สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

แว็พบราวเซอร์ภาษาไทย 2

Thai WAP Browser2

นายธีรยุทธ พิทักษ์พงศ์พนา

นายปวัตร สันติบุตร

เลขหมู่.....
เลขทะเบียน.....46153
วัน, เดือน, ปี 20 ส.ค. 2546

.b.....
.i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

แว็พบราวเซอร์ภาษาไทย 2

Thai WAP Browser2

โดย

นายธีรยุทธ พิทักษ์พงศ์พนา

นายปวัตร สันติบุตร

อาจารย์ที่ปรึกษา

ดร.วิศิษฏ์ หิรัญกิตติ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง แว็พบราวเซอร์ภาษาไทย 2

Thai WAP Browser 2

ผู้จัดทำ

1. นาย ธีรยุทธ พิทักษ์พงศ์พนา รหัสประจำตัว 41014193

2. นาย ปวัตร สันติบุตร รหัสประจำตัว 41014265

อาจารย์ที่ปรึกษา

(ดร. วิศิษฏ์ หิรัญกิตติ)

แว็พบราวเซอร์ภาษาไทย 2

นายธีรยุทธ พิทักษ์พงศ์พนา รหัส 41014193
นายปวัตร สันติบุตร รหัส 41014265
ดร. วิศิษฏ์ หิรัญกิตติ อาจารย์ที่ปรึกษา
ปีการศึกษา 2544

บทคัดย่อ

ปริญญานิพนธ์นี้ ได้นำเสนอการออกแบบและพัฒนาแว็พบราวเซอร์ภาษาไทยซึ่งทำหน้าที่เป็นอิมูเลเตอร์คือเสมือนแว็พบราวเซอร์ที่ใช้งานบนเครื่องพีซี ตัวบราวเซอร์นี้พัฒนาโดยภาษาจาวาทำให้สามารถใช้ได้บนทุกแพลตฟอร์มมีความสามารถในการแสดงแท็กหลักๆของภาษา WML ในส่วนการกำหนดรูปแบบตัวอักษรภาษาไทย ภาษาอังกฤษ การจัดตำแหน่งของข้อความในหน้าแว็พเพจ การแสดงรูปภาพสี การแสดงตารางและการเชื่อมลิงค์แบบต่างๆสำหรับแท็กที่บราวเซอร์ไม่สามารถแสดงผลได้คือแท็กที่จัดการเกี่ยวกับเหตุการณ์ที่สามารถกำหนดโดยผู้เขียนเอกสาร WML และการแสดงผล Form

วิธีการในการตีความหมายโค้ดของเอกสาร WML ให้ถือว่าเอกสาร WML เป็นเสมือนเอกสาร XML คือทำการพาร์สข้อมูลในเอกสารลงบรรจุในต้นไม้ เพื่อนำไปสู่การประมวลผลและการแสดงผลที่ถูกต้องโดยเฉพาะการแสดงผลภาษาไทย

Thai WAP Browser 2

Theerayuth Pitakpongpana

Pawat Santibhut

Dr.Visit Hirankitti Advisor

ABSTRACT

This project designs and develops a Thai WAP browser.This browser is a WAP browser emulator working on a PC rather than on a mobile phone.Since it is developed in Java, it is a cross-platform application. Our browser is able to handle major WML tags, i.e. tags for setting Thai and English font styles,tags for managing text alignments, tags for adding images, tables, and different types of links. The WML tags that our browser cannot handle are tags for specifying events, and tags for emanaging forms.

To be able to interpret and handle WML tags, the browser treats a WML document as if it were an XML document, it then processes the WML document using a normal XML parser, here we chose the DOM approach. The parser takes a WML document as its input and produces a parsed tree where WML tag data is parsed and stored as the elements of the tree.Using the information in the parsed tree, the browser can then analyse the meaning of each tag and handle it accordingly, e.g. displaying texts in the right fonts and styles, especially for Thai fonts.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลาย ๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ ก็คือ อาจารย์ วิศิษฏ์ หิรัญกิตติ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำ และช่วยเหลือเสมอมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

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

ธีรยุทธ พิทักษ์พงศ์พนา

ปวัตร สันติบุตร

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

สารบัญ

	หน้าที่
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญตาราง	VI
สารบัญรูปภาพ	VII
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญ	1
1.2 วัตถุประสงค์ของปริญญานิพนธ์	2
1.3 ขอบเขตของปริญญานิพนธ์	2
1.4 วิธีการดำเนินงาน	2
บทที่ 2 ความรู้เบื้องต้นเกี่ยวกับแว็พ	3
2.1 ความเป็นมาของแว็พ	3
2.2 ข้อกำหนดมาตรฐานแว็พ	4
2.3 การทำงานและโครงสร้างของแว็พ	7
2.3.1 สถาปัตยกรรมของแว็พ (WAP Architecture)	7
2.3.2 หลักการทำงานของแว็พ	11
2.3.3 โครงสร้างของแว็พเกตเวย์	13
2.3.4 หน้าที่หลักของ WAP Gateway	15
2.4 การกำหนด MIME Type เพื่อให้เซิร์ฟเวอร์สามารถรองรับแว็พได้	16
2.5 การเขียนโปรแกรมภาษา WML	17
2.5.1 โครงสร้าง ของเอกสาร WML	17
บทที่ 3 โพรโตคอลที่ใช้ในการสื่อสารทาง WAP	22
3.1 URL(Uniform Resource Locator)	22
3.2 HTTP Protocol	23
3.2.1 Initial Request Line	26
3.2.2 Initial Response Line	26
3.2.3 Header(s)	26
3.2.4 Message Body	27
3.3 การติดตั้ง Apache	27
บทที่ 4.XML Parser	29
4.1 XML	29

สารบัญ (ต่อ)

	หน้าที่
4.2 XML Parser	29
4.3 มองโครงสร้างเอกสารเป็นทรีด้วย DOM	29
4.3.1 มุมมองของ DOM	29
4.3.2 จุดเด่นของ DOM	31
4.3.3 จุดด้อยของ DOM	31
4.4 มองโครงสร้างเอกสารเป็นเส้นโครงด้วย SAX	31
4.4.1 จุดเด่นของ SAX	32
4.4.2 จุดด้อยของ SAX	32
4.5 เปรียบเทียบการทำงานระหว่าง DOM และ SAX	33
4.6 การติดตั้ง JDOM	35
4.6.1 การใช้ XML Parser	35
บทที่ 5 วิธีการเขียนโปรแกรมจัดการส่วนแสดงผล	38
5.1 ส่วนของ Xmlparser	38
5.2 ส่วนของ Processing	39
5.2.1 method Par()	40
5.2.2 method setStyle()	41
5.2.3 method setCo()	42
5.2.4 method show1()	42
5.3 สรุป TAG ที่สามารถทำได้ในโครงงานนี้	41
5.4 Problem Solving	44
5.4.1. ความผิดพลาดที่เกิดขึ้นกับ พาร์สทรี	44
5.5 สิ่งที่พัฒนาจากแว็พบราวเซอร์ภาษาไทยรุ่นก่อน	48
บทที่ 6 ผลการทดสอบ	49
6.1 การใช้ Browser ติดต่อกับแว็พไซต์	49
6.1.1 ส่วนแสดงเอกสาร WML	49
6.1.2 การแสดงผลบน Browser	50
6.2 การสร้าง เอกสาร WML โดยผู้ใช้	50
6.2.1 ส่วนแสดงเอกสาร WML	51
6.2.2 การแสดงผลบน Browser	51
6.3 รูปแบบการแสดงผลแบบต่าง ๆ	52
6.3.1 การเชื่อมโยง	52
6.3.2 แบบตัวอักษรภาษาอังกฤษ	55

สารบัญ (ต่อ)

	หน้าที่
6.3.3 แบบตัวอักษรภาษาไทย	55
6.3.4 เอกสารรูปแบบตาราง	56
6.4 ปัญหาที่เกิดขึ้นในการแสดงผลในแว็พบราวเซอร์ภาษาไทย	56
6.4.1 การแสดงผลของรูปภาพประเภท wbmp	56
บทที่ 7 บทวิจารณ์และสรุป	57
7.1 สรุปและวิเคราะห์การทดสอบ	57
7.2 แท็กที่สามารถรองรับได้	58
7.3 แนวทางการพัฒนาต่อ	58
ภาคผนวก ก	59
ภาคผนวก ข	62
บรรณานุกรม	68

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

สารบัญตาราง

	หน้าที่
ตารางที่ 2-1 การกำหนด MIME Type ให้กับเซิร์ฟเวอร์	17
ตารางที่ 2-2 แสดงแท็กต่าง ๆ ของภาษา WML	21
ตารางที่ 3-1 ตารางแสดง status code	26
ตารางที่ 5-1 ฟิลด์ทั่วไปของคลาส ElementInfo	40
ตารางที่ 5-2 ฟิลด์ที่เกี่ยวกับสไตล์ของฟอนต์ของคลาส ElementInfo	41
ตารางที่ 5-3 ฟิลด์ที่เกี่ยวกับค่าโคออดิเนตของคอนเทนต์ของคลาส ElementInfo	41
ตารางที่ 5-4 สรุป TAG ที่สามารถทำได้ในโครงงานนี้	43

สารบัญรูปภาพ

หน้าที่

รูปที่ 2-1 แบบจำลองแว็พสำหรับการเขียนโปรแกรม 4

รูปที่ 2-2 แสดงโครงสร้างจริงของเครือข่ายสื่อสารแว็พ 5

รูปที่ 2-3 สถาปัตยกรรมของแว็พ 7

รูปที่ 2-4 ระบบการรักษาความปลอดภัยของข้อมูลใน WAP และ อินเตอร์เน็ต 10

รูปที่ 2-5 ขั้นตอนการเข้ารหัสและถอดรหัสเพื่อรักษาความปลอดภัย ของข้อมูลทั้ง 2 ทิศทาง 11

รูปที่ 2-6 แสดงการใช้งานโปรแกรมแว็พบราวเวอร์ และเครื่องโทรศัพท์มือถือ ในการชมแว็พไซต์ 13

รูปที่ 2-7 โครงสร้างพื้นฐานของ WAP Gateway 14

รูปที่ 2-8 เด็ค และการ์ดของภาษา WML 18

รูปที่ 2-9 แสดงผลลัพธ์ของไฟล์เอกสาร 19

รูปที่ 3-1 เลเยอร์ที่ใช้ในการแลกเปลี่ยนข้อมูลผ่าน HTTP โปรโตคอล 23

รูปที่ 3-2 การส่งข้อมูลผ่าน HTTP โปรโตคอล 24

รูปที่ 3-3 การส่ง Request และ Responses message ระหว่าง client และ server 25

รูปที่ 3-4 HTTP Message 25

รูปที่ 4-1 ทำการพาร์สเซอร์โดยใช้โปรแกรม 36

รูปที่ 4-2 ทำการพาร์สเซอร์โดยใช้โปรแกรม เมื่อมี error ในการพาร์สเซอร์ 37

รูปที่ 5-1 ไดอะแกรมแสดงวิธีการจัดการส่วนแสดงผล 38

รูปที่ 5-2 ไดอะแกรมแสดงส่วน Xmlparser 38

รูปที่ 5.3 ทรีที่ได้จากการพาร์สซิ่ง 39

รูปที่ 5-4 ไดอะแกรมแสดงส่วน Processing 40

รูปที่ 5-5 ขนาดของหน้าจอที่ใช้ในโครงงาน 42

รูปที่ 5-6 ผลลัพธ์ที่ได้จาก test.wml 43

รูปที่ 5-7 พาร์สทรีของการพาร์สที่ไม่เห็นความแตกต่างระหว่างอีลิเมนต์ 2 ตัว 44

รูปที่ 5-8 พาร์สทรีของการพาร์สที่เพิ่มเติมแท็ก 45

รูปที่ 5-9 แผนภาพสถานะของการแทรกแท็ก เพื่อทำให้การแสดงผลถูกต้อง 45

รูปที่ 5-10 การแสดงผลเมื่อเกิดความผิดพลาดจากการพาร์สโดย xmlparser 47

รูปที่ 5-11 การแสดงผลเมื่อแปลงซอร์สโค้ดเพื่อ ให้การพาร์สโดย xmlparser ให้ผลลัพธ์ถูกต้อง 47

รูปที่ 5-12 การเปรียบเทียบการกำหนดรูปแบบอักษร 48

สารบัญรูปภาพ (ต่อ)

	หน้าที่
รูปที่ 6-1 การแสดงผลไฟล์เอกสาร WML ที่ได้จาก URL	49
รูปที่ 6-2 ผลลัพธ์จากหน้าจอแสดงผลของไฟล์ WMLที่ได้จาก URL	50
รูปที่ 6-3 แสดงไฟล์ WML ที่ผู้ใช้เขียนโปรแกรมขึ้นมา	51
รูปที่ 6-4 ผลลัพธ์ที่ได้หลังจากการคอมไพล์โปรแกรมที่ผู้ใช้ทดลองเขียนขึ้นมา	52
รูปที่ 6-5 รูปแบบของการเชื่อมโยง	52
รูปที่ 6-6 การทดสอบการเชื่อมโยง	53
รูปที่ 6-7 การเชื่อมโยงภายในไฟล์เดียวกัน	53
รูปที่ 6-8 การเชื่อมโยงไปยังไฟล์อื่น	54
รูปที่ 6-9 การเชื่อมโยงไปยังไฟล์อื่น พร้อมทั้งระบุคาร์ดได้	54
รูปที่ 6-10 การแสดงผลภาษาอังกฤษจากการกำหนดรูปแบบตัวอักษรแบบต่างๆ	55
รูปที่ 6-11 การแสดงผลภาษาไทยจากการกำหนดรูปแบบตัวอักษรแบบต่างๆ	55
รูปที่ 6-12 การแสดงผลของเอกสารรูปแบบตาราง	61

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

ในปัจจุบัน เทคโนโลยีพัฒนาไปอย่างรวดเร็ว โดยเฉพาะทางด้านคอมพิวเตอร์และอินเตอร์เน็ต และยังมีแนวโน้มจะเป็นไปในรูปของ **เทคโนโลยีไร้สาย (Wireless Technology)** ส่งผลให้เกิดผลิตภัณฑ์ใหม่ ๆ เช่น เครื่องคอมพิวเตอร์มือถือ อย่าง Palm, Palm Compatible รวมทั้ง PocketPC เป็นต้น นอกจากนี้ในส่วนของโทรศัพท์มือถือก็มีเทคโนโลยี **WAP (Wireless Application Protocol)** ซึ่งเกิดขึ้นจากการเพิ่มคุณค่าของโทรศัพท์มือถือให้มีมากกว่าการเป็นแค่อุปกรณ์สื่อสารด้วยเสียง (voice) และบริการส่งข้อความสั้น ๆ (Short Message Service) โดยเพิ่มคุณสมบัติการสื่อสารข้อมูล (data) เข้ามาด้วย ซึ่งข้อมูลที่ว่านี้ก็คือข้อมูลในเครือข่ายอินเตอร์เน็ตนั่นเอง

แนวโน้มดังกล่าวนี้ มีที่มาสืบเนื่องมาจากความต้องการบริโภคอินเตอร์เน็ตที่เพิ่มขึ้นอย่างรวดเร็ว ขณะเดียวกันในภาพรวมของโลกของเรา ก็มีจำนวนการใช้โทรศัพท์มือถือมากกว่าคอมพิวเตอร์อย่างมาก ดังนั้นจึงมีแนวคิดที่จะนำโทรศัพท์มาเป็นเครื่องลูกข่ายสำหรับเรียกดูข้อมูลในอินเตอร์เน็ตอันเป็นที่มาของคำว่า **Mobile Internet** หรือ **อินเตอร์เน็ตไร้สาย** โดยที่ WAP ก็เป็นเพียงรูปแบบหนึ่งของ Mobile Internet เท่านั้น

เพื่อให้ผู้ให้บริการบนระบบเครือข่าย ผู้ผลิตอุปกรณ์สื่อสารไร้สาย และองค์กรที่ทำหน้าที่ให้บริการข่าวสารได้ใช้เป็นมาตรฐานเดียวกันโดยกำหนดเอาภาษา WML เป็นภาษาที่ใช้ในการแสดงข้อมูลในเครือข่ายอินเตอร์เน็ตบนอุปกรณ์สื่อสารไร้สาย สำหรับนักพัฒนาแว็พ จำเป็นจะต้องมีเครื่องมือช่วยพัฒนา (Toolkit) โดยการใช้เครื่องคอมพิวเตอร์จำลองการทำงานของโปรแกรมที่พัฒนาขึ้นมา รวมทั้งการนำเอาข้อมูลที่ถูกส่งมาจากแว็พไซต์ต่าง ๆ มาแสดงผล ซึ่งจะเป็นประโยชน์ทั้งนักพัฒนาแว็พ ในด้านการอำนวยความสะดวก และสำหรับบุคคลทั่วไปในด้านที่ได้จำลองการทำงานของแว็พ แม้จะไม่มีอุปกรณ์สื่อสารไร้สาย

แว็พบราวเซอร์ที่นักพัฒนาส่วนใหญ่ใช้กันจะสนับสนุนภาษาอังกฤษ จึงส่งผลให้แว็พไซต์ต่าง ๆ ออกแบบให้เป็นภาษาอังกฤษ ซึ่งไม่สะดวกในการที่จะพัฒนาแว็พไซต์ภาษาไทย ดังนั้นถ้าหากเรามี แว็พบราวเซอร์ที่สนับสนุนภาษาไทย จะช่วยให้นักพัฒนาแว็พที่เป็นคนไทย มีกำลังใจที่จะพัฒนาแว็พไซต์ภาษาไทย รวมไปถึงการดึงดูดให้คนไทยหันมาใช้บริการแว็พกันมากขึ้น

ด้วยเหตุผลดังกล่าวปริญญานิพนธ์เล่มนี้จึงเกิดขึ้นเพื่อนเป็นการรองรับเทคโนโลยีแว็พในการสนับสนุนการพัฒนาแว็พไซต์ต่าง ๆ ที่เป็นแว็พไซต์ภาษาไทยที่กำลังจะเกิดขึ้นอันเนื่องมาจากความนิยมที่กำลังจะเกิดขึ้นในปัจจุบันเพื่อสนับสนุนการบริการกับผู้ใช้บริการที่เป็นคนไทยซึ่งคาดว่าเทคโนโลยีแว็พจะเติบโตในระยะเวลาอันใกล้นี้

1.2 วัตถุประสงค์ของปริญญานิพนธ์

- เพื่อสร้างแว็พบราวเซอร์ภาษาไทย ที่สามารถแสดงผลภาษาไทยได้ และรับรองแท็ก WML ให้สมบูรณ์ที่สุด
- เพื่อสร้างแว็พอิดิเตอร์ที่สามารถให้ผู้ใช้สามารถเขียนแท็ก WML แล้วคอมไพล์เพื่อดูผลลัพธ์ได้
- เพื่อเป็นแนวทางในการพัฒนาแว็พบราวเซอร์ภาษาไทยต่อไปได้
- พัฒนาปรับปรุงงานวิจัยของโครงงานก่อนหน้านี้ [9]

1.3 ขอบเขตของปริญญานิพนธ์

- ในส่วนบราวเซอร์สามารถแสดงผลได้ถูกต้องตามหลักการของไวยากรณ์ภาษา WML และรองรับแท็กภาษา WML ได้มากขึ้น
- ในส่วนของบราวเซอร์สามารถแสดงผลข้อมูลที่เป็นภาษาไทยได้อย่างถูกต้อง
- ในส่วนของอิดิเตอร์สามารถทำให้เขียนโปรแกรมภาษา WML และทำการแสดงผลลัพธ์จากเอกสาร WML ที่ผู้ใช้เขียนขึ้นมาบนบราวเซอร์ได้ดีขึ้นและสามารถแสดงผลภาษาไทยได้ถูกต้อง

1.4 วิธีการดำเนินงาน

- ศึกษาโปรโตคอลของแว็พ เพื่อให้เข้าใจการดำเนินงาน และโครงสร้างของแว็พ
- ศึกษาวิธีการติดตั้งเซิร์ฟเวอร์เพื่อให้สามารถรองรับภาษา WML ได้
- สำรวจหาเครื่องมือต่าง ๆ ที่มีเพื่อใช้ในการเขียนและพัฒนาภาษา WML
- ศึกษาโปรโตคอล HTTP ที่ใช้ในการเชื่อมต่อกับแว็พเซิร์ฟเวอร์
- ศึกษา XML Parser เพื่อใช้ในการ Parsing เอกสาร WML
- ทำการเขียนโปรแกรมเพื่อเชื่อมต่อกับ เซิร์ฟเวอร์ โดยใช้ไลบรารี URL ของภาษา Java
- ทำการเขียนโปรแกรมเพื่อ Parsing เอกสาร WML โดยใช้ไลบรารี XMLParser ของภาษา Java
- เตรียมรายงาน เอกสาร และปริญญานิพนธ์

บทที่ 2
ความรู้เบื้องต้นเกี่ยวกับแว็พ

2.1 ความเป็นมาของแว็พ

แว็พ (WAP : Wireless Application Protocol) เป็นตัวกลางสำคัญในการเชื่อมต่อโลกของโทรศัพท์เคลื่อนที่เข้ากับเครือข่ายอินเตอร์เน็ต ซึ่งในบางกรณีก็สามารถประยุกต์ใช้กับเครือข่ายอินทราเน็ตภายในองค์กร จุดประสงค์เพื่อขยายขอบเขตในการให้บริการเสริมให้กับผู้ใช้บริการโทรศัพท์เคลื่อนที่โดยไม่คำนึงถึงประเภทของเครือข่ายบริการ และเครื่องลูกข่ายแต่อย่างใด ผู้ใช้บริการสามารถดึงข้อมูลจากเครือข่ายอินเตอร์เน็ตได้จากโทรศัพท์เคลื่อนที่ของตน โดยตัวเครื่องอาจมีขนาดเท่าฝ่ามือแต่มีขีดความสามารถในการทำงานเทียบเท่ากับเครื่องคอมพิวเตอร์ส่วนบุคคล

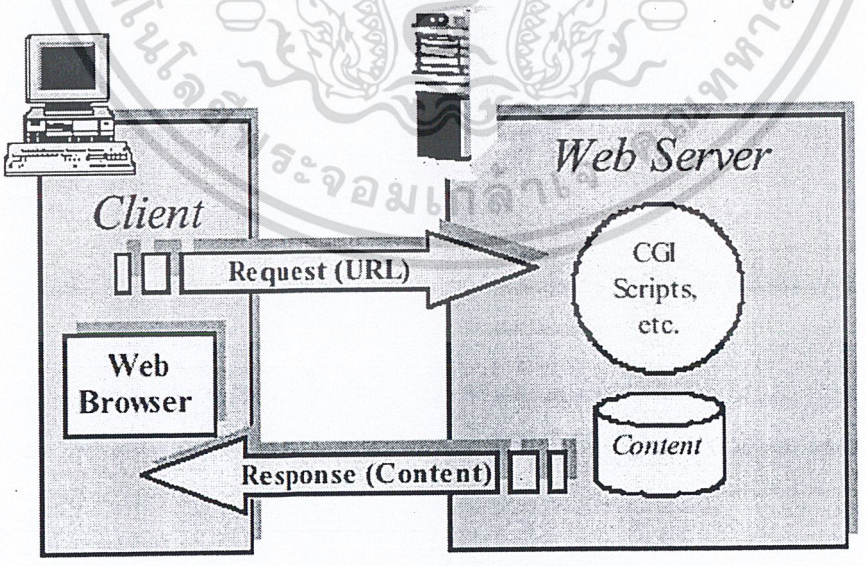
แว็พเป็นมาตรฐานสื่อสารสากลซึ่งเป็นมาตรฐานที่สร้างขึ้นจากความนิยม หรือที่เรียกว่า de-facto Standard ซึ่งมิได้ถูกออกแบบกำหนดและควบคุมโดยบริษัทใดบริษัทหนึ่งแต่เป็นผลมาจากการรวมกันวางข้อกำหนดระหว่างบริษัทอีริคสัน โนเกีย โมโตโรล่า และบริษัทอันไวร์ แพลเน็ต ซึ่งรวมกันก่อตั้งองค์กรที่มีชื่อเรียกว่า WAP Forum ขึ้นในปี พ.ศ. 2540 จุดมุ่งหมายในเบื้องต้นก็คือการวางข้อกำหนดทางอุตสาหกรรมสำหรับสนับสนุนการพัฒนาบริการพิเศษผ่านเครือข่ายสื่อสารไร้สาย ข้อกำหนดแว็พเป็นการระบุถึงกลุ่มโพรโตคอลหรือข้อกำหนดทางการสื่อสารที่มีใช้งานในระดับต่าง ๆ ตามแบบจำลอง OSI ซึ่งช่วยทำให้ผู้ให้บริการเครือข่ายโทรศัพท์เคลื่อนที่ ผู้ผลิตอุปกรณ์สื่อสาร และบริษัทที่ทำหน้าที่วิจัยและพัฒนา บริการพิเศษสามารถร่วมมือกันสร้างบริการพิเศษไม่จำกัดรูปแบบ ผ่านเครือข่ายสื่อสารไร้สาย ในปัจจุบัน มีองค์การต่าง ๆ ซึ่งเป็นตัวแทนทั้งจากผู้ให้บริการเครือข่าย ผู้ผลิตอุปกรณ์สื่อสาร ผู้ให้สัมปทาน บริษัทผู้ผลิตซอฟท์แวร์ ผู้ขายสื่อข้อมูลต่าง ๆ มากกว่า 1,000 รายเข้ารวมเป็นสมาชิกของ WAP Forum เพื่อผลักดันมาตรฐานแว็พให้มีการประยุกต์ใช้งานจริงในเครือข่ายโทรศัพท์เคลื่อนที่ทั่วโลก ซึ่งสามารถเข้าชมเว็บไซต์ของ WAP Forum ได้ที่ http://www.wapforum.org

นอกจากนี้ ข้อกำหนดแว็พยังมีการกล่าวถึงสภาวะแวดล้อมของการประยุกต์ใช้งาน (Wireless Application Environment) หรือ WAE โดยมีจุดประสงค์เพื่อให้บรรดาผู้ให้บริการระบบเครือข่าย ผู้ผลิตอุปกรณ์สื่อสาร และบริษัทซึ่งทำหน้าที่ให้บริการข่าวสารข้อมูล ได้มีโอกาสในการสร้างบริการพิเศษให้กับผู้ใช้บริการของตน โดยสามารถสร้างบริการเสริมพิเศษที่มีความแตกต่างจากคู่แข่งของตน ข้อกำหนด WAE มีการระบุถึงเรื่องของไมโครบราวเซอร์ (Microbrowser) การเขียนสคริปต์ อิเล็กทรอนิกส์เมล์ บริการแลกเปลี่ยนข่าวสารระหว่างเว็บไซต์กับเครื่องลูกข่ายโทรศัพท์เคลื่อนที่ (World Wide Web-to-mobile-handset messaging) และบริการรับส่ง เทเลแฟกซ์ผ่านโทรศัพท์เคลื่อนที่ ซึ่งทั้งนี้จะเห็นว่าการเชื่อมต่อระหว่างเครือข่ายทั้งสองนี้กระทำผ่านอุปกรณ์ที่มีชื่อว่า WAP Gateway เมื่อเปรียบเทียบกับแนวคิดของเทคโนโลยีอินเตอร์เน็ตตามปกติแล้ว เครื่องลูกข่ายโทรศัพท์เคลื่อนที่จะทำหน้าที่เป็นเว็บไมโครบราวเซอร์ ในขณะที่ข่าวสารและโปรแกรมประยุกต์ใช้งานต่าง ๆ ก็ยังคงอยู่บนเซิร์ฟเวอร์ในเครือข่ายอินเตอร์เน็ตดังเช่นเดิม

2.2 ข้อกำหนดมาตรฐานแว็พ

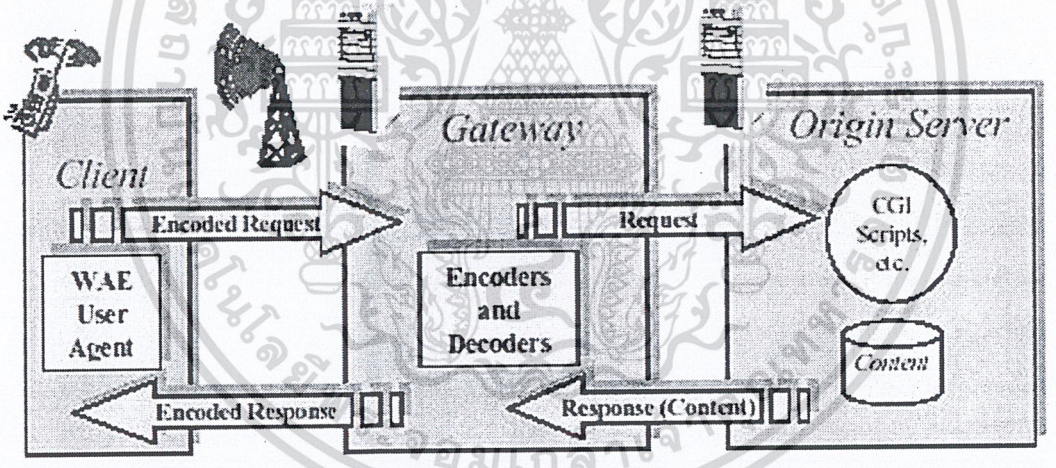
ความสำเร็จของข้อกำหนดมาตรฐานแว็พ มีผลต่อเนื่องมาจากการที่มาตรฐานดังกล่าวได้รับการจัดวางให้เป็นข้อกำหนดแบบเปิดมาตั้งแต่แรก ผนวกกับการเลือกใช้โพรโตคอลและโครงสร้างของสถาปัตยกรรมระบบสัญญาณที่เป็นมาตรฐานสากลอยู่ก่อนแล้ว ทั้งหมดนี้ก็เพื่อให้สามารถรองรับการใช้อุปกรณ์สื่อสารไร้สายในการเชื่อมต่อกับเครือข่ายอินเตอร์เน็ต นอกจากนั้นข้อกำหนดมาตรฐานแว็พยังเปิดทางออกให้กับปัญหาต่าง ๆ ที่ยังไม่ได้รับการแก้ไขจากองค์การสื่อสารอื่น ๆ ไม่ว่าจะเป็น W3C, ETSI, TIA หรือ IETF และด้วยการที่แว็พมีฐานะเป็นตัวเร่งให้เกิดความสนใจในการพัฒนาและกำหนดมาตรฐานการสื่อสารไร้สายอื่น ๆ ส่วนประกอบที่สำคัญของมาตรฐานแว็พประกอบไปด้วยสิ่งต่าง ๆ ต่อไปนี้

2.2.1 ข้อกำหนดของแบบจำลองสำหรับในการพัฒนาโปรแกรมแบบแว็พ รูปที่ 2-1 มาตรฐานการกำหนดแบบจำลองดังกล่าวก่อให้เกิดประโยชน์ต่อกลุ่มนักพัฒนาโปรแกรมเป็นอันมาก ไม่ว่าจะเป็นเรื่องของแบบจำลองที่ไม่แตกต่างไปจากมาตรฐานการเขียนโปรแกรมสำหรับใช้งานกับเครือข่ายอินเตอร์เน็ตที่เคยมีอยู่ก่อนแล้ว การออกแบบโครงสร้างสำหรับเขียนโปรแกรมที่ได้รับการยอมรับ และความเป็นไปได้ในการจัดหาอุปกรณ์หรือเครื่องมือในการเขียนโปรแกรในตระกูล XML นอกจากนี้ทาง WAP Forum ยังได้กำหนดความเหมาะสมและเปิดทางเลือกเพิ่มเติมในตัวของแบบจำลอง เพื่อให้สามารถนำไปใช้งานได้อย่างมีประสิทธิภาพกับสภาพการสื่อสารไร้สายผ่านเครือข่ายแต่ละประเภท ไม่ว่าจะเป็นเครือข่ายวิทยุติดตามตัวแบบ 2 ทิศทาง มาตรฐานการสื่อสารและการเขียนโปรแกรมที่ใช้งานอยู่ก่อนแล้วจะถูกนำมาปรับปรุงหรือนำมาใช้งานโดยตรงกับข้อกำหนดแว็พ อยู่ตลอดเวลา



