XMCL - the eXtensible Media Commerce Language

Initial Draft 2001-06-19
Jeffrey Ayars, RealNetworks Inc.
jeffa@real.com

Abstract

This document specifies XMCL, the eXtensible Media Commerce Language. The language is an interchange format that describes usage rules that apply to multi-media content. It is designed to communicate these rules in an implementation independent manner for interchange between business systems and DRM implementations responsible for enforcing the rules described in the language.

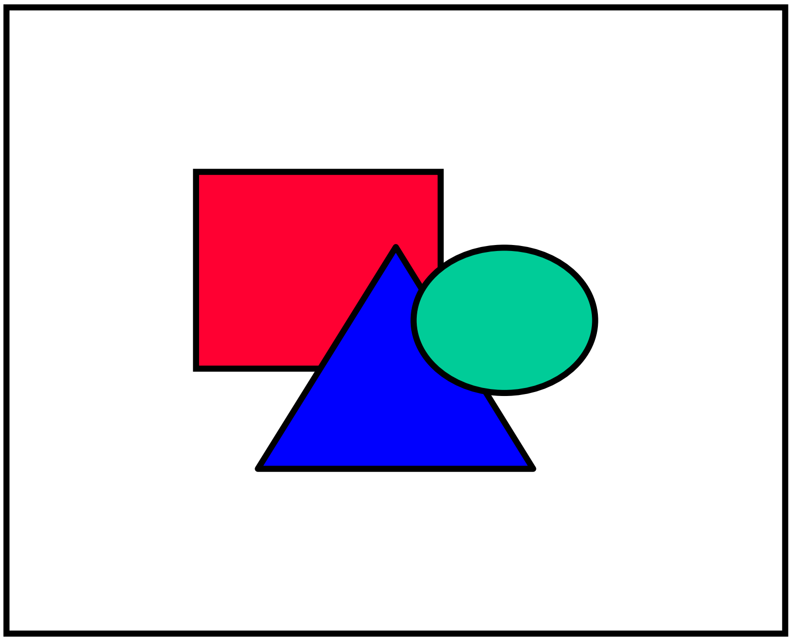
Status

This document is a draft of XMCL language specification. This specification is intended for submission to an international standards body for open community debate and finalization. This initial draft is produced by RealNetworks and will be superseded by a working draft produced by the standards body that takes up work on finalizing the language specification. This draft is not complete, where elements or attributes are not fully described, their intended use described as a placeholder. Items in red are editorial notes for the authors/contributors to consider while working on the document. They are not meant to be part of the specification.

Table of Contents

Introduction

This document specifies the Syntax and Semantics of XMCL, an eXtensible Media Commerce Language. Digital media in a rights management system flows through a number of steps on its journey to a consumer's eyes and ears. The steps are: create, package, publish, distribute, license, and consume. At least a subset of these abstract steps is implemented by all rights management systems today. The service or business owner that manages one or more of these steps varies widely depending on the relationships negotiated for a specific piece of content. However, there is a natural break between the back-end systems for publishing and licensing on one side and the trusted system that packaged the content and enforces the business rules for the content on the other. This division is between the systems that describe the business rules for the content and the specific implementation that enforces those rules.



XMCL is a "rights specification language", as defined by the Association of American Publishers [DRMEBOOKS]. The purpose of XMCL is for interchange of business rules to be applied to media between business systems (e.g. web store fronts, customer tracking and management) and trusted delivery and playback systems (e.g. a DRM implementation that will enforce the rights described in the XMCL document). Through the use of XMCL business systems are completely free of knowledge of specific trusted system implementations. This separation of the business systems and the trusted system allow businesses to support one or more trusted systems and provides the option of changing trusted systems as conditions change without changes to the business systems.

XMCL describes the minimum, self-complete set of business rules under which digital media is licensed for consumer use. These business rules support multiple business models including rental, subscription, ownership, and video on demand/pay-per-view. When a business system authorizes a customer transaction for digital media, it generates a XMCL document that is then acted upon and enforced by a specific trusted system. The generated XMCL document is submitted to the trusted system through the APIs of the trusted system (e.g. HTTP POST, RPC call, API call).

