บทที่ 7 ชั้นสื่อสารโปรแกรมประยุกต์ (The Application Layer)

ภายหลังจากที่ได้ศึกษาการทำงานในชั้นสื่อสารทั้ง 6 ชั้นต้นมาแล้วซึ่งทำหน้าที่ในการให้บริการการสื่อสารที่ไว้วางใจได้แต่ก็ไม่ได้เป็นส่วนที่ติดต่อกับผู้ใช้โดยตรง ในบทนี้จะได้กล่าวถึงชั้นสื่อสารชั้นที่ 7 หรือชั้นสื่อสารโปรแกรมประยุกต์ (Application Layer) ซึ่งเป็นที่อยู่ของโปรแกรมประยุกต์ทั้งหลาย

อย่างไรก็ตาม แม้ว่าจะเป็นชั้นสื่อสารโปรแกรมประยุกต์ แต่ก็ยังต้องอาศัยโพรโตคอลต่างๆ มาสนับสนุนการทำงานเพื่อให้โปรแกรมประยุกต์ของผู้ใช้สามารถทำงานได้ บทนี้จะได้กล่าวถึง DNS ซึ่งทำหน้าที่ในการจัดการเกี่ยวกับชื่อที่ใช้ในระบบอินเตอร์เน็ต จากนั้นจะได้ศึกษาโปรแกรมประยุกต์ที่มีใช้งานจริงคือ อีเมลล์, World Wide Web, และมัลติมีเดีย

7.1 DNS-The Domain Name System

ในทางทฤษฎีแล้ว โปรแกรมอาจหมายถึงโฮส เมลบ็อกซ์ หรือทรัพยากรอื่นใดที่อ้างอิงได้ด้วยหมายเลขที่อยู่บนระบบเครือข่าย เช่น IP address หมายเลขที่อยู่เหล่านี้มักจะเป็นสิ่งที่จดจำได้ยาก และการส่งอีเมลล์ไปยัง tena@128.111.24.41 หมายความว่า ถ้า ISP (Internet Service Provider) ของ tena หรือองค์กรที่เธอทำงานอยู่ไปย้ายเครื่องเมลล์เซิบเวอร์ไปยังเครื่องอื่นที่ใช้หมายเลข IP ที่แตกต่างไปจากเดิมจะทำให้ที่อยู่อีเมลล์ของ tena เปลี่ยนแปลงไปด้วย ดังนั้นจึงได้มีการนำเอาชื่อที่สะกดด้วยรหัส ASCII (ตัวอักษรตามปกติที่ใช้ในเครื่องคอมพิวเตอร์) มาทำการตั้งชื่อเครื่องซึ่งเป็นการแยกออกจากหมายเลขเครื่อง ด้วยวิธีการนี้ที่อยู่ของ Tena จึงเปลี่ยนไปอยู่ในรูป tena@art.ucsb.edu อย่างไรก็ตามระบบเครือข่ายมีความเข้าใจเฉพาะหมายเลขที่อยู่ที่เป็นตัวเลข เช่น หมายเลข IP ที่ยกมาข้างต้น ดังนั้นจึงต้องมีกลไกบางอย่างที่นำมาใช้ในการแปลงที่อยู่ในรูปรหัส ASCII ให้กลายเป็นหมายเลขที่อยู่บนระบบเครือข่าย ในหัวข้อต่อไปนี้จะได้กล่าวถึงวิธีการแปลงที่อยู่ ที่นำมาใช้ในระบบอินเตอร์เน็ต

ย้อนกลับไปในระบบ ARPANET จะมีแฟ้มข้อมูลอยู่แฟ้มหนึ่งชื่อ hosts.txt ที่แสดงรายการชื่อของโฮสและหมายเลข IP ในทุกๆ คืน โฮสทุกตัวที่อยู่ในความดูแลของสถานที่หนึ่งจะทำการอ่านข้อมูลที่อยู่ในแฟ้มข้อมูลนี้สำหรับระบบเครือข่ายที่มีเครื่องคอมพิวเตอร์อยู่ไม่มากนักวิธีการนี้ก็สามารถใช้งานได้ดี

อย่างไรก็ตาม ทุกคนทราบดีว่าในสถานที่ที่มีเครื่องคอมพิวเตอร์อยู่นับพันเครื่องหรือหลายพันเครื่องทั้งเครื่องมินิคอมพิวเตอร์และพีซีที่เชื่อมต่อเข้ากับระบบเครือข่ายจะไม่สามารถใช้วิธีการนี้ได้ตลอดไปด้วยเหตุผลหนึ่งคือขนาดของแฟ้มข้อมูลนี้ซึ่งจะโตขึ้นเรื่อยๆ จะมีขนาดใหญ่เกินไป อย่างไรก็ตาม สิ่งที่มีความสำคัญมากกว่าก็คือจะมีความขัดแย้งในเรื่องชื่อของโฮสอยู่เสมอนอกจากชื่อของโฮสจะได้รับการจัดการที่ศูนย์กลางเพียงแห่งเดียวซึ่งเป็นสิ่งที่ยากที่จะจินตนาการได้ว่าระบบเครือข่ายที่มีขนาดใหญ่โตมหาศาลนี้จะมีปริมาณข้อมูลอยู่มากเพียงใดและจะต้องใช้เวลาเท่าใดในการจัดการให้เรียบร้อย เพื่อแก้ปัญหานี้จึงได้มีการคิดค้นระบบ DNS (Domain Name System) ขึ้นมาใช้งาน

สิ่งที่สำคัญที่สุดของระบบ DNS นี้ก็คือ การออกแบบโครงสร้างลำดับขั้นของการตั้งชื่อโดยอาศัยโดเมนเป็นแกนหลักและระบบฐานข้อมูลแบบกระจายสำหรับการสร้างระบบชื่อนี้ขึ้นใช้งาน หน้าที่หลักคือการใช้สำหรับการแปลงชื่อโฮสและชื่ออีเมลล์ให้เป็นหมายเลขที่อยู่ IP และสามารถใช้งานด้านอื่นได้ด้วย DNS ได้รับการกำหนดมาตรฐานไว้ใน RFC 1034 และ 1035

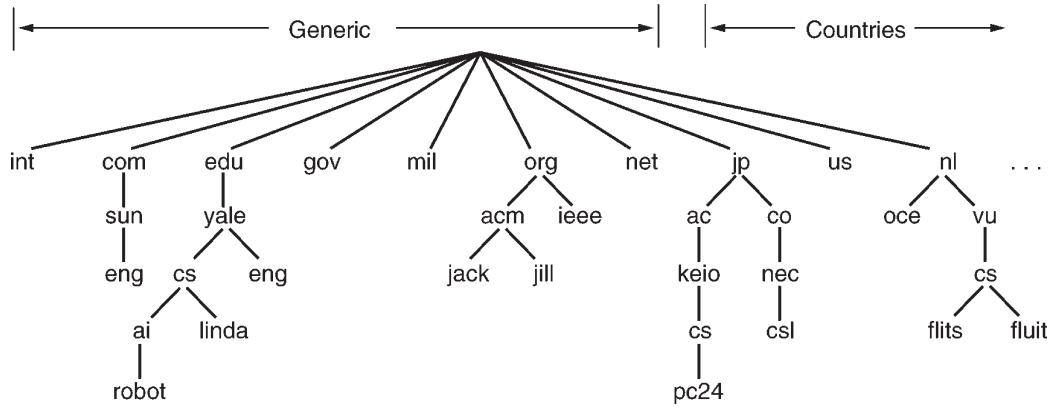
กล่าวโดยสั้นๆ วิธีการทำงานของ DNS เป็นดังนี้ ในการแปลงชื่อไปเป็นหมายเลขที่อยู่ IP โปรแกรมประยุกต์จะเรียกใช้ library ชื่อ resolver และจัดการส่งชื่อ (ASCII) มาให้เป็นพารามิเตอร์ตัวหนึ่ง รูป 6-6 ได้แสดงตัวอย่างการใช้ resolver ชื่อ "gethost-byname" Resolver จะส่ง UDP แพ็กเก็ตไปยัง DNS server ซึ่งจะทำการค้นหาชื่อนั้นและส่งหมายเลข IP กลับมาให้แก่ผู้เรียก เมื่อได้รับหมายเลข IP แล้วโปรแกรมประยุกต์นั้นก็จะจัดตั้งการเชื่อมต่อ TCP ขึ้นมาหรือจัดการส่ง UDP แพ็กเก็ตไปยังเป้าหมายที่ต้องการ

7.1.1 The DNS Name Space

การบริหารจัดการกลุ่มของชื่อขนาดใหญ่ที่มีการเปลี่ยนแปลงเกิดขึ้นอยู่เสมอ เป็นปัญหาใหญ่มากในระบบไปรษณีย์ การบริหารจัดการชื่อจะกระทำโดยการบังคับให้ต้องเขียนชื่อประเทศ ชื่อจังหวัดหรือรัฐ ชื่อเมือง และชื่อถนนพร้อมทั้งเลขที่บ้าน ในส่วนของที่อยู่ผู้รับ โดยการกำหนดให้ใช้โครงสร้างลำดับขั้นแบบนี้ทำให้ไม่เกิดความสับสนในที่อยู่ของผู้รับ ระบบ DNS ก็มีลักษณะคล้ายคลึงกัน

ระบบอินเตอร์เน็ตได้ถูกแบ่งออกเป็นส่วนๆ เรียกว่า โดเมน (Domain) มากกว่า 200 ส่วน แต่ละโดเมนจะครอบคลุมการให้บริการแก่โฮสจำนวนหนึ่ง แต่ละโดเมนจะถูกแบ่งออกเป็นส่วนย่อยเรียกว่า subdomain ซึ่งก็อาจจะถูกแบ่งออกไปเป็นส่วนย่อยเล็กลงไปอีก โดเมนเหล่านี้สามารถเขียนแทนได้ด้วยโครงสร้างแบบต้นไม้ดังแสดงในรูป 7-1 ใบของต้นไม้จะใช้แทนความหมายของโดเมนที่ไม่มีการแบ่งย่อยลงไปอีก (แต่มีเครื่องคอมพิวเตอร์อยู่) โดเมนหนึ่งอาจประกอบด้วยโฮสเพียงเครื่องเดียวหรืออาจจะหมายถึงองค์กรที่มีโฮสอยู่นับพันเครื่องก็ได้

โดเมนที่อยู่ในระดับบนสุดแบ่งออกเป็นสองชนิดคือ ทั่วไป (generic) กับ ประเทศ (countries) โดเมนชนิดทั่วไปที่มีตั้งแต่เริ่มก่อตั้งระบบอินเตอร์เน็ตได้แก่ "com" (commercial) "edu" (education) "gov" (US. government) "int" (international organizations) "mil" (the U.S. armed forces) "net" (network providers) และ "org" (nonprofit organizations) โดเมนที่เป็นชื่อประเทศประกอบด้วยหนึ่งประเทศต่อหนึ่งชื่อ ดังที่ได้กำหนดไว้ใน ISO 3166



รูป 7-1 ส่วนหนึ่งของ Internet domain name space

ในเดือนพฤศจิกายน ค.ศ. 2000 องค์กร ICANN (Internet Corporation for Assigned Names and Numbers) ได้กำหนดชื่อโดเมนระดับชั้นบนสุดขึ้นมาใหม่อีก 4 ชื่อคือ "biz" (businesses), "info" (information) "name" (peoples names) และ "pro" (professions) นอกจากนี้ยังได้กำหนดชื่อใหม่ขึ้นมาอีกสามชื่อคือ "aero" (aerospace industry) "coop" (Co-operatives) และ "museum" (museums)

นอกจากที่กล่าวถึงแล้วเมื่อระบบอินเตอร์เน็ตได้ถูกนำมาใช้ในทางการค้ามากยิ่งขึ้นก็ยิ่งมีปัญหาการโต้แย้งมากยิ่งขึ้น เช่น โดเมน "pro" ได้ถูกกำหนดให้เป็นกลุ่มของผู้มีอาชีพเฉพาะที่มีใบรับรองการประกอบอาชีพ แต่ก็ยังมีปัญหาขึ้นมาว่า ใครคือผู้มีอาชีพ และจะต้องได้รับการรับรองจากใคร แพทย์หรือนักกฎหมาย นั้นอยู่ในกลุ่มนี้แน่นอนแต่สำหรับนักถ่ายรูปอิสระ ครูสอนเปียนโน นักมายากล ผู้ซ่อมประปา ผู้กำจัดปลวก และอื่นๆ จะให้จัดว่าผู้มีอาชีพเหล่านี้ควรอยู่ใน โดเมน "pro" ด้วยหรือไม่ และถ้าใช่ จะให้ใครเป็นผู้ให้การรับรองในอาชีพอื่นๆ อีก

โดยทั่วไปแล้ว การจัดตั้งโดเมนในระดับที่สอง (second-level domain) เช่น "name-of-company.com" นั้นมีความง่ายกว่า นั่นคือการไปลงทะเบียนยังโดเมนในระดับบนที่เกี่ยวข้องกับองค์กรของตนเอง เพื่อทำการตรวจสอบว่าชื่อที่ต้องการจะตั้งขึ้นใหม่นั้นยังว่างอยู่หรือไม่ หรือว่าเป็นชื่อที่องค์กรอื่นได้ลงทะเบียนใช้งานไปแล้ว ถ้าชื่อนั้นใช้ได้ผู้ลงทะเบียนก็มักจะต้องเสียค่าธรรมเนียมเป็นรายปี เพื่อให้สามารถใช้ชื่อนั้นได้

แต่ละโดเมนจะถูกตั้งชื่อโดยใช้ชื่อ path ตั้งแต่ชื่อองค์กรขึ้นไปจนถึงชื่อโดเมนระดับบนสุดและชื่อ root (ซึ่งไม่มีชื่อ) ชื่อแต่ละส่วนจะถูกแยกจากกันโดยใช้เครื่องหมาย "." (ออกเสียงว่า "dot") ดังนั้นแผนกเอ็นจิเนียของบริษัท Sun Microsystems จึงอาจใช้ชื่อว่า "eng.sun.com" สังเกตว่าชื่อในส่วน "eng" ในที่นี้จะไม่สับสนกับชื่อ "eng.yale.edu" ซึ่งอาจจะหมายถึง Yale English department

ชื่อโดเมนอาจใช้ในลักษณะของ absolute หรือ relative ก็ได้ ชื่อเต็มหรือ Absolute domain name จะลงท้ายด้วย "." เสมอในขณะที่ชื่อย่อหรือ relative name ไม่จำเป็นต้องลงท้ายด้วย "." ชื่อย่อจะต้องได้รับการแปลความหมายในบางลักษณะเพื่อให้ทราบชื่อเต็มที่เป็นชื่อเฉพาะ ในทั้งสองกรณีชื่อโดเมนจะหมายถึง โหนดใดโหนดหนึ่งในต้นไม้นั้นและโหนดทั้งหมดที่อยู่ภายใต้โหนดนั้น

ชื่อโดเมนนั้นมีลักษณะที่เรียกว่า case insensitive นั่นคือ "edu" "Edu" "EDU" ล้วนแต่มีความหมายเป็นชื่อเดียวกัน ชื่อในระดับรองลงมาอาจมีความยาวได้ไม่เกิน 63 ตัวอักษร และชื่อเต็มจะต้องมีความยาวรวมกันไม่เกิน 255 ตัวอักษร

โดยพื้นฐานแล้วโดเมนสามารถถูกใส่เข้าไปในต้นไม้ได้สองวิธี เช่น "cs.yale.edu" อาจถูกใส่เข้าไปในใต้โดเมน "us" ซึ่งจะมีชื่อเต็มเป็น "cs.yale.ct.us" แต่อย่างไรก็ตามชื่อโดเมนขององค์กรส่วนใหญ่ในสหรัฐอเมริกาจะอยู่ภายใต้ชื่อทั่วไป และองค์กรส่วนใหญ่ในประเทศอื่นๆ จะอยู่ภายใต้ชื่อประเทศของตนเอง แต่ก็ไม่มีกฎข้อบังคับใดที่จะบังคับให้ใช้ชื่อใดชื่อหนึ่ง

แต่ละโดเมนจะควบคุมการบริหารโดเมนภายใต้โดเมนของตัวเอง เช่น ประเทศญี่ปุ่นมีชื่อโดเมน "ac.jp" และ "co.jp" ซึ่งเป็นโดเมนสำเนาของ "edu" และ "com" ในขณะที่ประเทศเนเธอร์แลนด์

ไม่มีทั้งสองโดเมนย่อยนี้และใช้ชื่อโดเมนทั้งหมดอยู่ภายใต้ "nl" ตัวอย่างชื่อโดเมนของมหาวิทยาลัย 3 แห่งเป็นดังนี้

1. cs.yale.edu (Yale University, in the United States)
2. cs.vu.nl (Vrije University, in The Netherlands)
3. cs.keio.ac.jp (Keio University, in Japan)

ในการสร้างโดเมนขึ้นมาใหม่ จะต้องได้รับอนุญาตจากโดเมนหลักก่อนเสมอ เช่น VLSI group ที่เริ่มก่อตั้งขึ้นในมหาวิทยาลัย Yale ต้องการที่จะใช้ชื่อโดเมนเป็น "vlsi.cs.yale.edu" กลุ่มนี้จะต้องไปขออนุญาตจากหน่วยงานที่เป็นเจ้าของ "cs.yale.edu" ในลักษณะเดียวกันถ้ามีมหาวิทยาลัยตั้งขึ้นมาใหม่และต้องการลงทะเบียนใช้ชื่อภายใต้ "edu" ก็จะต้องไปขออนุญาตของหน่วยงานที่รับผิดชอบ "edu" เสียก่อน ด้วยวิธีการนี้จะสามารถขจัดปัญหาความขัดแย้งเรื่องชื่อโดเมนได้ และแต่ละโดเมนเองก็จะต้องดูแลโครงสร้างโดเมนส่วนย่อยของตนเอง เมื่อโดเมนใหม่ได้รับการลงทะเบียนเรียบร้อยแล้วก็จะสามารถสร้างโดเมนย่อยนั้นขึ้นใช้งานได้โดยไม่ต้องไปขออนุญาตจากโดเมนที่อยู่ในระดับบนขึ้นไปอีก เช่น "vlsi" จะต้องขออนุญาตจาก "cs.yale.edu" แต่ไม่ต้องไปขออนุญาต "yale.edu" หรือ "edu" เลย

การตั้งชื่อนั้นกระทำภายใต้ขอบเขตการบริหารองค์กรไม่ใช่ขอบเขตของระบบเครือข่ายองค์กร ตัวอย่างเช่น ถ้าคณะคอมพิวเตอร์และคณะวิศวกรรมไฟฟ้าตั้งอยู่ในอาคารเดียวกันและอาจจะอยู่ในวง LAN วงเดียวกันก็จะสามารถตั้งชื่อโดเมนแยกจากกันได้ ในทางกลับกันถ้าคณะคอมพิวเตอร์มีสาขาแยกไปอยู่ คนละอาคาร (หรือแม้แต่อยู่คนละเมือง) ก็จะสามารถใช้ชื่อโดเมนเป็นชื่อเดียวกันได้ (ทั้งคณะมีชื่อโดเมนเดียว)

7.1.2 Resource records

ทุก ๆ โดเมนไม่ว่าจะเป็นโดเมนระดับบนสุดหรือโดเมนขนาดเล็กที่มีโฮสอยู่เพียงโฮสเดียวจะมี resource records อยู่ชุดหนึ่งเสมอ สำหรับโฮสเดี่ยว สิ่งที่บรรจุอยู่ใน resource record ก็คือหมายเลขที่อยู่ IP แต่ก็อาจจะมีอย่างอื่นบรรจุอยู่ด้วยก็ได้ เมื่อโปรแกรม resolver ส่งชื่อโดเมนมาให้กับ DNS สิ่งที่จะได้รับกลับไปคือ resource record ที่เกี่ยวข้องกับชื่อโดเมนนั้น ดังนั้นหน้าที่หลักของ DNS ก็คือการแปลงชื่อโดเมนไปเป็น resource records

resource record หนึ่ง ๆ ประกอบด้วย 5 ส่วน แม้ว่าโดยปกติจะอธิบายไว้ในลักษณะของเลขฐานสองเพื่อประสิทธิภาพในการใช้งานแต่เมื่อให้แสดงรายละเอียดก็จะอยู่ในรูปแบบของตัวอักษรซึ่งจะแสดงให้เห็นแต่ละ resource record ต่อหนึ่งบรรทัด โครงสร้างของการแสดงผลคือ

Domain name Time_to_live Class Type Value

คอลัมภ์ Domain name จะบอกชื่อโดเมนของ record นั้น โดยปกติอาจจะมีหลาย record ในโดเมนหนึ่ง ๆ และแต่ละฐานข้อมูลจะเก็บข้อมูลของหลาย ๆ โดเมน คอลัมภ์นี้จึงมักจะถูกใช้เป็นคีย์หลักสำหรับการค้นหาข้อมูลที่ต้องการ ลำดับที่เก็บข้อมูลอยู่นั้นไม่ได้บอกความสำคัญใด ๆ

คอลัมภ์ Time_to_live บอกให้ทราบถึงสถานะของแต่ละ record ข้อมูลที่มีความเสถียรมากจะได้รับกำหนดค่าให้สูงมาก เช่น 86400 (จำนวนของวินาทีในหนึ่งวัน) ข้อมูลที่อาจเปลี่ยนแปลงได้ก็จะมีค่าน้อย เช่น 60 (หนึ่งนาที) จะกลับมากล่าวถึงเรื่องนี้อีกครั้งหนึ่งเมื่อกล่าวถึงเรื่อง cache

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

รูป 7-2 The principal DNS resource record types for IPv4

คอลัมภ์ที่สามคือ Class สำหรับข้อมูลที่ใช้กับอินเตอร์เน็ตจะมีค่าเป็น IN เสมอ สำหรับข้อมูลของระบบอื่นอาจใช้ค่าอื่นๆ ได้ แต่โดยทั่วไปมักไม่ค่อยได้ใช้งานมากนัก

คอลัมภ์ Type ใช้บอกชนิดของ record รายละเอียดแสดงไว้ในรูป 7-2

SOA record นำเสนอชื่อของแหล่งที่มาหลักของข้อมูลเกี่ยวกับชื่อของ zone server (อธิบายในลำดับต่อไป) ที่อยู่อีเมลล์ของผู้บริหารระบบ หมายเลขลำดับเฉพาะ flags และ timeouts

Record ที่มีความสำคัญมากที่สุดคือ record A (Address) ซึ่งจะเป็นที่อยู่ IP ขนาด 32 บิต ของโฮสตัวหนึ่ง โฮสแต่ละตัวจะต้องมีหมายเลข IP อย่างน้อยหนึ่งหมายเลขเพื่อให้คอมพิวเตอร์เครื่องอื่นสามารถติดต่อได้ โฮสบางส่วนจะมีการเชื่อมต่อกับระบบเครือข่ายมากกว่าหนึ่งจุด ซึ่งในกรณีนี้จุดเชื่อมต่อแต่ละจุดจะมี type A resource record เป็นของตนเองหรืออาจกล่าวได้ว่าจะต้องมี record นี้ประจำหมายเลข IP แต่ละเลข DNS จะสามารถถูกกำหนดให้ใช้งานจุดเชื่อมต่อหมุนเวียนกันไป เช่น ในการร้องขอครั้งแรก DNS จะส่งค่า resource record แรกกลับไป เมื่อมีคำร้องขอครั้งที่สองก็จะส่ง resource record อันที่สองและเวียนไปเรื่อยๆ จนครบทุกตัวแล้วจึงวนกลับไปที่ record แรกอีกครั้งหนึ่ง

Record ที่มีความสำคัญลำดับที่สองคือ MX record ใช้ในการระบุชื่อโฮสที่มีหน้าที่รับอีเมลล์สำหรับโดเมนหนึ่งๆ มีความจำเป็นต้องใช้ record ชนิดนี้เนื่องจากบางเครื่องก็ไม่ได้เตรียมพร้อมที่จะรับอีเมลล์หรือไม่สามารถที่รับอีเมลล์ได้โดยตรง เช่น บางคนต้องการที่จะส่งอีเมลล์ไปยัง bill@microsoft.com โฮสที่ทำการส่งอีเมลล์จะต้องค้นหา mail server ที่ microsoft.com ที่พร้อมที่จะรับอีเมลล์ MX record จะสามารถให้ข้อมูลนี้ได้

NS record จะทำการระบุ name server ตัวอย่างเช่น ทุกๆ ฐานข้อมูลใน DNS โดยปกติมักจะมี NS record ประจำสำหรับทุกๆ โดเมนที่อยู่ในระดับบนขึ้นไป ดังนั้นช่วยให้สามารถส่งอีเมลล์ขึ้นไปยังโดเมนระดับบนได้ จะกลับมากล่าวถึงหัวข้อนี้อีกในภายหลัง

CNAME record ช่วยในการสร้างชื่อรอง (aliases) ตัวอย่างเช่น ผู้ใช้คนหนึ่งที่มีความคุ้นเคยกับการตั้งชื่อบนระบบอินเตอร์เน็ตต้องการที่จะส่งข้อความไปยังผู้ใช้อีกคนหนึ่งที่ใช้ชื่อว่า "paul" ที่ทำงานอยู่ที่ computer science department at M.I.T. ก็คาดเดาว่าชื่อติดต่อของคนผู้นี้น่าจะเป็น "paul@cs.mit.edu" แต่ความจริงแล้วชื่อนี้ไม่สามารถใช้งานได้เพราะชื่อโดเมนของคณะคอมพิวเตอร์คือ

"lcs.mit.edu" อย่างไรก็ตาม เพื่อเป็นการให้บริการแก่ผู้ที่ไม่ทราบข้อเท็จจริงนี้ ทาง M.I.T. สามารถสร้าง CNAME ขึ้นมาเพื่อใช้ในการเปลี่ยนทิศทางข้อมูลให้เป็นไปตามที่ต้องการได้ เช่น

```
cs.mit.edu    86400    IN    CNAME    lcs.mit.edu
```

PTR record มีความคล้ายคลึงกับ CNAME record ในการแปลงชื่อหนึ่งไปเป็นชื่ออื่น อย่างไรก็ตาม CNAME ซึ่งทำหน้าที่เป็นเสมือนการกำหนด macro อย่างหนึ่ง แต่ PTR เป็นข้อมูลชนิดหนึ่งของ DNS ซึ่งการนำไปใช้งานนั้นขึ้นอยู่กับค่าที่กำหนด ในทางปฏิบัติ มักจะนำมาใช้ในการแปลงชื่อให้เป็นหมายเลข IP เสมอ เพื่อช่วยในการค้นหาหมายเลข IP และการส่งค่าคืนไปยังเครื่องที่เรียกใช้ วิธีการนี้เรียกว่า reverse lookups

HINFO record ช่วยให้ผู้ใช้สามารถทราบได้ว่าเครื่องและระบบปฏิบัติการของเครื่องโดเมนเป็นชนิดอะไร ท้ายที่สุด TXT record ช่วยให้โดเมนสามารถแสดงตนเองได้ในวิธีการเฉพาะ record ทั้งสองชนิดนี้มีไว้เพื่อความสะดวกในการใช้งานของผู้ใช้ ซึ่งไม่ได้เป็นส่วนที่บังคับให้ใช้งาน ดังนั้นโปรแกรมทั้งหลายจึงไม่ได้คาดหวังว่าจะได้รับข้อมูลนี้(และอาจจะไม่ทราบว่าจะทำอย่างไรถ้าได้รับข้อมูลนี้)

ท้ายที่สุด คอลัมภ์ Value อาจจะเป็นตัวเลข ชื่อโดเมน หรือชื่อที่เป็นกลุ่มตัวอักษร ความหมายของค่านี้ขึ้นอยู่กับชนิดของ record คำอธิบายโดยย่อของ value fields สำหรับ record แต่ละชนิดแสดงในรูป 7-2

ตัวอย่างของข้อมูลนี้ที่ผู้ใช้อาจพบได้ในฐานข้อมูลของ DNS แสดงให้เห็นในรูป 7-3 ข้อมูลนี้แสดงส่วนหนึ่งของข้อมูลสำหรับโดเมน "cs.vu.nl" ซึ่งแสดงในรูป 7-1 ข้อมูลนี้แสดง resource record จำนวน 7 ชนิด

ข้อมูลที่ปรากฏในบรรทัดแรกให้ข้อมูลทั่วไปเกี่ยวกับโดเมน สองบรรทัดถัดมาให้ข้อมูลเกี่ยวกับสถานที่อยู่ของโดเมน จากนั้นตามด้วยสถานที่สองแห่งที่จะสามารถส่งอีเมลล์ไปได้ คำว่า "zephyr" เป็นชื่อเครื่องคอมพิวเตอร์ซึ่งเป็น mail server ตัวแรกที่จะรับข้อมูล ถ้าไม่สามารถใช้งานได้ก็จะลองที่เครื่อง "top" เป็นลำดับต่อไป

หลังจากบรรทัดว่างที่เพิ่มเข้าไปเพื่อช่วยให้อ่านข้อมูลได้ง่ายขึ้น เป็นข้อมูลที่บอกให้ทราบว่าเครื่องที่ "flits" ใช้งานอยู่นั้นคือเครื่อง Sun workstation ที่ใช้ระบบปฏิบัติการ Unix และได้ให้หมายเลข IP สองหมายเลข จากนั้นเป็น mail server จำนวน 3 เครื่อง เครื่องแรกคือ "flits" ลำดับสองคือ "zephyr" และลำดับที่สามคือ "top" ต่อไปเป็นชื่อรองของ "www.cs.vu.nl" ทำให้ที่อยู่นี้สามารถใช้งานได้โดยไม่ต้องมีเครื่องเป็นของตนเอง การตั้งชื่อรองนี้ทำให้ "cs.vu.nl" สามารถเปลี่ยน World Wide Web server ได้โดยไม่ต้องบอกให้ผู้ใช้คนอื่นๆ เปลี่ยนตามเหตุการณ์อย่างเดียวกันเกิดขึ้นกับชื่อ "ftp.cs.vu.nl"

ข้อมูล 4 บรรทัดต่อมาเป็นข้อมูลสำหรับเครื่องพีซีหรือ workstation ธรรมดาเครื่องหนึ่ง ในกรณีนี้คือ "rowboat.cs.vu.bl" ข้อมูลนี้จะบอกหมายเลข IP บอก mail server หลักและรอง และบอกข้อมูลเกี่ยวกับเครื่อง จากนั้นตามมาด้วยเครื่องที่ไม่ใช้ระบบปฏิบัตการ Unix ซึ่งไม่สามารถจะรับเมลล์ได้ด้วยตนเอง ปิดท้ายด้วยข้อมูลของเครื่องเลเซอร์พริ้นเตอร์ที่เชื่อมต่อเข้ากับระบบอินเตอร์เน็ต

```
; Authoritative data for cs.vu.nl
cs.vu.nl.        86400   IN  SOA     star boss (9527,7200,7200,241920,86400)
cs.vu.nl.        86400   IN  TXT     "Divisie Wiskunde en Informatica."
cs.vu.nl.        86400   IN  TXT     "Vrije Universiteit Amsterdam."
cs.vu.nl.        86400   IN  MX      1 zephyr.cs.vu.nl.
cs.vu.nl.        86400   IN  MX      2 top.cs.vu.nl.

flits.cs.vu.nl.  86400   IN  HINFO   Sun Unix
flits.cs.vu.nl.  86400   IN  A       130.37.16.112
flits.cs.vu.nl.  86400   IN  A       192.31.231.165
flits.cs.vu.nl.  86400   IN  MX      1 flits.cs.vu.nl.
flits.cs.vu.nl.  86400   IN  MX      2 zephyr.cs.vu.nl.
flits.cs.vu.nl.  86400   IN  MX      3 top.cs.vu.nl.
www.cs.vu.nl.    86400   IN  CNAME   star.cs.vu.nl
ftp.cs.vu.nl.    86400   IN  CNAME   zephyr.cs.vu.nl

rowboat                  IN  A       130.37.56.201
                         IN  MX      1 rowboat
                         IN  MX      2 zephyr
                         IN  HINFO   Sun Unix

little-sister            IN  A       130.37.62.23
                         IN  HINFO   Mac MacOS

laserjet                 IN  A       192.31.231.216
                         IN  HINFO   "HP Laserjet IIISi" Proprietary
```

รูป 7-3
ส่วนหนึ่งของโดเมน cs.vu.nl ในฐานข้อมูล DNS

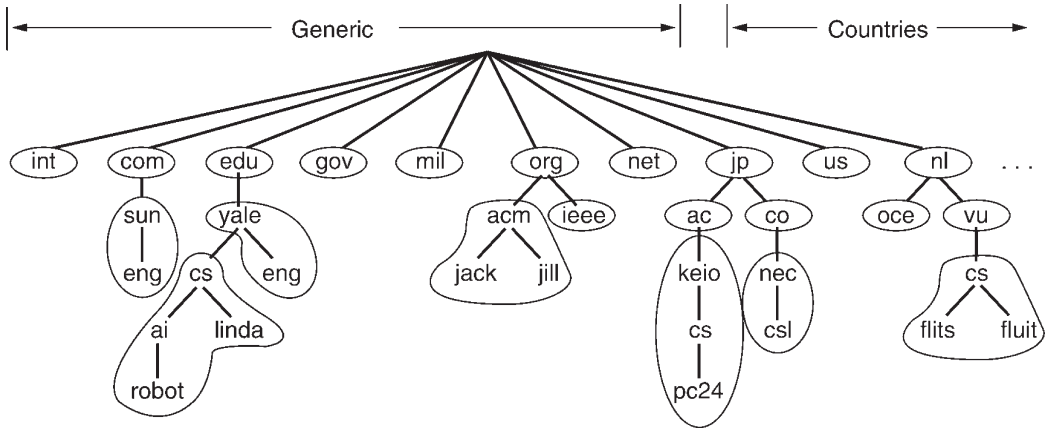
สิ่งที่ไม่ได้แสดงให้เห็นคือ หมายเลข IP จำนวนหนึ่งที่ใช้ในการค้นหาโดเมนระดับบน หมายเลข IP นี้ถูกใช้ในการหาโฮสที่อยู่ไกลออกไป แต่เนื่องจากไม่ได้เป็นส่วนหนึ่งของโดเมน "cs.vu.nl" จึงไม่มีข้อมูลอยู่ในแฟ้มข้อมูลนี้ แต่จะได้รับมาจากเครื่องเซิบเวอร์ที่ทำหน้าที่เป็น root แทนซึ่งจะเก็บหมายเลข IP ทั้งหมดไว้ใน system configuration file และถูกนำเข้ามาใน DNS cache เมื่อตอนเริ่มต้นการทำงาน ในโลกนี้มี root server อยู่ประมาณ 12 เครื่องกระจายอยู่ทั่วไป แต่ละเครื่องจะทราบหมายเลข IP ของโดเมนเซิบเวอร์ระดับบนทุกเครื่อง ดังนั้นถ้าคอมพิวเตอร์เครื่องหนึ่งทราบหมายเลข IP ของ root server อย่างน้อยหนึ่งตัวก็จะสามารถค้นหาข้อมูลชื่อ DNS ตัวใดก็ได้

7.1.3 Name Servers

ในทางทฤษฎีแล้วเครื่อง name server เพียงเครื่องเดียวจะสามารถจัดเก็บฐานข้อมูล DNS ไว้ได้ทั้งหมดและสามารถให้บริการแก่ผู้ใช้ได้ แต่ในความเป็นจริง เซิบเวอร์เครื่องนี้จะทำงานจนเกินขีดความสามารถจนกระทั่งไม่สามารถให้บริการได้ ยิ่งกว่านั้น ถ้าเครื่องนี้หยุดทำงานลงเมื่อใด ระบบอินเตอร์เน็ตทั้งระบบก็คงจะต้องหยุดทำงานไปด้วย

เพื่อหลีกเลี่ยงปัญหาที่เกี่ยวกับการมีแหล่งข้อมูลอยู่เพียงแหล่งเดียว DNS name space จึงถูกแบ่งออกเป็นหลายส่วนที่ไม่คาบเกี่ยวกันเรียกว่า "โซน" (zone) หรือเขตให้บริการ วิธีการหนึ่งที่จะแบ่งพื้นที่ให้บริการในรูป 7-1 แสดงให้เห็นในรูป 7-4 แต่ละโซนบรรจุไว้ด้วยส่วนหนึ่งของต้นไม้และมี name server ที่บันทึกข้อมูลเกี่ยวกับพื้นที่ให้บริการนั้น โดยปกติ โซนหนึ่งจะมี name server ตัวหลักหนึ่ง

รูป 7-4
ส่วนหนึ่งของ DNS name space ที่แสดงให้เห็นการแบ่งโซนการให้บริการ



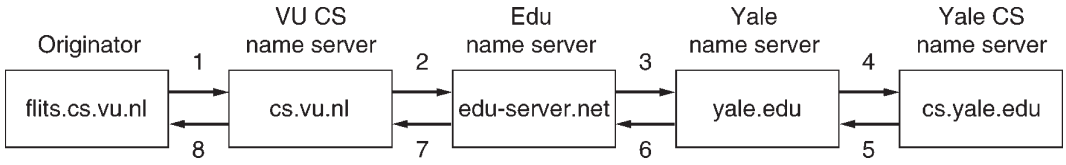
ตัวซึ่งจะได้รับข้อมูลมาจากแฟ้มข้อมูลในดิสก์ของตนเอง และอาจมี name server ตัวรองอย่างน้อยหนึ่งตัว ซึ่งจะได้รับข้อมูลไปจาก name server ตัวหลัก เพื่อเพิ่มความไว้วางใจได้ให้แก่ระบบเครื่องเซิบเวอร์ส่วนหนึ่งอาจจะอยู่ภายนอกโซนก็ได้

การแบ่งขอบเขตของโซนที่อยู่ภายในโซนโซนหนึ่งนั้นขึ้นอยู่กับผู้บริหารโซนนั้นๆ การตัดสินใจนี้ส่วนใหญ่ขึ้นอยู่กับจำนวนและสถานที่ตั้งของ name server ที่ต้องการมีใช้งาน ตัวอย่างเช่น ในรูป 7-4 Yale มีเซิบเวอร์สำหรับ yale.edu ซึ่งจะให้การดูแล eng.yale.edu แต่ไม่เกี่ยวข้องกับ cs.yale.edu ซึ่งเป็นโซนที่แยกอยู่ต่างหากและมี name server เป็นของตนเอง การตัดสินใจนี้จะต้องกระทำในกรณีที่คณะ English ไม่ต้องการที่จะมี name server เป็นของตนเอง แต่คณะคอมพิวเตอร์นั้นต้องการ ผลที่ตามมาก็คือ cs.yale.edu นั้นเป็นโซนที่แยกออกจาก yale.edu ในขณะที่ eng.yale.edu อยู่ในโซน yale.edu

เมื่อ resolver มีคำถามเกี่ยวกับชื่อโดเมน ก็จะค้นหาคำตอบโดยการส่งคำถามไปยัง name server ตัวหนึ่งที่อยู่ในโซนของตนเอง ถ้าโดเมนที่กำลังตามหานั้นอยู่ภายใต้การดูแลของ name server เช่น กำลังตามหา ai.cs.yale.edu ที่อยู่ในความดูแลของ cs.yale.edu ก็จะส่ง authoritative resource record กลับคืนมา Authoritative record คือข้อมูล record ที่ส่งมาจากผู้มีอำนาจในการบริหาร record จึงเป็นข้อมูลที่ถูกต้องเสมอ Authoritative record นั้นแตกต่างไปจาก cache record ซึ่งอาจจะล้าสมัยได้

อย่างไรก็ตาม ถ้าโดเมนนั้นอยู่ไกลออกไปและไม่มีข้อมูลเกี่ยวกับโดเมนนั้นอยู่ที่ name server ของตนเองเลย name server จะทำการส่งข่าวสารคำถามไปยัง name server ของระดับบนสุดของโดเมนที่กำลังตามหาอยู่ รูป 7-5 แสดงขั้นตอนรายละเอียดที่จะช่วยอธิบายได้ชัดเจนยิ่งขึ้น ในที่นี้ resolver บนเครื่อง flits.cs.yale.edu ต้องการทราบหมายเลข IP ของโฮส linda.cs.yale.edu ขั้นตอนที่หนึ่ง resolver จะส่งคำถามไปยัง name server ของตนเองคือ cs.vu.nl คำถามนี้บรรจุชื่อโดเมนที่กำลังตามหาอยู่นั่นคือ record type A และ class IN

สมมุติว่า name server ของโซนนั้นไม่เคยได้รับคำถามเกี่ยวข้องกับโดเมนนี้มาก่อนเลย และไม่มีข้อมูลใดๆ อยู่เลย name server ตัวนี้จะทำการค้นหาผ่าน name server ใกล้เคียง (ซึ่งสมมุติว่าไม่มี



รูป 7-5 วิธีการที่ resolver ค้นหาชื่อโดเมนที่อยู่ไกลออกไปใน 8 ขั้นตอน

เครื่องไหนทราบเลย) ก่อนที่จะส่ง UDP แพ็กเก็ตไปยังเซิบเวอร์ของ edu ซึ่งมีข้อมูลเก็บอยู่ในฐานข้อมูล "edu-server.net" มีความเป็นไปได้มากที่เซิบเวอร์จะไม่รู้จักที่อยู่ของ linda.cs.yale.edu และอาจจะไม่รู้จัก cs.yale.edu ด้วยแต่จะต้องรู้จักโดเมนที่อยู่ภายใต้ตนเองทั้งหมด ดังนั้นจึงส่งคำถามไปยัง name server ของ yale.edu (ขั้นตอนที่ 3) เซิบเวอร์ตัวนี้จะส่งคำถามต่อไปยัง cs.yale.edu (ขั้นตอนที่ 4) ซึ่งจะต้องมี authoritative resource record หลังจากนั้นก็จะจัดส่งข้อมูลนี้กลับไปยังผู้ร้องขอข้อมูลในขั้นตอนที่ 5-8 ตามลำดับ

เมื่อ record นี้เดินทางกลับมาถึง name server ของ cs.vu.nl แล้ว ข้อมูลนี้ก็จะถูกเก็บไว้ใน cache เผื่อว่าจะมีใครต้องการเรียกใช้งานอีก อย่างไรก็ตาม ข้อมูลนี้จะไม่ใช่ authoritative record เนื่องจากการแก้ไขเปลี่ยนแปลงที่ cs.yale.edu จะไม่ถูกส่งต่อไปยัง cache ในทุกเซิบเวอร์ ด้วยเหตุผลนี้ข้อมูลที่เก็บอยู่ใน cache จึงไม่ควรถูกเก็บไว้นานเกินไป นี่คือเหตุผลที่จำเป็นต้องมีเขตข้อมูล time_to_live ใส่ไว้ในแต่ละ resource record ข้อมูลนี้จะบอกให้ name server ทราบว่าควรจะเก็บข้อมูลไว้ใน cache ไว้นานเท่าใด สำหรับเครื่องที่มีหมายเลข IP เดียวกันเป็นระยะเวลานานหลายปีก็จะปลอดภัยเมื่อเก็บข้อมูลนี้ไว้ใน cache เป็นเวลาหนึ่งวัน สำหรับข้อมูลอื่นที่มีการเปลี่ยนแปลงบ่อยกว่านี้ก็ควรที่จะลบข้อมูลทิ้งในทุก ๆ นาที เป็นต้น

วิธีการค้นหาข้อมูลที่กล่าวถึงนี้เรียกว่า recursive query เนื่องจากแต่ละเซิบเวอร์ที่ไม่มีข้อมูลที่ถูกถามนั้นจะทำการถามต่อและได้รับข้อมูลตอบกลับมาในภายหลัง ทางเลือกอีกวิธีหนึ่งก็อาจเป็นไปได้นั่นคือ เมื่อ name server ของโซนนั้นไม่สามารถตอบคำถามได้ คำถามนั้นจะถือว่าล้มเหลว (fail) แต่จะส่งค่า name server ตัวต่อไปมาให้แทน ทั้งนี้เนื่องจากเซิบเวอร์บางส่วนไม่สนับสนุนการทำงานแบบ recursive query และมักจะส่งชื่อ name server ตัวต่อไปมาให้แทน

ข้อสังเกตประการหนึ่งคือ ถ้าผู้ใช้ไม่ได้รับการตอบรับจากเซิบเวอร์ก่อนที่จะหมดเวลารอคอย เครื่องผู้ใช้ก็จะพยายามติดต่อกับเซิบเวอร์ตัวถัดไปในครั้งต่อไป เนื่องจากจะตั้งข้อสมมุติฐานว่า เครื่องเซิบเวอร์อาจไม่ทำงาน มากกว่าที่คำถามหรือคำตอบจะหายไปในระหว่างการสื่อสาร

DNS นั้นมีความสำคัญอย่างยิ่งยวดต่อการทำงานของระบบอินเตอร์เน็ตนั่นคือการแปลงชื่อที่เป็นสัญลักษณ์ทางตัวอักษรให้กลายเป็นหมายเลข IP ได้อย่างถูกต้อง DNS ไม่ได้ช่วยตามหาผู้คน ทรัพยากร บริการที่มีให้ หรือวัตถุใด ๆ สำหรับการค้นหาสิ่งเหล่านี้จะเป็นหน้าที่ของบริการที่เรียกว่า LDAP (Light-weight Directory Access Protocol) ซึ่งเป็นบริการขนาดย่อยของโพรโตคอล X.500 directory service ที่อธิบายไว้ในมาตรฐาน RFC 2251 LDAP จัดการรวบรวมข่าวสารไว้ในลักษณะโครงสร้างแบบต้นไม้ ซึ่งยินยอมให้ค้นหาข่าวสารได้ จึงทำหน้าที่คล้ายกับสมุดโทรศัพท์นั่นเอง

7.2 อีเมลล์

อีเมลล์ หรือ Electronic mail (e-mail) เป็นบริการที่ได้รับการนำมาใช้งานมากกว่ายี่สิบปีมาแล้ว ก่อนปี ค.ศ. 1990 อีเมลล์ถูกใช้งานอยู่ในแวดวงการศึกษาเท่านั้น ต่อมาได้รับการเผยแพร่และนำมาใช้งานทั่วไปและมีจำนวนผู้ใช้เพิ่มขึ้นอย่างรวดเร็วจนกระทั่งในปัจจุบันนี้มีจำนวนผู้ใช้อีเมลล์มากกว่าจำนวนผู้ส่งจดหมายธรรมดาเสียอีก

อีเมลล์ก็มีลักษณะคล้ายกับการสื่อสารในรูปแบบอื่นๆ นั่นคือมีข้อตกลงและวิธีการเป็นของตนเอง อีเมลล์เป็นวิธีการสื่อสารที่ไม่เป็นทางการและสามารถใช้งานได้ง่าย ผู้คนที่ไม่เคยฝันถึงการโทรศัพท์หรือแม้กระทั่งการเขียนจดหมายไปหาบุคคลที่มีความสำคัญมาก จะสามารถใช้ประโยชน์จากการเขียนอีเมลล์แบบง่ายๆ ไปหาผู้นั้นได้อย่างง่ายดาย

อีเมลล์นั้นเต็มไปด้วยข้อความและสัญลักษณ์ย่อต่างๆ มากมาย เช่น BTW (By the way) คนจำนวนหนึ่งนิยมใช้สัญลักษณ์ที่เรียกว่า smileys หรือ emoticons ในอีเมลล์ของตน ดังแสดงตัวอย่างในรูป 7-6 รูปเหล่านี้จะมีความหมายมากขึ้นเมื่อเอียงดูด้วยมุม 90 องศา

ระบบอีเมลล์ในรุ่นแรกประกอบด้วยโพรโตคอล FTP (File transfer protocol) โดยมีข้อตกลงว่าบรรทัดแรกของจดหมายอีเมลล์จะเป็นที่อยู่ของผู้รับ เมื่อเวลาผ่านไปข้อจำกัดนี้ได้กลายมาเป็นอุปสรรค ซึ่งสรุปได้ดังนี้

- ไม่สะดวกในการส่งจดหมายไปยังผู้รับเป็นกลุ่ม ผู้บริหารมักจะต้องการใช้การส่งจดหมายเป็นกลุ่มเสมอเพื่อใช้ส่งจดหมายไปยังลูกน้อง
- ตัวจดหมายเองไม่มีโครงสร้างภายใน ทำให้การประมวลผลของเครื่องคอมพิวเตอร์ทำได้ยาก ตัวอย่างเช่น การดึงเอาส่วนของจดหมายที่ต้องการให้ผู้รับส่งต่อไปให้ผู้อื่นอีกทอดหนึ่ง (หรือหลายทอด) นั้นทำได้ยาก
- ผู้ส่งจดหมายจะไม่ทราบว่าจดหมายนั้นไปถึงผู้รับหรือไม่
- ถ้าผู้ใช้วางแผนที่จะไปทำธุระที่อื่นหลายวันหรือหลายสัปดาห์ และต้องการให้เลขานุการของเขารับจดหมายอีเมลล์แทน เป็นสิ่งที่จัดการได้ลำบากยิ่ง
- ส่วนติดต่อผู้ใช้นั้นไม่ได้รับการออกแบบที่ดีทำให้ผู้ใช้ต้องสร้างไฟล์เอกสารขึ้นมาก่อน จากนั้นออกจากโปรแกรมสร้างเอกสารแล้วจึงจัดการส่งจดหมายได้
- ไม่สามารถส่งจดหมายที่มีทั้งข้อความ ภาพวาด แฟกซ์ และไฟล์เสียงไปพร้อมกันได้

Smiley	Meaning	Smiley	Meaning	Smiley	Meaning
:-)	I'm happy	=I:-)	Abe Lincoln	:+)	Big nose
:-(	I'm sad/angry	=):-)	Uncle Sam	:-))	Double chin
:-I	I'm apathetic	*<:-)	Santa Claus	:-{)	Mustache
;-)	I'm winking	<:-(	Dunce	#:-)	Matted hair
:-(O)	I'm yelling	(-:	Australian	8-)	Wears glasses
:-(*)	I'm vomiting	:-)X	Man with bowtie	C:-)	Large brain

รูป 7-6 Smileys

ในปี พ.ศ. 2525 หลังจากที่ได้รับประสบการณ์มากยิ่งขึ้น ARPANET ได้นำเสนออีเมลล์รุ่นใหม่ซึ่งได้ประกาศไว้ในมาตรฐาน RFC 821 และ 822 รวมทั้ง RFC 2821 และ 2822

ในปี พ.ศ. 2527 องค์กร CCITT ได้เขียนร่างสำหรับระบบอีเมลล์มาตรฐาน X.400 ภายหลังจากที่ต่อสู้กันมานานประมาณ 20 ปี ระบบอีเมลล์ก็ได้รับการเห็นพ้องให้ใช้มาตรฐาน RFC 822 และยกเลิกการใช้ X.400 ไปในที่สุด

7.2.1 สถาปัตยกรรมและการให้บริการ

ในหัวข้อนี้จะได้กล่าวถึงเรื่องทั่วไปเกี่ยวกับสิ่งที่ระบบอีเมลล์สามารถทำงานได้และมีการจัดโครงสร้างอย่างไร ระบบอีเมลล์ประกอบด้วยสองส่วนย่อยคือ user agents ซึ่งช่วยให้ผู้ใช้สามารถอ่านและส่งอีเมลล์ได้ และ message transfer agents ซึ่งช่วยในการเคลื่อนย้ายข่าวสารจากแหล่งที่ผลิตข้อมูลไปยังผู้รับ User agent เป็นโปรแกรมที่อาจจะเป็นลักษณะ การเลือกใช้คำสั่งทีละคำสั่ง (command-based) หรือการใช้คำสั่งจากการเลือกหัวข้อที่ปรากฏ (menu-based) หรือใช้กราฟฟิก ในการทำงานร่วมกับระบบอีเมลล์ ส่วน message transfer agent มักจะเป็นโปรแกรมของระบบปฏิบัติการที่เรียกว่า daemons ซึ่งก็คือโปรเซสที่กำลังทำงานอยู่เบื้องหลัง (background process) ที่มีหน้าที่ในการเคลื่อนย้ายอีเมลล์ไปยังเป้าหมาย

โดยทั่วไประบบอีเมลล์ประกอบด้วยการทำงาน 5 อย่าง ดังนี้

Composition หมายถึงโปรเซสสำหรับการสร้างข่าวสารและคำตอบ แม้ว่าจะสามารถใช้โปรแกรมประมวลผลคำใดๆ (text editor) ในการสร้างตัวข้อความ (ที่จะส่งไปกับอีเมลล์) ขึ้นมาก็ตาม ตัวระบบเองจะสามารถให้ความช่วยเหลือในการจัดการกับที่อยู่ (ทั้งผู้ส่งและผู้รับ) และข้อมูลสำหรับเขตข้อมูลต่างๆ ในข้อมูลส่วนหัว (header) ที่ติดไปกับข่าวสารแต่ละชิ้น ตัวอย่างเช่น เมื่อทำการตอบจดหมายอีเมลล์ ระบบอีเมลล์จะจัดการดึงที่อยู่ของผู้ที่ส่งอีเมลล์นั้นมาแล้วใส่เข้าไปในช่องที่อยู่ของผู้รับให้โดยอัตโนมัติ

Transfer หมายถึงการเคลื่อนย้ายข่าวสารจากผู้ผลิตส่งไปยังผู้รับ ในภาพรวมแล้ว การทำงานนี้จะต้องจัดตั้งการเชื่อมต่อจากผู้ส่งไปยังผู้รับหรือเครื่องคอมพิวเตอร์ที่ทำหน้าที่รับข้อมูล จัดการส่งข้อความอีเมลล์ และยกเลิกการติดต่อ ระบบอีเมลล์ควรจะทำงานในขั้นตอนนี้ได้อย่างอัตโนมัติโดยไม่ต้องอาศัยความช่วยเหลือใดๆ จากผู้ใช้เลย

Reporting มีหน้าที่ในการส่งข่าวไปบอกผู้ที่ทำการส่งอีเมลล์ว่าเกิดเหตุการณ์ใดขึ้นกับข้อความที่ได้ส่งไปแล้ว นั่นคือ ได้ไปถึงผู้รับ หรือถูกส่งย้อนกลับมา หรือสูญหายไป มีงานประยุกต์มากมายที่ต้องการทราบสถานะภาพของการส่งอีเมลล์ งานบางอย่างก็อาจมีความเกี่ยวพันทางกฎหมายด้วย

Displaying ผู้ใช้มีความจำเป็นที่จะต้องทราบว่ามีข่าวสารถูกส่งเข้ามาจึงจะสามารถอ่านอีเมลล์เหล่านั้นได้บางครั้งการเปลี่ยนรูปแบบของข้อมูลที่ส่งมาก็มีความจำเป็นซึ่งอาจจะต้องใช้โปรแกรมพิเศษในการดูข่าวสารนั้น เช่นในกรณีที่ข่าวสารอยู่ในรูปของ postscript file หรือ digitized voice เป็นต้น โดยทั่วไปจะมีการเปลี่ยนแปลงรูปแบบของข่าวสารเกิดขึ้นเล็กน้อยเสมอ

Disposition ขั้นตอนสุดท้ายเกี่ยวข้องกับว่าผู้ใช้จะทำอย่างไรกับข่าวสารที่ได้รับเรียบร้อยแล้ว สิ่งที่อาจเป็นไปได้ก็คือ ลบข้อความนั้นทิ้งไปก่อนที่จะอ่าน ลบข้อความนั้นทิ้งหลังจากที่ได้อ่านแล้ว จัดการ

บันทึกข้อความนั้นไว้ในแฟ้มข้อมูล และอื่นๆ บางครั้งอาจมีความจำเป็นที่จะต้องการอ่านข้อความที่ได้อ่านไปแล้วและได้บันทึกไว้ทำการส่งข้อความนั้นต่อไปยังผู้อื่นหรือจัดการข้อความนั้นไปในทางใดๆ ก็ตาม

นอกเหนือจากบริการพื้นฐานที่กล่าวถึงนี้ ระบบอีเมลล์บางระบบโดยเฉพาะระบบที่มีใช้งานภายในองค์กร จะมีขีดความสามารถอื่นๆ เพิ่มเติมเข้ามาอีก เช่น เมื่อผู้ใช้ไม่อยู่เป็นเวลาหลายวันผู้ใช้จะสามารถให้ระบบอีเมลล์จัดการส่งต่ออีเมลล์ที่เข้ามาทั้งหมดไปยังผู้อื่น (เช่น เพื่อน หรือเลขานุการ) ให้ได้โดยอัตโนมัติ

ระบบอีเมลล์ส่วนใหญ่จะอนุญาตให้ผู้ใช้สามารถสร้าง mailbox ขึ้นมาได้เพื่อใช้ในการจัดเก็บอีเมลล์ที่ส่งเข้ามา ในการนี้จะมีคำสั่งชุดพิเศษเพิ่มเติมเข้ามาเพื่อจัดการกับ mailbox เช่น การสร้างขึ้นมาใช้งาน การตรวจสอบข้อความที่เก็บอยู่ภายใน การใส่และลบข้อความที่เก็บอยู่ และอื่นๆ

ผู้บริหารองค์กรมักจะต้องการส่งข่าวสารไปยังลูกน้องแต่ละคน หรือลูกค้า หรือบริษัทผู้ผลิตสินค้าต่างๆ ในคราวเดียวกันหลายคนอันเป็นที่มาของการจัดทำ mailing list ซึ่งก็คือรายการที่อยู่อีเมลล์กลุ่มหนึ่ง ข้อความที่ซ้ำกันจะถูกส่งไปยังผู้รับแต่ละคนที่มีรายชื่ออยู่ใน mailing list

ขีดความสามารถอื่นๆ เช่น carbon copies, blind carbon copies, high-priority e-mail, secret e-mail (เช่นการเข้ารหัส) ผู้รับสำรองในกรณีที่ผู้รับหลักไม่สามารถรับข้อความได้ และความสามารถพิเศษในการอ่านอีเมลล์ของเจ้านายได้ เป็นต้น

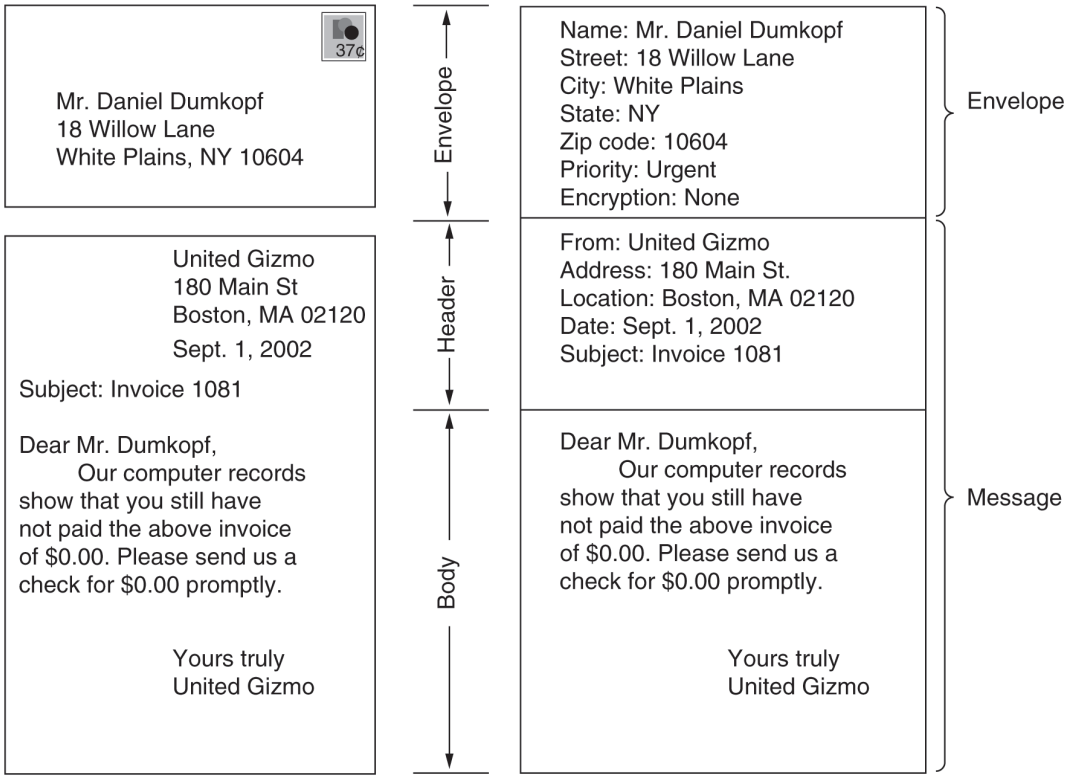
ในปัจจุบันระบบอีเมลล์ได้รับการนำมาใช้งานอย่างกว้างขวางสำหรับการติดต่อสื่อสารระหว่างองค์กร ระบบอีเมลล์ช่วยให้พนักงานที่อยู่ไกลออกไปมากสามารถร่วมทำงานในโปรเจ็คขององค์กรได้ ด้วยการตัดสิ่งที่เป็นอุปสรรคต่างๆ เช่น ตำแหน่ง อายุ และเพศออกไป การส่งข้อความผ่านระบบอีเมลล์จึงเน้นไปในเรื่องการนำเสนอแนวความคิดไม่ใช่สถานะในองค์กร ด้วยระบบอีเมลล์ ความคิดอันเยี่ยมยอดของนักศึกษาที่กำลังพักผ่อนในช่วงฤดูร้อนจะสามารถสร้างผลกระทบอันยิ่งใหญ่ได้มากกว่าความคิดที่ไม่เข้าท่าที่อาจส่งมาจากผู้บริหารบางคน

แนวความคิดหลักของระบบอีเมลล์ คือการแยกความแตกต่างระหว่างการจ่าหน้าซอง (envelop) กับข้อความที่อยู่ภายใน การจ่าหน้าซองเป็นการห่อหุ้มข้อความ ซึ่งจะมีข้อมูลที่จำเป็นทั้งหมดที่ต้องใช้ในการส่งข้อความนั้น เช่น ที่อยู่ของผู้รับ ระดับความสำคัญ และระดับการรักษาความลับ ซึ่งทั้งหมดนี้ไม่ใช่ส่วนหนึ่งของข้อความภายในเลย Message transport agent ใช้ envelop ในการจัดส่งข้อความเหมือนกับที่ไปรษณีย์ใช้

ข้อความที่อยู่ภายใน envelop ประกอบด้วยข้อมูลสองส่วนคือ ข้อมูลส่วนหัว (header) และตัวข้อความ (body) ข้อมูลส่วนหัวจะบรรจุข้อมูลสำหรับการควบคุมที่จะถูกนำไปใช้โดย user agent ตัวข้อความเป็นข้อความสำหรับการสื่อสารระหว่างมนุษย์ รูป 7-7 แสดงโครงสร้างของ envelop และข้อความ

