Bibliographic Metadata Harvesting to Support the Management of an Institutional Repository

Ricardo Miguel Loureiro da Costa
Instituto Superior Técnico
Technical University of Lisbon
Avenida Rovisco Pais, 1 — 1049-001 Lisboa
Email: ricardo.costa@ist.utl.pt

***Abstract*—This thesis approaches the problem of automatic harvest of bibliographic metadata records from indexing services to populate institutional repositories. Manual insertion of records is a tedious and error-prone task and the automation of the process intends to facilitate the management of a repository. However, processes for automated harvesting of records must deal with author's identification and duplicate records consolidation. Approaching the aforementioned processes, we introduce a system that is able to harvest bibliographic metadata from different sources, identify and consolidate the retrieved records and make them available to external parties interested in the information, such as an institutional repository.**

The proposed system was tested with real bibliographic metadata of scientific publications from a subset of faculty members at Instituto Superior Técnico. The results of the evaluation show that, despite the required time for consolidation, the merged records contain a valid and rich aggregation of the available information and can be efficiently accessed by external entities through a machine-to-machine interface.

***Keywords*—Bibliographic Metadata, Automated Harvesting, Institutional Repositories, Duplicate Consolidation**

I. Introduction

Scholarly communication has been made through a traditional model run by publishers where several stakeholders interact in order to try and disseminate the knowledge among the scientific community. That model introduces a large scattering of publications among several journals [1] making works are not easily accessible.

The introduction of institutional repositories (IR) (section II-A), whose purpose is to collect the intellectual work produced by the community members, has been contributing to concentrate it in a structured environment, providing tools for access to and evaluation of that work. However, the population of an institutional repository can be troublesome due to the several services that provide information and the problem of author name management [2].

A. Problem

This thesis proposes to approach the problem of automatic harvest of bibliographic metadata from different external services to populate an institutional repository, as well as make available through a functional interface the existent information. It also approaches the problem of finding and consolidating duplicate records in a large set of bibliographic metadata. Given the problem of name matching [3], this thesis is expected to be able to manage the identities of authors in different services.

The following research questions summarize the problem approached by this thesis:

- **Research Question 1:** Assuming that a digital library might replicate contents already existing in other digital libraries or services, how can we update the contents of our system from those external ones with the minimal human effort? In this work we addressed this challenge only at the level of metadata, leaving in open the issue of synchronizing the information objects described by the metadata.
- **Research Question 2:** Assuming that a digital library is required to share information (metadata records) with other digital libraries, what are the most relevant scenarios where that can occur nowadays, and what are the technical requirements and best reference implementations to make that possible in the most efficient and effective way?
- **Research Question 3:** Assuming that a digital library might harvest records from multiple external services, how can we identify possible duplicate records and produce a consolidated set of records available to outside parties?

B. Structure

This document is structured as follows: **Section II** presents the related work. We approach the main topics that were considered relevant for this thesis. A summary of an analysis to possible sources of metadata records is presented in **section III**. **Section IV** describes the main functionalities of the designed and implemented system. **Section V** reports the evaluation of the implemented solution in several scenarios. Finally, **Section VI** presents some conclusions and points directions for future work.

II. Related Work

This section presents a summary of the related work in the areas that were considered relevant for this thesis.

A. Institutional Repositories and Scholarly Communication

Institutional repositories (IR), a type of digital library, are intended to store, select, offer access to, preserve and ensure

persistence over time of a set of resources, in the scope of an institution's intellectual capital. Lynch [4] identifies as factors to the arise of IR the dropping costs of storage, the existence of relevant standards, and the recognition of the need for preservation. Also, initial occurrences of publicly available journal articles showed how scholarly communication could change.

With a repository, it is easier to demonstrate the scientific, social and financial value of a university. However, the implementation of an IR faces challenges that may threaten its success. Faculty collaboration is vital and that must be shown to the researchers, as authors do not receive direct compensation for their publications; they do it to achieve professional recognition and career advancement [1].

