2004

Import/Export in Digital Rights Management

R. Safavi-Naini
University of Wollongong, rei@uow.edu.au

N. P. Sheppard
University of Wollongong, nps@uow.edu.au

Takeyuki Uehara
University of Wollongong, takeyuki@uow.edu.au

Publication Details

Copyright ACM 2004. This is the author version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version of the article was published as: Safavi-Naini, R, Sheppard, NP & Uehara, T, Import/Export in Digital Rights Management, in ACM Workshop on Digital Rights Management, 2004, 99-110. Original article available here.

Import/Export in Digital Rights Management

Abstract

The inherently controlled nature of digital rights management systems does little to promote inter-operability of systems provided by different vendors. In this paper, we consider import and export functionality by which multimedia protected by one digital rights management regime can be made available to a multimedia device that supports a different digital rights management regime, without compromising the protection afforded to the content under the original regime. We first identify specific issues to be addressed by developers of digital rights management import/export regimes and outline a variety of methods by which these regimes may be implemented. We then apply our observations to the specific example of import and export of content between the digital rights management regimes defined by the Motion Picture Exports Group and the Open Mobile Alliance.

Keywords

digital rights management, rights expression languages, MPEG-21, Open Mobile Alliance

Disciplines

Physical Sciences and Mathematics

Publication Details

Copyright ACM 2004. This is the author version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version of the article was published as: Safavi-Naini, R, Sheppard, NP & Uehara, T, Import/Export in Digital Rights Management, in ACM Workshop on Digital Rights Management, 2004, 99-110. Original article available here.

ACM Workshop on Digital Rights Management 2004, pp. 99-110

Import/Export in Digital Rights Management[*]

Reihaneh Safavi-Naini Nicholas Paul Sheppard Takeyuki Uehara

Abstract

The inherently controlled nature of digital rights management systems does little to promote inter-operability of systems provided by different vendors. In this paper, we consider import and export functionality by which multimedia protected by one digital rights management regime can be made available to a multimedia device that supports a different digital rights management regime, without compromising the protection afforded to the content under the original regime. We first identify specific issues to be addressed by developers of digital rights management import/export regimes and outline a variety of methods by which these regimes may be implemented. We then apply our observations to the specific example of import and export of content between the digital rights management regimes defined by the Motion Picture Exports Group and the Open Mobile Alliance.

1 Introduction

The increasing availability of network technologies has made electronic distribution an attractive mode of distribution for multimedia content due to the convenience and low cost of copying and distributing digital multimedia. However, this convenience applies equally to legitimate and illegitimate distribution channels, and fears of widespread copyright infringement and other misues of multimedia content are often blamed for the slow uptake of electronic distribution by content owners.

Digital rights management ("DRM") allows content owners to control and monitor the distribution of multimedia content through electronic channels. Content owners' fears of widespread copyright violation via electronic distribution has seen digital rights management become a fast-growing field of research and development in recent years, and a number of systems are now commercially available.

DRM systems, however, do not always provide end-users of multimedia content with the experience they have come to expect in older models of multimedia distribution. In physical distribution channels, multimedia content is bound to a physical object such as a book or compact disc, which can be sold, freely used in any device, shared within a household or amongst friends, and so on. Existing DRM systems, however, offer only much more restricted options for using content – often content can only be used on one particular device or set of devices specified at the time of purchase.

In this paper, we focus on movement of rights-managed multimedia content from one multimedia terminal to another, and, in particular, between two terminals provided by different manufacturers and potentially conforming to two different DRM regimes.

Currently, there exists no standard regime for DRM and existing DRM systems have been developed with little or no regard for inter-operability. At least some providers of DRM and compression technologies are even reported to use incompatibility as a deliberate strategy to lock users into buying particular hardware devices [31] from which the provider makes money.

Furthermore, the nature of DRM does not lend itself to terminal inter-operability. Content is distributed in a protected form that is, by design, inaccessible to any entity that does not conform to the

[*] ©ACM, 2004. This is the author's version of this paper. It is posted here by permission of the ACM for your personal use. Not for re-distribution. The definitive version was published in the ACM Workshop on Digital Rights Management 2004.

DRM vendor's specification. Therefore, users may not be able to make use of content on terminals supporting a different DRM regime, even if they have legitimately acquired the content and the second terminal is from a reputable vendor otherwise trusted by the original content provider.

This leads to a number of inconveniences for all of users, content providers and terminal manufacturers:

- users may not be able to use all of their content on all of their devices;
- content providers may need to supply their content in several different formats;
- content in older formats may not be usable by newer terminals (and vice versa);
- control of the DRM market can be used to distort the market for multimedia terminals by controlling which terminal manufacturers are given access to DRM technologies [1, 31].

In this paper, we will consider *import* and *export* in digital rights management systems, that is, functionality by which content protected by one DRM regime can be converted into content protected by another DRM regime. In this way we can hope to enable transfer of rights-managed content from systems provided by one vendor to systems provided by another vendor, in much the same way as it is possible to convert music on compact disc to music on a hard drive, for example, in order to listen to music on a device equipped with a hard drive but no compact disc player.

We will give an overview of previous work in inter-operabilty in DRM in Section 2, then give an overview and general model of import and export in Section 3. We will develop each component of the model at an abstract level in the following sections, and finally we will consider the specific example of import and export of protected content between DRM systems defined by MPEG-21 and the Open Mobile Alliance in Section 9.

2 Inter-operability in DRM

A number of recent authors [18, 28, 29] have recognised the importance of inter-operability amongst DRM systems in providing an attractive experience to end-users of protected content, and a healthy competitive market for content providers when choosing systems for protecting their content. However, achieving inter-operability in DRM is a difficult task and so far only fairly limited progress has been made in developing actual inter-operable systems.

Koenen, et al. [18] suggest three approaches to creating inter-operable DRM systems:

Full-format inter-operability. All protected content conforms to some globally standardised format.

Configuration-driven inter-operability. End-user devices can acquire the ability to process content protected by any DRM regime by downloading appropriate "tools".

Connected inter-operability. On-line third parties are used to translate operations from one DRM regime to another.

Full-format inter-operability would clearly provide the most convenience for multimedia users, affording them the same convenience that they currently enjoy when using widely-available standardised formats such as the compact disc. However, it is not easy to define a single standard that is appropriate for all conceivable multimedia terminals and it seems inevitable that some conversion of format between terminals with different capabilities will be required – even widely-used standards like the compact disc are not available on every existing playback device, for example, and older standards will one day be replaced by newer ones. Furthermore, a breach in the security of the standardised regime could be catastrophic and standards bodies are not typically able to move with the speed required to effectively respond to security breaches.

Configuration-driven inter-operability is the approach taken by the Motion Picture Experts Group

("MPEG") [28], well-known for its successful MPEG-1, MPEG-2 and MPEG-4 standards for coding audio-visual presentations. MPEG's work in this area is still in its infancy and it remains to be seen how well this approach promotes inter-operability. In particular, it is not clear that all terminals will necessarily be capable of accessing all tools (which might only be available for one particular computing platform, for example) or that all devices, particularly smaller ones, would necessarily have the resources to store and execute all of the tools required to access all of the multimedia owned by one user.

