XML 概論

Christian Wittern
クリスティアン・ウィテルン

1　マークアップ入門

1.1　テキストの概念

「テキスト」とは用途や文脈によって、以下のように分類できる：

* 日常では、物事を表現するために書かれた幅広い言葉である。
* 言語学では、伝達行為であり、原文の本質を充たすものである。
* 文学理論では、テキストは研究対象であり、小説、詩、映画、広告など、言語構成を伴うものである。この広い使用は 1980 年代の記号学や文化研究によって活気づけられた。
* 情報処理では、テキストは文字データのことである。

テキストは、非常に簡単、または非常に複雑に構造化することができる。通常、構造は理解しやすくできるものである。テキストは何らかの表記法による、文字で作られたスクリプト（文字集合）である。

1.2　テキストエンコーディング

テキストエンコーディング（テキストの符号化）とはデジタル方式でテキストを書き直すことである。テキストエンコーディングは、たまにに文字エンコーディング（文字の符号化）と混同することがある。文字エンコーディングとは電子化したい文字列をなんらかの符号情報に置き換えることである。

テキストエンコーディングは文字エンコーディングを含んでいるが、電子形態でテキストの構造を作り直すことに関係があるために、それ以上の問題も含んでいる。

また、テキストエンコーディングはマークアップと混同することもある。マークアップとはテキストの構造や状況、または他の特徴についての情報を表現することで、テキストエンコーディングで使われる方法論である。

1.3　マークアップ

マークアップという用語は編集者が伝統的な出版において、サイズ、文字の太さページの位置やマージンなど文章がどう印刷されるべきであるかの印刷情報を伝えるために使っていた注釈や記号に由来する。

マークアップの概念は、例えば、ページレイアウト、フォント、句読点といった、文章によるコミュニケーションのある一面を含ませることである。これらのことは何人かのマークアップの理論家によって汎用的なものとなった。

通常、電子テキストのマークアップは表象的、手続き的、記述的といった使用方法によって異なった分類がされる。

1.3.1　表象的マークアップ

表象的マークアップとは原文の体裁そのものを電子テキストで空白や改行などを使ってページ上に配置することである。このマークアップは追加処理なしで出力装置（プリンタかディスプレイ）に送信することができる。

1.3.2 手続き的マークアップ

手続き的マークアップとは出力において望まれる効果を生成するために電子テキストに特別なマークアップ記号を挿入することである。「フォーマッタ」と呼ばれる特別なプログラムはこれらのマークアップ記号を解釈し、中間状態を生成し、出力装置に送信するものである。

例えば、文字をイタリック体に変えるための開始マークアップ記号として `.i` や `<i>`、そしてイタリック体を解除するためのマークアップ記号として `.i` や `</i>` といった特別なコードがある。フォーマッタは、これらのコードを解釈し、望む結果が実現するように必要な処理を行なう。

1.3.3 記述的マークアップ

記述的マークアップとは、直接必要な文章を整形するマークを挿入する代わりに、文章の特徴を記述または識別するための情報をテキストに挿入することである。ここでは、付加的な抽象化層と、これらの特徴をどのようにして描画するのかという情報を保持する方法についてそれぞれ導入する。フォーマッタは、望む結果が得られるように、記述的マークアップと書式情報両方を使う。

直接、望む結果を指定する代わりに、先に述べたように、例えば、文章で記載された書名があるのを見て、その書名に対し `<title>` とし、それをマークする。そして、印刷する時に書名タイトルをイタリック体で表現して欲しいと指定したとする。しかしながら、ディスプレイ上では読み易さを重視してイタリック体であるよりむしろ色でそれを表現するほうが好ましいかもしれない。

記述的マークアップは異なる装置（プリンターやディスプレイなど）おいて、柔軟性を提供するだけでなく、このマークアップにおいて含まれる情報はさらなる目的にも利用可能である。例えば、記載された書名の一覧や、見出しの索引などを作ったりなどいろいろ利用できる。

記述的マークアップは著者による組版またはテキストの複製や出版などに対して多くの利益を提供する。

* 文章の構成は簡易
* 構造指向の編集に対応
* 一般的なテキストエディタに対応
* テキストの複数の見方が可能
* 一括した形式の指定や変更が可能
* 索引や付録などがを自動的に作成可能
* 多くの出力装置に対応
* 移植性、永続性を最大限に発揮
* 情報検索に対応
* 分析操作に対応

