Développer des projets en XML

XML et la composition de documents

Introduction pratique de XML

- XML permet de se concentrer sur la structuration et la spécification de la nature du contenu des données.
- Les langages de style et transformation (CSS, XSL-T) décrivent comment l 'information doit être présentée:

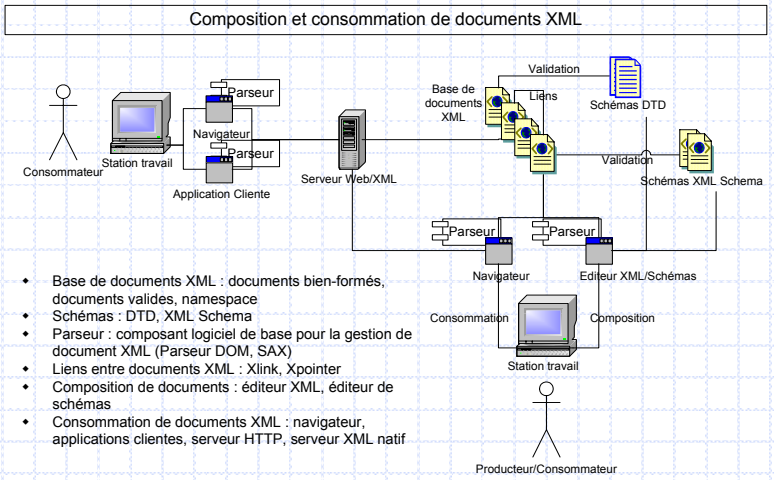
  HTML : Programmation Java et XML

  XML:

  <book-title> Programmation Java et XML</book-title>

  book-title {font-style: italic;}
- XML permet à la fois de spécifier le contenu (data) et la nature du contenu (metadata).

Architecture système et XML



- Base de documents XML : documents bien-formés, documents valides, namespace
- Schémas : DTD, XML Schema
- Parseur : composant logiciel de base pour la gestion de document XML (Parseur DOM, SAX)
- Liens entre documents XML : Xlink, Xpointer
- Composition de documents : éditeur XML, éditeur de schémas
- Consommation de documents XML : navigateur, applications clientes, serveur HTTP, serveur XML natif

Introduction pratique de XML

Exemple de tableau de données chiffrées

(Dollars in Millions)

	Notes	1997	1996
Current Assets			
Cash and cash equivalents		$ 7,106	$ 7,687
Marketable securities	J	447	450
Notes and accounts receivable		16,850	16,515

Introduction pratique de XML

- HTML se concentre sur la présentation.

```
<i>(Dollars in millions)</i>
<table>
  <tr><th></th><th>1997</th><th>1996</th></tr>
  <tr>
    <td>Current assets</td>
    <td></td><td></td>
    </tr>
  <tr>
    <td>Cash and cash equivalents</td>
    <td>$ 7,106</td><td>$ 7,687</td>
    </tr>
  <tr>
    <td>Marketable securities</td>
    <td>447</td><td>450</td>
    </tr>
```


Introduction pratique de XML

- Un langage conçu à l'aide d'XML permet de se concentrer sur la structuration du contenu, selon les besoins de l'application.

1. `<amount>41,000,000</amount>`
2. `<amount period="1998">41,000,000</amount>`
3.
```
<asset>
  <amount period="1998">41,000,000</amount>
  </asset>
```

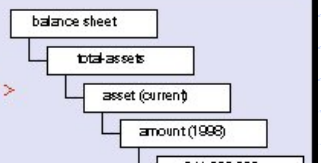
Introduction pratique de XML

- XML spécifie le contenu:
  - ex. 1 indique que le texte est un montant
  - ex. 2 indique que le texte est un montant et l 'attribut période nous donne une information supplémentaire
  - ex. 3 complète l 'information en incluant le montant dans un contexte de « résultat »

Introduction pratique de XML

- La description peut ainsi être complétée si nécessaire afin de créer une structure hiérarchique naturelle

```
<balance-sheet>
   <total-assets>
      <asset type="current">
         <amount period="1998">
            $41,000,000
            </amount>
         </asset>
      </total-assets>
   </balance-sheet>
```

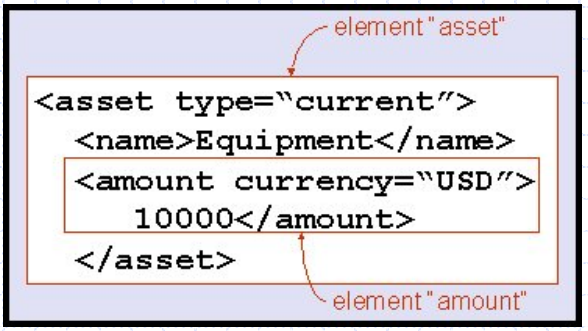


Le contenu d 'un document XML

- Les différents composants d 'un document XML et la terminologie sont :
  - Balises
  - Eléments
    - contenu
    - parent
    - enfant
  - Attributs
  - Entités

