

XrML 2.0 Technical Overview

Version 1.0

March 8, 2002



Table of Contents

Table of Contents	2
1.0 Introduction	3
2.0 XrML 2.0 Data model	4
2.1 <i>Principal</i>	4
2.2 <i>Right</i>	5
2.3 <i>Resource</i>	5
2.4 <i>Condition</i>	5
2.5 <i>Data Model is Encapsulated in XML Schema</i>	6
3.0 XrML Basic Data Constructs	7
3.1 <i>License</i>	7
The license issuer.....	7
3.2 <i>Grant</i>	8
4.0 Structure and Organization of the Language	9
4.1 <i>Mandatory vs. Optional terms</i>	10
Example of a minimal license	11
4.2 <i>Core Schema</i>	11
Overview of Core Schema.....	12
4.3 <i>Standard Extension</i>	14
Overview of Standard Extension	14
4.4 <i>Content Extension</i>	17
Overview of Content Extension	18
5.0 System Features of the Language	20
5.1 <i>Trust Specification</i>	20
5.2 <i>Confidentiality</i>	22
5.3 <i>Web Service Specification</i>	22
5.4 <i>Pattern Matching</i>	23
6.0 Summary	24

1.0 Introduction

XrML is a language to specify rights. XrML is an XML-based usage grammar for specifying rights and conditions to control the access to digital content and services. XrML had its roots in Xerox Palo Alto Research Center. Digital Property Rights Language (DPRL) was first introduced in 1996. DPRL became XrML when the meta-language (used to construct the language) was changed from a lisp-style meta-language to XML in 1999.

Using XrML, anyone owning or distributing digital resources (such as content, services, or software applications) can identify the parties allowed to use those resources, the rights available to those parties, and the terms and conditions under which those rights may be exercised. These four elements are the Core of the language and determine the full context of the rights that are specified. In other words, it is not sufficient to just specify that the right to view certain content has been granted, but also *who* can view it and under *what* conditions.

Since its inception, the language has evolved through industry feedback, critical review, and product implementation. The language has become *comprehensive* by providing a framework to express rights at different stages of a workflow or lifecycle, *generic* by defining a large body of format and business neutral terms (about 100) and using these terms to specify rights to any digital content and service, and *precise* through the development of a grammar and processing rules that enable unique interpretation of the language. XrML is by far the most advanced and mature rights language in use today. Since 1999, the emphasis has been to get the language implemented in real life systems. This experience has resulted in additional system-related features (trust, for example) that are now part of the language. The current version of the language is 2.0.

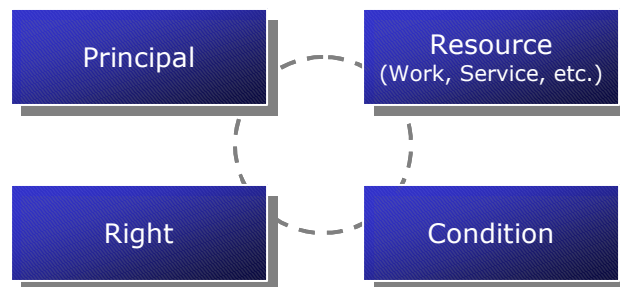
The purpose of this paper is to present the technical details of the language. A more general introduction to the language can be found in the *Need for a Rights Language* white paper and the definitive manual for the language can be found in the *XrML 2.0 Language Specification*.

2.0 XrML 2.0 Data model

XrML 2.0 adopts a simple and extensible data model for many of its key concepts and elements. The XrML data model consists of four entities and the relationship between those entities.

The basic relationship is defined by the XrML assertion "grant". Structurally, an XrML grant consists of the following:

- The principal to whom the grant is issued
- The right that the grant specifies
- The resource that is the direct object of the "right" verb
- The condition that must be met for the right to be exercised



2.1 Principal

A principal encapsulates the identification of a party to whom rights are granted. Each principal identifies exactly one party. In contrast, a set of principals, such as the universe of everyone, is not a principal.¹

A principal denotes the party that it identifies by information unique to that party. Usefully, this information has some associated authentication mechanism by which the principal can prove its identity. The Principal type supports the following identification technologies:

- A keyHolder, meaning someone identified as possessing a secret key such as the private key of a public/private key pair. KeyHolders are represented using the KeyInfo technology from XML DSIG.
- A principal that must present multiple credentials, all of which must be simultaneously valid, to be authenticated.
- Other identification technologies that may be invented by others.

¹ Through the mechanisms of variable definition and pattern matching, XrML can define the principal as "any one in the universe". However, during the interpretation of the rights expression, the principal must be resolved to a single party--a specific entity or person in "the universe".

2.2 Right

A right is the "verb" that a principal can be granted to exercise against some resource under some condition. Typically, a right specifies an action (or activity) or a class of actions that a principal may perform on or using the associated resource.

The XrML 2.0 Core provides an abstract right element to encapsulate information about rights. It also provides a set of commonly used, specific rights relating to other rights, such as issue, revoke, and obtain. Extensions to the XrML Core define rights that appropriate to using specific types of resources. For instance, the XrML Content Extension defines rights appropriate to using digital works (for instance, play and print rights).

