Models and Languages for Digital Rights

Carl A. Gunter
University of Pennsylvania
gunter@cis.upenn.edu

Stephen T. Weeks
InterTrust Technologies Inc.
sweeks@intertrust.com

Andrew K. Wright
InterTrust Technologies Inc.
wright@intertrust.com

Abstract

Digital Rights Management (DRM) devices provide persistent protection, the means to control the rendering of digital content to users. This enables new kinds of agreements between parties involved in trading intangible goods such as digital music. In this paper we propose a language and model capable of expressing a range of licenses of the kind that DRMs may be expected to support.

1. Introduction

A Digital Rights Management (DRM) system governs rendering of content. DRMs enable sellers of digital content to move beyond current distribution models. Currently a physical medium like a CD is sold to a consumer who receives the right to render all of the works on the CD for his or her own enjoyment as often as desired. This approach is limited in many ways now that media capable of carrying greater content are becoming available, and the Internet can be used to distribute content in bulk transfers or via streaming applications. The sale of a CD is similar to the sale of a good, whereas an ongoing service model may better fit the new distribution techniques. For instance, it may be better to sell a user the right to hear songs from a digital library at a low fixed price per rendering rather than selling the library as a set of hundreds of CDs at a high fixed price. Payments for the service could occur periodically, like paying for using electricity or a cell phone. However, digital goods have some different characteristics than these other services: some experimentation will be required to derive appropriate service models.

The aim of this paper is to make progress on this problem by developing a model and a language for describing licenses to digital works. We focus on licenses likely to be feasible using new DRM technologies such as those developed by InterTrust (`http://www.intertrust.com/drm/index.html`). While a full-fledged DRM system will allow content providers to make rendering contingent on a variety of events and conditions, such as payment, time and date, identity of the user, membership in a club, identity of the device, return of usage reports, recent contact with a clearinghouse, *etc.*, we begin by studing simple licenses that consist only of payment and rendering events. We develop a mathematical model capable of capturing the meanings of a wide variety of such licenses with precision. We then develop a language of licenses whose semantics is defined by reference to the model. This insures that every license has a completely clear and unambiguous interpretation that is defined without reference to any particular implementation of a DRM system.

2. A Semantic Model for DRM

Our model consists of a domain of sequences of events called *realities*, and a domain of sets of realities called *licenses*. Then, in a manner similar to that of denotational semantics for general-purpose programming languages [1], we can express the semantics of a rights management language as a function that maps terms of the language to elements of our domain of licenses. The character of our semantics is similar to those used for concurrency [2] where language constructs are modeled as traces of allowed events.

Our abstract model represents an *event*, $e \in$ *Event*, as a pair of a *time*, $t \in$ *Time*, and an *action*, $a \in$ *Action*:

$$e ::= t : a$$

Time is totally ordered by $<$, and the function $+ : \mathit{Time} \times \mathit{Period} \to \mathit{Time}$ adds a period, $p \in$ *Period*, to a time. There are two kinds of actions:

$$a ::= \mathsf{render}[w, d] \mid \mathsf{pay}[x]$$

where $w \in$ *Work* denotes a rights-managed *work*, $d \in$ *Device* represents a DRM-enabled device, and x is a decimal. Action $\mathsf{render}[w, d]$ represents rendering of work w by rights-enabled device d. Action $\mathsf{pay}[x]$ represents a payment of amount x of some currency from a licensee to

a license issuer. Thus the event $t:\mathsf{render}[w,d]$ means that at time t, work w was rendered on device d. The event $t:\mathsf{pay}[x]$ means that at time t, a payment of amount x was made.

A *reality*, $r \in \mathit{Reality}$, is a finite set of events, such that all events occur at distinct times:

$$r \in \mathit{Reality} = \{\ E \in \mathcal{P}(\mathit{Event}) \mid E \text{ is finite and for all } t:a \in E \text{ if } t:a' \in E \text{ then } a = a'\ \}$$

where $\mathcal{P}(E)$ represents the powerset of E (the set of all subsets of E). While there are infinitely many possible events, a reality is finite because it represents some set of events that may occur over the lifetime of the rights management system. We require events to occur at distinct times so that we may use their times to identify them. We write $r_{\leq t}$ to represent the prefix of r that occurs at or before time t; that is:

$$r_{\leq t} = \{t' : a \in r \mid t' \leq t\}.$$

We write $r \sqsubseteq r'$ to indicate that r is a prefix of r'; that is, there exists a t such that $r = r'_{\leq t}$.

