A Licensing and Payment System for Distribution of Digital Content

Russell Perry, Matthew M. Williamson
Information Infrastructure Laboratory
HP Laboratories Bristol
HPL-2002-167
June 12th , 2002*

The ease with which digital content can be copied and distributed is having a dramatic impact on all media industries. Media companies are understandably alarmed at the potential scale of piracy. To date media companies have tried to lock up their content using encryption technology. On it's own, this is fundamentally flawed and will ultimately degrade the customer experience. Instead, a change in the distribution and pricing of digital media is more urgently required. In this report we describe an approach to distribution of digital media that exploits word of mouth recommendation. A licensing system that monitors the distribution and payment of e-Books has been implemented to demonstrate the approach.

* Internal Accession Date Only
Approved for External Publication

Contents

1 Introduction

Digital media can be reproduced perfectly, with ease and at virtually no cost. Over the last few years, the internet has also enabled people to distribute media easily and cheaply. The combination of these two trends is having a dramatic impact on all media companies. Just as the photocopier and the lending library caused suspicion and fear in the past, media companies are again concerned about loss of control over their media. They are especially alarmed at the threat and potential scale of piracy.

This fear is not without foundation. For example, the original Napster [4] has shown how easy it is for users to make their music freely available on the Internet, where the media company is no longer collecting direct revenue for each distributed copy.

On the other hand the Internet offers enormous marketing and distribution opportunities, being able to deliver a wide variety of content immediately, all over the world. These characteristics make piracy easy, but also are an opportunity for media companies. This is particularly important because the value of media is proportional to the size of the audience [1]. Media companies must determine how to exploit this new distribution channel while still collecting money from customers.

To date, media companies have largely looked to technology companies to provide some sort of encryption technology to "lock up" the digital content and make it more difficult to copy. This would allow them to migrate their present business models onto the Internet unchanged. Unfortunately, such an approach is fundamentally flawed for a number of reasons. The technical difficulty and cost of managing security is likely to be high due to the variety of different access devices that will be supported, the inherent complexity of encryption based software and inevitable customer support costs (which will be significantly higher than customer support on CDs!). In addition the use of encryption degrades the customer experience, and would restrict the very distribution that is essential to generate interest.

We believe that this is a business model problem, not a technical one. The media industry is facing a shift from a business model based on direct revenue (sales of individual products) to one where the emphasis is on providing for (and charging for) a service. Revenue can be generated through advertising, through the ongoing relationship with the consumer (branded products etc.) through subscription and also through direct product sales. This shift will also change the way that consumers pay for their content.

The technical solutions that are required are ones that complement this new way of selling media. This means systems to easily connect users to the services, the services themselves, ways to express the flexible and dynamic product offerings that could be presented to customers.

In this paper we present one such technical solution, a working system that uses peer pressure and a service model to encourage legitimate usage and payment for media, without requiring any encryption. Users are encouraged to share media using a reward scheme, so exploiting the distribution opportunities presented by the Internet. Users connect to a service where they can browse new books, and buy any books that have been given to them by other users.

The system itself consists of an application for the Pocket PC that makes it easy for users to share books with one another, using the InfraRed interface to "beam" content. An application for the PC is also used so that whenever the PocketPC is docked with the host PC, the user is connected to an "e-book service". This service can examine the content on the PocketPC, decide what content is already owned by the user, and provide a means for users to buy content that they have got from their friends. If the user does buy the book, the originator of the book receives a reward, which they can use to purchase other books. The service also has a shop and offers a variety of different payment means. The service uses a networked repository of licenses to store the details of who owns what, allowing the possibility of customizing different products. The overall system shows how a variety of non-encryption mechanisms can be plausibly used to encourage sharing and still collect payment.

The rest of this paper describes the overall system, then goes into details of the implementation of the different parts: the PocketPC sharing software, the PC application, the e-book service, the representation used to express ownership of the books, and the payment mechanisms used. Appendix A contains an analysis of the trends that are emerging in the content industry.

2 System overview

We have built a prototype system that demonstrates the use of a service model for delivering content. The service is based on the use of peer pressure to encourage the sharing of media, and to encourage paying for it. There are two parts to the system, the first being an application developed for the Pocket PC, which allows users to share electronic books with one another. The second is a server based e-book service used to manage the books on the Pocket PC devices. An example scenario that illustrates the various aspects of the system is described below.

Each user of the system has a Pocket PC device, and loaded on it is an application that they can use to manage their e-books and share those books with other Pocket PC owners. A screenshot of the application is shown in Figure 1. This user (Matt Williamson) has bought two books and can read them on her device. For each book is stored a license that records the metadata of the book (title, author, publisher), along with the date of purchase, and the origin of the book, as shown in Figure 2. Since Matt legitimately owns the book, the license is a permanent one.

If Matt wants to share a book with a second user (Chuck Peters), he uses the send/receive buttons on the application (Figure 1) to transfer the book over Infra Red to Chuck. Chuck can immediately start reading the new book. In addition to the book, Matt also constructs a new temporary license for the book, which is also transferred. This license (shown in Figure 3) records that Chuck does not own the book, and that the origin of the book is Matt. This temporary license effectively gives Chuck the right to read the book until he next connects his device to the internet, which he would normally do regularly to sync contacts, calendar etc.

Figure 1: Screen shot of the Pocket PC. This user owns two books, "IP on the Net" and "Code".

```
<license>
        <ceoid>123456789</ceoid>
        <doi>ISBN:123</doi>
        <title>IP on the Net</title>
        <author>Esther Dyson</author>
        <publisher>edventure.com</publisher>
        <from>hp e-book service</from>
        <status>Permanent</status>
        <filename>iponnet.rdr</filename>
        <date>Tue, 10 Jul 2001 12:31:25+0100</date>
</license>
```

Figure 2: License stored on Pocket PC for IP on the Net. The ceoid field is an internal identifier, and the doi field is an identifier of the content. The from field indicated the origin of the license. The license has status "Permanent" because the book is legitimately owned

```
<license>
        <ceoid>123456789</ceoid>
        <doi>ISBN:123</doi>
        <title>IP on the Net</title>
        <author>Esther Dyson</author>
        <publisher>edventure.com</publisher>
        <from>Matt Williamson</from>
        <status>Temporary</status>
        <filename>iponnet.rdr</filename>
        <date>Thr 3 January 2002 13:51:25+0000</date>
</license>
```

Figure 3: Temporary license transmitted to the second user, with status "Temporary" and recording that the license came from Matt Williamson.

