Chillout - the Open Source DRM Software

Filippo Chiariglione – CEDEO.net
Yuqiang Liao – Joint Digital Laboratory China
Abstract

This paper presents Chillout [1], the Open Source DRM Software implementing the open Digital Media Project’s (DMP) Interoperable DRM Platform (IDP) [3] specification. The authors believe that only an Open Source DRM implementing an open standard can break the stalemate that is preventing the users of digital media Value Chains to fully exploit the power of digital media.

The authors are also contributors to the IDP specification, and among the main developers of Chillout.

1. INTRODUCTION

Today’s market of digital media is still far from the take up that many had expected a few years ago, when the digital technologies, replacing the analog ones, brought completely new features to the users of media value chains. One of the main reasons is that, up to now, there is no viable choice for Content Providers to make content available. Indeed either content is released without any Digital Rights Management (DRM), arguably without means to get direct revenues, or it is locked to one of the many DRM providers’ solutions, an option that is not practicable for many, and leads to user dissatisfaction.

The alternative is providing all users in the Media Value Chain with a DRM standard which is open, interoperable and enables the use of content as users traditionally experience [2]. In April 2006 the Digital Media Project (DMP), a non-for profit International Organization, has released the Interoperable DRM Platform (IDP) Specification [3], an open DRM Standard providing solutions at any point in the Media Value Chains.

In parallel to this the DMP has started developing the reference software for the IDP as an Open Source project called Chillout, released under the Mozilla Public Licence V. 1.1 (MPL 1.1) [4]. The reason for choosing MPL 1.1, among the many OS licenses available, is that it allows users to create commercial implementations based on the software without further obligations and it clearly separates the licence of the code from the licence of patents that may be needed for practicing the code.

Chapter 2 explains the vision behind the Chillout project. Chapter 3 presents some basic requirements for setting-up value chains, and highlights the main components required. Chapter 4 gives an overview of the Chillout software and analyses the software requirements that have led to the actual choices made by the project. Chapters 5, 6 and 7 detail the Chillout software layers. Chapter 8 shows an alternative use of Chillout, while Chapter 9 presents some conclusions and envisages further steps of development.

2. THE VISION

DRM is a technology that is destined to play a very important role in the future society where an increasing amount of information exchanged between the members of society will need to be managed and more and more assets will be immaterial and will need to be securely traded.

So far many media companies have considered DRM as the technology of choice when implementing digital media value chains. However, many of these have led users to feel like they have been deprived of the right to express what digital media value chains should look like.

The Digital Media Project’s open DRM specification [3] makes one step toward allaying the concerns of users. Indeed such specification can trigger the creation of horizontal markets with a lower cost of the DRM technology and an easier access to value chains than in the case of proprietary DRM.

However, as long as the implementation of DRM remains a specialist’s job, there is not much hope that DRM will become the tool that members of the society – which of course include media companies – can use to serve their needs for exchanging information and trading digital assets.

The DMP has decided to start an Open Source Software project called Chillout, because it was felt that successful use and acceptance of DRM hinges on the existence of a potentially vast community of users and developers that makes a DRM solution that is openly accessible, satisfies disparate user requirements, is robust and capable to evolve. Moreover, the fact that the code can be inspected by anyone should convince those who have been brainwashed by various no-DRM initiatives in the last few years that DRM is no evil; instead it provides an answer to some obvious user demands, can improve media life and enable a fair exploitation of digital media.

3. SETTING UP VALUE CHAINS

A digital media value chain is a network of business players (called users) who perform functions on the media flowing through it. Users need devices to perform the functions on the digital media.

Figure 1 below exemplifies a rather general case of value chain, where it can be seen that all entities in the value chains (data and devices) need to be identified. This is done by

- The 3 leftmost devices in the figure that provide unique and unambiguous numbers to:
  - content, i.e. XML structures attached to resources and describing them (called DMP Content Information - DCI)
• licences describing the rights of a user to perform functions on content
• DRM Tools used to perform DRM functions such as decryption, key management, etc.

- The Device Identification Device that provide unique and unambiguous numbers to devices.