2.3 Resource

A resource is the "object" to which a principal can be granted a right. A resource can be a digital work (such as an e-book, an audio or video file, or an image), a service (such as an email service, or B2B transaction service), or even a piece of information that can be owned by a principal (such as a name or an email address).

The XrML 2.0 Core provides mechanisms to encapsulate the information necessary to identify and assign rights to a particular resource. Patterns allow identification of a collection of resources with some common characteristics. Extensions to the XrML Core define resources appropriate to specific business models.

2.4 Condition

A condition specifies the terms, conditions, and obligations under which rights can be exercised. A simple condition is a time interval within which a right can be exercised. A slightly more complicated condition is to require the existence of a valid, prerequisite right that has been issued by some trusted entity. Using this mechanism, the eligibility to exercise one right can become dependent on the eligibility to exercise other rights. Moreover, a list of conditions can be put in conjunction to form a condition requiring that the conditions all be met simultaneously.

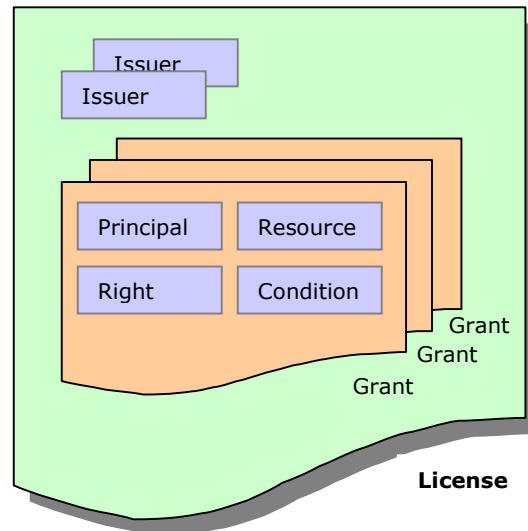
The XrML 2.0 Core defines an abstract condition element to encapsulate information about conditions and some very basic conditions. Extensions to the XrML Core could define conditions appropriate to specific distribution models. For instance, the XrML Content Extension defines conditions appropriate to using digital works (for instance, watermark, destination, and renderer).

2.5 Data Model is Encapsulated in XML Schema

Since XrML is defined using the XML Schema recommendation from W3C, its element model follows the standard one that relates its elements to other classes of elements. For example, the "grant" element is related to its child elements, "principal", "right", "resource", and "condition".

3.0 XrML Basic Data Constructs

As indicated in the previous section, the basic XrML construct is the assertion "grant". In a system environment, there is the additional need to determine other things such as the identification of the principal who issued the grant and the identification of the license. For this reason, in a system environment, the central XrML construct is a "license". Conceptually, a license is the issuance of grants by their issuing parties.



3.1 License

The basic structure of a license contains the following:

- A set of grants that convey to certain principals certain rights to certain resources under certain conditions
- An identification of the principal or principals who issued the license and thus bestow the grants upon their recipients
- Additional information such as a description of the license and validity date

Those familiar with digital certificates and other similar structures might notice the absence of the identification of the principal or principals to whom certain rights are conveyed. This notion has been regularized within the structure and terms of each individual grant.

The license issuer

The principal who issues a license can digitally sign it, signifying that the issuer does indeed bestow the grants contained in it. Syntactically, multiple issuers may sign a given license. However, no additional semantic is associated with the joint signing; it is as if each had signed a copy of the license independently.

3.2 Grant

A grant is the element within the license that bestows an authorization upon some principal. It conveys to a particular principal the sanction to exercise an identified right against an identified resource, possibly subject to first fulfilling some conditions.

Structurally, a grant consists of the following:

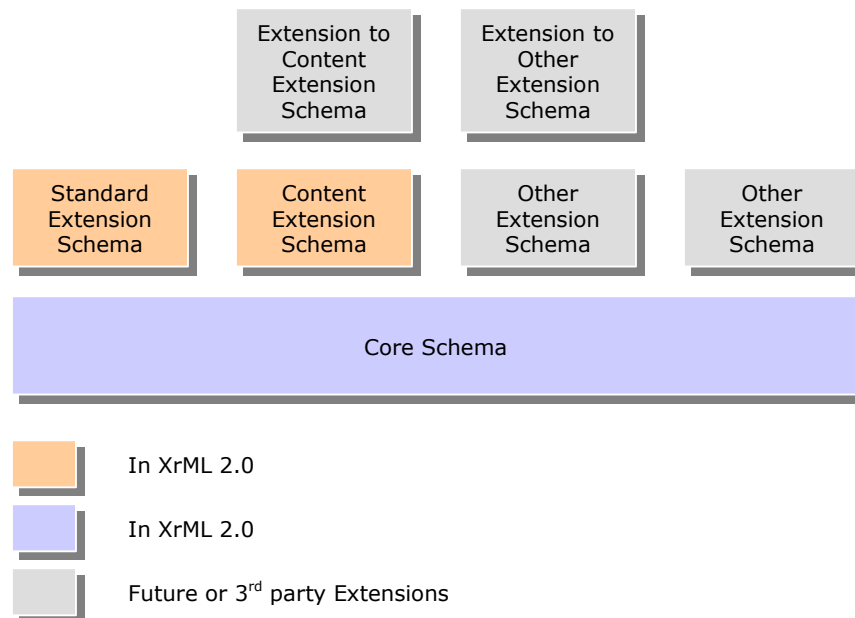
- The principal to whom the grant is issued
- The right that the grant conveys to the specified principal
- The resource against which the specified principal can exercise or carry out this right
- The condition that must be met before the right can be exercised

