# DIGITAL OBJECT RIGHTS MANAGEMENT
## *Interoperable Client-side DRM Middleware*

Carlos Serrão, Miguel Dias
*Adetti, ISCTE/DCTI, Ed. ISCTE, Av. Das Forças Armadas, 1600-082 Lisboa, Portugal*

Jaime Delgado
*Universitat Pompeu Fabra, DMAG, Pg. Circumvallació 8, E-08003 Barcelona, Spain*

Keywords: DRM, middleware, security, network protocols, web-services, IPR, cryptography.

Abstract: In a more and more interconnected world where the available bandwidths are increasing at a pace hard to imagine some time ago, multimedia e-content distribution over digital networks has become one of the biggest available services online. Powered not only by the network high availability but also by the emergence of new compression techniques and digital content consumer device, digital content is gaining momentum. However the same factors that power this emergence are also causing some problems, specially related with the digital content IPR management and protection. These problems are being handled employing DRM - Digital Rights Management technology which lack interoperability. This paper presents and discusses a solution that provides interoperability to DRM-protected content through the employment of a client-side DRM middleware layer. This middleware layer sits at the client-side of a broader DRM system (called DoRM) providing the necessary mechanisms to achieve interoperability between the different digital content rendering applications that the users possesses.

## 1 INTRODUCTION

Digital open networks have changed many aspects of our current lives – technologically, economically and socially. In particular it has mostly changed the way (digital) information is exchanged. This digital network has changed the way multimedia digital content is distributed and obtained (Duhl et al, 2003). The modern multimedia distribution chain is entirely digital: from content author, content producer, content provider, service provider, network provider and final consumer. However, the distribution of digital multimedia content over digital open networks has some serious problems. One of the most common and important problems refers to the content piracy or copyright circumvention that affects directly the players that are part of the multimedia distribution chain.

This is a strong barrier to those who use this distribution channel – especially to authors and content owners. To overcome some of the problems created by unauthorized digital content distribution and usage, some technologic measures were put in place. Common and simple technologic procedures include content encryption, scrambling and watermarking which are parts of a broader definition of content protection (C/P). More complex measures may include the definition of business rules and content usage rules, managed by an infrastructure called Digital Rights Management (DRM). Efficient IPR protection measures arise from the conjunct usage of effective C/P technologies and DRM platforms. DRM is a set of technological components which may be capable of performing different functions, such as content protection, content management, authentication, authorization and accounting, payment and rights management (Duhl et al, 2003). DRM can uphold specific conditions to specific groups of users, devices or domains in terms of digital content usage. This is something that goes inline with the content owners and authors expectations, on what concerns piracy prevention, but sometimes it creates some obtrusiveness on the user side. One of the most important requirements for users is that DRM technology doesn't alter its user experience with the



Serrão C., Dias M. and Delgado J. (2006).
DIGITAL OBJECT RIGHTS MANAGEMENT - Interoperable Client-side DRM Middleware.
In *Proceedings of the International Conference on Security and Cryptography*, pages 229-236
DOI: 10.5220/0002101602290236

multimedia digital content when compared with non-DRM protected content (Serrão et al, 2005). One of the main reasons for this user side obtrusiveness is mostly due to the fact that most of the user requirements on what concerns the digital content experience are disregarded and only content owners and authors are considered in the design of the actual DRM-solutions (Serrão et al, 2005). There is an important dichotomy between the content owners and final user's interests. These results, in most of the cases, that the user experience with DRM copyrighted and protected digital content is not the best and the most obvious alternative for users is to find alternative ways to enhance such experience – obtaining illegal pirated content on P2P networks, for instance. An important source for this user's obtrusiveness results from the fact that most of the existing DRM solutions are proprietary and vertical, following a 1-to-1 strategy on what concerns DRM. This means that they have their own protected-content formats, specific protection tools and specific rights expression mechanisms. In practice, a user that buys a music track from an online digital content store that uses a specific DRM protection scheme is unable (most of the times) to render that content on its preferred music player application, and is limited to use whatever music player application which implements the DRM protection scheme that is the result of a business partnership between the digital content provider and the DRM technology provider. Sometimes, this 1-to-1 approach causes more problems than it solves.

