Towards A Unified OAI-PMH Registry

Samuel Goebert (1,2), Bettina Harriehausen-Mühlbauer (1), Steven Furnell (2)
(1) University of Applied Sciences Darmstadt, Germany, (2) Plymouth University, UK

Abstract

The Open Archives Protocol for Metadata Harvesting (OAI-PMH) has been widely adopted as an approach to allow harvesting of metadata from archives. Automated discovery of Providers is not part of the protocol. Service Providers have the additional burden of searching the web for new Data Providers. This leads to duplicate effort since every Service Provider maintains a private collection of Data Providers.

This paper proposes a decentralized registry. It is open for external contributions by design and has no single point of failure. All participants together build up a single global collection of Data Providers. New Data Providers only have to register with a single member and the entry is distributed to all participating Service Providers. Building a single collection with distributed discovery allows Service Providers to refocus on the value added service and eases the spreading of data for Data Providers.

Introduction

With the introduction of the Open Archives Protocol for Metadata Harvesting (OAI-PMH) [1] automated harvesting of metadata from archives became possible. This allowed so called service providers to build search engines around the data [5]. A single query allows users to search the catalog of different archives at once. The catalogs can be synchronized efficiently with the updates that happen at source in the catalog once the catalog is found by a service provider.

OAI-PMH is a distributed protocol at the core that allows every service provider to connect with every data provider. A global access point for registration is not defined by the protocol. With many service providers this leads to a situation where a new data provider has to register its archive with every service provider in order to distribute his data. On the other side, service providers want to find as many data providers as possible since they want to provide a search over a complete body of knowledge.

The OAI hosts a centralized registry under their domain. With a prevalidation of a submitted URL only genuine archives are allowed into the registry. The registration process is an optional step and there is now way for service providers to automatically query the data via a defined interface. Since service providers want to spread their data as far as possible, they began to register with service providers directly and the registry under the OAI domain became one of many. Every service provider maintains their own list of archives and no formalized structure for exchange of this information exists as highlighted by [10].

This paper details a novel approach to building a collection of information about OAI data providers. The approach does not need a centralized server to exchange data and is based on Archive Networks as defined by Goebert et al. in [3]. Members connect into a distributed network of participants and exchange information with other members through a defined interface as it becomes available.

Like the OAI-PMH, the network is based on a distributed protocol at the core to avoid single point of failures and become resilient against member fluctuation. Archive Networks are best suited for building a collection with many participants but not rely on a centralized infrastructure for synchronization. [3] details the following features:

- No single point of truth needed
- Auditable and temper proof data storage
- Easy to join an existing network via a meta data file
- provides a permanent infrastructure to access the data as
- long as one machine remains in the network
- Mirrors can edit local data and changes are synchronized
- with the network.

Concept

The protocol allows everybody to participate in a network but Service Provider will have a special interest in running a node that collects URLs from Data Providers as they can be seen as interface for the end user to the data. Service Providers are rather stable services with a stable number of providers who take great effort in order to be discovered.

The Service Provider runs software that implements the handling of the Archive Network Protocol (APN) independently from the existing search engine infrastructure. The protocol implementation provides a REST [11] programming interface in order to query data. The integration between these two services has to be done by the service provider.

In order to connect data providers a key has to be provided. The key for discovery in Archive Networks is a meta data file that contains information about how to find other interested nodes. Many Archive Networks exists for different collections but an Archive Network is always scoped to the usage of the same key. This allows several installations who serve different collections to coexist next to each other on one installation. In this paper we will focus only on the usage as a registry for the OAI-PMH protocol.

The key contains the URL to a tracker service [12]. The tracker service maintains information about other machines that registered themselves for the same key. The format of the key file is compatible with the already existing bittorrent infrastructure [12]. The tracker service returns IP addresses of machines who participate in the same network.

The key is brought into the ANP implementation and used to contact the tracker. After receiving information about other members of the network, the implementation starts to contact those in order to validate that they are really members of the network with the interested of building a collection.

For Service Providers the local registry provided by the ANP implementation is not different to the one found on the OAI domain. A simple HTML form provides an interested user with the ability to enter the URL to his service. The web service run by the service provider that receives the request for inclusion onto the list of known repositories, validates the URL by calling it and making sure that the protocol is honored by the service behind the URL. This avoids including a repository which is not behaving correctly defined by protocol or including a wrongfully entered URL into the list of know repositories.

So far the data provider is only stored locally. In order to share it with the other members of the network it is send to the ANP implementation. The implementation tests the URL again and after successful validation wraps the URL in a transaction. A transaction is an xml structure that provides information about the URL. Transactions have to be wrapped in an xml structure called a block in order to be send to other members. A block can contain more than on transactions and a link to the previous block to build a chain.

