

Autour d'XML

1

Autour de XML

- Hyperdocuments (liens et pointeurs) : Xlink et Xpointer
- Méta-données pour XML : RDF
- Programmer avec XML : SAX, DOM
- Communiquer avec XML : SOAP

2

Hyperdocuments XML

Xlink et XPointer

3

Faiblesses de HTML/liens

- Pas de désignation facile d'un morceau de document (juste insertion d'étiquette)
- pas de liens bidirectionnels ou à cibles multiples, pas de sémantique sur les liens, pas d'ajout de lien sur un document sans le modifier

4

XLL

- XLL = XPointer + XLinks
- XPointer = désignation d'une partie de documents
- XLinks = liens entre documents
- Standard pas encore stabilisé et surtout peu (pas ?) supporté par les outils

5

XLinks

- Différents types de liens (simple, étendu, inclus, externe)
- lien simple : analogue ancre HTML
- attributs d'un lien :
 - title et role (optionnel pour sémantique)
 - show (affichage du document pointé) : replace, new, embed, other, none
 - actuate (traversée du lien, automatique ou non) : onRequest, onLoad, other, none

6

Exemple de lien simple

```
<SEARCH xlink:type="simple"
xlink:href="http://www.google.fr"
xlink:title="Recherche avec Google"
xlink:show="replace"
xlink:role="http://www.google.fr/help.html">
Recherche avec Google
</SEARCH>
```

7

DTD associée

- Si le document est typé, la DTD doit contenir la déclaration des attributs du lien

```
<!ELEMENT SEARCH (#PCDATA)>
<!ATTLIST SEARCH
  xmlns:xlink CDATA #FIXED "http://www.w3.org/1999/xlink"
  xmlns:type CDATA #FIXED "simple"
  xmlns:href CDATA #REQUIRED
  xmlns:role CDATA #IMPLIED
  xmlns:title CDATA #IMPLIED
  xmlns:show CDATA #IMPLIED>
```

8

Liens étendus

- Lien étendu est un graphe orienté entre documents
- ce graphe peut être défini à l'extérieur des documents pointés (personnalisation possible de documents, indépendance de la navigation par rapport aux documents)

9

Description d'un lien

- Un lien étendu est décrit par un élément XML composé de sous-éléments (**éléments de liaison**) qui décrivent les constituants de ce lien :
 - locators : ressources distantes,
 - resource : ressources locales,
 - arc : arcs,
 - etc.
- Ces éléments de liaison, ont des attributs spécifiques :
 - type : type de l'élément,
 - href : URI d'une ressource distante,
 - role, title : sémantique d'un lien, d'une ressource ou d'un arc,
 - actuate, show : comportement d'un arc, c'est à dire quand et comment est affichée la ressource d'arrivée.

10

Exemple de lien étendu (1)

```
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink"
  xlink:type="extended" >
  <NAME xlink:type="resource">Cafe au lait</NAME>
  <HOMESITE xlink:type="locator"
    xlink:href="http://ibiblio.org/javafaq"/>
  <MIRROR xlink:type="locator"
    xlink:href="http://sunsite.kth.se/javafaq"/>
  <MIRROR xlink:type="locator"
    xlink:href="http://sunsite.cnlab-switch.ch/javafaq"/>
</WEBSITE>
```

11

Exemple de lien étendu (2)

```
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink"
  xlink:type="extended" >
  <NAME xlink:type="resource" xlink:label="source" >
    Cafe au lait</NAME>
  <HOMESITE xlink:type="locator" xlink:label="hs"
    xlink:href="http://ibiblio.org/javafaq"/>
  <MIRROR xlink:type="locator" xlink:label="se"
    xlink:href="http://sunsite.kth.se/javafaq"/>
  <MIRROR xlink:type="locator" xlink:label="ch"
    xlink:href="http://sunsite.cnlab-switch.ch/javafaq"/>
  <CONNECTION xlink:type="arc" xlink:from="source" xlink:to:"ch"
    xlink:show="replace" xlink:actuate="onRequest"/>
  ...
</WEBSITE>
```

12

XPointer

- Capacité d'adresser une partie de document sans marquage préalable de celui-ci
- comment faire charger à un navigateur une partie seulement d'un document (pb http, non résolu)
- est ce qu'une partie de document XML est forcément bien formée ? (non, pb non résolu)

13

Comment désigner

- On veut une désignation plus riche que la marquage
- le 3ème paragraphe, l'élément de structure de nom « titi »
- idée : document XML est une structure d'arbre dont les nœuds sont les éléments de structure
- langage de désignation portant sur des références absolues (racine, nœud de nom XXX) et relatives (3ème fils, ...)

