Towards a General Framework for Digital Rights Management (DRM)

A dissertation submitted to the Department of Computer Science,
Faculty of Science at the University of Cape Town
in fulfilment of the requirements for the degree of

Doctor of Philosophy

in
Computer Science

— Alapan Arnab —

Supervisor
Adjunct Professor Andrew Hutchison

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

June, 2007

Abstract

Digital rights management (DRM) can be defined as a technology that enables persistent access control. The common understanding of DRM is that of a technology that enables means to thwart piracy of digital multimedia through limiting how the media is used by the consumer. It can be observed that many of these restrictions can be applied to any type of data. Therefore, it should be possible to create a two part DRM system – a common DRM system that enforces the basic access controls (such as read, write and execute) and an application specific DRM system that enforces the application specific access controls (such as print and play). The aim of this dissertation is to create such a framework for distribution independent DRM systems.

Most vendors promote DRM as a copyright protection mechanism, and thus consumers expect a number of rights that are allowed by copyright legislation, but which are not available for the DRM protected media. However, DRM is not an enforcement of copyright law, but rather an enforcement of a licensing regime. Thus, there is incorrect (and possibly false) marketing of DRM enabled media from the vendors of DRM enabled media, leading to dissatisfied consumers. We think that one of the main reasons for the current situation, is that there is no defined legal framework governing the operation of DRM systems. In this dissertation, we address this gap, by developing a legal framework for DRM systems as one of the components of our DRM framework.

Negotiation can be defined as the process which leads to the conclusion of a contract. Since DRM is the enforcement of licensing agreements, there is a need to cater for negotiation protocols in DRM systems. Negotiations provide the consumer with the power to request different rights packages, especially when consumers have a legitimate need for rights not granted normally to other consumers (for example, disabled consumers have needs that may not be met with standard rights set). Negotiations also allow the possibility for the licensors to extract the maximum value from the consumers. For this reason, the inclusion of negotiation protocols in DRM systems can become a powerful tool, and in this dissertation we present the first negotiation protocols for DRM systems.

Even though the definition of DRM as an access control model has existed since at least 2002, there has been no formal description of DRM as an access control model. Thus, there are no formal models for any of the rights expression languages which express DRM access control policies, and various authors have commented on ambiguities present in interpretation and enforcement of licenses expressed in these languages – a result of a lack of formal definition of these languages. In this dissertation, we develop a formal model for a **Li**censing **R**ights **E**xpression **L**anguage (LiREL), which is designed to provide a mechanism to express access control policies which are also sound legal license documents. Our formal model also discusses the enforcement of the access control policies, and is thus the first formal model for DRM as a mechanism for access control.

Access control is a two part process: *authentication* of the parties involved and *authorisation* of the parties to access the resources. Authorisation in DRM provides some unique challenges: there is a need to support multiple platforms, without guaranteed network connectivity and minimal trust between the parties involved. For this reason, the associated authentication framework becomes more complex.

While many access control models define user management as part of their model, we have taken a different approach, and removed user management from the core DRM system. Instead, our authorisation process requires a trusted verification of the user's credentials and then decides on the access control request. For this reason, our user authentication framework is ticket based, and shares similarities to Kerberos tickets.

DRM also requires a strong data identity management. However, all the current identity systems for data do not provide verification service for data identity. For this reason, we developed **V**erifiable **D**igital **O**bject **I**dentity (VDOI) System, to address this gap.

These components are combined towards a general framework for digital rights management that advances the understanding, organisation and implementation of DRM compared to approaches or solutions which are currently available.

ACKNOWLEDGEMENTS

Prof. Andrew Hutchison for his guidance, supervision and patience during the course of my degree. He is an incredible supervisor, and I have enjoyed working with him.

Prof. Julien Hofman, who has been my unofficial, secondary supervisor. His insight of electronic law, and his guidance in exploring the legal position of DRM gave a new dimension to the direction of my research.

Mr. Marlon Paulse and Mr. Duncan Bennett, who took the challenge of implementing a DRM controller (and still follow my design) for the GNU-Linux kernel, and succeeded.

Ms. Mary Faragher for proof reading and editing the draft of this dissertation. All mistakes that remain, are off course all mine.

Ms. Vicky Weissman, Dr. Sussanne Guth, Dr. Renato Iannella and the rest of the ODRL v2 working group, for our discussions, and their comments.

Mr. Martin Springer and DMP for our discussions and their comments.

Prof. Pieter S Kritzinger and the rest of the DNA Research group for their continued support and suggestions. It has been a great experience!

Mr. Nicholas Hall for drawing the faces of Alice, Bob, Charlie and their friends.

To my family and friends that have supported me during this journey.

And last, but not least, a thank you to the financial support from National Research Foundation (NRF) of South Africa, and the University of Cape Town (UCT) Postgraduate Funding Office. Their support made this journey easier.

CONTENTS

Abstract i

Acknowledgements iii

Contents iv

List of Figures xiv

List of Tables xvii

1 Introduction 1

1.1 Users of a DRM System 3

1.2 Scope 4

1.3 Layout 4

I Requirements and Related Work 7

2 Requirements for a General Framework for DRM 9

2.1 Sources for Requirements 9

2.2 Core Requirements 10

2.3 Usability Requirements 13

2.4 Legal and Social Requirements 16

2.5 Summary 17

3 A Discussion of Existing DRM Systems 19

3.1 Apple iTunes Music Store 20

3.1.1 Brief Overview 20

3.1.2 Core Requirement Analysis 21
3.1.3 Usability Requirements Analysis 21
3.1.4 Legal and Social Usability Analysis 23
3.2 Microsoft Windows Media (WM) 24
3.2.1 Brief Overview 24
3.2.2 Core Requirement Analysis 25
3.2.3 Usability Requirement Analysis 26
3.2.4 Legal and Social Requirement Analysis 26
3.3 Helix DRM 27
3.3.1 Brief Overview 27
3.3.2 Core Requirement Analysis 28
3.3.3 Usability Requirement Analysis 28
3.3.4 Legal and Social Requirement Analysis 29
3.4 Open Mobile Alliance (OMA) DRM 30
3.4.1 Brief Overview 30
3.4.2 Core Requirement Analysis 31
3.4.3 Usability Requirement Analysis 32
3.4.4 Legal and Social Requirement Analysis 33
3.5 Adobe Document Security 33
3.5.1 Brief Overview 34
3.5.2 Core Requirement Analysis 34
3.5.3 Usability Requirement Analysis 34
3.5.4 Legal and Social Requirement Analysis 35
3.6 Authentica's Active Rights Management Platform (ARM) 36
3.6.1 Brief Overview 36
3.6.2 Core Requirement Analysis 37
3.6.3 Usability Requirement Analysis 37
3.6.4 Legal and Social Requirements Analysis 38
3.7 Microsoft Rights Management Services (RMS) 39
3.7.1 Brief Overview 39
3.7.2 Core Requirements Analysis 40
3.7.3 Usability Requirement Analysis 41
3.7.4 Legal and Social Requirements Analysis 42
3.8 Standardisation Efforts 42

3.9 Summary 43

4 Theoretical Foundations for DRM Systems 47

4.1 Role Players in a DRM System 47

4.1.1 Discussion 47

4.1.2 Analysis of Role Players in Existing DRM Systems 48

4.2 Security Taxonomy 49

4.2.1 Discussion 49

4.2.2 Characteristics of the Security Architectures 51

4.2.3 Categorisation of Existing Systems 51

4.3 Rights Expression Languages (RELs) 51

4.4 The DRM Reference Model 54

4.5 A Layered Approach to DRM 55

4.5.1 Discussion 55

4.5.2 Analysis of Existing Systems 57

4.6 Interoperability in DRM 57

4.6.1 Types of DRM Interoperability 60

4.6.2 REL Interoperability 61

4.6.3 Interoperability through Intermediaries 61

4.7 Protocol and System Modelling 62

4.8 Summary 63

II Framework 65

5 An overview of our General Framework for DRM 67

6 Legal Framework For DRM 69

6.1 Copyright & DRM: A Background 70

6.2 Copyright & DRM: Licensing 72

6.2.1 Reproduction and Distribution in the Digital Arena 73

6.2.2 Abuse: Protecting what should not be protected 75

6.2.3 Camp and Copyright Management 75

6.2.4 “Personal Use” 76

6.2.5 The Focus on Use and the Licensing Paradigm 76

6.3 Licensing in DRM: Requirements and Regulation 77

6.3.1 Requirements for Contracts 77
6.3.2 Licensing and the Problems With Current DRM Systems 78
6.3.3 Form and Content of Licensing Contracts 80
6.3.4 Privatisation of Copyright – Buying vs Licensing 83
6.3.5 Copyright Tribunal and License Arbitration 84
6.4 Fairer Use 85
6.4.1 Fairer Use through Negotiations 85
6.4.2 Fairer Use through the use of Credentials 87
6.5 Other Legal Issues 88
6.5.1 Globalisation Effect of the Internet 89
6.5.2 Trade Practices 90
6.5.3 Privacy 90
6.5.4 Electronic Transactions Act 91
6.6 Summary: Towards a Legal DRM System 92

7 A Negotiations Framework for DRM **93**
7.1 A note on terminology 94
7.2 Related Work in Electronic Negotiations 94
7.3 Types of Negotiations 95
7.4 Requirement for Negotiation for DRM 96
7.4.1 Factors affecting Negotiation in DRM 97
7.4.2 DMP requirement for negotiations 98
7.5 Bidding 98
7.5.1 Process 99
7.5.2 Protocol 99
7.5.3 Modelling 101
7.6 Bargaining 101
7.6.1 Process 103
7.6.2 Protocol 103
7.6.3 Modelling 107
7.7 RELs and the Expression of Negotiations 107
7.8 Negotiation Requirement Analysis 108
7.8.1 Raiffa’s Factors Affecting Negotiation 108
7.8.2 Satisfying DMP Requirements 109

7.9 Example 110
7.10 Summary 110

8 A Formal Model for DRM 113
8.1 Existing Access Control Mechanisms 114
8.1.1 DAC, MAC and RBAC 114
8.1.2 Differences between DRM and Existing Access Control Models 115
8.1.3 XACML and RELs 116
8.2 Additional Requirements for RELs 117
8.3 A Licensing REL: LiREL 120
8.4 A Formal Description for LiREL 120
8.4.1 Language Semantics and Syntax vs Language Vocabulary 121
8.4.2 Obligations 121
8.4.3 Constraints 122
8.4.4 The Licensors 122
8.4.5 The Licensees 123
8.4.6 The Third Parties 124
8.4.7 The Resources 125
8.4.8 The Agreement 125
8.4.9 Catering for Negotiations 128
8.4.10 Visual Model 130
8.4.11 Comparison to Current RELs 130
8.4.12 XML Schema 133
8.5 Analysis of Access Control Enforcement 133
8.5.1 Validity of Use Licenses 134
8.5.2 Enforceability of Use Licenses 134
8.5.3 Conflict Resolution 135
8.5.4 Deciding a Request 135
8.5.5 Determining Cardinality 136
8.6 Summary 136

9 An Authorisation Framework for DRM 139
9.1 The Authorisation Process 139
9.1.1 The DRM Controller 140
9.1.2 Rights Levels 141

9.1.3 Multiple Use Licenses and Authorisation . . . 142
9.1.4 The Process . . . 143
9.1.5 Verification of the correctness of DRM Controllers . . . 145
9.2 Motivation for Decoupling Authentication and Authorisation . . . 145
9.3 Management Console . . . 146
9.4 Modelling The Authorisation Process . . . 148
9.5 Summary . . . 150

10 An Authentication Framework for DRM 151
10.1 Related work on Authentication . . . 152
10.1.1 Ticket based User Authentication . . . 152
10.1.2 Resource Authentication . . . 153
10.1.3 Device Authentication . . . 157
10.2 User Authentication through Time Limited Tickets . . . 157
10.2.1 General Overview . . . 158
10.2.2 Ticket Design . . . 159
10.2.3 Security Considerations . . . 160
10.3 Other Forms of User Credentials . . . 161
10.4 Verifiable Digital Object Identity System . . . 163
10.4.1 Basic Setup . . . 163
10.4.2 Identifier Resolution . . . 163
10.4.3 Identifier Format . . . 164
10.4.4 Identifier Verification . . . 165
10.4.5 Management of Identifiers . . . 166
10.4.6 Security Considerations . . . 168
10.5 Summary . . . 169

11 Creating Protected Works 171
11.1 The Packaging Service . . . 171
11.1.1 The Packaging Workflow . . . 172
11.1.2 Encryption Keys . . . 174
11.1.3 Regulation of the Packaging Service . . . 174
11.2 A Layered Approach to DRM Packages . . . 174
11.2.1 The Data Layer . . . 176
11.2.2 The Compression Layer . . . 176

11.2.3 The Packaging Layer 177
11.2.4 Unpacking the Protected Package 177
11.2.5 Comparison to other Packaging Formats 178
11.3 Summary 178

12 Web of Trust & Key Distribution in DRM 181
12.1 Web of Trust for DRM 182
12.1.1 The DRM Controller 182
12.1.2 An Independent Verification Authority 182
12.1.3 License Server 183
12.1.4 Authentication and Credentials Service 184
12.1.5 Packaging Service 184
12.2 Key Distribution in DRM 185
12.2.1 Public Keys for Signature Verification 185
12.2.2 Decryption Keys for Protected Data 188
12.3 Comparison to the to OMA DRM Approach 189
12.4 Summary 190

III Implementation, Analysis and Conclusions 191

13 Experiences in Implementing a Kernel-Level DRM Controller 193
13.1 Related Work 194
13.2 System Design 195
13.2.1 DRM Controller – Management Daemon Interaction 195
13.2.2 File Format 196
13.2.3 Rights Expression Language (REL) 197
13.3 Implementation Details 197
13.3.1 Setup 197
13.3.2 Permissions and Constraints Implemented 198
13.3.3 The Enforcement Engine: The Operating System Kernel Module 199
13.3.4 Motivation for our approach 202
13.4 Experimental Evaluation 203
13.4.1 Experiment Setup 203
13.4.2 Test Environment 204

13.4.3 Results 204
13.4.4 Analysis 204
13.5 Analysis of our Approach 207
13.5.1 Application Level Transparency 207
13.5.2 Wide range of rights 207
13.5.3 Performance 207
13.5.4 Comprehensive Protection 208
13.5.5 Modification of Protected Files 208
13.5.6 Correct Identification of Accesses 208
13.5.7 Stream Encryption 209
13.5.8 Compensating for Application Behaviour 209
13.5.9 Implications for hardware based DRM systems 209
13.6 Summary 209

14 Conclusions and Future Work **211**
14.1 Requirement Analysis of our General Framework for DRM 213
14.1.1 Core Requirements 213
14.1.2 Usability Requirements 215
14.1.3 Legal and Social Requirements 216
14.2 Future Work 217
14.3 Conclusions 217

IV Appendices **219**

A Abbreviations **221**

B XML Schemas: LiREL **223**

C XML Schemas: A Sample Permission Set for LiREL **229**

D Negotiation using LiREL **233**
D.1 The Initial Enquiry 233
D.2 The First Offer 234
D.3 The Counter Offer 236
D.3.1 Rejection of the Offer 236
D.3.2 The Counter Offer 238

D.4 The Counter-Counter Offer (Another Offer) 240
D.4.1 Rejection of the Counter-Offer 240
D.4.2 The Offer 242
D.5 The Acceptance of Offer 244
D.6 The Concluding of an Agreement 246

Bibliography **249**

LIST OF FIGURES

1.1 The DMP DRM Value Chain [63] 4
1.2 Categorisation of DRM Systems [149] 5

3.1 Overview of the Helix Platform [101] 27
3.2 DRM Types in OMA 1.0 [146] 30
3.3 Overview of OMA DRM 2 Architecture [147] 31
3.4 Overview of Authentica Active Rights Management Platform [44] 36
3.5 Deploying a RMS system 39

4.1 Categorisation of DRM Systems [149] 50
4.2 The DRM Reference Model [168] 54
4.3 Proposed DRM Layers 56

5.1 A General Framework for DRM 67

6.1 Simple Negotiations Protocol 86

7.1 The DMP DRM Value Chain [63] 97
7.2 Bidding Process 99
7.3 Bidding Petri Net 101
7.4 Flowchart for a bargaining protocol 102
7.5 Bargaining Petri Net 106

8.1 Comparing RELs functionalities [90] 120
8.2 UML model of **L** 131
8.3 Simple XrML Model [142] 132
8.4 ODRL 2 Model (Draft) [105] 133

9.1 DRM Controller Decision Flowchart 144
9.2 Management Console interfacing with a number of different DRM Controllers . 146
9.3 Petri Net of the Authorisation Process 149

10.1 A layered view of DRM Packages 156
10.2 Authentication Ticketing System 159
10.3 Proposed ticket format 160
10.4 XML schema diagrams for a customer receipt as discussed in [38] 162
10.5 XML schema each step of the identifier registration protocol for VDOI system. 167

11.1 DRM Packaging Service Workflow 172
11.2 XML schema of a signed receipt from the service producer 173
11.3 XML schema of a signed request to create a DRM package 173
11.4 A layered view of DRM Packages 175

12.1 Web of Trust with only the DRM Controller 183
12.2 Web of Trust with the DRM Controller and an Independent Verification Authority 184
12.3 Web of Trust with the DRM Controller, an Independent Verification Authority and a License Server 185
12.4 Web of Trust with the DRM Controller, an Independent Verification Authority, a License Server and Authentication & Credentials Service Providers 186
12.5 Web of Trust with the DRM Controller, an Independent Verification Authority, a License Server, Authentication & Credentials Service Providers and a Packaging Service 187

13.1 The DRM Controller Architecture and Communications 195
13.2 Layered approach to DRM file formats (left) and the implemented file format (right) 196
13.3 The BNF grammer for our simplified REL 197
13.4 The interaction of the various components within the kernel module 199
13.5 Intercepting system calls and redirecting file access requests in the kernel . . . 201

LIST OF TABLES

3.1 Core Requirement Analysis for iTunes FairPlay 22

3.2 Usability Requirement Analysis for iTunes FairPlay 23

3.3 Legal and Social Requirement Analysis for iTunes FairPlay 23

3.4 Core Requirement Analysis for Windows Media 25

3.5 Usability Requirement Analysis for Windows Media 26

3.6 Legal and Social Requirement Analysis for Windows Media 27

3.7 Core Requirement Analysis for Helix DRM 29

3.8 Usability Requirement Analysis for Helix DRM 29

3.9 Legal and Social Requirement Analysis for Helix DRM 30

3.10 Core Requirement Analysis for OMA DRM 1 and OMA DRM 2 32

3.11 Usability Requirement Analysis for OMA DRM 1 and OMA DRM 2 33

3.12 Legal and Social Requirement Analysis for OMA DRM 1 and OMA DRM 2 . . 33

3.13 Core Requirement Analysis for Adobe Document Security 35

3.14 Usability Requirement Analysis for Adobe Document Security 35

3.15 Legal and Social Requirement Analysis for Adobe Document Security 36

3.16 Core Requirement Analysis for Authentica ARM 38

3.17 Usability Requirement Analysis for Authentica ARM 38

3.18 Legal and Social Requirement Analysis for Authentica ARM 39

3.19 Core Requirement Analysis for RMS 41

3.20 Usability Requirement Analysis for RMS 41

3.21 Legal and Social Requirement Analysis for RMS 42

3.22 Summary of our requirement ratings for the various DRM systems which we discussed in this chapter. 45

4.1 Identifiable Role Players in Current Systems 49

4.2 Characteristics of DRM Architectures [149] 52
4.3 Categorisation of Existing Systems 52
4.4 Analysis of Media DRM Systems against the Layered Architecture 58
4.5 Analysis of Enterprise DRM Systems against the Layered Architecture 59

8.1 Illocutions for a logic-based negotiation language as discussed by Wooldridge and Parsons 129
8.2 Illocutions for negotiation in DRM 129

9.1 Classification of Level 1 and Level 2 Rights 142

13.1 The types and sizes of the files used during the performance evaluation experiment. 204
13.2 Comparing the duration of the read() system call when handling DRM protected and non-DRM protected data on a DRM-enabled and DRM-free system. 205
13.3 Comparing the duration of the rename() system call when handling DRM protected and non-DRM protected data on a DRM-enabled and DRM-free system. 205

CHAPTER 1

INTRODUCTION

Protection of confidential data has been a problem since the ancient times. Protection of data has always hinged on two important factors: keeping the data a secret and trusting the people who have access to the secret. Since the time of Julius Caesar, secrecy has been achieved through various encryption schemes, which over time have become better and better, to the extent that current computing power will not be able to break a strongly encrypted message in usable time. However the issue of trust has not been resolved, and intentionally or unintentionally, holders of secret information can do what they want to do with the secret information.

Protection and control of information dissemination is not limited to the military (as in the case of Caesar). For example, in December 2004, Apple Computer Inc.[1] took three online publications, Powerpage, Apple Insider and Think Secret, to court for publishing detailed information on new unannounced products [86]. The aim was to force the publishers to reveal their sources on who leaked the information (and thus violated Apple's non-disclosure agreements). This situation arose because Apple could not protect their trade secrets from being distributed freely to unauthorised persons. Apple is not the only company with this problem and there have been a number of estimates on the cost to enterprises through intellectual property loss. Furthermore, there is a number of new legislations promoting data privacy and protections, such as the Health and Financial sectors such as HIPAA in the USA [189], which require technical and business processes that can control the flow of protected and sensitive data.

Data does not necessarily have to be sensitive to require control. Reproduction of digital data is an inherently simple task; and with the growth of the Internet and wider access to broadband Internet access, distribution of digital data is becoming cheaper, faster and easier. For this reason, control of copyrighted digital works that are currently not in the public domain, is of great interest to copyright holders. Even when copyrighted works are not sensitive in nature;

[1] http://www.apple.com/. Apple Computer Inc. is now known as Apple Inc.

there are many rights holders who feel that they need to control the access to their works, primarily because of the ease in distributing digital data.

Enterprises are not the only entities that wish to control access to their data. Ordinary users also have a need to maintain private communications, and control data they consider sensitive. It is not just a need to control access to their health and financial data; but other data not necessarily considered generally sensitive. Private data, like home videos, holiday photos and journals may require limited distribution to close friends and family but not necessarily to far flung relatives or distribution over the Internet.

Digital rights management (DRM) refers to a set of technologies that can be used to define, manage and track rights associated with a digital object [170]. Another definition, by the Digital Media Project (DMP), is that DRM is a system that provides "*controlled communication between two or more Users*" [20]. This latter definition is closer to our preferred definition, also defined by Rosenblatt and Dykstra in [170], as a set of technologies that provide "*persistent access control"*. Thus, DRM aims to provide access control to digital resources regardless of where the data is located. In our opinion, this is the only definition that covers the scenarios we previously discussed. This definition also encompasses the functions desired in a DRM system – provide control over the use of data and provide a secure association of usage rules with the data – as discussed by Koenen et al. in [118].

DRM is an often maligned term on the Internet [8] giving rise to an Anti-DRM Day [152], while the executive editor of one of the largest technology news websites, CNet.com, labelled DRM as "*a load of C.R.A.P*" [53]. Even Microsoft chairman Bill Gates (whose company has produced a number of different DRM systems) has commented on the problems associated with current DRM systems [21]. Most of these criticisms are associated with DRM systems designed to protect copyrighted media on the Internet (referred to as *media DRM systems* from now on) while criticisms for DRM systems focused for the enterprise are hardly publicised. Furthermore, as discussed by Bott in [58], detractors of DRM do not acknowledge that limited content protection mechanisms have been in existence for a long time, including satellite videos or pay per view content on cable TV networks. These systems have shown that there are successful business models which are based on the limited distribution of content.

In [87], Gaudet outlines the differences between media DRM and Enterprise Rights Management (ERM) systems. The two differences the author outlines are:

1. DRM focuses on protection of static content, while ERM often needs to protect dynamic content, tied to business processes.

2. DRM aims to maximise economic returns for content, while ERM tries more to control content life cycle.

Gaudet seems to be favouring a separation of DRM systems, depending on the application. However, this scenario is not practical in the long run. With separate systems, users will need different DRM systems for their work, entertainment and personal data. Also, if the current scenario persists, there is the potential that different DRM systems will arise depending on the device or platform and according to the application scenarios. This means that consumers will need to utilise different applications and platforms for different protected works, causing inconvenience and thus, ultimately, unusable. For this reason, we believe that such a scenario is intractable and a common, generalised approach is required for DRM.

Furthermore, many proposals for media DRM systems aim to protect works through their entire lifecycle [63], particularly, as discussed by Byers et al. in [60], illegal versions of copyrighted works are not necessarily created and distributed by users. This means that media DRM systems will need to have many more of the features of enterprise DRM systems; thus it does not make much sense to separate the two.

Access control is a two part process: authentication of the parties involved and authorisation of the parties to access the resources. In this thesis, we discuss a framework to provide the means for persistent access control that can offer the core functionalities required by any type of DRM, but which is also flexible enough to accommodate specific use cases as required.

1.1 Users of a DRM System

As identified by Bartolini et al. [46] and also by the DMP [63], there are a number of parties involved in the DRM value chain. While the value chain as discussed by the DMP, as shown in figure 1.1, can be considered largely linear, DRM value chains should also be able to handle more complex tree-like access control hierarchies, such as the ones discussed by Sandhu et al. with regards to RBAC models in [174].

Regardless of the length or the shape of the DRM value chain, users in the value chain can be categorised into one or both of ***producer*** or ***consumer***. As a producer, the user is involved in creating or modifying the protected work. As a consumer, the user is involved in using the protected work in its current form. Thus, users such as a content provider or a service provider can be considered as consumers in the value chain, while an instantiator is both a producer and a consumer.

It is only when considering DRM systems in the view of a producer-consumer scenario, that a

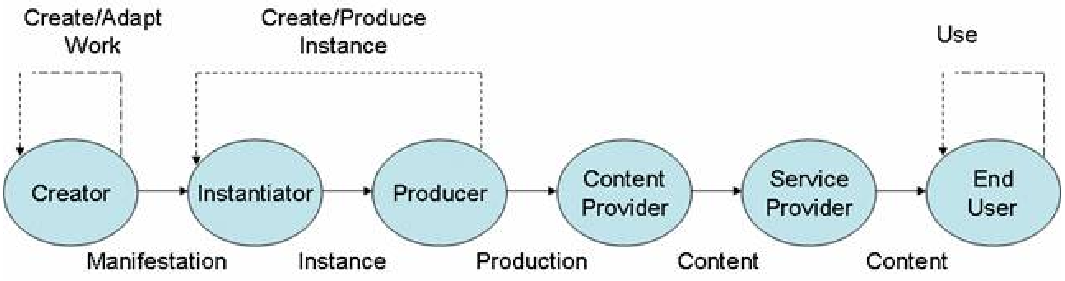


Figure 1.1: The DMP DRM Value Chain [63]

stark difference between enterprise and media DRM systems emerge. Enterprise DRM systems will, in general, consist of a lot more users who would be considered as both producers and consumers; while the majority of the users in a media DRM system will be consumers only. However, even in media DRM systems, a user can be considered a consumer of one resource but a producer of another; thus there is no reason to consider the different systems in isolation.

1.2 SCOPE

In [149], Park et al. categorised secure content distribution, as shown in figure 1.2. One such category depended on the distribution mechanism. In this dissertation, we discuss distribution agnostic DRM systems – i.e. DRM systems that do not depend on the type of distribution as part of the access control mechanism. For this reason, we do not discuss DRM with respect to streaming or broadcast media. And although we do discuss some details of CD and DVD protection mechanisms, we do not discuss systems that aim to protect such media (CDs, DVDs, Blu-Ray discs etc), since such systems are dependent on the distribution medium.

Furthermore, we discuss DRM from the view of digital data only. Since data is usually rendered in an analogue form, it is inevitable that digital data can be reproduced in some sort of analogue form. DRM provides access control of data while in its digital form, and we do not discuss access control for the data once it can be represented in analogue form.

1.3 LAYOUT

This dissertation is divided into four parts. In Part 1, comprising of chapters 2 to 4, we present what could be considered related and background work. We first detail requirements of DRM systems, extracted from a number of different sources in chapter 2. Following this, we discuss a number of existing DRM systems in chapter 3, and evaluate how well they satisfy the requirements. In chapter 4, we examine a number of key contributions to the foundations to DRM systems such as rights expression languages and the DRM reference model.

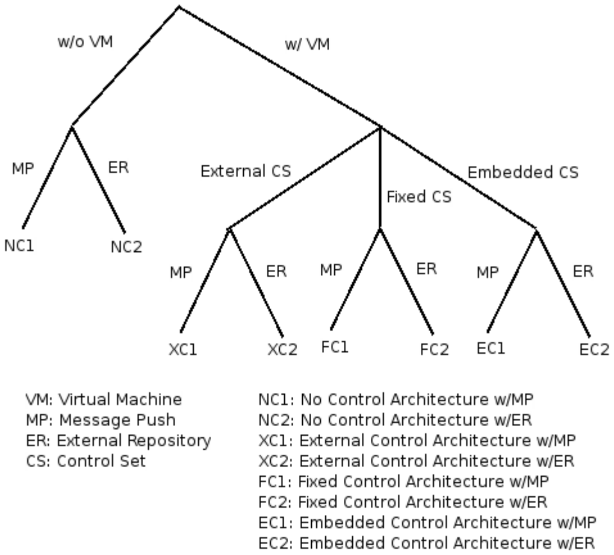

w/o VM
w/ VM
MP
ER
NC1
NC2
External CS
Fixed CS
Embedded CS
MP
ER
MP
ER
MP
ER
XC1
XC2
FC1
FC2
EC1
EC2
VM: Virtual Machine
MP: Message Push
ER: External Repository
CS: Control Set
NC1: No Control Architecture w/MP
NC2: No Control Architecture w/ER
XC1: External Control Architecture w/MP
XC2: External Control Architecture w/ER
FC1: Fixed Control Architecture w/MP
FC2: Fixed Control Architecture w/ER
EC1: Embedded Control Architecture w/MP
EC2: Embedded Control Architecture w/ER

Figure 1.2: Categorisation of DRM Systems [149]

The second part of this dissertation, is our framework itself, and we introduce the framework in chapter 5, and the detailed layout of the chapters involved. The framework is covered between chapters 6 and 12.

In the third part of the dissertation, we present a discussion of our prototype implementation of some of the key parts of our framework, followed by the conclusion, where we also conduct an evaluation of our framework using the requirements we outlined.

Part 4 is the appendices, which include a XML schema for LiREL in appendix B together with a sample data dictionary in C. Appendix D, details a complete bargaining transaction using LiREL and the sample data dictionary.

Part I

Requirements and Related Work

Chapter 2

Requirements for a General Framework for DRM

Due to the number of interested parties in DRM systems, there have been a number of publications regarding the requirements for DRM. However, there have been two major influences in the drafting of these requirements:

1. DRM systems have been developed according to the intended functions, and thus requirements have been geared specifically for the systems; and may not apply in the general case.
2. The different parties drafting these requirements have different agendas and this results in often conflicting globals set of requirements.

In this chapter, we discuss the requirements for a general framework for DRM, which we have categorised into three categories: core requirements for access control, usability requirements and legal and social requirements. We have drawn these requirements from a number of sources, which we discuss in section 2.1.

2.1 Sources for Requirements

We have examined a number of different sources to draw up the requirements discussed in this chapter. In [46], Bartolini et al. were one of the first to discuss requirements for content protection systems, and they were also one of the first to discus the parties involved in a DRM value chain. In [149], Park et al. discuss different taxonomies for content protection systems and a few technical requirements for DRM.

There are a number of different organisations that have released comprehensive technical requirements for DRM. The requirements by Networked Audiovisual Systems and Home Platforms (NAVSHP) [16], the Digital Media Project (DMP) [19] and the TIRAMISU project [124] are quite comprehensive, but all focus specifically on media DRM systems. The requirements documents, especially from the DMP have gone through many iterations and refinements, and differ drastically from their initial versions [177].

Unfortunately, technical requirements rarely acknowledge legal or social requirements for DRM systems; and when they do specify such requirements, they are often vague and even contradictory. For example, conflicting needs specified by the TIRAMISU project are the need for detailed tracking and monitoring of content usage information and the need for user privacy. We have drawn our requirements for social and legal requirements from two main sources: Mulligan et al.'s evaluation of media DRM systems on the infringement of "personal use" in [143] and the Center for Democracy & Technology's criteria for evaluating media DRM in [22].

In 2005, we conducted an online survey which aimed to investigate the respondents' usage patterns for physical media, their habits and attitudes concerning digital piracy as well as their attitude to DRM and other content protection mechanisms. There have been other surveys that investigated respondents attitude towards DRM such as the INDICARE survey [78], but we believe that our survey was the first to include existing physical media usage patterns. We published some of our important results in [33], and we have drawn up some of the requirements based on the results of our survey and the INDICARE survey.

An additional source for requirements can be found in feature descriptions of enterprise DRM systems, particularly when they describe certain use cases. In most cases, these features are already covered in the other sources mentioned above, but the use cases often bring a unique perspective on the requirements. These documents include systems from Adobe [17], Authentica [44] and Microsoft [137].

2.2 Core Requirements

For any protected resource, access should only be granted if the consumer fulfils all the requirements. Should the resource be moved to a device which cannot interpret and enforce the access control policies, no access should be given.

Requirement 1: Provide Persistent Protection: Access to the resource should only be granted when the consumer fulfils all the requirements set out by the access control policies. Furthermore, the resource should not be accessible if the device cannot interpret or enforce the access control policies.

Access control is inherently a two part process – there is an authentication process, and once the parties have been authenticated, there is an authorisation process – i.e. is the person who they claim to be, and is that person allowed access to that resource. The user in a DRM system is not necessarily the end user, but includes any person (natural or legal) involved in a DRM system, such as the rights holder or a third party processing payment.

Requirement 2: Represent User Identity: Before a user can be authenticated, the user must be identified. The user identity has to be globally unique, since DRM protection has to be applied globally. User identity is often closely associated with authentication protocols, and thus;

Requirement 3: Support multiple User Authentication Protocols: Authentication proves the validity of a claimed identity. Since there are already a number of existing secure authentication protocols, it is easier to accommodate these protocols than to create new authentication protocols.

The user is not the only party in a DRM transaction. Because DRM aims to control the use of content, there is a need to identify and authenticate data content as well as the devices which wish to render the content.

Requirement 4: Represent and Authenticate Resource Identity: To control access to a resource, the resource first needs to be identified in a globally unique scheme. Once a resource can be identified, there is also a need to verify the correctness of the association between the resource and its identity.

Requirement 5: Represent and Authenticate Device Identity: To confine access to a resource to a particular device, there needs to be a mechanism to define the identity of the device. Once a device has been identified, there is a need to verify the correctness of the association between the device and its identity.

While access control can be defined for an individual user, resource and device, it is also useful to control access through defined groups of users, resources and devices. For example, it could be useful to define location domain, such as a department in an enterprise, where data can be accessed. Furthermore, access control can also be defined though the function of the user in an organisation [85](or society) and this functionality should also be catered for in DRM.

Requirement 6: Represent and Authenticate User Groups: There is a need to control access to data to a defined group of users, and there should be a mechanism to define and authenticate such a grouping.

Requirement 7: Represent and Authenticate User Roles: There is a need to control access to data through a definition of the functionality/role played by the user in an organisation.There should be a mechanism to define and authenticate that the user belongs to the defined role.

Requirement 8: Represent and Authenticate Resource Groups: There is a need to control access to data to a defined group of resources, and there should be a mechanism to define and authenticate such a grouping.

Requirement 9: Represent and Authenticate Device Groups: There is a need to control access to data to a defined group of devices, and there should be a mechanism to define and authenticate such a grouping.

The above requirements are all involved in authenticating the parties involved in a DRM transaction. Once the parties are authenticated, authorisation for the parties is required.

Requirement 10: Represent the Authorisation (Use License): There is a need to define the authorisation for an individual person, group or role to access one or more resources on one or a group of devices. The data unit that provides such an authorisation is hereby referred to as a ***use license***

Requirement 11: Authenticate the Use License: There is also a need to ensure that the authorisation is created by the appropriate party (the rights holders or one of their representative), as well as the integrity and correctness of the use license.

Requirement 12: Support User Duties: Consumers may be required to perform certain actions before they are authorised access. For example, in a media DRM setting; the resource may have a limited free evaluation stage (e.g. shareware) and then the consumer is required to pay for further usage. Thus, the consumer would need to show proof of payment before getting authorisation for use. This scenario is not restricted to the media DRM scenario, and may also be required in an enterprise scenario; for example, a consumer may only be granted access to an application, once (s)he has passed the training for that application.

Requirement 13: Revocation of Rights: Access control rights may need to be revoked for a number of reasons such as the consumer violating the terms of the agreement or the consumer getting a different use license for the same work. Thus, there needs to be an efficient mechanism to revoke access control rights; but at the same time revocation has to be controlled in a fair manner.

Requirement 14: Update of Rights: Access control rights may need to be changed for a number of reasons such as the consumer acquiring an extension of the agreement beyond the stated period. Thus, there needs to be an efficient mechanism to update access control rights; but at the same time updates have to be controlled in a fair manner.

We contend that the requirements discussed above constitute the core requirements for any DRM system, and define the minimum requirements for persistent access control. However, there are other requirements for a general DRM systems which we are going to discuss in the remainder of this section.

2.3 USABILITY REQUIREMENTS

There are a group of requirements that, though not strictly required to provide access control, are very desirable for a DRM system. These requirements, such as portability, are demanded from the users (producers and consumers) of the system and can be termed essential features required for a DRM system, and systems that have these features are likely to have more success and attract less criticism.

Chief amongst this group of requirements is *portability*, and the lack of support for portability in DRM systems has attracted the most criticism in DRM systems in academia [143], consumer groups [22, 78], public comment [8] and the media [53, 21]. Mulligan et al. defined portability as:

> the ability to use acquired content on any suitable device, regardless of ownership in the device or its physical surroundings. Portability also refers to the ability to shift the format of a copy. [143]

We believe that portability can be distinguished into four different categories, and each category can be seen as an individual requirement for any DRM system; although the importance of the requirements can vary between different applications of DRM.

Requirement 15: Time Shifting: Time shifting refers to the ability of the DRM systems to regulate when the consumer accesses a protected work. In the media DRM space, time shifting could be used to rent works to consumers for limited time periods (similar to a movie rental or library). In the enterprise, access of corporate infrastructure at "odd hours" is already used in intrusion detection and other security infrastructure and time shifting restrictions in DRM could be used in a similar vein.

Requirement 16: Format Shifting: Format shifting refers to the ability of the DRM systems to regulate the consumer's ability to change the format of the data file (without affecting the access control rules), and Mulligan et al. argued that it is an important portability issue in media DRM systems [143]. Format shifting could be important in an enterprise for similar reasons – for example, the enterprise could keep internal data stored in a certain format and in a different format when released to other companies or even to the public (in the case of financial statements for example). Format shifting should also allow for easier integration between different applications across different platforms.

Requirement 17: Space Shifting: Space shifting refers to the ability of the DRM systems to regulate which devices the consumer can use to access the protected resource. Space shifting operates within the same type of device and operating system.

Requirement 18: Platform Shifting: Platform shifting refers to the availability of a DRM system across multiple devices and operating systems. In a world with multiple types of convergent devices, users can expect to access protected resources on mobile phones, portable computers, desktop computers and hand-held computers to name a few. Each of these devices also support a number of different operating systems, sometimes from the same vendor. Thus, platform shifting refers to the DRM systems ability to support multiple operating systems and devices.

For portability, DRM systems need to be able to support the regulation of time shifting, format shifting and space shifting while implementations of DRM systems need to support the implementation of platform shifting. There are requirements other than portability which can be grouped under usability requirements, and we discuss them next.

Requirement 19: Integration with Existing Applications: Current DRM systems introduce their own applications in order to enforce the access control policies. This strategy is however not feasible in the long term: in the media DRM space, consumers want the choice for which applications they want to use, because of familiarity or features provided by these applications. Similarly, enterprises often use software developed for their specific needs and resources that need to be protected are not necessarily generated by applications supplied by current vendors of DRM systems.

Requirement 20: Delegation of Rights: At least one user has to have the right to delegate rights to other users (e.g. an author has to be able to delegate the right to view to a reader); but there is a need to control delegation of rights. We are however interested in delegation of rights after the initial delegation, and if possible, all aspects of delegation must be possible. Delegation of rights could also include the transfer of rights (either temporary or permanent) between consumers of a work. A permanent transfer of rights can effectively be seen as a revocation followed by a granting of rights, without the rights holder exerting additional payment (or other duties).

Requirement 21: Fine Grained and Flexible Access Control Specification: The use cases for DRM systems in general demand fine grained access control – not just control on the traditional read, write and execute but also application specific controls. Furthermore, restrictions on specific controls would also be preferred. Because of the variety of use cases for DRM, there is also a need to create flexible access control specification, with the definition of the controls differing according to the use cases.

Requirement 22: Tracking and Monitoring: Although not a feature of DRM itself, monitoring the access and usage patterns of users (both consumers and producers) can be easily achieved. In an enterprise, monitoring the usage of confidential data is crucial, and should compromises to security take place, access logs and usage patterns would be very useful in tracking down the source of the compromise.

However, with the ability to track and monitor usage, privacy does become a concern. Monitoring employee activity in the workplace is already a contentious issue, while monitoring consumers in media DRM systems could be illegal in some countries, and already attract criticism in academia [143] and the public [8]. Thus, the ability to track and monitor usage is only desired in certain applications of DRM, and care needs to be taken to stay within the boundaries of appropriate monitoring.

Requirement 23: Offline Usage: Communication networks are not perfect, and there are many situations where consumers (or even producers) may not have access to the Internet (for example on an aeroplane), or may need to access resources on devices which have no connectivity (such as the current generation of Apple iPods). Thus offline usage is desirable; but does have its drawbacks – offline usage reduces monitoring and tracking capabilities; and may also limit the control desired. For example in an enterprise, if an employee is fired and the employee has protected data that can be accessed offline, the employee could still retain access to the protected data. Generally, online usage could be required periodically but not necessarily all the time for operation

2.4 Legal and Social Requirements

Ultimately, transactions regulated in a DRM system are governed by applicable laws. When disputes do occur, the only fair solution to the disputes can be found through arbitration or the courts. However, current DRM systems (and arguably most computer systems) do not consider the legal implications of the services they provide; nor do they have clearly identifiable legal frameworks under which they operate.

Requirement 24: A Legal Framework for DRM: There is a need to identify and position DRM systems, and the transactions in a DRM system, in a comprehensive legal framework. The legal framework should address concerns relating to copyright law and fair use and personal uses such as portability and archiving.

Consumer organisations have also commented on some of the requirements for media DRM systems with respect to features consumers expect from DRM systems. Some of these requirements can be addressed as part of a legal framework, while other requirements do not necessarily have any legal backing. These requirements also apply (although at times, to a lesser degree) to enterprise DRM systems.

Requirement 25: Transparency: DRM systems should clearly disclose the restrictions imposed. The restrictions should be available to the user before the resource is acquired, and the user should also have the means to view the restrictions during use. Furthermore, the restrictions should be clearly stated and documented, and not hidden away in complex agreements and user agreements.

Requirement 26: Privacy and Anonymity: The requirement for privacy and anonymity, goes against a previously stated requirement for monitoring and tracking. While both can coexist in the same system, a system that provides tracking and monitoring cannot claim to provide complete user privacy and vice versa. However, some parts of both are required for generalised DRM systems, and the range of the features implemented will depend on the exact application.

If monitoring and tracking are implemented, the user must be informed of what data is being collected, how the collected data is going to be used, who will have access to the collected data and how long the collected data will be stored.

Requirement 27: Do Not Alter Platform Functionality and Performance: Implementation of a DRM system should not drastically alter the functionality or performance of the whole system for non-protected works. This includes the introduction of security loopholes through the DRM system, or the need for additional software or hardware to regain functionality or performance.

2.5 Summary

In this chapter we have discussed 27 requirements for a generalised DRM systems, which we have drawn from a broad number of different sources. These requirements are categorised into three categories: the core requirements for access control, usability requirements and the legal and social requirements. We have discussed each requirement in detail, together with our motivations regarding their importance in DRM systems. As the categories imply, not all of the requirements are necessary to achieve persistent access control, but, these additional requirements provide a greater usability of the DRM system for all the parties involved.

CHAPTER 3

A DISCUSSION OF EXISTING DRM SYSTEMS

Before creating a comprehensive framework for DRM, there is a need to identify the successes and problems present in existing solutions. In particular, there is a need to identify the various areas which hinder the evolution of existing systems to become a general DRM system.

In this chapter, we discuss a number of different DRM systems, both media and enterprise. For each system, we give an overview of the players involved in the system, how protected files are assembled and how access controls are enforced. We also conduct a requirement analysis of the systems using the requirements we have already discussed in chapter 2. For the requirement analysis, we use a simple 3 point rating system for each requirement, defined below. The ratings are then used to compare different systems, and identify which requirements are least satisfied by current DRM systems.

0 : The requirement is not supported at all.

1 : There is limited support for the requirement.

2 : The requirement is supported in full.

The requirement analysis is done with respect to the DRM system itself and what is possible with the DRM system, and not what is expected by consumers or consumer organisations. Certain requirements could be considered fully met in design but have no support in implementations. In these cases, we often score the systems as limited support, especially if the decisions behind non-implementation are driven by the vendor's business model and practices; and not because of adequate market demand.

3.1 Apple iTunes Music Store

In March 2003, Apple Computers debuted the Apple iTunes Music Store[1] which allowed customers to download individual music tracks for 99 US cents, and whole albums for US $9.99. Within one year, iTunes Music Store had become the dominant online music store, selling over 50 million songs [66] and it currently commands the majority of legal online music downloads [91]. With over 2 Billion downloaded songs, 50 million TV episodes and 1.3 Million movie downloads, it can be considered to be the most successful DRM system [28].

Apple's iTunes Music Service is a proprietary system, and the details we present below are compiled from the user documentation provided by Apple [26], and from some well known compromises of the system – Playfair (subsequently renamed Hymn) [1, 11, 12] and PyMusique [193, 166] and its successor SharpMusique [111].

3.1.1 Brief Overview

iTunes DRM system can be considered as a set of two services. Initially, iTunes started as a music player system for Apple Macintosh, and later became the interface between the Apple Macintosh and their portable music player, the iPod. iTunes (the software) has evolved to become a comprehensive media player, and is available on Microsoft Windows and Apple Mac OS operating systems. The second part of the system is the iTunes Music store, which sells music, audio books, movies and TV episodes. All media sold on the music store is protected by the Apple DRM system known as FairPlay, which is the focus of our discussion.

When a user first registers in the iTunes Music Store, a digital certificate, including a key, is generated for that machine. We believe that this certificate is signed by the iTunes Music Store. Currently, the user can register five machines in total, which would all carry the same digital certificate. It is possible to deregister the machine (thereby removing the key from the key store), but it is not possible to register beyond the maximum – even if those machines are no longer registered.

When protected data (currently music and video) is transfered to a portable device (currently only the Apple iPod is supported), the digital certificate associated with the data is also transfered. It is therefore possible to utilise data from different users on the portable device.

There are no other keys or digital certificates involved in the iTunes Music Service, and thus Apple manages to keep a very tight control over the chain of trust. However, the short chain (only two types of keys) introduces its own problems, primarily with regards to portability.

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

Using the primary key (on the iTunes service) provides no user authentication; and thus any data that is protected with the key is accessible to every registered user. Thus, the iTunes service utilises the user key to protect content, requiring user authentication to access data. This means that portability between different users is not possible; a limitation which will affect every user who reaches their maximum limit on registered computers.

A flaw in the system, uncovered by PyMusique, is that the protection of content does not occur on the iTunes server; presumably because it would require the iTunes service to keep track of all the keys. Instead, the protection is only applied once the data is downloaded to the registered computer. PyMusique and SharpMusique took advantage of this flaw, and interacted with the iTunes music server directly, but did not protect the data once it was downloaded. In late 2006, the protocol used by iTunes to interact with iTunes Music Store changed, and SharpMusique no longer works [111].

Hymn (also known as PlayFair) is a more direct attack on the service, where the user's key is recovered from the portable device and used to decrypt the protected content. Once the data is decrypted, the user can do what they wish with the content.

3.1.2 Core Requirement Analysis

FairPlay cannot be considered a complete DRM system. Its primary protection relies on secure distribution, which has already been compromised. Furthermore, the user identity system is basic, and two different users cannot be accommodated on the same iTunes system directly (it is possible to cater for two different users if they use different user profiles on the consuming computer's operating system). Furthermore, even though Apple is introducing iTunes enabled media into other areas of the home through products such as Apple TV[2], mechanisms to manage device or user groups do not exist. And even though music is sold as albums, songs are effectively treated as individual and not as a group of resources. This could be due to the fact that the system does not make use of any licensing structure. Revocation of rights is also poorly supported, and only allows the complete revocation of user's rights and not selective revocation. The rights profile can be updated (and have changed since the inception of the store), but requires an update to the iTunes software, and users who do not update their software do not get the updated rights profile. Table 3.1 provides a summary of our core ratings.

3.1.3 Usability Requirements Analysis

Portability is often seen as a major consumer problem with DRM and FairPlay scores interesting ratings with respects to portability. FairPlay currently does not have any time shifting restric-

[2] http://www.apple.com/appletv/

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	1
3	Support Multiple User Authentication Protocols	0
4	Represent and Authenticate Resource Identity	n/a
5	Represent and Authenticate Device Identity	2
6	Represent and Authenticate User Groups	0
7	Represent and Authenticate User Roles	0
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	0
10	Represent the Authorisation (Use License)	0
11	Authenticate the Use License	0
12	Support User Duties	0
13	Revocation of Rights	1
14	Update of Rights	1

Table 3.1: Core Requirement Analysis for iTunes FairPlay

tions imposed on the user – once the user downloads a song or video clip, they can render it however often they wish. The subscription model, which requires time shifting is not supported in FairPlay, although rumours of such as a service have been reported by the media [59, 117]. Since we do not have access to any internal documentation on FairPlay, we assume that Time Shifting is simply not supported, and future support will require an update of the iTunes service.

FairPlay is openly acknowledged as a DRM wrapper, thus there is no reason why multiple format types cannot be accommodated, or why format conversion cannot be performed between these different formats. However, only two formats are currently implemented: AAC for audio and Quicktime Video for video. Likewise, limited platform portability is supported through the evidence of support for different operating systems and the iPod, but the support remains tightly controlled. Space shifting however is well supported.

Apart from Offline Usage, where FairPlay does not require any Internet connection other than for initial acquisition of media, other usability requirements are not catered for by FairPlay. iTunes does not support delegation of rights, and the lack of licensing structure means that there are no fine grained access controls. With the tight integration of iTunes software, other applications are ignored and the iTunes music store claims that it does not track or monitor usage of protected works.

A summary of the usability requirement ratings is given in table 3.2.

	Requirement	**Rating**
15	Time Shifting	0
16	Format Shifting	1
17	Space Shifting	2
18	Platform Shifting	1
19	Integration with Existing Applications	0
20	Delegation of Rights	0
21	Fine Grained and Flexible Access Control Specification	0
22	Tracking and Monitoring	0
23	Offline Usage	2

Table 3.2: Usability Requirement Analysis for iTunes FairPlay

3.1.4 Legal and Social Usability Analysis

While it is well known that the iTunes Music store sells media to the general public, the exact legal position of the store is, at best unknown, and has attracted legal attacks in some countries like Norway [159], which would like to force portability of FairPlay beyond the iPod. If iTunes sells media under copyright law, then a number of functions allowed under copyright are not permitted and that means it can be considered as illegal. If it is under a licensing system, then do they fulfil the requirements necessary for concluding valid contracts? Thus, with an unclear and unknown legal position, it scores a 0 rating for the legal framework rating.

iTunes Music store is very transparent in identifying all the restrictions placed by iTunes, but does not provide much detail on how much personal data is collected (sales records etc) or if data is correlated to keep track of general trends. Apple does have a comprehensive privacy policy, and is easily available to all users [27], and thus our higher score. Finally, studies have shown that battery life of portable media players diminishes faster when playing DRM enabled media when compared to non DRM enabled media [116], but that is understandable considering the extra CPU operations required to render the media. Apple iPod had one of the lowest reported degradation in performance, and hence the full rating. Our ratings are summarised in table 3.3.

	Requirement	**Rating**
24	A Legal Framework for DRM	0
25	Transparency	2
26	Privacy and Anonymity	2
27	Do Not Alter Platform Functionality and Performance	2

Table 3.3: Legal and Social Requirement Analysis for iTunes FairPlay

3.2 Microsoft Windows Media (WM)

Recent versions of Microsoft's Windows Media (WM) Audio and Windows Media Video were designed to compete with iTunes Music Store and thus feature DRM protection. Unlike Apple, Microsoft's initial strategy was to support multiple online music stores and multiple portable media players. But, even though the formats are supported by one vendor; different versions support different features and are often incompatible. Microsoft's first release, Windows Media 9, did not support automatic time shifting expiry and was confined to online connectivity only [127]. This was solved in a subsequent release, "Play For Sure". This is possibly the most widely used version, and is supported by many device manufacturers and online music stores. More recently, Microsoft released yet another specification, which is incompatible with "Play For Sure", and geared towards its own portable device player, Zune [3, 181]. With this strategy, it seems that Microsoft is trying to emulate Apple's tight integration strategy.

In this section, we primarily discuss "Play For Sure", although all three versions use a similar architecture. Like iTunes, Windows Media is a proprietary solution, although there is a lot more documentation available on its operation than iTunes' operation. We use documentation released by Microsoft [136, 140, 133, 134, 156, 135, 155] and reports on the breaking of the protection (through a tool FairUse4WM) in [57] and the original details posted in [76] and an earlier attack using a tool DrmDbg [75].

3.2.1 Brief Overview

Unlike Apple, Microsoft has chosen not to control the distribution and licensing components of the DRM system. Furthermore, because the playback libraries can be licensed, theoretically, playback is possible in a wide range of platforms and devices.

One of the key features of the Microsoft approach is the default support for super distribution – it does not matter how the consumer acquires the media, but to access the media, the consumer will require a license. However, the licenses are non transferable, and usually tied to the device. This approach severely limits portability, as consumers cannot use multiple devices even if the consumer owns these devices.

Unlike iTunes, DRM protected media files are encrypted before distribution. The key to access these files are distributed through the use license. Licenses also carry a revocation list of consuming devices that should be refused access to the protected media. Because the licenses are locked to the consuming device, there is no need for user authentication. Unfortunately, neither the public documentation nor the break indicate conclusively whether the use license itself is encrypted.

The documentation does confirm that the use licenses are signed by the licensors. The licensors' root certificates in turn have to be signed by Microsoft directly. This means, that while, Microsoft may not control the entire DRM value chain, they still control the chain of trust. This also means that Microsoft ultimately can decide which licensors to trust, and can refuse licensors a signed certificate, thereby refusing them access to utilise the Windows Media DRM system.

Like Hymn, FairUse4WM (based on an earlier attack DrmDbg) is a direct attack, extracting the key(s) from the licenses and then extracting the content from the protected package.

3.2.2 **Core Requirement Analysis**

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	0
3	Support Multiple User Authentication Protocols	0
4	Represent and Authenticate Resource Identity	n/a
5	Represent and Authenticate Device Identity	2
6	Represent and Authenticate User Groups	0
7	Represent and Authenticate User Roles	0
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	0
10	Represent the Authorisation (Use License)	2
11	Authenticate the Use License	1
12	Support User Duties	0
13	Revocation of Rights	1
14	Update of Rights	1

Table 3.4: Core Requirement Analysis for Windows Media

Since protection offered by WM is not distribution dependent, it offers a more persistent access control protection. However, like iTunes FairPlay, there is no provision for individual user identity, and instead user identity is replaced by device identity. Also, like iTunes FairPlay, there is no provision to identify groups of devices, resources or users; and revocation of rights is tied to the device, and not individual rights or resources. Unlike iTunes FairPlay, WM does make use of use licenses and these use licenses are authenticated through the digital signature of the issuer. However, there does not seem to be any mechanism to verify the authenticity of the issuer (whether it is still trusted for example) or whether the license issuer can differ from the actual packager of the media (i.e. can an artist package the media him(her)self and then use music stores to market and sell licenses?). Rights can be updated through the issuing of a new license, but we are not aware of any media store that supports such a practice. Since current revocations are based on device rather than license, we are not sure if license revocation is possible. Our core requirement analysis ratings are summarised in table 3.4.

3.2.3 Usability Requirement Analysis

Initially, WM was used exclusive in subscription business models, and thus time shifting was a key feature, although this initially required online connectivity. Like iTunes FairPlay, WM uses a wrapper and thus it potentially can support other media formats. However, no other implementations exist, and thus cannot be verified any further. Unlike Apple, Microsoft has been more liberal in allowing WM to be incorporated into other devices and platforms. However, the recent change with Zune, and because of the fact that Microsoft can still refuse to license to any vendor they wish, we have lowered the platform shifting scores. Since WM is based on a set of libraries, it is possible to render media on compatible software that makes use of the libraries, although not all applications may work. Because media licenses are device specific, space shifting is not possible under most schemes.

Even though WM does feature a licensing mechanism, the lack of individual users make fine grained access control difficult, and this difficulty is compounded by a relatively small set of access control primitives. Without user identification, delegation is also not possible. WM does support tracking and monitoring support, but the extent is not documented; and both “Play For Sure” and Zune versions support offline usage. Our usability requirement analysis ratings are summarised in table 3.5.

