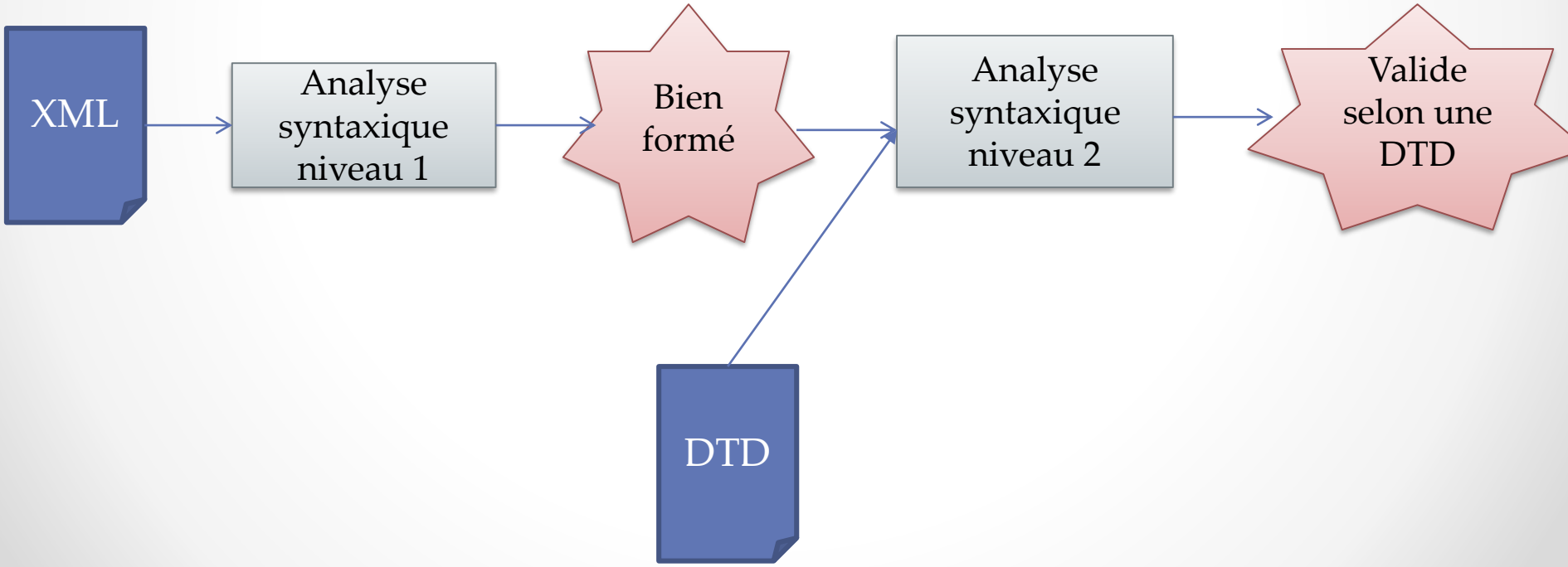
III. DTD (Definition Type Document)

Validation d'un document

- Un document valide est obligatoirement bien formé



Validation d' un document

- La validation est un moyen pour vérifier que votre document est conforme à une grammaire.
- Les éléments qui décrivent un document peuvent être définis dans une DTD (Definition Type Document)
- Une DTD permet de définir des classes de documents
- Un document bien formé conforme à une DTD est dit **valide** par rapport à cette DTD

DTD (Document Type Definition)

- DTD: forme de grammaire issue de SGML (Standard Generalized Markup Language)

- A partir du moment où une DTD est associée au document à valider, on peut valider à l'aide :

  - d'un logiciel spécialisé dans le traitement des documents XML (*XMLSpy*, *Editix*, *Eclipse*, Cooktop . . .)
  - par programme en utilisant les bibliothèques de traitement de XML disponibles dans beaucoup de langages (*java*, *php* , *perl*, ...)

Comment lier une DTD à un document XML?