Several software packages exist:

- DSpace[1], an open-source system, serves as repository for all kinds of digital content produced by members of an university or organization. It intends to be a simple system that at the same time supports all the needs of a research organization. The information model is focused on the concept of "Communities", sub-units within an institution that can adapt the system to their own needs, and manage their own submission process [5]. DSpace uses a qualified Dublin Core metadata schema for describing items and it supports interoperability through OAI-PMH, SWORD (both as a server, receiving content, or as a client, submitting content to other repositories) and OpenSearch. According to Markey et al. [6], DSpace is the most used institutional repository system in the United States as of 2007.
- Fedora[2] does not provide a full solution for a repository but instead a set of web services for storing, managing and accessing digital content. It uses the concept of "complex object" that aggregates multiple objects of different kinds [7]. Similarly to DSpace, Fedora also supports dissemination of information via OAI-PMH.
- EPrints[3] provides a web interface for managing, publishing and searching among the repository's contents and metadata. It also supports several interoperability features, such as OAI-PMH, SWORD protocol, and through usage of plug-ins, BibTeX, Dublin Core, METS, MODS, among others.

Investigators in research areas often come to conclusions regarding their scientific work, that are then translated into the form of written publications. Viewing scientific evolution as a process based on small contributions for a bigger purpose, it is natural for some researchers to build based upon previous work of someone else. Crediting one's contribution, i.e. referencing, contributes for the enrichment of an article. Authors are, therefore, encouraged to provide a listing of all the bibliographic references used in their work. Several styles exist for references, such as Chicago[4], Harvard[5], Modern Language Association[6], Nature[7], Science[8] and many others. The used style depends on the discipline, and despite the natural differences among different publishers styles, the provided information contains descriptive metadata about the reference.

B. Identifiers

Identification is crucial when dealing with collections of resources, because it is the way to refer to a specific resource. When analysing identifiers, we can classify them according to four different axis [8]: uniqueness, resolution, interoperability and persistence.

Regarding the identification of resources, there are several identifier systems related to bibliographic metadata, such as ISBN [9], ISSN[9], URI, URL [10] and DOI [8]. These identifiers all have a specific purpose and can be classified according to the mentioned axis as presented in table I.

Identifier	Uniqueness	Resolution	Interoperability	Persistence
ISBN	YES	YES	YES	YES
ISBN	YES	YES	YES	NO
URI	YES	YES	YES	NO
URL	YES	YES	YES	NO
DOI	YES	YES	YES	YES

Table I
COMPARISON OF RESOURCE IDENTIFIERS.

Unlike as it happens when identifying a resource, where typically the title is enough to tell two resources apart, uniquely identifying an author presents different challenges, as there is no way to sure that the name really belongs to the person that wrote the publications. Several factors contribute to this uncertainty [2] [11]:

- Different people have the same name;
- Sometimes initials are mixed into the name;
- Non-Latin names are converted into the same equivalent (e.g. Zhang);
- People change name over time (e.g., marriage and divorce);
- Spelling errors;
- Affiliation and contact information gets outdated.

The proposed solution for this problem is to create an identifier for each author, and associate the publications to the correct identifier [2]. Different author identifier systems are already in place such as International Standard Name Identifier (ISNI) [10], Thomson Reuters' ResearcherID [11], Elsevier's Scopus Author Identifier [12], Open Research & Contributor ID (ORCID) [13], Digital Author Identifier (DAI) [14]. However, the large number

[1]http://www.dspace.org
[2]http://www.fedora-commons.org
[3]http://www.eprints.org
[4]http://www.chicagomanualofstyle.org/toolscitationguide.html
[5]http://libweb.anglia.ac.uk/referencing/harvard.htm
[6]http://www.mla.org/style
[7]http://www.nature.com/nature/authors/gta/
[8]http://www.sciencemag.org/site/feature/contribinfo/prep/res/refs.xhtml
[9]http://www.issn.org
[10]http://www.isni.org/
[11]http://www.researcherid.com/
[12]http://www.info.sciverse.com/scopus/scopus-in-detail/tools/authoridentifier
[13]http://about.orcid.org/
[14]http://www.surf.nl/en/themas/openonderzoek/infrastructuur/pages/digitalauthoridentifierdai.aspx

of existent system becomes itself a problem, as the usefulness of such as system comes from it being universal [11]. Otherwise, the problem persists, because there's no way to be sure that author A in system 1 is the same author B in system 2.

C. Metadata

Metadata is defined as data about data or information about information. On a more formal level, *"metadata is structured information that describes, explains, locates or otherwise makes it easier to retrieve, use or manage an information resource"* [12]. It can be seen from different perspectives and therefore it is commonly divided into three main types [12]:

- **Descriptive Metadata** refers to information about the resource itself.
- **Structural Metadata** tells how resources might be related among themselves, possibly as parts of another resource in a higher level scope.
- **Administrative Metadata** is related to the intrinsic properties of the resource.