The following section of the paper will start by presenting a short description of an open, secure and distributed DRM platform - called Digital object Rights Management (DoRM). This paper will focus mainly on how the platform generates and manages digital content usage rights (Serrão, 2004). Then the description of a client-side middleware layer that manages the rights at the end user side, allowing interoperability between different content rendering applications is also presented.

## 2 THE DORM SOLUTION

Interoperability is a problem that can be addressed from different sides. In what concerns digital multimedia content interoperability it can be seen in terms of content format interoperability, content protection methods interoperability, rights expression interoperability and many others. In what concerns the object of this paper, interoperability refers to rights management interoperability, although some of the concepts may also be applicable to other types of interoperability such as content protection, for instance. The approach followed to achieve this rights management interoperability will consist in two different components:

1. At the server-side, at the DoRM platform, using a set of pre-defined rights templates that will enable digital multimedia content stores to align their own business model with the DRM enforcing layer (Serrão et al, 2005). These templates are defined by the content provider using a specific Rights Expression Language (REL) that defines the business rules;
2. At the client-side, the establishment of a DRM middleware layer, between the content rendering applications and the rights definition, allowing any content rendering application to ask to this middleware layer, the permission for conducting an operation over the content.

Both approaches are being developed as part of a DRM solution called Digital Rights object Management. DoRM is a distributed service-oriented DRM platform (Serrão et al, 2003, 2004), independent from the type of content, the content protection and the business model to be used. It can be used with multiple communication protocols and is based on the emerging service-oriented paradigm (SOAP (W3C-SOAP, 2001), WSDL and UDDI) approach (Serrão et al, 2003), called Service Oriented Architecture (SoA). DoRM (Figure 1) can cover most of the content lifecycle phases: from content authoring, distribution and management of the related rights up to the final user. The communication between each of the single components that compose the DoRM platform is typically performed over an insecure network. This introduces special needs regarding the security of this communication. DoRM defined and uses two different security layers (Figure 1). The first security layer is established at the communication level, which provides the necessary secure and authenticated communication channel that allows components to communicate securely with each other. This first layer uses SSL and X.509 certificates installed on the hosting servers enabling this first authentication and ciphering layer. The second security layer is established at the application level, ensuring the integrity, authentication and non-repudiation mechanisms needed by the different components which provide the DoRM services. This second layer, which corresponds to the exchange of SOAP messages internally between the different DoRM services, and between external actors and

