

An Introduction to eXtensible Markup Language

โดย

จุลภาพ บุรณจรรยากุล	รหัส 46654059
ธรรมสร ปรีวัฒนศักดิ์	รหัส 46654174
นภวรรณ ชมบุญ	รหัส 46654216
หนึ่งฤทัย ชัยเสวิกุล	รหัส 46654570
อิทธิชัย ลอดระกุล	รหัส 46654661
น.ต. อุกฤษฏ์ รอดสุทธิ	รหัส 46654679
ศุภเดช แก้วศรีช่วง	รหัส 46654471

เสนอ

รศ.ดร. ครรชิต มัลย์วงศ์ ราชบัณฑิต

วิชา 214552 Managing Information Technology

โครงการปริญญาโท สาขาวิชาเทคโนโลยีสารสนเทศ
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเกษตรศาสตร์

สารบัญ

เรื่อง	หน้า
■ What is XML?	1
■ XML vs HTML	3
■ Structure of XML Documents	4
■ XML Document Type Definition	8
■ Other XML Specifications	11
■ Reference	18

What is XML?

XML คืออะไร

เนื่องจากทุกวันนี้โลกของเรามีการนำเทคโนโลยีสารสนเทศเข้ามาใช้ในชีวิตประจำวันมากขึ้น และได้เข้ามาเป็นส่วนหนึ่งของการทำงาน ที่เห็นได้ชัดคือ เทคโนโลยีบนอินเทอร์เน็ต ไม่ว่าจะเป็นการเข้าไปเยี่ยมชมเว็บไซต์ต่าง ๆ หรือการส่งจดหมายอิเล็กทรอนิกส์ที่เรียกว่า E-mail ซึ่งกำลังเป็นมาตรฐานของการติดต่อสื่อสารสำหรับอนาคต ทำให้ต้องมีการคิดเพื่อพัฒนาให้มีความก้าวหน้ามากขึ้น การเขียนเว็บไซต์ในปัจจุบันนี้ โดยปกติแล้วสิ่งที่จะใช้สร้างเว็บได้คือ ภาษา HTML ที่เรารู้จักกันดีเท่านั้น แต่ວ່ວวันนี้โลกได้มีการพัฒนาเทคโนโลยีใหม่ที่ใช้สำหรับการเขียนเว็บ นั่นคือ XML ซึ่งเป็นสิ่งที่หลายผลิตภัณฑ์ให้การสนับสนุน

Extensive Markup Language (XML) เป็นภาษาที่ให้ความชัดเจนในการให้รายละเอียดเกี่ยวกับข้อมูล และการเปลี่ยนแปลงข้อมูลโดยแอปพลิเคชันบนเว็บและใช้ฟอร์มที่ยืดหยุ่นได้ ตามมาตรฐาน HTML หรือ Hyper Text Markup Language ได้เปิดโลกแห่งการนำเสนอข้อมูลต่างๆ ส่วน XML ทำงานกับข้อมูลโดยตรงซึ่งเสริมกับการทำงานของ HTML

XML เป็นฟอร์แมตที่อธิบายถึงรายละเอียดของโครงสร้างและแบบของข้อมูลเป็นภาษาหรือชุดคำสั่งมีการพัฒนาและมีศักยภาพในส่วนของการสร้างข้อมูล ด้วย XML จะทำให้การจัดการข้อมูลหรือเรียกใช้ข้อมูลจากแอปพลิเคชันต่างๆ จะเข้าสู่มาตรฐานเดียวกัน

ความเป็นมาของ XML (Extensive Markup Language)

โปรโตคอลอินเทอร์เน็ต (Internet Protocol-IP), HypertText Markup Language (HTML)และ HyperText Transport Protocol (HTTP) ได้เป็นการปฏิวัติและสร้างมิติใหม่ในการกระจายข้อมูลและสารสนเทศ การนำเสนอ ตลอดจนการค้นคืน โดยให้ผู้ใช้สามารถใช้สารสนเทศที่ต้องการได้ง่ายด้วยเบราว์เซอร์ และมี search engine หรือเครื่องมือ ในการช่วยค้นหา นอกจากนั้นยังมีการประยุกต์ไปใช้กับเครือข่ายในสำนักงานหรืออินเทอร์เน็ต และใช้สำหรับการบริการข้อมูลสำหรับลูกค้าและคู่ค้าให้สามารถตอบสนองทางด้านสารสนเทศที่ต้องการ ได้อย่างมีประสิทธิภาพมากขึ้น

สำหรับ XML จะให้ประโยชน์อย่างเต็มที่เมื่อทำงานร่วมกับ HTML ด้วยเหตุที่ว่า XML ได้มีความพร้อมในแง่ของรายละเอียด และการนำข้อมูลตลอดจนโครงสร้างข้อมูลมาแสดงได้ในรูปแบบ Text ผ่านทาง HTTP ที่เปิดให้ข้อมูลขึ้นใหม่และมีความสามารถในการจัดข้อมูลได้อีกด้วย ในการเขียนเว็บเพจเมื่อใช้ HTML ผู้พัฒนาสามารถ

กำหนดได้ว่าส่วนไหนจะเป็นตัวหนา ตัวเอียง หรือตัวอักษรเป็นแบบไหน ส่วน XML นั้นจะเป็นการเตรียมส่วนของข้อมูลที่จะนำไปใส่ในช่องที่กำหนดตามการเขียนของ HTML ยกตัวอย่างเช่น ข้อมูลด้านราคาสำหรับการจัดรายการส่งเสริมการขาย อัตราภาษี ค่าขนส่ง เป็นต้น

XML ถือได้ว่าเป็นส่วนหนึ่งของ Standard Generalized Language Markup Language (SGML) ที่เป็นข้อกำหนดในการสร้างหรือจัดทำเอกสารในรูปแบบอิเล็กทรอนิกส์ที่กำหนดโดย W3C หรือ World Wide Web Consortium ที่มีโครงสร้างและรูปแบบที่เปิดให้แอปพลิเคชันต่างๆ สามารถเรียกไปใช้งานได้ เช่น บนเว็บไซต์ต่างๆ เป็นต้น และทางไมโครซอฟท์ได้มีการทำงานร่วมกับ W3C เพื่อพัฒนามาตรฐานข้อมูลบนเว็บที่ให้ HTML สามารถแสดงข้อมูลที่ XML ได้เตรียมไว้ได้ และทางไมโครซอฟท์เองได้มีการเปิดตัว เบราวเซอร์ตั้งแต่ IE 4.0 เป็นต้นไป ที่สามารถเรียกดู และเป็นข้อกำหนดให้ เบราวเซอร์เวอร์ชันใหม่ของค่ายไมโครซอฟท์สนับสนุน XML ด้วย

สิ่งที่ถือได้ว่าเป็นเสน่ห์ของ XML นั้นคือความสะดวกในการจัดการด้านระบบการติดต่อกับผู้ใช้จากโครงสร้างของข้อมูล เราสามารถนำข้อมูลจากหลายแหล่งมาแสดงผลและประมวลผลรวมกันได้ ไม่ว่าจะเป็นข้อมูลลูกค้า รายการสั่งซื้อ ผลการวิจัย รายการรับชำระเงินข้อมูลเวชระเบียน รายการสินค้าหรือข้อมูลสารสนเทศอื่นๆ ก็สามารถแปลงให้เป็น XML ได้ และในส่วนของข้อมูลสามารถปรับให้เป็น HTML ได้

สำหรับประโยชน์ในการใช้งานนั้น เราจะสามารถนำมาใช้สำหรับการเข้าถึงระบบข้อมูลขนาดใหญ่ให้กับระบบเครือข่ายในองค์กร หรืออินเทอร์เน็ต เพื่อดูข้อมูลหรือเรียกใช้ข้อมูลที่ทำให้การแสดงผลทางหน้าจอที่รวดเร็วและง่ายในการจัดการ