Metadata usage has been growing in line with the growth of digital content [13], and it has been applied in several areas with a growing importance. Those areas include resource discovery, resource organization, and interoperability [14]. Table II presents examples of different metadata schema and groups them according to their type.

Descriptive Metadata	Structural Metadata
MARC21/UNIMARC	METS
MODS	
BibTeX	
DCMES	
ESE	

Table II
MAPPING BETWEEN METADATA SCHEMAS AND TYPES

D. Functional Interfaces

1) Z39.50: Initially defined as the "ANSI/NISO Z39.50" [15] United States national norm and later as the international norm "ISO 23950"[16], Z39.50 is a client-server protocol which specifies data structures and interchange rules that allow information retrieval over a database of records. Since different databases may have different ways of describing the contained information, a common model is necessary for that description to which each implementation should be mapped [15].

2) OAI-PMH: The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is a protocol for harvesting records stored in repositories. Components that participate in this framework are classified as *Data Providers*, that expose metadata, and *Service Providers*, those who use the harvested metadata. Each item within a repository has a unique identifier that must follow the URI syntax. Identifiers are used when listing records present in the repository or the identifiers themselves, or when a request for a record in a specific metadata format from an item issued. Figure II-D2 shows an informal overview of an environment of services based on OAI-PMH. Records present in a repository may follow multiple metadata schemas; however, all repositories must implement Dublin Core format for purposes of interoperability [16].

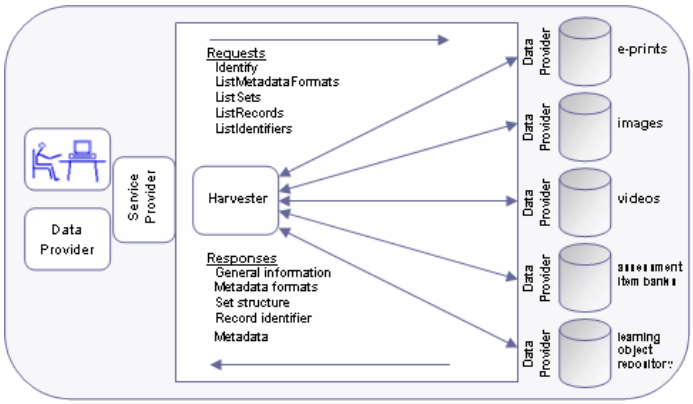

Figure 1. An informal overview of an environment of services based on OAI-PMH

OAI-PMH relies upon the Representational State Transfer (REST) architectural style introduced by Fielding in his PhD Thesis [19]. This style intends to transfer a representation of a resource, typically in XML or JSON, that exists within a server and has a state. REST is not a standard, but it does rely on standards such as HTTP, URL, XML, among others. In order for a service to be considered RESTful it must meet the following constraints [19] [20]: Client-Server, Stateless, Cache, Uniform Interface, Layered System and Code on demand (optional).

3) SWORD: Simple Web-service Offering Repository Deposit[17] (SWORD) is a protocol for depositing content from one location to another and a profile of the Atom Publishing Protocol. SWORD aims at depositing any kind of files into a remote repository in a standardized way. It supports querying a repository for information about collections available for deposit and inserting an object into the repository [17]. Different deposit interfaces have been developed for EPrints, Fedora and DSpace repositories [18].

E. Duplicate Detection

When consolidating information, often occurs that information ends up being duplicated. In that way, it is necessary to eliminate the duplicate information. Since most information is represented as strings, string similarity algorithms are used for duplicate detection, such as Edit Distance, Bag Distance, Smith-Waterman, Longest Common Sub-string, q-grams, Jaro or Winkler [3]. A common problem related to bibliographic references is to find whether two different strings represent the same real name of a person, e.g., "John Smith" and "Smith, J.".

[15]http://www.loc.gov/z3950/

[16]http://www.iso.org/iso/catalogue_detail.htm?csnumber=27446

[17]http://swordapp.org/

In addition to the already pointed pattern matching algorithms, others exist that take into account the sound of the words. Those are known as phonetic algorithms and some examples are: Soundex, Phonex, Phonix and Double-Metaphone [3].

