

権利を階層的に定義可能な権利情報定義言語: XML Ticket

中嶋 良彰 寺田 雅之 藤村 考
yoshiaki@isl.ntt.co.jp te@isl.ntt.co.jp fujimura@isl.ntt.co.jp

NTT 情報流通プラットフォーム研究所
〒 239-0847 神奈川県 横須賀市 光の丘 1-1

あらまし 本稿では物理的媒体によって実世界を流通する多様な権利を、電子情報として定義可能な権利情報定義言語「XML Ticket」を提案する。この言語は、記述シンタックスに XML を利用し、可読性の高い多様な権利内容の記述を可能にする。しかしながら、一般的に、XML 文書は冗長な記述となるため、実装時の効率が期待できない。そこで、本稿では、権利定義を階層的に記述することを提案する。この方法によって、記述効率、処理効率を高め、サービス構築時の性能向上やコスト低減を可能にする。また、階層関係に付与する制約情報によって、権利定義の改竄、不正な階層関係を検出し、安全性の向上を図ることが可能である。

キーワード 権利, チケット, 継承, XML, オブジェクト指向言語, 電子商取引

XML Ticket: Rights Definition Language with Secure Inheritance

Yoshiaki Nakajima Masayuki Terada Ko Fujimura
yoshiaki@isl.ntt.co.jp te@isl.ntt.co.jp fujimura@isl.ntt.co.jp

NTT Information Sharing Platform Laboratories
1-1 Hikarino-oka, Yokosuka, Kanagawa, 239-0847 Japan

Abstract This paper proposes the digital rights definition language 'XML Ticket'. It is a language designed to specify diverse types of rights that are currently circulated as pieces of physical medium in the real world. As its name implies, 'XML Ticket' adopts the XML language syntax, which makes its description human-readable. Typically, XML-based documents suffer from the inefficiency caused by their verbosity. 'XML Ticket' solves this problem by adopting hierarchical descriptions. Describing digital rights hierarchically enables us to efficiently define and process the digital rights. Our language also supports secure inheritance for specifying hierarchical relationships, thereby enabling the detection of falsified or invalid inheritance of rights definitions.

key words Rights, ticket, inheritance, XML, object oriented language, electronic commerce

1. はじめに

実世界において、権利の所有者であることは、権利を表象する物体を保有する（もしくは存在する）ことによって表される。例えば、コンサートチケット、運転免許証、ソフトウェアライセンス証などは、複製が困難な物理的媒体に、権利内容が文字や絵として印刷された証明書であり、これを保有することによって、権利の所有者であることを表す。

そこで、このような物理的媒体を用いて実現される権利を電子データとして表現することによって、インターネット上での権利流通を促進し、権利の購入、譲渡、利用時における利用者の利便性を大幅に向上させる方法が提案されている。^[1]

物理的媒体によって表わされる権利を電子化するには、(1) 偽造、改竄、複製といった攻撃から十分に保護され、(2) 多様な権利内容を電子的に表現可能であることが重要である*。

電子データの偽造、改竄に対しては、電子署名を用いる方法が一般的である。また、不正な複製については、IC カード、安全なサーバ等の装置を利用して、原本性を保証する方式が提案されている。^{[2][3]}

一方、多様な権利内容を表現可能、かつ装置やプログラム等によって機械的に処理可能な共通形式を定める方法が提案されている。^[4] このような共通形式を用いる方法は、権利の処理部分が特定の形式に特化しないため、処理装置が権利の種類に依存しにくく、装置の処理対象となる権利の種類が多い場合や種類が不特定である場合に有効である。

権利内容を電子的に表現するための共通形式が満たすべき要件としては、(1) 多様な権利内容を表現可能であること（多様性）が必須であり、さらに、(2) 実用的な性能や構築コストおよび管理コストが達成できる形式であること（効率性）、がある。

