การทำอินเด็กซ์อัตโนมัติสำหรับห้องสมุดเสมือน

AUTOMATIC INDEXING FOR VIRTUAL LIBRARY

นางสาว นภาพร จิรุภัณฑ์สวัสดิ์

นางสาว นฤมล วงศ์อริยะกวี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

การทำอินเด็กซ์อัตโนมัติสำหรับห้องสมุดเสมือน

AUTOMATIC INDEXING FOR VIRTUAL LIBRARY

โดย

นางสาว นภาพร จิรุภัณฑ์สวัสดิ์

นางสาว นฤมล วงศ์อริยะกวี

อาจารย์ที่ปรึกษา

อาจารย์ อภิเนตร อูนากูล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การทำอินเด็กซ์อัตโนมัติสำหรับห้องสมุดเสมือน

Automatic Indexing for Virtual Library

ผู้จัดทำ

1. นางสาว นภาพร จิรุภัณฑ์สวัสดิ์ รหัสประจำตัว 40010363
2. นางสาว นฤมล วงศ์อริยะกวี รหัสประจำตัว 40010373

อาจารย์ที่ปรึกษา

(อาจารย์ อภิเนตร อูนากูล)

การทำอินเด็กซ์อัตโนมัติสำหรับห้องสมุดเสมือน

น.ส. นภาพร จิรุภัณฑ์สวัสดิ์ 40010363
น.ส. นฤมล วงศ์อริยะกวี 40010373
อ. อภิเนตร อูนากูล อาจารย์ที่ปรึกษา
ปีการศึกษา 2543

บทคัดย่อ

ในปัจจุบันความก้าวหน้าทางด้านเทคโนโลยีทางด้านอินเตอร์เน็ตมีเพิ่มขึ้นมาก การจัดเก็บข้อมูลหรือเอกสารต่างๆก็เริ่มหันมาทำการจัดเก็บในรูปแบบอิเล็กทรอนิกส์เพราะมีข้อดีในการจัดเก็บที่ง่าย และสะดวกในการรักษาข้อมูลด้วย ดังนั้นจึงต้องมีวิธีการในจัดเก็บข้อมูลที่เหมาะสม รวมถึงต้องสามารถค้นคืนเอกสารได้อย่างสะดวกและรวดเร็วด้วย

ด้วยเหตุผลนี้จึงได้มีการจัดทำอินเด็กซ์อัตโนมัติ ซึ่งมีความจำเป็นในการที่จะช่วยมนุษย์ในการจัดเก็บเอกสาร โดยจะสามารถหาคำที่เป็นอินเด็กซ์ที่จะเป็นตัวแทนของเอกสารนั้นได้ เพื่อที่จะสามารถสื่อได้ว่าเอกสารนั้นๆเกี่ยวข้องกับอะไร และเมื่อผู้ใช้ทำการค้นคืนเอกสารก็จะสามารถค้นหาข้อมูลโดยทำการเรียกข้อมูลที่ตรงกับอินเด็กซ์ของเอกสารนั้นๆออกมา

ในด้านของการจัดแบ่งประเภทของเอกสารก็เช่นเดียวกัน จะมีความสะดวกยิ่งขึ้นในการที่จะให้โปรแกรมทำการแบ่งแยกเอกสารให้ได้ว่า เอกสารนั้นๆควรจะถูกแบ่งประเภทไว้ในประเภทใด เพื่อให้ผู้ใช้สามารถที่จะตรวจดูได้ทันทีว่า ในประเภทของเอกสารนั้นๆ มีเอกสารใดบ้างที่น่าสนใจอยู่

อย่างไรก็ตามต้องยอมรับว่า ถึงแม้จะมีโปรแกรมเหล่านี้มาช่วยในการจัดทำอินเด็กซ์อัตโนมัติและแบ่งประเภทของเอกสาร แต่ความถูกต้องในการจัดทำนั้นไม่สามารถให้มีความถูกต้องสมบูรณ์ครบถ้วน 100 เปอร์เซ็นต์ได้ ถึงอย่างไรการตัดสินใจต่างๆก็ยังคงต้องให้มนุษย์เป็นผู้ตัดสินใจหรือพิจารณาอีกครั้งหนึ่ง เพียงแต่ว่าจะโปรแกรมเหล่านี้จะเป็นส่วนช่วยในการทำงานของมนุษย์เท่านั้น

Automatic Indexing For Virtual Library

Napaporn Jirupunsawat
Naruemon Wongariyakawee
Apinetr Unakul advisor

ABSTRACT

Nowadays, the Internet technologies have been continuously developed as they're a convenient and easy way of communication. There's a large amount of information on the Internet, so it's hard to find out the documents that you are interested in.

Virtual Library is used to help people in searching and retrieving documents in form of electronic resource. To build Virtual Library we have to collect and classify the information. In classification process, at first we have to find out the index terms, the important words, of documents for classifying the categories of them. Because of the large amount of electronic resource it's hard work for human to find the index terms.

So in this thesis we provide "Automatic Indexing for Virtual Library" for helping people in indexing and classification. However, these programs are only used to help people in making decision. We have to accept that the results of automatic indexing and classification are not completely accurate as being done by human.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลาย ๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ อาจารย์ อภิเนตร อูนากูล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

และบุคคลที่สำคัญที่คอยเป็นกำลังใจและให้ความช่วยเหลือเป็นอย่างดีมาตลอด คือ บิดา มารดา ผู้ซึ่งเลี้ยงดูอบรมมาเป็นอย่างดี จนถึงทุกวันนี้ ขอขอบพระคุณเป็นอย่างสูง

ขอขอบคุณพี่ๆทุกคนในห้องฮาร์ดแวร์ที่ให้สถานที่และอาหารในระหว่างการทำโปรเจ็ค

ขอขอบคุณพี่เต๋าและพี่ชิตที่คอยให้คำแนะนำที่ดีมาตลอด

ขอขอบคุณ เพ็ญ โป้ง แบ็ง พงศ์ ที่ให้คำปรึกษาที่ดีมากในการทำอินเตอร์เฟส

สุดท้ายที่จะลืมไม่ได้ ขอขอบคุณนายวิทยา สำหรับความช่วยเหลือในทุก ๆ เรื่องจนทำให้โปรเจ็คชิ้นนี้ สำเร็จลุล่วงไปได้ และหนอนน้อย สำหรับความน่ารัก ใจดี ที่มาคอยนั่งคุยเป็นเพื่อน และคอยรับฟังเรื่องบ่นต่าง ๆ

น.ส. นภาพร จิรุภัณฑ์สวัสดิ์

น.ส. นฤมล วงศ์อริยะกวี

สารบัญ

สารบัญตาราง

หน้าที่

สารบัญภาพ

บทที่ 1 บทนำ

1.1 ความเป็นมาของโครงงาน

หลายปีที่ผ่านมามีการเพิ่มจำนวนของข้อมูลแบบดิจิตอลอย่างมาก รวมถึงการพัฒนาระบบคอมพิวเตอร์, ระบบเน็ตเวิร์ค และการติดต่อสื่อสาร ส่งผลให้การสร้างและการแพร่กระจายของข้อมูลดิจิตอลมีเพิ่มมากขึ้น, มีแหล่งเก็บข้อมูลในรูปแบบใหม่ๆ และมีการส่งข้อมูลในรูปแบบใหม่ๆด้วย ดังนั้นจึงต้องมีโครงสร้างในการเก็บข้อมูลที่ดีและสามารถดึงข้อมูลเหล่านั้นมาใช้งานได้อย่างมีประสิทธิภาพ โดยผู้ใช้ก็จะสามารถเข้ามาค้นหาข้อมูลได้อย่างสะดวกและง่ายดาย จึงได้นำเสนอโครงงานในการทำอินเด็กซ์อัตโนมัติสำหรับห้องสมุดเสมือน (Automatic Indexing for Virtual Library) ขึ้น

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาวิธีการในการค้นหาข้อมูล (Information Retrieval) ที่เหมาะสม และได้ประสิทธิภาพสูงสุด

1.2.2 เพื่อศึกษาโครงสร้างของห้องสมุดเสมือน (Virtual Library)

1.2.3 เพื่อศึกษาโครงสร้างในการจัดเก็บฐานข้อมูลและในการดึงข้อมูลมาใช้งาน

1.2.4นำเสนอและพัฒนาระบบเพื่ออำนวยความสะดวกแก่ผู้ดูแลระบบในการจัดเก็บข้อมูลและแบ่งแยกประเภทของเอกสาร ภายในห้องสมุดเสมือน

1.2.5 สามารถนำระบบไปประยุกต์ใช้ได้

1.3 ขอบเขตของโครงงาน

โครงงานนี้เป็นโครงงานในการจัดทำอินเด็กซ์อัตโนมัติ (Automatic indexing) ซึ่งจะช่วยให้มีความสะดวกในการหาคำซึ่งเป็นตัวแทนของเอกสารนั้น เนื่องจากในปัจจุบันมีเอกสารที่เป็นข้อมูลทางดิจิตอลเป็นจำนวนมาก การจัดทำอินเด็กซ์โดยคนนั้นเป็นเรื่องที่ยากลำบากและเสียเวลาอย่างมาก ดังนั้นทำให้การจัดทำอินเด็กซ์อัตโนมัติเป็นสิ่งที่จำเป็นมากยิ่งขึ้น โดยในโครงงานนี้เราจะจัดทำอินเด็กซ์อัตโนมัติเพื่อช่วยในการทำห้องสมุดเสมือน ให้ระบบสามารถทำงานได้สะดวกและรวดเร็วมากขึ้น เพราะในห้องสมุดเสมือนจะมีการจัดเก็บเอกสารเป็นจำนวนมากมาย

โดยในโครงงานนี้ จะจัดให้สไปเดอร์ (Spider) เป็นตัวไปหาเอกสารตามเว็บไซด์ต่างๆแล้วทำการดึงข้อมูลซึ่งเป็นเอกสารทางวิชาการและมีรูปแบบข้อมูลที่เป็นรูปแบบมาตรฐานมา โดยไฟล์ที่ทำการดึงมานั้นจะเป็นไฟล์ที่มีสกุล ".xml" จากนั้นก็นำเอกสารมาผ่านพาร์เซอร์ (parser) เพื่อดึงข้อมูลจากแท็ก (tag) ต่างๆ แล้วนำมาผ่านกระบวนการจัดทำอินเด็กซ์อัตโนมัติ ซึ่งจะได้อินเด็กซ์เทอมของแต่ละเอกสารออกมา สุดท้ายจะเป็นการจัดแบ่งหมวดหมู่ของเอกสารนั้น แล้วทำการจัดเก็บไว้ในฐานข้อมูลเพื่อให้ผู้ใช้สามารถเข้ามาค้นหาข้อมูลได้

ซึ่งโดยภาพรวมของโครงงานจะแบ่งงานหลักออกเป็น 4 ส่วน คือ

1.3.1 ส่วนของ spider ในการดึงเอกสารต่างๆจากเว็บไซด์ ซึ่งไฟล์ที่ดึงมาจะมีสกุล ".xml"

1.3.2 ส่วนของการจัดทำอินเด็กซ์อัตโนมัติเพื่อให้ได้อินเด็กซ์เทอมออกมา

1.3.3 ส่วนของการจัดแบ่งแยกประเภทของเอกสาร

1.3.4 ส่วนของผู้ใช้บริการจะเป็นเว็บไซด์ที่ให้บริการเหมือนเสิร์ชเอ็นจินแบบไดเรกทอรีแบบหนึ่ง

1.4 วิธีการดำเนินงาน

การดำเนินในโครงงานนี้จะเริ่มด้วยการศึกษาทฤษฎีพื้นฐานต่างๆที่เกี่ยวข้องกับโครงงานทั้งหมด ได้แก่ โครงสร้างในการทำห้องสมุดเสมือน กระบวนการจัดทำอินเด็กซ์อัตโนมัติ การแบ่งแยกประเภทของเอกสาร กระบวนการในการทำเสิร์ชเอ็นจิน

ภายหลังจาการศึกษาทฤษฎีต่างๆแล้ว ก็จะเริ่มทำการติดตั้งซอฟท์แวร์ที่จำเป็นต่างๆ แล้วเริ่มทำการเขียนโปรแกรมทำการทดสอบในแต่ละส่วน ซึ่งแบ่งเป็น 4 ส่วนใหญ่ๆ ดังที่ได้นำเสนอไปแล้วใน หัวข้อขอบเขตของโครงงาน แล้วนำส่วนต่างๆมาประกอบให้เสร็จสมบูรณ์เป็นระบบรวมทั้งหมด

ขั้นตอนสุดท้ายจะเป็นการทดลองและทดสอบระบบเพื่อสรุปผลของการทำงาน โดยจะต้องมีการจัดหาเอกสารต่างๆ เพื่อมาทดสอบระบบให้มากเพียงพอ และในบทสุดท้ายก็จะเป็นการสรุปผลการทดลองและแนวทางการพัฒนาต่อไป

บทที่ 2 ทฤษฎีและหลักการ

2.1 ความเป็นมาของห้องสมุดเสมือน

สมัยก่อนนั้นการที่จะค้นคว้าหาข้อมูลต่างๆนั้นเป็นเรื่องที่ค่อนข้างยากลำบากทำให้ผู้ใช้ไม่ค่อยสนใจหรืออยากที่จะค้นคว้าหาข้อมูลมากนัก ซึ่งจะส่งผลต่อการพัฒนาประเทศชาติ และความรู้ของประชาชนโดยทั่วไปด้วย ซึ่งจะยกปัญหาที่สำคัญๆดังต่อไปนี้

- จะต้องทำการเดินทางไปยังห้องสมุดเพื่อการค้นหาข้อมูลซึ่งก็เป็นการไม่สะดวกในการเดินทางของผู้ใช้บริการด้วย
- รวมถึงถ้าต้องการข้อมูลใดๆก็จะต้องทำการยืมหนังสือเล่มนั้นๆออกจากห้องสมุด และบางเล่มก็อาจจะไม่อนุญาตให้นำออกจากห้องสมุดได้อันอาจจะเนื่องมาจากความเก่าแก่ของหนังสือ หรือความสำคัญของหนังสือเล่มนั้น
- ถ้าสามารถนำหนังสือเล่มนั้นๆออกมาได้ บุคคลอื่นที่ต้องการจะยืมหนังสือเล่มนั้นก็ไม่สามารถทำการยืมหนังสือเล่มนั้นได้ในเวลาเดียวกัน นั่นคือในเวลาหนึ่งๆจะมีผู้เข้ามานำหนังสือหรือข้อมูลนั้นๆไปได้เพียงคนเดียวเท่านั้น หรือถ้าต้องการข้อมูลนั้นๆแล้วไม่มี ก็ต้องทำการเดินทางไปค้นหาข้อมูลนั้นจากห้องสมุดอื่น หรืออาจจะให้ทางห้องสมุดค้นหาจากห้องสมุดอื่นให้แล้วทำการส่งผ่านมาซึ่งมันไม่ใช่เรื่องง่ายเลย
- นอกจากนี้ห้องสมุดต่างๆจะทำการเก็บข้อมูลในรูปแบบของกระดาษซึ่งกระดาษก็มีโอกาสที่จะย่อยสลายไปได้ตามกาลเวลา อาจจะทำให้ผู้ใช้ไม่ได้รับอนุญาตที่จะทำการค้นหาข้อมูลนั้น ซึ่งเป็นการจำกัดความรู้และข้อมูลแก่ผู้ใช้ด้วย ดังนั้นจะเห็นได้ว่ามีข้อจำกัดในการค้นหาข้อมูลของห้องสมุดในยุคแรกๆมากมาย

เมื่อพบปัญหาต่างๆ ในการค้นหาข้อมูลของห้องสมุดนั้น ต่อมาจึงได้มีการพัฒนารูปแบบในการจัดเก็บข้อมูลของเอกสารต่างๆให้อยู่ในรูปแบบดิจิตอล (Digital Documents) หรือสื่ออิเล็กทรอนิกส์ เพื่อให้ง่ายต่อการดูแลและรักษา ไม่ต้องมาดูแลรักษาเหมือนการเก็บหนังสือที่เป็นกระดาษว่ายิ่งเก็บนานกระดาษก็จะยิ่งย่อยสลายไป หรือการที่มีผู้มาหยิบดูมากก็จะทำให้หนังสือเปื่อยได้

2.1.1 วิวัฒนาการของการพัฒนาห้องสมุด

ความต้องการที่จะทำการเก็บและรวมรวมข้อมูลและทำการเข้าถึงข้อมูลได้ง่ายโดยมีการนำเทคโนโลยีต่างๆเข้ามาช่วยซึ่งจะมีการพัฒนาจากหนังสือที่เก็บเอาไว้อย่างเก่าแก่โบราณมาจัดเก็บรวมรวมใหม่ในรูปแบบ digital document เพื่อเผยแพร่สู่สาธารณชนผ่านทางสื่อ internet และ www ซึ่งจะมียุคของการพัฒนา library ทั้งหมด 3 ยุค เริ่มตั้งแต่

1. **Traditional Library**

จะมีข้อมูลที่สำคัญมากๆ อาจจะเป็นข้อมูลเก่าแก่ทางประวัติศาสตร์หรือเป็นข้อมูลที่ต้องเก็บเอาไว้ให้ดีที่สุดตลอดไป ไม่ให้เกิดการสูญหายได้ ซึ่งข้อมูลจะเป็นประเภทของหนังสือ ผู้เขียนหรือผู้ประพันธ์หนังสือ จะเป็นผู้เขียนและตกลงกับบรรณาธิการเมื่อต้องการทำการแก้ไขข้อมูลที่แสดงต่อสาธารณชนแล้ว บรรณารักษ์จะเป็นผู้ตัดสินใจที่จะซื้อข้อมูลนั้นมานำเสนอ

2. **Specialized Library**

จะมีแหล่งข้อมูลใหม่เพิ่มขึ้นมาคือ เอกสา. ซึ่งจะเป็นข้อมูลที่เป็นทางการ เน้นที่จะต้องเก็บรักษาให้คงอยู่ตลอดไป แต่ต้องการจัดเก็บให้คงอยู่เพียงแค่ช่วงเวลาหนึ่งเท่านั้น มันยากสำหรับ Traditional library ที่จะติดตามทิศทางการเพิ่มขึ้นของข้อมูลในทุกๆสาขาที่เพิ่มขึ้นอย่างมากมาย ดังนั้น traditional library จึงผันตัวมาเป็น specialized library ซึ่งคณะการทำงานนี้จะเป็นคนแบ่งแยกประเภทและจัดการบริการไว้ให้กับลูกค้า

3. **Networked Digital Library**

จะมีความเปลี่ยนแปลงมากขึ้น เนื่องจากคอมพิวเตอร์เข้ามามีบทบาทมากยิ่งขึ้น สื่อต่างๆที่จะเข้ามาช่วยในการนำเสนอข้อมูลก็มีมากขึ้น เช่น เข้าถึงข้อมูลผ่านทาง www หรือ electronic mail ซึ่งปัจจุบัน ผู้ใช้บริการมีศักยภาพมากขึ้น คือมักจะมีเครื่องคอมพิวเตอร์ใช้เป็นส่วนตัว จึงเป็นการง่ายกับผู้ใช้ที่จะทำการเข้าถึงข้อมูลผ่านทาง internet และข้อมูลแบบใหม่ ที่เพิ่มเข้ามาคือ idea

ต่อมาก็มีการพัฒนามาเรื่อยๆเพื่อให้เหมาะสมกับเทคโนโลยีและความทันสมัยนั่นคือมีการปรับปรุงรูปแบบของห้องสมุดให้อยู่ในรูปของเว็บไซต์ให้บริการผู้ใช้ผ่านทางอินเตอร์เน็ต ทำให้ผู้ใช้สามารถเข้าถึงห้องสมุดได้ทุกหนทุกแห่งที่มีการออนไลน์ต่อเข้ากับระบบอินเตอร์เน็ต

จนกระทั่งปัจจุบันนี้ก็มีห้องสมุดเสมือนซึ่งให้บริการการค้นหาห้องสมุดและมีฟังก์ชันต่างๆที่สามารถเข้าไปใช้บริการได้อย่างมากมาย เพราะมีการพัฒนาไปไกลตามเทคโนโลยีใหม่ที่เพิ่มเข้ามา

2.1.2 การวิเคราะห์ข้อมูลของห้องสมุด

ในการสร้าง Virtual Library จะต้องมีการแปลงเอกสารในรูปแบบ เอกสาร เป็น ดิจิตอล นอกจากนี้เราจะต้องทำการจัดหมวดหมู่ให้กับเอกสารนั้น ซึ่งในการจัดแบ่งหมวดหมู่นั้นเราจะต้องพิจารณาจากเนื้อหา รายละเอียดของเอกสารนั้นๆ ในระบบห้องสมุดทั่วไปเราอาจจะต้องอ่านเอกสารทั้งหมดก่อนเพื่อที่จะทำการจัดหมวดหมู่ให้แก่เอกสารนั้นๆ ซึ่งเป็นการสูญเสียเวลาและทรัพยากร เราจึงใช้ Machine Learning ในการจัดแบ่งหมวดหมู่เอกสารแทนการอ่านของมนุษย์

document analysis จะเป็นการวิเคราะห์ layout ของเอกสารโดยรวม จะพิจารณาว่าเอกสารนั้นเป็น text, image,.....

document understanding เมื่อทำการวิเคราะห์ layout ของเอกสารแล้ว ขั้นตอนต่อไปเราก็จะทำการดูส่วนประกอบต่างๆในเอกสาร เช่น title , authors,..... ซึ่งเราจะเรียกส่วนต่างๆเหล่านี้ว่า logical objects และเมื่อนำ logical objects เหล่านี้มาจัดเป็นโครงสร้าง เราจะเรียกว่า logical structure

document classification คือ การจัดแบ่งเอกสารออกเป็นหมวดหมู่โดยพิจารณาจาก layout และ logical structure

ในขั้นตอนของ document analysis นั้นเราจะทำการแยกรูปแบบของเอกสารโดยแบ่งเป็นส่วนๆ แต่ละส่วนสามารถแยกได้ว่าเป็นแบบ namely text, picture, graphic, horizontal and vertical solid black line เราจะพิจารณาแค่ลักษณะภายนอกเท่านั้น ไม่สนใจรายละเอียดภายในของแต่ละส่วน เราทำขั้นตอนนี้เพื่อลดความซับซ้อนในขั้นตอนต่อๆไป ต่อไปเราก็จะต้องพิจารณาว่าแต่ละหมวดหมู่ที่เราจะแบ่งนั้นมี model แบบใด หลังจากนั้นเราก็จะต้อง หา model ที่เป็น logical structure ของ แต่ละ class model ของแต่ละหมวดหมู่เราจะใช้ในการทำ document classification ส่วน model ที่เป็น logical structure เราจะใช้ในการทำ document understanding

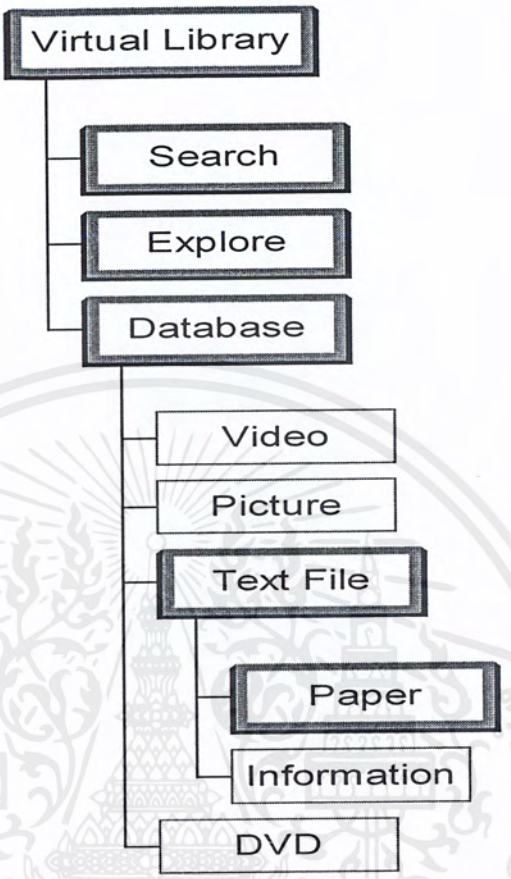
ในขั้นตอนการทำ document understanding นั้นมีความเป็นไปได้ที่แต่ละ logical object จะมีความสัมพันธ์กัน เช่น สาขาวิชาของผู้แต่ง อาจมีความเกี่ยวข้องกับหัวข้อเรื่อง และสามารถสรุปได้ว่า การพิจารณาความเกี่ยวข้องระหว่าง logical object จะช่วยเพิ่มความถูกต้องในการจัดหมวดหมู่ และมีความเป็นไปได้ที่เราจะดูสถิติต่างๆเพื่อที่จะหาความสัมพันธ์ระหว่าง logical object ต่างๆก่อนที่จะเริ่มการทำขั้นตอนต่างๆ

2.1.3 ความหมายของห้องสมุดเสมือน (Virtual Library)

ห้องสมุดเสมือน คือ การจัดการทรัพยากรจากหลายสื่อ ให้อยู่ในรูปดิจิตอลมีการออกแบบการเข้าถึงเนื้อหาสารสนเทศ ให้เป็นประโยชน์แก่ผู้ใช้ และมีเครื่องมือ หรือวิธีการช่วยค้นสารสนเทศในระบบเครือข่าย ที่เชื่อมกันได้ทั่วโลก

การเก็บข้อมูลของห้องสมุดเสมือนประกอบไปด้วยเอกสารดิจิตอล และแหล่งข้อมูลจากอินเทอร์เน็ต ซึ่งจะทำการเชื่อมโยงไปยังเอกสารดิจิตอลอื่นที่อยู่ในที่ใดก็ได้ในระบบอินเทอร์เน็ต ห้องสมุดเสมือนจะทำการควบคุมเพียงแค่การเชื่อมโยงเท่านั้น จะไม่มีส่วนเกี่ยวข้องกับข้อมูลที่มีการเชื่อมโยงไปถึง นอกจากนี้แล้ว ห้องสมุดเสมือนยังอาจมีการจัดทำ Digital catalogs ซึ่งจะมี metadata ของเอกสารที่ได้เก็บไว้ และอาจมีการกำหนดขอบเขตในการเข้าถึงข้อมูลที่แตกต่างกันสำหรับผู้ใช้แต่ละคนอีกด้วย

2.1.4 ส่วนประกอบของห้องสมุดเสมือน



รูปที่ 2-1 แสดงโครงสร้างของห้องสมุดเสมือน

จากรูปข้างต้นจะเป็นส่วนประกอบต่างๆที่มีให้บริการในห้องสมุดเสมือน ในส่วนของโครงงานนั้นจะจัดทำเฉพาะในส่วนที่ระบายกรอบเข้มเท่านั้น

ข้อมูลที่มีการติดต่อผ่านทางอินเตอร์เน็ตก็มีหลายชนิด ซึ่งข้อมูลเหล่านี้ก็อาจจะเป็นส่วนหนึ่งของห้องสมุดเสมือนด้วย สามารถแบ่งออกได้เป็น 3 ประเภท คือ

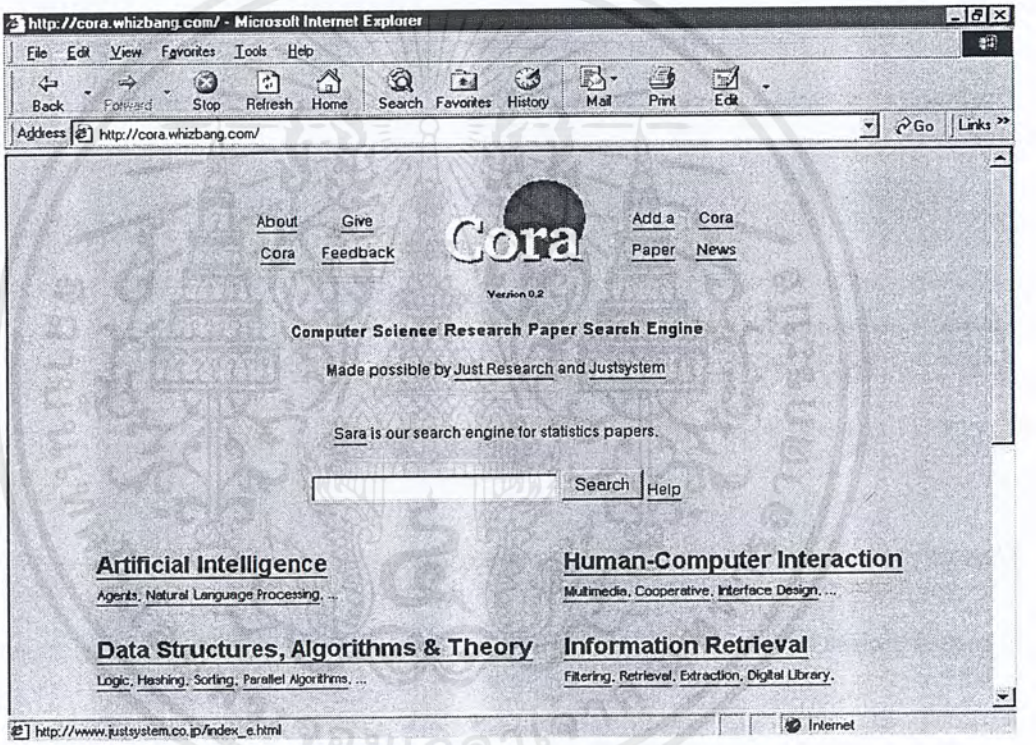
1. Informal documents – เป็นข้อมูลที่ไม่เป็นทางการ อาจเป็นข้อมูลส่วนตัว เช่น โฮมเพจของแต่ละบุคคล ซึ่งจัดทำได้โดยไม่มีข้อจำกัดใดๆ
2. Formal documents – เป็นข้อมูลซึ่งหามาจากหลายที่หรือหลายแหล่งข้อมูล
3. Official documents – เป็นผลงานที่หามาโดย centralized library เอง เช่น theses, reports, dissertations

Official documents informal และ formal document จะถูกจัดการเก็บไว้ในเซิร์ฟเวอร์หลายๆ เซิร์ฟเวอร์แยกกันไป (distributed server) เมื่อต้องการข้อมูลแบบ formal document ก็จะทำการก็อปปี้ข้อมูลนั้นจากห้องสมุด