Une DTD peut être associée de2 façons à un document XML :

1. **DTD interne :** toutes les règles sont dans le document XML.

```
<?xml version= "1.0"?>
<!– Définition de la DTD Interne-->
<!DOCTYPE bonjour
[<!ELEMENT bonjour (#PCDATA)>]>
<bonjour> bonjour tout le monde </bonjour>
```

2. **DTD externe :** toutes les règles à respecter sont décrites dans un fichier spécifique (.dtd) .

```
<?xml version= "1.0"?>
<!– Lien avec la DTD->
<!DOCTYPE bonjour SYSTEM "bonjour.dtd" >
<bonjour> bonjour tout le monde </bonjour>
```

```
<!ELEMENT bonjour (#PCDATA)>
```

bonjour.dtd

DTD Interne

- **Déclaration:** <!DOCTYPE racine [...texte de la DTD....]>
- Le document respecte la structure définie par la DTD
- Le nom de l'élément racine doit être le même que celui de la DTD (ici document)

```
<?xml version="1.0" encoding="UTF-8"?>

 <!DOCTYPE document [
   <! ELEMENT document (personne)>
   <!ELEMENT personne (nom, prenom)>
   <!ELEMENT nom (#PCDATA)>
   <!ELEMENT prenom (#PCDATA)>
 ]>

 <document>
     <personne>
         <nom>Einstein</nom> <prenom>Albert</prenom>
     </personne>
 </document>
```

DTD Externe

Déclaration:

DOCTYPE dans le document XML:

```
<?xml version= "1.0"?>
<!DOCTYPE livres SYSTEM “biblio.dtd “>
Ou
<!DOCTYPE livres SYSTEM
 “file://H://prive/biblio.dtd “>
Ou
<!DOCTYPE livres SYSTEM “http://... “>
```

biblio.xml

Définition de la DTD externe

```
<!ELEMENT livres    livre+   >
<!ELEMENT livre     (titre, auteur+, biblio?) >
<!ELEMENT titre     (#PCDATA) >
<!ELEMENT auteur (#PCDATA) >
<!ELEMENT biblio    (reference)+ >
<!ELEMENT reference    (#PCDATA) >
```

biblio.dtd

Options:

- **SYSYEM: DTD privée:** <!DOCTYPE nom SYSTEM "Fichier/URL">: signifie que la DTD sera généralement définie par un nom de fichier

- **PUBLIC**: **DTD externe publique (connue, e.g DTD HTML 4.0)**

<!DOCTYPE chapter PUBLIC "**-//OASIS//DTD DocBook XML//EN**" "../dtds/chapter.dtd">

- La déclaration d'un type de document (DTD) est composée d'une suite de déclarations :
  - déclarations d'éléments ;
  - déclarations des attributs d'un élément ;
  - déclarations d'entités.

Déclaration d'éléments

Syntaxe: <!ELEMENT *nom_élément modèle_de_contenu*>

- **Modèle de Contenu** peut être:

1)Vide (EMPTY) contenu vide même pas de blanc(que des attributs mais pas de texte)

```
<exemple/>
<exemple></exemple>
<exemple />
```

```
<!ELEMENT exemple EMPTY>
```

DTD

2) Textuel (#PCDATA) contient du texte (éventuellement vide) analysable dans l'encodage du document (Parsable Character Data)

```
<exemple> ceci est un exemple
</exemple>
```

Document XML

```
<!ELEMENT exemple (#PCDATA)>
```

DTD

Déclaration d'éléments

3) Composé: Séquence d'éléments= liste ordonnée d'éléments **(nom_1, nom_2)**

```
<personne>
   <nom>Martin</nom>
   <prenom>Jacques</prenom>
   <profession>sans emploi</profession>
</personne>
```

Document XML

```
<!ELEMENT personne (nom, prenom, profession)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT prenom(#PCDATA)>
<!ELEMENT profession(#PCDATA)>
```

DTD

4) Alternatif: Liste de choix **(nom_1| nom_2)**

```
<reponse> <oui/> </reponse>
```

ou

```
<reponse> <non></non> </reponse>
```

```
<!ELEMENT reponse (oui|non)>
<!ELEMENT oui EMPTY>
<!ELEMENT non EMPTY>
```