The block is sent to all connected machines who verify the information in the block and accept it as an advancement of the local timeline. The block is then sent on until every member of the network has received the block. The solution to a cryptographic puzzle [3] that has to be included in a block makes sure the network is not flooded by blocks and order becomes a matter of creation time. The puzzle takes some time to solve and is new for every block.

New members of the network download all blocks in order to sync with the network. To validate that the blocks have not been tempered with the key to the network also contains the first block. The second block in the network has to link to the first block in the key. Subsequent blocks have to link the previous block. This way a new member can verify the chain of blocks he received from the network back until the first block. If the chain of blocks is valid until the first block contained in the key the timeline is valid and has not been tempered with.

Periodically the Service Providers should query the APN implementation if new change sets have arrived. If yes the service validates a new URL locally and if successful adds it to the local database in order to prepare for meta data harvesting of the newly found provider.

A reference implementation can be found at: https://github.com/bigcurl/oai-registry-distributed

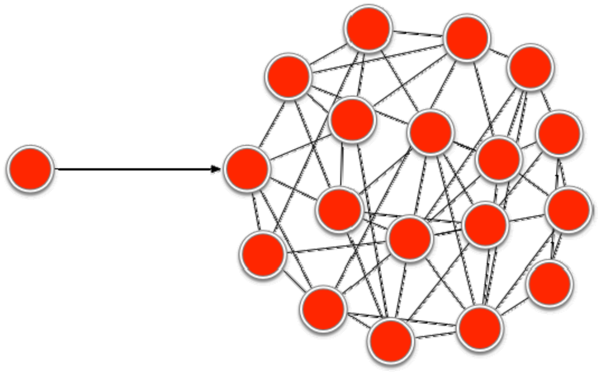

***Figure 1**. Distribution of data in an Archive Network*

Integration

This chapter describes the communication between the existing search engine infrastructure and the protocol implementation. A node running the APN implementation is totally independent from the system that wants to harvest information. The application can talk to the protocol implementation via a REST interface. The protocol itself is based on the friends' container described by [1]. The node is running a webserver internally that supports the following commands:

GET /urls
POST /urls

GET /urls

Function:
Returns the entries of all URLs in order of appearance in the friends container format specified by the OAI-PMH.

Return Status:
200: OK
400: The request could not be understood by the server due to malformed syntax
500: Internal server error

Return values if return status 200:
id: A unique alphanumeric value for the URL.
baseURL: A URL to a OAI repository.

Example:

```
<BaseURLs>
   <baseURL id='234987234...'>
      http://url.to.repository.com
  </baseURL>
</BaseURLs>
```

POST /urls

Function:
Sends a candidate URL for inclusion into the chain of blocks

Post Parameters:
baseURL: The URL to a repository

Return Status:
200: OK
400: The request could not be understood by the server due to malformed syntax
409: Conflict. Resource already exists.
500: Internal server error

baseURL=http://url.to.repository.com

Integration

The protocol format internal to the Archive Network must be tailored to situation of handling URLs. We adapt the primitives in the following way.

Transaction

A transaction wraps a single baseURL with metadata.

```
<transaction>
  <tx-hash>390d0002...</tx-hash>
  <created-at>2014−05−29T09:30:10Z</created-at>
  <baseURL>http://url.to.repository.com</baseURL>
</transaction>
```

tx-hash: is a SHA512 hash taken from the URL contained in link.
creation-at: Date when the URL was created in ISO:8601 format
baseURL: A URL to a OAI repository

Block

A block contains one or more transactions and provides the fields to validate it later.

Example:

```
<block>
  <previous−hash>0023987...</previous−hash>
  <hash>0002390d . . .</hash>
  <transactions−hash>3246234...</transactions−hash>
  <nonce>439</nonce>
  <transactions>
  <transaction>
    ...
  </transaction>
  <transaction>
    ...
  </transaction>
  </transactions>
</block>
```

previous−hash: The hash of the previous block this block wants to be linked to
hash: A hash of the current block. SHA512(previous−hash + transactions−hash + nonce)
transactions−hash: A hash taken from the content of the transactions fields.
SHA512(<transaction>...</transaction> + ...)

nonce: The solution to the cryptographic puzzle as described in [3].

Future Work

Build Robust Community

The more members the network have the better. Every member has a full copy of the data and acts as an entry point into the network.

The ability to run a node in the network is not limited to service providers. Data Providers and even end users can run a node in the system and strengthen the data set against member fluctuation. Recruiting work has to be done to encourage people to run a node themselves.

Export Format

The export format is based on the friend container description from the OAI-PHM. Other protocols exists that might be better suited for the given task. The friend container format is flat and only contains a link to a repository. Other formats can be supported in the future that contain more information about the repository itself.

Automated Registration

Existing repository implementations could be changed to automatically announce themselves to the network. This would take a burden from the Service Providers and the Data Providers equally. The Service Providers do not need to take additional steps to find the repository and Data Providers do not need to register the service initially or every time a change to the main URL happens.