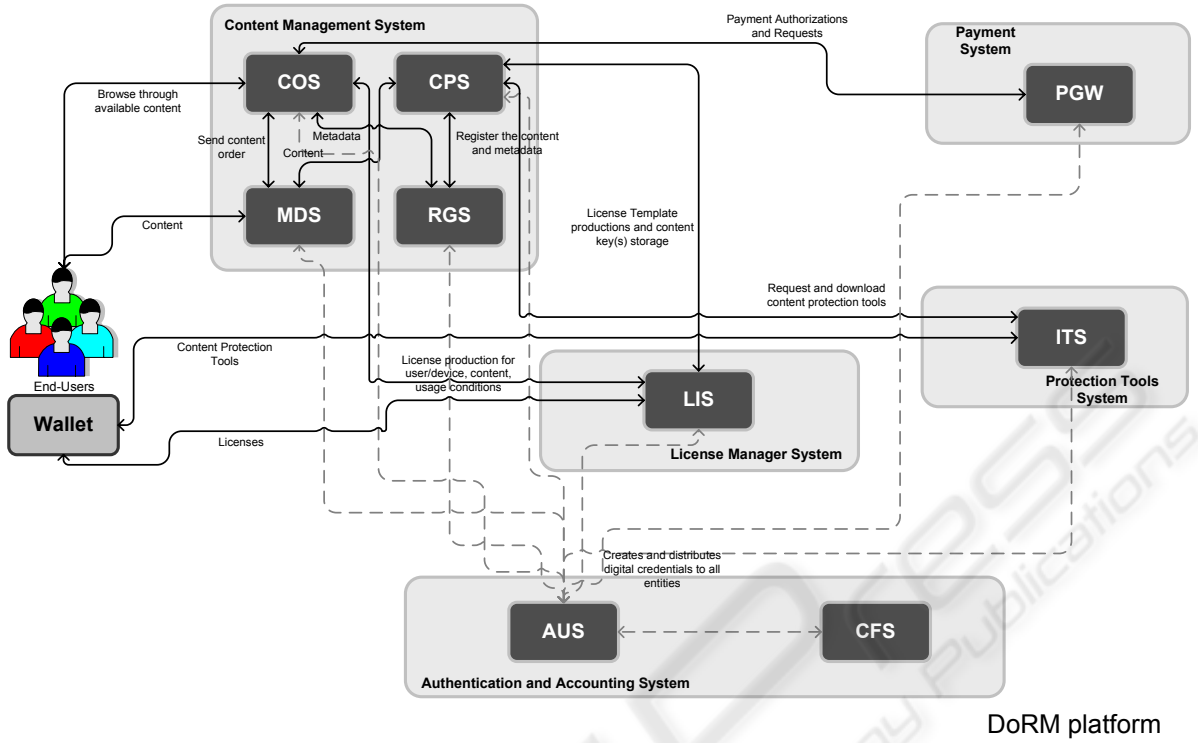


Figure 1: DoRM internal architecture.

those same services, uses the possibilities offered by the W3C XML security-related recommendations: SOAP extensions to use XML Signature. This is part of the new XML security-related architecture (WS-Security).

The authentication between components is provided using a specific developed mapping of X.509 certificates in XML. Each of the DoRM platform services uses this two layered security model. While the underlying layer provides the authenticity of the web-servers (in which web-services are deployed and hosted) and the confidentiality of the communication channel between these servers, the upper layer provides only authentication, integrity and non-repudiation for the different services and actors that interact over the platform, which may be independent of the web-server itself. The other main reason for using SSL is because the DoRM communications are mostly point-to-point and without big scalability requirements. In a technical approach, DoRM is composed by a set of external actors or elements and a set of internal components (Serrão et al, 2003). The internal components are oriented towards the service they supply, and are described in more detail in the following section. These internal components are self-descriptive, in the sense that they expose an open WSDL description of the services and functionalities they provide, and any authenticated component can connect to it and the services and functionalities it provides – DRM services. These internal components communicate with each other using SOAP messages (Serrão et al, 2005). The discovery and identification of services is currently being provided by a central configuration component, but in the future this service will be provided by an UDDI server which will provide information about the services provided by the platform and how to use them.

## 3 THE DORM ARCHITECTURE

The DoRM rights management platform is composed of a set of distributed components implemented using a web-services approach, that exchange standardized messages over open networks (such as the Internet) (Serrão et al, 2003). The DoRM conceptual architecture (presented in Figure 1) defines a scenario capable of handling a multiplicity of different business models for content distribution.

DoRM is composed by a set of external actors or elements and a set of internal components (Serrão et al, 2003). The internal components are oriented

towards the service they supply, and are described in more detail in the next section. These internal components are self-descriptive, in the sense that they expose an open WSDL description of the services they provide. Any authenticated component on the DoRM platform can connect to any of the services it provides – DRM services – to implement its business logic. These internal components communicate with each other using secured SOAP messages (Serrão et al, 2003). The discovery and identification of services is provided by a central configuration component (CFS), an UDDI server that provides information about the services subscribed at platform and information on how to use them. The present DRM state of the art, offers a fragmented landscape of proprietary offerings where the knowledge of how to bridge the different islands resides nowhere. Current DRM technology can be best described as a set of islands that don't have any bridges between them. Thus, most of DRM platforms use vertical approaches to the rights management problem, assuming that along the entire digital content value chain, from the digital rights owner to the final end-user, the same DRM technology will be used. This is an approach that affects both digital rights owners and end-users – in the case of digital rights owners they see their task complicated by the fact that they have to handle with a much higher complexity in their digital contents provision (multiple formats, multiple devices, multiple rights expression and management); in the end-users case, users will have to deal with a multiplicity of different players and devices that are dedicated to render a specific type of DRM-protected content.

DoRM differs from other vertical DRM technologies assuming a horizontal approach. Unlike other DRM solutions, DoRM is completely independent from type of content, the delivery mechanism, the adopted business model and even the methods used to protect the content itself. Another crucial difference between DoRM and other DRM initiatives resides in the fact that all the DRM services are split and distributed over an open network. DoRM was developed having in mind the concept of DRM interoperability, and new functionalities are being added to allow the interoperability with other proprietary DRM systems. The DoRM conceptual architecture is composed of three different types of components: the user (not necessarily the end-users) roles; a set of external entities to the DRM process itself; and the internal DRM entities which provide the DRM functionality.

