บทที่ 3

J2ME ติดต่อเว็บเซอร์วิส

3.1 เว็บเซอร์วิส

เว็บเซอร์วิส คือ เว็บแอพพลิเคชันยุคใหม่ ที่ประกอบด้วยส่วนย่อยๆมีความสมบูรณ์ในตัวเอง สามารถติดตั้ง ค้นหา เริ่มทำงานได้ผ่านเว็บ เว็บเซอร์วิส สามารถทำอะไรก็ได้ตั้งแต่งานง่ายๆ เช่นดึงข้อมูล จนถึงกระบวนการทางธุรกิจที่ซับซ้อน เมื่อ เว็บเซอร์วิสตัวใดตัวหนึ่งเริ่มเว็บเซอร์วิสตัวอื่นก็สามารถรับรู้ และเริ่มทำงานได้อีกด้วย

หลายคนอาจจะถามว่าทำไมต้องเป็น เว็บ เพราะโดยปกติมี Middle Ware อื่นๆมากมายเช่น RMI Jini CORBA DCOM ฯลฯ แม้ Middle Ware เหล่านี้จะสามารถรองรับได้ แต่ไม่มีตัวใดตัวหนึ่งที่เด่นจริง แต่ในเมื่อ Web มีจุดเด่นในเรื่องของการให้บริการข้อมูลที่สะดวก ใช้งานง่าย จึงกลายเป็นตัวประสาน Middle Ware ต่างๆ เข้าด้วยกันซึ่งจะให้คุยกันเองคงยากยิ่ง Web ได้ทำหน้าที่เป็นตัวกลางให้ Middle Ware เหล่านี้สามารถคุยกันได้ และมีประสิทธิภาพกว่าวิธีการเดิมๆ มาก

หากเรามองจากกรณีของ N-Tier แอพพลิเคชัน จะพบว่า เว็บเซอร์วิส คือกลไกในการเข้าถึงบริการที่แต่ละ Middle Ware ให้บริการ การเข้าถึงจะอาศัย Listener และส่วนประกอบที่ระบุถึงบริการต่างๆ ที่รองรับการทำงาน โดยการทำงานจริงๆ นั้นก็ใช้วิธีการปกติของ Middle Ware นั้นๆ
พื้นฐานของเว็บเซอร์วิสก็คือ XML กับ HTTP ซึ่งจะพบว่า HTTP ก็เป็นที่รู้จักกันดี และไปได้ทั่วทุกแห่งที่มี อินเทอร์เน็ต ส่วน XML คือภาษาสากลที่คุณสามารถปรับแต่งได้ตามใจชอบ เพื่อให้เกิดกิจกรรมระว่างไคลเอนต์ และบริการ หรือระหว่างส่วนประกอบต่างๆ เบื้องหลังเว็บเซอร์วิสก็คือ ข้อความ XML จะถูกแปลงให้การขอบริการจาก Middle ware และผลที่ได้ก็จะแปลงกลับมาในรูป XML โดยมาตรฐาน XML นี้ทำให้เราไม่ต้องกังวลเรื่องของการเชื่อมโยงอีกต่อไป และโพรโตคอล ที่ส่งก็คือ HTTP นั่นเอง ถ้าท่านเชื่อมโยงกับ HTTP (หรือเว็บ) ได้ ท่านก็ใช้บริการทุกอย่างใด

แต่อย่างไรก็ตามการเข้าถึงและการสั่งงานนั้นยังเป็นเพียงโครงสร้างพื้นฐาน แต่ในความเป็นจริงยังมีอะไรมากกว่านั้น เช่น การค้นหา การทำธุรกรรม ความปลอดภัย การพิสูจน์ตัวตน และอื่นๆ อันเป็นบริการที่ทำให้เป็นบริการพื้นฐานจริงๆ ระบบเพิ่มเติมที่ต้องมีและต้องรักษาความสะดวกและใช้งานง่ายไว้ด้วย พื้นฐานของเว็บเซอร์วิส เต็มรูปแบบคือ XML + HTTP + SOAP + WSDL + UDDI หรือในระดับสูงกว่านั้น แต่ไม่ได้ถือเป็นสิ่งจำเป็นเสมอไปคือต้องเพิ่มเทคโนโลยี XAML, XLANG, XKMS, XFS เป็นต้น สำหรับรายละเอียดในบางหัวข้อมีดังต่อไปนี้คือ