7.2.2 The User Agent

ระบบอีเมลล์แบ่งออกเป็นสองส่วนคือ user agents และ message transfer agent ในหัวข้อนี้จะกล่าวถึง user agent โดยปกติแล้ว user agent เป็นโปรแกรมหนึ่ง (บางครั้งเรียกว่า mail reader) ที่ยอมรับคำสั่งจำนวนมากสำหรับการ ประกอบอีเมลล์ (composing) การรับ (receiving) และ



รูป 7-7
Envelop และ ข้อความ
(a) จดหมายทั่วไป
(b) อีเมลล์

การตอบกลับ (replying) ข้อความรวมถึงการจัดการ mailbox ด้วย User agent บางส่วนมีการสร้างส่วนติดต่อผู้ใช้เป็นอย่างดีที่เป็นกราฟฟิก ในขณะที่ยังมีอีกส่วนหนึ่งที่ใช้วิธีการออกคำสั่งแบบง่ายๆ อยู่เหมือนเดิม ในแง่มุมของการทำงานแล้ว ทั้งสองแบบนี้สามารถทำงานได้เหมือนกัน

การส่งอีเมลล์

ในการส่งข้อความผ่านระบบอีเมลล์ ผู้ใช้จะต้องมีข้อความที่จะส่ง ที่อยู่ของผู้รับ และอาจจะต้องมีค่าสำหรับพารามิเตอร์บางตัว (ขึ้นอยู่กับโปรแกรมที่เลือกใช้) ข้อความอาจถูกสร้างขึ้นมาด้วยโปรแกรมประมวลผลคำง่ายๆ ที่เรียกว่า text editor หรือโปรแกรมที่มีความซับซ้อนเช่น Word processing หรืออาจจะเป็นโปรแกรมที่สร้างมาพร้อมกับ user agent ก็ได้ ที่อยู่ของผู้รับจะต้องเขียนในรูปแบบที่ user agent รู้จัก เช่น user@dns-address

อย่างไรก็ตาม เป็นที่น่าสังเกตว่ารูปแบบที่อยู่ผู้รับแบบอื่นๆ ก็มีอยู่ เช่นในระบบ X.400 จะมีลักษณะดังนี้

/C=US/ST=MASSACHUSETTS/L=CAMBRIDGE/PA=360 MEMORIAL DR./CN=KEN SKITH/

ที่อยู่นี้กำหนดชื่อประเทศ ชื่อรัฐ ชื่อหน่วยงาน ที่อยู่ไปรษณีย์ และชื่อผู้ใช้ ในรูปแบบการกำหนดที่อยู่อาจมีการใส่คุณสมบัติต่างๆ เข้าไปได้เพื่อให้ผู้ใช้สามารถส่งอีเมลล์ไปยังบุคคลที่ไม่รู้จักที่อยู่บนอีเมลล์อย่างแน่ชัดได้ในเมื่อมีคุณสมบัติอื่นชัดเจน (เช่นตำแหน่งในบริษัท) แม้ว่าชื่อที่อยู่ในระบบ X.400 จะไม่สะดวกในการใช้งานเท่าในระบบ DNS แต่ระบบอีเมลล์ส่วนใหญ่จะมีชื่อรอง (alias) หรือชื่อเล่น (nickname) ที่ช่วยให้ผู้ใช้มีความสะดวกในการอ้างถึงชื่อผู้ใช้ในรูปแบบที่ง่ายกว่าและสามารถอ้างอิงถึงชื่ออีเมลล์ที่ถูกต้องได้

ระบบอีเมลล์ส่วนใหญ่สนับสนุนการใช้ mailing list ทำให้ผู้ใช้สามารถส่งข้อความเดียวกันไปยังรายชื่อของผู้รับกลุ่มหนึ่งด้วยคำสั่งเพียงคำสั่งเดียว ถ้า mailing list ได้รับการดูแลจากเครื่องของผู้ใช้เอง user agent ก็เพียงแค่ส่งข้อความแยกต่างหากไปยังผู้รับแต่ละคน อย่างไรก็ตาม ถ้า mailing list ได้รับการดูแลโดยคอมพิวเตอร์เครื่องอื่นแล้ว user agent จะส่งข้อความออกไปเพียงข้อความเดียวไปยังผู้ดูแลนั้น และปล่อยให้เป็นหน้าที่ของผู้ดูแลที่จะจัดการส่งข้อความนั้นไปยังผู้รับแต่ละคนในรายชื่อกลุ่มนั้น เช่น รายชื่อของนักดูนกถูกรวบรวมไว้ใน mailing list ชื่อ birders ซึ่งติดตั้งไว้ที่เครื่อง meadowlark. arizona.edu ดังนั้นข้อความใดก็ตามที่ถูกส่งไปยัง birders@meadowlark.arizona.edu จะถูกส่งต่อไปที่ University of Arizona และจัดการสร้างสำเนาข้อความขึ้นมาเท่ากับจำนวนชื่อผู้รับที่อยู่ในรายการ birders แล้วส่งไปยังผู้รับแต่ละคน

การอ่านอีเมลล์

โดยทั่วไปเมื่อ user agent เริ่มทำงาน มันจะตรวจดูข้อมูลที่อยู่ใน mailbox ของผู้ใช้เพื่อหาอีเมลล์ใหม่ที่เพิ่งจะถูกส่งเข้ามาก่อนที่จะแสดงข้อความใดๆ ขึ้นบนหน้าจอภาพ หลังจากนั้นก็จะประกาศจำนวนข้อความที่มีอยู่ใน mailbox หรือแสดงข้อความสรุปของแต่ละอีเมลล์ที่รับเข้ามาและรอรับคำสั่งต่อไป

ให้ดูตัวอย่างว่า user agent ทำงานอย่างไรโดยพิจารณาจากสถานะการณ์ทั่วไปที่อาจเกิดขึ้นจริง หลังจากที่เริ่มต้นใช้งาน user agent ผู้ใช้จะขอดูรายการสรุปดังที่แสดงตัวอย่างในรูป 7-8 แต่ละบรรทัดหมายถึงอีเมลล์หนึ่งฉบับ ในตัวอย่างนี้มีอีเมลล์อยู่ 8 ฉบับ

ข้อมูลในแต่ละบรรทัดประกอบด้วยหลายเขตข้อมูลที่ดึงออกมาจาก envelop หรือ ข้อมูลส่วนหัว (header) ของอีเมลล์แต่ละฉบับ ในระบบอีเมลล์แบบง่ายๆ ข้อมูลที่แสดงให้เห็นนั้นถูกกำหนดมาโดยโปรแกรมนั้นไว้แล้ว แต่ในระบบอีเมลล์ที่มีความซับซ้อนมากกว่าจะอนุญาตให้ผู้ใช้จัดการเลือกชนิดของข้อมูลที่ต้องการดูได้เอง โดยการกำหนดไว้ใน user profile ซึ่งเป็นไฟล์ที่ใช้ในการอธิบายรูปแบบการแสดงผลข้อมูลบนจอภาพ ในตัวอย่างที่ยกมานี้ เขตข้อมูลแรกคือหมายเลขของอีเมลล์ เขตข้อมูลที่สอง "Flags" อาจเป็น "K" ซึ่งหมายถึงไม่ใช่ข้อความใหม่แต่ถูกอ่านแล้วและเก็บไว้ใน mailbox "A" หมายถึงข้อความนั้นได้ถูกตอบกลับไปแล้ว และ "F" หมายถึงข้อความนั้นได้ถูกส่งต่อไปยังผู้อื่นแล้ว เป็นต้น

#	Flags	Bytes	Sender	Subject
1	K	1030	asw	Changes to MINIX
2	KA	6348	trudy	Not all Trudys are nasty
3	K F	4519	Amy N. Wong	Request for information
4		1236	bal	Bioinformatics
5		104110	kaashoek	Material on peer-to-peer
6		1223	Frank	Re: Will you review a grant proposal
7		3110	guido	Our paper has been accepted
8		1204	dmr	Re: My student's visit

รูป 7-8 ตัวอย่างการแสดงข้อมูลเกี่ยวกับ mailbox

เขตข้อมูลที่สาม "Bytes" ใช้บอกขนาดของข้อความ และเขตข้อมูลที่สี่ "Sender" บอกให้ทราบว่าใครเป็นผู้ส่งข้อความนั้นมา เนื่องจากเขตข้อมูลนี้ถูกดึงออกมาจากข้อความที่ส่งมา จึงสามารถแสดงชื่อต้น ชื่อเต็ม ชื่อย่อ ชื่อในระบบคอมพิวเตอร์ หรือชื่ออะไรก็ได้ที่ผู้ส่งเลือกที่จะใช้ ท้ายที่สุด เขตข้อมูล "Subject" ใช้บอกประเภทของข้อความหรือข้อความย่อที่บอกให้ทราบว่าข้อความเต็มนั้นเป็นเรื่องเกี่ยวกับอะไร โดยปกติผู้ใช้ที่ไม่ใส่ข้อมูลมาใน "Subject" มักจะไม่ค่อยได้รับความสนใจจากผู้รับมากนัก

หลังจากที่ทำการแสดงข้อมูลนี้แล้ว ผู้ใช้อาจเลือกที่จะทำงานได้หลายอย่าง เช่น แสดงข้อความที่ต้องการ ลบข้อความทิ้ง และอื่นๆ ระบบอีเมลล์แบบเก่ามักจะเป็นระบบที่แสดงตัวอักษรเพียงอย่างเดียว (text-based) ซึ่งมักจะใช้คำสั่งเป็นตัวอักษรเพียงตัวเดียว เช่น "T" (พิมพ์ข้อความ) "A" (ตอบข้อความ) "D" (ลบข้อความ) และ "F" (จัดส่งข้อความต่อไปยังผู้อื่น) ระบบอีเมลล์สมัยใหม่จะเป็นการแสดงด้วยกราฟฟิก ผู้ใช้จะเลือกข้อความที่ต้องการโดยใช้เมาส์และ "คลิ๊ก" ไปที่ไอคอนที่ต้องการในการพิมพ์ การตอบ การลบ หรือการจัดส่งข้อความได้ตามต้องการ

7.2.3 รูปแบบของข้อความ

ในหัวข้อนี้จะได้กล่าวถึงรูปแบบของข้อความที่อยู่ในอีเมลล์ ในรูปแบบแรกคือ ตัวอักษร ASCII ตามมาตรฐาน RFC 822 จากนั้นจะดูส่วนที่เป็นกราฟฟิกซึ่งกล่าวรายละเอียดไว้ใน RFC 822 ส่วนขยาย

RFC 822

ข้อความในอีเมลล์ประกอบด้วย envelop ข้อมูลส่วนหัว (header) บรรทัดว่าง และข้อความจดหมาย แต่ละเขตข้อมูลในข้อมูลส่วนหัวประกอบด้วยข้อมูลที่เขียนอยู่ในบรรทัดเดียวกันที่อยู่ในรูปแบบตัวอักษร ASCII โดยเริ่มต้นด้วยชื่อเขตข้อมูล เครื่องหมายโคลอน (:) และค่าสำหรับเขตข้อมูลนั้น มาตรฐาน RFC 822 ได้รับการออกแบบมานานหลายสิบปีมาแล้วจึงไม่ได้กำหนดความแตกต่างที่ชัดเจนระหว่างเขตข้อมูลสำหรับ envelop ออกจากเขตข้อมูลของข้อมูลส่วนหัว แม้ว่าจะได้รับการปรับปรุงใหม่ตามมาตรฐาน RFC 2822 ก็ไม่สามารถเปลี่ยนแปลงรูปแบบเดิมโดยสิ้นเชิงได้เนื่องจากเป็นรูปแบบที่มีการใช้งานอย่างกว้างขวางมากแล้ว ในการใช้งานปกติ user agent จะสร้างข้อความและส่งต่อให้กับ message transfer agent ซึ่งจะใช้ข้อมูลบางส่วนจากข้อมูลส่วนหัวในการสร้าง envelop ขึ้นมาอีกต่อหนึ่ง จึงเป็นรูปแบบที่ผสมระหว่างข้อความและ envelop

เขตข้อมูลส่วนหัวที่ทำหน้าที่ขั้นพื้นฐานที่เกี่ยวข้องกับ message transport agent แสดงในรูปที่ 7-9 เขตข้อมูล "To:" เป็นที่อยู่ DNS ของผู้รับหลัก อีเมลล์อาจจะถูกส่งต่อไปยังผู้รับหลายคนได้ เขตข้อมูล "Cc:" เป็นที่อยู่ของผู้รับรอง ในการนำส่งอีเมลล์นั้นไม่มีการแยกความแตกต่างระหว่างผู้รับหลักและผู้รับรอง มันเป็นความแตกต่างทางด้านจิตวิทยาซึ่งอาจจะมีความสำคัญต่อคนที่ใช้งานแต่ไม่ใช่สำหรับระบบอีเมลล์ คำว่า "Cc:" ย่อมาจากคำว่า Carbon copy นั้นเป็นคำเก่าเนื่องจากคอมพิวเตอร์ไม่ใช้กระดาษ carbon copy แต่คำนี้ก็ได้รับการนำมาใช้จนได้ เขตข้อมูล "Bcc:" หรือ Blind carbon copy นั้นมีความคล้ายคลึงกับ "Cc:" ยกเว้นสิ่งที่อยู่บนบรรทัดนี้จะถูกลบทิ้งไปเมื่อนำส่งอีเมลล์ไปยังผู้รับหลักและผู้รับรอง คุณสมบัติข้อนี้ช่วยให้ผู้ใช้สามารถส่งข้อความเดียวกันไปยังบุคคลที่สามได้โดยไม่ให้ผู้รับหลักและผู้รับรองทราบ

รูป 7-9
เขตข้อมูลต่างๆ ใน ข้อมูลส่วนหัวตาม มาตรฐาน RFC 822 ในส่วนที่เกี่ยวข้องกับ message transport

Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

เขตข้อมูลสองเขตต่อมาคือ "From:" และ "Sender:" บอกให้ทราบว่าใครคือผู้เขียนข้อความนั้น และใครคือผู้ส่ง ตามลำดับ บุคคลทั้งสองคนนี้ไม่จำเป็นจะต้องเป็นคนคนเดียวกัน ตัวอย่างเช่น ผู้บริหารอาจเป็นผู้เขียนข้อความขึ้นมาและให้เลขานุการของเขาเป็นผู้ส่งข้อความนั้นก็ได้ ในกรณีนี้ ชื่อผู้บริหารจะอยู่ในตำแหน่ง "From:" และชื่อของเลขานุการจะอยู่ในตำแหน่ง "Sender:" ในกรณีที่ผู้เขียนข้อความเป็นคนคนเดียวกับผู้ส่งข้อความก็ไม่จำเป็นต้องใช้เขตข้อมูล "Sender:" เขตข้อมูลทั้งสองนี้มีความจำเป็นต้องใช้ในกรณีที่ไม่สามารถส่งข้อความนี้ไปยังผู้รับได้และจำเป็นจะต้องส่งข้อความนี้กลับคืนมาที่ผู้ส่ง

บรรทัดต่อมาคือ "Received:" ข้อมูลจะถูกใส่เข้ามาโดย message transfer agent ในระหว่างทางที่นำส่ง บรรทัดนี้จะบรรจุเอกลักษณ์ของ agent วันและเวลาที่ข้อความนี้ถูกนำส่งไปถึงผู้รับ และข้อมูลอื่นๆ ที่สามารถนำมาใช้ค้นหาข้อผิดพลาดที่เกิดขึ้นในระหว่างการนำส่ง

เขตข้อมูล "Return-Path:" ถูกเติมข้อมูลเข้ามาโดย message agent ตัวสุดท้ายและถูกออกแบบมาเพื่อจะได้ส่งข้อความย้อนกลับไปหาผู้ส่งได้ ในทางทฤษฎี ข้อมูลนี้สามารถรวบรวมได้จาก "Received:" ทั้งหมดในข้อมูลส่วนหัว (ยกเว้นชื่อของ mailbox ของผู้ส่ง) แต่โดยทั่วไปก็มักจะถูกเติมเข้ามาตามลำดับและมักจะมีที่อยู่ของผู้ส่งอยู่ในนี้ด้วย

นอกเหนือจากเขตข้อมูลที่แสดงในรูป 7-9 แล้ว มาตรฐาน RFC 822 ยังมีเขตข้อมูลอื่นๆ อีกมาก ซึ่งสามารถนำไปใช้งานได้โดย agent ของผู้ใช้และคนที่รับข้อความนั้น เขตข้อมูลที่ใช้กันทั่วไปแสดงในรูป 7-10 เขตข้อมูลส่วนใหญ่สามารถทราบความหมายได้จากชื่อของเขตข้อมูลนั้น

เขตข้อมูล "Reply-To:" ในบางครั้งถูกนำมาใช้เมื่อผู้ที่สร้างข้อความขึ้นมาและผู้ที่ส่งข้อความออกไปไม่ต้องการที่จะเห็นคำตอบรับจากการส่งข้อความนี้ ตัวอย่างเช่น ผู้จัดการด้านการตลาดเขียนอีเมลล์ไปบอกลูกค้าเกี่ยวกับผลิตภัณฑ์ตัวใหม่ ข้อความนี้ถูกส่งออกไปโดยเลขานุการของเขา แต่ "Reply-To:" เป็นชื่อของหัวหน้าฝ่ายขายซึ่งเป็นผู้ที่จะสามารถตอบคำถามแก่ลูกค้าและสามารถรับคำสั่งซื้อจากลูกค้าได้ เขตข้อมูลนี้ก็มีประโยชน์เมื่อผู้ส่งมีที่อยู่อีเมลล์สองบัญชีที่ต้องการใช้สำหรับการส่งหนึ่งบัญชีและสำหรับการรับอีกหนึ่งบัญชี

เอกสารประกอบมาตรฐาน RFC 822 ระบุไว้อย่างชัดเจนว่าผู้ใช้ได้รับอนุญาตให้สร้างข้อมูลส่วนหัวขึ้นมาใหม่ได้เพื่อนำมาใช้เป็นการส่วนตัว โดยมีเงื่อนไขว่าข้อมูลเหล่านี้จะต้องเริ่มต้นด้วย "X-" เพื่อหลีกเลี่ยงข้อขัดแย้งระหว่างเขตข้อมูลที่เป็นทางการกับที่ใช้งานส่วนตัว

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	E-mail address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

รูป 7-10
เขตข้อมูลบางส่วน
ที่ใช้ใน RFC 822

หลังจากข้อมูลส่วนหัวก็จะเป็นตัวข้อความ (body) ผู้ใช้สามารถใส่ข้อความใด ๆ ก็ได้ลงในส่วนนี้

MIME-The Multipurpose Internet Mail Extensions

ระบบอีเมลล์ที่ได้รับการคิดค้นขึ้นมานานแล้วนั้นไม่สามารถตอบสนองความต้องการในปัจจุบันได้ เช่น

1. ข้อความที่เขียนด้วยภาษาอื่นนอกจากภาษาอังกฤษ

2. ข้อความที่ใช้ตัวอักษรแตกต่างออกไป

3. ข้อความในภาษาที่ไม่ใช้ตัวอักษร

4. ข้อความที่ไม่มีตัวอักษรอยู่เลย

หนทางแก้ไขได้รับการกำหนดไว้ในมาตรฐาน RFC 1341 และปรับปรุงใน RFC 2045-2049 วิธีการแก้ปัญหานี้เรียกว่า MIME (Multipurpose Internet Mail Extensions) ซึ่งได้รับการนำมาใช้งานอย่างกว้างขวางในปัจจุบัน

แนวความคิดพื้นฐานของ MIME คือการที่ยังคงใช้มาตรฐาน RFC 822 แต่ว่าได้เพิ่มเติมโครงสร้างในส่วนตัวข้อความและกำหนดวิธีการเข้ารหัสข้อมูลสำหรับข้อความที่ไม่ใช้รหัส ASCII แต่ก็ไม่ได้แยกตัวออกไปจาก RFC 822 นั่นคือข้อความใน MIME สามารถถูกส่งได้โดยใช้โปรแกรมและโพรโตคอลที่มีใช้อยู่ในปัจจุบัน สิ่งที่ต้องได้รับการแก้ไขคือโปรแกรมสำหรับส่งและสำหรับรับข้อความ ซึ่งผู้ใช้สามารถทำได้เอง

MIME ได้กำหนดเขตข้อมูลสำหรับข้อมูลส่วนหัวขึ้นมาใหม่ 5 เขตข้อมูลดังแสดงในรูป 7-11 เขตข้อมูลแรกบอกให้ user agent ที่รับข้อความนี้ทราบว่ากำลังรับข้อความแบบ MIME และเป็นรุ่นที่เท่าใด ข้อความที่ไม่มี "MIME-Version:" นั้นจะถือว่าเป็นข้อความภาษาอังกฤษล้วน (แบบเก่า)

เขตข้อมูล "Content-Description:" เป็นข้อมูลประเภทตัวอักษร (ASCII) ที่บอกให้ทราบว่าข้อมูลใดอยู่ในข้อความที่ส่งมา เขตข้อมูลนี้มีความจำเป็นเพื่อบอกให้ผู้รับทราบว่าข้อความที่ส่งมาเป็นเรื่องอะไรมีความสำคัญเพียงใดที่จะต้องอ่านข้อความนั้น เช่น ถ้าเป็นข้อความที่บอกว่า "Photo of Bird-Thongchai" ถ้าผู้รับไม่ใช่แฟนเพลงของเบิร์ด-ธงไชยก็จะไม่ให้ความสนใจกับข้อความ (รูป) ที่ส่งมานี้

เขตข้อมูล "Content-Id:" ใช้ในการระบุตัวตนของสิ่งที่ส่งมา เป็นข้อมูลประเภทเดียวกับ "Message-Id:" ซึ่งอาจจะเป็นหมายเลขเฉพาะอย่างหนึ่ง

รูป 7-11
ข้อมูลส่วนหัวที่เพิ่มเติมเขตข้อมูลของ MIME เข้าไป

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

เขตข้อมูล "Content-Transfer-Encoding:"บอกให้ทราบว่าตัวข้อความนั้นถูกห่อหุ้มเพื่อการจัดส่งผ่านระบบเครือข่ายอย่างไร ซึ่งอาจจะเป็นสัญลักษณ์อื่นนอกเหนือจากตัวอักษร ตัวเลข และเครื่องหมายบางอย่าง MIME ได้นำเสนอวิธีการไว้ 5 แบบ แบบที่นิยมมากที่สุดคือ ASCII text ซึ่งเป็นตัวอักษรมาตรฐานซึ่งเข้ารหัสด้วยข้อมูลขนาด 7 บิตและสามารถที่จะใช้งานร่วมกับโพรโตคอลอีเมลล์ได้โดยตรง จะมีขนาดไม่เกิน 1000 ตัวอักษร

แบบที่สองเป็น ASCII เหมือนกันแต่เข้ารหัสด้วยข้อมูลขนาด 8 บิต ทำให้มีรูปร่างของตัวอักษรได้ถึง 256 แบบ การเข้ารหัสแบบนี้ได้ขัดแย้งต่ออินเตอร์เน็ตอีเมลล์โพรโตคอล (รุ่นดั้งเดิม) ที่ยังคงมีใช้งานอยู่ในบางส่วนในระบบอินเตอร์เน็ต

แบบที่สามเป็นการส่งข้อความที่เข้ารหัสด้วยภาษาไบนารี่ (binary) ไฟล์ประเภทนี้จะไม่มีขอบเขตอยู่ที่ 1000 ตัวอักษร ตัวอย่างได้แก่ โปรแกรมที่สามารถประมวลผลได้ (executable program) ไม่มีการรับประกันว่าไฟล์ประเภทนี้ที่ถูกส่งมากับข้อความจะมาถึงผู้รับอย่างถูกต้อง

วิธีการที่ถูกต้องในการส่งข้อมูลประเภทไบนารี่ คือการเข้ารหัสข้อมูลดังกล่าวด้วยวิธี Base64 encoding บางครั้งเรียกว่า ASCII armor ด้วยวิธีการนี้จะทำการแบ่งข้อมูลออกเป็นกลุ่ม กลุ่มละ 24 บิต ข้อมูลแต่ละกลุ่มจะถูกแบ่งย่อยออกเป็นกลุ่มย่อยละ 6 บิตซึ่งจะเขียนแทนด้วยตัวอักษร ASCII หนึ่งตัว การเข้ารหัสนั้นจะกำหนดให้ "A" แทน "0" และ "B" แทน "1" ไปจนครบ 26 ตัวตามด้วยตัวอักษรตัวเล็กอีก 26 ตัว ตามด้วยตัวเลขอีก 10 ตัว และแทน "62" และ "63" ด้วยเครื่องหมาย "+" และ "/ " ตามลำดับ รวมกันเป็นทั้งหมด 64 ตัว เครื่องหมาย "==" และ "=" ใช้แทนความหมายว่าข้อมูลในกลุ่มสุดท้ายมีขนาดเพียง 8 และ 16 บิตตามลำดับ ตัวอักษร carriage return และ line feed นั้นจะถูกละเลยจึงสามารถใส่แทรกเข้าไปในตำแหน่งใดก็ได้เพื่อทำให้แต่ละบรรทัดนั้นสั้นลง การส่งข้อมูลไบนารี่จึงสามารถส่งได้อย่างปลอดภัยด้วยวิธีการนี้

สำหรับข้อความที่ประกอบด้วยตัวอักษร ASCII เกือบทั้งหมดแต่มีข้อมูลบางตัวที่ไม่ใช่รหัส ASCII การใช้รหัส Base64 encoding นั้นจะมีประสิทธิภาพต่ำมาก จึงเปลี่ยนไปใช้การเข้ารหัสอีกวิธีหนึ่งเรียกว่า "quoted-printable encoding" นั่นคือใช้รหัส ASCII ขนาด 7 บิตและตัวอักษรทั้งหมดที่มีรหัสสูงกว่า 127 ขึ้นไปด้วยเครื่องหมาย "=" ตามด้วยหมายเลขของตัวอักษรนั้นที่เขียนเป็นเลขฐานสิบหกขนาดสองหลัก

โดยสรุปแล้ว ข้อมูลไบนารี่ควรถูกส่งโดยการใช้วิธีเข้ารหัสแบบ base64 หรือ quoted-printable แต่เมื่อมีเหตุผลที่ดีพอที่จะไม่ใช้การเข้ารหัสแบบที่กล่าวถึงนี้ ก็มีความเป็นไปได้ที่จะใช้วิธีการเข้ารหัสแบบที่ผู้ใช้เป็นผู้กำหนดเองซึ่งจะถูกระบุไว้ใน "Content-Transfer-Encoding:"

ข้อมูลส่วนสุดท้ายที่แสดงในรูป 7-11 เป็นสิ่งที่น่าสนใจมากที่สุดเพราะใช้เป็นตัวกำหนดชนิดของตัวข้อความ ซึ่งมาตรฐาน RFC 2045 ได้กำหนดไว้เป็น 7 ชนิดโดยที่แต่ละชนิดอาจแบ่งเป็นชนิดย่อยได้อีกจำนวนหนึ่งชนิด และชนิดย่อยนั้นอธิบายแยกกันด้วยเครื่องหมาย "/" ดังเช่น

Content-Type: video/mpeg

ชนิดย่อยนั้นจะต้องถูกระบุไว้อย่างชัดเจนในข้อมูลส่วนหัวเพราะไม่มีการกำหนดค่า default เอาไว้ให้ รูป 7-12 แสดงรายการชื่อชนิดและชนิดย่อยที่สามารถนำมาใช้งานได้ และยังมีชนิดและชนิดย่อยใหม่ๆ อีกจำนวนหนึ่งที่ได้รับการเพิ่มเติมเข้าไปและสามารถที่จะเพิ่มเติมได้อีกเมื่อมีความจำเป็น

ข้อความชนิด "text" หมายความว่าข้อความที่ส่งมานั้นจะเป็นตัวอักษร ASCII text ชนิดย่อย "text/plain" ใช้สำหรับบอกชนิดของข้อความปกติทั่วไปที่สามารถจะนำไปแสดงผลบนหน้าจอภาพได้ในทันทีที่ได้รับโดยไม่มีวิธีการเข้ารหัสข้อมูลใดๆ ที่จะต้องทำการถอดรหัสก่อน ตัวเลือกนี้ช่วยให้สามารถส่งข้อความที่เป็นตัวหนังสือผ่าน MIME ได้โดยใช้ข้อมูลส่วนหัวเพียงเล็กน้อย

"text/enriched" ใช้ในการสร้างภาษาประเภท markup language แบบง่ายๆ (คล้ายกับภาษา HTML แต่มีขีดความสามารถน้อยกว่ามาก) ภาษาประเภทนี้จะสนับสนุนวิธีการสร้างการแสดงผลที่แตกต่างไปจากการแสดงผลตัวอักษรธรรมดา เช่น ตัวอักษรหนา ตัวอักษรเอียง หรือการขีดเส้นใต้ใต้ข้อความที่ต้องการ ภาษาประเภทนี้พัฒนาขึ้นมาโดยอาศัยมาตรฐานจาก SGML (Standard Generalized Markup Language) ซึ่งเป็นมาตรฐานเดียวกันกับภาษา HTML ตัวอย่างเช่น

The <bold> time </bold> has come the <italic> walrus </italic> said ...

เมื่อแสดงผลจะเห็นเป็นดังนี้

The ***time*** has come the *walrus* said ...

Type	Subtype	Description
Text	Plain	Unformatted text
	Enriched	Text including simple formatting commands
Image	Gif	Still picture in GIF format
	Jpeg	Still picture in JPEG format
Audio	Basic	Audible sound
Video	Mpeg	Movie in MPEG format
Application	Octet-stream	An uninterpreted byte sequence
	Postscript	A printable document in PostScript
Message	Rfc822	A MIME RFC 822 message
	Partial	Message has been split for transmission
	External-body	Message itself must be fetched over the net
Multipart	Mixed	Independent parts in the specified order
	Alternative	Same message in different formats
	Parallel	Parts must be viewed simultaneously
	Digest	Each part is a complete RFC 822 message

รูป 7-12
ชนิดและชนิดย่อยของข้อความใน MIME

อย่างไรก็ตาม ขึ้นอยู่กับเครื่องของผู้รับที่จะแสดงผลให้ออกมาในลักษณะอย่างใด ถ้าสนับสนุนการแสดงตัวหนา (bold) และตัวเอียง (italic) ข้อความก็จะปรากฏตามที่แสดงตัวอย่างให้เห็น แต่ถ้าเครื่องผู้รับไม่สนับสนุนการแสดงผลทั้งสองแบบนี้ ผลที่ออกมาก็จะเป็นตัวอักษรตามปกติ คุณสมบัติอื่นๆ ได้แก่ ตัวสีต่างๆ ตัวกระพริบ ตัวอักษรขีดเส้นใต้ เป็นต้น

เมื่อเว็บได้เริ่มเป็นที่นิยมใช้งาน ชนิดย่อยชนิดใหม่ได้เกิดขึ้นนั่นคือ "text/html" (RFC 2854) ทำให้สามารถส่งเว็บเพจเข้ามาเป็นส่วนของข้อความในอีเมลล์ได้ และชนิดย่อยใหม่อีกชนิดหนึ่งคือ "text/xml" ซึ่งใช้สำหรับภาษา extensible markup language ซึ่งได้กำหนดมาตรฐานไว้เป็น RFC 3023

MIME ชนิดต่อไปคือ "image" ซึ่งใช้ในการส่งรูปภาพ รูปแบบของรูปภาพที่ใช้ในการสร้างรูปภาพในปัจจุบันนั้นมีอยู่มากมายทั้งแบบที่มีการบีบอัดข้อมูลและแบบที่ไม่มีการบีบอัดข้อมูล รูปแบบที่เป็นที่นิยมใช้งานกันสองแบบได้แก่ GIF และ JPEG ได้รับการสนับสนุนโดยโปรแกรมบราวเซอร์ทุกชนิด

ข้อความชนิด "audio" และ "video" ใช้สำหรับเก็บข้อมูลเสียงและภาพวิดีทัศน์ตามลำดับ สำหรับภาพวิดีทัศน์ในที่นี้เก็บได้เฉพาะภาพเท่านั้น ส่วนเสียงนั้นไม่สามารถเก็บไว้ได้ ถ้าต้องการส่งทั้งภาพและเสียงก็จะต้องใช้ทั้งข้อมูล audio และ video ซึ่งอาจจะต้องแยกกันส่ง ทั้งนี้ขึ้นอยู่กับระบบการเข้ารหัสข้อมูลที่เลือกใช้ รูปแบบของข้อมูลวิดีทัศน์แบบแรกที่ได้รับการพัฒนาขึ้นมาคือ Moving Picture Experts Group (MPEG) นอกเหนือจาก "audio/basic" แล้วข้อความชนิดใหม่คือ "audio/mpeg" ได้รับการกำหนดเป็นมาตรฐาน RFC 3003 ทำให้ผู้ใช้สามารถส่งเพลงที่เข้ารหัสแบบ MP3 ได้

ชนิด "application" เป็นรูปแบบแปลกใหม่ที่ต้องอาศัยการประมวลผลจากภายนอกช่วยเหลือ "octet-stream" นั้นเป็นเพียงข้อความที่เป็นลำดับของไบต์ที่ไม่มีการแปลความหมาย เมื่อได้รับข้อความประเภทนี้ user agent จะแสดงผลโดยการเสนอแนะให้ผู้ใช้ทำการสำเนาข้อมูลนี้เก็บไว้ในแฟ้มข้อมูล การประมวลผลในลำดับต่อไปก็ขึ้นอยู่กับผู้ใช้

ข้อความชนิดต่อไปคือ "postscript" ซึ่งอ้างอิงถึงภาษา PostScript language ที่พัฒนาขึ้นมาโดยบริษัท Adobe Systems และถูกนำมาใช้อย่างกว้างขวางในการอธิบายข้อมูลที่จะพิมพ์ออกทางเครื่องพิมพ์ แม้ว่า user agent จะสามารถเรียกใช้โปรแกรมภายนอก (PostScript interpreter) เพื่อทำการแสดงผลออกทางจอภาพได้ แต่การกระทำดังกล่าวก็อาจมีอันตรายแอบแฝงอยู่ก็ได้ เนื่องจาก PostScript เป็นภาษาสำหรับโปรแกรมชนิดหนึ่ง ซึ่งถ้าให้เวลามากเพียงพออาจมีคนที่สามารถเขียนโปรแกรมคอมพิวเตอร์โดยใช้ภาษา PostScript ก็ได้ การแสดงผลข้อความ PostScript ที่รับเข้ามานั้นกระทำโดยการประมวลผลโปรแกรม PostScript ที่มาพร้อมกับข้อความนั้น นอกเหนือจากการแสดงข้อความหรือภาพบนหน้าจอแล้วโปรแกรมนี้ก็มีขีดความสามารถในการอ่าน แก้ไข หรือลบแฟ้มข้อมูลของผู้ใช้ได้หรืออาจจะมีผลกระทบเลวร้ายในทางอื่นอีกก็ได้

ข้อความชนิด "message" ช่วยในการใส่ข้อความหนึ่งเข้าไปในอีกข้อความหนึ่งได้วิธีการนี้มีประโยชน์อย่างมากในการส่งอีเมลล์ไปให้คนอื่นต่อไป (forwarding)

ชนิดย่อย "partial" ช่วยในการแบ่งข้อความที่ซ่อนอยู่ในข้อความอื่น (encapsulated message) ออกเป็นส่วนๆ และจัดการนำส่งแยกกัน (อาจนำไปใช้ในกรณีที่ข้อความที่ซ่อนอยู่ในข้อความอื่นนั้นมีขนาดยาวเกินไป) การใช้พารามิเตอร์จะช่วยให้สามารถนำชิ้นส่วนต่างๆ กลับมาประกอบกันเป็นข้อความเดิมได้อย่างถูกต้อง

ข้อความชนิด "external-body" นำมาใช้กับข้อความขนาดยาวมาก เช่น วีดิทัศน์ แทนที่จะรวมไฟล์ mpeg เข้าไปในข้อความด้วยเลยก็จะใส่ที่อยู่ ftp address เข้าไปแทนและ user agent ของผู้รับก็จะสามารถดึงไฟล์นั้นผ่านระบบเครือข่ายมาได้เมื่อต้องการในภายหลัง คุณสมบัติข้อนี้มีประโยชน์เป็นอย่างมากเมื่อต้องการส่งวีดีทัศน์ไปยังผู้รับที่อยู่ใน mailing list ซึ่งอาจมีเพียงไม่กี่คนที่จะสนใจดูวีดิทัศน์นั้น

ข้อความชนิด "multipart" ช่วยให้ข้อความนั้นประกอบไปด้วยหลายส่วนได้ โดยสามารถกำหนดจุดเริ่มต้นและจุดสิ้นสุดของแต่ละส่วนได้อย่างชัดเจน ข้อความชนิด "mixed" ระบุว่าข้อความแต่ละส่วนนั้นแตกต่างกันโดยไม่จำเป็นต้องมีโครงสร้างใดๆ โปรแกรมอีเมลล์จำนวนมากอนุญาตให้ผู้ใช้แทรกข้อความเข้าไปได้หลายข้อความ ข้อความแทรกนี้เรียกว่า attachment ซึ่งถูกใส่เข้าไปโดยใช้ข้อความชนิด "multipart"

ในทางกลับกันข้อความชนิด "alternative" ช่วยในการแทรกข้อความหนึ่งเข้าไปได้หลายจุดแต่ถูกแสดงในหลายรูปแบบ ตัวอย่างเช่น ข้อความเดียวกันอาจถูกส่งในรูปแบบ ASCII text , enriched text, และ PostScript โปรแกรม user agent ที่ได้รับการออกแบบอย่างเหมาะสม เมื่อได้รับข้อความประเภทนี้แล้วเลือกที่จะแสดงข้อความในรูปแบบ PostScript ก่อนเป็นลำดับแรก ถ้าไม่สามารถทำได้ก็จะใช้รูปแบบ enriched text ถ้าทำไม่ได้ก็จะใช้รูปแบบ ASCII text ในการแสดงผล ตัวเลือกควรจะได้รับการจัดเรียงในลักษณะที่เรียงจากรูปแบบที่ง่ายที่สุดไปถึงรูปแบบที่ซับซ้อนที่สุด ข้อความชนิด "alternative" สามารถนำมาใช้กับข้อความที่เขียนด้วยหลายภาษาได้

ตัวอย่างการใช้ข้อความแบบมัลติมีเดียแสดงในรูป 7-13 ในที่นี้เป็นข้อความแสดงความยินดีในวันคล้ายวันเกิดที่ถูกส่งออกไปสองรูปแบบ คือรูปแบบตัวอักษรและรูปแบบเสียงเพลง ถ้าผู้รับมีขีดความสามารถในการเล่นเพลงได้ user agent ของผู้รับจะทำการดึงไฟล์เสียงชื่อ "birthday.snd" และเล่นให้ฟัง แต่ถ้าไม่มีขีดความสามารถนี้ก็จะแสดงข้อความบนหน้าจอภาพแต่เพียงอย่างเดียว ข้อความทั้งสองส่วนจะถูกแยกจากกันโดยมีเครื่องหมาย "--" ปรากฏอยู่ข้างหน้าตามด้วยข้อความที่เป็นตัวหนังสือชุดหนึ่ง (ไม่สามารถอ่านเป็นคำได้) ที่สร้างขึ้นมาโดยซอฟต์แวร์นำมาใช้เป็นพารามิเตอร์สำหรับการกำหนดขอบเขตของข้อความนั้น

สังเกตว่าข้อความชนิด "Content-Type" เกิดขึ้นสามตำแหน่งในตัวอย่าง ที่ตำแหน่งบนสุด จะเป็นการบอกให้ทราบว่าข้อความนี้แบ่งออกเป็นหลายส่วน ในแต่ละส่วนจะมีการระบุชนิดและชนิดย่อยของข้อความในส่วนนั้น ภายในตัวข้อความของส่วนที่สอง มีการระบุชนิดของไฟล์ภายนอก (external file) ให้ user agent ทราบ ส่วน "content-transfer-encoding" นั้นก็มีความต้องการที่จะทราบในกรณีที่ตัวข้อความในส่วนที่อยู่ภายนอกนั้นใช้วิธีการเข้ารหัสข้อมูลแบบใดที่นอกเหนือไปจาก ASCII ขนาด 7 บิต

ข้อความที่แบ่งออกเป็นหลายส่วนนั้นยังมีความเป็นไปได้ในการแสดงผลลัพธ์เป็นสองกรณีคือชนิดย่อย "parallel" ซึ่งบอกให้ทราบว่าข้อความย่อยทุกส่วนจะต้องถูกนำมาแสดงพร้อมกัน เช่น ภาพยนต์มักจะแบ่งออกเป็นสองส่วน คือส่วนภาพและส่วนเสียง ซึ่งทั้งสองส่วนนี้จำเป็นที่จะต้องเล่นไปพร้อมกัน

รูป 7-13
ข้อความที่แบ่งออกเป็นหลายส่วนที่ประกอบด้วย enriched text และ ไฟล์เสียง

```
From: elinor@abcd.com
To: carolyn@xyz.com
MIME-Version: 1.0
Message-Id: <0704760941.AA00747@abcd.com>
Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
Subject: Earth orbits sun integral number of times

This is the preamble. The user agent ignores it. Have a nice day.

--qwertyuiopasdfghjklzxcvbnm
Content-Type: text/enriched

Happy birthday to you
Happy birthday to you
Happy birthday dear <bold> Carolyn </bold>
Happy birthday to you

--qwertyuiopasdfghjklzxcvbnm
Content-Type: message/external-body;
        access-type="anon-ftp";
        site="bicycle.abcd.com";
        directory="pub";
        name="birthday.snd"

content-type: audio/basic
content-transfer-encoding: base64
--qwertyuiopasdfghjklzxcvbnm--
```

ท้ายที่สุด ชนิดย่อย "digest" ถูกนำมาใช้เมื่อข้อความหลายข้อความถูกนำมาใส่รวมเข้าไว้ด้วยกัน เรียกว่า composite message เช่น กลุ่มสนทนาบนระบบอินเตอร์เน็ตได้รวบรวมข้อความการสนทนาจากสมาชิกเข้าไว้ด้วยกันและจัดการส่งต่อไปยังสมาชิกในกลุ่ม ในลักษณะที่เป็นข้อความเดียวแบบ "multipart/digest"

7.2.4 Message Transfer

ระบบ message transfer เกี่ยวข้องกับการถ่ายทอดข้อความจากผู้ผลิตไปยังผู้รับ วิธีการที่ง่ายที่สุดในการจัดการนี้ก็คือจัดตั้งช่องสื่อสารข้อมูลขึ้นมาจากผู้ส่งไปยังผู้รับและจากนั้นก็จัดการจัดส่งข้อความ

SMTP-The Simple Mail Transfer Protocol

ภายในระบบอินเตอร์เน็ตอีเมลล์ถูกนำส่งโดยการให้เครื่องที่ผลิตข้อความขึ้นมานั้นจัดตั้งช่องสื่อสาร TCP ไปยังพอร์ตหมายเลข 25 ของเครื่องเป้าหมาย การรอรับการติดต่อของพอร์ตนี้เป็นหน้าที่ของ e-mail daemon ที่ใช้โพรโตคอล SMTP (Simple Mail Transfer Protocol) โปรแกรม daemon ตัวนี้จะคอยรับสัญญาณการขอจัดตั้งช่องสื่อสารและทำการสร้างสำเนาข้อความที่ถูกส่งเข้ามาไปส่งยังmailbox ที่ถูกต้อง ถ้าข้อความไม่สามารถถูกนำส่งได้ก็จะสร้างรายงานความล้มเหลวที่ประกอบด้วยส่วนแรกของข้อความที่ส่งมาและจัดการส่งกลับไปยังเจ้าของข้อความนั้น

SMTP เป็นโพรโตคอลแบบง่ายอย่างหนึ่งที่ใช้คำสั่งในการติดต่อเป็น ASCII text ภายหลังจากที่จัดตั้งช่องสื่อสาร TCP เข้ากับพอร์ต 25 แล้ว ผู้ส่งข้อมูลจะทำหน้าที่เป็นผู้ใช้บริการที่รอให้เครื่องผู้รับซึ่งทำหน้าที่เป็นผู้ให้บริการส่งสัญญาณติดต่อมาก่อน ผู้ให้บริการจะเริ่มต้นด้วยการส่งข้อความ text มาบรรทัดหนึ่งเพื่อบอกสัญลักษณ์ตัวตน (identity) และบอกให้ทราบว่าพร้อมที่จะรับเมลล์แล้ว แต่ถ้าไม่มีการส่งข้อความนี้ออกมาจะถือว่าเป็นการติดต่อที่ล้มเหลว ผู้ใช้บริการจะต้องยกเลิกการเชื่อมต่อนี้และพยายามใหม่ในภายหลัง

ถ้าผู้ให้บริการพร้อมที่จะรับเมลล์ ผู้ใช้บริการจะประกาศให้ทราบว่าเมลล์นี้ถูกส่งมาจากใครและจะถูกส่งไปให้ใคร ถ้าผู้รับมีตัวตนอยู่ในเครื่องเป้าหมาย ผู้ใช้บริการก็จะจัดการส่งเมลล์นั้นมาให้ และผู้ให้บริการก็จะส่งสัญญาณรับทราบ ในที่นี้ไม่มีความจำเป็นจะต้องตรวจสอบความถูกต้องของข้อมูลด้วย

```
                S: 220 xyz.com SMTP service ready
C: HELO abcd.com
                S: 250 xyz.com says hello to abcd.com
C: MAIL FROM: <elinor@abcd.com>
                S: 250 sender ok
C: RCPT TO: <carolyn@xyz.com>
                S: 250 recipient ok
C: DATA
                S: 354 Send mail; end with "." on a line by itself
C: From: elinor@abcd.com
C: To: carolyn@xyz.com
C: MIME-Version: 1.0
C: Message-Id: <0704760941.AA00747@abcd.com>
C: Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
C: Subject: Earth orbits sun integral number of times
C:
C: This is the preamble. The user agent ignores it. Have a nice day.
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: text/enriched
C:
C: Happy birthday to you
C: Happy birthday to you
C: Happy birthday dear <bold> Carolyn </bold>
C: Happy birthday to you
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: message/external-body;
C:          access-type="anon-ftp";
C:          site="bicycle.abcd.com";
C:          directory="pub";
C:          name="birthday.snd"
C:
C: content-type: audio/basic
C: content-transfer-encoding: base64
C: --qwertyuiopasdfghjklzxcvbnm
C: .
                S: 250 message accepted
C: QUIT
                S: 221 xyz.com closing connection
```

รูป 7-14
การถ่ายทอดข้อความจาก elinor@abcd.com ไปยัง carolyn@xyz.com

checksum เนื่องจาก TCP จัดตั้งช่องสื่อสารที่ไว้ใจได้ถ้ายังคงมีอีเมลล์ที่จะต้องส่งอีกผู้ใช้บริการก็สามารถจัดส่งได้ต่อไป เมื่ออีเมลล์ทั้งหมดถูกส่งมาเรียบร้อยแล้ว การเชื่อมต่อก็จะถูกยกเลิก ตัวอย่างการจัดส่งข้อความในรูป 7-13 รวมทั้งโค้ดที่ใช้ในการสื่อสารในระบบ SMTP แสดงให้เห็นในรูป 7-14 ข้อความในบรรทัดที่ส่งโดยผู้ใช้บริการจะขึ้นต้นด้วย "C:" และข้อความที่ส่งโดยผู้ให้บริการจะขึ้นต้นด้วย "S:"

คำสั่งแรกที่ส่งไปจากผู้ใช้บริการคือ "HELO" (ย่อมาจากคำว่า "hello") ในรูป 7-14 ข้อความถูกส่งมาโดยผู้รับเพียงคนเดียว ดังนั้นจึงตอบกลับไปด้วย "RCPT" เพียงครั้งเดียว คำสั่งประเภทนี้สามารถส่งไปยังผู้รับหลายคนได้ ซึ่งผู้รับแต่ละคนก็จะสามารถตอบรับหรือตอบปฏิเสธก็ได้ แม้ว่าผู้รับบางคนจะตอบปฏิเสธ (เนื่องจากไม่มีผู้รับอยู่ที่ปลายทางนั้น) ข้อความก็จะยังคงถูกส่งไปยังผู้รับที่ตอบตกลงได้

เพื่อให้ได้รับความรู้สึกที่แท้จริงของการทำงานของ SMTP และโพรโตคอลอื่นอีกบางตัวที่กล่าวถึงในที่นี้ก็ให้ลองปฏิบัติดู เริ่มต้นด้วยการไปยังเครื่องคอมพิวเตอร์ที่ใช้ระบบปฏิบัติการ Unix และมีการเชื่อมต่อเข้ากับระบบอินเตอร์เน็ต จากนั้นป้อนคำสั่งต่อไปนี้เข้าไป

```
telnet mail.isp.com 25
```

ให้แทนที่ชื่อ DSN "mail.isp.com" ด้วยชื่อ mail server ของ ISP ที่ผู้ใช้ ใช้งานอยู่ในระบบ Windows ก็ให้ป้อนคำสั่งดังกล่าวลงไปได้เหมือนกัน คำสั่งนี้จะจัดการเชื่อมต่อ TCP เข้ากับพอร์ต 25 ของเครื่อง mail server พอร์ต 25 เป็น SMTP พอร์ตดังนั้นผู้ใช้ควรได้รับคำตอบกลับมาดังนี้

```
Trying 192.30.200.66
Connected to mail.isp.com
Escape character is "^]".
220 mail.isp.com Smail #74 ready at Thu, 25 Sept 2002 13:26 +0200
```

สามบรรทัดแรกเป็นข้อความที่ส่งมาจาก telnet บอกให้ทราบว่ากำลังทำอะไรอยู่ บรรทัดสุดท้ายนั้นมาจาก SMTP server บนเครื่องผู้รับประกาศให้ทราบว่าเครื่องนั้นพร้อมที่จะรับคำสั่งต่อไปหรืออีเมลล์ ถ้าต้องการทราบว่ามีคำสั่งอะไรบ้างให้พิมพ์คำสั่ง

```
HELP
```

จากจุดนี้การส่งคำสั่งต่างๆ เช่นที่ได้แสดงในรูป 7-14 นั้นสามารถทำได้โดยเริ่มต้นจากการส่งคำสั่ง "HELO"

เป็นที่น่าสังเกตว่าการใช้คำสั่ง ASCII นั้นไม่ใช่เป็นเรื่องที่เกิดขึ้นโดยบังเอิญ เพราะว่าโพรโตคอลบนอินเตอร์เน็ตส่วนใหญ่ทำงานในลักษณะเดียวกันนี้ การใช้ ASCII text ช่วยให้สามารถทดลองและแก้ไขโพรโตคอลได้โดยง่าย โพรโตคอลเหล่านี้จะสามารถถูกทดสอบได้โดยการส่งคำสั่งทีละคำสั่งซึ่งก็จะได้รับคำตอบกลับมาดังที่เห็น

แม้ว่าโพรโตคอล SMTP จะได้รับการออกแบบมาเป็นอย่างดี แต่ก็อาจมีปัญหาบางอย่างเกิดขึ้นได้ ปัญหาประการหนึ่งเกี่ยวข้องกับความยาวของข้อความ SMTP รุ่นเก่าบางส่วนไม่สามารถรับข้อความที่มี

ความยาวเกิน 64 KB ได้ อีกปัญหาหนึ่งเกี่ยวกับระยะเวลารอคอย ถ้าผู้ใช้บริการและผู้ให้บริการมีระยะเวลารอคอยที่แตกต่างกัน หนึ่งในนั้นอาจยกเลิกการทำงานแล้วในขณะที่อีกฝ่ายหนึ่งยังคงทำงานอยู่ ซึ่งเป็นการยกเลิกการติดต่ออย่างที่ไม่คาดฝัน ประการสุดท้าย ในบางกรณีที่อาจจะเกิดขึ้นได้น้อยมาก อาจมีการส่งข้อความเข้ามาอย่างไม่จำกัด ตัวอย่างเช่น ถ้าโฮส 1 มี mailing list A และโฮส 2 มี mailing list B แต่ละรายการมีชื่อผู้รับของอีกฝ่ายหนึ่งอยู่ ดังนั้นข้อความที่ส่งไปถึงอีกฝ่ายหนึ่งก็จะกระตุ้นให้เกิดการส่งข้อความกลับมายังอีกฝ่ายหนึ่งได้อย่างไม่รู้จบ

เพื่อแก้ปัญหาบางอย่างที่กล่าวถึงนี้ จึงได้มีการสร้าง extended SMTP (ESMTP) ขึ้นมาและกำหนดไว้ในมาตรฐาน RFC 2821 ผู้ใช้บริการที่ต้องการใช้ระบบ ESMTP จะเริ่มต้นด้วยการส่งข้อความ "EHLO" มาแทนที่คำสั่ง "HELO" ถ้าผู้ให้บริการตอบปฏิเสธก็แสดงว่าเป็นผู้ให้บริการที่ใช้โพรโตคอล SMTP รุ่นก่อน แต่ถ้า "EHLO" ได้รับการตอบรับ ผู้ใช้บริการก็จะสามารถใช้คำสั่งรุ่นใหม่ได้

7.2.5 Final Delivery

จนกระทั่งถึงบัดนี้ข้อสมมุติฐานเดิมที่ใช้อยู่ก็คือเครื่องผู้รับนั้นมีความสามารถในการรับและส่งอีเมลล์ดังที่ได้กล่าวไปแล้วอีเมลล์ได้รับการนำส่งโดยการจัดตั้งการเชื่อมต่อ TCP ไปยังผู้รับจากนั้นจึงส่งอีเมลล์ผ่านมาทางช่องสื่อสารนี้รูปแบบการทำงานนี้สามารถใช้งานมาได้นานนับสิบปีเมื่อโฮสทุกตัวในระบบ ARPANET นั้นทำงานอยู่ตลอดเวลาในการยอมรับการเชื่อมต่อแบบ TCP

อย่างไรก็ตามเมื่อมีเหตุการณ์ที่ผู้ใช้ที่ติดต่อเข้าสู่ระบบอินเตอร์เน็ตทาง ISP โดยใช้โมเด็มและพยายามที่จะจัดตั้งการเชื่อมต่อ TCP แต่ไม่สำเร็จ ปัญหาที่เกิดขึ้นคือ ถ้า Elinor ต้องการส่งอีเมลล์ไปยัง Carolyn แต่ Carolyn ไม่ได้ใช้งานคอมพิวเตอร์อยู่ Elinor ก็จะไม่สามารถจัดตั้งการเชื่อมต่อ TCP ขึ้นมาได้จึงไม่สามารถใช้งานโพรโตคอล SMTP ได้

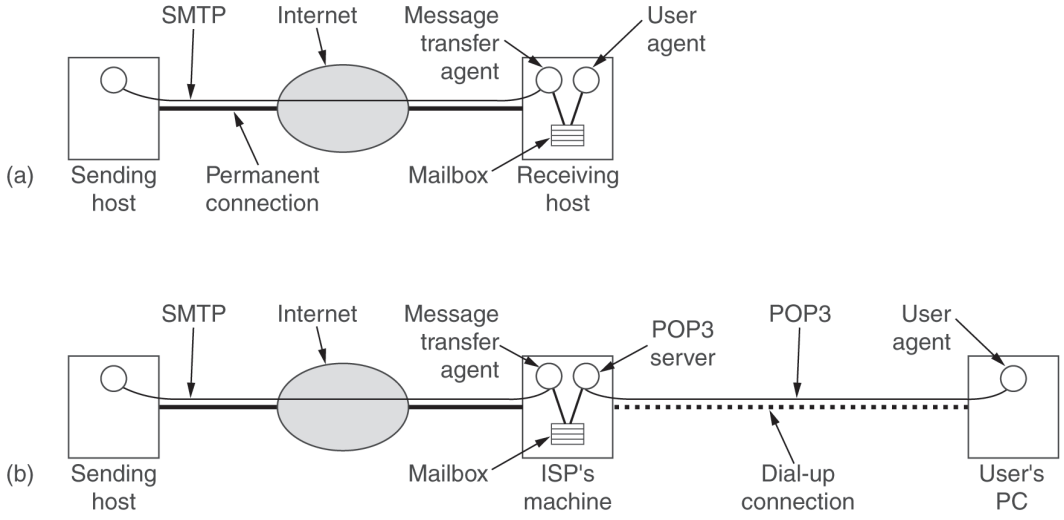
หนทางแก้ปัญหาทางหนึ่งก็คือให้ message transfer agent บนเครื่อง ISP รับอีเมลล์แทนผู้ใช้ของตนเองและจัดการเก็บอีเมลล์นั้นไว้ใน mailbox บนเครื่อง ISP เนื่องจากเครื่องของ ISP จะทำงานอยู่ตลอดเวลา ดังนั้นจึงสามารถส่งอีเมลล์ได้ตลอดเวลาเช่นกัน

POP3

อย่างไรก็ตาม การแก้ปัญหาในลักษณะที่กล่าวถึงนี้ได้สร้างปัญหาใหม่ขึ้นมาคือ ผู้ใช้จะอ่านอีเมลล์ของตนเองจาก message transfer agent ของ ISP ได้อย่างไร หนทางแก้ปัญหานี้ก็คือ จะต้องสร้างโพรโตคอลใหม่ขึ้นมาที่ยอมให้ transfer agent ของผู้ใช้ (บนเครื่องพีซีของผู้ใช้เอง) สามารถติดต่อกับ message transfer agent (บนเครื่อง ISP) และจัดการส่งสำเนาอีเมลล์จาก ISP มายังผู้ใช้ได้ โพรโตคอลประเภทนี้เรียกว่า POP3 (Post Office Protocol Version 3) ซึ่งตรงกับมาตรฐาน RFC 1939

สถานะการณ์ที่เคยเกิดขึ้นคือทั้งผู้ส่งและผู้รับมีการเชื่อมต่อบนอินเตอร์เน็ตอย่างถาวร แสดงให้เห็นในรูป 7-15(a) ส่วนสถานะการณ์ที่ผู้ส่งกำลังทำงานอยู่ในขณะที่เครื่องผู้รับไม่ได้ทำงาน แสดงให้เห็นในรูป 7-15(b)

POP3 เริ่มทำงานเมื่อผู้ใช้เริ่มประมวลผลโปรแกรมอ่านเมลล์ โปรแกรมนี้จะติดต่อไปยัง ISP (นอกจากว่าจะมีการเชื่อมต่อเกิดขึ้นแล้ว) และจัดตั้งการเชื่อมต่อ TCP ขึ้นมาโดยที่ message transfer agent ทำงานอยู่ที่พอร์ต 110 เมื่อการเชื่อมต่อเกิดขึ้นแล้ว POP3 จะเปลี่ยนสถานะการทำงานเป็นสามขั้นตอนดังนี้



รูป 7-15 (a) การส่งและรับอีเมลล์เมื่อผู้รับมีการเชื่อมต่อกับระบบอินเตอร์เน็ตอย่างถาวร และ user agent งานอยู่บนคอมพิวเตอร์เครื่องเดียวกับ message transfer agent (b) การอ่านอีเมลล์เมื่อผู้รับใช้วิธีการเชื่อมต่อผ่านโมเด็มไปยัง ISP

1. การตรวจสอบผู้ใช้ (Authorization)

2. รายการทำงาน (Transaction)

3. การปรับปรุงข้อมูล (Update)

สถานะการตรวจสอบผู้ใช้ หมายถึงขั้นตอนที่บังคับให้ผู้ใช้ทำการ log-in เข้าสู่ระบบ สถานะรายการทำงานอยู่ในขั้นตอนที่ผู้ใช้ถ่ายโอนอีเมลล์ไปที่เครื่องของตนเองและทำเครื่องหมายสำหรับการลบข้อความอีเมลล์นั้นๆ ทิ้งจาก mailbox ขั้นตอนการปรับปรุงข้อมูลจะทำการลบข้อความอีเมลล์ที่ได้ทำเครื่องหมายไว้แล้วออกจาก mailbox

สิ่งที่เกิดขึ้นนี้สามารถสังเกตเห็นได้ด้วยการใช้คำสั่ง

```
telnet mail.isp.com 110
```

โดยที่ "mail.isp.com" หมายถึงชื่อ DNS ของ mail server ของ ISP ของผู้ใช้ คำสั่ง "telnet" จะทำการจัดตั้งการเชื่อมต่อ TCP เข้ากับพอร์ต 110 ซึ่งเซิบเวอร์ของ POP3 ทำงานอยู่ เมื่อยอมรับการจัดตั้งช่องสื่อสารนี้แล้ว เซิบเวอร์จะส่งข้อความเป็น ASCII text ประกาศว่าการเชื่อมต่อเรียบร้อยแล้ว โดยปกติจะเริ่มต้นข้อความด้วย "+OK" ตามด้วย comment รูป 7-16 แสดงตัวอย่างการทำงานที่เกิดขึ้นเริ่มจากการที่ได้จัดตั้งช่องสื่อสาร TCP เสร็จเรียบร้อยแล้ว ดังเช่นเคย คำย่อ "C:" นั้นหมายถึงข้อความที่ส่งมาจากผู้ใช้ และ "S:" นั้นเป็นข้อความที่ส่งมาจากเครื่องผู้ให้บริการ (message transfer agent บนเครื่อง ISP)

ในระหว่างที่อยู่ในขั้นตอนการตรวจสอบผู้ใช้ ผู้ใช้บริการจะส่งชื่อผู้ใช้ (user name) และตามด้วยรหัสผ่าน (password) มาให้ ภายหลังจากที่กระบวนการ log-in เสร็จสิ้นแล้ว ผู้ใช้บริการจะสามารถส่งคำสั่ง "LIST" ซึ่งเซิบเวอร์จะตอบโดยการส่งรายการข้อความอีเมลล์ทั้งหมดที่มีอยู่ใน mailbox มาให้ (บรรทัดละฉบับ) และบอกความยาวของแต่ละข้อความด้วย รายการทั้งหมดจะปิดท้ายด้วยบรรทัดว่างที่มีเพียงเครื่องหมาย "."

```
                S: +OK POP3 server ready
C: USER carolyn
                S: +OK
C: PASS vegetables
                S: +OK login successful
C: LIST
                S: 1 2505
                S: 2 14302
                S: 3 8122
                S: .
C: RETR 1
                S: (sends message 1)
C: DELE 1
C: RETR 2
                S: (sends message 2)
C: DELE 2
C: RETR 3
                S: (sends message 3)
C: DELE 3
C: QUIT
                S: +OK POP3 server disconnecting
```