วัตถุประสงค์หลักของ XML

1. XML มีการใช้งานโดยตรงบนเครือข่าย Internet XML จะถูกออกแบบมาสำหรับจัดเก็บและจัดส่งข้อมูลบนเว็บ
2. XML มีการสนับสนุนโปรแกรมที่หลากหลาย ถึงแม้ว่าวัตถุประสงค์ที่สำคัญคือ การจัดส่งข้อมูลบนเว็บผ่านทางเซิร์ฟเวอร์และโปรแกรมเบราวเซอร์ XML จะถูกออกแบบมาเพื่อใช้กับโปรแกรมที่มีรูปแบบต่าง ๆ ตัวอย่างเช่น การแลกเปลี่ยนข้อมูลระหว่างโปรแกรมทางการเงิน การเผยแพร่และปรับปรุงโปรแกรมให้ทันสมัยและการเขียน Voice Script ให้สื่อสารได้ด้วยโทรศัพท์
3. XML จะต้องเข้ากันได้กับ SGML
4. XML จะต้องง่ายต่อการเขียนโปรแกรมเพื่อประมวลผลเอกสาร
5. จำนวนของทางเลือกเฉพาะของ XML ควรมีจำนวนน้อยที่สุดหรือไม่ควรมีเลย
6. เอกสาร XML จะต้องอ่านเข้าใจง่ายและมีความชัดเจน

7. XML ออกแบบมาเพื่อให้พัฒนาโปรแกรมได้อย่างรวดเร็ว
8. การออกแบบ XML ต้องมีรูปแบบที่เหมาะสมและกะทัดรัด
9. สามารถสร้างเอกสาร XML ได้ง่าย
10. Markup ของ XML ต้องไม่รวบรัดมากเกินไป

สถาปัตยกรรม XML

ภาษา XML ได้รับการสนับสนุนจาก W3C ให้นักพัฒนาเว็บแอปพลิเคชันได้หันมาใช้เป็นส่วนประกอบของการพัฒนาเว็บไซต์ เพราะ XML มีประสิทธิภาพและมีความน่าเชื่อถือสูงในการแปลงข้อมูลและโครงสร้างข้อมูลให้สามารถนำไปใช้งาน สถาปัตยกรรม 3 Tiered ที่ XML สามารถสร้างขึ้นจากระบบข้อมูลที่ใช้โมเดลของ 3-tier โครงสร้าง ของข้อมูลต่างๆ สามารถนำมาแสดงตามข้อกำหนด หรือรูปแบบที่ต้องการตามการใช้งานได้ โครงสร้างทางสถาปัตยกรรมของ XML ที่ยืดหยุ่นเหมาะกับการพัฒนาร่วมกับเว็บแอปพลิเคชัน

ส่วนประกอบของข้อมูล

XML เป็นการทำงานในระดับกลาง middle tier ที่สามารถเรียกใช้ฐานข้อมูลได้หลากหลายระบบฐานข้อมูลและโอนข้อมูลให้อยู่ในรูปแบบของ XML และมีการให้รายละเอียดเกี่ยวกับตัวข้อมูล โครงสร้างต่างๆ ของระบบฐานข้อมูลได้ XML เป็นระบบเปิดที่นำเสนอข้อมูลในรูปแบบ text ผ่านทาง HTTP เหมือนกับ HTML แต่จะมีคุณสมบัติในการให้ข้อมูลแบบ real time อัปเดตหรือเปลี่ยนแปลงได้ตามข้อกำหนด การแสดงข้อมูลจาก XML ใน HTML จะเป็นการเพิ่มในส่วนของรายละเอียดข้อมูล ที่มีการเรียกใช้จากแหล่งหรือฐานข้อมูลที่เชื่อมโยงกันในหลายแหล่ง เพื่อให้ HTML มีความสมบูรณ์มากขึ้น

ในอนาคตการพัฒนาเว็บหรือการเขียนและสร้าง HTML ไม่จำเป็นต้องมีการเขียนชุดคำสั่งที่ยุ่งยากซับซ้อนมากก็สามารถทำงานร่วมกับระบบข้อมูลได้อย่างมีประสิทธิภาพ XML จะทำการกำหนดคำสั่งสำหรับโครงสร้างข้อมูลที่จะนำไปแสดงใน HTML นอกจากนั้นยังสามารถนำไปสนับสนุนระบบการแลกเปลี่ยนข้อมูลข่าวสารทาง Electronic ได้อย่างดีอีกด้วย

XML vs HTML

ภาษา XML ไม่ใช่ทั้ง เวอร์ชันใหม่ของ HTML และไม่ใช่ส่วนขยายของ HTML รวมทั้งไม่ใช่ภาษา XHTML ด้วย แต่ XML เป็นภาษาที่ถูกออกแบบมาเพื่อช่วยเพิ่มความสามารถให้กับ HTML

ภาษา XML มีจุดประสงค์ในการใช้งานแตกต่างกับภาษา HTML โดยที่ HTML ถูกออกแบบมาให้นับการแสดงผลข้อมูลเพียงอย่างเดียว (โดยที่ไม่รู้ว่าข้อมูลที่กำลังแสดงอยู่คืออะไร) เช่น การแสดงข้อมูลเป็นตัวหนา ตัวเอน เป็นต้น ซึ่งก็คือ เว็บเพจที่เห็นกันทั่วไป

แต่สำหรับ XML แล้ว จะแตกต่างกันอย่างสิ้นเชิง เพราะว่า XML สามารถล่วงรู้หรืออธิบายได้ว่าข้อมูลส่วนนี้คืออะไร ส่วนเรื่องการตกแต่งหรือแสดงผลข้อมูลดังกล่าวออกมา จะมอบภาระให้ภาษาอื่น ทำหน้าที่แทน เช่น XSL เป็นต้น

อาจกล่าวได้ว่า การใช้งาน XML คือการนิยามข้อมูลขึ้นมา โดยใช้ แท็กระบุว่าคุณมุนั้นคืออะไร เป็นการมอง XML ในลักษณะเป็นแหล่งข้อมูลที่เราสามารถเรียกใช้งานจากภาษาใดก็ได้ ส่งผลให้เราสามารถแลกเปลี่ยนข้อมูลกับภาษาใดที่รู้จักกับ XML ได้

Structure of XML Documents

ลักษณะโครงสร้างของ XML

XML เป็นการใช้ข้อความเพื่อบ่งบอกโครงสร้างของเอกสาร พิจารณาตัวอย่างรูปแบบโครงสร้างของหนังสือ เมื่อหนังสือประกอบด้วยจำนวนบท 2 บท ในแต่ละบทประกอบด้วยเนื้อความ (Text)

Begin Book

Begin Chapter 1

Text for Chapter 1

End Chapter 1

Begin Chapter 2

Text for Chapter 2

End Chapter 2

End Book

หนังสือที่มีอยู่ในปัจจุบัน จะมีโครงสร้างที่มีรายละเอียดที่ซับซ้อนมากกว่านี้ เช่น บทนำ, สารบัญ, เป็นต้น เช่นเดียวกัน ภายในส่วนเนื้อความ (Text) ยังประกอบด้วยโครงสร้างย่อย คือ ย่อหน้า (Paragraph) แต่ละย่อหน้ายังประกอบขึ้นจาก ประโยค คำ และตัวอักษรด้วย

ลักษณะของเอกสาร XML นั้น สามารถอธิบายโดยใช้ตัวอย่างที่ 1 ได้ ดังนี้
ตัวอย่างที่ 1

```
<?xmlversion="1.0"encoding="windows-874"?>
```

```
<mali>
```

```
<malisorn>ขึ้นต้นด้วยมะลิซ้อน</malisorn>
```

```
<malila>พอแตกใบอ่อนเป็นมะลิลา</malila>
</mali>
```