Connected inter-operability is the approach taken by the "Networked Environment for Media Orchestation" ("NEMO") described by Koenen, et al., and in more detail by Bradley and Maher [5]. NEMO establishes a peer-to-peer architecture in which each node exposes an interface standardised by NEMO to its peers. If one node cannot satisfy a user request by itself – it cannot authenticate a given user, or it cannot determine the permissibility of an operation, for example – it conducts a search via its peers hoping to find a node that can. NEMO is a work in progress, and while Bradley and Maher say that

> we are looking at ways to provide an end-to-end interoperable media distribution system that does not rely on a single set of standards for media format, rights management, and fulfillment protcols,

no detail is provided on how NEMO might go about making content protected by one digital rights management regime available to a node that supports a different regime. Presumably this could be achieved by locating a NEMO node with suitable translation capabilities; in this paper, we will establish requirements for translators and suggest options for implementing them.

Another approach to connected inter-operability is the "intermediated digital rights management" of Schmidt, et al. [29]. Intermediated DRM is similar to NEMO in that rights management tasks are performed by a third-party server (the *intermediary*) on behalf of the content providers and end-users, though the architecture proposed by Schmidt, et al. is somewhat simpler than that postulated by NEMO.

Schmidt, et al. identify four tasks to be carried out by the intermediary in transferring content in the format used by the content provider to the format required by the end-user:

Content and Rights Re-formatting. The content, together with the meta-data used to communicate the rights granted to the end-user over it, must be re-written in the format used by the user's DRM system.

Data Management. The intermediary may need to store rights meta-data and provide a convenient method of accessing it.

Condition Evaluation. The intermediary may need to make decisions about how to handle constraints specified by the content provider, but cannot be expressed in the DRM regime supported by the end-user's terminal.

Dynamical State Evaluation. This is an extension of condition evaluation in which the intermediary evaluates constraints on behalf of the end-user's terminal.

Schmidt, et al., however, do not discuss how any of the above tasks might be implemented. We will examine content and rights re-formatting in detail in Sections 5, 6 and 7 of this paper, and condition evaluation in Sections 3.3 and 6.2

Of the three approaches identified by Koenen, et al., the approach taken in this paper is nearest in spirit to the connected approach. We will see in Section 4 that connectedness is not necessarily the defining feature of our approach, however. Specifically, we do not consider the use of on-line third-party condition evaluators and the like that are suggested by NEMO and intermediated DRM – the systems that we consider are designed such that the importing terminal can make use of any content provided to it without on-line assistance.

3 Import/Export in DRM

3.1 DRM Systems

3.1.1 DRM Architecture

Details vary from vendor to vendor, but a typical modern DRM system involves four parties, shown in Figure 1:

a content provider who holds the legal rights to the content;

a distributor who receives content from the content provider and makes it available to end-users;

a user who uses the system to obtain and make use of content; and

a rights issuer who handles any financial transactions or monitoring associated with the issue and use of rights.

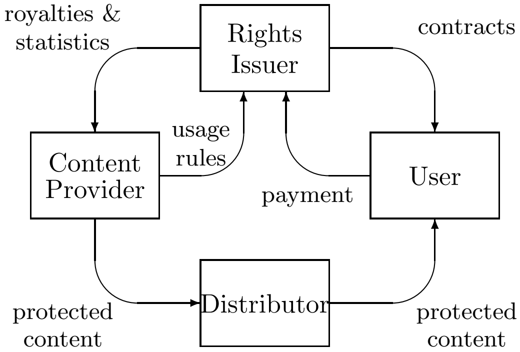

Figure 1: A typical DRM system

Content is distributed in a protected (for example, encrypted or watermarked) form in which it cannot be accessed without a secret key. In order to gain access to the content, a user must obtain a valid *contract* that describes the terms and conditions under which the user may use the content, and contains the keys required to access the content. Contracts will be described in more detail in Section 3.1.2.

The content may only be accessed by a *terminal* trusted to enforce the terms of the contract. The terminal may obtain the keys required to access the content using the contract, but neither these keys nor the unprotected content will be made accessible to the user, except for the analogue output of the terminal. Thus the user can only use the content according to the terms specified in the contract that he or she has bought from the content provider.

Each DRM system will conform to some *digital rights management regime* defined by some particular vendor or standards body. Each regime defines its own set of file formats, encryption methods, authentication protocols, etc. used to implement a DRM system. The purpose of this paper is to consider issues and challenges that arise in making content protected according to one DRM regime accessible to systems supporting a second, different, DRM regime.

3.1.2 DRM Contracts

A contract (also known as a *licence*) is a document written in a machine-readable *rights expressions language* that describes the conditions under which the associated content may be used. Each rights expression language is associated with a *rights data dictionary* that defines the meaning of the terms used in the language. The contract can be interpreted by a multimedia terminal, which ensures that the user's use of the content conforms to the conditions supplied by the rights issuer.

In this paper, we will follow the model and terminology proposed by Guth, et al. [12, 11] in describing digital rights management contracts. A very similar but less comprehensive model is proposed by Chong, et al. [7].

A contract is composed of a collection of *contract objects* that may be of one of five types:

Resource. A unique object associated with some identifier.

Subject. A party to the contract, being either a *rights-holder* who owns the rights over a resource, or a *beneficiary* to whom rights to access a resource can be granted. Each subject is associated with a unique identifier.

Permission. The right to perform a particular operation on a particular resource.

Role. A group to which subjects may belong, and to which permissions may be assigned, not used in this paper.

Constraint. A conditional expression that must be satisfied in order for some permission to be exercised.

A contract is an aggregation of permissions awarded by some rights-holder to some beneficiary. When a user (beneficiary) wishes to perform some particular operation on a particular resource, the terminal checks that the user possesses a contract granting that permission, and that any constraints associated with the permission are satisfied. If the permission does not exist, or the constraints are not satisfied, the terminal will refuse to carry out the operation.

To prevent dishonest users from modifying or fabricating contracts that purport to grant them permissions that they don't have, each valid contract must be digitally signed by the rights-holder. In the model described in Section 3.1.1, the rights-holder in this sense is the rights issuer, who is acting on behalf of the content provider.

Before permitting a user to exercise any permissions contained in a contract, the terminal must verify that a contract carries a valid signature from a recognised rights issuer. If the terminal cannot verify the authenticity and integrity of a contract, it will refuse to exercise permissions supposedly granted by that contract.

3.2 Import/Export

A user may own a range of devices that allow him or her to render a given item of content. In traditional methods of multimedia content distribution, content is stored on a physical object such as a compact disc that can be transported from terminal to terminal as the user wishes. So long as all of the users' terminals conform to the relevent specification for reading the recording, the user may access his or her content using any terminal.

In order to preserve this property in a DRM environment, it needs to be possible for a user to transport his or her content from one device to another. Some systems for transferring content between terminals within a single household are proposed in [20] and [25], for example.

A problem arises when a user owns a range of devices that have the capacity to render a given piece of content, but the devices conform to different DRM regimes. Since content is cryptographically or otherwise protected, a simple format conversion – as might be used to convert music on a compact disc to music on a cassette tape, for example – is not possible, at least, not without the co-operation of some entity with access to the keys required to access the protected content.

In this paper, we consider transfer of content protected according to one DRM regime $\mathcal{D}_1$ to a system supporting another DRM regime $\mathcal{D}_2$. We will refer to $\mathcal{D}_1$ as the *exporting* regime and $\mathcal{D}_2$ as the *importing* regime.

For import and export to be possible in an orderly fashion, the governing bodies of the importing and exporting regimes need to reach some *import/export agreement* that sets out the terms under which import and export is possible, and defines mappings of entities from one regime to another.