Around the DoRM platform there are a set of external actors systems. The external actors are: the End-User, the Device Provider, the Content Provider, the Security Tools Providers and the IPR societies. There are also some external systems which may interact with the DoRM platform that are: the Devices, the Content Delivery Systems, the Content Selection system, the Financial System and the Certification System. The Certification System is a very important component on the system and it's responsible for receiving requests for and issuing credentials to entities. These credentials will be used by entities to authenticate themselves to each other, allowing the establishment of secure and authenticated communication channels between them (this is part of the establishment of one of the two DoRM security layers) (Serrão et al, 2003). All the components in the DoRM architecture communicate using the channel security provided by the SSL/TLS protocol (Serrão et al, 2003). This Certification Authority may be internal to DoRM, and therefore entirely managed by some entity, or it may be an external commercial entity, such as Verisign or Thawte (Serrão et al, 2004).

The internal components of the DoRM platform include: Content Management System, License Manager System, Payment System, Content Protection System and Authentication and Accounting System.

The Content Management System is a system component whose role is to assign unique identifiers to content and to register metadata information for that specific content. The service assigns unique identifiers to content using the MPEG-21 (ISO/IEC 21000-3) directives about Digital Item Identification (DII), using a reduced version of the MPEG-21 DII Digital Object Identifiers (Dalziel, 2002). This server component is also responsible for notifying the appropriate content servers that a given content has been requested and that needs to be feed to the final user. This Content Management System handles also the content preparation. It receives raw content from a specified source or sources and encodes it on a specified format, adds metadata and protects it. It is not implemented using the WS approach, although it uses some components that provide such approach. This system component exposes three major functionalities: Content Preparation Server (CPS), the Media Delivery Server (MDS) and the Registration Server (RGS).

The License Manager System is a system component responsible for house-keeping the rules associating a user, the content and his/her corresponding access rights. This component will accept connections from

authenticated content rendering application clients for downloading licenses, which will be applied to the protected content through an appropriate protection tool. By default the licenses are XML formatted using Open Digital Rights Language (ODRL), but other licenses format can also be supported. It exposes a major functionality, the License Server (LIS).

The Payment System is a system component responsible for verifying and validating the payment methods provided by the User to a Commerce Server. It exposes a major functionality, the Payment Gateway (PGW).

The Content Protection System is a system component responsible for registering new protection tools and for receiving authenticated client content rendering application requests for the downloading of a specific protection tool. It is also responsible for making protection tools available to the Content Preparation service to allow the protection of content. The system exposes a major functionality, the IPMP tools server (ITS).

The Authentication and Accounting System is a key-component of the system. It is responsible for authenticating all the internal services and components as well as some external actors to the DRM system. It validates the access rights of all of them working as a single sign-on point, registering and managing components and users on the system. It uses cryptographic XML credentials to authenticate both components and users in order to authenticate the transactions exchanged between them (XML Encryption and XML Signature) (W3C-SOAPSEC, 2001). The system exposes two major functionalities: the Configuration server (CFS) and the Authentication server (AUS).