When Chuck connects or docks with a computer connected to the internet, a "Library Console" application automatically runs on the host computer to connect him to the e-book

service. This is illustrated in Figure 4. The console application reads the licenses from the Pocket PC, and sends them to the e-book service, where they are compared with a store of licenses in a LicenseBank. The service thus determines for each piece of content on Chuck's device whether

- It is new content, so Chuck should be offered a chance to buy it
- It is new content, but Chuck already owns it, so an updated license should be provided
- It is content that Chuck claims to own, but that he does not own it according to the LicenseBank. This indicates that an error has occurred and the metadata has been corrupted. In this case Chuck should also be offered a chance to buy the content

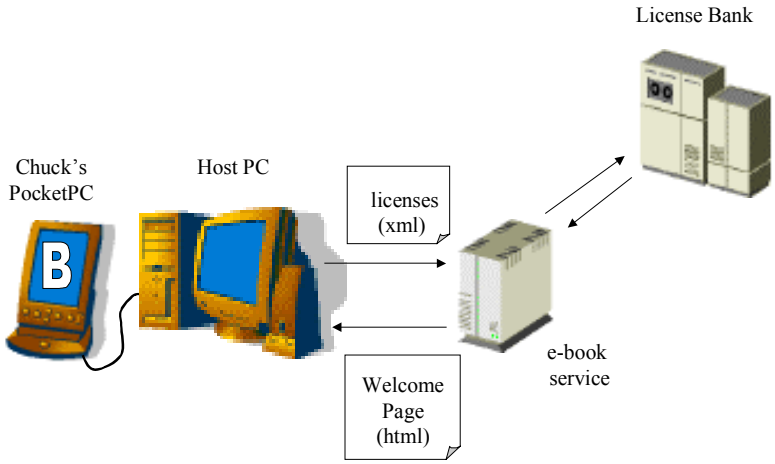

Figure 4: When Chuck docks, the licenses on his Pocket PC are read by the Library Console application running on the Host PC and passed to the e-book service. The service then checks each license against the stored licenses in the LicenseBank, and returns a welcome html page

The service returns a welcome screen such as that shown in Figure 5, giving Chuck an opportunity to purchase any new books. If Chuck decides to buy the book, he proceeds down a fairly conventional e-commerce web-site purchasing routine, and purchases a permanent license for the book. That license is then deposited in the LicenseBank as a permanent record that Chuck owns that particular book. The license is shown in Figure 6. As a payment instrument he can use a scalable payment system. This is a product called "Jalda" [2], as illustrated in Figure 7. The act of paying also triggers a reward to be made to A, for acting as part of the ad-hoc sales force.

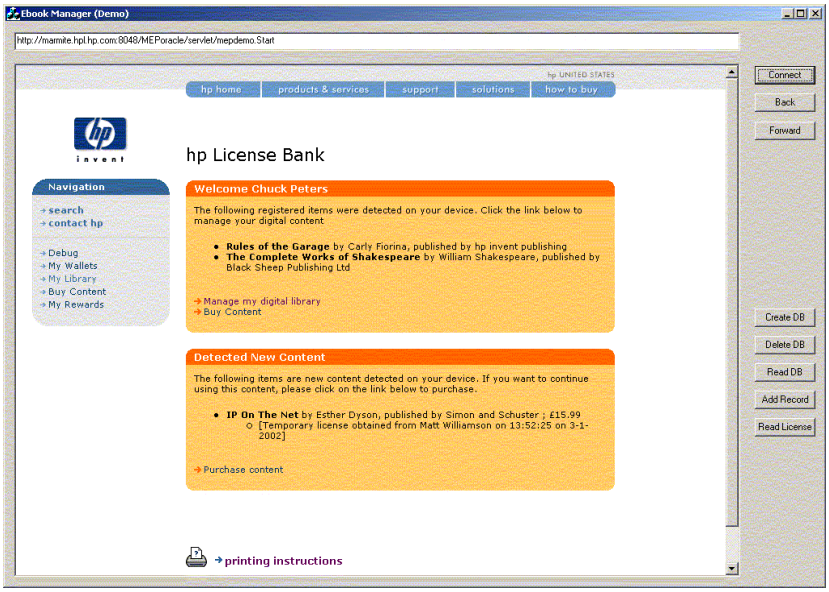

Figure 5: Screen shot showing the welcome screen, giving the user the opportunity to purchase new books.

```
<license>
        <ceoid>12344321</ceoid>
        <doi>ISBN:123</doi>
        <title>IP on the Net</title>
        <author>Esther Dyson</author>
        <publisher>edventure.com</publisher>
        <from>hp e-book service</from>
        <status>Permanent</status>
        <filename>iponnet.rdr</filename>
        <date>Thr, 3 January 2002 13:54:22+0000</date>
</license>
```

Figure 6: Chuck's new license after purchasing the book he originally received from Matt.

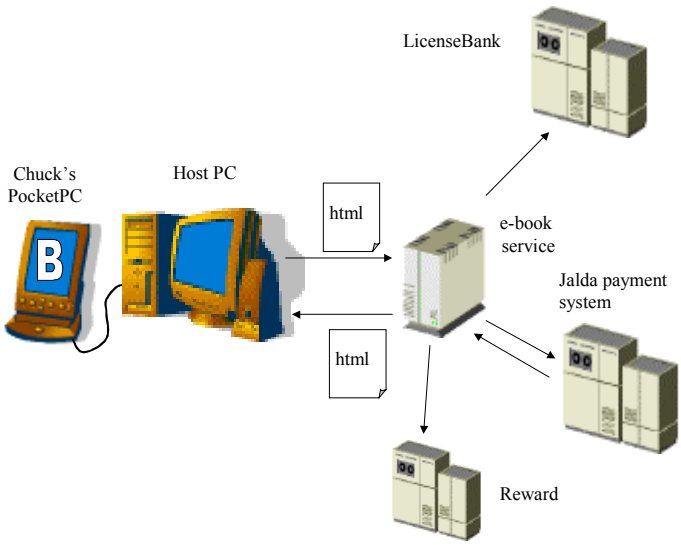

Figure 7: When Chuck decides to buy the book, using his browser, the new license is deposited at the LicenseBank. The payment is handled by the Jalda payment system. The originator of the book (the person that beamed it to Chuck) is given a reward for further distributing the book.

After paying, Chuck gets an opportunity to manage the books on his device, and view all the books that he owns on the MyLibrary screen of the service, shown in Figure 8 . Pressing the

"Update PocketPC" button causes all the selected books to be updated or downloaded to his device. This can mean a combination of updates to licenses, and copying of books, as illustrated in
Figure 9. If Chuck does not pay, then the book will be deleted from his device.

Since there is no encryption in the system, there is little to stop Chuck from cheating, by copying the book onto some external storage for example. However, the premise of this system is that by reducing the barrier to usage of legitimate systems, one might hope that cheating the system would not be worth the hassle. Repeat offenders could be sanctioned in other ways, for example banning them from the service.

