메타데이터와 XML 스키마

2002. 8. 26.

인천대학교 컴퓨터공학과
채 진 석

1

Terminology Clarification

- Schema
  - Refers to the W3C XML Schema
- Schemas
  - Refers to one or more industry standards
- schema (lowercase)
  - Refers to an information model
  - Can be a DTD or Schema
  - May also be a relational database schema

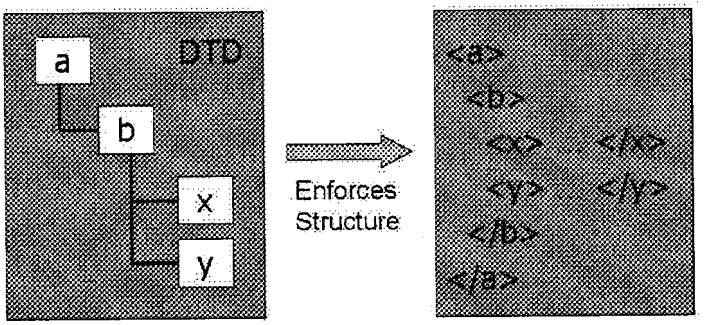
General Concepts

DTD and XML Schema

3

Document Type Definition

- An information modeling syntax

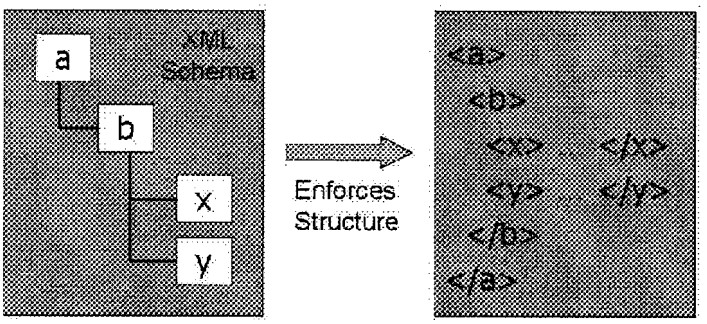


<!ELEMENT x (#PCDATA)>
<!ELEMENT y (#PCDATA)>

x and y are always interpreted as String, regardless of how they are used.

XML Schema

- An information modeling syntax in XML

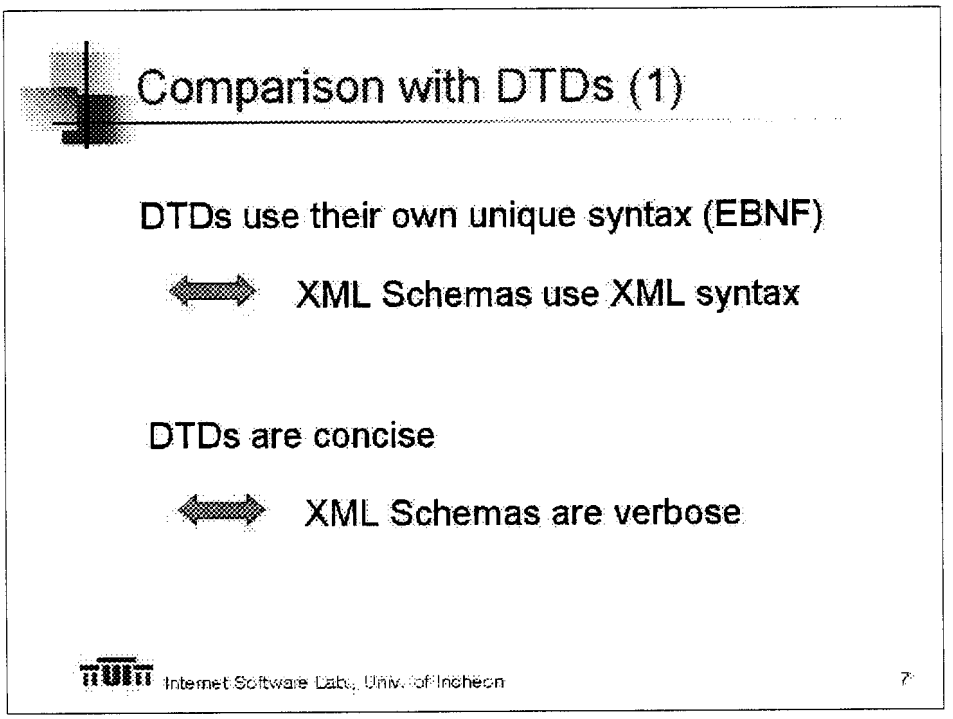


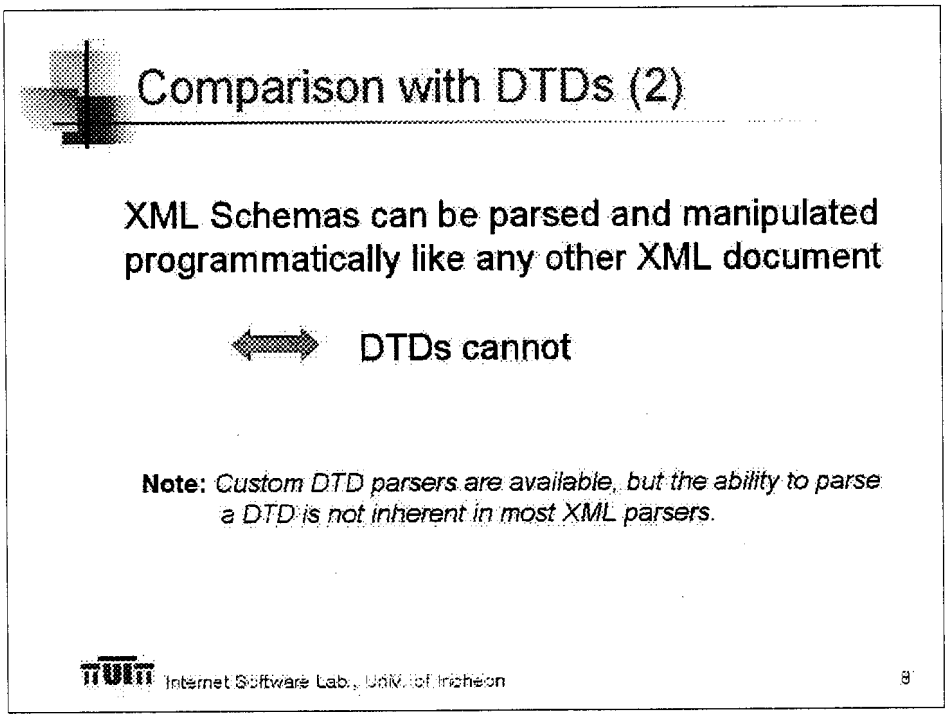
<element name="x" type="boolean" />
<element name="y" type="integer" />

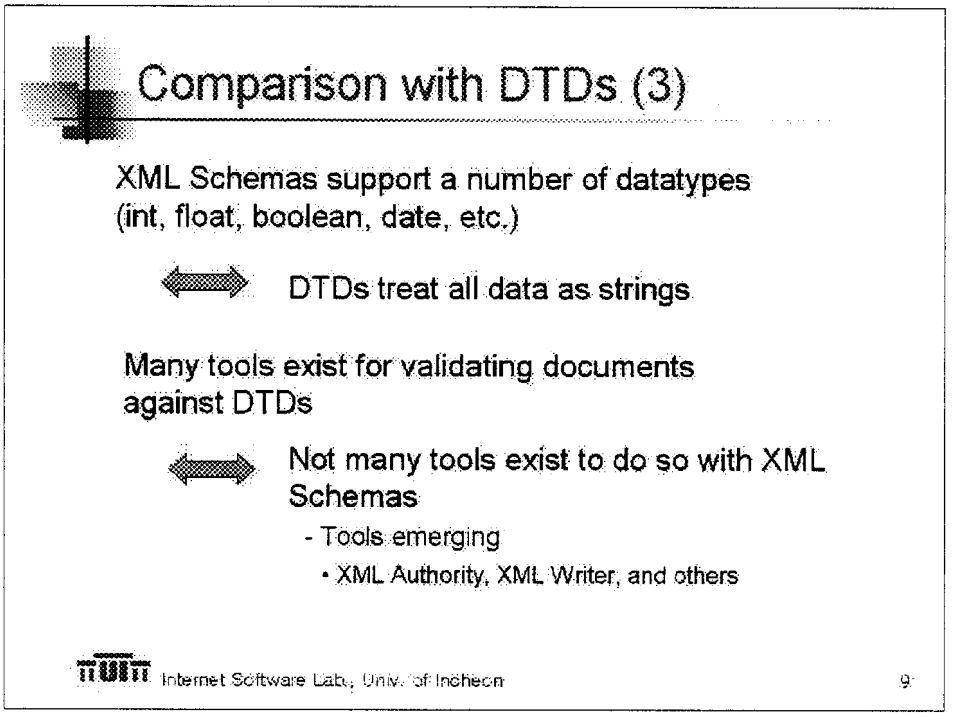
x and y now have enforceable data types

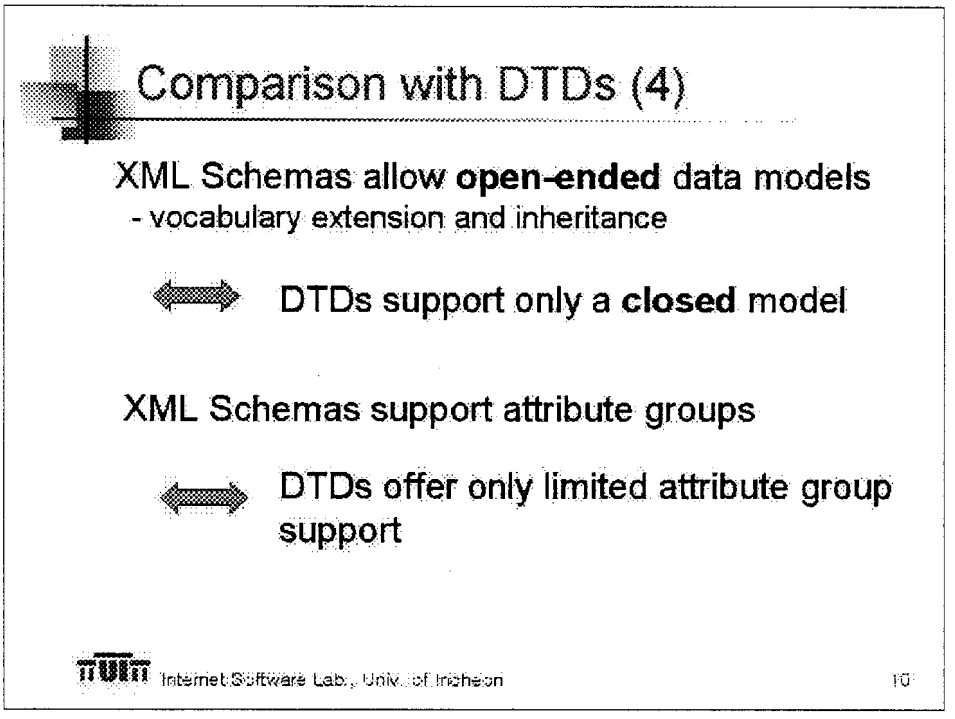
Schema Validation

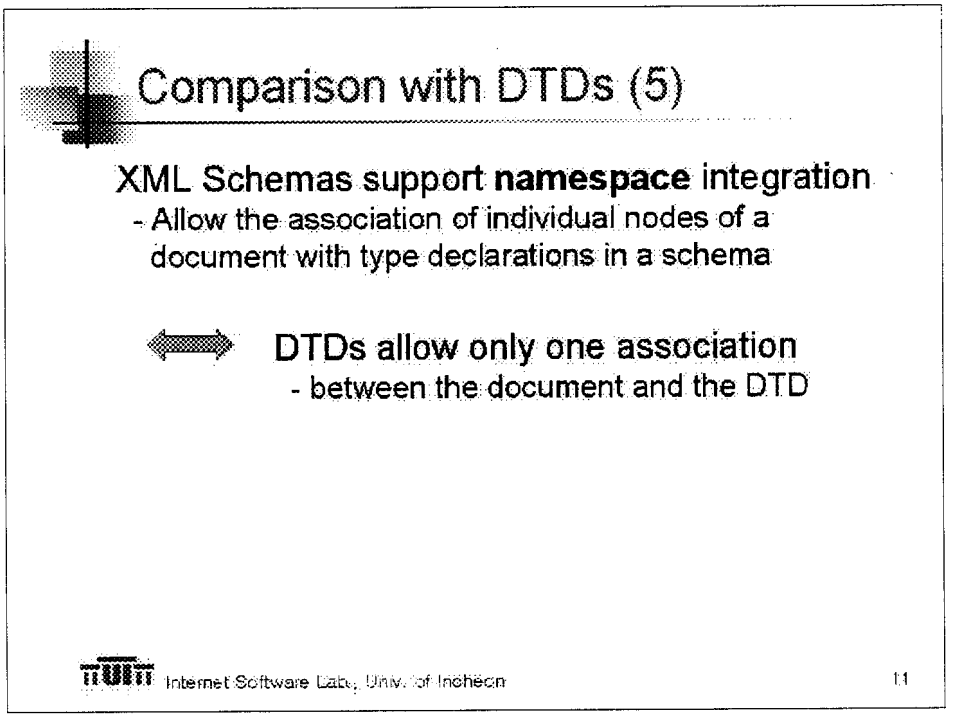
- Schemas can be used to validate a document in two ways
- **Content Model Validation**
  - Checks order and nesting of elements
  - Similar to DTD validation
- **Data Type Validation**
  - Checks the element content for valid type and range
  - Example: month element is an integer between 1 and 12
    - <month>15</month> INVALID!!
    - <month>5</month> VALID!!