2.1.5 ตัวอย่างของเว็บไซต์ที่มีการให้บริการเหมือนห้องสมุดเสมือน

1. www.cora.whisbang.com

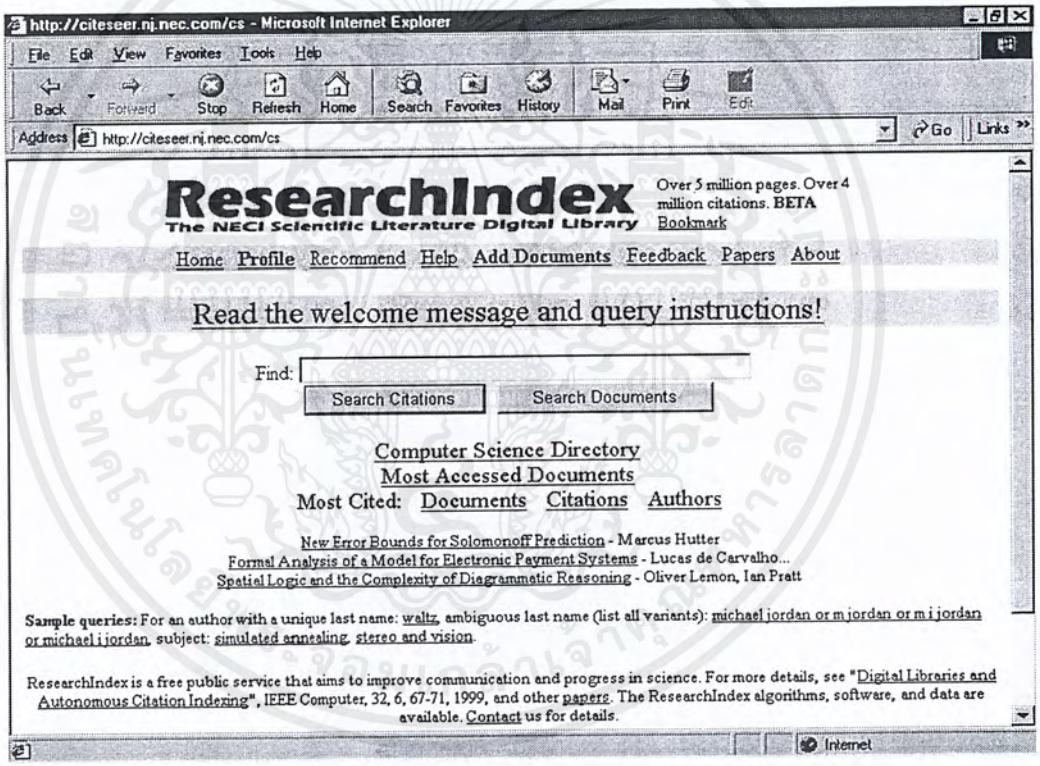
cora เป็นเว็บไซต์ที่รวบรวมข้อมูลต่างๆเกี่ยวกับคอมพิวเตอร์โดยมีการแยกเป็นหมวดหมู่อย่างชัดเจนถึง 10 หมวดหมู่ ได้แก่ Artificial Intelligence, Data Structure, Algorithms & Theory, Databases, Encryption & Compression, Hardware & Architecture, Human-Computer Interaction, InformationRetrieval, Networking, Operating Systems และ Programming โดยในแต่ละหมวดหมู่จะแบ่งเป็น subcategory ย่อยๆลงไปอีกด้วย ผู้ใช้สามารถเข้ามาทำการ search หาข้อมูลที่ต้องการได้ และยังสามารถเข้ามาทำการ explore ดูได้ว่าแต่ละหมวดหมู่แยกย่อยเป็น subcategory ใดอีกบ้าง และมีข้อมูลอะไรอยู่บ้าง



รูปที่ 2-2 ตัวอย่างเว็บไซต์ของ cora

2. www.citeseer.com

โดยในเว็บไซต์นี้ผู้ใช้สามารถที่จะเข้ามาค้นหาข้อมูลได้ แต่ไม่สามารถที่จะเข้ามาทำการเปิดดูหมวดหมู่ต่างๆได้เพราะไม่ได้ทำการจัดแบ่งหมวดหมู่ไว้ โดยเมื่อ citeseer ทำการดาวน์โหลดข้อมูลมาก็จะทำการ parse เพื่อให้ได้ข้อมูลในหัวข้อต่างๆที่ต้องการ เช่น หัวข้อเรื่อง บทคัดย่อ ความถี่ของคำ และ citation list ใน citeseer จะมีความสามารถพิเศษคือสามารถดูได้ทั้งเอกสารที่มี cite มายังเอกสาร ปัจจุบัน และยังสามารถดู เอกสารที่ เอกสาร ปัจจุบัน cite ไปด้วย ลักษณะเด่นอีกอย่างหนึ่งของ citeseer คือ มีการเก็บข้อมูลของผู้ใช้ว่าสนใจข้อมูลเกี่ยวกับเรื่องอะไร และเมื่อมีข้อมูลใหม่ๆเกี่ยวกับเรื่องนั้นก็จะบอกผู้ใช้โดยการส่งไปทางอีเมล์ หรือผ่านทาง web-based interface นอกจากนี้ข้อมูลของผู้ใช้จะมีการเปลี่ยนแปลงในเรื่องของความสนใจโดยใช้ระบบตอบกลับ (feedback) และ machine learning โดยที่จะดูลักษณะในการลิ้งค์ และการตอบสนองต่อรายการข้อมูลที่ได้แนะนำผู้ใช้ไป การใช้ระบบดังกล่าวนี้จะทำให้ข้อมูลที่จะแนะนำ ผู้ใช้ ในครั้งต่อๆไปเกี่ยวข้องกับสิ่งที่ผู้ใช้สนใจมากยิ่งขึ้น



รูปที่ 2-3 แสดงตัวอย่างเว็บไซต์ของ researchIndex

3. http://vlib.org/

4. http://www.csu.edu.au/education/library.html

5. http://wdvl.internet.com/Vlib/

6. http://www.alcazar.com/wwwvl_idc/

2.1.6 บริการของห้องสมุดเสมือนโดยทั่วไป

2.1.6.1 มีการสร้างการเข้าถึงข้อมูลข่าวสารของห้องสมุด โดยจะมีการบริการค้นหาข้อมูลตามที่ผู้ใช้ต้องการไม่ว่าจะเป็นค้นหาจาก ผู้ประพันธ์, สำนักพิมพ์, ปีที่พิมพ์ หรือคีย์เวิร์ดต่างๆ และจะแสดงผลแอดเดรสของเว็บไซต์ที่เกี่ยวข้องให้ทราบทั้งหมด

2.1.6.2 จัดทำ online catalog นั่นคือสามารถให้ผู้ใช้ร้องขอเอกสารตามที่ผู้ใช้ต้องการได้ และทางเจ้าของเว็บไซต์ห้องสมุดเสมือนจะทำการจัดส่งเอกสารที่ผู้ใช้ต้องการไปให้ผู้ใช้ โดยอาจจะมีการจัดส่งไปทางอีเมล์ของผู้ใช้ที่มีการสมัครเป็นสมาชิกของห้องสมุดเสมือนเอาไว้

2.1.6.3 มี online reference มีการจัดหมวดหมู่ของ online full text , multimedia documents ที่น่าสนใจ ทำให้ผู้ใช้สามารถเข้าไปดูได้ว่าในแต่ละหมวดหมู่มีข้อมูลหรือเอกสารใดที่น่าสนใจบ้าง

2.1.6.4 มีการ link ไป abstract ของเอกสาร หรือ เอกสารทั้งหมด เพื่อให้ผู้ใช้สามารถทราบรายละเอียดหรือบทคัดย่ออย่างคร่าวๆของเอกสารนั้นๆ

2.1.6.5 มี document delivery service ส่งเอกสารให้แก่ผู้ใช้

2.1.7 ห้องสมุดเสมือนกับเสิร์ชเอ็นจิน

Search Engine จะสามารถแบ่งได้เป็น 2 ประเภท คือ robot (web indexing) และแบบ directory (web cataloging)

ห้องสมุดเสมือนเป็นเสิร์ชเอ็นจินตัวแบบไดเรกทอรี นั่นคือ มนุษย์จะเป็นผู้ทำการเลือกแหล่งข้อมูลและทำการจัดแบ่งเป็นหมวดหมู่เอง การค้นหาข้อมูลในห้องสมุดเสมือนนั้น อาจจะทำการค้นหาจากคีย์เวิร์ดที่พิมพ์ลงไป จากนั้นผลที่ได้จากการค้นหาจะแสดงเอกสารที่เกี่ยวข้องกับคีย์เวิร์ดนั้นทั้งหมด เมื่อทำการเลือกเอกสารใดเอกสารหนึ่งก็จะแสดงรายละเอียดทั้งหมดของเอกสารนั้นขึ้นมา รวมถึงจะบอกรายละเอียดคร่าวๆของเอกสารนั้น และแอดเดรสเพื่อที่จะลิงค์ไปยังแหล่งเอกสารเหล่านั้นได้ด้วย

2.1.7.1 เสิร์ชเอ็นจินแบบโรบอตและแบบไดเรกทอรี

โดยทั่วไปแล้ว ผู้ใช้ส่วนมากจะเรียกเสิร์ชเอ็นจินโดยไม่ได้มีการแยกแยะว่าเป็นเสิร์ชเอ็นจินแบบโรบอตหรือแบบไดเรกทอรี ซึ่งในความเป็นจริงแล้วทั้งเสิร์ชเอ็นจินแบบโรบอตและเสิร์ชเอ็นจินแบบไดเรกทอรีทั้งสองไม่เหมือนกัน ต่างกันดังนี้

- เสิร์ชเอ็นจินแบบโรบอต

เช่น Hotbot จะทำการสร้างลิสต์ได้โดยอัตโนมัติ เสิร์ชเอ็นจินจะไปค้นหาข้อมูลที่ต้องการหาทั่วทุกเว็บไซต์ตามแต่ว่าแต่ละเว็บไซต์จะสามารถลิ้งค์ต่อไปยังเว็บไซต์ใดได้บ้าง ถ้าเว็บเพจใดมีการเปลี่ยนแปลง เสิร์ชเอ็นจินก็จะพบความเปลี่ยนแปลงนี้ได้โดยอัตโนมัติ ซึ่งการทำงานของเสิร์ชเอ็นจินจริงนั้น มันจะทำการไปค้นหาข้อมูลจากเว็บไซต์ต่างๆในขณะนั้นทันที ไม่ได้ไปค้นหาข้อมูลจากที่มีเก็บอยู่ในฐานข้อมูลของเว็บไซต์ที่เป็นเสิร์ชเอ็นจินนั้นๆ

- เสิร์ชเอ็นจินแบบไดเรกทอรี

เช่น Yahoo ต้องขึ้นอยู่กับมนุษย์ที่ทำลิสต์ โดยมนุษย์จะเป็นผู้เก็บคำอธิบายเกี่ยวกับเว็บไซต์แต่ละเว็บไซต์เอาไว้เอง แล้วเมื่อถึงเวลาค้นหาก็ไปค้นหาในฐานข้อมูล จากนั้นก็เปรียบเทียบว่าตรงกับที่ต้องการหรือไม่ถ้ามีการเปลี่ยนแปลงใดๆเกิดขึ้น เว็บเพจก็จะไม่มีการเปลี่ยนแปลงอะไรในลิสต์จนกว่าผู้ที่คุมอยู่จะใส่ข้อมูลใหม่เข้าไปในฐานข้อมูล

2.1.7.2 ส่วนประกอบ และการทำงานของเสิร์ชเอ็นจิน

เสิร์ชเอ็นจิน มีส่วนใหญ่ๆอยู่ 3 ส่วน

1. ส่วนที่ใช้สำหรับรวบรวมข้อมูล เรียกว่าโรบอตหรือสไปเดอร์ (Spider) หรือจะเรียกว่าครอเลอร์ (Crawler) ก็ได้ โรบอตจะเป็นตัวเข้าหาเว็บเพจเพื่ออ่าน แล้วตามลิงก์ไปสู่หน้าอื่นๆ ในไซต์ โดยจะเข้าไปเป็นประจำอาจจะทุกๆ เดือน หรือ 2 เดือน เพื่อดูความเปลี่ยนแปลง
2. ทุกอย่างที่โรบอทเจอจะส่งต่อไปที่ส่วนที่ 2 คือ อินเด็กซ์หรือเรียกอีกอย่างว่าแคตาล็อก (catalog) เป็นเหมือนกับหนังสือเล่มยักษ์ที่มีส่วนที่เป็นสำเนาของทุกๆ เว็บเพจที่โรบอทเข้าไป ถ้าเว็บเพจมีการเปลี่ยนแปลงก็จะมีการเปลี่ยนแปลงข้อมูลใหม่ด้วย
3. ส่วนที่ใช้สำหรับค้นหาข้อมูลจากส่วนที่สองคือ โปรแกรมเสิร์ชเอ็นจิน เป็นส่วนที่ 3 เป็นโปรแกรมที่เข้าไปหาเป็นทุกๆ เพจ ที่ถูกเก็บไว้ในอินเด็กซ์ ว่าตรงกับคำที่หาหรือไม่ แล้วเรียงตามลำดับความสัมพันธ์ว่าอันไหนใกล้เคียงมากที่สุด

ทุกๆ เสิร์ชเอ็นจิน มีส่วนประกอบเหมือนกันทั้ง 3 ส่วน แต่แตกต่างกันตรงที่การนำเอาทั้ง 3 ส่วนมารวมกัน ดังนั้นจึงเป็นเหตุผลว่าทำไมการหาคำคำเดียวกันจึงได้ผลที่ต่างกันใน เสิร์ชเอ็นจิน แต่ละตัว

2.1.8 ขั้นตอนในการสร้างห้องสมุดเสมือน

1. **Define the Collection**

จะต้องทำการเลือก Internet resources ที่ต้องการเก็บไว้ โดยมีปัจจัยในการพิจารณา 3 อย่าง ดังต่อไปนี้

- Subject หัวข้อหรือประเภทของข้อมูลที่ต้องการจัดเก็บ
- Scope จำกัดประเภทของ resource ที่ต้องการเก็บ
- Audience – ผู้เข้าชมเว็บไซต์ หรือ ผู้ใช้ นั่นเอง

2. **Determine if a Similar Collection Already Exists**

ลองหาดูว่ามี collection ที่ใกล้เคียงกับที่วางแผนไว้หรือไม่ ถ้ามีอาจทำการเปลี่ยนแปลง focus ไปเล็กน้อย แต่ถ้าที่มีอยู่แล้วไม่ค่อยดี คิดว่าทำได้ดีกว่าก็ควรทำต่อไป

3. **Determine How Much Material Exists for Your Collection**

อาจจะลองค้นหาข้อมูลในอินเตอร์เน็ต เพิ่มเติมเพื่อดูว่ายังมีข้อมูลอะไรอย่างอื่นอีกหรือไม่ที่เราควรเก็บไว้ใน collection

4. **Decide about Descriptive Cataloging**

พิจารณาว่าใน catalog ควรจะมีองค์ประกอบอะไรอยู่บ้าง เช่น ผู้ประพันธ์, ชื่อเรื่อง, บทคัดย่อ, สำนักพิมพ์, ปีที่พิมพ์, คีย์เวิร์ดของเว็บเพจนั้นๆ, URL ... โดยสิ่งที่เก็บอาจขึ้นอยู่กับประเภท หรือรูปแบบของสิ่งที่เก็บ

5. **Plan the subject access**

ทำการออกแบบว่าเราจะสามารถให้ผู้ใช้ทำการเข้าถึงข้อมูลได้กี่วิธี และแต่ละวิธีมีการทำงานอย่างไรบ้าง เช่น ผู้ใช้อาจทำการ search จาก ผู้ประพันธ์, ชื่อเรื่อง, บทคัดย่อ ซึ่งถ้าเรากำหนดให้ผู้ใช้สามารถเข้าถึงข้อมูลได้หลายวิธี ความซับซ้อนในการเก็บข้อมูลใน database ก็จะยิ่งเพิ่มมากขึ้น

6. **Build the Database**

โดยชนิดของฐานข้อมูลที่จะใช้เก็บนั้นขึ้นอยู่กับจำนวนของข้อมูลที่ต้องการเก็บ ถ้าไม่เกิน 100 items ก็อาจเก็บในหน้าเอชทีเอ็มแอล ถ้าระหว่าง 100-5000 items ก็ควรเก็บไว้ใน personal database product เช่น Microsoft Access ถ้าเกิน 5000 items ก็ควรเก็บไว้ในฐานข้อมูลที่มีประสิทธิภาพสูง เช่น Oracle หรือ Microsoft SQL ดังนั้นในการเลือกใช้ฐานข้อมูลก็จะต้องพิจารณาเลือกใช้ฐานข้อมูลที่เหมาะสมเพื่อให้มีประสิทธิภาพในการทำงานสูงสุด

7. **Select Items for the Core Collection**

หา Items ที่ต้องการเก็บไว้ใน collection โดยแหล่งที่ดีที่สุดในการหาคือ collection ที่พบในข้อ 2

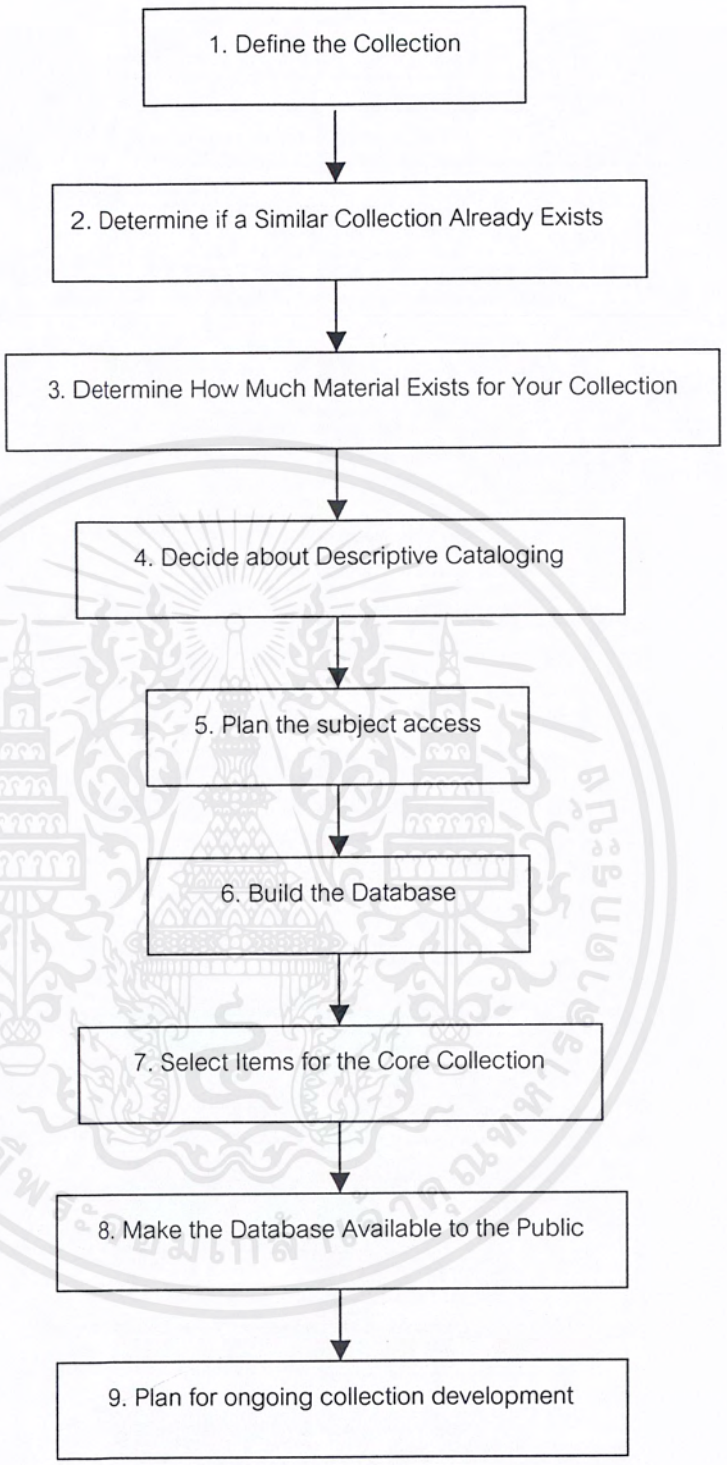
8. **Make the Database Avaliable to the Public**

อาจมีการสร้างการเข้าถึงแบบโดยตรง อาจทำเป็นเอชทีเอ็มแอลหรือซีจีไอ

9. **Plan for ongoing collection development**

อินเตอร์เน็ตมีการเปลี่ยนแปลงอย่างรวดเร็ว จึงต้องมีการดูแลบำรุงรักษา Internet collection

- Dealing with currently existing items ดูว่า items ที่มีอยู่แล้วนั้น site มีการเปลี่ยนแปลงหรือไม่
- Finding new items หา resource ใหม่ๆ โดยมีหลายวิธีในการหาเช่น ขอคำแนะนำจากผู้ใช้

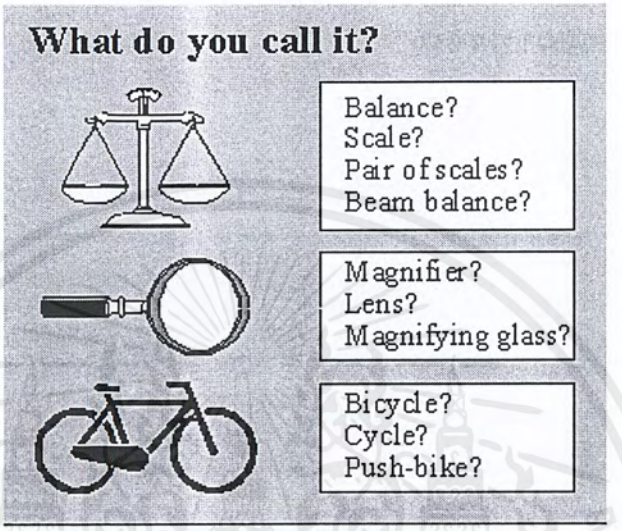


รูปที่ 2-4 แสดงขั้นตอนการสร้างห้องสมุดเสมือน

2.2 Thesaurus

เพื่อให้ได้ประสิทธิภาพสูงสุดในการค้นหาข้อมูลจึงได้มีการนำ Thesaurus มาใช้เพื่อช่วยในการค้นหาข้อมูลนั้นๆ เพราะว่าวัตถุหรือสิ่งของแต่ละอย่างนั้นมีชื่อเรียกได้หลายอย่างแล้วแต่ผู้ใช้ โดยแต่ละชื่อนั้นอาจจะมีความหมายเหมือนกันหรือใกล้เคียงกันก็ได้

เช่น



รูปที่ 2-5 แสดงความเข้าใจการใช้ Thesaurus

โดยในการค้นหาข้อมูลนั้น มีจุดมุ่งหมายว่าไม่ว่าผู้ใช้ จะทำการค้นหาด้วยคำว่า bicycle, cycle หรือ push-bike ผลของข้อมูลที่ออกมาควรจะใกล้เคียงกัน

จุดประสงค์หลักของ Thesaurus ก็คือ สามารถนำคำที่ผู้ใช้ใช้ในการค้นหาไป match กับอินเด็กซ์เทอม ที่มีในระบบได้ โดยคำศัพท์ที่มีความหมายเหมือนกันหรือใกล้เคียงกัน เราจะใช้คำศัพท์เพียงคำเดียวเท่านั้นเป็นตัวแทนในการจัดแบ่งหมวดหมู่ เช่น frocks และ dresses เราอาจจะเลือกใช้คำว่า dresses ในการค้นหาแทน frocks โดยเราจะการลิงก์คำเหล่านั้น โดยใช้ USE และ USE FOR

เช่น	Dresses	USE FOR	Frocks
	Frocks	USE	Dresses

ถ้าผู้ใช้เข้ามาค้นหาโดยใช้คำว่า Frocks เราก็จะทำการค้นหาโดยใช้คำว่า Dresses แทน ดังนั้น ข้อมูลที่ได้ออกมาจะเหมือนกันไม่ว่า เราจะใช้คำใดในการค้นหาก็ตาม

เราจะใช้ความสัมพันธ์ USE และ USE FOR กับคำศัพท์ที่มีความหมายเหมือนกันหรือใกล้เคียงกัน ซึ่งเป็นคำที่เราไม่ต้องการแยกหมวดหมู่

Nuclear power	USE FOR	Nuclear energy
Nuclear energy	USE	Nuclear power
Perambulators	USE FOR	Baby carriages
Baby carriages	USE	Perambulators
Perambulators	USE FOR	Prams
Prams	USE	Perambulators

การทำ Thesaurus ในลักษณะนี้จะเป็นการเพิ่ม access points ในการค้นเอกสารนั้นๆ ทำให้เราสามารถเข้าถึงเอกสารนั้นได้ง่ายขึ้น

การทำ Thesaurus นั้นเราต้องคำนึงถึงความสัมพันธ์แบบลำดับชั้นด้วย (Hierarchical relationships) โดยจะมีความสัมพันธ์แบบ narrow term (NT) และ broader term (BT) โดย narrow term จะมีความหมายที่เฉพาะเจาะจงมากกว่า broader term โดยคำศัพท์ที่เป็น narrow term จะมีลักษณะที่ได้รับ (inherit) มาจาก broader term ความสัมพันธ์ใน Thesaurus นี้จะต้องเป็นจริงเสมอโดยไม่มีความเกี่ยวข้องกับ context

เช่น Mice เป็น narrow term ของ Rodents Mice BT Rodents

Rodents เป็น broader term ของ Mice Rodents NT Mice

แต่ Mice ไม่เป็น narrow term ของ Pests เพราะ หนูที่เป็นสัตว์เลี้ยง ไม่ได้เป็นสัตว์ที่ทำการรบกวน

Jackets

NT Anoraks
Blazers
Boleros
Dinner jackets
Donkey jackets
Flying jackets
Sports jackets

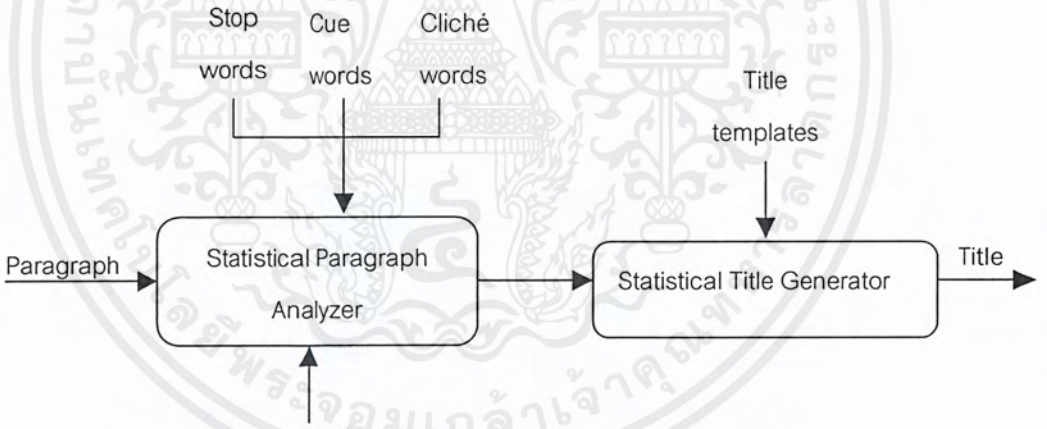
โดยในการ search คำว่า jackets นั้น ควรจะทำการ search NT ของมันด้วย

บทที่3 การจัดทำรายการอัตโนมัติ

การจัดหาอินเด็กซ์เทอมของเอกสารนั้น ไม่ว่าจะเป็นการจัดทำอินเด็กซ์อัตโนมัติ (Automatic Indexing) การจัดทำรายการย่ออัตโนมัติ (Automatic Abstracting) และการจัดทำหัวข้อเรื่องอัตโนมัติ (Automatic Title generation) จะมีจุดประสงค์หลักของการทำงานเหมือนกัน คือ การโปรแกรมคอมพิวเตอร์ให้สามารถแยกแยะหัวข้อหรือแยกแยะคำต่างๆที่สำคัญที่เป็นใจความหลักและสัมพันธ์กับหัวข้อของเอกสารนั้นๆ โดยทำการวิเคราะห์จากเอกสารที่ได้มา โดยภาพรวมของระบบคือให้เท็กซ์ไฟล์เป็นอินพุทของระบบ และเอาท์พุทของระบบต้องการอินเด็กซ์เทอม หัวข้อเรื่อง เรือรายการย่ออัตโนมัติก็ตาม ซึ่งสิ่งเหล่านี้คือใจความหรือคำศัพท์ที่สำคัญของเอกสารนั้นๆ

3.1 การจัดทำหัวข้อเรื่องอัตโนมัติ (Automatic Title generation)

การจัดทำหัวข้อเรื่องอัตโนมัติ คือการโปรแกรมคอมพิวเตอร์เพื่อให้โปรแกรมสามารถที่จะหาหัวข้อเรื่องที่เป็นตัวแทนของเอกสารนั้นมาให้ได้ โดยทำการวิเคราะห์จากข้อมูลเท็กซ์ไฟล์ว่ามีคำใดบ้างที่สื่อถึงความสัมพันธ์และความสำคัญของเอกสารนั้น



รูปที่ 3-1 แสดงขั้นตอนในการสร้างหัวข้อเรื่องอัตโนมัติ

จากรูปจะเห็นได้ว่าขั้นตอนในการทำหัวข้อเรื่องอัตโนมัตินั้นจะแบ่งออกเป็น 2 ส่วนที่สำคัญ คือ

1. Statistical Paragraph Analyzer

จะเห็นได้ว่าการจัดทำหัวข้อเรื่องอัตโนมัตินั้นจะรับอินพุทของระบบเข้ามาเป็นข้อความของเอกสารทั้งหมดและผ่านกระบวนการวิเคราะห์ข้อมูลเหล่านั้น โดยจะต้องนำ Stop word list มาพิจารณาด้วย ซึ่ง

Stop word list คือ คำที่ไม่มีความหมาย หรืออาจจะมีความหมายแต่ไม่สามารถสื่อความหมายที่สำคัญได้ นอกจากนี้ยังรวมถึงคำที่พบบ่อยๆในเอกสารซึ่งคำเหล่านี้จะมีความถี่ของคำมากเกินไป จนไม่สำคัญกับเอกสารที่จะนำมาวิเคราะห์ เช่น

- Articles - a, an, the
- Pronoun – he, she, it, they, I, you
- Verb – is, am, are, can, should
- Conjunction – and, but
- Adjective – all, still
- Adverb – already, about, also, along
- Noun – example

หลังจากนำ Stop word list มาพิจารณาและทำการตัดคำศัพท์ที่มีอยู่ใน Stop word list ซึ่งจะเป็นการตัดคำศัพท์ที่ไม่มีความจำเป็น (Insignificant Words) ออก ได้ประมาณถึง 40-50% ของข้อความทั้งหมดที่รับเข้ามาเป็นอินพุท

เมื่อทำการตัด Stop Word List แล้ว ก็จะใช้ Online Webster Dictionary เพื่อนำมาช่วยในการจัดกลุ่มคำที่มีความหมายเหมือนกัน (Synonym of Words) โดยจะทำการ Matching คำแต่ละคำกับ Synnonym List เพื่อค้นหาคำที่มีความหมายเหมือนกันและสามารถจัดรวมเป็นกลุ่มเดียวกันได้ จากนั้นทำการ Stemming นั่นคำทำให้คำนั้นๆเปลี่ยนกลับเป็นรากศัพท์เดิม (Root Word) เช่น ate -> eat, eaten -> eat และเริ่มทำการนับความถี่ของคำ

นอกจากนี้ยังต้องวิเคราะห์โครงสร้างของ paragraph นั้นด้วย โดยจะต้องนำ Cue words (คำที่จะบ่งบอกถึงความสำคัญของคำที่ตามมา) และ Cliché words (คำที่ใช้ซ้ำๆบ่อยที่มักจะนำคำเหล่านั้นมาจัดทำเป็นหัวข้อเรื่อง) มาพิจาณาด้วย เพื่อจะได้ทราบความสำคัญของคำแต่ละคำว่ามีความหมายมากน้อยเพียงใด แล้วทำการวิเคราะห์ข้อมูลที่ได้มา

2. Statistical Title Generator

การพิจารณาถึงแต่ละข้อความ แล้วทำการเลือกข้อความที่สามารถแสดงถึงความหมายของข้อมูลนั้นได้ดีที่สุด เพื่อทำการเลือกหัวข้อเรื่องขึ้นมานั้น จะมีกระบวนการในการตัดสินใจโดยจะทำการพิจารณาจากแต่ละ Templates ซึ่งจะมีกฎในการตัดสินใจเลือกแต่ละ Templates และที่สำคัญควรจะเลือก Templates ที่ไม่ซับซ้อน ดังตัวอย่างต่อไปนี้

T1 : <NOUN PHRASE>

T2 : <NOUN> "AND" <NOUN PHRASE>

T3 : <NOUN PHRASE 1> "AND" <NOUN PHRASE 2>

T4 : <NOUN> ["," <NOUN>]["AND" <NOUN>]

