ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Εισαγωγή στην Επιστήμη και Τεχνολογία των Υπηρεσιών

Ενότητα 5: Schema - 2

Χρήστος Νικολάου
Τμήμα Επιστήμης Υπολογιστών

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται στην άδεια χρήσης **Creative Commons** και ειδικότερα

Αναφορά – Μη εμπορική Χρήση – Όχι Παράγωγο Έργο v. 3.0

(Attribution – Non Commercial – Non-derivatives)

- Εξαιρείται από την ως άνω άδεια υλικό που περιλαμβάνεται στις διαφάνειες του μαθήματος, και υπόκειται σε άλλου τύπου άδεια χρήσης. Η άδεια χρήσης στην οποία υπόκειται το υλικό αυτό αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

XML
Schema - 2
605.444 / 635.444

David Silberberg

Lecture 6

Global Definitions

- Elements can be reused multiple times
- Sometimes, it is worthwhile to declare it one and reuse it
- Suppose there is a comment field that is used in multiple places
- Global comment field is defined at the top level of the Schema

```
<schema …>
...
<element name="comment" type="string" />
…
</schema>
```

Global Definitions Embedded in Schema

- Use a **ref** attribute in the element definition
- Other attributes for the elements can still be defined

```
<complexType name="NameType">
    <sequence>
        <element name="first" type="string" />
        <element name="middle" type="string" />
        <element name="last" type="string" />
        <element ref="comment" minOccurs="0" />
    </sequence>
    <attribute name="title" type="string" />
</complexType>
```

Naming Conflicts

- Conflicts
  - Two types with the same name
  - Even a simpleType and complexType conflict
- No conflict
  - Type and element
  - Type and attribute
  - Elements within different types (e.g., name in both customer and item)
  - Type in XML Schema and your own defined type
    - Namespaces resolve conflicts

Anonymous Type Definition

- Types are named if they are to be reused
- It is not necessary to name a type if it is to be used once
  - For example, anonymous classes can be defined in Java
  - Similarly, in Java, each layer of a multi-layer data structure does not need to be typed
- It does not matter either way if types are defined or are anonymous
- Anonymous type definitions are identified by the lack of the **type** attribute in the element definition

Anonymous Type Example

- Instead of NameType:

```
<element name="name">
    <complexType>
        <sequence>
            <element name="first"     type="store:ProperName" />
            <element name="middle" type="store:ProperName"
                    minOccurs="0" maxOccurs="unbounded"/>
            <element name="last" type="store:ProperName" />
        </sequence>
        <attribute name="title" type="store:Title"  use="optional"
            default="Ms." />
    </complexType>
</element>
```

Mixed Content

- XML Schema enables
  - Definition of text embedded in elements
  - Definition of text that surrounds element values
- Example XML file

```
<letter>
    Sample letter:
    <salutation>
        Dear Mr. <name>Fred Jones</name>:
    </salutation>
    <body>
        I often enjoy reading your editorial in the <newspaper>New York
        Times</newpaper>. However, your editorial yesterday …
    </body>
</letter>
```

Mixed Content Definition

- Use the **mixed** attribute of the type declaration

```
<element name="letter">
    <complexType mixed="true">
        <sequence>
            <element name="salutation">
                <complexType mixed="true">
                    <sequence>
                        <element name="name" type="string" />
                    </sequence>
                </complexType>
            </element>
```

Mixed Context Definition (2)

```
                <element name="body">
                    <complexType mixed="true">
                        <sequence>
                                <element name="newspaper" type="string" />
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </complexType>
</element>
```

Empty Complex Type

- Schema provides a mechanism for defining empty complex types
  - No elements
  - Attributes only
- Example - purchase type

```
<element name="purchase" minOccurs="0" maxOccurs="unbounded">
    <complexType>
        <attribute name="date" type="date" />
        <attribute name="items" type="IDREFS" />
        <attribute name="qty" type="store:IntegerList" />
    </complexType>
</element>
```

Complex Types From Simple Types

- Declaring an element that has an attribute and contains a simple value.

