ระบบเอเยนต์ในอินเตอร์เน็ต

INTERNET AGENTS

โดย

นางสาวธีรา ทานตวณิช รหัสประจำตัว 36014199

นางสาวอารีวรรณ เหมทานนท์ รหัสประจำตัว 36014566

อาจารย์ที่ปรึกษา

อาจารย์อภิเนตร อูนากูล

วัน เดือน ปี -1 ต.ค. 2541
เลขทะเบียน 038310
เลขเรียกหนังสือ T 54330 ธ๑๙๙

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2539

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2539

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง... ระบบเอเยนต์ในอินเตอร์เน็ต

INTERNET AGENTS

ผู้จัดทำ...

1. นางสาว ธีรา ทานตวณิช รหัสประจำตัว 36014199
2. นางสาว อารีวรรณ เหมทานนท์ รหัสประจำตัว 36014566

อาจารย์ที่ปรึกษา

(อาจารย์อภิเนตร อูนากูล)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียด	หน้าที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบเอเยนต์ในอินเตอร์เน็ต

ธีรา ทานตวณิช รหัส 36014199
อารีวรรณ เหมทานนท์ รหัส 36014566
อาจารย์อภิเนตร อูนากูล อาจารย์ที่ปรึกษา
ปีการศึกษา 2539

บทคัดย่อ

ระบบเอเยนต์ทางอินเทอร์เน็ตเป็นโปรแกรมที่ทำงานให้สำเร็จตามเป้าหมายในนามของผู้ใช้งานมันและเพื่อประโยชน์ของผู้ใช้งานนั้น โดยผู้ใช้งานเพียงแค่บอกจุดหมายของงาน ไม่ต้องกำหนดรายละเอียดในการทำงานนั้นให้ชัดเจนแน่นอน โปรแกรมผู้ช่วยหรือเอเยนต์ก็จะทำงานโดยพิจารณาตัดสินใจว่าจะทำงานนั้น "อย่างไร" และ "เมื่อไร" โปรแกรมเอเยนต์นี้จะทำงานบนสภาวะแวดล้อมของมันซึ่งเป็นสภาวะแวดล้อมที่ซับซ้อน ซึ่งทั้งตัวโปรแกรมและสภาวะแวดล้อมของมันต่างก็เป็นระบบในเทคโนโลยีที่เน้นการทำงานแบบเอเยนต์ เราได้ทำการศึกษาระบบและเทคโนโลยีนี้ และได้รวบรวมความรู้เป็นเอกสารรวบรวมความรู้เกี่ยวกับเทคโนโลยีเอเยนต์และโปรแกรมแสดงการทำงานแบบเอเยนต์ นอกจากนี้ เรายังได้สร้างระบบเอเยนต์ขึ้นมา 1 ระบบ ซึ่งเป็นระบบที่มีโปรแกรมเอเยนต์ทำหน้าที่ช่วยในการค้นหาไฟล์ที่เครื่องเซฟเวอร์ (Server) โดยผู้ใช้งานเรียกใช้โปรแกรมนี้ทางเวบเบราเซอร์ (Web Browser)

เครื่องเซฟเวอร์ (Server) ของระบบมีวินโดว์เอ็นที (Windows NT) เป็นระบบปฏิบัติการ ส่วนเครื่องไคเอนต์ (Client) นั้นใช้วินโดว์ 95 (Windows 95) ต่อเชื่อมกันโดยใช้โปรโตคอล (Protocol) TCP/IP ระบบถูกเขียนด้วยภาษาจาวา (Java) และใช้ Microsoft Visual J++ เป็นตัว Compiler

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

INTERNET AGENTS

THEERA THANTAWANIT 36014199
AREEWAN HAMTANON 36014566
APINETR UNAKUL ADVISOR
1996

ABSTRACT

A software agent is a piece of software which acts to accomplish tasks on behalf of its user. Many agents are based on the idea that the user need only specify a high-level goal instead of issuing explicit instructions, leaving the "how" and "when" decisions to the agent. Agent-oriented technology is one kind of distributed-object application which inhibits a complex, dynamic environment, this technology will be studied. The aim of this thesis is divided into two parts : the system study and the software implementation. The delivables will be from both parts which are the lecture note about the complete knowledge and the sotfware demonstrating on how agent works.

The demonstrating agent system is a FileFinder agent which searches for user-defined files in the server. The user can gt the agents rub by loading agent from users's web browser. The whole system is wtitten in Java with Microsoft's J++ compiler , the server is implemented in Windows NT and the client is implemented in Windows 95. Both are connected together using TCP/IP protocol.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมา

นับได้ว่าทุกวันนี้ เครือข่ายอินเตอร์เน็ตได้เข้ามามีบทบาทสำคัญต่อการดำเนินชีวิตของมนุษย์เป็นอย่างมาก ทั้งนี้เพราะว่าเป็นแหล่งของข้อมูลต่างๆ มากมาย ซึ่งมีทั้งที่มาจากองค์กรที่ไม่แสวงหาผลกำไร ได้แก่ หน่วยงานภาครัฐและเอกชน ผู้เชี่ยวชาญ สถาบันการศึกษาต่าง ๆ เป็นต้น และข้อมูลส่วนที่ได้รับมาจากองค์กรที่แสวงหาผลกำไร ได้แก่ บริษัท ห้างร้าน ซึ่งมักอยู่ในรูปของการดำเนินธุรกิจ ไม่ว่าจะเป็นธุรกิจการค้า การบริการและสื่อโฆษณา ต่างหันมาให้ความสนใจเพื่อลงทุนผ่านทางอินเตอร์เน็ตด้วยกันทั้งสิ้น

ด้วยเหตุนี้เองทำให้เกิด แหล่งข้อมูลขนาดใหญ่ขึ้น ดังนั้น การเผยแพร่ข้อมูลข่าวสาร การศึกษาค้นคว้า การแลกเปลี่ยนหรือเสนอแนะความคิดเห็น การดำเนินธุรกิจค้าขาย สามารถดำเนินการได้ง่ายขึ้น แต่ทั้งนี้ในการค้นหาข้อมูลที่เราต้องการในแต่ละครั้งนั้น บางครั้งกว่าที่เราจะเจอข้อมูลในส่วนที่เราต้องการจริง ๆ เราก็จะต้องเสียเวลาค้นหา และต้องเลือกข้อมูลนั้นอีกเป็นเวลานานเลยทีเดียว

แนวความคิดในเรื่องของ เทคโนโลยีเอเยนต์ (Agent Technology) ถูกพัฒนาขึ้นมาเพื่ออำนวยความสะดวกแก่ผู้ใช้งาน เกี่ยวกับการทำหน้าที่หรือปฏิบัติงานแทนบุคคลที่เป็นผู้ใช้งาน หรือเป็นเจ้าของงาน หรือเป็นผู้มีอำนาจออกคำสั่ง ให้เอเยนต์ทำงานตามคำสั่งที่ได้รับมอบหมายนั้นให้สำเร็จลุล่วงไปโดยที่ผู้ใช้ไม่ต้องลงมือกระทำเอง ตัวอย่างเช่น ให้เอเยนต์ค้นหาข้อมูลแทนเราในยามค่ำคืน และเมื่อตื่นขึ้นมาตอนเช้าเอเยนต์จะแสดงข้อมูลที่เราต้องการขึ้นบนหน้าจอคอมพิวเตอร์ และหากเป็นเอเยนต์ ที่มีระบบซับซ้อนมากขึ้น ก็จะสามารถใช้ Statistical Mapping ในการค้นหาเอกสาร เพลง หรือ ภาพยนตร์ ได้เป็นอย่างดี และในอนาคต เราอาจใช้ให้เอเยนต์ ช่วยซื้อของที่เราต้องการแทนเรา ผูกมิตรกับคนอื่นๆ หรือแม้แต่ให้เอเยนต์ คอยเฝ้าจับตาดูคู่แข่งของเรา

1.2 ประโยชน์ที่ได้รับ

1.2.1) ในแง่ของงานวิจัย

1. ศึกษาความเป็นมาและรายละเอียดต่าง ๆ ที่เกี่ยวข้องกับเทคโนโลยีเอเยนต์
2. สามารถระบุความจำเป็น รวมทั้งประโยชน์ที่ได้รับจากนำเทคโนโลยีเอเยนต์มาใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. สามารถวิเคราะห์แนวโน้มของเทคโนโลยีใหม่ ๆ ที่กำลังจะเกิดขึ้นมาในอนาคตเพื่อมารองรับการทำงานของเทคโนโลยีเอเยนต์ ให้มีประสิทธิภาพสูงขึ้น

4. เทคโนโลยีเอเยนต์ที่มีการพัฒนาขึ้นมานี้ สามารถแก้ปัญหาในเรื่องของการกลั่นกรองข้อมูลให้ตรงกับความต้องการของผู้ใช้ได้เป็นอย่างดี เพราะมีความสามารถเหนือกว่าการค้นหาข้อมูลแบบเดิมที่มีการใช้งานกันอยู่ในอินเตอร์เน็ต

1.2.2) ในแง่ของประโยชน์ทั่วไปที่ได้รับ

1. ได้รับความรู้จากการค้นคว้าข้อมูลเพิ่มขึ้น เนื่องจากเนื้อหาส่วนใหญ่จะไม่ค่อยมีอยู่ในตำราหรือหนังสือประกอบการเรียน ดังนั้นผู้จัดทำจะต้องค้นหาข้อมูลจากแหล่งต่างๆ เท่าที่คาดว่าจะหาได้ ตามที่อ้างอิงไว้ในส่วนท้ายของปริญญานิพนธ์

2. มองเห็นวิวัฒนาการต่างๆ ที่เกี่ยวกับการนำเทคโนโลยีเอเยนต์ ไม่ว่าจะเป็นในด้านการดำเนินธุรกิจบนอินเตอร์เน็ต ด้านการแพทย์ ด้านการติดต่อสื่อสาร รวมไปถึงการนำไปใช้ประโยชน์ในชีวิตประจำวัน

3. เข้าใจแนวคิดและรูปแบบการทำงาน ของเทคโนโลยีเอเยนต์ได้อย่างชัดเจน

1.3 ขอบเขตของโครงงาน

การทำงานแบ่งออกเป็น 2 ส่วน

1.3.1) รวบรวมงานวิจัยที่เกี่ยวข้องกับสถาปัตยกรรมของระบบเอเยนต์

1.3.2) พัฒนาตัวอย่างโครงงานเพื่อแสดงลักษณะการทำงาน

1.4 วิธีการดำเนินงาน

การดำเนินงานของโครงงานนี้มีขั้นตอนดังนี้

1.4.1 ศึกษา ค้นคว้า รวบรวมความรู้ต่างเกี่ยวกับระบบเอเยนต์ในอินเทอร์เน็ต ถึงลักษณะ ความหมาย โครงสร้าง รูปแบบ ความสามารถในการทำงาน และการนำไปประยุกต์ใช้ เพื่อให้เกิดความเข้าใจถึงหลักการทำงานของระบบเอเยนต์อย่างแท้จริง

1.4.2 ทำการออกแบบระบบเอเยนต์ที่จะนำมาพัฒนาเป็นระบบเอเยนต์ตัวอย่าง โดยอาศัยหลักการและเหตุผลจากการศึกษาค้นคว้าที่ได้

1.4.3 พัฒนาและทดสอบโปรแกรมเอเยนต์ที่ได้ออกแบบมานั้น เพื่อนำผลการทดสอบมาวิเคราะห์ถึงการนำทฤษฎีและความรู้ที่ได้จากการศึกษามาปฏิบัติจริง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ความรู้พื้นฐาน

2.1 ความรู้พื้นฐานเกี่ยวกับซอฟต์แวร์เอเยนต์

2.1.1 คำนิยามของเอเยนต์

เอเยนต์ เป็นคำที่เราใช้กันในชีวิตประจำวัน ซึ่งหมายถึงบุคคล หรือ หน่วยงานที่ทำหน้าที่ประสานหรือเป็นตัวแทนให้ผู้ใช้บริการในการติดต่อสื่อสาร หรือการหาข้อมูลจากที่ต่างๆ เช่น เราสามารถใช้บริการ เอเยนต์ จัดหางานเพื่อติดต่อหรือหางานที่เราต้องการ หรือ เราใช้บริการตัวแทนขายหุ้น (Broker) ในการซื้อขายหุ้นของบริษัทต่างๆที่เราต้องการถือหุ้นด้วย

สำหรับเทคโนโลยีทางคอมพิวเตอร์ซึ่งมีการพัฒนาอย่างรวดเร็วในทุกวันนี้ นักวิจัย รวมทั้งบรรดาบริษัทเอกชนต่างๆ ได้พยายามพัฒนาเทคโนโลยีทางคอมพิวเตอร์ขึ้นมา เพื่อช่วยให้มนุษย์สามารถทำงานได้สะดวก รวดเร็ว และลดงานที่ซ้ำซ้อนไม่จำเป็น งานต่างๆเหล่านี้ แน่นอนว่าจะต้องเป็นงานที่ใช้คอมพิวเตอร์เป็นหลัก และเป็นงานที่โปรแกรมคอมพิวเตอร์ สามารถทำงานได้ดีมีประสิทธิภาพเท่าเทียม หรือมากกว่า มนุษย์

โปรแกรมคอมพิวเตอร์เหล่านี้จะถูกเรียกรวมๆว่า " โปรแกรมตัวแทนผู้ช่วย (Software Agents) "

เอเยนต์ เป็นแนวความคิดใหม่ ซึ่งยังต้องการการพัฒนาและค้นคว้าอีกมากมาย เราไม่สามารถให้คำจำกัดความเดี่ยวๆว่า ซอฟต์แวร์ เอเยนต์ คืออะไรได้ ผู้ที่ทำการศึกษาเรื่อง เอเยนต์ ได้ให้คำนิยามเกี่ยวกับ เอเยนต์ ไว้ต่างกัน ดังเช่น

คำนิยามโดย Sankat Virdhagriswaran นักวิจัยจากบริษัท Crystaliz จำกัด กำหนดความหมายของ เอเยนต์ ไว้ว่า " แสดงถึง 2 แง่มุมของแนวคิด คือ หนึ่งความสามารถของ เอเยนต์ ในการทำงานโดยอัตโนมัติ และ สองความสามารถของ เอเยนต์ ในการทำงานได้ตามขอบเขตที่กำหนดไว้ Sankat Virdhagriswaran ให้ความสำคัญของคำว่า " อัตโนมัติของ เอเยนต์ "

Russell และ Norvig กล่าวว่า "เอเยนต์ สามารถรับรู้ความเปลี่ยนแปลงเกี่ยวกับสิ่งที่อยู่ รอบตัวมันหรือ สิ่งแวดล้อมของมันผ่านทางเซนเซอร์ (Sensors) และมีปฏิกิริยาตอบต่อการเปลี่ยนแปลงนั้น " คำนิยามนี้เน้นที่สิ่งแวดล้อมตัวของ เอเยนต์ ซึ่งการกระทำของ เอเยนต์ นั้นจะเป็นอย่างไรก็ขึ้นอยู่กับ สิ่งแวดล้อมนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่สามารถอธิบายได้ว่า เอเยนต์ เป็นโปรแกรมที่มนุษย์ให้ข้อมูลและเป้าหมายในการทำงานใดๆ แล้ว เอเยนต์ นั้นก็นำเอาข้อมูลที่ได้และรายละเอียดที่เก็บเป็นเรคคอร์ด ภายในตัวมันเองมารวมกันและประมวลผลเพื่อทำงานที่มนุษย์สั่งให้ทำ โดย เอเยนต์ นั้นจะมีสิทธิอำนาจตามแต่ที่ผู้เป็นเจ้าของหรือผู้สั่งงานมอบให้ และ เอเยนต์ จะมีการทำงานภายใต้สภาพแวดล้อมของมัน

2.1.2 คุณสมบัติของ เอเยนต์ (Agent Attributes)

โปรแกรมเอเยนต์แตกต่างจากโปรแกรมทั่วไป กล่าวได้ว่า เอเยนต์ทุกเอเยนต์เป็นโปรแกรมแต่แต่ละโปรแกรมไม่จำเป็นต้องเป็นเอเยนต์เสมอไป ซึ่งโปรแกรมที่จะเป็นเอเยนต์ได้จะต้องมีคุณสมบัติดังนี้

1. เป็นอัตโนมัติ คือสามารถเริ่มต้น, จบ และควบคุมการทำงานโดยตัวมันเองได้ (Autonomous)

2. มีความฉลาด เพราะเอเยนต์จะมีการทำงานแบบ AI ซึ่งทำให้มันสามารถตัดสินใจในการทำงานได้ โดยจะต้องมีการเรียนรู้และดัดแปลงในเหมาะสม (Intelligent)

3. ความหนักแน่นและแน่นอนที่จะทำงานให้สำเร็จผลโดยไม่มีการเปลี่ยนแปลงการทำงานไปจากที่ได้รับมอบหมาย (Persistent and Goal-directed)

4. เดินทางไปตามสถานที่หรือเครื่องอื่นๆเพื่อเรียกใช้ทรัพยากรได้ (Mobile)

5. ยืดหยุ่นการทำงานได้ เช่นมีการจัดลำดับความสำคัญของงาน (Flexible)

6. ติดต่อสื่อสารกับเครื่องอื่นๆที่เดินทางไปใช้ข้อมูล, ติดต่อสื่อสารกับเอเยนต์ด้วยกันเอง และติดต่อสื่อสารกับผู้ใช้งานมันเองได้ (Communicative)

7. มีระบบรักษาความปลอดภัยและดูแลรับผิดชอบทรัพย์สินของผู้ใช้งานได้ : กรณีที่เป็นเอเยนต์ในเรื่องการเงิน (Secure)

8. มีปฏิกิริยาโต้ตอบ สามารถสังเกตุ, รู้สึก และมีปฏิกิริยาตอบสนองต่อการเปลี่ยนแปลงในสภาพแวดล้อมของมันและมีการตัดสินใจว่าจะทำอย่างไรกับการเปลี่ยนแปลงนั้น (Reaction)

บางเอเยนต์อาจจะมีไม่ครบทุกคุณสมบัตินี้และบางเอเยนต์อาจจะมีคุณสมบัตินอกเหนือจากนี้ ซึ่งนั่นก็ขึ้นอยู่กับการออกแบบการใช้งานของเอเยนต์

ปัจจุบันมีการพยายามสร้างเอเยนต์ให้เป็นเหมือนคน คือสามารถโต้ตอบและมีอารมณ์ของเอเยนต์เข้ามาเกี่ยวข้อง ซึ่งนั่นทำให้ในอนาคตเมื่อเราใช้งานคอมพิวเตอร์เราจะรู้สึกเหมือนกับทำงานกับคนคนหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.3 โปรแกรมที่ถูกเรียกว่าเอเยนต์ต่างจากโปรแกรมทั่วไปอย่างไร ?

โดยทั่วไปแล้ว " Script " หรือโปรแกรมทั่วไป คือ เซ็ทการกระทำของ pre-programmed ปกติการทำงานจะเป็นแบบลีเนียร์ ตามรูปแบบที่กำหนดไว้ชัดเจน สคริปต์มักจะถูกเขียนในรูปของ Procedural language ซึ่งคำสั่งทั่วไปของโพรซีเดอร์จะเกี่ยวกับ สมมติเงื่อนไขล่วงหน้า , ดำเนินการกระทำ และ ส่งผลลัพธ์ที่ได้กลับคืน เช่น สับรูทีนย่อยของ Perl หรือ ฟังก์ชันในภาษา C++ เป็นต้น

ภาษาโพรซีเดอร์ใช้ได้ดีกับงานหลายชนิด โดยเฉพาะอย่างยิ่งงานที่กำหนดข้อแก้ปัญหาไว้เป็นอย่างดีแล้ว แต่โชคร้ายที่ มันใช้ได้ไม่ดีกับปัญหาที่นำไปสู่เรื่องของงานที่มีลักษณะโต้ตอบซึ่งกันและกันอย่างกระจัดกระจาย เช่น ท่ามกลางเหตุการณ์การเจรจากันระหว่างเอเยนต์หลายตัว ในสถานการณ์เช่นนั้นการแสดงออกหรือแสดงการกระทำย่อมไม่แน่นอน ดังนั้น แผนงานและจุดมุ่งหมาย ของปัญหาเช่นนี้มักจะนำ Declaration language มาใช้งาน ซึ่งประสิทธิภาพการทำงานจะที่ดีกว่าใช้ภาษาโพรซีเดอร์

เนื่องจากเอเยนต์ มีลักษณะการทำงานที่ไม่สามารถมีข้อกำหนดไว้อย่างชัดเจนได้ คล้ายการจ้างงานมนุษย์ที่ต้องมีส่วนอื่นๆ มาเกี่ยวข้องด้วย เช่น ความรู้สึก , ความเชื่อถือ และ ข้อผูกมัด ซึ่งเอเยนต์บางตัวทำได้เหมือนอย่างมนุษย์จริง ๆ ก็มี เมื่อให้กลุ่มเงื่อนไขในลักษณะเดียวกันและเป้าหมายเดียวกัน พบว่า สคริปต์จะให้ผลลัพธ์การทำงานที่เหมือนเดิมเสมอ แต่ในขณะที่เอเยนต์ จะให้ผลลัพธ์ที่แตกต่างกันขึ้นอยู่กับสภาพความรู้สึกในขณะนั้น , ความเชื่อ และอื่นๆ

เอเยนต์สามารถเรียนรู้ได้และปรับตัวให้เข้ากับสถานการณ์ใหม่ ๆ ได้ ในขณะที่สคริปต์ ที่ฉลาดมาก ๆ นั้น ทำได้เพียงแค่คร่าว ๆ เท่านั้น

เอเยนต์ที่ได้ชื่อว่าเป็น Intelligent Agents

ลักษณะการทำงานของโปรแกรมซอฟต์แวร์ประเภทซอฟต์แวร์เอเยนต์ (หรือ Software Helpers) ซึ่งมีขั้นตอนการทำงานดังต่อไปนี้

1.ผู้ใช้คอมพิวเตอร์ป้อนข้อมูลรายละเอียดส่วนตัว พร้อมระบุชนิดประเภทของข้อมูลที่ตนเองสนใจ อย่างเช่น ผลรายงานพยากรณ์อากาศ ข้อมูลคร่าวๆ ของภาพยนตร์ที่กำลังจะลงโรงหรือที่กำลังฉายอยู่ ข้อมูลราคาซื้อขายหลักทรัพย์ หรือข่าวสารเกี่ยวข้องกับธุรกิจของตนเอง ฯลฯ โดยอาจจะระบุลงไปด้วยว่าต้องการให้โปรแกรมทำการอัพเดตข้อมูลบ่อยแค่ไหน

2. ระหว่างที่คอมพิวเตอร์ไม่ได้ถูกใช้งาน (สภาพ Idel ที่ถึงจะไม่ได้ถูกใช้งาน แต่คอมพิวเตอร์ต้องถูกเปิดให้มีกระแสไฟฟ้าไหลเข้ามาเลี้ยงด้วย) เช่น ในช่วงหลังเที่ยงคืนตัวโปรแกรมเอเยนต์ จะวิ่งออกไปสำรวจอินเตอร์เน็ต แล้วเก็บรวบรวมข้อมูลใหม่ ๆที่เกี่ยวข้องมาไว้ในหน่วยความจำสำรองบนเครื่องคอมพิวเตอร์ของตนเอง

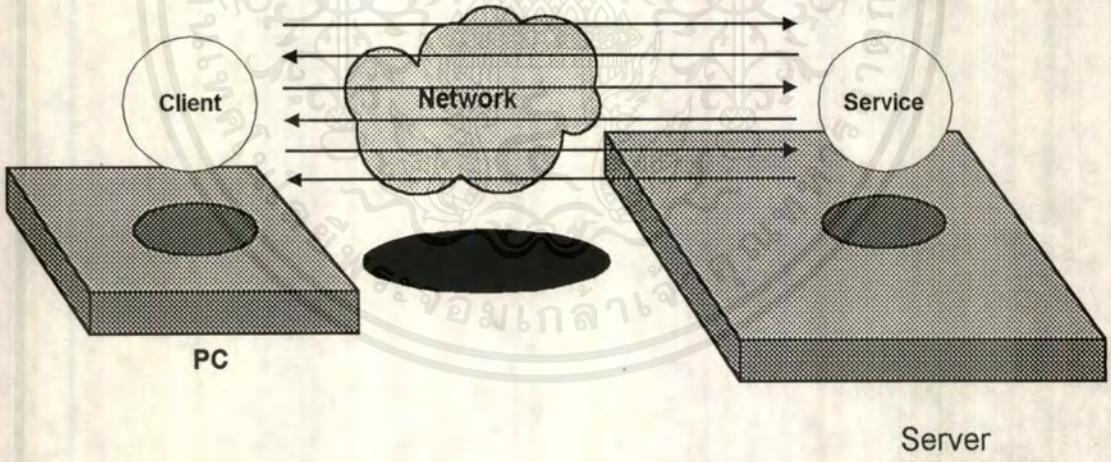
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. เมื่อผู้ใช้กลับมาใช้งานคอมพิวเตอร์ของตนอีกครั้ง ข้อมูลใหม่ ๆ ที่ถูกดาวน์โหลดมาเก็บไว้ในหน่วยความจำสำรอง ก็จะถูกวางรอให้ผู้ใช้เรียกเปิดดูได้ทันที จึงเพิ่มความสะดวกสบายให้กับการใช้ชีวิตของผู้ใช้คอมพิวเตอร์ได้อย่างมากและไม่ต้องเสียเวลารอการติดต่อเข้าเว็บไซต์

2.1.4 การโปรแกรมแบบเอเยนต์ต่างจากการโปรแกรมแบบไคลเอ็นต์/เซอร์ฟเวอร์อย่างไร? (Agent-oriented programming)

<u>การใช้งานเดิม</u>

ในปัจจุบัน เราเคยชินกับระบบไคลเอ็นเซอร์ฟเวอร์ (Client/Server) หรือที่มีเครื่องคอมพิวเตอร์เป็นศูนย์กลางในการเก็บข้อมูลเรียกว่า เครื่องเซอร์ฟเวอร์ โดยมีเครื่องคอมพิวเตอร์ซึ่งเป็นเครื่อง เวอร์กสเตชัน ทำงานต่างๆบนตัวเครื่องของมัน และมีการเรียกใช้ข้อมูลจากเครื่องเซอร์ฟเวอร์ผ่านทาง เน็ตเวิร์ก ซึ่งการทำงานแบบนี้จะเป็นแบบ 1 คำตอบต่อ 1 คำสั่ง หรือ 1 คำตอบต่อ 1 คำถาม หรือ Remote Procedure Calling (RPC) คือเครื่องที่เป็นตัวเวอร์กสเตชัน จะส่งการร้องขอไปยังเครื่องเซอร์ฟเวอร์ เพื่อขอใช้แหล่งข้อมูลเครื่องเซอร์ฟเวอร์นั้น ก็จะส่งการตอบรับ แล้วเครื่องทั้ง 2 ก็จะเริ่มการส่งข้อมูลกันตาม รูปที่ 1



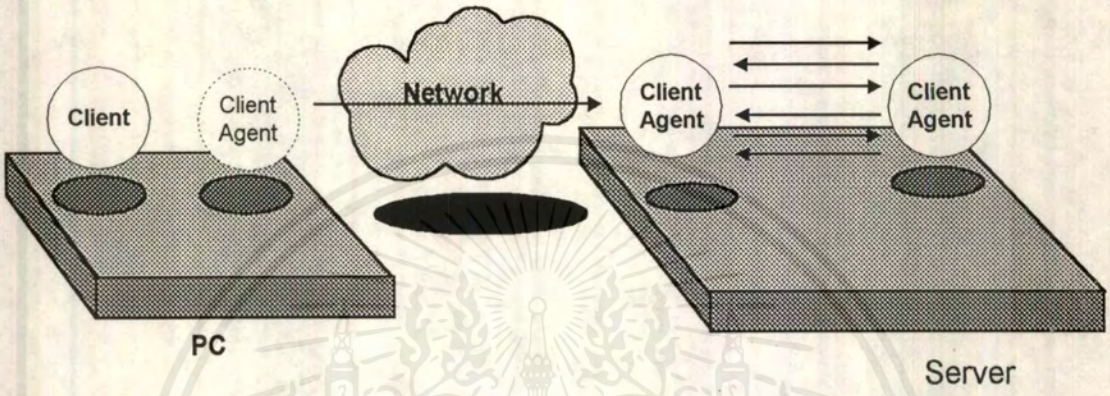
รูปที่ 1 การทำงานแบบ ไคล์เอ็นต์/เซอร์ฟเวอร์

การทำงานจะอยู่ภายในเครื่องคอมพิวเตอร์ที่เรียกใช้งาน ดังนั้นในการทำงานแบบไคล์เอ็นต์/เซิร์ฟเวอร์ นี้จะทำให้ระบบ เน็ตเวิร์ก แน่นไปด้วยข้อความที่แต่ละเครื่องส่งผ่านกันไปและมีการส่งข้อมูลกันมากซึ่งบางครั้งบางข้อมูลเกิดการผิดพลาดหรือสูญหายได้

<u>เอเยนต์ ช่วยได้</u>

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แนวความคิดใหม่ที่นำเอาเทคโนโลยีของ เอเยนต์ มาใช้เรียกว่า การทำงานระยะไกล หรือที่เรียกว่า Remote Programming (RP) เป็นการส่งผ่านตัวโปรแกรมจากคอมพิวเตอร์เครื่องหนึ่งสู่อีกเครื่องที่ไม่เพียงแต่เรียกใช้ข้อมูลหรือ Procedure แต่ต้องมีการจัดหาข้อมูลที่มันจะใช้หรือที่ เซอร์ฟเวอร์ ต้องใช้ด้วย ตัวโปรแกรมที่ส่งผ่านนั้นก็คือ เอเยนต์ นั่นเอง

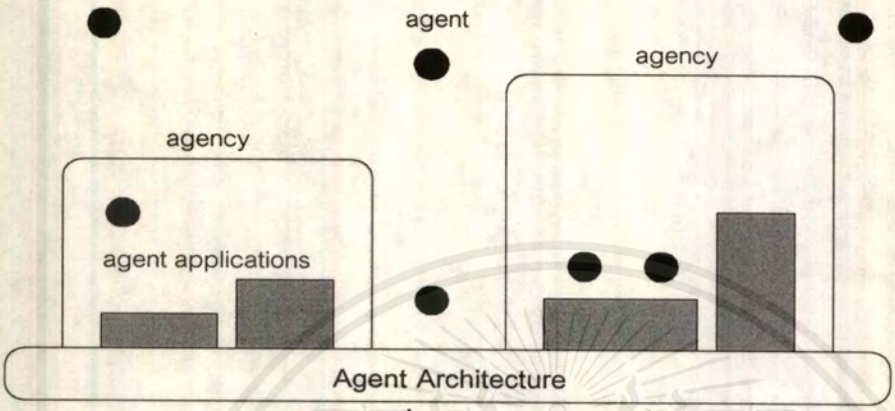


รูปที่ 2 การทำงานระยะไกลของเอเยนต์

การทำ RP นี้เป็นการส่งข้อมูลแบบ 2 ทาง (Bi-directional) จากรูป Telescript2 ตัวเอเยนต์ เดินทางจากเครื่องของผู้ใช้งานไปยังเครื่องเซอร์ฟเวอร์ บางครั้งก็อาจมีเอเยนต์จากเครื่อง เซอฟเวอร์ เดินทางมายังเครื่อง เวอร์กสเตชัน ก็ได้ ถ้าเป็นมนุษย์ก็คงจะเรียกว่าเป็น " การเคาะประตูบ้าน " เลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 สถาปัตยกรรมของ เอเยนต์ (Agent Architecture)



รูปที่ 3 แสดงสถาปัตยกรรมเอเยนต์

2.2.1 สถาปัตยกรรมของ เอเยนต์ คืออะไร

คำตอบง่ายๆ คือ สิ่งแวดล้อมที่อยู่รอบตัวมัน ซึ่งเป็นที่ที่ตัวซอฟต์แวร์เอเยนต์ เกิด, มีชีวิตอยู่, และตาย

ถ้ากล่าวในแง่ที่เป็นเทคนิคมากขึ้น ก็จะกล่าวได้ว่าสถาปัตยกรรมของ เอเยนต์ เป็นฐานรากที่ เอเยนต์ จะเริ่มขึ้นโดยเกิดจากหน่วยเล็กๆ 1 หน่วยแล้วค่อยขยายโดยเชื่อมเอาหน่วยต่างๆ มารวมกัน สร้างเป็นระบบของเอเยนต์ขึ้นมา เอเยนต์แอพพลิเคชันที่เป็นโปรแกรมใหญ่ๆจะมีส่วนประกอบจากหน่วยเล็กๆนี้มากมาย

สถาปัตยกรรมของ เอเยนต์ เป็นตระกูลเดียวกับพวกระบบปฏิบัติการ คือมันจะจัดเตรียมแหล่งข้อมูลให้กับตัว เอเยนต์ และ เอเยนต์ แอพพลิเคชัน ต่างๆ, ช่วยส่งข้อมูลในระดับชั้นต่ำๆ, และช่วยให้ผู้พัฒนาเอเยนต์ เกิดความสะดวกสบายในการสร้างเอเยนต์ขึ้นมาเหมือนกับระบบปฏิบัติการวินโดว์ ช่วยให้การเขียนโปรแกรมง่ายขึ้น เช่นนั้น

2.2.2 การเดินทางของ เอเยนต์ (Motivition for an Agent Architecture)

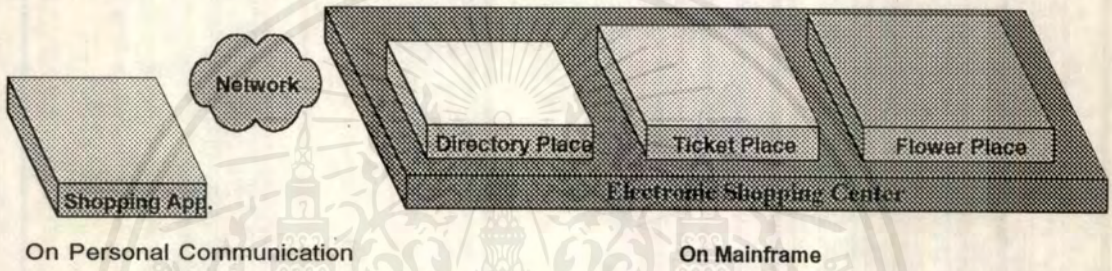
ในสถาปัตยกรรมของ เอเยนต์ มีส่วนประกอบใหญ่ๆ ทั้งหมด 3 ส่วน ซึ่งทั้ง 3 ส่วนนี้คือ อ็อบเจค ซึ่งประกอบด้วยแพคเก็จ ของโค้ดที่ถูกเขียนขึ้นมาเพื่อทำงานต่างๆกัน, คาต้าอ็อบเจค ,สแต็ก และ ตัวชี้คำสั่งเป็นส่วนประกอบของการทำงานของ เอเยนต์ ส่วนประกอบทั้ง 3 ได้แก่ Place, Agents และ Agent Applications

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

Place

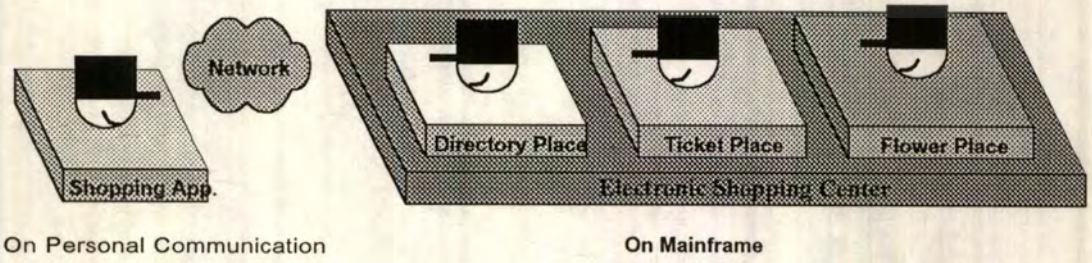
เป็นโค้ด ที่มีการระบุแอดเดรสหรือที่อยู่ของ Place ที่ตั้งขึ้นมานั้น อย่างเป็นเอกเทศ และเป็นที่ที่เปิดให้มีการติดต่อกันระหว่างข้อมูลภายในและภายนอกได้ Place เป็นที่ที่เอเยนต์ จะไปทำงานต่างๆตามที่มันได้รับคำสั่งมา โดยที่การทำการติดต่อหรือการทำ โพรเซส ใดๆ จะต้องไม่เกินขอบเขตที่กำหนดไว้ Places เป็นสถานที่เอเยนต์ ใช้เพื่อการพบปะหรือกระทำกิจกรรมที่มีส่วนเกี่ยวข้องกับการปฏิบัติหน้าที่ของตัวมันเองที่ได้รับมอบหมายให้ดำเนินการ ดังเช่นในรูป จะประกอบด้วย Places ต่างๆ คือ Directory Ticket และFlower



รูปที่ 4 Place ของเอเยนต์

Agents

เป็นตัวโปรแกรมที่ถูกส่งไปทำงานต่างๆ Agents จะได้รับคำสั่งให้ไปทำงานโดยจะมีข้อมูลติดตัวมันไปด้วย ได้แก่ ข้อมูลเกี่ยวกับงานที่จะทำ, แอดเดรส ของ Place ที่ต้องไปทำ, เวลาในการทำงานนั้น และบางครั้งอาจมีรายละเอียดของ เอเยนต์ ที่ต้องไปติดต่อแลกเปลี่ยนข้อมูลด้วย Agents ในแต่ละสถานที่ ที่เราสร้างขึ้นมานั้นก็จะมี เอเยนต์ ที่มีหน้าที่ดูแลจัดเสนอการให้บริการที่สถานที่นั้นแก่ เอเยนต์ที่มาขอใช้บริการ



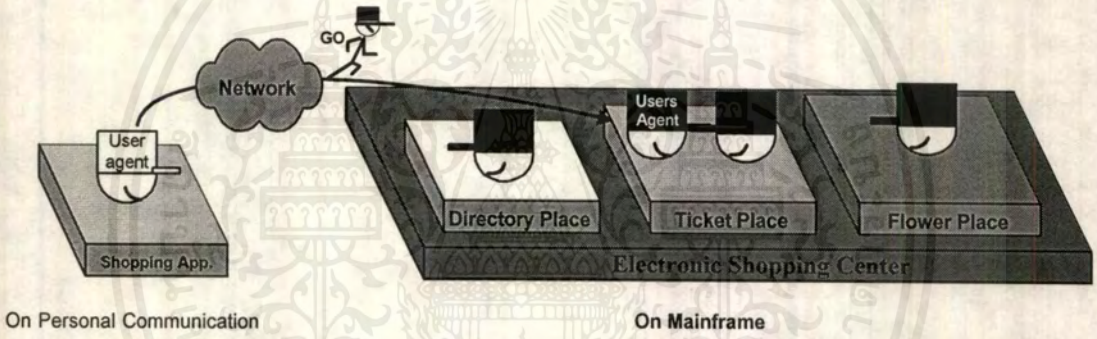
รูปที่ 5 รูปลักษณ์ของเอเยนต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Agent Applications

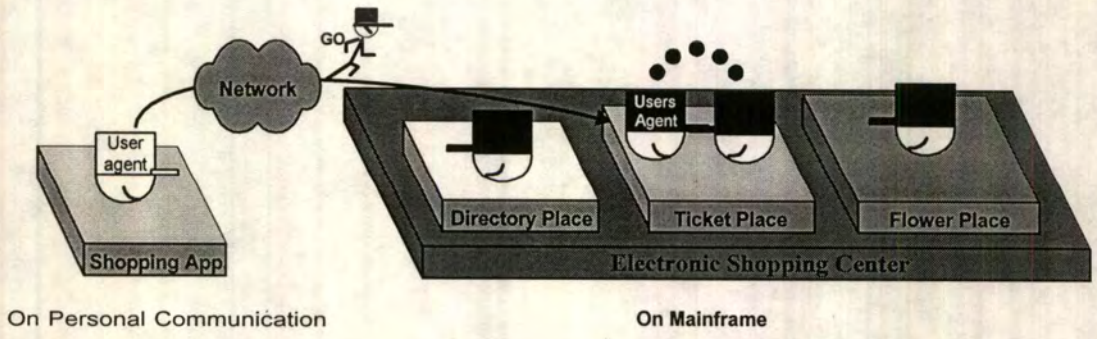
คือตัวโปรแกรมที่ช่วยให้มนุษย์ผู้ใช้งานสามารถติดต่อกับเอเยนต์ และช่วยในการออกคำสั่งให้กับเอเยนต์ซึ่ง Agent Applications จะรวบรวมข้อมูลต่างๆให้กับ เอเยนต์ ได้แก่ ข้อมูลเกี่ยวกับการเชื่อมโยงเข้าสู่ระบบฐานข้อมูลและการเชื่อมโยงกับแอพพลิเคชันต่างๆทางธุรกิจเพื่อติดต่อและขอใช้ข้อมูลหรือโปรแกรมเหล่านั้น นอกจากนี้ Agent Application ยังจัดหาข้อมูลจากผู้ใช้, เรื่องบริการต่างๆจากระบบปฏิบัติการ และเรื่องอื่นๆ ให้กับ เอเยนต์ ด้วย ซึ่งการทำงานของเอเยนต์แบ่งได้เป็นฟังก์ชั่นต่างๆดังนี้

- Travel : มีการเดินทางของเอเยนต์จากสถานที่หนึ่งไปสู่อีกที่หนึ่งได้ โดยผ่านทางเครือข่าย เพื่อดำเนินการติดต่อกับเอเยนต์ตัวอื่น ๆ ในระบบได้



รูปที่ 6 Travel

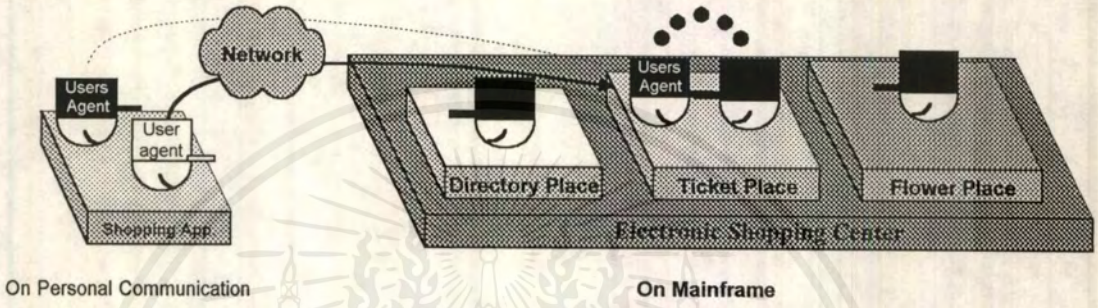
- Meetings : เมื่อมีการเดินทางข้ามเครือข่ายไปยังสถานที่อื่น ๆ ได้แล้ว ระหว่างตัวเอเยนต์แต่ละตัวนั้นย่อมมีความสามารถที่จะ ทำการเจรจาหารือกันได้



รูปที่ 7 Meeting

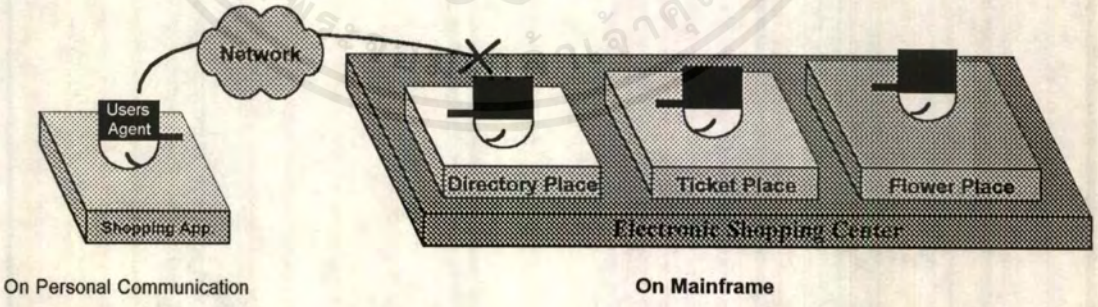
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Connections : ในระหว่างที่เอเยนต์ กำลังทำการเจรจาหารือกันอยู่ ที่สถานที่แห่งหนึ่ง ตัวเอเยนต์ที่เดินทางมาเจรจา สามารถจะทำการติดต่อกลับไปยังเครื่องที่มันอาศัยอยู่ได้ เพื่อช่วยส่งเสริมการตัดสินใจของผู้ใช้ในการทำงานกับระบบเอเยนต์



รูปที่ 8 Connection

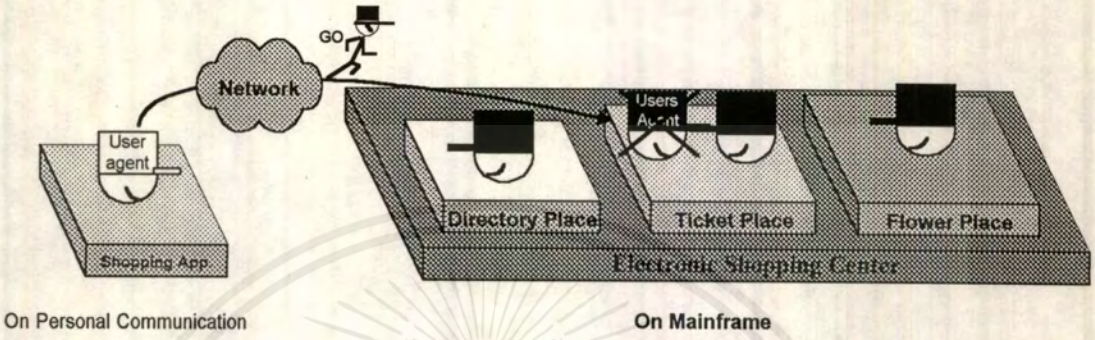
- Authorities : เป็นการมอบอำนาจหรือกรรมสิทธิ์ ให้ตัวเอเยนต์แต่ละตัว เพื่อให้มีความสามารถในการเลือกตัดสินใจได้ว่า จะเลือกติดต่อหรือเสนอบริการให้แก่เอเยนต์ใดได้บ้างและเอเยนต์ตัวใดที่จะปฏิเสธการให้บริการ



รูปที่ 9 Authorities

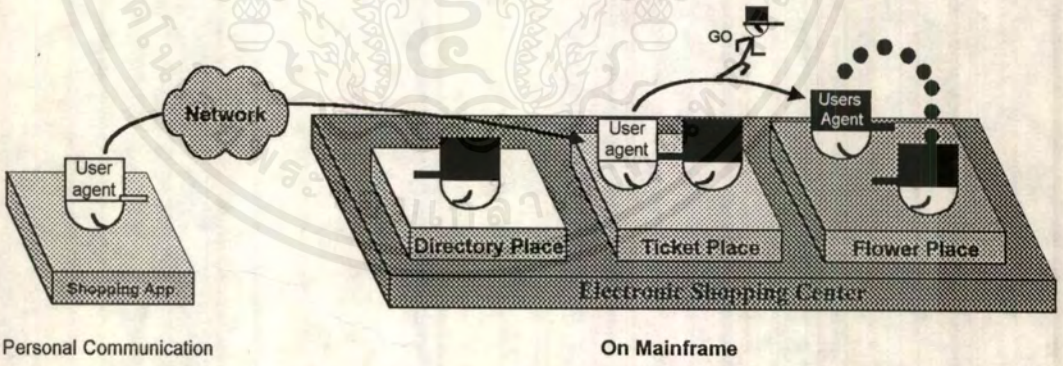
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Permits : กำหนดให้ตัวเอเยนต์ที่มีหน้าที่คอยเสนอบริการให้แก่ เอเยนต์ที่ต้องการมารับบริการนั้นมีสิทธิอย่างเต็มที่ว่า จะยอมให้ใช้บริการได้หรือไม่