	Requirement	**Rating**
15	Time Shifting	2
16	Format Shifting	1
17	Space Shifting	0
18	Platform Shifting	1
19	Integration with Existing Applications	1
20	Delegation of Rights	0
21	Fine Grained and Flexible Access Control Specification	1
22	Tracking and Monitoring	n/a
23	Offline Usage	2

Table 3.5: Usability Requirement Analysis for Windows Media

3.2.4 Legal and Social Requirement Analysis

As with Apple iTunes FairPlay, there is no legal certainty to the legal position of online media stores using WM. Furthermore, because individual media stores set their own privacy policies, and their own disclosure rules, we cannot rate these factors. With regard to performance and battery life, the reviews mentioned earlier found that WM enabled devices fared the worst, losing battery life up to 25% faster [116]. This is significantly higher than the iPod and thus our

	Requirement	Rating
24	A Legal Framework for DRM	0
25	Transparency	n/a
26	Privacy and Anonymity	n/a
27	Do Not Alter Platform Functionality and Performance	1

Table 3.6: Legal and Social Requirement Analysis for Windows Media

lower rating. Our legal and social requirement analysis ratings are summarised in table 3.6.

3.3 HELIX DRM

Real Network's Helix DRM is built on the Helix platform, a multi-format, multi-platform, media platform; co-developed with the Helix Community, an open source initiative. Large parts of the Helix platform are available as open source, but the DRM component, while integrating with the larger Helix platform, remains proprietary. Helix DRM does cater support for both Windows Media and iTunes fair play [163], but has been largely unsuccessful. The information we present in this section is summarised from documentation released by Real Networks[3] [145, 162], Michiels et al.'s brief discussion in [132] and some of the Helix Community documentation including [101, 102, 100, 99].

3.3.1 Brief Overview

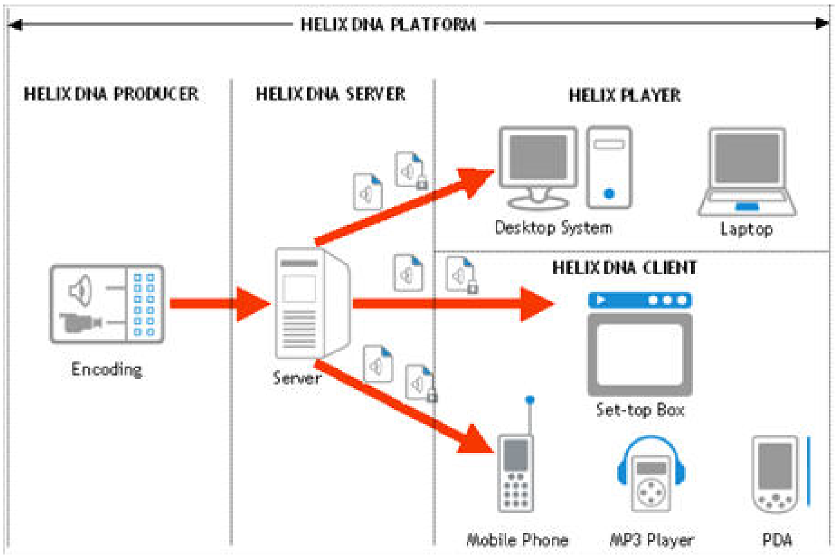


Figure 3.1: Overview of the Helix Platform [101]

As illustrated in figure 3.1, the Helix Platform consists of three players: a producer to create encoded content, a server to distribute content and clients that render the content. Helix DRM

[3] http://www.real.com

system adds a further player – Helix DRM License Server – while the other components of the Helix DRM system; the Helix DRM Packager and the Helix DRM Client fit into the existing Helix Platform players producer and client respectively.

Helix DRM does not depend on distribution as a mechanism to enforce access control, and instead, like Windows Media, supports acquisition of licenses for protected data regardless of how the user actually acquired the data. Helix DRM also provides protection for streaming media, which provides distribution dependent access control. The use licenses contain keys to decrypt the protected data, but further key management information was not available.

3.3.2 Core Requirement Analysis

Helix DRM clients can be categorised in two ways: primary devices, which have connections to the Internet and are responsible for the acquisition and management of licenses; and secondary devices, which are largely offline devices that depend on primary devices to manage both the data and licenses. Thus, Helix DRM system does have quite a comprehensive device management system. There is no information on user management, and the Helix Platform does not have any user management components. We therefore assume that user management functionality is provided simply by the licenses themselves and additional user management functionality does not exist.

There is no information in the documentation we are aware of with regards to revocation or change of rights; although the issuing of a new license should allow the user to acquire a different rights profile. We are also unsure of whether different service providers need to create their own clients in order to create the chain of trust; or whether clients can interoperate between different service providers. In the later case, a secure way to authenticate the license issuer is required; and there is no documentation available on how Helix DRM achieves this. Our ratings for Helix DRM's core requirements are summarised in table 3.7.

3.3.3 Usability Requirement Analysis

Since the Helix DRM supports a number of formats, format shifting could be implemented at the client interface. However, no such mechanism exists. Likewise, the strong emphasis on the number of platforms supported allows for a strong rating for platform shifting. Rights expressions are not predefined and Helix DRM encourages flexible access control specifications. Although logging does form a part of the Helix Platform, the documentation suggests that such functionality is for error detection [102] and not for tracking and monitoring usage.

Helix DRM requires its own clients, or significant patches to existing clients before they can be

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	0
3	Support Multiple User Authentication Protocols	0
4	Represent and Authenticate Resource Identity	n/a
5	Represent and Authenticate Device Identity	2
6	Represent and Authenticate User Groups	0
7	Represent and Authenticate User Roles	0
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	2
10	Represent the Authorisation (Use License)	2
11	Authenticate the Use License	n/a
12	Support User Duties	0
13	Revocation of Rights	0
14	Update of Rights	1

Table 3.7: Core Requirement Analysis for Helix DRM

used with the system. Thus, it does not offer any seamless integration with existing applications. The lack of user management also means that there is no posibility of rights delegation between users. Our usability requirement analysis ratings are summarised in table 3.8.

	Requirement	**Rating**
15	Time Shifting	2
16	Format Shifting	1
17	Space Shifting	2
18	Platform Shifting	2
19	Integration with Existing Applications	0
20	Delegation of Rights	0
21	Fine Grained and Flexible Access Control Specification	2
22	Tracking and Monitoring	0
23	Offline Usage	2

Table 3.8: Usability Requirement Analysis for Helix DRM

3.3.4 Legal and Social Requirement Analysis

Helix DRM is hard to analyse for legal and social requirements as these are dependent on the implementations of the systems. Real Networks does not provide any legal framework, and there is no performance analysis of devices that support Helix DRM.

	Requirement	Rating
24	A Legal Framework for DRM	0
25	Transparency	n/a
26	Privacy and Anonymity	n/a
27	Do Not Alter Platform Functionality and Performance	n/a

Table 3.9: Legal and Social Requirement Analysis for Helix DRM

3.4 Open Mobile Alliance (OMA) DRM

The Open Mobile Alliance (OMA) is a consortium of parties interested in the mobile communications arena including service providers, network companies and equipment manufacturers. The OMA specifications are often implemented by its members, which ensures compatibility across the globe. OMA DRM was developed to provide DRM for all mobile phones and can be considered as one of the largest deployments of DRM systems, and currently, the only standardised DRM platform. The information in this section is summarised from both the OMA 1.0 and 2.0 specifications [146, 147]. OMA DRM 1 is widely deployed, and even though a stable version of OMA DRM 2.0 was released in mid 2006, it has not been widely deployed.

3.4.1 Brief Overview

The OMA DRM 1 specifications cater for three types of DRM specifications, increasing in flexibility and complexity in its implementation, as shown in figure 3.2 and discussed below. However, only the last type, *separate delivery*, caters for delivery agnostic DRM.

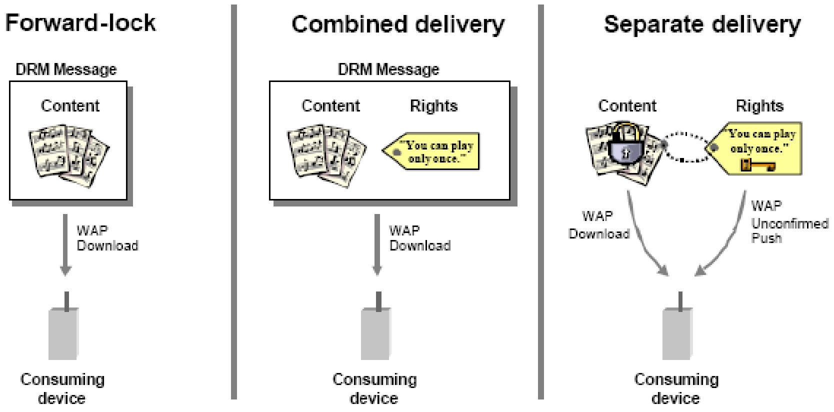


Figure 3.2: DRM Types in OMA 1.0 [146]

1. **Forward Lock:** The protected content is locked to the consuming device and can be only used by the consuming device. There is a very limited set of rules (rights) that are

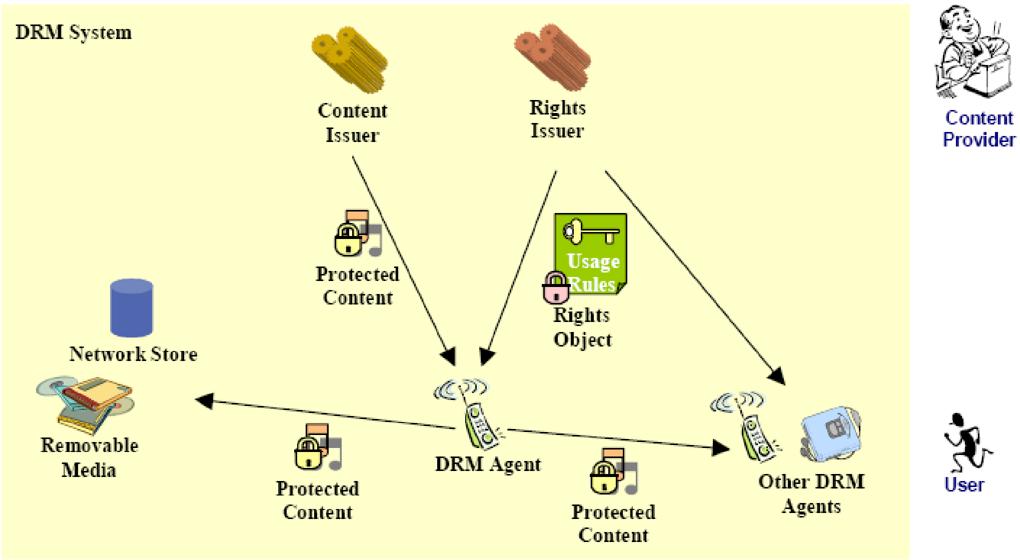


Figure 3.3: Overview of OMA DRM 2 Architecture [147]

implemented, and these are standard for all consuming devices implementing OMA DRM 1.

2. **Combined Delivery:** Like Forward Lock, the content is locked to the consuming device. However, unlike Forward Lock, there is flexibility in the rules (rights) that can be implemented, giving the rights holders a lot more flexibility.

3. **Separate Delivery:** The content is not locked to the consuming device, but to access the content, a separate use license needs to be acquired. The license is usually locked to the consuming device. It is only in this scenario that the content is encrypted. Other modes all use unencrypted content.

OMA DRM 2, attempts to address the entire content lifecycle, and an overview of the architecture is shown in figure 3.3. Unlike OMA DRM 1, only encrypted content is used and a separation of licensing and content is preferred. In many respects, OMA DRM 2.0 builds on the separate delivery approach of OMA DRM 1. There are other differences between OMA DRM 1 and OMA DRM 2, including the use of device domains and key management.

3.4.2 Core Requirement Analysis

Although OMA DRM 2 does have some discussion on user authentication, user authentication is not explicitly supported in OMA DRM 2, and it did not feature in OMA DRM 1 either. In OMA DRM 2, user identity was discussed as part of the architecture, but was left for future development, and not implemented. In fact, protection in OMA DRM has always revolved around the device; and mobile devices already feature strong identity systems. OMA DRM 2 further enhances this position with the addition of device domains (where devices are grouped together). Like many other systems, revocation of rights require the revocation of the device; and rights to individual resources is not supported. Resources feature unique identifiers but there is no verification support for these identifiers.

	Requirement	**OMA DRM 1**	**OMA DRM 2**
1	Provide Persistent Protection	2	2
2	Represent User Identity	0	0
3	Support Multiple User Authentication Protocols	0	0
4	Represent and Authenticate Resource Identity	1	1
5	Represent and Authenticate Device Identity	2	2
6	Represent and Authenticate User Groups	0	0
7	Represent and Authenticate User Roles	0	0
8	Represent and Authenticate Resource Groups	0	0
9	Represent and Authenticate Device Groups	0	2
10	Represent the Authorisation (Use License)	2	2
11	Authenticate the Use License	2	2
12	Support User Duties	0	0
13	Revocation of Rights	1	1
14	Update of Rights	0	0

Table 3.10: Core Requirement Analysis for OMA DRM 1 and OMA DRM 2

3.4.3 Usability Requirement Analysis

In OMA DRM 1, portability did not really exist, and data was bound to the device. This scenario was slightly improved in OMA DRM 2, where the data can be bound to the domain of devices instead of the device itself. However, since there is no user management by default, we are unsure how portability between devices not in the same domain can be achieved. OMA DRM caters for any type of data, but there is no mechanism to shift the format once the data has been protected.

Since OMA DRM is implemented at the device level and not completely in software, special software is not necessarily required for its use. In OMA DRM 2, it is possible to track the distribution of the protected content through an element in the file format. Thus, limited tracking ability is possible, but full usage monitoring is not possible. The remaining usability requirement analysis is summarised in table 3.11.

	Requirement	**OMA DRM 1**	**OMA DRM 2**
15	Time Shifting	0	2
16	Format Shifting	1	1
17	Space Shifting	0	1
18	Platform Shifting	0	1
19	Integration with Existing Applications	2	2
20	Delegation of Rights	0	0
21	Fine Grained and Flexible Access Control Spec.	1	2
22	Tracking and Monitoring	0	1
23	Offline Usage	2	2

Table 3.11: Usability Requirement Analysis for OMA DRM 1 and OMA DRM 2

3.4.4 Legal and Social Requirement Analysis

Since OMA DRM is a wide specification and not an individual implementation, it is difficult to analyse this set of requirements. As we discuss later in section 6.3.2, there are implementations of OMA DRM 1 systems that can be considered illegal. However, neither OMA DRM 1 nor OMA DRM 2 provide any discussion on the legal framework under which they aim to operate nor do they advise potential implementors the social and legal requirements that need to be considered by the implementors.

	Requirement	**OMA DRM 1**	**OMA DRM 2**
24	A Legal Framework for DRM	0	0
25	Transparency	n/a	n/a
26	Privacy and Anonymity	n/a	n/a
27	Do Not Alter Platform Functionality and Performance	n/a	n/a

Table 3.12: Legal and Social Requirement Analysis for OMA DRM 1 and OMA DRM 2

3.5 Adobe Document Security

Adobe's DRM system was one of the earliest implementations in the consumer space. Originally (and still) used for Adobe's eBook format, Adobe's DRM system is now also available to enterprises. Unlike some of the DRM systems mentioned previously, Adobe Document Security provides protection for only one file format – PDF. But, the DRM system is the only system that can claim to support a wide range of platforms, from portable devices to different desktop architectures and operating systems. The details of the system are summarised from public documents published by Adobe [17, 13, 14].

3.5.1 Brief Overview

Adobe's DRM solution has two components – a server to create secure documents, and Adobe's popular PDF rendering application: Acrobat Reader. Note that, not all versions of the reader can access protected files, but Adobe does have capable readers for a number of different platforms. Documents are encrypted using symmetric keys, and these keys are protected in the license using the user's public key. The user's private key serves as the authentication mechanism, as well as the means to extract the symmetric key to decrypt content. It is therefore possible to use super-distribution since use licenses are separated from the content.

Adobe does not retain control over the chain of trust – Acrobat Readers are capable of installing new certificate authorities, and trusted users; which means that there is no reason to enforce any restrictions for creating secure documents. Adobe's approach does have one flaw – there is no restriction on the user distributing multiple copies of his/her private keys across the Internet; thus allowing more than one person access to the protected data. To counter this, Adobe's system allows for tracking usage and access of the protected data.

3.5.2 Core Requirement Analysis

Access control is not governed by the distribution of the data, and access to protected data is not confined to a defined boundary; and thus persistent access control is achieved. The system does not regulate device management, and while digital certificates provide user identity control, this control can be replicated and does not provide any advanced identity system functionality such as groups or roles. Revocation of licenses, once issued, is virtually impossible as the licenses can easily be replicated and restored if revoked. The system does have mechanisms to update rights through issuing new licenses (which most probably invalidate existing licenses). Our core requirement analysis ratings are summarised in table 3.13.

3.5.3 Usability Requirement Analysis

Adobe provides protection on only one document format – PDF, and thus provides no means for format shifting, without loosing the protections. However, the system scores very well on other portability factors such as time shifting, space shifting and platform shifting. Because almost any document type can be converted into PDF, the system provides for easy integration to existing systems; although the rendering is restricted to only one application. The system provides for restricting any functionality provided by Adobe Reader, and thus scores well in this respect. Our usability requirement analysis ratings are summarised in table 3.14.

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	1
3	Support Multiple User Authentication Protocols	0
4	Represent and Authenticate Resource Identity	n/a
5	Represent and Authenticate Device Identity	0
6	Represent and Authenticate User Groups	0
7	Represent and Authenticate User Roles	0
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	0
10	Represent the Authorisation (Use License)	2
11	Authenticate the Use License	2
12	Support User Duties	0
13	Revocation of Rights	0
14	Update of Rights	2

Table 3.13: Core Requirement Analysis for Adobe Document Security

	Requirement	**Rating**
15	Time Shifting	2
16	Format Shifting	0
17	Space Shifting	2
18	Platform Shifting	2
19	Integration with Existing Applications	1
20	Delegation of Rights	0
21	Fine Grained and Flexible Access Control Specification	2
22	Tracking and Monitoring	2
23	Offline Usage	2

Table 3.14: Usability Requirement Analysis for Adobe Document Security

3.5.4 Legal and Social Requirement Analysis

Unlike media DRM systems, enterprise DRM systems are on a sounder legal footing, as they regulate the control of corporate data, often within a closed boundary. However, how sound their legal footing is, is unknown, especially when one considers different legislations with regards to employee privacy, monitoring of employees and employee access to information. Furthermore, Adobe Document Security can be used as a media DRM system, but then the system's legal footing needs to be examined in terms of the implementation. In terms of transparency, Adobe Reader can easily display all the restrictions placed on the data. There is no performance data therefore this cannot be rated. The ratings for the legal and social requirement analysis are summarised in table 3.15.

	Requirement	Rating
24	A Legal Framework for DRM	1
25	Transparency	2
26	Privacy and Anonymity	n/a
27	Do Not Alter Platform Functionality and Performance	n/a

Table 3.15: Legal and Social Requirement Analysis for Adobe Document Security

3.6 Authentica's Active Rights Management Platform (ARM)

Authentica markets itself as the "leader in enterprise rights management (ERM) solutions" [43], and there are a number of companies that provide services similar to what is provided by Authentica. Authentica (and their competitors) all provide proprietary solutions, and the details we present here are summarised from four documents published by Authentica: [43, 42, 44, 73][4] as well as a comparative analysis by Rosenblatt in [169], which contrasts Authentica's solution with Microsoft's Rights Management Services (discussed in chapter 3.7). The later report was commissioned by Authentica, and while the comparison may be biased, it does shed light on the inner workings of the product.

3.6.1 Brief Overview

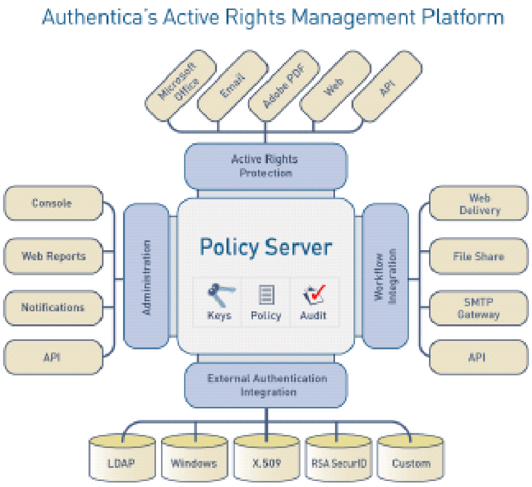


Figure 3.4: Overview of Authentica Active Rights Management Platform [44]

ARM offers rights management for selected versions of Microsoft Office, Adobe PDF, Microsoft Outlook, Lotus Notes, Microsoft Internet Explorer and Netscape browsers. Effectively, the clients are patches for these products and provide enforcement of the access control. Other

[4] [44] seems to be a more recent version of [73]. [73] is no longer directly accessible on the Authentica website.

components of ARM include a policy server and plug-ins for authentication and administration. An overview of the solution is provided in figure 3.4. When protecting data, the data is wrapped in a cryptographic envelope (using AES encryption), and with the controls embedded into the file. User authentication is controlled through a number of different schemes, and once the user is authenticated; access control is applied to various aspects of the document.

3.6.2 **Core Requirement Analysis**

ARM is not distribution dependent, and once a protected package is created, it can be distributed in any manner. ARM relies on external user identity management, and can support a number of different schemes. Thus, if schemes support roles and user groups, then these can easily be incorporated into ARM. However, ARM has no controls based on device, and does not offer any access control functionalities based on the target device. Revocation and update of rights is catered for, and both can be enforced through a forced requirement of online connectivity. Revocation can be resource based; and not only for a user or device.

We could not find any versioning support offered by ARM by default, and thus there is no way to distinguish different versions of a document without opening the document. This has potential for problems if a user is allowed to access one version but excluded from the other version (for example the user is allowed to access version 1 but not version 2). If the use license is not forced to be renewed, then the user should be able to continue accessing both versions.

ARM has a mechanism to define policies, and these policies are not embedded with the protected data. However, the format used for specifying these policies is not specified. The ratings for the core requirement analysis are summarised in table 3.16.

3.6.3 **Usability Requirement Analysis**

Time and space shifting are well supported by ARM, but because it has limited application support; format shifting and platform shifting is severely limited – although, ARM does support the conversion of documents into secure PDF. Likewise, ARM currently only supports later versions of Microsoft Office and there is no seamless integration with existing applications. The documentation provided by Authentica is unclear on the possibility of offline usage and whether rights can be delegated to other users.

Access control specifications are dictated by the application, and cannot be customised for different deployments. However, there are a wide range of existing controls already specified by ARM. There is also great support for tracking and monitoring, but we are unsure of how well offline usage is supported. The ratings for the usability requirement analysis are summarised in

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	2
3	Support Multiple User Authentication Protocols	2
4	Represent and Authenticate Resource Identity	n/a
5	Represent and Authenticate Device Identity	0
6	Represent and Authenticate User Groups	2
7	Represent and Authenticate User Roles	2
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	0
10	Represent the Authorisation (Use License)	2
11	Authenticate the Use License	n/a
12	Support User Duties	0
13	Revocation of Rights	2
14	Update of Rights	2

Table 3.16: Core Requirement Analysis for Authentica ARM

	Requirement	**Rating**
15	Time Shifting	2
16	Format Shifting	1
17	Space Shifting	2
18	Platform Shifting	1
19	Integration with Existing Applications	1
20	Delegation of Rights	n/a
21	Fine Grained and Flexible Access Control Specification	2
22	Tracking and Monitoring	2
23	Offline Usage	n/a

Table 3.17: Usability Requirement Analysis for Authentica ARM

table 3.17.

3.6.4 Legal and Social Requirements Analysis

Unlike media DRM systems, enterprise DRM systems are on a sounder legal footing, as they regulate the control of corporate data, often within a closed boundary. However, how sound their legal footing is, is unknown, especially when one considers different legislations with regards to employee privacy, monitoring of employees and employee access to information. In terms of transparency, there are no clear mechanisms on establishing the extent of the protections placed on the data without application support. ARM's performance cannot be rated because there is no performance data. The ratings for the legal and social requirement analysis are summarised in table 3.18.

	Requirement	**Rating**
24	A Legal Framework for DRM	1
25	Transparency	n/a
26	Privacy and Anonymity	n/a
27	Do Not Alter Platform Functionality and Performance	n/a

Table 3.18: Legal and Social Requirement Analysis for Authentica ARM

3.7 Microsoft Rights Management Services (RMS)

Microsoft's Rights Management Services (RMS) is a DRM system geared for the enterprise, and is the only DRM system we are aware of that has access control support in the operating system kernel. In this section, we examine the effectiveness of RMS as a general DRM system. RMS is also a proprietary solution, but like for Windows Media, there is a lot documentation released by Microsoft that can be used to examine the system [138, 137, 139]. We have also used Rosenblatt's comparative analysis of RMS and Authentica ARM [169] to confirm our own analysis. However, unlike WM and iTunes FairPlay, we do not know of any compromises that can reveal any additional information on RMS' operations.

3.7.1 Brief Overview

RMS can be used to protect any type of data, although its user authentication design has meant that it is used primarily for intra-enterprise use. Like Windows Media, RMS does not have any restrictions on distribution, but cannot really be super distributed. Although the use license is distributed separately, the restrictions on which user(s) (or user groups) can access the data is linked to the data itself. Thus, it is not possible for a user to acquire a license for the data, if they do not have preallocated access.

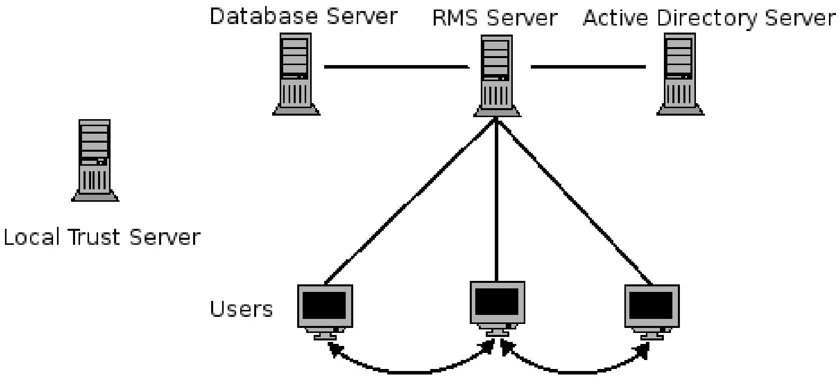


Figure 3.5: Deploying a RMS system

To use RMS in an enterprise, the enterprise must first set up an infrastructure of trusted entities. At the core of this set-up is a local certification authority that can create other trusted entities.

For this purpose, RMS requires a server (running Windows Server 2003) to be enrolled as a trusted server. Like with Windows Media DRM, this server requires its public/private key pair to be signed by Microsoft directly, through its RMS Server Enrolment Server. Once this is done, the server can act as a certificate authority for the enterprise, and enrol other servers (license servers, active directory for authentication and other certificate authorities), client machines and users as certified entities. The local trust server does not need to be connected to the machines once they have been enrolled as trusted entities.

Client machines require the installation of a RMS operating system component (which enforces some of the license conditions) and RMS-aware applications which can interact with the RMS operating system component. RMS-aware applications can assemble and use protected data. An overview of the enterprise roll-out is shown in figure 3.5.

3.7.2 Core Requirements Analysis

Unlike Windows Media, RMS does feature user identification through the use of Microsoft Active Directory or through Passport. Using Active Directory also allows the enterprise to regulate usage through the use of groups or roles. However, the use of Active Directory does mean that there is little scope for portability outside the enterprise, as any such measure would require the external systems to be within the trust realm and the user identities have to be integrated with the enterprise's Active Directory. Passport is not really a viable choice considering its poor security history [50]. More recent versions of RMS also allow for authentication through X.509 certificates, but the addition of other identification mechanisms during deployment is not possible. The trusted realms allows for device identification, but does not seem to cater for grouping devices under different security levels or categories.

RMS enabled data itself does not have versioning control by default, thus there is no way to distinguish between different versions of a document without opening the document. This has potential for problems if a user is allowed to access one version but excluded from the other version (for example the user is allowed to access version 1 but not version 2). If the use license is not forced to be renewed, then the user should be able to continue accessing both versions.

The use of dual licenses (an embedded license with the data, and an external use license), both of which are required, is puzzling; especially as it thwarts easier portability, rights revocation and renewal or extension of license terms. For example, update of rights is allowed, but requires the document to be repackaged with the new conditions and then redistributed to the recipient. Even with high speed networks, this can be impractical, especially considering the size of certain documents such as high resolution images or presentations. The ratings for the core requirement analysis are summarised in table 3.19

	Requirement	**Rating**
1	Provide Persistent Protection	2
2	Represent User Identity	2
3	Support Multiple User Authentication Protocols	1
4	Represent and Authenticate Resource Identity	0
5	Represent and Authenticate Device Identity	2
6	Represent and Authenticate User Groups	2
7	Represent and Authenticate User Roles	2
8	Represent and Authenticate Resource Groups	0
9	Represent and Authenticate Device Groups	0
10	Represent the Authorisation (Use License)	2
11	Authenticate the Use License	1
12	Support User Duties	0
13	Revocation of Rights	1
14	Update of Rights	1

Table 3.19: Core Requirement Analysis for RMS

3.7.3 Usability Requirement Analysis

Space shifting is easily possible, as the use licenses are not tied to specific devices, and because RMS does not specify formats, format shifting is only hindered by applications implementing RMS. Time shifting is also possible. However, platform shifting is not that easy, as it requires Microsoft to support RMS in the target platform, and requires applications in the target platform to be RMS enabled. Thus, we have to score RMS low on platform portability especially if one considers the fact that Microsoft does not support RMS on every version of Windows (even if legacy operating systems such as Windows 98 are excluded). RMS can be integrated with any application, but every application requires to be patched or upgraded to enable even the basic protections like read and write, which cannot be enforced by the operating system. However, the use of application support does allow for very fine grained and flexible access

	Requirement	**Rating**
15	Time Shifting	2
16	Format Shifting	2
17	Space Shifting	2
18	Platform Shifting	1
19	Integration with Existing Applications	0
20	Delegation of Rights	1
21	Fine Grained and Flexible Access Control Specification	2
22	Tracking and Monitoring	2
23	Offline Usage	1

Table 3.20: Usability Requirement Analysis for RMS

control specifications which can control any part of the application behaviour. Rights can be delegated but it requires the repackaging of the protected media, which could be expensive operation considering the size of certain documents such as presentations.

RMS has a comprehensive tracking and monitoring system, and it can be configured to handle various levels of data. Offline usage is possible, but depends on the flexibility of Active Directory or Passport to enable offline usage. The ratings for the usability requirement analysis is summarised in table 3.20.

3.7.4 **Legal and Social Requirements Analysis**

Unlike media DRM systems, enterprise DRM systems are on a sounder legal footing, as they regulate the control of corporate data, often within a closed boundary.However, it is unknown how sound their legal footing is, especially when one considers different legislations with regards to employee privacy, monitoring of employees and employee access to information. In terms of transparency, there are no clear mechanisms on establishing the extent of the protections placed on the data without application support. There is no performance data with RMS, thus it cannot be rated. The ratings for the legal and social requirement analysis are summarised in table 3.21.

	Requirement	**Rating**
24	A Legal Framework for DRM	1
25	Transparency	1
26	Privacy and Anonymity	n/a
27	Do Not Alter Platform Functionality and Performance	n/a

Table 3.21: Legal and Social Requirement Analysis for RMS

3.8 Standardisation Efforts

One of the main problems with current DRM systems is the lack of technical interoperability between the various implementations. Thus, there has been a drive from some participants in the vendor market to create standardised DRM frameworks, which could promote interoperability. At the current moment, there are three standardising efforts, but only one, OMA DRM (discussed in section 3.4), has been deployed commercially. In this section, we give a brief overview of two other standardisation efforts: MPEG-21 and Marlin. Both these systems are targeted primarily for the protection of media.

MPEG-21 is the oldest standardisation effort, and is made up of a number of parts, with each part focusing on a particular area of DRM [141]. For example, MPEG-21 Part 5 covers the Rights Expression Language (REL) specification. It is envisaged that these parts are implemented as Intellectual Property Management and Protection (IPMP) Tools, which can be then combined to form a complete DRM system. But, as noted by Sheppard in [179], MPEG-21 does not give many details on how to create these tools. One of the problems with this approach, as noted by Sheppard is that two implementations of the MPEG-21 standard could in fact not provide complete interoperability.

Marlin is the youngest standardisation effort, and has yet to release any approved output. Marlin aims to create a DRM system primarily for home networks for consumer electronics such as IPTV set-top boxes and portable media players, and the main players in the group are consumer electronics manufacturers [129]. Marlin is the only standardisation effort that has a bridge to another standardisation effort (OMA) allowing for the possibility for interoperability between the two standardising efforts. This is unsurprising as both standardising groups share common members. Marlin's architecture is similar to that of OMA DRM 2.0, although Marlin supports extra features such as user level authentication and user groups.

3.9 SUMMARY

In this chapter we conducted a requirement analysis of a number of different DRM systems, and our ratings are summarised in table 3.22. We focused our analysis on how easily the systems can be used as general DRM systems instead of focussing on the consumer evaluation of DRM systems, as Mulligan et al. did in [143].

Our analysis shows a few clear general issues:

1. DRM systems do not focus on users and thus have very poor user-management systems. Authentica ARM and Microsoft RMS are both designed for enterprises and are the notable exceptions. User management is a crucial component of access control and the lack of user management shows the immaturity of most of these solutions.

2. Using device identity as the base for identity management is clearly the popular approach, despite the fact that users own and operate a number of different devices; and in fact a single device can be owned and operated by multiple users. The use of device groups have largely been ignored except by OMA DRM v2 and Real Helix. Furthermore, electronic devices often have short lifespans (when compared to analogue media) but restrictions on space and platform portability severely hamper the user experience.

3. Revocation and change of access rights is also important in any access control schemes; and DRM systems have rather poor support for both of these. DRM systems currently revoke entire devices or users instead of focusing on individual resources; and in the long

run this is not a sustainable strategy.

4. Vendors of DRM systems do not advertise, and possibly do not understand, the legal and social requirements for their systems. Vendors like Microsoft and organisations like the OMA, who want to license their systems need to set up clear guidelines on what consumers expect from DRM protected media.

We believe that a successful DRM system needs to successfully address all the requirements we have outlined; and in the remainder of this dissertation, we outline a framework that addresses these requirements.

	Requirement	iTunes	WM	Helix	OMA v1	OMA v2	Adobe	ARM	RMS
1	Provide Persistent Protection	2	2	2	2	2	2	2	2
2	Represent User Identity	1	0	0	0	0	1	2	2
3	Support Multiple User Authentication Protocols	0	0	0	0	0	0	2	1
4	Represent and Authenticate Resource Identity	n/a	n/a	n/a	1	1	n/a	n/a	1
5	Represent and Authenticate Device Identity	2	0	2	2	2	0	0	2
6	Represent and Authenticate User Groups	0	0	0	0	0	0	2	2
7	Represent and Authenticate User Roles	0	0	0	0	0	0	2	2
8	Represent and Authenticate Resource Groups	0	0	0	0	0	0	0	0
9	Represent and Authenticate Device Groups	0	0	2	0	2	0	0	0
10	Represent the Authorisation (Use License)	0	2	2	2	2	2	2	2
11	Authenticate the Use License	0	1	n/a	2	2	2	n/a	1
12	Support User Duties	0	0	0	0	0	0	0	0
13	Revocation of Rights	1	1	0	1	1	0	2	1
14	Update of Rights	1	1	1	0	0	2	2	1
15	Time Shifting	0	2	2	0	2	2	2	2
16	Format Shifting	1	1	1	1	1	0	1	2
17	Space Shifting	2	0	2	0	1	2	2	2
18	Platform Shifting	1	1	2	0	1	2	2	1
19	Integration with Existing Applications	0	1	0	2	2	1	1	0
20	Delegation of Rights	0	0	0	0	0	0	n/a	1
21	Fine Grained and Flexible Access Control Specification	0	1	2	1	2	2	2	2
22	Tracking and Monitoring	0	n/a	0	0	1	2	2	2
23	Offline Usage	2	2	2	2	2	2	n/a	1
24	A Legal Framework for DRM	0	0	0	0	0	1	1	1
25	Transparency	2	n/a	n/a	n/a	n/a	2	n/a	1
26	Privacy and Anonymity	2	n/a	n/a	n/a	n/a	n/a	n/a	n/a
27	Do Not Alter Platform Functionality and Performance	2	1	n/a	n/a	n/a	n/a	n/a	n/a

Table 3.22: Summary of our requirement ratings for the various DRM systems which we discussed in this chapter.

CHAPTER 4

THEORETICAL FOUNDATIONS FOR DRM SYSTEMS

In this section, we discuss a number of key technical contributions that we believe can be considered as part of the foundations for DRM. These contributions include the components of a DRM system and structure of DRM systems. While we have mentioned some of these contributions previously, in this section we provide a more detailed discussion together with criticisms, if any. We also examine all the systems detailed in chapter 3 in terms of some of these contributions.

This chapter does not examine the related work of all aspects of DRM systems, but rather a selected number of related works that have a wide impact on DRM systems. Each component of our framework details further related work, which have particular significance in the development of that particular component.

4.1 ROLE PLAYERS IN A DRM SYSTEM

It is important to identify the role players involved in any system. In one of the earliest papers in the DRM field, Bartolini et al. mapped out a set of requirements for content management systems to a set of role players [46]. Some of these roles are also discussed by Koenen et al. in [118]. The roles described by Bartolini et al. are discussed next.

4.1.1 Discussion

1. **The Author** or the *creator* responsible for creating the work. The author of the work does not necessarily have to be human, and could be a machine (for example a computer program).

2. **The Right Holder** is either the copyright holder or an assigned representative of the copyright holder of the work. It must be noted that the author is not necessarily

the copyright holder and usually the right holder is also referred to as the *owner* of the work.

3. **The Service Producer** is the entity that is responsible for creating the protected DRM package from an unprotected work. In our discussion we refer to this service as the *packaging service*.

4. **The Media Distributor** is the entity that is responsible for distributing the protected work, and could also be responsible for collecting revenue from the consumers. In many current systems, the media distributor is also the service producer (e.g. Apple iTunes music store).

5. **IPR Register** or database is a server that maps the rights associated with the work to the consumer. In subsequent literature, this role player is referred to as the *license server*, and we also make use of this term.

6. **Unique Number Issuer (UNI)** is responsible for issuing a unique identifier to each DRM package.

7. **The Controller** is a Trusted Third Party (TTP) that is responsible for ensuring that all the transactions have been carried out legally. It can be seen as both a monitoring service and a service to address disputes between rights holders and consumers.

8. **The Certification Authority** is another TTP that is responsible for issuing digital certificates to the other role players, which can be used as the means for authentication.

However Bartolini et. al. does not take into consideration the consumer of the work as a role player in the DRM system. Consequently, the chain of users that can occur in the lifecycle of a protected work (as discussed by the DMP in [63]) is missing.

4.1.2 Analysis of Role Players in Existing DRM Systems

Even though all the DRM systems we discuss below were developed and released after the discussion by Bartolini et al., most current systems do not feature most of these role players as part of their system.

In table 4.1, we summarise the identifiable role players in current systems. The absence of the distributor role player is not problematic; both Adobe and RMS are both distribution agnostic and thus do not need this role player. With regards to the UNI role, most DRM systems do not publish details regarding how protected data is identified.

Role Players	iTunes	WM	Helix	OMA v1	OMA v2	Adobe	ARM	RMS
Author	X	X	X	X	X	√	√	√
Rights Holder	X	√	√	X	√	√	√	√
Service Producer	X	√	√	X	√	√	√	√
Distributor	√	√	√	√	√	X	√	X
License Server	X	√	√	√	√	√	√	√
UNI	n/a	n/a	n/a	√	√	n/a	n/a	√
Controller	X	X	X	X	X	X	X	X
Certificate Authority	n/a	√	√	√	√	√	√	√

Table 4.1: Identifiable Role Players in Current Systems

4.2 Security Taxonomy

In 2000, Park et al. [149] looked at all the different possible security architectures that could be implemented for securing content distribution, and categorised DRM systems using three factors: the presence of a virtual machine, the type control sets and the distribution style.

4.2.1 Discussion

Figure 4.1 shows the different distribution architectures with the notation used by Park et al. to distinguish between them.

The Virtual Machine

The first level of distinction is the presence of a virtual machine, or a DRM controller as defined by Rosenblatt and others [81, 168]. The virtual machine is described by Park et al. as "software that runs on top of vulnerable computing environment and employs control functions to provide the means to protect and manage access and usage of digital information" [149]. A virtual machine can be in the form of a plugin that controls access to DRM enabled data, and currently most DRM products make use of a DRM controller [81, 168]. Systems that do not have a virtual machine cannot manage and control access and usage of a secure data that the recipient receives. For this reason, systems without virtual machines to enforce usage policies, are unsuitable as part of a DRM solution.

Control Set

The second level of distinction is in the type of control set used. Control sets are the rules governing the use of a DRM enabled work, and this has given rise to Right Expression Languages (REL) that allow for the description and specification of the control sets. Park et al.

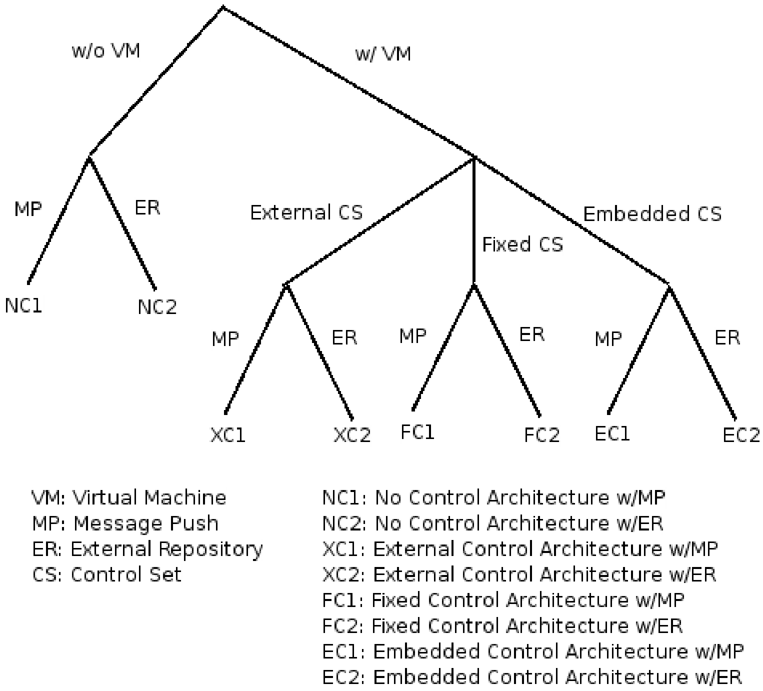

w/o VM
w/ VM
MP
ER
NC1
NC2
External CS
Fixed CS
Embedded CS
MP
ER
MP
ER
MP
ER
XC1
XC2
FC1
FC2
EC1
EC2
VM: Virtual Machine
MP: Message Push
ER: External Repository
CS: Control Set
NC1: No Control Architecture w/MP
NC2: No Control Architecture w/ER
XC1: External Control Architecture w/MP
XC2: External Control Architecture w/ER
FC1: Fixed Control Architecture w/MP
FC2: Fixed Control Architecture w/ER
EC1: Embedded Control Architecture w/MP
EC2: Embedded Control Architecture w/ER

Figure 4.1: Categorisation of DRM Systems [149]

categorised control sets into three types: fixed control sets, embedded control sets and external control sets [149].

Embedded and external control sets are more adaptive to the needs of the user. In an embedded control set, the DRM enabled work comes with the control set embedded into the work. This can be done by encapsulating the control set and the work in a security envelope [149]. In an external control set, the DRM enabled work and the control sets arrive separately. The obvious advantage of a system of this nature is that a single control set can be used to define rights for multiple works of the same type. On the other hand, external control sets are usually held on a network server and are required to be accessed each time a DRM work is accessed [168]. This creates a network overhead as well as leading to questions of user privacy and right holder control. Both of these systems can be further combined with a fixed control set.

Distribution Taxonomy

The third and final level of distinction is in the distribution process. Park et al. differentiated between *message push* and *external repository* [149]. In a message push system, the data is transferred between the sender and recipient by a direct communication channel such as e-mail. In an external repository, the recipient fetches the data from a central repository and there is no need for the recipient to store the data locally. Thus, in external repository based systems, distribution could be considered part of the control infrastructure. These systems are frequently associated with subscription and streaming media services.

4.2.2 Characteristics of the Security Architectures

Park et al. also discussed some of the characteristics of the security architectures described above. In these characteristics, they did not take into account any restrictions imposed by other elements in a DRM system such as the right expression languages. Table 4.2 summarises some of the security and functional characteristics of the distribution systems, as discussed by Park et al.

As we have already stated, we are focusing on distribution agnostic systems, and thus are mainly interested in architectures FC1, NC1 and XC1. As expected, fixed and embedded control sets offer lower flexibility, with regards to update of access control permissions, but largely offer the same range of functionality.

The majority of our requirements, as discussed in chapter 2 are not reflected in the characteristics discussed by Park et al. A large number of the requirements discussed by Park et al. are focused on the implementation of the access control policies (such as authentication), which are enabled in the same manner, regardless of the architecture. The majority of the usability requirements are also dependent on the implementation of the virtual machine, thus not covered by the taxonomy.

4.2.3 Categorisation of Existing Systems

We have categorised the systems we discussed in chapter 3 in table 4.3. RMS and OMA DRM v1 offer more than one form of control set. Windows Media and Helix both offer message push services as part of the subscription model. iTunes uses a message push architecture to disseminate data, but also uses local storage. Helix DRM and Windows Media DRM both have support for subscription music stores and streaming media, which are largely based on message push distribution.

4.3 Rights Expression Languages (RELs)

In DRM systems, access control rules for an object are expressed in a *use license*. Use licenses are expressed in *rights expression languages (RELs)*, of which there are two major, general purpose RELs – MPEG REL based on XrML and ODRL [70].

In the XrML 2.0 specifications [6] the requirements for a REL are given as:

- *Comprehensive*: A language that shall be capable of expressing simple and complex rights

Characteristics	**FC1**	**FC2**	**EC1**	**EC2**	**XC1**	**XC2**
Disseminator can control access and usage of disseminated digital information	Y	Y	Y	Y	Y	Y
Disseminator can change recipient's access rights after dissemination				Y	Y	Y
Re-disseminated digital works can be protected	Y	Y	Y	Y	Y	Y
Special Client Software (Virtual Machine) is vulnerable to attacks	Y	Y	Y	Y	Y	Y
Tracking re-disseminated digital works is possible	Y	Y	Y	Y	Y	Y
Disseminated digital container is reusable for other recipients by re-dissemination					Y	Y
Digital information does not have to be on recipient's storage		Y		Y		Y
Digital information can be accessible from any machine if it is connected to network		Y		Y		Y
Recipient should carry digital information to access it from multiple machines	Y		Y		Y	
Control center trusted by both distributors and recipients is mandatory					Y	Y

Table 4.2: Characteristics of DRM Architectures [149]

System	**FC1**	**FC2**	**EC1**	**EC2**	**XC1**	**XC2**
Apple iTunes Fairplay	Y	Y				
Microsoft Windows Media DRM					Y	Y
Real Helix DRM					Y	Y
OMA DRM v1	Y		Y		Y	
OMA DRM v2					Y	
Adobe Document Security					Y	
Authentica Active Rights Management					Y	
Microsoft Rights Management Services			Y		Y	

Table 4.3: Categorisation of Existing Systems

in any stage in a workflow, lifecycle or business model.

- *Generic*: A language shall be capable of describing rights for any type of digital content or service (an ebook, a file system, a video or a piece of software)

- *Precise*: a language shall communicate precise meaning to all players in the system.

There are a number of criticisms of current REL implementations; particularly with regards to the expression of legal requirements when enforcing copyright [142, 82]. Mulligan et al. argues that RELs like XrML cannot be considered comprehensive until users are able to request additional rights [142]. They argue that this ability is crucial for the enabling of fair use. Felten on the other hand argues that DRM systems will never allow fair use since the languages cannot handle the expressions and the AI complexities in fair use [82]. In these respects, they argue that current RELs are not comprehensive.

Bechtold however argues that many of the XrML rules and definitions like rights transfers are not implemented in current DRM systems [48] and thus the failure of DRM systems to have fair use is not hampered by the language. Bechtold maintains that a suite of programs that can implement all the rules and definitions available in XrML will be able to achieve most of the requirements of DRM systems with less compromise from right holders [48]. This would require users to communicate with the right holder to request additional rights or changes in rights, as argued by Mulligan et al. In [29], we detailed extensions to XrML and ODRL that would allow for bi-directional communications. We further elaborated our bi-directional extensions for ODRL in [30]. The latest models for ODRL v2.0 has some of these features incorporated [105].

While the above arguments have been in favour of extending the capabilities of RELs, Jamkhedkar et al. argued in [108], that there are a number of problems with current approaches to RELs. First, general purpose RELs have become too complicated, and by trying to address all the parts of DRM, they do not address any of the part completely. Although RELs are already modelled on access control models [142], Jamkhedkar et al. argued that there is no real definition of an access control model for DRM, and thus there is no mechanism to evaluate and inter-operate between different RELs. The authors promoted the need for a simpler model, encompassing a stateless, language-neutral, rights model; but did not present any rights model for DRM.

There have been no formal investigations into RELs from the vendors themselves. Halpern and Weissman detailed formalised semantics to XrML in [95], while Pucella and Weissman detailed a similar investigation into ODRL in [158]. Both papers investigated the current problems with interpreting use licenses, and the sources of ambiguities. Pucella and Weissmann also discussed a logic for reasoning about and interpretation of license agreements in [157]. In some respects, these investigations found that both XrML and ODRL were not necessarily precise in conveying their intended meaning.

While these investigations examined the interpretation of existing languages, they did not seek to put the languages on a completely formal base. In [94], Guth et al. did investigate the requirements for a contract language, and developed a contract schema (CoSa). In [93], Guth had a more comprehensive discussion on CoSa, the relationship between contracts and use licenses and the requirements for rights expression languages. While CoSa provided a formal structure, this approach does not extend to RELs and there is currently no formal description for any REL.

Thus, while both ODRL and XrML are generic in nature, and can cater for any type of digital resource [70], both can be considered to be non comprehensive and imprecise.

4.4 THE DRM REFERENCE MODEL

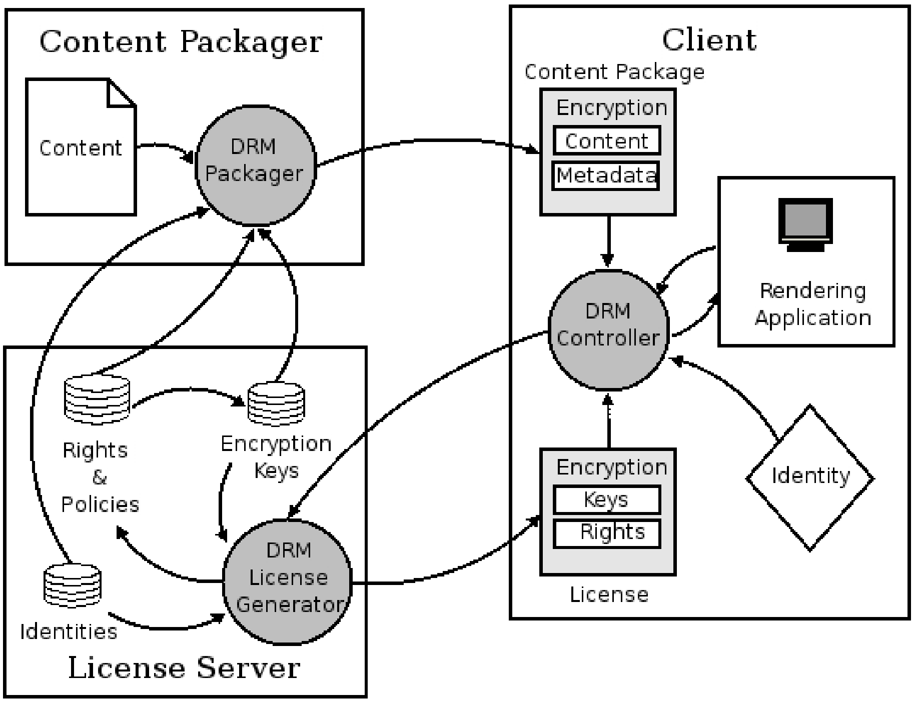


Figure 4.2: The DRM Reference Model [168]

The DRM reference model examines the components of a DRM system, and is similar to the role players discussed in section 4.1, except that a number of prominent role players such as the author and rights holder seem to be missing. This approach primarily maps the interaction between the core components of a DRM system, another feature missing from the discussion of Bartolini et al. Figure 4.2 gives an overview of a DRM system as described by Rosenblatt in [168], and subsequently in [169]. Similar reference models have also been discussed by Erickson in [81] and Schmidt et al. in [175].

One of the main differences between the two approaches is the inclusion of the consumer, and how the consumer relates to both the packaged work and the license server. The architecture also introduces the concept of the *DRM controller*, the entity responsible for interpreting and enforcing the access control rules and policies. There are also two identity interfaces for this architecture – one on the producer side and one on the consumer side.

All current DRM systems, except iTunes, make use of a modified architecture resembling the architecture above. The most notable change to the above architecture is the centralisation of identity management (as a separate component) in DRM systems like RMS and ARM. In iTunes, there is no license server, with the rendering application implementing a fixed set of access controls.

4.5 A Layered Approach to DRM

In the OSI 7-layer network model, communication between two applications over a network was abstracted to a set of independent layers, each performing a specific function in the process. With the independence of the layers, each layer could have a multiple number of protocols, but each of these protocols could utilise the layers below and above them. While the OSI 7-layer protocol has not been fully implemented, the success of modern computer communication networks can be attributed to the model. Communications using different application protocols (like HTTP and SMTP) can communicate over the same network. This model has also allowed for the possibility of using the same application protocols over the different physical networks; for example the use of HTTP over both wired and wireless networks.

In [107], Jamkhedkar and Heileman describe a layered architecture for DRM systems. They argue that the layered architecture promoted the development of open standards and promoted interoperability between systems; both of which are lacking in the DRM space. The authors also argue that one of the key reasons for the success of the Internet is the standardisation and usage of only one protocol in the "middle layer" – the use of IP in the network layer. IP is also a very simple protocol, and its stability has allowed for the growth of the Internet. Similarly, the authors argue that the DRM layered model requires a simple, standardised middle layer that can remain stable and usable even with rapid changes in other layers.

4.5.1 Discussion

Figure 4.3 shows the layered solution to DRM proposed by Jamkhedkar and Heileman in [107]. The Application and Negotiation layers form the upper layers, the Rights Expression and Rights Interpretation layers form the middle layers while the Rights Enforcement layers form the lower layers. Below we summarise the roles played by each layer.

Application Layer: Based on the application of DRM (envisaged for different content types), different application protocols can service the specific needs of the application. For example the rendering requirements for a book and a music file are different, and the appli-

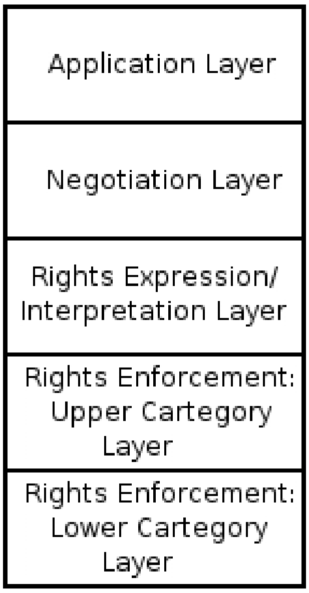

Application Layer
Negotiation Layer
Rights Expression/ Interpretation Layer
Rights Enforcement: Upper Cartegory Layer
Rights Enforcement: Lower Cartegory Layer

Figure 4.3: Proposed DRM Layers

cation layer caters for these differences.

Negotiation Layer: The negotiations layer would allow the vendor and the user to negotiate the terms and conditions for accessing content. For example, a vendor may wish to only allow devices running a specific operating system to access the content. The protocols in this layer would allow the user to prove this condition to the vendor in some manner (for example through the use of a digital certificate from the OS publisher).

Rights Expression / Interpretation Layer: In [107], the authors propose this layer to be the "notch of the hour glass" – a layer that is the least complex and most standardised (similar to the IP layer in networking). This would require the standardisation on one REL (or developing a new one) and should offer completeness and consistency to remain in use for a substantial period of time.

The authors argue that not only do RELs need to be standardised, but also how they are interpreted, since RELs only offer mechanisms to express rights. The interpretation could be tied to the application of the REL. In a subsequent paper, [98], the authors used the abbreviation REI to refer to this layer.

Rights Enforcement – Upper Category Layer: Both rights enforcement layers can be subdivided to further sublayers. The upper layer is responsible for handling content according to its type – for example granting access to editors and rendering content to the user.

Rights Enforcement – Lower Category Layer: The lower layer is responsible for handling enforcement regardless of content type. These include preventing illegal access to data (for example if the REL specifies that the data should not be accessible only through a specific application), authenticating device drivers etc.

4.5.2 Analysis of Existing Systems

In [132], Michiels et al. examined a few DRM systems against the layered model, as part of an effort to extract high level use cases for DRM systems. They also categorised some of the components identified in section 4.4, into the various layers presented by Jamkhedkar and Heileman.

