Proceedings of the First International ODRL Workshop (Eds. Renato Iannella & Susanne Guth) 22–23 April, 2004, Vienna, Austria

REAP
A System for Rights Management in Digital Libraries

Øyvind Vestavik
Norwegian University of Technology and Science
Oyvind.Vestavik@idi.ntnu.no

Abstract

This paper presents REAP, a system for rights management in digital libraries. REAP is aimed at demonstrating that intellectual property can be published in the Internet by digital libraries in accordance with copyright laws. The article proposes an architecture/paradigm for managing rights when publishing information through Digital Libraries involving digital rights management. Based on this architecture a working prototype called REAP has been implemented and is documented and discussed in this paper.

1. Introduction

The requirements for a Digital Rights Management system to be used in a digital library are a combination of the needs of the library's users and authors. Authors and rights holders want control over their work and possibly economic compensation for their effort. Information consumers want free access to as much information as possible without any administrative interaction like registering personal information like personal names, email address etc. The library tradition seems now to be challenged by the rise of the Internet as an information medium. Because of the ease of access to information in the Internet, users are beginning to vote out the traditional libraries when searching for information. For libraries to be able to continue their role as a society's source of quality information and a society's memory in a networked environment, libraries have to open up to the Internet and use it progressively to let users access the information and knowledge present in their collections and holdings. This poses radical challenges to the library tradition and the new actors in the library world. Digital libraries must, as far as possible, satisfy the needs of both its users and authors to survive. On the one hand, a too restricted access to information will not be satisfactory for information consumers as they will not be able to get the information they need. In these cases consumers will often turn to other sources of information. On the other hand, authors will not be willing to publish their information unless the digital library is able to handle the rights over the material in the digital library properly. It is therefore in the interest of a digital library to implement systems that preserves the needs of both groups. This means that the digital library must be able to trade with material or rights to material on behalf of its rights holders, providing its users with the material they need in a manner consistent with copyright and without violating the rights of users and authors. Well designed and adequate DRM Systems can hopefully balance the needs of the different patrons in a digital library. As a first attempt to create a DRM system for use in a digital library setting we have created a system called REAP, a Rights Enforcing Access Protocol. This system consists of a rights language to express rights over material in the digital library and a server side software that should be able to control that access to digital material published in the digital library is granted or denied based on rights description for the material expressed in the rights expression language. The rights language is based on ODRL **[ODRL]** and the software is inspired by a reference architecture for rights management systems proposed by **[Rosenblatt, Trippe, Mooney 2002]**

The rest of the paper is organized as follows: In part 2 the basis of REAP in terms of entities and concepts is discussed. Then, in part 3, a suggestion of how REAP fits into the larger setting of a digital library is given. Next, in part 4, the REAP software and what it does is presented before in part 5, the motivations and choices for a rights expression language for REAP is explained. Finally, in part 6, the properties of REAP is discussed.

2. The Basis of REAP

REAP **[Vestavik2002]** is an access protocol and is aimed at controlling users access to information resources on a server. It is not intended to support end-to-end chain services. As such, REAP can be seen as a part of the services a repository of a digital library offers its users. The architecture of REAP is based on a defined set of logical entities as shown in figure 1 below.