# 4 DIGITAL CONTENT RENDERING APPLICATIONS DRM INTEROPERABILITY

Interoperability is a key issue in DRM and it is extremely hard to achieve. It is currently one of the hottest topics in the DRM arena, and there are several initiatives that are trying to deal with this – one is the Digital Media Project (DMP) (Chiariglione et al, 2005). This section of the paper discusses only one of the many DRM-interoperability issues. The approach that is described and discussed on this paper is based on the establishment of a rights management interoperability layer at the client-side. The objective of this interoperability layer is to free the content rendering applications from the burden of having to support multiple rights expression languages processing and different authorization modules implementation. This interoperability layer provides such functionalities to all registered content rendering applications. In order to achieve such interoperability level, a DRM middleware layer is defined and built at the client-side. The goal of such interoperability middleware layer is to allow that different content rendering applications can be abstracted from the inner DRM mechanisms that will uphold the content provider user rights at the end-user side. This doesn't reduce however the need for content rendering applications to support the necessary cryptographic mechanisms that will be needed to access to the content. However, this interoperability allows content rendering applications to be implemented and distributed independently of the DRM system used to protect the digital content. There are also some trends in the DRM world that uphold the abstraction of the content rendering applications from the content protection technologies – the Intellectual Property Management and Protection (IPMP) tools from MPEG (ISO/IEC 14496-1, 2003). This DRM middleware layer is capable of managing the access to protected content by different Content Rendering Applications (CRA). Each time a CRA wishes to perform an operation over the content, it contacts the DoRM middleware layer that authorizes or not such operation according to what is specified on the rights expression. This layer allows the coexistence of many DRM-protected content files and DRM-enabled applications on a single client system, presenting a horizontal approach to DRM. Most of the DRM approaches existing nowadays are mostly vertical: examples of this include Microsoft Windows Media Rights Management (Microsoft, 2004) or even Apple iTunes (Lenzi, 2003). While a solution like Microsoft Windows Rights Management is end-to-end Microsoft system-dependent (even at the client-side) relying on Windows Media Player to obtain the licenses and enforce them over the content (Microsoft, 2004), DoRM follows a more horizontal approach in which several content applications can share the access to content, mediated by the DoRM middleware layer. This represents in fact an important interoperability layer at the client-side (Figure 2).

This DRM middleware architecture acts as an intermediary between the DoRM server platform and the different CRA that are installed on the user system. This middleware layer is composed by a set

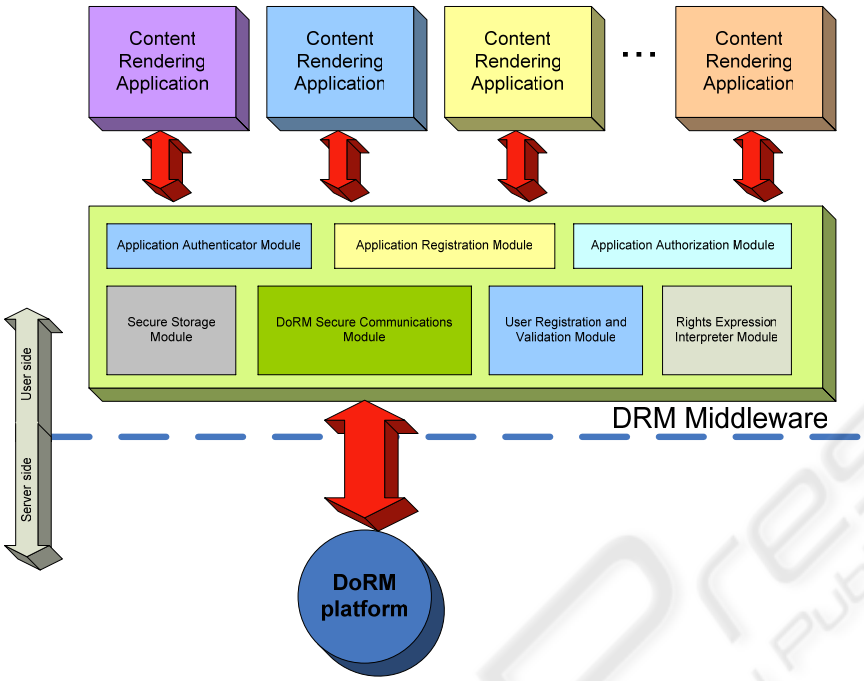


Figure 2: The client-side DoRM layer.

of different functional components. These components are:

- Secure Storage Module: this module is the responsible for storing information at the end-user side in a secure manner. This secure storage uses cryptographic mechanisms to cipher information of the file system, based on the AES cipher algorithm and on information provided by the user and information collected from the system. This module stores information about the user, the CRA and the licenses which are associated to content;
- Application Registration Module: this module handles the CRA registration requests. It responsible for receiving requests from client-side content rendering applications that will be able to handle DRM-protected content. This module registers the application generating cryptographic credentials that will be used latter for application validation;
- Application Authorization Module: this is a module that will receive requests from CRA to perform some action over a DRM-protected item. This module verifies if the CRA is authorized to perform such action over the item, checking the license stored in the system, and returns that clearance to the application together with the content encryption key (or keys);
- Application Authenticator Module: the main task of this module is to authenticate a client-side CRA that is requesting access to a DRM-protected content item. This authentication is based on the credentials that the CRA supplies (issued previously by the Application Registration Module) and that this module verifies;
- DoRM secure communications module: this module handles all the secure communications performed between the DRM middleware layer and the CRA and between the DRM middleware layer and the server-side DoRM platform;
- User registration and validation module: the purpose of this module is to handle the end-user registration at the server-side DoRM platform, establishing the basis for the creation of the secure storage and is also responsible for validating the users that try to access to the DRM middleware layer;
- Rights Expression Interpreter module: this module is capable of interpreting the XML formatted license expressed using a Rights Expression Language, and provide meaningful information to uphold the user content rights

