บทที่ 2

เทคโนโลยีที่ใช้ในการดำเนินการวิจัย

2.1 อินทราเน็ต

อินทราเน็ต (Intranet) คือการประยุกต์ใช้เทคโนโลยีอินเทอร์เน็ตภายในองค์กร ซึ่งให้ประโยชน์หลายอย่างแก่องค์กร เช่น การใช้โปรแกรมค้นผ่านเว็บช่วยให้การจัดการข้อมูล และเผยแพร่ข่าวสารภายในบริษัท ทำได้สะดวกและง่ายดายขึ้น ในอดีตตัวบริการเว็บ (WEB Server) เน้นไปที่การเก็บข้อมูลที่ไม่มีการเปลี่ยนแปลง โดยใช้โปรแกรมค้นผ่านเว็บเป็นโปรแกรมแสดงข้อมูล ซึ่งใช้ เฮชทีทีพี โพรโทคอล(HTTP protocol)ในการโอนย้ายข้อมูลจากตัวบริการเว็บไปยังโปรแกรมค้นผ่านเว็บ สำหรับปัจจุบันเทคโนโลยีอินเทอร์เน็ตได้พัฒนาจนกระทั่งสามารถทำการบรรจุโปรแกรมผ่านเครือข่ายส่งไปประมวลผลที่โปรแกรมค้นผ่านเว็บได้ตามต้องการ อีกทั้งโปรแกรมที่ถูกบรรจุลงเครือข่ายยังสามารถทำงานบนระบบปฏิบัติการใดก็ได้ที่สนับสนุนการทำงานของจาวาเวอร์ชัว แมชชีน (Java virtual machine) ซึ่งโปรแกรมค้นผ่านเว็บที่มีอยู่ในปัจจุบันนี้สนับสนุนการทำงานของจาวาเวอร์ชัว แมชชีน

โครงสร้างขั้นพื้นฐานของอินทราเน็ต ประกอบด้วย เครือข่ายทีซีพี /ไอพี (TCP/IP) ที่เชื่อมโยงเครื่องให้บริการและเครื่องรับบริการเข้าด้วยกัน โดยที่อาจจะต่อเชื่อมไปยังเครือข่ายอินเทอร์เน็ตหรือไม่ก็ได้ การให้บริการของเครื่องให้บริการแก่เครื่องรับ บริการอาศัยโพรโทคอลมาตรฐานที่ใช้อยู่ในเครือข่ายอินเทอร์เน็ตดังแสดงไว้ในตารางที่ 2.1

Service	Protocol
Browsing	HTTP
File service	NFS
Mail Service	IMAP4/SMTP
Naming Service	DNS/NIS+
Directory Services	DNS/LDAP
Booting Services	Bootp/DHCP
Network Administration	SNMP
Object Services	IIOP(CORBA)

ตารางที่ 2.1 การให้บริการและโพรโทคอลที่ใช้ในแต่ละบริการบนเครือข่ายอินเทอร์เน็ต

2.1.1 ตัวบริการเว็บ

หมายถึงเครื่องคอมพิวเตอร์ที่มีโปรแกรมที่ทำหน้าที่ให้บริการในรูปแบบของเวิล์ด ไวด์ เว็บ (World wide web) ทำงานอยู่ โปรแกรมที่ทำงานนี้ เรียกว่าเซิฟเวอร์โปรเซส (server process) ทำหน้าที่รอรับข้อความร้องขอบริการจากโปรแกรมค้นผ่านเว็บที่ติดต่อเข้ามาโดยอาศัยมาตรฐานการสื่อสารเฮชทีทีพีโพรโทคอล โดยที่ระบบปฏิบัติการของเครื่องตัวบริการเว็บและโปรแกรมค้นผ่านเว็บไม่จำเป็นต้องเป็นแพลทฟอร์ม (platform) เดียวกัน กล่าวคือตัวบริการเว็บทำงานอยู่บนระบบปฏิบัติการยูนิกซ์ ในขณะที่โปรแกรมค้นผ่านเว็บทำงานอยู่บนระบบปฏิบัติการวินโดว์สเอ็นที หรือ วินโดว์ส 95 ก็ได้

2.1.2 โปรแกรมค้นผ่านเว็บ

คือโปรแกรมที่ใช้ในการสื่อสารกับตัวบริการเว็บ มีหน้าที่ดึงข้อมูลและแสดงผลข้อมูล ซึ่งข้อมูลอาจจะอยู่ที่เครื่องคอมพิวเตอร์ที่โปรแกรมค้นผ่านเว็บทำงานอยู่ หรืออยู่ในเครื่องตัวบริการเว็บที่ห่างไกลออกไป โดยผู้ใช้สามารถระบุแหล่งที่อยู่ของข้อมูลได้ในรูปแบบของ ยูอาร์แอล (Uniform Resource Locator) ซึ่ง โปรแกรมโปรแกรมค้นผ่านเว็บได้จัดเตรียมช่องสำหรับใส่ตำแหน่งของข้อมูลไว้ให้แล้ว เมื่อผู้ใช้ระบุยูอาร์แอลที่ต้องการลงไป โปรแกรมค้นผ่านเว็บจะทำการดึงข้อมูลนำมาแสดงบนหน้าจอของโปรแกรมค้นผ่านเว็บ

โปรแกรมค้นผ่านเว็บสามารถแสดงข้อมูลได้หลายรูปแบบ เช่น ข้อความ (text) ข้อความหลายมิติ (Hypertext) เสียง และ รูปภาพ ซึ่งมีลักษณะการเก็บข้อมูลในรูปแบบต่างๆกัน

2.1.3 เฮชทีทีพี โพรโทคอล

คือรูปแบบและขั้นตอนการสื่อสารที่กำหนดขึ้นเพื่อใช้ในการรับส่งข้อมูลระหว่างตัวบริการเว็บและโปรแกรมค้นผ่านเว็บ การติดต่อสื่อสารเป็นแบบ สเตทเลส (stateless) และ อ็อปเจ็กออเร็นเต็ด โพรโทคอล (object-oriented protocol)

สเตทเลส คือ ตัวบริการเว็บไม่มีข้อมูลของโปรแกรมค้นผ่านเว็บที่ติดต่อเข้ามาเก็บไว้ วิธีการนี้มีข้อดีคือเมื่อตัวบริการเว็บต้องทำการเริ่มต้นทำงานใหม่ (restart process) โปรแกรมค้นผ่านเว็บเพียงเกิดความล่าช้าในการได้รับข้อมูลตามที่ร้องขอเท่านั้น

