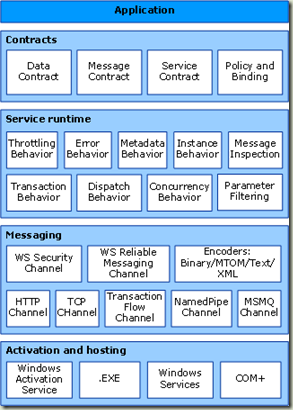
**บทที่ 2**

**ทฤษฎีและงานวิจัยที่เกี่ยวข้อง**

**2.1 Windows Communication Foundation**

Windows Communication Foundation (WCF) มีวัตถุประสงค์คือ เพื่อช่วยเพิ่มความสามารถของ ASP.NET Web Service ที่มีข้อจำกัดในด้านเครื่องมือและเทคโนโลยีที่มีความแตกต่าง จะทำให้สื่อสารกันได้ลำบาก ไมโครซอฟต์พัฒนา WCF ขึ้นเพื่อรองรับลักษณะงานประเภท SOA ในกรณีที่มีระบบหรือผู้ใช้ที่มีความหลากหลายในด้านมาตรฐานการสื่อสาร (Protocol) ให้สามารถสื่อสารกันได้เป็นรูปแบบเดียวกัน จะช่วยให้มีประสิทธิภาพการทำงานที่รวดเร็วขึ้นกว่าเดิม สามารถนำ Service ที่พัฒนาไว้นั้นไปใช้งานบน Host ชนิดต่างๆ ได้



**รูปที่ 2.1** โครงสร้างพื้นฐานของ WCF โดยได้แบ่งเอาไว้เป็นชั้นๆ (Layer)

* Application Layer คือส่วนที่ทำการ implement code ตัวเซอร์วิสไว้ เช่น method, code logic, entity ต่างๆ ที่ใช้ในการทำงานของตัวเซอร์วิส
* Contract Layer เป็นชั้นที่เชื่อมต่อกับ Application Layer ซึ่งนักพัฒนาจะทำงานโดยตรงกับส่วนนี้เพื่อใช้ในการสร้างตัวเซอร์วิสต่างๆ สามารถแบ่งได้ ดังนี้

1. **Data Contract**  เป็นส่วนที่กำหนดโครงสร้างข้อมูลต่างๆ ที่มีรับส่งข้อมูลทั้งเข้าและออกโดย Operation Contract ของ Service Contracts ซึ่งอาจจะประกอบไปด้วย ชนิดข้อมูลพื้นฐาน (Primitive Data Type) เช่น string, int, decimal, float, double, boolean  และกลุ่มข้อมูลโครงสร้างที่ซับซ้อน (Complex Data Type) อย่างเช่น Class, Enumeration Data ต่างๆ
2. **Message Contract** โดยปกติข้อมูลที่ส่งข้ามกันระหว่าง Service และ Client นั้น ถูกกำหนดรูปแบบอยู่ใน SOAP Message แต่บางครั้งเนื่องจากความต้องการในการรองรับการทำงานต่างๆ เช่น ต้องการจัดรูปแบบโครงสร้างข้อมูลใหม่, อยากใช้ Header, การเพิ่มเติม Status เหล่านี้  เราจะต้องอาศัย Message Contract เป็นตัวช่วย
3. Service Contracts ส่วนที่บ่งบอกว่าเซอร์วิสที่เราสร้างขึ้นนั้นมี operation (Method) อะไรไว้ให้บริการบ้าง
4. **Policies and Binding** ส่วนนี้เป็นตัวกำหนดตั้งค่าเงื่อนไขและวิธีการให้เซอร์วิสมีการติดต่อสื่อสารในรูปแบบใด (Protocol) ใช้ระบบรักษาปลอดภัย (Security) แบบไหน

- Service Runtime ส่วนที่กำหนดพฤติกรรม (Behavior) การทำงานของเซอร์วิส ขณะที่เซอร์วิสทำงานอยู่ มีสามารถแบ่งได้ ดังนี้

1. Throttling Behavior ใช้กำหนดจำนวนของ Message ที่จะเข้ามาทำงานในเซอร์วิส
2. Error Behavior หากมีปัญหาเกิดขึ้นภายในเซอร์วิส ใช้มาช่วยทำการจัดการปัญหา ดังกล่าว
3. Metadata Behavior ใช้กำหนดการเปิดเผยข้อมูลโครงสร้าง (Meta Data) ของเซอร์วิส แก่ผู้ใช้งาน
4. Instance Behavior ใช้กำหนดจำนวนของผู้ใช้ที่จะเข้ามาใช้งานในเซอร์วิสโดยพร้อม ๆ กันขณะที่เซอร์วิสทำงานอยู่
5. Transaction Behavior ช่วยในการย้อนกลับ (Rollback) การทำงานหาก Operationเกิดการทำงานที่ผิดพลาดขึ้น
6. Dispatch Behavior เป็นตัวควบคุมว่า Message ต่างๆ นั้นจะเข้าการโปรเซสด้วยสถาปัตยกรรมใดของ WCF

#### **Messaging** Layer เป็นการจัดการ Message ที่ส่งเข้าออกภายในเซอร์วิส ทั้งการกำหนด รูปแบบของ Message โดยการวิธีการแลกเปลี่ยน Message ซึ่งแบ่งได้เป็นสองกลุ่มดังนี้