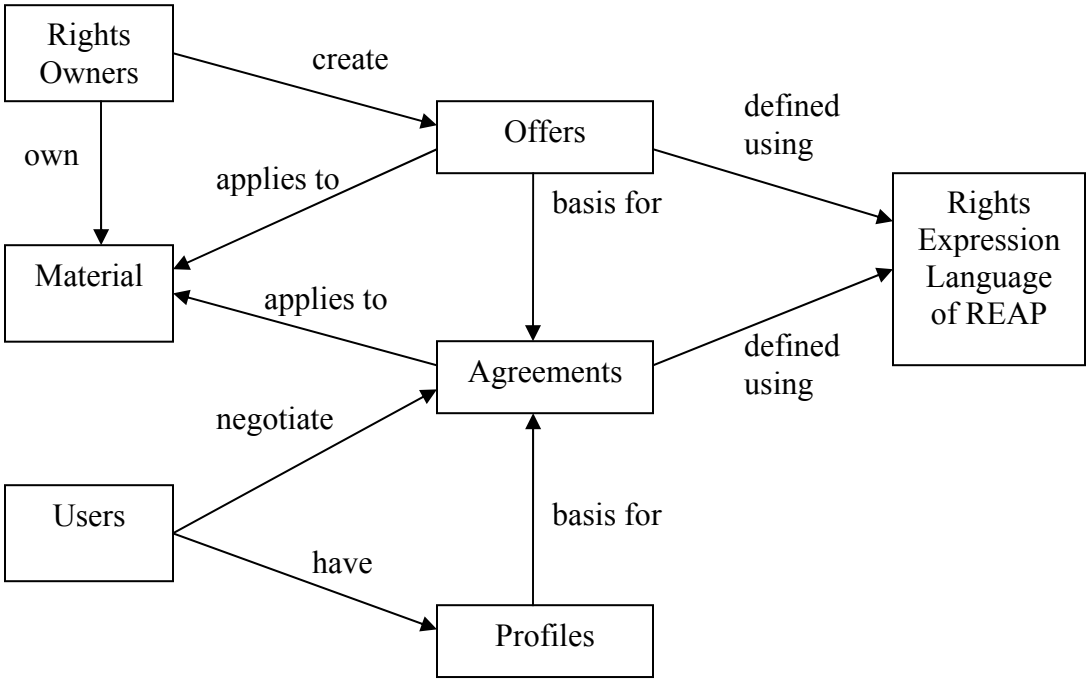

Figure 1: *Logical Entities in REAP*

Rights owners create rights offers for their material giving the rights that can be granted to users, the requirements to be fulfilled by users in order to be granted those rights and the constraint limiting the extent of the rights. *Note that conditions expiring rights is not supported by REAP*. Users have to register with the system to be able to create an agreement giving them access to rights over the material in the library and during registration a profile for the user is created. Both Offers and Agreements are expressed in an application specific rights language based on ODRL.

3. REAP and ADEPT

The proposed model/prototype is concerned with rights management in connection with retrieval of material. All other functions of a digital library is the responsibility of components and systems that are outside the scope of the prototype. For practical reasons all functionality normally associated with digital libraries is assumed to be carried out in Alexandria Digital Library (ADL) / Alexandria Digital Earth Prototype (ADEPT), a distributed digital library for georeferenced information **[Alexandria]**.

Using an ADEPT client, users can search for information resources as shown in **figure 2**. The client sends a search/query to the middleware component of ADL/ADEPT which distributes the search to distributed collections located around the world. These collections have metadata collections containing metadata formatted for ADL, so-called ADEPT views or metadata for the ADL bucket framework on which the query is executed. The result of the query on the local metadata catalog is returned to the middleware which

returns it to the calling client at the user's location. Based on the responses from all collections that were queried an ADEPT access report presents the overall search results to the user.

The access report contains information about where to retrieve the information resource. This information is given as an URL. For REAP to be able to enforce rights policies in the retrieval process, the given URL must be formed as an http get request to the REAP software giving the resource to be retrieved as a parameter to the request. This means that the URL for the resource must be encoded in the metadata presented to the search engine of ADEPT. In REAP the identifier for a material is given by a combination of a collection id identifying the collection the resource is located in and an item id identifying the material within the given collection. In practice this information could have been given as a DOI, PURL or other persistent identifier as long as the resolved URL is of the form given below:

http://fenris.idi.ntnu.no:8080/REAP/get_Document?CollectionID=4?ItemId=2

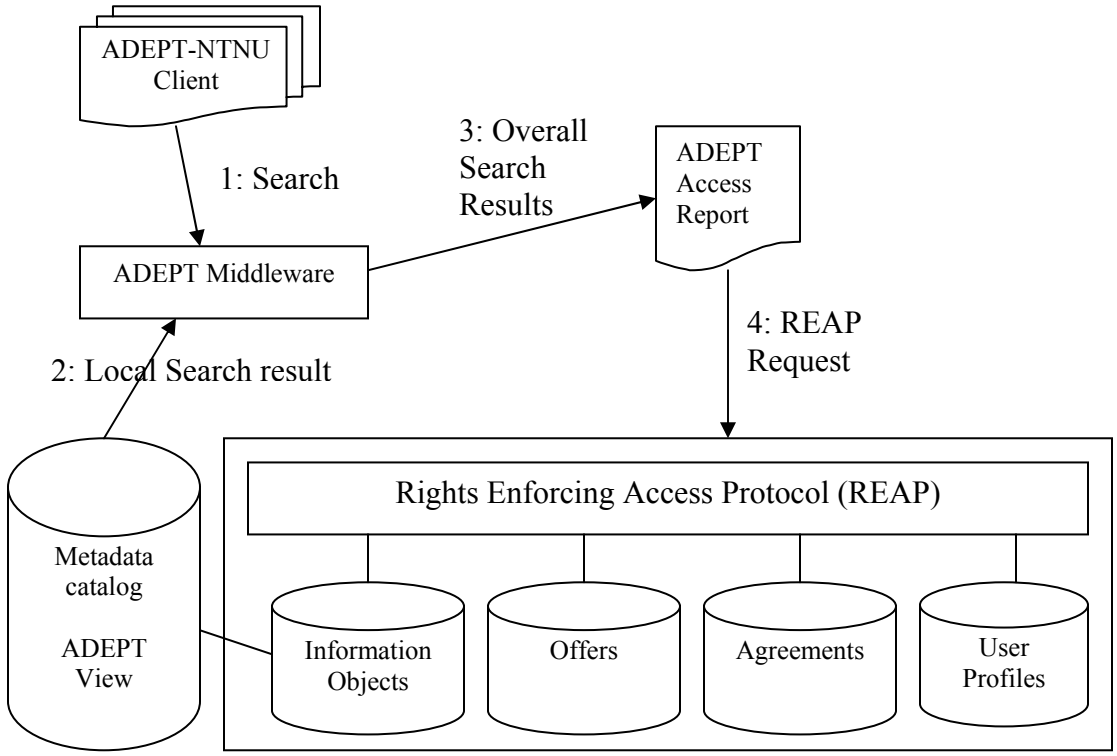

Figure 2: *REAP in an ADEPT context*

Although REAP is designed to be used with ADEPT, the intention is that the system is to be as autonomous as possible. This means that the system can be used with any system as long as the identifier for a material points to the REAP system giving the local identifier for the material as parameters to the http request. Also, REAP is not part of the ADEPT system itself, and collections under ADEPT do not have to implement or use REAP or any other rights enforcing software.

4. The REAP software

The REAP software is realized as a set of java servlets running on a Jakarta Tomcat Servlet Engine and uses an Apache Xindice native database to store rights description and other information.

When a user request access to information protected by REAP for the first time, an agreement between REAP and the user is set up regulating how the user can use the given resource. This agreement is based on the initial rights offer given for the material by the owner of the material, the information known about the user (recorded during registration), the requirements the user is willing to fulfill and the selection of rights from the initial offer that the user is interested in obtaining. Both offer and agreement is based on an application specific Rights Expressing Language described later.

First REAP checks if the given URL is well formed, that is if it contains a collection id and an item id. Then the system checks if the user is logged on. The system uses session variables, so being logged in means that there is a session registered on the server for the IP and process number of the browser/machine the user is using. If the user is not logged in the user is presented with a login page with a link to a registration page. Once the user is registered / logged on to the system checks whether the resource identified by the collection id / item id exists. If the user already has an agreement for accessing rights to the material the user can start executing rights over the material. If not, the system will interact with the user to set up such an agreement by letting the user select one or more rights from the initial rights offer for the material that he/she wants access to. Once the agreement is created, the user can start exercising the rights transferred in the agreement.