DTD

Déclaration d'éléments

5) Indicateurs d'occurrence

- Élément optionnel (0 ou 1 fois): **(nom) ?**
- Élément répétitif (0 à n fois): **(nom) ***
- Élément répétitif (1 à n fois): **(nom) +**

6) Combiné: L'utilisation des parenthèses permet de combiner les contenus composés et alternatifs :

```
<personnes>

    <personne>
    <nom/>
        <adr>7 avenue de Paris</adr>
    </personne>
    <personne>
        <nom/>
        <email>martin@...</email>
    </personne>
</personnes>
```

```
<!ELEMENT personnes (personne+)>
<!ELEMENT personne (nom,(adr|email))>
<!ELEMENT nom(#PCDATA)>
<!ELEMENT adr(#PCDATA)>
<!ELEMENT email(#PCDATA)>
```

DTD

Déclaration d'éléments

7) Contenu mixte a pour modèle de contenu mélangeant le texte et les éléments

(#PCDATA| $\mathbf{nom_1}$ |...| $\mathbf{nom_n}$)*

```
<p>Un <i>exemple</i>
    <b>important</b> de paragraphe <tt>HTML</tt>
</p>
```

```
<!ELEMENT p (#PCDATA| b | i |tt)* >

<!ELEMENT i (#PCDATA)>
<!ELEMENT b (#PCDATA)>
<!ELEMENT tt (#PCDATA)>
```

8) Libre (ANY): l'élément peut contenir tout type de données

```
<doc>Ceci est une <i>documentation</i>
        tres simple
</doc>
```

```
<!ELEMENT doc ANY>
<!ELEMENT i ANY>
```

Déclaration d'éléments

<!ELEMENT plan (introduction?, chapitre+, conclusion?)>

L'élément **plan** contient un élément **introduction** optionnel, suivi d'au moins un élément **chapitre** et se termine par un élément **conclusion** optionnel également.

<!ELEMENT chapitre (auteur*,paragraphe+)>

L'élément **chapitre** contient 0 à n éléments **auteur** suivi d'au moins un élément **paragraphe**.

<!ELEMENT livre (auteur, chapitre)+>

L'élément livre contient une séquence couples (auteur, chapitre)

Déclaration d'éléments

<!ELEMENT titre (#PCDATA|chiffre)*>

- L'élément **titre** contient un mélange de caractères textuels et d'éléments **chiffre**.
- C'est la seule façon de noter ce « mélange »
- Fichier XML autorisé:

<titre>
Il a parie <chiffre>7 </chiffre> contre <chiffre>1</chiffre> et il a gagné
</titre>

<!ELEMENT auteur (nom, prénom+, initiale?)>

- L'élément **auteur** contient un élément **nom**, un ou plusieurs **prénom**, zéro ou une **initiale**, et rien d'autre.
- Exemple de fichier XML autorisé:

<auteur>
<nom>Shaw</nom>
<prénom>George</prénom>
<prénom>Bernard</prénom>
</auteur>

Déclaration d'attributs

Syntaxe: <!ATTLIST ***nom_élément***
nom_attribut1 Type1 Déclaration1
nom_attribut2 Type 2 Déclaration2 ...>

- Déclaration:

'valeur '	La valeur par défaut de l'attribut (est une chaîne quelconque de caractères délimitée par des apostrophes 'valeur' ou des guillemets "valeur")
#REQUIRED	attribut obligatoire
#IMPLIED	attribut optionnel
#FIXED "valeur "	attribut toujours présent avec une valeur

Déclaration d'attributs

1) **Contenu textuel (CDATA** (Character Data)**):** contient des chaînes de caractères (ne contenant pas <,>,&, ' ou ").

Exemple:

```
<personne nom="martin" prenom="toto"> ... </personne>
```

```
<!ELEMENT personne ANY>
<!ATTLIST personne nom  CDATA #IMPLIED
                   prenom  CDATA #IMPLIED>
```

2) **Type énuméré**

Exemple:

```
<!ELEMENT prix (#PCDATA)>
<!ATTLIST prix monnaie (euros|francs) #REQUIRED>
```

```
<prix monnaie="euros"> 10 </prix>
```

Déclaration d'attributs

- <!ATTLIST chapitre
  Titre CDATA #REQUIRED
  Auteur CDATA #IMPLIED>

L'élément **chapitre** possède ici un attribut **titre** obligatoire et un attribut **auteur** optionnel.