อ็อปเจ็ก ออเร็นเต็ด โพรโทคอล คือ การสื่อสารในแต่ละครั้ง เป็นการร้องขอบริการจากโปรแกรมค้นผ่านเว็บไปยังตัวบริการเว็บ และเมื่อได้รับข้อมูลที่ต้องการแล้ว ถือว่าเป็นการสิ้นสุดการติดต่อ

การสื่อสารเกิดขึ้นเมื่อโปรแกรมค้นผ่านเว็บ ส่งข้อความร้องขอบริการไปยังตัวบริการเว็บ โดยข้อความที่ส่ง จะต้องมีความสมบูรณ์ของข้อมูล เพื่อให้ตัวบริการเว็บมีข้อมูลเพียงพอในการให้บริการ เนื่องจากทั้งตัวบริการเว็บและโปรแกรมค้นผ่านเว็บไม่มีการติดต่อสื่อสารระหว่างกัน ก่อน

การส่งข้อความร้องขอบริการใดๆ และไม่มีการติดต่อจากเว็บโปรแกรมค้นผ่านเว็บเพื่อขอข้อมูลเพิ่มเติม ซึ่งมีทั้งหมด 4 ขั้นตอนตามที่ได้แสดงไว้ในรูปที่ 2.2

1) การสร้างเส้นทางการสื่อสาร

เมื่อตัวบริการเว็บเริ่มทำงานซ็อกเก็ตพอร์ต (socket port) หมายเลข 80 จะถูกจองไว้สำหรับรอรับคำขอบริการ เช่นเดียวกับการใช้โทรศัพท์ ก่อนที่จะสื่อสารกันได้จะต้องมีการหมุนหมายเลขปลายทางเพื่อสร้างเส้นทางการสื่อสารเสียก่อน สำหรับเฮชทีทีพี โพรโทคอล ก็เช่นเดียวกัน เมื่อโปรแกรมค้นผ่านเว็บต้องการส่งข้อความร้องขอบริการ จะต้องทำการติดต่อไปยังซ็อกเก็ตพอร์ตหมายเลข 80 เพื่อให้ ตัวบริการเว็บทำการสร้างเส้นทางการสื่อสารก่อนที่โปรแกรมค้นผ่านเว็บจะทำการส่งข้อความอื่นใดต่อไป

2) การส่งข้อความร้องขอบริการ

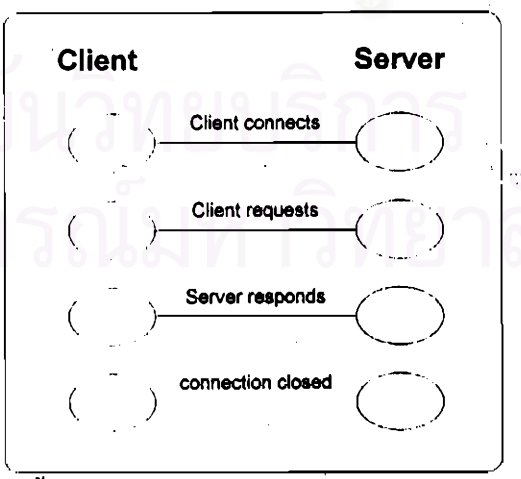
โปรแกรมค้นผ่านเว็บทำการส่งข้อความร้องขอบริการ ไปยังตัวบริการเว็บด้วยเส้นทางการสื่อสารที่ได้สร้างไว้แล้ว ในขั้นตอนที่ 1 ข้อความร้องขอบริการอาจมีมากกว่า 1 บรรทัด ชนิดของบริการที่ต้องการ และยูอาร์แอลจะอยู่ในบรรทัดแรกของข้อความที่ส่งไปยังตัวบริการเว็บ

3) การส่งผลลัพธ์กลับไปยังโปรแกรมค้นผ่านเว็บ

เมื่อตัวบริการเว็บได้รับคำร้องขอบริการจากโปรแกรมค้นผ่านเว็บ ตัวบริการเว็บจะทำการตอบกลับไปเสมอไม่ว่าจะทำงานตามที่ร้องขอสำเร็จหรือไม่ก็ตาม

4) การปิดเส้นทางการสื่อสาร

การปิดเส้นทางการสื่อสารสามารถปิดได้ทั้งจากโปรแกรมค้นผ่านเว็บ และ ตัวบริการเว็บ โดยที่ผลการทำงานสำเร็จ หรือ ไม่สำเร็จ ก็ได้



รูปที่ 2.1 ขั้นตอนการสื่อสารของเฮชทีทีพี โพรโทคอล

2.2 การพัฒนาซอฟต์แวร์ด้วยเทคโนโลยีเชิงวัตถุ

2.2.1 เทคโนโลยีเชิงวัตถุ (Object Oriented Technology)

ในการพัฒนาระบบงานขนาดใหญ่ ปัญหาที่มักจะเกิดขึ้นก็คือการสิ้นเปลืองกำลังคน และทรัพยากร ตลอดจนงบประมาณจำนวนมาก นอกจากนี้ยังมีความยากลำบากในการติดตามดูแลระบบให้ถูกต้องอยู่เสมอ และหากมีการพัฒนาระบบใหม่โดยมากแล้วเรามักจะไม่สามารถใช้ระบบงานที่เคยพัฒนาไว้ได้ ต้องทำการพัฒนาขึ้นใหม่ทั้งหมด เพื่อแก้ไขในลักษณะดังกล่าวจึงมีการคิดค้นวิธีการในการพัฒนาระบบขึ้นมาใหม่ เรียกว่า การพัฒนาซอฟต์แวร์แบบเทคโนโลยีเชิงวัตถุ (Object-oriented technology) ซึ่งทำให้การออกแบบและพัฒนาระบบสามารถทำได้รวดเร็วและมีข้อผิดพลาดน้อยลง