There is no rendering application created for REAP. The result of requesting access to execute rights on material as specified in the user's agreement for the material is a ticket granting or denying access to exercise the requested right. The system will keep track of the rights the user accesses and make sure he/she does not overstep the bounds of the agreement. The execution paths involved in getting access to material is given in **figure 3** below.

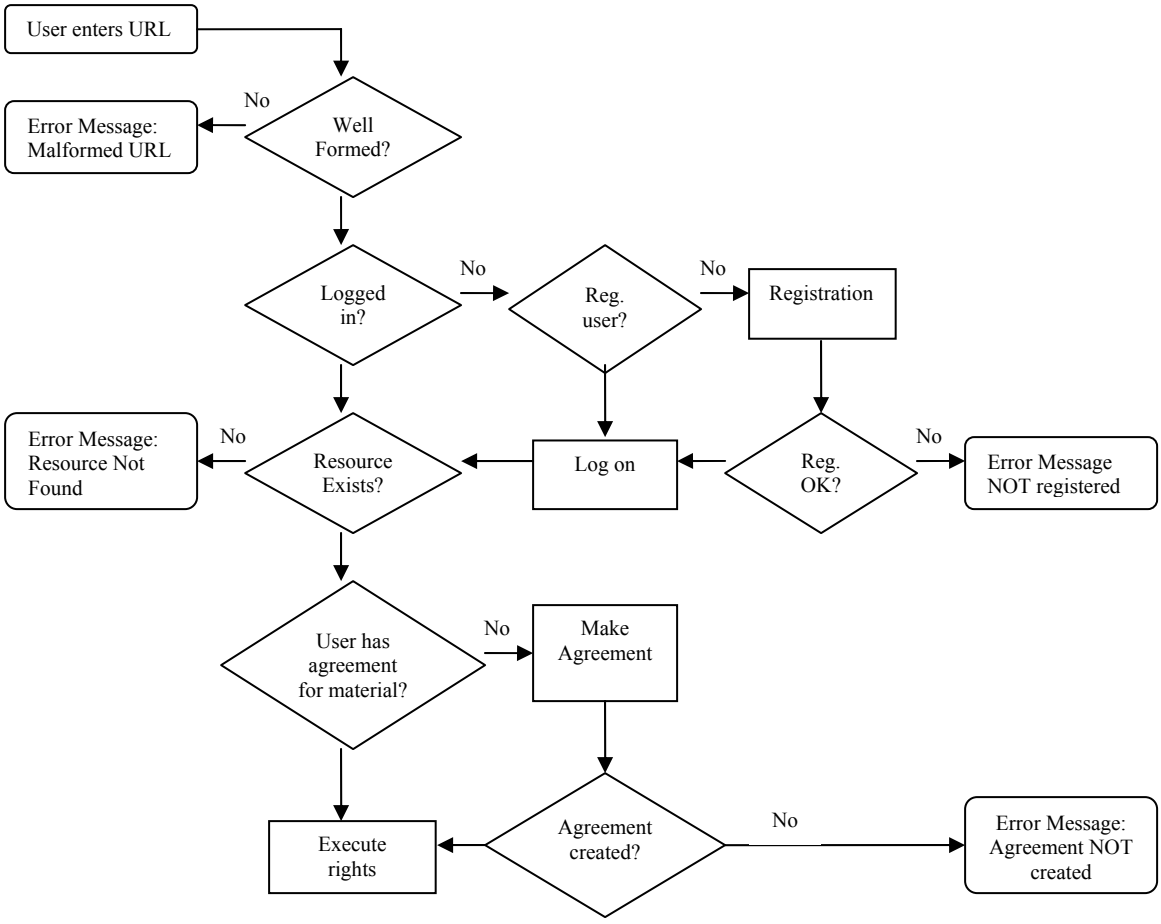

Figure 3: *Execution Paths in REAP*

5. The REAP Rights Expression Language

The rights expression language created for REAP is a subset of the ODRL rights expression language created during development of the prototype implementation with a few local adjustments and changes. Since the Rights Expression Language created for REAP is defined as a sublanguage to ODRL, the best way of describing it is to describe which parts of ODRL is not part of the language and which local additions have been made. This section presents a short introduction to the most important cut-aways and modifications made to ODRL to create the REAP rights language. Readers are assumed to be familiar with the ODRL rights language.

The advantage of using ODRL (or some other Rights Expression Language) without modifications would be that everyone familiar with the language (including scripts or programs translating from logical choices made by an author to a formal rights description) could make their own rights descriptions that REAP could understand and act upon. This would facilitate trading of rights over material over the entire life cycle of a material from creation to use and reuse, possibly including redistribution. However, the REAP software would then be required to take the entire ODRL language as input, leading to a situation where the software either had to be able to interpret all rights expressions in ODRL and act upon them, or to ignore parts of rights expressions it could not interpret correctly, denying users access to rights for which the software could not verify all requirements, constraints and conditions. Neither creating software that could understand all aspects of ODRL nor creating software that could detect expressions it could not verify seemed feasible given the limited time available for the development of REAP.

First of all, since the language is only intended to be used for describing which rights to be given to direct users of the library, the REAP rights language can only describe usage rights (display, print, execute and play). REAP is assumed to have been given the rights to the material needed for licensing these usage rights to the users of the library.

