

eXtensible Markup Language

XML

Alexis NEDELEC

LISYC EA 3883 UBO-ENIB-ENSIETA
Centre Européen de Réalité Virtuelle
Ecole Nationale d'Ingénieurs de Brest

enib ©2009



Origine du XML

Un peu d'Histoire

- 1969 : **ARPA** (Advanced Research Project Agency)
- 1970 : **ARPANET**(work) : Network Control Protocol (NCP)
- 1972 : **INWG** (InterNetwork Working Group) EU + UE,
- 1973 : **TCP/IP**
(Transmission Control Protocol/Internet Protocol)
- 1980 : dans les domaines scientifique et publique
- 1990 : **HTML** (HyperText Markup Language)
T. Berners-Lee (CERN)

Le trio gagnant

- TCP/IP + HTTP + HTML
- fondements du World Wide Web (W3)

HTML

Une page personnelle

```
<html>
  <head>
    <title> Ma page Personnelle </title>
  </head>
  <body>
    <h1> section de ma page </h1>
    <h2> sous-section de la page </h2>
    <p> premier paragraphe </p>
    <p> deuxieme paragraphe </p>
  </body>
</html>
```

HTML

La même page personnelle

```
<html>
  <head>
    <title> Ma page Personnelle </title>
  </head>
  <body>
    <b>
      <font size="6">section de ma page </font><br><br>
      <font size="5">section de ma page </font><br><br>
    </b>
    <font size="4"> premier paragraphe </font><br><br>
    <font size="3"> deuxieme paragraphe </font><br><br>
  </body>
</html>
```

SGML

Standard Generalized Markup Language

- séparer la structure logique d'un document
 - titres, chapitres, paragraphes ...
- de sa mise en page
 - livre, journal, écran ...

Normalisation de documents

- **GED** : Gestion Electronique des Documents
- **EDI** : Echange de Données Informatisées
- **XML** : e**X**tensible Markup **L**anguage

Présentation

Description

- textuel simple à écrire, raisonnablement lisible
- auto-descriptif, règles strictes de validation
- multilingue (Unicode), programmable (arbre)

Parseur XML : MSXML, Xerces, libXML ...

- cohérence des documents XML (syntaxe, conformité)
- chargement de l'arbre du document en mémoire
- accès aux éléments, noeuds du document (tokens)

Processeur XML : MSXML, Xalan, libXSLT ...

- modifier, transformer un document XML
- XSLT (XML Stylesheet Language Transformation)

Présentation

Objectifs

- **HTML** : présentation (affichage) de données,
- **XML** : représentation (description) de données

Structuration de l'information

- **DTD** : règles de validation XML
- **XML Schemas**: amenés à remplacer les DTD (spécifications W3C)
- **XSLT**: transformation de documents XML
- **XPath**: associé à XSLT pour naviguer dans un arbre XML

Outils XML

Manipulation, contrôle, liaison de documents XML

- **X Query**: langage de requêtes pour accéder au contenu
- **DOM** : XML objet manipuler le document XML
- **SAX** : XML événementiel, pour identifier des parties
- **XLink**: navigation hypertexte entre documents XML
- **XPointer**: accès aux sous-ensembles de documents XML
- **XForms**: équivalent des formulaires HTML

Document XML

Structuration de documents

- prologue, instructions de traitement
- arbre d'éléments

Exemple de document

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:output method="html"/>
<html>
  <head>
    <title> Ma page Personnelle </title>
  </head>
  <body>
    ...
  </body>
</html>
```

Syntaxe XML

Briques de construction

- Déclarations: `<?xml version="1.0"?>`
- Balises: `<tag>...</tag>`, marquage des éléments
- Éléments: `<tag>balise avec son contenu</tag>`
- Attributs: information sur les éléments:
`...`
- Instructions de traitement : `<?cible valeur?>`
- Entités (référence sur): variables communes aux documents XML (` `; `>`; `"`; `'`; `&`...)
- PCDATA: données à traiter par le parseur XML
- CDATA: données brutes, non-traitées par le parseur XML

Syntaxe XML

Entités XML

```
<?xml version="1.0"? encoding="ISO-8859-1">
<!DOCTYPE monDocument [
<! ENTITY monEntite "toto">
]>
<monDocument>
  mon entité est : &monEntite;
</monDocument>
```

Données brutes

- `<inegalite> deux<trois>un </inegalite> : KO`
- `<inegalite> [CDATA [deux<trois>un]] </inegalite> : OK`

Exemple de document