3.1.1 XML

3.1.1.1 แนะนำ XML

XML ย่อมาจาก eXtensible Markup Language XML ถูกออกแบบมาเพื่อใช้อธิบายข้อมูล ใน XML จะไม่มีแท็ก (tag) ที่กำหนดไว้ล่วงหน้า เราจะต้องสร้างแท็กของเราขึ้นมาเอง และจะใช้ Document Type Definition (DTD) ในการอธิบายรูปแบบของข้อมูล

ข้อแตกต่างที่สำคัญระหว่าง XML และ HTML คือ XML ถูกออกแบบมาเพื่อใช้แลกเปลี่ยนข้อมูล ไม่ใช่มาแทนที่ HTML XML กับ HTML ถูกออกแบบมาเพื่อจุดประสงค์ที่ต่างกัน HTML จะเกี่ยวข้องกับการแสดงข้อมูล XML จะเกี่ยวข้องกับการอธิบายข้อมูล แท็กของ HTML เช่น <H1></H1> เป็นแท็กสำหรับบอกรูปแบบการแสดงผล ในขณะที่แท็กของ XML ต้องสร้างขึ้นมาเองเช่น

```
<BOOK>
   <NAME>Beginning XML</NAME>
   <ISDN>1-861003-41-2</ISDN>
   <PAGE>800</PAGE>
<BOOK>
```

จะใช้แสดงข้อมูลของหนังสือที่ประกอบด้วย element BOOK , NAME , ISDN และ PAGE ข้อมูลที่อยู่ในรูปเท็กซ์นี้เป็นอิสระต่อซอฟต์แวร์และฮาร์ดแวร์ จึงเหมาะที่จะนำมาใช้ในการแลกเปลี่ยนข้อมูลระหว่างระบบบนอินเทอร์เน็ต การเปลี่ยนข้อมูลให้อยู่ในรูป XML เป็นการลดความซับซ้อนและสามารถสร้างข้อมูลที่ถูกอ่านโดยแอพพลิเคชันใด ๆ ก็ได้ นอกจากนั้นการเพิ่มเติม element จะไม่ทำให้แอพพลิเคชันเสียหาย เนื่องจากการอ่านข้อมูลโดย XML parser จะใช้วิธีแยกข้อมูลจาก element ที่ต้องการจาก XML ข้างต้นนี้จะแยกข้อมูล element NAME , ISDN และ PAGE หากเพิ่ม element AUTHOR เข้าไป แอพพลิเคชันก็ยังคงดึงข้อมูลใน element เดิมโดยจะมองข้าม element ที่ไม่ได้กำหนดให้แยกข้อมูลออกมาในตอนพัฒนาแอพพลิเคชัน

3.1.1.2 ประโยชน์จาก XML

สำหรับประโยชน์ของ XML นั้น เป็นด้านความยืดหยุ่นในการใช้งานสำหรับแอพพลิเคชันที่อิงกับ Web Base ที่ใช้ง่ายในการค้นหาข้อมูล มีความยืดหยุ่นในการพัฒนาเว็บ สามารถผสมผสานข้อมูลจากหลายแหล่ง จากแอพพลิเคชันที่ต่างกัน สามารถแสดงข้อมูลแบบต่างๆ และสามารถ update ข้อมูลให้ทันสมัยเสมอ และคาดว่าจะเป็นมาตรฐานใหม่ของระบบเปิด ซึ่งนับเป็น format ใหม่สำหรับการส่งข้อมูลบนเว็บที่มากด้วยข้อมูลหลายแบบ แต่ส่งผ่านด้วยเทคโนโลยีที่บีบอัดข้อมูลที่ให้ความเร็วได้รับการสนับสนุนจากผลิตภัณฑ์ค่ายไมโครซอฟท์