III. Information Sources

We define an **information source** as an external service that registers an identifier for an author and, optionally, offers the possibility of harvesting the author's publications metadata records. From this definition we assume that a valid information source in the scope of this thesis always has, at least, one publicly available identifier for each author. Therefore, and given the problem of author identification discussed in section II-B, we exclude services that provide publications listings only through queries with the name of the author or some other criteria.

An analysis was conducted prior to the system implementation in order to assess which existent services could play the role of information source as previously described. That analysis was based on the following criteria:

- Existence of an unique identifier for each author;
- Covered domain;
- Access to the system information;
- Data format of retrieved information;
- Required response document manipulation.

The analysed services are indicated in table III.

Information Source	URL
Google Scholar	http://scholar.google.com/
Microsoft Academic Search	http://academic.research.microsoft.com/
ACM Digital Library	http://dl.acm.org/
DBLP	http://dblp.uni-trier.de/
IEEE Xplore	http://ieeexplore.ieee.org
ResearcherID	http://www.researcherid.com/
ScienceDirect	http://www.sciencedirect.com/
Nature	http://www.nature.com/nature/index.html
CiteSeer	http://citeseer.ist.psu.edu/index
Mendeley	http://www.mendeley.com/

Table III
Homepage URLs of the analysed services.

From table IV, that summarizes the results of the analysis, it is possible to see that of all analysed services that may act as information sources, 7 of them provide a unique identifier for each Author. As this a mandatory requirement for an information source, services such as IEEE and ScienceDirect are excluded.

For the other ones, and after a deeper analysis, it is possible to conclude that the most suitable services to act as information sources are: Google Scholar, Microsoft Academic Search, ACM Digital Library and DBLP. Despite meeting all requirements, ResearcherID requires a high manipulation of the retrieved document, as well as user interaction simulation to be allow harvesting of all records of an author. Because of that, it was not included in the set of information sources considered for this work. Mendeley provides an API for retrieving information but requires authentication for each author to harvest his publications. Both ResearcherID and Mendeley will be approached in section VI-A dedicated to future work.

Information Source	Unique Identifier	Domain	Access	Data Format	Transformation
Google Scholar	YES	Multiple	Single HTTP GET	BibTeX	None
Microsoft Academic Search	YES	Multiple	Single HTTP GET	BibTeX	Minimal
ACM Digital Library	YES	Single	Single HTTP GET	BibTeX	Minimal
DBLP	YES	Single	Multiple HTTP GET	XML	Major
IEEE Xplore	NO	Single	-	-	-
ResearcherID	YES	Multiple	Multiple HTTP GET	HTML	High
ScienceDirect	NO	Multiple	-	-	-
Nature	NO	Multiple	-	-	-
CiteSeer	YES	Multiple	Multiple HTTP GET	HTML	High
Mendeley	YES	Multiple	API	JSON	High

Table IV
Comparison of different potential information sources.

IV. Proposed Solution

This section presents some of the details regarding the implemented solution. It starts by introducing the domain entities that compose the core of the system as well as architectural view of it. It then presents the main functionalities implemented into the system.

A. Use Cases

Taking into account the problem this work proposes to approach, stated in section I-A, it is possible to define a number of expected use cases for the system:

- **Register unit:** Register a unit in the system so that it is possible to associate an author to it.
  Actors: System administrator; librarian.
- **Register information source:** Register an information source in the system, being importable or not, so that it is possible to associate an author to it through an identifier.
  Actor: System Administrator.
- **Register author:** Register an author in the system by providing, at least, the authoritative name and the unit to which the author belongs to.
  Actors: System administrator; librarian.
- **Set aliases of author:** Set the aliases of an author so that it is possible to find more information about that author.
  Actors: System administrator; librarian.

- **Register identifier:** Associate an identifier to a pair (author, information source), allowing the importation of records of such author from the given information source. **Actors:** System administrator; librarian.
- **Get list of records:** Get a list of records according to some criteria such as the format, or information source. **Actors:** Librarian, client.
- **Configure system:** Configure the different options that affect the behaviour of the system. **Actors:** System administrator; librarian.
- **Order records harvest:** Order a harvest process. **Actors:** Librarian, client.
- **Import records:** Import metadata records. **Actor:** Bibliographic source.