  <internationalPrice currency="EUR">423.46</internationalPrice>

- Create a <complexType> that uses a <simpleContent>

```
<element name="internationalPrice">
    <complexType>
        <simpleContent>
            <extension base="decimal">
                <attribute name="currency" type="string"/>
            </extension>
        </simpleContent>
    </complexType>
</element>
```

Unrestricted Type

- anyType
  - All XML types inherit from the base type: anyType
  - No restrictions or constraints on the content data
  - By default, the type is anyType
  - The following are equivalent

```
<element name="anything" type="anyType" />

<element name="anything" />
```

Annotations

- XML Schema drops comments from the definition
- Annotations provide a mechanism for maintaining comments through the processing of the Schema file
- <annotation> … </annotation> wraps annotation elements
  - <documentation> … </documentation> maintains any free text associated with the annotation
  - <appInfo> … </appInfo> provides a mechanism for passing annotation information to the parser in a machine readable way

```
<annotation>
    <documentation>Only high-quality items are sold in this store.</documentation>
</annotation>
```

Groups

- Enables the definition of element and attribute groups
- Groups are referenced by other Schema constructs

```
<group name="USAddressGroup">
        <sequence>
                <element name="state" type="store:USState" />
                <element name="zip" type="store:ZipCode" />
        </sequence>
</group>

<attributeGroup name="PurchaseGroup">
        <attribute name="date" type="date" />
        <attribute name="items" type="IDREFS" />
        <attribute name="qty" type="store:IntegerList" />
</attributeGroup>
```

Referencing Groups

- Groups are referenced with the **group** or **attributeGroup** XML Schema attributes

```
<group ref="store:USAddressGroup" />

<element name="purchase" minOccurs="0" maxOccurs="unbounded">
    <attributeGroup ref="PurchaseGroup" />
</element>
```

Choice Groups

- Enables the definition of multiple choices of elements and attributes
- The store schema allows both US and foreign addresses

```
<complexType name="AddressType">
    <sequence>
        <element name="street" type="store:MixedName" />
        <element name="city" type="store:ProperName" />
        <choice>
            <element name="country" type="store:ProperName" />
            <group ref="store:USAddressGroup" />
        </choice>
    </sequence>
</complexType>
```

All Group

- Enables elements to be listed in any order
- However, each element must have a minOccurs = 0 or 1 and a maxOccurs = 1

```
<complexType name="ItemType">
        <all>
                <element name="description" type="string" />
                <element name="in_stock" type="nonNegativeInteger" />
                <element name="price" type="decimal" />
                <element name="cost" type="decimal" />
        </all>
        <attribute name="item_no" type="ID" />
        <attribute name="supplier_id" type="IDREF" />
</complexType>
```

Import and Include

- Include
  - Includes schemas with no target namespace or the same target namespace
  - Example:

    <include schemaLocation="included_schema.xsd"/>
- Import
  - Imports schemas with different target namespaces
  - Example:

    <import
    namespace="http://www.bank.org/some_namespace"
    schemaLocation=
    "http://www.bank.org/some_namespace/imported_schema.xsd">

Defining the Schema Element

- Schema element defines
  - XML Schema Namespace
    - XML simple types
    - XML elements
    - If not default, namespace must be specified by all types and elements
  - Namespace of the current Schema document
    - Enables references to new types defined in current document
    - If not default, namespace must be specified by all types and elements

Defining the Schema Element (cont.)