รูป 7-16
การใช้ POP3 ในการดึงข้อความอีเมลล์ 3 ข้อความ

จากนั้นผู้ใช้บริการจะรับข้อความอีเมลล์ได้โดยใช้คำสั่ง "RETR" และทำเครื่องหมายเพื่อการลบทิ้งด้วยคำสั่ง "DELE" เมื่อได้รับข้อความอีเมลล์ครบทั้งหมดแล้ว ผู้ใช้บริการจะออกคำสั่ง "QUIT" เพื่อยกเลิกสถานะการทำรายการและเข้าสู่สถานะการปรับปรุงข้อมูล ผู้ให้บริการจะทำการลบข้อความอีเมลล์ที่ได้ทำเครื่องหมายลบเอาไว้ทั้งหมดจากนั้นก็จะยกเลิกการเชื่อมต่อ TCP

โพรโตคอล POP3 สนับสนุนความสามารถในการดาวน์โหลดข้อความอีเมลล์บางข้อความหรือกลุ่มข้อความแล้วทิ้งข้อความค้างไว้ที่เซิบเวอร์ แต่โปรแกรมอีเมลล์ส่วนใหญ่จะทำการดาวน์โหลดข้อความอีเมลล์ทั้งหมดมาจากเซิบเวอร์แล้วปล่อยให้ mailbox ว่างเปล่า ซึ่งหมายความว่าผู้ใช้จะมีข้อความเก็บไว้ที่ตนเองเพียงที่เดียว หากเกิดเหตุการณ์ที่ทำให้ข้อความเหล่านั้นสูญหายผู้ใช้ก็จะไม่สามารถกู้ข้อความอีเมลล์ใดๆ กลับคืนมาได้เลย

ต่อไปจะกล่าวโดยสรุปให้เห็นภาพการทำงานของอีเมลล์สำหรับลูกค้าของ ISP Elinor ทำการสร้างข้อความเพื่อที่จะส่งไปให้ Carolyn โดยใช้อีเมลล์โปรแกรมและจัดการส่งอีเมลล์ฉบับนั้น โปรแกรมอีเมลล์จะส่งข้อความอีเมลล์ไปยัง message transfer agent ของโฮสของ Elinor โปรแกรม message transfer agent จะเห็นว่าข้อความนั้นถูกส่งไปที่ carolyn@xyz.com จึงใช้ DNS ในการร้องขอ MX record สำหรับ xyz.com (ซึ่งก็คือ ISP ของ Carolyn) คำตอบที่ได้รับจาก DNS จะเป็นชื่อ mail server ของ xyz.com โปรแกรม message transfer agent ก็จะค้นหาหมายเลข IP ของ mail server นี้ผ่านทาง DNS อีกครั้งหนึ่ง เช่นการใช้คำสั่ง "gethostbyname" จากนั้นจะทำการจัดตั้งเชื่อมต่อ TCP เข้ากับ SMTP server ที่อยู่บนพอร์ต 25 ของเครื่อง mail server นั้น หลังจากนั้นจะใช้คำสั่ง SMTP คล้ายกับที่แสดงในรูป 7-14 เพื่อส่งข้อความอีเมลล์ไปเก็บไว้ที่ mailbox ของ Carolyn และยกเลิกการเชื่อมต่อ

เมื่อ Carolyn ทำการเปิดเครื่องคอมพิวเตอร์ของเธอ การเชื่อมต่อไปยัง ISP ของเธอ และเริ่มใช้โปรแกรมอีเมลล์ โปรแกรมจะทำการจัดตั้งการเชื่อมต่อ TCP เข้ากับ POP3 server ที่พอร์ต 110 ของเครื่อง mail server ของ ISP ชื่อ DNS หรือหมายเลข IP ของเครื่อง mail server นี้มักจะได้รับการกำหนดขึ้นเมื่อทำการติดตั้งโปรแกรมอีเมลล์หรือเมื่อสมัครเป็นสมาชิกของ ISP เมื่อจัดตั้งการเชื่อมต่อ ISP ได้แล้ว โปรแกรมอีเมลล์ของ Carolyn จะใช้โพรโตคอล POP3 ในการดึงข้อความอีเมลล์ที่เก็บอยู่ใน mailbox มาเก็บไว้ที่ฮาร์ดดิสก์ของเครื่องพีซีของเธอ (คล้ายกับขั้นตอนที่แสดงในรูป 7-16) เมื่อข้อความอีเมลล์ทั้งหมดได้รับการถ่ายทอดมาแล้ว ก็จะยกเลิกการเชื่อมต่อ TCP อันที่จริงการเชื่อมต่อเข้ากับ ISP ก็สามารถจะยกเลิกไปพร้อมกันได้เนื่องจากอีเมลล์ทั้งหมดอยู่ที่เครื่องของ Carolyn แล้ว ถ้าเธอต้องการส่งอีเมลล์ตอบกลับไปยังผู้ใดก็ตาม ก็จะต้องอาศัยการเชื่อมต่อไปยัง ISP อีกครั้งหนึ่ง ดังนั้นจึงไม่นิยมที่จะยกเลิกการเชื่อมต่อกับ ISP ในทันที

IMAP

สำหรับผู้ใช้ที่มีบัญชีอีเมลล์เพียงบัญชีเดียวที่ ISP แห่งหนึ่งและใช้ POP3 ในการเชื่อมต่อเสมอ โพรโตคอล POP3 จะสามารถใช้งานได้ดีและถูกนำไปใช้งานอย่างกว้างขวางเนื่องจากเป็นโพรโตคอลที่ทำงานง่ายแต่มีความแข็งแกร่งในการทำงานสูง อย่างไรก็ตาม เป็นธรรมดาในวงการคอมพิวเตอร์ที่ว่าเมื่อใดมีอะไรสักอย่างที่ทำงานได้ดี ก็มักจะมีคนต้องการอะไรบางอย่างที่สามารถทำงานได้มากกว่าเดิมเสมอ จุดอ่อนของ POP3 ก็คือในทุกครั้งที่มีการเชื่อมต่อ มันจะส่งข้อความอีเมลล์ทั้งหมดมาเก็บไว้ที่เครื่องที่เชื่อมต่อกันอยู่ ดังนั้นถ้าผู้ใช้อีเมลล์ทำการติดต่อจากเครื่องหมายในเครื่องหลายสถานที่ในหลายโอกาส ก็จะทำให้ข้อความอีเมลล์ของเขานั้นกระจายอยู่ตามเครื่องต่างๆ

โพรโตคอลอีกชนิดหนึ่งที่สามารถนำมาใช้งานได้ก็คือ IMAP (Internet Message Access Protocol) ซึ่งกำหนดไว้ในมาตรฐาน RFC 2060 ในขณะที่โพรโตคอล POP3 ตั้งสมมุติฐานว่าผู้ใช้จะลบข้อความอีเมลล์ทั้งหมดออกไปจาก mailbox ในทุกครั้งที่ทำการติดต่อเข้ามา แต่โพรโตคอล IMAP จะตั้งสมมุติฐานว่าข้อความอีเมลล์ทั้งหมดจะยังคงอยู่ที่เซิบเวอร์ไปตลอดและอาจถูกเก็บไว้ในหลาย mailbox ก็ได้ IMAP นำเสนอกลไกอย่างมากมายในการอ่านข้อความหรือบางส่วนของข้อความซึ่งเป็นคุณสมบัติที่สำคัญเมื่อผู้ใช้ทำการติดต่อผ่านโมเด็มที่ทำงานได้ช้า โดยเฉพาะอย่างยิ่งในการอ่านข้อความแบบหลายส่วนที่ประกอบด้วยไฟล์เสียงและไฟล์วีดีทัศน์ขนาดใหญ่ เนื่องจากการสมมุติฐานของทำงานนั้นคือจะไม่ถ่ายทอดข้อความไปยังเครื่องคอมพิวเตอร์ของผู้ใช้เพื่อจัดเก็บโดยถาวร IMAP จึงนำเสนอกลไกสำหรับการสร้าง การลบทิ้ง และการจัดการข้อความอีเมลล์ในหลาย mailbox ที่เก็บอยู่บนเครื่องเซิบเวอร์ ด้วยวิธีการนี้ผู้ใช้จะสามารถเก็บรักษา mailbox สำหรับการโต้ตอบแต่ละส่วนและเคลื่อนย้ายข้อความอีเมลล์มาจาก inbox หลังจากที่ได้อ่านข้อความนั้นแล้ว

IMAP มีขีดความสามารถหลายด้าน เช่น ความสามารถในการจัดเรียงเมลล์นอกเหนือจากเรียงตามลำดับที่เข้ามาดังที่แสดงในรูป 7-8 โดยสามารถจัดเรียงตามคุณสมบัติอื่น เช่น ให้แสดงข้อความอีเมลล์ที่ส่งมาจาก Bobbie IMAP ยังแตกต่างจาก POP3 ตรงที่สามารถจัดส่งอีเมลล์ให้ได้เหมือนกับการรับอีเมลล์

รูปแบบทั่วไปของ IMAP ก็คล้ายคลึงกับ POP3 ดังที่แสดงในรูป 7-16 ยกเว้นแต่ว่ามีคำสั่งใช้งานมากกว่าหลายเท่า IMAP จะทำงานอยู่บนพอร์ต 143 การเปรียบเทียบระหว่าง POP3 และ IMAP แสดงในรูป

Feature	POP3	IMAP
Where is protocol defined	RFC 1939	RFC 2060
TCP port used	110	143
Where is e-mail stored	User's PC	Server
Where is e-mail read	Off-line	On-line
Connect time required	Little	Much
Use of server resources	Minimal	Extensive
Multiple mailboxes	No	Yes
Who backs up mailboxes	User	ISP
Good for mobile users	No	Yes
User control over downloading	Little	Great
Partial message downloads	No	Yes
Are disk quotas a problem	No	Could be in time
Simple to implement	Yes	No
Widespread support	Yes	Growing

รูป 7-17
การเปรียบเทียบ POP3 กับ IMAP

7-17 เป็นที่น่าสังเกตว่า ISP บางส่วนจะไม่สนับสนุนโพรโตคอลทั้งสองแบบ และโปรแกรมอีเมลล์บางส่วนก็ไม่สนับสนุนโพรโตคอลทั้งสองแบบเช่นกัน ดังนั้นเมื่อเลือกใช้โปรแกรมอีเมลล์ จึงมีความจำเป็นที่จะต้องทราบว่าโปรแกรมนั้นสนับสนุนโพรโตคอลแบบใด และจะต้องแน่ใจว่า ISP ที่ใช้งานอยู่นั้นสนับสนุนโพรโตคอลชนิดเดียวกันด้วย

คุณสมบัติในการนำส่งอีเมลล์

ไม่ว่าจะใช้โพรโตคอล POP3 หรือ IMAP ระบบส่วนใหญ่จะนำเสนอความสามารถเพิ่มเติมในการประมวลผลอีเมลล์ที่เพิ่งรับเข้ามา คุณสมบัติที่สำคัญมากอย่างหนึ่งที่ผู้ใช้อีเมลล์ส่วนมากต้องการก็คือการจัดตั้ง "filters" หรือ "ตัวกรองข่าวสาร" ซึ่งหมายถึงกฎข้อบังคับที่จะถูกนำมาตรวจสอบเมื่อมีอีเมลล์ถูกส่งเข้ามาหรือเมื่อ user agent เริ่มทำงาน กฎแต่ละข้อจะกำหนดเงื่อนไข (condition) และการกระทำ (action) ที่ต้องการ ตัวอย่างเช่น กฎอาจระบุว่าข้อความอีเมลล์ใดก็ตามที่รับมาจากเจ้านายจะต้องนำไปเก็บไว้ที่ mailbox 1 ข้อความใดที่ส่งมาจากเพื่อนในกลุ่มหนึ่งจะต้องนำไปเก็บไว้ที่ mailbox 2 และข้อความใดที่มีคำต้องห้ามอยู่ใน "Subject" จะให้ลบทิ้งในทันที

ISP บางส่วนนำเสนอการกรองข่าวสารโดยการจัดประเภทของอีเมลล์ที่ถูกส่งเข้ามาว่าเป็นข้อความที่สำคัญ หรือเป็น spam (junk mail-อีเมลล์ที่มักจะเป็นการเสนอขายสินค้าหรือบริการหรือเป็นเรื่องไร้สาระต่างๆ) และจัดการเก็บอีเมลล์ไว้ใน mailbox สำหรับข้อมูลแต่ละประเภท การกรองข่าวสารดังกล่าวจะทำการตรวจสอบดูว่าอีเมลล์ที่ส่งเข้ามานั้นส่งมาจากที่อยู่ที่รู้จักหรือไม่ (เช่นส่งมาจากหมายเลข IP ที่มักจะส่งโฆษณาเข้ามาเป็นประจำ) จากนั้นจะสำรวจคำที่อยู่ในบรรทัด "Subject" ถ้าผู้รับเป็นจำนวนมากได้รับข้อความที่มี "Subject" เหมือนกันก็แสดงว่าอีเมลล์กลุ่มนี้อาจเป็นพวก spam เทคนิคอื่นๆ ก็ถูกคิดค้นขึ้นมาเพื่อใช้ตรวจจับ spam อีเมลล์

คุณสมบัติในการนำส่งอีเมลล์อีกอย่างหนึ่งก็คือ ความสามารถในการจัดส่งอีเมลล์ไปยังผู้รับคนอื่น (mail forwarding) หมายเลขที่อยู่นี้อาจเป็นที่อยู่ของเครื่องคอมพิวเตอร์ที่ให้บริการส่งข้อความไปยัง

วิทยุติดตามตัว (pager) ซึ่งเมื่อได้รับอีเมลล์แล้วก็จะส่งข้อความไปยังวิทยุติดตามตัวของผู้ใช้โดยจะแสดงข้อความที่อยู่ในบรรทัด "Subject" ไปให้ผู้ใช้ทราบ

ความสามารถอีกอย่างหนึ่งคือการติดตั้ง "vacation daemon" ซึ่งเป็นโปรแกรมที่ตรวจสอบอีเมลล์ที่ส่งเข้ามาและจัดการส่งอีเมลล์ตอบกลับไปยังผู้ส่งด้วยข้อความ เช่น

"Hi! I'am on vacation. I'll be back on the 24th of August. Have a nice day."

"สวัสดี ข้าพเจ้ากำลังอยู่ในระหว่างพักร้อน จะกลับมาทำงานในวันที่ 24 สิงหาคม ขอให้โชคดี"

การตอบกลับไปเช่นนี้อาจจะบอกวิธีการติดต่อในกรณีเร่งด่วนเอาไว้ หรือบอกชื่อผู้ที่สามารถติดต่อแทนได้ในเรื่องบางเรื่อง เป็นต้น Vacation daemon ส่วนใหญ่จะเก็บบันทึกการตอบกลับไปในลักษณะนี้เอาไว้เพื่อจะได้ไม่ต้องตอบกลับไปซ้ำอีก โปรแกรมที่ดีจะทำการตรวจสอบดูว่าข้อความที่ได้รับนั้นถูกส่งออกไปจากตนเองโดยใช้ mailing list หรือไม่ ถ้าใช่ก็จะไม่ส่งข้อความไปบอกอีก

Webmail

หัวข้อสุดท้ายที่น่าสนใจคือ Webmail เว็บไซต์บางแห่งเช่น Yahoo และ Hotmail ให้บริการอีเมลล์แก่ใครก็ได้ที่ต้องการใช้งาน ระบบนี้ทำงานดังนี้ เว็บไซต์จะมี message transfer agent ทำงานอยู่ที่พอร์ต 25 เพื่อรอรับการเชื่อมต่อ SMTP ตามปกติ ในการติดต่อ เช่น ติดต่อไปที่ Hotmail ผู้ใช้จะต้องถามหา DNS MX record โดยการส่งคำสั่ง

```
host -a -v hotmail.com
```

ในระบบยูนิกส์ สมมติว่า mail server นั้นคือ mx10.hotmail.com จากนั้นผู้ใช้จึงส่งคำสั่ง

```
telnet mx10.hotmail.com 25
```

ผู้ใช้ก็จะสามารถจัดตั้งการเชื่อมต่อ TCP ซึ่งจะสามารถใช้คำสั่ง SMTP ได้ โดยปกติเว็บไซต์พวกนี้จะยุ่งมากดังนั้น ผู้ใช้อาจต้องใช้ความพยายามหลายครั้งกว่าที่จะเชื่อมต่อได้สำเร็จ

ส่วนที่น่าสนใจคือ อีเมลล์จะถูกส่งมาได้อย่างไร โดยพื้นฐานแล้ว เมื่อผู้ใช้ไปที่อีเมลล์เว็บเพจ จะมีใบกรอกข้อความขึ้นมาซึ่งผู้ใช้จะต้องกรอกชื่อผู้ใช้และรหัสผ่านเข้าไป ซึ่งจะถูกส่งไปที่เซิบเวอร์เพื่อทำการตรวจเช็ค ถ้าเป็นผู้ใช้ที่ถูกต้องการ log-in ก็จะประสบผลสำเร็จโดยเซิบเวอร์จะค้นหา mailbox ของผู้ใช้และสร้างรายการคล้ายกับที่แสดงในรูป 7-8 แต่เป็นการจัดรูปแบบการแสดงด้วยภาษา HTML ที่แสดงผ่านเว็บเพจ เว็บเพจจะถูกส่งมาที่บราวเซอร์เพื่อการแสดงผลทำให้ข้อความอีเมลล์สามารถถูกอ่าน ลบ หรือทำอย่างอื่นได้ตามต้องการ

7.3 The World Wide Web

World Wide Web (WWW) เป็นสถาปัตยกรรมโครงสร้างสำหรับการติดต่อไปยังเอกสารที่ถูกเชื่อมต่อเข้าด้วยกันที่มีการนำมาใช้งานอย่างแพร่หลาย ความสนใจที่ได้รับอย่างแพร่หลายนี้มาจากส่วนติดต่อผู้ใช้ที่หลากสีสรรและง่ายต่อการใช้งานแม้สำหรับผู้ที่ไม่เคยใช้มาก่อนเลยก็ตาม

เว็บ หรือ WWW เริ่มต้นขึ้นในปี ค.ศ. 1989 ที่ CERN (the European Center for Nuclear Research) ซึ่งเป็นสถานที่ที่ทำการค้นคว้าวิจัยเกี่ยวกับอนุภาคทางฟิสิกส์ ทีมผู้เชี่ยวชาญที่มาจากหลายประเทศทั่วยุโรปมาร่วมกันทำงานซึ่งจะต้องมีการรายงานความก้าวหน้าในเรื่องที่ตนเองค้นคว้าวิจัยอยู่ เว็บจึงเกิดขึ้นเพื่อสนองความต้องการในการกระจายข่าวสารเหล่านี้ซึ่งมีทั้งเอกสาร แผนภาพ รูปภาพ และอื่นๆมากมายไปยังผู้ร่วมทำการค้นคว้าวิจัยหลายพันคนที่ต้องการข่าวสารที่ใหม่สดเสมอ

ข้อเสนอเริ่มต้นของเว็บที่เชื่อมต่อเอกสารต่างๆ เข้าด้วยกันมาจาก Tim Berners-Lee ในเดือนมีนาคม ค.ศ. 1989 เอกสารทดลองชิ้นแรกที่เป็นตัวหนังสือเพียงอย่างเดียวจึงเกิดขึ้นในอีก 18 เดือนต่อมา ในเดือนธันวาคม ค.ศ. 1991 การนำเสนอต่อสาธารณะจึงเกิดขึ้นที่การประชุม Hypertext 91 conference ที่ San Antonio, Texas

การแสดงการทดสอบนี้ได้กระตุ้นความสนใจของผู้วิจัยอื่นๆ ซึ่งนำไปสู่การพัฒนาที่นำโดย Mark Andreessen แห่งมหาวิทยาลัย University of Illinois ในการคิดค้นบราวเซอร์ที่เป็นกราฟฟิกตัวแรกขึ้นมาคือ Mosaic โปรแกรมนี้ได้รับการเผยแพร่ออกไปในเดือนกุมภาพันธ์ ค.ศ. 1993 ซึ่งได้รับความนิยมในการใช้งานในเวลาต่อมา Andreessen ได้ลาออกจากมหาวิทยาลัยไปจัดตั้งบริษัทของตนเองขึ้นมา ใช้ชื่อว่า Netscape Communications Corp. ซึ่งมีวัตถุประสงค์ในการพัฒนาโปรแกรมสำหรับผู้ใช้บริการ เซิบเวอร์ และซอฟต์แวร์สำหรับเว็บแบบอื่นๆ เมื่อ Netscape ได้เปิดเผยตนเองสู่สาธารณะชนในปี 1995 ซึ่งได้รับความสนใจเป็นอย่างมากถึงขนาดที่ทำให้หุ้นของบริษัทมีมูลค่าสูงถึง 1.5 พันล้านเหรียญสหรัฐฯเลยทีเดียว สามปีต่อมา Netscape ได้ทำสงครามทางการค้ากับ Microsofts Internet Explorer ซึ่งทั้งสองฝ่ายได้คิดค้นพัฒนาให้ซอฟต์แวร์ของตนเองมีขีดความสามารถมากขึ้นเรื่อยๆ จนกระทั่งในปี ค.ศ. 1998 บริษัท America Online จึงได้ซื้อบริษัท Netscape เป็นเงินถึง 4.2 พันล้านเหรียญซึ่งเป็นการปิดฉาก ความเป็นอิสระของบริษัท Netscape ลง

ในปี ค.ศ. 1994 CERN และมหาวิทยาลัย M.I.T. ได้ทำข้อตกลงในการจัดตั้งองค์กร World Wide Web Consortium (W3C) ขึ้นมามีวัตถุประสงค์ในการพัฒนาเว็บ โพรโตคอลมาตรฐาน และกระตุ้นให้เกิดความสามารถในการทำงานร่วมกันระหว่างเว็บไซต์ต่างๆ และได้จัดตั้งโฮมเพจขององค์กรนี้ขึ้นที่ www.w3.org

7.3.1 สถาปัตยกรรม

จากมุมมองของผู้ใช้ เว็บประกอบด้วยเอกสารจำนวนมหาศาลที่เรียกว่าเว็บเพจ (web page) หรือเรียกสั้นๆ ว่า เพจ (page) แต่ละเพจประกอบด้วยการเชื่อมต่อ (link) ไปยังเพจอื่นที่ใดก็ได้ในโลก ผู้ใช้สามารถติดตามการเชื่อมต่อนั้นไปได้ด้วยการใช้เมาส์คลิ๊กซึ่งจะพาไปยังเพจที่เชื่อมต่ออยู่นั้น กระบวนการนี้สามารถที่จะเกิดขึ้นซ้ำแล้วซ้ำเล่าได้ไม่จำกัด แนวความคิดในการให้เพจหนึ่งชี้ไปยังเพจอื่นเรียกว่า “hypertext” ซึ่งได้รับการคิดค้นขึ้นมาโดย Vannevar Bush ในปี ค.ศ. 1945

เพจจะได้รับการนำมาแสดงโดยโปรแกรมเรียกว่า บราวเซอร์ (browser) ซึ่งได้แก่ Netscape Navigator และ Internet Explorer บราวเซอร์จะทำการดึงข้อมูลจากเพจที่ถูกเรียกขึ้นมาใช้งาน ทำการแปลความหมายของข้อความและรูปแบบของคำสั่งต่างๆ และจัดการแสดงผลลัพธ์ที่ได้บนหน้าจอภาพอย่างถูกต้อง ตัวอย่างดังในรูป 7-18(a) กลุ่มของข้อความที่เชื่อมต่อไปยังเพจอื่นเรียกว่า hyperlink

รูป 7-18
(a) เว็บเพจหน้าหนึ่ง
(b) เพจที่ถูกอ้างอิงถึง
โดยการใช้เมาส์คลิ๊กไป
ที่ Department of
Animal Psychology

WELCOME TO THE UNIVERSITY OF EAST PODUNK'S WWW HOME PAGE

- Campus Information
  - Admissions information
  - Campus map
  - Directions to campus
  - The UEP student body
- Academic Departments
  - Department of Animal Psychology
  - Department of Alternative Studies
  - Department of Microbiotic Cooking
  - Department of Nontraditional Studies
  - Department of Traditional Studies

Webmaster@eastpodunk.edu

(a)

THE DEPARTMENT OF ANIMAL PSYCHOLOGY

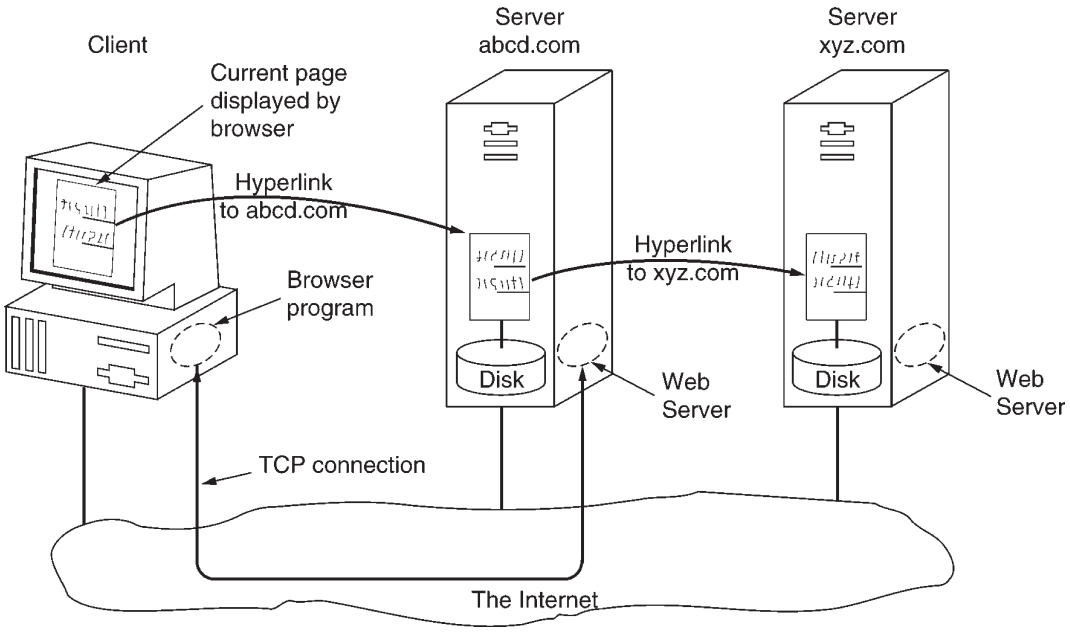
- Information for prospective majors
- Personnel
  - Faculty members
  - Graduate students
  - Nonacademic staff
- Research Projects
- Positions available
- Our most popular courses
  - Dealing with herbivores
  - Horse management
  - Negotiating with your pet
  - User-friendly doghouse construction
- Full list of courses

Webmaster@animalpsyc.eastpodunk.edu

(b)

มักจะได้รับการแสดงด้วยสีที่เด่นกว่าปกติ หรือขีดเส้นใต้ หรือใช้สีที่แตกต่างไปจากตัวอักษรอื่น เพื่อทำการตามการเชื่อมต่อนั้นไปยังเพจอื่น ผู้ใช้จะใช้เมาส์ลาก cursor (ตัวชี้ตำแหน่งของเมาส์) ไปยังคำที่แสดงการเชื่อมต่อและคลิ๊กที่เมาส์

จากในรูป ผู้ใช้ที่สนใจใน Department of Animal Psychology จะสามารถได้รับข้อมูลเพิ่มเติมโดยการคลิ๊กที่ชื่อนั้นบราวเซอร์จะทำการดึงข้อมูลมาจากเพจที่ข้อความนั้นเชื่อมต่อไปถึงดังแสดงในรูป 7-18(b) ข้อความที่ขีดเส้นใต้สามารถที่จะเชื่อมต่อไปยังเว็บไซต์อื่นๆ ได้ ข้อความในแต่ละเว็บไซต์อาจจะถูกนำมาแสดงในเครื่องเดียวกันกับที่เว็บไซต์นั้นตั้งอยู่หรืออาจจะอยู่ห่างกันคนละซีกโลกก็ได้ซึ่งผู้ใช้จะไม่มีทางทราบได้เลย การดึงเพจใหม่ๆ ขึ้นมานั้นเป็นหน้าที่ของบราวเซอร์โดยที่ผู้ใช้ไม่ต้องเข้าไปช่วยเหลือเลย



รูป 7-19
ส่วนหนึ่งของรูปแบบการทำงานของเว็บ

รูปแบบการทำงานของเว็บแสดงให้เห็นในรูป 7-19 ในที่นี้ บราวเซอร์กำลังแสดงเว็บเพจบนเครื่องผู้ใช้ เมื่อผู้ใช้คลิ็กบนข้อความที่เชื่อมต่อไปยังเพจ abcd.com โปรแกรมบราวเซอร์จะตามการเชื่อมต่อนั้นไปโดยส่งข้อความไปบอก abcd.com server เพื่อขอข้อมูลในเพจมาแสดง เมื่อข้อมูลในเพจนั้นเดินทางมาถึงบราวเซอร์ก็จะจัดการแสดงบนหน้าจอภาพ ถ้าเพจนั้นมี hyperlink เชื่อมต่อไปยังเพจอื่นบน xyz.com ซึ่งผู้ใช้ทำการคลิ็กบนข้อความนั้น บราวเซอร์ก็จะส่งความต้องการไปยังเซิบเวอร์นั้นเพื่อขอข้อมูลในเพจมาแสดงให้ต่อไป

การทำงานทางฝั่งผู้ใช้บริการ

ต่อไปดูการทำงานที่เกิดขึ้นทางฝั่งผู้ใช้ที่เกิดขึ้นในรูป 7-19 โดยความจริงแล้วโปรแกรมบราวเซอร์คือโปรแกรมที่สามารถแสดงเว็บเพจและติดตามการเคลื่อนที่ของเมาส์ได้ตลอดเวลา เมื่อข้อความหนึ่งบนเว็บถูกเลือกบราวเซอร์จะติดตามการเชื่อมต่อนั้นไปยังเป้าหมายและนำเพจนั้นขึ้นมาแสดง ดังนั้น hyperlink จะต้องมีวิธีการในการกำหนดชื่อเพจอื่นบนเว็บได้ ชื่อเพจจะถูกอ้างอิงถึงโดยใช้ URLs (Uniform Resource Locators) ดังตัวอย่าง

http://www.abcd.com/products.html

ชื่อ URL แบ่งออกเป็นสามส่วนคือ ชื่อโพรโตคอล (http) ชื่อ DNS ของเครื่องเซิบเวอร์ที่เก็บเพจนั้นไว้ (www.abcd.com) และตามด้วยชื่อไฟล์ที่เก็บเพจนั้นไว้ (products.html)

เมื่อผู้ใช้คลิ็กที่ hyperlink บราวเซอร์จะทำงานหลายขั้นตอนเพื่อที่จะนำเพจที่ถูกอ้างอิงนั้นขึ้นมาแสดง สมมุติว่าผู้ใช้ทำการดูที่เพจและพบการเชื่อมต่อไปยัง Internet telephony ที่ชี้ไปยังโฮมเพจของ ITU ซึ่งก็คือ http://www.itu.org/home/index.html ขั้นตอนการทำงานที่เกิดเป็นดังนี้

1. บราวเซอร์ตรวจดู URL ที่ถูกเลือก
2. บราวเซอร์ร้องขอหมายเลข IP ของ www.itu.org ไปยัง DNS ของตนเอง

3. DNS ตอบมาด้วยหมายเลข 156.106.192.32

4. บราวเซอร์จัดตั้งการเชื่อมต่อ TCP ไปยังพอร์ต 80 บนเครื่อง 156.106.192.32

5. บราวเซอร์ส่งคำร้องขอไฟล์ /home/index.html

6. เซิบเวอร์ของ www.itu.org ส่งไฟล์ /home/index.html มาให้

7. ยกเลิกการเชื่อมต่อ TCP

8. บราวเซอร์แสดงข้อความที่เป็นตัวอักษรที่อยู่ในไฟล์ /home/index.html

9. บราวเซอร์แสดงรูปภาพที่อยู่ในไฟล์ /home/index.html

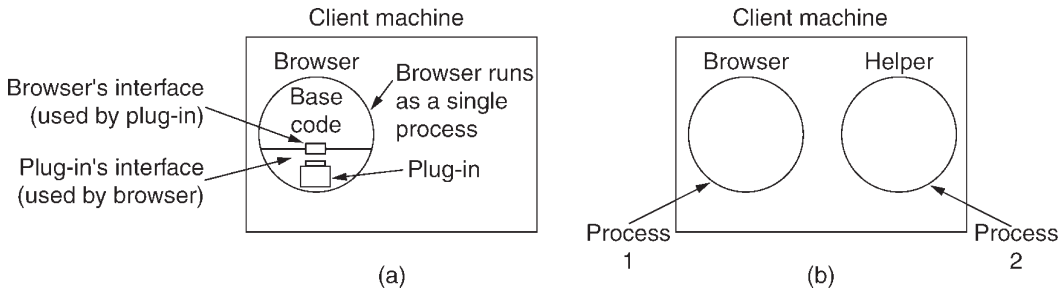
บราวเซอร์จำนวนหนึ่งจะแสดงการทำงานที่เกิดขึ้นทีละขั้นตอนไว้ที่บรรทัดแสดงสถานะการทำงานที่ตอนล่างสุดของจอภาพ ด้วยวิธีการนี้ ถ้าประสิทธิภาพการแสดงผลนั้นอยู่ในระดับต่ำ ผู้ใช้จะสามารถทราบได้ว่าต้นเหตุนั้นเกิดมาจากการที่ DNS ไม่ตอบสนอง หรือเซิบเวอร์ไม่ตอบสนอง หรือเนื่องมาจากความคับคั่งที่เกิดขึ้นบนระบบเครือข่ายในการถ่ายทอดข้อมูลแต่ละเพจ

เพื่อให้สามารถแสดงเพจใหม่ บราวเซอร์จะต้องเข้าใจรูปแบบของข้อมูลนั้นๆ เพื่อให้บราวเซอร์ทั้งหมดเข้าใจเว็บเพจทุกเพจ เว็บเพจจึงถูกเขียนขึ้นมาด้วยภาษามาตรฐานเรียกว่า HTML (Hypertext Markup Language)

แม้ว่าบราวเซอร์จะทำหน้าที่เป็นเพียงตัวแปลความหมาย HTML (interpreter) แต่บราวเซอร์ส่วนใหญ่มีคุณสมบัติจำนวนมากในการทำให้การแสดงเว็บเพจต่างๆ นั้นเป็นไปได้โดยง่าย เช่น มีปุ่มสำหรับคลิ๊กเพื่อย้อนกลับไปดูเว็บเพจที่ดูผ่านมาแล้ว หรือดูเพจต่อไป หรือปุ่มที่ไปยังเพจเริ่มต้น เป็นต้น นอกจากนี้บราวเซอร์ส่วนใหญ่ยังมีเมนูสำหรับ bookmark เพื่อให้ง่ายต่อการจดจำและสามารถย้อนกลับไปดูเพจนั้นได้โดยตรง เพจยังสามารถบันทึกลงดิสก์หรือพิมพ์ออกทางเครื่องพิมพ์ บราวเซอร์ยังมีขีดความสามารถอื่นๆ ในการควบคุมการแสดงผลบนหน้าจอภาพและกำหนดความต้องการเฉพาะของผู้ใช้แต่ละคนได้

นอกเหนือจากการแสดงผลข้อความที่เป็นตัวอักษรธรรมดาและข้อความที่เป็น hypertext แล้ว เว็บเพจยังมีไอคอน การวาดเส้น การแสดงแผนที่ การแสดงรูปถ่าย และอื่นๆ ขีดความสามารถแต่ละอย่างสามารถที่จะนำมาใช้ในการเชื่อมต่อไปยังเพจอื่นได้เช่นเดียวกับ hypertext และนอกจากการใช้ภาษา HTML แล้วในเพจยังอาจมีข้อมูลที่เป็นรูปแบบ PDF ไอคอนที่เป็นรูปแบบ GIF รูปภาพในรูปแบบ JPEG เพลงในรูปแบบ MP3 วีดิทัศน์ในรูปแบบ MPEG และรูปแบบอื่นอีกนับร้อยอย่าง เนื่องจากการเชื่อมต่ออาจนำไปสู่การเชื่อมต่อเข้ากับข้อมูลพิเศษเหล่านี้ ดังนั้นบราวเซอร์อาจมีปัญหาในการแสดงผลได้

แทนที่จะขยายขีดความสามารถของบราวเซอร์ให้สามารถแสดงผลข้อมูลประเภทต่างๆ เหล่านี้ได้โดยตรงซึ่งจะทำให้โปรแกรมบราวเซอร์มีขนาดใหญ่ขึ้นเรื่อยๆ บราวเซอร์ส่วนใหญ่จะเลือกใช้วิธีการที่ง่ายกว่านี้ กล่าวคือ เมื่อเซิบเวอร์ส่งคืนเพจกลับมาเซิบเวอร์จะส่งข้อมูลเกี่ยวกับเพจนั้นกลับมาด้วย ข้อมูลนี้อยู่ในรูปของ MIME ซึ่งจะบอกให้ทราบถึงชนิดของข้อมูลที่ส่งมา ดังที่แสดงในรูป 7-12 เพจชนิด text/html จะสามารถนำไปแสดงผลโดยบราวเซอร์ได้โดยตรงรวมทั้งเพจชนิดอื่นบางชนิดที่บราวเซอร์สามารถจัดการได้เอง ถ้า MIME บอกชนิดของข้อมูลที่บราวเซอร์ไม่รู้จัก บราวเซอร์จะขอข้อมูลจากตาราง MIME type ซึ่งจะบอกให้ทราบว่าจะใช้โปรแกรมใดในการนำข้อมูลนั้นๆ ไปแสดงผล



รูป 7-20
(a) A plug-in
(b) A helper

วิธีการที่เป็นไปได้สองหนทางคือ การใช้ plug-in และ helper Plug-in เป็นโปรแกรมส่วนหนึ่งที่บราวเซอร์นำมาจากไดเรกทอรี่พิเศษที่อยู่บนดิสก์และทำการติดตั้งเป็นส่วนขยายเพิ่มเติมให้แก่บราวเซอร์นั้นๆ ดังแสดงในรูป 7-20(a) เนื่องจาก plug-in ทำงานอยู่ในตัวบราวเซอร์เองจึงอาจจะทำการติดต่อกับข้อมูลในเพจนั้นและสามารถแก้ไขเปลี่ยนแปลงลักษณะภาพที่ปรากฏอยู่ได้ เมื่อ plug-in ได้ทำหน้าที่เสร็จแล้วโปรแกรม plug-in ก็จะถูกลบออกจากหน่วยความจำของบราวเซอร์

อีกวิธีการหนึ่งที่จะขยายขีดความสามารถของบราวเซอร์คือการใช้ helper application ซึ่งเป็นโปรแกรมที่แยกเป็นอิสระต่างหากจากบราวเซอร์และเมื่อทำการประมวลผลโปรแกรมนี้ก็จะแยกเป็นโปรเซสคนละโปรเซสกับบราวเซอร์ ดังที่แสดงในรูป 7-20(b) เนื่องจาก helper เป็นโปรแกรมแยกต่างหากจึงไม่สามารถติดต่อกับบราวเซอร์และใช้บริการต่างๆ ของบราวเซอร์ได้ โดยปกติ helper จะรับชื่อไฟล์ที่เก็บรวบรวมข้อมูลเอาไว้ทำการเปิดใช้ไฟล์นั้น และนำข้อมูลที่อยู่ภายในไฟล์มาแสดงให้เห็น โดยปกติ helper จะเป็นโปรแกรมอิสระที่มีขนาดใหญ่ เช่น Adobes Acrobat Reader ซึ่งใช้ในการแสดงข้อมูลในไฟล์ประเภท PDF หรือ ไมโครซอฟต์เวิร์ด

โปรแกรม helper บางส่วนจะใช้ MIME type application เช่น application/pdf สำหรับไฟล์ประเภท PDF และ application/msword สำหรับไฟล์ประเภท word ด้วยวิธีการนี้ URL จะสามารถชี้ไปโดยตรงได้ที่ไฟล์ PDF หรือ word และเมื่อผู้ใช้คลิ๊กที่ข้อความนี้ โปรแกรม Acrobat หรือ Word ก็จะเริ่มต้นทำงานได้โดยอัตโนมัติ ผลจากการใช้วิธีนี้ทำให้บราวเซอร์สามารถจัดการแสดงผลไฟล์ได้แทบจะทุกชนิดโดยไม่ต้องการแก้ไขเปลี่ยนแปลงที่ตัวบราวเซอร์เลย

Helper application ไม่จำเป็นต้องใช้ MIME application type เช่น Adobe Photoshop ใช้ image/x-photoshop และ RealOne Player ซึ่งใช้ในการเล่นเพลงประเภท MP3 จะใช้ type เป็น audio/mp3 เป็นต้น

บราวเซอร์สามารถที่จะเรียกใช้ไฟล์ที่เก็บอยู่ในเครื่องผู้ใช้ได้ แทนที่จะเรียกใช้จากเว็บไซต์อื่น เนื่องจากไฟล์ที่เก็บอยู่ในเครื่องผู้ใช้จะไม่ใช้ MIME type บราวเซอร์จึงจะต้องหาวิธีในการเรียกใช้ plug-in หรือ helper ให้ได้ เพื่อจัดการกับไฟล์ภายใน Helper สามารถถูกเรียกใช้โดยอาศัยข้อมูลจาก file extension รวมทั้ง MIME type โดยเฉพาะ Internet explorer จะอาศัยข้อมูลจาก file extension (เช่น .pdf, .doc เป็นต้น) ในการเรียกใช้ helper มากกว่าการใช้ MIME type

การทำงานทางฝั่งผู้ให้บริการ

ดังที่ได้กล่าวข้างต้น เมื่อผู้ใช้ได้ป้อนที่อยู่ URL หรือคลิ๊กที่ข้อความที่เป็น hypertext บราวเซอร์จะตรวจดูที่อยู่ URL นั้นและทำการแปลงความหมายระหว่าง "http://" กับเครื่องหมาย "/" อันต่อไป

ซึ่งเป็นชื่อ DNS มาทำการหาที่อยู่ที่เป็นหมายเลข IP จากนั้นจะทำการจัดตั้งช่องสื่อสาร TCP เข้ากับพอร์ต 80 ทางฝั่งเซิบเวอร์ และจัดการส่งคำสั่งส่วนที่เหลือใน URL ซึ่งมักจะเป็นชื่อไฟล์ของทางฝั่งเซิบเวอร์ เซิบเวอร์จะส่งไฟล์นั้นคืนไปเพื่อการแสดงบนหน้าจองผู้ใช้

เว็บเซิบเวอร์ก็ทำหน้าที่คล้ายเครื่องเซิบเวอร์ที่แสดงในรูป6-6ในทั้งสองกรณีมีขั้นตอนการทำงานดังนี้

1. ยอมรับการจัดตั้งการเชื่อมต่อ TCP จากผู้ใช้ (บราวเซอร์)
2. อ่านชื่อไฟล์ที่ผู้ใช้ต้องการ
3. อ่านข้อมูลในไฟล์ขึ้นมาจากดิสก์
4. ส่งไฟล์คืนไปที่ผู้ใช้
5. ยกเลิกการเชื่อมต่อ TCP

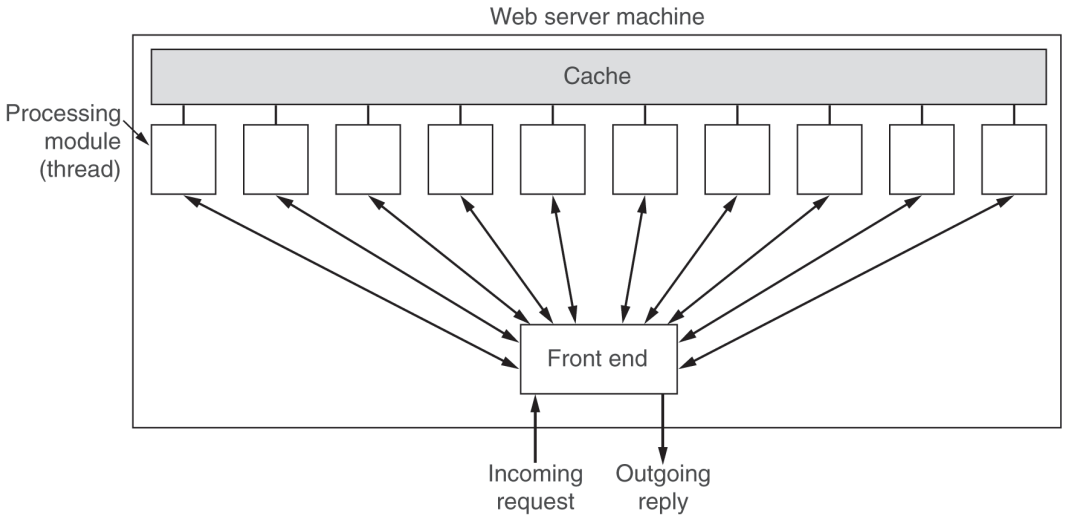
แม้ว่าเว็บเซิบเวอร์สมัยใหม่จะมีขีดความสามารถในการทำงานมากขึ้น แต่โดยพื้นฐานแล้วนี่คือขั้นตอนที่เว็บเซิบเวอร์ทำงาน

ปัญหาของการออกแบบการทำงานวิธีนี้ก็คือคำร้องขอทุกครั้งจะทำให้เกิดการติดต่อขอข้อมูลจากดิสก์ ทำให้เซิบเวอร์ไม่สามารถให้บริการแก่ผู้ใช้ต่อวินาทีได้มากกว่าจำนวนครั้งที่สามารถติดต่อกับดิสก์ได้ต่อวินาที ดิสก์ที่มีขีดความสามารถสูงเช่น SCSI ใช้ระยะเวลาในการติดต่อดึงข้อมูลเฉลี่ยประมาณ 5 มิลลิวินาที ทำให้ไม่สามารถให้บริการแก่เซิบเวอร์ได้มากกว่า 200 คำร้องขอต่อวินาที และอาจจะน้อยกว่านี้ถ้าผู้ใช้ร้องของไฟล์ขนาดใหญ่ สำหรับเว็บไซต์ขนาดใหญ่แล้ว อัตราการให้บริการนี้ถือว่าต่ำมาก

วิธีการปรับปรุงที่ได้ผลวิธีหนึ่งที่ใช้โดยเว็บเซิบเวอร์คือการรักษาข้อมูลไฟล์ที่ถูกเรียกใช้ล่าสุดจำนวน n ไฟล์ไว้ในหน่วยความจำ cache และก่อนที่จะไปค้นหาไฟล์ข้อมูลในดิสก์ให้เซิบเวอร์ทำการเช็คไฟล์ข้อมูลที่เก็บไว้ใน cache ก่อน ถ้าพบไฟล์ที่ต้องการก็จะสามารถจัดส่งไฟล์นั้นไปได้โดยตรงโดยไม่ต้องติดต่อกับดิสก์ (ซึ่งเสียเวลามาก) แม้ว่าการใช้ cache ที่มีประสิทธิผลที่ดีจะต้องใช้เนื้อที่ในหน่วยความจำเป็นจำนวนมากและจะต้องใช้เวลาในการประมวลผลมากในการตรวจสอบข้อมูลใน cache และการบริหารจัดการ แต่การประหยัดเวลาที่ต้องเสียไปนั้นให้ผลดีเกินกว่าสิ่งที่จะต้องลงทุนเพิ่มเติม

ขั้นต่อไปในการสร้างเซิบเวอร์ที่สามารถให้บริการได้อย่างรวดเร็วก็คือการทำให้เซิบเวอร์ทำงานแบบ multithread ในการออกแบบวิธีหนึ่ง เซิบเวอร์ประกอบด้วย front-end module ที่คอยรับคำร้องขอที่ถูกส่งเข้ามาทั้งหมด ดังที่แสดงในรูป 7-21 Thread ทั้งหมดเป็นส่วนหนึ่งของโปรเซสอันเดียวกันทำให้ thread ที่กำลังทำงานอยู่นั้นสามารถติดต่อกับ cache ตัวเดียวกันที่อยู่ภายในเนื้อที่การทำงาน (address space) ของโปรเซสนั้นได้ เมื่อมีคำร้องขอเข้ามา front-end module จะรับไว้และสร้าง record สั้นๆ เพื่ออธิบายความต้องการนั้น จากนั้นก็จะส่ง record ต่อไปให้กับ thread ที่กำลังทำงานอยู่ ในการออกแบบอีกวิธีหนึ่ง front-end module จะไม่ถูกนำมาใช้ แต่จะให้แต่ละ thread พยายามที่จะติดต่อรับคำร้องเอาเอง ซึ่งก็จะต้องมีการสร้างโพรโตคอลอันหนึ่งที่คอยกันไม่ให้เกิดการแย่งชิงคำร้องระหว่าง thread ต่างๆ

Thread หรือเรียกว่า processing module จะตรวจสอบใน cache ว่ามีไฟล์ที่ต้องการอยู่หรือไม่ ถ้ามีอยู่ก็จะทำการปรับปรุงข้อมูลใน record เพื่อใส่ตำแหน่งของไฟล์นั้นเข้าไป แต่ถ้าไม่มีอยู่ thread



รูป 7-21
A multithread Web server with a front end and processing modules

ก็จะทำการติดต่อกับดิสก์ในการอ่านไฟล์นั้นเข้ามาเก็บไว้ใน cache (อาจจะนำเข้ามาแทนที่ไฟล์เก่าบางไฟล์ที่ไม่ได้ใช้งานมานานแล้ว) เมื่อไฟล์นั้นเข้ามาเรียบร้อยแล้วก็จะส่งไฟล์นั้นกลับไปยังผู้ใช้บริการที่ร้องขอมา

ข้อได้เปรียบของวิธีการนี้ก็คือ ในขณะที่ thread บางส่วนไม่สามารถทำงานต่อไปได้เนื่องจากต้องรอรับไฟล์ที่กำลังถูกอ่านเข้ามา ในเวลาเดียวกัน thread ส่วนที่เหลือก็สามารถให้บริการแก่คำร้องขออื่นๆ ได้ แน่นอนว่าเพื่อให้ประสิทธิภาพของ multithreaded ดีกว่าการใช้ single-threaded นั้น จำเป็นจะต้องใช้ดิสก์หลายตัว เพื่อทำให้ดิสก์หลายตัวสามารถทำงานพร้อมกันได้ ถ้ามี thread จำนวน k therad และมีดิสก์จำนวน k ตัวแล้วผลลัพธ์ที่เกิดขึ้นจากการทำงานสามารถที่จะมีมากขึ้นเป็น k เท่าได้เมื่อเปรียบเทียบกับ single-threaded server ที่มีดิสก์เพียงตัวเดียว

ในทางทฤษฎีแล้ว single-threaded server และมีดิสก์ k ตัวก็สามารถที่จะมีประสิทธิภาพเป็น k เท่าได้เหมือนกันแต่การเขียนโปรแกรมจะต้องยุ่งยากซับซ้อนมากเนื่องจากการใช้คำสั่ง READ เพื่ออ่านข้อมูลจากดิสก์เข้ามานั้นจะไม่สามารถใช้งานได้ แต่เมื่อใช้ multithreaded server โปรแกรมจะสามารถใช้คำสั่ง READ ธรรมดาได้เพราะการรอผลจากคำสั่งนี้จะมีผลกับ thread ที่เป็นผู้ออกคำสั่งเท่านั้น ทำให้ thread อื่นสามารถทำงานต่อไปได้

เว็บเซิบเวอร์สมัยใหม่ทำมากกว่าการยอมรับชื่อไฟล์เข้ามาและส่งไฟล์คืนกลับไป อันที่จริงกระบวนการทำงานที่เกิดขึ้นจริงของคำร้องแต่ละอันนั้นอาจจะมีความซับซ้อนมากก็ได้ ด้วยเหตุผลนี้ แต่ละ thread ในเซิบเวอร์จึงแบ่งการทำงานออกเป็นหลายขั้นตอน Front-end จะส่งคำร้องเข้ามายัง thread ที่กำลังว่างงานซึ่งก็จะทำงานตามขั้นตอนเหล่านี้ การทำงานบางขั้นตอนจะขึ้นอยู่กับรายละเอียดของคำร้องที่ส่งเข้ามา

1. จัดการแก้ปัญหาชื่อของเว็บเพจที่ร้องขอเข้ามา
2. ตรวจสอบผู้ใช้
3. ตรวจสอบการควบคุมการเข้าใช้ข้อมูลของผู้ใช้

4. ตรวจสอบการควบคุมการเข้าใช้ข้อมูลของเว็บเพจ

5. ตรวจสอบ cache

6. ดึงไฟล์ข้อมูลที่ต้องการมาจากดิสก์

7. ตรวจสอบ MIME type ที่จะต้องส่งไปพร้อมกับไฟล์นั้น

8. จัดการแก้ปัญหาต่าง ๆ ที่อาจเกิดขึ้น

9. จัดการส่งคำตอบไปยังผู้ใช้

10. จัดการบันทึกการใช้งานไว้ใน server log

ขั้นตอนที่ 1 นั้นมีความจำเป็นเพราะคำร้องที่รับเข้ามานั้นอาจจะไม่มีชื่อของไฟล์ในรูปแบบของตัวอักษร เช่น URL ที่รับเข้ามาคือ http://www.cs.vu.nl ซึ่งไม่มีชื่อไฟล์อยู่ด้วยเลย ในที่นี้ก็จำเป็นจะต้องนำค่าสำรอง (default) มาใช้ซึ่งเป็นชื่อไฟล์ชื่อหนึ่ง บราวเซอร์สมัยใหม่ยอมให้มีการกำหนดภาษาที่ใช้สำหรับบราวเซอร์นั้น ๆ (เช่น ภาษาไทย หรือภาษอังกฤษ) ซึ่งมีความเป็นไปได้ที่เซิบเวอร์จะเลือกเว็บเพจที่สร้างขึ้นมาโดยใช้ภาษานั้น โดยทั่วไปการใช้ชื่อที่สร้างขึ้นมาจากชื่อสำรองนั้นไม่ใช่กระบวนการที่ง่ายเลย เนื่องจากมีความหลากหลายและมีข้อเลือกมาก

ขั้นตอนที่ 2 คือขั้นตอนที่ทำการตรวจสอบตัวตนของผู้ใช้ ขั้นตอนนี้มีความจำเป็นเนื่องจากข้อมูลบางเพจก็ไม่อนุญาตให้คนทั่วไปเข้ามาดูได้

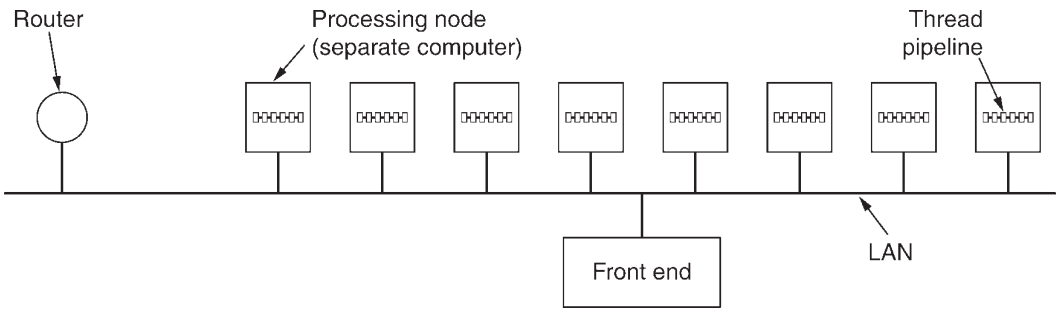
ขั้นตอนที่ 3 ตรวจสอบดูว่ามีข้อบังคับใด ๆ หรือไม่ที่ผู้ใช้ซึ่งได้แสดงตัวตนในขั้นที่ 2 แล้วและที่อยู่ของผู้ใช้ที่ทราบได้จากหมายเลข IP จะต้องกระทำเพื่อให้สามารถดูข้อความในเพจนั้นได้

ขั้นตอนที่ 4 ตรวจสอบดูว่ามีการป้องกันใด ๆ ที่เกิดขึ้นกับข้อความในเพจนั้นเองหรือไม่ เช่นไฟล์ที่อนุญาตให้บุคลากร ภายในองค์กรเท่านั้นที่จะสามารถดูได้

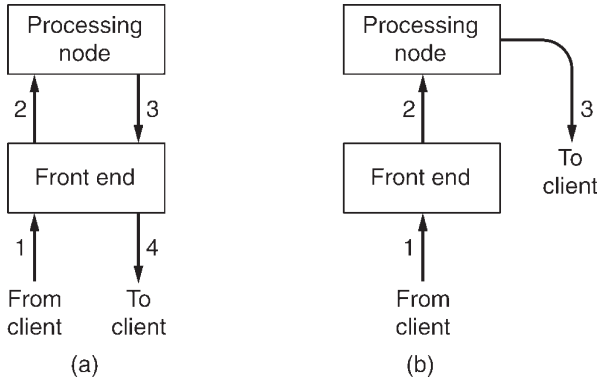
ขั้นตอนที่ 5 และ 6 เกี่ยวกับการให้ได้มาซึ่งเพจนั้น ๆ

ขั้นตอนที่ 7 เกี่ยวกับการตรวจสอบ MIME type จาก file extension หรือประเภทของไฟล์ หรือจากข้อความตอนต้นของไฟล์ (ใช้ในระบบยูนิกส์) หรือจากแหล่งข้อมูลอื่น

ขั้นตอนที่ 8 นั้นใช้สำหรับการทำงานอื่น ๆ ที่อาจเข้ามาเกี่ยวข้อง เช่น การสร้างรูปแบบการใช้งาน (profile) ของผู้ใช้ หรือเก็บข้อมูลทางสถิติ



รูป 7-22
A server farm



รูป 7-23
(a) ขั้นตอนการส่งคำร้องและการตอบสนองตามปกติ
(b) การใช้เทคนิค TCP handoff

ขั้นตอนที่ 9 เป็นการจัดส่งไฟล์ที่ได้รับกลับไปยังผู้ใช้ที่ร้องขอเข้ามา และ**ขั้นตอนที่** 10 เป็นการบันทึกรายละเอียดในการทำงานที่เกิดขึ้นซึ่งมีไว้ใช้เป็นข้อมูลในการบริหารจัดการเว็บไซต์ข้อมูลนี้อาจกลายเป็นข้อมูลที่มีความสำคัญเป็นอย่างมากในการดูพฤติกรรมของผู้ใช้ เช่น ลำดับของเพจที่ผู้ใช้เรียกดู

ถ้ามีคำร้องขอเข้ามาเป็นจำนวนมากเกินกว่าที่จะรับได้ในแต่ละวินาที ซีพียูจะไม่สามารถรองรับการทำงานนี้ได้ไม่ว่าจะมีจำนวนดิสก์อยู่เท่าใดก็ตามที่สามารถทำงานได้พร้อมๆ กัน หนทางแก้ปัญหาคือการเพิ่มเครื่องคอมพิวเตอร์เข้าไปในระบบและอาจจะมีดิสก์ที่เก็บข้อมูลซ้ำซ้อนกันอยู่ด้วยเพื่อหลีกเลี่ยงปัญหาการแย่งกันใช้ข้อมูลในดิสก์ตัวเดียวกัน วิธีการนี้นำไปสู่การจัดการแบบที่เรียกว่า "server farm" ดังแสดงในรูป 7-22 Front end ยังคงทำหน้าที่ในการรับคำร้องจากผู้ใช้เข้ามา แต่จะกระจายคำร้องเหล่านั้นไปยังคอมพิวเตอร์หลายเครื่อง แทนที่จะเป็นหลายๆ thread ทั้งนี้เพื่อลดปริมาณงานให้กับคอมพิวเตอร์และเครื่อง คอมพิวเตอร์แต่ละเครื่องก็อาจทำงานในแบบ multithread server ก็ได้

ปัญหาหนึ่งที่เกิดขึ้นกับ server farm ก็คือจะไม่มีการใช้ cache ร่วมกันเนื่องจากคอมพิวเตอร์แต่ละตัวนั้นมีหน่วยความจำเป็นของตนเอง นอกจากจะใช้หน่วยความจำร่วมกันระหว่างซีพียูทั้งหลายซึ่งเป็นวิธีการที่เสียค่าใช้จ่ายแพงมาก วิธีการหนึ่งที่นำมาช่วยแก้ปัญหาประสิทธิภาพที่ตกลงไปคือการให้ front end คอยติดตามดูว่าได้ส่งคำร้องแต่ละอันไปที่ใดและเมื่อมีคำร้องที่มาจากที่เดิมก็ให้ส่งคำร้องนั้นไปยังซีพียูตัวเดิม การทำเช่นนี้จะทำให้คอมพิวเตอร์แต่ละเครื่องมีความชำนาญในแต่ละเพจทำให้ไม่ต้องเปลืองเนื้อที่ใน cache (ของซีพียูแต่ละตัว) เพื่อเก็บข้อมูลชุดเดียวกัน

อีกปัญหาหนึ่งของ server farm คือการเชื่อมต่อ TCP นั้นจะสิ้นสุดลงที่ front end ทำให้การส่งข้อมูลกลับไปยังผู้ใช้นั้นจะต้องทำผ่าน front end สถานะการณ์เช่นนี้แสดงให้เห็นในรูป 7-23(a) ที่ซึ่งคำร้องขอ (1) และคำตอบ (4) จะต้องส่งผ่าน front end ทั้งคู่ บางครั้งก็มีการใช้เทคนิคเรียกว่า TCP handoff เพื่อแก้ปัญหานี้ เทคนิคนี้จะส่งผ่านจุดสิ้นสุดการเชื่อมต่อ TCP ไปยังซีพียูที่ทำการตอบสนองต่อคำร้องนั้นๆทำให้มันสามารถส่งคำตอบกลับไปยังผู้ใช้ได้โดยตรง ดังแสดงในขั้นตอนที่ (3) ของรูป 7-23(b) การทำงานเช่นนี้จะไม่เกี่ยวข้องกับผู้ใช้เลย

URLs-Uniform Resource Locators

ดังที่ได้กล่าวมาแล้วว่าเว็บเพจอาจจะมีการเชื่อมต่อไปยังเว็บเพจอื่นๆ ได้ การเชื่อมต่อจะต้องอาศัยกลไกสำหรับการตั้งชื่อและการหาตำแหน่งของเว็บเพจ โดยทั่วไปจำเป็นจะต้องตอบคำถามสามข้อให้ได้ก่อนที่เว็บเพจที่ถูกเลือกจะถูกนำมาแสดง คือ