เทคโนโลยีเชิงวัตถุเป็นแนวคิดของการพัฒนาโปรแกรมหนึ่ง โดยผนึกเอาข้อมูลและโปรแกรมเข้าด้วยกันเป็นวัตถุ (object) และกำหนดตัวประสาน (interface) ระหว่างวัตถุต่างๆ โดยถ้าวัตถุหนึ่งต้องการใช้ข้อมูลในอีกวัตถุหนึ่งจะต้องติดต่อผ่านส่วนติดต่อนี้เท่านั้น จากหลักการนี้จะพบว่าการเปลี่ยนแปลงแก้ไขโปรแกรมและข้อมูลใด ในวัตถุหนึ่งจะมีผลกระทบต่ออีกวัตถุหนึ่งน้อยมาก ซึ่งจะทำให้การโปรแกรมทำได้สะดวกและรวดเร็วยิ่งขึ้น เป็นการลดต้นทุนในการบำรุงรักษาโปรแกรม นอกจากนี้ยังมีการถ่ายทอดคุณสมบัติจากวัตถุหนึ่งไปยังอีกวัตถุหนึ่งได้อีกด้วย ทำให้สามารถนำโปรแกรมที่มีอยู่เดิมกลับมาใช้ใหม่ได้ (reuseable) โดยผู้พัฒนาโปรแกรมไม่ต้องทราบเลยว่าโปรแกรมที่มีอยู่นั้นเขียนขึ้นมาอย่างไร และทำการเพิ่มคุณสมบัติอื่นๆ ที่ผู้พัฒนาโปรแกรมต้องการเข้าไป ดังนั้นการพัฒนาโปรแกรมใหม่ๆจะทำได้รวดเร็วขึ้น

2.2.2 แนวคิดในการนำซอฟต์แวร์กลับมาใช้ใหม่

จากการสังเกตุการพัฒนาซอฟต์แวร์ (Bertrand Meyer, 1988) พบว่าการทำงานของคนเขียนซอฟต์แวร์ต้องอาศัยซอฟต์แวร์เดิม ๆที่เคยทำไว้แล้ว นำกลับมาพัฒนาต่อเพื่อเป็นการประหยัดเวลา ซึ่งซอฟต์แวร์โดยทั่วไปมักจะหนีหลักการทั่วไปเหล่านี้ไม่พ้น เช่น การเรียงลำดับข้อมูล (Sort) การค้นหา (Search) การอ่าน และเขียนแฟ้มข้อมูล การเปรียบเทียบ เป็นต้น แต่การนำซอฟต์แวร์เก่าๆ มาใช้ใหม่ไม่ง่ายอย่างที่คิด สาเหตุก็เพราะการพัฒนาซอฟต์แวร์ของแต่ละคนมีรูปแบบไม่เหมือนกัน อีกทั้งเวลาที่ผ่านไปทำให้ไม่สามารถจดจำการทำงานของซอฟต์แวร์เก่าๆ เหล่านั้นได้ ยิ่งถ้าในการพัฒนาที่ไม่เป็นระบบก็ยิ่งทำให้ไม่สามารถนำมาใช้ได้อีก ต้องพัฒนาใหม่ทั้งหมด

เพื่อแก้ไขการนำซอฟต์แวร์กลับมาใช้ใหม่ จึงมีการพัฒนาหลักการต่างๆ มาใช้ดังนี้

1) ด้านบุคลากร โดยมีการสับเปลี่ยนหมุนเวียนบุคลากร เพื่อให้เกิดการเรียนรู้การทำงานของระบบต่างๆ เมื่อมีบุคลากรออกไป การพัฒนาก็สามารถทำต่อได้ และสามารถติดตามซอฟต์แวร์ที่เคยพัฒนานำกลับมาใช้ได้ สาเหตุที่ต้องทำเช่นนี้ก็เพราะอัตราการเปลี่ยนงานด้านคอมพิวเตอร์มีสูง

2) ด้านการออกแบบระบบ โดยมีการออกแบบระบบที่ดี และใช้การโปรแกรมแบบโครงสร้าง (Structure Programming)

3) ด้านตัวโปรแกรม (Source code) โดยมีการพิมพ์ตัวโปรแกรมไว้ และมีการจัดทำคู่มือบันทึกการส่งผ่านค่าและตัวแปรต่างๆ อย่างละเอียด

นอกจากกรณีดังกล่าว การพัฒนาซอฟต์แวร์เชิงวัตถุก็เป็นแนวทางหนึ่งในการนำซอฟต์แวร์กลับมาใช้ใหม่ และยังทำให้เกิดความคล่องตัว และความรวดเร็วในการพัฒนาซอฟต์แวร์ด้วย เพราะในการพัฒนาซอฟต์แวร์เชิงวัตถุมีการสร้างวัตถุให้มีความสามารถในการนำไปเชื่อมต่อกับวัตถุอื่นหรือวัตถุของระบบอื่นได้ ไม่ว่าวัตถุนั้นจะถูกดัดแปลงแก้ไขให้เป็นวัตถุใหม่ใดๆ วัตถุแต่ละอันก็ยังคงนำมาเชื่อมต่อได้ นั่นคือวัตถุสามารถนำกลับมาใช้ใหม่ได้ เช่นมีการสร้างวัตถุที่ช่วยคำนวณภาษีชื่อ TAX ซึ่งประกอบด้วยตารางการเสียภาษี การคิดภาษี เราสามารถนำวัตถุนี้ไปเชื่อมต่อกับระบบบัญชีเงินเดือนที่ประกอบด้วยวัตถุอื่นๆ อีก การคำนวณภาษีจากระบบเงินเดือน ทำโดยการส่งข้อความ (Message) ที่อาจประกอบด้วยอาร์กิวเมนต์ (Argument) เช่น สถานภาพสมรส จำนวนบุตรไปยังวัตถุที่ชื่อ TAX วัตถุนี้จะให้คำตอบออกมา และถ้าบริษัทอื่นต้องการใช้วัตถุที่ชื่อ TAX นี้ก็สามารถนำไปเชื่อมต่อกับระบบบัญชี ได้เลยโดยไม่ต้องคำนึงว่าระบบบัญชีนั้นมีการออกแบบอย่างไร แต่ต้องเป็นการใช้การพัฒนาเชิงวัตถุเหมือนกันเท่านั้นเอง

2.2.3 ขั้นตอนในการพัฒนาโปรแกรมเชิงวัตถุ

ในการพัฒนาโปรแกรมเชิงวัตถุได้มีผู้นำเสนอแนวทางในการทำงานหลายแนวทาง สำหรับในการวิจัยครั้งนี้ได้ดำเนินการตามหลักการของเกรดี บูช ซึ่งประกอบด้วย 3 ขั้นตอนดังนี้

