

- | |
|---|
| (1) 表題：Web 時代におけるマークアップ言語の教育と普及～SGML と XML を題材として |
| (2) 著者：奥井康弘 |
| (3) 著者の所属, Email アドレス：(株) 日本ユニテック 総合研究所 yokui@utj.co.jp |

はじめに

印刷用の制御コードが徐々にユーザー向けに変化を遂げて出来上がったマークアップ言語は、SGMLを経て現在XMLという形でWebにおける汎用データ記述言語となっている。

XMLが当初想定していた文書記述への適用においては、可視化された文書構造を対象とするため、ユーザーはある程度直観的にタグ付けの習得が可能だった。しかし、現在のWeb環境において、XMLはアプリケーション間のメッセージ交換にも用いられ、そのような適用領域においては、モデリング技術やプログラミング言語との連携も視野に入れて論じる必要もある。その場合、比較的単純なXML文法に対し、オブジェクト指向のクラス設計やプログラミングなど難易度の高い設計手法とのギャップをどのように埋めるかがマークアップ言語の技術者教育における課題となる。

本稿では、習得すべき技術を整理した上で、マークアップの対象領域、対象利用者を分類し、それぞれにカスタマイズされた教育の必要性を指摘する。

1. 電算写植時代のマークアップ言語とその教育

マークアップ言語の当初の適用領域である文書処理では、かつて「電算写植」方式が用いられていた。電算写植は、組版対象となる文字列に専用出力装置に固有な制御コードを埋め込んだデータを処理して紙面への文字レイアウトを含むデザインを行うというものである。従って、そのようなデータはダンプしてプログラマーが解析することはあったとしても、一般ユーザーが読み書きする対象ではなかった。

20 数年前のこの時代、人間がエディターを使ってある程度編集できるレイアウト指定言語の利用が始まった。それが数学者ドナルド・クヌースの考案した TeX である。TeX は、学術論文、特に数式を必要とする文書生成に威力を発揮し、多くの支持者を集めることになった。さらに、これをマクロ化した LaTeX も広く使われるようになった。この辺りから、文書処理の実作業が専門のコンピュータ技術者から文書執筆者へと移った。このため、TeX や LaTeX の学習を一般の執筆者が行うという現象が生じたのである。この場合、学習内容としては、一般的な組版知識と、そのレイアウトを制御するコマンドを覚えることであった。

2. 構造化文書という考え方の出現～SGML の登場～

TeX の普及によって、徐々にコンテンツとしての文字列の中にタグを埋め込むという考え方が一般の人にも受け入れられる素地が出来つつあったが、TeX が普及していた頃、ISO では SGML (Standard Generalized Markup Language) というマークアップ言語が国際

標準となった(1986年)。これは、IBM社の組版システムの機能として提供されていたGML (Generalized Markup Language) を基に汎用的なメタ言語としたものである。

SGMLはメタ言語であり、GMLやTeX/LaTeXのような特定の組版コマンドを規定したものではない。従って、SGMLを理解するには文書をタグで表現するという「考え方」の理解が必要となったのである。この考え方とは、「文書データの構造化」であり、「データのモジュール化」である。

プログラミングのモジュール化はその当時ですでにダイクストラの提唱した「構造化プログラミング」の手法によってプログラマーの間では浸透していたが、SGMLの出現によって「データのモジュール化」に相当する「構造化文書」という考え方が文書処理の世界にも持ち込まれたのである。

文書の構造化のメリットとして、

- データの再利用が可能
- 分割編集が可能

などが挙げられるが、分割編集を行うには、編集に携わる人それぞれが同等の知識を持ってタグ付けすることが必要である。ここに、単なるコマンドの習得とは異なる文書処理分野でのマークアップ言語教育の必要性が発生した。

