DReaM-MMI Profile for Creative Commons Licenses

Sun Microsystems Laboratories

Draft Profile 1.0
May 22, 2007

This document defines a draft specification for supporting Creative Commons licensed media within the DReaM framework. In its final form, this profile will detail all that a conforming system, both client and server, needs to interoperate and support Creative Commons licensed media.

1. Introduction

Traditional copyright assigns **all** rights to the copyright owner. If a content creator opts for no copyright, this leaves the content creator without control of the usage of their work. Creative Commons[1] has developed a middle ground: a set of licenses in which the content creator/owner retains some control over the usage of their documents, while allowing the user to have some creative control over the content.

Creative Commons enables content creators and users the ability to offer and use content for creative collaboration: the photographer who slices and dices and creates a new work, the teacher who pulls together a set of interpretations of a historical event (this could be film, art, text), the musician who creates a sampling remix. At first Creative Commons licenses would seem to have little in common with Digital Rights Management (DRM) systems, which, after all, seek to delimit what users can do with content. In fact, DRM systems enable users to do certain things with content but not others; Creative Commons licenses do this as well. The distinction is that most DRM systems are typically concerned with restrictions, such as prohibitions against copying, whereas Creative Commons licenses are fundamentally about sharing while controlling such issues as attribution, derivation, and non-commercial use.

DReaM, being a general-purpose rights system, can provide support for Creative Commons licensed content. There are two possible types of support: notification and enforcement. In keeping with the spirit of Creative Commons, DReaM support is limited to notification and does not include enforcement. We discuss this further in section 3.

Note that Creative Commons licenses require that works be distributed in a manner that does not restrict fair use rights. This means that Creative Commons licenses apply upon the instant of sharing content, and not before, since acting on content in a "personal space" is a fair use right. The DReaM support for Creative Commons licenses occurs in that context.

This profile defines how Creative Commons licensed content is supported in DReaM. It details the specific terms used to request permission to share, and the correct responses from the server in each case. Typical client behavior is outlined, and several examples are given of full exchanges. Using this profile, interoperable clients and servers can be developed to support Creative Commons licensed content. **Note: The authors of this specification have worked to comply with Creative Commons principles, but that does not imply Creative Commons endorsement of the specification**.

2. Background

DReaM is a core technology developed by Sun Microsystems Inc. and contributed by

1 http://creativecommons.org

Sun to the Open Media Commons initiative[1]. DReaM's goal is to provide a general-purpose, open, royalty-free digital rights system based on identity. Rights Expression Languages (RELs) typically combine all the functionality needed in a digital-rights management system into the REL; the DReaM architecture has been designed to separate out different functionality, thus enabling varied types of DRM implementations. A key piece of DReaM is the DReaM-MMI (henceforth referred to as MMI[2]) protocol, which defines all interaction between the client and a DReaM server (see section 4.1.1. of the DReaM Overview document as well as the DReaM MMI specification[3]).

Rather than issuing simple requests for content and returning complex license statements, MMI instead lets the client issue complex requests, and returns rather simple responses. In this sense, MMI looks more like a traditional access control protocol than a typical DRM query format. This is done in part to push more power into the client, and in part to avoid infringing on RELs.

In the MMI protocol a Request contains Verbs: statements about what actions are being requested for given content. If the action is allowed, then an MMI Response will simply return ”granted”. The action might also be denied. In either case, an MMI Response MAY return any number of Hints. Hints are notices to the Client about restrictions for access. For instance, a Client might ask for access to a file, and the Hint might come back saying "ask again, but this time specify that you'll only use the content on your current device." The Client can now form a more specific Request, often by providing parameters to the Verbs that were previously specified. Hints are optional in the MMI specification, and used in this profile as an optimization.

The main purpose of this profile is to document how MMI is used by Clients to access Creative Commons licensed content, and how a DReaM Server interprets these licenses and responds to Client requests. Because notification is the only action taken, this is quite simple to explain, as well as to implement. Section 3 describes the Creative Commons licenses in some detail. Section 4 of this profile defines this model in more depth. Section 5 provides specific terms that the client provides in requests, and Section 9 has the responses that the server uses. Sections 6-8 contain details of the security model and license management. Section 11 ties all these together with specific examples.