สถาปัตยกรรม XML

ภาษา XML ได้รับการสนับสนุนจาก W3C ให้นักพัฒนาเว็บแอพพลิเคชันได้หันมาใช้เป็นส่วนประกอบของการพัฒนาเว็บไซต์ เพราะ XML มีประสิทธิภาพและมีความน่าเชื่อสูงในการแปลงข้อมูลและโครงสร้างข้อมูลให้สามารถนำไปใช้งาน สถาปัตยกรรม 3 Tier ที่ XML สามารถสร้างขึ้นจากระบบข้อมูลที่ใช้โมเดลของ 3-Tier โครงสร้าง ของข้อมูลต่างๆ สามารถนำมาแสดงตามข้อกำหนด หรือรูปแบบที่ต้องการตามการใช้งานได้

XML เป็นการทำงานในระดับกลาง middle tier ที่สามารถเรียกใช้ฐานข้อมูลได้หลากหลายระบบ ฐานข้อมูลและโอนข้อมูลให้อยู่ในรูปแบบของ XML และมีการให้รายละเอียดเกี่ยวกับตัวข้อมูล โครงสร้างต่างๆ ของระบบฐานข้อมูลได้ XML เป็นระบบเปิดที่นำเสนอข้อมูลในรูปแบบ text ผ่านทาง HTTP เหมือนกับ HTML แต่จะมีคุณสมบัติในการให้ข้อมูลแบบ real time อัพเดทหรือเปลี่ยนแปลงได้ตามข้อกำหนด การแสดงข้อมูลจาก XML ใน HTML จะเป็นการเพิ่มในส่วนของรายละเอียดข้อมูล ที่มีการเรียกใช้จากแหล่งหรือฐานข้อมูลที่เชื่อมโยงกันในหลายแหล่งเพื่อให้HTMLมีความสมบูรณ์มากขึ้น ในอนาคตการพัฒนาเว็บหรือการเขียน และสร้าง HTML ไม่จำเป็นต้องมีการเขียนชุดคำสั่งที่ยุ่งยากซับซ้อนมากก็สามารถทำงานร่วมกับระบบข้อมูลได้อย่างมีประสิทธิภาพ XML จะทำการกำหนดค่าสำหรับโครงสร้างข้อมูลที่จะนำไปแสดงใน HTML นอกจากนั้นยังสามารถนำไปสนับสนุนระบบการแลกเปลี่ยนข้อมูลข่าวสารทาง Electronic ได้อย่างดีอีกด้วย

3.1.1.3 บทสรุปของ XML

XML มีความยืดหยุ่นพอในการนำเสนอข้อมูลแบบต่างๆ ที่สามารถให้รายละเอียดโครงสร้างข้อมูลตามระดับและความต้องการในการนำไปใช้งาน