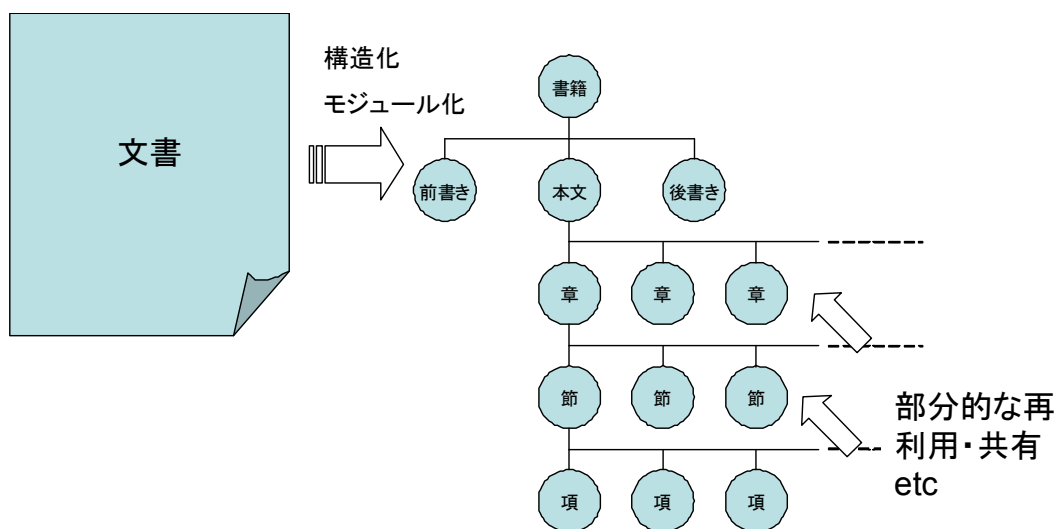


図1. 文書の構造化・モジュール化

3. SGML 普及の壁と HTML の出現

SGMLの教育は文書処理を目的としたものであったため、対象利用者は、組版の知識に加え、文書の構造化という考え方を理解する必要があったが、これが一つの壁となった。

紙面に文字を順番に埋めてゆくという単純な配置という理解からの転換が困難だったのである。構造化の考え方は、プログラマーには馴染みがあったが、一般の執筆者にはタグを入れ子にしてゆくという考え方がなかなかできなかった。そして「なぜ自分が文書構造を考えながらタグを振るという面倒な作業を行わなければならないのか？」という疑問や反発も生じさせることとなった。

そこに HTML (HyperText Markup Language) が登場した。これは、インターネットのブラウザを用いてある程度の文書表示を行うことが目的だったため、SGML と同じ形式のタグは使っていたものの構造化という考え方は前面に出て来なかった。入れ子を含む複雑な文書構造を意識せずに済んだので、SGML が特定の分野（自動車業界などで政府規制で要求されたり、購入先のマニュアルが SGML で来てしまったりなど強制力があつた分野）以外での広がりが抑えられ特殊技能と見られていた中で、HTML はホームページを書きたい一般ユーザーに広く受け入れられていった。

HTML は構造が複雑ではなく、コマンドを埋め込む感覚で記述して表示が可能だったため、構造化という考え方抜きでタグを付けることができた。さらには多少タグ付けを間違ってもパーサーがエラーをださず適当にレイアウトして表示してくれる。この負担の軽さが HTML を受け入れやすいものとし、それによって様々なホームページが作られ、インターネット上の資料の貴重な資産となっている。ユーザーへの普及という観点で HTML は成功を見たのである。このような状況を振り返ると、マークアップ教育を行う上でまず次の認識を持つ必要がある。

基礎認識：

構造化・階層化の考え方の習得は一般利用者にはハードルが高い

4. SGML から XML へ

HTML の爆発的な普及に刺激を受け、SGML 関係者はインターネットに目を向けた。そして SGML 仕様の軽量化が進められ、1998 年 2 月に XML (Extensible Markup Language) が W3C 勧告となったのである。XML は、クライアントでの処理の負担を軽くするために DTD チェックをオプションとし、「整形形式 (Well-formed)」という処理モードを導入した。これによって、処理速度は著しく改善された。また、開始タグと終了タグの対応さえ取れば良く「自由に」¹タグが書けるということで心理的なハードルはかなり低くなった。

もう一つ XML によって大きく変わった状況がある。それは、マークアップが文書処理の世界からコンピュータープログラムが扱う他のあらゆる分野のデータ記述言語へと活用が広がったという点である。この状況によって、タグで表現する対象物が多岐に渡り、それ

¹ この「自由に」というのは誤解を招く表現であって、文書交換を行うのであれば、交換するもの同士ではタグの意味や構造に関する合意がなければならない。したがって、勝手にタグを付けたとしても、それはその個人でしか使うことができない。XML 普及の要因となったとは言え、実際には意味がないデータとなってしまうのでこのような XML の捉え方には注意が必要である