記述的マークアップは最も多目的で、柔軟性のあるマークアップ法である。そして、この方法がこの講義で使うマークアップ形式である。

1.4 順序を持つコンテンツオブジェクトの階層としてのテキスト

記述的マークアップとそれを伴う方法論はあまりにも成功していたため、ある研究者はこの方法はテキスト処理についての便利な手法だけではなく、やはり根本的に≪正しい≫方法であると信じていた：“記述的なマークアップは一番合理的な方法だけではなく、想像できる以上に一番優れた方法である” (Coombs et. al, 1987)。この見解は記述的なマークアップの基にあるモデルだけが正しく≪テキストの正しい理解≫ (DeRose et. al., 1990) を反映させる。このモデルではテキストは、章、節、段落、文などといった階層を持つ論理的な構造によって決定し

ている。ページ、列、行、フォントの変更、空白などといった物理的な構造のことではない。この見解では、テキストは≪順序を持つコンテンツオブジェクトの階層≫ (OHCO) である。記述的マークアップは、この階層を明白に記述することで功を奏する。

OHCO の見解はテキストエンコーディングの強力なモデルとなり、実在するテキストの多くの特徴の合理的な処理につながる（例えば、章の見出しは必ず章の始めにあり、途中にはない。詩の句は詩の中にあるから句となる、等)。しかし、適用できない場合もある。例えば、詩の一文は必ず一句の中に入らない。句も文に入るとは限らない。つまり、文と句は違う階層に属する。同様に、小説などでは語り手が直接話法で割り込む場合もある。それにもかかわらず、OHCO はテキスト処理に於いて他のモデルより優れた処理を可能にするので（例えば、テキストは単なる文字列であるなどの見方に比べると）SGML、XML 等のマークアップ言語の支配的な考えになり、テキストエンコーディングに広く適用されている。

2 XML 入門

2.1 歴史

1960 年代から、記述的マークアップ方法の標準化を目指した。多くのプロジェクトがあったが、中でも IBM の Charles Goldfarb 率いるメンバーが開発した GML (Generalized Markup Language) が最も成功し、さらに、ANSI (American National Standards Institute) とともに SGML (Standard Generalized Markup Language) を開発し、1986 年に ISO (International Standards Organizations) は SGML を標準 (ISO 8879: Information Processing - Text and Office Systems - Standard Generalized Markup Language) として認可した。SGML はマークアップ言語というよりメタ言語（言語を記述するための言語、ここではマークアップ言語を定義する言語）の側面のほうがむしろ強く、テキストに付加された情報と本文とを区別するのに使用される文字列のようなものをいう。SGML は特定用途向けマークアップ言語を定義するのに使用でき、これらを SGML のアプリケーション（応用）と呼ばれる。そのようなアプリケーションは、文書のどの場所にどんなタグが使われるかといった仕様書を作成することによって構成される。

DTD (Document Type Definition) と呼ばれる仕様書は書いた文書が規格に沿って書かれているかをチェックするのに利用でき、もし仕様に沿ってなければすぐに判るようになっている。こういう目的に使われるソフトウェアをパーサ（構文解析ツール）と呼び、文書の検証過程に使われる。

2.2 HTML

SGML の一つの応用に HTML (Hyper Text Markup Language) がある。これは WWW (World Wide Web) の基本的なマークアップ言語である。HTML はウェブページを作成する際に良く使われる要素（タグ）のセットを定義している。例えば、段落なら `<p>`、レベルの異なる見出しに対しては `<h1>`、`<h2>` などがあり、ページ内に画像を埋め込むには `<img>`、他のページへのリンクを張るには `<a>` といったタグが用意されている。非常に限られたセットではあるが、これに対応するソフトウェアを書くことは簡単であり（この手のソフトウェアをブラウザまたはウェブブラウザと呼ばれている)、WWW の大規模な成長と継続的なサポートを見れば、十分に成功しているといえる。