Content owners use a rights management system to make a work available to an owner of a device under a license. In our model, a *license*, $l \in \mathit{License}$, is a set of realities:

$$l \in \mathit{License} = \mathcal{P}(\mathit{Reality})$$

Informally, a license authorizes the events of one of its realities. For example, the license

$$l_A = \left\{ \begin{array}{l} \{8{:}00:\mathsf{pay}[\$1],\ 8{:}01:\mathsf{render}[w_1,d_1]\}, \\ \{8{:}00:\mathsf{pay}[\$1],\ 8{:}02:\mathsf{render}[w_1,d_1]\}, \\ \{8{:}00:\mathsf{pay}[\$1],\ 8{:}03:\mathsf{render}[w_1,d_1], \\ \qquad\qquad\quad 8{:}04:\mathsf{render}[w_1,d_1]\}, \\ \{8{:}00:\mathsf{pay}[\$1]\} \end{array} \right\}$$

requires a payment at 8:00 and allows w_1 to be rendered on device d_1 at 8:01, 8:02, both 8:03 and 8:04, or not at all. Most licenses will consist of infinitely many realities in order to allow use of a work at one or more of infinitely many times during some period. The license above is odd from a commercial perspective since it insists, for instance, that if the work is w_1 is rendered at 8:03 then it must be rendered again at 8:04! In the next section we will define some more typical licenses.

To give a more formal meaning to a license, suppose $\mathbf{r}$ is the (unique) complete reality that actually occurs over the entire lifetime of the DRM system. A DRM system must attribute every event of $\mathbf{r}$ to a license, and, in general, should not attribute any event to more than one license (a single payment event should not satisfy the payment requirements of two different licenses). How events are attributed to a license (*e.g.*, whether the user designates which license to use for an event, or the system chooses an appropriate license, *etc.*) is not a concern of this model. Let $\mathbf{r}[l]$ be those events of $\mathbf{r}$ attributed to license l. We say a reality $r \in l$ of a license l is *viable* at some time t if its events up to that time are consistent with $\mathbf{r}[l]$.

Definition: Reality $r \in l$ of license l is *viable* for $\mathbf{r}[l]$ at t iff $\mathbf{r}[l]_{\leq t} \sqsubseteq r$.

We say a license is *fulfilled* if enough events attributed to it have occurred to satisfy one of its realities.

Definition: License l is *fulfilled* by $\mathbf{r}[l]$ at t iff $\mathbf{r}[l]_{\leq t} \in l$.

A license is breached if it has no viable realities, which means that either too many events attributed to it have occurred, or too few have occurred and time has progressed past when the missing events must occur.

Definition: License l is *breached* by $\mathbf{r}[l]$ at t iff there does not exist $r \in l$ that is viable for $\mathbf{r}[l]$ at t.

If too many events have occurred, $\mathbf{r}[l]_{\leq t}$ will contain events that cannot be found in any reality of l. If too few events have occurred, $\mathbf{r}[l]_{\leq t}$ will not contain enough events to match any reality of l.

To illustrate these definitions, consider

$$\mathbf{r}[l_A] = \{\ 8{:}00:\mathsf{pay}[\$1],\ 8{:}01:\mathsf{render}[w_1,d_1],\ 8{:}05:\mathsf{render}[w_1,d_1]\ \}.$$

All four realities of license l_A are viable for $\mathbf{r}[l_A]$ at $t <$ 8:01. Only the first reality is viable for 8:01 $\leq t <$ 8:05. No reality is viable for $t \geq$ 8:05. License l_A is unfulfilled by $\mathbf{r}[l_A]$ for $t <$ 8:00, fulfilled for 8:00 $\leq t <$ 8:05, and breached for $t \geq$ 8:05. It is breached for $t \geq$ 8:05 because event 8:05 occurred. For

$$\mathbf{r}'[l_A] = \{8{:}00:\mathsf{pay}[\$1],\ 8{:}03:\mathsf{render}[w_1,d_1]\ \}$$

license l_A is unfulfilled for $t <$ 8:00, fulfilled for 8:00 $\leq t <$ 8:03, unfulfilled for 8:03 $\leq t <$ 8:04, and breached for $t \geq$ 8:04. It is breached for $t \geq$ 8:04 because event 8:03 occurred but event 8:04 did not occur. Note that a license that is fulfilled at time t can be unfulfilled at a later time $t' > t$. A license that is breached at time t is breached for all $t' \geq t$. A license can never be both fulfilled and breached.

3. Standard Licenses

Using the model in the previous section, we define several standard families of licenses that we expect content

providers will use in a DRM system. The first three are defined directly as parameterized functions.