Specifically, consider a user who has some protected multimedia object $\mathcal{O}$ associated with contract C on a multimedia terminal T_1, which conforms to regime $\mathcal{D}_1$. Further suppose that the user wants to access $\mathcal{O}$ on a terminal T_2 that does not conform to $\mathcal{D}_1$, but to a different regime $\mathcal{D}_2$. For this to be possible while maintaining the integrity of the protection,

- $\mathcal{O}$ must be converted into a (protected) representation understood by T_2;
- C must be converted to a contract conforming to $\mathcal{D}_2$ that allows the user no greater rights over $\mathcal{O}$ than what is granted by C in $\mathcal{D}_1$; and
- T_1 must establish that T_2 is a trusted terminal that will comply with the terms of the converted contract.

We will discuss the first and second requirements in more detail in the next section. Mechanisms for satisfying all of the requirements must be established by the import/export agreement between $\mathcal{D}_1$ and $\mathcal{D}_2$, and will be discussed in the body of the paper.

3.3 Content and Contract Adaptation

When translating a multimedia object $\mathcal{O}$ and contract C (or portions thereof) from a representation in $\mathcal{D}_1$ to a representation in $\mathcal{D}_2$, we need to consider two possibilities:

Isomorphism. The mapping between $\mathcal{D}_1$ and $\mathcal{D}_2$ is one-to-one, that is, the representation of $\mathcal{O}$ and C in $\mathcal{D}_1$ and $\mathcal{D}_2$ is exactly equivalent; or

Adaptation. The mapping from $\mathcal{D}_1$ to $\mathcal{D}_2$ is an *adaptation* in which the representation of $\mathcal{O}$ and C in $\mathcal{D}_1$ is modified in order to suit the requirements of $\mathcal{D}_2$.

In the first case, there is an isomophism between $\mathcal{D}_1$ and some (not necessarily proper) subset of $\mathcal{D}_2$. Given the representation of $\mathcal{O}$ and C in $\mathcal{D}_2$, it is possible to recover their original representation in $\mathcal{D}_1$ by reversing the mapping.

In general, however, we can expect $\mathcal{D}_1$ to support content and rights expressions that cannot be expressed in $\mathcal{D}_2$, and that some form of adaptation is required to make $\mathcal{O}$ and C representable using $\mathcal{D}_2$. We identify three possibilities for adaptation:

Reduction. Expressions that cannot be expressed in $\mathcal{D}_2$ are deleted from $\mathcal{O}$ and C.

Pre-enforcement. Constraints that cannot be checked or satisfied within $\mathcal{D}_2$ are enforced by the exporting entity in $\mathcal{D}_1$ prior to export, and removed from the representation of C in $\mathcal{D}_2$.

Storage. Expressions that cannot be expressed in $\mathcal{D}_2$ are stored (possibly by a third party, such as the intermediary of Schmidt, et al., that is not one of the terminals involved in the transaction) for possible retrieval when $\mathcal{O}$ and C are re-translated into $\mathcal{D}_1$, but are not available in $\mathcal{D}_2$.

Any particular import/export regime may use a combination of any of the above options for translating any given multimedia object or contract: some contraints (such as payment) may be pre-enforcable, but others (such as constraints with time periods) may not be pre-enforcable and must be deleted, for example.

4 Translation Architectures

Suppose we have two multimedia terminals T_1 and T_2 conforming to digital rights management regimes $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. The user has some protected content stored on T_1, but wants to transfer the content to T_2 and make use of it there. We identify three basic architectures in which this might be achieved.

4.1 Translation Services

Suppose T_1 and T_2 have access to a third party *translation server* that implements the import/export agreement between $\mathcal{D}_1$ and $\mathcal{D}_2$. The translation server is effectively a member of both $\mathcal{D}_1$ and $\mathcal{D}_2$, and can establish mutual trust relationships with terminals from both regimes. Given content protected according to $\mathcal{D}_1$, it can produce content according to $\mathcal{D}_2$ under the terms of the import/export agreement.

The intermediary of Schmidt, et al. discussed in Section 2 is akin to our translation server, except that we don't consider terminals with an on-going connection to the server. That is, our server performs only the "content and rights re-formatting" and "condition evaluation" functions of the intermediary.

4.2 Terminal Translation

Suppose that T_1 has an export function that is able to translate content and contracts from $\mathcal{D}_1$ into the format required by $\mathcal{D}_2$ by itself, or that T_2 has an analogous import function. Then translation between $\mathcal{D}_1$ and $\mathcal{D}_2$ can be handled by one of T_1 or by T_2 without the assistance of the other terminal or a third party. The 'export' function provided by the Open Mobile Alliance's DRM specification [23], for example, implies this kind of architecture though the specification leaves the actual mechanism for export undefined.

This might be a useful architecture where one terminal is more powerful than the other, for example, where content is stored under regime $\mathcal{D}_1$ on a desktop PC T_1 but can be exported to a mobile phone T_2 that supports $\mathcal{D}_2$.

Of course, we could also think of the terminal that performs the export or import operation as being a translation server. However, we make a distinction between the terminal translation architecture and the translation server architecture as there is a signficant practical difference between using one of the user's own terminals as the translator, as compared to having to contact an on-line third party to perform the transaction. Obviously, however, implementing the exporting or importing terminal presents much the same task as the implementation of a translation server.

4.3 Pre-export

Suppose the original content provider and rights issuer supply sufficient information at the time of purchase to enable T_1 and T_2 (either working alone or by co-operating) to construct protected content conforming to $\mathcal{D}_2$. For example, suppose the rights issuer supplies contracts conforming to both $\mathcal{D}_1$ and $\mathcal{D}_2$ to T_1, from which T_1 can construct content conforming to $\mathcal{D}_2$ by re-packaging and its own content.

Of course this requires the content provider to have foreknowledge of what regimes content may be exported to, and increases the amount of storage required by the user. However, it reduces the amount of computation needed to be performed by T_1 and T_2 at the time of transfer, without requiring the on-line assistance of a third party.

As for the terminal translation architecture, we could also think of the pre-export architecture as being a translation server architecture in which the original rights issuer acts as the translation server. Again, however, there is a significant practical difference between a pre-export arrangement and the use of an on-line third party, though implementation of the rights issuer here is much the same as implementation of a stand-alone translation server.

5 Content Translation

In order to convert a multimedia object $\mathcal{O}$ from one recognised by an exporting terminal T_1 to an importing terminal T_2,

- the file format and codec used to represent $\mathcal{O}$ on T_1 must be converted to a file format and codec recognised by T_2; and
- $\mathcal{O}$ must be re-packaged using a protection scheme supported by T_2.

File format and codec conversion is straightforward, and will not be considered further in this paper.

Re-packaging, in general, requires a new key to be generated for the relevent protection algorithm. The details for doing this may vary from system to system but in general we could expect to simply execute the key generation algorithm used by the importing regime. In most cases, therefore, re-packaging does not appear to present a great challenge.

6 Contract Translation

Translation of a contract from one DRM regime $\mathcal{D}_1$ to another regime $\mathcal{D}_2$ seems, in general, to be somewhat more difficult than translation of content. The set of permissions and constraints available in one rights expression language may not be the same as those available in another rights expression language, and even where equivalences exist, it may not be easy to find a mapping between the two.

In general, a single contract document may be associated with multiple resources or multiple contract documents may be associated with a single resource. During translation from one regime to another, it may be necessary (or merely desirable) to convert a single contract document of the exporting regime into multiple contract documents of the importing regime, or vice versa.

For our present purposes, however, we can consider all of the relevent contract documents as if they were a single abstract contract containing all of the permissions granted in all of the actual contract documents. An actual contract document referring to multiple resources, etc., can be similarly considered to be a collection of abstract contracts referring to a single resource, etc. each.