This document does not define interoperability end-to-end, but takes an important first step in that direction. As noted in the summary of the recent W3C Workshop on DRM, none of the standards efforts they evaluated "[deal] with what we think of as the essential first step for the Web: the simple expression and communication of IPR information and policies." [WWWDRM].

Conformance conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in RFC2119 [KEYWORDS]:

> "they MUST only be used where it is actually required for interoperation or to limit behavior which has potential for causing harm (e.g., limiting retransmissions)"

Consequently, we use these capitalized keywords to unambiguously specify requirements over protocol and application features and behavior that affect the interoperability and security of implementations. These key words are not used (capitalized) to describe XML grammar; schema definitions[XSCHEMA-STRUCTURE] unambiguously describe such requirements and we wish to reserve the prominence of these terms for the natural language descriptions of protocols and features. For instance, an XML attribute might be described as being "optional." Compliance with the XML-namespace specification [XMLNAMES] is described as "REQUIRED."

XMCL Overview and Examples

This section provides an overview and examples of XMCL for a few common use cases. This section is informative. Refer to Processing Rules (section 3) and Core XMCL Syntax (section 4) for the normative definition of the language. XMCL documents are intended to be the interchange that bridges between a back end business system and a specific trusted system. All XMCL documents have the following structure (where "?" denotes zero or one occurrence; "+" denotes one or more occurrences; and "*" denotes zero or more occurrences)

```
<xmcl>
  (clientInfo)?
  (license)+
  (auth)?
</xmcl>
```

The clientInfo section will be specific to each enforcement system. It contains parameters that are required for the trusted system to generate the license (e.g. cryptographic keys, request or response authentication data, etc.). There will be one or more license sections, one for each license that should be granted. The license section describes the specific business rules that should be applied to a specific piece of content. XMCL supports multiple licenses in a single request, which would enable the licensing of some purchased content to include a promotional license to another piece of content. The auth section optional and contains authentication information so the receiver can validate the request is authentic if required. There are a number of business models that feed into the requirements for XMCL. They are rental, subscription, ownership, video on demand/pay per view, promotion, and corporate internal communications. Example XMCL documents for rental, subscription and ownership follows. In the examples, attributes and elements may be omitted, as the intent is simply to get a feel for the language, not provide a definition or guide for the language.

Ed note: are there any more significant business models that should be included?

Rental Example

For the rental business model a small set of rules need to be specified that simply describe the rental period. The following is an example XMCL document describing a 24-hour rental license for the movie "First Blood". The rental period begins when the movie is first watched and must occur within a week of purchase. For this example, the XMCL document always follows a template with only information regarding the specific customer and date/time information modified.

```
<xmcl>
     <license>
        <contentInfo>
           <contentId type="GUID">
             13AC7DE5-8028-42fe-95CE-0DC2221891C7
           </contentID>
           <ds:KeyInfo
               xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
               <ds:KeyName>ContentKey</ds:KeyName>
               <ds:KeyValue>
```

```
                  <key algorithm="urn:nist-gov:tripledes-ede-cbc">
                3812A419C63BE771 AD9F61FEFA20CE63 3812A419C63BE771
                  </key>
                <ds:KeyValue>
              <ds:KeyInfo>
               <rdf:RDF xmlns:rdf=
                    "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
                    xmlns:dc="http://purl.org/dc/elements/1.1/">
                <rdf:Description>
                  <dc:title>First Blood</dc:title>
                  <dc:subject>
                       movie, action, adventure
                  </dc:subject>
                </rdf:Description>
             </rdf:RDF>
           </contentInfo>
           <validPeriod start="2001614T184300"
                      end="2001621T184300"/>
           <usageRights>
              <useDuration length="24h" begin="firstUse"/>
           </usageRights>
       </license>
      </body>
```

```
34.   </xmcl>
```

