推薦論文

木・表構造間写像モデルに基づく XML 入力画面自動生成方式

今村 誠[†] 森口 修[††] 大場雅之[†††]
鈴木克志[†] 依田文夫[†]

インターネットの普及と電子商取引の進展にともなって，Web ブラウザから XML 文書を入力参照するシステムの開発がさかんになった．しかし，画面中の入力枠に対して XML 要素を対応付ける従来の表示中心画面設計では，対象 XML 文書の論理構造が複雑になるにつれて，XML を HTML に変換するスタイルシートの開発が難しくなるという問題点があった．特に，データ項目の内容に応じて表の行や列の反復数が変わる可変 XML 文書を扱うことが困難であった．本論文では，この問題を解決するために，テーブル形式のユーザインタフェースを用いて，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，および担当業務ごとのアクセス権限情報などを付与するだけで，XML 入力用スタイルシートを生成する XML 入力画面自動生成方式を提案する．本方式の特長は，XML 文書が持つ木構造から HTML の表組みへの写像を類型化する木・表構造間写像モデルにより，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，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，本方式は,「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基づくツールと比較して，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．

An XML Input Form Generation Method Based on a Tree-table Mapping Model

Makoto Imamura,[†] Osamu Moriguchi,[††] Masayuki Ooba,[†††]
Katsushi Suzuki[†] and Fumio Yoda[†]

Many web browsers based XML document processing systems have been developed with the spread of EC. But existing view centered input-form design methods, corresponding input fields in the form to XML elements, are difficult to treat documents which have complicated logical structure. For example, they are difficult to treat variable structure, 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, which can generate a style-sheet for XML input forms by specifying style, check rules and access authorization rules with spread-sheet interface. The feature of the method is a tree-table mapping model, which enables XSLT stylesheet generation by only specifying table patterns to XML elements. Evaluation of our method for engineering design specifications and Web-EDI input forms shows the following results. (1) It can treat variable structure, which is difficult to treat by view centered input-form design methods. (2) It can decrease 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電子商取引の進展にともなって，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されるようになった．

† 三菱電機株式会社情報技術総合研究所
MITSUBISHI ELECTRIC CORPORATION, Information Technology R&D Center
†† 三菱電機インフォメーションシステムズ株式会社ビジネスソリューション事業本部
MITSUBISHI ELECTRIC INFORMATION SYSTEMS CORPORATION
††† 三菱電機株式会社ビルシステム業務統括部
MITSUBISHI ELECTRIC CORPORATION, Building Systems Group, Planning & Administration Div.

本論文の内容は 2004 年 1 月のデジタル・ドキュメント研究会にて報告され，DD 研究会主査により情報処理学会論文誌への掲載が推薦された論文である．

しかし，対象とする XML 文書の論理構造が複雑になるにつれて，お絵かきツール風の GUI（Graphical User Interface）を用いて作成した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，XML を入力画面用 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反復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．

本論文では，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，および担当業務ごと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 入力画面自動生成方式を提案する．本方式の技術特長は，XML 文書が持つ木構造を HTML の表組みレイアウトへと変換する写像を類型化する木・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．

本論文の構成は，以下のとおりである．2 章では，本論文で扱う技術課題について述べる．3 章では，本論文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 入力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（Electronic Data Interchange）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

本章では，本論文で扱う技術課題について述べる．2.1 節では，エンジニアリング分野における設計支援を例として，設計仕様書に対する XML 文書入力画面開発における要求を分析する．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 入力画面生成方式の技術課題を整理する．

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向けの文書処理では，設計対象物の構成や設計業務の内容を反映するため，設計仕様書やデータシートなどを対象とする文書処理への要求機能がオフィス分野と比べて複雑になる．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．