รูปที่ 2-1 แบบจำลองแว็พสำหรับการเขียนโปรแกรม

ข้อกำหนดเทคโนโลยีแว็พ มีความคล้ายคลึงกับเทคโนโลยีบราวเซอร์บนเครือข่ายอินเตอร์เน็ต ซึ่งความแตกต่างอันเกิดจากคุณลักษณะทางกายภาพ และอัตราเร็วในการรับส่งข้อมูลของเครือข่ายโทรศัพท์เคลื่อนที่เท่านั้น หากจะมองโครงสร้างของการเชื่อมต่อตามเทคโนโลยีแว็พให้เป็นภาพของการสืบค้นข้อมูลในเครือข่ายอินเตอร์เน็ตก็จะเป็นไปตามรูปที่ 2-1 ผู้ใช้บริการอินเตอร์เน็ตคือผู้ใช้อุปกรณ์ปลายทางทำการส่งคำสั่งขอดาวน์โหลดข้อมูลจากเครื่องคอมพิวเตอร์เซิร์ฟเวอร์ซึ่งอยู่ที่ใดที่หนึ่งในโลก สิ่งที่ผู้ใช้บริการร้องขอจะได้รับการตอบรับจากเครื่องคอมพิวเตอร์เซิร์ฟเวอร์โดยมีการส่งเนื้อหา (Content) ตอบกลับมายังผู้บริการในโลกของการสื่อสารอินเตอร์เน็ต เครื่องคอมพิวเตอร์ของผู้ใช้บริการจะมีการติดตั้งโปรแกรมบราวเซอร์ ในขณะที่เครื่องคอมพิวเตอร์เซิร์ฟเวอร์มีสคริปต์ CGI ทำงานอยู่ เมื่อมองกลับมาถึงโครงสร้างการเชื่อต่อที่แท้จริงของเทคโนโลยีแว็พ ตามรูปที่ 2-2 จะพบว่ามีอุปกรณ์เกตเวย์แว็พคั่นกลางอยู่ระหว่างอุปกรณ์ปลายทางซึ่งก็คือเครื่องลูกข่ายโทรศัพท์เคลื่อนที่กับเว็บเซิร์ฟเวอร์ คำสั่งและเนื้อหาที่รับส่งระหว่าอุปกรณ์ปลายทาง กับเกตเวย์แว็พจะอยู่ในรูปแบบคำสั่งที่ถูกแปลงรูป หรืออาจเรียกได้ว่าเข้ารหัสไว้ โดยเกตเวย์แว็พจะเป็นตัวกลางในการแปลงข้อมูลดังกล่าวกลับไปเป็นข้อมูลโต้ตอบมาตรฐานดังรูปที่ 2-2



รูปที่ 2-2 แสดงโครงสร้างจริงของเครือข่ายสื่อสารแว็พ

สิ่งที่เห็นได้ชัดเจนก็คือเครื่องลูกข่ายโทรศัพท์เคลื่อนที่สามารถดึงจากเว็บไซต์ทั่วไปที่มีอยู่ในเครือข่ายอินเตอร์เน็ตได้ โดยไม่จำเป็นต้องมีการแก้สคริปต์หรือเนื้อความใด ๆ บนเว็บไซต์เหล่านั้น ทั้งนี้เกตเวย์แว็พจะรับหน้าที่ในการแปลงรูปแบบข้อมูลให้เหมาะสมกับตัวเครื่องลูกข่ายโทรศัพท์เคลื่อนที่เอง อย่างไรก็ตามผู้ออกแบบเว็บไซต์หรือแม้กระทั่งผู้ให้บริการระบบเครือข่ายโทรศัพท์เคลื่อนที่ ก็สามารถออกแบบสร้างเว็บไซต์ ที่มีเนื้อหาเฉพาะสำหรับส่งไปให้กับเครื่องลูกข่ายได้โดยตรง ทั้งนี้จากการเขียนเว็บไซต์ด้วย ภาษา WML และการเขียน WMLscript

2.2.2 ภาษาและรูปแบบที่ใช้ในการเขียนโปรแกรมบนมาตรฐานแว็พนั้นมีรากฐานมาจากมาตรฐานภาษา XML WML (Wireless Markup Language) ที่ใช้ในมาตรฐานแว็พนี้ได้รับการสร้างขึ้นเพื่อ

สนับสนุนกิจกรรมทางการสื่อสารซึ่งต้องอาศัยการประมวลผลที่สูงมากบนอุปกรณ์สื่อสารที่มีข้อจำกัดหลายประการ ที่สำคัญก็คือภาษา WML และ WMLscript ตามมาตรฐานแว็พ จะไม่มีการยึดติดกับรูปแบบการป้อนข้อมูลเข้าจากคีย์บอร์ดตามมาตรฐาน QWERTY ซึ่งเป็นรูปแบบมาตฐานทั่วไปตามเครื่องคอมพิวเตอร์ แต่สามารถทำงานได้กับการป้อเข้าข้อมูลในหลายรูปแบบ ซึ่งเป็นสิ่งจำเป็นอันเนื่องมาจากข้อกำหนดทางกายภาพของอุปกรณ์สื่อสารไร้สายเหล่านี้ ยิ่งไปกว่านั้น ภาษาดังกล่าวยังได้รับการออกแบบมาให้สามารถทำงานกับอุปกรณ์แสดงผลที่มีขนาดเล็ก

สิ่งที่ต่างออกไปจากรูปแบบเอกสาร HTML ที่ใช้กับเครือข่ายอินเตอร์เน็ตก็คือ เอกสารที่ถูกสร้างขึ้นตามมาตรฐาน WML ก็คือเอกสารแต่ละชุด ซึ่งก็เปรียบเสมือนเว็บไซต์แต่ละแห่ง จะมีการจัดการข้อมูลภายใน โดยแบ่งออกเป็นหน้าเช่นเดียวกับ ที่ปรากฏในเครือข่ายอินเตอร์เน็ต แต่ละหน้าจะถูกเรียกเป็นการ์ด (Card) โดยการ์ดแต่ละแผ่นจะมีคุณสมบัติพิเศษที่ช่วยให้ผู้ใช้งานสามารถสั่งย้อนถอยไปสู่ข้อมูลในหน้าที่ผ่านมา หรือเดินหน้าไปดูข้อมูลที่อยู่หน้าถัดไป การจัดสรรข้อมูลในลักษณะดังกล่าวทำให้โครงสร้างของแฟ้มข้อมูลเอกสาร WML มีขนาดเล็กกว่า HTML มากส่งผลให้มีความเหมาะสมที่ถูกรับส่งผ่านเครือข่ายสื่อสารไร้สาย อันมีข้อจำกัดในเรื่องของ แบนด์วิดธ์ ได้ดี ในส่วนของอุปกรณ์เกตเวย์แว็พ ทำหน้าที่เป็นจุดเชื่อมต่อระหว่างเครือข่ายอินเตอร์เน็ตกับเครือข่ายไร้สายนั้น จะมีการส่งข้อมูล WML ที่ได้รับจากเครือข่ายไร้สาย ไปสู่เครือข่ายอินเตอร์เน็ต โดยข้อมูลเหล่านี้ จะถูกแปลงรูปแบบให้เป็นไปตามมาตรฐาน HTML 1.1 ซึ่งหมายความว่าบรรดาผู้พัฒนาเว็บไซต์ ในเครือข่ายอินเตอร์เน็ต สามารถใช้โครงสร้างรวมถึงเครื่องมือในการพัฒนาที่มีอยู่เดิมของตนในการทำธุรกิจผ่านอุปกรณ์ผ่านอุปกรร์สื่อสารเคลื่อนที่ได้

2.2.3 ข้อกำหนดทางเทคนิคของโปรแกรมบราวเซอร์ขนาดเล็ก (Microbrowser) ที่ฝังอยู่ในเครื่องลูกข่ายอุปกรณ์สื่อสารไร้สาย อันได้แก่โทรศัพท์เคลื่อนที่แบบแว็พที่อำนวยความสะดวกให้กับผู้ใช้งานในการเข้าสู่โลกอินเตอร์เน็ตผ่านอุปกรณ์สื่อสารของตนนั้น มีรูปแบบการทำงานที่สอดคล้องกับเว็บบราวเซอร์ (WEB Browser) ที่มีใช้งานอยู่ทั่วไปบนเครือข่ายอินเตอร์เน็ต หน้าที่การทำงานที่สำคัญของบราวเซอร์ขนาดเล็กก็คือการจัดการแปลงเนื้อหาที่ปรากฏอยู่ในเอกสาร WML หรือ WMLscript ให้ไปแสดงผลบนหน้าจออุปกรณ์สื่อสารไร้สายเพื่อให้ผู้ใช้งานรับทราบ บราวเซอร์เหล่านี้ได้รับการออกแบบมาเป็นพิเศษเพื่อให้สามารถทำงานได้กับอุปกรณ์สื่อสารไร้สายได้อย่างมีประสิทธิภาพ ทั้งในเรื่องของอัตราการประมวลผลที่ไม่สูงนักอันเนื่องมาจากเนื้อหาของเอกสาร WML ที่มีความกระชับ และความรวดเร็วในการรับส่งข้อมูลที่ปรากฏต่อสายตาผู้ใช้งาน นอกจากนั้นความคล่องตัวในการทำงานก็คือกุญแจสำคัญสำหรับบราวเซอร์เหล่านี้ ตัวอย่างเช่น ผู้ออกแบบเอกสาร WML สามารถใส่รูปภาพที่มีขนาดไม่ใหญ่มากลงในเอกสาร WML เพื่อแสดงเครื่องหมายการค้าหรือรูปภาพบางอย่างเพื่อให้ไปแสดงผลบนหน้าจอของอุปกรณ์สื่อสารได้นอกเหนือจากตัวอักษรทั่วไป

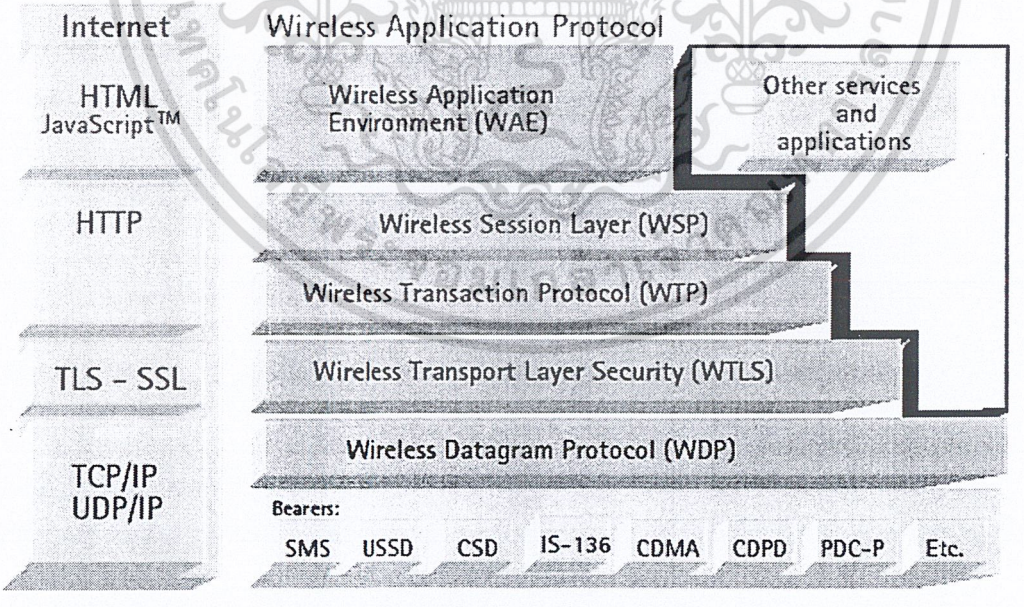
2.2.4 โครงสร้างของแบบจำลองโพรโตคอลที่ได้รับการออกแบบมาด้วยจุดประสงค์ที่ต้องการทำให้ข้อมูลที่ถูกรับส่งผ่านเครือข่ายสื่อสารไร้สายมีขนาดกะทัดรัดที่สุดนั่นหมายถึงการที่ข้อมูลควบคุมซึ่งเป็นของโพรโตคอลในระดับต่าง ๆ ที่มีขนาดเล็กทำให้ไม่ไปแย่งแบนด์วิดธ์ ในการรับส่งข้อมูลของตัวเนื้อหาหลัก นอกจากนั้นการออกแบบโครงสร้างโพรโตคอลตามมาตรฐานแว็พ ยังเปิดช่องทางให้กับนักพัฒนา สามารถสร้างโปรแกรมการประยุกต์ใช้งานหรือบริการพิเศษที่สำคัญโดยอาศัยบริการที่มีบนมาตร

ฐานแว็พได้อย่างเต็มที่ และที่สำคัญหากพิจารณาในส่วนสถาปัตยกรรมของแว็พ จะเห็นว่ามาตรฐานแว็พสามารถทำงานได้กับ มาตรฐานการสื่อสารไร้สายหลากหลายประเภท

2.2.5 การพัฒนาความสามารถในด้าน WTA (Wireless Telephony Application) ซึ่งจะช่วยทำให้ผู้พัฒนาโปรแกรมสามารถควบคุมการทำงานของโทรศัพท์เคลื่อนที่ได้จากเว็บไซต์ โดยการทำงานกลาย ๆ ประการของ?โรศัพท์เคลื่อนที่จะถูกกำหนดให้อยู่ในรูปของชุดคำสั่งของการเขียนโปรแกรม ฟังก์ชันโดยทั่วไปที่สามารถควบคุมได้ก็คือ การโทรออก การจัดการสมุดบันทึกเลขหมายโทรศัพท์ และข้อมูลในส่วนของ Short Message ซึ่งผู้พัฒนาเว็บไซต์ที่ให้บริการจัดสรรข้อมูลส่วนบุคคล (Organizer) โดยมีมาตรการในการรักษาความปลอดภัยของข้อมูลอย่างดี ผู้ใช้งานแต่ละรายสามารถใช้เครื่องคอมพิวเตอร์ ตามสำนักงานเข้าไปจัดการปรับปรุงข้อมูลเว็บไซต์ แล้วจึงสั่งให้มีการดาวน์โหลดข้อมูล ใหม่ลงสู่เครื่องลูกข่ายโทรศัพท์เคลื่อนที่ของตน ผ่านทางแว็พ อีกตัวอย่างก็คือการควบคุมการรับสายเข้า โดยเมื่อมีสายเรียกเข้าสู่ผู้ใช้บริการรายหนึ่ง ก็จะมีเอกสาร WML ส่งไปเพื่อแจ้งให้ผู้ใช้บริการเลือกว่าจะรับสายหรือโอนสายไปยังเลขหมายต่าง ๆ ตามต้องการ ทั้งนี้สามารถกำหนดได้อีกด้วยว่าจะให้มีการจัดการเลขหมายที่เรียกเข้าได้อย่างไร แนวทางในการดำเนินธุรกิจจริงคงจะมีมากกว่านี้ ขึ้นอยู่กับความคิดของผู้ประกอบการแต่ละราย

2.3 การทำงานและโครงสร้างของแว็พ

2.3.1 สถาปัตยกรรมของแว็พ (WAP Architecture)



WAP and Internet protocol stacks

รูปที่ 2-3 สถาปัตยกรรมของแว็พ

โมเดลของ ชั้นสื่อสาร ใน WAP นี้ พัฒนามาจากชั้นสื่อสารในระบบเน็ตเวิร์ค ที่เรียกว่า OSI โมเดล ซึ่งมีโปรโตคอลหลัก ๆ คือ HTTP, TCP และ IP ช่วยในการรับ-ส่ง ข้อมูลระหว่างเว็บบราวเซอร์ และเว็บเซิร์ฟเวอร์ สำหรับชั้นสื่อสารใน WAP แต่ละเลเยอร์ แบ่งได้ดังนี้

2.3.1.1 WAE (Wireless Application Environment) เป็นส่วนที่มีการทำงานเหมือนกับ HTML และ Java Script คือการกำหนดภาษา WML และ WMLScript ที่ทำหน้าที่แสดงข้อมูล ออกทางหน้าจอเครื่องมือสื่อสารไร้สายผ่านทางแว็พบราวเซอร์ ถือเป็นมาตรฐานที่เอื้ออำนวยในการพัฒนา แอปพลิเคชัน

สำหรับ เครือข่ายแบบไร้สาย เป็นตัวบ่งบอกให้ทราบว่า เมื่อเราต้องการพัฒนา แอปพลิเคชันสำหรับ WAP จะต้องมีอะไรมาเกี่ยวข้องบ้าง ซึ่งก็ได้แก่ ภาษา WML WMLScript และรวมไปถึงแว็พบราวเซอร์ด้วย

องค์ประกอบของชั้นสื่อสาร WAE สามารถแบ่งได้เป็น 2 ส่วนได้แก่

1) User Agent แว็พบราวเซอร์ถือเป็น User Agent ประเภทหนึ่งซึ่งเป็นซอฟต์แวร์ที่ทำหน้าที่แปล (Interpret) เนื้อหาของเอกสาร (ภาษา WML, WMLScript) แล้วนำมาแสดงผล ซึ่งในชั้นสื่อสาร WAE นี้มี User Agent 2 ชนิดคือ

- WML User Agent เช่น WAP Browser หรือไมโครบราวเซอร์ ซึ่งติดตั้งอยู่ใน อุปกรณ์สื่อสารไร้สาย หรือใน WAP Emulator ต่าง ๆ
- WTA (Wireless Telephony Application) User Agent ทำงานในส่วนที่เกี่ยวกับฟังก์ชันการทำงานของโทรศัพท์

2) รูปแบบข้อมูลและบริการ หมายถึง สิ่งที่จะนำไปใช้กับ User Agent ยกตัวอย่างเช่น WML, WMLScript ก็สามารถนำไปแสดงใน WML User Agent (WAP Browser) ได้

2.3.1.2 WSP (Wireless Session Protocol) และ WTP (Wireless Transaction Protocol) เป็นส่วนประกอบที่มีการทำงานคล้ายกับ โปรโตคอล HTTP ก็คือส่วนที่ทำหน้าที่เชื่อมต่อการสื่อสารระหว่างเครื่องโทรศัพท์มือถือ และเซิร์ฟเวอร์ หรือเกตเวย์ ในส่วนนี้จะเกี่ยวข้องกับเทคโนโลยีการสื่อสารแบบไร้สายโดยตรง เช่น ความเร็วของข้อมูลที่สามารถส่งได้ หรือเวลาที่ใช้ในการต่อเข้ากับเซิร์ฟเวอร์

- **WSP** ทำหน้าที่เหมือนโปรโตคอล HTTP (version 1.1) ในรูปแบบไบนารี มีบทบาทในการจัดการเกี่ยวกับการรับ – ส่งข้อมูล ระหว่างไคลเอนท์ และ เซิร์ฟเวอร์ ทางฝั่งเครือข่ายไร้สาย ซึ่งในมุมมองของ WSP นั้น ไคลเอนท์ก็คือ โทรศัพท์มือถือ ส่วนเซิร์ฟเวอร์ ก็คือแว็พเกตเวย์ หรือ แว็พเซิร์ฟเวอร์ด้วยก็ได้ (WAP Server คือ WAP Gateway + WAP Application Server) ข้อมูลส่วนที่เป็น Header ก็ต้องอยู่ในรูปแบบไบนารี เพื่อความเหมาะสมในการรับ – ส่งผ่านเครือข่ายไร้สายที่มีข้อจำกัด

WSP เป็นโปรโตคอลที่จัดอยู่ใน Session Layer ซึ่งสำหรับ WAP แล้ว ใน Layer นี้ ยังแบ่งได้เป็น 2 โปรโตคอลย่อย คือ

- **WSP/B** เป็นโปรโตคอลที่ไม่ต้องสร้างการเชื่อมต่อ หรือ Session ระหว่างไคลเอนต์ กับ เกตเวย์แว็พ ก่อน การส่งข้อมูลจะไม่มีการตรวจสอบความถูกต้องโดย WTP แต่จะอาศัย WDP ในการส่งข้อมูลโดยตรงเลย
- **WSP** มีข้อกำหนดในลักษณะตรงข้าม คือต้องมีการสร้าง Session ระหว่างไคลเอนต์ กับ เกตเวย์แว็พ ที่มั่นคงและยาวนาน เพื่อให้การรับ – ส่งข้อมูลไม่มีเหตุขัดข้อง และในอีกแง่หนึ่งยังต้องสามารถระงับการติดต่อชั่วคราว **(Suspend)** และสามารถ **เรียกการติดต่อกลับมาใหม่ (Resume)** โดยไม่เปลืองทรัพยากรของระบบมากนัก เหตุที่บางครั้งต้องระงับการเชื่อมต่อชั่วคราวก็คือ ในกรณีที่มีการหยุดนิ่งนาน ๆ โดยไม่มีการรับ – ส่งข้อมูล เป็นการสิ้นเปลืองทรัพยากรของระบบ นอกจากนี้ยังต้องมีการตรวจสอบความถูกต้องของข้อมูลตามข้อกำหนดของ **WTP** ด้วย เพื่อนำข้อมูลจากชั้น **WSP/B** ไปสู่ **WDP** แล้วจึงส่งข้อมูลออกไป

การส่งข้อมูลที่ผ่าน **WSP -> WTP -> WDP** เป็นการติดต่อแบบ Connection – Oriented โดยข้อมูลจะส่งจาก **WSP** ไปตรวจสอบความถูกต้องใน **WTP** ที่อยู่ติดกัน จากนั้นก็จะนำออกสู่เครือข่ายไร้สายโดยส่งข้อมูลที่ตรวจสอบแล้วผ่าน **WDP** ส่วนการติดต่อแบบ Connectionless ก็คือ กรณีที่ใช้ **WSP/B** และอาศัย **WDP** ในการส่งข้อมูลโดยตรง โดยไม่ต้องมีการตรวจสอบความถูกต้องของข้อมูลโดย **WTP**

ความเหมาะสมในการเลือกใช้การติดต่อแบบ Connection–Oriented, Connectionless อยู่ที่การตัดสินใจของผู้พัฒนา WAP Gateway แต่ละราย เท่าที่เป็นอยู่ในปัจจุบันส่วนใหญ่ใช้การติดต่อแบบ Connectionless เนื่องจากความรวดเร็วในการรับ – ส่งข้อมูลสูง ไม่ต้องตรวจสอบความถูกต้อง ซึ่งการสังเกตได้ว่า การติดต่อแบบ Connection – Oriented จะใช้พอร์ต 9201 และสำหรับการติดต่อแบบ Connectionless จะใช้พอร์ต 9200 (ซึ่ง การกำหนดค่าเพื่อใช้บริการ WAP ต้องระบุ IP Address ของ WAP Gateway และต้องระบุพอร์ตด้วยตามที่ผู้ให้บริการ WAP Gateway กำหนดไว้)

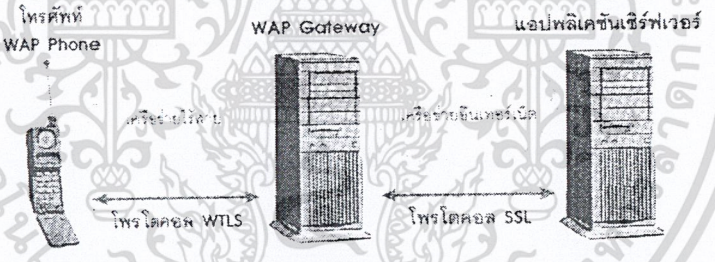
- **WTP** เกี่ยวข้องกับการรับประกันความน่าเชื่อถือของการส่งข้อมูล ซึ่งคล้ายกับหน้าที่บางส่วนของ โปรโตคอล **TCP** จะแตกต่างกันที่ **TCP** จะมองในเชิงการเชื่อมต่อ หรือ **Connection-Oriented** รวมทั้งควบคุมการส่งข้อมูลด้วย แต่ **WTP** จะมองในเชิงกระบวนการรับ – ส่งข้อมูลไปมา หรือ **Transaction-Oriented** เพราะหน้าที่ในการเชื่อมต่ออยู่ในหน้าที่ของ **WSP** แล้ว

2.3.1.3 WDP (Wireless Datagram Protocol) หรือ Transport Layer มีการทำงานเหมือนโพรโตคอล **TCP** หรือ **UDP** ที่ทำหน้าที่ในการส่งข้อมูลระหว่าเลเยอร์ชั้นบนกับเครื่องเซิร์ฟเวอร์หรือเครื่องโทรศัพท์มือถือ (client) ผ่านระบบเครือข่ายไร้สายชนิดต่าง ๆ ซึ่งไม่ว่าในกรณีใดก็ตาม การรับ - ส่งข้อมูลใน **WAP** แต่ละครั้งต้องพึ่งพาความสามารถของ **WDP** เสมอ เพราะว่าโพรโตคอลนี้ถือว่าเป็นชั้นที่ใช้สำหรับสื่อสาร (Transport Layer) ของ **WAP** โดยเฉพาะ

โปรโตคอล WDP นี้จะคอยดูแล การส่งข้อมูลไปในเครือข่าย แต่เนื่องจากชนิดของ เครือข่ายไร้สาย (bearer) มีหลายรูปแบบ เช่น GSM, CDMA, GPRS ฯลฯ ดังนั้นคุณลักษณะสำคัญของโพรโตคอล WDP ก็คือ ความเป็นอิสระไม่ผูกติดกับเครือข่าย (bearer independence) โดย WDP จะคอยอำพรางโปรโตคอลซึ่งอยู่เหนือขึ้นไปว่า กำลังทำงานกับเครือข่าย ชนิดไหน หรือส่งมาด้วยวิธีใดดังนั้นด้วยคุณลักษณะของโปรโตคอล WDP นี้เอง การพัฒนา WAP Application จึงไม่ต้องกังวลเรื่องเครือข่ายไร้สายเลย

2.3.1.4 WTLS (Wireless Transport Layer Security) เป็นเลเยอร์ที่สร้างมาจากโปรโตคอลมาตรฐานTLS (Transport Layer Security) ซึ่งรู้จักกันดีในชื่อของ SSL (Secure Sockets Layer) รุ่นของ SSL ที่ WTLS อ้างอิงถึงเป็น SSL 3.0 ซึ่ง WTLS จะทำหน้าที่ในการรักษาความปลอดภัยของข้อมูลและป้องกันภัยจากผู้บุกรุกชนิดต่าง ๆ (attacker) โดยมีบริการของ การตรวจสอบความถูกต้องของข้อมูล (Data Integrity), การป้องกันข้อมูลที่เป็นความลับ (Confidential), การพิสูจน์ตน (Authentication) และการป้องกันการมาใช้บริการโดยไม่ได้รับอนุญาต (Denial of Service) มาเป็นเครื่องมือ

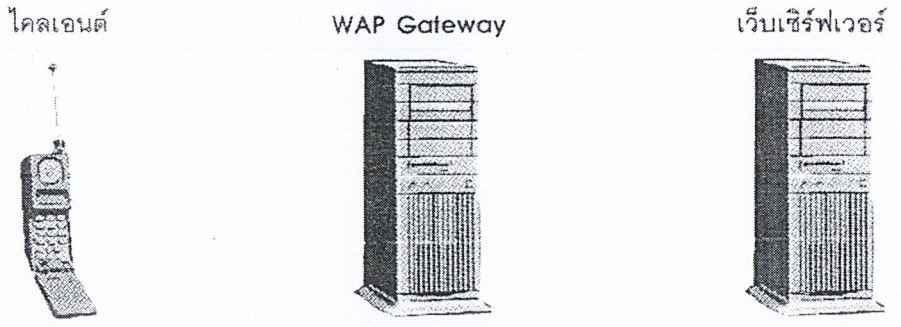
โปรโตคอลนี้ จะมีหรือไม่มีก็ได้ ทั้งนี้ขึ้นอยู่ที่ WAP Gateway และโทรศัพท์มือถือ WAP Phone ว่ารองรับโปรโตคอล WTLS หรือไม่ จากรูป 2-4 ในส่วนการเชื่อมต่อ ระหว่าง WAP Gateway และตัว User Agent จะมีการเชื่อมต่อด้วย เครือข่ายไร้สาย ซึ่ง WLTS คอยดูแลความปลอดภัยอยู่



รูปที่ 2-4 ระบบการรักษาความปลอดภัยของข้อมูลใน WAP และอินเตอร์เน็ต

กลไกรักษาความปลอดภัย

การเข้ารหัสด้วยโปรโตคอล WTLS จะเกิดขึ้นได้ ก็ต่อเมื่อ WAP Gateway และโทรศัพท์มือถือล้วนรองรับโปรโตคอล WTLS ทั้งคู่ ซึ่งในกรณีนี้ข้อมูลที่จะส่งไป - มาระหว่าง WAP Gateway และโทรศัพท์มือถือ จะต้องถูกเข้ารหัสด้วยข้อกำหนดของโปรโตคอล WTLS และบีบอัดให้มีขนาดเล็ก จะได้เหมาะกับการส่งไปในเครือข่ายไร้สาย ส่วนลำดับการทำงานก็แบ่งได้เป็น 2 กรณี คือ



- ขาส่ง (จากโทรศัพท์มือถือ -> เว็บเซิร์ฟเวอร์) :

เข้ารหัส WTLS --- ----> ถอดรหัส WTLS ----------> เข้ารหัส SSL --------> ถอดรหัส SSL

- ขารับ (จากเว็บเซิร์ฟเวอร์ --> โทรศัพท์มือถือ) :

ถอดรหัส WTLS <-------- เข้ารหัส WTLS <---------- ถอดรหัส SSL <--------เข้ารหัส SSL

รูปที่ 2-5 ขั้นตอนการเข้ารหัสและถอดรหัสเพื่อรักษาความปลอดภัยของข้อมูลทั้ง 2 ทิศทาง

ซึ่งกรณีขาส่งนั้น ทางฝั่งโทรศัพท์มือถือจะเข้ารหัสข้อมูลด้วยข้อกำหนดของโปรโตคอล WTLS แล้วข้อมูลจะถูกส่งผ่านเครือข่ายไร้สายมาถึง WAP Gateway ณ จุดนี้ จะมีขั้นตอนเกิดขึ้น 2 ขั้นตอนย่อย ๆ ซึ่งกินเวลาสั้นมาก ๆ ในหน่วยมิลลิวินาที คือการถอดรหัสข้อมูลด้วยข้อกำหนดของโปรโตคอล WTLS จากนั้นก็เข้ารหัสอีกครั้งตามข้อกำหนดโปรโตคอล SSL เพื่อส่งข้อมูลผ่านเครือข่ายอินเทอร์เน็ต และเมื่อมาถึงเซิร์ฟเวอร์ ข้อมูลก็จะถูกถอดรหัสตามข้อกำหนดของโปรโตคอล SSL เพื่อนำไปใช้หรือประมวลผลอีกที ส่วนกรณีของขารับ ก็จะดำเนินกระบวนการตรงกันข้าม