1. Transport channels ทำการอ่านและเขียน Message จากเน็ตเวิร์กภายในหรือภายนอกองค์กร มี transports บางตัวจำเป็นต้องมีการแปลง (encoder) Message จากหน่วยสตรีมไบท์ที่ใช้รับส่งกันในเน็ตเวิร์กมาอยู่ในรูป XML เช่น HTTP, named pipes, TCP, และ MSMQ
2. Protocol channels เป็นการทำงานโปรโตคอลระดับ Message โดยส่วนใหญ่เป็นการเขียนอ่านในส่วนของหัว Message เช่น WS-Security Protocol เป็นการกำหนดใช้ความปลอดภัยในระดับชั้น Message และ WS-Reliability Protocol เป็นการการันตีการส่ง Message ระหว่างกัน

#### **Activation and Hosting** เซอร์วิสที่ได้สร้างไว้เป็นเพียง Library เท่านั้น เมื่อต้องการนำไปรันบน Host เพื่อให้เซอร์วิสสามารถใช้งานได้ ซึ่งมีหลายวิธีดังนี้

#### Windows Activation Service (WAS) มาพร้อมกับ IIS 7.0 จะทำงานเมื่อมีการร้องขอ รองรับได้หลายโปรโตคอล HTTP, TCP, Named Pipes, MSMQ

1. Self-Hosting สามารถแปลงให้อยู่ในรูป exe ได้เช่น console app, win app, WPF app เป็นต้น ไม่สามารถทำงานอัตโนมัติได้ ต้องรันโปรแกรมทุกครั้งเพื่อให้ Host ทำงาน
2. Windows Service คล้ายกับ Executable Host แต่จะทำงานอัตโนมัติทุกครั้งที่ เริ่มระบบ Windows
3. COM+ ส่วนใหญ่ไม่มีคนได้ใช้งาน จะเกี่ยวกับ Windows Component Programming

**2.2 Web Application**

Web Application เป็นแอพพลิเคชันที่สามารถเรียกใช้งานผ่านบราวเซอร์ต่างๆ โดยที่ผู้ใช้งานไม่ต้องติดตั้งแอปพลิเคชันเหล่านี้ ลงไปในเครื่องคอมพิวเตอร์ Web Application จะถูกติดตั้งไว้ที่ Server คอยให้บริการกับ Client ทำให้สามารถเรียกใช้งานได้ทุกที่ทุกเวลาผ่านระบบอินเตอร์เน็ต

ข้อดีของ Web Application

* เหมาะกับองค์กรขนาดเล็กเพราะมีค่าใช้จ่ายต่ำ
* การใช้งานในองค์กรทำได้ง่าย เพียงแค่มีเว็บบราวเซอร์ก็สามารถใช้งานได้
* ข้อมูลจัดเก็บที่เดียว ง่ายต่อการจัดการ และไม่เกิดความซ้ำซ้อน
* ไม่จำเป็นต้องใช้เครื่องคอมพิวเตอร์ที่มีประสิทธิภาพสูงซึ่งมีราคาแพง
* สามารถทำงานได้ทันที โดยไม่ต้องติดตั้งโปรแกรม
* ไม่ต้องมีบุคคลากรด้านเทคนิคเป็นของตัวเอง เพราะผู้ให้บริการดูแล Server และการบำรุงรักษาเองทั้งหมด
* ส่วนมากใช้ได้หลากหลายระบบปฏิบัติการ เช่น Windows, Linux และ Mac ทำให้องค์กรสามารถเลือกใช้ และลดค่าใช้จ่ายได้
* เชื่อมต่อกับ Web Application ตัวอื่นหรือบริการออนไลน์อื่นๆได้ง่าย

ข้อเสียของ Web Application

- การใช้งานมีได้จำกัด อาจไม่เหมาะกับงานบางประเภทที่ต้องการรูปแบบโปรแกรมที่แตกต่างจากโปรแกรมทั่วไปเช่น โปรแกรมตกแต่งรูป โปรแกรมตัดต่อวีดีโอ

- Web Application ส่วนใหญ่ต้องการอินเตอร์เน็ตในการใช้งานเสมอ แต่บางตัวที่สามารถทำงานแบบออฟไลน์ได้

**2.3 Application Programming Interface**

**Application Programming Interface (API)** คือช่องทางการเชื่อมต่อระหว่างเว็บไซต์หนึ่งไปยังอีกเว็บไซต์หนึ่ง หรือเป็นการเชื่อมต่อระหว่างผู้ใช้งานกับ Server หรือจาก Server เชื่อมต่อไปหา Server ซึ่ง API เปรียบได้เป็นภาษาคอมพิวเตอร์ที่ทำให้คอมพิวเตอร์สามารถสื่อสารและแลกเปลี่ยนข้อมูลกันได้อย่างอิสระ

API ทำหน้าที่ช่วยในการเข้าถึงข้อมูลต่างๆ หรือจะเป็นการจัดการข้อมูลต่างๆบนเว็บไซต์ โดยเจ้าของเว็บไซต์ที่มี API สามารถกำหนดขอบเขตในการเข้าถึงบริการต่างๆของทางเว็บไซต์ได้ ประโยชน์ของ API สามารถแบ่งออกมาได้หลายอย่างคือ

* ช่วยในการพัฒนาเว็บไซต์หรือ Application ได้ง่ายและรวดเร็วซึ่ง API จะเป็นตัวช่วยที่นักพัฒนาไม่ต้องเข้าไปแก้ไข Code คำสั่งเลยทำให้สะดวกสบายในการใช้งาน
* ช่วยให้นักพัฒนาเว็บไซต์หรือเจ้าของเว็บไซต์สามารถขยายฐานผู้ชมเว็บไซต์ให้มากขึ้น
* ทำให้ผู้ใช้งานเว็บไซต์ต่างๆที่มีการติดตั้ง API ของอีกเว็บไซต์หนึ่ง ไม่ต้องเข้าหน้าเว็บไซต์ที่เป็นเจ้าของ API เกิดสะดวกในการใช้งานของผู้ใช้งานเว็บไซต์
* API สามารถรับส่งข้อมูลข้าม Server ได้

