XML DTD - Document logical structure

Thu 28.11.02

G. Falquet L. Nerima

Outline

- DTD
- Document modeling - DTD design
- DTD exercice
- Other modeling issues
- XML Schema - introduction

DTD - in brief

Document Type Definition (DTD)

Define the valid structure for a (set of) document(s)

What for ?

- Modeling
- XML DTD validating parser
- Customised XML text editors
- future applications

XML - reminder

XML is not a language but a meta-language

You must define your own language, namely your vocabulary (tags)

-> use DTD for tags definition … and more

Example - a bookstore DTD

```
<!ELEMENT bookstore (Book+)>
<!ELEMENT book (title, author, date, publisher)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
```

a document

```
<bookstore> <!-- this is a bookstore -->
    <book>
        <title>The XML companion (third edition)</title>
        <author>Neil Bradley </author>
        <date>2001</date>
        <publisher>Addison-Wesley</publisher>
    </book>
    <book> <!-- this is an other book --> …
    </book>
</bookstore>
```

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 5

DTD located in the document

The DTD helds within the document it describes

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE bookstore [
<!ELEMENT bookstore (Book+)>
<!ELEMENT book (title, author, date, publisher)>
…
]>
<bookstore> <!-- this is a bookstore -->
   <book>
     <title>The XML companion (third edition)</title>
     …
   </book>
   <book> <!-- this is an other book -->
     …
   </book>
</bookstore>
```

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 6

DTD in the document - drawback

DTD valid for one document only

redundancy (many documents with the same declarations)

maintenance

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 7

DTD in an external file

Document file

AbookStore.xml

```
<?xml version="1.0" encoding=… ?>
<!DOCTYPE bookstore
SYSTEM "BookStore.dtd"
>
<bookstore>
     <book>…</book>
     <book>…</book>
     …
</bookstore>
```

DTD file

BookStore.dtd

```
<!ELEMENT bookstore …>
<!ELEMENT book …>
<!ELEMENT title …>
<!ELEMENT author …>
<!ELEMENT date …>
<!ELEMENT publisher …>
```

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 8

DTD structure

composed of a number of declarations

<! … > markup must enclose each declaration

4 types of declaration
- ELEMENT (tag definition)
- ATTLIST (attribute definition)
- ENTITY (entity definition)
- NOTATION (non XML data type notation definition)

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 9

ELEMENT declaration

Tag definition

Structure declaration (remember classes in UML)

Element declaration:
<!ELEMENT element_name …>

rules for the element name:
- must begin with a **letter**, _ or a **:**
- continue with **letters**, **digits,** _ and **some ponctuation** (**.** and **-**)
- examples: p, X:123, aLongElementName

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 10

ELEMENT declaration: 3 kinds

Empty element
<!ELEMENT image EMPTY>

There is an image at this point <image …/> in the text.

Any element -> may contain all other elements declared in the DTD
<!ELEMENT p ANY>

Model group -> define the structure of the element
<!ELEMENT book (title, author, date, publisher)>

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 11

Model group

define an element that has mixed content (child elements, text, or mixture of childs and free text)

a model group is bound by brackets

(title, author, date, publisher)

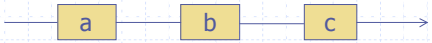
different kinds of child organization

I S I © G. Falquet, L. Nerima, CUI - Université de Genève 12

Sequence organization

, is the sequence connector

(a , b , c) define a sequential structure between elements

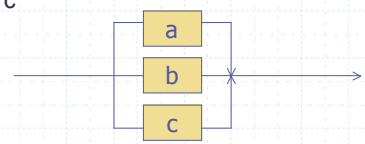


in the bookstore example, we stated that a book is a sequence of a title, an author, a date and a publisher element
<!ELEMENT book (title, author, date, publisher)>

Choice organization

| is the choice connector

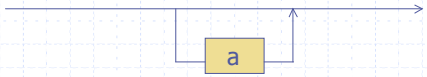
(a | b | c) denotes a choice between the elements a, b, and c



Option operator

? is the option indicator

(a?) indicates that the a element is optional

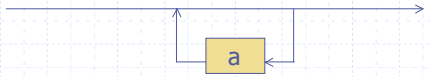


"a person have a name and possibly a tel number" is defined by
<ELEMENT! person (name, telNumber?) >

0 or many element

□ is the 0 or many indicator

(a□) indicates that element a may appear many times (possibly 0)

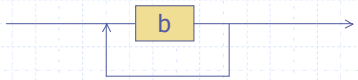


"an article may contain any number of author elements" is denotes by
<ELEMENT! article (author□) >

1 or many

+ is the at least one (possiby many) indicator

(a+) indicates that element a must appear, and may repeat



"a book requires at least one embedded chapter element"
<ELEMENT! book (chapter+) >

Text element

#PCDATA is the keyword indicates where text is allowed in a document (*parsable character data*)

#PCDATA represents 0 or more characters

Example
<!ELEMENT emph (#PCDATA)>

<emph>some text</emph>

Mixed content

Mixture of text (PCDATA) and child elements
rules:
- the PCDATA must be the first token in the group
- the group must be a choice group
- the group must be repeatable (0 or many)

<!ELEMENT emph (#PCDATA | sub | super)* >
<!ELEMENT sub (#PCDATA) >
<!ELEMENT super (#PCDATA) >

<emph>H₂0 is water.</emph>

Model group ambiguities

a classical problem in parsing theory

there is no connector precedence
-> (a , b , c | d) is ambiguous

mix of , and | connector is not legal

use additional brackets
((a , b , c) | d) -> d is an alternative to all
(a , b , (c | d)) -> d is an alternative to c

Model group ambiguities (cont)

the parser is supposed to be a "one token ahead" parser

example of ambiguity
(item? , item)
-> solution
(item , item?)

((surname, employee) | (surname, person))
-> solution ?

Attribute declaration

separate from the element

refers to the element that will contain the attributes

general form:

```
<!ATTLIST element_name      att_name  type defValue
                            att_name  type defValue
                            …         …       … >
```

the type parameter defines the type of the attribute

a defValue (default value) may be specified but is optional

Attribute declaration example

three attributes are defined for use in the Sequece List element «seqlist»

```
<!ATTLIST seqlist       first CDATA
                        offset NMTOKEN
                        type ( alpha | number )>
```


Attribute type

CDATA	string of characters
NMTOKEN	a word or a token (name rules apply)
NMTOKENS	serie of tokens
ENTITY	entity reference
ENTITIES	serie of ..
ID	hypertext linking
IDREF	"
IDREFS	"
NOTATION	embedded non XML data (e.g TeX, tiff)
name token group	set of possible token values

ID and IDREF type

XML basic links

ID type attribute defines a target element

IDREF type attribute defines an source element which make a reference to the target element

ID and IDREF type - example

```
<!ELEMENT section (…) >
<!ATTLIST section target ID #REQUIRED >

<section target="S6">
<title>This is section 6</title>
…
</section>


<!ELEMENT xref (…)
<!ATTLIST xref link IDREF #REQUIRED>

<para>Please refer to <xref link="S6">Section 6</xref> for more detail
```


Default attribute values

specifies a default value for an attribute if the document author does not enter a value

or states that an author must enter a value
#REQUIRED

or states that an attribute can be absent
#IMPLIED

Default value example

```
<ATTLIST seqlist    sepchar NMTOKEN #REQUIRED
                    offset NMTOKEN #IMPLIED
                    type (alpha | num) "num" >

<seqlist sepchar="; "> … </seqlist>

<seqlist sepchar="; " offset="5mm"> … </seqlist>

<seqlist sepchar="; " type="alpha">…/seqlist>
```

Entity declarations

define a general entity that will be used in the document

<!ENTITY CUI "Centre universitaire d'informatique">
<!ENTITY UN "United Nations">

<report> ... &UN; ... &CUI; ...

Predefined entities:
< < , > >, & &, ' ', " "

Parameter entity definition

define a general entity that is used in the DTD

share a commun structure

Example:
<!ENTITY % common "(para | list | table)">

<!ELEMENT chapter ((%common;)+, section+)>
<!ELEMENT section (%common;)+>

Notation declaration

an element or entity may contain non-XML format data

specifies which format may be embedded

specifies the location of the application can process the data (if available)
<!NOTATION TIFF SYSTEM "c:\apps\show_tiff.exe">

Document modeling - DTD design

Document analysis

Future-use analysis
- information reuse
- XML - database interaction
- exploit hypertext links
- target multiple application

Granularity

or
<name>John Smith</name>
<name><f>John </f><s>Smith</s></name>

Similar to static structure modeling with UML classes

Element or attribute ?

information represented by an element OR an attribute

guideline:

- nested structure -> choice an element declaration
- small unit, no nesting -> attribute
- in doubt -> element

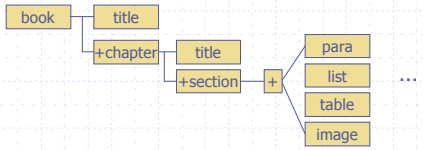
Element or attribute - example

```
<city>Paris</city>
<item>
   <category>food</category>
   <desc>A special cake made of ... </desc>
   <price>SFr. 56.30</price>
</item>

<item category="food" price="SFr. 56.30">
   <desc>A special cake made of ... </desc>
   <evaluation>no toxic element found</evaluation>
</item>
```


DTD - hierarchical structure

example: a book



remark: more parent elements may refer to the same child element -> in that case, the hierarchy is not strict (not a tree)

The "ECE Monthly" exercice

design a DTD for the "ECE Monthly"

focus on information structure (not on document layout)

DTD element declaration summary

sequence	<!ELEMENT name(child1, child2,…,childn)>
choice	<!ELEMENT name(child1| child2 |…| childn)>
repetition	<!ELEMENT name(child1*) > <!ELEMENT name(child1, child2,…,childn)*>
repetition	<!ELEMENT name(child1+)> <!ELEMENT name(child1, child2,…,childn)+>
option	<!ELEMENT name(child1?)>

A solution for the "ECE Monthly" exercice

<!ELEMENT ece-monthly (issue-number, issue-month, table, just-issued*, end-note)

<!ELEMENT table (meeting-group | meeting)*>

<!ELEMENT meeting-group(topic, meeting*) >

<!ELEMENT meeting(dates, location, title, comments)>

<!ELEMENT just-issued (publi-title, publi-ref, ISBN?, price?)>

<!ELEMENT end-note(text, address, tel, fax, e-mail, url)>

<!ELEMENT issue-number (#PCDATA)>

<!ELEMENT issue-month (#PCDATA)>

…etc.

Solution variation - use of attributes

an alternative solution for the meeting element:

<!ELEMENT meeting (title, comments)>

<!ATTLIST meeting dates CDATA #REQUIRED
location CDATA "Palais des Nations Geneva">

if no value is specified, the default location is the "Palais des Nations Geneva"

title and comments are modeled as elements because they have more complex inner structure (that will be possibly modeled in the future)