- <!ATTLIST crayon
  couleur (rouge|vert|bleu)>

- L'élément crayon possède un attribut couleur dont les valeurs font partie de l'ensemble rouge, vert, bleu

Déclaration d'attributs

<!ATTLIST document version CDATA #FIXED "1.0">

```
<document version="1.0" >
...
</document>
```

valide

```
<document version="2.0" >
...
</document>
```

Non valide

<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED
nom-epouse CDATA #IMPLIED >

```
<nom titre="Mme" nom-epouse="Lenoir">
    Martin
</nom>
```

valide

```
<nom titre="M." nom-epouse="Lenoir">
    Martin
</nom>
```

valide

```
<nom titre="M." >
    Martin
</nom>
```

valide

```
<nom titre="Madame" nom-epouse="Lenoir">
    Martin
</nom>
```

Non valide

Exercice

- **Écrire une DTD sur les livres soit interne au livre1.xml soit externe sous le nom « livre1.dtd » et modifier livre1.xml**

En y ajoutant les informations suivantes sur un livre sont :

- des informations générales sur son édition comme par exemple le nom de l'éditeur, le lieu d'édition, son numéro ISBN et annee.
- Un champ optionnel sera réservé pour un avis personnel
- faites de l'élément annee un attribut de type énuméré, prenant comme valeurs possibles 2000, 2001, 2002, "avant_2000" et "inconnue" et proposant comme valeur par défaut inconnue.

Exercice

```
<!ELEMENT livre (auteur+, section, tome*, edition, avis?)>
<!ELEMENT auteur (#PCDATA)>
<!ELEMENT section (chapitre+)>
<!ELEMENT chapitre (paragraphe+)>
<!ELEMENT paragraphe (#PCDATA)>
<!ELEMENT edition (editeur, lieu_edition,isbn)>
<!ELEMENT editeur (#PCDATA)>
<!ELEMENT lieu_edition (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT avis (#PCDATA)>
<!ATTLIST livre titre CDATA #REQUIRED>
<!ATTLIST auteur nom  CDATA #REQUIRED
                 prenom  CDATA #REQUIRED>

<!ATTLIST section titre CDATA #REQUIRED>
<!ATTLIST chapitre titre CDATA #REQUIRED>
<!ATTLIST edition annee (2000 | 2001 | 2002 | avant_2000 | inconnue) "inconnue">
```

Identifiant (ID) et référence (IDREF)

- Un attribut **ID** sert à référencer un élément, la valeur de cette référence pouvant être rappelée dans des attributs **IDREF ou IDREFS**.

```
<texte>
    <p id="intro"> Ceci est une intro... </p>

        Ce detail est explique dans
    <section ref="intro">l'introduction</section>
</texte>
```

```
<!ELEMENT texte ANY>
<!ELEMENT p (#PCDATA)>
<!ATTLIST p id ID #IMPLIED>
<!ELEMENT section (#PCDATA)>
<!ATTLIST section ref IDREF #REQUIRED>
```

- Un élément ne peut avoir au plus qu'un attribut ID et la valeur associée doit être unique dans le document XML. Cette valeur doit être un nom XML (donc pas un nombre et doit commencer par une lettre).

