University of Amsterdam

System and Network Engineering

Identifying and retrieving digital objects: A Study of the Handle System

by

Taarik Hassanmahomed BSc

Coordinator: Dr. Ir. C. de Laat and Dr. Karst Koymans, University of Amsterdam

Supervisor: Dr. Paola Grosso, University of Amsterdam

Research Project 2

Academic year 2009-2010

July 26 final 1.3

Abstract

This report covers the research on the Handle System and its applicability in AMPAS/CineGrid.org. The CineGrid.org community stores and exchanges a lot of digital content with it members. One of the partners is AMPAS - the Academy of Motion Picture Arts and Sciences. Currently, AMPAS is handling their metadata explosion in an inefficient way and is looking for alternatives, like the Handle System. The Handle System is a general purpose distributed information system to identify and retrieve digital objects that work with metadata. The objective of this document is to cover general addressing and metadata issues concerning digital content, analyse some of the Handle System features with respect to storage and retrieval of digital content and compare this to some other systems. In this report I also present the results of a prototype implementation of the Handle System and its applicability for the CineGrid Web portal in Amsterdam. I conclude with a series of recommendation to AMPAS and CineGrid.

Contents

1 Introduction

1.1 Research question

More and more content (music, movies, books, articles, etc.) is becoming digitised and need to be accessed by various users. Some content is used for public use and other content is restricted to companies, governments or other users with privileged access.

Millions of users watch, share and comment on online video every day. Companies, like Apple, are making use of this trend by producing popular products, like the iPhone and iPad, which can access these contents in a mobile way and are pleasing the communities need for more ways for access to them.

During this project I will focus on the following issues:

- How are object identifiers being used to store find, retrieve and preserve digital content and handle the ever growing supply of content?
  - How is metadata being used to support this process and what are the important requirements?

More specifically I will try to address the following question:

- How does the Handle System compare here and can it be used to make storage, search, retrieval and preservation of digital content more efficient and reliable within AMPAS/CineGrid in particular?

To access and preserve digital content communities, like CineGrid.org [1] and AMPAS [2]- the Academy of Motion Picture Arts and Sciences -, a partner in the CineGrid.org community organisation, are looking into having their digital archives managed in a more efficient and reliable way.

This is because AMPAS is currently working with models that are beginning to have trouble handling the data explosion within the movie scene. AMPAS is challenged with archiving over 500 mainstream movies a year, with each having millions of objects, because audio, special effects and other complex metadata, need to be archived on a per frame basis.

A possible candidate for handling this is the Handle System [3][4][5]. The Handle System is a general purpose distributed information system to identify and retrieve digital objects. It was developed by Bob Khan [6], one of the founding fathers of the Internet, with support from Defense Advanced Research Projects Agency (DARPA) and the Corporation for National Research Initiatives (CNRI), which continues to develop and manage it.

One of the key characteristics of the Handle System is its persistent identifiers and metadata storage, which are location and owner independent[13]. This characteristic is also the main feature of the Digital Object Identifier (DOI) System, which has already over 40 million registered handles in the Handle System and provides "DOI" names to entities for identifying content objects in the digital environment for use on digital networks. [8][9][10] [11]

1.2 CineGrid.org current system

The CineGrid.org community is composed of members from all over the globe. The members actively produce and store digital content, which is made available to other members of the community through lightpaths. Lightpaths are direct optical data transport connection over fiber with high bandwidth. A picture of the oceanic and continental lines of which, lightpaths can make use of to collaborate with its members, can be seen in figure 1

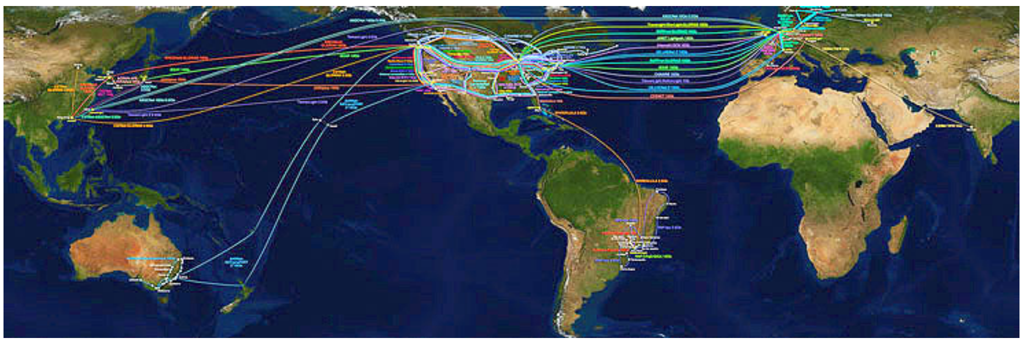


Figure 1: transatlantic lines

Like mentioned in subsection 1.1 AMPAS and the CineGrid community are currently looking into a new way to manage their ever growing amount of digital content; e.g. popular digital content at CineGrid includes the 4k movies, these are 4 times sharper than HD movies. Next to storage of the digital content itselves, there is a need to store all the metadata, which comes with the content. Currently CineGrid.org and AMPAS are working with methods, which still do the job, only in an inefficient and not in a standardised way.

For instance at the CineGrid.org community storage of digital content is organised in a direct way with the use of simple names and simple directories for each file or group of files and with no extra information (metadata) on particular servers. To keep this system available and consistent the software component IRODS [36] is used to replicate the important parts of the system to at least two other servers.

Lookup of digital content is possible through a name or description, which needs to be send via email to an authorised person with access to a local spreadsheet with some information about all the content. This person will redirect you to the site that houses the content you are looking for. After contacting the particular site you can get an SSH account, which gives remote access to the directory with at least the file you are looking for. Because of the simple naming structure, other files will seem to have a name similar to the one you need. This means you need to manually check the content of a group of files to get to the file you need.

Naming has, therefore, been a discussion point within the community. There is a need for a unique identifier, but one that is also descriptive in order to make it easier to recognise a file. Of course this can give issues when using different file systems, the number of characters, languages, punctuations, etc.

Continuing with the naming issue we see that AMPAS is currently working with "smart names" where they put metadata information onto the filename. The problem here is that they have no standard way of doing this within AMPAS and, therefore, these names only work locally.

Currently CineGrid.org is looking into alternatives for the current system, one of the alternatives is Collective Access, a system primarily used for storage of art objects together with metadata. Use of this within CineGrid.org and AMPAS is still in the development stage and official use of it has already started.

CineGrid Amsterdam is currently looking into something more universal, which does not rely on software packages. What they use is a portal that operates with locally configured metadata to initiate a work-flow system, that uses RDF files to find out where what is and how to access it with in network paths, but this is part of research in process. A picture of the architecture used by CineGrid Amsterdam can be seen in figure 2, where the red lines illustrate the extracting of the metadata from the storage for display onto the portal.

Both AMPAS and CineGrid.org are still looking into other alternatives, the Handle System could, therefore, also be a possible candidate, if it can address most of the issues.

1.3 Scope and Methodology

For this research I will look into the following issues:

- addressing and naming digital content and efficient use of metadata[14]. (Section 2 and 3)
- available systems and their features, use of metadata, costs and limitations. (Section 5)

Concretely I will:

- Compare the Handle System to some of these systems. (Section 5.4)
- Setup of an implementation scenario for the CineGrid/AMPAS usecase. (Section 6)

The outcome of my research will be a recommendations list to CineGrid/AMPAS regarding the applicability of the Handle System for addressing and naming content. (Section 7)

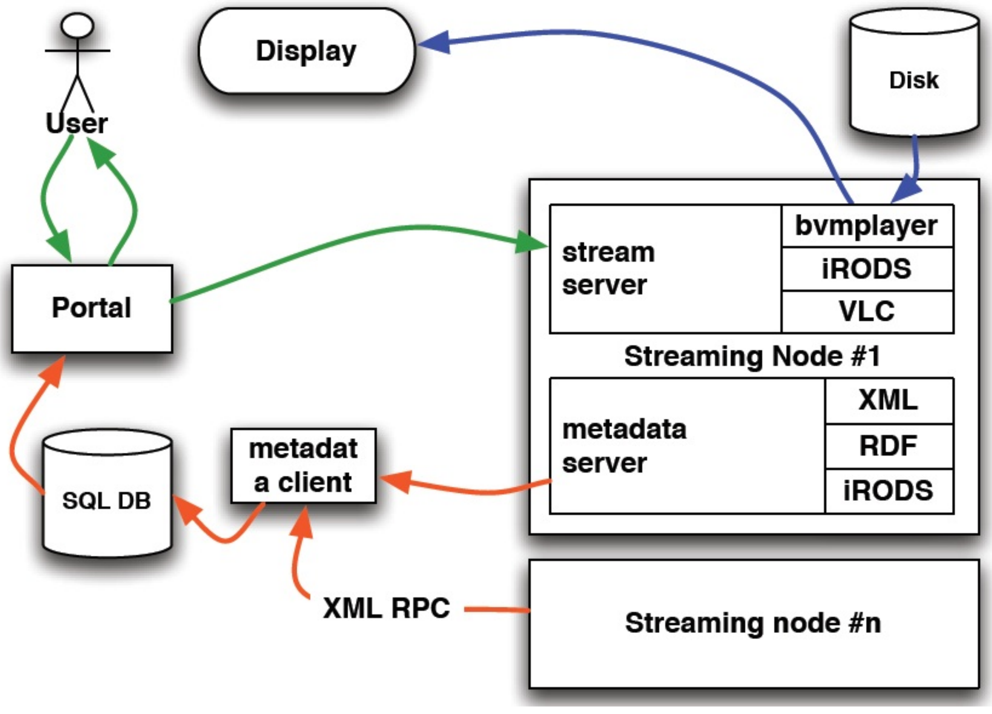
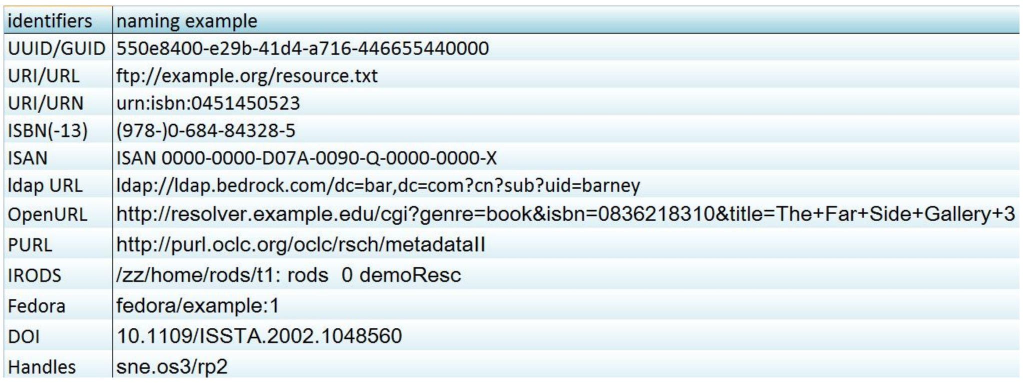


Figure 2: Portal CineGrid Amsterdam Architecture

2 Addressing and naming digital content

The number of people that are getting connected and are producing and storing digital content is growing rapidly, to make this content readily retrievable it needs a label (name) and a way to access it (address). When looking into the addressing and naming digital content I see that there is no single way of identifying digital content. Furthermore people from various communities, including political, philosophical and scientific, are still discussing on what conventions to use for naming. Examples of common naming conventions are Uniform Resource Identifier (URI)[38] (which includes Uniform Resource Locator (URL) and Uniform Resource Name (URN)), International Standard Book Number (ISBN) [33], Digital Object Identifier (DOI)[8], Universal Unique Identifier (UUID) which includes Global Unique Identifier (GUID)[15], etc. and, of course, locally used conventions from within the different communities. In figure 3 you can see examples of some naming conventions, some of which I will look into more detail in section 5.



identifiers	naming example
UUID/GUID	550e8400-e29b-41d4-a716-446655440000
URI/URL	ftp://example.org/resource.txt
URI/URN	urn:isbn:0451450523
ISBN(-13)	(978-)0-684-84328-5
ISAN	ISAN 0000-0000-D07A-0090-Q-0000-0000-X
ldap URL	ldap://ldap.bedrock.com/dc=bar,dc=com?cn?sub?uid=barney
OpenURL	http://resolver.example.edu/cgi?genre=book&isbn=0836218310&title=The+Far+Side+Gallery+3
PURL	http://purl.oclc.org/oclc/rsch/metadataII
IRODS	/zz/home/rods/t1: rods 0 demoResc
Fedora	fedora/example:1
DOI	10.1109/ISSTA.2002.1048560
Handles	sne.os3/rp2

Figure 3: Examples of some naming conventions

All these have their own restrictions to provide some or all of the following benefits of proper naming:

- Understanding the context of an object by giving additional information (i.e., metadata) with the identifier.
- Interoperability
  - between users in a development environment
  - between namespaces
- To ease automation when making use of tools for searching.
- To easily recognise an object and avoid possible ambiguity.
- To make it look more professional (not allowing long names, funny names, or common shortcuts).

What I can do is look into some of the general issues that concern naming and addressing of digital content namely:

- Namespaces.
- Identifier Encoding.
- Location dependency.
- Scalability.
- Preservation.
- Human-friendly identifiers.

2.1 Namespaces

To efficiently work with large number of digital objects in a distributed system we need identifiers that can find and identify digital objects over the full life-cycle of the resource. It is, therefore, important to look into the issues of:

- Uniqueness and Persistence

Uniqueness

Any name defined within a system needs to be resolved unambiguously to its corresponding object especially when a local object needs to be moved to other systems.

Persistence

When an object needs to move over time we should always be able to identify the same resource, even when that resource ceases to exist or is no longer accessible.

When looking at persistence more specifically we can look at the context of the object as described by Moore et al. (2000) [16]

- Digital object representation, which is about all the attributes that describe or are connected to the object.
- Data collection representation, a subset of the previous to organise different digital objects.
- Presentation representation, the multiple structured views through which collection interactions are defined.

Besides focusing on the technology issue, persistence is much easier to maintain with long-term governance and policy support at institution, sector and global levels [17].

2.2 Identifier Encoding

Encoding decides how we will experience the namespace. Next are a couple of encoding approaches that have been commonly used in naming [18]:

The approach of limited to character:

The URL specification, which limits the namespace to printable ASCII characters. This is used a lot because of its descriptive nature and a usually maintainable unique encoding in the global scope. An extreme case is the digital identification, like GUID [15], used for instance for MAC-addresses to identify network interfaces. This ensures uniqueness regardless of the character set, but this could eventually require a corresponding descriptive name system to describe the digital identification and looses it advantages a little. The classic example is that of IP addresses, where Jon Postel invented DNS to give a descriptive name to the IP addresses.

The Use of Encoding Tags:

This is the approach used by MIME[22], LDAP[23] and many other application level protocols. It generally attaches a tag (an ASCII string) before or after the original context. The tag uniquely identifies the context. There are over one hundred encoding schemes defined by the ISO standards and the many equivalent names from various encodings make it unnecessarily complex. To make it easier for applications to recognise this you need a default encoding.

The Unicode Based Encoding:

A single character set encoding for all characters of all the native languages used around the world. This has the problem of interoperability and compatibility on certain systems and platforms. A variation of Unicode encoding, UTF-8 [24] provides a better practical solution because no overlapping of encoding will occur among different native alphabets, hence ensuring global uniqueness. Because of the lack of software that supports UTF-8 and the existence of various native computing environments, it is still not practical to force every name to be entered in UTF-8 encoding.

2.3 Location dependency

Location dependency is closely related to the persistence discussed previously. Whenever we want to find a certain object we must know its location. There are several common ways possible [25]:

Simple solutions:

When networks are small, retrieving objects could be as easy as broadcasting you request over the network, when networks grow broadcasting becomes inefficient, because it can use of too much bandwidth and even interrupt other requests. This problem can be resolved with multicast addresses, which would then become location services.

Home-based approaches:

Another possibility that works for mobile object is the home-based approach where a home agent keeps track of the current location of the object. This works for mobile object, but may increase communication latency, if the object and the home agent are apart. Caching is a commonly used solution to help in this situation.

Distributed Hash Tables (structured P2P):

A third possibility is the use of a distributed hash table, where objects are unequally assigned a random key from a large identifier space. This identifies the object based on some distance metric, that can reduce the search of an object to the $O \log n$.

Hierarchical location service:

The fourth possibility is dividing a network into a collection of domains. The lowest domain is called a leaf domain, which is usually at the local area network level. Each domain D has a higher level directory node that keeps track of the objects in the domain. This leads into a tree of directory nodes with a top-level domain called the root node, which knows about all objects. Lookup happens by forwarding the request up the tree until a record is found directing the request back down a different path through the tree.

2.4 Scalability

As access to digital content relies on its underlying system, scalability of those systems is an important issue when the amount of digital content grows beyond the efficiency of a single centralised system. Scalability of a system can be measured with respect to at least three things [25]:

- Its size, meaning easily adding more digital content to the system.
- Geographically scalable, where users can easily access digital content from far away.
- Administratively scalable, digital content can still be easily managed over these multiple systems.

Techniques for scaling can be seen as one of the following three[25]:

Hiding communication latencies:

By either moving communicative processes from the server to the client (e.g. initial checks of webforms) or other forms of asynchronous communication.

Distribution:

This involves splitting a component into smaller parts and spreading those parts across the systems. Possibilities are either vertically where you go deeper in the tree of hierarchical connections between servers or even horizontally where you get islands of servers working together, like in a peer-to-peer system.

Replication:

You practically have a copy nearby and can choose to balance the load between components to increase availability. A special form of replication is caching, which result in making a more local copy of an object or reference to speed up performance. One problem that comes with replication is keeping data consistent, especially with large amount of servers. It will take a while before changes have been propagated to every server

Of course scaling in size comes with more cost for hardware and software. A system need to be designed to minimise the need for expensive resources, although the cost for use of software might be brought down by use of open source, but license fees for services and support that come with the software might not be as "free". Other costs are related to implementation and management effort.

Once digital content is stored a domain administrator providing access to the content becomes the "owner" of all the digital content and the domain name associated. What is needed for digital content is a general purpose naming system that allows remote management and enables administrators to create and manage a smaller group of content in a distributed and efficient way.

2.5 Preservation

> *"Digital preservation refers to the series of managed activities necessary to ensure continued access to and preservation of digital materials"* [26].