6.1 Rights Expression Modeling

Unless the rights expression languages of the exporting and importing regimes are so close as to make translation trivial, it seems necessary to develop some eneric description to which the exported contract set

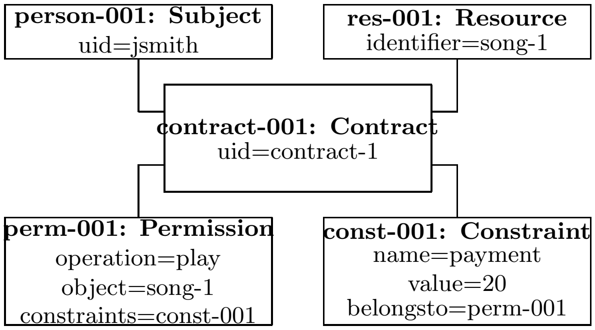

Figure 2: CoSa model for a simple contract

can be reduced, and from which the imported contract can be generated.

6.1.1 Generalised Contract Schema

Guth, et al. [11] present a *generalised contract schema* ("CoSa") which they use as a generic model for accessing contracts written in arbitrary rights expression languages. CoSa represents all of the contract objects referred to by a contract in a linear array. Each object is associated with

contract attributes such as identifier, parameters, etc. that are properties of the contract object itself;

intrinsic attributes that relate one contract object another, for example, by associating a constraint with a permission; or

application-specific attributes that are supported by specific applications but not by CoSa, and require an application-specific extension of CoSa.

Figure 2 shows the CoSa model for a simple contract in which a beneficiary 'jsmith' has permission to 'play' a resource 'song-1', subject to a constraint requiring the beneficiary to pay 20 currency units.

Guth, et al. use a rights expression language (in their case, ODRL [21]) interpreter to build the CoSa model for any given contract, and use the CoSa model to query the permissibility of any given operation under the contract. While the implemention described in their paper could not perform the reverse operation of converting a CoSa model to a contract, they provide an example (presumably constructed by hand) of how an XrML contract [9] can be constructed from a simple CoSa model derived from an ODRL contract.

6.1.2 LicenseScript

Chong, et al. [6] propose a rights expressions language called "LicenseScript". While they presumably intend for LicenseScript to be a more expressive replacement for the widely-known ODRL and XrML languages, they also show how LicenseScript can be used to represent contracts written in either XrML or ODRL [7].

Unlike the established languages, which are based on XML and Stefik's rights model [32], LicenseScript is based on multi-set re-writing and logic programming. A contract is a triple of a resource, a set of *clauses* and a set of *bindings*. The bindings store the state information of the contract – such as the name of the beneficiary, the number of times a particular permission can be exercised, and so on – and can be referenced by the clauses.

Each clause is a logical expression that, if true, permits some operation to be performed, that is, it is roughly equivalent to a permission with constraints in the model of Guth, et al. When a user wishes to perform an operation, the contract interpreter executes a *query* on the relevent clause and, if the result is true, returns a new contract reflecting any changes to the state of the contract (for example, with the number of available uses of the resource reduced by one).

Chong, et al. do not specify any mechanical way of translating their XrML and ODRL contracts into LicenseScript, and the examples given in their paper are presumably constructed by hand. However, it does not seem exceptionally difficult, in general, to convert an XrML or ODRL permission with constraints into a logical expression, since this is more or less equivalent to writing a computer programme that checks the constraints and applies the relevent operation. Given a set of pre-written logic programmes representing each permission and constraint, automatic conversion of an arbitrary contract to LicenseScript seems straightforward.

It is not so easy to see how the reverse operation might be performed, however, since this would require transforming a set of low-level logical expressions into a high-level permission and constraint. Chong, et al. do not consider how LicenseScript contracts might be translated back into XrML and ODRL – in fact, one of the purposes of their paper is to argue that, in general, they cannot be, as LicenseScript is more expressive than the other two. We will leave the development of automated translators as future work.

6.2 Untranslatable Expressions

Unless all rights expressions languages are made to conform to some universal model, it is inevitable that a contract written in one rights expression language may contain expressions that cannot be expressed in another rights expression language, and that some form of adaptation must occur.

If the rights expression language of the importing regime does not support some particular kind of subject, resource or operation, this is presumably because the importing system does not support that kind of subject, resource or operation. For example, a digital rights management regime designed for mobile phones is unlikely to support a 'burn-to-CD' permission since mobile phones do not have CD burners. In this case, it makes sense to simply delete the associated permission from the exported contract (possibly storing it for retrieval during reverse translation), since the importing system cannot make any use of that permission, anyway.

Where the rights expression language of the importing regime does not support some particular constraint, however, the exporting terminal may have the option of pre-enforcing the constraint. For example, if exercising a permission is available in exchange for a kind of payment not supported by the importing system, the exporting terminal may pre-enforce the constraint by charging the payment in exchange for export. The associated permission would then be available on the importing terminal free of charge.

If pre-enforcement is not possible, or the user does not wish to satisfy the conditions for pre-enforcement (for example, does not want to pay), the contract contains a constraint whose satisfaction cannot be checked by the importing terminal. To prevent the associated permission being exercised outside of the constraints specified in the contract, the permission must be deleted from the exported contract.

For example, a simple DRM system might not support limiting the number of times a resource is used. An exporting terminal with a permission constrained to a finite number of uses cannot be certain that the number of uses will, in fact, be constrained if the permission is granted to a terminal implementing only the simple system. To protect the integrity of the constraint, therefore, the permission should be deleted from any contract exported to the simple system.

6.3 Establishing the Integrity of Exported Contracts

Any terminal in a DRM regime is assumed to be able to verify a signature produced by the rights issuer from that regime. The authenticity and integrity of a contract issued by that rights issuer can be verified by any terminal in that system by checking the signature supplied by the original rights issuer on the original contract.

When a contract is exported to a system supporting another regime, however, it is translated to a different representation on which the original rights issuer's signature is no longer valid. Furthermore, the importing terminal does not, in general, have any (direct) trust relationship with the original rights issuer and cannot verify signatures made by it even if they are otherwise valid. In order for the importing terminal to verify an imported contract, therefore,

- a new signature must be generated for the translated contract (in effect, the translator acts as a rights issuer); and
- the authenticity of the new signature must be linked to some entity that the importing terminal trusts.

We consider two cryptographic mechanisms by which this could be achieved (possibly in combination): *certificate chains* and *proxy signatures*.

6.3.1 Certificate Chains

Suppose that, under the terms of the import/export agreement between two regimes $\mathcal{D}_1$ and $\mathcal{D}_2$, the rights issuer of $\mathcal{D}_2$ agrees to sign the public key of some root *translation authorities*. The root translation authorities may further sign the public keys of subordinate translation authorities, which themselves may create their own subordinate translation authorities, and so on. Of course this is the same idea as the well-known X.509 public key infrastructure [17].

The rights issuer of $\mathcal{D}_2$ is then at the root of a tree of translation authorities, each of whom is considered to have been granted the power to translate contracts by its superior in the translation authority hierarchy. We assume that entities will only be made translation authorities if their trustworthiness can be suitably established. For example, in a terminal translation architecture where terminals from $\mathcal{D}_1$ can export content and contracts themselves, the mutual authentication protocol of $\mathcal{D}_1$ could be used for building the translation authority tree. Any translation authority is therefore trusted to create contracts only under the conditions imposed by the import/export agreement between $\mathcal{D}_1$ and $\mathcal{D}_2$.