This XMCL document contains a single <license> element. The <contentID> element on line 4 defines the unique identifier for the content. The type attribute specifies that the id used is a GUID. The <ds:KeyInfo> element on line 7 specifies the symmetric key for decryption of the content. KeyInfo is borrowed from [XMLDSIG]. Lines 19-24 describe meta information that further defines the content using Dublin Core elements as defined in [DCMES-XML]. The <validPeriod> element on line 26 describes that the license is valid from June 14, 2001 at 18:43 to June 21, 2001 at 18:43. The useDuration element on line 29 says the content is licensed for 24 hours from first use.

Ed note: reference [MM] after description of DC use is done

2.2 Subscription Example

The following is an example XMCL document describing a subscription license for "Warren's Great Music Collection" valid during the month of June. The subscription period begins when the license is issued and the user may obtain a new license without revisiting a web site.

```
1. <xmcl>

2.      <license>

3.         <contentInfo>

4.            <contentId type="GUID">

5.              13AC7DE5-8028-42fe-95CE-
   0DC2221891C7

6.            </contentID>

7.             <ds:KeyInfo

8.
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

9.
   <ds:KeyName>ContentKey</ds:KeyName>

10.               <ds:KeyValue>

11.                 <key algorithm="urn:nist-
   gov:tripledes-ede-cbc">

12.               3812A419C63BE771
   AD9F61FEFA20CE63 3812A419C63BE771
```

```
                 </key>
               <ds:KeyValue>
             <ds:KeyInfo>
             <rdf:RDF xmlns:rdf=
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
               <rdf:Description>
                 <dc:creator>Warren Piece</dc:creator>
<dc:type>Collection</dc:type>
                 <dc:title>
                   Warren's Great Music Collection
                 </dc:title>
                 <dc:date>2001</dc:date>
                </rdf:Description>
             </rdf:RDF>
          </contentInfo>
          <validPeriod start="20010601T000000"
end="20010701T000000"/>
          <usageRights>
            <useDuration length="P1M" begin="issue">
            <subscriptionRenewalURL
contents="http://subscription.garagebands.com/"/>
```

```
        </usageRights>
    </license>
   </body>
</xmcl>
```

Ed note: need to describe this example. The <subscriptionRenewalURL> element is just an idea, there may be more primitive elements that gain the same functionality.

2.3 Ownership Example

The following is an example XMCL document describing an ownership license for the song "Bang the Drum Slowly".

```
<xmcl>
     <license>
        <contentInfo creatorID="Frank LeMedere"
                     licensorID="Audiotrax">
           <contentID type="GUID">
             13AC7DE5-2180-42fe-95CE-0D18028891C7
           </contentID>
           <ds:KeyInfo
               xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
               <ds:KeyName>ContentKey</ds:KeyName>
                <ds:KeyValue>
                  <key algorithm="urn:nist-gov:tripledes-ede-cbc">
                3812A419C63BE771 AD9F61FEFA20CE63 3812A419C63BE771
                  </key>
                <ds:KeyValue>
           <ds:KeyInfo>
           <rdf:RDF xmlns:rdf=
             "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:dc="http://purl.org/dc/elements/1.1/">
             <rdf:Description>
               <dc:creator>Emmy Lou Harris</dc:creator>
               <dc:title>
                 Bang The Drum Slowly
               </dc:title>
               <dc:date>2001</dc:date>
             </rdf:Description>
           </rdf:RDF>
        </contentInfo>
        <validPeriod start="2001614T184300" />
        <usageRights>
          <useDuration length="infinite">
         <usageRights>
        <copy>
          <toCD value="true"/>
          <toPD value="true"/>
          <toPC value="false"/>
        </copy>
    </license>
   </body>
</xmcl>
```

Ed note: How should ownership be designated? The length attribute is a Schema duration type which doesn't take the value "infinite" and the <validPeriod> element requires and end date. Digital Certificate theory tells us certs must have end periods but ownership of something doesn't end.

