Abstract—As the electronic distribution of digital media becomes more widespread, the use of digital rights management (DRM) technologies takes on increasing importance. Unfortunately, the use of DRM technologies complicates the problem of providing consistent, predictable behavior to consumers across a wide variety of devices. This paper describes some of the difficulties inherent in providing interoperability in a secure environment and introduces the Coral Consortium, an organization created to address the problem of DRM interoperability. A discussion of the technology behind Coral is followed by an example scenario demonstrating the ability of the Coral framework to integrate with existing DRM systems in a minimally disruptive manner.

Index Terms—Coral, Digital Rights Management (DRM), DRM interoperability, NEMO

1. THE DRM INTEROPERABILITY PROBLEM

It is becoming increasingly apparent that the lack of interoperability among DRM systems is one of the major factors inhibiting the development of a rich, competitive electronic media distribution market. Interoperability impacts every participant in a content value chain:

- **Consumers** naturally expect that content they have legitimately acquired will work with all of their devices, not just a few of them. Too often, the opposite is true: content acquired illegally is typically far more portable than electronically distributed content sold through legitimate means. Until this imbalance is addressed, consumers will not fully embrace the electronic distribution of secure media.

- **Distributors** of content are forced to make technology choices that limit the utility of their services to consumers. In essence, the DRM technology choices adopted by a media distributor are in turn imposed upon their customers, who are forced to choose from a limited subset of media devices available in the market.

- **Content Providers** have a smaller addressable market and a declining amount of control over the business models under which their content is distributed. The choice of a single DRM technology restricts the business models that can be supported by media distributors, which in turn limits the options available to content providers.

- **Device Manufacturers** are constrained by economic and usability considerations, limiting the number of protection technologies that can be integrated into devices. Device manufacturers are forced to choose, making the success of devices dependent more upon the success of services that integrate the same DRM technology than on the features or desirability of the devices themselves.

The lack of interoperability leads to a fragmented marketplace and to the development of clusters of devices and services aligned only by their use of the same DRM technology. As a result, non-aligned, innovative technologies are squeezed out of the market, reducing competition and consumer choice.

2. DRM STANDARDIZATION IS DIFFERENT

The need for interoperability in media distribution systems is not new, but the increasing use of DRM systems to govern access to media adds new complexities to an old problem. Some of the approaches that have proved successful in the past are less applicable. Secure digital media distribution places constraints on solutions to the interoperability problem, constraints that must be acknowledged and embraced in order to produce a viable solution. DRM systems in the market today:

- **Do not trust one another.** The earliest DRM systems were designed quite explicitly not to interoperate with other content protection systems. Designers of these systems believed that ceding control of content to an external system was a violation of their contract with content providers, a security risk, and bad for business. Many of the DRM systems in broad use today still rely on proprietary license formats and protocols.

  In recent years this tendency has relaxed slightly, and many DRM systems have acquired the ability to export content to other systems for limited uses such as rendering. However, there is no general mechanism for secure content interchange between systems, even when they are regarded as roughly equivalent by content providers.

- **Are not equally expressive.** DRM systems vary greatly in terms of the usage models they can express. The broadcast copy control (CCI) bits used in the United States rely on two bits of signaling to convey rights, while other systems (such as MPEG-21 REL) are almost infinitely expressive. Given this disparity, fully automating the task of DRM license conversion is impractical in most cases. Content providers are the ultimate arbiter of equivalence.

- **Are still evolving.** The state of the art in DRM systems is advancing rapidly as consumers are exposed to the uses and limitations of such systems. Content providers have
learned through hard experience that overly restrictive models, intended to protect their interests, do quite the opposite in practice. These factors have tended to push more recent DRM systems in a more consumer-friendly direction. As a result, standardization around a particular set of fixed usage models is still premature.

- **Are difficult to integrate.** Implementing, integrating, and operating DRM systems are expensive and time-consuming tasks. For economic reasons, most content distributors choose to deploy a single technology. That choice ripples through the value chain as consumers choose to buy (or more pointedly, not to buy) devices that integrate the same technology, and device manufacturers strive to build devices that consumers will accept. Similar economic reasoning leads device manufacturers to limit the number of DRM technologies they build into devices. The tremendous sunk costs provide major disincentives for updating the technology or incorporating new technologies as the state of the art advances.