```
messages.xml
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<message>
  <expediteur>Nedelec</expediteur>
  <destinataire> EdT </destinataire>
  <sujet>A propos de XML</sujet>
  <contenu> quel jour ? </contenu>
</message>
```

Remarques

- utilisation de XML version 1.0
- codage de jeux de caractères :
 - ASCII, Unicode, UTF-8 ...
- définir les balises <message>, <destinataire>, ...

Element XML

Un élément XML contient

- uniquement des éléments (element content)
- des éléments et des données (mixed content)
- des données simples, donc du texte (simple content)
- être vide (empty content)
- avoir des attributs (ex: `<message date="01/02/03">`)

Sur l'exemple précédent

- `<destinataire>Nedelec</destinataire>` : est un élément
- `destinataire` : est l'appellation de l'élément
- `Nedelec` : est la donnée de l'élément
- PCDATA (Parsed Character DATA) :
le contenu d'un élément qui ne contient pas d'autres balises

Element XML

Element ou attribut ?

```
<message date="01/02/03"> ... </message>
```

```
<message>
```

```
  <date>01/02/03</date>
```

```
  ...
```

```
</message>
```

```
<message>
```

```
  <day>01</day>
```

```
  <month>02</month>
```

```
  <year>03</year>
```

```
  ...
```

```
</message>
```

Attribut XML

Element ou attribut ?

- les attributs ne peuvent pas contenir de valeurs multiples
- les attributs ne peuvent pas décrire des structures
- on peut enrichir les éléments pour les faire évoluer
- les attributs sont difficiles à manipuler par des programmes

Contre-Exemple

```
<message day="01" month="02" year="03"  
  from="Nedelec" to="EdT"  
  subject="A propos de XML"  
  content="quel jour ?"  
</message>
```

Attributs XML

Exemple d'utilisation d'attributs

```
<messages>
  <message id="001">
    <from>Nedelec</from>
    <to>EdT</to>
    <subject>A propos de XML</subject>
    <content>quel jour ?</content>
  </message>
  <message id="002">
    <from>EdT</from>
    <to>Nedelec</to>
    <subject>Re: A propos de XML</subject>
    <content>le: <day>01</day>...<year>03</year></content>
  </message>
</messages>
```

Syntaxe XML

Règles syntaxiques

- tout document a un élément racine :
`<root></root>`
- toute balise doit-être ouverte et fermée :
`<tag>...</tag>` ou `<tag... />`
- les balises sont sensibles à la "Casse" :
`<tag>`, `<Tag>`, `<TAG>`
- tout sous-élément (fils) doit-être correctement imbriqué:
`<root>`
 `<child><subchild>`
 `...`
 `</subchild></child>`
`</root>`

Syntaxe XML

Règles syntaxiques

- définition des attributs dans la balise ouvrante :
`<Chapitre numero="1">`
- valeurs d'attributs entre guillemets :
`<Chapitre numero="1" media="paper">`
- neutraliser les caractères spéciaux par CDATA :
 - `<inegalite>[CDATA[deux<trois>un]]</inegalite>`
- conventions de nommage de balise, éviter les .,-
`<nom_de_balise>` : OK, `<nom.de-balise>` : KO
- commentaire: `<!-- Ceci est un commentaire -->`

Syntaxe XML

Document XML syntaxiquement correct

Well formed XML document

- le document XML a une balise racine
- les éléments XML sont fermés
- ils sont sensible à la "case"
- ils sont bien imbriqués
- la valeur des attributs XML est entre "guillemets"

Validation des documents XML

valid XML document

- Document Type Definition (DTD)
- XML schema (XSD)

DTD :Document Type Definition

Structuration de document XML

- le document `<!DOCTYPE ...>`
- les éléments `<!ELEMENT ...>`
- les attributs `<! ATTLIST ...>`
- les entités `<! ENTITY ...>`
- les données `CDATA`, `#PCDATA`

Document XML valide

- document XML bien **formé**
- respecte les règles de DTD interne et/ou externe

DTD :Document Type Definition

DTD interne

```
<?xml version="1.0"?>
<!DOCTYPE messages [
  <!ELEMENT message (from,to,subject,content)>
  <!ELEMENT from      (#PCDATA)>
  <!ELEMENT to        (#PCDATA)>
  <!ELEMENT subject   (#PCDATA)>
  <!ELEMENT content   (#PCDATA)> ]>
<messages><message>
  <from>Nedelec</from/>
  <to> EdT </to>
  <subject>A propos de XML</subject>
  <content>quel jour ? </content>
</message></messages>
```

DTD :Document Type Definition

DTD externe (message.dtd)