There are a variety reasons why preservation of digital content is needed[27].

Bit-rot, this is the lowest level where preservation is needed. A book can be stored for a hundred years and still contain readable content, in contrary digital content usually won't survive a decade without any form of "decay". This is because digital content is more fragile when it comes to physical damage to storage media/data carriers that are needed to keep the data useful. Solutions for preventing this decay are:

- Refreshing the storage media, examples are CD/DVD disks, which have shorter lifespan when stored improperly.
- Replicating the data over multiple storage solutions helps by surviving in numbers, although this comes with certain extra costs with respect to storage and managing of more copies.

The next reason is the problem with hardware/software **obsolescence**. Nowadays less and less people are using floppy disks because of the more efficient alternatives, like the USB sticks. The digital content might have survived on the floppy disk, but with no floppy disk reader available the content can be regarded as lost. Solutions for coping with this obsolescence are:

- Refreshing or Migration of the data to the new system environment, still the new environment would not be compatible, because of differences in formatting. This would then also need some conversion, which, although does the job, does not guarantee that all functions will supported in the new environment.
- An alternative to conversion is to emulate the functionality of the system that was used to run with the data. This,of course, requires the description of the architecture to be perfect and complete.

A third main reason is the **loss of meaning** of the digital object, because of lost specifications, or because the people with knowledge about the data are no longer around. Solutions to the loss of meaning are:

- Storage of additional descriptive data about the data itself in the form of metadata and maybe together with relational schemas, like RDF (Resource Description Framework), to further describe to relations between the data.

A fourth reason is **preserving provenance** and source of data so we can trust an objects authenticity. Solutions to loss of authenticity are:

- proper recording of versioning history and maybe cryptographic solutions (encrypting, signing, hashing).

2.6 Human-friendly identifier

To get more information into an identifier one could use either a simple identifier or a compound identifier [28].

A compound identifier is an identifier that not only carries a reference uniquely identifying an object, but also some explicit meaning or interpretable information about the object for example, information about intellectual property rights or other metadata.

A simple identifier could be used to link to a table of look up data or a registry of metadata.

Both simple and compound identifiers could achieve the same end of being useful in any environment, but with differing consequences: a compound identifier removes the need for central registries of metadata, but makes the identifier much more lengthy and complex to use.

3 Use of metadata

"An element of metadata describes an information resource, or helps provide access to an information resource. A collection of such metadata elements may describe one or many information resources" [29].

From the previous citation we can conclude that the use of metadata is an important part in understanding the semantics of digital content. The power of metadata lies in choosing the right set of element.

ISO [31] for instance provides many different standards for different purposes or types of information resources, they are referred to as metadata schemes. But there are literally hundreds of metadata schemas to choose from and the number is growing rapidly, as different communities seek to meet the specific needs of their members. The resource community, therefore, may define some logical grouping of the elements or leave it to the encoding scheme. For example, Dublin Core may be used as basis to which extensions may be added. Some of the interesting standard schemas are [32]:

Dublin Core focuses on Networked resources and is an interoperable online metadata standard.

VRA Core focuses on Arts and provides a categorical organization for the description of works of visual culture as well as the images that document them.

ONIX focuses on the Book industry and is an international standard for representing and communicating book industry product information in electronic form.

IEEE LOM focuses on Education and specifies the syntax and semantics of Learning Object Metadata.

indecs focuses on Intellectual property and is the Indecs Content Model - Interoperability of Data in E-Commerce Systems addresses the need to put different creation identifiers and metadata into a framework to support the management of intellectual property rights.

PBCore focuses on Media and is a Metadata & Cataloging Resource for Public Broadcasters & Associated Communities

MPEG-7 focuses on Multimedia and is The Multimedia Content Description Interface MPEG-7 is an ISO/IEC standard and specifies a set of descriptors to describe various types of multimedia information and is developed by the Moving Picture Experts Group.

These schemas are usually applied together with one or more descriptive schemes to help with the semantics of the metadata schema. Important schemes include [35]:

- HTML (Hyper-Text Markup Language)
- SGML (Standard Generalised Markup Language)
- XML (eXtensible Markup Language)
- RDF (Resource Description Framework)
- MARC (MAchine Readable Cataloging)
- METS (Metadata Encoding and Transmission Standard)
- MIME (Multipurpose Internet Mail Extensions)

Metadata itself can be categorised in the following ways[30]:

General vs. specialistic

Dublin Core has only fifteen core element that relate to most content and is regarded generic, while on the other hand IEEE LOM is classed as specialist metadata since it is designed for a specific community.

Minimalistic vs. rich

Minimalistic schemas offer less flexibility for specifying specific element information and tend to describe objects in isolation either with very cursory or no relationship data. General metadata is often minimalistic in nature with specialist metadata schemas being richer in data collected.

A rich metadata schema looks through the eyes of a specific community. This is usually to a fine level of granularity. One of the earliest schemas to do this was AACR (Anglo-American Cataloguing Rules), which since the 1960s has been encoded using MARC (MAchine Readable Cataloging). AACR2R (second edition revised) is the bibliographic standard used by libraries to describe what is in their collections.

Rich schemas can also be used at a minimalistic level. A few elements of a schema could be used to describe resources with much detail. For example, the VET metadata application profile (Vetadata) recommends just five elements to help with basic search and lets organisations share and exchange resource information.

Hierarchical vs. linear

Hierarchical schemas identify and display relationships between elements. An example of a hierarchical schema is IEEE LOM.

A linear schema is characterised by the absence of element relationships. Each element is unique and defines a specific data element. Dublin Core is an example of a linear schema.

Machine generated vs. human authored

Metadata can be setup in structured or unstructured ways. When structured it becomes easier for machines to extract this information by using knowledge about the used schema or by using other algorithms.

Applications use complex algorithms to increase the accuracy of the machine generated metadata. Google is an example of a system that creates and uses machine generated metadata.

Structured vs. unstructured

When certain rules are followed simple or complex data structures can be setup.

Embedded vs. detached

HTML can be used to record metadata known as embedded metadata as well as the instructions for rendering information on a web page. The two most common tags used for embedded metadata are DESCRIPTION and KEYWORDS.

Detached metadata can be either stored in file container or package called a record away from the object with a link to the data itself or set of records in a database. Another possibility is that the metadata is collected in a repository with other metadata.

4 Handle System Overview

In the early 1990s Bob Khan [6] was working on the Digital Object Architecture [37] as part of the "Computer Science Technical Reports" (CSTR) project for the Corporation for National Research Initiatives (CNRI). Around that time Bob Khan also discussed some issues from this architecture with other researchers, in relation to digital collections and linking electronic libraries. Some of the issues discussed were the role of semantics in identifiers and various intellectual property issues.

From the architecture Bob Khan developed one key component, a *general purpose resolution system* with support from Defense Advanced Research Projects Agency (DARPA) at the Corporation for National Research Initiatives (CNRI), which continues to develop and manage it.

> *"It is probably best to view the Handle System as a name-attribute binding service with a specific protocol for securely creating, updating, maintaining and accessing a distributed database."*[3]

The Handle System can be applied for different situations:

- Meta-data services for digital publications.
- Identity management services for virtual identities.
- Any other applications that require resolution and/or administration of globally unique identifiers.

There are over 1,000 handle services running today, located in 63 countries, on 6 continents; more than 750 of them are at universities and libraries. Handle services are being run by user federations, national libraries, national laboratories, universities, computing centres, national and local government agencies, contractors, corporations and research groups. The number of prefixes, which allow users to assign handles, is growing and exceeds 212,000. There are four top-level Global Handle Registry (GHS) servers in the United States that receive (on average) 68 million resolution requests per month.[40]

There is also a special GHS in China, this one is a collaboration between CNRI and the China Internet Network Information Center (CNNIC), the state network information centre of China. CNNIC is China's domain name registry, which is operator and administrator of the ".CN" country code top level domain (ccTLD) and Chinese Domain Name (CDN) system. A Handle-DNS integration system has been developed to integrate Handle with DNS through the .cn domain. This service co-exists with the existing DNS operation and does not force the DNS client to change. [10]

The Handle System is characterised by the following features[3][4]:

- Globally unique handles through the use of a prefix and suffix.

- Handle name persistence by removing the dependency between the entity and their names.
- Registration of multiple instances for availability of resources.
- Registration of multiple attributes for persistence of resources.
- Extensible namespace for integration of any local namespace in to the global handle namespace by assigning a prefix.
- International support by basing handle encoding on Unicode 3.0, but mandating UTF-8.
- Distributed service model allowing replication of servers and distribution of handles to a cluster of individual servers.
- Secured name service through client and server authentication and service authorization with a secret key (phrase) or public key (default is 1024 bit DSA).
- Per handle distributed administration service, allowing secure management by one or more administrator at any network location.

The main parts of the system are its Local Handle Service (LHS), its Global Handle Service (GHS) and its namespaces. The Global Handle **namespace** is the superset of many Local Handle namespaces. The naming of a Handle is hierarchically divided in to the GHS part (prefix) and the LHS part (suffix), both the prefix and suffix need to be unique in their space.

- The prefix is the naming authority and can be subdivided with a . (dot) i.e. parent.child, the prefix names can contain any UTF-8 character except for the . (dot) and / (slash).
- The prefix and suffix are divided by a / (slash).
- The suffix is part of the local namespace of a naming authority and can be any Unicode 2.0 character.

The names are case sensitive unless configured otherwise. To get a naming authority (prefix) registered on the GH server you must agree with the policies of the Handle System and pay a yearly maintenance fee per prefix.

Global Handle Service is the Global Handle Registry used to dispatch resolution request for Local Handel Services and manage any handle namespace by maintaining naming authority (prefix) handles at the 0.NA/prefix handle on the GHS server. The idea is to maintain service information describing "home" service (sites and interfaces), which are the servers that are allowed to be LHS for the registered prefix. The information also describes how the servers are distributed and replicated.

Local Handle Services is hosted by organization with administrative responsibility for handles under certain naming authorities. The LHS may consist of one or more replicated services SITES, which may consist one or more handle servers, which have the same set of handles distributed among each other through a hashing mechanism. This is distributed service model is also applied to the GHS. The LHS is where the handles are located and stored and, therefore, rely on the existence of the LHS.

An example of a handle is:

CineGrid.AMPAS/NICEMOVIE4k

When creating a handle each handle may have a set of attributes or instances (metadata) registered to it, which are build out of:

- a uniquely 32-bit index number for an attribute or instance, giving almost 4,3 billion possible entries for registering attributes and or instances within a handle.
- Type for identification and syntax of an attribute or instance.
- Data for semantics of the attribute or instance.
- Permissions (read/write/execute by public/admin) per attribute or instance.
- Time to Live (TTL) for caching purposes.
- Date to indicate creation or modification.
- and possibly some references to other handles.

But a handle must always have a handle name for identification and an administrator set as the HS_ADMIN type, which is part of the distributed administration service and allows secure name service. The HS_ADMIN tells the handle where to look for the authorization information of the administrator of the handle. Usually this is the global 0.NA/prefix HS_PUBKEY value at index 300 on the global server. This can also be a HS_VLIST type at a different index (usually 200), which contains a list of the administrators and their index:handle HS_PUBKEY locations.

The HS_VLIST can contain any index:handle value, but it is usually used to group administrators.

The Handle System has next to HS_ADMIN, HS_PUBKEY/HS_SECKEY, HS_VLIST the following other types registered with the GHS:

- HS_SITE
- HS_ALIAS
- `HS_NA_DELEGATE`

- HS_SERV
- HS_PRIMARY

If you want to add or delete servers or SITES, you do this by changing the number of servers in the HS_SITE type itself, which is one of the entries of the 0.NA/prefix and is part of the distributed service model. You can have multiple servers be part of the same HS_SITE value if you want to distribute your handles to servers. When you create multiple index entries of the type HS_SITE, this means you want to replicate your servers.

Yet another type is used when you want to redirect a handle to another, you can then use the HS_ALIAS to indicate the handle to redirect to.

`HS_NA_DELEGATE` is used to delegate certain child naming authorities (sub-domain of the prefix) to the LHS described by the `HS_NA_DELEGATE` values.

HS_SERV is used to share service information among multiple naming authorities, it allows changes to service configuration to be made in one place rather than on every naming authority handle involved.

HS_PRIMARY to indicates the preferred primary server to managed the handle when the HS_SITE contains multiple primaries.

Other components of the Handle System are the HS Caching service to ease the load on the GHS and the HS Proxy server (hdl.handle.net), which uses DNS to resolve handles.

5 Other Available Systems

The Handle System is composed of different part, its part naming and resolving system and part archiving system. The following sections give an overview of other available systems.

5.1 Naming Systems

This section gives an overview of the following naming/resolving systems:

- URI [38].
- DNS [41][42].
- Directory Services [19][20][21][23].
- ISBN [33] [34].
- ISAN [45].
- OpenURL [48][49].
- PURL [53][54].

URIs are used on the Web as a single naming system to refer to digital content and comes in two forms. A URL is a URI that identifies the location of digital content and a URN identify an object globally unique, location-independent and persistent.

The actual syntax of URI depends on its scheme, which is for instance http, ftp, file, etc. URL uses a scheme to define how to access an object and uses a DNS name with optionally a port and local pathname to find where to access an object. URN is a URI that uses the urn scheme, URN itself is not restricted to a schema, although official URN namespaces need to be registered at IANA[39]. The resolution of a URN is usually done at the server that knows about the URN scheme.

Domain Name System(DNS) was invented by Paul Mockapetris in 1983 at the request of Jon Postel and specified in RFC 1034 and RFC 1035. It is one of the largest distributed naming services in use today. It is used on the Internet for lookup of IP addresses of hosts and mail servers. Digital objects on the WWW are usually linked to a URL, which is a location currently based on DNS. Once an object moves the URL by itself does return anything related to the object. DNS is hierarchically organised as a tree, separating subdomains from the top level domains (TLDs) down.

According to the Domain Report [43] from VeriSign (the operator of two of the Internet's thirteen root nameservers, the generic top-level domains for .com and .net) there are over 193 million registered domain names across all of the Top Level Domain Names (TLDs), the .com and

.net has a base of 99.3 million domain names and had an average daily DNS query load of 54 billion per day with peaks as high as 63.2 billion per day during first quarter 2010.

Still DNS does not scale well with large data and relies a lot on caching to avoid swamping of higher levels. And because of its security and manageability (restricted to network administrators at DNS zone level), it is not suitable for general-purpose resource naming.

An interesting remark from Bob Kahn about DNS and the Handle System,

> *"In the late 90s, Jon Postel (who oversaw the DNS until ICANN took over) and I had come to an agreement to consider using the Handle System to replace or at least augment the DNS, but he couldn't make the decision by himself . . . "* [44]

And then came ICANN and Postel's death. The existing system persisted.

Directory Services (X.500/LDAP) are naming systems that describe object with attributes. Designing an appropriate set of attributes is not trivial and its design is usually done manually. X.500 defines a hierarchical data and information model with a set of protocols to allow global name lookup and search. But has implementation and compatibility issues with other applications. LDAP is a simpler and easier version and has two different versions LDAPv2 is a local directory access protocol and LDAPv3 is distributed service protocol. LDAP servers are best known for providing attribute-based searching.

International Standard Book Number (ISBN) identifies an edition of a monographic work, an ISBN is not given to work with slight correction or variations, only if the work is changed dramatically, like in a new edition, will a new ISBN be assigned. The number is defined by the standard ISO 2108 in 1970. It consist of 10 digits 1-5 digits for the country or language, 1-7 for a publisher, 1-7 for a title and the last digit is a check digit that can be X for 10. The digits are divided in a variable way, where countries or publisher that publishes a lot will get smaller identification digits, resulting in more for the title. As of January 2007 ISO changed this to 13 digit, the extra digits are added at the front to indicate an industry, that is 978 for the 10 digit ISBN to indicate "book publishing". To get an ISBN you need to buy one from an ISBN Agent, the cost is grouped into different sizes. For example the US Agent asks for 10 ISBNs: $275.00 [47].

International Standard Audiovisual Number (ISAN) is a unique identifier for audiovisual works and related versions, it is comparable to ISBN for books and developed within an ISO (International Organization for Standardization) TC46/SC9 working group. The ISAN is a 96-bit/24-hex number comprising three segments: a root (core part), an episode or part and a version and a metadata system.

ISANs can only be issued by appointed Registration Agencies (RAs) via the ISAN System following a strict set of guidelines to ensure data accuracy and system integrity. ISAN registration is typically done digitally in a process that involves the vetting of XML data to the ISAN-IA 24/7 online central repository in Geneva (Switzerland) for consideration. After data duplication and quality checks are performed, most requests are quickly approved and registered. Once an ISAN is assigned it cannot be deleted, it can only be modified by the registrant, the registrant's designated registration agency, or the International Agency. The ISAN identification and metadata system supports over 90 different content specific tags and more than 50 worldwide ratings systems in over 30 languages. The ISAN database counts currently more than 550.000 works [46].

OpenURL is a type of Uniform Resource Locator (URL) to make it easier to find appropriate copies of an object by combining metadata with the URL. In April 2005 The OpenURL Framework for Context-Sensitive Services became an ANSI/NISO standard (Z39.88-2004). Broading the use of the original OpenURL concept created by Herbert Van de Sompel and Patrick Hochstenbach at Ghent University in Belgium as part of a research project (called SFX, "special effects") (1998-2000).

How it works is that by changing the base URL or query string you can get a different copy of the same digital content in a different location or version. For example:
`"http://resolver.example.nl/cgi?url_ver=Z10.22-2010"`
, which has a base representing the resolver at example.nl, changing this to another server, could result in the same object on that server. Changing some of the metadata provided in this example as `url_ver=Z10.22-2010` in the URL string to `url_ver=Z10.22-2009` could give a different version of the same object. At this time OpenURL resolvers are mostly private and look in local databases to find objects. There are different researches [50][51][52] on how to use OpenURL efficiently.

