デジタル・ドキュメント 29－1
（２００１．７．２７）

# 電子申請における XML 文書内容検証方式

## － 複数 XML 文書の内容間制約を記述する文書規約記述言語 DRDL －

今村 誠　　鈴木 克志
E-mail: {imamura, suzuki}@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

行政では XML(eXtensible Markup Language)を用いた電子申請のインフラ整備を進めている。そしてこのインフラ整備では、申請書本文とその申請内容を補足説明する添付文書からなる複数文書一式が、申請書の記載要領に相当する文書内容に関する規約(文書規約)に従っていることを保証するしくみが重要になる。しかし、XML Schema のような既存の標準では、要素間や複数文書間にわたる文書内容制約を表現できないという問題があった。そこで、複数 XML 文書一式に対する文書規約を表現・検証するしくみの提供を目的として、文書規約記述言語 DRDL(Document Rules Description Language)と、その処理系(DRDL プロセッサ)を開発している。DRDL の特長は、以下の 2 点である。(1) Xpath(XML Path Language)を基本要素として、等式、限量子、および論理演算子を用いて構成される論理式で文書規約を表現することにより、複数文書一式に対する内容間制約を簡潔に記述できる。 (2) 同一の文書規約を、文書内容検証用と文書変換用の双方の規則として解釈できるようにすることにより、申請書様式の電子化や改訂に伴う内容検証機能や変換機能の開発効率を向上させることができる。

# A validation method of XML documents for electronic application systems

Makoto IMAMUMA　Katsushi SUZUKI
E-mail: { imamura, suzuki }@isl.melco.co.jp

Mitsubishi Electric Corporation
Information Techonology R & D Center　Human Media Technology Dept.

Japanese Administration has developed XML-based Electronic Application infrastructure. In this development, it is important to establish a method to validate whether a set of documents consisting of a body document and attached documents satisfies document rules in application manuals. But XML Schema, a standard to describe document rules, can not represents a constraint among contents in multiple XML documents. This paper presents Document Rules Description Language (DRDL) and its processor (DRDL Processor) in order to provide a framework for description and validation of constraints among contents in multiple XML documents. The features of DRDL are the following. (1) DRDL can concisely describe constraints among contents in multiple XML documents with logical formula which consist of Xpath (XML Path Language), equality, quantifiers, and logical operators. (2) We can develop validation functions and transformation functions in application forms efficiently, as a DRDL processor can interpret one document rule as both validation procedure and transformation procedure.

## 1 はじめに

官公庁や自治体などの行政では、国民への行政サービス向上のニーズに応えるため、高度情報化社会とネットワーク社会に対応した「スーパ電子政府」を実現すべく、インターネットで許認可申請を行う「電子申請」のインフラ整備を進めている([1])。このインフラ整備では、申請書の形式を XML(eXtensible Markup Language)[2]で標準化するだけでなく、申請書本文とその申請内容を補足説明するための添付文書からなる複数文書一式を民間・行政間で交換するために、複数文書一式が申請書の記載要領に相当する文書内容に関する規約(文書規約)に従っていることを保証するしくみが重要になる。

しかし、XML 文書における文書規約を表現・検証するための従来技術である XML Schema([3][4])では、「要素内容間の制約(内容間制約)」、「要素の内容と、要素の繰返し数との関係制約(内容・構造間制約)」、および「ある XML 文書中の要素内容と、別の XML 文書の存在有無に関する制約(文書添付制約)」を表現できないという問題があった。

そこで、電子申請システムの開発で必要とされる複数 XML 文書一式に対する文書規約を表現・利用するしくみの提供を目的として、文書規約記述言語 DRDL(Document Rules Description Language )と、その処理系(DRDL プロセッサ)を開発している([5])。DRDL の主要特長は、以下の 2 点である。

(1) 高い表現能力により、電子申請アプリ要請に対応

検証対象要素を指し示す Xpath(XML Path Language)[6]を基本構成要素として、等式、限量子(すべて、ある)、および論理演算子(かつ,または,ならば,でない)を用いて構成される論理式で文書規約を表現することにより、内容間制約、内容・構造間制約、および文書添付制約などの電子申請アプリの要請に応じた複雑な内容制約を簡潔に記述できる。