Ed note: <copy> element isn't included in syntax. It's obvious what it's intent is but it's expression is not consistent with the rest of the syntax. Are there a common set of allowed "destinations" like CD, VHS, DVD, PD (portable device), PM (portable media), that they can be described?

Or something like:

```
     <allowedUse>
       <param name="burnToCD" value="true"/>
       <param name="copyToPD" value="true"/>
       <param name="copyToPC" value="false"/>
    </allowedUse>
```

Processing Rules

This section is normative. An implementation MUST parse, understand, and enforce a license described using the core syntax. If an implementation cannot understand or enforce any rules described in the license section, it MUST return an error and fail to return a license. An implementation MUST support at least one <license> element per document. Support for multiple <license> elements is OPTIONAL. An implementation MUST support XML namespaces [XMLNAMES]. Namespace extensions that are not recognized MUST be ignored. If a license if valid without the namespace qualified elements, it MUST be considered to be understood. If a license is not valid when qualified elements are ignored, an implementation MUST return an error and fail to return a license.

Ed note: are there other processing rules?

Core XMCL Syntax

The general structure of an XMCL document is described in the XMCL Overview (section 2). This section provides detailed syntax of the core XMCL features. Features described in this section are mandatory to implement unless otherwise indicated. The XML Schema definitions used are from [XMLSCHEMA]

The <xmcl> element

The xmcl element is the outer structure element in the language. There MUST be only one xmcl element per document. It contains all of the licenses along with any trusted system specific information required to generate or enforce the license.

```
Schema Definition:

<element name="xmcl">
   <complexType>
      <sequence>
         <element ref="clientInfo" minOccurs="0"/>
         <element ref="license" maxOccurs="unbounded"/>
         <element ref="revocationList" minOccurs="0"/>
         <element ref="auth" minOccurs="0"/>
      </sequence>
   </complexType>
</element>
DTD:

<!ELEMENT xmcl (clientInfo?, license+, revocationList?,
auth?)>
```

The <clientInfo> element

The clientInfo element contains any parameters that are needed by a specific trusted system to generate or enforce the license(s) described by the <license> elements. There are no defined attributes for this element at this time. Base XML attributes defined in [XML] SHOULD be allowed. The clientInfo element contains one or more <param> elements.

```
Schema Definition:

<element name="clientInfo">
   <complexType>
      <sequence>
         <element ref="param" maxOccurs="unbounded"/>
      </sequence>
   </complexType>
</element>
DTD:

<!ELEMENT clientInfo (param+)>
```

Ed note: Should clientInfo be better defined, not defined at all? Something that is outside the XCML fragment all together?

The <param> element

The param element is borrowed from [XHTML] XHTML 1.0 and is used to define name value pairs to convey implement specific parameters to the trusted system.

```
Schema Definition
<element name="param">
   <complexType>
      <attribute name="id" type="ID"/>
      <attribute name="name" type="string"/>
      <attribute name="value" type="string"/>
      <attribute name="valuetype" use="default"
value="data">
        <simpleType>
           <restriction base="NMTOKEN">
              <enumeration value="data"/>
              <enumeration value="ref"/>
              <enumeration value="object"/>
           </restriction>
        </simpleType>
      </attribute>
      <attribute name="type" type="string"/>
   </complexType>
</element>
DTD:
<!ELEMENT param EMPTY>
<!ATTLIST param
   id ID #IMPLIED
   name CDATA #IMPLIED
   value CDATA #IMPLIED
   valuetype (data | ref | object) #IMPLIED
   type CDATA #IMPLIED
>
```