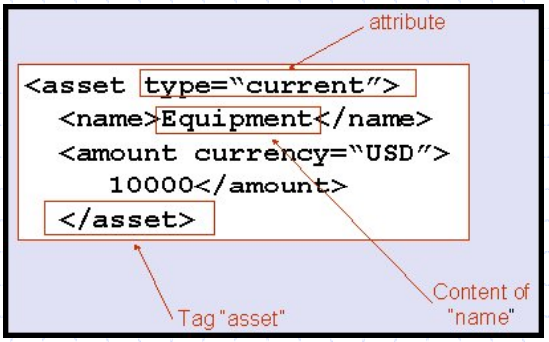
Elément d 'un document XML

- Un élément est une balise et son contenu



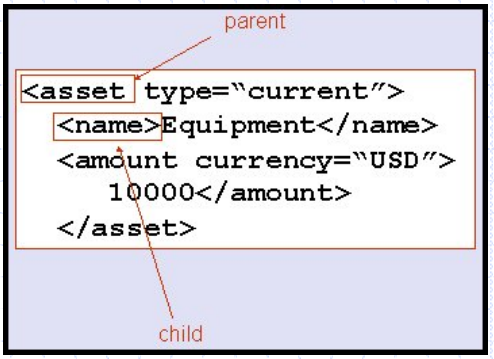
Elément d 'un document XML

- Les attributs qualifient les éléments.



Elément d 'un document XML

- Les éléments sont imbriqués : parent, enfant, frère



Type de document XML

- Distinction entre:
  - **document well-formed (bien-composé)** : respectant la syntaxe et grammaire de XML
  - **document valide** : la description du document fait l'objet d'un schéma (DTD, document type definitionou XML Schema), repris dans le document (DTD dans le prolog) ou dans un document externe.

Régles de syntaxe de base de XML

- Les règles suivantes doivent être respectées, aussi bien pour les documents "bien-formé" que pour les "valides":
  - contenir au moins un élément
  - le document complet doit être contenu entre une paire unique de balises, appelée élément racine (root element) ou élément document.
  - toutes les autres balises seront imbriquées et ne peuvent pas se recouvrir.
  - tout élément doit avoir une balise ouvrante et une fermante: <produit></produit>

    <produit/> (élément vide)

Régles de syntaxe de base de XML

- les éléments doivent être imbriqués et ne peuvent pas se superposer :

  <asset><amount></amount></asset>

  Mauvais :

  <asset><amount></asset></amount>

- XML est sensible à la casse (minuscule, majuscule) :

  <Produit>…</produit> (incorrect)

  <Produit>…</Produit> (ok)

Régles de syntaxe de base de XML

- Exemples:
  - "Bonjour le monde !" (mauvais)
  - <texte> "Bonjour le monde !"</texte> (bon)
  - <texte> "Bonjour le monde !"</texte>
    <texte> "Bonjour toi !"</texte> (mauvais)
  - <document><texte> "Bonjour le monde !"</texte>
    <texte> "Bonjour le monde !"</texte></document>
    (bon)
  - <texte>"Texte"</texte> (mauvais)
  - <texte>"Texte"</texte> (bon)

Nom de balise en XML

- Nom de balise doit commencer par une lettre ou _ (underscore) et peut contenir des lettres, chiffres et les caractères "_-.". (pas d'espace, ni':')

Noms : exemples

- <copyright-information>
- <p>
- <base64>
- <décompte.client>
- <first_name>

Noms : règles avancées

- le caractère ":" est réservé pour les namespaces (voir chapitre suivant)
- un nom ne peut pas commencer par "xml"
- majuscules et minuscules sont différentes
  - <CLIENT>
  - <client>
  - <ClIeNt>

Attributs

- paire nom/valeur attachée à un élément
  - `<tel preferred="true">513-744-8889</tel>`
- noms suivent les mêmes règles
  - séparé de la valeur par un égal
- spécifié une seule fois dans la tag ouvrant ou vide
- la valeur est séparée par des apostrophes simples ou doubles
  - `<confidentiality level="I don't know"/>`
  - `<confidentiality level='approved "for your eyes only"'>`

Éléments vides

- sans contenu textuel, ils bénéficient d'une notation abrégée
  - `<email href="mailto:jdoe@emailaholic.com"/>`
  - `<email href="mailto:jdoe@emailaholic.com"></email>`
- le "/" s'insère dans la première balise