B. Domain

The domain of the system is based upon five different core entities:

- **Information Source:** The information source entity represents an external service that registers an identifier for an author and, optionally, offers the possibility of harvesting the author's publications metadata records. It also contains the required information that allows a fully automated mechanism of records harvesting.
- **Identifier:** The identifier entity represents the unique identifier each information source has for an author, establishing a connection between the two entities. It holds the identifier that is later used when harvesting records. Also, the identifier is necessary in order to provide a link to the profile page of an author in the information source system.
- **Record:** The record entity represents a bibliographic metadata record. It contains the information regarding a harvested record and keeps an association to the author of the described publication and the information source from where it was retrieved.
- **Author:** The author entity represents any faculty member or researcher that has authored scientific publications. It keeps an association to the Identifier entity and an association to the Publication entity. Besides those associations, an author entity also has a one-to-one association with the unit entity. This means that an author always belongs to one, and only one, unit.
- **Unit:** The unit entity represents an organizational unit in the scope of the organization where the system is installed. As mentioned before, a unit aggregates authors in a one-to-one relationship.

C. Architecture

The implemented solution is composed of 5 different modules, as it can be seen in figure 2:

- **Data Access:** Responsible for accessing the data storage system to perform read and write operations regarding domain entities. It is also responsible for making elaborated queries to retrieve collections of domain entities.
- **Duplicate Detection:** Responsible for the process of finding and consolidating into one the records that are duplicate descriptions of the same publication.
- **Import:** The import module is responsible for the harvesting of metadata records from the different information sources.
- **User Interface:** The user interface module implements the GUI (Graphical User Interface) for the human to machine interaction with system.
- **REST Services:** The REST Services module is responsible for the implementation of the machine-to-machine interface of the system.

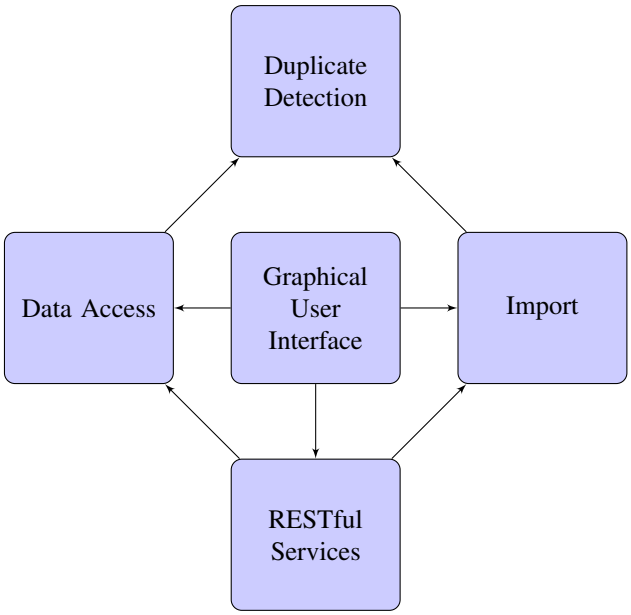

Figure 2. Dependency view of the architecture of the system. Each arrow establishes a "uses" relationship from the start to the end of the arrow. For instance, the RESTful Services module uses the Data Access module.

D. Harvest

The main functionality required for the system is the automated harvest of descriptive metadata records from external services. In order to achieve that, two things are necessary: 1) The format of the URL from where to retrieve the document with the metadata records; 2) The identifier of the author in the desired information source.

After acquiring that information, the harvesting processing is as follows:

- Fetch the document using the specific URL for the author's publications;
- Transform the document into a BibTeX format;
- Parse the BibTeX document;
- For each parsed record, check if it is new or a modified version of an already existent record and save it.

The retrieved documents from each information source are not homogeneous. All of them require some transformation in order to be parsed by the BibTeX library. But even with document transformations, sometimes it is not possible to parse a metadata record. The system follows a best effort

approach: it parses one record at a time, discarding just the ones it cannot parse.

After a successful parsing of a record, the system tries to save it. Two things can happen: 1) it is a new metadata record harvested from the given information source; 2) it is a metadata record that was harvested in a previous import process for the same information source. In order to make this distinction the system tries to find a duplicate record from the same author and information source. If it does not exist, it is considered a new record and saved into the database. On the other hand, if the system finds an existent record matching the criteria, then it is considered another version of an existent one. In this situation, the system checks if there is any modification comparing to the existent record, and if it exists, replaces the record.

The previous description of the import process shows that it is possible to automatically harvest metadata records. However, such process requires action from the user to start and research question 1 (see section I-A) sets as an objective the minimal human effort in the harvesting process. This goal is achieved by fully automatizing the import the following way: when an information source is added to the system, a periodic timer associated to that source is activated. When the defined period ends, the harvesting process is automatically initiated.

