Digital Rights Management - A Survey of Existing Technologies

Sam Michiels *Wouter Joosen* *Eddy Truyen*
Kristof Verslype

Report CW 428, Nov. 2005

Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Digital Rights Management - A Survey of Existing Technologies

Sam Michiels *Wouter Joosen* *Eddy Truyen*
Kristof Verslype

Report CW 428, Nov. 2005

Department of Computer Science, K.U.Leuven

Abstract

Systems that provide Digital Rights Management (DRM) are highly complex and extensive: DRM technologies must support a diversity of devices, users, platforms, and media, and a wide variety of system requirements concerning security, flexibility, and manageability. This complexity and extensiveness poses major challenges to DRM development due to fragmentation of individual solutions, limited reuse and interoperability between DRM systems, and lack of a domain-specific structure that supports and guides the design and implementation of DRM systems and their applications. In order to handle these challenges in an effective and efficient way, we need to understand the DRM context and the internal structure of current DRM solutions. This report presents (1) an easy to understand introduction to DRM by proposing a high level architectural view, (2) an overview of the most important DRM technologies, both proprietary point solutions and open standards, and their mapping onto the architectural view, (3) an overview of the most important rights expression languages, and (4) a discussion on the evolution of DRM that can be expected in the near future. Identifying key DRM services and rights enforcement technologies, and locating them in an overall architecture brings us one step closer to a software architecture for DRM.

Contents

Digital Rights Management - A Survey of Existing Technologies

Sam Michiels, Wouter Joosen, Eddy Truyen, Kristof Verslype
DistriNet, Dept. of Computer Science, K.U.Leuven
Celestijnenlaan 200A, B-3001 Leuven, Belgium
{sam.michiels}@cs.kuleuven.be

September 2005

Abstract

Systems that provide Digital Rights Management (DRM) are highly complex and extensive: DRM technologies must support a diversity of devices, users, platforms, and media, and a wide variety of system requirements concerning security, flexibility, and manageability. This complexity and extensiveness poses major challenges to DRM development due to fragmentation of individual solutions, limited reuse and interoperability between DRM systems, and lack of a domain-specific structure that supports and guides the design and implementation of DRM systems and their applications. In order to handle these challenges in an effective and efficient way, we need to understand the DRM context and the internal structure of current DRM solutions. This report presents (1) an easy to understand introduction to DRM by proposing a high level architectural view, (2) an overview of the most important DRM technologies, both proprietary point solutions and open standards, and their mapping onto the architectural view, (3) an overview of the most important rights expression languages, and (4) a discussion on the evolution of DRM that can be expected in the near future. Identifying key DRM services and rights enforcement technologies, and locating them in an overall architecture brings us one step closer to a software architecture for DRM.

1 Introduction

The domain of digital rights management (DRM) is currently lacking a generic architecture that supports interoperability and reuse of specific DRM technologies. This lack of architectural support is a serious drawback in light of the rapid evolution of a complex domain like DRM. It is highly unlikely that a single DRM technology or standard will be able to support the diversity of devices, users, platforms, and media, or the wide variety of system requirements concerning security, flexibility, and efficiency.

This report is based on [69] and adds to it a study on DRM standards and Rights Expression Languages (RELs). The first objective of this report is to provide an easy to understand introduction to the global working principles of state-of-the-art DRM systems. We want to present a high-level architectural view on DRM that highlights the main building blocks a DRM system is composed of. Secondly, we want to offer the reader an overview of the most important technologies currently available and map the generic DRM architecture onto each particular technology. On the one hand, we enumerate the most relevant point solutions for DRM that we are currently aware of; these solutions are typically proprietary, although some of them integrate open standards into their solution. On the other hand, we offer an overview of the most important DRM standards and RELs. The third objective is to make clear how far DRM is developed today and what evolution we can expect in the near future.

The report analyses state-of-the-art DRM technologies and extracts from them high level usage scenarios according to content consumers, producers, and publishers. In addition, the key services

are identified both from a functional and security perspective. Identifying key DRM services and locating them in an overall structure brings us one step closer to a software architecture for DRM. Having available a software architecture should help the DRM community in reasoning about DRM systems, and in achieving reuse and interoperability between multiple domain-specific DRM technologies and standards.

The remainder of this report is structured as follows. This introduction sketches the context of DRM by defining key terminology and introducing the basic concepts. It also summarizes the main requirements from three perspectives: security, flexibility, and efficiency. Section 2 presents a generic DRM architecture from three application viewpoints: the content consumer, the content producer, and the content publisher. This architecture allows for high-level reasoning about DRM solutions, independent from a specific implementation. Section 3 presents an overview of state-of-the-art DRM technologies and maps each of them onto the generic architecture presented in Section 2. Section 4 presents open DRM standards and rights expression languages. Section 5 concludes by summarizing the highlights in this report.

1.1 An evolving technology with evolving requirements

For a long time, people who wanted to hear a song had to listen to the radio or needed a physical medium such as a gramophone record. Only when having the original medium, one was able to consume it (i.e. listen to a song, read a book, view a video or image). In the eighties, copying became not only possible but also affordable. Yet, copying data always implied a significant decrease in quality. The quality of a movie recorded from television on VHS tape, for instance, is clearly distinguishable from the original transmission. Further copying the VHS tape leads to further degradation of video quality.

During the last decade, one could notice an enormous increase in capabilities of both data copying and data storage. A copy can now retain almost perfectly the original quality, and it can be stored easily. Moreover, copying and storage technologies have become affordable. The popularity of the Internet and its increasing bandwidth and transfer speeds has enabled the distribution of popular (multimedia) content around the globe, just by performing a few mouse clicks.

The ease of copying and sharing digital content without a significant decrease in quality has resulted in a huge amount of piracy. Obviously, content owners and distributors are searching for ways to stop illegal data copying and distribution. Clearly, some law suits to deter most people did not really work. A more effective way to tackle the problem is to use technology to make it impossible, or at least very hard, to ignore or circumvent copyright rules. This is where Digital Rights Management (DRM) comes into play. At present, existing DRM technologies are not yet perfect, but they already have become useful in practice.

1.2 Definition of DRM

DRM is a technological way to allow owners of digital content to control access to this content and to restrict its usage in various ways that can be specified by the owner or his delegates. Under digital content we understand audio, video, images, text, other data constructs and any combinations of these in a digitally represented format. Obviously, computer programs are also considered to be digital content.

DRM can thus be defined[1] as the management and enforcement of usage rules, including copyright rules, on digital content. Optionally, it can also be responsible for all the accounting aspects.

The purpose of DRM can further be summarized as follows:

- *DRM allows content to be associated with access rights and copyrights.* DRM enables the owner of digital content to associate the notion of access rights to that content, and to enable

[1] We are aware of the existence of other DRM definitions [65], but we use the one that is most common and accepted in the research community.

consumers of the content to acquire the necessary rights that enable them to use the content for their needs.

- *It can enforce that the copyright on digital content is respected.* Copying may only be possible for people who have the necessary rights. This can be based on their role and identity in an organization; or it can be based on the fact that they have paid (or have committed to pay) for the proper copyright. The copyright on digital content isn't respected if the technology is based on the fair-use principle.
- *It allows content owners to add extra (more sophisticated) usage rules.* The usage of DRM opens a whole set of new possibilities, much more than just enforcing the copyright. DRM enables, for example, to restrict the consumption of digital content to a predefined time-space, to limit the number of times digital content can be consumed, or to specify the allowed actions (e.g. view, but not print).
- *It supports management of accounting aspects.* The fee that consumers usually pay for consuming content has to be managed and distributed among various parties. The content owner may want a compensation in proportion to the effective usage of his content. In addition, the content publisher should get a compensation for making content available. Finally, the network provider may have a financial agreement based on the amount of download traffic associated with specific content. The DRM system must support accounting and financial aspects, or at least enable to cooperate with external accounting or payment systems.

DRM will only become an acceptable technology if it takes into account technological, as well as economical, social and legal concerns [50]. From a technological perspective, functional features must be offered in a user-friendly way. Economically, consumers want to obtain digital content in various formats via new content delivery services. This must result in new business models. It is a social challenge to convince consumers that DRM is an appropriate mechanism to enjoy content and, by consequence, provides added-value. And finally, from a legal point of view, the evolution of DRM must involve efforts towards international harmonisation and protection of intellectual property.

1.3 Objectives

The first objective of this report is to provide an easy to understand introduction to the global working principles of state-of-the-art DRM systems. We want to present a high-level architectural view on DRM that highlights the main building blocks a DRM system is composed of. Secondly, we want to offer the reader an overview of the most important technologies currently available and map the generic DRM architecture onto each particular technology. On the one hand we enumerate the most relevant point solutions for DRM that we are currently aware of; these solutions are typically proprietary, although some of them integrate open standards into their solution. The third objective is to make clear how far DRM is developed today and what evolution we can expect in the near future.

1.4 Basic concepts

In order to discuss the high level concepts of state-of-the-art DRM systems in further detail, we briefly sketch the DRM context by introducing the key terminology first. After that, we elaborate on the basic concepts by discussing (1) how content is protected, (2) how rights are expressed, (3) how rights are associated to protected content, (4) how different sets of rights can be specified, (5) how content and rights are distributed and, finally (5) how content can be used.

1.4.1 Context

Content refers to the data to protect. This can be audio, video, images, formatted or unformatted text, software or other data constructs. Different actions on content are possible (e.g. play, print,

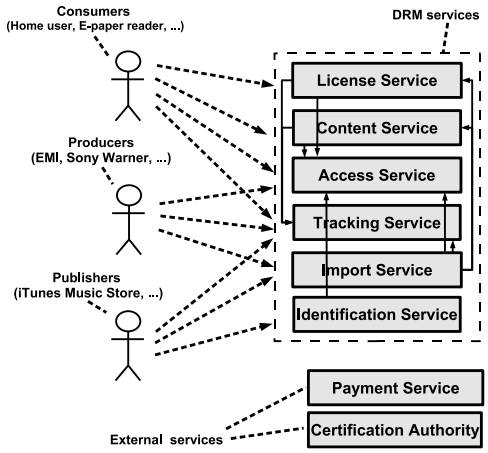

Figure 1: The typical roles involved in DRM are the producer, the consumer and the publisher role. The consumer uses the Consumer Tool to obtain content. The publisher uses the Publisher Tool to publish the content created by the producer.

or save), each of which may be content type dependent. For example, what could we mean by printing a song? Performing an action on content is called content *consumption*.

The main roles (see figure 1) involved in DRM are the producer, the consumer and the publisher role. Each role represents one or more persons, services, or companies that play a coherent role in the DRM processing chain. The *producer* is the entity that produces content, or at least owns the rights to distribute and sell it, and represents the starting point of the DRM chain. This can be, for instance, an author, a musician, a director, or a record label (such as EMI, Sony and AOL Time Warner). At the other end of the DRM chain, the *consumer* is the entity who wants to obtain and consume content. For example, a home user that wants to download and play music. The *publisher* is the entity that owns and manages the DRM system used to distribute content, and forms the bridge between consumers and producers. Most DRM technologies available today focus on the consumer aspect.

The online *DRM system* is the set of DRM related services offered by the publisher to consumers and producers. The *DRM client* is the entity at consumer side that is responsible for performing the DRM-specific operations in a secure way while obeying the right specifications. The *producer tool* enables producers to add content and corresponding contracts to the DRM system. In this way, content becomes publicly available and licenses can be issued. The *publisher tool* for its part allows a publisher to manage and maintain the DRM system.

The DRM client is the only entity at the consumer side that is allowed to unprotect the protected content; it must not expose this content to other hard- or software. Usually, some kind of encryption is used to protect content. This abuse *prevention mechanism* is often combined with a *detection mechanism*, which allows to identify the source of misuse when illegally distributed content is found.

To formalize the rights a consumer may obtain on some content, usage rules can be defined in a *Rights Expression Language (REL)*. A REL defines a language and vocabulary that enable the specification and interpretation of usage rules in an unambiguous way. For example it can be used to formally describe that a person Bob can listen to a specific DRM protected song only five times in the year 2005. The two most successful RELs are ODRL and MPEG REL [60, 56].

Usage rules must be associated with the corresponding piece of DRM protected content. The concept of *licenses* introduces a separation of DRM protected content and the sets of usage rules to be associated with it. Licenses are typically bound to protected content, but may also be associated with one or more specific devices or legitimate consumers. In the latter case, only these specific devices or persons are able to consume the corresponding protected content using that license. Each distinct set of usage rules can define another *license type*. Different license types

may correspond to the same content.

Producers often want to specify the license types themselves, i.e. they want to specify different sets of usage rules describing the rights consumers can obtain. The producer may also want to associate business information with a license type, such as the price and distribution period. All this information is negotiated with the publisher and combined in a *contract* corresponding to the submitted content. This contract contains all information that is needed by the online DRM system to issue different license types.

1.4.2 Expressing rights

In the past, things were not that complicated. Given the medium on which content resided, one could view the video or listen to a song. As sketched above, the concept of DRM opens a whole set of new possibilities, not only in enforcing new rules on how the content can be consumed, but also in the way a consumer can distribute its rights to consume content on other devices or by other persons. Here we will discuss the possibilities conceptually, defining rules that can be combined using first order logic and that can possibly even easily expressed by the content producer.

Defining these rules is done using a Right Expression Language (REL). A REL defines a language and vocabulary that enables the specification and interpretation of usage rules in an unambiguous way. To give an example; it enables to formally describe that a person Bob can listen to a DRM protected song only 5 times in the year 2005. The two most successful, generally applicable, flexible and extensible RELs are ODRL and MPEG-21 REL. Also, some RELs for more specific business domains appeared. One such language is PRISM [35], which is directed to newspaper publishers.

A REL must specify a number of concepts:

- The *content* to which the rights are related. This may also be a subpart of multi-part content.
- The *parties* that obtain the rights: who or what devices may be able to consume the content. All the people in an office, for example, may be able to view a document only on devices set by that company. This makes it impossible to read the document at home or useless to send it to others.
- The *actions* that can be allowed by a party on content. We distinguish between consumption actions and distribution actions. Some of the former are print, play, execute, display, install, delete, verify, restore, uninstall, save, backup, modify, annotate, and aggregate, while some of the latter are sell, lend, give, lease, move and duplicate. Actions are often content type specific
- The *conditions* that must be fulfilled before being able to do one or more associated actions on the content. Conditions can be grouped:
  - *Party bounds* (what subject). Only devices or users with certain properties may be able to do certain actions. A device may need for example a valid certificate that identifies it as a secure device, or it may have to meet certain hardware specifications.
  - *Content bounds* (what object). The associated action is only possible on a part of the content. The content can for example only be consumed if it has a certain quality or format or if it has other properties.
  - *Repeating bounds* (how long, how many). The associated action can only be done a limited number of times (e.g. 10 times) or the accumulated time of the action can be limited (e.g. you can listen to the song max. 60 minutes altogether).
  - *Geographical bounds* (where). It is possible to specify where content can be consumed, for example, only in Belgium.

– *Temporal bounds* (when). It is possible to specify when content can be consumed. This can be a period (e.g. 2 months) that starts either at a predefined time (e.g. 1 January 2006) or at the time the associated action is performed for the first time. When the period is set to unlimited and the start date is fixed, content can become accessible starting from that date. Another option is to define intervals within limited time zones such as a day, a week, a month or a year. For example, usage rules can define that a movie can only be viewed between 8.00 p.m. and 11.00 p.m. each Saturday and Sunday.

Using these concepts, usage rules can be composed, which define rights on the content for the consumer. We distinguish between consumption rules and distribution rules. The former are related to consumption actions, while the latter to distribution actions.

Because the extended number of different usage and types of content, that are often business or content specific, it is impossible to give a full coverage of all conditions, actions, parties and content. General RELs such as ODRL (see Section 4.3) and MPEG-21 REL (see Section 4.1) give a good basis that can be used to build more specific RELs. At the moment, most DRM systems do not use the full expressiveness of current RELs (or do not use it at all). But as time will pass, they will become increasingly flexible in defining rights.

1.4.3 Associating rights to protected content

As we have mentioned before, rights are defined by usage rules. These have to be associated with the digital content. The rights have to be bound to a specific piece of protected content. In the past, attempts were made to embed usage rules in an imperceptible way into the content itself. This approach, followed by SDMI [38] - which used a watermark, failed [57].

A second and inflexible method still used today is defining a fixed, technology dependent set of usage rules. In this case, each piece of content that is protected using a particular DRM technology has associated with it the same set of usage rules. For example, each piece of content protected using FairPlay (see Section 3.2), can be copied no more than four times. Clearly, this approach is too rigid and has serious drawbacks: content producers should, for instance, accept the consumption limitations which the DRM company stipulated, and business models such as pay-per-view are not possible.

The third and most flexible method introduces the concept of licenses. Here, a separation between content and (sets of) usage rules is made: on the one hand we have protected content and on the other hand we have licenses containing usage rules corresponding to that protected content. Different license types can correspond to the same content. Sometimes, one license contains usage rules corresponding to multiple pieces of content. The producer of a sitcom, such as 'Friends', can use for example one license for a complete season.

In the remainder of the text, we will focus on the license scheme, combined with contracts.

1.4.4 Specifying different sets of rights

Different sets of rights corresponding to the same piece of content can be specified, resulting in different license types. Different users can thus have different rights on the same content.

Content owners or their delegates that are able to submit content to a DRM system (i.e. what we call the producers) often want to specify the license types themselves, i.e. they want to specify different sets of usage rules specifying the rights that can be obtained. The producer also wants to set some business information, such as the price and distribution period, for the different license types. All this information is combined in a contract associated with the submitted content. This contract contains all information that is needed by the online DRM system to issue different license types.

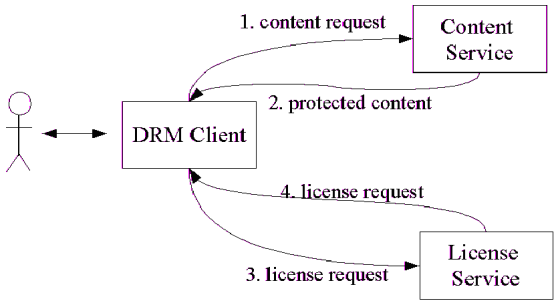

Figure 2: A typical DRM use case: a consumer uses its DRM client to contact an online Content Service and search for content. Only when a license has been acquired via the License Service, content can be consumed.

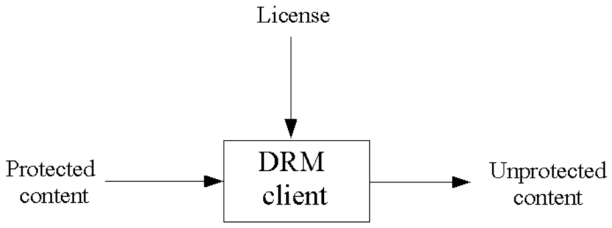

Figure 3: License usage

1.4.5 Distributing content and rights

Before consumers are able to use protected content, they first have to obtain it along with a corresponding license that enables them to use the content according to the usage rules described therein. In a typical, simplified scenario, which is illustrated in Figure 2, a consumer uses its DRM client to contact an online *Content Service* (1). This service enables to look up and download protected content (2). Content can be consumed , after a license has been requested (if it is required by the DRM technology) (3) and acquired (4) via the *License Service* using super-distribution (for example free licenses) or another distribution channel.