ปัจจุบันเว็บไซต์ขนาดใหญ่ หลายเว็บไซต์จะมีการเปิดให้ใช้งาน API ซึ่งสามารถเห็นการใช้งาน API ได้มากขึ้นโดยเฉพาะเว็บไซต์ที่เน้นด้านการติดต่อสื่อสาร Social Network และ E-commerce เช่น บริการของ Amazon มี API ที่เปิดให้ผู้ที่สนใจได้นำสินค้าที่มีขายอยู่ใน Amazon ไปติดไว้ในเว็บไซต์หรือบล็อคของตัวเองได้ และบริการของ Facebook ที่สามารถให้ผู้ใช้โปรโมตเพจของตนเองผ่านเว็บไซต์หรือบล็อคได้

**2.4 ภาษา C#**

C# คือ ภาษาคอมพิวเตอร์ประเภทการเขียนโปรแกรมเชิงวัตถุ (Object - Oriented Programming หรือ OOP) พัฒนาโดยบริษัทไมโครซอฟท์ (Microsoft) โดยมีจุดมุ่งหมายในการรวมความสามารถการคำนวณของ C++ ด้วยการโปรแกรมที่ง่ายกว่าของ Visual Basic โดย C# มีพื้นฐานจาก [C++](http://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/2183-c++-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html)และมีส่วนการทำงานคล้ายกับภาษา [Java](http://www.mindphp.com/%E0%B8%84%E0%B8%B9%E0%B9%88%E0%B8%A1%E0%B8%B7%E0%B8%AD/73-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3/2185-java-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3.html)  
     C# ได้รับการออกแบบให้ทำงานร่วมกับ .NET platform ของไมโครซอฟท์ จุดมุ่งหมายคือ อำนวยความสะดวกในการแลกเปลี่ยนสารสนเทศและบริการผ่านเว็บ และทำให้ผู้พัฒนาสร้างโปรแกรมประยุกต์ในขนาดเล็ก C# ทำให้โปรแกรมสื่อสารง่ายขึ้นผ่านการใช้ Extensible Markup Language (XML) และ Simple Object Access Protocol (SOAP) ซึ่งยอมให้เข้าถึงวัตถุ (Object) ของโปรแกรมหรือเมธอด โดยปราศจากความต้องการให้ผู้เขียนโปรแกรมเขียนคำสั่งเพิ่มในแต่ละขั้นตอน เนื่องจากผู้เขียนโปรแกรมสามารถสร้างบนคำสั่งที่มีอยู่ แทนที่การคัดลอกซ้ำ C#  ภาษา C# ถูกพัฒนาขึ้นโดยเป็นส่วนหนึ่งในการพัฒนาโครงสร้างพื้นฐานของ .NET Framework เป็นการการนำข้อดีของภาษาต่างๆ (เช่นภาษา Delphi, ภาษา C++) มาปรับปรุงเพื่อให้มีความเป็นโปรแกรมเชิงวัตถุมากขึ้น ขณะเดียวกันก็ลดความซับซ้อนในโครงสร้างของภาษาลง ซึ่งเรียบง่ายกว่าภาษา C++ และมีสิ่งที่เกินความจำเป็นน้อยลงเมื่อเทียบกับJava  
     C# ถูกรับรองจากหน่วยงาน ECMA ซึ่งเป็นหน่วยงานกำหนดมาตรฐานสากลด้านเทคโนโลยีสารสนเทศ และ ISO และปัจจุบันไมโครซอฟท์ยังพัฒนาภาษานี้อย่างต่อเนื่อง

**2.5 GIT**

**Git** คือ Version Control ประเภทหนึ่ง ซึ่งเป็นระบบที่มีหน้าที่ในการจัดเก็บการเปลี่ยนแปลงของไฟล์ในโปรเจค มีการสำรองโค้ดที่เขียนไว้ โดยสามารถที่จะเรียกดู หรือย้อนกลับไปดู Version ต่างๆ ของโปรเจคที่ใดเวลาใดก็ได้ หรือแม้แต่ดูว่าไฟล์นั้นๆใครเป็นคนเพิ่มหรือแก้ไข หรือจะดูว่าไฟล์นั้นๆ ถูกเขียนโดยใครบ้างก็ได้ ทำให้ **Git** เป็นเครื่องมือที่เหมาะอย่างยิ่งสำหรับนักพัฒนา จะช่วยให้การพัฒนาแบบเป็นทีมมีประสิทธิภาพในการพัฒนามากขึ้น **โดย** Git **มีคุณสมบัติคร่าวๆดังนี้**

- สามารถ Backup และ Restore ไฟล์ได้

- Synchronization ไฟล์ทำให้ไฟล์นั่นเป็นเวอร์ชันใหม่เสมอไม่ว่าจะมีผู้ใช้งานไฟล์นั่นกี่คนก็ตาม