1) วิเคราะห์ความต้องการ (Requirement Analysis) เป็นขั้นตอนในการพิจารณาว่าลูกค้าต้องการให้ระบบทำอะไร โดยการระบุถึงฟังก์ชันหลัก ที่ระบบสามารถทำงานได้ กำหนดขอบเขตของระบบ และจัดทำเป็นเอกสารเกี่ยวกับ การปฏิบัติงานหลัก และ นโยบายขององค์กร ที่ระบบจะต้องสนับสนุนการทำงานนั้นๆ ในการทำงาน โมเดลยูสเคส เป็น เครื่องมือหนึ่งที่สามารถนำมาใช้ช่วยในการวิเคราะห์ความต้องการของระบบได้

2) วิเคราะห์และกำหนดขอบเขตของปัญหา (Domain analysis) เป็นกระบวนการในการกำหนดให้ชัดเจน และแน่นนอน ถึงความสัมพันธ์ของระบบ โดยระบุวัตถุหลัก รวมถึงข้อมูล และขั้นตอนการปฏิบัติงาน โดยมีขั้นตอนในการปฏิบัติงานดังนี้

2.1) กำหนดคลาส คือโครงสร้างของข้อมูลที่ประกอบไปด้วยลักษณะประจำ และบริการ

2.2) กำหนดความสัมพันธ์ของคลาสต่างๆ โดยแสดงการสืบทอด และการเรียกใช้งานของคลาสต่างๆ

2.3) กำหนดการปฏิบัติงาน คือการกำหนดว่าจะต้องมีฟังก์ชันอะไรบ้าง และมีขั้นตอนในการปฏิบัติงานอย่างไร

2.4) ทบทวน และปรับปรุงโมเดล รวมถึง กระบวนการ ที่ได้กำหนดขึ้น

3) ออกแบบระบบ (System design) เป็นกระบวนการที่มุ่งเน้นไปที่การพิจารณาว่าทำอย่างไรจะสามารถติดตั้งระบบให้ได้ตามความต้องการที่ได้ศึกษาและวิเคราะห์ไว้แล้ว โดยให้คุ้มกับต้นทุนที่ต้องเสียไปมากที่สุด ซึ่งไม่มีขั้นตอนและวิธีการที่แน่นอนในการได้มาซึ่งการออกแบบระบบที่ดี สำหรับขั้นตอนการออกแบบที่กล่าวไว้นี้เป็นเพียงแนวทางหนึ่งในการออกแบบระบบเท่านั้น

3.1) กำหนดสถาปัตยกรรมในการพัฒนาระบบ คือการกำหนดโครงสร้างของโปรแกรม โครงสร้างที่ดีคือการกำหนดเป็นระดับชั้นของโปรแกรม โดยการกำหนดให้โปรแกรมที่เรียกใช้งานโปรแกรมในระดับชั้นอื่นไม่จำเป็นต้องรู้จักกับการทำงานภายในระดับชั้นที่ต้องการเรียกใช้ เป็นต้น

3.2) วางแผนการพัฒนาฟังก์ชัน คือกำหนดเวลาที่แต่ละฟังก์ชันจะต้องพัฒนาเสร็จ

3.3) พัฒนาคลาส รายละเอียดของการปฏิบัติงาน และ ข้อจำกัดในการเข้าถึงวัตถุ

2.2.4 โมเดลยูสเคส

โมเดลยูสเคสพัฒนาโดย อิวา จาคอบสัน และมีการรวมเข้าไปในหลายๆ เมทโธโดโลยี (Methodology) เพื่อใช้ในการระบุความต้องการของผู้ใช้ โมเดลนี้มุ่งเน้นไปที่มุมมองของผู้ใช้ ที่ติดต่อกับระบบ โดยใช้ข้อความและรูปภาพในการแสดงการติดต่อระหว่างผู้ใช้กับระบบ

โมเดลยูสเคสใช้แอคเตอร์ (Actor) นิยามเอนทิตี้ภายนอกที่ติดต่อกับระบบ และยูสเคส (Use case) ในการนิยามฟังก์ชันการทำงานของระบบ

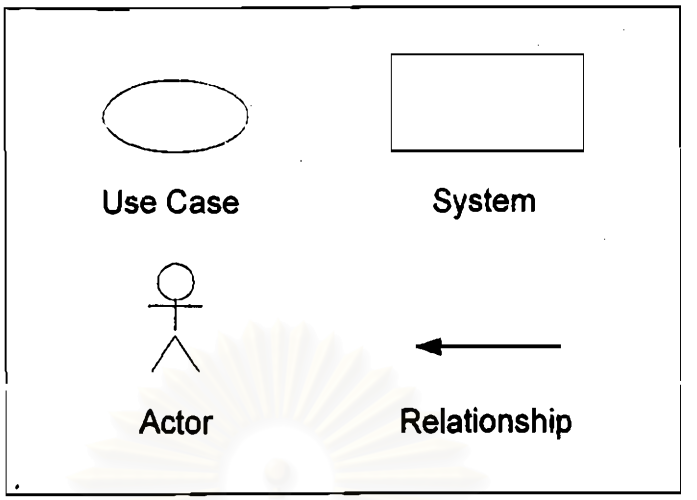
โมเดลยูสเคสประกอบด้วย

1) ระบบ (System) แสดงโดยรูปสี่เหลี่ยม ซึ่งแสดงถึงขอบเขตระหว่างแอคเตอร์ภายนอกระบบกับฟังก์ชันการทำงานภายในระบบ

2) ยูสเคส (Use case) แสดงโดยรูปวงรีโดยภายในวงรีจะระบุชื่อฟังก์ชันที่กระทำโดยระบบตามมุมมองของผู้ใช้

3) แอคเตอร์ (Actor) แสดงถึงบุคคล ระบบหรืออุปกรณ์ที่ต้องการแลกเปลี่ยนสารสนเทศกับระบบแอคเตอร์จะแสดงถึงบทบาท (Role) ที่บุคคล หรือระบบได้รับมากกว่าที่จะแสดงถึงบุคคล (Person) ซึ่งบุคคลคนหนึ่งอาจมีได้หลายบทบาท เช่น สมศรีเป็นพนักงานในบริษัทขนส่งมีบทบาทหลายบทบาท คือ ผู้ส่งของ ผู้รับของ ผู้ดูแลคลังสินค้า หรือผู้ต้องขอรายงาน เป็นต้น

4) ความสัมพันธ์ (Relationship) แสดงโดยเส้นตรงที่ลากระหว่างแอคเตอร์กับยูสเคส หรือระหว่างยูสเคสกับยูสเคสเพื่อแสดงถึงความสัมพันธ์ระหว่างแอคเตอร์กับยูสเคส



