การเรียกใช้เว็บเซอร์วิสที่มีทรานแซกชัน

จากส่วนแรก การเรียกใช้งานก็เป็นไปตามปกติเหมือนการเรียกใช้เว็บเซอร์วิสโดยทั่วไป ในหัวข้อนี้จะพูดถึงการเรียกใช้เว็บเซอร์วิสให้เป็นส่วนหนึ่งของทรานแซกชันภายใต้คอนเท็กซ์ของไคลเอนต์แอพพลิเคชัน จากเอกสารของไมโครซอฟท์กล่าวว่า

"เว็บเซอร์วิสเมธอดแต่ละอันจะเป็นส่วนหนึ่งของทรานแซกชันของตัวเองเท่านั้น เนื่องจาก เว็บเซอร์วิสเมธอด สามารถเป็นได้เพียง root ของทรานแซกชัน เท่านั้น "

หมายความว่า เราไม่สามารถที่จะสร้างแอพพลิเคชันที่เรียกใช้เว็บเซอร์วิสในทรานแซกชันคอนเท็กซ์ในโลคอลเมธอดได้

ปัญหาที่เกิดขึ้นก็คือ สมมุติว่า เรามี ASP.NET แอพพลิเคชัน มีการเรียกใช้เว็บเซอร์วิสเมธอดสองตัวที่เป็นทรานแซกชันเมธอด โดยที่ ASP.NET เรามีการกำหนดให้เป็นทรานแซกชันคอนเท็กซ์ด้วย ถ้าเว็บเมธอดอันที่สองเกิดการ roll back เมธอดแรกจะไม่ roll back ด้วย

ถึงแม้เราจะการกำหนดให้เป็นทรานแซกชันทั้งระดับของโลคอลใน ASP.NET และระดับเว็บเซอร์วิสก็ตามแต่ก็ไม่สามารถแก้ปัญหานี้ได้เนื่องจาก เว็บเซอร์วิสไม่ได้เป็นส่วนหนึ่งของทรานแซกชันที่โลคอลแอพพลิเคชัน

6.5.7.3 ปัญหาในการเรียกใช้ทรานแซกชันเว็บเซอร์วิส

โดยตัวนิยามแล้ว เว็บเซอร์วิสแล้วไม่ตรงตามมาตรฐานของ ACID มันเป็น loosely coupled มันจึงไม่สามารถควบคุมได้ จากแต่เดิมใน DTC (distributed transaction control)ใช้ข้อมูลของเน็ตเวิร์กเลเยอร์ในการควบคุมความถูกต้องของทรานแซกชัน เนื่องจากทุกๆเน็ตเวิร์กทราฟฟิกของไมโครซอฟท์ ทรานแซกชันแบบกระจาย จะรายงานความถูกต้องผ่านเน็ตเวิร์กไปยังผู้เรียก โดยตัว DTC จะใช้ข้อมูลนี้ในการพิจารณาตัดสินใจ roll back ในกรณีที่ต้องทำ ถ้าเกิดมีการล้มเหลวระดับเน็ตเวิร์ก ตัว DTC จะรับรู้แล้วจัดการตามสิ่งที่เกิดขึ้น

เรามี WSDL ในการอธิบาย อินเทอร์เฟสของรีโมทเซอร์วิส ซึ่งไม่ได้อธิบายเกี่ยวกับการเรียกใช้งานกับ DTC ซึ่งก็เป็นความจริงที่ว่า คอมพลัสไม่ได้จัดการกับทั้งหมดของโลก .NET และไม่ได้เกี่ยวข้องกับ WSDL

จากเหตุผลที่กล่าวมาข้างต้น เราจังไม่สามารถที่จะรวมเว็บเซอร์วิสเข้ากับ ทรานแซกชันใน ASP.NET ได้ นี่เป็นอีกจุดอ่อนหนึ่งของเว็บเซอร์วิสโดยทั่วไป ไม่ได้เฉพาะการอิมพลีเมนต์เว็บเซอร์วิสของไมโครซอฟท์เท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.5.7.4 แนวทางการแก้ไขปัญหา

ในการพัฒนาแอพพลิเคชันที่มีการทำงานของทรานแซกชันด้วยเว็บเซอร์วิส ระบบจำเป็นต้องมี Transaction Monitor ซึ่งทำหน้าที่ในการดูแล และจัดการทรานแซกชันทั้งหมด (เช่น roll back ในกรณีที่มีปัญหาเกิดขึ้นกับทรานแซกชัน) แน่นอนว่า TM จะต้องทำงานร่วมกันกับแอพพลิเคชันอย่างใกล้ชิด เพื่อให้มีคุณสมบัติพิเศษ เช่น การล็อกข้อมูลที่ใช้งานร่วมกัน หรือการย้อนกลับเมธอดที่ทำงานไปแล้ว ในกรณีของการ roll back เป็นต้น ดังนั้น TM จึงจำเป็นต้องมีการทำงานกับระบบเว็บเซอร์วิสอย่างใกล้ชิดซึ่งส่งผลให้ความอิสระต่อกันในเชิงโครงสร้างของซอฟท์แวร์สูญหายไป

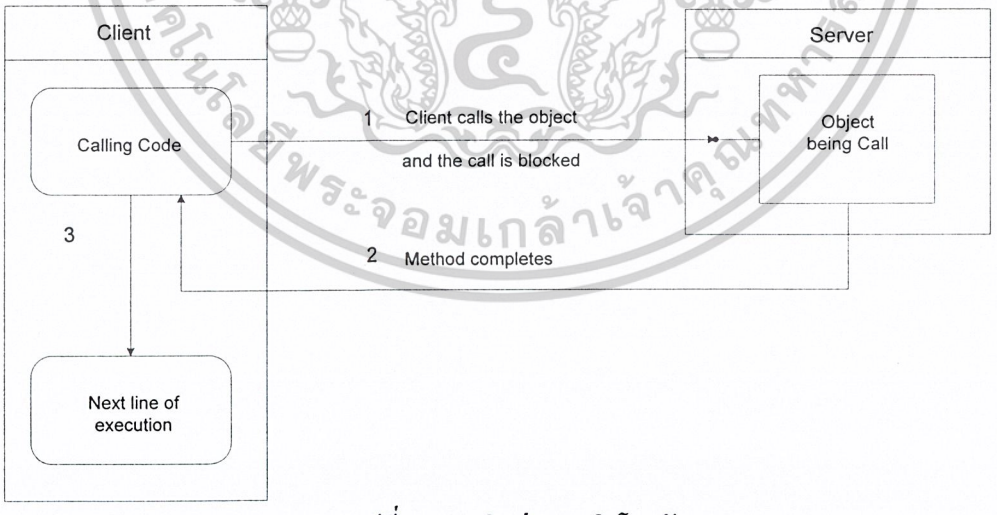
มีโซลูชั่นในการแก้ปัญหาที่ถูกนำเสนอออกมาใช้กับระบบงานแบบทรานแซกชันซึ่งผู้ที่เกี่ยวข้องกำลังพัฒนามาตรฐานสำหรับเว็บเซอร์วิส เพื่อให้รองรับระบบการจัดการทรานแซกชัน ตัวอย่างของมาตรฐานเหล่านี้ได้แก่ Transaction Authority Markup Language (XAML) และ Business Transaction Protocol (BTP) ซึ่งคาดว่าจะได้รับการแก้ไขโดยสมบูรณ์ราวปี 2547

อีกวิธีการแก้ปัญหาหนึ่งคือ สำหรับระบบที่มีการใช้งานภายในองค์กร ที่มีการใช้งาน .NET เพียงระบบเดียว เราก็สามารถเลือกใช้โปรโตคอลการสื่อสาร .NET Remoting ได้ ซึ่งถูกพัฒนามาจาก DCOM ซึ่งสามารถจัดการปัญหาดังกล่าวได้

6.6 การเรียกใช้เว็บเซอร์วิสแบบอซิงโครนัส(Asynchronous)

6.6.1 การประมวลผลแบบซิงโครนัส(Synchronous Processing)

การทำงานแบบซิงโครนัสนั้น ประกอบไปด้วย ฟังก์ชันและคอมโพเน็นต์ ที่มีการเรียกใช้งาน การเรียกใช้งานจะถูกบล็อกจนกว่าการทำงานของฟังก์ชันนั้นจะครบถ้วนเรียบร้อย รูปแบบการทำงานเป็นไปตามรูปที่ 6-6



รูปที่ 6-6 การติดต่อแบบซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.ไคลเอนต์ทำการเรียกใช้ ฟังก์ชันการทำงานไปยังอ็อบเจ็กต์คอมโพเน็นต์ การทำงานฝั่งไคลเอนต์ถูกบล็อก

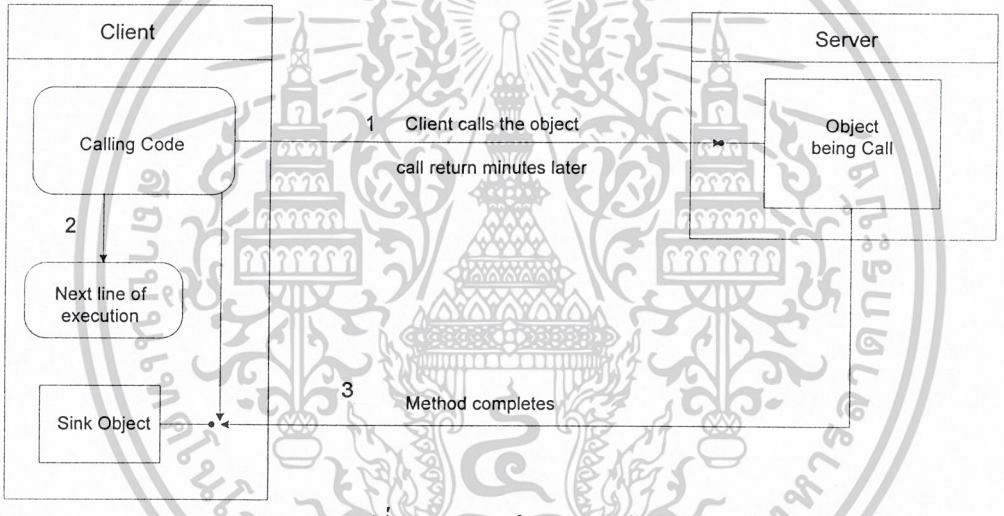
2.เมื่อฝั่งเซิร์ฟเวอร์ทำงานเสร็จส่งผลลัพธ์กลับมา

3.เมื่อได้รับผลลัพธ์ ไคลเอนต์ ทำการประมวลผลต่อได้

การเรียกใช้งานเว็บเซอร์วิส แบบซิงโครนัสเป็นการเรียกใช้งานตามแบบปกติทั่วไป ซึ่งง่ายในการจัดการ แต่มีข้อเสียคือเรื่องของประสิทธิภาพโดยฝั่งไคลเอนต์ต้องคอยรอผลลัพท์จากเว็บเซอร์วิส

6.6.2 การประมวลผลแบบอซิงโครนัส(Asynchronous Processing)

การเรียกใช้งานแบบอซิงโครนัส จะไม่มีการบล็อกจากคอมโพเน็นต์ฝั่งเซิร์ฟเวอร์ ไคลเอนต์จะรู้ว่าการประมวลผลเสร็จเรียบร้อยจาก polling หรือจาก การอินเทอร์รัพจากโปรแกรม โมเดลการทำงานแบบนี้ดังรูปที่ 6-7



รูปที่ 6-7 การติดต่อแบบอซิงโครนัส

1.ไคลเอนต์ทำการเรียกไปยังเซิร์ฟเวอร์

2.ไคลเอนต์สามารถที่จะประมวลผลโค้ดในบรรทัดต่อไปได้เลย โดยไม่ต้องรอผลลัพธ์

3.เมื่อฝั่งเซิร์ฟเวอร์ทำงานเสร็จจะส่งผลลัพธ์กับมาให้ ยังจุดที่สร้างขึ้นเพื่อรอรับผล(Sink point)

การเลือกโมเดลในการพัฒนาระบบ ไม่ได้เป็นเพียงเรื่องเล็กน้อย เราต้องทำการตัดสินใจจากความต้องการของระบบที่เราพัฒนาอย่างรอบคอบเพื่อตัดสินใจใช้งานการเรียกใช้แบบอซิงโครนัสหรือไม่ โดยอาจจะนำคำแนะนำต่อไปนี้เพื่อมาช่วยในการตัดสินใจ

- การเรียกใช้งานแบบอซิงโครนัส จะช่วยขยายการรองรับการใช้งานของระบบ(scalability) ถ้าไคลเอนต์เป็น ASP.NET แอพพลิเคชัน หรือเป็น เว็บเซอร์วิสก็ตาม เนื่องจากการเรียกใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แบบซิงโครนัสจะมีการบล็อกทำให้เทรดของ ASP.NET ที่ทำงานเกิดการรอ(stall) ส่งผลให้การร้องขออื่นๆมายังแอพพลิเคชัน ต้องเข้าคิว ส่งผลกระทบต่อประสิทธิภาพของระบบ การเรียกใช้แบบอซิงโครนัสแทนอย่างน้อยก็สามารถที่จะปลดปล่อยเทรด ASP.NET ที่ไม่ได้เรียกใช้เว็บเซอร์วิสในขณะนั้น

- การเรียกใช้งานแบบรีโมทโพรซีเยอร์คอลไปยังเว็บเซอร์วิสนั้นอาจจะใช้เวลานานก่อนการส่งผลลัพท์กลับ ในกรณีนี้การเรียกใช้แบบอซิงโครนัสจะมีประโยชน์ โดยฝั่งไคลเอนต์สามารถที่จะทำงานอย่างอื่นได้ ถ้าไม่จำเป็นต้องใช้ผลลัพธ์ในทันที
- ในกรณีที่ไคลเอนต์ต้องการเรียกใช้งานไปยังหลายเว็บเซอร์วิสพร้อมๆกันหลายที่ หรือที่เดียวกันแต่หลายฟังก์ชันการใช้งานแบบอซิงโครนัสจะเป็นประโยชน์มากเนื่องจากไคลเอนต์สามารถที่จะเรียกใช้งานไปยังทุกๆที่พร้อมๆกันได้เลย ไม่ต้องคอยรอผลลัพธ์ของแต่ละที่แล้วค่อยเรียกต่อแบบในกรณีของซิงโครนัส ทำให้เพิ่มประสิทธิภาพได้

6.6.3 การเรียกใช้แบบอซิงโครนัสใน .NET

.NET มีโมเดลการเรียกใช้งานแบบอซิงโครนัส มาให้พร้อมอยู่แล้วดังนี้

- .NET เฟรมเวิร์ก จะให้บริการเซอร์วิสที่จำเป็นในการรองรับโมเดลการใช้งานแบบอซิงโครนัส
- การเรียกใช้อยู่ที่โค้ดไคลเอนต์ เรียกใช้งานในฟังก์ชันที่จำเป็นต้องใช้
- ไม่จำเป็นที่อ็อบเจ็กต์ที่ถูกเรียกต้องทำการเขียนโค้ดเพิ่มเติมเพื่อรองรับการใช้งาน ตัว CLR จะทำการจัดการให้
- รันไทม์ CLR ให้บริการตรวจสอบเกี่ยวกับความปลอดภัยในเรื่องของชนิดตัวแปร โดยการเรียกใช้ ดีลิเกต(delegate) ซึ่งก็เหมือนกับ พอยน์เตอร์ไปยังฟังก์ชันที่มีส่วนของการอธิบายตัวเองด้วยซึ่งต่างจากแต่ก่อนใน C++ พอยน์เตอร์ไปยังฟังก์ชันจะไม่บอกจำนวนพารามิเตอร์ที่ส่งผ่าน ชนิด แขของตัวแปรพารามิเตอร์ และพอยน์เตอร์ฟังก์ชัน ก็เหมือนพอยน์เตอร์ทั่วๆไป (เป็น void)

6.6.4 การเรียกใช้งานเว็บเซอร์วิสแบบอซิงโครนัส

การเรียกใช้เว็บเซอร์วิสแบบอซิงโครนัส ทำให้เราสามารถไปทำงานอย่างอื่นได้โดยไม่ถูก block เพื่อที่จะรอผลของ services นั้นๆ มีประโยชน์ในกรณีที่ services นั้นมีการเวลาประมวลผลนาน หรือในกรณีที่เราเรียกใช้ services จากหลายที่ เราส่งผลไปพร้อมกันทั้ง 3 ที่ โดยไม่ต้องมานั่งรอผลทีละที่

การทำงาน จะเรียกใช้ฟังก์ชันโดยเติม Begin นำหน้า เช่นฟังก์ชัน Hello() จะเป็น BeginHelllo() โดยที่ฝั่งเว็บเซอร์วิสเราไม่ต้องเขียนโค้ดเพื่อรองรับเลยการเรียกใช้กำหนดตามความต้องการทางไคลเอนต์ เราสามารถจบการทำงาน asynchronous ได้หลายวิธี

1. ทำการจบโดยเรียกใช้ EndHello() ซึ่งจะจบก็ต่อเมื่อ services นั้นทำงานเสร็จ ไม่งั้นก็จะถูกบล็อกรอเช่นกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ตรวจสอบจาก flag IAsncResult.IsComplete ซึ่งจะมีค่าเป็น true เมื่อทำงานเสร็จสิ้น

ตัวอย่างการใช้งาน

```
IAsyncResult AsyncResult; //ประกาศตัวแปรแบบเพื่อรอรับผลของการเรียกใช้
AsyncResult = ObjectWS.BeginHello(); //เรียกใช้งานแบบอซิงโครนัส
while(AsyncResult.IsCompleted == false)
{
        //other job here
        // วนลูปทำการเช็คจากแฟล็กว่าเสร็จรึยัง
}
```

3. ในขณะเรียก BeginHello() ทำการใส่พารามิเตอร์ ดิลีเกต(delegate) ที่ต้องการให้ทำงาน เมื่อ เว็บเซอร์วิสทำงานเสร็จจะเรียกกลับมายังฟังก์ชันหรือดิลีเกตนั้นๆ

ขั้นตอนการใช้งาน

- ทำการประกาศ call back delegate โดยส่งค่าฟังก์ชันที่ต้องการให้เรียกกลับไปด้วย

```
AsyncCallback AsyncCallback = new AsyncCallback(MyCallBack);
```

- ฟังก์ชันที่จะทำการเรียกใช้ ต้องรับค่าพารามิเตอร์ IasyncResult ด้วย และภายในฟังก์ชันต้องมีการทำการเรียก END เพื่อจบการใช้งานฟังก์ชันที่เรียกไปยังเว็บเซอร์วิสด้วย

```
private void MyCallBack(System.IAsyncResult AsyncResult
```

- การเรียกใช้งาน Begin Hello() ต้องมีการส่งผ่านค่าพารามิเตอร์ดังนี้

```
AsyncResult = objWS.BeginHello(parameter1,AsyncCallback,asycState);
```

Parameter1 เป็นพารามิเตอร์ที่ฟังชัน Hello ต้องการอาจมีหลายพารามิเตอร์ก็ได้

AsyncCallback ค่าของ delegate ที่ต้องการให้มีการเรียกกลับเมื่อทำงานเสร็จ

AsycState ใช้ในการส่งข้อมูลเกี่ยวกับข้อมูลของการเรียกแบบอซิงโครนัสไปยัง delegate สำหรับตัวแปรสองตัวท้ายนั้น ถ้าไม่ต้องการใช้งานก็ทำการส่งค่า null ไปแทน

6.6.5 ปัญหาที่เกิดขึ้นกับการนำไปใช้งานจริง

จากการทำการทดสอบการใช้งานใน 3 รูปแบบข้างต้นสามารถทำได้ตามทฤษฎีในไคลเอนต์วินโดวส์ฟอร์ม แต่สำหรับไคลเอนต์ที่เป็น ASP.NET แอพพลิเคชัน ผลลัพธ์ที่ได้จากการทดลองผิดพลาดดังนี้

ผลการทดลองในรูปแบบของASP.NET เว็บฟอร์ม

1. แบบ END ฟังก์ชัน : ได้ผลลัพธ์ถูกต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ทำการเช็คผลจากแฟล็ก AsyncResult.IsCompleted : รอผลที่ได้นานมาก จนสุดท้ายไม่มีผลขึ้นแสดง
3. ทำการเรียกกลับมายัง delegate : ไม่สามารถ เรียกไปยังฟังก์ชัน delegate ที่ต้องการได้ ผลที่ได้คือหน้าจอเดิมที่ทำการเรียก

จากการเข้าไปค้นหาข้อมูลจากอินเทอร์เน็ตได้ข้อมูลดังนี้ สาเหตุที่ไม่สามารถทำงานในเว็บฟอร์มได้เนื่องจาก ตัวไคลเอนต์ซึ่งในที่นี้คือ เว็บเซิร์ฟเวอร์จะไม่ส่งผลลัพธ์กลับมาจนกว่า การเรียกใช้งานทั้งหมดจะเสร็จเรียบร้อย หมายถึงขั้นตอนตั้งแต่เว็บแอพพลิเคชันถูกเรียกใช้งาน, เรียกไปยังเว็บเซอร์วิส จนกระทั่งเว็บส่งผลลัพธ์กลับมา วิธีที่เดียวจะเรียกใช้แบบอซิงโครนัสได้คือ เรียกใช้โดยตรงจากไคลเอนต์ที่ไม่ผ่านเว็บเซิร์ฟเวอร์ ซึ่งสามารถทำได้โดยไคลเอนต์สคริปต์ เช่น java script ซึ่งต้องมีการกำหนด time out ให้นานพอสมควร

การเรียกใช้งานแบบอซิงโครนัสโดยไคลเอนต์เป็นจาวานั้นไม่สามารถทำได้เนื่องจากตัวจัดการการเรียกใช้แบบนี้ ฝังอยู่ในอินฟราสตรักเจอร์ของ CLR ใน .NET เราจึงไม่สามารถเรียกใช้ความสามารถนี้ในจาวาได้

เนื่องจากมันไม่สามรถเรียกใช้จากค่ายจาวาได้ รวมทั้งค่อนข้างยุ่งยากในการเรียกใช้งานกรณีที่ไคลเอนต์เป็นเว็บฟอร์ม ดังนั้นจึงไม่เหมาะในการนำมาพัฒนา เราจะทำการศึกษาการทำงานแบบอซิงโครนัส โดยใช้เทคโนโลยีของคิว (MSMQ) ต่อไป

การแก้ปัญหาอ้างอิงจาก

www.dotnet247.com/247reference/msgs/21/106614.aspx

www.dotnet247.com/247reference/msgs/24/123480.aspx

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 7

ADO.NET

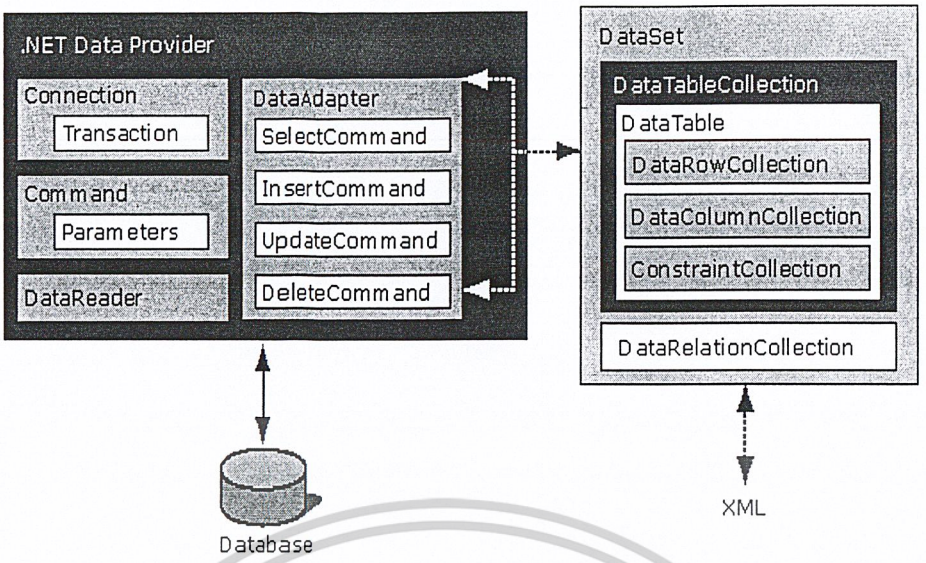
7.1 สถาปัตยกรรมของ ADO.NET

เนื่องจากในการพัฒนาโปรแกรมด้านฐานข้อมูลแต่เดิมนั้นได้ใช้เทคโนโลยี ADO (ActiveX Data Object) เป็นเครื่องมือในการติดต่อและจัดการข้อมูลในแบบ connected database คือจะเปิดการเชื่อมต่อกับฐานข้อมูลไว้ตลอดช่วงเวลาใช้งานซึ่งจะทำให้การเพิ่ม ลบ แก้ไขหรือการกระทำใดๆก็ตามที่เกี่ยวข้องกับข้อมูลจะเป็นการกระทำกับข้อมูลจริงในไฟล์ฐานข้อมูลเป็นจำนวนมากและพร้อมๆกันหรือการประมวลผลข้อมูลที่มีปริมาณมากจนทำให้ระยะเวลาที่ต้องเชื่อมต่อกับ server นานเกินไป ซึ่งจะทำให้อัตราการสิ้นเปลืองทรัพยากรบนเครื่อง server สูง ส่งผลให้ประสิทธิภาพโดยรวมของระบบลดลงได้ ดังนั้นใน Visual Studio .NET จึงได้นำเทคโนโลยีตัวใหม่ล่าสุดคือ ADO.NET ที่พัฒนาจาก ADO มาใช้จัดการข้อมูลแบบ disconnected database คือสามารถจัดการข้อมูลในฐานข้อมูลโดยไม่ต้องเปิดการเชื่อมต่อไว้ตลอดเวลาเหมือนใน ADO เนื่องจากข้อมูลจะถูกอ่านจากฐานข้อมูลมาเก็บไว้ในหน่วยความจำก่อน จากนั้นจะตัดการเชื่อมต่อกับฐานข้อมูลทำให้การกระทำใดๆหลังจากนี้จะเป็นการกระทำกับข้อมูลในหน่วยความจำเท่านั้น เมื่อเสร็จสิ้นกระบวนการแล้วจึงจะบันทึกกลับลงฐานข้อมูล ซึ่งจะช่วยประหยัดทรัพยากรของระบบและเพิ่มประสิทธิภาพการทำงานของ server (ที่จริงใน ADO เองก็มีความสามารถในการทำ disconnected database ได้ แต่โปรแกรมเมอร์จะต้องเป็นผู้ควบคุมด้วยตนเองว่าจะให้มีการเชื่อมต่อหรือยกเลิกการเชื่อมต่อเมื่อใด ซึ่งต่างกับใน ADO.NET ที่จะตัดการเชื่อมต่อให้โดยอัตโนมัติ เมื่อไม่มีความจำเป็นต้องใช้ฐานข้อมูลในแบบ online และจะเปิดการเชื่อมต่อให้ไม่เมื่อจำเป็นต้องทำงานร่วมกับฐานข้อมูลจริง)

ในอีกประเด็นหนึ่งคือ ปัจจุบันการส่งข้อมูลเป็นลักษณะผ่าน Web มากขึ้น ภาษา XML มีบทบาทมากขึ้น โดยส่วนของ ADO.NET ก็จะมีความสามารถในการรับส่งข้อมูลเป็น XML อีกด้วย

7.2 องค์ประกอบของคอมโพเนนต์ใน ADO.NET

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 7-1 สถาปัตยกรรม ADO.NET

ADO.NET ประกอบด้วย 2 คอมโพเนนต์หลักคือ DataSet และ .NET data provider ซึ่งประกอบด้วย กลุ่มของ คอมโพเนนต์คือ Connection, Command, DataReader และ DataAdapter ดังรูปที่ 7-1

ADO.NET DataSet เป็นคอมโพเนนต์หลักในการเชื่อมต่อข้อมูลแบบ Disconnected, DataSet ถูกออกแบบมาให้มีความเป็นอิสระจากแหล่งข้อมูลด้วยเหตุนี้ มันจึงสามารถใช้ได้ใน Data Source หลายประเภท

DataSet ประกอบด้วยกลุ่มของ DataTable ซึ่ง Table ที่ว่านี้ก็เกิดมาจากการส่ง Query ไปดึงมาจาก ฐานข้อมูล โดย DataTable ก็จะประกอบไปด้วย DataRow, DataColumn รวมถึง primary key, foreign key, ข้อจำกัดต่างๆ พวก Data
Integrity และ ความสัมพันธ์ของข้อมูล (DataRelation) ใน DataTable เอง ในส่วนของการส่งข้อมูล DataSet ได้ใช้ XML เป็นตัวกลางในการส่งข้อมูลระหว่าง Tier ทำให้สามารถนำไปใช้กับการทำ Web Service ได้

ส่วนหลักอีกส่วนหนึ่งคือ .NET Data Provider ซึ่งสร้างมาเพื่อการประมวลผลข้อมูลแบบ รวดเร็วและแบบส่งไปข้างหน้าอย่างเดียว (forward-only) และเป็นการเข้าถึงข้อมูลแบบอ่านอย่างเดียว (read-only) โดยประกอบด้วยส่วนย่อยๆ ดังต่อไปนี้คือ

1. Connection ทำหน้าที่ติดต่อกับฐานข้อมูล SQL Server หรือ OLE DB เช่นเดียวกับออบเจ็ค Connection ใน ADO

2. Command จะใช้ร่วมกับออบเจ็ค Parameter ในการประมวลผลคำสั่ง SQL, Query หรือ Stored Procedure ในแบบที่สามารถส่งผลลัพธ์คืนกลับมาหรือจะไม่ส่งคืนก็ได้เช่นเดียวกับการใช้ออบเจ็ค Command ใน ADO

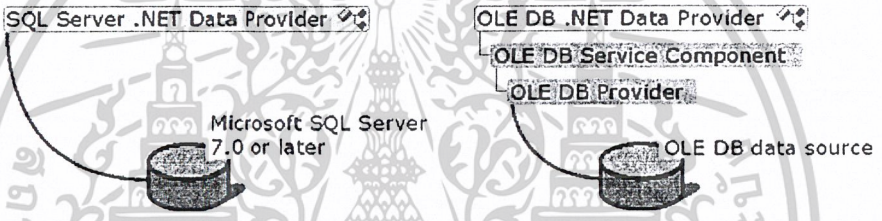
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. DataReader จะใช้สำหรับการดึงข้อมูลแบบทีละเรคอร์ดจากฐานข้อมูลบน server โดย server cursor จะเป็นแบบ forward-only คือเดินหน้าอย่างเดียวถอยหลังไม่ได้และไม่สามารถแก้ไขข้อมูลได้ด้วย (read-only) ทำให้ความเร็วในการเข้าถึงข้อมูลสูง

4. DataAdapter จะเป็นเหมือนสะพานเชื่อมต่อระหว่าง DataSet และ Data Source โดยการทำงานคือเมื่อมีการเปิด Connection และกำหนด Query String SQL แล้ว DataAdapter ใช้ Command Object เพื่อประมวลผลคำสั่งภาษา SQL และทำการดึงข้อมูลลงมา ที่ DataSet เพื่อใช้งานต่อไป โดย .NET Framework มี .NET data provider ให้ใช้ 2 แบบคือ

1. SQL Server .NET data provider ใช้ติดต่อและทำงานร่วมกับไฟล์ฐานข้อมูล Microsoft SQL Server ประกอบด้วยคลาส SqlConnection, SqlCommand, SqlDataReader และ SqlDataAdapter

2. OLE DB .NET data provider ใช้ติดต่อและทำงานร่วมกับไฟล์ฐานข้อมูลเช่น Microsoft Access, Oracle เป็นต้น ประกอบด้วยคลาส OleDbConnection, OleDbCommand, OleDbDataReader และ OleDataAdapter



รูปที่ 7-2 เปรียบเทียบระหว่าง SQL Server .NET Data Provider และ OLEDB .NET Data Provider

ก็คือ ถ้าใช้ SQL Server .NET Data Provider ต้องใช้กับ Microsoft SQL Server 7.0 หรือใหม่กว่าเท่านั้น โดย คลาสของ SQL Server .NET Data Provider กำหนดโดย namespace System.Data.SqlClient สำหรับเวอร์ชันที่เก่ากว่าก็ใช้ OLE DB .NET Data Provider แทน

ในการเลือกใช้ DataReader หรือ DataSet ใน แอปพลิเคชั่นชัน นั้น ควรพิจารณาถึงชนิดของหน้าที่การทำงานของแอปพลิเคชั่น โดยจะเลือกใช้ DataSet ก็ต่อเมื่อ

- มีการส่งข้อมูลระหว่างเทียร์ หรือมีต้องมีการใช้ XML Web service
- มีการติดต่อข้อมูลแบบไดนามิค โดยผูกติดกับ Window From หรือ การรวมข้อมูลที่เกี่ยวข้องกันมาใช้งานจากแหล่งข้อมูลหลายๆที่
- ให้มีการเก็บข้อมูลลงในแคชของเครื่อง
- มีการประมวลผลข้อมูลโดยไม่ต้องเปิดการเชื่อมต่ออยู่ตลอดเวลา ซึ่งทำให้ไคลเอนต์อื่นๆ สามารถเข้ามาใช้งานฐานข้อมูลได้มากขึ้น

โดยถ้าไม่มีความจำเป็นเหล่านี้ ก็สามารถใช้ DataReader แทนได้ โดยการทำงานจะเป็นแบบ Forward-Only and read-only คือเป็นการส่งข้อมูลให้อย่างเดียว และส่งไปข้างหน้าเท่านั้น และเป็นแบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อ่านได้อย่างเดียว คือ Update ข้อมูลกลับไปไม่ได้ โดยการใช้ DataReader จะเป็นการประหยัดหน่วยความจำลงไปได้มาก

7.3 สรุปข้อเปรียบเทียบระหว่าง ADO และ ADO.NET

1. ในการใช้งานกับ Web ซึ่งเป็นลักษณะ ดึงข้อมูลเว็บเพจนั้นเสร็จแล้วก็ทำการตัดสายอัตโนมัติ ซึ่งส่งผลทำให้การทำงานเร็วขึ้น ใช่ว่า ADO จะไม่มี แต่ ADO.NET ทำให้มันสะดวกขึ้นโดยทำโดยอัตโนมัติแทนที่โปรแกรมเมอร์ต้องไปควบคุมทุกขั้นตอน

   หมายเหตุ (ADO.NET จะทำ snapshot ของฐานข้อมูลที่เรากำลังติดต่ออยู่เก็บไว้ในหน่วยความจำของเครื่อง โดยการทำ Snapshot นั้นไม่ได้ทำทั้งฐานข้อมูล แต่ทำเฉพาะส่วนที่เรา query เท่านั้น โดยเมื่อดึงข้อมูลเสร็จก็จะทำการ Disconnect ทันที)
2. ส่วนการติดต่อกับฐานข้อมูลและส่วน การจัดการ client cursor (ภาษา ADO เรียกว่า Recordset) นั้นจะเป็นอิสระต่อกันโดยสิ้นเชิง ทำให้ทั้งสองส่วนพัฒนาได้โดยอิสระ
3. ใน ADO เมื่อก่อน Recordset ไม่สามารถพัฒนาได้มากนักเพราะว่าต้องผูกติดกับ Database Server ที่ตัวเองติดต่อด้วย ดังนั้นเพื่อให้สามารถรองรับได้กับ Database ทุกประเภท ทาง Recordset จึงจำเป็นต้องถูกออกแบบมาให้ทำงานเป็นแบบกลางๆ ทำให้ความสามารถด้อยลงไป ดังนั้น ใน ADO.NET จึงแยกในส่วนการติดต่อ Database ออกมา โดยจะมีหนึ่งคลาสต่อหนึ่งต่อหนึ่งประเภท Database เช่น ถ้าเราต่อกับ Oracle ต้องใช้คลาส หนึ่ง และถ้าต่อกับ Access ต้องใช้อีกคลาสไม่เหมือนกับ ADO ที่ใช้ OLEDB , คลาส ADODB.Connection เพื่อติดต่อกับทุก Database ข้อดีข้อการแยกก็จะทำให้การเชื่อมต่อกับ DBMS แต่ละยี่ห้อเป็นไปได้โดยอิสระและสามารถดึงเอาประสิทธิภาพมาใช้ได้อย่างเต็มที่
4. ส่วนของ Client Cursor นั้นปรับปรุงใหม่ แทนที่จะเป็น Recordset ซึ่งรองรับเพียงแค่ตารางเดียวเหมือนเมื่อก่อน แต่ใน ADO.NET สามารถรองรับทั้ง database และแต่ละ table สามารถมี reference integrity ด้วย
5. การส่งผ่านข้อมูลระหว่าง ADO.NET และ DBMS นั้น แทนที่จะส่งโดยใช้ DCOM ซึ่งจะมีปัญหากับ Firewall ทำให้การสื่อสารยุ่งยาก แต่ ADO.NET นั้นจะแปลงข้อมูลอยู่ในรูปของ XML ก่อนแล้วค่อยส่งออกไป ทำให้ใช้ port ของ web มาตรฐาน ทะลุกำแพง firewall ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 8

Microsoft .NET Technology

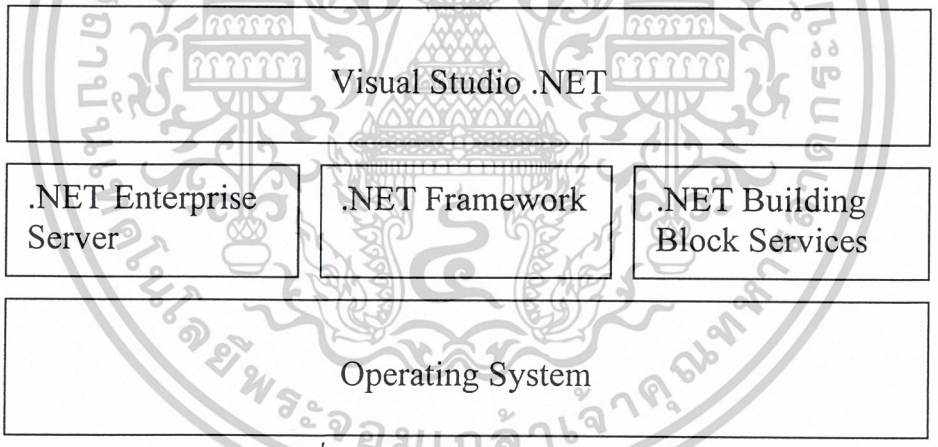
8.1 รู้จักกับ Microsoft .NET

Microsoft .NET หรือเรียกสั้นๆว่า .NET เป็นเทคโนโลยีใหม่ล่าสุดที่พัฒนาขึ้นโดยบริษัทไมโครซอฟท์ .NET คือ แพลตฟอร์มในการพัฒนาซอฟต์แวร์สำหรับระบบปฏิบัติการ Windows โดยถือเป็นก้าวสำคัญในการพัฒนาซอฟต์แวร์ยุคใหม่ซึ่งนำเสนอหลักการพัฒนาซอฟต์แวร์ด้วยภาษาอะไรก็ได้ที่เราถนัด และสามารถเรียกใช้งานโปรแกรมที่เขียนด้วยภาษาอื่นๆได้อย่างกลมกลืน ใน .NET มีภาษาโปรแกรมใหม่ๆเกิดขึ้นเช่น Visual Basic .NET, C# .NET, C++ .NET, J# .NET หรือแม้กระทั่ง COBOL .NET เป็นต้อน