(2) 柔軟な文書規約の解釈方式により、内容検証機能の開発効率を向上

DRDL プロセッサは、宣言的に記述された文書規約を解釈する際に、文書規約を満たすかどうかを検証する手続き(文書内容検証)としてだけでなく、要素の生成や要素内容の代入により文書規約を満たすような文書を生成する手続き(文書変換)として解釈することができる。すわなち、従来の Java Script や Java 言語での文書規約の実装では異なるプログラムとして表現されていた内容検証処理と変換処理を、一つの規則で実現できるので、申請書様式の電子化や改訂に伴う内容検証機能や変換機能の開発効率を向上させることができる。

本稿の構成は、以下の通りである。2 章では、電子申請における XML 文書内容検証方式の課題について述べる。3 章では、2 章で述べた課題を解決するための中核技術である文書規約記述言語 DRDL について述べる。4 章では、DRDL によって 2 章であげた課題を解決できることを、具体的な DRDL 記述の例をあげて説明する。最後の 5 章では、まとめを述べる。

## 2 電子申請における XML 文書内容検証方式の課題

本章では、電子申請システムにおける XML 文書内容検証機能への要求を整理し、その要求に対応する際に生じる現状技術の問題点について述べる。

### 2.1 電子申請システムにおける XML 文書内容検証機能への要求

現状では、XML 文書の内容検証機能を開発する際には、XML Schema で表現できるような文書規約に対しては、XML Schema 対応の XML パーサの機能を用いて検証し、XML Schema で表現できないような文書規約は、ハードコーディングした Java Script や Java のプログラムにより検証するのが通常であった。そのため、XML Schema で表現できないような文書規約に対する検証ロジックは、アプリケーションプログラムの中に埋め込まれることになり、「アプリケーション毎に文書規約を検証するロジックをコーディングする必要が生じる」、また「文書規約が変更になった場合には、アプリケーションプログラムを修正する必要がある」という問題があった。

以下では、申請書の作成支援ソフトウェアや審査支援ソフトウェアを実現する場合にそくして、上記の問題を解決するための XML 文書内容検証機能への要求を、「文書規約の表現能力」と「文書規約の再利用」に分けて説明する。

#### 2.1.1 文書規約の表現能力への要求

添付文書を含む申請書類一式の作成支援ソフトウェアには、申請書の記載内容に依存して必要な記載項目や添付文書が異なることをガイダンスしたり、また、必要な記載項目や添付文書が漏れなく記載・添付されているかを検証する機能があることが望ましい。このような機能を実現するには、XML 文書中のある項目の記載内容に応じて、他の項目の記載内

容や添付文書の必要性有無が決まるという規約を電子化する必要があるが、このような項目間や複数文書間にわたる動的な規約の表現は、XML Schemaでは対象外であった。項目間や複数文書間にわたる動的な規約の例としては、以下に示すような「内容間制約」、「内容・構造間制約」、および「添付文書制約」がある。

(1) 内容間制約

XML文書における内容間制約とは、要素内容間の制約のことである。申請書でいえば、記載項目間の制約に相当する。例えば、「単価に記載された値の合計値を、合計に記載する」とか、「設備に関する申請の場合、設備の許可年月日は、新規申請の場合には不要であるが、変更申請の場合には必須項目である」というような制約である。

(2) 内容・構造間制約

XML文書における内容・構造間制約とは、ある要素の内容と、別の要素の出現有無や繰返し数との関係制約のことである。申請書でいえば、ある記載項目の数値に依存して、別の記載項目の記入個数が決まるような制約である。例えば、設備に関する申請の場合、『申請書の記載項目「装置数」に記入した数と同じ数だけ、装置毎の説明項目を記載する必要がある』というような制約である。

(3) 添付文書制約

XML文書における添付文書制約とは、あるXML文書中の要素内容に依存して、別のXML文書の存在有無やそのXML文書中の内容が決まるという制約である。申請書でいえば、申請書本文の記載内容に応じて、必要な添付文書やその文書中の内容が決まるような制約に相当する。例えば、「申請書本文中に記載される申請種別に応じて、必要な添付文書が変わる。また、申請書本文中に記載された添付文書タイトルと、添付文書中に記載されたタイトルが同じでなければならない。」というような制約である。