3. Creative Commons Licenses

Creative Commons enables content creators --- artists, musicians, filmmakers, photographers, human beings --- to share and reuse content while retaining some control over the reuse. To that point, Creative Commons provides three versions of each license: a human-readable commons deed, a lawyer-readable human code, and a machine-readable digital code. The machine-readable code is RDF.

---

1 http://openmediacommons.org/

2 MMI stands for “Mother May I,” since the client is querying the network in much the way a player queries the omnipotent force in the children's game, “Mother May I?”

3 http://www.openmediacommons.org/documentation.html

The first two aspects of Creative Commons licenses are quite straightforward to implement (even if their application to content shared over the globe is not). We all know how to read human-readable languages, and the lawyers among us know how to handle legalese. It is the third version of a Creative Commons license, the machine-readable digital code, that provides interesting challenges. Machine-readable code is machine-readable code, nothing more, nothing less. It does not provide any enforcement of Creative Commons licenses. That is not an accident. Creative Commons encourage sharing and reuse. We begin by discussing the various Creative Commons licenses, and then discuss what it means to implement them in DReaM.

3.1. **Creative Commons License Details**

This is a high-level overview of the terms specified by Creative Commons licenses. The definitions provided are not intended as complete examples. They are merely defined to make use of the terms clear.

As described by Creative Commons, the various sharing licenses give a wide variety of choice in how the content creators license their work[1]:

Attribution. You let others copy, distribute, display, and perform your copyrighted work --- and derivative works based upon it --- but ... only if they give credit the way you request.

Example: Jane publishes her photograph with an Attribution license, because she wants the world to use her pictures provided they give her credit. Bob finds her photograph online and wants to display it on the front page of his website. Bob puts Jane's picture on his site, and clearly indicates Jane's authorship.

Noncommercial. You let others copy, distribute, display, and perform your work --- and derivative works based upon it --- but for noncommercial purposes only.

Examples: Gus publishes his photograph on his website with a Noncommercial license. Camille prints Gus' photograph. Camille is not allowed to sell the print photograph without Gus's permission.

No Derivative Works. You let others copy, distribute, display, and perform only verbatim copies of your work, not derivative works based upon it.

Example: Sara licenses a recording of her song with a No Derivative Works license. Joe would like to cut Sara's track and mix it with his own to produce an entirely new song. Joe cannot do this without Sara's permission (unless his song amounts to fair use).

Share Alike. You allow others to distribute derivative works only under a license

1 The following text describing Creative Commons comes from http://creativecommons.org/about/licenses

identical to the license that governs your work.

Note: A license cannot feature both the Share Alike and No Derivative Works options. The Share Alike requirement applies only to derivative works.

Example: Gus's online photo is licensed under the Noncommercial and Share Alike terms. Camille is an amateur collage artist, and she takes Gus's photo and puts it into one of her collages. This Share Alike language requires Camille to make her collage available on a Noncommercial plus Share Alike license. It makes her offer her work back to the world on the same terms Gus gave her.

Sampling[1]: People can take and transform pieces of your work for any purpose other than advertising, which is prohibited. Copying and distribution of the entire work is also prohibited.

Sampling Plus People can take and transform pieces of your work for any purpose other than advertising, which is prohibited. Noncommercial copying and distribution (like file-sharing) of the entire work are also allowed. Hence, "plus".

Developing Nations[2]: The Developing Nations license allows you to invite a wide range of royalty-free uses of your work in developing nations while retaining your full copyright in the developed world.

3.2. Notification and Enforcement of Creative Commons Licenses

There are two possible types of support: notification and enforcement. Notification that content is under a Creative Commons license is straightforward to implement and clearly follows the spirit of Creative Commons licensing. Enforcement is more complicated.

Enforcement of Creative Commons is challenging, How can one implement Creative Commons licenses in a digital-rights management system, which, by its very nature, restricts usage? Would any kind of restrictions be appropriate? What information should be exchanged between client and server? What kind of user information would the server collect (since the philosophy of encouraging sharing and reuse argues for privacy-preserving DRM system)? How does one ensure that enforcement is done according to the content creator's wishes? One person's requirements for attribution may be quite different from another's. These are the issues we wrestled with as we sought to implement Creative Commons licenses in a DReaM.