ทุกภาษาที่สนับสนุน .NET จะอยู่ภายใต้กฎเกณฑ์มาตรฐานเดียวกันที่เรียกว่า Common Language Specifications (CLS) และโครงสร้างพื้นฐานตั้งแต่ชนิดข้อมูล, ชุดคำสั่งพื้นฐานเช่นการจัดการ I/O, ฐานข้อมูล ที่อยู่ภายใต้มาตรฐานเดียวกัน ทำให้เราสามารถพัฒนาซอฟต์แวร์โดยใช้ภาษาโปรแกรมได้หลายภาษา

8.2 .NET Platform

องค์ประกอบของแพลตฟอร์ม .NET แบ่งออกได้เป็น 3 ชั้น



รูปที่ 8-1 องค์ประกอบ .NET Platform

ชั้นระบบปฏิบัติการ (Operating System)

ชั้นถัดมาแบ่งออกเป็น 3 ส่วน ได้แก่

- .NET Enterprise Server : SQL Server 2000, BizTalk Server, Exchange 2000
- .NET Framework : เป็น Framework ที่สำคัญในการพัฒนา .NET
- .NET Building Block Services : Microsoft passport, Notification and Messaging, Personalization , XML Store, Calendar, Directory and Search

ชั้น Visual Studio .NET เครื่องมือในการพัฒนา .NET

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

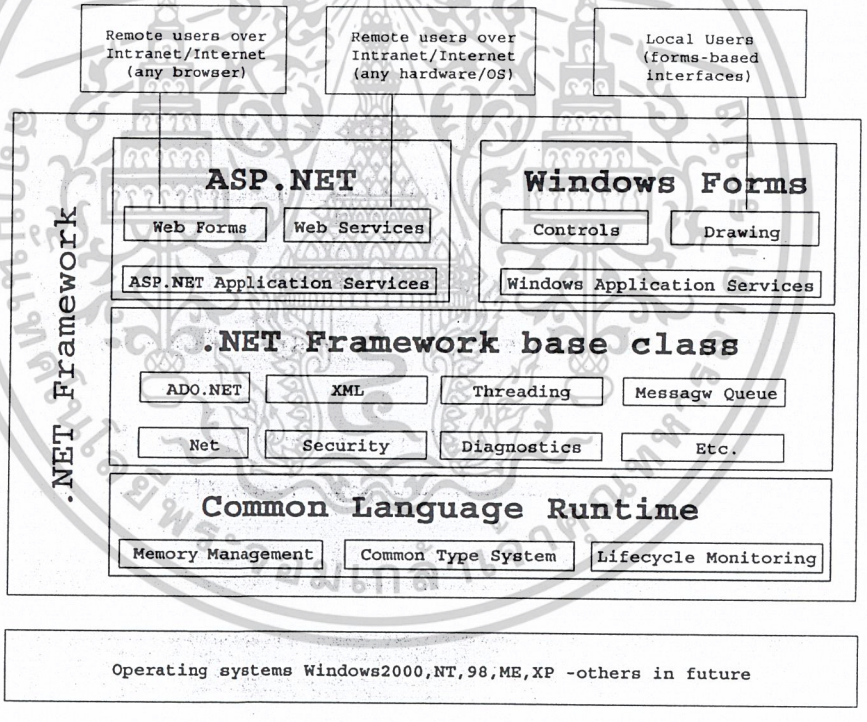
8.3 .NET Framework

.NET Framework มีองค์ประกอบ ดังนี้

1. Common Language Runtime (CLR) มีหน้าที่ในการจัดเตรียมบริการต่างๆที่เกี่ยวข้อง เพื่อให้โปรแกรมที่ขอใช้บริการสามารถทำงานได้

2. .NET Framework Class Library (Unified programming classes) คือไฟล์ไลบารีที่ทำหน้าที่จัดเก็บและรวมรวมข้อมูลที่จำเป็นต้องเรียกใช้งาน ในการพัฒนาโปรแกรม

3. ASP.NET เป็น Web application model ซึ่งประกอบไปด้วย กลุ่มของ control และ infrastructure ที่ใช้ในการสร้าง ASP Web applications , ASP.NET ประกอบไปด้วยส่วนของ control ซึ่งรวมถึงส่วนของ HTML ด้วย ส่วนของ control รันบนเซิร์ฟเวอร์ ส่วนHTML รันบน browser บนเซิร์ฟเวอร์นั้นส่วน control จะถูกเขียนในรูปแบบของ object-oriented programming , ASP.NET ยังให้บริการส่วน infrastructure ด้วยเช่น session state management และ process recycling ยิ่งไปกว่านั้น ASP.NET ก็ใช้คอนเซปนี้ในการพัฒนา จาก ซอร์ฟแวร์ เป็น เซอร์วิส โดยผ่าน กลไกสำคัญคือ XML เราสามารถเขียน business logic และใช้ ASP.NET infrastructure เพื่อให้บริการ ผ่าน SOAP



รูปที่ 8-2 องค์ประกอบ .NET Framework

8.3.1 Common Language Runtime (CLR)

CLR จะทำหน้าที่เป็นตัวกลางในการจัดเตรียมบริการและทรัพยากรสำหรับรองรับการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

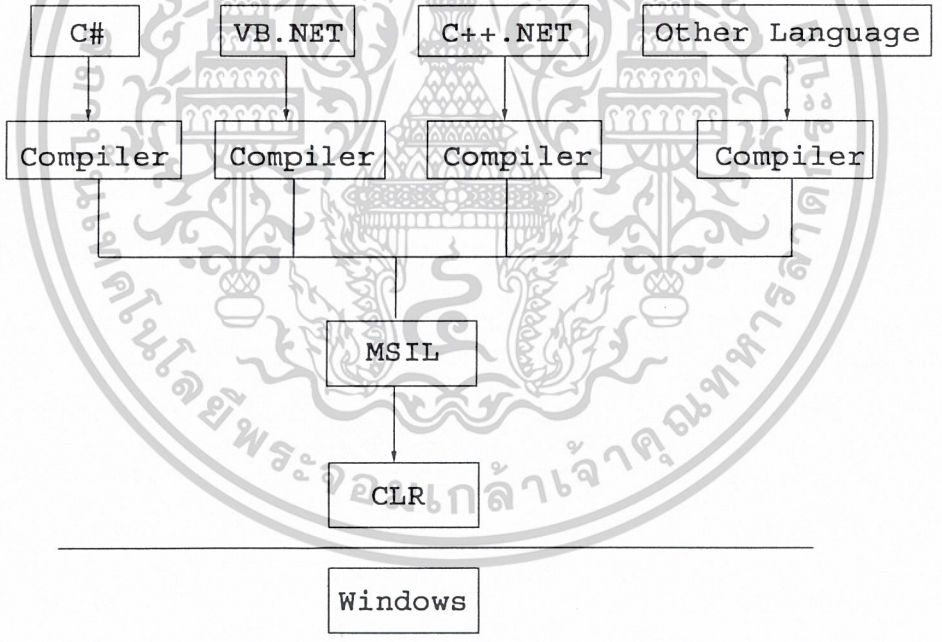
ประมวลผล และการทำงานของโปรแกรมประยุกต์ที่ทำงานบนเทคโนโลยีของ .NET เช่น การจัดการหน่วยความจำ, ความปลอดภัยในการเข้ารหัสโปรแกรม ด้วยความสามารถของ CLR ทำให้เราสามารถพัฒนาโปรแกรมโดยไม่ต้องขึ้นกับระบบปฏิบัติการ

การทำงานของโปรแกรมนั้น เริ่มจากคอมไพเลอร์ของแต่ละภาษาจะคอมไพล์โค้ดให้เป็นแบบ Microsoft's Intermediate Language (MSIL) หรือเรียกสั้นๆว่า IL Code ซึ่งจะมีลักษณะคล้ายภา Assembly จาก IL Code ก็แปลงเป็นโปรแกรมที่รันโดย CLR อีกทีหนึ่ง

นักพัฒนาสามารถที่จะเลือกได้ว่า เราจะ build แอพพลิเคชั่นไปอยู่ในรูปของ .exe หรือ MISL ซึ่งจะกลายเป็น Just-In-Time(JIT) คือเมื่อเราได้แอพพลิเคชั่นในรูปแบบของ MISL แล้ว เมื่อรันโปรแกรมใช้งานจริง มันจะถูก คอมไพเลอร์ JIT ทำการคอมไพล์โค้ด MSIL ในส่วนที่ต้องการใช้ไปเป็น Native Code อีกทีซึ่งนำไปให้เครื่องรันต่อ หากมีการใช้โค้ดในส่วนเดิมอีกก็ไม่ต้องมีการคอมไพล์ซ้ำ

การแปลงโค้ด MSIL ไปเป็น Native Code ประโยชน์ที่เราจะได้คือ

1. เราสามารถแปลงไปเป็น Native Code ที่เหมาะสมกับระบบปฏิบัติการนั้นรันอยู่
2. เราสามารถได้ Native Code ที่ใช้ความสามารถของ CPU ได้อย่างเต็มที่ เช่นใน CPU P4 เราสามารถใช้คำสั่งในส่วนของ SSE2 ได้



รูปที่ 8-3 รูปแบบการคอมไพล์โค้ดไปเป็น IL Code

Managed Code เป็นโค้ดที่คอมไพล์ให้ทำงานใน .NET Framework
Native Code เป็นโค้ดที่คอมไพล์เป็นภาษาเครื่องโดยตรง ๆไม่ต้องผ่านตัว CLR ในการรันโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Unmanaged Code เป็นโค้ดที่ไม่ได้คอมไพล์เพื่อให้ทำงานใน .NET Framework เช่น COM Object ใน version ก่อนหน้านี้ ซึ่ง .NET ก็ยังสามารถรันโค้ดพวกนี้ได้ แต่ไม่สามารถใช้ความสามารถของ .NET ได้อย่างเต็มที่

8.3.1.1 Memory Management

สำหรับ .NET แล้วจะมี Garbage Collector (GC) ซึ่งทำหน้าที่คอยรวบรวมหาหน่วยความจำที่ถูกทิ้งไว้เป็นขยะ แต่ไม่มีใครอ้างอิงเรียกใช้แล้ว โดยการทำงานจะแตกต่างจากใน VB Runtime รุ่นเก่าที่ให้ Object รับผิดชอบในการจัดการ Garbage Collector เป็นผู้ดูแลจัดการทั้งหมด โดย GC จะเรียก method ชื่อ Object.Finalize โดยอัตโนมัติเมื่อ object เลิกใช้งาน

การจัดการหน่วยความจำที่ดีกว่าของ .NET เกิดขึ้นเมื่อ object ถูกสร้างขึ้นมา CLR จะจัดการกันพื้นที่ส่วนหนึ่งไว้สำหรับโปรแกรมที่เรียกว่า Heap โดยเริ่มต้นจากพื้นที่หน่วยความจำที่ว่าง จากนั้นก็อ้างอิงชี้ตำแหน่งไปยังส่วนบนสุดของหน่วยความจำ เมื่อหน่วยความจำถูกใช้แล้วยกเลิกไปเรื่อยๆ จะเหลือพื้นที่ว่างเป็นช่วงๆ ซึ่ง GC จะเข้ามาทำการ Compact หน่วยความจำเสียใหม่ ลองนึกถึงหลักการทำงานแบบเดียวกันกับตอนที่ทำ Defragment บนฮาร์ดดิสก์

8.3.1.2 Common Type System

ใน .NET จะมี common Type System ที่จะให้ทุกภาษามี Types ที่เหมือนกัน เป็นมาตฐาน โดยที่ทุก Types ที่สนับสนุนโดย Common Types System นั้น จะสืบทอดคุณสมบัติมาจาก System.object ดังนั้นคุณจะพบว่า Object ทั้งหลายส่วนมากแล้วจะสนับสนุน Method เหล่านี้คือ

Equal(Object) = Boolean

GetHashCode() = Int32

GetType () = Type

Tostring () =String

ข้อดีของ Common Language Runtime

- ไม่มีปัญหาเรื่องการรัน บน MS Windows Platform ต่างๆ เนื่องจาก CLR จะทำการตรวจสอบระบบปฏิบัติการให้โดยอัตโนมัติ และทำการจำลองสภาพให้เหมาะสมกับการทำงานของโปรแกรมโดยอัตโนมัติเช่นเดียวกัน

- ไม่ต้องสนใจเรื่อง Registry การเข้าถึง คอมโพเนนต์ ต่างๆที่เกี่ยวข้องจะอยู่ในความดูแลของ CLR ทั้งหมด เราจำเป็นต้องรู้แค่ว่าใช้ Namespace ตัวใดในการอ้างอิงถึงเท่านั้น

- รองรับการพัฒนาจากหลายภาษา ในปัจจุบันเราสามารถใช้หลายภาษามาทำงานร่วมกันได้ด้วยเทคโนโลยีของ COM แต่อย่างไรก็ตาม เนื่องจากโปรแกรมยังผูกติดกับภาษาอยู่ ความเร็วของโปรแกรมที่ได้ก็จะแตกต่างกัน ขึ้นอยู่กับความสามารถของ Compiler ภาษานั้นๆ Microsoft Common Language Specification จะทำให้ compiler ทุกตัวที่ใช้ .NET ทำการแปลภาษาไปยังเป้าหมายเดียวกัน คือ การทำงานกับ CLR ได้อย่างไร้ปัญหา ส่งผลให้ โปรแกรมที่ได้จากและภาษามีผลลัพธ์ที่เท่าเทียมกันเมื่อทำงานอยู่ภายใต้การดูแลของ CLR

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- นำ Source code มาใช้งานใหม่ได้ COM ยังถูกจำกัดการ reuseในวงแคบเนื่องจากมีการผูกติดกับระบบปฏิบัติการ .NET แยกออกได้ 3 รูปแบบได้แก่

1. เราสามารถเขียน คลาส ด้วยภาษาหนึ่ง สามารถเรียกใช้ผ่านอีกภาษาหนึ่งได้

2. คลาสสามารถ Inherit ข้ามภาษาได้

3.เว็บเซอร์วิส เป็นอีกรูปแบบหนึ่งโดยที่เสนอบริการให้แก่ทุกภาษาที่อ้างอิง CLR ได้

4. ชนิดของข้อมูลมีประสิทธิภาพ จากข้อผิดพลาดเกี่ยวกับตัวแปรเช่น ประกาศตัวแปรขนาด 10 ไบต์ แต่ทำงานจริงกลับมีขนาด 20 ไบต์ ถ้าไม่มีการเขียนโปรแกรมดักจับ โปรแกรมจะทำงานผิดพลาด ใน .NET สิ่งเหล่านี้จะไม่เกิดขึ้น CLR จำตรวจสอบการทำงานของ code อย่างใกล้ชิดก่อนที่จะประมวลผลขั้นสุดท้าย CLR จะไม่อนุญาตให้โปรแกรมใช้ตัวแปรที่มีการผิดพลาดเหล่านั้น

5. การ Debug ที่มีประสิทธิภาพ CLR จะทำการแยกส่วนของโปรแกรมเป็นส่วนๆทำให้การตรวจสอบข้อผิดพลาดทำได้ง่ายสะดวกและรวดเร็ว การแก้ไขสามารถแก้ไขเฉพาะส่วนที่ผิดพลาดได้โดยไม่รบกวนส่วนอื่นๆของโปรแกรม และเนื่องจาก CLR จัดการกับทุกภาษาด้วยกัน ข้อผิดพลาดต่างๆที่แสดงออกมาในแต่ละภาษาจะอยู่ในรูปแบบเดียวกัน

6. ระบบจัดสรรทรัพยากรที่ดีขึ้น การดำเนินการที่เกี่ยวข้องกับทรัพยากรต่างๆ ระหว่างโปรแกรมไม่ว่าจะเป็นระบบการจัดการไฟล์ระบบการเชื่อมต่อ หรือพื้นที่ว่างของหน้าจอ เป็นต้น จะถูกดูแลจัดการอย่างใกล้ชิด โดย CLR ไม่สามารถมีโปรแกรมตัวใดตัวหนึ่งครอบครองทรัพยากรไว้แต่เพียงผู้เดียว

7. การแจกจ่ายโปรแกรมง่ายขึ้น การติดตั้งโปรแกรมแบบเดิม โปรแกรมหนึ่งๆมีไฟล์เป็นจำนวนมาก ,ต้องมีการลงทะเบียนใน registry ,การสร้าง short cut , การจัดเตรียมระบบที่เกี่ยวข้อง ใน .NET การติดตั้งโปรแกรมเป็นไปโดยง่าย เนื่องจาก CLR จะทำการแยกโปรแกรมออกเป็นส่วนๆ การติดตั้งโปรแกรมจึงเป็นการ คัดลอก ไฟล์ที่เกี่ยวข้องมาไว้บน hard disk เท่านั้น ในทางกลับกัน การยกเลิกการติดตั้ง ก็เพียงลบไฟล์เหล่านั้นออก

8. ระบบความปลอดภัยที่ดีขึ้น ระบบรักษาความปลอดภัยใหม่นี้จะทำการตรวจสอบพฤติกรรมการทำงานของโปรแกรมอย่างใกล้ชิด อย่างในระบบเดิมนั้น การทำงานของ ภาษาสคริปต์ต่างๆหรือโปรแกรมที่มาจาก Internet อาจเป็นโปรแกรมที่สร้างช่องโหว่ให้กับระบบได้

8.3.2 .NET Framework Class Library

ไฟล์ไลบรารีที่ทำหน้าที่จัดเก็บและรวมรวมข้อมูลที่จำเป็นต้องเรียกใช้งาน ในการพัฒนาโปรแกรม ซึ่งไฟล์เหล่านี้ เป็นไลบรารีร่วมกันในทุกภาษา ส่งผลให้เราสามารถ inheritance ,error handling, debugging ข้ามภาษาได้ และการที่ .NET Framework เสนอบริการสำเร็จรูปผ่าน Class Library เหล่านี้ทำให้ รูปแบบการเขียนโปรแกรมเปลี่ยนไป โดยมีความเป็นระบบ และเป็นลักษณะของ OOP เพิ่มขึ้น เช่นใน VB.NET ต้องมีการอ้างอิงถึง class library ที่เกี่ยวข้องและต้องการใช้งาน

8.3.3 Presentation Layer

การแสดงผลสามารถทำได้ 2 รูปแบบหลัก คือในรูป Windows form และ Web form ซึ่งก่อนนี้ทั้งคำสั่ง และกระบวนการเขียนโปรแกรม ของทั้ง 2 แบบ แต่ใน .Net แล้วได้รวมทั้งสองอย่างมาทำให้เหมือนกันที่สุด คือเราเขียนโปรแกรมบน Windows อย่างไร คุณก็เขียน Web Application ด้วยวิธีเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยเรียกใช้ คลาส ด้วยคำสั่งที่เหมือนๆกัน บน Windows จะอยู่ใน คลาส ที่ชื่อ system.windows แต่ Web จะอยู่ใน คลาส ชื่อ system.web.ui ซึ่งวิธีใช้เหมือนกันมาก ที่พิเศษคือ มี Object สำหรับ Web ให้คุณเขียนโปรแกรม แบบ visual basic ธรรมดา แต่เมื่อรันบน browser แล้วจะสร้าง tag html และ javascript ให้อัตโนมัติ ทำให้ browser ทั่วๆไปก็สามารถรันได้ ไม่จำเป็นต้องสนับสนุน .net โดยเฉพาะ
ตัวอย่างเช่น เราสร้างเว็บเพจที่มี ปฏิทิน ด้วย(นึกถึง calendar ใน VB) ในส่วนของ รูปแบบ web จะแสดงในรูปของ html code ส่วนปฏิทินจะอยู่ในรูปของ java script ซึ่งทางโปรแกรมจะทำการแปลงให้เอง

8.4 .NET Compact Framework

.NET Compact Framework คือ platformของ Microsoft .NET ที่ใช้ในการพัฒนาอุปกรณ์เล็กๆ ความจริงแล้ว .NET Compact Framework ก็คือ .NET Framework ที่ตัดเอาส่วนที่ไม่สำคัญหรือไม่ได้ใช้ออกไป เพื่อทำให้ขนาด เล็กลงจนขนาดที่สามารถ install ลงบนอุปกรณ์เล็กๆ ที่ไม่ใช่คอมพิวเตอร์ได้ เช่น โทรศัพท์มือถือเป็นต้น

คุณสมบัติหลักของ .NET Compact Framework

- .NET Compact Framework ถูกออกแบบให้เป็นplatformในการพัฒนา สำหรับการเขียนและเรียกใช้ XML เว็บเซอร์วิส
- เนื่องจาก .NET Compact Framework ใช้ programming model เดียวกันกับ desktop .NET Framework จึงง่ายต่อ developer ในการเขียนapplication ใหม่บน .NET Compact Framework
- ใช้ Visual Studio .NET ตัวเดียวกับที่ใช้ใน desktop .NET Framework ในการเขียน program device application
- ใช้ระบบ evidence-base security คืออนุญาตให้เฉพาะ applicationที่ผ่านการรับรอง จากผู้ผลิต device นั้น จึงจะสามารถ access secure resourceได้ ถ้าเป็น application ที่ยังไม่ผ่านการรับรองความปลอดภัย จะ access ได้เพียง basic system resourceเท่านั้น
- ออกแบบสำหรับอุปกรณ์ที่มี resource จำกัด เช่น โทรศัพท์มือถือ โดยที่ตัว frameworkเองใช้ memory อย่างมีประสิทธิภาพ และมีระบบดึง resource ที่ไม่ได้ใช้แล้วกลับมาจาก application อย่างมีประสิทธิภาพ
- ใช้ Just–in-time complier ซึ่งมีประสิทธิภาพดีกว่า ระบบ device programming อื่นๆ ที่ใช้ code interpreter
- เพิ่มประสิทธิภาพและลดต้นทุนในการพัฒนา เนื่องจากใช้ programming model และ toolset ของ Visual Studio .NET อันเดียวกันกับของ desktop .NET framework ทำให้นักพัฒนาสามารถใช้ความรู้ความชำนาญที่มีอยู่แล้วมาพัฒนา application บน อุปกรณ์ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 9
แอพพลิเคชั่นระบบ Land and House

9.1 Overview ของระบบ Land and House

เป็นระบบจัดการการขายบ้านและที่ดิน ซึ่งเป็นเว็บเซอร์วิสให้บริการจัดหาบ้านและที่ดินตามข้อมูลที่ต้องการ โดยผู้ใช้บริการสามารถค้นหาข้อมูลต่างๆได้ตามต้องการ โดยกรอกข้อมูลรายละเอียดที่จำเป็นในการค้นหาลงไป เช่น ต้องการบ้านที่มีห้องน้ำ 2 ห้อง ห้องนอน 3 ห้อง เป็นต้น

เมื่อผู้ใช้บริการได้บ้านตามที่ต้องการแล้วก็สามารถทำการเลือกซื้อได้โดยจะมีรายละเอียดของสินเชื่อของแต่ละธนาคารได้ให้เลือก ซึ่งหลังจากซื้อแล้วก็สามารถยกเลิกการซื้อได้ ส่วนเรื่องการชำระเงินจะเป็นการชำระผ่านบัตรเครดิต เมื่อชำระค่าบริการเรียบร้อยแล้ว หากผู้ใช้บริการต้องการยกเลิกรายการที่จองไว้ ผู้ใช้บริการจะไม่ได้ค่าบริการคืนซึ่งเป็นกฎของทางระบบ Land And House

ระบบ Land And House สามารถค้นหาบ้านและที่ดินตามเงื่อนไขที่ต้องการ ทำการซื้อ ชำระเงิน ดูรายละเอียดการซื้อ และยกเลิกรายการซื้อ

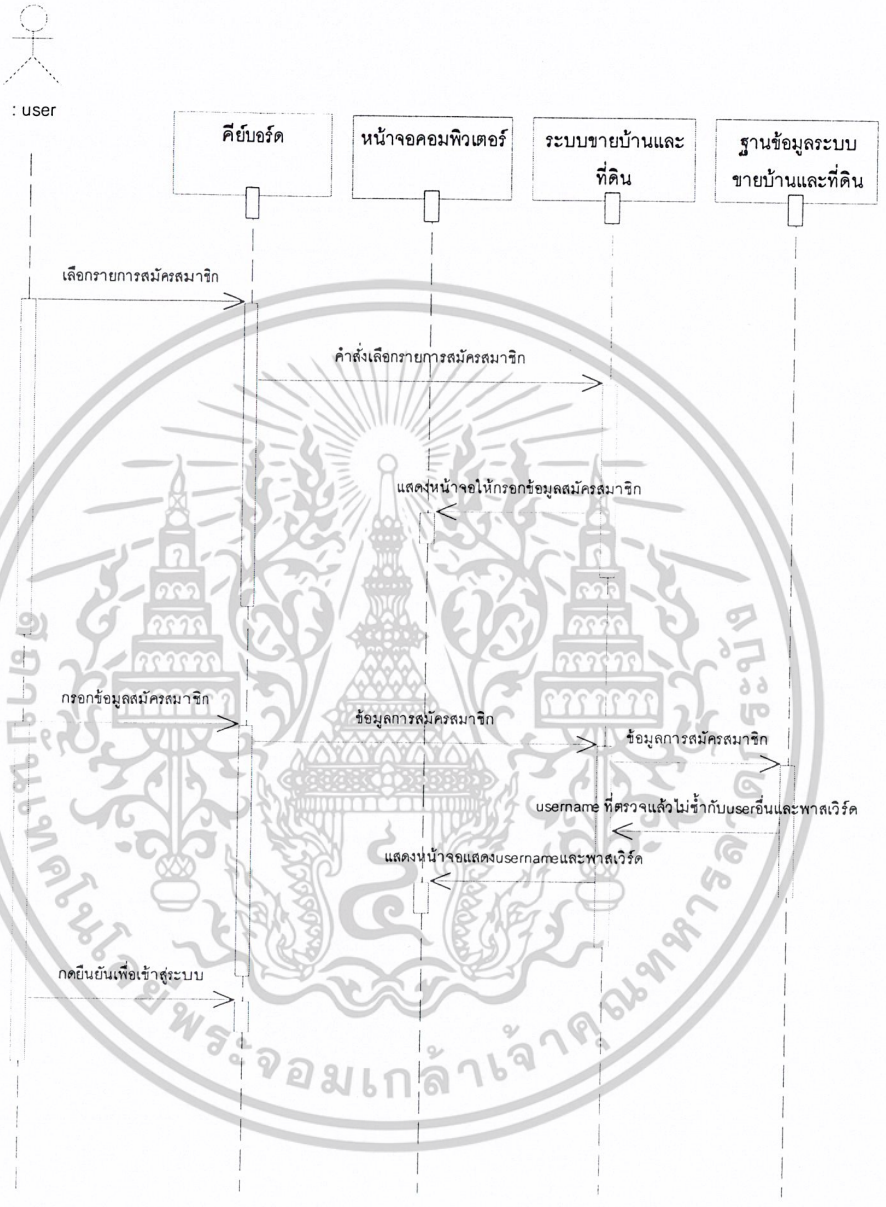
9.2 Use Case Diagram ระบบ Land And House

รูปที่ 9-1 Use Case Diagram ระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.3 Sequence Diagram ระบบ Land And House

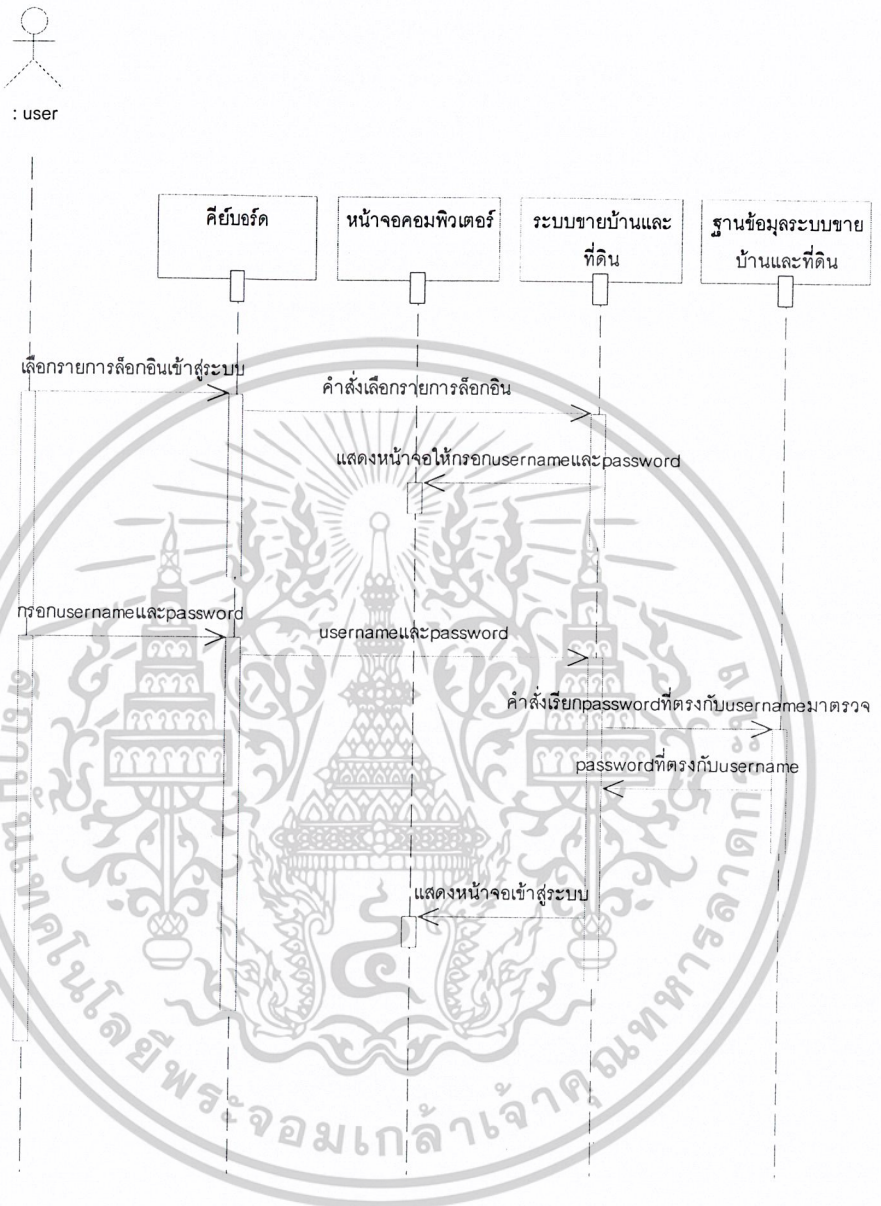
9.3.1 สมัครสมาชิก



รูปที่ 9-2 Sequence Diagram การสมัครสมาชิกของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

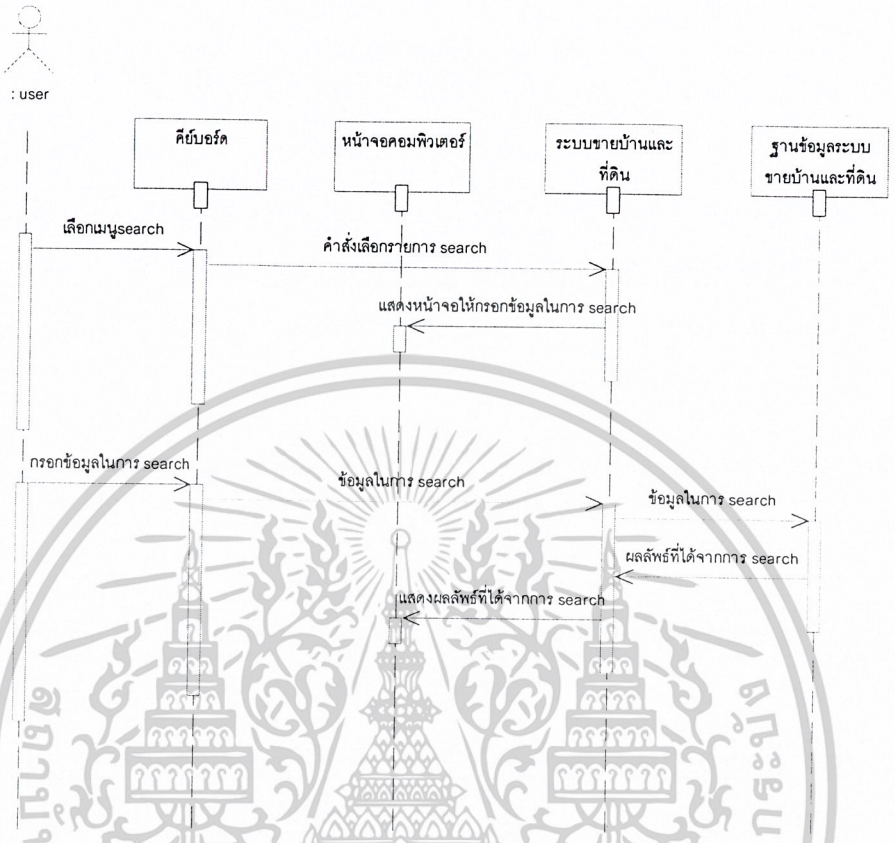
9.3.2 การล็อกอิน



รูปที่ 9-3 Sequence Diagram การล็อกอินของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

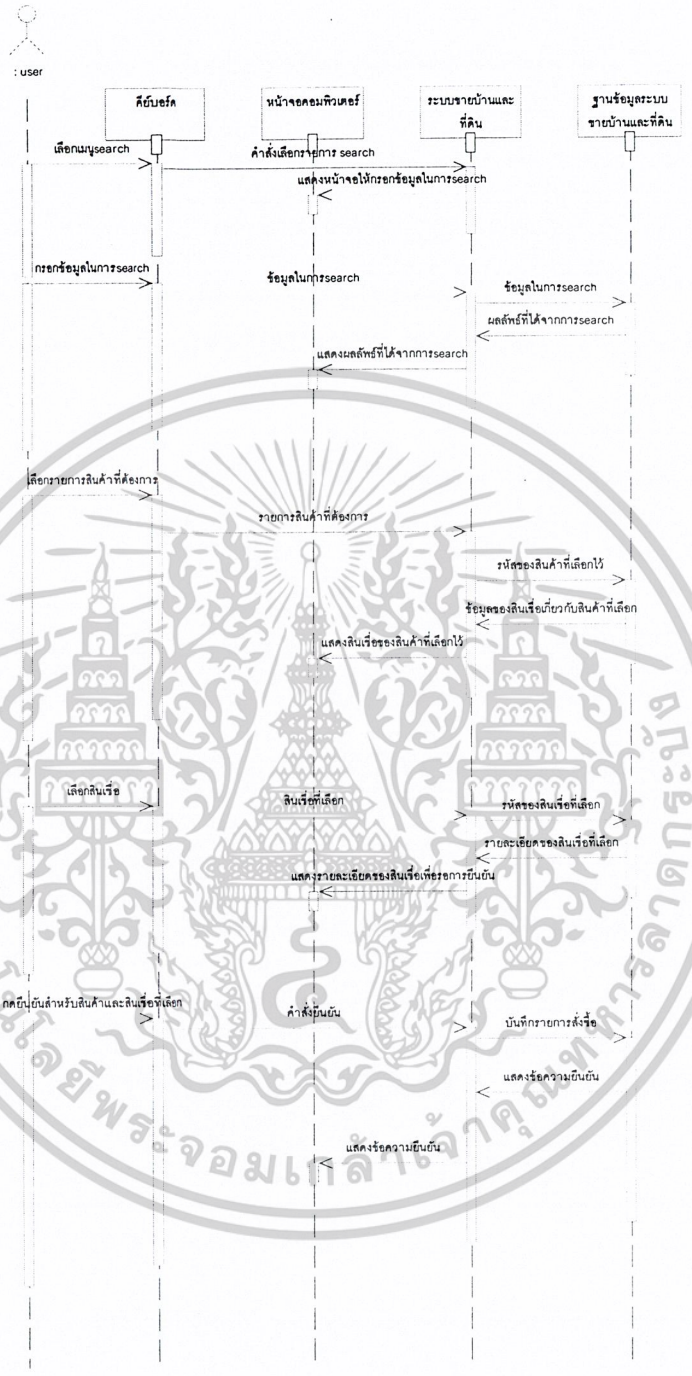
9.3.3 การค้นหาบ้านและที่ดินตามเงื่อนไข



รูปที่ 9-4 Sequence Diagram การค้นหาบ้านและที่ดินตามเงื่อนไข ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

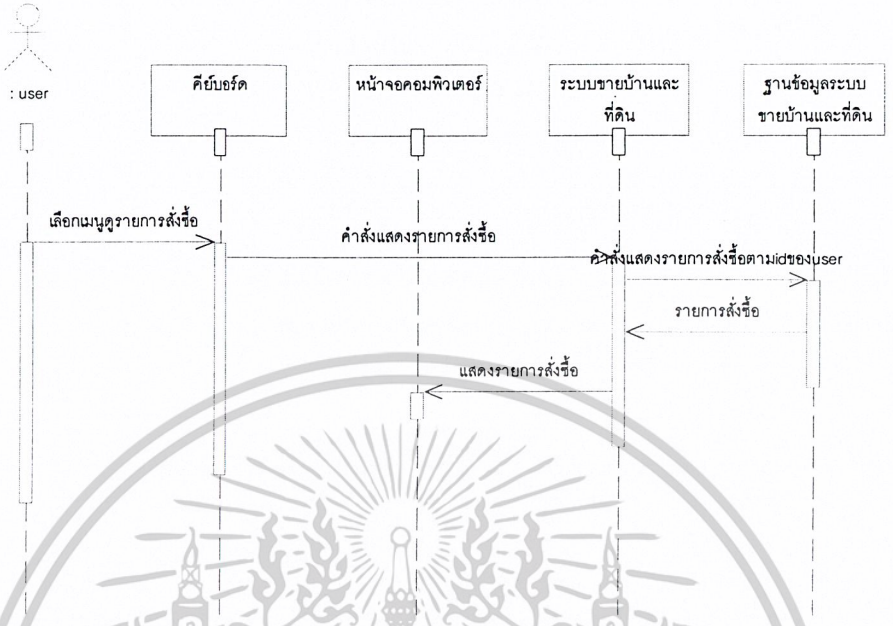
9.3.4 การซื้อสินค้า



รูปที่ 9-5 Sequence Diagram การซื้อสินค้า ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