รูปที่ 10 Permits

- Putting things together : จัดให้มีการนำเอาแนวคิดทั้งหมดที่มีอยู่มาใช้ร่วมกัน เพื่อกำหนดรูปแบบการทำงานของระบบเอเยนต์ที่เราต้องการได้ ว่ามีลักษณะเป็นอย่างไร



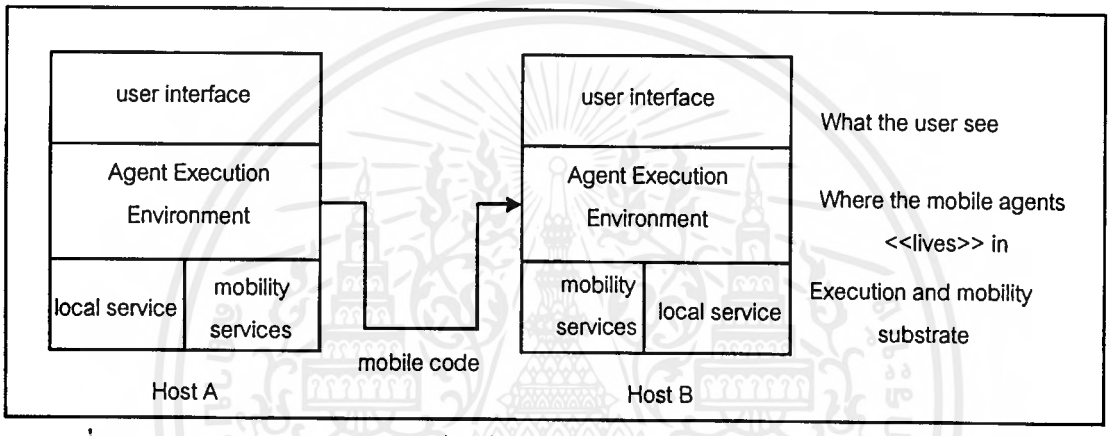
รูปที่ 11 Putting things together

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.3 ความสามารถในการเคลื่อนที่ของเอเยนต์ (Agent Mobility)

จากที่ได้กล่าวไปแล้วข้างต้น จะเห็นได้ว่าการเดินทางของเอเยนต์ หรือจริงๆแล้วเป็นการส่งตัวโปรแกรมข้ามระบบเครือข่าย จากที่หนึ่งไปอีกที่หนึ่ง, จาก เซอร์ฟเวอร์ หนึ่งไปยังอีก เซอร์ฟเวอร์ หนึ่งใน LAN เดียวกัน หรือแม้แต่จาก เซอร์ฟเวอร์ ใน LAN สู่ Domain อื่นๆใน Internet

รากฐานเอเยนต์ : โมบายล์โค้ด (The Foundation of Agents : Mobile Code)



รูปที่ 12 แสดงสถาปัตยกรรมซอฟต์แวร์ของระบบโมบายล์เอเยนต์ (Mobile Agent system)

จากรูป เราสามารถแบ่งขอบเขตได้เป็น 3 ส่วน โดยชั้นบนสุดเอเยนต์สามารถติดต่อกับมนุษย์ ผู้เป็นเจ้าของได้เพื่อแสดงผลลัพธ์การทำงานหรือรับคำสั่งการทำงานที่แน่นอน นี่คือสิ่งที่ผู้ใช้สามารถมองเห็นได้ โดยที่ตัวเอเยนต์เองนั้นจะอาศัยอยู่ในชั้นกลาง ที่ประกอบไปด้วยเครื่องคอมพิวเตอร์ทั้งหมดที่จัดเตรียมเอาไว้เป็นสภาพแวดล้อมการทำงานของเอเยนต์ ท้ายสุดในสภาพแวดล้อมเหล่านี้จะถูกนำมาเชื่อมต่อร่วมกันไว้สำหรับคอยให้บริการที่ช่วยให้ เอเยนต์สามารถกระโดดข้ามจากโหนดหนึ่งไปยังอีกโหนดหนึ่ง เข้าไปขอใช้บริการที่โฮสต์ของเครือข่ายท้องถิ่นที่อยู่ห่างไกล

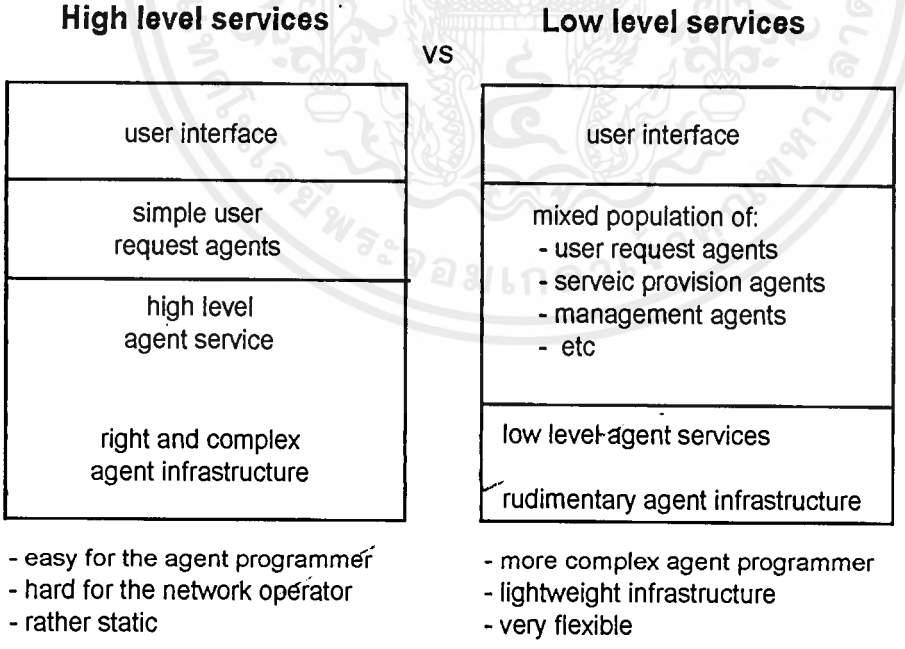
ในส่วนของแนวคิดเรื่องการเคลื่อนย้ายนั้นเป็นเรื่องง่าย เพราะเพียงแค่สร้างโมดูลพิเศษที่จัดการในเรื่องของการเคลื่อนย้ายขึ้นมา อย่างไรก็ตามวิธีการนี้ทำให้เราต้องประสบกับระบบใหญ่ที่เทอะทะ มาก เพราะจำนวนผู้ใช้งานที่มีมากนั่นก็หมายถึง ประชากรเอเยนต์ที่มีก็จะต้องมีความแตกต่างกัน หากเราสร้างระบบย่อย ๆ ขึ้นมาให้พอกับความต้องการทั้งหมดคงทำได้ยาก ดังนั้นอีกแนวทางหนึ่งคือ เสนอให้ส่วนที่ทำหน้าที่ดูแลเครือข่ายยุ่งเกี่ยวกับการเคลื่อนที่น้อยที่สุด ดังนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมเมอร์ที่สร้างเอเยนต์จึงต้องรับภาระในการสร้างความสามารถในการเคลื่อนที่ หรือต้องรวบรวมบริการต่าง ๆ ที่ต้องการด้วยตัวเอง เพราะไม่มีการรองรับในเครือข่าย

ในลักษณะแรกเป็นส่วนที่มีความสำคัญมากต่อปัจจุบัน คือ เปิดอำนาจการควบคุมดูแลอย่างมาก ให้แก่ผู้ให้บริการโครงสร้างรากฐานการเคลื่อนที่ เพราะคำนึงถึงความปลอดภัยในทรัพย์สินของผู้เป็นเจ้าของเป็นสำคัญ เป็นการต่อการแสดงการทำงานของเอเยนต์เพราะยังอยู่ในระดับที่สามารถเข้าใจได้ แต่อย่างไรก็ดี ลักษณะเช่นนี้ยังมีจุดอ่อนอยู่ที่ เป็นระบบของการทำงานที่ไม่มีความยืดหยุ่น หากเราต้องการเสนอบริการเพิ่มเติมเข้าไป จำเป็นอย่างยิ่งที่ต้องปรับเปลี่ยนการทำงานให้กับโฮสต์ ทุกตัวที่ใช้งานด้วย

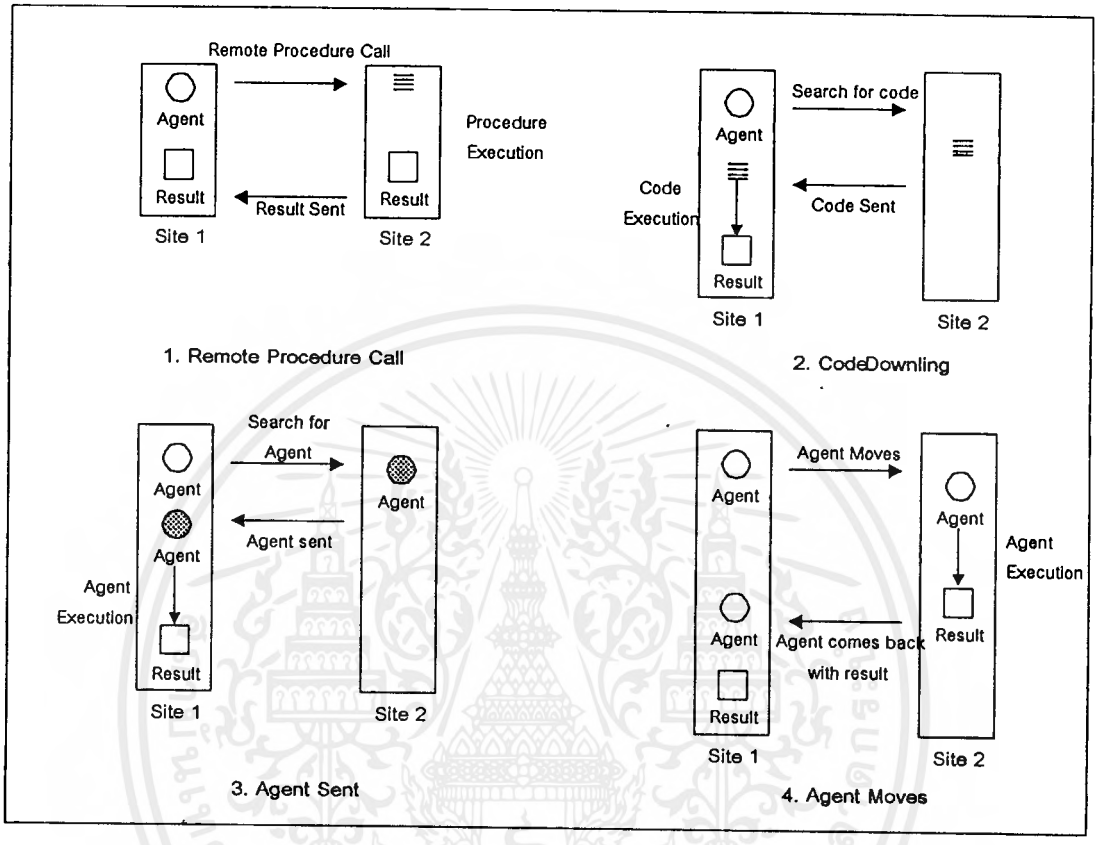
เรามีแนวทางที่จะสร้างบริการเบื้องต้น ที่คาดว่าเพียงพอต่อการใช้งานในอนาคตได้ ดังนั้นการสร้างบริการเสริมใหม่ ๆ จึงสร้างมาจากการเพิ่มข้อกำหนดต่าง ๆ ของการบริการเอเยนต์ ให้แก่ระบบ โดยปราศจากการเปลี่ยนแปลงโครงสร้างรากฐานเดิม ซึ่งลักษณะเช่นนี้เป็นการลดการแข่งขันในเรื่องความแตกต่างของเอเยนต์ลงได้มาก แต่ก็ยังมีจุดอ่อนอยู่ที่ เทคนิคใหม่จำนวนมากของข้อกำหนดบริการพื้นฐานของเอเยนต์ต้องได้รับการพัฒนาขึ้นมาเสียก่อน



รูปที่ 13 แสดง ข้อโต้แย้งของระดับชั้นที่ถูกต้องเพื่อสนับสนุนการเคลื่อนที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อย่างไรก็ตามการเดินทางบนระบบเครือข่ายของ เอเยนต์ สามารถทำได้ 4 รูปแบบคือ



รูปที่ 14 แสดงประเภทของโมบายล์เอเยนต์ 4 ประเภท

สมัยก่อนเรื่องของความสามารถในการเคลื่อนที่ (Mobility) เคยเสนออยู่ที่ระดับภาษา ซึ่งปกติเป็นทางเดียวที่ติดต่อได้ โดยโปรแกรมเมอร์ติดต่อกับสภาพแวดล้อมการทำงานที่สร้างให้มีการเคลื่อนไหวได้แท้จริง ดังนั้นแนวทางอย่างง่ายที่สุดในการที่จะทำให้มีการเคลื่อนที่ได้ ในยุคแรกนั้นจะมีก็เพียงแต่การใช้ Remote Procedure Call เท่านั้นมีการนำมาใช้งาน แต่ต่อมาภายหลังพบว่า จริง ๆ แล้วมันยังไม่ใช่การเคลื่อนที่อย่างแท้จริงเพราะเป็นเพียงการส่งผลลัพธ์การทำงานจากฝั่งรีโมตกลับมาเท่านั้น

ต่อมาเริ่มมีการดาวน์โหลดโค้ด อย่างง่าย ๆ เพิ่มเติมเข้ามา แต่ก็ยังไม่มีการเคลื่อนที่ของเอเยนต์อย่างจริงจัง แค่เรียกโค้ดมาทำงานแล้วก็ได้ผลลัพธ์ที่เครื่องเอเยนต์ได้เลย โดยที่การทำงานในลักษณะนี้ยังไม่มีการเก็บสถานะการทำงานและส่วนโค้ดที่ต้องดูแลเพราะว่าเรียกมาใช้งาน ในขั้นต่อมา มีการค้นพบภาษาที่ทำให้อ็อบเจค อพยพมาอยู่ร่วมกันและเก็บสถานะการทำงานของมันได้ แต่อ็อบเจคก็ยังไม่มีความเป็นอิสระในการเคลื่อนย้ายเพราะว่า การเคลื่อนย้ายอ็อบเจค เกิดมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเรียกใช้งานที่มาจากการร้องขอของอ็อบเจคตัวอื่น ๆ ตัวอ็อบเจคยังไม่มีความสามารถเคลื่อนที่ได้ตามความต้องการของตัวมันเอง

ความพยายามที่กำลังเกิดขึ้นอยู่ตอนนี้ เราพบการเคลื่อนที่โดยอัตโนมัติ มีความเป็นอิสระของตัวอ็อบเจคเอง ในตัวเอเยนต์มีการรวมสถานะการทำงานและพฤติกรรมต่าง ๆ ของตัวมันเองเข้ากับความสามารถในการเคลื่อนย้ายตัวเองไปยังที่ต่าง ๆ โดยไม่ต้องมีการร้องขอจากอ็อบเจคตัวอื่น จะเป็นลักษณะของการที่เอเยนต์มีเจตนาจะไปทำงานเอง

2.2.4 หน่วยทำงานของAgent (The Execution Facility)

Execution Facility หรือ "ที่ที่ทุกอย่างเกิดขึ้น" เป็นหัวใจของสถาปัตยกรรม เอเยนต์

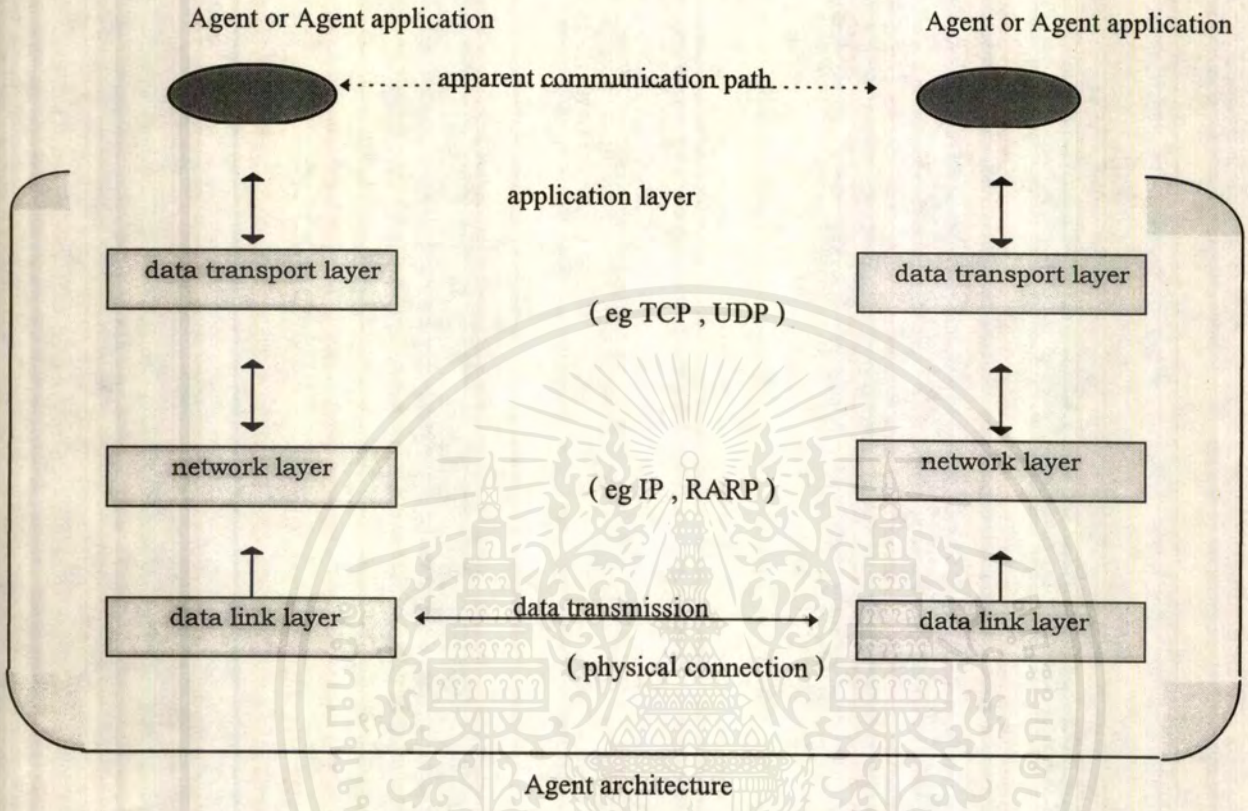
ถ้าเราเปรียบสถาปัตยกรรม เอเยนต์ เป็นโลกใบใหญ่ , Place เป็นสถานที่ที่ เอเยนต์ เดินทางไปเพื่อทำงานต่างๆ จะได้ว่า Execution Facility เป็นห้องประชุมใหญ่ในบ้านสำหรับให้ เอเยนต์ มาทำงานต่างๆในห้องนี้

Execution Facility จัดหาสภาพแวดล้อมการทำงาน ให้กับ เอเยนต์ใช้เวลาทำงาน เช่นเดียวกับทีระบบปฏิบัติการ จัดหาสภาพแวดล้อมการทำงานให้เวลาผู้ใช้งานทำโพรเซสใดๆ ซึ่งหมายความว่า Execution Facility นี้จะต้องมีการรองรับสิ่งต่างๆด้วยเช่น สิ่งที่ต้องใช้เวลา เอเยนต์ สื่อสารกันระหว่าง เอเยนต์ ด้วยกัน หรือระหว่าง เอเยนต์ กับ Place ต่างๆ, ช่วยให้ เอเยนต์ หาทรัพยากรและใช้ทรัพยากรนั้นๆได้อย่างง่าย, ช่วยให้ เอเยนต์ เดินทางจากที่หนึ่งไปอีกที่ และตรวจดูถ้ามีสิ่งที่ผู้ใช้งาน เอเยนต์ นั้นต้องการตาม Place ต่างๆ

Execution Facility ในแต่ละ Place จะเป็นที่ที่ โค้ด ของ เอเยนต์ ถูกแปลที่ละคำสั่งแล้วถูกส่งต่อไปยังระบบปฏิบัติการที่อยู่ข้างใต้ของแพลตฟอร์มนั้นให้ทำงานต่อไป ซึ่งโครงสร้างและรูปแบบของการทำงานทั้ง Execution Facilty และการติดต่อกับระบบปฏิบัติการก็ขึ้นอยู่กับภาษาที่ใช้เขียน เอเยนต์ นั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.5 การสื่อสาร(Communication)



รูปที่ 15 แสดงการติดต่อกันระหว่างเอเยนต์ใน OSI Layer

2.3 ภาษาของเอเยนต์ (Agent Languages)

2.3.1 ภาษาที่ใช้ในการเขียนโปรแกรมเอเยนต์ (Agent Programming Languages) และโมบายล์โค้ด (Mobile Object)

เทคโนโลยีที่นำมาพัฒนาเอเยนต์รู้จักกันในชื่อว่า Agent-oriented Programming คือมีการมองการทำงานของ อ็อบเจค ใน Object-oriented Programming ว่าเป็น อ็อบเจค ที่สามารถเดินทางได้จากที่หนึ่งไปยังอีกที่หนึ่งได้บนสภาวะแวดล้อมที่มีข้อตกลงเกี่ยวกับการเดินทางและการทำงานของมัน สภาวะแวดล้อมนั้นเรียกว่า Mobile Object System

Mobile Object System จะอนุญาตให้ อ็อบเจค สามารถเคลื่อนที่จาก การทำงาน ณ. ที่สภาวะแวดล้อมหนึ่งบนเครื่องคอมพิวเตอร์ ไปยังอีกสภาวะแวดล้อมหนึ่งบนเครื่องคอมพิวเตอร์อีกเครื่องหนึ่งได้ โดยที่สภาวะแวดล้อมต่างๆ เหล่านี้อาจจะมีการทำงานอยู่บนแพลตฟอร์ม ที่แตกต่างกัน, ทำงานกับระบบปฏิบัติการต่างชนิดกัน , และ graphicผู้ใช้ interface ที่แตกต่างกันก็ตาม ดังนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โค้ด ของ อ็อบเจค ที่ใช้งานนั้นบางครั้งอาจจะไม่สามารถทำงานได้ หรือถ้าหากว่าสามารถทำงานได้ มันก็อาจจะทำงานด้วยเวอร์ชันที่แตกต่างกัน ซึ่งอาจจะสามารถหรือไม่สามารถทำงานร่วมกับอ็อบเจค ตัวอื่นๆ ที่ต้องทำงานร่วมกันได้เลย

เพื่อความชัดเจน Mobility จะอ้างถึง การเคลื่อนย้ายข้อมูล สถานะของการทำงาน และ บางส่วนในการรับ-ส่งข้อมูลของแอพพลิเคชั่นนั้น ทั้งหมดที่ได้กล่าวมานี้จะถูกนำมารวมกันกับทรัพยากรที่มีอยู่เพื่อสนับสนุนการทำงานที่สภาวะแวดล้อมของการทำงานนั้น

ระบบ เอเยนต์ ส่วนใหญ่ที่ใช้ในปัจจุบันจะเป็นระบบที่สร้างมาจากภาษาที่มีการแปลแบบที่ละคำสั่ง (Interpreted Language)

ภาษาที่สามารถนำไปใช้งาน (Languages and Systems)

ลักษณะของภาษาที่มีความสำคัญต่อการตัดสินใจพัฒนาระบบเอเยนต์ ควรมีลักษณะดังนี้ :-

- **Portable representation**

ทั้งโค้ดและข้อมูล จะต้องอยู่ในรูปแบบสถาปัตยกรรมที่มีความเป็นอิสระ เช่น ในบางแอพพลิเคชัน อาจจะไม่ต้องการใช้ในลักษณะซอร์สโค้ด

- **ระบบปฏิบัติการและไลบรารี GUI**

บริการที่จัดเสนอที่แต่ละไซท์นั้น จะต้องมีความเป็นอิสระไม่ขึ้นกับระบบปฏิบัติการใดๆ (Operating system) และ ส่วนกราฟฟิคติดต่อการใช้งาน (GUI : Graphicผู้ใช้ Interface)

- **ความปลอดภัย**

ในการทำงานของอ็อบเจคหลายตัวที่ทำงานเฉพาะอย่างที่แตกต่างกัน ตามหน้าที่ของตัวเองจะต้องสามารถทำงานได้บนโหนดเดียวกันโดยปราศจากการแทรกแซงในการทำงานระหว่างกัน

ซึ่งในกรณีเช่นนี้ทำให้ภาษาที่ใช้นั้นควรจะต้องมีการทำ pointer-safe , type-safe และมีการกำหนดในเรื่องของ หน่วยความจำที่ใช้สำหรับอ็อบเจค ที่แตกต่างกันนั้นอย่างชัดเจน

- **ประเภทของข้อมูลที่ใช้งาน**

มีความจำเป็นในเรื่องของการแก้ไขชนิดที่ถูกต้องของอ็อบเจค ที่ได้รับมาจากเครือข่าย

- **การคอมไพล์แบบไดนามิก**

ในเรื่องของการคอมไพล์ (run-time compilation) ซึ่งมีความจำเป็นมากในการที่ต้องทำการคอมไพล์โค้ดที่ได้รับมาจากเครือข่ายใหม่อีกครั้ง ทั้งนี้ก็เพื่อเป็นการปรับปรุงความสามารถในการทำงานและเพื่อเกี่ยวกับในเรื่องการถ่ายทอดลักษณะของโลคอลโค้ด (Local code)

- **การจัดเนื้อที่หน่วยความจำ**

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อาจจะไม่มีความต้องการอย่างแท้จริงแต่ก็จำเป็นต้องมี

นอกจากนี้ปัญหาที่สำคัญก็ยังมีในเรื่องของ

- จะทำอย่างไรในการแยกระหว่างอ็อบเจคที่อยู่ที่เครือข่ายกับอ็อบเจคที่ต้องเคลื่อนที่
- โค้ด ในส่วนใดของ แอพพลิเคชัน ที่จำเป็นต้องมีการเคลื่อนย้ายการทำงาน
- สามารถจะรวมการทำงานของอ็อบเจคและคลาส อีกครั้งที่ สภาพแวดล้อมการทำงานที่เครือข่ายท้องถิ่นได้อย่างไร

ภาษาที่สามารถนำมาประยุกต์ใช้งานได้ :-

- Telescript
- Java
- Smalltalk
- Python
- Limbo[Inferno]
- IBM's Aglet

2.3.2 Telescript

Telescript เป็นแพลตฟอร์ม ซึ่งเกิดจากการพัฒนาทางเทคโนโลยีทางด้านภาษาและรูปแบบเพื่อให้รองรับการทำงานในระบบ เอเยนต์ ได้เป็นอย่างดี ซึ่งในที่นี้เราเรียกแพลตฟอร์ม ที่สนับสนุนการทำงานของ เอเยนต์ นี้ว่าเป็น Agent Operation Environment หรือ สภาวะแวดล้อมที่ เอเยนต์ ทำงาน ซึ่ง Telescript นี้เป็นรากฐานทางด้าน Agent Operation Environment โดยมันจัดรูปแบบสถาปัตยกรรมของ เอเยนต์ เพื่อช่วยจัดการการทำงานของ เอเยนต์ Telescript นี้ เป็นแพลตฟอร์ม แรกที่ผลิตออกจำหน่ายและเป็นต้นแบบให้กับการพัฒนาแพลตฟอร์ม ต่างๆสำหรับใช้สร้าง เอเยนต์ และระบบของมัน ในปัจจุบัน

หลักการในการเขียนโปรแกรมภาษา Telescript คือคอมพิวเตอร์ทั้ง 2 เครื่องที่จะติดต่อกันตามแบบ Remote Programming ได้จะต้องมีชุดคำสั่งและรูปแบบข้อมูลที่ถูกต้องตามข้อกำหนดของทั้งสองฝ่าย ซึ่งข้อกำหนดนี้ก็คือ ภาษาที่ใช้ในการเขียนโปรแกรม เอเยนต์ นั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาษาที่ใช้ในการสร้าง เอเยนต์ ประกอบด้วยชุดคำสั่งที่ยอมให้ Procedure ตัดสินใจการทำงาน, พิจารณา และเปลี่ยนแปลงสถานะของมัน และที่สำคัญคือสามารถเรียกใช้ Shared Procedure ที่ตัวเครื่องคอมพิวเตอร์ที่มันเดินทางไปหาได้ การเรียกใช้นี้ (Procedure Call) จะเป็น Local มากกว่าที่จะเป็น Remote

Telescript ถูกพัฒนาขึ้นโดยบริษัท General Magic จาก Sunnyville California ประเทศสหรัฐอเมริกา โดยมีวัตถุประสงค์หลักในการทำการส่งผ่านข้อความอิเลคทรอนิคส์ (Electronic Messaging) และการทำ Telephony ซึ่งใช้แนวความคิดตามหลักการของ Agent-oriented ที่มีการกระจายการทำงานตามแหล่งข้อมูลต่าง หรือ Remote Programming ที่กล่าวไปแล้วข้างต้น

Telescriptแพลตฟอร์ม ประกอบด้วย Agent Programming Language and interpreter ซึ่งจะทำงานบน AT&T PersonaLink Network ภาษาที่อยู่ใน Telescript เป็น Object-oriented และ Mobility-centric ซึ่งหมายความว่า เอเยนต์ สามารถเคลื่อนย้ายหรือเดินทางไปที่ต่างๆได้โดยยังสามารถรักษาสถาวะการทำงานภายในของตัว เอเยนต์ เองได้

Telescript จริงๆแล้วเป็นเพียงแพลตฟอร์ม สำหรับการสร้างโปรแกรมเท่านั้น ส่วนตัวโปรแกรมหรือ แอพพลิเคชัน จะถูกเรียกว่า Mobile Agent หรือ เอเยนต์ ที่เคลื่อนที่ได้ เนื่องจากความสามารถของแพลตฟอร์ม ที่ช่วยจัดการเรื่องการส่งผ่าน เอเยนต์ ไปตามเครื่องคอมพิวเตอร์ต่างๆในระบบ เน็ตเวิร์ก

โครงสร้างภายในของ Telescript ในวง LAN ซึ่งติดต่อกับแบบ Local Processing นั้น นอกจากจะมี Place และ เอเยนต์ แล้ว ยังมี Telesscript Engine ซึ่งจะอยู่ที่แต่ละ เน็ตเวิร์ก โฮสต์คอยทำหน้าที่ควบคุมและช่วยเหลือการทำงานของ เอเยนต์ Telecript Engine จะทำงานแบบทำได้หลายหน้าที่ (Multiple Processes) และยังคอยสับเปลี่ยนเอเยนต์ ที่เข้าทำงานบนเครื่องแต่ละเครื่องใน เน็ตเวิร์ก ที่ว่างให้ เอเยนต์ เข้าไปทำงานได้ Engine จึงรองรับ เอเยนต์ หลายตัวได้ในเวลาเดียวกันโดยที่ เอเยนต์ เหล่านั้นก็ผลัดกันใช้ข้อมูลและผลัดกันทำงานบน เซอร์ฟเวอร์ ต่างๆ

ข้อมูลเพิ่มเติมเกี่ยวกับ Telescript และบริษัท General Magic อยู่ที่ เว็บ Site http://www.genmagic.com

2.3.3 JAVA

เนื่องจากการพัฒนาของเทคโนโลยีด้านภาพ ความสามารถและความเร็วบนเว็ปบราวเซอร์ได้เพิ่มขึ้นอย่างรวดเร็ว เราสามารถโหลดเว็บเพจได้ในเวลารวดเร็ว ,ภาพจะถูกโหลดขึ้นมาพร้อม

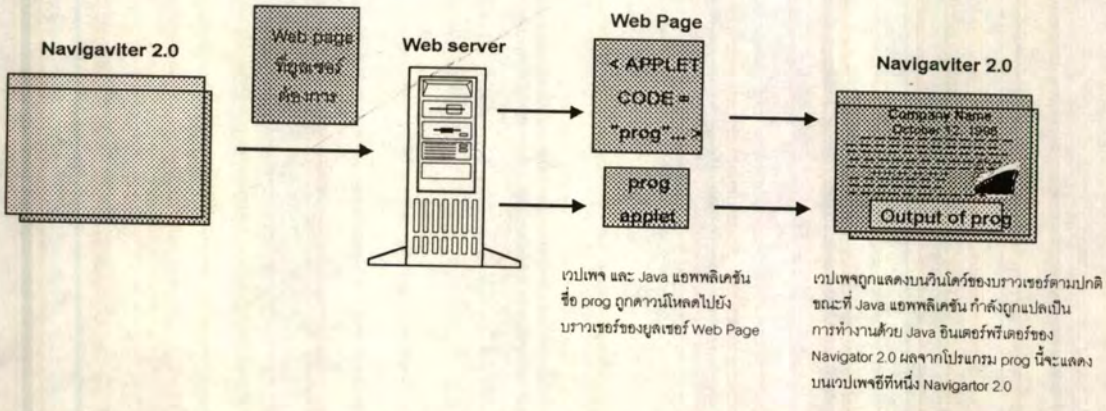
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กันหลายๆภาพ นอกจากนี้ภาษาเฉพาะของเว็บที่เรียกว่า HTML ก็ได้ถูกเพิ่มเติมและเปลี่ยนเวอร์ชันเป็นเวอร์ชัน 3.0 ทำให้เว็บเพจที่ดูผ่านบราวเซอร์เหล่านี้น่าตื่นตาตื่นใจและสวยงามกว่าเดิมมาก ไม่ว่าจะเป็นเรื่องของฉากหลัง , การควบคุมการติดต่อกับผู้ใช้ , ตัวอักษรและสีสันที่สวยงาม

มีการพัฒนาภาษาโปรแกรมแบบใหม่เพื่อใช้สำหรับเว็บ ที่มีชื่อว่า Java (จาวา) โดย Sun Microsystem และทาง Sun ยังได้ออกบราวเซอร์ตัวใหม่ที่ชื่อ HotJava ซึ่งใช้มาตรฐาน HTML 3.0 และสนับสนุน จาวาอย่างเต็มที่ ภาษา จาวาเป็นภาษาที่พัฒนามาจากภาษา C++ ซึ่งเป็นภาษาแบบออปเจ็ค จึงกล่าวได้ว่า ภาษาจาวา มีคุณสมบัติดังนี้ :-

- เป็นภาษาที่ง่าย (Simple)
- เป็นแบบออปเจ็ค (Object-orient)
- ประมวลผลแบบกระจาย (Distribute)
- ทำตามลำดับขั้น (Interprete)
- แข็งแกร่ง (Robust)
- มีความปลอดภัย (Secure)
- มีสภาปัตยกรรมที่เป็นกลาง (Architecture neutral)
- เคลื่อนย้ายได้ (Portable)
- ประสิทธิภาพสูง (High Performance)
- มัลติเธรด (Miltithred)
- ไม่อยู่นิ่ง (Dynamic)

จาวา แอพพลิเคชันทำงานอย่างไร



รูปที่ 16 แสดงการทำงานของจาว้าแอพแพล็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติในเรื่องพอร์ตเทเบิลและความปลอดภัยของจาวา

พอร์ตเทเบิลของจาวา จะช่วยลดเวลาในการดูแลแก้ไขและการทำตลาดในเรื่องของแอพพลิเคชันหลายแพลตฟอร์มได้ โดยเพียงแต่เขียนแอพพลิเคชันเพียงครั้งเดียว หลังจากคอมไพล์แล้วแอพพลิเคชันสามารถจะรันได้ทุกแพลตฟอร์มที่มีจาวาบราวเซอร์ทำงานอยู่ ซึ่งหมายความว่าแอพพลิเคชันนี้เป็นอิสระจากตัวรันไทม์ของแต่ละแพลตฟอร์มและคลาสไลบรารีที่ใช้ เช่น ภาษานี้จะไม่สนใจว่าผู้ใช้มีหน้าจอที่เขียนอยู่นี้ทำงานอยู่บนวินโดวส์หรือ MacOS ผู้เขียนเพียงแต่เขียนคำสั่งหรือเขียนการแสดงผลที่ต้องการบนหน้าจอโดยใช้แอทริบิวส์ปกติเท่านั้น แม้การเขียนออปเจคแบบกลาง ๆ นี้จะทำให้ผู้เขียนไม่สามารถใช้คุณสมบัติที่มีในแต่ละแพลตฟอร์มได้ แต่วิธีนี้ช่วยเพิ่มคุณสมบัติพอร์ตเทเบิลให้กับแอพพลิเคชันได้ เมื่อมีแพลตฟอร์มใหม่ๆ เกิดขึ้นเพียงแต่ย้ายตัวรันไทม์ของจาวา ไปบนแพลตฟอร์มใหม่นี้เท่านั้น แอพพลิเคชันก็จะสามารถทำงานบนแพลตฟอร์มใหม่นี้ได้

จาวา เป็นภาษาแบบอินเตอร์พรีเตอร์ที่แตกต่างจากตัวอื่น ๆ เพราะมันสามารถทำงานจากซอร์สโค้ดที่มีอยู่ได้ ภาษาจะถูกคอมไพล์ให้เป็นภาษากลางที่เรียกว่า architecture - neutral bytecode ซึ่งมีขนาดเล็กแต่ง่ายต่อการแปลงเพื่อนำไปใช้งานมากกว่าซอร์สโค้ด หลังจากคอมไพล์เพียงครั้งเดียว โปรแกรมนี้จะถูกเก็บในเว็บเซอร์ฟเวอร์รอจนกว่าจะมีบราวเซอร์ที่สนับสนุน จาวาติดต่อเข้ามา และดาวน์โหลดมันไปแล้วตัวรันไทม์ในบราวเซอร์นั้นจะรันภาษากลางนี้เอง

เรื่องความปลอดภัยของ จาวา นั้นมีคุณสมบัติในการจำกัดและควบคุมการใช้ทรัพยากรที่ภาษาจะทำได้ เช่น แอพพลิเคชันที่เขียนขึ้นได้รับการอนุญาตให้สร้างคอนเน็คชันกับโฮสต์ของตนบนเน็ตเวิร์กเท่านั้น และไม่สามารถสร้างโปรเซสเซอร์ใหม่เองได้ ขอบเขตในการเข้าถึงไฟล์ในฝั่งโลคอลมีน้อยมาก

บราวเซอร์ที่สนับสนุนจาวาจะส่งโค้ดภาษานี้ไปยังตัวตรวจสอบไบต์โค้ดของจาวา เพื่อตรวจว่าโค้ดดังกล่าวไม่ผิดต่อกฎในเรื่องความปลอดภัย ต่อมาจึงรันตัวโหลดคลาสจาวาเพื่อตรวจสอบว่าโค้ดดังกล่าวไม่เรียกฟังก์ชันที่ไม่มีในคลาสที่มีอยู่

2.3.4 Smalltalk

ลักษณะของภาษา :-

- **ภาษา :**

เป็นภาษาที่เป็น อินเตอร์พรีเตอร์สามารถเก็บคำสั่งในรูปแบบที่เป็นไบต์โค้ด (Byte codes)

- **รูปแบบอ็อบเจค :**

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นภาษาที่มีหลักการทำงานเป็นแบบอ็อบเจคโอเรียลเต็ทที่แท้จริง สามารถจัดการในเรื่องของ single inheritance , dynamic binding ,unconstrained polymorphism และ strict encapsulation

♦ สภาพแวดล้อมการทำงาน :

มีการจัดการเนื้อที่ใช้งานในหน่วยความจำและคอมไพล์แบบไดนามิก

♦ คลาสที่นำกลับมาใช้งาน :

สามารถนำคลาสเดิม ที่มีอยู่กลับมาใช้ได้อีก

♦ **Platforms :**

สามารถรันได้บน UNIX (X-Window), PC (MS-DOS,MS-Windows,OS/2) และ Macintosh

2.3.5 Inferno(Tm)

ภาษาที่ใช้งาน คือ Limbo language

อาจจะมีความจำเป็นสำหรับบางแอพพลิเคชัน ซึ่งจะขึ้นกับฮาร์ดแวร์หรือทรัพยากรที่มีการใช้งานอื่นๆ เช่น โหลดโมดูลโปรแกรมที่ต่างกันเพื่อกระทำฟังก์ชันบางอย่าง Inferno(tm) คือ ระบบปฏิบัติการ สำหรับเป็นสื่อกลางที่จะบริการการติดต่อสื่อสารให้แก่ผู้ที่จะใช้งาน โดยมีวัตถุประสงค์เพื่อนำมาใช้งานกับสภาพการณ์ของระบบเครือข่าย ในรูปแบบต่างๆ ได้อย่างมาก เช่น การติดตั้งเครื่องรับโทรทัศน์เข้ากับระบบสายเคเบิล ,การติดตั้งระบบโทรศัพท์ หรือแม้แต่ อุปกรณ์ที่สามารถพกพาหรือนำติดตัวได้ และ ระบบเครือข่ายคอมพิวเตอร์ที่มีค่าใช้จ่ายไม่สูงมากนัก ส่วนใหญ่ที่มักพบเห็นได้ทั่วๆไป เช่น เคเบิลทีวี ,การส่งสัญญาณผ่านดาวเทียม และ อินเทอร์เน็ต

ลักษณะของภาษา :-

ลักษณะข้อกำหนดในส่วนของ Portability & versatility ที่มีหลายรูปแบบ ได้แก่

- **Portability across processors :** สามารถรันได้ บนโครงสร้างสถาปัตยกรรมของ Intel, MIPS และ AMD 29K และสามารถต่อเข้ากับโครงสร้างอื่นๆได้เช่นกัน
- **Portability across environment :** ทำงานกับระบบปฏิบัติการที่เป็น stand-alone บนเครื่อง IBM compatible Pcs และ บนเครื่อง 29K-based palm-top machine อีกทั้งยังสามารถรัน เป็นแอพพลิเคชันของผู้ใช้ บน Unix,Window NT,Window 95 และ Plan 9 โดยที่ environment ทั้งหมดจะกระทำการอินเตอร์เฟสกับอินเฟอร์โน application
- การทำงานแบบกระจาย: ในสภาพแวดล้อมการทำงานที่เป็นชเหมือนกันที่ได้จัดตั้งไว้ที่เครื่องเทอร์มินอลของผู้ใช้ และที่เซิร์ฟเวอร์ โดยแต่ละที่อาจจะมีการนำทรัพยากรจากที่อื่นมาใช้ ซึ่งได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ช่วยเหลือการติดต่อสื่อสารกันในเรื่องเวลาในการเอ็กซีคิ้วของระบบ โดยทำให้สามารถแยกแอพพลิเคชั่นที่จะทำงานระหว่าง ไคล์เอนต์และเซอร์ฟเวอร์ ได้ง่าย

- **ต้องการฮาร์ดแวร์น้อย** : สามารถจะรันแอพพลิเคชันบนเครื่องแบบสแตนอโลน โดยใช้หน่วยความจำเพียงเล็กน้อย ประมาณ 1 MB และไม่ต้องใช้ฮาร์ดแวร์เพื่อที่จะทำการจัดหน่วยความจำ
- **แอพพลิเคชันที่เคลื่อนย้ายสะดวก** : สามารถสร้างอินเฟอร์โน application โดยใช้ภาษา Limbo(tm) ซึ่งเป็นภาษาที่สร้างขึ้นมาโดยเฉพาะ และมีการแทนค่าในรูปเลขฐานสองเหมือนกันกับแพลตฟอร์ม อื่นๆ ทั้งหมด
- **การปรับเปลี่ยนแบบไดนามิก** : อาจจะเป็นบางแอพพลิเคชัน ซึ่งขึ้นอยู่กับฮาร์ดแวร์หรือทรัพยากรอื่นๆ ที่สามารถใช้งานได้, โหลดโมดูลจากโปรแกรมอื่น เพื่อทำงานกับฟังก์ชันที่กำหนดไว้ เช่น ในแอพพลิเคชั่นของ เครื่องเล่นวีดีโอ อาจจะต้องการใช้โมดูลของตัวถอด รหัสที่แตกต่างกันจำนวนมากก็ได้

อินเตอร์เฟสของอินเฟอร์โน (Inferno interface)

บทบาทหน้าที่ของระบบอินเฟอร์โน (Inferno system) คือ สร้างมาตรฐานการอินเตอร์เฟสแบบต่างๆ ให้แก่แอพพลิเคชั่นของระบบ

แอพพลิเคชั่น ใช้ทรัพยากรต่างๆ ที่มีภายในระบบ เช่น วิชวลแมชชีน (virtal machine) ที่ใช้รันโปรแกรมของแอพพลิเคชัน รวมทั้งโมดูลไลบราลีที่แสดงการบริการ ตัวอย่างที่เห็นได้ชัดได้แก่ การถ่ายเทสตริง ไปยังการบริการในรูปแบบกราฟฟิค สำหรับใช้งานกับ ข้อความ(Text) ,รูปภาพ(Pictures) และ วีดีโอ(Video)

แอพพลิเคชั่น มีทรัพยากรที่ใช้งานอยู่ภาพนอกสภาพแวดล้อมการทำงาน เช่น ไฟล์ข้อมูลที่สามารถอ่านและแตะต้องใช้งานได้ รวมทั้ง อ็อบเจค ที่ถูกกำหนดและใช้งานได้คล้ายกับไฟล์แต่จะใช้งานได้ดีกว่า และอุปกรณ์ (เช่น เครื่องควบคุมระยะไกล - remote control , MPEG decoder หรือ network interface) ที่แสดงตัวเองต่อแอพพลิเคชันในลักษณะไฟล์เช่นกัน

มาตราฐานโปรโตคอลที่มีใช้เพื่อในการติดต่อสื่อสารทั้งภายในเครื่อง และระหว่างเครื่องที่แยกกันรันอินเฟอร์โน เพื่อที่ว่าแอพพลิเคชันสามารถทำงานร่วมกันได้

และในขณะเดียวกันอินเฟอร์โน ก็จะใช้การอินเตอร์เฟสสนับสนุนสภาพแวดล้อมการทำงานที่มีอยู่ ทั้งที่เป็นฮาร์ดแวร์แท้ๆ หรือ มาตรฐานระบบปฏิบัติการและโปรโตคอล

External Enviroment of Inferno Application

ในการออกแบบมีหลักสำคัญ 3 ประการคือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.ทรัพยากรทั้งหมดจะต้องถูกกำหนดไว้ก่อน และสามารถเข้าถึงได้คล้ายไฟล์ในกลุ่มของการจัดลำดับชั้นของระบบไฟล์

2.การจัดลำดับชั้น (Hierarchies) ที่ไม่มีการติดต่อถึงกัน ซึ่งมีการจัดการกับบริการที่แตกต่างจะถูกนำมาจัดไว้รวมกันเป็น ลำดับชั้นส่วนตัวเดี่ยวๆ

3.โปรโตคอลที่ใช้ในการติดต่อสื่อสาร เรียกว่า Styx ถูกนำมาใช้ให้เป็นรูปแบบในการเข้าถึงทรัพยากรเหล่านี้ ทั้งที่อยู่ที่โลคอล และรีโมต

2.3.6 IBM Aglets Workbench

Aglets Workbench เป็นมุมมองการเปลี่ยนแปลงพาราไดม์หรือรูปแบบแห่งวิธีการเข้าถึงข้อมูลที่ใกล้จะมีขึ้นในการประมวลผลบนเครือข่าย ยึดหลักบนภาวะฉุกเฉินของการปฏิบัติภาระกิจของโมบายเอเยนต์ในสภาพแวดล้อมต่าง ๆ และเกี่ยวเนื่องกับสิ่งต่าง ๆ ที่อยู่ภายในเรื่องของ Aglets Workbench และสัมผัสกับการออกคำสั่งที่จำเป็นด้วย เช่น ความปลอดภัย และกฎเกณฑ์มาตรฐาน

Java Aglet Application Programming Interface (J-AAPI)

เป็นมาตรฐานสำหรับการติดต่อของ Aglet และสภาพแวดล้อมของมัน นักพัฒนาแอพพลิเคชันสามารถเขียน Aglets บนแพลตฟอร์มอิสระ และคาดหวังให้ aglets เหล่านั้นรันได้บนเครื่องโฮสต์ใด ๆ ที่สนับสนุนการทำงานของ J-AAPI

Java Agent Transfer and Communication Interface (J-ATCI)

เป็นระบบเอเยนต์และมาตรฐานโปรโตคอลของเอเยนต์ที่เป็นอิสระ เพื่อให้เอเยนต์ได้เคลื่อนย้ายได้และทำการติดต่อสื่อสารภายในเครือข่ายคอมพิวเตอร์

Conference Papers

Mobile Agents: A New Paradigm for Distributed Object Computing on the WWW

รูปแบบใหม่ของการประมวลผลการทำงานแบบกระจายบนเวิร์ดไวด์เว็บ เป็นลักษณะการรวมกันของเทคโนโลยีสองตัว คือ เวิร์ดไวด์เว็บและการทำงานแบบกระจาย

Specifications

Agent Transfer Protocol (ATP)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Agent Transfer Protocol (ATP) เป็นโปรโตคอลระดับแอพพลิเคชัน ที่ถูกสร้างขึ้นมาเพื่อรองรับระบบการทำงานของเอเยนต์ที่มีการกระจายข้อมูล ATP นำเสนอรูปแบบโปรโตคอลที่ทำความเข้าใจได้ง่ายและไม่ขึ้นกับแพลตฟอร์มใดๆ เพื่อให้เอเยนต์สามารถเคลื่อนที่จากคอมพิวเตอร์เครื่องหนึ่งไปสู่อีกเครื่องหนึ่งในเครือข่ายได้อย่างสะดวก

Mobile Agent Facility (MAF) Specification

MAF เป็นส่วนหนึ่งของประโยชน์ที่มีของ CORBA โดยมี OMG เป็นผู้สนับสนุนซึ่งมีบริษัทที่ร่วมเป็นสมาชิกอยู่มากกว่า 600 บริษัท

IBM Aglets Workbench

Programming Mobile Agents in Java

โปรแกรมการทำงานโมบายเอเยนต์ด้วยจาวา

Aglets Workbench เป็นสภาพแวดล้อมเสมือนชนิดแรก ซึ่งมีไว้เพื่อสร้างแอพพลิเคชันต่างๆ บนเครือข่ายไว้สำหรับให้โมบายล์เอเยนต์ได้ทำการค้นหาข้อมูล เข้าถึงข้อมูลและจัดการกับเหล่าข้อมูลและข่าวสารอื่น ๆ ได้

โมบายล์เอเยนต์เป็นโปรแกรมที่สามารถส่งจากคอมพิวเตอร์เครื่องหนึ่งและเคลื่อนที่ไปทำงานยังคอมพิวเตอร์อีกเครื่องที่อยู่ไกลกัน เมื่อเดินทางมาถึงยังเครื่องที่อยู่ไกลแล้ว ก็จะแสดงตัวเองและเข้าไปขอใช้บริการต่าง ๆ และข้อมูลของคอมพิวเตอร์เครื่องนี้ เครื่องคอมพิวเตอร์ที่อยู่ไกลนั้นอาจจะเสนอตัวเองเป็นนายหน้า โดยจะนำเอเยนต์ต่าง ๆ ที่มีความสนใจคล้ายกันและจุดมุ่งหมายแบบเดียวกันมาอยู่รวมกัน ด้วยเหตุนี้จึงได้มีสถานที่ (Place) ไว้สำหรับให้เอเยนต์เหล่านั้นติดต่อพบปะกัน

ทำไมต้องมีโมบายล์เอเยนต์ ?