บรรทัดที่ 1 นั้นหมายความว่าเราประกาศเอกสารนี้เป็นเอกสาร XML และมีการเข้ารหัสอักขระเป็น windows-874 เพื่อให้ใช้ภาษาไทยได้ จากนั้นเราจะเห็น สิ่งที่เราคุ้นเคยที่เราเรียกว่า tag แต่จริงๆแล้วในภาษา XML จะแบ่งโครงสร้างเป็น 2 ส่วนใหญ่ ๆ คือ tag & element สามารถอธิบายเพิ่มเติมได้จากตัวอย่างที่ 2

ตัวอย่างที่ 2

```
<root>
  <element>
    <tag></tag>
  </element>
</root>
```

ความหมายของ Tag กับ Element

Tag

สำหรับใน XML แล้ว tag มีความหมายในลักษณะเดียวกับที่ใช้ใน HTML tag คือข้อความที่อยู่ระหว่าง

สัญลักษณ์ < และ >

- Tag เปิด (Start tag)

```
<book>
```

จากตัวอย่างที่แสดง ด้านบนถูกเรียกว่า tag เปิด ดังนั้น tag เปิดจึงมีสัญลักษณ์คือ <...>

- Tag ปิด (End Tag)

```
</book>
```

tag ที่ถูกเรียกว่า tag ปิด ต่อเมื่อใน tag มีเครื่องหมาย / อยู่หลังสัญลักษณ์ < ดังนั้นลักษณะของ tag ปิดจึงมีรูปแบบคือ </...> หากพิจารณาระหว่าง tag เปิดกับ tag ปิดแล้ว ข้อแตกต่างอีกข้อหนึ่งคือ tag เปิด เป็น tag ที่สามารถใส่ข้อมูล attribute ลงไปภายใน tag ได้ แต่ tag ปิดจะไม่ทำกัน

Element

ในที่นี้คือ โครงสร้างหลักของ XML ซึ่งอยู่ในรูปของ tag เช่นเดียวกัน

```
<root>
  <element>
```

```
</element>
```

```
</root>
```

Element จะมีลักษณะซ้อนกันเป็นชั้นๆ

- Root element เป็น element แรกสุดของเอกสาร XML

```
<chap number="1">Text for Chapter 1</chap>
```

สังเกตจากลักษณะ ทั้งหมดตั้งแต่ <chap จนถึง </chap> ถูกเรียกว่า element หรือถ้ามองง่าย ๆ คือ element เริ่มต้นที่ tag เปิด และสิ้นสุดที่ tag ปิดใน tag คำสั่งเดียวกัน

Content

เนื้อหา หรือ Content ถือได้ว่าเป็นข้อมูลเพื่อใช้ในการแสดง กล่าวอีกนัยหนึ่งคือ Content อยู่หลัง tag เปิด และจบที่ก่อนถึง tag ปิดนั่นเอง

Attribute

Attribute คือตัวอักษรสีน้ำเงิน จากตัวอย่าง และจะเห็นว่า ถูกบรรจุอยู่ใน tag เปิด และ attribute นี้ในบางครั้ง อาจจะมีหรือไม่มีก็ได้ รวมถึงถ้ามีอาจมีได้มากกว่าหนึ่งตัวใน tag เปิด attribute คือข้อมูลความหมาย

การประยุกต์ใช้ XML

ตัวอย่างการประยุกต์ใช้งาน XML ตัวอย่างแรก ของเอกสาร XML:

```
<book>
```

```
<chap>
```

```
Text for Chapter 1
```

```
</chap>
```

```
<chap>
```

```
Text for Chapter 2
```

```
</chap>
```

```
</book>
```


จากรูปเป็น ไวยากรณ์ที่ถูกใช้ใน XML เมื่อเปรียบเทียบตัวอย่างนี้กับเอกสารที่เห็นในส่วนต้นของบทความ สำหรับโครงสร้างของหนังสือ โดยเปรียบเทียบทีละส่วน โดยพิจารณาที่ส่วนของ element (สังเกตกลุ่ม element จากสัญลักษณ์ <...> จนถึง </...>) ดังนั้นโครงสร้างของ XML จะมี element เป็นส่วนประกอบ ใช้กำหนดโครงสร้างของเอกสาร

ตัวอย่างถัดไปด้านล่าง เป็นการปรับปรุงตัวอย่าง XML ก่อนหน้าที่มี element อยู่ โดยเพิ่มส่วนที่เรียกว่า attribute ลงใน element คือตัวเลขบอกเลขที่ของบท โดยชื่อ attribute ใช้คำว่า number และ ตามด้วยตัวเลขบอกความหมายของบท

```
<book>
  <chap number="1">
    Text for Chapter 1
  </chap>
  <chap number="2">
    Text for Chapter 2
  </chap>
</book>
```

จากตัวอย่างโครงสร้าง เมื่อพิจารณาจะเห็นได้ว่า มีรูปแบบของจำนวนบทอยู่ 2 บท ในแต่ละบทมีข้อความอยู่บางส่วน เช่น บทที่ 1 มีข้อความ Text for Chapter 1 ที่อยู่ระหว่าง element chap จะมี attribute ชื่อ number โดยบทที่ 1 ใช้ number="1" เป็นข้อมูลของ attribute

จากโครงสร้างเอกสาร XML ข้างต้นสามารถนำไปผ่านขั้นตอนการ render เพื่อให้ได้เอกสารใช้งานจริงๆ ได้ในหลากหลายรูปแบบ ข้อมูลที่อยู่ใน attribute number มีความหมายที่จะสามารถทำให้กลไก render สามารถแยกแยะเอกสารตามรายละเอียด ของเนื้อหาในแต่ละบทได้ ดังนั้นหากต้องการให้มุมมองของการมองเห็นแตกต่างกัน ก็ย่อมทำได้เช่นเดียวกัน ยกตัวอย่างเช่น หากนำเอกสาร XML ผ่านขั้นตอนการ render สำหรับหนังสือที่เป็นกระดาษ การจัดพิมพ์ย่อจะเกิดหน้าว่างหนึ่งหน้า ของแต่ละบทเสมอ ในขณะที่เอกสารตัวเดียวกัน หากนำไปผ่านขั้นตอนการ render ของมอนิเตอร์แล้ว ก็ไม่จำเป็นต้องทำให้เกิดหน้าว่างก่อนบท

เอกสาร XML เพียงบอกแค่ ข้อมูลในส่วนที่เป็น Content ซึ่งถูกแยกแยะข้อมูลออกเป็นส่วนๆ ในลักษณะเอกสารที่มีโครงสร้าง หรือกล่าวอีกนัยหนึ่งคือ XML ใช้สำหรับแยกแยะเอกสารออกเป็นส่วนๆ เท่านั้น โดยที่ข้อมูลในเอกสาร ไม่มีส่วนที่ใช้บอกว่าควรจะแสดง Content ในลักษณะการมองเห็นแบบไหน

การใช้งานเอกสาร XML

ในอนาคตคาดว่า รูปแบบการ rendering มีการพัฒนาไปอย่างมาก ดังจะเห็นได้จาก บริษัทยักษ์ใหญ่ ค่ายต่าง ๆ ทางด้านไอที และคอมพิวเตอร์ ได้ร่วมมือกัน และพัฒนาโครงสร้าง หรือกลไกการ rendering ไว้ในแผนการของตัวเองแล้ว หากเราสามารถนำข้อมูลของเราเอง มาเก็บไว้ในรูปแบบของ XML จะทำให้ข้อมูลของเรา มีคุณค่าในอนาคต และสอดคล้องกับเทคโนโลยีที่จะมีมาด้วย