It was not difficult to see that enforcement of certain Creative Creative licenses (notably Noncommercial, Share Alike, Sampling, Developing Nations,which are described in the next section) simply does not make sense in any realistic model of client devices. Enforcement would either be technically infeasible or too heavy handed, in terms of software or hardware requirements, to be reasonable. For two

1 This is taken from http://creativecommons.org/about/sampling
2 This is taken from http://creativecommons.org/license/devnations

other licenses --- Attribution and No Derivative Work it seemed at first that it was possible that a limited form of enforcement could be done in a DRM system. However, closer examination showed that not to be the case.

Consider attribution. It is our belief that any attempt at enforcement would require substantial capabilities of the client. The difficulty is that humans are extremely flexible and the issue thus becomes to have the client recognize that attribution was being given in a way acceptable to the content owner. Does the attribution fit the author's requirements (which may be much less than usual or more specific or even just peculiar). If not, enforcement of the Creative Commons license would have to prevent sharing of the content. An example demonstrates the issues involved.

The common flavor of attribution is by content creator's name: Thus for the quote "But, in a larger sense, we can not dedicate -- we can not consecrate -- we can not hallow -- this ground" the natural form of attribution would be Abraham Lincoln (one might also include that the words are from the Gettysburg address, but that is not necessary). However, another, perfectly acceptable, form of attribution would be to state "this is by the sixteenth president of the United States." How does a piece of software know that the latter form of attribution is sufficient? The content owner may find many forms of attribution acceptable: Abraham Lincoln, A. Lincoln, sixteenth president of the United States (or Vincent van Gogh, van Gogh, V. van Gogh, Vincent).

One could posit intelligence on the part of the client, but even though software design should be as general as possible, positing that type of artificial intelligence capabilities on the client appears unreasonable. In particular, because the philosophy underlying Creative Commons licenses concerns sharing and reuse of content, **it is important not to create technical impediments to usage where none currently exist**. So while we would have preferred to enable enforcement capabilities in the Creative Commons specification --- thus utilizing the full capabilities of the DReaM system --- this is simply not appropriate. It was too likely to impose limits where none existed in the Creative Commons licenses themselves. Therefore the DReaM implementation of Creative Commons licenses is limited to notification.

4. **Abstract Model**

We seek to provide a simple, privacy-preserving scheme for implementing Creative Commons licenses in a DRM system. In particular, the specifications have been written to minimize the amount of information needed from the client.

What follows is a technical description of the components of the MMI Request and Response used to support Creative Commons licensed content. This is how all Creative Commons clients will interact with a DReaM service.

The basic model used in this profile is the same as the generic DReaM model. Clients form Requests for content specifying Verbs (the actions they want to take on content), and Responses are returned that MAY include Hints (input from the server about why

an action is allowed or denied). These Hints MAY force the Client to send a new Request. All server-side decisions are based on policy derived from a Creative Commons license.

Creative Commons licenses have been designed with privacy in mind, and so design decisions in this profile were made in that spirit. This means that the model does not support complex attributes for making access decisions, something that is supported by the DReaM model itself. It also means that specific Verbs are not used to describe direct interaction with content (see Section 5). Generally, the MMI protocol is used to model a Trust Negotiation protocol, where the two parties (Client and Server) need share only a minimal amount of detail to start a negotiation, and can use multiple rounds of exchanges before finally agreeing on how and whether to share content.

In this profile, Hints are used to ensure that a Client can always start with as generic a Request as possible. This maintains some privacy and flexibility for clients. In other words, the initial Request for content need contain almost no detail about why the Client wants a given piece of content or what the Client intends to do with the content. If the policy doesn't need any further detail, then access is granted. If more information is needed, then the Client is prompted for this detail, and the Client can choose whether to provide the information or give up asking to share the content. This profile also defines a mechanism for use when Hints are not supported that nonetheless preserves the Client's privacy.

In this profile we require only one identification mechanism be supported: anonymous access (see Section 6). While a system MAY choose to implement other schemes, all systems supporting Creative Commons licenses MUST support anonymous access. This is to ensure that clients have privacy protection. An initial Request for content MAY always specify anonymous access. If the system wants to use stronger authentication, then this is specified in the Response, and the Client has the choice of whether to proceed.

What follows this section are the normative sections that detail each aspect of a compliant system.