- Schema element defines (cont.)
  - Target Namespace
    - Defines the resulting namespace created by this Schema
  - Form defaults
    - Whether or not elements and/or attributes must be qualified with the Namespace of the XML Schema definition document
    - Global elements still need to be qualified even though "unqualified" is specified
    - If "unqualified" is specified, elements and/or attributes **cannot** be qualified

Schema Element Example

- Schema Definition

```
<schema
    xmlns="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    targetNamespace="http://www.apl.jhu.edu/%7Edavids/605742/"
    xmlns:store="http://www.apl.jhu.edu/%7Edavids/605742/">
```

works with this

- XML Document definition

```
<store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://www.apl.jhu.edu/%7Edavids/605742/"
        xsi:schemaLocation="http://www.apl.jhu.edu/%7Edavids/605742/ store.xsd">
    <cust_list>
        <customer cid="C-1" >
        …
    </supplier_list>
</store>
```

Qualified Elements and Attributes

- Schema Definition

```
<schema
    xmlns="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.apl.jhu.edu/%7Edavids/605742/"
    xmlns:store="http://www.apl.jhu.edu/%7Edavids/605742/"
    elementFormDefault="qualified"
    attributeFormDefault="qualified">
…
```

- XML Document definition

```
<st:store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:st="http://www.apl.jhu.edu/%7Edavids/605742/"
        xsi:schemaLocation="http://www.apl.jhu.edu/%7Edavids/605742/ store.xsd">
      <st:cust_list>
            <st:customer st:cid="C-1" >
            …
      </st:supplier_list>
</st:store>
```

The Store Schema (1)

```
<?xml version="1.0" ?>
<schema
      xmlns="http://www.w3.org/2001/XMLSchema"
      elementFormDefault="qualified"
      targetNamespace="http://www.apl.jhu.edu/%7Edavids/605742/"
      xmlns:store="http://www.apl.jhu.edu/%7Edavids/605742/">

      <element name="store" type="store:StoreType"/>
      <simpleType name="ProperName">
            <restriction base="string">
                  <pattern value="[A-Z][a-zA-Z]*"/>
            </restriction>
      </simpleType>

      <simpleType name="MixedName">
            <restriction base="string">
                  <pattern value="[\sa-zA-Z0-9\-]*"/>
            </restriction>
      </simpleType>
```

The Store Schema (2)

```
<complexType name="StoreType">
    <all>
        <element name="cust_list" type="store:CustListType" minOccurs="0"/>
        <element name="inventory_list" type="store:InventoryListType"
                  minOccurs="0"/>
        <element name="supplier_list" type="store:SupplierListType" minOccurs="0"/>
    </all>
</complexType>


<complexType name="CustListType">
    <sequence>
        <element name="customer" type="store:CustomerType" minOccurs="0"
                  maxOccurs="unbounded"/>
    </sequence>
</complexType>
```

The Store Schema (3)

```
<complexType name="CustomerType">
   <sequence>

     <element name="name">
       <complexType>

          <sequence>
             <element name="first" type="store:ProperName"/>
             <element name="middle" type="store:ProperName" minOccurs="0"
                        maxOccurs="unbounded"/>
             <element name="last" type="store:ProperName"/>
          </sequence>

          <attribute name="title" type="store:Title" use="optional" default="Ms."/>

       </complexType>
     </element>
```

The Store Schema (4)

```
    <element name="address" type="store:AddressType"/>
    <element name="purchase" minOccurs="0" maxOccurs="unbounded">
      <complexType>
          <attribute name="date" type="date"/>
          <attribute name="items" type="IDREFS"/>
          <attribute name="qty" type="store:IntegerList"/>
      </complexType>
    </element>
  </sequence>

  <attribute name="cid" type="ID"/>
  <attribute name="ctype" type="string" use="optional" default="good"/>

</complexType>
```

The Store Schema (5)

```
<simpleType name="IntegerList">
        <list itemType="integer"/>
</simpleType>

<simpleType name="Title">
        <restriction base="string">
                <enumeration value="Mr."/>
                <enumeration value="Ms."/>
                <enumeration value="Mrs."/>
                <enumeration value="Miss"/>
                <enumeration value="Dr."/>
        </restriction>
</simpleType>
```

