社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2005－DSM－38（4）
2005／8／5

XMLを使った仕様書策定の効率化

只木進一*
佐賀大学学術情報処理センター

大学の基幹情報システムは、多数のサーバと多数の端末から構成されるようになり、またそのカバーすべき領域が教育・研究から事務処理まで拡がりつつある。そのため仕様書の内容は次第に複雑になっている。一方、仕様書は構造化及び定型化された文書の典型でもある。本講演では、仕様書の特性を整理し、XML を活用した作業フローを実例とともに提案する。共同作業やデータベース化の可能性についても議論する。

Effective Writing Processes of System Specifications by XML

Shin-ichi Tadaki
Computer and Network Center, Saga University

University main computer systems become complex mixtures of many servers and terminals. They are requested to support research and education activities, and extended to include systems for university administration managements. Specification documents for university computer systems, as a result, include large amount of description about various types of systems. On the other hand, a specification document is the typical one with well-formed structures and styles. We will discuss a work flow for writing specifications with XML. The possibility of cooperative processes and using database will be discussed.

1　はじめに

情報処理技術が研究だけでなく、日常的な教育や事務処理にまで普及するに伴って、大学の基幹情報システムの構成は大きく変わってきている。利用者情報などの共通情報の統合や機器の統合による効率化、セキュリティーを含んだ一貫した運用方針策定、さらに情報管理組織の統合などによって、基幹情報システムがカバーすべき領域は、大学における活動のほぼ全ての領域へと拡大している。

*e-mail:tadaki@cc.saga-u.ac.jp

大学の基幹情報システムは、1990 年頃までの汎用機とその端末という構成から、サーバ機能を持つある程度の規模の計算機と PC などの独立性を持った多数の小型計算機の構成へと変化した。業務アプリケーションや管理機能という側面では、そうした機能のサーバへの集中化の傾向はあるものの、基幹情報システムは多様で多数の機器から構成されることになる。

基幹情報システムの導入時には、仕様策定が行われる。上記の状況を受け、基幹情報システムの仕様書は膨大な内容を持つものとなり、仕様策定作業に要する人的負荷は次第に過重なものとなっ

ている。特に、基幹情報システムが大学の広い業務をカバーする傾向にあることから、仕様策定において、それぞれの業務担当者との共同作業が不可欠になっている。一方、仕様書は、文書全体に渡って整合的な文体と内容でなければならないため、取りまとめ役には非常に大きな負担がかかる。

一方、仕様書はその形式や構造が整理された典型的な定型文書である。例えばサーバは、ハードウェアとソフトウェアについて箇条書きでその仕様が定められる。更に、同じ仕様に対応する項目は同じ表現でなければならない。また、大規模なシステムの場合には、仕様書と完全に整合する評価基準表も作成しなければならない。

このような定型文書の作成及び管理の方法として XML (eXtensible Markup Language) の活用が考えられる。XML は、文書の構造、文書の印刷書式、及び文書の内容を分離して作成することができる記述言語である。定型的な文書の作成に適しているとともに、同じ内容の文書を異なる書式に出力することも可能である。また、文書は WEB ページ記述言語である HTML と似た単なるテキストであるため、多様な計算機環境で作成することも可能である [1,2]。つまり、共同作業に適している。

本講演では、XML を活用することで仕様策定を効率化することを実例とともに提案する。

2 従来の作業環境とその問題点

2.1 ワードプロセッサ

本節では、まず、従来の作業環境とその問題点について整理する。

最も広く使われている文書作成の作業環境、あるいは多くの人が使うことができる環境は、ワードプロセッサを使うものであることが予想される。

ワードプロセッサの大きな特徴は、文書の内容とその表現が不可分になっている点である。その利点は、作業している文書の印刷形式を直接的にみる、あるいは容易に想像することができることである。また、その機能の一部として、簡単な操作で箇条書きなどの書式制御を行うことができる。しかし、一般にワードプロセッサは動作する OS に強い制限があり、それが共同作業の妨げとなることも多い。