らに関わる人の層も広がった。一言で XML 教育と言っても、対象業務領域別、対象利用者別に論じなければならなくなったのである。

5. 対象利用者・対象業務領域に合わせたマークアップ言語教育の必要性

このような変遷を経て現在の汎用マークアップ言語 (XML) が存在するが、XML に関わる業態・業種にはどのようなものが存在するのかを整理することが、マークアップ教育を考えるにあたっての出発点となる。そこで、XML に関わる仕事を「対象業務領域」「対象利用者」という 2 つの軸で整理してみよう。

まず「対象業務領域」としては大きく分類して以下のものが挙げられる。

表 1. マークアップの対象業務領域

業務領域	説明
文書処理	出版業などで用いられる本格的な印刷機能をもった出版システムから DTP、ワープロ、Web ブラウザに至るまでのテキスト情報の可視化。
メッセージ交換	SOA などによってモジュール化されたシステムコンポーネント間の入出力となるデータを XML として送受信する。
アプリケーション内部データ	オフィス製品などの格納データを XML として表現し、場合によっては ZIP などによって圧縮し利用する。

「対象利用者」としては、「データ作成者」「プログラマー」「スキーマ開発者」「システム設計者」が存在する。

表 2. マークアップの対象利用者

利用者	説明
データ作成者	与えられた XML スキーマに従って XML インスタンスを作成する。
プログラマー	出来上がった XML インスタンスを加工処理する。データベースへの格納／取り出し、データ変換、データ統合など一般的なデータ処理を行う。
スキーマ開発者	XML 利用の基礎となる XML のタグセットを情報の構造化を行いつつ定義する。
システム設計者	XML を利用するシステム全体の設計を行う。

本稿では「教育」をテーマとしているので、教育対象となる対象利用者に注目して、習得すべき (教育すべき) 技術について考察する。

5.1 「データ作成者」の習得技術

データ作成者は、タグを自らあるいは補助手段を用いてテキストに埋め込む必要がある。したがって、XML のタグ付けを行うための基本文法の習得がまず必要となる。これは「文書処理」の分野に特有な習得技術である（ここでは、「文書処理」は、Web のフォーム形式での入力のような項目入力も含めたものとして捉えている）。

HTML のようにフラットな構造の場合はどのように書いても表示は可能であるが、DocBook や DITA などの場合には、スキーマ（DTD や XML Schema など）で定義されたタグの階層構造をしっかりと意識してタグ付けを行わなければならない。例えば、SGML 規格書（ISO 8879）の付録で記述されていた General DTD では、リスト構造のネストがあった場合のタグの書き方が非常に複雑であり、これを理解して書くことには多くの人が困難を覚える。このような場合、ブラウザやフォーマッターを使ってレイアウトを確認しつつタグ付けすることになるだろう。

「メッセージ交換」「アプリケーション内部データ」の対象業務領域においては、データ作成者は一般に介在しない。

5.2 「プログラマー」の習得技術

プログラマーの扱うデータ処理は、対象業務領域によって XML に関連した周辺技術が異なるため、それぞれに分けて習得技術を整理する。

■ 「文書処理」の場合

この場合は、XSLT/XPath などを使って、例えば XBRL (eXtensible Business Reporting Language : 財務報告などの情報を表現できる XML ベースの言語) などの業界標準スキーマから HTML などを生成し、画面表示させるなどの操作を行うことになる。本格的な印刷仕様に対応するのであれば、XSL-FO などの学習も必要であろう。

■ 「メッセージ交換」の場合

Web サービスの基礎となる SOAP、WSDL を習得し、それらを受け付けるシステムの使い方を知る必要がある。

■ 「アプリケーション内部データ」の場合

DOM や SAX などを使ったプログラム作成を行う。最近では XML を扱う API も充実してきているので、XML を扱うプログラマーは、データ処理の基本として XML ツリーのノードを扱うための API の知識が必要となる。ここでも重要なのはツリー構造の考え方である。XSLT/XPath を使う場面も多い。注意が必要なのは DOM モデルと XSLT で使用する XPath モデルが異なるという点である。

5.3 「スキーマ開発者」の習得技術

スキーマ開発も対象業務領域によって扱うデータの構造が大きく異なるため、それぞれに対して習得技術を整理する。もちろん XML Schema 文法の習得は大前提である。