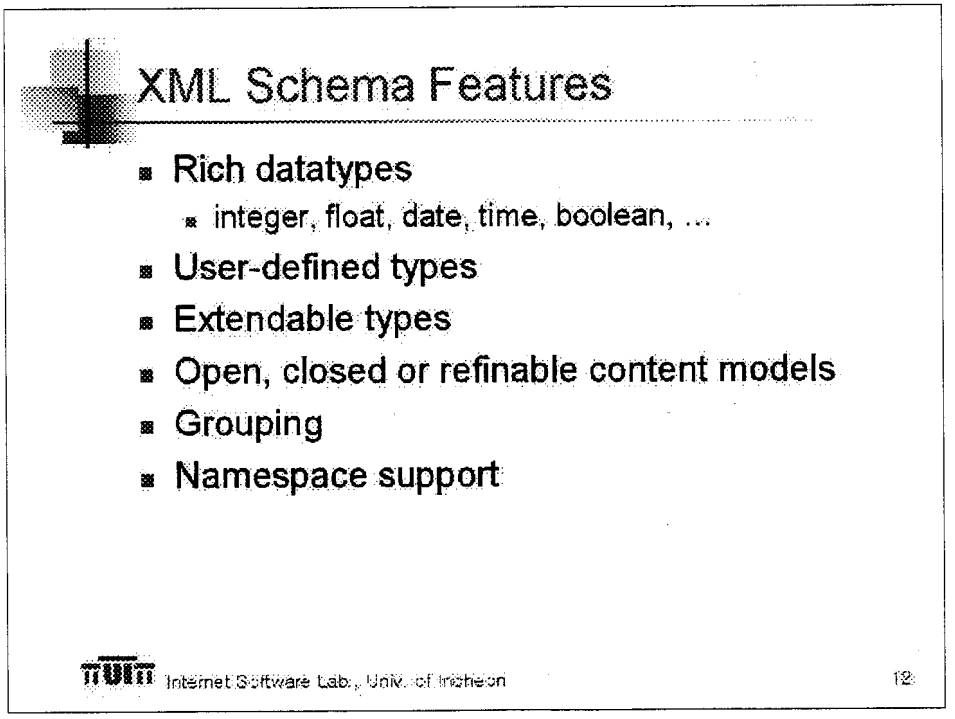

Comparison with DTDs (1)
DTDs use their own unique syntax (EBNF)
XML Schemas use XML syntax
DTDs are concise
XML Schemas are verbose
Internet Software Lab., Univ. of Incheon
7


Comparison with DTDs (2)
XML Schemas can be parsed and manipulated programmatically like any other XML document
DTDs cannot
Note: Custom DTD parsers are available, but the ability to parse a DTD is not inherent in most XML parsers.
Internet Software Lab., Univ. of Incheon
8


Comparison with DTDs (3)
XML Schemas support a number of datatypes (int, float, boolean, date, etc.)
DTDs treat all data as strings
Many tools exist for validating documents against DTDs
Not many tools exist to do so with XML Schemas
- Tools emerging
• XML Authority, XML Writer, and others
Internet Software Lab., Univ. of Incheon
9


Comparison with DTDs (4)
XML Schemas allow open-ended data models
- vocabulary extension and inheritance
DTDs support only a closed model
XML Schemas support attribute groups
DTDs offer only limited attribute group support
Internet Software Lab., Univ. of Incheon
10

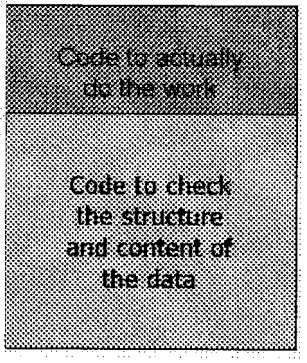

Comparison with DTDs (5)
XML Schemas support **namespace** integration
- Allow the association of individual nodes of a document with type declarations in a schema
⟺ DTDs allow only one association
- between the document and the DTD
Internet Software Lab., Univ. of Incheon
11


XML Schema Features
Rich datatypes
integer, float, date, time, boolean, ...
User-defined types
Extendable types
Open, closed or refinable content models
Grouping
Namespace support
Internet Software Lab., Univ. of Incheon
12

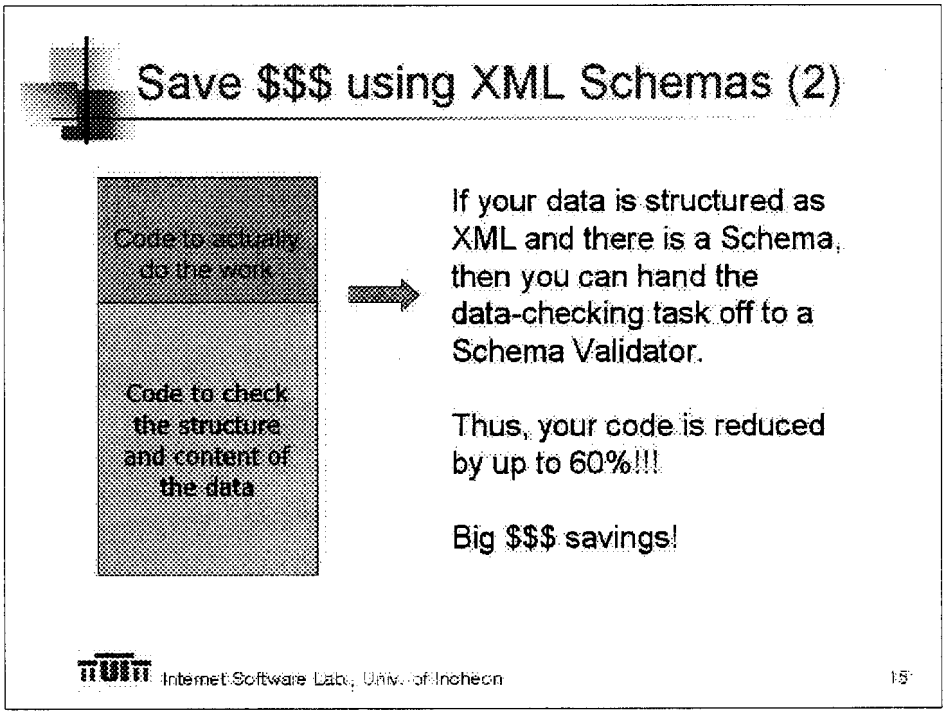
XML Schemas vs. DTDs

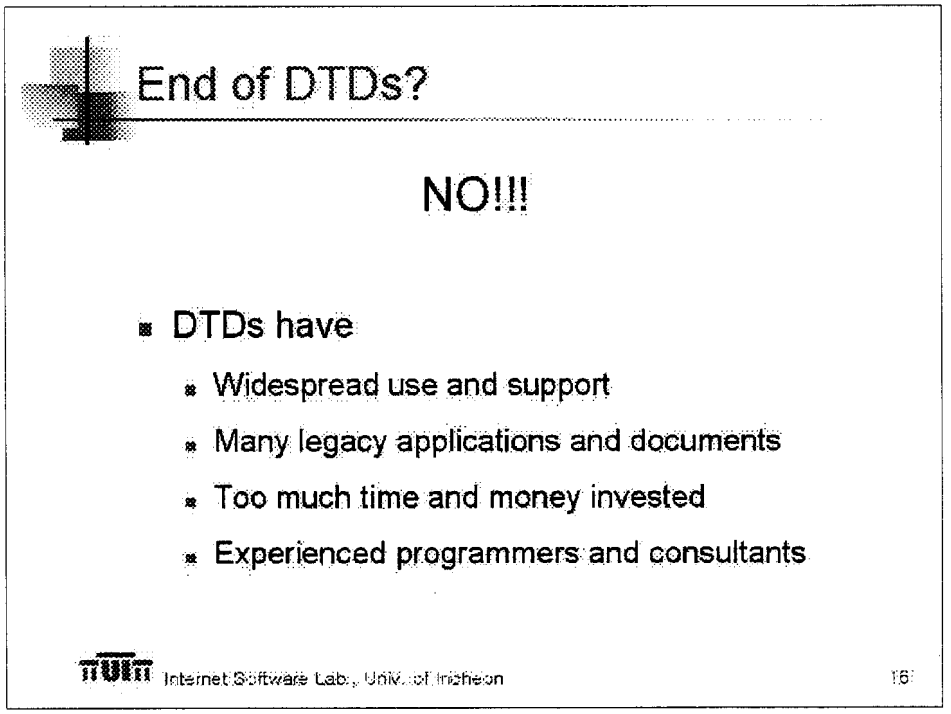
XML Schemas	DTDs
Support namespaces	n/a
Written in XML syntax	n/a
Extensive datatype support	Very limited
Full, object-oriented extensibility	Extended via string substitutions
Open, closed or refinable content models	Closed only

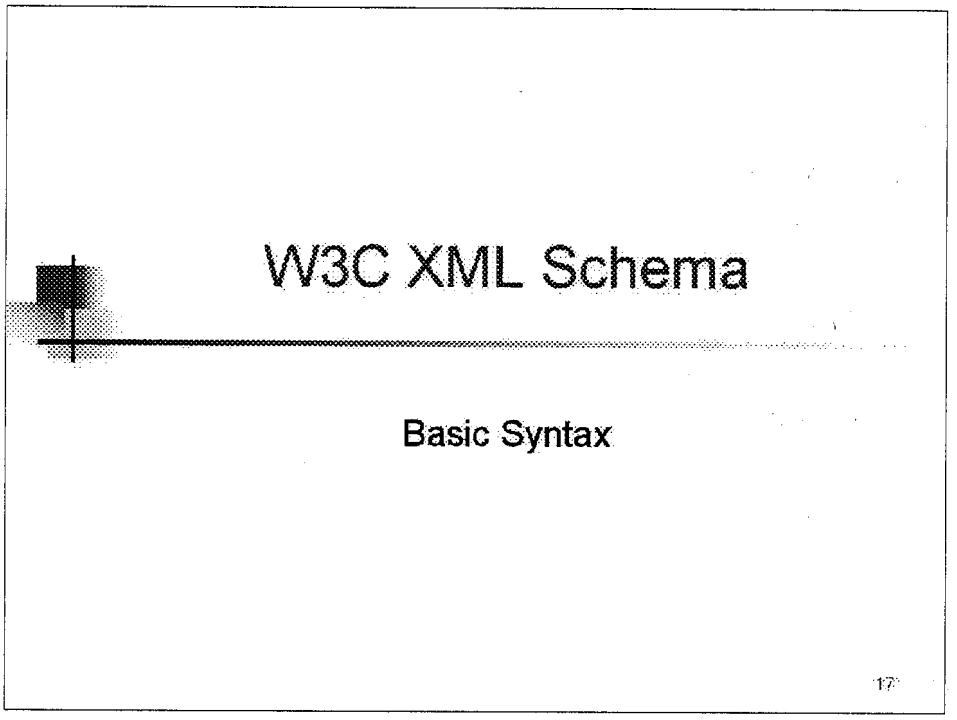
Save $$$ using XML Schemas (1)