ทุกวันนี้ ส่วนหลักของพาราไดม์ (Paradigm)หรือรูปแบบแห่งการเข้าถึงข้อมูล จะติดอยู่กับเทคโนโลยีดิสทริบริวเต็ดอ็อบเจคทั้งหมด รวมเข้าด้วยกันเป็นการเข้าถึงข้อมูลในรูปแบบ Synchronous message-passing ซึ่งทุก ๆ อ็อบเจคจะถูกกระจายทั่วไปแต่มันจะอยู่ประจำที่ และจะทำการติดต่อกับอ็อบเจคตัวอื่น ๆ ได้ โดยผ่านทางเมสเสจ (message-passing) ซึ่งพาราไดม์ลักษณะเช่นนี้ยังไม่สมบรูณ์ และจำเป็นต้องมีการปรับปรุงส่งเสริมลักษณะบางอย่างของรูปแบบแห่งการเข้าถึงข้อมูลให้ดีขึ้น เช่น Asynchronous message-passing , Object mobility และ Active objects

พาราไดม์ใหม่ที่เกิดขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โมบายล์เอเยนต์สามารถจัดการกับพาราไดม์ที่มีรูปแบบเป็นหนึ่งเดียว เพื่อการประมวลผลแบบกระจาย มีทั้งแบบซิงโครไนซ์และอะซิงโครไนซ์ เมสเสจพาสซิ่งและอ็อบเจคพาสซิ่ง และอ็อบเจคที่อยู่ประจำที่และอ็อบเจคที่ต้องเคลื่อนที่ โมบายล์เอเยนต์เป็นหน่วยมูลฐานของการประมวลผลแบบกระจาย

- อ็อบเจคพาสซิ่ง (Object-passing) เมื่อโมบายล์เอเยนต์เคลื่อนที่ ตัวอ็อบเจคถูกส่งผ่านนั้นคือ ทั้งโค้ด ข้อมูล สถานะการทำงานและกำหนดการของการท่องเที่ยว จึงถูกส่งไปพร้อมกัน
- อัตโนมัติ (Autonomous) โมบายล์เอเยนต์มีข้อมูลข่าวสารเพียงพอต่อการตัดสินใจว่า จะทำอะไรที่ไหน และ เมื่อไร
- อะซิงโครไนซ์ (Asynchromous) โมบายล์เอเยนต์จะมีเธรดในการทำงานเป็นของตัวเอง และสามารถทำงานแบบไม่เป็นจังหวะได้
- (Local interaction)โมบายล์เอเยนต์ ติดต่อได้ทั้งโมบายล์เอเยนต์อื่น ๆ หรือ อ็อบเจคที่อยู่ประจำที่ได้ หากมีความจำเป็นก็สามารถส่ง เอเยนต์ส่งข้อความ (Messager Agents) หรือ Surrogate Agents ซึ่งเป็นโมบายล์เอเยนต์ทุกตัว เพื่ออำนวยความสะดวกแก่การติดต่อทางไกล
- การปฏิบัติงานเมื่อไม่มีการเชื่อมต่อ (Disconnected operation) โมบายล์เอเยนต์สามารถกระทำงานของตัวเองเมื่อการเชื่อมต่อสู่เครือข่ายถูกเปิดหรือปิดได้ ถ้าการเชื่อมต่อเครือข่ายถูกปิดและโมบายล์เอเยนต์จำเป็นต้องเคลื่อนที่ มันสามารถคอยจนกว่าการเชื่อมต่อจะเปิดอีกครั้ง
- การประมวลผลแบบขนาน (Parallel execution) มีโมบายล์เอเยนต์มากกว่าหนึ่งตัวสามารถถูกส่งไปยังที่ต่าง ๆ เพื่อทำงานเป็นแบบขนานกัน

ประโยชน์จากการนำเทคนิคในด้านโมบายล์เอเยนต์มาใช้มีอยู่มากมาย และไม่ได้มีเพียงทางเลือกเดียวจากการทำงานที่สามารถเสนอให้ นอกจากเพื่อสนับสนุนการบริการของเครือข่ายที่มีอยู่แล้ว โมบายล์เอเยนต์ยังทำให้เกิดบริการแบบใหม่ที่มีความเป็นไปได้ ด้วยเหตุนี้จึงทำให้เกิดการดำเนินธุรกิจรูปแบบใหม่ขึ้นอีกด้วย

Aglets Framework

จุดหลักของ Aglets Workbench คือ เอเจ็ทท์เฟรมเวิร์ก ซึ่งมีพื้นฐานอยู่บน ระบบการเขียนโปรแกรมด้วยภาษาจาวา เฟรมเวิร์กนี้จะจัดการกับองค์ประกอบที่เกี่ยวข้องกับการทำงานของโมบายล์เอเยนต์

การใช้งานคลาสของ Aglet จะเป็นวิธีที่ให้ความสะดวกแก่ผู้ที่จะระบุลักษณะเอเยนต์ เพื่อถ่ายทอดคุณสมบัติและหน้าที่แก่โมบายล์เอเยนต์ ซึ่งต้องมีรายละเอียดดังนี้

- มีข้อกำหนดของการตั้งชื่อ สำหรับเอเยนต์ที่เป็นรูปแบบอันเดียวกัน

เอกสารนี้เผยแพร่เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- หมายกำหนดการท่องเที่ยวสำหรับระบุรูปแบบการเดินทางที่ซับซ้อน ซึ่งมีที่หมายปลายทางมากกว่าหนึ่งที่และสามารถจัดการกับความผิดพลาดรวมทั้งข้อบกพร่องได้อย่างอัตโนมัติ
- กลไกกระดานขาว (White board) ที่อนุญาตให้เอเยนต์ทั้งหลายช่วยกันทำงานและใช้ข้อมูลข่าวสารร่วมกันแบบอะซิงโครไนซ์ได้
- ข้อกำหนดของการส่งเมสเสจในการติดต่อสื่อสารกันของเอเยนต์นั้น ต้องเป็นการสื่อสารแบบซิงโครไนซ์ เพียร์-ทู-เพียร์
- การโหลดคลาสเอเยนต์ในเครือข่าย ต้องยอมให้ไบต์โค้ดของเอเยนต์ที่เป็นภาษาจาวาและสถานะข้อมูลสามารถส่งผ่านในเครือข่ายได้
- สภาวะแวดล้อมการประมวลผลที่จัดให้เอเยนต์ ต้องเป็นลักษณะของสภาพแวดล้อมที่มีความเป็นอิสระไม่ขึ้นกับระบบการทำงานที่แท้จริงของเครื่องที่กำลังประมวลผล

อีกทางหนึ่งของเฟรมเวิร์กที่สนับสนุนการพัฒนาเอเยนต์ โดยนำการใช้แพทเทิร์นเข้ามาใช้งาน พื้นฐานการออกแบบและการพัฒนาแพทเทิร์น ได้รับการปรับปรุงให้ประสบความสำเร็จมากขึ้นมาแล้วในที่ต่าง ๆ เป็นจำนวนมาก ได้มีการเพิ่มจำนวนการใช้แพทเทิร์นเอเยนต์ระดับสูงมากขึ้นบนเฟรมเวิร์ก การใช้แพทเทิร์นนี้ได้อธิบายถึงลักษณะของคู่เอเยนต์ที่มีความสัมพันธ์ร่วมกัน เช่น แบบมาสเตอร์-สลาฟ (Master-Slave) แบบผู้ส่ง-ผู้รับ (Messager-Receiver) และแบบผู้แจ้งล่วงหน้า-ประกาศ (Notifier-Notification) ซึ่งลักษณะของแพทเทิร์นดังกล่าวนี้ จะถูกแทนในรูปของคลาสจำนวนหนึ่ง ที่สามารถใช้เป็นเทมเพทแก่นักพัฒนาเอเยนต์ต่อไป

โดยรวมแล้ว Aglets Framework ทั้งหมดเขียนด้วยจาวาซึ่งรับประกันได้ในเรื่องของความสามารถในการเคลื่อนย้าย ในคลาส API ของจาวาซึ่งมีเอกสารประกอบอยู่มากมาย จะอนุญาตให้เรานำโมบายล์เอเยนต์มารวมกันได้ทันทีบนเครือข่ายที่มีอยู่ในปัจจุบันและอนาคตเพื่อร่วมกันใช้ระบบข้อมูล

การพัฒนาสภาพแวดล้อมเสมือน

ตัวสร้างเสมือน (Visual Builders) เป็นหัวใจสำคัญแก่โปรแกรมเมอร์ที่มีประโยชน์ เปรียบได้กับการลากและปล่อย (Drag-and-drop) ตัวสร้างเสมือนใน Aglets Workbench จะอนุญาตให้เราตัวสร้างเสมือนนี้มีชื่อว่า Tazza ทำให้นักพัฒนาเอเยนต์สามารถสร้างเอเยนต์แอพพลิเคชันที่สมบรูณ์แบบในลักษณะของพฤติกรรมการและฟรอนต์เอ็นแบบเสมือนได้ง่ายขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเข้าถึงข้อมูลร่วมกัน

การเข้าถึงฐานข้อมูลที่ใช้ร่วมกัน จะมีความสำคัญต่อแอพพลิเคชันโมบายล์เอเยนต์มากมาย Aglets Workbench ได้เสนอแพคเก็จหลายตัวสำหรับการเข้าถึงข้อมูล รวมทั้ง JDBC/DB2 และ JoDax

โปรโตคอลการโยกย้ายเอเยนต์

โปรโตคอลการโยกย้ายเอเยนต์ (The Agent Tranfer Protocol : ATP) ถูกนำมาใช้ในการโยกย้ายเอเยนต์ข้ามเครือข่าย ซึ่ง ATP เป็นมาตรฐานโปรโตคอลระดับแอพพลิเคชันแก่ระบบข้อมูลแบบกระจายของเอเยนต์ มีความมุ่งหมายในการใช้งานบนอินเตอร์เน็ตและใช้ URL (Universal Resource Locators) บอกที่ตั้งทรัพยากรให้แก่เอเยนต์ ATP ได้เสนอรูปแบบและแพลตฟอร์มที่เป็นอิสระของโปรโตคอลเพื่อการโยกย้ายเอเยนต์ไปมาระหว่างเครือข่ายคอมพิวเตอร์

ในขณะที่โมบายล์เอเยนต์อาจจะถูกเขียนด้วยภาษาที่แตกต่างกันและตามความหลากหลายของระบบเอเยนต์ที่ระบุมาจากผู้ขาย ATP ได้เสนอโอกาสที่ดีเพื่อดูแลการเคลื่อนที่ของเอเยนต์ทั่ว ๆ ไปและอยู่ในรูปแบบเดียวกัน ตัวอย่างเช่น เครื่องโฮสต์ของเอเยนต์จะมีเพียงชื่อเดียวและเป็นหนึ่งเดียวเท่านั้น ไม่ขึ้นกับระบบเอเยนต์ของกลุ่มผู้ขายใด ๆ ที่มันสนับสนุนการทำงาน อีกทั้ง ATP ยังดูแลกลไกการขนส่งเอเยนต์ที่เป็นแบบเดียวกันและอนุญาตให้มีการกำหนดข้อซักถามที่ง่ายขึ้นเพื่อใช้งานผ่านทางเครือข่ายได้

เวอร์ชันแรกของ ATP มีชื่อว่า ATP/0.1 ซึ่งถูกสร้างขึ้นอย่างเป็นอิสระและเต็มไปด้วยแพคเก็จซึ่งมีเอกสารประกอบอยู่มากมายในเฟรมเวิร์ก ทั้งหมดเขียนด้วยจาวา ด้วยความสามารถสูงในการเคลื่อนย้ายของคลาสเหล่านี้ ได้เสนอมาตรฐานATP เพื่อสร้าง ATP deamon เชื่อมต่อไปยังไซต์ ATP และสร้างการร้องขอ-การตอบรับ ของ ATP ซึ่งแพคเก็จเหล่านี้จะไม่ขึ้นกับการสร้างเอเยนต์ใด ๆ เอเยนต์หนึ่งโดยเฉพาะ

ผู้จัดการเสมือน

Tahiti เป็นผู้จัดการเอเยนต์แบบเสมือนที่มีใน Aglets Framework โดยที่ Tahiti ใช้การติดต่อใช้งานแบบกราฟฟิค เพื่อตรวจตราดูการทำงานและควบคุมการทำงานของ aglets บนเครื่องคอมพิวเตอร์ผ่านการติดต่อลักษณะลากและปล่อย เราสามารถทำให้ aglets สองตัวทำการติดต่อสื่อสารกันถึงกันและกันได้ หรือส่ง aglet ไปยังไซต์ที่ระบุได้ Tahiti จะเป็นยิ่งกว่าเครื่องมือดูแลระบบมันเป็นเครื่องมือแบบตั้งโต๊ะไว้สำหรับผู้ใช้งานเอเยนต์ ทำนองเดียวกับการที่เว็บบราวเซอร์ได้กลายมาเป็นเครื่องมือพื้นฐานสำหรับเว็บยูสเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความสามารถของ Aglets บนเว็บ

เนื่องจาก เวิลด์ ไวด์ เว็บ นั้นเป็นรากฐานที่สำคัญมากสำหรับโมบายบ์เอเยนต์ เอเยนต์ที่พัฒนาบน Aglets Workbench สามารถฝังตัวในเว็บเพจ ซึ่งอนุญาตให้ผู้ขอรับบริการปล่อยเอเยนต์เหล่านี้ไปบนอินเตอร์เน็ตได้ในทันทีด้วยเว็บบราวเซอร์ที่สนับสนุนการทำงานของจาวา

ภายใน Aglets Workbench จะมีเอเยนต์ที่เป็นเว็บลันช์เชอร์ (Web Luncher) ซึ่งใช้ชื่อว่า Fiji รวมอยู่ด้วย โดยที่ Fiji นี้เป็นจาวาแอพเพ็ท ทีมีอยู่ใน Aglets Workbench และด้วยเหตุนี้ จึงมีความสามารถที่จะสร้าง aglets หรือถอน aglet ที่มีอยู่แล้วเข้าไปยังเว็บบราวเซอร์ของผู้ที่ขอรับบริการได้ แอพเพ็ท Fiji จะนำ URL ของเอเยนต์มาเป็นพารามิเตอร์และฝังลงในเว็บเพจได้อย่างง่ายดายด้วยการใช้ HTML คล้ายกับจาวาแอพเพ็ท และก็เช่นเดียวกับแอพเพ็ทอื่น ๆ ซอฟต์แวร์ทั้งหมดที่ต้องการ จะดาวน์โหลดแบบไดนามิกให้แก่บราวเซอร์เมื่อถึงคราวจำเป็นต้องใช้ ซึ่งทำให้ผู้ขอรับบริการไม่ต้องยุ่งยากทำการดาวน์โหลดและติดตั้ง Aglets Workbench เพื่อใช้เอเยนต์

หนทางให้อำนาจเว็บเพจด้วยโมบายล์เอเยนต์

เราสามารถให้อำนาจเว็บไซต์ด้วยโมบายล์เอเยนตได้ ถ้าเว็บเซิร์ฟเวอร์ถูกนำมาคู่กับ ATP daemon ให้สนับสนุน aglets แล้วจะทำให้แอพเพ็ท Fiji มีความสามารถที่จะส่ง aglets ไปที่เว็บไซต์ที่อยู่ห่างไกลเพื่อทำการค้นหา หรือ ปิดการตรวจตราการทำงานเว็บไซต์ ซึ่งเป็นการรวมกลุ่มของเว็บที่ให้บริการเอเยนต์ (Web Service Agents) ให้แก่เว็บไซต์ที่ใช้งาน aglets ได้

ความปลอดภัยของ Aglets จะเป็นอย่างไร

ความปลอดภัยเป็นส่วนที่มีความสำคัญมากต่อผู้ใช้งานโมบายล์เอเยนต์ ในขณะที่โมบายล์เอเยนต์ได้เสนออำนาจดีเยี่ยม ในส่วนที่สามารถใช้งานให้บรรลุผลสำเร็จมูลค่ามากมาย ในขณะเดียวกันนั้นมันสามารถกลับกลายเป็นแหล่งก่อตัวของซอฟต์แวร์ไวรัสได้เช่นกัน การรับเอเยนต์ที่ไม่รู้จักข้ามเครือข่ายเข้ามาเป็นความสามารถที่ซ่อนเร้นอยู่ภายในที่จะเปิดรับปัญหาทุกชนิดเข้ามาได้ทั้งหมด

โครงร่างของ Aglets สนับสนุนให้มีการขยายลำดับชั้นของรูปแบบการรักษาความปลอดภัย ในชั้นแรกของระบบรักษาความปลอดภัยมากจากระบบความปลอดภัยของภาษาจาวาเอง การแบ่งส่วนของโค้ดที่นำมาใช้ใน เอเยนต์นั้น มีจุดประสงค์ก็เพื่อจัดลำดับการตรวจสอบความถูกต้อง โดยเริ่มจาก ทดสอบให้แน่ใจว่าโค้ดที่ได้มามีรูปแบบถูกต้อง และจบท้ายด้วยลำดับของการตรวจสอบความถูกต้องโดยตัวพิสูจน์จาวาไบต์โค้ด

ในชั้นถัดมาจะเป็นผู้ดูแลรักษาความปลอดภัย ซึ่งอนุญาตให้ผู้ใช้โครงร่าง Aglets ทำการสร้างกลไกการป้องกันของตนเองขึ้นมา ความท้าทายที่เป็นเยี่ยมและความสำคัญสูงสุดในการสร้าง Tahiti เพื่อเป็นระบบที่รับประกันความปลอดภัยแก่เหล่าเอเยนต์ได้ Tahiti เพิ่มลักษณะผู้ดูแลความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปลอดภัยโดยจัดหาความปลอดภัยระดับสูงสำหรับเครื่องโฮสต์ของระบบและคอมพิวเตอร์ของผู้ที่เป็นเจ้าของ ซึ่งในตอนนี้ลักษณะการรักษาความปลอดภัยแบบนี้ยังมีข้อจำกัดอยู่มาก บ่อยครั้งที่ความพยายามของเอเยนต์ในการเข้าถึงไฟล์ ซึ่งการเข้าถึงข้อมูลนั้นจะไม่ยินยอมให้เอเยนต์ทำได้ เนื่องจากจะต้องคำนึงถึงความปลอดภัยเป็นอย่างมากและเอเยนต์จะไม่ได้รับอนุญาตให้เข้าถึงตามที่กำหนดไว้ได้

ในชั้นที่สามจะเป็นระบบรักษาความปลอดภัยของจาวา API นั้นคือเป็นโครงร่างอย่างง่ายสำหรับนักพัฒนาเอเยนต์เพื่อรวมฟังก์ชันการทำงานในการรักษาความปลอดภัยลงไปในตัวเอเยนต์ด้วย ฟังก์ชันการทำงานเหล่านี้ถูกนำมารวมกันเป็นรหัสลับด้วยวิธีการต่างๆ ได้แก่ การเข้ารหัสสัญญาณข้อมูลดิจิตอล (Digital Signatures),การเข้ารหัส (Encrytion) และการรับรองสิทธิ (Authentication)

มาตราฐานที่รับรอง

ATP และ โครงร่าง API ของ Aglets Workbench ได้ถูกนำเสนอต่อ OMG เพื่อเป็นข้อเสนอMobile Agent Facility ของ OMG

ในอนาคต รายละเอียดของ ATP/0.1 เป็นเอกสารที่ได้รับการเผยแพร่ไปทั่วอย่างไม่มีขีดจำกัด ด้วยโปรโตคอลที่ถูกออกแบบมาเหนือกว่า HTTP และประกอบกับแนวทางมาตราฐานของการเคลื่อนย้ายเอเยนต์ที่เป็นอิสระ ไม่ขึ้นกับระบบใดๆ ระบบหนึ่งโดยเฉพาะ

การทำให้ Aglets เป็นที่นิยมอย่างแพร่หลาย

ความสำคัญสูงสุดคือการทำให้ Aglets ถูกใช้งานอย่างแพร่หลาย โดยที่สามารถดาวน์โหลดได้ที่ไซต์ในอเมริกาและญี่ปุ่น โดยไม่เสียค่าธรรมเนียม จะมีการดำเนินความพยายามเพื่อที่จะหล่อหลอม Aglets Workbench เข้ากับเซิร์ฟเวอร์ข้อมูลข่าวสาร เช่น เว็บเซิร์ฟเวอร์และดาต้าเบสเซิร์ฟเวอร์ เพื่อจะได้ถึงจุดมุ่งหมายของการทำให้ Aglets เป็นที่ใช้กันอย่างแพร่หลาย

สรุป

Aglets Workbench จาก IBM นั้นได้เสนอรูปแบบของสภาพแวดล้อมเสมือนเพื่อสร้างแอพพลิเคชันบนเครือข่ายที่โมบายล์เอเยนต์สามารถใช้งานได้ ซึ่งมีแพคเก็จต่าง ๆ ดังนี้ คือ

- โครงร่างของ Aglets (Aglets Framework) สำหรับโมบายล์เอเยนต์
- โปรโตคอลการเคลื่อนย้ายเอเยนต์ (เอเยนต์ Transfer Protocol : ATP)
- ตัวสร้างเอเยนต์เสมือน (Visual เอเยนต์ builder) ใช้ชื่อว่า " Tazza "
- ฐานข้อมูล (JDBC) สำหรับ DB2
- JoDex การเข้าถึงข้อมูล
- ตัวจัดการเอเยนต์เสมือน (Visaul Agent manger) ใช้ชื่อว่า " Tahiti "

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- เว็บสำหรับปล่อยเอเยนต์ (Agent web launcher) ใช้ชื่อว่า " Fiji "

Aglets Workbench จัดหาแนวทางที่เป็นหนึ่งเดียวและมีความสามารถสูงและทำให้มีพาราไดม์ซึ่งรวมลักษณะต่าง ๆ ดังต่อไปนี้

- อ็อบเจคที่มีการเคลื่อนที่ (Mobile objects) และ อ็อบเจคที่มีที่อยู่ประจำ (Stationary objects)
- การส่งผ่านทั้ง อ็อบเจค,เมสเสจ และข้อมูล
- ความเป็นอิสระ(Autonomous) และ Passive objects
- กระบวนการทำงานแบบอะซิงโครไนซ์และแบบซิงโครไนซ์
- โลคอลอ็อบเจคและรีโมตอ็อบเจค
- ปฏิบัติงานเมื่อ ปิด/เปิด การเชื่อมต่อเครือข่าย
- การประมวลผลการทำงานแบบขนาน (Parallel) และแบบตามลำดับ (Sequential)

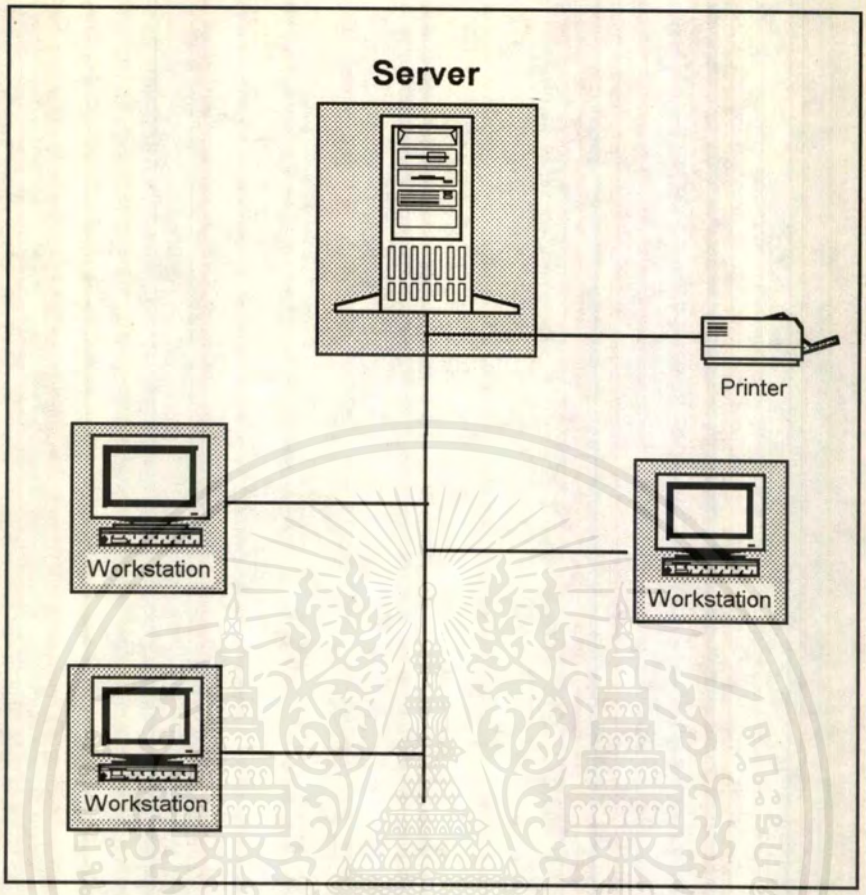
2.4 Technical Agent Issues

2.4.1 การประมวลผลแบบกระจาย (Distributed Programming)

กระบวนการทำงานแบบไคล์เอ็นต์/เซอร์ฟเวอร์ (Client/Server Processing)

ลักษณะการทำงานแบบไคลเอ็นต์เซอร์ฟเวอร์นี้ แอพพลิเคชันจะทำงานทั้งในเครื่องเซอร์ฟเวอร์และเครื่องเวอร์กสเตชันด้วย เมื่อผู้ใช้ต้องการข้อมูลบางส่วนที่อยู่ในดาต้าเบสไฟล์ขนาดใหญ่ เครื่องเวอร์กสเตชันจะส่งคำร้องขอข้อมูลนั้นไปที่ตัวเซอร์ฟเวอร์ เซอร์ฟเวอร์จะทำการคำนวณและดึงข้อมูลเหล่านั้นออกมา และส่งเฉพาะข้อมูลนั้นๆ กลับมายังเครื่องเวอร์กสเตชันที่ร้องขอมา เมื่อเครื่องเวอร์กสเตชันได้รับข้อมูลแล้วมันก็จะนำมาคำนวณต่อไป ซึ่งจะเห็นได้ว่า ข้อมูลที่ส่งในระบบนี้จะส่งเฉพาะข้อมูลที่จำเป็นเท่านั้น ดังนั้นจึงเป็นข้อดีที่เราไม่ต้องกังวลในเรื่องของ traffic ที่จะเกิดขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

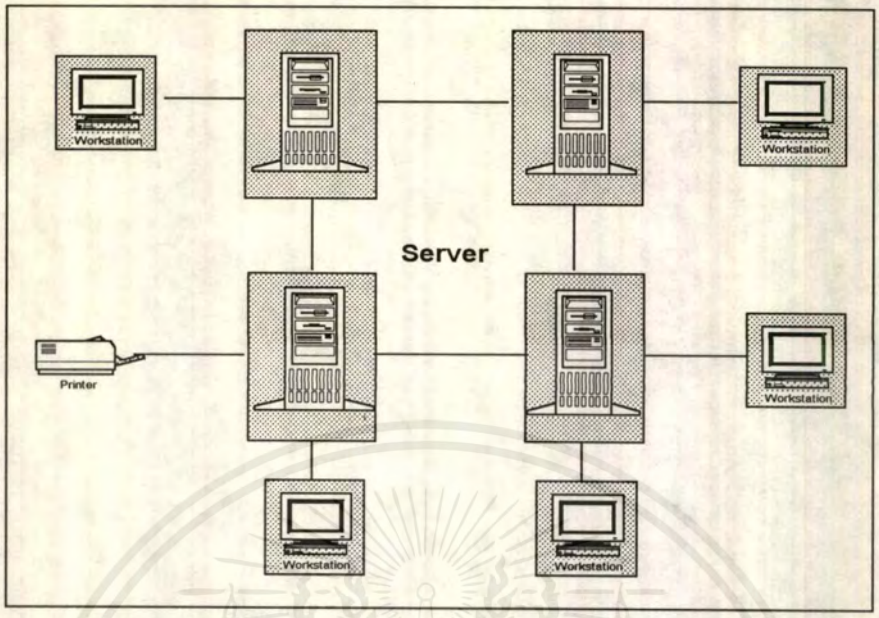


รูปที่ 17 การทำงานแบบ ไคล์เอ็นต์/เซอร์ฟเวอร์

กระบวนการทำงานแบบเพียร์ทูเพียร์(Peer-to-Peer Processing)

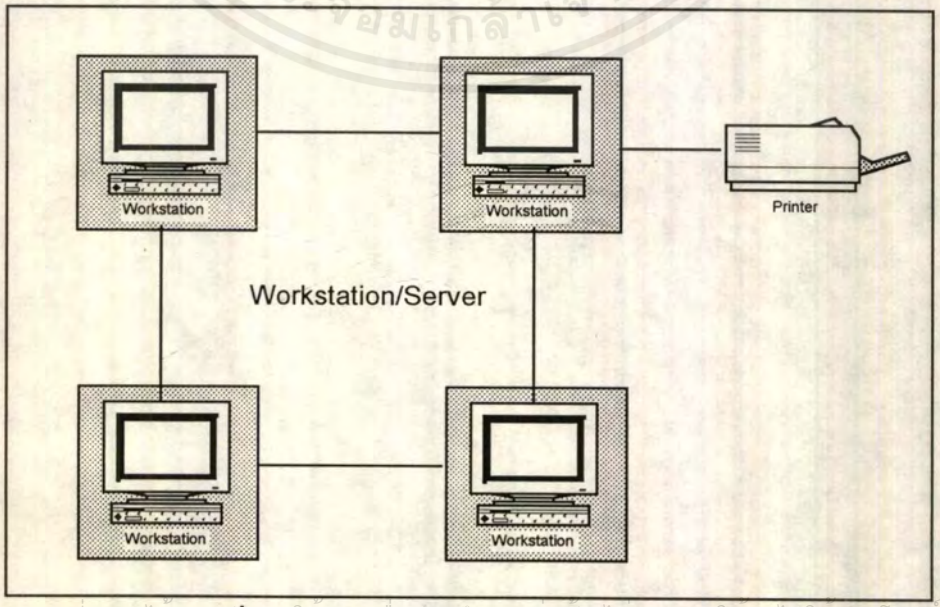
การทำงานจะถูกแบ่งออกเป็น 2 แบบ แต่หลักทำงานของมันจะใกล้เคียงกันมากและมีพื้นฐานการทำงานมาจาก ไคล์เอ็นต์/เซอร์ฟเวอร์ Processing นั่นเอง แทนที่จะมีแต่ผู้ใช้ที่ทำงานอยู่ในเครื่องเวอร์กสเตชั่น ที่จะร้องขอการบริการจากเครื่องเซอร์ฟเวอร์เท่านั้นตัวเซอร์ฟเวอร์เองก็สามารถร้องขอการบริการได้ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 18 ลักษณะการทำงานแบบ เพียร์-ทู-เพียร์ แบบที่ 1

จากรูปที่ 18 เซอร์ฟเวอร์แต่ละตัวจะต่อเชื่อมถึงกันและสามารถร้องขอข้อมูลจากเซอร์ฟเวอร์ตัวอื่นๆ ได้ผู้ใช้ไม่จำเป็นต้องรู้เลยว่าข้อมูลที่ตนต้องการนั้นอยู่ในเครื่องเซอร์ฟเวอร์ตัวใด เมื่อมันร้องขอข้อมูลไป เครื่องเซอร์ฟเวอร์แต่ละตัวจะทำหน้าที่เหมือนเป็นไคล์เอ็นต์และเซอร์ฟเวอร์สลับกันไปมาเพื่อค้นหาข้อมูลที่ต้องการ เมื่อได้มาแล้วมันก็จะทำตัวเป็นเซอร์ฟเวอร์อีกครั้งเพื่อส่งข้อมูลกลับไปให้เวอร์กสเตชันไคลเอ็นต์ที่ร้องขอมา



รูปที่ 19 ลักษณะการทำงานแบบ เพียร์-ทู-เพียร์ แบบที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากการทำงานดังรูปที่ 19 เวอร์กสเตชันแต่ละตัวจะทำหน้าที่ได้ทั้งเซอร์ฟเวอร์และไคล์เอ็นต์ ซึ่งหมายความว่า แต่ละตัวสามารถทำการร้องขอและให้บริการตัวอื่นๆ ได้ในเวลาเดียวกัน ดังนั้นในระบบเช่นนี้จะเป็นการร้องขอการบริการกันโดยตรงทั้งสองทางแทนที่จะต้องผ่านการร้องขอไปให้เครื่องเวอร์กสเตชันตัว ๆ อื่น

2.4.2 Distributed Architecture for Intelligent Information Processing and Problem Solving :

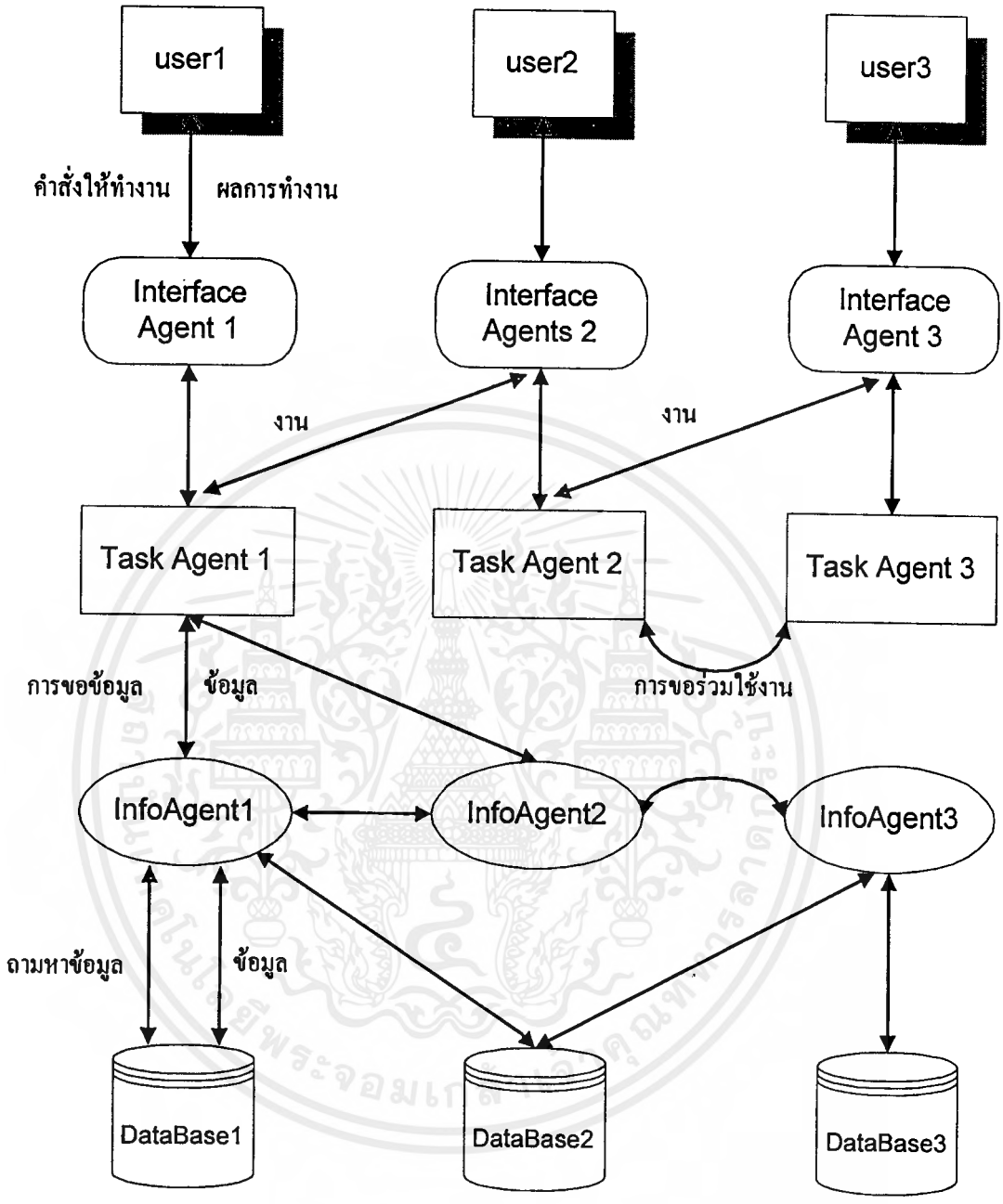
เหตุผลที่ต้องออกแบบสถาปัตยกรรมเอเยนต์ให้เป็นแบบ Distributed Architecture สำหรับการทำงานของเอเยนต์ในปัจจุบัน

- Distributed Information Source :
- Sharability :
- Complexity Hiding :
- Modularity and Reusability :
- Flexibility :
- Robustness :
- Quality of Information :
- Lagacy Data :

ข้างต้นเป็นเหตุผลที่ทำให้ต้องมีการพัฒนาระบบที่เป็น Distributed Software Agents เพื่องานรวบรวมข้อมูลและการช่วยงานต่างๆบนสภาวะแวดล้อมของข้อมูลต่างๆบน Internet แต่มีปัญหาที่สำคัญว่า จะสร้างโครงสร้างและจัดการกับระบบที่มีส่วนการทำงานหลายๆส่วนประกอบกันได้อย่างไร คำตอบก็คือ การออกแบบให้มีเอเยนต์ในทำงานและรับผิดชอบงานแต่ละส่วนแตกต่างกันไป โดยต้องมีการวางแผนในเอเยนต์เหล่านั้นทำงานสอดคล้องกันได้เป็นอย่างดี ยิ่งระบบมีส่วนประกอบในการทำงานมากเท่าไหร่ก็ควรจะมีเอเยนต์รับผิดชอบมากขึ้นเท่านั้น

ฉะนั้นเราจึงสามารถแยกประเภทของเอเยนต์ตามการทำงานในระบบได้ดังนี้ เพื่อการมองเห็นภาพรวมของระบบได้ดีขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 20 สถาปัตยกรรมของระบบเอเยนต์ที่มีเอเยนต์มากกว่า 1 ตัว

- **Interface Agents :**

มีหน้าที่ตอบโต้กับผู้ใช้ คือทำการรับคำสั่งข้อมูลจากผู้ใช้ และแสดงผลการทำงานให้ผู้ใช้ ทราบ โดยในตัว Interface Agents นี้จะทำการเก็บข้อมูลมา, ออกแบบการทำการติดต่อในขั้นตอนต่างๆ และจัดการประยุกต์หน้าที่ของมันเข้ากับสิ่งที่ผู้ใช้ ต้องการ ซึ่งเอเยนต์จะมีข้อมูลเกี่ยวกับการทำงานของมันที่เกี่ยวกับผู้ใช้ นั้นๆเพื่อให้ความสะดวกกับผู้ใช้ มากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไปอีกเหมือนมีเลขานุการส่วนตัว ตัวอย่างของเอเยนต์ประเภทนี้ เช่น เอเยนต์ที่ทำหน้าที่กลั่นกรองจดหมายทางอิเลคทรอนิกส์ (Electronic Mails) ก็จะมีการตอบโต้หรือการแสดงภาพทางหน้าจอหรือวิธีการที่ใช้ติดต่อกับผู้ใช้ แตกต่างไปแต่ละผู้ใช้

ฟังก์ชั่นของ Interface Agents :

- รวบรวมข้อมูลจากผู้ใช้ เพื่อกำหนดแนวทางการทำงาน
- เสนอข้อมูลต่างๆให้กับผู้ใช้ ไม่ว่าจะเป็นผลลัพธ์การทำงานหรือคำอธิบายต่างๆ
- สอบถามผู้ใช้ ถึงข้อมูลเพิ่มเติมในระหว่างการทำงาน เพื่อให้ทำงานได้ผลลัพธ์ตรงกับที่ต้องการจริงๆ
- สอบถามการยืนยันการตัดสินใจจากผู้ใช้ ในบางข้อบางกรณี

ข้อมูลความรู้ที่ควรจะมีใน Interface Agents :

- ต้องนำเสนออะไรแก่ผู้ใช้ และเมื่อใดที่ต้องนำเสนอ
- รูปแบบของการติดต่อกับผู้ใช้
- หน่วยทำงานย่อยๆที่อาจต้องมีการติดต่อกับผู้ใช้
- ข้อกำหนดต่างๆในการติดต่อกับ Task Agents

- **Task Agents :**

มีหน้าที่รับเอาข้อมูลที่ได้จาก Interface Agents มาเพื่อทำงานให้สำเร็จตามเป้าหมายของระบบ โดยเอเยนต์นี้จะมีแผนการดำเนินงานเพื่อรองรับคำสั่งในรูปแบบต่างๆของผู้ใช้ ซึ่งการทำงานนี้อาจจะเป็นการประมวลผลภายในตัวเอเยนต์เองหรืออาจมีการแลกเปลี่ยนและขอใช้ข้อมูลจากแหล่งต่างๆด้วย ซึ่งแหล่งต่างๆเหล่านั้นก็คือ เอเยนต์อีกประเภทที่ทำหน้าที่ดูแลข้อมูลนั่นเอง

ฟังก์ชั่นของ Task Agents :

- รับคำสั่งจาก Interface Agents ถึงสิ่งที่ผู้ใช้ ต้องการ
- นำคำสั่งเหล่านั้นแปลออกมาเป็นขั้นตอนสิ่งที่ต้องทำ
- แยกขั้นตอนต่างๆเหล่านั้น และให้แต่ละขั้นวางแผนการทำงาน
- แต่ละขั้นแต่ละส่วนทำงานต่างๆของตน โดยที่ทุกส่วนมีการเชื่อมการทำงานถึงกัน เพื่อให้ข้อมูลของการทำงานเป็นไปอย่างถูกต้อง

ข้อมูลความรู้ที่ควรจะมีใน Task Agents :

- รูปแบบของการทำงานที่ Agent ถูกออกแบบมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ต้องใช้อะไรบ้างในการทำงานดังกล่าว
- การรวบรวมข้อมูลที่จำเป็นสำหรับการทำงานต่างๆ
- แผนการดำเนินงานต่างๆ ในกรณีที่แตกต่างกันออกไป
- แผนการดำเนินงานต่างๆเมื่อเกิดปัญหาต่างๆ
- รู้จัก เอเยนต์ ตัวอื่นๆ ที่ถูกออกแบบมาให้ทำงานคล้ายกันหรือทำงานส่วนใดๆ ที่ให้ผลลัพธ์ออกมาเป็นประโยชน์กับมัน
- ข้อกำหนดต่างๆสำหรับการติดต่อกับเอเยนต์ต่างๆที่เกี่ยวข้อง

- **Information Agents :**

มีหน้าที่ดูแลข้อมูลที่ตนรับผิดชอบและคอยจัดนำข้อมูลเหล่านั้นมาให้บริการกับ Task Agents ให้สามารถมีการใช้ข้อมูลอย่างมีประสิทธิภาพ ซึ่งข้อมูลจากแหล่งต่างๆนั้นก็ไม่จำเป็นต้องเป็นข้อมูลประเภทเดียวกัน รูปแบบเดียวกัน ดังรูปที่ 20 Task Agents จะหาสอบถามไปยัง Information Agents ต่างๆถึงข้อมูลที่ต้องการ Information Agents ก็จะค้นหาข้อมูลที่ตรงกันนั้นในฐานข้อมูลที่ตัวเองดูแลอยู่ แล้วจึงส่งกลับไปยัง Task Agents เพื่อทำการประมวลต่อไป นอกจากนี้ Information Agents ยังต้องมีการตรวจสอบดูการเปลี่ยนแปลงของฐานข้อมูลนั้นๆด้วย

ข้อมูลความรู้ที่ควรจะมีใน Information Agents :

- รูปแบบและข้อมูลลักษณะของข้อมูลที่เอเยนต์รับผิดชอบดูแลอยู่ ไม่ว่าจะเป็นขนาดของข้อมูล, เวลาเฉลี่ยในการทำงานกับข้อมูลต่อ 1 คำร้อง เป็นต้น
- วิธีในการทำงานติดต่อกับฐานข้อมูลนั้น
- แผนการแก้ปัญหาต่างๆ
- ข้อกำหนดต่างๆสำหรับการติดต่อกับเอเยนต์ต่างๆที่เกี่ยวข้อง