3.1.2 SOAP

3.1.2.1 แนะนำ SOAP

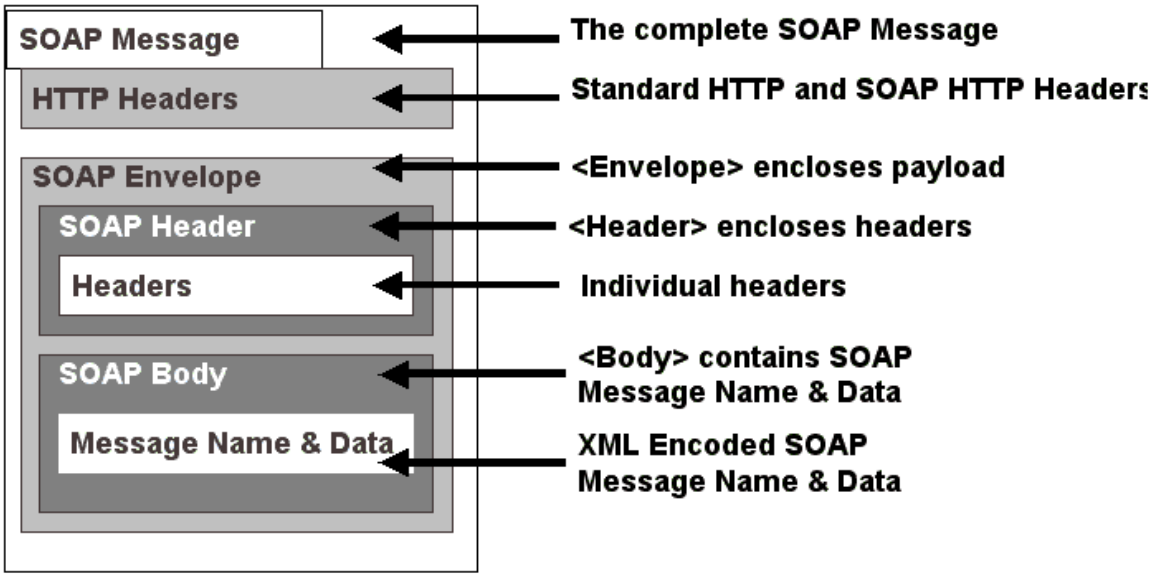
SOAP (Simple Object Access Protocol) เป็นโพรโตคอลที่ใช้ XML เป็นพื้นฐาน เพื่อให้ซอฟต์แวร์คอมโพเนนต์ และแอพพลิเคชันสามารถติดต่อกันผ่าน HTTP ซึ่งเป็นมาตรฐานอินเตอร์เน็ตโพรโตคอลได้

เนื่องจากการสื่อสารระหว่างแอพพลิเคชันจำเป็นต่อการพัฒนาแอพพลิเคชันบนอินเตอร์เน็ต เป็นอย่างมาก แต่แอพพลิเคชันแบบกระจายกันทำงาน (distributed application) ในปัจจุบันใช้ Remote Procedure Call (RPC) สื่อสารกันระหว่างออบเจ็กต์ เช่น DCOM และ CORBA ซึ่งไม่ได้ใช้พอร์ตเดียวกับ HTTP ที่เป็นพอร์ตมาตรฐานสำหรับให้บริการเว็บ ดังนั้น RPC จึงนำมาปรับใช้กับอินเตอร์เน็ตได้ยาก และมีปัญหาทางด้านความปลอดภัย ไฟร์วอลล์และพร็อกซีเซิร์ฟเวอร์จะไม่ยอมให้ส่งข้อมูลชนิดนี้ได้ตามปกติ วิธีที่ดีกว่าคือใช้ HTTP เพราะเป็นที่ยอมรับโดยอินเตอร์เน็ตบราวเซอร์ และเซิร์ฟเวอร์ทุกชนิด ซึ่ง SOAP ถูกสร้างมาเพื่อใช้ในกรณีนี้

ข้อดีของ SOAP ก็คือ SOAP ไม่ขึ้นกับคอมโพเนนต์เทคโนโลยี และภาษาการเขียนโปรแกรมใดๆ สามารถเขียนได้ง่ายและขยายเพิ่มเติมได้

3.1.2.2 ส่วนประกอบของ SOAP

SOAP เมสเซจใช้ไวยากรณ์ของ XML ในการสร้าง ประกอบด้วย 3 อีลีเมนต์มาตรฐาน คือ SOAP Envelope, SOAP Header และ SOAP Body



รูปที่ 3-1 โครงสร้างของ SOAP

SOAP เมสเซจ จะต้องเป็นไปตามกฎนี้

- Envelope เป็นอีลีเมนต์ที่อยู่บนสุด ต้องมีอีลีเมนต์นี้เสมอ
- Header อาจจะมีหรือไม่มีก็ได้ แต่ถ้ามีต้องเป็น child element แรกของ Envelope
- Body ต้องเป็น child element แรกของ envelope หรือ header

3.1.2.2.1 Envelope

envelope element เป็นอีลีเมนต์บนสุดของเอกสาร XML ที่ใช้แสดงเมสเซจ อาจประกอบด้วยแอตทริบิวต์ คือ envelope namespace และ encodingStyle