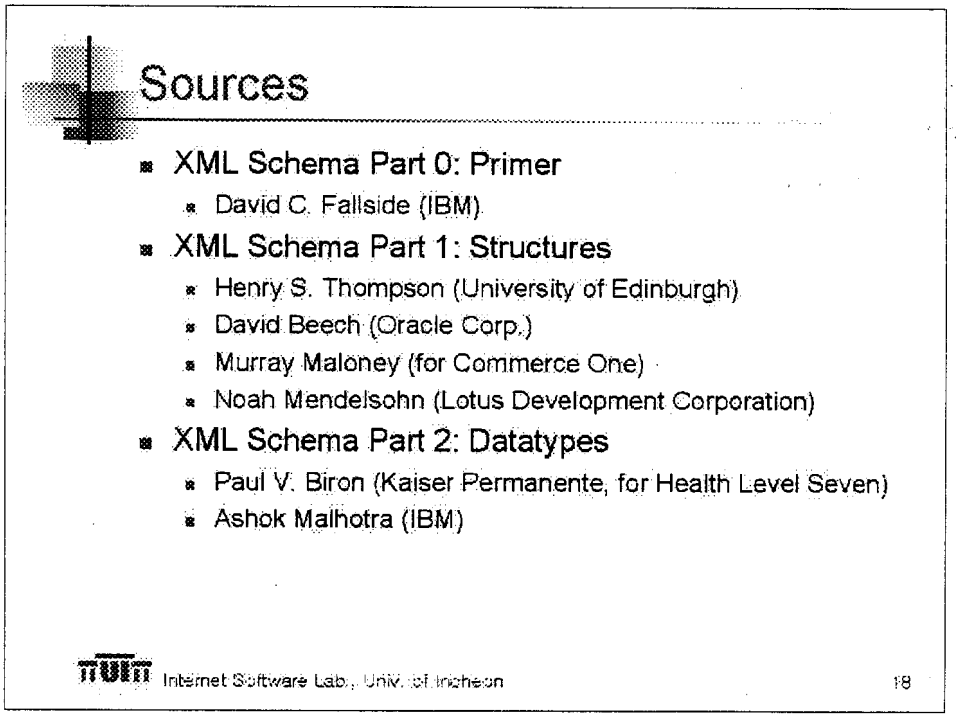


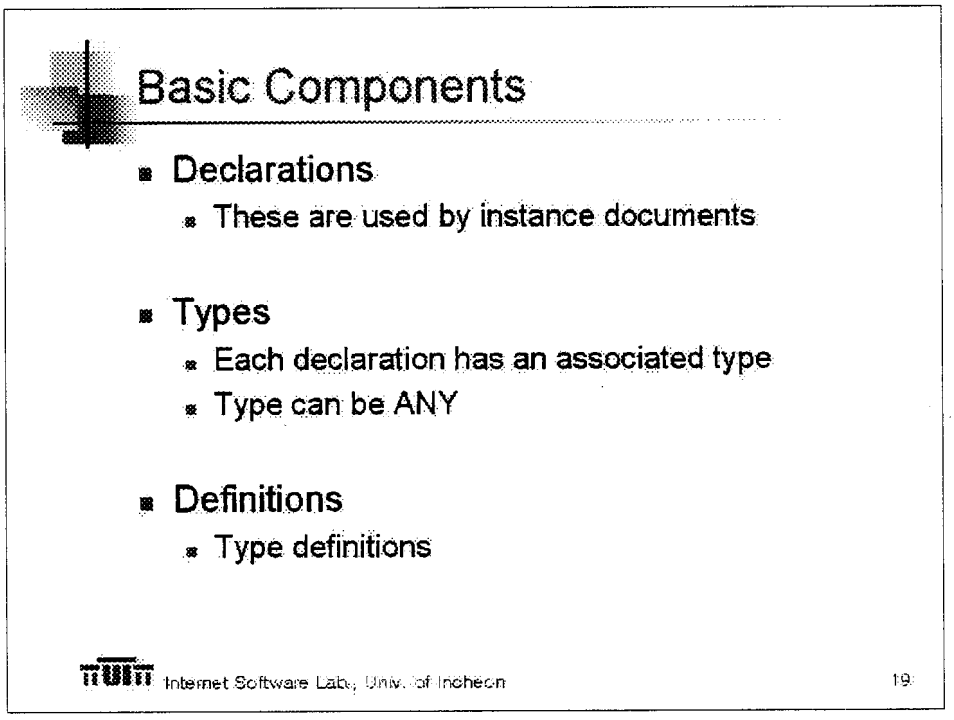
In a typical program, up to 60% of the code is spent checking the data!

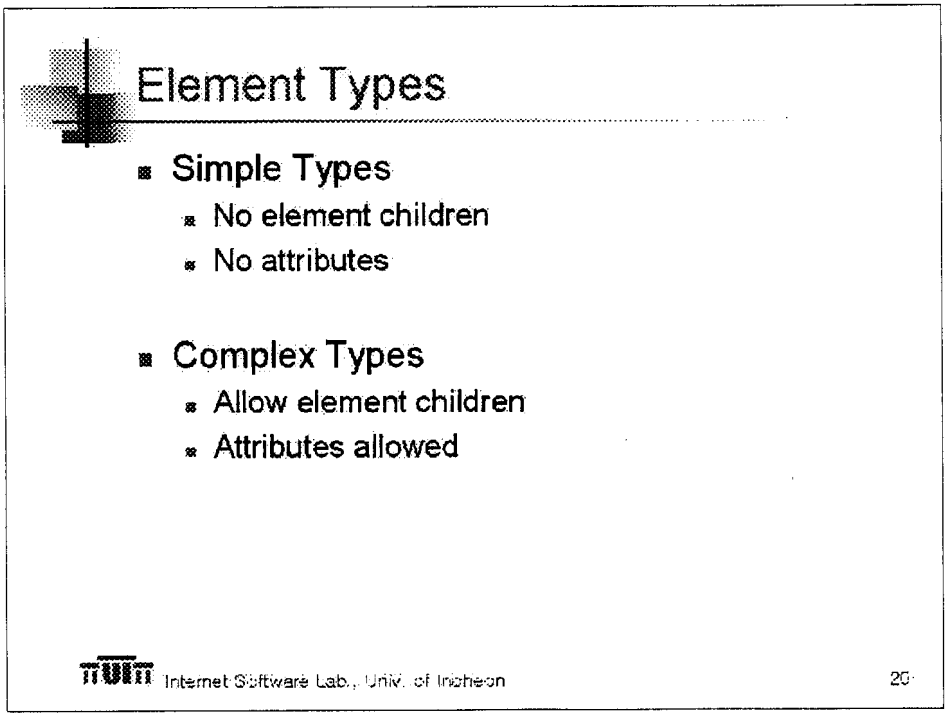

Save $$$ using XML Schemas (2)
Code to actually do the work
Code to check the structure and content of the data
If your data is structured as XML and there is a Schema, then you can hand the data-checking task off to a Schema Validator.
Thus, your code is reduced by up to 60%!!!
Big $$$ savings!
Internet Software Lab., Univ. of Incheon
15

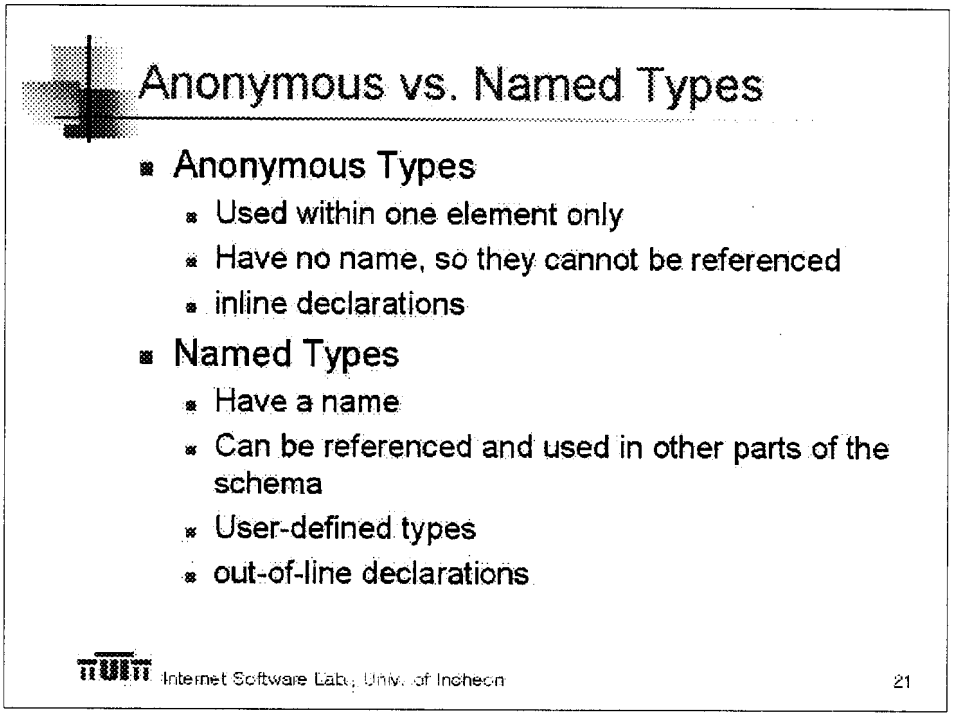

End of DTDs?
NO!!!
DTDs have
Widespread use and support
Many legacy applications and documents
Too much time and money invested
Experienced programmers and consultants
Internet Software Lab., Univ. of Incheon
16

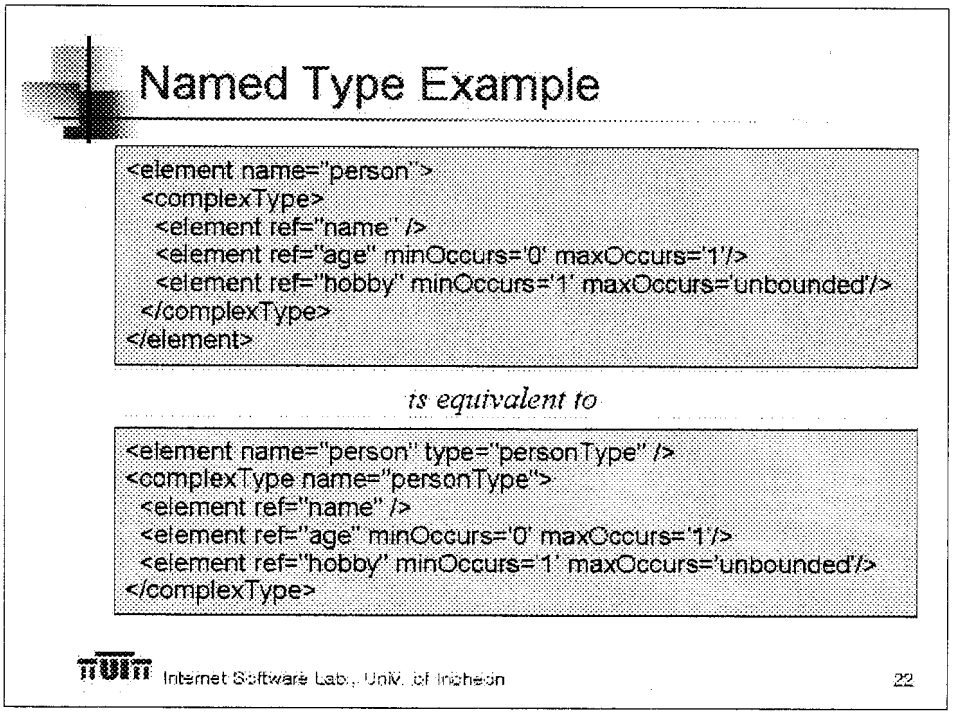

W3C XML Schema
Basic Syntax
17

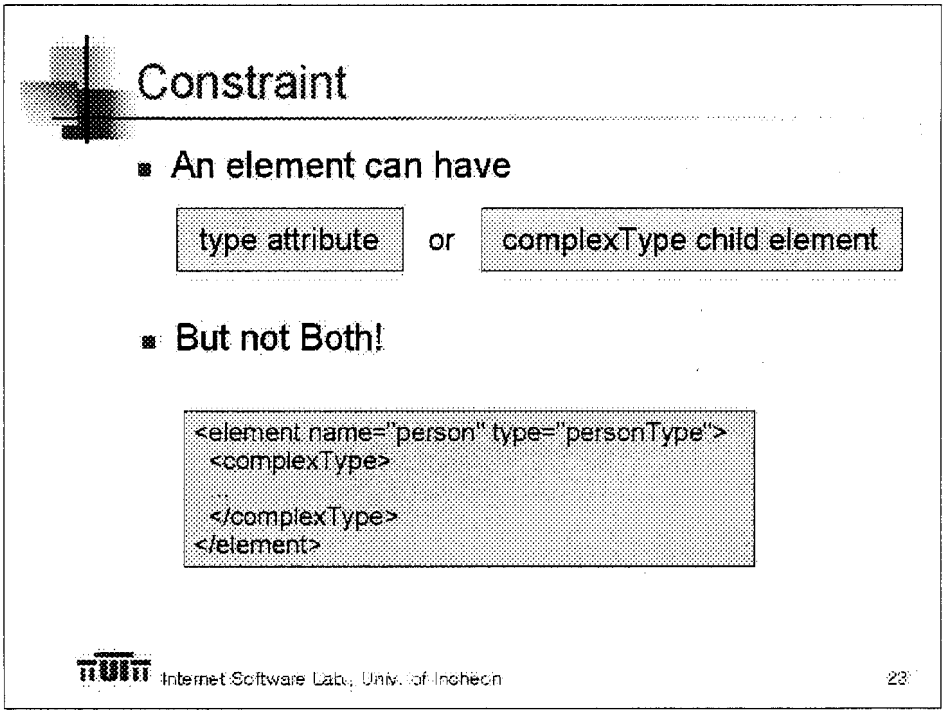

Sources
XML Schema Part 0: Primer
David C. Fallside (IBM)
XML Schema Part 1: Structures
Henry S. Thompson (University of Edinburgh)
David Beech (Oracle Corp.)
Murray Maloney (for Commerce One)
Noah Mendelsohn (Lotus Development Corporation)
XML Schema Part 2: Datatypes
Paul V. Biron (Kaiser Permanente, for Health Level Seven)
Ashok Malhotra (IBM)
Internet Software Lab., Univ. of Incheon
18

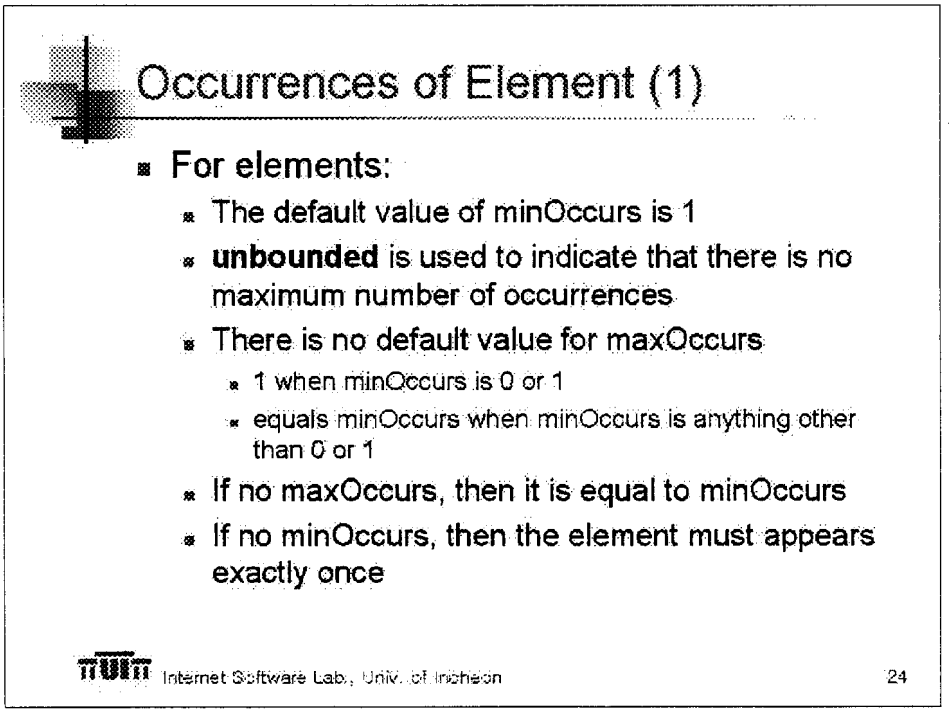

Basic Components
Declarations
These are used by instance documents
Types
Each declaration has an associated type
Type can be ANY
Definitions
Type definitions
Internet Software Lab., Univ. of Incheon
19