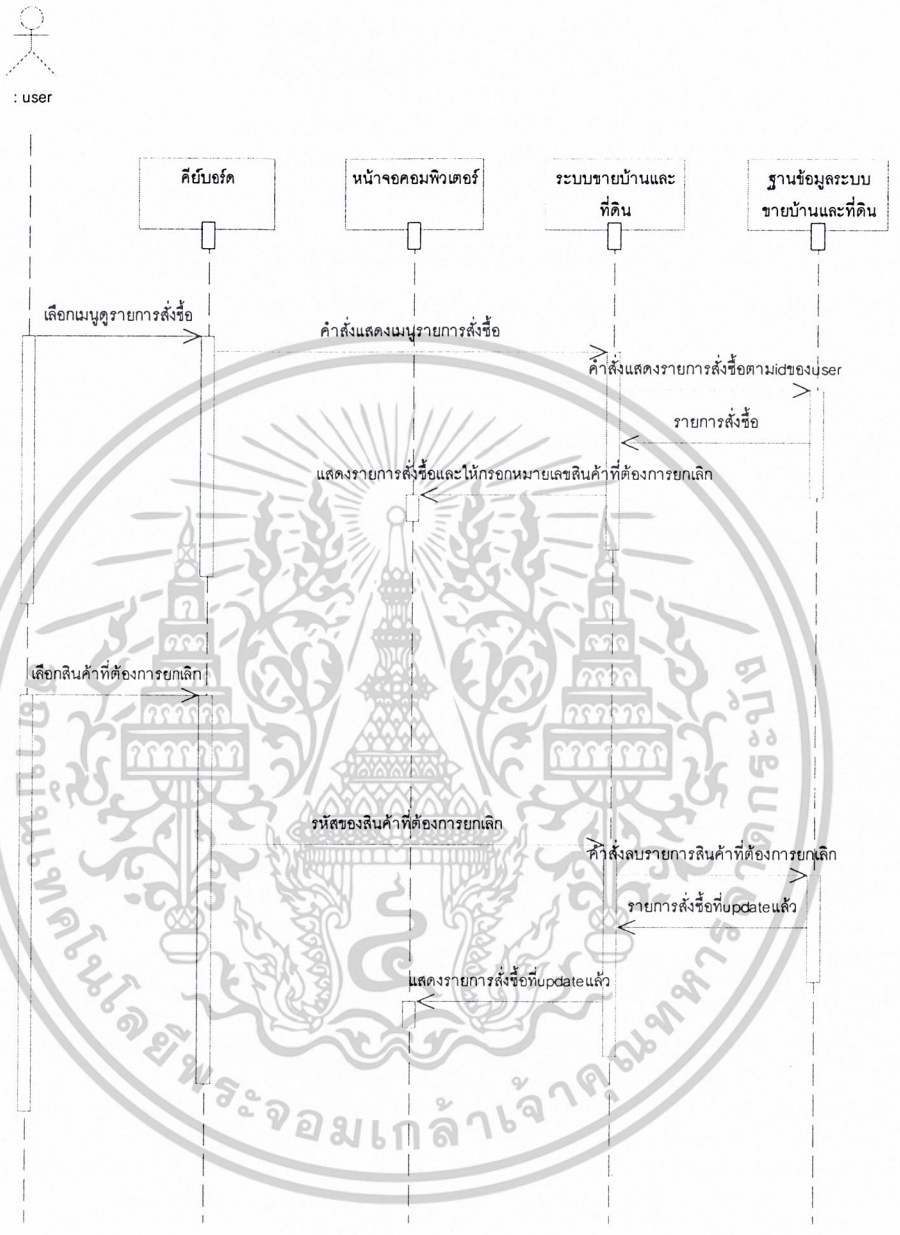
9.3.5 การดูรายการสั่งซื้อสินค้า



รูปที่ 9-6 Sequence Diagram การดูรายการสั่งซื้อ ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

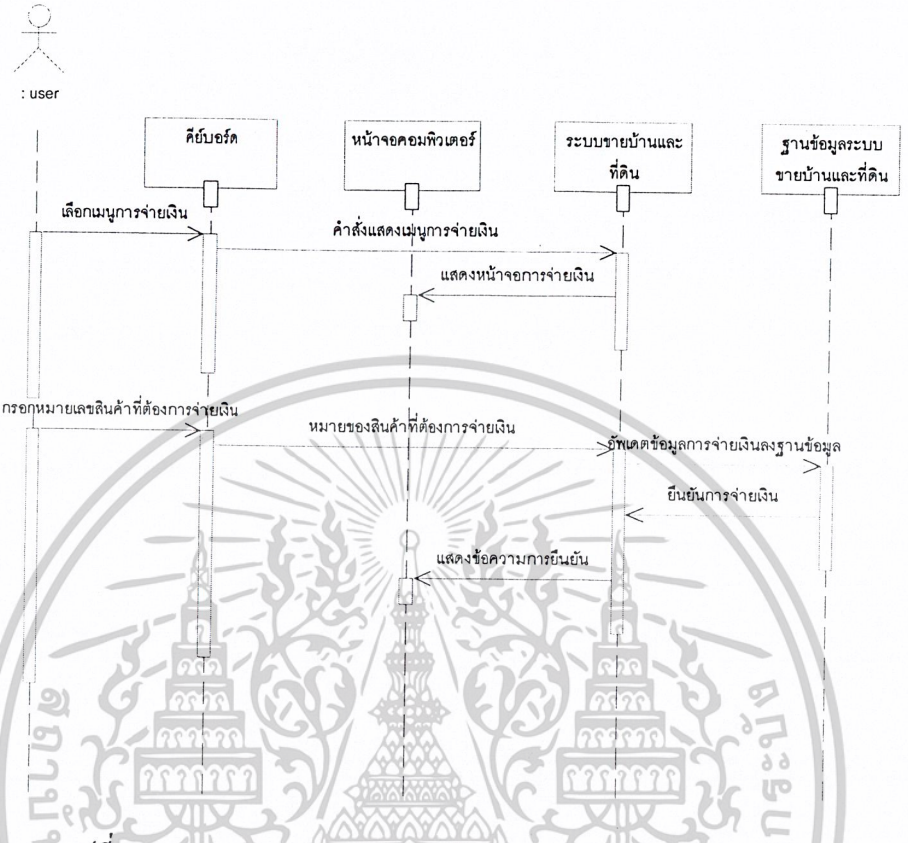
9.3.6 การยกเลิกรายการสั่งซื้อ



รูปที่ 9-7 Sequence Diagram การยกเลิกรายการสั่งซื้อ ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

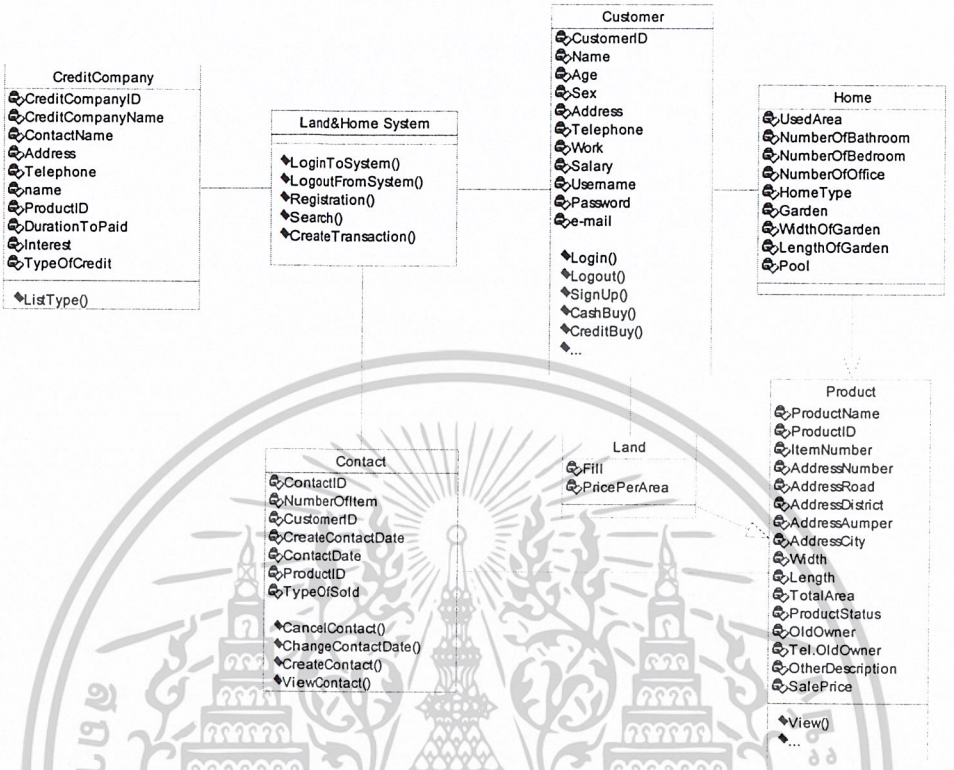
9.3.7 การชำระเงิน



รูปที่ 9-8 Sequence Diagram การชำระเงิน ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

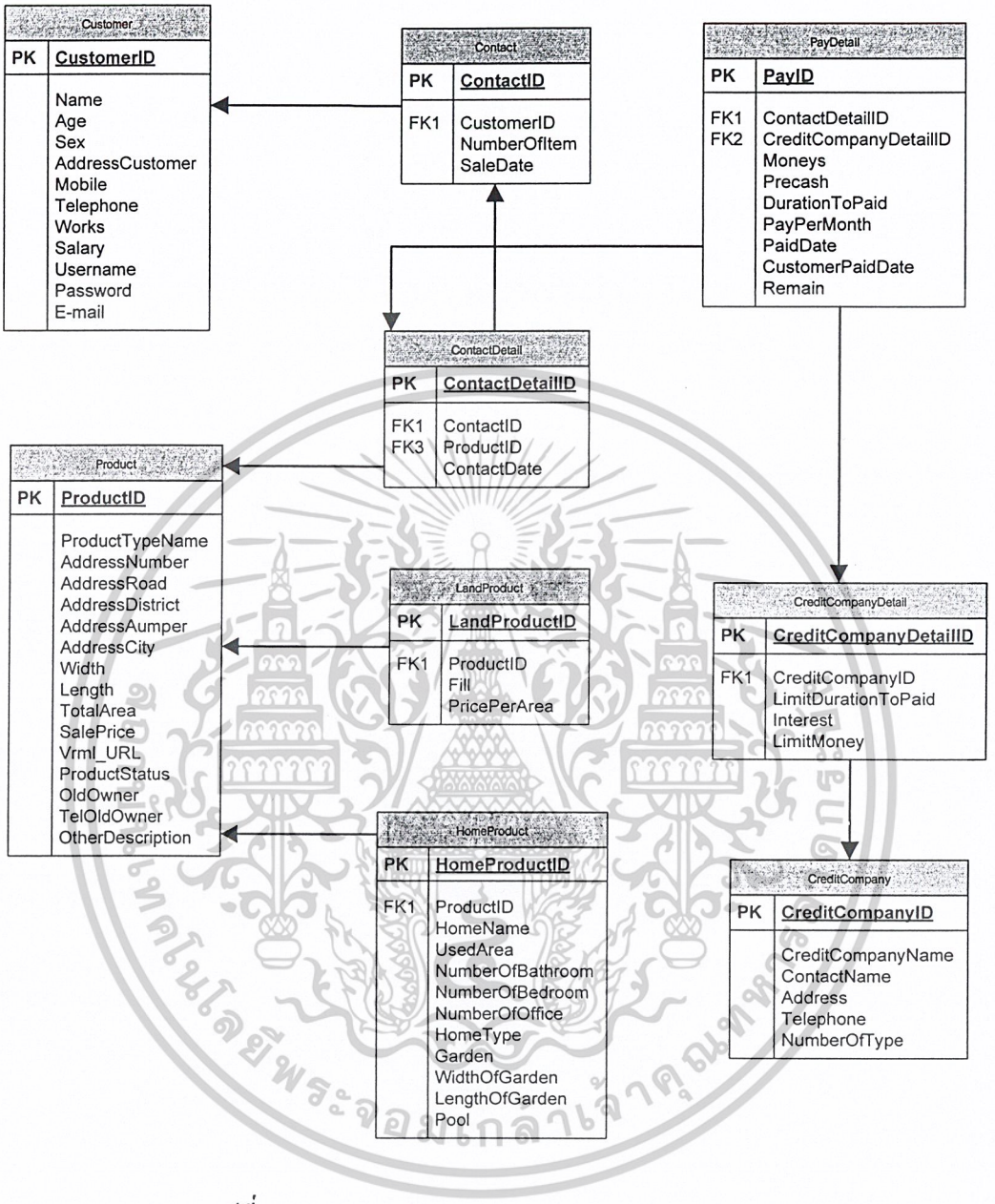
9.4 Class Diagram ระบบ Land and House



รูปที่ 9-9 Class Diagram ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.5 Database Diagram ของระบบ Land And House



รูปที่ 9-10 Database Diagram ของระบบ Land And House

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.6 คู่มือการใช้งานระบบ Land And House

9.6.1 ขั้นตอนเริ่มการใช้งาน

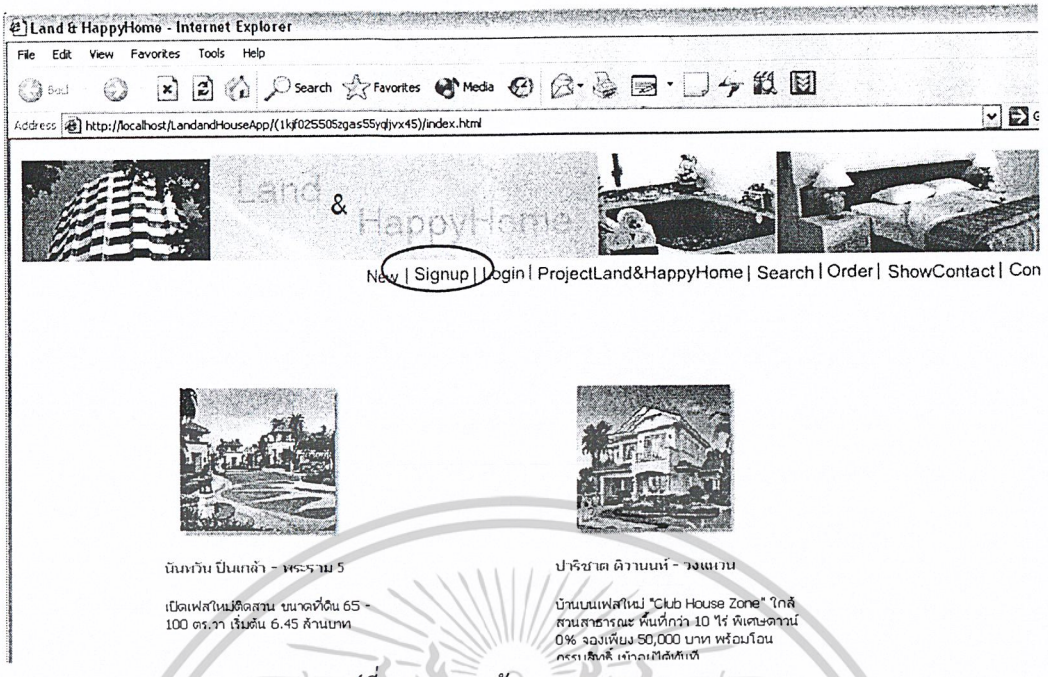
เริ่มต้นการใช้งานโดยที่เราจะเข้าไปที่ index.html เพื่อที่จะทำการซื้อสินค้าต่าง ๆ

รูปที่ 9-11 การหน้าจอหลัก

9.6.2 ขั้นตอนการ Signup

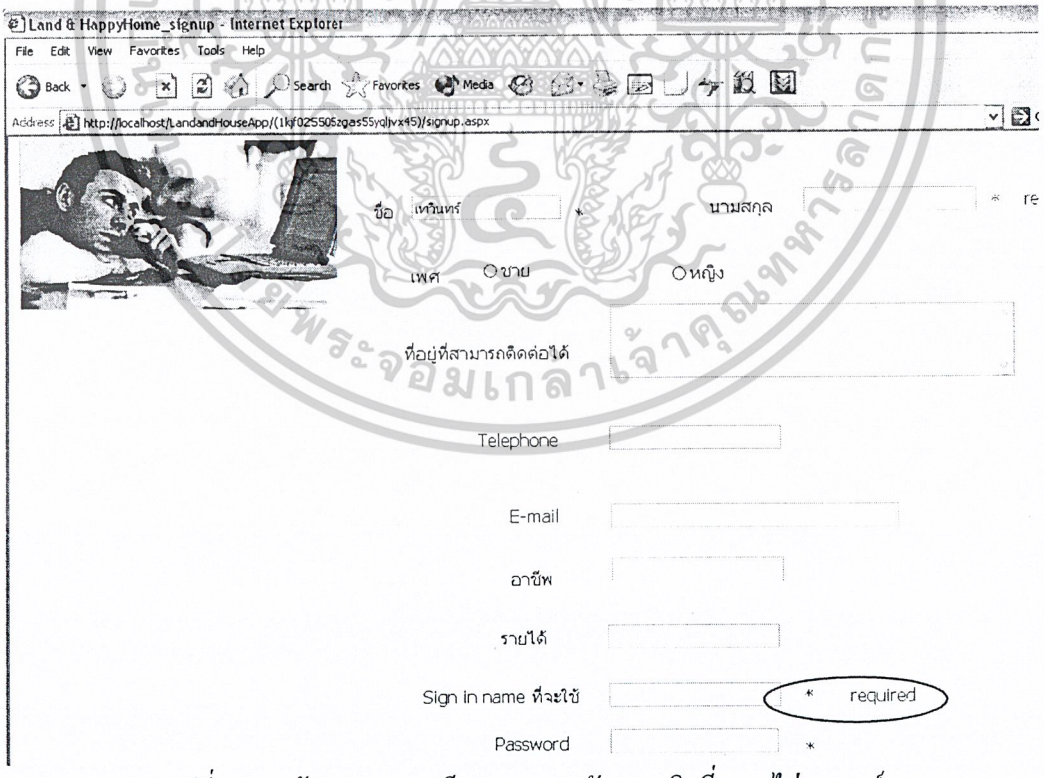
เราก็เข้าไปที่ page signup เพื่อที่จะทำการสมัครเป็นสมาชิกก่อนเพราะว่าบาง page ของเราต้องการที่จะให้ page นั้นเข้าถึงได้เฉพาะสมาชิกเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-12 การหน้าจอแสดงการ signup

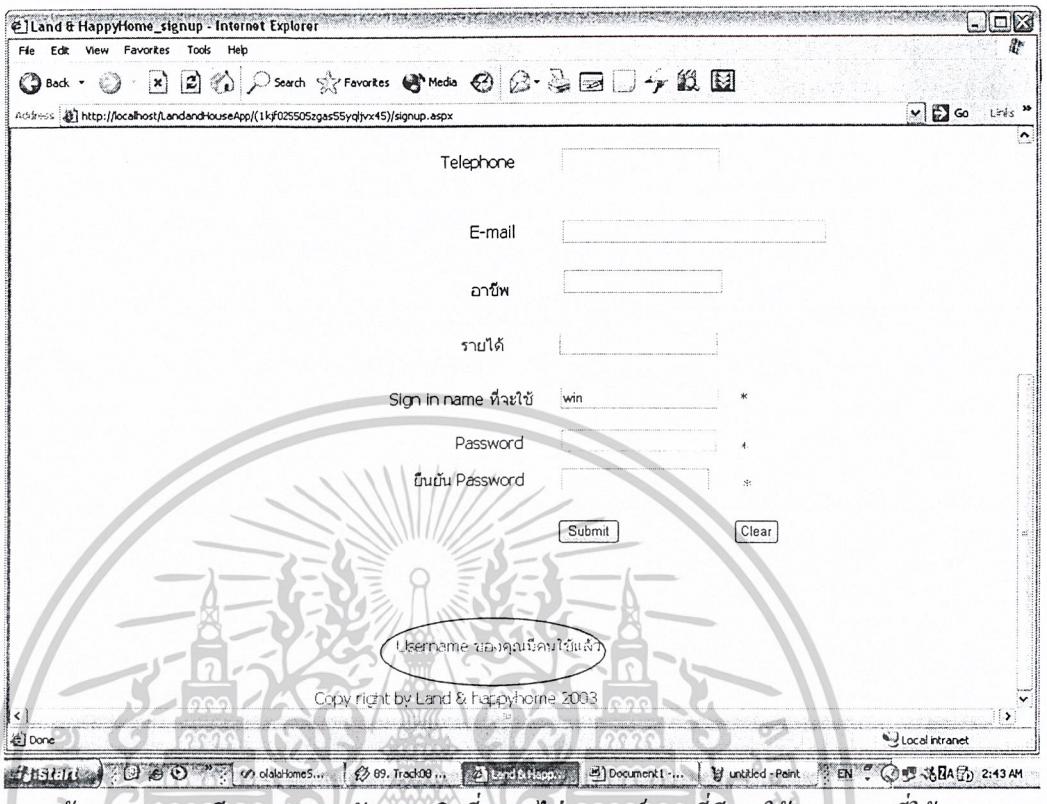
จากนั้นเราก็ไปกรอกข้อมูลให้ครบในช่องที่กำหนดไว้ ถ้ากรอกไม่ครบก็จะปรากฏดังรูป



รูปที่ 9-13 หน้ากรอกรายละเอียดของการสมัครสมาชิกที่กรอกไม่สมบูรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นได้ว่าถ้าไม่ได้กรอกข้อมูลในช่องไหนก็จะขึ้นคำว่า required มาเพื่อให้ user ทราบว่าช่องนั้นต้องการที่จะให้กรอก

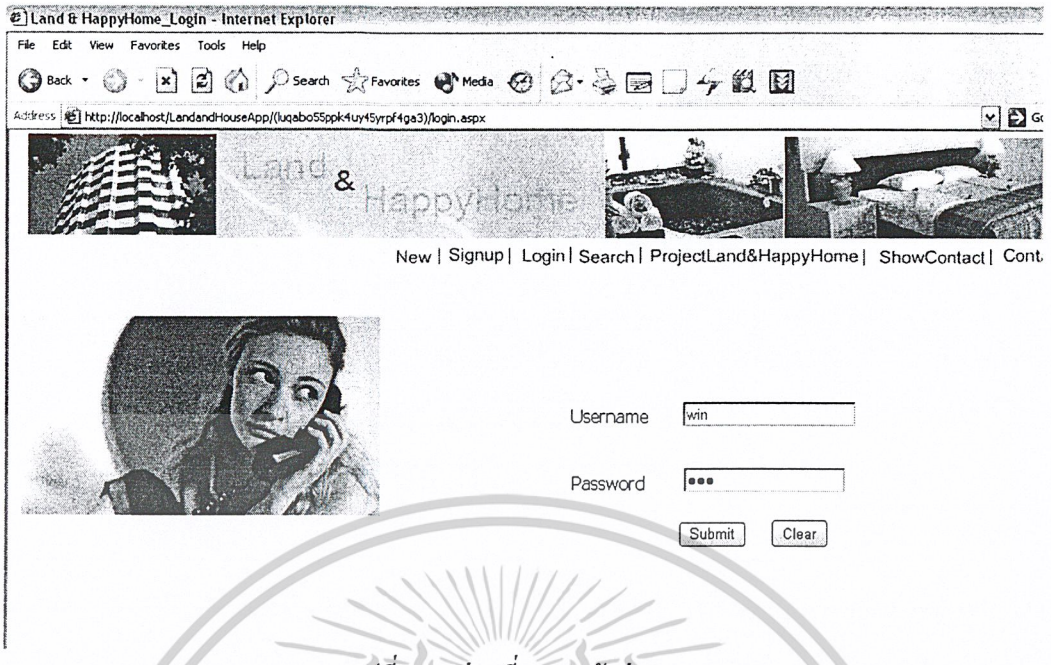


รูปที่ 9-14 หน้ากรอกรายละเอียดของการสมัครสมาชิกที่กรอกไม่สมบูรณ์แบบที่มีการใช้ usename ที่ใช้งานแล้ว

ถ้า user ของคุณมีคนใช้แล้วระบบก็จะฟ้องออกมาว่า “User ของคุณมีคนใช้แล้ว” ดังนั้นเราจึงต้องเปลี่ยนไปใช้ user อื่นแทน เมื่อทำการ signup เรียบร้อยแล้วระบบจะทำการ redirect ไปที่ page Login เพื่อที่จะทำการให้ user login

9.6.3 ขั้นตอนการ Login

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-15 ส่วนที่ login เข้าสู่ระบบ

จากนั้นก็ทำการกดที่ปุ่ม submit เพื่อที่จะทำการ login เข้าสู่ระบบ

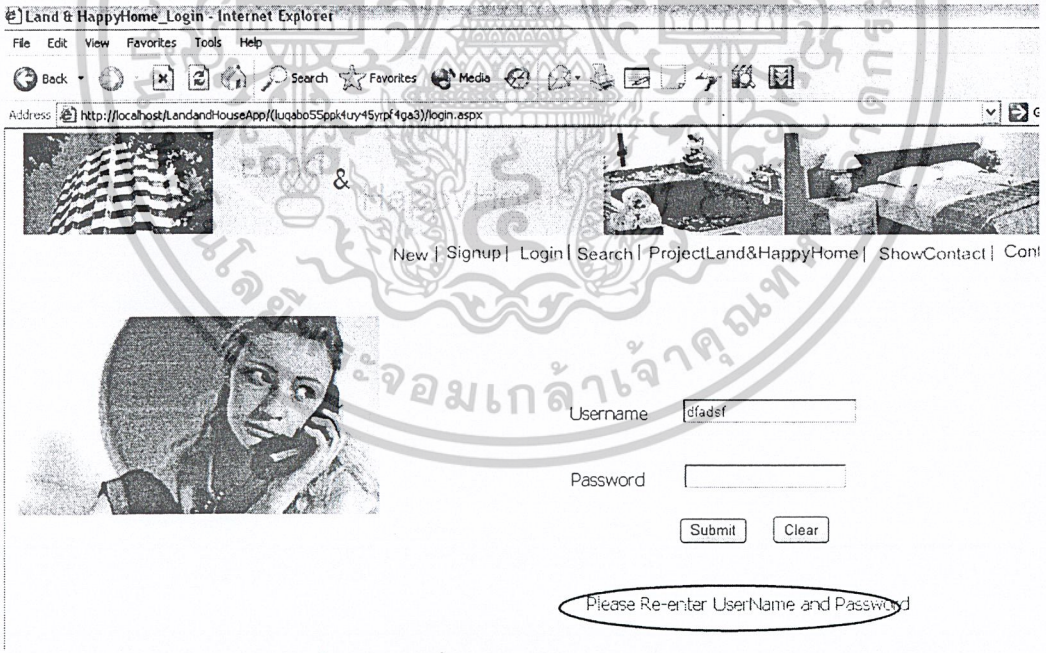
รูปที่ 9-16 ส่วนที่ login เข้าสู่ระบบโดยการกดปุ่ม submit

ถ้าเกิด login ได้ระบบจะนำเราเข้าสู่หน้าแรกของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 9-17 ถ้า login เข้าสู่ระบบเรียบร้อย

ถ้าเกิด login ไม่ถูกต้องระบบจะฟ้อง error มาดังรูป



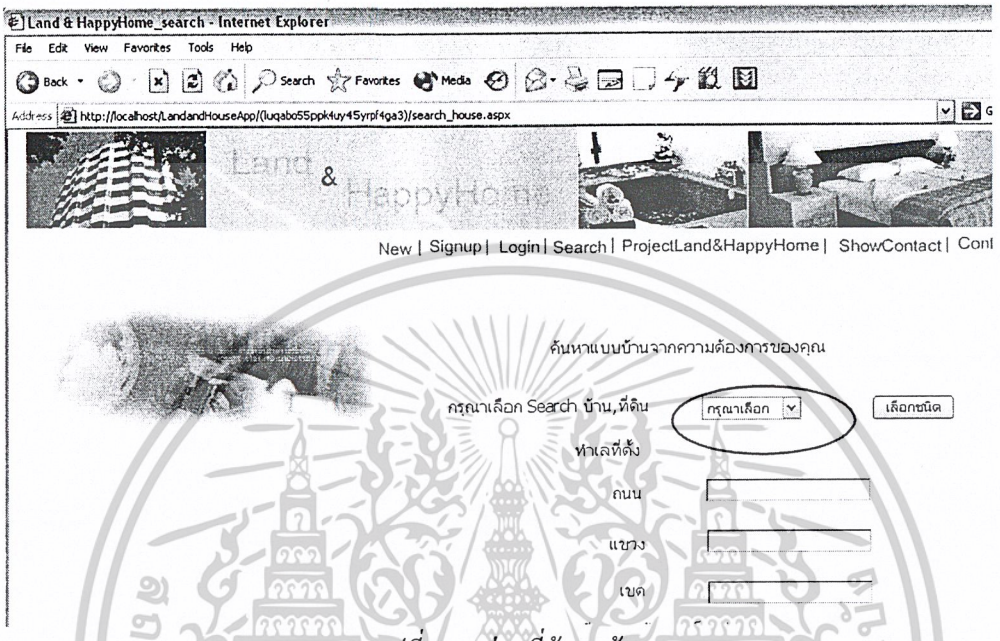
รูปที่ 9-18 ส่วนที่ login เข้าสู่ระบบแล้วไม่สมบูรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจากที่เราได้ทำการ login เรียบร้อยแล้วเราก็จะสามารถเข้าได้ทุก page ที่ต้องการได้เพราะถ้าอย่างนั้นเราก็ต้อง ทำการ login ก่อน

9.6.4 ขั้นตอนการ Search

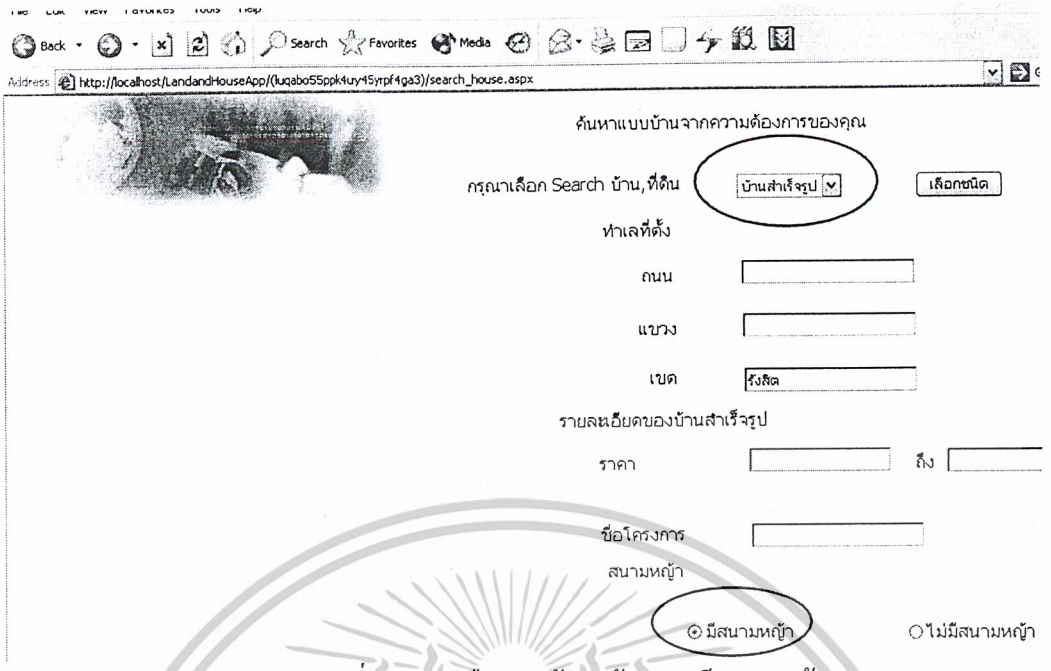
หลังจากนั้นเราก็จะทำการ search หาบ้านที่เราต้องการ โดยที่เราสามารถใส่ค่าทุกชนิดทุกอย่างที่ต้องการ search ลงไปใน page



รูปที่ 9-19 ส่วนที่ค้นหาข้อมูล

จากนั้นเราก็ต้องเลือกว่าจะเลือก search บ้านสำเร็จรูปหรือที่ดิน โดยที่ทำการเลือกที่ dropdownlist ว่าเราจะเลือกที่จะ search บ้านหรือที่ดิน จากในที่นี้เราเลือกที่จะ search บ้านสำเร็จรูป

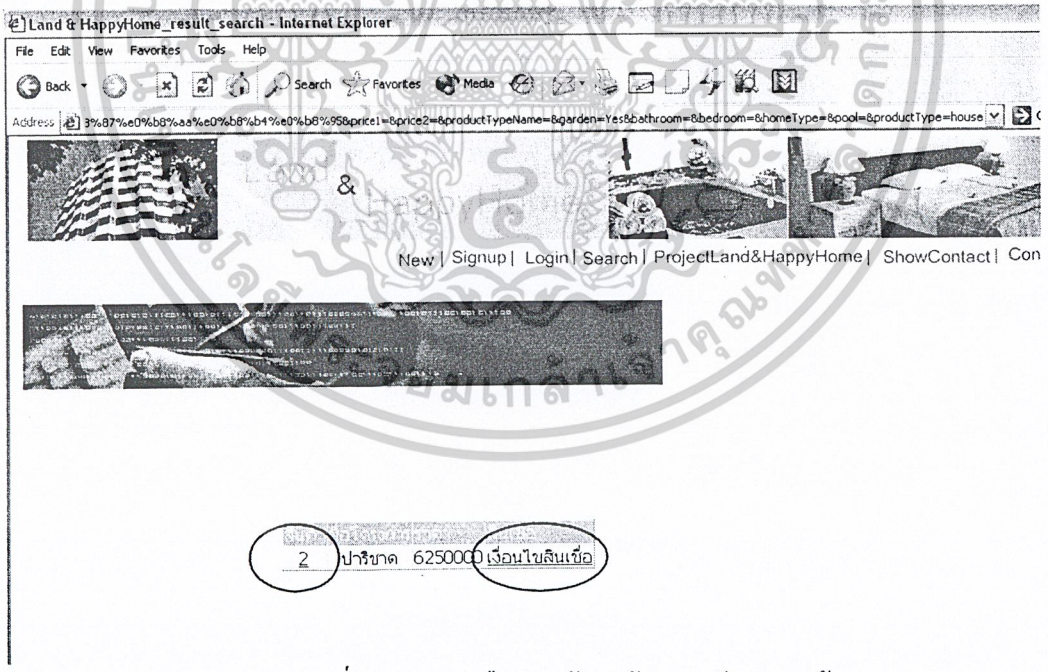
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-20 การเลือกการค้นหาบ้านแบบมีสนามหญ้า

9.6.5 ขั้นตอนการซื้อสินค้า

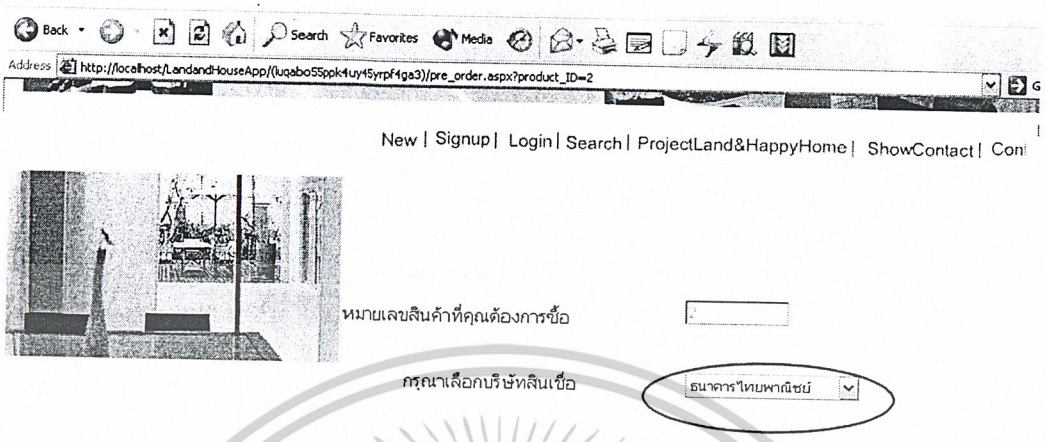
จากในที่นี้เราได้ทำการ search บ้านสำเร็จรูปที่อยู่ใน เขตรังสิต และมีสนามหญ้าจะทำให้เราได้ผลการ search ดังนี้



รูปที่ 9-21 ผลการเลือกการค้นหาบ้านแบบมีสนามหญ้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

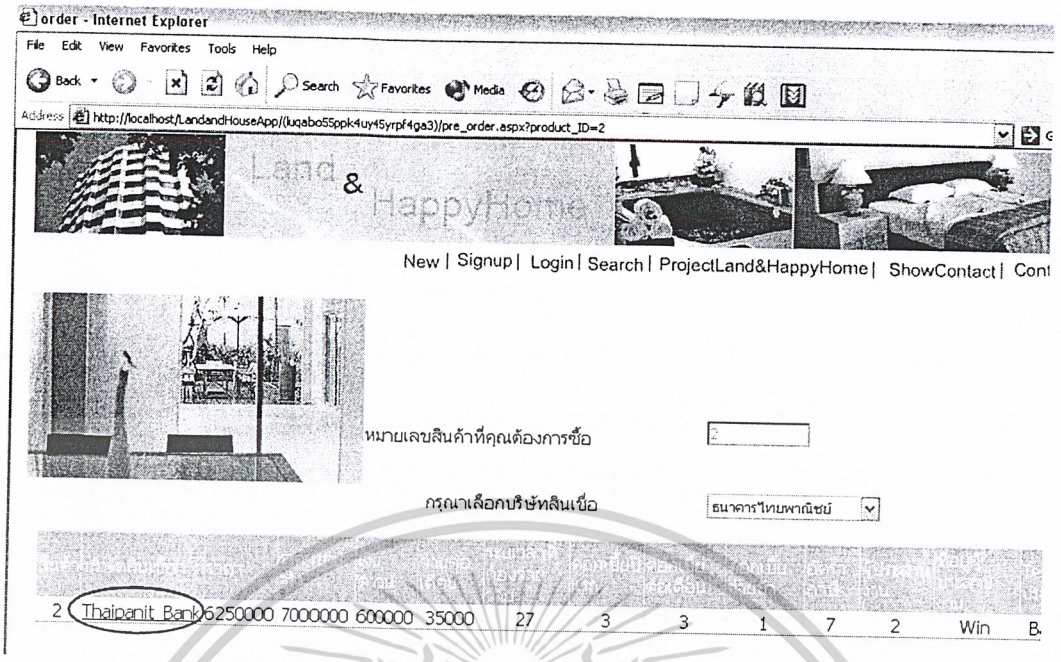
เราสามารถดูรายละเอียดของบ้านได้โดยที่ไปช่องสินค้า ในที่นี้จะเป็นสินค้าชนิดที่ 2 และเราสามารถดูเงื่อนไขสินเชื่อได้โดยการที่กดเลือกที่ช่องสินเชื่อ จะได้ดังรูป