- La valeur de déclaration pour un attribut ID est obligatoirement **#REQUIRED** ou **#IMPLIED**

Identifiant (ID) et référence (IDREF)

- **Exemple**

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<!DOCTYPE livre [
  <!ELEMENT livre    (section)*>
  <!ELEMENT section (#PCDATA | reference | references)*>
  <!ATTLIST section  num  ID #IMPLIED>
  <!ELEMENT reference     EMPTY>
  <!ATTLIST reference   ref  IDREF  #REQUIRED>
  <!ELEMENT references  EMPTY>
  <!ATTLIST references refs IDREFS #REQUIRED>
]>
```

Chaque section a un attribut num de type ID

reference et references ont respectivement des attributs ref de type IDREF et refs de type IDREFS

```
<livre>
 <section num="sec0">Une référence <reference ref="sec1"/></section>
 <section num="sec1">Des références <references refs="sec0 sec2"/></section>
 <section  num="sec2">Section sans référence</section>
 <section  num="sec3">Une auto-référence <references refs="sec3"/></section>
</livre>
```

→ Référencer, dans le contenu d'une section, une (par reference) ou plusieurs (par references) autres sections.

- Proposer une DTD en utilisant les attributs ID et IDREFS

```
<famille>
   <personne num="Jane"
             mere="Mary" pere="John">
      <nom>Jane Doe</nom>
   </personne>
   <personne num="John"
             enfants="Jane Jack">
      <nom>John Doe</nom>
   </personne>
   <personne num="Mary"
             enfants="Jane Jack">
      <nom>Mary Doe</nom>
   </personne>
   <personne num="Jack"
             mere="Mary" pere="John">
      <nom>Jack Doe</nom>
   </personne>
</famille>
```

```
<!ELEMENT famille
(personne*)>
<!ELEMENT personne (nom)>
<!ELEMENT nom (#PCDATA)>
<!ATTLIST personne
   num ID #REQUIRED
   mere IDREF #IMPLIED
   pere IDREF #IMPLIED
   enfants IDREFS #IMPLIED >
```

Identificateurs NMTOKEN et NMTOKENS

Contient un ou plusieurs identificateurs XML séparés par des blancs (ne peuvent contenir que des lettres, chiffres, points, tirets, underscore et deux-points)

Exemple: Attribut dates d'un élément concerts, de la forme mm-jj-aa

```
<concerts dates="08-21-2014 08-23-2014 08-27-2014">
    ColdPlay
</concerts>
```

```
<!ELEMENT concerts ANY>
<!ATTLIST concerts dates NMTOKENS #REQUIRED>
```

La différence entre CDATA et NMTOKEN est que ce dernier n'accepte pas les espaces et les caractères de ponctuation autres que le point, le tiret, le underscore et les deux-points.

Identificateurs NMTOKEN et NMTOKENS

```
<!ELEMENT attributes (#PCDATA)>
<!ATTLIST attributes
        aaa CDATA #IMPLIED
        bbb NMTOKEN #REQUIRED
        ccc NMTOKENS #REQUIRED>
```

```
<attributes aaa="#d1" bbb="a1:12" ccc="3.4 div  -4"> </attributes>
```

valide

```
<attributes aaa="#d1" bbb="#d1" ccc="#d1"> </attributes>
```

Non valide car # est interdit dans les attributs de type NMTOKEN et NMTOKENS