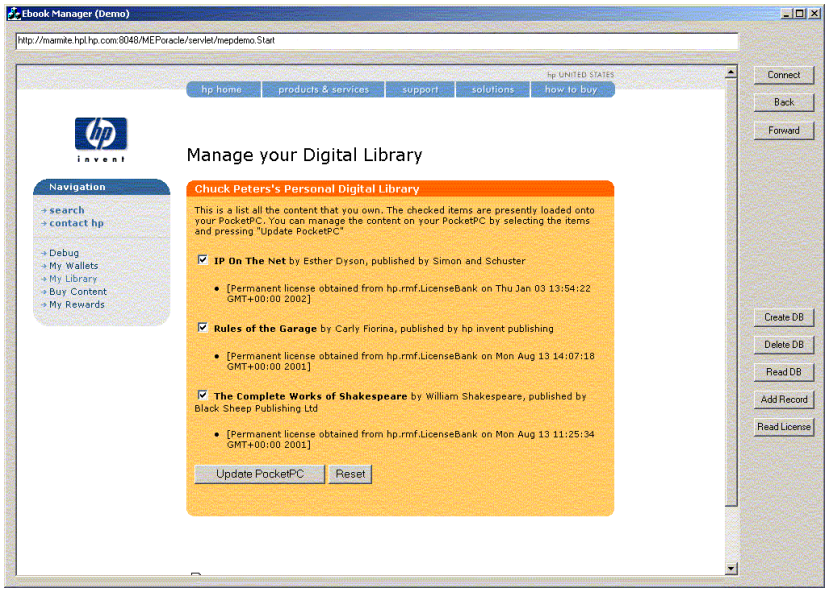

Figure 8: The library screen. This allows the user to manage the books on his device loading those books with check marks onto the device when the "Update PocketPC" button is pressed

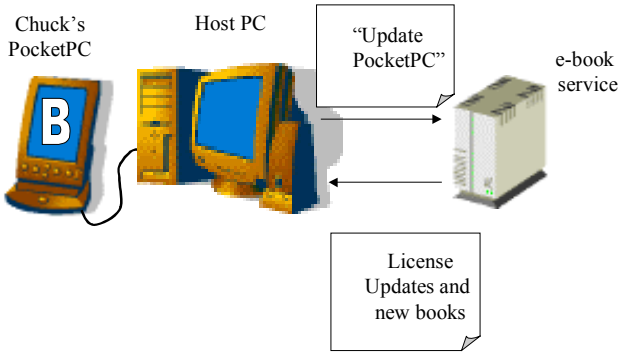

Figure 9: When the "Update PocketPC" button is pressed, the licenses on the PocketPC are updated, and any new books are copied down to the device.

Some time later, Matt might dock to recover his rewards, as shown in Figure 10. He will connect to the service in the same way that Chuck did, with the licenses on her device being compared against the LicenseBank. He can claim his rewards and use them to purchase more content in the online shop. He can of course pay for this content using the Jalda payment system.

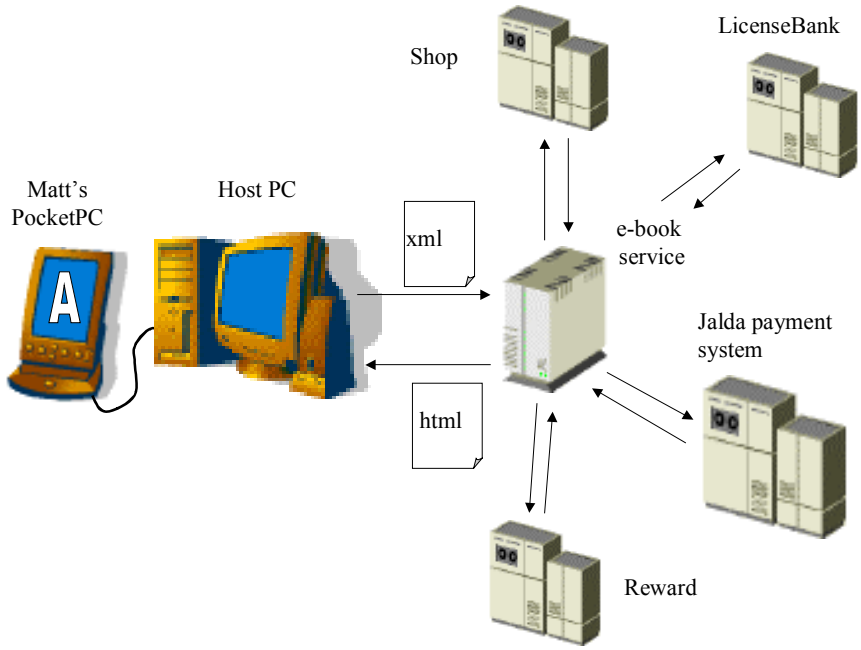

Figure 10: When Matt docks, the same process occurs, with the licenses being taken from the Pocket PC, and compared against the LicenseBank. Matt can check his rewards, and use them in the online shop to purchase more content. He could also purchase more content using the Jalda payment systemt.

The overall system aims to be a compelling alternative to encryption-based solutions by

- Using peer pressure to encourage the sharing of media and loyalty to the service – through making sharing easy between devices, and rewarding users if they pay.
- Making the whole system easy to use – by connecting to the service automatically on docking and by making payment easy by pre-registering details
- Using a service model for content – the service provides a shop and could be extended to offer more services.
- Having a means to express rich products – by using a flexible internal representation of licenses, storing those licenses in a LicenseBank
- Having a means to support flexible payment – by using a scalable payments system.
- The following sections provide more technical information and background to these five aspects.

3 System details

This section examines each aspect of the system in more detail.

3.1 Transferring e-Books between Pocket PC's

The user interface for the application used for sharing books on the Pocket PC is illustrated in Figure 1. A list of all the e-books held on the P/PC is displayed. E-books that have been fully paid for are marked with a green tick, otherwise they are marked with a yellow tick.

The send and receive buttons are used to transfer e-books between P/PCs. To make a transfer the first user highlights the book they wish to send and presses the send button. The second user then clicks on the receive button. This process establishes a socket connection, over infrared, between the two P/PCs. During a transfer the book, and a temporary license are transferred (license shown in Figure 3). The license contains the time when the transfer took place and details of the first users original license. This helps the e-book service track the movement of e-books, which is beneficial from a marketing perspective (e.g. to reward the first user with spending tokens) and for detecting fraudulent behaviour.

On the second P/PC the license and book are stored. The recipient of the book can immediately read the book, but in the present implementation is prevented from sharing the book further. The licenses are stored on the P/PCs using the CE database. Licenses are not stored as native XML, but are readily converted to and from XML through an adaptor. XML is used only for the transfer of the licenses.

3.2 Connecting to the e-book service

When the Pocket PC is docked with the host PC, a "Library Console" application is automatically started. Figure 5 and Figure 8 show the console display. The library console is automatically run when the Pocket PC is docked with the desktop; it largely intermediates between the P/PC and the e-Book service. The library console application connects to the P/PC through the *Rapi* interface, which allows it to read from and write to the P/PC license database. The console also contains an embedded web browser, which serves as the user interface to the e-book service. Some additional buttons are available which are used for testing purposes.