T5 : <Ti> ":" "DET" <CLICHÉ WORD>

T6 : "COMMENTS ON" <Ti>

รูปที่ 3-2 แสดงเท็มเพลตของการสร้างหัวข้อเรื่องอัตโนมัติ

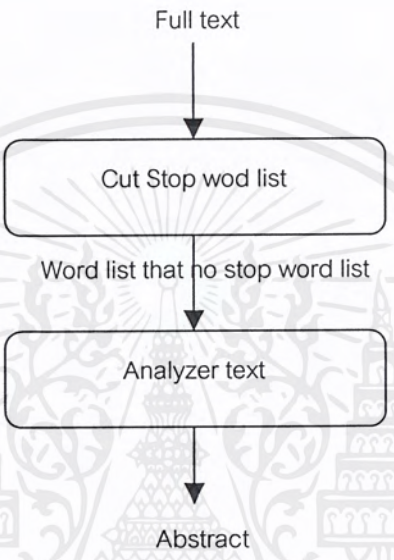
กฎในการตัดสินที่จะทำการ Match แต่ละ phrase กับ Template

1. ทำการเลือก T1 ถ้า Noun Phrase ตัวใดตัวหนึ่งมีค่าสถิติหรือค่าความถี่สูงและแตกต่างกับ Noun Phrase ตัวอื่นมาก
2. ทำการเลือก T2 ถ้า Noun Phrase ตัวใดตัวหนึ่งมีค่าทางสถิติหรือค่าความถี่สูง แต่ไม่บรรจุ Noun ที่มีค่าทางสถิติสูงด้วยนั้น ให้ทำการรวมคำทั้งสองโดย นำ Noun ไว้ หน้า Noun Phrase แล้วเชื่อมด้วยคำว่า "AND"
3. ทำการเลือก T3 ถ้าค่าทางสถิติสูงสุดของ Noun Phrase 2 ตัวใดๆ มีค่าใกล้เคียงกันให้ทำการรวม Noun Phrase ทั้ง 2 ตัว นั้น แล้วเชื่อมด้วยคำว่า "AND"
4. ทำการเลือก T4 ถ้าค่าทางสถิติของ Noun Phrase ไม่สูงเพียงพอที่จะเลือก Noun Phrase นั้นมาใช้ได้ ให้ทำการเลือก Noun ที่มีค่าทางสถิติสูงชุดหนึ่ง แล้วทำการเชื่อม Noun เหล่านั้น ด้วย "," และให้เชื่อม Noun ตัวสุดท้ายด้วย "AND"

   T1 – T4 จะถูกทำการเลือกขึ้นมาขึ้นอยู่กับแต่ละสภาวะหรือเหตุการณ์ของค่าทางสถิติที่คำนวณหาออกมาได้ เพื่อทำการเลือกหัวข้อเรื่องที่เหมาะสมออกมา นอกจากนี้ยังสามารถที่จะ switch ไปยัง T5 – T6 เพื่อให้เกิดการปรับปรุงหัวข้อเรื่องให้มีคุณภาพขึ้นได้ นั่นคือ จะมีการเพิ่มหรือขยายความในส่วนของหัวข้อเรื่องนั่นเอง
5. Switch มายัง T5 ถ้าค่าทางสถิติของ Noun และ Noun Phrase ไม่สูงเพียงพอ แต่ Noun หรือ Noun Phrase นั้น บรรจุคำที่เป็น Cliché word (คำที่มักจะเลือกนำมาใช้เป็นหัวข้อเรื่องอยู่บ่อยๆ) อยู่ ให้ทำการต่อท้าย Template Ti (T1 – T4) ที่ทำการเลือกมาแล้ว ด้วย " : " และต่อท้ายด้วยคำที่เป็น Cliché word นั้นลงไป
6. Switch มายัง T6 ถ้าค่าทางสถิติของ Noun และ Noun Phrase ต่ำ และ นอกจากนี้ Noun หรือ Noun Phrase นั้นไม่ได้บรรจุคำที่เป็น Cliché Word อยู่ (ไม่ได้ Switch เข้า T5 หรือ T5 Failed นั่นเอง) ในกรณีนี้แสดงว่า เนื้อหาของข้อมูลค่อนข้างมีความคลุมเครือไม่สามารถที่จะกำหนดหัวข้อเรื่องหรือเฉพาะเจาะจงได้อย่างชัดเจน ดังนั้นให้เติมคำว่า " Comments on" ที่ด้านหน้าของประโยคที่เลือกมาจาก Template Ti (T1 - T4)

3.2 การจัดทำรายการย่ออัตโนมัติ (Automatic Abstracting)

การจัดทำรายการย่ออัตโนมัติ คือ การโปรแกรมคอมพิวเตอร์ให้สามารถเลือกประโยคจากเอกสาร โดยที่ประโยคที่ทำการเลือกมานั้นจะต้องบอกถึงภาพรวมของเอกสารได้ ซึ่งอาจจะมองว่าเป็นบทคัดย่อของเอกสารโดยรวมนั่นเอง โดยจะต้องมีการวิเคราะห์โครงสร้างของประโยคว่ามีความหมายหรือมีความสัมพันธ์ของแต่ละประโยคหรือแต่ละวลีอย่างไร และจะต้องมีการเจาะลึกไปยังความหมายของคำในแต่ละประโยคให้ถูกต้องด้วย



รูปที่ 3-3 รูปแสดงขั้นตอนการจัดทำรายการย่ออัตโนมัติ

จากรูปข้างต้นซึ่งแสดงขั้นตอนการจัดทำรายการย่ออัตโนมัตินั้นก็จะรับอินพุทของระบบเป็น Full-text เข้ามา และทำการจัดแบ่งคำและตัด Stop word list ออก ซึ่งหลังจากนั้น จะมาผ่านกระบวนการของการวิเคราะห์

กระบวนการในการจัดทำรายการย่ออัตโนมัติ

Stop word list จะกำจัดคำสรรพนาม, article หรือคำศัพท์อื่นๆ ที่ไม่จำเป็นออกก่อนเพื่อความสะดวกในขั้นต่อไป

คำที่เหลือจะถูกนับความถี่ของแต่ละคำ แล้วทำการจัดลำดับความถี่ของคำ จากค่าที่มากที่สุดไปยังค่าที่น้อยที่สุด

คำที่มีความถี่มากกว่าค่าที่ตั้งไว้ (Threshold value) จะถือว่าเป็น significant , high frequency เป็นคำที่สำคัญในเอกสารนั้นๆ

ประโยคที่น่าสนใจที่ควรจะต้องพิจารณา คือ ประโยคที่มี high frequency word อยู่ เนื่องจากเป็นประโยคที่มีคำที่สำคัญ

ปัจจัยสำคัญที่จะต้องคำนวณในแต่ละประโยค

- จำนวน cluster ในแต่ละประโยค (cluster คือ กลุ่มคำที่ยาวที่สุดโดยมีขอบเขตของกลุ่มคือ high frequency word และแต่ละ high frequency word จะมีคำอื่นมากั้นได้ไม่เกิน 4 คำ) ต้องพิจารณาว่ามีจำนวน cluster มากน้อยเพียงใด และมีความสำคัญกับประโยคมากแค่ไหน เพื่อจะได้นำมาพิจาณาต่อไปได้
- จะต้องนับจำนวน high frequency word ในแต่ละ cluster แล้วนำมาคำนวณจาก จำนวน high frequency word ^{2} / จำนวน word ทั้งหมดใน cluster
- นำค่าที่คำนวณได้ของแต่ละ cluster ในประโยคมารวมกัน ประโยคที่มีค่าสูงสุดจะถูกเลือก

  เช่น a b c **D** e **F G** h i **J** k l m n o p q r

  สมมุติให้ อักษรตัวใหญ่เป็น high frequency word cluster ของประโยคนี้คือ D - J และ ค่า significance factor = 4*4/7 = 2.3

วิธีการอื่นๆในการจัดทำรายการย่ออัตโนมัติ

- Key method เหมือนกับการนับความถี่
- Cue method จะมี Cue dictionary โดยจะเก็บ list ของคำไว้ว่าคำไหนมี positive weight / negative weight
- Title method วิธีนี้จะคิดว่าคำศัพท์ที่พบใน title หรือ subhead จะเป็นคำที่บอกเนื้อหาภายในเอกสารหรือสามารถเป็นตัวแทนของเอกสารนั้นได้ดี ประโยคที่มีคำศัพท์เหล่านี้อยู่ก็จะถูกเลือกเป็น abstract (ควรใช้วิธีการนับความถี่ร่วมด้วย)
- Location method จะดูตำแหน่งของคำในประโยค และทำการให้ค่าถ่วงน้ำหนักในแต่ละส่วนของประโยคตามความสำคัญ เช่น ประโยคแรก หรือประโยคสุดท้ายของ paragraph, paragraph แรก หรือ paragraph สุดท้าย หรือดู text ที่ตามหลังคำว่า Introductions หรือ Conclusions ประโยคที่พบในบริเวณเหล่านี้จะมี weight มาก

ผลกระทบที่มีต่อการจัดทำรายการย่ออัตโนมัติ

1. Contextual Influence จะต้องพิจารณาสิ่งที่อยู่รอบๆคำนั้นเพราะ จะช่วยบอกว่าควรจะเลือกประโยคนั้นหรือไม่ โดยจะ match text กับ word control list ซึ่งแบ่งเป็น

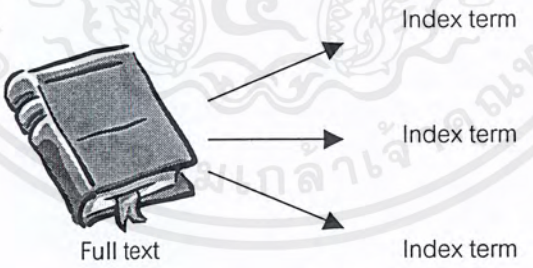
- rejection expression : จะเป็นคำที่บอกว่าประโยคนี้จะเกี่ยวข้องกับ background ไม่ได้บอกเกี่ยวกับวัตถุประสงค์ หรือ ผลของงานในปัจจุบัน
- selection expression : จะเป็นคำที่บอกว่าประโยคนี้น่าจะเกี่ยวข้องกับภาพรวมของเอกสาร เช่น this เอกสาร, this study

2. Intersentence reference เป็นการดูว่าความหมายของประโยคที่เลือกมานั้นขึ้นกับประโยคก่อนหน้าหรือไม่เช่น hence, for this reason ถ้าพบคำเหล่านี้จะแสดงว่าประโยคก่อนหน้านั้นมีความเกี่ยวข้องกันกับประโยคภายหลังคำเหล่านี้ ซึ่งจะต้องนำประโยคข้างหน้ามาร่วมพิจารณาด้วย (ได้มากสุดไม่เกิน 3 ประโยค) ถึงแม้ว่าประโยคข้างหน้านั้นจะไม่มีความเกี่ยวข้องเลยก็ตาม ดังนั้นจึงต้องมีกระบวนการที่เหมาะสมในการที่จะวิเคราะห์ว่าประโยคก่อนหน้านั้นมีความสำคัญมากน้อยเพียงใด หรือว่าถ้าไม่สามารถที่จะวิเคราะห์ได้ก็ต้องหาวิธีการที่จะทำให้เกิดข้อผิดพลาดในการจัดทำรายการย่อให้ได้มีประสิทธิภาพได้มากที่สุด ขั้นตอนนี้จะช่วยให้ abstract ที่ได้มีความต่อเนื่องของประโยคมากขึ้น

3.3 การจัดทำอินเด็กซ์อัตโนมัติ (Automatic Indexing)

การจัดทำอินเด็กซ์อัตโนมัติคือการโปรแกรมคอมพิวเตอร์โดยให้โปรแกรมนั้นสามารถที่จะรับอินพุทของระบบเป็นเท็กซ์ไฟล์ (Text file) และสามารถที่จะให้เอาท์พุทของระบบออกมาเป็นอินเด็กซ์เทอมได้ ซึ่งอินเด็กซ์เทอมนั้นจะเป็นตัวแทนของเอกสารที่จะบอกได้ว่าเอกสารนั้นกล่าวถึงหรือเกี่ยวข้องสัมพันธ์กับเรื่องใด

โดยส่วนมากแล้วการจัดทำอินเด็กซ์อัตโนมัตินั้นจะจัดทำเพื่อที่จะนำไปใช้เกี่ยวกับการจัดเก็บข้อมูลและค้นคืนข้อมูล (Retrieval Information) ให้ได้เร็วที่สุด และเป็นการจัดเก็บข้อมูลให้เป็นระเบียบหมวดหมู่ที่มีประสิทธิภาพด้วย เช่น ใน search engine ต่างๆที่มีการให้บริการ ก็จะทำการจัดเก็บข้อมูลพร้อมทั้งอินเด็กซ์เทอมของเอกสารนั้นๆ เมื่อมีผู้ใช้ต้องการเข้ามาค้นหาข้อมูลต่างๆ ก็ทำการพิมพ์คีย์เวิร์ดที่ต้องการลงไป search engine นั้นก็จะทำการดึงข้อมูลซึ่งจะทำการเปรียบเทียบคีย์เวิร์ดกับอินเด็กซ์เทอมที่มีการจัดเก็บเอาไว้ เพื่อที่จะดึงข้อมูลที่ตรงกับที่ผู้ใช้ต้องการได้



รูปที่ 3-4 ภาพรวมของการจัดทำอินเด็กซ์

3.3.1 Automatic Extraction Indexing

Extraction Indexing คือการหาคำศัพท์ หรือ phrase ที่พบจากเอกสารนั้นมาใช้แทนเนื้อหาของเอกสารทั้งหมด โดย อาจจะดูจากความถี่ที่ได้พบคำนั้น หรือ location ที่พบคำนั้นเช่นใน title, summary

หรือดูจาก context ของคำนั้น โดยปกติการจัด index โดยคนนั้นจะใช้วิธี Assignment Indexing คือ จะเลือกคำจาก Controlled Vocabulary มาใช้

การทำ Automatic Indexing โดยการนับความถี่นั้น จะเริ่มจากการหาคำที่ซ้ำๆ แล้วนับจำนวนครั้งที่พบไว้ จากนั้นนำไปเปรียบเทียบกับ stop list (articles, prepositions, conjunctions) เพื่อตัดคำที่ไม่เกี่ยวข้องออก แล้วนำมาจัดอันดับ คำที่มีอันดับสูงสุดก็จะเป็น index term

จำนวน index term ของแต่ละเอกสารนั้น อาจมีการจำกัดไว้เป็นค่าคงที่, ขึ้นอยู่กับความยาวของเอกสาร, หรือพิจารณาค่าความถี่ว่าเกินค่า threshold ที่ตั้งไว้หรือไม่ นอกจากนี้ index term ที่เก็บจะต้องเป็น root ของคำนั้น เช่น ตัด ed, ing ทิ้ง

3.3.2 Automatic Assignment Indexing

การทำ Assignment Indexing โดยคอมพิวเตอร์ยากกว่า Extraction Indexing การทำ Assignment Indexing นั้นเราจะต้องจัดทำ กลุ่มคำศัพท์ที่เกี่ยวข้องกับอินเด็กซ์เทอมนั้นๆ เช่น acid rain อาจมีคำที่เกี่ยวข้องคือ acid precitation, air pollution, sulfer dioxide

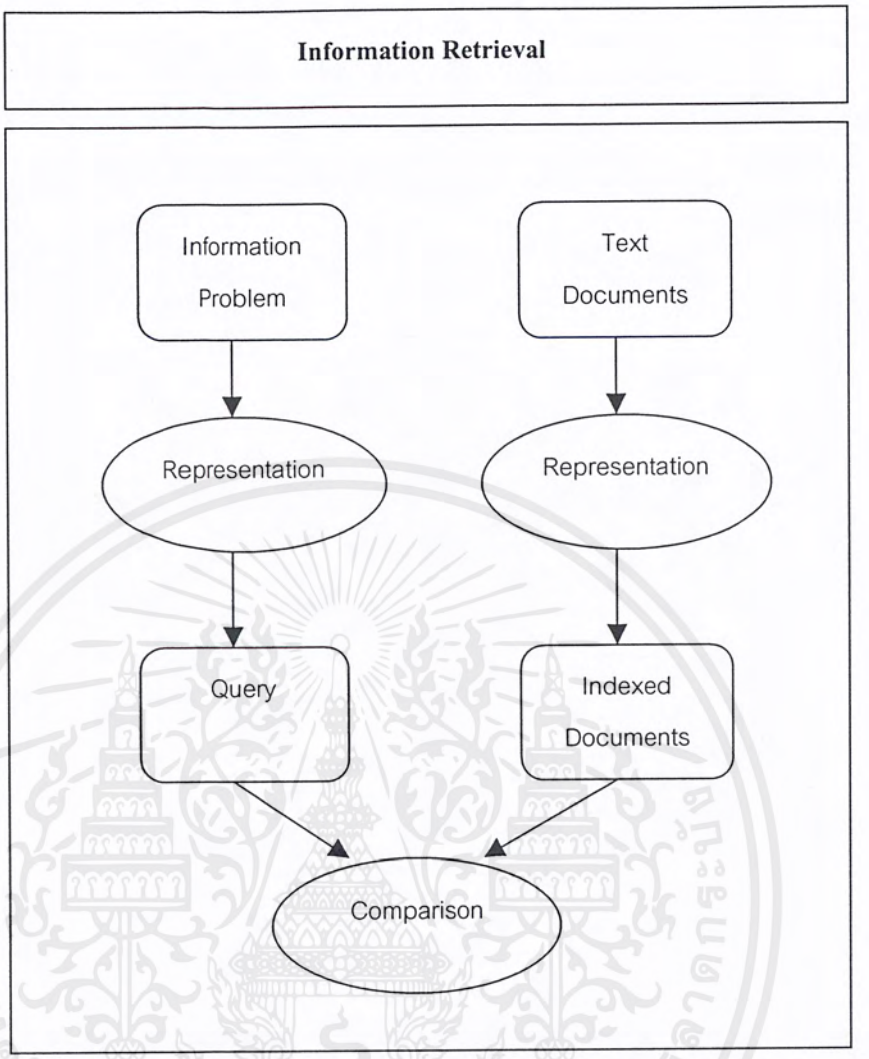
เราจะหาคำศัพท์โดยอาจดูจากความถี่เหมือน Extraction Indexing หรืออาจนำคำศัพท์ในหัวข้อเรื่องมา แล้วนำคำศัพท์ที่ได้มา match กับ กลุ่มคำที่เกี่ยวข้อง เพื่อที่จะกำหนดอินเด็กซ์เทอนั้นให้แก่เอกสาร

- underassignment คือ คำบางคำที่ควรกำหนดแต่ไม่ได้กำหนด (ถ้าคนทำคนจะกำหนดคำนั้นให้เป็นอินเด็กซ์เทอมด้วย)
- overassignment คือ การกำหนดอินเด็กซ์เทอมที่ไม่ควรนำมากำหนด

ระบบ Automatic Indexing โดยทั่วไปแล้วใช้เพื่อช่วย indexer ไม่ได้ทำงานอัตโนมัติจริงๆ เป็น machine-aided

1. ในระบบออนไลน์ ถ้า indexer assign nonstandard term ระบบก็จะทำการตรวจสอบและแจ้งให้ indexer ทราบทันที
2. ระบบจะทำการอ่านเท็กซ์ แล้วเลือกอินเด็กซ์เทอมมาก่อน indexer จะทำการตรวจสอบอีกครั้ง โดยอาจทำการลบหรือเพิ่มอินเด็กซ์เทอม

การจัดทำอินเด็กซ์อัตโนมัตินั้นจะมีประโยชน์ในการค้นคืนข้อมูล (Information Retrieval) ด้วย นั่นคือจะสามารถค้นหาเอกสารได้โดยการเช็คกับอินเด็กซ์เทอม



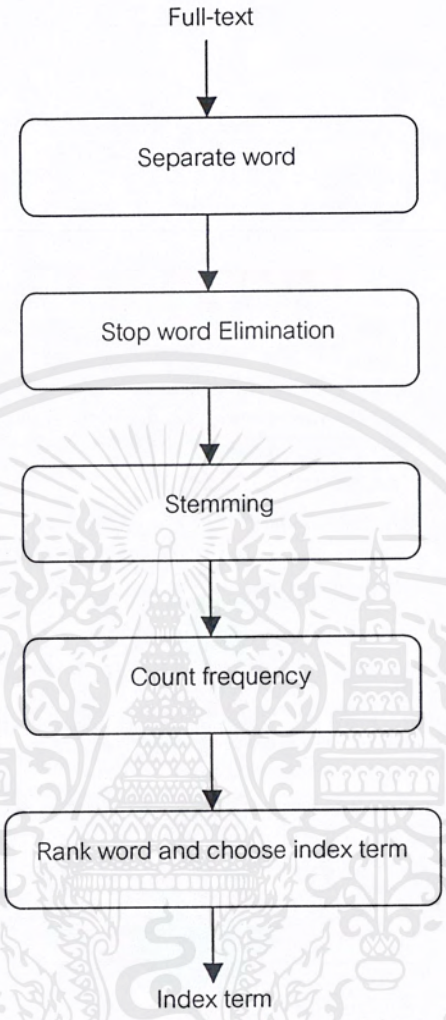
รูปที่ 3-5 การค้นคืนเอกสารจากอินเด็กซ์เทอม

จากแผนภาพข้างต้นจะเห็นได้ว่า จะมีการเก็บอินเด็กซ์ของแต่ละเอกสารเอาไว้ และเมื่อต้องการค้นคืนข้อมูลก็จะทำการคิวรี (Query) ข้อมูลออกมา โดยจะทำการเปรียบเทียบคีย์เวิร์ดที่ผู้ใช้ป้อนลงไปเพื่อค้นหาเอกสาร แล้วนำคีย์เวิร์ดนั้นไปเปรียบเทียบกับอินเด็กซ์เทอมที่มีอยู่ จากนั้นก็แสดงผลข้อมูลของเอกสารทั้งหมดที่มีอินเด็กซ์เทอมตรงกับคีย์เวิร์ดที่ผู้ใช้ป้อนลงไป

3.4 ขั้นตอนของการจัดทำอินเด็กซ์อัตโนมัติ

ในขั้นตอนของการจัดทำอินเด็กซ์อัตโนมัติของโครงงานนี้เราจะมองในแง่มุมที่ว่าอินพุทของระบบเป็นไฟล์ซึ่งเป็นข้อมูลทางวิชาการมีสกุลของไฟล์เป็น ".xml" ซึ่งเราจะนำไฟล์ข้อมูลทางวิชาการนี้มาจัดทำการหาอินเด็กซ์เทอมซึ่งจะออกมาเป็นเอาท์พุทของระบบ โดยจะต้องทำการพิจารณาทั้งในส่วนของ

Title, Abstract และ Description ของข้อมูลเท็กซ์ไฟล์ที่ต้องการหาอินเด็กซ์เทอม โดยจะต้องมีการให้ค่าถ่วงน้ำหนักในแต่ละส่วนต่างกัน ดังจะได้กล่าวต่อไป



รูปที่ 3-6 ขั้นตอนการจัดทำอินด็กซ์อัตโนมัติ

3.4.1 Separate Word

เมื่ออินพุทไฟล์ของระบบเป็นไฟล์สกุล ".xml" ดังนั้น จะมี tag ของข้อมูลต่างๆ ตามแต่ที่กำหนดเอาไว้ โดยโครงงานนี้จะทำการตั้งสมมุติฐานว่า รูปแบบของข้อมูลวิชาการโดยทั่วไปแล้วจะมีรูปแบบที่เป็นมาตรฐานโดยจะกำหนดให้มาตรฐานเป็นไปในรูปแบบดังต่อไปนี้

<information>

<title> </title>

<abstract> </abstract>

<description> </description>

</information>

ซึ่งถ้ามี tag อื่นที่นอกเหนือจากนี้ จะไม่พิจารณา เนื่องจากจะเลือกเฉพาะที่มีความสำคัญเท่านั้น จากนั้นจะทำการแบ่งแยกคำโดยจะต้องมีการตัดสัญลักษณ์หรืออักขระพิเศษออกให้เหลือแต่เฉพาะคำศัพท์เท่านั้น เช่น

I am a boy. จะได้คำว่า I am a boy ซึ่งคำว่า boy จะไม่มี (.) ลงท้าย

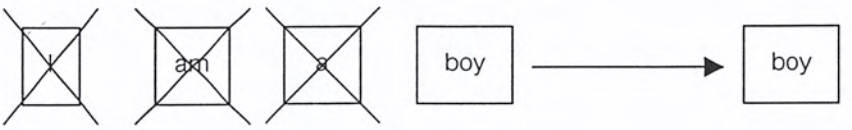
Are u a girl? จะได้คำว่า Are u a girl ซึ่งคำว่า girl จะไม่มี (?) ลงท้าย

หรือสัญลักษณ์อื่นๆ + - * / นอกเหนือจากนี้จะไม่พิจารณาตัวเลขด้วย คือถ้าพบตัวเลขจะไม่สนใจเก็บคำๆนั้นเอาไว้ จะตัดคำที่เป็นตัวเลขนั้นทิ้ง

3.4.2 Stop Word Elimination

ในขั้นตอนนี้จะทำการตัดคำที่ไม่จำเป็นออก ซึ่งได้กล่าวมาบ้างแล้วในหัวข้อของการจัดทำหัวข้อเรื่องอัตโนมัติ นั่นคือ จะทำการเปรียบเทียบคำที่ได้มาทั้งหมดว่าตรงกับคำใน Stop Word List หรือไม่ ถ้าตรงก็จะทำการตัดคำเหล่านั้นทิ้งไป ไม่นำมาพิจารณา เพราะคำเหล่านั้น ไม่ได้เป็นตัวบ่งชี้ที่สำคัญในการกล่าวถึงเอกสารใดๆ หรืออาจจะเป็นคำที่ไม่มีความหมายใดๆเลย

เช่น I am a boy. จะได้คำว่า I, am, a, boy จะทำการตัดคำ I, am, a ทิ้ง



เนื่องจากคำว่า I, am, a เป็นคำซึ่งอยู่ใน Stop Word List จึงทำการตัดคำเหล่านี้ออก

3.4.3 Stemming

คำศัพท์ต่างๆซึ่งมีการเปลี่ยนรูปเพื่อเปลี่ยนหน้าที่ของคำและนำคำนั้นๆไปใช้ในประโยค ซึ่งคำต่างๆเหล่านั้นยังคงเป็นคำที่มีความหมายเดียวกันอยู่ ดังนั้นจึงมีความจำเป็นอย่างยิ่งที่จะต้องทำกระบวนการในการเปลี่ยนคำศัพท์ต่างๆเหล่านั้นให้เป็นรากศัพท์ (Root Word) ของมัน โดยจะมีกฎของการเปลี่ยนรูปและการแปลงรูปต่างๆ ซึ่งต้องศึกษาจากโครงสร้างของคำและประโยคในภาษาอังกฤษโดยเฉพาะ เช่น

การเปลี่ยนรูปดังต่อไปนี้

a t i o n a l	–>	a t e	relational	->	relate
			conditional	->	condition
tional	->	tion	valency	->	valence
ency	->	ence	hesitancy	->	hsitance
ancy	->	ance	digitizer	->	digitize
izer	->	ize	conformably	->	conformable
ably	->	able	radically	->	radical
ally	->	al	differently	->	different
ently	->	ent	vilely	->	vile
ely	->	e	analogously	->	analogous
ously	->	ous	vietnamizaiton	->	vietnamize
ization	->	ize	predication	->	predicate
ation	->	ate	operator	->	operate
ator	->	ate	feudalism	->	feudal
alism	->	al			

การตัดคำท้ายทิ้ง

er	worker	-> work
ly	finally	-> final
ing	eating	-> eat
ed	walked	-> walk
en	eaten	-> eat
ee	employee	-> employ
ful	successful	-> success
s	eyes	-> eye

การเปลี่ยนรูปของคำศัพท์ต่างๆนั้นจะต้องเป็นไปตามกฎทางโครงสร้างประโยคในภาษาอังกฤษ ซึ่งการเปลี่ยนรูปของคำศัพท์ต่างๆนั้น เมื่อทำการเปลี่ยนรูปตามกฎเหล่านี้แล้ว ต้องทำการเช็คด้วยว่าคำที่ทำการเปลี่ยนรูปไปแล้วนั้นยังคงมีความหมายหรือไม่ โดยจะต้องทำการเช็คกับพจนานุกรมภาษาอังกฤษด้วย

พจนานุกรมภาษาอังกฤษที่กล่าวถึงนี้ ไม่ได้หมายถึงพจนานุกรมที่มีคำศัพท์ครบทุกคำทั้งหมด แต่จะทำการนำพจนานุกรมครบทุกคำนั้นมาทำการลดทอนลงให้เหมาะสมกับในการนำมาใช้ในโครงงานนี้ โดยจะทำการตัดคำที่ไม่ใช่รากศัพท์ออกทั้งหมด

เช่น ในพจนานุกรมเดิมมีคำว่า eat , eating, eaten จะต้องทำการเช็คว่าคำที่ลงท้าย ้วยตัวเสริมพิเศษต่างๆ (เช่น ing, ly, ed, en) ถ้าตัดออกแล้ว คำๆนั้นยังปรากฏอยู่ในพจนานุกรมหรือเปล่า ถ้าปรากฏก็ให้ทำการตัดคำที่ลงท้ายด้วยตัวเสริมพิเศษเหล่านั้นทิ้งเสีย อย่างในตัวอย่างเมื่อทำการตัดตัวเสริมพิเศษของคำว่า eating ออก จะได้คำว่า eat ก็นำมาเช็คว่ามีคำว่า eat ในพจนานุกรมหรือไม่ ปรากฏว่ามีก็จะทำการตัดคำว่า eating ออก เมื่อทำการตัดคำเหล่านี้ออกทั้งหมด สุดท้าย ก็จะเหลือแต่พจนานุกรมที่มีแต่รากศัพท์ของคำเท่านั้น

ในขั้นของการเปลี่ยนคำศัพท์ต่างๆให้เป็นรากศัพท์นั้นต้องทำการเช็คว่าคำนั้นเป็นรากศัพท์แล้วหรือยังกับพจนานุกรมภาษาอังกฤษ (ที่จัดเตรียมไว้ตั้งแต่ข้างต้น) ถ้าตรงกับรากศัพท์แล้ว ก็ไม่ต้องทำการแปลงรูปใดๆทั้งสิ้น แต่ถ้าไม่ตรงกับรากศัพท์แล้ว ก็ต้องทำการมาเช็คตามกฎต่างๆ

เช่น

feed เมื่อทดลองตัดคำท้าย ซึ่งลงท้ายด้วย ed ออก จะได้ fe เมื่อนำมาเช็คกับพจนานุกรมปรากฏว่า คำนี้ไม่มีความหมายทางพจนานุกรม เพราะฉะนั้นจะใช้กฎการตัดข้อนี้ไม่ได้ ต้องลองทำการใช้กฎอื่นต่อไป และต้องมาเช็คกับพจนานุกรมภาษาอังกฤษทุกครั้งว่ายังคงมีความหมายอยู่หรือไม่

กรณีที่คำศัพท์มีการเปลี่ยนรูปไปแล้วแต่ยังคงความหมายเดิมอยู่ กรณีส่วนมากจะเกิดกับคำศัพท์ที่เป็นกริยา เมื่อคำกริยาเหล่านั้นมีการผันรูปพิเศษเป็นกริยาช่อง 2 และกริยาช่อง 3 ซึ่งเปลี่ยนรูปไปเลยจะไม่ตรงตามกฎข้างต้น ดังนั้นจะต้องมีข้อมูลอีกชุดหนึ่งซึ่งเป็นการเช็คในกรณีสุดท้ายว่าถ้าไม่ตรงตามกฎข้างต้นแล้วมีการเปลี่ยนรูปไปหรือไม่ โดยข้อมูลชุดสุดท้ายนี้จะมีการเก็บคำศัพท์ที่เป็นรากศัพท์และคำศัพท์ที่ผันรูปไปแล้ว