รูปที่ 9-22 การเลือกดูสินเชื่อของแต่ละธนาคาร

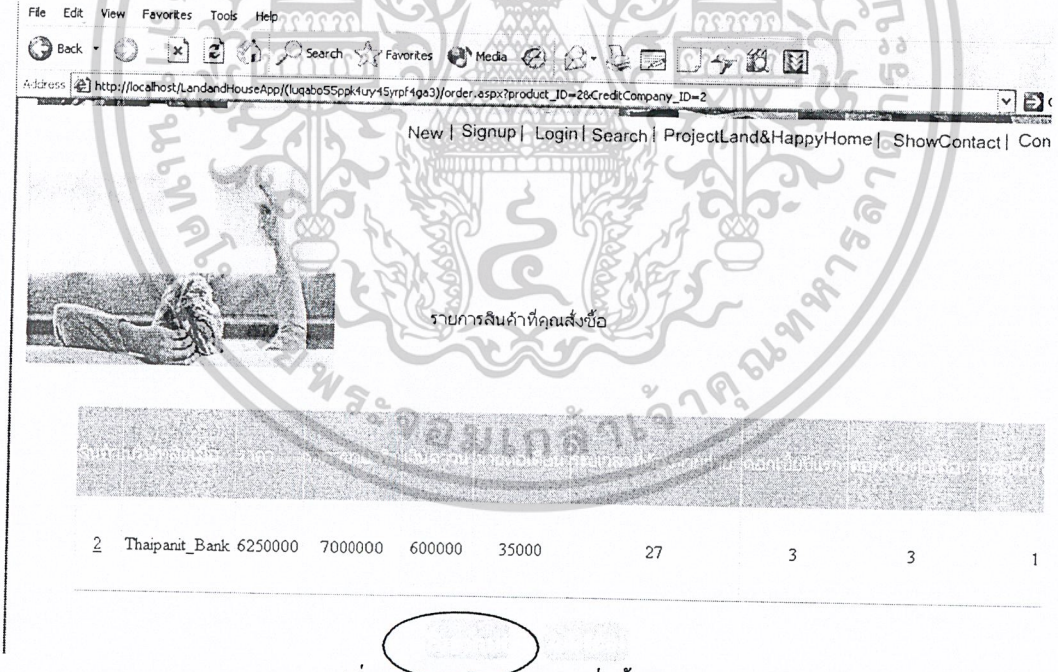
จากนั้นเราก็กดเลือกธนาคารที่เราต้องการทราบว่าอัตราการผ่อนเป็นอย่างไง ในที่นี้เรากดเลือกธนาคารไทยพานิชย์จากนั้นเราก็กดเลือกปุ่ม Submit เพื่อที่จะได้ดูรายละเอียด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-23 การเลือกดูสินเชื่อของธนาคาร

กดเลือกที่ช่องบริษัทสินเชื่อเพื่อเป็นการยืนยันการผ่อนชำระว่าเราต้องการที่จะผ่อนกับบริษัทนี้ จากนั้นระบบก็จะทำการแสดงรายละเอียดออกมาให้

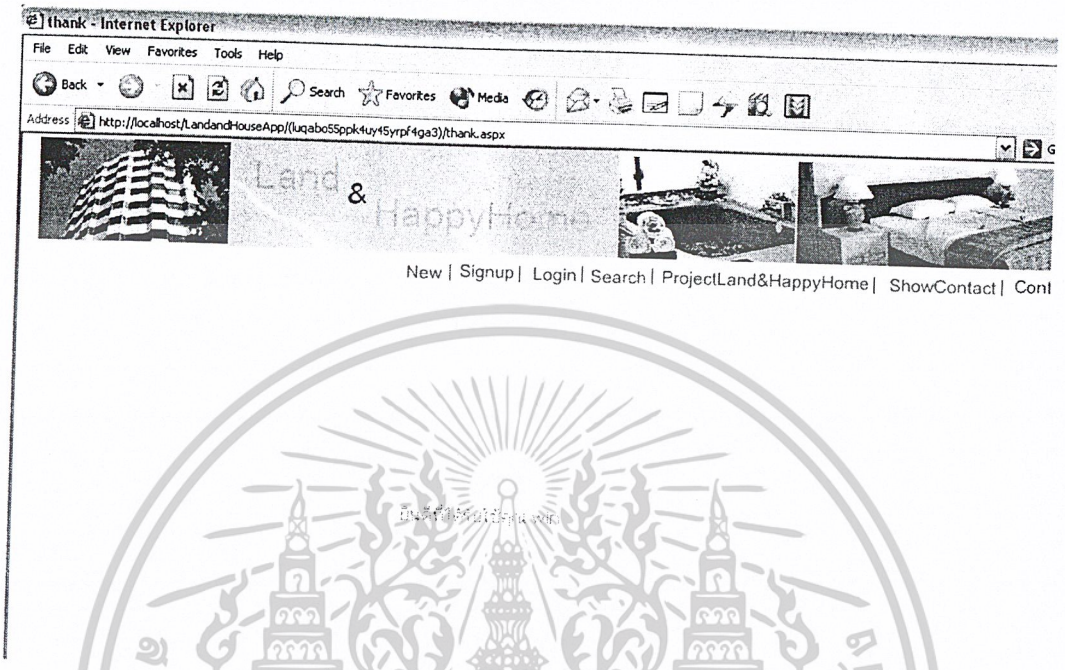


รูปที่ 9-24 การยืนยันในการสั่งซื้อสินค้ากับธนาคาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.6.6 ขั้นตอนการยืนยันในการสั่งซื้อ

กด submit เพื่อเป็นการยืนยันว่าเราต้องการซื้อ



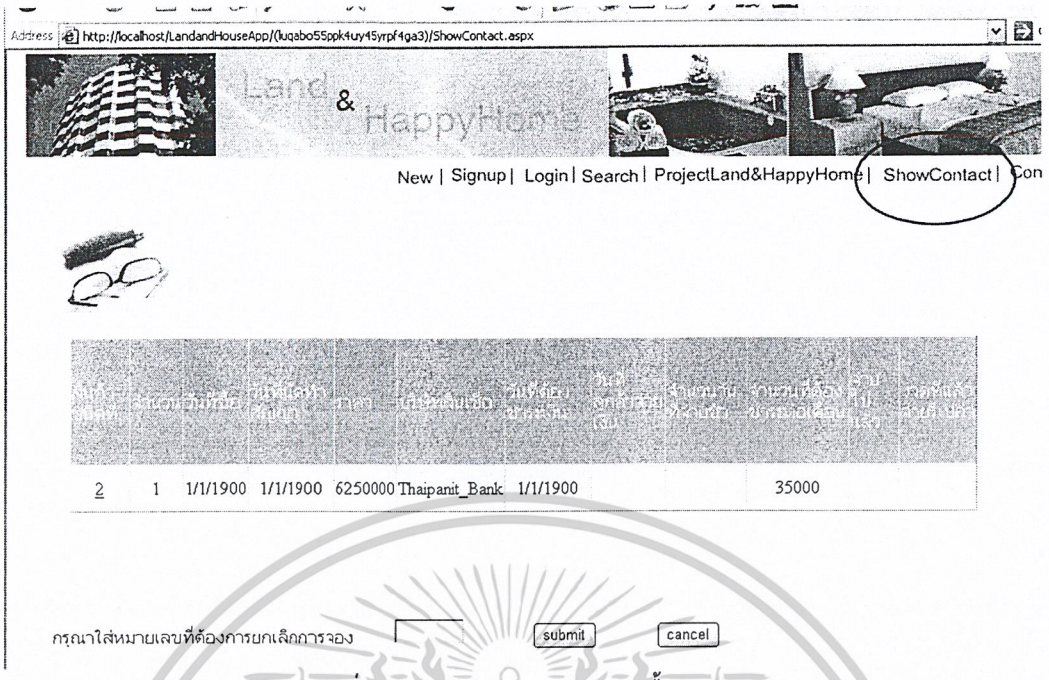
รูปที่ 9-25 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย

จากนั้นระบบจะนำเราเข้าสู่หน้า thank.aspx เพื่อเป็นการขอบคุณ

9.6.7 ขั้นตอนการดูรายละเอียดของสินค้าที่ได้ซื้อไป

เราสามารถดูว่าเราได้ซื้อสินค้าอะไรไปแล้วบ้างโดยที่กดที่ ShowContact

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



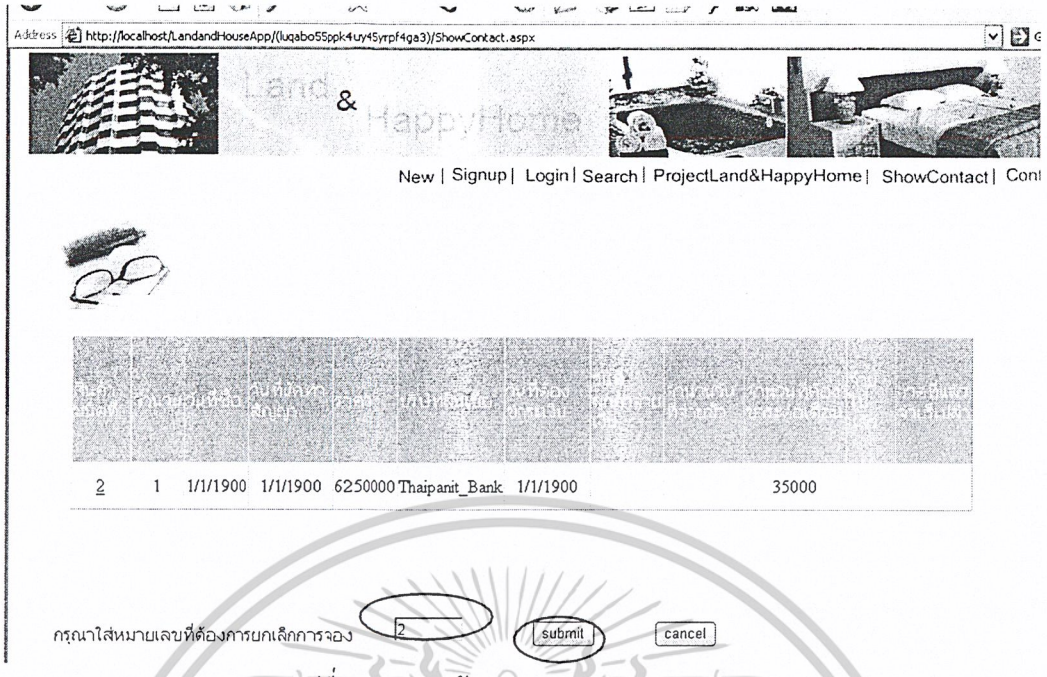
รูปที่ 9-26 หน้าจอรายละเอียดการซื้อสินค้า

จะทำให้เราได้เห็นว่าเราได้ซื้อสินค้าอะไรไปแล้วบ้าง ในที่นี้เราได้ซื้อสินค้าไปแล้ว 1 ชิ้นคือ สินค้าชนิดที่ 2 จำนวน 1 ชิ้น

9.6.8 ขั้นตอนการ cancel การสั่งซื้อ

และถ้าต้องการที่จะ cancel contact เราก็สามารถทำได้โดยเราใส่สินค้าชนิดที่ลงไปในช่องแล้ว click ปุ่ม submit เพื่อเป็นการยืนยันว่าเราต้องการที่จะ cancel contact

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

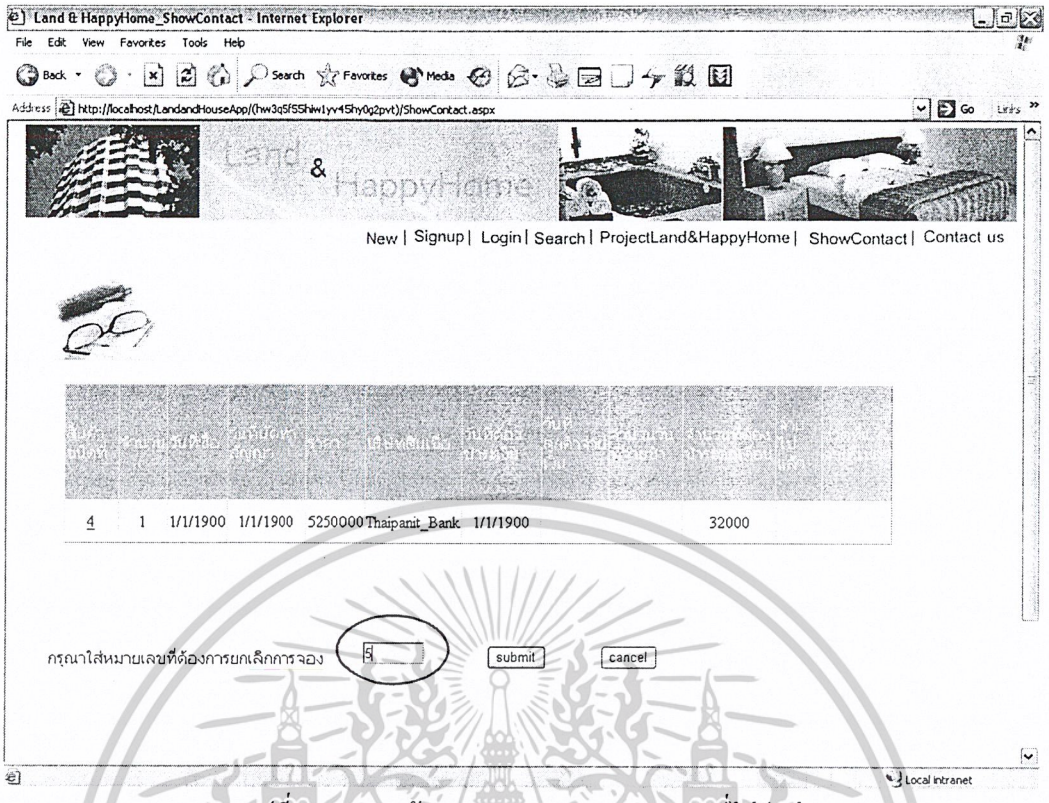


รูปที่ 9-27 การหน้าจอแสดงการ cancel contact

เมื่อ cancel และจะได้ดังรูป

รูปที่ 9-28 การหน้าจอแสดงการ cancel contact เมื่อสมบรูณ์

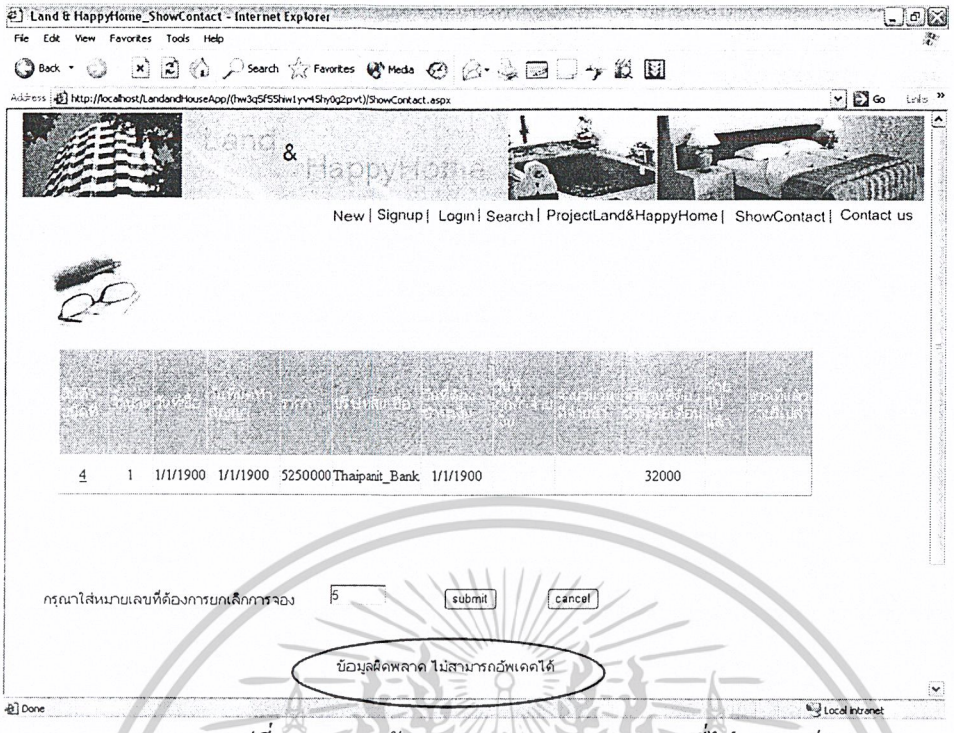
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-29 การหน้าจอแสดงการ cancel contact ที่ไม่มีอยู่

แต่ถ้าเรา cancel ผิด โดยที่ในรูปเรามีแค่สินค้าชนิดที่ 4 แต่เราจะไป cancel สินค้าชนิดที่ 5 ทำให้ไม่สามารถทำได้ ระบบจะทำการฟ้อง error ออกมาดังรูป ทำให้เราไม่สามารถ cancel ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9-30 การหน้าจอแสดงการ cancel contact ที่ไม่สมบูรณ์

ระบบจะทำการฟ้อง error ออกมาดังรูป ทำให้เราไม่สามารถ cancel ได้

9.7 ฟังก์ชันที่เปิดให้บริการเป็นเว็บเซอร์วิส

9.7.1. การสมัครสมาชิกของลูกค้าใหม่

Function: boolean signup(String name,String surname,String sex,String address,String telephone,String email, String job, String salary ,String username, String password)

Output: ถ้าสำเร็จ return true

ถ้าไม่สำเร็จ return false

9.7.2. เป็นการตรวจสอบว่า ชื่อและพาสเวิร์ดได้มีคนสมัครไปแล้วหรือยัง

Function: String login(String user,String pass)

Output: ถ้าสำเร็จ return id

ถ้าไม่สำเร็จ return “”

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.7.3. การหารหัสของสินค้าที่ชื่อที่เราต้องการทราบ

Function: Dataset search(String street, String district, String aumper, String price1, String price2, String productTypeName, String garden, String bathroom, String bedroom, String homeType, String pool, String productType)

Output: ถ้าสำเร็จ return Dataset

9.7.4. ทำการสั่งซื้อสินค้าตามรายการ

Function: Dataset order(String product_ID,String CreditCompany_ID)

Output: ถ้าสำเร็จ return Dataset

9.7.5. ดูใบเสร็จสินค้าที่ต้องชำระเงิน

Function: Dataset showcontact (String Customer_ID)

Output: ถ้าสำเร็จ return dataset

9.7.6.ดูรายละเอียดของบริษัทสินเชื่อ

Function: Dataset showCredit(String product_ID, String CreditCompanyName)

Output: ถ้าสำเร็จ return dataset

9.7.7.การตกลงที่จะซื้อสินค้าชนิดนั้น

Function: Dataset submit(String Customer_ID, String SaleDate, String SaleType ,String ContactDate, String SalePrice, String CreditCompanyName, String PaidDate, String PaidPerMonth ,String Product_ID)

Output: ถ้าสำเร็จ return dataset

9.7.8.การยกเลิกการสั่งซื้อ

Function: boolean CancelContact (String Customer_ID,String Product_ID)

Output: ถ้าสำเร็จ return true

ถ้าไม่สำเร็จ return false

9.7.9.การจ่ายเงิน

Function: boolean pay (String product_ID)

Output: ถ้าสำเร็จ return true

ถ้าไม่สำเร็จ return false

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9.7.10.การแสดงรายละเอียดของการจ่ายเงิน

Function: Dataset showdetailforpay (String product_ID, String customer_ID)

Output: ถ้าสำเร็จ return dataset

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 10
แอพพลิเคชั่นระบบ Olala Home Services

10.1 Overview ของระบบ Olala Home Services

ระบบ OLALA Home Services เป็นระบบจัดการการซื้อขายบ้านแบบสำเร็จรูป ที่เป็นเว็บเซอร์วิสให้บริการจัดเกี่ยวกับการซื้อขายบ้านสำเร็จรูป, ขายสินค้าตกแต่งบ้าน, ขายแบบบ้านและรับสร้างบ้าน โดยผู้ใช้บริการสามารถค้นหาข้อมูลต่างๆได้ตามต้องการ โดยกรอกรายละเอียดที่จำเป็นในการค้นหาลงไปเช่น ต้องการที่อยู่แถวไหน, มีห้องนอนกี่ห้อง, มีห้องน้ำกี่ห้อง, ราคาบ้านเท่าไหร่ หรือมีสินค้าตกแต่งบ้านชนิดนั้นอยู่หรือไม่ จากนั้นทางระบบจะเรียกใช้เซอร์วิสต่างๆ ที่ เว็บเซอร์วิสมีให้เรียกใช้

เมื่อผู้ใช้บริการทำการจองเรียบร้อยแล้ว จะสามารถเข้าตรวจสอบดูรายละเอียด, เปลี่ยนแปลงและยกเลิกการจองครั้งนั้นๆได้ ส่วนเรื่องการชำระค่าบริการจะเป็นการชำระผ่านบัตร

ในปริญญานิพนธ์ฉบับนี้จะแบ่งการออกแบบบริการต่างๆเป็น 2 เทคโนโลยี คือ เทคโนโลยีของ .NET เฟรมเวิร์ค และ เทคโนโลยีของจาวา เพื่อทำการศึกษาการเรียกใช้คอมโพเนนต์ข้ามสถาปัตยกรรม โดยทำการติดต่อสื่อสารกันผ่านโปรโตคอล SOAP

เทคโนโลยีของ .NET เฟรมเวิร์คประกอบด้วย

- ระบบขายบ้านสำเร็จรูปและที่ดิน
- ระบบขายแบบแปลนบ้านและรับสร้างบ้าน

เทคโนโลยีของจาวาจะประกอบด้วย

- ระบบขายสินค้าตกแต่งบ้าน

รายละเอียดของแอพพลิเคชั่น Home&Decor

แอพพลิเคชั่น Home&Décor เป็นระบบที่ให้บริการเกี่ยวกับการเลือกซื้อสินค้าตกแต่งบ้าน ทั้งผ่านทางหน้าเวบโดยตรง และผ่านทางเวบเซอร์วิส โดยสินค้าที่นำเสนอผ่านทางเวบนั้น จะมีการนำความรู้ในด้านเทคโนโลยี VRML มาประกอบด้วย ดังนั้น สินค้าที่แสดงจะมีทั้งที่เป็นรูปสองมิติ และ ที่เป็นสามมิติ โดยใช้เทคโนโลยี VRML ซึ่งรายการสินค้าที่แสดงในแต่ละหมวดหมู่นั้นจะเป็นการสุ่มขึ้นมาแสดง นั่นคือการเข้ามาชมสินค้าแต่ละครั้ง จะเห็นสินค้าที่แสดงเหมือนเดิม หรือ ต่างจากเดิมก็ได้

บริการของระบบ Home&Décor ประกอบด้วย

1. การลงทะเบียน
2. การค้นหาข้อมูลสินค้า ดังหัวข้อต่อไปนี้
   2.1. ค้นหาเครื่องใช้ตามหมวดหมู่(แผนก)สินค้า
   2.2. ค้นหาเครื่องใช้ตามคำที่ต้องการ
3. การดูรายการสินค้าที่สั่งซื้อ
4. การแก้ไขรายการสั่งซื้อที่ทำกับระบบก่อนมีการชำระเงิน
5. การสั่งซื้อสินค้าผ่านทางระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. เกร็ดความรู้ในการตกแต่งบ้าน เป็นการดูเกร็ดความรู้ผ่านทางเวบเพจ Home&Décor เท่านั้น ไม่ให้บริการผ่านเวบเซอร์วิส
7. การทำงานในส่วนของผู้ดูแลระบบ

รายละเอียดของแอพพลิเคชั่น Home For You

แอพพลิเคชั่น Home For You เป็นระบบที่ให้บริการเกี่ยวกับการเลือกซื้อแบบบ้านและรับสร้างบ้าน ทั้งผ่านทางหน้าเวบโดยตรง และผ่านทางเวบเซอร์วิส บริการของระบบ Home For You ประกอบด้วย

โดยจะมีเซอร์วิสที่เปิดไว้เพื่อให้เว็บอื่นรวมถึงเว็บแอพลิเคชันได้เรียกใช้ ได้แก่

1. เซอร์วิสที่เกี่ยวกับการสั่งซื้อแบบบ้าน
2. เซอร์วิสที่เกี่ยวกับการสร้างบ้าน
3. เซอร์วิสที่เกี่ยวกับการค้นหาแบบบ้าน
4. เซอร์วิสที่เกี่ยวกับระบบสมาชิก
5. เซอร์วิสที่เกี่ยวกับการนำเอาโมเดลบ้านไปแสดงบนหน้าเว็บ
6. เซอร์วิสที่เกี่ยวกับการดูราคาของแบบบ้าน

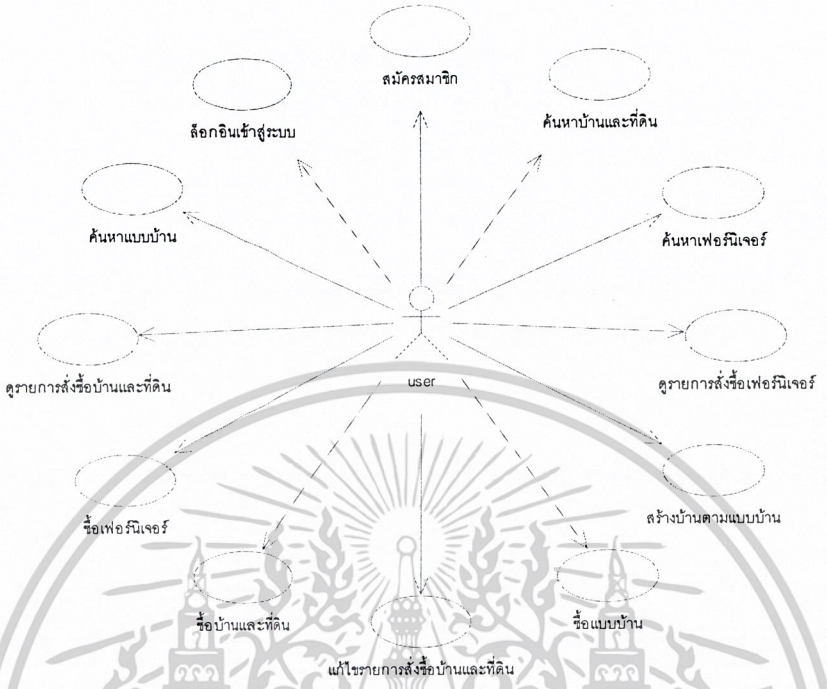
รายละเอียดของแอพพลิเคชั่น Land and Home

แอพพลิเคชั่น Land and Home เป็นระบบที่ให้บริการเกี่ยวกับการเลือกซื้อบ้านสำเร็จรูปและที่ดิน ทั้งผ่านทางหน้าเวบ โดยตรง และผ่านทางเวบเซอร์วิส บริการของระบบ Land and Home ประกอบด้วย

1. การลงทะเบียน
2. การสมัครสมาชิก
3. การค้นหาข้อมูลสินค้า ซึ่งค้นหาบ้านสำเร็จรูปและที่ดินตามคำที่ต้องการ
4. การดูรายการสินค้าที่สั่งซื้อ
5. การแก้ไขรายการสั่งซื้อที่ทำกับระบบก่อนมีการชำระเงิน
6. การชำระเงิน
7. การสั่งซื้อสินค้าผ่านทางระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.2 Use Case Diagram ระบบ Olala Home Services

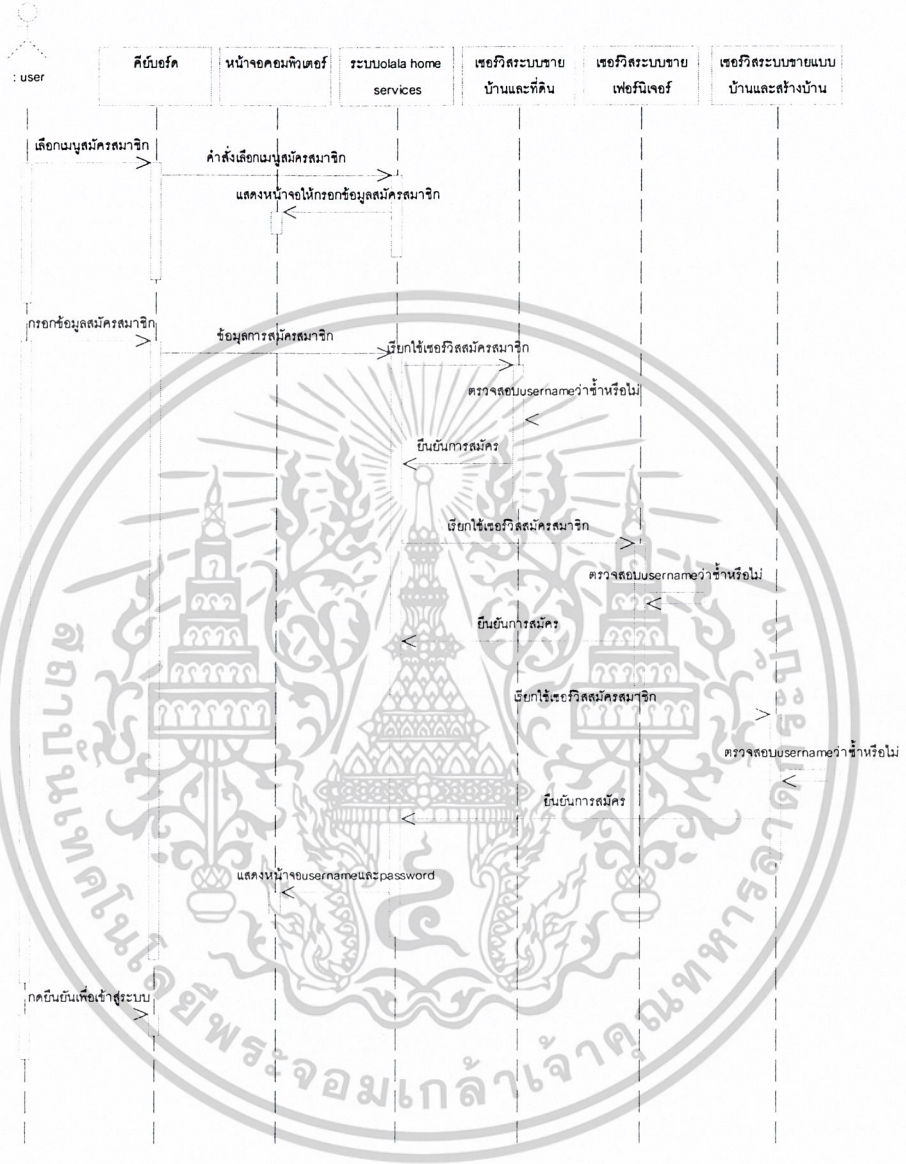


รูปที่ 10-1 Use Case diagram ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3 Sequence Diagram ของระบบ Olala Home Services

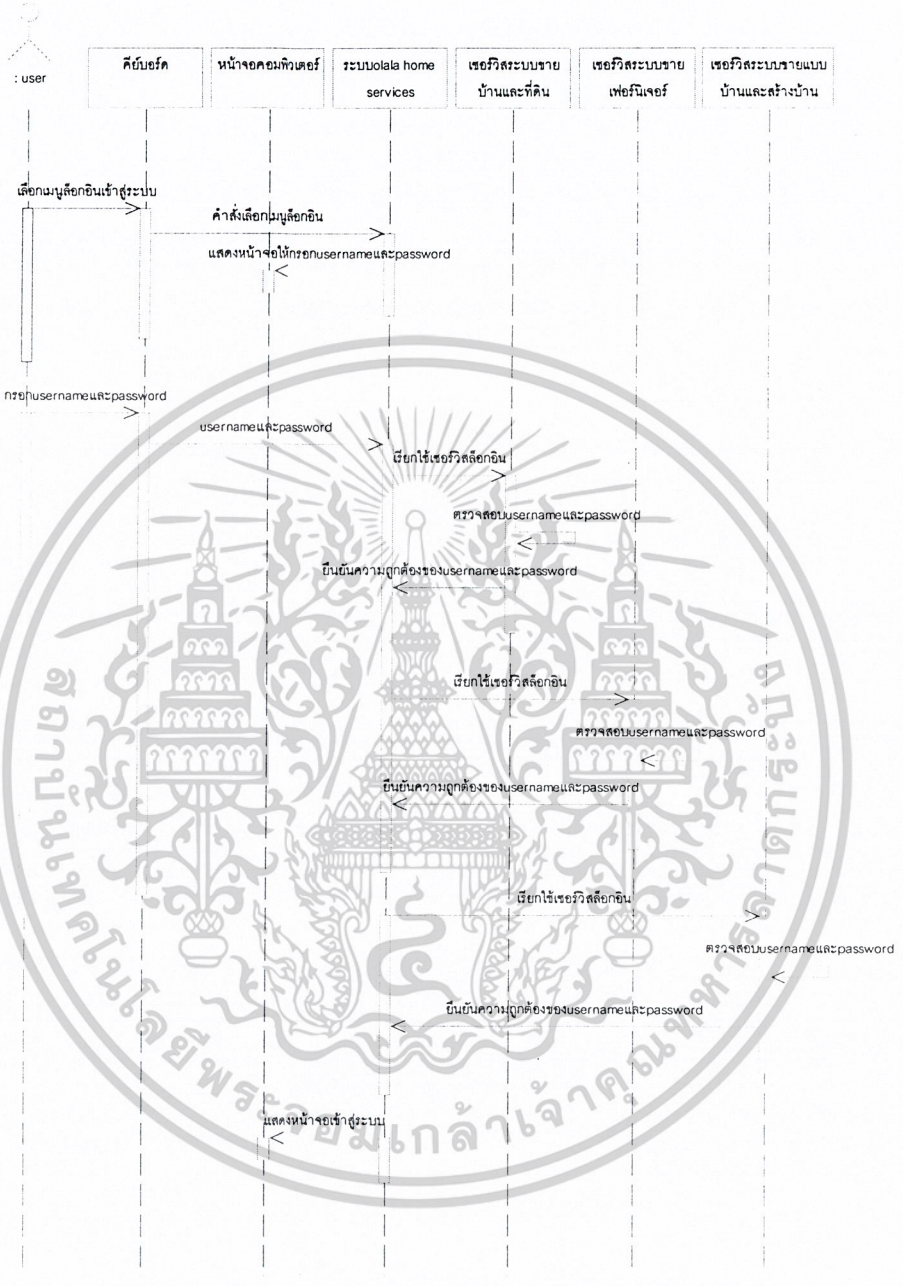
10.3.1 การสมัครสมาชิก



รูปที่ 10-2 Sequence Diagram การสมัครสมาชิก ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

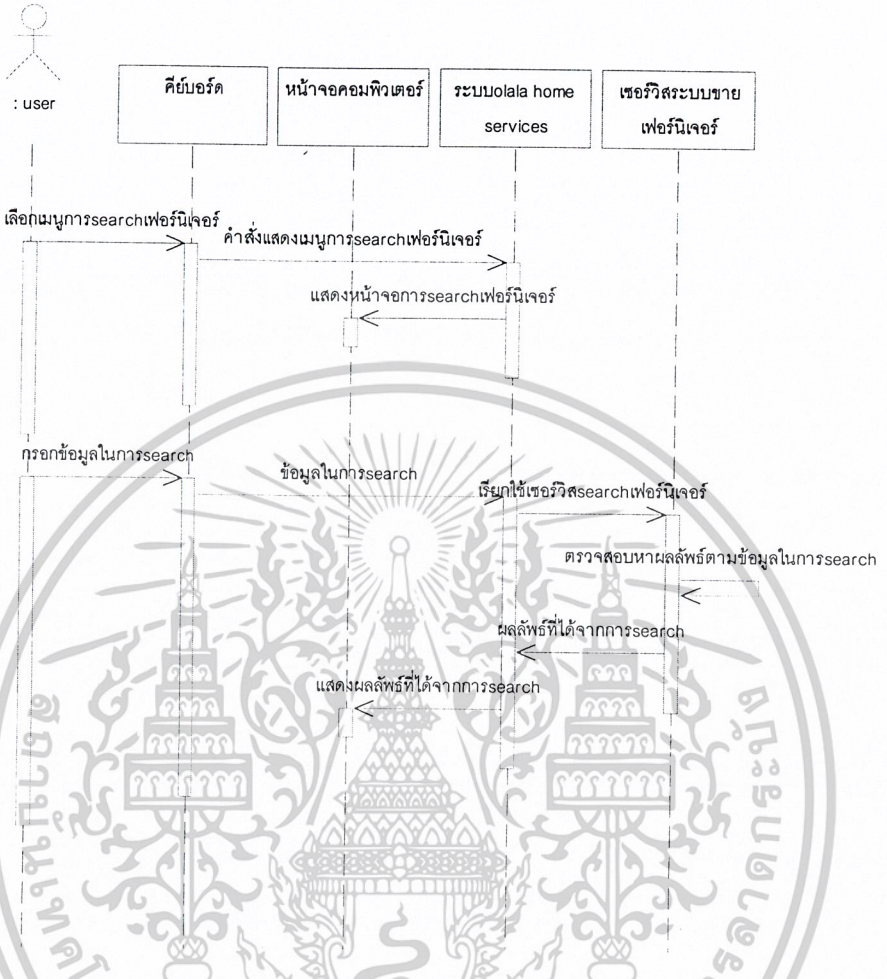
10.3.2 การล็อกอินเข้าสู่ระบบ



รูปที่ 10-3 Sequence Diagram การล็อกอิน ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