本稿では、上記の要件を満たす権利情報定義言語 XML Ticket を提案する。XML Ticket では、記述シンタックスに XML^[5] を使用する。XML には、記述の多様性、高い可読性、豊富な周辺ツールの存在などのメリットがある。しかし、シンタックスに XML を利用すると権利の記述が冗長となり、記述コスト、処理コストが増大する傾向があり、上記の (2) の効率性の要件を

充足できない。そこで、XML Ticket では、この問題に対して、階層的な権利の記述方式によって記述効率や処理効率を向上させる。また、階層関係に制約を付与することによって、階層関係の安全性を高めている。

2. XML Ticket の設計方針

権利を電子化し、プログラムや装置によって機械的に処理するためには、権利定義（権利内容を表わす電子データ）を機械解釈可能な形式とする必要がある。権利の記述者は、施設の入場権、商品の引換権といった権利内容から構成要素を抽出し、上記の形式に従って権利定義を記述する。

一方、人間の目で権利内容を確認する場合は、断片的に抽出された情報の集合よりも、自然言語によって表現された情報の方が理解し易く都合が良い。例えば、権利内容の確認を人間の目で行なう場合、約款のように自然言語の形式によって表わす方が権利内容の意味が詳細かつ明確になる。

上記のような要求に対して、XML Ticket では、記述の多様性、可読性、インターネットにおける標準的な情報交換の形式であることを考慮し、シンタックスに XML を利用する。XML の利用によって、豊富な周辺ツールによる開発コストの低減、既存システムとの相互接続性等の利点も得ることができる。

XML 文書のデータ構造は簡易な木構造であり、機械解釈を容易に行なえる構造である。また、XML 文書は、テキストデータであるため、木構造を構成するノード内に、自然言語による文字情報を含めることができる。このため、機械解釈可能な形式であると同時に、木構造と自然言語による多様な権利記述が可能である。

しかし、XML を単に利用するだけでは、多様性の要件には対応できても、以下のような効率性の問題が解決されない。

記述効率 XML 文書は冗長であるため、権利定義の記述量が多くなる。

処理効率 XML 文書は冗長であるため、権利定義のデータ量が多くなり、解釈、通信、蓄積コストなどが大きくなる。

上記の問題点を解決するために、XML Ticket では、オブジェクト指向の継承機構による差分プログラミング技術に着目し、権利定義の記述方法に継承の概念を導入した。すなわち、上位の権利定義の内容を、下位の権利定義が受け継ぐこととし、権利定義の差分記述を可能にする。これによって、権利定義の再利用や

* 例えば、紙媒体によるチケットシステムを電子化するには、紙チケットシステムと電子チケットシステムの相互接続性が要求されることが多く、物理的媒体による権利表現と相互変換可能であることが重要であるが、実装および運用面で対処可能と考え、本稿の議論からは割愛する。

多重継承による記述の効率化、権利定義のグループ化を実現できる。また、権利定義の階層関係の中で、処理頻度が特に高い権利定義については、処理結果を繰り返し用いることによって、処理効率を向上させることができる。

しかし、インターネットのようなオープンなネットワーク上で、相互に信用できない発行者、改札者、利用者の間を権利が流通する場合は、権利定義、および継承関係を信用するために正当性の検証手段が必要になる。例えば、上位の権利定義を参照するために単に URL や ID を用いる方法では、次のような問題が生じる可能性がある。

- 下位の権利定義の記述後に、上位の権利定義の内容変更(または改竄)が行なわれることによって、下位の権利定義の記述者が意図していなかった権利内容が継承されること(または意図していた権利内容が継承されなくなること)。
 - 上位の権利定義の記述者が意図していない権利内容の差分追加が、下位の権利定義でなされること。
- また、これによって、継承による正しいグループ化が行なえなくなる。