เช่น

กริยาช่อง 1	กริยาช่อง 2	กริยาช่อง 3
eat	ate	eaten
bring	brought	brought

เมื่อมีคำว่า ate เข้ามาแล้วทดสอบตามกฎข้างต้นแล้วไม่สามารถแปลงเป็นรากศัพท์ได้ ก็ทำการเช็คกับข้อมูลที่มีอยู่ชุดนี้ ว่าเป็นคำศัพท์แปลงรูปของรากศัพท์ตัวใดหรือไม่ ถ้าเป็นก็ทำการส่งค่ารากศัพท์คืนมาได้

3.4.4 Count Frequency and Tag Weight

หลังจากขั้นตอนของการแปลงคำศัพท์เป็นรากศัพท์แล้วนั้น ก็ทำการนับความถี่ของคำแต่ละคำว่ามีค่าเป็นเท่าไหร่ และทำการคูณด้วยค่าถ่วงน้ำหนักใน tag นั้นๆด้วย โดยการกำหนดค่าถ่วงน้ำหนักนั้น จะกำหนดให้ค่าถ่วงน้ำหนักใน tag ของ Title มากที่สุด เพราะ Title จะสามารถสื่อถึงความหมายของเอกสารได้ดีที่สุด รองลงมาก็จะเป็น tag ของ Abstract ซึ่งเป็น tag ที่สื่อถึงบทคัดย่อเป็นบทสรุปโดยรวมของเนื้อหาของเอกสารนั้นๆ และ tag ที่ให้ค่าถ่วงน้ำหนักน้อยที่สุดคือ tag ของ description ซึ่งจะเป็นการบรรยายเนื้อหาของเอกสารนั้นๆแล้ว

เช่น

```
<information>
    <title>  eat  swim  eat  </title>
    <abstract>      eat      </abstract>
    <description>   eat      </description>
</information>
```

จะได้ว่า eat ใน tag title มีความถี่ 2 คำ จะได้ 2 x weight ของ title

eat ใน tag abstract มีความถี่ 1 คำ จะได้ 1 x weight ของ abstract

eat ใน tag description มีความถี่ 1 คำ จะได้ 1 x weight ของ description

ผลรวมของค่าของคำว่า eat =

(2 x weight ของ title) + (1 x weight ของ abstract) + (1 x weight ของ description)

3.4.5 Rank Word and Choose Index Term

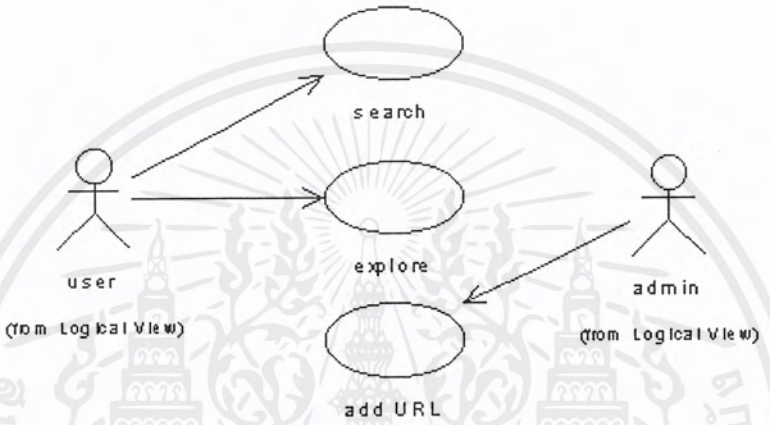
เมื่อมีการคำนวณหาค่าของแต่ละคำศัพท์ได้แล้วก็มาทำการจัดลำดับของค่าเหล่านั้น โดยทำการเรียงลำดับค่าของคำศัพท์ที่มีค่ามากที่สุดไปยังคำศัพท์ที่มีค่าน้อยที่สุด ในที่นี้จะพิจารณาเลือกคำศัพท์ที่มีค่าสูงสุด 20 อันดับแรกมาเป็นอินเด็กซ์เทอม ซึ่งใช้เป็นตัวแทนของเอกสารนั้นๆ

บทที่ 4 การสร้างและการวิเคราะห์ออกแบบระบบ (UML Model)

4.1 การวิเคราะห์และออกแบบระบบโดยใช้ UML

การวิเคราะห์และการออกแบบระบบ จะเป็นการมองการทำงานโดยรวมของระบบทั้งหมดว่ามีการทำงานเป็นอย่างไรบ้าง โดยจะมีการวิเคราะห์ตามรูปแบบดังต่อไปนี้

4.1.1 USE CASE



รูปที่ 4-1 แสดง Use Case ของระบบ

Use Case จะเป็นโมเดลที่อธิบายฟังก์ชันการทำงานหลักของระบบ โดยจากรูปจะเห็นว่าระบบของเราสามารถให้ผู้ใช้เข้ามาทำการค้นหาหาข้อมูลที่ต้องการได้ นอกจากนี้ผู้ใช้ยังสามารถเข้ามา explore เพื่อดูว่าในหมวดหมู่ต่างๆมีเอกสารใดอยู่บ้าง นอกจากผู้ใช้แล้วยังมี actor อีกประเภทคือ administrator คือ ผู้ดูแลระบบ จะมีการทำงานหลักคือการเพิ่มแอดเดรสเข้าไปในฐานข้อมูล โดยผู้ดูแลระบบจะทำการเพิ่มแอดเดรส โดยสั่งให้เว็บโรบอทเป็นตัวนำข้อมูลมาอีกทีหนึ่ง

4.1.2 SCENARIOS

Scenario เป็นตัวตนของ Use Case เป็นโมเดลที่กำหนดลำดับของแมสเซจของระบบรวมถึงโครงสร้างของวัตถุซึ่งทำงานร่วมกันเพื่อให้ระบบสามารถทำหน้าที่รับผิดชอบได้ ช่วยให้สามารถทำความเข้าใจกับระบบได้ง่ายขึ้น โดยในโครงงานนี้จะนำเสนอ Scenarios ในรูปแบบของ Sequence Diagram

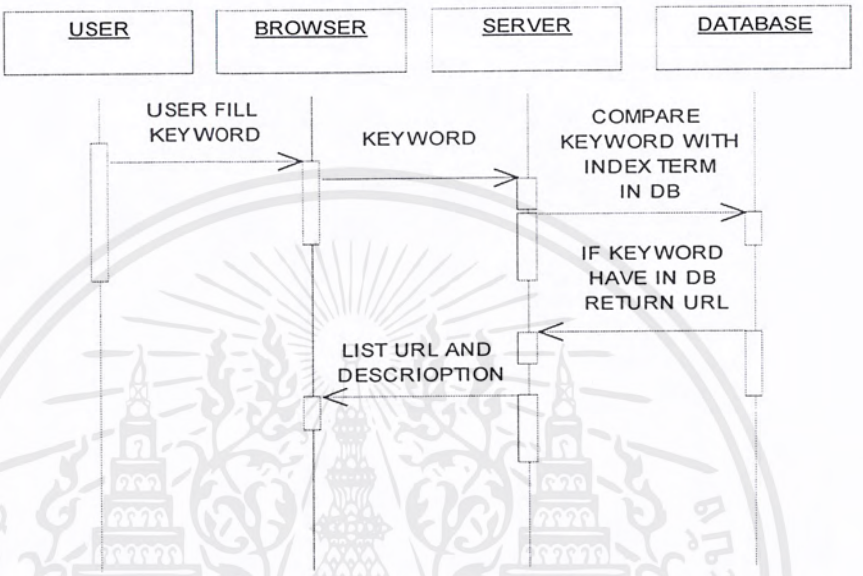
โดยจะแบ่งเป็น Scenario สำหรับการ search, explore และ add URL

- **Search**

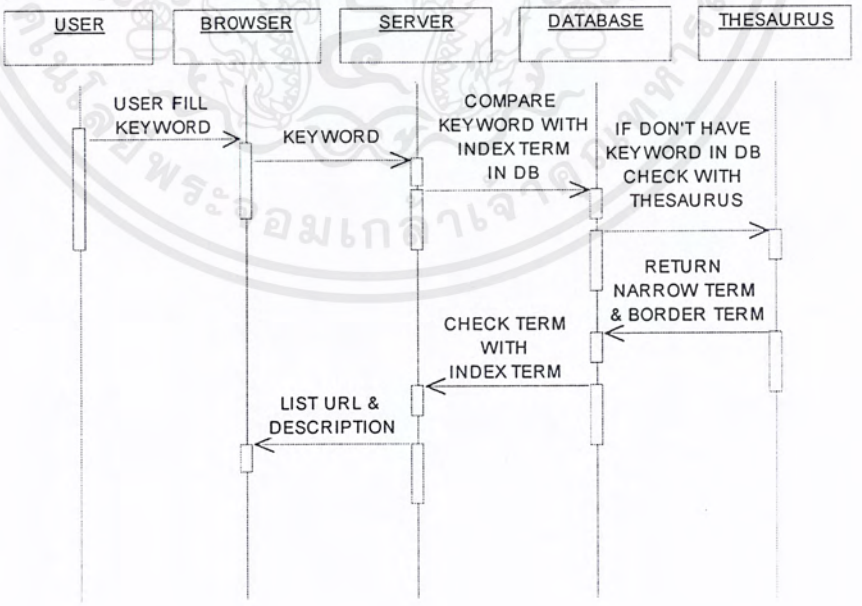
สำหรับ Scenario ในการค้นหาข้อมูลนั้นได้ออกแบบโดยแบ่งเป็น 2 Scenarios ย่อย

โดยใน Scenario แรกจะกล่าวถึงในกรณีที่คีย์เวิร์ดที่ผู้ใช้ป้อนให้แก่ระบบนั้นตรงกับอินเด็กซ์ ในฐานข้อมูลของระบบ ระบบก็จะสามารถแสดงแอดเดรสของเอกสารที่เกี่ยวข้องได้ทันที

ส่วนใน Scenario ที่สองนั้นได้กล่าวถึงในกรณีที่คีย์เวิร์ดที่ผู้ใช้ป้อนไม่ตรงกับอินเด็กซ์เทอมที่มีในฐานข้อมูล ระบบก็จะต้องใช้ Thesaurus เป็นตัวช่วยเพื่อที่จะได้สามารถหาข้อมูลอื่นที่มีความใกล้เคียงกับข้อมูลที่ผู้ใช้ต้องการได้

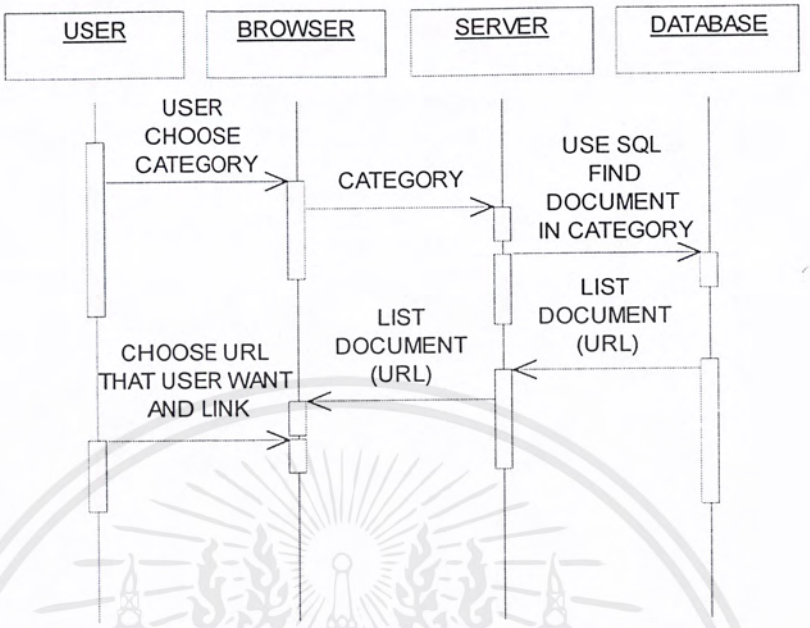


รูปที่ 4-2 Scenario สำหรับการค้นหา ในกรณีแรก (คีย์เวิร์ดตรงกับอินเด็กซ์เทอม)



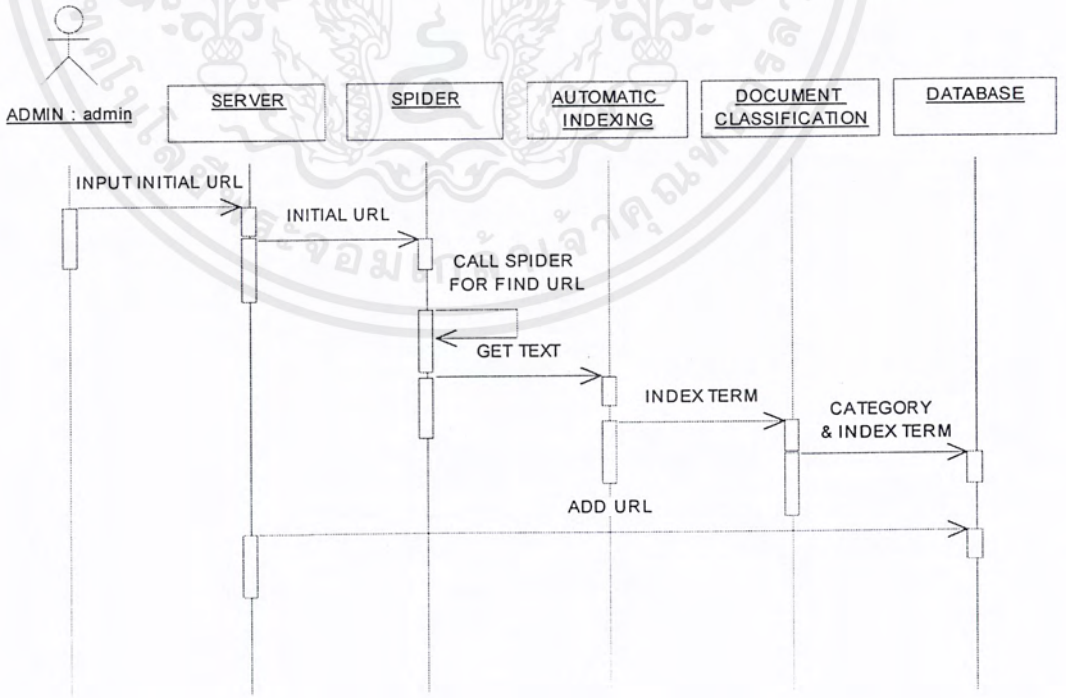
รูปที่ 4-3 Scenario สำหรับการค้นหา ในกรณีที่สอง (คีย์เวิร์ดไม่ตรงกับอินเด็กซ์เทอม)

- **Explore** Scenario ในการ explore ใช้เพื่อให้ผู้ใช้สามารถเข้ามาดูว่าแต่ละหมวดหมู่ประกอบไปด้วยข้อมูลอะไรบ้างที่น่าสนใจ



รูปที่ 4-4 แสดงการ explore ดูข้อมูลในแต่ละประเภท

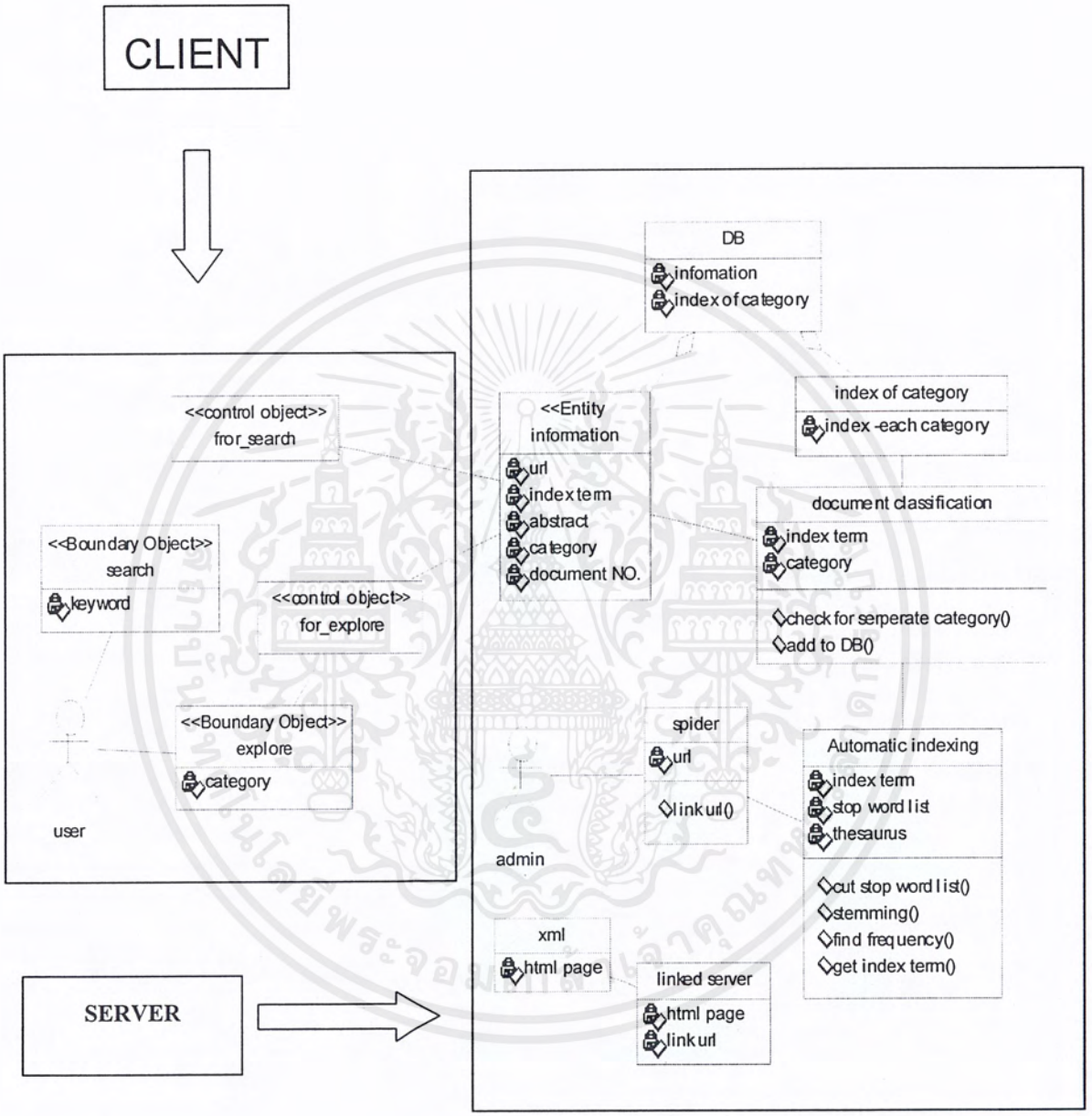
- **Add URL** เป็น Scenario ที่อธิบายลำดับการทำงานเมื่อผู้ดูแลระบบจะทำการเพิ่มแอดเดรสโดยสั่งให้สไปเดอร์ไปนำข้อมูลต่างๆมา



รูปที่ 4-5 แสดงการเพิ่มเว็บไซต์โดยสั่งรันสไปเดอร์และหาอินเด็กซ์เทอมของเอกสาร

4.1.3 CLASS DIAGRAM

เป็นโมเดลที่แสดงโครงสร้างของวัตถุและคลาสที่มีในระบบรวมทั้งแสดงความสัมพันธ์ด้วย จะใช้ในการแสดงโครงสร้างหลักของระบบ

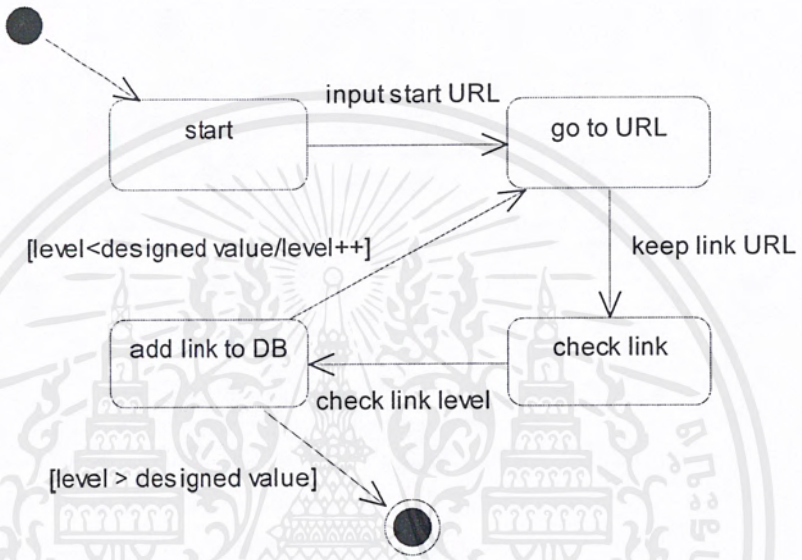


รูปที่ 4-6 แสดงคลาสไดอะแกรมของระบบ

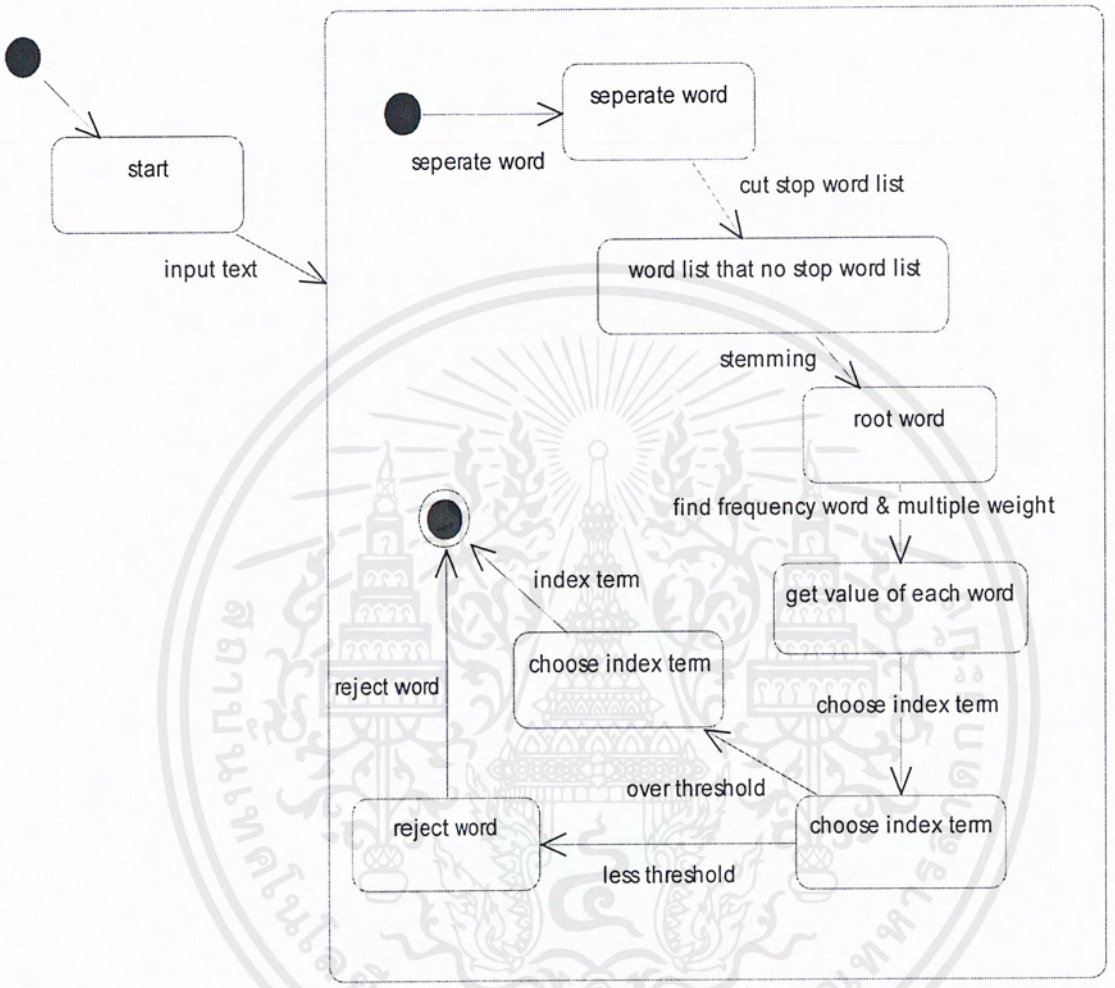
4.1.4 STATE DIAGRAM

เราจะกำหนดให้แต่ละคลาสมีแผนภาพสถานะ 1 แผนภาพ ซึ่งจะแสดงถึงสถานะต่างๆในแต่ละส่วนของระบบ

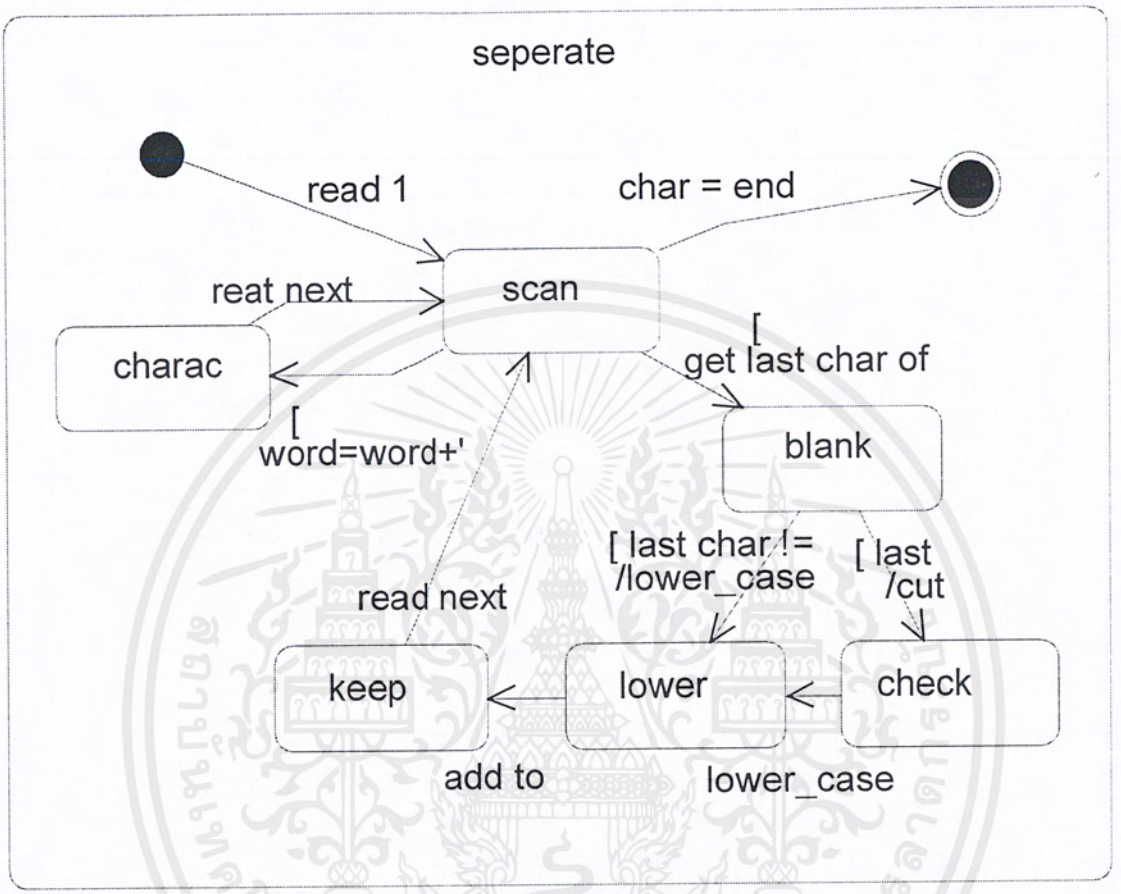
State Diagram จะเป็นการอธิบายถึงกระบวนการและขั้นตอนอย่างละเอียดในการทำงานแต่ละสเต็ป ซึ่งจะทำให้เห็นภาพได้ชัดเจนถึงแต่ละสเตทจากจุดเริ่มต้นไปยังจุดจบของสเตทไดอะแกรม



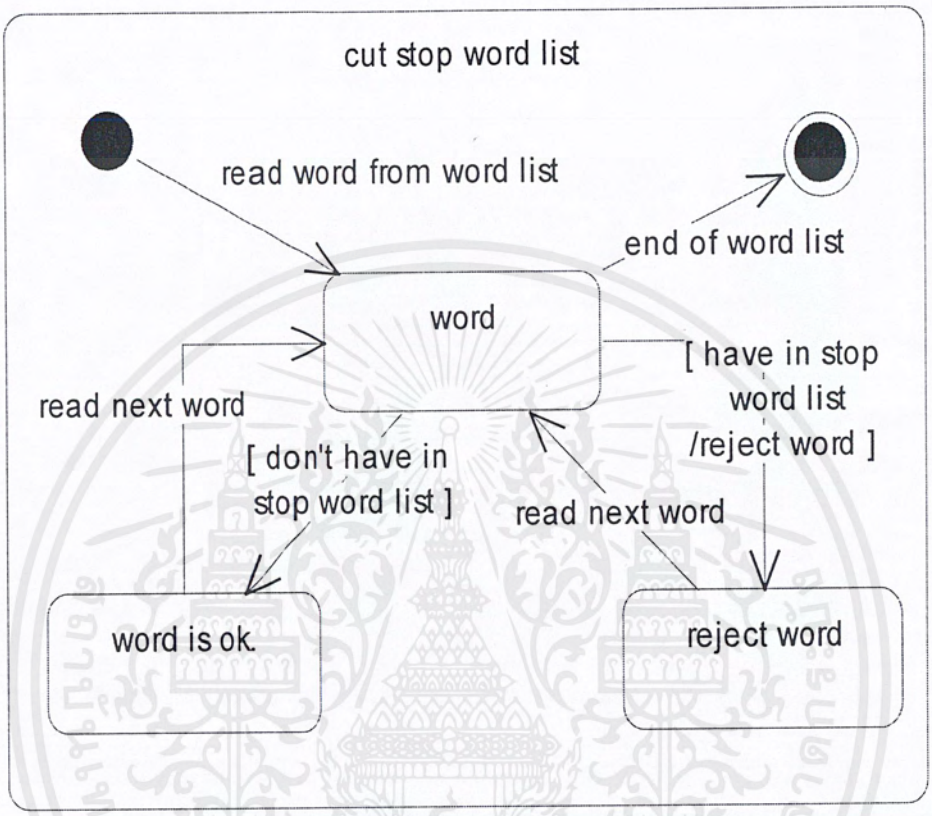
รูปที่ 4-7 แสดงสเตทไดอะแกรมของสไปเดอร์



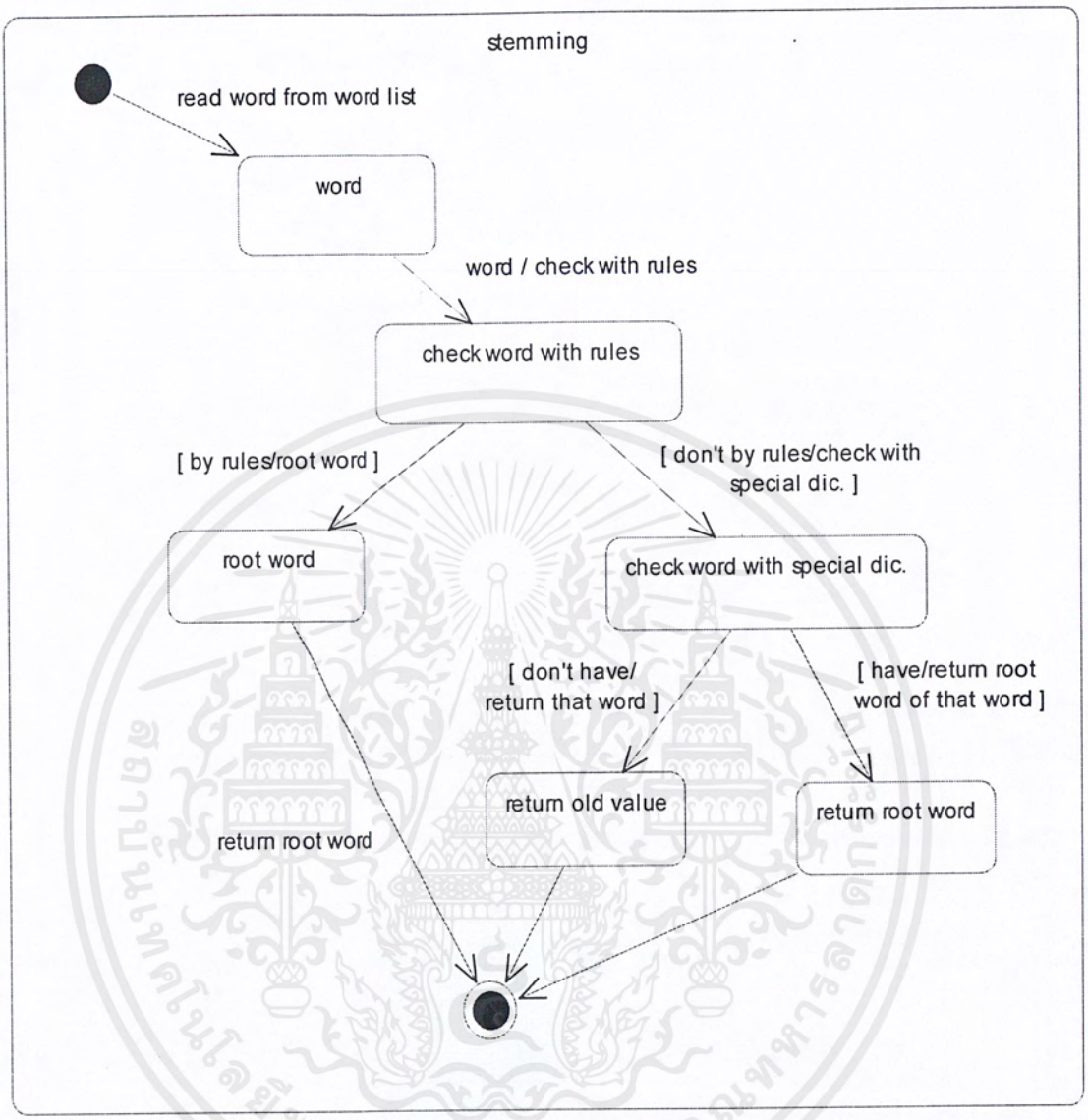
รูปที่ 4-8 แสดงสเตทไดอะแกรมของการทำอินเด็กซ์อัตโนมัติ