![Figure 1 – An example of Media Value Chain](image)

The Content Creation Device is a device that can be used to create content, while the next 3 devices from the left are those that actually contain and provide the entities to other users in the value chain. End users can employ two types of devices.

- Portable devices (PAV), not connected to the network and needing a PXD to do so
- Stationary devices (SAV), directly connected.

A domain is a group of devices (e.g. belonging to a home) identified by a Domain Identification Device and actually managed by a Domain Management Device.

In abstract terms a device exchanges data with other devices using specially defined protocols. The larger part of the IDP specification concerns:

- the way data are digitally represented
- the protocols carrying such data.

4. CHILLOUT OVERVIEW

4.1. Language requirements

As many other Open Source Software project, Chillout is written in the Java programming language. The reasons for choosing Java are manifold: for a start, because Java is an outstanding language, it possesses excellent cross-platform capabilities and it is supported by many international companies operating in various fields. Therefore Chillout expects to involve more and more interested volunteers via such a popular and living technology.

On the other hand, being the IDP-2 specification based on XML and Web Services, any other programming language could have been chosen instead of Java. At the time this paper was written, more initiatives aiming at developing parallel implementation of Chillout in other languages such as .NET were about to start. The benefit of additional implementations in different languages could highlight those circumstances where a language is preferred to another, and highlight any interoperability problem between different implementations.

4.2. Software architecture requirements

The field of digital media is certainly one among those where innovation plays a key role. The better performing technologies become widely used in a very short time but may also become obsolete in a very short time. The same happens to digital media business models. With such premises, a monolithic DRM specification would never live a long life.

This is the reason why IDP-2 was conceived since the beginning as a toolkit, and the same approach has been adopted in Chillout. Depending on their business model users may chose a subset of the IDP tools, configure them (as specified in the specification) for their specific needs and use them straight away.

For this purpose, Chillout is structured in four layers, as shown in Figure 2.

![Figure 2 - Chillout Software Layers](image)

The high level description of each layer is given below:

- **Java Platform Layer**: this layer provides the Java running environment on which Chillout devices depend on. It comprises of the Java Development Kit (JDK), plus a number of add-ons provided by third parties, such as the Apache Tomcat servlet container [5] to power web applications, the Apache Axis [6] SOAP implementation, providing web-service capability, the EJBCA [7] providing a Certificate Authority for authentication and authorization, the Java Media Framework [8] for rendering media resources, etc.

- **Core library**: library of classes implementing the Primitive functions defined in the Technical Specification. This software is normative as much as the IDP-2 specification [3], in the sense that the two are meant to be technically aligned.

- **Auxiliary library**: library of classes encapsulating the functionalities that every device must have
when operating in a real environment. This library is not normative, as these modules may be replaced by those a developer needs.

- **Applications**: a set of sample applications with the purpose of showing how to use the Core and Auxiliary libraries. This category includes a number of Devices, such as a SAV, a Content Creation Device (CCD), a License Provider Device (LPD), a Content Provider Device (CPD), etc. This category also includes testing applications such as those for testing protocols between devices, well-formedness of DCIs, files, licences, etc. (see conformance later in this paper)

The separation of Chillout software in layers allows any user wishing to set-up or become part of a media Value Chain to replace any Auxiliary module with his own proprietary ones, without the need of modifying the core library, if he wishes to do so. In the future, thanks to the power of Open Source, it is expected that a plethora of Auxiliary modules at "product level" could be part of the larger Chillout ecosystem.

5. **THE CORE LIBRARY**

The Chillout Core Library provides the functionalities needed to:

- **Represent**: a set of classes to generate and parse the XML structures defined by IDP.
- **Package**: a set of classes to generate and parse the DMP Content File (DCF) and Streaming (DCS) formats.
- **(perform the) Protocols**: a set of classes to generate and parse the messages exchanged by two devices performing the Protocols, and providing the functionality to send and receive such messages.

5.1. **The Represent classes**

The DCI, i.e. the information related to content, such as its structure, metadata, identifiers, licences, protection etc. as well as the payload of any message exchanged between two devices is expressed in XML.