The "Up Front" license provides access to any work in set $W \in \mathcal{P}(\textit{Work})$ on any device in set $D \in \mathcal{P}(\textit{Device})$ beginning at time t_0 for period p, for an up-front payment of x:

$$\begin{array}{l}\text{UpFront}(t_0, x, p, W, D) = \\ \quad \{\ t_0 : \mathsf{pay}[x], \\ \qquad t_1 : \mathsf{render}[w_1, d_1],\ \ldots,\ t_n : \mathsf{render}[w_n, d_n] \\ \quad \mid n \geq 0, \\ \quad\ t_0 < t_1 < \ldots < t_n < t_0 + p, \\ \quad\ w_1, \ldots, w_n \in W,\ d_1, \ldots, d_n \in D\ \}\end{array}$$

This is similar to admission to a theme park where all rides are "free," but the admission is good only until the end of the day. A similar deal is often used for cable television, where access to a given set of channels is obtained for a month with an up-front payment. In this case the contract may be repeated each month with the cable company setting the price for the service on a monthly basis.

The "Flat Rate" license provides access to any work in set W on any device in set D beginning at time t_0 for period p, for a payment of x at the end of the period:

$$\begin{array}{l}\text{FlatRate}(t_0, x, p, W, D) = \\ \quad \{\ t_1 : \mathsf{render}[w_1, d_1],\ \ldots,\ t_n : \mathsf{render}[w_n, d_n], \\ \qquad t_{n+1} : \mathsf{pay}[x] \\ \quad \mid n \geq 0, \\ \quad\ t_0 < t_1 < \ldots < t_n < t_{n+1} < t_0 + p, \\ \quad\ w_1, \ldots, w_n \in W,\ d_1, \ldots, d_n \in D\ \}\end{array}$$

This is similar to paying for a meal at a restaurant with a fixed-price menu. The price is known in advance, but paid at the end of the meal.

Finally, the "Per Use" license provides access to any work in set W on any device in set D beginning at time t_0 for a period of length p, for a payment of x per use at the end of the period:

$$\begin{array}{l}\text{PerUse}(t_0, x, p, W, D) = \\ \quad \{\ t_1 : \mathsf{render}[w_1, d_1],\ \ldots,\ t_n : \mathsf{render}[w_n, d_n], \\ \qquad t_{n+1} : \mathsf{pay}[nx]\ \mid \\ \quad\ n \geq 0, \\ \quad\ t_0 < t_1 < \ldots < t_n < t_{n+1} < t_0 + p, \\ \quad\ w_1, \ldots, w_n \in W,\ d_1, \ldots, d_n \in D\ \}\end{array}$$

This is similar to a utility bill like telephone or electricity minus any fixed or minimum per-period charge.

Let us call these *simple* licenses. We now consider how to compose them to obtain others. Simple licenses specify a single, specific starting time. A content owner will often wish to offer a license that a consumer can accept any time before some future date. For any of the three families defined above, such a license can be constructed by quantifying over t_0; for example:

$$\bigcup_{t_0 < t_{\text{expires}}} \text{UpFront}(t_0, x, p, W, D)$$

This license allows the period of use to begin anytime prior to t_{expires}.

Some care must be taken to get meaningful results in carrying out compositions. For instance, if one of the simple licenses offered 3 free renderings before beginning payment, then repeated exploitation of the union license above would enable a licensee to render as many works as he or she wishes while never incurring a payment obligation!

Simple licenses all hold for only a single service period. After the period ends, no further use is permitted. Often it is desirable to construct a license that holds for a number of service periods. Such a license can guarantee a regular payment schedule to a content provider, or lock in a rate to benefit a consumer. For the former case, given l_1 and l_2 such that all of the events of l_1 occur at different times from the events of l_2, let

$$l_1 \triangle l_2 = \{r_1 \cup r_2 \mid r_1 \in l_1,\ r_2 \in l_2\}.$$

With this operator, we can construct multi-period non-cancellable licenses by composing single-period licenses. $\text{UpFront}^{\triangle}(t_0, x, p, W, D, m)$ provides access to any work in set W on any device in set D beginning at time t_0 for m periods of length p, for payments of x at the beginning of each period:

$$\begin{array}{l}\text{UpFront}^{\triangle}(t_0, x, p, W, D, m) = \\ \quad \text{UpFront}(t_0, x, p, W, D) \triangle \ldots \triangle \\ \quad \text{UpFront}(t_{m-1}, x, p, W, D) \\ \qquad \text{where } t_i = t_0 + ip \text{ for } i \text{ from } 0 \text{ to } m - 1\end{array}$$

An $\text{UpFront}^{\triangle}$ license cannot be unilaterally cancelled—payments in all periods must be made, and service must be available for all paid-for periods at the pre-agreed rate. This kind of contract is similar to the lease on an apartment where commonly a fixed rate is paid at the beginning of each month for a year. We can define multi-period FlatRate and PerUse license families similarly. A multi-period PerUse license would be similar to a telephone contract where the provider offered a fixed per-use charge for a year. Such contracts are unusual, however, because there is no obligation for the user to use the service. Hence such an arrangment is usually part of a more complex contract that also insists on a regular base charge on a FlatRate basis.