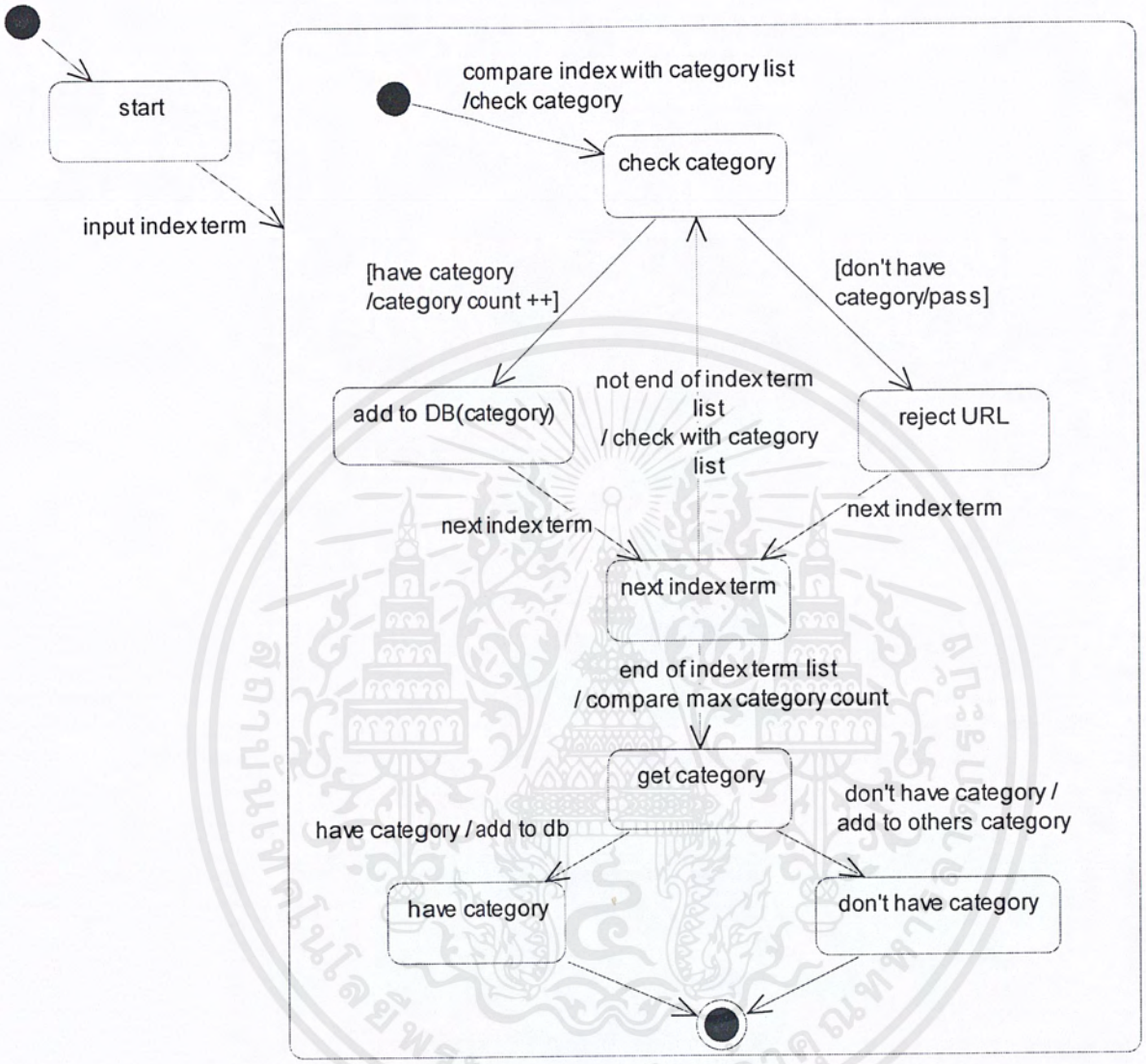
รูปที่ 4-9 แสดงสเตทไดอะแกรมในการแยกคำ (สเตทย่อยของการทำอินเด็กซ์อัตโนมัติ)



รูปที่ 4-10 แสดงสเตทไดอะแกรมของการตัด stopr word list ออก (สเตทย่อยของการทำอินเด็กซ์อัตโนมัติ)



รูปที่ 4-11 แสดงการทำ stemming (สเตทย่อยของการทำอินเด็กซ์อัตโนมัติ)

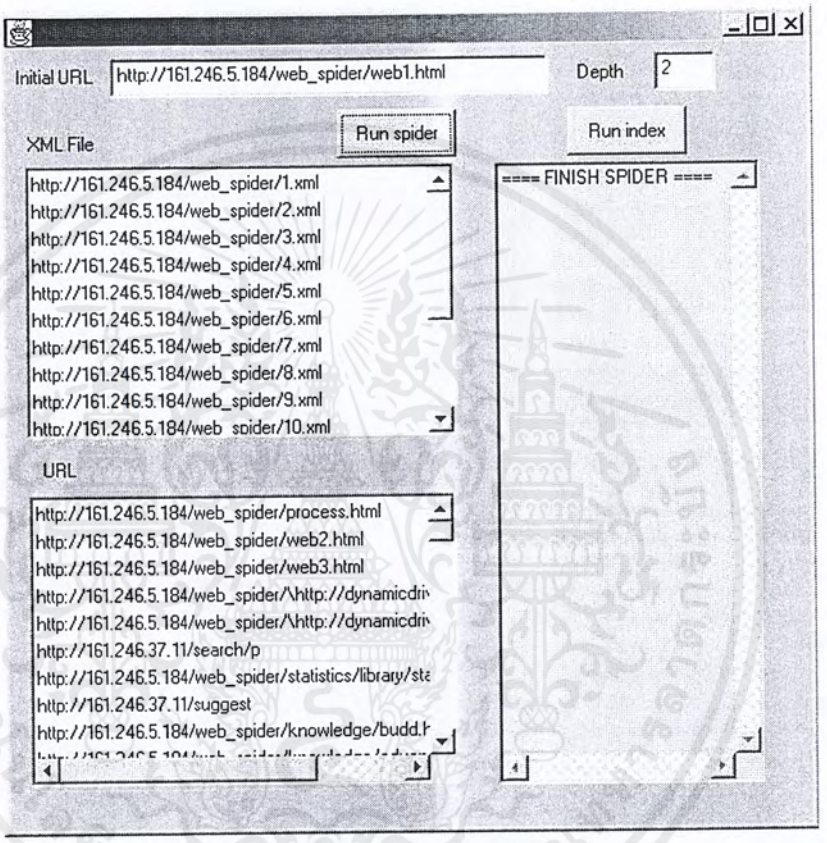


รูปที่ 4-12 แสดงสเตทไดอะแกรมของการแยกประเภทเอกสาร

4.2 Detail Design

4.2.1 สไปเดอร์

เป็นส่วนของการดึงข้อมูล ซึ่งเป็นเท็กซ์ไฟล์สกุล ".xml" โดยจะเริ่มต้นจากการที่ผู้ดูแลระบบ จะมีการป้อนอินพุท 2 ค่า คือ ค่าแอดเดรสเริ่มต้น (Initial URL) และค่าความลึก (Depth) ของการลิงค์ไปเว็บไซค์ต่างๆ



รูปที่ 4-13 แสดงหน้าจอสำหรับผู้ดูแลระบบในการรันสไปเดอร์และหาอินเด็กซ์ของ

เมื่อผู้ใช้ระบบทำการป้อนค่าทั้ง 2 ค่าครบแล้ว เมื่อสั่งรันโปรแกรมก็จะเข้าไปทำการพาร์เอกสารและตรวจจับแท็กที่มีการลิงค์ต่อไป โดยจะต้องทำการเช็คก่อนว่าลิงค์เหล่านั้นเป็นการลิงค์ไปยังไฟล์ xml หรือลิงค์ไปยังหน้าจอเอชทีเอ็มแอลถัดไป ถ้าเป็นการลิงค์ไปยังไฟล์ xml ก็ทำการเซฟไฟล์นั้นเก็บเอาไว้ แต่ถ้าเป็นลิงค์ไปยังหน้าจอเอชทีเอ็มแอลก็ให้ทำการลิงค์เหล่านั้นเอาไว้ทังหมดก่อนแล้วจึงนำมาเช็คลิงค์ถัดไปอีก โดยจะทำวนไปเป็นจำนวนเท่ากับค่าความลึกที่ผู้ดูแลระบบทำการป้อนอินพุทเข้ามา

4.2.2 การจัดทำอินเด็กซ์อัตโนมัติ

หลังจากที่ทำการสั่งรันตัวสไปเดอร์แล้ว จะได้ไฟล์สกุล ".xml" มาเก็บไว้ และจะนำไฟล์สกุล ".xml" แต่ละไฟล์มาผ่านกระบวนการการจัดทำอินเด็กซ์ โดยจะผ่านกระบวนการดังต่อไปนี้

ขั้นตอนที่ 1 : Parser นั่นคือจะนำไฟล์ xml นั้น มาผ่านพาร์เซอร์เพื่อทำการเช็ค syntax ว่าถูกต้องตรงตามข้อกำหนดและรูปแบบของ xml หรือไม่

ขั้นตอนที่ 2 : ทำการดึงข้อมูลในแต่ละแท็กข้อมูลออกมา ซึ่งในโครงงานนี้จะพิจารณา 3 แท็ก ข้อมูลที่จำเป็นเท่านั้น สำหรับโครงงานนี้จะกำหนดรูปแบบ DTD ขึ้นมาดังต่อไปนี้

```
<?xml version="1.0"?>
<!DOCTYPE INFORMATION [
        <!ELEMENT INFORMATION (TITLE, ABSTRACT, DESCRIPTION)>
        <!ELEMENT TITLE (#PCDATA)>
        <!ELEMENT ABSTRACT (#PCDATA)>
        <!ELEMENT DESCRIPTION (#PCDATA)>
]>
<INFORMATION>
        <TITLE>          </TITLE>
        <ABSTRACT>       </ABSTRACT>
        <DESCRIPTION>    </DESCRIPTION>
</INFORMATION>
```

เมื่อทำการดึงข้อมูลจากแต่ละแท็กมาแล้วก็ทำตามกระบวนการการจัดทำอินเด็กซ์อัตโนมัติตามที่กล่าวไว้แล้วในบทที่ 3 หัวข้อของการจัดทำอินเด็กซ์อัตโนมัติ โดยจะมีการกำหนดค่าถ่วงน้ำหนักให้กับแต่ละแท็กด้วย จากนั้นก็จะได้ลิสต์ของคำและค่าที่คำนวณได้ซึ่งบ่งบอกถึงความสำคัญของคำนั้นมา แล้วทำการเลือกกลุ่มคำที่มีค่าสูงสุดมาเป็นอินเด็กซ์เทอม

4.2.3 การแบ่งแยกประเภทของเอกสาร

เมื่อผ่านขั้นตอนของการจัดทำอินเด็กซ์โดยอัตโนมัติแล้ว ก็จะได้อินเด็กซ์เทอมของแต่ละเอกสารออกมา จากนั้น จะทำการเปรียบเทียบว่าอินเด็กซ์เทอมเหล่านั้น เป็นเอกสารประเภทใด ซึ่งถ้าเป็นในระบบห้องสมุดจริงก็จะเปรียบเหมือนกับว่าเราหาเลขหมู่ของหนังสือมาได้แล้ว และกำลังที่จะเอาหนังสือเหล่านั้นเก็บใส่ชั้นหนังสือให้ถูกต้องตามล็อคของหนังสือนั้นๆ

สำหรับโครงงานชิ้นนี้ได้แบ่งประเภทของเอกสารเอาไว้ 9 ประเภท ได้แก่

1. Agent
2. Artificial Intelligent
3. Database

4. Information Retrieval
5. Network
6. Operating System
7. Programming
8. Security
9. Software Engineering

ในการเปรียบเทียบว่าเอกสารนั้นจัดอยู่ในประเภท (category) ใด ต้องอาศัยข้อมูลชุดหนึ่ง (Category List) ซึ่งจะมีการจัดเก็บอินเด็กซ์ใดๆว่าควรจะจัดเก็บเป็นเอกสารประเภทไหน จากนั้นก็นำอินเด็กซ์ที่ได้จากการทำอินเด็กซ์อัตโนมัติมาเปรียบเทียบดูว่า คำเหล่านั้นตรงกับประเภทของเอกสารชนิดใด แล้วก็นำเอกสารนั้นไปจัดเก็บไว้ในประเภทนั้น

ข้อมูลที่ใช้ในการแบ่งประเภทของเอกสารนั้น จะเรียกว่า Category List โดยจะสามารถทำได้ดังตัวอย่างต่อไปนี้

TERM	CATEGORY
intelligent	AI
expert	AI
thread	OS
operating	OS

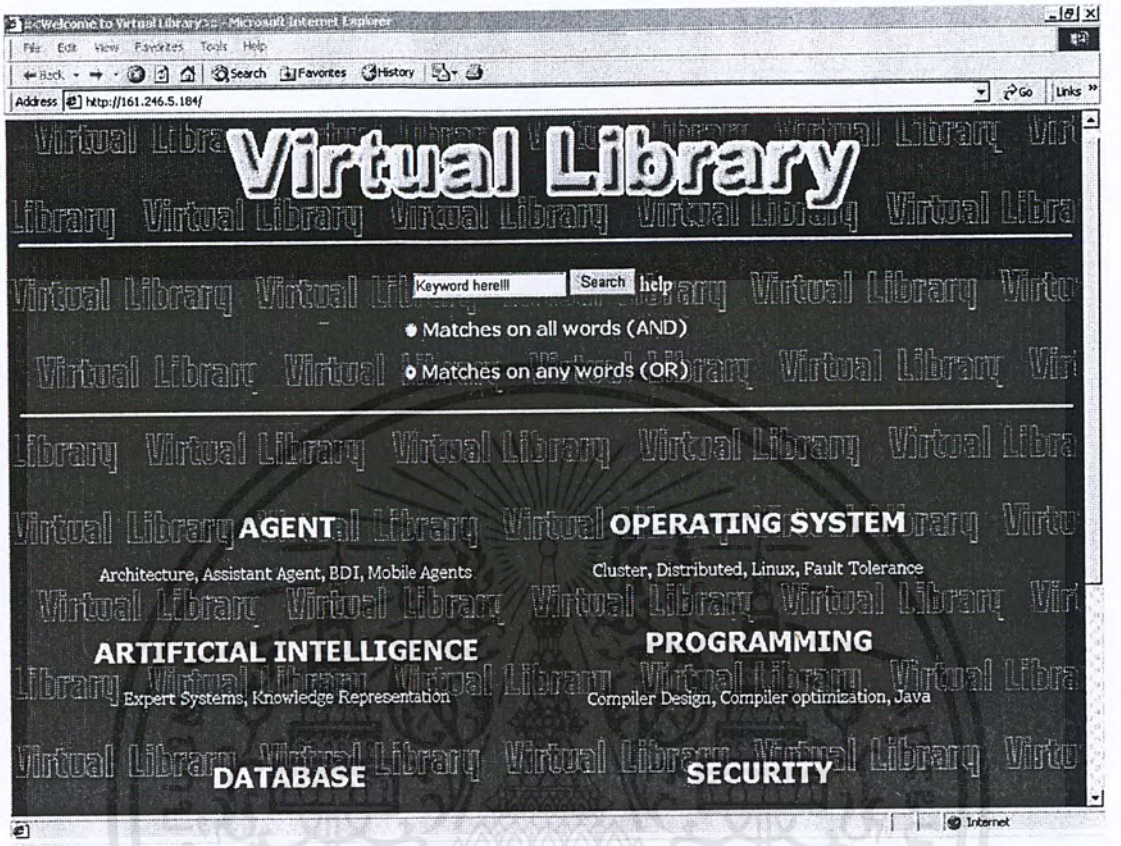
เมื่อได้อินเด็กซ์เทอมมาแล้วก็ทำการเปรียบเทียบกับคำใน Category List ว่าตรงกับเอกสารประเภทใด เช่น ถ้าอินเด็กซ์เทอมที่หามาได้ มีคำว่า intelligent หรือ expert ก็จะจัดอยู่ในประเภทของเอกสารแบบ Artificial Intelligent

สำหรับ Category List นี้จะต้องทำการลิสต์เองด้วยมนุษย์ นั่นคือจะต้องทำการศึกษาและค้นคว้ารวบรวมข้อมูลว่าเอกสารในแต่ละประเภทนั้น ควรจะต้องมีอินเด็กซ์เทอมใดบ้าง ซึ่งขึ้นอยู่กับประสบการณ์และการค้นคว้าหาข้อมูลว่าจะสามารถรวบรวมข้อมูลได้มากน้อยเพียงใด

การจัดเก็บข้อมูลเมื่อผ่านขั้นตอนของการจัดทำอินเด็กซ์อัตโนมัติและการแบ่งแยกประเภทของเอกสารจะต้องมีการจัดเก็บอินเด็กซ์เทอมของเอกสาร, ประเภทของเอกสาร รวมถึง URL, Title และ Abstract เพื่อใช้เป็นข้อมูลในการให้บริการในห้องสมุดเสมือน สำหรับผู้ใช้ในการค้นหาข้อมูลและประเภทเอกสารต่างๆตามที่ผู้ใช้ต้องการได้

4.2.4 ส่วนของการติดต่อกับผู้ใช้

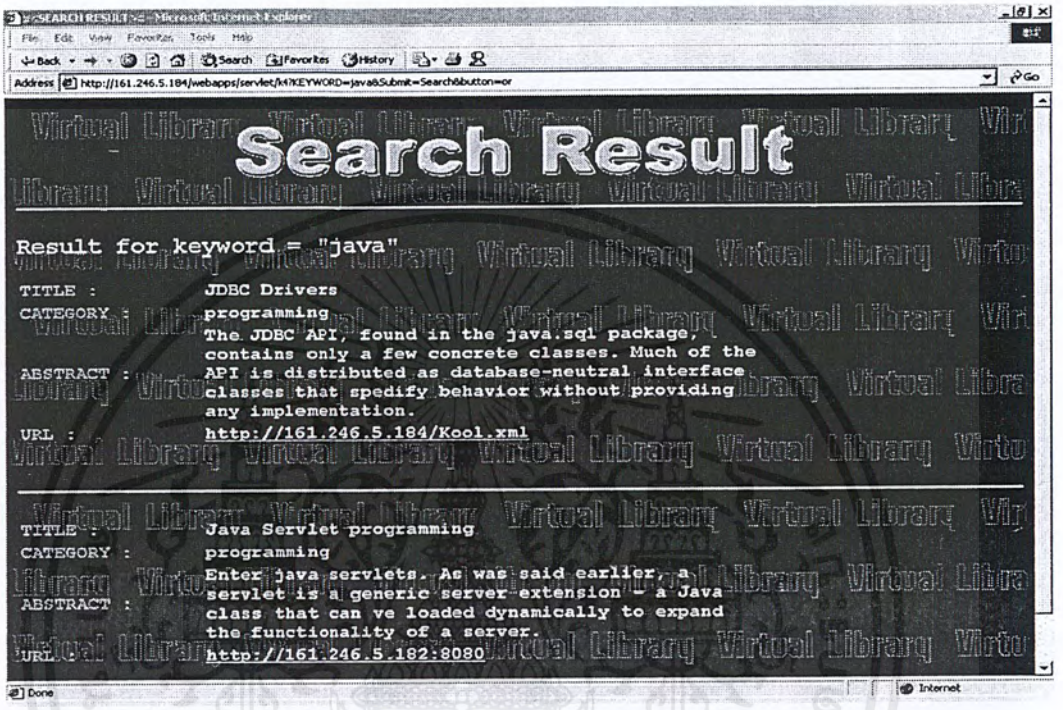
จะเป็นการให้บริการกับผู้ใช้ในการสืบค้นข้อมูลซึ่งจะสามารถสืบค้นข้อมูลได้ 2 แบบ นั่นคือ



รูปที่ 4-14 แสดงหน้าจอของเวบไซต์ห้องสมุดเสมือน

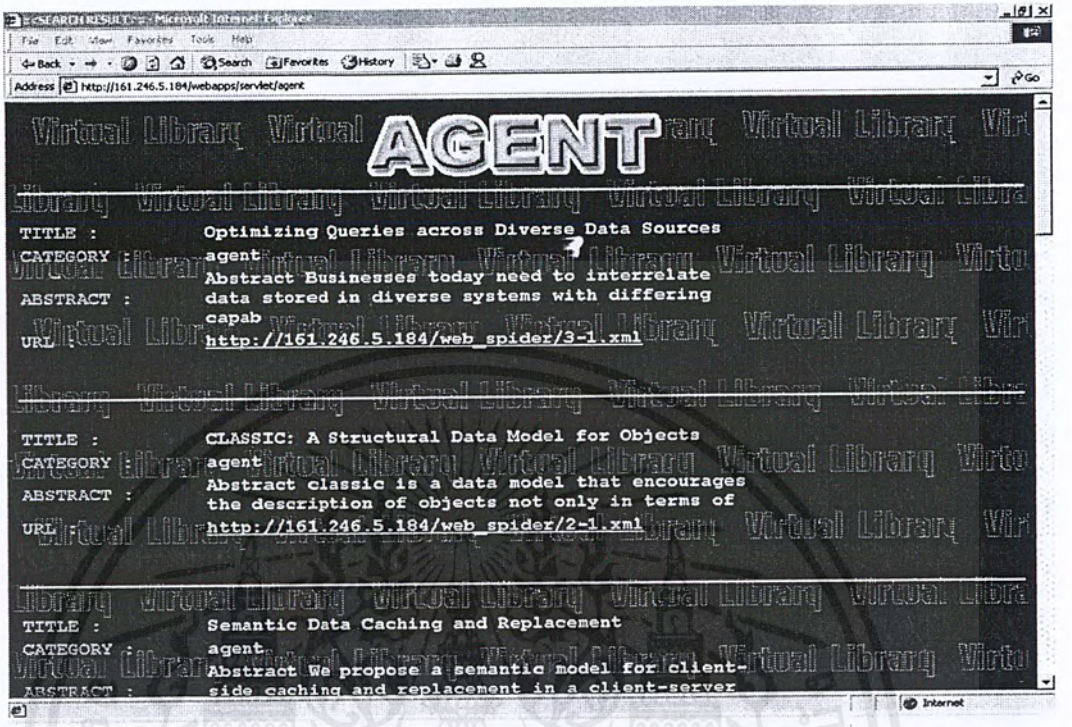
- การสืบค้นโดยทำการป้อนคีย์เวิร์ดลงไปแล้ว จากนั้นระบบจะทำการค้นหาข้อมูลหรือเอกสารทั้งหมดที่เกี่ยวข้องกับคีย์เวิร์ดนั้น ถ้าไม่พบ ก็จะยังสามารถที่จะหาคำที่มีความหมายใกล้เคียงกับคีย์เวิร์ดนั้น โดยทำการเช็คกับ Thesaurus แล้วทำการแสดงเอกสารที่เกี่ยวข้องกับคำเหล่านั้นทั้งหมดมาให้ผู้ใช้

จากตัวอย่างข้างล่างเป็นตัวอย่างผลลัพธ์ในการค้นหาเอกสารโดยป้อนคีย์เวิร์ดคำว่า java เมื่อทำการค้นหาเอกสารก็จะได้ผลลัพธ์ดังรูปข้างล่างนี้ ซึ่งจะแสดง title, category, abstract, URL



รูปที่ 4-15 แสดงผลลัพธ์ของหน้าจอในการค้นหาข้อมูลจากคีย์เวิร์ด

- การสืบค้นโดนการเข้าไปดูในแต่ละประเภทของเอกสารเลย โดยผู้ใช้สามารถจะคลิ๊กเข้าไปดูในแต่ละประเภทที่ทางเว็บไซต์ได้จัดเตรียมแบ่งประเภทเอาไว้ 10 ประเภทแล้ว ก็จะสามารถดูได้ว่าในแต่ละประเภทนั้น มีเอกสารใดที่น่าสนใจบ้าง

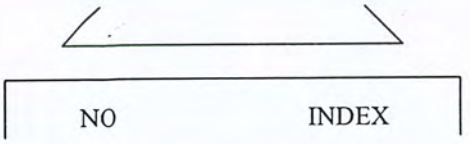


รูปที่ 4-16 แสดงผลลัพธ์ในการค้นหาข้อมูลจากประเภทของข้อมูล

4.3 ฐานข้อมูลที่จำเป็นต้องใช้

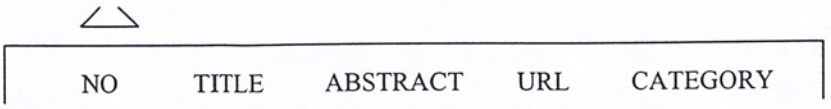
4.3.1 ตารางในการจัดเก็บอินเด็กซ์

สำหรับจัดเก็บหมายเลขของเอกสาร และอินเด็กซ์เทอมของเอกสารนั้นใช้สำหรับ ค้นหาว่าคีย์เวิร์ดที่ผู้ใช้ป้อนเข้ามาจะตรงกับคำอินเด็กซ์เทอมใดเพื่อที่จะทำการแสดงผลของเอกสารที่ตรงกับคีย์เวิร์ดที่ผู้ใช้ต้องการ



4.3.2 ตารางในการจัดเก็บรายละเอียดของเอกสาร

สำหรับจัดเก็บหมายเลขของเอกสาร หัวข้อเรื่อง บทคัดย่อ (เพียงส่วนหนึ่งเท่านั้นไม่ใช่ทั้งหมด) แอดเดรสของเอกสารนั้น และประเภทของเอกสาร

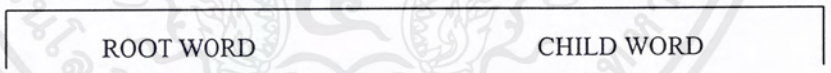


4.3.3 Thesaurus

สำหรับใช้ช่วยเพิ่มประสิทธิภาพในการค้นหาคำหรือข้อมูลต่างๆ โดยในกรณีที่มีกาาค้นหาคำใดๆแล้วไม่ตรงกับในอินเด็กซ์เทอมของเอกสารใดๆเลย ก็จะมีการนำคำๆนั้น มาหา Broader Term เมื่อได้ Broader Term แล้วจึงนำ Broader Term ไปเทียบกับอินเด็กซ์ของเอกสารอีกครั้งหนึ่ง ซึ่งจะทำให้ผู้ใช้สามารถพบข้อมูลที่มีความหมายใกล้เคียงหรือความหมายที่กว้างขึ้น

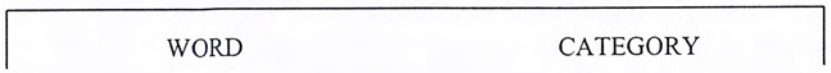
4.3.4 พจนานุกรมรากศัพท์พิเศษ

สำหรับคำศัพท์ที่มีการเปลี่ยนรูปแบบของคำโดยไม่เป็นไปตามกฎทั่วไป ซึ่งเมื่อทำการกระบวนการ Stemming โดยเช็คตามกฎแล้วไม่ตรงไปตามกฎ ก็จะทำการเช็คกับพจนานุรมรากศัพท์พิเศษเพื่อดูว่ามีการเปลี่ยนรูปของคำหรือไม่



4.3.5 Category List

เป็นการรวบรวมคำที่จะสามารถบ่งชี้ในการแบ่งแยกประเภทของเอกสารได้ โดยจะนำอินเด็กซ์เทอมที่ได้มาเช็คกับคำใน Category List นี้ เพื่อแบ่งแยกประเภท



บทที่ 5 การทดสอบระบบและผลการทดลอง

5.1 การทดสอบหาค่าถ่วงน้ำหนัก (weight)

ค่าถ่วงน้ำหนักที่เหมาะสมในแต่ละแท็กของข้อมูลนั้น ได้จากการทดลองเลือกค่าถ่วงน้ำหนักหลายๆชุดข้อมูลแล้วทำการทดสอบดูว่าชุดของค่าเหล่านั้น ชุดใดที่ให้ชุดของคำอินเด็กซ์ออกมาแล้วมีความหมายตรงกับประเภทของข้อมูลมากที่สุด

โดยทำการเตรียมไฟล์สกุล xml เอาไว้ทั้งหมด 9 ประเภทของข้อมูล แต่ละประเภทประเภทละ 10 ไฟล์ รวมทั้งหมด 90 ไฟล์ จากการทดสอบเบื้องต้นพบว่าค่าถ่วงน้ำหนักนั้น ควรจะต้องให้ค่าถ่วงน้ำหนักของแต่ละแท็กเป็นดังนี้ คือ ค่าถ่วงน้ำหนักของหัวข้อเรื่อง > ค่าถ่วงน้ำหนักของบทคัดย่อ > ค่าถ่วงน้ำหนักของเนื้อหา เมื่อมีการให้ค่าถ่วงน้ำหนักของหัวข้อเรื่อง /บทคัดย่อ /เนื้อหา