<u>2.4.3 การบริหารระบบการทำงาน แบบมัลติทาส์กกิ้ง</u>

เมื่อหลายปีก่อน ระบบมัลติโพรเซสซิ่ง (Multiprocessing system) ซึ่งเป็นระบบการทำงานที่อนุญาตให้เราทำการรันโปรแกรม หรือรันโพรเซส ได้ครั้งละหลาย ๆ ตัวในเวลาเดียวกัน หลังจากนั้นไม่นาน เริ่มมีระบบปฏิบัติการจำนวนมากได้เริ่มทำการสนับสนุนการทำมัลติเธรดดิ้ง (Multithreading) ซึ่งยอมให้โปรแกรมเมอร์เขียนโปแกรมให้มีการสร้างโพรเซสย่อย (Subprocess) หรือ เธรด (threads) เอาไว้ภายในโพรเซสแต่ละตัวได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เธรดมีลักษณะแตกต่างจากโพรเซส ในระบบที่ไม่มีการทำมัลติเธรดดิ้งจะมีโพรเซสและโพรเซสที่สัมพันธ์กันเท่านั้น ซึ่งโพรเซสที่สัมพันธ์กันนี้สามารถเปิดไฟล์เพื่อใช้งานร่วมกันเท่านั้น ในระบบมัลติเธรดดิ้ง แต่ละโพรเซสถูกสร้างมาจากเธรดเป็นจำนวนตั้งแต่ 1 ถึง n ตัว และเธรดทั้งหมดที่อยู่ในโพรเซสสามารถใช้งานกับตัวแปรร่วมกันได้ อย่างที่รู้จักกันเหมือนอย่างการจัดการเปิดไฟล์ ซึ่งการใช้ตัวแปรร่วมกันเช่นนี้ทำให้เป็นเรื่องง่ายมากสำหรับเธรด ที่จะร่วมทำงานด้วยกัน โดยพวกมันจะเอาใจใส่อยู่อย่างเดียวเฉพาะกับงานที่พวกมันได้รับมอบหมาย

เพื่อที่ระได้รับสิ่งที่ดีกว่าจากการยกระดับของการโปรแกรมการทำงานจากแบบเธรดเดียวมาเป็นแบบมีเธรดหลาย ๆ ตัว ภาพบริษัทที่ลูกจ้างที่ทำงานกับบริษัทมีตึกทำงานเป็นของตัวเอง (สมบรูณ์พร้อมด้วย ระบบโทรศัพท์ ห้องน้ำ โรงอาหาร อุปกรณ์สำนักงาน และ สถานที่จอดรถมากพอ) ลูกจ้างแต่ละคนสามารถจะทำงานของเขาได้ยาวนาน เท่าที่ทุกสิ่งทุกอย่างที่เขาต้องการอยู่ในสำนักงานของเขา เป็นเงื่อนไขที่บริษัทที่ต้องคำนึงเป็นอย่างมาก งานใดๆ ที่ต้องเข้าถึงการทำงานของลูกจ้างแต่ละคนจะทำให้มีความยุ่งยากมากมายและต้องทำการติดต่อสื่อสารกันเป็นอย่างมาก ถ้าเจ้านายต้องการให้ทำอะไรเขาก็จะยกหูโทรศัพท์ นี่คือ ลักษณะการทำงานแบบ หนึ่งโพรเซสที่มีเธรดตัวเดียว ถึงตอนนี้ หากเราจะนำลูกจ้างทั้งหมดมาอยู่ในสำนักงานเดียวกัน เราจะลดความยุ่งยากในต่อการติดต่อสื่อสารน้อยลงในทันทีและเพิ่มประสิทธิภาพในการใช้ประโยชน์ทรัพยากรมากขึ้น ลักษณะเช่นนี้เป็นการทำงานแบบมีโพรเซสเดียวที่มีเธรดหลายตัว

จาวาเป็นภาษาที่มีมัลติเธรดดิ้งแฝงอยู่ด้วย เป็นต้นว่า มันจะขึ้นอยู่กับระบบปฏิบัติการที่สนับสนุน เธรดและการทำมัลติเธรดดิ้ง นี่คืออีกเหตุผลหนึ่งที่ไม่ใช่สำหรับจาวา ตามคำกล่าวข้างต้นนั้นถูกส่งไปให้กับระบบปฏิบัติการที่เป็นที่นิยมอย่างกว้างขว้างคือ ระบบปฏิบัติวินโดวส์ 3.1 ตามความเป็นจริงแล้ว หากมองแค่เพียงว่า จาวาจะทำมัลติเธรดดิ้งอย่างไรนั้น เราเพียงแค่มองไปที่อ๊อบเจค ซึ่งเป็นคลาสพื้นฐานสำหรับทุก ๆ คลาสในจาวา เมื่ออ๊อบเจคจำนวน 6 ตัวมีการใช้เมธอดที่ประกาศเป็นของส่วนกลาง 12 เมธอด ทำให้มีการควบคุมดูแลเธรด และการติดต่อสื่อสารกันภายในเธรดด้วย

2.4.4 Mobile Code Design

Design dimension 1 : Instruction Set Agent Language

ในการที่เราจะส่ง โมบายล์โค้ด (Mobile Code) ผ่านเครือข่ายจะเป็นจริงได้ ก็ต่อเมื่อ โค้ดที่ส่งไปนั้นสามารถทำการเอ็กซีคิ้วได้จริงที่อีกฝั่ง ซึ่งการส่งในลักษณะของคำสั่งภาษาเครื่อง (Machine Code) นั้นไม่ใช่ทางเลือกที่ดี เพราะจะมีแค่เพียงเครื่องที่สามารถเข้าใจการทำงานที่ตรงกับภาษาเครื่องที่ส่งมาเท่านั้น จึงจะสามารถเอ็กซีคิ้วได้ นี้คือเหตุผลที่บอกว่า ทำไมโค้ดของโม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บายล์เอเยนต์ จะต้องไม่ขึ้นกับรูปแบบของหน่วยประมวลผลใด ๆ แนวทางมาตรฐานคือ ออกแบบคอมพิวเตอร์จำลอง(ชุดคำสั่งการทำงาน) และนำตัวแปลภาษา (Interpreter) ไปไว้ในโฮสต์ทุก ๆ เครื่องที่ทำงานและทดลองให้มีการเคลื่อนที่ เอเยนต์จะถูกระบุในชุดคำสั่งการทำงานที่เป็นกลาง

การออกแบบ ชุดคำสั่งการทำงาน จะจำเป็นอย่างมากที่ไม่ควรเป็นระดับต่ำเพื่อให้ใช้กับหน่วยประมวลผลโดยตรง มีระบบเอเยนต์จำนวนมาก ที่ใช้การโปรแกรมด้วยภาษาระดับสูงเป็นชุดคำสั่งการทำงาน ซึ่งจะทำให้สามารถรองรับการทำงานกับสภาพแวดล้อมการทำงานได้หลากหลายเพิ่มขึ้นทำนองเดียวกับโครงสร้างซอฟต์แวร์ของอ็อบเจคโอเรียลเต็ท อาจจะเป็นไปได้ที่เราจะนำภาษาสคริปต์ที่มีอยู่แล้วมาใช้ เช่น Tcl และ เพื่อสามารถนำกลับมาใช้ได้อีกในโดเมนเอเยนต์บางระบบจะใช้ภาษาทั้งสองระดับ ที่ระดับต่ำ จะมีชุดคำสั่งการทำงานอย่างง่าย ที่ใช้ชุดคำสั่งจำลองที่สมมติไว้เครื่องกับคอมพิวเตอร์ และในขณะเดียวกันจะมีภาษาระดับสูงที่เป็นอ็อบเจคโอเรียลเต็ทซึ่งจะถูกใช้งานได้โดยตรงจากโปรแกรมเมอร์ ส่วนใหญ่แล้วจะมีเฉพาะชุดคำสั่งของภาษาระดับต่ำที่จะถูกส่งข้ามเครือข่าย โดยจะมีโปรแกรมคอมไพล์เลอร์ชนิดพิเศษไว้คอยทำการแปลโค้ดของโปรแกรมเมอร์ที่เขียนไว้ให้อยู่ในรูปไบต์โค้ดระดับต่ำ

Design dimension 2 : Host Security

สิ่งแรกที่เราคาดหวังไว้ได้คือ สร้างความน่าเชื่อถือและไว้วางใจได้แก่รากฐานโครงสร้างและกำหนดข้อบังคับคอยควบคุมว่าให้ใครบ้างที่สามารถรับการติดต่อกับโมบายล์เอเยนต์ได้ หมายความว่า ตัวโมบายล์เอเยนต์แต่ละตัวจะมีจำกัดสิทธิโดยระบบเมื่อเดินทางมาถึงโดยจะมีการตรวจสอบแหล่งที่มาของมันและการร้องขอของบุคคลที่สาม เพื่อยืนยันถึงความปลอดภัยของมัน เราจะสังเกตได้ว่า นี่คือลักษณะการจัดตั้งข้อบังคับโดยวิธีทางเทคนิคเป็นส่วนใหญ่ และส่วนกรณีของความผิดพลาดที่เกิดจากการดูแลการทำงาน ยังไม่สามารถป้องกันได้เช่นกัน นี้คือคำตอบว่าทำไมบางทางออก จึงเลือกที่จะไม่สร้างระบบความปลอดภัยไว้มากเกินไปกับการเอ็กซีคิ้วและการเคลื่อนย้ายของโมบายล์เอเยนต์ แต่จะเลือกออกแบบกับส่วนอื่นๆ แทน เช่น ดูแลในเรื่องของทรัพยากร

อย่างไรก็ตาม ถ้าชุดคำสั่งการทำงานมีอยู่ในชุดคำสั่งพิเศษที่ทำให้ ทรัพยากรภายในโฮสต์สามารถนำไปใช้ประโยชน์ให้แก่เอเยนต์ได้ หรือยอมให้เอเยนต์เอ็กซีคิ้วโปรแกรม พวกเขาสามารถดูแลเข้มงวดกับการเข้าใช้งานได้ด้วยการใช้รหัสผ่าน (Password) ก็ย่อมทำได้

Design dimension 3 : Agent Security

การดำเนินการป้องกันเฉพาะที่โฮสต์เพียงอย่างเดียวยังไม่เพียงพอสำหรับความปลอดภัย เพราะว่า ตัวโมบายล์เอเยนต์เอง ก็ยังเป็นเป้าหมายของการทำอันตรายได้มากมายเช่นกัน
ส่วนที่ยากมากที่สุด คือ ความเป็นส่วนตัวของเอเยนต์ ซึ่งจำเป็นอย่างยิ่งเมื่อเอเยนต์ต้องพกพาเงินที่ใช้ในโลกอิเลคทรอนิกส์ หรือข้อมูลอื่น ๆ ที่เป็นความลับ ตามหลักแล้วเอเยนต์จะต้องป้องกันการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปะทะอย่างมากมายจากโฮสต์และผู้ไม่หวังดีที่ต้องการจะขโมย สิ่งของนั้น นั้นคือทำให้การให้อำนาจจะต้องทำขึ้นในสองทาง คือ ให้ใช้ได้มีเฉพาะโฮสต์ที่มีการมอบอำนาจแล้วเท่านั้นที่จะสามารถเอ็กซีคิ้วเอเยนต์ได้ แต่อย่างไรก็ตามประการนี้ก็ยังไม่สามารถป้องกันอุบัติเหตุในความน่าเชื่อถือของรากฐานโครงสร้างได้

Design dimension 4 : Name Space Conventions

เมื่อมีการนำโฮสต์มาอยู่ร่วมกันแล้วนั้น การที่โฮสต์เหล่านั้นสารถเข้าใจคำสั่งการทำงานของเอเยนต์ ได้อย่างตรงกันยังไม่เพียงพอ ยิ่งไปกว่านั้นเราจะต้องตั้งแบบแผนข้อกำหนดเกี่ยวกับการใช้งานในส่วนการตั้งชื่อเพื่อเรียกใช้ในการทำงาน อีกทั้งบริการต่าง ๆ ที่จัดเสนอให้แก่เอเยนต์นั้นควรจะมีรูปแบบแนวทางที่ครอบคลุมว่า จะต้องแสดงอะไรบ้าง และจะเข้าใช้งานได้อย่างไรเพื่อให้เอเยนต์สามารถใช้งานได้

Design dimension 5 : Resource Control

การดูแลการใช้งานทรัพยากร เป็นสิ่งที่ใช้ในการปฏิบัติเพื่อให้การทำงานของระบบเอเยนต์เป็นไปอย่างราบรื่นและมีลำดับขั้นตอน ซึ่งการดูแลในเรื่องนี้เป็นเรื่องที่ซับซ้อนมากรวมทั้งเป็นงานที่ยากมากพอสมควร โดยเฉพาะอย่างยิ่งเมื่อต้องมีการกำหนดให้ใช้งานเป็นช่วงจำนวนเวลาเข้ามาเกี่ยวข้องด้วย

Design dimension 6 : Programing support and Agent Steering

มีลักษณะคล้ายกับระบบกระจายการทำงาน ที่มีอยู่หลายระบบ สำหรับแอพพลิเคชั่นที่สร้างมาเพื่อใช้เป็นพื้นฐานของระบบการทำงานของเอเยนต์นั้น จำเป็นอย่างมากสำหรับการจับตามองการทำงานของระบบ (Monitoring) ที่เราต้องทำได้ การย้อนรอยเพื่อค้นหาที่มาของปัญหาได้ (Tracing) การตรวจสอบหาที่ผิดพลาดจากการทำได้ (Debugging)

Design dimension 7 : Efficieny

เป็นส่วนที่เกี่ยวข้องในเรื่องของประสิทธิภาพการทำงาน ซึ่งเป็นสิ่งที่มีอิทธิพลอย่างมากต่อการออกแบบระบบเอเยนต์ โค้ดที่สามารถเคลื่อนย้ายที่สำหรับการเอ็กซีคิ้วได้นั้นนับว่ามีราคาสูงมากในเทอมของความสามารถของคอมพิวเตอร์ ทำให้เกิดสถานะของการทำงาน การนำมาอยู่ร่วมกันเป็นแพตเก็จและส่งออกไป การให้อำนาจแก่เอเยนต์ที่มีเข้ามา การติดตั้งตัวที่จะทำการแปลการทำงานของเอเยนต์ และทำการแปลแต่ละคำสั่งที่ประกอบมาเป็นส่วนเกิน ในการออกแบบแต่ละครั้งจะต้องคิดประเมินเพื่อเป็นการลดค่าใช้จ่ายให้น้อยลง เนื่องจากว่าเอเยนต์ยังอาจใช้อีกเวลานานที่ยังคงถูกจำกัดเกี่ยวกับแอพพลิเคชันที่มีควาทเหมาะสมกับสภาพการทำงานของเอเยนต์อยู่อีกมาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งระบบไคลเอ็นต์-เซอร์ฟเวอร์ ก็ยังไม่มีความเหมาะสมเพียงพอ หรือยังต้องมีความซับซ้อนมากกว่านี้ด้วย

2.4.5 ระบบความปลอดภัยของเอเยนต์ (Agent Safety and Security)

เนื่องจากเอเยนต์เป็นแค่เพียงโปรแกรมคอมพิวเตอร์ แต่มีการดูแลในเรื่องระบบรักษาความปลอดภัยของมัน จะมีข้อแตกต่างจากแอพพลิเคชันโปรแกรมอื่น ๆ อย่างมาก ที่เป็นเช่นนี้มีสาเหตุมาจาก โครงสร้างพื้นฐานของเอเยนต์นั้นอ้างอิงเกี่ยวกับกระบวนการเคลื่อนย้ายโปรแกรมและข้อมูล เพื่อทำงานกับระบบคอมพิวเตอร์ที่แตกต่างกันและต้องมีการเริ่มต้นการทำงานด้วยตัวเองในแต่ละครั้งที่มันทำงาน ซึ่งลักษณะต่าง ๆ เช่นนี้หากไม่มีการควบคุมเรื่องของระบบรักษาความปลอดภัยให้รัดกุม อาจนำมาซึ่งผลเสียหายอย่างมากมายได้

การควบคุมเอเยนต์ในใช้งานทรัพยากร

ถึงแม้ว่าการทำงานในระบบเอเยนต์ จะต้องการลักษณะความเป็นมิตร (Friendly) ก็ตาม แต่ในระบบส่วนมากยังจำเป็นต้องมีอำนาจดูแลเกี่ยวกับ ข้อปฏิบัติดูแลการเข้าถึง (Access Control) ข้อมูลของเอเยนต์ในระบบที่แตกต่างกันตามแต่ชนิดของเอเยนต์นั้น ๆ เป็นต้นว่า ในระบบที่เก็บข้อมูลเกี่ยวกับการจองตั๋วเครื่องบิน ควรที่จะต้องให้ข้อมูลรายละเอียดในเรื่องของ เที่ยวบินที่สามารถจองได้แก่เอเยนต์เพื่อเอเยนต์จะได้นำผลลัพธ์กลับไปให้ผู้ใช้ แต่ไม่จำเป็นที่ต้องให้ข้อมูลส่วนนี้แก่เอเยนต์ตัวอื่น ๆ ขณะเดียวกัน เอเยนต์ที่จะรับข้อมูลจากระบบได้ก็ต้องเป็นเอเยนต์ที่มีการกำหนดสิทธิเรียกใช้บริการ ที่เอเยนต์ได้รับอนุญาตให้ช้งานได้

การดูแลความปลอดภัยการใช้งานเอเยนต์ ในหัวข้อนี้จะมุ่งไปยังเรื่องของความเสี่ยงที่เกิดจากการไม่มีส่วนร่วมและไม่ยอมรับ ให้ทำงาน

แนวคิดเกี่ยวกับวิธีการป้องกัน

ปกติในระบบคอมพิวเตอร์ที่ไม่มีการทำงานกับเอเยนต์ กลไกด้านการรักษาความปลอดภัยจะหมายเน้นไปในเรื่องของการป้องกันระบบคอมพิวเตอร์และผู้ใช้ ในส่วนของข้อมูลและโปรแกรมของผู้ที่ใช้งานบนระบบให้ปลอดภัยจากการกระทำของผู้ใช้คนอื่น ๆ ทั้งที่กระทำโดยเจตนาและไม่เจตนา เมื่อมีการนำโมบายล์เอเยนต์เข้ามาใช้งานในระบบคอมพิวเตอร์จึงต้องมีการป้องกันส่วนต่าง ๆ ที่เกี่ยวข้องกับการทำงานที่มีอยู่จากผู้ที่ริเริ่มสั่งงานเอเยนต์และเอเยนต์ที่ทำงานไม่เว้นแม้แต่เครื่องที่โมบายล์เอเยนต์ต้องไปทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในระบบเอเยนต์ที่ปลอดภัยนั้น จำเป็นอย่างยิ่งที่ต้องป้องกัน ผู้ที่เริ่มต้นสั่งงานเอเยนต์และเอเยนต์ที่รับหน้าที่ทำงาน จากส่วนต่าง ๆ ที่เกี่ยวข้องกับการทำงานและต้องป้องกันผู้เริ่มสั่งงานจากเอเยนต์ของเขาเองด้วย

อันตรายที่เกิดจากการทำงาน ของ เอเยนต์/ผู้สั่งงาน ที่เกิดขึ้น

- เอเยนต์ที่ไม่มีอำนาจทำงาน เนื่องจากผู้สั่งงานที่ไม่ได้รับสิทธิที่จะสั่งงานเอเยนต์ได้
- การกระทำหรือพฤติกรรมต่าง ๆ มากมาย ที่เกิดจากเอเยนต์ที่รับสิทธิอย่างถูกต้อง เช่น ได้รับอนุญาตให้เข้าถึงโฮสต์และแก้ไขหรือดัดแปลงข้อมูลโดยที่ไม่ต้องขออนุญาต
- ความผิดพลาดโดยบังเอิญที่เกิดจากเอเยนต์ที่ได้รับสิทธิอย่างถูกต้อง

อันตรายที่ เอเยนต์/ผู้สั่งงาน อาจได้รับจากการใช้งาน

- มีการขัดขวางการทำงานของเอเยนต์ ที่ยังสามารถทำงานต่อได้แต่ไม่ตรงกับความต้องการ
- การตัดทอนข้อมูลส่วนตัวบางชนิด ที่ไม่อนุญาตให้ใช้งาน เช่น หมายเลขบัตรเครดิต ออกจากการทำงานของเอเยนต์ เพื่อควบคุมในเรื่องความปลอดภัย

อันตรายที่อาจเกิดขึ้นกับ ผู้สั่งงาน เมื่อเรียกใช้งานแอพเพ็ต

- แอพเพ็ตที่ทำงานกับข้อมูลที่เป็นอันตราย ของผู้สั่งที่ได้รับอนุญาตอย่างถูกต้อง
- แอพเพ็ตที่เปิดเผยข้อมูลให้แก่บุคคลที่สาม ซึ่งเป็นผู้ที่ไม่ได้รับอนุญาตให้ใช้งาน
- แอพเพ็ตนำกิจกรรมที่ไม่ต้องการมาทำงาน เช่น เชื่อมต่อเส้นทางการทำงานเองโดยพละการ

เราสามารถป้องกันความเสียหายที่เกิดจากการทำงานของเอเยนต์ ในระบบด้วยขั้นตอนพื้นฐานสำคัญสองประการ ได้แก่

ประการแรก เครื่องคอมพิวเตอร์ที่จะรันเอเยนต์ ต้องสามารถกำหนดได้ว่า ใครบ้างที่ได้รับสิทธิที่จะเป็นผู้ใช้งานเอเยนต์อย่างถูกต้อง เพื่อที่ จะได้กำหนดทรัพยากรที่เอเยนต์สามารถใช้งานได้

ประการที่สอง พฤติกรรมต่าง ๆ เอเยนต์จะต้องถูกจำกัดการใช้งานทรัพยากรอื่น ๆ นอกเหนือจากที่ได้กำหนดให้ใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิธีการป้องกันอันตรายจากการใช้งานเอเยนต์

จากแนวคิดในเรื่องของการป้องกัน ที่ใช้หลักของกล่องที่มีความปลอดภัย (Safety boxes) และนำการเข้ารหัสมาใช้งาน เพื่อเป็นส่วนหนึ่งของระบบความปลอดภัยในการทำงานของเอเยนต์ ซึ่งสิ่งต่าง ๆ เหล่านี้เป็นเทคนิคอย่างหนึ่งที่ช่วยลดความเสียหายที่เกิดขึ้นในระบบเอเยนต์

วิธีการป้องกันที่รีโมต์ไซต์ จากการทำงานของเอเยนต์

- ทำงานกับเอเยนต์ภายในบริเวณที่ปลอดภัย
- กำหนดสิทธิเฉพาะเอเยนต์ที่สามารถมาทำงานได้
- แก้ไขโค้ดของเอเยนต์
- ตรวจเช็คโค้ดของเอเยนต์ ในขณะที่กำลังทำงาน
- ตรวจสอบการกระทำของเอเยนต์ที่ควรงดหรือละเว้น
- ใช้กลไกควบคุมการเข้าใช้งานกับทรัพยากรของเครื่อง

วิธีการป้องกันอันตรายให้แก่ เอเยนต์/ผู้สั่งงาน จากสิ่งต่าง ๆ ภายนอก

- สร้างการตรวจสอบข้อผิดพลาดไว้ในโค้ดของเอเยนต์และข้อมูล
- เข้ารหัสการตรวจสอบข้อผิดพลาดในโค้ดของเอเยนต์และข้อมูล
- ใช้กลไกต่อต้านการทำงานซ้ำอีกครั้งหลังจากเสร็จสิ้นแล้ว
- ใช้กลไกการต่อต้านการก็อบปี้

วิธีการป้องกันความเสียหายของ ผู้สั่งงาน จากการใช้งานแอพเพ็ต

- จัดให้มีที่ ที่มีความปลอดภัยเพื่อให้แอพเพ็ตทำงาน
- มีการกำหนดให้สิทธิเฉพาะแอพเพ็ตที่สามารถทำงานได้
- แก้ไข้โค้ดของแอพเพ็ตบางส่วน
- ประมวลผลการทำงานโค้ดของแอพเพ็ตแบบแปลทีละคำสั่ง
- ตรวจสอบโค้ดของแอพเพ็ตขณะกำลังทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 การใช้งานเอเยนต์ (Agent Uses)

2.5.1 บทบาทในเชิงธุรกิจของเทคโนโลยีเอเยนต์

การนำเอาเทคโนโลยีเอเยนต์มาประยุกต์ใช้กับธุรกิจนั้นเป็นสิ่งที่จะทำให้การดำเนินธุรกิจนั้นมีความเจริญก้าวหน้าและสะดวกสบายมากยิ่งขึ้น เพราะในปัจจุบัน เครือข่ายอินเทอร์เน็ตได้ขยายออกไปทั่วโลก และผู้คนทั่วไปก็เริ่มที่จะติดต่อกันทางอินเทอร์เน็ตมากขึ้น

ดังนั้นระบบเครือข่ายอินเทอร์เน็ตจึงเป็นแหล่งการทำการค้าที่สำคัญอีกแหล่งหนึ่ง แต่การทำการค้าขายกับผู้คนรอบโลกนั้น ออกจะเป็นการยากลำบากในการที่จะหาผู้ซื้อหรือผู้ขายที่ต้องการทำการซื้อขายของเฉพาะอย่างที่ต้องการ เอเยนต์จึงเข้ามามีบทบาทในการช่วยลดเวลาที่เสียไปในการเลือกหาหรือเลือกซื้อสินค้าต่างๆเหล่านั้นแก่ยูสเซอร์ และเป็นตัวกลางในการทำการซื้อขายของร้านต่างๆอีกด้วย

2.5.1.1 ตัวอย่างเอเยนต์ที่ใช้งานทางธุรกิจที่มีอยู่บน WWW

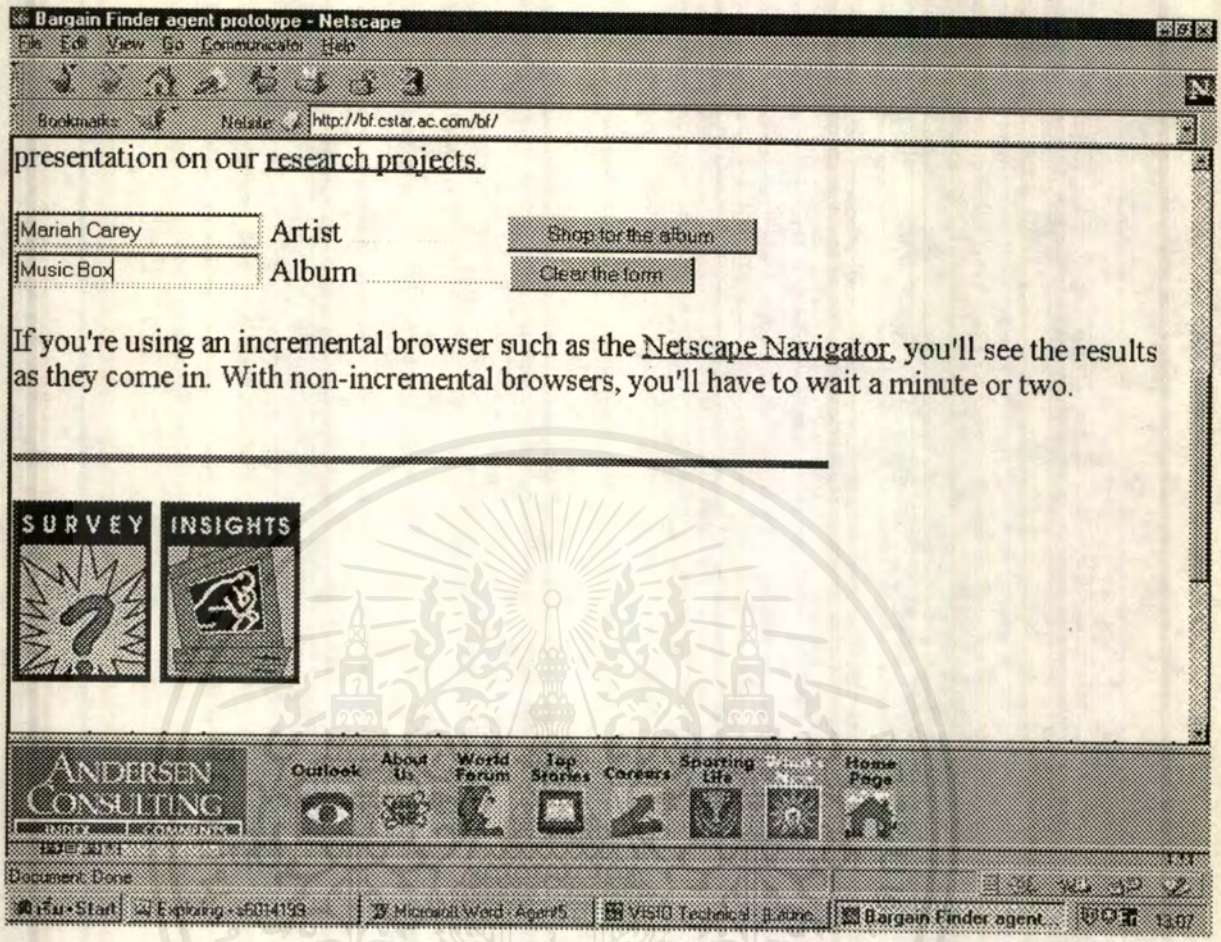
BargainFinder : เอเยนต์ช่วยหาราคาซีดีบนเว็บ

http://bf.cstar.ac.com/bf/

เอเยนต์ชื่อ BargainFinder นี้ เป็นเอเยนต์ที่ถูกพัฒนาโดยห้องปฏิบัติการของบริษัท Andersen Consulting และศูนย์ SMART STORE ของสหรัฐอเมริกา เพื่อทำงานเป็นเอเยนต์ช่วยในการค้นหาข้อมูลเกี่ยวกับราคาของซีดีของศิลปินใดๆ ที่มีวางขายตามร้านขายจริงๆ จำนวน 8 ร้าน โดยเอเยนต์นี้จะรวบรวมข้อมูลเกี่ยวกับราคาของแต่ละร้านค้าที่ขายโดยอัตโนมัติ และทำการเปรียบเทียบราคาเหล่านั้นของร้านต่างๆทั้ง 8 ร้าน ว่าร้านใดมีราคาถูกที่สุดถึงแพงที่สุดในการขายซีดีที่ลูกค้าระบุชื่อนั้น

เพียงแค่ เข้า เว็บเพจ ของเอเยนต์โดยเรียกจาก URL http://bf.cstart.ac.com/bf/ เอเยนต์นี้ก็พร้อมจะให้บริการดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

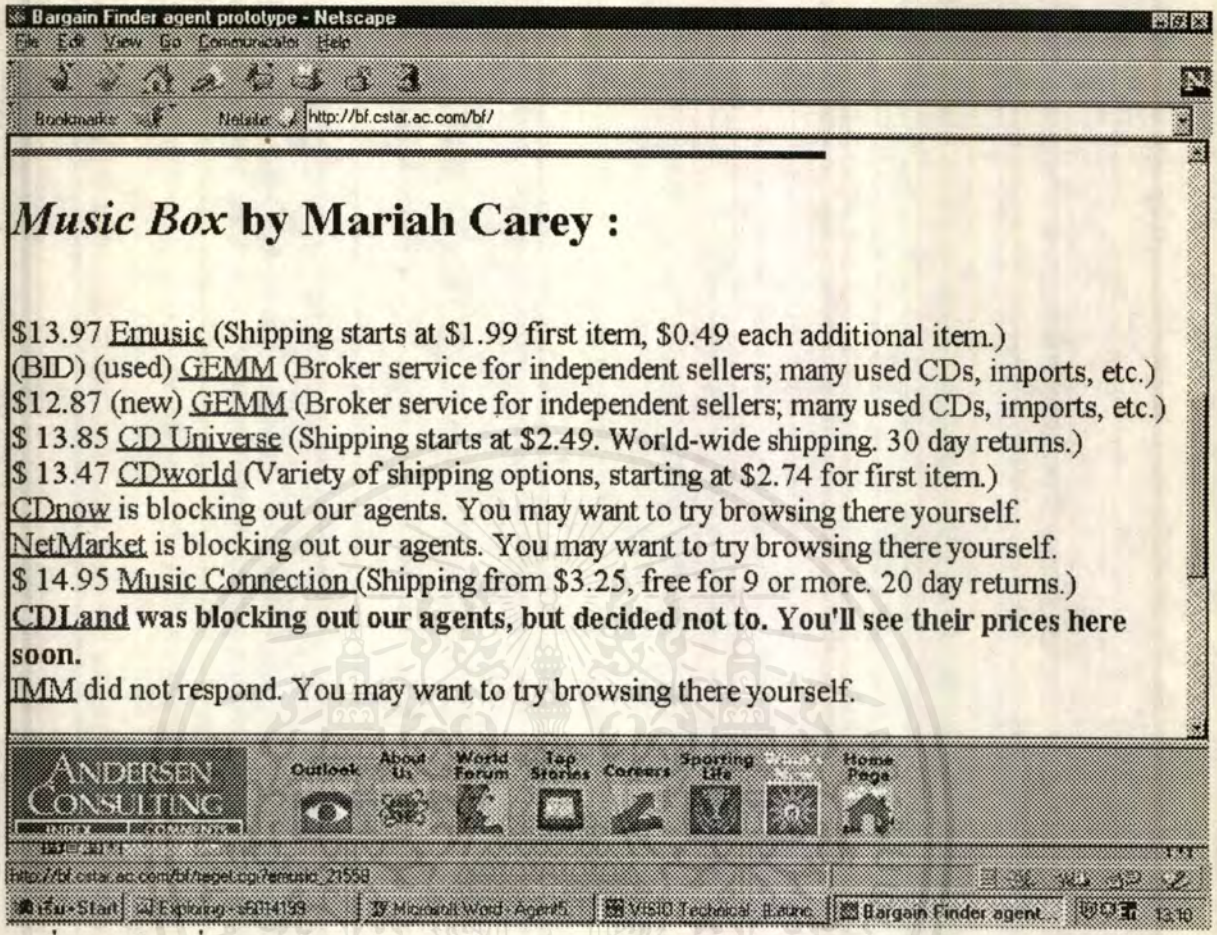


รูปที่ 21 หน้าจอที่ของ BargainFinder

จากนั้นผู้ใช้ ก็เพียงใส่ชื่อศิลปินและชื่อ Album ที่ผู้ใช้ ต้องการทราบราคาแล้วกดปุ่ม Start เอเยนต์ก็จะเริ่มทำงาน ทำการค้นหาราคาของซีดีที่ผู้ใช้ ระบุไว้ เมื่อการทำงานเสร็จสิ้นลง เอเยนต์ก็จะมีหน้าจอรายงานผลการทำงานกลับมายังผู้ใช้

ตัวอย่าง เราทดลองใส่ชื่อศิลปิน Mariah Carey และใส่ชื่อ Album : Music Box ลงไปจะได้ผลการค้นหา ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 22 หน้าจอที่แสดงผลของ BargainFiner

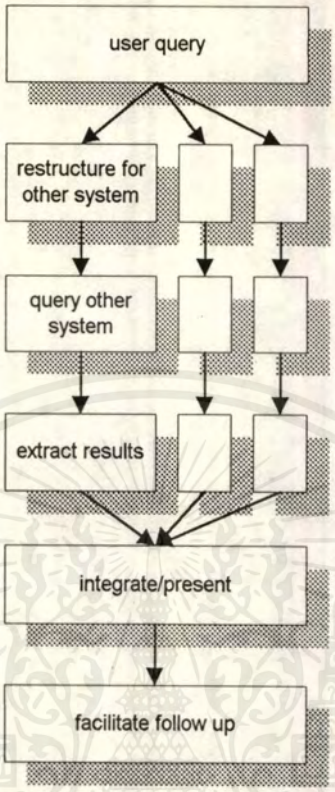
การออกแบบ BargainFinder :

BargainFinder นี้เป็นอีกตัวอย่างหนึ่งของเอเยนต์ที่เป็นประเภท Information Integration Agents ซึ่งเป็น Intelligent Agent ที่ช่วยผู้ใช้ ในการจัดการข้อมูลขนาดใหญ่ๆ โดยเอเยนต์จะทำการรวบรวมข้อมูลเหล่านั้นเข้าเป็นรูปแบบสำหรับเอเยนต์เพื่อให้ง่ายต่อการใช้งาน ซึ่งข้อมูลเหล่านั้นก็จะมาจากแหล่งต่างๆหลายแหล่งด้วยกัน และการรวบรวมข้อมูลของเอเยนต์ก็เพื่อที่จะทำการประมวลผลให้ได้ตามเป้าหมายที่ผู้ใช้ ต้องการนั่นเอง

หลักการทำงานของ BatgainFinder นี่คือ ตัวเอเยนต์จะทำการรวบรวมเอาข้อมูลเกี่ยวกับราคาของคลาสเซ็ทเทปและซีดีจาก เว็บเซอร์ฟเวอร์ ต่างๆที่เก็บข้อมูลเหล่านี้และรับสั่งซื้อทาง Internet (เรียก เซอร์ฟเวอร์ เหล่านี้ว่า Online Store) จากนั้น เอเยนต์ก็จะเลือกเอาข้อมูลที่ผู้ใช้ ต้องการมาทำการเรียบเรียงและจัดลำดับว่า ร้านใดที่เอเยนต์ไปเอาข้อมูลมานั้นขายคลาสเซ็ทเทปและซีดีที่ผู้ใช้ ต้องการซื้อหรือถามหาในราคาถูกที่สุดเรียงไปถึงราคาแพงที่สุด นอกจากนั้น หลังจากที่ผู้ใช้ทราบรายละเอียดที่ต้องการแล้ว ก็สามารถเปิดดูเว็บเพจ หรือทำการสั่งซื้อที่ร้านค้านั้นๆตามที่เอเยนต์แสดงผลออกมาได้แต่ละร้านเลยทีเดียว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลักการออกแบบทั่วไปของเอเยนต์แบบ Information Integration Agents จะเป็นดังรูป 23



รูปที่ 23 Design of Information Integration Agents

เอเยนต์จะรับคำสั่งจากผู้ใช้ และแปลงเป็นคำสั่งสำหรับแต่ละ เซอร์ฟเวอร์ แล้วส่งไปตาม เซอร์ฟเวอร์ ต่างๆ เซอร์ฟเวอร์ ต่างๆนั้นก็จะส่งเว็บเพจ ที่บรรจุไปด้วยข้อมูลจำนวนมากเกี่ยวกับของที่มีขายใน เซอร์ฟเวอร์ นั้นๆและราคาของของแต่ละชิ้น เอเยนต์ก็จะรวบรวมข้อมูลจากทุกๆ เซอร์ฟเวอร์ ที่มันติดต่อ มาคัดเลือกเอาข้อมูลที่ผู้ใช้ ต้องการและแสดงออกมาเป็นหน้าจอผลลัพธ์ โดยหลังการแสดงผลลัพธ์แล้วมันก็จะคอยส่งผู้ใช้ ไปยังเว็บเพจ สำหรับการซื้อขายของชิ้นนั้นๆ ตาม เซอร์ฟเวอร์ ต่างๆที่ผู้ใช้ ต้องการ

เอเยนต์ประเภทนี้อำนวยความสะดวกให้ผู้ใช้ ได้ใช้งานจากข้อมูลและบริการต่างๆที่มีอยู่บน Internet ในปัจจุบัน โดยเฉพาะอย่าง BargainFinder จะมีข้อได้เปรียบและเป็นเอกลักษณ์ซึ่งต่างจากพวก Search Engine ทั่วไป คือ BargainFinder ถูกออกแบบมาเพื่อประโยชน์เฉพาะอย่างนั่นคือการหาข้อมูลเกี่ยวกับคลาสเซ็ทเทปและซีดีโดยอย่างเดียวและผู้ใช้ ก็สามารถไปขอสั่งจองเทปและซีดีเหล่านั้นโดยการช่วยเหลือของเอเยนต์ตัวนี้นั่นเอง

ข้อได้เปรียบอีกอย่างหนึ่งของ BargainFinder ก็คือ การค้นหาข้อมูลของ BargainFinder เป็นการค้นหาจากระบบฐานข้อมูลที่ เว็บเซอร์ฟเวอร์ เก็บไว้ ไม่ใช่เป็นการค้นตามเว็บเพจ เหมือนกับบรรดา Search Engine ซึ่งข้อมูลในระบบฐานข้อมูลเหล่านี้ Search Engine เหล่านั้นไม่สามารถที่จะไปอ่านได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลการทำงานของ BargainFinder

แม้ว่า BargainFinder จะเป็นเพียงตัวทดลองทางด้านทฤษฎีเอเยนต์และเป็นเพียงตัวอย่างเท่านั้น แต่บริษัท Andersen Consulting ก็เอา BargainFinder มาให้บริการบน Internet มาเป็นระยะเวลานานพอสมควรแล้ว ทั้งนี้ก็ได้มีการพัฒนาตัวเอเยนต์ไปเรื่อยๆ โดยได้เริ่มนำออกสู่สาธารณชนเมื่อกลางปี 2538 โดยมีจุดประสงค์เพื่อเป็นตัวอย่างและสำหรับการอ้างอิงในแนวทางสำหรับการให้บริการแบบเอเยนต์ในอนาคต และเพียงภายใน 1 สัปดาห์ที่ออกสู่ระบบ Internet มันก็ถูกเรียกใช้มากกว่า 2000 ครั้งต่อวัน ซึ่งการวัดการทำงานของเอเยนต์ที่เป็นเอเยนต์เกี่ยวกับธุรกิจอย่าง BargainFinder นี้ จะต้องวัดว่าผู้ใช้ มีผลตอบสนองกับเอเยนต์อย่างไร

ทั้งนี้ บริษัท Andersen's Consulting ได้ออกแบบสอบถามให้ผู้ที่ได้ทดลองใช้ BargainFinder เพื่อจุดประสงค์ในการพัฒนาระบบการธุรกิจซื้อขายทาง WWW ซึ่งได้ผลการทดสอบดังนี้ : (ข้อมูลจาก หนังสือ Internet Agents โดย Fah-Chun Cheong , New Riders Publishing, 1996)

จากผู้ใช้งาน BargainFinder 9,417 คน (สุ่มเลือก) คิดเป็น

- เปอร์เซ็นของผู้ใช้ที่คลิกไปที่ร้านที่เสนอราคาถูกที่สุด 90% 8,484 คน
- เปอร์เซ็นของผู้ใช้ที่คลิกไปที่ร้านที่เสนอราคาถูกที่ 2 8% 737 คน
- เปอร์เซ็นของผู้ใช้ที่คลิกไปที่ร้านที่เสนอราคาถูกที่ 3 2% 196 คน

คำถาม : ท่านยินดีที่จะจ่ายเงินเพิ่มสำหรับการสั่งซื้อของทางอินเทอร์เน็ตที่มีบริการต่างๆไว้บริการเหมือนกับการสั่งซื้อของโดยวิธีอื่นหรือไม่

- ไม่ยินดี 47%
- ยินดี แต่ต้องเพิ่มขึ้นไม่เกิน 1 ดอลล่าร์สหรัฐฯ 29%
- ยินดี แต่ควรเพิ่มเกิน 10-15% 7%
- ขึ้นอยู่กับว่าสินค้าและบริการนั้นสมควรจะมีการคิดเงินเพิ่มหรือไม่ 6%
- ไม่รู้

11%

คำถาม : ท่านใช้บริการเอเยนต์แบบ BargainFinder ในการซื้อของทางอินเทอร์เน็ตบ่อยเพียงใด

- ไม่เคยใช้บริการมาก่อน 7%
- บางโอกาส 44%
- บ่อย แต่ไม่ทุกครั้งที่ต้องการซื้อสินค้า 39%

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

■ ประจำทุกครั้ง 10%

นอกจากนั้นยังได้ทำการสำรวจปฏิกิริยาตอบสนองของบรรดาร้านค้าที่ BargainFinder ไปขอข้อมูลเพื่อมานำเสนอกับผู้ใช้ ซึ่งร้านค้าเหล่านั้นเป็นร้านที่ปกติมีการซื้อขายกันแบบ Online อยู่แล้ว แต่ผู้ที่ต้องการซื้อของที่ร้านใดๆจะต้องเรียกไปที่ URL ของร้านนั้น ต่อเมื่อ BargainFinder ได้ติดต่อขอข้อมูลจากร้านเหล่านั้น ร้านเหล่านั้นก็มีสิทธิที่จะร่วมมือให้ข้อมูลกับ BargainFinder หรือปฏิเสธการให้ข้อมูลใดๆ หรือ อาจจะอยู่เฉยๆแต่อนุญาติให้ BargainFinder Link มาที่ร้านของตนได้ถ้าผู้ใช้ ต้องการจะทราบราคาจริงๆ

หลังจากทำการทดลองใช้ BargainFinder นี้ 8 เดือน 1 ใน 8 ร้านหยุดทำการไป, 3 ร้านปฏิเสธที่จะตอบคำถามใดๆจาก BargainFinder, 1 ร้านให้ความร่วมมือเป็นอย่างดี และ อีก 3 ร้านอยู่เฉยๆอนุญาติให้ BargainFinder ทำการ Link ไปได้ และยังมีอีก 6 ร้านที่ไม่ได้ร่วมการทดลองครั้งแรกมาขอร่วมด้วย

เหตุผลที่ร้านต่างๆต้องปฏิเสธหรือไม่ให้ความร่วมมือ คือ ด้วยเหตุผล 2 ประการ ประการแรกคือ ร้านค้าต่างๆไม่อยากที่จะถูกประเมินว่าขายถูกแพงด้วยการเปรียบเทียบเพียงแค่ราคาสินค้าต่อ 1 หน่วยเพียงอย่างเดียว บางร้านอาจจะมีบริการเสริมอื่นๆด้วย ซึ่งบริการเหล่านี้อาจจะมีค่ามากกว่าและเป็นสิ่งจูงใจลูกค้ามากกว่า ร้านค้าประเภทนี้จึงไม่นิยมนำราคาสินค้าของตนไปเปรียบเทียบกับร้านอื่นซึ่งอาจไม่เน้นเรื่องบริการดังกล่าว

ประการที่สองคือผู้ใช้ ที่ใช้งาน BargainFinder ส่วนใหญ่จะสนใจการทำงานของตัวเอเยนต์มากกว่าที่จะต้องการมาเลือกสั่งซื้อของจริงๆ ซึ่งผู้ที่เรียกใช้ BargainFinder ส่วนใหญ่จะเป็นผู้ที่สนใจเทคโนโลยีทางด้านเอเยนต์ ซึ่ง BargainFinder เป็นตัวอย่างหนึ่งเพียงไม่กี่ตัวที่มีอยู่บนอินเทอร์เน็ตจริงๆในปัจจุบัน ดังนั้นการที่ผู้ใช้ ส่งเอเยนต์ออกไป เอเยนต์ต้องทำการถามต่อไปยังร้านค้าต่างๆนั้น จะทำให้ร้านเหล่านั้นต้องเสียเวลาในการตอบ และที่ร้ายที่สุดคือ บรรดา Internet Service Provider บางแห่งอาจจะคิดราคาเพิ่มสำหรับร้านต่างๆที่มีการถามตอบจากภายนอก เนื่องจากทำให้มีข้อมูลที่ต้องทำการส่งเข้าออกในระบบมากขึ้น ร้านค้าที่ต้องจ่ายเพิ่มเหล่านี้ จึงไม่ยินดีที่จะต้องคอยตอบคำถามและจ่ายเงินต่อการถามกับคำถามจากเอเยนต์ ซึ่งไม่แน่ว่าผู้ใช้ ที่ส่งเอเยนต์นั้นออกมาจะซื้อสินค้าของตนหรือไม่