Giving the possible constraints imposed by external services (for instance, many requests over a small period of time may lead to blockage of requests), it becomes necessary to parametrize the frequency of the harvesting processes. The used parameters are part of the information source entity (section IV-B) and define the date/time of first import, how long it will wait until the next import and the maximum number of authors whose records will be harvested. This approach allows for the system administrator to define the parameters that better suit each information source. The authors whose records will be imported are chosen based on the date of the last import: the authors with the oldest records in the system.

E. Duplicate Detection

In order to identify which records represent the same publication, a duplicate detection mechanism was developed. It has as input two records and returns true if they are considered duplicate, and false otherwise. The records are identified as duplicates when both of them: 1) are of the same type (e.g., article); 2) have the same amount of fields; 3) have the same fields with the same value.

The fields to compare between records are determined by a configuration file (see section IV-F). If a record does not contain a field specified as duplicate criteria, it is not possible to draw conclusions regarding the duplicate status, because the missing field could be the one establishing the difference. Summarizing, two records are considered duplicates if and only if they both have the same value for all the specified fields. The duplicate detection process is invoked in two different situations: 1) When a new record is being added to the system, in order to know if that record is another version of an existing one; 2) At the end of each harvesting process, when the record consolidation mechanism occurs.

With the duplicate detection algorithm is possible to identify different records for the same publication. Nevertheless, the goal is to generate a **consolidated** list of records. To achieve that, the system goes through all existent records and puts them in a list. However, the algorithm first checks if there is duplicate record already in the list; if so, it merges both records. This way, a duplicate-free list of enriched records is generated.

Analysing the complexity of the algorithm, we can see that it has a complexity of $\mathcal{O}(N \times M)$ where N is the total number of records in the system and M is the number of consolidated records.

With a large number of records in the system, the consolidation process is expected to take longer. Generating the consolidated list every time it is requested would render the system unusable due to long response times. This problem is solved by generating cache files with the consolidated records lists. This cache is refreshed after each harvest process and when a unit, an author or an information source is deleted. The choice for these moments follows this rationale: the records in the system can only change when an harvesting process occurs or a deletion is performed. In between those moments, the records are the same and do not change. Relying on that fact, there is the guarantee that consolidated lists generated between harvests will always contain the same results. Therefore, the system generates cache files and, when a client requests a consolidated list, it reads the cache to generate an answer. With this approach, the system is expected to be much faster when answering requests for consolidated data.

F. Configuration

The implemented solution is designed to run continuously. However, at setup time, the system is empty and the task of inserting domain entities may become tedious and lead to errors. To avoid that situation, the system has two configuration files from which it reads the necessary data for loading the defined information sources and organizational units.

In order to provide flexibility to the system, there are other two configuration files. The first one defines which fields in the metadata records will be used as criteria for duplicate detection. The other configuration file defines the fields that must be present in a record for it to be considered BibTeX compliant. The file follows a specific format to specify the entry types that are supported, as well as the fields that are required even in situation when only one of two fields are required.

G. Functional Interface

In order to facilitate the access to the stored information, the system provides a machine-to-machine functional interface using RESTful services (section **??**). They are organized in an hierarchical structure of endpoints, with clean URLs that provide a meaning for other developers that build systems that interact with the defined services.

The implemented services can be divided into three categories:

1) Access services, corresponding to HTTP GET requests, that intend to retrieve information contained in the system.
2) Addition services, corresponding to HTTP POST requests, that intend to add information to the system. The data regarding the added entity is passed through the body of the request that is validated before any change is made.
3) Deletion services, dedicated to delete some entity in the system. These services correspond to HTTP DELETE requests and permanently delete the designated entity, which may involve cascading deletions.

V. Validation

The validation section of this document aims at presenting the main results of the evaluation of the system. It starts by presenting the setup used for the executed tests, and then presents the evaluated scenario and respective results. It finishes with a small discussion of the results.

A. Validation Setup

All tests were executed on a localhost server and therefore network latency times are minimal. The presented execution times are recorded within the system so that they do not depend on external factors. The machine on which the tests were run has the following specification:

- Intel Core i7 - 2670QM @2.2GHz CPU
- 8GB of RAM memory
- 64bit operating system

All tests were performed using Postman - REST Client[18], a plug-in for Google Chrome browser that allows interaction using RESTful services.