จุดที่สำคัญคือ กระบวนการที่เกิดขึ้นในหน่วยความจำของ WAP Gateway อันเป็นช่วยรอยต่อระหว่างเปลี่ยนโปรโตคอล (WTLS <-> SSL) ซึ่งข้อมูลจะมิได้เข้ารหัสใด ๆ เลย ถึงแม้ว่าช่วงเวลานี้จะสั้นมาก ๆ ก็ตาม แต่ผู้พัฒนา WAP Gateway ก็จำเป็นต้องให้ความสำคัญและ ระมัดระวังอย่างมากในการออกแบบระบบ อย่าให้มีการเก็บข้อมูลที่ถูกถอดรหัสไว้ในแหล่งเก็บข้อมูลอื่น ๆ รวมทั้งต้องมีระบบป้องกันความปลอดภัยในเสี้ยววินาทีที่ถอดรหัสจากโปรโตคอลหนึ่ง และเข้ารหัสใหม่ด้วยอีกโปรโตคอลหนึ่ง สำหรับ WAP Server (อุปกรณ์ที่เป็นทั้งเว็บเซิร์ฟเวอร์ และ WAP Gateway) จะไม่มีช่วงเปลี่ยนโปรโตคอล เพราะใช้เพียงโปรโตคอล WTLS เพียงอย่างเดียว ซึ่งถือว่าเป็นข้อดีในแง่ของความปลอดภัย

2.3.2 หลักการทำงานของแว็พ

การทำงานของแว็พ มีแนวคิดมาจากเว็บ นั่นคือ มีการปรับปรุงเทคโนโลยีของเว็บ ให้เหมาะสมกับการสื่อสารในเครือข่ายไร้สาย สำหรับแว็พนั้น จะมีเครื่องมือพิเศษเข้ามาทำงานเป็นตัวกลางเชื่อมต่อระหว่างโปรโตคอล แว็พ และ HTTP ซึ่งก็คือ WAP Gateway

WAP Gateway มีความสามารถเป็นได้ทั้ง Proxy Server และ แคช ด้วย แต่หน้าที่หลักจริง ๆ ของ WAP Gateway ก็คือ แปลงรูปแบบการสื่อสารระหว่างฝั่งเครือข่ายอินเตอร์เน็ต และฝั่งเครือข่ายไร้สาย หรือที่เรียกว่า **Protocol Conversion** การเรียกดูข้อมูล WAP ที่อยู่ในเว็บเซิร์ฟเวอร์ (แอปพลิเคชั่นเซิร์ฟเวอร์) โดยใช้โทรศัพท์มือถือเรียกผ่าน WAP Gateway เป็นไปตามขั้นตอนดังนี้

2.3.2.1 ผู้ใช้โทรศัพท์มือถือ (ไคลเอนต์) ส่ง URL ของเอกสารที่ต้องการไปยัง WAP Gateway โดยส่งเป็นคำร้องขอในรูปแบบโปรโตคอล WSP (WSP Request)

2.3.2.2 WAP Gateway ถอดรหัส (decode) คำร้องขอที่อยู่ในรูปแบบ ไบนารี (WSP Request) เพื่อแปลงให้อยู่ในรูปแบบของคำร้องขอแบบ HTTP (HTTP Request) โดยอาจอาศัย ตาราง **Mapping Table** ที่มีอยู่ใน WAP Gateway เป็นตัวช่วย

2.3.2.3 WAP Gateway สร้างการเชื่อมต่อ (Connection) ไปยังเว็บเซิร์ฟเวอร์แล้วส่งคำร้องขอตามไปในรูปแบบ โปรโตคอล HTTP (HTTP Request)

2.3.2.4 เว็บเซิร์ฟเวอร์จะประมวลผลคำร้องขอนั้น และตรวจสอบดูว่า เอกสารตามที่ร้องขอเป็นลักษณะซอร์ซโค้ด WML ธรรมดา (Static) หรือไม่ หากเอกสารนั้นเรียกการทำงานของสคริปต์ต่าง ๆ เช่น CGI, ASP ก็จะต้องประมวลผลสคริปต์นั้นก่อน เพื่อให้กลายเป็นเอกสาร WML ธรรมดา ซึ่งประกอบไปด้วย แท็ก และข้อความ

2.3.2.5 เว็บเซิร์ฟเวอร์ส่งเอกสารกลับมายัง WAP Gateway โดยส่งเป็นคำตอบกลับในรูปแบบโปรโตคอล HTTP (HTTP Response)

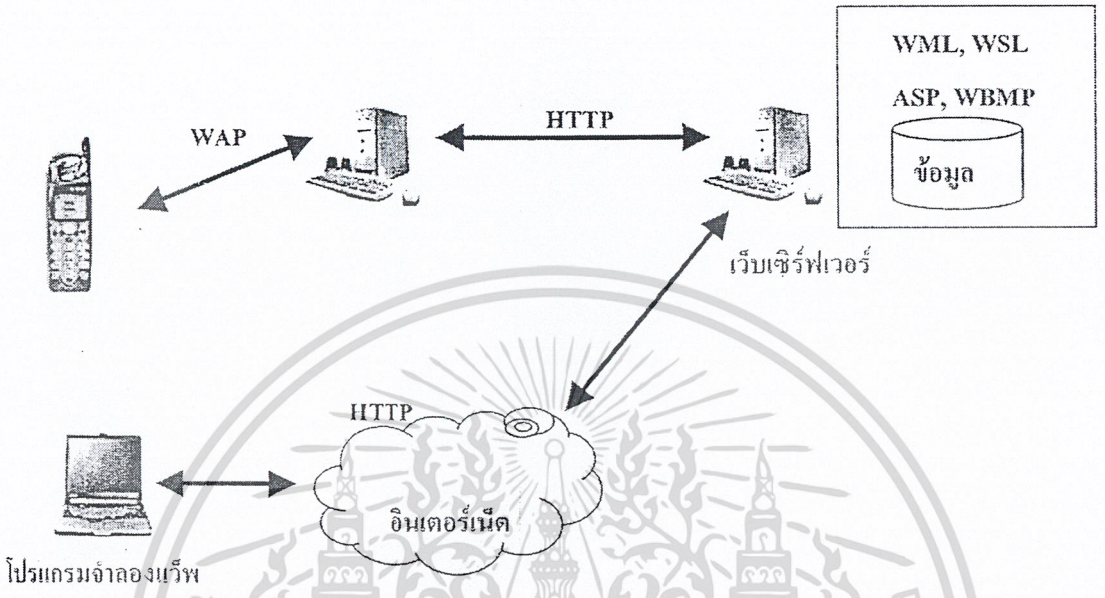
2.3.2.6 WAP Gateway จะเข้ารหัสเอกสาร (encode) ไปเป็นรูปแบบไบนารี โดยอาจจะอาศัยตาราง Mapping Table เป็นตัวช่วย

2.3.2.7 WAP Gateway สร้างการติดต่อ (Connection) ไปยังไคลเอนต์ แล้วส่งข้อมูลไบนารีนั้น เป็นคำตอบกลับในรูปแบบโปรโตคอล WSP (WSP Response) ไปยังไคลเอนต์ต่อไป

จากขั้นตอนการทำงานข้างต้นนี้ แสดงให้เห็นอย่างชัดเจนว่า หน้าที่หลักของ WAP Gateway คือ การแปลงรูปแบบข้อมูลให้เหมาะกับการสื่อสารในแต่ละฝั่ง เอกสารที่ส่งมาจากเว็บเซิร์ฟเวอร์ จะอยู่ในรูปของข้อความและแท็ก WML โดยอาศัย โปรโตคอล HTTP ช่วยในการส่งไฟล์เอกสาร แต่เมื่อมาถึง WAP Gateway แล้ว เอกสารเหล่านั้นต้องถูกเข้ารหัส (encode) ให้เป็นข้อมูลรูปแบบไบนารี เพื่อลดขนาดของข้อมูลให้เหมาะกับการสื่อสารในเครือข่ายแบบไร้สาย ทั้งนี้เป็นข้อกำหนดของ โปรโตคอล WSP ซึ่งบังคับให้รูปแบบข้อมูลเป็น แบบไบนารี จึงจะส่งได้

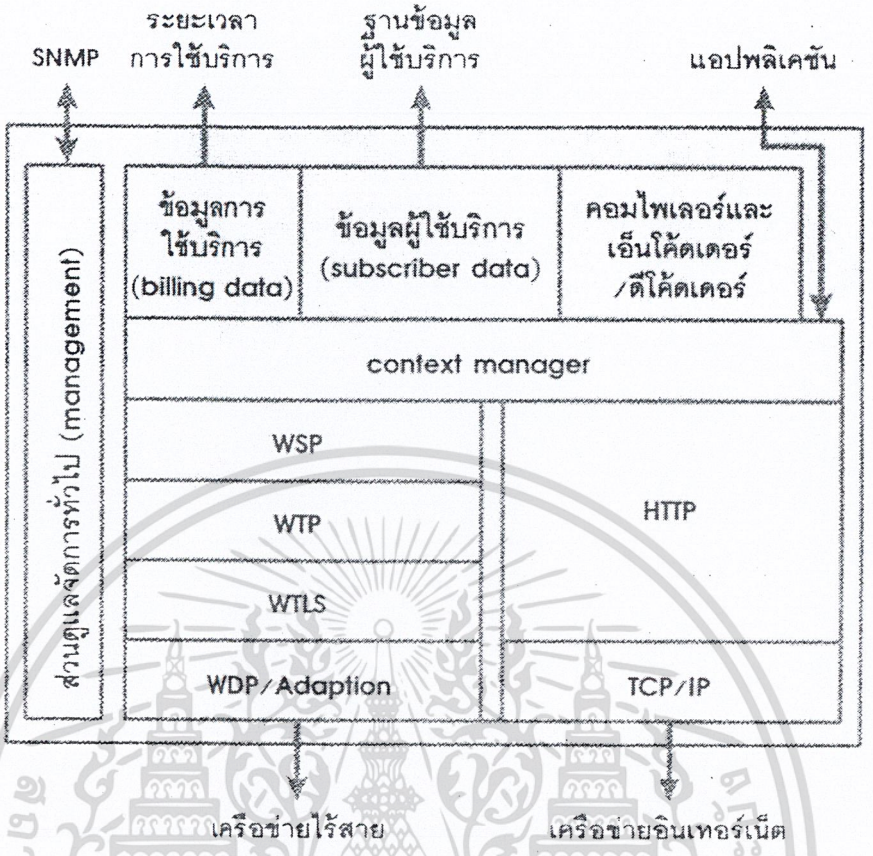
Mapping Table คือ ตารางที่ใช้แปลง แท็กคำสั่งและข้อมูลในเอกสาร WML ให้เป็นรหัสไบนารี ซึ่งจะต้องเป็นไปตามข้อกำหนด **WSP Specification** ซึ่งขั้นตอนการเข้ารหัสข้อมูล ก็คือการเปลี่ยนข้อมูลแบบ Text ไปเป็นรหัสไบนารี โดยอาศัย Mapping Table นั่นเอง ส่วนการถอดรหัสข้อมูลก็จะเป็นกระบวนการตรงกันข้าม นั่นคือ เปลี่ยนจากข้อมูลไบนารี (WSP Request) ไปเป็นข้อมูลแบบ text (HTTP Request)

นอกจากที่จะเข้าชมแว็พไซต์ โดยใช้โทรศัพท์มือถือแล้ว ยังสามารถใช้โปรแกรมจำลอง หรือที่เรียกกันว่า Emulator มาท่องโลกของแว็พ เพราะมีโปรแกรมแว็พบราวเซอร์อยู่แล้ว หรือจะเข้าไปชมจากในเว็บไซต์ที่มีโปรแกรมเหล่านี้อยู่ก็ได้ ดังแสดงในรูปที่ 2-6



รูปที่ 2-6 แสดงการใช้งานโปรแกรมแว็พบราวเวอร์ และเครื่องโทรศัพท์มือถือในการชมแว็พไซต์

2.3.3 โครงสร้างของแว็พเกตเวย์



รูปที่ 2-7 โครงสร้างพื้นฐานของ WAP Gateway

ใน WAP Gateway มีทั้งชุดโปรโตคอล WAP (WSP, WTP, WTLS, WDP) และชุดโปรโตคอล TCP/IP ทั้งนี้เนื่องจาก WAP Gateway เป็นตัวกลางระหว่างเครือข่ายอินเทอร์เน็ต และเครือข่ายแบบไร้สาย จึงต้องมีคุณสมบัติรองรับโปรโตคอลที่ใช้ในแต่ละฝั่ง เมื่อติดต่อกับ เว็บเซิร์ฟเวอร์ก็ต้องอาศัยชุดโปรโตคอล TCP/IP ส่วนหารติดต่อกับไคลเอนต์ หรือโทรศัพท์ที่ใช้ระบบ WAP ก็ต้องอาศัยชุดโปรโตคอล WAP นั่นเอง

กลุ่ม WAP Forum ได้เสนอชุดโปรโตคอลสำหรับ WAP เพื่อการสื่อสารด้วยเครือข่ายไร้สายโดยเฉพาะ ซึ่งแนวคิดของชุดโปรโตคอล WAP นี้ ก็เป็นอีกเรื่องที่นำเอาแนวคิดของชั้นสื่อสารในระบบอินเทอร์เน็ตมาประยุกต์อีกครั้ง โปรโตคอลทั้งหลายในชั้นสื่อสาร WAP ก็เทียบเคียงได้กับโปรโตคอลในอินเตอร์เน็ต เช่น HTTP, TCP/IP หรือแม้กระทั่งระบบรักษาความปลอดภัยแบบ SSL เช่นกัน

องค์ประกอบของแว็พเกตเวย์จากภาพที่ 2- 7

2.3.3.1 WDP เป็นระดับชั้นขนส่งข้อมูลซึ่งทำหน้าที่ส่งและรับข้อมูลข่าวสาร จากเครือข่ายต่าง ๆ ที่เกตเวย์ทำการเชื่อมต่ออยู่ด้วย ไม่ว่าข้อมูลนั้นจะอยู่ในรูปของ SMS, USSD, CSD, CDPD, IS-136 และ GPRS

2.3.3.2 WTLS เป็นระดับชั้นที่ทำหน้าที่รักษาความปลอดภัยของข้อมูล ซึ่งผู้ให้บริการจะกำหนดให้มีหรือไม่ก็ได้ ระดับชั้นดังกล่าวมีฟังก์ชันที่ทำหน้าที่ในการเข้ารหัสข้อมูลซึ่งเหมาะสำหรับการใช้งานกับบริการบางประเภทเช่น m-Commerce

2.3.3.3 WTP เป็นระดับชั้นที่ทำหน้าที่สนับสนุนการสร้างกระบวนการ การส่งผ่านข้อมูล พร้อมทั้งช่วยเพิ่มระดับความน่าเชื่อถือให้กับการเชื่อมต่อแบบ Datagram ที่กระทำโดยระดับชั้น WDP

2.3.3.4 WSP เป็นระดับชั้นที่ให้การสนับสนุน เพิ่มประสิทธิภาพในการแลกเปลี่ยนข้อมูล ระหว่างโปรแกรมประยุกต์ต่าง ๆ

2.3.3.5 HTTP Interface ช่วยให้เครื่องลูกข่ายโทรศัพท์เคลื่อนที่สามารถดึงข้อมูลในรูปแบบของ แว็พจากเครือข่ายบน อินเตอร์เน็ต

2.3.3.6 คอมไพเลอร์ และเอ็นโคเดอร์/ดีโค้ดเดอร์ เป็นส่วนสำคัญที่ขาดไม่ได้สำหรับ WAP Gateway เพราะ **encoder** เป็นตัวจัดการเกี่ยวกับการเข้ารหัสเอกสาร WML ที่ส่งมากจากเว็บเซิร์ฟเวอร์ ให้เป็นข้อมูลรูปแบบไบนารี เพื่อส่งผ่านเครือข่ายไร้สายไปให้แก่ไคลเอนต์ ให้เป็นข้อมูลในรูปแบบที่ใช้ในโปรโตคอล HTTP เพื่อส่งไปยังเว็บเซิร์ฟเวอร์ ส่วน **Compiler** เป็นตัวจัดการเกี่ยวกับการคอมไพล์โค้ด WMLScript ที่ส่งมาจากเซิร์ฟเวอร์ ก่อนส่งต่อไปยังเครือข่ายไร้สาย ซึ่งขั้นตอนการทำงานเหล่านี้ แสดงในรูปที่2-7 เช่นกัน

2.3.3.7 ส่วนของ **ข้อมูลการใช้บริการ (billing data)** และ **ข้อมูลผู้ใช้บริการ (Subscriber data)**จะเกี่ยวข้องกับข้อมูลต่าง ๆ ของผู้ใช้โทรศัพท์ เช่น ตรวจสอบ username และ password ว่าถูกต้องหรือไม่, หน้าจอเริ่มต้นการใช้งานของโทรศัพท์เป็นอย่างไร, บันทึกระยะเวลาใช้งานของผู้ใช้โทรศัพท์ เป็นต้น

ข้อมูลในรูปแบบของแว็พ ไม่ว่าจะอยู่ในรูปของ WML หรือ WMLScript จะถูกแปลงให้อยู่ในรูปข้อมูลไบนารีชนิดพิเศษสำหรับส่งผ่านการเชื่อมต่อทางอากาศ จากเครือข่ายโทรศัพท์เคลื่อนที่ไปยังเครื่องลูกข่าย จากนั้นไมโครเบรอเซอร์ในเครื่องลูกข่ายนะทำการอ่านข้อมูลดังกล่าวแล้วแสดงผล

2.3.4 หน้าที่หลักของ WAP Gateway

1. รองรับโปรโตคอล WAP และชุดโปรโตคอลในอินเตอร์เน็ต
2. Protocol Conversion
3. เข้ารหัสเอกสาร WML ให้เป็นข้อมูลรูปแบบไบนารี
4. คอมไพล์โค้ด WMLScript
5. เป็น Proxy Server เพื่อให้บริการข้อมูลที่ถูกเรียกใช้บ่อย ๆ
6. ดูแลจัดการด้านการรักษาความปลอดภัยของข้อมูล
7. เปลี่ยนเอกสาร HTML ที่ได้รับจากเว็บเซิร์ฟเวอร์ ให้เป็นเอกสาร WML

Protocol Conversion

ลักษณะของ Protocol Conversion โดยปกติ สำหรับการส่งข้อมูล ไป - มา ระหว่างบราวเซอร์และเว็บเซิร์ฟเวอร์ซึ่งอาศัยโปรโตคอล HTTP นั้น นอกจากตัวเอกสาร HTML ที่ประกอบไปด้วยแท็กคำสั่งต่าง ๆ จะต้องมีข้อมูลอีกส่วนหนึ่งอยู่ที่ช่วงต้นของเอกสาร ซึ่งก็คือ เฮดเดอร์ (HTML Header) ซึ่งเป็นตัวบ่งบอกรายละเอียดเกี่ยวกับเอกสารนั้น ๆ เช่น ประเภทของข้อมูลในเอกสารว่าเป็นรูปภาพหรือข้อความธรรมดา, ความยาว หรือขนาดของข้อมูล เป็นต้น

ในกรณีที่เอกสารถูกส่งไปยังเครือข่ายแบบไร้สาย WAP Gateway ก็ต้องเข้ารหัสเฮดเดอร์ให้เป็นแบบไบนารีด้วย เพื่อลดข้อจำกัดทางด้านแบนด์วิดธ์ และ latency ของเครือข่ายไร้สาย เช่นเดียวกันกับเอกสาร WML ซึ่งเฮดเดอร์แบบเดิมอยู่ในระบบโพรโตคอล HTTP การเข้ารหัสจึงเป็นการแปลงเฮดเดอร์นั้นให้สามารถส่งต่อไปในระบบโปรโตคอล WSP ฉะนั้น เฮดเดอร์เดิมซึ่งเรียกว่า **HTTP Header** ก็จะกลายเป็น **WSP Header**

การเข้ารหัสเอกสาร WML ให้เป็นข้อมูลไบนารี

เอกสาร WML จะถูกส่งมาจากเว็บเซิร์ฟเวอร์ (หรือแอปพลิเคชัน เซิร์ฟเวอร์) ผ่านเครือข่ายอินเตอร์เน็ต โดยมีรูปแบบเป็นข้อความที่สามารถอ่านได้ แต่เมื่อ WAP Gateway ได้รับเอกสารแล้ว ก่อนที่จะเข้ารหัส (encode) ให้เป็นข้อมูลแบบไบนารีเพื่อส่งต่อไปยังไคลเอนต์ WAP Gateway ก็ต้องตรวจสอบความถูกต้องตามหลักไวยากรณ์ของเนื้อหา ว่าเป็นไปตามกฎของ **XML (Extensible markup Language)** หรือไม่ เพราะว่าภาษา WML เป็นรูปแบบหนึ่งของภาษา XML และต้องยึดถือกฎเกณฑ์ของ XML ด้วย หาก WAP Gateway พบว่าเนื้อความในเอกสาร WML ไม่ถูกต้องตามหลักไวยากรณ์ของ XML สำหรับ WAP Gateway บางตัวก็จะส่งต้อความ error มาแสดงที่ไคลเอนต์เลย แต่บางตัวก็อาจแสดงข้อความทีไม่ถูกต้องออกมาเลย โดยไม่แจ้ง error

เหตุที่เราไม่ผลักภาระการตรวจสอบไวยากรณ์ให้กับเครื่องไคลเอนต์ เนื่องมาจากหลักการของ WAP จะพยายามไม่ผลักภาระการประมวลผลใด ๆ ให้กับเครื่องไคลเอนต์ (โทรศัพท์มือถือ) ที่มีข้อจำกัดต่าง ๆ มากมาย เช่น มีขีดความสามารถจำกัดในการประมวลผล, มีพลังงานจำกัด เพราะใช้แบตเตอรี่เป็นต้น ดังนั้นหลักการนี้ก็สามารถนำไปใช้ในการสร้าง WAP Application ได้

WBXML (Wireless Binary XML) คือข้อมูล XML ในรูปแบบไบนารี และเนื่องจากเอกสาร WML ก็จัดเป็นรูปแบบหนึ่งของเอกสาร XML ดังนั้น เราจึงถือได้ว่า เอกสาร WML ที่ถูกเข้ารหัส (encode) ภายใน WAP Gateway กลายเป็นข้อมูลรูปแบบไบนารี เพื่อลดขนาดแล้วส่งไปในเครือข่ายไร้สาย ก็ถือเป็น WBXML ด้วย

การคอมไพล์ โค้ด WMLScript

โค้ด WMLScript ที่ส่งมาจากฝั่งเว็บเซิร์ฟเวอร์ จำเป็นต้องให้ WAP Gateway คอมไพล์ เหมือนกับโปรแกรมที่เขียนด้วยภาษาอื่นๆ เช่นกัน โดยที่ WAP Gateway ต้องตรวจดูไวยากรณ์ (syntax) ของภาษาก่อน ผลการคอมไพล์จะได้ข้อมูลเป็น ไบต์โค้ด (bytecode) ซึ่งเป็นข้อมูลไบนารีแบบหนึ่ง เมื่อไบต์โค้ดเหล่านี้ถูกส่งต่อไปยังโทรศัพท์มือถือ ก็จะต้องผ่านกระบวนการแปลและประมวลผลเพื่อให้ทำงานตามโค้ดที่เขียนเอาไว้

การเข้ารหัสเอกสาร WML กับการคอมไพล์โค้ดภาษา WMLScript สามารถอธิบายแจกแจงได้ดังนี้

1) การเข้ารหัสเอกสาร WML อาศัยการเปรียบเทียบ (mapping) แท็ก WML ต่าง ๆ กับรหัสไบนารีที่กำหนดไว้แล้วในตาราง mapping table ซึ่งเรียกว่า WSP Specification

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

2) การคอมไพล์โค้ด WMLScript เป็นการบวนการอีกลักษณะหนึ่ง ซึ่งคล้ายกับการคอมไพล์โปรแกรม คือต้องมีการตรวจสอบไวยากรณ์ของภาษาด้วย ไม่ได้อาศัยการเปรียบเทียบหรือ mapping เหมือนกับการเข้ารหัสภาษา WML

3) เนื้อหา WML ที่ผ่านการเข้ารหัสเป็นข้อมูลแบบไบนารีมาจาก WAP Gateway แล้ว ก่อนที่จะแสดงผลจริง ๆ โดยบราวเซอร์ที่ไคลเอนต์ได้ ก็จะต้องมีการถอดรหัสกลับคืน โดยอาศัยการเปรียบเทียบ(mapping) ในทำนองเดียวกับการเข้ารหัส

4) ส่วนในกรณีของ WMLScript นั้น เมื่อไบต์โค้ดมาถึงไคลเอนต์ ก็จะถูกแปลกลับคืนโดยอาศัยอินเทอร์พรีเตอร์ (interpreter) ซึ่งวิธีนี้ต้องไปยุ่งเกี่ยวกับสถานะของตัวแปร ค่าพอยน์เตอร์ และอื่น ๆ อีกมาก ตามลักษณะของการทำงานของโปรแกรมโดยทั่วไป

2.4 การกำหนด MIME Type เพื่อให้เซิร์ฟเวอร์สามารถรองรับแว็พได้

เมื่อสร้างแว็พเพจเสร็จแล้ว ต้องทำการให้ข้อมูลที่สร้างขึ้นมาสามารทำให้โทรศัพท์มือถือและ แว็พบราวเซอร์ สามารถอ่านข้อมูลเหล่านี้ได้ โดยจำเป็นต้องทำการกำหนด MIME Type ของเซิร์ฟเวอร์ให้สามารถสนับสนุนการทำงานของแว็พได้โดยต้องทำการกำหนด MIME Type เป็นดังตารางที่ 2-1 เพื่อให้เซิร์ฟเวอร์นั้นสามารถรองรับการบริการจากโทรศัพท์มือถือและแว็พบราวเซอร์ได้

ข้อมูลที่ถูกกำหนด MIME Type จะถูกแบ่งออกเป็น 3 ส่วนคือ

1. ประเภทเอกสาร จะเป็นข้อมูลในรูปแบบอักษร ได้แก่ เอกสาร WML ซึ่งจะมีนามสกุลเป็น *.wml และ เอกสาร WMLScript จะมีนามสกุลเป็น *.wmls แต่อาจจะพบว่ามีการใช้นามสกุลเป็น *.ws หรือ *.wmlscript ด้วย

2. ประเภทรูปภาพซึ่งเป็นไฟล์รูปภาพที่ภาษา WML รองรับ ได้แก่ ไฟล์รูปภาพแบบ WBMP หรือมีนามสกุลเป็น *.wbmp

3. ประเภทแอปพลิเคชัน ใช้กับข้อมูลที่ถูกแปลงจากเอกสาร WML หรือ WMLScript ให้เป็นรหัส ไบนารีหรือไบต์โค้ดแล้ว ไฟล์เหล่านี้จะมีนามสกุลเป็น *.wmlc และ *.wmlsc

ชนิดของเอกสาร	MIME Type	นามสกุลของไฟล์
เอกสาร WML	text/vnd.wap.wml	.wml
ไฟล์รูปภาพ WBMP	image/vnd.wap.wbmp	.wbmp
เอกสาร WML ที่ได้คอมไพล์แล้ว	Applcation/vnd.wap.wmlc	.wmlc
WML สคริปต์	text/vnd.wap.wmlscript	.wmls
WML สคริปต์ที่ได้คอมไพล์แล้ว	Application/ vnd.wap.wmlscripts	.wmlsc

ตารางที่ 2-1 การกำหนด MIME Type ให้กับเซิร์ฟเวอร์

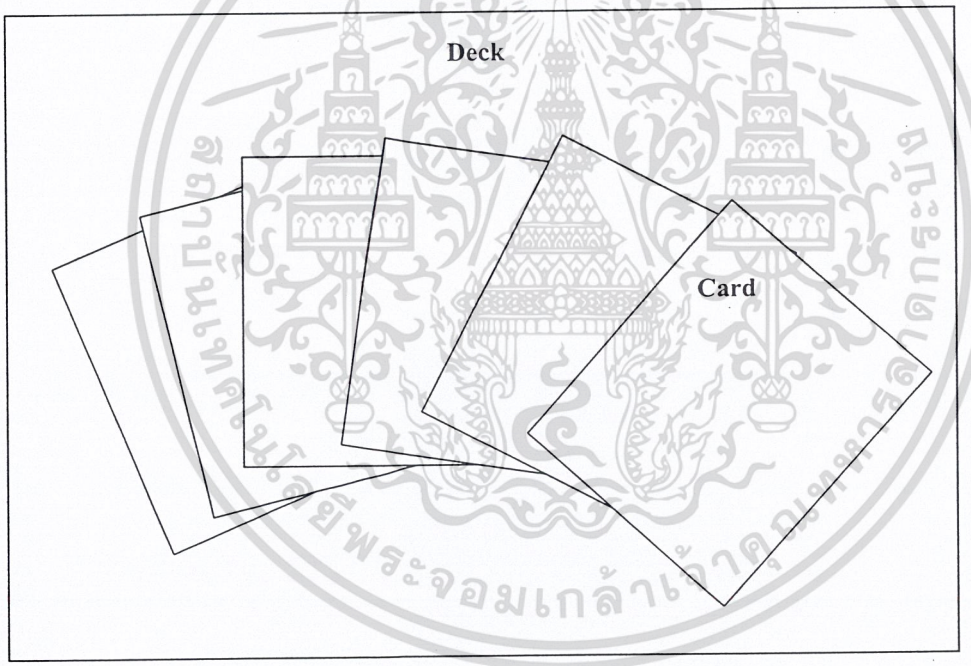
46153

2.5 การเขียนโปรแกรมภาษา WML

2.5.1 โครงสร้าง ของเอกสาร WML

โครงสร้างของเอกสาร WML นั้นถูกออกแบบมาให้มีการทำงานเหมือนกับไพ่ โดยที่ไพ่แต่ละใบหรือการ์ด (Card) เปรียบได้กับแว็พเพจหนึ่งหน้า และเมื่อรวมการ์ดหรือแว็พเพจหลาย ๆ หน้าเข้าด้วยกันเป็นสำรับ หรือที่เรียกว่า เด็ค (Deck) ก็จะได้เป็นแว็พไซต์ ซึ่งการที่ WML จะถูกออกแบบให้มีลักษณะเหมือนกับไพ่หลาย ๆ ใบในสำรับหนึ่ง ก็เพราะว่าอุปกรณ์แว็พจะได้ไม่เสียเวลาในการส่งคำร้องขอไปยังเซิร์ฟเวอร์บ่อย ๆ ซึ่งเป็นการช่วยลดเวลาและแบนด์วิดธ์ ลงได้

ดังนั้น โครงสร้างที่ใหญ่ที่สุดของเอกสาร WML ก็คือเด็คซึ่งก็คือเนื้อหาทั้งหมดที่อยู่ในเอกสาร WML ในแต่ละเด็คจะประกอบไปด้วยกลุ่มเอกสารย่อย ๆ ซึ่งก็คือการ์ด โดยในแต่ละการ์ดจะเป็นที่เก็บข้อมูลต่างๆ ที่ต้องการให้แสดงที่หน้าจอ รวมไปถึงคำสั่งต่าง ๆ ที่ต้องการให้เอกสาร WML ตอบสนองต่อความต้องการของผู้ใช้ดังแสดงในรูปที่ 2-8