Indentation dans les documents

- l'indentation est du texte qui est généralement ignoré par l'applicatif
  -
```
<name>
    <fname>Jack</fname>
    <lname>Smith</lname>
</name>
```
  - `<name><fname>Jack</fname><lname>Smith</lname></name>`

Conventions

- tendance aux noms explicites
  - <address-book>
  - <adbk>
- séparer les mots par un "-"
  - <address-book>
- capitaliser la première lettre d'un mot
  - <AddressBook>
- d'autres conventions existent

Document valide

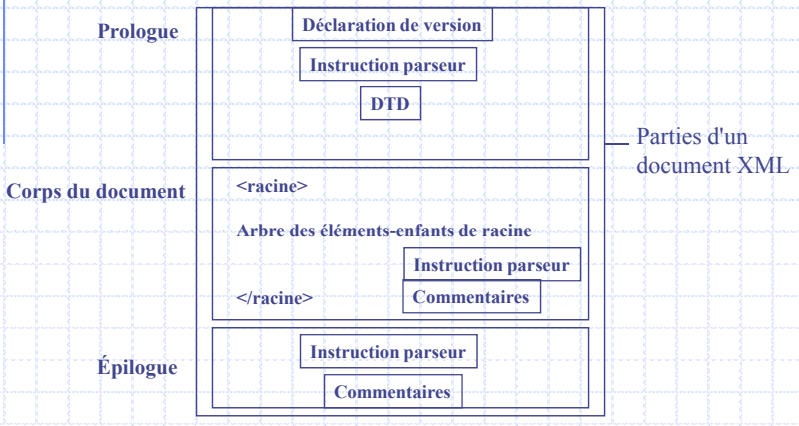
- XML est un langage stricte et simple ne nécessitant pas absolument de schéma.
- Le parseur peut déduire la structure du document à partir des balises. (différent en SGML)
- Un document valide est un document "bien formé" qui fait l'objet d'un schéma associé.
- Avantage d'avoir un schéma:
  - règles formalisées de conception de document
  - utiles pour travail en équipe
  - outil de vérification et validation du contenu
  - mise à disposition du schéma

Document Type Definition

- DTD est un héritage de SGML
- Un document SGML est toujours accompagné d 'un DTD, en XML, c 'est optionnel
- La syntaxe utilisé pour le DTD n 'est pas en XML mais un langage particulier.
- Langage contraignant, strict et parfois limité (EBNF)
- L'avenir pourrait voir les DTD remplacés par les schémas, nouveau standard en cours de normalisation par le consortium W3C.

Structure document XML et schéma

- Le schéma en DTD peut être dans la partie "prolog" d'un document XML (pas XML Schéma) ou dans un fichier externe



La section "prolog"

- La section prolog peut être vide ou contenir:
  - déclaration de version:
    <?xml version="1.0"?>
  - déclaration type de document (DTD)
    <!DOCTYPE ...>
  - commentaires
    <!-- ... -->
  - instruction de traitement (PI, processor instruction)
    <?xml-stylesheet type=« text/css » href=« mystyle.css »?>
- La section doit être spécifiée avant tout autre élément (avant élément racine) du document
- Le < de la déclaration de version doit être le premier caractère de la structure.

Déclaration de version

- La déclaration de version avant tout élément (pas d 'espace ni ligne vide) et la déclaration minimale doit contenir la version du langage utilisée :
  <?xml version="1.0"?>
- Cette déclaration peut contenir une indication sur le jeux de caractères utilisés :
  <?xml version="1.0 " encoding="iso-8859-1"?>
- Egalement une indication si le document contient des références vers des fichiers externes :
  <?xml version="1.0 " encoding="iso-8859-1"
  standalone="yes"?>
- L 'ordre des spécifications est à respecter.

Caractéristiques d 'un document XML

- Jeux de caractères est : UNICODE
- UNICODE : alphabet international, sous-ensemble de la norme ISO/IEC 10646
- UNICODE rassemble 34.168 symboles de 24 alphabets différents
- Un parseur doit au moins supporté UTF-8 et UTF-16 (sous-ensemble de Unicode, UTF-8 = ASCII)
- La plupart des parseurs supporte l 'encodage national tel que ISO-8859-1 (Latin 1 de Unicode)

Caractéristiques d 'un document XML

- Les caractères : « », « \t », « \r » et « \n » sont considérés comme des espaces.
- Ils sont, la plupart du temps, ignorés par l 'agent utilisateur permettant de visualiser le document.