14

Exemples de désignation

- `http://----- #xptr(id(« ebnf »))`
- `http://-----#1/14/2`
- `http://-----
#xptr(descendant::language[position()=2])`
- language : nom de l'élément
- descendant : nœud fils du document
- position()=2 : deuxième fils

15

Méta-données pour XML

Resource Description Framework

16

RDF

- Resource Description Framework
- besoin d'avoir un référentiel commun pour exprimer des recherches sur des documents
- Dublin Core : première proposition de référentiel de documents (ensemble de descripteurs généraux prédéfinis et non modifiable : titre, créateur, sujet, ...)
- besoin de construire des référentiels par domaine

17

RDF

- RDF méta-modèle pour construire des référentiels
- approche très générale (ensemble de triplets ressource, propriété, valeur)
- `http://www-inf.int-evry.fr/~defude -> créateur -> "bruno defude"`

18

Exemple de RDF

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-
  rdf-syntax-ns#"
  xmlns:dc="http://purl.org/DC/"
  <rdf:Description
    about="http://mica/~oracle/">
    <dc:CREATOR>Bruno Defude</dc:CREATOR>
    <dc:TITLE>Tutoriel BD INT</dc:TITLE>
  </rdf:Description>
</rdf:RDF>
```

19

Programmer avec XML

SAX, DOM

20

Programmer avec XML

- Java le plus abouti
- C, Perl, Python etc. aussi
- support Unicode est le plus compliqué
- SAX, Simple Api for Xml (dirigé par les événements) -> SAX2
- DOM, the Document Object Model (permet de manipuler l'arbre)

21

Exemple SAX

Document en entrée

```
<message>
  <date>1998-08-03</date>
  <expediteur>Pierre Truc</expediteur>
  <destinataire>Paul Savon</destinataire>
  <corps>Bonjour, il fait beau</corps>
</message>
```

22

Sortie souhaitée

Début élément racine : message

Balise ouvrante : date

Fin de l'élément : date

Balise ouvrante : expediteur

Fin de l'élément : expediteur

...

23

Programme Java / SAX

```
import org.xml.SAX.HandlerBase;
import org.xml.SAX.AttributeList;
public class MonHandler extends HandlerBase {
    public void startElement (String name, AttributeList atts) {
        System.out.println("Balise ouvrante : "+name); }
    public void endElement (String name) {
        System.out.println("Fin de l'element : "+name); }
}
```

24

Application Java/SAX

```
import org.xml.SAX.Parser;
import org.xml.SAX.DocumentHandler;
import org.xml.SAX.helpers.ParserFactory;
public class MonAppli {
static final String parserClass ="foo.bar.XML.SAXDriver";
public static void main (String args[]) throws Exception {
    // creation d'un nouveau parser
    Parser parser = ParserFactory.makeParser(parserClass);
    // creation d'un nouveau handler
    DocumentHandler handler= new MonHandler();
    // enregistrement du handler
    parser.setDocumentHandler(handler);
    // traitement du document passe en argument
    for (int i= 0;i <args.length; i++) { parser.parse(args[i]); }
}}
```

25

Communiquer avec XML

Simple Object Access Protocol

26

SOAP

- Simple Object Access Protocol
- SOAP = envoi de message ou RPC en XML
- Protocole de transport : pas fixé mais souvent HTTP
- XML = représentation de l'information

27

Objectifs de SOAP

- HTTP est un modèle de programmation réparti trop simple (pas de support de session, passage de paramètres simpliste)
- mais facile à déployer, très utilisé, passe bien les firewalls
- SOAP peut être vu comme une couche applicative sur HTTP

28

Philosophie SOAP

- HTTP + XML : standardisé par le W3C
- protocole minimal pour appeler des méthodes sur des serveurs, services, objets, composants
 - sans imposer une API ou un runtime
 - ne pas imposer un ORB ou un httpd particulier
 - ne pas imposer un modèle de programmation (multi-modèles)
 - ne pas réinventer une nouvelle technologie
- portable sur toute plate-forme et technologie

29

C'est quoi SOAP

- **Modèle de message** (Enveloppe : structure du msg, entête : partie non applicative i.e transactions, corps)
- **règles d'encodage** : mécanisme de sérialisation permettant de construire le msg à partir de chaque type de données pouvant être échangé
- **fonctionnement en modèle C/S** (représentation des appels de procédure et des réponses)
- **une mise en oeuvre sur HTTP (et d'autres)**

30

Conclusion

- XML semble un bon compromis entre SGML et HTML
- Quelques problèmes de stabilité (c'est jeune)
- certaines parties sont puissantes mais complexes!
- Support côté client est faible
- plus orienté utilisation professionnelle que page perso
- attention à la mode "tout XML"

31

Bibliographie

- XML langages et applications, Alain Michard, 2nd édition, Eyrolles, 2001
- XSLT Programmer's Reference, Michael Kay, Wrox Press, 2000
- <http://www.w3c.org>
- <http://www-inf.int-evry.fr/~defude/XML>

32