1. Envelope namespace – SOAP เมสเซจจะแสดงเวอร์ชันโดยใช้เนมสเปซของ envelope element namespace จะถูกนำไปใช้ขึ้นต้น envelope, header และ body element
2. EncodingStyle Attribute – ใช้แสดงวิธี serialization ของ SOAP เมสเซจ แอตทริบิวต์นี้สามารถมีได้ในทุกอีลีเมนต์และจะมีผลกับรายละเอียดของอีลีเมนต์นั้นและ child element ทั้งหมดของมันที่ไม่ได้ประกาศแอตทริบิวต์

3.1.2.2.2 Body

เป็นส่วนของข้อมูลที่ใช้ในการแลกเปลี่ยน โดยทั่ว ๆ ไปข้อมูลจะเป็นการ marshall RPC call และ error report child element ของ body element จะถูกเรียกว่า Body entry

body entry จะต้องเป็นไปตามกฎนี้

- body entry จะต้องแสดงด้วยชื่อเต็มที่ประกอบด้วยเนมสเปซ URI และ ชื่อของมัน (local name)
- SOAP encodingStyle attribute อาจจะมีได้ เพื่อแสดงถึงวิธี encode ของ body entry นั้น

3.1.2.2.3 Header

เป็นส่วนเพิ่มเติมของเมสเซจสามารถประกอบด้วยข้อมูลที่ระบุไปยังแอพพลิเคชัน ส่วนเพิ่มเติมนี้อาจนำไปใช้อิมพลีเม็นต์เป็น authentication, transaction management เป็นต้น ทุก ๆ child element ของ header element จะถูกเรียกว่า Header entry

header entry จะต้องเป็นไปตามกฎดังนี้

- header entry จะต้องแสดงด้วยชื่อเต็มที่ประกอบด้วยเนมสเปซ URI และ ชื่อของมัน (local name)
- อาจมี SOAP encodingStyle attribute ที่ใช้สำหรับ header entry
- SOAP mustUnderStand attribute และ SOAP actor attribute จะมีหรือไม่ก็ได้ เพื่อใช้บอกว่าจะจัดการกับ entry นั้นอย่างไรและโดยใคร

3.1.3 UDDI

UDDI (Universal Description, Discovery and Integration) เป็นมาตรฐานที่จัดตั้งขึ้น โดยบริษัท IBM, Microsoft และบริษัทยักษ์ใหญ่ทางธุรกิจ B2B (Business-to-Business) อื่น ๆ UDDI ถูกสร้างขึ้นมาเป็นมาตรฐาน ในการค้นหาบริการเว็บเซอร์วิสสำหรับคู่ค้าทางธุรกิจ ซึ่งเปรียบได้กับฐานข้อมูล ขนาดใหญ่ ซึ่งมีข้อมูลของเว็บเซอร์วิสที่เปิดให้บริการ โดยที่เว็บไซต์สำหรับค้นหาเว็บเซอร์วิสมีอยู่หลายที่อย่างเช่น

- http://uddi.microsoft.com/search.aspx
- http://www-3.ibm.com/services/uddi/testregistry/find
- http://uddi.org

3.1.4 WSDL

WSDL (Web Services Description Language) เกิดจากความร่วมมือระหว่าง IBM และ Microsoft WSDL เป็นภาษาที่ใช้อธิบายคุณลักษณะของเว็บเซอร์วิสและวิธีการติดต่อกับเว็บเซอร์วิสนั้นๆ โดยใช้ไวยากรณ์ของภาษา XML ซึ่ง WSDL อยู่ในความดูแลของ W3C เวอร์ชั่นล่าสุด คือ WSDL 1.1 รายละเอียด สามารถหาอ่านเพิ่มเติมได้จากเว็บไซต์ของ W3C ที่ http://www.w3.org/TR/wsdl ส่วนในทางปฏิบัติ หากเราต้องการ สร้าง web service ขึ้นมาเป็นของตนเอง ก็สามารถสร้างเอกสาร WSDL ได้โดยอัตโนมัติ เราจึงไม่ต้องไปกังวลในรายละเอียด ในข้อกำหนดใน WSDL มากนัก

