2004－DD－42　(5)
2004／1／30

木・表構造間写像モデルに基づく XML-HTML 変換用スタイルシート自動生成方式

今村 誠　　増塩 智宏　　伊藤 山彦
E-mail: {imamura, masushio, titoh}@isl.melco.co.jp
三菱電機（株）情報技術総合研究所 音声・言語処理技術部

インターネットの普及と EC(Electronic Commerce)の進展に伴って、資材伝票や設計仕様書を XML(Extensible Markup Language)形式で Web ブラウザから入力参照するシステムがさかんに開発なされるようになった。しかし、対象とする XML 文書の論理構造が複雑になるについて、XML 文書インスタンスの画面中の入力枠に対して XML 要素を対応付けていく従来の表示中心画面設計では、入力画面用に XML を HTML(HyperText Markup Language)に変換するスタイルシートの作成が難しくなるという問題点があった。特に、あるデータ項目の内容に応じて表の行や列の繰返し数が変わるような可変 XML 文書を扱うことが困難であった。本稿では、この問題を解決するために、テーブル形式の GUI(Graphical User Interface)を用いて、XML 構造定義に対して、画面レイアウト用の文書スタイル、入力チェック用の文書内容制約、および担当業務毎のアクセス権限情報などを付与するだけで XML 入力用のスタイルシートを自動生成する XML 文書画面自動生成方式を提案する。本方式の技術特長は、XML 文書がもつ木構造から HTML 文書がもつ表構造への写像を表現する木・表構造間写像モデルを用いることにより、XML 要素に対して表パターンを指定するだけで表組みスタイルを自動生成できる点にある。また、本方式は、エンジニアリング用の設計仕様書と Web-EDI(Electronic Data Interchange)用の資材伝票を対象とする評価により、「従来の表示中心画面設計では困難であった可変 XML 文書を扱うことができること」、また「従来の表示中心画面設計による XML 入力画面作成支援ツールと比較して、入力画面開発の工数を約 1/2 削減できること」を確認した。

A Stylesheets Generation Method for XML-HTML Transformation Based on a Tree-Table Mapping Model

Makoto IMAMUMA　Tomohiro MASUSHIO　Takahiro ITOH
E-mail: {imamura, masushio, titoh}@isl.melco.co.jp
Mitsubishi Electric Corporation Information Technology R & D Center Human Media Technology Dept.

Many web browser based XML document processing systems for EDI or engineering have been developed in a company with the spread of　EC. But existing view centered input-form design methods, corresponding XML elements to input frames of an XML document instance, are difficult to cope with documents which have complicated logical structure. For example they are difficult to cope with variable structure document, in which the number of columns or rows of a table depends on the content of the other XML element. This paper presents an XML input form generation method, corresponding style information, input check rules, access authorization rules to XML elements by spread sheet GUI. The feature of the method is based on a tree-table mapping model, which enables XSLT stylesheet generation function for tables.　Evaluation of our method for engineering design manuals and Web-EDI input forms shows the following results. (1) It can cope with variable structure documents, which are difficult to be cope with by view centered input-form design methods. (2) It can cut down the half of the time for developing XML input forms compared with the view centered input-form design methods.

1 はじめに

インターネットの普及と EC の進展に伴って、資材伝票や設計仕様書を XML 形式で Web ブラウザから入力参照するシステムがさかんに開発なされるようになった。しかし、対象とする XML 文書の論理構造が複雑になるについて、お絵かきツール風の GUI を用いて作成した XML 文書インスタンスの画面中の入力枠に対して XML 要素を対応付けていく従来の表示中心画面設計では、入力画面用に XML を HTML に変換するスタイルシートの作成が難しくなるという問題点があった。特に、あるデータ項目の内容に応じて表の行や列の繰返し数が変わるような可変 XML 文書を扱うことが困難であった。可変 XML 文書を扱う具体的なアプリケーションとしては、エンジニアリング分野における設計仕様書がある。

本稿では、この問題を解決するために、テーブル形式の GUI を用いて、XML 構造定義に対して、画面レイア

ウト用の文書スタイル、入力チェック用の文書内容制約、および担当業務毎のアクセス権限情報などを付与するだけで XML 入力用のスタイルシートを自動生成する XML 文書画面自動生成方式を提案する。本方式の技術特長は、XML 文書がもつ木構造から HTML 文書がもつ表構造との写像を表現する木・表構造間写像モデルを用いることにより、XML 要素に対して表パターンを指定するだけで表組みスタイルを自動生成できる点にある。

本稿の構成は、以下の通りである。2 章では、本稿で扱う技術課題について述べる。3 章では、本稿で提案する要素技術の中核である木・表構造間写像モデルについて述べる。4 章では、3 章で述べたモデルに基づく XML 文書画面自動生成方式を提案する。5 章では、エンジニアリング用の設計仕様書と Web-EDI 用の資材伝票を対象とする評価により、4 章で提案した方式が、2 章で述べた課題を解決していることを示す。6 章では、提案方式の効果に対する考察と今後の課題を述べ、7 章ではまとめを述べる。

2 XML 入力画面作成方式における課題

2 章では、本論文で扱う技術課題について述べる。より詳細には、2.1 節で、エンジニアリング分野における設計支援を例としてとりあげ、設計仕様書に対する文書処理への要求を分析する。2.2 節では、従来の表示中心画面設計の特徴と問題点を中心に、XML 入力画面開発方式の技術課題について整理する。