Our analysis is slightly different to Michiels et al. and we focus in identifying how the components and processes of existing systems fit into the layered architecture. We have also condensed our discussion on enforcement as a single layer. Our analysis is split into two parts; we discuss DRM systems focused on media in table 4.4, while the enterprise DRM systems are discussed in table 4.5.

4.6 Interoperability in DRM

One of the main motivations for the layered architecture discussed in section 4.5, is to promote interoperability between DRM systems. In a later paper, [98], Heileman and Jamkhedkar discussed how the layered approach helps with DRM interoperability, and the effect of standardising a complete layer (as opposed to just the communication between layers) on the surrounding layers.

Heileman and Jamkhedkar defines interoperability as

> The ability of one technology to interact with another technology in order to implement some useful functionality [98]

The role of standards as a means to enable interoperability is well understood, but while there are market forces that push for interoperability, Alvestrand points out that there are also business models that rely on non-interoperability [23]. Furthermore, if interoperable technologies reduce the functionality offered by the product, there is a chance that these technologies will not succeed in the marketplace, as product could be percieved as inferior to competing non-interoperable products.

DRM System	Application Layer	Negotiation Layer	REI Layer	Enforcement Layer
iTunes	The iTunes Software package and iPod music players are the only applications capable of rendering protected files.	The music store's interface is similar to a traditional e-commerce website. There is no negotiation beyond the point of establishing whether the consumer can pay, and is allowed to purchase the media.	The exact details of Fair Play remain unpublished.	Enforcement is at the application layer (upper layer) through the iTunes Software and could be considered to be at the lower layer for the iPod.
Windows Media DRM	Applications make use of the Windows Media DRM libraries to render content. These libraries are available for embedded platforms.	There are no negotiation capabilities in the DRM system itself.	The REI details are unpublished.	Enforcement is at the application layer (upper layer) through the libraries and could be considered to be at the lower layer for the embedded devices.
Helix DRM	Applications make use of the relevant libraries to render content	There are no negotiation capabilities in the DRM system itself.	The REI details are unpublished.	Enforcement is at the application layer (upper layer) through the libraries and could be considered to be at the lower layer for the embedded devices.
OMA DRM 1 & 2	DRM controller is partly part of the mobile device, partly at the software level.	There are no negotiation capabilities in the DRM system itself.	OMA makes use of a subset of the ODRL 1 REL. The small subset removes the need for an interpretation layer	Enforcement is largely at the lower layer through hardware, but upper layer enforcement through software is possible.

Table 4.4: Analysis of Media DRM Systems against the Layered Architecture

DRM System	Application Layer	Negotiation Layer	REI Layer	Enforcement Layer
Adobe Document Security	For the consumer, only the Adobe PDF Reader can interface with the DRM system.	There are no negotiation capabilities in the DRM system itself.	The REI details are unpublished.	Enforcement is purely at the upper layer through the Adobe PDF Reader.
Authentica Active Rights Management Platform	Authentica creates patches for existing applications to interface with the DRM system	There are no negotiation capabilities in the DRM system itself.	The REI details are unpublished.	Enforcement is purely at the upper layer through the applications
Microsoft Rights Management Services	Applications have to be RMS enabled to interface with the DRM system. Any RMS enabled application can interface with the system.	There are no negotiation capabilities in the DRM system itself.	RMS makes use of XrML REL. The interpretation is application dependent.	The RMS enabled applications provide upper layer enforcement, while lower level enforcement is provided through the operating system module. Both enforcement mechanisms have to be present for access to be granted though.

Table 4.5: Analysis of Enterprise DRM Systems against the Layered Architecture

Interoperability is seen as a major issue for DRM systems, and a number of different authors have discussed the issue. In [113], Kalker discussed how a lot of opposition to DRM, such as [53], stems from the lack of interoperability rather than the technology itself. In this section, we look at some of these discussions.

4.6.1 Types of DRM Interoperability

In [118], Koenen et al. defined three types of DRM interoperability: full format interoperability, connected interoperability and configuration driven interoperability. The authors also discussed each approach, together with their advantages and disadvantages.

Full Format Interoperability

In full format interoperability, the representation and processing of the digital content is the same. This creates a need for standardised data representation, encoding, key management etc. There are a number of advantages to this approach, and it shares all the advantages expected from a standardised format. This includes a wide number of participants in both the production of DRM content and rendering applications.

The main disadvantage of this approach is the long time required to develop standards, and adoption often requires critical mass. Furthermore, there can be multiple competing standards. There is also the potential for a wider effect of security vulnerabilities – a vulnerable format standard exposes every compliant system.

Despite these drawbacks, Koenen et al. feel that full format interoperability is the most desired state, and they urge the adoption of full format interoperability where possible.

Connected Interoperability

Connected interoperability depends on online connectivity. Users connect to an appropriate web service that can be used to translate required data between the various DRM systems. The main advantage of this system is that different systems can co-exist and it removes the burden of interoperability from the vendors. We discuss this in more detail in section 4.6.3.

Configuration-driven Interoperability

In this approach, the consumer can convert protected data between various DRM systems through tools. Thus, a consumer can use a tool to convert all protected works for a device that implements only DRM System A. The main difference between this approach and connected interoperability is that this is driven by the consumer and not a web service. The main disadvantage of this approach is that there will be a need a high degree of trust in the tools from the involved vendors.

4.6.2 REL Interoperability

Since RELs form an important component of DRM systems, there have been approaches to promote interoperability between various RELs. However, there are a number of challenges in providing such a service.

As discussed by Schmidt et al. in [175] and Coyle in [70], there are scenarios when, expressions in one language cannot be translated to another language. This is partly due to a limitation in the vocabulary (although RELs such as XrML and ODRL can extend the vocabularies) but it can also be due to the syntax of the language itself.

In [172], Safavi-Naini et al. discuss their experiences in converting between XrML based MPEG-21 and ODRL based OMA-REL. Since OMA-REL has a significantly smaller vocabulary, translation from MPEG-21 to OMA-REL is not possible in the generic case. In [175], Schmidt et al. discuss how some expressions possible in XrML are simply not possible in ODRL 1, because such semantics do not exist in the language.

4.6.3 Interoperability through Intermediaries

In [175], Schmidt et al. also proposed connected interoperability as a solution to interoperability. They presented a scheme where a trusted intermediary converted a protected work from one system to be used in another.

In a later development, the Coral Consortium[1] has created an interoperability framework that is based on this approach. They have documented their approach in [67, 68, 69]. As suggested by both [118] and [175], the Coral Consortium provides a trusted intermediary that can convert both protected works and their associated licenses between DRM systems. They also detail various “ecosystems” in which detailed use cases have different requirements for the intermediary services.

However, we think that there are two major disadvantages to this approach. Firstly, vendors must still provide the necessary interfaces to allow online services to provide connected interoperability. However, as Alvestrand discusses in [23], one of the business models behind non-interoperability is to lock consumers into a particular device and operating system, and thus they may be averse to releasing the necessary details required for connected interoperability.

[1] http://www.coral-interop.org

Secondly, there is a duplication in data and licenses, and even with cheap storage, the storage requirements for the consumer increases in direct proportion to the number of different DRM systems the consumer wishes to use. Thus, if this approach is used for all the media DRM systems discussed in chapter 3, the consumer will need a copy of every data and license 5 times (for each of the systems). In the long term, and with large media libraries, this approach is impractical.

4.7 PROTOCOL AND SYSTEM MODELLING

An important criterion for any system or protocol design, is to ensure the integrity and correctness of the protocol. However, similar to formal models for RELs, there are currently no published models of DRM systems or the protocols used in DRM systems. We make use of Petri net modelling in two instances: in our discussion of negotiation protocols for DRM systems in chapter 7, and the authorisation process discussed in chapter 9.

Petri nets have been widely used to model and formally represent discrete distributed systems [47], and we have chosen to use coloured Petri nets (which are a subset of Petri nets) to model our protocols. Furthermore, as discussed by Guth in [93], Petri net modelling has been widely used in modelling business processes, and is thus appropriate for modelling negotiation protocols. Petri nets are place-transition nets comprising a non-empty set of places, transitions, arcs connecting places to transitions and tokens to define the value of a given place. Coloured Petri nets provide for systems with more than one type for a given place, and any coloured Petri net can be described as a traditional Petri net [47].

In [47], the authors discuss the following properties of Petri nets, and for each property, we discuss its implications in modelling protocols or system workflow.

1. **Reachability:** The reachability set of a Petri net is the set of all possible states achievable for a given system. Proving that a Petri net is reachable implies that every state in the associated system is achievable. Furthermore, reachable Petri nets are required before a system can have steady state distribution [47] (where the performance of the system is not affected regardless of the number of iterations of the protocol run, e.g. lack of buffer overflows). For these reasons, it is highly desirable to have reachability in communication protocols.

2. **Liveness:** A Petri net is live if there is at least one possible transition between two different states of the net. A protocol whose Petri net is not live, has a deadlock and the protocol cannot continue to execute. Thus, it is necessary for a protocol to have a live Petri net.

3. **Boundedness:** A Petri net is bounded if there is a maximum number of tokens of each type, regardless of the number of iterations. Thus, an unbounded net often indicates a flaw in protocol, like the possibility of buffer overflows. Thus, it is necessary for a protocol to have a bound Petri net.

4. **Safety:** A Petri net is safe if the maximum number of tokens of a given type, at any given place is 1.

Circular deadlock detection is not a direct property of Petri nets. However, if a Petri net representation of a protocol is *bound* and *reachable*; then it follows that the protocol does not have a circular deadlock. This follows from the fact that, if there is a circular deadlock, certain states of the Petri net will no longer be reachable.

Petri net modelling for our protocols cannot prove that they are correct in their intended functions, but can prove that they have no obvious flaws. It is for this purpose, that we have used Petri net modelling. We have used a well known Petri net tool, CPNTools[2], version 2.0.0 for GNU-Linux, to create and analyse our Petri nets.

4.8 Summary

In this chapter, we outlined a number of key contributions to the field of DRM in general, and we have also discussed how some of the existing systems relate to some of the architectures discussed. In [46], Bartolini et al. were one of the first to identify roleplayers and requirements for DRM systems. Despite this, many of the roleplayers are missing from current systems. Later, Rosenblatt and others, created a reference model for DRM system, featuring most of the roleplayers identified by Bartolini et al. Most modern DRM systems can be said to be based on this reference model.

In [149] Park et al. discussed various distribution taxonomies for DRM systems. They also discussed the relative advantages and disadvantages of the various strategies. Most modern DRM systems use external control sets, in a distribution agnostic environment (also known as super distribution).

RELs have been the focus of most interoperability efforts. However, as discussed by Koenen et al. in [118], there are also other aspects to interoperability. Other forms of interoperability such as connected interoperability and configuration driven interoperability are not as powerful as full format interoperability, but could be easier to implement. Despite the focus on RELs, there are still no formal definition for REL, and the interpretation of RELs. In [107], Jamkhedkar and

[2] http://wiki.daimi.au.dk/cpntools/cpntools.wiki

Heileman discussed a layered approach to DRM systems, motivated by the famous OSI 7-layer for computer networks. They argue that this approach would be easier to standardise, and thus make it easier to achieve interoperability. They proposed to make standardised RELs the core of such a model.

We also discussed the usage of Petri nets for modelling systems, and the use of modelling to prove the integrity and robustness of the system design.

Part II

Framework

CHAPTER 5

AN OVERVIEW OF OUR GENERAL FRAMEWORK FOR DRM

The layered model for DRM that we detailed in chapter 4, differentiated the various parts of DRM systems, but it did not give any detail on the contents of the layers themselves. In the rest of this dissertation, we present a general framework for DRM which is complementary to the layered model; and presents a general approach for any DRM system. An overview of our framework is shown in figure 5.1 and detailed below.

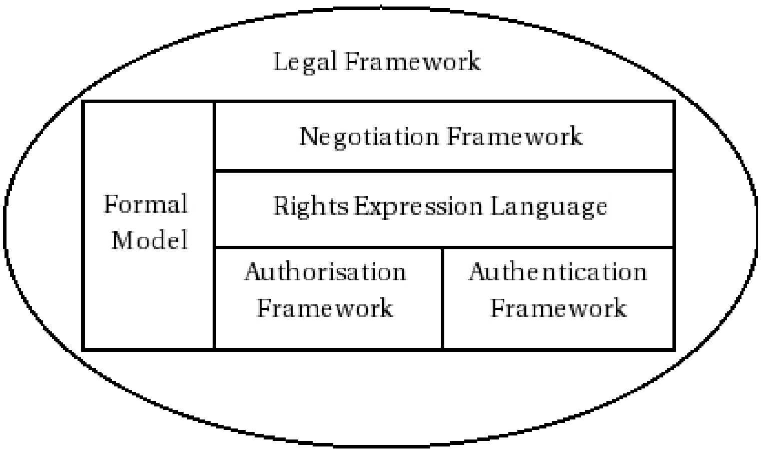


Figure 5.1: A General Framework for DRM

In **chapter 6**, we present a legal framework for DRM that forms the outer layer. It is advantageous for both consumers and producers of any computer system to have a clear legal position. Since many DRM systems have been used primarily as a mechanism to thwart piracy, and promoted as a means to enforce copyright, a legal framework for DRM is a necessity.

Our approach to DRM is to look for the common ground between media and enterprise DRM systems; and is thus largely application agnostic. For this reason, we do not investigate the application layer of the layered model for DRM.

Negotiations can be defined as the means to concluding a contract, and is thus influenced by our legal framework. While Jamkhedkar and Heileman motivated the need for negotiations in DRM [107], we detailed real advantages offered by such a facility in DRM systems in [31]. In **chapter 7**, we discuss negotiations in more detail, and present a negotiation framework for DRM, complete with two formal negotiation protocols, and petri-net modelling to verify the correctness of the protocols.

While there have been attempts at formalising DRM systems, there is still no formal description of access control as defined by DRM. In **chapter 8**, we present a formal model for DRM, which governs all the layers of the DRM model. Furthermore, we use the formal model to present a formal base for RELs, another aspect missing from current DRM systems.

Access control is a two step process: authentication of the parties involved and the authorisation of the parties to perform a certain task. For this reason, our enforcement (and lower layers of the framework) are split into two parallel discussions. In **chapter 9**, we present an authorisation framework for DRM, and also discuss upper and lower enforcement categories. In **chapter 10**, we present an authentication framework, and discuss different aspects of authentication in DRM systems.

Lastly, while not part of the framework itself, we feel that there is a need to discuss how DRM packages are assembled and a discussion on file formats to enable full format interoperability. We do this in **chapter 11**. There is also a need to define the trust relationships between the different layers of our framework, and the distribution of keys within the DRM system. We do this in **chapter 12**.

This framework is envisaged as a complete, single framework for DRM. However, the frameworks are modular in nature, and some frameworks could be implemented independently. For example, the legal framework is applicable to any DRM framework, while the authentication and authorisation frameworks are symbiotic and thus would be difficult to implement independently of each other. The framework is also expandable: additional legal context can be added to the legal framework, additional negotiation protocols can be added to the negotiation framework; while the Rights Expression Language can be extended to cater for different scenarios. The formal model can also be seen as independent of the different frameworks, and is a formal description of how all these frameworks work together. Thus, while the framework is envisaged as a complete framework for DRM, it can be implemented in parts.

Chapter 6

Legal Framework For DRM

Since the very beginning of civilisation, society has tried to define fair conduct between its members. As discussed by Maine in [128], these laws first defined conduct within a family, then a tribe (or group of families, usually with the same lineage), then a group of tribes within a certain area (i.e. a society), before expanding to cover nations. Over time, laws have evolved to cover conduct between two persons, a person and the state and between two states. As further discussed by Maine, every law can be broken into three parts: a command by the law giver, the obligations imposed on the parties and the threat of a sanction should there be disobedience. Laws governing economic transactions follow a similar pattern, and strive to create a fair balance for both parties (usually a buyer and a seller) in the transaction.

Copyright law was initially developed to strike a fair balance between authors and users of books, but has since expanded to cover a large proportion of intellectual property. Since DRM aims to control access to digital data, it is governed by copyright law. It is therefore important that DRM systems operate in a well defined legal framework, free from ambiguities. This is especially required in the case of media DRM systems, which has experienced a number of well documented criticisms, because consumers argue that these systems often have terms and conditions that infringe their legal rights.

As discussed in chapter 3, current DRM systems have chosen largely to ignore legal requirements, or to discuss how DRM systems fit into current legal systems. Furthermore, as discussed in chapter 2, vendors often ignore the legal position of DRM systems in the requirements and features of the systems. In this chapter, we review some problems highlighted by current DRM systems, the accommodation of DRM systems in existing legal frameworks and the issues that need to be addressed by DRM systems to comply with the existing legal frameworks – such as enabling fair use through negotiations.

6.1 Copyright & DRM: A Background

Digital data, regardless of its content, form and means of creation, is a storage medium for information. The primary legal model governing the protection of information in most countries and territories has been copyright law. While copyright law has different forms in different legal systems, most countries and territories are signatories of the Berne Convention [195], and various World Intellectual Property Organisation (WIPO) treaties [196]; and have thus defined the core protections embodied in these treaties. Since DRM is a set of technologies that aim to provide protection to information, it is vital that DRM is aligned to the legal protection provided by copyright law.

DRM, however, does not always confine itself to implementing copyright law. There have been a number of criticisms in both public forums [8, 10] and academia [49, 143] regarding the amount of control exercised by DRM systems over consumers, and sometimes even asserting control that a copyright holder does not necessarily have [173, 65]. Another criticism of DRM is that it often does not allow for *fair use* (USA) or *copyright exceptions* (EU) or *fair dealing* (UK, South Africa) – a set of circumstances where some parameters of copyright law can be broken by the user.

However, as has been widely discussed in both academia [82, 81, 173] and in the press and public forums [8, 10, 97], fair use as defined in the US copyright law is almost impossible to implement on a computer. Fair use is necessarily vague, and Felten described fair use as a "*feature for lawyers*" – applications should be argued in court on an individual basis [82]. Felten further argued that evaluating fair use would require sophisticated AI, and the factors involved are "*AI-hard problems*" [82]. This scenario is not necessarily the same in many European countries, the United Kingdom and South Africa, with their copyright laws providing a few detailed circumstances for copyright exceptions and fair dealing respectively [5, 184, 4]. However, it is still required to be argued in court and thus can still be regarded as a "feature for lawyers". Referring to US copyright law, Mulligan et al. had previously argued that Rights Expression Languages (RELs) could not express or even approximate most of the limitations posed in copyright law, and thus DRM systems in fact "*distort copyright law*" [142]. Thus, in terms of fair use[1] alone, DRM systems are not aligned to the requirements of copyright law.

In England, before the first copyright law (The Statute of Anne, 1709), reproduction and distribution of information through the printworks were largely regulated by the Stationer's Com-

[1] Although the terms fair use, fair dealing and copyright exceptions promote a similar idea, they are not the same. In fact, a particular usage can be considered as fair use in one country (or territory) and not be considered as a fair use in another. However, it can be argued that fair use is a more well known term and we have thus used this term to refer to the overall set of terms and conditions detailing copyright exceptions.

pany, established in 1556. Various acts, including the Licensing Act of 1637, gave the Stationer's Company a monopoly, for two main reasons [56, 72, 96]:

1. Allowed booksellers (effectively the printers and publishers) to recover their sizable investments in printing works.
2. Provided a mechanism to enforce and regulate what information was reproduced and distributed; i.e. censorship

The licensing act expired in 1694, and for a while there was no regulation or restrictions on reproductions and distribution of printed works. Prominent authors, like Daniel Defoe, pushed for a new law that would recognise and protect the author's contribution, the argument was that such protection would encourage learning and more authors would emerge, thus being good for society at large [56, 72]. This argument was also adopted by the publishers, and the Statute of Anne in 1709, provided authors with a limited monopoly of 7 or 14 years, to exploit their works for economic benefit. As discussed by Leafer in [122], a similar argument is present in Article 1, Section 8, Clause 8 of the Constitution of the United States of America; which forms the basis of US copyright law:

> To promote the progress of science and the useful arts, by securing for limited times, to authors and inventors, the exclusive right to their respective writings and discoveries. [2]

One of the most significant developments since the Statute of Anne, was the adoption of the Paris Convention of 1883 and the Berne Convention in 1886, a treaty between a number of countries and territories promoting a common base for copyright law [122]. Since 1886, the treaty has been revised a few times, and most countries of the world are signatories of the convention [122, 195]. The Berne convention provides two important functions, with regards to digital works:

1. Copyright protection from the time of creation, without the need to publish, register or even affix a © symbol to, the work.
2. Copyright protection across all countries who are signatories of the Berne Convention.

The first function means that a large proportion of works that can be considered protected by copyright law, are not necessarily meant for public consumption, and are actually produced and consumed inside companies and businesses. The second function means that the works are protected regardless of distribution across the Internet. However, because the specifics of copyright law is not uniform, an infringement in one territory may not be an infringement in another. This particularly affects older works, whose copyright protection might have expired in some territories and not in others.

Apart from legal protection of information, in [61], Camp argues that copyright law indirectly offers a few other functions. For the authors, continued reproduction, distribution and archiving, apart from monetary compensation, has created their reputation, allowed for a generally high level of persistence, enforced their human right of expression and asserted their moral rights to the content. For the users of the work, this process has created a trust in the information contained in the work and allowed a mechanism to establish the integrity of a given copy of the work. Furthermore, because users are able to annotate copies of these works, as well as cite, criticise and incorporate the information contained in these works in further works; the original argument used by Defoe and others that copyright encourages learning is further enhanced.

Furthermore, copyright law has also allowed users to trade in the imprints they have purchased (with some restrictions); and allowed authors to trade their economic rights. These rights have allowed a greater degree of economic activity in intellectual property, not only by enriching authors and publishers; but also traders who sell the works to the consuming public.

Thus, any system that is designed to protect digital information, will not only have to take account of the limits of the legal protection offered by copyright law, but also retain the functions provided by the current environment.

6.2 COPYRIGHT & DRM: LICENSING

If copyright law is still to govern the boundaries of what rights can be regulated by DRM, the question arises of how the relationship between copyright law and DRM should be defined. Many, like Harper [97], Leafer [122], Litman [125] and Masango [130] have argued that in the past copyright law has been updated in parallel to the invention of new technologies. However, Litman argues that, in the United States at least, pressure to update the law has come from the publishing industry rather than the consumers. Consequently, most of these updates were largely motivated by interested stake holders from industry instead of lawmakers. For this reason, copyright law is highly complex, involving a number of exceptions; but largely ignoring the public's input [125].

Litman further argues that the public can only obey laws that they believe in (for example, laws against murder) or respect laws they feel have a symbolic power, even if ineffectual (like antidrug laws). But because copyright law is too complex to understand for most people, and it does not wield any symbolic power, copyright law is largely ineffective. Instead, Litman argued for the need of a new paradigm to regulate information which is not based on reproduction as current copyright law, because the current reproduction paradigm is inherently flawed in the digital arena.

While reproduction is often referred to as the fundamental right in copyright [122], there are other exclusive rights conferred by copyright law. As discussed by Leafer in [122] and Stokes in [184], copyright law awards copyright holder, monopoly in regulating 5 types of rights:

- **Right 1:** Reproduction
- **Right 2:** Distribution
- **Right 3:** Adaptation
- **Right 4:** Public performance, display, or communication of the work to the public (via broadcasting for example)
- **Right 5:** Rental to the public

The last right is the most varied, with some countries disallowing rental in some forms of works (for example software cannot be rented in the USA [125]). Furthermore, the monopoly on the adaptation right is also limited; with many countries defining adaptation with the intention of creating a new work (such as parody), for purposes of citation or review as fair uses.

6.2.1 Reproduction and Distribution in the Digital Arena

Litman argued that the reproduction monopoly right poses inconsistencies in the digital arena [125]. But it is not only reproduction that poses such inconsistencies, but also distribution, and in fact, the inconsistencies could be argued for all the rights categories.

Reproduction of digital data is a very simple task, requiring only memory space. Most high level operating systems can perform this task, with minimal effort from users. With the advent of high speed Internet access, distribution of digital data is fast, efficient and inexpensive. Furthermore, new distribution systems, such as peer-to-peer networks, have led to a large increase in the efficiency of distribution [15].

But legislating the illegality of distribution and reproduction for copyrighted digital data is counter-productive. The very act of viewing digital data requires multiple reproductions and distribution cycles. For example, viewing a text file stored on a compact disc on a computer, involves the reproduction of the data in the system memory, transformation of the data to a suitable format for presentation, and possibly another reproduction in the graphics card of the computer before presenting it to the user.

If the computer is a terminal in a distributed client/server system, the data will need to be distributed across the network to the terminal, which will make a reproduction of the data in its own memory before presenting it to the user. If the file is accessed over the Internet, it could be stored at various points on the transit path to the consuming device – and at all points before the final presentation, the data is usually fully recoverable.

For this reason, some copyright legislation defines a copyright infringement only if the reproduction and distribution takes place for purposes other than presenting or delivering the data in its final form, like Article 33 in the preamble of the European Copyright Directive [5], or the application of copyright law to diffusion services in the South African Copyright Act [4]. However, these legislations do not address the scenario when the data is recovered from memory (for example during final presentation) or how the data should be treated after the task is complete. These ambiguities and shortfalls lead to some interesting questions like, does the caching of data in proxy servers constitute a copyright infringement, or, how long can data be cached in a router or proxy server before becoming a copyright violation, or, if data is recovered from a router or proxy server's cache, does it constitute a copyright violation?

Furthermore, creating reproductions and distributing digital data is very easy, cheap and requires little effort and knowledge. For this reason, it is very difficult to prevent reproduction of digital data or to stop its distribution. As long as computing devices are componentised, it is easy to bypass most restrictions on reproduction and distribution of digital data. Currently, it can be argued that only mobile phones provide difficulty for the average user for these tasks, and increasingly, even these come equipped with data cables and multiple connectivity options to make these tasks simpler.

There is also ambiguity as to what constitutes a purpose of presenting or delivering a data. If the digital data is copied onto a CD for the purpose of using it at an offline location, by the legal owner, does this constitute a copyright infringement – since the data was copied for the purpose of presenting. The boundary between fair use and illegal copyright infringement is usually determined through the calculation of the economic harm to the copyright holder; but there is no standard measure on how this value is calculated. In fact it has been argued that not all instances of unauthorized distribution of copyrighted content on the Internet have led to an actual loss in sale [190, 33] (because for example, the work is no longer published and therefore not available for sale).

Reproduction over the Internet also creates a problem in that copyright legislation differs in different countries and territories. Does a copyright infringement occur if the reproduced data is distributed from a foreign territory, and if so, what is the value of such a reproduction? How does an average user differentiate between data that is stored in different territories – and if it is illegal, is it reasonable for the average user to know that?

For these reasons, it is impractical to enforce restrictions on reproduction and distribution of digital data, and most DRM systems do not have any explicit functionality to restrict reproduction and distribution of digital data.

6.2.2 Abuse: Protecting what should not be protected

While copyright law provides rights holders with a monopoly, it must be stressed that the monopoly is not absolute. In addition to fair use scenarios, already introduced in section 6.1 and discussed further in section 6.4, the other limit to this monopoly is the time limit of the monopoly. As discussed in [56] and [72], the case for perpetual monopoly for copyright was raised and even implemented in the early versions of copyright legislations, but subsequently monopoly granted to the rights holders by copyright law has been limited.

However DRM can be used to enforce restrictions on works that are no longer under copyright – either because the work was under copyright when it was produced, or because the work was placed under protection during production. For example, in [109] there was a discussion raised by Hugh Huddy from the Royal National Institute of the Blind, where bibles were distributed in the form of protected Adobe e-books. However, there were no provisions available for the use of the e-book by the visually impaired because Adobe's screen reader function was disabled by the protection. Furthermore, there was no possibility for affected users to contact the distributors to reverse this protection. In this particular case, protection was being abused in both forms: copyright law generally gives an exemption to allow transformations for handicapped persons and secondly the protection was applied to content in the public domain.

6.2.3 Camp and Copyright Management

In [61], Camp argues that functions provided by DRM systems are fundamentally different to the protection of copyright, and copyright management. Camp argues, that while DRM provides for publishers and authors receiving compensation for their efforts; and largely allows for most of the author's rights discussed previously; the user's experience is largely diminished. The author argues that many DRM systems limit access to their works, limiting the potential for trading both the goods themselves or access to the goods, and substantially lowering archiving potential of protected work. Furthermore, annotation is usually impossible; although the integrity of the work is well assured. Even with programs that circumvent the protections, such as DeCCS, many of the features available on printed works are absent from their digital counterparts.

6.2.4 **"Personal Use"**

In [143], Mulligan et al. described a set of rights that the user expects to have. These rights do not necessarily have any legal backing, but are practiced none the less by the majority of users. The authors categorised the rights they discussed into three categories: portability, excerpting and limited relationship and interaction with the copyright holder. Under copyright law in some countries, limited portions of the above mentioned categories are defined under various fair use clauses. For example, in the South African Copyright Act, excerption for the purpose of citation or review is mentioned under the fair dealing section [4]. The authors did not explore other uses that could fall under "*personal use*" nor did they discuss how users felt about DRM systems that limit these uses.

Of the personal uses discussed by Mulligan et al., lack of portability is perhaps the biggest concern raised by consumers. In [113], Kalker traced portability as the key underlying reason behind movements such as "anti-DRM day" and the backlash against DRM in general. We have previously discussed four different forms of portability in section 2.3, and the importance of these forms of portability will depend on the application of the DRM system.

Portability is not necessarily a legal issue, and copyright law for the most part restricts users from transforming media into different formats. However, portability is a major business and technical issue, and needs to be addressed in DRM systems. While the business issues are not discussed in this dissertation, we do discuss the technical requirements for interoperability.

6.2.5 **The Focus on Use and the Licensing Paradigm**

Instead of regulating the rights allowed in copyright law, DRM systems work by imposing restrictions on how the user *uses* the protected work. For example, most music stores restrict users to playing music they have bought to a small number of different PCs and devices. Copyright laws, like the South African Copyright Act, place very few restrictions on how a copyrighted work is used [4]. In most cases, the act provides restrictions for public broadcast, public performance and adaptations. Similarly, Chapter 2 of the European Copyright Directive (ECD), Rights and Exceptions, is divided up into four sections – reproduction rights; rights of communication to the public of works and rights of making available to the public other subject-matter;

distribution rights; and exceptions and limitations [5]. There are also no specific restrictions on how a user makes use of a copyrighted work. This is the same scenario in US copyright law, and for this reason Samuelson observes that DRM systems go beyond copyright law [173]. Thus, as we have discussed previously in [31] and by Camp in [61], DRM is about neither enforcing copyright nor providing copyright management, but rather about restricting how protected works are used.

Copyright law does allow a mechanism to regulate usage, through licensing agreements between the copyright holders and the user(s). Since we have established that DRM systems enforce restrictions on the usage of protected works, they easily fit into the conventional copyright legal system through the licensing provisions. The licensing model can be applied to both enterprise and consumer usage of DRM, and thus, provides an uniform legal model for DRM. In the rest of this chapter, we discuss the requirements and regulation of licensing in DRM systems.

6.3 Licensing in DRM: Requirements and Regulation

Licensing is a contractual process between the parties involved, i.e. the copyright holders and the consumers of the license. Note that the consumer of the license does not necessarily have to be the consumer of the work itself – for example, a television channel licenses a movie to play on its network, but the work is consumed by the viewers of the channel.

With respect to works aimed for the general public, the purchase of physical media did not create any contract regulating the use of the physical media. Copyright law allowed the user almost any non-commercial use of the physical media. Licensing of works to the private persons is therefore a new business model for copyright holders, and thus needs to discussed as such. Furthermore, the consuming public also needs to be aware of the differences between the two approaches.

Another aspect of contracts is that they are often arrived at after negotiations between the parties. Current DRM systems offer only shrink-wrap or click through licenses, and do not allow the users any input on the terms of the use licenses. Thus users could make use of copyright tribunals to protest current terms in use licenses especially if the work is available only in the digital form. The prospect of negotiating terms and conditions in a DRM use license itself create new business models and offers solutions to certain problems encountered by current DRM systems. We discuss some of the legal aspects of negotiations in this section, but the bulk of the technical details regarding protocols for negotiations and implementation of negotiations, please refer to chapter 7.

6.3.1 Requirements for Contracts

In [178], Sharrock defines a contract as "*an agreement which creates an obligation or obligations between the parties to the agreement*". In [64], Christie defines a contract as "*an agreement that is intended to be enforceable at law*". Both Sharrock and Christie further discuss, that while all contracts are agreements, not all agreements are contracts, as agreements do not necessarily give rise to obligations (and are thus not required to be enforced). Sharrock defines an agreement as "*meeting of minds (consensus ad idem) on all aspects of the transactions*", and thus for a contract to be valid, all parties must fully understand and interpret the contents of the contract, in the same manner. There is another form of contract, first discussed in Roman Law and detailed in Gaius' Institutes (in 160AD), when a contract arises from a statute of law [64]. However, we are not aware of any such legislation that affects DRM systems; and will not discuss it any further.

For a contract to be valid, there needs to be an agreement between the parties, it must be complete, lawful[2], possible[3] and both parties must be of contractable age[4]. Many of these issues are not addressed by current DRM systems.

In the DRM arena, there are usually two parties to an agreement – the rights holder (usually represented by the distributor like Apple, Napster etc.) and the user. Typically, in a DRM use license, the user agrees to pay for the right to access and use a digital resource, which is provided by the other party, under the terms and conditions laid out in the use license. This creates an obligation from the user in addition to the obligation created by the rights holder/distributor with respect to the supply and quality of the digital resource. Thus, use licenses can be considered as a contract between the user and the rights holder/distributor.

6.3.2 Licensing and the Problems With Current DRM Systems

In DRM systems designed for the general public, many users do not understand the terms and conditions imposed upon them. For example, the INDICARE group's 2005 survey on European users of DRM enabled media reported that over 70% of their respondents did not know or understand the limitations [78]. In our own survey (which we discussed briefly in section 2.1), we reported a lower value, where 30% of the respondents did not understand or know of of the restrictions while a further 10% of the respondents only understood some of the restrictions [33].

[2] The contract itself should not involve actions that break the law. This could be problematic in the digital arena due to the scope of international law.

[3] Terms and conditions of a contract should be possible to achieve by both the parties concerned.

[4] For licensing agreements, a party could be younger than the contractable age but there is always a possibility that the contract could be cancelled if the minor's guardian objects. In such a case, all the parties need to return the objects exchanged.

Another factor, which we discussed in [33] is that distributors do not inform the consumers about the restrictions implemented on the protected files. The lack of information means that consumers are led to believe, and expect that they have the same freedoms and rights as they would with non protected analogue media. Furthermore, in some cases, the non disclosure of restrictions are of a bigger concern, especially when the DRM systems are security risks to the average user. These problems are clearly demonstrated in two DRM systems: Vodafonelive! and the SONY-BMG Rootkit CD Protection. Note, that these problems are not found in every DRM system – Apple's iTunes Music service for example, provides well publicised documentation on its music store.

Vodafonelive!

Vodafonelive! is a portal for most of the Universal Mobile Transmission System (UMTS, also referred to as 3G) services that fall directly under or through subsidiaries of the Vodafone Group. In South Africa, the portal is administered and run by Vodacom, a subsidiary of Vodafone. The portal provides a variety of services including music and video downloads for which the user pays a fee (debited to their mobile phone account). The content provided by the Vodafonelive! portal is protected using the OMA DRM 1 standard using the Forward Lock specifications. We have previously discussed OMA DRM 1 in section 3.4.

Despite marketing downloading of music tracks as similar to buying CDs, the Vodafonelive! system is different to consumer expectations. In particular, despite a high rate of mobile phone replacement [112], the data is locked to the phone, and cannot be moved to other devices or onto portable storage media. Consumers that wish to play the same music file on their different phones, or migrate their purchase to different phones are unable to do these actions. However, as we detailed in [33], consumers use media for a long period of time and would thus expect the same from digital media, especially as there is no change in format.

But the biggest problem with Vodafonelive! (in its South African deployment) is the lack of information regarding the restrictions. In the entire documentation, the restrictions are only mentioned once, and that is in the Frequently Asked Questions (FAQ) section under help. The restrictions are not mentioned in the terms and conditions during purchase, nor in the terms and conditions for use of the system listed on the main website.

Under these circumstances, it cannot be claimed that a "meeting of minds" has been reached. Thus, current DRM systems cannot claim to enforce any agreement, let alone any licensing agreement between the rights holders and the users. While it is true that some terms and conditions can be implied [178], DRM is a new technology not well known to the public; and thus cannot be deemed to be implied.

In our opinion, the users using Vodafonelive! are not legally bound by the DRM restrictions. However, OMA DRM is almost impossible to bypass, and most common users will not be able to do so. Furthermore, because the cost of the content is relatively cheap, and due to the financial position of Vodacom as one of the biggest companies in South Africa, it would not be financially feasible for most users to argue the restrictions in court. Thus, even though the users have a legal right not to have their content restricted, they do not seem to have a recourse to bypass the restrictions.

Sony-BMG Rootkit

In October 2005, Mark Russinovich, a computer security expert at the USA based SysInternals wrote details of a rootkit installed by a copy protected CD published by SONY-BMG [171] on his blog. Rootkits are programs developed to bypass standard operating system security protocol and interact directly with the operating system. For this reason, rootkits allow the program complete control of a computer, and can provide distributors of the rootkit free reign on the target system.

While the SONY-BMG rootkit did not seem to grant access to hackers, it fundamentally changed the operation of Windows XP, and the operating system's control over the computer's devices [171]. Furthermore, the license agreement[5] did not state the nature of the program and its potential harm to the operating system of the user. In either case, it is highly likely that such a use license would be ruled invalid, if contested. In effect, the rootkit was no different to most spyware and a major security risk for affected users, and there were subsequent attacks that were made possible due to the rootkit [123]. Because there was no easy way to remove the rootkit, security measures had to be introduced to combat the potential effects in many companies [194].

While the SONY-BMG issue was subsequently settled out of court, it does raise issues relating to trade practices. Like Vodafonelive! customers, the consumers were not properly informed of the full implications of using the product.

6.3.3 Form and Content of Licensing Contracts

Licensing contracts can be formless, and there is very little (if any) information that is mandatory in a licensing contract. However, in most legal systems, it is often recommended for a licensing agreement to have some information that removes ambiguities should a dispute arise. In addition to the details of the usage terms and conditions, we believe that use licenses for DRM systems should consider the inclusion of the following information.

[5] The complete EULA was subsequently posted by Mark Russinovich at: http://www.sysinternals.com/blog/sony-eula.htm (last accessed: 2006-04-09).

Jurisdiction and Dispute Resolution

Jurisdiction and the dispute resolution mechanisms go together and are very useful in resolving contract disputes speedily and efficiently. In dispute resolution, the parties resolve to go to a mutually agreed party to arbitrate their dispute. This is often a cheaper avenue to pursue than full litigation. Should it fail, or if a party does not wish to take this route, they can choose to sue the other party. Jurisdiction determines where a party can be sued.

Choice of Law

Because of the international nature of DRM, it should be possible to choose which law governs the licensing agreement. This is potentially very important in DRM. For example, in countries that have implemented the ECD, the rights holder does not have to provide for fair use if the work is provided under a licensing agreement [5, 79]. However copyright law in most other countries does not have such restrictions. Thus, it would be advantageous for music labels to base their licensing agreements under EU law rather than, say South African or Australian law. The choice of law would be influenced by the chosen jurisdiction.

Liabilities

Many contracts related to the provision of goods and services limit the liability of the provider (for example: “vehicle parked at owner’s risk”). Like dispute resolution, current products offered with DRM protection probably do not require such statements, but future works might. It could also be interesting to provide different pricing models according to the liability risk carried by the supplier. For example, if the license covers software, the rights holder could provide the software at a very low price but with no guarantees on performance while at a much higher price with guarantees on performance or number of bugs.

Time of Contract

Some recent laws governing electronic commerce, like South Africa’s Electronic Communications and Transactions (ECT) Act of 2002, provide guidelines to determine the time of a contract if the information does not exist on the contract [7] (using timestamps on email messages for example). However, not all countries have such guidelines, and parties may wish to draft their own time of contract. A time of contract is also crucial if the license is valid for a specified period (such as one year) or the parties are in different timezones.

Signatures

In most countries, valid signatures are not required for a valid contract. However, signatures are often seen as the primary mechanism for proving non-repudiation of the parties in a contract. In recent times, many countries have passed regulations authorising digital signatures (utilising encryption and message digest algorithms) as equivalent to the traditional signature. Digital signatures are also useful in proving the integrity of a digital object. Most RELs including ODRL already provide for digital signatures.

Lifetime or Duration

An offer usually lasts until either a specified time or a "reasonable time". After that, the offeror is not bound to honour the terms and conditions specified in the offer. However, in an electronic medium, there are very few (if any) precedents that can be used to determine a reasonable time for an offer. For this reason, it is useful (and good practice) to specify the lifetime of an offer.

Contracts can also be valid for a specific duration, and current DRM systems choose to have "never ending" use licenses. But, copyright law protects works only for a finite period of time, and use licenses cannot artificially extend the period of protection. Thus, the lifetime feature for offers should therefore be extended to also cover the final use license.

Offer, Counter Offer and Requests

An *offer* can be defined as a "*a proposal of certain terms of performance made with the intention of being agreed to by another person*" [178]. A *counter offer* could be then be defined as a "*a proposal from the offeree accepting the offer but on different terms*". In [178], Sharrock explains that a *request* is different to a counter offer, because a counter offer effectively voids the previous offer while the request has no legal effect.

However, offers and counter offers create specific requirements themselves. Specifically, electronic transaction law in some countries (like South Africa's ECT Act [7]) insist on certain regulations, especially regarding consumer protection from the party that makes the offer. Thus, when a client makes a counter offer, (s)he would be required to abide by such regulation removing certain features like the potential for anonymous negotiations (from the client).

Country of residence (for offeree)

Regardless of the nationality of the offeror, the offeree's country of residence places obligations for the offeror in terms of some laws especially consumer protection laws and tax implications. For example, in the case of a EU citizen, a rights holder outside the EU cannot store personal information unless they have signed a safe harbour agreement.

Fair Use Policy

In [31], we introduced a mechanism to enable a degree of fair use through the use of negotiations, which is discussed in more detail in section 6.4. This element in the license allows the offeror to detail their approach to fair use. This approach will remove the ambiguities present in current DRM systems.

6.3.4 Privatisation of Copyright – Buying vs Licensing

In [125], Litman has argued against the licensing approach and the increasing privatisation of copyright. However, others like Harper [97], have argued that full contractual DRM systems can spawn different business models. This argument is partly supported by the current music download business[6].

The leading music store, Apple's iTunes, has a less restrictive rights policy when compared to its peers. Currently, music stores making use of DRM make use of two different business models – the rental model in the subscription music stores and the pay per song or album model. But these business models can be further extended. For example, would there be consumers willing to pay lower than current per song downloads if the rights they have are further restricted? Conversely, would consumers be willing to pay more for music that has very few (if any) restrictions? Results in the 2005 INDICARE survey suggests that consumers are interested in these different business models, as long as the pricing is fair [78].

Licensing of copyrighted works is already used to regulate the commercial use of copyrighted work. In South Africa, a *copyright tribunal* is available to anyone who has disputes with regards to licensing agreements, and we discuss the potential future uses of copyright tribunals in section 6.3.5.

Current DRM systems offer only shrink-wrap or click through licenses, and do not allow the users any input on the terms of the use licenses. Thus users could make use of copyright tribunals to protest current terms in use licenses especially if the work is available only in the digital form. In section 6.4, we discuss two mechanisms on how such fairer license terms can be achieved in DRM.

With these recourses, we do not think a licensing approach would necessarily be bad for the consumer. However, there need to be clearer regulations and strong enforcement of these recourses

[6] We assume that other factors such as usability of the system and the types of portable music players supported also play a key role in how successful an online music download business is.

for the balance to be maintained.

6.3.5 Copyright Tribunal and License Arbitration

Some copyright laws, like in South Africa [4], provide for a *copyright tribunal*. The tribunal's main function is to settle disputes arising between rights holders and current or potential licensees. Many other countries do have copyright tribunals, but it is not a provision under the Berne Convention [195], and thus may not be an universal construct amongst Berne Convention signatories. Under current law, almost any person or organisation can be a potential licensee and thus approach the tribunal to settle license disputes.

There are three areas that the copyright tribunal can be particularly important in regulating DRM use licenses (section 33, part 2 and 3 of SA Copyright Act 98 of 1978 [4]):

1. A potential licensee is refused a license even when the licensee meets the requirements set out in the license.

2. The rights holders have refused or failed to grant a license "*in the circumstances it is unreasonable that the license should not be granted*".

3. The charges, terms and conditions set out by the rights holders for the license are unreasonable.

The above regulations should allow the public a fairer use license policy, particularly in allowing for a fairer price and terms of access. This is particularly important when DRM is used to protect scholarly works, and the rights holders have a monopoly on the particular subject.

As far as we are aware, use of the copyright tribunal for settling license disputes has been minimal, at best. However, if the use of DRM does spread, the number of licensing agencies and licensees will increase substantially, implying a potential increase in the use of the copyright tribunal. The function and the appropriate staffing of the copyright tribunal will therefore play an important role in the success of DRM.

A further potential role for copyright tribunals could involve the setting up of use license templates for the common DRM use licensing scenarios. This could be beneficial for both consumers and rights holders – rights holders know the boundaries of the minimum requirements for consumer DRM use licenses and consumers know the minimum they can expect from rights holders.

6.4 Fairer Use

As discussed earlier, we have used the term "*fair use*" to refer to the class of circumstances that allow users to circumvent copyright law. While fair use varies widely from country to country, in most countries; fair use is ultimately decided in a court of law; and depends on the circumstances and the nature of the work. Because of this, referring to US copyright law, Felten argued that enabling fair use automatically in computer systems is almost impossible [82]. Despite this, many updated copyright laws, like the European Copyright Directive (ECD), requires technological measures that protect copyright law to enact the exceptions allowed by the ECD [79, 5].

Dusollier however argued in [79] that the ECD has a loophole in the regulations, as rights holders could potentially side step the fair use provisions all together if the work is "*made available to the public on agreed contractual terms in such a way that members of the public may access them from a place and at a time individually chosen by them*" [5]. Since we have argued that DRM is the enforcement of licensing agreements, this clause means that DRM use licenses based in the EU do not have to provide for fair use.

Another argument on fair use revolves around the appropriateness of current fair uses in the digital environment. In [130] and in [97], Masango and Harper respectively argue that current fair uses are unsuitable for the digital environment. Masango argues that, in the past, invention of new technologies led to the evolution of both copyright law and copyright exceptions. However, there have been no such updates for digital works. Both argue that if fair use is to be successfully implemented in a digital environment, a new definition of fair use in a digital environment is needed.

Ultimately, what is desired by consumers, and required by contract law, is that the DRM use licenses are fair to both parties. While fair use, in its current form, is almost impossible to implement, it should be possible to implement a mechanism that can enable a high degree of fair use scenarios. In this section, we detail two such approaches, which we have collectively termed "*fairer use*", which we have discussed in detail previously in [31]. This section discusses the overall approaches only, and the technical details on how these approaches can be implemented are discussed in chapter 7.

6.4.1 Fairer Use through Negotiations

As discussed earlier, use licenses in DRM systems are contracts between the rights holders and the users. However, while most contracts have inputs from the both parties; use licenses have only the rights holders' inputs. In fact, current DRM systems do not have mechanisms to allow the end user to have input on the terms and conditions of the use license. In [142], Mulligan et al. commented on the fact that there are no RELs that allow users to express their needs, and the protocols for creating use licenses for end users do not take any inputs from the end user. Thus to allow users to communicate with the rights holders, there is a need for both a protocol and a language to express the communication.

We think that users should be allowed to negotiate the use license with the rights holder. In [107], Jamkhedkar and Heileman also proposed the use of negotiations to allow for flexible security levels for DRM packages. As far as we are aware, our paper, [31], was the first description of the syntax and the protocol for conducting negotiations in DRM systems.

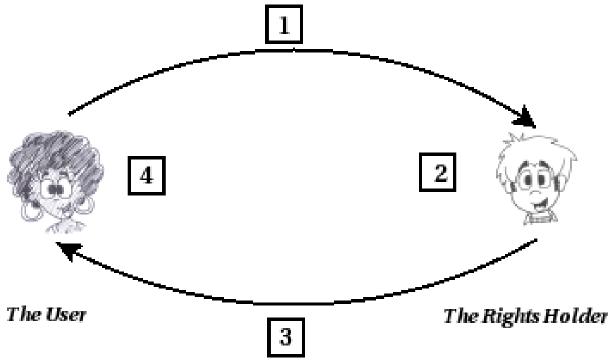


Figure 6.1: Simple Negotiations Protocol

There are two parties in a simple use license negotiation – the rights holder (or his/her proxy) and the end user. Negotiations make use of a simple *request-response* model, and can be broken up into four simple steps, as shown in figure 6.1. This is a simple overview, and a more detailed protocol is discussed in chapter 7.

Step 1: The end user requests the rights holder for a set of rights (or changes to an existing set of rights).

Step 2: The rights holder evaluates the request. The rights holder can ask for additional credentials before granting a request. For example rights holders may allow only accredited journalists the right to excerpt from a protected work.

Step 3: The rights holder presents the user with one or more sets of rights that match closely to the requested rights.

Step 4: The user can request refinements to the offered rights packages until he or she is satisfied (and start from step 1) or can choose one of the rights packages on offer.

This process immediately offers a new business model; allowing different sets of rights at different prices. For example, a basic rights package for a movie on demand service could allow the user to play the movie on three different devices expiring in three days from the rental date. The user could then be allowed to purchase additional rights to increase the number of devices or extend the expiry date. With a separate use license, this model can be further extended to allow end users to "renew" the rental after the initial purchase has been completed. It also allows for added flexibility – a journalist could have rented the movie for personal reasons, but could then decide to include it in his next movie reviews segment. The journalist could then request excerpt rights using a journalist credential. These use models are not possible in current DRM systems, and our mechanism allows DRM to enhance and facilitate "licensed usage" through accommodating greater degrees of flexibility.

In [185] Su et al. also discussed two important components for electronic negotiations:

- a formal protocol, and
- an effective AI agent to carry out the task.

In [45], Bartolini et al. added two further requirements for automated negotiation:

- a language to define rules of negotiation, and
- a language to express negotiation proposals.

In chapter 7, we discuss formal protocols for DRM negotiations, as well as modifications required to current RELs that would allow bi-directional communication, and thus allow them to be used as a language to express negotiation proposals. We also discuss a few strategies that could be used by AI agents to automate negotiations, but do not explore this subject in too much detail.

6.4.2 Fairer Use through the use of Credentials

Most RELs already have syntax that allows for rights to be exercised under certain circumstances. ODRL for example has a *constraint* construct, which can be used to limit the parameters of a certain right – for example, a print right can have a constraint, number, with a value of 5 to restrict the end user to only print the media a maximum of 5 times. The *condition* construct in XrML provides similar functionality.

Our second approach, uses a *credential* constraint/condition in a REL. The parameters of the constraint would then be credential ticket type that the user needs to produce to exercise the right. We acknowledge that this approach would require a very strong identity management system to work and that is a deterrent for adoption of this mechanism.

A single credential constraint is not enough but neither is a list of credentials. Rights holders may wish to require different sets of credentials for a particular right. For example, rights holders may require either a user to present both an academic and a researcher credential or a journalist and a Reuters credential to access a certain right. A list of credentials will not be able to express this, and thus a credentials model must be able to express and evaluate boolean expressions.

A credential system can also be extended to provide semi location-based authentication through secondary (and maybe untrusted) credentials. A secondary credential could be issued to prove that the user is using a specific machine or is present at a specific location. It would be very difficult to allow trusted credentials servers for these functions, but even if they are untrusted, they can be used to provide some limits for the end user. Using the journalist example again, the rights holders may decide only to offer an excerpt right for journalists if the journalist makes use of a machine logged onto the news organisation's network during excerption. The primary credential would prove that the user is an accredited journalist, while a secondary credential could be used to prove that the journalist is logged onto the network before being allowed to excerpt. This approach could be very useful for enterprise DRM systems, where the aim is to control the use of sensitive corporate data.

Alternatively, secondary credentials could be used to indicate what type of work the user is engaged in. Thus the academic can request a local credential server to provide a "research" credential before making an excerption. While this scenario could be implemented in the workplace (as it allows employers to monitor the productivity of the users indirectly), it is not feasible for use in a private home; as any such online system has great potential to infringe the user's privacy.

We discuss the technical implementation of credentials for DRM systems in more detail in section 10.3.

6.5 Other Legal Issues

While copyright law and contract law are the overwhelming base for licensing agreements for DRM, other legal issues also need to be considered for the implementation of DRM systems. Most of these issues affect consumer DRM systems only, but enterprise DRM systems still need to take some of these issues into account, especially when works released in a consumer environment are used in an enterprise setting (for example, music used in an internal corporate presentation). This section briefly highlights a few of these areas.

6.5.1 Globalisation Effect of the Internet

The Internet is a global medium, and data available on the Internet is largely accessible to all countries and territories in the world. However, legal systems differ in every country, and thus laws differ in every country and territory. Even with international treaties such as the Berne Convention, do not imply uniformity in the implementation and interpretation of laws. For this reason, there is a large conflict, and resulting ambiguity, in the expectations and the protections offered to both rights holders and consumers for digital media.

With respect to DRM protection for digital media, the following factors need to be considered:

1. **Differing Fair Use Clauses:** As mentioned already, fair use clauses differ from country to country, and only a few fair use cases can be considered universal.
2. **Duration of Copyright:** Even with the Berne Convention, the longevity of copyright differs from country to country, and on the nature of the media. DRM protection cannot be applied to works that are not protected under copyright.
3. **Age of Majority:** The age from which persons can enter into contracts differ from country to country and may also depend on the cultural and social environment of the person. In most cases, if a dispute arises based on the age of one of the contracting parties, all the parties need to return the objects exchanged. This subject becomes more complex if the work is educational in nature (and required for the student's education) but the student cannot get his/her guardian's consent for acquiring the work.
4. **Special Licensing Provisions:** As discussed in [122] and [125], in the USA, copyright law has special licensing provisions for various industry groups. Such provisions might also occur in other countries, and rights holders need to take this into account while setting up their licensing agreements.

Rights holders can take two different approaches to overcome the above problems, both with their advantages and disadvantages. The first approach, similar to the approach adopted by existing DRM systems, is to create licenses catered for specific countries and scenarios. The main advantage of this approach, is that it would allow the rights holders to exert the maximum control over their works, while consumers get use licenses that are geared specifically for their country. The main disadvantage of this approach is firstly the cost in establishing multiple contracts and secondly the potential loss in sales where use license contracts are not available.

The second approach is to create a use license that meets the minimum requirements for all the affected countries and territories. However, this means that some special cases may not be catered for and the amount of control available to the rights holders is lowered. However, it is possibly a cheaper approach and easier to manage.

6.5.2 Trade Practices

Laws regulating trade practices affect DRM protected works in two different ways. Firstly, contracts should be fair, and trade practices legislation govern the creation and enforcement of contracts where one party is contracting from a position of strength. Since rights holders have monopoly over their content, they are naturally in a position of strength; and thus these laws hold particular importance. For example, the Consumer Affairs Act of South Africa (Act 71 of 1988) has a clause regarding "*harsh terms of contracting*" [178], a provision arguably broken by the Sony-BMG rootkit (discussed in section 6.3.2) where the DRM system introduced security vulnerabilities for the consumer's system.

Secondly, in certain instances, as discussed by Leafer in [122] trade practice legislations dictate the distribution of copyrighted works, when imported works are priced cheaper than a locally produced and distributed version. This is a problem from the beginning of copyright law, as discussed by Birrell in [56] and Deazley in [72]; and this problem increases with the globalisation effect of the Internet – since there are no effective boundaries nor any physical delivery.

6.5.3 Privacy

With the potential for greater control for rights holders, there is also the potential for the rights holders to monitor the exact usage of protected works. In an enterprise scenario, this is a particularly welcome scenario, and enterprises have a high degree of freedom in monitoring and tracking the use of enterprise resources without raising issues related to the employees' personal privacy. However, in the consumer scenario, there is a lot less freedom, and privacy is a big concern.

As discussed previously, Mulligan et al. identified limited relationship and interaction with the copyright holder as one of the expectations consumers have from their past interactions [142]. Furthermore, privacy legislations in some countries give consumers just a limited relationship, and thus license agreements that require monitoring of usage could be found to be illegal. Furthermore, in [148], the OECD[7] discusses how privacy makes business sense and thus encourages its member countries to implement consumer friendly legislation, and also encourages

[7] The Organisation of Economic Cooperation and Development (OECD) comprises of 30 industrial nations, and is involved in conducting research and policy recommendations for the member nations.

enterprises to respect consumer privacy.

Technologies such as stenography can allow for passive monitoring and allow for the identification of infringing copies. This approach could be used as a balance between the rights holders' needs to monitor for infringement and the consumers' right for privacy.

6.5.4 Electronic Transactions Act

A few countries have enacted legislation to regulate electronic transactions, and many of these legislations are based on the UNCITRAL Model Laws[8] for electronic transactions [188]. These laws regulate the requirements for businesses that transact online over the Internet, and are intended to provide greater protection for both the consumer and the business. Sometimes, these laws also incorporate consumer protection laws based on either existing laws, or practices in other areas (the EU for example, has a number of such legislations). In this section we discuss some of the requirements set out by The South African Electronic Communications and Transactions Act (ECT Act) of 2002 [7].