Element Types
Simple Types
No element children
No attributes
Complex Types
Allow element children
Attributes allowed
Internet Software Lab., Univ. of Incheon
20


Anonymous vs. Named Types
Anonymous Types
Used within one element only
Have no name, so they cannot be referenced
inline declarations
Named Types
Have a name
Can be referenced and used in other parts of the schema
User-defined types
out-of-line declarations
Internet Software Lab., Univ. of Incheon
21


Named Type Example
<element name="person">
<complexType>
<element ref="name" />
<element ref="age" minOccurs='0' maxOccurs='1'/>
<element ref="hobby" minOccurs='1' maxOccurs='unbounded'/>
</complexType>
</element>
ts equivalent to
<element name="person" type="personType" />
<complexType name="personType">
<element ref="name" />
<element ref="age" minOccurs='0' maxOccurs='1'/>
<element ref="hobby" minOccurs='1' maxOccurs='unbounded'/>
</complexType>
Internet Software Lab., Univ. of Incheon
22


Constraint
An element can have
type attribute
or
complexType child element
But not Both!
<element name="person" type="personType">
<complexType>
...
</complexType>
</element>
Internet Software Lab., Univ. of Incheon
23

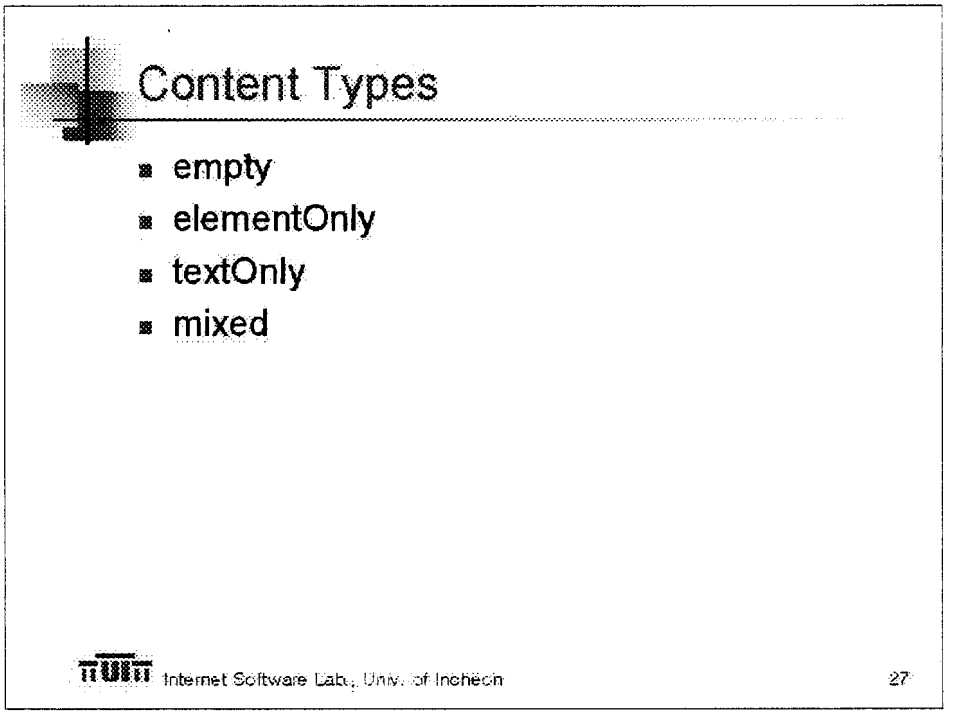

Occurrences of Element (1)
For elements:
The default value of minOccurs is 1
unbounded is used to indicate that there is no maximum number of occurrences
There is no default value for maxOccurs
1 when minOccurs is 0 or 1
equals minOccurs when minOccurs is anything other than 0 or 1
If no maxOccurs, then it is equal to minOccurs
If no minOccurs, then the element must appears exactly once
Internet Software Lab., Univ. of Incheon
24

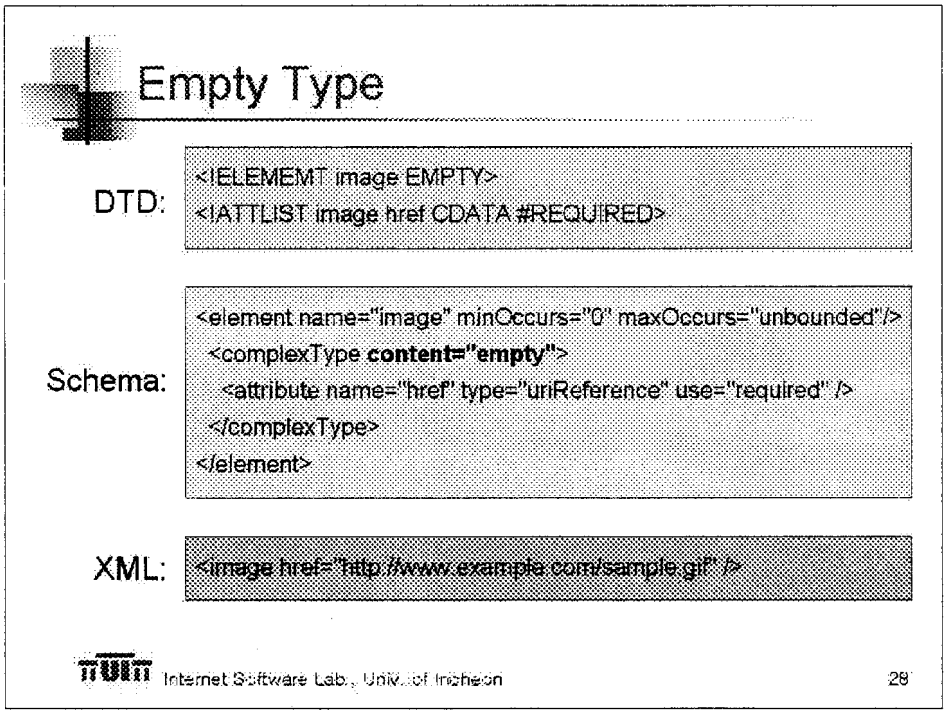
Occurrences of Element (2)

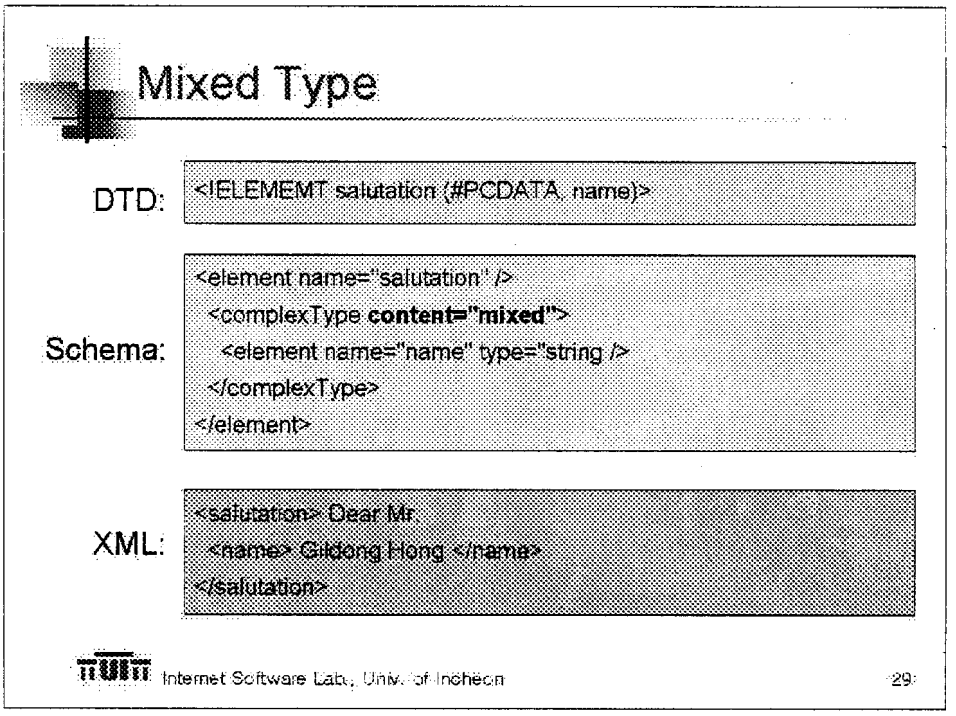
Element	minOccurs	maxOccurs	DTD
A	1	1	A
A	0	1	A?
A	1	unbounded	A+
A	0	unbounded	A*

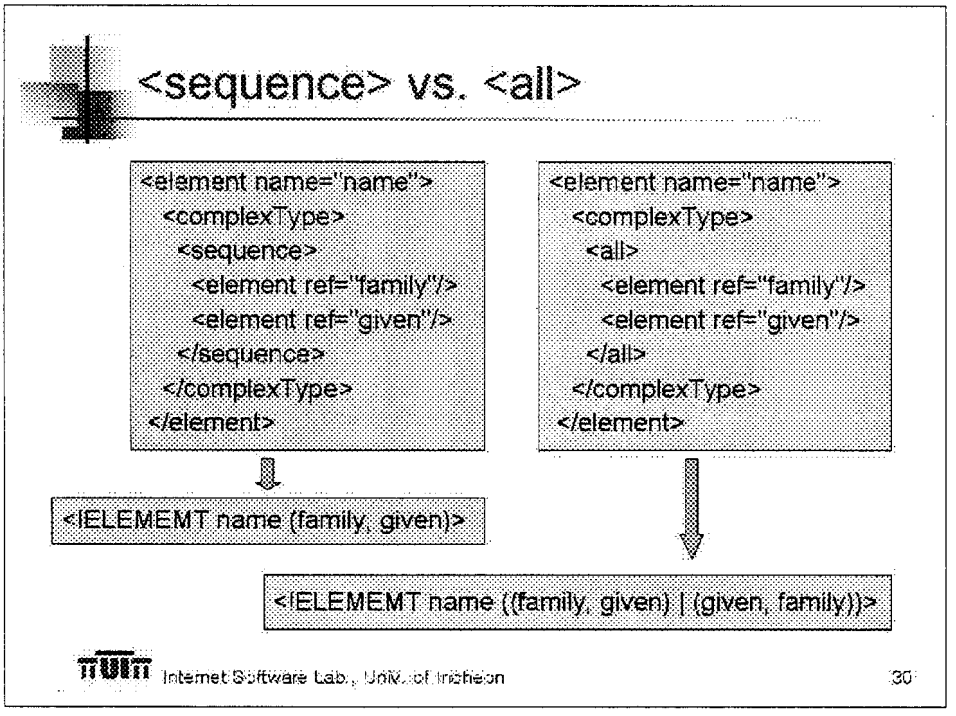
Occurrences of Attribute

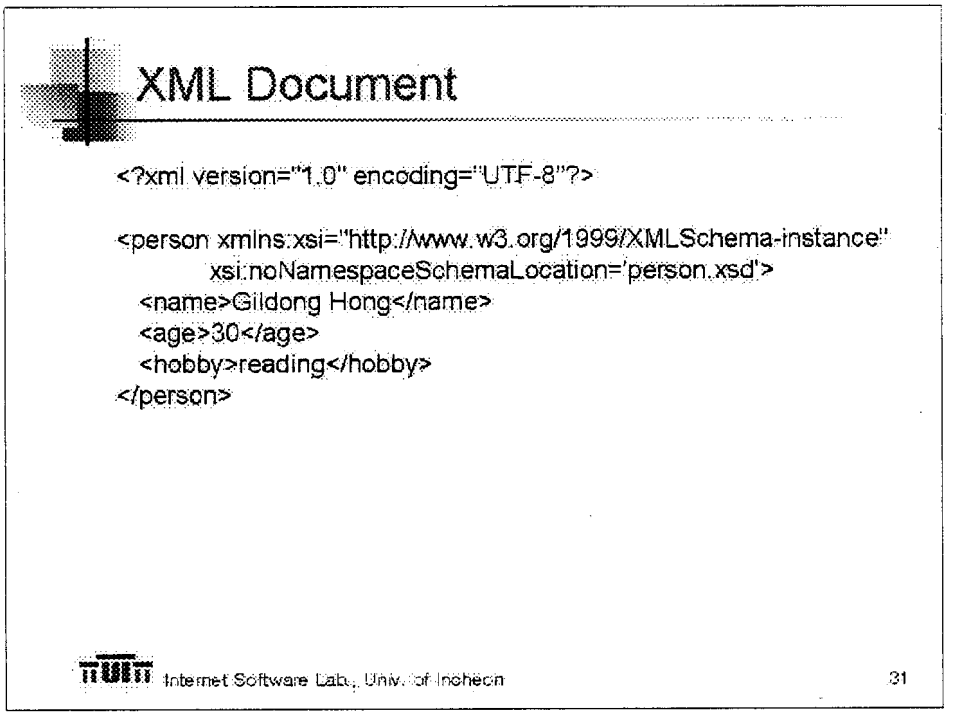
- For attributes:
  - Attributes may appear once or not at all
  - A **use** attribute is used in an attribute declaration to indicate whether the attribute's value is **required** or **optional**
  - If **optional**, whether the attribute's value is **fixed** or whether there is a **default** value can also be specified
  - An attribute, **value**, provides any value that is called for