The Library Console functions are as follows

a) On starting: to read the licenses on the Pocket PC, and send them to the e-book service. This is accomplished by sending an HTTP POST request, containing the XML license file, to the e-book service.
b) Thereafter, the user interacts with the e-book service using the embedded web browser.
c) When the user completes using the service and updates their device, the console has to manage the synchronization of the device contents. This includes deleting licenses and books, updating licenses, and copying new books and licenses onto the P/PC.

When the user disconnects their P/PC the library console application automatically closes.

3.3 e-book Service

The e-book service is implemented as a database backed web site, and is implemented in Java servlets running on Jakarta Tomcat. The database used is Oracle 8i, which is accessed through JDBC. A web site was chosen at it was an easy way to create a user interface for the License Console. The service implements has the following functions:

- Process the licenses from the Pocket PC to check their validity. It does this by submitting them to the License Bank (described in the following section).
- Provide a series of web pages making up a payment sequence (select what to pay for, how to pay, confirmation etc.)
- Provide a web page to manage the users library of content, allowing the user to choose what books they would like on their devices, and download the required books and licenses. This screen is illustrated in Figure 8.
- Provide a way for a user to enter and change their registered payment details.
- Provide a place where the user can view their rewards.
- Provide a shop where books can be bought. This allows users to get books both from each other, and from more conventional sales.
- Provide a means for communicating license updates back to the License Console, when the user wants to update his PocketPC.

3.4 LicenseBank

The LicenseBank is used to store the record of ownership for each user of the e-book service. The ownership is represented using an electronic contract, and these electronic contracts are stored by and accessed from the LicenseBank. There are two main functions for the LicenseBank, firstly to check the licenses from the Pocket PC to see if the user actually owns the books that he has on his device, and secondly to allow a new contract to be lodged after it has been bought.

The following sections describe in more detail the conceptual framework used to represent the contracts, the actual implementation used, and some implementation details for the LicenseBank itself.

3.4.1 e-Contract Framework

The model used for defining the contracts is derived from the one described in cite [3]. This abstracts a contract into essentially three parts

> *The parties involved – who is signed up to the contract. This is defined as a rolename (a local name for the contract), and the real name of the party.*
>
> *The about items – what is the contract about.*

The promises – the detailed dynamics of the relationship defined by the contract. These are split into two types, rights (which confer a right from one party to another) and obligations (which must be fulfilled). The interaction between rights and obligations (e.g. what rights are available if which obligations are fulfilled) is the dynamics of the contract. The framework allows for the promises to contain state or data that records the particular state of a promise. This state can be updated when actions are taken, and queried to determine what rights are available etc. The participants in the promises are specified by local rolename, which resolves to the actual party through the parties definition.

Referring to

Figure *11, a contract for the e-book service might be between a Customer "Matt Williamson" and a Service Provider "hp e-book service", be about "book where title=IP on the Net" and have two promises: An obligation on the Customer to pay a one off fee, and an ownership right dependent on that obligation being fulfilled.*

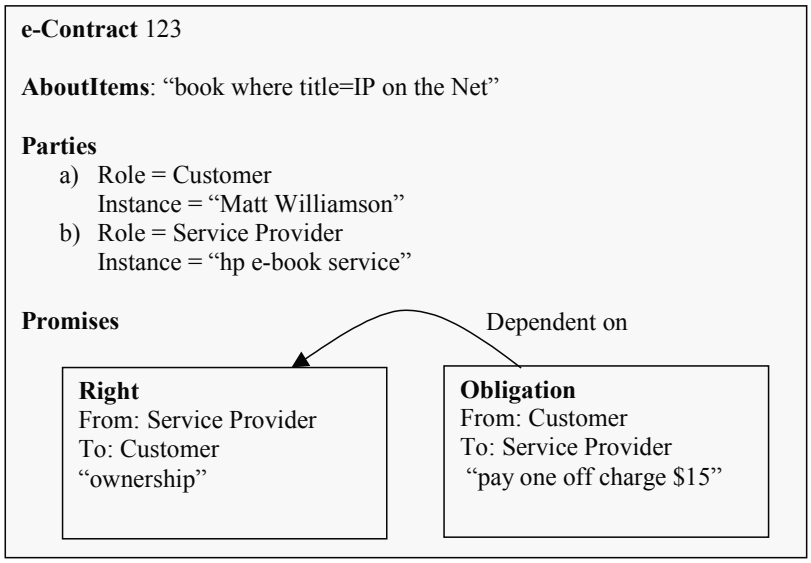

Figure 11: An example electronic contract representing the ownership right of a Customer of the book "IP on the Net" if they have paid a one off fee. The contract is formatted with respect to the e-contract framework, specifying the parties, about items and promises. The promises are defined with respect to the role names defined by the Parties.

This simple example shows the framework, and shows the type of contract used in our implementation of the e-book service, but does not show its flexibility. The definitions of the parties, about items and promises all allow a variety of contracts to be cleanly expressed. This flexibility is discussed in the following section.

3.5 e-Contract Implementation

Each concept in the framework is modelled as a class as shown in Figure 12.

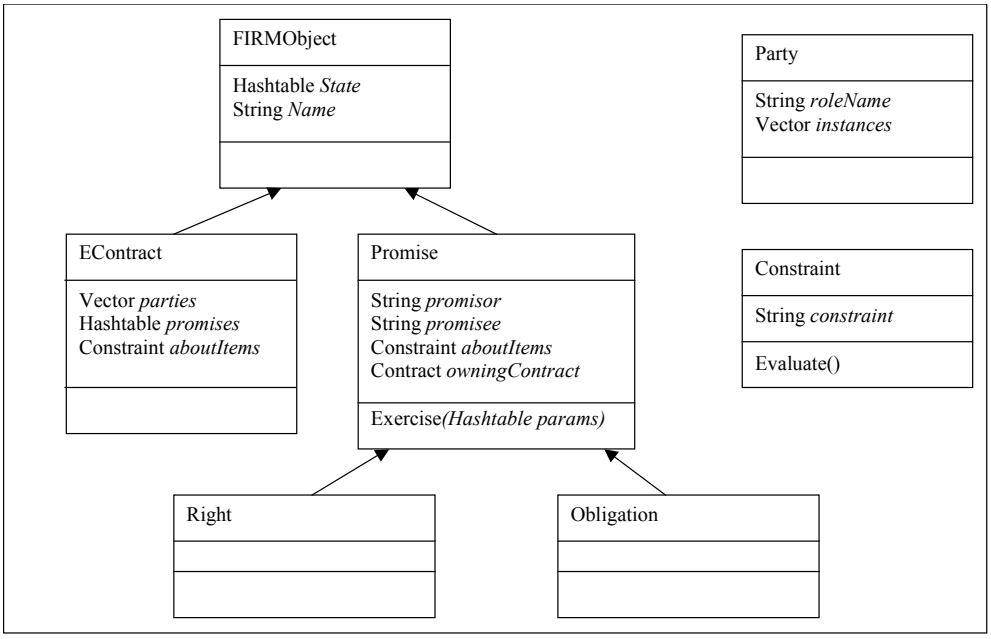

Figure 12: This figure shows the class diagram of the e-Contract model. The root object is the FIRMObject, which includes a unique name and a Hashtable of name-value pairs specifying the state. This is extended by the EContract class that records the parties, promises and about items. The Party object defines the rolenames used within the scope of the contract, and the identity of the people signed up to play those roles (instances). The Promise class also extends FIRMObject, adding references to promisor, promisee, and any subset of about items. It also includes a back reference to the owning contract and an exercise method that is used to update the state. The Promise class is itself extended by the Right and Obligation classes. The Constraint class is used to model queries.