すなわち、上記の効率化を目的として権利定義の記述に継承を導入するにあたっては、以下の安全性の課題を解決する必要がある。

安全性 異なる事業主による多様な権利定義の分散的な作成を可能にすると、権利定義の改竄、偽造、不正な継承が行なわれる可能性がある。

上記の課題に対して、本稿の方式では、下位の権利定義が上位の権利定義のダイジェスト値を保持し、この値を比較することによって、上位の権利定義が下位の権利定義を記述した時点の権利定義と相違ないことを検証可能にする。また、上位の権利定義に差分記述時の規則を定めることにより、上位の権利定義の記述者が意図していない記述を、下位の権利定義の記述者が行なうことを禁止可能にする。

3. XML Ticket の定義方式

本章では、XML Ticket における権利定義の継承機構の利用目的、継承規則、権利定義の情報構造、内容取得の処理方法について述べる。

3.1 継承機構の利用目的

以下で、ファーストフードショップの商品引換券を例に、権利定義の継承機構の利用例を示し、継承機構の利用目的を説明する。

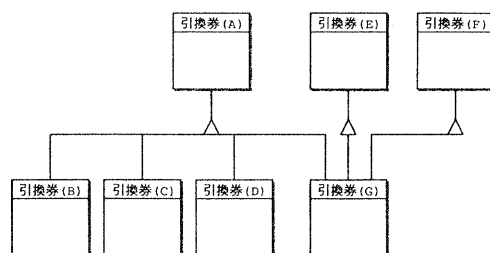


図1 商品引換券の継承関係

権利の特殊化 全国ファーストフードショップで利用可能なハンバーガ引換券(A)を継承して、横須賀市内の店舗のみで利用可能な引換券(B)を定義できる。

権利の拡張 全国ファーストフードショップで利用可能なハンバーガ引換券(A)を継承して、ジュースもセットで引き換えられる引換券(C)を定義できる。

権利のグループ化 引換券(A)を継承して作られた引換券(B)や引換券(C)は、ハンバーガ引換券(A)を対象とするサービスを利用することができる。すなわち、引換券(B)や引換券(C)は、引換券(A)によって、グループ化される。引換券(A)を対象とするサービスは、サービス開始後でも、引換券(A)を継承する引換券(D)を定義することによって、サービスの対象権利を動的に追加できる。

権利の複合化 ハンバーガ引換券(A)、ジュース引換券(E)、ポテト引換券(F)を継承して、セット引換券(G)を定義できる。

XML Ticket を設計するにあたっては、以上の例を実現可能な継承機構を備えることを目標とした。なお、上記の商品引換券の継承関係を図1に示す。

3.2 継承規則

第2章で述べた権利定義の継承機構を実現するにあたって、権利定義間に階層関係を構築することのみでは、安全性の面で不十分である。例えば、上位の権利定義の記述者と下位の権利定義の記述者が異なる場合、権利定義のグループ化のためには、上位の権利定義の記述者の意図する範囲で、下位の権利定義の記述者に権利定義を記述させたい場合がある。また、権利定義がインターネット上で転送され得るため、取得した権利定義の内容を必ずしも信用できない。そこで、筆者らは、権利定義の継承機構に次の2つの機能を備えることを提案する。

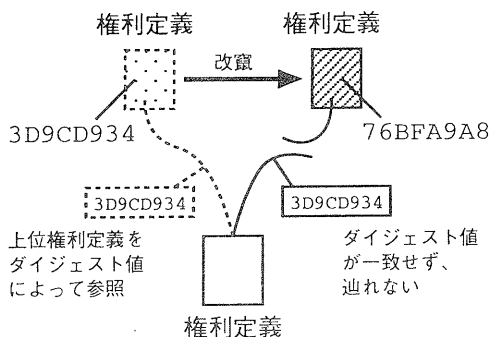


図 2 ダイジェスト値による上位権利定義の改竄の検出

- (1) 下位の権利定義の記述者が、上位の権利定義の変更を禁止する機能
- (2) 上位の権利定義の記述者が、下位の権利定義による再定義を制御する機能

前者については、下位の権利定義から、上位の権利定義を指定する際に、権利定義のダイジェスト値を用いることによって実現する。図 2 に示すように、上位の権利定義が変更された場合、権利定義のダイジェスト値も変化するため、上位の権利定義の変更を検出することができる。