Since the system is expected to interact with external parties using the machine-to-machine technologies, the graphical user interface, expected to be used mostly by librarians and system administrators, was not subjected to evaluation. The evaluated scenarios use, when appropriate, a subset of authors from IST, consisting of 10 randomly chosen faculty members of Departamento de Engenharia Informática.

B. Results

1) Add information sources: Figure 3 shows the results of adding information sources to the system. The x-axis represents the number of information sources inserted using the same HTTP request, and the y-axis the time (in milliseconds) required to add all specified information sources. A linear tendency is observed when the number of instances increases but the average time to insert an instance is approximately constant.

2) Add units: Figure 4 shows the results of adding units to the system. The x-axis represents the number of units inserted using the same HTTP request, and the y-axis the time (in milliseconds) required to add all specified units. Like when adding information sources (section V-B1), a linear tendency is observed when the number of instances increases.

[18]https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm?utm_source=chrome-ntp-icon

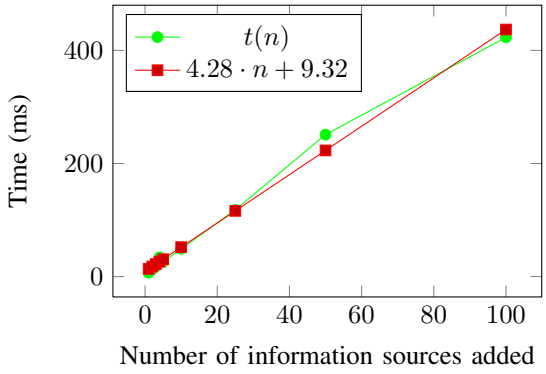

Figure 3. Time to insert information sources.

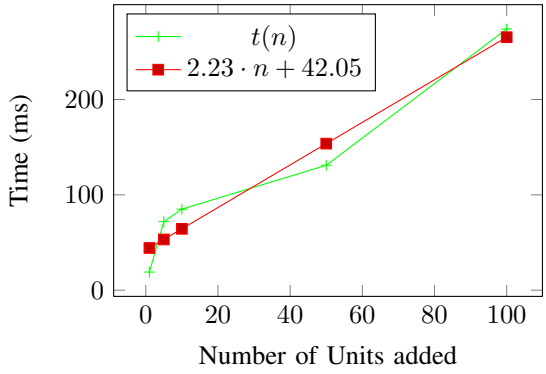

Figure 4. Time to insert units.

3) Add authors: This test evaluates the functionality of adding authors to the system. In figure 5, the x-axis represents the number of authors inserted using a single HTTP request, and the y-axis the time (in milliseconds) required to add all specified authors. As observed in the previous tests, a linear tendency is observed, being the time to add a single instance approximately constant with time.

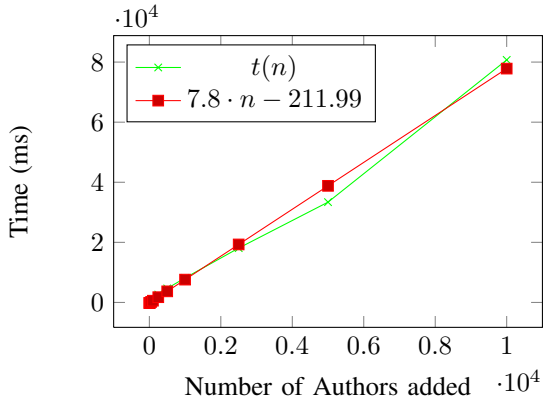

Figure 5. Time to insert authors.

4) Harvest records: This test evaluates the main functionality of the system: harvesting records from different information sources (IS)). All authors have an identifier in each of the IS, so that a real comparison is possible. In figure 6, the x-axis represents the number of harvested records and the y-axis the time required to harvest the records. It is possible

to conclude that the growth in time is largely affected by the number of harvested records and possible other factors that will be discussed in section V-C.