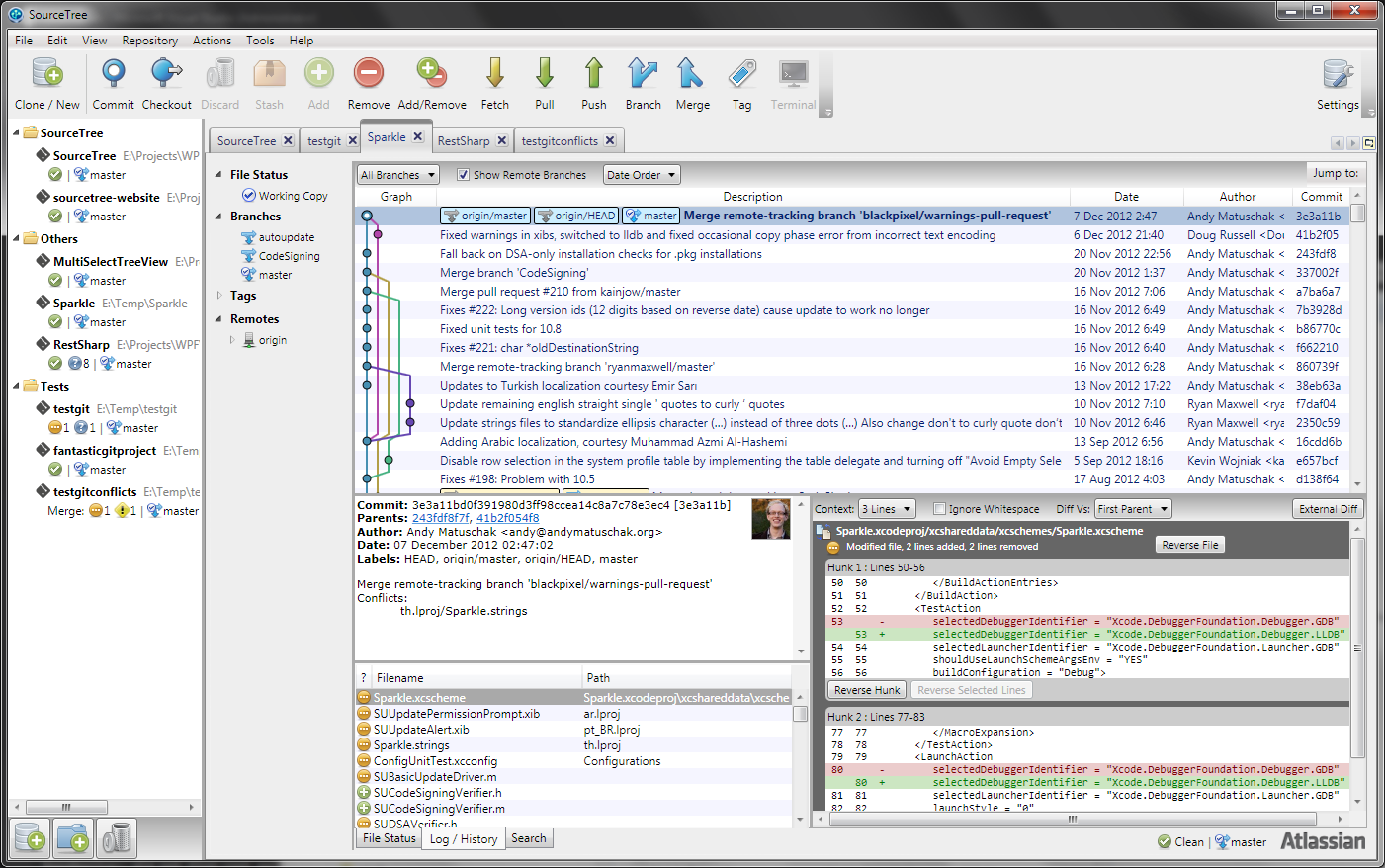
- สามารถ Track back history ของแต่ละไฟล์ได้

- ใน History นั่นสามารถบอกรายละเอียดได้มากกว่าการจัดเก็บแบบธรรมดา คือสามารถบอก

ได้ว่าไฟล์นี้ใครเป็นเจ้าของ และใครบ้างที่แก้ไขไฟล์นี้

- ควบคุมจัดการได้ง่ายเมื่อต้องการควบคุมโค้ดที่พัฒนาโดยโปรแกรมเมอร์หลายคนในทีม

ปัจจุบัน Git เป็นสิ่งที่ขาดไม่ได้ในการพัฒนาซอฟแวร์ โดยมีบริการให้เลือกใช้มากมาย โดยนักพัฒนาส่วนใหญ่นิยมใช้บริการของ Github และ Bitbucket ในการฝากไฟล์โปรเจค



**รูปที่ 2.2** โปรแกรม SourceTreeที่ใช้สำหรับจัดการ Git

**2.6 ภาษา HTML**

HTML หรือ Hyper Text Markup Language เป็นภาษามาร์กอัป (Markup) ที่ใช้ในการสร้างเว็บเพจ ถูกพัฒนาและกำหนดมาตรฐานโดยองค์กร World Wide Web Consortium (W3C) สำหรับการสร้างเว็บเพจโดยใช้ภาษา HTML สามารถทำได้โดยใช้โปรแกรม Text Editor หรือเครื่องมือช่วยสร้างเว็บเพจ เช่น Dreamweaver ส่วนการเรียกใช้งานหรือทดสอบการทำงานของเอกสารHTML จะใช้บราวเซอร์ เช่น Chrome, Internet explorer, Firefox เป็นต้น

โครงสร้างของ HTML จะประกอบไปด้วยส่วนของคำสั่ง 2 ส่วน คือ ส่วนที่เป็นส่วนหัว (Head) และส่วนที่เป็นเนื้อหา (Body) โดยมีรูปแบบคำสั่งดังนี้

1) คำสั่งหรือแท็ก (Tag)

แท็กคือคาสั่งที่ใช้ในภาษาเอชทีเอ็มแอลอยู่ภายในเครื่องหมายน้อยกว่า (<) และเครื่องหมายมากกว่า (>) แท็กในภาษาเอชทีเอ็มแอลแบ่งออกเป็นสองชนิดดังนี้

- แท็กเดี่ยว เป็นแท็กที่ไม่ต้องมีแท็กปิด เช่น <HR> <BR>

- แท็กคู่ เป็นแท็กที่ประกอบด้วยแท็กเปิดและแท็กปิด โดยแท็กปิดจะต้องใส่เครื่องหมายทับ (/) นำหน้าคำสั่งในแท็กนั้น ๆ เช่น <b>....</b>, <p>….</p> เป็นต้น