#### 2.1.2 文書規約の再利用への要求

申請書の作成支援ソフトウェアや審査支援ソフトウェアを実現する場合には、本質的には同じ意味をもつ制約であるにもかかわらず、その制約を満たしているかどうかを検証する文書内容検証処理と、その制約を満たすような文書作成を支援する文書編集支援処理とが、別プログラムとして実現されることが通常であった。そのため、様式の記載要領の電子化や変更時に、文書内容検証用と文書編集支援用の二つのプログラムの作成/保守が必要になるという問題があり、できれば両プログラムを共用することが望ましい。

例えば、前節の(1)の例でいえば、「単価の合計値が、合計に記載された値と等しいことを検証する処理」と「単価がすべて埋まった時点で、単価の合計値を、合計に代入する処理」とが、別のプログラムとして実現されていた。また、前節の(2)の例でいえば、『申請書の記載項目「装置数」に記入した数と同じ数だけ、装置毎に必要とされる説明項目が記載されているかどうかを検証する処理』と『申請書の記載項目「装置数」が記入された時点で、その数と同じ個数分だけ、装置毎に必要とされる説明項目の記載入力枠を生成する処理」が別のプログラムで実現されていた。

以下では、DRDLと従来技術との比較を容易にするために、本項で述べたような文書編集支援処理を、編集対象XML文書に対する要素追加や要素内容代入により文書規約を満たす文書を生成する文書変換処理として扱う(変換処理で、変換前文書を変換後文書で上書きする場合として扱う)。

### 2.2 検証機能実現に必要な要素技術の現状レベルと課題

本節では、前節の要求を現状のXML要素技術で実現しようとする際に生じる課題について述べる。前節では、文書規約の表現能力と再利用への要求をあげたが、再利用への要求に応える現状技術はよく知られていないので、本節では、表現能力についての課題を中心に扱う。

以下では、XMLに対する文書規約の表現に関係する既存技術として、XML Schema、Xpath、およびXSLT(XSL Transformations)[7]をとりあげ、これらの技術で前節の要求を実現する際に生じる問題点と、DRDLによる解決スタンスについて述べる。

#### 2.2.1 XML Schema

XML文書の構造や内容に関する制約を記述する言語として、XML Schemaがある。XMLパーサを用いれば、与えられたXML文書がXML Schemaで記述した制約を満たすかどうかを検証することができる。XML Schemaは、XML1.0のDTD(Document Type Definition)に対して、データ型や継承を導入するなどの制約記述能力を向上させているが、2.1.1項で述べた「複数文書の要素内容間制約」、「内容・構造間

制約」、および「添付文書制約」は記述できない。そのため、これらの制約の検証機能の開発では、DOM(Document Object Model)の API などを用いたハードコーディングが必要になるので、申請書様式の電子化を効率化するためには、内容検証機能を簡単に作成／修正するためのしくみが望まれる。

DRDL は、XML Schema との併用利用を想定して、XML Schema では記述できない上記の制約を簡潔に記述する枠組みを提供することを目的としている。

#### 2.2.2 Xpath

Xpath は、単一 XML 文書内の一部分を指し示す(アドレッシング)するための言語である。Xpath では、ファイルシステムのような階層構造をたどる(トラバースする)ための式(以下では、Xpath 式と呼ぶ)を提供しており、「ある要素の内容がある値に等しい」、「要素 A の内容と要素 B の内容が等しい」、また、「要素 A の内容が、繰返し出現する要素 B の内容の合計値に等しい」などの要素内容に関する制約を宣言的に記述することができる。

しかし、「IF 文に相当する条件分岐に関する制約が記述できない」、また『「繰り返し出現する要素 A の内容は、すべてがある値以上である」といった全称記号で束縛された制約を記述できない』という問題がある。

DRDL では、Xpath がもつ宣言的な内容制約記述という特徴を保ちながら、IF 文に相当する条件分岐に関する制約や、全称限量子で束縛された制約などを記述できるようにしている。