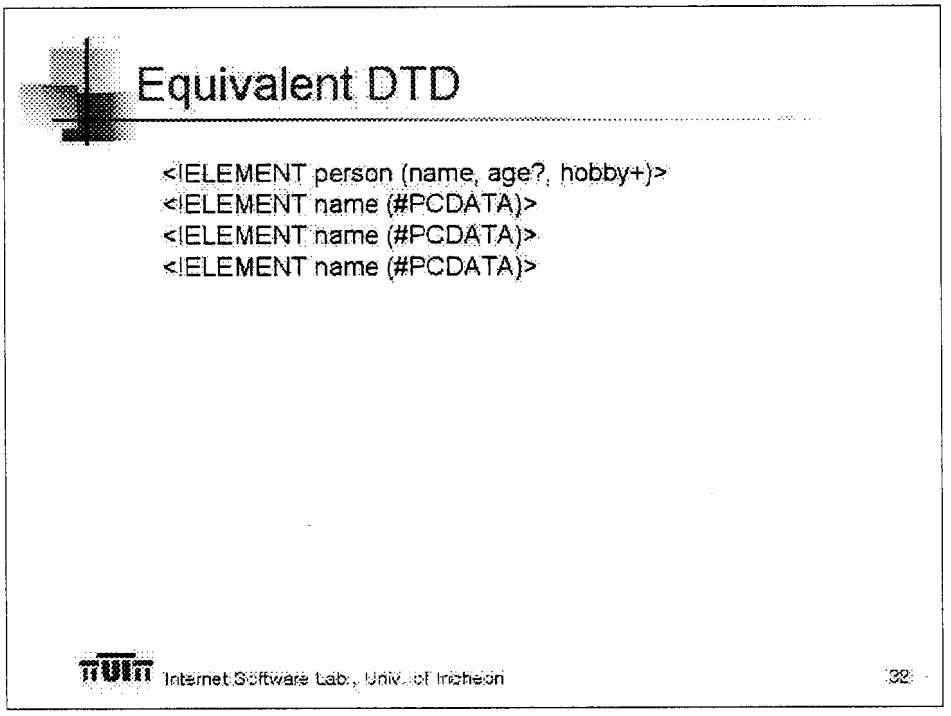

Content Types
empty
elementOnly
textOnly
mixed
Internet Software Lab., Univ. of Incheon
27

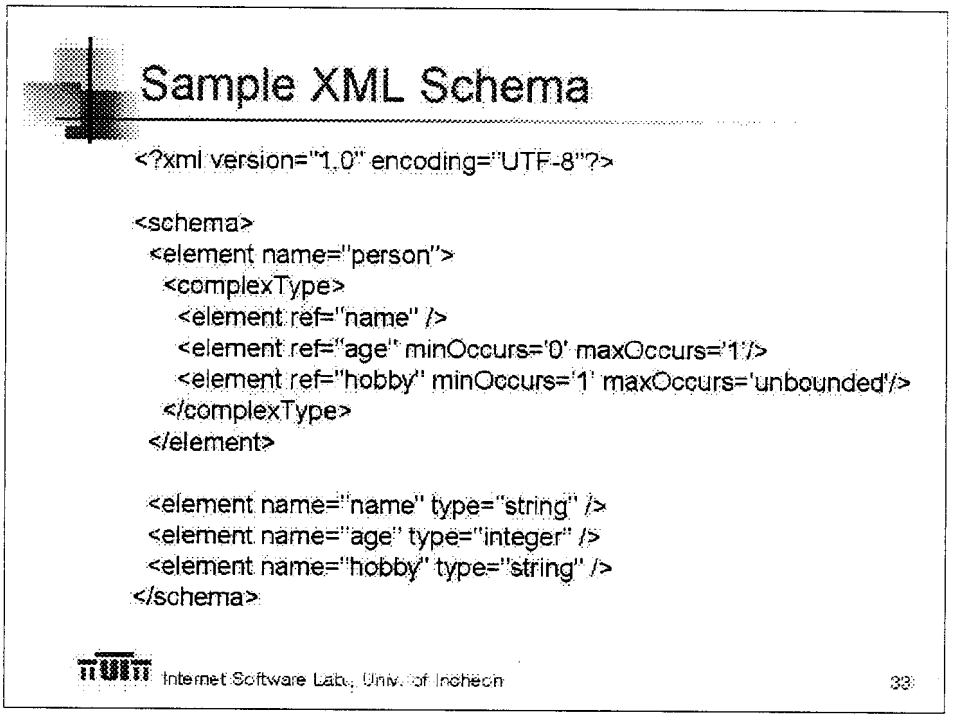

Empty Type
DTD:
<!ELEMEMT image EMPTY>
<!ATTLIST image href CDATA #REQUIRED>
Schema:
<element name="image" minOccurs="0" maxOccurs="unbounded"/>
<complexType content="empty">
<attribute name="href" type="uriReference" use="required" />
</complexType>
</element>
XML:
<image href="http://www.example.com/sample.gif" />
Internet Software Lab., Univ. of Incheon
28

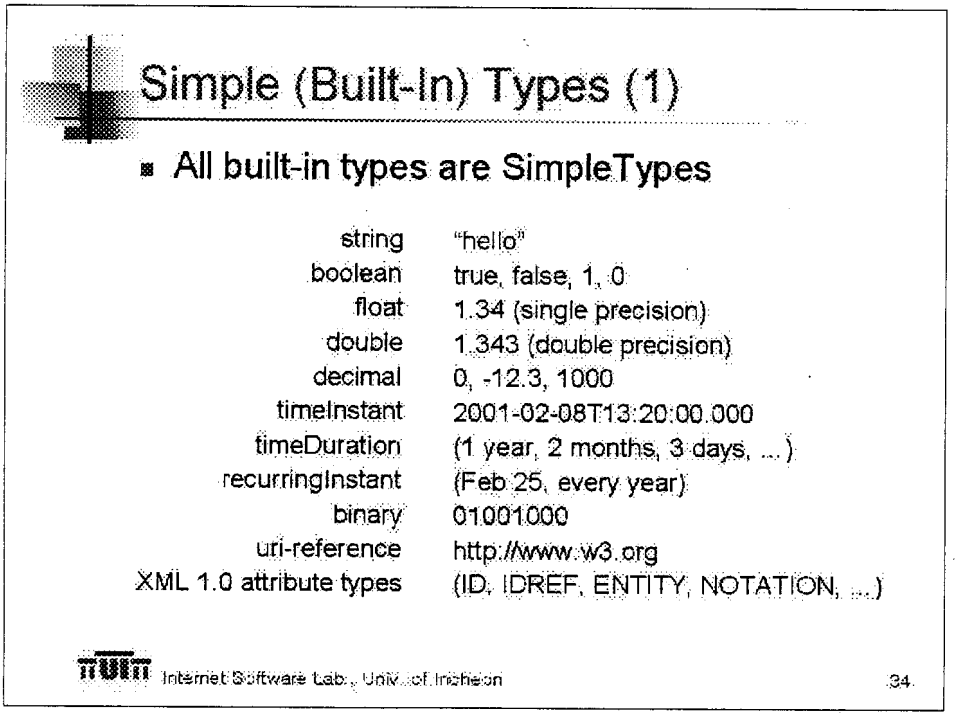

Mixed Type
DTD:
<!ELEMEMT salutation (#PCDATA, name)>
Schema:
<element name="salutation" />
<complexType content="mixed">
<element name="name" type="string />
</complexType>
</element>
XML:
<salutation> Dear Mr.
<name> Gildong Hong </name>
</salutation>
Internet Software Lab., Univ. of Incheon
29

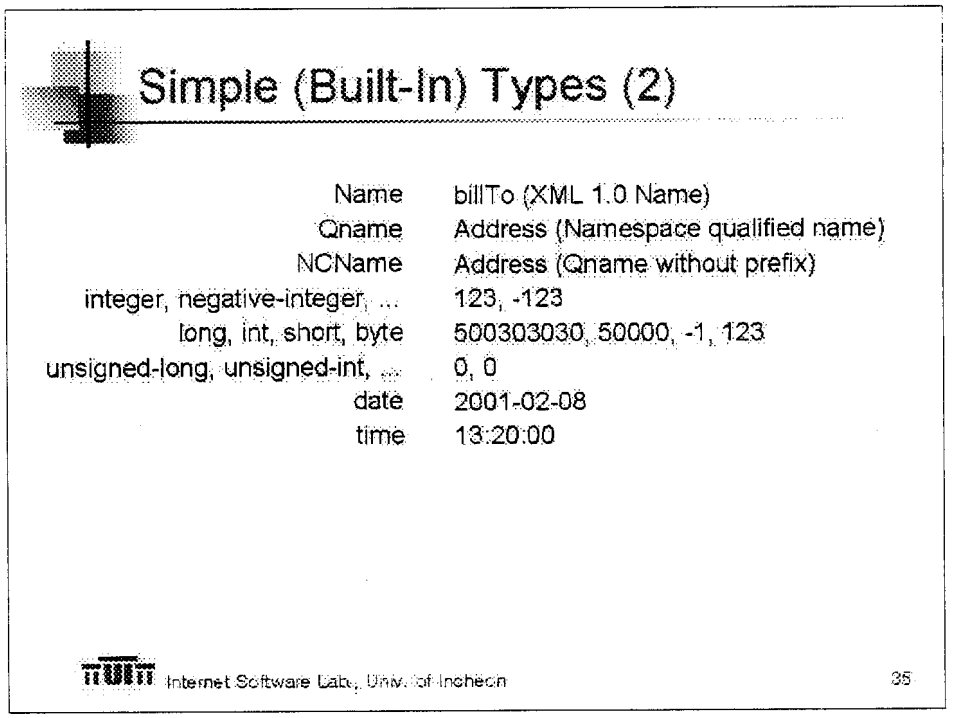

<sequence> vs. <all>
<element name="name">
<complexType>
<sequence>
<element ref="family"/>
<element ref="given"/>
</sequence>
</complexType>
</element>
<element name="name">
<complexType>
<all>
<element ref="family"/>
<element ref="given"/>
</all>
</complexType>
</element>
<!ELEMEMT name (family, given)>
<!ELEMEMT name ((family, given) | (given, family))>
Internet Software Lab., Univ. of Incheon
30

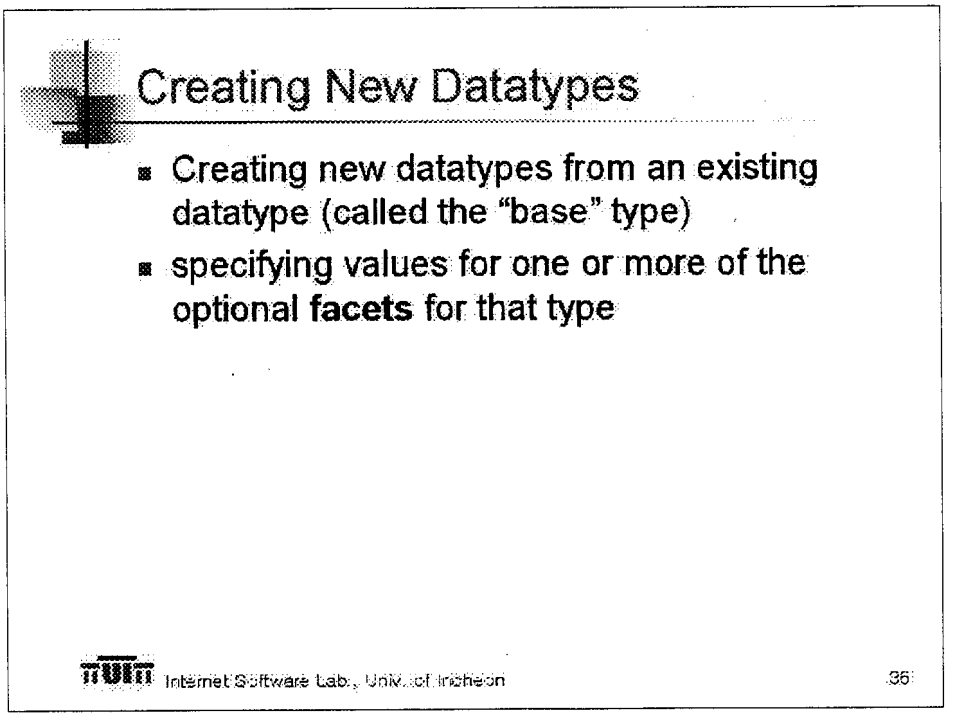

XML Document
<?xml version="1.0" encoding="UTF-8"?>
<person xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xsi:noNamespaceSchemaLocation='person.xsd'>
<name>Gildong Hong</name>
<age>30</age>
<hobby>reading</hobby>
</person>
Internet Software Lab., Univ. of Incheon
31