後者については、上位の権利定義内に、再定義の「禁止」および「義務づけ」を記述する。例えば、再定義の「禁止」は、上位の権利定義で定められる権利の有効期間を、下位の権利定義によって再定義されないために設定する。一方、「義務づけ」は、上位の権利定義が、記述時点では値が定まらない構成要素を持つ場合に、下位の権利定義による継承時に、構成要素の値の再定義を義務付けるために設定する。

これらの機能を実現するために、本方式では、下位の権利定義による内容変更（再定義）が禁止されていることを示す「固定宣言 (fixed)」^{*}、下位の権利定義によって内容を設定する必要があることを示す「必須宣言 (required)」^{*}を、上位の権利定義の各構成要素に対して継承制約 (constraint) として付与し、再定義の禁止と義務づけを表わす。

3.3 権利定義の構造

XML Ticket によって記述される権利定義の構造は、第 3.1 節の例に見られる継承機構を実現するために、次の通りとする。

- 権利定義 (RD) は、権利内容 (R) と上位の権利定義を指し示す参照情報 (P) によって構成される。
($RD = \langle R, P \rangle$)

```
<RightDef>
<View>
<Name>ハンバーガー|換券</Name>
<Digest>ハンバーガー 1 個と無料引き換え</Digest>
<Image
resource="http://ffood.co.jp/ticket.gif"/>
</Image>
</View>
<Validity>
<Period>
<Start>2000-07-01T00:00+09:00</Start>
<End>2001-01-01T00:00+09:00</End>
</Period>
</Validity>
<Issuer>
<Cond>
<HasKey hash="3F87110AFCB251D36B18"/>
</Cond>
</Issuer>
</Validity>
<Promise>
<上限額 単位="円">300</上限額>
</Promise>
</RightDef>
```

図 3 権利定義の記述例 1

```
<RightDef>
<Parent hash="7113F8B250AFCB181D36">
<Promise merged="true">
<利用可能店舗>YRP 野比店</利用可能店舗>
</Promise>
</RightDef>
```

図 4 権利定義の記述例 2

- 権利内容 (R) は、表示情報 (View)、有効条件 (Validity)、固有情報 (Promise)、継承制約 (Constraint) によって構成される。($R = \langle View, Validity, Promise, Constraint \rangle$)
- 参照情報 (P) は、任意個 (0 個以上) の上位の権利定義を参照し得る。($P = \{p_1, p_2, \dots, p_n\}$)
- 参照情報 (P) が示す上位の権利定義の権利内容は、権利定義 (RD) の権利内容 (R) に引き継がれる。ただし、下位の権利定義は、上位の権利定義が持つ継承制約 (Constraint) を満たさなければならない。
- 参照情報 (P) の各要素 (p_i) は、上位の権利定義のダイジェスト値^{*}とする。

XML Ticket では、上記の構造に基づき、XML のシンタックスに従って権利定義を記述するものとする。なお、権利定義の記述例を図 3、図 4 に示す。図 4 の権利定義は、図 3 の権利定義を継承している。

3.4 内容取得処理

図 5 は、任意の権利定義 (RD) 内の構成要素 (e) を取得する処理の擬似コードである。処理を行なうにあたって、プログラムは権利定義 DB から処理対象となる権利定義の XML 文書を取得し、RightDef クラス

^{*} SHA-1 や MD5 などの一方向性関数によって求められる権利定義のハッシュ値。

のインスタンスを生成後、引数に取得対象要素名 (e) を指定して `getElementValue()` メソッドを呼び出す。以下に、`getElementValue()` の処理の流れを示す。