When translating a contract, the translation authority signs the new contract using its own private key. It then appends its public key together with the signature made by its superior authority, then the public key of its superior authority together with the signature made by the next authority in the hierachy, and so on, until the signature of the rights issuer of $\mathcal{D}_2$ is reached.

In order to verify the authenticity and integrity of an imported contract, a terminal from $\mathcal{D}_2$ first checks the signature of its own rights issuer on the public key of the root translation authority, then the signature of the root translation authority on the first subordinate translation authority, and so on, until it has checked the signature of the translation authority that actually created the contract. If the chain of signatures is broken at any point, the contract is rejected as invalid.

The cost of storing the certificate chain and of checking its validity increases linearly with the depth of the translation authority tree. If the tree is very deep, the amount of storage and computation required by terminals from $\mathcal{D}_2$ could become prohibitive for low-powered terminals.

Of course this system may not work at all if the signature algorithm of $\mathcal{D}_2$ is different from the signature algorithm used in forming the translation authority hierarchy, since in this case terminals from $\mathcal{D}_2$ may not know how to execute the signature verification algorithm used to build the tree.

6.3.2 Proxy Signatures

A *proxy signature* [19] is a signature scheme by which on *original signer* can delegate its power to create signatures to a *proxy signer*. Signatures created by the proxy signer can be verified using the public key of the original signer.

In the present application, the original signer is the rights issuer of the importing system. The import/export agreement between the two systems gives proxy signing keys to any entities trusted to perform contract translation.

Upon translating a contract, the translator signs the translated contract using its proxy key. The signature on the signed contract can then be checked by an importing terminal using the public key of its own rights issuer.

This approach may reduce the amount of storage and computational power required by the importing terminals as compared to use of a certificate chain. However, if a large number of devices have the power to translate contracts – for example, in the terminal translation architecture – the proxy signature scheme would need to support a very large number of proxy signers.

6.3.3 A Note on Pre-export Architectures

In a pre-export translation architecture, the entity performing the export operation is not, in general, the same entity as the one that creates the translated contract.

In the simplest case, the contract for the importing regime $\mathcal{D}_2$ is created by the rights issuer of the exporting regime $\mathcal{D}_1$, and signed by it using a certificate chain or proxy signature to link the new signature to the rights issuer of $\mathcal{D}_2$. The signed pre-exported contract is then transmitted along with the usual contract for $\mathcal{D}_1$ to the user's terminal (which is from $\mathcal{D}_1$). If the user wishes to export the associated content to a terminal T_2 belonging to $\mathcal{D}_2$, T_1 simply copies the pre-exported contract, and the associate signature, to T_2.

We can also postulate a more sophisticated case in which the rights issuer of $\mathcal{D}_1$ does not provide a whole contract for $\mathcal{D}_2$, but a set of "contract modules" from which one of the terminals involved in the actual translation operation can construct a complete contract for $\mathcal{D}_2$ according to circumstances. This might allow, for example, a user to choose whether to accept pre-enforcement of a constraint or to have the associated permission deleted at the time of translation. For this to be possible, the exporting terminal must have some way of creating a signature for the constructed contract from the signatures on the contract modules provided by the rights issuer of $\mathcal{D}_1$ (or, conceivably, by the rights issuer of $\mathcal{D}_2$ if $\mathcal{D}_2$ provides signed contract modules as part of the import/export agreement).

An approach by which this could be achieved is suggested by Boyer [4], who proposes *document subset signatures* as a method of creating signatures on documents that may undergo certain restricted changes. Boyer specifically considers documents formed by taking geometric areas of forms input by users, but the signature structure is valid for any situation in which a document might be created by taking a subset of another one.

A document subset signature is a signature computed on some subset of a document – in Boyer's case, for example, a set of elements from an XML document. By choosing a collection of appropriate subsets of a document, and providing subset signatures on each of them, it is possible for a party to verify the integrity of a document constructed from the original document, so long as the document is constructed in an acceptable way.

For example, in order to give users the choice between pre-enforcement or deletion, two document subset signatures could be provided: one for the contract after pre-enforcement (which contains the relevent permission), and another for the contract after deletion (which does not). We will leave the development of sophisticated pre-export systems with contract modules as future work.

7 Identifier Translation

Many expressions in rights expressions languages bind permissions to specific individuals, terminals or other objects using some identifier specified in the contract. When exporting the contract to another digital rights management regime, each identifier in the original contract may need to be converted to an identifier in the importing system that refers to the same object.

In some cases, this may be trivial: if some universal labeling scheme (or, at least one common to the two regime) is available, conversion of identifiers in one regime to identifiers in another regime may be a simple format conversion. For example, if users or devices are authenticated by their possession of the private key corresponding to a public key specified in the contract, and both systems support the same public key encryption algorithm(s), it is trivial to translate identifiers between regimes.

In other cases, things may be more difficult or even impossible. The contract to be exported may, for example, refer to objects of a kind not recognised at all by the importing regime, or there may be no obvious way of securely mapping an object's identifier in one regime to its identifier in another regime.

In general, we would expect that the import/export agreement between two digital rights management regimes would need to specify a mapping between the identifiers used within each regime. When an item of content is exported, the entity performing the export would then need to apply the mapping from the agreement.

In the case where an identifier in the exporting regime has no equivalent in the importing regime, an untranslatable expression is created. Such an expression can be discarded or handled as described in

Section 6.2.

8 Mutual Authentication of DRM Systems

DRM systems require that content be accessed only by trusted terminals. A regime $\mathcal{D}_1$ presumably has some way of identifying remote terminals trusted within its regime, but in order to securely export content to another regime $\mathcal{D}_2$, it needs to establish

- that $\mathcal{D}_2$ is a reputable regime trusted by $\mathcal{D}_1$ to handle the exported content; and
- that a particular importing terminal T_2 is a trusted terminal according to $\mathcal{D}_2$.

The former requirement would presumably be satisfied by the existence of an import/export agreement between the governing bodies of $\mathcal{D}_1$ and $\mathcal{D}_2$. The entity responsible for checking the latter requirement, which is an entity trusted by $\mathcal{D}_1$, would act only under the terms of this agreement.

The latter requirement could be checked using schemes similar to the ones described for contract verification in Section 6.3, so that each terminal in $\mathcal{D}_2$ can compute a signature that will be recognised and accepted by a terminal from $\mathcal{D}_1$. However, this would require every terminal from $\mathcal{D}_2$ to be able to compute signatures from $\mathcal{D}_1$, which may not be practical in some scenarios. In the remainder of this section, we consider several other mechanisms that might be used for checking the latter requirement, and do not require devices from $\mathcal{D}_2$ to be able to compute signatures from $\mathcal{D}_1$.

8.1 Using a Device Trustworthiness Layer

Suppose that both the exporting and importing regimes are built on top of a common specification for trusted devices – such as the prominent specification developed by the Trusted Computing Group [30] – so that the trustworthiness of any device is checked in the same way regardless of which digital rights management regime it belongs to. Then it should be possible for the exporting and importing devices to check each others' trustworthiness directly using the mutual authentication protocol of the underlying trusted device specification.

Of course, one might argue that this is simply moving the problem to another layer since there doesn't seem to be any reason to believe that trusted devices should be any more inter-operable than DRM systems. Nonetheless, an architecture in which device trustworthiness can be checked independently of a specific digital rights management regime seems as valid an approach as any, even if lack of a universal standard for trusted devices makes this approach less straightforward than that what is suggested by the above.