รูปที่ 2.2 สัญลักษณ์ที่ใช้ในโมเดลยูสเคส

2.2.5 ภาษาการโมเดลแบบยูนิฟาย

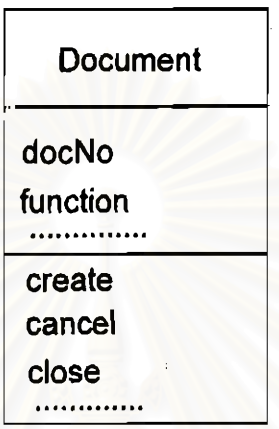
ภาษาการโมเดลแบบยูนิฟายเป็นภาษาการโมเดลเชิงวัตถุยุคที่ 3 ซึ่งดัดแปลงและเพิ่มเติมจากการโมเดลเชิงวัตถุของเกรดี บูช (Grady Booch) จิม แรมเบริก (Jim Rumbaugh) และ อิวา จาคอบสัน (Ivar Jacobson) [Booch94 OMT91 OOSE92] ในปัจจุบันการโมเดลแบบยูนิฟายได้ถูกเสนอเข้าไปในกลุ่มการจัดการวัตถุ (Object Management Group) เพื่อให้กลายเป็นภาษาการโมเดลมาตรฐานสำหรับการพัฒนาเชิงวัตถุ เพราะว่าการโมเดลเชิงวัตถุแบบยูนิฟายสามารถประยุกต์ใช้ในการโมเดลระบบทุกรูปแบบ ได้แก่ ระบบเวลาจริง (real-time systems) ระบบรับ-ให้บริการ (client/server system) และระบบซอฟต์แวร์มาตรฐานชนิดอื่นๆ การโมเดลแบบยูนิฟายเป็นขอบเขตพื้นฐานของหลักการโมเดลด้วยสัญลักษณ์ (notation) ที่เป็นรูปแบบ แต่โดยตัวมันเองไม่ใช่วิธีการ (method) เพราะไม่มีการระบุในเรื่องกระบวนการทำงาน (process) การโมเดลแบบยูนิฟายใช้การวิเคราะห์และออกแบบเชิงวัตถุเพื่อสร้างโมเดลระบบเชิงวัตถุ การวิเคราะห์และออกแบบระบบสนับสนุนการจัดการด้านการจัดส่งเครื่องโทรศัพท์ส่วนบุคคลนี้ ใช้การโมเดลแบบยูนิฟาย และ ใช้กระบวนการทำงานตามการวิเคราะห์และออกแบบระบบงานเชิงวัตถุตามหลักการของเกรดี บูช

1) แผนภาพคลาส

แผนภาพคลาสเป็นแผนภาพหลักในการโมเดลเชิงวัตถุ แผนภาพคลาสแสดงให้เห็นนามธรรม (abstraction) ที่สำคัญในระบบและความสัมพันธ์ของนามธรรมเหล่านั้น ส่วนประกอบหลักที่พบในแผนภาพคลาสได้แก่สัญรูปคลาส (class icon) และสัญรูปความสัมพันธ์ (relationship icon)

1.1) คลาส ลักษณะประจำและบริการ

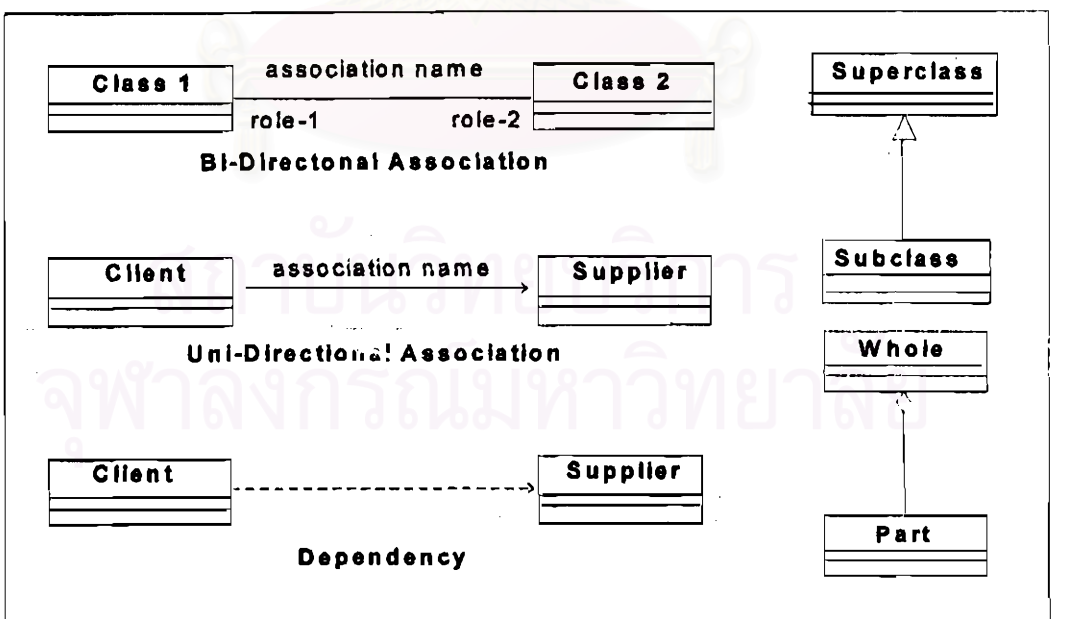
คลาสแต่ละคลาสถูกแสดงในโมเดลเชิงวัตถุด้วยรูปสี่เหลี่ยมที่ประกอบด้วยส่วนหนึ่ง สอง หรือสามส่วนดังรูปที่ 2.3 แสดงตัวอย่างของคลาส ส่วนประกอบแรกแสดงชื่อของคลาสซึ่งจำเป็นต้องมี ส่วนที่สองและสามแสดงลักษณะประจำและบริการของคลาสตามลำดับซึ่งอาจมีหรือไม่ก็ได้



รูปที่ 2.3 สัญรูปคลาสกับส่วนประกอบลักษณะประจำและส่วนประกอบบริการ

1.2) ความสัมพันธ์

คลาสในระบบจะมีความสัมพันธ์กัน ซึ่งความสัมพันธ์เหล่านี้มีหลายรูปแบบ ดังแสดงในรูปที่ 2.4