The <license> element

The license element contains the business rules that the trusted system should use to generate the implementation specific license.

```
Schema Definition

<element name="license">
   <complexType>
      <sequence>
         <element ref="contentInfo"/>
         <element ref="validPeriod"/>
         <element ref="usageRights"/>
         <element ref="copyRights" minOccurs="0"/>
         <element ref="transferRights" minOccurs="0"/>
      </sequence>
      <attribute name="numCopies" type="integer"
use="default"
                 value="1"/>
      <attribute name="allowTransfer" type="boolean"
use="default"
                 value="false"/>
   </complexType>
</element>
DTD:

<!ELEMENT license (contentInfo, validPeriod, usageRights,
copyRights?, transferRights?)>
<!ATTLIST license
   numCopies CDATA #IMPLIED
   allowTransfer CDATA #IMPLIED
>
```

The <contentInfo> element

The contentInfo element identifies and contains information specific to the piece of content being licensed.

```
Schema Definition:
<element name="contentInfo">
   <complexType>
      <sequence>
         <element ref="contentID" minOccurs="0"/>
         <element ref="ds:KeyInfo" minOccurs="0"/>
         <any
namespace="http://purl.org/dc/elements/1.1/"
           minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
   </complexType>
</element>
DTD:

<!ELEMENT contentInfo (contentID?, ds:KeyInfo?, *)>
```

The <contentId> element

Contains an identifier for the content. It contains no other elements. The type attribute specifies the type of data contained by the element.

```
Schema Definition:

<element name="contentID" minOccurs="0">
   <complexType>
      <simpleContent>
         <extension base="string">
           <attribute name="type" use="default"
value="guid">
              <simpleType>
                 <restriction base="NMTOKEN">
                    <enumeration value="guid"/>
                    <enumeration value="????"/>
                 </restriction>
              </simpleType>
           </attribute>
        </extension>
      </simpleContent>
   </complexType>
</element>
DTD:

<!ELEMENT contentID (#PCDATA)>
<!ATTLIST contentID
   type CDATA #IMPLIED
>
```

Ed note: need to define values for the type attribute. What standard id formats are there? Use another name or merge the attribute?

The <ds:KeyInfo> element

This element is taken from XML Digital Signatures [XMLDSIG] and is used to communicate cryptographic keys from system to system.

See XML Digital Signatures [XMLDSIG] for further explanation of the elements semantics.

```
Schema Definition:

<element name="KeyInfo" type="ds:KeyInfoType"/>
   <complexType name="KeyInfoType" mixed="true">
     <choice maxOccurs="unbounded">
       <element ref="ds:KeyName"/>
       <element ref="ds:KeyValue"/>
       <element ref="ds:RetrievalMethod"/>
       <element ref="ds:X509Data"/>
       <element ref="ds:PGPData"/>
       <element ref="ds:SPKIData"/>
       <element ref="ds:MgmtData"/>
       <any processContents="lax" namespace="##other"/>
        <!-- (1,1) elements from (0,unbounded) namespaces
-->
     </choice>
     <attribute name="Id" type="ID" use="optional"/>
   </complexType>
</element>
DTD:

<!ELEMENT KeyInfo
(#PCDATA|KeyName|KeyValue|RetrievalMethod|
              X509Data|PGPData|SPKIData|MgmtData
%KeyInfo.ANY;)* >
<!ATTLIST KeyInfo
    Id  ID   #IMPLIED
>
```

The <key> element

This spec extends the <ds:KeyValue> [XMLDSIG] element with the key element. The key element opens the algorithm list by providing a container for the key data and the algorithm attribute to identify the algorithm. The algorithm attribute uses the algorithm URI's from XML Encryption [ALGORITHMS].

```
Schema Definition:

<element name="key">
   <complexType>
      <simpleContent>
         <extension base="string">
           <attribute name="algorithm"
type="uriReference"
              use="required"/>
        </extension>
      </simpleContent>
   </complexType>
</element>
DTD:

<!ELEMENT key (#PCDATA)>
<!ATTLIST key
   algorithm CDATA #REQUIRED
```

```
>
```