■「文書処理」の場合

単純なメモから書籍やマニュアルなどの本格的な文書情報を表現するには、様々な形態の文書構造を理解している必要がある。特に基本モジュールとしての段落構造の階層化があるが、これにはリストの入れ子構造から強調句の入れ子に至るまで複雑な構造を把握してタグ付けに反映させる作業となる。これらの習得方法に関しては、マークアップ言語の発展に伴い文書処理系のスキーマ（当初は SGML DTD）が長年工夫されてきたので、それらを系統的に学習することが効果的である。

既存 DTD を理解した上で試行錯誤しつつ習得するといった側面もあるので、経験も大きく影響してくる。ある意味でセンスも必要とされる分野である。

■「メッセージ交換」の場合

メッセージは、データの受け渡しやパラメータの受け渡しなどに用いられるので、文書処理とは大きく様相が異なる。これはアプリケーションのデータ設計の一部である。昨今のアプリケーション開発手法であるオブジェクト指向的なアプローチが求められる。そこで、スキーマ開発者には UML などのオブジェクト指向のモデリング技術の習得も求められることになる。後述の 6 章で説明する手法などを用い、そのようにして出来上がったモデルを XML のスキーマへとマッピングするのである。

■「アプリケーション内部データ」の場合

アプリケーションの XML 化は、多くの場合、データ分析を行ってタグの構造設計をするというボトムアップではなく、既存のデータ構造をどのように XML にマッピングするかを検討作業と考えられる。これには、表現手段は要素か属性か、要素なのか処理命令なのかという判断が含まれる。特に階層構造をまたがるような制御情報を XML としてどのように実現するかは悩みの種であろう（SGML の場合には CONCUR 機構によって構造をまたがる情報を記述することも可能だったが XML にはそのような機構がないため）。

5.4 「システム設計者」の習得技術

システム設計者は、文書処理、メッセージ交換、アプリケーション内部構造のいずれにおいても、業務分析、システム分析を行うこととなる。もちろん、対象業務領域が違えば、分析内容は異なる。XML 全般に渡る知識、経験を基礎に必要があるという点を除けば、一般のシステムアナリスト、SE が行っている仕事と同じような作業と考えることができる。

5.5 対象利用者・対象業務領域別の習得技術の整理

ここまでに説明した内容を「対象利用者」「対象業務領域」の組み合わせで整理すると、習得すべき技術（教育内容）は以下のようなになる。

表 1. 対象利用者・対象業務領域別の習得技術

対象業務領域 対象利用者	文書処理	メッセージ交換	アプリケーション内部 データ
データ作成者	<ul style="list-style-type: none"> XML 文法 文書の階層構造の理解 		
プログラマー	<ul style="list-style-type: none"> XSLT/XPath XSL-FO 	<ul style="list-style-type: none"> SOAP、WSDL 	<ul style="list-style-type: none"> DOM、SAX XSLT/ XPath
スキーマ開発者	<ul style="list-style-type: none"> XML Schema 文法 文書分析 既存スキーマの分析 (経験に基づくセンス) 	<ul style="list-style-type: none"> XML Schema 文法 データ分析 UML などのモデリング言語 	<ul style="list-style-type: none"> XML Schema 文法 アプリケーション固有のデータ構造の理解
システム設計者	<ul style="list-style-type: none"> 業務分析 システム分析 	<ul style="list-style-type: none"> 業務分析 システム分析 	<ul style="list-style-type: none"> 業務分析 システム分析