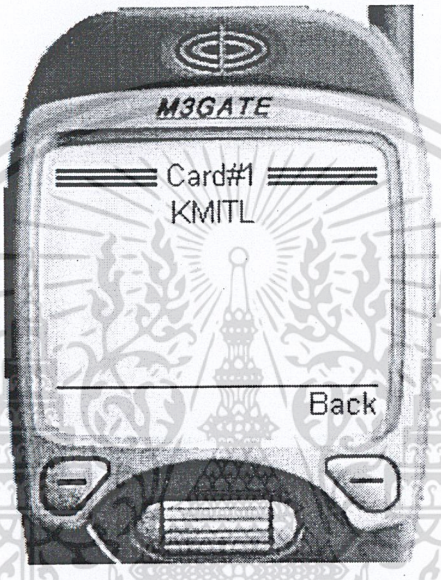
รูปที่ 2-8 เด็ค และการ์ดของภาษา WML

โดยเอกสาร WML นั้นอย่างน้อยต้องประกอบไปด้วยส่วนต่าง ๆ ที่ได้แสดงให้เห็นดังนี้

บรรทัดที่	เอกสาร WML
1	<?xml version=”1,0”>
2	<!DOCTYPE wml PUBLIC “-//WAPFORUM//DTD WML 1.1//EN”
3	“http://www.wapforum.org/DTD/wml_1.1.xml”>
4	<wml>
5	<card>

```
6       <!--สามารถเขียนข้อมูลลงไปได้-->
7       <p>KMITL</p>
8       </card>
9       </wml>
```

ซึ่งผลลัพธ์ที่ปรากฏอยู่บนโทรศัพท์มือถือจะเป็นดังรูปที่ 2-9



รูปที่ 2-9 แสดงผลลัพธ์ของไฟล์เอกสาร

ความหมายในแต่ละบรรทัดมีดังนี้

บรรทัดที่	ความหมาย
1 - 3	เป็นส่วนที่เรียกว่า เฮดเดอร์ (Header) ซึ่งเป็นส่วนที่บอกบราวเซอร์ว่าเอกสารนี้เป็นเอกสาร WML เวอร์ชัน 1.0 ในเอกสาร WML ทุกเอกสารจะต้องมีเฮดเดอร์นี้ มิฉะนั้นจะทำให้การแสดงข้อมูลไม่ถูกต้อง
4 และ 9	เป็นจุดเริ่มต้นและจุดสิ้นสุดของเด็ค (<wml>…</wml>) จะเห็นว่าลักษณะของแท็กจะคล้ายกับ HTML
5 และ 8	เป็นจุดเริ่มต้น และจุดสิ้นสุดของการ์ด (<card>…</card>)
6	เป็นหมายเหตุ <- -…- -> ข้อความที่อยู่ในแท็กหมายเหตุจะไม่ถูกแสดงผลในบราวเซอร์
7	เป็นส่วนข้อมูลที่ออกสู่บราวเซอร์ (…) ข้อความต่าง ๆ ที่อยู่ภายในแท็กนี้จะถูกนำมาแสดงผลที่บราวเซอร์

2.5.2 แท็กต่าง ๆ ในภาษา WML

แท็กต่าง ๆ ในภาษา WML นั้นประกอบด้วยแท็กต่าง ๆ มากมายดังตารางต่อไปนี้

ชื่อแท็ก	คำอธิบายแท็ก
a	ใช้เชื่อมโยงระหว่าการ์ดหรือ หน้าเว็บเพจ ซึ่งจะแตกต่างกับ do ตรงที่จะไม่มีการแสดงในเมนูให้เห็น
access	ใช้ในการควบคุมการเข้าถึงข้อมูลของผู้เข้าชมหรือให้เข้าชมเฉพาะบางกลุ่มก็ได้
anchor	ใช้เชื่อมโยงระหว่าการ์ดหรือ หน้าเว็บเพจ ซึ่งจะแตกต่างกับ do ตรงที่จะไม่มีการแสดงในเมนูให้เห็น
b	ทำให้ตัวอักษรหนาขึ้น
big	ทำให้ตัวอักษรใหญ่ขึ้น
br	ใช้สำหรับกำหนดให้ขึ้นบรรทัดใหม่สำหรับข้อความที่อยู่ต่อจากแท็กนี้
card	กำหนดจุดเริ่มต้นและจุดสิ้นสุดของการ์ด ในเด็คเด็คหนึ่งสามารถมีการ์ดได้หลาย ๆ การ์ด
do	ใช้เชื่อมโยงจากการ์ดที่แสดงอยู่ไปยังการ์ดอื่น หรือ URI อื่น ๆ ได้โดยการรับจากเชื่อมโยงโดยใช้เมนู และปุ่มต่าง ๆ ที่แสดงบนหน้าจอ
em	ทำตัวอักษรให้เด่นขึ้น ซึ่งจะเห็นเป็นตัวเอียง
fieldset	ช่วยในการทำให้แว็พบราวเซอร์ แสดงส่วนรับข้อมูลต่าง ๆ ได้ดีขึ้น
go	ใช้ในการกำหนดให้แว็พบราวเซอร์เปลี่ยนหน้าของแว็พเพจไปตามตำแหน่งที่กำหนดในแอททริบิวต์
head	เป็นส่วนที่ใช้ในการเก็บข้อมูลและรายละเอียดที่มีความสัมพันธ์กับเอกสาร WML
i	ทำให้ตัวอักษรเป็นตัวเอียง
img	ใช้ในการใส่รูปลงในแว็พเพจ
Input	ใช้รับข้อมูลที่เป็นตัวอักษร
meta	ใช้ในการกำหนดคำอธิบาย เพื่อให้โปรแกรมสืบค้น (Search Engine) ได้ค้นหาแว็พเพจได้
noop	เป็นแท็กที่ไม่ทำอะไรเลย และใช้ในการยกเลิกการทำงานของ template ได้
noevent	ใช้ในการกำหนดเหตุการณ์ต่าง ๆ เช่น ontimer ได้
optgroup	ทำหน้าที่เป็นตัวจัดกลุ่มให้กับตัวเลือกซึ่งทำงานร่วมกับ select และ option
option	ใช้สำหรับทำหน้าที่เป็นตัวแสดง ตัวเลือกในลักษณะเมนู โดยทำงานร่วมกับ select
p	ใช้สำหรับแสดงข้อมูลออกสู่จอภาพ โดยมีการกำหนดการจัดวางตัวอักษร เช่น ชิดซ้าย ชิดขวา และตรงกลาง และการขึ้นบรรทัดใหม่เอง ถ้าข้อความยาวเกินกว่าหน้าจอ
postfield	ใช้ในการส่งค่าต่าง ๆ ที่รับจากส่วนรับข้อมูลไปยัง cgi
prev	ใช้สำหรับแสดงลิงค์กลับไปที่หน้าเก่า เพื่อให้สามารถย้อนกลับไปยังหน้าที่ผ่านมาแล้วได้
refresh	ใช้ในการโหลดหน้าแว็พเพจ ขึ้นมาแสดงใหม่หรือเพื่อโหลดข้อมูล รวมทั้งค่าของตัวแปรล่าสุด ออกมาแสดงอีกครั้งหนึ่ง

ชื่อแท็ก	คำอธิบายแท็ก
select	ใช้สำหรับทำหน้าที่เป็นตัวแสดงตัวเลือก ในลักษณะเมนู โดยทำงานร่วมกับ option
setvar	ใช้สำหรับกำหนดค่าของตัวแปร
small	ทำให้ตัวอักษรเล็กลง
strong	ทำให้ตัวอักษรเด่นขึ้น ซึ่งจะเป็นตัวหนาขึ้น
table	ใช้กำหนดจุดเริ่มต้น และจุดสิ้นสุดของตาราง
td	ใช้กำหนดจุดเริ่มต้น และจุดสิ้นสุดของคอลัมน์หนึ่งในตาราง
template	ทำหน้าที่เกี่ยวกับการกำหนดคำสั่ง งาน หรือเหตุการณ์ต่าง ๆ ซึ่งคำสั่งของเหตุการณ์นี้สามารถนำไปใช้ในทุกการ์ดในเด็คนั้น
timer	ใช้กำหนดเวลาในการทำงานอย่างใดอย่างหนึ่ง เช่นกำหนดเวลาในการเชื่อมโยงไปยังอีกการ์ดหนึ่งเป็นต้น
tr	ใช้กำหนดจุดเริ่มต้น และจุดสิ้นสุดของแถวหนึ่งแถวในตาราง
u	ทำให้ตัวอักษรถูกขีดเส้นใต้
wml	เป็นจุดเริ่มต้น และจุดสิ้นสุดของเอกสาร WML

ตารางที่ 2-2 แสดงแท็กต่าง ๆ ของภาษา WML

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทที่ 3
โพรโตคอลที่ใช้ในการสื่อสารทาง WAP

3.1 URL(Uniform Resource Locator)

URL เป็นคำที่ย่อมาจาก Uniform Resource Locator เป็นหลักการกำหนดชื่ออ้างอิงของทรัพยากรต่างๆที่อยู่ภายในเครือข่ายอินเตอร์เน็ต หากเปรียบเทียบกับการจัดเก็บข้อมูลในคอมพิวเตอร์ทั่วไปก็คือ ชื่อของไฟล์, ชื่อของไดเร็คเทอรี หรือชื่อของโฟลเดอร์ เป็นต้น แต่เนื่องจาก URL นั้นต้องรองรับการทำงานภายใต้เน็ตเวิร์ก ดังนั้นรูปแบบของ URL จึงซับซ้อนมากกว่าชื่อไฟล์ หรือชื่อของคอมพิวเตอร์ทั่วไป ซึ่ง URL จะต้องสามารถบ่งบอกชื่อหรือแอดเดรสของเครื่องคอมพิวเตอร์ในเน็ตเวิร์ก โปรโตคอลที่ใช้งาน รวมทั้งพารามิเตอร์และออปชั่นต่างๆด้วย

สำหรับเวิลด์ไวด์เว็บที่ใช้งานเว็บเพจต่างๆนั้น URL สามารถระบุชื่อของแว็พเซิร์ฟเวอร์จนถึงที่เก็บไฟล์ WML ของแว็พเพจนั้นๆ และในการลิงค์ไปยังเพจอื่นด้วย

ซึ่งรูปแบบมาตรของ URL มาตรฐานประกอบด้วย

```
<Protocol>:<protocol><Protocol-specific name>
```

ส่วนต่างๆของรูปแบบ URL มีรายละเอียดดังนี้

1) <Protocol>

จะทำหน้าที่กำหนดโปรโตคอล (หรือบริการที่ต้องการ) ที่จะใช้งาน ตัวอย่างเช่น HTTP ใช้อ้างถึงแว็พไซท์, FTP ใช้อ้างถึงเซิร์ฟเวอร์ที่ต้องการดาวน์โหลดไฟล์ เป็นต้น

2) <Protocol-specific name>

เป็นส่วนที่กำหนดรายละเอียดของแต่ละโปรโตคอลเพื่อให้ทราบถึงรายละเอียดเพิ่มเติมในการใช้งาน เช่น ชื่อของเซิร์ฟเวอร์หรือไดเร็คทอรี่ที่เก็บไฟล์ เป็นต้น ดังนั้นจึงสามารถปรับรูปแบบ URL มาตรฐานใหม่ได้ดังนี้

```
<Protocol>://<user>:<password>@<server>:<port>/<path>
```

2.1) *<user>*

จะกำหนดชื่อของผู้ใช้งานพร้อมกับรหัสความปลอดภัย

2.2) *<password>*

จะต้องระบุในกรณีที่ใช้งานในบางโปรโตคอลที่ต้องการ เช่น การใช้งาน FTP ที่ระบุผู้ใช้แตกต่างจาก anonymous ต้องกำหนด Password เป็นต้น

2.3) *<server>*

จะใช้ระบุชื่อโดเมนของเซิร์ฟเวอร์ที่ต้องการใช้งาน หรือระบุเป็นหมายเลข IP ก็ได้

2.4) *<port>*

ในกรณีที่เซิร์ฟเวอร์มีการใช้งานหมายเลขพอร์ตพิเศษแตกต่างจากหมายเลขพอร์ตทั่วๆ ไปของแต่ละโปรโตคอลนั้นผู้ใช้สามารถระบุหมายเลขพอร์ตใน URL

2.5) *<path>*

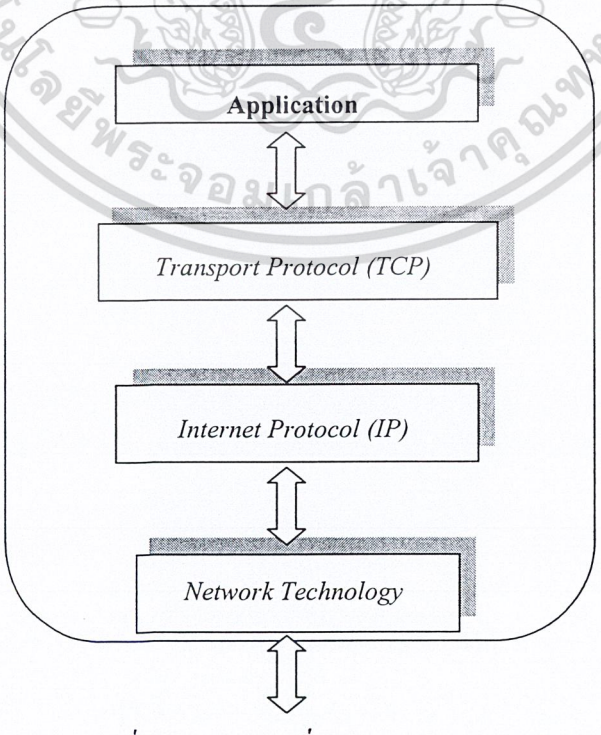
ใช้เมื่อต้องการอ้างถึงชื่อไฟล์หรือชื่อไดเร็คทอรี่ เช่น การดาวน์โหลดด้วยโปรแกรม FTP หรือการดึงไฟล์แว็พเพจจากไดเร็คทอรี่ซึ่งต่างจากที่กำหนดไว้

ตัวอย่างการใช้งานในรูปแบบของ URL เช่น http://161.246.5.239/public/index.wmlโดย http จะเป็นส่วนที่กำหนดโปรโตคอลที่ใช้งาน ในที่นี้ http หมายถึงการใช้งานเวิร์ดไวด์เว็บทั่วไป ต่อมา 161.246.5.239 คือส่วนที่เป็นชื่อของเว็บไซท์ที่ต้องการติดต่อ ส่วน public เป็นส่วนที่กำหนดชื่อของไดเร็คทอรี่ในเครื่องที่เป็นแว็พเซิร์ฟเวอร์นั้นๆ และไฟล์ WML ที่ต้องการเรียกมาใช้งานคือไฟล์ชื่อ index.wml

ซึ่งในโครงงานนแว็พบราวเซอร์ภาษาไทยนี้ใช้การส่งเดต้าผ่าน http โปรโตคอล โดยหลักการทำงานของ http โปรโตคอลมีดังนี้คือ

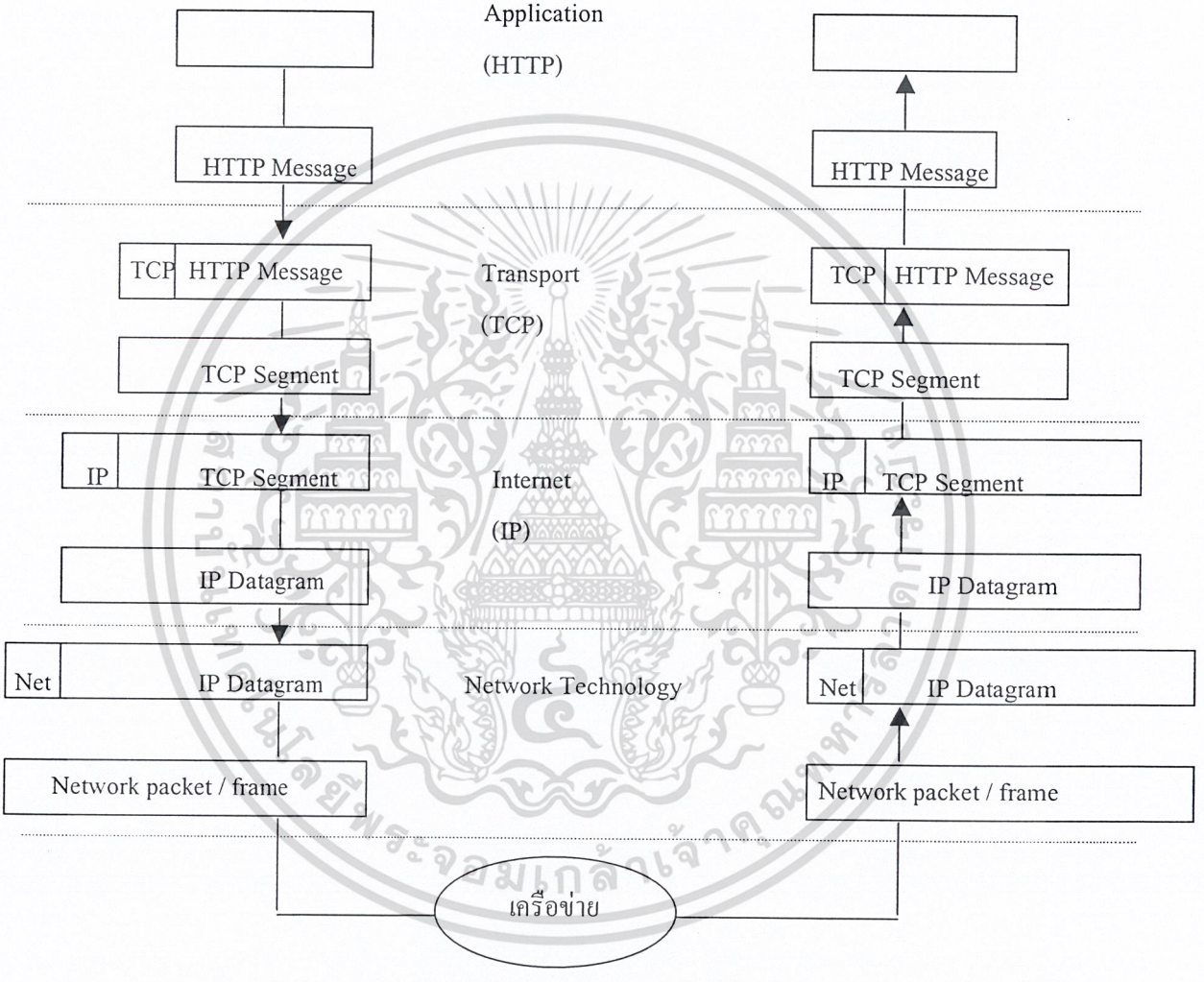
3.2 HTTP Protocol

HTTP มาจากคำว่า Hypertext Transfer Protocol ซึ่งเป็นโปรโตคอลที่ใช้ในการส่งเดต้าต่าง ๆ ในโลกของเวิร์ดไวด์เว็บ เดต้าต่างๆเหล่านี้โดยทั่วไปมักจะถูกเรียกว่า *Resource* โดย Resource เหล่านี้อาจจะเป็นไฟล์ เช่น HTML ไฟล์, image ไฟล์ หรือคำสั่งต่าง ๆ (Query String) เช่น คำสั่งที่ส่งไปที่ cgi โปรแกรม หรืออาจจะ binary stream ในกรณีของการ download/upload ไฟล์ หรืออาจจะเป็นสิ่งอื่น ๆ อีกมากมาย ตามแต่จะกำหนดขึ้น HTTP เป็นโปรโตคอลที่อยู่ในส่วนของ แอพพลิเคชั่นเลเยอร์ ในโปรโตคอล สแตก โดยเดต้าต่าง ๆ จากเลเยอร์ นี้จะถูกจากส่งผ่านไปยังเลเยอร์ อื่น ๆ ที่ต่ำกว่า



รูปที่ 3-1 เลเยอร์ที่ใช้ในการแลกเปลี่ยนข้อมูลผ่าน HTTP โปรโตคอล

โดยในการรับส่งข้อมูลจริงๆแล้วในแต่ละชั้น จะแปะเฮดเดอร์ของตนไปจากเลเยอร์บนลงสู่เลเยอร์ที่ต่ำกว่าเมื่อเลเยอร์ที่ต่ำกว่าได้รับก็จะไม่สนใจว่าข้อมูลเป็นอะไร และจะแปะเฮดเดอร์ของเลเยอร์ตนเพิ่มไป จะเป็นเช่นนี้ไปทุกชั้น และเมื่อถึงชั้นสุดท้าย(ล่างสุด) ก็จะส่งออกไปยังเครือข่าย เมื่อผู้รับได้รับก็จะนำ เดต้าที่ได้รับมาก็จะนำมาถอดเฮดเดอร์ที่แปะมากับเดต้าออก เสร็จแล้วจึงส่งขึ้นไปยังเลเยอร์ชั้นบนของตนจนถึงเลเยอร์บนสุด ดังรูป

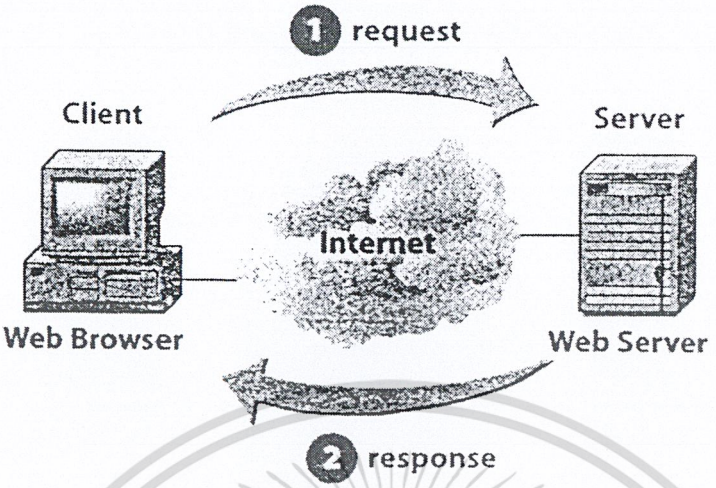


รูปที่ 3-2 การส่งข้อมูลผ่าน HTTP โปรโตคอล

HTTP เป็น network protocol ที่ใช้หลักการของ client-server model ในการติดต่อสื่อสารซึ่งหลักการทำงานอย่างคร่าว ๆ มีดังนี้

1. HTTP Client จะทำการสร้างคอนเนคชั่นไปหา HTTP Server หลังจากนั้น HTTP Client จะทำการส่งคำสั่ง (request) ซึ่งอยู่ในรูปของ message ไปให้ HTTP Server เพื่อทวงถามถึง resource ที่ต้องการ

2. HTTP Server จะทำการตีความคำสั่งที่ได้และส่งผล (response) ซึ่งเป็น resource ที่ HTTP Client ต้องการกลับมา (ผลที่ส่งกลับมาจะเป็นลักษณะของ message คล้ายกับ requet ของ HTTP Client ที่ส่งมาให้ HTTP Server) ดังรูป



รูปที่ 3-3 การส่ง Request และ Responses message ระหว่าง client และ server

3.2 HTTP Structure

จากหลักการทำงานของ HTTP ข้างต้น เราจะเห็นว่าสำหรับการติดต่อสื่อสารระหว่าง Client กับ Server แล้ว ตัว message จะเป็นตัวกลางที่ใช้ในการติดต่อสื่อสารเสมอ และเพื่อให้ง่ายต่อการใช้งานรูปแบบของ message ที่ใช้ใน request หรือ response จึงมีลักษณะคล้าย ๆ กันโดยจะใช้ text เป็นหลักซึ่งโครงสร้างของ message จะประกอบไปด้วย

1. บรรทัดเริ่มต้น (initial line)
2. Header(s)
3. บรรทัดว่าง (a blank line) ซึ่งก็คือ CRLF* หรือการเว้นหนึ่งบรรทัด
4. message body ซึ่งอาจจะใช้บรรจุไฟล์, คำสั่ง (Query String) หรืออาจจะเป็น output ที่มาจาก server โดยส่วนนี้จะมีหรือไม่มีก็ได้ขึ้นอยู่กับจุดประสงค์ของการใช้งาน

* CRLF = Carriage return (\r) และ Line Feed (\n) ซึ่งก็คือ \u00A และ \u00D ใน ACSII นั่นเอง

รูปแบบโดยทั่วไปของ message จะเป็น

<initial line>
Header1: value1
Header2: value2
Header3: value3
[blank line]
<message body, optional >

รูปที่ 3-4 HTTP Message

3.2.1 Initial Request Line

initial line ของ request จะแตกต่างจาก initial line ของ response เล็กน้อยโดย initial line ของ request จะมีสามส่วนคือ

1. ชื่อของ method เช่น GET, POST, HEAD, TRACE
2. local path ของ resource ที่ client ต้องการ
3. เวอร์ชั่นของ HTTP/x.x ที่ HTTP Client ใช้

สามส่วนนี้จะประกอบกันเป็น initial line โดยแต่ละส่วนจะถูกแยกออกจากกันโดยใช้ช่องว่าง (space) ดังตัวอย่างข้างล่าง

GET /public/index.wml HTTP/1.0

ตัวอย่างนี้ใช้ method ที่ชื่อ GET เพื่อขอ resource (ในที่นี้คือไฟล์) ชื่อ /public/index.wml โดยใช้ HTTP/1.0 โปรโตคอล

3.2.2 Initial Response Line

โดยทั่วไป initial response line มักจะเรียกว่า status line ซึ่งประกอบไปด้วยสามส่วนย่อย คือ

1. เวอร์ชั่นของ HTTP/x.x ที่ server ใช้สำหรับส่ง message
2. response status code ซึ่งเป็นตัวบอกว่าผลของ request ที่ Client ส่งมาเป็นอย่างไร
3. reason phase เป็นตัวอธิบายความหมายของ response status code อีกทีหนึ่ง

โดย response status code จะอยู่ในลักษณะของเลขสามหลัก โดยหลักแรกจะบอกถึงความหมายโดยทั่ว ๆ ไปของ response

1xx	เป็น code ที่ใช้บอกถึงเหตุการณ์ต่าง ๆ ที่เกิดขึ้นในการสื่อสารระหว่าง Client และ Server
2xx	เป็น code ที่บ่งบอกว่า request ที่ส่งจาก Client ถูกทำให้ complete แล้ว
3xx	เป็น code ที่บอกให้ Client ทำการ redirect ไปที่ url อื่นแทน
4xx	เป็น code ที่บ่งบอกถึงความผิดพลาดที่เกิดขึ้นจากส่วนของ Client
5xx	เป็น code ที่บ่งบอกถึงความผิดพลาดที่เกิดขึ้นจากส่วนของ Server

ตารางที่ 3-1 ตารางแสดง status code

3.2.3 Header(s)

โดยทั่วไป header จะเป็นส่วนที่บอกถึงรายละเอียดของ request (ที่กำลังส่งไปให้ Server) หรือ response (ที่กำลังส่งกลับมายัง Client) ยกตัวอย่างเช่น

- Client จะส่ง Header ที่บอกถึงชนิดและขนาดของข้อมูลที่อยู่ข้างใน message body ในกรณีที่ Client ต้องการ upload ไฟล์ไปยัง server

- Server อาจจะส่ง Header ที่เกี่ยวกับชนิดและขนาดของ resource ที่ Client กำลังจะได้รับโดยใช้ Content-Type และ Content-Length

header จะเป็นลักษณะของ text format โดยในหนึ่งบรรทัดจะถูกใช้สำหรับหนึ่ง header ซึ่งจะอยู่ในรูปของ "Header-Name: value" และจบด้วย CRLF

3.2.4 Message Body

เมื่อไรก็ตามที่ client หรือ server ต้องการส่งเดต้าไปกับ message, ส่วนที่เป็น message body จะเป็นส่วนที่ใช้สำหรับเก็บเดต้าดังกล่าว ในกรณีของ client ตัว message body อาจใช้สำหรับบรรจุไฟล์ที่ต้องการ upload หรือใช้สำหรับเก็บเดต้าที่มาจาก element ต่าง ๆ ของ HTML form แล้วส่งไปยัง server ก็ได้ ในกรณีของ server ตัว message body จะเป็นส่วนที่ใช้เก็บ resource ที่ client ทวงถามหรืออาจใช้สำหรับเก็บคำอธิบาย ต่าง ๆ ในกรณีที่มี error เกิดขึ้นก็ได้ ถ้า message มีส่วนของ message body, client หรือ server มักจะเพิ่ม header ที่ช่วยบอกถึงรายละเอียดของ message body ดังกล่าวด้วย ยกตัวอย่างเช่นถ้า server ต้องการส่ง resource ที่เป็นไฟล์ image มาให้ client, header ที่ถูกเพิ่มขึ้นมาก็จะเป็น

Content-Type: image/gif เป็น header ที่บ่งบอกถึง MIME type ของ data ที่อยู่ใน message body

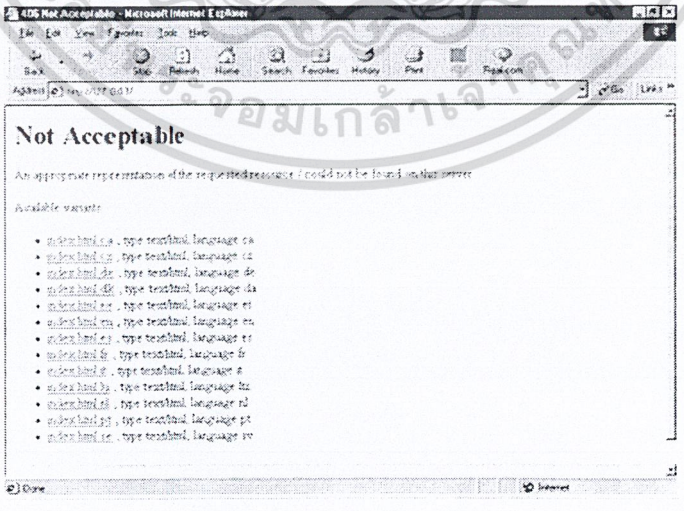
Content-Length: 1026 เป็น header ที่บ่งบอกถึงขนาดของ message body ในหน่วย byte

3.3 การติดตั้ง Apache

ในการทดลองโครงงานนี้ได้ใช้ Apache Webserver1.3.12 ในการทดลอง ซึ่งหาดาวน์โหลดได้จาก **www.apache.org/dist/** เมื่อติดตั้งเสร็จแล้วก็ทดลองรันโดยเข้าไปที่ Start -> Programs -> apache Web Server -> Start Apache เมื่อรันแล้วจะได้ข้อความลักษณะดังนี้

Apache/1.3.12 (Win32) running...