The root class is **FIRMObject** which contains a hashtable of name-value pairs that define the state of each object. It also defines a unique name. Extending **FIRMObject** is the **EContract** class, which contains parties (a Vector of **Party** objects), promises (a hashtable of **Promise** objects, indexed by promise name) and aboutItems, defined as a **Constraint**. Each Party consists of a String specifying the roleName of the party, and a Vector of instances (references to people) signed up to be members of the role for this contract. This allows contracts to be drafted in abstract terms of roles, binding in the actual participants later.

The **Promise** has a String promisor (the giver of the promise) and promisee (the receiver), both references to the roleNames defined in the parties. Promises are extended by the **Right** and **Obligation** classes. Promises extend the **FIRMObject** class and so have a name and hashtable of name-value pairs for state. This state is used to record the state of the Promise – e.g. how much of a right has been consumed, how much payment is required and how much has been made etc. This state is altered using the exercise() method, which is called with specific parameters, dependent on the domain in question.

For example, to model a simple payment obligation, the obligation may have two states `amount_to_pay` and `amount_paid`. Exercising that obligation might require an extra parameter `payment`, and calling this method would increment `amount_paid` by the `payment`. The obligation thus holds its present state. A right which is dependent on this obligation might have state to record how many times it was exercised, and the code in the exercise method might check the states of the payment obligation to ensure that `amount_paid >= amount_to_pay`. Each promise is allowed access to the other promise's state, within the scope of the contract. For this reason, the **Promise** contains a back

pointer to the contract (owningContract). The exercise method also checks whether the caller is allowed to call the method (it checks that the promisee matches a role name in the parties, and that the caller of the method is registered with that Party).

The aboutItems are defined as a **Constraint**. This casts the about items as a query, for example "all books where title=The Complete Works of Shakespeare". The inclusion of any particular item can be checked by calling the evaluate method on the **Constraint** object. This allows the about items to be flexibly defined to be single items or arbitrary subsets e.g. "albums where artist=Rolling Stones" or "books where publisher=Wiley" or "books where ISBN=12346789".

To implement a new form of contract, the process is as follows

1. Decide what roles will be involved in the contract
2. Decide what each role will promise the other roles, and split these promises into rights and obligations
3. Decide what state needs to be saved for each promise
4. Extend the Right and Obligation classes, and write the code in the exercise method to implement the interactions between the promises, by accessing the state of these other promises. The authentication aspect is handled by the superclass.
5. Make a contract class and add instances of the rights and obligations as required. Add the party classes, and record the actual parties involved.

This contract model has been used to model a variety of different contracts, ranging from various consumer related models (pay per use, subscription, present proof of purchase, unlimited rights, limited rights), as well as to represent warranties on electronic equipment and access control into parts of xml documents. For all these cases, all that was required was writing the code to implement the dynamical relationship between the rights and obligations. We have also demonstrated how one can write a simple language, which will compile to generate this code.

In summary, the model for electronic contracts given above has the following properties

- It allows the expression of arbitrary complex interactions between rights and obligations, and so can model complex contracts.
- It is a dynamic model, because the state of each promise can be updated and queried, giving an up to date picture of the overall state of the relationship/contract
- It is flexible, because the model represents the generic structure of a contract within which domain specific information can be easily represented. This means that a variety of different contracts can be represented and managed with the same machinery. In addition, the use of constraints to model about items gives flexibility allowing arbitrary subsets of items, and even items that do not exist (e.g. "issues of the economist where publication year = 2002") to be specified.
- Each contract defines a unique relationship between the parties within the context of the about items. This means that each customer can have a

customized contract, but that since they conform to an overall framework, the management of that complexity is made easy.

3.5.1 LicenseBank

Given a representation of the contract, all that is required is a method to manage, query and manipulate that representation, and in the context of the e-book service provide a repository and record of what users own which pieces of content. The code used to fulfil these functions is called the LicenseBank.

The LicenseBank is a repository for the electronic licenses, handling persistence of those contracts. It has methods allowing contracts to be lodged and removed from the repository, and methods to allow interaction and querying of the contracts. The LicenseBank is implemented as a Java class, with the contracts stored serialized as XML in an Oracle database, accessed through JDBC. The API for the LicenseBank is defined as:

```
public interface LicenseBank {


  /**
   * Lodge a contract in the LicenseBank and
   * issue a new identifier
   */

  public String registerContract(Contract offer,
                  EPersonRef actor);

  /**
   * returns a contract by id
   */

  public Contract getContract(String cid);

  /**
   * exercise - exercises rights or obligations on a particular
   * contract
   */
  public LicenseBankResponse exercise(EPersonRef actor,
                   String cid, LicenseBankParams params);

  /**
   returns all the contracts that this actor currently holds
   */

  public Vector getContracts(EPersonRef actor);

   /**
    * returns the ids of all the contracts that this actor
    * currently holds
    */
   public Vector getContractIDs(EPersonRef actor);
}
```

Figure 13 The registerContract() method takes a contract and inserts it into the repository, returning a contract id (cid). The contract can then be extracted using that cid with getContract(). Exercising a particular promise is done with the exercise() method, which takes as parameter a ***LicenceBankParams****, an object carrying the promise name and any extra domain specific parameters. Exercise() returns a* ***LicenseBankResponse****, another object which wraps a boolean response, and an explanation. For example "promise exercising refused because actor is not authorized". In all the methods an* ***EpersonRef*** *object is used to carry the actor's identity.*

The LicenseBank is used for all aspects of the e-book service relating to the ownership of books. When a new contract has been bought, it is registered with the LicenseBank using the registerContract() method. When each license on the Pocket PC is being checked against the

LicenseBank, getContracts() method is used to pull out all the contracts that the user owns. These can then be checked using the exercise() method, passing in parameters such as the metadata of the book. If the exercise() method returns, then the book is owned, and if not, then the book is not owned, and the e-book service will request payment.

3.6 The Payment Server

The payment server is used to handle payments for the e-book service. It acts as a third party and manages payments from the customer to the e-book service provider. The e-book server acts as a proxy between the customer and the payment provider. If the customer is able to successfully authenticate them self to the payment provider and pay for the content, then the e-book service will release content. The payment server can thus aggregate payments and present one bill to customers, covering a variety of different merchants. This arrangement has the advantage that customers can make purchases without disclosing their details to merchants, who might abuse this information (by for example tracking purchases and selling that information). It also overcomes the overhead costs associated with small value transactions.