- XML จะเกิดความสะดวกในระบบพาณิชย์อิเล็กทรอนิกส์

ตัวอย่างหนึ่งคือเทคโนโลยี XML และ Java ได้มีการกำหนดข้อตกลงต่างๆ เพื่อพัฒนาขบวนการค้า หรือ E-Commerce ในการประชุม Software Development East ที่ผ่านมา และพวกเขาเชื่อมั่นว่า ในระบบการค้าอิเล็กทรอนิกส์ ข้อมูลจะถูกเก็บอยู่ในรูปแบบของเอกสาร XML ทั้งหมด อย่างเร็วที่สุดในปลายปี 2001

- XML ทำให้ผู้ค้าและผู้ใช้มีความอิสระ

ผู้พัฒนาเทคโนโลยี มีความอิสระในการเพิ่มศักยภาพในผลิตภัณฑ์ของตัวเอง ไม่จำเป็นต้องมากังวลรูปแบบการสื่อสาร ที่จะต้องออกแบบมาเฉพาะ ทำให้ลูกค้าต้องยึดติดกับผลิตภัณฑ์รายใดรายหนึ่ง เมื่อข้อมูลอยู่ในเอกสาร XML แล้ว ผลิตภัณฑ์ต่างๆ ก็จะมีรูปแบบข้อมูลที่สามารถแลกเปลี่ยนกันได้ โดยที่ไม่ต้องไปกังวลกับเทคโนโลยี ที่ไม่สอดคล้องกันในปัจจุบันได้

- XML ลดค่าใช้จ่าย

แน่นอน ผลที่ตามมา กับ เทคโนโลยีที่แลกเปลี่ยนข้อมูลกันได้ ทำให้ค่าใช้จ่ายที่จะต้องสูญเสียไป ในปัจจุบันสำหรับการสื่อสารข้อมูล ที่มีรูปแบบที่หลากหลายถูกขจัดออกไป นั่นเป็นผลดีสำหรับยุคการสื่อสารด้วย XML

XML Document Type Definition

ข้อกำหนดโครงสร้างสำหรับ XML

DTD คืออะไร

"DTD คือแฟ้มข้อมูล (หรือหลายแฟ้มข้อมูลที่ทำงานร่วมกัน) ซึ่งบรรจุข้อกำหนด และกฎเกณฑ์ของเอกสาร ชุดข้อกำหนดเหล่านี้ สำหรับการกำหนดรูปแบบ element ตัวอย่างเช่น หากต้องการเอกสารที่มี element <LIST> ที่มี element <ITEM> บรรจุอยู่ภายใน ข้อกำหนดในแฟ้มข้อมูล DTD จะมีรูปแบบดังนี้

```
<!ELEMENT item (#pcdata)>
```

```
<!ELEMENT list (item)+>
```

ซึ่งอธิบายความหมายคือ element items บรรจุข้อความใดๆ และ element list บรรจุ element item อีกที ดังนั้น DTD เป็นรูปแบบภาษา ซึ่งทำให้สามารถตรวจสอบเอกสาร ที่นำเอาข้อกำหนด DTD ไปใช้ ว่าถูกจัดสร้าง

ตามความต้องการหรือไม่ ทำให้ระบบการ rendering สามารถเข้าใจตัวเอกสารได้ดี และดึงไปใช้งานได้อย่างถูกต้อง"

DTD มีความซับซ้อน

จากคำอธิบายในส่วนต้น DTD มีความซับซ้อน หมายถึงการสร้าง DTD มีรายละเอียดและขั้นตอนที่ยุ่งยากพอสมควร ถึงแม้ DTD จะทำให้การสร้างเอกสารมีความยุ่งยากก็ตาม แต่มีเหตุผลอยู่ 2 ข้อที่ควรรู้ก่อน

XML ไม่จำเป็นต้องใช้ DTD เสมอไป

ถ้าแม้จะต้องใช้ DTD แต่ก็จะมีการสร้าง DTD ที่เป็นมาตรฐานให้ใช้อยู่แล้ว DTD ที่เป็นมาตรฐาน ถูกพัฒนาโดยผู้ที่สนใจ ในเอกสารเฉพาะด้าน และมีให้เรา ได้ดึงมาใช้งาน เพื่อให้เกิดรูปแบบมาตรฐานที่ชัดเจน และเข้าใจข้อมูลกัน ในเอกสารประเภทเดียวกัน 3 ส่วนสำคัญ เอกสาร XML มีส่วนประกอบที่สำคัญอยู่สามส่วน สองส่วนเป็นสิ่งที่จำเป็น ในขณะที่อีกส่วนเป็นทางเลือก ที่จะมีหรือไม่ก็ได้

1. **ส่วนแรก** คือส่วนของเนื้อหา หรือ Content นั่นเอง ซึ่งทำให้เอกสารมีข้อมูลสำหรับดูโดยผู้อ่าน Content นี้เป็นได้ทั้งข้อความ รูปภาพ ส่วนนี้ถูกสร้างขึ้นมาจาก element ในที่นี้คือไฟล์ XML
2. **ส่วนที่สอง** คือกฎเกณฑ์และข้อกำหนด โครงสร้างของเอกสาร ในที่นี้คือไฟล์ DTD ส่วนนี้ถือเป็นทางเลือก ซึ่งจะเลือกใช้หรือไม่ก็ได้
3. **ส่วนสุดท้าย** คือ StyleSheet คือ ลักษณะข้อกำหนดสำหรับการแสดงผลพอร์นั่นเอง ในที่นี้คือไฟล์ XSL

DTD คือออปชั่น XML สามารถถูกบังคับโครงสร้างด้วย DTD ซึ่งจะใช้หรือไม่ใช้ก็ได้ แต่ StyleSheet คือสิ่งที่จำเป็นที่จะสามารถนำข้อมูลในเอกสาร มาแสดงด้วยการ rendering StyleSheet ในบางครั้งไม่ต้องการ หากเรามีกลไก rendering ที่จัดสร้างไว้พร้อมแล้ว สำหรับเอกสาร XML ที่ต้องผ่านขบวนการแปลงด้วย XSL เพื่อให้เป็น HTML ในการแสดงผ่านบราวเซอร์ อย่างนี้ก็ต้องจำเป็นในการใช้งาน StyleSheet

เอกสาร DTD

ในการจัดสร้าง DTD ทำให้เอกสารมีความเป็นรูปแบบ well-formed เรียกง่าย ๆ กันว่า เอกสารที่ถูกต้อง ในบางครั้งหากเรา ต้องการสร้างเอกสาร XML ที่สามารถแลกเปลี่ยนกับคนอื่น ๆ หรือส่วนอื่นๆ และให้สามารถเข้าใจโครงสร้างเดียวกัน นั่นก็คือ เอกสาร XML ของเรา และส่วนอื่นๆ จำเป็นต้องมีรูปแบบโครงสร้างเดียวกัน นั่นก็คือควรที่จะเลือกใช้ DTD เดียวกัน เพื่อบังคับโครงสร้างให้เหมือนกัน

เอกสารที่ถูกต้อง

เอกสารที่อยู่ในรูปแบบ well-formed เป็นเอกสารที่เรียกว่าถูกต้องตามข้อกำหนดของ XML แต่เอกสารที่ไม่ถูกต้องตามรูปแบบ XML ก็สามารถเป็นเอกสารที่ดีได้เช่นกัน เนื่องจากในปัจจุบันข้อมูล บนระบบอินเทอร์เน็ตมีอยู่