เมื่อทดลองใน Web browser โดยระบุ Url เป็น 127.0.0.1 จะได้ข้อความดังรูป



แสดงว่า Apache ของเราสามารถรันได้แล้ว ต่อมาเราต้องทำให้ Apache รู้จักกับเอกสารตระกูล WAP ทำได้โดยเข้าไปกำหนด MIME TYPE เพิ่มโดยสามารถเข้าไปแก้ไขได้ที่ไฟล์ httpd.conf ซึ่งเก็บอยู่ที่ ./conf โดยหาคำว่า Addtype แล้วพิมพ์คำเหล่านี้เพิ่มลงไป

```
AddType text/vnd.wap.wml wml
AddType application/vnd.wap.wmlc wmlc
AddType text/vnd.wap.wmlscript wmls
AddType application/vnd.wap.wmlscriptc wmlsc
AddType image/vnd.wap.wbmp wbmp
```

และต้องเพิ่มข้อความต่อไปนี้ ในไฟล์ ".conf/web.xml"

```
<web-app>
    <mime-mapping>
        <extension>
            wml
        </extension>
        <mime-type>
            text/vnd.wap.wml
        </mime-type>
    </mime-mapping>
</web-app>
```

เมื่อเข้าไปแก้ไข ไฟล์ httpd.conf เรียบร้อยแล้วให้ save แล้วก็ปิดโปรแกรม Apache แล้วรันใหม่เป็นอันเรียบร้อย

การตั้งค่าเพื่อให้ไฟล์ชื่อ "index.wml" เป็นไฟล์เริ่มต้นในการตอบสนอง HTTP request ต้องเพิ่มข้อความดังนี้ ในไฟล์ ".conf/web.xml" ให้เป็นคอนเทนต์ของ web-app

```
<welcome-file-list>
    <welcome-file>
    index.wml
    </welcome-file>
</welcome-file-list>
```

บทที่ 4

XML Parser

4.1 XML

XML(Extensible Markup Language) เป็นภาษาประเภท Markup Language คล้ายกับ html แต่ต่างกันตรงที่ xml สามารถนิยามภาษาอื่นได้ เช่นภาษา wml ก็มีรากฐานมาจาก xml โดย xml จะมีกฎเกณฑ์การใช้ดังนี้

- เมื่อมีแท็กเปิดต้องมีแท็กปิดเสมอ เช่น

   ย่อหน้าที่หนึ่ง ,
 เป็นต้น

- อักษรตัวใหญ่และตัวเล็กมีความแตกต่างกัน เช่น

  <Memo> ถูกต้อง </Memo> , <Memo> ผิด </memo>

- แท็กต้องปิดและเปิดเป็นวงๆไปตามลำดับ เช่น

  <I> ถูกต้อง </I> , <I> ผิด </I>

- แอตทริบิวต์ของ xml ต้องอยู่ในเครื่องหมายฟันหนู(“) เสมอ เช่น

4.2 XML Parser

ในการที่พาร์สเซอร์จะนำเอกสาร XML มาทำวิเคราะห์โครงสร้างนั้น ในปัจจุบันมีมาตรฐานหรือวิธีการหลักๆอยู่ 2 อย่างได้แก่ DOM (Document Object Model) และ SAX (Simple API for XML) ซึ่งทั้งสองวิธีต่างก็มีข้อดีและข้อเสียอยู่ในตัว ดังนั้นจึงได้มีบางคนพยายามคิดวิธีการอื่นๆ ขึ้นมา แต่อย่างไรก็ตาม ก็ยังคงยึดแนวทางของ DOM และ SAX เป็นหลักอยู่นั่นเอง

4.3 มองโครงสร้างเอกสารเป็นทรีด้วย DOM

XML DOM คือแอพพลิเคชันโปรแกรมมิงอินเทอร์เฟซ (AppplicationProgramming Interface, API)สำหรับเอกสาร XML ซึ่งจะกำหนดว่าเอกสาร XML จะมีการเข้าถึงและการโยกย้ายรวมไปถึงการจัดการต่างๆ ได้อย่างไรบ้าง DOM เป็นมาตรฐานของ W3C โดยมีเป้าหมายคือต้องการสร้างอินเทอร์เฟสที่เป็นมาตรฐานและจะต้องใช้กับแอพพลิเคชันได้หลากหลาย ซึ่ง DOM ก็สามารถใช้ได้กับภาษาโปรแกรมมิ่งหรือระบบปฏิบัติการใดก็ได้ ด้วยวิธีการแบบ DOM โปรแกรมเมอร์จะสามารถสร้างเอกสาร XML ขึ้นมา เข้าไปในโครงสร้างระดับต่างๆ ได้ รวมทั้งเพิ่ม ลบ และแก้ไขอีลิเมนต์ต่างๆ ของเอกสารได้

4.3.1 มุมมองของ DOM

การทำงานของ DOM จะเริ่มต้นเมื่อโปรแกรมหรือแอพพลิเคชันไปเรียก (call) พาร์สเซอร์ให้ไปโหลดเอกสาร XML เข้าสู่หน่วยความจำของคอมพิวเตอร์ เมื่อเอกสารถูกโหลดเสร็จเรียบร้อยแล้ว DOM ก็จะทำให้โปรแกรมสามารถดึงข้อมูลมาใช้งานหรือดำเนินการต่างๆ กับข้อมูลได้

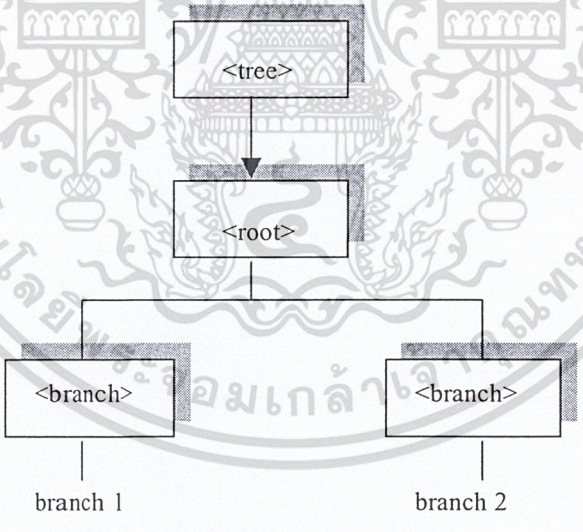
DOM จะมีมุมมองไปยังเอกสาร XML เป็นโครงสร้างแบบต้นไม้หรือทรี (tree) ระดับบนสุดของทรีเราจะเรียกว่า “documentElement” ซึ่งอีลิเมนต์ดังกล่าวจะแตกสาขาออกไปเป็นอีลิเมนต์ย่อยๆ ตั้ง

แต่ 1 อีลิเมนต์ขึ้นไป โดยอีลิเมนต์ย่อยที่แตกออกจาก documentElement เราจะเรียกว่า “โหนดลูก” (childNodes)

ลองพิจารณาจากเอกสารซึ่งมีรายละเอียดของโค้ดดังนี้

```
<tree>
    <root>
        <branch> branch1 </branch>
        <branch> branch2 </branch>
    </root>
</tree>
```

DOM จะมองเอกสารข้างต้นเป็นทรีดังนี้คือ



การจะเข้าถึงในแต่ละโหลดในทรีเราจะใช้ Node Interface Model โดย Microsoft XML parser จะสนับสนุนฟังก์ชันที่จำเป็นทั้งหมดที่ใช้ในการเข้าถึงแต่ละโหนดของทรี, ค่าแอตทริบิวต์ของโหนด และจัดการกับค่าต่างๆของโหนด อาทิ การเพิ่ม, ลบ และการแก้ไข เป็นต้น

4.3.2 จุดเด่นของ DOM

- โปรแกรมจะพาร์สเอกสารเพียงครั้งเดียว เมื่อ DOM ได้สร้างทรีขึ้นมาแล้ว ทรีนั้นก็จะอยู่ในหน่วยความจำตลอดจนกว่าเราจะสั่งยกเลิก และในขณะที่ทรีอยู่ในหน่วยความจำ เราก็สามารถเข้าถึงข้อมูลที่อยู่ภายในทรีได้อย่างรวดเร็ว
- ข้อมูลที่สร้างเป็นทรีจะทำให้ข้อมูลเกิดความสัมพันธ์ ทำให้การเรียกใช้งานทำได้ง่าย

4.3.3 จุดด้อยของ DOM

- ในกรณีที่เอกสารมีขนาดใหญ่และซับซ้อน การสร้างทรีจะทำได้ช้า และกินหน่วยความจำมาก
- ข้อมูลที่สร้างเป็นทรีจะทำให้ข้อมูลเกิดความสัมพันธ์ ทำให้การเรียกใช้งานทำได้ง่าย
- ผู้พัฒนาโปรแกรมจำเป็นจะต้องเข้าใจถึงโครงสร้างโดยรวมของเอกสารเป็นอย่างดี จึงจะสามารถเขียนโปรแกรมโดยใช้ DOM ได้ กล่าวคือในส่วนของการเขียนโค้ดด้วยวิธีการของ DOM นั้นจะค่อนข้างยาวและยุ่งยากอยู่พอสมควร

4.4 มองโครงสร้างเอกสารเป็นเส้นโครงด้วย SAX

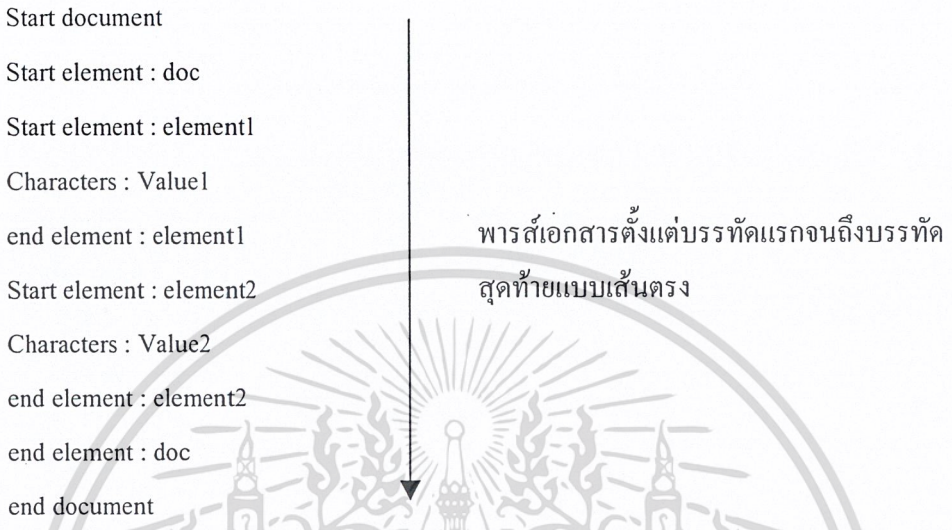
SAX (simple API for XML) เป็นแอพพลิเคชันโปรแกรมมิ่งอินเทอร์เฟซ (API) ที่มีลักษณะการทำงานก็ต่อเมื่อมีเหตุการณ์ (เช่น การร้องขอข้อมูล) ใดๆ เกิดขึ้นหรือเรียกโมเดลการทำงานแบบนี้ว่า event-based model กล่าวคือ SAX จะไม่มีการสร้างภาพรวมของเอกสารไว้ก่อนเลยว่าเอกสารมีโครงสร้างเป็นเช่นไร เมื่อแอพพลิเคชันมีการร้องขอมาจึงจะรายงานหรือทำงานตามเหตุการณ์ของการทำพาร์สซิง (parsing) ไปยังแอพพลิเคชันดังกล่าว

หลักการทำงานของ SAX จะมองเพียง "จุดเริ่มต้น" กับ "จุดสิ้นสุด" ของเอกสารและของอีลิเมนต์เท่านั้น และจะสนใจเฉพาะสิ่งที่ต้องการเท่านั้น เมื่อเปรียบเทียบกับวิธีการของ DOM ซึ่งนำข้อมูลทั้งหมดมาสร้างโครงสร้างข้อมูลทั้งหมดมาสร้างโครงสร้างข้อมูลเป็นทรี (tree) ทำให้ DOM มีข้อจำกัดในเรื่องหน่วยความจำซึ่งจะต้องใช้หน่วยความจำมากหากทรีที่สร้างมีขนาดใหญ่มากๆ ส่วน SAX จะเข้าถึงข้อมูลได้ง่ายกว่า การเขียนโปรแกรมทำได้ง่ายกว่าเพราะมองข้อมูลในลักษณะของเส้นตรงและสามารถพาร์สเอกสารที่มีขนาดใหญ่มากกว่าจำนวนหน่วยความจำที่เรียกได้ ดังนั้นโปรแกรมระบบฐานข้อมูลใหญ่ๆหลายบริษัทจึงนิยมที่จะใช้วิธีนี้กัน

เพื่อให้เกิดความเข้าใจในการทำงานของ SAX มากขึ้น เราจะมาพิจารณาจากตัวอย่างเอกสาร XML ง่ายๆดังนี้

```
<doc>
        <element1>Value 1</element>
        <element2>Value 2</element2>
</doc>
```

SAX จะมองเอกสารข้างต้นไปทีละบรรทัดๆ ในลักษณะเหตุการณ์เชิงเส้น (linear events) ดังนี้



ตัวอย่างการพาร์สแบบ SAX ข้างต้นเป็นการพาร์สทั้งเอกสาร ในทางปฏิบัติแล้วเรามักไม่ได้มีการพาร์สเอกสารทั้งหมด เมื่อเราได้ข้อมูลที่ต้องการแล้วก็จะหยุด วิธีการพาร์สแบบ SAX จึงทำงานได้อย่างรวดเร็วและเขียนโปรแกรมได้ง่ายมาก

4.4.1 จุดเด่นของ SAX

- เข้าถึงข้อมูลได้ง่าย และรวดเร็ว
- เขียนโปรแกรมอิมพลีเมนต์ได้ง่าย เนื่องจากการเข้าถึงข้อมูลเป็นลักษณะเส้นตรง
- ใช้หน่วยความจำน้อย สามารถเข้าถึงข้อมูลหรือเอาสารที่มีขนาดใหญ่กว่าหน่วยความจำที่มีของเครื่องได้

4.4.2 จุดด้อยของ SAX

- ไม่สามารถบอกความสัมพันธ์ของข้อมูลได้
- เป็นวิธีการที่ไม่ค่อยมีประสิทธิภาพหากเป็นการเข้าถึงข้อมูลเดิมบ่อยๆ

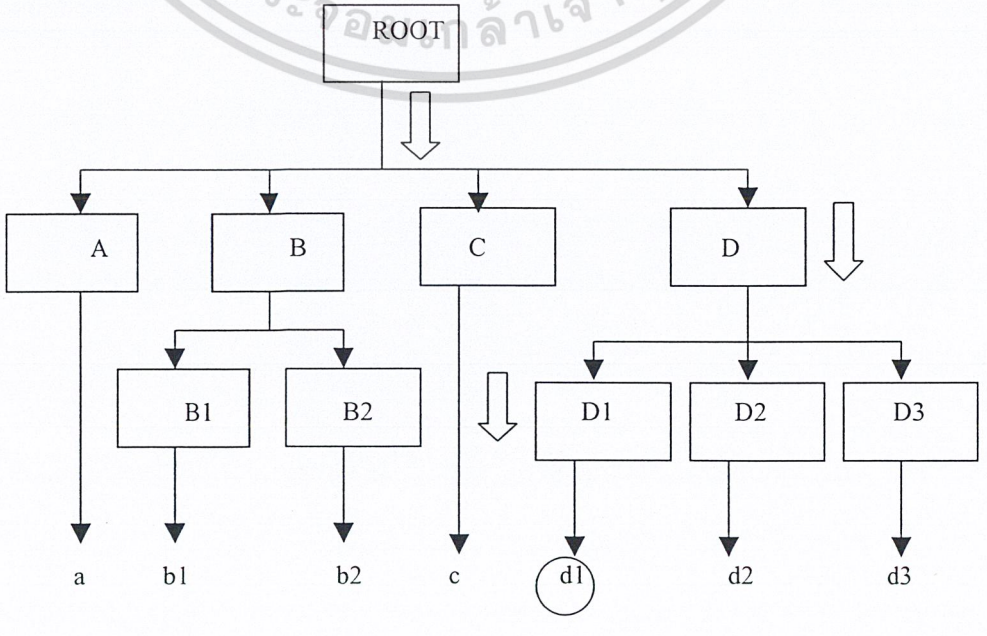
4.5 เปรียบเทียบการทำงานระหว่าง DOM และ SAX

ทั้ง DOM และ SAX ต่างมีจุดเด่นด้อยอยู่ในตัว การใช้งานจึงขึ้นอยู่กับดุลพินิจของผู้พัฒนาว่าแอพพลิเคชันที่ใช้เอกสาร XML นั้นมีความเหมาะสมกับวิธีการใดมากที่สุด

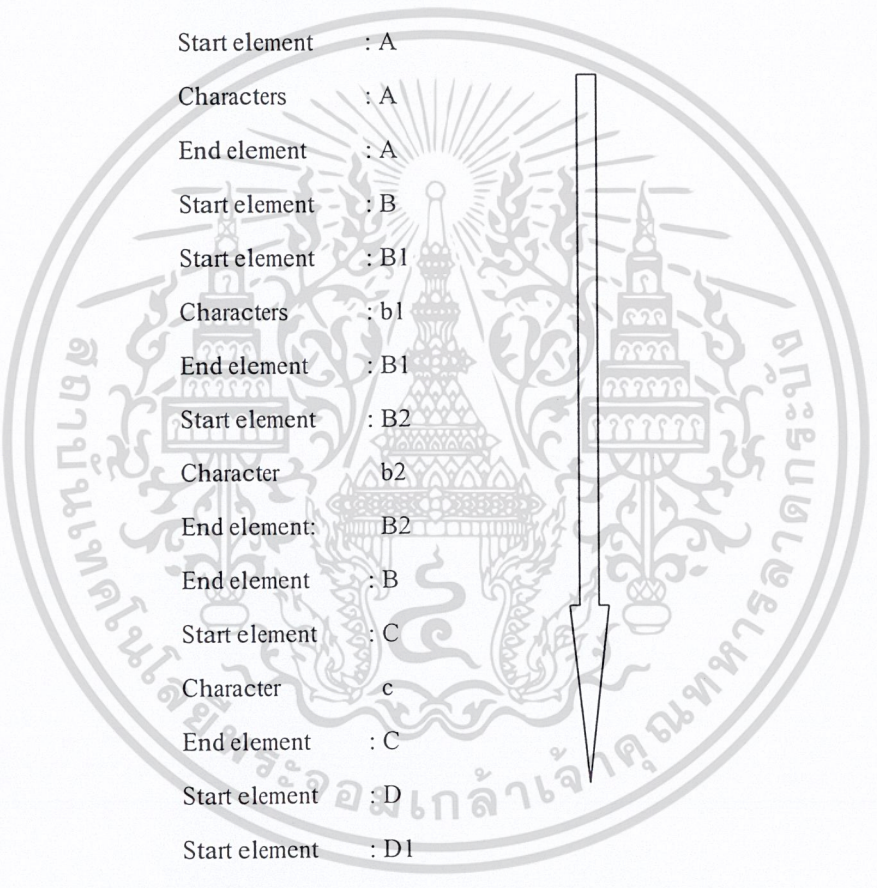
ลองเปรียบเทียบการทำงานระหว่าง DOM กับ SAX จากเอกสารชุดเดียวกัน โดยสมมติว่าเรามีเอกสาร XML ที่มีรายละเอียดดังนี้

```
<ROOT>
    <A>a</A>
    <B>
        <B1>b1</B1>
        <B2>b2</B2>
    </B>
    <C>c</C>
    <D>
        <D1>d1</D1>
        <D2>d2</D2>
        <D3>d1</D3>
    </D>
</ROOT>
```

จากเอกสาร XML ข้างต้น ถ้าเป็นวิธีการของ DOM จะมองเอกสารเป็นทรีที่เกิดจากการประกอบกันของโหนดต่างๆ และจะมองอีลเมนต์แต่ละตัวเป็นออบเจ็กต์ แล้วเมื่อเราต้องการเข้าถึงข้อมูลใด DOM ก็จะใช้วิธีเข้าไปยังโหนดต่างๆ ในทรีเพื่อหาข้อมูลที่ต้องการ เช่น เราต้องการหาค่าของอีลิเมนต์ D1 ว่ามีค่าเท่าใด ก็จะมีลักษณะการทำงานดังรูป



ส่วน SAX จะไม่มีการสร้างมุมมองใดๆ ของเอกสารไว้เลย การทำงานของ SAX จะเริ่มต้นก็ต่อเมื่อเกิดเหตุการณ์ใดๆขึ้นเท่านั้น เช่น เราต้องการรู้ว่าอีลิเมนต์ D1 มีค่าเท่าไร SAX ก็จะเริ่มอ่านเอกสารดังนี้



```
Start              document
Start element      : root
Start element      : A
Characters         : A
End element        : A
Start element      : B
Start element      : B1
Characters         : b1
End element        : B1
Start element      : B2
Character          b2
End element:       B2
End element        : B
Start element      : C
Character          c
End element        : C
Start element      : D
Start element      : D1
Characters         : d1
End element        : D1
.
.                  ส่วนที่เหลือนี้พาร์สเซอร์จะไม่สนใจ
```

ทั้ง DOM และ SAX ต่างมีจุดเด่นจุดด้วยไปคนละแบบ ในทางปฏิบัติจึงขึ้นอยู่กับลักษณะการใช้งานว่าจะเหมาะสมกับวิธีการพาร์สแบบใด เช่น หากให้ความสำคัญกับความสัมพันธ์ของข้อมูลก็ควรใช้

DOM แต่ถ้าต้องการวิธีการที่ทำให้เขียนโปรแกรมได้ง่ายและไม่เปลืองหน่วยความจำในการประมวลผลก็ควรใช้ SAX เป็นต้น ซึ่งในปัจจุบันก็มีอยู่หลายระบบงานที่ใช้ทั้งสองวิธีผสมผสานกันไป ในโครงงานนี้การใช้วิธีการพาร์สเซอร์แบบ sax เข้ามาช่วยเนื่องจาก เข้าถึงข้อมูลได้ง่ายรวดเร็วและที่สำคัญคือประหยัดหน่วยความจำ เนื่องจากหน่วยความจำบนโทรศัพท์มือถือมีอย่างจำกัด

4.6 การติดตั้ง JDOM

ก่อนอื่นที่จะใช้ XML Parser เราต้องติดตั้งคลาสของ jdom โดยสามารถดาวน์โหลดได้ที่ http://www.jdom.org/dist/source/ เมื่อเราดาวน์โหลดมาแล้วจะได้ไฟล์ jdom-b7.zip มาให้ทำการแตกไฟล์นี้ออกเสร็จแล้วให้ทำการติดตั้งโดยเข้าไปเรียกแบตช์ไฟล์ที่ชื่อว่า build โดยถ้ามีปัญหาให้ไป set variable ที่ชื่อว่า java_home ให้ไปที่ directory ของ jdk ที่เราทำการติดตั้งลงบนเครื่อง เช่น set java_home=c:\jbuilder4\jdk1.3
เมื่อลงเสร็จแล้วจะได้ไฟล์ jdom.jar มาซึ่งเป็นไฟล์ที่รวบรวมคลาสที่จำเป็นในการใช้ xml parser ซึ่งจะเก็บอยู่ใน /jdom-b7/build/

4.6.1 การใช้ XML Parser

เราต้อง import คลาสของ jdom เข้ามา โดยคลาสของ jdom จะอยู่ใน package org อีกทีนึง เช่น import org.jdom.*; เมื่อเราต้องการพาร์สซิ่งเอกสาร xml เราต้อง import org.jdom.input.SAXBuilder เช่น

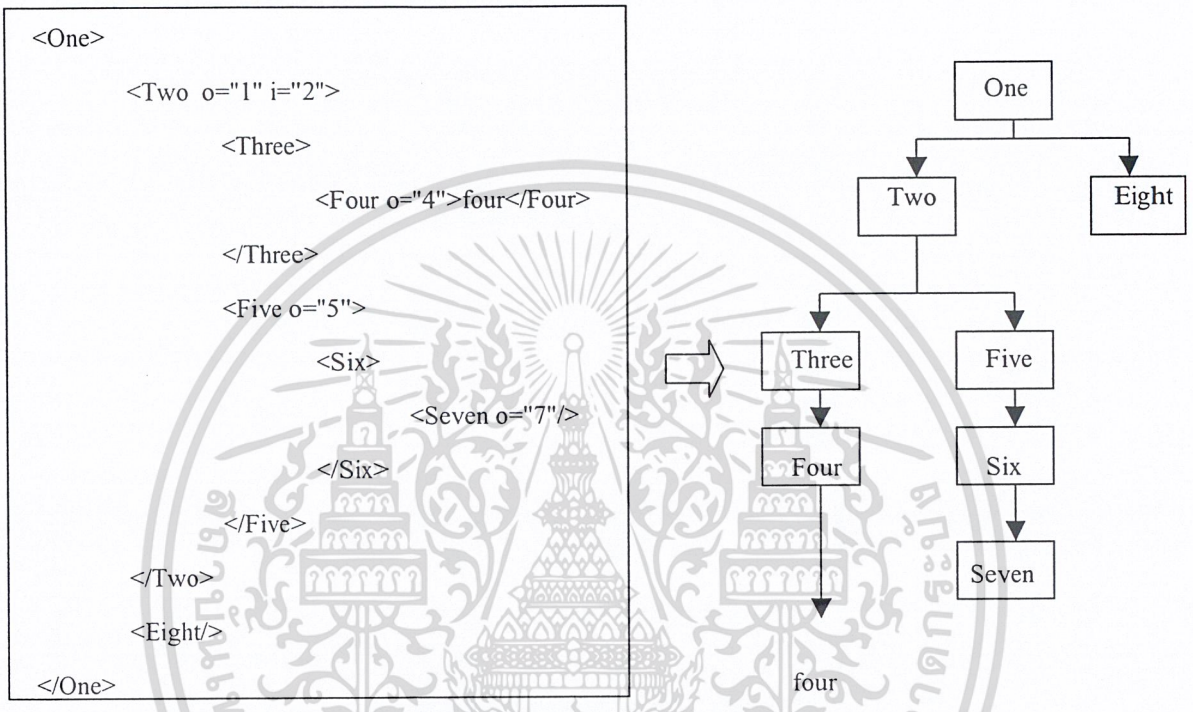
```
import org.jdom.*;
import org.jdom.input.SAXBuilder;

SAXBuilder builder= new SAXBuilder();
Document doc = builder.build("ไฟล์เอกสาร xml");
```

โดย method build นี้จะ throws JDOMException ออกมาเมื่อมี error เกิดขึ้นในขณะการพาร์สซิ่ง เพราะเอกสารไม่เป็นไปตามกฎเกณฑ์ของ XML ที่ได้กล่าวไว้ที่ข้างต้น
การเข้าถึงเอกสาร XML ก็มีอยู่หลาย method จะมีที่สำคัญๆ ซึ่งอยู่ในคลาส element เช่น

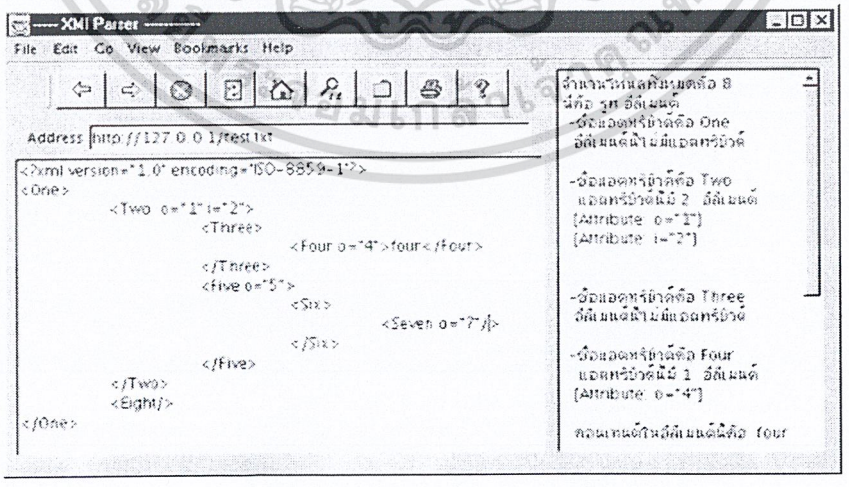
- String **getText**() เอาไว้นำเอา content ของ element นั้นๆออกมา
- public List **getAttributes**() เอาไว้นำชื่อและค่าของ Attribute ของ element นั้นๆ ออกมา
- public String **getAttributeValue**(String name) เอาไว้นำค่าของ Attribute ของ element นั้นๆ ออกมา
- public boolean **hasChildren**() เอาไว้ดูว่า element นี้มี element ลูกหรือไม่
- public List **getChildren**(String name) เอาไว้นำเอา element ลูกลงไป 1 ระดับออกมา

- String **getChildText**(String name) เอาไว้นำเอา content ของ element ลูกออกมา

นอกจากนี้ยังมี method อีกหลาย method ที่ช่วยสนับสนุนในการเข้าถึงเอกสาร XML นี้ ผลการทดลองการพาร์สเซอร์จากโปรแกรมโดยใช้ไฟล์ test.txt เป็นไฟล์ทดลองโดยเรียกผ่าน Apache



ไฟล์ test.txt *มองไฟล์ test.txt ในรูปของ tree*



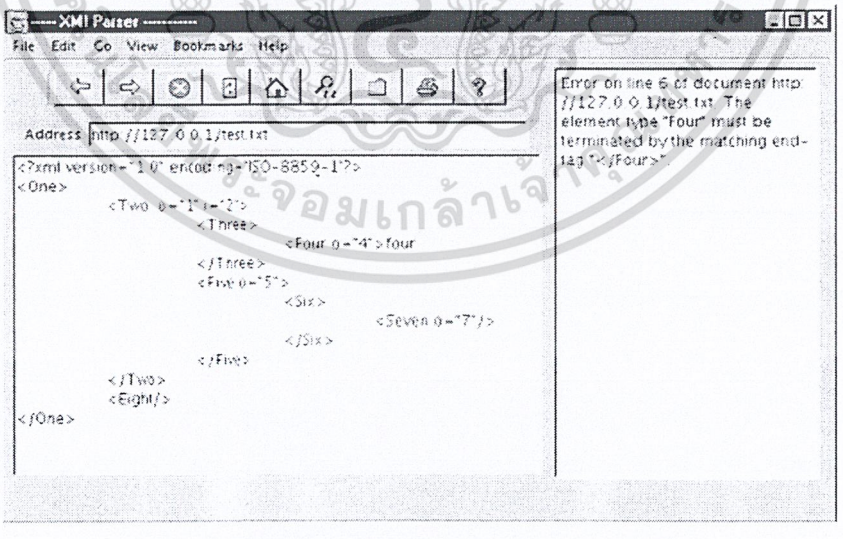
รูปที่ 4-1 ทำการพาร์สเซอร์โดยใช้โปรแกรม

เราใช้การท่องไปในทรีแบบ Depth First Search เพราะฉะนั้นจะไล่ลงมาตั้งแต่ node 1 (คือ node root) เรียงตามลำดับไปเรื่อยๆ จนถึง node ที่ 8 ได้ผลดังนี้

Number of all node of the tree is 8

- ชื่อแอตทริบิวต์คือ One แอตทริบิวต์นี้ไม่มีอีลีเมนต์
- ชื่อแอตทริบิวต์คือ Two แอตทริบิวต์นี้มี 2 อีลีเมนต์ [Attribute: o="1"] [Attribute: i="2"]
- ชื่อแอตทริบิวต์คือ Three แอตทริบิวต์นี้ไม่มีอีลีเมนต์
- ชื่อแอตทริบิวต์คือ Four แอตทริบิวต์นี้มี 1 อีลีเมนต์ [Attribute: o="4"]
  คอนเทนต์ในอีลีเมนต์นี้คือ four
- ชื่อแอตทริบิวต์คือ Five แอตทริบิวต์นี้มี 1 อีลีเมนต์ [Attribute: o="5"]
- ชื่อแอตทริบิวต์คือ Six แอตทริบิวต์นี้ไม่มีอีลีเมนต์
- ชื่อแอตทริบิวต์คือ Seven แอตทริบิวต์นี้มี 1 อีลีเมนต์ [Attribute: o="7"]
- ชื่อแอตทริบิวต์คือ Eight แอตทริบิวต์นี้ไม่มีอีลีเมนต์

ถ้าเราลบ </four> ในบรรทัดที่ 4 ออกจะทำให้เกิด error ขึ้นเนื่องจากผิดกฎของ XML ที่ว่ามีแท็กเปิดแล้วต้องมีแท็กปิด จะได้ผลคือ "Error on line 6 of document http://127.0.0.1/test.txt.The element type"Four" must be terminated by the matching end-tag"</Four>"