The Chillout Core library provides a set of classes that allow an application to easily generate any XML structure defined in IDP, and conversely to extract any information contained within. This was achieved by employing an Open Source software tool called Java Architecture for XML Binding (JAXB) [9]. By taking the IDP-2 schemas as input, JAXB can generate a set of Java classes for manipulating XML structures conformant to the schemas. Figure 3 shows how JAXB is employed to generate the Represent Core classes.

In addition to generating the classes to manipulate XML structures, JAXB is also employed at runtime, anytime an application using such classes needs to:

- Unmarshal the DCI or any of its parts to Java objects (the instances of the classes generated from the schemas).
- Marshal Java objects into XML structures.
- Verify if the incoming xml document is valid or not according to the schemas.

5.2. **The Package classes**

Users in Media Value Chains sometime require transferring media resources, DCIs and licences bundled together. IDP-2 specifies the DCF and DCS when the bundle is stored in a file or sent over a stream (IP or MPEG-2 TS) respectively.

The Core library provides a set of functions to bundle media resources with the DCI and other information in a DCF, and conversely to extract this information from a DCF. The DMP Content Format is based on the MPEG-21 File Format [10]. According to this specification, files are formed as a series of objects, called boxes. The Package classes allow any Java application to access the information contained in a DCF or to create a DCF based on data in input.

5.3. **The Protocols classes**

Devices in a media Value Chain, such as those shown in Figure 1, require a standard protocol to communicate. The description of the various protocols is done in WSDL [11]. Specifically, the payload exchanged between two devices performing the protocol is specified as XML Schema in IDP, and such payload is exchanged over the SOAP [12] protocol.

Figure 4 shows at a high level how two devices communicate.
Secure communication can be achieved either by transporting messages over the Secure Socket Layer (SSL) on the OSI network layer, or by following the Web Service Security (WSS) [13] specification.

The Core library provides a set of functionalities to generate and parse the XML messages exchanged between devices while performing a Protocol, while the Open Source Axis [6] tool is used to send and receive such messages over the SOAP protocol.

6. THE AUXILIARY LIBRARY

Unlike the Core library, the Auxiliary library has no normative value. It comprises those classes encapsulating the functionalities of a number of modules required for devices to operate according to the IDP specification. However, commercial application may well decide to implement those key DRM components in a proprietary way, even in hardware, therefore it was felt that Chillout would only provide a reference implementation of them.

On the other hand, in order to provide interoperability at a deeper level, some of the interfaces made available by the Auxiliary components may become part of the IDP-3 Technical Specification.

The main software components of the Auxiliary library are described below.

6.1. The Security Manager

The Security Manager is a module incorporating all those functionalities such as securely storing digital certificates and licences, performing operations involving Digital Signatures, etc...

The security functions provided by the Security Manager are based on the java.security and javax.crypto libraries part of the JDK.

6.2. The DRM Processor

The IDP does not specify any specific algorithm to govern media resources: this choice is left to the user implementing the specification. However, IDP specifies how to signal that one or more media resources part of a content item are governed, and which DRM Tools shall be employed to access it.

The DRM Processor is the SAV module in charge of instantiating and managing the DRM Tools protecting a media resource. After receiving the list of all the DRM Tools required for each governed media resource, the DRM Processor locates them either locally on the SAV or remotely, and instantiates them. Usually, before any governed resource is processed, the DRM Tools are required to mutually authenticate with the platform on which they operate. This operation, too, is made possible by the DRM Processor. If this operation succeeds, the DRM Processor initialises any DRM Tool requiring it, and acts as a message dispatcher between any DRM Tool and the SAV during the lifetime of the Tool.

The message payload exchanged by means of the DRM processor is defined in IDP-2. This allows a standard communication protocol between DRM Tools and the hosting SAV not just for authentication purposes, but also for exchanging various types of information, such as decryption keys extracted from the DCI.

The DRM Processor provided in the Chillout Auxiliary library provides a reference implementation of the functionalities described above. It is expected that this module could be replaced by a tamper-resistant component in commercial implementations.