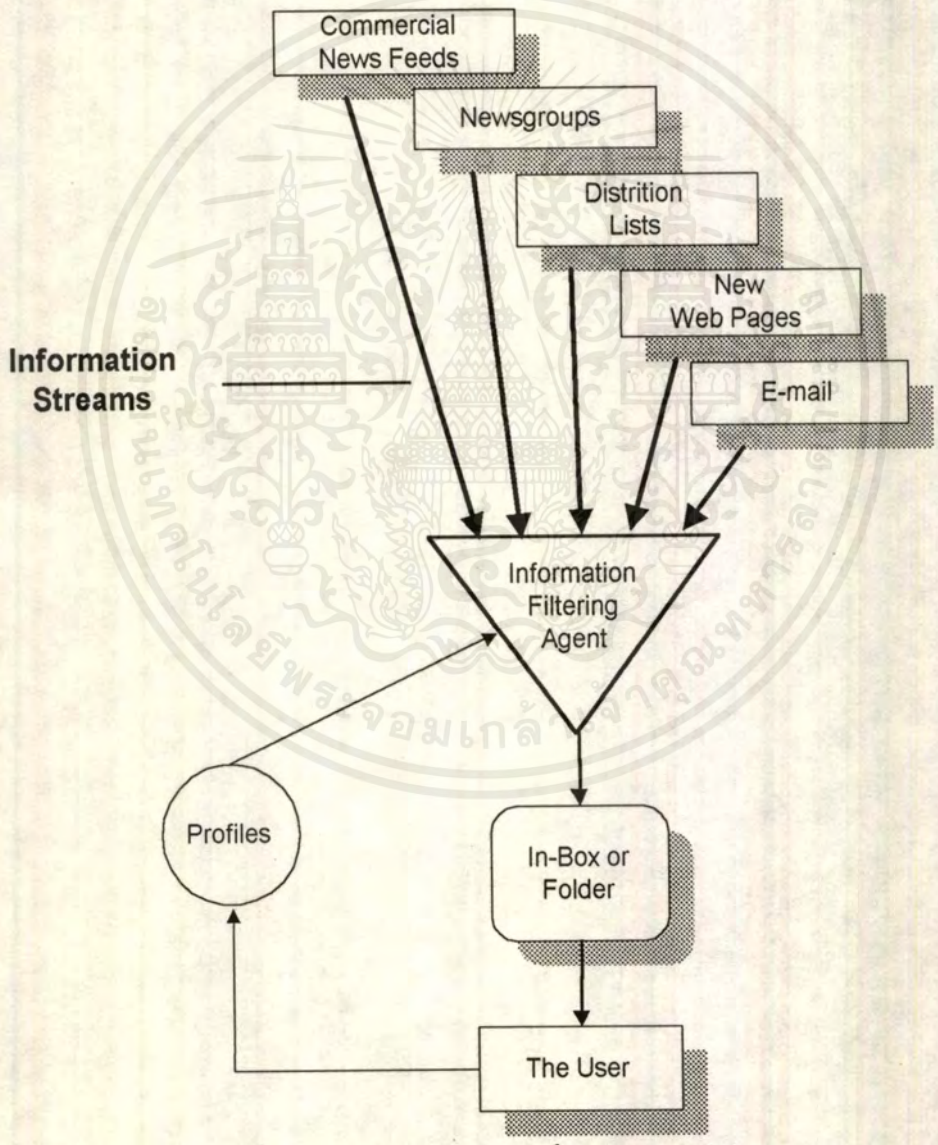
แต่อย่างไรก็ดี การทำงานของเอเยนต์ประเภทนี้ จำเป็นต้องมีการสนับสนุนจากร้านค้าต่างๆ ซึ่งเป็นแบบบริการตามสาย (Online Store) และยินยอมให้มีการซื้อขายสินค้าจริงๆ

2.5.2 การกรองข่าวสารข้อมูลจากอินเทอร์เน็ต

การกลั่นกรองข้อมูลคืออะไร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การกลั่นกรองข้อมูล (Filtering) คือ การตัดสินใจเลือกว่าสิ่งไหนที่เราต้องอ่านและสิ่งไหนที่ละเว้น ในแต่ละวันเราก็พบกับการกลั่นกรองข้อมูลข่าวสารกันอยู่แล้ว เมื่อพบเห็นพาดหัวข่าวหนังสือพิมพ์ เราก็จะอ่านตามเนื้อเรื่องที่เราสนใจ หรือไม่บางคนอาจจะข้ามไปอ่านในส่วนหัวข้อที่ตนชื่นชอบเลยก็ได้ เช่น ข่าว, กีฬา ,ธุรกิจ และสาระบันเทิง ประโยชน์ของกระบวนการนี้เอื้ออำนวยกับหนังสือพิมพ์และแม็กกาซีน เพราะมีการจัดรูปแบบไว้อย่างดีอีกทั้งเราคุ้นเคยกับรูปแบบเป็นอย่างดี เนื่องจากข้อมูลข่าวสารมีปริมาณมากและมีรูปแบบที่แตกต่างกัน เกิดขึ้นบนอินเตอร์เน็ตจึงทำให้ความต้องการกลั่นกรองข้อมูลเพิ่มขึ้นเช่นกัน



รูปที่ 24 แสดงกระบวนการกลั่นกรองข้อมูลข่าวสาร

จากรูปที่ 24 แสดงถึงขั้นตอนการกลั่นกรองข้อมูลข่าวสาร มีเอกสารต่าง ๆ มากมายหลายชนิดที่เอเยนต์กลั่นกรองข้อมูล ได้รับมาและทั้งหมดก็จะถูกนำมาดำเนินกระบวนการ เพื่อให้ตรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กับข้อมูลในโพรไฟล์ที่เราสนใจ แล้วตัวกรองข้อมูลจะนำเอกสารต่าง ๆ ที่มีความสัมพันธ์ ส่งไปไว้ที่กล่องรับข้อมูลหรือโฟลเดอร์ของผู้ใช้

สตรีมข้อมูล (Information stream) หมายถึง แหล่งที่มีข้อมูลข่าวสาร อันได้แก่ ที่ ๆ มีข้อมูลเก็บอยู่ ซึ่งข้อมูลที่ในสตรีมจะมีอยู่ตลอดเวลาและมีข้อมูลใหม่มาแทนที่ข้อมูลเก่าได้ อย่างข้อความในกล่องอี-เมลล์ ก็จัดได้ว่าเป็นสตรีมข้อมูลเช่นกัน

ข้อมูลที่อยู่ในความสนใจ (Information interest) หมายถึง ภาพลักษณ์ข้อมูลซึ่งเราจินตนาการขึ้นมาว่า เป็นเอกสารข้อมูลที่เราต้องการจะเห็น เราสามารถบอกได้ถ้าหากว่าเอกสารนั้นตรงกับความสนใจของเราเมื่อเราได้อ่านเอกสารนั้น เอเยนต์กลั่นกรองข้อมูลจะตัดสินใจได้ว่าเอกสารใดบ้างที่อยู่ในความสนใจของเรา ซึ่งมันจะทำการเปรียบเทียบเอกสารเหล่านั้นกับ โพรไฟล์ (Interest profile) ที่เราให้ความสนใจ โดยทั่วไปแล้วในภายในโพรไฟล์ที่เราสนใจนั้น จะมีเซ็ทของวลี (Phases) หรือถ้อยคำ (Terms) บรรจุอยู่ ซึ่งจะต้องปรากฏอยู่ในเอกสาร และอาจจะมีบ้างที่ถ้อยคำเหล่านั้น ไม่ปรากฏให้เห็นในเอกสาร

การกลั่นกรองข้อมุลข่าวสาร (Information filtering) หมายถึง กระบวนการที่เอเยนต์กลั่นกรองข้อมูลข่าวสาร จะอ่านเอกสารทั้งหมดที่มีอยู่ในสตรีมข้อมูลข่าวสารและนำไปเปรียบเทียบกับโพรไฟล์ที่เราสนใจ ถ้าหากว่าเอกสารนั้นตรงกับโพรไฟล์ ตัวกรองข้อมูลจะส่งเอกสารนั้นไปยังกล่องรับข้อมูลของผู้ใช้ ตามความเหมาะสมหรือนำไปเก็บไว้ที่ใดที่หนึ่งให้แก่ผู้ใช้ เพื่อเป็นที่พักของเอกสารที่จะนำไปใช้

ในตัวกรองข้อมูลบางตัว จะมีการคำนวณผลลัพธ์หา คะแนนความสัมพันธ์ (Relevance score) ซึ่งคะแนนนี้มาจากตัวประกอบต่าง ๆ อันได้แก่ จำนวนคำที่ค้นเจอในเอกสารนั้น และมีปรากฏบ่อยครั้งแค่ไหน ปกติแล้ว ถ้าคะแนนต่ำ ย่อมแสดงให้เห็นว่า เอกสารนั้นไม่เกี่ยวข้องกับสิ่งที่กำลังสนใจอยู่ในขณะนั้น และก็แน่นอนว่า ถ้าหากคะแนนที่ได้มีค่าสูง ก็แสดงให้ว่าเอกสารนั้นเกี่ยวข้องกับสิ่งที่อยู่ในความสนใจ

(Relevance threshold) เป็นค่าคะแนนต่ำสุด ซึ่งเอกสารทั้งหมดที่จะส่งไปยังผู้ใช้ได้ จะต้องมีคะแนนที่สูงกว่าคะแนนค่านี้เสมอ

(Relevance feedback) เป็นกระบวนการแก้ไขโพรไฟล์ที่เราสนใจด้วยการกำหนดคุณสมบัติของเอกสารทั้งที่มีความเกี่ยวข้องและไม่เกี่ยวข้อง เอเยนต์กรองข้อมูลที่สนับสนุน relevance feedback จะตรวจสอบสิ่งที่อยู่ในเอกสาร และอาจจะเปลี่ยนแปลงถ้อยคำที่ค้นหาในโพรไฟล์ไปเป็นถ้อยคำที่มีลักษณะใกล้คล้ายมาก-น้อยกับในเอกสารได้

คำว่า filter และ filtering มีใช้มากในเนื้อหาที่เกี่ยวกับวิทยาศาตร์คอมพิวเตอร์ , วิทยาการจัดการข้อมูล และในสาขาอื่นๆ ตัวกรอง (Filter) เป็นเอเยนต์ก็ต่อเมื่อ มันสามารถดำรงชีวิตเพื่อจะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำงานอยู่เสมอ เช่น โปรแกรมที่ทำงานอยู่อย่างต่อเนื่องหรือทำเป็นประจำสม่ำเสมอ , มีความเป็นอิสระในการปกครองตนเอง และดูแลในส่วนของการแสดงในสิ่งที่ผู้ใช้สนใจ เอเยนต์กรองข้อมูล (Filtering agents)บางตัวถูกกำหนดให้มีความฉลาดมากกว่าเอเยนต์ตัวอื่น ๆ บางตัวก็ออกแบบมาเพื่อแสดงข่าวสารที่กลั่นกรองได้ ในรูปแบบที่สร้างความดึงดูดใจได้เป็นอย่างดี ส่วนใหญ่แล้วเอเยนต์กรองข้อมูลจะมีขีดความสามารถที่เป็นประโยชน์

ข้อแตกต่างระหว่างการกลั่นกรองข้อมูลกับการค้นหา

Profession. Nicholas และ Bruce Croft ได้ชี้ให้เห็นข้อแตกต่างระหว่าง การกลั่นกรองข้อมูล (Filtering) และ การค้นหา (Normal Searching) ซึ่งแยกเป็นข้อได้ดังนี้

- ข้อมูลข่าวสารที่สนใจ (Information interest)

ในการกลั่นกรองข้อมูล จะมีโพรไฟล์ของข้อมูลที่เราให้ความสนใจไว้ ซึ่งระยะเวลาติดตามการทำงานสามารถเก็บไว้ได้เป็นระยะเวลานานเป็นสัปดาห์ ,เดือน หรือมากกว่านั้น แต่ในการค้นหาคำตอบที่ได้กลับมานั้นมักจะมีการค้นเจอในเวลาขณะนั้นทันที

- แหล่งข้อมูล (Information source)

สตรีมข้อมูลข่าวสารนั้นสามารถเปลี่ยนแปลงได้ตลอดเวลา เมื่อมีเนื้อหาใหม่เข้ามาแทนที่ ซึ่งในกรณีเช่นนี้ ตัวกรองข้อมูลก็ยังสามารถตรวจสอบข่าวสารที่รับมาใหม่นี้ได้ แต่ถ้าเป็นการทำงานของการค้นหาแล้ว ข้อมูลที่อยู่ในฐานข้อมูลจะต้องไม่มีการเปลี่ยนแปลงขณะที่กำลังดำเนินการค้นหาอยู่

- โครงสร้างเอกสาร (Document structure)

แต่ละเรคอร์ดที่อยู่ในฐานข้อมูล สร้างมาจากฟิลด์ต่าง ๆ ที่มีความหมาย ไม่ว่าจะเป็น ชื่อ ,รายละเอียดวิชา ,ลำดับเลขที่ และอื่น ๆ ทำให้การค้นหาข้อมูลในฟิลด์จึงทำได้ง่ายขึ้นเมื่อมีการระบุลักษณะข้อมูลที่เราสนใจลงไปตามความหมายของฟิลด์

โดยปกติตัวกรองข้อมูลจะจัดการกับข้อมูลที่ไม่มีโครงสร้าง หรือมีโครงสร้างที่เป็นกลาง ๆ เช่น เนื้อความอี-เมลล์ ฉะนั้น การทำงานกับข้อมูลที่ไม่มีโครงสร้างอย่างเนื้อความในจดหมายจึงไม่สามารถระบุความหมายได้ ดังนั้นตัวกลั่นกรองข้อมูลจึงต้องจับคู่ข้อความในเอกสารด้วยการเปรียบเทียบกับวลี (Phases)หรือ ถ้อยคำ (Terms) เช่น การแยกความหมายระหว่างคำว่า “ education ” กับ “ School of Education “ คงทำได้ยาก เพราะเนื้อหาของข้อความไม่มีโครงสร้าง

- การเคลื่อนย้าย กับ การค้นหา (Removing versus finding)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอเยนต์กลั่นกรองข้อมูลจะแยกเอกสารที่ไม่ต้องการออกไป แต่สำหรับการค้นหาจะไปค้นหาเอกสารที่เราต้องการมาให้

- **ปริมาณข้อมูล** (Amount of data)

เนื่องจากเอเยนต์สามารถทำงานนอกเวลาหรือเกินเวลาได้ ทำให้สามารถดำเนินการกับข้อมูลที่มีปริมาณมากได้ แต่สำหรับฐานข้อมูลมักมีขนาดที่เล็กกว่าเพราะว่าต้องเสียค่าใช้จ่ายสูงในส่วนของการจัดโครงสร้างข้อมูลและที่เก็บข้อมูล

- **ความเป็นอิสระ** (Autonomous)

เอเยนต์กลั่นกรองข้อมูลสามารถทำงานอย่างเป็นอิสระ ซึ่งจะทำงานตามคำสั่งของผู้ใช้ แต่ในระบบการค้นข้อมูล เมื่อข้อมูลที่ค้นมาได้มาถึงแล้วก็ยังคงต้องให้ผู้ใช้เรียกแสดงผลลัพธ์ โดยการป้อนคำสั่ง

- **ความเปลี่ยนแปลงของสิ่งที่สนใจ** (Changing interests)

การทำงานของเอเยนต์กลั่นกรองข้อมูลที่สนับสนุนการทำ relevance feedback อาจจะเปลี่ยนแปลงถ้อยคำหรือวลี ที่มีความสัมพันธ์กัน เพื่อนำไปตรวจสอบได้ แต่ในการค้นหาแล้ว คำตอบที่ได้จากการสั่งงานนั้น จะเป็นคำตอบที่มาจากการสั่งงานในแต่ละครั้ง

- **กระบวนการจับคู่** (Matching process)

ในกระบวนการค้นหา คำตอบที่ได้มากจากการจับคู่กับข้อมูลแต่ละเรคคอร์ดในฐานข้อมูล แต่หากเป็นกระบวนการของเอเยนต์กลั่นกรองข้อมูลนั้น เมื่อเอกสารอยู่ในสตรีมข้อมูลแล้ว เอเยนต์จะจับคู่เอกสารเหล่านั้นกับทุก ๆ โพรไฟล์ที่มีอยู่ และตัดสินว่าเอกสารใดที่ผู้ใช้จะนำไปใช้ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียด	*การกลั่นกรอง*	*การค้นหา*
• ข้อมูลที่สนใจ	ช่วงระยะยาว	ช่วงระยะสั้น
• แหล่งข้อมูล	สตรีมของข้อมูลเปลี่ยนแปลงได้	ฐานข้อมูลไม่เปลี่ยนแปลง
• โครงสร้างเอกสาร	ไม่มีโครงสร้าง หรือ เป็นกลางๆ	มีโครงสร้าง
• เคลื่อนย้าย &ค้นหา	ย้ายข้อมูลจากสตรีม	ค้นข้อมูลในฐานข้อมูล
• ปริมาณข้อมูลที่ได้	ปริมาณมาก	ปกติมีปริมาณน้อย
• ความเป็นอัตโนมัติ	เป็นอัตโนมัติ	ขึ้นอยู่กับตัวค้นหา
• ความเปลี่ยนแปลง-ของสิ่งที่สนใจ	เปลี่ยนไปได้เรื่อยๆ	สนองตอบ 1 ครั้งต่อ 1 คำถาม
• กระบวนการจับคู่	เอกสารถูกเปรียบเทียบกับกับทุกโพรไฟล์ทั้งหมด	คำถามถูกนำไปเปรียบเทียบเรคคอร์ดในฐานข้อมูล

ตารางที่ 1 แสดงความแตกต่างระหว่างการกลั่นกรอง (Filtering) และการค้นหา (Searching)

- **สถานที่ ที่เหมาะสมกับการกลั่นกรองข้อมูล**

Tak Yan และ Garcia Molina ชี้ให้เห็นในเรื่องของสถานที่ ที่สามารถกลั่นกรองข้อมูลว่ามีอยู่หลายๆ ที่ในสตรีมข้อมูล : ที่แหล่งกำเนิด , ที่บริเวณกลางทาง และ ที่ปลายทาง ซึ่งในแต่ละที่ก็มีข้อดีข้อเสียแตกต่างกันไป

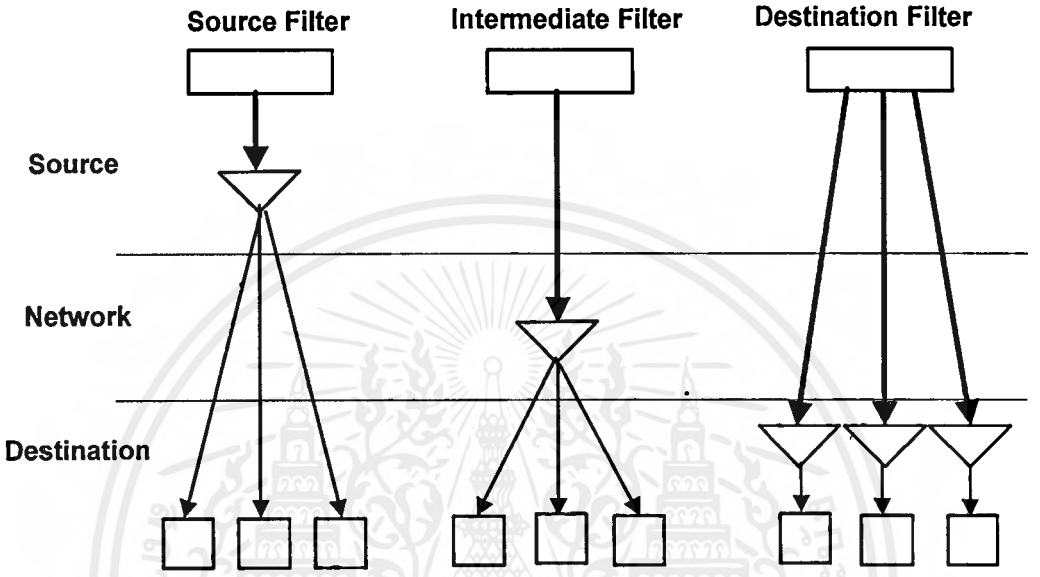
ที่แหล่งกำเนิด (Source) : เริ่มทำการกลั่นกรองข้อมูลจากแหล่งที่มาของข้อมูล ซึ่งทำให้ลดการส่งจำนวนข้อมูลบนเครือข่ายได้เป็นอย่างดี แต่ในลักษณะนี้ผู้ที่ใช้งานแต่ละคนต้องดูแลจัดการตัวกรองข้อมูลกับสตรีมข้อมูลที่ต้องการด้วยตนเอง มีการใช้งานน้อยกว่าอีก 2 แบบที่มีอยู่

ที่กลางทาง (Intermediate) : สามารถตรวจสอบสตรีมข้อมูลได้เป็นจำนวนมาก และสามารถบริการโพรไฟล์ข้อมูลได้จำนวนมากที่มาจากผู้ใช้ที่แตกต่างกันในเวลาเดียวกันได้ มีการใช้งานในส่วนนี้เป็นอย่างมาก

ที่ปลายทาง (Destination) : นั่นคือ เอเยนต์จะทำงานอยู่บนเครื่องของผู้ใช้เองหรืออาจอยู่บนเครื่องเซิร์ฟเวอร์ท้องถิ่น ซึ่งเท่าที่เห็นการทำงานพบว่ามีประสิทธิภาพการทำงานน้อยมากเมื่อเทียบกับแบบอื่น เพราะสิ่งที่อยู่ในสตรีมข้อมูลทั้งหมดจะกระจายไปยังผู้ใช้แต่ละคน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หรืออาจจะแต่ละเซิร์ฟเวอร์ อย่างไรก็ตามในเรื่องของระบบรักษาความปลอดภัยหรือความเป็นส่วนตัว ของการกลั่นกรองข้อมูลที่ปลายทางนับว่าเป็นแนวทางที่ดีที่สุดเพราะโพรไฟล์ข้อมูลที่เราสนใจก็ยังคงอยู่ในบนระบบท้องถิ่น ซึ่งจะได้รับความปลอดภัยเป็นอย่างดี



รูปที่ 25 แสดงบริเวณที่สามารถกลั่นกรองข้อมูลได้

ผู้ที่ควรจะใช้เอเยนต์กลั่นกรองข้อมูล

จะมีกลุ่มบุคคลบางประเภท ที่เหมาะแก่การกลั่นกรองข้อมูล เนื่องจากบางครั้งในบางสาขาวิชาหรือบางอาชีพ ก็ไม่จำเป็นสำหรับการกรองข้อมูลมากนัก ผู้ที่ควรใช้การกลั่นกรองข้อมูลมักจะได้แก่

- นักวิจัย
- บรรณารักษ์ เพื่อทราบรายละเอียดเกี่ยวกับเนื้อหาที่มีความสัมพันธ์ในสาขาต่างๆ
- นักวิเคราะห์ข้อมูล
- ผู้เชี่ยวชาญเฉพาะสาขา
- บุคคลที่ปรึกษา
- ผู้ดูแลเว็บไซต์
- ตัวแทนหาข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปัญหาการกลั่นกรองข้อมูล

ปัญหาต่างที่เกิดขึ้นเกี่ยวกับการทำงานของเอเยนต์กลั่นกรองข้อมูล มักพบได้ในลักษณะของข้อมูลที่ไม่ตรงกับความต้องการของผู้ใช้งาน เช่น นำข้อมูลที่ไม่มีความสัมพันธ์มาให้และอาจตัดข้อมูลบางส่วนที่เราต้องการออกไป ปัญหาอย่างอื่นที่พบได้ เช่น

- ข้อมูลที่ได้จากการกรองมีมากเกินไป

การทำงานของเอเยนต์กรองข้อมูลอาจจะนำมาซึ่งข้อมูลที่มากเกินความจำเป็น เราสามารถแก้ไขได้โดยเรียนรู้วิธีการสร้างโพรไฟล์ให้มีประสิทธิภาพที่ดีว่าทำอย่างไร โดยอาจจะมาจากประสบการณ์ที่ได้จากการใช้งานระบบ

- ปัญหาเรื่องของการใช้คำศัพท์

เนื่องจากในการใช้คำศัพท์ของบุคคลแต่ละคนนั้น มีข้อจำกัดแตกต่างกันอย่างมาก ในของสิ่งเดียวกันบางครั้งคำจำกัดความที่ได้มา ก็จะแตกต่างกันออกไปตามแต่ความรู้ความคุ้นเคยว่ามีมากน้อยแค่ไหน รวมทั้งประสบการณ์การใช้งาน

ยังมีอีกปัญหาในลักษณะการใช้คำศัพท์เช่นนี้อีกอย่างหนึ่ง คือ คำศัพท์บางคำสามารถมีได้มากกว่าหนึ่งความหมายในสาขาที่ต่างกัน เช่น คำว่า Filter ในสาขาวิทยาศาสตร์คอมพิวเตอร์ จะหมายถึง การแปลงข้อมูลจากรูปแบบหนึ่งไปเป็นอีกรูปแบบหนึ่ง เช่น แปลงจากไฟล์ GIF ไปเป็น ไฟล์ QPEG เป็นต้น ดังนั้นในการใช้คำ ควรจะเน้นให้อยู่ในหัวข้อที่มีข้อจำกัดที่แคบไว้ก่อน

- โครงสร้างเอกสาร

การกลั่นกรองข้อมูลจากเอกสารที่ไม่มีโครงสร้าง มักจะทำได้ยากกว่าการค้นหาที่สามารถระบุโครงสร้างเอกสารได้ ดังนั้น ถ้าหากสามารถระบุหรือเอเยนต์สามารถแยกโครงสร้างข้อมูลได้ว่าเป็นประเภทใด การกรองข้อมูลที่ตรงกับความต้องการก็จะทำได้อย่างมีประสิทธภาพยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.3 Search engines and Spiders

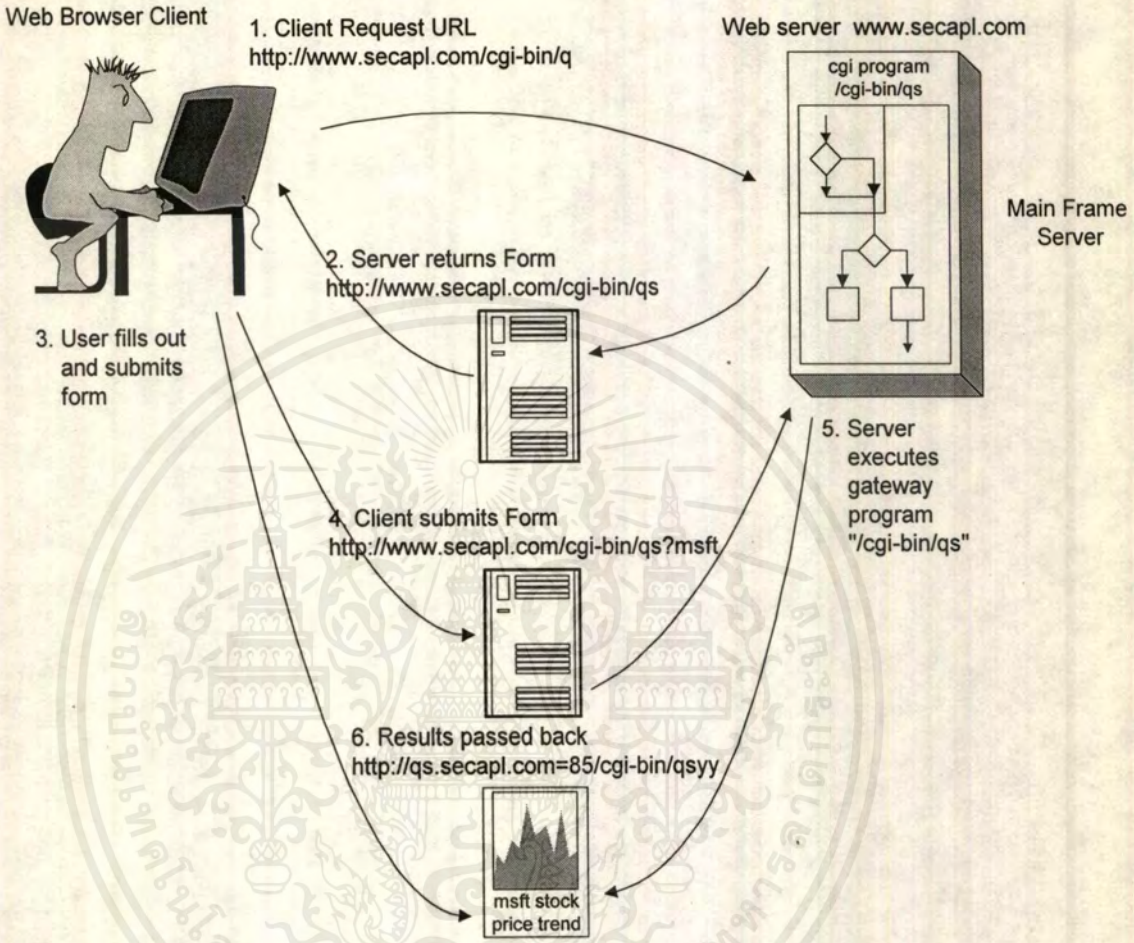
Search Engine ประเภทค้นหาคำตามเว็บเพจ ที่มีใช้อยู่อย่างแพร่หลายในระบบ Internet ปัจจุบันนั้น แต่ละตัวจะประกอบไปด้วยส่วนประกอบ 3 ส่วน ดังนี้

1. Robot : หรือจะเรียกว่า Spider หรือ Wanderer ก็ได้ จะเป็นส่วนที่ทำหน้าที่เดินทางไปตามที่ต่างๆใน Internet เพื่อเก็บคำและรายละเอียดของเว็บเพจ ต่างๆที่อยู่ทั่วโลก โดยทุกๆช่วงเวลาใดๆ มันจะเดินทางไปยัง เว็บเซอร์ฟเวอร์ ที่มันรู้จักและทำการเก็บเอาข้อมูลจาก เซอร์ฟเวอร์ นั้น ทั้งนี้เพื่อการปรับปรุงข้อมูลที่มันมีอยู่ให้ทันสมัยและทันต่อเหตุการณ์ จะเห็นได้ว่าข้อมูลที่ Robot แต่ละตัวเก็บมาได้นั้นจะไม่เหมือนกันและมีขนาดต่างกัน เช่น Robot ของ Search Engine ที่ชื่อ Lycos จะอ่านข้อมูลจำนวน 2 ล้านหน้าต่อวัน ในขณะที่ Yahoo อ่านข้อมูลเพียงวันละ 5 แสนหน้า แต่ก็มีข้อจำกัดว่าข้อมูลที่ Robot เก็บมาสะสมนั้นจะเป็นข้อมูลที่เป็นตัวอักษรล้วนๆ ซึ่งจะง่ายต่อการค้นหาต่อไป ส่วนข้อมูลประเภทภาพ หรือเสียง ที่ยากต่อการทำ Index นั้นจะไม่ถูกนำมารวมด้วย
2. Index of Web Page : จะทำการจัดหาแนวทางหรือวิธีการจัดการกับข้อมูลที่ Robot เก็บมาได้ เพื่อให้การใช้งานและการทำงานกับข้อมูลเหล่านั้น สามารถทำได้อย่างมีประสิทธิภาพและรวดเร็ว การจัดการข้อมูลอาจจะเป็นการจัดลำดับคีย์เวิร์ดและเว็บเพจที่ คีย์เวิร์ด นั้นๆบรรจุอยู่หรืออาจมีการจัด Index โดยใช้ อัลกอริธึม อื่นๆด้วยก็ได้
3. Search Engine : จะรับคำสั่งจากผู้ใช้ ผ่านทางผู้ใช้ Interface จากนั้นมันก็จะไปสอบถามที่ ส่วน Index และทำการค้นหา คำที่ได้รับมากับ Index เสร็จการค้นหา แล้วก็จะส่งผลการทำงานกลับมายังผู้ใช้

แต่เราจะเรียกทั้ง 3 ส่วนรวมๆว่า Search Engine และในปัจจุบันยังมีบริการเสริมและการทำงานอื่นๆนอกเหนือจากการค้นหา ธรรมดา อย่างเช่นการทำ Catalog ให้กับห้องสมุดและการทำเอาธุรกิจเข้าเกี่ยวข้องกับการให้บริการด้วย นั่นคือ ถ้าต้องการจูงใจให้ผู้ใช้ เข้ามาที่ เว็บ Site ของตน หลังจากที่ผู้ใช้ ได้ใช้บริการ Search Engine แล้วก็อาจต้องเสียค่าใช้จ่ายให้กับผู้ผลิต Search Engine ในการนำเอาข้อมูลหรือชื่อ เว็บ Site ของตนไว้ในลำดับต้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นตอนการทำงานของ Search Engine



รูปที่ 26 ขั้นตอนการทำงานของ Search Engine

Traversal Strategies

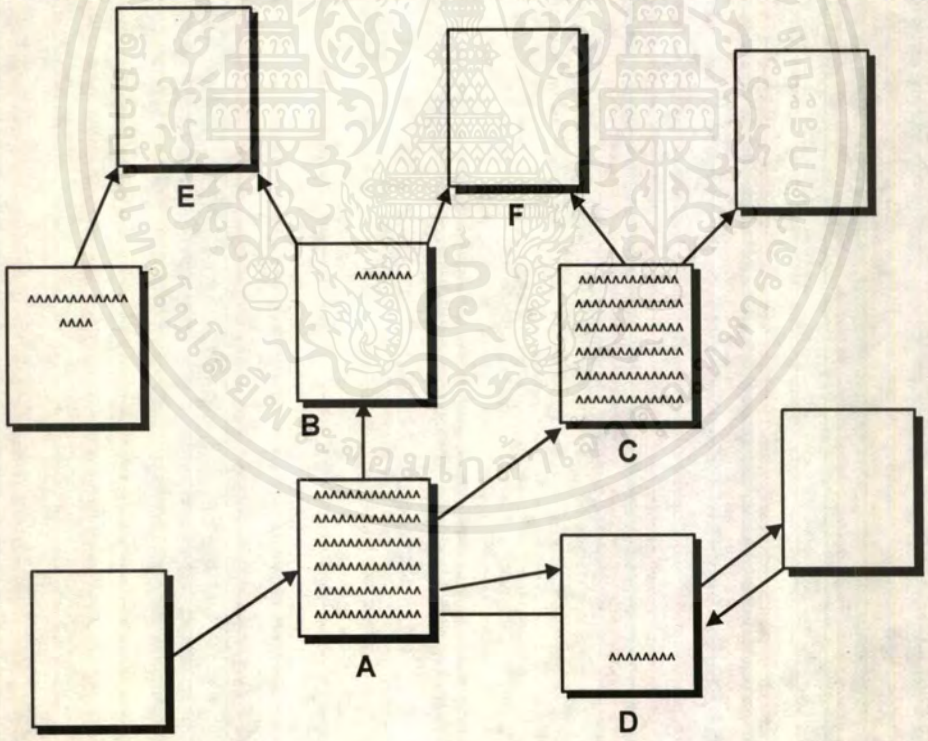
- Depth-First Search : เริ่มด้วยการมี URL เริ่มต้นอยู่ใน list จำนวนหนึ่งและทำการเก็บข้อมูลตามที่มีอยู่ใน URL นั้นๆ จากนั้นก็ดูว่าเว็บเพจ ต่างๆที่เก็บมาได้แล้วนั้นมีการ Link ไปยัง URL ไหนบ้าง ก็ทำการเก็บชื่อ URL นั้นลง list และไปเก็บข้อมูลที่ URL ดังกล่าว วิธีนี้จะพบว่าขนาดของ list จะโตขึ้นเรื่อยๆโดยไม่รู้ขีดจำกัด เนื่องจากปัจจุบัน ในเว็บเพจ มีการ Link ต่อกันไปมากมาย และเราอาจจะพบปัญหาว่า บาง URL ยังถูกเรียกหรือ Link บ่อย ถ้าในเว็บเพจ จากหลายที่มีการ Link ไปบ่อย ซึ่งเราต้องทำการข้ามไป หลักการเก็บข้อมูลแบบนี้ การค้นหา จะยิ่งลึกลงไปเรื่อยๆจนกว่าจะพบ Leaf Node หรือ เพจ ที่ไม่ได้ Link ไปที่ไหน และถ้าเราพบ โหนด ไหนที่เก็บ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลไปแล้ว ก็จะขึ้นมา 1 ขั้นแล้วไปตาม URL ที่ยังไม่ได้ไปต่อไป หรือถ้าเป็น Leaf Node ก็จะทำการเก็บข้อมูล แล้วค่อยถัดขึ้นมา 1 ขั้นก่อนที่จะไปที่ URL อื่น

เพื่ออธิบายหลักของการค้นหา แบบนี้ เราจะสมมติว่าเว็บเพจ มีการ Link กันดังรูปที่ 27

1. Robot เริ่มทำการค้นหา ที่ เพจ A มันจะเก็บชื่อ URLและข้อมูลของ A แล้วจึงสร้าง Pointer ชี้ไปยังที่ตั้งของ B, C และ D แล้วทำการ Mark ไว้ที่ A ว่าได้ค้นหา ไปแล้ว
2. Robot จะไปค้นหา ที่ B โดยทำการเก็บชื่อ URL และข้อมูลของ B แล้วจึงสร้าง Pointer ชี้ไปยังที่ตั้งของ E และ F แล้วทำการ Mark ไว้เช่นเดียวกับ A
3. Robot เริ่มค้นหา ที่ E แต่ E ไม่มี Link ไปที่ไหนอีกจึง ย้อนกลับขึ้นไปดูที่ B จึงพบว่ามี Pointer ชี้ไปที่ F อีก Robot จึงไปทำการค้นหา ที่ F
4. เสร็จจากการค้นหา ที่ F แล้ว Robot ต้องย้อนขึ้นไปดูที่ B แต่ B นั้นที่ตั้งที่ Pointer ชี้ไปก็ถูกค้นหา หมดแล้ว จึงย้อนขึ้นไปดูอีกขั้นจึงพบว่ามี Pointer ชี้ไปที่ C และ D อีกที่ยังไม่ถูกค้นหา จึงทำตามขั้นตอนเหมือนกับที่กล่าวไปแล้วต่อไป



รูปที่ 27 ตัวอย่างการเก็บข้อมูลของ Search Engine

- Breadth-First Search : เพียงเปลี่ยนการค้นหา ตาม Pointer ที่ชี้ไปในแต่ละขั้น คือ แทนที่จะทำการค้นหา ตามแนวดิ่ง ก็จะเปลี่ยนเป็นค้นหา ตามแนวนอน เมื่อค้นหา ตามแนวนอนในแต่ละขั้นครบแล้ว จึงจะลงไปค้นหา ในขั้นลึกลงอีก 1 ขั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อธิบายหลักการค้นหา ตามตัวอย่างรูปที่ 27 ได้ดังนี้

1. เริ่มต้นการค้นหา ที่ A หลังจากที่ Robot เก็บข้อมูลจาก A แล้ว ก็จะสร้าง Pointer อีกเช่นกัน เป็น Pointer ชี้ไปยัง B, C และ D
2. Robot จะไปทำการค้นหา ที่ B เป็นอันดับแรก และสร้าง Pointer ที่ B ให้ชี้ไปยัง E และ F
3. จากนั้น Robot ก็จะไปค้นหา ที่ C เพราะมี Pointer ชี้ไปยัง C อยู่ต่อจาก B ที่ C มี Link ไปที่ไหน ก็จะมีการสร้าง Pointer ชี้จากที่ C ไปยังที่นั่น
4. ค้นหา ที่ D ต่อ โดยทำเหมือนข้อ 3 ที่ค้นหา ที่ C
5. เสร็จการค้นหา ที่ D แล้ว ก็เป็นอันว่า Pointer ที่ระดับ A ถูกค้นหา ไปหมดแล้ว ก็จะมาค้นหา ที่ระดับ B,C,D กันบ้าง โดยดูที่ B ว่ามี Pointer ชี้ไปไหนบ้างนั้น จะพบว่ามีที่ E และ F
6. Robot จะไปค้นหา ที่ E ก่อนและดูว่า ที่ E มี Link ไปที่ไหนบ้างก็จะสร้าง Pointer ขึ้นที่ E ชี้ไปยังที่ต่างๆนั้น
7. เสร็จจาก E แล้วก็ไปค้นหา ที่ F โดยจะให้มีการสร้าง Pointer ลักษณะเดียวกัน
8. จากนั้น Robot จะไปค้นหา ที่ C และ D ตามลำดับ ซึ่งอยู่ระดับเดียวกับ B และมีการสร้าง Pointer ลักษณะเดียวกัน
9. เสร็จจากระดับ B,C,D Robot ก็จะลงไปอีก 1 ขั้นคือขั้นของ E และ F แล้วทำการค้นหาไปเรื่อยๆ อย่างนี้ต่อไป

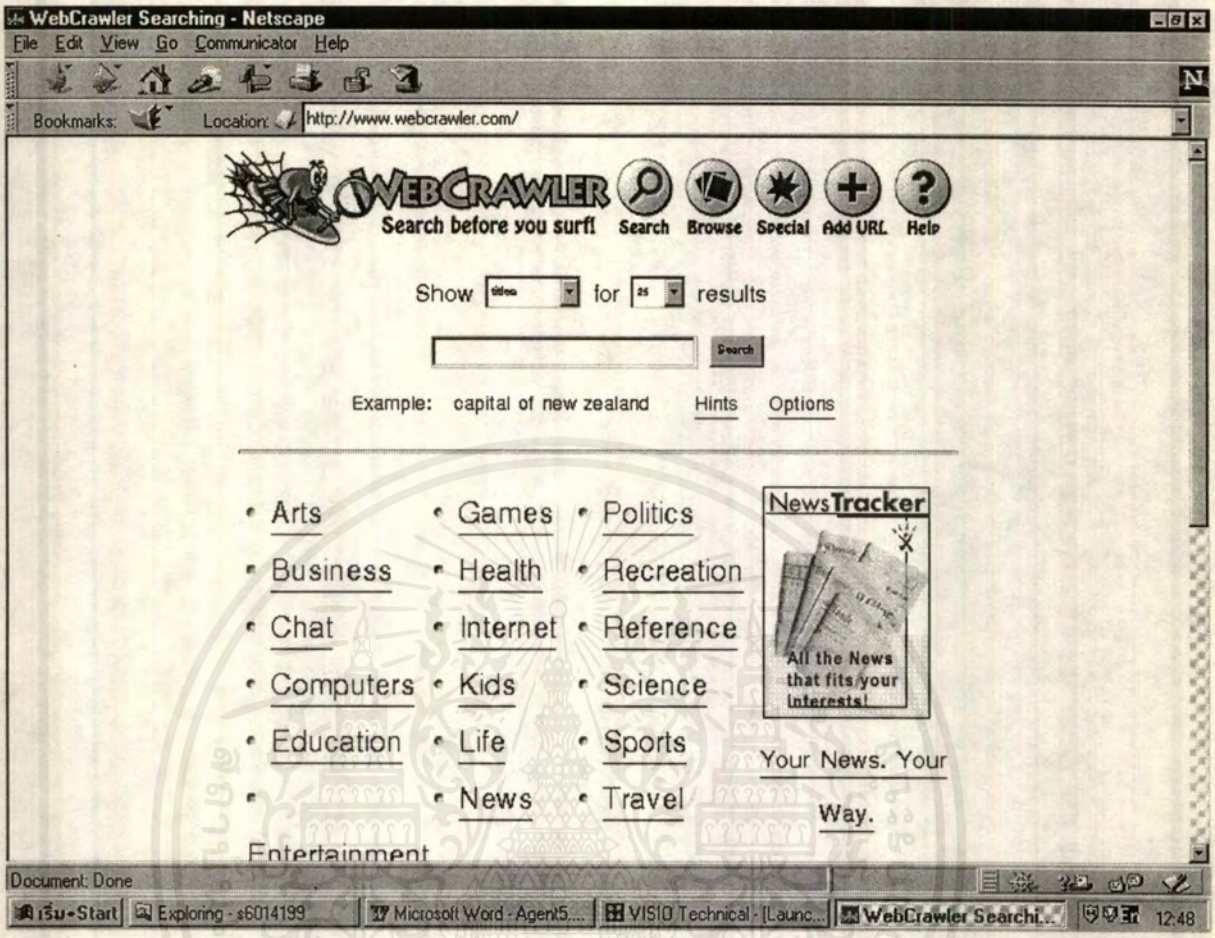
2.5.3.1 Examples of Internet Search Engines : Webcrawler and Lycos Search

ตัวอย่าง Search Engine ที่มีใช้อยู่ในปัจจุบันบน WWW

WebCrawler : http://webcrawler.com

WebCrawler เป็นโปรเจคแสดงตัวอย่าง Search Engine ของ Brian Pinkerton แห่ง University of Washington ในเมือง Seattle ประเทศสหรัฐอเมริกา ซึ่งทำงานเหมือนกับ Search Engine ทั่วไปคือรับคำสั่งจากผู้ใช้ และทำการค้นหาคำนั้นตาม Index ที่แสดงคำหลักๆของแต่ละเว็บ Site

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 28 หน้าจอของ webcrawler

WebCrawler มีความสามารถในการทำงานฟังก์ชั่นต่างๆ ดังนี้

- สร้าง Index บน เว็บ
- ทำงานค้นหาคำได้ตามคำสั่งโดยอัตโนมัติ

ผู้ใช้ ทั่วไปสามารถเรียกใช้งาน WebCrawler ได้โดยเรียกหน้าจอของ WebCrawler บน เว็บ Browser เช่น Netscape Navigator หรือ Mosaic หรือผู้ใช้ เฉพาะกลุ่มอาจจะเรียกใช้ WebCrawler Client มา Run เองเลยก็ได้

ข้อมูลเกี่ยวกับ WebCrawler :

การเก็บข้อมูลเป็นฐานข้อมูลของ Index : ประมาณ 100 เมกะไบต์

แหล่งข้อมูล : มากกว่า 150,000 แหล่งทั่วโลก

ลักษณะการเก็บข้อมูล : Incomplete breadth-first traversal

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะการสร้าง Index : ทั้ง Index ของหัวเรื่องและสารบัญของเว็บเพจ ต่างๆถูกเก็บเป็นแบบ Vector

ลักษณะการเดินทางของ WebCrawler

WebCrawler จะเดินทางไปเก็บข้อมูลต่างๆบน Internet ทีละข้อความเท่านั้น แล้วดูว่าจากข้อความนั้น จะเดินทางไปที่ใดต่อไปจึงจะดีที่สุด

- WebCrawler พยายามที่จะเพิ่มรายชื่อของ เว็บ เซอร์ฟเวอร์ ในลิสต์ให้มากที่สุดเท่าที่จะมากได้
- WebCrawler ใช้การค้นหาแบบ Breadth-first search ในการสร้าง Index ใหญ่ รวมความสัมพันธ์ของระหว่างเว็บเพจ เข้าด้วยกัน และเพื่อให้แน่ใจว่าทุก เซอร์ฟเวอร์ ที่ไป search มีอย่างน้อย 1 ข้อความที่ถูกค้นหา แล้วและเก็บเข้าไว้ใน Index แล้วด้วย
- ใช้ระบบการจัด Index แบบใช้สารบัญเป็นพื้นฐาน คือเรียงตัวอักษรเอาไว้ตามลำดับตัวอักษร ซึ่งการทำแบบนี้จะให้การจัด Index ที่มีคุณภาพสูง ส่วนการ Load ข้อความใดๆ จะ Load จาก เซอร์ฟเวอร์ ที่มี URL แน่นอนเท่านั้น