仕様書のような大規模定型文書作成の上で、ワードプロセッサには以下のような本質的問題がある。第一は、ワードプロセッサでは柔軟に文書が作成できることを優先しているため、文書の構造を明確に定義することができないことである。文書作成中にあらかじめ想定した構造から逸脱しても、それを明示的に指摘する機能はない。

また、章ごとに別のファイルに分割されたものに一貫した書式を適応したり、柔軟に書式を変更することが、一般には非常に難しい。更に、内容と書式が不可分であるため、仕様書と総合評価書を別に用意しなければならない。

2.2 LaTeX

ワードプロセッサと比較して利用者数は少ないが、大規模文書作成に使われるもう一つの汎用的環境として LaTeX がある [3]。LaTeX はワードプロセッサと比較して、OS の制限も緩く、文書内容と文書の書式と構造を分離することが容易であるという特徴がある。

LaTeX 文書の内容は単なるテキストであるため、OS に依存せずに編集することが可能であり、複数の人が共同作業するには適している。また、文書を複数ファイルに分割できるため、担当箇所だけを編集してもらうことも可能である。

文書の内容と書式をある程度分割可能であるため、作業当初は単なる箇条書きであったものを、最終的に仕様書に適した書式にすることができる。また、同じテキストから仕様書と総合評価表を出力することも可能である。

更に、柔軟に参照を作成できることと、マクロを定義できること、カウンタを設定できること、計算ができることなどが、大規模文書作成の際には非常に有用である。

一方、書式制御用の命令が文書中にあり目障りであることと、不慣れな利用者にとっては文書編集中には印刷形式が想像できないことが大きな問題点である。

3　XML

前節より、大規模で定型的な文書を作成する場合、以下のような作業環境が望ましいことが分かる。

- 文書の構造、書式、内容が分離できる
- 文書ファイルはテキストである
- 複数ファイルに分割して作業できる
- 定型的文言などをマクロとして定義できる

こうした条件を満たすものの一つが XML (eXtensible Markup Language) である。

XML は文書処理用言語として W3C (World Wide Web Consortium) [4] の推奨する言語であり、SGML (Standard Generalized Markup Language) の簡略版である。また、Web ページの記述言語である HTML (Hyper Text Markup Language) の拡張と考えることもできる。実際、主要な WEB ブラウザが XML ファイルを表示することができる。

HTML では、その言語仕様によってあらかじめ決められたタグによって書式が制御される。タグは文書構造よりも文書書式を記述することに重点が置かれている。例えばタグ<h1> は最上位のヘッダであるとともに、最も大きな文字で表現される文字列を表している。スタイルファイルを使って、構造と書式をある程度分離することは可能である。

また、HTML では Web ブラウザでとにかく表示されることを優先するために、開始タグに対応した終了タグが無いなどのタグの不整合は適宜解釈され表示される。従って、文書構造の不整合は厳しくチェックされることはない。

XML は “extensible” の名称が示すように、タグを定義することが可能である。特に重要な性質として、文書構造を定義する文書 DTD (Document Type Definition) あるいは XML スキーマ、文書を表示する方法を定義する XSL (eXtensible Stylesheet Language)、そして文書の内容を記述する XML に分割できることがある (図 1)。

一度 DTD あるいはスキーマが定義されると、各 XML 文書は、XML に対応したテキストエディタや WEB ブラウザ、あるいは SAXCount [5] などのツールによって DTD との整合性を厳しく確認される。つまり、文章構造の一貫性が守られる。また、複数の XSL を用意することで、xsltproc [6] などのツールにより、単一の XML ファイルから複数の書式、つまりプレビュー用の HTML、仕様書、及び総合評価表を作成することが可能となる。