2.1 文書処理アプリケーションの要求分析

エンジニアリング部門の設計支援業務における文書処理では、設計対象物の構成や設計業務の内容を反映して、設計仕様書やデータシートなどの対象文書を処理するための要求機能が複雑になる。その複雑さは、文書の構造や内容の制約の複雑さ、文書の画面 IF の複雑さ、およびワークフローでの文書データ活用の複雑さの３点に集約できる。以下、３つの複雑さについて順に説明する。

(1)**構造や内容の制約の複雑さ(文書制約設計の問題)**

DTD(Document Type Definition)や XML Schema などで規定される文書構造定義(文書中の記載項目の出現順序や繰返し出現数など)が設計対象物の構成を反映して階層木の構造をもつだけでなく、「ある記載項目の内容と、ある記載項目の存在有無や繰り返し数との関係制約(**内容・構造間制約**)」、「文書中の記載項目の内容間の制約(**内容間制約**)」、および「記載項目の内容と、前 Web 画面入力値との制約(**文書外制約**)」をもつという特徴がある。本稿では、構造や内容の制約の複雑さに対処するための文書設計上の問題を「**文書制約設計の問題**」と呼ぶ。

図 1に、エンジニアリング部門の設計支援向けの XML 文書例として、昇降機の設計仕様書の文書構造の一部を示す。昇降機の設計仕様書では、仕様が基本仕様、機械仕様、電気仕様、および意匠仕様などの仕様区分をもち、例えば以下の制約をもつ。

(i)　内容・構造間制約の例： 昇降機のかごの数やビルの階数に応じて、基本仕様のかごサイズ、機械仕様の停止階、電気仕様の非常用昇降機、および意匠仕様のドアタイプなどの仕様項目の数が動的に変わる。本稿では、文書構造定義を満たしながらXMLインスタンスの反復数や選択肢などの構造が変わる XML 文書の構造を**可変 XML 構造**と呼び、また、可変 XML 構造に対応する文書入力画面を**可変 XML 画面**と呼ぶ。

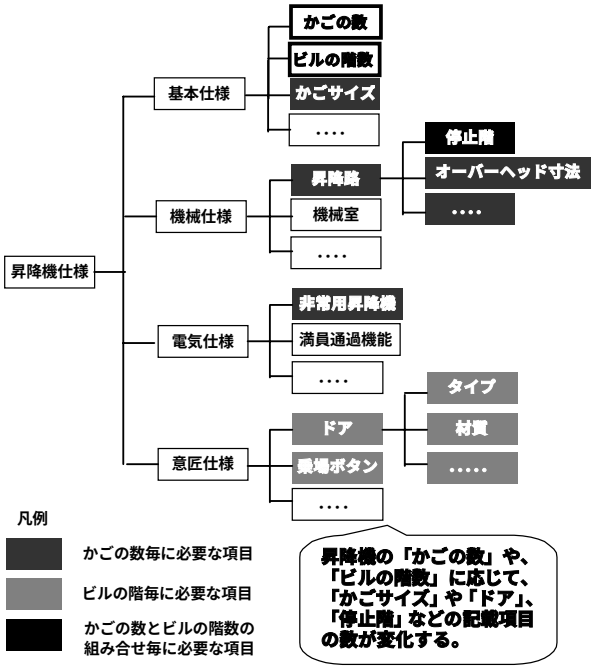


図 1　XML 設計仕様書の例

(ii)　内容間制約の例： 「停止階数と階高を変えると、レールの本数が変わる」、また「自動音声は、センサが無いと取り付けられない」など。

(iii)文書外制約の例： 「担当者名」や「製品番号」は、前の Web 画面で入力され、それらの値を仕様書の対応する項目の内容に挿入しなければならない。

(2)　**画面 IF の複雑さ(文書スタイル設計の問題)**

文書の構造や内容がもつ制約が複雑になるのに伴って、見やすさや入力チェック制御のための画面 IF への要求を満足する必要がある。本稿では、画面 IF の複雑さに対処するための文書設計上の問題を「**文書スタイル設計の問題**」と呼ぶ。

(i)**見やすさへの要求(階層構造の表による表示)**

文書がもつ階層構造を目次や表形式で見やすく表示したり、文書中のデータ項目間の関係をわかりやすく画面に提示する必要がある。ここでの主な問題は、「階層構造の表による表示」の問題である。これは、XML 文書中のデータを表形式で見やすく表示するためには、

XML 文書タグの階層構造(木構造)を行と列からなる表の構造への写像を指定する必要があるが、この指定は必ずしも容易でない。

(ii)**入力チェック制御**

文書入力画面では、(1)で述べた文書制約を満たしているかどうかを検証する入力チェック機能の起動タイミング、警告メッセージ表示、および警告メッセージから警告個所へのハイパーリンク付与などの機能が必要になる。

(3)**ワークフローでの文書データ活用の複雑さ(ワークフロー文書活用設計の問題)**

ワークフローシステムでは、「担当分野毎・アクション毎の画面提示」や「文書データの業務システム連携」の要求がある。本稿ではワークフローでの文書データ活用の複雑さに対処するための文書設計上の問題を「**ワークフロー文書活用設計の問題**」と呼ぶ。

(i) **担当分野毎・アクション毎の画面提示**

ワークフローシステム中の文書画面では、文書にアクセスする人が担当者か管理者か、あるいは、機械設計者か電気設計者かといった担当分野に応じて、どの項目を見せるか、またどの項目を編集させるのかが異なる場合がある。また、同じ担当分野でも、表示画面や入力画面によっても異なり、また、人手の内容チェックを支援するために、前版の文書との差分をわかりやすく表示する機能も要求される。