3.2 J2ME ติดต่อเว็บเซอร์วิส

การติดต่อเว็บเซอร์วิสจะต้องใช้ SOAP ที่มีมาตรฐานเป็น XML ในการสนทนา ดังนั้นถ้า J2ME ต้องการจะติดต่อกับเว็บเซอร์วิส จำเป็นต้องมีการแปลงข้อมูลต่างๆ ที่ต้องการจะติดต่อกับเว็บเซอร์วิสให้อยู่ในรูปของ SOAP ที่มีมาตรฐาน XML ซึ่งการแปลงข้อมูลต่างๆ นั้น J2ME ไม่มี APIs รองรับการแปลงข้อมูลให้อยู่ในรูปของ SOAP ที่มีมาตรฐาน XML ดังนั้นวิธีการติดต่อกับเว็บเซอร์วิสสามารถทำได้ 2 วิธี ดังนี้

- ใช้ kXML และ kSOAP
- ผ่าน JSP หรือ Servlet

3.2.1 ใช้ kXML และ kSOAP

kXML และ kSOAP เป็นคลาสไลบรารี่ที่ได้ดาวน์โหลดมาจาก http://me.enhydra.org/ ซึ่ง kXML และ kSOAP เป็นไลบรารี่ที่ถูกพัฒนาขึ้นมาเพื่อให้ผู้พัฒนาแอพพลิเคชั่นทั่วไปสามารถนำไปใช้ได้ โดย kXML และ kSOAP จะทำหน้าที่แปลงข้อมูลจาก J2ME ให้อยู่ในรูปของ SOAP ที่มีมาตรฐาน XML ติดต่อไปที่เว็บเซอร์วิสและทำการแปลงเอกสาร XML ที่ได้รับจากเว็บเซอร์วิสให้อยู่ในรูปแบบที่ J2ME สามารถเข้าใจได้ ดังนั้นจึงใช้ kXML และ kSOAP ช่วยให้ J2ME สามารถติดต่อกับเว็บเซอร์วิสได้

3.2.1.1 ปัญหาที่เกิดขึ้นระหว่างการทดลอง

ในการแปลงข้อมูลให้อยู่ในรูปแบบของเอกสาร XML นั้นการทำ SOAP Encoding มีการเข้ารหัส 2 แบบ คือ

- แบบ xsi มีลักษณะดังนี้

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">
<SOAP-ENV:Body>
<ns1:searchBook xmlns:ns1="http://soapinterop.org/" SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<bookResult xmlns:ns2="http://soapinterop.org/xsd" xsi:type="ns2:Book">
 <author xsi:type="xsd:string">Ed Roman</author>
 <name xsi:type="xsd:string">Mastering EJB</intro>
 <page xsi:type="xsd:int">500</page>
 </bookResult>
 </ns1:searchBook>
```

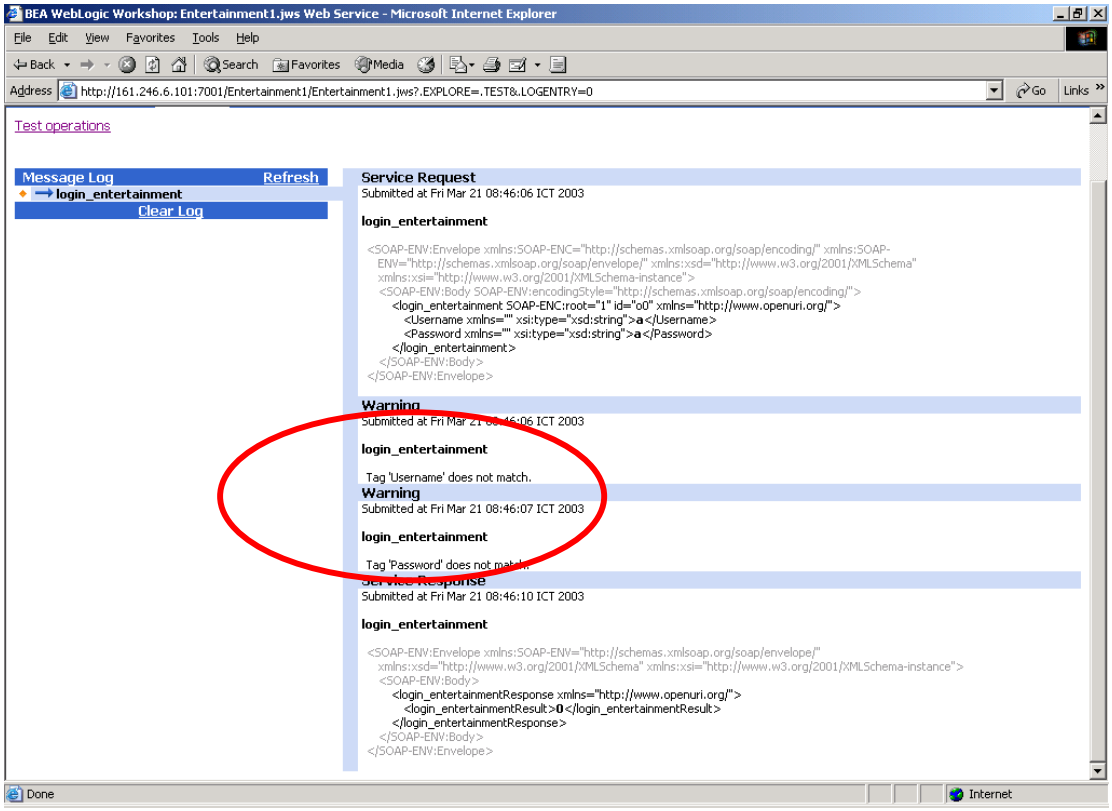
```
</SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