มากมาย ที่ไม่ได้สร้างเป็นเอกสารที่ถูกต้องตามแบบ XML นั่นคือความต้องการส่วนหนึ่งที่ DTD จะสามารถบังคับเอกสาร XML ที่สร้างมาจาก DTD เดียวกัน ให้เป็นเอกสารที่ถูกต้องตามแบบ XML (ต้องเป็น DTD ที่ถูกต้องด้วย)

เอกสารรูปแบบ well-formed

แนวคิดของเอกสารแบบ well-formed ถูกนำมาใช้สำหรับ XML ในกรณีที่ ไม่สามารถสร้างเอกสาร XML ด้วยข้อกำหนด DTD

"จากตัวอย่างของ tag ใน HTML เป็น tag ที่เรียกว่า element เปล่า เนื่องจากว่า tag ดังกล่าว ไม่จำเป็นต้องอาศัย tag ปิด เช่นเดียวกัน หาก XML มี element ที่มี tag ปิด จึงไม่เป็นเอกสารในรูปแบบ 'well-formed' "

เอกสาร HTML ไม่เป็นเอกสารในรูปแบบ well-formed

เนื่องจากคำสั่ง หรือ tag ในภาษา HTML มีหลาย tag ที่มีเฉพาะ tag เปิด โดยที่ไม่จำเป็นต้องใช้ tag ปิดเลย ทำให้เอกสารที่สร้างด้วย tag HTML จึงไม่ถือเป็นเอกสาร well-formed แต่เอกสาร XML ต้องเป็นเอกสารรูปแบบ well-formed ซึ่งมีลักษณะ ดังนี้

1. element ที่ถูกต้อง element ต้อง tag เปิด และต้องปิดด้วยด้วย tag ปิดเสมอ เช่น <...> </...> ยกเว้น element ว่า
2. ข้อมูล attribute ต้องอยู่ภายในเครื่องหมาย "'" หรือ "\"" หาก attribute ไม่ได้อยู่ในสัญลักษณ์ดังกล่าว เอกสารจะไม่ถือว่าเป็นแบบ well-formed
3. element ว่า หากต้องการใช้งาน element ว่า ต้องสร้าง element ว่าให้ถูกต้อง

เอกสาร XML ไม่จำเป็นต้องมี element ที่มี tag เปิดและ tag ปิดเสมอ เรายังสามารถสร้าง element ว่า หรือ Empty Element บรรจุลงในเอกสารได้ด้วย element ว่า จะต้องไม่บรรจุตัวอักษรใด ๆ ลงภายใน element ดังกล่าว สามารถเขียนได้ในสองรูปแบบ element ที่มี tag เปิดและ tag ปิด โดยที่ไม่มีข้อความ หรือ Content อยู่ระหว่าง tag (ยกเว้นตัวอักษรขึ้นบรรทัดใหม่)

```
<book></book>
```

element เฉพาะ ที่เรียกว่า element ว่าตามรูปแบบ <.../>

```
<book/>
```

element สามารถบรรจุข้อมูล attribute ได้ด้วย ดังตัวอย่างต่อไปนี้

```
<book author="baldwin" price="$9.95" />
```

4. ต้องไม่มีตัวอักษรประเภท Markup อยู่ในส่วน Content คือข้อมูล < หรือ &

เนื่องจากตัวอักษร < ถูกใช้สำหรับการสร้าง tag และ & ถูกใช้สำหรับการแสดงสัญลักษณ์พิเศษ

หากต้องการแสดงข้อมูลตัวอักษรดังกล่าว ในส่วน Content จริงๆ ให้ใช้สัญลักษณ์ ใช้งาน < สำหรับแสดงตัวอักษร < และ & สำหรับแสดงตัวอักษร &

5. Element ต้องซ้อนกันอย่างมีรูปแบบ หากมีการสร้างเอกสาร XML ที่มีการวางซ้อนกันของ element ในหลายชั้น การวางซ้อนกันต้องมีรูปแบบ เป็นชั้นๆ ห้ามวาง element ในลักษณะที่ซ้อนไขว้กัน

```
<book>
<chapter number="1">
<paragraph>
</paragraph>
</chapter>
</book>
```

ตัวอย่างด้านบนเป็นเอกสารที่ element วางซ้อนกันอย่างถูกต้อง

```
<book>
<chapter number="1">
<paragraph>
</chapter>
</paragraph>
</book>
```

ตัวอย่างด้านบนเป็นเอกสารที่วางซ้อน element อย่างไม่ถูกต้อง

Other XML Specifications

DOM และ SAX

ในกระบวนการนำข้อมูล XML มาใช้งานใน Application นั้น จะมี XML Parser เป็นตัวกลางในการดึงข้อมูลจากเอกสาร XML และ Application ซึ่งเป็น API ชนิดหนึ่ง โดย API ที่นิยมกันมากคือ DOM และ SAX ซึ่งต่างก็มีวิธีการดึงข้อมูลที่แตกต่างกันคือ

DOM จะมองเอกสาร XML ในลักษณะของโครงสร้างต้นไม้ (Tree)

DOM ย่อมาจาก Document Object Model โดยมีหลักการในการอ่านเอกสาร XML มาวางเป็น Tree ในหน่วยความจำของเครื่องที่กำลังทำงาน ประกอบด้วย Element หรือ Attribute ต่างๆ การเข้าถึงข้อมูลจึงเป็นการเดินไปตามกิ่งก้านต่างๆ ทั้งเป็นแบบต่อเนื่องไปเรื่อยๆ หรือจะอ้างอิงถึงก้านเฉพาะเจาะจงลงไป หรือ Random access ก็ได้

ข้อจำกัดของ DOM ก็อยู่ตรงปริมาณหน่วยความจำของเครื่องว่าจะสามารถรองรับข้อมูลได้ ใหญ่แค่ไหน เพราะวิธี DOM จะทำการอ่านข้อมูลทั้งหมดมาเก็บไว้ในหน่วยความจำเพียงครั้งเดียว แต่ข้อดีคือ เขียน Code ง่ายกว่ามาก

SAX จัดการเอกสาร XML ด้วยแนวทาง Event-Driven

Simple API for XML คือชื่อย่อของ SAX โดย SAX จะไม่ load ข้อมูลทั้งหมดเข้ามาในหน่วยความจำ แต่จะอ่านเอกสารจาก Disk ตั้งแต่เริ่มต้นไล่ไปเรื่อยๆ แล้วจะมีการสร้าง Event ออกมา เช่น เปิด Element ปิด Element หรือเจอ Attribute เป็นต้น คนเขียนโปรแกรมก็ต้องทำหน้าที่ ดักจับ Event เหล่านี้มาจัดการกับข้อมูลต่อ เช่นเจอ Event เมื่อถึง Element ชื่อ X ก็ต้องทำหน้าที่ในการดึง X มาใช้งาน เป็นต้น

สรุปว่าจะใช้งาน SAX เหมาะสำหรับการค้นหาข้อมูลเล็กน้อยบางอย่าง ซึ่งหากใช้ DOM ก็ต้อง Load ข้อมูลสัก 100 MB คงไม่คุ้ม แต่ถ้าต้องการ Random access และต้องเปลี่ยนแปลงแก้ไขข้อมูลก็คงต้องหันกลับไปใช้ DOM แนะนำว่า หากข้อมูลคุณอยู่แถวๆ 20 MB ก็คงยังพออยู่ได้ไปกับ DOM ได้ ถ้ามากกว่านี้ ตอน Load เอกสารก็อาจจะอืดหน่อย และขึ้นอยู่กับปริมาณหน่วยความจำของเครื่องด้วยนะครับ