- **Create artificial stickiness.** Technology choices made by service providers and device manufacturers force consumers to buy into a subset of devices available to them, based on their perception of the services that best meet their needs. Once the initial investment is made, subsequent investments in the same technology are necessary to avoid interoperability problems. Consequently, an overall lack of interoperability can benefit technology and service providers who obtain an initial edge in a particular market. Some of the most successful companies in the electronic media distribution space have exploited this fact as a competitive weapon.

The Coral Consortium was formed to address the DRM interoperability problem in the face of the technical and market realities described above. These realities led Coral to decide against standardization of a new DRM system, which is perhaps the most direct way to solve the DRM interoperability problem. Instead, the group decided to focus on the creation of a set of specifications that define interoperability between existing DRM system, requiring as few changes in those DRMs as feasible. This approach allows participants in a content value chain to compete and cooperate in new ways without fundamentally altering the dynamics of the current DRM market.

### 3. Coral Specifications

The specifications for the Coral interoperability framework are divided into three layers, ordered here from infrastructure to application.

1) **NEMO.** The Networked Environment for Media Orchestration [NEMO] is a set of specifications that enable trusted, secure communications between different entities in the value chain. Technically, it is a secure messaging architecture in which each participant has an identity (secured by a certificate) and a set of role attributes associated with that identity. Entities that hold certified identities and roles are known as nodes. Nodes communicate with one another using a web-services style architecture, using messages that are encrypted and digitally signed. The NEMO layer is infrastructural in Coral; specific application semantics are added at higher layers.

2) **The Coral Core Architecture.** The Coral Core Architecture [CCA] is an application layer built on top of NEMO. Where NEMO specifies a mechanism for establishing identities and roles, the Core Architecture defines a sequence of specific roles that appear in all Coral interoperability transactions and standardizes the messages transacted between systems playing these roles. Any set of deployed nodes based on this specification is referred to as a Coral deployment.

3) **Coral Ecosystems.** Even though it is characterized above as an application layer, the Coral Core Architecture contains a great deal of flexibility that can be used in different ways by different adopters. As such, it is perhaps best regarded as a middleware layer between NEMO and a Coral Ecosystem. A Coral Ecosystem defines a very specific configuration of the tools that are provided by the Coral Core Architecture, with enough detail to allow fully interoperable implementations. The differences between technology in the Core Architecture layer and in the Coral Ecosystem layers are characterized below.

### 4. The Coral Interoperability Model

The Coral interoperability framework is a thin layer of standardized technology that mediates between different DRM technologies to provide the type of consistent, predictable experiences that users have come to expect from interoperable media formats such as CD and DVD. In general, DRM systems prevent this type of consistent experience; the goal of the Coral Consortium is to provide technology that can be used to overcome this restriction.

The difficulties associated with the use of DRM technology in a media distribution environment, as described in Section 2 impose design constraints on interoperability solutions. Specifically:

- Licenses cannot be assumed to be exposed by DRM systems in a form than can be directly translated or mapped.
- DRM systems may vary in their level of sophistication; some concepts that can be expressed in one DRM system may be inexpressible in another.
- The interoperability solution must not force existing DRM systems to adopt a particular rights expression mechanism or enforcement technology and must require as few changes as possible to these systems.

Given these design constraints, the Coral framework adopts a *license derivation* strategy, in which DRM licenses are not directly inspected and translated, but rather, derived from a standardized policy artifact known as a *Rights Token*. 
4.1. Rights Tokens

A Rights Token is a data structure with three primary subcomponents:

1) A principal identifier indicating one or more binding targets for the rights. Every principal name is associated with a namespace in which it is meaningful. For example, a user name at a given storefront can be used as a principal in a namespace defined by that storefront. The names themselves need not be interoperable, but the method for expressing them is standardized.

2) A resource identifier containing the name of the content resource for which rights were acquired. Resource names are placed in a namespace context just like principal names, and these namespaces may be unique to a given system.