A **Persistent Uniform Resource Locator** (PURL) is a "persistent" Uniform Resource Locator (URL) that point to the location of a specific object. The PURL Service has been running since the beginning of January, 1996. Within three months it had serviced 178,000 resolution requests for the 5,500 PURLs in its local database for 5,000 different users. How it work is that a HTTP server domain is used to resolve PURLs to a specific object (another URL), instead of browsing directly to the object. The domain name on the HTTP server needs to be related to a PURL server e.g. `http://purl.org/net/example/MyPURL` is the link to the object. This is persistent since the PURL will not change if you want it to link to another object or if the object moves. Resolving unfortunately only works locally on the same server and not globally between purl servers. The use of PURL is free after registering on the PURL homepage [53].

5.2 Data Management Systems

An example of where the Handle System is being used as archiving software is at the Library of Congress, this subsection is an overview of other types of data management systems:

- Collective Access [55].
- IRODS [84] [85].
- OAI [86].
- Eprints [88] [87].
- Greenstone Digital Library Software [91].

Collective Access: In 2003 Whirl-i-Gig, a software development firm from New York working in the worlds of museums and related started, in partnership with a number of institutions in the United States and Europe, to work on development of general-purpose cataloging and collections management software. In 2006 Collective Access was publicly announced as an open source project. In 2009 version 0.6 of Collective Access was in use at least over 40 institutions. Version 1.0 is expected to be released in September 2010.

Collective Access is an open-source software used as alternative to proprietary software solutions for collections cataloging and publishing. It is a highly configurable cataloging tool and web-based applications for museums, archives and digital collections and allows users to manage collections, publish them to the Web and preserve these assets digitally. Its main components are Providence, the core cataloging tool and database application and Pawtucket a public-access module multi-purpose web-based GUI.

IRODS The integrated Rule-Oriented Data System is an open source system, which has been around for over 10 years and is derived from the Storage Resource Broker (SRB) of San Diego Supercomputing Center (SDSC) at the University of California, San Diego, who worked on it with General Atomics, the Data Intensive Cyber Environments Group (DICE) and National Science Foundation (NSF). SRB is a first generation Data Grid Management System (DGMS) providing a unified view based on logical naming concepts providing a physical-to-logical independence for client-level applications. IRODS is a second generation DGMS, which continues by abstracting the data management process itself called policy abstraction. iRODS builds upon the integrated envelope paradigm, but with more functionality and services and integrates the following functionalities:

- Data Transport.
- Metadata Catalog.

- Rule Engine for executing complex policies encoded as micro-services.
- Execution Engine for workflows.
- Scheduling System for queuing and execution.
- Messaging System for out-of-band communication.
- Virtualization system for logical naming paradigm.

iRODS data system components consist of the Server software, the Rule Engine running on each data server and the iRODS iCAT Metadata Catalog uses a database to track metadata describing data and everything that happens to it. iRODS is furthermore designed to be scalable, versatile, configurable, makes use of preservation environments and has a flexible architecture. This makes iRODS a good system for online content, but is regarded as a little complex at the same time. It is being used in Physical Sciences e.g. Astronomy Observatories, Persistent Archives and Digital Preservation / Humanities, Geosciences, High Performance and Grid Computing, plus by many international users and is growing. The CineGrid Amsterdam portal is also one of its users.

OAI The Open Archives Initiative (OAI) develops and promotes interoperability standards that aim to facilitate the efficient dissemination of content. Although it is not a Data Management Systems, it is used by many systems to enable metadata exchange through The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), which is a low-barrier mechanism for repository interoperability. This mechanism allows Service Providers to access metadata from Data Providers. The mechanism mandates at least the Dublin Core metadata standard.

Eprints is an open source software package for building open access repositories, which is a free, immediate, permanent online access to the full text of research articles for anyone, webwide and is developed by the University of Southampton, England. Eprints is compliant with the Open Archives Initiative Protocol for Metadata Harvesting and started around the same time as OAI in 2000. EPrints is a Web and command-line application written in Perl and XML . In 2006 it was the most widely used free open access, institutional repository software and it has since inspired the development of other software that fulfils a similar purpose, which have become more interesting than Eprints. Currently Eprints has shift its focus onto integrating with Current Research Information Systems, which manages projects, personnel and organisational details by adopting CERIF, the European standard for research management data. Some of the other open source repository software include dLibra, which is the first Polish system for building digital libraries [89], CiteSeerX a public search engine and digital library and repository for scientific and academic papers with a focus on computer and information science, which is currently very popular [90] and many more.

Greenstone Digital Library Software is an open-source suite of software for building and distributing digital library collections. Greenstone is produced by the New Zealand Digital Library Project at the University of Waikato and developed and distributed in cooperation with UNESCO and the Human Info NGO. The administration function build in it enables the items to authorize new users to build collection. It has full-text searching and can harvest metadata over OAI-PMH, supports Dublin Core, is scalable, configurable with plugins and flexible towards displaying changes to the collection. Greenstone is used by universities, libraries, repositories all over the world and including UN agencies and Humanitarian collections.

5.3 Handle System implementations

Besides the handle.net [60] software there are other implementations using the Handle System framework. Next is a description of the following few:

- DOI [10]
- D-Space [92] [94]
- Fedora [93]
- OpenHandle [66]

DOI(Digital Object Identifier) name is an identifier (not a location) for an entity on digital networks. It provides a system for persistent and actionable identification and interoperable exchange of managed information on digital networks. Unique identifiers are essential for the management of information in any digital environment. The International DOI Foundation is a non-profit organization established to develop and manage the DOI System. The DOI System evolved from a project of the Association of American Publishers in 1996, which joined forces with the International Publishers Association and the International Association of Scientific Technical and Medical Publishers to launch the Foundation in 1998. The International DOI Foundation (IDF) supports the needs of the intellectual property community in the digital environment, by the development and promotion of the DOI System as a common infrastructure for content management. The DOI System has four components:

- Numbering and naming: assigning a number or alphanumeric name to the intellectual property entity that the DOI name identifies. This is the NISO syntax, standardised as ANSI/NISO Z39.84-2000.
- Description: creating a description of the entity that has been identified with a DOI name, through the DOI Data Model, which is based on the open *indecs* initiative.

- Resolution: making the identifier "actionable" by providing information about what the DOI name should resolve to and the technology to deliver the services that this can provide to users; this uses the Handle System. The DOI System is an implementation of URI. The DOI System, has over 45 million registered handles and is the largest single contributor of the Global Handle Service.
- Policies: the rules that govern the operation of the system, in a social infrastructure. It delegates DOI registration to Registration Agencies, just like other naming organisations.

Major applications of the DOI system currently include:

- Persistent citations in scholarly materials (journal articles, books, etc.) through CrossRef, a consortium of around 3,000 publishers.
- Scientific data sets through DataCite, a consortium of leading research libraries, technical information providers and scientific data centres.
- European Union official publications through the EU publications office.

In December 2009 there were already 43 million DOI names assigned with 60 million DOI resolutions per month.

D-Space is a joint project of the MIT Libraries and HP labs. It helps create, index and retrieve various forms digital content. Interoperability between systems is built-in and it adheres to international standards for metadata format by making use of OAI-PMH. Dspace is a service model for open access and/or digital archiving for perpetual access. Dspace can make Institutional Repository and the collections are searchable and retrievable by the Web. Dspace makes use of Handle System's global resolution feature, but does not supply any handle server or prefix.

Fedora was developed jointly by Cornell University Information Science and the University of Virginia Library and inspired by the paper "A Framework for Distributed Digital Object Services" from Bob Khan, which is the bases of the Handle System. Fedora provides a general-purpose management layer for digital objects. Object management is based on content models that represent data objects (units of content) or collections of data objects. The objects contain linkages between datastreams (internally managed or external content files), metadata (inline or external), system metadata (including a persistent identifier that is unique to the repository) and behaviours that are itselves code objects that provide bindings or links to disseminators (software processes that can be used with the datastreams). Content models can be thought of as containers that give a useful shape to information poured into them; if the information fits the container, it can immediately be used in predefined ways. Features include OAI-PMH support, access data via web APIs, full text search, Web-based Administrator GUI and much more. The Fedora server architecture is based upon four main Application Programming Interfaces (APIs): manage, access, search and the Open Archival Initiative service (for metadata harvesting). The types of organizations in the Fedora Commons user community are diverse e.g. broadcasting and media, museums, publishing, etc.

OpenHandle is an open-source community project to expose Handle values in common text-based serializations to make the data stored within the records more accessible to Web applications in formats including JSON, RDF/N3 and RDF/XML. It is not really an implementation, but more a better front-end application written in Java to the Handle System. The development of the OpenHandle is at the moment low, but it is still able to extract handle values (metadata) in combination with program languages including C, Java, Python, but with others, like Java script, it is lacking behind.

5.4 Some Comparisons

In this section I will compare the Handle System with the following:

- a name service (ISAN) [45],
- a data management service (Collective Access) [81] [82] and
- a Handle System implementation (DOI) [57].

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Namespace	prefix/suffix	96-bit	printable ASCII	10.subprefix/suffix

Table 1: Namespace

Handle System: The prefix/suffix handle names are persistent and uniquely maintained by local LHS server itself. Encoding is UTF8/Unicode2.0 for international support.
ISAN: A 96-bit digit persistence and uniqueness is guaranteed by ISO Registration Authority processes and contracts with the ISAN International Agency. Encoding is restricted to binary.
Collective Access: According to the cataloging document from collectiveaccess.org [81] object identifiers are alphanumeric based on the naming conventions established by your institution, but an example from the collective access Tour [82] shows that non alphanumeric characters, like underscore, are also used. This suggest that it is not restricted to just alphanumeric and more likely printable ASCII.
DOI: Same as in the Handle System under leadership of IDF only with prefix `10`. RAs get a subprefix e.g. `10.1000` to use for assigning handles to registrants.

System aspect	Handle System	ISAN	Collective Access	DOI
Scalability	replication and distribution	secure and highly available	CPU-bound	RAs decides

Table 2: Scalability

Handle System: Replication of LHS to multiple SITES (separate groups of servers) and horizontal fragmentation of handles to multiple servers (within a SITE).
ISAN: The scalability of the central database is not publicly known, they do state that ISANs and its related descriptive information is hosted in secure and highly available data centres.
Collective Access: Next to database operations, which are often I/O bound collective access does media processing and is often CPU-bound, scalability gets improved with multiprocessor/-multicore architectures.
DOI: Same as in the Handle System, only RAs can decide to be a mirror of the IDF or not.

System aspect	Handle System	ISAN	Collective Access	DOI
Costs	annual $50 per prefix	$32 per ISAN	free to donate	RAs decides

Table 3: Costs

Handle System: The registration is $50 per request for prefixes and $50 per year per prefix, which allows unlimited handles.
ISAN: 35 CHF ($32) per ISAN.
Collective Access: Free to donate (open source).
DOI: "Registration Agencies" (RA) working with International DOI Foundation, Inc. (IDF) pay a couple of cents per DOI handle to the IDF, but charge registrants as they see fit.

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Preservation	locally with replication and metadata storage	guaranteed by ISAN and metadata storage	interoperability and metadata storage	guaranteed by IDF policy and metadata storage

Table 4: Preservation

Handle System: Handles are administrated at local level, replication and metadata storage provide preservation. The registration fee only ensures a globally unique namespace and global resolution is maintained at global level.
ISAN: Fee guarantees preservation of content through globally unique ISAN number and registered metadata.
Collective Access: The community is committed to data preservation and interoperability, providing support for Fedora, IRODS and OAI. But object and metadata are still administrated locally.
DOI: Registration fee guarantees preservation according to IDF policy and metadata storage of objects.

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Metadata	No restrictions, but also no advice	own interoperable generic model	supports those created and used by library and information professionals	*indecs* model for intellectual property.

Table 5: Use of Metadata and standards

Handle System: Not required, but also not restricting the use of storage of metadata at index entries of handles.
ISAN: Has its own generic model, with mandatory and recommended optional descriptive information, which supports over 90 different content specific tags and more than 50 worldwide ratings systems in over 30 languages.
Collective Access: Works with metadata standards and supports those created and used by library and information professionals. These are Dublin Core, PBCore, SPECTRUM, VRACore,

CDWA, CCO, DACS, DarwinCore, MARC and more.

DOI: Uses the *indecs* model, a standard for intellectual property. It is also used to group similar content.

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Community	run by experts, searching for new fields for use	large multimedia community and still growing rapidly	growing community of libraries, museums, archives and historical societies around the globe	Large customer base in the intellectual property.

Table 6: Community and growth

Handle System: Maintained by experts that have been in the business a long time, but its reputation has not grown as much compared to other systems.

ISAN: Has clients connected with airline video (WAEA), MPEG4 systems, broadcast standards such as ATSC, ARIB and DVB, content IDs, TV services, Apple iTunes, CableLabs VoD, OpenEPG, TVAnytime, Harrises D-Series software Optical media standards such as AACS (for HD DVD and Blu-ray) and Microsoft Windows Media Encoder Studio Edition. ISAN registrations have increased over 800% since this time last year largely due to the increased activity in the Blu-ray market as well as activity in various European countries such as France and Spain that have mandated the use of the ISAN standard for cable retransmission royalty processing [83].

Collective Access: Collective Access is used by a growing community of libraries, museums, archives and historical societies around the globe. These partner organizations help it to continually develop new features and keep it current with the real-world needs of today's collecting institutions.

DOI: Large customer base in the intellectual property and wide range of RAs. DOI is growing to become an ISO standard.

System aspect	Handle System	ISAN	Collective Access	DOI
Public Access	possibilities include web and non web based, could be better	through homepage	through webserver	Depends on Handle System for real global access.

Table 7: Public Access

Handle System: Access is possible through various ways, it is not restricted to DNS, but can make use of it. At this time it needs DNS for global resolution via the proxy resolver, until a global URN resolver arrives that can handle direct ”hdl” schema resolving. The possibilities of public access are using the global server proxy, your own local proxy, one from another organisation, building your own web interface with the Client API (OpenHandle) to the global server and using the client Admin tool or the commandline Java resolver that comes with the Handle.net distribution. Still servers can be anywhere and handles are available depending on the read permissions set by the administrator of the handle. For handles values set to be publicly readable, you do not need to authenticate. For private handle values you need authenticate with the handle.
ISAN: The general public can use the ISAN-IA's web site 24/7 (www.isan.org) to access an ISAN with limited metadata shown.
Collective Access: Via the publishing of content, people can search using GUI of the server giving access through guest accounts.
DOI: Mainly resolved via own proxy site, but the same possibilities as the Handle System.

System aspect	Handle System	ISAN	Collective Access	DOI
External Metadata Access	url parameters, OpenHandle or own custom build application	unclear	Open Archives Initiative interoperability	Same as Handle System.

Table 8: External Metadata Access

Handle System: Using ”handle?parameter&parameter” in the Web proxy URL to get specific metadata or using OpenHandle to provide easy access to metadata from handles and use for other systems. Alternatively you can build your own interface.
ISAN: It is mainly included as a reference data element in systems and used to manage and process information about audiovisual works. ISAN states that they facilitates electronic infor-

mation exchanges between different commercial and consumer focused database systems such as a studios financial system, or a theatre chains digital cinema distribution tracking database.

Collective Access: Currently they are working on interoperability with Open Archives Initiative (OAI), which have mechanisms for metadata repository interoperability.

DOI: Primarily available on many websites as links to intellectual property, the metadata is usually not extracted, but can also be extracted externally with OpenHandle or any other handle interface.

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Search	no internal engine, external possible	limited query possible via website	internal search engine	Same as Handle System.

Table 9: Search capabilities

Handle System: Has no search function, but running a dedicated search engine as a separate process, against the underlying handle data stores, off-loads the searching function from the name server, making the name server more efficient.

ISAN: You can query the ISAN database. A query can be made by title or by ISAN. Web site queries will return a limited list of matches displaying only original titles. In order to search for more matches and access more descriptive information, you need to be as a registered user. Registered users can retrieve the complete descriptive information recorded for all works and versions registered in the ISAN database.

Collective Access: Once logged into the system need to use the search option, with the * (star sign) you can get all the published object with the published metadata?

DOI: Same as Handle System.

System aspect	**Handle System**	**ISAN**	**Collective Access**	**DOI**
Backup	internal tool	secure and reliable	external tool needed (manual)	Same as Handle System.

Table 10: Backup

Handle System: Has a backup utility to make a copy of the database and make a checkpoint. The copies need to transferred manually for offline backup
ISAN: Backup strategy of the central database is not publicly known, but ISAN states disaster recovery and failover plans will meet or exceed common industry best practices for security and reliability.
Collective Access: You need to manually copy the database and/or files from their directories
DOI: Same as Handle System

System aspect	Handle System	ISAN	Collective Access	DOI
Administration and Configurability	decentralised and medium configurable	centralised and low configurable	centralised and very highly configurable	Delegated to RA's, but same as Handle System.

Table 11: Administration and Configurability

Handle System: Handles and prefixes can be administrated by anyone with the right access, expect for the prefixes, which can only be created by the GHS administrator. The LHS servers have a small set of configurable setting, which enable you to do most of the basics, including changing database, admins, replicas, ports, logs, etc.
ISAN: Registrants have the same privileges as ISAN Readers plus the right to register new ISANs and V-ISANs.
Collective Access: Owner of software has full administrative rights and it is highly configurable.
DOI: Delegated to RA's and their registrant the same way as with the Handle System.

6 Installing the Handle System Software

I installed the Handle System Software on a Dell PowerEdge 850 with Intel Pentium-D 930 (dual core with VT) and 2GB memory server with Ubuntu 9.04, Apache webserver 2.2.11 and Apache Tomcat 6.026 installed. For multiple servers I used Xen 3.3 to virtually run Ubuntu 9.04 with 256 MB memory.

6.1 Installing the LHS

Starting at the Start page of Handle.net [61], I found the initial steps for getting the Local Handle Service (LHS) and a prefix.

Once I read the License Agreement, which is a public license, similar to an open source license and the System Service Agreement, I needed to agree to pay a registration and annual fee for a prefix. This is used to ensure that prefixes are used in a manner that does not disrupt or interfere with the overall operation of the Handle System. Afterwards I downloaded the software from the distribution download page of Handle.net [60]. A detailed instalment procedure is available through the Handle System Technical Manual [62].

A quick overview of the instalment begins with the requirements for the software. This is at least Java 1.4.2 [63] because the GUI for starting and managing the LHS is made with Java. The main file of the system is a Java application named 'handle.jar', which only needs a working directory to store the configuration, handle databases, keys and logs.