Next, the language does not have a security model. This means that offers and agreements can not be digitally signed and material and rights descriptions cannot be encrypted. Also, the language does not support the condition construct of the permission element that act like triggers revoking rights when certain conditions are no longer met. Neither are revoke constructs that, when entered into the system, revokes the rights previously offered and traded with. Utility constructs and functionality like Containers, Expression Linking and Inheritance are also not supported.

Assets are identified locally by a REAP identifier consisting of a collection id and an item id. This local identifier enables REAP to translate internally to a file/document to be retrieved without disclosing the location of the files to the user. (If the asset is not available in digital form, this kind of identifier is still used, although a resolution of the identifier will not result in a reference to a file. Assets can, as in ODRL, be further described by a context element. The context element contains an element called uid, which is a unique identifier for the material. It is important that this element does not point to an alternative download location for the material.

The requirements model in the REAP rights expression language is highly simplified. The only requirement that can be defined is payment, which is limited to prepay, requiring the user to pay for access to rights before being given such rights.

As in ODRL, there is a Constraint model containing among other things a count element which is used to indicate how many times a given right can be executed over the material. Under the count element contains, in REAP, the elements max and executed. The executed element has been added to the Rights Language of REAP and is not part of the ODRL. It is initially set to zero and is incremented by the REAP software each time a right is executed. Se **figure 4** below. Access to a right constrained by count is granted or denied as a result of comparing the value of max and executed. However, the information about how many times a user has executed a right over a material should have been recorded in the profile of the user, not in the agreement. Storing this information in the agreement requires the software to write to the agreement each time a user accesses the right in question and probably makes digital signing of agreements impossible.

```
…..
<permission>
        <print>
                <constraint>
                        <count>
                                <max>2</max>
                                <executed>0</executed>
                        </count>
                </constraint>
        </print>
</permission>
…..
```

Figure 4: *The executed element is used to indicate how many times a given permission has been executed.*

There are also other constructs from ODRL that are not part of the REAP rights expression language. Most of them have been excluded in order to make a language easy to understand by the REAP software. The draw back is that the REAP rights expression language is not as flexible as the ODRL language. The logic of the rights expression language of REAP is however mainly the same as that of ODRL.