1. เพจนั้นเรียกว่าอะไร

2. เพจนั้นอยู่ที่ไหน

3. จะใช้งานเพจนั้นได้อย่างไร

ถ้าหากว่าทุกเพจมีชื่อเฉพาะเป็นของตนเองก็จะไม่มีปัญหาความคลุมเครือเกิดขึ้นในระหว่างการเรียกใช้เพจต่างๆ แต่อย่างไรก็ตาม ปัญหาก็ยังไม่หมดไป ในสหรัฐอเมริกาเกือบจะทุกคนจะต้องมีหมายเลขประจำตัวเองซึ่งเป็นหมายเลขเฉพาะที่ไม่ซ้ำกัน อย่างไรก็ตามถ้าคนมีเพียงหมายเลขประจำตัวแล้ว ก็จะไม่มีทางที่จะทราบว่าเขาคนนั้นอยู่ที่ไหนและไม่มีทางทราบว่าจะต้องใช้ภาษาใดในการสื่อสารกับเขา เว็บเพจก็มีปัญหาลักษณะเดียวกัน

หนทางแก้ปัญหาที่นำมาใช้กับเว็บเพจที่สามารถตอบคำถามได้ทั้งสามข้อคือการใช้ URL (Uniform Resource Locator) ซึ่งประกอบด้วยข้อมูลสามส่วนคือ โพรโตคอล, ชื่อ DNS ของเครื่องที่เก็บเว็บเพจนั้นไว้, และชื่อของเว็บเพจ (โดยปกติจะเป็นชื่อไฟล์ที่เก็บอยู่ในเครื่องนั้น) ตัวอย่างของเว็บไซต์สำหรับคณะคอมพิวเตอร์ที่ผู้เขียนทำงานอยู่ซึ่งมีวีดิทัศน์ของมหาวิทยาลัยอยู่จำนวนหนึ่งได้แก่

http://www.cs.vu.nl/video/index-en.html

ชื่อ URL นี้ประกอบด้วยสามส่วนคือ โพรโตคอล (http) ชื่อ DNS สำหรับโฮส (www.cs.vu.nl) และชื่อไฟล์ (video/index-en.html) ชื่อไฟล์จะถูกเก็บอยู่ในไดเร็กทอรี่ภายใต้ไดเร็กทอรี่ที่เก็บเว็บไซต์นั้น

เว็บไซต์จำนวนหนึ่งได้มีการสร้างชื่อที่เป็นทางลัด (shortcut) สำหรับชื่อไฟล์ เช่นการที่ไม่ได้ระบุชื่อไฟล์ของเว็บเพจที่ต้องการดูนั้นจะหมายถึงเว็บไซต์หลักขององค์กร (main homepage) โดยปกติเมื่อชื่อไฟล์เป็นเพียงชื่อไดเร็กทอรี่จะหมายถึงไฟล์ที่ชื่อ "index.html" และชื่อ "~user/" จะหมายถึงไดเร็กทอรี่ www และไฟล์ชื่อ index.html ในไดเร็กทอรี่นั้น เช่น โฮมเพจของผู้แต่คือ

http://www.cs.vu.nl/~ast/

แม้ว่าชื่อไฟล์ที่แท้จริงคือ index.html ที่อยู่ในไดเรกทอรี่นี้

ลำดับต่อไปหันมาพิจารณาว่า hypertext นั้นทำงานอย่างไร ในการสร้างข้อความ hypertext ผู้ที่สร้างเพจนั้นจะต้องให้ข้อมูลสองส่วนคือ ส่วนของข้อความที่ต้องการให้เป็น hypertext กับที่อยู่ URL ของเพจที่ต้องการเชื่อมโยงไปถึง เมื่อข้อความได้ถูกเลือกแล้ว บราวเซอร์จะค้นหาชื่อโฮสจากชื่อ DNS เมื่อทราบหมายเลขที่อยู่ IP ของโฮสนั้นแล้วบราวเซอร์จะทำการเชื่อมต่อ TCP เข้ากับโฮสนั้น จากนั้นก็จะส่งชื่อไฟล์โดยใช้โพรโตคอลที่กำหนดผ่านการเชื่อมต่อไปยังโฮส และได้รับเว็บเพจที่ต้องการกลับคืนมาในที่สุด

กลไกการใช้ URL นี้เป็นกลไกที่เปิดกว้างทำให้บราวเซอร์สามารถเลือกใช้โพรโตคอลที่เหมาะสมในการเรียกใช้ทรัพยากรต่างๆ อันที่จริง URL สำหรับโพรโตคอลที่มีใช้งานทั่วไปแบบอื่นๆ ก็ได้รับการพัฒนาขึ้นมาเหมือนกัน รูป 7-24 แสดงรูปแบบของ URL แบบต่างๆ ที่มีใช้งาน

โพรโตคอล http เป็นมาตรฐานที่ใช้กับเว็บทั่วไป HTTP ย่อมาจากคำว่า HyperText Transfer Protocol

Name	Used for	Example
http	Hypertext (HTML)	http://www.cs.vu.nl/~ast/
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
file	Local file	file:///usr/suzanne/prog.c
news	Newsgroup	news:comp.os.minix
news	News article	news:AA0134223112@cs.utah.edu
gopher	Gopher	gopher://gopher.tc.umn.edu/11/Libraries
mailto	Sending e-mail	mailto:JohnUser@acm.org
telnet	Remote login	telnet://www.w3.org:80

รูป 7-24
URL แบบต่างๆ

ซึ่งจะได้กล่าวถึงรายละเอียดในหัวข้อถัดไป

โพรโตคอล ftp ใช้ในการเข้าถึงไฟล์ข้อมูลผ่านระบบ FTP (File Transfer Protocol) ระบบ FTP ได้รับการพัฒนาขึ้นมาใช้งานนานกว่า ยี่สิบปีมาแล้ว และยังคงสามารถใช้งานได้ดีอยู่ในปัจจุบัน ทั่วโลกมี FTP server อยู่มากมายที่ยอมให้ผู้ใช้ในระบบอินเตอร์เน็ตสามารถเข้าไปใช้ประโยชน์จากข้อมูลต่างๆ ภายในเว็บไซต์ได้อย่างเสรี ต่อมาเมื่อมีการใช้เว็บ FTP ก็ไม่ได้สูญหายไปไหนนอกจากการเปลี่ยนแปลงส่วนติดต่อผู้ใช้ให้มีความน่าใช้งานและสะดวกสบายมากยิ่งขึ้น ซึ่งแม้ว่าจะไม่อาจเทียบเท่าโพรโตคอล http ได้แต่ก็มีความสามารถในการทำงานสูงกว่า เช่น FTP ยอมให้ผู้ใช้จากเครื่อง A สามารถถ่ายทอดไฟล์จากเครื่อง B ไปยังเครื่อง C ได้

เว็บเพจอาจจะเชื่อมต่อเข้ากับไฟล์ในคอมพิวเตอร์ของเครื่องที่กำลังใช้งานอยู่ก็ได้ โดยการใช้โพรโตคอล "file" หรือในรูปแบบที่ง่ายกว่าก็เพียงแค่พิมพ์ชื่อไฟล์ที่ต้องการลงไปใน URL เท่านั้น การทำงานของโพรโตคอลนี้คล้ายกับการใช้โพรโตคอล FTP เพียงแต่ไม่จำเป็นต้องมีเซิบเวอร์มาคอยให้บริการเท่านั้น แต่ก็จะทำงานกับไฟล์ที่อยู่ในเครื่องที่กำลังใช้งานอยู่เท่านั้น ไม่สามารถเข้าไปอ้างอิงไฟล์ที่อยู่ในเครื่องอื่นได้

ก่อนที่จะมีระบบอินเตอร์เน็ตใช้งานนั้น มีระบบการแจ้งข่าวสารผ่านระบบเครือข่ายเรียกว่า USENET news system เป็นระบบที่ประกอบด้วยกลุ่มข่าวสาร (newsgroup) มากกว่า 30,000 แห่ง ซึ่งมีผู้ใช้หลายล้านคนทั่วโลก กลุ่มข่าวสารเป็นกลุ่มคนที่มาสนทนาแลกเปลี่ยนความคิดเห็นซึ่งกันและกันในหัวข้อที่น่าสนใจต่างๆ โพรโตคอล "news" ถูกนำมาใช้ในการเรียกดูไฟล์ข่าวสารหรือการสนทนาในลักษณะเดียวกันกับเว็บเพจ บราวเซอร์จึงทำหน้าที่เป็นโปรแกรมสำหรับการอ่านข่าวไปในตัว อันที่จริงแล้ว บราวเซอร์ต่างๆ มักจะมีไอคอนหรือปุ่มสำหรับกดเรียกหรือเมนูที่ช่วยให้การอ่านข่าวผ่าน USENET นั้นสะดวกและใช้งานได้ง่ายกว่าโปรแกรมสำหรับอ่านข่าวเองเสียอีก

รูปแบบที่ใช้ในการอ่านข่าวแบ่งออกเป็นสองแบบ รูปแบบแรกทำการกำหนดกลุ่มข่าวสารและสามารถ นำมาใช้เรียกดูรายการข่าวสารได้จาก news site ที่ได้มีการกำหนดไว้ล่วงหน้า รูปแบบที่สองจะต้องป้อนชื่อของรายการข่าวสารที่ต้องการ เช่น AA0134223112@cs.utah.edu จากนั้นบราวเซอร์จะรายงาน ข่าวสารที่ต้องการมาจาก news site ที่กำหนดไว้ล่วงหน้าโดยใช้ NNTP (Network News Transfer Protocol)

โพรโตคอล "gopher" เคยถูกนำมาใช้ในระบบ Gopher ซึ่งได้รับการออกแบบมาโดยมหาวิทยาลัย University of Minnesota และตั้งชื่อตามชื่อทีมกีฬาของมหาวิทยาลัยนี้คือ "Golden Gophers" ระบบโกเฟอร์นั้นตั้งขึ้นมาใช้งานก่อนระบบเว็บหลายปี โดยใช้เป็นระบบที่ช่วยค้นหาข้อมูลที่ผู้ใช้ต้องการ ซึ่งก็มีความคล้ายคลึงกับระบบเว็บนั่นเอง แต่สนับสนุนการทำงานในรูปแบบของตัวหนังสือเท่านั้น ในปัจจุบันนี้ได้ยกเลิกการใช้งานไปแล้ว

โพรโตคอลสองลำดับสุดท้ายไม่ได้นำมาใช้ในการค้นหาข้อมูลเว็บเพจแต่ก็ยังคงมีประโยชน์ในการใช้งานอยู่ โพรโตคอล "mailto" ช่วยให้ผู้ใช้สามารถส่งอีเมลล์ได้จากเว็บบราวเซอร์ วิธีการใช้งานคือการใช้เมาส์เลือกที่เมนู OPEN และกำหนด URL ที่ต้องการซึ่งประกอบด้วยคำว่า "mailto" และตามด้วยชื่อและที่อยู่ผู้รับอีเมลล์ บราวเซอร์ส่วนใหญ่จะตอบสนองโดยการเรียกโปรแกรมอีเมลล์ขึ้นมาทำงานต่อไป

โพรโตคอล telnet นั้นใช้ในการจัดตั้งการเชื่อมต่อแบบออนไลน์ (on-line) เข้ากับเครื่องคอมพิวเตอร์เครื่องอื่นซึ่งเป็นการทำงานเหมือนกับการใช้โปรแกรม telnet ธรรมดา เมื่อผู้ใช้เรียกใช้โปรแกรมนี้ในบราวเซอร์ ก็จะได้รับการตอบสนองโดยการที่บราวเซอร์จะไปเรียกใช้โปรแกรม telnet ในฐานะของ helper application นั่นเอง

กล่าวโดยสรุปก็คือ URL ได้รับการออกแบบมาทั้งการใช้งานแบบเว็บและมีความสามารถในการติดต่อกับโปรแกรมประยุกต์อื่นๆ (helper application) คือ FTP, news, Gopher, e-mail, และ telnet ได้ด้วย เพื่อทำการติดต่อกับโปรแกรมที่ถูกออกแบบมาให้ทำงานพิเศษเหล่านี้โดยมีเว็บบราวเซอร์เป็นตัวกลางในการติดต่อ

นอกเหนือจากคุณสมบัติที่ดีเด่นที่ได้กล่าวถึงไปแล้ว อัตราการเติบโตในการใช้บราวเซอร์ที่เพิ่มสูงขึ้นอย่างรวดเร็วได้ทำให้เกิดเป็นจุดอ่อนของ URL ขึ้นมา นั่นคือ URL จะชี้ไปที่โฮสใดโฮสหนึ่งเท่านั้น สำหรับเพจที่ถูกใช้งานอย่างหนักก็มีความจำเป็นที่จะต้องมีสำเนาเก็บไว้ที่เครื่องอื่นที่อยู่ไกลออกไปเพื่อเป็นการลดปริมาณการใช้งานระบบเครือข่ายให้น้อยลงในขณะที่สามารถให้บริการแก่ผู้ใช้ได้ดีขึ้น แต่ปัญหาก็คือ URL ไม่ได้มีวิธีการสำรองในการอ้างอิงถึงเพจที่มีการอ้างอิงถึงเป็นจำนวนมากเอาไว้ เช่น "ฉันต้องการเพจ XYZ โดยไม่สนใจว่าบราวเซอร์จะไปดึงเพจนั้นมาจากที่เว็บนั้นโดยตรงหรือจากเว็บสำรองใดๆ" เพื่อแก้ปัญหานี้ และทำให้สามารถใช้เว็บสำรองได้ องค์กร IETF จึงได้คิดค้นระบบใหม่ขึ้นมาคือ URN (Universal Resource Names) ซึ่งเป็นการทำให้ URL สามารถใช้งานได้กว้างขวางมากขึ้นกว่าเดิม (ยังอยู่ในขั้นการค้นคว้าวิจัย)

Statelessness และ Cookies

ดังที่ได้เห็นแล้ว เว็บบราวเซอร์ทำงานในลักษณะที่เรียกว่า "stateless" นั่นคือผู้ใช้ไม่มีความจำเป็นจะต้อง log-in แต่อย่างใด บราวเซอร์ส่งความต้องการไปยังเซิบเวอร์และได้รับไฟล์กลับมา จากนั้นเซิบเวอร์ก็จะลืมว่าได้เคยติดต่อกับผู้ใช้คนนั้นแล้วหรือไม่

แรกที่เดียวเมื่อเว็บได้ถูกใช้ในการดึงข้อมูลเอกสารที่สามารถเผยแพร่ต่อสาธารณะได้ รูปแบบการใช้งานนี้ก็ถือว่าใช้ได้แล้วแต่เมื่อเว็บเริ่มที่จะมีขีดความสามารถมากขึ้นก็ได้เริ่มก่อปัญหาขึ้นมา ตัวอย่างเช่น เว็บไซต์บางแห่งบังคับให้ผู้ใช้จะต้องทำการลงทะเบียน (และอาจจะต้องเสียค่าใช้บริการ) เพื่อที่จะติดต่อ

เข้าไปยังเว็บไซต์นั้น ซึ่งก็ได้สร้างปัญหาขึ้นมาว่าเซิบเวอร์จะแยกความแตกต่างระหว่างผู้ใช้ทั่วไปกับผู้ใช้ที่ลงทะเบียนแล้วได้อย่างไร ตัวอย่างที่สองมาจากการใช้ e-commerce นั่นคือ ถ้าผู้ใช้เที่ยวตระเวณไปตามร้านค้าอิเล็กทรอนิกส์ และเลือกซื้อสินค้าโดยการใส่รายการสินค้าที่ต้องการไว้ในตระกร้าอิเล็กทรอนิกส์บ้างเป็นครั้งคราว แล้วเซิบเวอร์จะสามารถติดตามของที่อยู่ในรายการสินค้าในตระกร้าอิเล็กทรอนิกส์ได้อย่างไร ตัวอย่างที่สามได้แก่การดัดแปลงเว็บให้มีลักษณะเฉพาะตามที่ผู้ใช้ต้องการ ผู้ใช้สามารถกำหนดรายละเอียดเพจเริ่มต้นให้แสดงเฉพาะข้อมูลที่ผู้ใช้แต่ละคนต้องการทราบ แต่ปัญหาก็คือเซิบเวอร์จะทราบได้อย่างไรว่าจะแสดงเพจไหนในเมื่อเซิบเวอร์ไม่ทราบว่ากำลังติดต่ออยู่กับใคร

ท่านอาจจะคิดว่าเซิบเวอร์สามารถทราบได้ว่ากำลังติดต่ออยู่กับผู้ใดโดยการดูที่หมายเลข IP แต่แนวความคิดนี้ใช้ไม่ได้ ประการแรก ผู้ใช้จำนวนมากที่ใช้งานคอมพิวเตอร์เครื่องเดียวกัน (แน่นอนว่าใช้กันคนละเวลา) โดยเฉพาะอย่างยิ่งกับเครื่องคอมพิวเตอร์ที่องค์กร ดังนั้นหมายเลข IP จึงใช้ในการแสดงตนของเครื่องคอมพิวเตอร์ไม่ใช่คนที่ใช้คอมพิวเตอร์ ประการที่สอง ที่แย่ยิ่งกว่านั้นก็คือ ISP จำนวนมากใช้เทคนิค NAT ในการให้บริการลูกค้าทำให้แพ็กเก็ตข้อมูลทั้งหมดที่ส่งออกมาจาก ISP นี้มีหมายเลข IP เบอร์เดียวกัน นั่นคือเซิบเวอร์จะมองเห็นว่าผู้ใช้นับพันรายต่างก็ใช้หมายเลข IP เบอร์เดียวกัน

เพื่อแก้ปัญหานี้ Netscape ได้คิดค้นวิธีการที่ได้รับการติเตียนอย่างมากมายขึ้นมาเรียกว่า "cookies" ชื่อนี้ได้นำมาจากภาษาแสลงของโปรแกรมเมอร์สมัยเก่า คือเมื่อโปรแกรมเรียกใช้โพรซีเยอร์และได้รับข้อมูลบางอย่างกลับคืนไปและข้อมูลนั้นอาจจะถูกนำมาใช้ในการทำงานบางอย่างในภายหลัง ในลักษณะเช่นนี้ Unix file descriptor และ Windows object handle สามารถเปรียบเทียบได้กับ cookie อย่างหนึ่ง อย่างไรก็ตาม cookie ได้รับการกำหนดเป็นมาตรฐาน RFC 2109 ในเวลาต่อมา

เมื่อผู้ใช้ต้องการข้อมูลจากเว็บเพจ เซิบเวอร์จะให้ข้อมูลเพิ่มเติมมาบางอย่างพร้อมกับเว็บเพจนั้น ข้อมูลนี้อาจจะรวมถึง cookie ซึ่งก็คือ ไฟล์หรือสายอักขระ (string) ขนาดเล็ก (ประมาณ 4 กิโลไบต์) บราวเซอร์จะจัดเก็บ cookie ไว้ในไดเร็กทอรี่ cookie ในฮาร์ดดิสก์ของเครื่องคอมพิวเตอร์ของผู้ใช้ (นอกจากว่าผู้ใช้จะเลือกที่จะไม่ให้ระบบ cookie ทำงาน-disable cookie) Cookie เป็นเพียงไฟล์หรือสายอักขระข้อมูลเท่านั้น โดยพื้นฐานแล้ว cookie อาจจะติดไวรัสด้วยก็ได้แต่เนื่องจาก cookie ได้รับการปฏิบัติในฐานะเป็นเพียงข้อมูลจึงไม่มีทาง (เท่าที่ทราบ) ที่ไวรัสนั้นจะลุกขึ้นมาทำอันตรายใดๆ ได้ อย่างไรก็ตาม มีทางเป็นไปได้ว่าอาจจะมี hacker บางคนที่คิดหาวิธีทำให้ไวรัสใน cookie ลุกขึ้นมาทำงานได้โดยอาศัยช่องโหว่ (bug) ของโปรแกรมบราวเซอร์ (ที่ยังมีอยู่มากมาย)

Cookie อาจประกอบด้วยข้อมูลมากถึง 5 เขตข้อมูลดังแสดงในรูป 7-25 เขตข้อมูล Domain บอกให้ทราบว่า cookie นั้นมาจากที่ใด บราวเซอร์มีหน้าที่ที่จะต้องตรวจสอบว่าเซิบเวอร์ไม่ได้โกหกเกี่ยวกับข้อมูลนี้ แต่ละโดเมนจะไม่ได้รับอนุญาตให้จัดเก็บ cookie ได้มากกว่า 20 รายการต่อผู้ใช้หนึ่งคน

Domain	Path	Content	Expires	Secure
toms-casino.com	/	CustomerID=497793521	15-10-02 17:00	Yes
joes-store.com	/	Cart=1-00501;1-07031;2-13721	11-10-02 14:22	No
aportal.com	/	Prefs=Stk:SUNW+ORCL;Spt:Jets	31-12-10 23:59	No
sneaky.com	/	UserID=3627239101	31-12-12 23:59	No

รูป 7-25
ตัวอย่างของ cookie

เขตข้อมูล path หมายถึงเส้นทางหรือตำแหน่งของไฟล์ข้อมูลในไดเร็กทอรี่ของเซิบเวอร์ที่บอกให้ทราบว่าเส้นทางใดมีสิทธิที่จะใช้ cookie โดยปกติมักจะเป็นเครื่องหมาย "/" ซึ่งหมายถึงสามารถใช้งานได้ทั้งทุกไดเร็กทอรี่

เขตข้อมูล "Content" เป็นการกำหนดค่าโดยใช้รูปแบบ name = value โดยที่ทั้ง name และ value อาจเป็นอะไรก็ได้ที่เซิบเวอร์ต้องการ เขตข้อมูลนี้คือส่วนที่เก็บค่าของ cookie

เขตข้อมูล "Expires" กำหนดระยะเวลาที่ cookie หมดอายุการใช้งาน ถ้าเขตข้อมูลที่ไม่มีอยู่ บราวเซอร์จะยกเลิกการใช้ cookie เมื่อออกจากเว็บนั้นซึ่งเรียก cookie ประเภทนี้ว่า "nonpersistent cookie" แต่ถ้ามีการระบุอายุการใช้งานไว้จะเรียกว่า "persistent cookie" ซึ่งค่าที่กำหนดจะถูกเก็บไว้ใช้งานจนกว่าจะหมดอายุ เวลาที่ระบุนี้จะเป็นเวลามาตรฐานกลาง (Greenwich Mean Time) ถ้าต้องการที่จะลบ cookie ทิ้งจากที่เก็บอยู่ในเครื่องผู้ใช้ เซิบเวอร์เพียงแค่ส่ง cookie ตัวเดิมมาแต่เปลี่ยนเวลาหมดอายุเป็นเวลาในอดีตเท่านั้น

เขตข้อมูลสุดท้ายคือ "Secure" สามารถที่จะกำหนดขึ้นมาเพื่อใช้บอกบราวเซอร์ว่าให้ส่งค่า cookie กลับคืนไปเฉพาะกับเซิบเวอร์ที่ปลอดภัยเท่านั้น คุณสมบัติข้อนี้มีไว้ใช้สำหรับ e-commerce เว็บไซต์ธนาคาร และเว็บไซต์ที่ต้องการรักษาความปลอดภัยต่าง ๆ

หลังจากที่ได้ทราบว่า cookie นั้นได้รับมาอย่างไรแล้วขั้นต่อไปก็จะดูว่าจะนำไปใช้งานอย่างไร ก่อนที่บราวเซอร์จะส่งคำร้องขอข้อมูลไปยังเซิบเวอร์ บราวเซอร์จะทำการตรวจสอบ cookie directory ดูว่ามีข้อมูล cookie ที่ส่งมาจากเว็บไซต์ที่กำลังขอข้อมูลอยู่หรือไม่ ถ้าใช่ ก็จะรวบรวมข้อมูล cookie ของเว็บไซต์นั้นทั้งหมดส่งไปพร้อมกับคำขอข้อมูลเว็บเพจ เมื่อเซิบเวอร์นั้นได้รับข้อมูล cookie ก็จะทำการแปลความหมาย cookie และทำงานตามเหมาะสมต่อไป

ลองพิจารณาดู cookie ที่เป็นไปได้จากข้อมูลในรูป 7-25 ข้อมูลบรรทัดแรกถูกกำหนดขึ้นมาโดย "toms-casino.com" ซึ่งใช้ในการแสดงตัวตนของลูกค้า เมื่อลูกค้าเข้าไปยังเว็บไซต์นี้ (เป็นเว็บไซต์ที่เล่นการพนัน) อีกครั้งหนึ่ง บราวเซอร์จะส่งข้อมูลนี้ไปให้เซิบเวอร์ทำให้เซิบเวอร์ทราบได้ว่าลูกค้าผู้นี้คือใคร เมื่อทราบหมายเลข ID ของลูกค้า เซิบเวอร์จะสามารถค้นหาข้อมูลในฐานข้อมูลและใช้ข้อมูลเหล่านี้ในการสร้างเว็บเพจที่เหมาะสมกับลูกค้าแต่ละราย เช่น จากข้อมูลในฐานข้อมูลเซิบเวอร์อาจทราบรูปแบบการเล่นพนันของลูกค้า คือถ้าลูกค้าชอบเล่นไพ่ เว็บเพจก็จะแสดงรูปภาพและข้อความเชิญชวนให้ลูกค้าไปเล่นไพ่ชนิดต่าง ๆ ถ้าลูกค้าชอบเล่นพนันม้าแข่งก็อาจจะแสดงผลการแข่งม้าในรอบล่าสุดให้ลูกค้าได้รับทราบ ทั้งนี้โดยที่ลูกค้ายังไม่ได้ป้อนข้อมูลใหม่เพิ่มเติมเลย

ข้อมูล cookie บรรทัดที่สองส่งมาจากร้าน "joes-store.com" ภาพของเว็บไซต์นี้เกี่ยวกับการที่ลูกค้าเที่ยวชมสินค้าไปรอบ ๆ ร้าน (อิเล็กทรอนิกส์) เพื่อหาสินค้าที่ต้องการซื้อ เมื่อลูกค้าพบสินค้าที่ถูกตาในราคาที่ถูกใจ ลูกค้าก็อาจใช้เมาส์คลิกไปที่สินค้าชิ้นนั้น เซิบเวอร์ก็จะสร้าง cookie ที่มีหมายเลขของสินค้าและจำนวนที่ต้องการส่งมาเก็บไว้ที่เครื่องคอมพิวเตอร์ของลูกค้า เมื่อลูกค้าเที่ยวชมสินค้าอื่นต่อไป จะมี cookie ถูกส่งกลับมาในทุก ๆ เพจที่ลูกค้าเลือกชม เมื่อมีสินค้าที่เลือกซื้อรวมกันมากขึ้น เซิบเวอร์ก็จะเพิ่มเติมข้อมูลนี้เข้าไปใน cookie ในตัวอย่างในรูป 7-25 ลูกค้าได้เลือกซื้อสินค้าไว้ 3 ชนิดแล้ว ในขั้นตอนสุดท้าย เมื่อลูกค้าเลือก "PROCEED TO CHECKOUT" ซึ่งเป็นการสิ้นสุดการ

เลือกซื้อสินค้าและถึงขั้นตอนการเรียกชำระเงิน cookie ที่เต็มไปด้วยรายการสินค้าที่ลูกค้าเลือกไว้จะถูกส่งมาพร้อมกับคำขอชำระค่าสินค้านี้ทำให้เซิบเวอร์ทราบว่าลูกค้าได้เลือกซื้อสินค้าใดไปบ้างเป็นจำนวนเท่าใด

Cookie ในลำดับที่สามคือเว็บเกี่ยวกับตลาดหุ้น เมื่อลูกค้าคลิ๊กไปยังการเชื่อมต่อกับหุ้น บราวเซอร์จะจัดการส่ง cookie มาให้ ซึ่งจะบอกให้เซิบเวอร์ของบริษัทหุ้นได้ทราบและจัดการสร้างข้อมูลเกี่ยวกับหุ้นของบริษัท Sun Microsystems บริษัท Oracle และทีมฟุตบอล New York Jet มาให้ เนื่องจาก cookie (แต่ละรายการ) อาจมีขนาดมากถึง 4 กิโลไบต์ จึงมีเนื้อที่เหลือเฟือสำหรับการใส่ข้อมูลความชอบต่างๆ ของลูกค้าได้ เช่น ข้อมูลเกี่ยวกับหัวข้อข่าวหนังสือพิมพ์ สภาพอากาศท้องถิ่น ข้อเสนอพิเศษต่างๆ และอื่นๆ

Cookie อาจมีประโยชน์สำหรับตัวเซิบเวอร์เองก็ได้ ตัวอย่างเช่น สมมุติว่าเซิบเวอร์ต้องการติดตามดูว่ามีจำนวนลูกค้ามากเท่าใดที่เข้ามาเยี่ยมชมเว็บไซต์ของตน และมีข้อมูลเพจใดบ้างที่ลูกค้าเลือกดูก่อนที่จะออกจากเว็บไซต์ไป เมื่อคำของเว็บเพจแรกมาถึง อาจจะไม่มี cookie อยู่ด้วย ดังนั้นเซิบเวอร์จึงส่ง cookie ที่ทำการนับค่า "Counter=1" กลับไป ในการเยี่ยมชมครั้งถัดไปจะส่ง cookie กลับมาที่เซิบเวอร์ ในแต่ละครั้งค่าของ counter จะถูกเพิ่มขึ้นทีละ 1 และถูกส่งกลับไปเก็บไว้ที่ลูกค้า เมื่อทำการติดตามดูค่าของ counter จะทำให้เซิบเวอร์ทราบว่าลูกค้าคนใดเข้ามาชมเพียง 1 เพจ คนใดชม 2 เพจและอื่นๆ

Cookie อาจถูกนำไปใช้ในทางที่ไม่ดีได้เช่นกัน ในทางทฤษฎีแล้ว cookie นั้นควรที่จะถูกส่งกลับไปที่เซิบเวอร์ที่เป็นเจ้าของเท่านั้น แต่ hacker (พวกที่ชอบหาจุดบกพร่องของบราวเซอร์เพื่อใช้เป็นทางเข้าไปทำอันตรายต่อเครื่องของผู้ใช้) มักจะอาศัยหาประโยชน์จากจุดบกพร่องของโปรแกรมบราวเซอร์เพื่อดึงเอาข้อมูล cookie ไปใช้ประโยชน์ เนื่องจากเว็บไซต์ e-commerce บางแห่งใส่หมายเลขบัตรเครดิตของลูกค้าไว้ใน cookie พวก hacker จึงสามารถได้ข้อมูลนี้ไปใช้ประโยชน์ในทางที่ไม่ดีได้

การใช้ cookie ในการแอบเก็บรวบรวมข้อมูลเกี่ยวกับนิสัย (profile) การใช้งานบราวเซอร์อย่างลับๆ ของผู้ใช้เป็นดังนี้ บริษัทโฆษณาทำการตกลงกับเว็บไซต์ขนาดใหญ่เพื่อให้ใส่แถบป้ายโฆษณาสินค้า (banner) เข้าไปยังเว็บเพจเพื่อแลกกับค่าตอบแทนที่บริษัทจ่ายให้กับเว็บไซต์ แทนที่ใส่ข้อมูลภาพ GIF หรือ JPEG ไฟล์ลงในแต่ละเว็บเพจ เว็บไซต์นั้นก็จะใส่ URL เข้าไปในแต่ละเว็บเพจแทน ข้อมูลในแต่ละ URL จะมีหมายเลขเฉพาะของไฟล์อยู่ในนั้น เช่น

http://www.sneaky.com/382674902342.gif

เมื่อผู้ใช้เข้ามาเยี่ยมชมเพจแรก (P) ซึ่งมีข้อมูลโฆษณานั้นอยู่ บราวเซอร์จะทำการดึงข้อมูลจากไฟล์นั้นขึ้นมา จากนั้นบราวเซอร์จะตรวจสอบและพบว่ามีการเชื่อมต่อไปยังไฟล์ภาพที่เว็บไซท์ www.sneaky.com จึงส่งความต้องการไฟล์นั้นไปที่เว็บไซต์ดังกล่าว ภาพในรูปแบบ GIF (คือแถบป้ายโฆษณาสินค้า) ก็จะถูกส่งกลับมา รวมกับ cookie ที่มีหมายเลขประจำตัวของลูกค้า 3627239101 ดังที่แสดงในบรรทัดสุดท้ายในรูป 7-25 บริษัท Sneaky ก็จะบันทึกข้อเท็จจริงที่ว่าลูกค้าหมายเลขดังกล่าวได้เข้าเยี่ยมชมเว็บเพจ (P) แล้ว ทั้งนี้การบันทึกข้อมูลเป็นเรื่องที่ง่ายมากเนื่องจากไฟล์หมายเลข

382674902342.gif นั้นจะถูกอ้างอิงโดยลูกค้าคนนี้เท่านั้น (ลูกค้าแต่ละคนจะไปดึงข้อมูลจากไฟล์เฉพาะของตนเองมาใช้) เพียงเท่านี้บริษัท Sneaky ก็จะสามารถไปเรียกเก็บเงินจากบริษัทที่ว่าจ้างให้ตนเองโฆษณาให้ได้ในทุกครั้งที่ส่งภาพโฆษณาออกไป

ในเวลาต่อมาเมื่อผู้ใช้เข้าเยี่ยมชมเว็บเพจใดๆ ที่มีโฆษณาของบริษัท Sneaky อยู่ด้วย หลังจากที่บราวเซอร์ได้ดึงข้อมูลเว็จเพจนั้นมาจากเซิบเวอร์แล้ว ก็จะพบว่ามีการเชื่อมโยงไปยัง "http://www.sneaky.com/49365491 9923.gif" จึงทำการร้องขอไฟล์นั้นไป แต่ในคราวนี้ผู้ใช้คนนี้มี cookie จากบริษัท Sneaky อยู่ด้วยจึงส่ง cookie (ซึ่งมีหมายเลขประจำตัวลูกค้าอยู่ด้วย) ไปพร้อมกับความต้องการไฟล์นี้ บริษัท Sneaky ก็จะทราบว่าลูกค้าคนนี้ได้เยี่ยมชมโฆษณาของตนเองอีกเป็นครั้งที่สองแล้ว

ในไม่ช้าบริษัท Sneaky ก็จะสามารถสร้างนิสัยการเยี่ยมชม (profile) หรือการใช้งานบราวเซอร์ของผู้ใช้แต่ละคนขึ้นมาได้ แม้ว่าผู้ใช้เหล่านั้นจะไม่เคย "คลิ๊ก" ที่โฆษณาของบริษัท Sneaky เลยแม้แต่ครั้งเดียว แน่นอนว่าบริษัทฯ จะยังคงไม่ทราบชื่อของผู้ใช้ (แต่อาจจะทราบหมายเลข IP ของเขาแล้วซึ่งก็สามารถนำไปใช้อ้างอิงหาชื่อของเขาได้จากแหล่งข้อมูลอื่นๆ) อย่างไรก็ตาม ถ้าผู้ใช้ได้เคยให้ชื่อของตนเองไปกับเว็บไซต์ใดก็ตามที่ทำงานร่วมกับบริษัท Sneaky แล้วก็จะกลายเป็นข้อมูลที่สมบูรณ์ที่บริษัท Sneaky สามารถนำไปขายให้กับบริษัทใดก็ได้ที่ต้องการใช้ข้อมูลนี้ การขายข้อมูลอาจทำกำไรให้กับบริษัท Sneaky มากพอที่จะทำให้บริษัทนี้ขยายกิจการไปทำความตกลงกับเว็บไซต์ต่างๆ มากขึ้น ซึ่งก็จะทำให้บริษัท Sneaky มีข้อมูลที่สมบูรณ์มากขึ้นและมีปริมาณมากขึ้นเพื่อจะนำไปขายให้ได้กำไรมากขึ้นไปเรื่อยๆ เรื่องที่น่าเกลียดมากที่สุดของประเด็นที่หยิบยกมากล่าวนี้ก็คือผู้ใช้จะไม่มีโอกาสระวังตัวหรือทราบถึงพฤติกรรมการแอบเก็บข้อมูลอย่างลับๆ นี้ได้เลย และอาจจะนึกว่าตนเองปลอดภัยด้วยซ้ำไปเพราะว่าไม่เคยเข้าไปแตะต้องแถบโฆษณาทั้งหลายเลย

สมมุติว่าบริษัท Sneaky ต้องการทำตัวเป็น Supersneaky คือไม่มีการเปิดเผยตัวตนต่อสาธารณะชนเลยแม้แต่น้อยก็สามารถทำได้ง่ายอย่างเหลือเชื่อนั่นคือแทนที่จะส่งภาพที่เป็นแถบป้ายโฆษณาก็เปลี่ยนเป็นภาพขนาด 1 pixel (หนึ่งจุดเล็กๆ บนจอภาพ) ที่ใช้สีดำสนิทก็จะมีผลให้กระบวนการแอบซ่อนตัวนี้มีผลเหมือนเดิมทุกประการ แต่ที่แตกต่างกันก็คือผู้ใช้จะไม่ได้เห็นแม้กระทั่งภาพโฆษณา (ให้เป็นที่น่าสงสัย) อีกต่อไป

เพื่อเป็นการรักษาตนให้พ้นจากภัยมืดที่อาจจะเกิดขึ้นได้ ผู้ใช้บางส่วนจึงปรับแต่งบราวเซอร์ของตนเองไม่ให้ยอมรับ cookie เลย อย่างไรก็ตาม นี่อาจจะเป็นการสร้างปัญหาให้กับเว็บไซท์ที่ดีๆ ที่ใช้ cookie ได้ เพื่อเป็นการแก้ปัญหานี้ ผู้ใช้บางส่วนจึงติดตั้งโปรแกรมที่เรียกว่า cookie-eating software ซึ่งเป็นโปรแกรมที่ออกแบบมาเป็นพิเศษที่จะทำการตรวจสอบ cookie ทุกตัวที่ถูกส่งเข้ามาและทำการเลือกที่จะเก็บบันทึกไว้หรือลบทิ้งไปตามแต่ที่ผู้ใช้จะกำหนด (เช่นยอมรับเฉพาะ cookie ที่มาจากเว็บไซท์ที่เชื่อถือได้เท่านั้น) ซึ่งจะเป็นการเพิ่มความสามารถในการกลั่นกรอง cookie ที่ถูกส่งเข้ามาได้

7.3.2 เอกสารแบบสแตติก

งานพื้นฐานของเว็บก็คือการถ่ายทอดเว็บเพจจากเซิบเวอร์ไปยังผู้ใช้ ในรูปแบบที่ง่ายที่สุด เว็บเพจจะอยู่ในรูปแบบสแตติก (Static) นั่นคือเป็นไฟล์ที่ถูกเก็บอยู่ในเซิบเวอร์รอที่จะให้มีผู้ใช้มาเรียกไปดู ในความหมายนี้แม้แต่ไฟล์วีดิโอก็ถือว่าเป็นเว็บเพจแบบสแตติกเพราะว่าเป็นเพียงไฟล์ชนิดหนึ่งเท่านั้น

HTML-The HyperText Markup Language

เว็บเพจนั้นถูกสร้างขึ้นมาโดยใช้ภาษา HTML (HyperText Markup Language) ซึ่งสนับสนุนให้ผู้ใช้สร้างเว็บเพจที่มีทั้งตัวอักษร และการเชื่อมโยงไปยังเว็บเพจอื่นๆ ภาษา HTML เป็นภาษาที่เรียกว่า Markup language หมายความว่าเป็นภาษาที่ใช้อธิบายวิธีการที่จะแสดงข้อความในเอกสาร เช่นการบอกรูปแบบของตัวอักษร (font) ที่จะนำมาใช้เป็นต้น Markup language จะใช้คำสั่งที่ชัดเจนในการอธิบายรูปแบบของตัวอักษร เช่น สัญลักษณ์ หมายถึงให้เริ่มต้นแสดงตัวอักษรโดยใช้ตัวหนา (boldface) และ หมายถึงจุดสิ้นสุดการใช้ตัวอักษรแบบหนา เป็นต้น จุดเด่นของภาษาประเภทนี้ที่เหนือกว่าภาษาที่ไม่มีการใช้สัญลักษณ์แสดงรูปแบบอย่างชัดเจนคือการเขียนโปรแกรมบราวเซอร์จะทำได้อย่างตรงไปตรงมา นั่นคือบราวเซอร์จะต้องเข้าใจสัญลักษณ์ต่างๆ ที่มีใช้งานเท่านั้น ภาษา markup language อื่นที่รู้จักกันโดยทั่วไปได้แก่ Tex และ troff เป็นต้น

ด้วยการรวมเอาสัญลักษณ์แสดงรูปแบบเข้าไปใน HTML ไฟล์และทำให้สัญลักษณ์เป็นมาตรฐานเดียวกัน ทำให้เกิดความเป็นไปได้ที่บราวเซอร์จะสามารถอ่านและจัดการแสดงรูปแบบใหม่ของเว็บไซท์ใดๆ ได้ ซึ่งเป็นคุณสมบัติที่สำคัญมากเพราะเว็บเพจอาจถูกสร้างขึ้นมาให้มีขนาด 1600x1200 pixels โดยใช้สีขนาด 24 บิตอาจจะต้องถูกจัดรูปแบบใหม่ให้สามารถแสดงภาพนั้นในขนาด 640x320 pixels โดยใช้สีขนาด 8 บิต

ผู้ใช้สามารถใช้โปรแกรม text editor ใดๆ ในการสร้างไฟล์ HTML ขึ้นมาได้ แต่ถ้าไม่ต้องการจดจำสัญลักษณ์ต่างๆ ที่ใช้ก็อาจใช้โปรแกรมที่สร้างขึ้นมาสำหรับการสร้างไฟล์ประเภทนี้โดยเฉพาะเรียกว่า HTML editor ก็ได้

เว็บเพจประกอบด้วย head และ body ที่ใส่ไว้ภายในสัญลักษณ์ <html> และ </html> ซึ่งเรียกว่า tag หรือคำสั่งสำหรับการจัดรูปแบบ รูป 7-26(a) แสดงให้เห็น HTML ไฟล์อันหนึ่งที่ส่วนหัวถูกแยกให้เห็นชัดเจนโดยใช้ tag <head> และ </head> ส่วน body ก็อยู่ระหว่าง tag <body> และ </body> ข้อความที่อยู่ภายใน tag คู่หนึ่งเรียกว่า directives สัญลักษณ์ tag ที่ใช้ในภาษา HTML ส่วนใหญ่จะเป็นดังนี้คือ จะเริ่มต้นด้วย tag <something> เพื่อใช้บอกจุดเริ่มต้นของการจัดรูปแบบอย่างหนึ่ง และใช้ </something> เพื่อบอกจุดสิ้นสุดของการจัดรูปแบบนั้นๆ บราวเซอร์ส่วนใหญ่จะมีเมนูให้เลือก VIEW SOURCE หรือการขอดูข้อความบนเว็บเพจในลักษณะของ HTML ไฟล์

Tag อาจเขียนในรูปแบบตัวอักษรตัวพิมพ์ใหญ่หรือตัวพิมพ์เล็กก็ได้ เช่น <head> และ <HEAD> จะมีความหมายเป็นอย่างเดียวกันอย่างไรก็ตามมาตรฐานรุ่นใหม่กำหนดให้ใช้ตัวพิมพ์เล็กเท่านั้น ตำแหน่งการวาง tag แต่ละคู่นั้นไม่มีความหมายใด บราวเซอร์ส่วนใหญ่จะไม่สนใจว่าตำแหน่งของ tag จะอยู่ติดกับข้อความหรืออยู่บนบรรทัดเดียวกันหรือไม่ โดยจะพยายามจัดรูปแบบข้อความให้อยู่ภายในพื้นที่ที่ใช้แสดงข้อความนั้นเป็นสำคัญ ดังนั้นการใส่ตัวอักษรช่องว่าง (white space) เข้าไปจะมีผลเพียงทำให้ HTML ไฟล์นั้นอ่านได้ง่ายขึ้นหรือมีโครงสร้างที่สวยงามขึ้นเท่านั้น บรรทัดว่างเปล่าไม่สามารถนำมาใช้ในการแยกข้อความออกจากกันได้ จะต้องใช้ tag ในการแยกข้อความออกจากกัน

Tag บางชนิดจะมีพารามิเตอร์เป็นของตัวเองเพื่อใช้ในการกำหนดคุณลักษณะที่เรียกว่า attributes เช่น

<img src=”abc” alt=”foobar”>

หมายถึง tag <img> ที่มีพารามิเตอร์สองตัวคือ “srv” ที่กำหนดค่าให้เป็น “abc” และ “alt” ที่กำหนดค่าให้เป็น “foobar” มาตรฐาน HTML จะกำหนดค่าพารามิเตอร์ที่เป็นไปได้ให้สำหรับแต่ละ tag (ถ้ามี) รวมทั้งกำหนดความหมายและค่าที่เป็นไปได้ให้ด้วย เนื่องจากพารามิเตอร์แต่ละตัวมีชื่อเฉพาะ การวางลำดับของพารามิเตอร์จึงเป็นเรื่องที่ไม่จำเป็น

โดยทางเทคนิคแล้ว เอกสาร HTML นั้นเขียนขึ้นมาโดยใช้ตัวอักษรแบบ ISO 8859-1 Latin-1 แต่สำหรับแป้นพิมพ์ (keyboard) โดยทั่วไปจะมีเฉพาะตัวอักษร ASCII จึงต้องใช้ escape sequence ในการแสดงตัวอักษรที่ไม่มีอยู่บนแป้นพิมพ์ เช่นตัว _ เป็นต้น ตัวอักษร escape sequence เขียนโดยใช้ตัว “&” และปิดท้ายด้วย “;” เช่น “ ” หมายถึงตัวอักษรว่าง “è” หมายถึงตัว “_” และ “é” หมายถึงตัว “_” เป็นต้น และเนื่องจากตัวอักษร “<“, “>”, และ “&” มีความหมายเฉพาะอยู่แล้วดังนั้นถ้าจะให้ตัวสัญลักษณ์เหล่านี้ปรากฏอยู่บนข้อความในเว็บก็จะต้องใช้ escape sequence เท่านั้นคือ “<”, “>”, และ “&” ตามลำดับ

ต่อไปมาดูขีดความสามารถพิเศษอื่นๆ ที่แสดงในรูป 7-26 tag ทั้งหมดที่ใช้ในรูป 7-26 รวมทั้ง tag อื่นๆ ได้แสดงไว้ในรูป 7-27 tag <title> และ </title> ใช้ครอบข้อความที่บอกให้ทราบถึงชื่อของเว็บเพจนั้น การใช้ข้อความหลักขนาดใหญ่จะอยู่ภายใต้ tag <hn> และ </hn> โดยที่ n คือตัวเลขตั้งแต่ 1 ถึง 6 ดังนั้น <h1> จึงหมายถึงข้อความที่เป็นหัวข้อที่สำคัญที่สุด และ <h6> คือหัวข้อที่มีความสำคัญน้อยที่สุด การแสดงหัวข้อนี้จะขึ้นอยู่กับบราวเซอร์แต่ละตัวว่าจะใช้รูปแบบเป็นอย่างใด โดยปกติหมายเลขต่ำจะแสดงด้วยตัวอักษรขนาดใหญ่และเป็นตัวอักษรที่หนากว่าปกติ บราวเซอร์ยังอาจใช้สีที่แตกต่างกันอีกด้วย

Tag <b> และ <i> ใช้บอกการแสดงตัวอักษรแบบหนาและตัวหนังสือเรียงตามลำดับ ถ้าบราวเซอร์ไม่สามารถแสดงตัวหนังสือแบบใดได้ก็จะใช้วิธีการเปลี่ยนสีของตัวอักษรนั้นๆ แทน

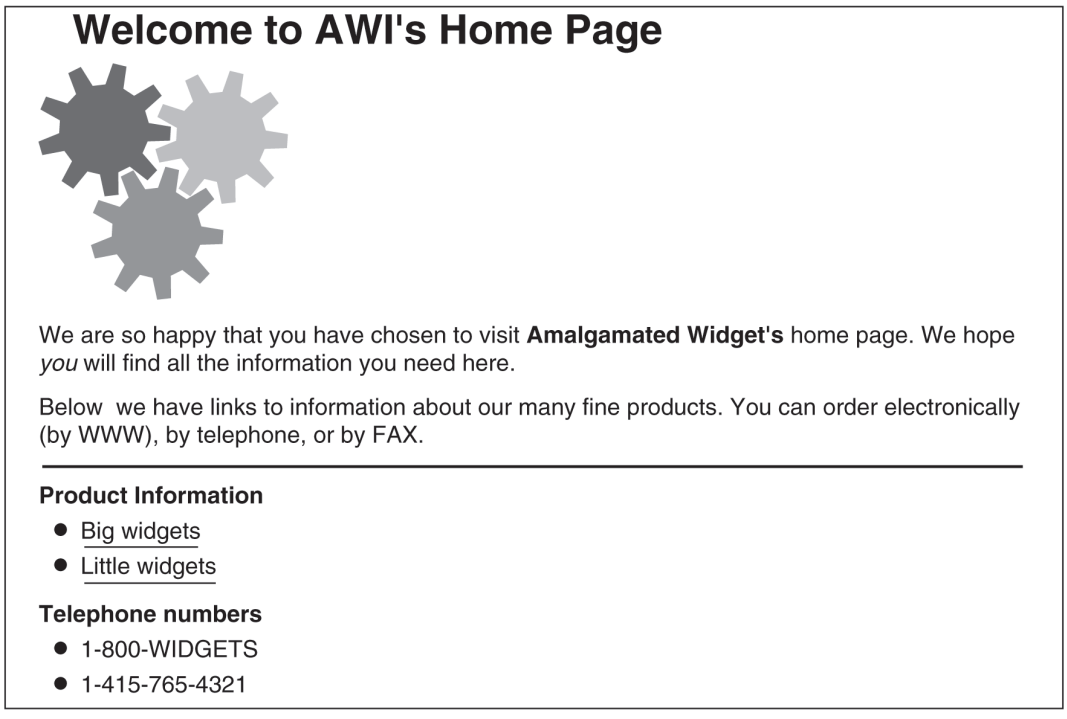
HTML สนับสนุนกลไกหลายแบบในการจัดโครงสร้างแบบลิสต์ (list) รวมทั้ง nested list ลิสต์เริ่มต้นโดยใช้ tag <ul> หรือ <ol> โดยมี เป็นตัวบอกจุดเริ่มต้นของรายการที่อยู่ในลิสต์ tag <ul> เป็นจุดเริ่มต้นของลิสต์แบบไม่มีการเรียงลำดับ (unordered list) แต่ละรายการที่อยู่ในลิสต์จะต้องใช้สัญลักษณ์ กำกับซึ่งจะปรากฏเป็นสัญลักษณ์ “(“ เมื่อแสดงผ่านบราวเซอร์ ส่วน tag <ol> นั้นใช้กับลิสต์แบบเรียงลำดับ (ordered list) เมื่อใช้คู่กับ tag <li> จะปรากฏเป็นหมายเลขลำดับเมื่อแสดงผ่านบราวเซอร์

Tag
, , และ <hr> ใช้ในการแสดงขอบเขตของแต่ละหัวข้อของข้อความ
 ใช้แสดงจุดแบ่งของข้อความ หรือการขึ้นย่อหน้าใหม่ ส่วน จะทำให้เกิดการเว้นเพื่อขึ้นย่อหน้าใหม่ที่คั่นด้วยบรรทัดว่าง ในทางทฤษฎีจะมี tag เพื่อใช้บอกจุดสิ้นสุดของย่อหน้า แต่ในทางปฏิบัติไม่จำเป็นต้องใช้ และ <hr> ใช้ในการแบ่งข้อความขึ้นย่อหน้าใหม่พร้อมทั้งตีเส้นกั้นหน้า (ตามแนวนอน) ให้ด้วย

HTML ยอมให้มีการแทรกรูปภาพเข้าไปในเว็บเพจโดยการใช้คำสั่งประเภท in-line ได้ tag <img> ใช้ในการกำหนดการแสดงรูปภาพ ณ ตำแหน่งที่อยู่ขณะนั้น ซึ่งจะมีพารามิเตอร์อยู่หลายตัวด้วยกัน คือ

```
<html>
<head> <title> AMALGAMATED WIDGET, INC. </title> </head>
<body> <h1> Welcome to AWI's Home Page </h1>
<img src="http://www.widget.com/images/logo.gif" ALT="AWI Logo"> <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's</b>
home page. We hope <i> you </i> will find all the information you need here.
<p>Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by fax. </p>
<hr>
<h2> Product information </h2>
<ul>
   <li> <a href="http://widget.com/products/big"> Big widgets </a>
   <li> <a href="http://widget.com/products/little"> Little widgets </a>
</ul>
<h2> Telephone numbers </h2>
<ul>
   <li> By telephone: 1-800-WIDGETS
   <li> By fax: 1-415-765-4321
</ul>
</body>
</html>
```

(a)



(b)

รูป 7-26
(a) ตัวอย่าง HTML ไฟล์
(b) ภาพที่แสดงผ่านบราวเซอร์

รูป 7-27
HTML tag บางส่วน

Tag	Description
<html> ... </html>	Declares the Web page to be written in HTML
<head> ... </head>	Delimits the page's head
<title> ... </title>	Defines the title (not displayed on the page)
<body> ... </body>	Delimits the page's body
<h *n*> ... </h*n*>	Delimits a level *n* heading
...	Set ... in boldface
...	Set ... in italics
...	Center ... on the page horizontally
...	Brackets an unordered (bulleted) list
...	Brackets a numbered list
...	Brackets an item in an ordered or numbered list
 	Forces a line break here
<p>	Starts a paragraph
<hr>	Inserts a horizontal rule
<img src="...">	Displays an image here
 ... 	Defines a hyperlink

"src" จะบอกชื่อของไฟล์รูปภาพโดยใช้ชื่อในรูปแบบ URL มาตรฐาน HTML ไม่ได้ระบุรูปแบบของภาพกราฟฟิกที่จะนำมาใช้ แต่ในทางปฏิบัติ บราวเซอร์ทุกตัวสนับสนุนรูปแบบ GIF และ JPEG ไฟล์บราวเซอร์มีอิสระในการสนับสนุนกราฟฟิกในรูปแบบอื่นแต่ก็อาจทำให้เกิดปัญหาขึ้นมาได้ กล่าวคือ ผู้ใช้อาจเลือกใช้บราวเซอร์ที่สนับสนุนภาพในรูปแบบ BMP ไฟล์จึงสร้างเว็บเพจที่มีรูปภาพประเภทนี้อยู่ด้วย แต่เมื่อคนอื่นซึ่งใช้บราวเซอร์ทั่วไปเข้ามาดูเว็บเพจนี้ ตรงส่วนที่เคยเป็นภาพสวยงามนั้นก็จะกลายเป็นเพียงพื้นที่ว่างเปล่า

พารามิเตอร์ตัวอื่นของ <img> คือ "align" ซึ่งควบคุมการวางตำแหน่งของรูปภาพ (top, middle, bottom) โดยยึดเอาบรรทัดที่ใช้แสดงข้อความเป็นฐานอ้างอิง "alt" เป็นทางเลือกในการแสดงข้อความแทนรูปภาพเมื่อบราวเซอร์ของผู้ใช้ไม่สนับสนุนการแสดงภาพกราฟฟิกหรือถูกสั่งให้ไม่แสดงรูปภาพ และ "ismap" เป็นพารามิเตอร์ที่ใช้แสดงว่ารูปภาพนั้นถูกใช้เป็น active map หรือไม่ (เป็นรูปภาพที่ผู้ใช้สามารถคลิ๊กเพื่อเชื่อมโยงไปยังเว็บเพจอื่น)

ท้ายที่สุดก็คือการเชื่อมโยง hyperlink ซึ่งใช้ tag <a> และ พารามิเตอร์ที่ใช้งานร่วมกันมีอยู่หลายตัวคือ "href" ซึ่งเป็นการอ้างอิงไปยังเว็บเพจอื่นโดยใช้ชื่อในรูปแบบ URL และ "name" เป็นการเชื่อมโยงโดยการใช้ชื่อที่เป็น hyperlink name ข้อความที่อยู่ระหว่าง <a> และ จะถูกแสดงบนเว็บเพจโดยบราวเซอร์ ถ้าผู้ใช้เลือกข้อความนี้ก็จะทำให้บราวเซอร์ไปดึงเว็บเพจใหม่ที่ระบุไว้ขึ้นมาแทนที่ ผู้ใช้อาจเลือกใช้รูปภาพแทนข้อความนี้ก็ได้โดยใช้ tag <img>

ตัวอย่างการใช้ tag <a> ลองพิจารณาคำสั่ง HTML ต่อไปนี้

 NASA's Home page

เมื่อนำไปแสดงบนบราวเซอร์จะปรากฏภาพในลักษณะนี้

NASA's home page

ถ้าผู้ใช้คลิ็กเมาส์ที่ข้อความนี้ บราวเซอร์ก็จะไปดึงเว็บเพจที่ http://www.nasa.gov มาแสดงในทันที

ตัวอย่างต่อไป ลองดูคำสั่ง HTML ดังนี้

 <img src= "shuttle.gif" alt= "NASA">

เมื่อแสดงเพจนี้บนบราวเซอร์จะปรากฏเป็นรูปภาพที่นำมาจากไฟล์ "shuttle.gif" และเมื่อคลิ็กที่รูปภาพนี้บราวเซอร์ก็จะดึงโฮมเพจของนาซาขึ้นมาแทนที่ เหมือนกับการคลิ็กข้อความที่ขีดเส้นใต้ในตัวอย่างก่อนหน้านี้ ถ้าผู้ใช้กำหนดให้บราวเซอร์ไม่ต้องแสดงรูปภาพ ก็จะปรากฏเป็นข้อความ "NASA" ขึ้นมาแทนที่ที่เคยเป็นรูปภาพ

Tag <a> อาจเลือกใช้พารามิเตอร์ "name" ในการวางตำแหน่ง hyperlink ก็ได้เพื่อให้บราวเซอร์ข้ามการแสดงผลส่วนตอนต้นของเว็บเพจไปที่จุดกึ่งกลางของเว็บเพจนั้นในทันที ตัวอย่างเช่น เว็บเพจอาจจะเริ่มต้นด้วยหน้าสารบัญที่ประกอบด้วยข้อความต่างๆ ที่สามารถคลิ็กเพื่อการเชื่อมโยงไปยังตำแหน่งต่างๆ ได้ เมื่อผู้ใช้คลิ็กที่ข้อความใดก็ตามก็จะเสมือนหนึ่งว่าผู้ใช้ได้เปิดเว็บเพจไปที่ตำแหน่งที่ข้อความนั้นเชื่อมโยงอยู่

HTML ยังคงได้รับการพัฒนามาอย่างต่อเนื่อง HTML 1.0 และ HTML 2.0 ไม่มีข้อมูลชนิดตาราง (table) อยู่ด้วยแต่ก็ได้รับการเพิ่มเติมเข้าไปใน HTML 3.0 ตาราง HTML ประกอบด้วยข้อมูลตามแนวนอนอย่างน้อยหนึ่งแถว แต่ละแถวประกอบด้วยเซล (cell) อย่างน้อยหนึ่งเซล ข้อมูลในแต่ละเซลอาจเป็นข้อมูลชนิดต่างๆ ได้หลายชนิดเช่น ข้อความ รูปภาพ ไอคอน หรือแม้กระทั่งเป็นตารางซ้อนอยู่ภายในตาราง เซลอาจถูกนำมารวมกันได้เช่น ข้อความที่ปรากฏคร่อมอยู่ในหัวแถวหลายแถวเพื่อบอกชนิดหรือกลุ่มของข้อมูล

รูป 7-28(a) แสดงการกำหนดข้อมูลแบบตารางและรูป 7-28(b) แสดงภาพที่จะปรากฏบนบราวเซอร์ ตัวอย่างนี้แสดงให้เห็นความเป็นไปได้ในการสร้างตารางซึ่งเริ่มต้นด้วยการใช้ tag <table> ส่วน tag <caption> ใช้ในการกำหนดข้อความที่ทำหน้าที่เป็นชื่อตารางข้อมูลในแต่ละแนวนอนจะเริ่มต้นด้วย tag <tr> แต่ละเซลจะเริ่มต้นด้วย <th> (table header) หรือ <td> (table data) เหตุที่แยกความแตกต่างออกจากกันก็เพื่อเปิดโอกาสให้บราวเซอร์สามารถแสดงภาพออกมาได้แตกต่างกันได้ พารามิเตอร์ต่างๆก็อาจถูกนำมาใช้เพื่อกำหนดคุณสมบัติ เช่นการกำหนดลักษณะการวางข้อความ (ชิดซ้าย หรือไว้ตรงกลาง) การกำหนดขนาดของเส้นขอบ การกำหนดกลุ่มเซล เป็นต้น

ใน HTML 4.0 ได้เพิ่มคุณสมบัติใหม่ๆ เข้าไปอีกเช่นการใช้อุปกรณ์พิเศษสำหรับผู้พิการ การรวมอ็อบเจ็คเข้าไว้ภายใน การสนับสนุน scripting language เพื่อเพิ่มความสามารถในการสร้าง dynamic content เป็นต้น

รูป 7-28
(a) An HTML table
(b) ภาพที่ปรากฏบนบราวเซอร์

```
<html>
<head> <title> A sample page with a table </title> </head>
<body>
<table border=1 rules=all>
<caption> Some Differences between HTML Versions </caption>
<col align=left>
<col align=center>
<col align=center>
<col align=center>
<col align=center>
<tr> <th>Item  <th>HTML 1.0  <th>HTML 2.0  <th>HTML 3.0 <th>HTML 4.0 </tr>
<tr> <th> Hyperlinks <td> x <td> x <td> x <td> x </tr>
<tr> <th> Images <td> x <td> x <td> x <td> x </tr>
<tr> <th> Lists <td> x <td> x <td> x <td> x </tr>
<tr> <th> Active Maps and Images <td>   <td> x <td> x <td> x </tr>
<tr> <th> Forms <td>   <td> x <td> x <td> x </tr>
<tr> <th> Equations <td>   <td>   <td> x <td> x </tr>
<tr> <th> Toolbars <td>   <td>   <td> x <td> x </tr>
<tr> <th> Tables <td>   <td>   <td> x <td> x </tr>
<tr> <th> Accessibility features <td>   <td>   <td>   <td> x </tr>
<tr> <th> Object embedding <td>   <td>   <td>   <td> x </tr>
<tr> <th> Scripting <td>   <td>   <td>   <td> x </tr>
</table>
</body>
</html>
```

(a)