Preparing these files goes with:

```
java -cp handle.jar net.handle.server.SimpleSetup /working/directory
```

The procedure is self explanatory; important is to think about the use of a pass phrase for this public key authentication with the server. Use of the passphrase can make startup more secure, but less easy when I want to automate startup. At the end of the steps I had a populated working directory with at least the files containing the public and private keys, 'config.dct', 'siteinfo.bin' and 'sitebndl.zip'. To get the prefix I need to go through the registration process at the Handle.net Registration page [65] and pay the fees requested. For this research I made direct contact with one of the administrators of the Handle System and requested a test handle. This was generously provided to me once I provided the 'sitebndl.zip' file. This file is used by the Handle System Administrator [64] to create a prefix in the Global Handle Registry at handle '0.NA/prefix' with all its index entries corresponding to my server information, which include the a HS_ADMIN, HS_SITE and HS_PUBKEY. The prefix I got was 10677. To start the server with this prefix I needed to add it into the 'config.dct' configuration file replacing the YOUR_NAMING_AUTHORITY under server_admins and then started the server with:

`java -cp handle.jar net.handle.server.Main /working/directory`.

What happens is that the server looks into its 'config.dct' and 'siteinfo.bin' to know what its HS_SITE configuration looks like. The 'siteinfo.bin' contains the information of the HS_SITE. The 'siteinfo.bin' is setup by the 'handle.jar' during installation, but can also be edited with the Admin GUI, more on that in section 6.4.1. With the following command I can start the Admin GUI called HandleTool:

`java -cp handle.jar net.handle.apps.gui.hadmin.HandleTool`.

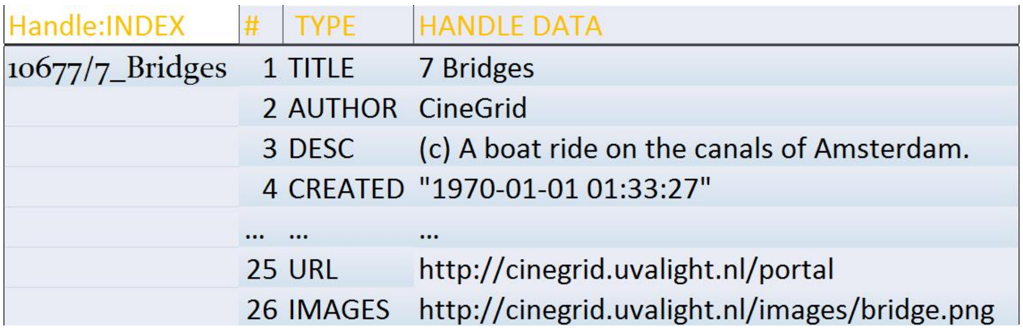
An alternative and more compact version AdminTool is also available, this version is discussed in the Technical Manual [62].

With the Admin tool I can easily manage handles. The first thing I needed to do was use the 'Server Admin/Home Naming Authority' option to tell all servers (if any) that the main server IP address is responsible for the prefix. For this I first need to authenticate with the prefix handle to gain administrative rights. Authentication is done by providing the private key, which is located in the working directory and matches the public key in the HS_PUBKEY and is specified in the HS_ADMIN of the prefix. After this I could query a handle with that prefix also for privileged handle values. Furthermore I could create, modify and remove handles and run batch programs to run operation from a file for the prefix. Screen shots of the tool can be seen in appendix A.

To create a handle I needed to at least provide an Admin value and a handle name. The application limits the number of values to 10 digit decimal long, which means I can have around a billion handle values per handle. More accurately testing the limit for the tool it was 2147483647 in decimal, which is exactly the max of 31 bit, apparently they used the 32nd bit for the -1. In the RFC3651 in section 3.1 the index is specified as an unsigned 32-bit integer, which in the implementation has apparently become 31 bits. Once I populated the Handle System I could use either, the proxy `http://hdl.handle.net/` or my own server, through port 8000, as proxy to resolve a handle. Alternatively I can use the query option in the Admin tool and a command line Java query tool. An example of the resolving of handle 10677/7_Bridges can be seen in figure 4

6.2 Installing OpenHandle

From here is where the Handle System stops providing clear documentation as to what I can do next. The documentation and handle.net site do provide the proxy servlet code [58] used, HDL Client API Javadoc [59] for developers to use and plugins for browsers to resolve Handles directly, which just redirection resolution requests in the browser through the global handle proxy.



Handle:INDEX	#	TYPE	HANDLE DATA
10677/7_Bridges	1	TITLE	7 Bridges
	2	AUTHOR	CineGrid
	3	DESC	(c) A boat ride on the canals of Amsterdam.
	4	CREATED	"1970-01-01 01:33:27"
	...	...	...
	25	URL	http://cinegrid.uvalight.nl/portal
	26	IMAGES	http://cinegrid.uvalight.nl/images/bridge.png

Figure 4: Handle 10677/7_Bridges

An alternative means of accessing the Handle System and getting access to the data is through the use of OpenHandle [66]. Through the ”Getting Started” page[67] on the Google code OpenHandle homepage it is possible to compile the OpenHandle program.

What it does is use the ”HANDLE.NET Client Libraries” to access the Handle System directly and provides a Java servlet front-end that can be used by other programming languages. The installation steps are:

- Using Subversion to checkout the checkout site [69] for the OpenHandle source.
- Downloading and extracting Maven from its homepage [70].
- Adding the maven bin directory to the PATH environment.
- Adding the java main directory to the JAVA_HOME environment.
- Downloading the HANDLE.NET Client Libraries [71] and using the handle.jar file with maven as follows:
  ```
  mvn install:install-file -Dfile=<path-to-handle.jar> \
  -DgroupId=net.handle -DartifactId=handle -Dversion=6 \
  -Dpackaging=jar
  ```
- To compile the source I just needed to run the following command in the checkout directory:
  ```
  mvn compile
  ```

I later choose to do ”mvn package” command to create a war file of the OpenHandle application, which can be deployed in my Tomcat/webapps directory and accessed through port 8080. Figures 27 and 28 in Appendix B show an example of the OpenHandle webpage.

The OpenHandle Google code site also shares a variety of code examples in currently seventeen programming languages [68]. These examples are able to use the OpenHandle program to access the separate fields of handles once I provide the location of the OpenHandle program concatenated with "/handle". Websites can then use this location to resolve handles and receive handle metadata by providing a handle.

6.3 CineGrid metadata test

To test the processing of metadata into a website, I tried recreating the portal of CineGrid Amsterdam [72] a screenshot of the portal is available see in figure 29 in Appendix C. I started off by creating handles for the movies Big Buck Bunny and 7 Bridges to get 10677/Big_Buck_Bunny and 10677/7_Bridges. I then started on the preparations for the webserver and copied part of the source of the portal to get a template (a sreenshot is available in figure 30 in Appendix C) to my webserver. I was eventually able to create a simple CGI Python script that showed a copy of a part CineGrid portal page from my server, only this one is with most of the metadata (title, author and description, links, etc.) filled in from the handles. I was, therefore, able to change the architecture from figure 2 into the "architecture" as in figure 5, replacing the metadata retrieval process with the OpenHandle-Python setup. The resulting figure 31 is also shown in Appendix C. Changes in handles were shown directly after a refreshed webpage, as long as the handle itself was not set to a TTL higher that zero. The default value for handles are typically set to a day.

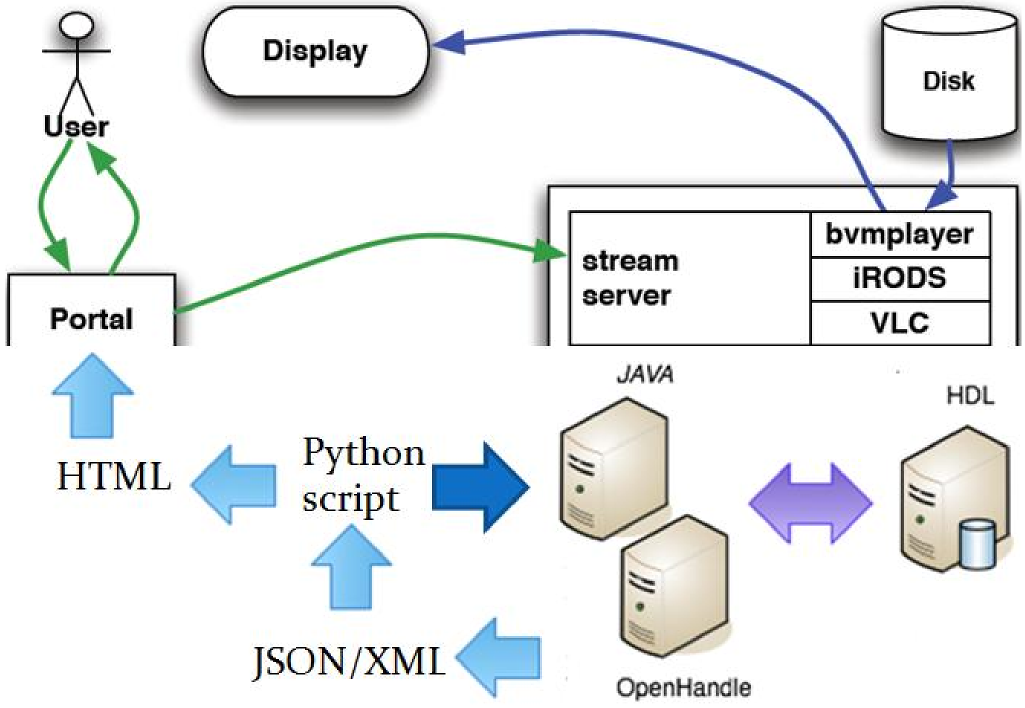


Figure 5: Portal CineGrid Amsterdam New Architecture

6.4 Scalability

6.4.1 Replication

For availability there is the possibility to create mirrors of the primary Handle server. The primary server is the only place where administration of the handles is allowed and which is a requirement of replication by the Handle System to ensure consistency.

To create a mirror I need to install again a server with the Java SimpleTool, only when asked for:

> ”Will this be a ”primary” server (i.e., not a mirror of another server)?(y/n)”

I will choose ’no’ and when prompted choose the primary server’s IP address as the location of the server to replicate and continue as usual. Once installed, I just needed to configure how the servers should authenticate with each other.

In the ’config.dct’ of the mirror I needed to specified the location of the public or secret key of the replicating server, which is usually added as index value to the prefix registration (0.NA/prefix) on the global server. This could be a new value prefix:301 (300 is the location of the public/private key of the administrator of the prefix). Of course I first need to add the public key to the prefix as an HS_PUBKEY value first or a HS_SECKEY for a secret key (the secret key, which is just a passphrase, will be set to Admin read and write only, to prevent non administrators to see the key). On the primary server I also need to add the location of this value in the ’config.dct’ to let the primary know with which keys servers are allowed to do replication. The location of the public/secret key does not have to be in the global prefix handle and can be in any available handle. It can even be part of an entry in the HS_VLIST type of a handle, as long as the primary and mirror can look it up. So this could also be in another handle administered by someone else.

I tested this by putting the public key of the mirror in the handle 10677/Taarik at index 303 and making both the mirror and primary authenticate to that public key. Of course it’s much harder to delete and lose keys when using the global 0.NA/prefix than with the local 10677/Taarik handle. This is because full administrative right to the 0.NA/prefix is controlled by the global server administrator, which gives limited access to local administrators using the handle.

When I have configured how to use the mirror and first start the mirror it will request a dump of the handles from the primary, the time it takes to finish the copies relies, of course, on the size and type of the database of the primary server. Changes to the primary database are recorded with transaction logs, before modifying the database. To stay consistent the primary uses a data centric model, where the mirror polls the primary every n minutes (generally between 1 and 5) with the last transaction ID received and date of last poll. The mirror then gets all the transaction that have occurred since, together with the latest transaction ID.

6.4.2 Fail-Over and resolving

Having mirror servers setup, I can now configure how to access them by adding a HS_SITE type with a new index and a value containing the IP-address of the replicated server to the prefix registration (0.NA/prefix), alternatively I could have load the 'siteinfo.bin' file directly from the replicated server into the new HS_SITE type and not have to configure the IP addresses and such by hand. The extra HS_SITE tells the global server that resolving a handle can be done through an alternative SITE. Appendix A shows a screenshot of the HS_SITE menu and the 'adding server' menu.

To test how this works, I made a Pearl script that looks at the interval between requesting a handle resolve and responds of handles server. I setup 25 test handles with each having a TTL (Time to Live) of 0 second to prevent cache responses and just requested them in two rounds via the following ways:

- using the Java command line resolver that comes with the Handle System software
- 'wget' from the proxy server "hdl.handle.net".
- 'wget' from my own proxy server

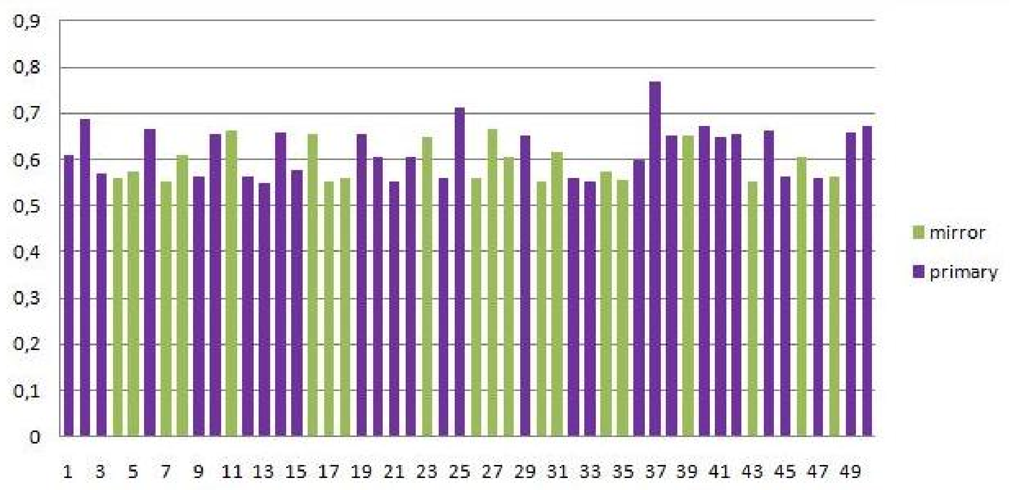


Figure 6: 50x Command Line Tool resolution in seconds from which LHS

Figure 6 shows which LHS eventually is responding to the commandline tool. I can see that this looks pretty "random". What I also see is that when I request a handle, the resolver "randomly" chooses one out of five different servers for the location of the local handle service. From figure 7 this "randomness" is not yet clear. Doing some extra resolutions we see that the request in figure 8 even out the ones in figure 7, a summary of the server is in figure 10. What happens here is that the resolver iteratively continues to search by receiving the HS_SITE information and again

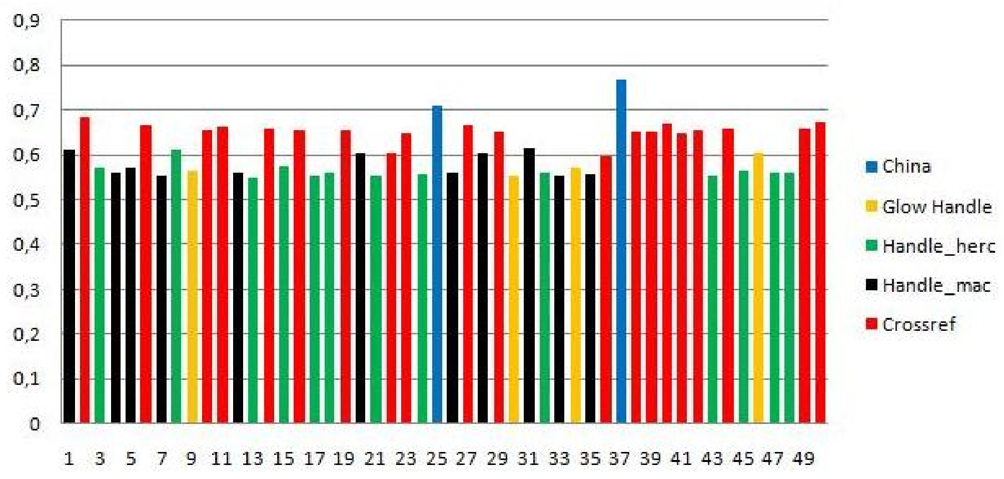


Figure 7: 50x Command Line Tool resolution in seconds through which GHS

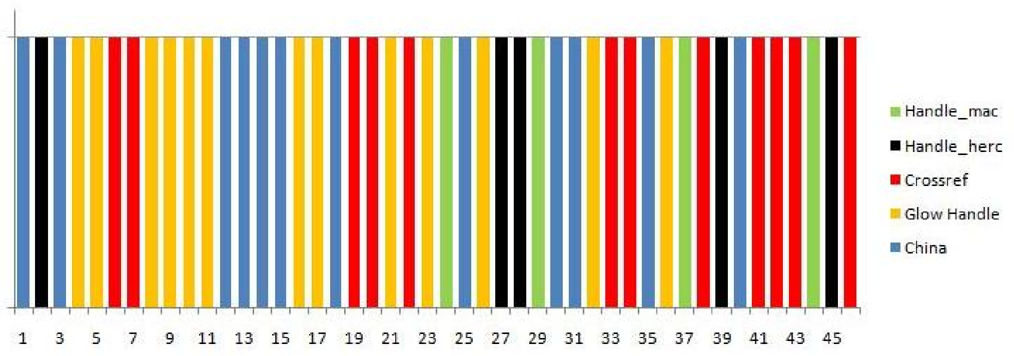


Figure 8: A continuation of figure 7 with 46x extra resolutions to just show randomness of responding servers

randomly chooses between the primary and mirrors (see figure 9 and 6). This could mean that fail-over here does not depend on any other performance variables.