5. MMI Requests

This profile defines two aspects of MMI Requests: the Verbs available and the way they are specified. This section defines these two aspects. For detail about how the server responds to these Requests, see Section 9.

5.1. Creative Commons Verbs

Creative Commons licenses are applied only at one point: when content is shared. To this end, this profile only considers the action of sharing content. Because Creative Commons does not define low-level access restrictions (like the ability to read, fast-forward, or backup media), other Verbs defined in MMI or elsewhere are not used with this profile, and MAY NOT be included in a Request that uses Creative

Commons Verbs. This is to conform with Creative Commons usage, which does not permit the restricted use of content except as specified by the Creative Commons licenses.

The following standard Verbs are defined by this profile:

SHARE-COMMERCIAL	Request the right to share for commercial purposes
SHARE-DERIVATIVE	Request the right to share a derivative work
SHARE-NO-NOTICE	Request the right to share without notice
SHARE-NO-SOURCE-CODE	Request the right to share without the content's source code
SHARE-NON-ATTRIBUTION	Request the right to share without attribution
SHARE-NOT-ALIKE	Request the right to share with a different license

Note that there are no Verbs for Reproduction or Distribution (see Section 8) because these are implicit in the act of Sharing Creative Commons licensed media. While these are explicitly identified in license terms, and therefore encoded into policy for completeness, there is no meaningful way to ask to Share resources without these two rights.

These Verbs MAY be used individually or collectively, as described in Section 5.2. They are based on standard sharing concepts from Creative Commons.

Additionally, a generic Verb is defined:

SHARE [Optional Parameters]

The optional Parameters provided to this Verb define how the Client wishes to share the content. This is useful when new Verbs are defined by Creative Commons that have not yet been profiled in an updated version of this specification. In the case that Hints are supported, the Server MAY communicate these new Parameters to the Client. The Client can also request these new rights directly.

A Client MAY use the generic Verb SHARE with no Parameters. This has the same meaning as requesting all rights. This is most useful when Hints are supported, as is explained in Section 9.2.

Note that there is no specific verb to ask for no special rights when sharing content; this is implicitly allowed. Because this profile only deals with Notification, and does not try to address any aspects of enforcement, there is never the need to ask for

permission to share content with no specific rights addressed in Creative Commons licenses.

5.2. Creative Commons Request Formats

All Requests use the MMI protocol and are therefore required only to follow the rules defined in the previous Section about Verbs. In order to achieve specific goals such as preservation of the Client's privacy and efficient policy processing, there are three canonical mechanisms for formatting Requests.

1. Multiple Elements	In this approach all Verbs are specified, each in its own MMIRightsRequestElement. If un-profiled Verbs are known they SHOULD also be included, each in its own element. This has the effect of revealing no user preference as long as a Client always starts with this Request. This is typically used only when Hints are not supported. While inefficient in space and time, this implementation will always protect the user's privacy while revealing the permissible sharing actions.
2. Single Element	In this approach, a single MMIRightsRequestElement is used that includes any Verbs being requested. This group of Verbs will be collectively Granted or Denied. This is typically used when the Client knows which specific rights they need, and the Client is not concerned about disclosing this information. This is often the case when the Client is actually part of the Server infrastructure, or if this Request is being made after an initial Request and Response that specified which Rights are allowed.
3. Use only SHARE	In this approach, a single MMIRightsRequestElement is used that includes only the Verb SHARE with no parameters. This is a Request for all Rights, and is functionally equivalent to a Single Element that explicitly includes all Verbs. This is typically used only when Hints are supported, since a Response of Deny leaves the Client with no detail about how to proceed. See Section 9.2 for details on how this works.

Because there is no initial disclosure about the intended use of the content if these strategies are always followed, the first and third approaches above protect client

privacy: In essence, this is how most Creative Commons licensed content is accessed today (since most licensed work is available off unprotected web pages). Note that the second approach has the same property if all Verbs are included, but as noted, there is no reason to do this since the third approach is a more compact representation of the same Request.

Should the server respond with any Hints about the client Request (see Section 9.2), the client MAY send a new Request. In a follow-up Request the client specifies permissible Verbs to explain in more detail the conditions for their access request, or what license aspects they recognize must be upheld.