ประเภทของเอกสาร	% ความถูกต้องในการแบ่งประเภทของข้อมูล 5/2/1	% ความถูกต้องในการแบ่งประเภทของข้อมูล 10/8/1	% ความถูกต้องในการแบ่งประเภทของข้อมูล 25/10/1
AGENT	50	80	80
ARTIFICIAL INTELLIGENT	40	40	50
DATABASE	60	90	90
INFORMATION RETRIEVAL	20	50	50
NETWORK	30	40	40
OPERATING SYSTEM	40	50	50
PROGRAMMING	40	40	60
SECURITY	50	60	50
SOFTWARE ENGINEERING	20	40	40
เฉลี่ยแปดร์เซ็นต์	๓๙.๘๘	๔๑	๕๕.๕๕

ตารางท 5-1 แสดงผลการจดแบงประเภทเอกสารโดยใชคาถวงนาหนกตางๆ

จากผลการทดลองจะพบว่า ถ้าพิจารณาเฉพาะการจัดหมวดหมู่ของประเภทของเอกสารนั้น ถ้ากำหนดค่าถ่วงน้ำหนักให้ ค่าถ่วงน้ำหนักของหัวข้อเรื่อง > ค่าถ่วงน้ำหนักของบทคัดย่อ > ค่าถ่วงน้ำหนักของเนื้อหา ก็จะสามารถจัดหมวดหมู่ของประเภทของเอกสารได้อย่างถูกต้อง หรืออีกนัยหนึ่ง นั่นคือ อินเด็กซ์เทอมที่ได้ออกมา ยังคงมีคำที่สามารถบ่งบอกประเภทได้อย่างถูกต้องอยู่

จากการทดสอบจึงเลือกค่าถ่วงน้ำหนัก 25/10/1 นั่นคือให้ ค่าถ่วงน้ำหนักหัวข้อเรื่อง – 25 ค่าถ่วงน้ำหนักบทคัดย่อ – 10 ค่าถ่วงน้ำหนักเนื้อหา – 1 เนื่องจากกลุ่มคำของอินเด็กซ์ที่ได้มีความสัมพันธ์เกี่ยวข้องกับประเภทของเอกสารมากกว่า

5.2 ความถูกต้องในการแบ่งประเภทของเอกสาร

ในการทดสอบระบบโดยทำการทดสอบไฟล์ทั้งหมดประมาณ 90 ไฟล์ เพื่อหาอินเด็กซ์เทอม และจัดแบ่งประเภทของเอกสารนั้น จากผลการทดลองจะมีความถูกต้องในการแบ่งประเภทของเอกสารดังนี้

ประเภทของเอกสาร	% ความถูกต้องในการแบ่งแยกประเภทของเอกสาร	%ความผิดพลาด	
		แบ่งเอกสารผิด	ไม่สามารถแยกประเภทได้
AGENT	80	10	10
ARTIFICIAL INTELLIGENT	50	20	30
DATABASE	90	0	10
INFORMATION RETRIEVAL	50	20	30
NETWORK	40	40	20
OPERATING SYSTEM	50	30	20
PROGRAMMING	60	20	30
SECURITY	50	20	30
SOFTWARE ENGINEERING	40	30	30
เฉลี่ยเปอร์เซ็นต์ความถูกต้อง	55.55	21.11	23.33

ตารางที่ 5-2 แสดงผลของการแบ่งประเภทเอกสารโดยใช้ค่าถ่วงน้ำหนัก 25/10/1

จากการค้นคว้าจาก http://www.lub.lu.se/desire/DESIRE36a-overview.html ได้มีการทดลองในการแยกประเภทเอกสารประมาณ 1,000 เอกสารพบว่ามีความถูกต้องประมาณ 59% ซึ่งเมื่อนำมาเปรียบเทียบกับผลการทดลองในโครงงานนี้แล้วพบวามีค่าที่มากกว่า ซึ่งอาจจะมีสาเหตุจากการที่ในโครงงานมีการทดลองเอกสารจำนวนน้อยเกินไปเปอร์เซ็นต์ที่ได้จึงอาจจะไม่ถูกต้องนัก ประกอบกับเอกสารก็เป็น

คนละชุดกันและกระบวนการในการแบ่งประเภทแตกต่างกัน รวมถึงจำนวนประเภทของเอกสารที่กำหนดขึ้นมาก็ความแตกต่างกันด้วย

บทที่ 6 บทวิจารณ์และบทสรุป

6.1 บทวิจารณ์และบทสรุป

โครงงานนี้จัดทำขึ้นมาเพื่อเป็นการจำลองการทำงานของห้องสมุดเสมือนซึ่งจริงๆแล้วก็คือเสิร์ชเอ็นจินแบบไดเรกทอรีแบบหนึ่ง แต่ว่าจะมีการนำ Spider ซึ่งจะมาช่วยมนุษย์ในการทำการรวบรวมเก็บข้อมูลในไว้ในฐานข้อมูลของทางห้องสมุดเอง

ในโครงงานอาจจะมีข้อจำกัดในการจัดทำหลายประการ เช่น จำนวนเว็บไซต์ที่จะทำการจัดเก็บไว้ในฐานข้อมูลของห้องสมุดเสมือนเอง จะต้องมีกระบ .นการจัดทำอินเด็กซ์อัตโนมัติก่อนและไฟล์ข้อมูลที่ดึงมานั้นจะต้องเป็นไฟล์สกุล ".xml" ดังนั้นจึงมีขีดจำกัดในการที่จะสร้างเว็บไซต์เหล่านี้ขึ้นมาเพื่อให้ spider ไปดึงข้อมูลเหล่านี้ ซึ่งยังไม่สามารถที่จะจัดสร้างเว็บไซต์จำนวนมากเหล่านี้ขึ้นมาได้ ทำให้ในฐานข้อมูลของโครงงานนี้อาจจะยังน้อยอยู่

ส่วนของการจัดทำอินเด็กซ์นั้น แม้ว่าจะให้โปรแกรมจัดทำได้ดีเพียงใดก็ตาม ก็ต้องยอมรับว่าการให้มนุษย์จัดทำอินเด็กซ์นั้น ย่อมทำได้ดีกว่าการให้โปรแกรมจัดทำ เนื่องจากมนุษย์จะสามารถเข้าในถึงความหมายของคำได้มากกว่า

ในด้านการแบ่งแยกประเภทของข้อมูลก็เช่นเดียวกัน ต้องยอมรับว่าไม่มีโปรแกรมใดๆที่สามารถจะจัดทำการแบ่งประเภทของข้อมูลได้อย่างถูกต้อง 100 เปอร์เซ็นต์ เนื่องจากโปรแกรมไม่ได้ทราบถึงความหมายของคำในแต่ละคำจริงๆเหมือนอย่างมนุษย์ แต่เราสามารถใช้โปรแกรมช่วยในการตัดสินใจของคนได้ เพราะโปรแกรมเป็นเพียงเครื่องมือที่จะช่วยมนุษย์ (Machine-Aided) เท่านั้น ซึ่งในการตัดสินใจจริงๆก็ยังคงต้องเป็นมนุษย์ที่จะทำการแบ่งแยกประเภทของเอกสาร แม้ในปัจจุบันนี้เว็บไซต์ใหญ่ๆ ชื่อดัง อย่าง mweb การจัดแบ่งประเภทของเอกสาร ยังคงให้มนุษย์เป็นผู้อ่านเอกสารและแบ่งประเภทของเอกสารเอง โดยไม่ได้ใช้โปรแกรมใดๆช่วยเหลือเลย

ดังนั้นจะเห็นได้ว่าไม่ว่าจะเป็นกา รจัดทำ อินเด็กซ์หรือการจัดประเภทของเอกสารนั้น เราไม่สามารถที่จะให้โปรแกรมเป็นผู้จัดทำและให้ผลลัพธ์ออกมาถูกต้อง 100 เปอร์เซ็นต์ได้ ซึ่งเราทำได้แค่เพียงการจัดให้ผลลัพธ์มีความถูกต้องมากที่สุดเท่านั้น

ปัญหาที่พบในการจัดแบ่งประเภทของเอกสารนั่นก็คือ อาจจะเป็นไปได้ว่าเอกสารใดๆ ในตัวเนื้อหานั้นอาจจะมีความคลุมเคลือ แม้ว่าจะให้มนุษย์เป็นผู้จัดแบ่งแยกประเภทของเอกสารเอง ก็ยังไม่สามารถที่จะแยกแยะได้ว่าเป็นเอกสารประเภทใด หรือว่าอาจจะจัดอยู่ในประเภทของเอกสารได้หลายประเภทก็ได้ เนื่องจากอาจจะมีเนื้อหาเกี่ยวข้องหลายอย่าง

แต่ถึงอย่างไรก็ตาม ในด้านฟังก์ชันการทำงานต่างๆของห้องสมุดเสมือนนั้น ยังคงสามารถทำได้อย่างมีประสิทธิภาพดี ไม่ว่าจะเป็นการค้นหาข้อมูลตามคีย์เวิร์ดที่ผู้ใช้ต้องการ หรือการ Explore เพื่อดูข้อมูลในแต่ละประเภท หรือการที่สามารถหาคำใกล้เคียงกับคีย์เวิร์ดที่ผู้ใช้ต้องการ โดยทำการเช็คกับ Thesaurus และดึงข้อมูลต่างๆออกมาได้

นอกจากนี้ยังมีข้อจำกัดในด้านของฮาร์ดแวร์ทั้งทางด้านความเร็วของซีพียู และแรมที่ใช้ ซึ่งเครื่องที่ทำการรันโปรแกรมให้ Spider ไปดึงข้อมูลพร้อมทั้งผ่านกระบวนการจัดทำอินเด็กซ์อัตโนมัติและทำการเพิ่มข้อมูลลงฐานข้อมูล จะใช้เวลามากพอสมควร

เนื่องจากเอกสาร ".xml" ที่นำมาใช้ในโครงงานนั้น ปัจจุบันอาจจะยังไม่เป็นที่นิยม แต่คาดว่าในอนาคตจะมีการจัดเก็บข้อมูลในรูปแบบนี้ เพราะมีข้อดีหลายประการดังที่ได้กล่าวไว้แล้วในเนื้อหาข้างต้น นอกจากนี้การทำอินเด็กซ์อัตโนมัติและการแยกประเภทของเอกสารนั้น ก็ยังช่วยผ่อนแรงมนุษย์ในการที่จะต้องมานั่งอ่านเพื่อหาคำที่เป็นอินเด็กซ์เทอมหรือหาประเภทของเอกสาร จึงนับได้ว่ามีประโยชน์พอสมควร ซึ่งในปัจจุบัน เว็บไซต์ใหญ่ๆบางแห่ง ก็ยังคงใช้วิธีการให้มนุษย์เป็นผู้อ่านข้อความต่างๆและจัดแบ่งประเภทเองด้วย

สุดท้ายนี้ก็หวังว่าโครงงานนี้จะสามารถนำมาใช้งานได้จริงในอนาคตอันใกล้นี้ หรืออาจจะนำมาเป็นต้นแบบในการพัฒนาต่อไป

6.2 แนวทางการพัฒนาต่อ

1. การใช้ Neural Network มาทำให้การทำอินเด็กซ์อัตโนมัติมีความฉลาดมากยิ่งขึ้น โดยจะต้องมีกระบวนการของ Artificial Intelligent มาร่วมด้วย คือเมื่อยิ่งมีการทำอินเด็กซ์มากขึ้น หรือนำเอกสารต่างๆมาป้อนเป็นอินพุทของระบบให้มากๆ จะเกิดการเรียนรู้ว่าควรจะเลือกอินเด็กซ์เทอมใดที่เหมาะสมที่สุด
2. ทำการทดลองต่อไปเพื่อหาค่า weight ของแต่ละ tag ที่เหมาะสมที่สุด เพื่อที่จะสามารถได้ index term และแยกประเภทได้อย่างถูกต้อง
3. อาจจะมีการกำหนดค่า weight ให้แต่ละคำใน category list ไม่เท่ากันเพื่อเพิ่มประสิทธิภาพในการแยกประเภทของเอกสาร
4. พัฒนาทางด้านความเร็วของการจัดทำอินเด็กซ์อัตโนมัติ
5. ปรับปรุงรูปแบบของเว็บไซต์ให้ทันสมัย หรือมีสิ่งที่ให้บริการผู้ใช้มากยิ่งขึ้น

ภาคผนวก

ก. การติดตั้ง TOMCAT

Tomcat นี้เกิดมาจากโปรเจ็คของอาปาเช่ (Apache) ซึ่งต้องการนำเทคโนโลยีจาวามาใช้อย่างเต็มรูปแบบ ทั้งตัวเซิร์ฟเล็ต และเจเอสพี (JSP) ของจาวา และทางอาปาเช่ก็ได้ตั้งชื่อโปรเจ็คนี้ว่าจากาตาร์โปรเจ็ค (Jakarta Project) ผู้เข้าร่วมใน Project นี้อย่างเช่น ไอบีเอ็ม ซัน เป็นต้น (ร่วมทั้งสมาชิกจาก Apache Jserv Project) โดยในจากาตาร์โปรเจ็คนี้จะมีโครงงานย่อยๆอยู่มากมาย อย่างเช่น Ant, ORO ,Regexp ,Slide ,Struts ,Taglibs ,Tomcat ,Velocity ,Watchdog ซึ่งในที่นี้เราจะกล่าวถึงแต่ตัว Tomcat เพียงอย่างเดียว

ในขั้นตอนแรกต้องนำโปรแกรม Tomcat มาก่อน ซึ่งปัจจุบันก็มีถึงเวอร์ชัน 3.1 (เราสามารถเช็คดูได้ที่ http://jakarta.apache.org/) ใน Tomcat 3.1 นี้จะประกอบไปด้วย Servlet v.2.2 และ JSP v.1.1 ส่วนของตัวโปรแกรม สามารถดาวน์โหลดได้ที่

- http://aajc.au.ac.th/program/
- http://jakarta.apache.org/ (Official Wesite ของ Tomcat)

ก่อนที่จะเข้าไปดาวน์โหลดก็ต้องดูก่อนว่าเราใช้ระบบปฏิบัติการใดอยู่ จะได้นำโปรแกรมมาใช้ได้ถูกต้องและตรงกับระบบปฏิบัติการ

เมื่อได้ตัว Tomcat มา ถ้าเป็นพวกซิปของวินโดว์ก็อันซิปออก (มันจะสร้างไดเรกทอรี ชื่อ jakarta-tomcat ให้เรา) โดยไม่ต้องอินสตอลอะไรทั้งสิ้นแค่ทำการอันซิปเท่านั้นพอ หรือถ้าเป็นยูนิกส์ก็จัดการ Gunzip แล้วก็ tar xvf ออกมาเราก็จะได้ไดเรกทอรีของ tomcat ซึ่งจะประกอบด้วยสับไดเรกทอรีที่สำคัญดังนี้

Bin	เอาไว้เก็บ File ต่างที่เอาไว้สำหรับ Execute Tomcat
Conf	เอาไว้เก็บ File ต่างที่เอาไว้สำหรับ Config ตัว Tomcat, Apache JServ , JNI สำหรับ Tomcat
Lib	จัดเก็บ API ทั้งหมดที่เกี่ยวกับ Tomcat
Logs	Log File เกี่ยวกับ Tomcat (จะถูกสร้างขึ้นเมื่อ มีการ startup Tomcat แล้วหนึ่งครั้ง)
Webapps	Directory ที่จัดเก็บ file ที่เป็น application ของเราอย่างพวก Servlet,JSP รวมทั้ง Web Page ด้วย (ซึ่งเป็น default ของ Tomcat)

ตารางที่ ก-1 แสดงสับไดเรกทอรีที่สำคัญของทอมแค็ท

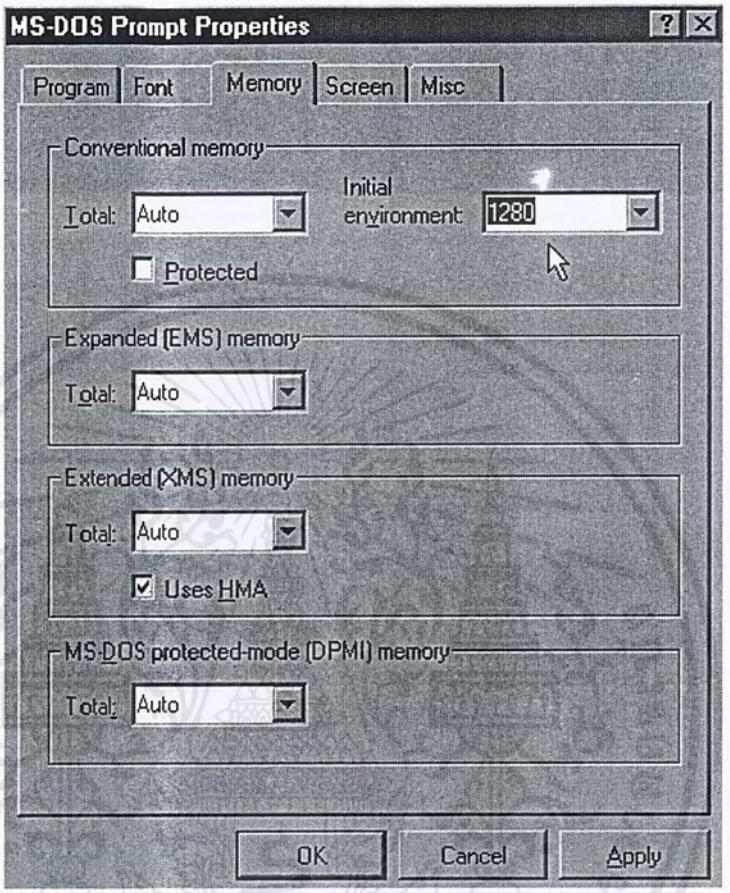
<u>วิธีการเรียก TOMCAT SERVER</u>

ไฟล์ที่เราจะใช้เรียก Tomcat ในวินโดว์ ได้แก่ ไฟล์ :

- startup.bat (จะมีหน้าที่เปิด Tomcat server)
- shutdown.bat (จะทำหน้าที่ปิด Tomcat server)

โดยจะมีการเรียกผ่านดอสพรอมพ์ ส่วนไฟล์สำหรับเรียก Tomcat ในระบบยูนิกส์ก็จะมีชื่อเหมือนกันเพียงจะต่างตรงนามสกุลที่ใช้เท่านั้น อย่างใน Solaris ก็จะเป็นพวก .sh เป็นต้น Ex. startup.sh

บางทีอาจจะเกิดปัญหาเมมโมรี ใน environment ของดอสไม่พอ เราจะต้องแก้ปัญหาโดยการตั้งค่าเมมโมรีใหม่ใน Dos Properties โดยตั้งค่าเมมโมรี ของ environment ให้เป็น 1280 ขึ้นไป ดังรูป



รูปที่ ก-1 แสดงการแก้ปัญหาในกรณีเมมโมรีใน environment ของดอสไม่พอ

เมื่อเราทำการ Startup Tomcat เมื่อไหร่ ตัว Tomcat ก็จะทำการตั้งตัวเองเป็นเว็บเซิร์ฟเวอร์เพื่อรอการร้องขอข้อมูล และทำหน้าที่ตอบกลับไปโดยเราสามารถกดเข้าไปดูว่าเว็บเซิร์ฟเวอร์ ขึ้นหรือยังโดยการเรียกผ่านเบราเซอร์ อย่าง IE หรือ Netscape ก็ได้โดยพิมพ์ URL ดังนี้

URL: http://localhost:8080

(8080 คือ default port ของ Tomcat ซึ่งเราสามารถ set ให้เป็น port 80 ** ได้ใน file: server.xml ที่อยู่ใน directory "conf" ซึ่ง Port 80 : เป็น port มาตรฐานสำหรับส่งข้อมูลแบบ protocol HTTP)

<u>วิธีการ Config Tomcat server (tomcat.properites)</u>

การ Config Tomcat เราจะทำที่ jakarta-tomcat/conf ซึ่งจะประกอบด้วยไฟล์ config จำนวนมากมาย แต่ ไฟล์ที่เราจะใช้คราวนี้จะมีอยู่ 2 ไฟล์ เท่านั้นนั่นคือ

- tomcat.properties (ทำหน้าที่ config environment ของ tomcat รวมทั้ง path ต่างๆใน Java environment)

- server.xml (ทำหน้าที่ set context ต่างๆ ใน tomcat)

configuration ใน tomcat.properties โดยปกติ เราจะใช้ Tomcat ได้เลยในระบบของวินโดว์แต่สำหรับในระบบของ Unix อาจจะต้องมีการตั้งค่าพาร์ธ ของ JVM ให้กับ Tomcat ด้วย โดยจะตั้งค่าตามนี้

(ของเก่า)

```
# The Java Virtual Machine interpreter.
# Syntax: wrapper.bin=[filename] (String)
# Note: specify a full path if the interpreter is not visible in your path.
wrapper.bin=@JAVA@
```

(ของใหม่)

```
# The Java Virtual Machine interpreter.
# Syntax: wrapper.bin=[filename] (String)
# Note: specify a full path if the interpreter is not visible in your path.
wrapper.bin=@c:\jdk1.2.2\bin@
```

ส่วน classpath จะตั้งค่าใน tomcat.properties เช่นกันแต่อยู่ในบรรทัดนี้

wrapper.classpath=@JSDK_CLASSES@ แก้เป็น wrapper.classpath=@c:\jdk1.2.2\bin\servlet.jar@

<u>วิธีการ Config Tomcat server (server.xml)</u>

server.xml fileตัวนี้ค่อนข้างมีความสำคัญอย่างมากเพราะจะเป็นตัวตั้งค่าพาร์ธของแอดเดรสของ Servlet Application ที่จะใช้ในเว็บไซต์และเป็นตัวตั้งค่าพอร์ทให้กับ Tomcat ด้วย

วิธีการตั้งค่าพอร์ทใหม่ให้กับ Tomcat เราจะทำดังนี้

(ของเก่า)

```
<Connector className="org.apache.tomcat.service.SimpleTcpConnector" >
<Parameter name="handler" value="org.apache.tomcat.service.http.HttpConnectionHandler"/ >
<Parameter name="port" value="8080" / >
</Connector>
```

(ของใหม่)

```
<Connector className="org.apache.tomcat.service.SimpleTcpConnector" >
<Parameter name="handler" value="org.apache.tomcat.service.http.HttpConnectionHandler"/ >
<Parameter name="port" value="80" / >
</Connector>
```

วิธีการเซ็ตแอดเดรสใหม่ให้กับ Tomcat เราจะแบ่งออกเป็นสองแบบ คือ

1. การเซ็ตโฮมเพจหน้าแรก ซึ่งเป็นการเซ็ตข้อมูลของโฮมเพจว่าอยู่ที่ไหน ซึ่งค่ามาตรฐาน (default) ของ Tomcat จะเป็นดังนี้

(ของเก่า)

<Context path="" docBase="webapps/ROOT" debug="0" reloadable="true" >

</Context >

(ของใหม่ ในแบบยูนิกส์)

<Context path="" docBase="home/aajc" debug="0" reloadable="true" >

</Context >

(ของใหม่ ในแบบวินโดว์)

<Context path="" docBase="c:\home\aajc" debug="0" reloadable="true" >

</Context >

พาร์ธของโฮมเพจที่เราสมมุติขึ้นอยู่ที่ /home/aajc/index.html แอททริบิวต์ของข้อความมีดังนี้

path	เป็นการบอก Url ต่อจาก Url หลักที่เราได้ regist ไว้ใน DNS เช่น http://aajc.au.ac.th/ ถ้าใส่เป็น path= "/jboy" ก็จะเป็น http://aajc.au.ac.th/jboy และถ้าใส่เป็น "" ก็จะกลายเป็น homepage หน้าแรก
docBase	เป็นการ set directory ที่เราเก็บ document file ไว้ถ้าเป็น file แบบ html เราไม่จำเป็นต้องสร้าง context เลยเพราะตัว Tomcat จะจัดการเองให้หมด แต่ถ้าเป็นแบบ Servlet เราต้อง set docBase ให้ (จะแสดงให้ดูในภายหลัง)
debug	เป็นการ set piority ของ debug
reloadable	เป็นการ set ให้ Servlet สามารถ reload ได้ถ้าเรามีการแก้ไข Servlet แล้ว

ตารางที่ ก-2 แสดงแอททริบิวต์ของข้อความ

2. การเซ็ต Servlet Application จะมีวิธีเซ็ตเหมือนกับการเซ็ตโฮมเพจโดยใช้ข้อความ เช่น HelloWorld.java ซึ่งเป็น Servlet Application แล้วนำไปเก็บไว้ในไดเรกทอรี /jakarta-tomcat/webapps/jservlet/WEB-INF/classes/HelloWorld.java (ในที่นี้สมมุติตั้งชื่อพาร์ธว่า jservlet แล้วต้องสร้างไดเรกทอรี "jservlet" ขึ้นมาด้วย) จากนั้นคอมไพล์ HelloWorld.java ให้เป็น .class Servlet Application ทุกๆตัวจะถูกเก็บอยู่ในสับไดเรกทอรี " /WEB-INF/classes" ซึ่งเราจะตั้งกี่ไดเรกทอรีก็ได้แต่จะต้องมีสับไดเรกทอรีนี้อยู่

(ของเก่า)

<Context path="/examples" docBase="webapps/examples" debug="0" reloadable="true">

< /Context>

(ของใหม่ ในแบบ Unix)

<Context path="/jservlet" docBase="webapps/jservlet" debug="0" reloadable="true" >

</Context >

ในตัวอย่างนี้ได้ set path="/jservlet" หมายความว่าเราจะได้ URL สำหรับที่เก็บ Servlet เป็น http://aajc.au.ac.th/jservlet/servlet (ใน Tomcat จะต่อ path "/servlet" ให้ด้วย) ส่วน docBase="webapps/jservlet" เราจะบอกว่า Servlet Application เราเก็บไว้อยู่ที่

" /jakarta-tomcat/webapps/jservlet/WEB-INF/classes "

ข. การติดตั้ง Parser สำหรับการ parse ไฟล์สกุล ".xml"

มีขั้นตอนดังต่อไปนี้

1. ดาวน์โหลดไฟล์ XML4J-3_1_0.zip จาก http://www.megginson.com/SAX/
2. นำไฟล์ XML4J-3_1_0.zip มาทำการ unzip แล้วนำไปเก็บไว้ในโฟลเดอร์เดียวกับ jdk
3. ทำการตั้งค่า class path กับ tool ที่ใช้ในการเขียนจาวา ในโครงงานชิ้นนี้ใช้ EditPlus ในการเขียนโปรแกรม ดังนั้นให้เข้าไปตั้งค่าที่ Tools / Configure User Tools

   ในช่องของ Argument ให้พิมพ์ข้อความเหล่านี้เพิ่มลงไป ทั้งใน Menu item – java, javac (การคอมไพล์จาวา และการรันจาวา) ในกรณีที่นำไฟล์ที่อันซิปไปใส่ไว้ในโฟลเดอร์ c:\jdk1.3

   -classpath .;C:\jdk1.3\XML4J-3_1_0\xml4j.jar;C:\jdk1.3\XML4J-3_1_0\xerces.jar;c:\jdk1.3\XML4J-3_1_0\xercesSamples.jar
4. ในส่วนของการเขียนโปรแกรมให้ทำการอิมพอร์ทค่าเหล่านี้ลงไปด้วย

```
import org.xml.sax.Attributes;
import org.xml.sax.ContentHandler;
import org.xml.sax.ErrorHandler;
import org.xml.sax.Locator;
import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;
import org.xml.sax.XMLReader;
import org.xml.sax.helpers.XMLReaderFactory;
```

ค. การติดตั้งคลาสเพื่อให้สามารถติดต่อกับฐานข้อมูลได้

มีขั้นตอนดังต่อไปนี้

1. นำไฟล์ jdbc-3.jar มาลงไว้ในโฟลเดอร์ c:\jdk1.3\lib\ หรืออาจจะนำไปลงไว้ในโฟลเดอร์อื่นใดก็ได้ แล้วเมื่อทำการตั้งค่า classpath ก็ต้องตั้งไปที่นั่น
2. ทำการตั้งค่า class path กับ tool ที่ใช้ในการเขียนจาวา สำหรับ EditPlus นั้น ให้เข้าไปตั้งค่าที่ Tools / Configure User Tools

   ในช่องของ Argument ให้พิมพ์ข้อความเหล่านี้เพิ่มลงไป ทั้งใน Menu item – java, javac (การคอมไพล์จาวา และการรันจาวา)

c:\jdk1.3\lib\jdbc-3.jar

ถ้ามีการนำไฟล์ jdbc-3.jar ไปเก็บไว้ในโฟลเดอร์อื่น ก็ให้เซ็ตค่า classpath ไปที่นั่น แล้วตามด้วยไฟล์ jdbc-3.jar

3. ในส่วนของการโปรแกรมให้ทำการอิมพอร์ทค่าต่อไปนี้ด้วย

```
import oracle.jdbc.driver.*;
```

ง. ทฤษฎีเบื้องต้นของ XML

Markup language เป็นมาตรฐานการทำงานอย่างหนึ่งของโลกงานพิมพ์ ซึ่ง mark up เป็นวิธีการดั้งเดิมในการจัดรูปแบบเอกสารด้วยการใช้สัญลักษณ์แทนรูปแบบต่างๆ ลงไปในเอกสารที่เกิดขึ้นในโลกของงานพิมพ์และการออกแบบ โดย Markup เป็นส่วนที่ทำการห่อหุ้มเอกสารอิเล็กทรอนิกส์ โดยมีจุดประสงค์หนึ่งในสองจุดประสงค์ต่อไปนี้

1. ปรับแต่งมุมมองและจัดรูปแบบของข้อความ
2. สร้างโครงสร้างและสื่อความหมายในแต่ละส่วนของเอกสารว่าต้องการแสดงผลเอกสารในรูปแบบใดด้วยสื่อต่างๆ เช่น เครื่องพิมพ์ หรือใน World Wide Web

XML (eXtensible Markup Language) คือ Markup languageใหม่ ที่ถูกพัฒนาขึ้นมาเนื่องจาก HTML (Hyper Text Markup Language) มีข้อจำกัดมากมาย โดยอาศัยโครงสร้างของ SGML (Standard Generalized Markup Language) เป็นหลักในการพัฒนา หรือจะกล่าวได้ว่า XML เป็น subset ของ SGML ก็ได้ XML นั้นถูกพัฒนาขึ้นมาเพื่อให้ใช้งานได้ง่ายในงานเว็บ และการถ่ายโอนข้อมูลระหว่างแอพพลิเคชัน และแพลตฟอร์มที่ต่างๆกัน โดย XML ไม่ได้มีไว้ใช้สำหรับเขียนโปรแกรมอย่างภาษา c++, pascal หรือ assembly แต่ว่าเป็นภาษาที่ใช้ในการอธิบายข้อมูลหรือจัดรูปแบบในการแสดงผลเพื่อแอพพลิเคชันอื่นจะนำ XML ไปประมวลผลเพื่องานต่างๆ