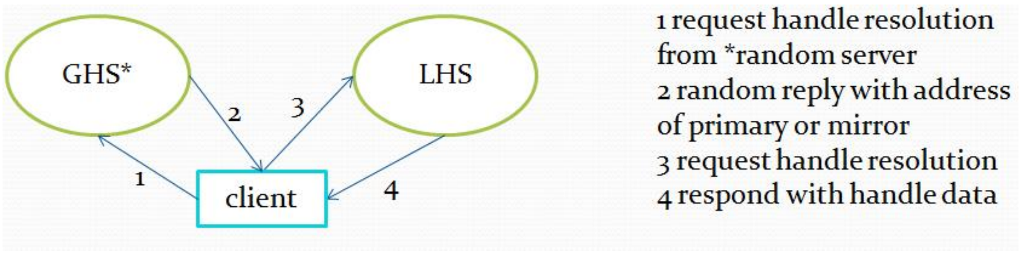

GHS*
LHS
client
1
2
3
4
1 request handle resolution from *random server
2 random reply with address of primary or mirror
3 request handle resolution
4 respond with handle data

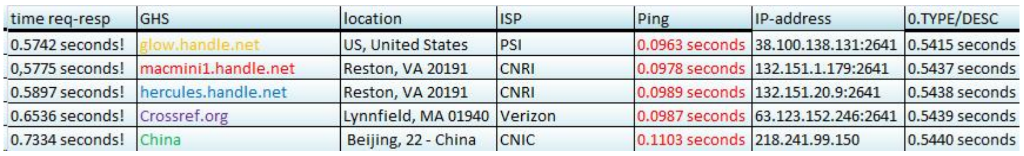
Figure 9: Iteratively Handle Resolving

When I disconnect the primary in a second test (figure 11) I see that when the primary is down the resolver still tries to contact the primary (in the figure this corresponds to the mirror*). What happens is that the tool tries to contact the primary server three time every time before choosing a mirror. This occurs 50% of the time when the primary is down. This confirm that fail-over is really random and static for at least the command line tool. In figure 10 I can see that a resolution takes in average about 600 milliseconds when the local server is on-line. A ping to one of the servers has an average of 90 milliseconds.

At handle.net they already did a resolution test where the server and client were on the same machine [75], their results were 30000 successful resolution in 8934 milliseconds which is 0,2978 milliseconds per resolution. They eliminated network delay, which I have not eliminated, still my result show how much larger this network delay can get. What is interesting is that the delay, when the primary is not available, goes up to an average of 6000 milliseconds.

Resolving a small GHS local handle (going through only step 1 and 2 from the iterative resolution) i.e. 0.TYPE/DESC (see figure 10) the delay goes down from 600 to about 540 milliseconds, because we do not need to continue searching onto the LHS after the first request, but this is still much more than the ping time, which equals the roundtrip time we would expect since local resolution is pretty fast according to the test by handle.net. The tool apparently has some unknown overhead, making it slow, but still workable as long as the servers are all online.

Doing the same with the handle.net proxy I see that in figure 12 resolution is faster, ignoring the ARP at the beginning. What is also different from the command line tool, is that instead of iteratively searching for the handle by us, the proxy recursively returns the result for us (see figure 13). For this I had tried two different resolutions, the first was an Authoritative Query, which according to handle.net ”forces the proxy to bypass the cache and go directly to the responsible server and then refresh the cache with the data for that handle.” [75]. This apparently does not use the mirror because once the primary was down no results could be returned. Doing a standard query I see the same speed in request, interesting is that when the primary fails it takes longer to timeout than with the commandline tool (see figure 14), but the proxy does not continuously retry to connect to the primary as is the case with the command line tool. It is much more flexible to changes as it only did it in 10% of the resolutions. Resolving again a small GHS local handle (going through only step 1 and 4 from the recursive resolution) i.e. 0.TYPE/DESC (see figure 15) we get again a much smaller responds time of about 200 milliseconds instead of about 300 milliseconds. This is about a ping time between the GHS and LHS, so that is to be expected for



time req-resp	GHS	location	ISP	Ping	IP-address	0.TYPE/DESC
0.5742 seconds!	glow.handle.net	US, United States	PSI	0.0963 seconds	38.100.138.131:2641	0.5415 seconds
0,5775 seconds!	macmini1.handle.net	Reston, VA 20191	CNRI	0.0978 seconds	132.151.1.179:2641	0.5437 seconds
0.5897 seconds!	hercules.handle.net	Reston, VA 20191	CNRI	0.0989 seconds	132.151.20.9:2641	0.5438 seconds
0.6536 seconds!	Crossref.org	Lynnfield, MA 01940	Verizon	0.0987 seconds	63.123.152.246:2641	0.5439 seconds
0.7334 seconds!	China	Beijing, 22 - China	CNIC	0.1103 seconds	218.241.99.150	0.5440 seconds

Figure 10: the current five GHSs used by the Java commandline tool, with their average time between request and responds of a resolution, their location, ISP, roundtrip time (ping), IP and local GHS resolution.

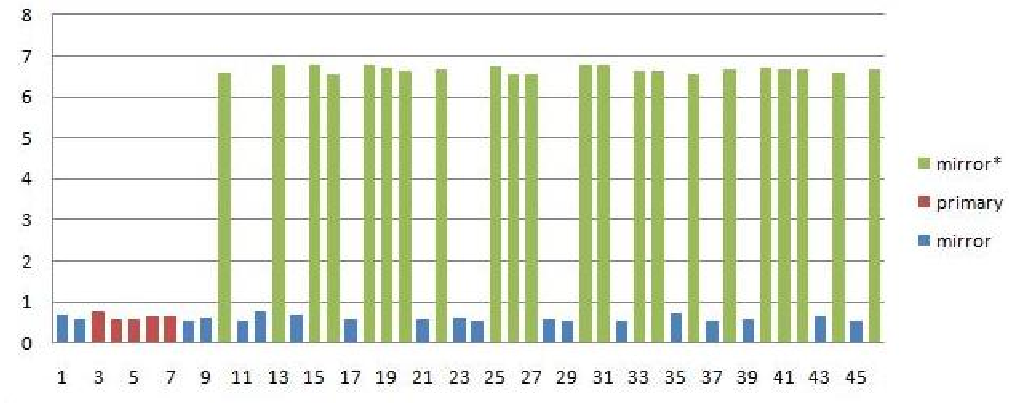


Figure 11: 46x Command Line Tool resolution in seconds from which LHS when primary down

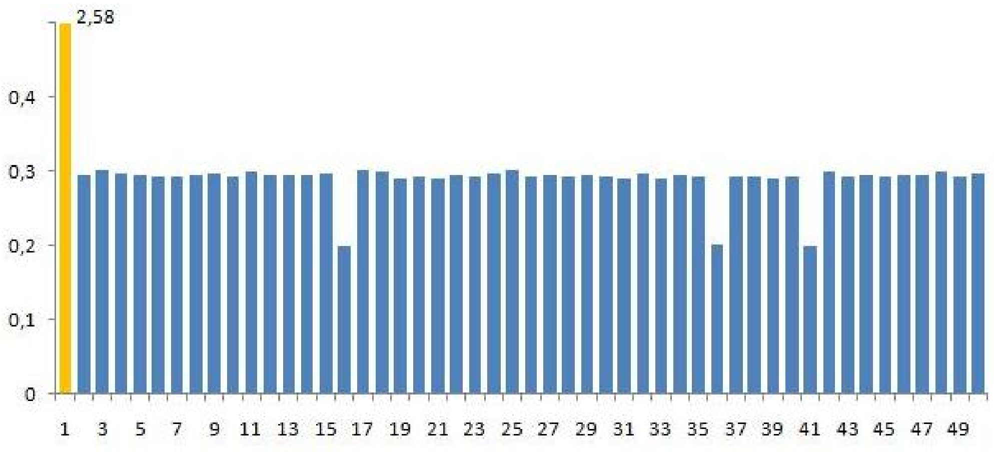


Figure 12: 50x Web Proxy resolution in seconds

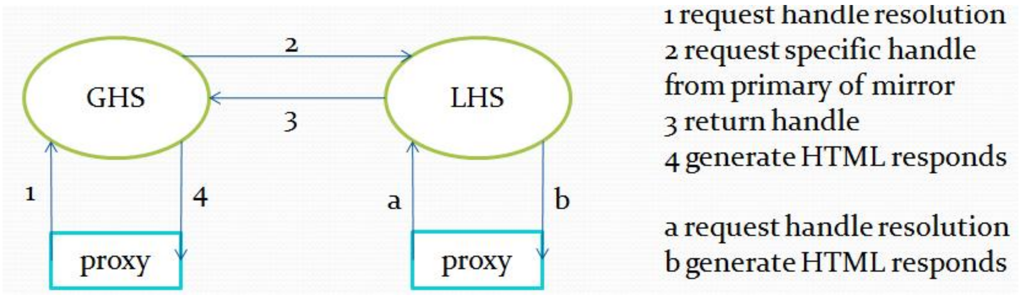

GHS
LHS
2
3
1
4
a
b
proxy
proxy
1 request handle resolution
2 request specific handle from primary of mirror
3 return handle
4 generate HTML responds
a request handle resolution
b generate HTML responds

Figure 13: Iteratively Handle Resolving

the time of step 2 and 3 combined from the recursive resolution. Still a ping to the web proxy is half the time and leaves almost 100 milliseconds of unknown overhead.

Finally resolving directly from my own proxy does the job in about 10 milliseconds because it does not need to contact the global server (This is even faster than ping to localhost(see figure 15)). Of course there is no possibility of fail-over because once the server goes down so does the

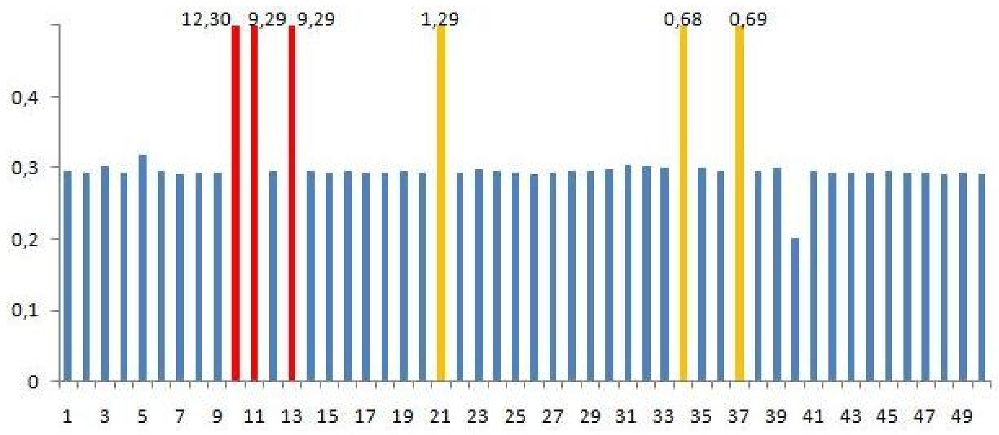
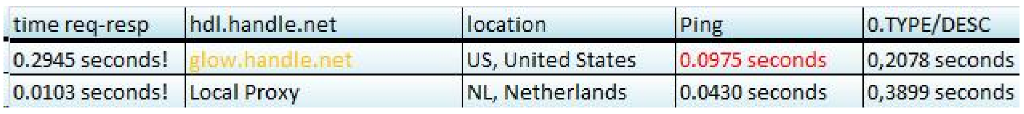


Figure 14: 50x Web Proxy resolution in seconds when primary down



time req-resp	hdl.handle.net	location	Ping	0.TYPE/DESC
0.2945 seconds!	glow.handle.net	US, United States	0.0975 seconds	0,2078 seconds
0.0103 seconds!	Local Proxy	NL, Netherlands	0.0430 seconds	0,3899 seconds

Figure 15: the GHS behind hdl.handle.net and its location roundtrip time (ping) and local handle resolution.

proxy.

6.4.3 Distribution

Next to replication there is also the option of distributing my handles to a group of servers (a.k.a. SITE). This works the same way as replication, I first used the java SimpleTool to make a new non primary. The configuration difference between replication and distributed is mainly in 'siteinfo.bin' and the prefix registration (0.NA/prefix). Here instead of adding a new HS_SITE and adding a single server IP address, I add all the server IP addresses, that belong to the same SITE to which I want to distribute my handles, to the same new HS_SITE and give the servers different 'serverIDs' to distinguish them from each other. A crucial next step is to save the HS_SITE to a file, namely the 'siteinfo.bin' in the working directory of all the servers. (This can be done with the AdminTool and is used during startup to do some checks.) And also changing the 'serverID' in the 'config.dct' to correspond to the right server in the 'siteinfo.bin'. If they do not match, the server will not start. Once I finished the configuration the server will have different handles in their database based on a chosen hashing range. When creating a handle a MD5 hash is done over either, the prefix, suffix or entire handle. The hash is divided by the number of servers and the remainder is used as sequence number to the appropriate server. A picture of the complete distributed model of the Handle System can be seen in figure 16. What we see is the Handle System architecture with at the top the services. Each service can (when zooming in) consist of multiple replicated

SITES for availability and of which, the servers on these SITES (when zooming in further) have the same group of handles distributed to their databases for performance. Although the GHS seems special, it is also a service and does the same replication and distribution to stay available and increase performance.

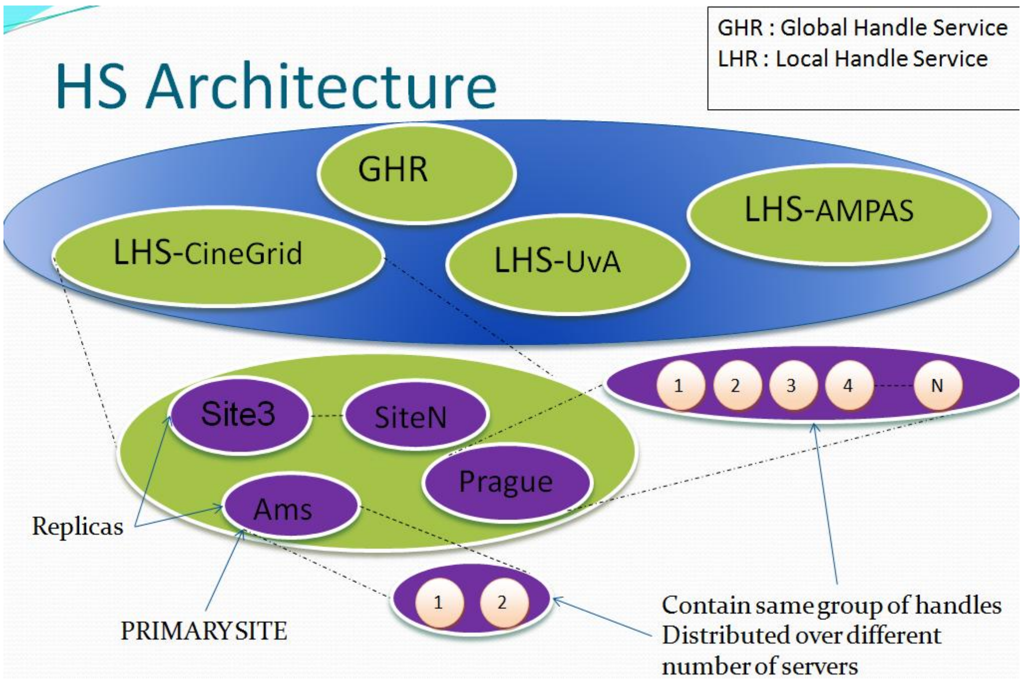

HS Architecture
GHR : Global Handle Service
LHR : Local Handle Service
GHR
LHS-AMPAS
LHS-CineGrid
LHS-UvA
Site3
SiteN
Prague
Ams
1
2
3
4
N
Replicas
PRIMARY SITE
1
2
Contain same group of handles Distributed over different number of servers

Figure 16: Handle System Architecture

There is however one issue, which I looked at by just trying it out. I cannot add a server to an existing HS_SITE without losing handles because this will change the hash calculation used to distribute the handles. So adding an extra server is allowed, but will only work correctly if I currently have no handles in the system, or if I start with all the servers in a SITE I want from the beginning. A work around for this issue for a mirror SITE is to remove the SITE and start over, this will cause the servers to "redump" the primary database. Of course this can take some time when I have a lot of handles.

When I want to put more servers in a primary SITE, then the workaround will not work just as easy. There is a procedure in the Technical Manual to split a primary server into multiple servers, but it requires much more effort and is even more complex if there are already multiple servers running and requires assistance from the GHS administrators, according to the Technical Manual [62].

6.5 Multiple Primaries

When the primaries are distributed in their own SITE, then the servers are only allowed to work with handles on their own database. For access right on the other primary servers I need to "HOME" the server. Homing tells all the servers in the SITE that the server is responsible for the handles in that prefix. As explained in section 6.4.1 it is necessary that only the primary server does all the changes to keep the databases consistent. There is however an option for "Multiple Primaries" [4] in the HS_SITE type (see screenshot in figure 24 in Appendix A) when I register SITES for my prefix. There is also an HS_Primary type for handles to indicate which primary is the handles main server. Taking a look at an overview of registered types at handle.net [76], I see that HS_Primary is strangely enough not registered. Testing this by using the option "Multiple Primaries" checked for a primary SITE with two servers in the same SITE and even in separating SITES did not do anything.

From Sean Reilly (lead developer at handle.net) I got informed that the HS_PRIMARY mechanism was not implemented and was never really designed properly. They have however designed and implemented a multi-primary mechanism that works a bit differently, but is only available in the 7.0 version of the handle software, which is finished, but not yet released.

6.6 Remote Administration

The Handle System is capable of giving administrative rights per handle. This is easily done by just adding a HS_ADMIN with the location of a public or a secret key to an index value of the handle. The location of the public/secret key can be stored as stated in section 6.4.1 anywhere, it can be in the handle itself or even in a more global/central location/handle, I am free to choose. The next step is to simply authenticate for instance with the Java Admin Tool that comes with the Handle.net distribution or an Admin Web interface (which I need to setup and configure myself with e.g. Apache TomCat) and providing my private key or secret key together with the handle and index of the HS_ADMIN and I will get the access to the handle as defined in the HS_ADMIN, e.g. handle permissions for deleting values. A full list of options can be seen in the screenshot of HS_ADMIN in figure 26 in Appendix A.