6.3. DRM Tools

DRM Tools are certainly among those DRM components that a user wishing to make a commercial implementation of Chillout would provide on his own, either in software or in hardware, as these may contain the “secrets” of his business. However, the Chillout project has provided as part of the Auxiliary library a DRM Tool performing DES decryption, which can be used either in real applications or as a reference to build new DRM Tools.

7. APPLICATIONS

Chillout does not only provide the source code of core and auxiliary components, but also a set of devices that are easy to configure and run. This helps promoting adoption of Chillout and the DMP specification by enabling those users not experienced in the DRM field, not enough skilled in programming or working for small companies, to experiment with a technology that so far could only be used by those experts in the field.

The Applications category contains source and executable Java code built on top of the Core classes, and integrated with Auxiliary classes.
In September 2006, Chillout made its first debut at the IBC Exhibition [14] in Amsterdam. The set-up is described by the figure below.

![Figure 5 – The Chillout demo at IBC 2006](image)

The features of the devices used in the demo are:

1. A **SAV** capable of playing audio-visual resources stored in files or streamed over the IP protocol. By means of a browser integrated in the application, users can browse a Content Provider's portal to select content for real time streaming or file download. The SAV supports playing both clear-text and encrypted resources.

2. A **Content Provider Device** from which every SAV can choose from a list content items to receive over the RTP protocol as a stream of data or downloaded as a file.

3. A **License Provider Device** issuing licences to SAVs to use content (play and record are the rights supported).

4. A **Content Identification Device** issuing identifiers to new content items created by a SAV whenever content is recorded while streaming.

The demo shows that in the case a SAV has a licence to play content, the user would not even notice that content is governed. However, if the user attempts to perform some additional functions on content without a licence, the SAV would acquire a licence from the LPD.

This operation requires the SAV to generate the "sought licence", the licence that would have been required in order to proceed with the user request. The sought licence is sent to the LPD as part of the Access License as File Protocol specified in IDP-2. In the case the requested licence can be issued by the LPD (according to LPD own business rules), the new licence would be returned to the SAV, and content would be accessible.

In case of a negative response from the LPD, if content is not encrypted, the user can decide to proceed with his action, although the operation is not permitted. Otherwise, if the resources is encrypted, the user would not be able to proceed, as the decryption key required to access the resources (conveyed in the licence) is not available.

Each devices developed is characterised by a Manager (e.g. the SAV Manager) that defines the business logic that the Application shall implement. Ideally, to implement a different business model, a developer only needs to change the Manager class in a device to implement the application business logic.

### 8. CONFORMANCE USING CHILLOUT

All open standards managed by a community need an open and fair regime whereby a provider of devices can have an implementation tested for conformance. In the case of a DRM environment, having a device successfully tested for conformance is the first step before the device can be certified.

Chillout will provide the tools to carry out conformance testing for

1. **Content and Content Elements**
2. **Protocols and Package Tools**
3. **Devices**.

A content item or content element under test is used as an input to a reference parser. The parser provides as output an indication of whether the syntax of the content item or a content element under test is correct or not. Depending on the implementation it may also provide indications of what in the content item or content element under test is wrong and, if possible, why (in the latter case).

An implementation of a protocol or a package tool to be tested for conformance is provided either as an implementation of one side of the protocol or package tool or as clearly separated parts of the two sides of the protocol or package tool. The side to be tested for conformance is installed on a device with the appropriate interfaces and the behaviour of the device is assessed when it communicates with a reference device implementing the other half of the protocol or package tool.

To test a device for conformance it is possible to use Chillout devices as reference implementations against which to test an implementation.

### 9. CONCLUSIONS

This paper has provided the rationale, the architecture, the state of the art and plans of the Open Source Chillout® Reference Software implementing the Digital Media Project Interoperable DRM Platform specification.
The vision behind the DMP Chillout project is to create an open and fair security infrastructure that can be employed for a variety of uses. This is based on an Open Source Software implementation of the IDP specification, managed by an active open source community dedicated to foster adoption of interoperable DRM. Adhering to the open source spirit, the Chillout community [1] welcomes everyone who wants to be part of the exciting project to set up a fair digital media world.

10. REFERENCES