1. **Opportunity to review and withdraw from a transaction (*Chapter VII*):** Requires the service to allow the consumer to review the product and withdraw from the transaction. Preview systems used by some DRM systems would satisfy this requirement.
2. **Business details of the service provider (*Chapter VII*):** The service provider is required to provide information such as postal address and contact details even if the service provided is completely virtual.
3. **Security and Privacy Policy (*Chapter VII and VIII*):** The service provider has to explicitly state how consumer data collected during the transaction is stored, secured and used.
4. **Mechanisms to correct material errors (*Section 20*):** If an electronic system (either automated or through another person) does not allow the consumer to correct errors during the transaction, or notify the transacting party as soon as possible of an error, the agreement is invalid.

Many e-commerce systems already fulfil such requirements, but these requirements are not necessarily standard and differ from country to country. For example, Europe has stricter regulation on how and where consumer data is collected, stored and used.

[8] The United Nations Commission on International Trade Law (UNCITRAL) drafted a set of "model laws" addressing electronic transactions. Model laws are not treaties or actual legislation, but provide a foundation for countries to base their laws upon.

6.6 Summary: Towards a Legal DRM System

In this chapter, we discussed how DRM systems are effectively license enforcement mechanisms as opposed to copyright management or copyright control mechanisms. Thus, for a DRM system to operate legitimately in the current legal framework, it will need to focus on a number of important areas:

1. Use licenses are contractual agreements. For a contract to be legally valid, there needs to be agreement between both parties. Thus both parties to the use license must be *fully* informed about the contents of the contract.

2. Use licenses themselves must follow the boundaries (if any) set by copyright and other laws. Use licenses cannot enforce restrictions that do not exist in copyright law (for example protection beyond the time period allowable by copyright law).

3. Use licenses need to be fair, and not impose restrictions that are harmful, unnecessary etc.

4. DRM systems should allow for a dispute resolution process, preferably through existing mechanisms such as copyright tribunals.

5. Media DRM systems can offer services that resemble fair use, and should implement mechanisms that can allow the user to exert these rights.

6. Media DRM systems require minimal interactions with the rights holders, including no active monitoring of consumer's usage of protected works.

As already discussed in detail by Camp in [61], DRM systems do not offer copyright management functionalities by default. DRM cannot be used to enforce copyright, or for functions such as archival, since licensing by its nature is time limited. For this reason, functions desired for archival purposes are best achieved without the use of DRM.

CHAPTER 7

A NEGOTIATIONS FRAMEWORK FOR DRM

In [154], Pruitt defines negotiation as "*a form of decision making in which two or more parties talk with one another in an effort to resolve their opposing interests*". Sharrock defines a contract as "*an agreement which creates an obligation or obligations between the parties to the agreement*" [178]. Thus negotiation is an important component of the contractual process, and can be defined as the process by which an agreement is formed. In chapter 6, we have already discussed how DRM can be seen as the enforcement of electronic contracts, and thus there is a need for a negotiation framework for DRM systems.

In [185] Su et al. discussed two important components for electronic negotiations:

- a formal protocol, and
- an effective agent to carry out the task ([185] concentrated on AI agent strategies for this task).

In [45], Bartolini et al. added two further requirements for automated negotiation:

- a language to define rules of negotiation which can be used by agents to evaluate negotiation proposals, and
- a language to express negotiation proposals.

In this chapter, we introduce two different protocols to conduct negotiations. Both of these protocols are more complex than the simple protocol we discussed in section 6.4. We also use Coloured Petri Nets to prove the integrity and robustness of the negotiation protocols, by proving that there are no deadlocks and that all the desired states of the protocol are reachable. Thus, our modelling assumes that the states correctly process the information they receive. We have previously discussed parts of this chapter in a proposal we submitted to the Digital Media Project's 9^{th} General Assembly in 2006 [32] and in [36].

In chapter 8 we show how these protocols can be integrated into RELs. The study of negotiation agents is already a very well-known discipline, and can be considered a study in its own right. For this reason, we do not discuss agent decision making strategies or the languages that can be used to define their behaviour.

7.1 A NOTE ON TERMINOLOGY

Thus far, we have discussed DRM from the perspective of a consumer interacting with a producer. However, as we defined in section 1.1, a producer is an entity that is involved in modifying or creating DRM protected works. In the DRM chain, there is another means to categorise users, and that is in respect to the use license associated with the protected work.

A ***licensor*** is a person (natural or legal) who has been authorised by the rights holders of the resource to license the access to the resource to prospective consumers.

A ***licensee*** is a person (natural or legal) who has concluded a license agreement with a licensor to access the resource.

Note, a licensor can be a producer but is not necessarily one, and a licensee can be a producer, a consumer or both.

7.2 RELATED WORK IN ELECTRONIC NEGOTIATIONS

There has been a great deal of research into negotiation protocols, tactics and other related fields. In economics and mathematics, game theory models have been used to discuss different tactics that could be used to arrive at the most rational outcome [110, 55]. In the social sciences, there has been a lot of research in how different parties act during negotiations and how these actions affect the eventual outcome [160, 154]. In Computer Science, the bulk of research in negotiation has focused on agent negotiation, focusing on agent decision tactics, efficiency and protocols [110]. Some of this research has been extended to e-commerce scenarios [185, 25].

Negotiation protocols have also been used in other aspects of electronic communication. For example, the Secure Socket Layer (SSL) protocol has a negotiation component in its handshake protocol to set-up encryption and MAC algorithms [183]. However, in most of these cases, the number of negotiable factors are small, and there are often strict guidelines which dictate the result of the negotiation.

In [160], Raiffa discusses a number of factors that affect negotiation strategy and protocol. These factors include:

1. The number of parties,
2. parties negotiating on behalf of a group,
3. repetitiveness of the negotiation process and its effect on reputation,
4. the number of terms being negotiated,
5. 3^{rd} party involvement

The use of negotiation in the consumer or end-user environment is almost non-existent. In [80], Elfatatry et al. suggest the use of negotiation to tailor software products in a Web Services environment, but do not specify any protocols for such a service. This chapter details the use of negotiations in an end-user environment, and this poses a further challenge, not catered for in most negotiation scenarios – the protocol must be able to cater for three different types of interactions:

1. The human end user and an agent representing the license holder
2. The human end user and a human representing the license holder
3. An agent representing the end user and an agent representing the license holder

Most negotiation protocols are developed for agents, and they are often moulded specifically for the agent's purpose. Furthermore, these protocols often incorporate the agent's decision making processes. For these reasons, existing negotiation protocols developed for agents are not fully appropriate for the scenarios we present in this chapter. The negotiation protocol must however take into account all the factors discussed by Raiffa, and the protocols we present in this chapter are similar to argumentation-based models described by Jennings et al. in [110] while the REL model which we discuss in chapter 8, has similar properties to the logic-based negotiation languages detailed in [197, 191].

7.3 Types of Negotiations

As discussed earlier, negotiation can be defined as a process whereby a contract is concluded. In [185], the authors distinguish three types of negotiations:

1. **Bidding:** The buyer specifies the service or product that he needs and asks for bids from potential suppliers. The buyer then selects one or more of the suppliers to provide the service or product. Currently, no DRM system can support bidding, and we discuss bidding in DRM systems in section 7.5.

2. **Auction:** The auction can be viewed as the opposite of bidding where the supplier of the product or service promises to perform the service or deliver the goods to the customer with the highest bid. There are a variety of auction types, and current DRM systems should be able to handle auctions as price is the only variable component of an auction. For this reason, we do *not* discuss auctions any further.

3. **Bargaining:** Bargaining is the most flexible type of negotiation allowing all the parties involved to dynamically change the terms and conditions to suit their needs. A lot of research in negotiations has focused in this area, and we discuss bargaining in DRM systems in section 7.6.

Current DRM systems only support transactions where the suppliers determine a fixed price for the product under fixed terms and conditions. There is no scope for bargaining. While these types of transactions are largely fine for most media oriented digital data (for example music), they are not useful for automating business use of digital data or for more non-media oriented use of digital data (for example large volume purchases for academic usage). As we have already discussed in section 6.4, bargaining could also be used as a mechanism to assure fairer usage of digital media, and opens up possibilities for allowing “fair use” scenarios not possible with current DRM systems.

7.4 Requirement for Negotiation for DRM

As discussed previously in section 7.2, Raiffa detailed a number of factors that need to be considered when determining strategies and protocols for DRM. In this section, we look at how these factors apply to DRM systems, and also discuss the requirements specified by the Digital Media Project (DMP)[1] in [74]. The DMP is a collaborative project between various interested parties, including device manufacturers and software vendors, to create a standardised platform for interoperable media DRM systems [63].

[1] http://www.dmpf.org/

7.4.1 Factors affecting Negotiation in DRM

The number of parties

As identified by Bartolini et al. [46] and also by the DMP [63], there are a number of parties involved in the DRM value chain. The DRM value chain, as defined by the DMP, is shown in figure 7.1. The negotiation protocols we present are focused on *two* parties – one party that has the right and ability to conclude use license agreements (usually the rights holder) and the consumer of the license (which could also be the producer or service producer).

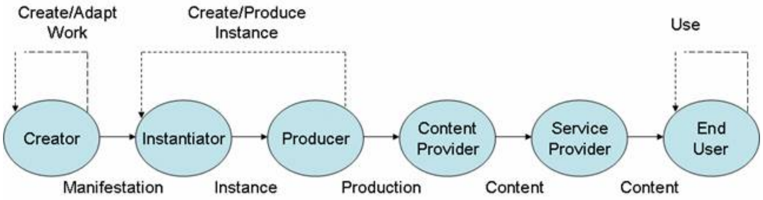


Figure 7.1: The DMP DRM Value Chain [63]

Parties negotiating on behalf of a group

Although only two parties are negotiating the use license, both parties could be representing larger groups. For example, in consumer music, a parent could be acquiring music that will be accessed by all the members of his/her immediate family.

Repetitiveness of the negotiation process and its effect on reputation

This factor is more relevant to the strategy employed by the parties and not to the protocols themselves. We do not discuss it in detail in this chapter. However, it must be noted that using reputation of the parties as part of the negotiation process could create different business strategies, not currently pursued. For example, users associated to well known organisations could be seen as more reputable, and thus given more rights than lesser known users. In [107], the authors proposed the use of reputation to determine the level of security for DRM packages.

The number of terms to be negotiated

Pruitt defines an issue as the topic under discussion [154]. Thus, in a DRM license, each permission or right is a separate issue, unless they are bundled together (e.g. the licensee can get read and play rights together or not at all). Individual terms of a specific right (e.g. restrictions on the number of times that right can be exercised) is the subject of negotiation, and not separate issues. Issues are often linked together, and an issue can often influence the negotiation position of a party for another issue (for example, the number of users covered by the license can affect the price of the license). Because, the number of possible rights in a DRM use license, and other issues like validity of the license and the number of users covered by the license, DRM use license negotiation can become very complex.

3rd party involvement

Previously, in section 6.3.5, we proposed the use of copyright tribunals to arbitrate use license disputes, and this could be a possible 3rd party involved in the negotiations. One of the findings in our survey (the details which we previously discussed in section 2.1, and can also be found in [33]), we found that users are more willing to use negotiation as a mechanism to enable fair use if the process is monitored by independent 3rd parties.

7.4.2 DMP requirement for negotiations

In [74], the DMP listed the following requirements for negotiations in DRM systems:

1. End-users can express their agreement or disagreements with proposed license terms.
2. The protocol shall support changes to any parameter of the license.
3. The protocol shall support automatic negotiation of license terms.
4. At every step a human readable license must be provided.
5. The protocol shall enable the setting of certain parameters as non subject of negotiation.
6. The protocol shall allow the determination of the degree of confidentiality (no eavesdrop) of the protocol.
7. The protocol shall not require revealing the real identities until the protocol has been successfully concluded.

7.5 Bidding

Bidding does not have much impact for consumer-oriented DRM products, but could have a massive impact in business transactions conducted over the Internet. For example, an advertising agency could be looking for classical music to accompany their television advertisements. For this purpose, they create a tender inviting musicians, bands etc. to supply the music under certain terms. Prospective parties can then formulate their offers, possibly offering different terms (for example a larger catalogue of music) and their prices. The advertising agency can then consider the offers and make their choice accordingly. This type of scenario cannot be handled by current DRM systems, nor by the simple negotiation protocol we previously discussed in section 6.4.

7.5.1 **Process**

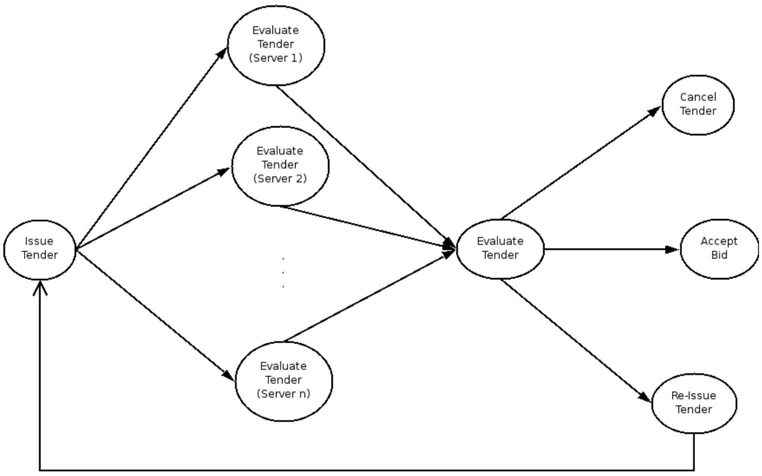

Evaluate Tender (Server 1)
Evaluate Tender (Server 2)
Evaluate Tender (Server n)
Issue Tender
Evaluate Tender
Cancel Tender
Accept Bid
Re-Issue Tender

Figure 7.2: Bidding Process

A flowchart showing the bidding process is presented in figure 7.2. The prospective licensee issues a *tender* outlining the data they are interested in, the terms and conditions they would like for having access to such data and the time limit for responses to the set of possible respondents. The bidders then evaluate the tenders and create appropriate offers (if interested) outlining their price. The interested licensors can then evaluate the various offers and choose a winner, re-tender or close down the tender process and not choose any of the bids. Feedback is not mandatory, although it could be good business practice to outline why a tender is rejected.

7.5.2 **Protocol**

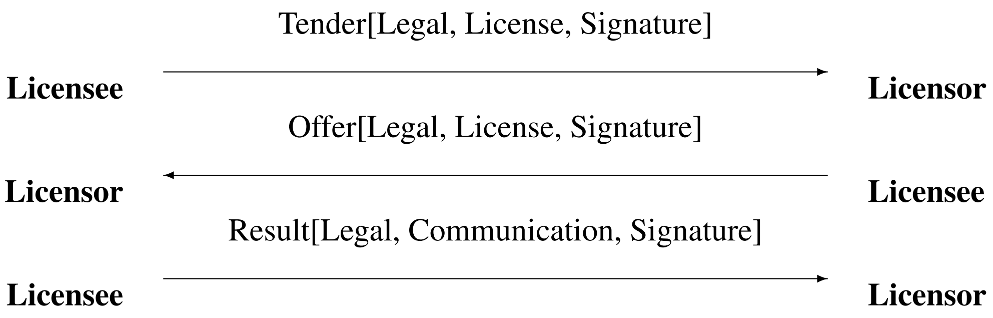
By its nature, bidding is not an interactive, instantaneous process. As shown in figure 7.2, there are only three parts to a bidding process:

1. The *announcement* of the tender requesting offers

2. The *submission* of offers

3. The *notification* of the outcome

For this reason, the bidding protocol has the following simple high level structure.


Tender[Legal, License, Signature]
Licensee
Licensor
Offer[Legal, License, Signature]
Licensor
Licensee
Result[Legal, Communication, Signature]
Licensee
Licensor

Message Format

The protocol is envisaged to run as SOAP messages between the two parties, with each step comprising of an XML encoded message between the parties. The exact representation of the message is discussed in more detail in section 7.7. The message can have a number of different components, and these are detailed below:

- **Communication:** This component comprises of the message elements that can be used to communicate between the parties, including the acceptance and rejection of offers.

- **Legal:** Use licenses are legal contracts, and there may be legal terms that need to be expressed, which do not form part of the main use license, for example liability disclaimers.

- **License:** The license forms the core component of the use license, and is effectively the terms and conditions being negotiated (like permission to play, read, modify etc).

- **Signature:** It is of paramount importance, that integrity of the communication is maintained. While some communication protocols have integrity checking, this is not guaranteed. For this reason, we contend that it is necessary to have a digital signature component for the message. Digital signatures also provide for non-repudiation, and are now considered legally binding in many countries. The entire message should be signed.

The same message content and notation is used in our bargaining protocol detailed in the next section.

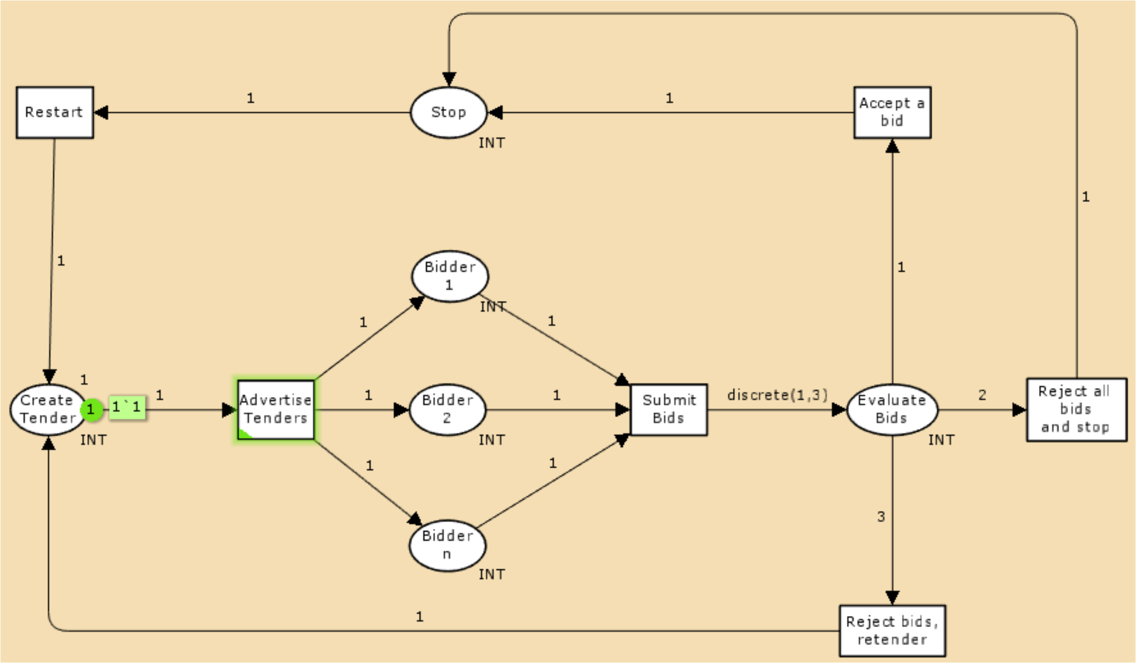


Figure 7.3: Bidding Petri Net

7.5.3 **Modelling**

Figure 7.3 shows the Petri net model for our bidding protocol. Certain steps in the protocol can have a number of different outcomes, which we have modelled using the discrete function in CPNTools. This function is a shortcut random number generating petri net function available in CPNTools, and has been used to keep the net compact.

We have used the state space analysis component of CPNTools to analyse the net. Due to the small net, analysis is not difficult and is rather straight forward. Analysis shows us that the Petri net is live, reachable and safe. This means that every state of the protocol is reachable with no deadlocked states. Because the net is safe, there will be no performance degradation in repetitive iterations of the protocol, although that scenario should not arise in bidding.

7.6 Bargaining

Bargaining is the most complex negotiation strategy, but it is also the most powerful as it can be used not only for new use licenses but also to change existing use licenses. We have previously discussed the use of bargaining as a mechanism to enable fairer use licenses for consumers in section 6.4, and as far as we are aware, this is the only technical approach to fair use for DRM systems.

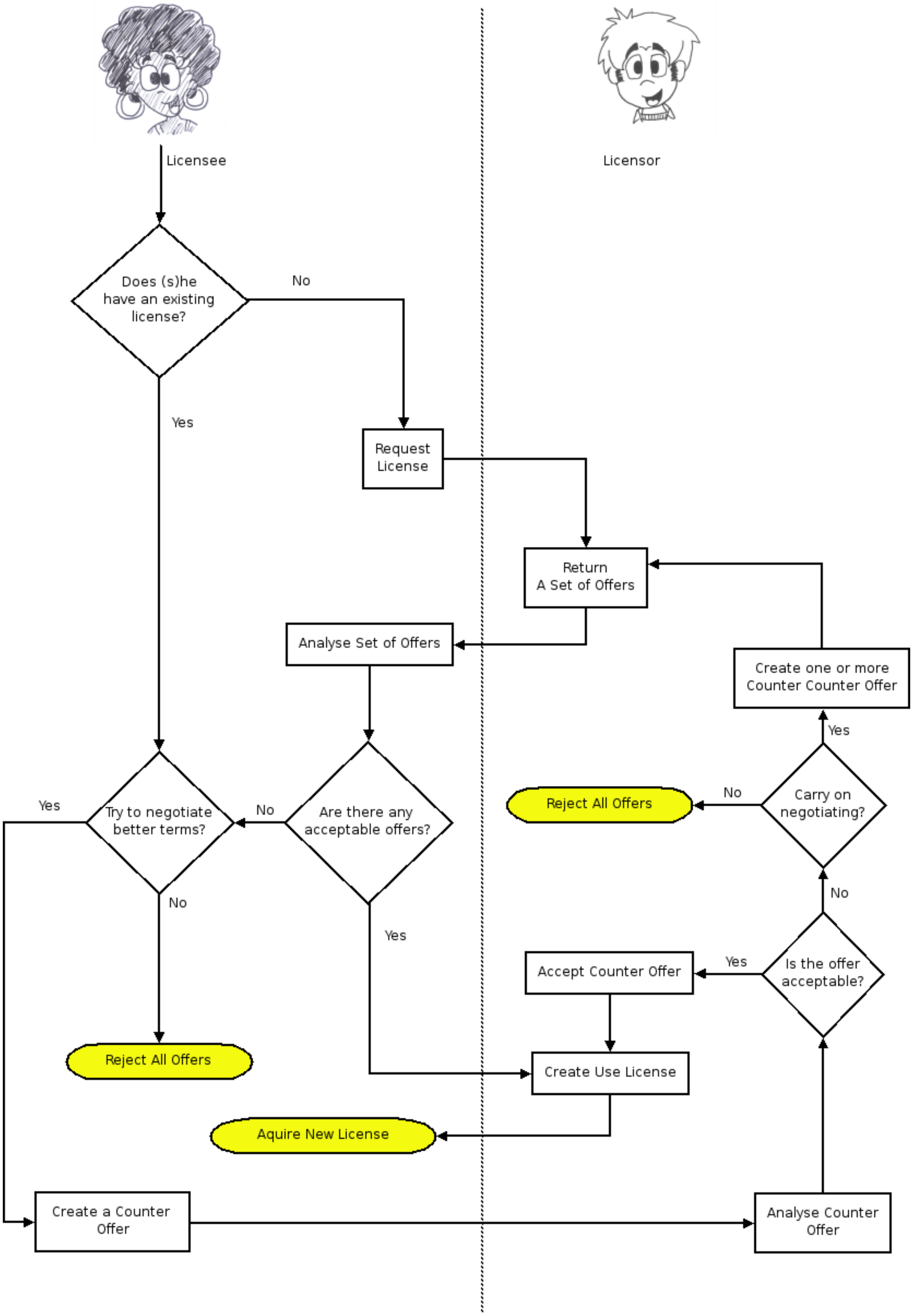


Figure 7.4: Flowchart for a bargaining protocol

7.6.1 **Process**

A bargaining protocol is shown in figure 7.4 and is a refinement of the simple negotiation protocol we have previously discussed in section 6.4, and is similar to the argumentation protocols discussed by Jennings et al. in [110]. The protocol assumes that the licensee communicates with the correct licensor. It also assumes that the licensor is initially willing to offer a license agreement to the licensee. Catering for scenarios in which the licensee communicates with the wrong licensor or the licensor is unwilling to, is not shown in the protocol, but is trivial to handle.

Analysis of a request, or counter-offers from licensees should depend on the business scenario presented by the licensee. The protocol can handle anonymous licensees, but anonymity may not be an ideal bargaining position for the licensee. For example, a well established licensee, with a long history of business association, could get more favourable terms and conditions compared to a new licensee. In such a case, the knowledge of the licensee's identity is required before the licensor makes their decision. Similarly, a licensee wanting a transaction of high monetary value could be allowed a discount. In this case, anonymity of the licensee can be preserved unless there is a legal requirement forbidding anonymity of the licensee. As discussed earlier, the business logic used for negotiation is not discussed in this proposal, but needs to be addressed before the full power of bargaining is realised.

7.6.2 **Protocol**

As shown in figure 7.4, the bargaining protocol is more complicated than the bidding protocol, mainly due to the two different start and end possibilities. But, since one of the start possibilities is also part of the general bargaining protocol (does the licensee have an existing license), we need to show only the one path for the protocol.

1. The licensee requests a new license. The identifier for the digital resource can be communicated using the "communication" element.

$$\textbf{Licensee} \xrightarrow{\text{Request[Legal, Communication, Signature]}} \textbf{Licensor}$$

2. The licensor sends back n offers (where n is a positive integer) to the licensee.

$$\textbf{Licensor} \xrightarrow{n,\ \text{Offer[Legal, License, Signature]}} \textbf{Licensee}$$

3. After analysing the offers, the licensee can do one of the following:

    (a) Accept one or more of the offers. Use of multiple licenses for the same digital resource is currently not handled by any DRM system, but there should be no reason why this should not be possible.

    $$\textbf{Licensee} \xrightarrow{n,\ \text{Acceptance[Legal, Communication, Signature]}} \textbf{Licensor}$$

    (b) Reject all the offers, and quit negotiations.

    $$\textbf{Licensee} \xrightarrow{\text{Rejection[Legal, Communication, Signature]}} \textbf{Licensor}$$

    (c) Reject all the offers, and enter negotiations based on one of the offers, or create counter offers from scratch. In the later case, a Counter-Offer is created instead of a Request. The licensee can create multiple requests or counter offers.

    $$\textbf{Licensee} \xrightarrow{\substack{\text{Rejection[Legal, Communication, Signature]},\ n, \\ \text{Request[Legal, License, Signature]}}} \textbf{Licensor}$$

4. Depending on the licensee's response, the licensor does one of the following:

    (a) If the licensee rejects all offers, the licensor closes down the negotiation system.

(b) If the licensee proposes a counter offer, the licensor can:

   i. Reject all proposals and close down negotiations.

$$\textbf{Licensor} \xrightarrow{\text{Rejection[Legal, Communication, Signature]}} \textbf{Licensee}$$

   ii. Reject all proposals, but carry on with negotiations by creating offers that try to satisfy the counter offers.

$$\textbf{Licensor} \xrightarrow{\substack{\text{Rejection[Legal, Communication, Signature]}, n, \\ \text{Offer[Legal, License, Signature]}}} \textbf{Licensee}$$

   iii. Accept one or more of the proposals (see the next step).

(c) If the licensee accepted an offer, or the licensor accepts one or more of the counter proposals: All other offers are deemed to have been rejected.

$$\textbf{Licensor} \xrightarrow{n, \text{ Agreement[Legal, License, Signature]}} \textbf{Licensee}$$

5. Depending on the licensor's actions:

   (a) If the licensor wishes to close down negotiations, the licensee also closes down its negotiation system.

   (b) If the licensor wishes to carry on with negotiations, the protocol goes back to step 1.

   (c) If the licensor offers agreements, store the agreements and close down the negotiation system.

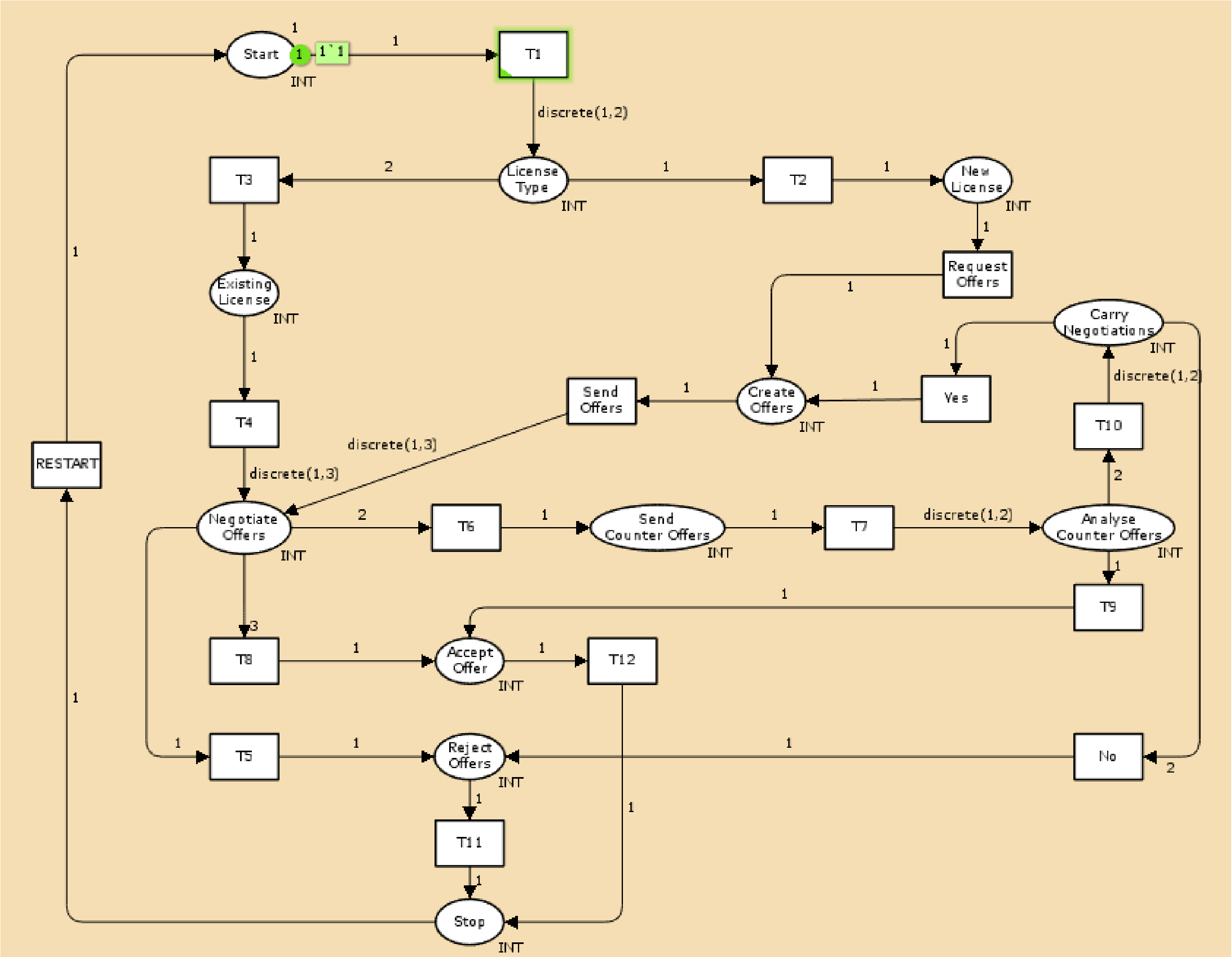


Figure 7.5: Bargaining Petri Net

7.6.3 Modelling

Figure 7.5 shows the Petri net model for our bargaining protocol. Like our Petri net model of our bidding protocol, certain steps in the protocol can have a number of different outcomes, which we have modelled using the discrete function in CPNTools.

We have used the state space analysis component of CPNTools to analyse the net. Analysis shows us that the Petri net is live, reachable and safe. This means that every state of the protocol is reachable with no deadlocked states. Because the net is safe, there will be no performance degradation in repetitive iterations of the protocol, and because bargaining inherently involves a number of iterations, this is an important result.

However, there is a possibility for a circular deadlock condition if the licensee refuses all the offers offered by the licensor but continues to attempt to negotiate a favourable license, while the license holder continues to offer new license terms. This does create an added problem should the licensee be automated as it could create a potential denial of service attack.

To avoid such a scenario, it could be useful to extend the protocol to keep count of the number of negotiation requests from a particular licensee during a particular session and stop negotiations after a predetermined number of negotiation cycles, but such decisions would depend on the business logic and the scenario of the negotiations.

7.7 RELs and the Expression of Negotiations

Rights expression languages (RELs) are often referred to as the most crucial component of DRM systems [107]. For this reason, there has been a lot of focus on developing RELs. In [70], Coyle distinguished RELs into three different categories:

1. expression of *copyright*
2. expression of *contract* or *license agreements*
3. control over *access* and/or *use*

We have already discussed that DRM should not be seen as a mechanism to enforce copyright law but rather as a mechanism to enforce contracts on access and usage of digital data in chapter 6. Thus, the primary role of a REL is not to express copyright but rather to express contractual agreements. Consequently, DRM systems can then be seen as the enforcement of such contracts. In [70], Coyle concluded that none of the current RELs – including general purpose RELs such as ODRL and MPEG-21/5 – have the full functionalities required for such purposes. For example, the lack of bi-directional expression (from the licensee to the licensor) has been cited as a deficiency in RELs by Mulligan et al. in [142].

The lack of bi-directional expression in RELs has meant that there is no formal mechanism for the licensee to communicate to the licensors. This has meant that one of the crucial parts of the contractual process – negotiation – is not possible. We have previously proposed bi-directional extensions to ODRL 1.1 in [30], as well as extensions for XrML in [29]. Furthermore, as we discussed in section 6.4, negotiations can be used as a technical solution to enable "*fair use*" in DRM systems.

We believe that negotiation of use licenses is closely linked with representation of the license, and thus Rights Expressions Languages (RELs) have a crucial role in this regard. This removes the need to convert between expression of negotiation terms and the final agreement, thus reducing errors that are possibly introduced during translation. However, as discussed by Jamkhedkar et al. in [108], RELs are already too complicated, and addition of negotiations increases their complexity. Thus, there is a need to create a balance between catering for negotiations inside RELs and the complexity introduced by such a move.

The protocols we have presented in this chapter are not dependent on the representation of the terms and conditions; and can actually cater for any type of license, as long as it has a clear structural base (i.e. a REL) such that parties can propose valid changes to the terms and conditions. Thus, we have provided negotiation support without requiring explicit support from the REL. However, there is some functionality that requires support from the REL, such as stating individual terms and conditions as non-negotiable (one of the requirements discussed by the DMP in [74], see section 7.4 for more details).

7.8 Negotiation Requirement Analysis

In section 7.4, we discussed the various requirements for negotiations. In this section, we examine how well these requirements are satisfied by our protocols.

7.8.1 Raiffa's Factors Affecting Negotiation

The number of parties

Our protocol enables negotiations between any two parties in the DRM value chain as identified in [46] and [63].

Parties negotiating on behalf of a group

Our protocols do not require the negotiating parties to be individuals. There is however a requirement for the language representing the negotiations to cater for multiple individuals to be represented as a single party or a group.

Repetitiveness of the negotiation process and its effect on reputation

Our protocol does not place a limit on repetition of the negotiation process, and as we discussed in section 7.6.3, this can introduce its own problems, such as denial of service attacks, and not just affect reputation. This can be solved by limiting the number of repetitions allowed by the negotiating parties. Since we do not discuss how agents evaluate the negotiation elements, the timing of when an agent decides to stop the negotiation could be related to the reputation assigned to its negotiating partner.

The number of issues

The protocol is not dependent on the number of issues involved.

3rd party involvement

The protocol does not explicitly support 3rd party involvement. However, if the third party is seen as the communicating medium, then it can be integrated into the protcol. Alternatively, third parties can also be involved post-negotiations (through arbitration or court action for example, when a party could claim the other party did not act in good faith, or evaluate a proposal in a fair manner). The use of signed messages in our protocols make it easier to prove the authenticity of any transaction.

7.8.2 Satisfying DMP Requirements

End-Users can express their agreement or disagreements with proposed License terms

Both licensees and licensors can express their agreement or disagreement over proposed license terms, and the protocol also supports the proposers presenting multiple licensing options.

The Protocol shall support changes to any parameter of the License

Our protocols make use of the complete available licensing infrastructure, and thus allow either party to change any parameter available to them.

The Protocol shall support automatic negotiation of license terms

The protocol does not prescribe how negotiation decisions are arrived at. Thus, it is possible for either agents or humans to make the negotiation decisions for either parties.

At every step a human readable license must be provided

The use license comprises a very tangible part of each step of the negotiation process. For this reason, it should be possible to extract a human readable license at any step of the negotiation, and there have already been some notable advances in this regard. For example, in [151], Peig and Delgado discussed the conversion of licenses based in ODRL 1 into human readable licenses.

The protocol shall enable the setting of certain parameters as non subject of negotiation

Our protocols can accommodate any REL to express the licensing terms as part of the protocol. Thus, the control of individual license terms require support from the underlying REL to cater for this functionality.

The protocol shall allow the determination of the degree of confidentiality (no eavesdrop) of the protocol

The protocol can easily be run through a secure communication tunnel (example SSL) or through encrypted SOAP messages. The level of security can be determined by individual parties concerned.

The protocol shall not require revealing the real identities until the protocol has been successfully concluded

The identity of the licensor is always required, but there is no such requirement for the licensee. In fact, some RELs like the proposed ODRL 2.0, allow for totally anonymous end use licenses. The protocol does not require licensee's identities, but licensee's identity could help in negotiation decisions (for example, frequent customers getting better deals).

7.9 Example

We show a complete bargaining example using LiREL (discussed in chapter 8) in appendix D.

7.10 Summary

Negotiation is a crucial component of the licensing process, but no current DRM systems support negotiations. There has been substantial research into negotiation protocols and strategies, but they have not been applied to DRM systems.

There are three major forms of negotiations: bidding, auctions and bargaining. Since auctions only negotiate the price of the license, and there are well known auction support systems (such as eBay), we have focused on the remaining forms of negotiations. In this chapter we have detailed bidding and bargaining protocols and motivated their correctness and completeness through the use of Petri net modeling. Our protocols can use any REL to express the terms and conditions during negotiations, and this approach eliminates the need for additional translation between the language of negotiation and the use license.

Thus, we have presented two of four required components for automatic, electronic negotiations as discussed in literature (formal protocol and expression of negotiation proposals), and our protocol supports the inclusion of the other two components (effecting agent to carry out negotiations and a language to define agent strategies).

CHAPTER 8

A FORMAL MODEL FOR DRM

As discussed by Jajodia et al in [106], access control has two distinct parts, which are dependent on each other to function properly:

1. The means to represent the policies controlling access to a resource.

2. The means to implement the policies correctly and effectively.

Because the latter is almost a purely logical process, most access control models have been based on some sort of logical notation. As already discussed in section 4.3, there are currently no formal models describing RELs (which represent policy representation in DRM systems) and similarly there are no formal models for interpretation and implementation of RELs, although, as discussed in section 4.3, Halpern, Pucela and Weissman have been involved in this area in different contributions [95, 157, 158]. In this chapter, we address these particular gaps.

Before we formulate a new access control model for DRM, there is a need to differentiate DRM from existing access control models, and explain why existing models cannot express DRM systems completely. We do this in section 8.1. Following this discussion, we discuss some requirements unique for DRM systems in section 8.2.

Since RELs are used to express policies, we present a brief discussion of the various approaches used to define different RELs. We then present a formal model for RELs in chapter 8.4, before discussing enforcement of DRM use licenses in section 8.5.

We have previously discussed parts of this chapter in [37].

8.1 Existing Access Control Mechanisms

Currently, there are three widely accepted access control models: Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role-Based Access Control (RBAC). In this section, we will briefly review these models, before we discuss how DRM differs from these models.

8.1.1 DAC, MAC and RBAC

In DAC based systems, access to data objects is restricted based on the identity of subjects and/or groups to which they belong [144]. Furthermore, in DAC based systems, a user with access to the protected data can delegate access to other users. DAC does have a number of specification levels, and criteria B3 to A1 requires implementors to allow creators to control the propagation of access delegation to other users (which would be necessary for DRM systems) [144], but as discussed by Reid and Caelli, modern operating systems implement a simpler version of DAC [164] which does not allow such controls. They argued that DAC based operating systems allow ordinary users of the system to define their own security. By granting ordinary users this ability, a user could reconfigure the security policy of the system to subvert the DRM protection. The authors also point out the inability of mainstream operating systems to support the principle of least privilege. Since system privileges are based on the users' identity, any program executing on behalf of a user is granted the same access control privileges as the user. There are no efficient mechanisms for restricting users' access control rights.

In MAC based systems, and in the associated Multi-Level security (MLS) systems proposed by Bell and LaPadula [52, 51], access control is assured through a central security administrator, and thus ordinary users of the system are prevented from reconfiguring the computer's security policy [165]. However, for the purposes of DRM, the rights holders (or the owners of the data) are not guaranteed any control over the consuming device.

In MAC based systems, access control is based on the user's credentials, with users classified under a hierarchical structure, discussed in the Bell-LaPadula model [51, 52]. The hierarchical structure allows greater rights for some users while allowing lesser rights for other users. Protected objects are classified under this structure, and the object does not determine the level of access for the user. This creates a problem for DRM systems, where it is sometimes necessary to determine access according to the nature of the object as opposed to the classification of the user. For example, the author of an article can be classified as a rights-holder and a reader. However, the classification of the author as a rights-holder means that he has access to other works that are not necessarily his own.

The third and newest, popular access control model is Role-Based Access Control (RBAC), first described by Ferraiolo and Kuhn in [85], and subsequently detailed further by Ferraiolo et al. in [84], as well as Sandhu et al. in [174] and von Solms and van der Merwe in [192]. Ferraiolo and Kuhn argued that access to data should be determined by the function of the users in relation to the data, which are usually defined by roles users play in an organisation [85]. For this reason, a role-based access control model is more suitable than the DAC or MAC based approaches that were available at the time. von Solms and van der Merwe further argued that the role-based approach is a combination of the resource-based approach (as found in DAC) and the user-based approach (as found in MAC) [192].

A pure role-based approach is however not suitable for DRM, for two reasons. Firstly, a pure role-based approach may not be able to distinguish access depending on the function of the object (as opposed to the function of the user), which would create a problem similar to the author-user problem discussed earlier. Secondly, the definition of roles is determined by individual organisations, and these roles vary from organisation to organisation; and sometimes differ within organisations. This problem could be solved by defining access roles with respect to the organisation (or department); but this would severely restrict portability of sensitive data between organisations.

8.1.2 **Differences between DRM and Existing Access Control Models**

The main difference between DRM and traditional access control however, remains on the boundary of control. Traditional access control models operate on an object within a defined boundary: either a system or organisation. DRM however aims to operate on objects that do not have any defined boundaries, and thus across different systems and organisations.

The definition of the boundaries in existing access control models determine user management of these systems. Traditional access control models are strongly coupled with user management, and it is therefore possible to specify the complete range of resources accessible to a particular user or role.

Since DRM does not operate in a defined boundary, such a specification is not easily made. Instead, a DRM policy should aim to specify whether a particular user has access to a particular resource. It is possible for a producer to keep track of which users have access to a particular resource, but will never be able to track *all* the resources accessible by a particular user; unless the user management is completely bounded to the DRM system. However, it should be possible for DRM systems to have no direct relationship with user management; as long as the user management provider is trusted by the producer.

Role and group membership evaluation is also different in DRM systems. In RBAC and MAC, the enforcement mechanism has to decide whether a user is part of a specific group or fulfils a specific role. Alternatively, the enforcement mechanism can ask another system (usually the user management system) for help in making such a decision.

However in a DRM system, the consumer is not guaranteed to be online, nor can the consumer's device be trusted to make such a decision (since group and role membership is dynamic). Thus, the user management system has to provide proof of such membership. For this reason, hierarchies associated with RBAC and MAC user management are not directly relevant to DRM systems. The user management system associated with the DRM system has to cater for such functionality, but the DRM system itself does not care for the structure of such hierarchies. This is a significant difference in the functionality of DRM when compared to RBAC and MAC.

Another effect of the lack of boundaries, is the lack of a single centralised authority that defines and regulates the access control rules. In this dissertation, we propose a general framework for DRM, and thus our framework is able to cater for multiple authorities that specify access controls, even if they use a single enforcement system.

8.1.3 XACML and RELs

While RELs have had no formal foundations, this is not the case for other access control specification languages, where there are numerous contributions. In [119], Kudo and Hada described a formal model for a XML based access control language: XML access control language (XACL), which can be considered as a fore-runner to the OASIS standardised eXtensible Access Control Markup Language (XACML) [89].

XACML is a generic access control specification language, and should therefore be able to specify DRM use licenses. However, as discussed by Guth in [93], generic languages cannot guarantee reliability in covering all the requirements, although they do provide a high degree in flexibility.

Kudo and Hada identified the following elements in the primary policy definition of XACL:

- the *object*,
- the *subject* wishing to acccess the object,
- the *action* the subject wishes to perform and
- the *context* in which the subject wishes to perform the action.

Other schemes, such as the discussions by Jajodia et al. [106], Dai and Alves-Foss [71] and Ferraiolo et al. [83] use similar main components, with the addition of *roles* but without taking context into account. As already discussed in chapter 6, DRM use licenses are contractual agreements. Thus, use licenses need to be equivalent to contracts in form; and thus require additional information not required in exisiting access control specifications. We discuss these additional requirements next, in section 8.2.

8.2 Additional Requirements for RELs

As we have previously argued in chapter 6, DRM should not be seen as a mechanism to enforce copyright law but rather as a mechanism to enforce contracts on access and usage of digital data. In such a view, the primary role of a REL is not to express copyright but rather to express contractual agreements between the user and the rights holders regarding the terms and conditions of accessing the digital resource. This legal model applies to enterprise and consumer DRM systems. While contracts are generally formless, they do have certain features which need to be represented in a REL model; which is a formal representation of the syntax and semantics of the language.

1. **The Licensor:** Current access control policies only denote the subject of the policies. However, a contractual agreement requires the specification of the party that will provide the service or product, and the subject can in fact be anonymous. The licensor does not have to be the actual rights holder (or even a producer), but rather a party that has been given the right to provide licenses to other parties.

2. **Agreement and Obligation:** As defined earlier, a contract is an agreement between two or more parties, creating obligations for the parties to uphold [178]. Agreements give rise to obligations (and hence becomes a contract) when penalties are declared should a party not fulfill its part of the agreement; i.e. the consumer is penalised if they do not pay (the access to the work is removed for example) or the producer is penalised if the quality of service promised is not delivered (the rights holder refunds the user for example). In [174], Sandhu et al. discussed role-based access control models that specifically did not handle obligations, because of the complexities involved. The requirements specifications for ODRL v2 also has a discussion on the need to include obligations associated to contracting parties and individual permissions [104].

3. **Contract Constraints:** In section 6.3.3 we detailed a few other legal considerations (such as period of validity) with respect to contracts in DRM use licenses. These can

be considered to be constraints affecting the entire contract, and may include other types of constraints, such as the number of devices that can be used by the user.

4. **Delegation of Rights:** A delegate can be defined as "*a person authorized to act as representative for another*" [24]. In rights delegation, the current licensor delegates the licensee the authority to act as a licensor to a set of other licensees. If possible, the licensor should be able to control every part of the use license for delegation. Delegation is important, as it allows the original rights holder to control delegation down the DRM value chain discussed in section 1.1.

5. **Support for Third Parties:** Contracts can specify third parties, who can be appointed in various capacities such as mediators, monitors, escrow agencies etc. Note that third parties in the contract are not involved in the agreement itself.

As discussed earlier in section 4.3, in [108], Jamkhedkar et al. categorised different features available in current RELs, and proposed that most of these features be removed. We discuss these categories in detail below including their effect on the REL model.

1. **Authentication Protocol:** Authentication is a vital component of any access control model. While the protocol may not be necessary, an access control model will still require information relating to the identities for various users and resources. Furthermore, as DRM operates in a global space, these identities must also operate in a global namespace.

2. **Payment Mechanisms:** The terms of payment is an obligation, and thus must be expressible as such. Thus, there is no need to cater for payment as a separate component of RELs.

3. **Rights Enforcement:** This refers to the possible semantics implied by the REL (for example, if the right is to play a song once, when does the counter get incremented). Jamkhedkar et al. proposed a separation between the expression and interpretation of access control rights, and thus proposed that rights enforcement should not be coupled with rights expression.

4. **Content Tracking:** Quite a few technologies have been proposed to try content tracking as part of DRM. For this reason, RELs have developed mechanisms to express these requirements. Content tracking can be part of the description of the resource itself, or part of the terms and conditions. It does not need to be part of the access control model, but does need to be catered for.

5. **License Management:** License management refers to a number of issues, like the delegation of licenses and aggregation of licenses. As discussed earlier, delegation needs to be part of the access control model, but other functions such as aggregation could be removed.

6. **Negotiation Protocol:** As discussed in chapter 7, the negotiation protocols are largely REL agnostic. However, some aspects of negotiation require support from the REL, but this is a matter of vocabulary and not the syntax and semantics of the REL. One exception is the attribute "*negotiable*", attached to every element of the offers, indicating whether the element is negotiable or not. We do not incorporate this attribute into the model directly, but needs to be incorporated in implementation. The default state of this attribute should be"*true*". Since a contract is also the end result of a negotiation process; and the REL must be seamlessly integrated with the negotiation protocols without imposing too much overhead.

While Jamkhedkar et al. have recently promoted a move to minimise the requirements for RELs, published requirements for ODRL are quite numerous [104], and most of these were published before their discussion. Some of these rights, such as delegation of rights have been discussed; but some REL specific requirements remain important. We have identified the following requirements, which have an impact on the REL model.

Req 1.2: ***Support the transfer of rights for content that is aggregated/dis-aggregated*** When there is delegation of rights, it may be necessary to delegate rights for only parts of a resource. The access control model needs to be powerful enough to express this.

Req 1.7: ***Provide a "NOT" expression*** In RELs, and in general, most access control models operate on granting access explicitly stated and denying any rights not stated. Jajodia et al. defined such policies as *open policies* [106]. In [104], the requirement for a "NOT" expression is motivated by the need to express agreements in the fashion of "allow rights for all actions except x". Such a requirement is analogous to *closed policies* defined by Jajodia et al. However, as we discuss in our model later, neither open nor closed policies make sense in the DRM space.

Req 1.8: ***Support rights and duties for all contract parties*** Obligations are not restricted to specific permissions, but could be tied to the entire agreement, and should be applicable to all the parties in the agreement and not just the licensee.

Some of the scenarios described as part of context by Kudo and Hadi [119] have been discussed above. The model for DRM we present here does not take any further contexts into account, and thus we do not include context as a separate entity in our model.

8.3 A Licensing REL: LiREL

In [90], Gonzalez commented that RELs can be seen in a spectrum, as shown in figure 8.1. At one end, there are RELs that try to represent copyright ideas and thus can express legal ideas such as fair use. On the other end of the spectrum are RELs that express access control rules. As we have discussed in chapter 6, DRM systems are not for the enforcement of copyright laws; but rather an enforcement of licensing terms and conditions, thus they are also a form of access control. For this reason, we think RELs should be able to express a legal contract, but at the same time, that contract should be able to define access control policies.

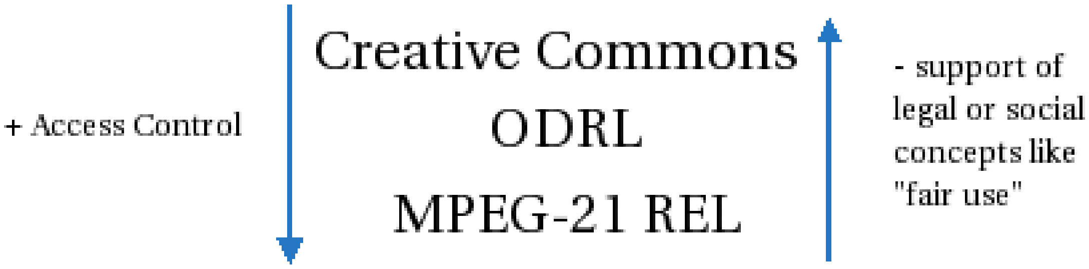


Figure 8.1: Comparing RELs functionalities [90]

In the next section, we present a formal definition for a *Li*censing *R*ights *E*xpression *L*angauge or LiREL. This language is structured in the form of a licensing contract but is also able to express the access control policies required for DRM systems.

8.4 A Formal Description for LiREL

In this section we define a formal description for LiREL, using set notation. The notation is comparable to the notation used by Pucella and Weissmann in [157] and is also similar to the notation used by Ferraiolo et al. in [84] and in other access control specifications. A formal description provides specific semantics around access control, a necessary step for creating interoperable, unambiguous implementations. We also draw upon some of the requirements defined in the draft versions of ODRL v2 specified in [104, 105] to specify some of the requirements for individual elements.

As already discussed, a DRM use license is essentially a contract between two parties: the licensor and the licensee (the consumer). A contract **C**, between two parties, a and b, any third parties involved (π) and the agreement α can expressed with the following tuple as:

$$\mathbf{C} = (a, b, \pi, \alpha)$$

For DRM use licenses, the contract needs to include details of the resources addressed in the license (r), the constraints for the contract (κ), and should be signed by representatives of the licensorsλ. Thus a DRM use license **L** can be expressed as:

$$\mathbf{L} = (C, r, \kappa)\|DSig_{\lambda}$$

This can be expanded, and (8.1) defines a DRM use license as:

$$\mathbf{L} = (a, b, \pi, r, \alpha, \kappa)\|DSig_{\lambda} \tag{8.1}$$

Note that there can be more than one signator for a license, and the consumer can also be a signator on a license. Also, it is not always necessary to have a signed contract, although it is considered to be good practice.

8.4.1 Language Semantics and Syntax vs Language Vocabulary

In this section we define the syntax and the semantics for a REL. In terms of natural languages, this is comparable to the definition of what is a noun, a verb or a pronoun etc., and how these can be combined to form a sentence. The vocabulary is the second component of the language, which allows for the expression of these terms. We do not focus on the vocabulary of LiREL; and while a standardised vocabulary is essential for interoperability, it will be difficult for us to define a comprehensive vocabulary for LiREL. For this reason, we focus solely on the syntax and semantics of LiREL and allow for the definition of the associated vocabulary separately. This approach is also used by ODRL and XrML although some comparisons of RELs tend to discuss the vocabulary of the languages [90, 175].

8.4.2 Obligations

Almost every term in a license can have obligations attached to it, and hence we begin our discussion with obligations, also referred to as duties by Guth [93]. As discussed earlier, some authors have felt that obligations add too much complexity to access control specifications [174], but obligations are a fundamental part of contracts, and thus necessary for DRM use licenses. Some obligations, like payment terms, can be enforced at machine level, but others are purely legal in nature; and disputes that arise will need to be resolved in arbitration or court. Obligations can also have constraints attached to define limitations to obligations.

An obligation definition in a use license should contain the necessary details of the obligation, and should be able to specify whether the obligation is negotiable.

8.4.3 **Constraints**

Constraints define restrictions for specific terms in a license (such as the number of times an action is allowed), or apply to the license on the whole (such as the period of validity of the license). License constraints apply to every member of both a and b, but not necessarily to any delegated parties. Constraints can also be applied to obligations. As discussed in [105], constraints can be used to express mathematical terms, such as:

> number of pages is less than 100

where "number of pages" and "100" are operands and "less than" is the operator.

In addition to constraints, ODRL 1 also used a *condition* model [103]. Conditions allowed a license or permission to be invalidated once they became activated. Thus, it is possible to rewrite a condition as a constraint, and conditions were subsequently removed from the draft specifications of ODRL 2 [104].

All constraints applicable to the license, obligation or permission must be met before access can be granted. Constraints themselves can have further organisation to support different groupings. In such a case, the semantics of the constraint will depend on the definition of the constraint. Consider the following license constraints:

1. Valid Until: 2007-12-31
2. Device restriction:
   (a) device id: 8755GHGT876
   (b) device id: 867453HGT97

The licensee can only be granted access to the resource if the date of access is before 31 December, 2007 and the device that is used to access the resource has one of the listed device identities.

A constraint definition in a use license should be able to specify whether it is negotiable, and should be capable of expressing mathematical terms involving two operands and one operator.

8.4.4 **The Licensors**

As discussed in section 7.1, a is the set of persons (natural or legal) who have been authorised by the rights holders of the resource, to license the access to resources r to prospective licensees.

a does not need to be a comprehensive list, but must satisfy the following:

$$a = \{k_1o_1, k_2o_2...k_no_n\}, \quad a \neq \emptyset, n > 0 \tag{8.2}$$

$$k \in a \Rightarrow \exists l \in r, k \in \text{ authorised licensors of } (l) \tag{8.3}$$

$$\forall l \in r, \exists k \in a, k \in \text{authorised licensors of } (l) \tag{8.4}$$

We have provided a definition for *a*, in (8.2), every licensor (*k*) has an associated set of obligations o, with (8.3) providing a definition of a licensor while (8.4) also describes the relationship between the licensor and the resources. This definition of a licensor allows for a license to reference licensors who do not operate licenses on works referenced in the license, thus catering for licensing of compilations of works.

a should be interpreted as a list of licensors, and there is no relationship necessary between licensors. The rights holders should be referenced through a globally unique identity scheme. Many such schemes exist and we discuss this further in chapter 10. It should be noted that obligations for the rights holders could be purely legal in nature (for example, 24-hour telephonic support), and not enforceable on a computer system. However, this should be seen as a strength since it gives the use license a sounder legal grounding.