คำสั่งที่สำคัญยกตัวอย่าง เช่น

คำสั่ง <HTML> เป็นคำสั่งเริ่มต้นในการเขียนโปรแกรมและคำสั่ง </HTML> เป็นการสิ้นสุดโปรแกรม คำสั่งนี้จะไม่แสดงผลในโปรแกรมเว็บบราวเซอร์ แต่ต้องเขียนเพื่อให้เกิดความเป็นระบบของงาน และเพื่อแจ้งให้เว็บบราวเซอร์ทราบว่าเอกสารนี้เป็นเอกสารของภาษา HTML

คำสั่ง <HEAD> คือ ส่วนหัวเป็นส่วนที่ใช้อธิบายข้อมูลเฉพาะของหน้าเว็บนั้นๆ เช่น ชื่อเรื่องของหน้าเว็บ (Title) ชื่อผู้จัดทำเว็บ (Author) คีย์เวิร์ดสาหรับการค้นหา (Keyword) เป็นต้น

คำสั่ง <BODY> คือส่วนเนื้อหาเป็นส่วนเนื้อหาหลักของหน้าเว็บ ซึ่งการแสดงผลจะต้องใช้แท็กจำนวนมาก ขึ้นอยู่กับลักษณะของข้อมูล เช่น ข้อความ รูปภาพ เสียง วีดิโอ หรือไฟล์ต่างๆ

2) แอทริบิวต์ (Attribute)

แอทริบิวต์เป็นส่วนขยายในแท็ก ใช้สาหรับจัดรูปแบบเพิ่มเติม เช่น ขนาด สีระยะห่าง

เป็นต้น ค่าของแอทริบิวต์จะอยู่ในเครื่องหมาย “…” เช่น

<p align="right"> ข้อความในย่อหน้านี้จัดวางในตำแหน่งชิดขวาหน้าจอ</p>

<a href = “https://www.google.com” > ไปยังเว็บไซต์ google.com </a>

**2.7 ภาษา CSS**

**CSS** ย่อมาก Cascading Style Sheets เป็นภาษาที่มีรูปแบบการเขียน Syntax ที่เฉพาะและถูกกำหนดมาตรฐานโดย W3C (World Wide Web Consortium) เช่นเดียวกับภาษา HTML และ XHTML ใช้สำหรับตกแต่งเอกสาร HTML/ XHTML ให้มีหน้าตา สีสัน ตัวอักษร เส้นขอบ พื้นหลัง ระยะห่าง ฯลฯ อย่างที่เราต้องการ ด้วยการกำหนดคุณสมบัติให้กับ Element ต่างๆ ของ HTML เช่น <body>, <p>, <h1> เป็นต้น

**ประโยชน์ของ CSS** มีคร่าวๆ ดังนี้

* การใช้ CSS ในการจัดรูปแบบการแสดงผล จะช่วยลดการใช้ภาษา HTML ในการตกแต่งเว็บเพจ ทำให้ code ใน HTML เหลือเพียงส่วนเนื้อหา ทำให้เข้าใจง่ายขึ้น การแก้ไขเอกสารทำได้ง่ายและรวดเร็ว
* เมื่อ code ภายในเอกสาร HTML ลดลง ทำให้ขนาดไฟล์เล็กลง ดาวน์โหลดได้เร็วขึ้น
* สามารถกำหนดรูปแบบการแสดงผลจากคำสั่ง style sheet ชุดเดียวกัน ให้มีผลกับเอกสาร HTML ทั้งหน้า หรือทุกหน้าได้ ทำให้เวลาแก้ไขหรือปรับปรุงทำได้ง่าย ไม่ต้องตามแก้ที่ tag ต่างๆ ทั่วทั้งเอกสาร
* สามารถควบคุมการแสดงผลให้เหมือนกัน หรือใกล้เคียงกันได้ในหลายบราวเซอร์
* สามารถกำหนดการแสดงผลในรูปแบบที่เหมาะกับสื่อชนิดต่างๆ ไม่ว่าจะเป็นการแสดงผลบนหน้าจอ, บนกระดาษเมื่อสั่งพิมพ์, บนมือถือ หรือบน PDA โดยที่เป็นเนื้อหาเดียวกัน

**2.8 ภาษา JavaScript**

จาวาสคริปต์เป็นภาษาสคริปต์ที่มีลักษณะการเขียนเชิงโพรโทไทป์ (Prototype-based Programming) ได้รับการพัฒนาขึ้นโดย Netscape โดยมีวัตถุประสงค์ เพื่อช่วยให้เว็บเพจสามารถแสดงเนื้อหาที่มีการเปลี่ยนแปลงได้ตามเงื่อนไขหรือสภาพแวดล้อมต่างๆ หรือสามารถตอบโต้ตอบกับผู้ใช้งานได้มากขึ้น ทั้งนี้เพราะภาษา HTML ไม่สามารถโต้ตอบกับผู้ใช้งานได้ทันที แต่ใช้แสดงผลเอกสารที่มีเนื้อหาคงที่แน่นอน

การทำงานของจาวาสคริปต์เกิดขึ้นบนบราวเซอร์ (Client-Side Script) มีข้อจากัดคือ ไม่สามารถรับและส่งข้อมูลต่างๆ กับเซิร์ฟเวอร์ได้โดยตรง เช่น การอ่านไฟล์จากเซิร์ฟเวอร์ เพื่อนำมาแสดงบนเว็บเพจ หรือรับข้อมูลจากผู้ใช้งาน เพื่อไปเก็บบนเซิร์ฟเวอร์ เป็นต้น ดังนั้นงานในลักษณะดังกล่าว ต้องอาศัยภาษาฝั่งเซิร์ฟเวอร์ เช่น พีเอชพี(PHP) เพิร์ล (Perl) เอเอสพี (ASP) เป็นต้น