(ii) **文書データの業務システム連携**

作成された文書データは、後工程の設計業務用システムなどのレガシーシステムに受け渡すために、データ変換が必要となる。

2.2 XML **入力画面開発方式の技術課題**

XML 入力画面開発が従来の入力画面方式と最も大きく異なるのは、画面表示レイアウトと文書構造定義(DTD や XML Schema をさす)との対応付け(**表示・論理対応付け**とよぶ)という工程が存在する点にある。画面自体は XML インスタンスに対応するのに対して、文書構造定義はインスタンスを生成する文法にあたるので、2.1 節で述べた可変 XML 画面を作成にする際には、この表示・論理対応付けが困難になるという問題が生じる。HTML では、そのフォームでは入力枠に１対１に対応した ID しか付与できないのに対して、XML では文書構造に文法記述を持ち込むことにより構造記述能力を向上させているので、XML では文書表示に関して新しい課題が生じたわけである。また、XML 文書設計という観点からは、表示設計と論理設計を分離することが重要なのであるが、可変 XML 画面では両者を簡単に分離できないという実際的な問題を提示しているといえる。

本節では、既存の XML 入力画面の開発方法を、XML 文書インスタンスの画面レイアウトに文書構造定義を対応させる**表示中心画面設計**と、文書構造定義に直接画面レイアウト定義を付与する**論理中心画面設計**に大別して、その各々の概略と問題点について述べる。

2.2.1 **表示中心画面設計**

表示中心画面設計に基づく XML 入力画面作成用ツールとしては、Adobe 社の FormDesigner[1]や Micorosoft 社の InfoPath[2]などがある。いずれもお絵かきツール風に入力画面を作成した後で、入力枠に対して XML 要素を対応づけていくことにより入力画面を作成していく。紙帳票のように A4 一枚に入力項目を精密に配置するといった複雑なレイアウトをもつ画面を作成する場合に適しているが、2.1 節で述べたようなエンジニアリング向けの文書構造自体が複雑な場合には、以下に示すような問題点がある。以下、2.1 節の要求毎に、表示中心画面設計の問題点について述べる。

(i) **文書制約設計での問題**

設計仕様書のように製品仕様に応じて記載項目が動的に変わる入力画面を作成することができない。あるいは、スクリプト言語を用いて画面制御プログラムを記述するので、お絵かきツール風の簡易な表示画面作成という特長が結局生かせなくなる。

(ii) **文書スタイル設計での問題**

GUI インタフェースにより入力枠や選択メニューなどのオブジェクトを貼り付けながら入力画面を作成していくので、表のように類似の記載項目が繰り返し規則的に出現する場合にはコピー貼り付けする手間が大きいという問題がある。あるいは、ツールがもつ表示・論理の対応付けパターンが限定されているために、お絵かきツールで作った自由なレイアウトをもつ画面と、与えられた文書論理構造とを対応付けができない場合がある。

(iii) **ワークフロー文書活用設計での問題**

担当分野毎に入力画面を作成するので、共通部分が変わった場合でも、担当分野毎画面の数だけ修正作業が必要になり、担当分野毎に異なる入力画面の作成や保守が手間であるという問題がある。

2.2.2 **論理中心画面設計**

論理中心画面設計に基づく XML 入力画面作成用ツールとしては、Altova 社の XML Spy[3]がある。論理中心設計では、XML 文書の構造定義に対して、画面スタイル定義を付与していく。具体的には、XML 要素に対して、テキストのフォントや色などの属性や、テーブルや段落といった表示上の設計情報を対応づけていく。

表示中心設計が特長とする複雑なレイアウト画面を作成する場合に適していないが、2.1 節で述べたようなエンジニアリング向けの文書構造自体が複雑な制約を自然に扱えるという特長がある。しかし、複雑な制約に

ついては、結局のところ、スクリプト言語によるプログラムミングが必要になるので、トータルな開発工数自体は、表示中心画面設計より大きくなる傾向にあるという問題点がある。

2.3 本稿で扱う課題

本稿の目的は,2.1 節で述べたアプリケーションの要請に応えることができ、かつ 2.2 節で述べた技術課題を解決する XML 入力画面自動生成方式を提案することにある。本方式における従来の問題点を解決するための工夫は、以下の 2 点に集約できる。

(1) 表・木構造間写像ﾓﾃﾞﾙによる表組みﾚｲｱｳﾄ自動生成

表示中心画面設計の問題点である可変XML構造を簡単に扱えるようにするために、表・構造間写像モデルを構築することにより、2 章の冒頭で述べた表示・論理対応付けの問題を解決する(3 章)。そして、論理中心画面設計の考え方に本モデルを組み込んだXML 入力画面自動生成方式を提案する(4 章)。本方式では、表・構造間写像モデルが提供する表写像型を構造定義に付与するだけで、表組みレイアウトを自動生成できるので、XML 入力画面の開発工数を削減することができる(5 章)。

(2)使いなれた表計算ｿﾌﾄｳｪｱによる XML 文書設計情報の編集

提案方式では、使いなれた表計算ソフトウェアを用いて、XML 要素毎に、2.1 節で述べた文書制約、文書スタイル、ワークフロー文書活用などの XML 文書設計情報を簡単に設定できるようにした。また、この XML 文書設計情報からXML 入力用のXSLT(XSL Transformations)記述を自動生成する機能を開発することにより、表計算ソフトウェアを用いて簡単に XML 入力画面を開発できるようになる。