8.2 Using a Trusted Third Party

Suppose that the exporting and importing devices have access to a third party trusted by both $\mathcal{D}_1$ and $\mathcal{D}_2$ to implement the import/export agreement between the two regimes. The two devices can check the trustworthiness of the third party using the mutual authentication protocol from their own digital rights management regime. Each device can therefore establish the trustworthiness of the remote device by virtue of its trust in the third party which vouches for the remote device.

9 Import/Export Between MPEG-21 and OMA

Probably the most prominent standardisation activities in the digital rights management field are

- the Motion Picture Experts Group's ("MPEG") MPEG-21 Multimedia Framework standard [3, 13]; and
- the Open Mobile Alliance's ("OMA") Digital Rights Management standard [23].

The former seeks to support a full-featured DRM environment in which virtually any function can be implemented, while the latter seeks to support only a relatively simple environment suitable for exchange of protected content on mobile phones and other portable devices.

It is easy to imagine a scenario in which a user has a library of content stored on his or her home computer, and wants to make use of the content on a mobile phone when travelling. Since the home computer is a full-featured terminal, it is unlikely to be an OMA terminal and the content on it may be stored using a full-featured system such as MPEG-21. The mobile phone, however, is likely to be an OMA terminal unable to process full-featured MPEG-21 material. Thus the user would like to export the MPEG-21 content to the OMA regime so that the content can be used on the mobile phone.

Export from the OMA device to the MPEG-21 device may also be desirable, for example, if the user purchases content in OMA format from his or her mobile phone provider and wishes to consolidate it with the MPEG-21 library on his or her home computer.

In this section, we will give an overview of the MPEG-21 and OMA regimes, and then consider the observations we made in earlier sections in the specific context of import and export between the MPEG-21 and OMA regimes. We will give only an outline of an import/export system here, and identify specific mappings that would need to be determined by an import/export agreement beyond the scope of this paper.

9.1 MPEG-21

The MPEG-21 Multimedia Framework standard seeks to define a generic framework for storing, distributing and using multimedia presentations. Unlike previous MPEG standards, it does not define the way in which individual multimedia presentations are encoded, but defines ways in which atomic multimedia objects can be combined, navigated and referenced. It consists of numerous parts, some of which have been ratified by the International Standards Organisation as the ISO/IEC 21000 series of standards, while others remain under development.

The core notion in MPEG-21 is the notion of a *digital item* [14], which represents a collection of multimedia objects related in some way. The objects within a digital item may be subject to *intellectual property management and protection* ("IPMP") [28], which is MPEG's term for digital rights management.

MPEG-21 takes the "configuration-driven interoperability" approach of Koenen, et al. [18] and seeks to define only the interface between an MPEG-21 terminal and third-party "IPMP tools" that implement some specific digital rights management system. In this respect, import into MPEG-21 can be seen as having no meaning since, in principle, an MPEG-21 terminal can be made to access any DRM regime by supplying it with appropriate IPMP tools.

On the other hand, MPEG-21 does define a specific rights expression language known as MPEG REL [16], based on the eXtensible Rights Markup Language ("XrML") [9]. We will focus on translation of MPEG REL, and the regime implied by it, in our present analysis.

In MPEG REL, a contract (called a *licence* in their terminology) is a collection of *grants*, each containing a permission in the terminology used in this paper. Each grant awards some *right* over some specified *resource* to a specified *principal*, i.e. beneficiary. Each grant may be subject to one or more *conditions*, i.e. constraints.

The *principal* and *resource* elements in MPEG REL are abstract types that must be instantiated by some concrete type implementing a particular method of identifying a principal or resource. In principle, therefore, it is possible for MPEG REL licences from different DRM systems to specify principals and resources in different ways. However, MPEG REL does provide the `keyHolder` and `diReference` concrete types for `principal` and `resource`, respectively. As these types seem likely to be used in many MPEG-21 implementations by due by virtue of their standardisation, we will use the `keyHolder` and `diReference` types in our discussion.

MPEG-21 does not explicitly support export of protected content to another regime; presumably the MPEG committee intends for import, at least, to be unnecessary by virtue of its configuration-driven architecture. MPEG REL does, however, define an 'export' right which, if granted, allows the user to make an unprotected version of the protected content. This is not truly export in the same sense as we use the term in this paper, though it could be argued that the right to create an unprotected version of the content implies the right to create a version suitably protected by another DRM regime.

For the purposes of this paper, we will assume that MPEG-21 content can be exported using some mechanism outside the current MPEG-21 standards, and leave the definition of any such mechanism up to the MPEG committee, or for the implementers of particular IPMP tools.

9.2 OMA

The Open Mobile Alliance ("OMA") is an industry body constituted for the development of standards for the mobile phone industry. Amongst a number of other standards for ensuring inter-operability amongst mobile phone services, OMA specifies a DRM regime ("OMA DRM") to be used for the secure exchange and use of copyrighted content on mobile phones.

In the OMA DRM system, a multimedia work such as a ring tone, a Java game or a composition of these is called a *media object* and its use is controlled by a *rights object* (contract), which is a collection of *permissions*, *constraints* and other attributes.

The media objects and rights objects are issued by a *content issuer* and a *rights issuer*, respectively, and these can be used on any OMA DRM-compliant *device*.

For the protection of content, the media object is encrypted and the encryption key is included in the associated rights object. The rights object is encrypted using the public key of the device for which it is intended. To obtain the rights object and thereby the encryption key, a device must undergo mutual authentication with the rights issuer.

OMA rights objects are written in the OMA DRM Rights Expression Language [24], which is a relatively limited langguage based on the full-featured Open Digital Rights Language ("ODRL") [21].

OMA DRM allows rights issuers to issue an explicit 'export' permission that allows media objects and rights objects to be exported from a mobile phone to other digital rights management regimes. There are two modes for export: *copy* and *move*. In the former, the rights object and content remain unchanged on the exporting device after export, but in the latter the rights object becomes permanently unusable on the exporting device.

Our Term	MPEG-21	OMA
Contract	`license`	`rights`
Permission	`grant(group)`	`agreement`
Resource	`resource`	`asset`
Constraint	`condition`	`constraint`
Beneficiary	`principal`	`individual`
Operations	`execute`	`execute`
	`play`	`display`
	`play`	`play`
	`print`	`print`
Constraints	`validityInterval`	`datetime`
	`exerciseLimit`	`count`
	`validityTimePeriodic`	`interval`
	`validityTimeMetered`	`accumulated`

Table 1: Correspondence between MPEG REL and OMA DRM REL elements

The export specification is informative only, and does not specify the exact rules for translating rights objects to other digital rights management regimes; this is left to be defined by the bodies governing the use of the importing digital rights management regimes.

9.3 Contract Translation

ODRL and XrML share their roots in Stefik's rights model, used in one of the earliest rights expression languages, the Digital Rights Property Language ("DPRL") [32] (XrML is in fact the direct descendent of DPRL). ODRL and XrML are thus broadly very similar in structure and semantics, so we might expect translation between the two to be relatively straightforward. Table 1 shows the correspondence between MPEG REL and OMA DRM REL language elements where it exists.

Taking advantage of the fact that ODRL and MPEG REL are both XML-based languages, Polo, et al. [26] propose a simple translator based on XML

Stylesheet Language Transformations [8]. Their translator simply applies a substitution table similar to our Table 1 to transform each ODRL element to an equivalent MPEG REL element. Polo, et al. report this to be sufficient for translating simple contracts.

It is not clear, however, if this approach would work for more complex contracts or what the translator would do if confronted with an element it did not recognise (of which there are many, in the implementation provided by their paper). Furthermore, there are some elements, such as `play` in MPEG REL, which may correspond to more than one element of the other rights expression language depending on context, and so cannot be correctly translated by simple string substitution.