This exchange requires no interaction with the user. Based on the Hints provided in the initial Response, this process can be automated. Indeed, it's possible to automate any number of rounds in this protocol without prompting the user. That said, a typical client implementation will likely prompt the user or require the user to provide preferences about acceptable use ahead of time. Typical client implementations MAY also notify the user of the conditions on use of the content, and allow the user to opt-out of requesting the content if the restrictions aren't amenable. See section 10 for details of required and typical client behavior.

6. Authentication

Access to content protected by Creative Commons licenses is **never** decided based on the user's identity, so it is never necessary to authenticate the user. Thus the only authentication mechanism required for DReaM support of Creative Commons licenses is for anonymous transactions.

To use the anonymous authentication provided by this specification, an MMI Request specifies an AuthServiceID value of:

urn:[insert our name-space here]:anonymous

This mechanism MUST NOT prompt the client for any identifying details, and MUST NOT provide any details to the rest of the DReaM system for use in Request authorization. If any kind of tracking or pseudo-anonymous access is required, a custom mechanism MUST be used.

7. Creative Commons License Management

The Creative Commons project has already defined mechanisms for encoding license terms in a machine-readable format and including this detail in certain content types. The following two sections define how these definitions are used by DReaM. The goal is to reuse existing mechanisms to maintain compatibility and simplicity.

In Creative Commons licensed content, license terms are provided through metadata. This metadata MAY use URLs to point to an online copy of the license being used, or there MAY be license properties in the metadata itself. The level of detail depends in

part on what is allowed by the given media format. For example, HTML data can include a single URL pointing to a license or richer RDF tags describing what the license permits, prohibits, and requires. When an application downloads CC-licensed content, the application can look in the content for license detail.

When content is added to the DReaM system for management, the content may already include Creative Commons licensing terms as previously described. If so, this license information is used for that content. If no licensing terms are embedded, then licensing information MUST be included separately. All conforming systems MUST support using the standard RDF definitions for license terms. Additionally, systems are free to use other definition formats.

Note that license terms are not required to be included in any content; they MAY be kept separate as long as DReaM can maintain the relationship. If licensing metadata is included in content, it SHOULD only be a pointer to the license terms, and not the terms themselves. This is in keeping with the DReaM model in which Responses and content shared with the Client include neither detailed license terms nor REL statements. If the content was provided to the DReaM system with embedded terms, these MAY be kept in the content or extracted and held separately.

If content is provided to the Client with Creative Commons metadata pointing to a license, the license MUST be resolvable by the Client. The canonical mechanism is via a URL. The DReaM system MAY choose to use another mechanism.

License detail MUST be provided to the Client using the standard Creative Commons formats. Additionally, custom formats MAY also be used.

8. **Deriving Policy from Creative Commons Licenses**

The Creative Commons licensing terms are used by DReaM to define policies for sharing content. These terms are taken directly from the Creative Commons RDF definitions[1] to simplify the mapping. Note that the Creative Commons tags define license elements in three categories: Permissions, Prohibitions, and Requirements.

In the case of Permissions, if an element is present in the license then it need not be encoded in the policy. For instance, if the permission for Derivative Works is present, then it doesn't matter if the Client wants to make a Derivative work or not, so the policy doesn't care. If an element is not present in the license, then it is represented in the policy. For instance, if the permission for Derivative Works is not present, then the policy MUST specify that Derivative Works are not allowed.

In the case of Prohibitions and Requirements, if an element is present in the license then it is also encoded in the policy. For instance, if the requirement for Attribution is present, then the policy MUST specify that Attribution is part of the Request.

1 http://creativecommons.org/technology/metadata/implement

Permissions (when absent)	Reproduction, Distribution, DerivativeWorks
Prohibitions (when present)	CommercialUse
Requirements (when present)	Notice, Attribution, ShareAlike, SourceCode

Note that while these are the core definitions, other terms MAY be defined by Creative Commons or other parties. An application SHOULD NOT fail to interpret these. In the case of Permissions, the system doesn't need to support arbitrary new terms, since doing so is generally impossible.

Rules in these policies not directly derived from the Creative Commons license terms MUST NOT be included. Doing so would effectively define new terms for the content that MAY not equate to a valid Creative Commons license.

Because these policies are internal to DReaM, a given system is free to choose how they express their policy. The policy MAY be expressed using the existing Creative Commons formats or any other language. Regardless of how the policy is expressed, it MUST be evaluated completely when the corresponding content is requested.