Instead of distributing content using online services, another scenario called super-distribution is possible. Super-distribution allows the distribution of the protected content using peer-to-peer networks, portable media, e-mail, FTP, etc. In fact, any way of distributing the protected content is allowed. This fastens the spreading of the protected content and lowers the load on the Content Service drastically. When one obtains the protected content this way, still a corresponding license has to be obtained by contacting the License Service. Not all technologies provide super-distribution. In most cases, one will have to pay before a license can be obtained. Free licenses can also be useful to the content producer. For example, a license with very limited usage rules can be issued for free and allows the consumer to hear the first 60 seconds of the corresponding song. To be able to hear the full song, another license offering a less restrictive set of rights has to be bought.

1.4.6 Using content

Licenses are not only bound to protected content, but also to one or more specific devices or persons that are legitimate consumers. Only these specific devices or persons are able to consume the corresponding protected content using that license. Distributing a license to other, unauthorized parties is thus useless. The DRM Client on the device is responsible for the enforcement of the rights: no other rights than these specified by the usage rules in the license may be possible by the consuming entity. This is illustrated in Figure 3.

1.5 Summary of requirements

This section presents the main requirements concerning three dimensions: security, flexibility, and network efficiency. These requirements will be used to validate state-of-the-art DRM technologies and to check to what extent each of them balances these three dimensions.

1.5.1 Security requirements

1. Core of the solution.

   - *Controlled distribution of rights.* Licenses may only be issued to authorized parties and the distribution of usage rules in a license must be according to the distribution rules in that license.
   - *Controlled consumption of content.* Digital content may only be consumed respecting the obtained rights. It has to be impossible to do more with the digital content than described in the usage rules in the license. If no license is found, the digital content cannot be consumed.
   - *Fraud Detection.* Even if digital content has been extracted out of his DRM context (e.g. by recording the output generated by a sound card), it must still be possible to detect mass abuse of such content. Imagine the DRM protection of a movie had been removed and the new file is spread on a file sharing network, it must still be possible to detect the source of the misuse.

   The first two requirements are prevention requirements, while the last is a detection requirement in case prevention failed or was circumvented.

2. Reliability and robustness of the solution.

   - *Permanent protection.* The content storage and transmission must be protected at all times. Content stored on servers or consumer devices must be protected in such a way it cannot be read by unauthorized parties. Eavesdropper must not be able to consume the content, even if they can observe all interactions between the online DRM System, the consumer and the producer.
   - *Tight coupling between content, license and consumer.*
     - A license, although physically separated from the content, must logically be associated with it. A license may only be used for consuming the corresponding content.
     - It must be possible to verify the validity of both licenses and content.
     - A license can only be used by the legitimate consumer on legitimate devices.
   - *Renewability.* When certain software parts in a DRM client are compromised or no longer considered secure due to new security threats, it must be possible to replace these by better parts. If, for example, the watermarking scheme used in a DRM system is broken, it must be possible to replace it. DRM Clients that are considered insecure (because they are not yet renewed), can be denied to use services offered by the online DRM service.
   - *Revocation.* It must be possible to revoke a device that has been fully compromised. Obtaining new licenses becomes impossible for that device.
   - *Tamper-resistant usage rules.* It must be impossible to change the usage rules described in the license.

1.5.2 Flexibility requirements

1. Overall applicability of a solution

   - *Rich content.* It must be possible to protect any type of content, including rich content. Each sub item in the rich content can have its own rights. A set of images, for example, can be used freely by everyone, while a piece of audio contained in the same content may not.
   - *Expressive usage rules.* It must be possible to define and enforce in a flexible way the usage rules that were described in the previous section, and future extensions of the rule set.
   - *Ubiquitous consumption.* It must be possible for the consumer to consume DRM protected content with the license on any DRM-enabled device.
   - *Application Programming Interface (API) available.* The API must be offered such that developers can integrate DRM functionality in their own applications.
   - *Multi-platform.* The DRM technology must be portable to different platforms.
   - *Producer tool.* A tool must be offered enabling content producers to add their content and corresponding contracts to the DRM System.
   - *Publisher tool.* A tool must be offered enabling the content publisher to maintain the system.
   - *Easy use.* Using a DRM client must be as easy as possible. A difficulty that can occur is the necessity of certificates. Not all consumers have one at the moment.

2. Supporting different exploitation models

   - *Super-distribution.* The load on the content server is heavily decreased while spreading of the content is fastened.
   - *Rights gift.* One should be able to buy rights (e.g. in the form of a license) for someone else.
   - *Recoverability.* After a system crash, consumers should be able to recover all their digital content and their rights to consume and distribute it.
   - *Easy payment.* The consumer has to be able to easily perform payments, enabling him/her to buy various licenses. Different business models are possible, such as pay-per-consume, subscription, pre-pay and pay-per-license basis.
   - *Fair-use.* DRM is potentially to restrictive. The consumers of DRM protected content want to be allowed to do everything they were used to do with content before the DRM era. They still want to be able to do small scale copying to different devices, friends, family, etc. and they want to consume content without restrictions. Of course severe violations of the law must be made impossible by DRM. This principle is called fair-use.

1.5.3 Efficiency requirements

- *Minimal network traffic.* The number of messages between client and server has to be minimized. Also, the size of the message must be minimized.
- *Minimal client resource usage.* Both processor and memory usage by the DRM client to perform its actions need to be minimized. Is special hardware needed at the consumer side?

As this report will show, it is not always possible to fulfill all of these requirements with the current state-of-the-art technologies. Some requirements can be considered as long term goals and some even contradict each other. For example, the super-distribution requirement is not compatible with the fraud-detection mechanism.

2 A generic DRM Architecture

Having sketched the working principles in recent DRM technologies, this section proposes a generic DRM architecture using the state-of-the-art concepts discussed in the previous section.

First, the high level DRM functionality is studied from different viewpoints (Section 2.1), followed by an identification of the main services that should be present in Section 2.2. Section 2.3 integrates security technologies by locating the security hot spots in the architecture and showing how security services can be established in particular hot spots using different cryptographic primitives.

2.1 Application level viewpoints

What is needed at application level to have a complete DRM system? In order to answer this question, three main roles are distinguished: the consumer, the producer and the publisher. In addition, some external roles are briefly discussed. Identifying multiple roles in a DRM system is crucial to view the complete picture.

Consumer. Consumers want to consume protected content in a user-friendly way. They want to be able to browse the content catalog of the online DRM system where the content at stake can be obtained. Since consumers also need a license, they must be able to select a license type and view the usage rules , in a human readable format, associated with it. Generally, consumers first have to pay, one way or another; different business models should be possible (e.g. subscription, pay-per-license, or pay-per-use). When time-based licenses expire, it must be possible to update them, which may also require some financial transaction. Consumers also want to browse their obtained licenses locally and view the usage rules in a human readable format. Finally, consumers want to consume the protected content, according to the usage rules associated with the corresponding license. In order to fetch licenses (and sometimes also protected content), consumers need to authenticate to the online DRM system.

Producer. Producers want to easily compose a contract. Both content and contract must be submitted to the online DRM system. After some time, they may want to update the contract or maybe even cancel it, i.e. stop the distribution of the content. Content producers expect a financial compensation from the DRM service for the trade of their content. Therefore, they want to receive statistical information from the DRM service about the number of downloads or content usage patterns. In order to query or submit content to the online DRM system, content producers need to authenticate themselves.

Publisher. When one or more DRM clients are no longer secure, their right to consume content must be revoked. It may also be necessary to update some parts of the DRM system (and the DRM client). Content publishers may want an overview of system usage patterns. When content is found mass-distributed, the source of abuse must be identifiable. Publishers need to authenticate to the DRM system first.

External roles. Some other roles include the financial institution that offers support for billing issues, and the owner of the network that is used to distribute content and licenses. When certificates are used, an external Certification Authority (CA) is needed to obtain certificates and to check the validity of certificates.

2.2 Core components

The previous section presented the main requirements from multiple application-level viewpoints. We are now able to discuss the different high level services and interactions that are needed in a complete DRM system to provide this application-level functionality. We first identify services needed from the consumer's viewpoint, and subsequently add the services needed from the producer's and the publisher's viewpoint. The last part of this section defines the interfaces and ports for each component in our general architecture.

We will put these entities together in a general high-level architecture using UML. We split the architecture in three views. The first view, shown in Figure 4, illustrates the interactions between

the online DRM system and the consumer. The second view, shown in Figure 5, focuses on the interactions between the online DRM system and the producer. The third view, shown in Figure 6, illustrates the interactions between the online DRM system and the publisher.

We first briefly explain some UML notation [41] we use in the figures to come. First of all, there are number of rectangles, called components. A component is a logical part of an application, consisting of several classes. A component only interacts with other components using predefined interfaces (synchronous) or ports (asynchronous). An interface is defined by a set of methods and is indicated graphically by a circle, connected with his component using a simple line. If a component can communicate with an interface of another component, an arrowed line is drawn from the former component to the specific interface of the latter component. A port only accepts or sends a certain type of messages; we only have to define these messages. A port is indicated by a small rectangle at the border of a component. Both sender and receiver of the asynchronous communication have a port. These ports are connected using an arrowed line, indicating the direction of the communication. A component can be distributed, but abstraction is made and they are seen as logical units. An extensive description can be found in [58].

2.2.1 Content consumer

We identify four key DRM services with respect to the content consumer: the Content Service, the License Service, the Access Service, and the Tracking Service. In addition, we describe two external services for payment and certification. See also Figure 4.

Content Service. When consumers want to obtain protected content, they use their DRM Client to contact the Content Service where they can search for content. Before a consumer obtains any content, the Content Service protects it. Since this protected content may be personalized, the Content Service needs certainty about the identity of the consumer. This is obtained by interaction (i.e. running a protocol) between the Content Service, the DRM Client and the Access Service. This identification phase, however, is not always needed. Dropping it disables the detection mechanism in case of mass-abuse but allows for super-distribution. The Content Service may need to send protection data, specific to that piece of content, to the License Service, to allow this service to issue associated licenses. The Content Service receives its content and potentially some additional data from the Import Service, which is discussed in the producer part.

License Service. Once the License Service has received the protection data from the Content Service, it is able to issue corresponding licenses upon consumer request. The License Service may offer different license types corresponding to the same content. The consumer requires a description of the different types and selects one. The consumer's DRM client provides some system or user specific parameters to generate a license that is only usable by that consumer. Before the license is issued, the consumer usually has to pay for it. Therefore, the License Service asks the Access Service if it may issue the license. The Access Service in turn contacts the external Payment Service and will only answer positively if the consumer has effectively paid. Expired licenses can be updated by sending them to the License Service, which will issue new ones to the consumer, often after a new payment. The License Service may also send statistical information about the license issuance to the Tracking Service. The Content Service receives its contracts and potentially some additional data from the Import Service, which is explained in the content producer part (see section 2.2.2).

Access Service. The Access Service is responsible for the authentication of the content producers, consumers (or its DRM clients) and publisher. It is also responsible for checking payments of both consumers and producers before allowing particular actions on the DRM system. The Access Service may, for example, deny access if some bills are not paid or if consumers fail to identify themselves.

Before content consumers are able to obtain licenses or even content, some registration procedure may be necessary beforehand. This procedure usually involves the Access Service and, if some financial transaction is needed, the External Payment Service.

Tracking Service. The DRM Client, the License Service and the Content Service can generate statistical usage information (e.g. the number of times a piece of content is downloaded, the type

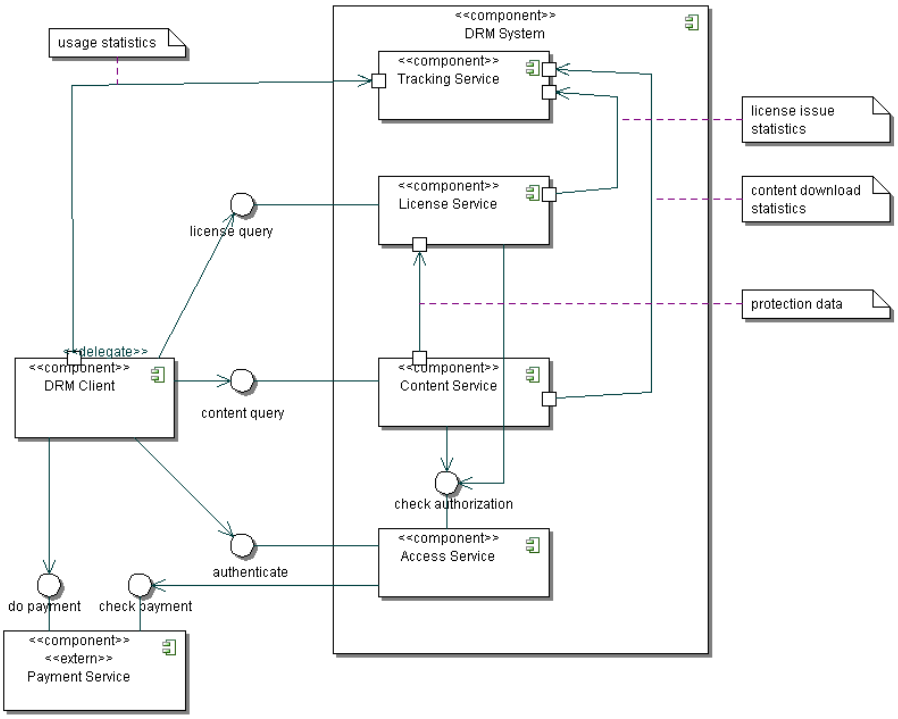

Figure 4: Content Consumer view on the generic architecture

of licenses issued for it, or the number of times it is consumed). These components (or a subset) can forward such information to the Tracking Service, which in turn can produce usage statistics.

Payment Service (external). The financial aspects are taken care of by an external Payment Service, typically a bank or another financial institution. Obviously, some interaction is needed between the external Payment Service, the Access Service and the DRM client, producer tool or consumer tool.

Certification Authority (external) is only necessary if certificates are used. The CA is responsible for the issuance, distribution and revocation of certificates.

2.2.2 Content producer

We describe three main DRM services offered to content producers: the Import Service, the Tracking Service and the Compensation Service. See also Figure 5.

Import Service. Before producers can put content into the DRM system by contacting the Import Service, they must identify themselves via the Access Service. This is necessary to prevent misuse of the DRM system: only the owners of the rights on the content may submit that content to the DRM system. After identification, the producer can submit the content to the Import Service, including meta-data and a corresponding contract. Upon receipt, the Import Service may first convert and manipulate the content such that the DRM system is able to deal with it. Subsequently, the content is sent to the Content Service, while the contract and additional information (such as the content identifier) are send to the License Service. In a similar way, content can be updated or removed from the DRM system.

Tracking Service. Producers can also ask the Tracking Service for usage statistics corresponding to their content and can check, by contacting the external Payment Service, if they are compensated accordingly.

Compensation Service. At regular times, on a monthly basis for example, the content producer gets a compensation for the trading of his content by the online DRM System. Therefore, the Compensation Service first asks the Tracking Service the necessary statistical information and

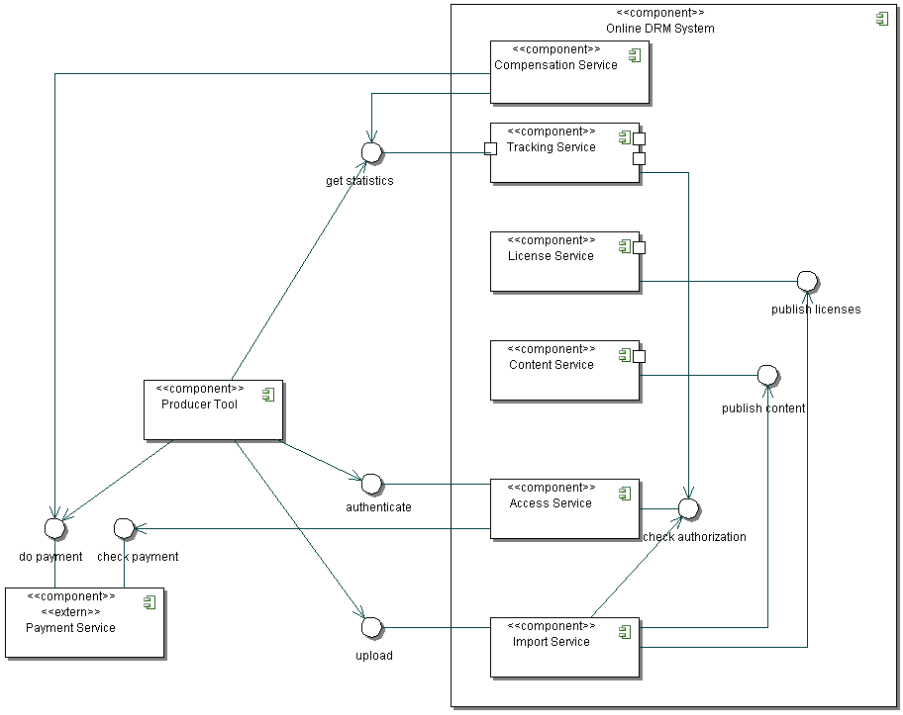

Figure 5: Content Producer view on the generic architecture

then contacts the external Payment Service to pay the content producer. The Compensation Service can also compensate other parties, such as the owner of the physical network.

2.2.3 Content publisher

The content publisher uses three major DRM services: the Access Service, the Identification Service, and the Tracking Service. See also Figure 6.

Access Service. The publisher first of all authenticates to the DRM System using the Access Service. Secondly, the Access Service allows the publisher to revoke a single DRM client or a whole class of clients, for example all DRM clients that have not yet been renewed after a security threat.

Identification Service. The publisher can send content that is found on mass-distribution networks to the Identification Service, which will reveal the identity of the source of abuse.

Tracking Service. Similar to the other roles, the content publisher is also able to obtain statistical information about the use of the DRM System.

2.2.4 Component interfaces

Now we have described the DRM system from content consumer, content producer and content publisher point of view, we list the different component interfaces.

License Service. The consumer contacts the license interface to get a list of the different license types (*getLicenseTypes(contentId:String):List*), wants to obtain a license (*getLicense(contentId:String, licenseTypeId:String, params:Params):License*) and wants, after some time, to update his expired licenses (*updateLicense(oldLicense:License):License*).