- (1) 構成要素 (e) が、`RightDef` インスタンス内の内部形式 (`internal_data`) に含まれていれば、 e に対応する値を呼出元に返却して終了する。
- (2) 上位権利定義リスト P の各要素 (p_i) に対して以下の処理を行なう。ただし、 P が空か、未処理の要素がない場合は、呼出元にエラーを返却して終了する。
- (3) p_i に対応する権利定義の XML 文書 (xml) を権利定義 DB から取得する。
- (4) xml を引数として `RightDef` クラスのインスタンス (rd) を生成する (XML 文書が内部形式に変換される)。
- (5) rd の継承制約を RD が満たしていることを検証し、満たしていなければ (2) に戻る。
- (6) rd の `getElementValue()` メソッドを、引数に e を指定して呼出し、値を取得できた場合は、呼出元に返却して終了する。取得できない場合は (2) に戻る。

以上の手順により、処理の呼出元には、権利定義の構成要素の内容が返却される[☆]。

なお、内部形式 (`internal_data`) には、システムの実装方式と相性が良く、権利定義 (XML 文書) と比較して処理コストが小さい形式を利用する。例えば、DOM^[6] 等がこのような形式に該当する。

また、上記の処理中の内部形式をメモリ内に格納すれば、以降は、同一の権利定義への内容取得処理の際に、メモリ内の内部形式を繰り返し利用することによって、権利定義の解釈処理 (内部形式への変換処理) を省くことができる。

4. 考 察

本章では、本稿の提案方式について、効率性、安全性の観点から考察を行なう。また、提案方式とオブジェクト指向の対応関係についても述べる。

4.1 効 率 性

本方式では、権利定義の差分記述によって、権利の特殊化、拡張、複数の権利定義の継承が可能であり、新規に権利定義を記述する場合と比較して、効率的な

[☆] 実装時には、継承制約の効率的な処理、エラー処理、多重継承時の名前衝突の解決など、多くの課題が考えられるが、本稿の議論からは割愛する。

```
class RightDef {
    :
    内部形式 internal_data;

    RightDef(XML 文書 xml) {
        internal_data = parse(xml); //内部形式に変換
    }

    public Value getElementValue(e) {

        if (構成要素 e が存在する) {
            return (e の値);
        } else {
            foreach p (上位権利定義リスト) {
                XML 文書 xml = 権利定義 DB.get(p); //定義取得
                RightDef rd = new RightDef(xml); //パース
                //継承制約の検証
                if (verify(rd) == true) {
                    Value v = rd.getElementValue(e); //値の取得
                    if (v != null) {
                        return v;
                    }
                }
            }
            return null;
        }
    }
    :
}
```

図 5 内容取得処理擬似コード

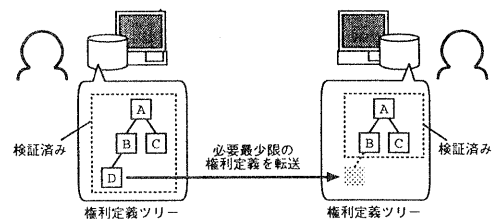


図 6 権利定義の転送

記述が可能である。

権利定義の階層関係の中で、使用頻度が高い権利定義については、権利定義、およびその処理結果 (内部形式) を繰り返し用いることによって、処理効率を向上させることができる。これは、図 6 のように、既に、転送、および検証済みの権利定義については、処理を省略することができるため、権利定義の転送コスト、正当性検証コスト、格納コスト、XML 文書から内部形式への変換コストを必要最小限に抑えられることに起因する。この効率化方法は、座席番号のみが異なるコンサートチケット、航空券のように、類似した権利定義を大量に処理する際に、特に有効である。

また、サービスの管理効率の向上については、第 3.1 節の例で述べたように、権利定義のグループ化による動的なサービス対象の追加によって対応可能である。

以上のように、権利定義の階層関係の活用によって、記述効率や処理効率の向上を図ることが可能である。

4.2 安 全 性

本方式では、効率向上を目的とした権利定義の継承機構の導入に伴ない、オープンなネットワークにおける継承関係の保護機構が必要となった。この保護機構に対しては、第3.2節で述べた継承規則の設定、ダイジェスト値による継承対象の指定方式が該当する。