XSLT

ภาษาไว้ใช้แปลง XML

เมื่อเรารู้ขั้นตอนวิธีในการใช้ XML เพื่อการจัดเก็บหรือ Mark Up ข้อมูลของเราเรียบร้อยแล้ว ตามที่เราต้องการ เพราะเราสามารถกำหนด Tag ต่างๆ ได้ตามใจชอบ เราก็จะกลายเป็นผู้รับผิดชอบ ควบคุมการใช้งานข้อมูลของเราเอง ใครจะมาขอเปลี่ยนชื่อ Element เราก็คงไม่ยอม และไม่อยากจะไปเกี่ยวข้องกับใคร และคงไม่อยากจะให้ใครรู้แทนที่ว่าเมื่อก่อน ลักษณะของการเก็บข้อมูลนั้นขึ้นอยู่กับเจ้าของโปรแกรมว่ากำหนดมาอย่างไร เช่นเราเขียนโปรแกรมใช้งาน Microsoft Access ไฟล์ ใครรู้บ้างว่าไฟล์ MDB นั้นมีโครงสร้างเป็นอย่างไร ซึ่งก็คงเป็นความลับที่ Microsoft คงไม่บอก

แต่ด้วยเทคโนโลยี XML ทำให้เราสามารถพัฒนารูปแบบการเก็บข้อมูลของเรา กำหนด Tag ต่างๆ ได้เอง ต่างคนก็ต่างกำหนดตามใจชอบ ไม่ว่าจะเป็นองค์กรต่างๆ บริษัทห้างร้านไหนๆ หรือแม้แต่ W3C เอง สิ่งเหล่านี้คือที่มาของความจำเป็นในการที่จะต้องแปลงลักษณะข้อมูลรูปแบบ XML แบบหนึ่งไปเป็นอีกแบบหนึ่ง

ทาง W3C จึงได้แนะนำให้ใช้ XSLT (Extensible Stylesheet Language Transformation) และก็ตัว XPath (XML Path Language) ซึ่งถือเป็นเครื่องมืออันทรงพลังที่จะใช้แปลงรูปแบบข้อมูล XML ให้อยู่ในรูปแบบต่างๆ เช่น HTML

หรือ XML หรือรูปแบบใดๆ ก็ได้ เราจึงต้องใช้ XSLT ซึ่งตัวมันเองใช้ XPath เพื่อจะสร้างผลลัพธ์ที่ต้องการจากข้อมูล mark-up ที่เป็น XML

XLINK

เป็น แอปพลิเคชันตัวหนึ่งของ XML ที่มีความสามารถในการทำลิงค์ให้กับเอกสาร XML ได้เช่นเดียวกับการลิงค์ที่เราเคยชินใน HTML แต่มีความแตกต่างตรงความสามารถที่มากกว่า HTML ของ Xlink คือ

- สามารถทำลิงค์ไปยังทรัพยากรได้มากกว่าหนึ่ง
- สามารถทำลิงค์ในเอกสารที่ไม่สามารถลิงค์กับข้อความได้
- เป็นลิงค์ที่ชาญฉลาดสามารถประยุกต์ใช้ได้อย่างไม่จำกัด และสามารถกรองเฉพาะรายการลิงค์ที่ต้องการได้
- สามารถลิงค์ได้ตามรูปแบบของข้อมูลที่ไม่สามารถสนับสนุนการทำลิงค์ เช่น รูปภาพ เป็นต้น

การสร้างลิงค์ด้วย Xlink มี 2 ชนิด คือ ลิงค์พื้นฐาน(simple link) และลิงค์เพิ่มขยาย(Extended Link) ซึ่งได้รับการออกแบบเพื่อมาปรับปรุงการสร้างลิงค์ของ HTML และจัดเตรียมพลังความสามารถที่สูงกว่า และขยายความสามารถได้ตามความต้องการซึ่งผู้สร้าง XLink ได้สร้างขึ้นตามเป้าหมายต่อไปนี้

1 Xlink ควรจะสามารถนำมาใช้บนอินเทอร์เน็ตได้โดยตรง เป้าประสงค์นี้เกี่ยวพันโดยตรงกับแนวคิดลิงค์พื้นฐานที่มีอยู่ใน XML และยังสนับสนุนแนวคิดการใช้งาน Xlink บน อินเทอร์เน็ตได้โดยง่าย โดยไม่คำนึงว่ามันจะนำความสามารถใหม่มาใช้

2 Xlink ควรจะสามารถนำไปใช้ได้อย่างหลากหลาย ทั้งลิงค์ภายในโดเมนและลิงค์ภายในแอปพลิเคชันซอฟต์แวร์ของคลาส เป้าประสงค์นี้แสดงความเป็นไปได้ที่ Xlink นี้สามารถขยายความสามารถเกินกว่าที่จะใช้กับเว็บแอปพลิเคชัน ดังนั้นในจึงต้องยืดหยุ่นเพียงพอที่จะนำมาใช้ทำงานภายในแอปพลิเคชันของซอฟต์แวร์อันหลากหลาย

3 ภาษาที่ใช้กับ Xlink ควรเป็น XML Xlink เป็นแอปพลิเคชันตัวหนึ่งของ XML เป้าประสงค์นี้ไม่ได้ต้องการสร้างภาษาขึ้นมาใหม่ แต่ต้องการสร้าง Vocabulary ของ XML ด้วยการใช้ภาษา XML ซึ่ง Xlink นำมาใช้ในการสร้างโครงสร้างและรูปแบบ และไม่ต้องให้ผู้สร้างหรือผู้ออกแบบแอปพลิเคชันซอฟต์แวร์ต้องเรียนรู้ภาษาใหม่เพิ่มขึ้นมาอีก

4 การออกแบบ Xlink ควรสามารถเตรียมการได้อย่างรวดเร็ว เป็นความจริงที่ตัวผู้สร้าง Xlink ขึ้นมาเองก็เข้าใจว่า ถ้าหากมันไม่สามารถทำงานได้เร็วแล้ว องค์กรหรือผู้ผลิตรายอื่นก็ต้องหาวิธีการสร้างลิงค์ของตัวเองขึ้นมา ซึ่งจะอุปสรรคต่อการทำให้เป็นมาตรฐานเปิดของภาษา XML

5 การออกแบบ Xlink ต้องมีรูปแบบเฉพาะและรัดกุม เป้าประสงค์นี้เป็นไปตามเป้าประสงค์ของภาษา XML ซึ่งผู้ออกแบบรู้ว่าถ้านำมาใช้ในแวดวงอุตสาหกรรมแล้ว ต้องทำให้มันง่ายต่อความเข้าใจและการใช้งาน ดังนั้นผู้ออกแบบจึงตัดส่วนที่ไม่มีประโยชน์ทิ้งไป โดยเน้นเฉพาะส่วนที่จำเป็นต้องมี และทำให้ขนาดเล็กที่สุดที่เป็นไปได้

6 Xlink ควรเป็นภาษาที่มนุษย์อ่านได้ สาเหตุที่ว่า Xlink ถูกสร้างขึ้นมาด้วยการใช้ XML ดังนั้นมันควรง่ายต่อการทำความเข้าใจ นั่นคือ Xlink ควรมีความง่ายเพียงพอที่มนุษย์จะสามารถอ่านเข้าใจได้ และควรอ่านได้โดยไม่ต้องมีการถอดรหัสที่ต้องอาศัยความช่วยเหลือจากเครื่องคอมพิวเตอร์ เพื่อให้เข้าใจได้ดีขึ้น

7 Xlink อาจจะอยู่ภายนอกเอกสารในที่มีทรัพยากรที่ต้องนำมาใช้ร่วมอยู่ เป้าประสงค์นี้เกี่ยวพันโดยตรงกับการขยายขีดความสามารถของภาษา Xlink ซึ่งเห็นได้จาก Xlink ได้ก้าวข้ามการสร้างลิงค์แบบทางเดียวของ HYML และมีความสามารถจัดเตรียมลิงค์จากจุดหนึ่งไปยังหลายๆจุดที่เป็นทรัพยากร ดังนั้นจึงเป็นเป้าประสงค์สำคัญอันหนึ่งที่ผู้ออกแบบ Xlink ควรจดจำ