The Import Service can add a new contract (*addContract(contract:Contract)*), can optionally submit some key data resulting from the pre-packaging (*addKeyData(contentId:String,*

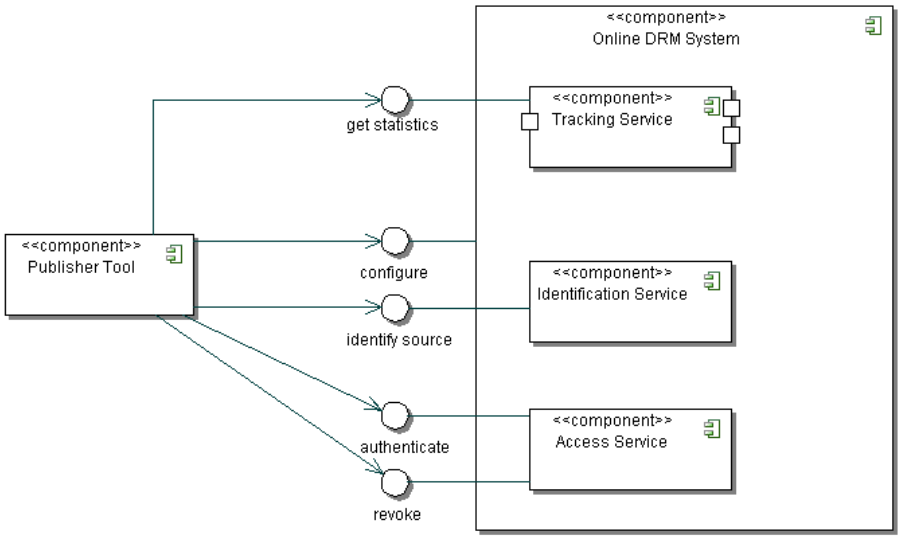

Figure 6: Content publisher view on the generic architecture

keyData:KeyData):boolean), can update a contract (*updateContract(oldContractId, newContract):boolean*) and stop the License Service from spreading licenses corresponding to a certain piece of content (*remove(contentId:String):boolean*).

Content Service. The content consumer can search for specific content (*search(query:String):List*) and can, potentially after a necessary identification, obtain some specific content (*getContent(contentId:String):Content*).

The Import Service can add new content (*addContent(content:Content):boolean*) and stop content from being provided by the DRM system (*removeContent (contentId:String):boolean*).

Tracking Service. The Tracking Service offers a public interface offering statistical information (*getStatisticalInfo(params:Params):StatisticalInfo*). Identification is needed beforehand. The tracking Service also receives messages containing statistical information gathered by the License Service, the Content Service or the DRM Client of the consumer.

Import Service. The Import Service offers the content producer an interface to submit new content to the DRM system (*addContent(content:Content, contract:Contract):boolean*), to update an existing contract (*updateContract(contentId:String, contract:Contract):boolean*) and to stop the distribution of content (*stopDistribution(contentId:String):boolean*). Identification is needed beforehand.

Compensation Service. The compensation Service has no explicit interface, but is triggered. A simple interface can be provided to the content publisher to check if compensations are done accordingly and to do some manually.

Access Service. The Access Service has to offer a way to the Content Producer, the content consumer (and/or their DRM clients) and the content publisher to prove their identity and right to do some actions (authenticate(...)). This can be done in a number of ways (certificate based, subscription based, ...). The DRM System wants to know if a consumer or consumer is allowed to do certain actions (*isAuthorized(user:User, action:Action, parmas:Params):boolean*). The content publisher can revoke a single DRM client (*revokeClient(clientId, String):boolean*) or a class of DRM clients (*revokeClientClass(params:Params)*).

External Payment Service. The external Payment Service is not a part of our general architecture. The methods we need are rather basic and will typically be present: we need a method to transfer money from one account to another (doPayment(money:Amount, Account:String):boolean) and one to check if a bill is already paid (checkPayment(bill:String)).

Identification Service. This service offers an interface to extract the identity from mass distributed content (revealIdentity(content:Content):Identity)

2.3 Integrating security technologies

While the previous section focused on the main functional components of a generic DRM architecture, this section switches attention to security. It first identifies hot spots in the architecture that highlight where to inject specific security services to establish a secure DRM system. Secondly, it provides an overview of the main cryptographic primitives and means that are required to implement the security services.

2.3.1 Locating security hot spots

Where in the proposed architecture should one inject specific security services to establish a secure DRM system? The major security services that are identified are unforgeability, integrity, authentication, confidentiality, non-repudiation, and anonymity. We discuss what services are needed to secure licenses, protected content, contracts, newly submitted content, the online DRM system and the consumer system.

Licenses. A secure license must first of all be *unforgeable*, meaning that it should be impossible to parties other than the License Server to construct a valid license. A second important property to be guaranteed is *integrity*, i.e. any change in a license will be detected and thus becomes invalid. Thirdly, a license must be *uniquely bound to* its corresponding content and to the owner of the license. Consumers authenticate themselves to a device which contacts the License Service on behalf of the consumer. This results in a person-bound license. It is still difficult in practice to associate with every person a certificate, and thus to bind a license to a physical person. Binding to one or more devices owned by that person is often applied instead. Initiatives, such as e-ID[14], may solve this problem.

Content When a content producer submits new content, a third party may not tamper with (*integrity*) or have access to (*confidentiality*) it. *Authentication* must be possible to determine who is responsible for the content submission to the DRM system. The protected content spread by the Content Service should only be consumable by owners of a legal corresponding license; to all others, *confidentiality* of the content is needed. Practice has shown that it is very hard to prevent consumers from getting hold of the content in an unprotected way. For this reason, the Content Service may also need to apply a detection mechanism on the unprotected content to support non-repudiation when mass-abuse is detected.

Contracts. If a person illegally submitted content to the DRM system, one must be able to prove that this person committed a violation. This requires *authentication* and *non-repudiation*, such that the suspect cannot deny the facts. *Integrity* also protects against tampering by a third party.

Consumer system. At the system of the consumer, only entities (such as the operating system, a sound card, or a DRM client) that can *authenticate* themselves as trustworthy may become authorized to get hold of the content in an unprotected way. In this way, *confidentiality* of the content is maintained. *Integrity* of authorized software entities should be maintained; if not, one might as well insert a Trojan horse that sends unprotected content to a file or another party. Because the rights defined in licenses are often time related, there is a need for *secure time*: it should be impossible to use an expired license by simply changing the system time. Another important property for DRM clients is *individualization*, which reduces the danger of global breaks if one specific DRM client is compromised.

Online DRM system. In the DRM system, sensitive data (unprotected content, key data) are stored and sent from one component to another. For example, the License Service holds secret data that are needed for creating valid licenses. Thus, *confidentiality* is needed and *integrity* must be kept for both the sending and storing of sensitive data. When sensitive data are sent from one component to another, mutual *authentication* is needed. When usage information is sent to the Usage Tracking Server, the *anonymity* of the sender must be respected. No third party may change the statistical information. This requires again *integrity* and *authentication* of both the sending and the receiving component. *No replay-attacks* are allowed to prevent usage information being sent more than once.

2.3.2 A reference to basic cryptographic primitives

Although both hardware and software mechanisms exist to build secure DRM systems, we focus specifically on software solutions that can be integrated in the generic DRM architecture. We investigate what cryptographic primitives can be used to establish the cryptographic building blocks presented in the previous section. We discuss the main cryptographic primitives, such as encryption, digital signatures, certificates, watermarking and secure clocks. At the end, we introduce some key paradigms and techniques to obtain integrity of the DRM client and confidentiality of protected content.

Encryption. One of the most accepted and most used cryptographic primitives, which converts readable data into cipher text [73]. Converting the cipher text back into readable data, which is only possible by authorized parties, is called decryption. A key is needed for both encryption and decryption. We distinguish two kinds of encryption/decryption:

- **Symmetric** encryption and decryption uses the same key. Examples are DES, triple DES and AES.
- **Asymmetric** decryption requires another key than asymmetric encryption. The decryption key cannot be derived from the encryption key. Typical examples are RSA, Rabin and ElGamal.

Typically, symmetric encryption is considerably faster because it needs smaller keys for the same level of security. Encryption may be used for *confidentiality*, both for storing and sending data: only the entities possessing the key can obtain the plain text.

User specific parameters such as biometric data, a unique token, a private key corresponding to a user certificate, system parameters (CPU id, etc.), ... can be used (combined or not) to generate a key(pair) in order to obtain *device or person binding*.

Digital signatures. A digital signature aims to guarantee the *integrity* and *unforgeability* of the signed data, yet may provide some other properties as well [73]. Digital signatures can only be placed using a private key and can only be verified using a corresponding public key. The private key is kept secret and is impossible to derive from the public key. Some digital signatures schemes are RSA, DSA and Schnorr.

Hashes. A hash function is a transformation H with input m of random length, and output h of fixed length (thus h = G(m)). The output h is called the hash of the input m. A cryptographic hash function has the following properties: it is easy to calculate the output if one has the input. However, if the output is known, it is computationally infeasible to calculate the corresponding input (one way). In addition, it is computationally infeasible to obtain two inputs m and m' with the same hashvalue (i.e. H(m) = H(m')). Examples of hash functions are MD5 and SHA-1.

Certificates. A certificate guarantees the binding of a public key with the identity of the signer [73]. In principle, every entity can obtain a certificate of another entity by using a PKI (Public Key Infrastructure). Certificates thus enable *authentication* of the sole entity having the corresponding private key. Certificates may be revoked. The most common certificate type is X.509. Using specific hardware (or even software) parameters of the device or computer as key data, a client application can obtain a certificate, binding the application to the device. When the application is copied to another device or computer, its certificate runs invalid. Since the

corresponding private key can only be derived using these hardware parameters, certificates enable *individualization* of the client applications (such as a DRM client).

Watermarking [62] embeds some information into digital images, video, audio and some other data structures (not including text documents) without noticeably changing the content. The watermark has to be undetectable by human perception capacity. We distinguish between weak and strong watermarks.

Strong watermarks survive content manipulations such as D/A (Digital/Analog) A/D (Analog/Digital) conversion, conversion to other formats (e.g. MP3 to AAC), compression and decompression. They may be detected, changed or removed without loss of quality using the corresponding key. Without this key, the watermark can only be removed or damaged causing very serious degrading of quality. When identification information of the content owner signed by that owner is embedded as a watermark into the content itself, the content is bound to the owner, which implies *non-repudiation*. Of course, it is possible to embed other information. Yet, the quantity of information that can be embedded is generally rather limited, because the imperceptible part of the audio, video or image data is rather small.

Weak watermarks disappear when certain, predetermined manipulations are performed on the content (e.g. D/A A/D conversion), but survive others (e.g. compression). This type of watermark is ideal for checking the *integrity* of content.

Although rather good watermarks exist for images and video, sound is more troublesome. Designing a watermark that survives some of the requirements, such as robustness, imperceptibility or data rate, is possible, but designing one that fulfills all of these simultaneously has shown to be very difficult. It is one of the least mature technologies used in DRM.

Secure clocks are clocks that cannot be changed by their owners and prevent system time tampering, thus providing *secure time*. A secure clock is usually a hardware clock.

2.3.3 Other techniques

Next to the traditional cryptographic primitives, some other techniques are necessary to obtain integrity of the DRM client as well as confidentiality of the protected content and sensitive secret key information that the consumer may not obtain. It must be impossible, or at least very hard for a consumer to get hold of the decryption key or the decrypted content. This is far from trivial because the DRM client is installed on a computer or device that the consumer potentially fully controls. Besides expensive hardware solutions, some software solutions already exist or are in the research pipeline. A key player on the commercial market is Cloakware [4]. Cloakware is active on each of the techniques summed up here.

Trusted computing [48, 40] is a paradigm used to keep sensitive material within a trusted environment. Data are always sent from one entity to another. In a consumer environment, an entity can be a software application that sends the decrypted content to the operating system, which in turn sends it to an output device (sound card, video card, etc.) or to an external device. The idea is that the next entity in the chain first of all convinces the previous entity that the content it receives will stay inside a safe environment. Therefore, authentication as safe entity is needed, which can be provided using certificates that may be revoked when the entity is compromised. Secondly, when the protected content is consumed (and thus has to become unprotected at some point) the non-authenticated hard- and software is disabled. The Trusted Computing Group (TCG [40]), an alliance of Microsoft, Intel, IBM, HP and AMD, develops and promotes open specifications for trusted computing.

Code obfuscation [77] is a collection of techniques used to hide (part of) a program's implementation details and functionality from its (potentially adversary) user. Obfuscation involves a number of transformations on the program, which results in a new program that it is much harder to analyze. obfuscators are Zelix Klassmaster [45] and Semantic Designs [39]. When an entity wants to download an application, the server can individualize it by applying code obfuscation based on the hardware parameters provided by that entity. Each client has thus a separate executable.

White-box cryptography [54] [66] assumes that the adversary has total visibility of the software implementation and execution of cryptographic functions, and thus also of the key information. For critical functions, extra input and output encoding is applied, which results in a new, stronger function. White-box cryptography can be very useful to hide key information in a software DRM client installed on the computer of the consumer. Otherwise, the adversary could get hold of the content decryption key info by extracting the secret data out of the DRM client.

Self checking [63] refers to the checking of the code integrity (i.e. tamper resistance) by the application itself while running. Static integrity checking checks the code only once; at start-up-time. Dynamic self-checking verifies the code integrity of the application multiple times during execution.

Fingerprinting A fingerprint is a content-based compact signature that summarizes a media recording. A fingerprinting function F should map a media object M, consisting of a large number of bits, to a fingerprint V of a relatively small number of bits. The algorithm F uses unique properties of M (e.g. beats per minute, bitrate, ...) to generate V. The fingerprint V will typically be used in a database application as an index to the metadata of the corresponding media M.

2.3.4 Establishing security services

Section 2.3.1 located security hot spots and identified the necessary security services. This section outlines how these security services can be established in particular hot spots. Since the same security service may be established by different cryptographic primitives, depending on the situation, this section presents a mapping between security hot spots and cryptographic primitives that can be used to secure them (e.g. digital signatures, certificates, encryption, and digital watermarks).

Licenses. *Unforgeability* and *integrity* can be realized by applying digital signatures. *Binding* the license to its corresponding content can be done by inserting a unique identifier of the content in the license. Such an identifier may be a hash of the data, a fingerprint of the content, a Digital Object Identifier [47], or some other kind of identification. The content can be bound with a user can be done by inserting the user's personal certificate or a unique identifier of that certificate.

Content. When content is submitted, the combination of digital signatures and certificates can be used to obtain *integrity* and *authentication*. *Confidentiality* can be obtained using encryption. Obtaining confidentiality of DRM protected content has shown to be far from trivial. Content may be encrypted with a key that can only be reconstructed by a specific DRM client, potentially with the help of the consumer. Key reconstruction may depend on hardware specific parameters such as the CPU serial number. Another possibility is putting key data in the corresponding licenses. *Non-repudiation* in the detection mechanism of protected content can be obtained by a combination of digital signatures, certificates and watermarking. Identifying data signed by the consumer can be embedded as watermark in the content before it is encrypted.

Contracts. *Integrity*, *authentication* and *non-repudiation* can be obtained by using digital signatures combined with certificates.

Consumer system. To obtain *authentication* of the different trustworthy entities, combined with *confidentiality* of the sensitive data, Trusted Computing [48] techniques can be used. To decrypt the DRM protected content, the DRM client needs a secret key that is not known to the consumer. To prevent consumers or other entities from getting hold of the secret decryption data, white box cryptography [53] can be used. Such key data can also be used to maintain *confidentiality* of license status info (e.g how often it has already been consumed). To obtain *integrity* of the DRM client (and possibly other software entities) self checking techniques [63] can be used. Code obfuscation [77] is a usable technique to hinder sensitive code analysis. It also provides a way for *individualization*. To obtain secure time, one can use secure clocks, i.e. tamper-resistant hardware clocks.

Online DRM System. *Confidentiality* can be obtained using encryption, *integrity* using digital signatures and *authentication* using digital certificates. *Anonymity* can be obtained by removing any identifying data out of the usage information, while the usage information itself is sent to the Tracking Service using anonymizers [72]. By using digital signatures for authentication,

anonymity would be lost. Therefore, zero-knowledge proofs [67] that contain the usage information should be used instead.

2.4 Summary

In this section we have presented the main requirements for a DRM system, and used them to build a generic DRM architecture. In addition, we have discussed how security technologies can be integrated in the architecture: we located security hot spots in the architecture, presented the cryptographic primitives needed to build a secure DRM system, and discussed how low level cryptographic primitives can be used to establish the security components. We are now able to discuss the most important proprietary technologies and map their architecture on the generic architecture presented in this section.

3 State-of-the-art DRM technologies

In this section we provide an overview of the state-of-the-art in DRM by presenting publicly available information on eleven DRM technologies: Windows Media DRM (WMDRM), FairPlay, Adobe PDF Merchant/Web Buy, Light Weight DRM (LWDRM), Electronic Media Management System (EMMS), OMA, Intertrust, DMDSecure, Helix DRM, AegisDRM, and SealedMedia.

In Sections 3.1 to 3.6, we discuss the first six technologies from seven perspectives: context, architecture, security model, licenses and rights expression, known security attacks, security evaluation, and flexibility evaluation. The architecture perspective discusses the main DRM service components that are offered by each technology, and shows the relationship between these components by mapping them on the generic architecture that we presented in Section 2. After this in-depth discussion we briefly present the other five DRM technologies in Section 3.7. We conclude the overview in Section 3.8 by comparing the eleven DRM technologies based on their support for the security and flexibility requirements that we presented in Section 1.5.

3.1 Windows Media DRM

3.1.1 Context

Microsoft Windows Media DRM (WMDRM [29]) allows protection of audio and video content. It is used in Windows Media Player and uses the Advanced Streaming Format (ASF), the Windows Media Audio (WMA) format, or the Windows Media Video (WMV) format to protect content. WMDRM offers a set of Software Development Kits (SDKs), which are freely available, and some tools that help to set up DRM services. WMDRM protected content can be played on a Windows PC or portable device.