#### 2.2.3 XSLT

XSLT は、XML 文書の要素毎に変換規則を記述する XML 文書変換用言語である。内容検証のエラー情報をファイル出力するという形態であれば、XSLT の提供するプログラム制御構造や組み込み関数を用いることで、1 文書内の内容間制約を表現することができる。

しかし、XSLT では、複数文書間にまたがる制約が記述できないという問題がある。また、同一文書内の要素内容間の制約を記述する際にも、XSLT のカレントノード毎に変換規則を適用するので、比較対象となる要素内容のコンテキストを指定する記述が煩雑になりやすく、プログラムの可読性や保守性が必ずしも高くないという問題もある。

DRDL では、XSLT と比較して以下の 3 つの特徴をもっている。

(1)DRDL では、文書制約を宣言的に記述することにより、一つの制約を文書内容検証用の規則としてだけでなく、文書変換用の規則としても利用できる。
(2)要素のコンテキストを指定しやすくするために、要素間の内容制約を簡潔に記述できる。
(3)XSLT では、対象文書を入力文書と出力文書に分けて規則を記述しているが、DRDL では、複数文書の内容間の制約を記述することにより、双方向性のある複数文書間の変換を規定できる。

## 3 文書規約記述言語 DRDL

本章では、XML Schema では記述できなかった内容間制約、内容・構造間制約、および添付文書制約を表現できるようにした文書規約記述言語 DRDL について述べる。以下、3.1 節では、DRDL の設計方針が、「要素内容間の制約を論理式として表現することにより、2.1.1 項の文書規約の表現能力への要求に応えること」と「一つの文書規約を文書内容検証と文書変換の双方に利用できるようにすることにより、2.1.2 項の文書規約の再利用への要求に応えること」であることについて述べる。3.2 節では、最初の設計方針の実現方式を説明するために、DRDL の構文と意味について述べる。3.3 節では、二番目の設計方針の実現方式を説明するために、DRDL の処理系である DRDL プロセッサの動作原理について述べる。

### 3.1 設計方針

DRDL の設計方針は、以下の 2 点である。

(1) 要素内容間の制約を論理式として表現

Xpath 式をベースにして、複数 XML 文書中の要素内容間の制約を、等式付一階言語の概念を用いて宣言的な論理式として表現する。この論理式を**パス制約式**と呼ぶ。パス制約式で表現された文書に対する規約を文書規約と呼ぶ。

(2) 一つの文書規約を文書内容検証と文書変換の双方に利用

パス制約式を、制約式を満たすかどうかを判定することにより内容検証機能を実現するだけでなく、制約式を満たすように要素の生成や要素内容への代入をする処理により、文書変換機能を実現する。

### 3.2 DRDL の構文と意味

本節では、DRDL による制約表現の本質部分を説明するために、パス制約式の中核サブセットの構文と意味について述べる。

#### 3.2.1 DRDL の構文

一階論理の論理式が項から構成されているように、パス制約式もパス項から構成される。以下、パス項とパス制約式の構文について述べる。

【定義】 パス項

(a) 変数は、パス項である。
(b) 定数は、パス項である。ここで、定数とは、XPath が定めるオブジェクトの基本型である"ノード集合"、"ブール値"、"数値"、"文字列"に属する値とする。
(c) Xpath 式は、パス項である。また、Xpath 式を評価する際のコンテキストである対象 XML 文書は、URI(Uniform Resource Indentifiers)で指定する(**コンテキストファイル**と呼ぶ)。また、コンテキストファイル中で処理対象となるノードは、Xpath 式で指定する(**コンテキストノード**と呼ぶ)。