9. MMI Responses

This section defines how Responses are formed and interpreted. Responses MAY include Hints to help optimize communication. However, because support for Hints is optional, a basic mode is also defined that does not use Hints in Responses. Responses MAY Grant a sharing request, or MAY require the Client to issue a new Request.

When the DReaM system receives an MMI Request to share Creative Commons licensed content, the first action is to check the policy protecting that content. The policy is defined as described in Section 8. Based on evaluating the policy, a Response of “denied” or “granted” is returned, possibly including Hints as described below. If a Response of “denied” is returned, then the Client MAY send a new Request based on details in the Response. Reaction to the policy is as follows.

Note that the terms “granted” and “denied” are used here, as these are the terms used by MMI. In all cases, these are not enforced access decisions, but instead notification of what the Creative Commons license allows. It is still at the Client's discretion how to identify the rights it needs and how to react to a given MMI Response.

9.1. Basic Responses (no Hints)

If Hints are not supported, then Responses simply communicate the standard MMI result for each Request. In the case of approach 1 from Section 5.2 this means that each element is separately granted or denied, so that the Client knows exactly which sharing actions are allowed. In this case the Client does not need to issue a new Request, since all permissible sharing actions are known.

With approaches 2 and 3 in Section 5.2, where only a single element is used, only a single response of granted or denied is returned. This means that, if the Request is denied, the Client doesn't know what aspects of the Request were denied and therefore has no insight into how to form a new Request. The Client MAY try a new combination of Verbs, although this may provide no more useful feedback, and doing so may reveal aspects of the user's preference. As discussed in Section 5.2 this will be acceptable to some clients but problematic for others.

Note that if the Request is granted the Client doesn't know what additional actions are allowed. In practice this is not often a problem. Most Clients that need this detail will want to know all permissible actions, and will gather this by asking for all rights, or by inspecting the license directly.

Note also that if approach 3 (a single element with the generic SHARE Verb) results in a Response of granted, this doesn't actually communicate to the Client all allowed sharing actions. This is because the SHARE Verb will not reveal to the Client a new Verb or Parameter about which the Client is unaware. Again, this is rarely a problem in practice. To ensure that the Client knows exactly which rights are granted, use Hints as described in the next section.

Hints MAY be used to communicate permissible sharing actions as described in the next section. Even when Hints are unavailable, allowed sharing actions MUST be communicated to the Client by, e.g., communicating the license details directly). The license data MAY be included in the content, or MAY be referenced by URL from the content.

9.2. **Optimized Responses (Hints)**

When Hints are available, there are several optimizations available. It is strongly recommended that Hints be used. They help with performance and with the negotiation capabilities of the protocol. The basic optimization provided by Hints is that the Client can ask for more rights than it needs, and the Server will respond with Hints that help the Client narrow its Request. Hints will always name Verbs defined in Section 5.1.

Note that when Hints are available, approach 1 from Section 5.2 cannot be optimized since Hints will never be used. When Hints are available, approach 2 can be optimized and provides the same information from approach 1 if all known Verbs are included in the single Request element. The Response will include “CanDo” and/or “CannotDo” Hints (see below), and MAY require a new Request to communicate specific actions.

However, with Hints available, it is suggested that approach 3 from Section 5.2 be used, which is cleaner and (arguably) optimal. This also expresses a request for all rights, but in the smallest possible Request. The Response will include “CanDo” and/or “CannotDo” Hints (again, see below) and the Client can then decide whether to send a new Request for specific rights. Note that this has the additional benefit of communicating to the Client all allowed Rights, some of which the Client may not know exist (i.e., if they haven't been profiled yet or if the Client hasn't been updated).

These MAY be expressible to the User.

Note that approach 2 is still useful as an initial Request form if the Client is only interested in a specific subset of rights and is willing to communicate this. Again, using Hints the Server tells the Client which of this subset of Rights is allowed. Note also that approach 2 is typically the form of any follow-up Request. That is, once an initial Request is responded to with Hints, the Client typically issues a new Request with a sub-set of Verbs in a single Request element.