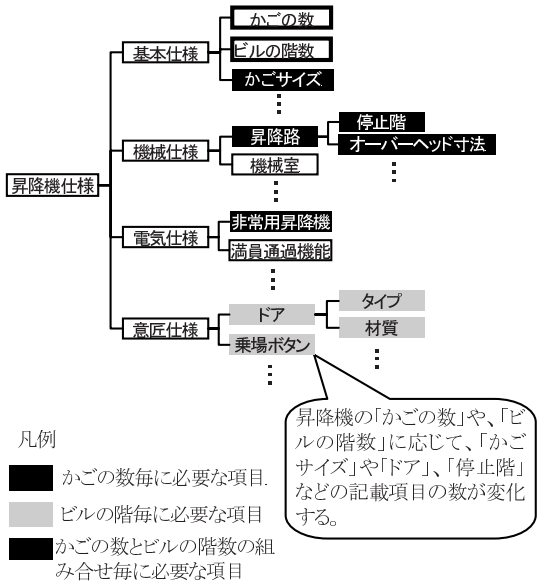


図 1 設計仕様書の XML 構造の例
Fig. 1 An XML structure of design specification.

(1) **構造・内容制約の複雑さ（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持つだけでなく，「ある記載項目の内容と，ある記載項目の存在有無や反復数との関係制約（**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**）」を持つという特徴がある．**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，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持ち，たとえば以下の制約を持つ．

(a) 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．本論文では，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 文書構造を**可変 XML 構造**と呼ぶ．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．

(b) 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，センサがないと取り付けられない」など．

(c) 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」など．

(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持つ制約が複雑になるに従って，見やすさや入力チェック制御のための画面 IF への要求も複雑になる．

(a) **見やすさへの要求（階層構造の表による表示）**

文書が持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分かりやすく画面に提示したりする必要がある．ここでの主な問題は，「階層構造の表による表示」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，「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と変換するルールを指定する」という問題であるが，この指定は必ずしも容易でない．

(b) **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，「担当分野ごと・アクションごとの画面提示」や「文書データの業務システム連携」の要求がある．

(a) **担当分野ごと・アクションごとの画面提示**

ワークフローシステム中の文書画面では，ユーザが担当者か管理者かなどの権限や，機械設計者か電気設計者かなどの担当分野に応じて，参照項目や編集項目が異なる場合がある．また，表示用と入力用とで画面は異なる．さらに，人手の内容チェックを容易にするために，前版の文書との差分を分かりやすく表示する機能が要求される場合がある．

(b) **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

本論文の目的は，入力画面の要求仕様から XML 文書入力用 HTML を自動生成することである．この目的を達成するには，入力画面の要求仕様から，XML を HTML に変換するスタイルシートを自動生成できればよい．このスタイルシート自動生成の問題は，XML 文書変換と入力画面生成という 2 つの側面を持つ．以下，順に各々の従来技術について説明する．

2.2.1 XML 文書変換の従来技術

スタイルシートを記述する XML 変換言語としては XSLT[1]（XSL Transformations）が広く普及しており，また，文献 2) などで形式的にも十分な変換能力があることも知られている．しかし，XSLT は言語仕様が複雑なため，コーディングに特別なスキルが必要なので，XSLT をより簡単に作成するための高級 XML 文書変換言語として，SynTree[3] や XTL（XML Transformation Language）[4] などが提案されるようになった．SynTree は森正規文法に従う言語間の変換モデルを提案しており，また，XTL は入力 XML 構造と出力 XML 構造の対からなる変換規則の記述方式を提案している．いずれも，XSLT と比較してより直感的なプログラミングを可能にする基盤を与えているが，2.1 節の要求の難しさの本質である「XML が持つ木構造のデータ構造」と「行や列が可変になる表形式の画面レイアウト構造」との対応関係を簡単に指定する方式は与えていない．

2.2.2 XML 入力画面生成の従来技術

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**と呼ぶ）という工程が存在する点にある．画面自体は XML インスタンスに直接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相当するので，2.1 節で述べた可変 XML 画面の作成では，表示・論理対応付けに工夫が必要になるという点が問題の本質である．より具体的には，HTML 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 はより高い文書構造記述能力を持つので，両者の記述能力のギャップを埋める必要があるということである．また，XML 文書設計では，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 画面の作成では，両者を簡単に分離できないという問題も付随している．