The Jalda (micro-payment) server was selected for the demonstration because it had many attractive features other systems did not provide [2]. It can handle both micro-payments and large value credit card type payments, it supports digital signing of contracts for non-repudiation purposes, and it provides a web interface which customers can use to log in and view their account details.

The system used for the e-book service used a pre-pay mechanism. Customers pay an amount into their Jalda account, which is then debited whenever they purchase a book.

The e-book service uses the Jalda API to interact with the Jalda payment server. The calls are used to initialise a Jalda session, setting up currency details, and then used to deduct the actual cost of the book from the users Jalda account. An outline of the interaction following initialisation with the payment sever is shown in Figure 14. The detailed process for setting up a session and carrying out a purchase is as follows:

1. The *JaldaSession* API call is used by the merchant (e-book service) to set up a connection with the payment server (jalda). The payment provider sends a contract (string) back to the e-book service. This contract is not the same as the contracts that are used to represent book ownership, rather they are the contract between the merchant and the payment provider about the terms of the payment session. Theses contracts are not currently standardised, although it is likely in future that legally binding contract templates will be framed to simplify deployment.

2. The e-book service signs the contract on behalf of the consumer using the customer's private key held by the license bank. The signed contract is returned to the payment provider for authentication. Authorisation to use the private key is provided by the user typing in a PIN number. This is the approach adopted to allow Jalda to be used to make micro-payments from mobile phones.

3. If the signed contract is valid, then the e-book service will be able to charge the users micro-payment account. To do this, the e-book service calls the *Tick* API. To

allow multiple ticks to be sent in one call, the API accepts the number of ticks to send as an input parameter. The real value of a tick, is held at the payment server. During connection set-up, the real value of a tick is placed in the contract sent by the payment server to the license bank. The payment server is responsible for conversion to a real value.

4. The session is terminated using the *stop* API call.

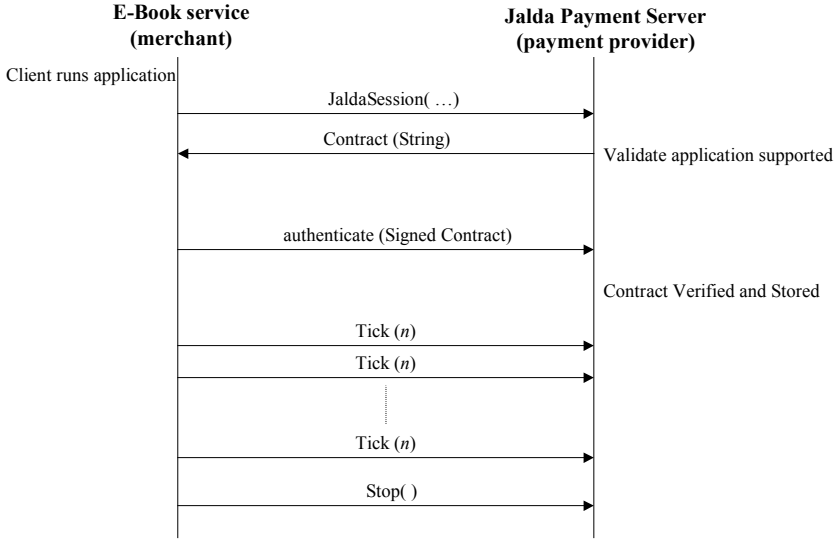

Figure 14: Interaction between the E-Book Service and Jalda Payment Service. The merchant (in this case the e-book service) initiates the connection by requesting a Jalda Session. After handshake and authentication messages have been exchanged the session is live. The merchant charges the payment provider by making "tick" calls that have a monetary value defined in the initial setup. The merchant can make any number of tick charges, and halts the session with a Stop command.

More details of the Jalda API can be found at [2].

4 Conclusions

The biggest challenge presented to the media industry by the Internet is a fundamental change in business model. Media is too easy to copy perfectly, and it is too cheap to distribute those copies widely for business models based on direct sales to be realistic. Attempts to prevent copying and distribution are technologically difficult and likely to fail. The alternative business model is service-based, using content to attract users to the service, where revenue can be obtained from advertising, some direct sales, branded merchandise, and subscription.

The response from some technology companies had been to offer technological solutions to enable direct sales (e.g. stop media being copied or distributed) mostly though encryption. While these technologies might be needed as an interim measure, we believe that alternative technologies are needed to support service-based models.

These technologies are: mechanisms to encourage people to share their media, so generating interest, mechanisms to encourage people to connect to the service, and technologies to create the service itself, particularly ways to represent the kind of flexible product offerings that are enabled by digital goods, and flexible ways to charge for those goods.

In this paper we have presented a system that embodies these technologies, consisting of an application for the PocketPC that encourages sharing of electronic books, and applications for the client and the server that encourage users to pay for their books based on usability and peer-pressure rather than strong encryption. We use a flexible means of representing the digital goods, expressing them as digital contracts, and use a scalable payment system.

Our system is implemented and has been used by 3 users sharing over 10 books. The system is somewhat limited in its ability to scale due to design decisions both in the design of the service web pages (they are better at showing 10 items than 1000), and also due to design decisions taken about how the clients and service communicate. There is no reason in principle why these problems could not be overcome.

In conclusion, while the media distribution mechanisms of the future might not look exactly like this system, the fundamental pieces – flexible definitions of ownership, flexible means of charging, and the use of a variety of mechanisms to obtain legitimate behaviour are likely to be critical for developing service models for media consumption.

A Appendix A: Analysis

This appendix reviews recent history in the media industry, and also discusses some of the technology offerings. It makes the point that most of the challenges faced by the media industry over their adoption of Internet distribution are fundamentally about business models and not technology. The appropriate technological solutions are ones that enable flexible service-based models. The appendix also argues that these include flexible payment mechanisms.

A.1 Copying and sharing

Digital media is easy to copy and furthermore each copy is an exact one, with no loss in quality. This fact, together with the increasing availability of compact file formats for various media types (pdf, mp3, divx) have made the sharing and copying of digital media more and more popular.

Over history the various publishing industries have been "attacked" by technical advances that increase the ability to copy works and reduce the control of these companies over their products. Some examples are the lending library, photocopier, video cassette, audio tape etc. The Internet has added a further dimension to this attack which is the ease of distribution of copied media. Systems such as Napster [4] (a peer-to-peer music sharing system presently being sued by the RIAA for copyright violation) shows some of the potential of this distribution.

By rights, the ease of distribution and sharing should be embraced by media companies wholeheartedly since the value of any of their products is determined by how popular they are [1]. This means that for media companies, allowing free sharing and distribution of media is a sensible step, because it increases the value of the product, while at the same time not incurring any extra distribution or marketing costs.

However, this approach requires a change in business model; it will be difficult for media companies to control and charge high prices for digital media directly and begs the question of what alternative models could be used.

A.2 Service models for content

The main alternative appears to be to use content to attract consumers to a service from which other forms of revenue can be derived. In order to be of value, service models must provide a service! What exactly these services could offer is still the subject of conjecture.