3) A usage model identifier, identifying the usages associated with the given principal and content pair. The usage model identifier is a flexible data structure that can be as expressive as needs dictate and can vary between systems. It is important to note that usage model identifiers are not rights expressions that are parsed and evaluated in consuming devices. They provide a vehicle for communicating usage policies in the larger context of a Coral Ecosystem.

Rights Tokens provide a way for systems built by different manufacturers to work together by providing a standardized syntax. Note, however, that each of the major subcomponents of a Rights Token may exist in a namespace that is understood only by a single system. How then is it possible to provide interoperability between systems?

4.2. Identifiers and Relations

Two techniques are used in the Coral framework to provide interoperability when handling principal identifiers, resource identifiers, and usage model identifiers from different namespaces: mapping and standardization.

4.2.1. Mapping Between Namespaces

In the mapping approach, identifiers are mapped across namespaces by known or discoverable relationships among identifiers. Each of the three subcomponents of a Rights Token can be mapped between namespaces using one of two types of relationship:

1) Equivalence Relations — If $A \leftrightarrow B$, then identifier $B$ can be substituted for identifier $A$, and vice-versa.

2) Directed Relations — If $A \rightarrow B$, then identifier $B$ can be substituted for identifier $A$, but not the other way around.

Examples of how such mappings might be used include:

- An identity federation system can be used to maintain equivalence relations between principal identifiers, so that rights established for a user identified in one system can be mapped to rights for an equivalent user in a second system.

- The same concept may apply to resource identifiers to allow different vendors to use their own numbering scheme for identifying content resources. The ability to realize such mappings allows Coral to integrate with legacy content distribution systems.

- Different types of identifiers can be mapped to one another via directed relations, which allows, for example, a single identifier for a group of devices to be mapped...
to distinct identifiers for each member of the group via directed relations. This setup allows individual device identifiers to be substituted for the group identifier, which provides the basis for implementing DRM-agnostic authorized domains in Coral.

The manner in which various identifiers map to one another is not constrained by the Core Architecture, but the semantics of these mappings must be understood by all systems that need to manipulate them. These mapping semantics are typically defined in a Coral Ecosystem specification.

To extend the authorized domain example above, a Coral Ecosystem might define a type of principal identifier for the group of devices, a type of principal identifier for the devices themselves, and conditions under which relations may be established between them. The Coral Ecosystem would also need to define specific entities (such as Domain Managers) that build on the Core Architecture Principal Manager role to maintain and expose the relationships between principal identities.

4.2.2. Standardizing Namespaces

The second strategy for providing interoperability between identifiers in different namespaces is standardization. No mapping would be necessary, for example, if the namespace and specific usage model identifiers that appear in Rights Tokens were drawn from a standardized set. The Coral framework allows Coral Ecosystems to standardize around namespaces when it makes sense to do so for a given deployment environment. For example:

- The participants in the Ecosystem may choose to use a consistent namespace for numbering resources, such as the Global Release ID system [GRid] for sound recordings.
- Ecosystem participants may choose to index a standard set of usage models with the numbers 1, 2, ..., n, and enforce the rule that each system built to the Ecosystem specification should understand the meaning of the enumerated models.

4.3. Deriving Licenses

Rights Tokens are the basic medium of exchange between systems in Coral. Their primary purpose is to describe how license for specific DRMs should be created in order to provide a consistent usage experience for consumers. In Coral, the derivation of licenses from Rights Tokens is a distributed process.

The Coral framework defines a role called a Rights Instantiator whose responsibility is to create DRM licenses from Rights Tokens that enforce the usages expressed in those Rights Tokens. A Rights Instantiator consists of at least three components:

1) A standardized interface called RegisterRights by which Rights Tokens are passed to the Rights Instantiator. The Rights Instantiator persistently stores the Rights Tokens.
2) A component that converts stored Rights Tokens into a sequence of interactions with a native DRM license server which ultimately leads to the creation of native
Figure 3. A Coral interoperability scenario. Coral roles are shown as green rounded boxes. DRM-specific technologies are shown as blue square boxes. The rights databases, which form an important point of integration between Coral and DRM systems, are shown in red. Solid lines represent standardized Coral interfaces; dashed lines represent interfaces not specified by Coral.