A licensor definition in a use license must specify the licensor's identifier.

8.4.5 **The Licensees**

As discussed in section 7.1, *b* is the set of persons (natural or legal) or roles representing the consumers of the resources defined in *r*, defined in (8.5), while we define what is meant by a user in (8.6). Note, that unlike *a*, *b* can be defined as an empty set, and thus accommodate anonymous users. *b* should primarily be interpreted as a list of users who are given the permissions defined by α.

$$b = \{k_1o_1, k_2o_2...k_no_n\} \quad n \geq 0 \tag{8.5}$$

$$k \in b \Rightarrow \forall l \in \ r, \text{k gets access to } l, \text{under conditions } \alpha \tag{8.6}$$

However unlike licensors, it should be possible to create relationships between the users (and roles). For example, it should be possible to define a user list as:

$$b = \{(Alice \wedge Journalist), (Bob), (Eve \wedge Teacher \wedge Mrs\ Smith)\}$$

Thus, unlike for *a*, the definition for *k* in *b* is no longer just a single identity, but rather a group of identities or roles as defined in (8.7). To meet (8.6), all the roles and identities comprising k_i must be satisfied for k_i to gain access to a resource. Individual identities/roles may also have obligations, as well as obligations that affect k_i as a whole.

$$k_i o_i = \{j_1 o_1 \wedge j_2 o_2 \wedge ... \wedge j_t o_t\} \cdot \ o_i \tag{8.7}$$

$$t > 0$$
$$0 \leq i \leq n$$

where *j* is a person, role or group associated with a identity scheme.

Like the licensors, licensees should be referenced through a globally unique identity scheme, preferably the same scheme used to reference licensors. Roles can be catered for by a credentials service, and although a global credentials service does not currently exist, setting up such a service should not be too difficult. Some identity systems, like Kerberos, already provide a credential service. We discuss these issues in more detail in chapter 10.

A licensee definition in a use license must specify the licensee's identifier and must be able to indicate whether the inclusion of the licensee in the use license is negotiable.

8.4.6 The Third Parties

π is the set of persons (natural or legal) who have been appointed as third parties by both *a* and *b* as part of the agreement α.

$$\pi = \{k_1 o_1, k_2 o_2 ... k_n o_n\} \quad n \geq 0 \tag{8.8}$$

Similar to licensees, π there could relationships between users and roles appointed as third parties. For example, third parties for an electronic contract could be:

$$\pi = \{(Verisign \wedge CertificateAuthority), (Thawte \wedge CertificateAuthority), (John \wedge Judge)\}$$

Thus, as for *b*, the definition for *k* in π is no longer just a single identity, but rather a group of identities or roles as defined in (8.9). The entire third party set comprising of k_i must satisfy their obligations as part of the contract. Individual identities/roles for the third parties may also have obligations, as well as obligations that affect k_i in as a whole.

$$k_i o_i = \{j_1 o_1 \wedge j_2 o_2 \wedge ... \wedge j_t o_t\} \cdot \ o_i \tag{8.9}$$

$$t > 0$$
$$0 \leq i \leq n$$

where *j* is a person, role or group associated with a identity scheme.

Like the licensors and licensees, third parties should be referenced through a globally unique identity scheme, preferably the same scheme used to reference licensors and licensees. We discuss these issues in more detail in chapter 10.

A third party definition in a use license must specify the third party's identifier and must be able to indicate whether the inclusion of the third party in the use license is negotiable.

8.4.7 The Resources

r is the set of resources, which the licensees *b* are given access to under conditions α by the licensors *a*. Licensees are given access to all the resources identified in *r*. Like the licensors, licensees and third parties, resources need a globally unique identity scheme. Furthermore, the DRM system needs to be able to perform identity verification for resources; i.e. *establish the truth of a claimed identity* [180]. However, unlike identity systems for users, most identity systems for digital resources do not provide a verification service. We discuss such a service also in chapter 10.

A resource definition in a use license must specify the resource's identifier and must be able to indicate whether the inclusion of the resource in the use license is negotiable.

8.4.8 The Agreement

The agreement α is the most important part of the use license, and defines the access control rules and policies. It is also the most complicated part of the model. α is composed of permissions (ρ), its constraints (κ), and obligations (o) associated with each of the permissions. Thus, the agreement can be defined as:

$$\alpha = \{\rho_1 \kappa_1 o_1, \rho_2 \kappa_2 o_2 ... \rho_n \kappa_n o_n\} \quad n > 0 \tag{8.10}$$

κ_i is a set of constraints applicable to the individual permission ρ_i. The interpretation and definition of κ_i will thus depend on ρ_i. Each permission ρ_i is part of a pre-defined permission set *PS*. The definition and interpretation of *PS* will differ according to the application of DRM, and is dependent ultimately on the implementation of the DRM controller (the system that interprets and implements the use license) and the rights holders, where the rights holders can choose a *PS* that is a subset of the entire set available from the DRM system. For example, in the traditional Unix file system:

$$PS = \{read,\ write,\ execute\}$$

In the Unix file system, a permission called "print" has no meaning, and thus cannot be enforced, even if it is expressed as part of the use license. The rights holders can choose to reduce *PS* to:

$$\acute{P}S = \{read, write\}$$

Thus, for a use license created with reference to $\acute{P}S$, *b* will always have the right to execute, even though the DRM controller can technically control that right. It can be argued that rights not defined in a PS should be unregulated instead of being allowed. The problem with this position however, is that DRM controllers can then choose to block rights that are unregulated (e.g. do not allow execute, even though execute $\notin$ $\acute{P}S$). Removing blocks on unregulated permissions could become difficult for the consumer, and thus our position is to allow any rights not defined in a PS.

Using these examples, we can motivate a generalised set of conditions for enforcement of rights in (8.11). There is a difference in this model to the conventional view of DRM use license interpretation, where only permissions granted explicitly in the use license should be enforced (closed policy). In the third case of (8.11), the license should be considered invalid.

$$\begin{array}{llcl} \rho \in \alpha & \rho \in PS & \Rightarrow & \rho \text{ granted} \\ \rho \notin \alpha & \rho \in PS & \Rightarrow & \rho \text{ denied} \\ \rho \in \alpha & \rho \notin PS & \Rightarrow & \rho \text{ granted} \\ \rho \notin \alpha & \rho \notin PS & \Rightarrow & \rho \text{ granted} \end{array} \tag{8.11}$$

A More Complex Agreement

In the current definition of agreement (8.10), authorised users are given all the permissions present in α. However, there can be use cases where a more complex agreement is required. For example, it may be desirable to create an agreement for a PDF document as follows (ε

represents the empty set of constraints/obligations):

$$\alpha = \{(view \cdot \varepsilon \cdot \varepsilon) \vee (view \;\cdot AdobePDFReader7 \cdot \varepsilon \wedge print \cdot 2 \cdot \varepsilon)\}$$

This agreement can be interpreted as, the user has the right to view; or the user can view in the application AdobePDFReader7, and also get the right to print the document twice. There are many other use cases where a more complex definition for agreement may be necessary as detailed in (8.12), where $\odot$ represents the relationship between $\rho_i \kappa_i o_i$ and $\rho_{i+1} \kappa_{i+1} o_{i+1}$.

$$\alpha = \{\rho_1 \kappa_1 o_1 \;\odot\; \rho_2 \kappa_2 o_2 \;\odot\; ... \;\odot\; \rho_n \kappa_n o_n\} \tag{8.12}$$

However, using the distributive laws, it is possible to simplify (8.12) to

$$\alpha = \{\varrho_1 \vee \varrho_2 \vee ... \vee \varrho_m\} \tag{8.13}$$

where

$$\begin{aligned} \varrho_i = \{\rho_1 \kappa_1 o_1 \wedge \rho_2 \kappa_2 o_2 \wedge ... \wedge \rho_t \kappa_t o_t\} \\ \varrho \neq \emptyset \\ 1 \leq i \leq m \\ m > 0 \\ t > 0 \end{aligned}$$

Thus, an agreement can be defined as a set of non-empty permission groups, ϱ, with each permission group consisting of a non-empty set of permissions and their associated constraints and obligations.

The "NOT" Permission

The "not" permission expression was initially envisaged as a means to create easier agreements where the majority of the permissions (in a permission set) are allowed [104], and would thus allow open policies in DRM. This is however simply an easier expression mechanism at the tool level, and an unnecessary feature for the model itself. Furthermore, using the definition of the permission set and (8.11), the use of a "NOT" permission is almost meaningless. For this reason we do not have any specific support for a "not" function.

Thus, our model cannot be categorised in either of the traditional definitions of open and closed policies. Instead, through the use of permission sets, we implement closed policies on only a defined set of operations.

In DRM systems, the use of global closed policies can be disadvantageous to the consumer; as it leaves the possibility for the enforcement engines to enforce restrictions not specified as part of the rights package, or in an extreme case, allows the producers to update the rights enforcement engines to enforce restrictions not specified as part of the rights package at a future date. The use of a permission set protects the consumer from such abuse, in the present and the future; and the consumer is guaranteed that enforcement policies will not change without a change in the policy itself.

Delegation

Delegation is effectively a complicated version of ρ, but one which should probably be considered as a standard part of the LiREL model, instead of being part of a permission set. Delegation (δ) is really a modified version of **L**, and can be defined (where *c* is the delegated party) as:

$$\delta = (\acute{b}, c, \acute{\pi}, \acute{r}, \acute{\alpha}, \acute{\kappa}) \;\; where\, \acute{r} \;\subseteq\; r, \;\; \acute{b} \subseteq\; b, \;\; c \not\subseteq\; b, \;\; c \not\subseteq\; a \tag{8.14}$$

Note that $\acute{\pi}$, $\acute{\alpha}$ and $\acute{\kappa}$ are not necessarily subsets of the current π, α and κ. This definition also means that it is possible to create infinitely long chain of delegation, as delegation can continue to be part of $\acute{\alpha}$.

Use License Specifications

A permission definition in a use license must specify the details of the permission and must be able to indicate whether the permission of the resource in the use license is negotiable.

8.4.9 **Catering for Negotiations**

As we discussed earlier, negotiations are the means to establishing a contract, but at the same time, as discussed by Jamkhedkar et al. there is a need to minimise the complexity of RELs [108]. Thus, it would be best, if it is possible to cater for negotiations without an increase in the terms of the LiREL model.

Wooldridge and Parsons discussed a logic-based language for bargaining-based negotiation in [197], which we have reproduced in table 8.1. In their specification, the final result of a successful negotiation is *accept(i,j,φ)*; which is very similar to our own specification of **L**.

Using the illocutions defined in table 8.1, and the definition of the final agreement in (8.1), we can define negotiations for DRM in table 8.2, which is equivalent to Wooldridge and Parson's

Illocution	Meaning
request(i,j,φ)	a request from *i* to *j* for a proposal based on φ
offer(i,j,φ)	a proposal of φ from *i* to *j*
accept(i,j,φ)	*i* accepts a proposal φ made by agent *j*
reject(i,j,φ)	*i* rejects a proposal φ made by agent *j*
withdraw(i,j)	i withdraws from negotiation with j

Table 8.1: Illocutions for a logic-based negotiation language as discussed by Wooldridge and Parsons

definitions with $\varphi = (\pi, r, \alpha, \kappa)$.

Illocution	Meaning
request(i,j,π, r, α, κ)	a request from *i* to *j* for a proposal based on α
offer(i,j,π,r, α, κ)	a proposal of α from *i* to *j*
accept(i,j,π,r, α, κ)	*i* accepts a proposal α made by *j*
reject(i,j,π,r, α, κ)	*i* rejects a proposal α made by *j*
withdraw(i,j)	i withdraws from negotiation with j
agreement(i,j,π,r, α, κ)	i concludes an agreement with j

Table 8.2: Illocutions for negotiation in DRM

From table 8.2, it is clear that, except for withdrawal, there is no real difference between each of the illocutions. In fact, each of these illocutions can be seen as a license state. Thus, we could cater for negotiations within LiREL without a substantial increase in the computational complexity of LiREL. For example, an offer could look like:

```
<license type="offer">

      Details of the offer

</license>
```

while an agreement would look like:

```
<license type="agreement">

      Details of the agreement

</license>
```

Other negotiation mechanisms such as auctions and bidding discussed in chapter 7 can also be accommodated by increasing the vocabulary of the license states. Thus the complete set, as we have discussed in chapter 7 would be:

- Agreement (the complete agreement, and the default state)
- Request
- Offer (can be used for both bargaining and bidding)
- Counter-Offer
- Accept
- Reject
- Tender

As discussed in the course of the individual license elements, each element other than the licensors can indicate whether they are negotiable or not. It does not make sense to allow licensees to negotiate the inclusion of licensors as part of the agreement.

8.4.10 **Visual Model**

From the definition of the model in this section, we can create a UML model of **L** as shown in figure 8.2. We have expressed *b* as a "licensee group" and π as a "third party group" to enable the more complex expressions which we described in section 8.4.5. Note that, this model could be compressed using abstract classes or through creating an inheritance model for common classes and elements. In our XML Schema descriptions (see Appendix B) we have used these techniques.

8.4.11 **Comparison to Current RELs**

There are a number of REL specifications available currently, but well known, standardised specifications of MPEG-REL (based on XrML) and OMA-REL (based on ODRL 1) are hardly used [108].

Even though RELs can be seen as an expression of access control [70], neither ODRL 1 nor XrML have formalised models for their specifications. Mulligan and Burstein did create a simple model for XrML in [142], as shown in figure 8.3, but it can not be considered a comprehensive model. However, it can be clearly seen from the model, that many of the features presented in our model, such as catering for duties and delegations, are missing from XrML.

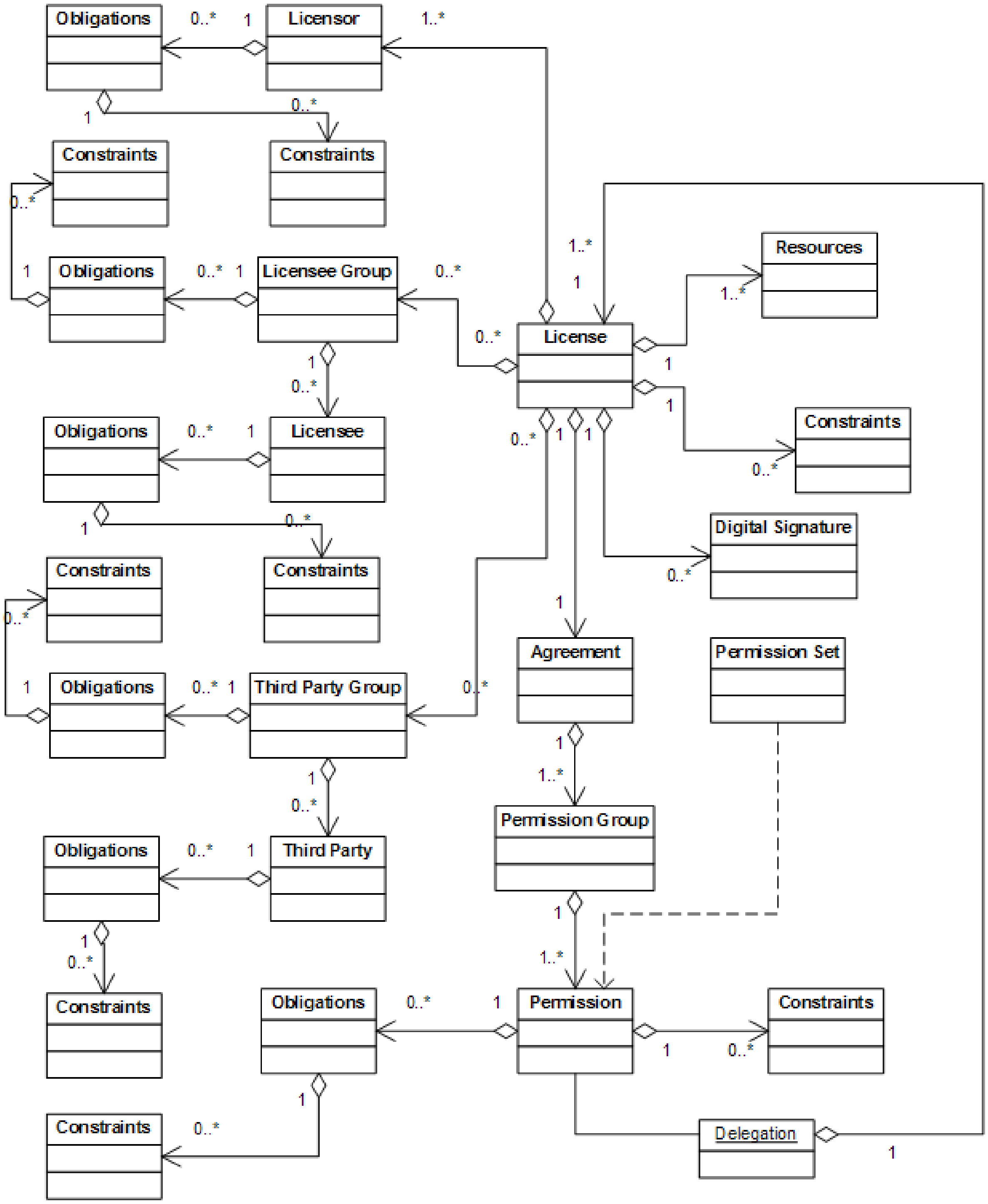


Figure 8.2: UML model of **L**

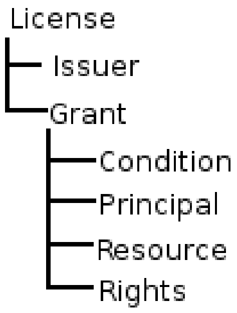


Figure 8.3: Simple XrML Model [142]

ODRL 1 (base language for OMA-REL) does not have a formal model either, although Guth did map various components of ODRL 1 to the Contract Schema (CoSa) she developed in [93]. This was recognised, and the requirements specification for ODRL 2, does state the need for a formal specification for the language [104]. The latest draft of the ODRL 2 model specification does have a UML model, which is shown in figure 8.4. This model is very similar to our model shown in figure 8.2, although the ODRL model does provide a lot more detail.

There are a few differences between the ODRL model and the model we have presented in this paper:

1. **Prohibition and Permission:** ODRL has an explicit support for the "NOT" permission, although there could be confusion in the case where permission and prohibition are present in the license at the same time. As we discussed earlier, we do not believe this approach makes sense, and thus have not implemented such a mechanism in our model.

2. **The Legal Element:** The legal element provides some of the functions that we have generalised as "contract constraints"; but more specifically geared towards providing a firmer legal basis for DRM use licenses.

3. **Separation of Parties:** In our model, we have separated the parties into their respective functions with respect to concluding licenses, while the ODRL model makes no such distinctions. One advantage of our approach is to differentiate the interpretation of the relationship between the parties and the resources, which is not possible with the approach adopted for ODRL v2.

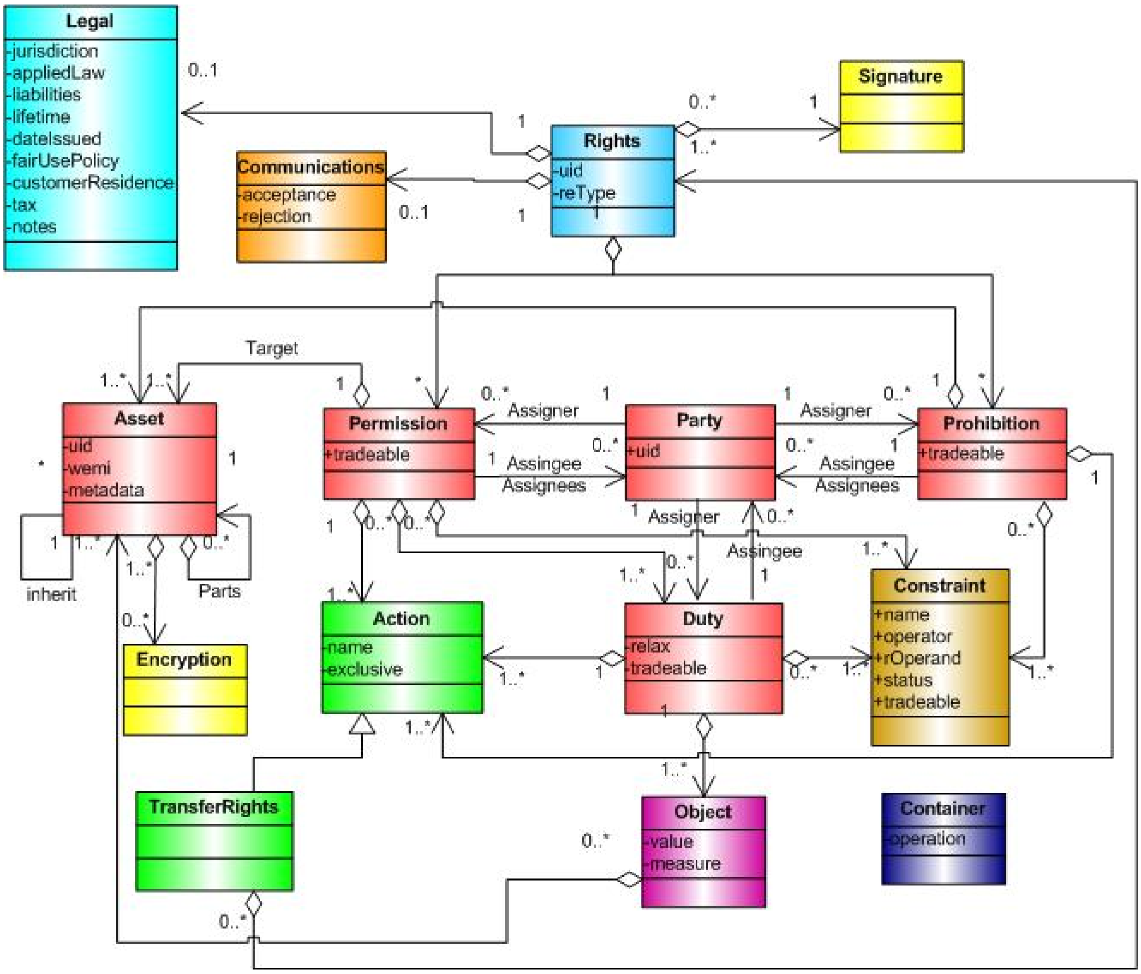


Figure 8.4: ODRL 2 Model (Draft) [105]

8.4.12 XML Schema

We have used the LiREL model presented in this section to create an XML based REL. LiREL is very similar to the draft ODRL v2 schema, due to the similarities in the model. The complete schema is listed in appendix B.

We have also created a sample permission set for LiREL, which is listed in appendix C. We have used LiREL, and the sample permission set to demonstrate a complete run of a bargaining protocol, which also serves as an example of use licenses created with LiREL and permission sets for LiREL. This example is listed in appendix D.

8.5 Analysis of Access Control Enforcement

With a formal description for the use license, it is now possible to examine the second part of access control as defined by Jajodia et al. in [106]: the means to implement the policies correctly and effectively. In this section, we are going to use the formal model of LiREL to examine the issues related to the enforcement of the use license.

8.5.1 **Validity of Use Licenses**

Before a license can be enforced, the license needs to be valid. A license can be invalidated for three reasons:

1. It is malformed.
2. The constraints for the license cannot be met
3. The license is legally invalid.

Malformed License

A license is malformed if the license does not meet the specifications detailed in the model. For example, if the license does not state any licensors, or there are other syntax errors. A license can also be malformed if there are permissions granted by the agreement that are not present in the permission set associated with the agreement. This is the third case described in (8.11).

Constraints cannot be met

There are constraints that apply to the entire license, that can no longer be satisfied. For example, the time limit placed on the license may have expired, or the license may have stated that it could only be used a certain number of times. Once these constraints can no longer be satisfied, the license becomes invalid.

Legally Invalid

There are two main reasons why a use license can become invalid legally. First, the license could have been drafted by illegal means – for example, the licensor was not authorised to provide licenses. The second reason could be due to license revocation. Licenses can be revoked for a number of reasons, but there are two main reasons: the licensee can acquire a new license under different terms or one of the parties of the license could have broken the terms of the license.

8.5.2 **Enforceability of Use Licenses**

Even if a license is valid, it does not guarantee that the license is enforceable at the target device. A license is only enforceable, if the permission set associated with that license is enforceable by the device. The consumer cannot be granted access if a license is not enforceable.

It could be possible to allow selective enforcement of use licenses. In section 9.1.2, we introduce the concept of rights levels, where we discuss the fact that some permissions can only be enforced at certain levels of the system (Jamkhedkar and Heileman referred to this as upper and lower level enforcement [107]). In such a case, it would also make sense to separate enforceability into two parts. But such a use license should ideally make use of two distinct permission sets, one for each of the levels of enforcement.

8.5.3 Conflict Resolution

In [106], Jajodia et al. defined conflict resolution as the process undertaken when there are conflicting authorisations for the same subject. Jajodia et al. discussed three approaches to conflict resolution:

1. **No Conflict:** A conflict state indicates an error in the access control system.
2. **Denials take precedence:** A negative authorisation takes precedence over a positive authorisation.
3. **Permissions take precedence:** A positive authorisation takes precedence over a negative authorisation.

We propose the use of permission precedence for DRM systems. As long as the user can present a valid and enforceable use license for a particular action over a resource, the action should be allowed. Jajodia et al. also discussed the conflicts arising from different delegated authorisations. Permission precedence addresses this issue, and the other factors discussed by Jajodia et al. thus do not apply.

8.5.4 Deciding a Request

Decision on whether a user is granted an action on a resource is undertaken by the enforcement agent, or the DRM controller. As already discussed, given a request for permitting an action, the DRM controller will try to allow the action, from its known set of valid, enforceable licenses. More formally, the DRM controller will grant an action *req* to a consumer *k*, if

$$\exists \text{ a license l, such that } req \in \alpha,\ k \in b, \tag{8.15}$$

and the following conditions are satisfied:

1. The obligations placed on the consumer are satisfied (assuming that the DRM controller can evaluate such obligations).

2. Any obligations attached to *req* are satisfied (assuming that the DRM controller can evaluate such obligations).

3. The constraints (if any) placed on *req* are met.

Note, from (8.11), *req* only needs to be evaluated if $req \in \acute{P}S$, where $\acute{P}S$ is the permission set enforceable by the DRM controller. If $req \in \acute{P}S$, then *req* needs to be evaluated against the permission set of the use license, as defined in (8.11). We discuss this further in chapter 9.

8.5.5 Determining Cardinality

One of the features found in many access control models, especially RBAC models such as [83], is determining cardinality of the roles in a deployed access control system. There are possibly two summaries that rights holders would be interested in:

1. The number of consumers who have a certain permission on a certain resource.

2. The number of resources (and their associated permissions) attached to a particular consumer.

While both summaries are possible to calculate, they are processing expensive operations (taking account of revocations etc). Furthermore, if the licensors make use of external identity management services, the calculation becomes more difficult, as the cardinalities for roles may be much larger than than the number of licenses issued. While this may be inconvenient for rights holders and licensors, this is a privacy boost for consumers.

8.6 Summary

There is no formal description of DRM systems, including the specification and interpretation of access control policies. We believe DRM is another form of access control, and there are a number of differences between DRM and other well known access control models.

In this chapter we presented a formal description of LiREL, a rights expression language that is able to express access control policies and contractual agreement in a single use license. Our formal description include:

1. The representation of the involved parties, individually or in groups. The three parties involved in a licensing agreement are the licensees, the licensors and third parties.

2. The representation of the resources covered by the license.

3. The details of the terms and conditions (the agreement) for access to the resources. Our model allows for the expression of multiple simultaneous conditions that need to be satisfied for access to be granted.

4. The representation of the constraints and obligations attached to individual parties and access terms. Constraints can also be attached to the entire license.

We also discussed the interpretation of LiREL, and the implications for the enforcement of DRM policies expressed in LiREL, including multiple conflicting licenses.

CHAPTER 9

AN AUTHORISATION FRAMEWORK FOR DRM

Access control can be seen as a two-part process: authentication of the parties and processes involved, followed by an authorisation that the parties and processes can take part in a transaction. In chapter 8, we discussed a formal model for DRM where we also discussed a formal representation of the access control rules and policies. In this chapter, we discuss an authorisation framework for DRM, while we discuss the authentication component of access control in chapter 10. We discuss authorisation first, as the authentication framework needs to supply input into this framework.

An authorisation is defined in RFC 2828 as "*a right or a permission that is granted to a system entity to access a system resource*". Thus, in simple terms, the authorisation component of the DRM system is the decision maker and decides whether to grant a user the requested permission to a protected resource.

The authorisation framework we present here comprises of two parts: identify and manage the elements that are required to make an authorisation decision and the logical process involved in making that decision.

9.1 THE AUTHORISATION PROCESS

As discussed by Rosenblatt in [168] and subsequently in [169], as well as in Erickson in [81] (and discussed earlier including in section 4.1); the DRM controller is the decision making component in the DRM system. Thus, the DRM controller is the central player in the authorisation process, and ultimately makes the decisions related to authorisation.

9.1.1 **The DRM Controller**

Classification of Different Types of Implementations

The DRM controller can be implemented at various levels in a computer system:

1. **Application Layer:** The DRM controller can be coupled with an application, and the application interprets and enforces the controls. As discussed in chapter 3, the majority of DRM systems currently use application layer DRM controllers.

2. **Operating System Layer:** The DRM controller can be incorporated as part of the operating system, and thus the enforcement could be transparent to the application. Microsoft's RMS is the only DRM system that has some OS level DRM controller implementation, although most of its functionality is still at the application layer.

3. **Hardware Layer:** The DRM controller can also be implemented as a hardware module, and possibly operate independent of the operating system. In [168], Rosenblatt argues that DRM controllers in the hardware layer should be the ultimate goal for DRM systems.

There can be multiple DRM controllers in one system, and there is no reason why different DRM controllers cannot co-operate to enforce different aspects of the access control policy. In fact, it is not possible for a DRM controller in only one of the above categories to implement the complete range of access control permissions in current RELs. For example, the concept of a document page makes no sense at the operating system or hardware layers, but is rather tied to the application interpreting the data format. Thus, DRM controllers at different layers are needed to enforce a wide range of access control policies.

Other Components of DRM Controllers

DRM Controllers may also need to keep track of the state of a use license; particularly licenses that constrain rights to a number of instances (for example, print a document 4 times). Two approaches could be used: licenses could be changed to reflect the change; or the DRM controller will need to keep track of the license's state. The first approach would require the DRM controller to re-sign the use license after every change to the use license, and portability could be allowed. The second approach does not guarantee portability of the license state; but does not require any modification of use licenses by the DRM controller. The latter approach is also easier to implement.

Another component of the DRM controller should be a revocation database, which tracks revoked use licenses as well as other data, such as revoked (untrusted) users, devices and could also be used to track untrusted license servers.

Finally, DRM controllers could also cache use licenses and other required information (such as user authentication) in local memory instead of acquiring the details from an external source every time. This is particularly useful for DRM controllers for specialised devices such as portable music players.

9.1.2 **Rights Levels**

Application level DRM controllers enforce a different set of rights compared to rights enforced by hardware and operating system level DRM controllers. Furthermore, because application level DRM controllers cannot control other applications in the operating system (such as applications monitoring data in memory), they offer a lower level of security compared to DRM controllers in the other levels.

For this reason, it is possible to categorise access control rights, as defined by existing RELs, into two levels. As discussed earlier in section 4.5, Jamkhedkar and Heileman also identified two layers for rights enforcement in DRM [107].

1. **Level 1:** These rights are **common to all operating systems and devices**. They include restrictions such as reading a file, and will in effect be persistent extensions of existing access control rules. These restrictions will be enforced at the operating system or hardware level. This was categorised as lower level enforcement by Jamkhedkar and Heileman.

2. **Level 2:** These rights are **only enforceable at the application layer** because only certain applications will be able to make sense of the rights. Rights like the permission to print a certain number of pages or the portions of a document that can be excerpted, only make sense to the application handling the file. This was categorised as upper level enforcement by Jamkhedkar and Heileman.

With the above categorisation, it will no longer be necessary to require application support if the DRM protection is restricted to Level 1 rights. It should be possible to specify in Level 1, the right to access a file only with a specific application thus preventing bypassing of Level 2 restrictions. In table 9.1, we categorise all the rights and permissions defined in the core ODRL 1.1 [103] and XrML [6] (the base for MPEG-REL) RELs. Some rights, such as Embed and Extract can be implemented at the operating system layer, but would require application support for maximum effect.

Level 1 Rights	**Level 2 Rights**
1. **Usage Rules:** Display, Execute, Play, Read 2. **Reuse Rules:** Aggregate, Edit, Embed, Excerpt, Extract, Modify, Write 3. **Asset Management:** Backup, Copy, Delete, Install, Export, Restore, Save, Uninstall, Verify	1. **Usage Rules:** Play, Print 2. **Reuse Rules:** Aggregate, Annotate, Edit, Extract, Excerpt, Embed, Modify, Write 3. **Asset Management:** Export, Save 4. **User Management:** Give, Lend, Lease, Load, Sell, Transfer

Table 9.1: Classification of Level 1 and Level 2 Rights

9.1.3 Multiple Use Licenses and Authorisation

Using the formal model we presented in chapter 8 it is important to note that authorisation is only granted according to the action the user wishes to perform with a protected object; i.e. the user cannot get a standalone list of actions that (s)he is authorised to perform. Furthermore, authorisation can only be granted if the DRM controller implements the *full* permission set associated with the use license.

Thus, recalling (8.15), a user's request *req* is authorised iff

$$\exists \text{ a license l, such that } req \in \alpha,\ k\ \in\ b,$$

and all the associated constraints and obligations specified in **l** are met.

Thus, if the user has a number of use licenses associated with the protected resource, the user requires only one valid license that authorises the user's request. Thus, the DRM controller will need to evaluate every valid use license presented by the user, until either authorisation is granted or the user presents no more use licenses to evaluate. In schemes where newer use licenses can invalidate a set of older use licenses (such as the scheme used by the DRM system associated with Blu-Ray discs [114]; the older licenses need to be listed in a revocation list, and the DRM controller must check a revocation list to ensure the validity of the use license presented by the user.

9.1.4 **The Process**

An overview of the decision flowchart is shown in figure 9.1.

Firstly, the DRM controller will get an input signal (from the application, the file system or the operating system depending on the implementation layer of the DRM controller), which would request permission to carry out an action on a particular resource. Depending on the implementation, the DRM controller first needs to decide whether the resource is a protected resource. If it is not, then the action is allowed by default and the DRM controller plays no further part.

If the resource is protected, then the DRM controller needs a license associated with the resource. The license can be embedded with the resource, or could be an external source. If there is no valid license[1], access to the resource is denied. If the license is unenforceable (because as discussed in chapter 8, the permission set is not implemented in the DRM controller), then access to the resource is denied.

If there is a valid license, the parties need to be authenticated against the license. As discussed in chapter 10, the parties that require authentication include the user, the rendering device and the resource itself (is the association between the resource and the license valid). The authentication requirements (if any) are included with the license. If any of the authentication requirements are not satisfied, access to the resource is denied.

If all the authentication requirements are met, the individual action requested by the user is evaluated against the use license. As discussed in chapter 8, the request can be evaluated against these criteria:

1. Does the request form part of the associated permission set, and is it granted by the use license?
2. Are the constraints and duties listed by the use license satisfied?

If the use license lists the request as part of the allowed actions, and the constraints and user duties are satisfied, then the request is granted. In many instances, it may be required to check the license state before determining whether a constraint is satisfied. If the use license does not

[1] A license can be invalidated in many ways, including the following factors: it has passed its lifetime, the license has been placed on a revocation list, the license is malformed, the license is unsigned.

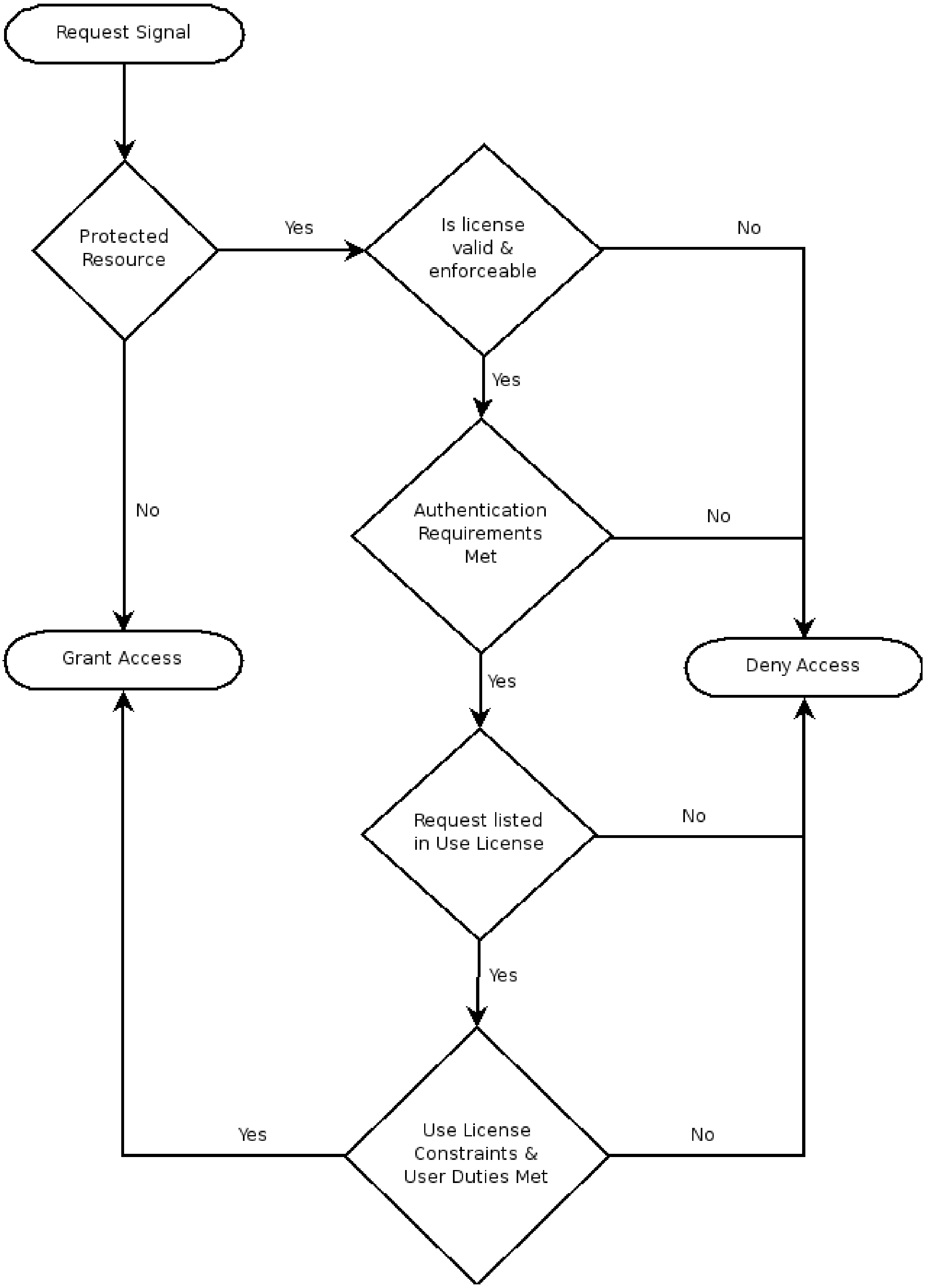

Request Signal
Protected Resource
Yes
Is license valid & enforceable
No
No
Yes
Authentication Requirements Met
No
Grant Access
Yes
Deny Access
Request listed in Use License
No
Yes
Yes
Use License Constraints & User Duties Met
No

Figure 9.1: DRM Controller Decision Flowchart

state the request as part of the allowed actions, or the constraints and user duties required by the use license for the associated request are not met; access to the resource is denied.

9.1.5 Verification of the correctness of DRM Controllers

For consumers, it is important to have assurance that the DRM controller is correct and enforces the use license correctly (and does not enforce usage rules that do not apply for example); and does not alter the platform's functionality and performance (requirement 27). For this reason, there is a need for a service that provides such an assurance to the consumer, preferably before the consumer makes use of the DRM controller.

Producers require a different verification service; they would like to be assured that the consumer is using the correct DRM controller, and while the use license can state such a requirement, a rogue DRM controller can easily bypass such a requirement from the use license. An assurance service could provide some level of certainty regarding the consumer's DRM controller, but since the assurance program cannot be expected to run in parallel to the DRM controller all the time, such an assurance can only be provided as a snapshot on a specific system, at a specific time.

9.2 Motivation for Decoupling Authentication and Authorisation

As we have already discussed in chapter 3, there are currently no DRM systems that provide comprehensive authentication services. Most systems perform authentication on one entity (usually the device) and, with the exception of Authentica's ARM, these authentication systems are strongly coupled with the underlying DRM systems, a point which was also noted by Michiels et al. in [132]. The main problem with this scenario is the lack of interoperability imposed by the strong coupling, as competing systems create their own authentication systems, and cannot cater for different authentication systems. In many instances the authentication systems themselves are proprietary, and thus even if new legislation forces systems to share data formats [18], they do not have any net effect as the rest of the system is not interoperable.

Strongly coupled systems which offer online authentication also mean that the rights holder has a complete knowledge of not only who has access to the protected data, but also when they are using the protected data. While, monitoring and tracking is a requirement for DRM systems (see our discussion in chapter 2), there needs to be a balance between monitoring and privacy of the users, especially in media DRM systems. Media DRM systems that communicate with rights holders without the knowledge of the consumer were exposed in [143], and the coupling of online authentication and the DRM controller makes it very easy to monitor consumer activity.

The decoupling of authentication from the control of the rights holders would therefore reduce the chance of correlation when users access protected works. This is further enhanced if the authentication system is used for other activities such as instant messenger or email access. However, full correlation between exact time of access and the user is still possible if the data is available. Thus monitoring and tracking will still be possible, if the authentication mechanism is controlled by the producer of the DRM content (as it is most likely to be the case in enterprise DRM systems).

9.3 Management Console

To make a decision, the DRM controller requires a use license as well as the means to authenticate the parties involved. Ideally, the operation of DRM controllers will be transparent to the users, with minimal usage interference for the user. However, if the DRM controller is implemented at the operating system or hardware layer, then there will be difficulty in implementing the interfaces required to handle the acquisition of use licenses and the authentication of the parties involved. Furthermore, there are devices that do not have the necessary Internet con-

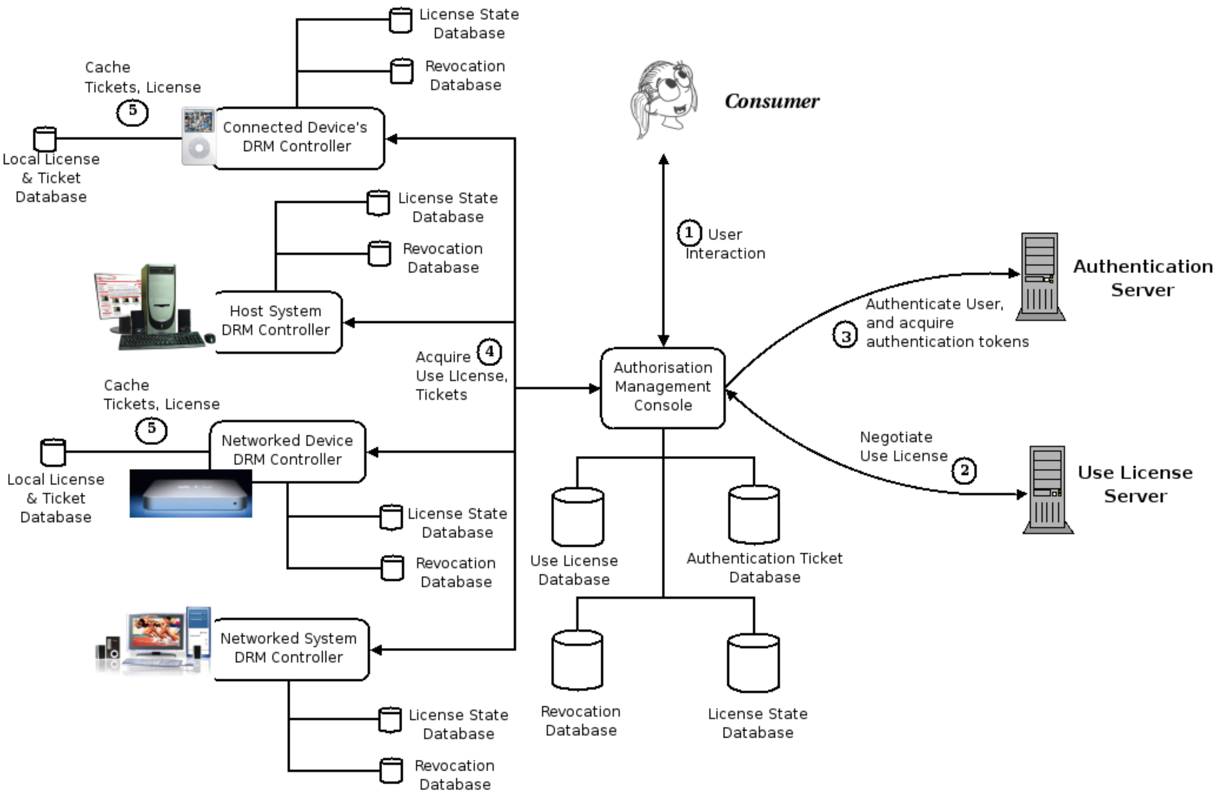


Figure 9.2: Management Console interfacing with a number of different DRM Controllers

nectivity to acquire use licenses or authenticate users. For this reason, we propose the use of a management console to manage use licenses and the authentication of the parties involved.

The management console is a user-level application that is responsible for the acquisition of licenses, management of licenses (such as viewing license details, removing invalid licenses), acquisition of authentication tickets (see chapter 10 for more details on authentication), and communicating these elements to the DRM controller. The console itself does not make any decisions, and thus, as long as there are open specifications on interfacing with the DRM controller, anyone can write their own management consoles.

The management console can also interface with other devices that connect to the host system, or even serve as a central console for a networked system. Home networked systems for example, can use one system to connect to the Internet and thus acquire use licenses and authentication tickets. These artefacts can then be distributed to the DRM controllers when required. An overview of how the management console can interface with different DRM controllers is shown in figure 9.2[2].

The management console can also be very useful for integrating different DRM controllers in the same device. For example, consider the following use license fragment:

```
<permissions>
    <read>
        <constraint application = “adobe.exe”, version = “7.0”/>
    </read>
    <print/>
</permissions>
```

The *read* permission can be best enforced by a DRM controller at the operating system level, while the *print* permission can be best enforced at the application level. Thus, there is a need for two different DRM systems to use the same use license, which can be easily facilitated by the management console. Furthermore, it would also not be needed for DRM controllers to provide interfaces for multitudes of different service providers including license servers and

[2] Device images were adapted from the following sources:
Computer, Host System: http://computerlink.com.pa/images/computer-linkplatinium.jpg
Computer, Networked System:
http://www.germes-online.com/direct/dbimage/50282793/Computer_for_Business_Use.jpg
Apple TV: http://images.apple.com/appletv/images/index123_20070109.jpg
Apple iPod: http://images.apple.com/ipod/images/indextwirl20060912.png

user authentication services. Instead, they can rely on a single interface, possibly with a standard interface protocol with a management console.

The management console has its own revocation and license state databases. These databases are there to provide information to the user, and can be synchronised from the databases hosted by the DRM controller. They can also be used to remove revoked or useless licenses[3] from the management console's database. Because the management console could be considered to be untrusted, it should not be used as either the source of revocation and license state information by the DRM controller or to decide the validity of use licenses and authentication tokens.

9.4 Modelling The Authorisation Process

To illustrate the robustness of our approach, we have modelled the authentication process using colour petri nets, and the graphical petri net is shown in figure 9.3. We have used different coloured tokens to define the various stages of the authorisation process (for example, 10 represents the authentication ticket, as presented by the consumer, while 11 represents an invalid authentication ticket and 12 represents a valid authentication ticket). In this way, it is easy to see what happens during a failure at a specific stage of the process. As with our other petri nets, this net was analysed using CPN-Tools. The petri net was modelled using a single license and single authentication token.

Reachability and Liveness

The petri net is reachable and live. Thus, every state in the authorisation model is possible, and there are no states that create a deadlock within the system. Thus, if correctly implemented, the system is guaranteed to process the request and not hang on the user.

Boundedness and Safety

The petri net is bound, and thus there is an efficient consumption of inputs to the system. Our net also shows that, should a license or authentication token not satisfy the requirements, the system will try to continue, with a replacement. This satisfies the model requirements we stated in (8.15).

The use of 1 token each for licenses and authentication tokens, ensures safety in the petri net. Increasing the tokens in the net yields an interesting result: there is a quicker recovery time in servicing the consumer if there is a failure during evaluation of either the authentication token

[3] A useless license is valid but cannot be used because there are no usable rights associated with it. For example a use license that provided the right to print once, and the user has exercised that right.

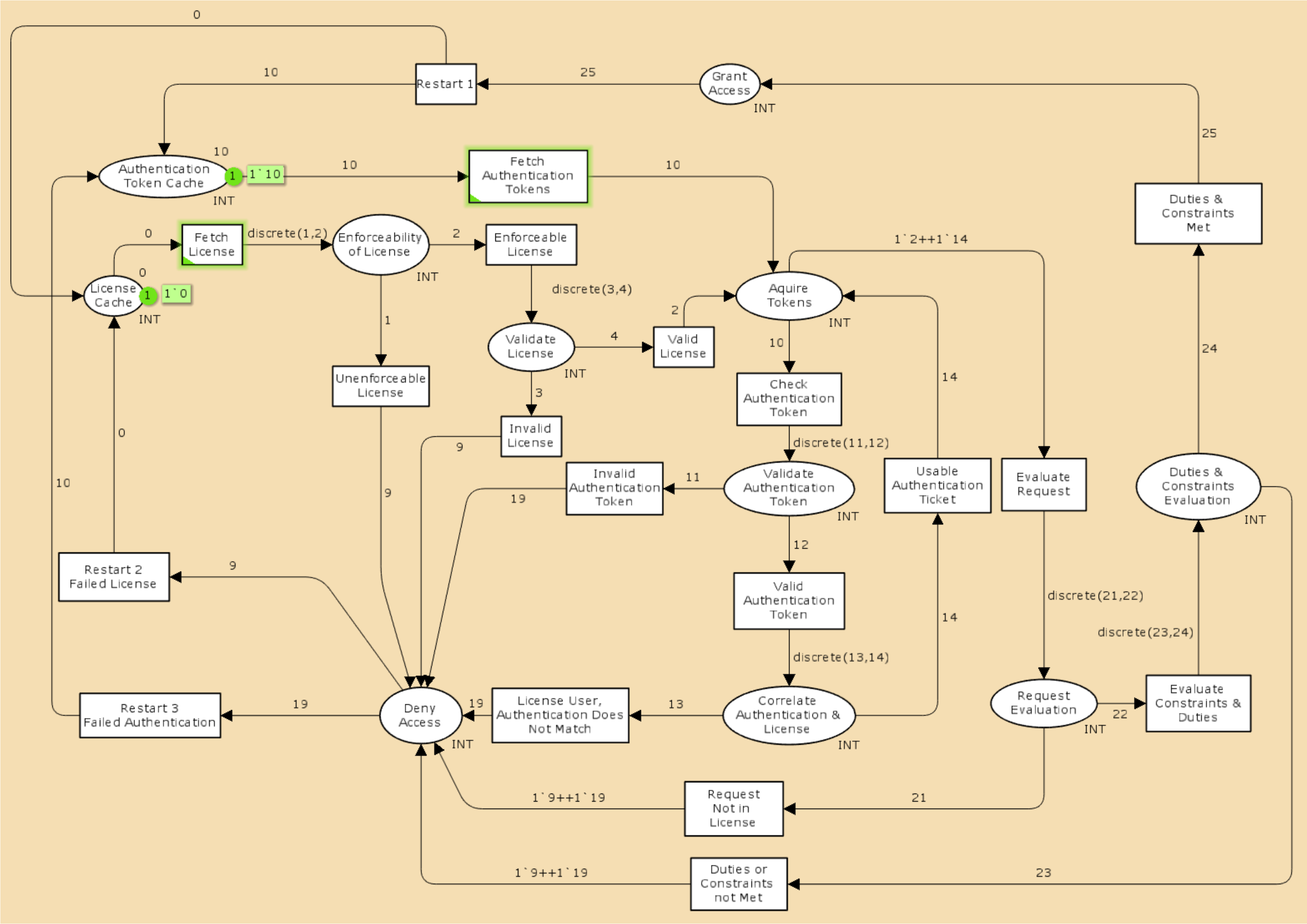


Figure 9.3: Petri Net of the Authorisation Process

or the use license. This further motivates the successful satisfaction of the requirement stated in (8.15).

9.5 Summary

In this chapter we have presented an authorisation framework for DRM, based on the formal model of DRM we presented in chapter 8. Authorisation is performed by the DRM controller, which can be placed at different levels in a computer system. DRM controllers places at the hardware and operating system level are more secure, but implement a smaller set of permissions when compared to application level DRM controller. In this chapter, we categorised the rights defined in the standard data dictionaries of ODRL 1 and XrML into level 1 and level 2 rights. Level 1 rights can be applied on any platform and are thus ideal for implementation at the operating system or hardware layer. In contrast, level 2 rights require application support.

Before authorisation can take place, users need to be authenticated. We motivated the need to separate administration and management of authentication tokens and use licenses from the core DRM controller itself. We also modelled our authorisation process using petri nets to demonstrate its robustness and verify that it meets the properties discussed in chapter 8.

CHAPTER 10

AN AUTHENTICATION FRAMEWORK FOR DRM

Access control can be seen as a two-part process: authentication of the parties and processes involved, followed by an authorisation that the parties and processes can take part in a transaction. In chapter 8, we detailed a formal description for DRM systems. In this chapter, we discuss an authentication framework for DRM, while we discuss the authorisation component of access control in chapter 9.

Authentication is defined in RFC 2828 as "*the process of verifying an identity claimed by or for a system entity*" [180]. RFC 2828 further elaborates the authentication process as a two part process:

1. Identification step: Presenting an identifier to the security system.
2. Verification step: Presenting or generating authentication information that corroborates the binding between the entity and the identifier.

As discussed in chapter 2, and elaborated in our model in chapter 8, DRM systems have three system entities that need to be authenticated: a system to identify users (licensors, licensees and third parties), a system to identify digital objects (including the use license) and a system to identify devices. One of the cornerstones of a DRM system is to ensure that the digital object and the use license correspond to each other; and should there be a mismatch, the consumer should not be allowed access to the digital object. Similarly, there are authentication steps matching the licensee and the use license; and the device and the use license.

In this chapter, we discuss both the identification and verification of these entities. Our discussions address the requirements 2 to 9 and requirement 11 and 12 discussed previously in chapter 2. We have previously published parts of the discussion we present in this chapter in [34, 35].

A further step in authentication, that is not included in the definition in RFC 2828, involves the *trustworthiness* of the parties that provide the core elements for identification and verification. We discuss a trust framework addressing these issues in chapter 12

10.1 RELATED WORK ON AUTHENTICATION

In this section, we discuss related work in ticket based user identity, resource identity and device identity systems. Unfortunately, the most common approaches to user management (an interactive username/password based system) is not completely ideal for DRM systems, and we discuss our observations before discussing ticket based identity systems.

10.1.1 Ticket based User Authentication

The problem with the naive approach to user authentication

Authentication in a DRM system usually occurs when the protected file is first accessed. In some designs, subsequent authentication steps take place when other rights are requested. User authentication requires the verification of the user's identity as prescribed in the protected work's use license. As we discussed in chapter 3, in systems such as Apple's iTunes, the user's identity is stored in a certificate and this certificate is matched to the work's license. In iTunes, this certificate is then locked onto the device thus prohibiting portability. In fact, the majority of DRM systems rely only on device authentication which contributes to a lack of interoperability. There are two key advantages afforded by device authentication:

1. The removal of user interaction during authentication.
2. Allow for offline authentication.

Both of these advantages are particularly useful in devices such as portable media players which are usually offline and have limited means of user interaction. But even in personal computers, the DRM systems that require users to provide regular authentication could alienate consumers and prove to be unusable. Similarly, despite the growth of Internet connectivity, it is still required to offer offline usage; and thus support offline authentication.

Thus, for DRM systems to successfully integrate user authentication and with it, other forms of profile based user authentication such as group membership, user authentication needs to offer the same advantages as device authentication. In this chapter we outline a credential/ticket based authentication service that overcomes the above problems; and we also outline mechanisms that could be used to integrate our approach to existing authentication systems.

Existing ticket based authentication systems

Ticket based or credential based authentication is best known through the Kerberos authentication system [183]. In Kerberos, users authenticate themselves to ticket granting servers which issue time limited tickets to be used for accessing Kerberos enabled services. This idea has been further improved and integrated with the latest generation of identity management systems including Liberty Alliance.

The Liberty Alliance Federated Identity framework allows for a single user identity to be used to access different services from a variety of different service providers. The protocol for cross-service authentication makes use of signed tickets from the main identity provider [92]. However, unlike the solution we outline later in the chapter, tickets in the Liberty Alliance framework provide only one-time authentication (i.e. they are not re-usable), and do not necessarily provide any control over how long access is granted to the service. We feel that these features are necessary for authentication in a DRM scenario.