【定義】 パス制約式

(a) s と t がパス項ならば、s=t、s!=t、s<=t、s>=t、s>t、および s<t は、パス制約式である。s=t を**パス等式**と呼ぶ。
(b) s が Xpath 式であり、t が XML Schema で定義された型ならば、s: t は、パス制約式である。s: t を**パス型式**と呼ぶ。
(c) s と t が Xpath 式ならば、occurs(s) = t、occurs(s) <=t、および occurs(s) >=t は、パス制約式である。occurs(s) = t を**出現数パス制約式**と呼ぶ。要素の出現数を規定する際に用いられる。
(d) x が変数、s が Xpath 式、そして Φ がパス制約式ならば、∀ x ∈ eval(s). Φ は、パス制約式である(**全称パス制約式**と呼ぶ)。ここで、eval(s)は、Xpath の仕様にしたがって、Xpath 式を評価して得られるノード集合を意味する。また、変数 x は、パス制約式 Φ 中の Xpath 式のコンテキストノードを指定するために用いられる(4.5 節と 4.6 節で具体例を示す)。
(e) x が変数、s が Xpath 式、そして Φ がパス制約式ならば、∃ x ∈ eval(s). Φ は、パス制約式である(**存在パス制約式**と呼ぶ)。また、全称パス制約式と存在パス制約式を総称して**限量子付パス制約式**と呼ぶ。
(f) Φ と Ψ がパス制約式ならば、Φ ∧ Ψ、Φ ∨ Ψ、¬ Φ、および Φ ⇒ Ψ は、パス制約式である。これらの制約式を順に、**連言パス制約式**、**選言パス制約式**、**否定パス制約式**、および**含意パス制約式**と呼ぶ。
(g) s と t が Xpath 式ならば、$s \wedge_{seq} t$ は、パス制約式である。$s \wedge_{seq} t$ を**連言パス順序制約式**と呼ぶ。

#### 3.2.2 DRDL の意味と従来技術との差異

パス制約式の意味とは、XML 文書が与えられたときに、そのパス制約式を満足するかどうか(充足可能性)を判定することである。パス制約式の充足可能性の判定手続きは、検証対象 XML 文書が与えられると、限量子で束縛された変数には具体的な XML 文書中の要素を割り当てることができるので、命題論理の充足可能性判定の手続きに帰着することができる。

パス制約式の意味の詳細については、4 章で具体例と共に説明することにして、本項では、各々の制約式の従来技術との差異を中心に説明する。

(a) パス等式 s=t は、Xpath の等式では表現できなかったノード集合としての等価性を表現できるようにするための構文である。また、パス項の定義で述べたコンテキストファイルを指定する構文により、複数 XML 文書の内容間制約を表現することができる。
(b) パス型式は、XML Schema を併用した制約記述を実現するためのものである。また、型は、制約が非常に複雑ないし特殊なため別プログラムでコーディングした検証ロジックを呼び出すためのしくみとしても利用できる。
(c) 出現数パス制約式は、2.1 節で述べた内容・構造間制約を記述する際に用いられる。XML Schema で要素の出現数を指定する属性 minOccurs や maxOccurs に、整数値だけでなく、XML 文書中の要素内容を参照できるようにしたものに相当する。
(d) 全称パス制約式は、Xpath 式では表現できなかった限量子を扱えるようにするための構文である。また、XSLT で for-each 構文を用いて手続き続き的に表現していた制約を宣言的に記述することができる。
(e) 存在パス制約式は、ほぼ Xpath 式で表現できる部分に相当する。
(f) 含意パス制約式は、通常のプログラミング言語の if-then 構文と同等の機能を持っているので、2.1 節で説明した内容間制約を表現するために用いられる。
(g) 連言パス順序制約式は、要素の出現順序に関する制約を記述するものであり、XML Schema の sequence 構文に相当する。

### 3.3 DRDL プロセッサ

DRDL プロセッサは、パス制約式を文書内容検証用の規則としてだけでなく、文書変換用の規則としても解釈できるようにしている。

文書内容検証用の規則として処理する場合には、処理対象の XML 文書がパス制約式を満たしているかどうかを判定する手続きとして解釈する。この解釈モードを、**内容検証モード**と呼ぶ。

また、文書変換用の規則として処理する場合には、処理対象の XML 文書がパス制約式を満たすように、XML 文書を変換する手続きとして解釈する。文書変

換手続きとしての解釈には、要素の内容が空の場合にのみ代入を許す**内容補完モード**と、要素の内容が空でない場合でも代入を許す**内容代入モード**とがある。以下、内容補完モードと内容代入モードの処理手続きについて簡単に説明する。