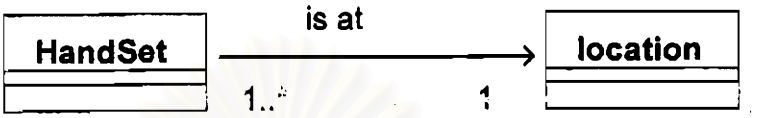


รูปที่ 2.4 รูปแบบความสัมพันธ์

รูปสี่เหลี่ยมแสดงคลาส เส้นระหว่างคลาสแสดงถึงรูปแบบความสัมพันธ์ต่างๆ ระหว่างคลาส ความสัมพันธ์เหล่านี้ได้แก่

1.2.1) แอซโซซิเอชั่น (Associations)

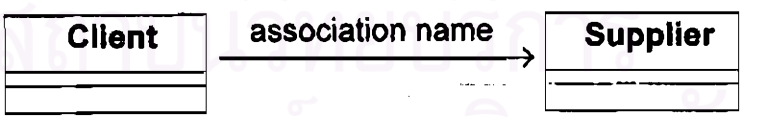
แอซโซซิเอชั่นใช้เพื่อแสดงโครงร่างการขึ้นต่อกันระหว่างวัตถุของคลาสที่ต่างกัน โดยข้อความบนเส้นใช้อธิบายความสัมพันธ์ เช่น คลาสแฮนด์เซ็ทและคลาสโลเคชั่น มีความสัมพันธ์กันแบบแอซโซซิเอชั่น โดยข้อความ is at แสดงให้เห็นความสัมพันธ์ว่า แฮนด์เซ็ทอยู่ที่โลเคชั่น



รูปที่ 2.5 ตัวอย่างความสัมพันธ์แบบแอซโซซิเอชั่น

การระบุจำนวนอินสตันที่มีส่วนร่วมในความสัมพันธ์ ซึ่งเรียกว่า มัลติพลิซิตี้ (multiplicity) ในการโมเดลเชิงวัตถุจะใช้การระบุตัวเลขที่จุดสิ้นสุดของเส้นที่แสดงความสัมพันธ์ เช่นจากรูปที่ 2.5 แฮนด์เซ็ท 1 เครื่องจะอยู่ที่โลเคชั่นหนึ่ง และโลเคชั่นหนึ่งมีหลายแฮนด์เซ็ท

โดยทั่วไปแอซโซซิเอชั่นจะเป็นความสัมพันธ์แบบสองทิศทาง (bidirectional) หมายความว่าอินสตันท์ของคลาสหนึ่งสามารถนำทาง (navigate) ไปหาอินสตันท์ของอีกคลาสหนึ่ง และในทำนองเดียวกัน อินสตันท์ของคลาสที่ถูกนำทางก็สามารถนำทางไปหาคลาสที่ทำการนำทางมันได้ นอกจากนี้แอซโซซิเอชั่นยังมีความสัมพันธ์แบบทิศทางเดียว (unidirectional) เพื่อแสดงความสัมพันธ์ในลักษณะผู้ขอบริการและผู้ให้บริการ (client-supplier relationship) โดยใช้สัญลักษณ์หัวลูกศรที่คลาสผู้ให้บริการ ดังเช่นรูปที่ 2.6 แสดงตัวอย่างความสัมพันธ์ระหว่างคลาสลูกค้า (client) และคลาสผู้จำหน่าย (supplier) กล่าวคือคลาสลูกค้ามีการขอบริการบางอย่างจากคลาสผู้จำหน่าย



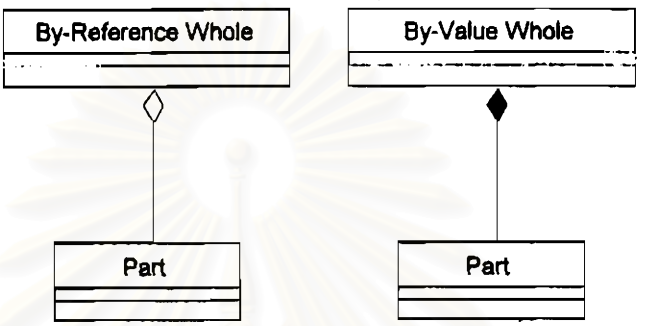
รูปที่ 2.6 ตัวอย่างความสัมพันธ์แบบแอซโซซิเอชั่นแบบทิศทางเดียว

1.2.2) แอกริเกชั่น (Aggreation)

แอกริเกชั่นเป็นรูปแบบพิเศษของแอซโซซิเอชั่นที่ซึ่งถูกใช้เพื่อแสดงความสัมพันธ์แบบส่วนประกอบ มีความหมายเหมือนกับความสัมพันธ์แบบโครงร่างทุกส่วน เช่น รถยนต์ประกอบด้วยล้อและเครื่องยนต์ ความสัมพันธ์แบบแอกริเกชั่นแสดงโดยใช้รูปสี่เหลี่ยมขนมเปียกปูนบนเส้นความสัมพันธ์

แอกริเกชั่นแสดงให้เห็นว่าช่วงชีวิต (lifetime) ของวัตถุส่วนประกอบขึ้นอยู่กับวัตถุแม่ กล่าวคือวัตถุส่วนประกอบไม่สามารถถูกสร้างได้จนกว่าวัตถุแม่จะถูกสร้าง ในทำนองเดียวกันวัตถุส่วนประกอบไม่สามารถถูกทำลายโดยวัตถุอื่นยกเว้นวัตถุแม่ของมัน

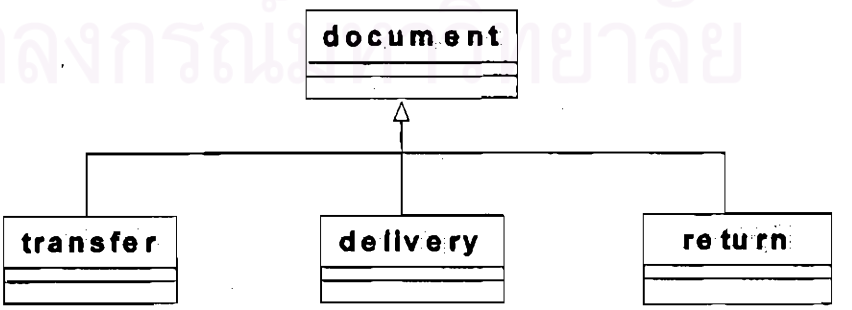
รูปแบบของแอกริเกชั่นมีอยู่ 2 รูปแบบคือส่วนประกอบแบบอ้างอิง และส่วนประกอบโดยค่า ส่วนประกอบโดยค่าหมายถึงวัตถุแม่มีการกำหนดอินสตันท์ของวัตถุส่วนประกอบที่แน่นอนภายในตัวมันเอง ดังรูปที่ 2.7 แสดงสัญลักษณ์ของแอกริเกชั่นทั้ง 2 รูปแบบ