6. Discussion

Being a first attempt to create an adequate DRM solution for digital libraries, there are some issues that have to be addressed in relation to REAP.

Works that are described by a rights language are often realized as a set of files. For instance, a work can often be a set of text documents, images, video and citations. HTTP transported documents resolves this by letting the rendering application (browser) issue subsequent request to the server, one for each part of the document. REAP is supporting one file pr rights description as the internal resolution from a collection id and an item id results in a path to a single file. Issuing successive requests for delivery of material to REAP in its present form would require an agreement to be set up for each part of the information object/asset.

REAP does not support renegotiation of agreements. If there is a constraint on a right so that the particular right can only be exercised 4 times, there is no functionality to renegotiate the agreement, letting the user obtain extended rights or new rights to a material.

The terms offer and agreement inherited from ODRL can be misleading. One would think that making an agreement would involve some kind of negotiation or two-way dialog between parties where the parties makes an agreement based on some middle ground. However, since the user cannot influence the initial rights offers given by authors REAP is based more on a take it or leave it concept where the only rights a user can obtain is a full or partial subset of the rights from the initial rights offer. It is a

Authors are meant to be able to use the Rights Expression Language of REAP to express rights in order to protect the material from usage in violation of copyright. However, there is nothing stopping authors from making expressions in the language more targeted at protecting the business interests of the authors or publishers than protecting copyright over the material. Copyright legislation in most countries is balancing the needs and power of authors and users of information. However, since distribution of material over the Internet is in its nature transnational and copyright is defined nationally and varies, Rights Languages have to be flexible enough to express copyright independently of any nation's legislation. The actual expressions made in a language should be in according to a certain copyright legislation, but the expressional power of the language itself should not be limited by single nation's legislation.

REAP is a prototype of a DRM system. It is an attempt at creating a rights management system and can be seen as a first step to create more adequate solutions. The development and testing of REAP has provided

valuable insight into the problems and opportunities inherent in DRM systems. The prototype is currently not under further development.

7. Acknowledgement

I wish to thank my supervisor Prof. Ingeborg Sølvberg for invaluable help during the work with REAP and my master thesis in general and for valuable comments on this paper.

8. References

[Alexandria]
Alexandria Digital Library Project
http://www.alexandria.ucsb.edu/

[DRMWatch]
DRMWatch
GiantSteps Media Technology Strategies
http://www.giantstepsmts.com/drmwatch.htm

[Erickson 2001]
Information objects and Rights Management
A Mediation-Based Approach to DRM Interoperability
John S. Erickson, Hewlett-Packard Laboratories
http://www.dlib.org/dlib/april01/erickson/04erickson.html

[Ianella2001]
Digital Rights Management (DRM) architechtures
Renato Ianella
Chief Scientist, IPR System
http://www.dlib.org/dlib/june01/ianella/06iannella.html

[ODRL]
Open Digital Rights Language Specification version 1.1
http://www.odrl.net/1.1/ODRL-11.pdf

[PayetteLagoze2000]
Policy-Carrying, Policy-Enforcing Digital Objects
Sandra Payette and Carl Lagoze
Research and advanced Technology for Digital Libraries
4th European Conference on ECDL
Lisbon, Portugal Sept 2000, Proceedings

[RosenBlatt, Trippe, Mooney 2002]
Digital Rights Management Business and Technology
Bill Rosenblatt, Bill Trippe, and Stephen Mooney
M & T Books 2002
ISBN 0-7645-4889-1

[Stefic1997]
Letting Loose the Light: Igniting Commerce in Electronic Publication
In: Internet Dreams: Archetypes, Myths and Metaphors
Authors Mark J Stefik, Vinton g.Cerf
ISBN: 0262692023 (may 9. 1997)

[Vestavik2002]

REAP Et system for rettighetsstyring i digitale bibliotek.
Øyvind Vestavik, 2002
Available in Norwegian from
http://www.idi.ntnu.no/~oyvindve/MasterThesis.pdf