これらの機構によって、上位の権利定義に対する下位の権利定義による不正な内容変更、上位の権利定義の改竄、偽造が検出可能であり、予想外の変更による悪影響から下位の権利定義を保護できる。すなわち、上位の権利定義の記述者、下位の権利定義の記述者の双方にとっての安全性を向上できる。

継承対象が改竄された場合は、図2のようにダイジェスト値が一致しないことから改竄の検出が可能であり、階層関係の最下層の権利定義を、第5.4節で述べる原本性保証技術によって保護するのみで良い。

継承対象の偽造については、広く公開されるダイジェスト値や安全な通信路によって信用できる配布者から取得するダイジェスト値を利用することによって、不正な権利定義を検出することができる。

また、電子データの改竄を防止するためには、電子署名を用いる方法が一般的であるが、本稿の方式ではダイジェスト値の比較による方法を探っている。ダイジェスト値を用いる方法は、電子署名を用いる方法に比べて、計算コストが低く、権利定義のデータ量も減少するため処理効率が良い。

以上のように、本稿の方式では、権利定義の処理効率の低下を抑えながらも、権利定義の不正な継承の検出、改竄や偽造の防止が可能である。

4.3 オブジェクト指向との対応関係

本方式では、権利定義の再利用性を高めるために、オブジェクト指向の継承機構に着目した。

オブジェクト指向における差分記述は、権利定義においては上位の権利定義に対する、下位の権利定義による権利内容の差分追加に対応する。差分追加の際に、権利内容に対する制約を追加する場合は権利の特殊化(is-a関係)となり、新たな権利要素を追加する(特性を追加する)場合は権利の拡張となる。

本方式では、継承を複数の権利定義に対して行なうことを許しており、権利定義の複合化を可能にしている。これは、多重継承に対応する。

上位の権利定義の記述者による継承規則の設定は、抽象宣言、定数宣言などに対応するが、ダイジェスト値による継承対象の安全な指定方法は、通常のオブジェクト指向には存在せず、継承関係がネットワーク

を経由し得る本方式に特有の概念である。

本稿では、権利定義の階層関係を「継承」と呼んでいる。しかし、権利定義における差分記述は、SmallTalk、C++、Javaのようなクラスベースのオブジェクト指向よりも、Actorモデル、Selfのようなプロトタイプベースのオブジェクト指向として捉える方が相応しい。すなわち、権利定義の差分記述はクラスの継承ではなく、プロトタイプベースのオブジェクト指向における権利定義の内容取得の委託(delegation)と考えるべきである。

5. 関 連 研 究

本章では、本稿の提案方式の関連研究として、アクセス制御、ポリシー記述言語、XML文書のスキーマ定義言語におけるオブジェクト指向の利用例について述べる。また、本方式と関連が深い原本性保証技術についても述べる。

5.1 アクセス制御

権限認証のための権限記述の規格として、IETFで標準化が進められている属性証明書がある。^[7]

この仕様は、X.509のID証明書の形式から、公開鍵情報を除き、属性記述のための領域を追加した形式であり、構造はASN.1に従う。汎用的な属性記述の領域を設けているため、汎用的に情報を格納できるという意味では、多様性の要件を満たしている。属性証明書は、現在、広く普及しつつあるX.509のID証明書による認証機構との相性が良く、システムの実現コストを低く抑えることができる。

本稿の方式が、権利定義の集中管理を前提としていないことに対して、属性証明書によるアクセス制御は、一般的には、認証局、属性証明局を中心とした集中管理型のシステム構成となり、適用領域が限られる。また、本方式のような属性証明書間に階層関係を付与することは考慮されていないこと、および電子署名が必要であることなどから、処理効率、可読性の面では、本稿の方式が有効である。

5.2 ポリシー記述言語

分散システムの管理の分野で、ネットワーク管理のためのポリシー記述に階層構造やオブジェクト指向を利用する考え方が古くから議論されている。^[8]これは、ネットワークの構成が階層関係をとることに起因する。