A cancellable multi-period license allows the licensee to discontinue use of the license at any period boundary. Such a license can allow a consumer to lock in a rate for some

longer term. A cancelled license is not renewable; that is, once a payment for one period has been skipped, no use of content in future periods is authorized by that license. Given l_1 and l_2 such that all of the events of l_1 occur at different times from the events of l_2, let

$$l_1 \triangleright l_2 = \{r_1 \cup r_2 \mid r_1 \in l_1,\ r_1 \neq \emptyset,\ r_2 \in l_2\} \cup \{\emptyset\}.$$

With this operator, we can define cancellable multi-period licenses. $\text{UpFront}^{\triangleright}(t_0, x, p, W, D, m)$ provides access to any work in set W on any device in set D beginning at time t_0 for m periods of length p, for payments of x at the beginning of each period, cancellable after any period:

$$\begin{aligned}&\text{UpFront}^{\triangleright}(t_0, x, p, W, D, m) = \\ &\quad \text{UpFront}(t_0, x, p, W, D) \triangleright \ldots \triangleright \\ &\quad \text{UpFront}(t_{m-1}, x, p, W, D) \\ &\qquad \text{where } t_i = t_0 + ip \text{ for } i \text{ from } 0 \text{ to } m-1\end{aligned}$$

Such a license is similar to a renter with rent protection. The landloard cannot raise the charge unless the lease is breached or not renewed by the tenant. Cancellable FlatRate and PerUse licenses can also be defined. We can quantify over the start time as with single-period licenses to build cancellable and non-cancellable multi-period UpFront, FlatRate, and PerUse licenses that are offered until some future date.

4. Language

The model and standard licenses developed in the preceding sections suggest a rudimentary language for expressing DRM licenses. The language, which we call *DigitalRights*, is defined by the following grammar:

$$\begin{aligned}e \quad ::= \quad & \left(\textsf{at } t \mid \textsf{until } t\right) \\ & \left(\textsf{for } p \mid \textsf{for } \left[\textsf{up to}\right]\ m\ p\right) \\ & \textsf{pay } x \left(\textsf{upfront} \mid \textsf{flatrate} \mid \textsf{peruse}\right) \\ & \textsf{for } W \textsf{ on } D\end{aligned}$$

A DigitalRights license is an expression, e, of the DigitalRights language. A DigitalRights license has four components. The first component, introduced by at or until, indicates the term of applicability of the license. Informally, at t means that the license begins exactly at time t; until t means that the license may be accepted anytime prior to time t. The second component, introduced by for, indicates the payment period (p), number of periods for which the license holds (m or 1 if m is missing), and whether the license may be terminated by the licensee at the end of any period (up to permits termination). The third component, introduced by pay, indicates the amount of each payment and when during the payment it must be made. The last component indicates which works the license applies to and on which devices it may be used. In this last component, our DigitalRights language is somewhat incomplete. The language should include explicit syntactic forms for describing sets of works and devices, but this is not the focus of this paper.

The DigitalRights license

> until 01/01/03
> for up to 12 months
> pay $10.00 upfront
> for "Jazz Classics"
> on "devices registered to the license holder"

provides access to "Jazz Classics" works for an up-front monthly payment of $10.00, beginning anytime prior to January 2003, for up to 12 months, on devices registered to the license holder. "Jazz Classics" might describe a set of tracks on a disk, or a section of an online digital music library.

Using our semantic model of DRM, we can give a precise interpretation of a DigitalRights license. The semantic function $\mathcal{M}[\![\cdot]\!]$, defined in figure 1, maps a DigitalRights license to an element of our model (z is a suffix of a license). This semantics can definitively answer some questions whose answers might not be clearly spelled out in an informal English specification, such as whether a license until t ... can be accepted at time t or only before time t.

The model gives a precise interpretation for the example DigitalRights license above, without limiting how a DRM system might enforce it. Since this license potentially applies to several devices, the DRM system might ensure that monthly payments occur by requiring each device to hold a valid month payment certificate obtained from a central clearing site. Alternatively, a device might acquire the necessary certificate from another device. Other enforcement mechanisms than certificates are also possible.