Some possible ideas are to offer a universal jukebox, with any piece of media available at any time (for example Napster for music), or offer storage of media for consumption on different devices (for example mp3.com). Other ideas are transcoding between different formats, cataloging, search, recommendations, metadata management etc. The service offering might extend beyond the media itself into developing a community, perhaps offering email, instant messaging, chat rooms etc. It might also be possible to offer unrelated products such as financial services e.g. [5].

A.3 Media industry response

The initial reaction of media companies to the nature of the Internet has not been to embrace service models but to attempt to force digital media to behave like a physical product (i.e. awkward to copy and share). This allows them to apply the same business model for both physical and digital products: largely transactional revenue with high prices for media. They have also used legal action to attempt to tamp the flow of copied media, shutting down both Napster and mp3.com.

The technologies that they are choosing are based on encryption, locking the content with a key, and giving that key to the consumer as proof of ownership. These systems with a variety of levels of sophistication are marketed by InterTrust, Adobe, Microsoft and a host of smaller companies [6]. To get the maximum protection from these technologies, the systems needs to incorporate dedicated trusted readers (so that the media cannot be grabbed at the screen/speaker after it had been decrypted), it needs to use a different key for each user (so that if a key is compromised only one copy is lost, not all of them as happened with the DVD format in 2000 [7]), and strong encryption needs to be used (so that the keys are difficult to crack).

Unfortunately these demands make the use of these systems expensive both to setup (perhaps by subsidizing trusted readers such as the Rocket e-book reader) and to maintain (handling all the key management issues, upgrades, etc.) The encryption software is inherently complicated and by its nature restricts consumer behavior, making the need to handle customer complains and technical problems inevitable, with further cost.

In addition these systems generally have no fall back plan once something leaks[1], which from recent experience also seems inevitable, given the number of people interested in breaking these systems with a variety of motives. Stephen King's e-book ``Riding the Bullet'' was cracked within a week of publication [8]. Once the content leaks it has gone, and relying on this technology to prevent leaks is not realistic.

Another important drawback is that the encryption generally binds the ownership of the media to the particular device used to consume it. This is somewhat brittle to equipment failure and upgrades, and is also not how users are used to consuming media. Since digital media by its very nature is so easy to copy, an advantage of being digital is the ability to consume it on all sorts of different devices and restricting this seems to immediately create a poor customer experience. It is also relevant that locking the content in this way does not say much for the trust of the customer by the media companies.

If these technologies have such strong disadvantages, why are they viewed as the only hope? There are at least two reasons for this, the first coming from people who do not believe that there are any ways to generate revenue without restricting the content, and secondly there is the fact that most or all media companies have a large retail channel, which they do not want to immediately undercut.

This second point explains why Bertelsmann and Napster are engaged in producing a service model for content, protected by InterTrusts encryption mechanism, due to start in 2001 [9]. It also explains the tentative moves by other media companies into making their content

[1] Although tracking systems exist for searching for watermarked content e.g. http://www.digimarc.com/

available on the Internet through organizations such as PressPlay.com and musicnet.com. This transitional phase is a difficult one, and one where we would expect some progress over the next few years.

A.4 An alternative view

An alternative view is that companies should exploit the new freedom of the Internet to actually encourage sharing and distribution of media. They should move from predominately transactional revenue to a mixture of service and transactional, using their content to attract users to compelling value added service. These services could provide rich and flexible products, for example bundling physical and digital products, offering live special events, concerts etc., chat rooms, playlists of the rich and famous etc. Providing these rich service based offerings will require a way of specifying and managing the relationship between the consumer and the service provider. Since the encryption technologies are inherently unreliable, relying solely on them is not a good strategy. We believe that a variety of options are available to obtain the required behavior of consumers (consuming lots of media and paying for it) without locking the content up.

The required tactic is to move from a single line of defense (the encryption technologies) to a range of softer defenses, all working together to achieve the same end. For example, media companies could provide valuable and useful services and ways of owning content, they could exploit peer pressure, they could provide services that were easier to use, or more fun, or faster than illegal methods of obtaining content.

A.5 Getting paid for Content and Services

Content owners to date have found it difficult to charge their customers for access to online content. One likely reason for this has been the proliferation of free content portals that make some or all of their revenue from advertisements. This is an example of an industry suffering from overcapacity and little product differentiation. Over time, as many of these portals fail and/or are taken over, it will become easier for the remaining sites to begin charging for content. In addition, increasing scepticism about web advertising will likely force companies to obtain a greater proportion of their revenue directly from their customers. For example 11 out of the top 100 sites now have fee-based tiers; up from 6 the previous year [10].

This section describes the pricing policy and payment collection options available to service providers.

A.5.1 Pricing Policy and Payment Instruments

A clear distinction should be made between pricing policy and payment instrument. A decision to charge for items on a per item basis is one of pricing policy. The payment instrument is used to collect payment and is a somewhat separate choice. Clearly these two aspects of the system are related since the cost of collecting payment to a large extent determines which pricing policies are viable.

In this section two popular pricing policies are considered; pay-per-item and subscription charges (pay-per-item can also be read as pay-per-unit-time). Pay-per-item pricing has

received a lot of interest over the last few years because of the ease with which digital content can be disaggregated and sold in pieces. For example, a music album can be separated out into its constituent tracks that can then be sold individually. The almost zero cost of digital content reproduction and consumer demand for lower prices has encouraged this trend.

The position of the service provider within the value chain from content creator to consumer also influences pricing policy. In the next section, therefore, the issues affecting pricing are considered for two different types of organisation; a content owner such as a music portal and a telecomm operator.

A.5.2 Pricing Policy for Content Owners

Per-item pricing has been suggested as an attractive charging mechanism for digital content. There are however, several drawbacks to this approach. Firstly disaggregation will enable customers to cherry pick the most valuable content and ignore the remainder. Using the example of a music album, it is unlikely that the content owner would be able to charge sufficiently extra for the most popular tracks to achieve the same level of profitability as selling a complete album. In addition, the price-setting activity for a large catalogue of content will be costly both in terms of management and usage metering. For example, although menu costs (the costs of changing prices and informing customers) are lower on-line, evidence collected to date suggests that the size and frequency of price changes are much smaller than expected [11]. Finally, the cost of collecting small payments is likely to be more expensive than at first appears, see section A.5.4.1.

From the consumers perspective per-item pricing is confusing. Evidence suggests that pricing models tend toward simplicity over time [12]. Bundling of TV channels in cable also subscriptions supports this view. Customers also seem to have a preference for all-you-can-eat pricing models.

For content owners, subscription charges are likely to be more attractive. There are several examples of this happening in the area of music portals (e.g. [9,14]). The Wall Street Journal has charged subscription fee from its inception. For the content owner, subscription charges are attractive because

- They provide payment ahead of delivery, improving cash flow.
- They avoid the administrative cost of pricing content on an item-by-item basis.
- They are easier for customers to understand.