Some Differences between HTML Versions

Item	HTML 1.0	HTML 2.0	HTML 3.0	HTML 4.0
Hyperlinks	x	x	x	x
Images	x	x	x	x
Lists	x	x	x	x
Active Maps and Images		x	x	x
Forms		x	x	x
Equations			x	x
Toolbars			x	x
Tables			x	x
Accessibility features				x
Object embedding				x
Scripting				x

ฟอร์ม

HTML 1.0 นั้นทำงานในลักษณะทิศทางเดียว นั่นคือผู้ใช้สามารถที่จะเรียกดูเว็บเพจได้จากผู้ให้บริการ แต่เป็นการยากมากที่ผู้ใช้จะส่งข้อมูลกลับไปหาผู้ให้บริการ เมื่อองค์กรทางธุรกิจได้หันมาให้ความสนใจกับเว็บมากขึ้นเรื่อย ๆ ก็ทำให้เกิดความต้องการที่จะให้ผู้ใช้สามารถส่งข้อมูลกลับไปหาผู้ให้บริการได้บ้าง ตัวอย่างเช่น องค์กรธุรกิจจำนวนมากต้องการที่จะรับคำสั่งซื้อสินค้าผ่านเว็บเพจ บริษัทที่ขายซอฟต์แวร์ต้องการขายซอฟต์แวร์ของตนผ่านเว็บและให้ผู้ใช้ลงทะเบียนลูกค้าทางเว็บ ส่วนบริษัทที่ให้บริการเกี่ยวกับการค้นหาข้อมูลบนเว็บก็ต้องการให้ลูกค้าสามารถส่งคำหรือข้อความที่ต้องการค้นหามาให้ทางเว็บ

ความต้องการเหล่านี้ได้ทำให้เกิดเป็น ฟอร์ม (Forms) ซึ่งเริ่มมีใช้ใน HTML 2.0 ฟอร์มประกอบด้วยช่องเติมคำหรือปุ่มกดที่ยอมให้ผู้ใช้สามารถเติมข้อความเข้ามาหรือกดปุ่มเลือกทางเลือกที่ต้องการ จากนั้นก็ส่งความต้องการกลับมาที่ผู้ให้บริการ ในที่นี้จะใช้ tag <input> เป็นตัวกำกับการทำงาน tag นี้มีพารามิเตอร์หลายตัวเพื่อกำหนดขนาด ลักษณะและการใช้งานช่องเติมคำ ฟอร์มที่นิยมใช้กันมากที่สุดคือฟอร์มว่างเปล่าที่เว้นว่างไว้เพื่อรับข้อมูลจากผู้ใช้ ปุ่มที่รอให้ผู้ใช้เลือกกด แผนที่ที่เป็น hyperlink และปุ่ม submit ดังที่แสดงตัวอย่างในรูป 7-29

ฟอร์มนั้นถูกสร้างขึ้นไว้ภายใน tag <form> และ </form> มีวิธีการสร้าง input box อยู่ 3 วิธี วิธีแรกอยู่ตามหลังคำว่า "Name" box นี้มีขนาด 46 ตัวอักษรและต้องการให้ผู้ใช้พิมพ์ข้อความ (ในที่นี้คือชื่อผู้ใช้) เข้ามา ซึ่งจะถูกนำไปเก็บไว้ในตัวแปรชื่อ "customer" tag <p> บอกให้บราวเซอร์แสดงข้อความที่ตามมาข้างหลังในบรรทัดต่อมาแม้ว่าในบรรทัดปัจจุบันจะยังคงมีเนื้อที่ว่างเหลืออยู่ก็ตาม ด้วยการใช้ tag <p> และอื่น ๆ ผู้สร้างสามารถกำหนดรูปร่างของฟอร์มนี้ได้ตามใจชอบ

บรรทัดต่อมาเป็นการถามถึงที่อยู่ของผู้ใช้ที่มีขนาดไม่เกิน 40 ตัวอักษร จากนั้นเป็นการถามชื่อเมือง รัฐ และประเทศที่อยู่อาศัย เนื่องจากไม่มี tag <p> อยู่ด้วยดังนั้นชื่อเมือง รัฐและประเทศจะปรากฏอยู่บนบรรทัดเดียวกัน (ถ้าเป็นไปได้) ในมุมมองของบราวเซอร์ ย่อหน้านี้ประกอบด้วย ข้อความ 3 ข้อความ และ box จำนวน 3 box ที่วางสลับกันอยู่ บราวเซอร์จะนำทั้งหมดนี้แสดงบนหน้าจอภาพโดยเริ่มต้นจากทางด้านซ้ายสุดของจอภาพไล่ตามลำดับมาทางขวามือ ถ้าเป็นไปได้ก็จะแสดงบนบรรทัดเดียวกัน แต่ถ้าหน้าจอเล็กเกินไปก็จะขึ้นบรรทัดใหม่ให้โดยอัตโนมัติ

บรรทัดต่อไปเป็นการถามหมายเลขบัตรเครดิตและวันที่บัตรหมดอายุ การส่งข้อมูลที่เป็นความลับเช่นนี้ควรทำในกรณีที่เป็นการสื่อสารที่ไว้ใจได้เท่านั้น

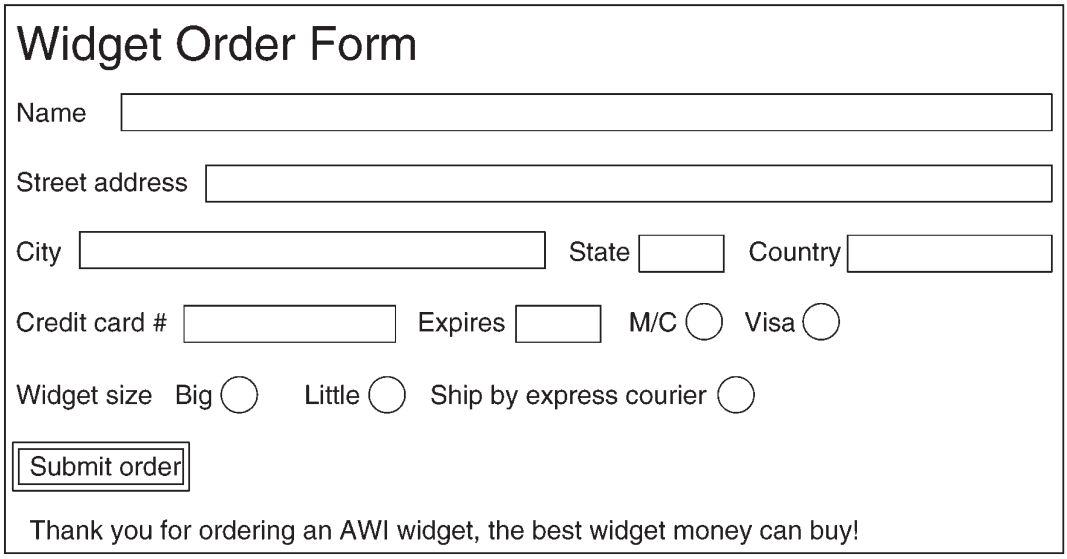
ภายหลังจากวันที่บัตรหมดอายุเป็นวิธีการที่ 2 ในการส่งข้อมูลกลับมายังผู้ให้บริการ เรียกว่า radio buttons ซึ่งจะถูกนำมาใช้ในกรณีที่ต้องการให้ผู้ใช้เลือกความเป็นไปได้จากข้อเลือกจำนวนสองข้อขึ้นไป บราวเซอร์จะแสดง box เหล่านี้ในรูปแบบที่อนุญาตให้ผู้ใช้สามารถเลือกหรือไม่เลือกด้วยการใช้เมาส์คลิ๊กปุ่มที่ต้องการหรือใช้แป้นพิมพ์ เมื่อคลิ๊ก "เลือก" ที่ปุ่มหนึ่งจะมีผลให้ปุ่มที่เหลือในกลุ่มเดียวกันมีสถานะเป็น "ไม่เลือก" ทั้งหมด การแสดงให้เห็นบนจอภาพเป็นหน้าที่ของบราวเซอร์ ขนาดของสินค้าก็ใช้เป็น radio button เหมือนกัน ซึ่งทั้งสองกลุ่มนี้จะแยกจากกันโดยการใช้ชื่อ "Name" ที่แตกต่างกัน

พารามิเตอร์ "value" ใช้ในการบอกให้ทราบว่าปุ่ม radio button ไหนที่ถูกเลือก เช่น เมื่อผู้ใช้เลือกปุ่มที่เขียนว่า "M/C" ตัวแปร "cc" ก็จะถูกกำหนดค่าให้เป็น "mastercard" และถ้าเลือกปุ่มที่เขียนว่า "Visa" ค่าของ "cc" ก็จะกลายเป็น "visacard"

รูป 7-29
(a) คำสั่ง HTML สำหรับการสร้างฟอร์ม
(b) เพจที่ได้รับ

```
<html>
<head> <title> AWI CUSTOMER ORDERING FORM </title> </head>
<body>
<h1> Widget Order Form </h1>
<form ACTION="http://widget.com/cgi-bin/widgetorder" method=POST>
<p> Name <input name="customer" size=46> </p>
<p> Street Address <input name="address" size=40> </p>
<p> City <input name="city" size=20> State <input name="state" size =4>
Country <input name="country" size=10> </p>
<p> Credit card # <input name="cardno" size=10>
Expires <input name="expires" size=4>
M/C <input name="cc" type=radio value="mastercard">
VISA <input name="cc" type=radio value="visacard"> </p>
<p> Widget size Big <input name="product" type=radio value="expensive">
Little <input name="product" type=radio value="cheap">
Ship by express courier <input name="express" type=checkbox> </p>
<p><input type=submit value="submit order"> </p>
Thank you for ordering an AWI widget, the best widget money can buy!
</form>
</body>
</html>
```

(a)



(b)

ภายหลังจาก radio button จำนวน 2 ชุดผ่านไปแล้วก็มาถึงวิธีการป้อน input ให้แก่ผู้ให้บริการอีกชนิดหนึ่งคือการใช้ checkbox ปุ่มชนิดนี้จะมีอยู่เพียงสองค่าคือ "ใช่" และ "ไม่ใช่" สิ่งที่แตกต่างไปจาก radio button ก็คือ ข้อมูลทุกอย่างที่เป็น checkbox จะมีค่าเป็นอิสระจาก checkbox ตัวอื่นๆ เสมอ

นอกจากนี้แล้ว สำหรับรายการเลือกที่ยาวมาก การใช้ radio button อาจจะไม่สะดวกต่อการใช้งาน ดังนั้นจึงใช้ tag <select> และ </select> ในการกำหนดขอบเขตของตัวเลือกที่มีแต่ยังคงความหมายของ radio button เอาไว้ (นอกจากจะกำหนดใช้ตัวเลือก "multiple" ซึ่งจะมีผลให้ผู้ใช้สามารถเลือก radio button ได้หลายตัวเลือกคล้ายกับ checkbox) บราวเซอร์บางตัวก็ใช้วิธีการแสดง radio button โดยใช้เป็นลักษณะเมนูแทน

นอกจากวิธีการป้อนข้อมูลแบบ radio button และ checkbox แล้ว วิธีการที่สามที่ใช้ในการส่งข้อมูลให้แก่ผู้ให้บริการก็คือ "text" แต่เนื่องจาก input ชนิดนี้เป็นค่า default อยู่แล้วคือจะถูกนำมาใช้งานเมื่อไม่ได้ระบุวิธีการแบบอื่นเอาไว้ ดังนั้นจึงไม่มีความจำเป็นจะต้องระบุว่า "type = text" การป้อนข้อมูลอีกสองวิธีคือ "password" และ "textarea" ข้อมูลชนิด "password" นั้นมีลักษณะคล้าย text ยกเว้นแต่ว่าตัวอักษรที่ผู้ใช้พิมพ์เข้ามานั้นจะไม่ถูกนำไปแสดงบนหน้าจอภาพ ส่วน textarea นั้นก็คล้ายกับ text ยกเว้นแต่ว่าสามารถบรรจุข้อความได้หลายบรรทัด

เมื่อมองย้อนกลับไปดูตัวอย่างในรูป 7-29 ก็มาถึงปุ่มที่มีความสำคัญมากคือปุ่ม "submit" เมื่อผู้ใช้เลือกใช้ปุ่มนี้ข้อมูลทั้งหมดของผู้ใช้ที่ปรากฏอยู่ในฟอร์มจะถูกส่งกลับไปยังเครื่องผู้ให้บริการ พารามิเตอร์ "value" ในที่นี้จะใช้เป็นข้อความเขียนกำกับปุ่มนี้ซึ่งจะถูกแสดงบนบราวเซอร์ สำหรับ text box ค่าที่เก็บอยู่ในเขตข้อมูล value จะถูกนำมาแสดงพร้อมกับฟอร์มแต่ผู้ใช้จะสามารถแก้ไขหรือลบทิ้งได้ checkbox และ radio box สามารถที่จะถูกกำหนดค่าเริ่มต้นได้ โดยใช้เขตข้อมูลที่เรียกว่า "checked"

เมื่อผู้ใช้เลือกปุ่ม submit บราวเซอร์จะทำการรวบรวมข้อมูลที่ผู้ใช้ป้อนเข้ามาไว้เป็นข้อความจำนวนหนึ่งบรรทัด (ที่อาจจะยาวมาก) และจัดการส่งไปให้ผู้ให้บริการทำการประมวลผล เครื่องหมาย "&" ใช้ในการแยกข้อมูลแต่ละเขตข้อมูลออกจากกัน และเครื่องหมาย "+" ใช้แทนความหมายของตัวอักษรว่าง (space) สำหรับตัวอย่างที่อาจได้รับจากการใช้ฟอร์มในรูป 7-29 นั้นแสดงให้เห็นในรูป 7-30 (ในที่นี้เขียนเป็น 3 บรรทัดเนื่องจากหนังสือมีความยาวไม่พอที่จะเขียนข้อความทั้งหมดได้ในบรรทัดเดียว)

```
customer=John+Doe&address=100+Main+St.&city=White+Plains&
state=NY&country=USA&cardno=1234567890&expires=6/98&cc=mastercard&
product=cheap&express=on
```

รูป 7-30 การตอบรับที่อาจเป็นไปได้จากฟอร์มในรูป 7-29

XML และ XSL

HTML ไม่ว่าจะใช้หรือไม่ใช้ฟอร์มก็ไม่ได้ทำให้เว็บเพจมีโครงสร้างขึ้นมาแต่อย่างใด นอกจากนี้ยังเป็นการผสมผสานระหว่างข้อความภายในกับการจัดรูปแบบข้อความเข้าด้วยกัน เมื่อ e-commerce และโปรแกรมประยุกต์อื่นๆได้เริ่มถูกนำมาใช้งานมากขึ้นทำให้เกิดความต้องการที่จะให้เว็บเพจมีโครงสร้างที่ชัดเจนและจัดการแยกข้อความออกจากการจัดรูปแบบข้อความ ตัวอย่างเช่นโปรแกรมที่ทำการค้นหาข้อมูลบนเว็บเพื่อที่จะค้นหาหนังสือหรือแผ่นซีดีเพลงที่มีการเสนอราคาดีที่สุด จะต้องมีความสามารถ

ในการวิเคราะห์เว็บเพจเพื่อให้ได้ทั้งชื่อสินค้าและราคาที่ต้องการ เป็นการยากมากที่จะเขียนโปรแกรมเพื่อค้นหาว่าส่วนใดคือชื่อ ส่วนใดคือราคาสินค้าบนเว็บเพจที่เขียนขึ้นมาด้วยภาษา HTML

ด้วยเหตุผลนี้ องค์กร W3C จึงได้พัฒนาภาษา HTML ขึ้นมาใหม่เพื่อให้เว็บเพจมีโครงสร้างที่ชัดเจนและสามารถประมวลผลได้โดยอัตโนมัติ ผลที่ได้รับคือภาษาใหม่สองภาษา ภาษาแรกคือ XML (eXtensible Markup Language) ซึ่งสามารถอธิบายข้อมูลที่อยู่ภายในเว็บเพจในลักษณะของการมีโครงสร้างที่ดี และภาษาที่สองคือ XSL (eXtensible Style Language) ซึ่งสามารถอธิบายรูปแบบการนำเสนอโดยเป็นอิสระจากข้อความที่อยู่ภายใน ภาษาทั้งสองนี้มีความสลับซับซ้อนมาก ในที่นี้จึงเป็นเพียงการอธิบายเพื่อให้เห็นภาพว่าภาษานี้ทำงานอย่างไรเท่านั้น

รูป 7-31 แสดงตัวอย่างเอกสาร XML ซึ่งมีการอธิบายโครงสร้างว่าเป็น "book_list" ซึ่งก็คือรายการหนังสือนั่นเอง หนังสือแต่ละเล่มประกอบด้วย 3 เขตข้อมูลคือ "title" "author" และ "year" โครงสร้างนี้เป็นแบบที่ง่ายมาก ซึ่งอนุญาตให้มีเขตข้อมูลซ้อนกันได้ เช่น หนังสือที่มีผู้แต่งหลายคน เขตข้อมูลทางเลือก (เช่นชื่อของ CD-ROM ที่มาพร้อมกับหนังสือ) และเขตข้อมูลอื่นๆ (เช่น URL ของเว็บไซต์ที่ขายหนังสือเล่มนี้)

ในตัวอย่างนี้ เขตข้อมูลทั้งสามเป็นอิสระต่อกันและกันแต่ก็อนุญาตให้แบ่งเป็นเขตข้อมูลย่อยได้ ตัวอย่างเช่น เขตข้อมูล "auther" อาจถูกแบ่งออกเป็นเขตข้อมูลย่อยที่มีความละเอียดมากยิ่งขึ้นเพื่อใช้ในการค้นหาข้อมูลได้ดังนี้

```
<author>
<first_name> Andrew </first_name>
<last_name> Tanenbaum </last_name>
</author>
```

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="book_list.xsl"?>

<book_list>

<book>
   <title> Computer Networks, 4/e </title>
   <author> Andrew S. Tanenbaum </author>
   <year> 2003 </year>
</book>

<book>
   <title> Modern Operating Systems, 2/e </title>
   <author> Andrew S. Tanenbaum </author>
   <year> 2001 </year>
</book>

<book>
   <title> Structured Computer Organization, 4/e </title>
   <author> Andrew S. Tanenbaum </author>
   <year> 1999 </year>
</book>

</book_list>
```

รูป 7-31 รูปแบบเว็บเพจในภาษา XML

เขตข้อมูลอาจถูกแบ่งออกเป็นเขตข้อมูลย่อย เขตข้อมูลย่อยอาจถูกแบ่งเป็นเขตข้อมูลที่ย่อยเล็กลงไปอีกซึ่งสามารถแบ่งออกได้เป็นหลายระดับขั้น

สิ่งที่ปรากฏอยู่ในรูป 7-31 คือการกำหนดรายชื่อหนังสือที่ประกอบด้วยหนังสือ 3 เล่ม ซึ่งไม่ได้บอกอะไรเลยเกี่ยวกับรูปแบบการจัดแสดงข้อมูลของหนังสือทั้งสามเล่ม เพื่อการอธิบายเกี่ยวกับรูปแบบการแสดง จำเป็นต้องใช้ไฟล์ที่สองชื่อ book_list.xsl ซึ่งประกอบด้วยการกำหนดความหมายด้วยภาษา XSL ไฟล์นี้เรียกว่าเป็น style sheet ซึ่งบอกให้ทราบว่าจะแสดงข้อมูลในลักษณะใดบนเว็บเพจ (อันที่จริงมีทางเลือกที่ไม่จำเป็นต้องใช้ style sheet เช่นการแปลง XML ไฟล์ให้ไปอยู่ในรูป HTML ไฟล์ก็ได้)

ตัวอย่างของ XSL ไฟล์ของรูป 7-31 แสดงอยู่ในรูป 7-32 หลังจากส่วนที่ใช้อธิบายข้อกำหนดบางประการรวมทั้ง URL ของมาตรฐาน XSL ข้อมูลในไฟล์ประกอบด้วย tag <html> และ <body> ซึ่งเป็นการกำหนดจุดเริ่มต้นของเว็บเพจตามปกติ ต่อมาคือการกำหนดใช้งาน table ที่รวมทั้งการกำหนดชื่อคอมลัมภ์ทั้งสามคอลัมภ์ การกำหนดรูปแบบของ XML และ XSL นั้นมีความเข้มงวดมากกว่า HTML ดังนั้นในที่นี้จึงเห็นการใช้ tag </th> ซึ่งไม่เคยใช้มาก่อน อันที่จริงได้ระบุไว้ในมาตรฐานเลยว่าให้ปฏิเสธการปฏิบัติตามคำสั่งใดๆ ที่ใช้รูปแบบไม่ถูกต้องแม้ว่าบราวเซอร์จะทราบความหมายที่แท้จริงของคำสั่งนั้นก็ตาม แต่บราวเซอร์ก็สามารถที่จะแสดงข้อความเตือนในคำสั่งส่วนที่ใช้ไม่ถูกต้องได้

คำสั่ง <xsl:for-each select=book_list/book”>

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="/">

<html>
<body>

<table border="2">
  <tr>
    <th> Title</th>
    <th> Author</th>
    <th> Year </th>
  </tr>

  <xsl:for-each select="book_list/book">
  <tr>
    <td> <xsl:value-of select="title"/> </td>
    <td> <xsl:value-of select="author"/> </td>
    <td> <xsl:value-of select="year"/> </td>
 </tr>
 </xsl:for-each>
</table>

</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

รูป 7-32
A style sheet ใน ภาษา XSL

นั้นเปรียบเทียบได้กับคำสั่ง "for" ในภาษา "C" ซึ่งจะบังคับให้บราวเซอร์ทำการวนซ้ำในส่วน body (บอกจุดสิ้นสุดของ body ด้วย tag <xsl:for-each>) หนึ่งรอบต่อหนังสือแต่ละเล่ม การวนซ้ำแต่ละรอบจะแสดง output ออกมา 5 บรรทัด นั่นคือ <tr>, title, author, year, และ </tr> ผลลัพธ์ของการแปลความหมายใน style sheet นี้เหมือนกับการสร้าง table ขึ้นมาโดยใช้ภาษา HTML ดังที่ได้กล่าวมาแล้ว อย่างไรก็ตาม ในรูปแบบนี้ โปรแกรมจะสามารถวิเคราะห์ XML ไฟล์และสามารถทราบได้ว่าหนังสือเล่มใดที่พิมพ์หลังจากปี 2000 ได้อย่างง่ายดาย

XHTML-The eXtensible HyperText Markup Language

ภาษา HTML ได้รับการพัฒนามาอย่างต่อเนื่องเพื่อให้สามารถตอบสนองความต้องการใหม่ๆ ที่เกิดขึ้นได้ ผู้คนจำนวนหนึ่งมีแนวความคิดว่าต่อไปในอนาคตเครื่องใช้ประเภทที่สามารถสื่อสารได้แบบไร้สายจะเข้ามาแทนที่เครื่องพีซีในส่วนที่เกี่ยวข้องกับงานทางเว็บ อุปกรณ์เหล่านี้มีปริมาณหน่วยความจำที่จำกัดจึงไม่สามารถที่จะบรรจุบราวเซอร์ที่เต็มไปด้วยขีดความสามารถด้านต่างๆ รวมทั้งการแก้ไขเว็บเพจที่เขียนขึ้นมาอย่างไม่ถูกต้องได้ ดังนั้น HTML ต่อจากรุ่นที่ 4.0 จึงกลายเป็นรุ่น XHTML (eXtensible HTML) แทนที่จะเป็น HTML 5.0 เพราะมันคือ HTML 4.0 ที่จัดรูปแบบใหม่ให้เหมือนกับ XML

XHTML มีความแตกต่างจาก HTML 4.0 หลักๆ 6 เรื่องและเรื่องย่อยอีกหลายเรื่อง ประการแรก เพจใน XHTML และบราวเซอร์จะต้องเป็นไปตามมาตรฐานอย่างเคร่งครัด คุณสมบัติข้อนี้ได้มาจากคุณสมบัติของ XML นั่นเอง

ประการที่สอง tag และคุณสมบัติทั้งหมดจะต้องเขียนด้วยตัวอักษรตัวพิมพ์เล็กเท่านั้น

ประการที่สาม tag ที่เป็นตัวปิด </...> จะต้องมีคู่กับ tag ที่เปิดใช้ <...> เสมอ สำหรับ tag ที่ไม่มีตัวปิดเป็นของตัวเองจะต้องใช้เครื่องหมาย "/" วางไว้หน้าเครื่องหมาย ">" เช่น

ประการที่สี่ attribute จะต้องเขียนอยู่ภายใต้เครื่องหมายคำพูด เช่น นั้นไม่สามารถใช้ได้อีกต่อไป ที่ถูกต้องคือ ตัวเลข 500 จะต้องเขียนอยู่ภายใต้เครื่องหมายคำพูดแม้ว่า 500 จะหมายความถึงตัวเลขก็ตาม

ประการที่ห้า tag เปิดและปิดจะต้องอยู่ในลำดับที่ถูกต้อง เช่น   Vacation Pictures   นั้นเคยอนุโลมให้ใช้ได้แต่ต่อไปจะต้องอยู่ในลำดับที่ถูกต้องเท่านั้นคือ   Vacation Pictures

ประการที่หก เอกสารทุกฉบับจะต้องระบุชนิดของเอกสารนั้น ดังที่แสดงตัวอย่างในรูป 7-32

7.3.3 เอกสารแบบพลวัต

เว็บเพจเท่าที่กล่าวถึงมานี้ยังคงเป็นการทำงานในลักษณะที่แสดงในรูป 6-6 นั่นคือผู้ใช้บริการ จัดส่งชื่อไฟล์ไปยังเซิบเวอร์ซึ่งจะจัดการส่งไฟล์นั้นกลับคืนมา ในยุคนั้นเว็บเพจประกอบด้วยไฟล์ที่มีข้อมูลเหมือนเดิมตลอดเวลาจนกว่าผู้สร้างเว็บจะทำการแก้ไขเปลี่ยนแปลงข้อมูลในเว็บเพจนั้นๆ อย่างไรก็ตาม ในหลายปีที่ผ่านมานี้ข้อมูลที่แสดงในเว็บเพจได้กลายเป็นข้อมูลที่เปลี่ยนแปลงได้ตลอดเวลาที่เรียกว่า

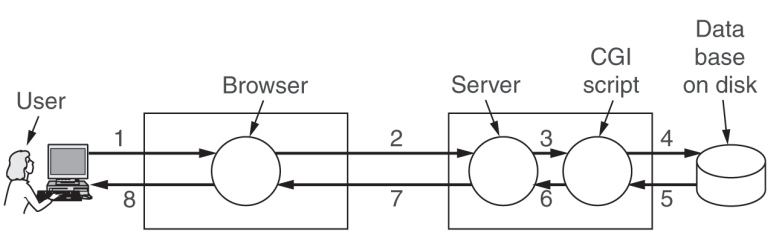
Dynamic web page นั่นคือเว็บเพจจะถูกสร้างขึ้นมาตามความต้องการการใช้งานมากกว่าที่จะถูกเก็บไว้ในดิสก์ การสร้างข้อมูลเพื่อนำเสนอนั้นอาจเกิดขึ้นทางฝั่งเซิบเวอร์หรือทางฝั่งผู้ใช้ก็ได้

การสร้างเว็บเพจแบบพลวัตทางฟั่งเซิบเวอร์

เพื่อให้เห็นว่าการสร้างเว็บเพจขึ้นมาโดยฝั่งเซิบเวอร์นั้นมีความจำเป็นอย่างไรก็ให้พิจารณาตัวอย่างการสร้างฟอร์มที่กล่าวถึงมาแล้ว เมื่อผู้ใช้ป้อนข้อมูลเข้าสู่ฟอร์มและคลิ๊กไปที่ปุ่ม submit ข้อมูลนั้นจะถูกส่งมาที่เซิบเวอร์ซึ่งประกอบด้วยชื่อเขตข้อมูลและข้อมูลที่ผู้ใช้ป้อนเข้ามา ข้อมูลนี้ไม่ใช่ชื่อของไฟล์อีกต่อไป สิ่งที่ต้องการก็คือข้อมูลนี้จะต้องถูกส่งต่อไปที่โปรแกรมหรือ script เพื่อทำการประมวลผล โดยปกติการประมวลผลจะเกี่ยวข้องกับข้อมูลที่ผู้ใช้ป้อนเข้ามาซึ่งจะถูกนำไปใช้ในการค้นหาข้อมูลในฐานข้อมูลที่เก็บอยู่ในดิสก์ของเครื่องเซิบเวอร์และนำมาสร้างเป็น HTML เพจที่มีความเหมาะสมกับผู้ใช้รายนั้นซึ่งก็จะถูกส่งกลับมาแสดงผลบนบราวเซอร์ของผู้ใช้ ตัวอย่างเช่น ในโปรแกรมประยุกต์ทางด้าน e-commerce เมื่อผู้ใช้คลิ๊กที่ "PROCEED TO CHECKOUT" บราวเซอร์จะส่ง cookie ที่ประกอบด้วยรายการสินค้าที่ผู้ใช้ต้องการซื้อกลับคืนมา โดยที่โปรแกรมบางตัวหรือ script บางอย่างที่ทางฝั่งเซิบเวอร์จะต้องถูกเรียกขึ้นมาเพื่อจัดการ cookie นั้นและสร้างเป็น HTML เพจขึ้นมาแสดงให้ผู้ใช้ได้ทราบ HTML เพจอาจจะแสดงรายการสินค้าที่ผู้ใช้เลือกซื้อและที่อยู่สำหรับการส่งของที่เกิดขึ้นครั้งล่าสุดของผู้ใช้รายนั้น รวมทั้งต้องการให้ผู้ใช้ตรวจสอบรายการสินค้า ยืนยันการสั่งซื้อและบอกวิธีการชำระเงิน ขั้นตอนที่จำเป็นต้องใช้ในการประมวลผลข้อมูลลูกค้าที่ได้รับมาจากฟอร์มแสดงให้เห็นในรูป 7-33

วิธีการดั้งเดิมที่ใช้ในการจัดการฟอร์มและข้อมูลแบบโต้ตอบทันทีอื่นๆ ของเว็บเพจนั้นคือระบบที่เรียกว่า CGI (Common Gateway Interface) ซึ่งหมายถึงวิธีการมาตรฐานของส่วนติดต่อที่ยอมให้เว็บเซิบเวอร์สามารถติดต่อสื่อสารกับโปรแกรม back-ends และ scripts ที่ยอมรับข้อมูลเข้ามาทำการประมวลผลและสร้างเป็น HTML เพจในการตอบสนอง โดยปกติโปรแกรม back-end และ script เหล่านี้จะถูกเขียนขึ้นมาโดยใช้ภาษา Perl scripting language เนื่องจากภาษานี้มีความง่ายและสามารถเขียนขึ้นมาใช้งานได้อย่างรวดเร็ว ตามที่ทราบกันโดยทั่วไปโปรแกรมเหล่านี้จะถูกเก็บไว้ในไดเร็กทอรี่ชื่อ cgi-bin ซึ่งสามารถอ้างอิงถึงได้โดยใช้ URL บางครั้งก็อาจใช้ภาษาอื่นเช่น Python ก็ได้

เพื่อเป็นตัวอย่างให้เห็นว่า CGI ทำงานอย่างไรลองพิจารณากรณีที่ลูกค้าไปซื้อสินค้าจากบริษัทแห่งหนึ่งซึ่งไม่มีใบรับประกันสินค้าอยู่ด้วยแต่ทางบริษัทให้ลูกค้าเข้าไปที่ "www.tgpc.com" เพื่อทำการลงทะเบียนแบบออนไลน์ (on-line) แทน ในเพจที่ลงทะเบียนนั้นจะมีข้อความ hyperlink เขียนว่า



1. User fills in form
2. Form sent back
3. Handed to CGI
4. CGI queries DB
5. Record found
6. CGI builds page
7. Page returned
8. Page displayed

รูป 7-33 ขั้นตอนการประมวลผลข้อมูลที่ได้รับมาจากฟอร์มของ HTML

Click here to register your product

ข้อความนี้ได้เชื่อมโยงไปยัง Perl script สมมุติว่าเป็นที่ "www.tgpc.com/cgi-bin/reg.perl" เมื่อ script นี้ถูกเรียกใช้โดยไม่มีพารามิเตอร์ส่งมาด้วย มันจะส่ง HTML เพจที่เป็นแบบฟอร์มใบลงทะเบียนมาให้ เมื่อลูกค้าเติมข้อความลงไปในฟอร์มนี้และคลิ๊กที่ submit ข้อมูลที่เติมเข้าไปจะถูกส่งไปยัง script โดยใช้ style อย่างเช่นที่แสดงในรูป 7-30 Perl script จะทำการตรวจสอบพารามิเตอร์ต่างๆ จัดการนำข้อมูลใหม่ใส่เข้าไปในฐานข้อมูลสำหรับลูกค้าใหม่ และจัดการส่ง HTML เพจกลับมาเพื่อบอกหมายเลขลงทะเบียนและเบอร์โทรศัพท์ติดต่อต่างๆ แม้ว่าจะมีวิธีการมากมายที่ใช้ในการจัดการกับฟอร์มแต่วิธีนี้เป็นวิธีที่นิยมใช้กันมากที่สุด

CGI script ไม่ใช่วิธีการเดียวที่นำมาใช้ในการสร้างเว็บเพจแบบพลวัต (dynamic content) ทางฝั่งผู้ให้บริการ อีกวิธีการหนึ่งที่นำมาใช้ได้คือการใส่ script ขนาดเล็กเข้าไปใน HTML เพจและให้เซิบเวอร์ทำการประมวลผลเพื่อสร้างเพจนั้นขึ้นมา ภาษาที่นิยมใช้ในการทำงานแบบนี้คือ PHP (PHP: Hypertext Preprocessor) เครื่องเซิบเวอร์จะต้องเข้าใจภาษา PHP (แบบเดียวกันกับที่บราวเซอร์เข้าใจภาษา XML) โดยปกติไฟล์ประเภทนี้จะใช้นามสกุลไฟล์เป็น ".php" แทนที่จะเป็น ".htm" หรือ ".html"

รูป 7-34 แสดงตัวอย่าง PHP script ขนาดเล็กซึ่งสามารถที่จะทำงานร่วมกับเซิบเวอร์ใดๆ ที่ติดตั้ง PHP เอาไว้ จากตัวอย่างจะเห็นว่าเป็นไฟล์ HTML ธรรมดายกเว้นในส่วนที่ใช้ tag <?php ... ?> ในที่นี้สิ่งที่จะเกิดขึ้นก็คือเว็บเพจจะถูกสร้างขึ้นมาเพื่อบอกว่าบราวเซอร์ได้ร้องขอข้อมูลอะไรบ้าง โดยปกติบราวเซอร์จะส่งข้อมูลบางส่วนมาพร้อมกับคำร้องขอข้อมูล (และอาจมี cookie ติดมาด้วย) ข้อมูลนี้จะถูกนำไปใส่ไว้ในตัวแปรที่ชื่อว่า "HTTP_USER_AGENT" สมมุติว่าข้อความในรูป 7-34 ถูกใส่เข้าไปในไฟล์ "test.php" ใน WWW ไดเร็กทอรี่ของบริษัท ABCD เมื่อใช้บราวเซอร์เข้ามาที่ "www.abcd.com/test.php" เซิบเวอร์จะสร้างเว็บเพจเพื่อบอกให้ผู้ใช้ได้ทราบว่า กำลังใช้บราวเซอร์อะไร ภาษาอะไร และระบบปฏิบัติการแบบไหนอยู่

PHP มีความเหมาะสมในการจัดการฟอร์มได้ดีเป็นพิเศษและมีความง่ายในการใช้งานมากกว่า CGI script เพื่อดูตัวอย่างการทำงานเกี่ยวกับฟอร์มให้ดูรูป 7-35(a) รูปนี้แสดง HTML เพจที่เป็นแบบฟอร์มทั่วไป สิ่งที่แปลกกว่าธรรมดาก็คือคำสั่งในบรรทัดแรกที่กำหนดว่าไฟล์ "action.php" จะถูกเรียกขึ้นมาเพื่อใช้จัดการกับพารามิเตอร์ภายหลังจากที่ผู้ใช้ได้กรอกแบบฟอร์มนี้และส่งกลับมาแล้ว เพจนี้แสดง textbox 2 แห่ง แห่งหนึ่งคือการขอชื่อผู้ใช้ และอีกแห่งหนึ่งขอทราบอายุ หลังจากที่ textbox ทั้งสองได้ถูกเติมข้อมูลเข้ามาแล้ว เซิบเวอร์จะทำการตรวจดูพารามิเตอร์แล้วใส่ชื่อผู้ใช้เข้าไปที่ "name" และอายุไว้ที่ "age" จากนั้นก็จะเริ่มทำการประมวลผล "action.php" ดังที่แสดงในรูป 7-35(b) เพื่อสร้างเป็นเพจตอบรับส่งมาให้ผู้ใช้ ในระหว่างที่ทำการประมวลผลไฟล์นี้ คำสั่ง PHP จะถูกเรียกมาใช้งาน ถ้าผู้ใช้พิมพ์ข้อความ "Barbara" และ "24" เข้ามาใน textbox เซิบเวอร์จะสร้างเว็บเพจดังที่แสงในรูป

```
<html>
<body>
<h2> This is what I know about you </h2>
<?php echo $HTTP_USER_AGENT ?>
</body>
</html>
```

รูป 7-34 ตัวอย่างไฟล์ HTML ที่มี PHP แฝงอยู่ด้วย

```
<html>
<body>
<form action="action.php" method="post">
<p> Please enter your name: <input type="text" name="name"> </p>
<p> Please enter your age: <input type="text" name="age"> </p>
<input type="submit">
</form>
</body>
</html>
```

(a)

```
<html>
<body>
<h1> Reply: </h1>
Hello <?php echo $name; ?>.
Prediction: next year you will be <?php echo $age + 1; ?>
</body>
</html>
```

(b)

```
<html>
<body>
<h1> Reply: </h1>
Hello Barbara.
Prediction: next year you will be 25
</body>
</html>
```

(c)

รูป 7-35
(a) แบบฟอร์ม
(b) PHP script
(c) ผลที่ได้รับจากการประมวลคำสั่ง PHP script

7-35(c) ส่งกลับไปที่ผู้ใช้ ดังนั้นการจัดการเกี่ยวกับฟอร์มจึงกลายเป็นเรื่องง่ายเมื่อใช้ PHP

แม้ว่า PHP จะง่ายในการใช้งานแต่ตัว PHP เองก็เป็นภาษาที่มีขีดความสามารถสูงมากในการเป็นตัวเชื่อมต่อระหว่างเว็บและฐานข้อมูลของเซิบเวอร์ PHP มีตัวแปร, strings, arrays, และคำสั่งควบคุมการทำงานเป็นจำนวนมากที่สามารถพบได้ในภาษาคอมพิวเตอร์ เช่นภาษาซี แต่คำสั่งบางส่วนก็มีประสิทธิภาพมากกว่าด้วย PHP เป็นภาษาประเภท Open source code และแจกจ่ายให้โดยไม่คิดค่าใช้จ่าย PHP ถูกออกแบบมาให้สามารถทำงานร่วมกับซอฟต์แวร์ Apache ซึ่งเป็น Open source code เหมือนกัน และเป็นเว็บเซิบเวอร์ที่มีผู้นำไปใช้งานมากที่สุดในโลก

วิธีการแบบที่สามในการสร้างเว็บเพจพลวัต (dynamic) คือการใช้ JSP (JavaServer Pages) ซึ่งมีความคล้ายคลึงกับ PHP ยกเว้นในส่วนที่เป็นข้อมูลพลวัตนั้นจะถูกสร้างขึ้นมาด้วยภาษาจาวาแทนที่จะเป็น PHP เว็บเพจที่สร้างขึ้นมาด้วยวิธีนี้จะใช้นามสกุล ".jsp" วิธีการแบบที่สี่เรียกว่า ASP (Active Server Pages) ซึ่งเป็นของบริษัทไมโครซอฟต์ที่พัฒนาขึ้นมาเพื่อแข่งขันกับ PHP และ JSP วิธีการนี้จะใช้ภาษา Visual Basic Script ในการสร้างข้อมูลพลวัต และใช้นามสกุลไฟล์เป็น ".asp" สำหรับผู้ใช้งานทั่วไป การที่จะเลือกใช้ ASP PHP หรือ JSP นั้นมีความแตกต่างกันน้อยมากเพราะทั้งสามวิธีมีความสามารถใกล้เคียงกัน จึงอยู่ที่ว่าผู้ใช้ชอบผลิตภัณฑ์ของบริษัทใดมากกว่ากันเท่านั้น การสร้างเนื้อหาของเว็บเพจในขณะที่เว็บเพจนั้นถูกเรียกใช้งานนี้เรียกว่า dynamic HTML

การสร้างเว็บเพจแบบพลวัตทางฝั่งผู้ใช้บริการ

CGI PHP JSP และ ASP scripts ได้ช่วยแก้ปัญหาในการจัดการกับฟอร์มและการติดต่อกับฐานข้อมูลบนเครื่องเซิบเวอร์ ช่วยเพิ่มขีดความสามารถในการรับข้อมูลมากจากฟอร์ม ค้นหาข้อมูลจากฐานข้อมูล และนำข้อมูลที่ค้นพบมาสร้างเป็น HTML เพจ แต่สิ่งที่เทคนิคเหล่านี้ไม่สามารถทำได้ก็คือการตอบสนองการเคลื่อนที่ของเมาส์ หรือการโต้ตอบกับผู้ใช้โดยตรง สำหรับวัตถุประสงค์นี้ มีความจำเป็นที่จะต้องสร้าง script ไว้ใน HTML เพจซึ่งจะถูกประมวลผลด้วยเครื่องของผู้ใช้เองแทนที่จะเป็นเครื่องเซิบเวอร์ ตั้งแต่ HTML 4.0 เป็นต้นมาผู้สร้างเว็บสามารถแทรก script เข้าไว้ภายใน HTML เพจได้โดยใช้ tag <script> ภาษา script ที่นิยมนำมาใช้งานมากที่สุดคือ JavaScript

JavaScript เป็นภาษาคอมพิวเตอร์ประเภท scripting language ซึ่งได้รับการพัฒนามาจากภาษาจาวาสำหรับการโปรแกรม อย่างไรก็ตาม JavaScript ไม่ใช่ภาษาจาวา ภาษานี้มีความสามารถมาก เช่น ในคำสั่งเพียงบรรทัดเดียวสามารถที่จะสร้าง box สำหรับการสนทนา รอคอยข้อมูลที่ผู้ใช้จะต้องป้อนเข้ามา และจัดเก็บผลลัพธ์ไว้ในตัวแปรได้ ความสามารถเช่นนี้ทำให้ JavaScript มีความเหมาะสมในการออกแบบเว็บเพจที่สามารถโต้ตอบกับผู้ใช้ได้ แต่ในทางกลับกันเนื่องจากภาษานี้ไม่ได้รับการกำหนดมาตรฐานขึ้นใช้งานและมีการพัฒนาขึ้นมาใช้อย่างรวดเร็ว ทำให้เว็บเพจที่เขียนขึ้นมาโดยใช้ภาษา JavaScript นั้นมีความเป็นมาตรฐานน้อยมากจนไม่สามารถที่จะนำไปใช้งานได้กับเครื่องคอมพิวเตอร์ทุกประเภท

รูป 7-36 แสดงตัวอย่างเว็บเพจที่เขียนขึ้นมาโดยใช้ JavaScript เว็บเพจนี้จะแสดงแบบฟอร์มเพื่อขอชื่อและที่อยู่ของผู้ใช้จากนั้นจะคำนวณอายุของผู้ใช้ในปีหน้า ส่วนที่เป็น body ของเพจนั้นแทบจะเหมือนกับตัวอย่างของ PHP สิ่งที่แตกต่างกันก็คือการกำหนดปุ่ม submit และการกำหนดค่าให้กับตัวแปรซึ่งจะใช้วิธีการเรียก script ชื่อ response ขึ้นมาเมื่อผู้ใช้คลิ๊กเมาส์ที่ปุ่ม submit และจัดการส่งค่าที่ได้ไปให้กับตัวแปร

สิ่งที่ใหม่ในที่นี้ก็คือการกำหนดฟังก์ชัน response ในส่วน head ของ HTML ไฟล์ ฟังก์ชันนี้จะทำการสำเนาค่าที่เก็บอยู่ในตัวแปร "name" ของฟอร์มเข้าไปเก็บไว้ในตัวแปร "person" ที่มีใช้ในฟังก์ชัน และจัดการสำเนาค่าใน "age" แปลงเป็นเลขจำนวนเต็มด้วยฟังก์ชัน "eval" จัดการบวกค่าที่ได้เข้าไปอีก "1" และเก็บผลลัพธ์ไว้ที่ตัวแปร "years" จากนั้นเปิดเว็บเพจสำหรับการสร้างผลลัพธ์ ทำการเขียนข้อมูลลงไป 4 ครั้งโดยใช้คำสั่ง "writeln" และปิดเว็บเพจ เว็บเพจที่ได้จะเป็น HTML เพจซึ่งสามารถแสดงผลได้ในบราวเซอร์ทั่วไป

ผู้อ่านจำเป็นที่จะต้องทำความเข้าใจว่า script ในรูป 7-35 นั้นมีความคล้ายคลึงกับในรูป 7-36 มาก แต่การประมวลผลนั้นแตกต่างกันโดยสิ้นเชิง ในรูป 7-35 หลังจากที่ผู้ใช้คลิ๊กที่ปุ่ม submit แล้วบราวเซอร์จะเก็บรวบรวมข้อมูลเข้าไว้ในสายอักขระ (string) ที่ยาวมาก (ในรูปแบบที่แสดงในรูป 7-30) และส่งไปยังเซิบเวอร์ที่เป็นผู้ส่งเว็บเพจนั้นเข้ามา เมื่อเซิบเวอร์มองเห็นชื่อ PHP ไฟล์ก็จะจัดการประมวลผล PHP script จะสร้างเว็บเพจขึ้นมาใหม่และจัดการส่งเว็บเพจนั้นกลับไปยังผู้ใช้อีกครั้งหนึ่งเพื่อแสดงผลบนบราวเซอร์ของผู้ใช้ ส่วนในรูป 7-36 เมื่อผู้ใช้คลิ๊กที่ปุ่ม submit แล้วบราวเซอร์ของผู้ใช้จะทำการแปลความหมายของ JavaScript ฟังก์ชันที่มีอยู่ในเว็บเพจนั้น การประมวลผลทั้งหมดจะเกิดขึ้นภายในบราวเซอร์ในเครื่องคอมพิวเตอร์ของผู้ใช้ จะไม่มีการติดต่อกับเซิบเวอร์เกิดขึ้นเลย (ในช่วงนี้) ผลที่

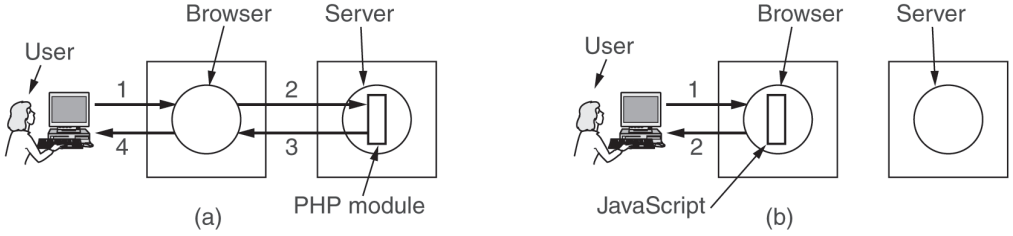
```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
   var person = test_form.name.value;
   var years = eval(test_form.age.value) + 1;
   document.open();
   document.writeln("<html> <body>");
   document.writeln("Hello " + person + ".<br>");
   document.writeln("Prediction: next year you will be " + years + ".");
   document.writeln("</body> </html>");
   document.close();
}
</script>
</head>

<body>
<form>
Please enter your name: <input type="text" name="name">
<p>
Please enter your age: <input type="text" name="age">
<p>
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

รูป 7-36
ตัวอย่างการใช้ JavaScript

เกิดขึ้นก็คือผลลัพธ์ที่ได้จะสามารถแสดงผลบนบราวเซอร์ได้ในเกือบจะทันทีในขณะที่วิธี PHP อาจต้องรอหลายวินาทีกว่าที่จะเห็นการแสดงผล ความแตกต่างของการประมวลผลที่ server-side scripting กับ client-side scripting นั้นแสดงให้เห็นในรูป 7-37

ความแตกต่างนี้ไม่ได้บ่งบอกว่า JavaScript นั้นดีกว่า PHP เนื่องจากวิธีการทำงานนั้นแตกต่างกัน PHP (รวมทั้ง ASP และ JSP) มักจะนำมาใช้ในการติดต่อกับฐานข้อมูลที่อยู่ในเครื่องเซิบเวอร์ ส่วน JavaScript นั้นจะใช้เมื่อต้องการการโต้ตอบกับผู้ใช้ที่เครื่องผู้ใช้เอง ดังนั้นจึงมีความเป็นไปได้ที่ในเว็บเพจหนึ่งอาจมีทั้ง PHP และ JavaScript อยู่ด้วยแม้ว่าทั้งสองนี้จะไม่ถูกนำมาใช้ทำงานเดียวกันและใช้ปุ่มบังคับปุ่มเดียวกัน

JavaScript นั้นเป็นภาษาที่ได้รับการขยายขีดความสามารถเป็นอย่างมากที่มีทั้งขีดความสามารถของภาษาซีและภาษาจาวาอยู่ในตัวเดียวกัน ภาษานี่มีทั้งตัวแปร strings, arrays, objects, functions, และฟังก์ชันควบคุมการทำงานที่สำคัญทั้งหมด ภาษานี้ยังมีสิ่งอำนวยความสะดวกที่สร้างขึ้นมาเพื่อเว็บเพจโดยเฉพาะรวมทั้งความสามารถในการจัดการ windows และ frames, set and get cookie, forms และการจัดการ hyperlink รูป 7-38 แสดงตัวอย่าง JavaScript โปรแกรมที่ใช้ recursive function



รูป 7-37
(a) Server-side scripting with PHP
(b) Client-side scripting with JavaScript

รูป 7-38
โปรแกรม JavaScript ที่ใช้คำนวณและแสดงผลค่า factorial

```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
   function factorial(n) {if (n == 0) return 1; else return n * factorial(n – 1);}
   var r = eval(test_form.number.value);      //  r = typed in argument
   document.myform.mytext.value = "Here are the results.\n";
   for (var i = 1; i <= r; i++)                        // print one line from 1 to r
      document.myform.mytext.value += (i + "! =  " + factorial(i) + "\n");
}
</script>
</head>
<body>
<form name="myform">
Please enter a number: <input type="text" name="number">
<input type="button" value="compute table of factorials" onclick="response(this.form)">
<p>
<textarea name="mytext" rows=25 cols=50> </textarea>
</form>
</body>
</html>
```

JavaScript ยังมีความสามารถในการติดตามการเคลื่อนที่ของเมาส์ผ่านอ็อบเจ็คต่าง ๆ ที่อยู่บนหน้าจอภาพได้ด้วย เว็บเพจที่เขียนด้วย JavaScript จำนวนมากมีคุณสมบัติที่เมื่อเคลื่อนเมาส์ผ่านข้อความบางส่วนหรือรูปภาพบางรูปแล้วจะมีอะไรบางอย่างเกิดขึ้นได้ เช่น มีเสียงหรือมีคำอธิบายเพิ่มเติมเกิดขึ้น รูปภาพอาจเปลี่ยนไป หรือมีเมนูเกิดขึ้น เป็นต้น พฤติกรรมเหล่านี้สามารถโปรแกรมให้ทำงานได้โดยง่ายด้วย JavaScript ซึ่งทำให้เว็บเพจมีความน่าตื่นตาตื่นใจมากยิ่งขึ้นดังแสดงตัวอย่างในรูป 7-39

JavaScript ไม่ใช่วิธีการเดียวที่ทำให้เว็บเพจมีชีวิตชีวามากขึ้น อีกวิธีการหนึ่งที่นำมาใช้กันคือ การใช้ applets ซึ่งเป็นโปรแกรมจาวาขนาดเล็กที่ได้ถูกแปลงเป็นภาษาเครื่องคอมพิวเตอร์และพร้อมที่จะทำงานบนเครื่องคอมพิวเตอร์เสมือนที่เรียกว่า JVM (Java Virtual Machine) Applet สามารถที่จะแทรกตัวอยู่ใน HTML เพจได้ (อยู่ระหว่าง tag <applet> และ </applet>) และสามารถแสดงผลได้โดยบราวเซอร์ที่สนับสนุน JVM เนื่องจาก applet ถูกแปลความหมายในลักษณะของ interpretation แทนที่จะเป็นการประมวลผลโดยตรง ตัว Java interpreter จึงสามารถถูกโปรแกรมให้ป้องกันการทำงานที่อาจสร้างผลร้ายให้แก่ผู้ใช้ได้ (ในทางทฤษฎี) ในความเป็นจริงแล้ว ผู้ที่เขียน applet จะพบข้อบกพร่องที่ไม่มีที่สิ้นสุดใน Java I/O libraries ซึ่งสามารถนำมาใช้ประโยชน์ (ในทางร้าย) ได้

บริษัทไมโครซอฟต์ได้ตอบโต้การพัฒนา Java applet ด้วยการนำเทคโนโลยีที่เรียกว่า ActiveX control มาใช้ ซึ่งหมายถึงโปรแกรมที่ถูก compiled (แปลเป็นภาษาเครื่องที่พร้อมจะทำงานในทันที) บนเครื่องพีซีตระกูล Pentium ที่สามารถทำการประมวลผลได้โดยตรง วิธีการนี้ทำให้โปรแกรมสามารถถูกประมวลผลและทำงานได้ด้วยความเร็วสูงสุดและมีความอ่อนตัวมากกว่า Java applet เพราะมันสามารถทำงานได้ทุกอย่างเท่าที่โปรแกรมหนึ่ง ๆ จะสามารถทำได้ เมื่อบราวเซอร์ เช่น Internet Explorer มองเห็น ActiveX control อยู่ในเว็บเพจ ก็จะทำการดาวน์โหลดโปรแกรมนั้นมา ทำการ

```
<html>
<head>
<script language="javascript" type="text/javascript">
if (!document.myurl) document.myurl = new Array();
document.myurl[0] = "http://www.cs.vu.nl/ ast/im/kitten.jpg";
document.myurl[1] = "http://www.cs.vu.nl/ ast/im/puppy.jpg";
document.myurl[2] = "http://www.cs.vu.nl/ ast/im/bunny.jpg";
function pop(m) {
   var urx = "http://www.cs.vu.nl/ ast/im/cat.jpg";
   popupwin = window.open(document.myurl[m],"mywind","width=250,height=250");
}
</script>
</head>
<body>
<p> <a href="#" onMouseover="pop(0); return false;" > Kitten  </a> </p>
<p> <a href="#" onMouseover="pop(1); return false;" > Puppy  </a> </p>
<p> <a href="#" onMouseover="pop(2); return false;" > Bunny  </a> </p>
</body>
</html>
```

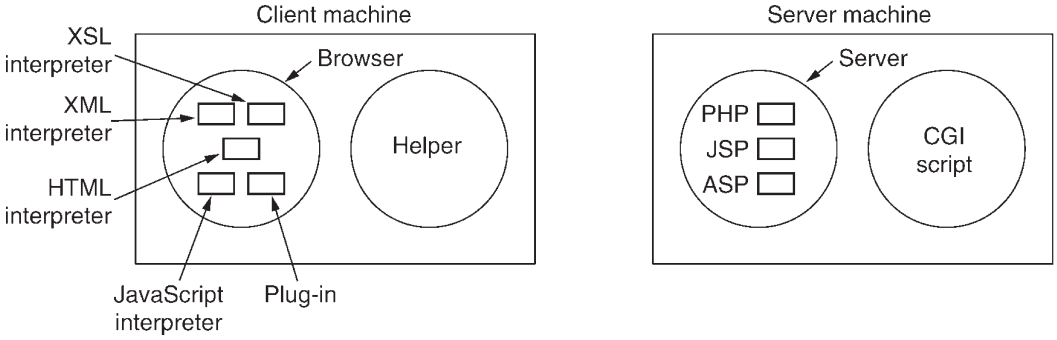
รูป 7-39
เว็บเพจที่โต้ตอบกับผู้ใช้ได้ที่ตอบสนองต่อการเคลื่อนที่ของเมาส์

ตรวจสอบตัวตน (identity) ของโปรแกรมนั้น และดำเนินการประมวลผล อย่างไรก็ตาม การดาวน์โหลดโปรแกรมและการประมวลผลโปรแกรมอื่นบนเครื่องของผู้ใช้นั้นได้สร้างปัญหาในเรื่องความปลอดภัยขึ้นมาซึ่งจะได้กล่าวถึงในบทที่ 8

เนื่องจากบราวเซอร์เกือบทุกตัวสามารถแปลงความหมายของโปรแกรม Java และ JavaScript ได้ นักออกแบบที่ต้องการสร้างเว็บเพจที่มีการโต้ตอบกับผู้ใช้ได้ดีจึงมีทางเลือกอย่างน้อยสองทาง และถ้าไม่สนใจที่จะใช้งานเว็บเพจนั้นบนเครื่องอื่นนอกเหนือจากเครื่องพีซีแล้ว ActiveX control ก็เป็นอีกทางเลือกหนึ่ง กฎทั่วไปที่นำมาใช้พิจารณาทางเลือกทั้งสามทางนี้ก็คือ JacaScript นั้นเขียนขึ้นมาได้ง่าย Java applet สามารถประมวลผลได้เร็วกว่า และ ActiveX control สามารถประมวลผลได้เร็วที่สุด และเนื่องจากบราวเซอร์ทั้งหมดสร้าง JVM ขึ้นมาเหมือนกันทุกตัว แต่บราวเซอร์จำนวนน้อยที่จะแปลความหมายของ JavaScript ได้เหมือนกัน Java applet จึงมีความสามารถในการนำไปใช้งานได้กว้างขวางกว่าการใช้ JavaScript

ในหัวข้อนี้สามารถสรุปได้ว่าเว็บเพจที่สมบูรณ์สามารถสร้างขึ้นมาได้ในขณะที่ใช้งานโดยการใช้ script แบบต่าง ๆ บนเครื่องเซิบเวอร์ เมื่อบราวเซอร์ได้รับผลแล้วเว็บเพจนั้นก็จะเป็นเสมือน HTML เพจธรรมดาที่สามารถนำแสดงบนหน้าจอได้ Script อาจเป็นแบบ Perl, PHP, JSP หรือ ASP ดังแสดงในรูป 7-40

การสร้างเว็บเพจอาจเกิดขึ้นได้ที่ฝั่งของผู้ใช้ เว็บเพจอาจถูกเขียนขึ้นมาโดย XML แล้วจึงถูกแปลงกลับไปเป็น HTML เพจตามข้อกำหนดที่ระบุไว้ใน XSL ไฟล์ โปรแกรม JavaScript สามารถทำการคำนวณบางอย่างได้ ท้ายที่สุด plug-in และ helper applications สามารถนำมาใช้แสดงผลข้อมูลได้ในหลายรูปแบบ



รูป 7-40
วิธีการต่างๆ ที่ใช้ใน การสร้างเว็บเพจ

7.3.4 HTTP-The HyperText Transfer Protocol

โพรโตคอลที่ใช้ในการถ่ายทอดข้อมูลผ่าน World Wide Web นั้นคือ HTTP (HyperText Transfer Protocol) ที่กำหนดไว้ในมาตรฐาน RFC 2616 โพรโตคอลนี้จะกำหนดข้อมูลที่ผู้ใช้สามารถส่งไปยังเซิบเวอร์และข้อมูลที่เซิบเวอร์จะส่งคืนมาที่ผู้ใช้ ในแต่ละวงรอบการทำงานประกอบด้วยคำร้องขอข้อมูลที่อยู่ในรูป ASCII text ตามด้วยข้อมูลตอบสนองในรูปแบบที่คล้ายกับ มาตรฐาน RFC 822 MIME ผู้ใช้และเซิบเวอร์ทั้งหมดจะต้องปฏิบัติตามโพรโตคอลนี้

การเชื่อมต่อ

วิธีการปกติที่บราวเซอร์ใช้ในการติดต่อกับเซิบเวอร์คือ การจัดตั้งช่องทางการสื่อสาร TCP เชื่อมกับพอร์ต 80 บนเครื่องเซิบเวอร์แม้ว่าขั้นตอนนี้ไม่ได้ถูกกำหนดเป็นข้อบังคับไว้ก็ตาม คุณค่าของการใช้การเชื่อมต่อ TCP ก็คือทั้งผู้ใช้และเซิบเวอร์ไม่ต้องคอยพะวงกับการรับ-ส่งข้อมูลว่าจะสูญหายไปหรือไม่ มีข้อมูลซ้ำเกิดขึ้นหรือเปล่า ข้อมูลมีความยาวเกินไปหรือไม่ และจะต้องตอบรับอย่างไร เพราะว่าโพรโตคอล TCP คอยจัดการปัญหาทั้งหมดนี้ให้อยู่แล้ว

ใน HTTP 1.0 ภายหลังจากที่ได้จัดตั้งการเชื่อมต่อขึ้นมาแล้ว จะมีการส่งคำร้องขอข้อมูลไปครั้งหนึ่ง ซึ่งก็จะมีการส่งข้อมูลตอบรับมาครั้งหนึ่งเหมือนกัน จากนั้นการเชื่อมต่อ TCP ก็จะถูกยกเลิก ในโลกที่เต็มไปด้วยเว็บเพจที่ประกอบด้วย HTML text แต่เพียงอย่างเดียว วิธีการนี้จัดได้ว่าดีเพียงพอต่อการใช้งาน อย่างไรก็ตาม เว็บเพจในปัจจุบันนั้นประกอบด้วยไอคอนจำนวนมาก รูปภาพต่างๆ รวมทั้งข้อมูลพิเศษอื่นๆ อีกหลายชนิด ทำให้การจัดตั้งการเชื่อมต่อ TCP ในทุกครั้งที่มีการถ่ายทอดไอคอนแต่ละอันหรือรูปภาพแต่ละรูปนั้นกลายเป็นสิ่งที่ไม่มีประสิทธิภาพเอาเสียเลย