3 木・表構造間写像モデル

木・表構造間写像モデルは、表組みレイアウトを末端の構造としてもち、末端の表組みをリスト配置や入れ子見出しなどにより組み合わせ構成していく構造をもつ**階層表組み画面**用のスタイルシートを自動生成するための基礎となるモデルである。本モデルを実装することにより、4 章で説明する XML 文書画面自動生成方式では、木・構造写像型を指定するだけで、可変 XML 画面を自動生成できるようになる。

本章では、3.1 節で、XML 断片から表組みレイアウトへの写像を表現するテーブル写像について述べ、3.2 節で XML 文書構造から階層表組み画面への写像を表現する階層表組み写像について述べる。

3.1 テーブル写像

テーブル写像には、基本テーブル写像と、基本テーブル写像の張り合わせによる連結テーブル写像とがある。

3.1.1 基本テーブル写像の像と原像

基本テーブル写像は、XML 構造の断片から HTML の表組みレイアウトへの写像である。本項では、写像の像を表現する表組みスタイルと、表組みスタイルへの写像の原像を表現する XML 断片の構造パターンについて述べた後に、基本テーブル写像の型を定義する。

(1)基本テーブル写像の像(HTML の表組み画面)

基本テーブル写像の像である表組みスタイルは、図 2 に示すように、値部分と見出し部分からなる。値部分が行と列の構造をもっており、XML の要素内容に対応する部分である。行と列が XML の可変要素に対応するかどうかで、固定、行可変、列可変、行列可変の 4 つのタイプに分類できる。また、見出しは、行と列の説明に相当し、階層構造をもつことができる。

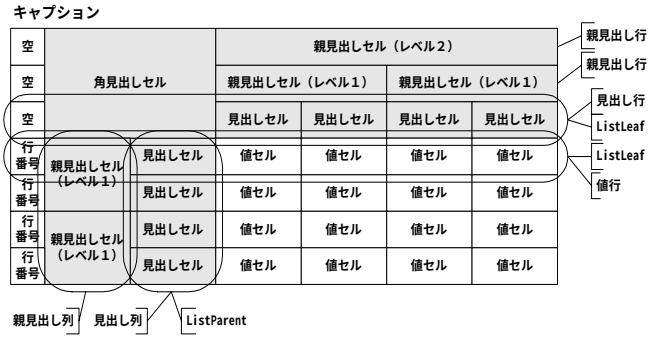


図 2　表組みスタイル

(2)基本テーブル写像の原像(XML 断片)

基本テーブル写像の原像の構造パターン(**XML 断片型**)は、表 1のように、S 型(sequence)、R 型(Repetition 型)、S(R)型、R(S)型、および R(R)型に類別できる。各々の型は、表組みスタイルの値セルを格納する XML 要素を示す Xpath 式、または、Xpath 式のリストにより特徴づけられる(**特徴 Xpath、**または **Xpath リスト**と呼ぶ)。

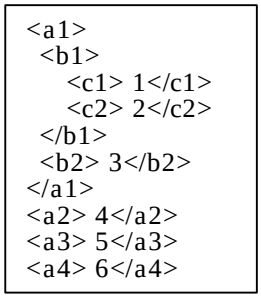
表 1　基本テーブル写像の原像となる XML 断片パターン

型	説明
S 型	要素の並びにより構成される反復要素を含まない構造。構成する要素を指し示す Xpath のリストにより特徴づけられる。
R 型	反復構造。反復する要素を指し示す Xpath により特徴づけられる。
S(R)型	R 型構造が並ぶ構造であり、かつ R 型断片の反復数が互いに等しい構造。反復要素を指し示す Xpath(PCDATA を内容として含む末端要素を指し示す Xpath)のリストにより、特徴づけられる。
R(S)型	S 型構造の反復により構成される構造。シーケンシャルな並びとして出現する要素を指し示す Xpath(PCDATA を内容として含む末端要素を指し示す Xpath)のリストにより特徴づけられる。
R(R)型	2 重の反復要素の入れ子により構成される構造。一段目の反復要素を指し示す Xpath(外側特徴 Xpath)と、二段目の反復要素を指し示す Xpath(内側特徴 Xpath)により特徴づけられる。

3.1.2 基本テーブル写像

基本テーブル写像は、XML 断片型の特徴 XPath リストが、表組みスタイルの値セルにどのように対応するかによって、以下の 10 種類に類別できる。

1) S 型断片の特徴 Xpath リストが指し示す XML 要素内容のリストを行の順に対応させる写像(N 行 M 列の表のセルに対して、1 行 1 列,1 行 2 列,…,1 行 M 列,2 行 1 列,…,2 行 M 列,…,N 行 M 列の順に対応させる)を、**S 型テーブル写像 T[S]**と呼ぶ。像は固定表組みである。図 3 に例を示す。



S 型写像の原像の例

特徴 Xpath リスト
= [a1/b1/c1, a1/b1/c2,
a1/b2, a2, a3, a4]

1	2	3
4	5	6

S 型写像の像の例
(2 行 3 列の固定表組み)

図 3 S 型テーブル写像の例

また、逆に、S 型断片の特徴 Xpath リストが指し示す XML 要素内容のリストを列の順に対応させる写像(N 行 M 列の表のセルに対して、1 行 1 列,2 行 1 列,…,N 行 1 列,1 行 2 列,…,N 行 2 列,…,N 行 M 列の順に対応させる)を、**S^t 型テーブル写像 T[S^t]**と呼ぶ。像は、固定表組みである。