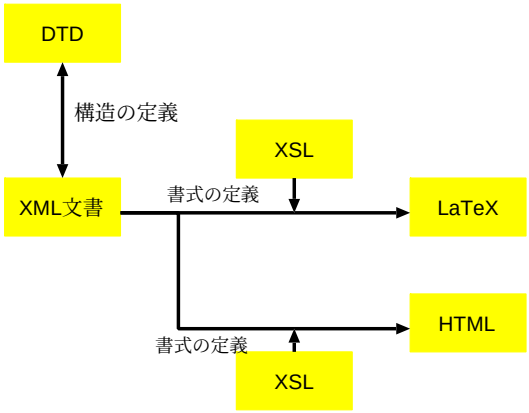

図 1: XML を使った文書作成の流れ：文書の定義は DTD で行われ、他の書式への変換が XSL によって行われる。

XML ファイルそのものは、テキストファイルであり、HTML と非常によく似たファイルとなる。従って、複数人数での共同作業の障害は低くなる。

4　XML を使った仕様策定

4.1　構造の定義

本節では、佐賀大学の例を念頭に、具体的な XML による仕様策定過程を検討する。

仕様書本体 (導入手順などを除く部分) は、大きく三章から構成されている。第一章は、機能に関するものである。ここでは、教育用計算機システムの認証機能や運用体制、事務系システムの業務に必要な機能など、システムアプリケーションに関わる記述が含まれている。第二章は、性能に関するものである。ここでは、使用する各計算機について、そのハードウェアスペックや基本的ソ

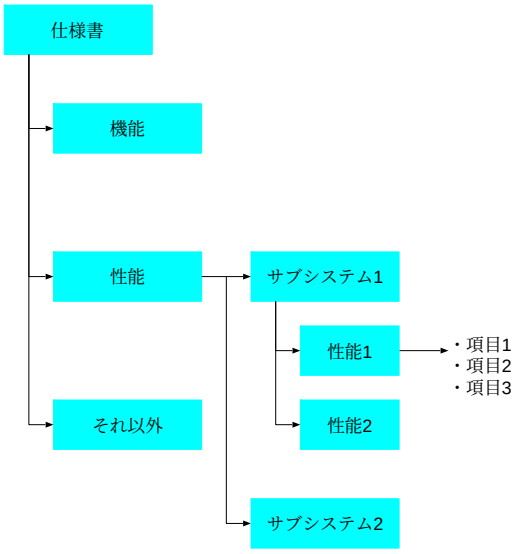

図 2: 仕様書の構造概要：仕様書は、機能、性能、それ以外の「章」から構成され、各機能や各機器は、その詳細の箇条書きで構成されている。

フトウェア構成が記述される。第三章は、導入過程や保守に関するものである (図 2)。

仕様書を XML で記述するために、これらに対応した DTD の階層を定義する。各章をタグ <Chapter>、各サブシステムを <Section>、各機能や各機器を <Subsection> と命名することで、通常の文書のような階層を導入し、作業することができる。CPU 性能や必要なソフトウェアなどの細かい仕様は、最も下層の箇条書きに対応する。この部分のタグには、HTML に対応したタグ を用いることで、XML に不慣れな共同作業者に対応することができる。

仕様書では、他の仕様項目の参照機能が不可欠である。これは、LaTeX の流儀にならって、タグ<label> と <ref> で記述し、区別する文字列を属性として定義する。端末 1、端末 2 などと仕訳されて番号が付けられる場合に対応するために、更に仕訳を区分する属性を定義する。

仕様書では、同じ要件は同じ文言で記述されなければならない。例えば、ハードディスクの要件は、「物理容量◯◯ GB 以上の内蔵ハードディスクを有すること」と容量の数値以外は一貫して記述する必要がある。そこで、そうした定型文を定義するタグ<Desc> を定義する。このタグには、定型文を区別する属性と、数値等の可変部分を与える属性を定義する。例を図 3 に示す。

4.2 表示書式の定義

XML では、文書の内容とその表現が分離されている。XSL は XML からテキストや HTML への変換を定義する XML 形式の文書である。仕様書の場合、内容が整合的な仕様書と総合評価表の組を作成する必要があり、少なくとも二種類の XSL ファイルが必要である。また、プレビュー用として HTML への変換を定義する XSL を作成すると、便利である。ここでは、二種類の LaTeX 文書への変換を中心に説明する。

仕様書の場合、通常の文書と同様にタグ<Chapter> を\chapter へなどと変換すればよい。また最も下層の構造 を箇条書き\item とする。ただし、最下層まで章や節のラベルが付くような LaTeX スタイルが必要である。