本論文では，既存の XML 入力画面の開発方法を，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．

(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[5] や Micorosoft 社の InfoPath[6] 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 要素を対応付けていくことにより入力画面を作成していく．FromDesigner は，紙帳票のように A4 1 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持つ固定帳票画面を作成する場合に適しているが，2.1 節で述べたような文書構造自体が複雑な場合には適していない．また，InfoPath は，テーブルが表の行の反復にしか対応していないことや，スクリプト記述において DOM[7]（Document Object Model）の API レベルの記述が必要などの問題点がある．以下，2.1 節の要求ごとに，表示中心画面設計の問題点について述べる．

(a) **文書制約設計での問題**

表示中心画面設計は，「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」という問題がある．

(b) **文書スタイル設計での問題**

表示中心画面設計は，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．また，ツールが持つ表示・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持つ画面と，与えられた文書論理構造とを対応付けられない場合がある．

(c) **ワークフロー文書活用設計での問題**

表示中心画面設計は，担当分野ごと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，担当分野ごとの画面の数だけ修正作業が必要になり，担当分野ごと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) **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[8] がある．論理中心画面設計では，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付けていく．表示中心画面設計が得意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 本論文で扱う課題

本論文の目的は，2.1 節で述べたアプリケーションの要請に応え，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．従来の問題点を解決するための提案方式の工夫は，以下の 2 点に集約される．

(1) **表・木構造間写像モデルによる表組みレイアウト自動生成**

表示中心画面設計では扱いが困難であった可変 XML 構造に対応するために，表・構造間写像モデルを提案し，2 章で述べた表示・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 要素ごとに，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 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，XML 文書が持つ木構造を HTML の表組みレイアウトへと変換する写像を類型化したものである．本モデルは，XML 文書構造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」と，XML 文書構造から「表組みをリストや入

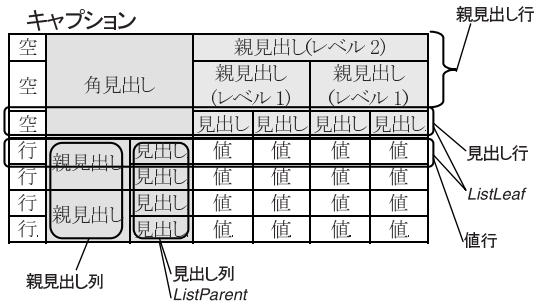


図 2 表組みスタイル
Fig. 2 Tabular style.

れ子に配置した階層表組み画面」への写像を表現する「階層テーブル写像」から構成される．以下，順に説明する．

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り得られる連結テーブル写像とがある．以下，順に説明する．

3.1.1 基本テーブル写像

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．以下では，まず，テーブル写像の像をモデル化する表組みスタイルと，テーブル写像の原像をモデル化する XML 断片の型を規定する．続いて，XML 断片から表組みスタイルへの写像として，基本テーブル写像を定義する．

(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．値部分は行と列の構造を持っており，XML の要素内容に対応する部分になる．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，階層構造を持つことができる．

(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），S(R) 型，R(S) 型，および R(R) 型に類別できる．表組みレイアウトの行または列が，R(S) 型となるような R(R(S)) 型や S(R(S)) 型などより複雑な XML 断片も定義できるが，本論文では，簡明さのため省略する．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（**内 XPath** と呼ぶ）と，内 XPath の上位要素で行または列に対応する要素を示す**外 XPath** とにより特徴付けられる（総称して，**特徴 XPath** と呼ぶ）．また，内 XPath は，末端の XML 要素が並びと反復のいずれで出現するかによって，**並び内 XPath** と**反復内 XPath** とに分類される．外 XPath も同様に，**並び外 XPath** と**反復外 XPath** とに分類される．

表 1 基本 XML 写像の原像となる XML 断片の型
Table 1 XML fragment types as inverse mages of basic table mappings.