しかしながら、HTML で有効なタグセットは少なかった、そしてすぐに限界が見えてしまい、各ブラウザメーカーは競って独自のタグを考案しようとした、当然独自のタグは他のブラウザには対応していないだろう。他には SGML の仕様が複雑であり、検証ソフトを作ることが困難という問題もすぐに生じた。そのようなことから、多くのユーザーは使用したブラウザが文書を表示することができるなら、画面を見ることで HTML 文書の検証の代わりにしていた。一方で、ブラウザメーカはユーザの底辺を拡大するために、多少文法が間違っていてもできるだけ推測して画面に表示するように努力した。

2.3 SGML から XML へ

この状況を解決して、最適な道筋を経て WWW に提供するために、W3C (World Wide Web Consortium) は使い易く、実装し易い SGML の簡易型のバージョンを開発する任務を主だった専門家のワークグループに課した。W3C とは WWW で利用される技術の標準化をすすめる団体のことで、WWW 技術に関わりの深い企業、大学・研究所、個人などが集まって、1994 年 10 月に発足した。SGML の簡易バージョンの開発の結果、1998 年 2 月に XML (eXtensible Markup Language) として W3C 勧告として公表された。

XML は文書型（ドキュメントタイプ）を簡単に定義でき、また、定義なしでも動作する。XML は多言語文書のために文字コードに Unicode を採用していて、しかも XML 文書の検証に対して簡潔であり、効率的な規則がある。産業界は XML の開発と標準化に関わり、即座の成功を立証した。現在では、XML はテキストエンコーディングに広く使用されるだけではなく、メタデータ、データ交換、およびメッセージング（データの送受信）に使用されたりする。XML マークアップのボキャブラリには、コンピュータグラフィック、数式、化学業界、地理情報、書誌学の記録、企業取引、報道などといった多くの分野に適した仕様がある。

2.4 XML を詳しく見ていこう

XML 文書は 7 つの異なった構成要素を含んでいる：

* 要素 (Element)
* テキスト (Text)
* 属性 (Attribute)
* 実体 (Entity)
* コメント (Comment)
* 処理命令 (Processing Instructions)
* CDATA （文字データ (Character Data)）

さらに、文字コードや XML 文書であると認識させるためや、 DTD (DOCTYPE 宣言) を指し示すために文書に先だつ幾つかの構成要素がある。

最小構成の XML 文章は以下のようになる：

```
<firstDoc n="1">
  <p>This is an instance of a "firstDoc" document</p>
</firstDoc>
```

今までのところ述べてこなかった XML ファイルを規定するいくつかのルールがある。一つは、最初の要素が他のすべての要素を含まなければならない、これをルート要素という。上の例では <firstDoc> がルート要素になる。このケースでは、ルート要素は <p> という要素を含んでいる。もう一つは、すべての要素が互いにネストになっていなければならない。その結果、下記の例のような構造は妥当な XML ではない：

```
<p>Two <a>elements, <b>overlapping</a> each</b> other.</p>
```

要素 <a> の開始の後に、要素 <b> が開始しているが、要素 <b> が終了する前に要素 <a> が終了しているために妥当ではない。妥当な XML にするには以下のように要素 <b> を終了してから、要素 <a> を終了させなければならない：

```
<p>Two <a>elements, <b>overlapping</b> each</a> other.</p>
```

例では、要素 <firstDoc> は属性名「n」、属性値「1」を含んでいる。通常、属性は要素に含まれたテキスト（要素の値）に関する追加情報を伝え、また、属性の指定方法を規定するいくつかのルールがある。最も重要なルールは属性の値の開始と終了はシングル・クォーテション「'」かダブル・クォーテション「"」のどちらかのペアで区切らなければならず、属性の名前と属性の値とは「=」で繋げなければならない。

最初のうちは分かりにくいと思うが、専用の XML エディタなら細かなことに注意を払ってくれるので、誰でもすぐに使いこなせるようになる。

2.5 XML スキーマ言語

前節で述べたように、オリジナルの XML の仕様では DTD を使った文書構造とコンテンツの設計（いわゆるスキーマ）を可能にしていた。どのように使用するのかを上記の例を使い、XML 文書に DTD の定義を含めると以下のようになる：

```
<!DOCTYPE firstDoc [
  <!ELEMENT firstDoc (p+) >
  <!ATTLIST firstDoc n CDATA #IMPLIED>
  <!ELEMENT p (#PCDATA)>
]>
<firstDoc n="1">
  <p>This is an instance of a "firstDoc" document</p>
</firstDoc>
```