8 Xlink ควรจะแทนโครงสร้างและความหมายของลิงค์ ผู้ออกแบบไม่เพียงแต่ต้องจัดเตรียม Xlink เป็นเครื่องมือในการสร้างลิงค์ XML แต่ลิงค์ที่ใช้ Xlink ยังต้องสามารถนิยามความหมายของลิงค์ และแสดงความสัมพันธ์ของอ็อบเจกต์ หรือทรัพยากรที่มันติดต่อกันอยู่ เพื่อจัดเตรียมเป็นข้อมูลให้กับผู้ใช้ ตัวอย่างเช่น ทรัพยากรที่ลิงค์ชี้

9 Xlink ต้องใช้งานได้ เนื่องจาก Xlink ต้องอ่านเข้าใจได้ง่ายโดยมนุษย์ และง่ายต่อผู้สร้าง นอกจากนี้ยังต้องนำมาใช้งานได้ง่ายต่อการพัฒนาแอปพลิเคชัน ผู้ออกแบบภาษาจึงต้องทำให้สามารถใช้งานได้ในวงกว้างและง่ายต่อการนำไปใช้ ดังนั้นแอปพลิเคชันใช้ Xlink เพิ่มขึ้นเท่าใด ก็ยิ่งเข้าใจเป้าประสงค์มากขึ้นเท่านั้น

XML Schema

บ่อยครั้งที่มีเอกสารซึ่งประกอบด้วยเนื้อหาที่มีชนิดต่างกัน และต้องการตรวจสอบ Data Type เหล่านี้ได้ แต่ DTD ไม่ได้ถูกออกแบบมาเพื่อตรวจสอบ Data Type เหล่านี้หรือการตรวจสอบของเซตของค่า(Value) DTD ยังไม่เข้าใจ Namespace อีกด้วย

Schema สร้างขึ้นมาเพื่อแก้ปัญหาเหล่านี้ Schema ต่างจาก DTD ตรงที่มีรูปประโยค (Syntax) เป็นของตัวเอง ส่วน XML Schema นั้นถูกเขียนขึ้นใน XML นอกจากการจัดสร้างข้อมูลที่ DTD นำเสนอแล้ว Schema ยังช่วยกำหนด Data Type ใช้ Namespace และกำหนดช่วงค่าของแอตทริบิวต์และเอเลเมนต์

Data Type ของ Schema แบบธรรมดา

โดยส่วนใหญ่เอกสาร XML แบ่งออกได้เป็น 2 ชนิด คือ Document-Oriented และ Data-Oriented เอกสาร XML ที่เป็น Document-Oriented ประกอบด้วยส่วนของข้อความที่ถูกนำมารวมกับข้อมูลในฟิลด์ ในขณะที่เอกสาร XML ที่เป็น Data -Oriented ประกอบด้วยข้อมูลในฟิลด์เท่านั้น

ตัวอย่างเอกสารที่เป็น Document-Oriented คือ Message เช่นที่แสดงข้างล่างนี้

```
<message priority="high" date="2000-01-11">
```



```
<from>Jake Sturm</from>
<to>Gwen Sturm</to>
<subject>DNA Course</subject>
<body>
The new DNA course that we are offering is now complete. It will provide a complete
overview discussion of designing and building DNA systems, including DNS, DNA,
COM, and COM+. The course is also listed on the Web site, at http://ies.gti.net.
</body>
</message>
```

Message นี้มีเนื้อหาส่วนที่เป็นข้อความที่ใหญ่มาก แต่ยังคงมีแอตทริบิวต์ไว้ด้วยคือ date และ priority ซึ่ง date Attribute มี Data Type เป็น Date (วันที่) และ priority Attribute มี Data Type เป็น enumerated ซึ่งมีประโยชน์มากในการตรวจสอบแอตทริบิวต์เหล่านี้ว่ามีรูปแบบที่ถูกต้องสำหรับ Data Type ทั้งสองหรือไม่ โดย Schema ยอมให้คุณทำสิ่งเหล่านี้ได้

เอกสารที่เป็น Data-Oriented มีลักษณะดังนี้

```
<bill>
<OrderDate>2001-02-11</OrderDate>
<ShipDate>2001-02-12</ShipDate>
<BillingAddress>
<name>John Doe</name>
<street>123 Main St.</street>
<city>Anytown</city>
<state>NY</state>
<zip>12345-0000</zip>
</BillingAddress>
<Voice>555-1234</Voice>
<fax>555-5678</fax>
</bill>
```

เอกสารทั้งหมดนี้ประกอบด้วยฟิลด์ข้อมูลที่ต้องใช้ในการตรวจสอบ การตรวจสอบฟิลด์ข้อมูลที่เป็นสิ่งที่จำเป็นสำหรับเอกสาร XML รูปแบบนี้

ในจุดนี้เราจะดูเฉพาะเอกสาร XML ที่เป็น Document-Oriented ซึ่งประกอบด้วย Data-Type ชนิดเดียวเท่านั้น (รูปแบบสตริง) เพราะว่า DTD ทำงานได้ดีกับเอกสาร XML ที่เป็น Document-Oriented ซึ่งบรรจุเฉพาะ Data Type แบบสตริงเท่านั้น เนื่องจาก Schema ยอมให้คุณตรวจสอบข้อมูลของ Data Type ในตอนนี้ให้คุณดูชนิดข้อมูลที่ถูกในข้อกำหนดของ Schema

คำว่า Data Type ในมาตรฐาน Schema ที่สอง ซึ่งคุณสามารถพบได้ที่ <http://www.w3.org/TR/xmlschema-2/> โดย Data Type แทนที่ชนิดของข้อมูลเช่น String, Integer และอื่นๆ เป็นต้น มาตรฐานของ Schema ที่สองจะกำหนด Simple Data Type ในรายละเอียด ซึ่งเป็นสิ่งที่เรากล่าวถึงในบทนี้

องค์ประกอบด้านประเภทข้อมูลของ Schema

ใน Schema นั้น Data Type มีส่วนประกอบ 3 ส่วนคือ Value Space, Lexical Space และ Facet ซึ่ง Value Space คือช่วงค่าของ Data Type ที่ยอมรับได้ Lexical Space เป็นชุดของ Literal ที่ถูกต้องซึ่งแทนวิธีการที่ Data Type สามารถแสดงผลได้ เช่น 100 และ 1.0E2 เป็น Literal สองแบบที่แตกต่างกัน แต่ทั้งคู่แสดงค่า Floating Point เหมือนกัน Facet คือลักษณะพิเศษของประเภทข้อมูล ซึ่ง Data Type อาจประกอบด้วย Facet จำนวนมาก โดยแต่ละตัวจะกำหนดหนึ่งลักษณะพิเศษตั้งแต่หนึ่งหรือมากกว่า Facet กำหนดวิธีที่ Data Type หนึ่งแตกต่างจาก Data Type อื่น ๆ Facet จะระบุ Value Space ของ Data Type