ข้อสังเกตนี้ได้นำไปสู่การพัฒนา HTTP 1.1 ซึ่งสนับสนุนการเชื่อมต่อแบบ persistent connections นั่นคือจะมีการจัดตั้งการเชื่อมต่อ TCP ขึ้นมา มีการส่งคำร้องขอข้อมูลและส่งข้อมูลตอบสนองกลับไป จากนั้นก็สามารถส่งคำร้องขอข้อมูลใหม่และส่งข้อมูลตอบสนองมาใหม่ได้เรื่อยๆ โดยใช้การเชื่อมต่อ TCP อันเดิม ด้วยการทำงานในลักษณะนี้ทำให้การใช้งาน TCP มีประสิทธิภาพมากยิ่งขึ้น อีกทั้งยังสามารถทำ pipeline คำสั่งได้ด้วย คือสามารถส่งคำร้องขอข้อมูลครั้งที่ 2 ไปได้ก่อนที่ข้อมูลตอบสนองของคำร้องแรกจะเดินทางมาถึงผู้รับ

Methods

แม้ว่า HTTP ได้ถูกออกแบบมาสำหรับการใช้งานบนเว็บโดยตรง แต่ก็ได้ถูกออกแบบมาดีพอที่จะนำมาใช้กับงานทั่วไปโดยมีเป้าหมายที่โปรแกรมประยุกต์เชิงวัตถุ (Object-oriented applications) ด้วยเหตุผลนี้จึงสนับสนุนสิ่งที่เรียกว่า methods นอกเหนือไปจากการร้องขอเว็บเพจ คำร้องขอข้อมูลแต่ละครั้งประกอบด้วยข้อมูลที่อยู่ในรูป ASCII text หนึ่งหรือสองบรรทัด โดยที่คำแรกที่อยู่ในแต่ละบรรทัดจะเป็นชื่อของ method ที่เรียกใช้ รูป 7-41 แสดงรายชื่อ method ที่สร้างมาพร้องกับโพรโตคอล HTTP ในการติดต่อกับอ็อบเจ็คทั่วไปก็อาจมี method เฉพาะของอ็อบเจ็คนั้นๆ เพิ่มเติมเข้ามาด้วย ชื่อของ method นั้นมีลักษณะ case sensitive นั่นคือ "GET" เป็นชื่อ method หนึ่งแต่ "get" นั้นไม่ใช่

Method "GET" ร้องขอให้เซิบเวอร์ส่งเว็บเพจ (ซึ่งในที่นี้ก็คืออ็อบเจ็คอย่างหนึ่ง แต่ในทางปฏิบัติมักจะหมายถึงไฟล์ไฟล์หนึ่ง) มาให้ ข้อมูลในเพจจะถูกเข้ารหัสแบบ MIME รูปแบบทั่วไปของ "GET" คือ

GET filename HTTP/1.1

โดยที่ filename เป็นชื่อของข้อมูล (ไฟล์) ที่ต้องการและ "1.1" เป็นรุ่นของโพรโตคอลที่ใช้

Method "HEAD" เป็นการร้องขอ message header โดยไม่มีเว็บเพจติดมาด้วย method นี้สามารถนำมาใช้ในการหาข้อมูลเกี่ยวกับเวลาที่เว็บเพจนั้นถูกแก้ไขเป็นครั้งสุดท้าย หรือเพื่อรวบรวมข้อมูลเพื่อการทำสารบัญข้อมูล หรือใช้ในการทดสอบ URL เพียงอย่างเดียว

Method "PUT" เป็นการทำงานตรงกันข้ามกับ "GET" นั่นคือแทนที่จะอ่านข้อมูลจากเซิบเวอร์ก็จะเป็นการบันทึกข้อมูลเว็บเพจไว้ที่เซิบเวอร์ วิธีการนี้ช่วยในการสร้างเว็บเพจมาจากเซิบเวอร์เครื่องอื่นที่อยู่ไกลออกไป ส่วน body ของคำร้องจะบรรจุเว็บเพจเอาไว้ซึ่งอาจจะเข้ารหัสแบบ MIME ก็ได้และในบรรทัดที่ตามหลัง PUT อาจจะรวมข้อมูลประเภท "Content-Type" และ authentication headers เพื่อใช้ในการพิสูจน์ว่าผู้ที่เรียกมานั้นได้รับอนุญาตให้ทำงานตามคำสั่งนี้

Method "POST" ทำงานคล้ายกับ "PUT" แต่แทนที่จะนำข้อมูลเว็บเพจใหม่มาแทนที่เว็บเพจเดิม "POST" จะนำข้อมูลใหม่มาเพิ่มเติมให้กับเว็บเพจเดิม ตัวอย่างเช่น การส่งข้อความใหม่ไปยัง newsgroup หรือการเพิ่มเติมไฟล์ข้อมูลเข้าไปใน bulletin board ในทางปฏิบัติทั้ง "PUT" และ "POST" ไม่ได้ถูกนำมาใช้งานมากนัก

Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

รูป 7-41
The built-in HTTP request methods

รูป 7-42
โค้ดแสดงสถานะ

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

Method "DELETE" ทำการลบข้อมูลทิ้ง ซึ่งการใช้ method นี้จำเป็นจะต้องมีการตรวจสอบผู้ใช้และการอนุญาตให้ใช้คำสั่งนี้ก่อนที่จะสามารถลบข้อมูลใด ๆ ทิ้งไปได้ อย่างไรก็ตาม ไม่มีการรับประกันว่าคำสั่งนี้จะทำงานโดยสมบูรณ์ เนื่องจากว่าเซิบเวอร์อาจจะต้องการลบข้อมูลทิ้ง แต่ไฟล์ ดังกล่าวอาจได้รับการปกป้องจากระบบปฏิบัติการคือไม่อนุญาตให้ HTTP เซิบเวอร์ทำการลบไฟล์นี้

Method "TRACE" ถูกใช้สำหรับตรวจสอบข้อผิดพลาดการทำงานที่เกิดขึ้น คำสั่งนี้จะบอกให้เซิบเวอร์ส่งคำร้องขอข้อมูลกลับไปที่ผู้ใช้ method นี้มีประโยชน์เมื่อคำร้องของข้อมูลนั้นไม่ได้รับการตอบสนองอย่างที่ควรจะเป็นและผู้ใช้ต้องการทราบว่าเซิบเวอร์ได้รับคำสั่งใดไปจากตนเอง

Method "CONNECT" ไม่มีการกำหนดใช้งานในปัจจุบัน

Method "OPTIONS" สนับสนุนวิธีการที่ให้ผู้ใช้สอบถามไปยังเซิบเวอร์เกี่ยวกับคุณสมบัติของตนเองหรือคุณสมบัติของไฟล์ใด ๆ

คำร้องขอข้อมูลทุกอันจะได้รับข้อมูลตอบสนองกลับมาประกอบด้วย สถานะและข้อมูลเพิ่มเติม (อาจเป็นส่วนหนึ่งหรือเว็บเพจทั้งหน้า) ข้อมูลสถานะประกอบด้วยโค้ดบอกสถานะที่เป็นตัวเลข 3 ตัว เพื่อบอกให้ทราบว่าคำร้องขอนั้นได้รับการตอบสนองด้วยดี แต่ถ้าไม่ได้รับการตอบสนองก็จะบอกให้ทราบว่าเป็นเพราะเหตุใด รหัสเลขตัวแรกนั้นใช้แบ่งการตอบสนองออกเป็น 5 กลุ่มใหญ่ ๆ ดังแสดงในรูป 7-42 โค้ด 1xx มักจะไม่ได้นำมาใช้ในทางปฏิบัติ โค้ด 2xx หมายความว่าคำร้องขอนั้นได้รับการตอบสนองเป็นอย่างดีและข้อมูลเว็บเพจนั้นกำลังถูกส่งกลับมา โค้ด 3xx บอกให้ผู้ใช้บริการไปขอข้อมูลจากที่อื่นโดยอาจจะใช้ URL หรือข้อมูลใน cache ของตนเองก็ได้ โค้ด 4xx หมายความว่าคำร้องขอข้อมูลนั้นล้มเหลวเนื่องจากข้อผิดพลาดที่มาจากผู้ใช้เอง เช่น ส่งคำร้องขอข้อมูลไม่ถูกต้อง หรือเว็บเพจ ที่ขอมานั้นไม่มีอยู่ในเซิบเวอร์ โค้ด 5xx บอกความผิดพลาดที่เกิดขึ้นที่เซิบเวอร์เอง เช่นอาจมีคำร้องขอข้อมูลเข้ามามากจนเซิบเวอร์ไม่สามารถจัดการได้

Message Headers

คำร้องขอข้อมูลที่ส่งมาในหนึ่งบรรทัดนั้น (เช่น บรรทัดคำสั่งที่ใช้ method "GET") อาจจะตามมาด้วยข้อมูลเพิ่มเติมในอีกหลายบรรทัดซึ่งเรียกว่า request header ข้อมูลเหล่านี้สามารถเปรียบเทียบได้กับตัวพารามิเตอร์ของการเรียกใช้โพรซีเยอร์หนึ่ง ข้อมูลที่ส่งตอบสนองกลับมาก็มี response header เป็นของตัวเองก็ได้ ข้อมูลส่วนหัว (header) บางส่วนสามารถนำมาใช้ได้ในทั้งสองทิศทาง รูป 7-43 แสดงตัวอย่าง header ที่มีใช้งานทั่วไป

ข้อมูลส่วนหัว User-Agent ช่วยผู้ใช้ในการแจ้งให้เซิบเวอร์ได้ทราบข้อมูลเกี่ยวกับบราวเซอร์ ระบบปฏิบัติการ และคุณสมบัติอื่น ๆ ในรูป 7-34 เซิบเวอร์ได้รับทราบข้อมูลนี้และสามารถนำมาสร้าง

Header	Type	Contents
User-Agent	Request	Information about the browser and its platform
Accept	Request	The type of pages the client can handle
Accept-Charset	Request	The character sets that are acceptable to the client
Accept-Encoding	Request	The page encodings the client can handle
Accept-Language	Request	The natural languages the client can handle
Host	Request	The server's DNS name
Authorization	Request	A list of the client's credentials
Cookie	Request	Sends a previously set cookie back to the server
Date	Both	Date and time the message was sent
Upgrade	Both	The protocol the sender wants to switch to
Server	Response	Information about the server
Content-Encoding	Response	How the content is encoded (e.g., gzip)
Content-Language	Response	The natural language used in the page
Content-Length	Response	The page's length in bytes
Content-Type	Response	The page's MIME type
Last-Modified	Response	Time and date the page was last changed
Location	Response	A command to the client to send its request elsewhere
Accept-Ranges	Response	The server will accept byte range requests
Set-Cookie	Response	The server wants the client to save a cookie

รูป 7-43
Some HTTP message headers

เป็นข้อมูลบนเว็บเพจได้ตามต้องการผ่าน PHP script ข้อมูลส่วนหัวนี้จึงมีไว้ให้ผู้ใช้ ใช้ในการแจ้งข่าวสารให้เซิบเวอร์ได้ทราบ

ข้อมูลส่วนหัว Accept ทั้ง 4 ชนิด บอกเซิบเวอร์ว่าข้อมูลชนิดใดบ้างที่มีขีดจำกัดในการรับไว้ใช้งานได้ ชนิดแรกกำหนดชนิดของ MIME ที่ยอมรับได้ (เช่น text/html) ตัวที่สองเป็นชนิดของชุดตัวอักษร (เช่น ISO-8859-5 หรือ Unicode-1-1) ตัวที่สามเกี่ยวข้องกับวิธีการบีบอัดข้อมูล (เช่น gzip) ตัวที่สี่กำหนดการใช้ภาษา natural language (เช่น Spanish) ในกรณีที่เซิบเวอร์มีเพจให้เลือกหลายแบบ เซิบเวอร์ก็จะใช้ข้อมูลนี้ในการเลือกเพจที่เหมาะสมส่งให้กับผู้ใช้ แต่ถ้าไม่มีเพจที่เหมาะสมก็จะตอบสนองด้วยข้อความที่บอกถึงความผิดพลาดที่เกิดขึ้นและกำหนดให้คำร้องขอนั้นล้มเหลว

ข้อมูลส่วนหัว Host ใช้ระบุชื่อของเซิบเวอร์ซึ่งนำมาจาก URL ข้อมูลนี้เป็นส่วนที่บังคับให้ใช้ เนื่องจากหมายเลข IP บางตัวนั้นถูกนำมาใช้กับชื่อ DNS หลายชื่อ เซิบเวอร์จึงต้องการทราบว่าจะส่งข้อมูลไปให้โฮสตัวใด

ข้อมูลส่วนหัว Authorization ถูกนำมาใช้กับเพจที่ได้รับการป้องกัน ในกรณีนี้ ผู้ใช้มีความจำเป็นจะต้องพิสูจน์ว่าตนเองมีสิทธิที่จะดูข้อมูลในเพจนั้นๆ

ข้อมูลส่วนหัว cookie ถูกนำไปใช้โดยผู้ใช้เพื่อนำกลับมาส่งคืนให้กับเซิบเวอร์ในภายหลัง

ข้อมูลส่วนหัว Date สามารถใช้งานได้ทั้งสองทิศทางประกอบด้วยเวลาและวันที่ที่ข้อความถูกส่งออกมา

ข้อมูลส่วนหัว Upgrade ใช้ในการเปลี่ยนรุ่นของโพรโตคอล HTTP ในอนาคต ช่วยให้ผู้ใช้สามารถประกาศให้เซิบเวอร์ทราบว่าตนเองใช้งานรุ่นอะไรอยู่และเซิบเวอร์สามารถยืนยันรุ่นที่ใช้งานอยู่ได้

ในส่วนต่อไปนี้เป็นข้อมูลส่วนหัวที่ถูกนำมาใช้โดยเซิบเวอร์เท่านั้นเพื่อใช้ในการตอบสนองต่อคำร้องขอข้อมูลต่าง ๆ อันแรกคือ Server ใช้ในการบอกข้อมูลเกี่ยวกับตนเอง

ข้อมูลส่วนหัว 4 อันต่อไปซึ่งเริ่มต้นด้วย Content- ช่วยให้เซิบเวอร์อธิบายเกี่ยวกับคุณสมบัติของเพจที่ส่งไป

ข้อมูลส่วนหัว Last-Modified บอกให้ทราบว่าเพจนั้นถูกแก้ไขครั้งสุดท้ายเมื่อใดซึ่งเป็นส่วนที่สำคัญมากในการทำ page caching

ข้อมูลส่วนหัว Location มีไว้ให้เซิบเวอร์แจ้งแก่ผู้ใช้ว่าสมควรที่จะลองใช้ URL ที่อื่น สามารถนำมาใช้ได้ในกรณีที่เพจนั้นได้ถูกย้ายไปที่อื่นแล้วหรือมีหลาย URL ที่มีข้อมูลอย่างเดียวกัน (ในเซิบเวอร์ตัวอื่น) และใช้สำหรับองค์กรที่มีโฮมเพจอยู่ใน ".com" โดเมนแต่ต้องการเปลี่ยนให้ผู้ใช้หันไปใช้ URL ที่อื่นซึ่งอาจจะดีกว่าเมื่อพิจารณาจากหมายเลข IP ของผู้ใช้ เช่น เปลี่ยนไปใช้เว็บเพจที่นำเสนอเป็นภาษาเดียวกันกับผู้ใช้ เป็นต้น

ถ้าเว็บเพจนั้นมีขนาดใหญ่มาก ผู้ใช้ที่มีเครื่องคอมพิวเตอร์ขีดความสามารถต่ำอาจไม่ต้องการดูข้อมูลทั้งเพจนั้นในเวลาเดียวกัน เซิบเวอร์บางตัวจึงยอมรับคำร้องขอข้อมูลที่มีการกำหนดขนาด (byte range) ของเว็บเพจที่ต้องการเอาไว้ด้วยทำให้เพจนั้นสามารถถูกดึงขึ้นมาคราวละน้อย ๆ ได้ ข้อมูลส่วนหัว Accept-Range ใช้ในการประกาศว่าเซิบเวอร์จะยอมรับการขอข้อมูลทีละส่วนนี้

ข้อมูลส่วนหัวเกี่ยวกับ cookie ตัวที่สองคือ Set-Cookie เป็นวิธีที่เซิบเวอร์ใช้ในการส่ง cookie มายังเครื่องผู้ใช้ ผู้ใช้นั้นควรที่จะรับ cookie ที่ส่งไปนี้ และส่งกลับมาเมื่อมีการติดต่อในครั้งต่อไป

ตัวอย่างการใช้ HTTP

เนื่องจาก HTTP เป็นโพรโตคอลที่ทำงานโดยใช้ข้อมูลในรูปแบบ ASCII จึงเป็นการง่ายที่ผู้ใช้จะสามารถติดต่อกับเว็บเซิบเวอร์จากเครื่องเทอร์มินอลได้ สิ่งที่ต้องการในที่นี้ก็คือการเชื่อมต่อ TCP ไปยังพอร์ต 80 ของเครื่องเซิบเวอร์เท่านั้น คำสั่งต่อไปนี้สามารถนำไปทดลองใช้งานได้ (ควรจะเป็นเครื่อง Unix)

```
telnet www.ietf.org 80 > log
GET /rfc.html HTTP/1.1
Host: www.ietf.org
(บรรทัดว่างที่ต้องพิมพ์ใส่เข้าไปด้วย)
close
```

คำสั่งชุดนี้เริ่มต้นด้วยการใช้ telnet ในการเชื่อมต่อไปยังพอร์ต 80 ของเครื่องเซิบเวอร์ขององค์กร IETF คือ www.ietf.org ผลลัพธ์ที่ได้จากการโต้ตอบครั้งนี้จะถูกบันทึกไว้ในไฟล์ "log" ต่อไปเป็นคำสั่ง

GET ซึ่งได้บอกชื่อไฟล์และโพรโตคอลที่ใช้ บรรทัดต่อไปคือส่วนที่เป็น header ที่จำเป็นจะต้องมีบรรทัดว่างหนึ่งบรรทัดนั้นก็จำเป็นจะต้องมีเช่นกัน ซึ่งเป็นการบอกให้เซิบเวอร์ทราบว่าไม่มีส่วนของ header ตามมาอีกแล้ว คำสั่ง close เป็นการยกเลิกการเชื่อมต่อครั้งนี้

ไฟล์ log สามารถนำมาดูได้โดยใช้ text editor ใดๆ ก็ได้ สิ่งที่อยู่ในไฟล์นี้ควรมีลักษณะคล้ายกับที่แสดงในรูป 7-44 นอกจากว่าองค์กร IETF จะได้เปลี่ยนแปลงโฮมเพจไปจากเดิมแล้ว สิ่งที่ปรากฏในสามบรรทัดแรกเป็นผลมาจากโปรแกรม telnet ของผู้ใช้เองไม่ได้มาจากเซิบเวอร์ นับตั้งแต่บรรทัดที่มีข้อความ "HTTP/1.1 200 OK" นั้นเป็นข้อมูลที่ตอบสนองมาจากเซิบเวอร์ของ IETF ซึ่งยืนยันว่ายินดีที่จะสื่อสารกับผู้ใช้ จากนั้นก็เป็นส่วน header และตามด้วยสิ่งที่อยู่ภายในโฮมเพจ

7.3.5 การขยายประสิทธิภาพ

เนื่องจากความนิยมในการใช้เว็บได้เพิ่มสูงมากขึ้นเรื่อยๆ เซิบเวอร์ เราเตอร์ และสายสื่อสารมักจะมีปริมาณงานเกินขีดความสามารถที่จะรองรับได้อยู่บ่อยๆ หลายคนเริ่มตั้งฉายา WWW ว่าเป็น World Wide Wait ผลจากการที่มีระยะเวลารอนานมากนี้ทำให้นักวิจัยได้สร้างเทคนิคใหม่ๆ ขึ้นมาเพิ่มเติมประสิทธิภาพของทั้งระบบอันได้แก่ caching, server replication และ content delivery network

Caching

วิธีการง่ายๆในการเพิ่มประสิทธิภาพก็คือการบันทึกเพจที่ได้ร้องขอไปแล้วเอาไว้เผื่อในกรณีที่เพจนั้น

```
Trying 4.17.168.6...
Connected to www.ietf.org.
Escape character is '^]'.
HTTP/1.1 200 OK
Date: Wed, 08 May 2002 22:54:22 GMT
Server: Apache/1.3.20 (Unix) mod_ssl/2.8.4 OpenSSL/0.9.5a
Last-Modified: Mon, 11 Sep 2000 13:56:29 GMT
ETag: "2a79d-c8b-39bce48d"
Accept-Ranges: bytes
Content-Length: 3211
Content-Type: text/html
X-Pad: avoid browser bug

<html>
<head>
<title>IETF RFC Page</title>

<script language="javascript">
function url() {
 var x = document.form1.number.value
 if (x.length == 1) {x = "000" + x }
 if (x.length == 2) {x = "00" + x }
 if (x.length == 3) {x = "0" + x }
 document.form1.action = "/rfc/rfc" + x + ".txt"
 document.form1.submit
}
</script>

</head>
```

รูป 7-44 ส่วนเริ่มต้นของ output ที่ได้รับจาก www.ietf.org/rfc.html

จะต้องถูกนำมาใช้งานอีก เทคนิคนี้จะมีประสิทธิผลมากถ้าเป็นเพจที่เป็นที่นิยมและได้รับการร้องขอไปใช้งานเป็นประจำ เช่น www.yahoo.com และ www.cnn.com การนำเพจที่เคยเข้าชมแล้วนำกลับมาใช้งานใหม่เรียกว่า caching ขั้นตอนการทำงานโดยทั่วไปจะมีโพรเซสหนึ่งเรียกว่า proxy ทำหน้าที่ในการดูแล cache เมื่อต้องการใช้ cache ผู้ใช้สามารถกำหนดติดตั้งให้บราวเซอร์ทุกเพจที่มีการเรียกใช้นั้นจะต้องผ่าน proxy แทนที่จะไปยังเซิบเวอร์ของเพจนั้นๆ โดยตรง ถ้า proxy มีเพจนี้อยู่แล้วมันก็จะส่งเพจนี้มาให้ในทันที แต่ถ้าไม่มีอยู่มันก็จะไปดึงเพจนั้นมาจากเซิบเวอร์ จัดการบันทึกไว้ใน cache เพื่อการใช้งานในอนาคต และจัดการส่งเพจนั้นมายังผู้ใช้

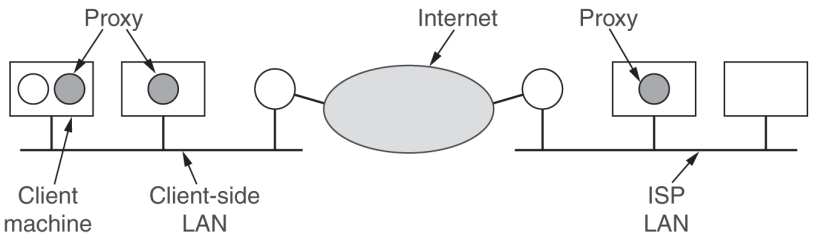
คำถามที่น่าสนใจสองข้อเกี่ยวกับ caching คือ

1. ใครควรจะเป็นผู้ทำ caching

2. แต่ละเพจจะถูกเก็บไว้ใน cache นานแค่ไหน

สำหรับคำถามแรกนั้นมีได้หลายคำตอบ เครื่องพีซีแต่ละเครื่องมักจะมี proxy เป็นของตนเองเพื่อให้สามารถค้นหาเพจที่เคยเข้าชมแล้วได้อย่างรวดเร็ว ในระบบเครือข่าย LAN ขององค์กร proxy มักจะทำงานอยู่บนเครื่องคอมพิวเตอร์แยกต่างหากซึ่งจะถูกใช้งานร่วมกับพีซีตัวอื่นๆ ทั้งหมดที่อยู่ในวง LAN นั้น เผื่อว่าถ้าผู้ใช้คนหนึ่งเข้าไปดูข้อมูลในเพจเพจหนึ่งแล้วมีผู้ใช้อีกคนหนึ่งต้องการเข้าไปดูข้อมูลในเพจนั้นด้วย คนที่ดูทีหลังก็จะได้รับข้อมูลในเพจนั้นอย่างรวดเร็วจากข้อมูลที่เก็บอยู่ใน cache ของเครื่อง proxy เซิบเวอร์ของ ISP (ผู้ให้บริการการเชื่อมต่อเข้ากับระบบอินเตอร์เน็ต) หลายแห่งก็มี proxy เป็นของตนเองทั้งนี้จะได้ช่วยให้การบริการแก่ลูกค้าของตนเองนั้นรวดเร็วยิ่งขึ้น โดยทั่วไป proxy เหล่านี้ทำงานร่วมกันอยู่ ดังนั้นคำร้องของข้อมูลจะถูกส่งไปยัง proxy ที่ใกล้ตัวที่สุดก่อนเสมอ ถ้าไม่สำเร็จก็จะส่งคำร้องต่อไปยัง proxy ของหน่วยงาน ลำดับต่อไปก็จะเป็น proxy ของ ISP ซึ่งในลำดับหลังสุดนี้จะต้องได้ข้อมูลกลับมาเสมอไม่ว่าจะมาจาก cache ของตนเอง หรือ cache ของ proxy ในระดับบนขึ้นไป หรือมาจากเซิบเวอร์ของเว็บเพจนั้นโดยตรง การค้นหาข้อมูลใน proxy ที่เรียงลำดับขึ้นไปเรื่อยๆ นี้เรียกว่า hierarchical caching รูป 7-45 แสดงความเป็นไปได้ในการสร้าง proxy ขึ้นมาใช้งาน

คำถามที่สองที่ว่าแต่ละเพจจะถูกเก็บไว้ใน cache นานแค่ไหนนั้นตอบได้ยากกว่า เนื่องจากข้อมูลบางเพจนั้นไม่ควรที่จะเก็บไว้ใน cache เลย ตัวอย่างเช่น เพจที่แสดงรายชื่อหุ้นของบริษัทต่างๆ ที่มีการซื้อขายมากที่สุดใน 50 ลำดับแรกนั้น ข้อมูลจะเปลี่ยนไปทุกวินาทีการจัดเก็บไว้ใน cache จึงเท่ากับเป็นการส่งข้อมูลเก่า (stale) ให้แก่ผู้ใช้ ในทางกลับกันเมื่อตลาดหุ้นได้ปิดทำการในวันนั้นหรือเป็นวันหยุด ข้อมูลในเพจนี้ก็จะยังคงเหมือนเดิมตลอดเวลาจนกว่าตลาดหุ้นจะเปิดทำการใหม่ ดังนั้น



รูป 7-45 Hierarchical caching ที่มี proxy อยู่ 3 ระดับ

ความสามารถในการ cache ข้อมูลจึงเปลี่ยนไปอยู่เสมอ

ประเด็นหลักในการกำหนดว่าเมื่อไหร่ที่เอาเว็บเพจออกไปจาก cache นั้นอยู่ที่ผู้ใช้ยินดีที่จะใช้ข้อมูลเก่าได้นานเพียงใด (เนื่องจากเพจที่เก็บอยู่ใน cache นั้นจะถูกเก็บไว้ในดิสก์ ดังนั้นปริมาณข้อมูลใน cache จึงไม่ใช่ประเด็นที่จะต้องนำมาพิจารณา) ถ้า proxy จัดการเอาเพจออกจาก cache อย่างรวดเร็ว proxy ก็จะไม่นำข้อมูลเก่าไปให้แก่ผู้ใช้ แต่ประสิทธิภาพในการทำงานของ proxy ก็จะต่ำ เพราะส่วนใหญ่จะหาเพจเก่าไม่พบ ถ้า proxy เก็บเพจไว้นานเกินไป ก็อาจจะมีประสิทธิภาพสูงขึ้นแต่ก็แลกมาด้วยการส่งข้อมูลเก่าให้แก่ผู้ใช้

มีวิธีการแก้ปัญหานี้อยู่สองแนวทาง แนวทางแรกให้ใช้ heuristic (ข้อมูลที่ได้มาจากการสังเกตพฤติกรรมในอดีต) ในการเดาว่าจะเก็บแต่ละเพจไว้นานเพียงใด วิธีการที่ใช้กันโดยทั่วไปคือการกำหนดระยะเวลาในการเก็บเพจเอาไว้โดยอาศัยข้อมูลจากเขตข้อมูล Last-Modified header ดังที่แสดงในรูป 7-43 ถ้าเพจนั้นถูกแก้ไขเมื่อชั่วโมงที่แล้ว เพจนั้นก็จะถูกเก็บไว้ใน cache เป็นเวลาหนึ่งชั่วโมง ถ้าเพจนั้นถูกแก้ไขเมื่อปีที่แล้ว แสดงว่าเป็นเพจที่มีข้อมูลที่ไม่ต้องการแก้ไขมากนักจึงสามารถเก็บเพจนั้นไว้ได้อีกปีหนึ่ง โดยหวังว่าเพจนั้นจะไม่ถูกแก้ไขในปีต่อมา วิธีการใช้ heuristic นี้มักจะทำงานได้ผลดีแต่ก็จะส่งข้อมูลเก่ามาบ้างเป็นครั้งคราว

อีกแนวทางหนึ่งนั้นมีค่าใช้จ่ายสูงกว่าแต่สามารถกำจัดโอกาสที่จะส่งข้อมูลเก่าไปให้ผู้ใช้ได้ โดยการใช้คุณสมบัติพิเศษของมาตรฐาน RFC 2616 ซึ่งอธิบายเกี่ยวกับวิธีการบริหารจัดการ cache คุณสมบัติที่เป็นประโยชน์มากที่สุดคือ header ส่วนที่เป็น If-Modified-Since ซึ่ง proxy สามารถส่งคำร้องนี้ไปยังเซิบเวอร์ได้ header นั้นจะบอกข้อมูลเกี่ยวกับเพจที่ proxy ต้องการและเวลาที่เพจนั้นถูกแก้ไขครั้งหลังสุด (ข้อมูลจาก Last-Modified header) ถ้าเพจดังกล่าวไม่ได้ถูกแก้ไขเลยนับจากเวลานั้น เซิบเวอร์ก็เพียงส่งข้อความสั้นๆ ตอบกลับมาคือ Not Modified (โค้ดบอกสถานะรหัส 304 ดังที่แสดงในรูป 7-42) ซึ่งจะบอกให้ proxy จัดการส่งเพจใน cache กลับไปให้ผู้ใช้ แต่ถ้าเพจนั้นถูกแก้ไขหลังจากเวลาที่ระบุนั้น เซิบเวอร์ก็จะจัดการส่งเพจใหม่กลับมาให้ แม้ว่าวิธีการนี้ต้องอาศัยการส่งคำร้องไปยังเซิบเวอร์และรอคำตอบกลับมา แต่คำตอบนั้นจะเป็นข้อความที่สั้นมากเมื่อเพจใน cache ยังสามารถนำไปใช้งานได้

แนวทางทั้งสองนี้อาจนำมารวมกันได้คือ ในช่วงเวลา ΔT แรกภายหลังจากที่ได้ดึงเพจนั้นมาใช้ proxy จะส่งเพจนั้นกลับคืนไปที่ผู้ใช้ในทันที ภายหลังจากที่เพจนั้นเก็บอยู่ใน cache เป็นระยะเวลาหนึ่งแล้ว proxy จึงจะส่งข้อความ If-Modified-Since ไปยังเซิบเวอร์ การเลือกค่า ΔT นั้นเกี่ยวข้องกับการใช้ heuristic บางอย่างขึ้นอยู่กับว่าเพจนั้นถูกแก้ไขมานานเท่าใดแล้ว

เว็บเพจที่มีข้อมูลที่เป็นพลวัต (dynamic content) (เช่น การใช้ PHP script) ไม่ควรที่จะถูกเก็บไว้ใน cache เลย เพื่อจัดการกับปัญหานี้จึงมีการคิดค้นกลไกทั่วไปสำหรับเซิบเวอร์เพื่อที่จะสามารถบอกให้ proxy ทุกตัวที่อยู่ในเส้นทางการส่งข้อมูลกลับมายังผู้ใช้ได้ทราบว่าไม่ให้ใช้เพจปัจจุบันซ้ำยกเว้นแต่จะได้รับการตรวจสอบแล้วเท่านั้น กลไกนี้สามารถนำมาใช้กับเว็บเพจใดๆ ที่มีการแก้ไขข้อมูลอยู่เสมอได้

อีกแนวทางหนึ่งในการปรับปรุงประสิทธิภาพเรียกว่า proactive caching เมื่อ proxy ดึงข้อมูล

มาจากเซิบเวอร์ proxy จะทำการตรวจสอบดูว่ามี hyperlink อยู่ภายในเพจนั้นหรือไม่ ถ้ามีอยู่ proxy ก็สามารถส่งคำร้องของข้อมูลไปยังเซิบเวอร์ที่เกี่ยวข้องได้เพื่อเป็นการดึงข้อมูลเข้ามาไว้ใน cache ล่วงหน้า เผื่อว่าผู้ใช้ต้องการดูข้อมูลในเพจนั้นก็จะสามารถส่งเพจใน cache ไปให้ได้ทันที วิธีการนี้อาจจะสามารถลดเวลาในการดึงข้อมูลเพจต่อ ๆ ไปลงได้แต่ก็อาจจะทำให้สายสื่อสารเต็มไปด้วยข้อมูลเพจต่าง ๆ ที่จัดส่งมาซึ่งอาจไม่ถูกเรียกใช้เลยก็ได้

Server Replication

Caching เป็นเทคนิคที่นำมาใช้ทางฝั่งผู้ใช้ในการปรับปรุงประสิทธิภาพ เทคนิคที่นำมาใช้ทางเซิบเวอร์ก็มีอยู่เช่นกัน วิธีการที่นิยมนำมาใช้มากที่สุดคือ การสร้างเซิบเวอร์เงาขึ้นหลายแห่ง กระจายกันอยู่ในที่ต่าง ๆ เทคนิคนี้เรียกว่า "Mirroring"

การใช้ mirroring โดยทั่วไปเป็นดังนี้ โฮมเพจขององค์กรจะประกอบด้วยข้อมูลน้อยมากและจะมีการเชื่อมต่อไปยัง mirror sites เช่น การเชื่อมต่อไปยังเซิบเวอร์ทางฝั่งตะวันออก ทางตะวันตก ทางเหนือ และทางใต้ ผู้ใช้ก็จะเลือก mirror site ที่อยู่ใกล้ตนเองมากที่สุด จากนั้นการโต้ตอบทั้งหมดจะเกิดขึ้นกับ mirror site ที่เลือกนั้น ไม่เกี่ยวกับโฮมเพจอีกต่อไป

Mirror sites มักจะเก็บข้อมูลประเภท static คือไม่มีการเปลี่ยนแปลงเลย องค์กรจะเป็นผู้เลือกว่าจะวาง mirror site ไว้ในที่ใด จัดการวางเซิบเวอร์ไว้ในแต่ละพื้นที่ และใส่เนื้อหาของเว็บไซต์ทั้งหมดไว้ที่ mirror site การวางเซิบเวอร์ไว้แต่ละที่นั้นมักจะถูกใช้งานเป็นระยะเวลานาน

อย่างไรก็ตาม บนเว็บมีปรากฏการณ์พิเศษเป็นของตนเองที่เรียกว่า flash crowds นั่นคือ เว็บไซต์ที่แต่เดิมมานั้นไม่มีคนรู้จัก ไม่มีใครเข้าไปขอข้อมูล แต่ในทันทีทันใดก็กลายเป็นศูนย์รวมข้อมูลที่ทุกคนเข้ามาใช้บริการ ปัญหาที่เกิดขึ้นก็คือเครื่องเซิบเวอร์นั้นไม่ได้ถูกเตรียมไว้สำหรับการให้บริการแก่ผู้ใช้จำนวนมาก ดังนั้นเซิบเวอร์ (แต่เดิมมาเคยให้บริการอย่างดี) จึงไม่สามารถให้บริการได้อีกต่อไป (overloaded)

สิ่งที่ต้องการก็คือ วิธีที่จะให้เว็บไซต์สามารถสังเกตเห็นปริมาณงานที่กำลังจะเพิ่มมากขึ้น เพื่อให้สามารถจัดการสร้างเว็บไซต์สำรองขึ้นมาเท่าที่จำเป็นและใช้เว็บไซต์เหล่านั้นช่วยทำงานจนกว่ากระแสการใช้งานอย่างมากมายนี้จะหมดไป ซึ่งก็จะสามารถปิดเซิบเวอร์สำรองบางส่วนหรือทั้งหมดลงได้ เพื่อให้มีขีดความสามารถนี้เว็บไซต์จำเป็นจะต้องได้รับการเตรียมการล่วงหน้า

Content Delivery Networks

ความฉลาดหลักแหลมในการลงทุนอย่างหนึ่งคือ การที่พยายามคิดว่าจะหาผลประโยชน์จาก World Wide Wait ได้อย่างไร ความคิดหนึ่งเป็นดังนี้ บริษัทแห่งหนึ่งชื่อ CDN (Content Delivery Networks) ทำการตกลงกับผู้ผลิตข้อมูล (เช่น เว็บไซต์เพลง หนังสือพิมพ์ และเว็บไซต์อื่น ๆ ที่ต้องการให้ผลิตภัณฑ์ของตนเองสามารถส่งถึงผู้ใช้ได้อย่างสะดวกและรวดเร็ว) และเสนอที่จะส่งผลิตภัณฑ์นั้น ๆ ไปยังผู้ใช้ได้อย่างมีประสิทธิภาพแลกกับค่าใช้จ่ายที่ผู้ผลิตต้องจ่ายให้ ภายหลังจากที่ตกลงกันได้แล้วผู้ผลิตก็จะส่งมอบผลิตภัณฑ์ของตนให้กับ CDN เพื่อการเตรียมตัวในการนำส่ง (ขั้นตอนนี้จะกล่าวถึงในภายหลัง)

จากนั้น CDN จะทำข้อตกลงกับ ISP (จำนวนมากที่สุดเท่าที่จะเป็นไปได้) โดยเสนอที่จะให้ค่า

ตอบแทนแก่ ISP เพื่อแลกกับการอนุญาตให้จัดตั้งเครื่องเซิบเวอร์ที่บรรจุข้อมูลอันเป็นผลิตภัณฑ์ของบริษัทต่าง ๆ มาเก็บไว้ในระบบเครือข่าย LAN ของ ISP เอง สำหรับ ISP แล้ว นี่ไม่เพียงเป็นแหล่งรายได้อย่างหนึ่งของตนเองแต่ยังเป็นการเพิ่มประสิทธิภาพในการตอบสนองต่อลูกค้าเพื่อให้สามารถได้รับข้อมูลจากเซิบเวอร์ของ CDN ที่บรรจุข้อมูลไว้มากมายได้อย่างรวดเร็ว ผลประโยชน์อีกประการหนึ่งก็คือจะเป็นการเพิ่มความสามารถในการแข่งขันกับ ISP อื่นในแง่ของการให้บริการที่ดีกว่าได้ด้วย ภายใต้เงื่อนไขนี้ จึงทำให้ ISP จำนวนมากทำความตกลงให้บริการร่วมกับ CDN ในปัจจุบันมีเซิบเวอร์มากกว่า 10,000 แห่งทั่วโลกที่ทำความตกลงในลักษณะนี้กับ CDN

ด้วยการสร้างสำเนาข้อมูลไปเก็บไว้ตามเซิบเวอร์นับพันแห่งทั่วโลก จึงทำให้มีความเป็นไปได้อย่างมากที่จะเพิ่มประสิทธิภาพการให้บริการ อย่างไรก็ตาม เพื่อให้ระบบนี้ทำงานได้จริงจำเป็นจะต้องมีวิธีการที่จะเปลี่ยนตำแหน่งการร้องขอข้อมูลของลูกค้าให้หันไปใช้ข้อมูลใน CDN เซิบเวอร์ที่อยู่ใกล้ลูกค้ามากที่สุด และถ้าเป็นไปได้ก็ควรจะเป็นที่ ISP ของลูกค้าเอง (ซึ่งทำความตกลงกับ CDN ไว้เรียบร้อยแล้ว) และการเปลี่ยนแปลงที่อยู่เป้าหมายนี้จะต้องไม่ไปแก้ไข DNS หรือส่วนหนึ่งส่วนใดของโครงสร้างมาตรฐานของระบบอินเตอร์เน็ต

กระบวนการทั้งหมดเริ่มต้นเมื่อผู้ผลิตสินค้าซอฟต์แวร์จัดส่งเว็บไซต์ของตนเองให้แก่ CDN บริษัท CDN จะทำการประมวลผลเว็บเพจแต่ละหน้าผ่าน preprocessor ซึ่งจะแก้ไข URL ทั้งหมดให้กลายเป็น URL ที่จัดเตรียมไว้แล้ว กลยุทธ์ที่นำมาใช้ในที่นี้ก็คือเว็บไซต์ของผู้ผลิตซอฟต์แวร์จะประกอบด้วยเพจจำนวนมากที่มีขนาดเล็ก (เช่นเพจที่มีเพียง HTML text) แต่เพจเหล่านี้มักจะเชื่อมโยงเข้ากับไฟล์ขนาดใหญ่ เช่น ไฟล์รูปภาพ ไฟล์เสียง และไฟล์วีดีโอ เพจ HTML ที่ถูกแก้ไขแล้วจะถูกเก็บไว้ในเซิบเวอร์ของผู้ผลิตซอฟต์แวร์ตามปกติและถูกเรียกใช้ตามปกติ จะมีเพียงไฟล์รูปภาพ ไฟล์เสียง และไฟล์วีดีโอเท่านั้นที่จะถูกอ้างอิงจาก เซิบเวอร์ของ CDN

เพื่อให้เข้าใจการทำงานของกลยุทธ์นี้ ขอให้พิจารณากรณีเว็บเพจของร้าน Furry Videoในรูป 7-46(a) ภายหลังจากที่ถูกแก้ไขข้อมูลแล้ว (ในขั้นตอนการทำ preprocessing) เว็บเพจดังกล่าวจะกลายเป็นเว็บเพจที่แสดงในรูป 7-46(b) และถูกนำไปเก็บไว้ในเซิบเวอร์ของร้าน Furry Video เช่น "www.furryvideo.com/index.html"

เมื่อผู้ใช้ (ใครก็ตาม) ติดต่อไปที่ URL "www.furryvideo.com" DNS จะส่งหมายเลข IP ของเว็บไซต์ของ Furry Video กลับคืนไปที่ผู้ใช้คนนั้น ซึ่งจะช่วยให้บราวเซอร์สามารถแสดงภาพโฮมเพจของร้านนี้ได้ตามปกติ แต่เมื่อ hyperlink ใดก็ตามที่ถูกเลือก (ใช้เมาส์คลิ๊ก) บราวเซอร์จะทำการค้นหาไฟล์นั้นจาก "cdn-server.com" จากนั้นบราวเซอร์จะส่งคำร้องของข้อมูลไปยังหมายเลข IP ของเว็บไซต์นี้ ซึ่งคาดหวังว่าคงจะได้รับไฟล์ข้อมูลกลับมา (ตามที่ควรจะเป็น)

แต่ในกรณีนี้ cdn-server.com ไม่มีข้อมูลอะไรเกี่ยวกับไฟล์นั้นอยู่เลย เพราะทำหน้าที่เป็นเพียงเซิบเวอร์ปลอมเท่านั้น เซิบเวอร์นี้จะตรวจสอบชื่อไฟล์ที่ผู้ใช้ต้องการและชื่อของเซิบเวอร์เพื่อให้ทราบว่าข้อมูลเพจใดของผู้ผลิตซอฟต์แวร์รายใดที่ผู้ใช้ต้องการ และจะตรวจสอบหมายเลข IP ของผู้ใช้เพื่อนำมาใช้ในการค้นหาข้อมูลในฐานข้อมูลซึ่งจะบอกให้ทราบว่าผู้ใช้คนนั้นอยู่ที่ใด เมื่อได้ข้อมูลนี้แล้ว เซิบเวอร์นี้จะกำหนดว่า CDN เซิบเวอร์ตัวใดที่จะสามารถให้ข้อมูลแก่ผู้ใช้คนนี้ได้ดีที่สุด การตัดสินใจนี้ค่อยข้างยากลำบากเพราะว่าการที่อยู่ในระยะทางที่ใกล้ที่สุดบนพื้นดินนั้นไม่จำเป็นว่าจะต้องเป็น

รูป 7-46
(a) เว็บเพจปกติ
(b) เว็บเพจที่ผ่านกระบวนการ preprocessor แล้ว

```
<html>
<head> <title> Furry Video </title> </head>
<body>
<h1> Furry Video's Product List </h1>
<p> Click below for free samples. </p>
<a href="bears.mpg"> Bears Today </a> <br>
<a href="bunnies.mpg"> Funny Bunnies </a> <br>
<a href="mice.mpg"> Nice Mice </a> <br>
</body>
</html>
```

(a)

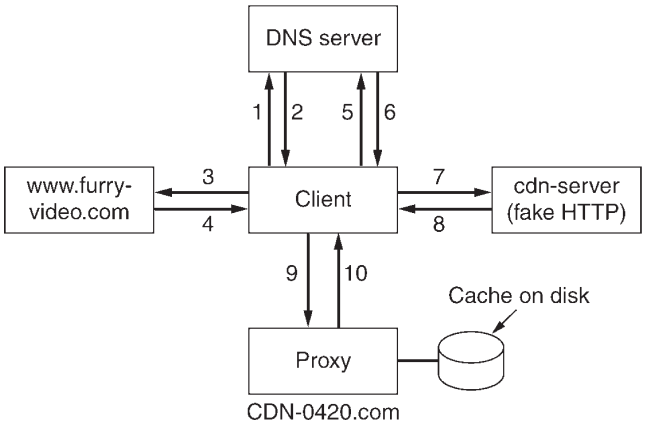
```
<html>
<head> <title> Furry Video </title> </head>
<body>
<h1> Furry Video's Product List </h1>
<p> Click below for free samples. </p>
<a href="http://cdn-server.com/furryvideo/bears.mpg"> Bears Today </a> <br>
<a href="http://cdn-server.com/furryvideo/bunnies.mpg"> Funny Bunnies </a> <br>
<a href="http://cdn-server.com/furryvideo/mice.mpg"> Nice Mice </a> <br>
</body>
</html>
```

(b)

ระยะทางที่ใกล้ที่สุดบนระบบเครือข่ายเสมอไป และเซิบเวอร์ที่อยู่ใกล้ที่สุดบนระบบเครือข่ายอาจจะกำลังมีงานทำยุ่งมากก็ได้ ภายหลังจากที่ได้เลือก CDN เซิบเวอร์แล้ว cdn-server.com จะส่งคำตอบกลับไปยังผู้รับซึ่งกำหนดให้ใช้โค้ดรหัส 301 (บอกให้บราวเซอร์ไปค้นหาข้อมูลจากแหล่งอื่น) และ Location header ที่เป็น URL ของไฟล์ที่อยู่ใน CDN เซิบเวอร์ที่เลือกเอาไว้ สำหรับตัวอย่างนี้ สมมุติว่า URL ที่กล่าวถึงคือ "www.CDN-0420.com/furryvideo/bears.mpg" บราวเซอร์ของผู้ใช้ก็จะไปดึงไฟล์นี้มาแสดงผลตามปกติ

ขั้นตอนที่กล่าวถึงนี้แสดงในรูป 7-47 ขั้นตอนแรกคือการค้นหา www.furryvideo.com เพื่อให้ได้หมายเลข IP มา หลังจากนั้นจะสามารถดึง HTML เพจมาแสดงได้ตามปกติ เว็บเพจนี้มี hyperlink เชื่อมไปยัง cdn-server.com อยู่สามแห่ง (ดูรูป 7-461(b)) สมมุติว่าผู้ใช้เลือก hyperlink อันแรก บราวเซอร์ก็จะไปค้นหาหมายเลข IP จาก DNS (ขั้นตอนที่ 5) และได้รับกลับคืนมาในขั้นตอนที่ 6 ในขั้นตอนที่ 7 บราวเซอร์จะร้องขอไฟล์ bears.mpg จาก cdn-server ซึ่งในขั้นตอนที่ 8 บราวเซอร์ของผู้ใช้จะถูกบอกให้ไปขอข้อมูลจาก CDN-0420.com แทน เมื่อทำตามนั้นในขั้นตอนที่ 9 บราวเซอร์จะได้รับไฟล์จาก cache ของ Proxy ในขั้นตอนที่ 10 คุณสมบัติที่ทำให้กลไกนี้สามารถทำงานได้คือขั้นตอนที่ 8 นั่นคือ เซิบเวอร์ปลอมได้เปลี่ยนทิศทางการขอข้อมูลของบราวเซอร์ผู้ใช้ไปยัง CDN proxy ที่อยู่ใกล้กับผู้ใช้

CDN เซิบเวอร์ที่ผู้ใช้ถูกเปลี่ยนให้ไปขอข้อมูลแทนเซิบเวอร์ปลอมนั้นมักจะเป็น Proxy ที่มี cache ขนาดใหญ่ซึ่งมีข้อมูลพร้อมจะให้บริการ อย่างไรก็ตาม ถ้าผู้ใช้ร้องขอไฟล์ที่ไม่มีอยู่ใน cache ไฟล์นั้นจะถูกดึงมาจากเซิบเวอร์ตัวจริงที่เป็นเจ้าของผลิตภัณฑ์ซอฟต์แวร์นั้นและถูกใส่เข้าไปใน cache เพื่อนำมาใช้งานในครั้งต่อไป การใช้ proxy แทนที่จะเป็นเซิบเวอร์สำรองที่มีข้อมูลอยู่ในตัวเองนั้นจะทำให้ CDN เซิบเวอร์สามารถปรับค่าต่างๆ เช่น ขนาดของดิสก์ เวลาที่ใช้ในการดึงข้อมูล และอื่นๆ ได้ตาม



1. Look up www.furryvideo.com
2. Furry's IP address returned
3. Request HTML page from Furry
4. HTML page returned
5. After click, look up cdn-server.com
6. IP address of cdn-server returned
7. Ask cdn-server for bears.mpg
8. Client told to redirect to CDN-0420.com
9. Request bears.mpg
10. Cached file bears.mpg returned

รูป 7-47 ขั้นตอนต่างๆ ในการค้นหา URL เมื่อมี CDN เข้ามาเกี่ยวข้อง

ความเหมาะสม

7.3.6 The Wireless Web

ในปัจจุบันมีผู้คนจำนวนไม่น้อยที่ให้ความสนใจในอุปกรณ์พกพาติดตัวขนาดเล็กที่มีขีดความสามารถในการเชื่อมต่อเข้ากับเว็บผ่านช่องทางการสื่อสารไร้สาย อันที่จริงก้าวแรกของการพัฒนาไปในทิศทางนี้ได้รับการนำมาใช้งานเรียบร้อยแล้วก้าวต่อไปคงจะต้องเกิดขึ้นอย่างไม่ต้องสงสัยภายในระยะเวลาอันสั้น ในหัวข้อนี้จะได้กล่าวถึงแนวความคิดเกี่ยวกับการใช้เครือข่ายไร้สายมาเชื่อมต่อกับเว็บ

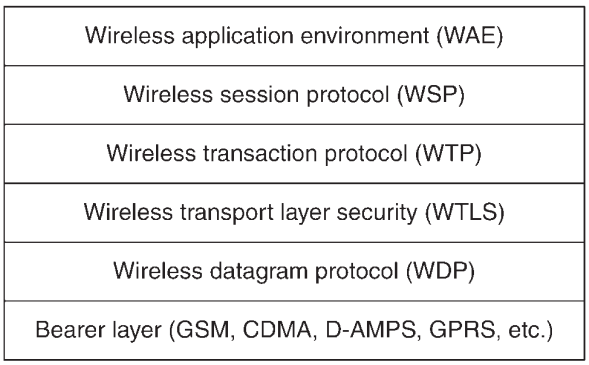
WAP-The Wireless Application Protocol

เมื่อระบบอินเตอร์เน็ตและโทรศัพท์มือถือได้กลายมาเป็นอุปกรณ์ที่มีใช้งานกันทั่วไปแล้ว ก็มีแนวความคิดที่จะนำทั้งสองสิ่งนี้เข้ามารวมกันกลายเป็นอุปกรณ์สื่อสารไร้สายที่มีหน้าจอสำหรับการใช้งานอีเมลล์และเว็บ กลุ่มบุคคลที่เริ่มต้นพัฒนาแนวความคิดนี้นำโดยบริษัทโนเกีย อีริคสัน โมโตโรล่า และ phone.com และในปัจจุบันได้มีสมาชิกเพิ่มขึ้นเป็นจำนวนหลายร้อยราย ระบบนี้ได้รับการตั้งชื่อว่า WAP (Wireless Application Protocol)

อุปกรณ์ WAP อาจจะเป็นโทรศัพท์มือถือที่ได้รับการดัดแปลง พีดีเอ หรือโน้ตบุ๊คคอมพิวเตอร์ที่ไม่มีขีดความสามารถทางด้านเสียงเข้ามาเกี่ยวข้อง แนวความคิดพื้นฐานคือการนำโครงสร้างการสื่อสารไร้สายในระบบดิจิตอลที่มีใช้งานอยู่แล้วมาใช้ให้เกิดประโยชน์ ผู้ใช้จะสามารถโทรศัพท์เข้าไปหา WAP เกตเวย์ผ่านระบบเครือข่ายไร้สายและส่งคำร้องขอเว็บเพจได้ เกตเวย์จะตรวจสอบ cache เพื่อค้นหาเพจที่ต้องการ ถ้ามีอยู่ก็จะจัดส่งมาให้ ถ้าไม่มีอยู่ก็จะติดต่อขอเว็บเพจไปยังเซิบเวอร์ผ่านระบบเครือข่ายที่ใช้สายสื่อสารตามปกติ โดยหลักการแล้ว WAP 1.0 ก็คือระบบ circuit-switched ที่มีอัตราการให้บริการการเชื่อมต่อแพงกว่าปกตินั่นเอง อย่างไรก็ตามผู้ใช้ทั่วไปก็ไม่ต้องการที่จะเชื่อมต่อเข้ากับระบบอินเตอร์เน็ตโดยมีจอภาพขนาดเล็กและจ่ายค่าบริการเป็นรายนาที ดังนั้น WAP จึงเป็นเสมือนเรื่องเหลวไหลและคงจะไม่มีผู้ใดนำมาใช้งาน อย่างไรก็ตาม WAP และคู่แข่งคือ i-mode (จะกล่าวถึงต่อไป) จึงมีแนวโน้มที่จะได้รับการพัฒนาออกมาในแนวทางเดียวกันใช้เทคโนโลยีคล้าย ๆ กัน ในที่นี้จะกล่าวถึงเทคโนโลยีที่ใช้ใน WAP 1.0 พอสังเขปดังนี้

WAP เป็นโพรโตคอลแสต็กสำหรับการเข้าใช้งานเว็บ แต่ได้รับการพัฒนาให้เหมาะกับการเชื่อม

รูป 7-48
โพรโตคอลแสตีก
ของ WAP

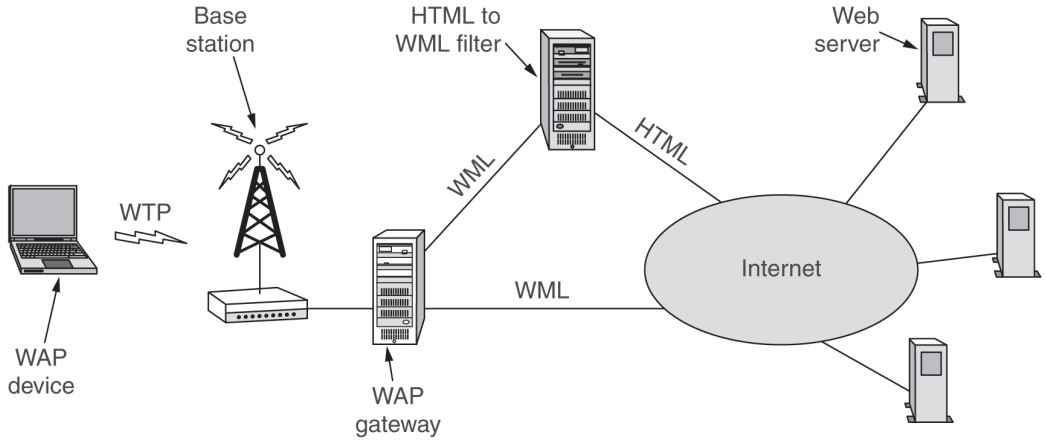


ต่อที่ใช้ช่องสื่อสารแคบโดยใช้อุปกรณ์สื่อสารไร้สายที่มีซีพียูขีดความสามารถต่ำ มีหน่วยความจำน้อย และจอภาพแสดงผลขนาดเล็ก ข้อจำกัดเหล่านี้แตกต่างไปจากข้อจำกัดที่มีในเครื่องพีซีแบบตั้งโต๊ะซึ่งทำให้โพรโตคอลมีลักษณะที่แตกต่างกันออกไป รูป 7-48 แสดงชั้นสื่อสารต่างๆ ที่มีอยู่

ชั้นสื่อสารระดับล่างสุดสนับสนุนการเชื่อมต่อกับระบบโทรศัพท์มือถือทุกระบบที่มีใช้งานอยู่ในปัจจุบันรวมทั้ง GSM, D-AMPS และ CDMA ความเร็วในการสื่อสารของ WAP 1.0 อยู่ที่ 9600 บิตต่อวินาที ชั้นสื่อสารเหนือขึ้นมาเป็นดาต้าแกรมโพรโตคอล WDP (Wireless Datagram Protocol) ซึ่งก็คือ UDP นั่นเอง ต่อไปเป็นชั้นสื่อสารเกี่ยวกับการรักษาความปลอดภัยซึ่งมีความจำเป็นสำหรับการสื่อสารแบบไร้สาย เรียกว่า WTLS (Wireless Transport Layer Security) ซึ่งเป็นส่วนย่อยของโพรโตคอล SSL (กล่าวถึงในบทที่ 8) ชั้นเหนือขึ้นไปคือโพรโตคอล transaction layer ซึ่งทำหน้าที่ในการจัดการคำร้องขอเว็บเพจ และการตอบสนองต่อคำร้องนั้นๆ ซึ่งสามารถทำงานได้ทั้งแบบเชื่อถือได้และแบบเชื่อถือไม่ได้ (reliable and unreliable) ชั้นสื่อสารนี้ทำหน้าที่แทนโพรโตคอล TCP ซึ่งไม่เหมาะกับการสื่อสารไร้สายเนื่องจากเหตุผลทางด้านประสิทธิภาพ ชั้นสื่อสารต่อไปคือ session layer ซึ่งทำหน้าที่คล้ายกับ HTTP/1.1 แต่มีข้อจำกัดมากกว่าและได้รับการพัฒนาให้มีความเหมาะสมกับการสื่อสารไร้สาย ชั้นสื่อสารชั้นบนสุดคือบราวเซอร์ขนาดจิ๋ว (WAE)

นอกเหนือจากค่าใช้จ่ายในการใช้งานแล้วปัญหาอีกประการหนึ่งที่เป็นอุปสรรคต่อการนำระบบ WAP มาใช้งานคือระบบนี้ไม่สนับสนุนการใช้ HTML ในชั้นสื่อสาร WAE (Wireless application environment) นั้นใช้ markup language ที่เรียกว่า WML (Wireless Markup Language) ซึ่งเป็นโปรแกรมประยุกต์ที่เขียนขึ้นมาโดยใช้ภาษา XML ดังนั้นอุปกรณ์ WAP จะสามารถเข้าใช้ข้อมูลจากเว็บเพจที่ได้รับการเปลี่ยนให้กลายเป็นภาษา WML แล้วเท่านั้น อย่างไรก็ตาม เนื่องจากคุณสมบัติข้อนี้ได้กลายเป็นข้อจำกัดคุณค่าของ WAP เป็นอย่างมากจึงได้มีการพัฒนาสถาปัตยกรรมที่สามารถเปลี่ยนข้อมูลในเว็บเพจจาก HTML ให้กลายเป็น WML ได้ในขณะที่ใช้งานดังแสดงในรูป 7-49

เมื่อพิจารณาด้วยความเป็นธรรมแล้ว เรียกได้ว่า WAP นั้นอาจจะเป็นโพรโตคอลที่สร้างขึ้นมาใช้งานก่อนเวลาอันควร เมื่อเริ่มพัฒนาโพรโตคอล WAP นั้น XML ยังไม่เป็นที่รู้จักกันนอกเหนือจากวงการ W3C ดังนั้นสื่อต่างๆ จึงได้ประโคมข่าวออกมาว่า WAP ไม่ได้ใช้ HTML ซึ่งแท้ที่จริงแล้วควรจะโฆษณา



รูป 7-49
สถาปัตยกรรม
ของ WAP

ออกมาว่า WAP ได้ใช้มาตรฐาน HTML อันใหม่มากกว่า

I-Mode

ในขณะที่สมาคมการสื่อสารต่างๆ กำลังวุ่นวายอยู่กับการออกมาตรฐาน HTML รุ่นใหม่อยู่นั้น ผู้หญิงชาวญี่ปุ่นคนหนึ่งชื่อ Mari Matsunaga ได้คิดค้นเว็บสำหรับการสื่อสารไร้สายขึ้นมาเรียกว่า i-mode (Information-mode) และได้เปิดให้บริการในประเทศญี่ปุ่นขึ้นในปี ค.ศ. 1999 ภายใน 3 ปี มีลูกค้าใช้บริการอยู่มากกว่า 35 ล้านรายซึ่งสามารถเข้าใช้บริการเว็บไซต์ที่เป็น i-mode ได้มากกว่า 40,000 แห่ง

ระบบ i-mode มีส่วนประกอบที่สำคัญสามส่วนคือ ระบบการนำส่งข้อมูลแบบใหม่ อุปกรณ์มือถือแบบใหม่ (handset) และภาษาใหม่สำหรับการออกแบบเว็บไซต์ ระบบการนำส่งข้อมูลประกอบด้วยระบบเครือข่ายสองระบบที่ทำงานแยกจากกันคือระบบ circuit-switched mobile phone สำหรับการสื่อสารโทรศัพท์มือถือที่มีใช้งานอยู่แล้ว (ซึ่งเปรียบเทียบได้กับระบบ D-AMPS) และระบบเครือข่าย packet-switched network ที่สร้างขึ้นมาใหม่เพื่อให้บริการ i-mode โดยเฉพาะ การสื่อสารที่เป็นเสียงสนทนาจะใช้ระบบ circuit-switched แบบเก่าที่คิดค่าบริการเป็นรายนาที ส่วน I-mode จะใช้ packet-switched ซึ่งจะมีการเปิดใช้งานอยู่ตลอดเวลา (เหมือนระบบ ADSL หรือ เคเบิ้ลโมเด็ม) จึงไม่มีการคิดค่าบริการในการเชื่อมต่อแต่จะใช้วิธีการคิดค่าบริการจากจำนวนแพ็กเก็ตที่ทำการรับ-ส่งจริง ผู้ใช้จะไม่สามารถใช้งานทั้งสองระบบได้ในเวลาเดียวกัน