3) One or more native DRM license servers capable of generating licenses that enforce the usage models standardized or mapped by a Coral Ecosystem. DRM clients acquire licenses from the DRM license server using the protocols defined by the DRM system.

Many Rights Instantiators may be available for a given Coral Ecosystem, each of which creates licenses for one or more DRM systems — there is no centralized entity responsible for mapping Rights Tokens to DRM licenses. Coral Ecosystem compliance rules ensure that Rights Instantiators produce DRM licenses that enforce the Ecosystem usage models to the satisfaction of the designers of that Ecosystem.

4.4. Rights Mediation

Rights mediation is the most important transaction defined in the Coral framework; it is the process by which rights move between systems with different DRMs. The Core Architecture defines a role (Rights Mediator) that orchestrates the mediation process.

During rights mediation:
- Rights Tokens are retrieved from entities that store them persistently, known as Rights Registries. Rights Registries can be characterized as DRM-agnostic rights lockers that store a representation of rights for later instantiation as DRM licenses.
- Mappings between the principal identifiers, content identifiers, and usage model identifiers contained in the Rights Tokens are verified. The systems that maintain these relations are defined by specific Coral Ecosystems, making the rights mediation process Ecosystem-dependent. An example of such a system is a Domain Manager that maintains mappings between an identifier for a set of devices and the identifiers for the devices themselves. Systems such as Domain Managers are instances of a more general type of entity known as a Principal Manager defined by the Coral framework.
- Policy decisions are made as to whether sufficient authorization exists to allow the requested movement of rights between systems. This policy decision may depend on factors such as:
  - The presence or absence of particular mappings between principal, resource, and usage model identifiers (e.g. domain membership, valid usage mappings)
  - The support of a given destination system for certain usage semantics (e.g. a tamper-resistant time source that supports subscription models)
  - The identity of the user requesting the rights transfer.
- Rights Tokens are delivered to Rights Instantiators that will convert them into DRM licenses. Ecosystem-specific compliance rules ensure that the generated licenses enforce the appropriate usage semantics.

5. AN INTEROPERABILITY SCENARIO

To put the discussion of the Coral interoperability framework in context, Figure 3 shows a typical transaction involving
two DRM systems, A and B. In this example, the two clients are authorized to obtain DRM licenses for the content by virtue of their membership in an authorized domain.

1) A user obtains the right to acquire licenses for a particular piece of content for all devices in the authorized domain. In this step, the DRM-A client also obtains a DRM-A license and protected content. These transactions are not standardized by Coral.

2) Service-side logic stores information associated with the purchase in a database of acquired rights. This information may or may not be stored in the form of a Rights Token. The precise nature of the integration is not specified by Coral, but may be subject to the compliance restrictions associated with a particular Coral Ecosystem.

3) At a later time, a DRM-B client that is integrated with the Coral Client role initiates a rights mediation process so that it can obtain a license for the content in its own DRM and media formats.

4) The Rights Mediator queries a Rights Registry at the point of purchase to ascertain the existence of the rights.

5) The Rights Registry queries the database of acquired rights and produces evidence of the rights in the form of a Rights Token. This is a mandatory integration point for which no implementation is standardized.

6) The Rights Mediator verifies that the requesting device is a member of the authorized domain by querying a Principal Manager, which is an internally integrated component in this example.

7) After verifying that the requesting Client is authorized to receive a license, the Rights Mediator registers the Rights Token with a Rights Instantiator, integrated with a second Coral-enabled service.

8) The service receiving the rights records them in a database of acquired rights, again possibly in the form of a Rights Token. This is a mandatory integration point for which no implementation is standardized.

9) Upon receiving confirmation that the rights mediation process has completed successfully, the Client, integrated with DRM-B technology, triggers the license acquisition process between the DRM-B client and the corresponding license service. Coral does not specify how this integration is to be achieved.

10) The DRM client requests the license. This is a non-Coral transaction.

11) Internally, the service verifies that the client requesting the license has the legitimate right to acquire it. Assuming that it does, the service can return the DRM license to the DRM-B client. The implementation of this step depends on the internal architecture of the server and is outside of the scope of Coral.

6. REFERENCES