以下写像の名称において、t を上付き添字としているのは、XML 断片における要素の出現順序がテーブルの行の順に対応している場合を正の写像とみなし、列の順に対応しているものは正の写像の転置とみなしていることによる。

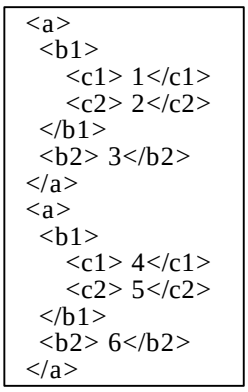
2) R 型断片の特徴 Xpath が指し示す XML 要素内容のリストを行の順に対応させる写像を、**R 型テーブル写像 T[R]**と呼ぶ。像は、行可変、または列可変の表組みである。また、逆に、R 型断片の特徴 Xpath が指し示す XML 要素内容のリストを列の順に対応させる写像を、**R^t 型テーブル写像 T[R^t]**と呼ぶ。像は、行可変、または列可変の表組みである。

3) S(R)型断片の特徴 Xpath リストの各々の Xpath が指し示す XML 要素内容のリストを行に対応させる写像を、**S(R)型テーブル写像 T[S(R)]**と呼ぶ。像は、固定表組みである。また、逆に、S(R)型断片の特徴 Xpath リストの各々の Xpah が指し示す XML 要素内容のリストを列に対応させる写像を、**S(R)^t 型テーブル写像 T[S(R)^t]**と呼ぶ。像は、固定表組みである。

4) R(S)型断片の特徴 Xpath リストの各々の Xpath が指し示す XML 要素内容のリストを列に対応させる写像を、**R(S)型テーブル写像 T[R(S)]**と呼ぶ(S(R)と R(S)では行と列の対応関係が逆になる)。像は、行可変表組みである。

また、逆に、R(S)型断片の特徴 Xpath リストの各々の Xpath が指し示す XML 要素内容のリストを行に対応させる写像を、**R(S)^t 型テーブル写像 T[R(S)^t]**と呼ぶ。像は、列可変表組みである。

5) R(R)型断片の外側特徴 Xpath が指し示す XML 要素リストを行に対応させ、内側特徴 Xpath が指し示す XML 要素リストの値を列に対応させる写像を、**R(R)型テーブル写像 T[R(R)]**と呼ぶ。像は、行列可変表組みである。また、R(R)型断片の外側特徴 Xpath が指し示す XML 要素リストを列に対応させ、内側特徴 Xpath が指し示す XML 要素リストの値を行に対応させる写像を、**R(R)^t 型テーブル写像 T[R(R)^t]**と呼ぶ。像は行列可変表組みである。



R(S)型写像の原像の例

特徴 Xpath リスト
= [a/b1/c1, a/b1/c2]
反復 Xpath
={ a }

1	2	3
4	5	6

R(S)型写像の像の例
(行可変表組み)

図 4 R(S)テーブル写像の例

3.1.3 連結テーブル写像

原像となる XML 断片の反復数が等しい基本テーブル写像 T1 と T2 は、反復要素である特徴 Xpath が指し示す XML 要素の像にあたる表組みスタイルの行または列を共通断面として連携することにより得られる表組みスタイルを像とするテーブル写像を構成することができる(**連結テーブル写像 T[T1・T2]**)

3.2 階層表組み写像

図 5に示すような表組みレイアウトを末端の構造としてもち、末端の表組みをリスト配置や入れ子見出し(ネスト見出し)などにより組み合わせ構成していく構造をもつ**階層表組み画面**と呼ぶ。

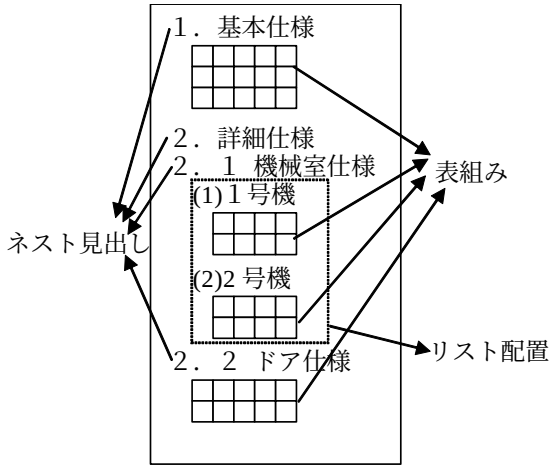


図 5 階層表組み画面

階層表組み写像 H は、XML 断片から階層表組み画面への写像であり、以下のように帰納的に定義される。

1) テーブル写像 T は、階層表組み写像 H である。

2) H1,・・・,Hn が階層表組み写像であるとき、各々の写像の原像 IIm(H1), ・・・, IIm(Hn)を連結することにより得られた XML 断片を、各々の写像の像 Im(H1) ・・・, Im(Hn)を見出し付で順に並べた像に写像する**ネスト連結写像 N(H1,....,Hn)**は、階層表組み写像である。また、逆像を連結する際には、親タグを追加してもよい。

3)　Hが階層表組み写像であるとき、逆像 IIm(H)が反復出現する XML 断片を、各々の写像の像 Im(H)を見出し付で順に並べた像に写像する**リスト連結写像 L(H)**は、階層表組み写像である。