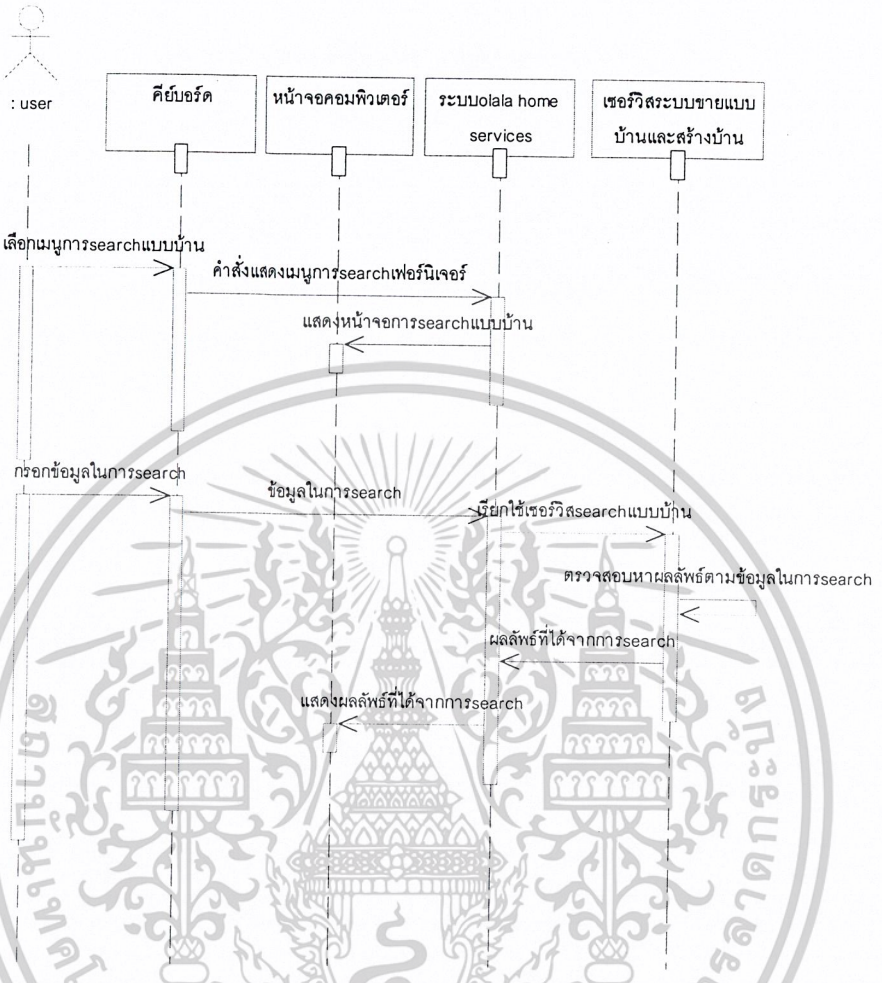
10.3.3 การค้นหาเฟอร์นิเจอร์



รูปที่ 10-4 Sequence Diagram การค้นหาเฟอร์นิเจอร์ ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

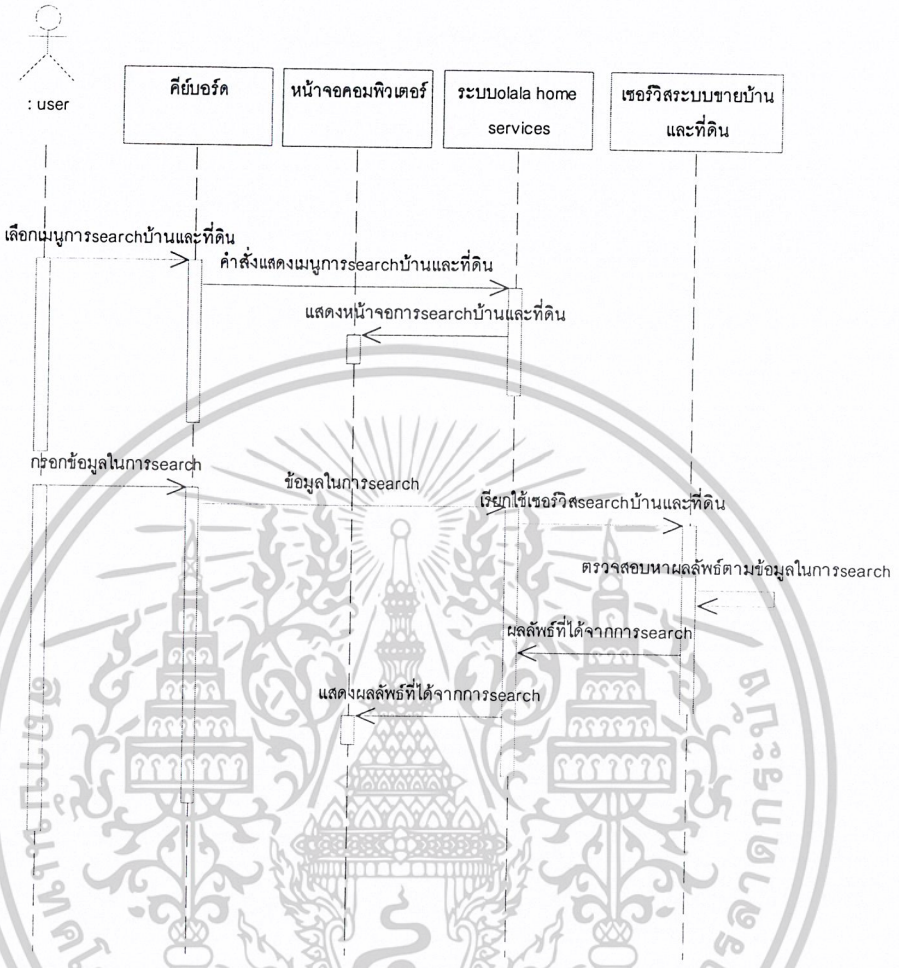
10.3.4 การค้นหาแบบแปลนบ้าน



รูปที่ 10-5 Sequence Diagram การค้นหาแบบแปลนบ้าน ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

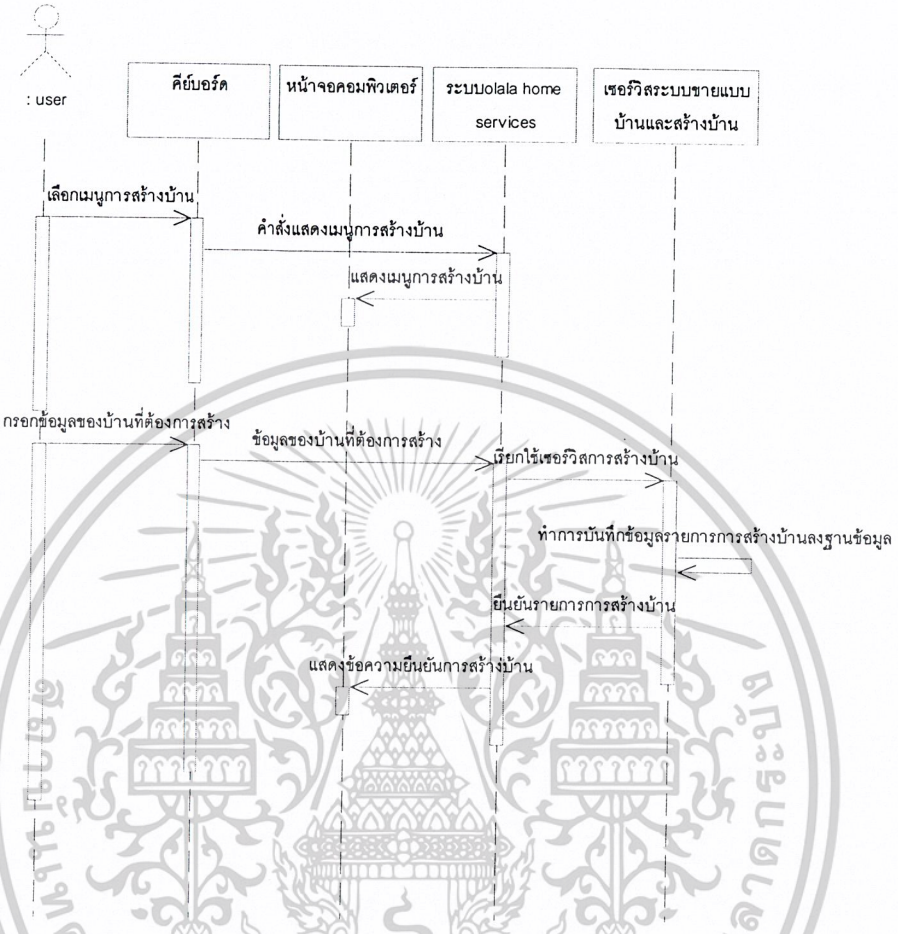
10.3.5 การค้นหาบ้านและที่ดิน



รูปที่ 10-6 Sequence Diagram การค้นหาบ้านและที่ดิน ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3.6 การสร้างบ้าน



รูปที่ 10-7 Sequence Diagram การสร้างบ้าน ของระบบ Olala Home Services

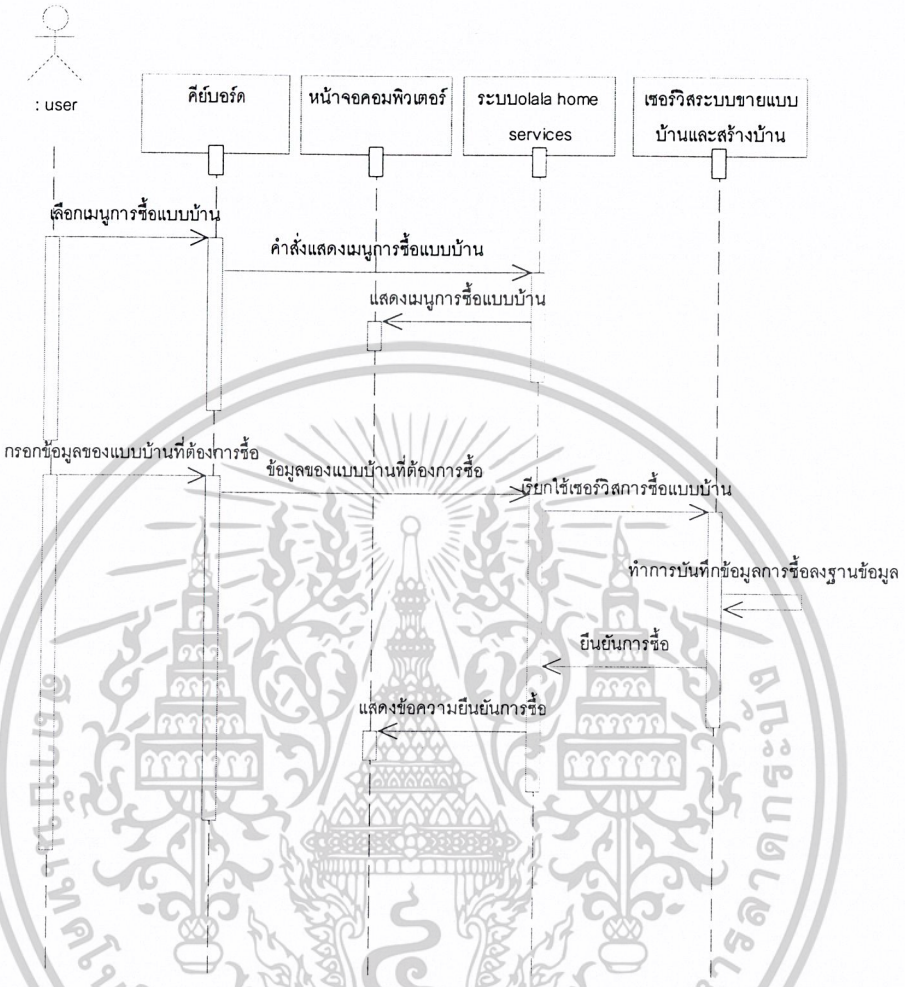
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3.7 การซื้อเฟอร์นิเจอร์

รูปที่ 10-8 Sequence Diagram การซื้อเฟอร์นิเจอร์ ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

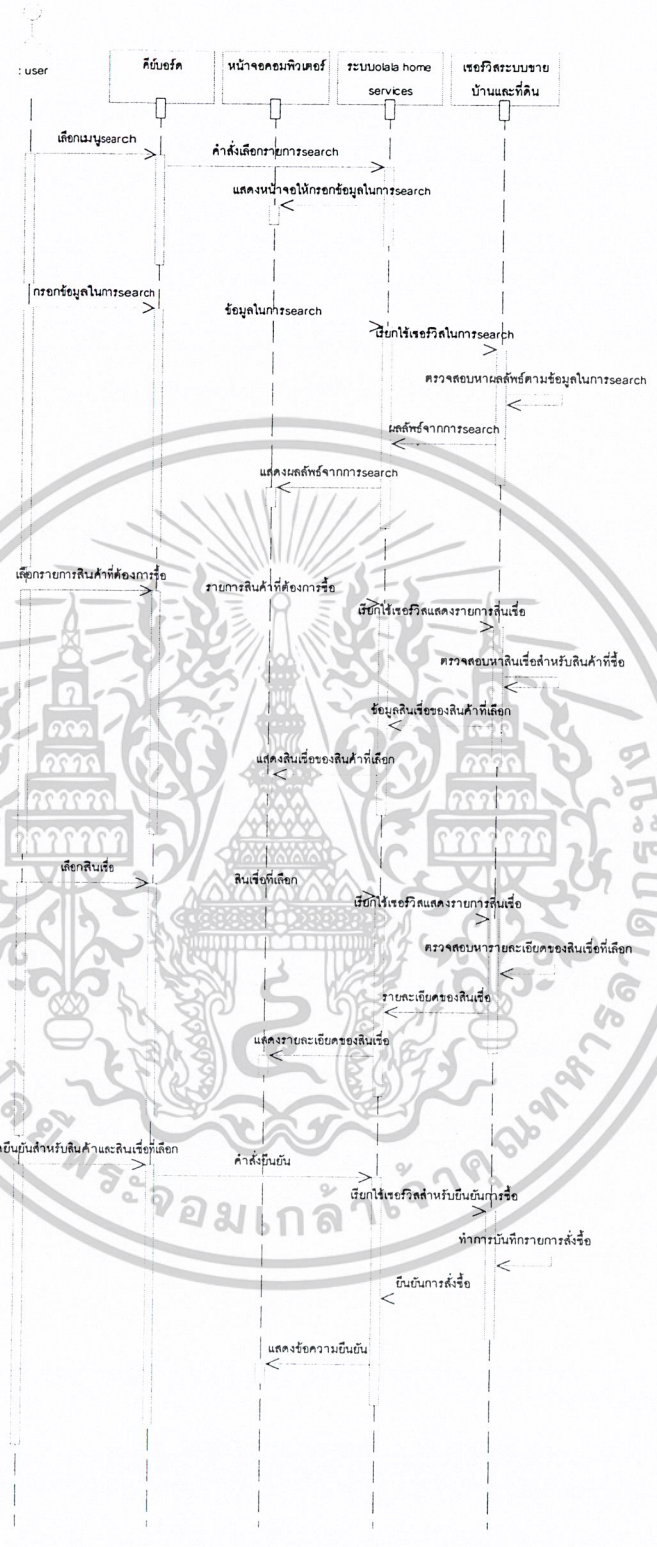
10.3.8 การซื้อแบบแปลนบ้าน



รูปที่ 10-9 Sequence Diagram การซื้อแบบแปลนบ้าน ของระบบ Olala Home Services

10.3.9 การซื้อบ้านและที่ดิน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-10 Sequence Diagram การซื้อบ้านและที่ดิน ของระบบ Olala Home Services

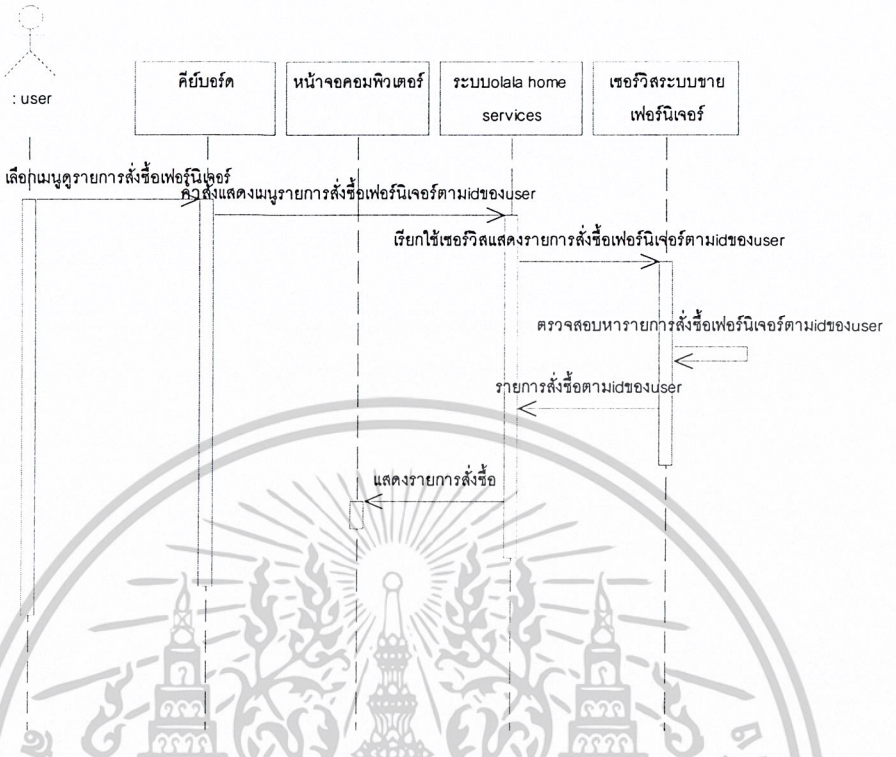
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3.10 การยกเลิกรายการสั่งซื้อบ้านและที่ดิน

รูปที่ 10-11 Sequence Diagram การยกเลิกรายการสั่งซื้อบ้านและที่ดิน ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

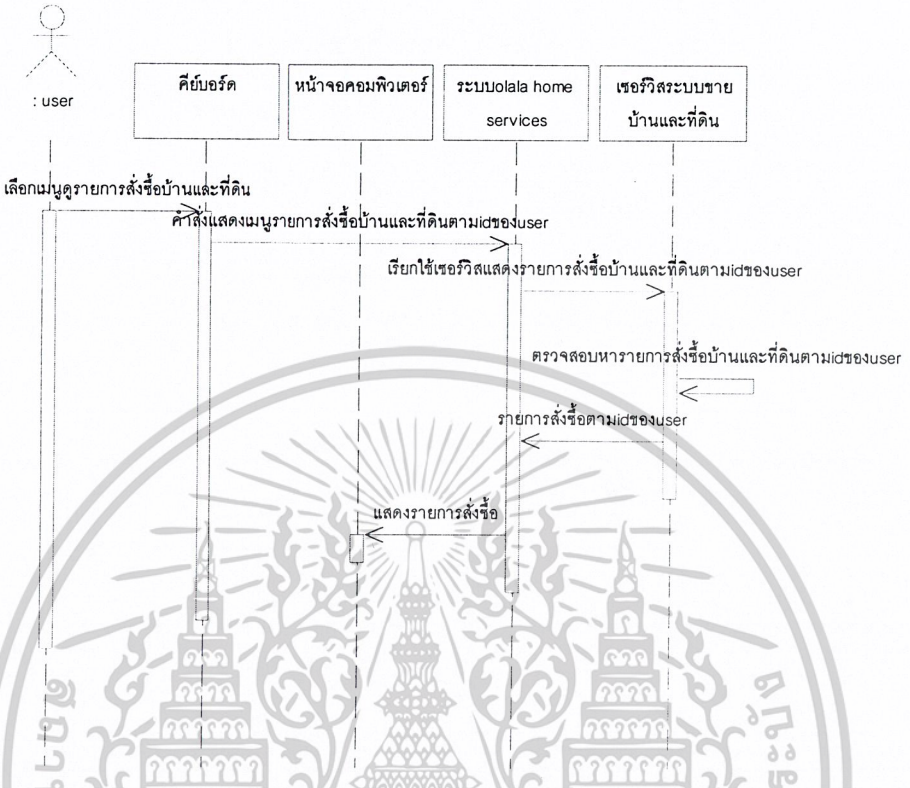
10.3.11 การดูบิลของการสั่งซื้อเฟอร์นิเจอร์



รูปที่ 10-12 Sequence Diagram การดูบิลของการสั่งซื้อเฟอร์นิเจอร์ ของระบบ Olala Home Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.3.12 การดูรายการการสั่งซื้อบ้านและที่ดิน



รูปที่ 10-13 Sequence Diagram การดูรายการการสั่งซื้อบ้านและที่ดิน ของระบบ Olala Home Services

10.4 คู่มือการใช้งานระบบ Olala Home Services

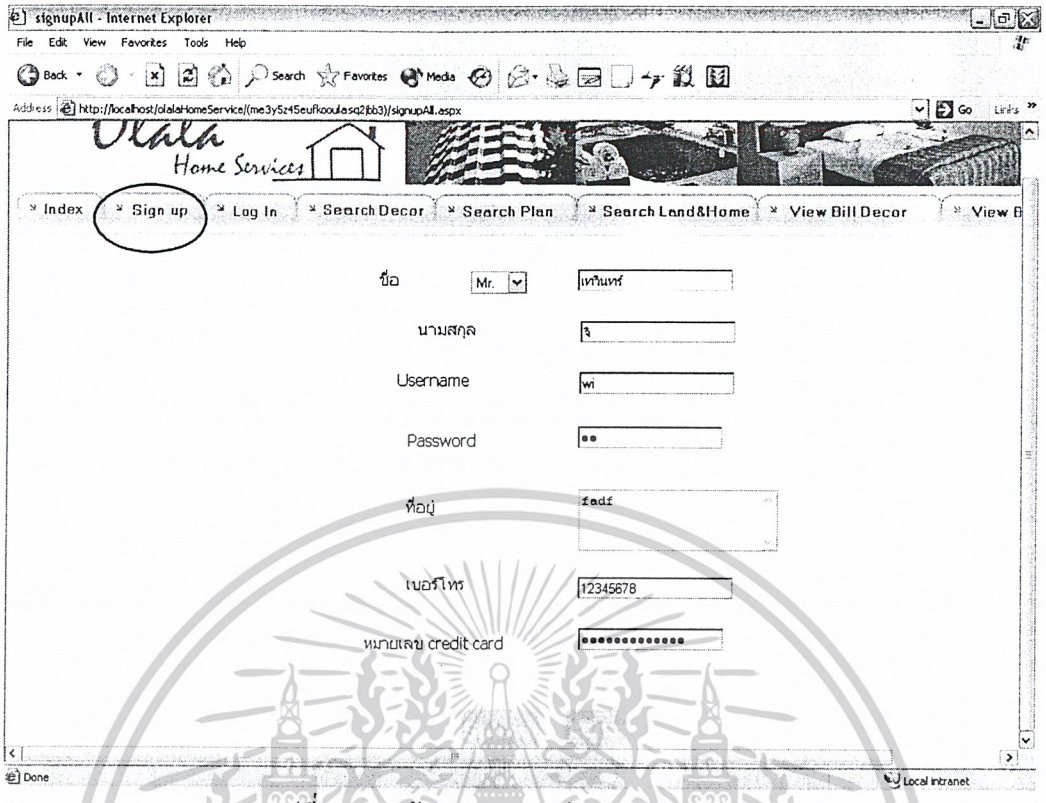
10.4.1 ขั้นตอนเริ่มการใช้งาน

เริ่มต้นการใช้งานโดยที่เราจะเข้าไปที่ index.html เพื่อที่จะทำการซื้อสินค้าต่าง ๆ

10.4.2 ขั้นตอนการ Signup

เราก็เข้าไปที่ page signup เพื่อที่จะทำการสมัครเป็นสมาชิกก่อนเพราะว่าบาง page ของเราต้องการที่จะให้ page นั้นเข้าถึงได้เฉพาะสมาชิกเท่านั้น จากนั้นเราก็กรอกข้อมูลให้ครบตามที่ต้องการ

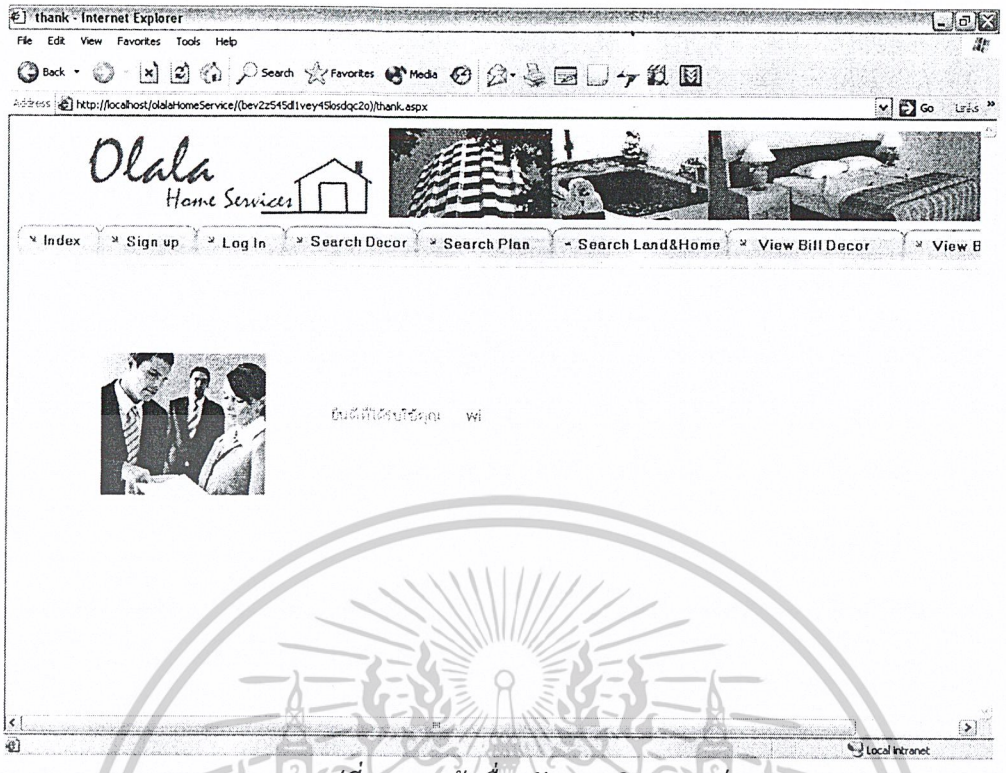
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-14 หน้ากรอกรายละเอียดการสมัครสมาชิก

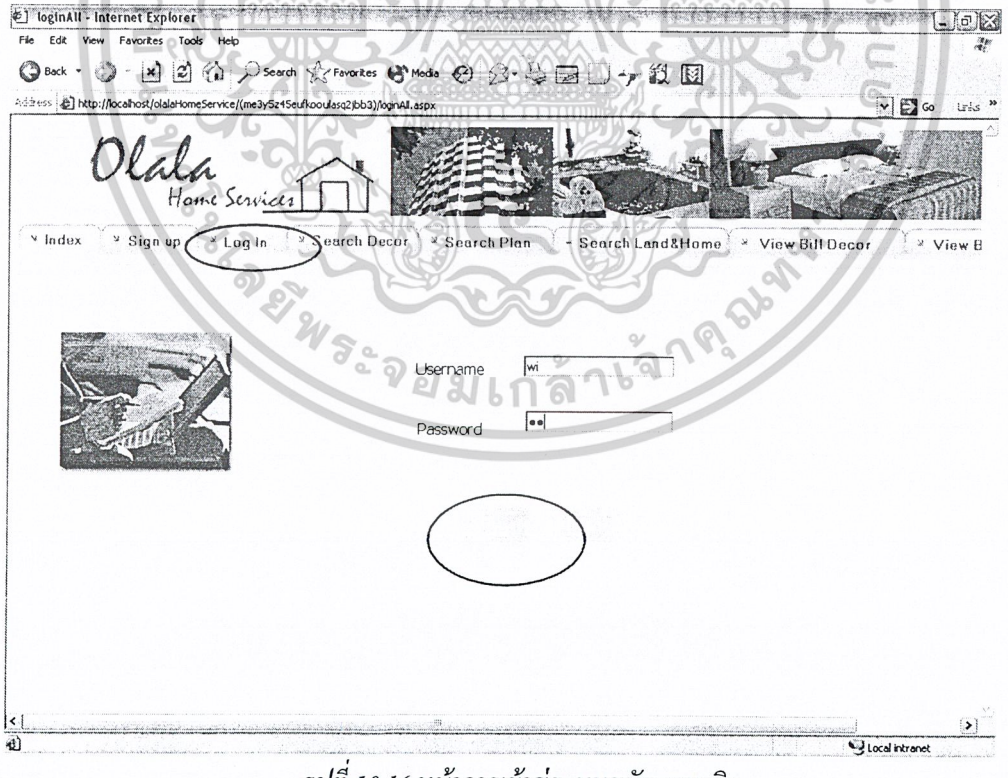
เมื่อทำการ signup เรียบร้อยแล้วระบบจะทำการ redirect ไปที่ page thank.aspx เพื่อที่จะทำการแสดงว่าระบบได้ทำการ signup ให้คุณเรียบร้อยแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-15 หน้าเมื่อสมัครสมาชิกสมบรูณ์

จากนั้นเราก็ทำการ login เพื่อที่ทำการเข้าสู้ระบบโดย click ที่ page Login เพื่อที่จะทำการให้ user login โดยให้ user กรอกข้อมูลคือ username และ password จากนั้นก็กดปุ่ม submit



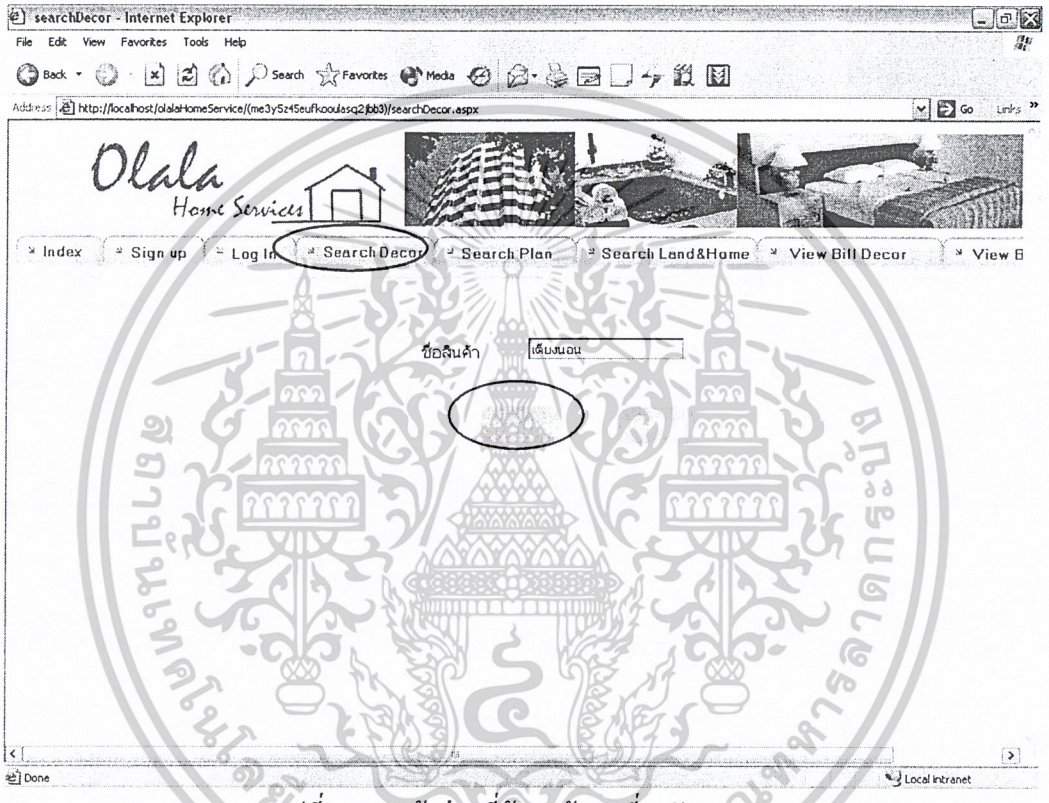
รูปที่ 10-16 หน้าการเข้าสู่ระบบสมัครสมาชิก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อถ้า login ได้แล้วระบบจะนำเราเข้าสู่หน้า index.aspx เพื่อให้เราสามารถเข้าไปทำกิจกรรมต่าง ๆ ของระบบได้

10.4.3 ขั้นตอนการ search décor

โดยเริ่มจากการที่ให้เราไป click ที่ปุ่ม search decor จากนั้นเราก็ทำการค้นหาสินค้าที่ต้องการซื้อ ในตัวอย่างนี้เราต้องการที่จะค้นหาสินค้าที่มีชื่อว่า "เตียงนอน" เมื่อกรอกลงไปในช่องแล้วให้กดปุ่ม submit



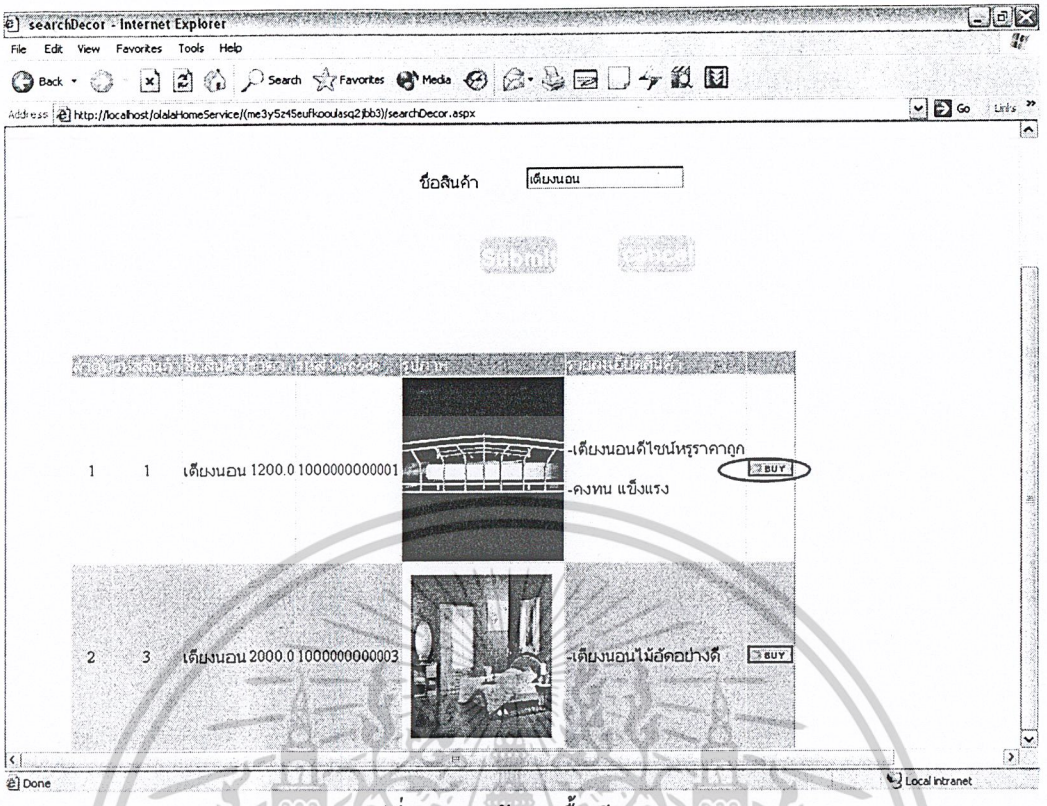
รูปที่ 10-17 หน้าส่วนที่ค้นหาข้อมูลเกี่ยวกับ Decor

ระบบจะแสดงข้อมูลต่าง ๆ ของสินค้าชนิดนั้นออกมาดังรูป ทำให้เราทราบว่าสินค้าชนิดนั้นมีรายละเอียดอะไรบ้าง

10.4.4 ขั้นตอนการซื้อ Decor

ถ้าเราต้องการที่จะซื้อสินค้าชนิดนั้นเราก็ไป click ที่ปุ่ม buy เพื่อเป็นการสั่งซื้อสินค้าชนิดนั้น โดยที่สินค้าชนิดนั้นจะลงไปอยู่ในตะกร้าก่อน เพื่อที่จะให้เราดูว่าต้องการที่จะเปลี่ยนแปลงหรือไม่

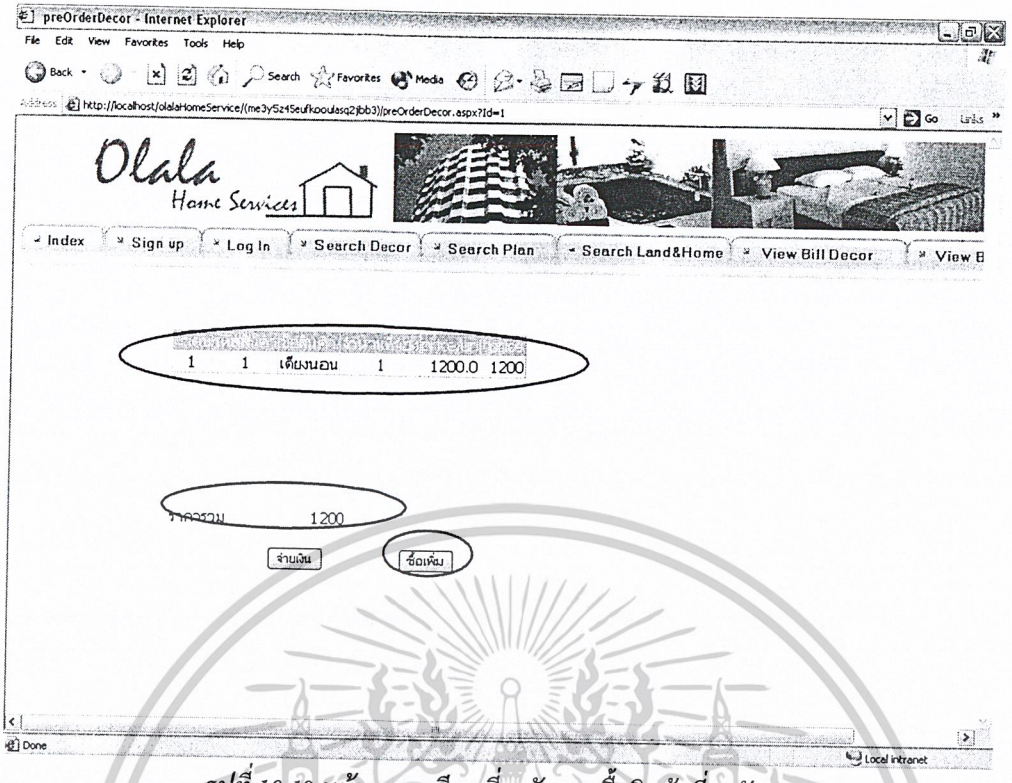
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-18 หน้าการซื้อเตียงนอน