The <validPeriod> element

The validPeriod element describes the dates during which the license is valid. It takes a start and end attribute. The start and end attributes define the calendar dates and times in the format described in [XSCHEMA-DATATYPES]. The enforcement system should only honor the rights expressed in the license after the date-time specified with the start attribute and until the date specified in the end attribute.

```
Schema Definition:

<element name="validPeriod">
   <complexType>
      <attribute name="start" type="dateTime"
use="required"/>
      <attribute name="end" type="dateTime"
use="required"/>
   </complexType>
</element>
DTD:

<!ELEMENT validPeriod EMPTY>
<!ATTLIST validPeriod
   start CDATA #REQUIRED
   end CDATA #REQUIRED
>
```

The <usageRights> element

The usageRights element describes the rights that are granted in the license.

```
Schema Definition:

<element name="usageRights" type="rights">
   <complexType name="rights">
      <all>
         <element ref="useDuration" minOccurs="0"/>
         <element ref="useCount" minOccurs="0"/>
         <element ref="userInteraction" minOccurs="0"
                 maxOccurs="unbounded"/>
         <element ref="require" minOccurs="0"
maxOccurs="unbounded"/>
         <element ref="allowedUse" minOccurs="0"
                 maxOccurs="unbounded"/>
         <element ref="impRights" minOccurs="0"
maxOccurs="unbounded"/>
      </all>
   </complexType>
</element>
DTD:

<!ELEMENT usageRights (useDuration?, useCount?,
userInteraction*, require*, allowedUse*, impRights*)>
```

The <copyRights> element

The copyRights element describes the rights that are granted in the copy license. If the element is empty, all the rights described in the <usageRights> are copied.

```
Schema Definition:

<element name="copyRights" type="rights">
   <complexType name="rights">
      <all>
         <element ref="useDuration" minOccurs="0"/>
         <element ref="useCount" minOccurs="0"/>
         <element ref="userInteraction" minOccurs="0"
                  maxOccurs="unbounded"/>
         <element ref="require" minOccurs="0"
maxOccurs="unbounded"/>
         <element ref="allowedUse" minOccurs="0"
                 maxOccurs="unbounded"/>
         <element ref="impRights" minOccurs="0"
maxOccurs="unbounded"/>
      </all>
   </complexType>
</element>
DTD:

<!ELEMENT copyRights (useDuration?, useCount?,
userInteraction*, require*, allowedUse*, impRights*)>
```

The <transferRights> element

The transferRights element describes the rights that are granted in the transfer license. If the element is empty, all the rights described in the <usageRights> are transferred.

```
Schema Definition:

<element name="transferRights" type="rights">
   <complexType name="rights">
      <all>
         <element ref="useDuration" minOccurs="0"/>
         <element ref="useCount" minOccurs="0"/>
         <element ref="userInteraction" minOccurs="0"
                 maxOccurs="unbounded"/>
         <element ref="require" minOccurs="0"
maxOccurs="unbounded"/>
         <element ref="allowedUse" minOccurs="0"
                 maxOccurs="unbounded"/>
         <element ref="impRights" minOccurs="0"
maxOccurs="unbounded"/>
      </all>
   </complexType>
</element>
DTD:

<!ELEMENT transferRights (useDuration?, useCount?,
userInteraction*, require*, allowedUse*, impRights*)>
```

The Rights elements

The following elements are used to specify rights in the <usageRights>, <copyRights>, and <transferRights> elements. They describe the interoperable right set as well as a generic right element, <impRight>, that can be used to describe rights specific to a particular implementation.

The <useDuration> element

The useDuration element specifies a length of time the content can be used after a specific event. The begin attribute specifies when the duration should begin. Its values are "issue" and "firstUse". The "issue" value specifies the duration starts at the issue of the license. The "firstUse" value specifies the duration starts when the content is first used. The length attribute specifies the length of time.

```
Schema Definition:

<element name="useDuration">
   <complexType>
      <attribute name="begin" use="default"
value="firstUse">
        <simpleType>
           <restriction base="NMTOKEN">
              <enumeration value="firstUse"/>
              <enumeration value="issue"/>
           </restriction>
        </simpleType>
      </attribute>
      <attribute name="length" type="duration"
use="required"/>
   </complexType>
</element>
DTD:

<!ELEMENT useDuration EMPTY>
<!ATTLIST useDuration
   begin (firstUse | issue) #IMPLIED
   length CDATA #REQUIRED
>
```