4 XML 文書入力画面自動生成方式の実装

本章では、4.1 節で XML 文書入力画面自動生成方式の全体構成について述べ、4.2 節でその中核モジュールである XML 文書設計エディタ XDDS-E(XML Document Desigin Support Editor)について述べる。

4.1 XML 文書入力画面自動生成方式の全体構成

XML 文書入力画面自動生成方式は、図 6に示すように、XML 文書設計支援エディタ XDDS-E、XSLT 自動生成モジュール XDDS-G、Web サーバ上のモジュール XDDS-S、および Web クライアント上のモジュール XDDS-C からなる。以下では、システム開発者による XML 入力画面作成から、エンドユーザによる XML 文書作成にいたる作業フローについて説明する(詳細は参考文献[4][5]に譲る)。

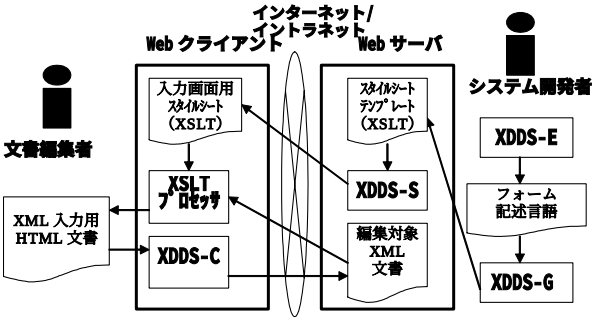


図 6　XDDS を用いた XML 文書作成

(1)XDDS-E と XDDS-G による XML 文書設計情報作成

システム開発者は、XDDS-E を用いて、文書スキーマ、文書制約、文書スタイル、および、ワークフロー文書活用情報などの XML 文書入力画面に必要な設計情報を指定することにより、フォーム記述を出力する。そして、XDDS-G は、フォーム記述を入力として、スタイルシートテンプレート(XSLT 記述)を変換出力する。スタイルシートテンプレート中の HTML によるスタイル記述は、階層表組み写像を用いて自動生成される。また、入力チェック等の文書内容制約は、テンプレートを用いて JavaScript に自動変換される。

(2)　Web サーバ側の入力フォーム生成処理

XDDS-S は、Web の前画面から、ワークフロー文脈情報として、文書編集者の担当分野、アクション、編集対象 XML 文書名、および製品 ID や担当者名などの次画面に受け渡すべき入力データを受け取る。そして、このワークフロー文脈情報とワークフロー活用制御ファイルから今回使用する XML 入力画面用スタイルシート(XSLT 記述)と編集対象 XML 文書を選定し、Web クライアント側に送る。また、XDDS-S は、前画面入力データや前版との差分表示用情報などのワークフロー活用のための制御情報を XDDS-C に送付する。

(3) Web ｸﾗｲｱﾝﾄ側の入力ﾌｫｰﾑ生成処理と文書編集

XDDS-C は、編集対象の XML 文書と XML 入力画面用スタイルシートを入力として、HTML 形式の入力フォームを出力し、Web ブラウザが入力フォームを画面上に表示する。ここで、XML 文書中の記載項目の内容に応じた記載項目の動的変更や、入力チェックなどは、XDDS-G が自動生成する JavaScript によって実現されている。

また、Web ブラウザ上の入力フォームの編集結果は、XDDS-C を通じて編集対象 XML 文書に反映され、編集が終了すると、Web サーバ側に返送され、Web サーバの XDDS-S によって、XML 文書ファイルとして、Web サーバ上に保存される。

4.2 XML 文書設計支援エディタ XDDS-E

XML 文書設計支援エディタ XDDS-E は、図 7に示すようなテーブル形式の GUI をもち、行毎に、XML 要素に対して、文書スキーマ、文書内容制約、文書スタイル、およびワークフロー文書活用などの XML 文書設計情報を設定することができる。

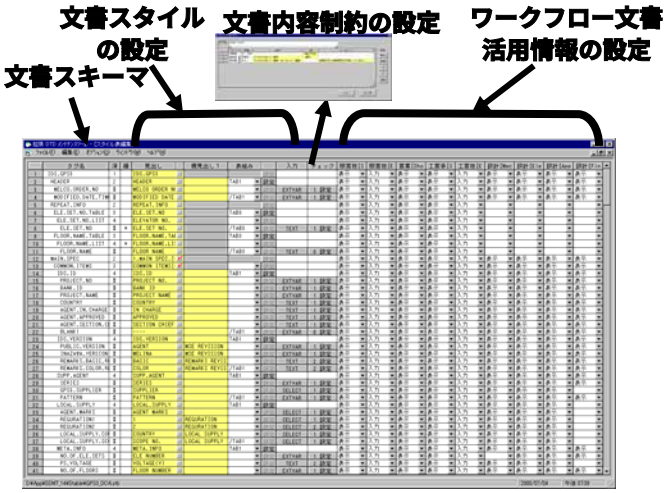


図 7　XML 文書設計支援エディタ

以下、作成される個々の画面設計情報を順に説明する。

(1)文書スキーマ記述部

DTD や XMLSchema で記述できる XML 文書の構造や内容の制約を記述する部分である。

(2)文書内容制約記述部

DTD や XMLSchema で記述できない 2.1 節(1)で述べた「内容・構造間制約」、「内容間制約」、および「文書外制約」を記述する部分である。

また、入力画面からのチェックルール呼び出し用制御情報を記述するために、文書制約毎に、「入力チェックの起動条件」、「起動タイミング(onload,onfocus,onchange など)」、「警告メッセージ内容」、および「警告メッセージの表示先」を記述できるようにしている。