```
<attributes bbb="A B C " ccc="A     B     C">
</attributes>
```

Non valide car l'espace est interdit dans le type NMTOKEN

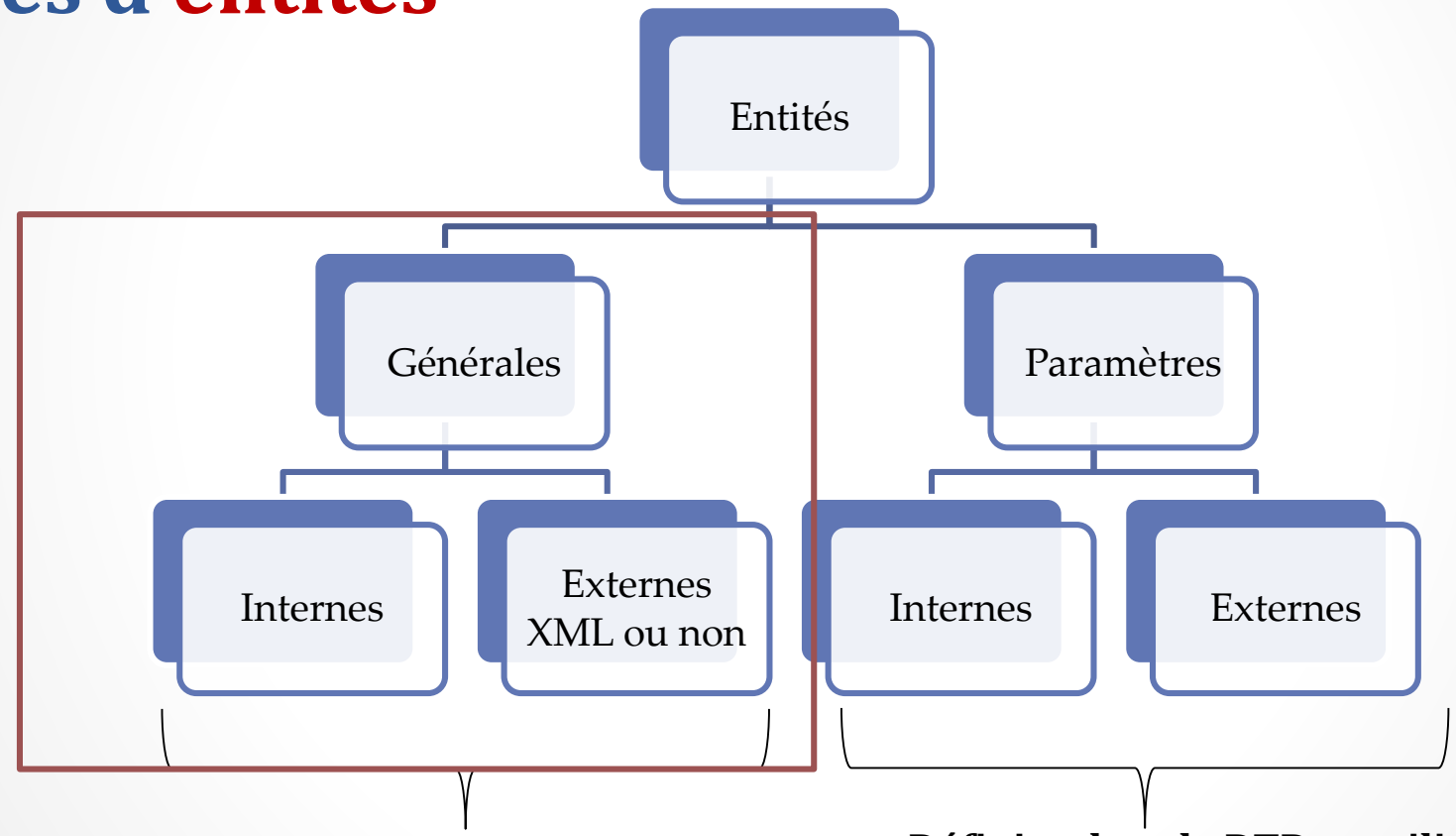
```
<attributes bbb="a1:12 " ccc="3.4  div  -4">
</attributes>
```

valide

Définition d'une entité

- Un nom pour un morceau de XML
- Sorte de *d'abréviations* (ou de macro) qui associe
  - un nom d'entité
  - à un contenu d'entité qui est un simple texte ou un fragment de document XML
- Déclaration: **ENTITY**

Types d'entités



Définies dans la DTD et utilisées dans les documents correspondants

Définies dans la DTD et utilisées dans la DTD elle-même (toujours analysées)

Entité Générale Interne

- Définie dans une DTD et représente du texte
- Sont des macros qui permettent de factoriser et de paramétrer les documents XML
- Des symboles définis dans une DTD et **utilisés dans un document XML** comme raccourcis d'écriture

- **Déclaration dans la DTD:** **<!ENTITY** *nom_entité* "***valeur***">
- **Référence dans le XML:** **&nom_entité;**