4.0 Structure and Organization of the Language

The syntax of XrML 2.0 is described and defined using the XML Schema technology defined by the Worldwide Web Consortium (W3C). Significantly more powerful and expressive than DTD technology, the extensive use of XML Schema enables XrML 2.0 to offer a high degree of richness and flexibility in its expressiveness and extensibility.

To that end, a principal design goal for XrML 2.0 is to enable and support a significant amount of extensibility and customizability without requiring actual changes to the XrML 2.0 Core. Indeed, XrML 2.0 makes use of this extensibility internally, even within the Core itself.

XrML 2.0 is organized into several parts:



- A Core Schema containing definitions of concepts that are at the heart of the XrML 2.0 semantics, particularly those having to do with evaluation of a trust decision
- A Standard Extension schema containing definitions of concepts that are generally and broadly useful and applicable to XrML 2.0 usage scenarios, but which aren't necessarily at the heart of XrML 2.0 semantics
- Other, domain-specific extensions, including a content management schema (XrML 2.0 Content Extension) that defines rights management concepts specifically related to digital works such as books, music, and video.

Each of these XML Schemas is a normative part of the overall XrML 2.0 specification. In particular, the Core Schema is a normative part of the XrML 2.0 Core. Other parties may, if they wish, define their own extensions to XrML 2.0. This is accomplished using existing, standard XML Schema and XML Namespace mechanisms.

4.1 Mandatory vs. Optional terms

Most of the terms defined by the language are, in fact, optional. Only certain elements are mandatory in order to produce a valid XrML license conveying at least one grant². The schema clearly indicates this property, and in the schema overview presented in this paper, the mandatory elements will be indicated and explained below. The use of mandatory and optional terms allows the creation of both simple and complex expressions. Optional terms are used as needed.

The set of mandatory elements is organized as follows:

- **License**
- **Grant** or **GrantGroup**
- **Right (abstract)** or one of its substitutions

Right is abstract, and it is replaced by Issue, Obtain, PossessProperty, Revoke or any right defined in an extension (for example the Play right defined in the Content Extension).

All the mandatory terms for defining a License are part of the Core Schema. All the terms in the Standard and Content Extensions are optional

The notion of the mandatory set within the Core is that of a **license granting a right**. When the right used is one of those in the Core, then the notion is that of establishing an identity and/or establishing certain rights to manipulate rights

PossessProperty is used to establish the digital identity of a principal. Precisely identifying a principal is the key element for the management of digital rights.

Rights (and conditions) are industry specific, for example a digital movie is played but cannot be printed. An e-book can be viewed and printed. Resources are also industry specific: a digital movie, a picture, a text manuscript, a file system, etc.

² If one is not constructing an **XrML License**, then there are no mandatory terms. In other words, one can create a valid XrML expression with any terms defined in the language.

Example of a minimal license

A simple and minimal license can be constructed using the mandatory terms and a couple of optional terms. The following license certifies that the holder of the (cryptographic) key has the common name "Alice Richardson"