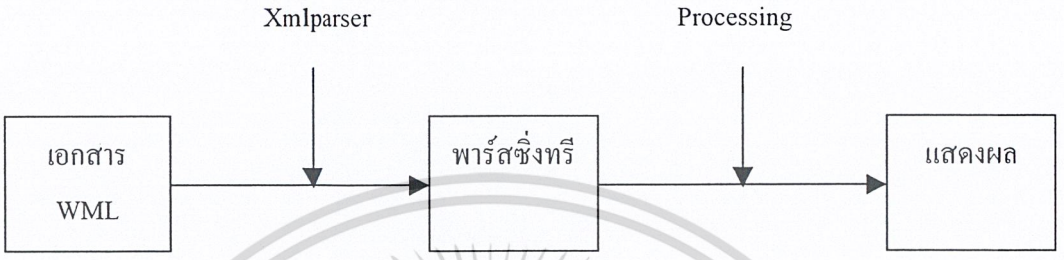


รูปที่ 4-2 ทำการพาร์สเซอร์โดยใช้โปรแกรม เมื่อมี error ในการพาร์สเซอร์

บทที่ 5

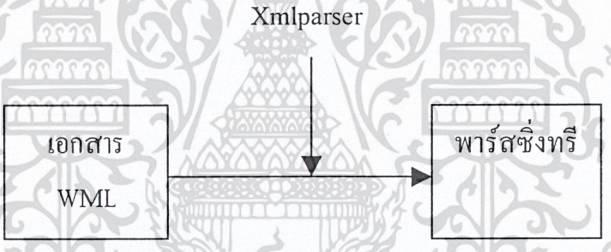
วิธีการเขียนโปรแกรมจัดการส่วนแสดงผล

หลังจากที่เราได้นำเอกสาร wml มาทำการผ่าน xmlparser เรียบร้อยแล้ว เราก็จะนำ ทรีที่ได้จากการพาร์สซิ่งมาทำส่วนแสดงผลโดยจะแบ่งเป็น 2 ขั้นตอนดังรูป



รูปที่ 5-1 ไดอะแกรมแสดงวิธีการจัดการส่วนแสดงผล

5.1 ส่วนของ Xmlparser



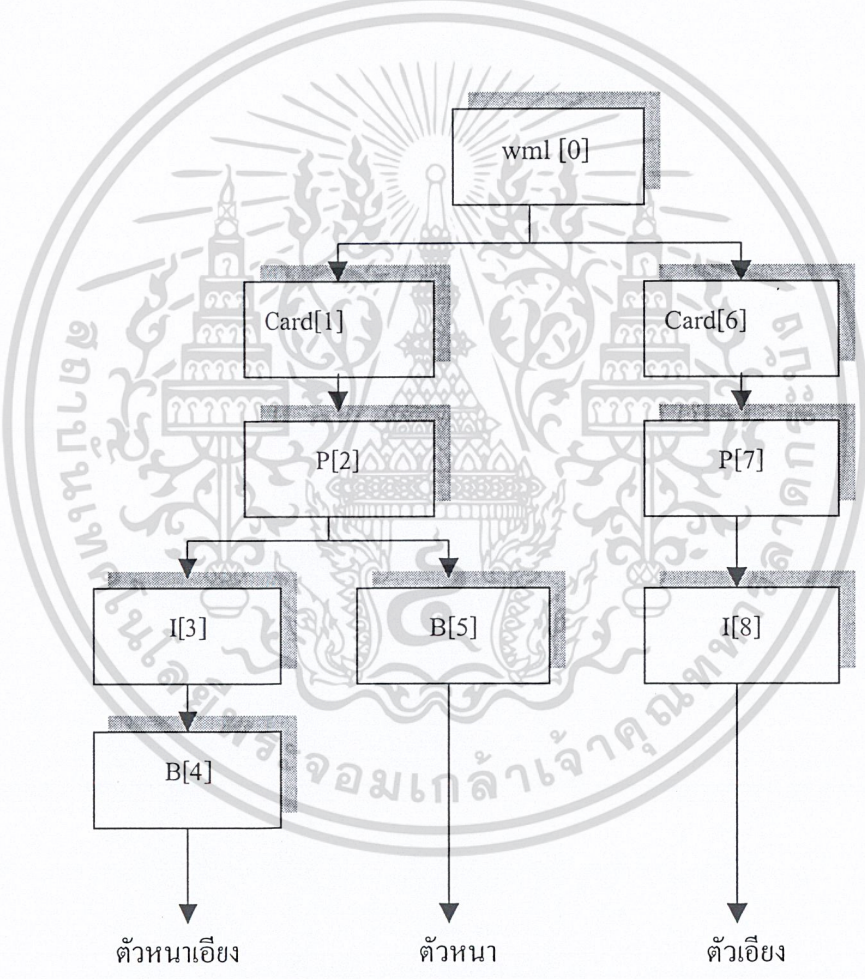
รูปที่ 5-2 ไดอะแกรมแสดงส่วน Xmlparser

ในส่วนนี้จะใช้ส่วนของ Xmlparser เข้ามาช่วยในการพาร์สซิ่งเอกสาร WML ให้ได้เป็นพาร์สซิ่งทรีตามที่ต้องการ เช่นถ้าเรามีเอกสาร test.wml ดังตัวอย่าง

```
<?xml version="1.0" encoding="MS874"?>
<wml>
<card id="card1" title="This is card1">
<p align="center">
        <i><b>ตัวหนาเอียง</b></i>
        <b>ตัวหนา</b>
</p>
</card>
<card id="card2" title="This is card2">
<p align="center">
```

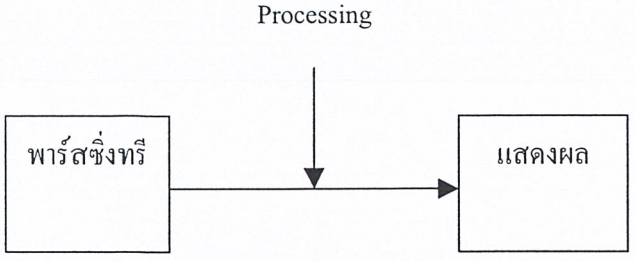
```
        <i>ไม่แสดงผล</i>
</p>
</card>
</wml>
```

เมื่อผ่านส่วนของ Xmlparser แล้วเราก็จะได้ พาร์สซิ่งทรีดังรูป



รูปที่ 5.3 ทรีที่ได้จากการพาร์สซิ่ง

5.2 ส่วนของ Processing



รูปที่ 5-4 ไดอะแกรมแสดงส่วน Processing

ในส่วนของการ Processing นั้นแบ่งเป็นหลายขั้นตอนได้แก่

5.2.1 method Par() ใช้ในการเก็บรายละเอียดของแต่ละ element จะเป็นการท่องไปในทรีแล้วทำการเก็บรายละเอียดของ Element ตัวเองไว้ โดยมี คลาส ElementInfo เป็นคลาสที่คอยเก็บรายละเอียดของแต่ละ Element ไว้ซึ่งมีรายละเอียดดังนี้

- รายละเอียดของ class ElementInfo

คลาส ElementInfo เป็นคลาสที่เก็บรายละเอียดของแต่ละ Element โดยมีฟิลด์(field) ที่สำคัญแบ่งเป็น 3 ชนิด ดังนี้คือ

1.ฟิลด์ทั่วไป ฟิลด์พวกนี้จะเก็บข้อมูลทั่วไปของ Element เช่น

ชื่อฟิลด์	หน้าที่
String Name	เก็บชื่อของ Element แต่ละ Element
Int Numattr	เก็บจำนวน แอตทริบิวต์ของแต่ละ Element
String[] attrbname	เก็บชื่อของแอตทริบิวต์ของแต่ละ Element
String[] attrbvalue	เก็บค่าของแอตทริบิวต์ของแต่ละ Element
String Content	เก็บคอนเทนต์ ของแต่ละ Element
String parent	เก็บชื่อของ Element ที่อยู่เหนือ Element ตนเองขึ้นไป 1 ระดับ
String position	เก็บตำแหน่งของ Element บนบราวเซอร์
Boolean canshow	เก็บค่าว่า Element นี้สามารถแสดงผลได้หรือไม่

ตารางที่ 5-1 ฟิลด์ทั่วไปของคลาส ElementInfo

2.ฟิลด์ที่เกี่ยวกับสไตล์ของฟอนต์

ชื่อฟิลด์	หน้าที่
boolean I	เก็บค่าของสไตล์ฟอนต์ของ Element นั้นว่าเป็นตัวเอียงหรือไม่
boolean b	เก็บค่าของสไตล์ฟอนต์ของ Element นั้นว่าเป็นตัวหนาหรือไม่
boolean u	เก็บค่าของสไตล์ฟอนต์ของ Element นั้นว่าเป็นตัวขีดเส้นใต้หรือไม่
boolean big	เก็บค่าของสไตล์ฟอนต์ของ Element นั้นว่าเป็นตัวใหญ่หรือไม่
boolean small	เก็บค่าของสไตล์ฟอนต์ของ Element นั้นว่าเป็นตัวเล็กหรือไม่

ตารางที่ 5-2 ฟิลด์ที่เกี่ยวกับสไตล์ของฟอนต์ของคลาส ElementInfo

3.ฟิลด์ที่เกี่ยวกับค่าโคออดิเนตของคอนเทนต์ ฟิลด์นี้จะเก็บค่าโคออดิเนต(x,y) ของคอนเทนต์ ว่าจะใส่ลงตำแหน่งไหนของบราวเซอร์

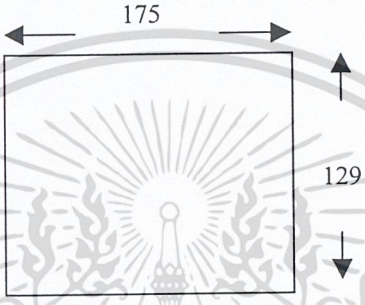
ชื่อฟิลด์	หน้าที่
int co_x	เก็บค่าพิกัดในแนวแกน x ของแต่ละ Element
int co_y	เก็บค่าพิกัดในแนวแกน y ของแต่ละ Element
int width	เก็บขนาดความกว้างของแต่ละ Element
int height	เก็บขนาดความสูงของแต่ละ Element

ตารางที่ 5-3 ฟิลด์ที่เกี่ยวกับค่าโคออดิเนตของคอนเทนต์ของคลาส ElementInfo

5.2.2 method setStyle() ใช้ในการเก็บรายละเอียดว่า Element ตนเองนั้นอยู่ภายใต้ Element อะไร ดังเช่นจาก test.wml นั้นเราจะทำการ setSytle() เราก็จะทำการท่องไปในทรีเพื่อทำการเก็บข้อมูลของแต่ละ Element โดยในเอกสาร wml นั้นจะสามารถแสดงผลได้ทีละคาร์ดเท่านั้นเมื่อท่องไปเจอ card [1] ก็จะเข้าไป set ฟิวล์ canshow ใน ElementInfo ให้มีค่าเป็นจริง แล้วจะทำการก๊อบปี้ไปให้แก่ Element ลูกทั้งหมดทำให้ TAG ทุก TAG ที่อยู่ภายใต้ Element นี้สามารถแสดงผลได้ส่วนอื่นก็จะไม่สามารถแสดงผลได้ เมื่อเจอ TAG P[2] ซึ่งเป็น TAG ที่เอาไว้จัดพารากราฟ ก็จะนำค่าจากแอตทริบิวต์ align มาจัดพารากราฟ ดังตัวอย่างข้างต้นก็จะ ก๊อปปี้ค่า align = “center” นี้ไปให้แก่ Element ลูกทั้งหมดของ p[2] นี้จะต้องอยู่กลางหน้า บราวเซอร์(Browser) ทั้งหมด เมื่อเจอ TAG I[3] ก็จะ set ฟิวล์ I ใน ElementInfo ให้มีค่า

เป็นจริงแล้วทำการ ก๊อบปี้ให้แก่ Element ลูกทั้งหมดทำให้ TAG ทุก TAG ที่อยู่ภายใต้ Element นี้จะต้องเป็นตัวเอียงทั้งหมด ส่วน Element card[6] นี้ก็จะไม่ถูกแสดงผลเนื่องจาก มี ฟิวล์ canshow ใน ElementInfo เป็น false ทำให้ Element ทุก Element ที่อยู่ภายใต้ card[6] นี้ไม่สามารถแสดงผลได้ เป็นต้น

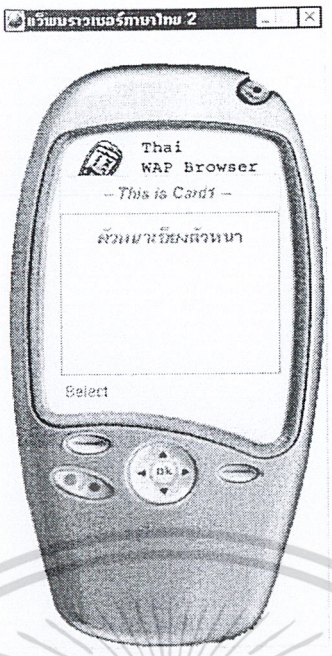
5.2.3 method setCo() เมื่อเราทราบแล้วรายละเอียดของแต่ละ Element แล้วเราก็จะทำการคำนวณพิกัดของแต่ละ Element ว่าอยู่ที่ตำแหน่งไหนของหน้าจอ ในโครงงานนี้ได้ออกแบบตัวหน้าจอไว้ขนาด 175*129 ดังรูป



รูปที่ 5-5 ขนาดของหน้าจอที่ใช้ในโครงงาน

ดังใน test.wml เราจะทำการคำนวณโดยท่องไปในทรีเมื่อเจอ Element B[4] ก็จะทำการเก็บค่าความยาวของ คอนเทนต์(content) นั้นไว้และจะบวกค่าความยาวไปเรื่อยๆ จนกว่าจะเจอ TAG BR หรือ TAG P เพื่อขึ้นบรรทัดใหม่ทำให้เราทราบความยาวของแต่ละบรรทัดได้ แล้วเราก็จะมาดูค่า position ว่า Element นั้นควรอยู่ที่ตำแหน่งไหนของหน้าจอ เช่นจากในตัวอย่าง ต้องอยู่ที่ตำแหน่งกลางหน้าจอเราก็จะใช้การคำนวณคือ Element แรกในบรรทัดนี้(B[4]) จะอยู่ที่ตำแหน่ง (175-ความยาวของคอนเทนต์ทั้งหมดในบรรทัดนี้)/2 ส่วน Element ถัดมาตำแหน่งก็จะเลื่อนไปเรื่อยๆ โดยการบวกค่าความยาวของคอนเทนต์ที่อยู่ก่อนหน้า Element ตนเองไปเรื่อยๆ ถ้าเกิดเจอ TAG BR หรือ TAG P ก็จะทำการคำนวณค่าพิกัดตรงนี้ใหม่ทำอย่างนี้ไปเรื่อยๆจนจบเอกสาร ก็จะได้ตำแหน่งของแต่ละ Element

5.2.4 method show1() เมื่อเราได้ตำแหน่งของแต่ละ Element เราก็จะทำการ add แต่ละ Element ลงไปในหน้าจอซึ่งในโครงงานนี้เราใช้ คอมโพเน็นต์(component) jEditorPane ในการ add แต่ละ คอมโพเน็นต์ลงไป เมื่อเสร็จแล้ว เราก็จะได้ผลลัพธ์ดังรูป



รูปที่ 5-6 ผลลัพธ์ที่ได้จาก test.wml

5.3 สรุป TAG ที่สามารถทำได้ในโครงงานนี้

ชื่อ TAG	*ผลที่ได้*
P	ได้ผลลัพธ์ถูกต้อง
B	ได้ผลลัพธ์ถูกต้อง
I	ได้ผลลัพธ์ถูกต้อง
EM	ได้ผลลัพธ์ถูกต้อง
BIG	ได้ผลลัพธ์ถูกต้อง
SMALL	ได้ผลลัพธ์ถูกต้อง
BR	ได้ผลลัพธ์ถูกต้อง
GO	ได้ผลลัพธ์ถูกต้อง
ANCHOR	ได้ผลลัพธ์ถูกต้อง
A	ได้ผลลัพธ์ถูกต้อง
IMG	ได้ผลลัพธ์ถูกต้องแต่ต้องเป็น file สกุล gif และ bmp เท่านั้น
TABLE	ได้ผลลัพธ์ถูกต้องแต่ไม่สามารถแสดงตารางที่มีความซับซ้อนได้
DO	ได้ผลลัพธ์ถูกต้องแต่ใช้ได้เฉพาะ task GO กับ PREV
ตัวอักขระพิเศษ	ได้ผลลัพธ์ถูกต้อง

ตารางที่ 5-4 สรุป TAG ที่สามารถทำได้ในโครงงานนี้

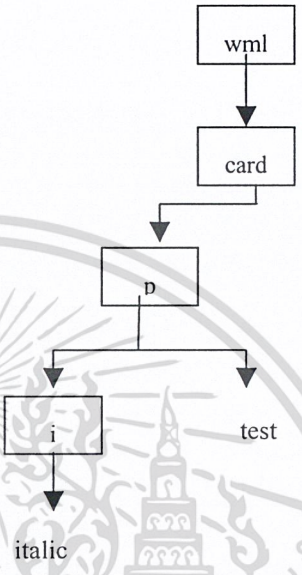
ส่วน TAG ที่ยังไม่สามารถทำได้ในโครงงานนี้ก็จะเป็น TAG ที่เกี่ยวกะ Event และ Form

5.4 Problem Solving

5.4.1. ความผิดพลาดที่เกิดขึ้นกับ พาร์สทรี

พิจารณา ซอสโค้ด :

```
<wml>
        <card id = “card 1”>
                <p>
                        test
                        <i> italic</i>
                </p>
        </card>
</wml>
```



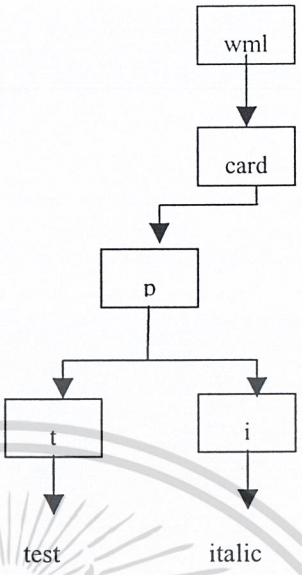
รูปที่ 5-7 พาร์สทรีของการพาร์สที่ไม่เห็นความแตกต่างระหว่างอีลิเมนต์ 2 ตัว

จาก ซอสโค้ด ด้านซ้าย เมื่อผ่านการพาร์ส โดย xmlparser จะสามารถแปลงได้เป็นดังทรี ด้านขวา ถึงแม้ว่าจะเปลี่ยน ซอสโค้ด เป็นดังนี้

```
<wml>
        <card id = “card 1”>
                <p>
                        <i> italic</i>
                        test
                </p>
        </card>
</wml>
```

ก็ยังคงให้ผล ดังทรี เดียวกัน แต่เมื่อเปลี่ยน ซอสโค้ด โดยการเพิ่มแท็ก เข้าไป ดังนี้

```
<wml>
        <card id = “card 1”>
                <p>
                        <t>test</t>
                        <i> italic</i>
                </p>
        </card>
</wml>
```

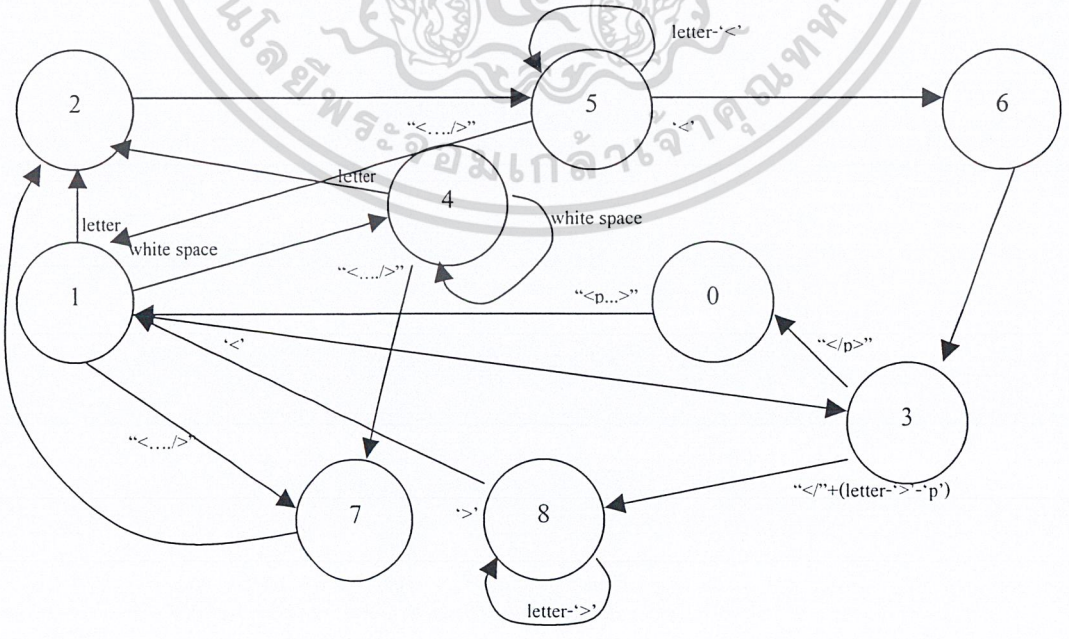


จะพบว่าการแสดงผลทำได้ถูกต้อง

รูปที่5-8 พาร์สทรีของการพาร์สที่เพิ่มเติมแท็ก

เมื่อพบข้อความ โล่ง ๆ ที่เป็นคอนเทนต์ ในอีลิเมนต์ <p> โดยไม่มีแท็กอื่นคร่อมอยู่ (ไม่เป็น content ของ element ใด ๆ) จะต้องมีการเพิ่มแท็กมาคร่อมคอนเทนต์ ที่เป็นข้อความนั้น เพื่อให้การพาร์สโดย xmlparser เป็นไปอย่างถูกต้อง โดยระบุใน เมธอด ชื่อ String AddTTag(String src)

เริ่มออกแบบด้วยการสร้างแผนภาพสถานะ (State Diagram)



รูปที่ 5-9 แผนภาพสถานะของการแทรกแท็ก เพื่อทำให้การแสดงผลถูกต้อง

รายละเอียดสถานะ (State)

สถานะที่ 1 : เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุท

สถานะที่ 4 : เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุท

สถานะที่ 2 : เลื่อนตัวชี้ถอยหลัง 1 ตัว

สถานะที่ 5 : แทรก “<t>” เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุท

สถานะที่ 6 : เลื่อนตัวชี้ถอยหลัง 1 ตัว แทรก “</t>”เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุท

สถานะที่ 3 : เลื่อนตัวชี้เดินหน้า 3 ตัว ดู อินพุท

สถานะที่ 0 : เลื่อนตัวชี้ถอยหลัง 3 ตัว ดู อินพุท

สถานะที่ 8 : เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุท

สถานะที่ 7 : เลื่อนตัวชี้เดินหน้า 1 ตัว ดู อินพุทเมื่อพบ “/>” แล้วเลื่อนตัวชี้ถอยหลังกลับไปยังจุดเดิมเมื่อตอนเริ่มเข้ามายังสถานะที่ 7 นี้

รายละเอียดเส้นทาง

จากสถานะที่ 1 ไปยังสถานะที่ 7 เมื่อพบ ‘<’ตามด้วยตัวอักษรใด ๆ จนกระทั่งพบ “/>”

จากสถานะที่ 1 ไปยังสถานะที่ 2 เมื่อพบ ตัวอักษรใด ๆ

จากสถานะที่ 1 ไปยังสถานะที่ 4 เมื่อพบไวท์สเปซ

จากสถานะที่ 4 ไปยังสถานะที่ 7 เมื่อพบ ‘<’ตามด้วยตัวอักษรใด ๆ จนกระทั่งพบ “/>”

จากสถานะที่ 4 ไปสถานะที่ 4 เมื่อพบ ไวท์สเปซ จนกระทั่งพบตัวอักษรใด ๆ

จากสถานะที่ 4 ไปยังสถานะที่ 2 เมื่อพบ ตัวอักษรใด ๆ

จากสถานะที่ 5 ไปยังสถานะที่ 1 เมื่อพบ ‘<’ตามด้วยตัวอักษรใด ๆ จนกระทั่งพบ “/>”

จากสถานะที่ 5 ไปยังสถานะที่ 5 เมื่อพบ ตัวอักษร ยกเว้น ‘<’

จากสถานะที่ 5 ไปยังสถานะที่ 6 เมื่อพบ ‘<’

จากสถานะที่ 3 ไปยังสถานะที่ 0 เมื่อพบ “</p>”

จากสถานะที่ 0 ไปยังสถานะที่ 1 เมื่อพบ “<p” ต่อด้วยตัวอักษรใด ๆ จนกระทั่งพบ ‘>’

จากสถานะที่ 1 ไปยังสถานะที่ 3 เมื่อพบ ‘<’

จากสถานะที่ 3 ไปยังสถานะที่ 8 เมื่อพบ “</” ต่อด้วยอักษรใด ๆ ยกเว้น ‘p’ และยกเว้น ‘>’

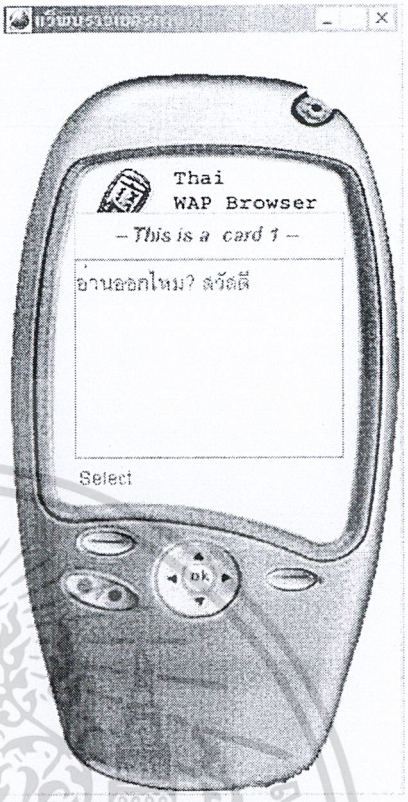
จากสถานะที่ 8 ไปยังสถานะที่ 8 เมื่อพบ ตัวอักษรใด ๆ ยกเว้น ‘>’

จากสถานะที่ 8 ไปยังสถานะที่ 1 เมื่อพบ ‘>’

ตัวอย่าง การแปลงซอร์สโค้ดเพื่อให้ผลการพาร์สโดย xmlparser เป็นไปอย่างถูกต้อง

จากโค้ดที่ยังไม่ผ่านการแปลง ซึ่งทำให้การแสดงผลผิดพลาด

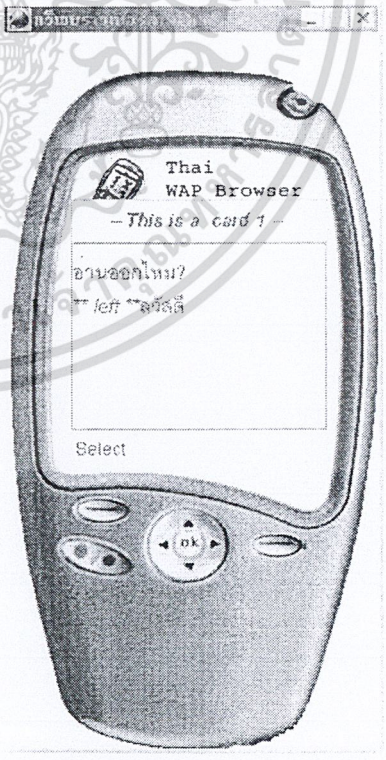
```
<?xml version="1.0" encoding="MS874"?>
<wml>
<card id="card1" title="This is a  card 1">
        <p align="left">
                อ่านออกไหม?<br/>
                <i>** left **</i>
                สวัสดี
        </p>
</card>
</wml>
```



รูปที่ 5-10 การแสดงผลเมื่อเกิดความผิดพลาดจากการพาร์สโดย xmlparser

เมื่อผ่านการแปลงแล้ว จะได้ผลลัพธ์ดังนี้

```
<?xml version="1.0" encoding="MS874"?>

<wml>

<card id="card1" title="This is a  card 1">
        <p align="left">
                <t>อ่านออกไหม?</t><t><br/></t>
                <i>** left **</i>
                <t>สวัสดี</t>
    </p>
</card>
</wml>
```



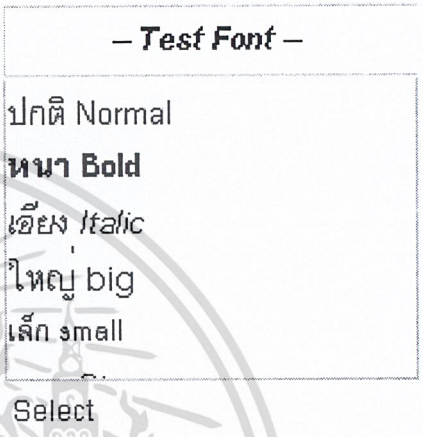
รูปที่ 5-11 การแสดงผลเมื่อแปลงซอร์สโค้ดเพื่อให้การพาร์สโดย xmlparser ให้ผลลัพธ์ถูกต้อง

5.5 สิ่งที่พัฒนาจากแว็พบราวเซอร์ภาษาไทยรุ่นก่อน

- รูปแบบตัวอักษร

แว็พบราวเซอร์ภาษาไทย

แว็พบราวเซอร์ภาษาไทย 2



รูปที่ 5-12 การเปรียบเทียบการกำหนดรูปแบบอักษร

- ความสะดวกในการท่องแว็พ

  แว็พบราวเซอร์ภาษาไทย 2 ได้จัดทำให้มีหน้าจอการใช้งานที่อำนวยความสะดวกให้ผู้ใช้

- รองรับแท็กภาษา WML ได้มากขึ้น
- เขียนส่วนแสดงผลเองทั้งหมดทำให้ไม่ต้องยึดติดกับโครงสร้างภาษา HTML

สาเหตุที่แว็พบราวเซอร์ภาษาไทยรุ่นก่อนนี้ ไม่สามารถแสดงผลภาษาไทยออกมาได้อย่างถูกต้องสมบูรณ์นั้นก็เนื่องมาจาก การใช้คุณสมบัติ Web Browser แสดงผลเอกสาร HTML ของ JEditroPane มาแสดงผล ซึ่งก่อนที่จะนำ เอกสาร WML มาแสดงผล บน JEditorPane นั้นต้องแปลงเอกสาร WML ให้เป็นเอกสาร HTML เสียก่อน ซึ่ง ลักษณะบางอย่างของเอกสาร WML ก็จะสูญเสียไป ไม่สามารถนำมาแสดงผลได้อย่างครบถ้วน