```
<!ENTITY  ADN "Acide désoxyribonucléique">
```

DTD

```
<génétique> L' &ADN;  est une molécule complexe.
</génétique>
```

Fragment du document XML valide

Entité Générale Externe XML

- Permettent d'inclure des documents XML identifiés par une URL.
- **Déclaration dans la DTD:** **<!ENTITY** *nom_entité* **SYSTEM** *nomfichier***>**
- **Référence dans le XML:** &nom_entité;

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE livre SYSTEM "livre.dtd"
  [
  <!ENTITY  chap1  SYSTEM  "chapitre1.xml ">
  <!ENTITY  chap2  SYSTEM  "chapitre2.xml ">
  ]
  >
<livre> &chap1; &chap2; </livre>
```

- Contraintes sur le document XML inclut :
  - il peut contenir un prologue (encodage),
  - pas de référence à une DTD,
  - il doit être bien formé

Entité Générale Externe non XML

- Une *entité externe non XML* peut contenir n'importe quoi (image, son, données, etc.).
- **Déclaration:** **<!ENTITY** *nom_entité* **SYSTEM** *uri* **NDATA** *format***>**
  - NDATA (Notation DATA) précise le type d'entité non analysée que le processeur XML doit traiter.
- Nous devons donc, au préalable, définir un type :

  <!NOTATION nom_du_type **SYSTEM** "url_associé_au_type">

Exemple :

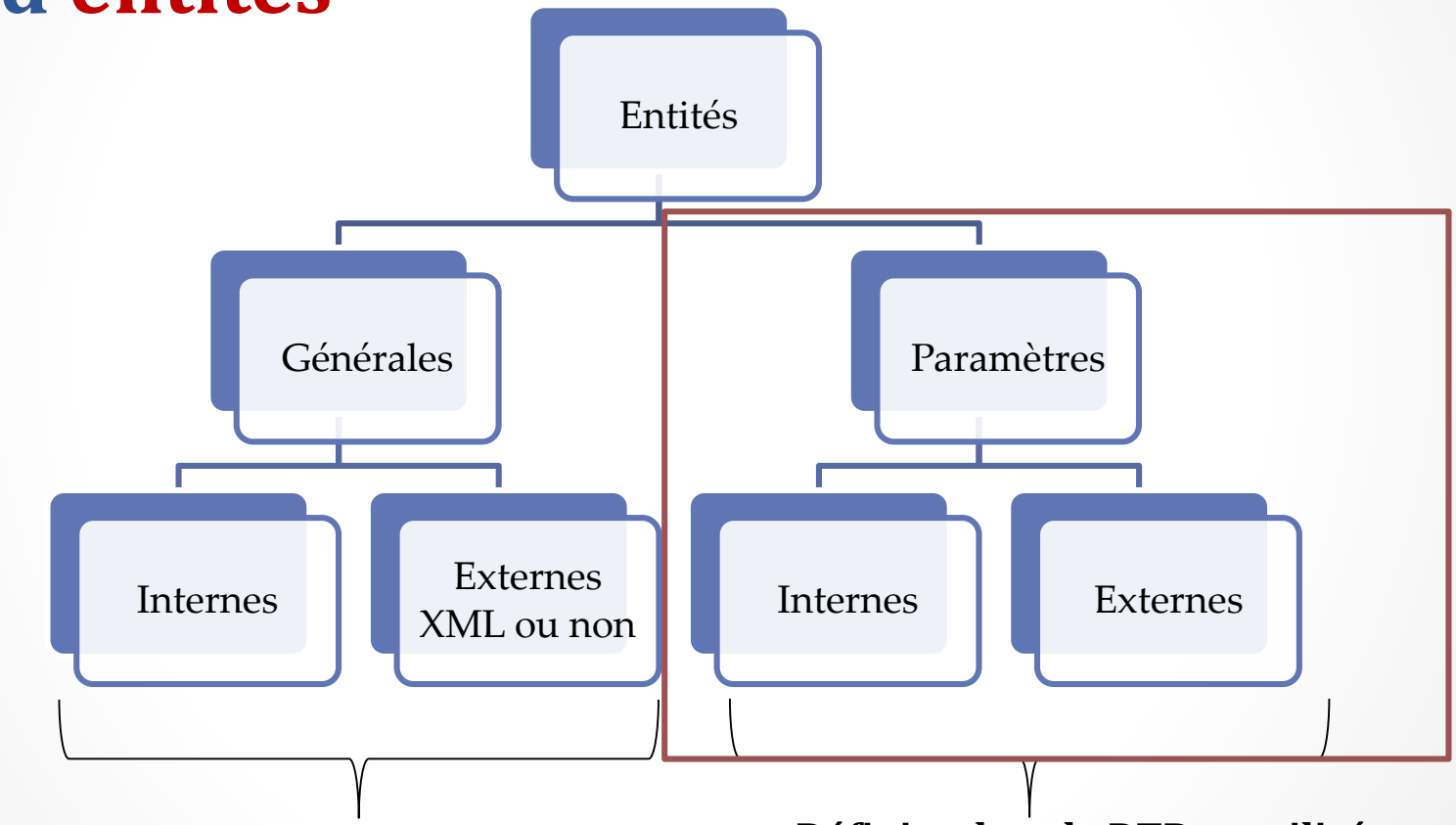
La référence &maison; est impossible (ce n'est pas du XML). Son utilisation doit passer par un attribut typé entité externe non XML (**type ENTITY)**