Ponder^[9]は、ネットワークの管理者や利用者の役割(role)と管理対象ノードを階層構造で管理し、個々の役割や管理対象ノードに対する、権限付与、義務付け、禁止、委譲許可の設定ポリシーをオブジェクト指

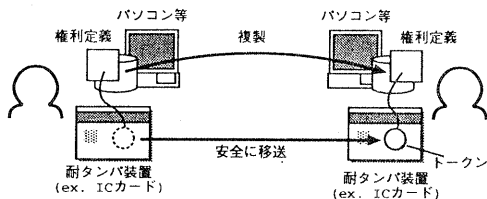


図7 FlexTokenによる権利流通

向の記述言語によって定義する。

記述形式にオブジェクト指向を用いる点は、本稿の方式と類似している。ただし、現状では、継承が記述の差分追加以上の意味は持たないこと、ポリシーの記述者は単一もしくは相互に信用可能であることを前提とすることなどが、本稿の方式と異なる。

5.3 スキーマ定義言語におけるオブジェクト指向

本稿の方式では、権利定義のシンタックスにXMLを利用している。XMLのスキーマ定義言語の分野では、XML文書の構造の規定にオブジェクト指向を利用することが検討されており、そのようなスキーマ定義言語の仕様として、SOX^[10]、XML Schema^[11]などがある。また、XMLをシンタックスとした汎用メタデータ記述言語であるRDF^[12]についても、RDFデータのスキーマ定義言語にオブジェクト指向を導入したRDF Schema^[13]がある。

これらの仕様の多くは、本方式と同様に、差分追加の機構を含んでおり、また、SOXにおいては多相性(polymorphism)など、オブジェクト指向を積極的に導入している。しかしながら、XML文書の場合、文書構造には順序性があるため、これを階層関係のみのオブジェクト指向の継承の概念と矛盾無く結合することは難しい。

一方、本稿の方式では、汎用的なXML文書の構造の規定を目的とするXMLスキーマ定義言語と異なり、差分定義の追加パターンを絞り込むことによって、上記のような権利定義の継承時に生じる木構造の合成方法の曖昧性を解消することができる。

5.4 原本性保証技術

権利の発行、譲渡、改札といった流通処理の際には、電子データの原本性を保証し、不正コピーを防止する必要がある。本稿の方式は、多様な権利内容を定義することが目的であり、第4.2節で述べたように、既存の不正コピーの防止技術を利用して、継承関係の最下層の権利定義のみを保護することによって、権利流通システムを構築することが可能である。