The Store Schema (6)

```
<group name="USAddressGroup">
  <sequence>
    <element name="state" type="store:USState"/>
    <element name="zip" type="store:ZipCode"/>
  </sequence>
</group>

<complexType name="AddressType">
  <sequence>
    <element name="street" type="store:MixedName"/>
    <element name="city" type="store:ProperName"/>
    <choice>
        <element name="country" type="store:ProperName"/>
        <group ref="store:USAddressGroup"/>
    </choice>
  </sequence>
</complexType>
```

The Store Schema (7)

```
<simpleType name="ZipCode">
        <restriction base="string">
                <pattern value="[0-9]{5}"/>
        </restriction>
</simpleType>

<simpleType name="USState">
        <restriction base="string">
                <enumeration value="AK"/>
                <enumeration value="AL"/>
                <enumeration value="MD"/>
        </restriction>
</simpleType>
```

The Store Schema (8)

```
<complexType name="InventoryListType">
  <sequence>
    <element name="item" type="store:ItemType" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
</complexType>

<complexType name="ItemType">
  <all>
    <element name="description" type="string"/>
    <element name="in_stock" type="nonNegativeInteger"/>
    <element name="price" type="decimal"/>
    <element name="cost" type="decimal"/>
  </all>
  <attribute name="item_no" type="ID"/>
  <attribute name="supplier_id" type="IDREF"/>
</complexType>
```

The Store Schema (9)

```
<simpleType name="StdTelNum">
    <restriction base="string">
        <pattern value="\d{1}\-\d{3}\-\d{3}\-\d{4}"/>
    </restriction>
</simpleType>

<simpleType name="StringTelNum">
    <restriction base="string">
        <pattern value="\d{1}-\d{3}\-[A-Z0-9]{7}"/>
    </restriction>
</simpleType>

<simpleType name="AllTelNum">
    <union memberTypes="store:StdTelNum store:StringTelNum"/>
</simpleType>
```

The Store Schema (10)

```
    <complexType name="SupplierListType">
        <sequence>

            <element name="supplier" minOccurs="0" maxOccurs="unbounded">
                <complexType>
                    <sequence>
                        <element name="company" type="store:MixedName"/>
                        <element name="telephone" type="store:AllTelNum"/>
                    </sequence>
                    <attribute name="sid" type="ID"/>
                </complexType>
            </element>

        </sequence>
    </complexType>

</schema>
```

Unique References

- Contents of elements, attributes, and combinations of elements and attributes can be specified to be unique
- Uniqueness can be
  - With respect to the entire document
  - With respect to some specific or wildcard XPath expression
  - XPath will be covered in more detail later in the course
- Example:

```
<unique name="FullName>
        <selector xpath="/store/cust_list/customer/name" />
        <field xpath="first" />
        <field xpath="last" />
</unique>
```

Full Document Uniqueness

- Uniqueness can be imposed on elements and/or attributes across the entire document
- For example, all **id** attribute values must be unique

```
<unique name="IDUnique">
    <selector xpath=".//*" />
    <field xpath="@id" />
</unique>
```

Document Keys

- Keys may be defined for attributes and/or elements
- Key references (to keys) can be defined, as well

```
<key name="SupplierKey">
        <selector xpath="/store/supplier_list/supplier" />
        <field xpath="@sid"/>
</key>
<keyref name="SupplierRef" ref="SupplierKey">
        <selector xpath=".//item" />
        <field xpath="@supplier_id" />
</keyref>
```

Definition in XML Schema

```
<key name="SupplierKey">
        <selector xpath="/store:store/supplier_list/supplier" />
        <field xpath="@sid"/>
</key>
<complexType name="ItemType">
        ...
        <attribute name="supplier_id" type="IDREF" >
                <keyref name="SupplierRef" ref="store:SupplierKey">
                        <selector xpath=".//item" />
                        <field xpath="@supplier_id" />
                </keyref>
        </attribute>
</complexType>
```

Τέλος Ενότητας