1 行目の <!DOCTYPE firstDoc [は「firstDoc」という DTD があることを宣言し、2 行目は要素 <firstDoc> が <p> 要素を 1 つ以上持たなければならない。3 行目で要素 <firstDoc> が任意で属性「n」を持つことを意味している。DTD の詳細は「XML 入門」を参照。

こうしている間にも、他のスキーマ言語が開発されている。今のところ、XML 文書では以下の 3 種類が広く使われている：

* DTD は SGML の開発ともにに開発された。ファイル拡張子は「.dtd」
* XML Schema は W3C によって開発された。ファイル拡張子は「.xsd」
* Relax NG（リラクシングと発音）は James Clark と Murata Makoto（村田真）によって開発され、OASIS と ISO で承認されている。ファイル拡張子は「.rng または .rnc」

ここでは細かな違いは述べないが、当面は幾つかの言語があることを知っていれば十分である。それら全ては、文書の形式を定義し、文章が妥当であるかを検証するのに使われるが、それぞれ使用制限は異なる。通常どのタイプのスキーマで書かれているかは、ファイルの拡張子から推測できる。例えば、図 1 で示すように、XML エディタの <oXygen/> は与えられたファイルを検証するためにスキーマと関連付ける機能を持っている。より詳しい <oXygen/> の使い方はユーザガイドを見て欲しい。

図 1：<oXygen/> のツールバー

3 TEI 入門

3.1 背景と歴史

人文科学の研究領域における機械可読なテキストの製作は、早くから始まり、すぐに広まった。文学科学業界のテキストエンコーディングの始まりは、1949 年のロベルト・ブサ (Roberto Busa) 神父が IBM のパンチカードを使った「Index Thomisticus」の仕事が始まりである。1960 年代中頃、3 つの人文科学に焦点を当てた雑誌と 1966 年に出版された機械可読なテキストの一覧表はすでに 25 ページにも達している (Carlson, 1967)。

1965 年のコンピュータと文学の会議で、「テキストエンコーディングに対する標準形式の設立」(Kay, 1965) に関して議論が交わされた。1 つ目の懸案事項は文字のエンコーディングと一貫した識別についてであったが、テキストの構造的特徴と分析的特徴のエンコーディングについても、同様の試みがなされた。

時は流れて、これらの問題はより差し迫ったものになり、1987 年に 欧米コンピュータ利用人文学学会 (ACH: Association for Computers in the Humanities) がニューヨークのヴァッサー・カレッジ (Vassar College) で会議を開催した。その会議では、北米、西欧、アジアといった国々から、各学会、図書館司書、アーカイブ作成者、プロジェクトといった 32 の異なる研究分野からの専門家が出席した。結果は一連の原則が出来上がった、それが TEI (Text Encoding Initiative) の設計原理となり、ACH と 2 つの学術団体の援助を受けて発足した [TEI ED P1, 1988]。

最初の勧告 (P1) の草案は 1990 年 6 月に公開された。その後、活動は 15 のワーキンググループに再編成され、改訂した提案 (P3) を作成し、1994 年の 5 月に初めて活字となって公開された [TEI P3, 1994]。これらのガイドラインは巨大な成功をおさめ、機械可読なテキストを伴うあらゆる人文科学の世界で幅広く使われている。2000 年に設立した TEI コンソーシアムは 2002 年に XML を使用した SGML ベースの P3 を再編成した「P4」を公開した。もう一つの大規模な改訂版である P5 は現在進行中で、名前空間、異なるスキーマ言語や外字と写本記述に対する新しいモジュール（交換可能な構成部分）の追加のような、より XML に対応できるようにする予定である。重要なことは、多くの異なった興味や視点、それから多くの異なる学術団体や公共機関からの専門家達の積極的な参加を伴う多くの研究分野と国々からの莫大な国際協力を通じて日々活発に開発および保守が行なわれているということである。