6.7 Handle editing

When administrating thousands and may be even millions of handles, you do not want to manual edit every handle when you decide to add a general index value to all handles. To speed up these processes you can make use of the 'Batch' utility that comes with the AdminTool. The utility takes input in a text format e.g. adding some values to a handle could look like this:

```
ADD CineGrid/7_Bridges
```

```
123 RATING 86400 1110 UTF8 Five Stars
124 AWARDS 86400 1110 UTF8 Best Picture 2010

ADD CineGrid/Big_Buck_Bunny
123 RATING 86400 1110 UTF8 No Stars
124 AWARDS 86400 1110 UTF8 Worst Picture 2010
```

Operations are space separated from handle names, index values with variables separated by spaces are on a new line and multiple handles are separated by a blank line. All the operations are:

- CREATE, here you need to supply the handle name, all the index values and data.
- ADD, just supply the handle name with the entries you want to add with their data.
- MODIFY, just like add, but it is used to replace the index values with the supplied values.
- DELETE, just supply the handle name e.g. CineGrid/7_Bridges.
- REMOVE, here you indicate the index you want to remove e.g. 4,5,7:CineGrid/7_Bridges.

So I could easily generate such a text file from a list of handles and list with new "metadata" and just feed the resulting text file into the batch program. Too speed up the batch process, I can also split a job, by using more independent files. The batch utility allows for submission of multiple jobs, so I can run them in parallel. The next release of the Handle System software, which still needs to be releases included a jython interpreter that makes it easy to write batch updates, unfortunately I did not have time to look into that part. What I did look at is the maximum amount of lines you can submit in a text file to the Batch utility in the AdminTool. When I exceeded 7707 lines I got a "java.net.SocketException: Connection reset sending HTTP request to server". The buffer size for the Java Socket was probably reached. There is also a commandline version, but I could not get the authentication right, it kept telling me that the authorisation is "null" for some unknown reason.

6.8 Handle delegation

Handles are persistent only as long as they are available and contain the correct metadata and objects references. Sometimes handles need to be moves to different part of an organisation or even to totally different organisations. There are only a couple of situations possible here.

- The organisation is using the main prefix e.g. "main_prefix/local_name". This means that all the Local Handle Service SITES have the same handles and administrative rights can be separated per handle and, therefore, also changed per handle without physically moving the handle between databases.

- The organisational is using subprefixes for different department of it organisation e.g. ”main_prefix.sub_prefix/local_name”. Here is where it gets trickier. The Handle System has an HS_ALIAS type to let me redirect to the handle now responsible for the content or you can do it the other way around and let the other department make if necessary a handle with an HS_ALIAS to the original handle. This way you do not need to move the handle to another database, but I still need to give the (new) person from possibly the other department who is going to be responsible for the handle administrative rights to the handle on my database. With the Handle System this is not a problem as might be the case with other systems, like DNS.

6.9 Database possibilities

The Handle.net distribution comes with a built in Java database to store handles. This database is good enough for a limited amount of handles. Once this number grows you will need a better database. The distribution is able to work with the following alternatives:

- The Berkeley DB JE database, which is very efficient, scalable and a widely used open source database.
- SQL database, which allows greater control over data and is able to handle complex data query. The Handle System Technical Manual [62] shows the configurations needed in the ’config.dct’ for MySQL[77], ORACLE[78] and PostgreSQL[79].

6.10 Configuring Global Handle Server locally

The Technical Manual gives an option to locally run you LHS without the GHS, unfortunately this does not work if you have more than one local handle gateway. The solution is to use a tool LocalInfoJPanel part of handle.jar file, but apparently this is no longer part of the latest version of handle.jar. According to Jane Euler (main administrator at handle.net):

> ”you can’t pick a number to use as a prefix for your independent handle service without going through CNRI. This goes against the Handle System Service Agreement. I would need to assign you a prefix to use and make a record of it even though resolution of the handles would not be using the proxies or Global. Among other things, this helps prevent any collision of prefixes at a later time.” [80]

I thought that they probably removed the tool to prevent the use of local prefixes without their permission, but apparently they left it out by mistake and it is still possible if you have just one

SITE. How it work, is that you use the DBTool that comes with the handle.jar to edit the local files used to check if you are homed to a prefix and, therefore, responsible for the prefix, but because of time constraints I did not finish testing this part. In the new release the "LocalInfoJPanel" will probably be back and testing it would be a good thing for future work.

7 Applicability of handles in AMPAS/CineGrid.org

In this section I will look at the applicability on the Handle System in the AMPAS/CineGrid.org community. I will look into the following parts and comment on issues that may come up.

- Naming local handles.
- How to use metadata?
- Extending metadata in handles.
- How to use the prefix?
- What about preservation?
- How good is the resolution?
- Cost for handles?
- System requirements?

Naming local handles: With the Handle System you can keep your local naming structure for object and even use them as namespace for your handles. Although people will want to give handles names that they can recognize it is best to use a non descriptive name for handles, this will keeps the handles unique and persistent over time more efficiently. Because it's more efficient to change the attributes of handles, than it is to change a handle itself. Changing handles to avoid collisions would become a continuous operation. Running a dedicated search engine as a separate process against the handle database, a search for e.g. a description returns all the handles with the matching description. Comparing to other naming systems with unique and persistent identifiers you are much more free to use your own and even reuse something, like the ISAN identifier, for you local handles namespace or if you want just have it as an attribute (metadata).

How to use metadata: The Handle System does not specify nor restrict the way to use metadata. It just provides an efficient way to store and link metadata and object with your handles. When making use of metadata schemas for any digital content it is best to start with the Dublin Core, this is one of the most used schemas for online content because of its general specification for use of metadata. Furthermore many metadata repositories make use of the Dublin Core and provide interoperability between repositories.

When making use of metadata for intellectual property then the *indecs* schema is more appropriate because the DOI is already using this for millions of online content and easier interoperability with those content would be possible.

When looking into the AMPAS situation I see that the MPEG7 with optionally MPEG21 schema is more appropriate as it is developed by the Moving Picture Experts Group and is specialistic for the content mostly used by AMPAS. But the schema could give problems when looking at the richness and related complexity.

Extending metadata in handles: Within CineGrid it could happen that extra attributes have to be added to a lot of the handles. There are a couple of possibilities with handles. Either all the handles need to be modified, the Handle System provides a fast batch process to accomplish this. Another possibility is to group metadata into a handle and make other handles reference that handle, of course, this requires the selection of metadata to be the same for all those handles. When new metadata needs to be added, which is the same for the whole group, you just need to add it to the referenced handle containing the grouped metadata.

How to use the prefix: The prefix of the Handle System enables global resolution of handles, you can compare the prefix to the domain name in DNS, only instead of having a whole tree of vertically distributed servers with separate administrative domains, you just have the global server, which can be horizontally replicated and distributed and still be the same administrative domain. This means that you can scale the prefix much more efficiently than with DNS. When looking at the CineGrid community, which consist of various members all over the world you need to decide how to manage the Handle System itself. Since it is not linked to any digital content you do not need to have it running at every member SITE. Just, like the Global Handle Service, you can have just one primary Local Handle Service, which requires just one prefix e.g. "CineGrid/" . The size of the Handle System does not have any non hardware limits because it can be replicated to many SITES for availability and it can at the same time also distribute its handles to multiple servers for performance. These replicated SITES can be maintained at different members, which they can at the same time use as resolvers for the handles. And because the handles are also located at their own replicated servers, resolution will be very fast. This can quickly become a feasible solution because handles can be administrated per handle in the Handle System by anyone having the right access key.

Another possibility is to use a primary LHS at every member, this means that every member has its own set of handles and resolution will be done through the GHS. In this situation a subprefix will be the best way to distinguish the members from each other. It's possible to give members different prefixes while still showing relation to each other. This is done by using a common part for all the members, resulting in prefixes, like "CineGrid.ampas/". (Although this is also not mandatory and every member can have their own prefix, since a subprefix costs the same as a prefix) A third option is to combine the previous two and use the main prefix for content you want to share quickly with other members and the subprefix for content you want to share only locally. Only this would mean that you could end up with the same information twice on your machines for possibly a lot of handles.

How good is the resolution: The resolution is currently not as global as DNS, currently it is done through Web proxies using DNS and client programs that directly communicate with the GHS, but can also be used globally. The 'hdl' schema is already registered as info URI and has potential of becoming a registered URN. But a problem for some people might be that currently there are only five sites that provide resolution, four of them are in the US and just one is in China. In China CNRI has already been able to collaborate with the China Internet Network Information Center (CNNIC), the state network information center of China. CNNIC is responsible for the ".CN" country code top level domain (ccTLD) and Chinese Domain Name (CDN) system. A Handle-DNS integration system has been developed to integrate Handles with DNS through the .cn domain.

Testing the resolution process with client programs and web proxies, I came to the conclusion that resolution is not as fast as DNS and that it does not always efficiently handle fail-over when a LHS server fails to respond, but it still resolves the handles eventually. Fail-over would require much more effort with other systems, like DNS, when comparing to the simple setup needed for the Handle System.

Although by using this resolution some efficiency will be lost, CineGrid handles will not only be available for the community, but also to the public.

Cost for handles: When making use of the Handle System you need to register for a prefix. The Handle System is capable of working without a GHS, but this is not recommended by CNRI and would go against the Handle System Service Agreement. You can, of course, discus some arrangement with CNRI concerning this. But the standard registration fee still is 50$ annual free per prefix and 50$ per registration request no matter the amount of prefixes. An alternative to using prefixes is using single handles, for this you can acquire a DOI handle from an IDF (International DOI Foundation) Registration Agency (RA). Usually they pay IDF a couple of cents per handle, but cost for registrant themselves can vary. How this works is that the IDF has already registered a prefix, namely 10, at the Global Handle Service and is using it together with their policy and the *indecs* metadata scheme to provide handles to others. They sell handles by delegating subprefixes of this 10 prefix (which they have also registered at the GHS) to Registry Agencies, which are members of IDF and which stores customer handle on their local servers or possible IDF owned servers. ISAN also has a central database for their identifiers, the cost for one identifier is 35CHF (Swiss Franc), which is currently little over 32$. The interest in ISAN is growing because, e.g. of its mandatory use with Blu-ray standards.

What about preservation: Of course when looking at preservation, your prefix will be maintained by the GHS, but the problem here is keeping you own LHS running all the time. The LHSs contain the important data you want to preserve, the Handle System provides preservation on the LHS by using replication to prevent bitrot, storage of metadata to prevent loss of meaning and signing to ensure authenticity. What it does not take care of is obsolescence. The only alternative way to preserve the data besides doing it yourself is to use another party, ISAN is one of those parties. They not only assign an identifier to your object and store the metadata on their central site in Geneva, Switzerland, they also promise to keep your data preserved. When acquiring your identifier you also pay for the preservation of your identifier with its metadata.

System requirements: The Handle System is a very light program, it uses a small 2MB Java application for most of the operation, the size of the Handle System is, therefore, more dependent on the database type and usage. The Handle System is more, like middleware and works alongside other utilities, like a registry and repository. Other data management systems, like Collective Access, provide a bigger package and can, therefore, have more features, this does not mean that all those features are necessary. The Handle System contains very good basic elements. Handle.net [75] has done a benchmark of the Handle System where they used the new Berkeley DB Java Edition database on a PowerBook G4 laptop with 1.5Ghz G4 CPU, 1 GB DDR RAM, Mac OS X 10.4. In this configuration, the client and server were both on the same machine to eliminate network delay. The best run times with 30 parallel threads, each performing 10,000 queries were:

- milliseconds: 8,934
- requests/second: 33,579
- successful resolutions: 30,000
- resolution failures: 0

8 Conclusion

My research shows that proper use of identifier and metadata for digital content is an important issue, reviewing my research questions from section 1.1 we see the following:

How are object identifiers being used to store find, retrieve and preserve digital content and handle the ever growing supply of content?

Object identifiers are being used by different systems in different ways, important is to use it in a unique and persistent way. This ensures that content that does not clash with other content and that it stays available over time. Important characteristics the help handle the growing supply of content are location independence, replication, distribution, migration, refreshing and metadata.

How is metadata being used to support this process and what are the important requirements?

Metadata describes an information resource, or helps provide access to an information resource. A collection of such metadata elements may describe one or many information resources. There are hundreds of schemas that help you get a standard way of using metadata, they all have different characteristics. Examples are generic minimalist metadata schemas, like Dublin Core and specilistic rich, like IEEE LOM. But eventually different communities seek to meet the specific needs of their members and are creating their own mix.

How does the Handle System compare here and can it be used to make storage, search, retrieval and preservation of digital content more efficient and reliable within AMPAS/CineGrid in particular?

The Handle System is a fairly good system for storage, retrieval and preservation of metadata and location of digital content, although for searching you need to use a search engine. The use of handles provide unique and persistent identifier for digital content. The use of replication and distribution in combination with the LHS and GHS ensures that digital content can continue to grow and still be available without loss of performance. Although the Handle System is much more than other naming systems it is not a complete product, like other data management systems and can be regarded as a form of middleware that can be incorporated into other systems.

9 Future work

Recently CNRI released its general-purpose Digital Object Repository software [73]. It comprises the second major component of the long-term Digital Object Architecture work [37] of which, the Handle System is the first piece. The third and final piece, the Digital Object Registry [74], will be also be released soon. See figure 17 for a picture. The continued use and evolution of the Handle System is in no way dependent on the repository or registry component, but may be useful in combination with Handles. The repository software, client, server and utilities are freely available under an open source style license. A study of those parts of the Digital Object Architecture work would also be an interesting study. Furthermore examining any new feature that may be released in the new version (version 7) of the Handle System software, e.g. the jython interpreter, the modified multiple primary option and the Local Global Handle Service option

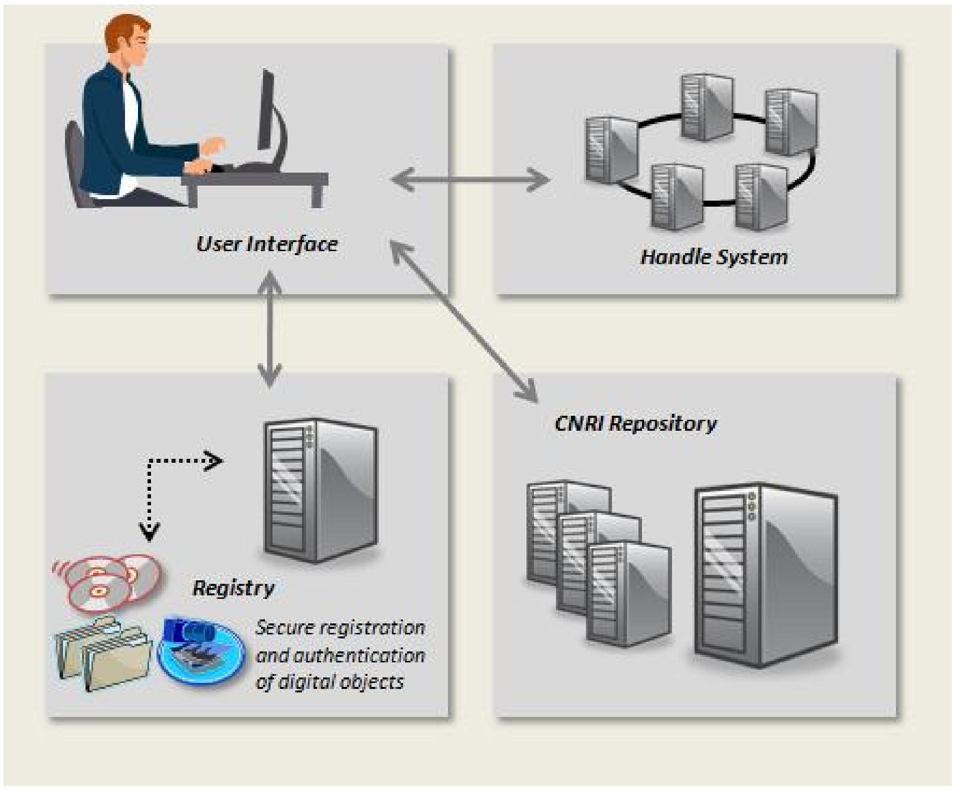


Figure 17: Digital Object Architecture Source:http://www.cnri.reston.va.us/doa.html

Figure 17 shows the Handle System, the Repository used for storage and access of digital content and the Registry provides secure registration and authentication for digital content.

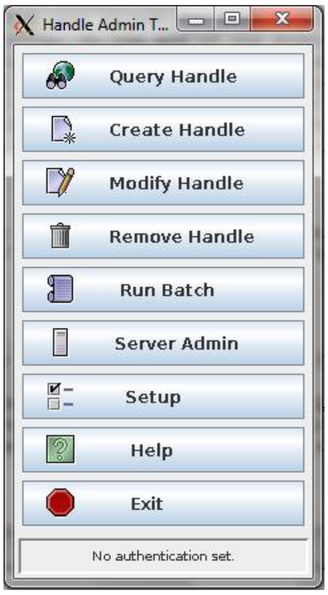
10 Acknowledgments

I would like to thank the following people for their help while conducting this research project.

- Paola Grosso, for her guidance throughout the research project.
- Ralph Koning, for sharing his view and knowledge of CineGrid
- Andy Maltz, for bringing me into contact with the people at Handle.net
- And lastly, those who gave their feedback on this document.

11 Appendix A

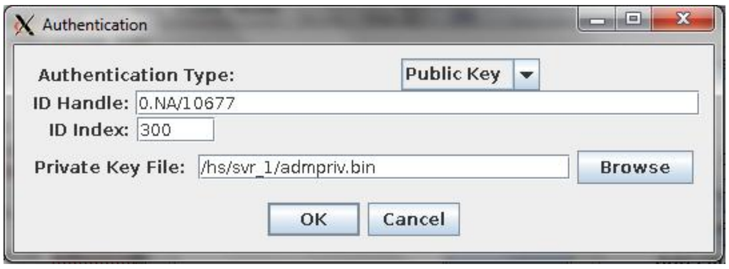
Figure 18 shows the main menu of the Admin Tool, here you can query handles, home and authenticate through the Server Admin menu and create, remove and modify handles at a push of a button. And you can also do this with the batch menu.



No authentication set.

Figure 18: Main Windows Admin Tool

Figure 19 and 20 shows the authentication menu, here you need to match the public key in an index of a handle with your private key, or match the secret key in an index of a handle with your secret plain text.