ข้อเปรียบเทียบระหว่าง HTML กับ XML

	HTML	XML
ประเภท	Markup Language	Markup Language
การใช้งาน	web	web, การถ่ายโอนข้อมูล และ งานเฉพาะสาขา
ลักษณะสำคัญ	กำหนดการแสดงผลโดยไม่สนใจว่าข้อมูลที่แสดงออกไปคืออะไร	จะสนใจว่าข้อมูลคืออะไร โดยจะแยกส่วนการแสดงผลกับข้อมูลออกจากกัน
ชนิดของไฟล์	text file	text file
นามสกุล	.HTM , .HTML	.XML
จำนวน tag	มีจำกัดตามที่กำหนดไว้ในแต่ละรุ่น	ไม่จำกัด เนื่องจากผู้ใช้สามารถกำหนดได้เอง
การแสดงผล	กำหนดการแสดงผลในตัว HTML เองเลย	ต้องใช้ stylesheet มากำหนดการแสดงผล
web browser	แทบทุกตัวที่มี เพียงแต่จะขึ้นอยู่กับรุ่นว่าสามารถสนับสนุนการทำงานได้เต็มที่หรือไม่	Internet Explorer 5 , Netscape 4.0.4 และ Mozilla 5

ตารางที่ ง-1 ตารางแสดงการเปรียบเทียบ xml และ HTML

XML มีความสัมพันธ์กับ HTML อย่างไร ? โดยมากจะมีการเข้าใจผิด คิดว่า XML จะเข้ามาแทนที่ HTML แม้ว่าจะมีความเป็นจริงอยู่บ้าง แต่แท้จริงแล้วทั้ง 2 เป็นส่วนเสริมการทำงานซึ่งกันและกัน การ

ทำงานของทั้งสองนั้น ในความเป็นจริงแล้วมีระดับการใช้งานที่แตกต่างกันไปตามแต่ชนิดของข้อมูลโดยที่ XML จะใช้ในการโครงสร้างและนิยามข้อมูลบนเว็บ และใช้ HTML ในการจัดรูปแบบให้กับข้อมูล แต่เพราะว่าเพจ HTML ส่วนใหญ่แล้ว สามารถจัดเก็บข้อมูลได้ดีไม่แพ้การจัดรูปแบบให้กับข้อมูลที่บรรจุอยู่ในโค้ด HTML บางครั้งเราจึงสามารถนำเพจ HTML มาใช้เป็นที่เก็บข้อมูลด้วย แต่ส่วนใหญ่แล้ว XML จะรับงานจัดเก็บข้อมูลทั้งหมดและ HTML จะถูกนำไปใช้ในการจัดรูปแบบและการทำสคริปต์

1. ส่วนประกอบของ XML

1.1 XML Document

คือ ส่วนที่เป็นข้อมูล ซึ่งมีลักษณะดังตัวอย่างต่อไปนี้

```
<?xml version= "1.0"?>
<root_tag>
        <first_child_tag>
                this is 1st child tag
                        <second_child_tag>
                                this is 2nd child tag
                        </second_child_tag>
        </first_child_tag>
</root_tag>
```

1.2. DTD (Document Type Definition)

XML นั้นเป็นภาษาที่ไม่มี tag ในรูปแบบที่ตายตัว คือ เราสามารถกำหนด tag ต่างๆขึ้นมาได้เอง ดังนั้นการที่เราจะสร้างแอพพลิเคชันใดๆขึ้นมาเพื่อให้อ่านข้อมูล XML นั้น เราจะต้องมีการกำหนดรูปแบบมาตรฐานของข้อมูล XML เพื่อที่ว่าแอพพลิเคชันนั้นๆ จะสามารถทำงานได้อย่างถูกต้อง ซึ่งจริงๆ แล้ว XML อาจจะไม่จำเป็นที่จะต้องมี DTD ก็ได้

การกำหนดมาตรฐานนั้น ก็คือ การกำหนดว่าใน XML นั้นๆ จะต้องมี tag อะไรบ้าง และ ห้ามมี tag อะไรบ้างกำหนดว่าในแต่ละ tag นั้นจะมีอะไรได้บ้าง เช่น มี tag ลูกได้หรือไม่ ถ้ามีได้ มีได้กี่อัน และต้องมีชื่อว่าอะไร เป็นต้น

เราใช้ DTD ในการกำหนดข้อกำหนดต่างๆ ของ XML และเราจะใช้ XML Parser ตรวจสอบว่า XML นั้นๆ ถูกต้องตาม syntax และข้อกำหนดของ DTD หรือไม่ โดยเราจะสามารถแบ่งความถูกต้องของเอกสาร XML ได้เป็น 3 แบบด้วยกัน คือ

1. Invalid XML คือ XML ที่ไม่ถูกต้องตาม XML syntax เช่น ไม่ได้ปิด tag : <a> หรือ มี tag ซ้อนกัน : <a>

2. Well-formed XML คือ XML ที่ถูกต้องตาม syntax แต่ว่าไม่มี DTD หรือ ถูกต้องตาม syntax แต่ว่าไม่ถูกต้องตาม DTD
3. Valid XML คือ XML ที่ถูกต้องตาม syntax และ DTD ที่กำหนดเอาไว้

1.2.1 การประกาศอิลิเมนต์ (Element Declaration)

ส่วนการประกาศอิลิเมนต์แต่ละตัวจะประกอบไปด้วย ชื่ออิลิเมนต์และชนิดข้อมูลของอิลิเมนต์ ซึ่งเรารวมเรียกว่า ข้อกำหนดรายละเอียดเนื้อหาเอกสาร (Content Specification) มีอยู่ทั้งหมด 4 ชนิด อิลิเมนต์แต่ละตัวสามารถเลือกใช้ได้ดังนี้

1. รายการอิลิเมนต์ต่างๆที่เราเรียกว่าโมเดลเนื้อหาเอกสาร (Content Model) จะเริ่มต้นด้วยการประกาศอิลิเมนต์ด้วยอิลิเมนต์ที่จัดเก็บโมเดลเนื้อหาเอกสารโดยจะทำการบรรจุอิลิเมนต์ลูกทุกตัวตามลำดับ
   เช่น <!ELEMENT INFORMATION (TITLE, ABSTRACT, DESCRIPTION)>
   นั่นคือใน root element จะมีอิลิเมนต์ลูกได้แก่ TITLE, ABSTRACT, DESCRIPTION
2. การประกาศอิลิเมนต์เปล่า (Empty element) การประกาศอิลิเมนต์ที่ไม่สามารถจัดเก็บเนื้อหาเอกสารได้ สามารถประกาศโดยใช้คำสำคัญ EMPTY
   เช่น <!element test empty>
   อิลิเมนต์ Test ในส่วนการประกาศข้างบนนี้ไม่สามารถจัดเก็บข้อมูลได้ และการประกาศนี้เป็นการแสดงว่ามันเป็นอิลิเมนต์เปล่า ตัวอย่างการเขียนอิลิเมนต์เปล่าคือ <TEST/> แม้ว่าอิลิเมนต์เปล่ามองดูแล้วเหมือนจะใช้ประโยชน์อะไรไม่ได้ แต่ว่ามันสามารถมีแอททริบิวต์ เพื่อจัดเตรียมข้อมูลในการนิยามหรือจัดเตรียมหน้าที่การทำงานพิเศษให้กับเอกสารได้
3. การประกาศอิลิเมนต์บางส่วน ANY (Any element) ตรงกันข้ามกับอิลิเมนต์เปล่า เพราะถ้าประกาศอิลิเมนต์ด้วยคำสำคัญ ANY อิลิเมนต์ตัวนั้น จะสามารถบรรจุทุกสิ่งที่ประกาศไว้ใน DTD ได้ โดยไม่จำเป็นต้องมีทั้งหมด แต่ต้องมีการลำดับที่ประกาศไว้
   ตัวอย่างเช่น <!ELEMENT TEST ANY>
4. อิลิเมนต์ที่มีการผสมเนื้อหาเอกสาร (Mixed content) สามารถเลือกค่าใดค่าหนึ่งจากข้อมูลที่ระบุในอิลิเมนต์ทุกค่า ซึ่งคั่นแต่ละค่าด้วยเครื่องหมายไปป์ (|)
   เช่น <!ELEMENT EXAMPLE (#PCDATA | X | Y | Z) * >

1.3 Stylesheet

เป็นส่วนกำหนดการแสดงผลของ XML ซึ่งมี 2 แบบด้วยกัน คือ CSS และ XSL โดยที่ XML ไม่จำเป็นจะต้องมี stylesheet ก็ได้ หากว่า XML นั้นถูกแอพพลิเคชันอื่นนำไปใช้ แต่ว่าหากเราต้องการแสดงผลออก browser ก็จะต้องมี stylesheet

1.3.1 CSS (Cascading Style)

เป็นภาษาที่ใช้กำหนดรูปแบบการแสดงผลของ XML โดยจะเป็นการกำหนดคุณสมบัติต่างๆ ตัวอย่างเช่น ขนาด, สี, พื้นหลังของตัวอักษร เป็นต้น ในตอนต้น CSS ได้ถูกพัฒนาขึ้นเพื่อใช้กับ HTML ใน

การกำหนดรูปแบบการแสดงผลของข้อความต่างๆ แต่เนื่องจาก XML มีข้อได้เปรียบที่สามารถกำหนด TAG ได้เองจึงทำให้สามารถกำหนดคุณสมบัติของแต่ละ TAG ได้ ทำให้กำหนดการแสดงผลได้ดีกว่า

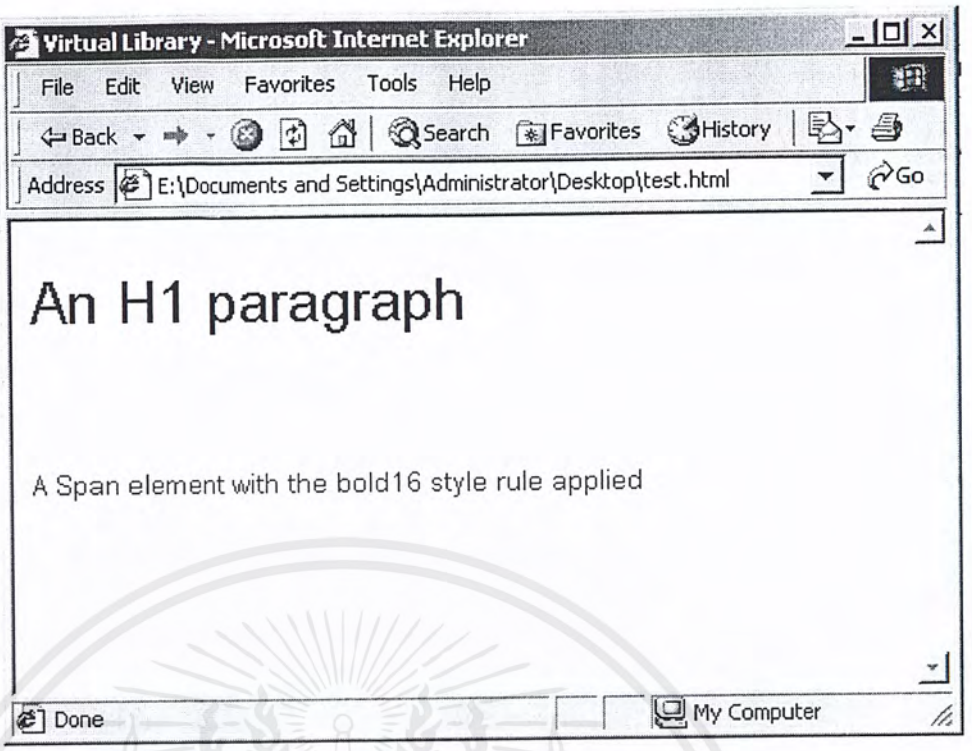
CSS มีส่วนประกอบพื้นฐาน 2 ส่วน คือ ส่วนของสไตล์ชีตที่จัดเก็บข้อกำหนดสไตล์ชีตหรือกฎที่จะนำมาใช้กับเอกสาร และอีกส่วนคือ ส่วนที่เป็นเอกสารที่จะนำสไตล์ชีตมาใช้ ตัวอย่างเช่น ไฟล์ "lst.css"

```
H1 { font – style : italic ;   font – size : 24 }
.blod16 { font – weight : bold;        font – size : 16 }
```

ต่อไปเป็นส่วนของเอกสารที่จะนำสไตล์ชีตเข้ามาใช้เป็นเอกสาร HTML สังเกตภายในตัวอย่างมีแท็ก <LINK> สำหรับนำสไตล์ชีตเข้ามาใช้ในเอกสาร

```
<!DOCTYPE HTML PUBLIC " - // W3C//DTD  HTML 3.2 Final //EN " >
<HTML>
    <HEAD>
        <TITLE> Virtual Library </TITLE>
        <LINK HREF = "lst.css" REL = STYLESHEET TYPE = "text/css">
    </HEAD>
    <BODY>
        <H1>An H1 paragraph </H1>
        <BR>
        <SPAN CLASS = bold16>
            A Span element with the bold16 style rule applied
        </span>
    </BODY>
</HTML>
```

จากตัวอย่างจะทำการสไตล์ไว้ 2 แบบ แต่ละแบบที่กำหนดไว้มีการใช้งานต่างกัน สไตล์แรกเป็นการกำหนดสไตล์ให้กับอิลิเมนต์ H1 ของเอชทีเอ็มแอล เมื่อใดก็ตามที่ใช้งานอิลิเมนต์ H1 รูปแบบการแสดงผลของเอชทีเอ็มแอล จะเป็นไปตามที่กำหนดไว้ในสไตล์ชีต สไตล์ที่สองคือสไตล์ bold16 สไตล์ตัวนี้เป็นคลาสของสไตล์ (สังเกตในรายการโค้ดที่ส่วนประกาศสไตล์ที่มีจุดอยู่หน้าชื่อของสไตล์) ดังนั้นการจะนำมาใช้งานต้องทำการประกาศใช้ด้วยแท็ก class ของอิลิเมนต์ Span เสียก่อน การประกาศ การใช้คลาสสไตล์ต้องประกาศให้ตรงกับชื่อคลาสสไตล์ที่ตั้งเอาไว้ในสไตล์ชีต ซึ่งผลที่ได้จะได้ตามรูปด้านล่างนี้



รูปที่ ง-1 แสดงการแสดงผลโดยใช้สไตล์ชีท แบบ CSS

1.3.2 XSL (XSL Translation Language)

XSL ใช้งานสไตล์ชีตสำหรับจัดรูปแบบเอกสารเช่นกัน ความยืดหยุ่นที่ได้ก็ไม่แพ้กับของ CSS แต่ว่าสไตล์ชีตของทั้ง 2 นั้นมีความแตกต่างกัน สำหรับ XSL นั้นใช้วิธีการสร้างแม่แบบซึ่งบางวิธีจะคล้ายกับของ CSS แม่แบบของ XSL ที่เราสร้างขึ้นมานั้นนำไปใช้ในการจัดเตรียมกลไกการจัดรูปแบบสารสนเทศให้กับข้อมูลที่ต้องการจัดสไตล์

ส่วนประกอบของ XSL ประกอบไปด้วย 2 ส่วน คือ ภาษาปริวรรต XFL (XSL Transformation Language) และภาษาสำหรับการจัดรูปแบบอ็อบเจ็กต์ (Formatting Object Spedification) ทั้ง 2 ส่วน นำมาใช้ในการจัดเตรียมรูปแบบการแสดงผลให้กับเอกสาร และนำไปใช้เป็นเนมสเปซของ XML

- ภาษาปริวรรต XSL (XSL Transformation language) ใช้บรรยายการโปรแกรมประมวลผลแปรรูปเอกสาร XML จากโครงสร้างหนึ่งไปยังอีกโครงสร้างหนึ่ง ส่วนใหญ่ที่นิยมและมักนำมาใช้คือการแปรรูปจากโครงสร้างภาษา (Semantic Structure) ไปเป็นโครงสร้างการแสดงผล (Display Structure) ตัวอย่างเช่น การแปรรูปจากเอกสาร XML ไปเป็นเอกสารเอชทีเอ็มแอลเป็นต้น แต่ว่าการแปรรูปที่เกิดขึ้นไม่จำเป็นต้องเป็นเหมือนกับตัวอย่างข้างต้นเสมอไป เนื่องจากกระบวนการแปรรูปไม่ยึดติดกับผลลัพธ์สุดท้ายที่ได้ การยินยอมให้เป็นเช่นนั้นเป็นความสามารถที่มีการเพิ่มขยายอันยิ่งใหญ่ที่จะมีขึ้นได้ในอนาคต ซึ่งจะเป็นการแปรรูปเอกสารสู่โครงสร้างอื่นๆที่เราอาจคาดไม่ถึง

- ภาษาสำหรับการจัดรูปแบบ (Formatting object specification) ใช้สำหรับสร้างภาษาสำหรับการจัดรูปแบบตัวใหม่ ซึ่งในขณะนี้ได้รับการพัฒนาเป็น Vocabulary ของ XML ดังนั้นเอ็นจินสำหรับการแสดงผลจึงสามารถประมวลผลข้อมูลการจัดรูปแบบที่มีอยู่ภายในเนมสเปซ fo ได้โดยตรง (ข้อมูลที่อยู่ภายในเนมสเปซ fo ไม่เหมือนกับข้อมูลในเนทสเปซ XSL) หรือก็คือโปรแกรมประมวลผลสามารถแปรรูปข้อมูลการจัดรูปแบบไปเป็นโครงสร้างการจัดรูปแบบในลักษณะอื่นๆ อย่างเช่น โค้ดเอชทีเอ็มแอล วิธีการนี้ต่างจากวิธีการของเนมสเปซ XSL นั่นคือวิธีการของเนมสเปซ fo เป็นวิธีการที่เกี่ยวพันเฉพาะกับภาษาสำหรับการจัดรูปแบบซึ่งคือการอนุญาตให้ Vocabulary แต่ละตัวที่สร้างขึ้นมาใช้ได้กับแอพพลิเคชั่นเฉพาะงานได้ เช่น มัลติมีเดีย เป็นต้น ความสามารถที่เด่นในการแปรรูปโมเดลอ็อบเจ็กต์เอกสารและเป็นอิสระจากภาษาสำหรับการจัดรูปแบบ

สไตล์ชีตของ XSL

สไตล์ชีตสามารถมีแม่แบบ (Template) ได้ตั้งแต่หนึ่งตัวขึ้นไป ซึ่งภายในแม่แบบประกอบไปด้วยแพทเทิร์น (Pattern) ต่างๆ เพื่อใช้จัดเตรียมโครงสร้างการแสดงผลของเอกสาร อิลิเมนต์ที่ใช้จะเป็นอิลิเมนต์ใดก็ได้ นอกจากนี้แม่แบบ XSL ไม่จำเป็นต้องมีการอ้างอิงไปยังข้อมูลของ XML

XSL ใช้แพทเทิร์นในการระบุรายละเอียดต่างๆให้กับอิลิเมนต์ของ XML ที่ต้องการประยุกต์ใช้งานแม่แบบ XSL การจับคู่กันระหว่างแพทเทิร์นและอิลิเมนต์ในลักษณะนี้ เป็นวิธีการที่ทำให้ XSL เป็นภาษาที่ต้องมีการประกาศ (Declarative Language) ซึ่งมีการทำงานตรงข้ามกับภาษากระบวนการ (Procedural Language) ดังนั้นแพทเทิร์นใน XSL จะต้องนิยามถึงรายละเอียดแต่ละกิ่งในต้นไม้เอกสารที่ตรงกับแพทเทิร์น ด้วยการแสดงแต่ละกิ่งเป็นลำดับชั้นในต้นไม้ เช่น ROOT/NODE1 แพทเทิร์นนี้แสดงว่า “อิลิเมนต์ node1 อยู่ภายในอิลิเมนต์ราก (Root)” การทำความเข้าใจโครงสร้างของแม่แบบ ดังตัวอย่างสไตล์ชีตต่อไปนี้

```
<CATALOG>
        <PLANT>
                <COMMON> Bloodroot  </COMMON>
                <BOTANICAL> Sanguinaria canadensis </BOTANICAL>
                <ZONE> 4 </ZONE>
                <LIGHT> Mostly Shady </LIGHT>
                <PRICE>$7.05 </PRICE>
                <AVAILABILITY USONLY = “true”> 02/01/99 <AVAILABILITY>
        </PLANT>
</CATALOG>
```

ต่อไปจะเป็นตัวอย่างการสร้างสไตล์ชีท XSL

```
<?xml version=”1.0”?>
```

```
<xsl:template xmlns:xsl="uri:xsl">
    <HTML>
        <BODY>
            <xsl:repeat for = "CATALOG/PLANT">
            <DIV>
                <SPAN STYLD="font-weight:bold; font-size:20">
                    <xsl:get-value for = "COMMON"/>
                </SPAN>
            </DIV>
            </xsl:repeat>
        </BODY>
    </HTML>
</xsl:template>
```

แพทเทิร์นแรกที่ปรากฏในโค้ดระบุให้ใช้อิลิเมนท์ Plant หนึ่งตัวจากอิลิเมนต์ Catalog (สังเกตว่าในแพทเทิร์นนี้มีค่าว่า repeat for อยู่) แพทเทิร์นตัวที่ 2 ระบุใช้อิลิเมนท์ Common เพื่อใช้ดึงข้อมูลใส่อิลิเมนท์ Span และกำหนดให้มีสไตล์ "font-weight:bold; font-size:20" ตามที่ได้กำหนดไว้ในอิลิเมนต์ Span ผลลัพธ์เมื่อรันออกมาแล้วจะได้เอกสารเอชทีเอ็มแอลที่มีโค้ดดังนี้

```
<HTML>
    <BODY>
        <DIV>
            <SPAN STYLE="font-weight:bold; font-size:20">
                Bloodroot
            </SPAN>
        </DIV>
    </BODY>
</HTML>
```

จะเห็นได้ว่าแม่แบบจะใช้แพทเทิร์นในการดึงข้อมูลจากอิลิเมนท์ทุกตัวที่ตรงกับแพทเทิร์น ดังนั้นอิลิเมนท์อื่นๆ ทุกตัวที่ไม่ตรงกบแพทเทิร์นจะไม่ถูกนำมาใช้แสดงผล การทำงานลักษณะนี้ทำให้การแสดงผลมีความสะดวก และง่ายดาย ทั้งนี้เป็นพลังความสามารถที่จัดเตรียมโดย XSL การจัดเรียงข้อมูลในเอกสารจึงทำได้ตามจุดประสงค์การใช้งาน เช่น ถ้าหากอิลิเมนต์ใดที่คุณไม่ต้องการใช้ในการแสดงผล ก็ไม่ต้องมีแพทเทิร์นให้กับมัน หรืออิลิเมนท์ใดที่ต้องใช้ หรือต้องการให้แสดงผลพิเศษกว่าอิลิเมนต์อื่น ก็สามารถระบุเพิ่มเติมไปในแพทเทิร์นได้อย่างง่ายดาย

1.4 XML Parser

เป็นโปรแกรมที่ใช้ตรวจสอบความถูกต้องของ XML ว่าเขียนได้ถูกตามข้อกำหนดและ DTD หรือไม่

1.5 Browser

ในขณะนี้มี browser ดังนี้ที่สนับสนุน XML

- Internet Explorer 5
- Netscape 4.0.4
- Mozilla 5 คาดว่าจะเป็น browser ที่สนับสนุน XML มากที่สุด

2. ข้อดีของ XML

- สามารถกำหนด tag ได้เอง ข้อมูล XML นั้นอธิบายตัวของมันเอง คือ เราจะกำหนดให้ tag สอดคล้องกับข้อมูลได้ ซึ่งต่างจาก HTMLตรงที่ tag จะกำหนดว่าให้แสดงข้อมูลอย่างไร เช่น

  <name> gabriel batistuta </name>

  <digital> 100 </digital>

  <pascal> 90 </pascal>

  <java> 85 </java>

  ถ้าเราอ่าน XML นี้ เราก็พอจะเข้าใจว่านี่คือคะแนนการสอบของใคร ได้คะแนนเท่าไหร่ในแต่ละวิชา

  แต่เราไม่สามารถทำเช่นนี้ได้ใน HTML เนื่องจาก tag ทั้งหมดในตัวอย่างนี้ไม่มีความหมายใน HTML
- XML แยกส่วนของข้อมูลออกจากส่วนของการแสดงผล หากเราต้องการเปลี่ยนรูปแบบการแสดงผล เราก็เพียงเปลี่ยน stylesheet ในขณะที่เราสามารถใช้ข้อมูลชุดเดิมได้เลย ซึ่ง HTML ไม่สามารถทำได้
- ข้อมูล XML 1 ไฟล์ สามารถใช้ได้กับแอพพลิเคชันหลายตัว เพราะว่าเป็นไฟล์ text แบบ ASCII เพียงแต่ว่าแอพพลิเคชันจะต้องเข้าใจโครงสร้างของ XML นั้นๆเท่านั้น เช่น

  <name> terrance mathis </name>

  <age> 30 </age>

  <birthday> 5/5/69 </birthday>

  <address> 3124 kirkwood dr. altanta,GA </address>

  <profession> professional footballer </profession>

  เราอาจจะมีแอพพลิเคชันตัวหนึ่งที่ใช้พิมพ์ซองจดหมาย โดยแอพพลิเคชันตัวนั้นก็จะเลือกข้อมูลในส่วน name และ address มาใช้ ในขณะเดียวกันแอพพลิเคชันอีกตัวอาจจะใช้ในการค้นหาว่ามีใครอายุ 30 โดยจะไปหาที่ age แล้วก็แสดงผลว่ามี ใครบ้างโดยใช้ข้อมูลจาก name เป็นต้น
- สามารถเขียน,แก้ไขและอ่านได้โดยโปรแกรมแทบทุกตัว เพราะว่าเป็น text file

- สามารถนำไปใช้ในแต่ละสาขาวิชาได้อย่างเหมาะสม เพราะว่าสามารถสร้าง tag ที่จำเป็นได้เอง เพียงแต่เขียนแอพพลิเคชันในงานด้านนั้นๆก็สามารถทำงานได้แล้ว
- สามารถใช้ได้กับทุก แพลตฟอร์ม

จ. JDBC (Java Database Connectivity)

JDBC (Java Database Connectivity) ถูกพัฒนาโดย JavaSoft Department ของบริษัท Sun Microsystems ซึ่งก็คือฟังก์ชันมาตรฐาน หรือ Java Application Programming Interface (API) สำหรับการเชื่อมต่อกับระบบฐานข้อมูล นักพัฒนาสามารถใช้ JDBC API และยังประกอบด้วยแพ็กเก็จอื่นๆด้วย ซึ่งนำเสนอในรูปแบบฟังก์ชันพิเศษ โดยทั่วไปการใช้ SQL database ในการติดต่อกับเฟรมเวิร์ค (framework) เพื่อพัฒนามาตรฐานในการติดต่อส่วนบนสุดของชนิดต่างๆ ของ database connectivity modules ซึ่งก็คือมาตรฐานของ ANSI SQL-2 Entry level database เพราะว่า relational database เกือบทั้งหมดในปัจจุบันใช้มาตรฐานของ SQL-2 Entry level

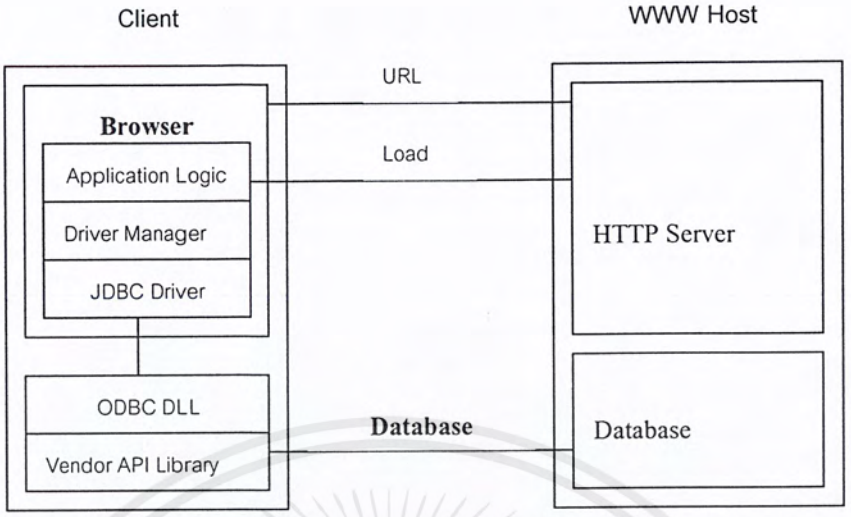
JDBC สร้างระดับการเชื่อมต่อเพื่อการสื่อสารกับฐานข้อมูลในรูปแบบที่คล้ายคลึงกับ ODBC (Open Database Connectivity ของบริษัทไมโครซอฟท์) หลักการทำงานของทั้ง JDBC และ ODBC ตั้งอยู่บนมาตรฐานเดียวกันคือ X/Open SQL Call-Level Interface ของระบบ X Window

โครงสร้างของJDBC

โครงสร้างของการเชื่อมต่อภายใน JDBC ประกอบด้วย 3 ระดับหลัก คือ JDBC API, JDBC Driver API และ JDBC Driver ดังรูป ระดับบนสุด JDBC API เป็นระดับฟังก์ชัน API ที่อำนวยความสะดวกให้แก่โปรแกรมประยุกต์ ระดับล่าง JDBC Driver (มีไดร์ฟเวอร์ที่ต่างกันอยู่ 4ช นิด) รายละเอียดการทำงานของไดร์ฟเวอร์แต่ละชนิดอธิบายได้ดังนี้

1. JDBC/ODBC bridge

JDBC/ODBC bridge ทำหน้าที่เป็นตัวกลางในการเข้าถึงฐานข้อมูล ได้โดยผ่านการทำงานของ ODBC โดยนำข้อดีของ ODBC-enabled data sources ที่มีใช้อยู่โดยทั่วไปอย่างมากมาย ฝั่งไคลเอนท์ จาวาแอปเพล็ตหรือจาวาแอปพลิเคชั่น จะถูกเขียนโดยใช้ JDBC API บริดจ์จะทำการแปลงโดยการเรียกใช้ JDBC ไปยัง ODBC และส่งค่า ODBC Driver ที่เหมาะสมสำหรับ back-end database



รูปที่ จ-1 การติดต่อ JDBC โดยรูปแบบที่1 JDBC-ODBC Bridge

ข้อดีของบริดจ์ ทำให้แอปพลิเคชั่นสามารถติดต่อกับฐานข้อมูลได้อย่างง่ายดาย โดยจากผู้ผลิตที่มีมากมาย โดยสามารถเลือก ODBC Driver ที่เหมาะสมอย่างไรก็ตามการติดต่อฐานข้อมูลประเภทนี้ต้องพิจารณาค่าใช้จ่าย (overhead) และความซ้ำซ้อน (complexity) เพราะว่าการเรียกใช้จะมีลำดับดังนี้คือ

JDBC ⟷ bridge ⟷ ODBC ⟷ Native client – API ⟷ database

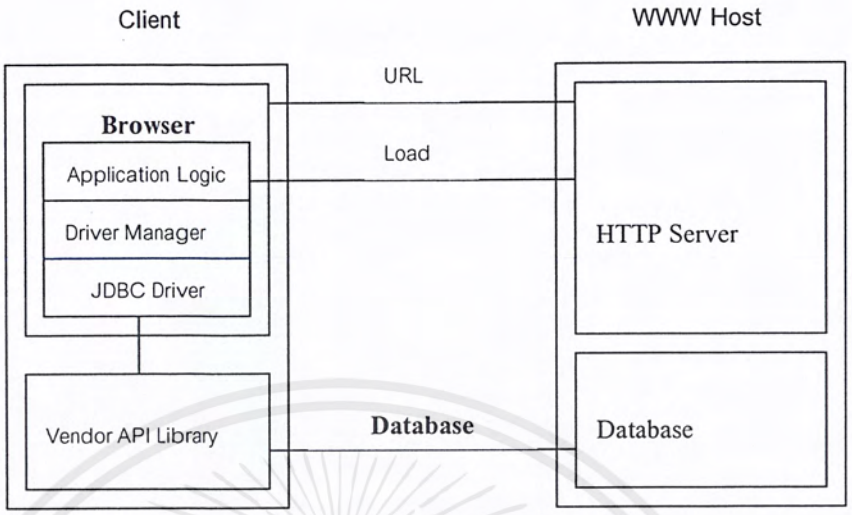
2. Native-API, partly Java driver

Native-API, partly Java driver ใช้ Vendor Library ในการแปลง JDBC function ไปยังคุณลักษณะของการใช้ภาษาไทยในการ Query เช่น ไลบรารีสำหรับออราเคิล คือ ocilib การติดต่อ JDBC ประเภทนี้จะเหมือนกับ JDBC/ODBC Bridge คือต้องติดตั้งโค้ดในเครื่องไคลเอนต์