การเดินทางไปตาม WWW เพื่อค้นหาข้อความใหม่ๆ เป็นสิ่งที่จำเป็นสำหรับการออกแบบ Webcrawler และต้องออกแบบให้รองรับการขยายตัวของระบบ WWW ในปัจจุบันด้วย

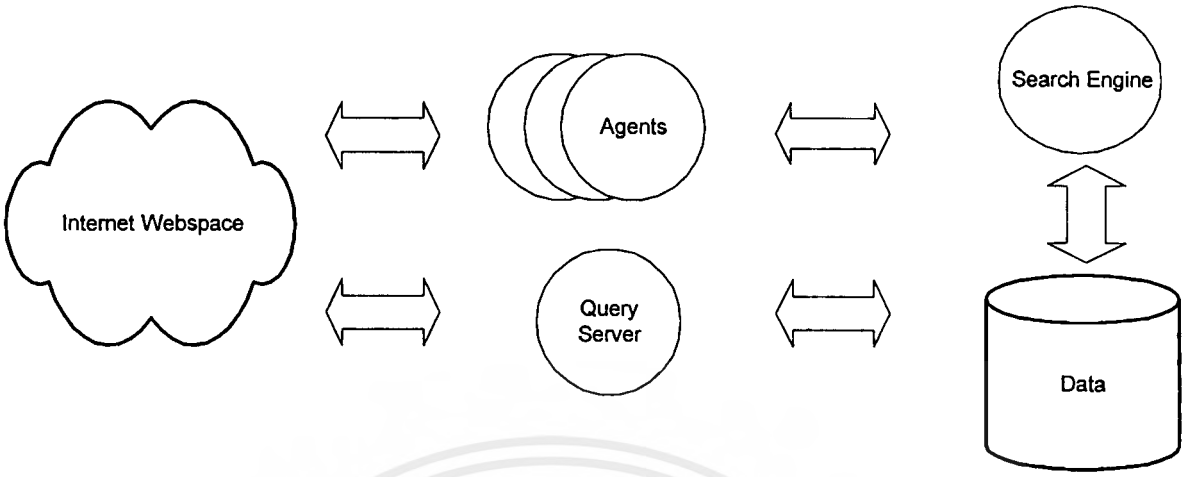
ดังที่ได้กล่าวไปแล้วถึงหลักการทำงานโดยทั่วไปของ Search Engine ในการเดินทางไปค้นหาข้อความใหม่ และเก็บรายละเอียดมานั้น WebCrawler ก็ใช้หลักการเดียวกัน นั่นคือ

1. ไปยังข้อความใหม่ที่พบนั้น
2. Mark ไว้ว่า ข้อความนี้ถูกเก็บรายละเอียดแล้วว
3. สร้าง Poniter ไปยังข้อความต่างๆที่ข้อความนี้ Link ไปหา
4. สร้าง Index ในสารบัญของข้อความนี้ในฐานข้อมูล

WebCrawler จะทำข้อ 1.-4. นี้วนไปเรื่อยๆจาก Link หนึ่งไปสู่อีก Link รอบโลก

WebCrawler มีสถาปัตยกรรมดังรูป 4.3 ซึ่งประกอบไปด้วย ส่วนประกอบ 4 ส่วน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 29 สถาปัตยกรรมของ WebCrawler

ส่วนประกอบทั้ง 4 ส่วนของ WebCrawler ได้แก่

1. Search Engine : จะพิจารณาว่าข้อความไหนใน WWW และประเภทไหนของข้อความที่จะไปทำการเก็บรายละเอียด ข้อมูลพวกที่จัด Index ไม่ได้อย่างเช่น รูปภาพ, เสียง, Postscript หรือข้อมูลที่เป็นเลขฐาน สอง จะไม่ถูกนำมาเก็บเข้าฐานข้อมูลของ WebCrawler

Search Engine มีการทำงาน 2 แบบ คือ แบบ Index Mode และแบบ Real-time Search Mode

- Index Mode : จุดมุ่งหมายของการทำงานแบบนี้ คือการสร้าง Index บน เว็บ ให้มากที่สุดเท่าที่จะมีเนื้อที่พอเก็บ Index นั้นได้ ซึ่ง WebCrawler เชื่อว่าจะเก็บข้อมูลให้ได้มากก็ต้องเดินทางไปตาม เซอร์ฟเวอร์ ต่างๆมาก มันจึงใช้ Breadth-first Search อัลกอริธึม เพื่อให้แน่ใจว่าทุกๆ เซอร์ฟเวอร์ จะมีอย่างน้อย 1 ข้อความแสดงอยู่บน Index ซึ่งในการทำงานแบบนี้ Search Engine จะมีขั้นตอน ดังนี้

1) เมื่อข้อความใดๆที่ เซอร์ฟเวอร์ ใหม่ที่ WebCrawler ไม่เคยมาเก็บข้อความใดๆ มาก่อนถูกพบ ชื่อ เซอร์ฟเวอร์ นั้นก็จะถูกจัดเข้าในรายชื่อ เซอร์ฟเวอร์ ที่จะต้องไปเก็บข้อมูลต่างๆทันที

2) หลังจากที่ติดต่อกับ เซอร์ฟเวอร์ ใหม่ตามชื่อที่ปรากฏอยู่บนรายชื่อที่มีอยู่ได้แล้ว WebCrawler จะต้องทำการเก็บ 1 ข้อความที่ เซอร์ฟเวอร์ นั้น และถูกจัดเข้า Index ก่อนที่จะไปอ่านข้อความอื่นๆต่อไป

3) เมื่อเดินทางไปเก็บข้อมูลจากทุกๆ เซอร์ฟเวอร์ มาแล้ว ขั้นตอนการทำ Index ก็จะเริ่มขึ้น โดยจัดลำดับการเอาคำต่างๆเข้า Index ตามลำดับของ เซอร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จนครบทุก เซอร์ฟเวอร์ จนกระทั่งมีการพบ เซอร์ฟเวอร์ ใหม่ ก็จะเริ่มทำตาม ข้อ 1) ใหม่ วนอย่างนี้เรื่อยไป

- Real-time Search Mode : จุดมุ่งหมายของการทำงานแบบนี้ คือ การหาข้อความบน WWW ที่บรรจุคำที่ผู้ใช้ ต้องการ หรือคำที่ใกล้เคียงที่สุดก็ได้ ซึ่งการทำงานแบบ Real-time Search Mode นี้มี อัลกอริธึม ดังนี้

  1) WebCrawler จะทำงานตามคำสั่งของผู้ใช้ โดยใช้ Index ของมัน เพื่อหารายชื่อของข้อความที่เกี่ยวข้องกับที่ผู้ใช้ ต้องการที่ปรากฏขึ้นมาก่อนเป็นอันดับแรกๆ
  2) จากรายชื่อข้อความเหล่านั้น ข้อความที่ตรงกับกรณีที่ผู้ใช้ ต้องการมากที่สุด จะถูกบันทึกไว้ รวมทั้ง Link ต่างๆในข้อความนั้นที่ยังไม่ถูกไปเก็บข้อมูลมา ก็จะถูกบันทึกไว้ด้วย
  3) เมื่อข้อความใหม่ๆไปถูกเก็บมาได้ ก็จะถูกรวมไว้ใน Index และคำสั่งในการค้นหาก็จะถูกเรียกขึ้นอีกครั้ง
  4) ข้อความที่ถูกค้นพบหลังการค้นหา จะถูกนำมาจัดเรียงตามลำดับความสำคัญของข้อความ คือ ถ้าข้อความใดตรงกับที่ผู้ใช้ ต้องการมากที่สุด ก็จะถูกนำมาอยู่ต้นๆ
  5) กระบวนการค้นหาจะหยุดลง เมื่อ WebCrawler พิจารณาว่า Link ที่เกี่ยวข้องต่างๆ ได้ถูกเดินทางไปค้นจนเป็นที่น่าพอใจแล้ว หรือ เวลาในการทำกระบวนการค้นหาหมดลง

2. Agent : ส่วน Agent จะถูกเรียกใช้โดยส่วน Search Engine สำหรับการทำการเก็บข้อความต่างๆที่ Search Engine หาได้ เนื่องจากว่าถ้า Search Engine ต้องมาทำหน้าที่นี้เสียเอง จะทำให้เสียเวลาในการเก็บข้อมูลจากที่ต่างๆมาก เพราะการรอรับข้อมูลจาก เซอร์ฟเวอร์ แต่ละ เซอร์ฟเวอร์นั้นอาจจะใช้เวลานาน เนื่องจากการได้รับการตอบสนองจากแต่ละ เซอร์ฟเวอร์ ไม่เหมือนกัน บาง เซอร์ฟเวอร์ อาจมีการตอบสนองช้าถ้ามีข้อมูลเข้า-ออกที่ เซอร์ฟเวอร์ นั้นมากๆ

Agent จะทำงานแยกส่วนต่างๆกันออกไป แต่ละส่วนทำงานของตน เพื่อให้ส่วนอื่นๆหรืองานอื่นๆที่น่าจะทำได้ระหว่างการ Agent ทำงานนี้ไม่ต้องมารอ WebCrawler มี Agent สำหรับใช้งานถึง 15 ตัวซึ่งทำงานโดยไม่ต้องรอหรือพึ่งพังกัน

ในการค้นพบข้อความใหม่ๆ Search Engine จะหา Agent ที่ว่างอยู่ และส่ง Agent นั้น ออกไปยัง URL นั้นแล้ว อาจได้ผลลัพธ์กลับมาเป็นข้อมูลจากข้อความนั้นๆ หรืออาจไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ได้ข้อมูลที่ต้องการกลับมาซึ่ง Agent ก็จะมีคำอธิบายว่าเหตุใดจึงไม่สามารถเก็บข้อมูลนั้นได้ หลังจากเสร็จการทำงานแล้วมันก็จะว่างอีกครั้งและพร้อมที่จะรับงานต่อไป

โปรแกรมของส่วนที่เป็น Agent นี้ ใช้ CERN WWW Library (หรือ libWWW) ซึ่งจะรองรับการทำงานกับข้อมูลหลายๆชนิดผ่านทางโปรโตคอลต่างๆ ไม่ว่าจะเป็น HTTP, FTP หรือ Gopher และการให้ Agent ทำงานแยกส่วนกันนี้จะช่วยการเสี่ยงที่อาจเกิดขึ้นต่อการทำงานหลักของ WebCrawler ในกรณีที่เกิดข้อผิดพลาดในตัว Agent หรือใน libWWW

3. DataBase : ฐานข้อมูลของ WebCrawler จะเก็บข้อมูล 2 อย่างคือ Index ที่เป็นการเรียงตัวอักษรล้วนๆและภาพแสดงการ Link ของ เว็บ แบบเป็น Graph ฐานข้อมูลนี้จะถูกเก็บลง Disk และจะถูก Update เมื่อมีข้อความเพิ่มเข้ามา ซึ่งการ Update นี้จะต้องกำหนดว่าต้องมีการ Committ ต่อการอ่านข้อความเพียง 200-300 ข้อความเท่านั้น ทั้งนี้เพื่อป้องกันการสูญเสียข้อมูลที่อาจเกิดขึ้นระหว่างการ Update ถ้าระบบเกิดเสียขึ้นมา

WebCrawler ใช้ IndexingKit ของ NeXTStep ในการสร้าง Index ตัวอักษรของมัน ซึ่งจะช่วยให้การทำ Query สามารถทำได้อย่างรวมเร็ว โดยการสร้าง Pointer ขึ้นมาและชี้ไปยังข้อความต่างๆที่มีคำนั้นบรรจุอยู่ ส่วนในการทำ Index นั้นจะมีการให้น้ำหนักกับคำที่ถูกค้นและคำที่ปรากฏอยู่ในเว็บเพจ บ่อยๆ จะให้อยู่ต้นๆของ Index List

ข้อมูลอื่นๆในฐานข้อมูลของ WebCrawler คือข้อมูลเกี่ยวกับ เซอร์ฟเวอร์, ข้อความต่างๆ และ Link ต่างๆ มันจะไม่เก็บ URL ของแต่ละที่เต็มๆชื่อ URL นั้น แต่จะแบ่ง URL นั้นออกเป็น อ็อบเจค ย่อยๆตามขั้นการตั้ง เซอร์ฟเวอร์ เพื่อแสดงรายละเอียดของ เซอร์ฟเวอร์ และข้อความนั้น

การ Link ระหว่างข้อความหนึ่งไปยังอีกข้อความ จริงๆแล้วก็เป็น Pointer ชี้ข้ามข้อความกันนั่นเอง แต่ละ อ็อบเจค (ที่ถูกแบ่งแยกออกมา) จะถูกเก็บใช้ใน Btree ต่างๆใน Disk Btree ก็ถูกแบ่งเป็น Btree ของข้อความ, Btree ของ เซอร์ฟเวอร์ และ Btree ของ Link การแบ่งข้อมูลออกเป็น อ็อบเจค และ Btree อย่างนี้ ทำให้ WebCrawler สามารถ Scan List ของ เซอร์ฟเวอร์ ได้อย่างรวดเร็ว และสามารถรู้ว่า เซอร์ฟเวอร์ ใด ยังไม่ได้ไปเก็บข้อมูล หรือ สามารถรู้ได้ว่า เซอร์ฟเวอร์ ใดที่เพิ่งจะไปเก็บข้อมูลมาเร็วๆนี้ ป้องกันการซ้ำซ้อน

4. Query Server : เป็นการรับการทำงานของ WebCrawler โดยผ่านทางแบบฟอร์มที่เป็น HTML ซึ่งเป็นแบบฟอร์มที่ติดต่อกับผู้ใช้ ซึ่งทำงานอย่างง่ายๆและได้ผลดีในการค้นหาข้อความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

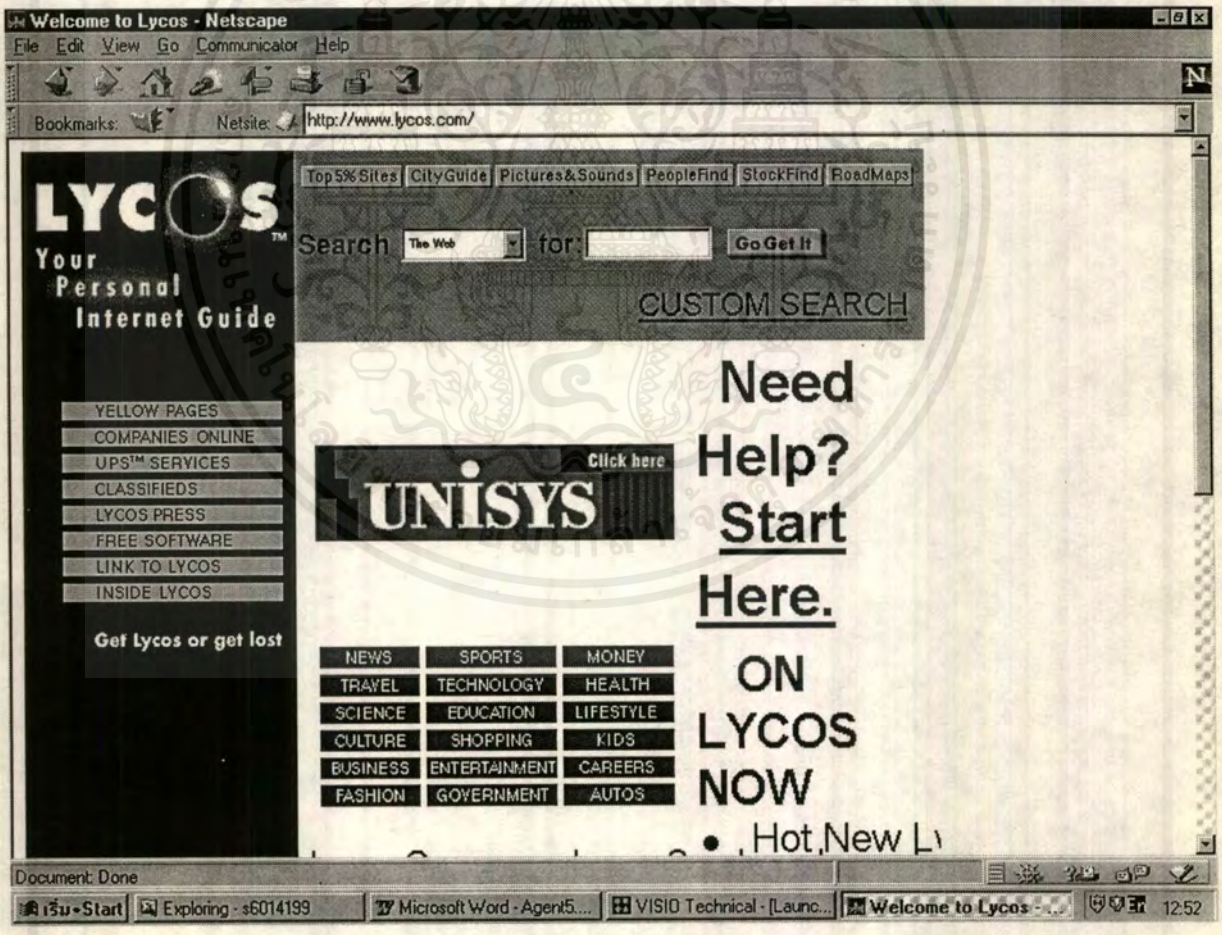
ต่างๆบน WWW เพียงผู้ใช้ ใส่คำที่ต้องการลงไปในแบบฟอร์ม Query Server ก็จะเอาคำนั้นมาสร้างเป็นคำสั่งค้นหาในการค้นหากับ Index ในส่วนของ Search Engine ต่อไปเพียงเท่านี้

Lycos Search : http://www.lycos.com

Lycos เป็นเอเยนต์ที่ทำหน้าค้นหา หาเว็บเพจที่บรรจุคำที่ผู้ใช้ ต้องการจะทราบว่าเว็บเพจใดบรรจุหรือมีหัวข้อหลักอย่างที่ผู้ใช้ กำลังมองหาอยู่เช่นเดียวกับ WebCrawler และ Search Engine อื่นๆที่มีใช้อย่างแพร่หลาย

Lycos เป็นผลงานของ Dr. Michael Mauldin แห่งมหาวิทยาลัย Carnegie Mellon ประเทศสหรัฐอเมริกา ซึ่งตั้งชื่อ Lycos จากตระกูลหนึ่งของแมงมุม

แมงมุม Lycos นี้ถูกนำออกสู่สาธารณะเมื่อกลางปี 1994 และถูกเรียกใช้งานจากผู้ใช้งาน Internet ทั่วโลก จนแมงมุม Lycos กลายมาเป็น Search Engine ที่แพร่หลายและมีประสิทธิภาพที่สุด



รูปที่ 30 หน้าจอ Lycos

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

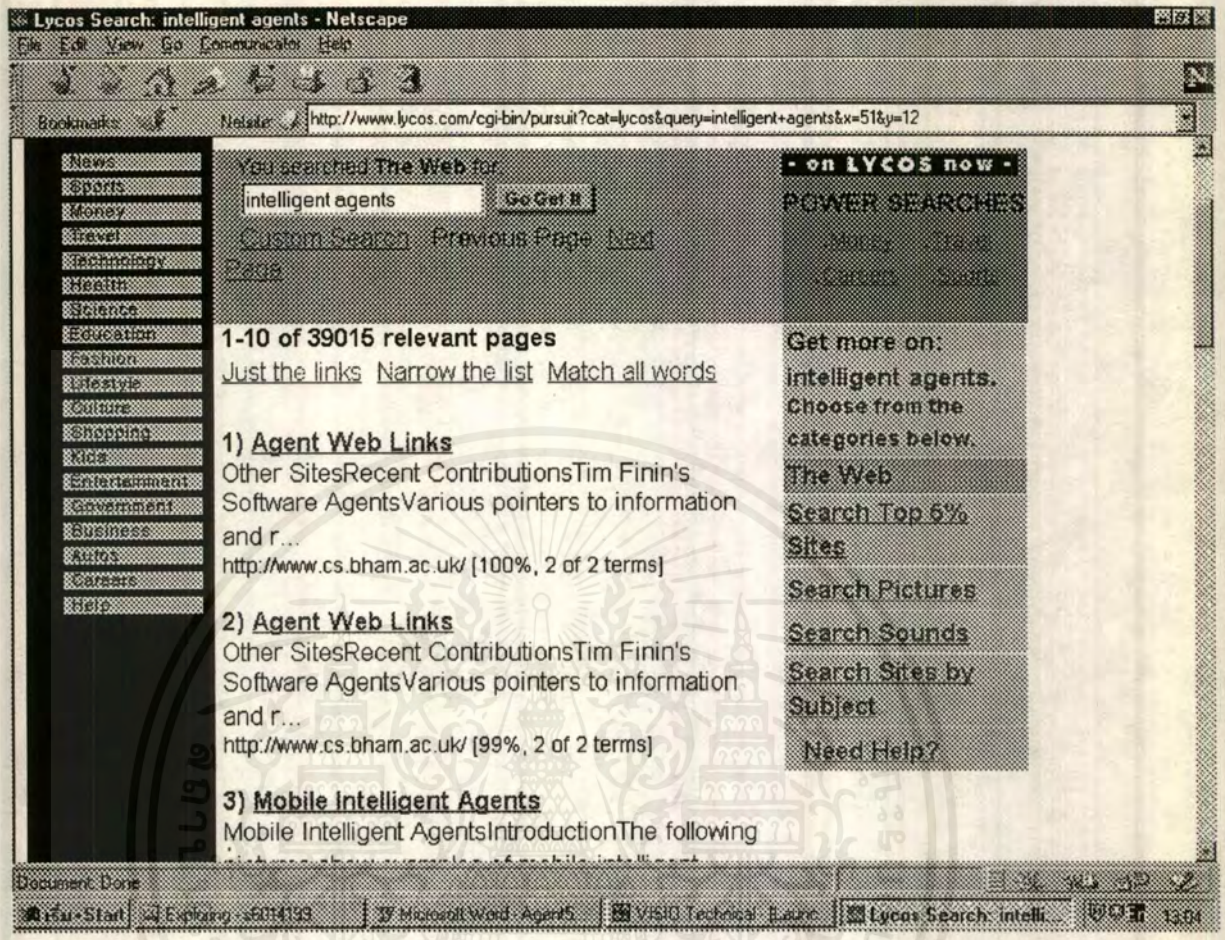
ข้อมูลที่ Lycos เก็บมีดังนี้ (ข้อมูลจาก หนังสือ Internet Agents โดย Fah-Chun Cheong , New Rider Publishing 1996)

- 5,077,834 URL ต่างๆโดยไม่ซ้ำที่กัน
- 1,177,750 ข้อความ (คิดรวมเป็น 8,7703,484,067 ไบต์)
- 3,900,084 URL ที่ยังไม่ได้ไปเก็บข้อมูล แต่ได้ไปสำรวจไว้แล้ว
- ข้อสรุปรวมจากตัวเจ้าแมงมุม 1,834,323 ไบต์
- Index 1,078,127,917 ไบต์

ฐานข้อมูลของแมงมุม Lycos ขยายตัวอย่างมาก จากเริ่มต้นเมื่อเดือน สิงหาคม 1994 มีจำนวนข้อมูลจาก URL เพียง 634,000 แหล่ง แต่ในเดือนเดียวกัน ปี 1995 ข้อมูลจากแหล่ง URL เพิ่มขึ้นเป็น 5.6 ล้านแห่งทั่วโลก ซึ่ง Lycos ให้ผลการทำงานจากการค้นหาคำเป็นข้อมูลจำนวนมากให้ผู้ใช้ ได้รับทราบผลการค้นหาอย่างเต็มที่

การใช้งาน Lycos ก็เหมือนกับการใช้งาน Search Engine ทั่วไป นั่นคือผู้ใช้ เรียกหน้าจอของ Lycos จาก URL ดังกล่าวข้างต้น จะได้หน้าจอดังรูปที่ 30 และใส่คำที่ต้องการหาในช่องและทำการส่งหน้าจอกลับมายัง เซอร์ฟเวอร์ ซึ่งหลังจากการทำงาน Lycos ก็จะแสดงผลการค้นหา ว่าคำที่ผู้ใช้ ใส่ลงไปนั้นมีบรรจุหรือกล่าวถึงในเว็บเพจ ใดบ้าง ดังรูปที่ 31

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 31 หน้าจอผลการค้นหาคำของ Lycos

Lycos แบ่งลักษณะของข้อมูลที่ต้องไปเก็บรายละเอียดดังนี้

1. ข้อมูลที่อยู่ใน HTTP
2. ข้อมูลที่อยู่ใน FTP
3. ข้อมูลที่อยู่ใน Gopher

Lycos จะไม่เก็บข้อมูลที่อยู่ในกลุ่มข้อมูลต่อไปนี้

1. WAIS
2. Usenet news
3. Mailto
4. Telnet
5. ไฟล์ข้อมูลของแต่ละที่ที่ไม่มีการเรียกใช้จากงานภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. ไฟล์ที่มีเริ่มชื่อเป็น "/dev/tty" หรือ ไฟล์ที่มีส่วนขยายเป็น AU,AVI,BIN, .DAT,DVI,EXE,FLI,GIF, GZ, HDF,HQX,JPEG,LHA,MAC,MPEG,PS, TAR,TGA,TIFF,UU,UUE,WAV หรือ ZIP

Lycos จะเก็บข้อมูลจากแต่ละข้อความดังนี้

1. หัวข้อ
2. หัวเรื่องและหัวเรื่องย่อย
3. คำที่สำคัญของข้อความนั้น 100 คำ
4. 20 บรรทัดแรก
5. ขนาดของข้อความ (หน่วยเป็นไบต์)
6. จำนวนคำทั้งหมดในข้อความนั้น

การเดินทางของ Lycos

ส่วนการทำงานที่เป็นการเดินทางไปบน WWW ของ Lycos นั้นจริงๆแล้วได้ดัดแปลงมาจากโปรแกรมชื่อ Longlegs ซึ่งเขียนโดย John Leavitt และ Eric Nyberg จากมหาวิทยาลัย Carnegie Mellon เช่นกัน

1. เมื่อแหล่งข้อมูลที่ URL ใดๆถูกอ่านเพื่อเก็บเข้าฐานข้อมูลของแมงมุม Lycos เจ้าแมงมุมก็จะตรวจดูว่าในสารบัญของ URL นั้น ว่ามีการอ้างอิงไปถึง URL อื่นๆใหม่ๆอะไรบ้าง ซึ่งถ้ามีก็จะทำการเพิ่มเข้าไปในคิวของ Lycos เองว่ามี URL ที่ต้องไปทำการเก็บข้อมูล

2. ในการเลือก URL ใหม่ๆใดๆที่จะไปเก็บข้อมูลนั้น Lycos ใช้วิธีสุ่มเลือกเอาจากกลุ่มข้อมูล 3 กลุ่มคือ HTTP หรือ FTP หรือ Gopher เอามาสร้างเป็นคิว

Lycos นิยมที่จะเลือกไปเก็บข้อมูลจากข้อความที่เป็นที่นิยมของคนทั่วไป ซึ่งควรจะเป็นข้อความที่มี Link จากที่ต่างๆหลายๆที่ Link เข้ามาหา และ Lycos ยังค่อนข้างที่จะชอบ URL ที่สั้นๆซึ่งการทำ Tree URL เหล่านี้จะอยู่ใกล้ "root" ซึ่งจะทำให้ใช้เวลาในการค้นหาน้อยกว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่3
ขั้นตอนการสร้างโปรแกรม

3.1 บทนำ

จากการศึกษาเรื่องระบบเอเยนต์บนอินเทอร์เน็ต จนได้รับความรู้ความเข้าใจระดับหนึ่ง เราต้องมีการพัฒนาระบบขึ้นมาระบบหนึ่งเพื่อใช้ในการสาธิตการทำงานของระบบเอเยนต์โดยอาศัยทฤษฎีและเทคโนโลยีเอเยนต์เป็นพื้นฐาน การพัฒนาระบบนี้เป็นระบบต้นแบบ เพื่อให้ผู้ที่สนใจความรู้ทางด้านเอเยนต์ได้เห็นว่า ทฤษฎีที่เรานำเสนอไปแล้วข้างต้นนั้น สามารถนำมาใช้และปฏิบัติจริงได้ นอกจากนี้ การทดลองออกแบบและพัฒนาระบบขึ้นมา ทำให้ผู้ศึกษาได้มีการนำความรู้ที่ได้จากการศึกษามาพัฒนาเป็นของจริง ซึ่งจะทำให้เกิดความเข้าใจและได้รับความรู้เพิ่มขึ้นจากการที่ไม่เคยรู้มาก่อนถ้าไม่ลองพัฒนาจริงๆขึ้นมา

ดังนั้นเราจึงทำการสร้างระบบเอเยนต์ขึ้นมา 1 ระบบ เป็นระบบที่สามารถนำมาใช้บนอินเทอร์เน็ตได้จริงๆ คือระบบเอเยนต์ที่รองรับการทำงานของเอเยนต์ได้หลายๆตัวในเวลาเดียวกัน และเอเยนต์เหล่านั้นก็ไม่จำเป็นว่าจะต้องมีการทำงานอย่างเดียวกันเสมอไป

จากระบบที่กล่าวไปข้างต้น เราก็ได้ออกแบบและพัฒนาเอเยนต์ขึ้นมา 1 ตัวเพื่อเป็นตัวอย่างของเอเยนต์ที่จะมาทำงานบนระบบที่เราได้สร้างขึ้นมา เอเยนต์ตัวนี้ เป็นเอเยนต์ชื่อ FileFinder ซึ่งจะช่วยในการหาไฟล์ในเซอร์ฟเวอร์ที่เราต้องการ ผู้ใช้งานเอเยนต์ตัวนี้เพียงเรียกชื่อเอเยนต์ขึ้นมาและบอกว่าต้องการให้เอเยนต์หาไฟล์ใดบ้าง เอเยนต์ก็จะไปหาว่ามีไฟล์ชื่อดังกล่าวอยู่ในเซอร์ฟเวอร์นั้นหรือไม่ แล้วกลับมารายงานใช้ผู้ใช้ ทราบ

3.2 ที่มา

ระบบเอเยนต์สามารถถูกพัฒนา ขึ้นมาได้หลายทางหลายประการ แต่จากการที่เราศึกษารูปแบบการทำงาน และสถาปัตยกรรมของระบบเอเยนต์ที่ผ่านมา เราได้ข้อสรุปและแนวคิดว่า ระบบเอเยนต์ที่น่าจะมีความสมบูรณ์และเป็นไปได้ในการนำมาพัฒนาเป็นของจริง น่าจะมีรูปแบบดังนี้

3.2.1 โครงสร้างของระบบน่าจะประกอบไปด้วย ส่วนเอเยนต์ที่ไคล์เอ็นต์ และส่วนเอเยนต์ที่เซอร์ฟเวอร์

3.2.2 ส่วนทั้ง 2 ส่วนต้องมีการติดต่อถึงกันและกัน โดยวิธีใดวิธีหนึ่งตามแต่จะตกลงกันไว้

3.2.3 ภาษาที่ใช้ในการเขียนโปรแกรมของระบบควรจะเป็นภาษาที่มีคุณสมบัติดังที่ได้กล่าวไปแล้ว และจะต้องมีการรองรับจากแพลต์ฟอร์มทั่วไป อีกทั้งยังต้องมีความสมบูรณ์ในตัวภาษามากพอ ที่เป็นที่นิยมของผู้พัฒนาทั่วไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากข้อสรุปดังกล่าว เราจึงเริ่มทำการพัฒนาระบบเอเยนต์ขึ้นเพื่อความสมบูรณ์ของการศึกษาค้นคว้า

3.3 จุดประสงค์

1. นำความรู้ ทฤษฎี และตัวอย่างของระบบที่ได้ศึกษาไว้ มาทำการรวบรวมและประยุกต์ใช้ เพื่อให้ได้ระบบที่สามารถนำไปใช้งานได้จริง หรือ เพื่อไปแนวทางในการพัฒนาต่อไปในอนาคต
2. สร้างขึ้นมาเป็นตัวอย่างเอเยนต์ คือ เอเยนต์ที่ช่วยค้นหาไฟล์ ทั้งๆที่จริงๆแล้วการค้นหาไฟล์ปกติทำได้โดยใช้ FTP อยู่แล้ว แต่การใช้เอเยนต์ตัวนี้ ผู้ใช้ไม่จำเป็นต้องเข้าไปหาทีละเซอร์ฟเวอร์ให้เสียเวลา เอเยนต์จะเดินทางไปยังเซอร์ฟเวอร์ต่างๆให้แทน

3.4 การเตรียมการสร้าง

1. เราต้องมีความรู้เรื่องระบบเอเยนต์, รูปแบบโครงสร้าง, ส่วนประกอบ, การทำงาน และอื่นๆ
2. การศึกษาวิธีการเขียนโปรแกรมและลักษณะการใช้งานของภาษาที่จะเป็นภาษาที่ใช้ในการเขียนโปรแกรมของระบบ ซึ่งอาจต้องใช้เวลาในการศึกษาภาษาให้เข้าใจอย่างท่องแท้และเพี ยงพอที่จะนำมาเขียนโปรแกรมต่อไป
3. การศึกษาและเรียนรู้ระบบปฏิบัติการที่จะมีเป็นระบบปฏิบัติการของระบบเอเยนต์ ไม่ว่าจะที่ตัวเซอร์ฟเวอร์และไคล์เอ็นต์ วิธีการติดตั้งเว็บเซอร์ฟเวอร์ (Web Server) และการกำหนดเลขที่ ไอพี (IP Address) ต่างๆ

3.5 การวางแผนการพัฒนาระบบ

สำหรับขั้นตอนการจัดทำระบบเอเยนต์ มีดังต่อไปนี้

3.5.1 <u>ศึกษาสถาปัตยกรรม</u> หรือโครงสร้างของระบบเอเยนต์ที่มีผู้พัฒนาอยู่บ้างแล้ว และศึกษาสถาปัตยกรรมของภาษาต่างๆ ที่จะมาใช้เขียนระบบเอเยนต์เพื่อออกแบบสถาปัตยกรรมของระบบเอเยนต์ของเราให้สอดคล้องกับภาษาที่จะเลือกใช้ต่อไป ซึ่งผลของการศึกษา เราได้นำมาเป็นส่วนความรู้พื้นฐานนี้แล้ว

3.5.2 <u>ศึกษาแต่ละภาษา</u> ที่มีคุณลักษณะเหมาะสมในการนำมาเขียนระบบ ว่าแต่ละภาษามีข้อดีข้อเสียอย่างไร และความเป็นไปได้ในการนำมาพัฒนาเป็นระบบเอเยนต์ จนเราได้ข้อสรุปว่า จะใช้ภาษาจาวาเป็นภาษาในการเขียนโปรแกรมระบบเอเยนต์ของเรา เนื่องจากว่าต้องการใช้ประโยชน์จากclassในภาษาจาวา 3 ข้อ ได้แก่ :-

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- แอ็พเพล็ต (Applet)
- โปรแกรมที่ใช้งานในเครื่องเดียว (StandAlone application)
- คลาสซึ่งสามารถใช้งานได้ทั้งในโหมดการทำงานแบบแอ็พเพล็ต และโปรแกรมเดี่ยว

ทั้งนี้การเลือกใช้ภาษาจาวาในการพัฒนาระบบเอเยนต์นี้ เนื่องจากข้อดี ของภาษาจาวาที่มีอยู่ 4 อย่างคือ:-

- มีการรองรับการทำงานบนระบบเน็ตเวิร์ก (Network aweraness)
- โปรแกรมที่ได้มีขนาดไม่ใหญ่มาก (Portability)
- มีระบบรักษาความปลอดภัย (Security)
- เป็นภาษาที่เป็นอ็อบเจ็คโอเรียนเต็ด (Object orientation)

เนื่องจากว่า คงเป็นไปไม่ได้ที่จะเขียนแอพพลิเคชันของระบบการทำงานเอเยนต์ นี้โดยปราศจากบางสิ่งที่มีอยู่ในจาวา นั่นคือ เรื่องของการติดต่อสื่อสารผ่านเครือข่าย (Network communication) สำหรับจาวาแล้วเป็นเรื่องที่ง่ายดาย เพราะในจาวาจะมีแพคเก็จ ที่สามารถใช้งานได้ จึงทำให้การสร้างโปรโตคอล ที่ใช้งานภายในระบบนั้นสามารถทำได้ง่าย กลไกต่าง ๆ ที่ใช้สำหรับทำการโหลดclass ที่มีในจาวาจะยิมยอมให้เราโหลดclassเหล่านั้นข้ามเครือข่าย และสามารถรันclassนั้น ได้อย่างสะดวกเหมือนว่า เราได้รันclass ที่อยู่ภายในเครือข่ายของเราเอง การรวมตัวกันระหว่าง classแอพเพ็ต (Applet Class) ของจาวา กับ เว็บบราวเซอร์ (Web Browser) ทำให้เราสร้างส่วนที่ติดต่อใช้งานของผู้ใช้(User interface) กับระบบ ได้เป็นอย่างดีตามแต่จะต้องการ

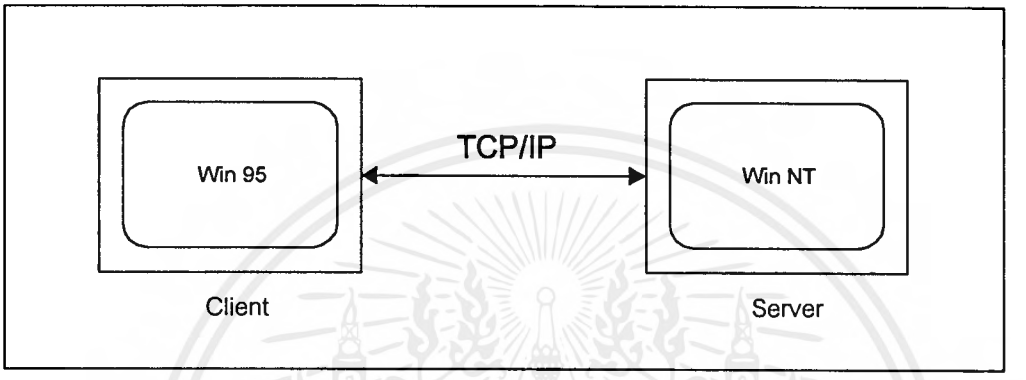
แต่สิ่งที่สำคัญเหนือประการอื่น คือ จาวามีความสามารถในเรื่องของการพกพา เคลื่อนย้ายได้ (Portability) เป็นอย่างดี สำหรับจาวาแล้วจะทำให้ :-

- เราสามารถเขียน AgentServer เพียงแค่ครั้งเดียว แล้วนำไปรันบนคอมพิวเตอร์เครื่องอื่น ที่ไหนก็ได้
- เราสามารถเขียน AgentLauncher เพียงแค่ตัวเดียว เพื่อจะให้รันอยู่ภายใต้การทำงาน กับเว็บบราวเซอร์ ที่ไหนก็ได้
- เราสามารถเขียน Agent เพียงตัวเดียว ซึ่งทำงานบนเครื่องเซอร์ฟเวอร์ที่เป็น AgentServer ที่เครื่องใดๆ ก็ได้

เมื่อส่วนประกอบทั้ง 3 นี้ อยู่ในสถานที่ใดที่หนึ่ง ใครก็ตามที่ต้องการจะสร้างเอเยนต์ ตัวใหม่ขึ้นมาใช้งาน สิ่งเดียวที่เขาต้องคำนึงถึงก็คือ อะไรคือสิ่งที่ เอเยนต์ จะต้องทำบ้าง โดยปราศจากความกังวลในส่วนของการท่องไปในเครือข่ายของเอเยนต์เลย หรือ แม้แต่ในส่วนของการแสดงผลลัพธ์ ว่าเอเยนต์จะส่งผลลัพธ์การทำงานกลับมายังตัวผู้ใช้ได้อย่างไร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.5.3 <u>เลือกแพลต์ฟอร์ม, ซอฟต์แวร์ และ เครื่องมือต่างๆ ที่จะนำพัฒนาระบบ</u> จากข้อจำกัดทางความรู้ความสามารถและประสบการณ์ของผู้ทำโครงงาน ที่ไม่มีความรู้หรือประสบการณ์การทำงานบนแพลต์ฟอร์มอื่น นอกจากระบบปฏิบัติการวินโดว์ เราจึงไม่สามารถทำงานหรือเขียนโปรแกรมในแพลต์ฟอร์มอื่นได้นอกจากวินโดว์ 95 และวินโดว์เอ็นที ที่เราเลือกมาเป็นแพลต์ฟอร์มของระบบ โดยใช้วินโดว์เอ็นที เป็นระบบปฏิบัติการของเซอร์ฟเวอร์ และใช้วินโดว์ 95 เป็นระบบปฏิบัติการของไคล์เอ็นต์



รูปที่ 32 แสดงระบบปฏิบัติการและโปรโตคอลของระบบ

คุณสมบัติของทั้ง 2 ระบบปฏิบัติการนี้ เหมาะสมที่จะนำมาเป็นระบบปฏิบัติการของเอเยนต์ เนื่องจากทั้ง 2 รองรับการทำงานแบบมัสติทาสกิ้ง ซึ่งจะทำให้เอเยนต์ ของเราสามารถทำงานได้หลายอย่างในเวลาเดียวกัน

ในด้านของคอมไพล์เลอร์ ในปัจจุบันมีบริษัทผู้ผลิตหลายรายที่ผลิตจาวาคอมไพล์เลอร์ออกมาเราได้ทำการทดสอบ คอมไพล์เลอร์จากหลายบริษัทด้วยกันและพบข้อดีข้อเสียของแต่ละคอมไพล์เลอร์จนเราได้ข้อสรุปว่า ระบบเอเยนต์ของเราจะใช้คอมไพล์เลอร์ของวิชวลจาวา (Visual Java : J++) จากบริษัทไมโครซอฟต์ เนื่องจากใช้สะดวกและมีเครื่องมือช่วยในการทำการติดต่อกับผู้ใช้ (User Interface)

3.5.4 <u>ทำการออกแบบสถาปัตยกรรมหรือโครงสร้างของระบบ</u> ในระบบเอเยนต์ที่ได้พัฒนาขึ้นมานี้ จะยินยอมให้เวปของผู้ใช้ (Web User) ทำการส่งโปรแกรมหรือเอเยนต์ ไปรันบนคอมพิวเตอร์เครื่องอื่นๆ ที่อยู่ในเครือข่ายอินเตอร์เน็ต และทำการส่งผลลัพธ์กลับมายังผู้ใช้

โดยในที่นี้ เอเยนต์ จะเป็น class หนึ่งของจาวา ซึ่งจะทำการสร้างไว้เป็นclassเอเยนต์พื้นฐาน (Base Agent Class) เพื่อความสะดวกแก่การพัฒนาเอเยนต์ ที่จะมีขึ้นใหม่ตามลักษณะที่ต้องการ ในที่นี้classเอเยนต์พื้นฐาน และ โปรแกรมเอเยนต์ทำงานบนเดี่ยวที่เซอร์ฟเวอร์ (Standalone

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Agent Server) จะเป็นส่วนที่จัดการลักษณะการกระจายการทำงานของเอเยนต์ทั้งหมดผ่านเครือข่าย

สำหรับสถาปัตยกรรมของระบบที่เราจะพัฒนานั้น ประกอบด้วยส่วน 3 ส่วนใหญ่ คือ Agent, AgentLauncher และ AgentServer ซึ่งแต่ละส่วนจะมีการทำงานต่างกันไป

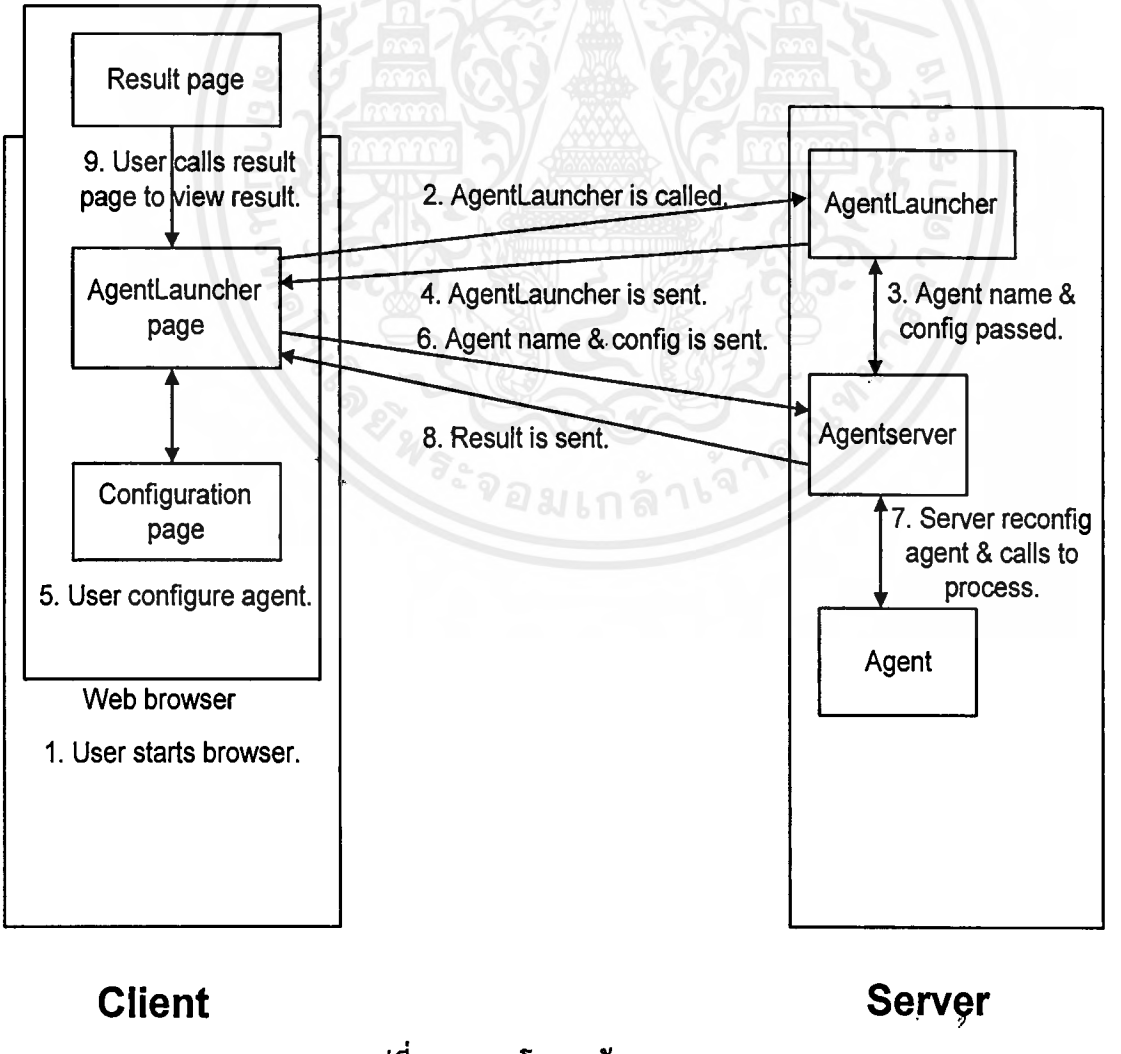
3.5.5 ออกแบบฟังก์ชั่น (Function) การทำงานภายในระบบ

3.5.6 ออกแบบการรักษาความปลอดภัยของระบบ

3.5.7 ออกแบบอัลกอริทึม (Algorithm) ของระบบ รูปแบบการเขียนโปรแกรม, ขั้นตอนการเขียนโปรแกรม, การสร้างฟังก์ชั่น, การสร้างตัวแปร และอื่น

ซึ่งข้อ 3.5.4, 3.5.5, 3.5.6 และข้อ 3.5.7 นี้ เราจะกล่าวถึงรายละเอียดในหัวข้อถัดไป