5. DRM Enforcement and Audit Functions

A DRM system is used to prevent or detect a breach in a license. In some cases the DRM can ensure that the license is never breached. For instance, if payment is made in advance, then the DRM device may stop rendering works after the paid-for service is completed. In other cases the DRM is used to extract accurate billing information. For instance, if payment per use is expected with payment on a monthly basis, then the DRM device could cease to function at the end of a month unless it is placed online to report use rate and receive confirmation of payment.

Limits of DRM protection mechanisms will determine which licenses are implementable. A license that relies on users to report render events on an honor basis will

$$
\begin{array}{rcl}
\mathcal{M}[\![\textsf{at } t\ z]\!] & = & \mathcal{M}_1[\![z]\!](t) \\
\mathcal{M}[\![\textsf{until } t\ z]\!] & = & \bigcup_{t'<t} \mathcal{M}_1[\![z]\!](t') \\
 & & \\
\mathcal{M}_1[\![\textsf{for } p\ z]\!](t_0, p) & = & \mathcal{M}_2[\![z]\!](t_0, p) \\
\mathcal{M}_1[\![\textsf{for up to } m\ p\ z]\!](t_0) & = & \mathcal{M}_2[\![z]\!](t_0, p) \triangleright \ldots \triangleright \mathcal{M}_2[\![z]\!](t_{m-1}, p) \\
 & & \text{where } t_i = t_0 + ip \text{ for } i \text{ from } 0 \text{ to } m-1 \\
\mathcal{M}_1[\![\textsf{for } m\ p\ z]\!](t_0) & = & \mathcal{M}_2[\![z]\!](t_0, p) \mathbin{\triangle} \ldots \mathbin{\triangle} \mathcal{M}_2[\![z]\!](t_{m-1}, p) \\
 & & \text{where } t_i = t_0 + ip \text{ for } i \text{ from } 0 \text{ to } m-1 \\
 & & \\
\mathcal{M}_2[\![\textsf{pay } x \textsf{ upfront for } W \textsf{ on } D]\!](t, p) & = & \text{UpFront}(t, x, p, W, D) \\
\mathcal{M}_2[\![\textsf{pay } x \textsf{ flatrate for } W \textsf{ on } D]\!](t, p) & = & \text{FlatRate}(t, x, p, W, D) \\
\mathcal{M}_2[\![\textsf{pay } x \textsf{ peruse for } W \textsf{ on } D]\!](t, p) & = & \text{PerUse}(t, x, p, W, D)
\end{array}
$$

Figure 1. Semantic function for DigitalRights.

probably not be commercially viable. A basic form of DRM protection is to hide a private key from the user of the device using a mechanism such as a smart card or software obfuscation. Then works can be sold to the user encrypted under his public key. Assuming persistent protection, this user will be able to render the purchased work as often as the license permits, but will not be able to make copies for friends. Licenses that limit the length of time the work is available would require that the enforcing DRM device have a tamper-proof clock. Service-oriented licenses may require the DRM device to go online periodically so that usage can be reported for billing and payment purposes. An ability to purchase rights for a set of devices may require the devices to periodically communicate with one another to ensure license enforcement.

These mechanisms are closely related to our language since licenses that cannot be practically enforced are not useful for DRMs. Thus the language provides an avenue for classification. The licenses in this paper are ones for which practical DRM enforcement mechanisms are anticipated.

6. Related Work

Very little published work is available in the area of languages for digital rights management. Most of this work is being done by companies who are developing proprietary systems, and to date these companies (including InterTrust) have released little information about any languages they employ.

Content Guard, a spin off from Xerox PARC, is developing a DRM system based on a proprietary rights language called XrML. While no details of XrML are publicly available, it is derived from earlier published work on a language called DPRL [3]. With DPRL one can specify licenses with per-use or up-front payment terms, but it does not appear possible to specify a flat-rate payment that must be made at the end of a period. Nor does it appear possible to construct licenses that hold for multiple periods. We are not aware of a complete description of DPRL that includes even an informal semantics.

7. Conclusion

We have developed a mathematical model and a simple language for describing licenses for digital rights management systems. This model paves the way toward developing more sophisticated DRM languages with precise semantics, while leaving DRM vendors free to choose different methods of enforcement.

References

[1] C. A. Gunter. *Semantics of Programming Languages: Structures and Techniques*. Foundations of Computing. The MIT Press, 1992.

[2] C. A. R. Hoare. *Communicating Sequential Processes*. Series in Computer Science. Prentice-Hall, 1985.

[3] A. Ramanujapuram and P. Ram. Digital content & intellectual property rights. *Dr. Dobb's Journal*, 1998.