The <useCount> element

The useCount element specifies the number of times the content can be used. It takes the count attribute and the threshold attribute.

```
Schema Definition:

<element name="useCount">
   <complexType>
      <attribute name="count" type="integer"
use="required"/>
      <attribute name="threshold" type="duration"
use="optional"/>
   </complexType>
</element>
DTD:
```

```
<!ELEMENT useCount EMPTY>
<!ATTLIST useCount
   count CDATA #REQUIRED
   threshold CDATA #IMPLIED
>
```

The <require> element

Ed note: The intent of the require element is to describe conditions that the trusted system are to require for playback. It should include things like employment water marking technology, use of a specific decompressor implementation, etc.

```
Schema Definition:

<element name="require" type="string"/>
DTD:

<!ELEMENT require (#PCDATA)>
```

The <userInteraction> element

Ed note: The intent of the userInteraction element is to describe something that requires user interaction before use of the licensed content will be allowed. This includes things like accepting a "click-wrap" license agreement at each playback.

```
Schema Definition:

<element name="userInteraction" type="string"/>
DTD:

<!ELEMENT userInteraction (#PCDATA)>
```

The <allowedUse> element

Ed note: The intent of the allowedUse element is to describe things like the content can only be included as a whole vs. included in a larger work.

```
Schema Definition:

<element name="allowedUse" type="string"/>
DTD:

<!ELEMENT allowedUse (#PCDATA)>
```

The <impRight> element

The impRight element is a generic extension mechanism that allows inclusion of implementation specific rights in a license. The implementation is identified by the implementationID attribute as a URI [URI].

```
Schema Definition:

<element name="impRights">
```

```
   <complexType>
      <sequence>
         <element name="param" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="implementationID"
type="uriReference"
                 use="required"/>
   </complexType>
</element>
DTD:

<!ELEMENT impRights (param+)>
<!ATTLIST impRights
   implementationID CDATA #REQUIRED
>
```

Ed Note: Should pull in <switch> element and systemRequired element from SMIL 2.0 for this section so a XMCL doc could describe implementation specific rights for multiple implementations and the parsing application could correctly pick the right one.

Extension Syntax

Ed Note: Need to describe extension mechanisms.

Ed Note: Should the attributes listed below be considered core or extensions?

The <revocationList> element

The purpose of the revocationList element is to provide a standard way to describe revocation certificates. Revocation is a key feature of DRM technology and the opportunity to standardize here seems the same as business rules. It MUST contain one or more <revokedItem> elements.

```
Schema Definition:

<element name="revocationList">
   <complexType>
      <sequence>
         <element ref="revokeItem"
maxOccurs="unbounded"/>
      </sequence>
   </complexType>
</element>
DTD:

<!ELEMENT revocationList (revokeItem+)>
```

The <revokedItem> element

The revokedItem element is a non-empty element that describes an item in the trusted system to be revoked. The type attribute describes the type of the data contained in the revokedItem element.

```
Schema Definition:

<element name="revokeItem">
   <complexType>
      <simpleContent>
         <extension base="string">
           <attribute name="type" use="default"
value="guid">
              <simpleType>
                 <restriction base="NMTOKEN">
                    <enumeration value="guid"/>
                    <enumeration value="hash"/>
                 </restriction>
              </simpleType>
           </attribute>
        </extension>
      </simpleContent>
   </complexType>
</element>
DTD:

<!ELEMENT revokeItem (#PCDATA)>
<!ATTLIST revokeItem
   type (guid | hash) #IMPLIED
>
```

Ed note: another type collision. What types of item formats are appropriate?