内容補完モードでは、(否定や含意の前提に含まれていない)存在パス制約式において、存在限量子で束縛しようとしている要素が存在しない場合には、その要素を生成する。また、存在パス制約式中のパス等式の右辺と左辺が指し示す要素(または、値)が存在しており、かつ、どちらか一方の要素の内容が空の場合にのみ、一方の要素内容に他方の値を代入する。また、存在限量子で束縛されている要素を含むコンテキストファイルが存在しない場合には、そのファイルを生成する。

内容代入モードでは、要素の内容が空でなくても、パス等式中の左辺の指し示す要素の内容に、右辺の値を上書き代入する点が、内容補完モードと異なる。

DRDL プロセッサは、DRDL 記述の手続き的な解釈を内容検証モードと内容補完モード(または、内容代入モード)とを呼び出し時に切りかえることにより、文書内容検証用の規則と文書変換用の規則を共有することができる。

## 4 DRDL による文書規約の記述例

本章では、2.1 節で述べた電子申請における文書規約の記述例と、3.2 節で述べた DRDL と従来技術との差異特徴を説明する記述例を示す。以下では、「複数文書間の内容制約」、「構造・内容間制約」、「添付文書制約」、「限量子付パス制約式を用いた制約」、「パス型式を用いた制約」、および「Xpath が提供する関数を用いた制約」について述べる。

DRDL では、3 章で述べたパス制約式を以下の表 1 に示すように、XML 構文を用いて記述する。

表1 パス制約式の XML 構文による表現

| パス制約式 | XML 構文による表現 |
|---|---|
| s=t | <path path="s" value-op="=" value="t"/><br>注)valu-op のディフォルト値は=である。 |
| s: t | <path path="s" type="t"/> |
| occurs(s) = t | <path path="s" occurs –op="=" occurs="t"/> |
| $\forall x \in eval(s).\Phi$ | <all.path path="s" > Φ </all.path><br>注) 変数は Φ中の path 属性の値中の.として参照される。 |
| $\exists x \in eval(s).\Phi$ | <exist.path path="s" > Φ </exist.path> |
| $\Phi \wedge \Psi$ | <and.path> Φ Ψ</and.path> |
| $\Phi \vee \Psi$ | <or.path> Φ Ψ</or.path> |
| $\neg \Phi$ | <not.path> Φ</not.path> |
| $\Phi \Rightarrow \Psi$ | <if-then.path><br><if.path> Φ </if.path><br><then.path> Ψ</then.path><br></if-then.path> |
| $s \wedge_{seq} t$ | <sequence.path> s t </sequence.path > |

### 4.1 複数文書間の内容間制約

図 1は、複数文書間の内容間制約を表現する DRDL 記述の例であり、『「本文.xml」の要素「種別」の内容が「A」であり、かつ、「添付.xml」の要素「選択」の内容が「変更」の場合には、「添付.xml」の要素「許可年月日」は空文字であってはならない。また、この制約を満たさない場合のエラーメッセージは、「種別 A の変更届けでは、許可の年月日の記載が必要です」である』という制約を表現している。

```
<if-then.path path-context-file="./添付.xml"
              f-msg="'種別 A の変更届けでは、許可
                     の年月日の記載が必要です'"">
 <if.path>
  <and.path>
   <exist.path path-context-file="../本文.xml"
               path="種別" value="'A'" />
   <exist.path path="選択" value="'変更'"/>
  </and.path>
</if.path>
 <then.path>
  <exist.path path="許可年月日" value-op="!="
              value="``"/>
 </then.path>
</if-then.path>
```

図 1 複数文書間の内容間制約の記述例 1

各々のパス制約式では、path-context-file を用いて、path 属性で指定される Xpath 式のコンテキストファイルである XML 文書を指定できる(同様に、path-context-node を用いて、コンテキストノードを指定できる)。path と value 毎に、パス制約式の入れ子階層に対応してディフォールトのコンテキストを継承できるようにしているので、XSLT と比較して、比較対象となる要素内容のコンテキストを簡潔に記述できるようになっている。

図 2も、複数文書間の内容間制約の記述例であり、『「申請書.XML」の要素「申請者名」の内容は、「ユーザ登録情報.XML」の要素「法人名」と「姓」と「名」を連結した文字列と等しい。』という制約を表現している。

```
<exist-path path-context-file="./申請書.XML"
  path="申請者名"
  value-context-file="./ユーザ登録情報.XML"
  value-context-node="申請者"
  value="cancat(法人名,' ',氏名/姓, 氏名/名)"/>
```