ปัจจุบันมีการใช้จาวาสคริปต์ที่ฝังอยู่ในเว็บบราวเซอร์หลายรูปแบบ เช่น ใช้เพื่อสร้างเนื้อหาที่เปลี่ยนแปลงเสมอภายในเว็บเพจ ใช้เพื่อตรวจสอบความถูกต้องของข้อมูลที่ผู้ใช้กรอกก่อนบันทึกลงระบบ ใช้เพื่อเข้าถึงข้อมูลที่อยู่ภายใต้โครงสร้างแบบ DOM เป็นต้น

**2.9 jQuery**

jQuery คือไลบารี่ของภาษาจาวาสคริปต์ ซึ่งถูกออกแบบมาเพื่อให้การเขียนจาวาสคริปต์นั้น ง่ายขึ้น เพราะสามารถเขียนโค้ดยาวๆ ของจาวาสคริปต์ให้สั้นลงได้ jQueryถูกพัฒนาโดย John Resig โดยเป็นการพัฒนาแบบโอเพนซอร์ส

คุณสมบัติหลักของ jQuery มีดังนี้

* การเข้าถึงเอชทีเอ็มแอลและซีเอสเอส

ส่วนนี้ถือได้ว่าเป็นคุณสมบัติหลักของ jQuery เพราะทำให้ผู้ใช้งานสามารถเข้าถึง อีลีเมนต์เอชทีเอ็มแอลและตัวเลือกซีเอสเอสต่างๆ ได้ง่ายขึ้น

* การจัดการเอชทีเอ็มแอลและซีเอสเอส

คุณสมบัติในหัวข้อที่แล้วเกี่ยวกับการเข้าถึงอีลีเมนต์ต่าง ๆ แต่สาหรับคุณสมบัตินี้จะใช้ในการดัดแปลง แก้ไข เพิ่ม หรือลดอีลีเมนต์ต่างๆ

* เหตุการณ์ (Events) ในจาวาสคริปต์

สำหรับจาวาสคริปต์จะพบปัญหาในเรื่องของเหตุการณ์บ่อยครั้ง เนื่องจากความไม่เข้ากันของเว็บบราวเซอร์ แต่เมื่อใช้ jQuery ปัญหาที่เกิดขึ้นจะลดน้อยลงหรือหมดไป เพราะjQuery จะช่วยตรวจสอบชนิดของเว็บบราวเซอร์ให้โดยอัตโนมัติ

* วิชวลเอฟเฟกต์และแอนิเมชัน (Visual Effects and Animation)

วิชวลเอฟเฟกต์และแอนิเมชันเป็นอีกเทคนิคหนึ่งที่กำลังได้รับความนิยมสำหรับการทำเว็บไซต์ในปัจจุบัน

* เอแจ็กซ์ (Ajax)

เอแจ็กซ์ก็เป็นอีกหนึ่งรูปแบบของการเขียนเว็บแอปพลิเคชันในปัจจุบัน โดย jQuery มีฟังก์ชันสำหรับการเรียกใช้งานเอแจ็กซ์ให้ง่ายขึ้นกว่าเดิม

* ส่วนประสานงานผู้ใช้ (User Interface)

ส่วนประสานงานผู้ใช้คือส่วนควบคุมต่างๆ ที่ควรมีหรืออยากให้มี เช่น เมนู (Menu) ปฏิทิน (Calendar) เป็นต้น ทำให้ผู้ใช้งานใช้งานได้สะดวกมากยิ่งขึ้น

**2.10 Ajax**

**Ajax** (Asynchronous JavaScript and XML) เป็นกลุ่มของเทคนิคในการพัฒนาเว็บแอปพลิเคชันเพื่อให้ความสามารถในด้านการโต้ตอบกับผู้ใช้งานได้ดีขึ้น โดยการรับส่งข้อมูลจะอยู่ในส่วนหลังบ้าน (Backend) ทำให้หน้าเว็บเพจไม่ต้องโหลดใหม่ทั้งหมดทุกครั้งที่มีการเปลี่ยนแปลง ส่งผลให้หน้าเว็บเพจสามารถตอบสนองต่อผู้ใช้งานได้รวดเร็วมากยิ่งขึ้น **Ajax** ไม่ใช่เทคโนโลยีที่คิดค้นขึ้นมาใหม่ แต่เป็นเทคนิคที่นำเทคโนโลยีหลายอย่างที่มีอยู่แล้วมารวมกันดังต่อไปนี้

* การแสดงผลลัพธ์และรูปแบบข้อมูล ใช้ภาษาเอชทีเอ็มแอลและซีเอสเอส
* การแสดงข้อมูลที่การเปลี่ยนแปลงให้โต้ตอบกับผู้ใช้งานทันที โดยใช้อีซีเอ็มเอสคริปต์ (ECMAScript) เช่น ภาษาจาวาสคริปต์ในการเข้าถึง เช่น ดอม (Document Object Model: DOM)
* การเลือกเปลี่ยนข้อมูลโดยที่ช่วงเวลาไม่สอดคล้องกันกับเซิร์ฟเวอร์ โดยใช้เอ็กเอชอาร์ (XMLHttpRequest: XHR)
* รูปแบบข้อมูลที่ใช้ในการแลกเปลี่ยน เช่น เอ็กเอ็มแอล เจสัน (JSON) อีบีเอ็มแอล (EBML) หรือ เพลนเท็กซ์

