XMLチュートリアル

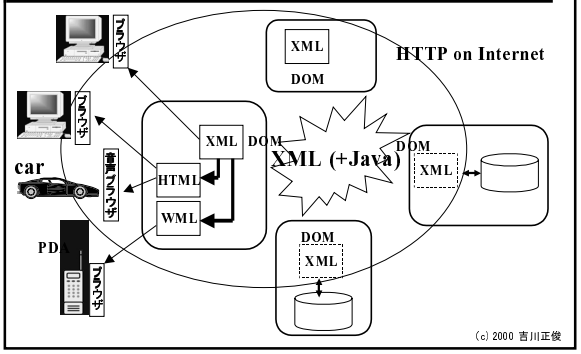
奈良先端科学技術大学院大学
吉川 正俊
yosikawa@is.aist-nara.ac.jp
http://db-www.aist-nara.ac.jp/members/Yoshikawa/home.html

Why XML ?

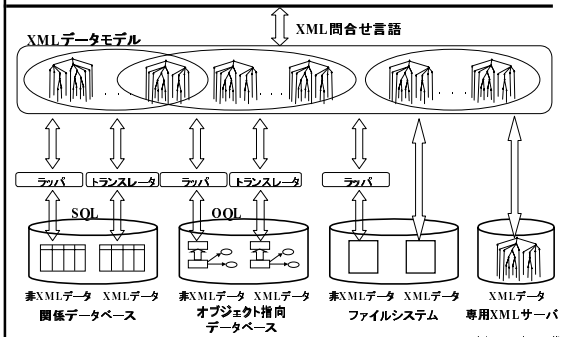
Why XML ?

- 応用プログラムやO.S.に依存しない標準のデータ交換フォーマットが欲しい
  - 特に, Internet や WWW 上での
- 重要なこと
  - 汎用性
  - 単純性
  - 既存技術との互換性

XMLの応用モデル



インターネット上の共通言語としてのXML



XMLの歴史と標準化の動向

歴史（菊田昌弘 bit, 1999年1月号）

- 1960年代に開始
  - GCA（Graphic Communications Association)では，異なるタイプセット間でのドキュメント交換を可能とするGenCode と呼ばれる汎用的なフォーマットコードを開発
  - IBMでは，GML (Generalized Markup Language)を，社内文書の管理に用いていた.
- 1980年代
  - GenCodeグループとGMLグループの代表者が ANSIにおいて，Computer Language for the Processing of Text の検討組織を開始し，ドキュメント中のマークアップを標準化するための作業を始めた.

歴史

- 1986 SGMLがISO標準になる
- 1992 最初のHTMLの仕様が出版される
- 1994秋 W3Cができる

WWWの爆発的な普及

- 1998 XML 1.0がW3C Recommendationになる
- 1999.9 PhaseIII: Query Language WG, XML Digital Signature

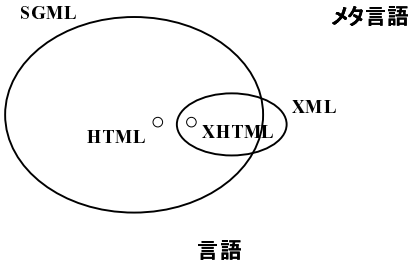
HTMLの歴史

- 当初のHTML
  - 単にSGML風のマークアップ言語でしかなかった
- HTML2.0
  - IETFによってまとめられ，正式なSGMLアプリケーションとなる
- HTMLの標準化の作業の舞台はW3Cへと移る
- HTML3.2を経て，現在はHTML4.01(1999.12)が最新版
- 現在ISOでも，HTML4.0をもとに，より厳格なサブセットという形で，ISO/IEC 15445（通称ISO-HTML)の標準化を進めている.
- XHTMLへ