In each of these cases, the Server can decide whether the Hints supplied are “CanDo”, “CannotDo”, or both. If the Client knows of all possible rights, then the Server can use only one type of Hints and communicate everything to the Client. Because this may not be the case, however, it is strongly suggested that the Server SHOULD always include “CanDo” Hints. The Server MAY also include “CannotDo” Hints, although in practice these are rarely needed given the “CanDo” Hints.

10.Client Behavior

A conforming Client MUST understand the MMI protocol and all of the specific logic defined in the previous sections. A Client MUST also communicate effectively with its user. While the details of how the Client handles these actions is outside the scope of this specification, this section does specify the nature of this communication, and some of the common features that a Client SHOULD provide its users.

How a Client determines what sharing rights it Requests is outside the scope of this specification. For example, it may be that a Client application knows how to recognize that a given piece of content has been used to make a derivative work. It is also valid to assume a Client application that has no such knowledge, and relies on the User to identify which rights they need. In all cases, the Client sends a Request to learn if the rights they are interested in exercising in sharing some content are allowed by the Creative Commons license.

When a Response is returned to a Client, it may specify a result of “granted” or “denied”. In either case, this is a notification of the license terms that MAY be shown to the User in some manner or MAY simply be interpreted by the application. If the result is “denied” then the original Request was for sharing rights that aren't granted by the Creative Commons license, and in this case the Client MAY try sending a new Request.

Note that in the ”denied” case, the typical result is that the Client will either form a new Request or notify the user of failure. The decision of whether to try a new Request is the user's. That is, even if it is technically possible to form a new Request that will meet the resource's license terms, the user SHOULD decide whether or not they want to do so (for example, for privacy reasons the user may wish to stop at any point).

In order to keep user interruptions to a minimum, it is strongly suggested that Clients manage some preferences on the user's behalf. This means that decisions about issuing

new Requests can be made without prompting the user. It is valid though discouraged to actually prompt the user with each step in the negotiation (such behavior tends to be overly burdensome for users). Client implementations are free to define their own mechanisms for managing user preference, as this is usually tightly coupled with the notification mechanism.

In the event that a Request cannot be formed after a Response, the user SHOULD be notified of the failure. The notification SHOULD contain some detail explaining why the negotiation failed.

Note that if the Response returns with no Hints, there will be a pointer to the license associated with the content. This license contains all the same information that Hints would have provided. It is strongly recommend that a Client fetch the license when no Hints are provided, and use this to notify the user about the license terms. This is effectively what current Creative Commons applications do.

In addition to notifying the user about the result of a Request, it's also valid to fetch the associated license before asking for the content. The license can be used on the client side to determine if the terms are amenable to the user. This MAY be done based on existing preference, or by prompting the user. This can help expedite the negotiation with the server, and further protect the privacy of the user (where some Requests will never be sent if it is known that the client won't accept the terms of the license).

11. **A Complete Example**

Consider a request for song1.mp3. This content is licensed with permission for reproduction and distribution, but not derivative work, and requires attribution[1]. In its RDF form, this looks like:

```
<rdf:RDF xmlns="http://web.resource.org/cc/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <Work rdf:about="http://example.com/song1.mp3">
        <dc:title>Song Number 1</dc:title>
        ...
        <license rdf:resource="http://creativecommons.org/licenses/by-nd/2.5/"/>
    </Work>
    <License rdf:about="http://creativecommons.org/licenses/by-nd/2.5/"/>
        <permits rdf="http://web.resource.org/cc/Reproduction"/>
        <permits rdf="http://web.resource.org/cc/Distribution"/>
        <requires rdf="http://web.resource.org/cc/Attribution"/>
    </License>
</rdf>
```

Let's start with the case where the Client doesn't know if the server supports hints:

MMIVersion = 1.0

1 This is known as the "Attribution No Derivatives" license or "by-nd", and can be found in more detail at http://creativecommons.org/licenses/by-nd/2.5/

```
MMIRightsRequest
    IdentitySegment
        AuthServiceID = urn:[insert our namespace here]:anonymous
    RightsSegment
        ProfileID = urn:[insert our namespace here]
            MMIRightsRequestElement
                ReqElemId = 1
                ContentId = urn:com:example:songs:song1.mp3
                VerbElement
                    VerbElemId = 1
                    Verb = SHARE
```

Now, assume that the server does not use hints in responses (we'll go through the hints example later). The response that comes back is fairly meaningless:

```
MMIVersion = 1.0
MMIRightsResponse
    Status = RequestOK
    ResponseId = 1
    RightsResponseElement
        ReqElemId = 1
        Notification = denied
```

The reason the Request is denied is that by using the SHARE verb, the Client requested all possible rights, including permission to share derivative work. This is disallowed by the license.