Commentaires

- Les commentaires peuvent être repris à n'importe quel endroit du document.
- les commentaires sont indiqué à l'aide de "<!--" et "-->"
  - `<!-- exemple de commentaire -->`
- ils sont ignorés par les outils XML
- ils ne peuvent pas apparaître dans le balisage

Attributs contre éléments

- Le choix se pose toujours entre une structuration orientée élément ou orientée attribut.
- Avantages et caractéristiques des éléments:
  - permet une structuration riche : un élément peut en contenir d'autres, l'héritage, etc.
  - Permet d'imposer un ordre et le parseur retourne les éléments dans l'ordre
  - Un seul attribut d'un certain nom, autant d'éléments du même nom que nécessaire
  - Élément peut contenir du texte non-nommé mais pas d'attributs non-nommés

Attributs contre éléments

- Avantages et caractéristiques des attributs
  - plus de contrôle au niveau du DTD (avec XML Schéma, le contrôle est identique).
  - Occupe moins de place
  - beaucoup d'outils d'édition gère l'attribut via une liste déroulante
  - Valeur par défaut et valeur pré-définie possibles

Nom des éléments

- la tendance est à l'utilisation de noms lisibles
  - éventuellement associés à des codes (via des liens XLink) pour assurer l'aspect multilingues
- ce qui est important c'est de tirer avantage de la structuration de XML
  - à proscrire

```
<enregistrement code="nom"
 value="Tintin"/>
<enregistrement code="email"
 value="Tintin@moulinssart.be"/>
```

Gestion des types

- ce n'est pas possible avec le DTD
- une solution populaire avec DTD
  - `<!ELEMENT date (#PCDATA)>`
    `<!ATTLIST date type CDATA #FIXED "date">`
- XML Schema, nouvelle norme pour les schémas propose de nombreux types et le concepteur peut lui-même créer ses types.

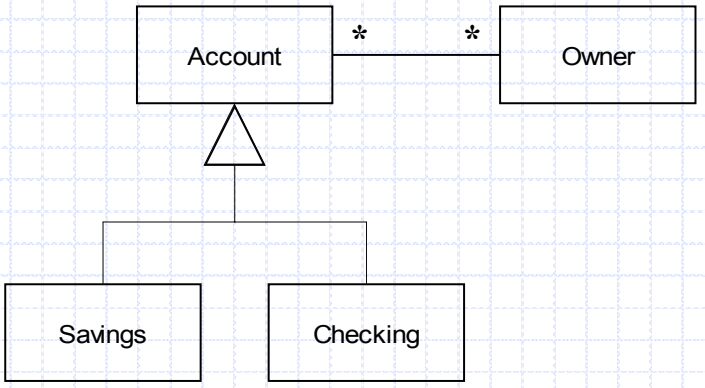
Conseils

- avant de démarrer la création des modèles
  - élaborer un guide reprenant toutes les possibilités
  - travailler sur base d'un modèle de haut niveau (UML, entité-relation)
  - la création du DTD peut être entièrement automatisée
    - de fait, il est possible de créer un DTD en partant d'un modèle en Rational Rose

Unified Modelling Language

- Langage normalisé pour la représentation de modèle OO
- Norme OMG
- Tendance marqué à l'élaboration d'un modèle avant le développement d'une solution
- Adaptation d'UML pour les technologies XML

Modèle UML

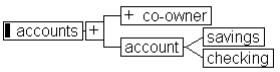


Premier modèle

- quelle racine
  - dépend de l'application
- un élément pour chaque objet du modèle

Correction

- premier modèle est incorrect
  - un compte peut avoir plus d'un propriétaire



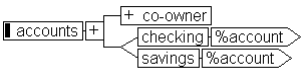
Document exemple

- inefficace
  - checking et savings sont inutiles

```
<?xml version="1.0"?>
<accounts>
  <co-owner>John Doe</co-owner>
  <co-owner>Jack Smith</co-
   owner>
  <account>
    <checking>170.00</checking>
  </account>
  <co-owner>John Doe</co-owner>
  <account>
    <savings>5000.00</savings>
  </account>
</accounts>
```

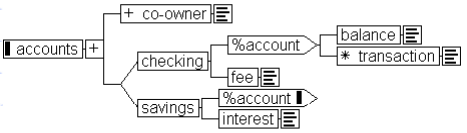

Version optimale

- une entité remplace certains niveaux d'héritage



Avec les attributs

- ajouter les attributs, sous forme de nouveaux éléments



Nouvel exemple

```
<?xml version="1.0"?>
<accounts>
  <co-owner>John Doe</co-owner>
  <co-owner>Jack Smith</co-owner>
  <checking>
    <balance>170.00</balance>
    <transaction>-100.00</transaction>
    <transaction>-500.00</transaction>
    <fee>4.00</fee>
  </checking>
  <co-owner>John Doe</co-owner>
  <savings>
    <balance>5000.00</balance>
    <interest>212.50</interest>
  </savings>
</accounts>
```