อุปกรณ์มือถือแบบใหม่มีลักษณะคล้ายกับโทรศัพท์มือถือทั่วไปโดยมีจอภาพขนาดเล็กติดตั้งเพิ่มเติม บริษัท NTT DoCoMo ซึ่งเป็นบริษัทผู้ให้บริการในประเทศญี่ปุ่นใช้หลักการโฆษณาว่า i-mode เป็นอุปกรณ์ที่ดีกว่าโทรศัพท์มือถือทั่วไปโดยไม่ได้บอกว่าเป็นการสื่อสารไปยังระบบอินเตอร์เน็ตแบบไร้สาย อันที่จริงแล้วลูกค้าส่วนใหญ่ไม่ทราบด้วยซ้ำไปว่าเขากำลังเชื่อมต่ออยู่กับระบบอินเตอร์เน็ต เพื่อให้รูปแบบการใช้ i-mode เป็นไปในลักษณะของการให้บริการอย่างหนึ่ง ชุดโทรศัพท์มือถือนี้จึงไม่อนุญาตให้ผู้ใช้เขียนโปรแกรมใดๆ เพิ่มเติมเข้าไปได้แม้ว่าตัวอุปกรณ์เองจะมีขีดความสามารถเทียบเท่ากับเครื่องพีซีในปี ค.ศ. 1995 และสามารถที่จะใช้ระบบปฏิบัติการ Windows 95 หรือ Unix ได้

เมื่อเปิดเครื่อง i-mode ผู้ใช้จะมองเห็นรายการบริการที่ได้รับการจัดไว้ล่วงหน้าอย่างเป็นทางการ

แล้ว ซึ่งมีทั้งหมดมากกว่า 1,000 อย่างที่ถูกแยกไว้เป็น 20 กลุ่ม บริการแต่ละอย่างซึ่งก็คือ เว็บไซต์ i-mode ขนาดเล็กที่มีเซิบเวอร์เป็นของตนเองบริหารจัดการโดยบริษัทต่างๆ กลุ่มบริการต่างๆ ได้แก่ อีเมลล์ นิวส์ ข่าวอากาศ กีฬา เกมส์ ช๊อปปิ้ง แผนที่ บันเทิง ท่องเที่ยว สูตรอาหาร และธนาคารเป็นต้น บริการเหล่านี้มุ่งเน้นเป้าหมายไปที่วัยรุ่นและกลุ่มคนในช่วงอายุ 20-30 ปี ซึ่งมีแนวโน้มที่จะชอบการใช้บริการทางอิเล็กทรอนิกส์ บริการที่ได้รับความนิยมมากที่สุดคืออีเมลล์ซึ่งยอมให้มีขนาดข้อความได้ไม่เกิน 500 ไบต์ ซึ่งดีกว่าระบบ SMS (Short Message Service) ที่อนุญาตให้ใช้ข้อความที่ยาวไม่เกิน 160 ไบต์ เว็บไซต์ที่ให้บริการ i-mode นั้นมีมากกว่า 40,000 แห่งซึ่งจะสามารถเข้าถึงได้ด้วยการพิมพ์ URL เข้าไปแทนที่จะเลือกจากเมนู

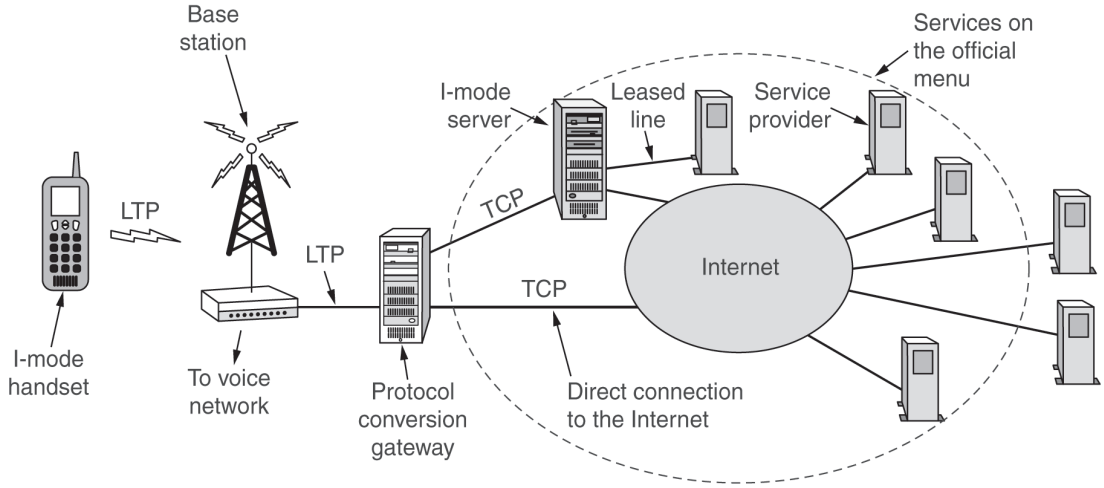
แม้ว่าระบบ i-mode จะประสบความสำเร็จเป็นอย่างมากในประเทศญี่ปุ่นแต่ก็ยังไม่เป็นที่ชัดเจนว่าจะเกิดความสำเร็จในระดับเดียวกันในทวีปยุโรปหรืออเมริกาทั้งนี้เนื่องจากความแตกต่างทางวัฒนธรรมเป็นสำคัญ กล่าวคือ ประการแรก ผู้ใช้ส่วนใหญ่ในโลกตะวันตกจะมีเครื่องพีซีที่มีจอภาพขนาดใหญ่อยู่ที่บ้านของตนเองกันเป็นส่วนมากและสามารถเชื่อมต่อกับระบบอินเตอร์เน็ตได้ด้วยความเร็ว 56 กิโลบิตต่อวินาทีหรือสูงกว่านั้น ในขณะที่ในประเทศญี่ปุ่นผู้ใช้ส่วนใหญ่จะไม่มีพีซีอยู่ที่บ้านและต้องเสียค่าใช้บริการโทรศัพท์ในอัตราที่แพงมาก ดังนั้นสำหรับชาวญี่ปุ่นส่วนใหญ่แล้ว i-mode เป็นวิธีการเดียวที่ใช้ติดต่อกับระบบอินเตอร์เน็ต

ประการที่สอง วิธีการคิดค่าบริการนั้นแตกต่างกันคือ ในญี่ปุ่นผู้ใช้จะต้องเสียค่าบริการเช่น หนึ่งเหรียญต่อเดือนสำหรับ Yahoo อีกหนึ่งเหรียญต่อเดือนสำหรับ Google และเว็บไซต์อื่นๆ รวมทั้งต้องเสียค่าดาวน์โหลดอีกต่างหาก ส่วนในประเทศตะวันตกจะเสียค่าบริการรายเดือนในระบบเหมาจ่ายที่ถูกมาก

ประการที่สาม ในประเทศญี่ปุ่นช่วงเวลาที่ใช้ i-mode ส่วนใหญ่คือช่วงเวลาเดินทางโดยรถไฟหรือรถไฟใต้ดินจากบ้านไปที่ทำงานหรือโรงเรียนและช่วงเวลาที่เดินทางกลับบ้าน ในยุโรปและอเมริกามีผู้คนจำนวนน้อยมากที่ทำเช่นนั้น ผู้คนส่วนใหญ่จะมีเวลามาใช้งานอินเตอร์เน็ตกับเครื่องพีซีที่บ้านมากกว่าในรถไฟ

ดังที่ได้กล่าวข้างต้นชุดมือถือของระบบ i-mode จะใช้การสื่อสาร circuit switching สำหรับการสนทนาและใช้ packet switching สำหรับการสื่อสารข้อมูล เครือข่ายการสื่อสารข้อมูลนั้นเป็นระบบ CDMA ที่ส่งข้อมูล 128 ไบต์ต่อแพ็กเก็ตที่ความเร็ว 9600 บิตต่อวินาที รูป 7-50 แสดงไดอะแกรมการทำงานของระบบเครือข่ายนี้ ชุดมือถือใช้โพรโตคอล LTP (Lightweight Transport Protocol) ผ่านการสื่อสารไร้สายไปเชื่อมต่อกับเกตเวย์ที่ทำหน้าที่ในการแปลงโพรโตคอล (protocol converter) เกตเวย์จะมีช่องสื่อสารความกว้างสูงที่ใช้สายใยแก้วนำแสงในการเชื่อมต่อกับ i-mode เซิบเวอร์ ซึ่งจะเชื่อมต่อกับบริการที่มีให้ทั้งหมด เมื่อผู้ใช้เลือกบริการจากในเมนู คำร้องขอข้อมูลจะถูกส่งไปที่ i-mode เซิบเวอร์ ซึ่งจะเก็บข้อมูลส่วนใหญ่ไว้ใน cache ของตนเองเพื่อเพิ่มประสิทธิภาพในการให้บริการ คำร้องขอข้อมูลที่ไม่อยู่ในเมนูจะถูกส่งข้าม i-mode เซิบเวอร์ไปยังเซิบเวอร์ผู้ให้บริการนั้น (บนระบบอินเตอร์เน็ต) โดยตรง

ชุดมือถือของ i-mode ประกอบด้วยซีพียูที่ทำงานที่ความเร็วประมาณ 100 MHz มีหน่วยความจำ



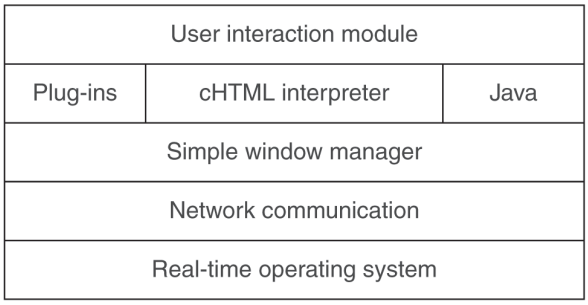
รูป 7-50
โครงสร้างของระบบเครือข่ายข้อมูลใน i-mode ที่แสดงให้เห็นโพรโตคอลการนำส่งข้อมูล

flash ROM หลายเม็กกาไบต์ มีหน่วยความจำ RAM ประมาณ 1 เม็กกาไบต์ และจอภาพแสดงผลขนาดเล็ก จอภาพมาตรฐานมีขนาด 72x94 pixels แต่ชุดมือถือราคาแพงอาจมีหน้าจอภาพขนาด 120x160 pixels ความสามารถในการแสดงสีอยู่ที่ 8 บิตหรือเท่ากับ 256 เฉดสีแม้จะไม่สามารถแสดงรูปภาพที่สวยงามได้แต่ก็เพียงพอที่จะแสดงภาพลายเส้นและภาพการ์ตูนขนาดเล็กได้โดยปกติจะไม่มีเมาส์ให้ใช้งานจึงต้องใช้ปุ่มลูกศรแทนการเคลื่อนที่ของเมาส์

รูป 7-51 แสดงโครงสร้างซอฟต์แวร์ที่ใช้งาน ชั้นล่างสุดประกอบด้วยระบบปฏิบัติการแบบเรียลไทม์ที่ใช้ควบคุมการทำงานของฮาร์ดแวร์ จากนั้นเป็นส่วนที่ควบคุมการสื่อสารข้อมูลซึ่งเป็นแบบเฉพาะของบริษัท NTT DoCoMo เรียกว่า LTP protocol เหนือขึ้นไปเป็นซอฟต์แวร์ควบคุมการทำงานของ window ที่จัดการแสดงผลข้อความและรูปภาพแบบง่าย (มักจะอยู่ในรูปแบบ GIF)

ชั้นสื่อสารระดับที่สี่เป็น Web page interpreter (เช่น บราวเซอร์) I-mode ไม่ได้ใช้ HTML ทั้งหมดจะเป็นเพียงส่วนย่อยของ HTML เรียกว่า cHTML (compact HTML) ชั้นสื่อสารนี้ยังสนับสนุนการทำงานของ helper application และ plug-in เช่นเดียวกับบราวเซอร์ทั่วไป ส่วนบนสุดคือส่วนที่ควบคุมการติดต่อกับผู้ใช้

ต่อไปจะพิจารณา cHTML ดังที่ได้กล่าวมาแล้วภาษานี้เทียบเท่าได้กับ HTML 1.0 โดยมีข้อยกเว้นบางประการและมีการเพิ่มเติมขีดความสามารถบางส่วนเข้าไปเพื่อให้เหมาะสมกับชุดมือถือของ i-mode



รูป 7-51
โครงสร้างของซอฟต์แวร์ในระบบ i-mode

ภาษานี้ได้รับการนำเสนอไปยังองค์การ W3C เพื่อกำหนดเป็นมาตรฐานแต่ก็ไม่ได้รับความสนใจจึงยังคงเป็นภาษาที่ไม่มีมาตรฐานรับรองในปัจจุบัน

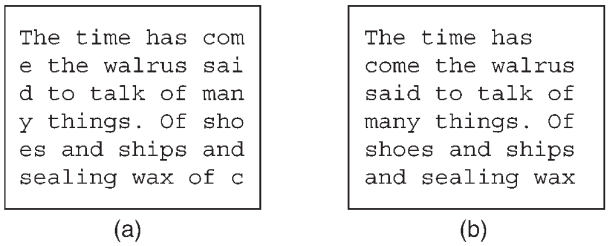
tag ที่มีใช้งานในภาษา HTML 1.0 ส่วนใหญ่ถูกนำมาใช้งานได้ตามปกติรวมทั้ง <html>, <head>, <little>, <body>, <hn>, , <ul>, <ol>, <menu>, <li>,
, <p>, <hr>, <img>, <form>, และ <input> แต่ tag <b> และ <i> ไม่สามารถนำมาใช้ได้

Tag <a> สามารถนำมาใช้ในการเชื่อมต่อเข้ากับเพจอื่นแต่ได้มีการเพิ่มคีย์ "tel" เพื่อใช้ในการหมุนหมายเลขโทรศัพท์ซึ่งมีความหมายคล้ายกับ "mailto" เมื่อ hyperlink ได้เลือกใช้ mailto บราวเซอร์จะแสดงหน้าจอพิเศษเป็นแบบฟอร์มที่ใช้ในการส่งอีเมลล์ไปยังเป้าหมายที่ได้กำหนดไว้ในการเชื่อมต่อ (hyperlink) นั้น เมื่อ hyperlink ที่กำหนดใช้ "tel" ถูกเรียกใช้งาน บราวเซอร์จะทำการหมุนโทรศัพท์ไปยังเป้าหมาย เช่น สมุดโทรศัพท์อาจประกอบด้วยรูปภาพของเจ้าของเบอร์โทรศัพท์ที่ได้บันทึกเอาไว้ เมื่อผู้ใช้เลือกรูปภาพหนึ่งขึ้นมาชุดมือถือของ i-mode ก็จะจัดการหมุนเบอร์โทรศัพท์ของคนผู้นั้นให้ มาตรฐาน URL เกี่ยวกับหมายเลขโทรศัพท์ได้กำหนดไว้ใน RFC 2806

บราวเซอร์ที่ใช้งานร่วมกับภาษา cHTML มีข้อจำกัดอยู่หลายอย่างคือ ไม่สนับสนุนการใช้งาน JavaScript, frames, style sheets, background color และ background images รวมทั้งไม่สนับสนุนการใช้รูปภาพในรูปแบบ JPEG เนื่องจากต้องใช้ระยะเวลานาน (สำหรับชุดมือถือของ i-mode) ในการถอดรหัสเพื่อการแสดงรูปภาพ อย่างไรก็ตามบราวเซอร์สนับสนุน Java applet แต่จำกัดขนาดไว้ที่ 10 กิโลไบต์เนื่องจากมีความเร็วในการถ่ายทอดสัญญาณต่ำมาก

แม้ว่าบริษัท NTT DoCoMo ผู้พัฒนา cHTML จะได้ยกเลิกการใช้ tag บางส่วนออกไปดังที่ได้กล่าวมาแล้ว แต่ก็ได้มีการเพิ่ม tag ใหม่เข้าไปบางส่วน เช่น tag <blink> จะทำให้ข้อความที่เลือกใช้ tag นี้กระพริบได้ และ tag <marquee> ซึ่งจะทำการหมุนข้อความที่เลือก

อีกคุณสมบัติหนึ่งที่น่าสนใจคือคุณสมบัติ "align" สำหรับ tag
 คุณสมบัตินี้มีความจำเป็นเนื่องจากหน้าจอแสดงผลซึ่งมีขนาดเล็กมากนั้นสามารถแสดงข้อความได้ 6 บรรทัด บรรทัดละ 16 ตัวอักษร จึงมักจะมีปัญหาเกี่ยวกับการแบ่งคำที่อาจเกิดขึ้นตรงกลางคำใด ๆ ได้ เช่นในรูป 7-52(a) คุณสมบัติ align จะช่วยลดปัญหานี้ได้โดยการแสดงผลคล้ายกับที่แสดงในรูป 7-52(b) อนึ่งปัญหานี้จะไม่เกิดขึ้นกับภาษาญี่ปุ่นที่ใช้ภาษาเรียกว่า "เคนจิ" เพราะตัวอักษรแต่ละตัวจะมีความหมายคล้ายกับภาษาอังกฤษหนึ่งคำ และตัวอักษรเคนจิแต่ละตัวสามารถเขียนแยกจากกันได้อยู่แล้วจึงไม่มีปัญหาในเรื่องการตัดคำระหว่างบรรทัดเกิดขึ้น



รูป 7-52
ปัญหาการตัดคำบนจอภาพแสดงผล

```
<html>
<body>
<h1> Select an option </h1>
<a href="messages.chtml" accesskey="1"> Check voicemail </a> <br>
<a href="mail.chtml" accesskey="2"> Check e-mail </a> <br>
<a href="games.chtml" accesskey="3"> Play a game </a>
</body>
</html>
```

รูป 7-53
ตัวอย่าง cHTML

คุณสมบัติใหม่อีกอย่างหนึ่งได้แก่ความสามารถของผู้ใช้ในการเลือก hyperlink โดยใช้แป้นพิมพ์ ซึ่งเป็นคุณสมบัติที่สำคัญสำหรับอุปกรณ์คอมพิวเตอร์ที่ไม่มีเมาส์ใช้งาน รูป 7-53 แสดงตัวอย่างโปรแกรมภาษา cHTML ที่ใช้คุณสมบัตินี้

แม้ว่าความสามารถทางด้านผู้ใช้งานจะจำกัดมากแต่ทางฝั่งเซิบเวอร์นั้นเป็นเครื่องคอมพิวเตอร์ความสามารถสูงที่สามารถทำงานต่าง ๆ ได้อย่างมีประสิทธิภาพ เซิบเวอร์สามารถสนับสนุนการใช้ CGI, Perl, PHP, JSP, ASP และสิ่งอื่น ๆ ทั้งหมดที่เว็บเซิบเวอร์ทั่วไปจะสามารถสนับสนุนได้

รูป 7-54 แสดงตัวอย่างการเปรียบเทียบความสามารถระหว่าง WAP กับ i-mode ข้อแตกต่าง

รูป 7-54
การเปรียบเทียบระหว่าง WAP 1.0 กับ i-mode

Feature	**WAP**	**I-mode**
What it is	Protocol stack	Service
Device	Handset, PDA, notebook	Handset
Access	Dial up	Always on
Underlying network	Circuit-switched	Two: circuit + packet
Data rate	9600 bps	9600 bps
Screen	Monochrome	Color
Markup language	WML (XML application)	cHTML
Scripting language	WMLscript	None
Usage charges	Per minute	Per packet
Pay for shopping	Credit card	Phone bill
Pictograms	No	Yes
Standardization	WAP forum open standard	NTT DoCoMo proprietary
Where used	Europe, Japan	Japan
Typical user	Businessman	Young woman

บางอย่างอาจดูเหมือนเป็นเรื่องเล็กน้อยแต่ก็เป็นเรื่องที่สำคัญมาก เช่น เด็กอายุ 15 ปีคงจะไม่มีบัตรเครดิตใช้งานดังนั้นการซื้อของผ่าน e-commerce จึงต้องใช้วิธีการคิดเงินรวมเข้าไปกับใบเรียกเก็บเงินค่าใช้โทรศัพท์รายเดือนแทน

เว็บไร้สายในยุคที่สอง

จากการพัฒนา WAP1.0 ให้กลายเป็นมาตรฐานสำหรับการสื่อสารไร้สายบนโทรศัพท์มือถือนั้นประสบกับความล้มเหลวในขณะที่ระบบ i-mode แม้ว่าจะประสบความสำเร็จในทางการตลาดแต่ก็เป็นเพียง

ตลาดในประเทศญี่ปุ่นเท่านั้นและไม่ได้รับการรับรองให้เป็นมาตรฐานแต่อย่างใด สิ่งที่สมาคม WAP consortium และ NTT DoCoMo ได้เรียนรู้ก็คือจะต้องให้เว็บไซต์จำนวนมากใส่ข้อมูลของตนเองเข้าไปในภาษาที่เป็นระบบเปิด มีความเสถียร และเป็นที่ยอมรับอย่างกว้างขวาง การต่อสู้กับด้วยรูปแบบภาษาที่แตกต่างกันนั้นไม่ส่งผลดีในทางธุรกิจแต่อย่างใด

ทั้งสององค์กรจึงได้ก้าวเข้าสู่โลกของเว็บไร้สายในยุคที่สอง WAP 2.0 ได้รับการพัฒนาออกมาเป็นลำดับแรก จากข้อดีของ WAP 1.0 คือความสามารถในการสนับสนุนระบบเครือข่ายต่างๆ สำหรับมือถือได้ทุกระบบจึงได้รับการคงขีดความสามารถนี้ไว้ อย่างไรก็ตาม การสื่อสารในยุคแรกนั้นใช้ระบบ circuit-switched network แต่ในยุคที่สองจะใช้ระบบ packet-switched network เช่นระบบ GPRS และ WAP 1.0 มีเป้าหมายในการสนับสนุนอุปกรณ์หลากหลายชนิดตั้งแต่ โทรศัพท์มือถือไปจนถึงโน้ตบุ๊คคอมพิวเตอร์ซึ่งในยุคที่สองก็จะคงคุณสมบัติข้อนี้เอาไว้

WAP 2.0 ยังเสนอขีดความสามารถใหม่ๆ ซึ่งได้แก่

1. ระบบผลักดันข้อมูล (push model) และระบบดึงข้อมูล (pull model)
2. สนับสนุนการรวมระบบโทรศัพท์เข้าไปกับอุปกรณ์ที่นำมาใช้งาน
3. การส่งข้อมูลมัลติมีเดีย
4. การใช้ pictogram จำนวน 264 รูป
5. การเชื่อมต่อเข้ากับอุปกรณ์บันทึกข้อมูล
6. สันบสนุน plug-in ในบราวเซอร์

ระบบการดึงข้อมูลนั้นเป็นระบบที่มีใช้งานกันอยู่ทั่วไปนั่นคือผู้ใช้จะร้องขอข้อมูลซึ่งเซิบเวอร์ก็จะจัดส่งมาให้ ระบบผลักดันข้อมูลคือ การจัดส่งข้อมูลมายังผู้ใช้โดยที่ไม่ต้องรอให้ผู้ใช้ร้องขอมาก่อน เช่น การส่งข้อมูลราคาหุ้นมาให้ผู้ใช้อย่างต่อเนื่อง หรือการส่งข้อมูลเกี่ยวกับสภาพการจราจรมาให้ตลอดเวลา

ข้อมูลที่เป็นเสียงพูดและข้อมูลดิจิตอลนั้นกำลังถูกนำเข้ามารวมไว้ด้วยกัน WAP 2.0 ก็สนับสนุนแนวทางนี้ในหลายรูปแบบ ตัวอย่างการรวมการบริการนี้มีให้เห็นในระบบ i-mode ในด้านการเชื่อมโยงข้อความหรือรูปภาพบนหน้าจอเข้ากับหมายเลขโทรศัพท์ดังที่ได้กล่าวมาแล้ว รวมทั้งสนับสนุนความสามารถในการส่งข้อมูลมัลติมีเดียด้วย

ในระบบ i-mode นั้นได้มีการพัฒนาอักษรรูปภาพใหม่ขึ้นมาใช้งานเรียกว่า emoji สมาคม WAP consortium จึงได้เลียนแบบด้วยการประดิษฐ์อักษรภาพขึ้นมาจำนวน 264 รูปซึ่งได้แก่ รูปสัตว์ เครื่องใช้ เสื้อผ้า หน้าตาที่แสดงความรู้สึก อาหาร ชิ้นส่วนของร่างกาย เพศ แผนที่ เครื่องดนตรี ต้นไม้ เวลา เครื่องมือ ยานพาหนะ อาวุธ และสภาพอากาศ (เรียกว่า pictogram หรืออักษรภาพ) สิ่งที่น่าสนใจก็คือ ตัวมาตรฐานเองได้กำหนดเฉพาะชื่อของอักษรภาพเอาไว้เท่านั้นแต่ไม่ได้กำหนดรูปภาพนั้นๆ ขึ้นมาใช้งานจริง (ปล่อยให้เป็นหน้าที่ของผู้ให้บริการแต่ละแห่งจะจัดทำขึ้นมาเอง) ทั้งนั้นเนื่องจากเกรงว่ารูปภาพบางภาพอาจไม่เหมาะสมกับวัฒนธรรมท้องถิ่นนั้นๆ ปัญหานี้ไม่เกิดขึ้นกับอักษรภาพ emoji เพราะเป็นอักษรภาพที่พัฒนาขึ้นมาสำหรับชาวญี่ปุ่นโดยตรง

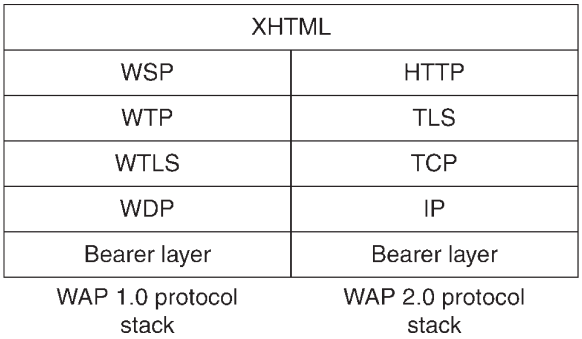
การสนับสนุนการเชื่อมต่อเข้ากับอุปกรณ์บันทึกข้อมูลไม่ได้หมายความว่าโทรศัพท์ WAP 2.0 จะต้องมีฮาร์ดดิสก์ขนาดใหญ่ติดมาด้วยเพราะว่า flash ROM ก็จัดว่าเป็นอุปกรณ์บันทึกข้อมูลขนาดใหญ่เหมือนกัน โทรศัพท์ที่มีกล้องถ่ายรูปดิจิตอลอยู่ในตัวสามารถที่จะเก็บรูปที่ถ่ายไว้ใน flash ROM ไว้ชั่วคราวก่อนที่จะส่งภาพนั้นเข้าสู่ระบบอินเตอร์เน็ตในภายหลัง ท้ายที่สุด plug-in สามารถนำมาใช้เพิ่มขีดความสามารถของบราวเซอร์ได้รวมทั้ง script language อื่นๆ ด้วย

ความแตกต่างทางด้านเทคนิคระหว่าง WAP 1.0 และ WAP 2.0 นั้นมีมากพอสมควร ข้อแตกต่างที่สำคัญสองประการก็คือ โพรโตคอลแสต็ก และภาษา markup language WAP 2.0 ยังคงสนับสนุนโพรโตคอลแสต็กแบบเก่าดังที่แสดงในรูป 7-48 แต่ในเวลาเดียวกันก็สนับสนุนมาตรฐานอินเตอร์เน็ตแสต็กที่มีทั้งโพรโตคอล TCP และ HTTP/1.1 ด้วย อย่างไรก็ตามได้มีการดัดแปลงโพรโตคอล TCP (แต่ยังสามารถทำงานร่วมกันได้) ใน 4 ประเด็นหลักเพื่อทำให้การเขียนโค้ดนั้นง่ายลงคือ (1) ใช้ window ขนาดคงที่ขนาด 64 กิโลไบต์ (2) ไม่ทำงานในระบบ slow start (3) มี MTU ขนาดสูงสุด 1500 ไบต์ และ (4) ใช้อัลกอริทึมในการจัดส่งข้อมูลใหม่แตกต่างไปจากเดิมเล็กน้อย โพรโตคอล TLS นั้นเป็น โพรโตคอลที่เกี่ยวข้องกับการรักษาความปลอดภัยที่กำหนดมาตรฐานโดยองค์กร IETF ซึ่งจะได้กล่าวถึงในบทที่ 8 รูป 7-55 แสดงโพรโตคอลแสต็กทั้งสองรุ่น

สิ่งที่แตกต่างกันประการที่สองคือภาษา markup language WAP 2.0 สนับสนุนภาษา XHTML (basic) ซึ่งสร้างขึ้นมาเพื่ออุปกรณ์ไร้สายขนาดเล็กโดยเฉพาะ เนื่องจากบริษัท NTT DoCoMo ได้ตกลงที่จะสนับสนุนการใช้ภาษานี้ นักออกแบบเว็บไซต์จะสามารถใช้ภาษานี้และแน่ใจได้ว่าเว็บไซต์จะสามารถใช้งานได้กับทั้งระบบอินเตอร์เน็ตปกติและระบบอินเตอร์เน็ตผ่านระบบสื่อสารไร้สาย การตัดสินใจนี้จึงเท่ากับเป็นการยุติศึกการเลือกใช้ภาษา markup language ที่หยุดยั้งการเจริญเติบโตของอุตสาหกรรมเว็บไร้สาย

ภาษา XHTML ได้ถูกสร้างขึ้นมาเพื่อใช้งานในโทรศัพท์มือถือ โทรทัศน์ พีดีเอ เครื่องขายของอัตโนมัติ เพจเจอร์ รถยนต์ เครื่องเล่นเกมส์ หรือแม้กระทั่งนาฬิกา ด้วยเหตุผลนี้จึงไม่สนับสนุนการใช้ style sheets, scripts หรือ frames แต่ tag ส่วนใหญ่ก็ยังคงมีอยู่ตามปกติ ซึ่งถูกนำมาจัดกลุ่มเป็น 11 กลุ่ม (modules) ดังแสดงตัวอย่างในรูป 7-56

แม้ว่าการตกลงในการใช้ภาษา XHTML basic จะประสบความสำเร็จแต่สิ่งที่อาจคุกคามอนาคตของ WAP และ i-mode ก็ยังคงมีอยู่นั่นคือ มาตรฐานการสื่อสารไร้สาย 802.11 เว็บไร้สายในยุคที่



รูป 7-55 WAP 2.0 สนับสนุนโพรโตคอลแสต็กทั้งสองแบบ

รูป 7-56
The XHTML Basic modules and tags.

Module	Req.?	Function	Example tags
Structure	Yes	Doc. structure	body, head, html, title
Text	Yes	Information	br, code, dfn, em, h*n*, kbd, p, strong
Hypertext	Yes	Hyperlinks	a
List	Yes	Itemized lists	dl, dt, dd, ol, ul, li
Forms	No	Fill-in forms	form, input, label, option, textarea
Tables	No	Rectangular tables	caption, table, td, th, tr
Image	No	Pictures	img
Object	No	Applets, maps, etc.	object, param
Meta-information	No	Extra info	meta
Link	No	Similar to <a>	link
Base	No	URL starting point	base

สองน่าจะทำงานที่ความเร็ว 384 กิโลบิตต่อวินาทีซึ่งดีกว่าความเร็วที่ใช้ในยุคแรกมาก (9.6 กิโลบิตต่อวินาที) แต่ก็ยังช้ากว่ามาตรฐาน 802.11 ที่ทำงานที่ความเร็ว 11 ล้านบิตต่อวินาทีหรือ 54 ล้านบิตต่อวินาที แม้ว่า 802.11 จะไม่มีใช้งานในทุกสถานที่เหมือนอย่าง WAP หรือ i-mode แต่สถานที่จำนวนมากเช่น โรงแรม ภัตตาคาร ร้านค้า สถานีขนส่ง สนามบินและอื่นๆ ต่างก็ติดตั้งสถานีฐานสำหรับ 802.11 มากยิ่งขึ้นดังนั้นผู้ใช้จำนวนมากคงจะยินดีที่จะเดินไปยังสถานที่เหล่านี้เพื่อใช้งานระบบสื่อสารไร้สาย 802.11 ในการติดต่อกับระบบอินเตอร์เน็ตด้วยความเร็วสูงมากกว่าที่จะใช้ระบบ WAP

อย่างไรก็ตาม แม้ว่าภัตตาคารอาจจะยอมลงทุนติดตั้งสถานีฐานสำหรับระบบ 802.11 แต่ชาวชนบทคงจะไม่มีใครทำเช่นนี้ ดังนั้นพื้นที่ให้บริการของระบบ 802.11 คงจะเป็นพื้นที่เป็นจุดแคบๆ (ระยะทางสื่อสารไม่กี่ร้อยเมตร) ที่อยู่ในตัวเมืองเป็นหลัก แนวโน้มที่น่าจะเป็นไปได้ก็คืออุปกรณ์ที่ใช้ระบบสื่อสารไร้สายคงจะทำงานร่วมกับทั้งสองระบบคือ เมื่อสามารถรับสัญญาณของระบบ 802.11 ได้ก็จะใช้ระบบนี้แต่ถ้ารับสัญญาณไม่ได้ก็คงเปลี่ยนไปใช้ระบบ WAP แทน

7.4 มัลติมีเดีย

โดยความหมายแล้วมัลติมีเดีย (multimedia) หมายถึงการใช้การสื่อสารตั้งแต่สองรูปแบบขึ้นไป ดังนั้นหนังสือเล่มนี้ก็อาจใช้คำว่าเป็นหนังสือมัลติมีเดียก็ได้เพราะใช้สื่อที่เป็นตัวอักษรและสื่อที่เป็นรูปภาพประกอบกัน อย่างไรก็ตามสำหรับคนส่วนมากแล้ว สื่อที่ใช้จะต้องเป็นสื่อชนิดต่อเนื่อง (continuous media) นั่นคือสื่อทั้งหลายจะต้องถูกนำมาแสดงในช่วงเวลาที่กำหนดไว้อย่างเหมาะสม และโดยปกติจะมีการโต้ตอบกับผู้ใช้ด้วย ในทางปฏิบัติสื่อทั้งสองชนิดจะหมายถึง "audio" และ "video" ซึ่งหมายถึง "เสียง" และ "ภาพวิดีทัศน์" นั่นเอง

อย่างไรก็ตาม ผู้คนจำนวนมากมักจะอ้างอิงถึง "audio" แต่เพียงอย่างเดียว เช่น "telephony" หรือ "Internet radio" ว่าเป็นมัลติมีเดียเหมือนกัน ซึ่งอันที่จริงแล้วสื่อประเภทนี้ควรจะถูกเรียกว่าเป็น "streaming media" จึงจะถูกต้องมากกว่า แต่ในที่นี้ก็จะอนุโลมเรียกว่าเป็นมัลติมีเดียชนิดหนึ่ง ในหัวข้อต่อไปนี้จะกล่าวถึงวิธีการที่คอมพิวเตอร์ทำการประมวลผล audio และ video วิธีการบีบอัด

ข้อมูลประเภทนี้ และการนำไปประยุกต์ใช้งานทางระบบเครือข่าย

7.4.1 แนะนำ digital audio

Audio (sound) wave หรือสัญญาณข้อมูลเสียงเป็นสัญญาณเสียงขนาดหนึ่งมิติ เมื่อสัญญาณเสียงเดินทางเข้าสู่หูของมนุษย์ ส่วนของใบหูเรียกว่า eardrum จะเกิดการสั่นสะเทือน ทำให้ส่วนของกระดูกชิ้นเล็กๆ ภายในหูสั่นตามไปด้วยและส่งสัญญาณทางประสาทไปยังสมอง ในทำนองเดียวกันเมื่อคลื่นเสียงกระทบไมโครโฟน ชิ้นส่วนภายในไมโครโฟนจะสร้างสัญญาณไฟฟ้าขึ้นมาเพื่อใช้แทนความหมายของสัญญาณเสียงด้วยความสูงของคลื่นเสียงเทียบกับหน่วยเวลา กระบวนการแทนคลื่นเสียงด้วยคลื่นไฟฟ้า การจัดเก็บ และการถ่ายทอดสัญญาณดังกล่าวเป็นองค์ประกอบหลักในการศึกษาเรื่องมัลติมีเดีย

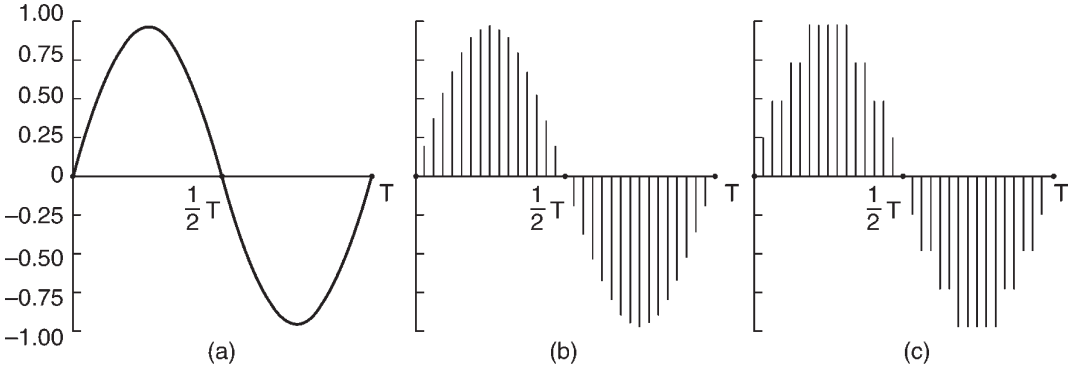
ช่วงคลื่นที่หูมนุษย์สามารถรับฟังได้อยู่ระหว่าง 20 Hz ถึง 20,000 Hz สัตว์บางชนิด เช่นสุนัขสามารถได้ยินเสียงที่มีความถี่สูงกว่านี้ หูจะได้ยินเสียงในลักษณะของ logarithm ดังนั้นอัตราส่วนของเสียงสองเสียงที่มีพลังงาน A และ B จะถูกแสดงในรูปแบบของเดซิเบล (dB; decibels) โดยมีสูตรการคำนวณคือ

$$dB = 10 \log_{10} (A/B)$$

ถ้ากำหนดให้ขีดจำกัดล่างของเสียงที่สามารถรับฟังได้ (ความดันประมาณ 0.0003 dyne/cm2) สำหรับคลื่น sine wave ขนาด 1 kHz เป็น 0 dB แล้วเสียงการสนทนาปกติจะมีความดังประมาณ 50 dB และเสียงที่ทำให้ปวดแก้วหูได้จะมีความดังประมาณ 120 dB

หูคนเรานั้นมีความไวในการรับรู้ความรู้สึกของเสียงได้เพียงไม่กี่มิลลิวินาทีในขณะที่สายตาคนเรานั้นจะไม่สามารถจับความเปลี่ยนแปลงที่เกิดขึ้นกับภาพภายในไม่กี่มิลลิวินาทีได้ ข้อสังเกตนี้ทำให้ทราบได้ว่าสัญญาณที่เกิดการกระตุกเพียงไม่กี่มิลลิวินาทีในระหว่างการถ่ายทอดสัญญาณมัลติมีเดียจะมีผลต่อคุณภาพในการรับฟังเสียงมากกว่าที่จะมีผลต่อคุณภาพในการดูภาพมัลติมีเดียนั้น

คลื่นสัญญาณเสียงสามารถที่จะเปลี่ยนแปลงไปอยู่ในรูปสัญญาณดิจิตอลได้โดยอุปกรณ์ที่เรียกว่า ADC (Analog Digital Converter) อุปกรณ์ ADC จะรับข้อมูลนำเข้าเป็นระดับแรงดันไฟฟ้าแล้วสร้างเป็นข้อมูลเลขฐานสองออกมาเป็นผลลัพธ์ ในรูป 7-57(a) เป็นตัวอย่างสัญญาณ sine wave ในการแปลงสัญญาณนี้ให้กลายเป็นสัญญาณดิจิตอล จะต้องทำการวัดสัญญาณ (เรียกว่าการทำ sampling) ในทุกๆ (T วินาทีดังแสดงในรูป 7-57(b)



รูป 7-57
(a) A sine wave.
(b) Sampling the sine wave.
(c) Quantizing the samples to 4 bits

ตัวอย่างสัญญาณดิจิตอลที่ได้มานั้นจะไม่เหมือนสัญญาณเดิมทุกประการ ตัวอย่างสัญญาณในรูป 7-57(c) จะยอมให้มีค่าเพียง 9 ค่าที่อยู่ระหว่าง -1.0 ถึง +1.0 โดยเพิ่มค่าครั้งละ 0.25 (นั่นคือ -1.0, -0.75, -0.5, -0.25, 0.0, 0.25, 0.5, 0.75, และ 1.0) ถ้าใช้การสุ่มสัญญาณขนาด 8 บิตก็จะมีค่าแตกต่างกันได้ 256 ค่า และ 16 บิตจะมีมากถึง 65,536 ค่า ความผิดพลาดที่เกิดขึ้นเนื่องจากการกำหนดจำนวนบิตคงที่ที่ใช้แทน sample ของสัญญาณเสียงเรียกว่า "quantization noise" ถ้ามีค่ามากเกินไปหูของคนเราจะสามารถสังเกตได้

ตัวอย่างที่รู้จักกันโดยทั่วไปสองตัวอย่างเกี่ยวกับการสุ่มวัดสัญญาณเสียงคือ โทรศัพท์ และซีดีเพลง ระบบโทรศัพท์จะใช้วิธีการเรียกว่า pulse code modulation ในการวัดสัญญาณเสียงเป็นขนาด 8 บิต ด้วยการวัด 8,000 ครั้งต่อวินาที ในทวีปอเมริกาเหนือและประเทศญี่ปุ่นจะใช้เพียง 7 บิตเหลืออีก 1 บิต ไว้สำหรับข้อมูลสำหรับการตรวจสอบความถูกต้อง ในยุโรปจะใช้ทั้ง 8 บิตในการบันทึกข้อมูล ระบบนี้จะทำให้เกิดอัตราการส่งข้อมูลที่ 56,000 บิตต่อวินาทีถึง 64,000 บิตต่อวินาที ด้วยการวัดสัญญาณ 8,000 ครั้งต่อวินาที สัญญาณเสียงที่มีความถี่เกิน 4kHz จะสูญหายไป

ซีดีเพลงมีการวัดสัญญาณเสียง 44,100 ครั้งต่อวินาที สามารถจับสัญญาณเสียงได้ถึงความถี่ 22,050 Hz ซึ่งดีมากสำหรับหูมนุษย์ สัญญาณแต่ละสัญญาณมีขนาด 16 บิตซึ่งสามารถแยกความแตกต่างได้ 65,536 ระดับ แม้ว่าสัญญาณ dynamic range สำหรับหูมนุษย์นั้นมีค่าประมาณ 1 ล้านเมื่อวัดเป็นระดับขั้นที่หูมนุษย์สามารถแยกความแตกต่างได้ ดังนั้นการใช้สัญญาณขนาด 16 บิตจึงทำให้เกิด quantization noise ด้วยอัตราการวัดสัญญาณที่ 44,100 ครั้งต่อวินาทีสำหรับข้อมูลครั้งละ 16 บิต ซีดีเพลงจะต้องการช่องสื่อสารขนาดไม่น้อยกว่า 705.6 กิโลบิตต่อวินาทีสำหรับเสียง mono และ 1.411 ล้านบิตต่อวินาทีสำหรับสัญญาณเสียง stereo แม้ว่าข้อมูลนี้จะมีปริมาณน้อยกว่าข้อมูลที่เกิดจากวิดีทัศน์ (จะกล่าวถึงต่อไป) แต่ก็มากพอที่จะทำให้ช่องสื่อสารแบบ T1 นั้นเกือบเต็มได้ในการถ่ายทอดสัญญาณเพลงซีดีแบบ stereo ในแบบเรียลไทม์โดยไม่มีการบีบอัดสัญญาณเลย

การแปลงสัญญาณเสียงเพลงเป็นข้อมูลดิจิตอล (เรียกว่าการทำ digitize) นั้นสามารถทำได้โดยซอฟต์แวร์ทั่วไปซึ่งผู้ใช้สามารถบันทึก แสดงผล แก้ไข ผสมเสียง และจัดเก็บสัญญาณเสียงที่มาจากหลายแหล่งได้ โดยทั่วไปแล้วการบันทึกเสียงทุกชนิดในปัจจุบันที่ทำโดยมืออาชีพจะใช้ระบบดิจิตอลทั้งนั้น

เสียงเพลงเป็นกรณีเฉพาะของสัญญาณเสียงโดยทั่วไปอย่างหนึ่ง สัญญาณเสียงที่สำคัญอีกอย่างหนึ่งคือเสียงพูดของมนุษย์ซึ่งอยู่ในช่วงความถี่ประมาณ 600 ถึง 6000 Hz เสียงพูดมนุษย์ประกอบด้วยเสียงสระ (vowels) และเสียงคล้องจองกัน (consonant) ซึ่งมีคุณสมบัติที่แตกต่างกัน เสียงสระนั้นเกิดขึ้นเมื่อกล่องเสียงไม่ถูกขวางกั้นทำให้เกิดเสียงสะท้อนที่มีความถี่ที่ขึ้นอยู่กับขนาดและรูปร่างของกล่องเสียงและตำแหน่งของลิ้นและกรามของผู้ออกเสียงนั้น เสียงชนิดนี้เกือบจะเป็นจังหวะในทุก ๆ 30 มิลลิวินาที ส่วนเสียงคล้องจองนั้นเกิดขึ้นเมื่อเส้นทางของกล่องเสียงนั้นถูกบดบังเป็นบางส่วนจึงมีความ ซับซ้อนในรูปแบบมากกว่าเสียงสระ

7.4.2 การบีบอัดข้อมูลเสียง

เสียงคุณภาพระดับซีดีจะต้องอาศัยช่องสื่อสารที่มีความกว้างถึง 1.411 ล้านบิตต่อวินาทีดังที่ได้

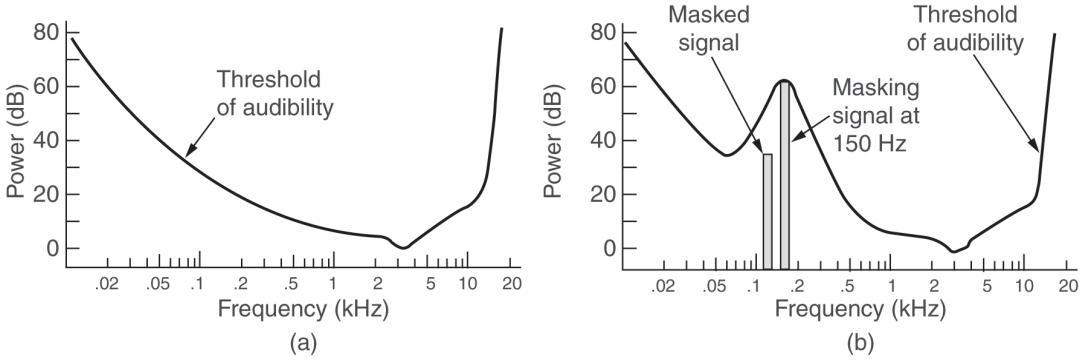
กล่าวมาแล้วซึ่งเห็นได้ชัดเจนว่าจะต้องใช้เทคนิคในการบีบอัดข้อมูลให้มีขนาดเล็กลงเพียงพอที่จะสามารถนำมาใช้งานได้จริงในระบบอินเตอร์เน็ต ด้วยเหตุผลนี้จึงได้มีการคิดค้นอัลกอริทึมในการบีบอัดข้อมูลขึ้นมามากมาย อัลกอริทึมที่ได้รับความนิยมมากที่สุดคือ MPEG ซึ่งประกอบด้วยการทำงานสามชั้น หนึ่งในสามชั้นนี้คือ MP3 (MPEG audio layer 3) ซึ่งเป็นส่วนที่รู้จักกันทั่วไป เพลงบนอินเตอร์เน็ตส่วนใหญ่ก็ใช้วิธีบีบอัดข้อมูลแบบนี้ MP3 เป็นส่วนบีบอัดข้อมูลเสียงของ MPEG ซึ่งใช้สำหรับการบีบอัดข้อมูลที่เป็นภาพยนต์ การบีบอัดข้อมูลวิดีทัศน์นั้นจะได้กล่าวถึงในหัวข้อต่อไป

การบีบอัดข้อมูลเสียงสามารถทำได้สองแนวทาง แนวทางแรกเรียกว่า waveform coding วิธีการนี้จะอาศัยการแปลงรูปแบบสัญญาณเสียงโดยใช้ฟังก์ชันทางคณิตศาสตร์คือ Fourier transform แปลงสัญญาณเสียงให้กลายเป็น frequency component รูป 2-1(a) แสดงตัวอย่างฟังก์ชันของเวลาและค่าของ Fourier amplitudes ค่า amplitude ของแต่ละ component จะถูกนำมาเข้ารหัสอีกต่อหนึ่ง เป้าหมายก็คือการสร้าง waveform ขึ้นมาใหม่ให้มีความเที่ยงตรงที่ทางฝั่งผู้รับโดยใช้จำนวนบิตให้น้อยที่สุด

อีกแนวทางหนึ่งเรียกว่า perceptual coding ซึ่งอาศัยประโยชน์จากความบกพร่องในการได้ยินของมนุษย์ในการเข้ารหัสสัญญาณให้สามารถแสดงเสียงได้คล้ายกับของจริงแต่จะมองเห็นข้อแตกต่างกันได้อย่างชัดเจนเมื่อใช้เครื่องมือตรวจวัด วิธีการนี้พัฒนาขึ้นมาจากศาสตร์ที่เรียกว่า "Psychoacustics" ซึ่งหมายถึงวิชาที่ว่าด้วยเรื่องการรับฟังเสียงของมนุษย์ วิธี MP3 ก็เป็นตัวอย่างหนึ่งที่อยู่ในกลุ่มนี้

คุณสมบัติที่สำคัญของ perceptual coding คือการที่สัญญาณเสียงบางส่วนสามารถนำมาใช้เป็น mask ของเสียงอื่นได้ ลองนึกดูว่าท่านกำลังนั่งเป่าขลุ่ยอยู่ในสวนสาธารณะในวันหนึ่ง ทันใดนั้น คนงานสร้างถนนก็เดินเครื่องเจาะถนนทำให้ไม่มีผู้ใดได้ยินเสียงขลุ่ยของท่านอีกต่อไปเนื่องจากเสียงขลุ่ยถูก "mask" โดยเสียงของเครื่องเจาะถนน สำหรับการถ่ายทอดสัญญาณเสียง มีความเป็นไปได้ที่จะทำการเข้ารหัสเฉพาะคลื่นความถี่ที่ใช้โดยเครื่องเจาะถนนเพราะผู้ฟังไม่สามารถจะได้ยินเสียงขลุ่ยอยู่ดี วิธีการนี้เรียกว่า "frequency masking" ซึ่งหมายถึงความสามารถของเสียงที่ดังกว่าที่อยู่ในย่านความถี่หนึ่งสามารถซ่อนเสียงอีกเสียงหนึ่งที่เบากว่าที่อยู่คนละย่านความถี่กันที่จะสามารถได้ยินได้เมื่อกรองเสียงที่ดังกว่าออกไป อันที่จริงแล้ว แม้ว่าเสียงเครื่องเจาะถนนจะหยุดไปแล้วเสียงขลุ่ยจะยังคงไม่ได้ยินอยู่อีกระยะเวลาหนึ่งเนื่องจากประสาทหูกำลังปรับตัวโดยปกติประสาทหูจะปรับลดระดับการได้ยินลงเมื่อมีเสียงดังเกิดขึ้นซึ่งจะต้องใช้ระยะเวลาหนึ่งในการปรับตัวเพิ่มขีดความสามารถในการได้ยินขึ้นเป็นปกติเมื่อเสียงดังนั้นได้หยุดไปแล้ว เรียกปรากฏการณ์นี้ว่า "temporal masking"

เพื่อทำให้ปรากฏการณ์นี้ชัดเจนยิ่งขึ้นลองพิจารณาการทดลองที่ 1 คนผู้หนึ่งนั่งอยู่ในห้องที่เงียบมากแล้วใส่หูฟังที่เชื่อมต่อเข้ากับการ์ดเสียงของเครื่องพีซีเครื่องหนึ่ง เครื่องพีซีนั้นทำการสร้างเสียงที่ความถี่ 100 Hz ขึ้นมาโดยเริ่มต้นด้วยเสียงเบาและค่อย ๆ เร่งให้ดังขึ้นเรื่อย ๆ คนผู้นั้นได้รับคำสั่งให้แสดงสัญญาณรับรู้ในทันทีที่ได้ยินเสียง เครื่องคอมพิวเตอร์จะทำการบันทึกระดับของพลังงานเสียงเมื่อคนผู้นั้นส่งสัญญาณ การทดลองนี้จะทำซ้ำกันหลายครั้งโดยเปลี่ยนความถี่เป็น 200 Hz, 300 Hz และความถี่อื่น ๆจนถึงระดับสูงสุดที่หูมนุษย์จะสามารถรับฟังได้ เมื่อทำการทดลองเช่นนี้กับคนหลายคนและนำค่าเฉลี่ยมาหาระดับความดังของเสียงในความถี่ต่าง ๆ ที่คนสามารถได้ยินได้แล้วสร้างเป็นรูปกราฟดังแสดงในรูป 7-58(a) สิ่งที่ได้รับโดยตรงจากการทดลองนี้ก็คือ รูปกราฟนี้บอกให้ทราบว่าไม่มีความจำเป็นในการเข้ารหัสความถี่เสียงใดที่มีระดับพลังงานต่ำกว่ารูปกราฟ เรียกเส้นกราฟนี้ว่า "threshold of audibility"



รูป 7-58
(a) The threshold of audibility. (b) The masking effect.