3.1.2 Architecture

WMDRM does not really provide a fixed architecture or (a set of) deployable applications. It consists of a set of SDKs that allow to use the core DRM services and combine them into various configurations. The following service components are offered in WMDRM:

- **Content Packaging:** protect compressed media
- **Content Hosting:** host and distribute digital media content
- **License Clearinghouse:** issue licenses and track transactions
- **Content Playback:** play protected digital media
- **Portable Device Playback:** transfer and play protected digital media
- **Network Device Playback:** play digital media from a remote source

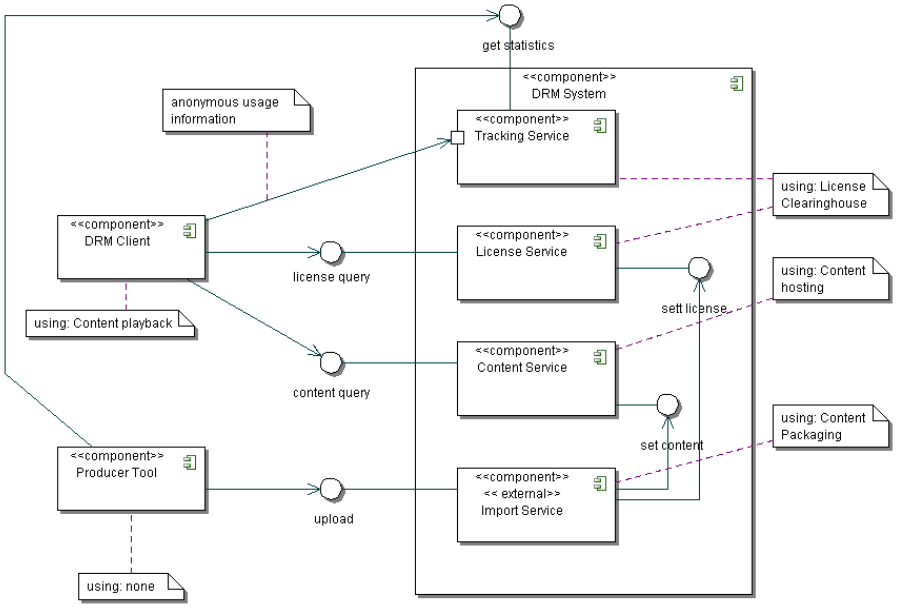

Figure 7: Possible simplified WMDRM architecture

In order to see how these service components can be combined into an overall DRM system, we map them on the generic DRM architecture. Figure 7 illustrates a relevant mapping and shows how each generic component maps onto a WMDRM component. As opposed to the generic architecture, WMDRM does not provide components for the Access Service and the Payment Service. In addition, the generic architecture distinguishes the Tracking Service and the License Service as separate components, while WMDRM combines these two in the License Clearinghouse component.

The architecture shown in Figure 7 is one of the possible architectures using SDKs offered by WMDRM. For clarity reasons, we have split the functionality of the Clearinghouse into a tracking and a licensing component. The figure does not show the access and payment components. We now discuss the individual components presented in this architecture.

Import Service. This service uses the Content Packaging SDK and has the same functionality as the Import Service in our generic architecture. It receives unprotected content and corresponding contracts from the content producer. It converts and protects the content and sends this result to the Content Service, while the protection data and contracts are sent to the License Service.

License Service. This service uses the License Clearinghouse SDK and has the same functionality as the License Service in our generic architecture. It receives the contracts and protection data from the Import Service.

Content Service. This service uses the Content Hosting SDK and corresponds to the Content Service in the generic architecture. It distributes content, but super-distribution is also possible. Both streaming and file download are supported. Also live streaming is possible.

Tracking Service. This service also uses the License Clearinghouse SDK and corresponds to the Tracking Service in the generic architecture. It gets its data from the different players. These data do not contain any identification data.

DRM Client. This component uses the Content Playback SDK.

3.1.3 Security model

The Packaging Service converts content to ASF, WMA or WMV. Also, an encryption key is generated from a secret master license key seed and a unique content key ID. The master license key seed is only known by the Import Service and the License Service. The key ID is put in the header of the file together with the URL of the license acquisition. The content is encrypted by the Import Service using the encryption key. The encryption is independent of the rights specification, which has the advantage that different sets of rights can be set for one protected file and rights can be changed without re-encrypting the content.

The License Service receives the content key ID from the Import Service. The License Service also knows the master license key seed and is able to reconstruct the content key, which is encrypted using the public key corresponding to the private key on the client device. This encrypted key and system parameters of the client device are part of the license. By consequence, the license is bound to the device. Also, a certificate obtained from the License Service is appended. The license is signed with the private key corresponding to that certificate.

WMDRM uses code individualization, renewability and revocation. The online DRM system can refuse service to DRM clients that are not yet renewed. Revocation is applied at two levels. Firstly, a blacklist with individual compromised clients can be checked. Secondly, it is possible to revoke third party players. Another technique is Secure Audio Path. This prevents interception of the audio on the data path inside the operating system during transfer from the media player to the sound card. A certified Microsoft component verifies that all downstream entities (including the sound card driver) are also certified. This is a kind of trusted computing. For DRM client security, Microsoft collaborates with Cloakware [4].

3.1.4 Licenses and right expression

WMDRM licenses are described by using the MPEG-21 rights expression language (REL). WMDRM licenses can specify varying usage restrictions, such as the number of times content can be played, time-related constraints, number of allowed copies, editing rules, or the required security level. In addition, licenses can enforce management rules that state, for instance, that playback on compromised or out-of-date systems must be prevented, or that periodic security updates are required. WMDRM licenses are device bound. The fair-use principle is not offered.

A consumer can download licenses directly and indirectly. A license can be acquired directly by a computer or device when it is connected to the Internet. Indirect license acquisition requires an intermediate computer proxy. In this case, the device is (temporarily) connected to a computer that contains licenses. If the license allows copies, the computer proxy can issue a new license to that device. This is enabled by a technology called "WMDRM 10 for portable devices". It is also possible to send content to authorized devices within a specified environment (e.g. a home network). This is enabled by a technology called "WMDRM 10 for Network Devices".

Some techniques used for licensing are worth mentioning:

- **License updating.** Expired or almost expired licenses can be updated quickly when connected to the Internet.
- **License chaining.** A root license can have several leaf licenses. A leaf license contains specific information about one piece of content (e.g. how many times it can be played). The root license is bound to a device and may, for instance, specify an overall validity period (e.g. one month). This allows to update a whole set of licenses by only updating the root license.
- **Support for secure clocks.** This allows the issuing of time-based licenses.
- **Output protection.** A license can specify what kinds of output a device may or may not produce (video/audio, digital/analog) and what kinds of security primitives must be attached to these outputs (e.g. Secure Audio Path).

Several license delivery scenarios are possible: pre-delivery (before the usage of the content), post-delivery (after the user tried to use protected content), silent delivery (without a dialog with the customer), or non-silent delivery (requires a dialog with the customer).

3.1.5 Attacks

A programmer with pseudonym "Beale Screamer" has released some documents [17] at the end of 2001 where he explains his vision on DRM together with the technical and security aspect of WMDRM. He also offered a tool called "FreeMe", which removes the WMDRM v2 protection of a WMA file. This is only possible when the user has a corresponding valid license. Because WMDRM does not use watermarks, the source of the abuse cannot be identified. Microsoft has been able to fix the security hole introduced by "Beale Screamer". Since Windows Media Player is integrated in the Windows operating system, the fixes could be downloaded and installed using the Windows Update tool.

Recently (August 2005), an unconfirmed message has been posted on the Internet announcing that WMDRM 10 has been cracked: with the help of a valid license, the DRM protection can be removed.

3.1.6 Security evaluation

WMDRM does not support watermarks, so there is *no fraud detection* system as second line of defense. The *control on distribution and consumption* was broken in 2001 but thanks to the *renewability* property it has been fixed fairly quickly. Yet, the fact that it might have been broken again in August 2005 indicates that features such as *permanent protection*, *tight coupling* between content, rights and consumer, and *tamper resistance* are under pressure. *Revocation* and *renewability* are present, as described in the security model. A pending danger is the single master key. Anybody who can get hold of it is able to decrypt all content from that Packager.

3.1.7 Flexibility evaluation

WMDRM only offers support for video and audio which can be offered as files, streaming, live streaming. The *rich content* requirement is thus not fulfilled. As explained in "Licenses and rights expressions", WMDRM offers *flexible usage rules*. The consumption of protected content is limited to the computer on which the license was downloaded, or one of the connected devices. Streaming within the home network is also possible, yet WMDRM does not support *ubiquitous consumption*. A disadvantage is the strong dependency on *Microsoft Windows platforms* and the lack of a directly deployable DRM system. WMDRM offers an *API* to the individual components, enabling developers to set up customized DRM systems in a flexible way. A *Publisher Tool* and a *Producer Tool* can be implemented.

Super-distribution is no problem in WMDRM, due to the absence of personalized protected content using watermarks. At the client side, backup of licenses can be made just as common data. Because the protected content is not personalized, this enables client-side recovery. It is also possible to implement the storage of licenses at server side. However, WMDRM does not offer explicit support for this. In case of a hardware failure, support can be offered to restore the licenses on another computer. WMDRM does not offer a *Payment Service*, but external payment services are available that can be used and integrated in WMDRM system. *Rights gifts* are absent but can be implemented as certificates.

3.2 FairPlay

3.2.1 Context

FairPlay is a DRM technology developed by Apple, but the company does not provide any information about the inner working of its DRM technology, making it difficult to discuss it thoroughly.

FairPlay is a fairly simple DRM technology and is used in the QuickTime multimedia technology and in the iTunes [26] application, which is intended to play, organize and buy music files. iTunes is used in the popular iPod music playing device. The only supported content type is audio, using protected AAC audio files. The actual decoding is performed by Apple's QuickTime, which encapsulates the DRM client of the consumer. In fact, every QuickTime player is capable of processing FairPlay encoded files, including RealPlayer and Media Player Classic.

FairPlay is considered a less restrictive DRM implementation, since it allows songs to be copied to any number of iPod devices and played on up to five authorized computers. In addition, tracks may be burned on a CD (which removes the DRM protection) and play lists may be burned on a CD up to seven times. Archiving to DVD is also provided.

3.2.2 Architecture

No information publicly available.

3.2.3 Security model

Most information about the security model has been leaked due to a successful attack.

The content is an encrypted AAC audio stream, packed within an MPEG-4 container. The encryption uses an AES symmetric key in combination with MD5 hashing for integrity. This symmetric key is called the master key and is stored on the server. When a user buys a song, for instance, the server generates a random user key that is used to encrypt the master key, which is then combined with the encrypted content into an MPEG-4 container which is sent to the user. Together with a content identifier, the user key is added to the account information on the server. This key is also sent to the user's device, where the iTunes application puts it into a key repository. Using the key repository iTunes is able to retrieve the master key. Using the master key, iTunes is able to decrypt the AAC audio stream and play it.

It is still unsure whether FairPlay, before the content is encrypted with the master key, embeds a watermark in the content containing the identity of the user. If the rumor is true, a Verance [42] watermark is used.

When a new computer is authorized, iTunes sends a unique machine identifier to the server, which in turn sends all the user key - content identifier pairs belonging to that user to the iTunes application. This ensures that a user can play all acquired iTunes songs on the newly registered computer, and that the maximum number of computers allowed to play content does not exceed the limit of five posed by apple.

When deauthorizing a computer, iTunes instructs the server to remove the unique machine identifier, and at the same time, it removes all the user keys from its encrypted key repository.

When a track is copied to an iPod device, iTunes also copies the user key, enabling the iPod to play the track. There is no limit on the number of iPods allowed to play content.

3.2.4 Attacks

After the launch of iTunes Music Store, different people made efforts to circumvent or even remove the DRM protection.

Jon Johansen, also known as author of the DeCSS program which removes the DVD protection, reverse-engineered the encryption technique in FairPlay and created an algorithm which is able to remove DRM protection without re-encoding. The first open-source application that was released was called called QTFairUse and was also written by Johansen. It intercepts the decrypted output and writes it to a file. This functionality has been integrated into the VLC media player [43]. Another software package called PlayFair does the same. Its successor is called Hymn [21]. Johansen also released his own tool, called Fairkeys [27].

There are a number of tools enabling the owner of FairPlay protected content to remove protection, even without loosing quality. These tools require, however, the user key, either from the server or the local iTunes or iPod key repository. Secondly, the watermark, if present, is left intact, which enables to identify who originally bought the file.

More recently (March 2005) a new tool, called PyMusique, appeared, which enables one to buy songs from iTunes, but instead of storing the protected audio, it first decrypts the audio. By consequence, the tool makes iTunes behave like a paid MP3 download site.

3.2.5 Security evaluation

By copying content to a CD, the prevention mechanism of FairPlay can be removed and so is any control on the *consumption and distribution of rights*. Depending on the presence of a watermark in the content, which is unclear today, *fraud detection* is possible.

The *permanent protection* requirement is not met due to the attack described above and the possibility to copy content to CD. Because no separate rights in the form of licenses are used, the *tight coupling of rights, user and content* and *tamper resistance* of consumption requirements cannot be supported. Revocation and renewability are likely not to be present.

Clearly, Apple uses the security-through-obscurity principle, which clearly has been proven not powerful enough to prevent successful attacks against it.

3.2.6 Flexibility evaluation

At the moment, FairPlay only supports music files. Thus, FairPlay offers *no rich content*. The usage rules are hard coded into the application. By consequence, FairPlay offers *fixed instead of flexible usage rules*. Five computers and unlimited number of iPod devices can be enabled to consume the content, but consumers cannot consume their content everywhere. The *ubiquitous consumption* requirement is thus not fulfilled. There is *no API* offered. FairPlay can only be used by players based on Apple technology. In July 2004, RealNetworks introduced the Harmony technology [18], which enables playing protected RealPlayer files on iPods. RealPlayer uses another DRM technology, called Helix DRM (see Section 3.7.3). Harmony transparently converts protected RealPlayer files to FairPlay-compatible protected files. iTunes is available on *MAC and Windows platforms* that run iTunes. It is not possible for a party to submit its own content to the system. There is thus neither a *Producer Tool*, nor a *Publisher Tool*. One reason of the success of FairPlay is probably its ease of use.

Super-distribution is not supported. It is possible to offer someone a *rights gift*: FairPlay calls it a gift certificate, which is in essence a digital pre-paid card, enabling one to download music until there is no more money left on the card. *Payment* is easy done using a pre-pay model: users can charge their account with an amount of money enabling them to download music files. Because the server stores the key-content identifier pairs, it is possible to offer *recoverability*.

It is a fairly simple system in the sense that it does not use licenses in which the usage and distribution rules can be set. These rules seem to be hard-coded.

3.3 Adobe PDF Merchant / Web Buy

3.3.1 Context

Adobe offers DRM protection for documents in PDF which is a popular digital document format that is open, publicly available, application and platform independent, and it allows to be printed exactly as shown on-screen. The DRM technology is described in [1] and consists of two entities: Adobe PDF Merchant and Adobe Web Buy. The first is the server application, the second the plug-in in acrobat reader, the PDF viewer at the client side.

Adobe PDF Merchant is a server-based technology designed to be integrated into existing e-commerce and transaction servers. It manages encryption of PDF files and the distribution of keys to access them. To provide consumers with a mechanism for purchasing and viewing electronic content produced using Adobe PDF Merchant, Adobe has implemented Web Buy, which is a standard plug-in in the free Acrobat Reader.

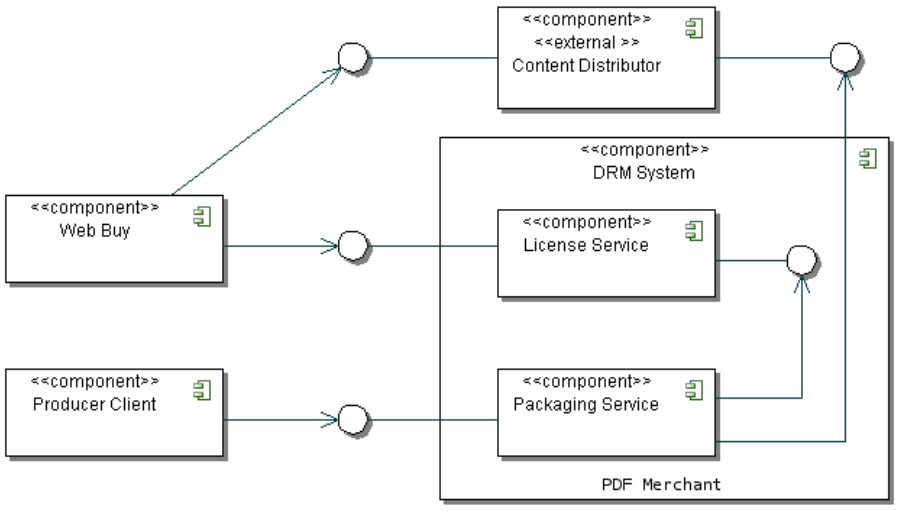

Figure 8: Possible architecture using Web Buy as DRM client and PDF Merchant as DRM system

3.3.2 Architecture

The functionality offered by PDF Merchant can be decomposed in multiple components. PDF Merchant does not have its own authentication or payment capabilities but can be integrated in other e-commerce applications that have their own authentication and payment handling. Also distribution is not a part of PDF Merchant; super-distribution (but also the distribution of unique copies) is possible using any possible medium or network.

Figure 8 illustrates one possible setup, where the functionality of PDF Merchant is spread over a License Service and a Packaging Service. Because authentication and payment are no parts of PDF Merchant functionality, we do not take them into account.

Web Buy. Web Buy corresponds to the DRM client of the consumer. When a consumer wants a license corresponding to some content, Web Buy contacts the license server, using the content-unique URL in the PDF file. To get a license (voucher) Web Buy provides the necessary identification information to the license server and accomplishes the payment.

Producer Client. The producer just has to submit its content and potentially corresponding rules to the Packaging Service. Contracts are not explicitly supported. No separate Producer Client is available but existing tools can be used to generate PDF documents. Some optimization of the document may be done, such as compression of the inserted images.

Packaging Service. This service, the Publisher in Web Merchant terminology, receives content from the content owner, and generates a content-unique key which is used to encrypt (lock) the content. The key is sent to the License Service. A unique URL is added to the file, pointing back to a website where the consumer can buy a license (called voucher). The Packaging Service also decides what identification data is required to allow users to access content.

Content Distributor. The encrypted content can be distributed in any possible way. An external Content Distributor can be used. Because the encryption is generally content (and not copy) dependent, super-distribution becomes possible.

License Service. This service creates licenses using the content-unique key and some identification data from the user side, such as CPU ID, removable disk ID, fixed disk ID, user name or e-mail. Each ID is put into the license together with an associated key, which allows the Web Buy plug-in at the user side to reconstruct the content-unique key.

3.3.3 Security Model

A license is an XML file that is signed with a 1024 bit key corresponding to a X.509 certificate and encrypted using 56-bit RSA encryption technology.

Because it is not possible to embed an imperceptible watermark into text, watermarking content with the buyer's id is not possible. It is, however, possible to show a visible watermark when the document is displayed or printed. The watermark can show variables such as transaction data and ID, user name, or the domain where it is consumable. This enables fraud detection by screen capturing or print rescanning. The watermark information is located in the license, not in the content. Typically, the Packaging Service will define whether a watermark must be present. Adobe expects more and more devices and computers to be equipped with a tamper-resistant clock, which is necessary to allow time dependent consumption rights.

3.3.4 Licenses and right expression

As pointed before, a license (voucher) consists of identification data (e.g. CPU ID) which compose the binding rules and can be combined by AND and OR connectives. Also, a UTC (Coordinated Universal Time) with corresponding key data can be added which allows to define time-related usage rules. There are no distribution rules in the current version. The usage rules involve printing, viewing, changing the document, selecting text and graphics, and adding or changing annotations and form fields. Their associated conditions are rather limited. Not only the licenses contain rules, but also the content itself. Rights granted by a license are only valid if they are also present in the content.

3.3.5 Attacks

A Russian cryptographer, Dmitry Sklyarov, has published in 2001 a tool that is able to remove Adobes protection on PDF files. He has been arrested by the FBI for violating the DMCA (The Digital Millennium Copyright Act). After his release, his employer has been accused. More information about the attack can be found on [8]. Now, some tools on the Internet allow easily removing DRM protection when a valid corresponding license is available.