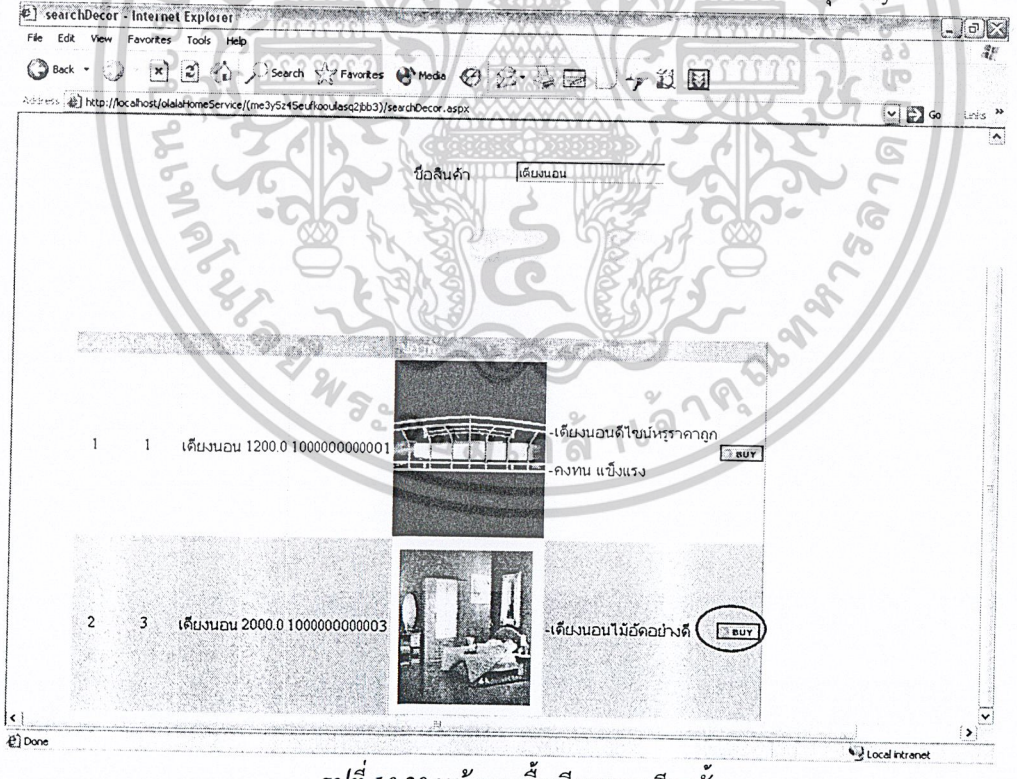
เมื่อ click ปุ่ม buy แล้วระบบจะนำเราไปสู่อีก page โดยที่ใน page นี้จะแสดงสินค้าทั้งหมดที่เราเลือกซื้อไว้ในขณะนั้น ในตัวอย่างนี้เราได้ทำการซื้อเตียงนอนไปแล้ว 1 อัน ราคา 1200 บาท และจะแสดงราคารวมให้ทราบด้วยว่าเท่าไหร่ และเราสามารถซื้อเพิ่มเข้าไปได้อีก โดยที่เรา click ต้องปุ่ม "ซื้อเพิ่ม"

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-19 หน้ารายละเอียดเกี่ยวกับการซื้อสินค้าเกี่ยวกับ Decor

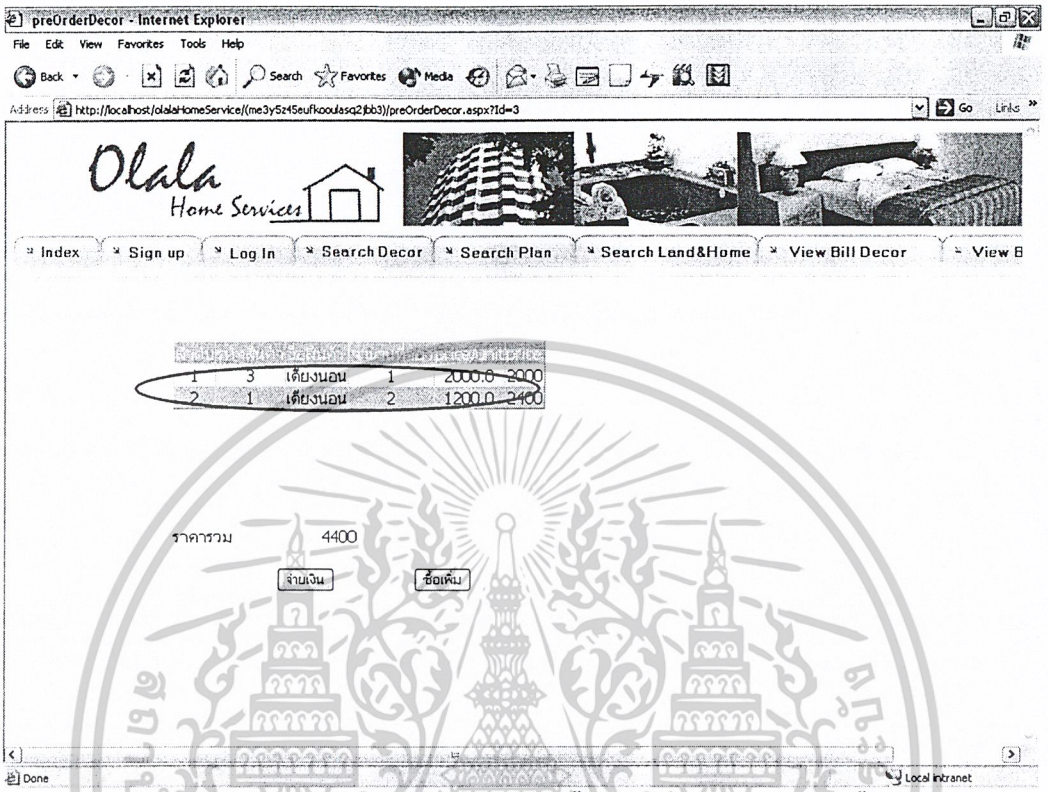
ในที่นี้เราต้องการที่จะซื้อเตียงนอนอีก 1 ชุดเราก็สามารถทำได้โดยการไป click ที่ ปุ่ม buy



รูปที่ 10-20 หน้าการซื้อเตียงนอน อีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

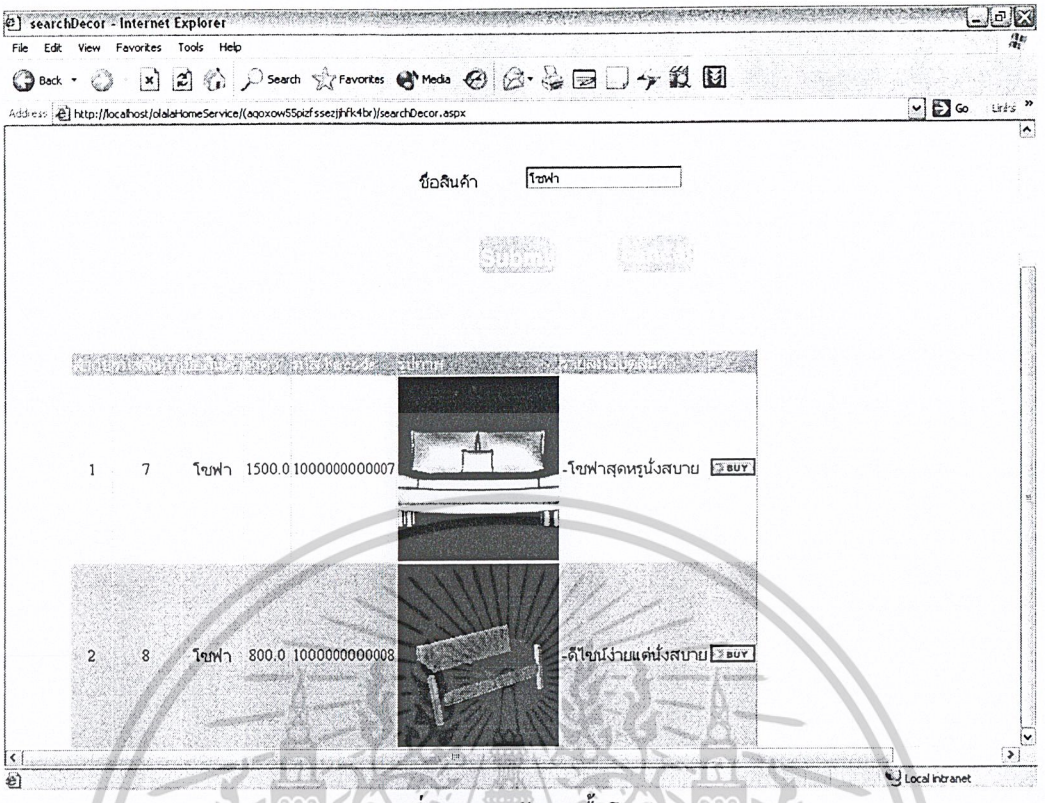
จากนั้นระบบก็จะทำการแสดงรายการสินค้าที่เราเลือกซื้อลงไป ในที่น้จะเห็นได้ว่ามีสินค้าเพิ่มมาอีก 1 อย่างคือเตียงนอน



รูปที่ 10-21 หน้ารายละเอียดเกี่ยวกับการซื้อสินค้าเกี่ยวกับ Decor อีกครั้ง

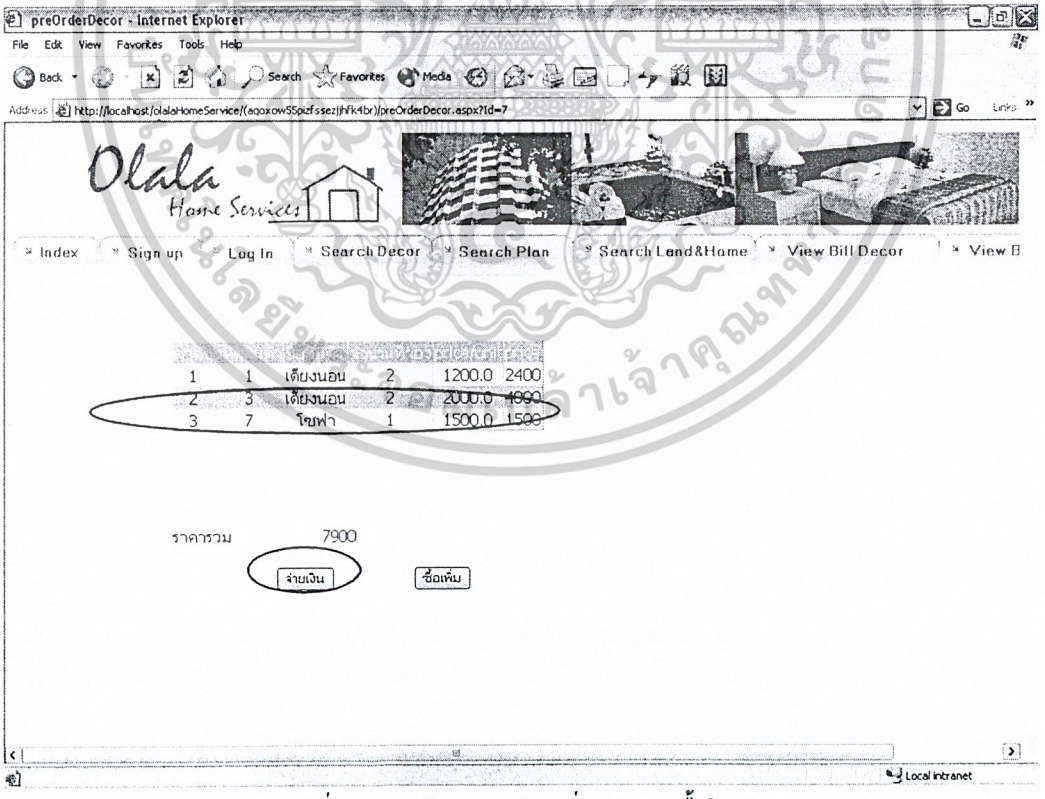
จากนั้นเราก็ทำการซื้อสินค้าอีกชิ้น 1 คือ “โซฟา”

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-22 หน้าการซื้อโซฟา

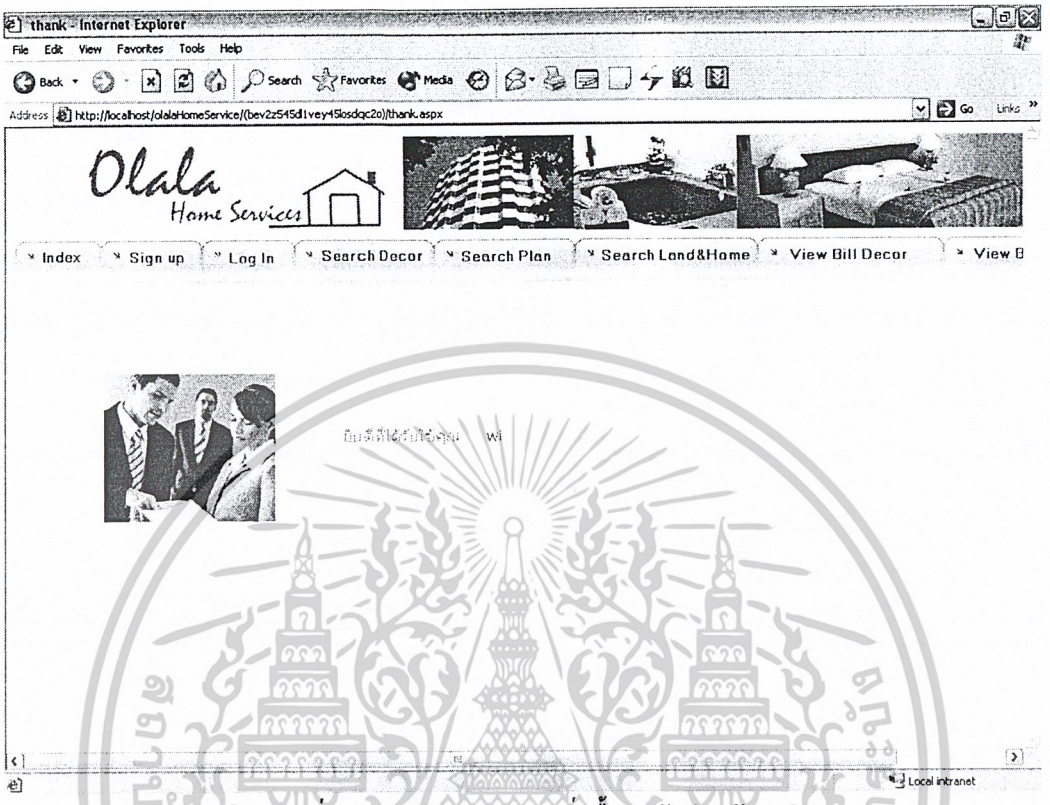
เมื่อทำการเลือกซื้อเสร็จแล้วก็ทำการกดปุ่ม “จ่ายเงิน” เพื่อเป็นการยืนยันการสั่งซื้อ



รูปที่ 10-23 หน้ารายละเอียดเกี่ยวกับการซื้อโซฟา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

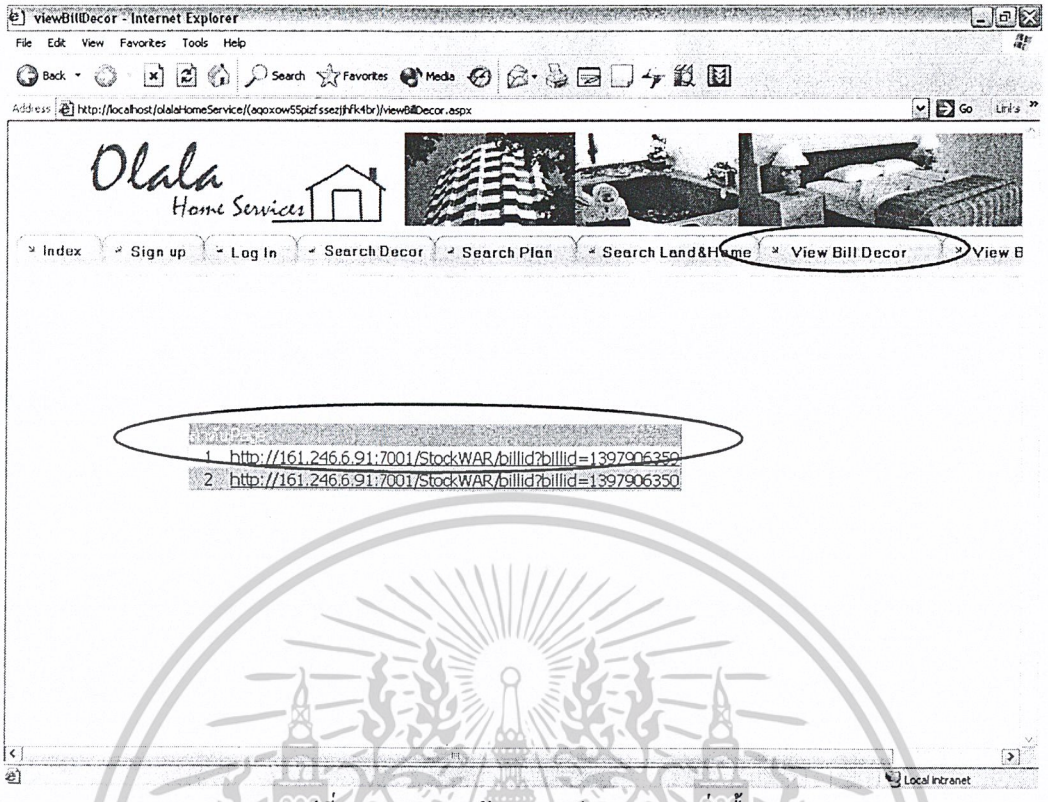
ถ้าทุกอย่างเรียบร้อยจะแสดงระบบจะทำการแสดงความถูกต้องดังรูป



รูปที่ 10-24 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย

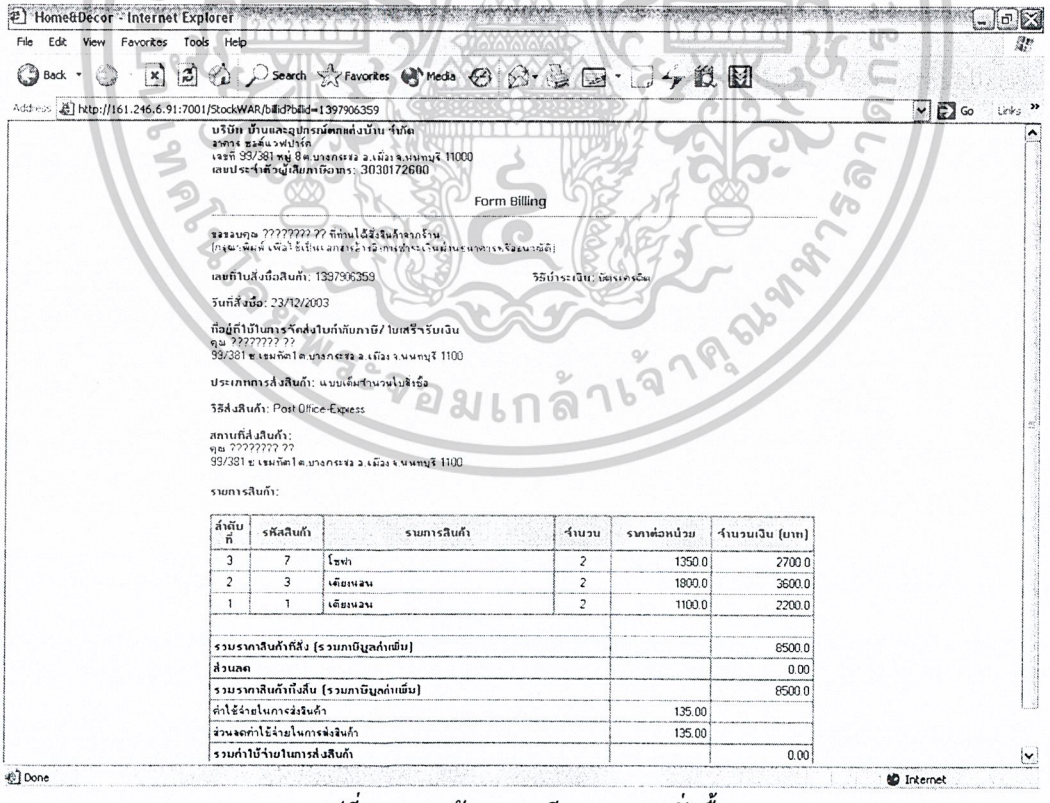
หลังจากที่เราได้ทำการซื้อสินค้าเสร็จแล้วและทำการยืนยันกลับระบบไปแล้ว เราสามารถดู bill ของเราได้โดยที่เข้าไป click ที่ “ViewDecor” เพื่อเป็นการดูรายละเอียดของ bill ทั้งหมดของเรา โดยที่ระบบจะแสดง page ที่ link ไปหาโดยที่แบ่งเป็น 2 อันเพราะว่าเราได้ทำการซื้อ 2 ครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-25 link หน้ารายละเอียดของการสั่งซื้อ

เมื่อทำการ click link ไปแล้วจะแสดง bill ของเราออกมา

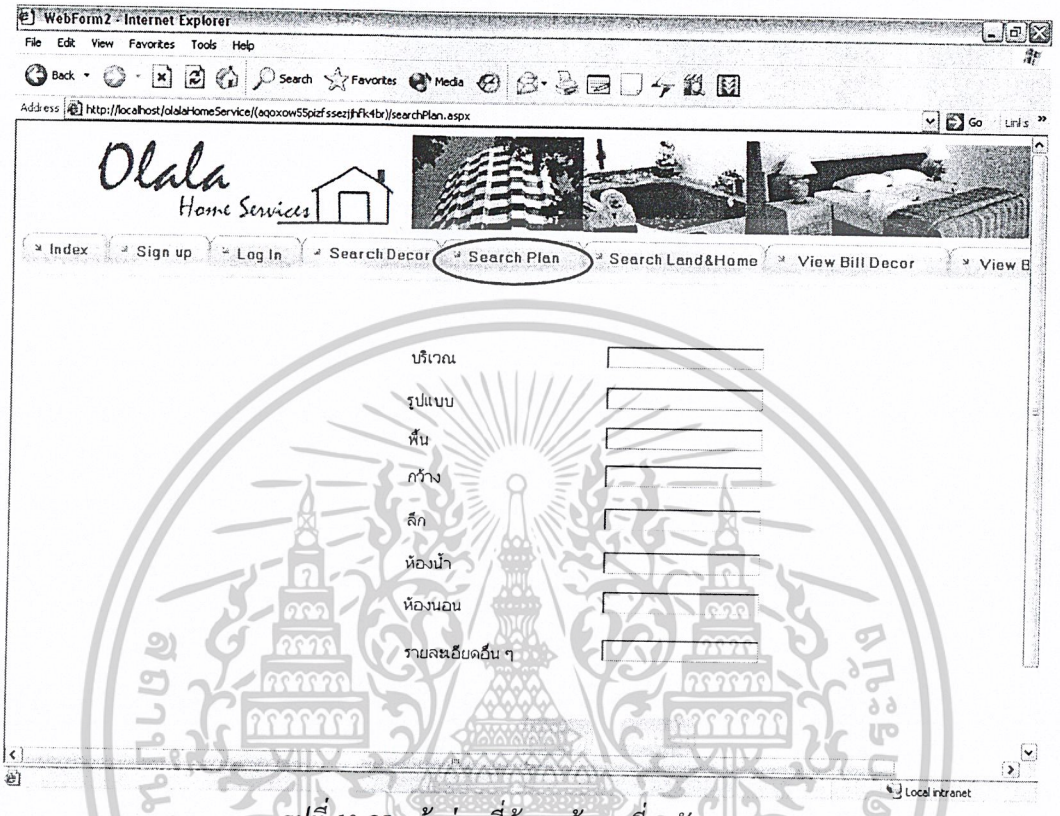


รูปที่ 10-26 หน้ารายละเอียดของการสั่งซื้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.4.5 ขั้นตอนการ Search plan

โดยที่เราไป click ที่ปุ่ม “search plan” เพื่อที่จะเป็นการค้นหารายการแบบบ้าน



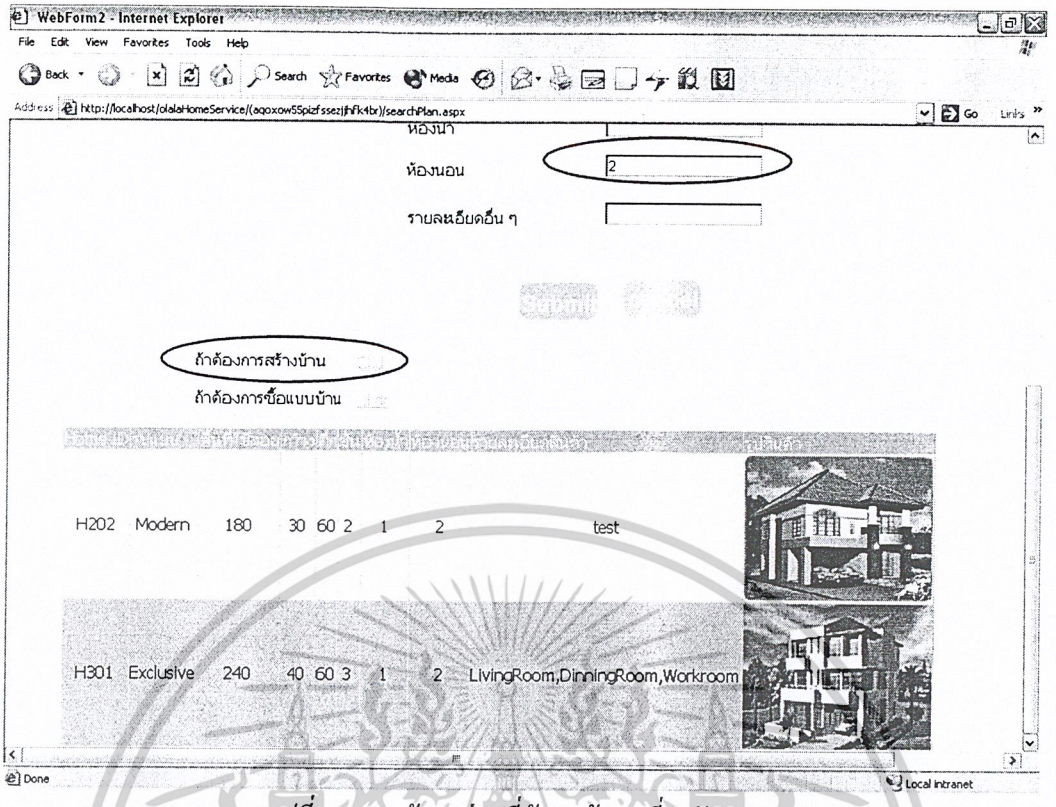
รูปที่ 10-27 หน้าส่วนที่ค้นหาข้อมูลเกี่ยวกับ Plan

ในที่นี้เราทำการ search โดยค้นหาแบบที่มีห้องน้ำ 2 ห้อง ระบบก็จะแสดงออกมาดังรูป

10.4.6 ขั้นตอนการสร้างบ้าน

ถ้าเราต้องการที่จะสร้างบ้านให้เรา click ไปที่ “ถ้าต้องการสร้างบ้าน click”

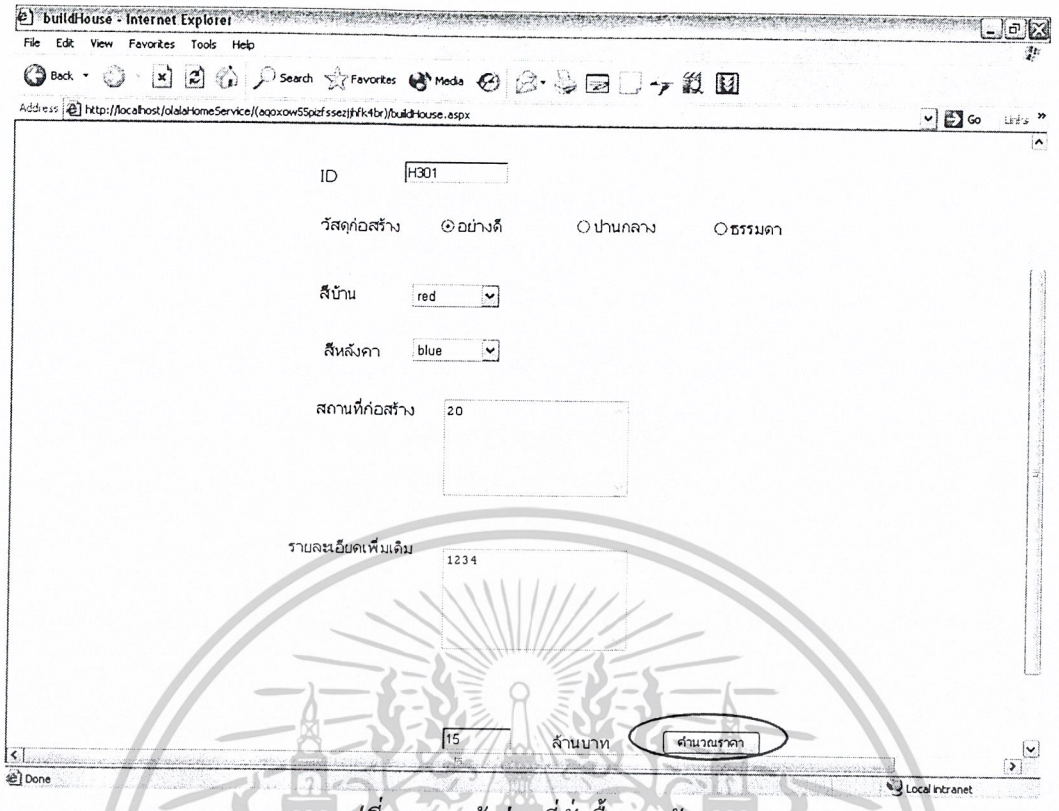
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-28 หน้าผลส่วนที่ค้นหาข้อมูลเกี่ยวกับ Plan

จากนั้นเราก็กรอกรายละเอียดเกี่ยวกับบ้านที่เราต้องการที่จะสร้าง ให้ครบทุกส่วนเพราะจะได้เป็นประโยชน์กับ user เองที่จะได้รับเกี่ยวกับการสร้างบ้าน จากนั้น click ปุ่ม "คำนวนราคา" เพื่อเป็นการคิดราคาจากสิ่งที่เราเลือกไว้ก่อน

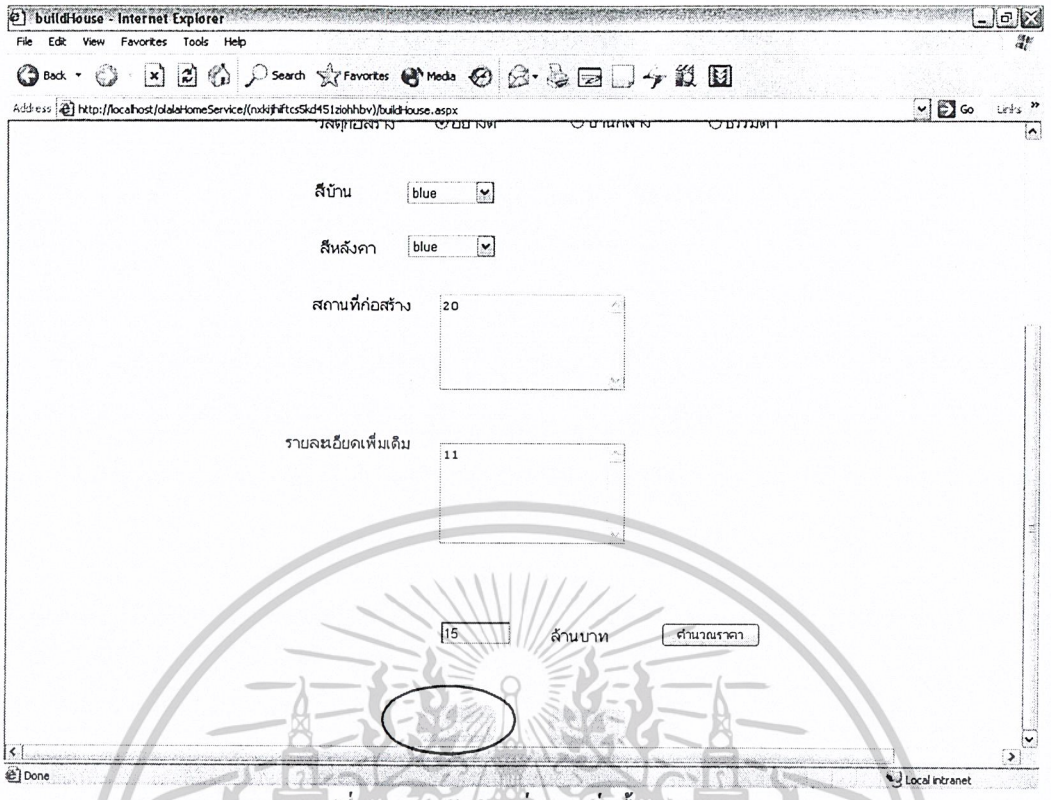
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-29 หน้าส่วนที่สั่งซื้อแบบบ้าน

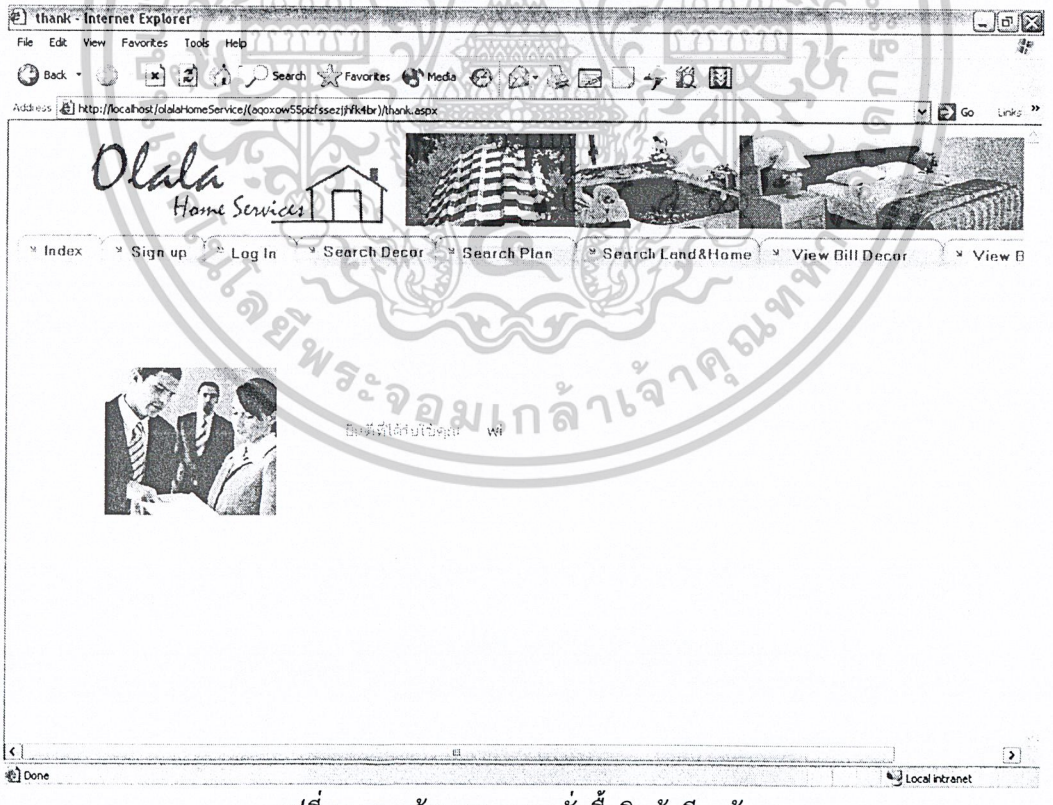
เมื่อต้องการที่จะยืนยันในการซื้อแล้วเราก็ทำการกดปุ่ม “submit” เพื่อเป็นการยืนยันในยืนยันในการซื้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-30 หน้าส่วนที่ตกลงสั่งซื้อแบบบ้าน

ระบบจะนำไปสู่หน้า "thank.aspx" เพื่อแสดงถึงความถูกต้องในการซื้อสินค้า

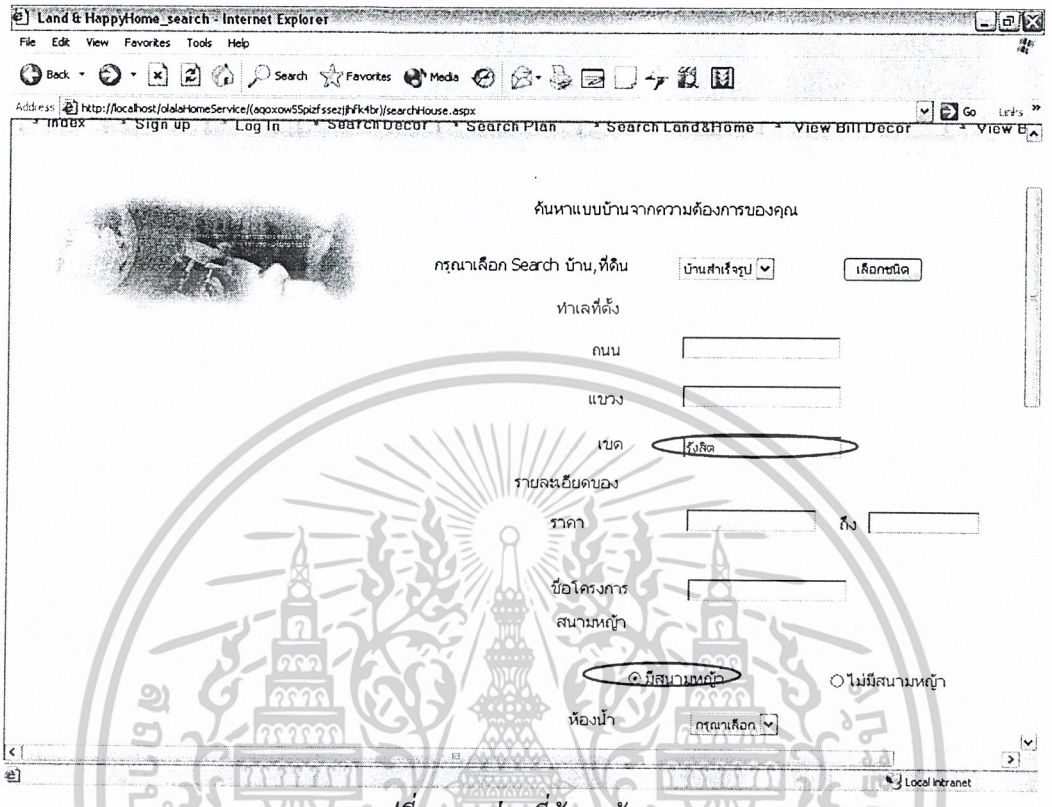


รูปที่ 10-31 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10.4.7 ขั้นตอนการ Search บ้านสำเร็จรูปและที่ดิน

หลังจากนั้นเราก็จะทำการ search หาบ้านที่เราต้องการ โดยที่เราสามารถใส่ค่าทุกชนิดทุกอย่างที่ต้องการ search ลงไปใน page