既存の原本性保証技術としては、安全なサーバを利

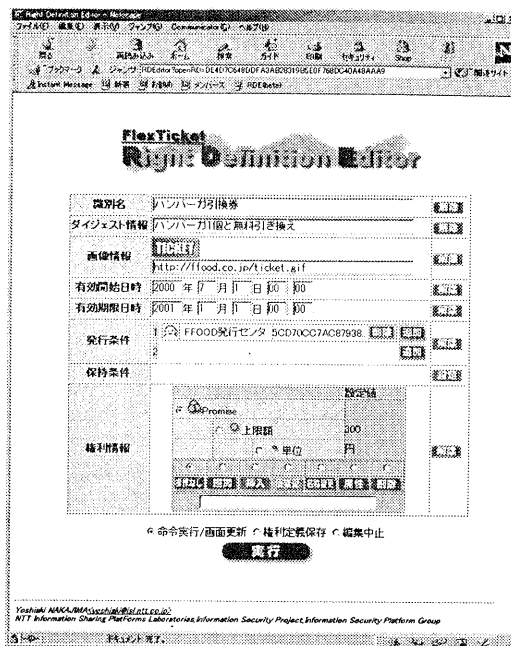


図8 権利定義エディタ

用する方法^[3]、FlexToken^[2]のような耐タンパ装置を利用する方法などがある。

FlexTokenは、図7に示すように、原本性の保証対象となる情報は、通常の格納媒体(ハードディスク等)に格納し、この情報に対応付けられるトークンと呼ばれる情報を耐タンパ装置に封入する。トークンは、耐タンパ装置の機能によって複製から保護され、また、トークンが持つ情報と保証対象の情報と照合することによって、原本性保証対象の改変も検出することができる。XML TicketとFlexTokenを組み合わせることによって、権利の発行は、権利定義に対応するトークンの生成、権利の消費はトークンの削除、権利の譲渡はトークンの移送によって実現される。すなわち、権利定義の役割は権利内容の表現のみとなり、権利の実体はトークンで表される。

なお、筆者らは、FlexTokenとXML Ticketを組み合わせることによって、インターネットのように参加者が相互に信用できない環境下での安全な権利流通を可能にするシステムを開発している。^{[14][15]}

6. まとめ

本稿では、多様な権利を汎用的に表現可能、かつ装置やプログラム等によって機械的に処理可能な共通形式として、権利内容を階層的に定義可能な方式を提案した。また、本方式が、権利記述、サービス構築・管

理, 処理性能などを効率化し, 階層関係に継承規則を付与することによって, 権利定義の安全な検証を可能にすることを示した.

今後は, 本稿で述べた方式の有効性を検証するために, 上記の権利流通システムにおける権利定義関連の処理性能の定量評価, 図 8 に示す権利定義記述用アプリケーションを利用した権利定義の記述実験などを通じて課題分析を行ない, 言語仕様への反映, および効率の良い実装方式の検討を行なう予定である.

参 考 文 献

- [1] Fujimura, K. and Nakajima, Y., General-purpose Digital Ticket Framework, *Proceedings of the 3rd USENIX Workshop on Electronic Commerce* (1998).
- [2] 寺田雅之, 久野浩, 花館蔵之, 権利流通基盤実現のための原本性保証方式, コンピュータセキュリティシンポジウム'99 予稿集 (1999).
- [3] Matsuyama, K. and Fujimura, K., Distributed Digital-Ticket Management for Rights Trading System, *Proceedings of the 1st ACM Conference on Electronic Commerce*, ACM (1999).
- [4] 中嶋良彰, 藤村考, 関根純, 権利流通基盤実現のための権利情報定義言語, コンピュータセキュリティシンポジウム'99 予稿集 (1999).
- [5] W3C, *Extensible Markup Language (XML) 1.0 W3C Recommendation* (1998).
- [6] W3C, *Document Object Model (DOM) Level 2 Specification Version 1.0 W3C Candidate Recommendation* (2000).
- [7] Farrell, S. and Chadwick, D., Limited AttributeCertificate Acquisition Protocol, *Internet Draft*, IETF (1999).
- [8] Moffett, J. D., Policy Hierarchies for Distributed Systems Management, *IEEE Journal on Selected Areas in Communications*, Vol 11, No.9 (1993).
- [9] Damianou, N., Dulay, N., Lupu, E. and Sloman, M., Ponder: A Language for Specifying Security and Management Policies for Distributed Systems, *The Language Specification Version 2.1, Imperial College Research Report DoC 2000/1* (2000).
- [10] W3C, *Schema for Object-Oriented XML 2.0 W3C Note* (1999).
- [11] W3C, *XML Schema Part1: Structures and Part2: Datatypes* (1999).
- [12] W3C, *Resource Description Framework (RDF) Model and Syntax Specification W3C Recommendation* (1999).
- [13] W3C, *Resource Description Framework (RDF) Schema Specification 1.0 W3C Candidate Recommendation* (2000).
- [14] 水野康尚, 千綿伸之, 大嶋嘉人, 松山一雄, 権利流通基盤実現のための汎用デジタルチケットシステム, コンピュータセキュリティシンポジウム'99 予稿集 (1999).
- [15] NTT, FlexTicket Official Home Page.
<http://info.isl.ntt.co.jp/flexticket/>.