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Copyright (C) 2001 ContentGuard Holdings, Inc. All rights reserved.
"ContentGuard" is a registered trademark and "XrML", "eXtensible rights Markup
Language", the XrML logo and the ContentGuard logo are trademarks of
ContentGuard Holdings, Inc. All other trademarks are properties of their
respective owners.-->
<!-- This is a simple certificate -->
<license xmlns="http://www.xrml.org/schema/2001/11/xrml2core"
  xmlns:sx="http://www.xrml.org/schema/2001/11/xrml2sx"
  xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.xrml.org/schema/2001/11/xrml2cx..\schemas\xr
ml2cx.xsd">
  <!-- Certify that the following key holder has the common name "Alice
Richardson"-->
  <grant>
    <keyHolder>
      <info>
        <dsig:KeyValue>
          <dsig:RSAKeyValue>
            <dsig:Modulus>Fa7wo6NYfmvGqy4ACSWcNmuQfbejSZx
7aCibIgkYswUeTCrms0h27GJrA15SS7TYZzsfas0xR91Z
dUEF0Th04w==</dsig:Modulus>
            <dsig:Exponent>AQABAA==</dsig:Exponent>
          </dsig:RSAKeyValue>
        </dsig:KeyValue>
      </info>
    </keyHolder>
    <possessProperty />
    <sx:commonName>Alice Richardson</sx:commonName>
  </grant>
</license>
```

As indicated above, the starting point or the basic premise for the management of Digital Rights is establishing an identity. The expression of most rights depend on the type of industry. Note that the above example has no issuer. Although, an issuer may be required in most applications, the example is a valid XrML license.

4.2 Core Schema

At the heart of XrML 2.0 is the Core Schema. The elements and types in the Core Schema define structural and validation semantics that comprise the essence of the specification. XrML 2.0 core concepts include license, grant, principal, right, resource and condition. The XrML Core defines elements for the last four of these concepts in an abstract fashion. Extensions to the XrML Core could extend these elements to address specific formats and business applications.

Overview of Core Schema

XrML 2.0 Core Schema Overview		
Parent Element	Element	Definition
license	-	Issuance of Grants by Issuers (Container of Grants)
	title	Descriptive phrase about the License that is intended for human consumption in user interfaces and the like.
	grant grantGroup	The grant and grantGroup elements contained in a License are the means by which authorization policies are conveyed in the XrML 2.0 architecture.
	issuer	The issuer element in a License contains two pieces of information: a set of issuer-specific details about the circumstances under which he issues the License (dates, revocation mechanisms, etc), and a digital signature for the License.
	inventory	Inventory provides a syntactic mechanism for reducing redundancy and verbosity in Licenses. This syntactic macro-like mechanism can be used throughout a License
	(any)	Using the wildcard construct from XML Schema, a License provides an extensibility hook within which License issuers may place additional content as they find appropriate and convenient, such as information related to authentication and authorization, but not part of the XrML 2.0 core infrastructure.
	encryptedLicense	Provides mechanism by which the contents of a License may be encrypted and so hidden from view from inappropriate parties.
grant	-	A grant is an XML structure that expresses an assertion that some Principal may exercise some Right against some Resource, subject, possibly, to some Condition. This structure is at the heart of the rights management and authorization policy semantics that XrML 2.0 is designed to express.
	forAll	The forAll element defines a variable that identifies members of a group or set and can be used by any of the elements in a grant. For example "everyone in the ABC Music Club". The set defined in the forAll element can be identified with the mechanism of pattern matching.
	principal	The principal element is abstract. Typically, an element that is substitutable with principal is used. For example "keyholder" or "allPrincipals".
	right	The right element is abstract. Typically an element that is substitutable with right is used. For example, any of the rights defined in the Content Extension, such as "play", "print", "copy" and "edit".
	resource	The resource element is abstract. Typically, an element that is substitutable with resource is used. For example, "digitalResource" as defined in the XrML Core.
	condition	
	delegationControl	Whenever a grant is issued, the issuer may optionally indicate that the grant may be delegated to others. Policies that control the circumstances under which the grant can be delegable are expressed by the semantics embodied in the delegationControl

XrML 2.0 Core Schema Overview		
Parent Element	Element	Definition
		element (for example, a <i>destination principal</i>).
	encryptedGrant	encryptedGrant indicates that the grant is encrypted and so hidden from view from inappropriate parties. This mechanism makes straightforward use of the XML Encryption Syntax and Processing standard.
grantGroup		Similar to grant, grantGroup can be associated with all the child elements of grant, but can include other grants or grantGroups as well. Useful when issuing several grants as a group where the grants have common elements (for example granting several rights but with a single condition to a single principal).
principal	allPrincipal	The allPrincipals element groups a set of principals acting together as one holistic identified entity.
	keyholder	Represents a principal that is identified by its possession of a certain cryptographic key, such as a private key (in a PKI environment).
right	issue	The issue element grants the right to issue licenses. This element can be used in multitier models where a distributor is given the authority to issue certain rights. It can also be used to specify peer-to-peer distribution where a peer is allowed to issue a license with specific rights to another peer.
	revoke	When a principal exercises the right to revoke, the authorization given by the (digital) signature is retracted. Issuing principals have an implicit right to revoke their own signatures. The revoke right explicitly grants the right to revoke a signature to others. This right is useful in an environment where a 3 rd party is responsible to oversee the compliance of rights and has been given the authority to revoke (the signature authorizing) those rights.