(3)文書スタイル記述部

XML 文書のタグ毎に、入力コントロールの種別(テキスト、ポップアップ、チェックリスト、ファイル添付等)、テーブル変換用パターン(3 章の「基本テーブル写像」の型に対応)、見出し指定、セル間スペースや枠線太さなどのスタイル情報を設定する。

(4)**ワークフロー文書活用記述部**

(i)**担当分野毎・アクション毎の画面提示**

XML 文書のタグ毎に、担当分野とアクションに応じて、そのタグの内容が「編集可」、「表示のみ」、「非表示」であることを指定することができる。

(ii)**文書データの業務システム連携**

XML 文書のタグ毎に、業務システムの入力フォーマットの対応する項目名を指定することができる。本記述をもとに、XML 文書を CSV(Comma Separated Values)ファイルに変換したり、XML 文書をパンチデータに変換することができる。

5 XML 入力画面自動生成方式の評価

本章では、5.1 節で、設計分野の文書処理アプリケーションへの適用を通じて、４章で述べた XML 入力画面自動生成方式を定性評価した結果を述べる。また、5.2 節では、論理中心画面設計に基づく XML 入力画面自動生成方式(以下、**論理中心ツール**)と、表示中心画面設計の市販ツール (以下、**表示中心ツール**と呼ぶ)との開発工数の定量比較結果について述べる。

5.1 文書処理アプリケーション適用による定性評価

本節では、提案する XML 入力画面自動生成方式を実業務に適用することにより確認した文書処理機能を、2.1 節で述べた要求に対応させて説明する。

(1) XML 文書の構造や内容の制約の複雑さへの対応

図 1 に示した構造をもつ設計仕様書を対象として、昇降機のかごの数やビルの階数に応じて、基本仕様のかごサイズ、機械仕様の停止階、電気仕様の非常用昇降機、および意匠仕様のドアタイプなどの仕様項目の数を動的に変える機能を実現できることを確認した。

・入力項目異なり数 298(反復出現要素を含めると約 1000)に対して 884 の内容検証規則をもつ設計仕様書の入力画面を作成することができた。

・担当者名や製品番号などの Web ブラウザでの他セッションの入力データを編集対象に取り込む機能により、ワークフローシステムのデータ管理機能との連携を実現できた。

(2) 画面 IF の複雑さへの対応

・約 1000 個の入力項目を 37 の表として分類表示する本文フレーム、該当する表へのハイパーリンクをもつ見出しからなる目次フレーム、本文における入力項目が選択項目に含まれない場合の特記仕様を記載するための特記フレーム、内容検証結果のエラーメッセージをエラー箇所へのハイパーリンクと共に提示するメッセージフレームからなる設計仕様書の入力画面を生成することができた。

・前版と対応するタグの内容を比較して、異なる部分は色を変えて表示する差分表示機能を実現した。

(3) ワークフローでの文書データ活用の複雑さへの対応

・営業所の営業設計部門の担当者と管理者、工場の営業設計書の管理者と担当者、製作所の工事設計の機械設計、電気設計、意匠設計、色設計の８担当業務毎に異なる入力画面と表示画面を作成することができた。

・ワークフローの対象業務である営業設計の後工程である工事設計支援システムとのデータ連携を、XML から CSV への変換プログラムにより実現できた。

5.2 XML 入力画面の開発工数の定量比較

表示中心ツールと、論理中心ツールとの XML 入力画面の開発工数の比較実験を実施することにより、約 1/2 の工数削減をはかれることを確認した。以下では、評価方法と評価結果について述べる。

5.2.1 評価方法

(1) 対象文書

2.1 節で分析対象とした昇降機設計仕様書に加えて、提案方式の一般性を確認するために、WebEDI の資材伝票を対象とした。

昇降機の設計仕様書の異なり項目数(反復出現する要素は 1 つと数えた)は 298 であり、チェックルール数は 884 である。画面中のテーブル数は 37 であり、内訳は S 型が 20、SR 型が 11、RS 型が 2、RR 型が 4 である。また、業務毎画面としては、すべての項目を含む営業設計向けをベースとして、営業、機械設計、電気設計、意匠設計、色設計の 5 つの業務毎画面を作成した。

WebEDI の資材伝票は、見積依頼書、発注書、納期回答書など 19 種を対象とした。19 種合計で、異なり総項目数は 490(平均 25.8)であり、チェックルール数は 204(平均10.7)である。画面中のテーブル数は91(平均4.8)であり、内訳は S 型が 56(平均 2.9)、RS 型が 35(平均 1.8)である。

(2) 時間計測の作業区分

XML 文書入力画面作成でおいて共通作業であり、かつアプリケーションの業務分析作業との分離が難しい「XML Schema 設計」と「画面レイアウト作成」自身は省いて、画面レイアウト設定作業、チェックルール作成作業、および業務毎画面カスタマイズ作業(昇降機設計仕様書のみ)の時間を測定した。

5.2.2 評価結果

以下では、昇降機設計仕様書と WebEDI 資材伝票の XML 入力画面作成時間の比較実験結果について順に述べる。

(1) 昇降機設計仕様書の XML 入力画面の作成時間

昇降機の設計仕様書の XML 入力画面作成時間を、表示中心ツールと論理中心ツールとで比較した結果を表 2 に示す。論理中心ツールは表示中心ツールと比較して、作業工数を約 1/2(52.7%)削減できた。