3.3.6 Security Evaluation

Web Buy/PDF Merchant is a fairly simple DRM technology that was broken a few years ago. The *control on distribution* and *control on the consumption* were broken. It is impossible to integrate in a text document a watermark that cannot be removed. However, a visible watermark protection exists to detect screen grabs or print-rescans, but this will be of no help against the attack described above. By consequence, there is only very limited *fraud detection* possible. Due to the attack described above, *permanent protection* is no longer the case. The attack also broke the coupling between the content and the license. The coupling of license and consumer is in fact a coupling with one ore more devices of the consumer. There is no support for *revocation* or *renewability*. The licenses are made *tamper-resistant* by using digital signatures.

3.3.7 Flexibility Evaluation

Adobe focuses only on PDF documents and not on *rich content*. There are no distribution rules, and also the conditions are rather limited. We thus have *not so flexible usage rules*. One or more devices are able to consume content. Extra restrictions on e-mail or user names used are possible, but due to security reasons, these will probably not be used without device binding. By binding content to a removable disk, the content can be consumed on any device. We thus have a *very limited form of ubiquitous computing*. *No API* is available. Adobe focuses on *Macintosh and Windows platforms*. There are no Linux/Unix implementations. Advantages are that the format is open, widely accepted, cross-platform. Web Buy is, as feature within other applications, very

user friendly. PDF-merchant can be used as *Producer Tool*. Their is no real publisher functionality present (except maybe limited configurability, which can be done using the PDF Merchant).

Both *super-distribution* and unique copies are possible. *Rights gifts* are not possible. *Rights recovery* can be provided by the retailer and *payments* are done using an external Payment Service. There is little or no support for *fair-use*.

3.4 LWDRM

3.4.1 Context

The usual DRM schemes are strong in the sense that they enforce the usage rules at the consumer side very strictly (e.g. a consumer can play a song only ten times or only on three devices). According to Fraunhofer institute [16] this is an important reason why DRM is currently not very well accepted by the market. To overcome this problem, they developed Light Weight DRM (LWDRM) [28] which allows consumers to do everything they want with the content they bought, except for large scale distribution. LWDRM is an application of the fair-use principle.

LWDRM uses two file formats: the Local Media File (LMF) format and the Signed Media File (SMF) format. An LMF file is bound to a single device by a hardware-driven key but can be converted to the SMF-format which can be played on whatever device that supports LWDRM. However, the identity of the legitimate owner is embedded in the content. If such a file is found mass distributed, the source of abuse can be retrieved and prosecution can follow. Based on personalization instead of copy protection, LWDRM tries to balance between good protection and user-friendliness. Because rights are not made explicitly, no licenses are needed.

Although development of LWDRM has stopped since 2004, the concepts and services offered are highly relevant when comparing DRM technologies.

3.4.2 Architecture

The architecture of LWDRM consists of five main components which are shown in Figure 9: the Client Tool, the Accounting Service, the Content Packer, the Payment Service, and a Certification Authority.

Client Tool. The Client Tool is a consumer side application with the DRM client encapsulated. It is used by the consumer to register and to search, obtain and play content. The only component interface that is queried by the Client Tool is the Accounting Service interface. For being able to download protected content, the Client Tool has to do the correct payment using the Payment Service. If the consumer has not yet registered the device or computer to the DRM system, the Client Tool has to provide its device specific public key first, together with a prove of identity using signatures and digital certificates. Once the Client Tool has received the LMF content, the consumer is able to generate a corresponding SMF locally afterwards.

Accounting Service. The interface of this service is queried by the Client Tool to communicate with the online DRM system. The Accounting Service administers relevant information about the available content, such as price, meta data and ownership, and stores the user accounts together with all associated transactions and purchases. The content itself can be stored in a database on the DRM system. The Client Tool can search for content using the Accounting Service interface. Upon receival of a request for content from a Client Tool, the Accounting Service asks the external Payment Service a payment confirmation. If registration has not yet been fulfilled, the Accounting Service asks the Client Tool to provide a signed device specific system key together with a certificate. By contacting the Certificate Authority the certificate is validated. When the Accounting Service is convinced of both the consumer's identity and the fulfillment of the payment, it sends to the Content Packer a request containing the device specific public key, transaction number and content identifier.

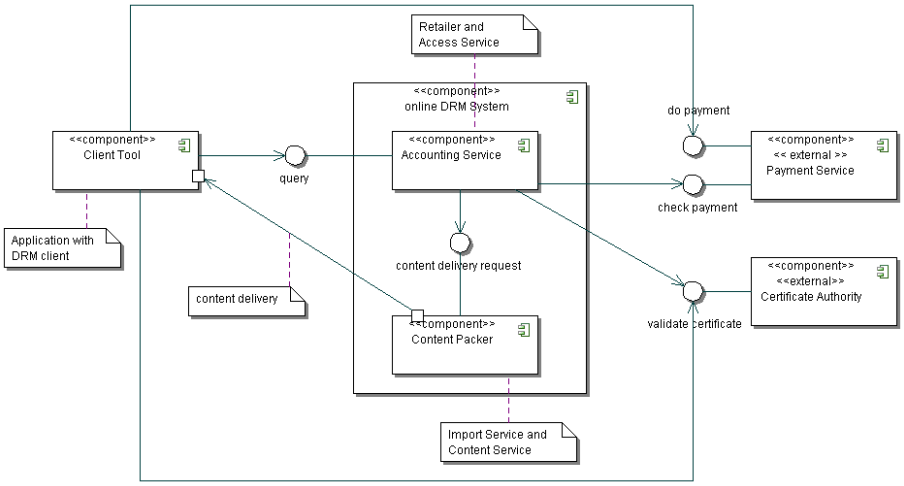

Figure 9: LWDRM architecture

The Content Packer. This component generates a new LMF file when it receives a request from the Accounting Service, and sends the LMF file, together with a receipt, directly to the Client Tool.

Payment Service. This service offers public interfaces to do and check payments.

Certification Authority. This is an external trusted third party which issues and validates certificates.

Currently no support is offered by Fraunhofer to add new content to the DRM system by different producers.

3.4.3 Security model

We will now explain in detail how both LMF and SMF are generated. We first explain LMF generation. When the Content Packer gets a request, it gets the device specific public key, the transaction number and (an identifier of) the content to protect. It also knows the provider ID. First of all, a strong watermark, consisting of a provider id and a unique transaction id, is embedded in the unprotected content. Secondly, a new, randomly generated (symmetrical) 128-bit AES session-key is used to encrypt the watermarked content. Then, the session key is encrypted using the (asymmetrical) device specific public RSA key. The resulting LMF file contains an asymmetrical encrypted session key and the watermarked content encrypted with the session-key. The LMF can only be decrypted if the system specific private key is known. The corresponding private key can only be derived by a single target device, using its hardware specific parameters. In case of mass abuse, the source of abuse can be determined using the transaction and provider id embedded in the content.

Next to an LMF file, the consumer receives a receipt signed by the DRM system with the private key of the provider who is using the DRM system. It serves as a proof of purchase for the user, containing information about the transaction, a content id, user id and provider id. If the rights holder is willing to grant the permission, it can also carry information about whether the content may be shared with others (”free” vs. ”private”).

To enable content consumption on multiple devices, consumers can export an LMF file to SMF format. The session-key contained in the LMF file must first be decrypted using the device specific private key. Then, the session key is encrypted with the private key corresponding to the users certificate. In this way, the binding of the content with a device is replaced by a binding with the

user. The SMF will contain the encrypted media-file, the encrypted session key, the certificate of the user, which is needed to decrypt the content, and a receipt signed by the content provider. The SMF contains all necessary information to play the file and it allows to verify the identity of the consumer who originally bought the song.

SMF content can be converted to LMF format, but such an LMF file cannot be converted back into SMF. In this way, content can be given to friends, but they are unable to distribute it further. It is also possible for consumers to convert unprotected content to LMF.

Although not mentioned in the official documentation released by Fraunhofer, it is likely that LMF and SMF are integrity protected by a signature placed by the content provider.

LWDRM has a serious privacy problem: an SMF container not only contains the certificate of the user, but also a signed receipt, containing information of where and when the song is bought. This can be solved by using the separation of duty principle as shown in [59]: one entity issues pseudonym certificates to users who can prove their own identity (typically by using a certificate). This pseudonym certificate is then used to buy content. When large scale abuse is detected, the pseudonym issuer can reveal the real identity to the police.

3.4.4 Security evaluation

Because the fair-use principle is used, *controlled consumption* and *controlled distribution* are not possible. If the watermarking schemes are strong enough, *fraud detection* is possible. LWDRM has never been seriously tested. So, the robustness of the watermarking schemes is still unknown.

Due to the fair-use principle, no rights are made explicit, and thus the *tamper-resistant usage rules* requirements can be omitted. In the *tight coupling between content, license and consumer* requirement, only the content-consumer binding is relevant and is fulfilled by embedding a watermark into the content. The trustworthiness of the DRM clients has not yet been tested. How good the *permanent protection* requirement is fulfilled, is also not yet clear. As far as we know, no information has been published on *revocation* or *renewability*.

3.4.5 Flexibility evaluation

LWDRM supports MPEG-4 audio and video. When using ISMA (Internet Streaming Media Alliance), not only files can be protected using LWDRM, but also media streams. It is possible to apply the same principle to images, if robust watermarking schemes are used. The *rich content* requirement is thus not fulfilled. The device bound LMF content can be converted to the device independent SMF content. *Ubiquitous consumption* is thus possible if you have access to the SMF content. When ordering content, the consumer gets a receipt, which can be used to prove its rights on the content after a system crash. At the content provider side, the Accounting Service keeps track of the transactions and purchases of each separate consumer. This can be used to *recover* the consumer's content. Disadvantages are the absence of a publicly available *API* (the default LWDRM application must be used) and the absence of the content producer side functionality. Nothing is mentioned about a *Publisher Tool* or *Producer Tool*. For being able to fetch protected content, a personal certificate is required. This is a drawback for user-friendliness.

Due to the application of watermarking, *super-distribution* is not possible. Due to the use of a private key associated with a personal certificate for obtaining content, it is not possible to give someone else content in a normal LMF. It is however possible to convert SMF content back to restricted LMF content. *Rights gifts* are thus only possible in a limited way. *Payment* functionality is integrated as pay-per-download in the default application. Some other exploitation models such as pay-per-consume and subscription are not possible in this scheme. The *fair-use* principle is used. The source of large scale abuse is still easily traceable. Because of this principle, it is not needed to explicitly formulate the usage rules and to use a REL. By consequence, the requirements on flexible usage rules do not apply here.

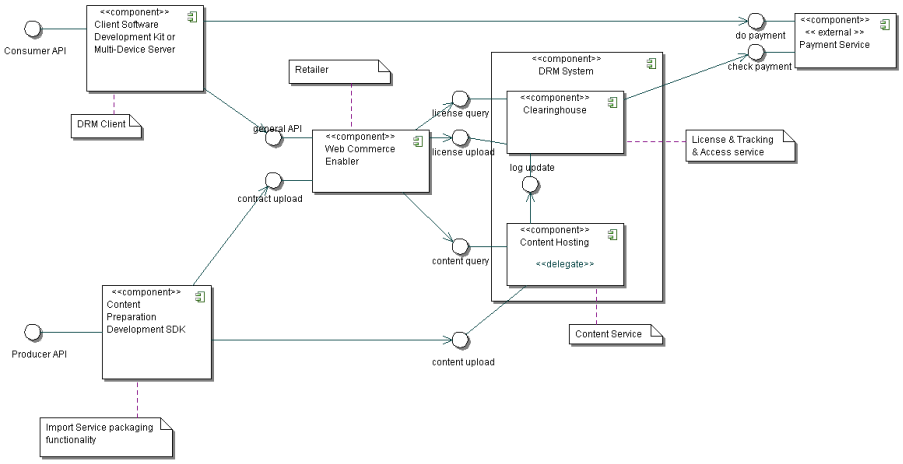

Figure 10: EMMS architecture

3.5 EMMS

3.5.1 Context

IBM's Electronic Media Management System (EMMS) [22] offers DRM protection for video, music, documents, rich media and software, independent of the format used. Streaming is supported. EMMS uses secure container technology to offer tamper resistant containers of data. EMMS plug-ins are available for RealJukeBox (RealNetworks) and MusicMatch JukeBox (MusicMatch) by downloading plug-in. Both MusicMatch and RealNetworks still support their own competing DRM technologies. They also support Intertrust technologies and WMDRM.

Recently (June 2005), IBM announced the withdrawal of EMMS from the market without releasing a replacement product. Nevertheless it is still interesting to map its high level components to the generic service components identified in Section 2.

3.5.2 Architecture

The whole architecture shown in Figure 10 consists of seven components which could be bought separate or in group. We will describe them and show how this fits in our general architecture.

Web Commerce Enabler. This service corresponds to the Retailer in the generic architecture. This component facilitates and supports the integration of EMMS DRM into web applications and enables consumers to find and obtain content and corresponding licenses. Transaction information of digital content is gathered and sent to the Clearinghouse. The Web Commerce Enabler allows a myriad of business models: subscription, geographically constrained, wholesale, time based, pay-per-use, promotions, etc. Also promotional data can be sent by the content producer to the Web Commerce Enabler.

EMMS Client Software Development Kit. This service corresponds to the DRM client and offers an API enabling development of consumer applications. It enables to browse, buy and download content and licenses on Web Commerce Enabler servers. It also offers an interface to a tamper resistant library for managing the DRM protected content. The Secure Access Manager, which is part of this SDK, facilitates control over transfer of content, meta data and usage conditions to external devices (e.g. CD-ROMs) in formats specific to the receiving device or application.

Multi-Device Server. This server offers support to transmit digital content to wireless devices and to implement on-demand production of CD's or portable medCd'sn accordance with the license. The content can be repacked in order to meet the requirements of the target device.

Content Preparation Development SDK. This corresponds to the Producer Client interface combined with the Import Service in the general architecture and enables the development of applications specific to the content industry. It is used by content owners and distributors. Content can be protected (i.e. put in a secure container), a corresponding contract can be made and meta data can be specified. The protected content can be distributed using any means, but is usually sent to a server running the Content Hosting Program. The corresponding key data is sent to the Clearinghouse via the Web Commerce Enabler. Extra promotional data can be provided to the Web Commerce Enabler. The EMMS Enabler for PDF is an addition to the Content Preparation Development SDK needed to package and disperse protected PDF documents. Instead of using an SDK, a full program called the EMMS Content Mastering Program can be used. This program however only supports the packaging of audio content.

EMMS Hosting Program. The task of the EMMS Hosting Program is pure distribution and sending statistical info about the distribution to the Clearinghouse. The functionality of the program corresponds to the Content Service in the general architecture and offers extensive support for hosting. This component receives protected content from the Producer Client (who uses the Content Preparation Development SDK). Multiple collaborating hosts can be set, allowing content replication, hierarchical topologies, caching of hosts, spreading of workload and geographical distribution. All distribution activity is tracked and sent to the Clearinghouse. The protection of the content was already done by the producer using the Content Preparation API.

Clearinghouse. This is the central control point that corresponds to the License Service, the Tracking Service and the Access Service in the general architecture. It receives its tracking data from the Web Commerce Enabler and the Content Hosting programs and can generate license transaction tracking data itself and is thus able to generate extended reports. It receives key data and contracts from the Web Commerce Enabler, which thus enables the Clearinghouse to issue licenses. Finally, authorization of transactions is done here. Therefore, contacting an external Payment Service may be needed.

3.5.3 Licenses and rights flexibility

Licenses are defined using ODRL, which offers an extensive and flexible way to define rules in the licenses. Region bound usage rules can be set, the number of times a defined action is possible on the content, the time interval in which it is consumable, the time a copy can exist. The possibility to share content is also present.

3.5.4 Security model

No relevant information publicly available.

3.5.5 Security evaluation

Controlled distribution of rights and *controlled consumption* are possible. There is no watermark present in the protected content and thus *no fraud detection*. Nothing is known about the reliability and robustness of the solution, except that no successful attacks are published. At the moment of writing.

3.5.6 Flexibility evaluation

A wide range of Common media formats such as MP3 are supported. EMMS is codec independent and covers most of the usual applications associated with multimedia content, such as eBooks, music or videos. It is not clear if combinations of these are supported and if they can be described separately. The rich content requirement can thus be fulfilled. The *flexibility of the usage rules* in EMMS is good. It is unclear to what level *ubiquitous consumption* is possible. Many business

models are possible. It is reasonable to state that *fair-use* is hard to achieve when using EMMS. The SDK allows the integration of EMMS into each player, but these *APIs* are not publicly available. The different components have to be bought. A cryptographic co-processor is needed for being able to run the EMMS Clearinghouse program and most components run only on Windows NT or Windows 2000 *platforms* (except the EMMS Web Commerce Enabler , which runs on Windows NT/2000, AIX, Sun Solaris, Hp-UX and DEC UNIX). Most components also need a DB2 database. Using the SDKs, *Publisher Tools* and *Producer Tools* can be developed. The protection is content specific and not copy-specific, which enables *super-distribution*. However, this introduces a risk since watermarks are not possible in combination with super-distribution. No information is found about *rights gifts*, *recoverability*, or *ease of use*.

3.6 OMA specification

The Open Mobile Alliance (OMA [34]) is a forum formed in June 2002 which tries to offer a comprehensive open specification to enable interoperability between service providers and mobile devices. OMA allows companies to implement different versions of the specification, while maintaining interoperability.

OMA focuses on DRM for mobile devices, i.e. protection of ringtones, backgrounds, screensavers, etc. In June 2004, OMA released the DRM Approval Enabler which includes three types of functionality: Forward Lock, Combined Delivery and Separate Delivery [61]. Forward Lock imeans that content meanskaged into a special container format, called a DRM message. The content is not encrypted, but the DRM message and the included media object may not leave the receiving device after reception. It may be stored on the device and consumed without restrictions, but strictly on that device. Forward Lock is typically applied to low value content. This type of protection can be used, for instance, by subscription-based services that send news to mobile devices.

The second type of protection, Combined Delivery, extends the concept of Forward Lock by giving the possibility to define more fine-grained rights and permissions than the simple forward-lock restriction. As in Forward Lock, the content is packaged into a special container format, the DRM message. However, in Combined Delivery a rights section is included in the DRM message, in addition to the media section containing the content. The rights section contains rights and permissions relating to the media object and specified using the OMA rights expressions language.

The third type, Separate Delivery, is a logical extension of Combined Delivery. As in Combined Delivery, it is possible to specify rights for media objects. However, rights and media objects are now transported separately in two objects, compared to the combined delivery where content and rights are transported in the same object. The media object is always encrypted and converted into the DRM content format (DCF). The device may forward (super-distribute) the protected DCF file to another device. However, rights objects are not allowed to be forwarded, i.e. the receiving device must acquire rights for the media object from the issuing server.