図 2 複数文書間の内容間制約の記述例 2

図 2のパス制約式を入力として、DRDL プロセッサの内容補完モードで解釈実行させると、『「申請書.XML」の要素「申請者名」の内容が空白の場合には、「ユーザ登録情報.XML」の要素「法人名」、「姓」、および「名」を連結した文字列を代入する』という手続きが実行される。このように、内容検証用規則と文書変換用規則を共通化することができる。

### 4.2 構造・内容間制約

図 3は、構造・内容間制約を表現する DRDL 記述の例であり、『要素「装置」の出現数は、要素「装置数」の内容に等しい』という制約を表現している。

```
<exist.path path="装置"   occurs-op="="
            occurs="装置数"/>
```

図 3 構造・内容間制約の記述例

図 4の XML 文書を処理対象として、図 3のパス制約式を DRDL プロセッサの内容補完モードで解釈実行させると、『要素「設備」を 3 つ追加して、要素「設備」の出現数を要素「装置数」の内容である 5 と等しくなるようにする』という処理が実行される。この処理を XML 文書入力支援ソフトウェアから呼び出すことにより、申請書中のある記載項目の内容に応じて、他の記載項目の入力枠数を可変にするような動的な入力フォームを実現することができる。

```
<装置数> 5 </装置数>
<装置リスト>
  <装置>  ... </装置>
  <装置>  ... </装置>
</装置リスト>
```

図 4 処理対象 XML 文書の例

### 4.3 添付文書制約

図 5は、添付文書制約の記述例であり、『「本文.XML」の要素「添付タイプ」の内容が「A」の場合には、「添付 A.xml」という XML 文書が存在して、かつその要素「タイトル」の内容は、「本文.XML」の要素「添付タイトル」の内容に等しい』という制約を表現している。

```
<if-then.path path-context-file="./本文.xml"
              value-context-file="./本文.xml">
 <if.path>
  <exist-path path="添付タイプ" value="A"/>
 </if.path>
 <then.path>
  <exist-path path-context-file="添付 A.xml"
              path-context-file-template="templateA.xml"
              path="タイトル"
              value="添付タイトル"/>
 </then.path>
</if-then.path>
```

図 5 添付文書制約の記述例

添付 A.xml という XML 文書が存在しない状態で、図 5のパス制約式を入力として、DRDL プロセッサの内容補完モードで解釈実行させると、『「本文.xml」の要素「添付タイプ」の内容が「A」の場合には、「templateA.xml」をコピーすることにより「添付 A.xml」という XML 文書を生成して、「添付 A.xml」の要素「タイトル」の内容に、「本文.xml」の要素「添付タイトル」の内容を代入する。』という手続きが実行される。

### 4.4 限量子パス制約式を用いた制約

図 6は、限量子パス制約式を用いた制約の記述例であり、図 7のような社員名簿を表現する XML 文書を対象として、『すべての課には、「役職」が「課長」である「社員」が少なくとも 1 人存在する。』という制約を表現している。

```
<all-path path="課" >
  <exist-path path="[./社員/役職]=`課長`]"/>
</all-path>
```

図 6 限量子パス制約式を用いた制約の記述例

全称パス制約式により、図 7のような XML 文書に対して、要素の繰返しをもつ入れ子構造をたどる(トラバースする)処理を記述することができる。

```
<課 課名="総務">
  <社員>
    <役職> 課長 </役職>
    <名前> ○○</名前>
  </社員>
  <社員>  ... </社員>      ...
</課>
<課 課名="人事">
  <社員>
    <役職> 課長 </役職>
    <名前> ○○</名前>
  </社員>
  <社員>  ... </社員>    ...
</課>
```

図 7 処理対象 XML 文書の例

### 4.5 パス型式を用いた制約

図 8は、パス型式を用いた制約の記述例であり、『要素「商品」の子要素「カテゴリ」が「通信」の場合には、「番号」の型は「通信機器コード」でなければならない』という制約を表現している。但し、ここで「通信機器コード」は、XML Schema のデータ型として定義されているものとする。

```
<all-path path-context="商品">
 <if-then.path>
  <if.path>
   <exist-path path="./カテゴリ"   value="通信"/>
  </if.path>
 <then.path>
   <exist-path path="./番号"   type="通信機器コード"/>
 </then.path>
</all-path>
```