The ability of a content owner to charge a subscription will be dependent on their ability to create a valuable service offering. Ultimately if the service does not justify a monthly or quarterly charge, in excess of the minimum required for a credit card transaction, then the service simply does not provide sufficient customer value. It must be remembered that free content, especially low value content will continue to be available.

From a consumer's perspective, subscription pricing commits them to a larger upfront payment. This can be unattractive especially when subscribing to a new service provider and can lead to inertia. More critically, if every content owner attempts to charge by subscription, the aggregate effect could be self-defeating. However, increased aggregation of content, by fewer service providers, will likely overcome these effects.

A.5.3 Pricing Policy for Telecomm Operators

Telecomm operators are in a substantially different position primarily due to their positioning in the content delivery channel. Customers are already very familiar with regular bills from their telecomm providers, which is very much like a subscription charge. In contrast to content providers, telecomm operators are also familiar with charging for small units of time and aggregating the prices into a monthly bill. So for telecomm operators, per-item charging is much more likely to be cost effective.

A good example of how such an approach could work is provided by the i-Mode service operated in Japan by NTT [13]. NTT through its i-Mode service has positioned itself as a content/service aggregator. Customers connect to the service providers through NTT who is then able to meter the traffic form each service provider to each customer. A dedicated billing system was developed to manage this. At the end of each billing period, NTT bills each customer based on his or her usage. NTT also sends payments to the service providers less a transaction handling fee. Effectively i-Mode aggregates the payments due to each service provider across all customers, and aggregates the amount owed by each customer over all service providers.

In general telecommunication providers are able to measure QoS and packet level usage at relatively little cost and are also able to add charges on to their customers monthly bills. This enables them to efficiently implement per-item charging.

A.5.4 Payment Instruments

There are many payment instruments that consumers can use to pay for goods and services. For every physical payment instrument, one or more on-line analogues have been developed. Detailed discussions of these are available in [15, 16]. This section describes the key features of each class of payment instrument.

A.5.4.1 Micro-payments

The trend towards lower prices for digital content has motivated the development of micro-payment systems. Micro-payment systems aim to make sub $1 transactions profitable. Most of these systems either mint a new digital cash currency or, piggyback the existing payment infrastructure [15, 16]. In the latter approach many small value purchases are aggregated into a single larger payment that can then be collected efficiently, usually with a credit card or as part of a regular monthly bill.

It has already been noted that telecomm operators are in an excellent position to be able to collect micro-payments. However, for content providers to do this themselves, there are several hidden costs. For example, many systems require the use of digital signatures using PKI that are difficult and costly to deploy and maintain. In addition, the payment systems are unfamiliar and complex for consumers to use. This will increase marketing and support costs. At a more fundamental level the cost of a content-based micro-payment system will always be higher then a subscription based approach. This follows because the process of disaggregating content also disaggregates the payments stream, which the micro-payment system must then aggregate again, a process that can never be costless.

A.5.4.2 Bill Aggregation

Ultimately paying for content services will become as familiar as paying existing utility bills. This suggests charges for content should be consolidated with other bills such as a phone bill. Given the telecomm operators unique position, it is likely that they will collect payment on behalf of the content providers. In fact this seems to be an increasingly attractive proposition for both operators and content providers [14, 17]. This new role for the telecomm operators will be a major threat to participants in the existing payment infrastructure.

A.5.4.3 Credit Cards

Credit cards have a very significant advantage from being the only internationally recognized payment instruments. For the web, this is a very major advantage. However, not all persons, especially the young own a credit card and many people are still hesitant to use a credit card online. Processing costs are also make charging for low value items uneconomic.

Credit card processing companies can be thought of as bill aggregators, transforming many infrequent or irregular payment streams into a single periodic monthly bill. In this sense credit card companies are in direct competition with other bill aggregators, principally telecomm operators. Competition will likely reduce processing costs.

A.6 Summary

The analysis can be summarized as follows:

- The revenues from content will move from being predominantly direct-sales to a more service based structure, with revenue from advertising, subscription, branded merchandise and other mechanisms.
- Encryption products that sustain a direct-sales approach will be useful as transitional products, for businesses that have a large retail channel that they do not want to undercut. In the long run, the expense of encryption will not be worth it.
- Digital products can be more flexibly sold than physical ones, so flexible representations are required. They might be customized on an individual level.
- The choice of pricing policy for digital content is dependent on the participant's role. Telecomm operators are ideally positioned to implement both per-item and subscription pricing. Currently, content providers can only realistically charge by subscription. They can effectively outsource payment collection to the telecomm providers should they want to charge on a per-item basis.
- Payment collection for content is probably best achieved by consolidating charges with a monthly bill, most obviously the phone bill. This is a familiar payment mechanism for consumers and is efficient to implement.

References

[1] Carl Shapiro and Hal R. Varian, ``Information Rules, A Strategic Guide to the Network Economy", Harvard Business School Press, November 1988. http://www.inforules.com.

[2] http://www.jalda.com/home/

[3] "A Framework for Interoperable Rights Management" by Martin Roscheisen PhD Thesis, Stanford 1997. Available at http://hci.stanford.edu/rmr/FIRM/

[4] http://www.napster.com

[5] David Bowie themed bank accounts, http://www.mobic.com/news/2000/03/david_bowie_unplugs_bowiebanc.htm (March 2000), http://www.bowiebanc.com/

[6] "Digital Rights Management: Business and Technology" Bill Rosenblatt, Bill Trippe and Stephen Mooney M&T Books (New York, 2001)

[7] "DVD encryption hacked", http://www.cnn.com/TECH/computing/9911/05/dvd.hack.idg/ November 5th 1999.

[8] "Hackers Crack King E-book", http://abcnews.go.com/sections/tech/DailyNews/e-bookhack000331.html December 17th 2000.

[9] "Napster, Bertelsmann To Develop Paid Service", http://www.newsfactor.com/perl/story/4703.html, November 1st, 2000.

[10] O'Leary, M. "No More Web Free Lunch" Online vol. 25, Num. 5, p 81., vol. 25, September 2001.

[11] Tang-Tang, F. and Lu, D. "Pricing Patterns in the Online CD Market: An Empirical Study" The International Journal of Electronic Commerce and Business Media. Vol. 11, Num. 3, pp 169-185

[12] Odlysko, A. "The history of communications and its implication for the Internet", AT&T Labs Research paper, June 16, 2000.

[13] I-Mode NTT http://www.palowireless.com/imode/articles.asp

[14] "Financial gravity bears down on the giants and minnows of new media", http://specials.ft.com/ftit/april2001/FT35PS3W0LC.html, April 4 2001

[15] O'Mahony, D., Peirce, M., Tewari, H., "Electronic Payment Systems", 1997, Artech House

[16] A good survey of payment instruments can be found at http://ganges.cs.tcd.ie/mepeirce/Project/oninternet.html

[17] Pearse, J., "Vodafone finds a way to let teenagers pay" New Media Age, p22(1) Nov 15, 2001