มี Facet อยู่สองชนิดคือ Fundamental และ Constraining ซึ่ง Fundamental Facet ทำหน้าที่กำหนด Data Type ส่วน Constraining Facet จะวางข้อกำหนดของ Data Type ตัวอย่างของ Fundamental Facet คือกฎที่กำหนดลำดับของอเลเมนต์ คือค่าสูงสุดหรือต่ำสุดที่ยอมรับให้เป็นไปได้ กำหนดลักษณะดั้งเดิมของ Data Type ว่าจำกัดหรือไม่จำกัด และกำหนดว่าอินสแตนซ์ (Instance) ของ Data Type นั้นต้องเป็นเช่นนั้นหรือไปได้สุดพิสดาร และกำหนดว่า Data Type (จำนวนอักขระสำหรับข้อมูลสตริงหรือจำนวนบิตของ Data Type แบบไบนารี) ความยาวต่ำสุดและความยาวสูงสุด, Enumeration และ Pattern

เราสามารถจัดหมวดหมู่ของชนิดข้อมูลได้หลายมิติดังนี้ อันดับแรก Data Type สามารถเป็น Atomic หรือ Aggregate สำหรับ Data Type แบบ Atomic ไม่สามารถแบ่งเป็นส่วนๆ ได้ค่า Integer หรือ Date ที่ถูกนำมาใช้แทนค่าสตริงที่มีอักขระเดียวคือ Data Type แบบ Atomic ถ้า Date ถูกนำเสนอเป็นค่า Day, Month และ Year แล้ว Data Type Date จึงเป็น Data Type แบบ Aggregate

Data Type ยังสามารถจำแนกได้เป็น Primitive หรือ Generated ซึ่ง Primitive Data Type ไม่ได้รับการสืบทอดมาจาก Data Type อื่นๆ แต่เป็นนิยามที่ถูกกำหนดขึ้น Generated Data Type ถูกสร้างมาจาก Data Type ที่มีอยู่แล้ว ซึ่งเรียกว่า Basetypes ซึ่งสามารถเป็น Primitive หรือ Generated Data Type ก็ได้ Generated Type ซึ่งจะถูกระบุในหัวข้อต่อไปของบทนี้สามารถเป็น Data Type แบบ Simple หรือ Complex ก็ได้เช่นกัน

Primitive Data Type ประกอบด้วยค่าดังต่อไปนี้ คือ String, Boolean, Float, Decimal, Double, timeDuration, recurringDuration, Binary และ uri นอกจากนั้นยังมี Data Type แบบ timeInstant ที่ได้รับมาจาก Data Type แบบ recurringDuration ท่ามกลาง Data Type แบบ Primitive มีสองค่าที่ใช้กำหนดใน XML Schema คือ timeDuration และ recurringDuration สำหรับ Data Type แบบ timeInstant ยังสามารถกำหนดไปยัง XML

โดยเฉพาะ

Data Type แบบ `timeInstant` แทนการรวมกันของค่า Date และ Time ซึ่งนำเสนอข้อมูลของเวลาแบบเจาะจง (ดังโค้ด) แบบแผนของการ Data Type นี้คือ

`CCYY-MM-DDThh:mm:ss.sss`

CC แทนศตวรรษ, YY คือปี, MM คือเดือน และ DD คือวัน ซึ่งจะถูกเติมเข้าไปข้างหน้าด้วยสัญลักษณ์แสดงหัวข้อเพิ่มเติมเพื่อแสดงถึงหมายเลขจำนวนลบ ถ้าเครื่องหมายนี้ถูกตัดออกถ้าจะนั้นนิฐานเป็นเครื่องหมายบวกโดยปริยาย ตัวอักษร T คือตัวแยกระหว่าง Date และ Time ส่วน hh, mm และ ss.sss แทนค่าชั่วโมง นาที และวินาที การเพิ่มตัวเลขสามารถนำมาใช้เพื่อเพิ่มความแน่นอนของเศษวินาทีได้ถ้าต้องการ เพื่อจัดรูปแบบค่าปีที่มากกว่า 9999 เราสามารถเพิ่มค่าตัวเลขไปยังด้านซ้ายของการนำเสนอ

การนำเสนอ `timeInstant` สามารถทำได้โดยกำหนดค่า Z ตามหลังทันทีเพื่อแสดง Universal Time Coordinate (UTC) ข้อมูลเขตเวลา (Time Zone) จะแสดงค่าตามความแตกต่างระหว่างเวลาที่ท้องถิ่นและ UTC ที่คุณสามารถกำหนดได้ทันทีต่อจากเวลาที่ซึ่งประกอบด้วยเครื่องหมายบวกหรือลบ แล้วตามด้วย hh:mm

Data Type แบบ `timeDudation` แสดงบางช่วงของเวลารูปแบบสำหรับ `timeDuration` ได้ดังนี้

`PyYmMdDThHmMsS`

ค่า Y แทนจำนวนของปี, M คือจำนวนของเดือน, D คือจำนวนของวัน, T คือตัวแบ่งระหว่างวันที่กับเวลา, H คือจำนวนชั่วโมง, M คือจำนวนนาที และ S คือจำนวนวินาที ส่วนค่า P ที่เริ่มต้นบอกให้รู้ว่าประโยคนี้นั้นแทนรอบของเวลา จำนวนของวินาทีที่สามารถเป็นตัวเลขทศนิยมเพื่อความแม่นยำ และการนำหน้าด้วยเครื่องหมายลบจะอนุญาตให้แสดงช่วงเวลาที่เป็ลบ ถ้าไม่มีการกำหนดเครื่องหมายไว้จะสันนิษฐานว่าช่วงเวลาคือเป็นบวก

Data Type แบบ `recurringDuration` แทนช่วงเวลาที่เกิดซ้ำ แบบแผนสำหรับ `recurringDuration` คือการตัดด้านซ้ายของการนำเสนอ `timeInstant` เช่น ถ้าค่าศตวรรษ CC ถูกละจากการนำเสนอใน `timeInstant` ซึ่ง `timeInstant` เกิดขึ้นทุกๆ พันปีเช่นเดียวกันถ้าไม่มีการกำหนด CCYY คุณจะพบว่า `timeInstant` เกิดขึ้นทุกๆ ปี ทุกๆ 2 ตัวอักษรของการนำเสนอที่ตัดทอนไปแสดงโดยเครื่องหมาย Hyphen อันเดียว เช่น เพื่อแสดงเวลา 1:20 PM ในวันที่ 31 เดือนพฤษภาคมของทุกๆ ปี ตามมาตรฐานเวลาฝั่งตะวันออก (Eastern Standard Time) ซึ่งจะช้ากว่าเวลา UTC 5 ชั่วโมง สามารถเขียนโค้ดได้ดังนี้

`--05-31T13:20:00-05:00`

สรุปประโยชน์จาก XML

สำหรับประโยชน์ของ XML นั้น เป็นด้านความยืดหยุ่นในการใช้งานสำหรับแอปพลิเคชันที่อิงกับ Web Base ที่ใช้ง่ายในการค้นหาข้อมูล มีความยืดหยุ่นในการพัฒนาเว็บ สามารถผสมผสานข้อมูลจากหลายแหล่ง จากแอปพลิเคชันที่ต่างกัน สามารถแสดงข้อมูลแบบต่างๆ และสามารถ update ข้อมูลให้ทันสมัยเสมอ และคาดว่าจะเป็มาตรฐานใหม่ของระบบเปิด ซึ่งนับเป็น format ใหม่สำหรับการส่งข้อมูลบนเว็บที่มากด้วยข้อมูลหลายแบบ แต่ส่งผ่านด้วยเทคโนโลยีที่บีบอัดข้อมูลที่ให้ความเร็วได้รับการสนับสนุนจากผลิตภัณฑ์ค่ายไมโครซอฟท์

Reference

<http://www.w3.org/TR/xml11>

<http://www.thaixml.com/>

การเขียนโปรแกรม XML และ SOAP สำหรับ BizTalk Server ,กิตติชัย ฟินิจดำ แปล

เข้าใจและใช้งานภาษา XML ฉบับโปรแกรมเมอร์ ,ศุภชัย สมพิท