構造定義を基にした入力画面の生成[1]

5W－6

西村 徹　由良 俊介　野瀬 昌禎[2]
NTT 情報流通プラットフォーム研究所[3]

1. はじめに

多種多様のシステムの情報を統合的に扱う手法として、項目を統合したデータ構造とデータ操作のための GUI 画面を定義し、統合したデータ構造と各システムのデータ構造の関係を別に定義し対応付けを行う手法が EAI などでは一般的である。

ビジネスを取り巻く環境の変化によって、各システムが管理すべき情報項目は変化し、また統合するシステムの対象範囲も広がる。このような状況下では、データ構造定義の項目の追加・変更が必要となり、その結果、データ操作のための画面やアプリケーションに修正が必要になる。

一方、多種多様なシステムのデータ構造の表現として、XML [1] を用いることが一般化している。これは、XML が企業間システムの連携等で利用されるためである。

本稿は XML の構造を定義した XML Schema [2] から、画面に対応する JSP (Java Server Pages, [3]) 等のコードを自動的に生成し、項目の追加や変更への対応を容易にする手法を提案する。

2. 要求条件

本稿では、構造定義を XML Schema をベースとして、定義に不必要な部分を排除し、不足する部分を拡張する方針を基本とする。ユーザセルフオペレーション等に利用することを想定し、画面は Web アプリケーションとして構築する。

自動生成されるアプリケーションは以下の条件を満たす必要がある。

(1) XML Schema 定義された構造にデータを配置するのに十分な入力画面であること
  (a) 要素の複数回出現(MaxOccurs)への対応
  (b) 選択(choice)・シーケンス(sequence)構造への対応
  (c) GUI フィールドに理解可能な名前を付加
  (d) 入力値のデータ構造への配置
(2) バックエンドシステムへのデータ登録・参照・更新が可能なこと
  (a) バックエンドシステムへの接続
  (b) Simple Type の型検査が可能
  (c) 入力画面と疎な関係

3. 実現手法

2 節の要求条件(2)(c)を満たすために、ビュー(画面)・アプリケーション(入力値のデータ構造への埋め込み)・データ(バックエンドシステムへのデータ登録等)の 3 層を分離した設計・開発が可能な J2EE を用いた実現手法について述べる。

3.1. 全体構成

本システムの全体構成を図 1 に示す。XML Schema は、システムで扱う情報項目の構造を定義する。要素の表示名定義ファイルは、GUI コンポーネントに理解可能な名前を付加するために（要求条件(1)(c)）、XML Schema で定義された要素に対して、GUI で表示する文字列を指定する。ジェネレータは、二つの定義ファイルを入力とし、入力フォームを構成する JSP ファイルと、Web ブラウザで入力され

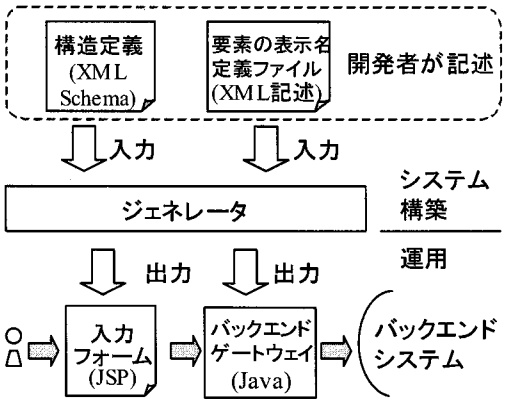

図1: システムの構成

[1] Input form creation based on structure definition
[2] Toru Nishimura, Shunsuke Yura, Masayoshi Nose
[3] NTT Information Sharing Platform Laboratories

た値に対する型検査、およびXML Schemaで定義された構造への変換、バックエンドシステムでの登録・更新の処理の呼び出し、の3つの処理を行うJavaアプリケーションを生成する。

3.2. ジェネレータによる画面(JSP)の生成

XML Schemaでの各要素定義に対して、どのような画面を表示すればよい、すなわち画面としての意味を与えることにより、ジェネレータは構造定義から画面を生成することができる。