In the following, we will consider more sophisticated translators, paying particular attention to expressions that cannot be directly translated from MPEG-21 to OMA, or vice versa.

9.3.1 Translating from MPEG-21 to OMA

Since OMA DRM REL is much simpler than MPEG REL, there are many expressions in MPEG REL that cannot be translated into OMA DRM REL. We can see no way of expressing the MPEG REL `loan`, `extract`, `copy`, etc. rights in OMA DRM REL, for example. Since OMA devices are presumably incapable of performing these operations, however, it makes sense to simply delete them from an exported contract.

For conditions, things are more complicated. The `trackReport` condition of MPEG REL, for example, cannot be expressed in OMA DRM REL and its requirements presumably cannot be satisfied by an OMA device; therefore any rights dependent on the `trackReport` constraint must be deleted from the exported contract. The `paymentFlat` condition, however, could potentially be satisfied by charging the fee for creating the exported contract.

Determining the best course of action for each untranslatable constraint is a significant task in its own right, and may vary from implementation to implementation depending on what mechanisms are available in each implementation. For example, some systems may be able to store untranslatable constraints for future reference but others may not. We will leave a complete definition of actions to take for untranslatable conditions to the designers of specific implementations.

9.3.2 Translating from OMA to MPEG-21

There are only three elements of OMA DRM REL – the `timed-count` constraint and the `export` permission and associated `system` constraint – that do not appear in Table 1. We have already noted that export in MPEG-21 requires some mechanism beyond that specified in the current standard, and we cannot say how an OMA `export` element can be rendered in MPEG-21 except to say we are assuming that it can be for the purposes of this paper. The `system` constraint, which specifies a DRM regime to which content can be exported, only occurs in the context of an `export` permission.

The `timed-count` constraint allows a permission to be associated with a counter and duration, with the counter being decremented every time the permission is exercised for a period greater than the duration specified by the constraint. A crude translation into MPEG REL would be to replace a `timed-count` constraint with an `exerciseLimit` condition, which decrements its counter every time the permission is exercised at all, regardless the duration of the exercise. This translation loses a significant amount of meaning, however, and considerably reduces the value of the content on the MPEG-21 terminal as compared to the OMA one.

A more sophisticated translation might replace a `timed-count` constraint with a combination of a grant containing a `validityIntervalFloating` condition and another grant containing an `exercise-Limit` condition. The former condition is equivalent to the timing component of `timed-count`, and the former grant can be used to provide free access to the permission for any period less than that specified in the OMA constraint. If the user wishes to use the permission for a longer period, he or she must exercise the latter grant, thereby reducing the counter by one.

9.4 Identifier Translation

9.4.1 User Identifiers

MPEG-21 provides the concrete keyHolder element as one possible instantiation of the abstract principal element. A keyHolder simply contains the public key of a user for whom the associated grant is intended, specified using the XML Signature standard [10]. That is, the grant is awarded to whoever can demonstrate knowledge of the private key corresponding to the public key given in the contract.

OMA does not directly award permissions to users, but has an individual constraint that restricts the use of an associated permission to a particular person. A permission without an individual constraint can be exercised by anyone in possession of the rights object. In MPEG-21, the same effect can be achieved by instantiating the principal element with an empty allPrincipals element, which is defined to indicate that the grant can be exercised by anyone.

Where an individual constraint exists, OMA can identify a specific beneficiary in one of two ways:

- by using the International Mobile Subscriber Identity ("IMSI") stored on the Subscriber Identity Module ("SIM") card of the user's mobile phone; or
- by using a Wireless Identity Module ("WIM") [22], which is a tamper-resistant device containing a private key and may be implemented by the mobile phone's SIM card or another device.

In order to convert an MPEG-21 keyHolder to an OMA individual, and vice versa, we must find a mapping between users' public keys and their SIM cards. The method of doing this would presumably be set out in an import/export agreement between MPEG-21 and OMA vendors.

Since the principal element in MPEG REL is an abstract type, and need not be a keyHolder element, an alternative approach would be to define a new subclass of principal referring to an IMSI or WIM. Of course such a principal would only be useful to terminals that understood IMSIs or WIMs, and a method of converting public keys to IMSIs or WIMs may still be required for translation of content in the MPEG-21 to OMA direction.

9.4.2 Resource Identifiers

MPEG REL provides the diReference and diItemReference concrete types as possible instantiations of the abstract resource element. Both of these elements identify a resource using a *digital item identifier* [15] that uniquely identifies the resource in one a variety of ways. In OMA, an asset is identified by a *uniform resource identifier* ("URI") [2] specifying the location of the file containing the protected resource.

Fortunately for our application, the definition of digital item identifiers includes uniform resource identifiers, so it is trivial to translate OMA asset identifiers into diReference elements in MPEG REL.

Where an MPEG-21 resource is specified using a digital item identifier that is not a uniform resource identifier, it is easy to construct a suitable uniform resource identifier using information from the MPEG-21 digital item declaration from which the translation is being made. For each atomic multimedia object contained in the digital item, the declaration has a resource element (not the same as the REL's resource element) specifying the uniform resource identifier of that resource. Translation is then simply a matter of copying the relevent uniform resource identifier from the the digital item declaration to the OMA asset identifier.

9.4.3 Device Identifiers

The MPEG REL helper, audioOutputPrincipal and videoOutputPrincipal conditions contain a principal element that specifies the identity of a helper application or rendering device to which the resource is permitted to be transmitted. The semantics of the principal are the same here as when it used to identify a user to whom a grant is given; for example, the helper or renderer can be identified as a keyHolder.

OMA DRM REL does not, in itself, support constraints that bind permissions to a particular device or piece of software. However, each OMA device has a public/private key pair and the OMA rights object is encrypted using the public key of the device for which the content is intended. Only that device can decrypt the rights object and thereore the media object, creating an implicit constraint equivalent to a pair of `audioOutputPrincipal` and `videoOutputPrincipal` constraints in MPEG-21.

Thus, MPEG-21 `audioOutputPrincipal` and `videoOutputPrincipal` constraints (if they are specified as `keyHolder`s) can be implemented by encrypting the exported rights object using the public key specified in the constraints.

Granting the OMA `export` permission presumably implies that content may be used on a device other than the one to which the original rights object was issued. In the absence of any way of specifying particular devices to which export is permitted, exported content is presumably accessible to any device supporting the importing regime, that is, there will no constraints on output device in the exported contract.

9.5 Summary

The configuration-driven inter-operability approach taken by MPEG-21 means that translation between MPEG-21 and other systems is not always meaningful. However, comparing MPEG-21 and OMA provides an example of the kinds of challenges that face the designers of import/export regimes.

We have seen that permissions and constraints in OMA can be mapped without significant loss into MPEG-21 REL, but the reverse is far from true. For MPEG REL expressions that cannot be mapped in OMA DRM REL, we need to define methods of pre-enforcement where possible, and deletion otherwise.

The abstract types provided by MPEG-21 allow place-holders to be defined for identifiers not understood by a host terminal. Of course such identifiers are not very useful to that terminal. However, this is a useful feature if the terminal is only a holding space for other devices that do understand the identifier, as in the example we gave in the introduction to this section.