The use of tickets to control access to resources was discussed in [115] by Kim et al. They used tickets as an authorisation mechanism in Globus-enabled grids. The scheme we present and its application to DRM systems share many similarities with the scheme discussed by Kim et al. However the scheme presented by Kim et al. does not make use of re-usable tickets, and focuses on providing fine-grained access control to grid resources.

In [120], Kuntze and Schmidt discuss a ticket system built on the Trusted Computing Group's (TCG) Trusted Platform Module (TPM). However, TPMs are hardware devices meant to be securely coupled with the device (currently PCs), to provide a trusted computer base. Thus, the ticket solution discussed by Kuntze and Schmidt is effectively a ticket based device authentication mechanism; and not suitable for user authentication.

10.1.2 Resource Authentication

In computer security, issues surrounding digital identity often revolve around user identity, in particular managing multiple user identities and various associated problems. A user's digital identity plays a very crucial role in determining which services and data can be accessed by that user, which makes it a corner-stone in the security building blocks of any system. However, what is often overlooked are the issues surrounding *data identity*, which also plays a crucial role as there is a need to ascertain that the user has the correct access permissions for a unit of digital data.

An identification system for a unit of digital data or digital object[1](e.g. a file) can be considered to be composed of two parts – a set of labels that can be used to uniquely reference each digital object and a mechanism to verify the identity (i.e. given a set of labels and a digital object, it should be possible to refute the claim that the labels correspond to the digital object, if untrue).

Currently, there are a number of different identification schemes for digital objects. The identifier format for many such schemes are based on the Universal Resource Identifier(URI), which defines a common standard for expressing identifier protocol and the label itself [180]. However, none of the schemes provide any verification support, and thus we consider them to be incomplete. In this chapter we will discuss a verifiable identification system for digital objects.

In [88], Gladney discussed the need for a strong identity system for digital libraries. Gladney argued that future users of digital libraries would need a strong assurance for the authenticity of the digital data, and discussed how a strong identity system for digital objects provides this assurance. However, the systems discussed by the author (most of which are discussed below) do not provide any verification service.

Current Digital Object Identity Systems

There are a number of different identity systems for digital objects, most of them independent of each other. However, most modern identity systems make use of the Universal Resource Identifier (URI) as the base for their identifier system. The URI system defines both the syntax for an identifier system and a grammar to interpret the identifiers [54] and enjoys almost universal support at application level. For this reason, our proposed identity system also bases its identifier format on the URI.

While URI provides a standard base for creating identifiers, identity systems for digital data also need to provide mechanisms for locating the digital data. Thus, using the identifier on a system (usually networked), the identity system locates the data through the use of resolution servers. Resolution of the identifier does not necessarily locate the data itself, but would usually locate metadata and a direct network address to the data. However resolution systems, such as the Universal Resource Locator (URL) used for web pages have a problem with persistence - digital data can be easily moved to different servers, web sites can be re-organised etc. - and thus do not provide an efficient mechanism as an identifier for digital objects. For this reason,

[1] We would like to define a digital object as “a stream of logical contiguous bits stored as a single unit, typically in a file system on disk or magnetic tape” (adapted definition from Wikipedia) but not consider structures used in programming languages (although some of the concepts discussed could apply).

most of the digital object identification systems have focused on allowing for the persistent identifiers – the location of the data can change without a change in the identifier.

Probably the most widely used persistent identity system for digital objects is the Digital Object Identifier (DOI) system [121], which is in turn based on the Handle system [150]. The Handle system is primarily an identifier resolution system similar to DNS. When it receives a query, it looks up the identifier in its database and finds an appropriate server. The Handle server then returns the server address to the requestor. Often, the resolution is not directly to the data itself, but to a website that has a direct link to the data. For example, in the ACM digital library, the DOI system resolves to the abstract page for a given entry (usually papers from various journals and conference proceedings). The Handle system also provides distributed administration and resolution mechanisms allowing for greater availability. Rosenblatt has previously advocated the use of DOI for DRM systems [167], but DOI lacks verification support (as discussed in more detail below), and thus we feel it is unsuitable for DRM systems.

Other digital object identification systems include the ARK Persistent Identifier Scheme, which is a persistent identification scheme [121] and the Extensible Resource Identifier (XRI), which is a broad identity scheme aimed at combining multiple resource identification schemes [9]. Although the ARK Specifications recognises that identification is an association between an object and a label, and verification is required [121], the scheme has no verification support.

Problems with non-verification

The problem with non-verification in current identifier mechanisms for digital objects is clearly demonstrated in the DOI Handbook's numbering system – the DOI handbook always has the identifier **doi://10.1000/182**. Thus edition 1 (released February 2001) has the same identifier as the fourth edition (released April 2004). While this does allow one identifier to identify the latest version, it also means that the identification for earlier versions of the document are effectively lost. Furthermore, the DOI (or any other similar system) does not have any mechanism to prove that a downloaded version of the document is the same as the document located through the resolution process. Thus, if the latest version is compromised (by a hacker, virus, disgruntled maintainer or even negligence), there is no mechanism for the user to know that the data is compromised.

The same problem exists in conventional web pages – there is no mechanism to inform the user that the page served by the server is in fact the intended page and not a defaced or outdated page. A verifiable identification system for digital objects would be able to overcome these problems.

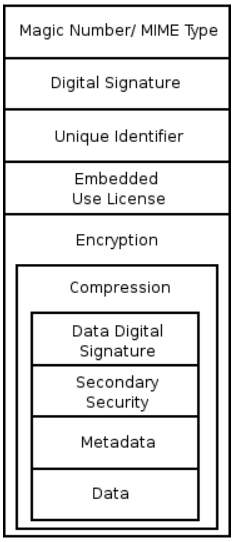


Figure 10.1: A layered view of DRM Packages

Problem with current approach to resource identity in current DRM systems

In figure 10.1 we show a DRM data package as a set of layers as we see it (as far as we know, this exact approach is not used by any current DRM system, but most systems would make use of something similar). A more detailed discussion can be found in chapter 11. Currently, to ensure integrity and thereby establish correspondence, DRM packages and use licenses are signed and a valid signature establishes the integrity of the respective digital objects. But, the digital signature only provides for the integrity of the package and does not actually provide any mechanism to verify that the identifier has any relation to the data, because the digital signature is for the integrity of the entire package.

But the identifier is related to the data and not to the package. In a flexible packaging system, it is necessary to allow for this type of flexibility where packages with the same data require different encryption, compression or secondary security features. However, a use license should not necessarily be restricted to a specific package but rather to the data contained in the package.

The above problem is not crucial in media DRM solutions, where there is typically only one DRM package for each digital object. However, in a broader context, for example enterprise DRM systems, there could be multiple versions of the data and simple user error (or identifier policy) could assign one identifier for multiple versions of the package. This could have serious security implications as it is possible that a consumer could have access rights to one version of the data and not other versions of the data - but the use license will not have any mechanism to correct this.

10.1.3 Device Authentication

As already discussed, device based authentication is the most popular form of authentication for current DRM systems. Currently, device based authentication can be applied by two different means; each with their own advantages and disadvantages.

1. **Cryptographic Based**. Systems like iTunes embed applications (or the underlying operating system) with a signed key (as part of a public/private key pair for example), which is then used as a means of authentication and for decryption purposes. The key is also backed up with other information, such as entries into the Microsoft Windows registry, or obfuscated in some manner, to prevent it from being extracted. However, as discussed earlier in sections 3.1 and 3.2, both iTunes and Windows Media were compromised through the extraction of these keys. Thereforem, the main problem with this approach is the secure storage and retrieval of these keys.
2. **Serial Number Based**. Most devices and device components contain serial numbers that can be used as an identifier. For example, GSM mobile phone systems have an Equipment Identity Register (EIR) that is used to identify individual mobile phones in the GSM network [131]. If this approach is used, it is necessary to ensure that it is not possible to falsify the serial number returned by the driver when queried; which requires the use of signed, trusted drivers to succeed in an open system. Thus, the main problem with this approach is the correct verification of the idenfier.

As discussed earlier, Kuntze and Schmidt's TPM based authentication tickets [120] provide a more secure approach to device authentication and their approach is effectively an evolution of both of the above approaches. This is because the TPM's goal is to provide secure key storage, and the identification scheme presented by Kuntze and Schmidt relies on the existing key pair embedded in the TPM.

However, until all forms of devices make use of TPMs, this approach cannot be used as the default case. Thus, device authentication will have to rely on either (or both) of the existing techniques. Thus, in addition to the problems raised by preventing portability, this raises the viability of solely relying on device authentication for DRM systems. The compound use of both user and device authentication would be a far better approach.

10.2 User Authentication through Time Limited Tickets

In this section we outline a ticket authentication solution for user authentication, including the system architecture and ticket design. Our solution focuses on proving a consumer's identity to the DRM controller, but the approach can be used to authenticate other parties in the DRM system. We also look at the security considerations for our solutions, and compare our solution to Kerberos.

10.2.1 General Overview

Instead of fixed certificates, we propose, like Kerberos, the use of time limited tickets for user authentication in DRM. Depending on the nature of the protected data and maybe the consumer's profile, the lifetime of the ticket needs to be variable. For example, music being transferred to a portable media player will get a longer ticket (e.g. 1 month) than a ticket for a computer connected to the Internet (e.g. 2 weeks). Tickets themselves will not be renewable, and thus should a ticket be compromised, it would be useless after it expires. This concept provides a solution to all the problems described earlier, although there is a trade-off between offline and online usage (online connectivity is still required to generate the tickets).

The authentication service needs to be trusted by the DRM system, and there are a number of different companies that can provide this service. These companies could be the equipment and component manufacturers (like Apple, Intel or Dell), network service providers (like AOL or Vodafone), content producers (like Sony or Universal), retailers (like Amazon.com) or third parties like Microsoft (through its Passport system) or Google (through its Jabber enabled user system). These services can also interoperate through federated identity management systems like Liberty Alliance.

An overview of the complete system is shown in figure 10.2, and is based on the overall system presented in chapter 9. The consumer only needs to interact with the authentication daemon, and can enter details such as authentication server, username, password and the target device identifiers (1). The authentication daemon interacts with the authentication server and maintains a database of valid authentication tickets (2,3). Depending on the authentication service itself, the password can either be sent over an encrypted session channel, or sent as a hash with the username. The aim of the ticket service is to attempt to integrate different authentication services under one authorisation ticket service.

When a DRM daemon needs to authenticate a consumer, it asks the authentication daemon for a ticket corresponding to some consumer (4), and then authenticates the consumer itself when it receives the ticket (5). Thus, the DRM controller is still ultimately responsible for granting access to data. If the consumer has multiple devices (or systems), (s)he does not need multiple authentication devices. Instead, other devices (or systems) can interact with the main authentication daemon to acquire tickets. Thus, mobile devices like iPods can acquire their tickets when they are being charged or being synchronised.

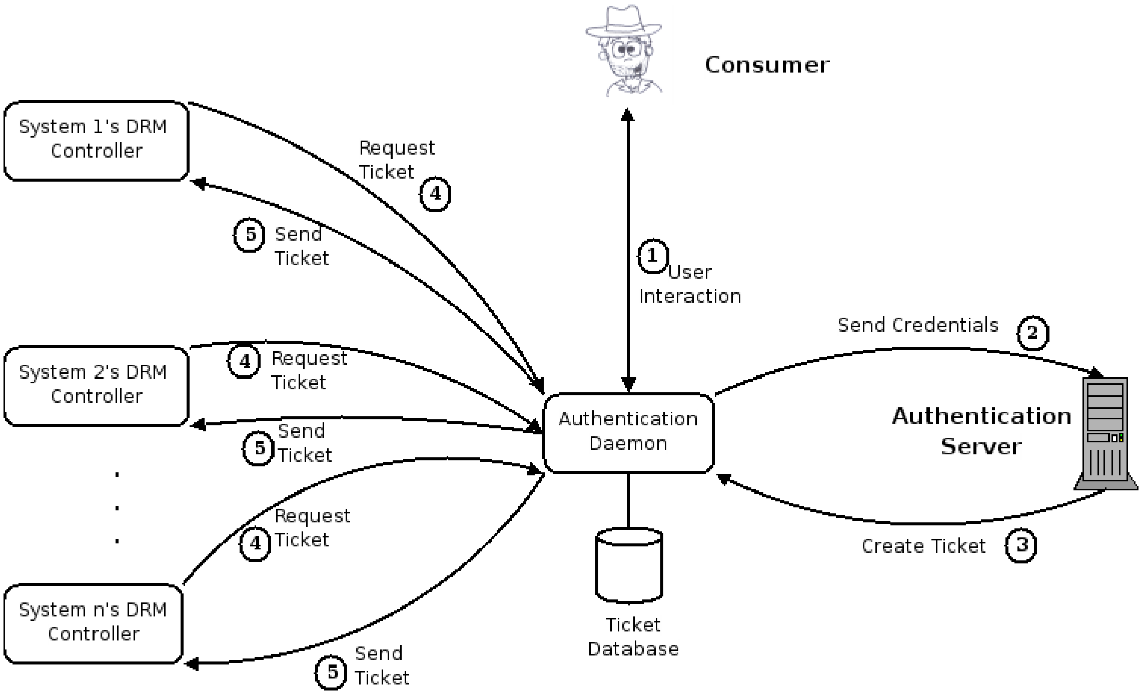


Figure 10.2: Authentication Ticketing System

If a requested ticket is not found, the consumer remains unauthenticated until the consumer acquires a ticket. While this does imply restricted access if the system is offline, it also ensures that access is granted only with access to valid tickets with the minimum intervention from the consumer.

10.2.2 Ticket Design

The general ticket solution does not solve the problem of replicated certificates (i.e. tickets themselves being distributed to attackers). Thus there needs to be a mechanism to restrict a ticket to a particular set of devices. If there is no such information, it is assumed that the ticket is valid for any device. However, this should not require the device itself making the request for tickets. As discussed previously, there are a number of mechanisms that could be used to identify devices, and we do not explore this any further.

The tickets could make use of a XML file (through the use of Security Assertion Markup Language (SAML) for example [62]), but with the absence of a tree structure, it could also be expressed as a flat file. Since XML processing is more expensive, and the relative simplicity of the ticket itself, we are in favour of a flat file description. The proposed file format is given in figure 10.3.

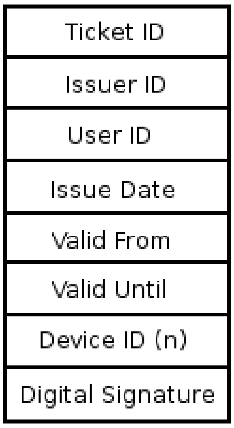


Figure 10.3: Proposed ticket format

X.509 Certificate and Kerberos Similarities

There are many similarities between the functions of a digital certificate and an authentication ticket and except for the device identifiers, an authentication ticket can be considered to be a subset of X.509 certificates and does follow most of the recommendations in the ISO Authentication Framework [176]. Data confidentiality is not a requirement for the authentication tickets, but ticket integrity is crucial. In this respect, this ticket format differs from Kerberos tickets [183].

Unlike Kerberos, the tickets are generated for use by the client for a specific device. Thus, replay attacks that are possible on Kerberos 4 and 5 [176] are not applicable to this scheme. However, attacks based on clock differences on client machines will still succeed.

Apart from a different ticket structure, our service is also more lightweight when compared to Kerberos. Unlike Kerberos, which utilises an encrypted message at all steps but one, our system requires only the communication between the daemon and the authentication service to be secured. Furthermore, there is only one device that needs online connectivity to function, as opposed to every device as required by Kerberos. We believe that our system is easier to adapt for current authentication systems, allows better scalability and is better for a wider range of devices when compared to Kerberos.

10.2.3 **Security Considerations**

Chain of Trust

The DRM controller needs to keep a list of authentication servers it trusts, and to limit the number of authentication servers that can be used with the system. If the DRM controller receives a ticket that it cannot recognise, it marks it as an invalid ticket, and refuses access to the protected work. Because the authentication daemon is a system that acquires tickets for usage and not the actual authentication of the consumer, a rogue authentication daemon cannot influence the decision of a DRM controller. Thus, the authentication daemon can remain untrusted without compromising the security of the system.

Rogue authentication daemons however can pose other problems as they can be used to harvest usernames and passwords from unsuspecting consumers; which could then be used to acquire tickets. This would be an indirect attack on the system, but is no different to phishing attacks on current authentication systems.

Secure Storage

Secure storage of tickets is not strictly required as long as ticket integrity can be assured. Because tickets are limited to devices and period of validity, replication of tickets does not pose any problems. Systems do however need to keep a public key chain of trusted authentication systems, which needs to be secure.

Ticket Confidentaility, Integrity and Non Repudiation

Except for privacy concerns, ticket confidentiality is not a requirement for the system. However, ticket integrity is of paramount importance, and as long as the private keys of the authentication servers are not compromised, the integrity and non-repudiation of the tickets are assured.

10.3 Other Forms of User Credentials

One of the features in current access control mechanisms, particularly MAC and RBAC based systems, is the provision of defining users as part of groups or roles[2], and allowing access to resources based on the membership of these groups or roles. Group and role based authentication is also a necessary feature for both media and enterprise DRM systems.

Media DRM systems are often used in their private capacity, and relationships between the different consumers will dictate the formation of these groups. For example, different members of the same family may purchase media, but every member of the family would like to use the purchased media. Group based user access is already a feature of most enterprise DRM systems, and these systems are closely aligned to the existing groups and roles established in the enterprise.

[2] As discussed by Jajodia et al. in [106], roles differ from group membership in that roles can be “activated” and “deactivated”, whiule group membership is more permanent.

The ticket presented in figure 10.3 could also be used as a role based credential ticket. The *User ID* field should contain the credential value instead of the user's id. This format would also be able to restrict a credential from being redistributed to other devices. The same process outlined for acquiring authentication tickets can be used to acquire role based credentials and thus, the type of ticket issued to the consumer will depend on the circumstances of usage.

So far credentials and tickets have been discussed as a form of establishing consumer identity. In RFC 2828, another potential usage of credentials is discussed: "*data that is transferred or presented to establish ... the authorizations of a system entity*" [180].

One of the requirements introduced by DRM systems, is the idea of *obligations* or *duties*. As discussed in chapter 8, all the parties of a use license can have obligations associated to them. Obligations could also be placed with regards to a specific permission in the use license. Obligations for the producer include quality of service, free upgrade guarantees while obligations for the consumer include payment for the product (perhaps after a period of evaluation) and providing feedback to the producer. Credentials can also be used to prove to the DRM controller the fulfilment of an obligation by a consumer.

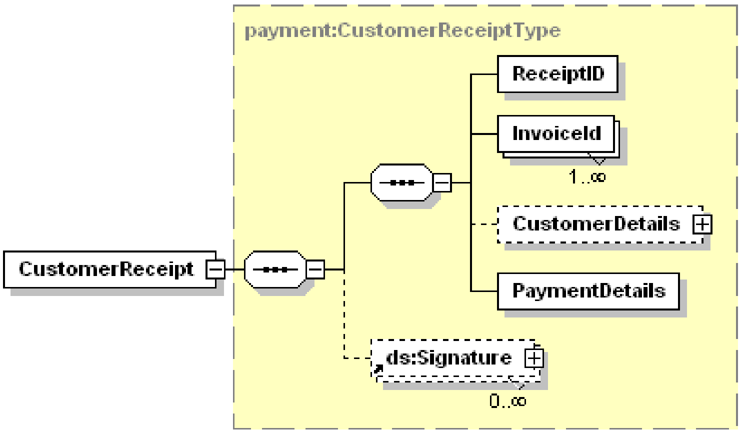


Figure 10.4: XML schema diagrams for a customer receipt as discussed in [38]

We will use the payment obligation to motivate this approach. In [38], we detailed an anonymous payment system for digital goods, with a focus on providing machine readable proof of payment. When a consumer pays for the service, the consumer is provided with a signed receipt (as shown in figure 10.4) to confirm their payment. Thus, this receipt can be considered as a credential to prove that the consumer has paid for their goods.

Note that existing e-commerce systems do not usually provide customers with signed receipts, and these receipts are usually not directly machine readable (without extensive pre-processing).

Thus, to use credentials to provde the fulfilment of obligations may require new infrastructure.

The proliferation of credentials implies a proliferation in the number of entities that need to be trusted in order to provide such a service. However, this can be minimised, if the use license lists the entities the licensors trust to provide a certain credential service. Thus, the providers of the credentials do not have to be directly trusted by the DRM controller, but only the licensors; which is the requirement in any case. This is discussed further in chapter 12.

10.4 Verifiable Digital Object Identity System

As discussed earlier, current identity systems for digital data do not provide any verification service. However, there is a simple solution to allow verification – with every mapping between the identifier and the object, include a digital signature of the digital object. Thus, resolving an identifier would not only locate the digital object, but would also verify whether the object is the intended object. A digital signature would also allow for verification of the object either offline (by including the signature as part of the identifier tag) or online through a related web-service. In this section we discuss the Verifiable Digital Object Identity (VDOI) System, which we believe is a better identity system for digital objects.

The VDOI system can be broken into four components:

1. Identifier Format
2. Identity Verification
3. Identifier Resolution
4. Management of Identifiers

10.4.1 Basic Setup

Like DOI, the VDOI is also an extension of the Handle service [150]. By basing the system on the Handle service, the identifier format and the resolution process will follow the protocols of the Handle service. This also provides persistence of identifiers. The VDOI system will also have a web service frontend that will handle the verification and management functions. SOAP will be the basis of the communication protocol for the web service functions.

10.4.2 Identifier Resolution

The VDOI System will based on the the Handle service, and thus will follow the protocols defined in RFC 3650 [186]. This should also mean that a Handle server should be able to resolve a VDOI identifier and vice versa.

Resolution should be possible by any VDOI server and can be handled in two ways. Firstly, the servers could keep a store of all possible identifiers and corresponding resolution addresses; but this could be a lot of data to store and thus be impractical. However, there should be a few "root" servers that could handle such storage in case of failures of original servers. The second approach would be to forward the resolution request to the appropriate server and then return the response back to the requestor. In this scenario, the server could also cache frequent requests for faster access.

10.4.3 **Identifier Format**

As explained earlier, VDOI is based on the Handle system and thus the identifier format is also based on the Handle service. It should be possible to extend this format to cater for internationalisation through the use of Internationalized Resource Identifiers (IRIs) as detailed in RFC 3987 [77]. However, the current scheme presented in this section, unfortunately does not conform to the IRI specifications. The proposed format is detailed below:

vdoi://dir_id.reg_server/object_class/id/version

The *vdoi* tag represents the service identifier. The *dir_id* and *reg_server* are part of the Handle service protocol for identifiers and this identifier is thus compatible for resolution with any Handle service. Verification support can however only be provided by the VDOI service. The registration server is responsible for the allocation of the actual identifier. The directory identifier represents the server that allocated the identity of the registration server (for example, 10 represents the DOI foundation). The directory identifier is handled by the Handle system.

The *id* is generated by the registration server and can be expressed in any alpha-numeric scheme desired. It is left up to the registration server to make sure that the id is unique. Combining the unique id with the rest of the identifier guarantees global uniqueness. Like the *id*, the version scheme does not have a prescribed format. A suggestion is to use

MajorVersion.SubVersion.MinorVersion

format. Thus two objects of different identifiers may have the same id, but the use of different version numbers provides globally unique identifiers. It is recommended that objects with different versions keep the existing identifiers (or maybe change identifiers at major versions). The major advantage is the flexibility in licensing, as licenses could therefore implement a wildcard scheme for access to objects.

The *object_class* tag is a feature aimed to ease administration of identifiers. We propose to use a set of integers to denote these classes, and thus standardised mapping could be useful. This approach also increases the number of identifiers that can be used by the registration server. Identifiers must be case insensitive. Although the handle system does prescribe the use of case sensitive identifiers, the DOI foundation have commented on the complexity of such a system [150].

10.4.4 **Identifier Verification**

When an identifier is issued to a digital object, a signed hash of the digital object is stored along with the resolution and supplementary information (identifier of owner of digital identifier for example). The signed hash will be used for the verification service provided by the VDOI server.

For verification, the user[3] (or service) submits the identifier of the digital object in question as well as the hash of the digital object (taken by the user or service) to the VDOI. The server will then verify the submitted information in relation to the information stored in its own database, and return either a message confirming validity or invalidity to the requestor. Thus this process removes the user's knowledge of the true identity of the hash if the object does not match the identifier reducing the chance of a successful attack in attaching a false identifier to a digital object.

Any server should be able to provide verification for a VDOI identifier. VDOI servers could approach this in two ways. Firstly, the servers could keep a store of all possible identifiers and corresponding hashes; but this could be a lot of data to store and thus be impractical. However, there should be a few "root" servers that could handle such storage in case of failures of original servers. The second approach would be to forward the verification request to the appropriate server and then return the response back to the requestor. In this scenario, the server could also cache frequent requests for faster access.

A self verification scheme can also be supported if the digital object is wrapped in an envelope with the identifier and the digital signature from the VDOI. However this scheme does allow for the possibility of attaching a false identifier to a digital object if both objects have the same hash. With the use of a strong and secure hashing algorithm however, the chances of successfully creating such an attack can be substantially lowered.

[3] The user of the service to create and maintain identifiers will always be a producer in the DRM value chain.

10.4.5 Management of Identifiers

Management of identifiers fall into two categories – the registration of a new identifier and the maintenance of the identifier. Because the identifiers are persistent, there is no need to delete an identifier once an association has been made. Maintenance of identifiers include updates of resolution addresses and updates to meta-data of the registration such as owner of the object etc.

Updates of meta-data and resolution addresses imply that other servers that had the information will also need to be updated. However, it is highly likely that updates will not be regular, thus synchronising servers at regular intervals should be sufficient.

In the remainder of this section, we present a protocol for registering a new identifier. All communication must take place using signed SOAP messages and through an established encrypted tunnel (like an SSL session). It is assumed that the server has access to the user’s public key.

```
V   => VDOI Server
R   => Requestor

Rid  => Requestor Identifier
Vid  => VDOI Server Identifier

Oc  => Object’s class
Ov  => Object’s version
Os  => Object’s digital signature
Oid => Object’s identifier
Ooid=> Object’s old identifier (optional)
Or  => Object’s resolution address

ad             => Additional Data (Optional)
t1, t2, t3, t4 => Timestamps
n1, n2, n3     => Nonces

R->V: Rid, Oc, Ov, t1, n1, Ooid
V->R: Vid, Rid, Oid, t2, n1, n2
R->V: Rid, Oid, Os,  Or, t3, n2, n3, ad
V->R: Vid, Rid, Oid, t4, n3
```

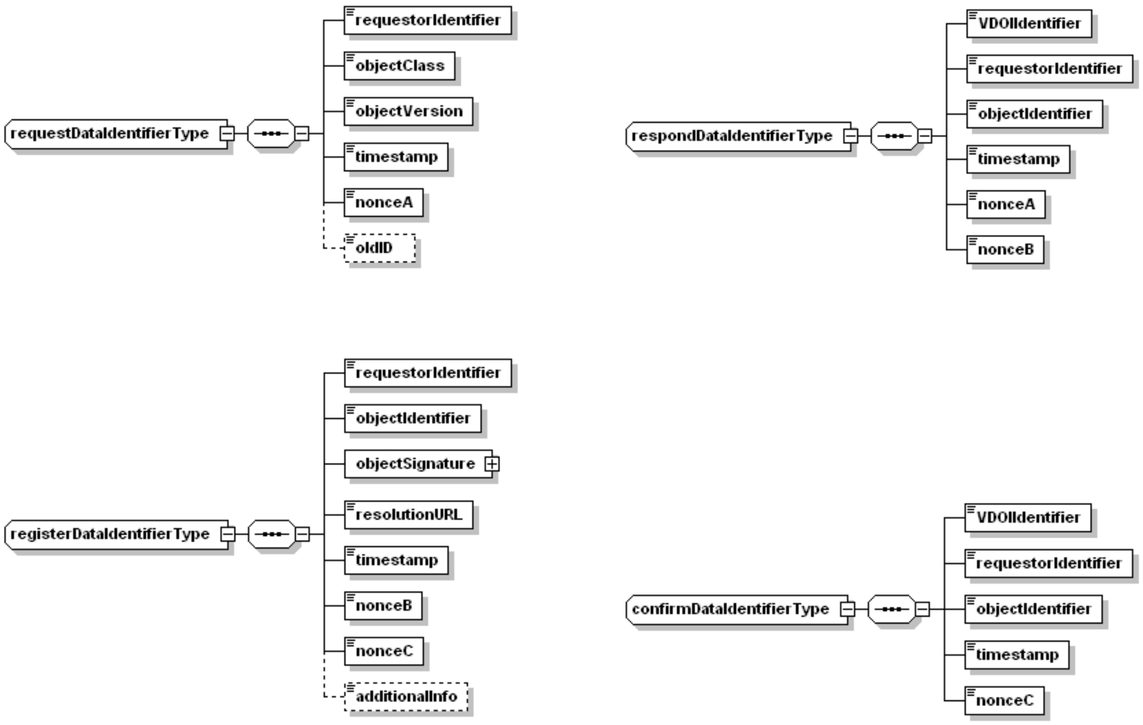


Figure 10.5: XML schema each step of the identifier registration protocol for VDOI system.

In the first step, Ooid refers to an existing identifier if this is a registration for a new version. Only the server that issued the existing identifier can issue a new version. The nonces are used to maintain linkages between communications, while the timestamps maintain freshness of messages. Timestamps are also useful in detecting dead connections should a requestor not follow through with the protocol. Additional information in step 3 could be information required by the registration server.

In step 2, an identifier is set aside for a set amount of time. The time interval allows the requestor to add the identifier to the object (for example in the title of document). Step 4 serves as a confirmation of registration for the object. XML schemas for each step of the registration protocol is shown in figure 10.5.

Once the server acquires the object's digital signature, it extracts the hash and signs it. The registration server never gets a copy of the digital object, just the metadata, the hash and the respective identifier. Thus the VDOI system can be used for sensitive data on an open network like the Internet. This promotes data privacy and security especially as data itself does not have to leave the control of the owner to receive an identifier.

10.4.6 Security Considerations

Verification of identity is a security service, and thus there is a need to ensure the integrity of the service. However, the service provided is essentially a public one, and thus accessibility of the service also needs to be taken into account. The identification system provides the following classes of services:

1. Resolution of identifiers
2. Verification of identifiers
3. Management of identifiers
4. Management of the system (administration etc.)

Resolution and verification are free public services that must be able to function anonymously. Registration and management of identifiers (for example the change of resolution address) could be a paid service and thus may have restricted access. Management of the service is a private service and thus must have restricted access. For the rest of this section, we examine the services against the 5 security services identified in ITU's X.800 specifications as well as availability, which is not explicitly mentioned in X.800 [183], and the chain of trust involved.

Chain of Trust

In the system, the user (registering an identifier) is always trusted to provide the correct data. The original registration server is also trusted not to tamper with the data (but any cached copies can be seen to be untrusted).

It is the registration server's responsibility to maintain and secure the records. Some of this functionality is provided by the Handle service framework. In an implementation, the registration and verification service is a business opportunity and thus there will be economic incentive for the service to maintain and secure the records.

Authentication and Access Control

Authentication and access control are services which are controlled by the administrators of the servers. These services are only required to authenticate administrators and for the management of identifiers; and thus restricted to server management.

Data Confidentiality

This system provides a security service, and thus it is paramount that the data is stored in a secure environment. The use of a secure tunnel for management provides data confidentiality at the transport level. Because the system does not require the actual data being registered, confidentiality of the original data is assured.

Data Integrity

This system provides a security service, and thus it is paramount that the data stored are correct. The hash of the object is signed by the registration service and this provides for a check of data integrity for the verification. The use of signed hashes of all the stored data could help with data integrity for the remainder of the records. The use of signed messages provides data integrity for all communication.

Non Repudiation

The use of signed SOAP messages for all communication provides non repudiation for all communication. Non repudiation of requests would depend on the requestor (user) management systems deployed.

The hashes are signed by the registration service and thus provide for non repudiation.

Availability

Availability is of critical importance for persistent identifiers. We propose the use of multiple root servers that hold copies of all data, and the use of a distributed architecture should allow for a higher tolerance of denial of service attacks or high load of requests.

10.5 Summary

Authentication is one half of evaluating access control, and in this chapter we have discussed all three forms of authentication associated with DRM systems: users, resources and devices.

We have presented a ticket and credential based user authentication system which provides a more flexible solution than current solutions such as Kerberos and X.509 certificates. This solution provides a balance between online and offline authentication, and requires minimal user interaction – a necessity when considering devices that provide limited user interaction capabilities. The extension of the system for credentials also provides a useful mechanism to prove the fulfilment of user obligations.

Existing resource identity systems focus on resource identifiers and do not provide any mechanisms for the verification of resource identity. In this chapter, we discussed Verifiable Digital Object Identity (VDOI) System, which provides a complete resource authentication system. We have previously discussed this system in [35].

Chapter 11

Creating Protected Works

Traditional access control approaches are closely tied to the underlying operating system, and thus, access control is only applied within a predetermined boundary. In DAC based systems, this boundary is the physical system. In MAC and RBAC based systems, this boundary extends to a network of systems controlled by a set of control systems. DRM however aims to provide access control without imposing any boundaries and having any single point of control.

Thus, the traditional access control approach of not needing additional operations on the target data (such as encryption, fingerprinting and signatures) is not possible for DRM. In fact, until every computing device provides the means to enforce DRM access control policies, protected data needs to be encrypted.

This means that there is a requirement for DRM systems to create a standardised packaging format – one that can accommodate any payload type, and accommodate a variety of security features. Furthermore, as discussed in chapter 4, Koenen et al. discussed the benefits of full format interoperability as opposed to other forms of interoperability [118]. A standardised packaging format is a crucial step towards full format interoperability. In this chapter, we introduce a more generic approach to DRM formats through a layered packaging format, together with a discussion of the workflow associated in creating DRM packages.

11.1 The Packaging Service

Earlier in chapter 1.1, we defined any entity that wishes to change the form of a DRM package as a producer. We noted that the producer does not have to be the original author of the work – for example, a distributor that packages supplementary data as part of the DRM package, is also a producer.

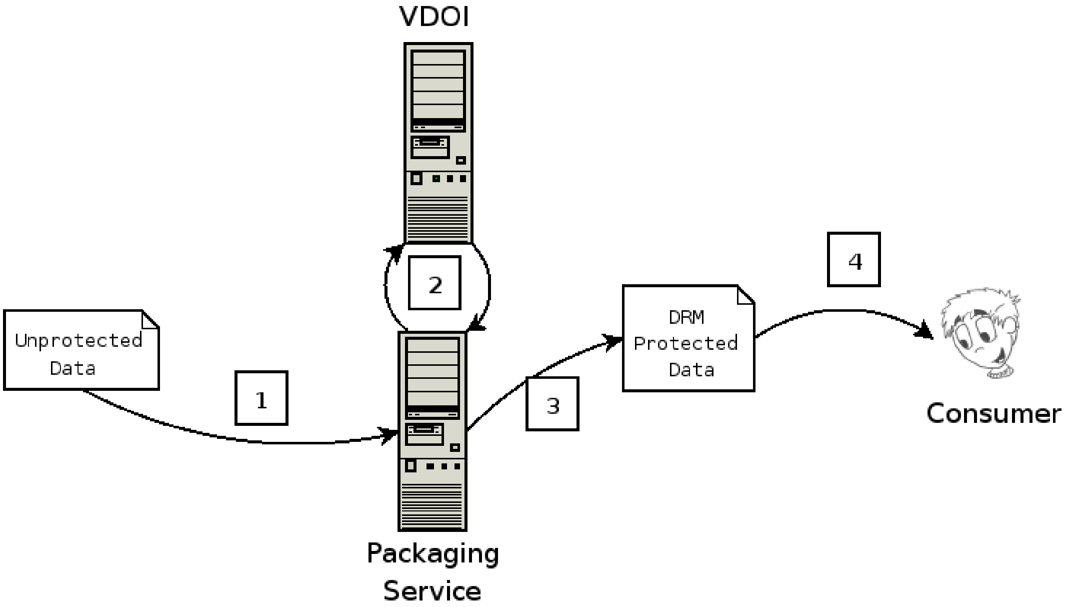


Figure 11.1: DRM Packaging Service Workflow

11.1.1 The Packaging Workflow

Production of DRM packages can be centralised to a service that serves a number of producers, or individual producers can have their own packaging service. Either way, the packaging service follows the same workflow outlined in figure 11.1.

Step 1: The producer transfers the data file(s) to the packaging service over a secure connection (for example through a SSL tunnel). The producer also selects the options he/she would like on the DRM package, typically including encryption format, watermarking, compression etc. These options are further discussed in section 11.2. Figure 11.3 shows the XML schema that could be used to communicate these options.

Step 2 & 3: The packaging service gets an identifier from the VDOI (see section 10.4 for more details on the VDOI service) and then assembles the DRM package. The details of the packaging operation are discussed in section 11.2.

Step 4: The protected package is then forwarded to the consumer of DRM package (or back to the producer). The producer of the package can also be given a signed receipt acknowledging the completion of the packaging operation. Figure 11.2 shows the XML schema of such a receipt.

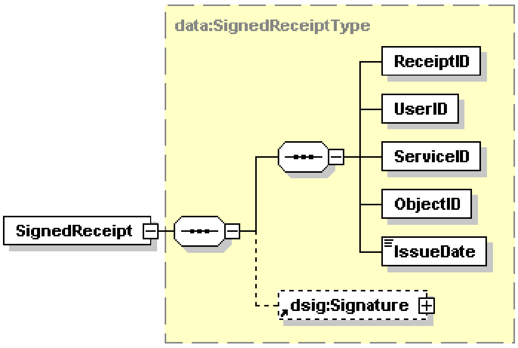


Figure 11.2: XML schema of a signed receipt from the service producer

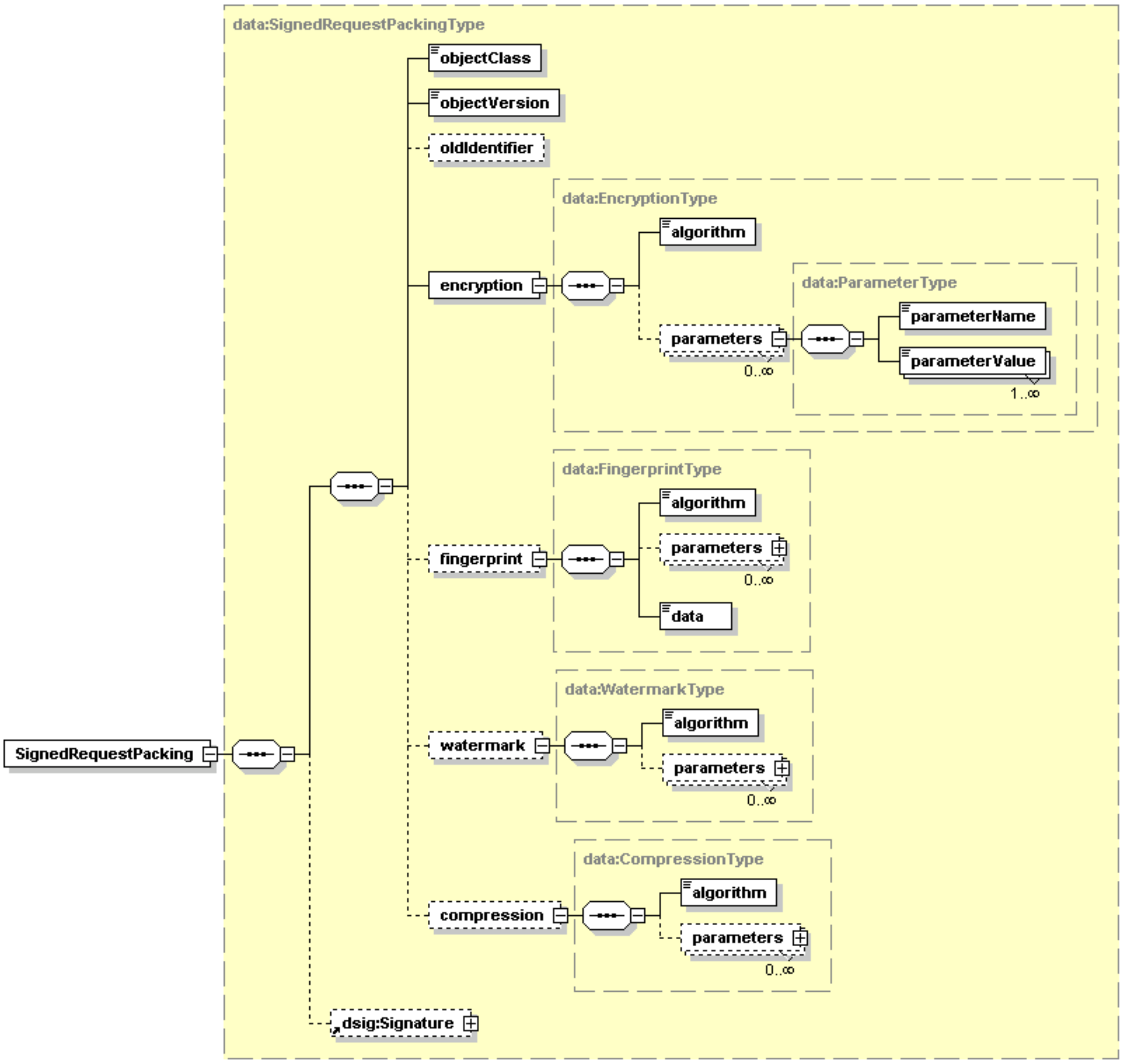


Figure 11.3: XML schema of a signed request to create a DRM package

11.1.2 Encryption Keys

Since the packaging service is responsible for encrypting the protected packages, it will need access to the encryption keys. There are two options: either the *producer* supplies the keys (as part of the encryption details in figure 11.3; or the *packaging service* provides the encryption keys. In the latter case, the packaging service will need to inform the producer which key was used (as the packaging service could host a number of different keys).

11.1.3 Regulation of the Packaging Service

Regardless of whether the packaging service is a centralised operation or not, it must be noted that the packaging service processes unprotected data, and thus can be a source of distribution for unprotected data. For this reason, there is a strong need for the packaging service to have a comprehensive set of regulations regarding how unprotected data is treated; including caching strategies and temporary storage of the unprotected data.

Furthermore, the packaging service also has access to encryption keys; provided by either the producer or belonging to the packaging service itself. This also provides a likely point of attack for the DRM chain – it is easier to attack a single service, which could yield the necessary keys to decrypt the content rather than attack the content itself.

For these reasons, there is a need for the packaging service to be strongly secured, and there needs to be strong control over data processed by the service. Regulation need not be through any statutory body; but could be simply well defined parameters available to the producers before they use the packaging service. If the packaging service is provided as a paid web service (for the general public), then regulation through security audits could be preferred by the producers.

11.2 A Layered Approach to DRM Packages

A standardised data format for DRM, is perhaps only second to a standardised REL in the quest for inter-operability. In [107], Jamkhedkar and Heileman promoted a layered approach to DRM. They argued that a layered approach to networking, through the 7-layer OSI model, has been instrumental in achieving interoperability for computer networks, even though the exact OSI model has not been followed. Thus, they argued, a layered approach to DRM was also necessary, and identified different service layers for DRM.

We have adopted a layered approach to data format for DRM packages for the same motivations – layered models provide the flexibility in catering for a wide number of capabilities and

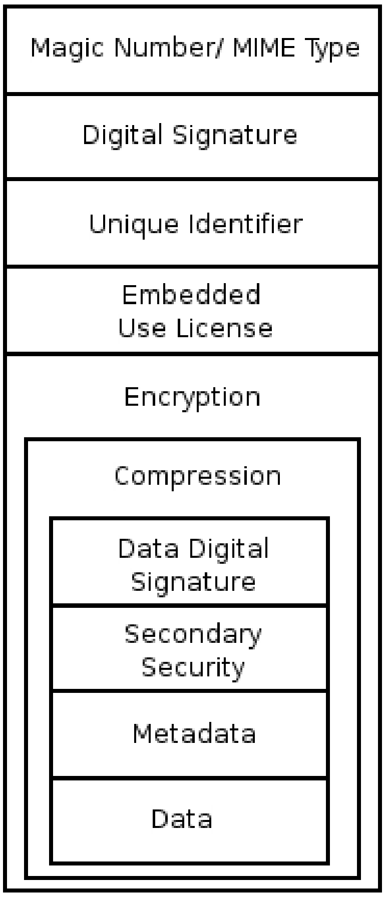


Figure 11.4: A layered view of DRM Packages

requirements, but at the same time, provide the possibility for interoperability.

Another advantage offered by a layered approach, is with respect to processing and assembling DRM packages. The layered approach intuitively follows the processing and assembling steps for DRM packages, and thus aids interoperability for DRM packages between different implementations of DRM systems.

A layered model for DRM data packages, as we propose is shown in figure 11.4, but unlike the OSI layer model, our approach is less linear. Our layered model can be divided into three distinct sections:

1. The Data Layer
2. The Compression Layer
3. The Packaging Layer

with the data layer nested withing the compression layer, which is in turn nested in the protection layer.

When discussing full format interoperability, Koenen et al. also discussed the long time it takes to adopt standards, especially with regards to specific technologies to be used as part of that standard [118]. The layered approach we propose reduces the scope for any potential standard, as the specific technologies do not need to be standardised, just the format of the layers themselves. For this reason, this approach would be more useful in achieving full format interoperability.

11.2.1 **The Data Layer**

The data layer consists of all the data that the producer wishes to include in the DRM package, in their unencrypted, usable form. This includes the data being protected, and metadata and supplementary data associated with the target data, such as album covers and bibliographic data.

The data layer also consists of information about secondary security features such as fingerprints and watermarks that can be applied to the target data (and possibly metadata and supplementary data). The information includes algorithms and parameters, that will be required by any application that wishes to test the data for such features. Both the metadata and secondary security layers are optional, and producers may wish not to include these layers in their packages.

The data layer can also have an optional data signature, which has the signatures for the data contained in the packages. The data signature is aimed at verifying the integrity of the data contained within the package, as opposed to the digital signature present in the packaging layer, which provides verification of integrity for the entire package. This approach is only useful if the package producer is not the creator of the data, and rights holders who control a greater proportion of the producer pipeline will not need this layer.

11.2.2 **The Compression Layer**

Pretty Good Privacy introduced a compression layer before encryption. In [182] and [183], Stallings detailed some virtues of this approach:

1. Cryptanalysis becomes more difficult, as the potential attacker has to work with a compressed binary file, which cannot be easily recognised, when compared to the target data.

2. Compression should reduce the size of the data files, and thus improve encryption and decryption performance.

For the above reasons, we have also included an optional compression layer before the package is assembled. The compression layer may not make sense when the target data is already in a compressed form, but should improve performance for other type of data files.

11.2.3 The Packaging Layer

The packaging layer comprises of a number of layers, responsible for assembling the actual DRM protected package. The innermost layer is the encryption layer, which encrypts the compressed package, preferably using strong RSA encryption. The flexibility of our approach means that different compression and encryption algorithms can be used, depending on the required level of protection. Encryption is not strictly required for access control; but for persistent protection to be achieved, encryption is required until every device respects DRM use license conditions.

Embedded use licenses are optional licenses that can be included with the DRM package. Many existing DRM systems use this approach; and while it is simpler to implement, it does create a problem with portability. We believe that, should there be an embedded use license and an external use license for the same DRM protected data, the embedded license should be considered invalidated.

The unique identifier refers to the data, and not necessarily to the DRM package. The unique identifier allows the DRM controller to establish the relationship between the data and the use license. We provide a comprehensive discussion on data identity in section 10.4. The unique identifier is a mandatory layer. The digital signature provides integrity for the entire package, as opposed to the digital signature in the data layer, which proves integrity of the data only. Unlike the latter layer, this is a mandatory layer.

Lastly, there is an optional magic number/MIME type layer. Currently DRM packages do not have any MIME types. One of the advantages offered by MIME Types is the easy identification of the data type, without necessarily examining the contents of the file. This is particularly useful for hardware and operating system layer DRM controllers, as quick identification of non DRM enabled packages should lower the performance overheads associated with such implementations.

11.2.4 Unpacking the Protected Package

One of the advantages of the layered approach discussed above, is that both production and consumption of the resource are easily described. Once the DRM controller identifies a file as DRM protected (through the magic number/MIME type), it verifies the integrity of the package through the package's digital signature. Using the package's unique identifier, it retrieves a use license and decides whether the consumer can be authorised to access the package (see chapter 9). The DRM controller can make use of the embedded use license if needed.

Once authorisation is received, the DRM controller decrypts and decompresses the package. Depending on the requirements, the DRM controller can verify the integrity of the data files as well as conduct secondary security checks, such as fingerprint verification. Once these tests are passed, the DRM controller allows the requesting application access to the data file.

11.2.5 **Comparison to other Packaging Formats**

The OMA DRM packaging format is one of the few open DRM packaging formats, and comes in two flavours: Discrete Media Profile (DCF) and Continuous (Packetized) Media Profile (PDCF) [147]. Both of these formats are based on the ISO Base Media File Format, which provide a file format suitable for the distribution of multimedia by providing certain information as part of the file headers. In the OMA DRM extensions, the payload contains the OMA DRM container, which contains data such as an embedded use licenses and a URI for the license server.

The packaging format presented in this section is more flexible and can cater for additional processing such as watermarks, fingerprints and compression. Furthermore, our format can easily be encoded in a carrier file format such as the ISO Base Media File Format if required for specific environments. For the generic implementation however, the ISO Base Media File Format adds additional processing overhead.

11.3 SUMMARY

In this chapter we outlined the process involved in creating DRM packagers. We introduced a layered approach to DRM packages, and outlined the composition of such a layered model, together with how to assemble and interpret the layers. Our layered model had three distinct layers, which are further subdivided into different layers. The innermost layer, is the data layer, comprising of the target data, associated metadata and secondary security features such as watermarks and fingerprints. The middle layer is the compression layer, compressing the contents of the data layer. The outer layer is the packaging layer, and perform operations such as encryption of the compression layer and adding identity information to the package.

We also discussed the workflow associated with assembling DRM packages, which could be implemented as a third party web service, or for individual users. Regardless of the implementation of the workflow, there is a need to have strong regulation and control over such a service;

particularly if the service is provided to the public by third parties..

CHAPTER 12

WEB OF TRUST & KEY DISTRIBUTION IN DRM

RFC 2828 defines trust as

> The extent to which someone who relies on a system can have confidence that the system meets its specifications [180].

So far, we have discussed the different components that can be involved in a complete DRM system, and how these components interact with each other. Thus, trust in a DRM system is determined by how much confidence the producers and consumers have in the implementations of these components.

However, the proliferation of the number of entities that need to be trusted for a system to be trusted, implies that trust in such a system can also be easily broken. This is because, as the required number of trusted entities increase, there is an increased risk in one of these entities being compromised; and if such a compromise does take place, then the entire system could be compromised.

In [98], Heileman and Jamkhedkar discussed how the flexibility and interoperability potential of a system decreases if specifications define concrete components of a system[1]. Similarly, specifying the specific trusted components of a system will also limit interoperability and flexibility.

Thus, there is a need to limit specifications of which components are trusted in a DRM system, and, as far as possible, allow the flexibility and interoperability by maximising the number of components that can be trusted in a DRM system. The traditional chain of trust is thus too

[1] In the paper, the authors discuss how specifying a complete layer in their layered framework for DRM system [107] limits the interoperability and flexibility of the surrounding layers

linear, and not completely suitable to describe the trust relationship between the components of the DRM system. For this reason, we discuss a *web of trust* for DRM systems. Note that this is not unlike the web of trust found in PGP systems for determining the trustworthiness of a PGP certificate [182, 183].

Once we describe the web of trust involved in our DRM framework, we also discuss the key distribution strategies of our framework. We recognise that key distribution and trust chains are not synonymous [180], but our key distributions follow the trust web which makes their discussion appropriate in this chapter.

Note that we do not discuss the specifics of what type of keys should be used in this chapter. Specific use cases will call for the use of specific key types - the value of the content could determine the key size for example, and the key distribution strategy could determine whether to use public key cryptography or symmetric keys instead. Likewise, the cryptographic algorithms used would also depend on the use cases.

12.1 Web of Trust for DRM

In this section, we will build the web of trust; motivating our reasons at each iteration. An arrow from A to B denotes that A is trusted by B. The web of trust is for a protected work, and it is between a producer and a consumer at any stage of the DRM usage chain.

12.1.1 The DRM Controller

We start at the DRM controller, a component that needs to be trusted by both consumers and producers of DRM packages. For producers, the DRM controller needs to be trusted to enforce the access control policies set out by themselves. Similarly, the consumer needs to trust that the DRM controller enforces the use license in a fair manner, and does not prohibit access or enforce access control policies when it is not meant to.

12.1.2 An Independent Verification Authority

Both producers and consumers can have difficulty trusting DRM controllers. Producers could be worried about the integrity of a DRM controller as implemented in the consumer's device; while the consumer could be worried about the trustworthiness of a DRM controller supplied by a producer. For this reason, an independent verification scheme can be envisaged to test the trustworthiness of various implementations. In [46], Bartolini et al. envisaged such a role player for DRM systems. The independent verification authority is a third party in our formal model.

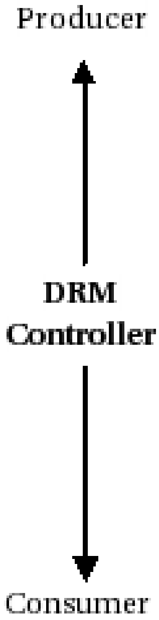


Figure 12.1: Web of Trust with only the DRM Controller

The producer's fears are harder to allay: verification of hardware or a operating system based DRM controllers is possible, but because DRM protected works are themselves not necessarily executable, there is no mechanism to ensure that the DRM controller is switched off or bypassed during actual usage by the consumer. It cannot be expected for a consumer to run a verification service parallel to the use of a protected work. Rather, it can be seen as a certification program, run at periodic intervals. And, as suggested by Jamkhedkar et al. in [107], the level of trust in a DRM controller could be a factor in determining security levels and license terms and conditions.

The consumer's fears are much easier to allay through a third party verification service. Independent verification on a particular DRM controller is easily achieved, and will be a similar process to existing software or hardware tests conducted by consumer organisations and the media. However, to prevent a repeat of dangerous DRM controllers, such as the SONY-BMG Rootkit [171], certification and verification has to be conducted before the controller is released.

12.1.3 **License Server**

After the DRM controller, the use license is the most important component of the protection mechanism, as it specifies the access control rules for the protected work. Thus, the DRM controller must trust the integrity and validity of the use license and its issuer, the license server. Likewise, the producer must trust the license server to provide accurate use licenses. The consumer however does not require a direct trust relationship with the license server. As we mo-

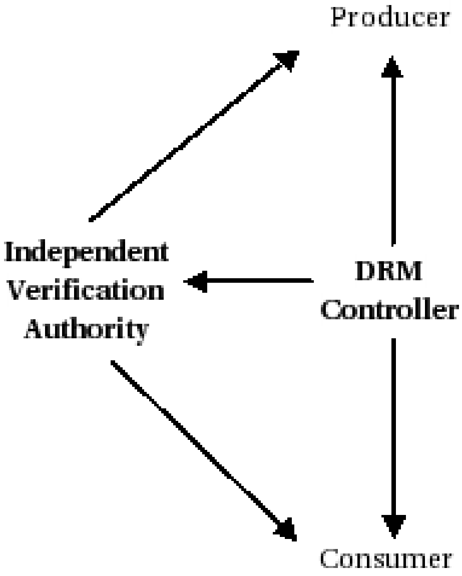


Figure 12.2: Web of Trust with the DRM Controller and an Independent Verification Authority

tivated in chapter 6, the use license is a contract between the consumer and the producer; and should the use license reflect terms that were not agreed to by either party; the agreement is non binding. Thus, should the consumer feel that a license server does not provide a service that it should, (s)he is free to use another service, or acquire legal help in the matter.

12.1.4 Authentication and Credentials Service

While the use license represents the authorisation component of access control, authentication is provided by authentication servers and credentials services (as discussed in chapter 10). These services need to be trusted by both producers and consumers. For producers, these services must provide legitimate authentication services for users, devices and resources. For consumers, these services must be provided without compromising sensitive private data, that may be required to establish the legitimacy of their claimed identities.

The DRM controller does not need to trust the authentication services directly. All it requires is proof of authentication that is acceptable for both consumers and producers of the protected work. The medium for such a communication is the use license, and thus, as long as the authentication credentials presented by the consumer are listed as a trusted source in the use license, the consumer can be considered trusted. For this reason, the authentication and credentials services need to be trusted by the license servers; but not by the DRM controller itself.