การทำงานของเว็บแอปพลิเคชันที่ใช้เทคนิค **Ajax** จะเป็นการทำงานแบบช่วงเวลาไม่ตรงกันหรือการติดต่อสื่อสารแบบไม่ต่อเนื่อง (**Asynchronous**) โดยเซิร์ฟเวอร์จะทำการส่งผลลัพธ์ไปยังเว็บเพจให้ผู้ใช้งานทันที โดยไม่ต้องรอให้ประมวลผลเสร็จก่อน เว็บเพจของผู้ใช้งานจะทำการดึงข้อมูลในส่วนต่างๆ ภายหลัง



**ภาพที่ 2.2** เปรียบเทียบหลักการทำงานระหว่างเว็บแอปพลิเคชันทั่วไปและเว็บแอปพลิเคชันที่ใช้ **Ajax**

**2.11 ภาษา SQL**

**SQL (Standard relational database Query Language)** ถือเป็นภาษามาตรฐานที่ใช้สาหรับระบบฐานข้อมูล เป็นส่วนหนึ่งของระบบฐานข้อมูลแบบรีเลชั่นเนล ที่ได้รับความนิยมมากเพราะเป็นภาษาที่เข้าใจง่าย ภาษา **SQL** แบ่งเป็น 3 ประเภท คือ

* ภาษานิยามข้อมูล (Data Definition Language : DDL) เป็นคำสั่งที่ใช้ในการสร้างฐานข้อมูล กำหนดโครงสร้างข้อมูลว่ามี Attribute ใด ชนิดของข้อมูล รวมทั้งการเปลี่ยนแปลงตาราง และการสร้างดัชนี เช่น คำสั่ง CREATE, DROP, ALTER เป็นต้น
* ภาษาจัดการข้อมูล (Data Manipulation Language : DML) เป็นคำสั่งที่ใช้ในการเรียกใช้ เพิ่ม ลบ และเปลี่ยนแปลงข้อมูลในตาราง เช่น คำสั่ง SELECT, INSERT, UPDATE, DELETE เป็นต้น
* ภาษาควบคุมข้อมูล (Data Control Language : DCL) เป็นคำสั่งที่ใช้ในการกำหนดสิทธิการอนุญาต หรือ ยกเลิก การเข้าถึงฐานข้อมูล เพื่อป้องกันความปลอดภัยของฐานข้อมูล เช่น คำสั่งGRANT, REVOKE เป็นต้น

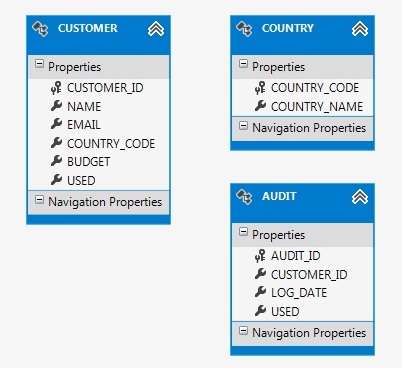
**2.12 Entity Framework**

Entity Framework คือ Library ที่ถูกสร้างมาเพื่อลดปัญหาของมาตรฐานในการติดต่อกับฐานข้อมูลบน .Net Framework ซึ่งในการเขียนโปรแกรม การออกแบบคลาสเพื่อติดต่อกับฐานข้อมูลมีความสำคัญอย่างมาก ยิ่งโปรแกรมมีความซับซ้อนมากจะทำให้จัดการได้ยาก เนื่องจากนักพัฒนาแต่ละคนอาจจะเรียกใช้งานคลาสแตกต่างกัน มีการเปิด-ปิดฐานข้อมูล เพื่อเขียน SQL Statement ให้ตรงความต้องการ จะเกิดการดึงข้อมูลออกมาใช้งานซ้ำๆ ไม่มีมาตรฐานว่าต้องเรียกใช้ตอนไหน และเรียกใช้อย่างไร Entity Framework จึงถูกออกแบบมาเพื่อแก้ปัญหานี้ โดยจะทำหน้าที่เป็น Entity Data Model อยู่ในชั้น Data Layer เป็นรูปแบบ **Object/Relational Mapping (O/RM)**



**รูปที่ 2. Entity Framework ทำหน้าที่เป็นตัวกลางในการติดต่อระหว่าง Application กับ Database**

Entity Framework จะสร้าง Layer ทำหน้าที่เป็น Database Model ขึ้นมาเป็นคลาสใน โปรเจค โดยจะ Mapping คลาสที่สร้างขึ้นใหม่กับ Table, View และ Procedure จากฐานข้อมูลมาไว้บนโปรเจค ซึ่งสามารถเรียกใช้ผ่านคลาสที่อยู่ในโปรเจคได้ โดยไม่ต้องเขียนคำสั่ง SQL Statement เช่น INSERT, UPDATE หรือ DELETE แล้วส่งไป Execute ที่ Table อีก



รูปที่ 2. Entity Framework จำลอง Model Entities

**Model Entities คือ Object ใหญ่สุดของ Entity Framework ซึ่งเปรียบเสมือนคลาสหลักที่รวบรวม Entity Set ที่ถูกตั้งชื่อให้เหมือนกับชื่อของ Table และ View มาไว้รวมกัน สามารถเพิ่มคุณสมบัติต่างๆ ของ Entity Set ได้ เช่น Relation, Data Type หรืออื่นๆ ภายในหนึ่งโปรเจคสามารถมีได้หลาย Model Entities และสามารถเรียกใช้งานได้มากกว่าหนึ่ง Entities ให้ทำงานพร้อมๆ กันได้**

ตัวอย่างการเปรียบเทียบการเขียน SQL Statement แบบปกติกับการเขียนแบบ Entity Framework