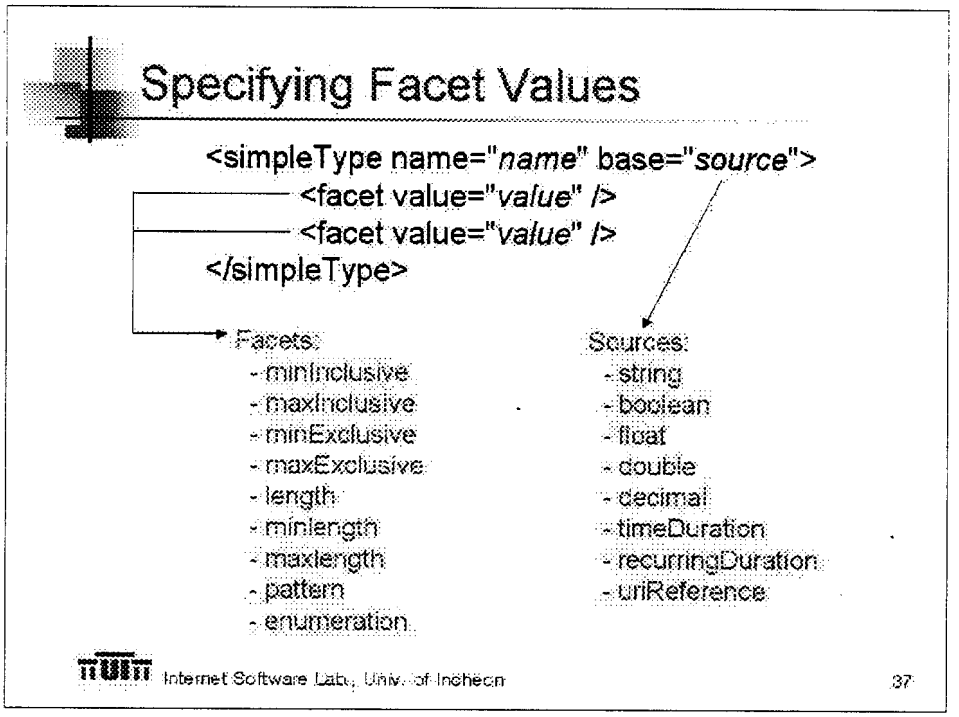

Equivalent DTD
<!ELEMENT person (name, age?, hobby+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT name (#PCDATA)>
Internet Software Lab., Univ. of Incheon
32

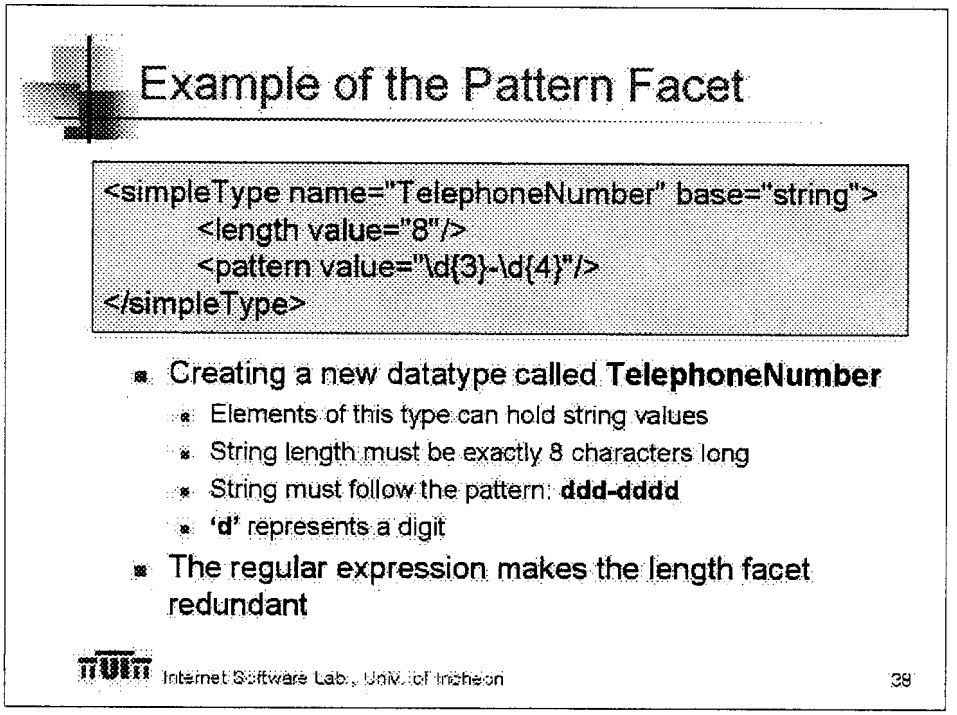

Sample XML Schema
<?xml version="1.0" encoding="UTF-8"?>
<schema>
<element name="person">
<complexType>
<element ref="name" />
<element ref="age" minOccurs='0' maxOccurs='1'/>
<element ref="hobby" minOccurs='1' maxOccurs='unbounded'/>
</complexType>
</element>
<element name="name" type="string" />
<element name="age" type="integer" />
<element name="hobby" type="string" />
</schema>
Internet Software Lab., Univ. of Incheon
33

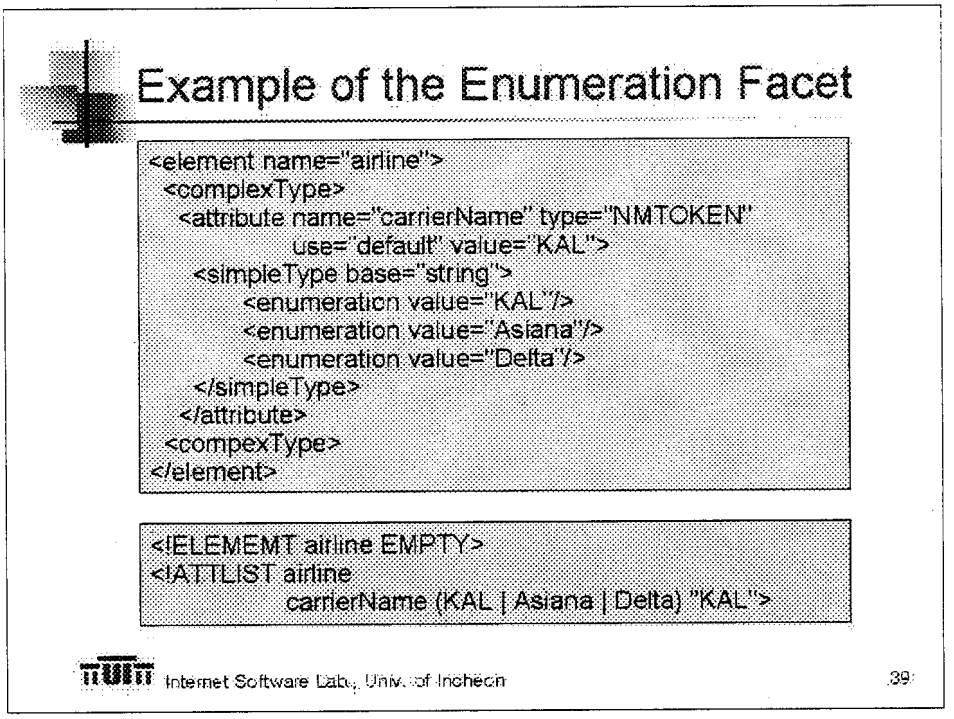

Simple (Built-In) Types (1)
All built-in types are SimpleTypes
string "hello"
boolean true, false, 1, 0
float 1.34 (single precision)
double 1.343 (double precision)
decimal 0, -12.3, 1000
timeInstant 2001-02-08T13:20:00.000
timeDuration (1 year, 2 months, 3 days, ...)
recurringInstant (Feb 25, every year)
binary 01001000
uri-reference http://www.w3.org
XML 1.0 attribute types (ID, IDREF, ENTITY, NOTATION, ...)
Internet Software Lab., Univ. of Incheon
34

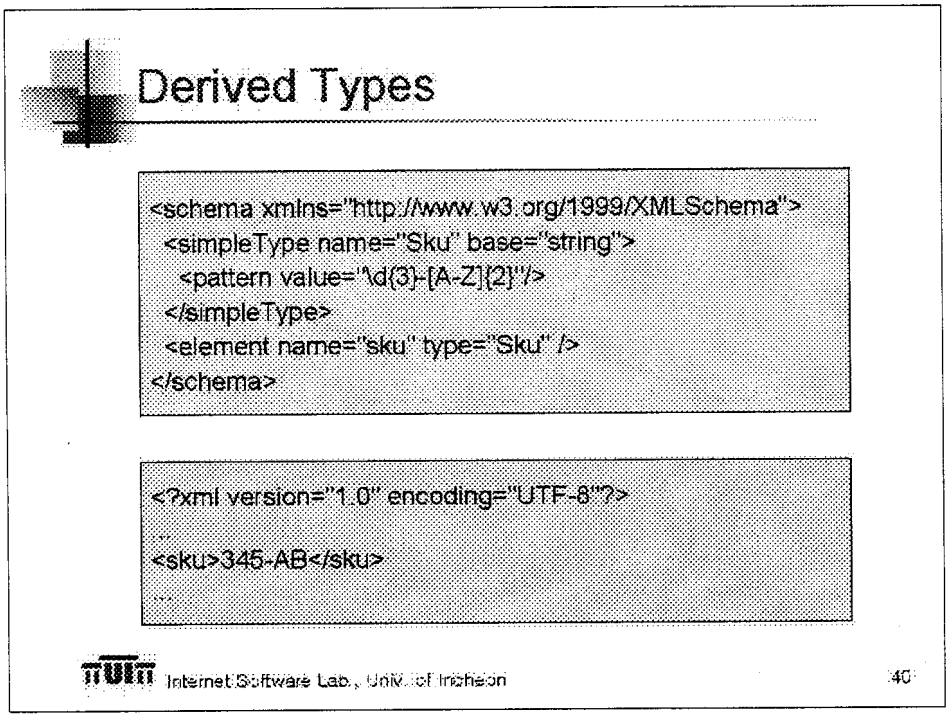

Simple (Built-In) Types (2)
Name billTo (XML 1.0 Name)
Qname Address (Namespace qualified name)
NCName Address (Qname without prefix)
integer, negative-integer, ... 123, -123
long, int, short, byte 500303030, 50000, -1, 123
unsigned-long, unsigned-int, ... 0, 0
date 2001-02-08
time 13:20:00
Internet Software Lab., Univ. of Incheon
35


Creating New Datatypes
Creating new datatypes from an existing datatype (called the "base" type)
specifying values for one or more of the optional facets for that type
Internet Software Lab., Univ. of Incheon
36


Specifying Facet Values
<simpleType name="name" base="source">
<facet value="value" />
<facet value="value" />
</simpleType>
Facets:
- minInclusive
- maxInclusive
- minExclusive
- maxExclusive
- length
- minlength
- maxlength
- pattern
- enumeration
Sources:
- string
- boolean
- float
- double
- decimal
- timeDuration
- recurringDuration
- uriReference
Internet Software Lab., Univ. of Incheon
37


Example of the Pattern Facet
<simpleType name="TelephoneNumber" base="string">
<length value="8"/>
<pattern value="\d{3}-\d{4}"/>
</simpleType>
Creating a new datatype called TelephoneNumber
Elements of this type can hold string values
String length must be exactly 8 characters long
String must follow the pattern: ddd-dddd
'd' represents a digit
The regular expression makes the length facet redundant
Internet Software Lab., Univ. of Incheon
38


Example of the Enumeration Facet
<element name="airline">
<complexType>
<attribute name="carrierName" type="NMTOKEN"
use="default" value="KAL">
<simpleType base="string">
<enumeration value="KAL"/>
<enumeration value="Asiana"/>
<enumeration value="Delta"/>
</simpleType>
</attribute>
<compexType>
</element>
<!ELEMEMT airline EMPTY>
<!ATTLIST airline
carrierName (KAL | Asiana | Delta) "KAL">
Internet Software Lab., Univ. of Incheon
39


Derived Types
<schema xmlns="http://www.w3.org/1999/XMLSchema">
<simpleType name="Sku" base="string">
<pattern value="\d{3}-[A-Z]{2}"/>
</simpleType>
<element name="sku" type="Sku" />
</schema>
<?xml version="1.0" encoding="UTF-8"?>
...
<sku>345-AB</sku>
...
Internet Software Lab., Univ. of Incheon
40