型	説明
S 型	表組み中の値が要素の並びにより構成される断片．並び内 XPath により特徴付けられる．
R 型	表組み中の値が要素の反復により構成される断片．反復内 XPath により特徴付けられる．
R(S) 型	S 型断片の反復により構成される断片．並び内 XPath と反復外 XPath により特徴付けられる．
S(R) 型	R 型断片の並びにより構成される断片．反復内 XPath と並び外 XPath により特徴付けられる．
R(R) 型	2 重の反復要素の入れ子により構成される断片．反復内 XPath と反復外 XPath により特徴付けられる．

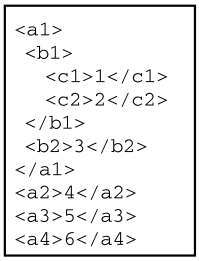
(3) **基本テーブル写像**

基本テーブル写像は，特徴 XPath 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 5 種類に類別できる（転置型を含めると 10 種類）．

(a) S 型断片の並び内 XPath が指し示す XML 要素内容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，S 型テーブル写像 T[S] の原像，特徴 XPath，および像の例を示す．また，逆に，S 型断片の並び内 XPath が指し示す XML 要素内容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]**（転置型）と呼ぶ．

(b) R 型断片の反復内 XPath が指し示す XML 要素内容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．**図 4** に，R 型テーブル写像 T[R] の原像，特徴 XPath，および像の例を示す．また，逆に，R 型断片の反復内 XPath が指し示す XML 要素内容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]**（転置型）と呼ぶ．像は，行可変，または列可変の表組みである．

(c) R(S) 型断片の並び内 XPath を列に対応させ，反復外 XPath を行に対応させる写像を，**R(S)**



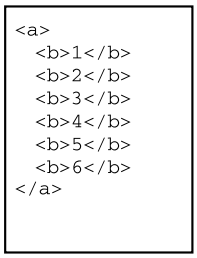
(a)T[S]型写像の原像の例

(b) (a)の並び内 XPath
= {a1/b1/c1, a1/b1/c2, a1/b2, a2, a3, a4}

1	2	3
4	5	6

(c) T[S]型写像の像の例
(2 行 3 列の固定表組)

図 3 S 型テーブル写像の例
Fig. 3 An example of S-type table mapping.



(a)T[R]型写像の原像の例

(b) 反復内 XPath
={a/b}

1	2	3
4	5	6

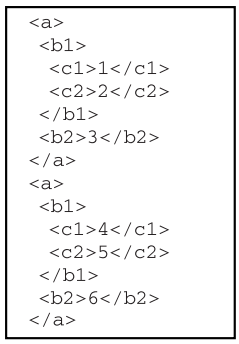
(c) R 型写像の像の例
(3 列の行可変表組)

図 4 R 型テーブル写像の例
Fig. 4 An example of R-type table mapping.

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．**図 5** に，R(S) 型テーブル写像 T[R(S)] の原像，特徴 XPath，および像の例を示す．また，逆に，行と列への対応関係を逆にした写像を，$\mathbf{R(S)}^t$ **型テーブル写像 T[R(S)^t]**（転置型）と呼ぶ．像は，列可変表組みである．

(d) S(R) 型断片の反復内 XPath を列に対応させ，並び外 XPath 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，列可変表組みである．**図 6** (b3) に，S(R) 型テーブル写像 T[S(R)] の原像，特徴 XPath，および像の例を示す．また，逆に，行と列への対応関係を逆にした写像を，$\mathbf{S(R)}^t$ **型テーブル写像 T[S(R)^t]**（転置型）と呼ぶ．像は，行可変表組みである．

(e) R(R) 型断片の反復内 XPath を列に対応させ，反復外 XPath を行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．図 6 (b1) に，R(R) 型テーブル写像 T[R(R)] の原像，特徴 XPath，および像の例を示す．また，逆に，行と列への対応関係を逆にした写像を，$\mathbf{R(R)}^t$ **型テーブル写像 T[R(R)^t]**（転置型）と呼ぶ．像は行列可変表組みである．



(a)T[R(S)]型写像の原像の例

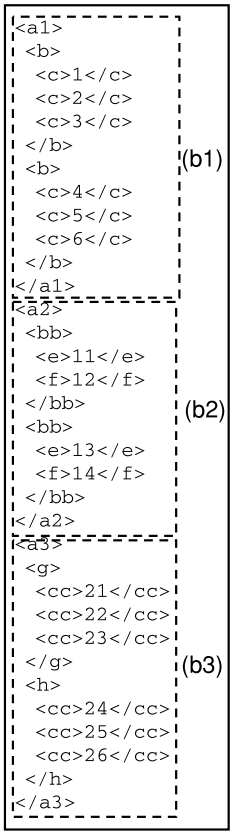
(b) (a)の並び内 XPath
= {a/b1/c1, a/b1/c2, a/b2}

(a)の反復外 XPath
= { a }

1	2	3
4	5	6

(c) T[R(S)]型写像の像の例
(行可変表組み)

図 5 R(S) テーブル写像の例
Fig. 5 An example of R(S)-type table mapping.