	possessProperty	The possessProperty element grants the right (to a principal) to claim ownership of property-like characteristics. For example, membership to a specific organization can be specified (and granted) with a license that contains the right possessProperty with a value "Member of IEEE Electron Society". The holder of the license can "claim" that he/she is a member.
	obtain	The obtain element represents the right to obtain rights. Typically, the use of the obtain right can be conceptualized as an "offer" or "advertisement" by the issuer of the grant g to, for example, "sell" the grant g'.
resource	digitalResource	Identifies the digital item by determining its location, either within the element itself or in an external file or web site location.
condition	allConditions	Indicates that every one of the conditions present within the allConditions condition must be satisfied in order for the allConditions condition to be satisfied.
	validityInterval	Indicates a contiguous, unbroken interval of time.
	revocationFreshness	Indicates the upper bound of a time interval by which an XrML enforcement mechanism must check for the validity of the authorization signature (check if the signature has been revoked).
	existRight	A condition that specifies a right that the principal must have

XrML 2.0 Core Schema Overview		
Parent Element	Element	Definition
		(valid or not) in order to exercise a right.
	prerequisiteRight	A condition that specifies a valid right that the principal must have (in the same or a separate license) in order to exercise a right.
Other Core Types and Elements		
aAny	trustedPrincipal	the trustedPrincipal element indicates a policy by which principals are identified as having the appropriate and necessary qualifications in order to be trusted for use in certain situations.
any	serviceReference	Used to specify Web Services. For example, to specify a web service where state information is stored and updated during the exercise of a right (i.e., a count on a per-view movie).
N/A	licenseGroup	Containers of licenses. Useful when several licenses are related in some fashion, such as a license that has authorized the issuance of a license and the license itself.

4.3 Standard Extension

The Standard Extension schema defines terms to extend the usability of the Core Schema. In particular, it extends the condition type in the XrML Core with additional terms to support the notion of external services required to exercise a right (for example, usage tracking), payment conditions and methods, and time conditions

Overview of Standard Extension

XrML 2.0 Standard Extension Schema Overview (all elements are optional)		
Parent Element	Element	Definition
grant	statefulCondition	Some conditions may be tied to an external authoritative piece of data. These arise when the exercise of a right is predicated upon querying or manipulating the value of this external piece of data. For example, the number of times some content may be rendered can be bounded by some value. the statefulcondition element covers such usages. It includes a child element, stateReference, which indicates the means with which communication is made with the state or service.
statefulCondition	stateReference	Indicates the means by which the state is to be queried or manipulated. It has type ServiceReference as defined in the XrML Core.
grant	stateReference-valuePattern	Pattern used to specify values for state information.
	exerciseLimit	Indicates the number of times the associated right can be executed (for example. the number of times a song can be

XrML 2.0 Standard Extension Schema Overview (all elements are optional)		
Parent Element	Element	Definition
		played).
	seekApproval	Indicates that the exercise of a right requires approval from a service. A stateReference element is used to communicate with the state or service.
	trackReport	Indicates that the exercise of a right is to be reported back to a state. A stateReference element is used to communicate with the state or service.
	trackQuery	Indicates a condition on a specific value that a state (of another condition) holds. The trackQuery element is commonly used together with trackReport to model the exercise of rights predicated on the exercise of other rights. For example, Alice may listen to a piece of music as many times as she pleases provided she has listened to some commercial. The grant related to the commercial has a trackReport condition. When Alice attempts to listen to the piece of music, the trackQuery condition allows exercise of the right only when the state value tracked by the trackReport condition has a value greater than zero.
	validityIntervalFloating	Indicates the length of time that a right may be exercised. For example, a grant can provide a right that can be exercised for one week after the first use.
	validityTimeMetered	Indicates the (non-contiguous) time allowed for the exercise of the associated right. Unlike the validityIntervalFloating condition, the length in question only takes into account the periods of time of actual exercise. Includes the child element quantum, which indicates the period by which the metered time is measured.
	validityTimePeriod	Indicates that the associated right exercisable on a periodic basis. For example, "every weekend after Jan 1 2001 for a total of ten times". Include the following child elements: start, period, phase, duration and period count. T these elements are defined in the XrML 2.0 specification.
	fee	Represents conditions related to payment methods. A grant can predicate that a fee be paid before a right can be exercised. Typically, fees involve a payment and a designation of the party to whom payment is made. Fee includes the following child elements: paymentAbstract, min (minimum amount due), max (maximum amount due) and to (to whom payment is made).
fee	paymentAbstract	This abstract element is substitutable by the payment extension (described later in this table). The payment extension represent more concrete forms of payment.
grant	territory	Specifies conditions that predicate rights be exercisable only in certain locations. These locations may correspond to physical or geographical regions, or they may correspond to virtual or digital locations. Child elements are location (/region, /county, /state, /city, /postalCode, /street) and domain (/url).