In December 2004, v2.0 Candidate Enabler was published. This enabler should counter the shortcomings of the v1.0 Approval Enabler and builds upon Separate Delivery. This section will discuss the V2.0 Candidate Enabler.

Beep Science [3] and DMDsecure (see Section 3.7.2) already provided an OMA v1.0 DRM implementations. CoreMedia [7] has a deal with Vodafone to make an implementation compliant with both OMA v1.0 and OMA v.2.0. SanDisk [36] announced on 14 march 2005 that it will produce OMA DRM compliant storage. Nokia and Microsoft announced in februari 2005 to buiFebruarydge between Windows Media DRM and OMA DRM.

3.6.1 Architecture

The architecture of OMA is shown in Figure 11. At the server side, we identify a Content Issuer and a Rights Issuer which correspond to the Content Service and the License Service in the generic architecture. The DRM Agent (i .e. the DRM client) receives protected content via super-distribution from another DRM Agent or can search protected content on, and download it from

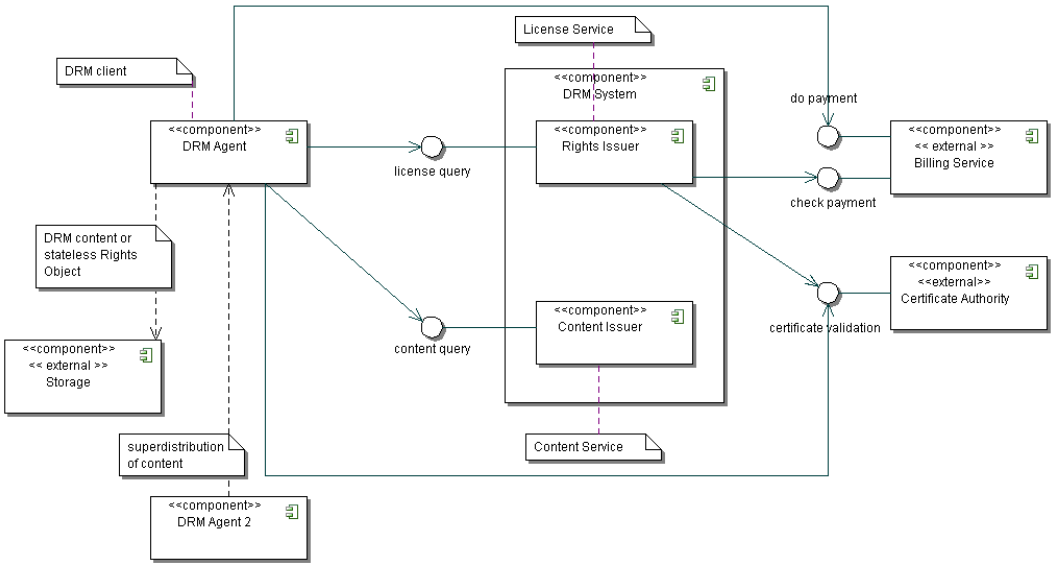

Figure 11: OMA architecture

a Content Issuer.

For being able to preview or consume content, the DRM Agent contacts the Rights Issuer to obtain a Rights Object (i.e. a license). Lost or damaged Rights Object can be restored via the Rights Issuer. A DRM Agent can also request the Rights Issuer to join or leave a domain which is a set of DRM Agents able to use the same Domain Rights Objects. A domain could, for example, be the set of devices owned by a specific user. Not all devices in a domain need access to the network for being able to consume content. Communication with another device in the domain on which the content and Rights Objects reside is enough. This is called Unconnected Device Support.

Before DRM Agents are able to download a Rights Object, they have to complete a registration procedure with the Rights Issuer. The DRM Agent is obliged to prove ownership of a certificate which is checked by the Rights Issuer for validity by contacting an external Certification Authority.

Content can be pulled by the DRM agent from the Content issuer, but it can also be pushed to the DRM Agent by the Content Issuer. This can be done for subscription based services where the DRM Agent receives for example the daily news items. A last possibility is push-initiated pull: a link to the protected content is sent by the Content Issuer to the DRM agent. How content can be added to the DRM system by content consumers is not specified in the OMA DRM Enablers. Components such as the Import Service or interactions with these components are not defined by OMA.

3.6.2 Security Model

Content is packaged in a DRM Content Format (DCF) secure container. It is encrypted using a symmetric Content Encryption Key (CEK) and signed by the Content Issuer. The key is content specific, which poses a security risk: if a device is compromised, its CEKs can be exposed. Therefore, it is recommended by OMA not to use the same CEK for all content instances. Pre-packaging by the Content Issuer using a CEK is easy. No watermarking is used to protect content.

The CEKs are sent to the Rights Issuer. This enables the Rights Issuer to create Rights Objects on request. A Rights Object contains, next to a set of usage rules, the CEK of the corresponding protected content. The CEK and other sensitive data is encrypted with a public key specific for a DRM agent, enabling the DRM Agent to consume the content. The Rights Objects are signed by the Content Issuer.

The DRM Agent has its own, unique key pair with an associated certificate, which is needed for authentication to the Rights Issuer. When a type of DRM agents is compromised, all certificates of that type of agent can be revoked. A DRM Agent Certificate also enables Rights Issuers to decide whether to trust a DRM Agent. A DRM Agent needs a secure clock for time based rules.

The Content Issuer has a certificate to authenticate itself to the DRM Agent. All DRM Agents in a domain have a Domain Key, which enables them to access all content within that domain.

The OMA DRM protection is independent of the type of content, and supports both delivery as a file or as a stream. The DCF container has some unencrypted fields where the URI can be found to fetch a Rights Objects. A DCF container can be stored off-device, typically as backup.

3.6.3 Licenses

A license is called a Rights Object in OMA context. A Rights Object is bound to a specific DRM Agent on a specific device, Rights Objects can also be bound to a group of DRM Agents, called a domain.

An important property is the possibility to assign different usage rules to different parts in a composite content object or one license to different pieces of content. Rights Objects can be copied and stored as backup if they are stateless. For example, when the number of plays is limited, a backup of the Rights Object is impossible.

Allowed conversions to other DRM Systems can be specified.

The Rights Expression Language used is OMA REL. This is an extension of ODRL.

3.6.4 Security evaluation

Revocation is provided in the specification. The other security aspects are implementation dependent, so no conclusions can be drawn concerning security characteristics of OMA.

3.6.5 Flexibility evaluation

Rich content with different rights for the subparts are possible. XrML is used as REL, which makes the *flexibility of the usage rules* very high. Domains of devices can be specified which allows a limited form of *ubiquitous computing*. Because this is an open specification, the *API* is evidently available. The specification is *platform* independent. *Super-distribution* and *recoverability* are covered. The *payment* methods do not belong to the OMA DRM part. *Fair-use* is not allowed. Nothing is said about *rights gifts*.

3.7 Other proprietary technologies

In this section we discuss five proprietary DRM technologies or DRM related companies, but not as extended as the previous six, since less relevant information is publicly available. We discuss Intertrust, Helix DRM, AegisDRM, DMDsecure and SealedMedia.

3.7.1 Intertrust

Intertrust [24] is a company which is founded in 1990 and specializes in DRM. Although it does not offer a complete DRM system, it is one of the most important actors in the field. It has an extended patent portfolio on intellectual property for DRM, Digital Policy Management (DPM), and trusted computing. It currently holds 31 U.S. patents, 12 international patents and over 100 patent applications. The company has license agreements with over 40 companies, among which Adobe, Microsoft, AOL Time Warner, Universal Music and BMG. In November 2002, the company was taken over by Sony and Philips. In April 2004, Intertrust and Microsoft reached a settlement in a lawsuit initiated by Intertrust in April 2001 due to patent infringement. In April 2005, Intertrust prevailed in a patent dispute with Macrovision. Intertrust is member of the Coral Consortium and its successor, Marlin Joint Development Association (Marlin JDA, see Section 4.2).

The patents of Intertrust cover a wide variety of different aspects, among which the most important ones are summarized below.

- **Management of web services:** Protection and management of digital applications and content wherever they travel, reside or are used.
- **Executable software integrity:** Authentication of software components to the operating system based on integrity and reliability rules.
- **Credentials/driver signing:** Authentication of executables to the operating system based on integrity and proper behavior.
- **Supply chain management through independent delivery of rules:** Enabling to define and enforce usage rules or policies related to digital content.
- **Managing media content or enterprise information:** Enabling entities to define and enforce usage rules, independent of the location where the content or information resides or travels.
- **Enterprise-to-enterprise transactions:** Automation of enterprise transaction according to enterprise policies. Possible transactions are authorizing, purchasing, auditing, reporting, and clearing of supplies and inventory.
- **Compliance:** Secured and automated tracking of transactions and usage based on enterprise policies and regulatory requirements.
- **Portability of rules:** Allowing to loan or move content between consumers or devices.
- **Nested policies within a single item:** Allows the association of multiple sets to different parts in the content.
- **Silicon protective measures:** Technologies for hardware security as an integrated component of a distributed trusted computing network.

3.7.2 DMDSecure

DMDsecure [12] is a company (recently taken over by SafeNet) that offers a range of DRM-related products: DMDlicenser, DMDmobile, DMDfusion, OMA DRM Server Toolkit and Multiple DRM Server Toolkit. DMDsecure focuses on online DRM clients.

- **DMDlicenser** [10] corresponds to the online DRM system in the generic architecture. It uses WMDRM 9 & 10 functionality (see Section 3.1) and provides all the DRM features provided by Microsoft. DMDsecure guarantees interoperability between WMDRM clients. A usage profile engine allows one to dynamically manage rights on content. A customer information engine allows the use of virtual user domains and usage metering. A virtual user domain manages the consumption of content on multiple devices. Usage metering does not need a permanent back-channel. The condition pipeline allows for integration with external authentication, subscriber management, and billing systems, while it also enforces contractual obligations such as limits on license volumes, or time-based constraints on license deliveries. DMDlicenser uses open and standardized APIs which enable easy integration into existing infrastructures. The corresponding Producer Tool that is offered is called the DMDpackager. Content can be distributed over any IP-based network.
- **DMDmobile** [11] is a server solution that implements the OMA DRM specification. DMDmobile consists of two basic components: a Protection Component and a License Component. The Protection Component corresponds to the Import Service in the generic architecture and protects content according to the OMA specifications. A web-based GUI is provided for content producers. The License Component corresponds to the combination of the Producer

Tool and the Import Service in the generic architecture. It allows content producers to compose sets of usage rules and to apply them to content. The issuance of licenses itself is done by a third party. Notice that DMDmobile does not offer a DRM-client implementation. In the spirit of OMA, DMDmobile is an open and pluggable architecture designed to be secure and extensible. It supports a wide variety of exploitation models.

- **OMA DRM Server Toolkit** [13] provides an API to core DRM operations as defined in OMA DRM as well as DRM features that are independent of the underlying DRM technology. Documentation and sample code is included in the toolkit. The core operations support content protection, rights management, license generation, and license delivery. The toolkit can be deployed in any Java environment.
- **DMDfusion 5** [9] is an interoperable DRM server solution. Currently, it implements and extends WMDRM and OMA DRM. The DMDfusion API is designed to integrate with existing encoding, content management, content delivery, subscriber management and billing infrastructures. It offers DRM agnostic functionality. Just as the DMDlicenser, a customer information engine, a conditions pipeline and a usage profile engine is present as extension to the basic functionality. The corresponding Producer Tool that is offered as part of DMDfusion is called the DMDpackager. Also a DMDmeter is present which corresponds to the Tracking Service in the generic architecture. Content can be distributed over any IP-based network.
- **Multiple DRM Server Toolkit** [13] provides an API to developers such that they can use all functionality required to include support for WMDRM and OMA DRM. The API is DRM technology agnostic. The same interface can be used for generating content and licenses of both OMA and WMDRM. The toolkit can be deployed in any Java or .NET environment, and it provides core implementations, documentation and sample code. To realize this abstraction of DRM technologies, a layered architecture is proposed, consisting of four layers: the Data Abstraction Layer, the Technology abstraction Layer, the Conditions Abstraction Layer and the Service Abstraction Layer which is optional.

3.7.3 Helix DRM

Helix DRM [20] was announced on January 9, 2003 by RealNetworks. It is designed to be integrated in existing e-commerce applications. Multiple business models can be applied such as subscription, pay-per-consume and promotions.

Helix DRM focuses on a wide variety of video and audio formats. It supports, for instance, RealAudio, RealVideo, AAC, MP3 and MPEG-4. Streaming, downloading and other delivery methods are possible (using super-distribution). Both the client plug-in and the DRM system itself can be installed on different platforms such as Win32, Sun Solaris, Linux 2.2, HP-UX and AIX.

The promote Helix DRM, RealNetworks released part of the source code [19]. The more 'sensitive' parts are not released by RealNetworks.

Helix DRM consists of four key components:

- **Helix DRM Packager** uses encryption and secure container technology to protect content. Watermarks are not mentioned. Streaming, live streaming, downloading and super-distribution are possible.
- **Helix DRM License Server** plays the role of the License Service, the Tracking Service and the Access Service. It verifies license requests and issues licenses, provides auditing information to facilitate royalty payments, and it allows to manage, authorize, and report content transactions.
- **Helix DRM Client** is the consumer's DRM client and allows streaming and playback of content and offers a tamper resistant environment to consume content according to the usage rules specified in the license.

- **Helix DRM for Devices** enables device manufacturers to equip their devices with Helix DRM support. Two integration models are currently offered: Secure Memory Device and Secure Streaming Device. These models allow manufacturers to deliver content to devices via a home-network or to interact directly with DRM services.

Helix DRM distinguishes between content and usage rules by using licenses. Multiple license types are possible on the same content. Usage rules can limit the times a piece of content can be consumed, the time-frame in which consumption is possible, and the duration content can be consumed (e.g. only the first minute). Revocation of licenses is possible. There are no distribution rules.

3.7.4 AegisDRM

AegisDRM [2] is a DRM technology that focuses on intra-company content distribution. The four main components that are offered are the Protector, the Protector add-in Module (PaM), the RightsServer, and the LicenseMaster.

- **Protector**^{TM} corresponds to the Producer Tool combined with a producer-side Packager. It allows content producers to specify various usage rules (e.g. for screen-grabbing, printing, copying, cutting-and-pasting, forwarding and saving). It offers a myriad of content file types that can be protected, such as HTML pages, flash, PowerPoint and Excel), and PDF documents.
- **PaM**^{TM} (Protector add-in) corresponds to the Producer Tool combined with a producer-side Packager. It is a plug-in for MS Office that allows producers of office documents to specify who is allowed to see which document and when.
- **RightsServer**^{TM} corresponds to the combination of the License Service (although no real licenses are used), the Tracking Service and the Access Service. Every time consumers want to perform actions on some protected content, they must contact the Content Service and authenticate to the RightsServer. Consumers can only perform the action if the RightsServer gives permission to the Content Service. The RightsServer keeps an audit trail of who did what and when. The RightsServer allows changing the rights at all times. Using the RightsServer, information can be changed or revoked at any time.
- **LicenseMaster**^{TM} corresponds to the License Service. It is an alternative to the RightsServer. When the LicenseMaster is used, super-distribution is possible. The LicenseMaster focuses on protecting applications. An application can be activated by using a single-use 25-digit code.

3.7.5 SealedMedia

SealedMedia [37] offers an intra-company DRM system which supports 14 Windows-based content formats: Microsoft Outlook, Lotus Notes, Microsoft Word, Microsoft Excel, Microsoft PowerPoint, Adobe PDF, HTML, GIF, JPEG, PNG, MP3, Apple QuickTime, MPEG-1, and MPEG-4. Apple supports the same formats except for Microsoft Outlook, Lotus Notes and Microsoft Office.

SealedMedia offers three main components: the Sealer, the Unsealer and the License Server.

- The **Sealer** is run by the producer and corresponds to the Producer Tool and the Import Service in our generic architecture. The Sealer communicates with the License Server to allow rights distribution.
- The **Unsealer** corresponds to the DRM Client and operates transparently for the consumer: it becomes only visible in exceptional circumstances, for instance when a license expires when the consumer is offline. The Unsealer communicates with the License Server to obtain licenses.

Security requirement	**WMDRM**	**FairPlay**	**Adobe**	**LWDRM**	**EMMS**	**OMA**
Controlled distribution	Broken	Content is exportable	Broken	Content is exportable	Yes?	N/A
Controlled consumption	Broken	Broken	Broken	No	Yes?	N/A
Fraud detection	No	?	No	Yes	No	N/A
Permanent protection	Broken	Broken	Broken	?	Yes?	N/A
Tight coupling	Broken	Broken	Broken	?	Yes?	N/A
Renewability	Yes	No	No	?	?	N/A
Revocability	Yes	No	No	?	?	Yes
Tamper-resistant usage rules	Yes	Yes	Yes	Yes	Yes	N/A

Table 1: Comparison of security requirements

- The **License Server** corresponds to the License Service and the Tracking Service in the generic architecture. Licenses are issued and all consumer activity can be logged. A graphical management console is offered allowing content publishers to control every aspect of the License Service operation.

Because SealedMedia is intended for intra-company use, no Payment Service is needed. Content is super-distributed, in practice the company network will be used for distribution of content. By consequence, no integrated Content Service is needed. An external Access Service is possible.

The renewability and revocability requirements are fulfilled: the License Server can force consumers to update their Unsealer before being able to consume content. SealedMedia uses its own trusted clock. Rights can be revoked instantly. SealedMedia supports three authentication mechanisms: its own user name and password authentication, NT authentication, and server-side authentication. NT Authentication checks the user name that is currently using the operating system. Server-side authentication uses an Access Service.

The producer defines who can access its content. Rights can be assigned to a single content item or a content group. Both a start-time and an end-time can be set. Rights can be revoked instantly. The maximum offline period for being able to consume content can be set. Some items can be locked. It is possible to determine whether content can leave the (company) network. The following actions can be set: print, annotate, copy-paste, save, seal, read-only access, edit. The degree of editing freedom can also be set: consumers may have full editing freedom, may add components or may only enter data. Changes can be tracked.

3.8 Summary

In this section, we have discussed the most important proprietary technologies and matched them to the requirements outlined in Section 1.5. By way of wrap up, we summarize how each technology supports these requirements in two tables. The first table shows an overview of security requirements, while the second summarizes flexibility requirements.

3.8.1 Security requirements

All the discussed technologies use the "security through obscurity" principle. When information about the fulfillment is found, it is often due to a successful attack. The different attacks are based on the same principle: sending the unprotected content to a file instead of the output channel (audio/video output, printer, display, ...). This attack breaks several requirements at once, as can be seen in table 1. WMDRM is the only technology could be fixed after a successful attack, but recently it has been broken again.

3.8.2 Flexibility requirements

Although most DRM technology providers do not disclose much information about their support for flexibility, we were able to deduce most of the implicit information and to check the requirements defined in Section 1.5. Table 2 summarizes our study on flexibility requirements.