現在、作業は Sourceforge というオープンソースの開発をサポートしてくれるサービスでを通じて提案書の現在の状況と利用可能な資料を公開する形態で実施されている［`http://tei.sf.net`］。

3.2 TEI コーディング方式の概観

TEI を使ってエンコードされたさまざまな異なる形式のテキストに対応するために、TEI コーディング方式は幾つかのモジュールに分けられ、必要に応じて選択したり組み合わせたりして使う。これらのモジュールのいくつかは、散文作品、詩、演劇、辞書のようにそれぞれの文書の仕様形式合った基本的なタグ一式を提供する。他のモジュールは例えば、異なる文献のテキストクリティカルなエンコーディング、原文の写本、図、リンク、外字、名称、日付などといったものを提供している。

それぞれのモジュールの要素は選択によってセットされた要素一式から削除したり、新たに追加したりでき、既存の要素は編集したり、新たに要素を追加したりできる。TEI のカスタマイズは TEI のガイドラインで定義および、TEI コンソーシアムから利用できるソフトウェアの提供を通じて、すべてなされている。【TEI コンソーシアムからはウェブアプリケーションとして 2 つのカスタマイズプログラムが提供されている。1 つは、P4 に対して［`http://www.tei-c.org/Pizza`］、もう 1 つは P5 に対して［`http://www.tei-c.org/Roma`］である。】TEI コンソーシアムが提供するカスタマイズは TEI エンコーディング方式のある一面である。

脅威的な TEI カスタマイズの発展の見通しに対して、TEI スキーマの 1 つである「**TEI Lite**」という簡単に始められるように、良く使う要素で作られたものがある。TEI Lite はカスタマイズの例や TEI の一般的な使い方

のチュートリアルなども提供している。英語版は［http://www.tei-c.org/Lite/］から利用でき、他の言語にも翻訳されている。

日本語版は［http://www.tei-c.org/Lite/teiu5_jp.html］である。

3.3 TEI 文書の一般的構造

どんな TEI 文書でも <teiHeader> と <text> の 2 つの要素を持つ。<teiHeader> は電子版に関する情報である「メタデータ」をから成り、<text> は原文自身の内容を記述する。

3.3.1 <teiHeader>

<teiHeader> 要素は図書目録に書かれているような著者、出版社、出版年、使用条件といった情報を含んでいる。

下位要素の <fileDesc> は TEI 文書において <teiHeader> の唯一必須な子要素で、電子化文書ファイルの書誌的記述をまとめる。

また、図書目録に書かれていないような情報をヘッダ部分に書くためのいくつかの付加的な項目がある。

* <encodingDesc> 要素は電子テキストとその元になる文書の関係をまとめる。
* <profileDesc> 要素は電子テキストの書誌的記述以外の事項を詳細に述べる。特に、使用言語（副言語）の種類、作成条件と特徴などの説明、関係者、背景等。
* <revisionDesc> 要素はファイルの更新履歴を簡潔にまとめる。

3.3.2 <text>

<text> 要素は単一または複合どちらかのテキストを含む。例えば、詩、演劇、エッセー集、小説、辞書、コーパス（言語資料）といったものが該当する。

この構造を持った西欧流儀の本は数多く出版されており、<text> は主に 3 つの要素、<front>、<body>、<back> を含むだろう。<front> はタイトルページ、目次、序文、献辞といった前置きである。<body> は作品本文で、<back> は付録などの後書きである。この三部構成になっていない文書は<body> 要素だけで記述する。

3.3.3 テキスト内のマークアップ区分

TEI の勧告では文書要素を「構造的」か「流動的」かで分類する。構造的要素は文書の中で現れる場所が制限される。例えば、<head> 要素または見出しは、<list> の中には現れない。流動的要素は名前の通り、より制限を受けず、ほぼ文章中何処でも現れる。例えば <note> や <date> 要素などがそうである。2 つの分類の中間的なもので、<list> や <cit> 要素といった固有の構造を持っているが流動的な特徴を持つものを「クリスタル」 と呼んでいる。