over the different CRA requesting access to DRM-protected items.

On the following section a description of the establishment of the client-side DRM middleware layer is presented.

## 4.1 Establishment of a DRM Middleware Layer

To establish such DRM middleware layer at the end-user side several steps need to be concluded. One of the most important steps that will need to be achieved is the end-user registration at the DoRM platform. This DoRM registration process occurs the first time the DRM middleware is boot up and uses the SSL/TLS protocol to establish a secure and authenticated channel with the DoRM platform servers. The process is composed by the following steps:

1. The DRM middleware layer software computes a key-pair ($K^{pub}_{mid}$, $K^{priv}_{mid}$);
2. The $K^{priv}_{mid}$ is internally stored on a secure repository (an encrypted file) which is ciphered with a key ($SSkey_{AES}$) which is computed hashing information of the pair username and password choose by the end-user and generic information collected from the device – $SSkey_{AES}\{K^{priv}_{mid}\}$;
3. The user will introduce some more information to the DRM middleware interface and then this information is sent to the DoRM platform;
4. The DoRM platform registers the user and returns back a certificate that will validate this DRM middleware installation ($Cert^{DoRM}_{mid}$). This certificate contains the $K^{pub}_{mid}$ and the $K^{pub}_{DoRM}$, signed by the DoRM platform;
5. This certificate is received by the DRM middleware and is also stored in the secure storage – $SSkey_{AES}\{Cert^{DoRM}_{mid}\}$. This concludes the user registration process. Every time the DRM middleware boots up, the user has to authenticate to it – this DRM middleware supports more than one user, meaning that each of the users has to individually register to DoRM, repeating this five steps.

After a successful boot up the DRM middleware can receive multiple requests from different CRA. But before this occurs, the CRA needs to be registered at the DRM middleware – this is necessary to ensure that only registered applications are allowed to use the platform. Only registered and authenticated CRA can request content operations to the DRM middleware (this may include receiving content deciphering keys provided in the rights expressions). This means that any of the CRA that wishes to use this system will need to know how to execute the following two processes:

1) Enrol to and request authentication to the DRM middleware, exchanging a set of credentials with the DRM middleware, to enable application authentication and the establishment of a secure channel between the application and the wallet. The process is performed in the following way:

- The first operation that the CRA needs to perform is to compute a key pair ($K^{pub}_{CRA}$, $K^{priv}_{CRA}$);
- The $K^{priv}_{CRA}$ should be stored securely by the CRA. The CRA sends $K^{pub}_{CRA}$ to the DRM middleware, to register it;
- The DRM middleware, registers the $K^{pub}_{CRA}$, and generates a certificate to be returned for the CRA ($Cert^{mid}_{CRA}$) – this certificate contains also $K^{pub}_{mid}$ and its signed by the DRM middleware;
- The certificate is received by the CRA and stored. The registration process is concluded with success.

After the CRA is registered on the DRM middleware it can use it to request access clearance to perform DRM-protected content operations. This process starts with an authentication between the CRA and the DRM middleware to establish a common secret key to create channel between them. This process is performed in the following way:

- The CRA sends its credentials to the DRM middleware: $Cert^{mid}_{CRA}$;
- The DRM middleware validates the certificate and computes a secret session key ($SessKey_{AES}$). This session key is ciphered with the CRA public key ($K^{pub}_{CRA}$) and returns it to CRA;
- CRA receives the session key and can use it ($SessKey_{AES}$).