Payment Extensions		
fee	paymentAbstract	This abstract element is substitutable by the more concrete forms of payment described below.
	cash	Type of payment involved.
	paymentFlat	Indicates a flat fee. Child elements are rate (amount) and paymentRecord (value indicating if payment has been made).
	paymentMetered	Indicates that payment is of a fee that is prorated according to the duration of usage. For example, one may have the right to play a game and pay a fee dictated by the length of time one plays the game. Child elements are rate (amount per period), per (period), by (period quantum to be used to compute amount), and phase (grace period/rounding until next calculation).
	paymentPerInterval	Indicates that the right can be exercised for a period of time that is paid for. For example, one may buy up some time to play a game and may keep playing the game until that time interval is used. Child elements are rate (amount per interval), per (time interval), and paidThrough (the time through which the amount is paid).
	paymentPerUse	Indicates that the right can be exercised upon payment of a fee for each use. For example, one may have the right to listen to a piece of music provided a payment is made for each listening. Child elements are rate (amount per each use or number of uses) and allowPrePay (indicates remaining uses and/or number of initial uses).
	bestPriceUnder	Represents a kind of payment that can be dynamic and is determined when the account is settled. It is used to accommodate special deals, rebates, and pricing that depends on information that is not available to the trusted repository at the time the usage right is exercised, but without communicating with a dealer before the purchase is authorized.
	callForPrice	callForPrice is similar to bestPriceUnder in that it is intended to accommodate cases where prices are dynamic. However, unlike bestPriceUnder, communication with a dealer to determine the price is required before the purchase is authorized; the transaction cannot be completed if the trusted repository is unable to communicate with the dealer.
	markup	Represents a fee that is computed as a percentage of other fees. For example, a distributor may want to add a flat ten percent overhead for selling copies of a digital work, or a government may want to tax sales of a digital works.

Name Extensions: Together with the possessProperty, the resource Name and its extensions allow licenses to straightforwardly express authorized association of names with principals.		
(Anywhere a resource is needed)	emailName	Typically contains a string that designates an internet email address (per rfc822/2822). Licenses can associate the element emailName with principals by using the right possessProperty.
	dnsName	Typically contains a string that designates a domain name. Licenses can associate the element dnsName with principals by using the right possessProperty.
	commonName	Typically contains a string that designates a colloquial name. Licenses can associate the element commonName with principals by using the right possessProperty.
	x509SubjectName	Typically contains a string that designates a subject name from some X509 certificate. Licenses can associate the element x509SubjectName with principals by using the right possessProperty.
	x509SubjectName- Pattern	Pattern that matches an x509SubjectName. Specifically, it matches the root of the x509SubjectName tree. (See Pattern Matching in section 5).
Extension to Revocation		
	revocable	Used with the right revoke. The revocable element identifies the signature (to be revoked) either by its literal value or by indirect means such as its cryptographic hash value.

4.4 Content Extension

The XrML Content Extension is an extension to XrML 2.0 that describes rights, conditions, and metadata for digital works.

The Content Extension expands the Core Schema by specifying terms that relate to digital works. Specifically, the Content Extension expands:

- Rights, by defining terms for render rights, transport rights (governing the movement of content from one repository to another), derivative works rights (governing the reuse of digital works to create new works), file management rights (governing access to directory information in repositories--needed when repositories communicate, make and restore back-up copies) and configuration rights (governing the addition or removal of system software from a repository)
- Resources, by defining terms for digital works and the metadata that describe the digital works
- Conditions related to the exercise of rights for digital works

Overview of Content Extension

XrML 2.0 Content Extension Schema Overview (all elements are optional)		
Type	Element	Definition
Extension to Right: Replaces the abstract type <i>right</i>		
File Management Rights	accessFolderInfo	Represents the right to deliver or reveal information about the works contained within folders.
	backup	Represents the right to make copies of a digitalWork for the purpose of guarding against the loss of the original due to accident or catastrophic media or equipment failure.
	delete	Represents the right to delete the digitalWork in a secured repository. The right to delete must be controlled if many people can log into a secured repository and delete files either accidentally or in malicious mischief.
	execute	Represents the right to execute a resource (for example an application like a word processor) from a secure repository.
	manageFolder	Represents the right to perform the following operations: create subfolders, name subfolders and reconfigure folders by moving files and folders among folders
	read	Represents the right to read the work from the repository.
	restore	Represents the right to restore a work from a backup copy, converting the backup copy to usable form.
	verify	Represents the right to check the authenticity of the work in the repository. Authentication typically involves verifying signatures of signed data using the public/private key pair. Integrity verification ensures that the data received exactly matches the data that was sent; it ensures that the data has not been tampered with.
	write	Represents the right to write or save the work in the secure repository.
Transport Rights	copy	Represents the right to create a new copy of the digitalWork, separate from the original that was copied.