เช่น เสียงที่ความถี่ 100 Hz ถ้ามีระดับความดังต่ำกว่า 20 dB ซึ่งเป็นระดับที่ต่ำกว่า threshold of audibility แล้ว (ดูรูป 7-58(a) ประกอบ) ก็ไม่มีความจำเป็นจะต้องสร้างเสียงนั้นขึ้นมาให้เสียเวลา เพราะผู้รับฟังจะไม่ได้ยินอยู่ดี

ตัวอย่างที่ 2 ให้ทำการทดลองแบบที่ 1 อีกครั้งโดยครั้งนี้ให้เปิดเสียงหนึ่งที่มีความถี่คงที่ เช่น 150 Hz ทำการรบกวนอยู่ตลอดเวลา สิ่งที่ค้นพบก็คือ threshold of audibility สำหรับความถี่ที่ใกล้เคียงกับ 150 Hz จะมีค่าสูงขึ้นดังที่แสดงในรูป 7-58(b)

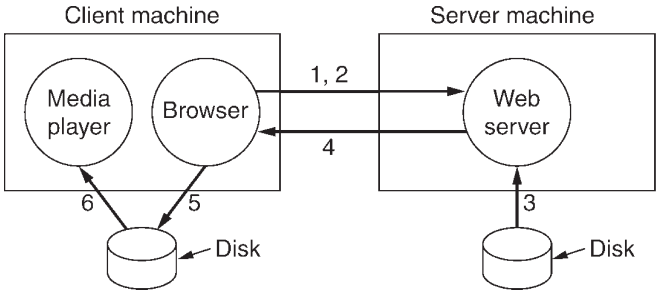
ผลจากการทดลองครั้งที่สองนี้ทำให้ทราบว่า ถ้ามีการตรวจสอบดูว่าเสียงในความถี่ใดที่ถูกบดบัง (masked) โดยเสียงที่มีระดับพลังงานสูงกว่าที่มีความถี่ใกล้เคียงกันก็จะสามารถลบเสียงนั้นออกไปได้ โดยไม่มีผลต่อการได้ยินของมนุษย์ แม้ว่าเสียงที่มีพลังงานสูงกว่านั้นจะได้สิ้นสุดไปแล้ว คุณสมบัติ temporal masking ทำให้สามารถลบเสียงที่ถูกรบกวนนั้นออกไปได้อีกระยะเวลาหนึ่ง (ช่วงเวลาที่หูทำการปรับตัว) หลักการที่สำคัญของ MP3 ก็คือการใช้ Fourier transform ในการแปลงรูปแบบสัญญาณเสียงเพื่อให้ได้ระดับความดังของแต่ละความถี่แล้วจัดการส่งสัญญาณไปเฉพาะความถี่ที่ไม่ถูกรบกวนเท่านั้น ทำให้การเข้ารหัสแบบนี้ใช้จำนวนบิตลดลงเป็นอย่างมาก

7.4.3 Streaming Audio

การนำเทคโนโลยีเสียงเพลงดิจิตอลไปประยุกต์ใช้งานในระบบเครือข่ายนั้นแบ่งออกได้เป็นสามกลุ่มใหญ่ ๆ กลุ่มแรกเรียกว่า streaming audio นั่นคือการฟังเพลงผ่านระบบอินเตอร์เน็ต หรือเรียกว่า music on demand อีกสองกลุ่มที่เหลือคือ Internet radio และ Voice over IP

ระบบอินเตอร์เน็ตเต็มไปด้วยเว็บไซต์ที่ให้บริการเพลงดิจิตอล เว็บไซต์ส่วนหนึ่งจะให้ผู้ใช้คลิ๊กเลือกเพลงที่ต้องการแล้วเล่นเพลงนั้นให้ฟัง บางแห่งก็ให้บริการฟรี บางแห่งก็คิดค่าบริการ รูป 7-59 แสดงขั้นตอนในการขอฟังเพลงดิจิตอลเหล่านี้

กระบวนการขอฟังเพลงเริ่มต้นด้วยการที่ผู้ฟังคลิ๊กเลือกเพลงที่ต้องการจากนั้นบราวเซอร์ก็จะเริ่มทำงาน ขั้นตอนที่ 1 บราวเซอร์จะทำการจัดตั้งช่องสื่อสาร TCP เข้ากับเว็บเซิบเวอร์ที่เป็นเจ้าของเพลงนั้น ขั้นตอนที่ 2 บราวเซอร์จะส่งคำร้องของข้อมูล GET ในรูปแบบ HTTP เพื่อของข้อมูลที่เป็นไฟล์เพลงนั้น ขั้นต่อไป (ขั้นตอนที่ 3 และที่ 4) เซิบเวอร์จะทำการดึงไฟล์เพลงนั้น (อาจจะอยู่ในรูปแบบ MP3 หรือแบบอื่นก็ได้) ขึ้นมาจากดิสก์และส่งกลับมาที่บราวเซอร์ ถ้าไฟล์นั้นมีขนาดใหญ่มากก็จะใช้วิธีการดึงขึ้นมาและจัดส่งทีละบล็อคจนกว่าจะหมด



รูป 7-59 การให้บริการฟังเพลงดิจิตอลบนอินเตอร์เน็ต

การใช้ MIME type เช่น audio/mp3 (หรือใช้นามสกุลของไฟล์เพลง) บราวเซอร์จะทำการตรวจสอบดูว่าจะทำการเล่นไฟล์เพลงนั้นได้อย่างไร โดยปกติมักจะมี helper application เช่น RealOne Player, Windows Media Player หรือ Winamp ที่สามารถนำมาใช้เล่นไฟล์เพลงได้ เนื่องจากวิธีการปกติที่บราวเซอร์สื่อสารกับ helper application คือการบันทึกข้อมูลที่ต้องการสื่อสารนั้นไว้ในไฟล์ชั่วคราว ดังนั้นบราวเซอร์จะทำการบันทึกไฟล์เพลงทั้งไฟล์ลงในไฟล์ชั่วคราวที่สร้างขึ้นมาไว้ในดิสก์ (ขั้นตอนที่ 5) จากนั้นบราวเซอร์จะไปเรียก helper application ขึ้นมาทำงานและจัดการส่งไฟล์ชั่วคราวนั้นไปให้ ขั้นตอนที่ 6 helper application จะทำการเล่นเพลงจากไฟล์ชั่วคราวนั้น

โดยพื้นฐานแล้ว แนวทางการทำงานแบบนี้ถูกต้องและจะสามารถเล่นเพลงที่เลือกไว้ได้ ปัญหาประการเดียวที่มีคือไฟล์เพลงทั้งไฟล์จะต้องถูกส่งผ่านระบบอินเตอร์เน็ตมาที่ผู้ฟังก่อนที่จะเริ่มเล่นเพลงนั้นได้ ถ้าไฟล์เพลงนั้นมีขนาด 4 ล้านไบต์ (ขนาดโดยเฉลี่ยของเพลง MP3) และผู้ฟังใช้โมเด็มที่มีความเร็ว 56 kbps ผู้ฟังจะต้องรอเป็นระยะเวลานานเกือบ 10 นาที (รอการดาวน์โหลดไฟล์เพลง) จึงจะเริ่มได้ยินเพลงที่ต้องการฟัง ผู้ฟังจำนวนไม่น้อยที่ไม่ต้องการรอนานขนาดนี้ในทุก ๆ เพลงที่ต้องการฟัง

เพื่อแก้ปัญหานี้โดยไม่ต้องเปลี่ยนวิธีการที่บราวเซอร์ทำงาน เว็บไซต์ที่ให้บริการเพลงจึงได้คิดหาวิธีใหม่ที่ทำงานดังนี้ ไฟล์ที่เชื่อมต่อเข้ากับ hyperlink ที่เป็นชื่อเพลงนั้นจะไม่ใช่ไฟล์เพลงตามปกติแต่จะเป็นไฟล์พิเศษเรียกว่า metafile ซึ่งเป็นเพียงไฟล์ขนาดเล็กมากที่มีเพียงชื่อของไฟล์เพลงที่ต้องการ เช่น

```
rtsp://joes-audio-server/song-0025.mp3
```

เมื่อบราวเซอร์ได้รับข้อความนี้ก็จะจัดการบันทึกลงในไฟล์ชั่วคราว จัดการเรียก helper application ขึ้นมาทำงาน และส่งต่อไฟล์ชั่วคราวนี้ไปให้ตามปกติ helper application จะอ่านข้อความในไฟล์ชั่วคราวและพบว่าเป็น URL อย่างหนึ่งจึงทำการติดต่อไปที่เซิบเวอร์ joe-audo-server เพื่อร้องขอไฟล์เพลง ด้วยวิธีการนี้ บราวเซอร์จะไม่เข้ามายุ่งเกี่ยวด้วยอีกต่อไป

ส่วนมากแล้วชื่อเซิบเวอร์ที่อยู่ใน metafile จะไม่เหมือนกับชื่อเว็บเซิบเวอร์ทั่วไป อันที่จริงแล้วก็ไม่ใช่ HTTP เซิบเวอร์ด้วยซ้ำไปแต่จะเป็นเซิบเวอร์พิเศษเรียกว่า media server ในตัวอย่างที่ยกมานี้คือเซิบเวอร์ประเภท RTSP (Real Time Streaming Protocol) กำหนดไว้ในมาตรฐาน RFC 2326

โปรแกรม media player (โปรแกรมที่เล่นเพลง) จะมีหน้าที่ 4 ประการคือ

1. จัดการส่วนติดต่อผู้ใช้

2. จัดการแก้ปัญหาข้อผิดพลาดที่เกิดขึ้นระหว่างการสื่อสาร

3. ทำการขยายข้อมูล (decompress) เพลงที่ส่งมา

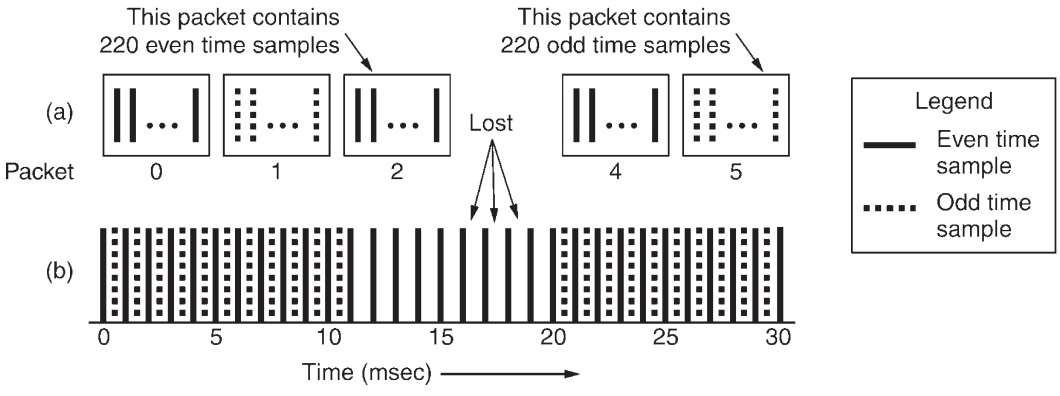
4. กำจัดอาการไม่ต่อเนื่องหรือ "กระตุก (jitter)" ออกไป

โปรแกรม media player ส่วนใหญ่มีส่วนติดต่อผู้ใช้ที่น่าตื่นตาตื่นใจ บางครั้งก็เป็นภาพจำลองของวิทยุสเตอรีโอที่มีปุ่มกด ปุ่มหมุน ปุ่มเลื่อน และการแสดงข้อมูลประกอบการเล่นเพลง โดยทั่วไปมักจะเป็นหน้าตาของวิทยุที่สามารถเปลี่ยนแปลงได้เรียกว่า "skin" โปรแกรมนี้จะทำหน้าที่เป็นตัวติดต่อกับผู้ใช้โดยตรง

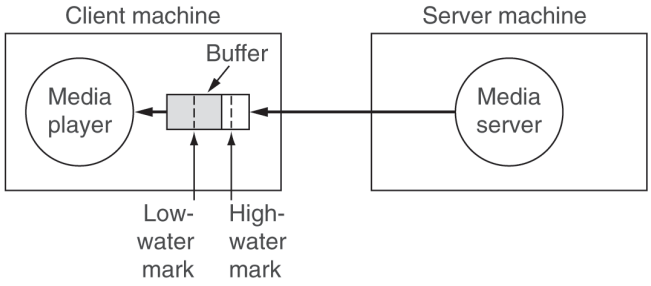
งานอย่างที่สองที่ต้องทำก็คือการแก้ปัญหาข้อผิดพลาดที่เกิดขึ้นในระหว่างการสื่อสาร การส่งข้อมูลเพลงแบบเรียลไทม์ (real time) นี้มักจะไม่ใช้การเชื่อมต่อแบบ TCP เพราะว่าข้อผิดพลาดที่อาจเกิดขึ้นและการจัดส่งแพ็กเก็ตซ้ำจะทำให้เกิดช่องว่างขึ้นในระหว่างการเล่นเพลงซึ่งเป็นสิ่งที่ยอมรับไม่ได้สำหรับผู้ฟังเพลง ดังนั้นจึงเปลี่ยนไปใช้โพรโตคอลอย่าง RTP ซึ่งได้กล่าวถึงไว้แล้วในบทที่ 6 โพรโตคอลนี้ทำงานอยู่บนโพรโตคอล UDP อีกชั้นหนึ่งทำให้แพ็กเก็ตข้อมูลอาจสูญหายในระหว่างการนำส่งได้ และเป็นหน้าที่ของโปรแกรม media player ที่จะจัดการแก้ปัญหานี้

ในบางกรณี เพลงที่จัดส่งมาจะถูกแยกส่งเป็นส่วนๆ สลับกัน (interleave) เพื่อช่วยให้การแก้ปัญหาข้อผิดพลาดทำได้ง่ายยิ่งขึ้น ตัวอย่างเช่น ข้อมูลในแพ็กเก็ตหนึ่งอาจจะมี stereo sample อยู่ 220 ชิ้น แต่ละชิ้นเป็นข้อมูลขนาด 16 บิตคู่หนึ่งซึ่งเป็นข้อมูลที่สามารถเล่นเพลงได้นาน 5 มิลลิวินาที แต่โพรโตคอลอาจทำการส่ง sample ในลำดับคี่มา 10 มิลลิวินาที และตามด้วย sample ลำดับคู่มาอีก 10 มิลลิวินาที ถ้าแพ็กเก็ตใดเกิดสูญหายไปในระหว่างการนำส่ง สิ่งที่ผู้ฟังจะได้รับก็จะไม่ใช่เกิดช่วงเงียบหายไป 5 มิลลิวินาทีแต่จะเป็นช่วงเงียบแทรกสลับกันอยู่ในระหว่าง 10 มิลลิวินาทีแทน วิธีการแก้ปัญหานี้ก็สามารถทำได้โดยให้โปรแกรม media player จัดการเล่นเพลงในช่วงที่มีข้อมูลอยู่ (ซ้ำ) แทนช่วงที่ข้อมูลหายไปแทน หรืออาจนำข้อมูลที่มีอยู่มาคำนวณค่าโดยประมาณของค่าที่หายไปก็ได้ การนำเทคนิค interleaving มาใช้ในการแก้ปัญหาข้อมูลที่สูญหายไปในระหว่างการนำส่งแสดงให้เห็นในรูป 7-60

หน้าที่ประการที่สามของโปรแกรม media player คือการขยายข้อมูล (decompression) เพลง ซึ่งแม้ว่าจะเกิดการคำนวณเป็นอย่างมากแต่ก็เป็นวิธีการที่ไม่ซับซ้อนแต่ประการใด



รูป 7-60
เทคนิค interleaving



รูป 7-61 การจัดเก็บข้อมูลไว้ในบัฟเฟอร์ที่ส่งมาจาก media server และการนำข้อมูลในบัฟเฟอร์ไปใช้

หน้าที่ประการที่สี่คือการกำจัดอาการไม่ต่อเนื่องหรือ "กระตุก (jitter)" ออกไปซึ่งเป็นเสมือนฝันร้ายของระบบเรียลไทม์ ระบบ streaming audio จะเริ่มต้นทำงานด้วยการบันทึกข้อมูลไว้ในบัฟเฟอร์ประมาณ 10-15 วินาทีก่อนที่จะเริ่มเล่นเพลง ดังแสดงในรูป 7-61 ในสภาพที่ควรจะเป็นนั้นเซิบเวอร์จะต้องเติมข้อมูลใหม่ไว้ในบัฟเฟอร์ในอัตราเดียวกันกับที่ข้อมูลในบัฟเฟอร์ถูกนำไปใช้โดยโปรแกรม media player แต่ในสภาพความเป็นจริงมักจะไม่เป็นเช่นนั้น

มีวิธีการสองแนวทางในการที่จะเติมข้อมูลไว้ในบัฟเฟอร์ วิธีแรกเรียกว่า pull server นั่นคือตราบเท่าที่มีเนื้อที่ว่างในบัฟเฟอร์ โปรแกรม media player จะส่งคำร้องขอข้อมูลไปยังเซิบเวอร์เสมอ วิธีนี้มีเป้าหมายที่จะให้บัฟเฟอร์มีข้อมูลเต็มอยู่เสมอ แต่ข้อเสียเปรียบของวิธีนี้ก็คือการที่ต้องส่งคำร้องขอข้อมูลไปยังเซิบเวอร์โดยไม่จำเป็น เพราะเซิบเวอร์ก็ทราบอยู่แล้วว่าจะต้องส่งข้อมูลทั้งไฟล์ไปให้ เหตุใดจึงต้องร้องขออีกเล่า ด้วยเหตุผลนี้ทำให้วิธีการนี้ไม่เป็นที่นิยมใช้

วิธีการที่สองเรียกว่า push server โปรแกรม media player จะส่งคำร้องขอ PLAY ไปยังเซิบเวอร์และเซิบเวอร์ก็จะส่งข้อมูลในไฟล์เพลงนั้นมาให้อย่างสม่ำเสมอ เหตุการณ์ที่อาจเกิดขึ้นได้มีสองประการคือ media server จะทำงานด้วยความเร็วตามปกติ หรือทำงานด้วยความเร็วสูงกว่าปกติ ในทั้งสองกรณีจะมีข้อมูลบางส่วนถูกใส่ไว้ในบัฟเฟอร์ก่อนที่จะถูกนำไปใช้ถ้าเซิบเวอร์ทำงานด้วยความเร็วปกติ ข้อมูลที่ถูกส่งมาจะถูกนำไปต่อท้ายข้อมูลที่ถูกส่งมาก่อนหน้า และ media player ก็จะนำข้อมูลในบัฟเฟอร์ไปใช้ตามลำดับที่ได้รับ ถ้าการทำงานทุกอย่างเป็นไปตามปกติปริมาณข้อมูลที่คงอยู่ในบัฟเฟอร์จะมีค่าเท่ากันอยู่เสมอ วิธีการนี้ทำงานอย่างง่าย ๆ เพราะไม่ต้องมีการส่งข้อมูลมาควบคุมการทำงานแต่อย่างใด

ในอีกวิธีการหนึ่งของ push server นั่นคือจะให้เซิบเวอร์จัดการส่งข้อมูลออกมาให้เร็วกว่าที่ต้องการ ข้อได้เปรียบของวิธีการนี้คือถ้าเซิบเวอร์ไม่สามารถทำงานได้อย่างสม่ำเสมอ ก็มีความเป็นไปได้ว่าเซิบเวอร์จะสามารถทำงานไล่ตามได้ทันภายหลังจากที่อาจจะทำงานช้าลงกว่าปกติหรือหยุดทำงานไปชั่วคราว แต่ปัญหาที่อาจเกิดขึ้นได้ก็คือข้อมูลอาจถูกส่งมาเร็วกว่าอัตราการใช้งานจนไม่มีเนื้อที่เก็บไว้ในบัฟเฟอร์

วิธีการแก้ปัญหานี้ก็คือการกำหนดระดับ low-water mark และ high-water mark ไว้สำหรับบัฟเฟอร์ โดยพื้นฐานแล้วเซิบเวอร์จะจัดส่งข้อมูลมาอย่างรวดเร็วจนกว่าปริมาณข้อมูลในบัฟเฟอร์จะถึงระดับ high-water mark จากนั้นโปรแกรม media player จะส่งข้อความไปบอกให้เซิบเวอร์หยุดทำงานชั่วคราว เนื่องจากข้อมูลจะยังคงถูกส่งมาอย่างรวดเร็วจนกว่าเซิบเวอร์จะได้รับข้อความนี้ ปริมาณข้อมูลในบัฟเฟอร์จาก high-water mark ไปจนถึงบัฟเฟอร์เต็มจะต้องมีค่ามากกว่าระยะเวลาหน่วงที่อาจเกิดขึ้นในการสื่อสารผ่านระบบเครือข่าย ภายหลังจากที่เซิบเวอร์หยุดส่งข้อมูลมาแล้วข้อ

รูป 7-62 คำสั่งของโพรโตคอล RTSP สำหรับ media player ในการควบคุมเซิบเวอร์

Command	Server action
DESCRIBE	List media parameters
SETUP	Establish a logical channel between the player and the server
PLAY	Start sending data to the client
RECORD	Start accepting data from the client
PAUSE	Temporarily stop sending data
TEARDOWN	Release the logical channel

มูลในบัฟเฟอร์ก็จะเริ่มน้อยลง และเมื่อถึงระดับ low-water mark เมื่อใดโปรแกรม media player จะต้องส่งข้อความไปบอกเซิบเวอร์ให้เริ่มส่งข้อมูลมาได้แล้ว ปริมาณข้อมูลที่มีเหลืออยู่ในบัฟเฟอร์ใต้ระดับ low-water mark จะต้องมีมากพอที่จะไม่ทำให้เกิดปัญหาขาดแคลนข้อมูลได้

ในการทำงานกับ push server โปรแกรม media player จะต้องมีความสามารถในการควบคุมจากทางไกลได้ซึ่งเป็นวิธีการที่สนับสนุนโดยโพรโตคอล RTSP ซึ่งได้กำหนดไว้ในมาตรฐาน RFC 2326 และจัดให้มีกลไกสำหรับ media player ในการควบคุมเซิบเวอร์ได้ แต่กลไกนี้ไม่ได้ถูกนำมาใช้กับข้อมูลที่เป็น data stream ซึ่งจะใช้โพรโตคอล RTP ควบคุม คำสั่งหลักๆ ที่ใช้ในการควบคุมแสดงไว้ในรูป 7-62

7.4.4 Internet Radio

การจัดตั้งสถานีวิทยุกระจายเสียงบนระบบอินเตอร์เน็ต (Internet radio) มีแนวทางการทำงานอยู่สองแนวทาง แนวทาวแรกคือโปรแกรมการกระจายเสียงจะถูกบันทึกไว้ล่วงหน้าไว้ในไฟล์ ผู้ฟังสามารถที่จะเชื่อมต่อเข้าไปยังสถานีวิทยุและเลือกโปรแกรมที่ต้องการฟัง ทำการดาวน์โหลดและฟังรายการนั้นได้ที่เครื่องพีซีของตนเอง อันที่จริงแนวทางนี้ก็คือ streaming audio ที่ได้กล่าวถึงไปแล้วนั่นเอง มีความเป็นไปได้ที่จะทำการบันทึกรายการสดที่พึ่งจะถ่ายทอดเสร็จ ทำให้ข้อมูลที่บันทึกอยู่ในดิสก์นั้นอาจจะช้ากว่าเวลากระจายเสียงจริงเพียงครึ่งชั่วโมงหรือน้อยกว่านั้นก็ได้ ข้อได้เปรียบของวิธีการนี้ก็คือเป็นวิธีการที่สามารถจัดทำได้โดยง่าย เทคนิคทุกอย่างที่พูดถึงสามารถนำมาใช้งานในกรณีนี้ได้ และผู้ฟังก็สามารถเลือกโปรแกรมที่ตนเองต้องการฟังได้

อีกแนวทางหนึ่งคือ การกระจายเสียงด้วยวิธีการถ่ายทอดสดผ่านระบบอินเตอร์เน็ต บางสถานีวิทยุสามารถทำการกระจายเสียงออกอากาศตามปกติและกระจายเสียงผ่านระบบอินเตอร์เน็ตได้ในเวลาเดียวกัน แต่ในปัจจุบันมีจำนวนสถานีวิทยุที่กระจายเสียงทางอินเตอร์เน็ตแต่เพียงอย่างเดียวเพิ่มขึ้นเรื่อยๆ เทคนิคบางอย่างที่ใช้ใน streaming audio สามารถนำมาใช้กับการกระจายเสียงได้โดยตรงแต่บางอย่างก็มีเทคนิคที่แตกต่างออกไป

สิ่งหนึ่งที่เหมือนกันก็คือ การใช้บัฟเฟอร์ทางฝั่งผู้รับฟังเพื่อแก้ปัญหาการกระตุกของสัญญาณเสียงที่ได้รับ การเก็บข้อมูลไว้ 10-15 มิลลิวินาทีก่อนที่จะเริ่มส่งกระจายเสียงออกมานั้นจะทำให้ได้คุณภาพเสียงที่ราบเรียบแม้ว่าจะเผชิญกับปัญหาการกระตุกอย่างรุนแรงของสัญญาณบนระบบเครือข่ายก็ตาม ตราบเท่าที่แพ็กเก็ตทั้งหมดเดินทางมาถึงผู้รับทันเวลาก็จะไม่มีผลเสียใดๆ เกิดขึ้น

สิ่งที่แตกต่างกันก็คือ วิธี streaming audio สามารถที่ส่งข้อมูลออกมาด้วยอัตราที่เร็วกว่าอัตรา

การใช้ข้อมูลได้เนื่องจากผู้รับสามารถสั่งให้เซิบเวอร์หยุดส่งข้อมูลออกมาได้เมื่อมีข้อมูลถึงระดับhigh-water mark วิธีการนี้เปิดโอกาสให้มีการส่งแพ็กเก็ตที่เสียหายหรือสูญหายออกมาซ้ำได้แม้ว่าจะเป็นวิธีการที่ไม่นิยมนำมาใช้ก็ตาม ในทางกลับกันการกระจายเสียงวิทยุนั้นจะมีข้อมูลไหลออกมาอย่างสม่ำเสมอตลอดเวลาซึ่งเป็นอัตราเดียวกันกับที่เล่นให้ผู้ฟังได้รับฟัง

ข้อแตกต่างอีกประการหนึ่งก็คือ การกระจายเสียงวิทยุนั้นโดยปกติจะมีผู้ฟังหลายร้อยหรือหลายพันคนพร้อมๆ กันในขณะที่ streaming audio นั้นเป็นการสื่อสารแบบจุดต่อจุดหรือมีผู้ฟังเพียงคนเดียวภายใต้สภาวะแวดล้อมเช่นนี้ Internet Radio สามารถใช้วิธีการกระจายข่าว (broadcasting) โดยใช้โพรโตคอล RTP/RTSP ซึ่งเป็นวิธีการที่มีประสิทธิภาพในการทำงาน

ในปัจจุบัน Internet Radio ไม่ได้ทำงานในลักษณะนี้ สิ่งที่เกิดขึ้นจริงก็คือ ผู้ฟังจะจัดตั้งการเชื่อมต่อ TCP กับสถานีส่งและข้อมูลก็จะถูกส่งออกมาทางช่องสื่อสารนี้ แม้ว่าวิธีการนี้จะสร้างปัญหาขึ้นมามากมาย เช่น การหยุดไหลของข้อมูลเมื่อ window ของการสื่อสารเต็ม การสูญหายของแพ็กเก็ต การส่งแพ็กเก็ตซ้ำ และอื่นๆ

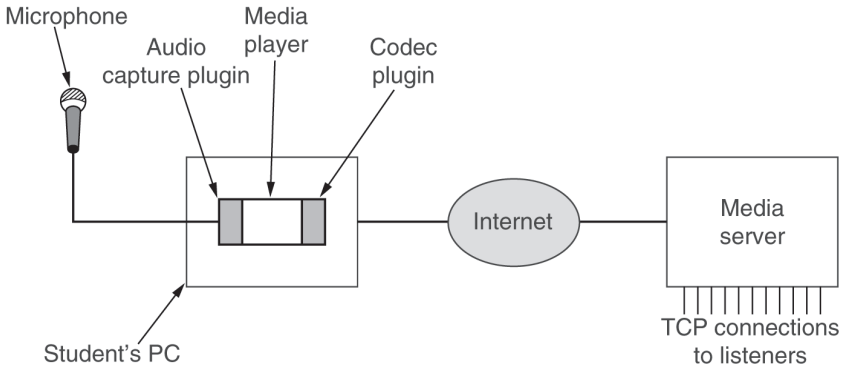
เหตุผลที่ใช้การเชื่อมต่อแบบ TCP แทนที่การกระจายข่าวแบบ RTP นั้นมีอยู่สามประการ ประการแรกมี ISP จำนวนน้อยมากที่สนับสนุนการกระจายข่าว ทำให้วิธีการนี้ไม่ใช่ทางเลือกที่ดี ประการที่สอง RTP นั้นไม่ค่อยเป็นที่รู้จักนักเมื่อเปรียบเทียบกับ TCP และสถานีวิทยุมักจะมีขนาดเล็กและไม่มีผู้เชี่ยวชาญด้านคอมพิวเตอร์อยู่ด้วย ดังนั้นจึงเป็นการง่ายกว่าที่จะใช้โพรโตคอลที่ตนเองรู้จักเป็นอย่างดีและได้รับการสนับสนุนโดยซอฟต์แวร์ทั้งหลาย ประการที่สาม ผู้คนจำนวนมากรับฟังสถานีวิทยุอินเตอร์เน็ตจากที่ทำงาน ซึ่งหมายถึงว่าเป็นสถานที่ที่อยู่ภายใต้การควบคุมโดยไฟร์วอล ระบบไฟร์วอลส่วนใหญ่จะถูกกำหนดให้ปกป้องผู้ใช้ภายในองค์กรจากข้อมูลที่ไม่ทราบที่มาอย่างชัดเจน โดยปกติจะอนุญาตให้มีการติดต่อผ่านพอร์ต 25 (SMTP สำหรับอีเมลล์) UDP แพ็กเก็ตจากพอร์ต 53 (DNS) และการเชื่อมต่อ TCP ผ่านพอร์ต 80 (HTTP สำหรับเว็บ) นอกเหนือจากนี้ (รวมทั้ง RTP) มักจะถูกบล็อคไม่ให้เข้ามาภายในองค์กร ดังนั้นวิธีการเดียวที่จะสามารถรับฟังวิทยุอินเตอร์เน็ตผ่านไฟร์วอลได้ก็คือการให้เว็บไซต์สถานีวิทยุนั้นทำตัวเสมือน HTTP เซิบเวอร์ทั่วไป (หรืออย่างน้อยที่สุดก็ให้ไฟร์วอลมองว่าเป็น HTTP เซิบเวอร์) และจะต้องเป็น HTTP เซิบเวอร์ที่ใช้โพรโตคอล TCP ด้วย

เนื่องจากสถานีวิทยุอินเตอร์เน็ตเป็นสิ่งแปลกใหม่จึงทำให้มีการต่อสู้กันในเรื่องรูปแบบที่จะนำมาใช้เป็นอย่างมาก RealAudio, Windows Media Audio และ MP3 ทำการแข่งขันกันอย่างดุเดือดเพื่อที่จะมาเป็นผู้นำในตลาดนี้ ผู้ที่พึ่งเข้ามาร่วมวงการต่อสู้ด้วยรายใหม่คือ Vorbis ซึ่งมีเทคนิคการทำงานที่คล้ายคลึงกับ MP3 แต่เป็นระบบเปิดและมีความแตกต่างกันพอสมควร

สถานีวิทยุอินเตอร์เน็ตโดยทั่วไปจะมีเว็บเพจที่แสดงผังรายการวิทยุที่จะส่งออกมา ข้อมูลเกี่ยวกับผู้จัดรายการวิทยุ (DJ) ประกาศต่างๆ และโฆษณา นอกจากนี้อาจมีไอคอนให้เลือกรูปแบบสัญญาณที่จะฟัง ไอคอนเหล่านี้นจะเชื่อมโยงเข้ากับ metafile ที่เหมาะสม

เมื่อผู้ฟังเลือกไอคอนที่ต้องการ metafile ขนาดเล็กจะถูกส่งไปที่เซิบเวอร์ บราวเซอร์จะใช้ MIME type หรือนามสกุลของไฟล์ในการเลือก helper application ที่เหมาะสมสำหรับ metafile นั้นๆ จากนั้นก็จะบันทึก metafile ลงในไฟล์ชั่วคราว เรียกโปรแกรม media player ขึ้นมา และส่งไฟล์

รูป 7-63
สถานีวิทยุอินเตอร์เน็ต
สำหรับนักศึกษา



ชั่วคราวนั้นไปให้ โปรแกรม media player จะอ่านข้อมูลในไฟล์ชั่วคราวซึ่งก็จะเห็น URL ของสถานีวิทยุ (มักจะใช้โพรโตคอล HTTP มากกว่า RTSP เพื่อให้สามารถเล่นผ่านไฟร์วอลได้) ทำการติดต่อกับเซิบเวอร์ และเริ่มทำหน้าที่เหมือนกับวิทยุเครื่องหนึ่ง

อีกสิ่งหนึ่งที่น่าสนใจสำหรับสถานีวิทยุอินเตอร์เน็ตก็คือการจัดการที่ง่ายทำให้ผู้ใดก็ได้โดยเฉพาะนักศึกษาสามารถจัดตั้งและบริหารสถานีวิทยุนี้ได้ด้วยตนเอง รูป 7-63 แสดงอุปกรณ์หลักที่ใช้ในสถานีวิทยุอินเตอร์เน็ต อุปกรณ์หลักชิ้นแรกก็คือเครื่องพีซีเครื่องหนึ่งที่มีการ์ดเสียงและไมโครโฟน ซอฟต์แวร์ประกอบด้วยโปรแกรม media player เช่น Winamp หรือ Freeamp ที่มี plug-in เป็นซอฟต์แวร์สำหรับการรับสัญญาณจากไมโครโฟนและอุปกรณ์ codec สำหรับเลือกรูปแบบของสัญญาณเสียง เช่น MP3 หรือ Vorbis

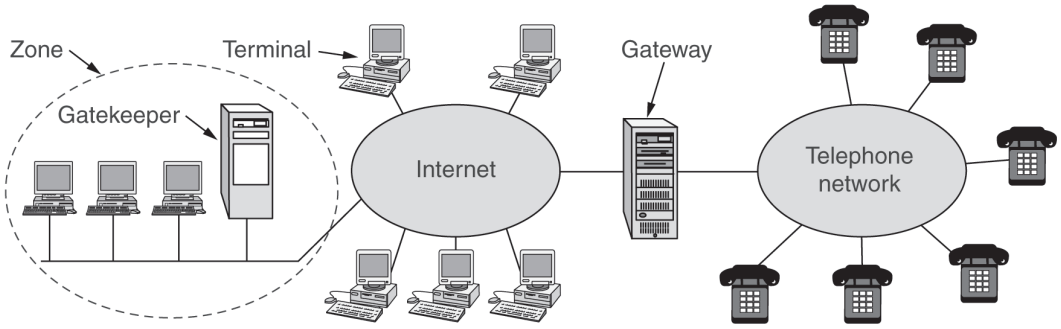
สัญญาณ audio stream ที่สร้างขึ้นมาจากสถานีวิทยุนี้ก็จะถูกถ่ายทอดเข้าไปในระบบอินเตอร์เน็ตไปยังเซิบเวอร์ขนาดใหญ่ซึ่งมีความสามารถในการกระจายข้อมูลไปยัง TCP connection จำนวนมากได้ เซิบเวอร์มักจะสนับสนุนสถานีวิทยุขนาดเล็กจำนวนมากได้ และมักจะมีรายการบันทึกว่ามีสถานีวิทยุอะไรบ้างและสถานีใดกำลังออกอากาศ ผู้ฟังก็สามารถเข้าไปที่เซิบเวอร์ เลือกสถานีที่ต้องการ และรับฟังการกระจายเสียงของสถานีนั้น

7.4.5 Voice over IP

ในสมัยก่อนระบบโทรศัพท์ถูกนำมาใช้เป็นสื่อหลักในการถ่ายทอดข้อมูลเสียง (voice) โดยมีปริมาณข้อมูลดิจิตอล (data) เป็นส่วนประกอบเพียงเล็กน้อย แต่ข้อมูลดิจิตอลก็ได้เริ่มเติบโตขึ้นอย่างรวดเร็ว ในราวปี ค.ศ. 1999 ปริมาณบิตข้อมูลที่ถูกส่งผ่านระบบเครือข่ายนั้นมีมากพอ ๆ กับปริมาณบิตเสียงสนทนา และในปี ค.ศ. 2002 ปริมาณบิตข้อมูลมีมากกว่าปริมาณบิตเสียงหลายเท่าตัวและยังมีอัตราการเติบโตอย่างรวดเร็วมาก ในขณะที่อัตราการเติบโตของบิตเสียงนั้นเติบโตช้ามาก (เพียงประมาณปีละ 5%) ผลที่ตามมาของตัวเลขข้อเท็จจริงนี้ทำให้ระบบเครือข่ายข้อมูล (packet-switched network) เริ่มที่จะสนใจในการนำบิตเสียงเข้ามารวมกับระบบเครือข่ายข้อมูลดิจิตอล ซึ่งเป็นที่มาของระบบ Internet Telephony หรือ voice over IP

H.323

สิ่งหนึ่งที่ชัดเจนสำหรับทุกคนตั้งแต่เริ่มต้นคือถ้าบริษัทใดก็ตามพยายามที่จะสร้างโพรโตคอลแสต็กของตนเองขึ้นมาใช้งาน ระบบนั้นจะไม่มีทางสำเร็จได้ เพื่อหลีกเลี่ยงปัญหานี้ บริษัทและองค์กร



รูป 7-64
สถาปัตยกรรม H.323
สำหรับ Internet
Telephony

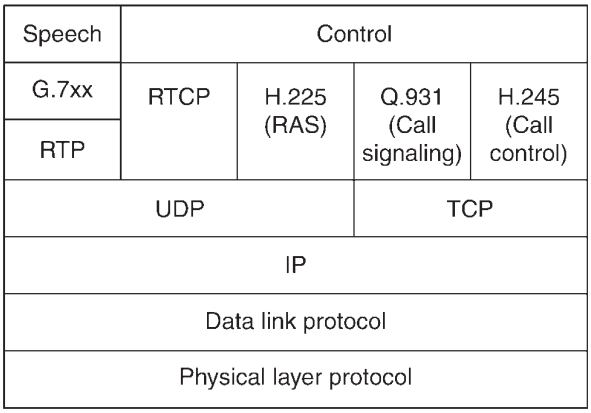
จำนวนมากจึงได้รวมตัวกันภายใต้การควบคุมขององค์กร ITU เพื่อที่จะคิดค้นมาตรฐานขึ้นมาใช้งาน ในปี ค.ศ. 1996 องค์กร ITU ได้ออกมาตรฐานแรกออกมาเรียกว่า H.323 หรือ "Visual Telephony System and Equipment for Local Area Networks Which Provide a Non-Guaranteed Quality of Service" มาตรฐานนี้ได้รับการแก้ไขในปี ค.ศ. 1998 ซึ่งได้กลายมาเป็นมาตรฐานที่ได้รับความนิยมอย่างกว้างขวาง ในระบบ Internet Telephony ทั่วไป

H.323 เป็นการกล่าวถึงสถาปัตยกรรมของระบบ Internet Telephony มากกว่าที่จะเป็นการกำหนดโพรโตคอลที่ใช้งาน มาตรฐานนี้ได้อ้างอิงถึงโพรโตคอลจำนวนมากสำหรับการเข้ารหัสเสียงพูด การโทรเข้า การส่งสัญญาณ การนำส่งข้อมูล และการทำงานอื่นๆ มากกว่าที่จะทำการกำหนดโพรโตคอลขึ้นมาเอง รูปแบบทั่วไปของ Internet Telephony แสดงให้เห็นในรูป 7-64 ที่ศูนย์กลางของระบบคือเกตเวย์ที่ทำการเชื่อมโยงระบบอินเตอร์เน็ตเข้ากับระบบเครือข่ายโทรศัพท์เกตเวย์จะใช้โพรโตคอล H.323 ทางฝั่งอินเตอร์เน็ตและใช้โพรโตคอล PSTN ทางฝั่งเครือข่ายโทรศัพท์ อุปกรณ์ที่ใช้ในการสื่อสารเรียกว่า เทอร์มินอล (terminals) ระบบเครือข่าย LAN อาจจะมี gatekeeper ซึ่งทำหน้าที่ควบคุมจุดเชื่อมต่อเรียกว่า โซน (Zone)

ระบบเครือข่ายโทรศัพท์ต้องใช้โพรโตคอลควบคุมการทำงานจำนวนหนึ่ง เริ่มต้นด้วยโพรโตคอลที่ใช้ในการเข้ารหัสและถอดรหัสเสียงพูด ระบบ PCM ที่กล่าวถึงในบทที่ 2 ได้ถูกกำหนดขึ้นโดยองค์กร ITU มาตรฐาน G.711 ทำหน้าที่ในการเข้ารหัสเสียงจากช่องสื่อสารหนึ่งช่องด้วยการ sampling สัญญาณ 8,000 ครั้งต่อวินาที โดยมีขนาดข้อมูล 8 บิตต่อหนึ่ง sample ทำให้ได้เป็นข้อมูล (ที่ยังไม่ได้บีบอัด) ขนาด 64 kbps ระบบ H.323 ทุกระบบจะต้องสนับสนุน G.711 อย่างไรก็ตาม โพรโตคอลอื่นที่นำมาใช้บีบอัดข้อมูลก็สามารถนำมาใช้งานร่วมกันได้ (ไม่ได้บังคับ) โพรโตคอลที่นำมาใช้เหล่านี้ใช้อัลกอริทึมในการบีบอัดข้อมูลต่างกันไปทำให้เกิดเป็นข้อดีและข้อเสียระหว่างคุณภาพและขนาดช่องสื่อสารที่ต้องการใช้ ตัวอย่างเช่น G.732.1 จะนำบล็อคข้อมูลขนาด 240 samples (30 มิลลิวินาทีเสียง) และใช้ predctive coding ในการลดขนาดลงเหลือเพียง 24 หรือ 20 ไบต์ ทำให้ได้เป็น output ขนาด 6.4 kbps หรือ 5.3 kbps (บีบอัดในอัตราส่วน 10 และ 12 เท่า) โดยมีคุณภาพเสียงลดลงไปเพียงเล็กน้อย

เนื่องจากอาจนำอัลกอริทึมบีบอัดข้อมูลมาใช้ได้หลายแบบ โพรโตคอลจำเป็นจะต้องกำหนดให้เทอร์มินอลสามารถต่อรองอัลกอริทึมที่จะนำมาใช้ได้ โพรโตคอลนี้คือ H.245 ซึ่งทำหน้าที่อื่นด้วยคือทำการต่อรองอัตราการส่งบิตข้อมูล (bit rate) โพรโตคอล RTCP ถูกนำมาใช้ในการควบคุมช่องสื่อสาร RTP นอกจากนี้มาตรฐาน ITU Q.931 เป็นโพรโตคอลที่ถูกนำมาใช้ในการจัดตั้งช่องสื่อสารและยกเลิก

รูป 7-65
H.323
โพรโตคอลแสตัก

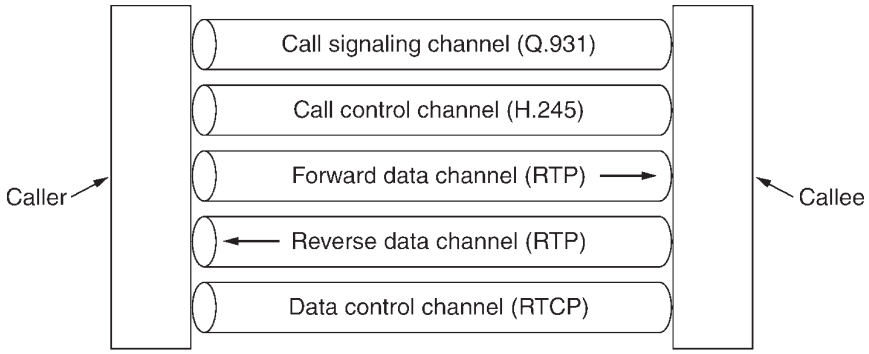


การใช้ช่องสื่อสาร การสนับสนุนเสียง dial tone การทำเสียงเรียกเข้า (ring tone) และส่วนอื่นๆ ของ Telephony ด้วย เทอร์มินอลก็ต้องการโพรโตคอลที่นำมาใช้สื่อสารกับ gatekeeper ซึ่งก็คือ H.225 ช่องสื่อสารระหว่าง พีซีไปยัง gatekeeper เรียกว่า RAS (Registration/Administration/Status) ช่องสื่อสารนี้ช่วยให้เทอร์มินอลสามารถเข้าร่วมหรือออกจากโซน ร้องขอและได้รับขนาดความกว้างของช่องสื่อสารที่ต้องการ และนำเสนอข้อมูลแสดงสถานะการทำงาน ท้ายที่สุดต้องมีโพรโตคอลสำหรับการถ่ายทอดข้อมูลจริงซึ่งก็คือ RTP และได้รับการควบคุมโดยโพรโตคอล RTCP ตำแหน่งของโพรโตคอลที่กล่าวมานี้แสดงในรูป 7-65

เพื่อให้เห็นว่าโพรโตคอลต่างๆ มีการประสานงานกันอย่างไรให้สมมุติเหตุการณ์ดังนี้ พีซีเทอร์มินอลเครื่องหนึ่งอยู่ในระบบ LAN (ที่มี gatekeeper อยู่ด้วย) ทำการเรียกไปยังโทรศัพท์เครื่องหนึ่ง เครื่องพีซีจะเริ่มด้วยการค้นหา gatekeeper จึงทำการกระจายข่าวแพ็กเก็ต UDP gatekeeper discovery packet ออกไปยังพอร์ตหมายเลข 1718 เมื่อ gatekeeper ตอบสนองกลับมา เครื่องพีซีก็จะทราบหมายเลข IP ของ gatekeeper ต่อไปพีซีจะทำการลงทะเบียนกับ gatekeeper ด้วยการส่ง RAS message เข้าไปใน UDP แพ็กเก็ต เมื่อแพ็กเก็ตนี้ได้รับการยอมรับแล้ว พีซีจะส่ง RAS admission message ไปยัง gatekeeper เพื่อขอใช้ช่องสื่อสารในขนาดที่ต้องการ เมื่อได้รับการตอบสนองแล้วการโทรเรียกจึงจะเริ่มต้นขึ้นได้ แนวความคิดในการร้องขอขนาดของช่องสื่อสารล่วงหน้าก่อนการใช้งานนั้นก็เพื่อให้ gatekeeper สามารถจำกัดจำนวนการโทรในแต่ละช่วงเวลาเพื่อไม่ให้เกิดปัญหามีผู้ใช้มากเกินกว่าที่สายสื่อสารจะรองรับได้และจะได้สามารถรับประกันคุณภาพการให้บริการได้ด้วย

ต่อไปพีซีจะจัดตั้งการเชื่อมต่อ TCP เข้ากับ gatekeeper เพื่อเริ่มต้นขั้นตอนการโทร (call setup) การจัดตั้ง call setup นี้จะใช้โพรโตคอลที่มีอยู่แล้วของระบบเครือข่ายโทรศัพท์ซึ่งเป็นการเชื่อมต่อแบบ connection-oriented จึงจำเป็นต้องใช้โพรโตคอล TCP ในทางกลับกันระบบโทรศัพท์ไม่มีโพรโตคอลที่ทำงานเหมือน RAS ในการประกาศความมีตัวตนของโทรศัพท์ใดๆ ดังนั้นผู้ออกแบบ H.323 จึงมีอิสระในการเลือกใช้ UDP หรือ TCP ก็ได้สำหรับ RAS ซึ่งในที่นี้ได้เลือกโพรโตคอลที่มีค่าดำเนินการต่ำกว่านั่นคือ UDP

เมื่อได้รับการจองขนาดของช่องสื่อสารแล้ว พีซีก็จะเริ่มส่ง Q.931 SETUP message ผ่านทางช่องสื่อสาร TCP ข้อความนี้จะบอกให้ทราบจำนวนโทรศัพท์ที่กำลังถูกเรียก (หรือหมายเลข IP และหมายเลข



รูป 7-66
ช่องทางสื่อสารที่เกิดขึ้นระหว่างผู้โทรศัพท์ทั้งสองฝ่าย

พอร์ต ในกรณีที่เครื่องที่ถูกเรียกเป็นเครื่องพีซีเหมือนกัน) gatekeeper ตอบสนองด้วย Q.931 CALL PROCEEDING message เพื่อเป็นการตอบรับ จากนั้น gatekeeper จะส่งข้อความ SETUP message นี้ต่อไปยังเกตเวย์

ที่เกตเวย์ซึ่งเป็นทั้งเครื่องคอมพิวเตอร์และ telephone switch ก็จะทำ phone call ตามปกติไปยังโทรศัพท์เป้าหมาย เครื่อง end office ที่เครื่องโทรศัพท์เป้าหมายต่ออยู่จะส่งสัญญาณเรียก และจัดการส่ง Q.931 ALERT message เพื่อบอกให้เครื่องพีซีที่เป็นผู้โทรเข้านั้นได้ทราบว่าได้จัดการส่งสัญญาณเรียกไปยังโทรศัพท์เป้าหมายนั้นแล้ว เมื่อคนที่อยู่ปลายทางยกหูโทรศัพท์ขึ้นมา เครื่อง end office จะจัดส่ง Q.931 CONNECT message มายังเครื่องพีซีที่ทำการเรียกเพื่อบอกให้ทราบว่าการเชื่อมต่อได้เกิดขึ้นแล้ว

เมื่อการเชื่อมต่อเกิดขึ้น gatekeeper จะหมดหน้าที่แต่เกตเวย์จะยังคงทำหน้าที่ต่อไป แพ็กเก็ตที่เกิดขึ้นนับจากนี้จะถูกส่งข้าม gatekeeper ไปยังเกตเวย์โดยตรง (ตามหมายเลข IP ของเกตเวย์) ที่จุดนี้เสมือนหนึ่งว่าได้จัดให้มีช่องทางสื่อสารโดยตรงระหว่างเครื่องพีซีกับเครื่องโทรศัพท์ที่ถูกเรียกขึ้นมาแล้ว ซึ่งเป็นเพียงการเชื่อมต่อในระดับชั้นการสื่อสารกายภาพสำหรับการเคลื่อนย้ายบิตข้อมูลโดยตรงเท่านั้น แต่ละฝ่ายจะไม่ทราบอะไรเลยเกี่ยวกับข้อมูลการทำงานของอีกฝ่ายหนึ่ง

ต่อมาโพรโตคอล H.245 ถูกนำมาใช้ในการต่อรองค่าพารามิเตอร์สำหรับการโทร ข้อมูลจะถูกส่งผ่าน H.245 control channel ซึ่งจะเปิดอยู่เสมอ แต่ละฝ่ายเริ่มต้นด้วยการประกาศขีดความสามารถของตนเอง เช่น สามารถจัดการข้อมูลวิดีทัศน์ได้หรือไม่ หรีการทำการประชุมทางอิเล็กทรอนิกส์ หรือใช้ codec แบบไหนเป็นต้น เมื่อต่างฝ่ายต่างก็ทราบขีดความสามารถของกันและกันแล้วจะทำการจัดตั้งช่องทางสื่อสารทางเดียว (unidirectional data channel) ขึ้นมาสองช่องทาง (ไปทางหนึ่งกลับทางหนึ่ง) โดยใส่ค่าพารามิเตอร์ที่เหมาะสมเข้าไปในแต่ละช่องทางสื่อสาร เนื่องจากแต่ละฝ่ายอาจมีอุปกรณ์ที่แตกต่างกันดังนั้นจึงมีความเป็นไปได้ที่ codec ทางช่องทางหนึ่งจะแตกต่างไปจากของอีกช่องทางหนึ่ง ภายหลังจากที่ได้จัดตั้งช่องสื่อสารเสร็จแล้วก็จะสามารถส่งข้อมูลได้โดยใช้โพรโตคอล RTP ซึ่งจัดการควบคุมโดย RTCP ทำหน้าที่ในการควบคุมความคับคั่งของข้อมูล ถ้ามีการส่งข้อมูลวิดีทัศน์ RTCP จะจัดการในเรื่องการเทียบจังหวะสัญญาณของ audio และ video รูป 7-66 แสดงช่องทางสื่อสารต่างๆ ที่เกิดขึ้น เมื่อฝ่ายหนึ่งฝ่ายใดวางหูโทรศัพท์ ช่องทางสื่อสร Q.931 จะถูกใช้ในการยกเลิกการเชื่อมต่อนี้

เมื่อการโทรสิ้นสุดลง เครื่องพีซีที่เป็นฝ่ายเรียกจะติดต่อกับ gatekeeper อีกครั้งหนึ่งด้วย RAS

message เพื่อบอกเลิกการขอใช้ช่องสัญญาณ หรืออาจจะเป็นการโทรครั้งใหม่ก็ได้

SIP-The Session Initiation Protocol

H.323 ได้รับการออกแบบพัฒนาโดยองค์กร ITU ซึ่งคนส่วนใหญ่ในวงการอินเตอร์เน็ตเห็นว่าเป็นโพรโตคอลที่มีขนาดใหญ่ มีความซับซ้อน และไม่มีความอ่อนตัว องค์กร IETF จึงได้กำหนดคณะทำงานขึ้นมาเพื่อออกแบบโพรโตคอลที่ง่ายกว่าและมีความเป็นโมดูลมากกว่าสำหรับการทำ voice over IP ผลที่ได้คือโพรโตคอล SIP (Session Initiation Protocol) ซึ่งได้กำหนดมาตรฐานไว้ใน RFC 3261 โพรโตคอลนี้อธิบายวิธีการจัดตั้ง Internet Telephone การประชุมทางอิเล็กทรอนิกส์ และการเชื่อมต่อมัลติมีเดียแบบอื่นๆ โพรโตคอล SIP แตกต่างไปจากโพรโตคอล H.323 ที่เป็นชุดของโพรโตคอลที่มีครบทุกหน้าที่ โดยที่ SIP เป็นเพียงโมดูลเดียวที่ได้รับการออกแบบมาให้สามารถทำงานร่วมกับโปรแกรมประยุกต์บนอินเตอร์เน็ตที่มีอยู่แล้วได้ ตัวอย่างเช่น ได้กำหนดให้หมายเลขโทรศัพท์มีรูปแบบในลักษณะเดียวกับ URL ทำให้สามารถบรรจุหมายเลขโทรศัพท์ไว้ในเว็บเพจได้ และยอมให้ผู้ใช้คลิ๊กบน hypertext เพื่อใช้เป็นการหมุนเลขโทรศัพท์ได้ (วิธีการเดียวกันกับที่ใช้ใน mailto ซึ่งเมื่อผู้ใช้คลิ๊กบนข้อความเชื่อมโยง hyperlink จะเป็นการเรียกโปรแกรมขึ้นมาเพื่อส่งข้อความอีเมลล์)

SIP สามารถจัดตั้ง two-party session (ที่เป็นการเชื่อมต่อโทรศัพท์ธรรมดา) ให้แก่ผู้ใช้คู่หนึ่งได้ หรือเป็น multi-party session ซึ่งทุกคนที่ร่วมอยู่ใน session นั้นจะสามารถได้ยินและพูดคุยกับผู้อื่นได้ และเป็น multicast session ซึ่งมีผู้พูดเพียงคนเดียวแต่มีผู้ฟังได้หลายคน ภายในแต่ละ session สามารถที่จะส่ง เสียงพูด วีดิทัศน์ หรือข้อมูลได้ (ซึ่งมีประโยชน์มากสำหรับการเล่นเกมส์เรียลไทม์ที่มีผู้ร่วมเล่นหลายคน) โพรโตคอล SIP จะจัดการในเรื่องการจัดตั้ง การบริหาร และการยกเลิก session โพรโตคอลอื่น เช่น RTP/RTCP จะถูกนำมาใช้ในการถ่ายโอนข้อมูล SIP เป็นโพรโตคอลในชั้นสื่อสารโปรแกรมประยุกต์ที่สามารถทำงานบนโพรโตคอล UDP หรือ TCP ก็ได้

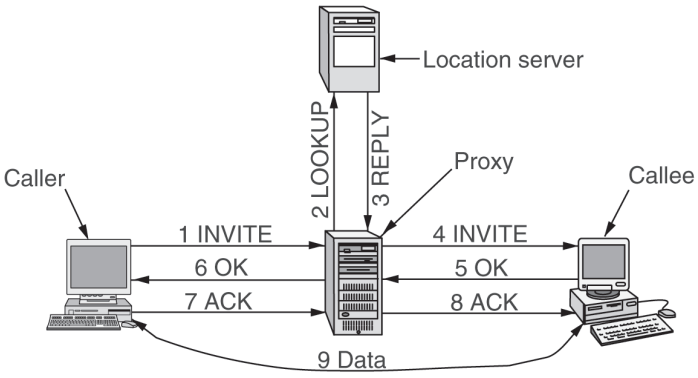
SIP สนับสนุนการให้บริการหลายอย่าง เช่น การค้นหาผู้โทรเข้า ตรวจหาขีดความสามารถของผู้ที่โทรเข้า และจัดการกลไกในการจัดตั้งการสื่อสารและยกเลิกการสื่อสาร ในกรณีที่ง่ายที่สุด SIP จะจัดตั้ง session ขึ้นมาระหว่างเครื่องคอมพิวเตอร์ของผู้โทรเข้าและเครื่องคอมพิวเตอร์ของผู้ถูกเรียก

หมายเลขโทรศัพท์ใน SIP จะอยู่ในรูปแบบ URL โดยใช้รูปแบบ "sip" เช่น sip:ilse@cs.university.edu สำหรับผู้ใช้ที่ชื่อ "Ilse" อยู่ที่โฮสที่ใช้ชื่อ DNS ว่า "cs.university.edu" URL ที่ใช้กับ SIP สามารถที่จะใช้ที่อยู่แบบ IPv4, IPv6, หรือ หมายเลขโทรศัพท์จริงๆ ก็ได้

โพรโตคอล SIP เป็นโพรโตคอลที่มีรูปแบบเป็น text-based บนโพรโตคอล HTTP ผู้ใช้คนหนึ่งจะ

Method	Description
INVITE	Request initiation of a session
ACK	Confirm that a session has been initiated
BYE	Request termination of a session
OPTIONS	Query a host about its capabilities
CANCEL	Cancel a pending request
REGISTER	Inform a redirection server about the user's current location

รูป 7-67 Method ของโพรโตคอล SIP



รูป 7-68
Use a proxy and redirection servers with SIP

ส่งข้อความมาในรูปแบบ ASCII text ประกอบด้วยชื่อ method name ในบรรทัดแรกตามด้วยบรรทัดต่อมาซึ่งคือ header สำหรับการส่งค่าพารามิเตอร์ header ส่วนใหญ่นำมาจาก MIME ช่วยให้ SIP สามารถทำงานร่วมกับโปรแกรมประยุกต์บนอินเตอร์เน็ตได้ รูป 7-67 แสดงชื่อ method จำนวน 6 ชนิดหลักที่กำหนดไว้ในมาตรฐานของโพรโตคอลนี้

ในการจัดตั้ง session ผู้เรียกจะทำการจัดตั้งการเชื่อมต่อ TCP ไปยังผู้รับและส่ง INVITE message ไปให้ หรือจัดการส่งข้อความนี้ผ่านทางแพ็กเก็ต UDP ก็ได้ ในทั้งสองกรณี header ที่อยู่ในบรรทัดที่สองและบรรทัดต่อมาจะอธิบายโครงสร้างของ message body ซึ่งจะบรรจุขีดความสามารถของผู้เรียก ชนิดของ media ที่ใช้ และรูปแบบเอาไว้ ถ้าผู้ถูกเรียกยินยอมรับการเชื่อมต่อนี้ก็จะตอบสนองด้วยโค้ดตอบสนองแบบ HTTP-type (ตัวเลข 3 ตัวที่จัดแบ่งกลุ่มไว้ในรูปที่ 7-42 เช่น 200 หมายถึงการยอมรับการเรียกเข้านี้) ตามมาด้วยข้อมูลที่บอกถึงขีดความสามารถ ชนิดของ media และรูปแบบที่ใช้

การเชื่อมต่อกระทำโดยใช้โพรโตคอลชนิด three-way handshake ดังนั้นผู้โทรจะตอบสนองการส่งข้อมูลของผู้ถูกเรียกด้วยการส่ง ACK message ซึ่งเป็นการสิ้นสุดการทำงานของโพรโตคอล และเป็นการยืนยันว่าผู้เรียกได้รับ message code 200 ของผู้ถูกเรียกแล้ว

แต่ละฝ่ายสามารถขอยกเลิก session ได้ด้วยการส่งข้อความที่มี BYE method ไปยังอีกฝ่ายหนึ่ง เมื่อฝ่ายตรงข้ามตอบรับ session นั้นก็ถือว่าสิ้นสุดลงแล้ว

Method OPTIONS ใช้ในการถามขีดความสามารถของเครื่องคอมพิวเตอร์ที่ตนเองกำลังใช้งานอยู่ method นี้ถูกนำมาใช้ก่อนที่จะเริ่มต้นการจัดตั้ง session เพื่อให้ทราบขีดความสามารถของตนเอง

Method REGISTER มีความเกี่ยวข้องกับความสามารถของ SIP ในการติดตามและเชื่อมต่อกับผู้ใช้ที่อยู่ไกลจากบ้านของตนเอง ข้อความนี้จะถูกส่งไปยัง SIP location server ซึ่งจะคอยติดตามดูว่าใครอยู่ ณ ที่ใด ผู้ใช้สามารถสอบถามไปยังเซิบเวอร์นี้เพื่อจะได้ทราบที่อยู่ของคนที่ต้องการติดต่อด้วย รูป 7-68 แสดงกระบวนการเปลี่ยนแปลงที่อยู่ของบุคคลที่กำลังติดตามหาตัวอยู่ ในที่นี้ผู้เรียกจะส่ง INVITE message ไปยัง proxy server ซึ่งจะทำการค้นหาตำแหน่งของผู้ที่ต้องการติดต่อด้วยจาก SIP location server แล้วจึงส่ง INVITE message ไปยังที่นั้น ข้อความ LOOKUP และ REPLY message ไม่ใช่ส่วนหนึ่งของ SIP ผู้ใช้สามารถเลือกใช้โพรโตคอลใดที่เหมาะสมก็ได้ในการติดต่อกับ SIP location server

การเปรียบเทียบ H.323 กับ SIP

โพรโตคอล H.323 และ SIP มีความคล้ายกันหลายประการและก็มีส่วนที่แตกต่างกันอยู่ด้วย ทั้ง

รูป 7-69
การเปรียบเทียบ
H.323 และ SIP

Item	H.323	SIP
Designed by	ITU	IETF
Compatibility with PSTN	Yes	Largely
Compatibility with Internet	No	Yes
Architecture	Monolithic	Modular
Completeness	Full protocol stack	SIP just handles setup
Parameter negotiation	Yes	Yes
Call signaling	Q.931 over TCP	SIP over TCP or UDP
Message format	Binary	ASCII
Media transport	RTP/RTCP	RTP/RTCP
Multiparty calls	Yes	Yes
Multimedia conferences	Yes	No
Addressing	Host or telephone number	URL
Call termination	Explicit or TCP release	Explicit or timeout
Instant messaging	No	Yes
Encryption	Yes	Yes
Size of standards	1400 pages	250 pages
Implementation	Large and complex	Moderate
Status	Widely deployed	Up and coming

สองโพรโตคอลยอมให้มีการจัดตั้งการสื่อสารแบบ two-party call และ multiparty call ซึ่งจุดที่ติดต่อด้วยนั้นอาจเป็นโทรศัพท์หรือเครื่องคอมพิวเตอร์ก็ได้ ทั้งสองแบบสนับสนุนการต่อรองค่าพารามิเตอร์สำหรับการสื่อสาร การเข้ารหัสข้อมูล และสนับสนุนโพรโตคอล RTP/RTCP รูป 7-69 แสดงข้อมูลสรุปความแตกต่างและความเหมือนกันของทั้งสองโพรโตคอลนี้

แม้ว่าขีดความสามารถจะคล้ายคลึงกันแต่ทั้งสองโพรโตคอลมีแนวคิดที่แตกต่างกันมาก H.323 เปรียบได้กับนักมวยรุ่นหนัก มาตรฐานที่กำหนดโดยอุตสาหกรรมโทรศัพท์ กำหนดโพรโตคอลแสต็กที่สมบูรณ์แบบ และกำหนดไว้อย่างชัดเจนว่าอะไรบ้างที่อนุญาตและอะไรบ้างที่เป็นข้อห้าม แนวทางนี้จึงนำไปสู่การกำหนด โพรโตคอลที่มีโครงสร้างอย่างดีในแต่ละชั้นสื่อสาร ทำให้การทำงานร่วมกับโพรโตคอลอื่นทำได้โดยง่าย สิ่งที่ต้องแลกมาก็คือเป็นโพรโตคอลขนาดใหญ่ มีความซับซ้อนมาก และมาตรฐานที่เข้มแข็งซึ่งยากต่อการที่จะนำไปปรับใช้กับโปรแกรมประยุกต์อื่นในอนาคต

ในทางกลับกัน SIP เป็นโพรโตคอลบนอินเตอร์เน็ตอย่างหนึ่งที่สามารถทำงานร่วมกันด้วยการแลกเปลี่ยนข้อความ ASCII text สั้นๆ เพียงไม่กี่บรรทัด เป็นโมดูลขนาดเล็กที่สามารถทำงานร่วมกับโพรโตคอลอื่นบนอินเตอร์เน็ตได้เป็นอย่างดี แต่ไม่สามารถทำงานร่วมกับระบบโทรศัพท์ในปัจจุบันได้ดีนัก เนื่องจากโมดูลของ voice over IP นี้มีความเป็นโมดูลอยู่มาก ดังนั้นจึงมีความอ่อนตัวและสามารถปรับตัวให้เข้ากับโปรแกรมประยุกต์ใหม่ๆ ได้เป็นอย่างดี แต่ก็มีข้อเสียที่อาจจะมีปัญหาในการทำงานร่วมกันได้

บทสรุปท้ายบท

การตั้งชื่อบนระบบอินเตอร์เน็ตใช้โครงสร้างตามลำดับขั้นเรียกว่า Domain Name System (DNS) ในระดับขั้นสูงสุดจะเป็นชื่อที่รู้จักกันทั่วไป เช่น com, edu และชื่อประเทศต่างๆ ทั่วโลกอีกกว่า 200 ชื่อ DNS ถูกสร้างขึ้นมาให้เป็นลักษณะฐานข้อมูลแบบกระจายโดยมีเซิบเวอร์ตั้งอยู่ทั่วโลก DNS จะบันทึกระเบียนข้อมูลที่ประกอบด้วยหมายเลข IP, mail exchange และข้อมูลสำคัญอื่นๆ เมื่อผู้ใช้ส่งคำถามมายัง DNS เซิบเวอร์จะเกิดกระบวนการค้นหาชื่อโดเมนและหมายเลข IP ของโดเมนนั้นซึ่งจะถูกนำไปใช้ในการติดต่อกับโดเมนโดยตรงในภายหลัง

อีเมลล์เป็นหนึ่งในงานประยุกต์ที่ได้รับความนิยมสูงสุดบนอินเตอร์เน็ต ทุกคนตั้งแต่เด็กเล็กไปจนถึงผู้ใหญ่ต่างก็ใช้อีเมลล์กันทั้งนั้น ระบบอีเมลล์ส่วนใหญ่ในโลกจะใช้ระบบเมลล์มาตรฐาน RFC 2821 และ 2822 ข้อความที่ถูกส่งในระบบนี้จะใช้ header ที่เป็นตัวอักษร ASCII ธรรมดาในการกำหนดคุณสมบัติของข้อความที่นำส่ง รายละเอียดภายในอาจกำหนดได้โดยใช้ MIME ข้อความจะถูกส่งโดยโพรโตคอล SMTP ซึ่งทำงานโดยการจัดตั้งการเชื่อมต่อ TCP จากโฮสผู้ส่งไปยังโฮสเป้าหมายและจัดการส่งอีเมลล์นั้นผ่านการเชื่อมต่อ TCP ที่จัดตั้งขึ้นนี้

งานประยุกต์อีกอย่างหนึ่งบนระบบอินเตอร์เน็ตที่ได้รับความนิยมไม่แพ้กันก็คือ World Wide Web หรือเรียกสั้นๆ ว่า "เว็บ" เว็บเป็นระบบที่ใช้ในการเชื่อมโยงเอกสารที่มี hypertext เป็นส่วนประกอบแต่เดิมมา เอกสารแต่ละชิ้นเป็นเว็บเพจที่เขียนขึ้นมาด้วยภาษา HTML โดยมี hyperlink เชื่อมโยงไปยังเว็บเพจอื่นๆ ในปัจจุบัน ภาษา XML กำลังเข้ามาแทนที่ภาษา HTML และข้อมูลที่อยู่ภายในเว็บเพจก็กลายเป็นข้อมูลพลวัตที่สามารถดัดแปลงให้เหมาะกับผู้ใช้แต่ละคนได้ โดยใช้ server-side script เช่น PHP, JSP และ ASP รวมทั้ง cilent-side script เช่น javaScript เป็นต้น โปรแกรมบราวเซอร์สามารถนำมาใช้แสดงเอกสารบนเว็บโดยการจัดตั้งการเชื่อมต่อ TCP เข้ากับเซิบเวอร์ ร้องขอเว็บเพจและยกเลิกการเชื่อมต่อ ข้อความที่ร้องขอไปนี้ประกอบด้วย header หลายอย่างในการสนับสนุนข้อมูลเพิ่มเติม การทำ Caching, Replication และ content delivery network ถูกนำมาใช้อย่างแพร่หลายเพื่อเพิ่มประสิทธิภาพของเว็บ

การใช้เว็บไร้สายพึ่งจะอยู่ในขั้นเริ่มต้น ระบบแรกได้แก่ WAP และ i-mode ซึ่งประกอบด้วยหน้าจอขนาดเล็กและมีช่องสื่อสารขนาดจำกัด แต่รุ่นต่อไปคาดว่าจะมีความสามารถเพิ่มขึ้น

มัลติมีเดียก็กำลังได้รับความนิยมเพิ่มขึ้นเป็นอย่างมาก ซึ่งเป็นระบบที่ทำการแปลงสัญญาณเสียงและภาพให้กลายเป็นข้อมูลทางคอมพิวเตอร์และสามารถนำส่งได้ด้วยวิธีการทางอิเล็กทรอนิกส์และแสดงผลทางฝั่งผู้รับได้ ข้อมูลเสียงมีความต้องการช่องสื่อสารขนาดเล็กกว่า Streaming audio, Internet Radio และ voice over IP สามารถนำมาใช้งานได้ดีในปัจจุบัน โดยมีงานประยุกต์ใหม่ๆ ที่นำขีดความสามารถนี้ไปใช้เพิ่มขึ้นเรื่อยๆ