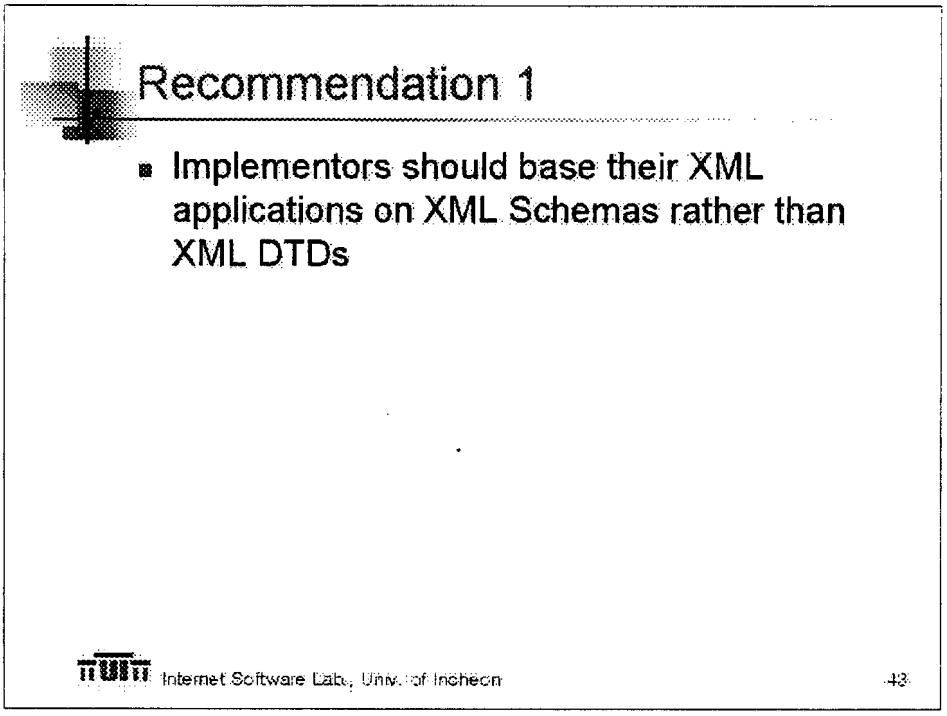
Guidelines for implementing Dublin Core in XML

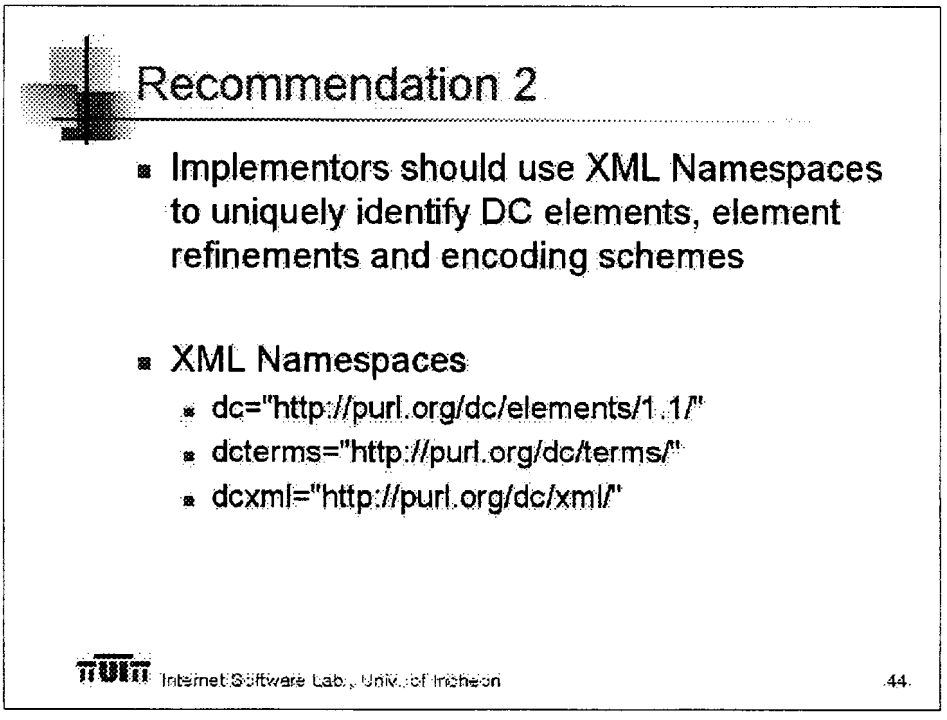
UKOLN
University of Bath

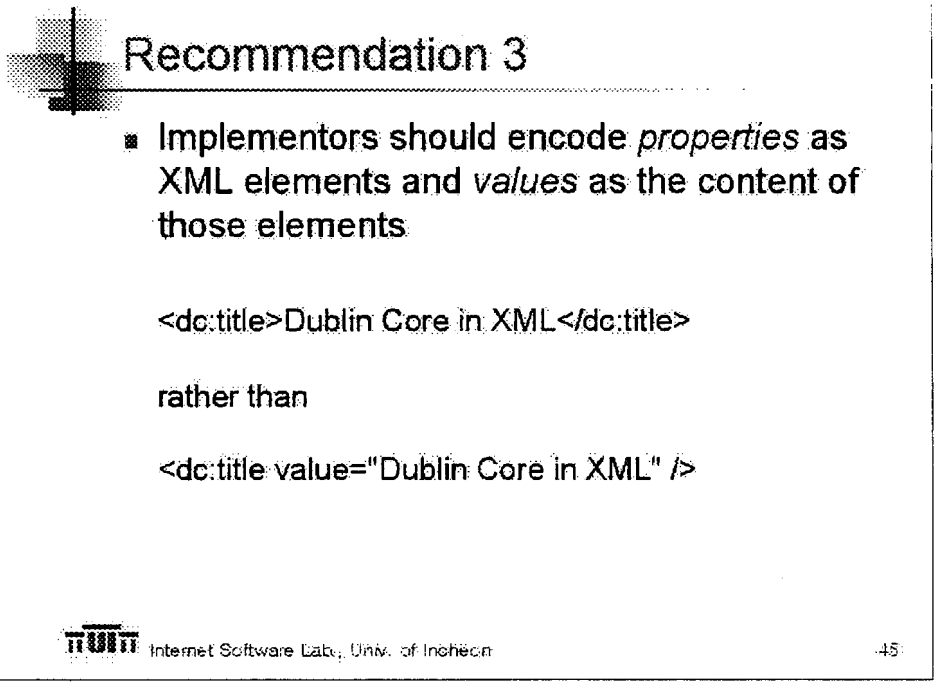
41

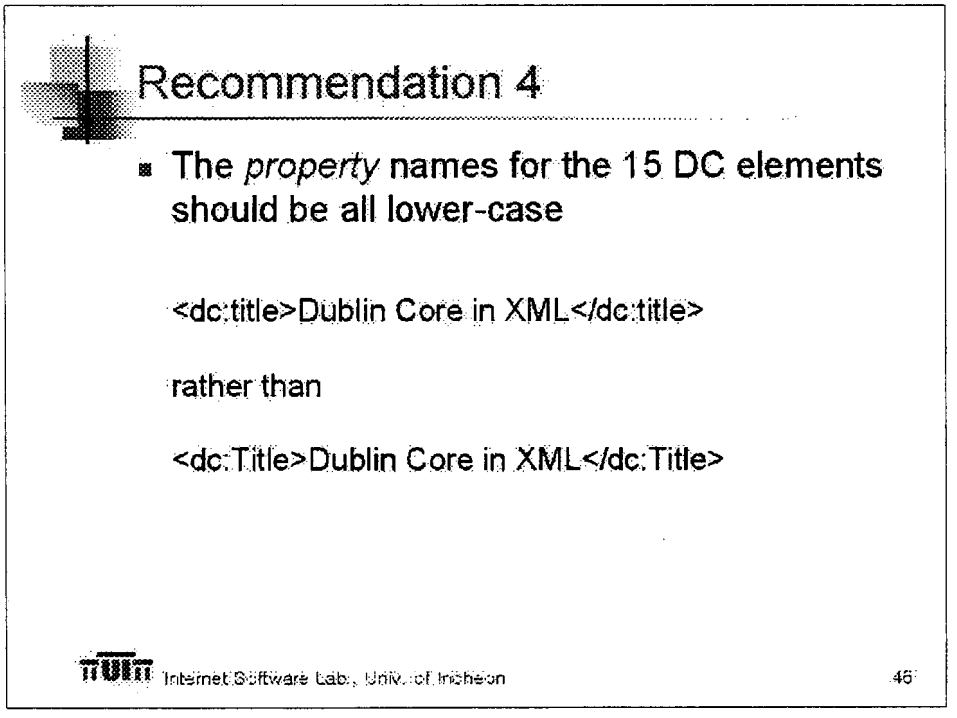
URLs

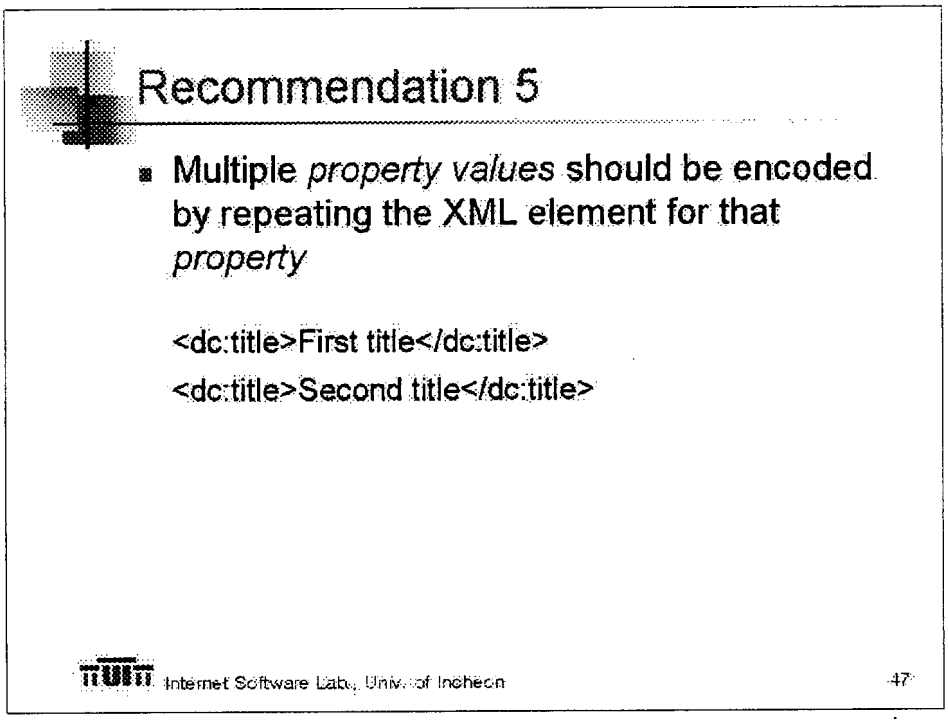
- Guidelines for implementing Dublin Core in XML
  - http://www.ukoln.ac.uk/metadata/dcmi/dc-xml-guidelines/
- Example Dublin Core XML Schemas
  - http://www.ukoln.ac.uk/metadata/dcmi/dcxml/examples.html

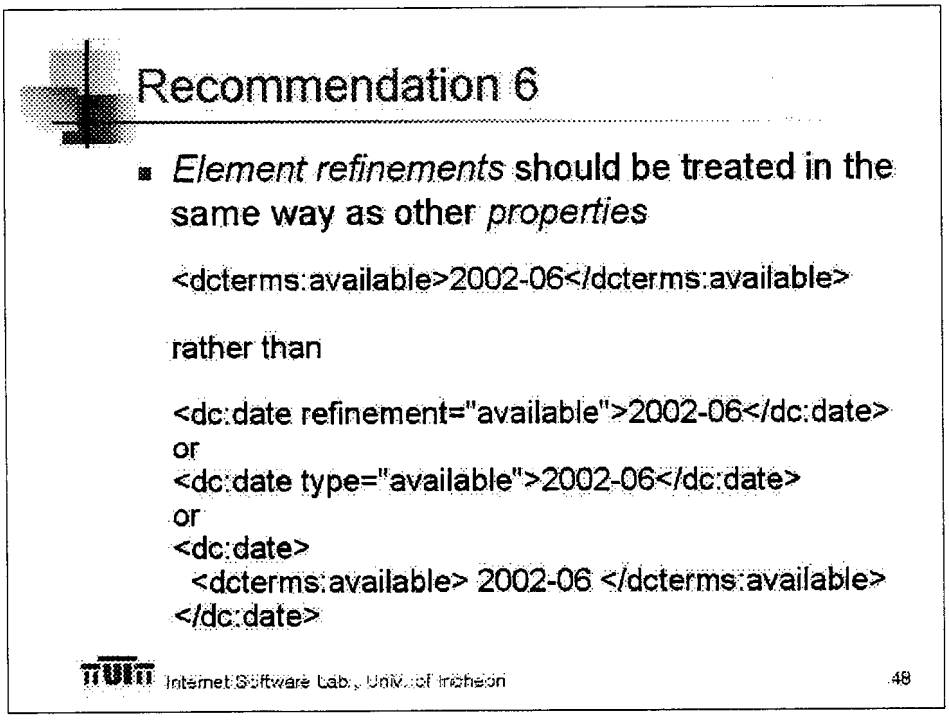

Recommendation 1
Implementors should base their XML applications on XML Schemas rather than XML DTDs
Internet Software Lab., Univ. of Incheon
43