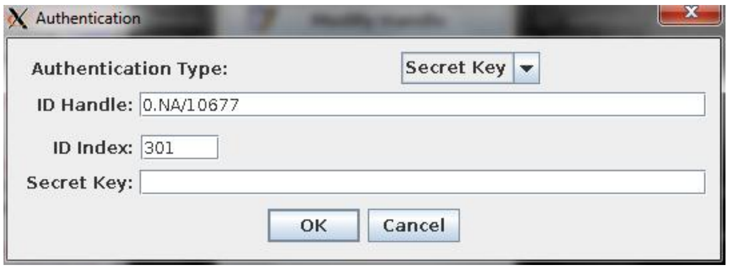


Authentication Type: Public Key
ID Handle: 0.NA/10677
ID Index: 300
Private Key File: /hs/svr_1/admpriv.bin

Figure 19: Authentication Admin Tool

Figure 21 shows the information needed to home a prefix. This only works if the authentication is successful

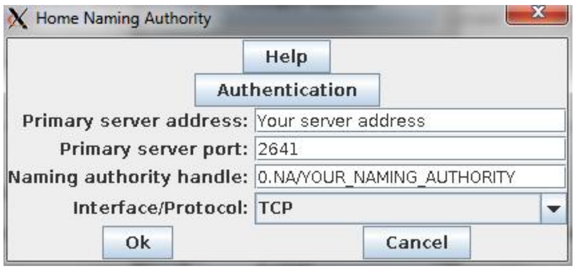
Figure 22 shows the create menu with the minimum values needed already filled in, namely the name and ADMIN value. The delete and modify menu are almost similar or a subset of this menu.



ID Handle: 0.NA/10677

ID Index: 301

Figure 20: Authentication Admin Tool



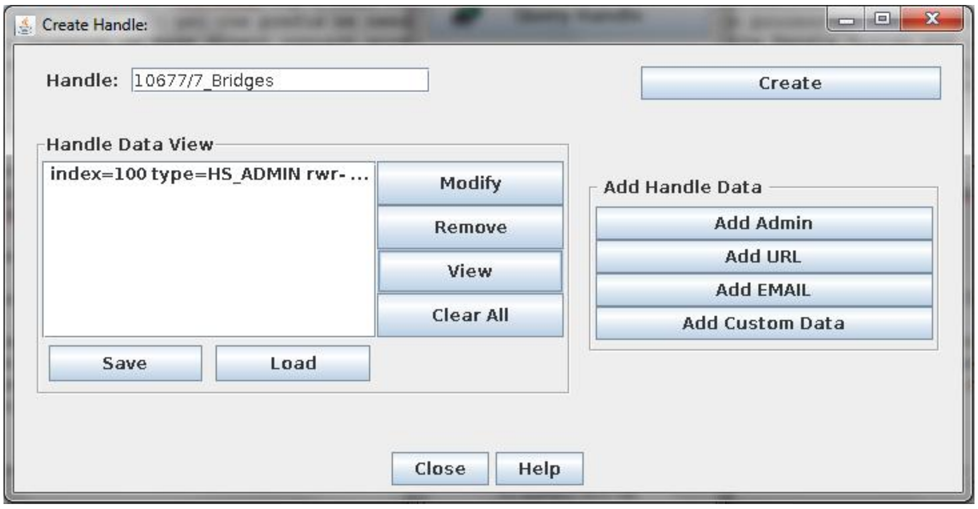
Primary server address: Your server address

Primary server port: 2641

Naming authority handle: 0.NA/YOUR_NAMING_AUTHORITY

Interface/Protocol: TCP

Figure 21: Home Admin Tool



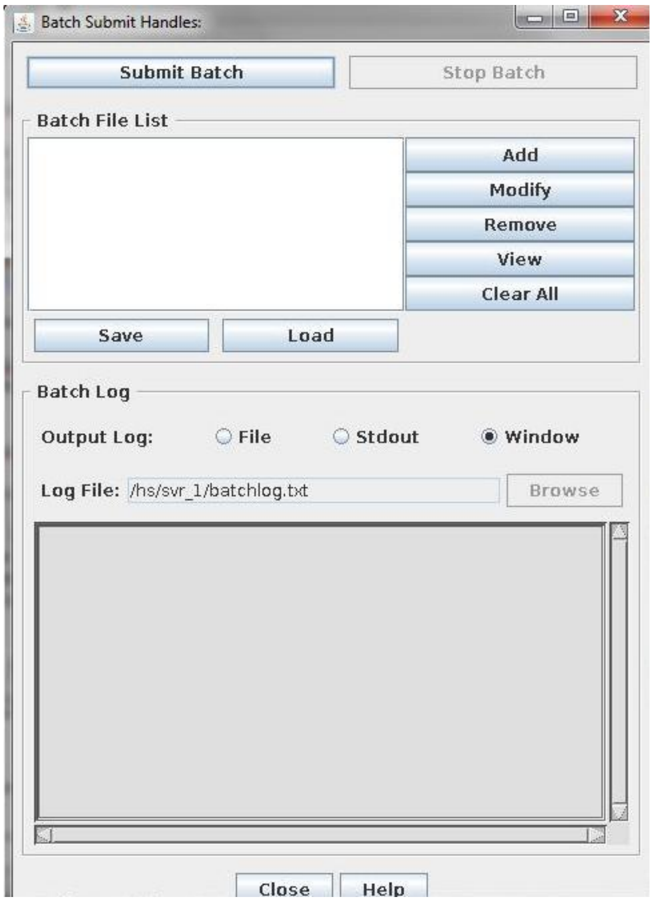
Handle: 10677/7_Bridges

index=100 type=HS_ADMIN rwr- ...

Figure 22: Create Admin Tool

Figure 23 shows the batch menu, you can submit files by adding them to the batch list.

Figure 24 show the configuration menu for HS_SITE reachable though the 'custom data' button in the create menu. The configuration can be saved as a 'siteinfo.bin' or loaded from a 'siteinfo.bin' file. If done manually then the servers can be added through the menu in figure 25. Here you give an IP-address, serverID, public key or generate one and specify the protocols and ports used to communicate with others. The serverID is used in the 'config.dct' to know which of servers the current server is.



Output Log: ( ) File ( ) Stdout [✓ Window]

Log File: /hs/svr_1/batchlog.txt

Figure 23: Batch Admin Tool

Figure 26 show the configuration option of the HS_ADMIN type i.e. location of the key, permissions at handle level and permissions and prefix level.

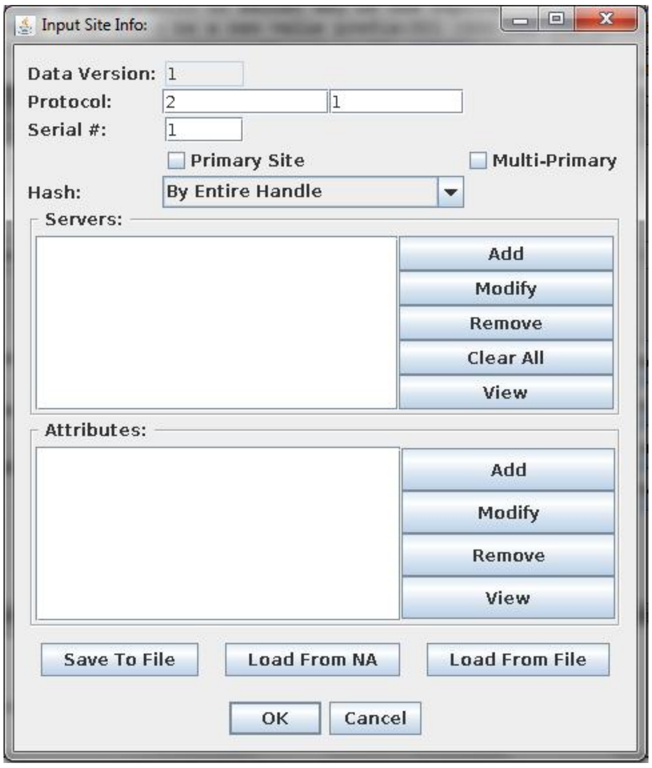


Figure 24: HS_SITE configuration

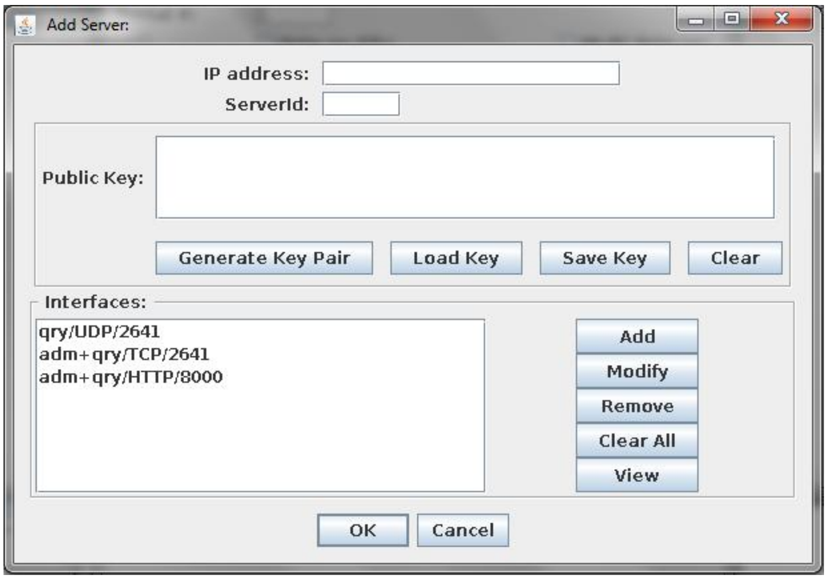
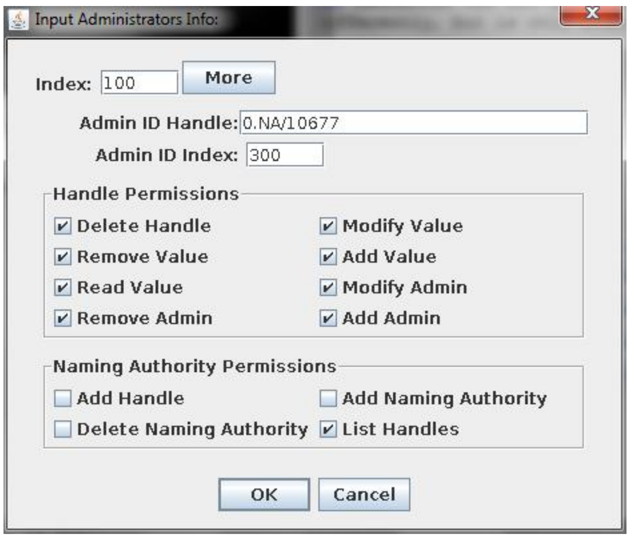


Figure 25: Adding server to HS_SITE



Input Administrators Info:

Index: 100

Admin ID Handle: 0.NA/10677

Admin ID Index: 300

Handle Permissions

- [✓] Delete Handle
- [✓] Modify Value
- [✓] Remove Value
- [✓] Add Value
- [✓] Read Value
- [✓] Modify Admin
- [✓] Remove Admin
- [✓] Add Admin

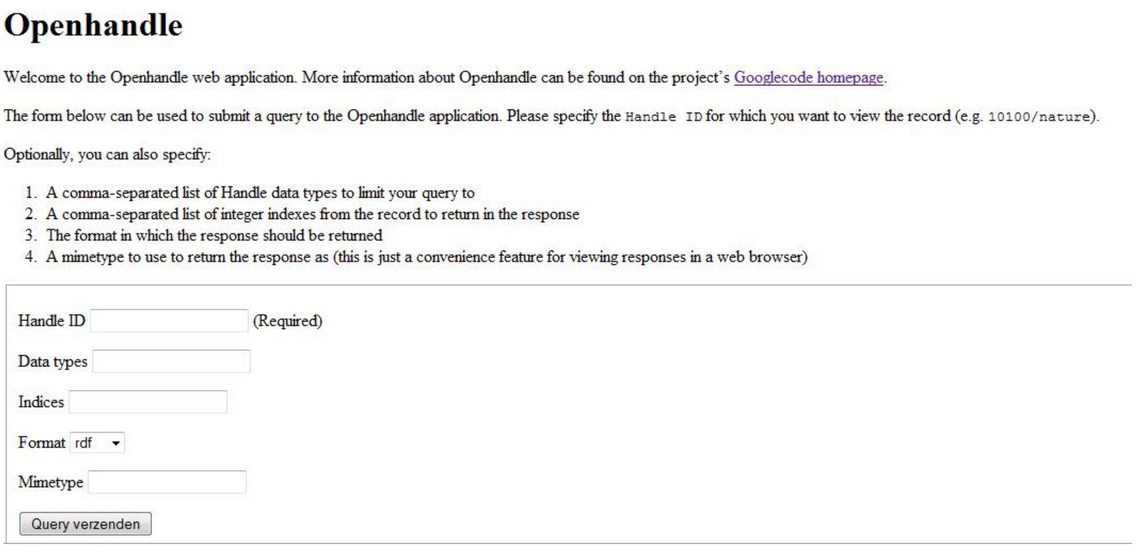
Naming Authority Permissions

- [ ] Add Handle
- [ ] Add Naming Authority
- [ ] Delete Naming Authority
- [✓] List Handles

Figure 26: Configuring HS_ADMIN

12 Appendix B

Figure 27 shows the web interface for OpenHandle, once you provide a handle, you can get the metadata returned in various formats, figure 28 show the result.



Openhandle

Welcome to the Openhandle web application. More information about Openhandle can be found on the project's Googlecode homepage.

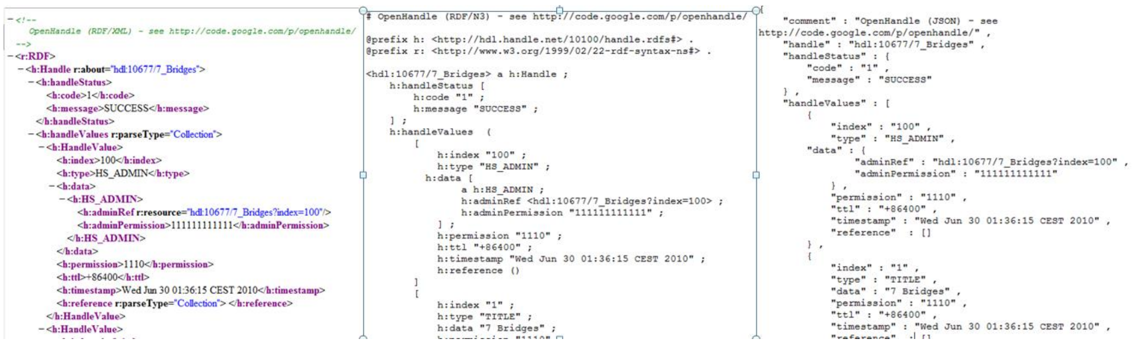
The form below can be used to submit a query to the Openhandle application. Please specify the `Handle ID` for which you want to view the record (e.g. `10100/nature`).

Optionally, you can also specify:

1. A comma-separated list of Handle data types to limit your query to
2. A comma-separated list of integer indexes from the record to return in the response
3. The format in which the response should be returned
4. A mimetype to use to return the response as (this is just a convenience feature for viewing responses in a web browser)

Handle ID	(Required)
Data types	
Indices	
Format	rdf
Mimetype	

Figure 27: OpenHandle



```
- <!--
   OpenHandle (RDF/XML) - see http://code.google.com/p/openhandle/
  -->
- <r:RDF>
  - <h:Handle r:about="hdl:10677/7_Bridges">
    - <h:handleStatus>
        <h:code>1</h:code>
        <h:message>SUCCESS</h:message>
      </h:handleStatus>
    - <h:handleValues r:parseType="Collection">
      - <h:HandleValue>
          <h:index>100</h:index>
          <h:type>HS_ADMIN</h:type>
        - <h:data>
          - <h:HS_ADMIN>
              <h:adminRef r:resource="hdl:10677/7_Bridges?index=100"/>
              <h:adminPermission>111111111111</h:adminPermission>
            </h:HS_ADMIN>
          </h:data>
          <h:permission>1110</h:permission>
          <h:ttl>+86400</h:ttl>
          <h:timestamp>Wed Jun 30 01:36:15 CEST 2010</h:timestamp>
          <h:reference r:parseType="Collection"> </h:reference>
        </h:HandleValue>
      - <h:HandleValue>
```

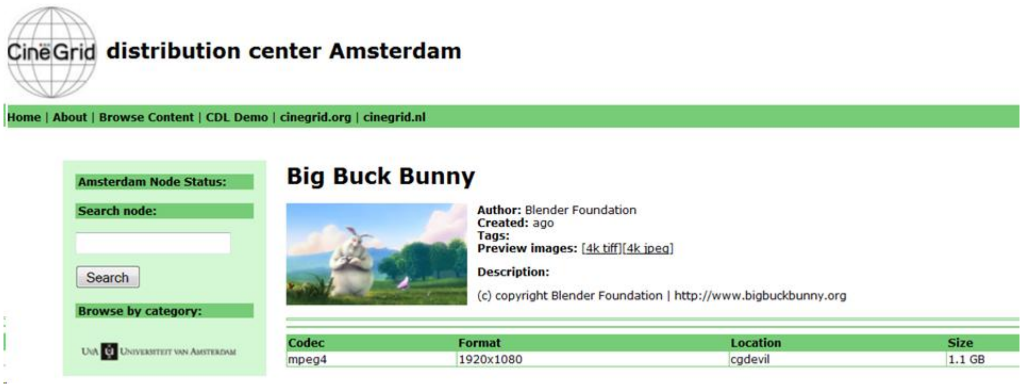
```
# OpenHandle (RDF/N3) - see http://code.google.com/p/openhandle/

@prefix h: <http://hdl.handle.net/10100/handle.rdfs#> .
@prefix r: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<hdl:10677/7_Bridges> a h:Handle ;
    h:handleStatus [
        h:code "1" ;
        h:message "SUCCESS" ;
    ] ;
    h:handleValues  (
        [
            h:index "100" ;
            h:type "HS_ADMIN" ;
          h:data [
                a h:HS_ADMIN ;
                h:adminRef <hdl:10677/7_Bridges?index=100> ;
                h:adminPermission "111111111111" ;
            ] ;
            h:permission "1110" ;
            h:ttl "+86400" ;
            h:timestamp "Wed Jun 30 01:36:15 CEST 2010" ;
            h:reference ()
        ]
        [
            h:index "1" ;
            h:type "TITLE" ;
            h:data "7 Bridges" ;
```

```
{
    "comment" : "OpenHandle (JSON) - see
http://code.google.com/p/openhandle/" ,
    "handle" : "hdl:10677/7_Bridges" ,
    "handleStatus" : {
        "code" : "1" ,
        "message" : "SUCCESS"
    } ,
    "handleValues" : [
        {
            "index" : "100" ,
            "type" : "HS_ADMIN" ,
        "data" : {
                "adminRef" : "hdl:10677/7_Bridges?index=100" ,
                "adminPermission" : "111111111111"
            } ,
            "permission" : "1110" ,
            "ttl" : "+86400" ,
            "timestamp" : "Wed Jun 30 01:36:15 CEST 2010" ,
            "reference"  : []
        } ,
        {
            "index" : "1" ,
            "type" : "TITLE" ,
            "data" : "7 Bridges" ,
            "permission" : "1110" ,
            "ttl" : "+86400" ,
            "timestamp" : "Wed Jun 30 01:36:15 CEST 2010" ,
```

Figure 28: OpenHandle results

13 Appendix C

Figure 29 show a page from the CineGrid Amsterdam portal, which is filled from an XML/RDF schema. Figure 30 shows the template used to fill with handles, which result in figure 31 for the case of the movies Big Bunny Buck and 7 Bridges.