3.6 การออกแบบระบบ

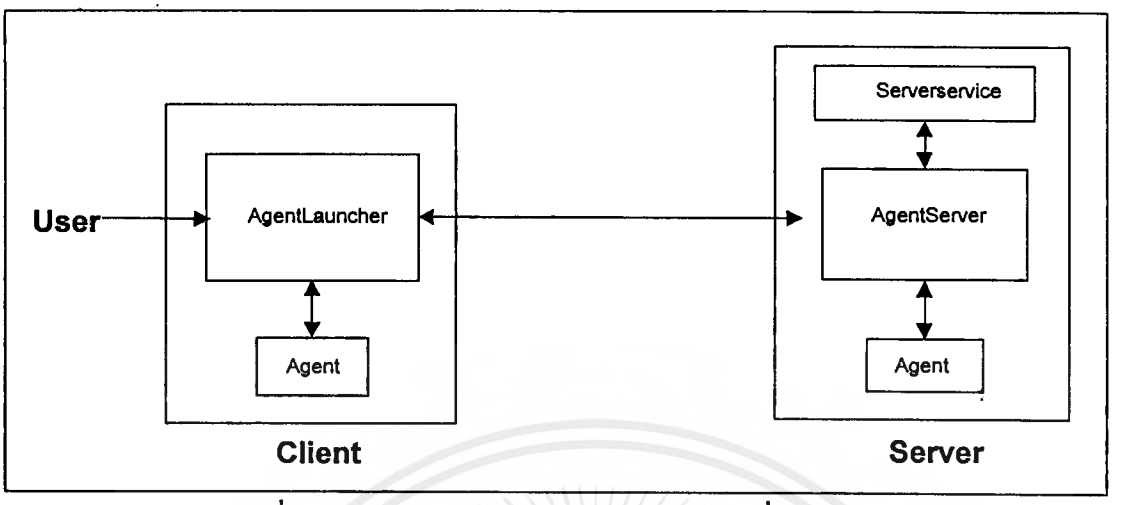


รูปที่ 33 แสดงโครงสร้างของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบของระบบ

ระบบเอเยนต์ มี 3 ส่วนใหญ่ๆ คือ



รูปที่ 34 แสดงสถาปัตยกรรมของระบบเอเยนต์ที่จะพัฒนา

3.6.1 Agent

เป็นส่วนที่จะเดินทางไปทำโพรเซส เมื่อผู้ใช้ เรียกใช้ โดยจะมีข้อมูลที่ผู้ใช้ กำหนดถูกส่งไปพร้อมกัน

<u>ฟังก์ชั่นการทำงาน</u> :

- ให้ ผู้ใช้ ทำการ Configure ตัว Agent ให้ทำงานตามที่ ผู้ใช้ ต้องการ
- รับค่าที่ ผู้ใช้ Configure นั้น
- เก็บ Context เพื่อให้ Server ตั้งค่าได้เมื่อต้องการจะติดต่อกับ Agent นี้
- เก็บค่าข้อมูลที่ ผู้ใช้ ตั้งเอาไว้เพื่อใช้ในการทำงานกับ AgentServer

3.6.2 Agent Launcher

เป็นเหมือนที่ตั้งของ Agent ซึ่ง AgentLauncher นี้ต้องเรียกใช้จากด้าน AgentServer ให้ไปทำงานในเครื่องไคล์เอ็นต์

<u>ฟังก์ชั่นการทำงาน :</u>

- ติดต่อกับผู้ใช้งาน
- รับข้อมูลเกี่ยวกับตัว Agent ในการให้ ผู้ใช้ เรียกใช้งานจาก AgentServer
- รับคำสั่งเริ่มการทำงานของ Agent จาก ผู้ใช้
- เปิด Socket ในการติดต่อกับ AgentServer
- ส่ง Agent ออกไปสู่ AgentServer โดยการเรียกโพรเซส ของ AgentServer ผ่านทาง Socket ที่เปิดไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ส่ง Message ติดต่อกับ Agent Server
- รับ Message จาก AgentServer
- โพรเซส ตามที่ได้รับ Message
- รับผลลัพธ์ที่ได้จากการทำงานของ Agent
- แสดงผลลัพท์ที่ได้จากการทำงานของ Agent
- ตั้งค่าเริ่มต้นต่างๆของการทำงานของ Agent
- ควบคุมการทำงานในสถานะต่างๆของ Agent

3.6.3 Agent Server : เป็นหน่วยที่เปิดไว้ที่เซอร์ฟเวอร์ เพื่อรอการติดต่อกับ Agent ที่ถูกส่งออกมาจากไคล์เอ็นต์

<u>ฟังก์ชั่นการทำงาน :</u>

- เปิด Socket เพื่อใช้ในการติดต่อกับ AgentLauncher
- ให้ข้อมูลเกี่ยวกับตัว Agent ที่ไว้ให้ ผู้ใช้ เรียกใช้งานกับ AgentLauncher
- บริการให้ Agent ทำงาน
- ส่งข้อความไปยัง AgentLauncher ถึง AgentLauncher บอกถึงสถานะการทำงานของ Agent และเพื่อการติดต่อกันในระบบ
- คอยรับการติดต่อจาก AgentLauncher
- เรียก Agent มาใช้งาน
- สร้างและเก็บไฟล์ที่แสดงผลการทำงานของ Agent เพื่อให้ผู้ใช้ได้เรียกดู

3.6.4 การทำงานของการค้นหาไฟล์

เราต้องสร้าง Dialog Box ขึ้นมาเพื่อรับคำสั่งจาก ผู้ใช้ ว่าไฟล์ ใดที่ ผู้ใช้ ต้องการจะให้ค้นหา จากนั้นก็เก็บเอาอาร์กิวเมนต์ (Argument) ต่างๆเหล่านั้นไปเป็นค่าพารามิเตอร์ (Parameter) ในคำสั่งสำหรับ Agent และเมื่อ Agent เดินทางมาถึงที่เซอร์เวอร์ใดๆแล้ว AgentServer ก็จะเอาพารามิเตอร์ ที่มากับ Agent ไปทำการโพรเซส ต่อไป คือ AgentServer จะติดต่อกับระบบไฟล์ (File system) ของเซอร์เวอร์และทำการหาไฟล์ที่มีชื่อดังกล่าว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.6.5 การส่งข้อมูลภายในระบบ (Message and Message Context)

ระบบเอเยนต์โพรเซสที่สำคัญที่สุดคือการติดต่อกันระหว่าง 3 ส่วนหลักโดยใช้ข้อความ (Message) ข้อความต่างๆนี้จะถูกกำหนดออกมาจากทั้ง 3 ส่วนของระบบ เราได้กำหนดให้ข้อความที่ใช้ส่งผ่านในระบบเป็นข้อความที่แบ่งได้ 7 ประเภท ดังตาราง

ข้อความ	ทิศทาง	ความหมาย
QueryAgentlist	AgentLauncher -> AgentServer	ส่งรายชื่อเอเยนต์ ที่เอเยนต์เซอร์ฟเวอร์สามารถส่งไปทำงานได้
Agentlist	AgentServer -> AgentLauncher	รายชื่อเอเยนต์ที่สามารถใช้งานได้ เท่าที่มีอยู่ในเอเยนต์เซอร์ฟเวอร์
Dispatch	AgentLauncher -> AgentServer	ส่ง classรายชื่อไปยังเซอร์ฟเวอร์ที่มีอยู่ในระบบทั้งหมด
Local	AgentServer -> AgentServer	โหลดและรันคลาส
Kill	AgentLauncher -> AgentServer	สั่งให้เอเยนต์หยุดการทำงาน
Start	AgentServer - > AgentServer ->AgentLauncher	เริ่มการทำงานของเอเยนต์แล้ว
Result	AgentServer -> AgentServer -> AgentLauncher	เอเยนต์รายงานผลลัพธ์การทำงาน ไปยัง URL ที่ผู้ใช้กำหนดไว้

ตารางที่ 2 ความหมายของข้อความ (Messages)ที่มีใช้ในระบบเอเยนต์

ข้อความต่างๆทั้ง 7 ประเภทจะมีรูปแบบ ซึ่งเราเรียกว่า Message Context ซึ่งแบ่งเป็นฟิลด์ (Field) ต่างๆ ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฟิลด์	ขนาด	คำอธิบาย
Message type	4 ไบต์	รายละเอียดที่แสดงประเภทของข้อความ เป็นรหัสแอสกี้ (ASCII) เช่น " Disp. "
Size of length	4 ไบต์	เลขจำนวนเต็ม (Integer) ที่ระบุขนาดของฟิลด์
lengh	ขนาดจำนวนไบต์	เลขจำนวนเต็ม (Integer) ที่ระบุขนาดของข้อความ
Field type	4 ไบต์	รายละเอียดที่แสดงประเภทของฟิลด์เป็นรหัสแอสกี้ (ASCII) เช่น " Clas. "
Size of length	4 ไบต์	เลขจำนวนเต็ม (Integer) ที่ระบุขนาดของฟิลด์
length	ขนาดจำนวนไบต์	เลขจำนวนเต็ม (Integer) ที่ระบุขนาดของข้อมูล ที่ใช้กับฟิลด์
Data	ความยาวไบต์	ข้อมูลที่ใช้งานจริงซึ่งได้ จากข้อความ ซึ่งจะส่งในรูปของไบต์โค้ด (Bytecodes) เกิดขึ้นเมื่อมีการส่งข้อความที่โหลดหรือ URL เมื่อต้องการแสดงผลลัพธ์การทำงาน

ตารางที่ 3 โครงสร้างข้อความ

3.6.6 การรักษาความปลอดภัยของระบบ

ในการทำงานแบบระบบเครือข่ายที่เครื่องคอมพิวเตอร์ติดต่อถึงกันได้และเรียกใช้งานทรัพยากรกันได้นั้น การรักษาความปลอดภัยเป็นสิ่งที่จำเป็นที่สุด เพราะถ้าไม่มีการป้องกันที่ดีแล้ว ระบบอาจถูกบุกรุกโดยบุคคลภายนอกหรือการทำการใดๆอันจะส่งผลให้ระบบเสียหายได้จากคนภายในและภายนอก

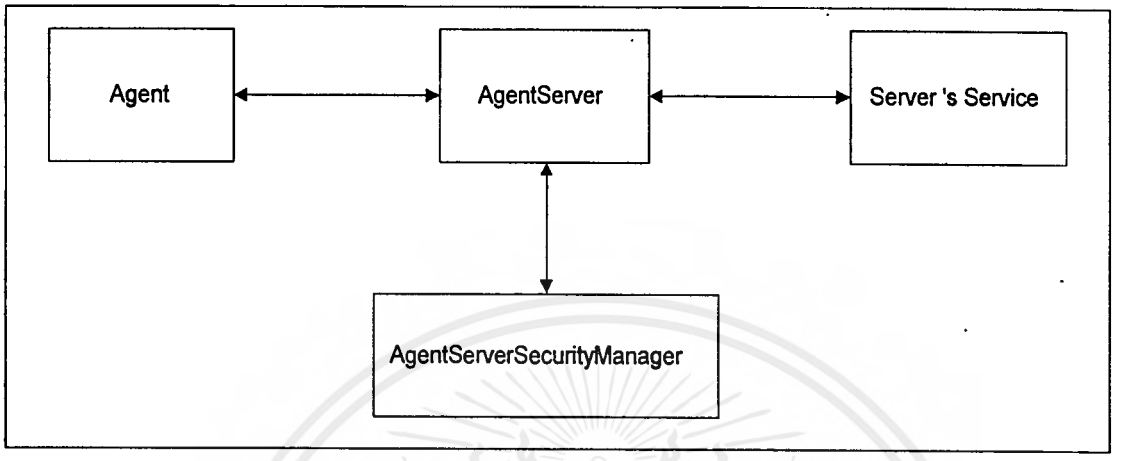
การทำงานผ่านเว็บบราวเซอร์ ก็มีขอบเขตการทำงานระดับหนึ่งเพื่อป้องกันความเสียหายที่อาจเกิดขึ้นได้ โดยการป้องกันของบราวเซอร์ มีดังนี้

- แอ็พเพล็ต ไม่สามารถเขียนหรืออ่านไฟล์ใดๆ ที่ระบบไฟล์ท้องถิ่น (Local File System)
- แอ็พเพล็ต มีความสามารถเพียงเปิดช่องการติดต่อไปยังโฮตส์ ที่แอ็พเพล็ต นั้นถูกโหลดมา

แต่ระบบความปลอดภัยเพียงเท่านั้นยังไม่เพียงพอ ในภาษาจาวา ได้กำหนดให้มีclass หนึ่งที่ทำหน้ารักษาความปลอดภัย คือ class SecurityManager ซึ่งจะมีการตรวจสอบแอ็กเซส (Access) ต่างๆ ที่เข้ามาทำงานในโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบเอเยนต์ของเรา เราได้กำหนด class AgentServerSecurity extends SecurityManager เพื่อไว้ตรวจสอบสิ่งต่างๆ ที่จะเข้ามาใน class AgentServer การตรวจสอบจะตรวจว่า ค่าหรือข้อมูลต่างๆที่ถูกส่งเข้ามาเป็นของ class Agent ที่ได้รับอนุญาตแล้วหรือไม่



รูปที่ 35 ระบบรักษาความปลอดภัยของระบบ

3.6.7 การเชื่อมต่อระหว่างไคล์เอ็นต์กับเซอร์ฟเวอร์

ในภาษาจาวา มีการรองรับการเขียนโปรแกรมบนเน็ตเวิร์ก ซึ่งใช้สำหรับการติดต่อระหว่าเครื่องหนึ่งไปยังอีกเครื่อง ซึ่งการกำหนดการติดต่อ เราต้องกำหนด " Socket " Socket เป็นเหมือนการเชื่อมต่อแบบเป็นนามธรรม คือเปรียบเสมือนเราเอาปลั๊กมาเสียบเชื่อมกันระหว่างเครื่องคอมพิวเตอร์ ซึ่ง Socket จะประกอบไปด้วย ที่อยู่ของเครื่องที่เราจะต่อด้วย และ พอร์ตที่จะใช้ในการติดต่อกันระหว่าง 2 เครื่องนี้ โดยการกำหนด Socket ติดต่อ ต้องกำหนดทั้งที่เครื่อง ไคล์เอ็นต์และเซอร์ฟเวอร์ และการกำหนดเบอร์พอร์ตต้องตรงกันระหว่าง 2 ฝ่ายด้วย 2 เครื่องจึงจะสามารถติดต่อกันได้ ซึ่งในการเขียนโปรแกรมเรามักใช้พอร์ตที่มีเบอร์มากๆ เพราะยิ่งเบอร์พอร์ตมากเท่าไหร่ โอกาสที่พอร์ตนั้นจะถูกใช้งานโดยระบบปฏิบัติการของเครื่องนั้นๆก็จะน้อยลง ในโปรแกรมของเรา เรากำหนดให้ระบบเอเยนต์ติดต่อกันโดยใช้พอร์ต 1037 ดังนั้นเวลาที่ส่วนไคล์เอ็นต์ หรือเซอร์ฟเวอร์ จะส่งข้อความถึงกัน ก็จะต้องส่งโดยใช้พอร์ตนี้

3.6.8 การจัดการสถานะต่างๆขณะโปรแกรมทำงาน

การทำงานของโปรแกรมจะให้ได้ผลดีและมีสถานะมั่นคงนั้น ควรจะมีการกำหนดสถานะต่างๆ ระหว่างการทำงานของโปรแกรม โดยเฉพาะโปรแกรมของเราที่เราต้องการให้มีการทำงานได้หลายๆอย่างพร้อมกันอย่างนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อมีการเรียกใช้งานจาก ผู้ใช้ โปรแกรมจะเข้าสู่สถานะเตรียมพร้อมและกำหนดค่าเริ่มต้นต่างๆของตัวแปรที่จะใช้ รวมทั้งมีการเรียกเอาการทำงานส่วนย่อยๆมาเตรียมไว้เพื่อรอการเรียกใช้งานต่อไป

เมื่อมีการรับอินพุทจากผู้ใช้ โปรแกรมก็จะเริ่มมีการทำงานเป็นส่วนๆตามที่เราได้ออกแบบไว้ และเมื่อการทำงานในขั้นตอนใดๆ สำเร็จลง สถานะของโปรแกรมก็จะเปลี่ยนไป เพื่อง่ายต่อการตรวจสอบว่า ขณะนั้นการทำงานของโปรแกรมได้ทำไปถึงขั้นตอนใดแล้ว และเพื่อทำให้การทำงานในขั้นตอนต่างเป็นลำดับขั้นตามต้องการด้วย

3.7 การสร้างโปรแกรม

จากการออกแบบ เราแบ่งการเรียกใช้งานได้เป็น 2 ส่วน คือส่วน ไคล์เอ็นต์ และส่วน เซอร์ฟเวอร์ ส่วน ไคล์เอ็นต์ จะมีแอ็พเพล็ต ที่ถูกเรียกเข้าสู่เว็บบราวเซอร์และรันบนเว็บบราวเซอร์นั้น ส่วนที่ เซอร์ฟเวอร์ จะเปิดโปรแกรมใช้รันอยู่ตลอดเวลาโดยไม่ต้องใช้เว็บบราวเซอร์ เรียกว่าเป็นโปรแกรมที่ทำงานบนเครื่องเดียว (Stand-alone Application)

3.7.1 การแบ่งโปรแกรมเป็นแพ็กเก็จ (Packages)

เราสามารถแบ่งส่วนต่างข้างต้นออกเป็นpackageตามความสามารถของภาษาจาวา โดยแบ่งclassที่ทำงานในส่วนเดียวกันไว้ด้วยกัน ได้ดังนี้

3.7.1.1 Package agent.Agent

ประกอบด้วย

7.1.1.1 class agent.Agent.Agent : class ของตัว Agent เอง

Method ของ class :

- Configure : จะถูกเรียกใช้โดย class AgentLauncher เพื่อเปิด Dialog Box สำหรับรับค่า Argument ที่ ผู้ใช้ กำหนด
- GetArguments : จะถูกเรียกใช้โดย class AgentLauncher เพื่อเก็บค่า Argument ที่ ผู้ใช้ กำหนดไว้จาก Method Configure
- SetAgentContext : จะถูกเรียกใช้โดย class AgentServer เพื่อ Set AgentContext ในการติดต่อระหว่าง class Agent กับ class AgentServer
- SetArgument : จะถูกเรียกใช้โดย class AgentServer เพื่อ Set Argument เกี่ยวกับ Agent นั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7.1.2 package agent.Launcher

ประกอบด้วย

7.1.2.1class agent.Launcher.AgentLauncher : class ของตัว AgentLauncher เอง

Method ของ class :

- Dispatch : Method ใช้ในการติดต่อเพื่อส่ง Agent
- changeAppState :
- start : Applet ใช้เริ่มการทำงาน เมื่อมี ผู้ใช้ เรียก Applet ขึ้นมา
- run : Main Loop ในการทำงาน
- SetRunning
- LoadAgentclass : Load class Agent ขึ้นมาเพื่อใช้งาน
- LoadAnimationImages : Load ภาพให้ปรากฏเป็นลำดับขั้น เพื่อใช้แสดงบนหน้าจอ
- StopAgent : หยุดการทำงานของ Agent
- addAgentDisplay :
- AddToState :
- endDispatch : หยุดการติดต่อ
- reportResult : Method ที่บอกให้ Agent ทำงานได้จบและรายงานผล ไม่ว่าจะมีผลลัพธ์หรือไม่ก็ตาม
- reportResults : เวลา Agent ส่งผลลัพธ์กลับมา จะมี class AgentConnectionHandler เอา Result ไปเก็บไว้ที่ Pointer ซึ่งจะ Point ไปยัง HTML Page เพื่อเก็บเป็น Result File
- startDispatch : เริ่มทำการติดต่อกับ AgentServer

class นี้มี Subclass เพื่อสนับสนุนการทำงานดังนี้

7.1.2.1.1 AgentDispatcher : มี Method ของ Subclass ดังนี้

- AgentDispatcher : เป็นตัวคอยติดต่อ Agent กับ AgentLauncher
- stopAgent : หยุดการทำงานของ Agent
- Dispatch : ติดต่อ
- ClientProcess : รับ Message ที่เข้ามายัง class AgentLauncher เพื่อตรวจสอบดูว่า Message นั้นเป็น Type ไหน แล้วจัดการส่งต่อไปยัง Process ต่างๆ ตามประเภทต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.1.2.1.2 Animation : ทำงานด้านการเปิดภาพหรือ icon ที่เป็นส่วน Interface กับ ผู้ใช้ โดยเรียกทำงานกับ class ต่างๆ ที่เป็น Object สำหรับแสดงบนหน้าจอ เช่น class AgentButton, class TestButton, class MessageBox, class TextOnlyButton, class OKButton, class PickDialog เป็นต้น

Method ของ Subclass ดังนี้

- AgentImage : Method แสดงภาพของ Agent
- FrameNumber : สำหรับตรวจนับดูว่าขณะนี้แสดงภาพถึง Frame ที่เท่าไหร่แล้ว
- Animation : ทำให้ภาพเคลื่อนไหว
- paint
- nextImage
- updateNextImage

3.7.1.3 package agent.Server

ประกอบด้วย

7.1.3.1 class agent.Server.AgentServer : class ของตัว AgentServer

Method ของ class :

- addDispatchedAgent : ใช้เพิ่ม Agent ที่จะใช้ติดต่อระหว่าง AgentLauncher กับ AgentServer
- addRunningAgent
- deleteDispatchedAgent
- deleteRunningAgent
- notifyResult : ใช้ส่ง Message ไปยัง AgentLauncher ว่าการทำงานของ Agent ที่ AgentLauncher ส่งออกได้เสร็จและมีผลลัพธ์ออกมาแล้ว
- notifyStart : ใช้ส่ง Message ไปยัง AgentLauncher ว่า Agent ที่ AgentLauncher ส่งออกมา ได้เริ่มทำงานแล้ว
- reportResults : Method ในการรายงานผลลัพธ์

7.1.3.2 Interface agent.Server.AgentContext : Interface ที่ Agent ใช้เรียก Process หรือ Service จาก Server

Method ของ Interface :

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- dispatch : เพื่อบอกให้ class AgentServer ติดต่อกับ Agent นั้นๆ
- getResultsURL : รับสตริง URL ของไฟล์เอาท์พุท เพื่อตัวเอเยนต์จะส่ง URL นี้กลับไปยัง AgentLauncher ผ่านทางMethod reportFinish
- reportFinish : เพื่อบอกให้ AgentLauncher รับทราบว่า เอเยนต์ทำงานเสร็จแล้ว และได้ส่ง URL ของไฟล์เอาท์พุท (ถ้ามี)
- reportStart : เพื่อบอกให้ AgentLauncher รับทราบว่า ตอนนี้ตัวเอเยนต์กำลังทำงานอยู่
- writeOutput : เขียน ไฟล์ HTML ที่ละบรรทัดไปยังไฟล์เอาท์พุทซึ่ง AgentServer ได้เปิดไว้ก่อนแล้ว

3.7.1.4 package agent.util

เป็น package ที่ทำงานกับการติดต่อกับระหว่าง 3 ส่วนประกอบหลัก ซึ่งทุกครั้งที่มีการติดต่อหรือส่งข้อมูลกันระหว่าง 3 ส่วนนั้นจะต้องใช้ package นี้ เป็นตัวสร้าง Message เหล่านั้น package agent.util นี้ประกอบด้วย

7.1.4.1 class Message : class ที่ใช้สร้าง Message และ Message Construction

7.1.4.2 class AgentListMessage : class ของ Message ที่ AgentServer ส่งไปให้ AgentLauncher จะมีรายชื่อของ Agent ที่สามารถมาทำงานกับเซอร์ฟเวอร์ นี้ได้ (ตามความเป็นจริงแล้ว ในระบบของเรามี Agent เพียง 1 ตัวเท่านั้น แต่ class นี้มีไว้เพื่อการพัฒนาในอนาคตที่จะสามารถรองรับการมี Agent ได้หลายตัว)

7.1.4.3 class DispatchMessage : class ของ Message ที่บรรจุชื่อของ Agent ซึ่ง AgentLauncher ส่ง Message นี้ให้ AgentServer เพื่อติดต่อโดยใช้ Agent ตัวชื่อเดียวกับชื่อใน Message นี้

7.1.4.4 class KillMessage : class ของ Message ที่บรรจุชื่อของ Agent ซึ่ง AgentLauncher ส่ง Message นี้ให้ AgentServer เพื่อให้ AgentServer หยุดการทำงานของ Agent ตัวชื่อเดียวกับชื่อใน Message นี้

7.1.4.5 class LoadMessage : class ของ Message ที่บรรจุชื่อของ class ไว้เพื่อให้ฝ่ายที่ได้รับทำการ Load, Initiate, Run class ที่ชื่อเดียวกับที่กำหนดไว้ใน Message นี้

7.1.4.6 class QueryAgentList Message : class ของ Message ที่ใช้เพื่อขอคำตอบเป็น Message AgentListMessage

7.1.4.7 class Result Message : class ของ Message ผลลัพธ์ ที่ Agent ส่งกลับมายัง AgentLauncher

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.1.4.8 class Start Message : class ของ Message ที่ถูกส่งกลับมายัง AgentLauncher เพื่อบอกว่า Agent ได้เดินทางไปถึง เซอร์ฟเวอร์ ที่กำหนดแล้ว และเริ่ม Process ที่ เซอร์ฟเวอร์ นั้นแล้วด้วย

3.7.1.5 package agent.FileFinder

ประกอบด้วย

7.1.5.1 class FileFinder : class ของ Agent ที่เป็น FileFinder Agent (เรียกว่าเป็นส่วนขยายของ class Agent) ซึ่งใน class จะมีการกำหนด Configuration ของ FileFinder Agent นี้ด้วย

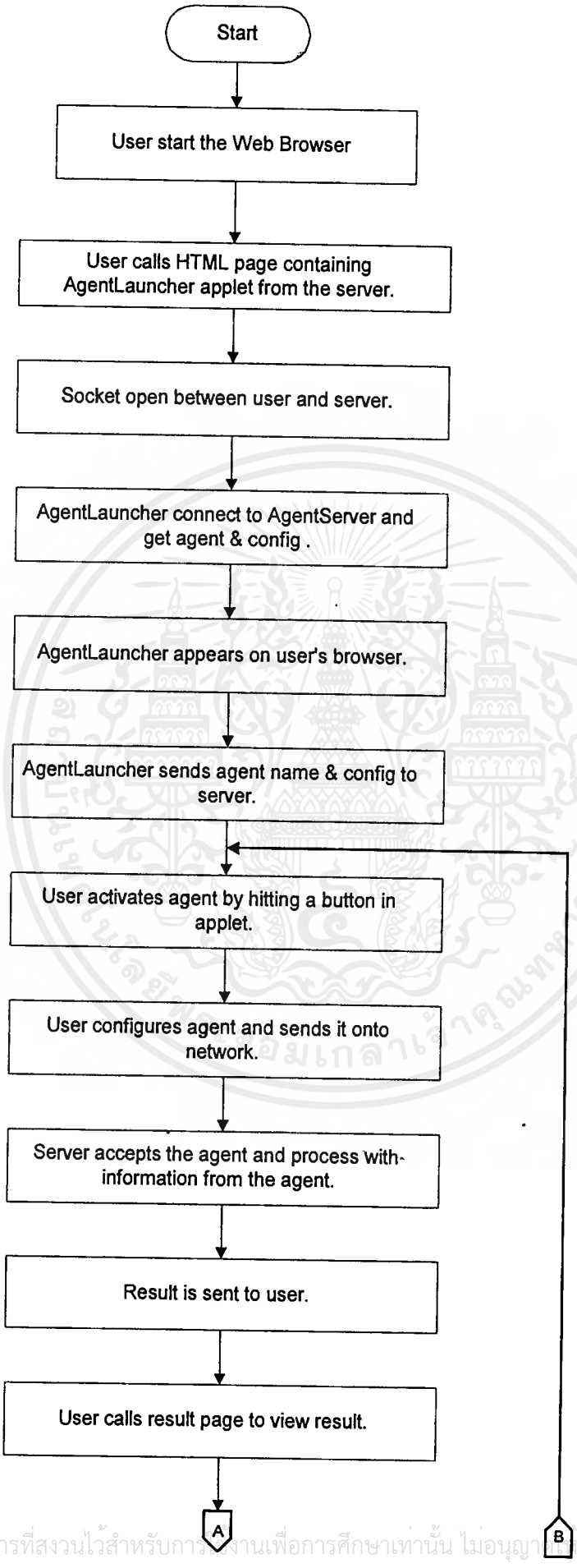
Method ของ class :

- Configure : จะถูกเรียกใช้โดย class AgentLauncher เพื่อเปิด Dialog Box สำหรับรับค่า Argument ที่ ผู้ใช้ กำหนด
- GetArguments : จะถูกเรียกใช้โดย class AgentLauncher เพื่อเก็บค่า Argument ที่ ผู้ใช้ กำหนดไว้จาก Method Configure
- run : Main Loop ของ Agent
- setArguments : จะถูกเรียกใช้โดย class AgentServer เพื่อ Set Argument เกี่ยวกับ Agent นั้น

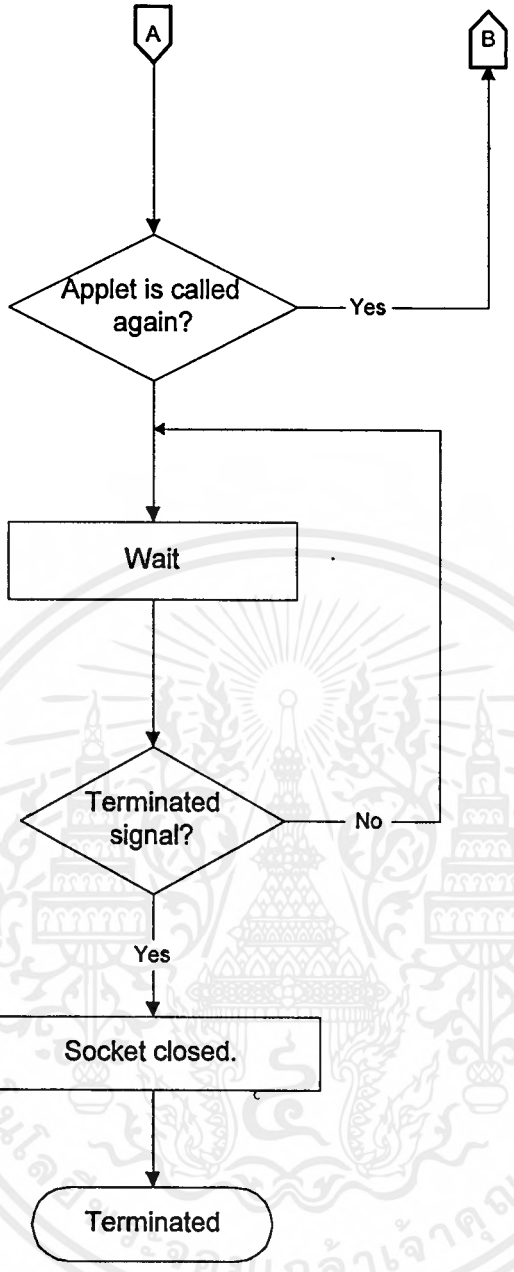
3.7.2 Algorithm ของโปรแกรม

โปรแกรมค้นหาไฟล์ใน เซอร์ฟเวอร์ มีโฟล์วชาร์ต (Flow Chart) ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 36 โฟลว์ชาร์ตแสดงการทำงานของระบบ

3.7.3 ขั้นตอนการทำงานของระบบ

1. ผู้ใช้ เรียกไฟล์ HTML ที่บรรจุ AgentLauncher applet ทางเว็บบราวเซอร์จากเว็บเซอร์ฟเวอร์ (AgentLauncher class Called)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Class AgentLauncher ส่ง Message ไปยัง class AgentServer เพื่อขอชื่อและรายละเอียดของ class Agent ที่ AgentServer จะใช้ในการติดต่อกันระหว่าง AgentServer กับ AgentLauncher
3. class AgentServer ส่ง Message ที่มีชื่อและรายละเอียดของ Agent ที่จะใช้ไปยัง AgentLauncher
4. เว็บบราวเซอร์เปิด Socket เพื่อทำการโหลด HTML Page จาก เซอร์ฟเวอร์ ซึ่งจะ class AgentLauncher ถูกโหลดและรัน
5. ผู้ใช้ สั่งเริ่มการทำงานของ Agent โดยการกดปุ่ม Agent Connect
6. Configuration Dialog Box จะปรากฏให้ ผู้ใช้ ได้ทำการสั่งรายละเอียดเป็นชื่อหรือส่วนของชื่อของไฟล์ที่ต้องการค้นหา (ใช้ Method Configure จากclass FileFinder extends Agent)
7. เมื่อ ผู้ใช้ กดปุ่ม OK ก็จะส่งให้ ข้อมูลต่างๆที่ ผู้ใช้ ใส่ใน Dialog Box นั้นถูก Set ลง Agent โดย Method getArgument และ setArgument class Agent
8. class AgentLauncher จะทำการส่ง Message ไปยัง class AgentServer ว่าจะส่ง Agent ไปยัง เซอร์ฟเวอร์ นั้น โดยใช้ Method Dispatch ของ class AgentLauncher
9. class Agent จะถูกโหลดไปยัง เซอร์ฟเวอร์
10. class AgentServer เมื่อได้รับ Agent แล้วจะทำการ Process โดยใช้ Parameter ที่ถูก Set มากับ Agent นั้น
11. เมื่อ Process ของ Agent ที่ Run ที่ เซอร์ฟเวอร์ เสร็จแล้ว Interface AgentContext ของ class AgentServer จะทำการเขียน Output String ลง File ที่ฝั่ง เซอร์ฟเวอร์ เป็น File HTML Page
12. class AgentServerส่ง Messageไปยังclass AgentLauncherเพื่อบอกว่า โพรเซสจากการทำงานของ Agent เสร็จแล้ว
13. class AgentLauncher จะแสดง Dialog Box บนหน้าจอว่าโพรเซสเสร็จแล้ว
14. ผู้ใช้ ทำการดูผลลัพธ์ได้โดยการเรียก HTML Page ที่เป็น Page ที่เก็บผลลัพธ์จากฝั่ง เซอร์ฟเวอร์
15. class AgentLauncher จะยังทำงานอยู่ โดยจะรอให้ ผู้ใช้ สั่งคำสั่งเพื่อเริ่มการทำงานของ Agent อีก
16. ถ้า ผู้ใช้ ปิดโปรแกรม class AgentLauncher ก็จะปิดตัวเอง และ Socket ที่ติดต่อระหว่างตัว ไคล์เอ็นต์ กับ เซอร์ฟเวอร์ ก็จะปิดลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7.4 การนำ Multithreading มาใช้ในระบบ

เนื่องจากระบบเอเยนต์ประกอบด้วยโปรแกรมหลายโปรแกรมทำงานร่วมกัน การที่จะใช้ระบบทำงานอย่างมีประสิทธิภาพ คือการที่เอเยนต์สามารถทำงานได้ทุกขณะที่ ผู้ใช้ ต้องการเรียกใช้ โดยไม่ต้องรอให้เอเยนต์ทำงานเก่าให้เสร็จสมบูรณ์เสียก่อนจึงค่อยสั่งงานใหม่ได้

ภาษาจาวารองรับการเขียนโปรแกรมให้มีลักษณะดังกล่าว เรียกว่าเป็นมัลติเธรด (Multithread) ซึ่งเอเยนต์สามารถรับคำสั่งได้ทุกเมื่อ และเราก็ได้นำคุณลักษณะในข้อนี้มาเขียนโปรแกรมระบบของเราด้วย นั่นคือเอเยนต์จะมีเมนลูป (Main Loop) ที่เป็นเธรด ในการทำงานอยู่ 1 เธรด โดยขณะที่เอเยนต์ทำงานอยู่ ณ ที่ใดก็ตามของเธรด ผู้ใช้ ก็สามารถสั่งงานให้เอเยนต์ ทำงานได้ตั้งแต่แรก โดยที่การทำงาน ณ ที่เดิมนั้นก็ยังคงอยู่ และดำเนินต่อไปได้

แต่ก็อาจมีปัญหาที่ว่า ในกรณีที่เราสั่งให้ เอเยนต์ ทำงานมากกว่า 1 ครั้งแล้ว เอเยนต์ อาจส่งผลลัพธ์กลับมาไม่เป็นไปตามลำดับคำสั่งที่เราสั่งออกไป

วิธีแก้ปัญหานี้คือ การกำหนด Method synchronized ในการทำ Method reportResults ใน class AgentLauncher วิธีนี้ Method reportResults จะดูลำดับการเริ่มทำงานของเอเยนต์ แต่ละครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่4

การทดลองและผลการทดลอง

เราต้องมีการทดสอบโปรแกรมที่เราได้ทดลองออกแบบและสร้างเพื่อทดสอบความรู้ทฤษฎีเอเยนต์ที่เราศึกษามา โดยได้ทดลองติดตั้งโปรแกรมลงระบบเน็ตเวิร์ก โดยทำการสร้างเซอร์ฟเวอร์สมมติขึ้นมาเพื่อเป็นเซอร์ฟเวอร์ในการทดลองครั้งนี้

4.1 จุดประสงค์ของการทดลอง

4.1.1 เพื่อทดสอบโปรแกรมที่สร้างขึ้นตามความรู้ทฤษฎีเอเยนต์ว่าจากการที่ได้ศึกษาความรู้พื้นฐานเกี่ยวกับเทคโนโลยีเอเยนต์ถึงรูปแบบการทำงานต่างๆนั้น รูปแบบที่เรานำมาออกแบบเป็นระบบเอเยนต์จะมีความสมบูรณ์และเหมาะสมกับการพัฒนาเป็นระบบที่ใช้งานจริงๆต่อไปหรือไม่อย่างไร

4.1.2 เพื่อเป็นแนวทางอ้างอิงในการสรุปและวิเคราะห์โดยใช้เหตุผลและผลการทดลองเป็นพื้นฐาน สำหรับการศึกษาความรู้เรื่องเทคโนโลยีเอเยนต์

4.2 การเตรียมอุปกรณ์และสภาวะการทำงานเพื่อทดลองโปรแกรม

เตรียมอุปกรณ์ที่จะใช้ในการรันระบบ ประกอบด้วย

- เครื่องคอมพิวเตอร์ 2 เครื่องขึ้นไป ต่อเครือข่ายกัน เครื่องที่เป็นเซอร์ฟเวอร์ ต้องมีวินโดว์เอ็นทีเป็นระบบปฏิบัติการ ส่วนเครื่องที่เป็นไคล์เอ็น ต้องมีวินโดว์ 95 เป็นระบบปฏิบัติการ

- กำหนดเซอร์ฟเวอร์ คือเครื่องที่เราต้องการให้เป็นเครื่องเซอร์ฟเวอร์นั้น จะต้องเป็นเว็บเซอร์ฟเวอร์ที่รองรับโปรโตคอล TCP/IP สิ่งที่ต้องกำหนดที่ เซอร์ฟเวอร์ นี้ มีดังนี้

- จาวาคอมไพล์เลอร์ ที่เราใช้คือ JDK 1.02 ของบริษัทซันไมโครซิสเต็ม โดยมีการกำหนด Classpath แล้วเรียบร้อย เป็น

  "CLASSPATH=C:\JDK102\java\lib\classes.zip;"

  สามารถรันไฟล์ที่เป็น Java Class ได้ (สำหรับรายละเอียดการใช้งานคอมไพเลอร์นี้ กรุณาอ่านที่ Web Site http://www.java.sun.com

- พอร์ตที่ต้องการให้เซอร์ฟเวอร์ใช้ติดต่อกับไคลเอ็นต์ จะต้องเป็นพอร์ตที่ไม่มีการใช้งานอื่นๆ (ส่วนใหญ่แล้วควรกำหนดค่าของพอร์ต สูงๆ เช่นที่ใช้ในโปรแกรมที่ทดลองนี้คือ 1037 ค่าหมายเลขพอร์ต ยิ่งสูงยิ่งไม่ค่อยมีการใช้งาน) ซึ่งในโปรแกรมเราได้กำหนดพอร์ตนี้เอาไว้แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

■ กำหนด Working Directory เพื่อรองรับไฟล์ต่างๆ ต่างๆ ที่จะได้จากการ คอมไพล์ โปรแกรมและการรันโปรแกรม

- ที่ไคล์เอ็นต์ต้องมีโปรแกรมเว็บบราวเซอร์ที่เป็น Java Enabled หรือสามารถเรียกใช้งาน Java Applet ได้ เช่น Netscape Navaigator version 2.0 ขึ้นไป, HotJava หรือ Microsoft Internet Explorer version 3.0 ขึ้นไป

4.3 ทำการทดลองโปรแกรม

4.3.1 ที่เครื่องเซอร์ฟเวอร์ รัน AgentServer.bat ซึ่งเป็น Batch File บรรจุคำสั่งในการรัน Class AgentServer แล้วปล่อยให้รันไว้อย่างนั้น โดยเรียกจากคอมมานด์พร็อมพ์

4.3.2 ที่เครื่องไคล์เอ็นต์เปิดโปรแกรมเว็บบราวเซอร์โดยกำหนด URL เป็น IP Address หรือ DNS ของเซอร์ฟเวอร์เรียกไปที่ไฟล์ AgentLauncher.HTM โดยกำหนด Directory ตาม Directory ที่เซอร์ฟเวอร์ โดยเรียกผ่านโปรโตคอล HTTP

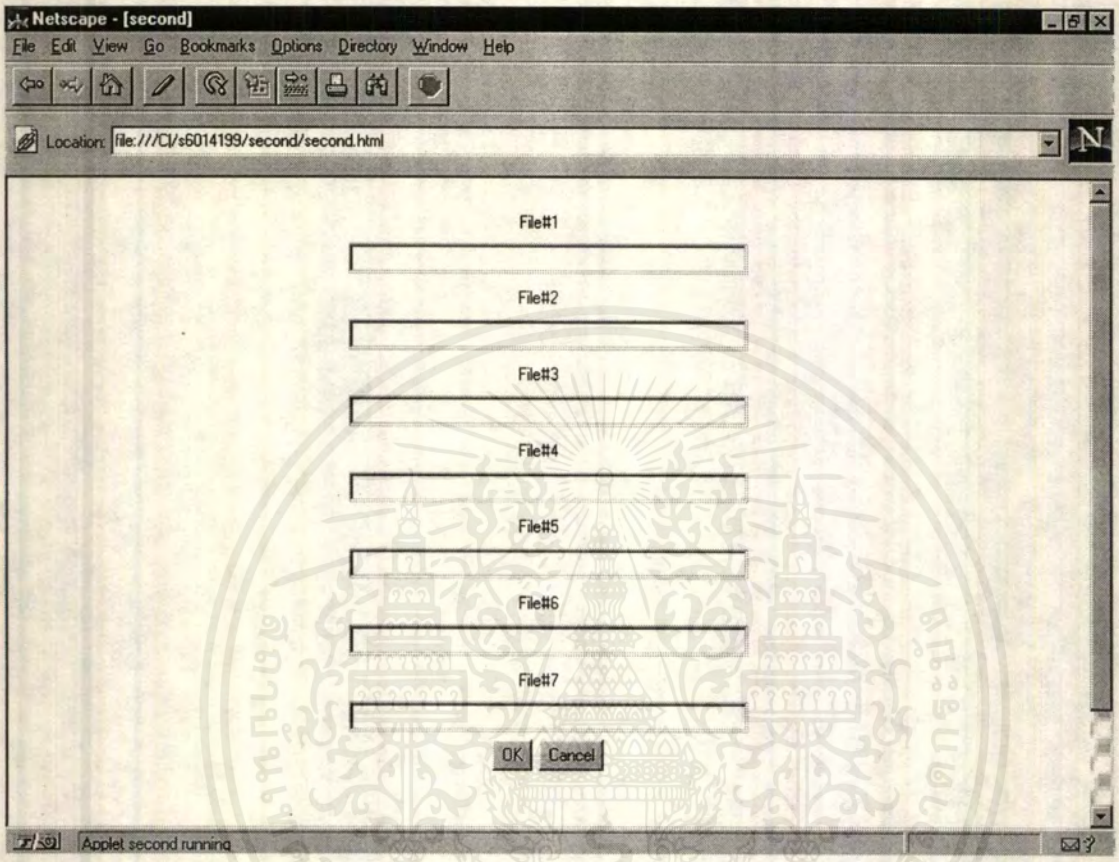
4.3.3 หน้าจอแรกปรากฏตามรูป

รูปที่ 22 หน้าจอที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.4 เราเริ่มการทำงานของ Agent ด้วยการกดปุ่ม " PICK " จะมี Dialog Box ปรากฏขึ้น เพื่อให้ใส่ชื่อไฟล์ ที่ต้องการค้นหาที่เซอร์ฟเวอร์นั้น ซึ่งชื่อไฟล์ ที่เราใส่ไปใน TextField นี้ก็อาจจะไม่เป็นชื่อเต็มๆ ก็ได้

หน้าจอของการให้ชื่อไฟล์ เป็นดังรูป



รูปที่ 23 หน้าจอที่ 2

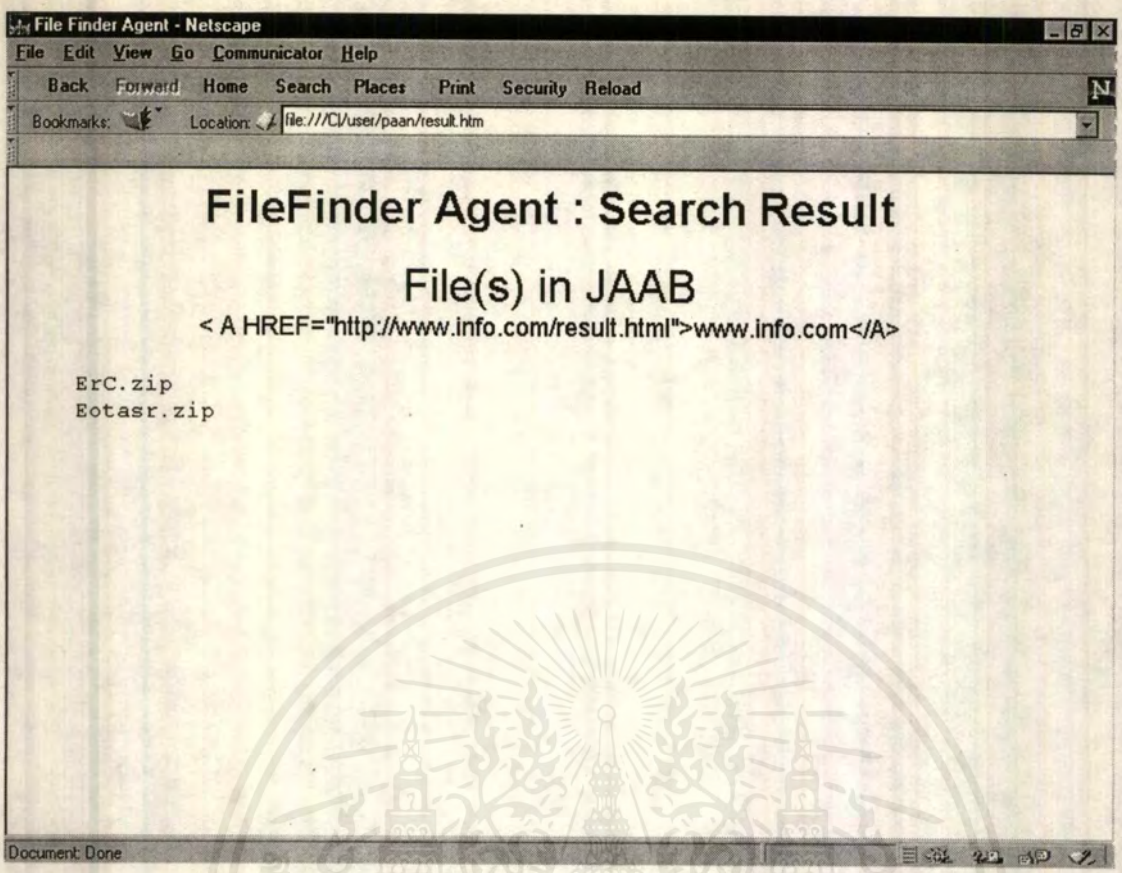
4.3.5 เราได้ทดลองใส่ E*.zip เพื่อให้เอเยนต์ค้นหาไฟล์ ที่มีชื่ออยู่ในกลุ่มดังกล่าว เมื่อคลิกที่ปุ่ม OK แล้ว หน้าจอก็จะกลับมาเป็นดังหน้าจอรูปแรก