(a)連結テーブル写像の原像の例

(b1) T[R(R)]型写像の原像の
反復内 XPath = {a1/b/c }
反復外 XPath = {a1/b}.

(b2) T[R(S)]型写像の原像の
並び内 XPath
={a2/bb/e, a2/bb/f}
反復外 XPath .
={a2/bb}.

(b3) T[S(R)型]写像の原像の
反復内 XPath
= {a3/g/cc, a3/h/cc}
並び外 Xpath
={a3/g,. a3/h}
連結制約
count(a3/g/cc)
= count(a3/h/cc)

(c1) (b1)と(b2)の連結制約
count(a1/b)
= count(a2/bb)

(c2) (b1)と(b3)の連結制約
count(a1/b/c)
= count(a3/g/cc)

1	2	3	11	12
4	5	6	13	14
21	22	23		
24	25	26		

(d)連結テーブル写像の像の例

図 6 連結テーブル写像の例
Fig. 6 An example of connected-table mapping.

表組みである．

3.1.2 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 を，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

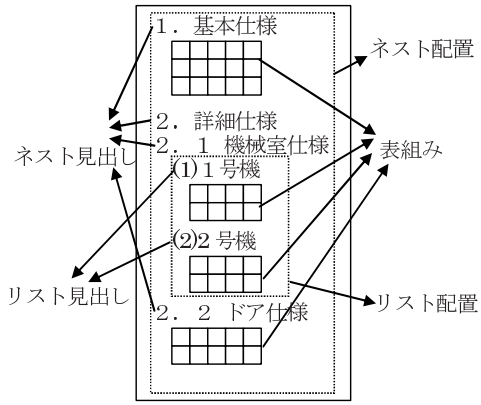


図 7 階層表組み画面
Fig. 7 A nested-table form.

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写像を**連結テーブル写像 T[T1・T2]** と呼ぶ．図 6 に，R(R) 型，R(S) 型，S(R) 型の基本テーブル写像を連結させた連結テーブル写像の例を示す．

図の (a) 中の (b1) で指し示される R(R) 型断片と，(a) 中の (b2) で指し示される R(S) 型断片が表組みとして連結するためには，(c1) に示すように,「a1/b が指し示す要素の数」と「a2/bb が指し示す要素の数」が等しいという制約を満たす必要がある（**連結制約**と呼ぶ）．同様に,「(a) 中の (b1) の R(R) 型断片」と「(a) 中の (b3) の S(R) 型断片」が表組みで連結するためには，(c2) の連結制約を満たす必要がある．

3.2 階層テーブル写像

図 7 に示すように，表組みレイアウトを末端構造として，リストと入れ子（ネスト見出しを含む）により構成される画面を**階層表組み画面**と呼ぶ．

階層テーブル写像 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（基本テーブル写像，または連結テーブル写像）T は，階層テーブル写像 H である．

(2) H1, ···, Hn が階層テーブル写像であるとき,「各々の写像の原像 IIm(H1), ···, IIm(Hn) を結合することにより得られる XML 断片」から「各々の写像の像 Im(H1) ···, Im(Hn) を見出し付きで順に並べた画面」への写像は，階層テーブル写像である．本写像を**ネスト写像 N(H1, ···, Hn)** と呼ぶ．また，原像を連結する際には，親タグを追加してもよい．

(3) H が階層テーブル写像であるとき,「原像 IIm(H) の反復を結合して得られる XML 断片」を「各々の写像の像 Im(H) を見出し付きで順に並べた画面」への写像は，階層テーブル写像である．本写像を**リスト写像 L(H)** と呼ぶ．

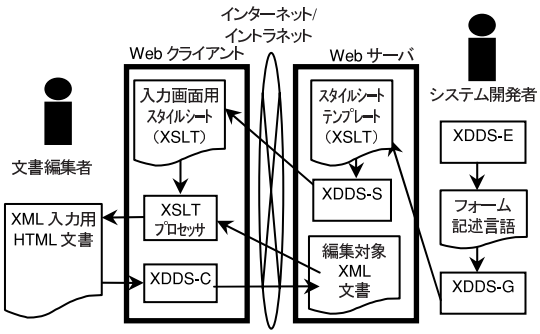


図 8 XDDS を用いた XML 文書作成
Fig. 8 Document processing with XDDS.