	loan	Represents the right to lend a work to another principal for a specific period of time. While the work is on loan, the original copy cannot be used.
	transfer	Represents the right to transfer a work to another repository, removing the work from the original location.
Derivative Works Rights	edit	Represents the right to make changes to a work to create a new work based on the original work. Edit is like extract in that it creates a new work. It differs from extract in that it confers the right to make changes to the work.
	embed	Represents the right to include the work as part of a composite work. An embed operation places a copy of the work inside the composite work.
	extract	Represents the right to use a portion of the work to create a new work. The extracted material is created as a new work, separate from the original work. Extract differs from edit in that it does not grant the right to modify a work.

XrML 2.0 Content Extension Schema Overview (all elements are optional)		
Type	Element	Definition
Render Rights	export	Represents the right to make a source copy of the work outside of the secure repository. For example, saving a protected (encrypted) work in clear-text. Export differs from copy in that an export is made to an in-the-clear non-secure repository, whereas a copy is made to another secure repository.
	play	Represents the right to render the work in a transient form, as appropriate to the content type (for example displaying a book, playing an audio clip, or showing a video).
	print	Represents the right to make a permanent non-digital rendered copy of the work outside the control of a repository. Exercising this right might result in printing a hard copy of a book or creating an audio recording on a magnetic tape.
Configuration Rights	install	Represents the right to make software executable (installed) on the repository, including, for example, checking that the software is certified, that it has not been tampered with, and that it is compatible with the repository.
	uninstall	Represents the right to disable software from running, restoring it to the state it was in before it was installed.
Extension to Resource		
	digitalWork	A resource that represents the content to which rights and conditions are being applied. Contains the child elements description (human readable description), metadata (referencing of metadata), locator (how to locate the work), and parts (identification of the parts of the work).
	simpleDigitalWork-Metadata	Set of metadata for digital works. Includes the child elements title, creator, publisher, publicationDate, owner and copyright. Note: This is an "arbitrary or minimal" set. Anyone can define a different set.
	securityLevel	An abstract indication of the security level that a principal has. Includes the element value.
Extension to Condition		
	destination	Indicates the repositories to which a work can be moved. Used with rights that involve movement of digital works (all rights except render rights).
	source	Indicates the source secured repository or device to use when exercising a right. Used with all rights except for render rights.
	helper	Indicates the software that can be used to exercise a right. For example, specifying a specific player to show a digital movie
	renderer	Identifies the device that can be used to render a work. Used with render rights.
	watermark	A list of information to be embedded in a copy of the work by a device while producing this copy.

5.0 System Features of the Language

One of the advanced features of XrML is the comprehension of system issues for interoperability. In other words, in a real system, specification of rights is not sufficient for interoperability. Some of the additional issues that must be addressed are:

- Can the systems trust the licenses?
- Can the language be extended without breaking the core specification?
- Can the systems interact with services that are part of, but outside of, the DRM system?

XrML has been designed to address these issues and to enable interoperability of components across DRM systems.

In particular, XrML provides system interoperability elements that are needed to interoperate on a system level. They include:

- **Trust specification:** Ensuring the authenticity of the XrML rights specification through the use and application of open and standard digital signatures technologies (W3C DSig). The language semantics enable protection of any semantically significant construct in the rights expression via open and standard cryptographic mechanisms such as digital signature.
- **Confidentiality:** Ensuring confidentiality of the XrML document by enabling encryption of any semantically significant construct in the rights expression.
- **Web Service Specification:** Supported through WSDL and UDDI.
- **Pattern Matching:** Supported through XPATH (see Section 5.4, Pattern Matching).

5.1 Trust Specification

A trust model is a central element of any system (such as a DRM system) that must determine whether a component can be trusted. When an application receives a rights specification, it must determine not only if it has been tampered with, but also if the issuing entity can be trusted³.

One option in DRM is to require an end-to-end system design, which is bound by a common trust model outside of the language. This leads to a proprietary design with enforcement mechanisms described by the proprietary design and not by the rights language. Such is the case in all current DRM systems.

XrML adopts an explicit trust model approach that allows any number of trust domains to cooperate in the enforcement of rights. The trust specifications in XrML help the different parties determine to what extent, for what purpose, and when to

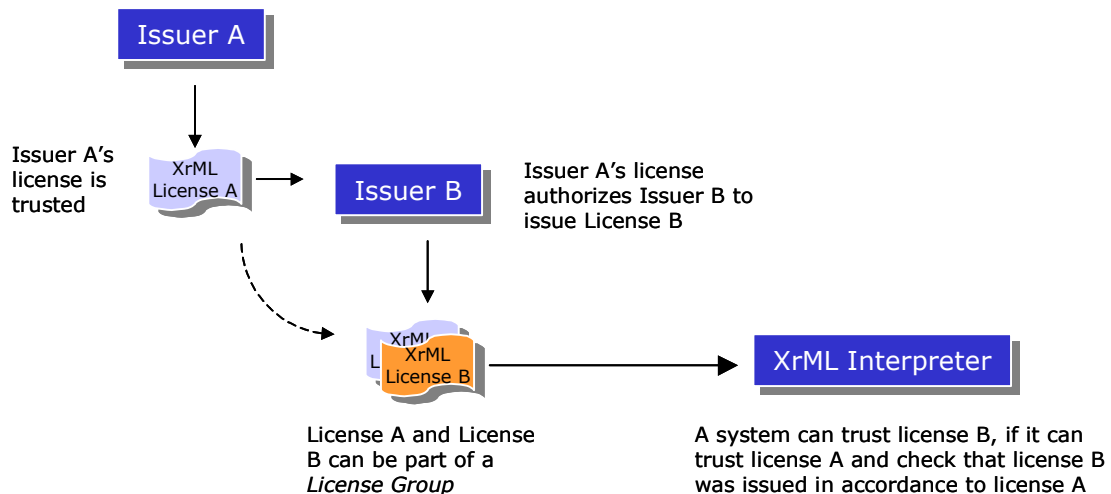
³ A trust model requires security technologies such as encryption, digital signature, tamper detection, and so on

trust other parties. As an end-to-end system design is no longer required, players are free to choose the best design for each part of the system.