* การเขียนแบบปกติ

String sql = SELECT \* FROM CUSTOMER ORDER BY CUSTOMER\_ID ASC;

* การเขียนแบบ Entity Framework

var sql = db.CUSTOMER.OrderBy(x => (x.CUSTOMER\_ID)).ToList();

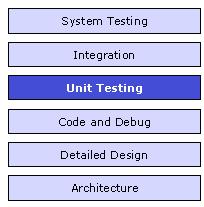
**2.13 Rabbit MQ**

Rabbit MQ คือ แอพพลิเคชั่นที่พัฒนาด้วยภาษา [Erlang](http://www.raspberrypithai.com/tag/erlang) เป็นตัวกลางในการสื่อสารหรือการกระจายข้อมูล เป็นการติดต่อสื่อสารกันได้ในระหว่างโปรเซส (Inter Process Communication : IPC) แบบ Message Queue ซึ่งสามารถช่วยให้แอพพลิเคชั่นต่างๆ สามารถสื่อสารกันได้โดย ผู้ส่งและผู้รับไม่จำเป็นต้องเขียนด้วยภาษาเดียวกัน ข้อมูลที่ส่งมีความปลอดภัยสูง Rabbit MQ รองรับการสื่อสารได้หลายโปรโตคอล เช่น AMQP 0-9-1, AMQP 1.0, MQTT, STOMP เป็นต้น

**2.14** **Unit Test**

Unit Test คือ การทดสอบระดับหน่วย เป็นการทดสอบรูปแบบหนึ่งที่ถูกทำโดยนักพัฒนา โดยใช้กับเมธอดหรือฟังก์ชันการทดสอบ คุณสมบัติ คลาส และองค์ประกอบ การทดสอบระดับหน่วยทำเพื่อตรวจสอบคุณภาพของโค้ดที่เขียน ทำงานได้ถูกต้องหรือไม่ ส่วนใหญ่มักนำการทดสอบระดับหน่วยเข้าไปใช้กับการพัฒนาด้วย เพื่อเป็นการป้องกันข้อผิดพลาดที่อาจละเลยไป

การทดสอบระดับหน่วยใช้กับการทดสอบการทำงานที่เป็นส่วนเล็กๆ โดยนักพัฒนาสามารถปรับปรุงแก้ไขโค้ดโดยไม่ต้องกังวลว่าถ้าหากมีการแก้ไขโค้ดบางส่วนขึ้นมาและจะกระทบกับการทำงานส่วนอื่นๆของโปรแกรม การทดสอบระดับหน่วยนี้จะทำการทดสอบโปรแกรมทีละโมดูล เพื่อหาข้อผิดพลาดที่จะเกิดขึ้นภายในการทำงานของแต่ละโมดูล จะทำก็ต่อเมื่อเขียนโค้ดเสร็จแล้ว ซึ่งหากรวมเข้ากับการพัฒนาแอพพลิเคชัน การทดสอบระดับหน่วยจะแสดงดังรูปที่ 2.



**รูปที่ 2.** แสดงการทดสอบระดับหน่วยในการพัฒนาแอพพลิเคชัน

จากรูปเห็นได้ว่าหลังจากเขียนโค้ดและหาข้อผิดพลาด (Debug) เรียบร้อยแล้ว จะต้องทำการทดสอบระดับย่อย เพื่อดูว่าแอพพลิเคชั่นแต่ละส่วนทำงานได้ถูกต้องหรือไม่ แล้วจะทำการรวม ก่อนจะทำการทดสอบแบบโดยรวมในขั้นตอนสุดท้าย

**2.15 ภาษา Gherkin**

ภาษา Gherkin คือ ภาษาหนึ่งที่ใช้สำหรับการทดสอบซอฟแวร์ ซึ่งอธิบายพฤติกรรมของซอฟท์แวร์ หรือ Business Logic โดยเป็นภาษามนุษย์ในรูปประโยคต่างๆ โดยจะบ่งบอกถึงการนำไปพัฒนาในการเขียนโปรแกรม โดย Gherkin มีลักษณะคือ ซึ่งประกอบด้วยคำอธิบายของเมธอดหรือฟังก์ชันที่ทำการทดสอบ ไฟล์ข้อมูลของ Gherkin จะมีชนิดเป็น .feature ภาษา Gherkin มีลักษณะเหมือนกับภาษา Python และ YAML โดย Gherkin เป็นภาษา line-oriented language ใช้ย่อหน้าสำหรับการกำหนดโครงสร้าง และในแต่ละบรรทัดจะจบด้วยคำสั่งและแต่ละบรรทัดจะเริ่มต้นด้วยคำหลักการคอมเม้นท์ในแต่ละบรรทัดในไฟล์ใช้เครื่องหมาย # ตัวแบ่งแยกคำจะแบ่งเป็นลักษณะ (Feature) ฉาก (Scenario) และขั้นตอน (Step) เมื่อทำการรันไฟล์ .feature จะทำการประมวลข้อมูลหลังจากคำหลักของแต่ละขั้นตอน

ตัวอย่างการเขียนภาษา Gherkin ดังรูปที่ 2.x เป็นการเขียนสำหรับการจัดการพาสเวิร์ด ซึ้งเป็นกรณีของการลืมพาสเวิร์ด โดยคำที่เป็นคำหลัก เช่น Feature, Scenario, Given, When, Then ไฟล์ดังกล่าวถูกเก็บในชื่อ password\_management.feature

Feature: Password management

Scenario: Forgot password

Given a user with email “cukes@cukes.info” exists

When I ask for a password reset

Then an email with a password reset link should be sent

**รูปที่ 2.x** แสดงตัวอย่างการเขียนภาษา Gherkin