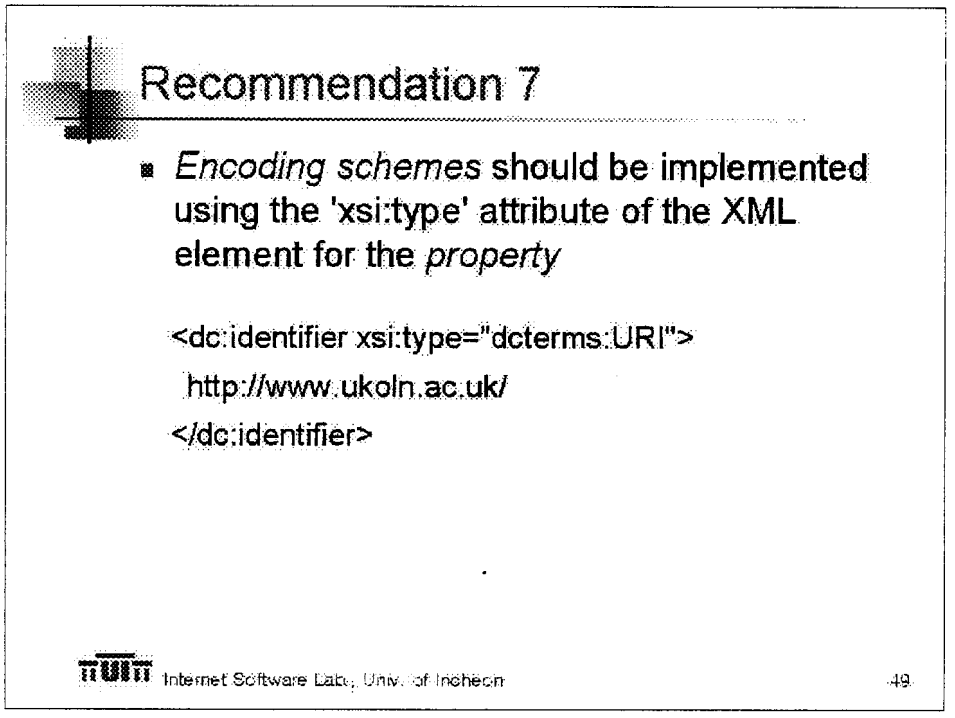

Recommendation 2
Implementors should use XML Namespaces to uniquely identify DC elements, element refinements and encoding schemes
XML Namespaces
dc="http://purl.org/dc/elements/1.1/"
dcterms="http://purl.org/dc/terms/"
dcxml="http://purl.org/dc/xml/"
Internet Software Lab., Univ. of Incheon
44

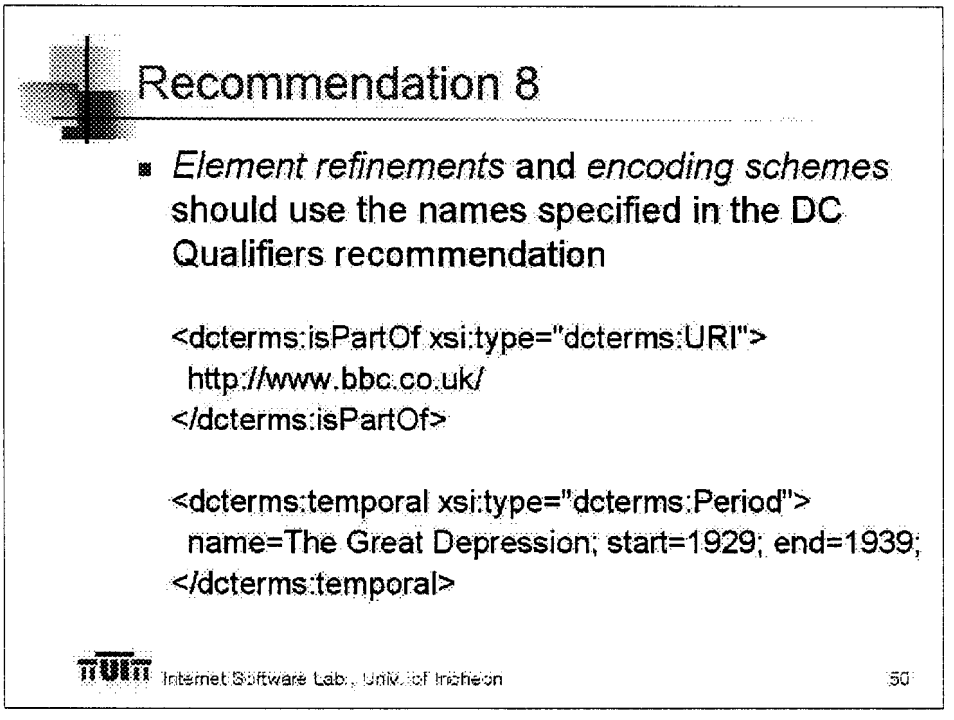

Recommendation 3
Implementors should encode properties as XML elements and values as the content of those elements
<dc:title>Dublin Core in XML</dc:title>
rather than
<dc:title value="Dublin Core in XML" />
Internet Software Lab., Univ. of Incheon
45

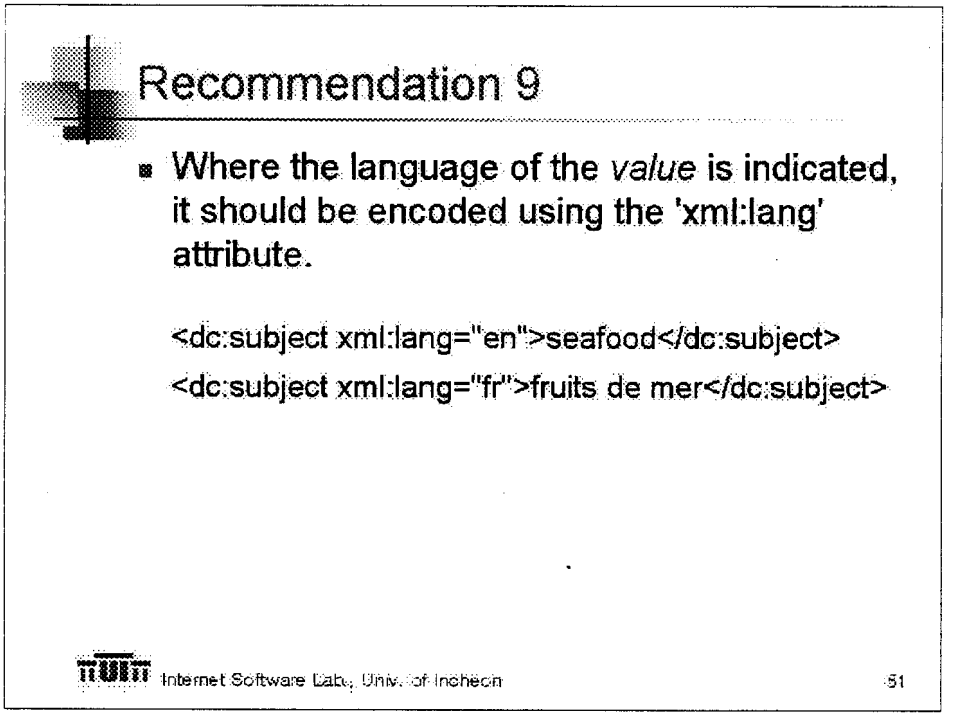

Recommendation 4
The property names for the 15 DC elements should be all lower-case
<dc:title>Dublin Core in XML</dc:title>
rather than
<dc:Title>Dublin Core in XML</dc:Title>
Internet Software Lab., Univ. of Incheon
46

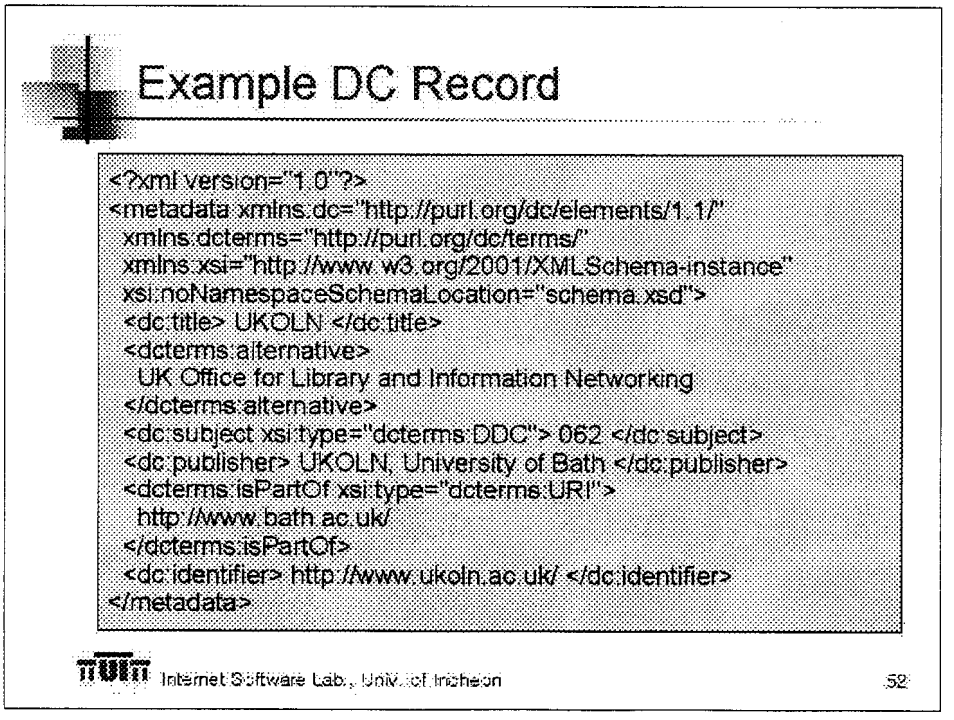

Recommendation 5
Multiple property values should be encoded by repeating the XML element for that property
<dc:title>First title</dc:title>
<dc:title>Second title</dc:title>
Internet Software Lab., Univ. of Incheon
47

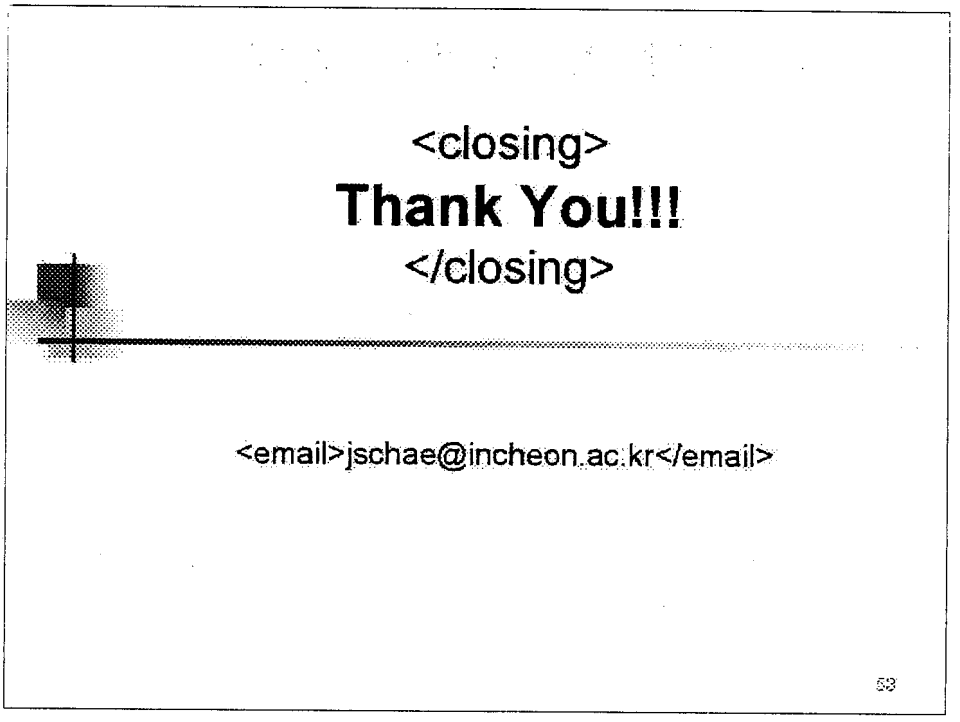

Recommendation 6
Element refinements should be treated in the same way as other properties
<dcterms:available>2002-06</dcterms:available>
rather than
<dc:date refinement="available">2002-06</dc:date>
or
<dc:date type="available">2002-06</dc:date>
or
<dc:date>
<dcterms:available> 2002-06 </dcterms:available>
</dc:date>
Internet Software Lab., Univ. of Incheon
48


Recommendation 7
Encoding schemes should be implemented using the 'xsi:type' attribute of the XML element for the property
<dc:identifier xsi:type="dcterms:URI">
http://www.ukoln.ac.uk/
</dc:identifier>
Internet Software Lab., Univ. of Incheon
49


Recommendation 8
Element refinements and encoding schemes should use the names specified in the DC Qualifiers recommendation
<dcterms:isPartOf xsi:type="dcterms:URI">
http://www.bbc.co.uk/
</dcterms:isPartOf>
<dcterms:temporal xsi:type="dcterms:Period">
name=The Great Depression; start=1929; end=1939;
</dcterms:temporal>
Internet Software Lab., Univ. of Incheon
50


Recommendation 9
Where the language of the value is indicated, it should be encoded using the 'xml:lang' attribute.
<dc:subject xml:lang="en">seafood</dc:subject>
<dc:subject xml:lang="fr">fruits de mer</dc:subject>
Internet Software Lab., Univ. of Incheon
51


Example DC Record
<?xml version="1.0"?>
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema.xsd">
<dc:title> UKOLN </dc:title>
<dcterms:alternative>
UK Office for Library and Information Networking
</dcterms:alternative>
<dc:subject xsi:type="dcterms:DDC"> 062 </dc:subject>
<dc:publisher> UKOLN, University of Bath </dc:publisher>
<dcterms:isPartOf xsi:type="dcterms:URI">
http://www.bath.ac.uk/
</dcterms:isPartOf>
<dc:identifier> http://www.ukoln.ac.uk/ </dc:identifier>
</metadata>
Internet Software Lab., Univ. of Incheon
52


<closing>
Thank You!!!
</closing>
<email>jschae@incheon.ac.kr</email>
68