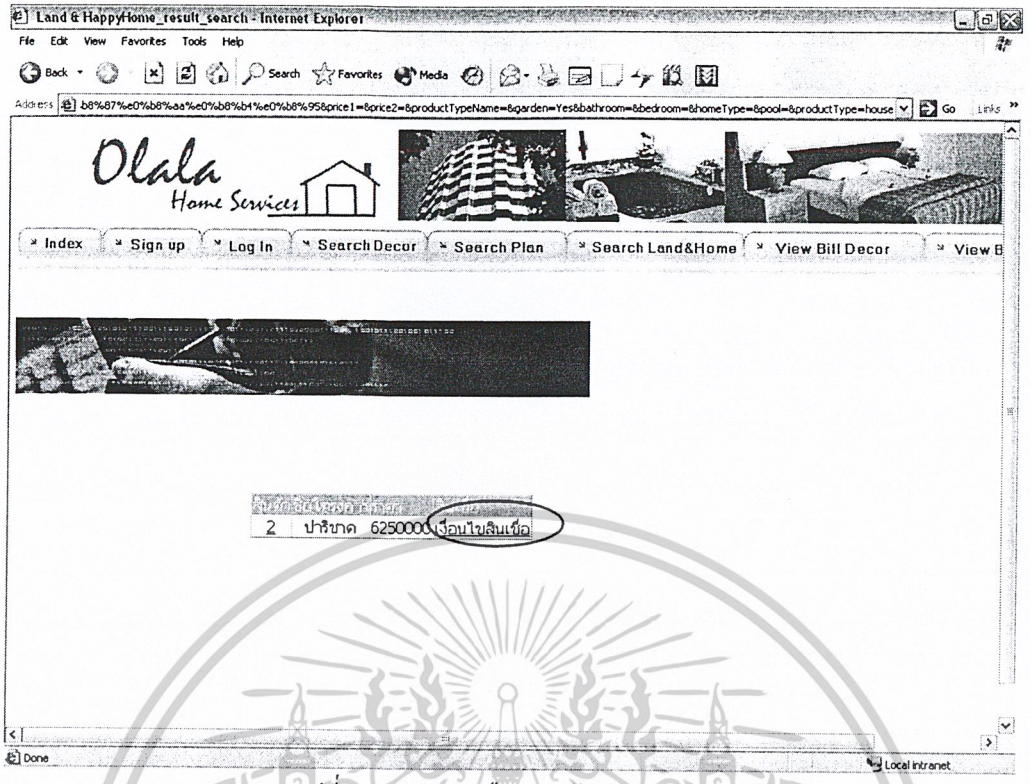


รูปที่ 10-32 ส่วนที่ค้นหาข้อมูล

10.4.8 ขั้นตอนการซื้อบ้านสำเร็จรูปและที่ดิน

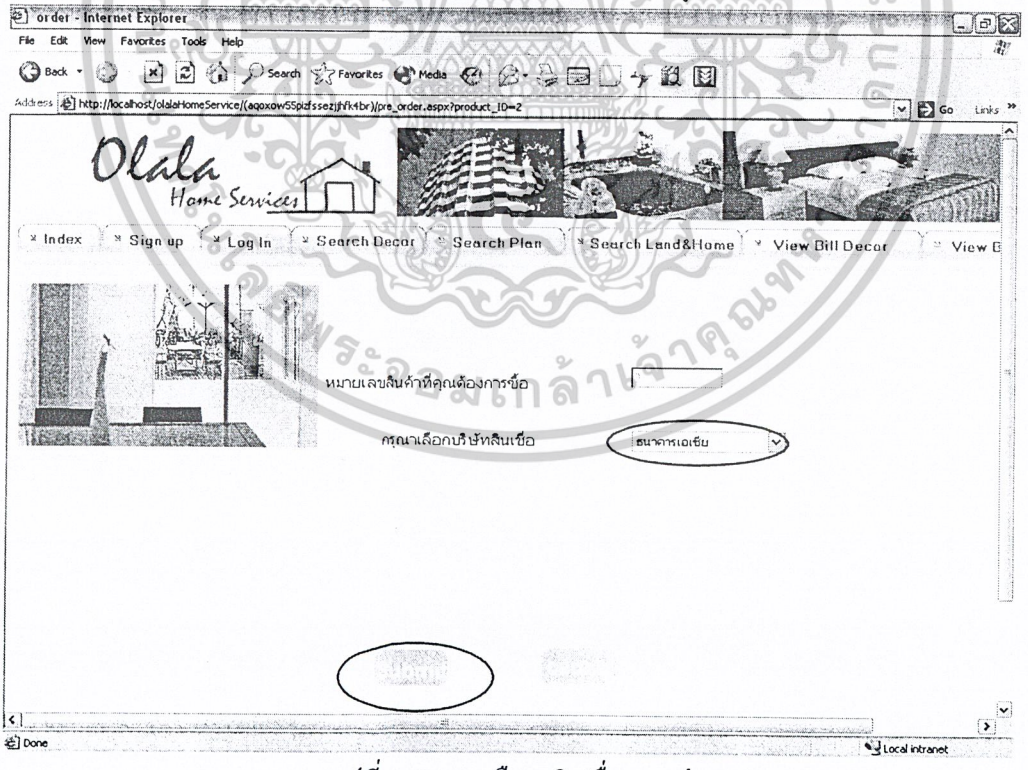
จากในที่นี้เราได้ทำการ search บ้านสำเร็จรูปที่อยู่ใน เขตรังสิต และมีสนามหญ้าจะทำให้เราได้ผลการ search ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-33 ผลการเลือกการค้นหาบ้านแบบมีสนามหญ้า

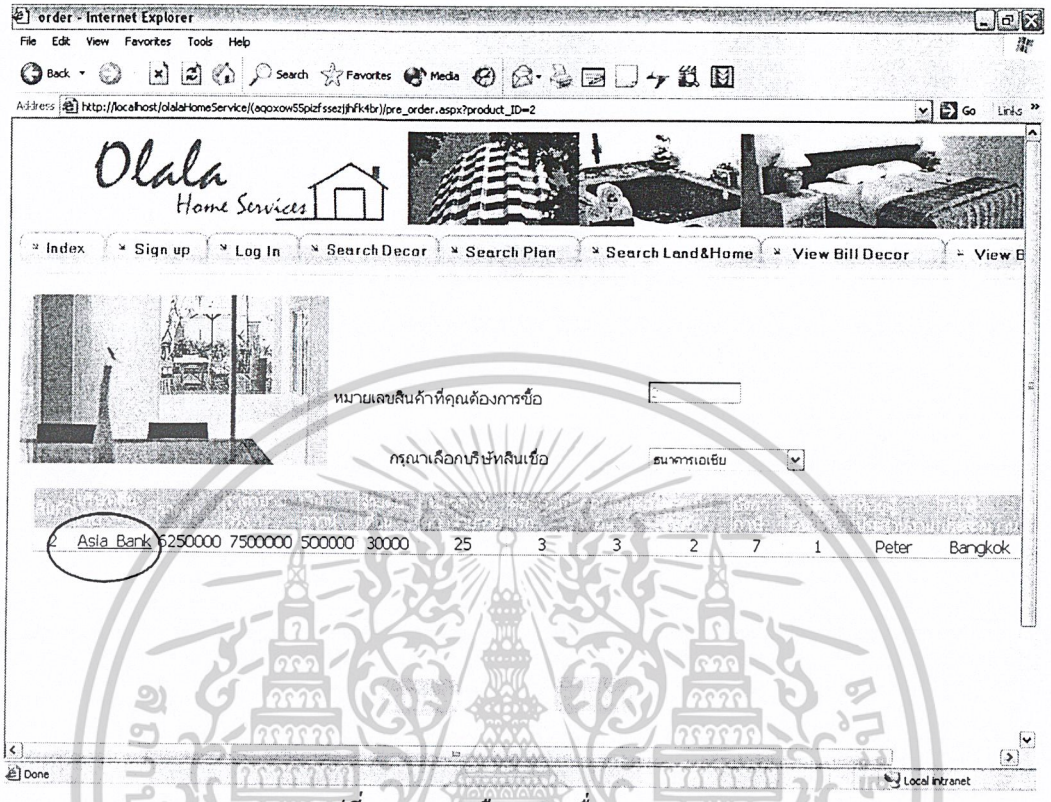
เราสามารถดูรายละเอียดของบ้านได้โดยที่ไปช่องสินค้า ในที่นี้จะเป็นสินค้าชนิดที่ 2 และเราสามารถดูเงื่อนไขสินเชื่อได้โดยการที่กดเลือกที่ช่องสินเชื่อ จะได้ดังรูป



รูปที่ 10-34 การเลือกดูสินเชื่อของแต่ละธนาคาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

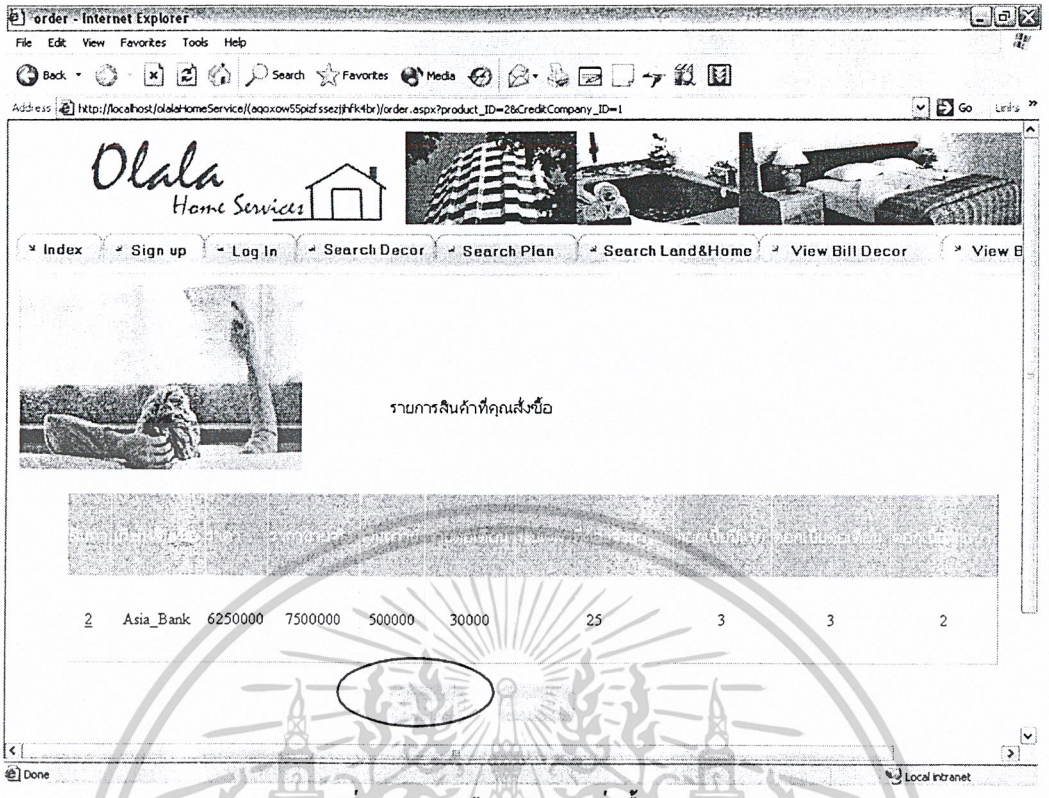
จากนั้นเราก็กดเลือกธนาคารที่เราต้องการทราบว่าอัตราการผ่อนเป็นอย่างไง ในที่นี้เรากดเลือกธนาคารไทยพานิชย์จากนั้นเราก็กดเลือกปุ่ม Submit เพื่อที่จะได้ดูรายละเอียด



รูปที่ 10-35 การเลือกดูสินเชื่อของธนาคาร

กดเลือกที่ช่องบริษัทสินเชื่อเพื่อเป็นการยืนยันการผ่อนชำระว่าเราต้องการที่จะผ่อนกับบริษัทนี้ จากนั้นระบบก็จะทำการแสดงรายละเอียดออกมาให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

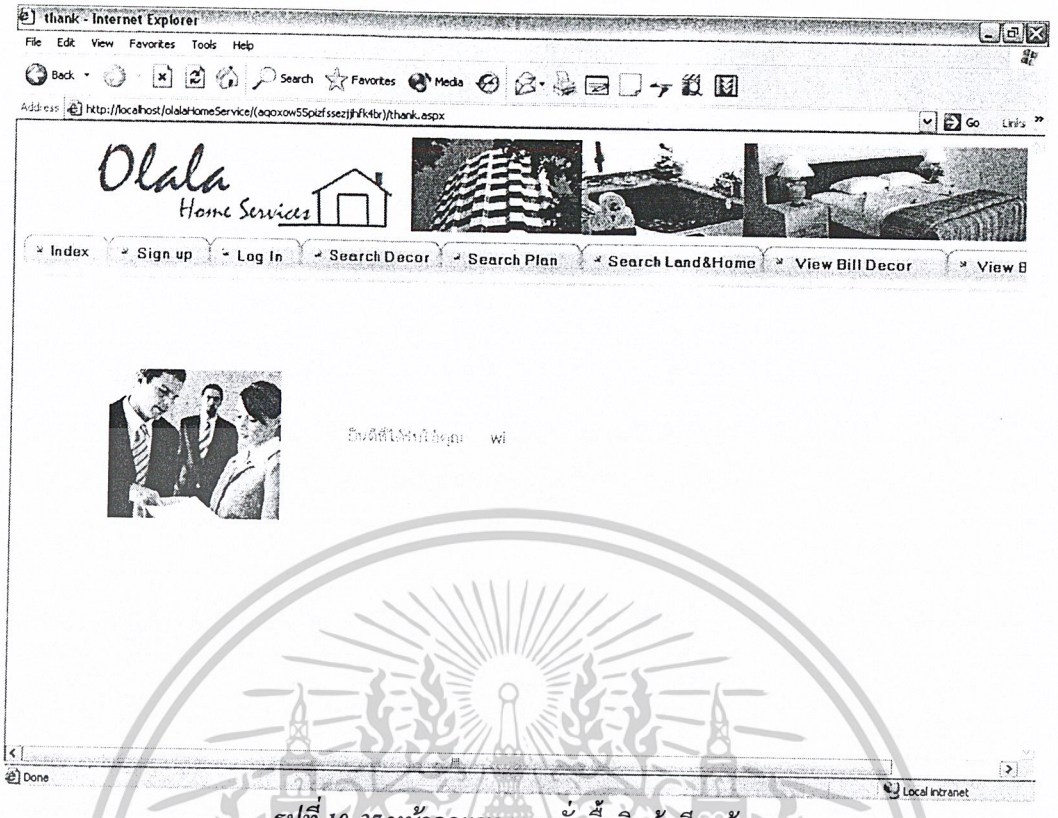


รูปที่ 10-36 การยืนยันในการสั่งซื้อสินค้ากับธนาคาร

10.4.9 ขั้นตอนการยืนยันในการสั่งซื้อ

กด submit เพื่อเป็นการยืนยันว่าเราต้องการซื้อ จากนั้นระบบจะนำเราเข้าสู่หน้า thank.aspx เพื่อเป็นการขอบคุณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

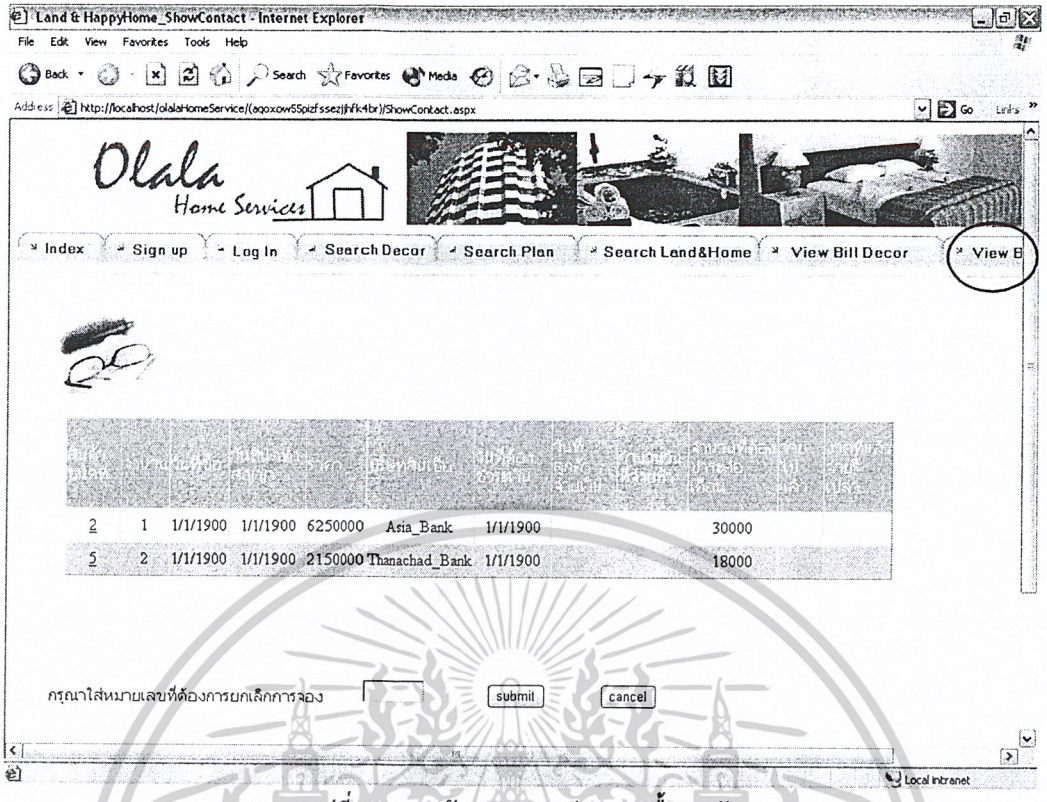


รูปที่ 10-37 หน้าจอแสดงการสั่งซื้อสินค้าเรียบร้อย

10.4.10 ขั้นตอนการดูรายละเอียดของสินค้าบ้านสำเร็จรูปและที่ดิน

เราสามารถดูว่าเราได้ซื้อสินค้าอะไรไปแล้วบ้างโดยที่กดที่ viewBill land and home

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-38 หน้าจอรายละเอียดการซื้อสินค้า

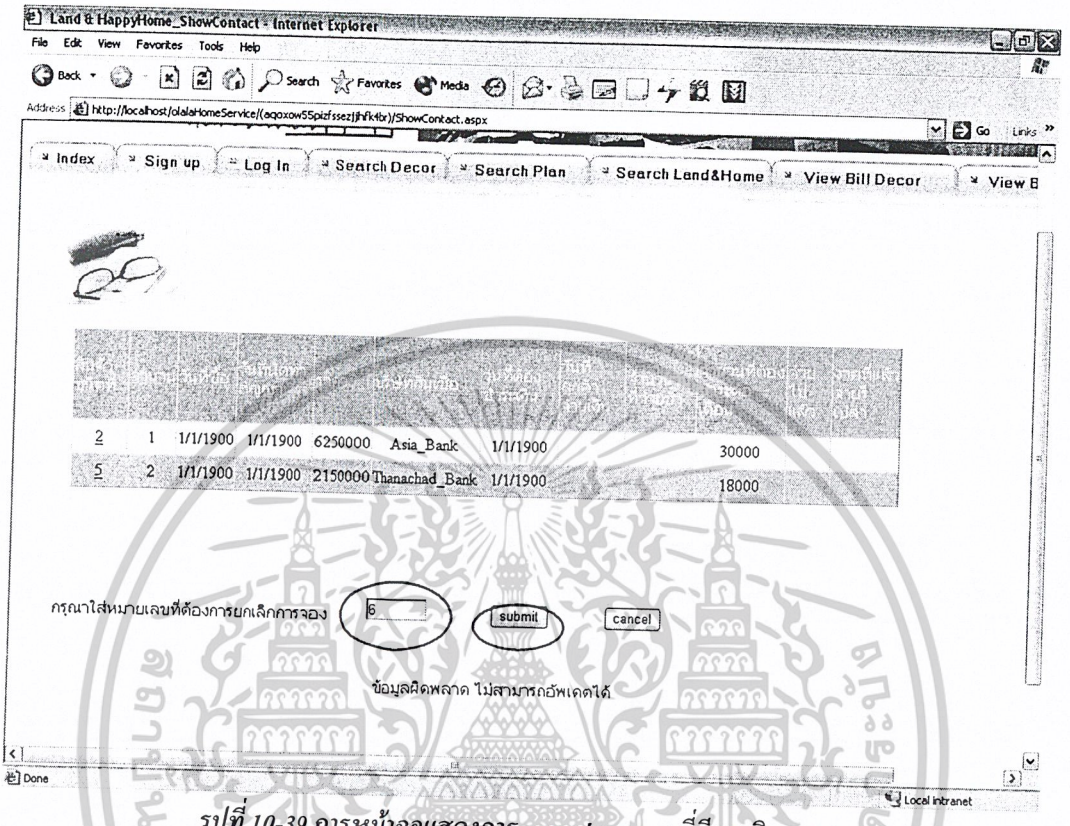
จะทำให้เราได้เห็นว่าเราได้ซื้อสินค้าอะไรไปแล้วบ้าง ในที่นี้เราได้ซื้อสินค้าไปแล้ว 1 ชิ้นคือสินค้าชนิดที่ 2 จำนวน 1 ชิ้น กับสินค้าชนิดที่ 5 จำนวน 2 ชิ้น

10.4.11 ขั้นตอนการ cancel การสั่งซื้อ

และถ้าต้องการที่จะ cancel contact เราก็สามารถทำได้โดยเราใส่สินค้าชนิดที่ลงไปในช่องแล้ว click ปุ่ม submit เพื่อเป็นการยืนยันว่าเราต้องการที่จะ cancel contact

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

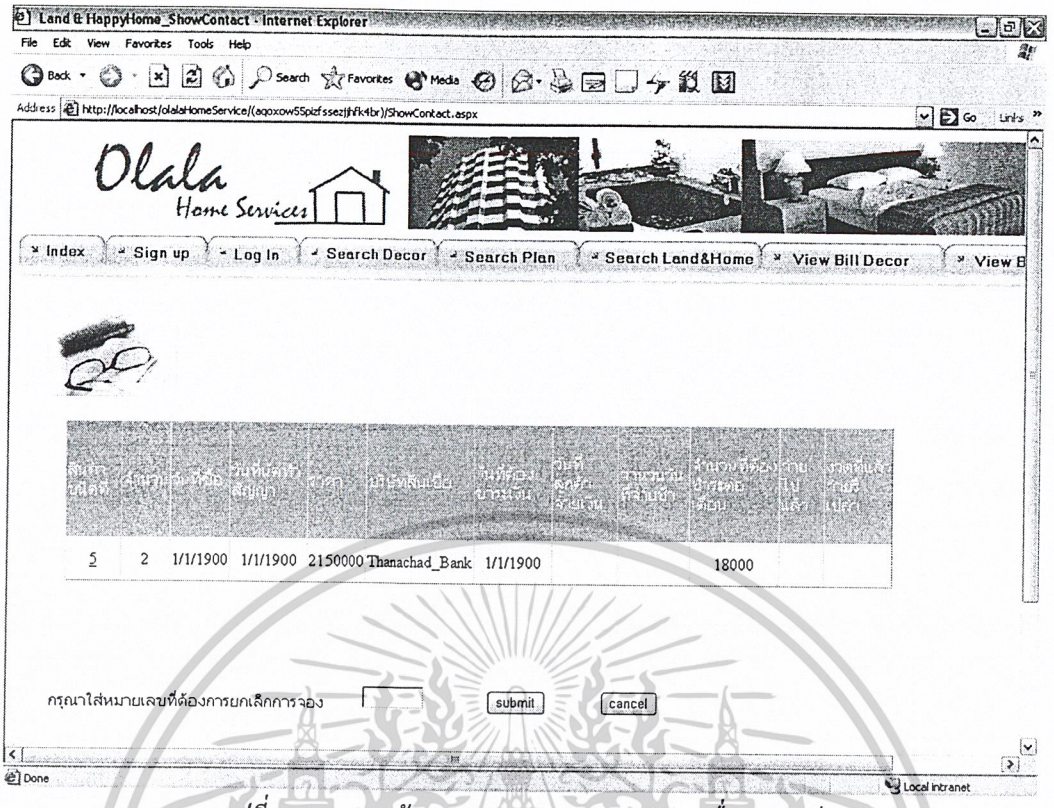
แต่ถ้าเรา cancel ผิด โดยที่ในรูปเรามีแค่สินค้าชนิดที่ 6 แต่เราจะไป cancel สินค้าชนิดที่ 5 ทำให้ไม่สามารถทำได้ ระบบจะทำการฟ้อง error ออกมาดังรูป ทำให้เราไม่สามารถ cancel ได้



รูปที่ 10-39 การหน้าจอแสดงการ cancel contact ที่มีการผิดพลาด

เมื่อ cancel แล้วจะได้ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 10-40 การหน้าจอแสดงการ cancel contact เมื่อสมบรูณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 11
บทวิจารณ์และสรุป

11.1 สรุปผลการดำเนินงาน

ในโครงงานนี้ได้แบ่งการดำเนินงานออกเป็น 4 ส่วนหลัก คือ

1. ศึกษาทฤษฎีพื้นฐานของ คอมโพเน็นต์ , .NET เฟรมเวิร์ก , ASP.NETเว็บเซอร์วิส และเทคโนโลยีอื่นๆ ที่เกี่ยวข้อง

2. ออกแบบระบบงาน เพื่อทำการทดสอบทฤษฎีต่างๆ ที่ได้ศึกษามา ซึ่งระบบงานที่สร้างขึ้นมาคือ ระบบโฮมฟอร์ยู(Home For You),ระบบโฮมแอนด์เดคอร์(Home&Decor) และระบบแลนแอนด์โฮม (Land and Home) โดยส่วนที่รับผิดชอบคือ ระบบแลนแอนด์โฮม(Land and Home) ซึ่งเป็นระบบซื้อขายบ้านสำเร็จรูปและที่ดินเปล่า

3. ทำการอิมพลีเมนต์ระบบงานที่ได้ออกแบบไว้ โดยใช้ภาษา VB.NET

4.ทำการอิมพลีเมนตระบบตรงกลางคือระบบโอลาลาโฮมเซอร์วิส (Olala Home Services) ซึ่งเป็นระบบตรงกลางไว้คอยเชื่อมต่อกับกับระบบอีก 3 ระบบที่เหลือ โดยที่เราทำการเรียกโฮมเซอร์วิสจากระบบทั้ง 3

11.2 แนวทางการพัฒนาต่อ

การออกแบบและพัฒนาระบบเว็บเซอร์วิสนั้น มีแนวโน้มที่จะมีการใช้งานอย่างกว้างขวางในอนาคตเนื่องจากมีความสามารถในด้านต่างๆ รวมถึงกระแสของระบบอีคอมเมอร์ซ ที่มีเพิ่มมากขึ้น ซึ่งสามารถนำเอาเว็บเซอร์วิสไปประยุกต์ใช้ได้ ระบบเว็บเวอร์วิสจึงเป็นทางเลือกที่น่าศึกษาและพัฒนาต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

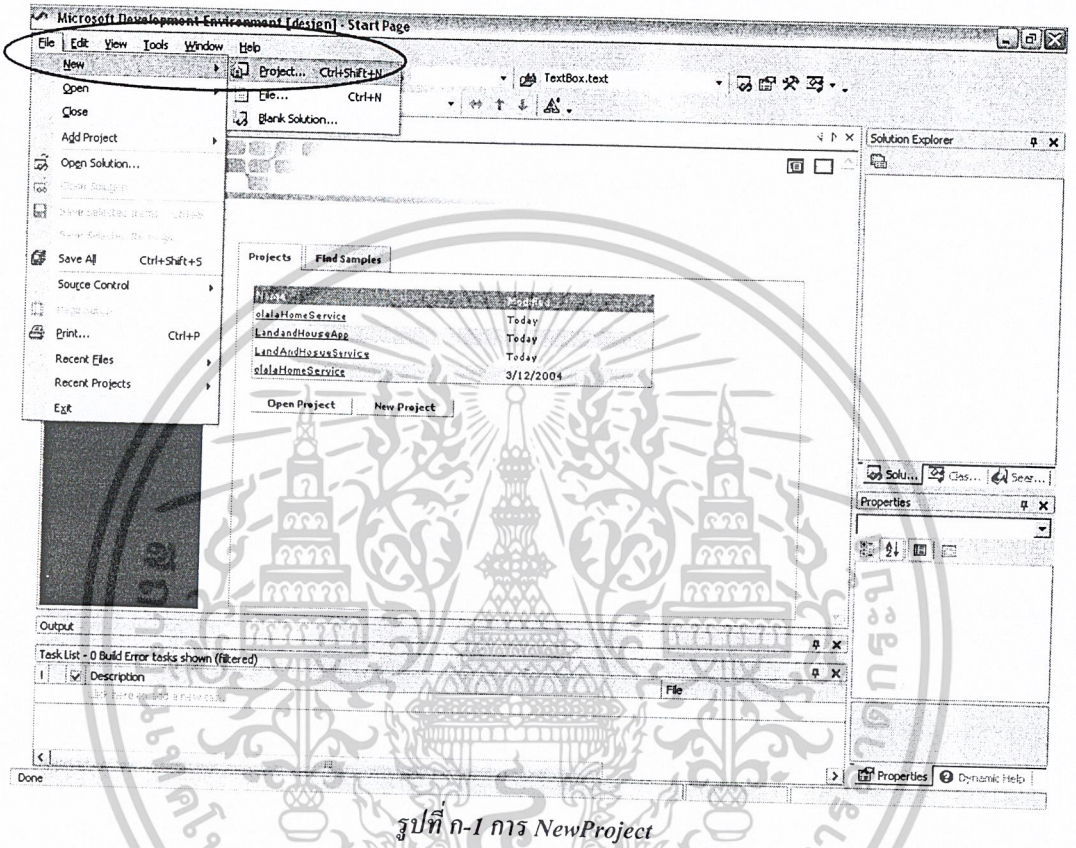
ภาคผนวก ก

คู่มือการสร้าง ASP.NET

1. ขั้นตอนการสร้าง ASP.NET

1.1 เลือกที่ New

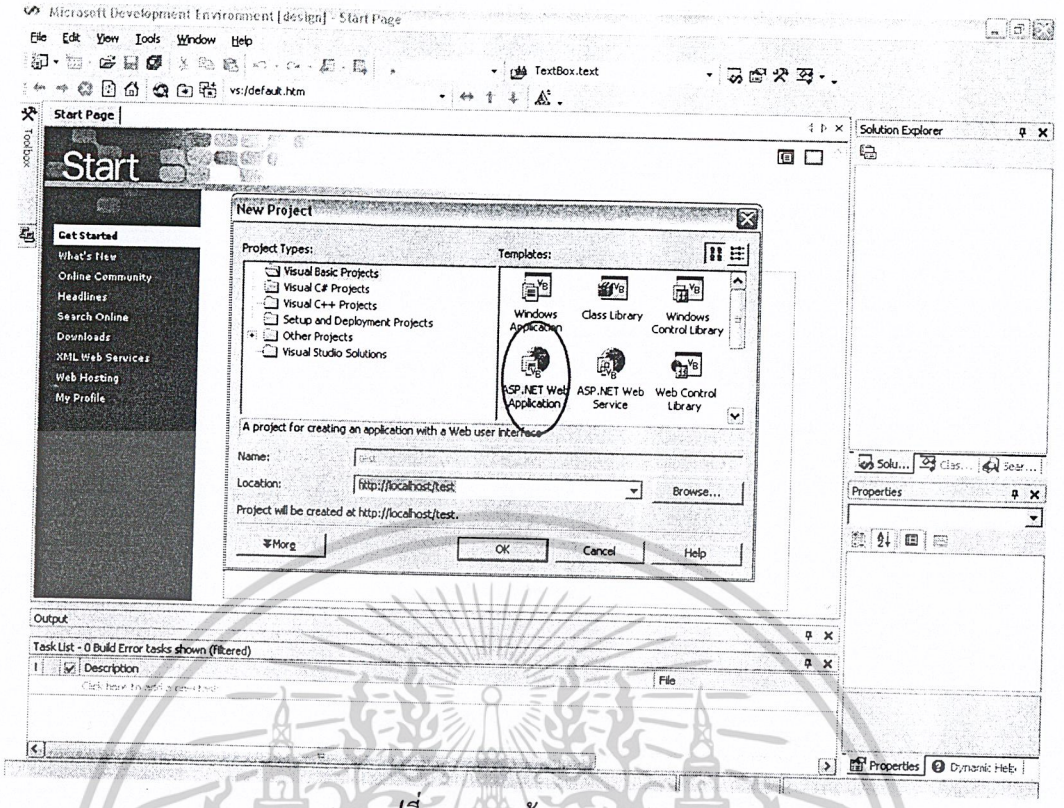
1.2 เลือกที่ Project



รูปที่ ก-1 การ NewProject

1.3 จากนั้นเลือกที่ ASP.NET

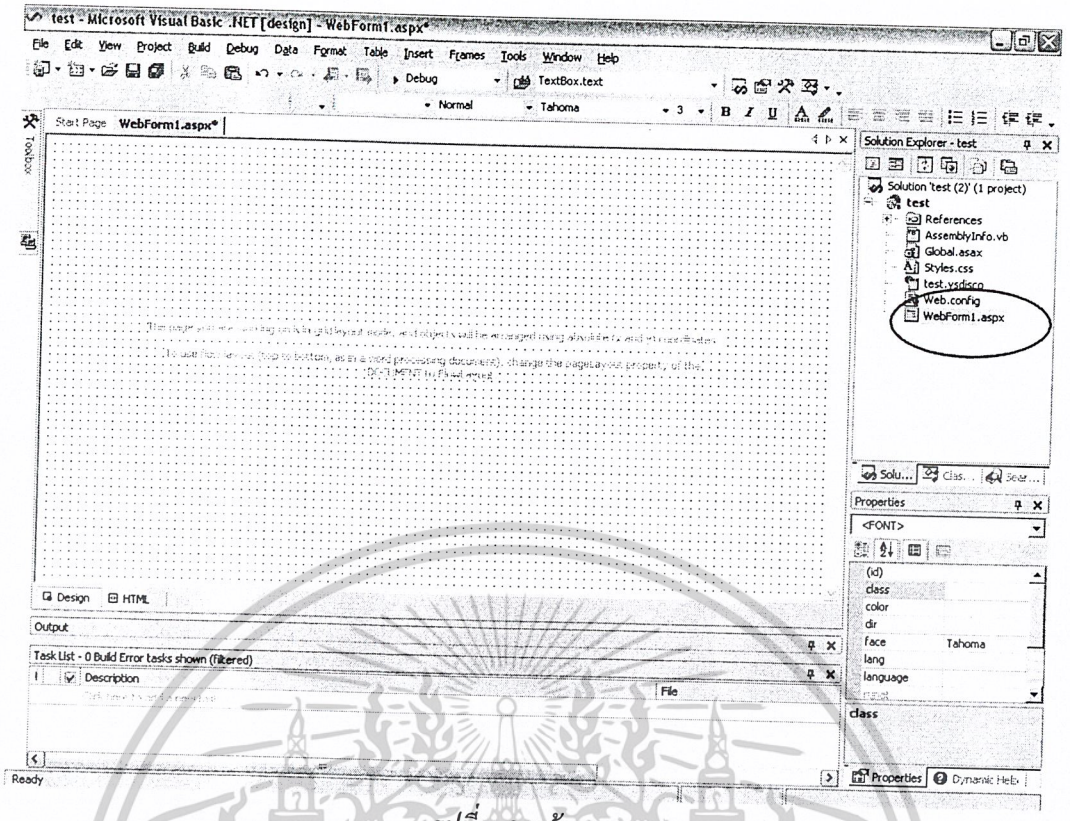
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-2 การสร้าง ASP.NET

1.4 จากนั้นเราจะเราจะได้ file มา 1 อันคือ webform1.aspx ซึ่งเป็น file งานของเราซึ่งเราจะได้พื้นที่ในการออกแบบ

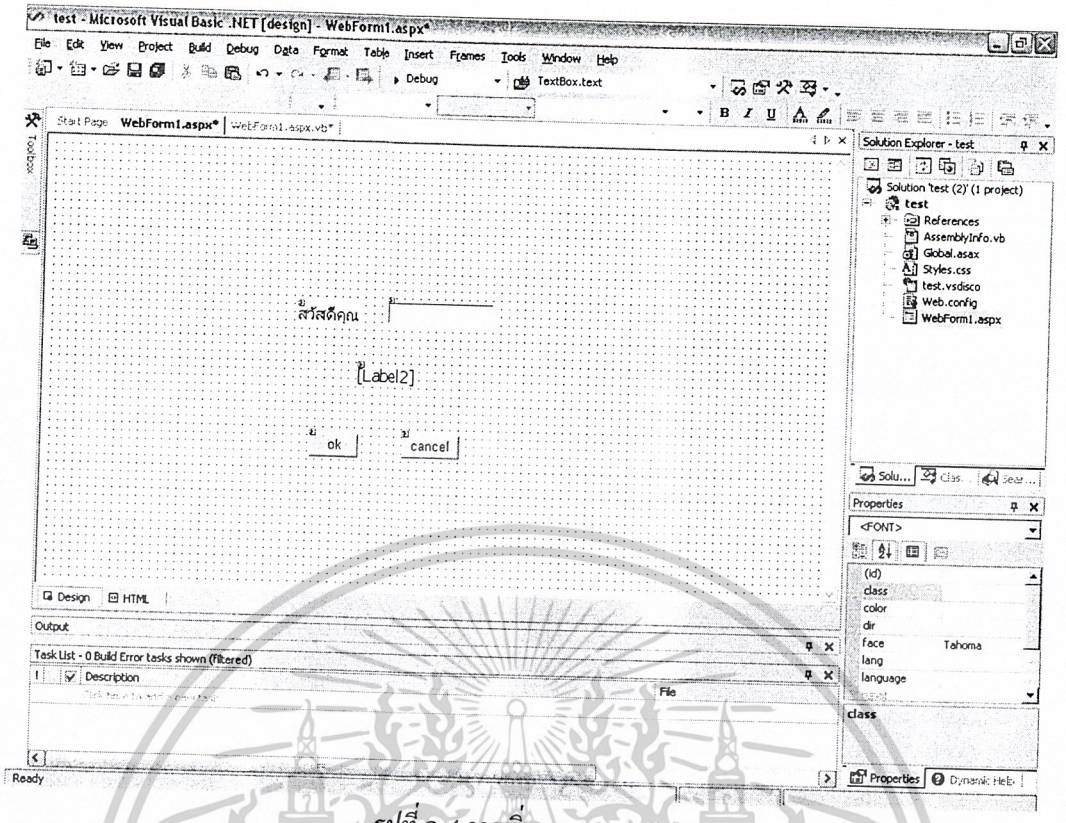
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-3 หน้าการออกแบบ