บทที่ 6

ผลการทดสอบ

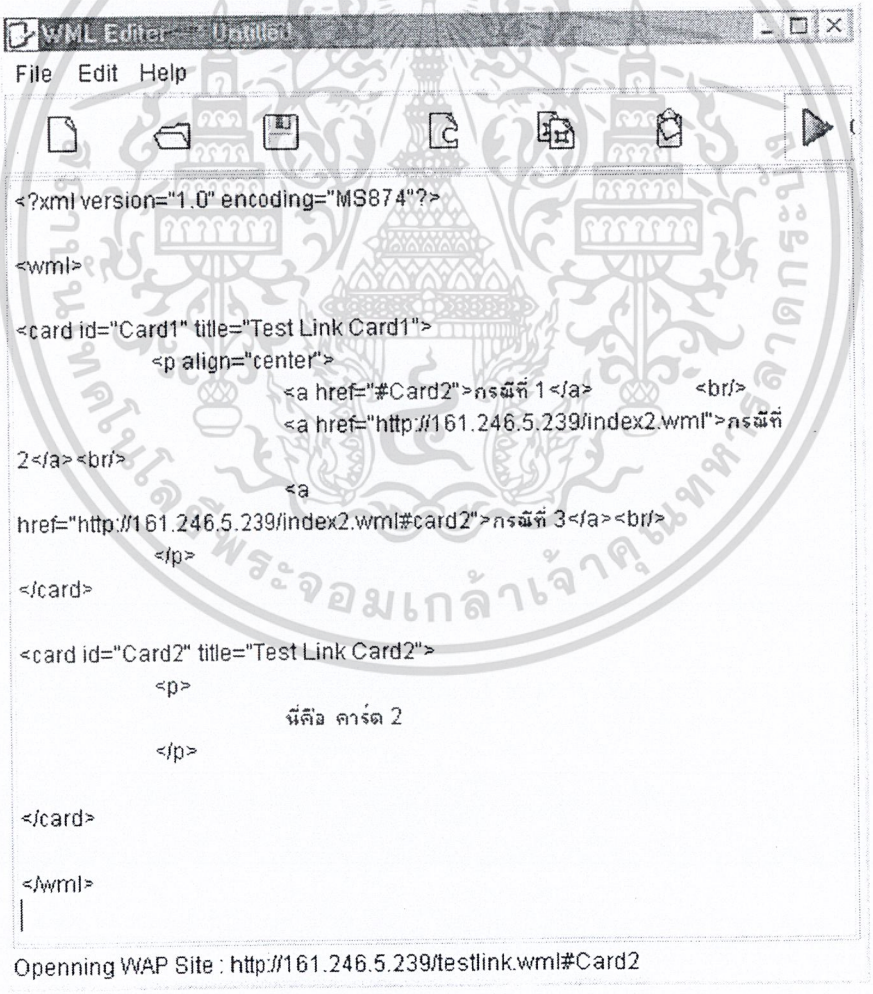
ในโครงงานแว็พบราวเซอร์ภาษาไทยนี้มีวัตถุประสงค์เพื่อต้องการให้แว็พบราวเซอร์นั้นสามารถแสดงผลเป็นภาษาไทยได้โดยได้รับมาจากผู้ใช้ 2 ทางคือจาก URL ที่ผู้ใช้ต้องการจะเข้าไปชมแว็พไซต์ และจากการเขียนโปรแกรมของผู้ใช้ดังนั้นผลการทดสอบจึงได้แบ่งเป็น 2 ส่วนหลักคือ

6.1 การใช้ Browser ติดต่อกับแว็พไซต์

ในการทดสอบจะใส่ URL ที่ต้องการซึ่งผลการทดสอบจะถูกแสดงออกมาเป็น 2 ส่วนคือส่วนที่แสดงผลในส่วนแสดงผลไฟล์เอกสาร WML และส่วนโทรศัพท์มือถือจำลองซึ่งผลการทดลองเป็นดังนี้

6.1.1 ส่วนแสดงเอกสาร WML

ในการทดสอบนั้นได้ผลการทดลองคือแว็พบราวเซอร์ภาษาไทยสามารถแสดงผลเป็นภาษาไทยได้ในส่วนที่ใช้แสดงผลไฟล์เอกสาร WML ดังแสดงในรูปที่ 6-1



รูปที่ 6-1 การแสดงผลไฟล์เอกสาร WML ที่ได้จาก URL

6.1.2 การแสดงผลบน Browser

จากการทดสอบนั้นผลที่ได้คือแว็พบราวเซอร์ภาษาไทยสามารถแสดงผลเป็นภาษาไทยได้บนหน้าจอของโทรศัพท์มือถือจำลองซึ่งได้แสดงไว้ในรูปที่ 6-2 และเนื่องจากแว็พเซิร์ฟเวอร์ปัจจุบันไม่มีการสนับสนุนข้อมูลที่เป็นภาษาไทยโครงงานนี้จึงได้สร้างเซิร์ฟเวอร์ขึ้นมาเอง

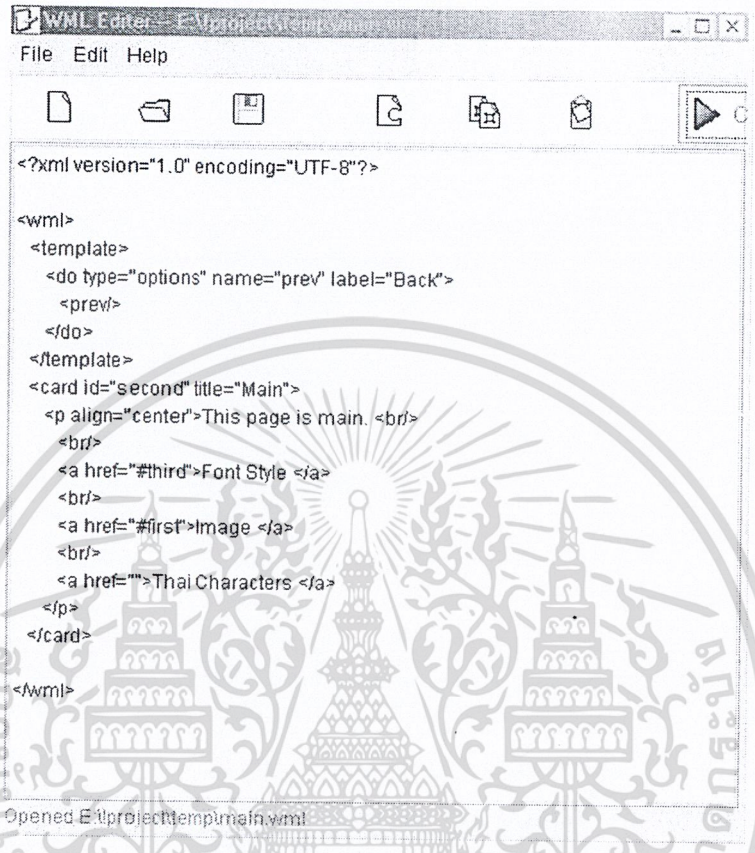
รูปที่ 6-2 ผลลัพธ์จากหน้าจอแสดงผลของไฟล์ WMLที่ได้จาก URL

6.2 การสร้าง เอกสาร WML โดยผู้ใช้

ในการเขียนโปรแกรมภาษา WML ด้วยโปรแกรมแว็พบราวเซอร์ภาษาไทย ผู้ใช้จะเขียนลงไปในส่วนแสดงผลไฟล์เอกสาร WML และทำการคอมไพล์ ผลลัพธ์จะถูกแสดงผลบนหน้าจอโทรศัพท์มือถือจำลองซึ่งผลการทดสอบเป็นดังนี้

6.2.1 ส่วนแสดงเอกสาร WML

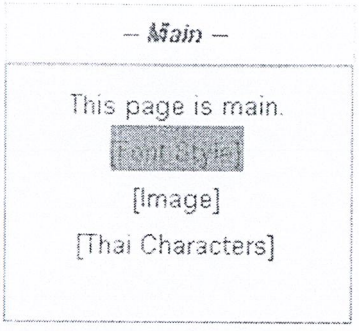
ในการทดสอบนี้ได้ทดลองโดยการพิมพ์แก้ไขลงไปซึ่งผลที่ได้นั้น คือสามารถแสดงผลส่วนที่แก้ไขได้ดังรูปที่ 6-3



รูปที่ 6-3 แสดงเอกสาร WML ที่ผู้ใช้เขียนขึ้นมา

6.2.2 การแสดงผลบน Browser

เมื่อผู้ใช้เขียนโปรแกรมเสร็จจะต้องบันทึกไฟล์ แล้วสั่งให้แสดงผลโดยการกดปุ่ม Display ซึ่งการทดสอบคือการแก้ไขเอกสาร WML ผลการทดสอบคือบนส่วน Browser สามารถแสดงผล อย่างถูกต้อง ได้ดังรูปที่ 6-4



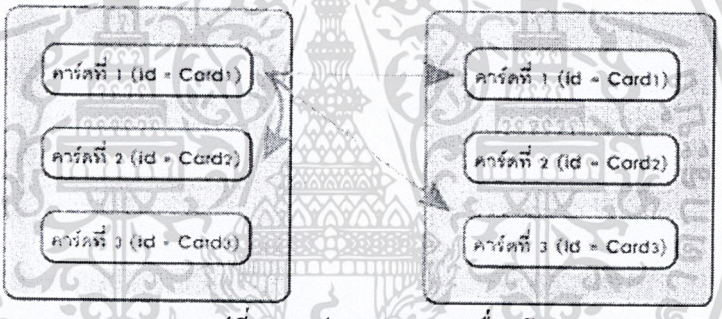
รูปที่ 6-4 ผลลัพธ์ที่ได้หลังจากการคอมไพล์โปรแกรมที่ผู้ใช้ทดลองเขียนขึ้นมา

6.3 รูปแบบการแสดงผลแบบต่าง ๆ

เมื่อลองกำหนดรูปแบบการแสดงผลแบบต่าง ๆ สำหรับการเขียนโค้ด ให้ผลดังนี้

6.3.1 การเชื่อมโยง

รูปแบบของการเชื่อมโยงยังสามารถแบ่งแยกย่อยออกไปได้อีกดังนี้

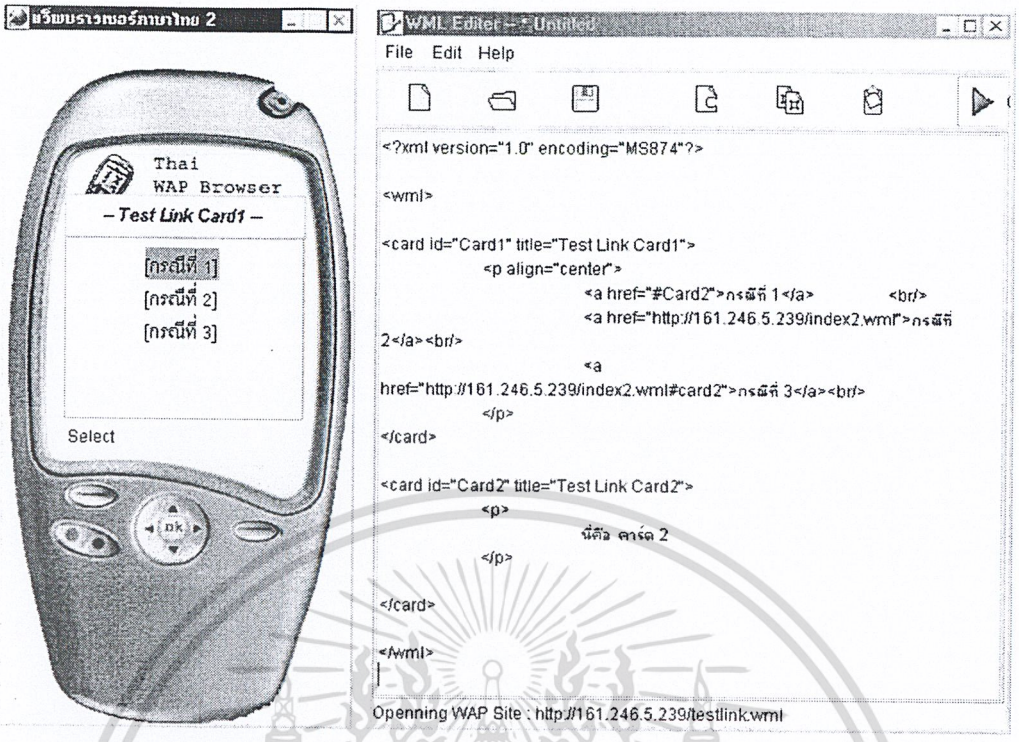


รูปที่ 6-5 รูปแบบของการเชื่อมโยง

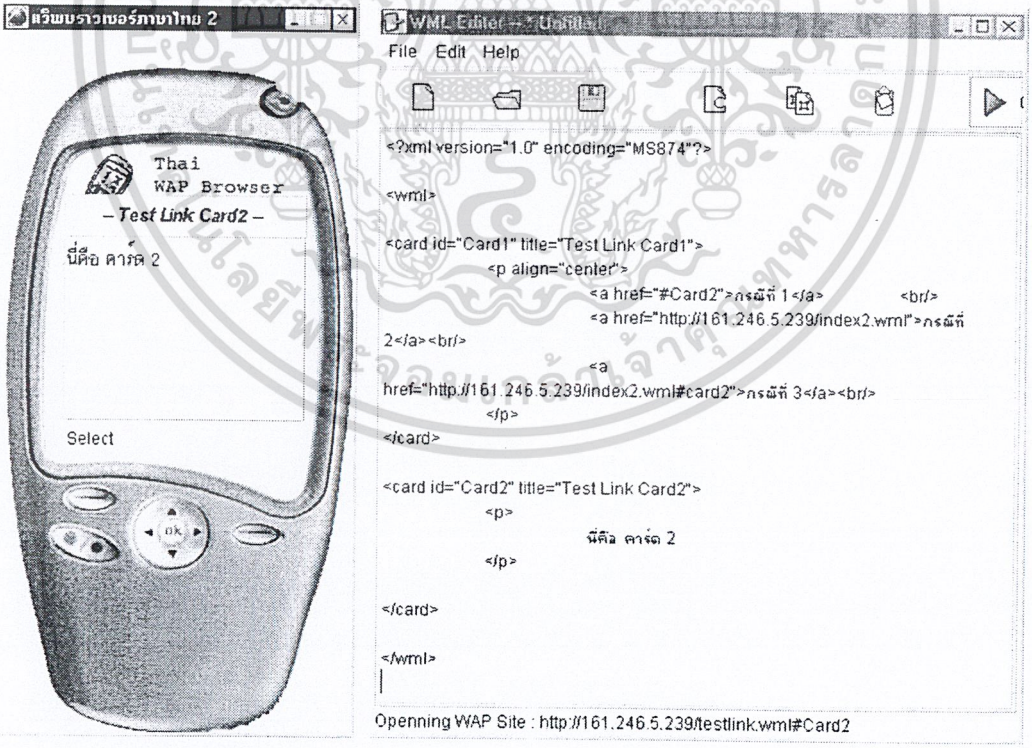
6.3.1.1 การเชื่อมโยงภายในไฟล์เดียวกัน ดังรูปที่ 6-7เป็นการเชื่อมโยงจาก คาร์ดที่ 1 ไปยัง คาร์ดที่ 2 ภายในไฟล์เดียวกัน

6.3.1.2 การเชื่อมโยงไปยังไฟล์อื่น ดังรูปที่ 6-8 เป็นการเชื่อมโยงจาก คาร์ดที่ 1 ของไฟล์หนึ่งไปยังคาร์ดที่ 1 ของอีกไฟล์หนึ่ง

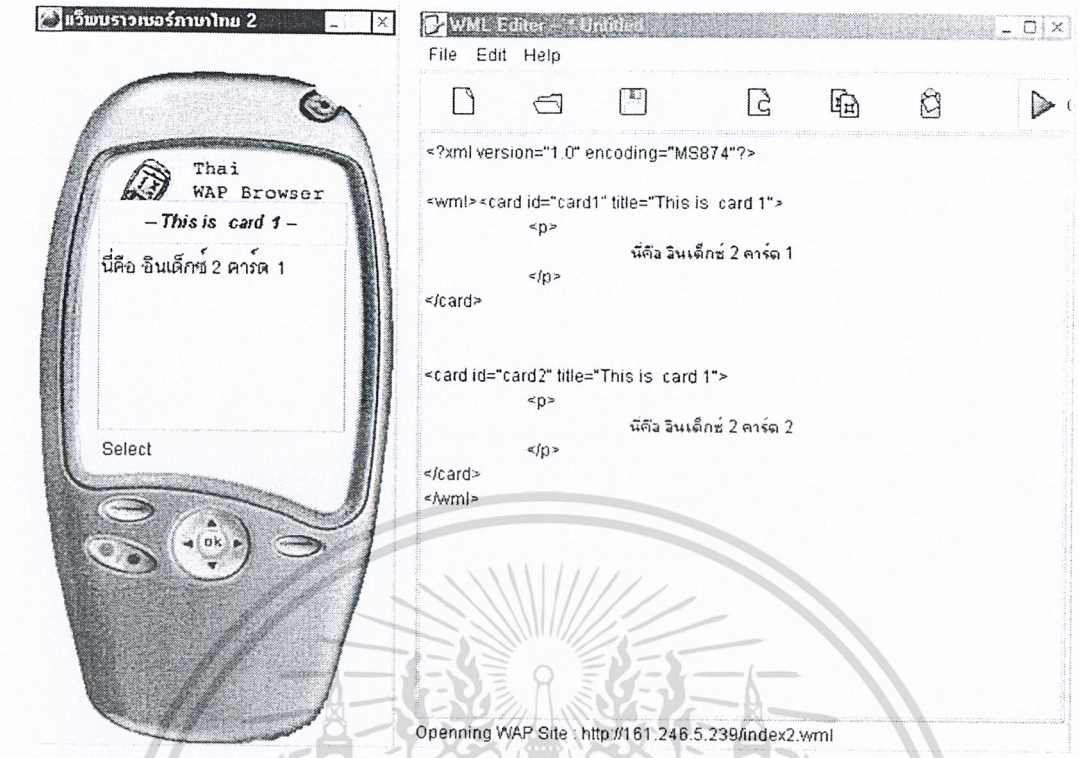
6.3.1.3 การเชื่อมโยงไปยังไฟล์อื่น พร้อมทั้งระบุคาร์ดได้ ดังรูปที่ 6-9 เป็นการเชื่อมโยงจาก คาร์ดที่ 1 ของไฟล์หนึ่ง ไปยังคาร์ดที่ 2 ของอีกไฟล์หนึ่ง



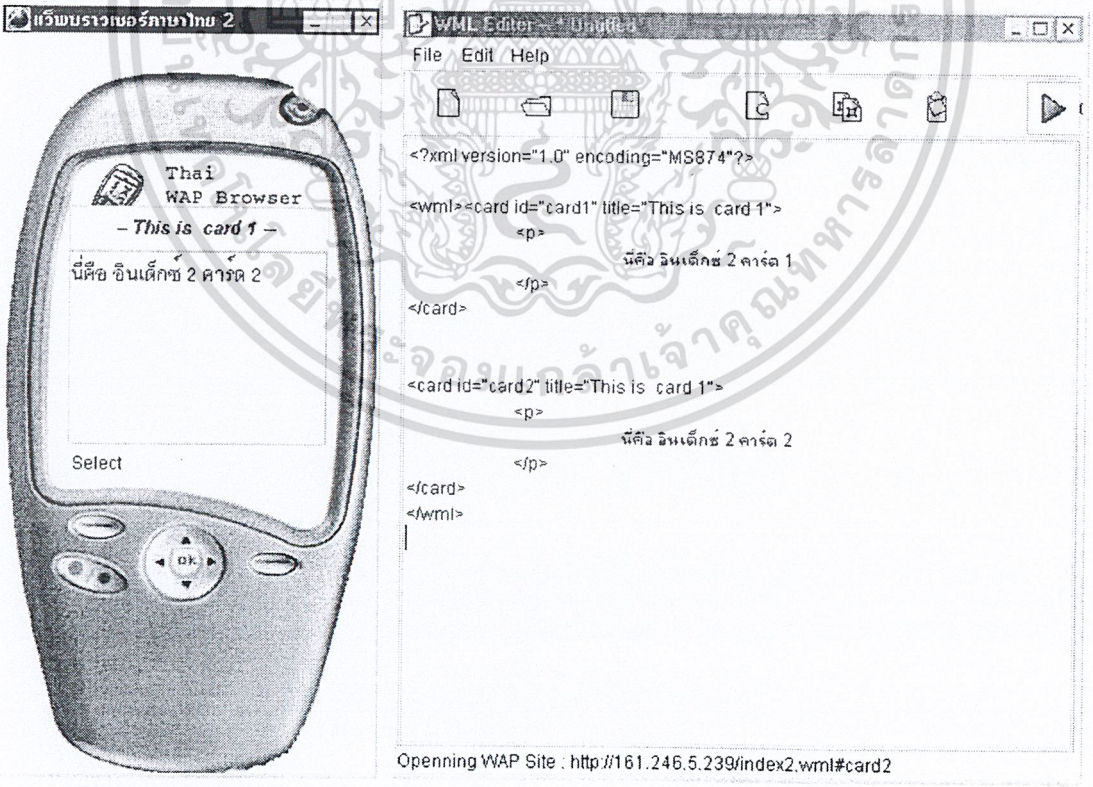
รูปที่ 6-6 การทดสอบการเชื่อมโยง



รูปที่ 6-7 การเชื่อมโยงภายในไฟล์เดียวกัน



รูปที่ 6-8 การเชื่อมโยงไปยังไฟล์อื่น

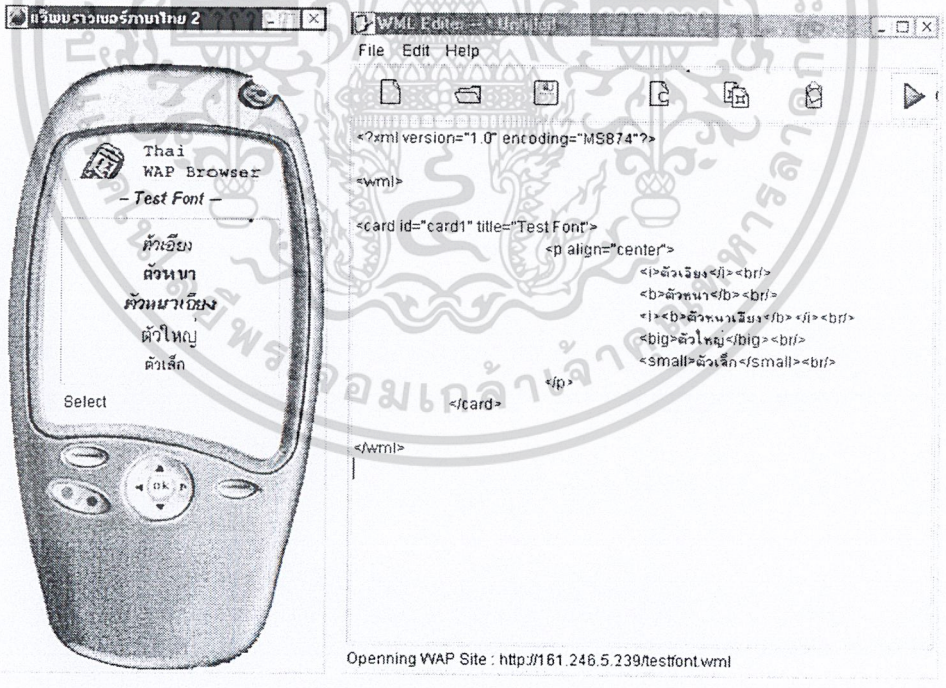


รูปที่ 6-9 การเชื่อมโยงไปยังไฟล์อื่น พร้อมทั้งระบุคาร์ดได้

6.3.2 รูปแบบตัวอักษรภาษาอังกฤษ

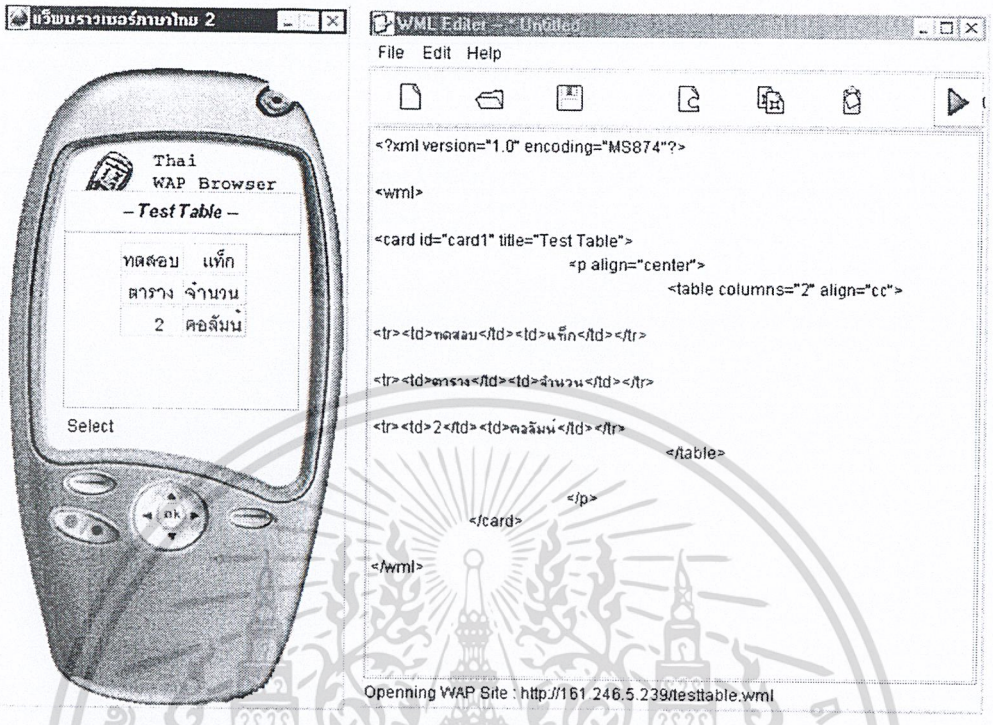
รูปที่ 6-10 การแสดงผลภาษาอังกฤษจากการกำหนดรูปแบบตัวอักษรแบบต่างๆ

6.3.3 รูปแบบตัวอักษรภาษาไทย



รูปที่ 6-11 การแสดงผลภาษาไทยจากการกำหนดรูปแบบตัวอักษรแบบต่างๆ

6.3.4 เอกสารรูปแบบตาราง



รูปที่ 6-12 การแสดงผลของเอกสารรูปแบบตาราง

6.4 ปัญหาที่เกิดขึ้นในการแสดงผลในแว็พบราวเซอร์ภาษาไทย

6.4.1. การแสดงผลของรูปภาพประเภท wbmp

เนื่องจาก ภาษา java รองรับการแสดงรูปภาพ แต่เฉพาะประเภท jpg gif และ bmp แต่ไม่รองรับกับการจัดการกับภาพประเภท wbmp ซึ่งเป็นประเภทของไฟล์รูปภาพที่ใช้กันโดยทั่วไปบนแว็พไซต์ต่าง ๆ

บทที่ 7
บทวิจารณ์และสรุป

โครงงานนี้จัดทำขึ้นเพื่อจำลองการทำงานของแว็พบราวเซอร์ และเพิ่มความสามารถในการจัดการกับตัวอักษรภาษาไทย เพื่ออำนวยความสะดวกให้กับนักท่องแว็พไซต์ รวมไปถึงนักพัฒนาแว็พแอ็พพลิเคชันทั้งหลาย ในการเข้าเยี่ยมชมแว็พไซต์ ทำความเข้าใจ และได้รับความรู้จากแว็พไซต์ต่าง ๆ เหล่านั้น นอกจากนี้ยังสามารถดูผลการเขียนโปรแกรมภาษา WML ได้ที่หน้าบราวเซอร์ที่แสดงผล

7.1 สรุปและวิเคราะห์การทดสอบ

โปรแกรมแว็พบราวเซอร์ภาษาไทยที่สร้างขึ้นจากโครงงานนี้ยังไม่สมบูรณ์ในหลายๆ ด้านเช่น การแสดงผลและการจำลองการทำงานของโทรศัพท์มือถือ โดยทำการวิเคราะห์ผลการทดสอบและเปรียบเทียบ กับแว็พบราวเซอร์อื่นๆทั่วไป สามารถแยกผลการทดสอบออกเป็นส่วนๆ ดังนี้

7.1.1 การทดสอบการเข้าชมแว็พไซต์ต่าง ๆ แสดงผลได้ถูกต้อง นอกเสียจากบางแว็พไซต์ที่มีโครงสร้างไม่เหมาะสมตามหลักการจัดรูปแบบซอสโค้ดภาษา WML

7.1.2 การทดสอบการเขียนโปรแกรมภาษา WML บันทึกเป็นไฟล์ แล้ว คอมไพล์เพื่อให้แสดงผลที่หน้าบราวเซอร์ สามารถแสดงผลได้ถูกต้องตามความประสงค์ของผู้เขียนโปรแกรม

7.1.3 การทดสอบการแสดงผลข้อความจากการเขียนโปรแกรมให้แสดงข้อความ สามารถแสดงข้อความได้ถูกต้องทั้งภาษาไทยและภาษาอังกฤษ รวมไปถึงจัดเรียงลำดับตัวอักษรภาษาไทยสามารถจัดเรียงได้ถูกต้องตามหลักของภาษาไทย และสำหรับการแสดงผลรูปแบบตัวอักษรให้เป็นตัวหนา ตัวเอียง ตัวเล็ก ตัวใหญ่ สามารถแสดงผลได้ถูกต้อง ยกเว้น ตัวอักษรที่มีการขีดเส้นใต้ไม่สามารถแสดงผลได้

7.1.4 การทดสอบการลิงก์ไปยังคาร์ดอื่น ๆ ในไฟล์เดียวกัน ไปยังแว็พไซต์อื่น ๆ หรือแม้กระทั่งระบุ คาร์ดของแว็พไซต์ปลายทางที่ต้องการเชื่อมโยงไปสามารถทำได้ถูกต้องตามหลักการของภาษา WML

7.1.5 การทดสอบการแสดงผลรูปภาพ สามารถแสดงผลไฟล์รูปภาพประเภท GIF JPG และ BMP ได้ถูกต้อง และเป็นภาพสี แต่ไม่สามารถแสดงรูปภาพที่เป็นไฟล์ประเภท WBMP ได้

7.1.6 การทดสอบการส่งข้อมูลไปประมวลผลที่เซิร์ฟเวอร์ ไม่สามารถทำได้

7.2 แท็กที่สามารถรองรับได้

แท็กที่แว็พบราวเซอร์ภาษาไทยสามารถรองรับได้นั้น ได้แก่ แท็กพื้นฐานของภาษา WML ทุกแท็กแต่จะมีข้อจำกัดอยู่บางแท็ก เช่น แท็ก img จะสามารถแสดงได้เฉพาะไฟล์ที่สกุล jpg และ gif เท่านั้น ไม่สามารถแสดงไฟล์ที่สกุล wbmp เหมือนที่แว็พไซต์ของจรองใช้ได้ แท็ก table ก็ไม่สามารถทำตารางที่ซับซ้อนได้ แท็ก do สามารถใช้ได้กับ task ชนิด Go และ Prev เท่านั้น

ส่วนแท็กที่ยังไม่ได้ทำได้แก่แท็กที่เกี่ยวกับ form และ event ต่างๆซึ่งสามารถนำไปพัฒนาต่อไปได้