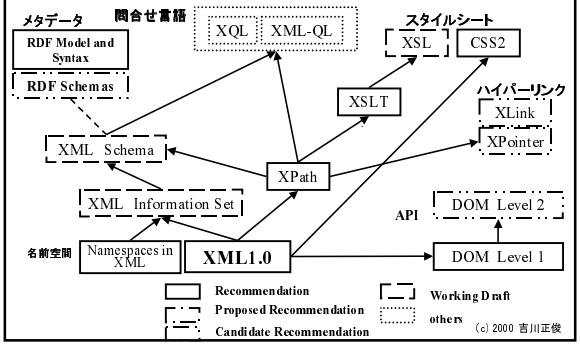
HTML と XML

- HTML
  - 言語
  - 内容の記述と表現法の記述が一緒になっている.
    - » 従って，異なる表現法への変換が非効率的
  - 自分でタグを拡張できない
- XML
  - メタ言語
  - 内容のみを記述
    - » 表現法は，たとえば CSSにまかせる
  - 自分でタグを拡張できる

SGML, XML, HTML and XHTML



種々の標準，提案の関連



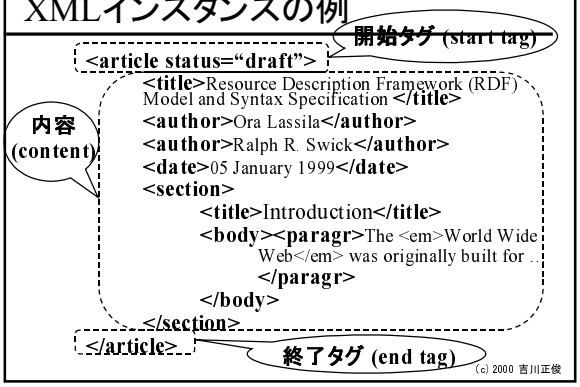
XMLの概要

XML (eXtensible Markup Language)

- データおよび構造化文書の表現方法
  - インターネット上での交換性を考慮
  - 汎用性
    - » すべてクリアテキストで表現
      ⟹ 計算機，O.S.，応用からの独立性
    - » 国際性 (ISO/IEC 10646)
  - 基本的に木構造を表現
- W3C (World Wide Web Consortium)により標準化

整形式のXML文書

XMLインスタンスの例

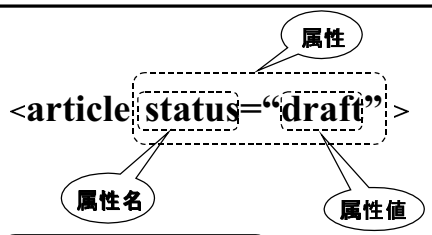


要素の階層構造

- ある要素は，その上位要素に完全に含まれていなければならない
  - <vp>恐れ<placename>入</vp>谷</placename>の鬼子母神のような文書(GDA日本語タギングマニュアル 橋田 浩一)や
  - 大腸菌ゲノムのような環状データ

  はそのままでは表現できない

XML属性



- 開始タグ中に記述
- “ または ‘ で囲む

実体参照 (entity reference)

- 要素の内容中に文字“<”を含めたいときはどうするか？

× 不等号<は，数学で使われる．

○ 不等号<は，数学で使われる．

実体参照の方法

&実体名;

実体参照

- いつでも(DTDなしで)使える実体

文字	実体名
<	lt
>	gt
&	amp
’	apos
”	quot

文字参照 (character reference)

- キーボードから直接入力できない文字はどう入力するか？

☆ ISO/IEC 10646文字コードを指定して文字を参照する

文字参照の方法

&#10進コード; **または** &#x16進コード;

整形式 (well-formed)のXML文書

- XMLの構文規則に従っている文書
  - 開始タグと終了タグの対応が正しく取れている
  - タグが正しく入れ子関係になっている
  - ルート要素（文書要素(document element)とも呼ぶ）は，唯一
  - 属性は開始タグに記述されている
  - …
- 先頭行にXML宣言を書く．
  - 例: **<?xml version=“1.0” encoding=“iso-2022-jp”?>**
  - **<?xml version=“1.0” encoding=“shift_jis”?>**

XMLインスタンスのデータ構造

タグの入れ子構造 ＋ 属性

⇩ **ゆえに**

基本的に木構造

妥当なXML文書

XML文書の構成

- [XML宣言]
  - バージョンと符号化宣言を指定
  - 例: <?xml version="1.0" encoding="iso-2022-jp"?>
- [DTD (Document Type Definition)]
  - 要素, 属性, エンティティの宣言
- XMLインスタンス*
  - 実際のタグ付き文書