4. XML 入力画面自動生成方式の実装

本章では，4.1 節で XML 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 入力画面自動生成方式の全体構成

XML 入力画面自動生成方式は，**図 8** 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を説明する（詳細は参考文献 9) に譲る）．

(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 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（独自言語）を出力する．次に，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，入れ子レイアウト写像を用いて自動生成される．また，入力チェックなどの文書内容制約は，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．次に，このワークフロー文脈情報とワークフロー活用制御

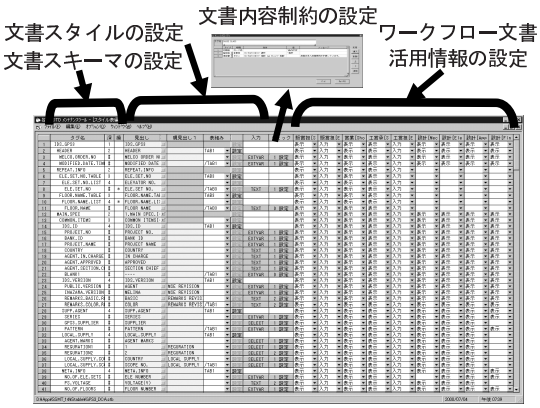


図 9 XML 文書設計支援エディタ
Fig. 9 XML document design support editor.

ファイルを用いて，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 XDDS-C が利用する制御情報も Web クライアントに送る．

(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

XSLT プロセッサは，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する．次に，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 により実現されている．次に，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る．最後に，XDDS-S により，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 9** に示すようなテーブル形式の GUI を持ち，行ごとに，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．以下，個々の文書設計情報の作成手順を説明する．

(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．制約を記述する言語についての詳細は，参考文献 10) に譲る．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指定するために，文書制約ごとに,「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．

図 10 XDDS エディタにおけるテーブル写像型の指定
Fig. 10 Assignment of table mapping type in the XDDS editor.

(3) 文書スタイル記述部

XML 文書のタグごとに，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，ファイル添付など），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，および枠線太さなどのスタイル情報を設定する．本論文のアイデアの中核であるテーブル変換用パターンは，**図 10** に示すように，テーブル写像の原像となる XML 断片中のタグに対応付けて指定される．たとえば，図 10 中の 11 行目のタグ RAIL.SIZE の下位構造からなる XML 断片には，テーブル写像 T[(R(R)] (図中のパターン名は RR）が指定されている．この場合は，反復内 XPath は/DATA/RAIL.SIZE/RAIL.SIZE.CAR であり，反復外 XPath は/DATA/RAIL.SIZE/である．

(4) ワークフロー文書活用記述部

(a) 担当分野ごと・アクションごとの画面提示

XML 文書のタグごとに，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」であることを指定できる．

(b) 文書データの業務システム連携

XML 文書のタグごとに，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．この指定情報を用いて，XML 文書を CSV（Comma Separated Values）データ，パンチデータ，および別形式 XML 文書などに変換できる．

5. XML 入力画面自動生成方式の評価

本章では，5.1 節で，提案した XML 入力画面自動生成方式が 2.1 節の要求を満たしているかどうかを評価した結果を述べる．また，5.2 節では，本論文で提案する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，表示中心画面設計として典型的なツールである Microsoft 社の InfoPath（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，2.1 節で述べた要求に対応させて，提案方式を設計分野の実用文書処理アプリケーションへ適用評価した結果を述べる（適用アプリケーションの詳細については，参考文献 11) に譲る）．

(1) **XML 文書の構造・内容制約の複雑さへの対応**

図 1 に示した構造を持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 1,000）に対して 884 の内容検証規則を持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) **画面の複雑さへの対応**