DTD: **<!NOTATION** jpeg **SYSTEM "JPG">**
<!ENTITY maison **SYSTEM** "maison.jpg" **NDATA** jpeg**>**
<!ELEMENT image EMPTY>
<!ATTLIST image src ENTITY #REQUIRED>

XML: <image src= " maison" **/>**
<!--pas de &..; -->

Ne pas oublier d'ajouter la directive standalone="no"

Types d'entités



Définies dans la DTD et utilisées dans les documents correspondants

Définies dans la DTD et utilisées dans la DTD elle-même (toujours analysées)

Entité paramètre interne

- Sont des macros qui permettent de factoriser et de paramétrer les **DTD.**
- **Déclaration:**

<!ENTITY % *nom_entite* "*valeur_entite*" >

(espace entre % et nom_entité)

- **Référence: %nom_entite;**

```
<!ENTITY % statut " statut (public|prive) #IMPLIED" >
<!ELEMENT article (#PCDATA) >
<!ATTLIST article date CDATA #IMPLIED
                    %statut; >
```

Entité paramètre externe

Permettent d'inclure des DTD externes

<!ENTITY % chapitre SYSTEM "chapitre.dtd" >

%chapitre;

DTD interne/externe

- La partie interne est traitée avant la partie externe.
- Lorsque la DTD est mixte, tout élément doit être déclaré dans la partie interne ou dans la partie externe mais pas dans les deux.
- Il est possible de déclarer plusieurs fois le même attribut. La première déclaration a la priorité. Il est ainsi possible de donner une nouvelle déclaration d'un attribut dans la partie interne puisque celle-ci est placée avant la partie externe.

Quelques règles

- Une DTD doit être facile à comprendre et à étendre
- Utilisation des commentaires et des espaces pour rendre le texte plus lisible
- Les déclaration ATTLIST sont conservées à proximité de la déclaration ELEMENT correspondante
- Les entités paramètre rendent les déclarations plus concises et permettent de modifier le comportement de plusieurs déclarations en une seule modification.

Limites des DTD

- Pas au format XML. Cela signifie qu'il est nécessaire d'utiliser un outil spécial pour manipuler un fichier, différent de celui utilisé pour l'édition du fichier XML.

- Le typage des données est extrêmement limité (que du #PCDATA):

- les DTD ne supportent pas les "espaces de nom"
  - En pratique, cela implique qu'il n'est pas possible, dans un fichier XML défini par une DTD, d'importer des définitions de balises définies par ailleurs

Liens et supports utiles

- XML recommandation W3C: http://www.w3.org/TR/REC-xml/
- Support de Cours en ligne Elisabeth Murisasco
- Jacques Le Maitre, Description et manipulation de documents XML, supports de cours en ligne http://lemaitre.univ-tln.fr/cours.htm
- Georges Gardarin, XML : Des bases de données aux services Web, Dunod, 2003 et supports de cours XML http://georges.gardarin.free.fr/
- François Role « Modélisation et manipulation de documents XML » Lavoisier
- Erik T.Ray « Introduction à XML » O'REILLY