2) Request authorization to the DRM middleware to perform operations over the content. This process includes the extraction of content unique identifier (CID) and requesting the DRM middleware the permission to use the content. This middleware is responsible for getting the license from the server, parsing it, analyzing the rights associated to it before the approval or rejection of the operation over the content (this may include passing the decryption key to the application or the appropriate protection tool). This authorization request is expressed in a XML notation. This is processed in following way:

- An authenticated CRA sends an authorization request to the DRM middleware: $SessKey_{AES}\{Authorization_{REQ}, C_{ID}\}$;

- This request is received by the DRM middleware that verifies if there are on the system licenses for the $C_{ID}$ and the User. If not, the DRM middleware connects to the DoRM platform and checks if there is a license for the User and CID specified. If this license exists, it is downloaded by the DRM middleware and securely stored. This license is interpreted locally at the DRM middleware, and the authorization that is requested by the CRA is checked against the rights expression present in the downloaded license. If the request authorization is a valid action over the content, the Content Encryption Key (CEK) is read from the license and returned to the CRA - $SessKey_{AES}\{CEK\}$;
- The CRA receives the $SessKey_{AES}\{CEK\}$ and deciphers the CEK. This CEK is then use to perform the operation over the content. This authorization request is performed each time the CRA when an operation is conducted over the content.

This DRM middleware acts an intermediary between the CRA and the rights management imposed by the DoRM platform.

# 5 CONCLUSIONS

This paper has presented a client-side DRM middleware providing interoperability to DRM-protected content rendering applications. This system uses a distributed service-oriented architecture that exchanges digital rights management related information over an open-network. To create secure and authenticated channels over this open-network, DoRM uses a two-layered system. A first one that is placed at the communication level supplied by the SSL/TLS protocol offering authentication and confidentiality of the services, and a second one, at the application level that offers authentication and non-repudiation of the messages exchanged between the services. The DRM interoperability issue is currently a hot topic. Most of the existing DRM solutions simply do not interoperate or interoperate through very painful mechanisms to the end-user – they are totally vertical. This results that each of the different CRA supports their own specific proprietary DRM system. This paper presented also possible solution, based on a client-side DRM middleware layer established between the CRA and the DoRM platform that is used to manage the content rights. These results in a system that is capable of supporting multiple CRA, multiple content types, and multiple rights expression technologies and can even support multiple DRM platforms – this represents a total horizontal approach to the DRM interoperability problem. A crucial issue on the system is the security of the Content Encryption Key(s). The details of the secure storage and the security of the communication channel between the CRA and the DRM middleware are also described.
This DRM interoperability layer is still at the prototype stage. Future plans include the support for several rights expression languages, the development a generic authorization model and the porting to several devices (PC, PocketPC and Mobile Phone).

# REFERENCES

Duhl J., Keroskian S., 2003, "Understanding DRM Systems", IDC White Paper

Serrão C., 2004, “Open Secure Infrastructure to control User Access to multimedia content”, WEDELMUSIC2004, Barcelona

Serrão C., Neves D., Kudumakis P., Barker T., Balestri M., 2003, "OpenSDRM - An Open and Secure Digital Rights Management Solution", IADIS 2003, Lisboa

ISO/IEC 21000-3 Information technology -- Multimedia framework (MPEG-21) -- Part 3: Digital Item Identification

Microsoft, 2004, “Architecture of Windows Media Rights Manager”, Microsoft Corporation, http://www.microsoft.com/windows/windowsmedia/howto/articles/drmarchitecture.aspx

Lenzi, R. et al, 2003, “Apple iTunes Music Store”, technical report, The Interactive-Music Network

W3C, 2001, “SOAP Security Extensions: Digital Signature”, W3C Note

W3C, 2002, “Open Digital Rights Language (ODRL) Version 1.1”, W3C Note

Dalziel, J., 2004, “DOI in a DRM environment”, White Paper, Copyright Agency Limited

Rosenblatt, B., 2002, "Enterprise Content Integration with the Digital Object Identifier: A Business Case for Information Publishers"

Chiariglione, L., et all, 2005, “NAVSHP (FP6) DRM Requirements Report v1.0”, Networked Audiovisual Systems and Home Platforms strategic objective

Serrão, C., Dias, M., Delgado, J., 2005, “Using ODRL to Express rights for different content usage scenarios”, 2nd ODRL Workshop 2005

ISO/IEC 14496-1:2003, Information technology -- Coding of Audio-Visual Objects -- Part 1: Systems, Amendment 3: IPMP Extensions