- 約 1,000 個の項目を 37 の表を用いて入力するための本文フレーム，該当する表へのハイパーリンクを持つ見出しからなる目次フレーム，入力項目が選択項目に含まれない場合に特記仕様を記載するための特記フレーム，エラー箇所へのハイパーリンクとともに内容検証結果のエラーメッセージを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できた．

(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，工場の工事設計の機械設計，電気設計，意匠設計，および，色設計の 8 担当業務ごとに異なる入力画面と表示画面を作成できた．
- 営業設計支援システムから工事設計支援システムへのデータ連携を，XML から CSV への変換により実現できた．

5.2 XML 入力画面の開発工数の定量比較

XML 入力画面開発工数の表示中心ツールと論理中心ツールとの比較実験により，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

比較実験での対象文書，時間測定の作業区分，および被験者について述べる．

(1) **対象文書**

提案方式の一般性を確認するために，2.1 節で対象とした昇降機設計仕様書に加えて，WebEDI の資材伝票も評価対象とした．

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，チェックルール総数は 884 である．画面中のテーブル総数は 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，業務ごと画面としては，すべての項目を含む営業設計向けをベースとして，営業設計，機械設計，電気設計，意匠設計，色設計の 5 つの業務ごと画面を作成した．

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（1 種あたり平均 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 入力画面作成において共通作業であり，かつアプリケーションの要求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」は除いて，「画面レイアウト設定」，「チェックルール作成」，および「業務ごと画面カスタマイズ（昇降機設計仕様書のみ）」の作業時間を測定した．

(3) **被験者**

本比較実験の被験者は，対象文書についての理解があり，かつ XML の入門的な知識を有するもの 2 名とした．作業時間は，2 名の平均として算出した．

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．

表 2 昇降機設計仕様書用入力画面の作成時間
Table 2 Working hours of creating input-forms for elevator design specification.

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

表 3 要素の追加削除に要する平均時間の比較
Table 3 Comparison in average hours of adding or deleting XML elements.

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

表 4 WebEDI 資材伝票用入力画面の作成時間
Table 4 Working hours of createing input-forms for Web-EDI.

	表示中心 [時：分]	論理中心 [時：分]	削減率 [%]
画面レイアウト設定	9:01	3:07	65.4
チェックルール作成	3:41	2:33	30.8
計	12:42	5:40	55.4

(1) **昇降機設計仕様書の XML 入力画面作成時間**

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．

また，業務ごと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ごと画面を作成した．要素の追加/削除に要する平均的な作業時間を**表 3** に示す．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 1/2 削減（49.5%）でき，また，要素削除時間は約 2/3 削減（66.7%）できた．

(2) **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 4** 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．

6. 考　　察

本章では，5 章で述べた開発効率向上の要因について考察し，今後の課題として，表示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) **開発効率向上の要因**

画面レイアウト作成，チェックルール作成，および業務ごとカスタマイズの各々について，開発効率向上の要因を考察する．

- 画面レイアウト設定時間の削減効果は，設計仕様書の場合で 54.9%，資材帳票の場合で 65.4%である．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時間の削減効果は，設計仕様書の場合で 47.3%，資材帳票の場合で 30.8%である．この効果は，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ることにより，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より簡単に記述できたことによる．
- 業務ごとカスタマイズ時間の削減効果は，設計仕様書の場合で 76.6%である．この効果は，表示中心ツールでは，業務ごと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，業務ごとに異なる要素を選択するだけでよいことによる．この効果の背景には，データアクセス権の指定は論理中心設計の方が分かりやすいという要因と，表計算ソフトウェアはデータ一覧性が良く直感的に分かりやすいという要因がある．

(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，論理展開を単純化するために，表示中心設計と論理中心設計とを対比させて述べてきたが，理想的には，「表示中心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「論理中心設計が特長とする可変構造に関する制約表現能力」とを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の本質は，2.2 節で述べた表示・論理対応付けにあるので，木・表構造間写像モデルは，この融合のための道具の 1 つとして役立つことが期待できる．具体的な融合方法の一例は，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．本論文で提案した XDDS-E ツールは，論理中心画面設計に基づいており，「XML 文書構造やその上の制約が複雑」，「要素の追加削除のカスタマイズが頻繁」，そして，「画面レイアウトは比較的定型」という要求を持つアプリケーションに対して特に有効である．