Conclusion

In this paper we detailed a new approach to build a distributed collection of providers for the OAI-PMH protocol based on archiving networks. Every member of the network owns a full copy of a globally shared collection that contains access information about service providers and data providers. New entries to one of the members of the network are replicated and synchronized as a candidate record among all others. After local validation of a new candidate’s data, the candidate record is accepted into the local timeline. This way the global collection timeline is advanced.

Explicit discovery of service providers or data providers is not specified by the OAI-PMH protocol. It is up to the creativity of the providers to find new data sources or service providers. This leads to duplicate effort since every provider maintains their own collection. Sharing of this information is also hindered since there is no official specification of how to exchange provider data.

With the help of an archive network consisting of members running the protocol to exchange provider data, the effort for data- and service providers to find each other is drastically minimized. A data provider only needs to register once to target all interested parties. This eases the task of discovery for new service providers to simply join the network and validate the data.

The complete distribution of records data to every member of the network ensures that the data is preserved and accessible until the last member leaves the network. The data is resilient against fluctuation of members. Members can join or leave at any time without interrupting the flow of data for the other members or having an impact on local availability. Downtime of local infrastructure is not a burden anymore since all it takes to regain missed data is to resync the local database with the network.

The protocol allows structuring and formalizing access to a trove of data that took many years to build. Every member of the network contributing storage and bandwidth makes sure that the initiative stays alive.

References

[1] Carl Lagoze et al., "The making of the Open Archives Initiative Protocol for Metadata Harvesting", Library Hi Tech 2003, Vol. 21 Iss: 2, pp.118 - 128

[2] Thomas Habing et al., "Developing a Technical Registry of OAI Data Providers", 8th European Conference, ECDL 2004, Bath, UK, September 12-17, 2004. Proceedings

[3] Goebert et al., Decentralized Hosting And Preservation Of Open Data, Archiving 2013 Final Program and Proceedings, Washington, DC; April 2013; p. 264-269;

[4] Simeon Warner, "The OAI Data-Provider Registration and Validation Service“, 9th European Conference, ECDL 2005, Vienna, Austria, September 18-23, 2005. Proceedings

[5] F. McCown et al., "Search engine coverage of the OAI-PMH corpus," Internet Computing, IEEE , vol.10, no.2, pp.66,73, March-April 2006

[6] Ann Apps, "A Registry of collections and their services : from metadata to implementation.", 2004 . In The International Conference on Dublin Core and Metadata Applications (DC2004), Shanghai (China), 11 - 14 October 2004.

[7] Herbert Van de Sompel et al., "Using the OAI-PMH ... Differently," D-Lib Magazine, Volume 9, Number 7/82, July/August 2003

[8] Xiaoming Liu et al., "Repository synchronization in the OAI framework“, In Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries (JCDL '03). IEEE Computer Society, Washington, DC, USA, 191-198.

[9] Michael L. Nelson et al. "Efficient, automatic web resource harvesting“, In Proceedings of the 8th annual ACM international workshop on Web information and data management (WIDM '06). ACM, New York, NY, USA, 43-50. DOI=10.1145/1183550.1183560 http://doi.acm.org/10.1145/1183550.1183560

[10] Shreeves, Sarah L., Thomas G. Habing, Kat Hagedorn, and Jeffery Young. 2005. Current developments and future trends for the OAI Protocol for Metadata Harvesting. Library Trends 53, no. 4: 576-589.

[11] Roy Thomas Fielding. 2000. Architectural Styles and the Design of Network-Based Software Architectures. Ph.D. Dissertation. University of California, Irvine. AAI9980887.

[12] B. Cohen. Incentives build robustness in BitTorrent. In P2PEcon, 2003.

Author Biography

Samuel Goebert is a computer science Ph.D. student at the University of Plymouth, England and the University Of Applied Sciences Darmstadt, Germany. Goebert has over 11 years of experience in software development and associated technologies. He currently researches peer-to-peer methodologies to improve preservation of digital cultural heritage. Further details at www.distributed-preservation.org

Prof. Steven Furnell is the head of the Centre for Security, Communications & Network Research at Plymouth University (UK), and an Adjunct Professor with Edith Cowan University (Australia). His research interests include information security and Internet technologies, and he has authored over 240 refereed papers in these topics. He is a Fellow of the BCS, a Senior Member of the IEEE, and full member of the Institute of Information Security Professionals. Further details at www.plymouth.ac.uk/cscan

Bettina Harriehausen-Mühlbauer is a Computer Science professor at the University of Applied Sciences Darmstadt, Germany and holds visiting professor positions at the University of California, Berkeley, the University of Utah, Salt Lake City, and the Xi'an University of Posts & Telecommunications, Xi'an/China. Before her academic career, she worked on knowledge based systems and NLP-technology at IBM. In academia, she has continued work in the NLP field but added research in the fields of mobile learning and development of mobile applications