図 8 パス型式を用いた制約の記述例

パス型式を他の制約パス式と併用することにより、XML Schema では記述できなかった「ある要素の内容に依存して決まるデータ型」や「要素内容間のデータ型制約」を記述することができる。

### 4.6 Xpath が提供する関数を用いた制約

図 9は、Xpath 式の数値演算関数「sum」を用いた制約の記述例であり、図 10のような XML 文書を対象として、『通貨がドルである小計の価格は、通貨がドルである商品の価格の合計値と等しい』という制約を表現している。sum や四則演算などの Xpath 式の提供関数を用いたパス制約式は、XML 文書入力支援ソフトウェアから DRDL プロセッサを内容補完モードで呼び出すことにより、表計算ソフトウェアのセル値間計算に相当する編集支援機能を実現できる。

```
<exit-path path="小計/価格[../通貨=`dollar`] "
           op="="
           value="sum(商品/価格[..通貨=`dollar`])" />
```

図 9 Xpath 式が提供する関数を用いた制約の記述例

```
<商品リスト>
  <商品>
    <通貨> dollar   </通貨>
    <価格> 200 </価格>
  </商品>
  <商品>   </商品>      . . . . . .
</商品リスト>
<合計>
  <小計>
    <通貨> dollar </通貨>
    <価格>        </価格>
  </小計>
  <小計>      </小計>   . . . . . .
</合計>
```

図 10 処理対象 XML 文書の例

## 5 おわりに

XML 文書の内容検証用規則を規定する標準である XML Schema と併用して、複数 XML 文書一式に対する文書規約を表現する言語 DRDL の構文と意味について述べた。そして、「電子申請システムの開発に必要とされる複数XML文書一式に対する文書規約をパス制約式により表現できること」、「パス制約式を内容検証モードで解釈実行することにより内容検証機能が実現できること、また、内容補完モードで解釈実行することにより文書変換機能が実現できること」を示した。

今後の課題は、以下に示す「DRDL による検証機能開発の効率の定量的評価」と「DRDL 記述の作成支援方式の開発」である。

(1) DRDL による検証機能開発の効率の定量的評価

文書規約を、従来の DOM API や XSLT による記述と比較することにより、DRDL の記述効率を定量的に評価する。また、同様に、文書内容検証機能や文書変換機能の開発効率を定量的に比較評価する。

(2) DRDL 記述の作成支援方式の開発

申請書様式の作成と保守を円滑に進めるためには、XML の専門家でなくても DRDL 記述を作成できるようにすることが望ましい。申請書様式で用いられる文書規約を類型化して、その類型に対応する DRDL 記述の雛形を提供するなどの機能をもった DRDL 記述作成支援方式を開発することは今後の課題である。

### [参考文献]


[1] IT 戦略本部： e-Japan 重点計画, http://www.kantei.go.jp/jp/it/index.html (2001)

[2] Bray T., Paoli J. and Sperberg-McQueen C. M.: "Extensible Markup Language (XML) 1.0", http://www.w3.org/TR/1998/REC-xml-19980210 (1998)

[3] Thompson H. S.他: XML Schema Part 1 Structures, http://www.w3.org/TR/xmlschema-1/ (2000)

[4] Biron P.V. and Malhotra A.: XML Schema Part 2 Datatypes, http://www.w3.org/TR/xmlschema-2/ (2000)

[5] 今村 誠,長浜 隆次,鈴木 克志,渡部 明洋：電子申請における XML 文書利用技術 -電子政府実現に向けた外為法 EDI システム(JETRAS)への適用-, 情報処理学会 デジタルドキュメントシンポジウム 2000 (2000)

[6] James Clark, Steve DeRose : XML Path Language (XPath) Version 1.0, http://www.w3.org/TR/1999/REC-xpath-19991116 (1999)

[7] James Clark : XSL Transformations (XSLT) Version 1.0, http://www.w3.org/TR/xslt (1999)