4 Open Standards

We have seen in the previous part the most important proprietary DRM technologies. These are generally closed source and offer limited interoperability. Some open standards are being developed to overcome this problem. We first discuss in Section 4.1 MPEG-21, which defines an open multimedia framework for the distribution and consumption of content. In Section 4.2 We briefly discuss Coral and its successor Marlin. Coral wants to offer interoperability between different existing DRM technologies, while Marlin uses this to develop a standard specification for content management and protection. In Section 4.3 and Section 4.1 we discuss two expressive and extensible competing Rights Expression Languages: ODRL and MPEG-21 REL [56, 60]. The former is adopted by OMA (Section 3.6), the latter by MPEG-21.

4.1 MPEG-21 framework

Both production and consumption of digital content is increasing at a rapid pace. Therefore, the Moving Pictures Experts Group started in 2001 with the specification of MPEG-21, an open, standardized multimedia framework. It enables the use of content across a wide range of networks and devices. Production, publication and consumption of content using many different platforms is taken into account. Of course, Digital Rights Management is an important property in the framework and a standardized protection mechanism is needed. More information about this can be found on [30] and a good introduction is available on [51].

MPEG-21 defines a framework using existing open standards, if present, and otherwise, it develops new standards itself or engages other organizations or groups to do so. Different Calls for Proposals where issued, resulting in the adoption of proposals as standards in MPEG-21. MPEG-21 is supported by OMA (see Section 3.6), ITU [23] and the Open eBook Forum [33].

The two key concepts in MPEG-21 are the Digital Item and the User. The Digital Item is what we called content, and is the basic entity of distribution and interaction. the User is the entity that interacts with Digital Items. The goal of MPEG-21 can now be redefined as "to access, consume, trade and otherwise manipulate Digital Items in an efficient, transparent and interoperable way". From a technical perspective, however, MPEG-21 makes no distinction between content producers and content consumers: both are users.

Besides client-server, also peer-to-peer application are taken into account while defining the framework.

MPEG-21 is organized and developed in several independent parts. A significant number is already finished:

Vision, Technologies and Strategy was the first technical report and was approved in September 2001. It describes the future multimedia framework and its architectural elements together with the functional requirements for their specification.

Digital Item Declaration (DID [52]) specifies abstract terms and concepts enabling to describe the structure of interrelationships between Digital Items. The DID part contains three sections: Model, Representation and Schema. Model describes the terms and concepts, Representation contains a normative descriptions of the syntax and semantics of each DID element in XML and Schema describes a normative XML schema comprising the grammar of the DID representation in XML.

We now briefly describe the most important concepts of the DID. Most of these are illustrated in Figure 12. The smallest identifiable, unambiguous locatable piece of content is called a resource. A fragment of a resource designates a specific point or range within that resource. A descriptor associates information with the enclosing element and is a component or a statement. A component

	WMDRM	FairPlay	Adobe	LWDRM	EMMS	OMA
Rich content	Audio, video	Audio	PDF-documents	Audio, video	Content: yes.	Yes
Flexible usage rules	Good, flexible	Hard coded	Average	N/A	Good	Good
Ubiquitous consumption	Devices, network bound	Max. 5 devices, exportable	Device bound	Device bound, exportable	?	Limited
API available	Yes	No	No	No	Yes	Yes
Publisher Tool	Implementable	No	?	No	Implementable	N/A
Producer tool	Implementable	No	Yes	No	Implementable	N/A
Easy use	-	-	-	Certificate needed	?	N/A
Multi-platform	Windows	Mac & Windows	Mac, Windows	Windows, Mac, Linux	Several platforms	Yes
Super-distribution	Yes	No	Yes	No	Yes	Yes
Rights gift	No, but extensible	Yes	No	No	?	?
Recoverability	Client side: yes Server side: possible	No, but possible in future	Possible	Possible	?	Possible
Easy Payment	External	Pre-pay	External	Per download	Many possibilities	External
Fair-use	No	No, but less restrictive	No	Yes	?	No

Table 2: Comparison of flexibility requirements

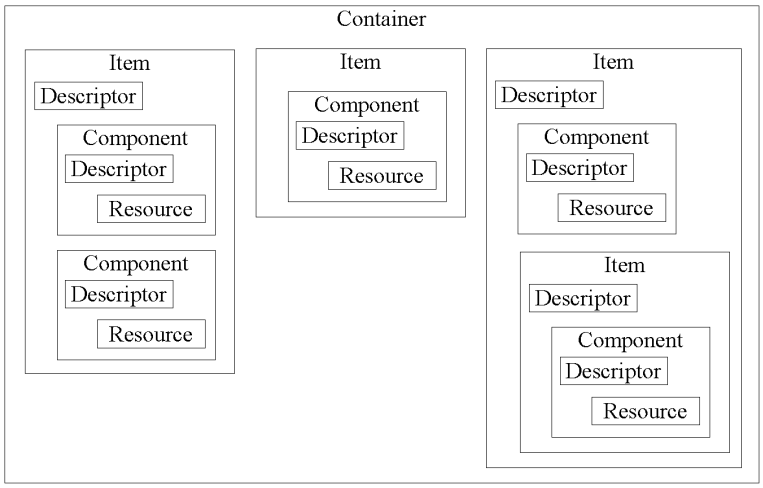

Figure 12: Some DID elements and their relationships

is a binding of a resource to its relevant descriptors. A statement is a literal textual value that contains information, for example descriptions, revision tracking, identifying information. An item is a grouping of sub-items and/or components that are bound to relevant descriptors. An item can contain choices, selections or can be conditional. A choice is a set of related selections. The selections describe each possible decision and will affect certain conditions when activated. A condition evaluates whether or not to enclose an optional element. Depending on the specific selection, other parts of the Digital Item can be shown. It could for example contain an image in different qualities and formats. A container is just a grouping of items and/or containers. An anchor binds descriptors to a fragment. An annotation can be added to an element and contains information about that element without changing it. An XML example is given in Figure 13.

.

Digital Item Identification (DII [52]) specifies unique identifiers for (parts of) Digital Items, related intellectual property or description schemes and how to to link Digital Items with related information such as meta data. A unique identifier for a Digital Item can be put as statement in an associated descriptor and will have the form of a URI. Digital Item Types also have different URIs, which are put in a statement of a Digital Item. Users may use different schemes to describe their specific content. Therefore, the XML mechanism of name spaces is used. MPEG-21 does not try to specify new identification schemes if good ones already exist.

Rights Expression Language (REL [46, 76]) defines a REL to use in MPEG-21. The winner in the Call for Proposals for a Rights Data Dictionary and Rights Description Language was XrML (eXtensible rights Markup Language [15]) (created by ContentGuard [5]). Its successor is MPEG-21 REL [46] and is an open, XML-based REL which offers a high degree of expressiveness, flexibility, extensibility and interoperability. MPEG-21 is still competing with ODRL.

Products from a range of companies have adopted MPEG-21 or XrML in their products. Some of them are Microsoft, Sony and DMDsecure. Many companies, such as IMB, HP, Cisco and Verisign participate in the MPEG-21 Rights Language Technical Committee ("RLTC") to create standards based on XrML. Some industrial partners are Microsoft, Adobe Systems, DMDsecure and Xerox Corp.

To describe the concepts and the interrelationships w.r.t. rights, XrML has developed a data model that is also used by MPEG-21 REL. This is also the base of expressing rights with the help of XML files.

A central entity in MPEG-21 REL is the Grant, which consists of four other central entities and their relationships: Principal, Right, Resource and Condition. This is illustrated in Figure

```
<?xml version="1. 0" encoding="UTF -8"?>
<DIDL
xmlns="urn:mpeg:mpeg21:2002:01 -DIDL-NS"
xmlns:xsi ="http://www.w3.org/2001/XMLSchema -instance"
xsi:schemaLocation ="urn:mpeg:mpeg21:2002:01 -DIDL-NS E:\Users\RVdW \Temp\DIDL.xsd">
    <Container >
        <Descriptor >
            <Statement mimeType ="text/plain">
                This information package was developed by University Records Unlimited
            </Statement >
        </Descriptor >
        <Item>
            <Descriptor >
                <Statement mimeType ="text/plain">
                    Copyright owner: University Records Unlimited
                    Permission: Read  Only
                </Statement >
            </Descriptor >
            <Descriptor >
                <Component>
                    <Resource ref="http://www.uruweb.org/logos/uow.jpg  " mimeType ="image/jpeg "/>
                </Component>
            </Descriptor >
            <Choice choice_id ="INFO_PICKER ">
                <Descriptor >
                    <Statement mimeType ="text/plain">
                        Choose the information you want to receive:
                    </Statement >
                </Descriptor >
                <Selection select_id ="VIDEO">
                    <Descriptor >
                        <Reference target="#VIDEO_TITLE "/>
                    </Descriptor >
                </Selection >
                <Selection select_id ="TEXT ">
                    <Descriptor >
                        <Reference target="#TEXT_TITLE "/>
                    </Descriptor >
                </Selection >
            </Choice >
            <Item>
                <Condition require="VIDEO"/>
                <Descriptor id="VIDEO_TITLE ">
                    <Statement mimeType ="text/plain">
                        Research over view video
                    </Statement >
                </Descriptor >
                <Component>
                    <Condition require ="VIDEO"/>
                    <Resource ref="http://www.uruweb.org/video/research.mp4  " mimeType ='video/mp4 "/>
                </Component>
            </Item>
            <Item>
                <Condition require="TEXT "/>
                <Descriptor id="TEXT_TITLE ">
                    <Statement mimeType ="text/plain">
                        Lecture notes
                    </Statement >
                </Descriptor >
                <Component>
                    <Condition require ="TEXT "/>
                    <Resource ref="http://www.uruweb.org/text/lecturenotes.txt  " mimeType="text/plain"/>
                </Component>
            </Item>
        </Item>
    </Container >
</DIDL >
```

Figure 13: Example of a MPEG-21 container representation in XML

14.

A *resource* can be a digital work (i.e. content), a service (such as an e-mail service), or a simple piece of information (such as a name or an e-mail address). The resource is also called the 'object' of the grant.

A *right* is an action or a class of actions that can be performed on a resource. It can be seen as the 'verb' of the grant. Some commonly used rights relating to other rights are issue, revoke

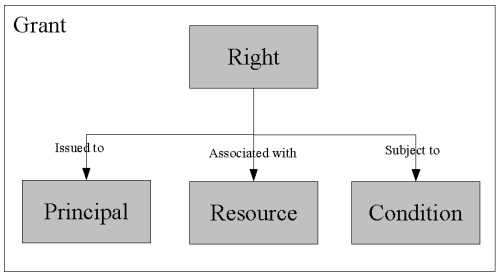

Figure 14: MPEG-21 entity "Grant"

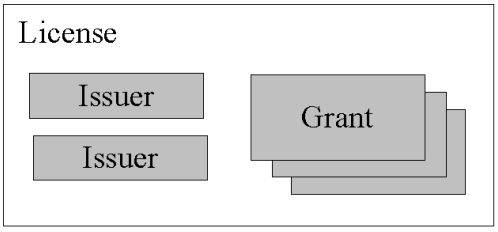

Figure 15: MPEG-21 entity "License"

and obtain. Other rights are content-type specific. Examples are print and play rights.

The *principal* is the party to whom the rights are granted. The identification is done by information unique to that principal. The Principal must be able to prove his identity using an authentication mechanism. Both public/private key mechanism and credentials are supported. Other identification technologies can still be added later. Each principal identifies exactly one party. A set of parties is thus not a principal. The principal is the 'subject' of the grant.

A *condition* specifies the terms, conditions and obligations under which the principal can exercise the rights on the resource. A simple condition is a time interval determining when the rights can be exercised. Other conditions are the existence of a watermark, destination and renderer. A set of conditions can be put in conjunction, so that all conditions have to be met first.

The final central entity in XrML is the *license* and is illustrated in Figure 15. A license in MPEG-21 REL is conceptually the issuance of grants by their issuing parties. This matches our definition of a license given in Section 1.4. A license consist first of all of a set of grants which give some principals a set of rights to some resources under certain conditions. Secondly, the license contains the identification of the issuers of the license. An issuer can digitally sign the license. Syntactically, multiple issuers may sign it. Finally, some additional information can be added, such as description or validity date. Notice the absence of an identification of the principal: this is already done in the grants.

As an example, a minimal license is shown in Figure 16.

The *structure* and organization of MPEG-21 REL is shown in Figure 17. MPEG-21 has a Core Schema at the basis. Here, basic concepts of MPEG-21 REL are defined. These include license, grant, resource, principal, right and condition. The last four of these concepts are defined in an abstract way and can be extended in the extensions of the core schema to become useful for specific content formats. Some rights defined in the Core Schema are revoke and issue.

A Standard Extension Schema contains general and broadly applicable definitions of concepts that are not at the heart of the MPEG-21 semantics. It supports the notion of external service required to exercise a right (e.g. usage tracking), payment conditions and methods, and time conditions.

The Content Schema defines rights management concepts specifically related to digital works such as movies and book, such as the right to lend, delete, copy, print, export or edit a digital work. An example of a condition is the presence of a watermark.

Other parties may define their own Extension Schemas.

The language semantics enables the authenticity of any semantically significant construct using open and standard cryptographic mechanisms such as digital signatures. The same is true for confidentiality, enabling encryption of any semantically significant construct.

Pattern matching is provided and offers a way to specify a set of principals, rights, resources and conditions.

Rights Data Dictionary (RDD [76]) is closely linked to the Rights Expression Language part and defines the exact semantic in the REL. It defines a methodology to define and catalogue terms in the context of rights management. RDD forms the basis to express rights and permissions

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

                     <dsig:Modulus>Fa7wo6NYfmvGqy4ACSWcNmuQfbejSZx
                     7aCibIgkYswUeTCrmS0h27GJrA15SS7TYZzSfaS0xR91Z
                     dUEF0ThO4w==</dsig:Modulus>
                  <dsig:Exponent>AQABAA==</dsig:Exponent>
               </dsig:RSAKeyValue>
            </dsig:KeyValue>
         </info>
      </keyHolder>
      <possessProperty />
      <sx:commonName>Alice Richardson</sx:commonName>
    </grant>
  </license>
```

Figure 16: Example of a minimal MPEG-21 REL License in XML

as described in REL. The RDD standard offers a basis set of terms and offers a methodology to make a dictionary. This basis set can be extended according to the needs of the specific content or context, resulting in different schemes. RDD offers a mapping and transformation of terms between different schemes, independent of the name of these terms.

Digital Item Adaptation (DIA [75]) specifies tools for the adaptation of Digital Items. One of the targets of MPEG-21 is interoperable, transparent access to (distributed) advanced multimedia content. Therefore, content may need to be adapted such that devices or networks with specific capabilities are able to access or deliver it. This is specified in this part and illustrated at high level in Figure 18. A Digital Item is converted to an adapted Digital Item by a Resource Adaptation Engine, who adapts the DID, resource and the description. For being able to convert content, it is necessary to specify the description of content format and usage environments, which is part of the DIA specification.

.

Digital Item Processing (DIP): DID is a static declaration of only information related to the item (structure, resources and meta data): no information is included about implied processing. The user has thus nothing that indicates how the content should be processed (e.g. where to download the Digital Item or the corresponding rights, presenting individual resources, ...). This kind of information is specified by DIP. DIP defines mechanisms for standardized and interoperable processing of information in the Digital Item.

Future work. The task of MPEG-21 has not yet been finished. Future work or work in progress includes: definition of an interoperable framework for Intellectual Property Management and Protection in a reliable way across networks and on devices, evaluation of methods for persistent association technologies, definition of MPEG-21 reference software in general and test bed for MPEG-21 resource delivery in general, definition of an MPEG-21 file format, exploration

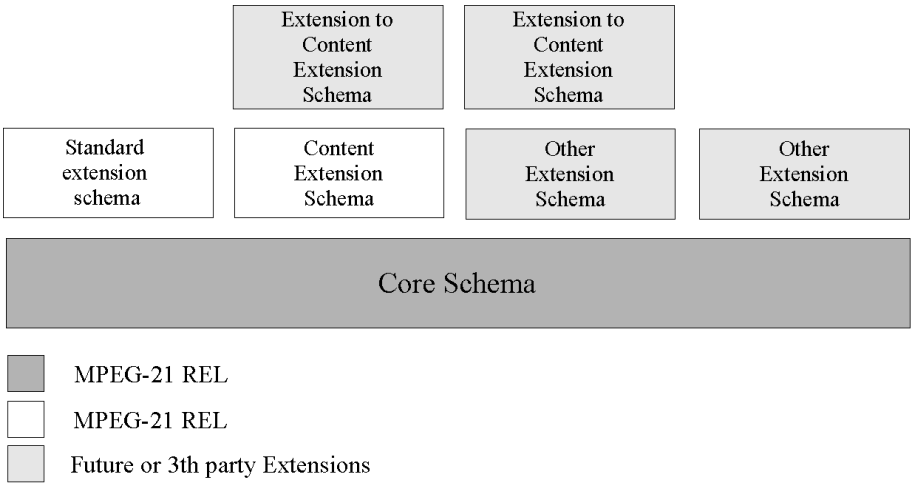

Figure 17: MPEG-21 REL structure and organization

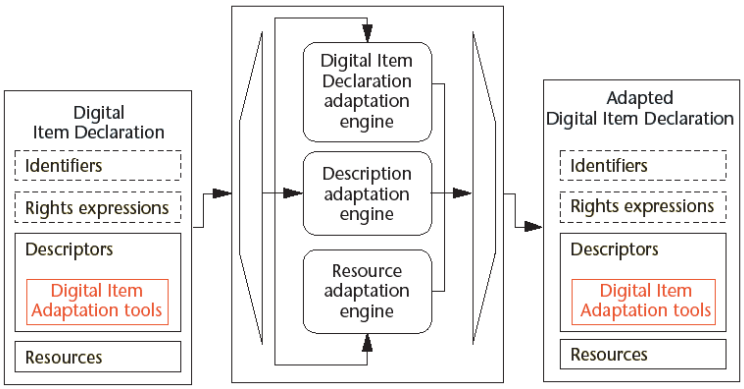

Figure 18: Digital Item Adaptation architecture

of requirements and technology for highly scalable video and audio coding and finally, specification of MPEG-21 event reporting mechanism which will allow for example tracking of content usage.

4.2 Coral Consortium and Marlin JDA

The Coral Consortium Corporation [6] is a cross industry consortium containing content providers, service providers and consumer electronics manufacturers. Its goal is to establish interoperability between different existing DRM technologies by developing and standardizing a set of specifications. The resulting interoperability layer supports the coexistence of multiple different DRM technologies and transparently offers a uniform experience to the users of it, not having to know which DRM technology they are using. Both interoperability of secure distribution over the Internet and home networks are supported. By using Coral gateways, existing devices will not get obsolete. On 1 March 2005, the Coral Consortium v1.0 specifications were released. Its founding members are HP, InterTrust, Philips, Matsushita (Panasonic), Samsung, Sony and Twentieth Century Fox. Now, the consortium counts nearly 30 members, under which the four major music labels: EMI, Universal, Warner and Sony-BMG.

In January 2005, a subpart of the Coral Consortium members started with the Marlin Joint Development Association (Marlin JDA) which builds upon the results of Coral. The goal of the Marlin JDA is to provide standard specification for content management and protection for the

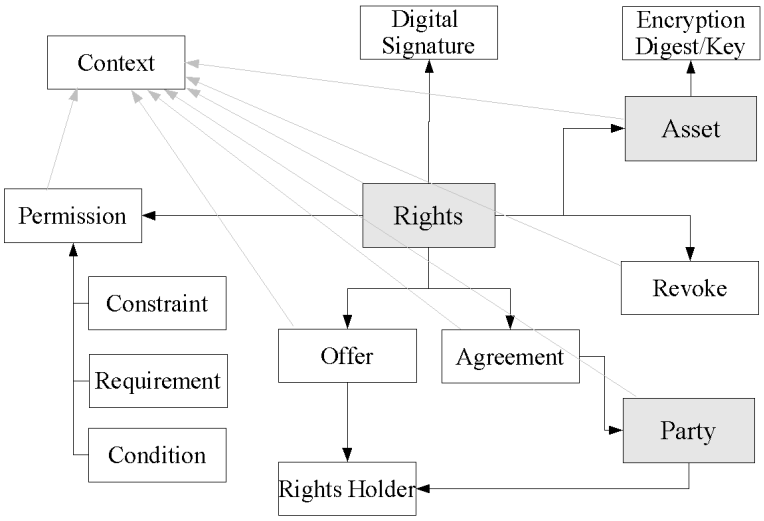

Figure 19: ODRL founding model

consumers electronics industry. This specification will allow consumer electronics companies to build DRM clients into their consumer devices. Popular distribution models, such as using the Internet or broadcast, will be supported. Convergence across consumer Internet, broadcast and mobile devices and services will be facilitated. The Marlin v1.0 specification is expected in the summer of 2005 and will be completely compatible with the CORAL specifications. Another goal of MARLIN is an enhanced user friendliness. The founding members are Sony, Samsung, Royal Philips Electronics, Panasonic, and InterTrust.

4.3 ODRL

ODRL (Open Digital Rights Language) [32] is an open, XML-based REL that offers a high degree of expressiveness, flexibility, extensibility and interoperability. ODRL offers a core set of semantics. Additional semantics can be layered on top of ODRL. ODRL was proposed by IPR Systems [25]. The last version (V.1.1) was announced in September 19, 2002. Some of the ODRL supporters are OMA (see Section 3.6), the W3C [44], AegisDRM (see Section 3.7.4), and Nokia [31]. A point of focus in ODRL is the reuse of of digital material both on the Internet as in physical media. Content can be reused, combined and extended at the infinitum, while rights are honored. Founding model The founding model describes at a conceptual level how rights specification is seen by ODRL. ODRL is build upon this viewpoint, and thus also the XML definitions. As illustrated in Figure 19, the three basis entities in the ODRL foundation model are 'assets', 'rights' and 'party'.

Assets are content, physical or digital, on which rights can be applied. These must have a unique identification. An asset can consist of many different subparts (also assets) and can exist in different formats. *Parties* can be both content consumers and content producers (Rights Holders) and can be humans, organizations or roles. *Right Holders* are usually parties that have played some role in the creation, production or distribution of the asset and can assert some form of ownership over the asset and/or its Permissions and correspond to our notion of producers. Rights Holders can also receive royalties.

A general description of Assets and Parties is outside the scope of ODRL. ODRL did however determine that these must be referenced by a URI. If the subparts of Assets are uniquely identifiable, then rights can be applied on these subparts.

Rights on Assets can be given to Parties. Rights include *Permissions*, which are actions that can be performed on assets (play, print, ...). These Permissions can contain Constraints, Requirements and Conditions. *Constraints* are limitations on the Permissions (e.g. play maximum 5 times). *Requirements* are the preconditions that must be fulfilled before exercising the Permission.