ข้อดีของไดร์ฟเวอร์ชนิดนี้ คือ สามารถใช้ Native-API Driver ติดต่อกับฐานข้อมูลนั้นโดยตรง โดยผ่านโพรโตคอลเดิมที่ใช้อยู่ก่อนแล้ว ทำให้เหมาะสมกับการเชื่อมต่อฐานข้อมูลแบบ Two-tier Client Server นอกจากนี้ยังเร็วกว่า ไดร์ฟเวอร์แบบแรก เพราะว่ามีเลเยอร์พิเศษของการแปลงเป็น ODBC ถูกจำกัดออกไป

JDBC ⟷ Native client-API ⟷ database

แต่มีบางส่วนของ Native-API Driver ถูกเขียนจากภาษา C++ (Partly-Java) จึงไม่สามารถดาวน์โหลดผ่านเครือข่ายอินเทอร์เน็ตได้ และไม่สามารถเชื่อมต่อฐานข้อมูลข้ามชนิดกันได้

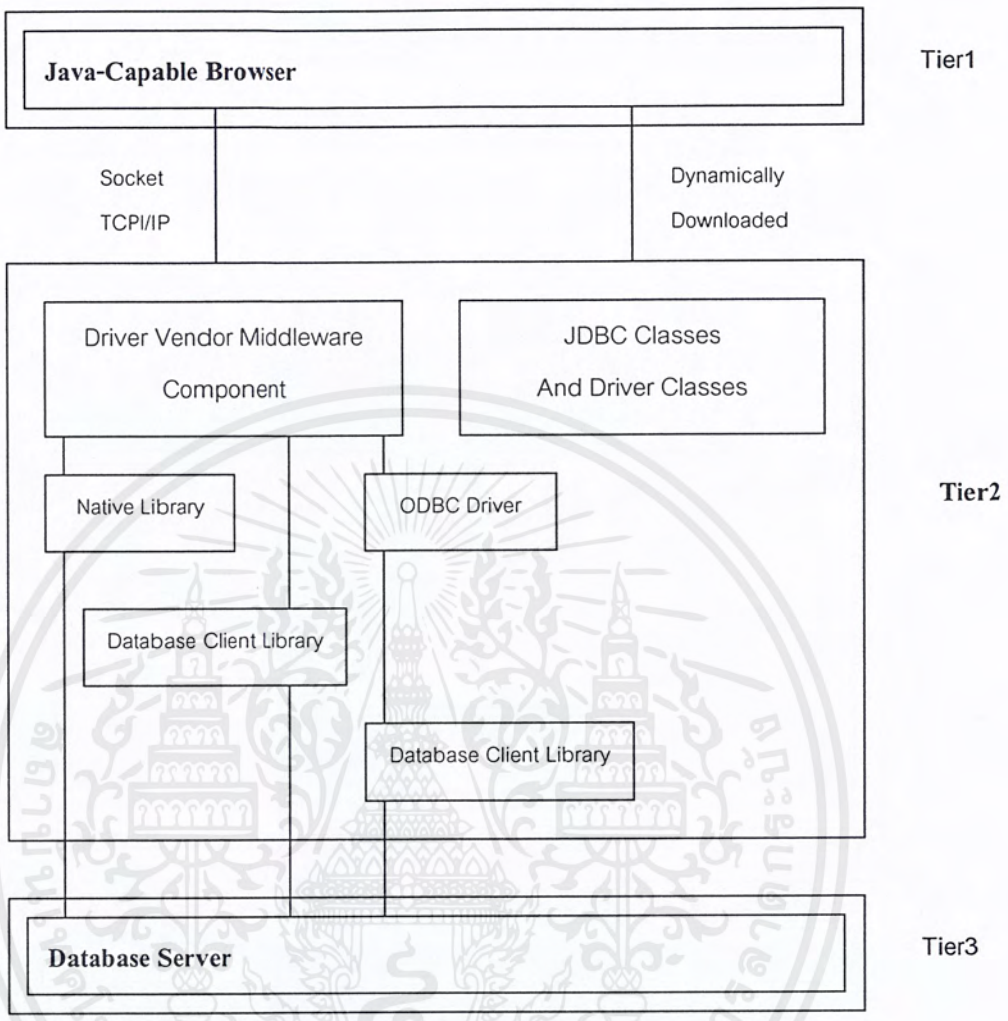


รูปที่ จ-2 การติดต่อ JDBC โดยรูปแบบที่2 Native-API, partly Java driver

3. Network-protocol, all-Java driver

Network-protocol, all-Java driver จะทำการแปลงการเรียกใช้ JDBC ให้อยู่ในรูปของเน็ตเวิร์คโพรโตคอลร่วม (DBMS-independent network Protocol) ซึ่งหลังจากนั้นจะถูกแปลงให้อยู่ในรูปเฉพาะของแต่ละฐานข้อมูล (database-specific API) บนเซิร์ฟเวอร์นั้นๆ รูปแบบการเชื่อมต่อจะเป็นลักษณะของ three tier ไดร์ฟเวอร์ชนิดนี้จะทำการเอ็กซิคิวท์บนไคลเอนต์ และส่งคำสั่ง SQL ไปยังเน็ตเวิร์ค เมื่อเซิร์ฟเวอร์ได้รับข้อมูล ก็จะจัดการเชื่อมต่อที่มีมาไปยังฐานข้อมูล

ไดร์ฟเวอร์ประเภทนี้เหมาะสมมากสำหรับ ระบบเครือข่ายอินเทอร์เน็ต-อินทราเน็ต และการทำงานที่มีผู้ใช้หลายคน เพราะมีความยืดหยุ่นคล่องตัวที่สุด เพราะเขียนขึ้นจากภาษาจาวาทั้งหมด ทำให้ไม่จำเป็นต้องมีไดร์ฟเวอร์ร่วมที่เขียนจากภาษาอื่น ซึ่งต้องติดตั้งเฉพาะฝั่งไคลเอนต์เท่านั้น และสามารถรันบนระบบใดก็ได้ที่สนับสนุนสภาพแวดล้อมเสมือนของจาวาหรือ JVM (Java Virtual Machine)



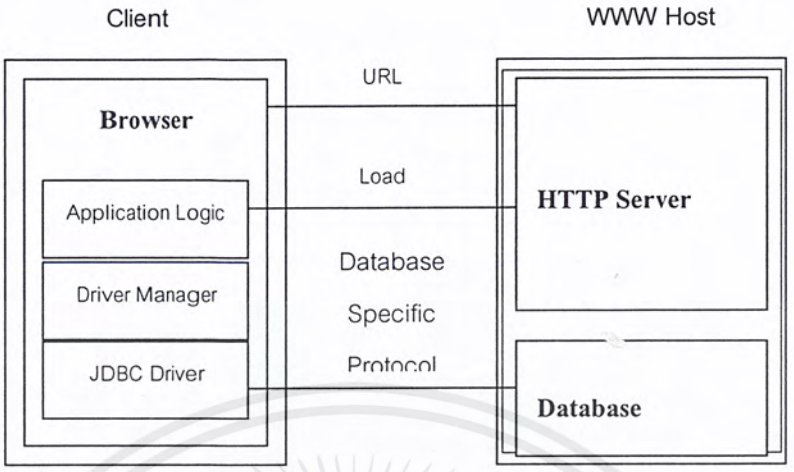
รูปที่ จ-3 รูปแสดงการติดต่อ JDBC แบบที่3 Network - protocol

4. Native-protocol, all-Java driver

Native-protocol, all-Java driver จะแปลงการเรียกใช้คำสั่งของ JDBC ให้อยู่ในรูปแบบของเน็ตเวิร์คโพรโตคอลเฉพาะของฐานข้อมูลนั้นโดยตรง เพราะใช้ไดร์ฟเวอร์ของตัวแทนจำหน่ายฐานข้อมูล ไดร์ฟเวอร์เหล่านี้สามารถเขียนในภาษาจาวา และติดต่อกับแอปเพล็ตแบบ just-in-time เพราะว่าไดร์ฟเวอร์เหล่านี้จะแปลง JDBC ตรงไปยัง native protocol โดยปราศจากการใช้ ODBC หรือ Native APIs ซึ่งสามารถจัดหามาสำหรับการติดต่อฐานข้อมูลที่มีประสิทธิภาพสูง

ข้อดีของไดร์ฟเวอร์ชนิดนี้คือ ไม่ต้องมีการปรับเปลี่ยนระบบฐานข้อมูลเดิมที่ใช้งานอยู่แล้วในแต่ละองค์กรและไม่มีความจำเป็นต้องติดตั้งไดร์ฟเวอร์ตัวกลาง

JDBC Driver ชนิดที่ 3 และ 4 คือไดร์ฟเวอร์ที่คาดว่าจะเป็นสิ่งที่ต้องการสำหรับการเชื่อมต่อกับฐานข้อมูลในอนาคต เพราะถูกเขียนขึ้นจากภาษาจาวาทั้งหมดซึ่งมีความปลอดภัยและคล่องตัวมากกว่า



รูปที่ จ-4การติดต่อ JDBC โดยรูปแบบที่4 Native-protocol, all-Java

ออราเคิลมี JDBC Driver 2 ประเภท คือ

- JDBC Thin เป็น JDBC Driver ชนิดที่4 ซึ่งใช้ซ็อกเก็ต (socket)เชื่อมต่อโดยตรงกับออราเคิลโดยผ่านโพรโตคอล TCP/IP การติดต่อทำโดยภาษาจาวา ได้แก่ จาวาแอปเพล็ตและจาวาแอปพลิเคชั่น ทำให้ไดร์ฟเวอร์ชันนี้มีคุณสมบัติคือ ไม่ขึ้นกับแพลตฟอร์ม (platform-independent)
- JDBC OCI เป็น JDBC Driver ชนิดที่2 ไดร์ฟเวอร์ชนิดนี้ใช้กับจาวาแอปพลิเคชั่น

ฉ. จาวาเซิร์ฟเล็ต (Java Servlet)

เซิร์ฟเล็ตมการทำงานแบบซีจีไอ โดยทำงานบนจาวาเวอร์ชวลแมชชีน (JVM) บนเซิร์ฟเวอร์ ซ่งต่างจากจาวาแอปเพล็ต คือ บราวเซอร์ไม่จำเป็นต้องสนับสนุนการใช้งาน และต่างจากซีจีไอตรงที่ซีจีไอจะใช้ Multiple-process เพื่อจัดการกับโปรแกรมหลายๆ โปรแกรมและรีเควสหลายๆครั้ง ส่วนเซิร์ฟเล็ต จะจัดการโดยแยกเป็นหลายๆเธรด ในการประมวลผลของเว็บเซิร์ฟเวอร์

1. ความสามารถของเซิร์ฟเล็ต

- สามารถทำงานข้ามแพลตฟอร์มได้ เซิร์ฟเล็ตเป็นภาษาจาวา ซึ่งมีความสามารถนี้ ดังนั้นเราอาจจะพัฒนาเซิร์ฟเล็ตบนวินโดว์ แล้วนำโปรแกรมไปใช้บนยูนิกซ์ก็ได้
- ความสามารถของจาวา เซิร์ฟเล็ตจะสามารถใส่ความสามารถของจาวาได้อย่างเต็มที่ เช่นการเข้าถึงระบบเน็ตเวิร์ก และ URL มัลติเธรด การเชื่อมต่อกับฐานข้อมูล การใช้ Remote Method Invocation (RMI)

- ความมีประสิทธิภาพ เซิร์ฟเล็ตจะทำงานได้อย่างมีประสิทธิภาพ เมื่อโหลดเซิร์ฟเล็ตขึ้นมาในหน่วยความจำในลักษณะของ single object instance เซิร์ฟเล็ตจะจัดการการร้องขอพร้อมๆกันได้ โดยใช้เธรด และสามารถคงสถานะเดิมของมันได้ เช่น การเชื่อมต่อกับฐานข้อมูล เพราะตัวของเซิร์ฟเล็ตเองจะอยู่ในหน่วยความจำของเซิร์ฟเวอร์
- ความปลอดภัย ซึ่งจะมีการรักษาความปลอดภัยที่เป็นคุณสมบัติที่สืบทอดมาจากคุณลักษณะของจาวา และยังมีคุณสมบัติในตัวมันเองของ API ด้วย คือมีพอยเตอ์ในการดูแลหน่วยความจำ และมีการรองรับความผิดพลาดต่างๆ ที่อาจเกิดขึ้นโดยใช้ Exception-handling mechanism ซึ่งจะ throw exception โดยไม่ทำให้ระบบเสียหาย
- ความสะดวกในการพัฒนา สามารถเรียก Servlet API มาใช้ได้ โดยมันจะประกอบไปด้วยคลาสต่างๆที่ช่วยในการทำงานของเซิร์ฟเล็ต ทำให้พัฒนาโปรแกรมได้ง่ายขึ้น
- ความยืดหยุ่น Servlet API ถูกออกแบบมาเพื่อรองรับการขยายงานในรูปแบบต่างๆ ได้ง่าย เช่นในปัจจุบันนี้ มีการพัฒนาเพื่อรองรับการทำงานสำหรับ HTTP Servlet

2. Http Servlet

พื้นฐานของ Http Requests, Response และ Headers นั้น การทำงานของโพรโตคอล Http นั้น ฝั่งไคลเอนท์ (ในที่นี้คือเว็บบราวเซอร์) จะทำการรีเควส และเว็บเซิร์ฟเวอร์ก็จะตอบสนองกลับมา ในการส่งรีเควสของไคลเอนท์ สิ่งแรกที่ต้องระบุถึงคือ Http command ซึ่งเป็นเมธอดที่จะบอกเซิร์ฟเวอร์ถึงแอคชันที่มันจะทำ บรรทัดแรกของการรีเควสจะกำหนดแอดเดรส และเวอร์ชันของโปรโตคอล Http ดังตัวอย่าง

Get/intro.html HTTP/1.0

ในการรีเควสนี้จะใช้เมธอด GET เพื่อร้องขอ intro.html โดยใช้ Http หลังจากรีเควสแล้วไคลเอ็นท์อาจส่งข้อมูลอื่นๆเช่น ซอฟท์แวร์ที่ไคลเอนท์ทำงานอยู่ ดังตัวอย่าง

User-Agent:Mazilla/4.0 (compatible; MSIE4.0;Windows95)

หลังจากได้รับรีเควสของไคลเอนท์แล้ว เซิร์ฟเวอร์จะทำการประมวลผลและส่งผลตอบสนองกลับไป บรรทัดแรกของการตอบสนองจะแสดงสถานะ (status line) ที่ระบุเวอร์ชันของโปรโตคอล HTTP เช่น HTTP/1.0 200 OK

หลังจากรับ status line แล้ว เซิร์ฟเวอร์จะส่งผลตอบสนองเพื่อบอกรายละเอียดเกี่ยวกับตัวมัน ดังตัวอย่าง

Date	:Saturday,23-May-98 03:25:12 GMT
Server	:JavaWebServer/1.1.1
MIME-version	:1.0
Content-type	:text/html
Content-length	:1029
Last-modified	:Thursday, 7-May-98 12:15:35

3. GET & POST METHOD

เมื่อไคลเอนท์ติดต่อกับเซิร์ฟเวอร์และทำ HTTP request รูปแบบของการร้องขอมีหลายรูปแบบ ที่มักจะใช้อยู่เป็นประจำคือเมธอด GET และ POST

เซิร์ฟเล็ตจะใช้คลาสและอินเตอร์เฟสจาก 2 แพ็คเก็จ คือ javax.servler และ javax.servlet.http การเขียนเซิร์ฟเล็ตต้องอิมพลีเมนท์อินเตอร์เฟส javax.servlet.Service หรือ javax.servlet.http.HttpServlet ในแต่ละครั้งที่เซิร์ฟเวอร์ส่งผ่านการร้องของ (request) ไปยังเซิร์ฟเล็ตมันจะไปเรียกเมธอด service() ของเซิร์ฟเวอร์

พื้นฐานอย่างหนึ่งของ HTTP เซิร์ฟเล็ต คือการแสดงหน้าเว็บเพจ HTML ซึ่งมันจะสามารถทำงานได้เหมือนกับซีจีไอทั้งด้านการทำฟอร์ม HTML หรือการติดต่อกับฐานข้อมูล คือตัวอย่างจะเป็นการใช้เซิร์ฟเล็ตเขียนโปรแกรมให้แสดงออกมาในรูปแบบของ HTML

ตัวอย่างโปรแกรม Hello World

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello World extends HttpServlet
{
        public void doGet(HttpServlet req,HttpServletResponse res)
        throws ServletException,IOException
        {
                res.setContentype("tetxt/html");
                PrintWriter out = rers.getWriter();
                out.println("<HTML>");
                out.println("<HEAD><TITLE> Hello World </TITLE></HEAD>");
                out.println("<BODY>");
                out.println("<BIG> Hello World <BIG>");
                out.println("</BODY></HTML>");
        }
}
```

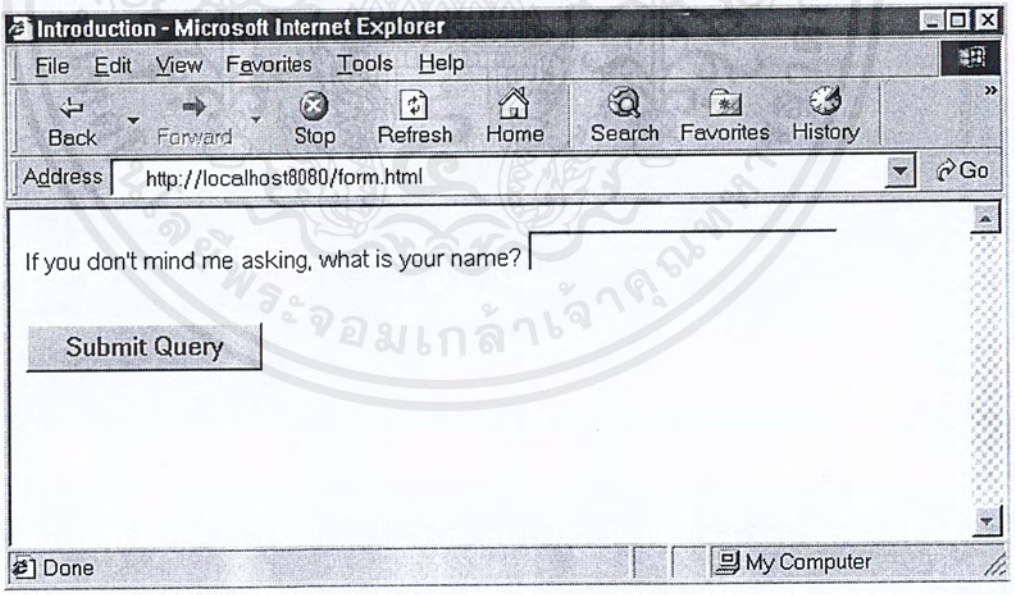
เซิร์ฟเล็ตจะเอ็กซ์เทนด์คลาส HttpServlet และโอเวอร์โหลดเมธอด doget() แต่ละครั้งที่เซิร์ฟเวอร์ได้รับรีเควสแบบ GET แล้วเซิร์ฟเวอร์จะเรียกเมธอด doget() เพื่อผ่านค่าออบเจ็ค HttpServletRequest และ HttpServletResponse

HttpServletRequest จะรองรับการร้องขอต่างๆของไคลเอ็นท์โดยออบเจ็คนี้จะเข้าถึงข้อมูลเกี่ยวกับไคลเอนท์และยังสามารถใช้ออบเจ็คเพื่อการเซ็ต Httpresponse header ด้วย

เมธอดที่ตัวอย่างโปรแกรมใช้ตอนแรก คือเมธอด setContent Type() ของออบเจ็ค response เพื่อดึงค่าของ content type ของการตอบสนองที่เป็น "text/html" และใช้เมธอด getWriter() เพื่อเก็บค่าของ PrintWriter ในการที่จะส่ง "Hello World" ในรูปแบบของ HTML ไปยังไคลเอนท์

เซิร์ฟเล็ตจะสามารถทำการสร้าง (generate) รูปแบบตามภาษาเอชทีเอ็มแอลได้ โดยจะมีค่าดังต่อไปนี้

```
<HTML>
    <HEAD>
        <TITLE> Introduction </TITLE>
    </HEAD>
    <BODY>
        <FORM METHOD = GET ACTION = "/servlet/Hello">
        If you don't mind me asking, what is your name?
        <INPUT TYPE = TEXT NAME = "name"><P>
        <INPUT TYPE = SUBMIT>
        </FORM>
    </BODY>
</HTML>
```



รูปที่ ฉ-1 แสดงการใช้เซิร์ฟเล็ตในการสร้างหน้าจอออกมา

จากแบบฟอร์มนี้เมื่อผู้ใช้ทำการกรอกชื่อลงไปในเท็กซ์ฟิลด์แล้ว ทำการกดปุ่ม Simit Query แล้วจะเกิด action ขึ้น คือชื่อของผู้กรอกจะถูกส่งไปที่ Hello Servlet และจะไปทำการเรียก method GET ซึ่งข้อมูลที่ถูกส่งไปจะส่งไปในรูปแบบต่อไปนี้

http://server:880/servlet/Hello?name=Automatic+indexing

ตัวอย่างโปรแกรม Hello

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello extends HttpServlet
{
        public void doGet(HttpServletRequest req,HttpServletResponse res)
        throws ServletException,IOException
        {
                res.setContenType("text/html");
                PringlnWriter out.res.getWriter()'
                String name = req.getParameter("name");
                out.println("<HTML>");
                out.println("<HEAD><TITLE> Hello,"+name+"</TITLE></HEAD>");
                out.println("</BODY></HTML>");
        }
}
```

เมธอด req.getParameter("name") จะใช้เป็นตัวแปรรับค่า name ของผู้ใช้

ตัวอย่างเมธอด POST จะมีรูปแบบดังนี้

```
public void doPost(HttpServletRequest req,HttpervletResponse res)
throws ServletException,IOException
{
}
```

โดยในฟอร์มของการส่ง HTML นั้นจะอยู่ในรูปแบบ

```
<FORM METHOD = POST ACTION = "/servlet/Hello">
```

ช. ภาษาเอสคิวแอล (SQL : Structured Query Language)

ภาษาเอสคิวแอลเป็นภาษายุคที่ 4 ภาษาหนึ่งซึ่งพัฒนาโดยบริษัทไอบีเอ็ม (IBM) ในปัจจุบันเป็นที่แพร่หลายกันมาก เป็นภาษาที่คล้ายกับภาษาอังกฤษ ใช้ในการปฏิบัติงานและควบคุมฐานข้อมูลเป็นภาษาที่คล้ายกับภาษาอังกฤษ ในการประมวลผลข้อมูลหรือมีเพียงเล็กน้อย ก็สามารถที่จะเรียนรู้โครงสร้างพื้นฐานของภาษาเอสคิวแอลได้อย่างรวดเร็ว และสำหรับผู้ที่อยู่ในระดับที่มีความรู้ด้านนี้สูง ก็จะพบว่าภาษาเอสคิวแอลนั้น จะให้คำสั่งซึ่งมีความสามารถ และมีความสมบูรณ์ในตัวในการดำเนินงานได้อย่างดี

ดังนั้นภาษาเอสคิวแอลจึงกลายเป็นภาษาร่วมกันระหว่างผู้ใช้งานธรรมดา กับผู้ที่มีความรู้ประสบการณ์ในการประมวลผล แต่สำหรับผู้ใช้ทั่วไปแล้ว คงเป็นการลำบากที่จะใช้ภาษาเอสคิวแอล ในการสร้างคำถามที่ซับซ้อนได้ ดังนั้น ตามความเป็นจริงแล้ว ในปัจจุบันภาษาเอสคิวแอบใช้โดยบุคคลในวงการระดับที่มีความรู้และประสบการณ์ในการประมวลผล, นักพัฒนาระบบงาน, นักบริหารข้อมูล (DBA : Database Administrator), ระดับผู้บริหาร และทีมงานนักสารนิเทศ

ภาษาเอสคิวแอลเป็นภาษาฐานข้อมูลแบบเชิงสัมพันธ์ (Relational Database) คือ ประกอบด้วยตาราง (Table) หลายตาราง และในตารางหนึ่งๆจะมี 2 มิติ ได้แก หลัก (columns) ในแนวตั้งและแถว (rows) ในแนวนอน

การใช้ภาษาเอสคิวแอลกระทำได้ 3 วิธี ได้แก่

1. ออกคำสั่งแบบออนไลน์ กล่าวคือ ผู้ใช้สามารถพิมพ์ประโยคคำสั่งผ่านทางเทอร์มินัล (Terminal) โดยที่คำสั่งเหล่านี้จะถูกปฏิบัติงานโดยทันที
2. ส่งคำสั่งในลักษณะงานออฟไลน์ (Off Line) หรืองาน batch ลักษณะการใช้งานประเภทนี้เหมาะกับการสร้างรายงาน หรือประเภทของงานที่ไม่จำเป็นต้องการทราบผลโดยทันที
3. สอดแทรกประโยคคำสั่งไว้ในโปรแกรมประยุกต์ที่เขียนขึ้นมาสำหรับการใช้ระบบฐานข้อมูล ซึ่งโปรแกรมประยุกต์เหล่านี้อาจจะเขียนด้วยภาษา โคบอล, ฟอร์แทรน, ภาษาซี, ภาษาจาวา ฯลฯ ก็ได้

ภาษาเอสคิวแอลมีประเภทคำสั่งโดยสรุปดังนี้

- การคิวรี (Query) เป็นการสอบถามข้อมูลจากฐานข้อมูล
- การดำเนินงานกับข้อมูล (Data Manipulation) ได้แก่ การเพิ่มเติม (insert), การลบ (delete) และการแก้ไข (update) ข้อมูลในฐานข้อมูล
- การกำหนดลักษณะของข้อมูล (Data Definition) ได้แก่ การกำหนดตาราง (tables), วิว (views) และดัชนีในการค้นหา (indexes) ในฐานข้อมูล
- การควบคุมข้อมูล (Data Control) ได้แก่ การป้องกัน ควบคุมข้อมูลให้ปลอดภัย จากผู้ใช้แต่ละคน

รูปแบบของคำสั่งเอสคิวแอลพื้นฐานโดยทั่วไป

Select ...

From (ชื่อตาราง)

Where (เงื่อนไข)

Order By ...

ซึ่งผู้ใช้สามารถระบุสิ่งต่างๆในคำสั่งได้ดังนี้

- ส่วนหลัง Select ใช้ระบุคอมลัมน์หรือกลุ่มของคอลัมน์ที่เราต้องการดูข้อมูล
- ส่วนหลัง From ใช้ระบุชื่อของตารางที่เราต้องการดูข้อมูล
- ส่วนหลัง Where ใช้ระบุเงื่อไขของข้อมูลที่เราสนใจซึ่งอาจจะมีหรือไม่มีก็ได้
- ส่วนหลัง Order by ใช้ระบุวิธีการเรียงลำดับการแสดงผลข้อมูลซึ่งอาจจะมีหรือไม่มีก็ได้

ตัวอย่างง่ายๆของภาษาเอสคิวแอล เช่น

Select *

From library;

จากคำสั่งนี้จะทำการแสดงผลข้อมูลของทุก rows และทุก columns ที่มีอยู่ในตารางชื่อ library

บรรณานุกรม

[1] Cairat Panpun : "*Analysis and Design of a System For Creation of Virtual Libraries*" [IEEE]

[2] Hideahi Kikuchi, Yuseke Mishina, Minoru Ashizawa, Noako Yamazaki, and Hiromichi Fujisawa :*"ผู้ใช้ interface for a digital library to support construction of a "Virtual personal library"* " [IEEE]

[3] Floriana Esposito, Donato Malerba, Giovanni Semeraro, Cesare Daniele Antifora, and Gioacchino de Gennaro : "*Information Capture and Semantic Indexing of Digital Libraries through Machine Learning Techniques*" [IEEE]

[4] Jose Luis Borbinha, Joal Ferreira, Joaquim Jorge, and Jose Delgado : "*Digital Library for a Virtual Organization*" [IEEE]

[5] ดร.น้ำทิพย์ วิภาวิน, อ.อุบล ทุติยะโพธิ, อ.สุกัญญา มกุฎอรถดี, ดร.สุนีย์ กาศจำรูญ, รศ.ดร.สุชาย ธนวเสถียร, อ.ธวัส จัตุรัส, อ.บุบผา เทวาหุดี, อ.นงนารถ ชัยรัตน์, รศ.ดร.พิมลพรรณ เรพเพอร์, อ. พรทิพย์ สุวันทรัตน์, อ.ปรียาพร ฤกษ์พินัย, อ.เพ็ญพิมล เชี่ยวนาวิน : "*ห้องสมุดยุคใหม่กับไอที (Library Automation & Digital Library)*" [SUM Publishing]

[6] Frederick Stielow : " *Creating a Virtual Library*" [Neal-Schuman Publisher]

[7] Kurt D. Bollacker, Steve Lawerence, and C. Lee Giles, NEC Research Institute (1996): "*Discovering Relevant Scientific Literature On The Web*" [IEEE]

[8] Willpower Information : "*Thesaurus principles and practice*"

[9] Cavan Mccarthy : "*The Virtual Library : Serving Society During The Coming Millenium*" [http://mingo.info-science.uiowa.edu/mccarty/virlibfull.html]

[10] Larry S. Bonura : "*The Art of Indexing*" [สำนักพิมพ์ WILEY]

[11] W.BruceCroff, Bella Hass Weinberg : "*Indexing – The state of our Knowledge and the state of our Ignorance*" [Learned Information, Inc]

[12] Hans H. Wellisch (1991) : *"Indexing from A to Z 2nd edition"* H.W. Wilson

[13] H.S. Heaps : "*Information Retrieval Computational and The Oretical Aspects*" [ACADEMIC PRESS Publisher]

[14] Chung-hsin Lin and Hsinchun Chen (1996) : *"An Automatic Indexing and Neural Network Approach to Concept Retrieval and Classification of Multilingual (Chinese-English) Documents"* [IEEE], 1996

[15] Robert D. Stueart (1997) : *"The Virtual Library and the Future of Scholarly Communications"* Asian Institute of Technology, 1997

[16] Didem Gokcay, Erhan Gokcay (1995) : *"Generating Titles for Paragraphs Using Statistically Extracted Keywords and Phrases"* , IEEE, 1995

[17] http://www.sunsite.berkeley.edu/ARL/definition.html : *"Definition and Purpose of a Digital Library"* , 2000

[18] http://www.www.csdl.tamu.edu(2000): *"The center for the Study of Digital Libraries A first step Toward Communication in Virtual Libraries Providing Social Interaction in the Digital Library"* ,2000

[19] http://www.clir.org/diglib/architectures/lycospub.htm (2000): *"Harvesting research metadata, Aims, objectives, planning process, Licensing Digital Information"*

[20] Jose Luis Borbinha (Lisbon Technical University), Joaquim Jorge (Lisbon Techinical University) (1998): *"A digital Library for a Virtua Organization"* IEEE, 1998

[21] H.Chen, P.Hsu, R. orwig, L. Hoopes, J.F. Nunamaker (2000) : *"Automatic Concept Classification of Text from Electronic Meetings"*

[22] Jason Hunter(1998), *"Java Servlet Programming"* O'REILLY,1998