The <auth> element

The <auth> element contains authentication information such that those processing the document can validate its authenticity.

```
Schema definition:

<element name="auth"/>
DTD:

<!ELEMENT auth (#PCDATA)>
```

Ed note: Use XML Signature for this? Allow for inclusion of X.509 certs? XML Cert?

Definitions

Business system

A business system is a system comprised of one or more business solution packages like database software, user management software, content/asset management software, and e-commerce software combined into a solution like a web storefront that provides downloadable media purchases.

Trusted system

A trusted system is a digital rights management system implementation that is capable of enforcing the rules described in an XMCL document during end user use of content. It includes the software that accepts requests for licenses, issues the licenses and the client software run by the end user that enforces the license.

License

A license is a set of usage rules that define how an end user is allowed to use a piece of content. It is specific to a piece of content and a specific end user.

References

[DCE11] "Dublin Core Metadata Element Set", Dublin Core Metadata Initiative, July, 1999. Available at http://purl.org/dc/elements/1.1/

[DCUSAGE] "Using Dublin Core", Diane Hillmann, April 2001. Available at http://dublincore.org/documents/usageguide/

[DCEMS-XML] "An XML Encoding of Simple Dublin Core Metadata", D. Beckett, E. Miller, D Brickly, December 2000. Available at http://dublincore.org/documents/2000/11/dcmes-xml/

[DRMEBOOKS] "Digital Rights Management for Ebooks: Publisher Requirements Version 1.0" Association of American Publishers, Inc, November, 2000.http://www.publishers.org/home/drm.pdf (see http://www.publishers.org/home/press/ebookpr.htm for more info)

[MM] "MusicBrainz Metadata Initiative", a portable and flexible means of storing and exchanging metadata related to digital audio and video tracks. Work in progress. http://www.musicbrainz.org/MM/

[SMIL20] "Synchronized Multimedia Integration Language (SMIL 2.0)". W3C Proposed Recommendation, work in progress. Available at http://www.w3.org/TR/smil20/

[URI] "RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax" T. Berners-Lee, R. Fielding, L. Masinter, August 1998. http://www.ietf.org/rfc/rfc2396.txt

[WWWDRM] "W3C Workshop on Digital Rights Management for the Web: Workshop Report" J. Erickson, R. Iannella, R. Wenning, January, 2001. Workshop Report at http://www.w3.org/2000/12/drm-ws/workshop-report.html

[XML] "Extensible Markup Language (XML) 1.0 Specification", T. Bray, J. Paoli, C. M. Sperberg-McQueen, 10 February 1998. Latest version available at: http://www.w3.org/TR/REC-xml

[XMLDSIG] "RFC 3075: XML-Signature Syntax and Processing" (XML Digital Signatures) D. Eastlake, J. Reagle, D. Solo, March 2001. Available at http://www.w3.org/TR/xmldsig-core/

[XMLNAMES] "Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 January 1999. XML namespaces provide a simple method for qualifying names used in XML documents by associating them with namespaces identified by URI. Latest version available at: http://www.w3.org/TR/REC-xml-names

[XSCHEMA-STRUCTURE] "XML Schema, XML Schema Part 1: Structures". W3C Recommendation 2 May 2001. Available at http://www.w3.org/TR/xmlschema-1/

[XSCHEMA-DATATYPES] "XML Schema Part 2: Datatypes", Paul V. Biron and Ashok Malhotra, eds., W3C, 2 May 2001. Available at http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[ALGORITHMS] "XML Encryption Syntax and Processing: 6.1Algorithm Identifiers and Requirements" Available at http://lists.w3.org/Archives/Public/xml-encryption/2000Dec/att-0024/01-XMLEncryption_v01.html#_Toc501424263

[XHTML] "The Extensible HyperText Markup Language: A Reformulation of HTML 4.0 in XML 1.0". W3C Recommendation 26 January 2000,
Available at http://www.w3.org/TR/xhtml1/

[KEYWORDS] "RFC 2119: Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. March 1997. http://www.ietf.org/rfc/rfc2119.txt