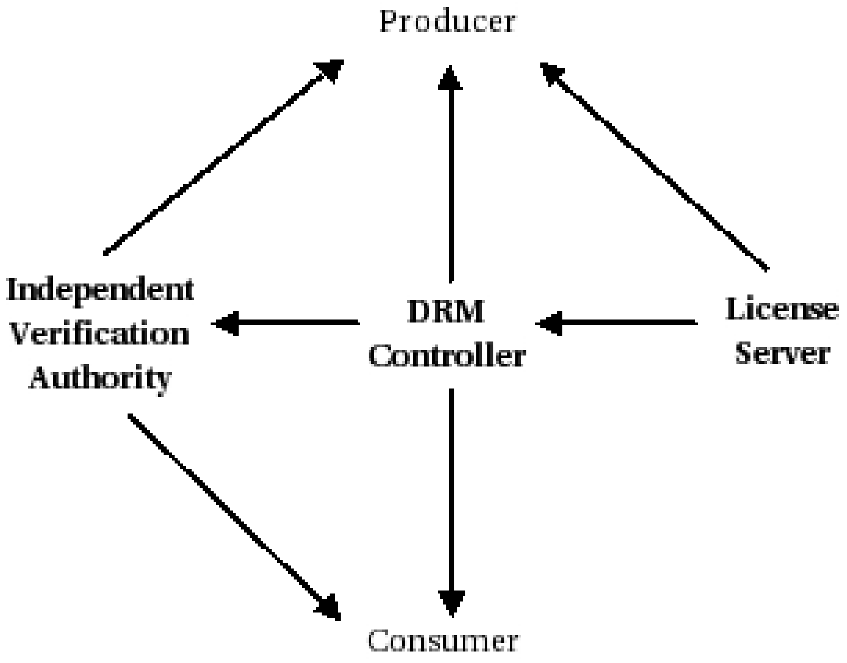


Figure 12.3: Web of Trust with the DRM Controller, an Independent Verification Authority and a License Server

12.1.5 **Packaging Service**

The packaging service needs to be trusted by the producer, and no other role player in the DRM system. The producer and the resource identity service are the only role players that interact with the packaging service; but the packaging service only handles sensitive data from the producer. Once the protected work is assembled, the packaging service does not play any further role in the DRM lifecycle, and thus does not interact with any other role player.

12.2 Key Distribution in DRM

After establishing the web of trust between the different role players in the DRM system, it is possible to define the key distribution between the role players. There are two type of keys that need to be distributed:

1. Public keys to verify digital signatures.
2. Decryption keys to decrypt data.

12.2.1 **Public Keys for Signature Verification**

The following role players will require their signature verified:

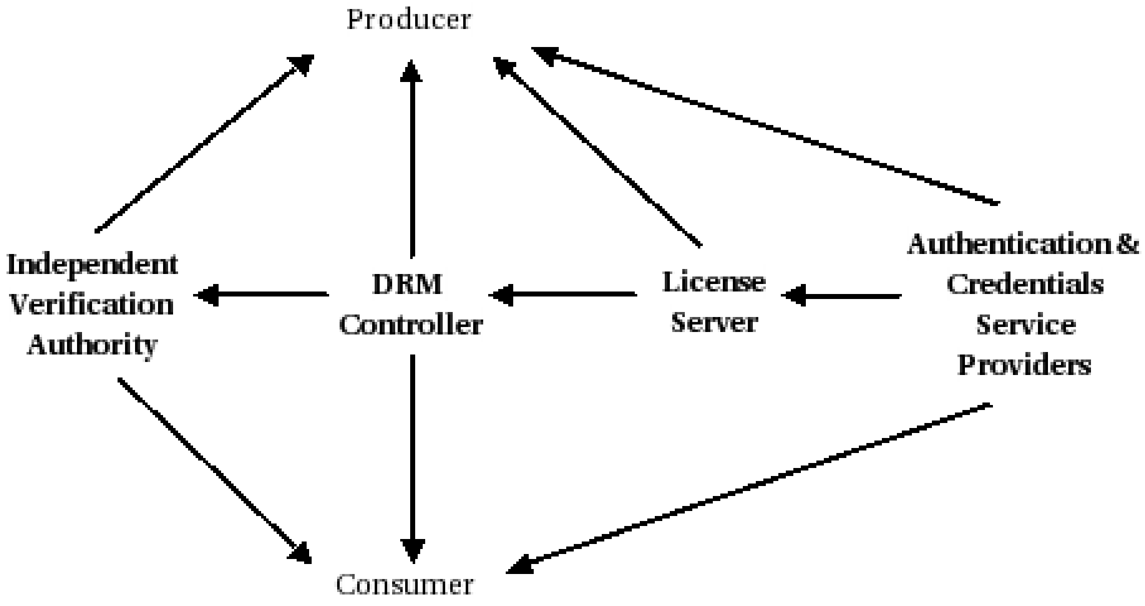

Producer
Independent Verification Authority
DRM Controller
License Server
Authentication & Credentials Service Providers
Consumer

Figure 12.4: Web of Trust with the DRM Controller, an Independent Verification Authority, a License Server and Authentication & Credentials Service Providers

1. The License Server
2. The Independent Verification Service
3. The Packaging Service
4. Authentication and Credentials Services

The verification service is an independent service from the DRM system, and thus it is not part of the DRM system itself. The verification it provides is geared for the producer and consumer, and thus, the public key distribution is geared for the producers and consumers.

The remaining keys are however all geared for other role players in the DRM system. And, in each case, the DRM controller is effectively the only role player that will actively verify the signatures of the license server, through the use license, the authentication and credential services through the credentials they provide and the packaging service through the protected work.

The naive solution, is for the DRM controller to keep a database of trusted license servers, authentication and credentials servers and packaging servers. While this solution achieves the verification requirements, it does have a potential to become inflexible. Authentication requirements, for example, will depend on the nature of the protected work; sensitive enterprise information will require authentication tied to the enterprise, while authentication for media may be

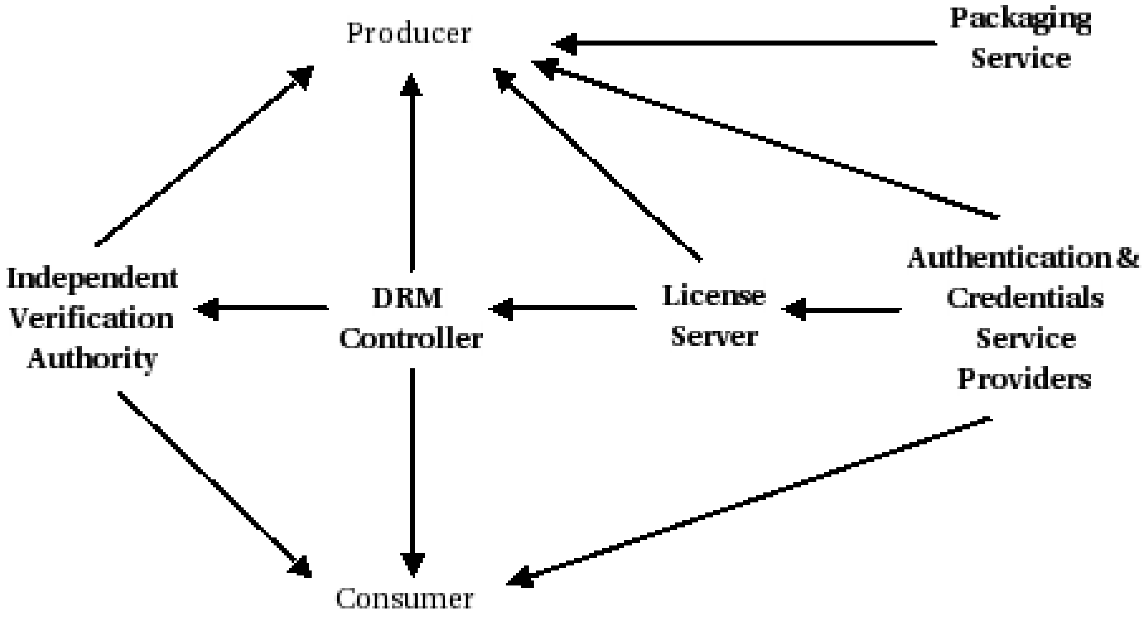

Producer
Packaging Service
Independent Verification Authority
DRM Controller
License Server
Authentication & Credentials Service Providers
Consumer

Figure 12.5: Web of Trust with the DRM Controller, an Independent Verification Authority, a License Server, Authentication & Credentials Service Providers and a Packaging Service

less strict.

Furthermore, with a need for the DRM controller to be secure, the control of which services are trusted is removed from the consumer and producer and replaced by the vendor of the DRM controller. It is impractical to ask the producer to define the domain of trusted servers for the DRM controller, as the producer does not have control over the consumer's DRM controller. While well known producers can influence the vendor's control of the trusted servers, smaller producers may not be in such a position, and thus may suffer.

Another approach would be to use trusted certification authorities (CAs), as used in current e-commerce systems. In this scenario, the DRM controller will have a database of trusted CAs, and accept any digital certificates that are signed by these CAs. However, this will mean that any service trusted by the CA can grant access, even if this service is not trusted by the producer, which breaks the web of trust.

Our approach is to rather use the use license to state the trusted authentication, credentials and packaging service (with regards to the specific protected work). In this approach the producer can directly specify the services they trust, providing a higher degree of flexibility. This approach means that the DRM controller needs only a database of trusted license servers, which will still mean that there could be license servers that can issue licenses for a protected work, without being trusted by the producer. We solve this through our approach to decryption key distribution.

12.2.2 Decryption Keys for Protected Data

Before we discuss the distribution of decryption keys, there is a need to identify the purpose of these keys. So far, we have only discussed one file that is encrypted: the protected work. Thus, there is a need to distribute the decryption key for the protected work, and the manner of this distribution needs to be secured.

In most of the current systems, this key is distributed through the use license. This approach makes sense in our case also: a use license cannot be used if it does not provide the means to decrypt the protected work. Thus, only license servers trusted by the producer can be used to generate use licenses.

This approach creates a problem however, a problem that has been exploited in current systems like iTunes and Windows Media (see chapter 3): how does one protect the use license? We have identified three different approaches that could be used, each with their advantages and disadvantages and we discuss them below.

Distribute Use License Key with Authentication Tokens

This approach could be the easiest to implement, but may have implications for consumer privacy. In this approach, the use license can no longer be used to detail the trusted authentication servers, but the producer still controls the trusted authentication server through the key distribution. However, there is a requirement for the producer to keep a tight control over the authentication server. This approach will be ideal for enterprise systems, but for systems designed for mass public usage, such tight controls have implications for consumer privacy, as we have previously discussed in chapter 10. Furthermore, the authentication ticket itself needs to be encrypted or deleted after usage, which adds another level of complexity to the system.

Identity Based Encryption

The use license can be encrypted using identity based encryption schemes, with the authentication tokens supplying the proof of identity and the decryption key for the use license. Like the previous approach, the producer's trust of the authentication service is assured, but this approach does not necessarily require the producer to have tight control over the authentication service. However, the information is in an authentication ticket and thus a determined attacker could try a brute force attack to decipher the decryption key.

In [187], Uludag et al. proposed the use of biometrics as a means of consumer authentication.

Their approach does not require substantial changes to the underlying cryptographic system – the hash or fingerprint of the biometric sample (such as the fingerprint) can be used in a similar fashion to identity pass phrases in identity based encryption schemes.

Either approaches to identity based encryption could complicate group licensing schemes. Unless these schemes are used in a broader group encryption scheme, individual group members will require their own use licenses.

Use the DRM Controller to Encrypt the Use License Key

In this approach, the use license is encrypted with a key supplied by the consumer's DRM controller, and could make use of Trusted Computing's Trusted Platform Module (TPM) for this purpose. However, this approach has one drawback.

The DRM Controller (and TPMs) are tied to the device, and thus offers no portability beyond the device. Thus, the consumer will require a separate use license for every device they want to use. This problem could be solved using group key schemes, as suggested by Pinkas in [153] with the shared devices being a part of the group key scheme. However, Pinkas also discussed the difficulties in updating the state of group membership (revocation and addition) after the formation of a group.

A similar approach to the TPM approach is the use of a broadcast encryption scheme, such as the scheme proposed by Lotspiech et al. in [126]. Their scheme, already used for protecting DVDs (as part of CCS), requires the pre-distribution of keys to the devices. Group memberships can be formed through the formation of clusters that decide on a single key between themselves. However, their scheme does not seem to support multiple group membership for a single device.

12.3 Comparison to the to OMA DRM Approach

We have previously discussed some of the key distribution approaches while discussing existing systems in chapter 3. Being the only open standard, OMA DRM is also the only system which has a complete open description of key distribution in their framework. For this reason, we provide a brief overview of the OMA DRM approach.

All devices implementing OMA DRM are issued with Public-Private key pairs during manufacturing, and thus form the base of trust for the rights holders. The devices also contain a list of keys that can be used to verify OMA DRM 2.0 license servers (Rights Issuers or RIs). OMA DRM 2.0 devices only talk to the RIs which are responsible for the distribution of use licenses and the decryption key (AES 128 bit) for the content in a package called the Rights Object (RO). The RO is signed by the the RI, and only ROs that can be verified as legitimate can be rendered by the device.

The current approach has one major drawback: devices cannot be upgraded to accept different keys and thus the number of accessible RIs could be limited. This also means that additional features such as user authentication (currently missing in OMA DRM 2.0) cannot be implemented without a change to the trust framework.

12.4 SUMMARY

In this section, we discussed the web of trust involving the various components of the DRM system. Following our discussion, we looked at how key distribution can be achieved without breaking the web of trust. There are two types of keys involved in a DRM system: public keys to verify signatures and shared keys to decrypt data. The key distribution strategies for decryption keys require the consumer to balance their privacy requirements with their portability requirements. In the schemes we have presented, maintaining the security of the decryption key requires a trade-off between portability and complete privacy for the user.

Part III

Implementation, Analysis and Conclusions

CHAPTER 13

EXPERIENCES IN IMPLEMENTING A KERNEL-LEVEL DRM CONTROLLER

The main aim of this dissertation was to create a framework for DRM that is generic enough to accommodate any type of data. In chapter 9, we outlined a number of rights that are generic to all types of platforms, and thus can be used a generic base for any DRM system.

In this chapter we discuss our experience in implementing a DRM controller as part of the GNU-Linux kernel. The aim was to create a DRM controller that could cater to a large number of level 1 rights and then analyse the effectiveness of implementing application agnostic DRM controllers. Effectiveness of our implementation would depend on:

1. The actual possibility for an application agnostic DRM controller
2. The performance degradation imposed by such a controller
3. The range of rights supported by such a controller

We have previously discussed our implementation in [40, 41], and the majority of the actual implementation was done by the co-authors of that work, Marlon Paulse and Duncan Bennett. The main purpose of this chapter is to analyse the factors mentioned above, and we have only included some details of the implementation to put the discussion in context.

Our implementation was based on the design we described in chapter 9, and we implemented most of the technologies we described in this dissertation. The following list details the technologies that we did not implement:

1. **Negotiation to Individual Terms:** Negotiation of individual terms of a use license would require an AI agent (for the licensor at the very least) capable of such negotiations.

Since the focus of this dissertation was not on such technologies, we did not implement negotiation of individual contract terms. However, we did implement a negotiable license acquisition, including the possibility of acquiring new terms.

2. **XML based Rights Expression Language:** We wanted to maximise performance of the DRM controller, and minimise errors introduced into the system. XML processing is expensive and requires addition of libraries normally not found in the GNU-Linux kernel. For this reason, we decided not to use a XML based REL. The REL we used is much simpler, and geared for single users (as opposed to multiple users and roles).

3. **Full VDOI System Implementation:** Our file format does provide for identity verification, but does not implement the full version of VDOI System as discussed in section 10.4. Such an implementation would require the DRM controller to make network access, and thus additional complexity.

4. **DRM Controller Revocation Database:** In chapter 9, the design used a revocation database attached to the DRM controller. This addition was placed as a result of our implementation, which showed a potential flaw that could be exploited if such a database was not used.

13.1 RELATED WORK

The majority of current DRM systems are implemented at the application level, and as discussed previously, only one system – Microsoft's Rights Management Services (RMS) – features a DRM controller in the operating system kernel.

Microsoft's RMS controller does not provide transparent DRM protection. Instead, it requires applications to be "*RMS enabled*" before they may interact with DRM protected files [137]. DRM Protected files in a non-RMS enabled kernel are seen as encrypted files and no actions can be performed on them. Applications which are not RMS-enabled cannot perform simple functions such as opening a file, even if the application is running in a RMS enabled kernel [137].

Because Microsoft RMS is a proprietary system, not much has been disclosed about its design, how its DRM controller interacts with the Windows NT kernel, and the performance impact its DRM controller has in relation to a normal Windows NT kernel. Since the enforcement of rights is through the RMS enabled application, it is difficult to do comparative analysis with our own operating system DRM model.

13.2 SYSTEM DESIGN

The system design was based on the management-console/DRM-controller design discussed in chapter 9. In this chapter we recap the management-console/DRM-controller interaction, as well as introduce the file format we used and our simple REL.

13.2.1 DRM Controller – Management Daemon Interaction

Figure 13.1 gives an overall view of the process involved in accessing a DRM protected file in our system. The following steps describe the interaction between the enforcement engine and the management daemon.

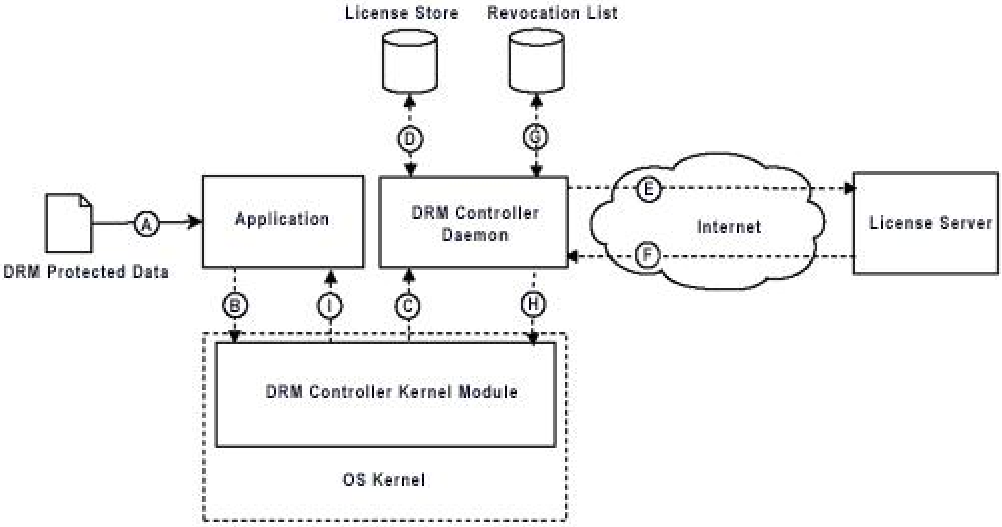


Figure 13.1: The DRM Controller Architecture and Communications

Step A: The application receives a DRM protected file as input.

Step B: The application requests access to the file. The enforcement engine intercepts this request.

Step C: The enforcement engine sends a request for the DRM use license details to the daemon.

Step D: The daemon checks the license store for a license. If a license exists, the daemon proceeds to step G. Otherwise, it proceeds to step E.

Step E: The daemon connects to a license server enabling the negotiation of a license download.

Step F: If a license is successfully negotiated, the daemon proceeds to step G. Otherwise, a message is sent to the enforcement engine to deny file access.

Step G: The validity of the license is checked against a revocation list. If a license is invalid, the daemon may return to step E to negotiate a new license.

Step H: The daemon sends the use license and the authentication ticket for the user, to the enforcement engine.

Step I: The enforcement engine performs a final check on the access request. The end-user and the request are referenced against the relevant fields in the use license. If these details are valid, the application is granted access to the requested file.

As discussed in chapter 9, the daemon also manages the authentication tickets, and the steps to acquire and manage authentication tickets is similar to steps D to G described above.

13.2.2 **File Format**

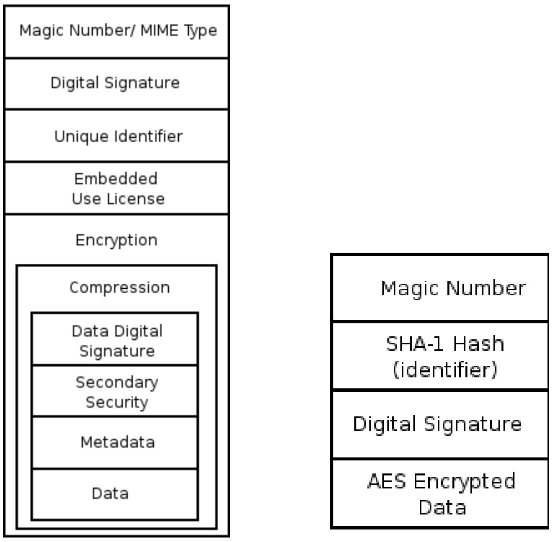


Figure 13.2: Layered approach to DRM file formats (left) and the implemented file format (right)

The file format used by our DRM controller implementation is presented on the right in figure 13.2. As discussed in chapter 11, the magic number gives the file format a unique MIME type identifier allowing the kernel module to quickly identify a DRM protected file and to ignore non-DRM protected files, and as far as we are aware, the number we have chosen is not being used by any other MIME type.

At the moment we use a SHA-1 hash of the unencrypted content as the identifier. We admit that this approach is not ideal, as there is a chance that two different objects can have the same message digest, and a more complete implementation of the VDOI system we discussed in section 10.4 is necessary in a full implementation. This approach however is better than a simple labelling scheme as it provides a primitive identity verification service, that is not provided by common identifier schemes. The protected file is encrypted using 128bit AES. The key is distributed with the license, and we used the authentication tokens as the means to encrypt the use licenses.

13.2.3 Rights Expression Language (REL)

```
License          ::= ( Permission )+ End
Permission       ::= PermissionStart Type ( Constraint )* End
Constraint       ::= ConstraintStart Type ( Argument )*
                      ( Constraint )* End
Argument         ::= ArgumentStart Type ArgumentValue End
ArgumentValue    ::= ( 1-9a-zA-Z )+
Type             ::= ( 1-9 )+
End              ::= ``;''
PermissionStart ::= ``!``
ConstraintStart ::= ''&''
ArgumentStart    ::= ``@``
```

Figure 13.3: The BNF grammer for our simplified REL

We wanted to reduce the complexities involved in our test implementation and thus used a flat file representation for the use license instead of an XML based license like ODRL or XrML. The format we used is shown in figure 13.3, and our format follows the core set of elements required for license enforcement as discussed by Guth et al. in [94]. Each license was signed and also contained the the consumer identifier. This flat file model is also compatible to the formal model we have discussed in chapter 8.

13.3 Implementation Details

13.3.1 Setup

The system implements a DRM controller responsible for enforcing rights protection on a consumer machine. As discussed earlier the aim of the implementation is to create a controller that supports multiple file formats and is transparent to the applications that access the files. Due to the wide availability of literature on GNU-Linux kernels, and its open source nature, we decided to implement the system on a GNU-Linux-based operating system with a *vanilla* 2.6.15 version GNU-Linux kernel.

The controller consists of two core modules: an operating system kernel module and a user-space daemon. The kernel module is responsible for enforcing the access control rules specified in DRM use licenses, while the daemon retrieves these use licenses from the content publisher's license servers and manages them in its local license store. When an application requests access to a DRM protected file, the daemon retrieves the DRM use license and sends it to the kernel module where the rights enforcement will occur. A more detailed description of this communication was given earlier in section 13.2.

The communication between the kernel module and the daemon is performed via a character device file in the */dev* directory. Ideally, this communication channel would need to be secure, tamper-proof and only accessible to the kernel module and the daemon. However, we were unaware of how to implement secure userspace-to-kernel communication in GNU-Linux. The management daemon does not modify any data it manages, and the DRM controller is responsible for deciding on the trustworthiness of the data it receives. Thus, the integrity of the communication is not breached by our approach.

As discussed earlier, there is a potential subversion in this approach. A rogue daemon could use a revoked license to access data, as this approach does not check revocation at the DRM controller. A revocation database at the DRM controller resolves this issue.

13.3.2 **Permissions and Constraints Implemented**

We implemented the following Level 1 permissions, which represent most of the Level 1 permissions from the ODRL 1.1 data dictionary [103].

1. **DELETE**
2. **DISPLAY**
3. **EXECUTE**
4. **MODIFY**
5. **SAVE**
6. **MOVE**

We also implemented the following two ODRL constraints, which are applicable to all of the above permissions.

1. **COUNT**
2. **DATETIME**

13.3.3 The Enforcement Engine: The Operating System Kernel Module

The kernel module consists of five components:

1. **Enforcement Component (EC),** enforces the rules specified in DRM use licenses
2. **Decision Component (DC),** decides whether an application may access a DRM protected file in a certain manner, based on the permissions and constraints specified in the DRM use license.
3. **Access Control Rules Manager (ACRM),** parses the kernel licenses (shown in figure 13.3) received from the user-space daemon into an internal data structure that allows for more efficient in-memory storage and look-ups.
4. **Data Handling Component (DHC),** is used to determine whether files being accessed are DRM protected, perform digital signature verification and to decrypt DRM files. This component is also responsible for storing the state of all open DRM protected files in the system.
5. **Communications Interface (CI),** implements the character device driver that provides the communication mechanism between the kernel module and the daemon.

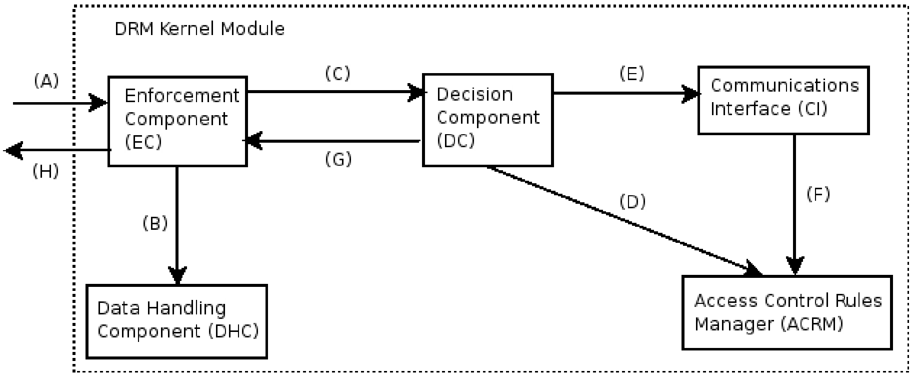

DRM Kernel Module
(A)
(H)
Enforcement Component (EC)
(C)
(G)
Decision Component (DC)
(E)
Communications Interface (CI)
(B)
(D)
(F)
Data Handling Component (DHC)
Access Control Rules Manager (ACRM)

Figure 13.4: The interaction of the various components within the kernel module

These components are shown in figure 13.4. The kernel module operates as follows:

Step A: The kernel module receives a file access request from an end-user application.

Step B: The EC uses the DHC to determine whether the file is DRM protected. If it is, the DCH verifies the digital signature embedded in the DRM protected file. If the digital signature is successfully verified, the DRM protected file is decrypted.

Step C: The EC checks with the DC whether the application can access the file in the manner it requested.

Step D: The DC requests the DRM use license in the kernel module's in-memory license cache from the ACRM.

Step E: If the DRM use license is not available in the kernel module's in-memory license cache, the DC retrieves the license using the CI.

Step F: The CI communicates with the daemon to retrieve the license from the daemon's local license store.

Step G: The license from the daemon is sent to the ACRM so that it can be parsed into a data structure that the DC can use to efficiently look-up permissions and constraints. The license is also saved in the kernel module's in-memory license cache, in case the end-user application performs another file access request in future.

Step H: The DH checks if the application may be granted access by looking-up the permissions in the license data structure constructed by the ACRM and verifies that all the license constraints are satisfied. The DH notifies the EC whether access may be granted to the application

Step I: Based on the response by the DH, the EC grants or denies the application access to the file.

It should be noted that the digital signature verification and file decryption processes only occur once: when the application opens the file. Subsequent file access requests, such as DISPLAY, do not incur this overhead. Also, for simplicity, decrypted DRM protected files are stored in the */tmp* directory. Temporary files are required because all the application's file requests are redirected from the encrypted file to the temporary, unencrypted file, and performing the operation in memory proved to be too expensive. We recognise the security risk introduced by this approach, and we discuss this in more detail in section 13.3.4 . To lower the security risk, the temporary files were given random names, and the redirection does not appear under normal process listing (such as ps). Once the application closes the file, these temporary files are deleted.

In order to enforce the DRM use license permissions listed previously, we define mappings between permission names and system call routines. These mappings are defined as follows:

1. **DELETE** : unlink

2. **DISPLAY** : read

3. **EXECUTE** : execve

4. **MODIFY** : write

5. **SAVE** : write

6. **MOVE** : rename

Therefore, whenever an application makes a request for a permission, all the kernel module has to do when access is granted is to execute the corresponding system call routine. If permission is denied, the kernel module simply needs to return from the system call routine with an error message. In order to implement this procedure, the EC replaces the I/O system calls in the GNU-Linux kernel with its own set of system calls. This means that whenever an I/O request is made, the EC's system calls are called instead of the original GNU-Linux kernel system calls, thus allowing the DRM kernel module to perform the DRM access control verification as described above. This is accomplished by replacing the addresses in the system call table that point to the original I/O system call routines with the addresses of the kernel modules I/O system calls.

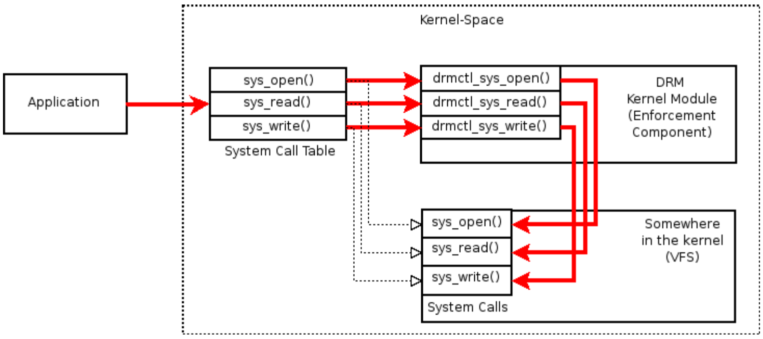


Figure 13.5: Intercepting system calls and redirecting file access requests in the kernel

Figure 13.5 illustrates this process. The dotted arrows show the normal flow of control from an application request (invoking a system call) to the original system call routines that service the request. By changing the values in the system call table, control instead passes to the kernel modules system call routines, as indicated by the bold arrows. If an end-user application is denied access to the DRM protected file, the EC simply exits its custom system call routines, returning an error code (*-EACCES*) to the end-user application. If access is granted, the EC calls the original system calls from within its custom system call routines, enabling the end-user application to carry out the action associated with the access request.

Management of a License Store

In order to retrieve DRM use licenses for the kernel module, the daemon stores licenses in a local license store. The daemon is responsible for acquisition of license (over the Internet), storing and indexing retrieved licenses and removing expired and revoked licenses.

Negotiate Licenses with a License Server

If a license is not available in the license store, the daemon initiates a negotiation for a new license with a content distributor's license server. The daemon activates a user interface, and a child process is started to await a response from the user interface. This frees the main daemon process to continue communicating with the kernel. Once the negotiation is complete, a message is sent to the child process. If the message contains a license, the license is added to the license store. The user can then try to attempt to access the file once more.

Management of authentication tickets

We use authentication tickets for user authentication. The daemon is responsible for acquiring authentication tickets (over the Internet) and the storing and indexing of authentication tickets, removing expired authentication tickets.

Communications with the Kernel

When requested by the kernel, the daemon finds the appropriate use license and associated authentication ticket and then sends them to the kernel.

13.3.4 Motivation for our approach

The use of a temporary decryption file is not ideal. Firstly, it requires that the entire DRM protected file be decrypted before the end-user application may access it, incurring a big performance penalty on large files. A better approach would be to implement on-the-fly decryption where the file decryption and application access may occur simultaneously. Also, using a temporary file complicated the updating of DRM protected files when an end-user application has MODIFY permissions. With our approach, if an update should occur, the kernel module would need to re-encrypt the temporary file and replace the original DRM protected file with the newly encrypted file. Not only does this incur further performance losses, but ensuring consistency between the original DRM protected file and the newly created protected file in multi- threaded or parallel processing environments would be very difficult.

Lastly, there may also not be enough space on the computer on which the DRM protected file is being accessed. This is especially a problem when the DRM protected file is large. However, despite these drawbacks, we still opted for this approach, as it is simple to implement and will provide a reasonable indication of the performance losses involved in decrypting DRM protected files in the kernel. Furthermore, using temporary files also gets around memory limitations that are experienced if the unencrypted file is stored in memory.

13.4 EXPERIMENTAL EVALUATION

In this section, we describe the experiment that was conducted to determine the performance cost imposed on the system by DRM controller implementation, and present and analyse the results thereof.

13.4.1 **Experiment Setup**

The following two system calls were used during this experiment: read() and rename(). These system calls correspond to the DRM use license permissions, DISPLAY and MOVE, respectively. They were chosen for this experiment because they represent the two types of access operations which can be performed by the file system of an operating system on the data stored on the disk. The read() system call performs sequential accesses on each byte of the data, whereas the rename() system call only operates on the entire chunk of data - a file - as a whole.

The experiment involved the following three tests. First, we measured the duration of the read() and rename() system calls in a standard Linux kernel when accessing non-DRM protected files of various sizes. Then, we determined the system call overhead of the two system calls when the DRM controller kernel module was enabled. As in the first test, all the files used in this test were non-DRM protected. Finally, we repeated the second test, but this time, used files which were DRM protected.

Two sets of files were used in our experiment. The first set contained regular non-DRM protected files, while the second set contained DRM-protected copies of the files in the first set (encrypted with AES in ECB mode with key length of 128 bits). Each set consisted of six files of various types and sizes as shown in table 13.1. We believe that these files represent a good selection of real world digital works that could be protected using DRM.

For each test-run, the system calls were invoked 100 times per file. The duration of the system call was then determined by taking the average of the 100 measurements. These results are presented in tables 13.2 and 13.3.

File type	File size
Text file	98 B
PDF document	38.517 KB
JPEG image	1757.283 KB
MP3 audio file	4.670 MB
MPG video file	23.848 MB
GZIP tarball	102.401 MB

Table 13.1: The types and sizes of the files used during the performance evaluation experiment.

13.4.2 Test Environment

The experiment was conducted on an Intel Celeron computer with a CPU clock speed of 1.7GHz, 512 MB of RAM and a 40Gb 7200RPM PATA hard drive. In most respects, these hardware specifications can be considered as representative of an average end-user machine. The computer was loaded with a typical desktop installation of Linux running a *vanilla* 2.6.15 version kernel.

13.4.3 Results

Tables 13.2 and 13.3 show the results of the three tests for the read() and rename() system calls respectively. In each table, the performance costs incurred by the DRM controller are shown.

13.4.4 Analysis

There are three areas in the DRM controller which contribute to a performance overhead:

1. Intercepting the access request and detecting DRM protected data.
2. Communicating with the daemon.
3. Parsing the use license and enforcing the rights specified in the license.

The cost of intercepting the access request and detecting whether the request applies to DRM protected data occurs regardless whether the data is DRM protected or not. This is the stage where the DRM controller distinguishes between access requests that need DRM protection and access requests that do not. Assuming that the daemon communications and rights enforcement cost is negligible, this cost will be the best-case performance cost that will be incurred by the DRM controller.

File size (B)	Std kernel Non-DRM data access time (μs)	Std kernel + DRM ctl. Non-DRM data access time (μs)	Std kernel + DRM ctl. DRM data access time (μs)	Non-DRM data overhead (%)	DRM data overhead (%)
98	23.476	30.758	55275.004	31.019	235353.246
39441	71.409	84.297	55303.169	18.048	77345.657
775458	835.512	905.349	83927.375	8.358	9945.023
4896677	4840.858	5041.487	236563.284	4.144	4786.805
25006182	24695.321	25528.383	1119149.073	3.373	4431.826
107375252	104913.480	205856.577	7664316.603	96.216	7205.369

Table 13.2: Comparing the duration of the read() system call when handling DRM protected and non-DRM protected data on a DRM-enabled and DRM-free system.

File size (B)	Std kernel Non-DRM data access time (μs)	Std kernel + DRM ctl. Non-DRM data access time (μs)	Std kernel + DRM ctl. DRM data access time (μs)	Non-DRM data overhead (%)	DRM data overhead (%)
98	49.695	62.341	73.251	25.447	47.401
39441	48.963	58.061	69.188	18.581	41.307
775458	50.041	60.247	70.498	20.395	40.880
4896677	50.434	60.779	71.460	20.512	41.690
25006182	49.641	67.955	71.009	36.893	43.045
107375252	50.667	62.769	76.860	23.885	51.696

Table 13.3: Comparing the duration of the rename() system call when handling DRM protected and non-DRM protected data on a DRM-enabled and DRM-free system.

Table 13.2 shows the result of the three tests for the read() system call. We see that the performance costs imposed by the DRM controller when accessing non-DRM protected information there is an increase in the overhead by 31% for a 98B file, decreasing until it reaches a near 3.4% performance overhead cost for a 23.8MB file. This suggests that the overhead from intercepting access requests and detecting DRM data is so small compared to the overhead of communicating with the daemon and enforcing the license rights, that it is negligible. The large jump in the overhead for large files however, does suggest that there might be other factors that have influence, other than the daemon-kernel communication.

Looking at table 13.3, we see the access times when non-DRM protected content in a standard kernel with the DRM controller enabled remain almost the same. This is as expected, as the rename() system call performs only one access on the data, regardless on the size of the data. We attribute the discrepancies in the access times to the arrangement of files on disc and possible variances in system load.

When accessing DRM-protected content on a DRM-enabled system using the read() system call, we observed a similar trend to the non-DRM protected content. Although the access times increase as the file sizes increase, the performance overhead decreases. Initially, we find a 235353% increase in performance cost. This high cost increase is due to the large overhead involved when communicating with the daemon, parsing the license, and traversing the in-memory license structure to enforce digital rights. As more read requests are performed this cost becomes less noticeable, and drops to approximately 4431% for a 23.8MB file.

Table 13.3 shows the performance results for rights enforcement where decryption is not required. In this case, the overheads are introduced in the daemon-kernel communication and the interpretation of the licenses. As can be expected, the overheads involved are almost constant, and have no user observable time performance effect.

In both the read() and rename() cases, the overhead due to the daemon communications and the rights enforcement far outweigh the cost of intercepting access requests. This, of course, raises questions regarding the infrastructure of the DRM controller. If a file is found to be DRM protected by the kernel, it must start an expensive communication with the daemon. This costly process might best be avoided by introducing a hardware implementation of a license store, instead of file-system based one which is managed by a user-space application. If the license store was managed by kernel, the need to request licenses from the daemon would be removed. However, there might still be a cost, as the the kernel still needs to establish communication with the daemon to allow it to retrieve licenses from remote license servers.

Even though there are high costs incurred, the net effect on time is minimal and will not be noticeable by the end user for single user machines. However, this will not hold for multi-user or high performance machines and thus improvements are necessary before it can be considered for deployment.

We also need to consider the frequency at which accesses to DRM protected content are made. If access to protected content is relatively infrequent, then the huge cost might still be judged to be negligible, since the performance cost is still sufficiently small that a human user would not notice it. However such a judgement cannot be reached without further investigation of file access patterns, such as those created by web servers, or a multi-user system where the majority of files are protected.

13.5 Analysis of our Approach

The current system has been built in order to test the feasibility of an operating system level DRM controller. We have tried to make it as complete as possible, but our implementation is not a complete solution for DRM. In this section, we discuss how well our approach works in achieving its goals, as well as detailing some issues that we feel need to be addressed for a more complete system.

13.5.1 **Application Level Transparency**

Our design allows for any application to access any DRM protected file, and application behaviour is not affected, other than the ability to modify a file. We have tested on a wide variety of common GNU-Linux applications, including various PDF readers (examples: Ghostview, xpdf, Gnome PDF Reader), different media players (examples: mplayer, xmms, mpg123) and text editors (examples: vim, gvim, kwrite, nano).

13.5.2 **Wide range of rights**

We have implemented most of the rights that can be enforced at an operating system level, and we feel that there certain rights (like printing) that can only be enforced at the application level. As discussed in chapter 9, level 2 rights require application level DRM controllers.

13.5.3 **Performance**

In [161], Raskin discusses how humans can easily pick up changes in application behaviour once they become used to the applications. He gives a small boundary of a couple of seconds, before changes like longer loading times, or sluggish operation will become noticable to the average user. In [39], Arnab and Nunez, determined experimentally that users are willing to wait, on average, 5.5 seconds longer if they are aware that there is a security operation that needs to be performed before they are able to access the data. We have therefore chosen to use this value as the threshold for performance degradation.

As detailed in table 13.2, for files up to the size of 23.8 MB, the performance degradation was quite low, with the largest file having a degradation of 1.1 seconds for a DRM enabled file. However, for a large compressed data file with a size over 102.4 MB, the performance degradation is quite significant at 7.7 seconds for a DRM enabled file. However, the performance degradation for non DRM enabled files remained negligible.

Thus, while our system is quite suitable for small data files like music and PDF documents, it does not meet the performance requirements for larger files, like movies.

13.5.4 Comprehensive Protection

Some rights may allow more actions than they intend to. For example, a license could allow the right to “read” a certain file, but not execute. But because the file can be read, it would be easy to copy the contents of the file onto another file, which can then be executed. Thus, there needs to be a closer co-operation with file and memory operations to thwart such actions.

13.5.5 Modification of Protected Files

Allowing and disallowing read only functionalities is easily accommodated, but the major problems occur when trying to cater for modification of protected data. To cater for modification of data, functionality to re-identify and repackage the data needs to be provided at the kernel level. If modification of the the protected file is frequent (capturing event data or even traditional office file), these operations will severely slow down saving of an application. In our solution, we did not provide these functionalities, and only provided for the outright prevention of modification of data. In our opinion, some level of application support is required before modification of data is seamless.

13.5.6 Correct Identification of Accesses

More complex applications may break a single user level access into several smaller accesses. For example, playing a music file may require multiple read attempts although only a single play permission is exercised. The DRM controller must be able to correctly identify the purpose of these calls. If it does not, a consumer’s access rights may expire prematurely, Consider the example of a “play” permission limited by a count constraint. The count must be decremented only when the media starts to play and not for each read access.

13.5.7 **Stream Encryption**

Our solution currently does not handle stream encryption even though, in theory, it would seem to be a faster and more secure solution. However, most applications tend to load files in their entirety instead of a portion of a file due to a variety of reasons including compression techniques and metadata storage. For this reason, stream ciphers would be impractical without application level support, and would make sense for only certain file types.

13.5.8 **Compensating for Application Behaviour**

Some applications behave unexpectedly. For instance, multiple access attempts may be made before an application determines that a file cannot be accessed. The daemon module must compensate and distinguish genuine requests from repeat requests. This is to avoid initiating multiple license negotiations for the same asset.

13.5.9 **Implications for hardware based DRM systems**

We think that there would have been a better performance from our system if the license and authentication tickets were stored in hardware. However, such a store would have restricted memory, and this could affect the overall system. Memory is also the main factor to consider for hardware DRM controllers. As we have discussed, stream based encryption is unpractical for the general case, and this implies that the hardware DRM controller will need to store the decrypted DRM file somewhere while it is being used. Making use of a dedicated memory store for the controller would be the most secure approach, but this would limit the number and size of secure data files that can be accessed simultaneously to available memory. Modification of files would remain a problem, although the process of repackaging should be faster. However, application level support will still be necessary to make the process seamless.

We think that hardware based DRM will ultimately offer the advantages offered by kernel level DRM controllers, with better performance. Furthermore, there should be no reason why open source software cannot make use of the hardware based DRM to provide persistent access control, as long as the relevant drivers are available.

13.6 SUMMARY

In this section, we discussed our experiences in implementing a DRM controller as part of the GNU-Linux kernel, based on the design we discussed in chapter 9. We proved that it is possible to create an application agnostic DRM controller, which can enforce most level 1 rights without any changes to the applications. Thus, it is possible to create a DRM system, where users can continue to use their favourite applications.

However, there is substantial performance degradation in our approach. For small files (such as music, most documents), this degradation could be considered as not noticeable by the user. For large files, users can expect to wait for longer than 6 seconds before they can access their files, and is therefore not acceptable. For this reason, there is a need to address the performance issues associated with kernel level DRM controllers, before they can become mainstream.

CHAPTER 14

CONCLUSIONS AND FUTURE WORK

Digital rights management aims to provide persistent access control, and even though DRM can be applied for different functions, the core requirements remain the same. Despite this, most DRM systems developed so far have focused on particular functionalities, usually delivering media to the general public. Furthermore, when we analysed existing systems (chapter 3) against a set of common requirements that we have derived from a wide range of sources (chapter 2) we showed that none of the systems satisfy all the requirements. In general, the main problems with existing approaches to DRM are:

1. DRM systems do not have adequate user authentication processes, and usually rely on locking down content to devices. This prevents portability of protected content, even between devices owned by the same user.
2. Most systems do not have the means to revoke or change the terms and conditions of particular protected content.
3. Vendors of DRM systems do not advertise, and possibly do not understand, the legal and social requirements of their systems.

There has been a lot of focus on achieving interoperability between different DRM systems from both academia and industry. Unfortunately, the best form of interoperability – full format interoperability – is also the hardest to achieve, as standardisation of the components required for full format interoperability is difficult and time consuming process. In this regard, Jamkhedkar and Heileman proposed a layered DRM architecture, motivated by the success of the OSI 7-layer network model. They did not however detail the inner workings of each of these layers.

In this dissertation, we aimed to create a general framework for DRM – a common framework that can provide persistent access control regardless of the type of data, or function of the data.

Our framework is complementary to Jamkhedkar and Heileman's layered model; and addresses a gaps in their architecture – notably the lack of a legal framework and the formal definition of DRM as an access control model – and also provides frameworks for the operation of the layers.

The first gap in the Jamkhedkar and Heileman layered architecture is the *lack of a legal framework* that encompasses the DRM architecture. All systems should operate in a manner that is consistent with legal practice and norms. DRM aims to control the access to data, especially data protected under copyright or trade secrets. However, with the provision of control, there is also potential for abuse; and thus there is a need to provide a firm legal grounding in which DRM can operate. While there have been a number of contributions on the legal position of DRM systems – most of which have focused on the relationship between DRM and copyright law – there has been little effort in trying to accommodate DRM systems under present legal systems. Our legal framework in chapter 6 addresses this gap.

In our legal framework, we discussed that DRM systems should be seen as systems that allow for the formation and enforcement of licensing agreements (a contractual process); and not as systems that enforce copyright law. Contracts are usually concluded after the parties negotiate the terms of the contract. In chapter 7, we presented two comprehensive negotiation protocols, complete with modelling and petri-net analysis to ensure the robustness of our protocols.

The second gap in the Jamkhedkar and Heileman layered architecture, as the authors discussed in a subsequent contribution, is the *lack of a formal description for DRM systems*. Even though DRM is seen by many as an access control mechanism, there has been no formal description of DRM as an access control mechanism. Even rights expression languages, which form one of the crucial components of DRM systems, have no formal base; although there have been some contributions towards formalising the languages. In chapter 8, we address this gap, with what we consider to be a comprehensive formal model for DRM. LiREL is a rights expression language that can express licensing agreements and at the same time provide the access control specifications required by DRM systems. Furthermore, we use LiREL to formalise the interpretation and enforcement of access control policies represented with LiREL.

Enforcement of access control is a two part process: authenticating the parties and authorisation of the request. In chapter 9 we created an authorisation framework for DRM, based on our formal model. The *authorisation framework* depends on the *authentication framework*, described in chapter 10. A key foundation of our framework is the separation of authentication and authorisation as separate functions, and thus allowing for a wider variety of authentication and authorisation processes to interoperate.

As part of the authentication framework, we also addressed a gap on the Internet. While there are a number of authentication mechanisms for users, authentication mechanisms for data do not really exist. Currently, data identity efforts have revolved around the provision of persistent identifiers, but none of the current systems provide the means to verify the association between the identifier and the data. For DRM systems to correctly enforce access control, a verification service is important. In this regard, we outlined the Verifiable Digital Object Identity (VDOI) system, which addresses this problem.

To provide completeness, we also discussed a layered approach to file formats, and discussed the trust relationships between the various parties in a DRM system. We also discussed an implementation of a general DRM controller as a kernel module in the GNU-Linux operating system. Although the performance degradation is very high, our prototype does show that it is possible to enable DRM enforcement without requiring modifications to applications that use the protected data. However, to be viable, the approach needs to provide efficient means to decrypt and store protected data files, the main cause for the performance degradation.

In chapter 2, we defined 27 requirements for DRM systems. To conclude this dissertation, in the remainder of this chapter, we present a requirement analysis of our framework, to show how our framework addresses the requirements we identified previously, before we discuss some possible future work.

14.1 Requirement Analysis of our General Framework for DRM

In chapter 2, we identified twenty seven requirements for a general DRM system. These requirements were categorised into three types: core requirements for access control, usability requirements and legal and social requirements. In this section, we will revisit these requirements and examine how well our framework addresses these requirements.

14.1.1 Core Requirements

Requirement 1: Provide Persistent Protection: In chapter 8 we formalised the access control decision making, and then discussed a framework for making such decisions in chapter 9

Requirement 2: Represent User Identity: Our formal model in chapter 8 discussed how user identity should be represented in use licenses.

Requirement 3: Support multiple User Authentication Protocols: Our model urges the inclusion of the authentication protocol as part of the representation. In chapter 10 we discuss how various user authentication protocols can be handled by the DRM controller through the use of authentication tickets.

Requirement 4: Represent and Authenticate Resource Identity: Our model in chapter 8 discussed the inclusion of resource identifiers, and in section 10.4, we presented VDOI, a verifiable resource identity system.

Requirement 5: Represent and Authenticate Device Identity: In chapter 10, we discussed current means for providing device identity. We could not provide any new means that can improve on the schemes that are already available. In LiREL, devices are part of the constraints that can be placed on a license, permission or obligation.

Requirement 6: Represent and Authenticate User Groups: Our framework provides two means of defining user groups. Firstly, as discussed in chapter 10, user groups can be provided as part of the user authentication service. Thus, the authentication service is responsible for defining and maintaining group membership, and can also provide the DRM system with role based credentials. The second way, as discussed in chapter 8, is through the representation of the use license. LiREL provides means to create associations and relationships between licensees and third parties. There was no need to do the same for licensors.

Requirement 7: Represent and Authenticate User Roles: As discussed in the preceeding requirement, and in chapter 10, our framework provides for the use of roles in DRM systems through the user authentication services used by the DRM system.

Requirement 8: Represent and Authenticate Resource Groups: There are two means of identifying and authenticating resource groups. Firstly, using wild card support provided in the VDOI system, it is possible to identify and authenticate multiple resources, acting as a group. And, although LiREL does not directly provide logical grouping of resources, it does allow for the identification of multiple devices covered by the use license, and hence provides support for grouping resources.

Requirement 9: Represent and Authenticate Device Groups: Devices are constraints and LiREL allows for the definition of multiple constraints for a particular license, permission or obligation. Furthermore, individaul constraint terms can be defined and the semantics of these terms can be decided during such a definition. Thus, device groups can be handled in LiREL.

Requirement 10: Represent the Authorisation (Use License): In chapter 8, we introduced LiREL, complete with a formal definition for the REL.

Requirement 11: Authenticate the Use License: A use license is also a digital object, and thus the VDOI system can also be used to identify and authenticate use licenses.

Requirement 12: Support User Duties: LiREL supports the expression of obligations for all parties in a licensing agreement. Furthermore, in chapter 10, we discussed how credentials can be used to prove the fulfilment of some types of obligations.

Requirement 13: Revocation of Rights: We motivated the use of use license revocation databases to control the revocation of rights. In this manner, individual licenses can be revoked and not the device of the consumer.

Requirement 14: Update of Rights: In our framework, use licenses can be updated by issuing new licenses and the revocation of the old license. This follows established practice in contracting.

14.1.2 **Usability Requirements**

Requirement 15: Time Shifting: Time is a constraint, and thus can be expressed in LiREL.

Requirement 16: Format Shifting: Our framework is format agnostic, and thus can support multiple formats. However, as we discussed with regards to implementation in chapter 13, updating and creating protected data requires support from other services to create and package the data. Our framework details such services in chapter 11, but this approach is not suitable for consumers.

Requirement 17: Space Shifting: Because there is a separation of the data and the use license, our framework supports space shifting, and requires the licenses to control the restrictions regarding devices.

Requirement 18: Platform Shifting: Because there is a separation of the data and the use license, our framework supports platform shifting, and requires the licenses to control the restrictions regarding devices.

Requirement 19: Integration with Existing Applications: In chapter 9, we identified two levels of access control. Level 1 access control can be implemented without impacting on the applications, but Level 2 access control requires application support. Our implementation exercise in chapter 13 demonstrated application agnostic enforcement.

Requirement 20: Fine Grained and Flexible Access Control Specification: LiREL provides for a flexible and fine grained specification of access control rights.

Requirement 21: Tracking and Monitoring: Tracking and monitoring is primarily achieved in conjunction with the user management system. Our flexible ticket solution can be configured to provide virtually continuous tracking and monitoring as well, or to provide privacy and anonymity, or somewhere in between.

Requirement 22: Offline Usage: Offline usage in our framework is controlled through the user management system, which can provide virtually perpetual offline usage to continuous online usage.

14.1.3 Legal and Social Requirements

Requirement 23: A Legal Framework for DRM: In chapter 6, we presented a comprehensive legal framework for our DRM framework, and advised on the issues that need to be addressed to achieve such a goal.

Requirement 24: Transparency: Transparency depends on the actual implementation of our framework. In our legal framework, presented in chapter 6, we discussed the transparency requirements for DRM systems.

Requirement 25: Privacy and Anonymity: As discussed earlier, our user management framework discussed in chapter 9, allows our framework to provide privacy and anonymity. Furthermore, LiREL supports the expression of ticket based licenses (licenses without specifying licensees) which provides for anonymous licensing.

Requirement 26: Do Not Alter Platform Functionality and Performance: This is dependent on the implementation of our framework. As we discussed in chapter 13, functionality of the platform is not affected, but there is a significant performance degradation introduced by DRM.

14.2 FUTURE WORK

In this dissertation, we presented a distribution independent, general framework for DRM. Therefore, this framework addresses half of the architectures identified by Park et al. in [149]. The next step will be to extend this framework to cater for the remaining architectures – distribution dependent DRM systems. While distribution dependent architectures do not provide true persistent access control, they are increasingly used to deliver content through streaming media. For this reason, we think it is necessary to consider distribution dependent DRM architectures as part of a complete DRM framework.

In chapter 8, we discussed a formal model for DRM, and presented it as a new access control model. Thus, we have positioned DRM as another access control model, together with existing access control models. In the global set of access control models, there has to be some degree of overlap between the various access control models, and there has been some work in finding the relationships between DAC, MAC and RBAC and how they fit in with each other. Therefore, the next step in the formalisation of DRM should focus on harmonising the DRM model with other access control models, and define the relationship between DRM, DAC, MAC and RBAC.

In chapter 7, we detailed negotiation protocols for DRM systems, and LiREL discussed in chapter 8, defined a language to express negotiation proposals. These constitute half of the requirements for electronic negotiations, and the remaining requirements, the development of an AI agent to conduct negotiations and a language to govern the operation of such an agent, still need to be developed. However, it should be possible to adapt existing agent technologies that focus on multi-issue negotiations to the requirements for DRM.

14.3 CONCLUSIONS

In the information age, economies run on information. For this reason, the value of information is high, and there is a need for owners of information to control how their information is used. Private individuals have a vested interest in controlling access to information they consider private. Likewise, enterprises have a vested interest in controlling access to information generated, used or owned by the enterprise.

DRM provides persistent access control, and thus provides the means to control access to protected data over an extended period of time. Furthermore, as the enforcement of licensing terms and conditions, it is possible to place DRM within a sound legal grounding, minimising abuse of the controls that could be placed. Thus, it is an ideal technology for meeting the requirements of both private individuals and enterprises.

For DRM to be effective, there is a need for a general, standardised solution, that works across multiple devices and is trusted by both the licensors and licensees of the information. In this dissertation, we have presented a framework to achieve these goals.