รูปที่ 2.7 รูปแบบของแอกริเกชั่น

1.2.3) การถ่ายทอด (Inheritance)

ความสัมพันธ์แบบการถ่ายทอดถูกใช้เมื่อคลาสหนึ่งคลาสมีส่วนร่วมของโครงร่างและลักษณะประจำกับคลาสอื่น ในโมเดลเชิงวัตถุเรียกคลาสที่มีคุณสมบัติทั่วไปว่าซุปเปอร์คลาส (superclass) และเรียกคลาสที่มีการถ่ายทอดคุณสมบัติเพิ่มเติมว่าซับคลาส (subclass) ซับคลาสมีคุณสมบัติและบริการเหมือนกับซุปเปอร์คลาส โดยที่ซับคลาสอาจมีลักษณะประจำและบริการเพิ่มเติม รูปที่ 2.8 แสดงให้เห็นตัวอย่างการถ่ายทอดคลาส คือคลาสเอกสารการโอน (transfer)คลาสเอกสารการจัดส่ง (delivery) และคลาสเอกสารการคืน (return) คือคลาสเอกสารย่อยที่ถ่ายทอดจากคลาสเอกสาร (document) หรือกล่าวอีกนัยหนึ่งว่า คลาสเอกสารการโอน คลาสเอกสารการจัดส่ง และคลาสเอกสารการคืนเป็นซับคลาสของคลาสเอกสาร



รูปที่ 2.8 ตัวอย่างแผนภาพแสดงการถ่ายทอด

1.2.4) การขึ้นต่อกัน (Dependency)

ความสัมพันธ์แบบแอซโซซิเอชั่นและการถ่ายทอดจะมีผลกระทบต่อโครงร่างของวัตถุที่สร้างจากคลาสที่มีความสัมพันธ์ แต่ในบางครั้งคลาสมีความสัมพันธ์กันในแง่ที่ต้องการเรียกใช้บริการของอีกคลาสหนึ่ง ในการโมเดลเชิงวัตถุเรียกความสัมพันธ์แบบนี้ว่าการขึ้นต่อกัน กล่าวคือคลาสของผู้ขอบริการขึ้นอยู่กับบริการของคลาสของผู้ให้บริการ แต่ไม่มีการขึ้นต่อกันภายในโครงสร้างของคลาส รูปแบบทั่วไปของความสัมพันธ์แบบขึ้นต่อกันจะพบเมื่อบริการของผู้ขอบริการมีการรับอาร์กิวเมนต์ที่มีรูปแบบเป็นคลาสของผู้ให้บริการ

2) แผนภาพซินแนริโอ (Scenarios diagram)

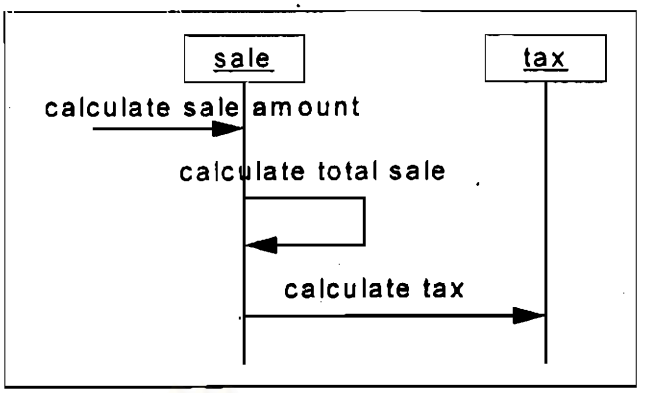
คือแผนภาพที่ใช้แสดงลำดับการติดต่อกันระหว่างวัตถุ เพื่อให้ระบบสารสนเทศทำงานได้เป็นผลสำเร็จ นั่นคือการแสดงความสัมพันธ์ของคลาสหนึ่งกับบริการอื่นๆ จากคลาสเดียวกันหรือต่างคลาสกัน แผนภาพซินแนริโอมีค่าเท่ากับรูทีนย่อยหรือโพรซีเดอร์ในภาษาโปรแกรมอื่นๆ หลักการของแผนภาพซินแนริโอประกอบด้วย 3 ส่วนคือ

- คลาสและบริการผู้ส่ง (the sender class and service)
- คลาสและบริการผู้รับ (the receiver class and service)
- พารามิเตอร์และจำนวนบริการที่ร้องขอ

ในแผนภาพซินแนริโอประกอบด้วยคลาสและบริการผู้ส่ง และ คลาสและบริการผู้รับ ส่วนพารามิเตอร์เป็นเพียงทางเลือก มีหรือไม่ก็ได้

เหตุผลของการใช้แผนภาพซินแนริโอคือ

- เพื่อหาวัตถุเพิ่มเติม
- เพื่อกระจายและขัดเกลาการตอบสนองให้ดีขึ้น (รูปแบบพื้นฐานสำหรับการตอบสนอง (Object responsibility) มี 3 รูปแบบคือลักษณะประจำของวัตถุ การชื่อมโยงวัตถุและบริการ
- เพื่อให้เข้าใจระบบดีขึ้น
- เพื่อประเมินความผิดพลาดของโมเดล
- เพื่อทดสอบโครงสร้างเชิงวัตถุด้วยตนเองว่าจะได้ผลตามที่ต้องการหรือไม่



รูปที่ 2.9 ตัวอย่างแผนภาพลำดับเหตุการณ์

การอ่านแผนภาพ

- วัตถุการขาย (sale) ถูกเรียกให้ทำการคำนวณผลรวมการขาย
- บริการคำนวณผลรวมการขาย ก่อให้เกิดบริการ คำนวณยอดขาย (calculate total sale)
- บริการคำนวณผลรวมการขายส่งคำร้องไปให้วัตถุภาษีทำบริการคำนวณภาษี (calculate tax)