総合評価表の場合、全体が表形式になるため、LaTeX の通常の文書構造を表す chapter などを使わず、単にそのラベルを使うようにする。また、タグ Chapter などに評価点を記述することで、点数表とすることができる。

相互引用は\label と\ref を用いることで、LaTeX が持つ標準の引用機能を使うことができる。また、区分された引用も、LaTeX が持つカウンタへと変換する。更に、<Desc> に対応した定型文も XSL に記述する。

5 まとめ

内容が複雑になり規模が大きくなっている基幹情報システムの仕様書を XML を活用することで効率的に作成する方法を提案した。仕様書は、構造が明確であるとともに、定型文の繰り返しである。従って、柔軟な記述が可能なワードプロセッサや LaTeX よりも、構造の定義に忠実な XML が適している。

```
<SubSection base="400" add="100" total="500">
  <Title>利用者用端末 1<ref cat="TERM" href="EDU:1"/>(230 台)</Title>
  <TopTerm base="200" add="100" total="300">
    <Title>ハードウェア</Title>
    <li><Desc name="specpc" arga="3.0"/></li>
    <li><Desc name="memory" arga="1GB"/></li>
    <li><Desc name="nohd"/></li>
    <li><Desc name="cdrom"/></li>
    <li><Desc name="pcvideo"/></li>
    <li><Desc name="pcdisplay"/></li>
    <li><Desc name="displaytop"/></li>
    <li><Desc name="network"/></li>
    <li><Desc name="shortkeyboard"/></li>
    <li><Desc name="pcmouse"/></li>
    <li><Desc name="usbport"/></li>
    <li><Desc name="audioport"/></li>

    <li>小型匡体 (<math tex="\mathrm{W}80\times\mathrm{D}300\times\mathrm{H}200">
        80*300*200</math>(mm) 以下) であること。</li>
    <li>盗難防止策を講じること。</li>
  </TopTerm>
  <TopTerm>
    <Title>ソフトウェア</Title>
    <!-- 以下省略 -->
  </TopTerm>
</SubSection>
```

図 3: XML で作成した仕様書の例 (部分)

XML では文書構造の定義、表示方法の書式、及び文書本体が分離されている。定義を作成することで、Emacs のような XML を理解するテキストエディタや SAXCount のようなツールで、XML 文書が定義に従った整形式であることを確認することができる。

表示方法が XSL という別ファイルに分離されているため、一つの XML から仕様書と総合評価表という二つの文書を生成することができる。また、HTML 用の XSL を用意することで、作業途中の状況を Web ブラウザで見ることができる。

XML は HTML とよく似た文書構造のため、LaTeX よりも多くの人が編集することができる。実際、佐賀大学での仕様策定では事務職員に担当するサブシステム部分の編集を依頼した。

XML は Java と親和性が高いと言われている [7,8]。実際、Java では XML を扱う DOM (Document Object Model) に対応したクラスが標準で利用できる。従って、専用のエディタを Java で構築することが可能である [9]。各サブシステム担当者に部分的に作業してもらう環境を構築するには、そのようなエディタが必要であろう。

仕様書を校正する各機能や機器の表やデータベースから仕様書を作成できれば、非常に便利で

あろう。データベースシステムやスプレッドシードから Java などを介して XML へ変換するツールの開発を行えば、専用エディタよりも策定の効率化が図れることが考えられる。

参考文献

[1] http://www.w3.org/XML/

[2] N. Pitts-Moultis and C. Kirk, *XML: Black Book* (Coriolis, 1999).

[3] http://www.latex-project.org/

[4] http://www.w3.org/

[5] http://xml.apache.org/xerces-c/

[6] http://xmlsoft.org/XSLT/

[7] http://java.sun.com/xml/

[8] B. McLaughlin, *Java & XML* (O'Reilly, 2000).

[9] XML 編集ツールの例：只木進一、日永田泰啓、大月美佳、渡辺健次、渡辺義明、「大学データベースにおける教員情報収集の問題点と解決法」、情報処理学会研究会報告 2004-DSM-34 (2004), pp.49.