3.3.4 構造的特徴

現在の TEI 勧告では、汎用の階層構造を定義しており、非常に（驚くほど）多くの文書に対応した構造が用意されている。文書は任意の <front>、必須の <body>、任意の <back> に分けられる。テキストの本文は（ 〜 でタグ付けされた）一連の段落の集合、または章、節、小節などである。後者の場合、<body> 要素は一般的に知られている「div」要素で分けられる。与えれた文書の最も大きな集合には、 でタグ付けする。その際、 要素には「type」属性にどのような区分であるのかを明示し、ネストになったすべての区分に対して 要素でマークアップしてもかまわない。散文的文章などは段落要素である でさらに分割される可能性がある。韻文などに対して、行は <l> でタグ付けされ、韻文のまとまりには、<lg> がある。

3.3.5 流動的特徴

先に述べたように、TEI ガイドラインは、多様な流動的特徴に対して名称や定義を提案している。例えば、タイトルやキャプション（一般的に、これらは特定の構造要素に結び付くので、流動的要素には適さない）に対する <head> 要素、引用文や会話文には <q>、リストには <list> 要素と <list> の中の項目に対する <item> 要素、注釈などには <note> 要素、文書作成者のテキストへの編集者の修正に対する <corr> 要素、他にも、語彙的に扱いにくいものを省略形で書くための <abbr> 要素、数字に <num>、名称に <name>、日付に <date>、文献的引用などに <cit>、住所に <address>、文書の言語と異なる単語や熟語に <foreign> といった流動的要素がある。

ここで述べたことは TEI エンコーディング方式のおおよその概略であることは言うまでもない。TEI ガイドラインについてもっと情報が欲しければ、[http://www.tei-c.org/P4X/] を見てほしい。

3.4 TEI 方式を使う

TEI エンコーディング方式を使うにあたって、いくつかの一般的アドバイスを提言する。

最初に、文書の構造の特定について述べ、次に必要な単語や熟語レベルの要素のエンコーディングの仕方に進む。

テキストをエンコードする際に必ず決めなければならないことがある。始める前にこれらについて考えてみよう。

* エンコーディングの際に何を取り込み、何を除外するのか
* どんな要素を使うのか
* どんな属性を使うのか

コーディングは、ある程度はテキストの解釈である。

* 自らの選択を明確にし、首尾一貫して実行する
* 「type」属性の値は、統一すべきである
* 要素は、文書の初めから終わりまで一貫して使うべきである
* 「id」属性は固有の識別に必要とされるが、必ず体系的にエンコーディングするようにする

「出来る」と「すべき」は同じではない

* すべての要素が使われる必要はない
* すべての属性が使われる必要はない
* 必ず、選択したすべてのタグについて選択理由を明確にする

TEI はテキストを画面に表示するのではなく、記述するものである。エンコーディングは埋め込まれたメタデータに焦点を合わすべきであり、そうすれば、よりテキストは柔軟性のあるものになる。汎用性を持たせるためにテキストの論理構造を重視してマークアップを行ない、画面に表示するためだけのマークアップは別の工程にすべきだろう。

参考文献

Carlson, G.,
“ Literary Works in Machine Readable Form ”, *Computers and the Humanities*, p.75-102

Coombs, James H.; Renear, Allen H.; DeRose, Steven J.,
“ Markup Systems and the Future of Scholarly Text Processing ”,
Communications of the ACM, p.933-947,
URL: `http://xml.coverpages.org/coombs.html`

Kay, G.,
“ Report on an Informal Meeting on Standard Formats for Machine-readable Text ”, *Litarary Data Processing Conference Proceedings*, p.327-328

Mylonas, Elli; Renear, Allen H.; DeRose, Steven J.; Durand, David G.,
“ What is text, really? ”, *Journal of Computing in Higher Education*, p.3-26

Sperberg-McQueen, Michael; Burnard, Lou (ed.),
Guidelines for Text Encoding and Interchange (TEI P3), ACH/ALLC/ACL Text Encoding Initiative: Chicago Oxford, 1994

Text Encoding Initiative, *TEI ED P1 Design Principles for Text Encoding Guidelines*, 1988

Text Encoding Initiative Consortium, *TEI @ Sourceforge*,
URL: `http://tei.sourceforge.net/`

Text Encoding Initiative Consortium, *TEI Lite*,
URL: `http://www.tei-c.org/Lite/`