1.5 จากนั้นเราก็เริ่มทำการออกแบบหน้า page ของเราเป็นอย่างตามต้องการ

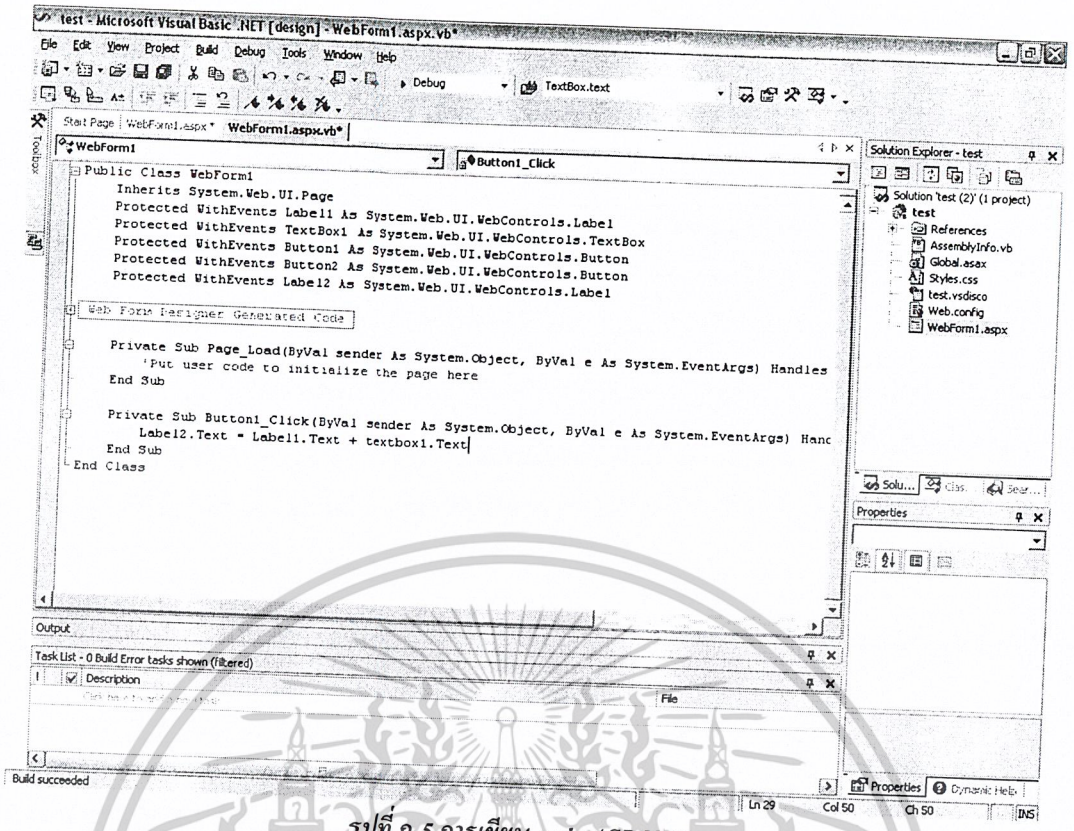
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-4 การเริ่มการออกแบบ

1.6 เริ่มเขียน code เราจะทำได้โดยการ double click ลงไปในที่ที่เราต้องการกำหนดเหตุการณ์ให้กับคอนโทลนั้น ๆ

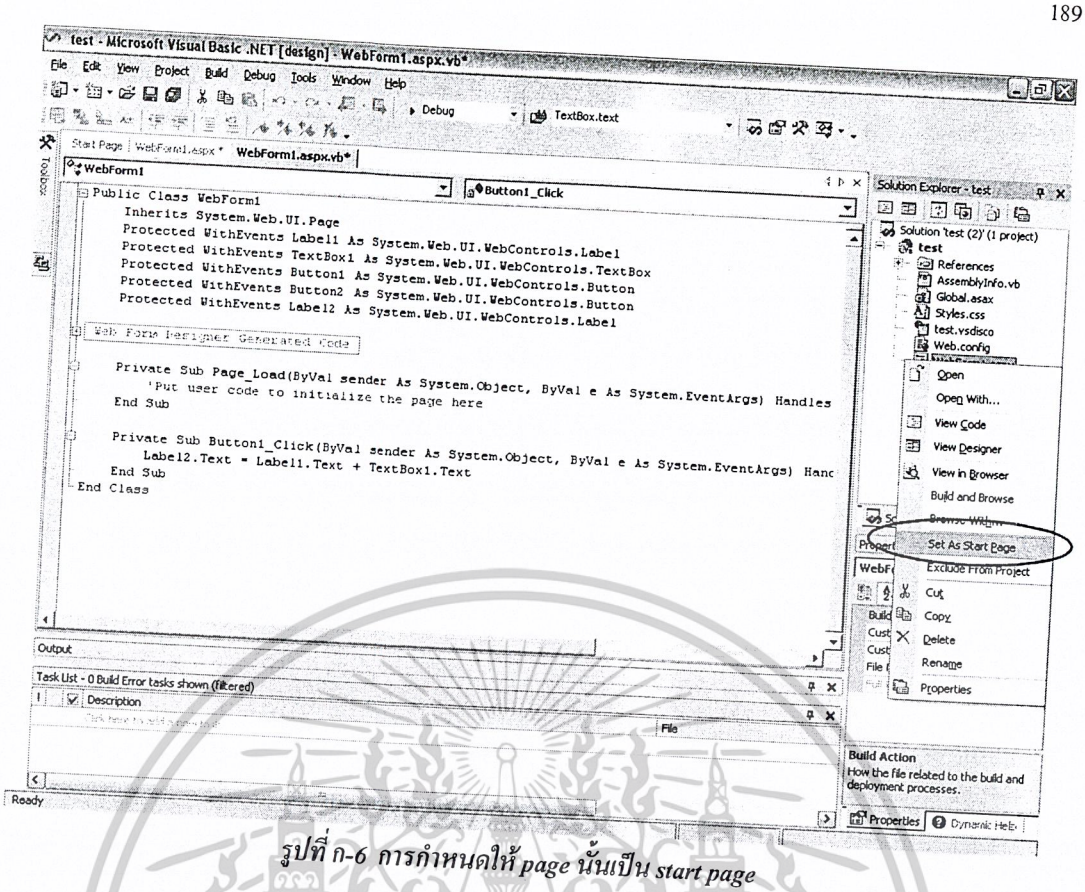
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-5 การเขียน code ASP.NET

1.7 set ให้ page นั้นเป็น start page โดยทำการ click ขวาที่ทางด้านขาวแล้วเลือก “set as start page” จะทำให้ page นั้นของเราเป็น page ที่ถูก load มาเป็น page แรก

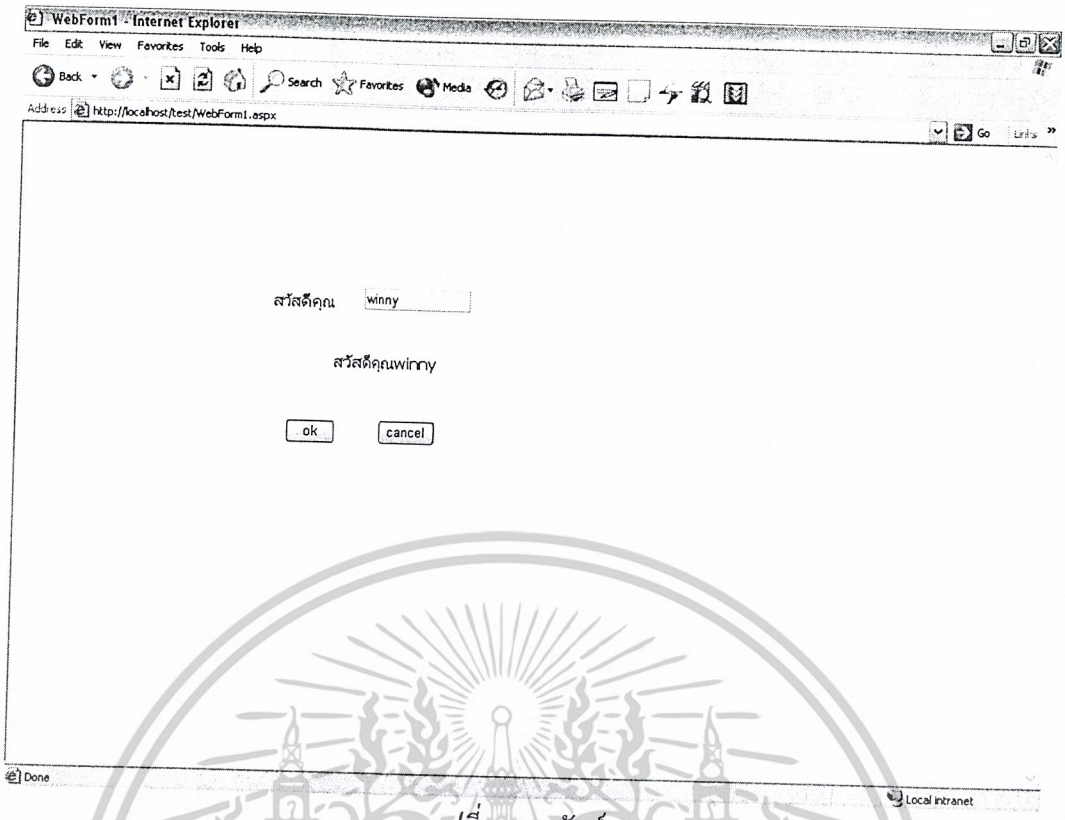
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-6 การกำหนดให้ page นั้นเป็น start page

1.8 ถ้าเราต้องการให้มันแสดงผลเราก็กดปุ่ม F5 เพื่อเป็นการสร้าง project จากนั้นเราก็ทำตามเหตุการณ์ต้องการที่เราต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-7 ผลลัพธ์

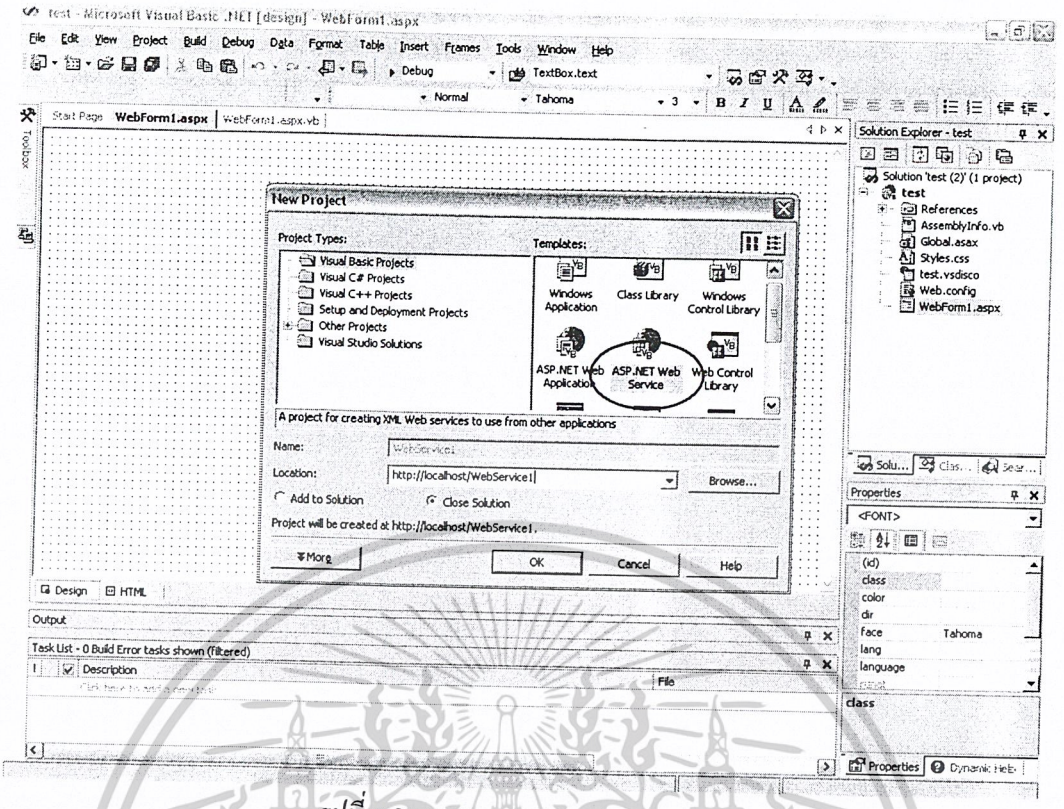
2. ขั้นตอนการสร้าง Web services

2.1 เลือกที่ New

2.2 เลือกที่ Project

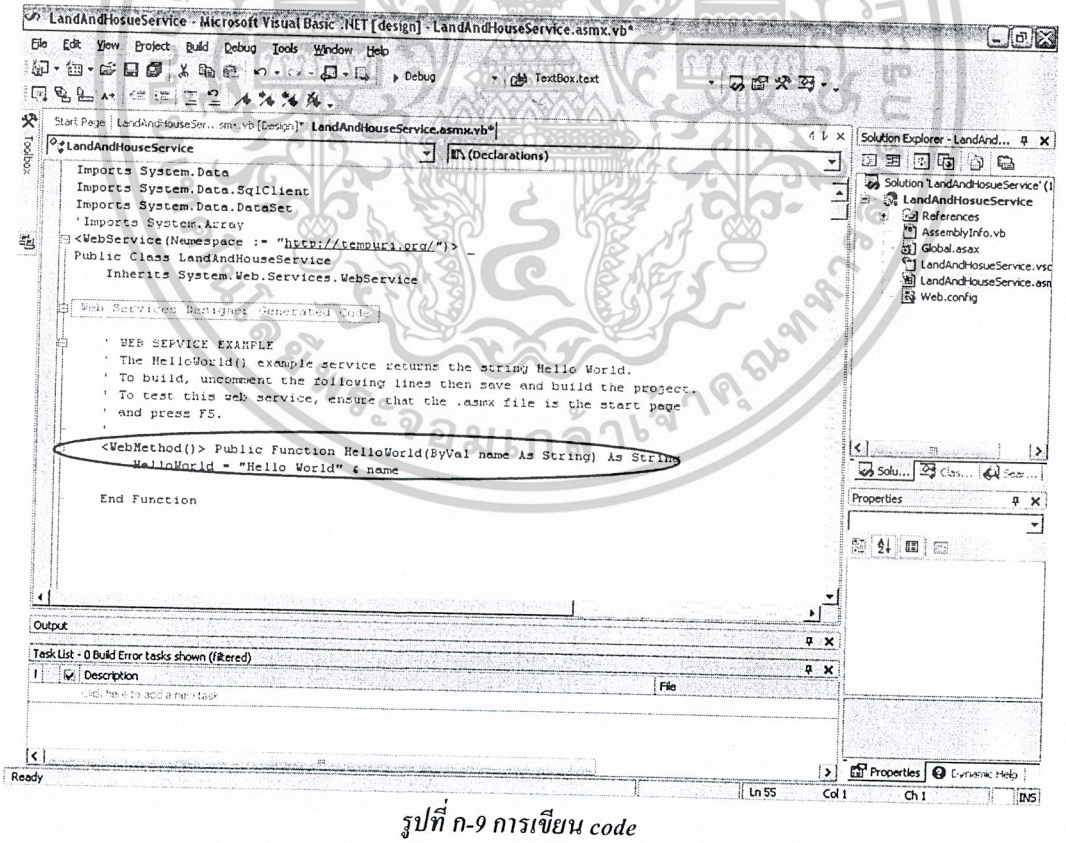
2.3 เลือกที่ ASP.NET Web Services

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-8 การ New Project Web Services

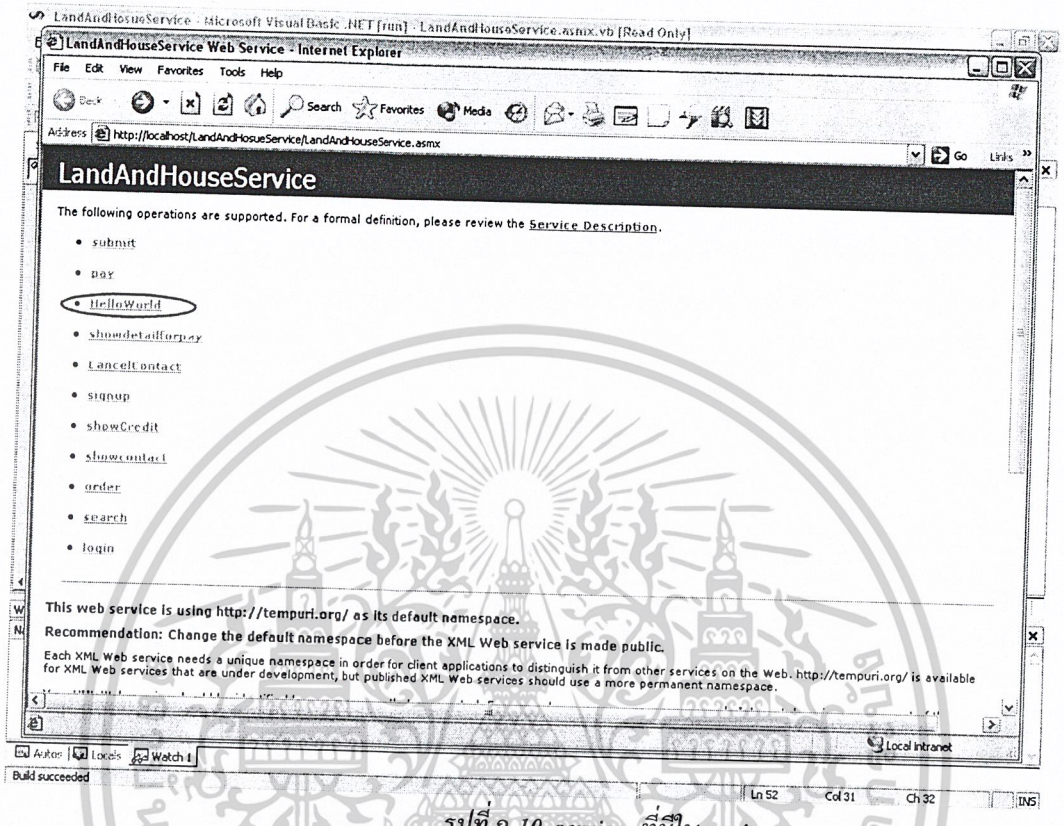
2.4 จากนั้นเราก็เริ่มเขียน code



รูปที่ ก-9 การเขียน code

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

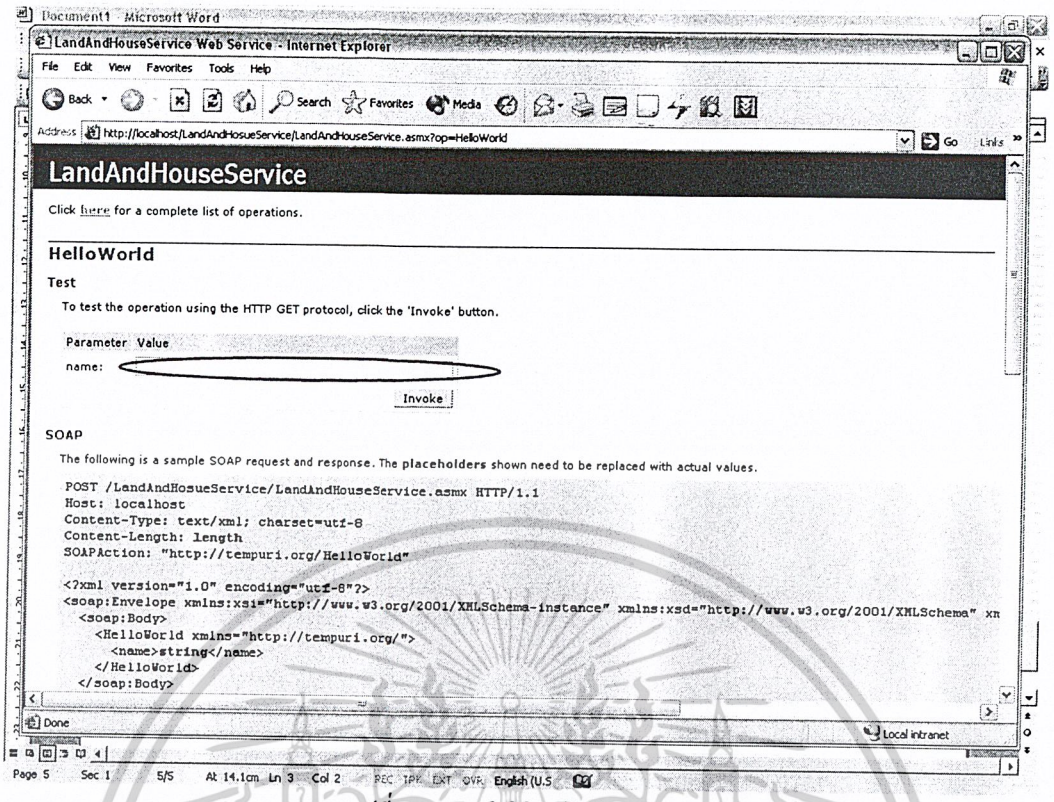
2.5 เมื่อเขียนเสร็จแล้ว เราต้องการให้มันแสดงผลเราก็กดปุ่ม F5 จะได้ผลดังรูป จากนั้นเลือกที่ method ที่เราสร้างขึ้นในที่นี้คือ HelloWorld



รูปที่ ก-10 services ที่มีใน project

2.6 เราทำการใส่ค่าตัวแปรที่ต้องการลงไป

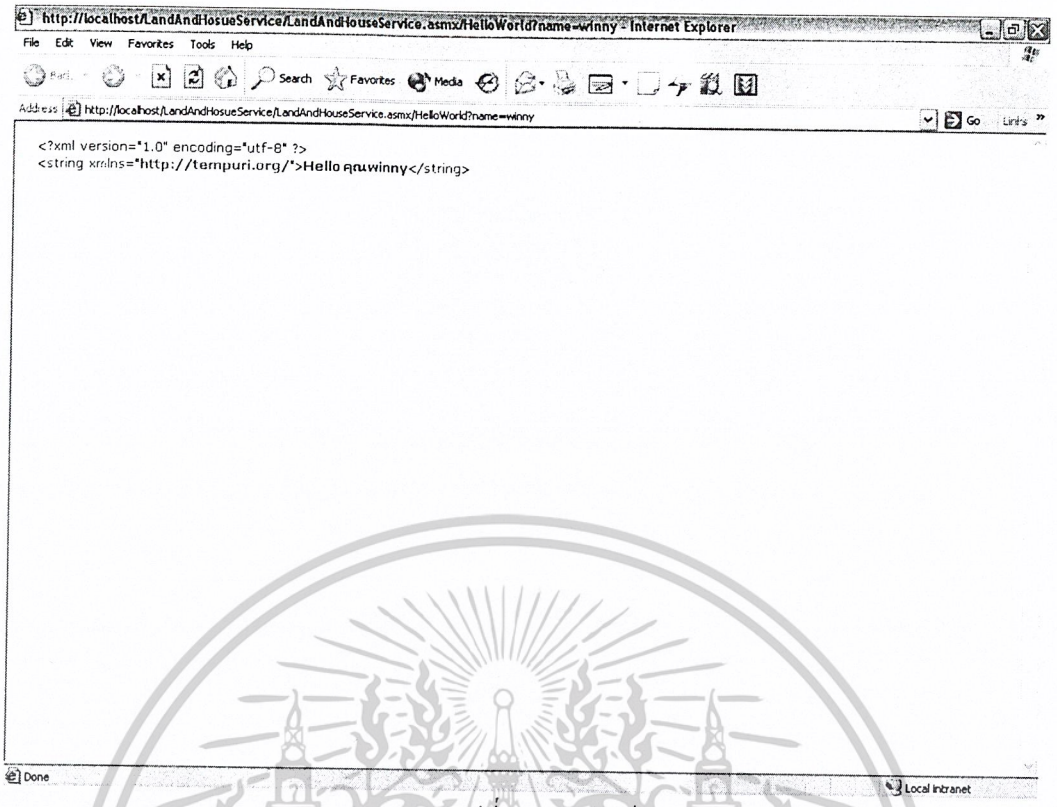
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-11 การใส่ค่าลงไปใน sevices

2.7 จะได้ค่าออกมาดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

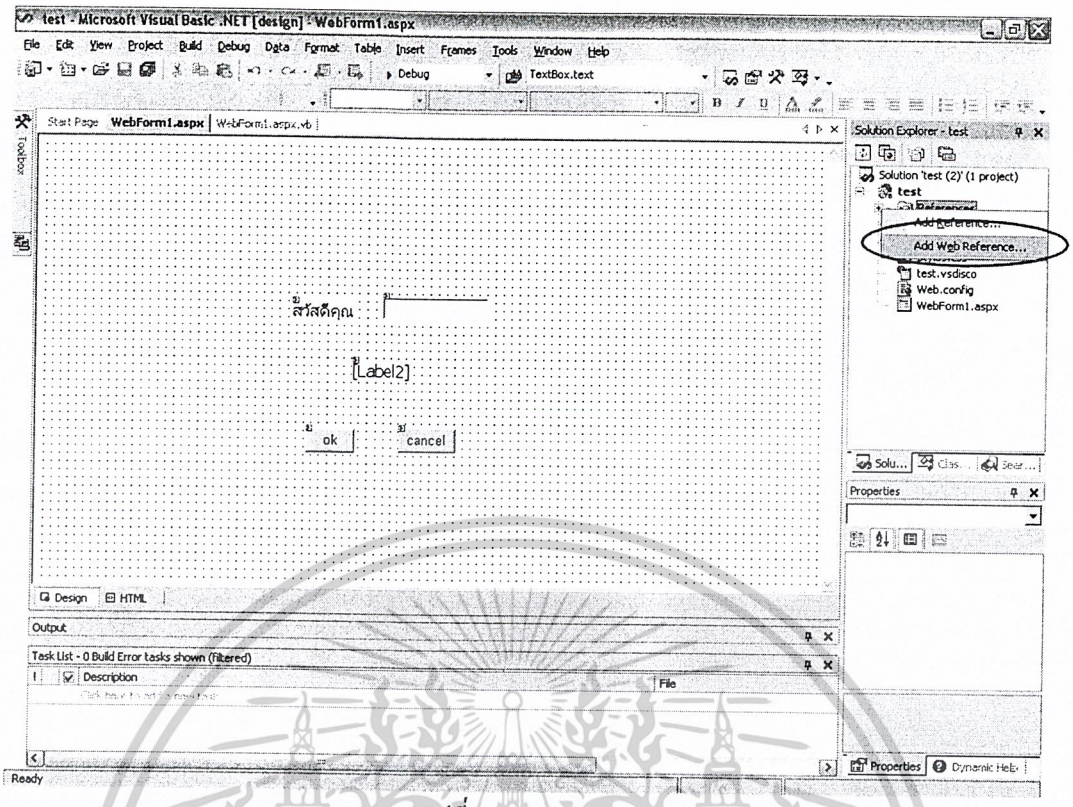


รูปที่ ก-12 ผลลัพธ์ของ Services

3. ขั้นตอนการสร้าง ASP.NET เพื่อเรียกใช้ Web services

3.1 ทำการ add web reference โดยทำการ click ขวาที่ทางขวา จากนั้นก็กดเลือก Add Web Reference

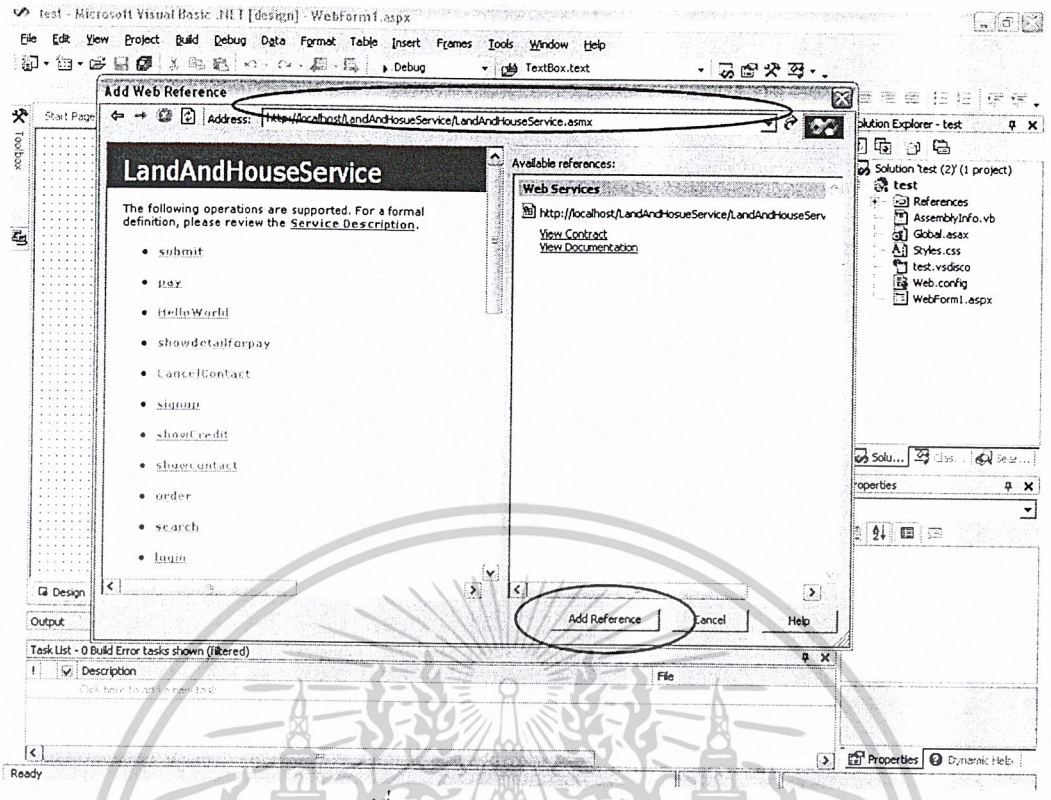
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-13 การออกแบบ Page

3.2 .ใส่ค่าของตำแหน่งที่อยู่ของ services นั้นลงไปในช่อง จากนั้นก็กด Add reference

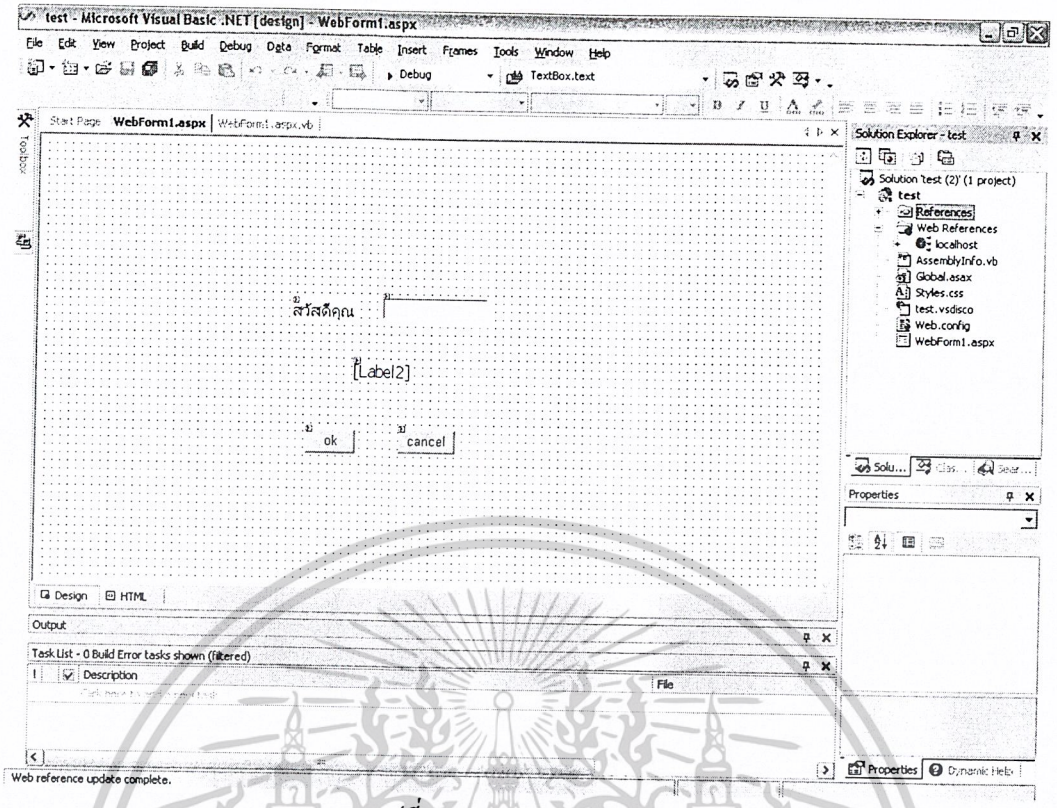
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-14 การ Add Web Reference

3.3 ออกแบบหน้าตาของ page เรา

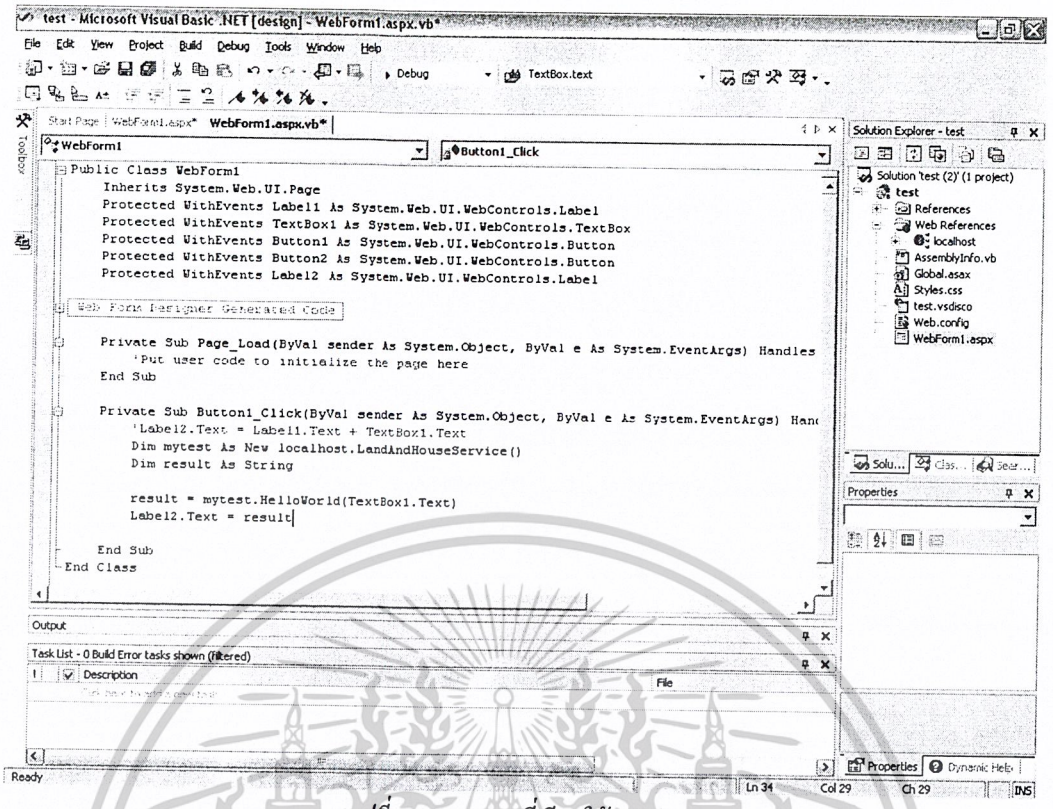
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-15 การออกแบบ Page

3.4เขียน code เพื่อกำหนดเหตุการณ์ให้กับ page นั้น ๆ

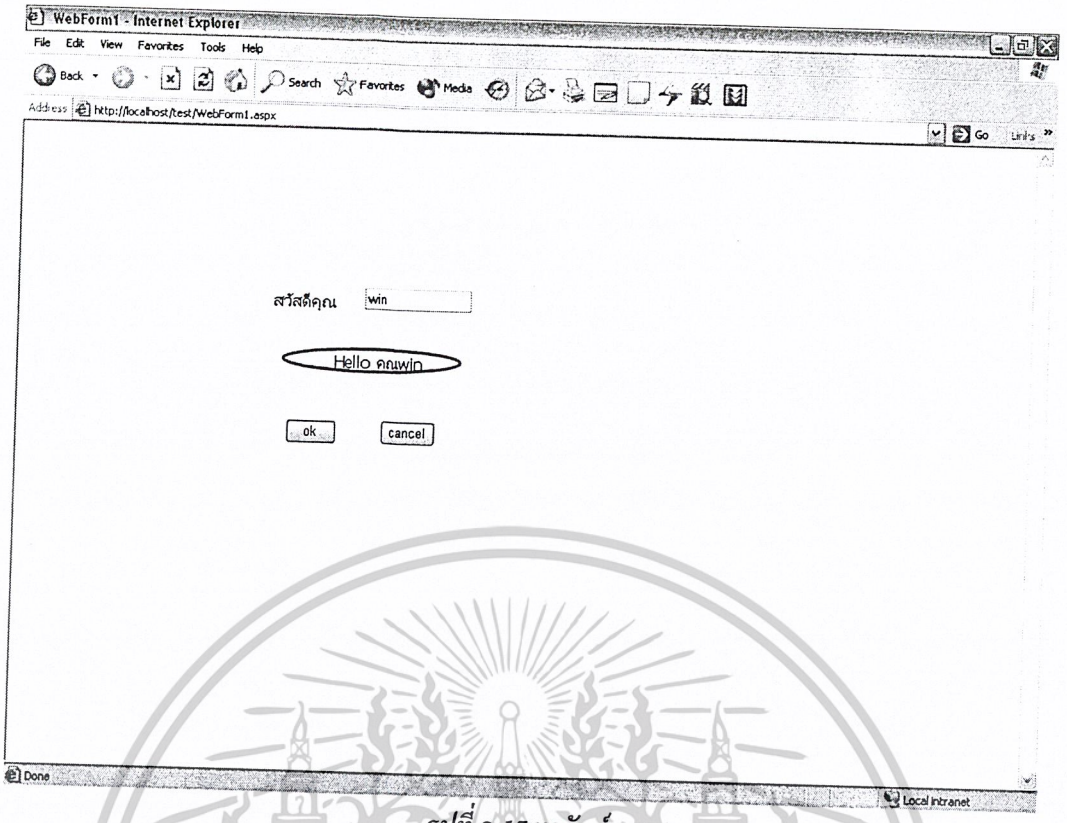
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-16 Code ที่เรียกใช้ services

3.5 ถ้าเราต้องการให้มันแสดงผลเราก็กดปุ่ม F5 เพื่อเป็นการสร้าง project จากนั้นเราก็ทำตามเหตุการณ์ต้องการที่เราต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก-17 ผลลัพธ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] สราวุธ อ้อยศรีสกุล, ถอดรหัส .NET + Web Services, H.N. Group, 2544

[2] บัญชา ปะสีละเตสัง, การเขียนโปรแกรม ASP .NET ด้วย VB.NET และ C#, บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน), 2546

[3] จักรพันธุ์ โพธิวรรณา, อัมรินทร์ เพ็ชรกุล, Microsoft Visual Studion .net, บริษัท ซัคเซส มีเดีย จำกัด

[4] สุรสิทธิ์ คิวประสพศักดิ์, นันทนี แขวงโสภา, อินไซท์ Visual Basic .NET (ฉบับสมบูรณ์), บริษัท โปรวิชั่น จำกัด, 2546

[5] ทวีชัย หงษ์สุมาลย์, สงวนชัย สุวรรณชีวะศิริ, อินไซท์ ASP.NET (ฉบับสมบูรณ์), บริษัท โปรวิชั่น จำกัด, 2546

[6] Harvey M. Deitel, Paul J. Deitel, Tem R. Nieto, Visual Basic. NET How to Program Second Edition, Deitel, 2002

[7] Harvey M. Deitel,Visual Basic .NET For Experienced Programmers , Deitel, 2002

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้