4.3.6 เราต้องกดปุ่ม "DISPATCH" เพื่อส่งเอเยนต์ออกไปทำงานที่เซอร์ฟเวอร์

4.3.7 จากนั้นเราจึงเรียก HTML Page มาจาก Server นั้นอีก โดยการกดปุ่ม "RESULT" คราวนี้เรียกไฟล์ชื่อ Agent.HTM ซึ่งเป็นไฟล์ที่ Class AgentServer ได้ทำการเขียนลงใน Working Directory เพื่อเป็น ResultFile หรือไฟล์ผลลัพธ์ ฉะนั้นเราจึงต้องเรียก HTML Page นี้ที่ Directory ที่กำหนดไว้

หน้าจอของ ResultFile จะมีดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 24 หน้าจอสุดท้าย

หน้าจอนี้มีการแสดงชื่อของเครื่องที่เป็นเซอร์ฟเวอร์ด้วย คือ JAAB และแสดง URL ของเซอร์ฟเวอร์นั้น รวมทั้งผลลัพธ์ที่ได้จากการทำงานของเอเยนต์คือ รายชื่อของไฟล์ ที่มีชื่อเหมือนกับที่เราได้ป้อนเข้าไป

4.4 ผลการทดลอง

เมื่อตรวจสอบดูที่เซอร์ฟเวอร์ก็พบว่า การทำงานของระบบถูกต้อง ผลลัพธ์ที่ได้เป็นผลลัพธ์ที่ถูกต้อง โดยเราสามารถแยกการตรวจสอบเป็นส่วนๆดังนี้

4.4.1 เมื่อเราเรียกใช้เอเยนต์โดยเรียกหน้าจอ HTML ผ่านทางเว็บบราวเซอร์ก็จะได้หน้าจอเอเยนต์พร้อมสำหรับรอรับคำสั่ง

4.4.2 เมื่อเรากดปุ่ม “PICK” Dialog Box สำหรับการใส่ชื่อไฟล์เพื่อการค้นหาก็ปรากฏขึ้นตามที่เราได้กำหนดไว้ ซึ่งหมายความว่า Class AgentLauncher ได้มีการเรียกใช้ Class FileFinder แล้ว

4.4.3 เมื่อกดปุ่ม “DISPATCH” หน้าจอทางฝั่งเซอร์ฟเวอร์มีการเปลี่ยนแปลงคือจากมีการแจ้งว่า ไม่มีเอเยนต์ทำงานอยู่บนเครื่องเซอร์ฟเวอร์นั้น ก็เปลี่ยนเป็นการแจ้งบอกว่ามีเอเยนต์ทำงานอยู่ 1 ตัว ซึ่งหมายความว่าการติดต่อระหว่างไคล์เอ็นต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กับเซอร์ฟเวอร์ประสบผลสำเร็จ และแสดงว่าพอร์ต 1037 ที่กำหนดนั้นใช้สำหรับงานเอเยนต์นี้ได้ ไม่ได้กำหนดพอร์ตซ้ำกับการทำงานของเครื่องแต่อย่างใด จึงไม่ต้องมีการกำหนดพอร์ตใหม่

4.4.4 เมื่อกดปุ่ม "RESULT" ก็ปรากฏหน้าจอ HTML ไฟล์ใหม่ ชื่อ Result.html ซึ่งเป็นหน้าจอแสดงผล หมายความว่า Class AgentLauncher ได้ทำการโหลดหน้าจอนี้ได้สำเร็จ

4.4.5 เมื่อทดลองกดปุ่ม "STOP" ให้หยุดการทำงานของเอเยนต์ ที่ส่งออกไป หน้าจอที่เซอร์ฟเวอร์ก็จะเปลี่ยนแปลงจากการที่มีตัวหนังสือบอกว่ามีเอเยนต์ทำงานอยู่บนเครื่องเซอร์ฟเวอร์ 1 ตัวก็จะเปลี่ยนเป็นไม่มีเอเยนต์ทำงานอยู่บนเครื่องเซอร์ฟเวอร์เลย

4.4.6 ถ้าเราทำการ "PICK" และ "DISPATCH" ใหม่ ก็จะมีการทำงาน ของเอเยนต์ที่เซอร์ฟเวอร์เป็นจำนวนเท่ากับที่เราส่งออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่5
บทวิจารณ์และสรุป

5.1 บทสรุปและวิเคราะห์ของตัวโปรแกรม

โปรแกรมที่เราได้ทำการออกแบบและพัฒนาโดยใช้หลักการและทฤษฎีความรู้ทางเทคโนโลยีเอเยนต์เป็นพื้นฐานโดยมีโครงสร้างและรูปแบบการพัฒนาและทดสอบไปตามที่ได้กล่าวในบทที่ 3 และ 4 แล้วนั้น เราได้ข้อสรุปและวิเคราะห์เกี่ยวกับตัวโปรแกรมที่เราสร้างดังนี้

1. ระบบนี้เป็นระบบที่ถูกออกแบบมาเพื่อสาธิตการทำงานแบบ เอเยนต์ ว่าในระบบเอเยนต์ ควรจะมีส่วนประกอบอะไรบ้าง, แต่ละส่วนทำงานอะไรอย่างไร และแต่ละส่วนติดต่อถึงกันได้อย่างไร ซึ่งการสร้างและพัฒนาระบบนี้ ทำให้เราเข้าใจหลักการทำงานของ เอเยนต์ มากขึ้น ถึงแม้ว่าระบบนี้อาจจะไม่เป็นประโยชน์นัก แต่ก็เป็นตัวอย่างได้ดี

2. การพัฒนาระบบเอเยนต์ของเราที่ใช้โครงสร้างอันประกอบไปด้วย AgentLauncher, AgentServer และ Agent ซึ่งมีส่วนประกอบย่อยเพื่อช่วยในการทำงานต่างๆเป็นไปตามรูปแบบที่ควรจะมีอยู่ในระบบเอเยนต์นั้น เราได้พบข้อจำกัดเกี่ยวกับโครงสร้างและรูปแบบที่เราได้ทำการสร้างต่างๆดังนี้

2.1 ในการออกแบบ เราต้องการให้ Class AgentLauncher รวมทั้งรายละเอียดเกี่ยวกับตัวเอเยนต์อยู่ทางด้านของไคล์เอ็นต์ เพื่อให้เป็นการทำงานแบบ Agent-Server อย่างสมบูรณ์ นั่นคือไคล์เอ็นต์เป็นฝ่ายปล่อยเอเยนต์ออกมาแต่เราประสบปัญหาในเรื่องข้อจำกัดทาง File I/O เมื่อใช้เว็บบราวเซอร์ ซึ่งเราไม่สามารถเขียนข้อมูลที่ได้เป็นผลลัพธ์ลงสู่ไฟล์ ที่เป็น Result File หลังจากที่ AgentServer ประมวลผลแล้ว

2.2 ในระบบที่ติดตั้งในการทำโครงงานในครั้งนี้ เราได้ติดตั้งเซอร์ฟเวอร์ เพียง 1 เครื่องเท่านั้น เนื่องจากปัญหาทางด้านงบประมาณ ไม่มีเครื่องพอที่จะกำหนดเป็นเซอร์ฟเวอร์ได้หลายๆเครื่องได้

3. จากการที่เราได้ใช้ภาษาจาวาในการเขียนโปรแกรมทั้งระบบ เราพบข้อจำกัดมากมาย เช่นกันดังนี้

3.1 Class AgentLauncher สามารถเปิด Socket เพื่อติดต่อกับเซอร์ฟเวอร์ ได้เพียงครั้งละ 1 Socket เท่านั้น นั่นหมายถึง เมื่อเราสั่งให้ เอเยนต์ ทำงาน เอเยนต์ นี้จะสามารถเดินทางไปตามเซอร์ฟเวอร์ต่างๆ ได้ทีละ 1 เซอร์ฟเวอร์เท่านั้น

3.2 การกำหนดเซอร์ฟเวอร์ของระบบนี้เป็นส่วนที่สำคัญที่สุด โดยเฉพาะการกำหนด Directory ในตัวคอมไพล์เลอร์ ซึ่งต้องมีการกำหนด Directory สำหรับทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

งานของระบบโดยเฉพาะ ถ้ามีการแก้ไขไปแม้นิดเดียวก็จะทำให้ระบบไม่สามารถทำงานได้ ทำให้ดูเหมือนระบบไม่เสถียร

3.3 การใช้ภาษาจาวาเขียนเป็นแอ็พเพล็ต เพื่อการทำงานโดยผ่านทางการเรียกใช้ทางเว็บบราวเซอร์นั้น เราพบว่าการทำงานไม่เสถียรเป็นอย่างมาก จากการเรียกใช้งานบางครั้งจากเครื่องไคล์เอ็นต์ บางเครื่องที่มีโปรแกรมเว็บบราวเซอร์ ต่างๆ ไม่สามารถส่งข้อความหรือติดต่อผ่านกันทางเว็บบราวเซอร์ได้คือ การติดต่อทางพอร์ตนั้น เว็บบราวเซอร์ก็ไม่มีการตอบสนองการเปิดพอร์ตติดต่อกับเซอร์ฟเวอร์ เนื่องจากเว็บบราวเซอร์มีการกำหนดพอร์ตที่ใช้สำหรับการติดต่อขอ Web Page จากเซอร์ฟเวอร์อยู่แล้ว ดังนั้นแอ็พเพล็ตที่ทำงานบนเว็บบราวเซอร์จะเปิดพอร์ตอีกพอร์ตเพื่อใช้ในการทำงานของเอเยนต์ อาจจะไม่สามารถเปิดได้ จึงทำให้การติดต่อระหว่าง ไคล์เอ็นต์ และ เซอร์ฟเวอร์ อาจไม่ประสบผลสำเร็จได้ในบางเครื่องไคล์เอ็นต์

3.4 การเขียนโปรแกรมที่ออกแบบให้เป็นมัลติทาสกิ้งนั้น การทำงานของโปรแกรมจะต้องขึ้นอยู่กับระบบปฏิบัติการโดยตรง ซึ่งเราพบว่าระบบเอเยนต์ของเรา มีการทำงานที่ประกอบด้วยเธรดหลายๆเธรดทำงานกันโดยมีระบบปฏิบัติการเป็นผู้เรียกเธรดต่างๆขึ้นมาทำงาน การทำงานในแต่ละเธรดบาง เธรดก็เป็นอิสระจากเธรดอื่นๆ แต่บางเธรดก็ต้องรอการเปลี่ยนสถานะของโปรแกรมเป็นบางสถานะจึงจะเริ่มทำงานได้ ซึ่งการเปลี่ยนสถานะนี้ จะได้จากการทำงานของเธรดอื่นๆ นั่นเอง

เราพบปัญหาที่ว่า เราไม่สามารถควบคุมให้ระบบปฏิบัติการรันเธรดตามลำดับที่เราต้องการได้ จึงส่งผลให้บางครั้งการทำงานของโปรแกรมไม่ประสบผลสำเร็จ เนื่องจากระบบปฏิบัติการทำงานเธรดต่างๆโดยไม่เป็นลำดับขั้น ดังนั้นการสั่งให้โปรแกรมทำงานบางครั้งอาจได้รับข้อผิดพลาดได้

5.2 การขยายขอบเขตของระบบ FileFinder ในอนาคต

จาวาเป็นภาษาสำหรับเขียนโปรแกรมใหม่ที่กำลังมีผู้พัฒนามากมายทั่วโลก ดังนั้นประสิทธิภาพของภาษาก็มีแนวโน้มที่ดีขึ้นกว่าปัจจุบันมากๆ โดยเฉพาะอย่างยิ่งผู้พัฒนาเหล่านั้น ได้หันมาพยายามสร้างและพัฒนาคอมโพเนนต์ ต่างๆสำหรับการเขียนโปรแกรมสำหรับใช้บนอินเทอร์เน็ต ซึ่งในอนาคต อาจจะมีระบบที่เสถียรและมีประสิทธิภาพกว่านี้มารองรับการทำงานของเอเยนต์ก็ได้ หรืออาจจะมีระบบปฏิบัติการสำหรับภาษาจาวานี้เลยก็ได้ ซึ่งขณะที่ทำวิทยานิพนธ์อยู่นี้ก็ได้มีบริษัทผู้ผลิตเครื่องบางแห่ง สร้างเครื่องสำหรับการทำงานของภาษาจาวาขึ้นมาแล้วแต่ยังเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างอยู่ นั่นคือ เครื่องจาวาสเตชั่น (Java Station) การทำงานของเครื่องนี้อาจนำมาพัฒนาใช้กับเอเยนต์ได้ในอนาคต

ด้านการขยายผลในส่วนของโปรแกรม เราได้ออกแบบให้ระบบ เอเยนต์ นี้ สามารถขยายขอบเขตการทำงานไปได้กว้างขวาง คือ ระบบสามารถรองรับการทำงานของ เอเยนต์ หลายๆตัวได้ ซึ่งแต่ละตัวก็สามารถทำงานได้แตกต่างกัน นั่นคือไม่จำเป็นต้องเป็น เอเยนต์ ที่ทำงานอย่างเดียวกันเสมอไปในการทำงานบน AgentLauncher และ AgentServer ซึ่งการกำหนดประเภทและการทำงานก็เพียงกำหนด Method และ Argument เพิ่มขยายจาก Class Agent เท่านั้น

จากระบบนี้จะเห็นว่ามีการทำงานแบบมัลติทาสกิ้งคือสามารถทำงานได้หลายๆอย่างในเวลาเดียวกัน ซึ่งผู้ใช้ก็สามารถส่งเอเยนต์ออกทำงานหน้าที่ต่างได้หลายๆตัวในเวลาเดียวกัน

และในส่วนการขยายผลทางด้านเซอร์ฟเวอร์ก็สามารถขยายเซอร์ฟเวอร์ ที่ต้องการใช้ร่วมในการหาๆไฟล์ คือนอกจากจะหาจากเพียงเซอร์ฟเวอร์เดียวก็สามารถหาจากเซอร์ฟเวอร์อื่นๆได้ด้วย เพียงแค่ เพิ่มชื่อเซอร์ฟเวอร์ลงไปในไฟล์ชื่อ Servers.lst ซึ่งเป็นไฟล์ที่เก็บรายชื่อเซอร์ฟเวอร์ ที่ให้เอเยนต์เดินทางไป และต้องกำหนดพอร์ตให้ตรงกันระหว่างเอเยนต์กับเซอร์ฟเวอร์ เหล่านั้นด้วย

และแน่นอนด้วยว่าการที่ให้ระบบมีมากกว่า 1 เซอร์ฟเวอร์และให้ เอเยนต์ เดินทางไปตามเซอร์ฟเวอร์ต่างๆ เหล่านั้นก็จะได้ผลลัพธ์ที่ถูกต้องเช่นเดิม

5.3 บทสรุปและวิเคราะห์ของแนวโน้มของเทคโนโลยีเอเยนต์

5.3.1 บทสรุปโครงสร้างที่เป็นเทคโนโลยีเอเยนต์ที่มีประสิทธิภาพและสามารถนำมาประยุกต์ใช้ได้จริง

หลังจากที่ได้ดำเนินการศึกษาค้นคว้า ข้อมูลและรายละเอียดต่าง ๆ ที่เกี่ยวข้องกับเทคโนโลยีเอเยนต์ รวมทั้งการทดลองสร้างระบบการทำงานของเอเยนต์ขึ้นมา เราพบว่าเอเยนต์แต่ละชนิดที่สร้างขึ้นมา ต่างก็ถูกออกแบบมาเพื่อรองรับงานที่จะนำเอเยนต์ไปใช้ประโยชน์ ไม่ว่าจะเป็น เอเยนต์ผู้ช่วยส่วนตัว เอเยนต์ค้นหาข้อมูล เอเยนต์เลือกซื้อสินค้า เอเยนต์จองตั๋วเครื่องบิน ฯลฯ เอเยนต์ต่าง ๆ เหล่านี้มีจุดประสงค์ในการทำงานที่แตกต่างกันไป แต่มีอยู่สามประกอบที่ควรเป็นองค์ประกอบพื้นฐานของระบบเอเยนต์ คือ

- **สถานที่ (Place)**

ได้แก่ สถานที่ ที่สำหรับให้เอเยนต์ทำงาน ,เป็นที่อยู่ของเอเยนต์ หรือเป็นที่พบปะเจรจากันระหว่างเอเยนต์ หากเปรียบเทียบ อินเตอร์เน็ตเป็นโลกของเอเยนต์ สถานที่ในที่นี้มักจะเป็น เครื่องเซอร์ฟเวอร์ต่าง ๆ ที่อยู่บนอินเตอร์เน็ตนั่นเอง

เมื่อเอเยนต์เดินทางมาถึงสถานที่แต่ละแห่ง เอเยนต์จะต้องสามารถใช้ทรัพยากรและบริการต่างๆ ที่มีอยู่บนเครื่องเหล่านั้นได้ ซึ่งก็แล้วแต่ว่า เอเยนต์แต่ละตัวจะมี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สิทธิมากน้อยเพียงใดที่จะใช้ทรัพยากรของเครื่องได้ ทั้งนี้ก็เพื่อป้องกันความเสียหายที่อาจเกิดขึ้นแก่ระบบได้

- **เอเยนต์ (Agent)**

ได้แก่ โปรแกรมซอฟต์แวร์ที่มีคุณลักษณะเป็นซอฟต์แวร์เอเยนต์ไม่ว่าจะเป็นความสามารถในการเคลื่อนที่ จากเครื่องหนึ่งไปยังอีกเครื่องหนึ่งเพื่อทำหน้าที่ตามที่ได้รับมอบหมายให้สำเร็จ เอเยนต์ต้องรู้หน้าที่ของมันว่า ต้องทำอะไรบ้าง ซึ่งการเคลื่อนย้ายการทำงานจะเป็นการเคลื่อนย้ายทั้งโค้ด ข้อมูล และสถานะการทำงานของเอเยนต์ไปด้วยพร้อม ๆ กัน แน่นอนว่าเมื่อเอเยนต์เดินทางไปทำงานยังเครื่องคอมพิวเตอร์เครื่องอื่น ๆ ย่อมต้องไม่เกิดข้อผิดพลาดที่เป็นเหตุทำให้การทำงานของเอเยนต์ต้องหยุดชะงัก หรือไม่ก็ต้องหาวิธีป้องกันไว้ก่อนที่จะเกิดเหตุการณ์นี้ขึ้น

- **การเดินทาง (Travel)**

ได้แก่ การเคลื่อนย้ายการทำงานของเอเยนต์จากที่หนึ่งไปสู่อีกที่หนึ่ง ซึ่งในแต่ละครั้งที่เอเยนต์เดินทาง ตัวเอเยนต์เองต้องรู้ก่อนแล้วว่า จุดหมายปลายทาง ที่จะต้องเดินทางไปนั้นคือที่ไหน แล้วจะต้องไปทำอะไร ดังนั้นการเดินทางของเอเยนต์ ก็ควรจะเป็นการเดินทางที่มีที่หมาย แต่หากเมื่อใดที่เอเยนต์เดินทางไปถึงที่หมายแล้วไม่สามารถทำหน้าที่ที่ได้รับมอบหมายให้สำเร็จได้ ก็เป็นความสามารถของเอเยนต์ที่ต้องแก้ปัญหาได้ เป็นต้นว่า เดินทางไปยังเครื่องอื่น ที่คาดว่า จะทำให้การทำงานสำเร็จลงได้

จากลักษณะดังกล่าว สรุปได้ว่า องค์ประกอบที่ควรคำนึงในการทำงานของเอเยนต์ ก็ควรจะเป็นการที่เอเยนต์สามารถเรียนรู้วิธีการที่จะทำงานที่ได้รับมอบหมายให้สำเร็จ ดังนั้นเอเยนต์จึงต้องรู้ว่าตัวมันเองจะต้อง ไปทำอะไร ที่ไหน เมื่อใด ได้อย่างอัตโนมัติ

5.3.2 บทสรุปลักษณะโปรแกรมที่เรียกว่าเอเยนต์

โปรแกรมเอเยนต์ยังไม่เป็นที่นิยมและรู้จักกันอย่างแพร่หลายในปัจจุบัน ผู้คนส่วนใหญ่ยังไม่เข้าใจว่าโปรแกรมเอเยนต์เป็นอย่างไรและแตกต่างจากโปรแกรมธรรมดาทั่วไปอย่างไร ซึ่งเราได้บทสรุปลักษณะสำคัญของโปรแกรมเอเยนต์ดังนี้

- มีความเป็นอัตโนมัติ คือผู้ใช้เพียงแค่บอกเอเยนต์ว่าให้เอเยนต์ทำอะไร เอเยนต์ก็สามารถไปทำงานนั้นได้สำเร็จ โดยผู้ใช้ไม่จำเป็นต้องบอกเอเยนต์ว่าต้องทำอะไรอย่างไรบ้างจึงจะทำงานนั้นได้ ดังตัวอย่างโปรแกรมเอเยนต์ FileFinder ที่เราพัฒนาขึ้นมา ผู้ใช้เพียงแค่บอกชื่อไฟล์เท่านั้น เอเยนต์ก็จะทำงานเป็นขั้นตอนต่างๆได้เอง
- มีการเดินทางระหว่างเครื่องคอมพิวเตอร์บนระบบเน็ตเวิร์ก ซึ่งการเดินทางของเอเยนต์ที่เป็นรูปแบบหนึ่งของการพัฒนาประสิทธิภาพของระบบไคล์เอ็นต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

/เซอร์ฟเวอร์ ซึ่งช่วยลดปัญหาการเรียกใช้งานได้หลายทาง ทั้งเรื่องความปลอดภัย, ความหนาแน่นของข้อมูลบนระบบเน็ตเวิร์ก และเรื่องความมั่นคงของโปรแกรมในการทำงานข้ามเครื่อง

- มีการติดต่อกันระหว่างเอเยนต์กับเอเยนต์ หรือ เอเยนต์กับสถานที่ เพราะการทำงานของเอเยนต์นั้นจะต้องมีการส่งผ่านพารามิเตอร์หรือข้อมูลต่างๆกัน การที่เอเยนต์ส่งพารามิเตอร์กันเราเรียกว่าเป็น การติดต่อระหว่างเอเยนต์ ซึ่งสิ่งที่เอเยนต์ส่งออกมาหรือรับเข้าไปจะเป็นคำสั่งและข้อมูลสำหรับการทำงาน เพื่อให้การทำงานของเอเยนต์ทำได้ดีขึ้น

ยังมีลักษณะของโปรแกรมเอเยนต์ที่เป็นรายละเอียดอีกมากที่เป็นส่วนเพิ่มเติมขึ้นจากนี้ ซึ่งลักษณะต่างๆที่มีผู้ศึกษาเรื่องของเอเยนต์ได้ทำการสรุปและกำหนดไว้ต่างๆนั้นสามารถค้นคว้าได้จาก Web Site อ้างอิงส่วนท้ายของวิทยานิพนธ์ หัวข้อ Agent Definition and Agent-Oriented Programming

5.4 ข้อจำกัดของเทคโนโลยีเอเยนต์

อย่างไรก็ตาม เทคโนโลยีเอเยนต์ยังมีข้อจำกัดในการพัฒนาเทคโนโลยีเพื่อการประยุกต์ใช้งานจริงๆ ซึ่งเราสามารถจำแนกข้อจำกัดออกเป็นด้านต่างๆดังนี้

5.4.1 ความหลากหลายในเรื่องของแพลตฟอร์ม

ในปัจจุบันการนำเทคโนโลยีเอเยนต์มาใช้งานยังไม่เป็นที่แพร่หลายมากนัก เพราะว่า การรองรับสถาปัตยกรรมเอเยนต์ยังไม่ดีพอ เป็นต้นว่า การทำงานของระบบเอเยนต์นั้นจริง ๆ แล้ว จะเป็นอิสระไม่ขึ้นกับระบบปฏิบัติการหรือแพลตฟอร์มใดแพลตฟอร์มหนึ่ง

โดยเฉพาะ เนื่องจากในแต่ละครั้งที่เอเยนต์เคลื่อนที่ไปทำงานยังแพลตฟอร์มที่แตกต่างกัน ก็ควรจะสามารถทำงานได้การใช้งานทรัพยากรต่าง ๆ ของระบบก็ควรเป็นไปตามลักษณะที่สนับสนุนการทำงานของเอเยนต์ได้ตามสมควร แต่ก็ใช่ว่า เอเยนต์จะมีอำนาจในการใช้งานทรัพยากรต่าง ๆ ได้อย่างเต็มที่ เพียงแต่ว่าน่าจะพอเพียงแก่การทำงานของเอเยนต์ให้สำเร็จลงได้ ด้วยความเสี่ยงต่ออันตรายที่อาจเกิดขึ้นแก่ระบบน้อยที่สุด

5.4.2 ข้อจำกัดทางด้านภาษาที่ใช้พัฒนาการเขียนโปรแกรมเอเยนต์

เพราะว่ายังไม่มีการกำหนดรูปแบบภาษาที่เป็นมาตรฐานเพื่อรองรับการทำงานของเอเยนต์อย่างแท้จริง และสามารถใช้งานในเครือข่ายเน็ตเวิร์กให้เป็นที่ยอมรับกันอย่างกว้างขวางเพื่อความสะดวกในการสื่อสารแลกเปลี่ยนข้อมูลกันระหว่างเอเยนต์ได้เป็นอย่างดี เข้าในทำนองที่ว่า เอเยนต์สามารถคุยกันได้รู้เรื่อง เพราะหากว่า เทคโนโลยีเอเยนต์ได้รับการพัฒนาให้ดีขึ้นเพื่อนำมาใช้ประโยชน์อย่างจริงจังนั้น เป็นเรื่องที่น่าคิดอย่างว่า ผู้ใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอเยนต์ควรจะมีความสามารถเพียงพอที่จะสร้างเอเยนต์ของตนเองขึ้นมาโดยมีขั้นตอนที่ไม่ยุ่งยากมากนัก แน่นอนว่า เอเยนต์ที่ถูกสร้างขึ้นมาย่อมมีลักษณะที่แตกต่างกันไปตามความต้องการของผู้ใช้ ดังนั้นการใช้งานเอเยนต์จึงไม่ควรจำกัดอยู่ที่ว่า เอเยนต์ที่จัดอยู่ในจำพวกเดียวกันหรือเป็นชนิดเดียวกันเท่านั้น ที่จะสามารถสื่อสารกันรู้เรื่อง

เป็นไปได้ว่า หากมีการสื่อสารกันระหว่างเอเยนต์ที่ทำหน้าที่แตกต่างกันหรือมาจากผู้ใช้งานคนอื่น ๆ สามารถเกิดขึ้นได้ในสถาปัตยกรรมเอเยนต์จริง ๆ ย่อมเกิดผลดีเป็นอย่างมาก เพราะสามารถขยายขีดความสามารถในการพัฒนาระบบเอเยนต์ไปใช้งานได้อย่างเป็นหนึ่งเดียว

5.4.3 ข้อจำกัดเกี่ยวกับความปลอดภัย

ควรจัดให้มีในระบบการทำงานของเอเยนต์ให้มากขึ้น เพราะว่าหากต้องการใช้งานเอเยนต์ได้อย่างแท้จริงแล้ว ความปลอดภัยในการทำงานของเอเยนต์กับข้อมูลที่เป็นความลับควรอยู่ในระดับที่น่าเชื่อถือได้ และเอเยนต์ควรจะต้องมีความเป็นส่วนตัวให้มากขึ้น ซึ่งในเรื่องความปลอดภัยนี้ หากจะมุ่งเน้นการใช้งานเอเยนต์บนอินเตอร์เน็ต ค่อนข้างที่จะเป็นเรื่องยากที่จะทำการควบคุมในเรื่องของระบบรักษาความปลอดภัยให้รัดกุม ยังคงต้องอาศัยข้อกำหนดที่คาดว่าน่าจะมีสำหรับควบคุมไว้บ้าง เช่น การขึ้นทะเบียนเอเยนต์ที่จะนำมาใช้งานในระบบ การขึ้นทะเบียนเซอร์ฟเวอร์ที่สนับสนุนการทำงานของระบบเอเยนต์ หรือแม้แต่การตั้งเว็บเซอร์ฟเวอร์ที่ทำหน้าที่ดูแลตรวจสอบการใช้งานเอเยนต์ ซึ่งบางครั้งการควบคุมในลักษณะเช่นนี้อาจเป็นการจำกัดขอบเขตการขยายตัวของเทคโนโลยีเอเยนต์ไปบ้างก็ตาม

จะเห็นว่าข้อจำกัดการพัฒนาเทคโนโลยีส่วนใหญ่จะเป็นเรื่องมาตรฐานของระบบเอเยนต์ โดยเฉพาะมาตรฐานในเรื่องทั้ง 3 ที่เรากล่าวไปแล้ว ถ้าเรามองการพัฒนาระบบเอเยนต์ให้เป็นไปในรูปแบบที่จะนำมาใช้งานได้จริงในอนาคต เราก็ควรที่จะมีการกำหนดรูปแบบของเอเยนต์เป็นมาตรฐาน และต้องเป็นมาตรฐานเดียวกันทั้งหมด

5.5 ข้อจำกัดที่ค้นพบเกี่ยวกับทฤษฎีเทคโนโลยีเอเยนต์หลังจากได้ทดลองพัฒนาโปรแกรม FileFinder เอเยนต์

จากประสบการณ์การทดลองการออกแบบและพัฒนาโปรแกรมระบบเอเยนต์ FileFinder เราได้พบข้อจำกัดในการพัฒนาโปรแกรมเอเยนต์ว่าไม่สามารถเป็นไปตามทฤษฎีได้เนื่องจาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.5.1 การที่เอเยนต์ไม่สามารถทำงานข้ามแพล็ตฟอร์มได้จริงทุกกรณีไป

การที่จะให้เอเยนต์ทำงานเป็นระบบเชื่อมโยงและติดต่อกันได้นั้น นอกจากจะต้องกำหนดรูปแบบข้อความและการใช้พารามิเตอร์ในการติดต่อส่งคำสั่งและข้อมูลแล้ว ระบบเอเยนต์ก็ควรที่จะทำงานอยู่บนแพล็ตฟอร์มเดียวกันอีกด้วย เนื่องจากการกำหนด Path การทำงานและคำสั่งต่างๆ ข้ามแพล็ตฟอร์มอาจจะยังไม่สามารถทำได้

นอกจากนี้ การรองรับการทำงานต่างๆแต่ละแพล็ตฟอร์มก็ไม่เหมือนกัน เช่นในระบบยูนิกซ์ ซึ่งมีระบบไฟล์เป็นแบบชื่อไฟล์ยาวได้ แต่เมื่อมาทำงานบนระบบดอสซึ่งไม่รองรับระบบชื่อไฟล์ยาวได้ก็อาจทำให้การทำงานข้ามระหว่าง 2 แพล็ตฟอร์มนี้มีปัญหาได้ นี่เป็นปัญหาที่เราพบ

5.5.2 การใช้คอมไพเลอร์จากต่างบริษัทกันก็อาจจะทำให้ได้ผลลัพธ์ต่างกัน

แม้กระทั่งการเขียนโปรแกรมภาษาเดียวกันแต่ใช้คอมไพเลอร์ต่างบริษัทกันก็อาจจะให้ผลลัพธ์ต่างกัน เนื่องจากระบบแวดล้อมและการสนับสนุนการทำงานของแต่ละคอมไพเลอร์นั้นไม่เหมือนกัน โปรแกรมเดียวกันแต่รันบนคอมไพเลอร์ต่างกันอาจได้ผลลัพธ์ไม่เหมือนกัน

ซึ่งจากโปรแกรม FileFinder นี้เราก็ประสบปัญหานี้ด้วยเช่นกัน คือเราใช้คอมไพเลอร์ชื่อ JDK 1.02 ของบริษัทซันไมโครซิสเต็มมาให้เอเยนต์ทำงานก็จะได้ผลลัพธ์ที่ถูกต้อง แต่เมื่อเราลองนำโปรแกรมเดียวกันไปทดลองทำงานใน Symantec Cafe กลับไม่สามารถทำงานได้เลย ซึ่งเหตุผลของข้อจำกัดของ Symantec Cafe นั้นเรายังไม่ทราบสาเหตุ แต่สันนิษฐานว่าอาจเป็นเพราะการไม่รองรับการทำงานการส่งชื่อ Path ที่เรากำหนดก็ได้

5.6 ลักษณะการนำเอเยนต์ไปใช้งาน

- ธุรกิจการค้าบนอินเตอร์เน็ต

ทุกวันนี้เรามักจะพบกับ ธุรกิจการค้าขาย ที่มีการสั่งซื้อสินค้าผ่านทางอินเตอร์เน็ตกันอย่างแพร่หลายหรือเรียกอีกอย่างว่า ธุรกิจการค้าแบบออนไลน์ ซึ่งนับได้ว่าสร้างความสะดวกให้แก่ผู้บริโภคเป็นอย่างมาก เพราะไม่ต้องเสียเวลาเดินทางไปยังร้านค้าเพื่อซื้อสินค้าที่ต้องการ เพียงแค่ผู้ซื้อเข้าไปยังเว็บไซต์ที่จำหน่ายสินค้าที่ตรงตามความต้องการ แล้วทำการกรอกข้อมูลรายละเอียดต่าง ๆ ลงในแบบฟอร์มสั่งซื้อสินค้าของบริษัท หลังจากนั้นทางบริษัทผู้ขายสินค้าก็จะนำสินค้ามาส่งถึงบ้าน

หากพิจารณาให้ดีจะพบว่า ในการสั่งซื้อสินค้าแบบออนไลน์ ผู้บริโภคยังได้รับประโยชน์จากธุรกิจประเภทนี้ไม่เต็มที่เท่าไหร่นัก เพราะปัญหาที่อาจเกิดขึ้นในการใช้งานยังไม่ได้รับการแก้ไขและป้องกันให้มีความไว้วางใจและความน่าเชื่อถือเท่าใดนัก เช่น

- ♦ การต่อรองราคาสินค้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- คุณภาพของสินค้า
- ความน่าเชื่อถือของบริษัทผู้ขาย

จากตัวอย่างดังกล่าวข้างต้น เรายังไม่อาจให้ความเชื่อถือกับการสั่งซื้อสินค้าบนอินเตอร์เน็ตได้อย่างมั่นใจ เพราะสิ่งที่กล่าวมานั้น ผู้ซื้อไม่สามารถทำการพิสูจน์ได้ด้วยตัวเองหรือไม่สามารถกระทำได้ ด้วยเหตุนี้ การนำเทคโนโลยีเอเยนต์มาใช้งาน ก็จะช่วยให้ผู้ซื้อสินค้าได้ประโยชน์จากการสั่งซื้อสินค้าบนอินเตอร์เน็ตได้อย่างมั่นใจเพิ่มขึ้น ไม่ว่าจะเป็น

1. *การให้เอเยนต์ช่วยต่อรองราคาสินค้า* ให้อยู่ในระดับที่น่าพอใจได้ก่อน แล้วจึงค่อยให้ เอเยนต์ตัดสินใจสั่งสินค้านั้น
2. *การเลือกลักษณะสินค้า* ที่ผู้ใช้สามารถป้อนข้อมูลที่เกี่ยวกับตัวสินค้าที่ต้องการแล้วมอบหมายหน้าที่ในการค้นหาสินค้าให้ตรงตามที่ต้องการให้มากที่สุดซึ่งเป็นความสามารถของเอเยนต์ที่ต้องเดินทางไปสำรวจยังไซต์ต่าง ๆ ไม่จำกัดอยู่เพียงที่ใดที่หนึ่งเท่านั้น
3. *ความน่าเชื่อถือของบริษัทผู้ขาย* หากไม่ติดกับหลักการของเครือข่ายอินเตอร์เน็ต ที่ว่าเป็นระบบข้อมูลแบบเปิด ซึ่งลักษณะเช่นนี้ทำให้ที่มาของข้อมูลบางครั้งเป็นการยากที่จะตรวจสอบหรือเชื่อถือได้ หากเรานำเอเยนต์มาใช้ แล้วให้เป็นหน้าที่ของเอเยนต์ที่จะทำการตรวจสอบความน่าเชื่อถือของบริษัทผู้ขาย เป็นต้นว่า อาจจะต้องกำหนดให้มีการขึ้นทะเบียนกับเว็บไซต์ที่ต้องการดำเนินธุรกิจค้าขายสินค้าบนอินเตอร์เน็ตไว้ด้วย แล้วให้เป็นหน้าที่ของเอเยนต์ในการตรวจสอบว่า เว็บไซต์นี้ได้ขึ้นทะเบียนไว้หรือไม่มีความน่าเชื่อถือมากน้อยเพียงใด เป็นต้น

5.7 ความก้าวหน้าในอนาคต

ระบบเอเยนต์ในอินทราเน็ต (Intranet Agent)

ข้อจำกัดที่มีแก่ระบบการทำงานของเอเยนต์ในอินเตอร์เน็ตนั้น ค่อนข้างจะเป็นเรื่องที่น่ากังวลอยู่พอสมควร หากไม่มีมาตรการใดที่จะให้ความมั่นใจในเรื่องของระบบความปลอดภัยให้อยู่ในระดับที่น่าเชื่อถือได้ คงเป็นไปได้ยากที่จะพัฒนาระบบเอเยนต์ในอินเตอร์เน็ตให้ใช้งานได้อย่างดี คงถูกจำกัดความสามารถอยู่ที่ระดับหนึ่ง ที่ไม่ต้องกังวลเกี่ยวกับความปลอดภัยของข้อมูลหรือการทำงานของเอเยนต์มากนัก แต่ถ้าเป็นข้อมูลที่ต้องเกี่ยวข้องกับเรื่องของความปลอดภัยของการดำเนินการ เช่น การชำระราคาสินค้าหรือบริการผ่านอินเตอร์เน็ตโดยใช้บัตรเครดิต ซึ่งเอเยนต์ต้องทำงานกับข้อมูลที่เป็น หมายเลขบัตรเครดิต ซึ่งควรจะมีการจัดระบบรักษาความปลอดภัยที่น่าไว้ใจได้ว่าจะไม่เกิดความเสียหายแก่ผู้ใช้งานในภายหลังได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แนวคิดหนึ่งที่คาดว่า น่าจะได้รับการพัฒนาให้ใช้งานควบคู่กับเทคโนโลยีเอเยนต์ได้เป็นอย่างดี คือ การพัฒนาระบบเอเยนต์ในอินทราเน็ต (Intranet Agent) ทั้งนี้เพราะว่า ในอินทราเน็ตเราสามารถควบคุมเรื่องของระบบรักษาความปลอดภัยได้ดีกว่า การดูแลรักษาความลับขององค์กรสามารถดำเนินการได้อย่างรัดกุม ถึงแม้ว่าความปลอดภัยในเรื่องความเป็นส่วนตัวของข้อมูลจากผู้ใช้งานจะมีได้ไม่มากนัก แต่ถ้าคำนึงถึงจุดประสงค์การใช้งานของระบบอินทราเน็ต เมื่อพิจารณาดูแล้วก็เห็นว่าไม่ผิดวัตถุประสงค์แต่อย่างไร

5.8 ข้อเสนอแนะเกี่ยวกับการพัฒนาระบบเอเยนต์เพื่อนำมาประยุกต์ใช้ได้จริงในชีวิตประจำวัน

เราได้มีระบบเอเยนต์ในอุดมคติมาหลายปีแล้ว และได้มีนักพัฒนาโปรแกรมทั่วโลกให้ความสนใจเกี่ยวกับแนวคิดใหม่ที่จะใช้เอเยนต์นี้และได้พยายามพัฒนาเป็นระบบขึ้นมาด้วยกันหลายระบบ แต่ระบบเหล่านั้นก็ยังเป็นเพียงระบบเพื่อทดสอบรูปแบบและการทำงานและเป็นเพียงตัวอย่างการทำงานแบบเอเยนต์เพื่อให้ทฤษฎีเอเยนต์ได้มีการมองเห็นภาพรวมชัดเจนขึ้น

แต่จะมีประโยชน์อะไรถ้าระบบเอเยนต์ที่ถูกพัฒนาขึ้นมาแล้วถูกเก็บไว้เฉยๆ ส่วนคนที่ต้องการศึกษาและค้นคว้าก็ต้องลองผิดลองถูกกันอยู่เรื่อยไป เราอยากจะเสนอแนะแนวทางการพัฒนาเทคโนโลยีเอเยนต์ให้ได้ผลดีและสามารถนำมาประยุกต์ใช้ได้จริง ดังนี้

5.8.1 ควรจะมีการรวมกลุ่มผู้ที่ทำการศึกษาและนำข้อสรุปของการทดลองเหล่านั้น มากำหนดเป็นมาตฐานการพัฒนาโปรแกรมเอเยนต์ต่อไป ทั้งในเรื่องของการเชื่อมการทำงานกันระหว่างเอเยนต์บนแพล็ตฟอร์มต่างๆ การกำหนดรูปแบบการเขียนโปรแกรมและพัฒนาโปรแกรมของภาษาต่างๆ เป็นต้น ทั้งนี้เพื่อการทำงานข้ามแพล็ตฟอร์มจึงจะสามารถพัฒนาได้จริงอย่างที่ทฤษฎีว่าไว้

5.8.2 บริษัทที่เป็นผู้พัฒนาแพล็ตฟอร์มและคอมไพเลอร์ต่างๆ ควรมีการตกลงกันถึงมาตรฐานที่เป็นกลางเพื่อการนำมาพัฒนาเป็นระบบเอเยนต์ต่อไป

5.8.3 ควรมีการกำหนดรูปแบบการรักษาความปลอดภัยที่ระบบเอเยนต์ควรมีซึ่งต้องเป็นมาตรฐานเดียวกัน

5.8.4 ร้านค้าประเภทออนไลน์ต่างๆควรให้ความร่วมมือสำหรับการทำงานของเอเยนต์เพื่อระบบเอเยนต์จะสามารถนำมาใช้ได้จริงในอนาคน เพื่ออำนวยความสะดวกให้แก่ผู้ซื้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.9 ข้อเสนอแนะสำหรับผู้สนใจที่จะพัฒนาระบบเอเยนต์

5.9.1 ออกแบบโครงสร้างให้มีส่วนประกอบอย่างชัดเจนและกำหนดรูปแบบการทำงานของแต่ละส่วนให้มีการทำงานสัมพันธ์กันแต่ไม่ซ้ำซ้อนกัน โดยอาศัยศึกษาจากเทคโนโลยีเอเยนต์และความรู้เรื่องเอเยนต์ที่มีอยู่ตาม Web Site ต่างๆเพื่อเป็นแนวทางในการออกแบบต่อไป

5.9.2 เลือกภาษาที่มีผู้พัฒนาหลายรายและเป็นที่แพร่หลายและมีการรองรับจากแพล็ตฟอร์มทั่วไป ทั้งนี้การภาษาที่เป็นที่นิยมและแพร่หลาย จะมีผู้ให้ความช่วยเหลือในกรณีที่เกิดปัญหาเกี่ยวกันตัวภาษาเองได้ และควรจะมีการศึกษาภาษานั้นอย่างท่องแท้ก่อนทำการพัฒนาโปรแกรม

5.9.3 กำหนดรูปแบบการสื่อสารระหว่างไคล์เอ็นต์และเซอร์ฟเวอร์ที่แน่นอน และควรจะมีความรู้เรื่องการใช้งานระบบไคล์เอ็นต์/เซอร์ฟเวอร์มาบ้างแล้ว

5.9.4 บางครั้งการเขียนโปรแกรมแบบที่เป็นโปรแกรมบนระบบเน็ตเวิร์ก อาจมีความไม่แน่นอนเกิดขึ้น เช่นความหนาแน่นบนระบบมากเกินไป หรือการที่ข้อมูลหายไประหว่างการส่ง ผู้พัฒนาต้องมีมาตรการการตรวจสอบที่ดีพอที่จะรองรับปัญหาที่อาจจะเกิดขึ้นเหล่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

คณะผู้จัดทำขอแสดงความนับถือและขอขอบพระคุณผู้มีพระคุณทุกท่านที่ทำให้วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี โดยได้รับความช่วยเหลือและคำแนะนำจากบุคคลต่อไปนี้

อาจารย์อภิเนตร อูนากูล อาจารย์ที่ปรึกษา

คุณพ่อคุณแม่ เพื่อนๆ พี่ๆ ที่ให้กำลังใจ ตลอดมา

และที่สำคัญที่สุดคือ พี่กุ๊กเก่ง เจ้าชายขี่ม้าขาว ..... รักป๊าเก่งที่สุดในโลกเลยค่ะ

คณะผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

[1]. Fah-chun Cheong : " *Internet Agent : Spider, Wondererd, Brokers and Bot.* " , New Rider.

[2]. John Rodley : " *Internet Programing Series WRITING JAVA APPLETS* ", Coriolis Group Books.

[3]. JOSEPH WILLIAMS : " *BOTS AND OTHER INTERNET BEASTIES* ", Sams.net Publishing.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เว็บไซต์อ้างอิง

Agent Definition and Agent-oriented Programming

[1]. http://www.ee.mcgill.ca/~belmarc/

[2]. http://www.msci.memphis.edu/~franklin/AgentProg.html

[3]. http://www-csli.stanford.edu/csl...95reps/interface9495-shoham.html

[4]. http://cdr.stanford.edu/NextLink/Expert.html

[5]. http://foner.www.media.mit.edu/

[6]. http://cuiwww.uunige.ch/OSG/Vitek/Agent/index.html

Agent Programming Language

[1]. http://logic.stanford.edu/kif/definitions.html

[2]. http://www.w3.org/pub/WWW/MobileCode/

[3]. http://www.genmagic.com/Telescript/

[4]. http://www.uni-kl.de/AG-Nehmer/Ara/

[5]. http://www.ibm.co.jp/tri/projects/aglets/

[6]. http://www.cs.dartmouth.edu/~agent/

Researches on Agents

[1]. http://agents.www.media.mit.edu:80/groups/agnets/research.html

[2]. http://lieber.www.media.mit.edu/...taching/Attaching/Attaching.html

[3]. http://foner.www.media.mit.edu/people/foner/agnets.html

[4]. http://agent.www.media.mit.edu/...agent/papers/aaai-ymp/aaai.html

Commercial Agents

[1]. http://ai.eecs.umich.edu/people/wellman/WalrACDS.html

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

[3]. http://www.cs.nccu.edu.tw/~jong/agent/agent.html

Agent News Webletters and Related Pages

[1]. http://www.cl.cam.ac.uk/users/rwabl/ag-pagess.html

[2]. http://cs.wpi.edu/Research/airg/Agent-hotlist.html

[3]. http://www.cs.umbc.edu/agents/agentnews/

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

BargainFinder : Available Sample Agents Searching For CDs

[1]. http://bf.cstar.ac.com/bf/

WebCrawler Search

[1]. http://www.webcrawler.com

Lycos Search

[1]. http://www.lycos.com

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้