Part IV

Appendices

APPENDIX A

ABBREVIATIONS

CA: Certificate Authority

DAC: Discretionary Access Control

DMP: Digital Media Project

DOI: Digital Object Identifier

DRM: Digital Rights Management

ECD: European Copyright Directive

LiREL: Licensing Rights Expression Language

MAC: Mandatory Access Control

RBAC: Role-Based Access Control

REL: Rights Expression Language

SAML: Security Assertion Markup Language

SSL: Secure Socket Layer

UMTS: Universal Mobile Transmission System

VDOI: Verifiable Digital Object Identity (System)

XACML: eXtensible Access Control Markup Language

Appendix B

XML Schemas: LiREL

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
Validated using XML Spy 2004 - 2007-04-13-09h56 SAST
-->
<!-- Naming Convention
  - All type names start with capital letters
  - 2 or more words are concattanated to form one word with no
    spaces, the first letter of each word is capitalised, except
    as detailed below
  - all element names start with small letters
  - all abstract elements start with "abstract" e.g. abstractAction
  - all abstract elements are declared after the type declaration.
  - all type names end with the word "Type"
-->
<xs:schema
  targetNamespace="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
  elementFormDefault="qualified" attributeFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel">

   <!--
    Import support for digital signatures and XML encryption. Note
    the schema location refers to local versions of the schema.
  -->

   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
     schemaLocation="./dep/xmldsig-core-schema.xsd"/>

   <xs:import namespace="http://www.w3.org/2001/04/xmlenc#"
     schemaLocation="./dep/xenc-schema.xsd"/>
```

```
<xs:element name="license" type="lirel:LicenseType">
   <xs:annotation>
      <xs:documentation>
         The root element of LiREL is the license
      </xs:documentation>
   </xs:annotation>
</xs:element>
<!--
  Defintion of core types and their associated abstract
  definitions for LiREL. Note that we haven’t defined the types
  for the individual attributes and elements.
-->

<xs:complexType name="ConstraintType">
   <xs:sequence>
      <xs:element name="value" minOccurs="0"/>
      <xs:element name="lOperand" minOccurs="0"/>
      <xs:element name="operator" minOccurs="0"/>
      <xs:element name="rOperand" minOccurs="0"/>
   </xs:sequence>
   <xs:attribute name="negotiable" use="optional"
     default="true"/>
</xs:complexType>

<xs:element name="abstractConstraint"
  type="lirel:ConstraintType" abstract="true"/>

<xs:complexType name="ObligationType">
   <xs:sequence>
      <xs:element name="detail"/>
      <xs:element ref="lirel:abstractConstraint" minOccurs="0"
        maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="negotiable" use="optional"
     default="true"/>
</xs:complexType>

<xs:element name="abstractObligation" type="lirel:ObligationType"
  abstract="true"/>
<xs:complexType name="PartyType">
   <xs:sequence>
      <xs:element name="identifier"/>
      <xs:element ref="lirel:abstractObligation" minOccurs="0"
        maxOccurs="unbounded"/>
      <xs:element ref="lirel:abstractConstraint" minOccurs="0"
        maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="LicenseeThirdPartyType">
   <xs:complexContent>
      <xs:extension base="lirel:PartyType">
         <xs:attribute name="negotiable" use="optional"
   default="true"/>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>

<xs:complexType name="LicenseeThirdPartyGroupType">
   <xs:sequence>
      <xs:element name="party"
type="lirel:LicenseeThirdPartyType"
maxOccurs="unbounded"/>
      <xs:element ref="lirel:abstractObligation" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="PermissionType">
   <xs:sequence>
      <xs:element ref="lirel:abstractConstraint" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element ref="lirel:abstractObligation" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="negotiable" use="optional"
     default="true"/>
</xs:complexType>

<xs:element name="abstractPermission"
  type="lirel:PermissionType" abstract="true"/>

<xs:complexType name="PermissionGroupType">
   <xs:sequence>
      <xs:element ref="lirel:abstractPermission"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
<xs:complexType name="ResourceType">
```

```
   <xs:sequence>
      <xs:element name="identifier"/>
   </xs:sequence>
   <xs:attribute name="negotiable" use="optional"
     default="true"/>
</xs:complexType>

<xs:complexType name="LicenseDetailsType">
   <xs:sequence>
      <xs:element name="licensor" type="lirel:PartyType"
maxOccurs="unbounded"/>
      <xs:element name="licenseeGroup"
type="lirel:LicenseeThirdPartyGroupType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="thirdPartyGroup"
type="lirel:LicenseeThirdPartyGroupType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="resource" type="lirel:ResourceType"
minOccurs="0" maxOccurs="unbounded"/>
      <xs:sequence>
         <xs:element name="permissionGroup"
   type="lirel:PermissionGroupType"
   maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:element ref="lirel:abstractConstraint" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>

<xs:complexType name="DelegationType">
   <xs:complexContent>
      <xs:extension base="lirel:PermissionType">
         <xs:sequence>
            <xs:element name="delegatedLicense"
      type="lirel:LicenseDetailsType"/>
         </xs:sequence>
      </xs:extension>
   </xs:complexContent>
</xs:complexType>

<xs:element name="delegation" type="lirel:DelegationType"
  substitutionGroup="lirel:abstractPermission"/>
<xs:complexType name="LicenseType">
   <xs:complexContent>
      <xs:extension base="lirel:LicenseDetailsType">
```

```
            <xs:sequence>
               <xs:element name="identifier"/>
               <xs:element name="dateOfIssue"/>
               <xs:element ref="ds:Signature" minOccurs="0"
         maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="type" use="required">
               <xs:simpleType>
                  <xs:restriction base="xs:NMTOKEN">
                     <xs:enumeration value="Request"/>
                     <xs:enumeration value="Offer"/>
                     <xs:enumeration value="CounterOffer"/>
                     <xs:enumeration value="Tender"/>
                     <xs:enumeration value="Agreement"/>
                     <xs:enumeration value="Reject"/>
                     <xs:enumeration value="Accept"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:attribute>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

APPENDIX C

XML SCHEMAS: A SAMPLE PERMISSION SET FOR LIREL

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Naming Convention
   - All type names start with capital letters
   - 2 or more words are concattanated to form one word with no
     spaces, the first letter is capitalised
   - all element names start with small letters
   - all abstract elements start with "abstract" e.g. abstractAction
   - all abstract elements are declared after the type declaration.
   - all type names end with the word "Type"
-->
<xs:schema
  targetNamespace="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
  elementFormDefault="qualified" attributeFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
  xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1">
   <xs:annotation>
      <xs:documentation>
        This schema provides a permission similar to the UNIX
permission set, with the addition of constraints.
      </xs:documentation>
   </xs:annotation>

   <xs:import namespace="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   schemaLocation="lirel.xsd"/>

   <!-- Permissions. Implementation of the standard UNIX file
     system permission set -->

   <xs:element name="execute" type="lirel:PermissionType"
```

```
      substitutionGroup="lirel:abstractPermission"/>

   <xs:element name="write" type="lirel:PermissionType"
      substitutionGroup="lirel:abstractPermission"/>

   <xs:element name="read" type="lirel:PermissionType"
      substitutionGroup="lirel:abstractPermission"/>

   <!-- Definition of obligations -->
   <xs:element name="prePay" type="lirel:ObligationType"
      substitutionGroup="lirel:abstractObligation"/>

   <xs:element name="postPay" type="lirel:ObligationType"
      substitutionGroup="lirel:abstractObligation"/>

   <xs:element name="qos" type="lirel:ObligationType"
      substitutionGroup="lirel:abstractObligation"/>

   <!-- Definition of Constraints -->
   <xs:element name="count" type="lirel:ConstraintType"
     substitutionGroup="lirel:abstractConstraint"/>

   <xs:element name="range"
     substitutionGroup="lirel:abstractConstraint">
      <xs:complexType>
         <xs:complexContent>
            <xs:extension base="lirel:ConstraintType">
               <xs:sequence>
                  <xs:element name="min" type="xs:decimal"
  minOccurs="0"/>
                  <xs:element name="max" type="xs:decimal"
  minOccurs="0"/>
               </xs:sequence>
            </xs:extension>
         </xs:complexContent>
      </xs:complexType>
   </xs:element>

   <xs:element name="money"
     substitutionGroup="lirel:abstractConstraint">
      <xs:complexType>
         <xs:complexContent>
            <xs:extension base="lirel:ConstraintType">
               <xs:sequence>
                  <xs:element name="currency" type="xs:string"
```

```
    minOccurs="0"/>
               </xs:sequence>
            </xs:extension>
         </xs:complexContent>
      </xs:complexType>
   </xs:element>

   <xs:element name="validUntil" type="lirel:ConstraintType"
     substitutionGroup="lirel:abstractConstraint"/>

   <xs:element name="validFrom" type="lirel:ConstraintType"
     substitutionGroup="lirel:abstractConstraint"/>

   <xs:element name="jurisdiction" type="lirel:ConstraintType"
     substitutionGroup="lirel:abstractConstraint"/>
</xs:schema>
```

APPENDIX D

NEGOTIATION USING LIREL

D.1 THE INITIAL ENQUIRY

John would like a license to access a protected eBook. John would like the license to offer read and write permissions.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
John would like a license for a protected ebook with ID
  vdoi://123.456/2/23/23.
The licensor of the ebook has a jabber enabled ID:
  jabber://licensor@exampleLiREL.net
John makes use of a jabber enabled ID:
  jabber://john@exampleLiREL.com
-->

<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Request">
   <!-- Licensor -->
   <lirel:licensor>
      <lirel:identifier>
         jabber://licensor@exampleLiREL.net</lirel:identifier>
   </lirel:licensor>
   <!-- Licensee -->
   <lirel:licenseeGroup>
      <lirel:party>
```

```
         <lirel:identifier>
            jabber://john@exampleLiREL.com</lirel:identifier>
      </lirel:party>
   </lirel:licenseeGroup>
   <!-- Resources under discussion-->
   <lirel:resource>
      <lirel:identifier>vdoi://123.456/2/23/23.</lirel:identifier>
   </lirel:resource>
   <!-- Contract Terms-->
   <lirel:permissionGroup>
      <dd:read/>
      <dd:write/>
   </lirel:permissionGroup>
   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
      <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>
   <dd:validUntil lirel:negotiable="true">
      <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>
   <!-- Request Identifier -->
   <lirel:identifier>Request1</lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.2 THE FIRST OFFER

The licensor of the eBook responds to John offering only read permissions for 10 Euros.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
The licensor resonds to the request with an offer worth 10 Euros,
and without a write permission-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Offer">

   <!-- Licensor -->
   <lirel:licensor>
```

```
    <lirel:identifier>
      jabber://licensor@exampleLiREL.net
    </lirel:identifier>
    <dd:qos>
      <lirel:detail>
        The license will provide access to a high resoultion
(>300 DPI) version of the eBook
      </lirel:detail>
    </dd:qos>
  </lirel:licensor>

  <!-- Licensee -->
  <lirel:licenseeGroup>
    <lirel:party>
      <lirel:identifier>
        jabber://john@exampleLiREL.com
      </lirel:identifier>
      <dd:prePay lirel:negotiable="false">
        <lirel:detail>
   The licensee has to pay before an agreement is
   concluded
</lirel:detail>
        <dd:money lirel:negotiable="false">
  <lirel:value>10</lirel:value>
  <dd:currency>EUR</dd:currency>
</dd:money>
      </dd:prePay>
    </lirel:party>
  </lirel:licenseeGroup>

  <!-- Resources under discussion-->
  <lirel:resource>
    <lirel:identifier>
      vdoi://123.456/2/23/23.
    </lirel:identifier>
  </lirel:resource>

  <!-- Contract Terms-->
  <lirel:permissionGroup>
    <dd:read/>
  </lirel:permissionGroup>

  <!-- Contract Constraints -->
  <dd:jurisdiction lirel:negotiable="false">
    <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Offer Identifier -->
   <lirel:identifier>vdoi://123.456/1/45/23</lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.3 THE COUNTER OFFER

John rejects the licensor's offer and creates a counter offer where he offers to pay an extra 10 Euros for the right to write.

D.3.1 Rejection of the Offer

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
John rejects the licensor's offer-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Reject">

   <!-- Licensor -->
   <lirel:licensor>
     <lirel:identifier>
       jabber://licensor@exampleLiREL.net
     </lirel:identifier>
     <dd:qos>
       <lirel:detail>
         The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
       </lirel:detail>
     </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
```

```
     <lirel:party>
       <lirel:identifier>
         jabber://john@exampleLiREL.com
       </lirel:identifier>
       <dd:prePay lirel:negotiable="false">
         <lirel:detail>
    The licensee has to pay before an agreement is
    concluded
 </lirel:detail>
         <dd:money lirel:negotiable="false">
   <lirel:value>10</lirel:value>
   <dd:currency>EUR</dd:currency>
 </dd:money>
       </dd:prePay>
     </lirel:party>
   </lirel:licenseeGroup>

   <!-- Resources under discussion-->
   <lirel:resource>
     <lirel:identifier>
       vdoi://123.456/2/23/23.
     </lirel:identifier>
   </lirel:resource>

   <!-- Contract Terms-->
   <lirel:permissionGroup>
     <dd:read/>
   </lirel:permissionGroup>

   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
     <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Rejection Identifier -->
   <lirel:identifier>
     vdoi://123.456/1/45/23/Reject1
   </lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.3.2 The Counter Offer

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  John creates a counter offer, offering to pay extra for the
  right.
-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/~aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/~aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/~aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/~aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="CounterOffer">

   <!-- Licensor -->
   <lirel:licensor>
     <lirel:identifier>
       jabber://licensor@exampleLiREL.net
     </lirel:identifier>
     <dd:qos>
       <lirel:detail>
         The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
       </lirel:detail>
     </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
     <lirel:party>
       <lirel:identifier>
         jabber://john@exampleLiREL.com
       </lirel:identifier>
       <dd:prePay lirel:negotiable="false">
         <lirel:detail>
    The licensee has to pay before an agreement is
    concluded
 </lirel:detail>
         <dd:money lirel:negotiable="false">
   <lirel:value>10</lirel:value>
   <dd:currency>EUR</dd:currency>
 </dd:money>
       </dd:prePay>
     </lirel:party>
   </lirel:licenseeGroup>

   <!-- Resources under discussion-->
   <lirel:resource>
     <lirel:identifier>
       vdoi://123.456/2/23/23.
     </lirel:identifier>
   </lirel:resource>

   <!-- Contract Terms-->
   <lirel:permissionGroup>
     <dd:read/>
   </lirel:permissionGroup>

   <dd:write>
     <dd:prePay>
       <lirel:detail>
         Will pay additional amount for the right to write
       </lirel:detail>
       <dd:money>
         <lirel:value>10</lirel:value>
         <dd:currency>EUR</dd:currency>
       </dd:money>
     </dd:prePay>
   </dd:write>

   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
     <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Counter Offer Identifier -->
   <lirel:identifier>
     vdoi://123.456/1/45/23/Counter1
   </lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.4 THE COUNTER-COUNTER OFFER (ANOTHER OFFER)

The licensor rejects John’s counter offer, but proposes another offer to John, which requires John to be a teacher to have the right to write.

D.4.1 Rejection of the Counter-Offer

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
The licensor rejects John’s counter offer-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Reject">

   <!-- Licensor -->
   <lirel:licensor>
     <lirel:identifier>
       jabber://licensor@exampleLiREL.net
     </lirel:identifier>
     <dd:qos>
       <lirel:detail>
         The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
       </lirel:detail>
     </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
     <lirel:party>
       <lirel:identifier>
         jabber://john@exampleLiREL.com
       </lirel:identifier>
       <dd:prePay lirel:negotiable="false">
         <lirel:detail>
    The licensee has to pay before an agreement is
    concluded
 </lirel:detail>
         <dd:money lirel:negotiable="false">
   <lirel:value>10</lirel:value>
   <dd:currency>EUR</dd:currency>
 </dd:money>
       </dd:prePay>
     </lirel:party>
   </lirel:licenseeGroup>

   <!-- Resources under discussion-->
   <lirel:resource>
     <lirel:identifier>
       vdoi://123.456/2/23/23.
     </lirel:identifier>
   </lirel:resource>

   <!-- Contract Terms-->
   <lirel:permissionGroup>
     <dd:read/>
   </lirel:permissionGroup>

   <dd:write>
     <dd:prePay>
       <lirel:detail>
         Will pay additional amount for the right to write
       </lirel:detail>
       <dd:money>
         <lirel:value>10</lirel:value>
         <dd:currency>EUR</dd:currency>
       </dd:money>
     </dd:prePay>
   </dd:write>

   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
     <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Rejection Identifier -->
   <lirel:identifier>
     vdoi://123.456/1/45/23/Counter1
   </lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.4.2 The Offer

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  The licensor responds. The offer requires the addition of a role
  of a "teacher" for write access to be granted.
-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="CounterOffer">

   <!-- Licensor -->
   <lirel:licensor>
     <lirel:identifier>
       jabber://licensor@exampleLiREL.net
     </lirel:identifier>
     <dd:qos>
       <lirel:detail>
         The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
       </lirel:detail>
     </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
     <lirel:party>
       <lirel:identifier>
         jabber://john@exampleLiREL.com
       </lirel:identifier>
       <dd:prePay lirel:negotiable="false">
         <lirel:detail>
    The licensee has to pay before an agreement is
    concluded
 </lirel:detail>
         <dd:money lirel:negotiable="false">
   <lirel:value>10</lirel:value>
   <dd:currency>EUR</dd:currency>
 </dd:money>
       </dd:prePay>
     </lirel:party>

     <lirel:party lirel:negotiable="false">
       <lirel:identifier>
         credential://TrustedCredentials/teacher
       </lirel:identifier>
     </lirel:party>
   </lirel:licenseeGroup>

   <!-- Resources under discussion-->
   <lirel:resource>
     <lirel:identifier>
       vdoi://123.456/2/23/23.
     </lirel:identifier>
   </lirel:resource>

   <!-- Contract Terms-->
   <lirel:permissionGroup>
     <dd:read/>
   </lirel:permissionGroup>

   <dd:write>
     <dd:prePay>
       <lirel:detail>
         Will pay additional amount for the right to write
       </lirel:detail>
       <dd:money>
         <lirel:value>10</lirel:value>
         <dd:currency>EUR</dd:currency>
       </dd:money>
     </dd:prePay>
   </dd:write>

   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
     <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Offer Identifier -->
   <lirel:identifier>
     vdoi://123.456/1/45/23/Offer2
```

```
   </lirel:identifier>
   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.5 The Acceptance of Offer

John accepts the new terms proposed by the licensor.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  John accepts the terms of offer 2.
-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/~aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/~aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/~aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/~aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Accept">

   <!-- Licensor -->
   <lirel:licensor>
     <lirel:identifier>
       jabber://licensor@exampleLiREL.net
     </lirel:identifier>
     <dd:qos>
       <lirel:detail>
         The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
       </lirel:detail>
     </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
     <lirel:party>
       <lirel:identifier>
         jabber://john@exampleLiREL.com
       </lirel:identifier>
       <dd:prePay lirel:negotiable="false">
         <lirel:detail>
    The licensee has to pay before an agreement is
    concluded
```

```
</lirel:detail>
        <dd:money lirel:negotiable="false">
  <lirel:value>10</lirel:value>
  <dd:currency>EUR</dd:currency>
</dd:money>
      </dd:prePay>
    </lirel:party>

    <lirel:party lirel:negotiable="false">
      <lirel:identifier>
        credential://TrustedCredentials/teacher
      </lirel:identifier>
    </lirel:party>
  </lirel:licenseeGroup>

  <!-- Resources under discussion-->
  <lirel:resource>
    <lirel:identifier>
      vdoi://123.456/2/23/23.
    </lirel:identifier>
  </lirel:resource>

  <!-- Contract Terms-->
  <lirel:permissionGroup>
    <dd:read/>
  </lirel:permissionGroup>

  <dd:write>
    <dd:prePay>
      <lirel:detail>
        Will pay additional amount for the right to write
      </lirel:detail>
      <dd:money>
        <lirel:value>10</lirel:value>
        <dd:currency>EUR</dd:currency>
      </dd:money>
    </dd:prePay>
  </dd:write>

  <!-- Contract Constraints -->
  <dd:jurisdiction lirel:negotiable="false">
    <lirel:value>South Africa</lirel:value>
  </dd:jurisdiction>

  <dd:validUntil lirel:negotiable="true">
```

```
      <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Acceptance Identifier -->
   <lirel:identifier>
      vdoi://123.456/1/45/23/Accept-Offer2
   </lirel:identifier>

   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

D.6 The Concluding of an Agreement

The licensor creates the final use license.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
   The licensor creates the final agreement
-->
<lirel:license
   xmlns:lirel="http://www.cs.uct.ac.za/˜aarnab/REL/lirel"
   xmlns:dd="http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1"
   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
   xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.cs.uct.ac.za/˜aarnab/REL/lirel
   lirel.xsd http://www.cs.uct.ac.za/˜aarnab/REL/lireldd1
   lirel-lv1-dd.xsd" lirel:type="Agreement">

   <!-- Licensor -->
   <lirel:licensor>
      <lirel:identifier>
         jabber://licensor@exampleLiREL.net
      </lirel:identifier>
      <dd:qos>
         <lirel:detail>
            The license will provide access to a high resoultion
 (>300 DPI) version of the eBook
         </lirel:detail>
      </dd:qos>
   </lirel:licensor>

   <!-- Licensee -->
   <lirel:licenseeGroup>
      <lirel:party>
      <lirel:identifier>
        jabber://john@exampleLiREL.com
      </lirel:identifier>
      <dd:prePay lirel:negotiable="false">
        <lirel:detail>
   The licensee has to pay before an agreement is
   concluded
</lirel:detail>
        <dd:money lirel:negotiable="false">
  <lirel:value>10</lirel:value>
  <dd:currency>EUR</dd:currency>
</dd:money>
      </dd:prePay>
    </lirel:party>

    <lirel:party lirel:negotiable="false">
      <lirel:identifier>
        credential://TrustedCredentials/teacher
      </lirel:identifier>
    </lirel:party>
  </lirel:licenseeGroup>

  <!-- Resources under discussion-->
  <lirel:resource>
    <lirel:identifier>
      vdoi://123.456/2/23/23.
    </lirel:identifier>
  </lirel:resource>

  <!-- Contract Terms-->
  <lirel:permissionGroup>
    <dd:read/>
  </lirel:permissionGroup>

  <dd:write>
    <dd:prePay>
      <lirel:detail>
        Will pay additional amount for the right to write
      </lirel:detail>
      <dd:money>
        <lirel:value>10</lirel:value>
        <dd:currency>EUR</dd:currency>
      </dd:money>
    </dd:prePay>
  </dd:write>
```

```
   <!-- Contract Constraints -->
   <dd:jurisdiction lirel:negotiable="false">
     <lirel:value>South Africa</lirel:value>
   </dd:jurisdiction>

   <dd:validUntil lirel:negotiable="true">
     <lirel:value>2007-12-31</lirel:value>
   </dd:validUntil>

   <!-- Agreement Identifier -->
   <lirel:identifier>
      vdoi://123.456/1/45/23/Agreement
   </lirel:identifier>

   <lirel:dateOfIssue>2007-04-13</lirel:dateOfIssue>
</lirel:license>
```

BIBLIOGRAPHY

[1] PlayFair.
URL: http://playfair.sourceforge.net/.

[2] The Constitution of the United States of America, Article 1, Section 8.8.

[3] Zune problems for MSN customers. *BBC News (online)* (06 Nov 2006).
URL: http://news.bbc.co.uk/2/hi/technology/6120272.stm.

[4] Copyright act 98 of 1978. 2000 ed., vol. 2 of *Statutes of South Africa*. Juta, 2001, pp. 2–214–2–234. As amended by (acts/year): 56/1980, 66/1983, 52/1984, 39/1986, 13/1988, 61/1989, 125/1992, 38/1997, 9/2002.

[5] Directive 2001/29/EC Of the European Parliament and Of the Council of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society. Online, 2001. Also known as the "European Copyright Directive".

[6] *eXtensible rights Markup Language (XrML) 2.0 Specification*, 2001.

[7] Electronic communications and transactions (ECT) act 25 of 2002, 2002.

[8] DRM From the Viewpoint of the Electronic Industry. *Slashdot* (2003).
URL: http://slashdot.org/article.pl?sid=03/11/25/1821218.

[9] Extensible Resource Identifier (XRI) general syntax and resolution specification, 2004.

[10] Linux and DRM? *Slashdot* (2004).
URL: http://ask.slashdot.org/article.pl? sid=04/02/10/2329229&mode=thread.

[11] New Tool Cracks Apple's Fairplay DRM. *Slashdot* (2004).
URL: http://apple.slashdot.org/comments.pl?sid=102992.

[12] Playfair relocates to India. *Slashdot* (2004).
URL: http://slashdot.org/article.pl?sid=04/04/13/1156258.

[13] Adobe livecycle document security. Data sheet, Adobe, 2005.
URL: http://www.adobe.com/products/server/securityserver/pdfs/docsecurityserver_ds.pdf.

[14] Adobe reader, 2005.
URL: http://www.adobe.com/products/acrobat/readermain.html.

[15] Give and yea shall receive – The Bittorrent Revolution. *SA Computer Magazine 12*, 10 (February 2005), 38 – 43.

[16] NAVSHP (FP6) DRM requirement report, 2005.

[17] A primer on electronic document security. Technical white paper, Adobe, 2005.
URL: http://www.adobe.com/security/pdfs/acrobat_security_wp.pdf.

[18] Online music makes up for cd sales losses: survey. *Reuters.com* (2006-03-27). Online, last accessed: 2006-04-23.

[19] Approved document no 1, wd 3.0 – technical reference: Value chain functions and requirements, phase iii. *The Digital Media Project* (2006-11-02).
URL: http://www.dmpf.org/open/dmp0861.zip.

[20] Approved document no 2 wd 3.0 – technical reference: Architecture, phase iii. *The Digital Media Project* (2006-11-02).
URL: http://www.dmpf.org/open/dmp0862.zip.

[21] Gates: Digital locks too complex. *BBC NEWS* (2006-12-15 11:38:36 GMT).
URL: http://news.bbc.co.uk/go/pr/fr/-/2/hi/technology/6182657.stm Last Accessed: 2006-12-30.

[22] Evaluating DRM: Building a marketplace for the convergent world. *Center for Democracy & Technology* (September 2006).

[23] ALVESTRAND, H. The role of standards process in shaping the internet. *Proceedings of the IEEE 92*, 9 (2004), 1371 – 1374.

[24] AMERICAN HERITAGE DICTIONARIES, Ed. *The American Heritage Dictionary of the English Language*, fourth ed. Houghton Mifflin Company, 2000.

[25] ANDREOLI, J.-M., PACULL, F., AND PARESCHI, R. XPECT: A framework for electronic commerce. *IEEE Internet Computing 1*, 4 (1997).

[26] APPLE INC. Apple iTunes – Overview.
URL: http://www.apple.com/itunes/overview.html.

[27] APPLE INC. Apple customer privacy statement, 2004-12.
URL: http://www.apple.com/legal/privacy/ Last Accessed: 2007-01-29.

[28] APPLE INC. iTunes Store tops two billion songs, 2007-01-09.
URL: http://www.apple.com/pr/library/2007/01/09itunes.html Last Accessed: 2007-01-29.

[29] ARNAB, A., AND HUTCHISON, A. Extending ODRL and XrML to enable Bi-directional communication. Departmental Technical Report, No. CS04-28-00, University of Cape Town, 2004.

[30] ARNAB, A., AND HUTCHISON, A. Extending ODRL to Enable Bi-Directional Communication. In *Proceedings of the 2^{nd} International ODRL Workshop* (2005).

[31] ARNAB, A., AND HUTCHISON, A. Fairer usage contracts for DRM. In *Proceedings of the fifth ACM Workshop on Digital Rights Management, Co-Located with ACM CCS 2005* (2005), R. Safavi-Naini and M. Yung, Eds., ACM, pp. 1 – 7.

[32] ARNAB, A., AND HUTCHISON, A. DRM use license negotiations using ODRL v2.0, 2006. Submitted to the discussions in the 9th General Assembly of the Digital Media Project (DMP), Laussanne, Switzerland.

[33] ARNAB, A., AND HUTCHISON, A. Piracy and content protection in the broadband age. In *Proceedings of the South African Telecomunication Networks and Applications (SATNAC) Conference 2006* (2006).

[34] ARNAB, A., AND HUTCHISON, A. Ticket based identity system for DRM. In *Proceedings of Information Security South Africa (ISSA) Conference 2006* (2006).

[35] ARNAB, A., AND HUTCHISON, A. Verifiable digital object identity system. In *Proceedings of the Sixth ACM Workshop on Digital Rights Management, Co-Located with ACM CCS 2006* (2006), K. Kurosawa, R. Safavi-Naini, and M. Yung, Eds., ACM.

[36] ARNAB, A., AND HUTCHISON, A. DRM use license negotiations using ODRL v2.0. In *Proceedings of the 5th International Workshop for Technical, Economic and Legal Aspects of Business Models for Virtual Goods incorporating the 3rd International ODRL Workshop* (2007), R. Grimm, B. Hass, and J. Nuetzel, Eds.

[37] ARNAB, A., AND HUTCHISON, A. Persistent access control: A formal model for drm. In *DRM'07 Proceedings of the 2007 ACM Workshop on Digital Rights Management* (2007), A. Kiayias and A. Reza-Sadeghi, Eds., ACM, pp. 41 – 53.

[38] ARNAB, A., AND HUTCHISON, A. Using payment gateways to maintain privacy in secure electronic transactions. In *Proceedings of IFIPSEC 2007* (2007).

[39] ARNAB, A., AND NUNEZ, D. Loading ... a short study into security delay frustration. Tech. Rep. CS06-03-00, University of Cape Town, 2006.

[40] ARNAB, A., PAULSE, M., BENNETT, D., AND HUTCHISON, A. Investigation of a kernel level DRM implementation". Tech. Rep. CS07-01-00, 2007.
URL: http://pubs.cs.uct.ac.za/archive/00000389/.

[41] ARNAB, A., PAULSE, M., BENNETT, D., AND HUTCHISON, A. Investigation of a kernel level DRM implementation. In *Proceedings of AXMEDIS 2007* (2007).

[42] AUTHENTICA. Ip on the move: Protecting intellectual property in a competitive manufacturing environment. White paper, Authentica, 2004.

[43] AUTHENTICA. Authentica delivers next-generation enterprise rights management platform. Press release, Authentica, 2005.
URL: http://www.authentica.com/news/pr2005/02-14-2005-ARM.aspx?pf=1.

[44] AUTHENTICA. Enterprise rights management for document protection. White paper, Authentica, 2005.

[45] BARTOLINI, C., PREIST, C., AND KUNO, H. Requirements for automated negotiation. In *The W3C Workshop on Web Services: Position Papers* (2001).
URL: http://www.w3.org/2001/03/WSWS-popa/paper19.

[46] BARTOLINI, F., CAPPELLINI, PIVA, A., FRINGUELLI, A., AND M, B. Electronic Copyright Management Systems: Requirements, Players and Technologies. In *Proceedings of the Tenth International Workshop on Database and Expert Systems Applications* (1999), IEEE, pp. 896–899.

[47] BAUSE, F., AND KRITZINGER, P. S. *Stochastic Petri Nets – An Introduction to the Theory*. Vieweg & Sohn Verlagsgesellschaft mbH, 1996.

[48] BECHTOLD, S. Digital Rights Management in the United States and Europe. IVir, Buma/Stemra - Copyright and the Music Industry: Digital Dilemmas.

[49] BECHTOLD, S. Reconciling DRM Technology with Copyright Limitations. IVir, Buma/Stemra - Copyright and the Music Industry: Digital Dilemmas.

[50] BECKER, D. Passport to nowhere? *C-Net News.com.*
URL: http://news.com.com/2100-7345_3-5177192.html.

[51] BELL, D. E., AND LAPADULA, L. J. Secure computer system: Unified exposition and multics interpretation. Mtr-2997 rev. 1, The MITRE Corporation. Online, last accessed: 2006-05-06.
URL: http://csrc.nist.gov/publications/history/bell76.pdf.

[52] BELL, D. E., AND LAPADULA, L. J. Secure computer systems: A mathematical model. *Journal of Computer Security 4*, 2/3 (1996), 229 – 263. Reprint of 1973 technical report M74 244, MITRE Corp.

[53] BERLIND, D. A load of C.R.A.P. *ZDNet.com* (2005).
URL: http://news.zdnet.com/html/z/wb/6035707.html Last Accessed: 2007-01-06.

[54] BERNERS-LEE, T., FIELDING, R., AND MASINTER, L. RFC 3986 – Uniform Resource Identifier (URI): Generic Syntax, 2005.
URL: http://www.faqs.org/rfcs/rfc3986.html.

[55] BINMORE, K., AND DASGUPTA, P. Nash bargaining theory: An introduction. *The Economics of Bargaining* (1987).

[56] BIRRELL, A. *Seven Lectures on the law and history of copyright in books*, 1971 reprint ed. Rothman Reprints, 1899.

[57] BLOCK, R. Fairuse4wm strips windows media DRM! *Engadget* (2006-08-25).
URL: http://www.engadget.com/2006/08/25/fairuse4wm-strips-windows-media-drm/ Last Accessed: 2006-08-26.

[58] BOTT, E. Imagine a world without DRM. *ZDNet.com* (2007-01-05).
URL: http://blogs.zdnet.com/Bott/?p=179 Last Accessed: 2007-01-06.

[59] BURROWS, P. I want my iTunes subscription service! *BusinessWeek.com* (2005-08-16).
URL: http://www.businessweek.com/the_thread/techbeat/archives/2005/08/i_want_my_itune.html Last Accessed: 2007-01-29.

[60] BYERS, S., CRANOR, L., KORMAN, D., MCDANIEL, P., AND CRONIN, E. Analysis of Security Vulnerabilities in the Movie Production and Distribution Process. In *Proceedings of the 2003 ACM Workshop on Digital Rights Management* (2003), ACM, pp. 1–12.
URL: http://doi.acm.org/10.1145/947380.947383.

[61] CAMP, L. J. DRM: doesn't really mean digital copyright management. In *Proceedings of the 9th ACM conference on Computer and communications security* (2002), ACM.

[62] CANTOR, S., KEMP, J., PHILPOTT, R., AND MALER, E., Eds. *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) v2.0.* OASIS, 2005. OASIS Standard; 15 March 2005.

[63] CHIARIGLIONE, L. A walkthrough in the DMP Phase II specification. *Digital Media Project* (2006).
URL: http://www.dmpf.org/documents/walkthrough_in_idp-2.htm last accessed: 2006-06-04.

[64] CHRISTIE, R. H. *The Law of Contract.* Butterworth Publishers (Pty) Ltd, 1994.

[65] COHEN, J. DRM and Privacy. *Communications of the ACM 46*, 4 (2003), 47–49.

[66] COHEN, P. iTunes hits the 50 million song mark. *The Industry Standard - Internet Business News* (2004).
URL: http://www.thestandard.com/article.php?story=20040315173205175.

[67] CORAL CONSORTIUM. Coral consortium whitepaper. Tech. rep.

[68] CORAL CONSORTIUM. Interoperable media & home networks. Tech. rep.

[69] CORAL CONSORTIUM. Providing interoperability with Windows Media DRM. Tech. rep.

[70] COYLE, K. Right Expression Languages, A report for the Library of Congress. Tech. rep., Library of Congress, USA, 2004.

[71] DAI, J., AND ALVES-FOSS, J. Logic based authorization policy engineering. In *Proceedings of the 6th World Multiconference on Systemics, Cybernetics, and Informatics* (2002), pp. 230 – 238.

[72] DEAZLEY, R. *On the origin of the right to copy.* Hart Publishing, 2004.

[73] DEMARINES, V. Authentica's Enterprise DRM-Suite for Document Protection. White paper, Authentica, 2004.

[74] DIGITAL MEDIA PROJECT (DMP). Requirements for new or extended idp-3 technologies, 2006.
URL: http://www.dmpf.org/open/dmp0661.doc.

[75] DOOM9 FORUMS. DRM 10 cracked?
URL: http://forum.doom9.org/showthread.php?t=89243 Last Accessed: 2006-08-26.

[76] DOOM9 FORUMS. FairUse4WM - a WM/DRM removal program.
URL: http://forum.doom9.org/showthread.php?t=114916 Last Accessed: 2006-08-26.

[77] DUERST, M., AND SUIGNARD, M. Internationalized Resource Identifiers (IRIs), 2005.
URL: http://www.faqs.org/rfcs/rfc3987.html.

[78] DUFFT, N., STIEHLER, A., VOGELEY, D., AND WICHMANN, T. Digital music usage and DRM – results from an European Consumer Survey. Report, INDICARE Project, 2005.

[79] DUSOLLIER, S. Fair Use By Design in the European Copyright Directive of 2001. *Communications of the ACM 46*, 4 (2003), 51–55.

[80] ELFATATRY, A., AND LAYZELL, P. Negotiating in service-oriented environments. *Communications of the ACM 47*, 8 (2004).

[81] ERICKSON, J. Fair Use, DRM and Trusted Computing. *Communications of the ACM 46*, 4 (2003), 34–39.

[82] FELTEN, E. Skeptical view of DRM and Fair Use. *Communications of the ACM 46*, 4 (2003), 57–59.

[83] FERRAIOLO, D. F., BARKLEY, J. F., AND KUHN, D. R. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security 2*, 1 (1999), 34 – 64.

[84] FERRAIOLO, D. F., CUGINI, J. A., AND KUHN, D. R. Role-based access control (RBAC): Features and motivations. In *Annual Computer Security Applications Conference* (1995), IEEE Computer Society Press.
Available online: http://csrc.nist.gov/rbac/ferraiolo-cugini-kuhn-95.pdf.

[85] FERRAIOLO, D. F., AND KUHN, D. R. Role-based access control. In *Proceedings of the 15th NIST-NSA National Computer Security Conference* (1992).
Available online: http://csrc.nist.gov/rbac/ferraiolo-kuhn-92.pdf.

[86] FRIED, I. Apple goes to court to smoke out product leaker. *C-Net News.com* (21 December 2004).
URL: http://news.com.com/2102-1047_3-5499814.html.

[87] GAUDET, E. DRM vs. ERM: battle to control data. *ComputerWorld*.
URL: http://www.computerworld.com.au/index.php/id;1027773335;fp;4194304;fpid;1;pf;1
Last Accessed: 2006-12-30.

[88] GLADNEY, H. M. Trustworrthy 100-year digital objects: Evidence after every witness is dead. *ACM Transactions on Information Systems 22*, 3 (2004), 406 – 436.

[89] GODIK, S., AND MOSES, T., Eds. *eXtensible Access Control Markup Language*. OASIS, 2003. OASIS Standard; 18 February 2003.

[90] GONZLEZ, R. G. *A Semantic Web Approach to Digital Rights Management*. PhD thesis, 2005. Online: http://rhizomik.net/%7Eroberto/thesis/Thesis.pdf.

[91] GRAHAM, J. Emusic's pitch: Download song & and own it. *USA Today* (2006-07-30). URL: http://www.usatoday.com/tech/products/services/2006-07-30-emusic_x.htm Last Accessed: 2007-01-29.

[92] GROSS, T., AND PFITZMANN, B. Proving a WS-Federation passive requestor profile. In *Proceedings of the 2004 ACM Workshop on Secure Web Services (SWS), Co-Located with ACM CCS 2005* (2004), ACM.

[93] GUTH, S. *Interoperability of DRM System*. Peter Lang, 2006.

[94] GUTH, S., NEUMANN, G., AND STREMBECK, M. Experiences with the enforcement of access rights extracted from ODRL-based digital contracts. In *Proceedings of the 2003 ACM workshop on Digital Rights Management* (2003), ACM, pp. 90–102.

[95] HALPERN, J. Y., AND WEISSMAN, V. A formal foundation for XrML. In *Proceedings of the Seventeenth IEEE Computer Security Foundations Workshop* (2004), pp. 251 – 263.
URL: http://www.cs.cornell.edu/People/vickyw/papers-talks/XrML/CSFW04.pdf.

[96] HARMS, J. Biotech Laboratories (Pty) Ltd v Beecham Group plc and Another 2002. SA 249 (SCA). Case No: 494/2000, in the Supreme Court of Appeal, Republic of South Africa.
URL: http://www.supremecourtofappeal.gov.za/judgments/sca_judg/sca_2002/49400.pdf.

[97] HARPER, G. Will we need fair use in the twenty-first century? *Copyright Crash Course, University of Texas* (2004).
URL: http://www.utsystem.edu/OGC/IntellectualProperty/fair_use.htm.

[98] HEILEMAN, G. L., AND JAMKHEDKAR, P. A. DRM interoperability analysis from the perspective of a layered framework. In *Proceedings of the fifth ACM Workshop on Digital Rights Management, Co-Located with ACM CCS 2005* (2005), R. Safavi-Naini and M. Yung, Eds., ACM, pp. 17 – 26.

[99] HELIX COMMUNITY. Helix DNA client overview, 2006-03-27.
URL: https://client.helixcommunity.org/2003/devdocs/intro/client_intro.htm Last Accessed: 2007-02-04.

[100] HELIX COMMUNITY. Helix device DRM, 2007.
URL: https://devicedrm.helixcommunity.org/ Last Accessed: 2007-02-04.

[101] HELIX COMMUNITY. Welcome to the helix community!, 2007.
URL: https://helixcommunity.org/ Last Accessed: 2007-02-03.

[102] HYCHE, E. Unified logging, 2004-12-07.
URL: https://common.helixcommunity.org/2005/devdocs/UnifiedLogging Last Accessed: 2007-02-03.

[103] IANNELLA, R., Ed. *Open Digital Rights Language (ODRL) 1.1*. IPR Systems Pty Ltd., 2002.
URL: http://odrl.net/1.1/ODRL-11.pdf.

[104] IANNELLA, R., AND GUTH, S., Eds. *Open Digital Rights Language (ODRL) Version 2 Requirements*. 13 Feb 2005.
URL: http://odrl.net/2.0/v2req.html.

[105] IANNELLA, R., AND GUTH, S., Eds. *ODRL V2.0 - Model Semantics*. 13 Jan 2007.
URL: http://odrl.net/2.0/WD-ODRL-Model-20070113.html last accessed: 2007-08-23.

[106] JAJODIA, S., SAMARATI, P., AND SUBRAHMANIAN, V. A logical language for expressing authorizations. In *Proceedings of 1997 IEEE Symposium on Security and Privacy* (1997), pp. 31–42.

[107] JAMKHEDKAR, P. A., AND HEILEMAN, G. L. DRM as a Layered System. In *Proceedings of the Fourth ACM Workshop on Digital Rights Management* (2004), A. Kiayias and M. Yung, Eds., ACM, pp. 11 – 21.

[108] JAMKHEDKAR, P. A., HEILEMAN, G. L., AND MARTINEZ-ORTIZ, I. The problem with rights expression languages. In *DRM '06: Proceedings of the ACM workshop on Digital rights management* (New York, NY, USA, 2006), ACM Press, pp. 59–68.

[109] JEGES, E., AND KERÉNYI, K. Human factors of DRM. Report on the 5th indicare workshop, INDICARE Project, 2005.
URL: http://www.indicare.org/tiki-download_file.php?fileId=193.

[110] JENNINGS, N., FARATIN, P., LOMUSCIO, A., PARSONS, S., WOOLDRIDGE, M., AND SIERRA, C. Automated negotiation: Prospects, methods and challenges. *Journal of Group Decision and Negotiation 10*, 2 (2001), 199 – 215.

[111] JOHANSEN, J. L. Sharpmusique, 2005-09-17.
URL: http://nanocrew.net/2005/09/17/sharpmusique-10/ Last Accessed: 2007-01-29.

[112] JONES, C., LASHFORD, R., AND BUSS, A. EMEA mobile device trends 2003. Analyst report, Canalys.com, 2003.

[113] KALKER, T. Invited talk: On DRM interoperability. In *DRM '06: Proceedings of the ACM workshop on Digital rights management* (New York, NY, USA, 2006), K. Kurosawa, R. Safavi-Naini, and M. Yung, Eds., ACM Press, pp. 45–46.

[114] KATZENBEISSER, S., KURSAWE, K., AND TALSTRA, J. Graceful infringement reactions in DRM systems. In *Proceedings of the Sixth ACM Workshop on Digital Rights Management, Co-Located with ACM CCS 2006* (2006), K. Kurosawa, R. Safavi-Naini, and M. Yung, Eds., ACM.

[115] KIM, B. J., HONG, S. J., AND KIM, J. Ticket-based fine-grained authorization service in the dynamic VO environment. In *Proceedings of the 2004 ACM Workshop on Secure Web Services (SWS), Co-Located with ACM CCS 2004* (2004), ACM.

[116] KIM, J. MP3 Insider: the truth about battery life. *CNET.com* (2006-03-13).
URL:http://reviews.cnet.com/4520-6450_7-6462771-1.html Last Accessed: 2007-01-29.

[117] KIM, J. MP3 Insider: all-you-can-eat itunes–why not? *CNET.com* (2006-05-30).
URL: http://reviews.cnet.com/4520-6450_7-6534082-1.html Last Accessed: 2007-01-29.

[118] KOENEN, R. H., LACY, J., MACKAY, M., AND MITCHELL, S. The long march to interoperable digital rights management. *Proceedings of the IEEE 92*, 6 (2004), 883 – 897.

[119] KUDO, M., AND HADA, S. XML document security based on provisional authorization. In *CCS '00: Proceedings of the 7th ACM conference on Computer and communications security* (New York, NY, USA, 2000), ACM Press, pp. 87–96.

[120] KUNTZE, N., AND SCHMIDT, A. U. Trusted ticket systems and applications. In *Proceedings of IFIP Sec 2007* (2007).

[121] KUNZE, J., AND RODGERS, R. The ARK persistent identifier scheme, 2005.
URL: http://www.ietf.org/internet-draft/draft-kunze-ark-10.txt.

[122] LEAFER, M. *Understanding Copyright Law.* Matthew Bender & Company, 1989.

[123] LEYDEN, J. First trojan using sony drm spotted. *The Register* (2005-11-10).
URL: http://www.theregister.co.uk/2005/11/10/sony_drm_trojan/
last accessed: 2006-04-09.

[124] LIFSHITZ, Z. TIRAMISU DRM requirements. *The Digital Media Project* (2004-05-10).
URL: http://www.dmpf.org/open/dmp0086.doc.

[125] LITMAN, J. *Digital Copyright.* Promotheus Books, 2001.

[126] LOTSPIECH, J., NUSSER, S., AND PESTONI, F. Anonymous trust: Digital rights management using broadcast encryption. *Proceedings of the IEEE 92*, 6 (2004), 898 – 909.

[127] MAGUIRE, J. Microsoft vs. iTunes. *NewsFactor.com* (2004).

[128] MAINE, H. S. *Ancient Law*, 10 ed. 1884. Reprinted 1970 by Peter Smith. 1st Edition 1861.

[129] MARLIN COMMUNITY. Marlin architecture overview. Tech. rep., 2006.
URL: http://www.marlin-community.com/images/wp/MarlinArchitectureOverview.pdf.

[130] MASANGO, C. Digital license agreements and its effects on academic library users. In *LIASA Acquisitions Interest Group (LACIG) - Third Southern African Library Acquisitions Conference* (2004).

[131] MEHROTRA, A. *GSM System Engineering*, first ed. Artech House, 1997.

[132] MICHIELS, S., VERSLYPE, K., JOOSEN, W., AND DECKER, B. D. Towards a software architecture for DRM. In *Proceedings of the fifth ACM Workshop on Digital Rights Management, Co-Located with ACM CCS 2005* (2005), R. Safavi-Naini and M. Yung, Eds., ACM, pp. 65 – 74.

[133] MICROSOFT. Features of windows media drm.
URL: http://www.microsoft.com/windows/windowsmedia/forpros/drm/features.aspx Last Accessed: 2006-08-23.

[134] MICROSOFT. Sdks and versions of windows media drm.
URL: http://www.microsoft.com/windows/windowsmedia/forpros/drm/sdkandversions.aspx Last Accessed: 2006-08-23.

[135] MICROSOFT. *Taking Advantage of Super Distribution with Windows Media Rights Manager 7*, 2000.
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwmt/html/wmrm7distribution.asp.

[136] MICROSOFT. Microsoft windows media data session toolkit, 2003.

[137] MICROSOFT. Technical overview of windows rights management services for windows server 2003. White paper, 2003.

[138] MICROSOFT. Windows right management services - data sheet, 2003.
URL: http://www.microsoft.com/windowsserver2003/techinfo/overview/rmsdatasheet.mspx.

[139] MICROSOFT. Windows rights management services: Protecting electronic content in financial, healthcare, government and legal organizations, 2003.
URL: http://www.microsoft.com/windowsserver2003/techinfo/overview/rmsverticals.mspx.

[140] MICROSOFT. Windows media DRM FAQ, October 2005.
URL: http://www.microsoft.com/windows/windowsmedia/forpros/drm/faq.aspx Last Accessed: 2006-08-23.

[141] MPEG 21 REQUIREMENTS GROUP. MPEG-21 overview v5. Approved version, 2002-10.
URL: http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm.

[142] MULLIGAN, D., AND BURSTEIN, A. Implementing Copyright Limitations in Rights Expression Languages. In *Proceedings of the 2002 ACM workshop on Digital Rights Management* (2002), ACM.

[143] MULLIGAN, D., HAN, J., AND BURSTEIN, A. How DRM Based Content Delivery Systems Disrupt Expectations of "Personal Use". In *Proceedings of the 2003 ACM workshop on Digital Rights Management* (2003), ACM, pp. 77–89.
URL: http://doi.acm.org/10.1145/947380.947391.

[144] NATIONAL COMPUTER SECURITY CENTER. A guide to understanding discretionary access control in trusted systems. NCSC-TG-003, September 1987.

[145] NETWORKS, R. Helix digital rights management. Data sheet, Real Networks, 2005.
URL: http://docs.real.com/docs/rn/datasheet/Helix_DRM_S308.pdf.

[146] OPEN MOBILE ALLIANCE (OMA). OMA digital rights management v1.0. Approved version, 2004-06-25.
URL: http://www.openmobilealliance.org/release_program/drm_v1_0.html.

[147] OPEN MOBILE ALLIANCE (OMA). OMA digital rights management v2.0. Approved version, 2006-03-03.
URL: http://www.openmobilealliance.org/release_program/drm_v2_0.html.

[148] ORGANISATION FOR ECONOMIC CO-OPERATION AND DEVELOPMENT. *Recommednation of the OECD Council concerning guidelines for consumer protection in the context of electronic commerce*. 1999.

[149] PARK, J., SANDHU, R., AND SCHIFALACQUA, J. Security architectures for controlled digital information dissemination. In *Proceedings of the 16th Annual Computer Security Applications Conference* (2000).

[150] PASKIN, N. *The DOI Handbook*, 4.0.0 ed. International DOI Foundation, 2004.

[151] PEIG, E., AND DELGADO, J. Embedding ODRL statements in dublin core. In *Proceedings of the 2^{nd} International ODRL Workshop* (2005).

[152] PETERB. October 3rd declared "day against drm".
URL: http://defectivebydesign.org/en/blog/announce_day_against_drm Last Accessed: 2006-12-30.

[153] PINKAS, B. Efficient state updates for key management. *Proceedings of the IEEE 92*, 6 (2004), 910 – 917.

[154] PRUITT, D. G. *Negotiation Behaviour*. Academic Press Inc, 1981.

[155] PRUNEDA, A. *Security Overview of Microsoft Windows Media Rights Manager*, 2001.
URL: http://msdn.microsoft.com/library/en-us/dnwmt/html/wmrm71security.asp.

[156] PRUNEDA, A. *Getting Started with Windows Media Rights Manager SDK*, June 2002.
URL: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwmt/html/wmrm7quickstart.asp.

[157] PUCELLA, R., AND WEISSMAN, V. A logic for reasoning about digital rights. *CoRR cs.CR/0405066* (2004).

[158] PUCELLA, R., AND WEISSMAN, V. A formal foundation for ODRL. *CoRR cs.LO/0601085* (2006).

[159] RAGAN, S. Trouble for Apple's iTunes in Norway. *Monsters and Critics* (2007-01-29). URL: http://tech.monstersandcritics.com/news/article_1252978.php/ Trouble_for_Apples_iTunes_in_Norway Last Accessed: 2007-01-30.

[160] RAIFFA, H. *The Art and Science of Negotiation*. Harvard University Press, 1982.

[161] RASKIN, J. *The Human Interface – New directions for designing interactive systems*. Addison-Wesley, 2000.

[162] REAL NETWORKS. Helix DRM 10 from Real.
URL: http://www.realnetworks.com/products/drm/.

[163] REAL NETWORKS. Real networks introduces Harmony, enabling consumers to buy digital music that plays on all popular devices. *Real Networks*. URL: http://www.realnetworks.com/company/press/releases/2004/harmony.html.

[164] REID, J. F., AND CAELLI, W. J. DRM, Trusted Computing and Operating System Architecture. In *Conferences in Research and Practice in Information Techology* (Newcastle, Australia, 2005), vol. 44, Australian Computer Society, Inc., pp. 127 – 136.

[165] RHODES, T. Chapter 15 – Mandatory Access Control. FreeBSD Handbook, FreeBSD.org. Online, last accessed: 2006-05-06.
URL: http://www.freebsd.org/doc/handbook/mac.html.

[166] ROJAS, P. Pymusique returns. *Engadget.com* (2005-03-22).
URL: http://www.engadget.com/2005/03/22/pymusique-returns/ Last Accessed: 2007-01-29.

[167] ROSENBLATT, B. Solving the dilema of copyright protection online. *The Journal of Electronic Publishing 3*, 2 (1997).
URL: http://www.press.umich.edu/jep/03-02/doi.html.

[168] ROSENBLATT, B. DRM for the Enterprise, 2004. Jupiter Media Webinar.

[169] ROSENBLATT, B. Technology comparison: Authentica active rights management and microsoft windows rights management services. White paper, Giantsteps Media Technology Strategies, 2005. Document Commissioned by Authentica.

[170] ROSENBLATT, B., AND DYKSTRA, G. Integrating content management with digital rights management - imperatives and opportunities for digital content lifecycles. White paper, Giantsteps Media Technology Strategies, 2003.
URL: http://www.giantstepsmts.com/drm-cm_white_paper.htm.

[171] RUSSINOVICH, M. Sony, rootkits and digital rights management gone too far.
URL: http://www.sysinternals.com/blog/2005/10/sony-rootkits-and-digital-rights.html
last accessed: 2006-04-09.

[172] SAFAVI-NAINI, R., SHEPPARD, N. P., AND UEHARA, T. Import/export in digital rights management. In *Proceedings of the Fourth ACM Workshop on Digital Rights Management* (2004), A. Kiayias and M. Yung, Eds., ACM, pp. 99 – 110.

[173] SAMUELSON, P. DRM {AND, OR, VS.} The Law. *Communications of the ACM 46*, 4 (2003), 41–45.

[174] SANDHU, R. S., COYNE, E. J., FEINSTEIN, H. L., AND YOUMAN, C. E. Role-based access control models. *IEEE Computer 29*, 2 (1996), 38 – 47.

[175] SCHMIDT, A. U., TAFRESCHI, O., AND WOLF, R. Interoperability challenges for DRM systems. In *IFIP/GI Workshop on Virtual Goods* (2004).

[176] SCHNEIER, B. *Applied Cryptography*, second ed. John Wiley & Sons, 1996.

[177] SCHULTZ, C., AND MERRIL, P. Proposed requirements for interoperable DRM platforms. *The Digital Media Project* (2004-02-13).
URL: http://www.dmpf.org/open/dmp0025.doc.

[178] SHARROCK, R. *Business Transactions Law*, sixth ed. Juta & Co, LTD, 2002.

[179] SHEPPARD, N. P. On implementating MPEG-21 intellectual property management and protection. In *DRM'07 Proceedings of the 2007 ACM Workshop on Digital Rights Management* (2007), A. Kiayias and A. Reza-Sadeghi, Eds., ACM, pp. 10 – 22.

[180] SHIREY, R. RFC 2828 – Internet security glossary, 2000.
URL: http://www.faqs.org/rfcs/rfc2828.html.

[181] SLATER, D. Microsoft's zune won't play protected windows media. *EFF DeepLinks* (15 Sep 2006).
URL: http://www.eff.org/deeplinks/archives/004910.php.

[182] STALLINGS, W. *Protect your Privacy – A guide for PGP users*, first ed. Prentice Hall, 1995.

[183] STALLINGS, W. *Network Security Essentials – Applications and Standards*, international second ed. Prentice Hall, 2003.

[184] STOKES, S. *Digital Copyright: Law and Practice*, second ed. Hart Publishing, 2005.

[185] SU, S. Y., HUANG, C., HAMMER, J., HUANG, Y., LI, H., WANG, L., LIU, Y., PLUEMPITIWIRIYAWEJ, C., LEE, M., AND LAM, H. An internet-based negotiation server for e-commerce. *The VLDB Journal 10*, 1 (2001), 72–90.

[186] SUN, S., LANNOM, L., AND BOESCH, B. RFC 3650 – Handle System Overview, 2003.
URL: http://www.faqs.org/rfcs/rfc3650.html.

[187] ULUDAG, U., PANKANTI, S., PRABHAKAR, S., AND JAIN, A. K. Biometric cryptosystems: Issues and challenges. *Proceedings of the IEEE 92*, 6 (2004), 948 – 960.

[188] UNITED NATIONS COMMISSION ON INTERNATIONAL TRADE LAW. *UNCITRAL Model Law on Electronic Commerce with Guide to Enactment*. 1996. with additional article 5 bis as adopted in 1998.

[189] UNITED STATES, D. O. H. . H. S. Office for Civil Rights - HIPAA: Medical privacy - national standards to protect the privacy of personal health information.
URL: http://www.hhs.gov/ocr/hipaa/.

[190] VAN DER MERWE, T. M. A profile of the distance computing student softlifter. In *Proceedings of Information Security South Africa (ISSA) Conference 2006* (2006).

[191] VAN VEENEN, J., AND PRAKKEN, H. A verifiable protocol for arguing about rejections in negotiation. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems* (New York, NY, USA, 2005), ACM Press, pp. 1165–1166.
http://doi.acm.org/10.1145/1082473.1082675.

[192] VON SOLMS, S. H., AND VAN DER MERWE, I. The management of computer security profiles using a role-oriented approach. *Computers and Security 13*, 8 (1994), 673 – 680.

[193] WATKINS, T., BROCIOUS, C., AND JOHANSEN, J. L. pyMusique, 2005.
URL: http://drmnews.com/pymusique/ Original Website Broken, Last accessed: 2007-01-29.

[194] WEARDEN, G. Sony 'rootkit' prompts office clampdown on cd use. *c—net News.com* (2006-11-16).
URL: http://news.com.com/2102-7355_3-5951177.html?tag=st.util.print
last accessed: 2006-04-09.

[195] WIPO. Berne convention for the protection of literary and artistic works.
URL: http://www.wipo.int/clea/docs/en/wo/wo001en.htm.

[196] WIPO. Vision and strategic direction of wipo.
URL: http://www.wipo.int/about-wipo/en/dgo/pub487.htm.

[197] WOOLDRIDGE, M., AND PARSONS, S. Languages for negotiation. In *Proceedings of the Fourteenth European Conference on Artificial Intelligence (ECAI-2000)* (2000), W. Horn, Ed., John Wiley & Sons.
http://citeseer.ist.psu.edu/wooldridge00languages.html.