XrML defines "issuer" and "trustedPrincipal", and allows the definition of licenses that specify the right to issue other licenses to encapsulate the explicit trust model. The issuer element in a License contains two pieces of information: a set of issuer-specific details about the circumstances under which he issues the License (dates, revocation mechanisms, etc), and a digital signature for the License. With this, the decision point is whether or not a system would trust and validate that signature. TrustedPrincipal indicates a policy by which Principals are identified as having the appropriate and necessary qualifications in order to be trusted for use in certain situations.

In many cases, a system may blindly trust that the issuer's signature is his own (unless it is known that the key has been hacked). However, a system should not blindly trust the issuer's right to issue whatever license has been issued. A system will usually have some sort of check, either:

- The system trusts the issuer itself (and thereby trust everything the issuer signs). This model is typical of a closed-trust system (monolithic trust). All the components work under a single certificate authority and all the signatures are trusted (unless revoked).
- The system sees a license that gives the issuer (issuer #1) the right to issue and the system trusts the second issuer's (issuer #2) right to issue that right to the first issuer (issuer #1). This is what XrML enables a system to check. The check for trust can chain up to a signature (or license) that is trusted by default. In this scenario, an "open-trust" environment is possible, where the trust of a license can be determined (or validated) by checking the licenses that have been used to issue subsequent licenses (see the figure below)
- The system trusts the issuer's right to issue the license by relying on some out-of-band means. For instance, the system knows that "Alice" is the author of a book and so it trusts "Alice" to issue people the right to play the book.



5.2 Confidentiality

Certain business models require that the details about rights and conditions be kept confidential. XrML defines EncryptedLicense and EncryptedGrant for this purpose. EncryptedLicense provides a mechanism by which the contents of a License may be encrypted and so hidden from view from inappropriate parties. EncryptedGrant indicates that the grant is encrypted and so hidden from view from inappropriate parties. This mechanism makes straightforward use of the XML Encryption Syntax and Processing standard and enables systems to understand how each part of a license has been protected through encryption.

5.3 Web Service Specification

Web service specification leverages the serviceReference element in XrML. a serviceReference indicates the location and the means and manner by which a client is to interact with a specific service. Specifically, a serviceReference element does the following:

- Identifies the location or address at which the service is found.
- Identifies a greater or lesser amount of metadata about the semantics of the service and the rules to which the client must adhere to interact with it
- Optionally specifies a set of concrete parameters that are to be provided when a client interacts with the service by dereferencing this particular serviceReference. These parameters provide a means by which a service might at run time distinguish between its uses from different XrML 2.0 contexts.

XrML 2.0 does not itself invent significant new infrastructure for describing services; rather, it draws on the considerable work being done in this area by others. Specifically, there are two architected technologies by which the location and metadata information of a ServiceReference may be provided (using an xsd:any element, ServiceReference provides for other technologies that may also be used):

- the Web Services Definition Language (WSDL)
- the Universal Description, Discovery, and Integration (UDDI) directory infrastructure

5.4 Pattern Matching

XrML uses pattern matching to specify sets of principals, rights, resources and conditions. The general notion of a grant (an XrML construct) is to **one principal** for one right over one resource under one condition (however complex it might be). It is then quite useful and important at times to be able to write in XML formal expressions that semantically denote particular sets of resources or other elements.

XrML defines the following elements to support pattern matching:

XmlPatternAbstract: All formal patterns in XrML 2.0 have types that derive from the type XmlPatternAbstract.

XmlExpression: XmlExpression provides a means by which patterns written in formal expression languages defined outside of XrML 2.0 can be straightforwardly incorporated. The particular expression language used is indicated by the **lang** attribute, which is a URI. The default value for **lang** is <http://www.w3.org/TR/1999/REC-xpath-19991116>, which indicates that the contents of the XmlExpression contains a string which is an XPath expression.

PrincipalPatternAbstract, RightPatternAbstract, ResourcePatternAbstract, and ConditionPatternAbstract: As an alternative to using patterns written in externally-defined expression languages, it is often useful to define new XML types and elements that, in their intrinsic semantic, define some pattern matching algorithm. This can, of course, be done by simply deriving from XmlPatternAbstract, but, in some situations, deriving from one of these four types might be more useful.

6.0 Summary

XrML has been carefully designed to provide a concise yet comprehensive set of elements and a grammar to construct rich expressions for rights. Its simple, yet flexible, data model is practical for both simple and complex expressions for diverse business models: from a simple specification of rights for a user viewing some content, to complex models for multi-tier distribution of content. We believe that XrML has achieved the "right balance" between compactness and comprehensiveness in an XML based language.

Because of its capability to address many different models, XrML is well positioned to be the language for interoperability between components and systems that deal with rights and conditions (such as DRM systems, Content Management Systems, Licensing Mechanisms, and so on).

By using contemporary standards, the language is extensible, open, and ready to meet the interoperability requirements for web services.

In short, XrML has been developed to be obsolesce-proof; simple, yet highly expressive; and forward looking to business models in the future.