```
<!ELEMENT message (from,to,subject,content)>  
<!ATTLIST message id CDATA #REQUIRED>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT subject (#PCDATA)>  
<!ELEMENT content (#PCDATA)> ]>
```

DTD externe (message.xml)

```
<?xml version="1.0"?>  
<!DOCTYPE messages SYSTEM "message.dtd">  
<messages><message id="001">  
  ...  
</message></messages>
```

XSD : XML Schema Definition

XML Schema

- alternative au DTD
- recommandations W3C (mai 2001)
- langage de schéma XSD

XML Schema vs DTD

- ils sont écrits en XML
- auto-descriptifs (même parseur, même règles)
- types plus riches (booléens, numériques, dates ...)
- 1 document XML / plusieurs schémas (espace de nommage)

Document XSD

Document XSD : message.xsd

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="message">
<xsd:complexType>
  <xsd:sequence>
    <xsd:element name="from" type="xsd:string"/>
    <xsd:element name="to" type="xsd:string"/>
    <xsd:element name="subject" type="xsd:string"/>
    <xsd:element name="content" type="xsd:string"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Document XSD

Racine du document XSD : <schema>

```
<?xml version="1.0"?>
<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  targetNamespace="http://www.monschema.org"
  xmlns="http://www.monschema.org"
  elementFormDefault="qualified">
  ...
</xsd:schema>
```

Attributs de l'élément <schema>

- `xmlns:...` : espace de nommage d'éléments
- `targetNamespace` : espace de nommage du schéma
- `elementFormDefault` : nommage des éléments du schéma

Liaison XML / XSD

Document XML

```
<?xml version="1.0"?>
<message
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  xsi:noNamespaceSchemaLocation="message.xsd">
  <from>Nedelec</from>
  <to>EdT</to>
  <subject>A propos de XML</subject>
  <content> Quel jour ?</content>
</message>
```

Liaison XML / XSD

Avec un espace de nommage

```
<?xml version="1.0"?>
<message xmlns:"http://www.monschema.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  xsi:schemaLocation=
    "http://www.monschema.org message.xsd">
  <from>Nedelec</from>
  <to>EdT</to>
  <subject>A propos de XML</subject>
  <content> Quel jour ?</content>
</message>
```

Définition d'un élément, attribut

`<element>` : définition d'un élément

- `<element name="aName" type="built-in-type"/>`

Types prédéfinis

- `decimal`, `integer`, `boolean`
- `string`, `date`, `time`

`<element>` : autres attributs

- `default` : une valeur par défaut modifiable
- `fixed` : une valeur par défaut non-modifiable

`<element>` : exemple

```
<xsd:element name="color"  
             type="xsd:string" default="red"/>
```

Définition d'un élément, attribut

<attribute> : définition d'un attribut

- `<attribute name="aName" type="built-in-type"/>`

Types prédéfinis

- decimal, integer, boolean
- string, date, time

<attribute> : autres attributs

- default : une valeur par défaut modifiable
- fixed : une valeur par défaut non-modifiable

<attribute> : exemple

```
<xsd:attribute name="color" type="xsd:string"
               default="red" use="required"/>
```

Valeurs d'éléments : <restriction>

<restriction> : sur une plage de valeur

```
<xsd:element name="age">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:minInclusive value="0"/>
      <xsd:maxInclusive value="120"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Valeurs d'éléments : <enumeration>

<enumeration> : sur un ensemble de valeurs

```
<xsd:element name="voiture">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Renault"/>
      <xsd:enumeration value="Citroen"/>
      <xsd:enumeration value="Peugeot"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Valeurs d'éléments : <pattern>

<pattern> : sur une série de valeurs

```
<xsd:element name="letter">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-z]"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Valeurs d'éléments : <length>

<length> : sur la longueur

```
<xsd:element name="password">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
<!--      <xs:length value="8"/> -->
      <xsd:minLength value="5"/>
      <xsd:maxLength value="8"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Element complexes : `<complexType>`

`<complexType>` : types d'éléments complexes

- contient d'autres éléments, attributs
- élément vide
- ne contient que des éléments
- ne contient que du texte
- contient des éléments et du texte

Element complexes : <complexType>

<complexType> : séquence d'éléments