- แบบ WSDL มีลักษณะดังนี้

```
<types>
   <schema target namespace='http://soapinterop.org/xsd'
     xmlns='http://www.w3.org/2001/XMLSchema'
     xmlns:SOAP-ENC='http://schemas.xmlsoap.org/soap/encoding/'
     xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
     elementFormDefault='qualified'>
       <complexType name="Book">
           <sequence>
               <element name='author' type='string'/>
               <element name='name' type='string'/>
               <element name='page' type='int'/>
           </sequence>
       </complexType>
    </schema>
</types>
```

ปัญหาที่เกิดขึ้นคือ kXML และ kSOAP ที่ดาวน์โหลดมาใช้งานจะทำการเข้ารหัสในรูปแบบ xsi แต่ว่าการถอดรหัสที่เว็บลอจิกจะถอดรหัสในรูปแบบของ WSDL ซึ่งการเข้ารหัสและถอดรหัสของ kXML และ kSOAP กับเว็บลอจิกต่างกัน ดังนั้นเมื่อเว็บลอจิกถอดรหัสเอกสาร XML และส่งค่าต่างๆ ไปยังเซอร์วิส ทำให้เซอร์วิสที่ถูกเรียกใช้ได้รับข้อมูลที่ไม่ถูกต้อง จึงทำให้การเรียกเว็บเซอร์วิสโดยใช้ kXML และ kSOAP ไม่สามารถทำได้ ดังรูปที่ 3-2 แสดงถึงการเรียกเว็บเซอร์วิสโดยใช้ kXML และ kSOAP ซึ่งสามารถเรียกเซอร์วิสได้ แต่การถอดรหัส (Decode) ข้อมูลภายในเอกสารไม่สามารถถอดรหัสข้อมูลให้ออกมาอยู่ในรูปแบบที่ถูกต้องจึงทำให้เซอร์วิสไม่สามารถทำงานต่อได้