* XMLインスタンスという用語は仕様書にはない。

DTD

- **要素型宣言** (element type declaration)
- **属性リスト宣言** (attribute-list declaration)
- **エンティティ宣言** (entity declaration)
- **記法宣言** (notation declaration)

DTDの例

```
<!ELEMENT article  (title, author+, date, section+)>
<!ATTLIST article   status (final|draft) "draft">
<!ELEMENT title    (#PCDATA)>
<!ELEMENT author   (#PCDATA)>
<!ELEMENT date     (#PCDATA)>
<!ELEMENT section  (title, body+)>
<!ELEMENT body     (figure | paragr)>
<!ELEMENT figure    EMPTY>
<!ATTLIST figure    file ENTITY #IMPLIED>
<!ELEMENT paragr   (#PCDATA|em)*>
```


要素型宣言

- **XMLインスタンスで使用できる要素の名前，階層構造を規定**
- **<!ELEMENT 要素名 内容モデル>**
  これにより要素の階層構造中の一つの親子関係を指定

内容モデル(要素内容)

- 要素内容
  - 子要素だけを含み文字データを含まない
  - **出現順序の指定：** 列(,) 選択(|)
  - **出現回数の指定：** 1回以上任意の回数(+) 0回以上任意の回数(*) 0回もしくは1回(?)

例)

```
<!ELEMENT article  (title, author+, date, section+)>
<!ELEMENT section  (title, body+)>
<!ELEMENT body     (figure | paragr)>
```


内容モデル(要素内容) --- 構文規則

```
children ::=
    (choice | seq) ('?' | '*' | '+')?
cp       ::=
    (Name | choice | seq) ('?' | '*' | '+')?

choice ::= '(' S? cp ( S? '|' S? cp )* S? ')'
seq    ::= '(' S? cp ( S? ',' S? cp )* S? ')'
```

(S は 「空白」)

内容モデル(混在内容)

- 混在内容
  - 子要素に混在して文字データが含まれる可能性がある

例) <!ELEMENT paragr (#PCDATA|em|footnote)*>
<!ELEMENT paragr (#PCDATA|em)* >
<!ELEMENT title (#PCDATA)>
#PCDATAは文字データを表す

内容モデル(空要素, 任意要素)

- 空要素 (EMPTY)

DTD:
<!ELEMENT figure EMPTY>
文書インスタンス:
<figure></figure>
または
<figure/>

- 任意要素 (ANY)

属性リスト宣言

<!ATTLIST article status (final|draft) "draft">
<!ATTLIST figure file ENTITY #IMPLIED>

要素型名

属性名

属性の型
- CDATA
- ID
- IDREF
- IDREFS
- ENTITY
- ENTITIES
- NMTOKEN
- NMTOKENS
- 列挙型

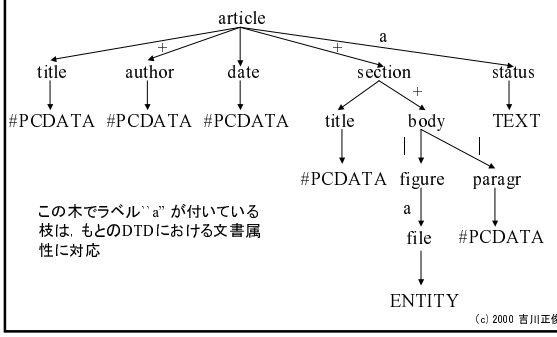
デフォルト
- #REQUIRED ... 必須
- #IMPLIED ... 任意
- 値 ... デフォルト値
- #FIXED 値
  ... デフォルト値(固定)

ID, IDREFの例

DTD:
<!ELEMENT emp (#PCDATA)>
<!ATTLIST emp id ID #required>
<!ATTLIST emp boss IDREF #implied>

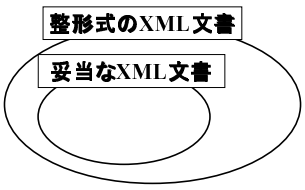
XMLインスタンス:
… <emp id="19" boss="3">上原君</emp>は,
<emp id="3">長嶋さん</emp>と同じ会社で働いている. ...