表 2　昇降機設計仕様書の作成時間の比較

	表示中心	論理中心	削減率
画面ﾚｲｱｳﾄ作成	7:58[時：分]	3:35[時：分]	54.9%
ﾁｪｯｸﾙｰﾙ作成	14:23[時：分]	7:35[時：分]	47.3%
業務毎ｶｽﾀﾏｲｽﾞ	2:34[時：分]	0:36[時：分]	76.6%
計	24:54[時：分]	11:46[時：分]	52.7%

また、業務毎画面カスタマイズ時には、すべての項目を含む営業設計向け画面から要素を追加/削除することにより業務毎画面を作成したが、要素の追加/削除に要する平均的な作業時間は、表 3に示す通りであった。論理中心ツールは表示中心ツールに対して、要素追加時間は約 2/3 削減(66.7%)で、要素削除時間は約 1/2 削減(49.5%)できた。

表 3　要素の追加削除に要する平均時間の比較

	表示中心	論理中心	削減率
要素追加	01:31[分：秒]	00:46[分：秒]	49.5%
要素削除	00:36[分：秒]	00:12[分：秒]	66.7%

(2) WebEDI の資材伝票の XML 入力画面の作成時間

WebEDI の資材伝票の XML 入力画面の作成時間(19 種類の合計)を、表示中心ツールと論理中心ツールとで比較した結果を表 3に示す。論理中心ツールは表示中心ツールと比較して、作業工数を約 1/2(55.4%)削減できた。

	表示中心	論理中心	削減率
画面ﾚｳｱｲﾄ作成	9:01[時：分]	3:07[時：分]	65.4%
ﾁｪｸﾙｰﾙ作成	3:41[時：分]	2:33[時：分]	30.8%
計	12:42[時：分]	5:40[時：分]	55.4%

6 考察

本章では、５章で述べた開発効率向上の原因について考察した後、今後の課題として表実中心画面設計と論理中心画面設計の融合の可能性について述べる。

(1) 開発効率向上の原因

画面レイアウト作成、チェックルール作成、および業務毎カスタマイズ毎に、開発効率向上の原因を考察する。

・画面レイアウト作成については、設計仕様書の場合で 54.9%、資材帳票の場合で 65.4%の削減効果であるが、この効果は主として、表パターン指定からの表組みスタイル自動生成機能による。

・チェックルール作成については、設計仕様書の場合で 47.3%、資材帳票の場合で 30.8%の削減効果であるが、この効果は、評価用に用いた表示中心ツールでは複雑なスクリプト作成を必要としたルールを、XDDS-E が提供する Xpath 風の簡易チェックルール記述言語を用いてより簡単に記述できたことによる。

・業務毎カスタマイズにについては、設計仕様書の場合で 76.6%の削減効果であるが、この効果は、表示中心ツールでは業務毎の画面の入力枠を直接手動で追加削除する操作が必要になるのに対して、XDDS-E では、業務毎に異なる要素を選択するだけでよいことによる。この効果は、データのアクセス権は論理中心設計の方がわかりやすい点と、表計算ソフトゥエアの操作が人間にとって直感的にわかりやすいことによる。

(2)表示画面中心設計と論理画面中心設計の融合について

本論文では、論旨を単純化するために、表示中心設計と論理中心設計を対比させて述べてきたが、理想的な XML 入力画面作成支援ツールでは、表示中心画面設計が特長とする自由かつ簡単なレイアウト表現能力と、論理中心画面設計がもつ可変構造に関する制約表現能力とを調和させながら融合することが重要になると著者は考える。この融合における困難さは、2.2 節で述べた表示・論理対応付けの困難さが本質的な問題なので、木・表構造間写像モデルは、この融合のための道具の一つとして役立つ。具体的な方向性の一つは、表示中心ツールにおいて、表組みと木構造との対応関係を内蔵したテーブルオブジェクトを導入することである。ただし、融合する形態は、ツールの対象アプリケーションと対象ユーザによって様々な形態がありうる。本稿で提案した XDDS-E ツールは、論理中心画面設計に基づいているが、XML 文書構造やその上の制約が十分複雑で要素の追加削除のカスタマイズを簡単にしたいが、画面は定型的なものでよいというアプリケーションで特に有効である。

7 おわりに

XML 文書がもつ木構造から HTML 文書がもつ表構造との写像を類別する木・表構造間写像モデルを用いることにより、XML 要素に対して表パターンを指定するだけで階層表組み画面を自動生成できる点を特長とする XML 文書入力画面作成支援方式を提案した。そして、エンジニアリング用の設計仕様書と Web-EDI 用の資材伝票を対象とする評価により、「従来の表示中心画面設計では困難であった可変 XML 文書を扱うことができること」、また「従来の表示中心画面設計による XML 入力画面作成支援ツールと比較して、入力画面開発の工数を約 1/2 削減できること」を確認した。

【参考文献】

[1]Adobe 社：FormDesigner, http://www.adobe.co.jp/products/server/formdesigner/
[2]Microsoft 社：InfoPath, http://www.microsoft.com/japan/office/infopath/
[3]Altova 社：XML Spy, http://www.xmlspy.jp/
[4] 今村 誠, 森口 修, 鈴木 克志, 辻 秀一:WWW ブラウザによる XML 文書入力方式について、情報処理学会デジタルドキュメント研究会資料 17-1、pp1-8(1999)
[5]今村 誠,森口 修,鈴木 克志: XML 文書ワークフロー構築方式, 情報処理学会 デジタルドキュメント研究会資 27-1, pp1-8 (2001)