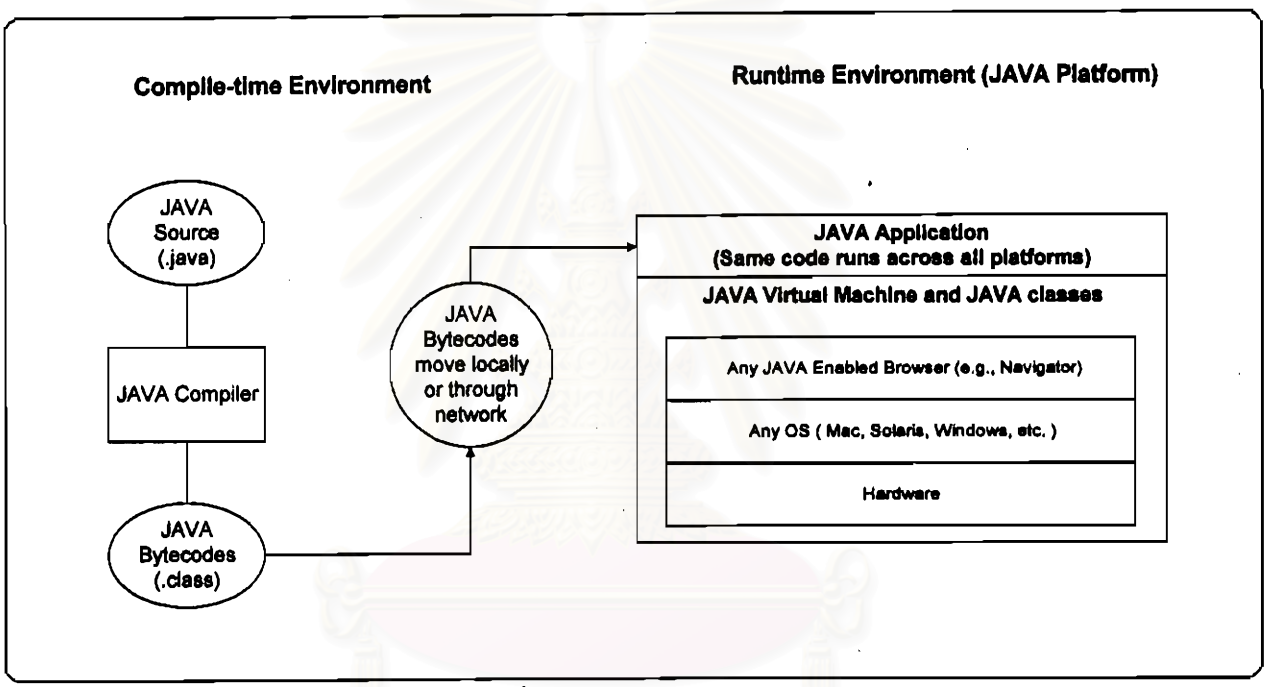
2.3 ภาษาจาวา

บริษัท Sun Micro System Co.,Ltd. เป็นผู้คิดค้น และ พัฒนาจาวาโดยต้องการให้จาวาเป็นแพล็ตฟอร์ม ใหม่ในการพัฒนาโปรแกรม กล่าวคือ ได้มีการออกแบบให้โปรแกรมที่พัฒนาด้วยภาษาจาวาสามารถทำงานอยู่บนฮาร์ดแวร์และระบบปฏิบัติการใดๆ ก็ได้

ภาษาที่ใช้ในการพัฒนาโปรแกรมในขณะนี้ เมื่อโปรแกรมผ่านการแปลจะได้เป็นไบนารีโค๊ด (binary code) ซึ่งสามารถทำงานได้บน ฮาร์ดแวร์และระบบปฏิบัติการที่ทำการแปลโปรแกรมเท่านั้น ซึ่งมีวลีที่ใช้เรียกโดยทั่วไปว่าแพล็ตฟอร์มสเปคซิฟิค (Platform specific) หากต้องการนำโปรแกรมไปทำงานบนระบบปฏิบัติการอื่น จะต้องทำการแปลใหม่ บนระบบการนั้นๆ เพื่อให้ได้ผลลัพธ์เป็นไบนารีโค๊ด สำหรับฮาร์ดแวร์และระบบปฏิบัติการนั้นๆ เสียก่อน โปรแกรมจึงจะสามารถใช้ทำงานได้ ตัวอย่างเช่น โปรแกรมไมโครซอฟ์เวิดล์ ผู้ใช้ไม่สามารถที่จะนำไบนารีโค๊ดที่ทำงานอยู่บนระบบปฏิบัติการเอ็นทีไปทำงานบนระบบปฏิบัติการดอสหรือ แมคอินทอชได้

สำหรับภาษาจาวา ลักษณะการเขียนคล้ายคลึงกับภาษา C++ ซึ่งอยู่ในรูปแบบอ็อปเจ็ก ออเร็นเต็ด โปรแกรมมิ่ง (Object Oriented Programming) เป็นภาษาที่ออกแบบให้ไม่ขึ้นกับสถาปัตยกรรมของฮาร์ดแวร์ คือไบนารีโค๊ดที่ได้หลังจากการแปลโปรแกรม มีรูปแบบเป็นมาตรฐานเดียวกันหมด ไม่ว่าจะทำการแปลโปรแกรมบนระบบปฏิบัติการ หรือฮาร์ดแวร์ใดๆ ซึ่งบริษัท Sun Micro System Co., Ltd. เรียกไบนารีโค๊ดนี้ว่า ไบท์โค๊ด (bytecode)

อีกจุดเด่นหนึ่ง คือความสามารถในการสร้างโปรแกรมขนาดเล็ก ที่เรียกว่าแอ็พเพล็ต เพื่อใช้งานผ่านเครือข่ายโดยอาศัยการทำงานในรูปแบบของระบบรับ-ให้บริการ ซึ่งประกอบไปด้วยตัวบริการเว็บและโปรแกรมค้นผ่านเว็บ โดยที่โปรแกรมค้นผ่านเว็บทำหน้าที่ร้องขอแอ็พเพล็ตที่ต้องการไปยังตัวบริการเว็บ เพื่อให้ทำการบรรจุโปรแกรมผ่านเครือข่ายมายังโปรแกรมค้นผ่านเว็บ โดยอาศัย เฮชทีทีพี โพรโทคอล เมื่อโปรแกรมค้นผ่านเว็บได้รับแอ็พเพล็ต โปรแกรมค้นผ่านเว็บจะทำหน้าที่เสมือนเป็นจาวาเวอร์ชัวแมชชีน ซึ่งทำงานกับแอ็พเพล็ตในรูปแบบของอินเตอร์พลีตเตอร์ (Interpreter) ดังรูปที่ 2.10 สถาปัตยกรรมจาวา



รูปที่ 2.10 สถาปัตยกรรมจาวา