5.6 マークアップ言語教育のために必要な習得技術スタック

表 1 から、対象業務領域によるある程度の差はあるが、対象利用者別の以下のような習得技術のスタック構造が見えてくる。

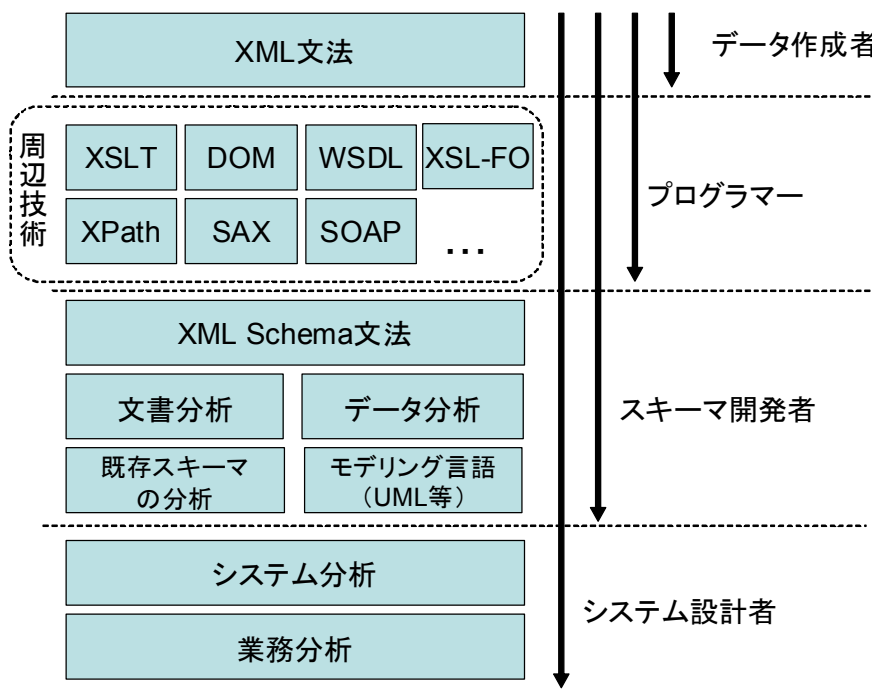


図 2. マークアップ言語のための習得技術スタック

マークアップ言語教育においては、このスタック構造を参考にカリキュラムや内容を検討することができるであろう。

6. スキーマ開発におけるオブジェクト指向的アプローチ

スキーマ開発に関連して触れたモデリング技術の活用について、ここでいづらか詳しく説明する。

XMLによるデータのモジュール化／階層化の考え方は、容易にモデリング技術との連携の可能性を想起させる。XML Schemaをデータのモデル化の結果だとすれば、前段のモデリング過程での理論的支柱があれば、技術者にとってそれに取り組みやすく、また方法論も確立することができる。このような観点で利用される技術にUML (Unified Modeling Language)がある。UMLはOMG (Object Management Group)で開発された技術であるが、UML1.4.2が国際標準ISO/IEC 19501となっている。UMLで定義されたモデル情報を交換するための仕様にXMI (XML Metadata Interchange)があるが、これもXMI2.0がISO/IEC 19503となっている。これらの標準技術を組み合わせてモデリング技術とXML定義を結び付けることが可能となる。

UMLで作ったデータをXMIのような中間表現を介してXMLスキーマへ変換するという試みもあるようであるが、この場合、UMLのクラス図をどのように一般的にはクラスをXMLの型とし、プロパティを子要素とすることが多いようである。このような場合、スキーマ開発者にはUMLなどのモデリング言語の知識も求められる。