```
<xsd:element name="message">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="from" type="xsd:string"/>
      <xsd:element name="to" type="xsd:string"/>
      <xsd:element name="subject" type="xsd:string"/>
      <xsd:element name="content" type="xsd:string"/>
    </xsd:sequence>
    <xsd:attribute name="id" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
```

Réutilisation d'éléments complexes

Attribut type

```
<xsd:element name="personalMsg" type="message"/>
<xsd:element name="publicMsg" type="message"/>

<xsd:complexType name="message">
  <xsd:sequence>
    <xsd:element name="from" type="xsd:string"/>
    . . . .
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:string" />
</xsd:complexType>
```

Réutilisation d'éléments complexes

éléments `<extension>`, `<complexContent>`

```
<xsd:element name="publicMsg" type="message"/>
<xsd:element name="privateMsg" type="fullMessage"/>
<xsd:complexType name="message">
  <xsd:complexContent>
    ...
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="fullMessage">
  <xsd:complexContent>
    <xsd:extension base="message">
      <xsd:element name="affinity" type="xs:string"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

Visualisation des données

Erreur syntaxique

```
XML Parsing Error: mismatched tag. Expected: </from>.
Location: file:///home/nedelec/message.xml
Line Number 3, Column 23:
```

```
<from>Nedelec</From>
-----^
```

Sans feuille de style

```
Nedelec EdT A propos de XML Quel jour ?
```

Cascading Style Sheets (CSS)

Feuille de style (message.css)

```
message {  
background-color: #ffffff; width: 100%; }  
expediteur {  
font-size: 20pt; margin-bottom: 30pt; margin-left: 0; }  
destinataire {  
font-size: 20pt; margin-bottom: 30pt; margin-left: 0; }  
sujet {  
display: block;  
color: #FF0000; font-size: 15pt;}  
contenu {  
color: #0000FF; font-size: 20pt;}  
}
```

Cascading Style Sheets (CSS)

Document XML et feuille de style

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="message.css"?>
<message>
  <from>Nedelec</from>
  <to>EdT</to>
  <subject>A propos de XML</subject>
  <content> Quel jour ?</content>
</message>
```

eXtensible Stylesheet Language (XSL)

Règles de transformation ("/messages")

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="message.css"?>
<xsl:template match="/messages">
  <html><body>
    <h2>Mes messages</h2>
    <xsl:apply-templates/>
  </body></html>
</xsl:template>
```

eXtensible Stylesheet Language (XSL)

Règles de transformation ("message")

```
<xsl:template match="message">
  <p>
    <xsl:apply-templates select="from"/>
    <xsl:apply-templates select="to"/>
    <xsl:apply-templates select="subject"/>
    <xsl:apply-templates select="content"/>
  </p>
</xsl:template>
```

eXtensible Stylesheet Language (XSL)

Règles de transformation ("from", "to")

```
<xsl:template match="from">
  Expéditeur: <span style="color:#ff0000">
    <xsl:value-of select="."/></span>
  <br />
</xsl:template>

<xsl:template match="to">
  Destinataire: <span style="color:#000fff">
    <xsl:value-of select="."/></span>
  <br />
</xsl:template>
```

eXtensible Stylesheet Language (XSL)

Règles de transformation ("subject", "content")

```
</xsl:template>
<xsl:template match="subject">
  Sujet: <span style="color:#ff0000">
    <xsl:value-of select="."/></span>
  <br />
</xsl:template>

<xsl:template match="content">
  Contenu: <span style="color:#ff0000">
    <xsl:value-of select="."/></span>
  <br />
</xsl:template>
```

Bibliographie

Ouvrages

- **D. Hunter** “Initiation à XML”
ed. Eyrolles Wrox Press 2000
- **S. Lecomte** “XML par la pratique”
ed. ENI 2005
- **K. Williams et. al.** “XML et les Bases de Données”
ed. Eyrolles Wrox Press 2000
- **A. Brilliant** “XML Cours et exercices”
ed. Eyrolles Wrox Press 2007
- **G. Chagnon, F. Nolot** “XML”
ed. Pearson Education, coll. Synthex 2007

Bibliographie

Liens Au Net

- le consortium W3C : www.w3.org
- l'organisme : www.xml.org
- les tutoriaux : www.w3schools.com
- cours en ligne de Gilles Chagnon :
www.licence.elec.upmc.fr/S_tec/coursEnLigne/xml/index
- Le club d'entraide des développeurs : www.developpez.com
- ... recherche sur www.google.org