以下では、代表的な要素定義に対して、対応する意味を与えることにより、画面を構成する例を示す(図2、要素“a”は要素の表示名定義ファイルにより、“フィールド1”と名付けられているとする)。

(1) 要素定義 要素定義に対して枠を用意し、その内部に、要素に割り当てた名前を表記する。また要素定義内でのタグを、maxOccurs属性が示す数だけ並べて再帰的に評価する。

(2) 要素定義(含む単純型定義) 要素定義に単純型定義を含む場合は、子要素の定義が存在しないことを意味する。よって入力用GUIコンポーネントと、それを一意に特定する属性名を割り当てる。枠と名前の表記に関しては、(1)と同様である。

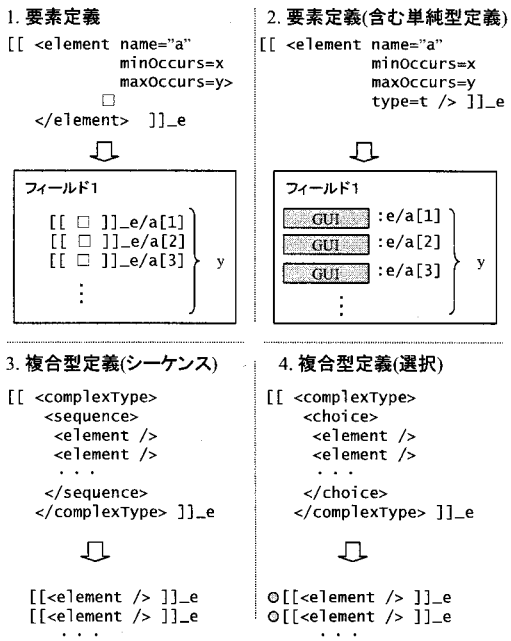

図2: 構造定義と画面の対応

(3) 複合型定義(シーケンス) sequenceタグの子タグで示される要素定義の評価をGUI上に並べる。

(4) 複合型定義(選択) (3)とほぼ同様であるが、要素の選択のためのラジオボタン等のGUIを、要素のためのGUIの先頭に付加する。

3.3. バックエンドゲートウェイ(Java)

GUIフォームに付けられた属性名(図2の2.の例の“e/a[1]”等)を、構造定義の末端の要素を示すXpathと構文的に等価とすることにより、バックエンドゲートウェイは、入力された値と構造との対応付けと値の型検査(図2の2.の例の場合、“e/a”の型は“t”)を行うことができる。

3.4. XML Schema定義の制限

以下に示す理由から、複合型定義において、選択・シーケンス定義での定義範囲を縮小する。

要素定義、<!ELEMENT a (b|(c,d))>(ここではDTDで表記する)を例に考える。3.1節で述べたように、画面表示は、要素に対して与えられる名前に基づくため、要素bとシーケンス(c,d)の選択において、(c,d)を表す名前を定義できない。従って、選択で定義される構造の直下に、要素定義以外のタグ[1]があってはならない。シーケンスと選択の並列性の観点から、シーケンス定義の直下もまた、要素定義以外のタグは含まれない。本制約は上記例に関して<!ELEMENT a (b|f)>、<!ELEMENT f (c,d)>と要素を定義することにより同一に構成できるため、データ構造の観点からは本質的な制約ではない。

4. まとめ

XML Schemaの構文に意味を与えることにより、入力フォームの画面を自動生成する方式を提案した。

参考文献

[1] Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, 6 October 2000.
[2] XML Schema Part 0: Primer, W3C Recommendation, 2 May 2001.
[3] Java Server Pages, “http://java.sun.com/j2ee/ja/jsp/,” Sun Microsystems.

[1]本稿では、XML Schemaの構文として定義されている要素と、XML Schemaを用いて定義する要素を厳密に区別するために、前者をタグ、後者を要素と呼称する。