```
<rights>
    <context>.
        <uid> ... </uid>
    </context>
    <offer>
        <asset> ... </asset>
        <permission>
            <permission-type>
                <requirement> ... </requirement>
                <constraint> ... </constraint>
            </permission-type>
            <condition> ... </condition>
        </permission>
        <party>
            <context> ... </context>
            <rightsholder> ... </rightsholder>
        </party>
    </offer>
    <agreement>
        <context> ... </context>
        <party> ... </party>
        <permission> ... </permission>
        <asset> ... </asset>
    </agreement>
</rights>
```

Figure 20: ODRL foundation model expressed in XML

(e.g. pay $ 5 for being able to play the video). *Conditions* specify exceptions; if one becomes true, it expires the Permissions (e.g. subscription to the DRM service expires). Re-negotiations may be needed.

With these three entities, Offers and Agreements can be specified. An *Offer* is a proposal from the Rights Holder to the consumer (other Party). An offer can be converted in an *Agreement*, which is a specific concretisation of an Offer. ODRL can also express the revocation of Offers and Agreements.

A *Context* can be linked with any entity and can describe extra information about that entity or relationships between different entities. The Context can be used for different purposes. When it is linked with a Parties entity, it may contain the name, the role or a unique identifier of that entity. An Agreement can have the time and location of the transaction as its context. A rights entity can have a context with a unique identifier for that rights expression.

ODRL can also describe how the content (Assets) is cryptographically manipulated. This is made possible by the *Digital Signature* and *Encryption Digest/Key* items.

An example of an XML representation based on the ODRL foundation model can be seen in Figure 20.

5 Highlights

Systems that provide DRM are highly complex and extensive [71]: DRM technologies must support a diversity of devices, users, platforms, and media, and a wide variety of system requirements concerning security, flexibility, and manageability. This complexity and extensiveness poses three major challenges to DRM development: fragmentation of individual solutions, limited reuse and interoperability between DRM systems, and lack of a domain-specific structure that supports and guides the design and implementation of DRM systems and their applications.

The first challenge relates to the fact that state-of-the-art DRM technologies are often ad-hoc, which leads to fragmented solutions and makes it very difficult to complete the global DRM

picture. A complete DRM solution should provide a single platform that can support every aspect of digital rights management [64].

The second challenge, limited interoperability, is partly caused by in-house developed solutions that are incompatible with similar systems produced by other parties. Although various research groups have produced "vertically integrated" designs in which their particular set of components are specifically conceived to collaborate, their solutions are unable to interoperate with components from other groups. Given the complexity and extensiveness of DRM, interoperability between specific DRM technologies is crucial to integrate existing solutions [64].

The third challenge, lack of guiding software structure, is typical for complex software systems in evolution, and providing such a context is often a sign of growing maturity of the application domain [68]. In order to evolve towards a complete set of interoperable DRM solutions, we need a well-defined software architecture that identifies the major services and defines how they interact [64, 49].

The challenges of integrating independent system components are well-recognized and are being addressed in other application domains than DRM, such as network protocol stacks, web services, or graphical user interfaces [74]. The Internet architecture, for instance, convincingly demonstrates how a properly chosen set of guiding principles can shape the evolution of a complex system across vast changes in technology, scale, and usage [55]. The power of the Internet lies not so much in the elegance or efficiency of its individual components, but in the overall ability to encompass tremendous growth in scale and diversity as usage and technology continue to evolve.

This report has proposed a first step towards a DRM software architecture that supports DRM developers in producing complete and interoperable systems. The architecture is approached both from a functional and a security perspective. The proposed architecture is validated by matching it to state-of-the-art DRM technologies.

A next step towards a software architecture for DRM has been presented in [70]. In this paper, a layered architectural style has been proposed. In this case, the functional perspective zooms in on the top layers, closest to the applications using the architecture. The security perspective focuses on the bottom layers, which offer cryptographic primitives to enforce digital rights. In other words, the cryptographic primitives at the bottom layers lay the foundation for the upper layers to build upon. The paper studies whether or not the main DRM services (i.e. content, license, access, tracking, payment, import, and identification) are supported by state-of-the-art DRM solutions. It shows that some services are provided almost uniformly by all technologies, while others are only offered sporadically. The Content and License Services are almost always implemented, which seems nothing but normal for such key services. Services for accessing, tracking, paying and importing are provided in approximately 50% of the cases, while the Identification Service is not implemented by any of the studied DRM techniques, at least not to our knowledge.

When relating these results with the three main DRM challenges (completeness, interoperability, and software architecture support), we can draw the following conclusions. First of all, the fact that so many different DRM technologies implement the same or similar services confirms our claim that we need an architecture that promotes reuse and interoperation of individual service components.

Secondly, the services with the highest benefit from reuse and interoperation are the Content and License Service. All DRM technologies that need these services would benefit from a reusable implementation.

Thirdly, since different DRM technologies implement different sets of services, trying to standardize 'the' DRM technology seems less efficient than focusing on particular services these technologies are composed of. This brings us back to the analogy with the Internet architecture, which clearly identifies service responsibilities and a common platform that can support a wide variety of networking services. The key message is that this architecture proves that a complete solution can be offered by a single platform if it allows reusable services to be plugged in, without trying to provide a single overall standard implementation. Until today, many different companies and organizations extend the TCP/IP architecture with protocols for quality-of-service, wireless communication, routing, media streaming, or security. If we are to provide complete DRM solutions, following the Internet approach seems to be a good idea.

6 Conclusion

Starting from the identification of major DRM components, this report has proposed a first step towards a generic DRM architecture and subsequently mapped six DRM systems onto it, thereby enabling a critical evaluation. The underlying model consists of a distributed view and perspectives from consumer, producer, as well as the publisher. This model has proven to be a useful framework to inventory, analyze, and discuss research in this field, and to set the agenda for the future. Inspection of this framework shows the use of Content and License Services to be well represented in the DRM technologies described. The uniform application of services like access, import, tracking, and payment services is less developed, whereas identification is absent. Special attention should go to three main DRM challenges: completeness, interoperability, and software architecture support. Therefore, if DRM is not to end as the umpteenth flash in the data protection pan, it may be high time to put software architecture design at the top of its research agenda.

Acknowledgements

Part of the research in this report was sponsored by IBBT, the Interdisciplinary institute for BroadBand Technology, and conducted in the context of the E-paper project. The authors are very grateful to Koen Buyens for his valuable comments on the report and for proof reading the text.

References

[1] Adobe and digital content for e-commerce. http://www.adobe.com/products/acrobat/webbuy/.

[2] Aegisdrm. http://www.aegisdrm.com.

[3] Beep science. http://www.beepscience.com.

[4] Cloakware. http://www.cloakware.com.

[5] Contentguard. http://www.contentguard.com.

[6] Coral consortium corporation. http://www.coral-interop.org.

[7] Coremedia. http://www.coremedia.com/coremedia.aspx/solutions/inhalte-verkaufen/drm-for-mobile-services/language=en/id=31080/drm-for-mobile-services.html.

[8] Digital rights management and privacy. http://www.epic.org/privacy/drm/default.html.

[9] Dmdfusion 5. http://www.dmdsecure.com/DMDfusionV3.htm.

[10] Dmdlicenser. http://www.dmdsecure.com/DMDlicenser.htm.

[11] Dmdmobile. http://www.dmdsecure.com/DMDmobile.htm.

[12] Dmdsecure. http://www.dmdsecure.com.

[13] Dmdsecure toolkits. http://www.dmdsecure.com/toolkits.htm.

[14] electronic identity card. http://eid.belgium.be/en/navigation/12000/index.html.

[15] extensible rights markup language. http://www.xrml.org.

[16] Fraunhofer ipsi, department merit. http://www.ipsi.fraunhofer.de/merit.

[17] Freeme documentation. http://home.wanadoo.nl/lc.staak/freeme.htm.

[18] Harmony. http://www.realnetworks.com/company/press/releases/2004/harmony.html.

[19] Helix community. http://www.helixcommunity.org.

[20] Helix drm. http://www.realnetworks.com/products/drm.

[21] Hymn project. http://hymn-project.org.

[22] Ibm electronic media management system. http://www-306.ibm.com/software/data/emms/.

[23] International telecommunication union. http://www.itu.int.

[24] Intertrust technologies corp. http://www.intertrust.com.

[25] Ipr systems. http://www.iprsystems.com.

[26] itunes. http://www.apple.com/itunes.

[27] John lech johansens blog. http://www.nanocrew.net/software.

[28] Lightweight digital rights management. http://www.lwdrm.com.

[29] Microsoft windows media rights management. http://www.microsoft.com/windows/windowsmedia/drm.

[30] The mpeg home page. http://chiariglione.org/mpeg.

[31] Nokia. http://www.nokia.com.

[32] The open digital rights language (odrl) initiative. http://odrl.net.

[33] Open ebook forum. http://www.openebook.org.

[34] Open mobile alliance. http://www.openmibilealliance.org.

[35] Publishing requirement for industry standard metadata. http://www.prismstandard.org.

[36] Sandisk. http://www.sandisk.com.

[37] Sealedmedia. http://www.sealedmedia.com.

[38] Secure digital music initiative. http://www.sdmi.org.

[39] Semantic designs: Source code obfuscators. http://www.semdesigns.com/Products/Obfuscators/.

[40] Trusted computing group. http://www.trustedcomputinggroup.org.

[41] Unified modelling language. http://udml.org.

[42] Verance. http://www.verance.com.

[43] Vlc media player. http://www.videolan.org/vlc.

[44] The world wide web consortium. http://www.w3c.org.

[45] Zelix klassmaster. http://www.zelix.com/klassmaster/.

[46] The mpeg-21 rights expression language - a white paper., 2003.

[47] The international doi foundation. the digital object identifier (doi), June 2005. http://www.doi.org.

[48] R. Anderson. Cryptography and competition policy: issues with 'trusted computing'. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 3–10, New York, NY, USA, 2003. ACM Press.

[49] L. Bass, P. Clements, and R. Kazman. *Software architecture in practice*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1998.

[50] E. Becker, W. Buhse, D. Günnewig, and N. Rump, editors. *Digital Rights Management - Technological, Economic, Legal and Political Aspects*, volume 2770 of *Lecture Notes in Computer Science*. Springer, 2003.

[51] I. Burnett, R. V. de Walle, K. Hill, J. Bormans, and F. Pereira. Mpeg-21: Goals and achievements. *IEEE MultiMedia*, 10.(4):60–70, 2003.

[52] I. S. Burnett, S. J. Davis, and G. Drury. Mpeg-21 digital item declaration and identification: Principles and compression. *Multimedia, IEEE Transactions on*, 7(3):400–407, 2005.

[53] S. Chow, P. A. Eisen, H. Johnson, and P. C. van Oorschot. A white-box des implementation for drm applications. In *Digital Rights Management Workshop*, pages 1–15, 2002.

[54] S. Chow, P. A. Eisen, H. Johnson, and P. C. van Oorschot. White-box cryptography and an aes implementation. In *SAC '02: Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography*, pages 250–270, London, UK, 2003. Springer-Verlag.

[55] D. D. Clark. The design philosophy of the DARPA internet protocols. In *SIGCOMM*, pages 106–114, Stanford, CA, Aug. 1988. ACM.

[56] K. Coyle. Rights expression languages. technical report. Technical report, A report for the Library of Congress, Feb. 2004. http://www.loc.gov.standards.

[57] S. A. Craver, M. Wu, B. Liu, A. Stubblefield, B. Swartzlander, D. S. Wallach, D. Dean, and E. W. Felten. Reading between the lines: Lessons from the SDMI challenge. In *Proc. 10th USENIX Security Symp.*, 13–17 2001.

[58] J. Garland and R. Anthony. *Large Scale Software Architecture*. Wiley and Sons, 2003.

[59] R. Grimm and P. Aichroth. Privacy protection for signed media files: a separation-of-duty approach to the lightweight drm (lwdrm) system. In *MM&Sec '04: Proceedings of the 2004 workshop on Multimedia and security*, pages 93–99, New York, NY, USA, 2004. ACM Press.

[60] S. Guth. Rights expression languages. In *Digital Rights Management*, volume 2770 of LNCS, pages 101–112, 2003.

[61] F. Hartung. Mobile DRM. In E. Becker, W. Buhse, D. Günnewig, and N. Rump, editors, *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, volume 2770 of *LNCS*, pages 138–149. Springer, Nov. 2003.

[62] F. Hartung and F. Ramme. Digital rights management and watermarking of multimedia content for m-commerce applications. In *IEEE Communications*, pages 78–84, 2000.

[63] B. Horne, L. R. Matheson, C. Sheehan, and R. E. Tarjan. Dynamic self-checking techniques for improved tamper resistance. In *Digital Rights Management Workshop*, pages 141–159, 2001.

[64] P. A. Jamkhedkar and G. L. Heileman. Drm as a layered system. In *DRM '04: Proceedings of the 4th ACM workshop on Digital rights management*, pages 11–21, New York, NY, USA, 2004. ACM Press.

[65] W. Ku and C.-H. Chi. Survey on the technological aspects of digital rights management. In *ISC*, pages 391–403, 2004.

[66] H. E. Link and W. D. Neumann. Clarifying obfuscation: Improving the security of white-box encoding. Cryptology ePrint Archive, Report 2004/025, 2004. `http://eprint.iacr.org/`.

[67] W. Mao. *Modern Cryptography: Theory and Practice*. Prentice Hall Professional Technical Reference, 2003.

[68] S. Michiels. *Component Framework Technology for Adaptable and Manageable Protocol Stacks.* PhD thesis, K.U.Leuven, Dept. of Computer Science, Leuven, Belgium, Nov. 2003.

[69] S. Michiels, W. Joosen, and K. Verslype. A study of drm aspects for content distribution. Technical report, Interdisciplinary institute for BroadBand Technology (IBBT), Sept. 2005. Internal report.

[70] S. Michiels, K. Verslype, W. Joosen, and B. D. Decker. Towards a software architecture for drm. In *In Proceedings of 5th ACM Workshop on DRM (DRM2005)*, Nov. 2005.

[71] D. K. Mulligan. Introduction. *Commun. ACM*, 46(4):30–33, 2003.

[72] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. In *IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection*, pages 482–494, Washington, DC, USA, May 1998. IEEE Computer Society.

[73] B. Schneier. *Applied Cryptography - Protocols, Algorithms and Source Code in C.* Wiley, New York, NY, USA, second edition, 1997.

[74] C. Szyperski. *Component Software: Beyond Object-Oriented Programming.* ACM Press/Addison-Wesley Longman Publishing Co., Inc., New York, NY, USA, 1998.

[75] A. Vetro. MPEG-21 digital item adaptation: Enabling universal multimedia access. *IEEE MULTIMEDIA*, 11(1):84–87, Jan./Mar. 2004.

[76] X. Wang, T. Demartini, B. Wragg, M. Paramasivam, and C. Barlas. The mpeg-21 rights expression language and rights data dictionary. *Multimedia, IEEE Transactions on*, 7(3):408–417, June 2005.

[77] G. Wroblewski. *General Method of Program Code Obfuscation.* PhD thesis, Wroclaw University of Technology, Institute of Engineering Cybernetics, 2002.