7. おわりに

XML 文書が持つ木構造から HTML 文書が持つ表構造へ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 XML 入力画面生成方式を提案した．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実用システム評価により，「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．

謝辞 研究と実用システム適用の機会を与えてくださいました三菱電機（株）の市岡洋一氏，平田政信氏，藤本俊氏，前川隆昭氏，森岡雄二氏，竹内康晃氏，川田卓嗣氏らの皆様に感謝いたします．また，システム開発にご協力いただいた三菱電機（株）の堀場一夫氏，岡崎友紀氏，早水俊樹氏，谷原武郎氏，加藤慶氏，三菱電機インフォメーションシステムズ（株）の天沼敏幸氏らの皆様に感謝いたします．

参考文献

1) Clark, J.: XSL Transformations (XSLT) version1.0 (1999). http://www.w3.org/TR/xslt
2) Bex, G.j., Maneth, S. and Neven, F.: A formal model for an expressive fragment of XSLT, *Information Systems*, Vol.27, No.1, pp.21–39 (2002).
3) Tang, X. and Tompa, F.W.: Specifying Transformations for Structured Documents, WebDB (2001).
4) 鬼塚 真，小西一也：変換結果スキーマ試行の XML 変換，情報処理学会デジタルドキュメント研究会（DD）39-9, pp.39–46 (2003).
5) Adobe 社：FormDesigner (2002). http://www.adobe.co.jp/products/server/formdesigner/
6) Microsoft 社：InfoPath (2003). http://www.microsoft.com/japan/office/infopath/
7) Wood, L., et al.: Document Object Model (DOM) Level 1 Specification Version 1.0 (1998). http://www.w3.org/TR/REC-DOM-Level-1
8) Altova 社：XML Spy (2003). http://www.xmlspy.jp/
9) 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（DD）17-1, pp.1–8 (1999).
10) 今村 誠，増塩智宏，伊藤山彦：素性論理に基づく XML 文書ルール記述言語 DRDL，情報処理学会デジタルドキュメント研究会（DD）43-7, pp.47–54 (2004).
11) 今村 誠，増塩智宏，伊藤山彦：XML 文書ルール記述言語 DRDL とその EC システムへの応用，情報処理学会デジタルドキュメント研究会（DD）39-5, pp.23–30 (2003).

(平成 17 年 1 月 6 日受付)
(平成 17 年 10 月 11 日採録)

推薦文

新規性があり，かつ実用的にも有効と思われるので，推薦する．

（デジタル・ドキュメント研究会主査 大野邦夫）

今村 誠（正会員）

1984 年京都大学工学部数理工学科卒業．1986 年同大大学院修士課程修了．同年三菱電機（株）入社．自然言語処理と文書処理の研究開発に従事．現在，同社情報技術総合研究所音声・言語処理技術部言語処理チームリーダ．

森口 修（正会員）

1987 年横浜国立大学工学部電子情報工学科卒業．1989 年同大大学院修士課程修了．同年三菱電機（株）入社．現在，三菱電機インフォメーションシステムズ（株）にて，セキュリティ/ネットワークソリューションのコンサルティング業務に従事．

大場 雅之（正会員）

1989 年金沢工業大学卒業．同年三菱電機（株）入社．現在，同社稲沢製作所にて，昇降機の営業設計・受注工事設計の支援システム開発に従事．

鈴木 克志（正会員）

1979 年東京大学工学部計数工学科（数理コース）卒業．同年三菱電機（株）入社．自然言語処理の研究開発を経て，現在，同社情報技術総合研究所音声・言語処理技術部音声通信チームリーダ．

依田 文夫（正会員）

1978 年東京工業大学工学部電子工学科卒業．1980 年同大大学院修士課程修了．同年三菱電機（株）入社．手書き・印刷漢字認識，画像処理，文書処理，音声認識処理の研究に従事．現在，同社情報技術総合研究所マルチメディア技術部門統括．