The Client now wants detail about which sharing actions are allowed. It can learn this either by calling out the specific sharing actions that are desired, or by enumerating all actions. Since the latter is the method that preserves privacy in this exchange, let's look at that request:

```
MMIVersion = 1.0
MMIRightsRequest
    IdentitySegment
        AuthServiceID = urn:[insert our namespace here]:anonymous
    RightsSegment
        ProfileID = urn:[insert our namespace here]
            MMIRightsRequestElement
                ReqElemId = 2
                ContentId = urn:com:example:songs:song1.mp3
                VerbElement
                    VerbElemId = 1
                    Verb = SHARE-COMMERCIAL

            MMIRightsRequestElement
                ReqElemId = 3
                ContentId = urn:com:example:songs:song1.mp3
                VerbElement
                    VerbElemId = 1
                    Verb = SHARE-DERIVATIVE
            ...
```

To keep this example from running too long, we have left out the fact that the Request has a separate MMIRightsRequestElement for each of the 6 concrete verbs, as well as extra entries for any known parameters to the generic SHARE verb. These must be in separate MMIRightsRequestElements so that the server can decide on each one separately. Were they included as separate VerbElements in a single MMIRightsRequestElement, then the server would once again respond with a single notification. Of course, with hints available this would be usable, but we're still assuming no hints.

Given this second Request, the server can send back this Response:

```
MMIVersion = 1.0
MMIRightsResponse
    Status = RequestOK
    ResponseId = 2
    RightsResponseElement
        ReqElemId = 2
        Notification = granted
    RightsResponseElement
        ReqElemId = 3
        Notification = denied
    ...
```

The SHARE-COMMERCIAL action is allowed, the SHARE-DERIVATIVE is denied, etc. This tells the Client which of the originally requested sharing actions is allowed.

Now let's move to a system where hints are used. Starting again from the initial Request, the Response is still "denied", but it can now include some useful information about what is allowed:

```
MMIVersion = 1.0
MMIRightsResponse
    Status = RequestOK
    ResponseId = 1
    RightsResponseElement
        ReqElemId = 1
        Notification = denied
        Hint
            HintIndexNum = 1
            Label = CanDo
            VerbElement
                VerbElemId = 1
                Verb = SHARE-COMMERCIAL
            VerbElement
                VerbElemId = 2
                Verb = SHARE-NO-NOTICE
            VerbElement
                VerbElemId = 3
                Verb = SHARE-NO-SOURCE-CODE
            VerbElement
                VerbElemId = 4
```

```
Verb = SHARE-NOT-ALIKE
```

This tells the Client about the complete set of allowed sharing actions, and the Client can now decide whether to issue a new Request to validate its understanding about permissible sharing modes. A second Request might now look like:

```
MMIVersion = 1.0
MMIRightsRequest
    IdentitySegment
        AuthServiceID = urn:[insert our namespace here]:anonymous
    RightsSegment
        ProfileID = urn:[insert our namespace here]
            MMIRightsRequestElement
                ReqElemId = 2
                ContentId = urn:com:example:songs:song1.mp3
                VerbElement
                    VerbElemId = 1
                    Verb = SHARE-COMMERCIAL
                VerbElement
                    VerbElemId = 2
                    Verb = SHARE-NOT-ALIKE
```

The Response, of course, looks like this:

```
MMIVersion = 1.0
MMIRightsResponse
    Status = RequestOK
    ResponseId = 2
    RightsResponseElement
        ReqElemId = 2
        Notification = granted
```

12. **Acknowledgments**

The authors would like to thank Hal Abelson, Ben Adida, Larry Lessig, and Mike Linksvayer for help with the Creative Commons aspects of this work. Their help does not imply any Creative Commons endorsement of these specifications. We would also like to thank Tom Jacobs, Vishy Swaminathan, Gerard Fernando, and all the others responsible for Project DReaM and the Open Media Commons who have helped us with this profile.