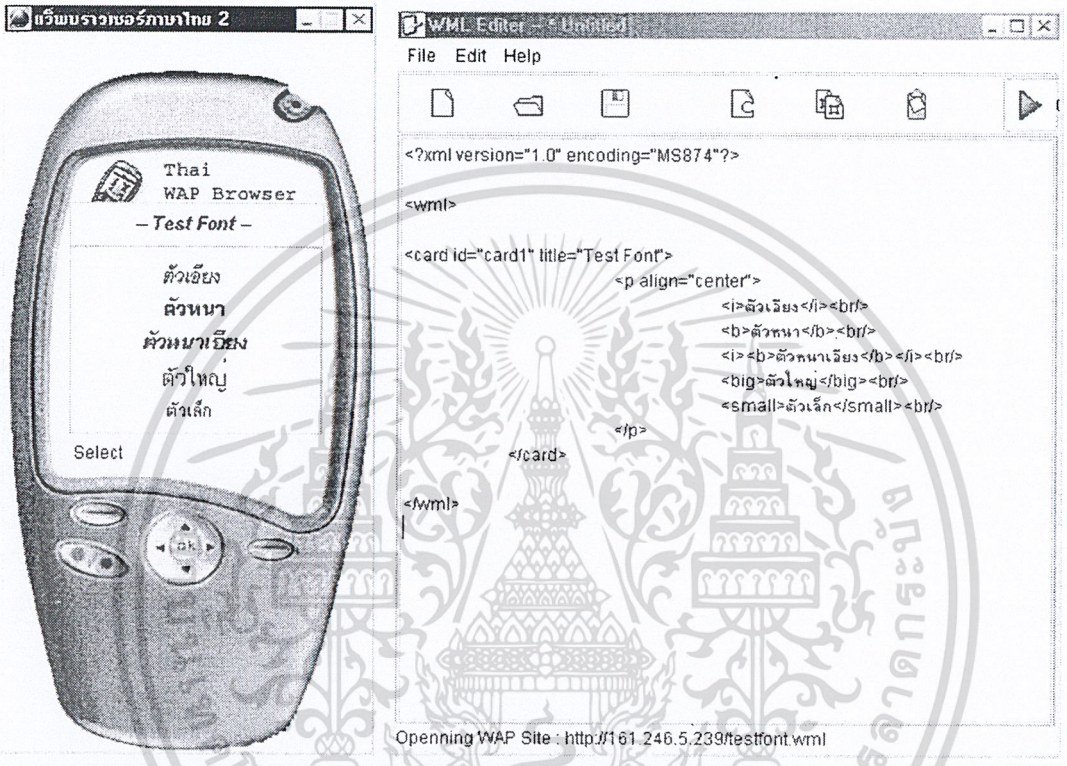
7.3 แนวทางการพัฒนาต่อ

7.3.1 ในส่วนของการพัฒนาให้แว็พบราวเซอร์ภาษาไทยสามารถสนับสนุนการรับการแสดงผลจากส่วน WML Script และการรับ CGIได้ทั้งสองแบบคือ GET และ POST ได้สมบูรณ์แบบมากขึ้นและสนับสนุนได้ทุกแท็กของภาษา WML

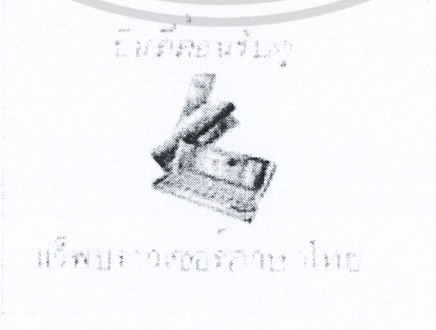
7.3.2 ในส่วนของการพัฒนาให้แว็พบราวเซอร์ภาษาไทยมีความฉลาดขึ้น (Intelligent) เช่น สามารถรับสคริปต์จากผู้ใช้เพื่อสามารถกำหนดการทำงานให้กับตัวบราวเซอร์โดยอัตโนมัติ

ภาคผนวก ก

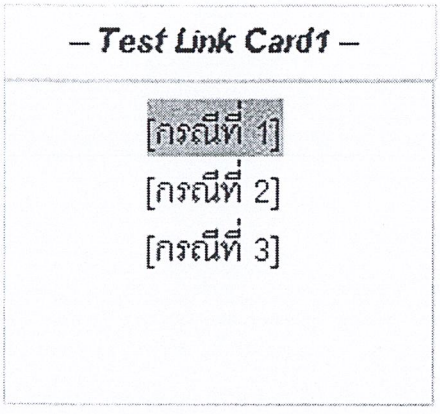
ผลลัพธ์ที่ได้จากการทดสอบโปรแกรมแว็พบราวเซอร์ภาษาไทย 2โดยเขียนโปรแกรมภาษา WML ขึ้นมาเพื่อทดสอบแท็กต่างๆ ที่สำคัญสำหรับการแสดงผลออกสู่หน้าจอของมือถือจำลอง



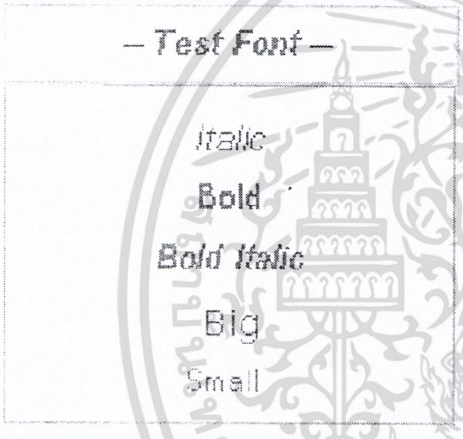
รูปที่ ก-1 ตัวอย่างโปรแกรมแว็พบราวเซอร์ภาษาไทย



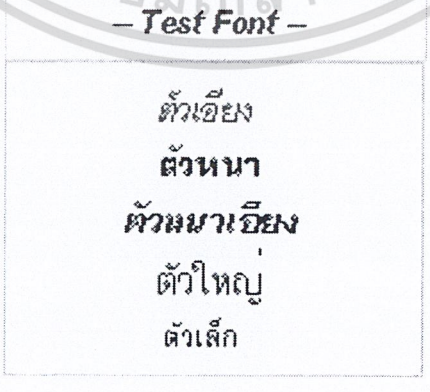
รูปที่ ก-2 รูปขยายหน้าจอการแสดงผลข้อความ



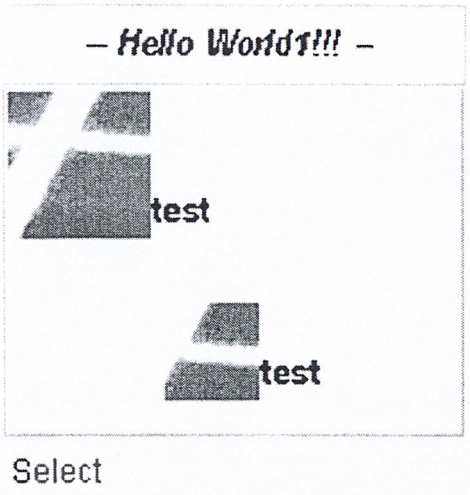
รูปที่ ก-3 รูปขยายหน้าจอการแสดงผลลิงค์



รูปที่ ก-4 รูปขยายการแสดงผลการกำหนดรูปแบบตัวอักษรภาษาอังกฤษ



รูปที่ ก-5 รูปขยายการแสดงผลการกำหนดรูปแบบตัวอักษรภาษาไทย



รูปที่ ก-6 รูปขยายการหน้าจอการแสดงผลรูปภาพ

รูปที่ ก-7 รูปขยายการหน้าจอการแสดงผลตาราง

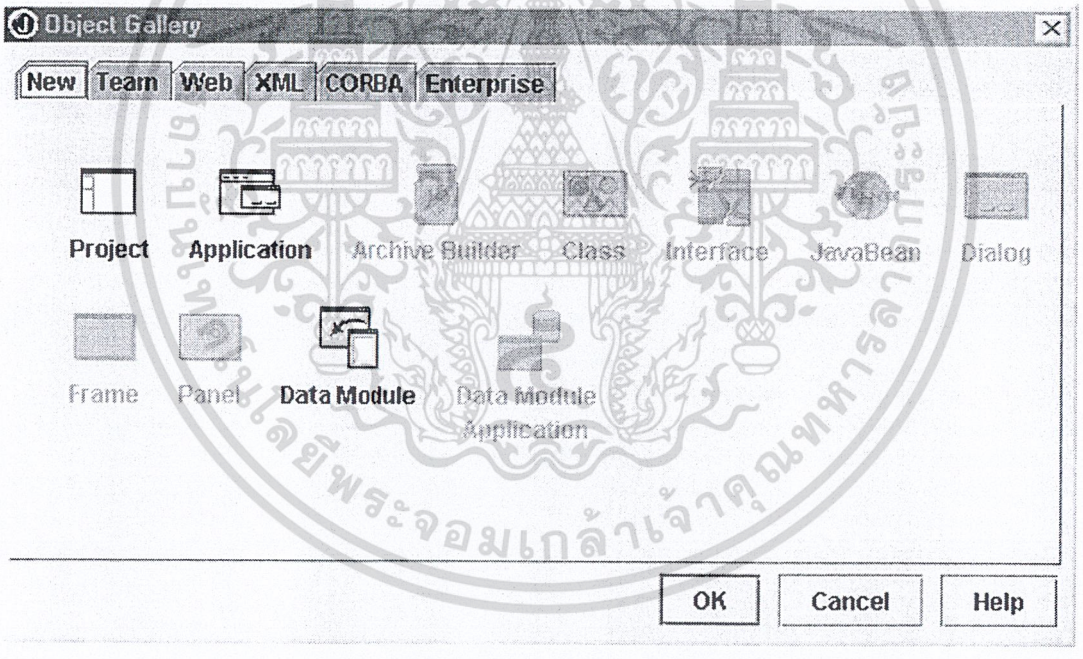
ภาคผนวก ข

การใช้งานเครื่องมือ (Tool) ในการพัฒนาโปรแกรมแว็พบราวเซอร์ภาษาไทย 2

เครื่องมือที่ใช้ คือ JBuilder 5 เป็นเครื่องมือที่ช่วยอำนวยความสะดวกในการเขียนโปรแกรมภาษาจาวา ก่อนที่จะใช้งานเครื่องมือตัวนี้ต้องศึกษาวิธีการใช้งานให้เข้าใจเสียก่อน ซึ่งการใช้งานมีขั้นตอนดังนี้

1. **สร้างโครงงาน (Project)**

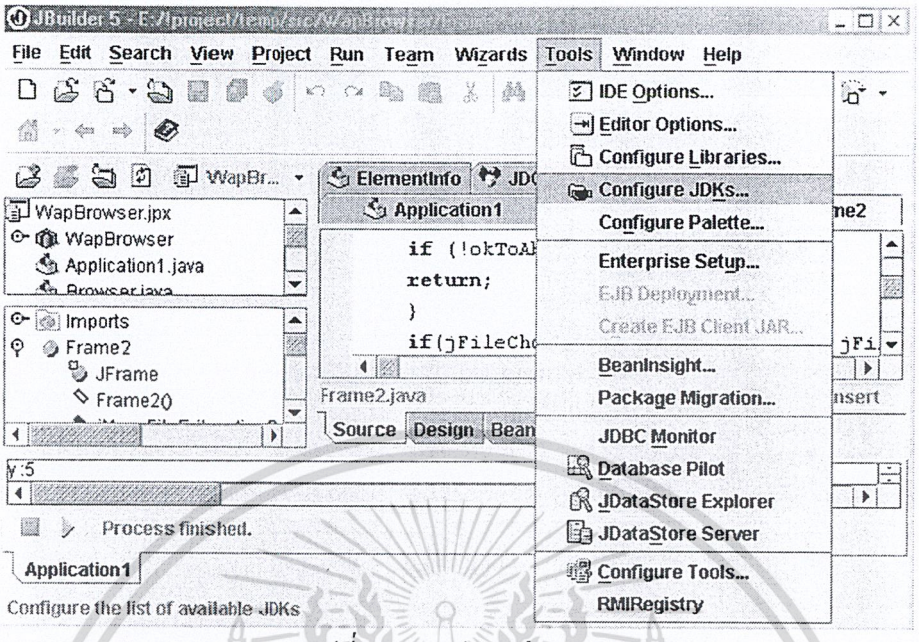
ต้องมีการสร้างโปรแกรมประยุกต์ (Application) และแต่ละโปรแกรมประยุกต์ ต้องมีการสร้างกรอบ (Frame) โดยที่ ตัวโปรแกรมประยุกต์จะมี Method ชื่อ Main อยู่เพื่อใช้ในการสั่งให้โปรแกรมทำงาน (Run) ซึ่งเมธอดที่ชื่อ Main จะเรียกให้ สร้างตัวแทน (Instance) ของ class ของกรอบต่าง ๆ ที่เป็นของโปรแกรมประยุกต์แต่ละโปรแกรม ในโครงงานแว็พบราวเซอร์ภาษาไทย 2 นี้มีการสร้างโปรแกรมประยุกต์ขึ้นมา 1 โปรแกรมชื่อ Application1 และ กรอบอีก 2 กรอบ ชื่อ Browser และ Frame2



รูปที่ ข-1การสร้าง Object Gallery

2. **การเลือก JDK (Java Develop Kit) ที่จะใช้คอมไพล์ซอสโค้ด ที่เราเขียนขึ้น**

โดยปกติ JBuilder5 จะใช้ JDK1.3 เป็นตัวมาตรฐานอยู่แล้ว แต่สำหรับการเปลี่ยน JDK ไปเป็นรุ่น (Version) อื่นก็สามารถทำได้ โดยคลิกที่เมนู Tool แล้วเลือก Configure JDKs... จะปรากฏหน้าต่าง "Configure JDKs" ให้คลิกที่ปุ่ม New... เพื่อสร้าง JDK ที่จะใช้อ้างอิง

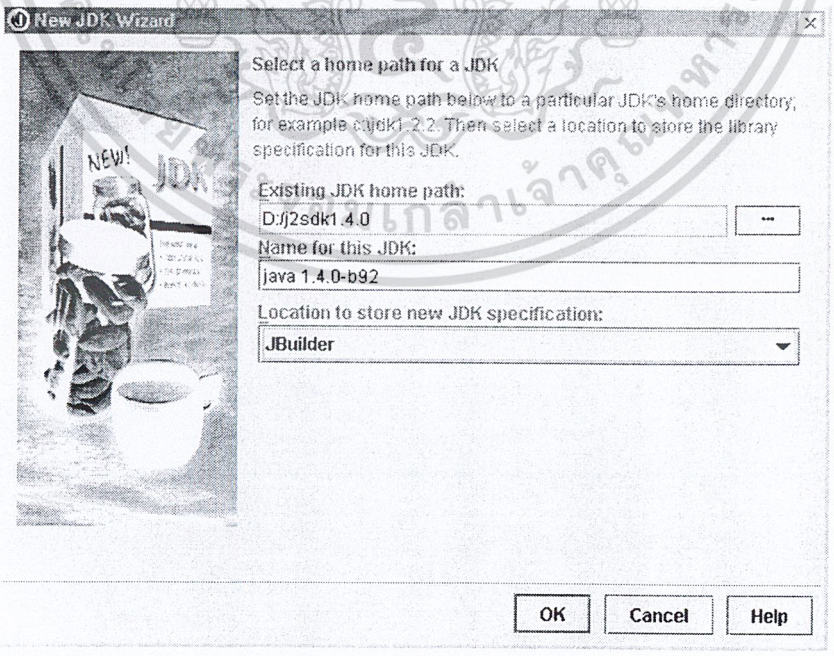


รูปที่ ข-2 การปรับแต่ง JDK

2.1. โดยที่ช่อง Existing JDK home path ให้เลือกตำแหน่งของ JDK ที่เราต้องการเลือกไว้ (เช่น "c:\jdk1.4.0")

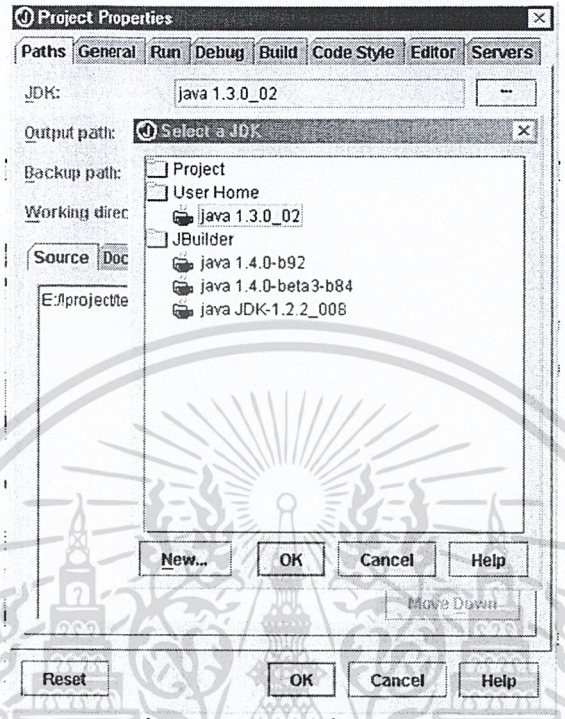
2.2. หลังจากนั้น จะปรากฏชื่อของ JDK นี้ขึ้นที่ช่อง Name of this JDK

2.3. ส่วนที่ช่อง Location to store new JDK specification ให้เลือก JBuilder เพื่อให้ผู้ใช้ทุกคนบนเครื่องคอมพิวเตอร์นี้ใช้ร่วมกันได้



รูปที่ ข-3 เลือกตำแหน่งของ JDK

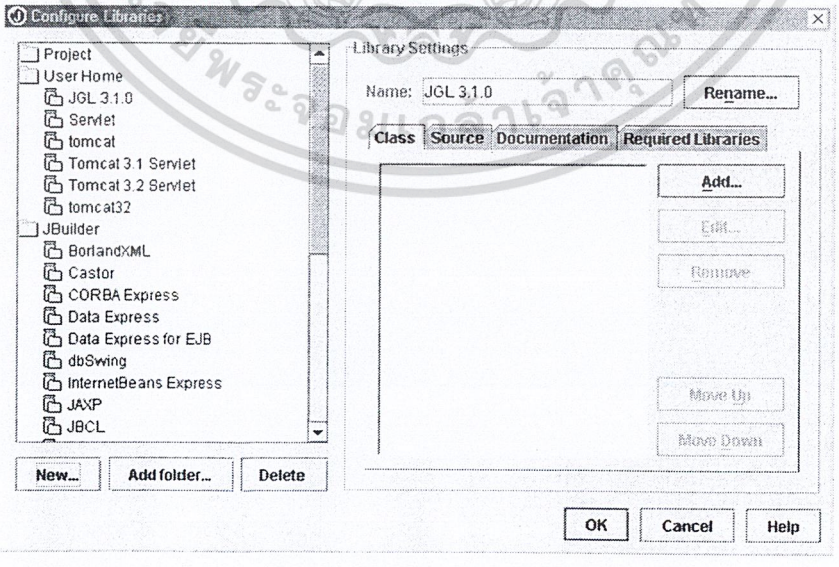
หลังจากนั้นจะได้ JDK ตัวใหม่สำหรับอ้างอิงแล้ว เราจะเลือก JDK นี้เมื่อใดก็ได้ โดยคลิกที่เมนู Project แล้วเลือก Project Properties จะปรากฏหน้าต่างขึ้นมา โดยที่ป้าย Paths ช่อง JDK เราสามารถเลือก JDK ที่เราตั้งขึ้นมาใหม่ได้



รูปที่ ข-4 เลือก JDK ที่สร้างไว้แล้ว

3. **การเพิ่มไลบรารีเพื่อนำมาใช้กับโครงงาน**

สามารถทำได้เช่นเดียวกันกับการเพิ่ม JDK โดยคลิกที่เมนู Tool เลือก Configure Libraries... จะปรากฏหน้าต่างขึ้นมา

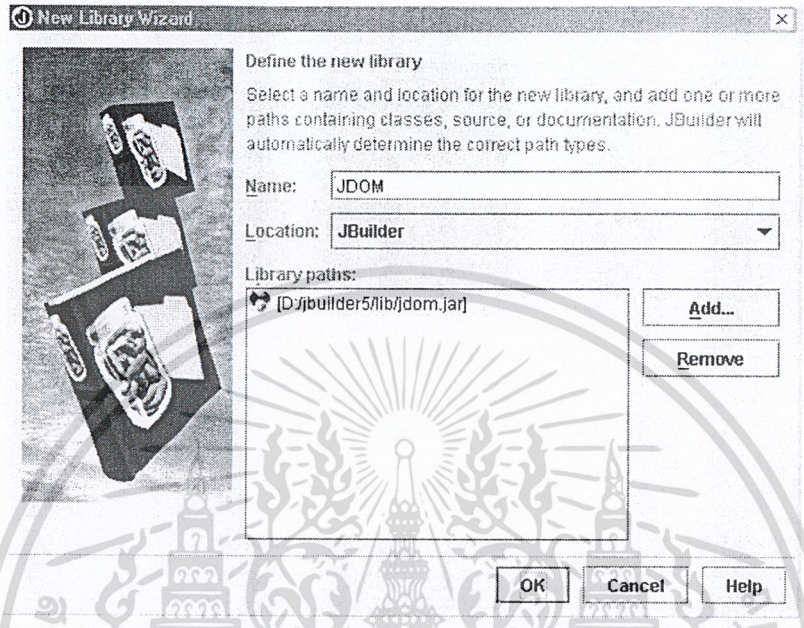


รูปที่ ข-5 การปรับแต่งไลบรารี

3.1. ที่ช่อง Name ให้ตั้งชื่อของไลบรารีที่ต้องการเพิ่ม

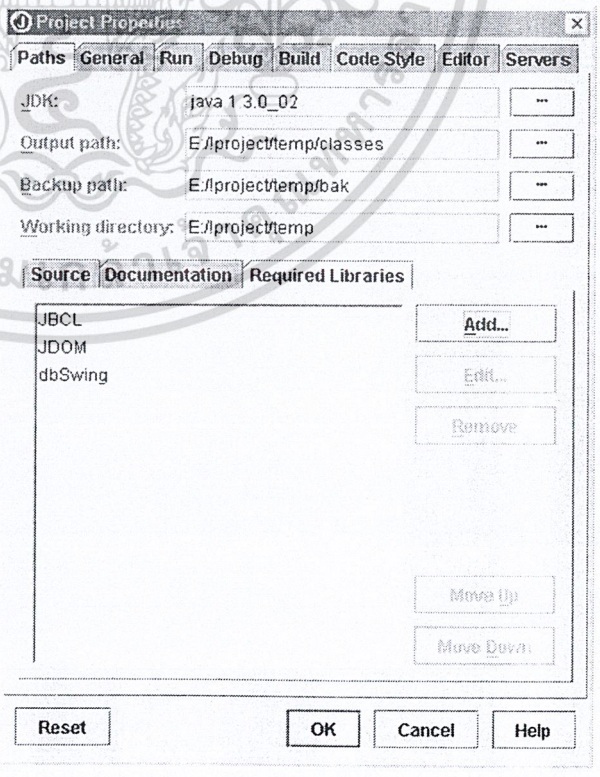
3.2. ที่ช่อง Location เลือกไปที่ JBuilder เพื่อให้ผู้ใช้ทุกคนบนเครื่องคอมพิวเตอร์นี้ใช้ร่วมกันได้

3.3. ที่ช่อง Library paths คลิกที่ปุ่ม add แล้วเลือกตำแหน่งที่เก็บไลบรารี ที่เราต้องการเพิ่ม



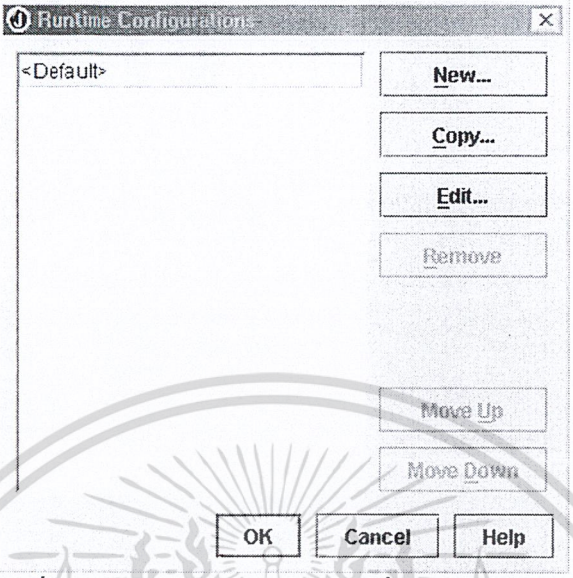
รูปที่ ข-6 การตั้งชื่อไลบรารี และระบุตำแหน่ง เพื่อสร้าง Library

หลังจากนั้น เราสามารถเพิ่มไลบรารีที่เราสร้างขึ้นมาใหม่เพื่อนำไปใช้กับโครงงานของเราได้ โดยเลือกเมนู Project เลือก Project Properties จะปรากฏหน้าต่างขึ้นมา ที่ป้าย Path จะพบป้าย Requierd Libraries คลิก Add... แล้วเลือกไลบรารี ที่เราเพิ่มเข้าไป



รูปที่ ข-7 การเลือกไลบรารีที่สร้างเอาไว้แล้ว

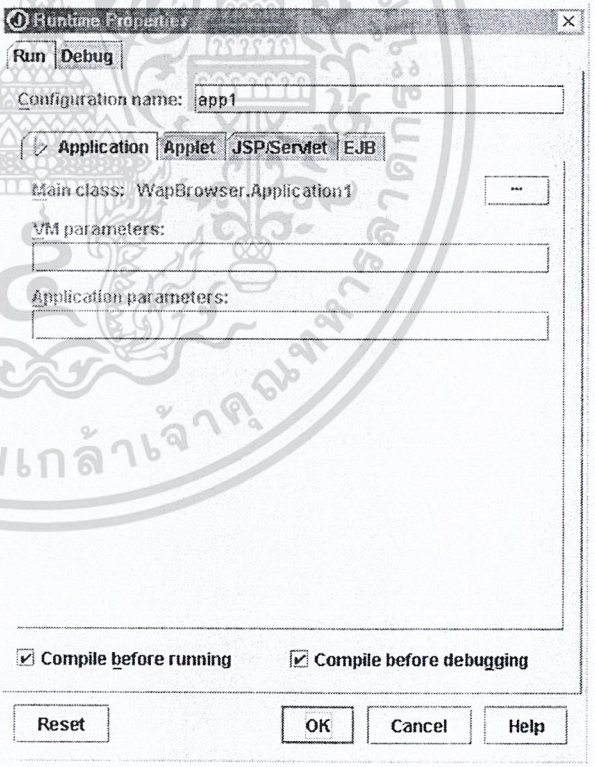
4. การเลือกโปรแกรมประยุกต์เพื่อสั่งให้โปรแกรมทำงานโดยที่โปรแกรมประยุกต์ที่เลือกต้องมีเมธอดชื่อ Main



รูปที่ ข-8 การระบุโปรแกรมประยุกต์ที่เราเลือกให้ทำงาน

4.1. ที่เมนู Run เลือก Configurations… จะปรากฏหน้าต่าง Runtime Configurations ขึ้นมา

4.2. คลิกที่ปุ่ม New จะปรากฏหน้าต่าง Runtime Properties ขึ้นมาให้เราตั้งชื่อที่ช่อง Configuration name และที่ป้าย Application ที่ช่อง Main class ให้เลือกคลาสที่เราต้องการสั่งให้ทำงาน ซึ่งตัวมาตรฐาน จะเป็น Application1 เมื่อเราเลือกแล้วทุกครั้งที่เราสั่งให้โปรแกรมทำงานก็จะเรียก คลาสของโปรแกรมประยุกต์ หรือกรอบ ที่เราเลือกเอาไว้โดยโปรแกรมประยุกต์หรือกรอบเหล่านั้น ต้องมีเมธอดชื่อ Main



รูปที่ ข-9 การเลือกการปรับตั้งค่าในการรันโปรแกรม

5. การดีบักที่เมนู Run เลือก Debug Project หลังจากนั้นเราสามารถเลือกได้ดังนี้

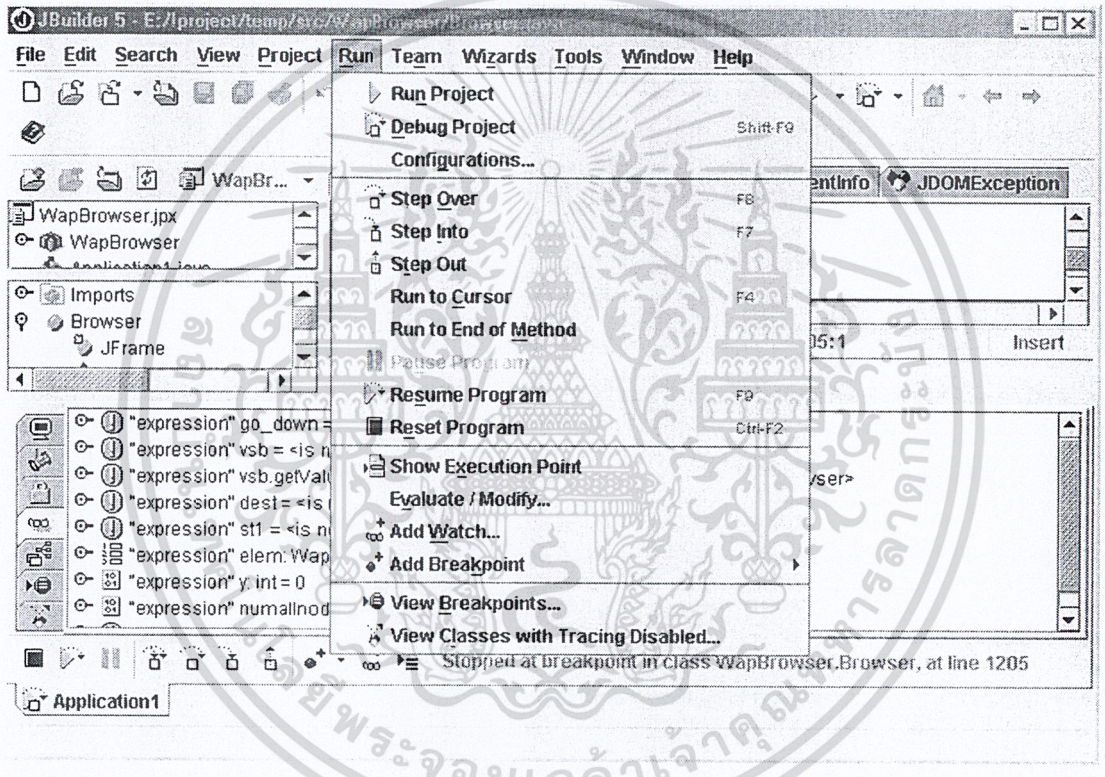
   5.1. Step Over คือการทำทีละขั้น แต่ถ้าพบการเรียกเมธอดให้ทำเมธอดนั้นให้เสร็จทันที

   5.2. Step Into เหมือนกับ Step Over แต่ถ้าพบการเรียกเมธอดจะเข้าไปทำทีละขั้นในเมธอดที่เรียกใช้นั้น ๆ

   5.3. Run to Cursor คือการทำงานไปจนถึง บรรทัด ที่เคอร์เซอร์อยู่

   5.4. Add Breakpoint คือเพิ่มบรรทัดในการหยุดพักการรัน

เมื่อมีการ ดีบัก Project จะมีป้ายต่าง ๆ ที่ส่วน Message (ด้านล่างของ JBuilder)



รูปที่ ข-10 การดีบักโปรเจ็ค

บรรณานุกรม

[1] ดร.วีระศักดิ์ ซึงถาวร: "JAVA Programing VolumeI. ", บริษัท ซีเอ็ดยูเคชั่น จำกัด(มหาชน), 2543

[2] สราวุธ อ้อยศรีกุล:" เปิด Mobile Internet ด้วย...WAP . ", บริษัท วิตตี้ กรุ๊ป จำกัด,2544

[3] Stephen A.Thomas:" HTTP Essentials protocol for Secure, Scaleable Web sites. ",Wiley Computer Publisher,John Wiley &Sons,Inc.

[4] Patrick Chan & Rosanna Lee:"The Java Class Libraries An Annotated Reference ", Corporate & Professional Publishing Group.

[5] "XML Parser Reference ",http://www.ibiblio.org/xml.

[6] " WML Reference Version 1.1 ",http://www.forum.nokia.com .

[7] H.M. Deitel, P.J. Deitel: "JAVA HOW TO PROGRAM", PRENTICE HALL, 1999.

[8] ขยัน จันทรสถาพร:"เรียนลัด xml ฉบับรู้เต็มร้อย",Success Media, 2544.

[9] นายคมสันต์ ทิวงศ์, นายบุญร่วม ศิริพันธ์: แว็พบราวเซอร์ภาษาไทย, 2543