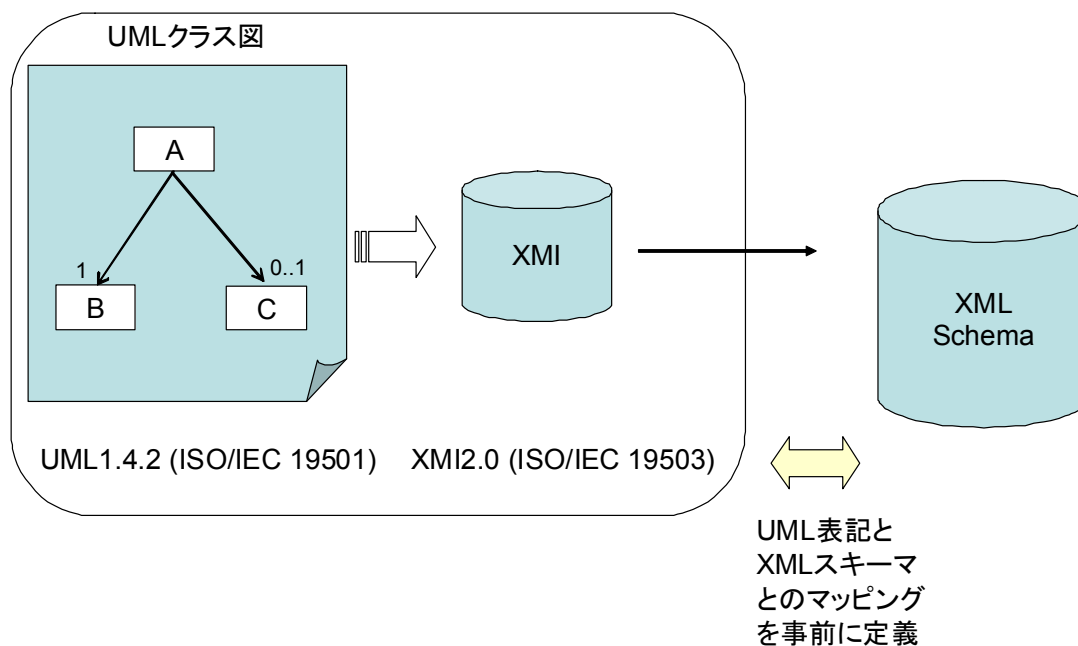


図3. モデリング技術とスキーマ定義の連動

7. データ作成習得のためのツールの利用可能性

マークアップ言語の導入にあたっては「誰がタグ付けを行うのか」という問いが投げかけられる場合があることを前述したが、XML インスタンスが蓄積されなければ XML の利用価値はない。しかし、XML はそのタグ付けが難解であると受け止められ、なかなかデータ作成者が育たないというジレンマがある。

文書データをタグ付けするには、レイアウト情報という視覚情報を基に、段落などの矩形の配列から構造情報を想起し、それをツリー構造にマッピングするという思考が求められる。これは一般ユーザーには至難の業である。そこで、“ツールを使ったタグ付け支援” “タグを意識させない入力” などのニーズが発生する。

ユーザーがタグやその構造を意識せずにタグ付けするためには、何らかの支援ツールが必要であるが、アイコンなどを使った汎用入力ツールなどが SGML 時代から開発され市場に提供されていた。しかし、文書処理のタグは数も多く構造も複雑であり、実用上問題があった。複雑なタグ構造の例としては以下のものが挙げられる。

- 文書構造のネスト ⇒ 章・節・項の入れ子など
- 段落構造のネスト ⇒ リストの入れ子など
- 段落内のネスト ⇒ 強調句の入れ子など

比較的フラットな構造で、穴埋め形式で情報を入れることのできる極力簡単で固定的なスキーマでなければ人手でタグ付けを行うのは難しい。ここに、マークアップ言語を使いこなせるデータ作成者が一般に育たず、普及が阻まれる要因を見ることができる。

これまで 10 数年に渡ってワープロソフトや DTP ソフトで SGML や XML のタグへのマッピングを実現すべく様々なソフトウェアが市場に出てきたが、特定の業務領域において専門的なカスタマイズや教育を行った上で利用することを除けば、一般ユーザーにまで普及を見ていないのが現状である。タグをユーザーから隠すのであれば、数式を MathML に変換するソフトウェアのように、対象とするタグが完全に固定的な専用ソフトウェアにならざるを得ない。

結果として、ツールについて言えば、特定のスキーマにのみ固定的に対応したものや、度現在の Web フォームのような穴埋め的なもののみがかろうじて使用に耐えるのみである。

まとめ

XML に代表されるマークアップ言語の昨今の幅広い利用分野を考えると、その教育には、XML 文法、XSLT などの周辺技術、XML Schema といった XML 関連技術だけでなく、プログラミング言語、モデリング言語などを含めた総合的な知識を学習者に提供することが求められている。

一方で、XML をデータ表現の一つの手段と捉えれば、規格内容の理論的学習だけでは不

十分である。スキーマ開発やインスタンス作成においては、学習と同時に慣れによる習熟も必要である。本文中でも触れたが、経験、ノウハウ、さらにはセンスが関係する場合もある。これは、本稿で整理したマークアップ言語（主に XML）のための「習得技術スタック」に対して対象利用者向けの理論的教育を施すと同時に、「見て」「書いて」「動かして」という実習・実地訓練にも時間を割く必要があることも意味している。

Web 時代にあって XML によるデータ交換は進み、アプリケーションの内部データも圧縮技術と共に XML 化される可能性がある。まさにマークアップ言語教育はこれからの ICT 技術教育の要として位置づけるべきである。その広がりを見ると、対象業務領域、対象利用者を念頭においた体系的、実践的な教育が求められている。

参考文献

- [1] 日本ユニテック SGML サロン『はじめての SGML』技術評論社、1995
- [2] 中山 幹敏, 奥井 康弘 編著『改訂版 標準 XML 完全解説 (上) (下)』技術評論社、2001
- [3] 日本ユニテック Web 技術研究グループ『はじめて読む XML～標準データ記述言語入門～』アスキー、2005