For terminals that are to be used to access the content, however, a mapping between identifiers from the two regimes needs to be defined. We have seen that translation of resource identifiers is trivial, and that translation of device identifiers is either straightforward or unnecessary. However – assuming the `keyHolder` element is used to represent MPEG-21 principals – mappings need to be established between users' IMSIs/WIMs and their public keys. This assumes, of course, that the users involved actually have public keys. If not, a new concrete `principal` type would need to be defined that provided an alternative method of user identification.

10 Conclusion and Future Work

We have given an overview of translation of protected content between differing digital rights management regimes, and identified a series of issues to be addressed by designers of digital rights management import/export regimes. Specifically we have considered

- architectures for translation systems;
- translation of content from one regime to another;
- translation of documents written in one rights expression language to another;
- translation of identifiers from one regime to another; and
- establishing trust between two multimedia devices from different digital rights management regimes.

We further gave an overview of methods by which translation functions might be implemented.

Finally, we applied our observations to the specific example of translation of content between MPEG-21 and OMA regimes. We identified points where translation could be accomplished under the existing standards, and also points requiring further specification to enable full translation.

This paper has focused on the technical aspects of import and export in digital rights management. We are further aware of legal and economic issues that may need to be resolved in the development of import/export regimes:

- DRM vendors – hoping they will achieve market dominance by becoming the *de facto* standard for rights-managed content – may not want to enter into import/export agreements, since such an agreement implies that users may leave the exporting regime's vendor for the importing regime's vendor (who is presumably a competitor of the exporting vendor).
- In the event that a DRM system is compromised and protected content is made freely available, the affected content owners may seek to hold vendors liable for any resultant losses. Where export is possible, the vendor of the compromised system may not be the one with whom content owners have a contractual or other relationship detailing liability.

We will leave these non-technical issues as future work.

We also note that, between the initial submission of this paper and its publication, RealNetworks announced the availability of its "Harmony" technology, claiming that this technology enables "consumers to buy digital music that plays on all popular devices" [27]. Most notably, Harmony is reported to be able export protected music in Real's RealAudio DRM format to Apple iPods, which use Apple's own FairPlay DRM system. As neither Real's Rhapsody service or Apple's iTunes service are available to Australian users, however, the present authors have not had the opportunity to examine either Real's or Apple's technology first-hand. We will therefore leave a detailed examination and evaluation of Real's approach as future work.

11 Acknowledgements

This work was partially funded by the Co-operative Research Centre for Smart Internet Technology, Australia.

References

[1] R. Anderson. 'Trusted computing' and competition policy – issues for computing professionals. *Upgrade*, IV:35–41, 2003.

[2] T. Berners-Lee, R. Fielding, U. C. Irvine, and L. Masinter. Uniform resource identifiers (URI): Generic syntax. RFC 2396, 1998.

[3] J. Bormans and K. Hill. MPEG-21 overview v.5. http://www.chiariglione.org/mpeg/standards/mpeg-21/ mpeg-21.htm, 2002.

[4] J. M. Boyer. Bulletproof business process automation: Securing XML forms with document subset signatures. In *ACM Workshop on XML Security*, pages 104–111, Fairfax, USA, 2003.

[5] W. B. Bradley and D. P. Maher. The NEMO P2P service orchestration framework. In *Thirty-seventh Hawaii International Conference on System Sciences*, pages 290–299, 2004.

[6] C. N. Chong, R. Corin, S. Etalle, P. H. Hartel, W. Jonker, and Y. W. Law. LicenseScript: A novel digital rights language and its semantics. In *Third International Conference on the Web Delivery of Music*, pages 122–129, Los Alamitos, USA, 2003.

[7] C. N. Chong, S. Etalle, and P. H. Hartel. Comparing logic-based and XML-based rights expressions lanuages. In *On the Move to Meaningful Internet Systems*, pages 779–792, Catania, Italy, 2003.

[8] J. Clark. XSL transformations (XSLT) version 1.0. http://www.w3.org/TR/xslt, 1999.

[9] ContentGuard. Extensible rights markup language. http://www.xrml.org, 2004.

[10] D. Eastlake, J. Reagle, and D. Solo. XML-signature syntax and processing, 2002.

[11] S. Guth, G. Neumann, and M. Strembeck. Experiences with enforcement of access rights extracted from ODRL-based digital contracts. In *ACM Digital Rights Management*, pages 90–102, Washington, DC, USA, 2003.

[12] S. Guth, G. Neumann, and M. Strembeck. Toward a conceptual framework for digital contract composition and fulfillment. In *International Workshop for Technology, Economy, Social and Legal Aspects of Virtual Goods*, Ilmenau, Germany, 2003.

[13] International Standards Organisation. Information technology – multimedia framework (MPEG-21) – part 1: Vision, technologies and strategy. ISO/IEC 21000-1:2001.

[14] International Standards Organisation. Information technology – multimedia framework (MPEG-21) – part 2: Digital item declaration. ISO/IEC 21000-2:2003.

[15] International Standards Organisation. Information technology – multimedia framework (MPEG-21) – part 3: Digital item identification. ISO/IEC 21000-3:2003.

[16] International Standards Organisation. Information technology – multimedia framework (MPEG-21) – part 5: Rights expression language. ISO/IEC 21000-5:2004.

[17] International Telecommunications Union. Information technology – open systems interconnection – the directory: Public-key and attribute certificate frameworks. Recommendation X.509, 2000.

[18] R. H. Koenen, J. Lacy, M. Mackay, and S. Mitchell. The long march to interoperable digital rights management. *Proceedings of the IEEE*, 92:883–897, 2004.

[19] M. Mambo, K. Usuda, and E. Okamoto. Proxy signatures for delegating signing operation. In *Third ACM Conference on Computer and Communications Security*, pages 48–57, New Delhi, India, 1996.

[20] T. S. Messerges and E. A. Dabbish. Digital rights management in a 3G mobile phone and beyond. In *ACM Digital Rights Management*, pages 27–38, Washington, DC, USA, 2003.

[21] Open Digital Rights Language Initiative. The Open Digital Rights Language Initiative. http://www.odrl.net, 2004.

[22] Open Mobile Alliance. OMA wireless identity module version 1.1, 24th October 2002.

[23] Open Mobile Alliance. OMA DRM specification version 2.0, 28th February 2004.

[24] Open Mobile Alliance. OMA DRM rights expression language version 2.0, 8th April 2004.

[25] F. Pestoni, J. B. Lotspiech, and S. Nusser. xCP: Peer-to-peer content protection. *IEEE Signal Processing Magazine*, 21:71–81, 2004.

[26] J. Polo, J. Prados, and J. Delgado. Interoperability between ODRL and MPEG-21 REL. In *First International ODRL Workshop*, Vienna, Austria, 2004.

[27] RealNetworks, Inc. RealNetworks introduces Harmony, enabling consumers to buy digital music that plays on all popular devices. Press release, RealNetworks, 26 July 2004. http://www.realnetworks.com/company/press/releases/2004/ harmony.html.

[28] N. Rump. Can digital rights management be standardized? *IEEE Signal Processing Magazine*, 21(2):63–70, 2004.

[29] A. U. Schmidt, O. Tafreschi, and R. Wolf. Interoperability challenges for DRM systems. In *International Workshop for Technology, Economy, Social and Legal Aspects of Virtual Goods*, Ilmenau, Germany, 2004.

[30] Trusted Computing Group. Trusted Computing Group, 2004.

[31] L. van Grinsven and B. Warner. Music downloaders hit by acronym cacophony. *Reuters*, 5 July 2004.

[32] Xerox Corporation. The Digital Rights Property Language: Manual and tutorial – XML edition. http://www.oasis-open.org/cover/DPRLmanual-XML2.html, 1998.