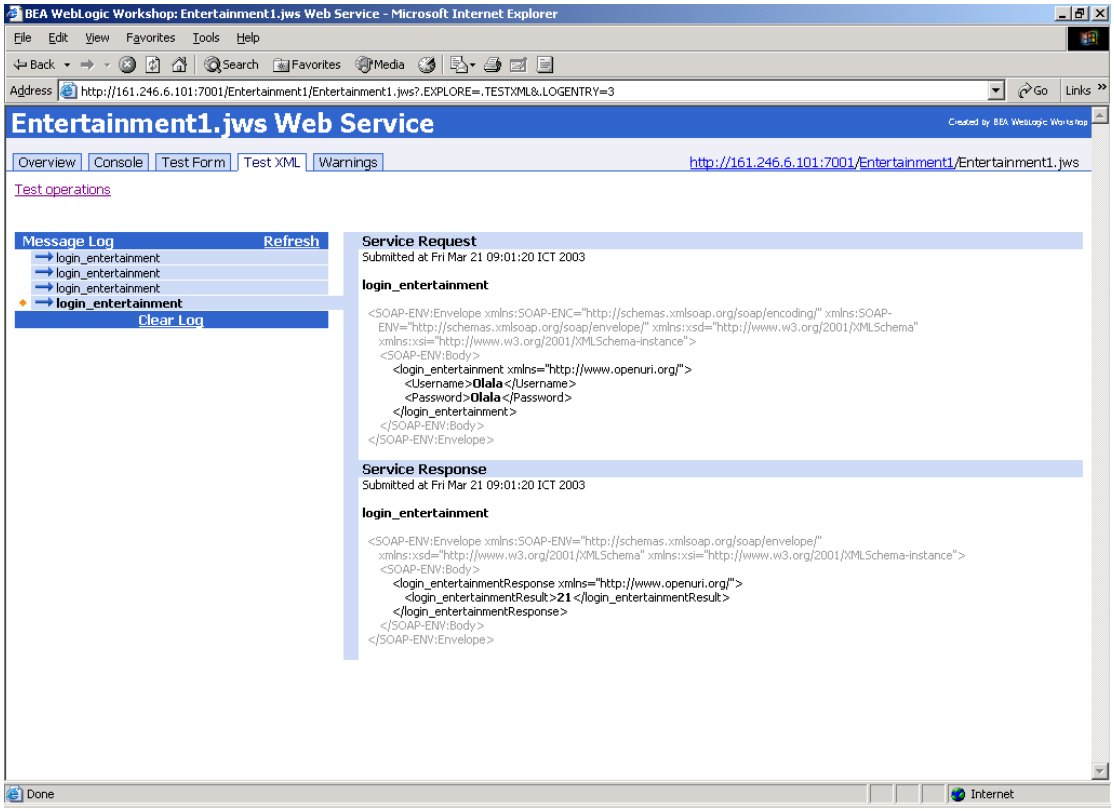
จากปัญหาที่เกิดขึ้นจึงไม่สามารถใช้ kXML และ kSOAP ในการติดต่อกับเว็บเซอร์วิสได้ เนื่องจากการถอดรหัสของเว็บลอจิกเซิร์ฟเวอร์ต่างกัน หากต้องการจะใช้ kXML และ kSOAP ในการติดต่อกับเว็บเซอร์วิส จะต้องใช้ติดต่อกับเว็บเซอร์วิสที่ดีพลอยบนเซิร์ฟเวอร์ที่ถอดรหัส SOAP ในรูปแบบของ xsi เท่านั้น เช่น Apache SOAP เป็นต้น



รูปที่ 3-2 แสดงการเรียกเว็บเซอร์วิสโดยใช้ kXML และ kSOAP

3.2.2 ผ่าน JSP หรือ Servlet

เนื่องจากเกิดปัญหาดังกล่าวจึงจำเป็นต้องใช้ JSP หรือ Servlet เป็นตัวกลางระหว่าง J2ME และเว็บเซอร์วิส โดยการติดต่อระหว่าง J2ME กับ JSP นั้นสามารถทำได้โดยใช้ HttpConnection ซึ่งเป็น APIs ที่ J2ME รองรับและการติดต่อระหว่าง JSP กับเว็บเซอร์วิสก็สามารถทำได้โดยผ่านทาง Proxy จึงทำให้วิธีการติดต่อระหว่าง J2ME กับเว็บเซอร์วิสสามารถติดต่อกันได้อย่างไม่มีปัญหา



รูปที่ 3-3 แสดงการเรียกใช้เซอร์วิสโดยผ่าน JSP หรือ Servlet