DTDの木表現



妥当 (valid) なXML文書

- 整形式 であり、その上さらにDTDを持ち、DTD中の宣言に従っているXML文書



XML文書 --- 二つの側面

- 単なる文字列である
- 構造化データであり，しかもその構造は基本的に木でモデル化できる
  - より厳密には，ID/IDREFによる相互参照なども考慮する必要があるため，木ではなく有向グラフでモデル化する必要がある

DTDの指定方法

- DTDが別ファイルにある場合
  <!DOCTYPE　ルート要素の名前
  SYSTEM　ファイル名>
- DTDを直接書く場合
  <!DOCTYPE　ルート要素の名前
  [DTDの記述]>

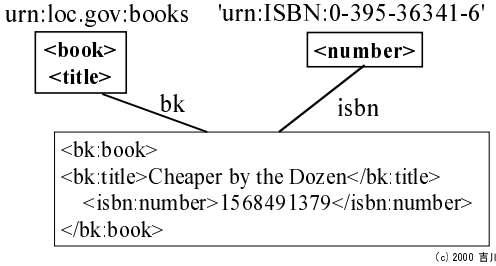
persons.dtd

```
<?xml version="1.0" encoding="iso-2022-jp"?>

<!ELEMENT persons       (person)+>

<!ELEMENT person        (name, jobtitle?, affildept?, affilorg?,
email, homepage?)>

<!ELEMENT name          (firstname, middlename?, familyname)>
<!ELEMENT firstname     (#PCDATA)>
<!ELEMENT middlename    (#PCDATA)>
<!ELEMENT familyname    (#PCDATA)>

<!ELEMENT jobtitle      (#PCDATA)>
<!ELEMENT affildept     (#PCDATA)>
<!ELEMENT affilorg      (#PCDATA)>
<!ELEMENT email         (#PCDATA)>
<!ELEMENT homepage      (#PCDATA)>
```


名前空間

名前空間

- **ネットワークを介してスキーマ部品（たとえば 要素型や属性）を再利用するための機構**



名前空間宣言の例

```
<x xmlns:edi='http://ecommerce.org/schema'>
      <!--the "edi" prefix is bound to
      http://ecommerce.org/schema for the "x"
      element and contents -->
</x>
```

- ediというprefixと，上記名前空間名との対応は，<x …> と </x> の間でのみ有効
- **この文書のDTDで x の属性として xmlnsを宣言する必要はない.**

なぜ“URI:”のように直接URIを参照しないか？

- URI参照はXMLの名前として許されない文字列を含んでも良いため，名前空間のprefixとして直接使うことはできない．

⇩

- 従って，名前空間prefixを導入して，それによってURIを参照するようにしている．

名前空間の利用例

```
<x xmlns:edi='http://ecommerce.org/schema'>

      <!-- 要素型の例 -->
      <edi:price units='Euro'>32.18</edi:price>

      <!-- 属性の例 -->
      <lineItem edi:taxClass="exempt">Baby
      food</lineItem>
</x>
```


複数の名前空間の利用

```
<?xml version="1.0"?>
<!-- both namespace prefixes are available
throughout -->
<bk:book xmlns:bk='urn:loc.gov:books'
         xmlns:isbn='urn:ISBN:0-395-36341-6'>
   <bk:title>Cheaper by the Dozen</bk:title>
   <isbn:number>1568491379</isbn:number>
</bk:book>
```


名前空間の有効範囲の例

```
<?xml version="1.0"?>
<book xmlns='urn:loc.gov:books'
        xmlns:isbn='urn:ISBN:0-395-36341-6'>
  <title>Cheaper by the Dozen</title>
  <isbn:number>1568491379</isbn:number>
 <notes>
      <p xmlns='urn:w3-org-ns:HTML'>
           This is a <i>funny</i> book!
      </p>
  </notes>
</book>
```