CineGrid distribution center Amsterdam

Home | About | Browse Content | CDL Demo | cinegrid.org | cinegrid.nl

Amsterdam Node Status:

Search node:

Search

Browse by category:

UvA Universiteit van Amsterdam

Big Buck Bunny

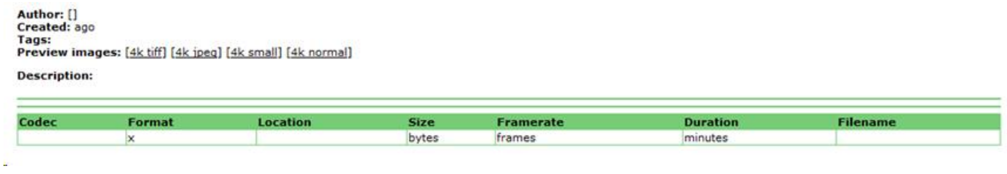
Author: Blender Foundation
Created: ago
Tags:
Preview images: [4k tiff][4k jpeg]

Description:

(c) copyright Blender Foundation | http://www.bigbuckbunny.org

Codec	Format	Location	Size
mpeg4	1920x1080	cgdevil	1.1 GB

Figure 29: Original Portal webpage

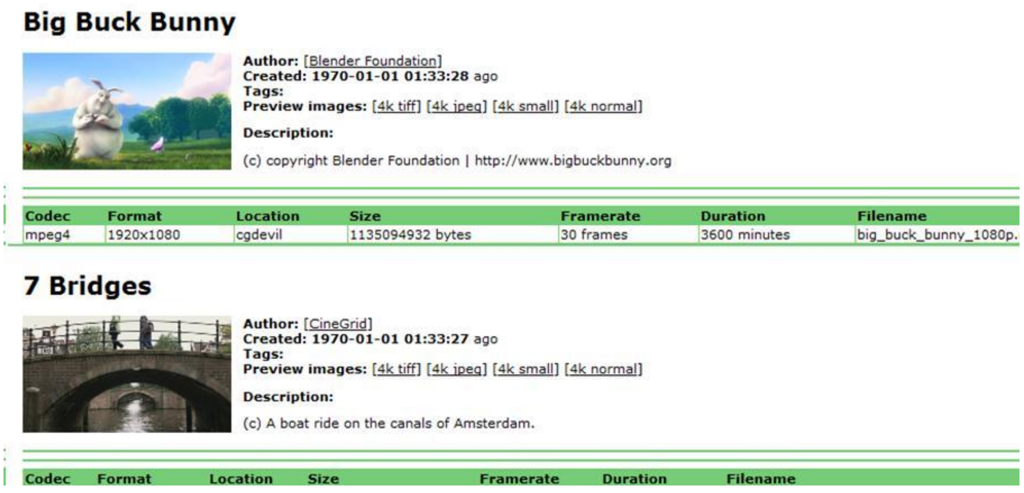


Author: []
Created: ago
Tags:
Preview images: [4k tiff] [4k jpeg] [4k small] [4k normal]

Description:

Codec	Format	Location	Size	Framerate	Duration	Filename
	x		bytes	frames	minutes	

Figure 30: Portal template



Big Buck Bunny

Author: [Blender Foundation]
Created: 1970-01-01 01:33:28 ago
Tags:
Preview images: [4k tiff] [4k jpeg] [4k small] [4k normal]

Description:

(c) copyright Blender Foundation | http://www.bigbuckbunny.org

Codec	Format	Location	Size	Framerate	Duration	Filename
mpeg4	1920x1080	cgdevil	1135094932 bytes	30 frames	3600 minutes	big_buck_bunny_1080p.

7 Bridges

Author: [CineGrid]
Created: 1970-01-01 01:33:27 ago
Tags:
Preview images: [4k tiff] [4k jpeg] [4k small] [4k normal]

Description:

(c) A boat ride on the canals of Amsterdam.

Codec	Format	Location	Size	Framerate	Duration	Filename

Figure 31: Portal template filled with handles

References

[1] The CineGrid community web site Accessed June 2010 www.CineGrid.org

[2] The AMPAS web site Accessed June 2010 http://www.oscars.org/

[3] ”Handle System Overview”. Internet Engineering Task Force (IETF) Request for Comments (RFC) Sun, S., Lannom, L. and Boesch, B. RFC 3650, November 2003 http://www.ietf.org/rfc/rfc3650.txt

[4] ”Handle System Namespace and Service Definition”, Internet Engineering Task Force (IETF) Request For Comments Sun, S., Reilly, S. and Lannom, L. (RFC) 3651, November 2003 http://www.ietf.org/rfc/rfc3651.txt

[5] ”Handle System Protocol (ver 2.1) Specification”. Internet Engineering Task Force (IETF) Request for Comments (RFC), Sun, S., Reilly, S., Lannom, L. and Petrone, J. RFC 3652, November 2003. http://www.ietf.org/rfc/rfc3652.txt

[6] ”A Framework for Distributed Digital Object Services”. Kahn, R. and Wilensky, R. International Journal on Digital Libraries, Springer, Volume 6, Number 2, April 2006 (First published by the authors in May 1995.)

[7] The Handle System (CNRI) Homepage Accessed June 2010 http://www.handle.net/

[8] The DOI system Homepage Accessed June 2010 http://www.doi.org/

[9] Paskin, Norman. ”Naming and Meaning of Digital Objects ” (pre-print). Paper reviewed and accepted for publication as part of 2nd International Conference on Automated Production of Cross Media Content for Multi-channel Distribution, Leeds, UK, December 2006. http://www.doi.org/topics/060927AXMEDIS2006DOI.pdf

[10] Paskin, Norman. ”Naming and Meaning: key to the management of intellectual property in digital media” (pre-print). Paper reviewed and accepted for publication as part of The Europe-China Conference on Intellectual Property in Digital Media (IPDM06), Shanghai, October 2006. http://www.doi.org/topics/060922IPDM_China_Paskin_preprint.pdf

[11] Paskin, Norman: ’Digital Object Identifier (DOI) System’, in: Encyclopedia of Library and Information Sciences, 3rd Edition (2008). http://www.doi.org/overview/DOI-ELIS-Paskin.pdf

[12] List of some Internet naming services from the early 2000’s Accessed June 2010 http://www.linktionary.com/n/name_services.html

[13] Laurence Lannom, CNRI. Video of presentation 11 June 2009. http://www.oscars.org/science-technology/council/projects/metadata-symposium/media/dmpms_09_lannom.pdf

[14] The *indecs* metadata framework Version 2.0, June 2000 http://www.doi.org/topics/indecs/indecs_framework_2000.pdf

[15] A Universally Unique IDentifier (UUID) URN Namespace P. Leach, M. Mealling, R. Salz RFC 4122 July 2005 http://www.ietf.org/rfc/rfc4122.txt

[16] Collection-Based Persistent Digital Archives - Part 1 Reagan Moore, Chaitan Baru, Arcot Rajasekar, Bertram Ludaescher, Richard Marciano, Michael Wan, Wayne Schroeder and Amarnath Gupta D-Lib Magazine March 2000 http://www.dlib.org/dlib/march00/moore/03moore-pt1.html

[17] PILIN Project: Project Closure Report PILIN Team Date: 20/12/2007 http://www.pilin.net.au/Closure_Report.pdf

[18] Internationalization of the Handle System, A Persistent Global Name Service Sam X. Sun Corporation for National Research Initiatives Reston, VA USA Tokyo, April 1998 http://www.cnri.reston.va.us/unicode-paper.ps

[19] The Directory: Overview of Concepts, Models and Services ITU-T Rec. X.500 1993.

[20] Understanding X.500 - The Directory D. W. Chadwick Chapman & Hall ISBN: 0-412-43020-7.

[21] ”Lightweight Directory Access Protocol (v3)” Wahl, M., Howes, T. and S. Kille RFC 2251, December 1997. http://www.ietf.org/rfc/rfc2251.txt

[22] N. Freed & N. Borenstein, ”Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies,” RFC 2045, November 1996. http://www.ietf.org/rfc/rfc2045.txt

[23] D. Goodman, C. Robbins, ”Understanding LDAP & X.500”, August 1997.

[24] UTF-8, A Transform Format for Unicode and ISO10646 Yergeau, Francois, RFC2044, October 1996. http://www.ietf.org/rfc/rfc2044.txt

[25] Distributed Systems: Principles and Paradigms, 2/E Andrew S. Tanenbaum, Maarten Van Steen Prentice Hall 2007 ISBN-13: 9780132392273

[26] Preservation management of digital materials: Handbook. Technical report M. Jones and N. Beagrie. York, UK, 2002.

[27] Digital Preservation Europe Homepage Accessed June 2010 `http://www.digitalpreservationeurope.eu`

[28] Information Identifiers Norman Paskin Learned Publishing (1997) 10, 135,156

[29] Metadata: an overview Paper given by Dr. Warwick Cathro, Assistant Director-General, Services to Libraries Division at the Standards Australia Seminar, "Matching Discovery and Recovery" August 1997 `http://www.nla.gov.au/nla/staffpaper/cathro3.html`

[30] Types of Metadata Date Created: 24 July 2006 Last Modified: 15 August 2006 14:31:28 14:31:28 Authorised By: Martine Booth, Information Planning and Architecture, Information Services Maintainer: Ebe Kartus, Information Management Accessed June 2010 `http://www.infodiv.unimelb.edu.au/metadata/add_info.html`

[31] ISO Homepage Accessed June 2010 `http://www.iso.org/`

[32] Understanding Metadata National Information Standards Organization (NISO) (2004) NISO Press, Bethesda, USA. `http://www.niso.org/standards/resources/UnderstandingMetadata.pdf`

[33] Information and documentation, International standard book number (ISBN)] ISO1: NISO/ANSI/ISO 2108:2005

[34] ISO FAQs Accessed June 2010 `http://www.lac-bac.gc.ca/isn/041011-1020-e.html`

[35] An Introduction to Metadata Paper written by Chris Taylor Manager, Information Access Service University of Queensland Library 29 July 2003 `http://www.library.uq.edu.au/iad/ctmeta4.html`

[36] IRODS Homepage Accessed June 2010 `https://www.irods.org/index.php/IRODS:Data_Grids,_Digital_Libraries,_Persistent_Archives,_and_Real-time_Data_Systems`

[37] Digital Object Architecture project Homepage Accessed June 2010 `http://www.cnri.reston.va.us/doa.html`

[38] Uniform Resource Identifiers (URI): Generic Syntax T. Berners-Lee, R. Fielding, U.C. Irvine, L. Masinter RFC 2396 August 1998 `http://www.ietf.org/rfc/2396`

[39] Uniform Resource Names (URN) Namespaces Official IANA Registry of URN Namespaces 2008-12-09 `http://www.iana.org/assignments/urn-namespaces/urn-namespaces.xml`

[40] Handle.net Factsheet, Updated 10 May 2010 `http://www.handle.net/factsheet.html`

[41] Domain names concepts and facilities P. Mockapetris RFC1034 November 1987 `www.faqs.org/rfc/rfc1034.html`

[42] Domain names implementations and specifications P. Mockapetris RFC1035 November 1987 `www.faqs.org.rfc/rfc1035.html`

[43] Domain Report VeriSign, Inc. VOLUME 7 ISSUE 2 JUNE 2010 `http://www.verisign.com/domain-name-services/domain-information-center/industry-brief/index.html`

[44] Online Registries: The DNS and Beyond... Esther Dyson 16 September 2003 Edventure.com

[45] ISAN Homepage Accessed June 2010 `http://www.isan.org`

[46] ISAN FAQS Accessed June 2010 `http://www.isan.org/portal/page?_pageid=166,41960&_dad=portal&_schema=PORTAL#FAQ1`

[47] ISBN List of Services & Fees Accessed June 2010 `http://www.isbn.org/standards/home/isbn/us/isbn-fees.asp`

[48] The OpenURL Framework for Context-Sensitive Services. NISO. (2005). Z39.88-2004, Accessed June 2010 `http://www.niso.org/standards/standard_detail.cfm?std_id=783`

[49] Open Linking in the Scholarly Information Environment Using the OpenURL Framework. Van de Sompel, H. and Beit-Arie, O. (2001). D-Lib Magazine, 7(3). ¡doi:10.1045/march2001-vandesompel¿]

[50] OpenURL OCLC web page page content 11 August 2009, prototype 26 November 03. Accessed June 2010 `http://www.oclc.org/research/activities/openurl/default.htm`

[51] OpenURL CalTech Archive Accessed June 2010 `http://library.caltech.edu/openurl/`

[52] OpenURL DLib web page May 2006 `http://www.dlib.org/dlib/may06/apps/05apps.html#6`

[53] PURL Homepage Accessed June 2010 `http://www.purl.org/`

[54] Competitive Evaluation of PURLs Larry Stone, March 22 2000 http://web.mit.edu/handle/www/purl-eval.html

[55] Collective Access Homepage Accessed June 2010 http://www.collectiveaccess.org/

[56] Whirl-i-Gig Homepage Accessed June 2010 http://www.whirl-i-gig.com/

[57] DOI Handbook International DOI Foundation 2006 http://www.doi.org/handbook_2000/intro.html

[58] Proxy Servlet for Handle System Updated 2 November 2006 http://www.handle.net/proxy_servlet.html

[59] Java API for Handle System Accessed June 2010 http://www.handle.net/hs-source/api_javadoc/index.html

[60] Distribution download page of Handle.net Updated 29 April 2009 http://www.handle.net/download.html

[61] Start page of Handle.net Updated 29 April 2009 http://www.handle.net/start.html

[62] Technical Manual (Version 2) of HANDLE.NET (version 6.2) Corporation for National Research Initiatives March 2007 http://www.handle.net/tech_manual.html

[63] Java SE from SUN.com Accessed June 2010 http://java.sun.com/j2se/

[64] Handle System Administrator mailing address hdladmin@cnri.reston.va.us

[65] Handle.net registration page Updated 23 September 2008 http://www.handle.net/registration_agreement.html?x=253&y=22

[66] Open Handle Google Code Homepage Accessed June 2010 http://code.google.com/p/openhandle/

[67] Open Handle Getting Started Page Accessed June 2010 http://code.google.com/p/openhandle/wiki/GettingStarted

[68] Open Handle CodeExamples Accessed June 2010 http://code.google.com/p/openhandle/wiki/OpenHandleCodeExamples

[69] OpenHandle Google code checkout page Accessed June 2010 http://code.google.com/p/openhandle/source/checkout

[70] Maven 2.2.1 Download page Accessed June 2010 http://maven.apache.org/download.html

[71] Handle.net client libraries Updated 10 August 2009 http://www.handle.net/client_download.html

[72] Portal to CineGrid Amsterdam Accessed June 2010 http://CineGrid.uvalight.nl/portal/

[73] DOA repository 19 January 2010 http://dorepository.org/

[74] DOA registry 10 May 2010 http://doregistry.org/

[75] Handle.net FAQs Updated 1 March 2010 http://handle.net/faq.html

[76] Handle Value Types Updated 17 September 2009 http://www.handle.net/overviews/types.html

[77] MySQL Homepage www.mysql.com

[78] ORACLE Homepage www.oracle.com

[79] PostgreSQL Homepage www.postgresql.org

[80] Question about MESSAGE_FORMAT_ERROR on pubkey authentication at Handle-info August 2007 http://www.handle.net/mail-archive/handle-info/msg00192.html

[81] Generic Cataloging document version 2 about Collective Access Accessed June 2010 http://www.collectiveaccess.org/docs/Cataloging_generic_v2.pdf

[82] Collective Access Tour 2010 Accessed June 2010 http://www.collectiveaccess.org/tour

[83] Secure Path Announces Increase of ISAN Registrations Askwebhosting.com Friday June 4, 2010 http://www.askwebhosting.com/story/24532/Secure_Path_Announces_Increase_of_ISAN_Registrations.html

[84] iRODS V1.0 updated 28 January 2008 https://www.irods.org/index.php/iRODS_V1.0

[85] Storage Resource Broker (SRB) updated 16 May 2006 http://www.sdsc.edu/srb/index.php/What_is_the_SRB

[86] Open Archives Initiative Homepage Accessed June 2010 http://www.openarchives.org

[87] Registry of Open Access Repositories (ROAR) Tim Brody University of Southampton, UK. Accessed June 2010 http://roar.eprints.org/?action=browse#version

[88] Open Access and Institutional Repositories with EPrints Accessed June 2010 http://www.eprints.org

[89] d-Libra Homepage Accessed June 2010 http://dlibra.psnc.pl/index.php?lang=en

[90] Citeseerx Homepage Accessed June 2010 http://citeseerx.ist.psu.edu/about/site

[91] Greenstone Homepage Accessed June 2010 http://www.greenstone.org/

[92] DSpace Homepage Accessed June 2010 http://www.dspace.org/

[93] Fedora Commons Homepage Accessed June 2010 http://www.fedora-commons.org

[94] An evaluative study on the open source digital library softwares for institutional repository: Special reference to Dspace and greenstone digital library Goutam Biswas and Dibyendu Paul Accepted 5 November, 2009 http://www.academicjournals.org/ijlis/PDF/pdf2010/Feb/Biswas%20and%20Paul.pdf