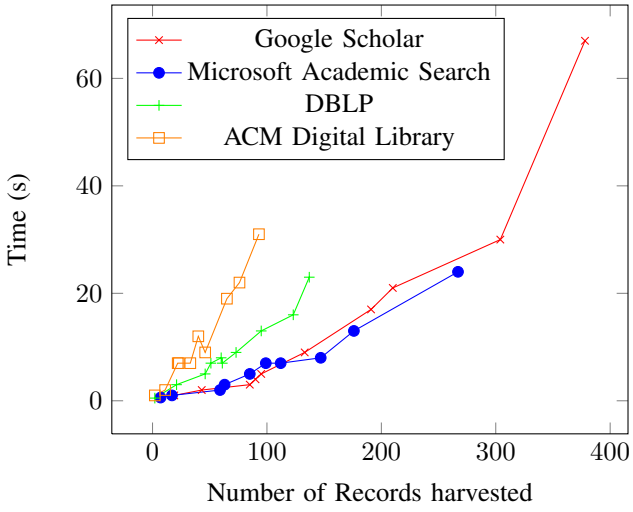

Figure 6. Time to harvest records, per information source.

5) Consolidate and access records: Another important result is related to the efficiency of the consolidation algorithm. In figure 7, the x-axis represents the total number of records in the system, that is, the number of records to consolidate, and the y-axis the time required to perform the task, in seconds. It is possible to conclude that the growth in time is largely affected by the number of records to consolidate. Section V-C presents a more detailed discussion as to the possible reasons for the obtained results.

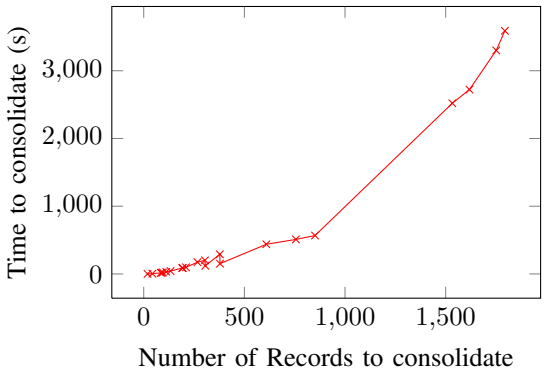

Figure 7. Time to consolidate records.

Figure 8 relates the total number of records in the system (x-axis) with the time in milliseconds required to generate and send the response to the client (y-axis). Opposite to the consolidation test, the results of accessing all existent records show a linear trend.

Figure 9 compares the results of the consolidation and accessing functionalities of the system. It is possible to see that the access time is nearly zero compared to the consolidation time. This is one of the most important results in our analysis and will be subject of an explanation in section V-C.

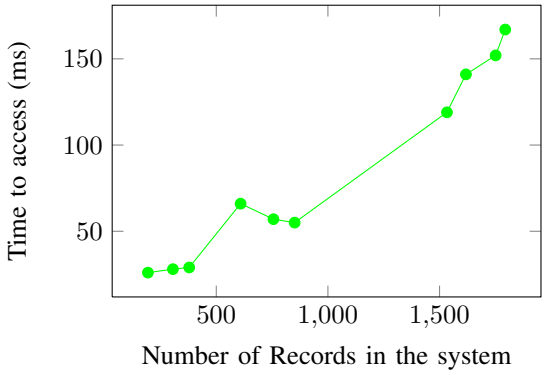

Figure 8. Time to access all consolidated records.

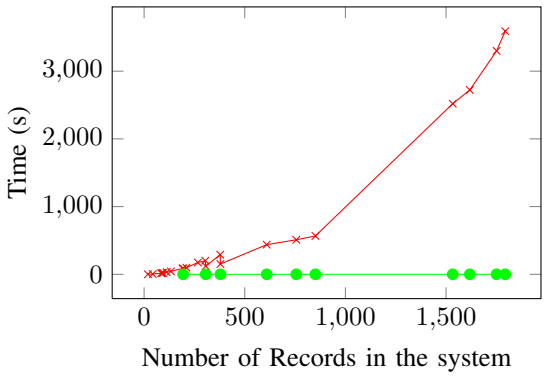

Figure 9. Comparison between time to consolidate all records and time to access all records.

C. Discussion

Analysing the results of first three tests, regarding scalability, we can see that the time required to add instances of entities grows linearly with the number of created instances. This leads us to assume that the system will have a behaviour within acceptable response times, even when adding a large number of authors.

According to figure 6, the choice of one information source has a big impact on the quantity of harvested metadata, as different information sources have a different number of records associated to an author. Google Scholar and Microsoft Academic Search are the sources that provide more records. They are also the services for which it takes the less time to harvest the same amount records. Two factors may influence these results: *1*) The transformations on the retrieved document, since Google Scholar and Microsoft Academic Search already provide the records in BibTeX format. DBLP requires an additional request for each record and ACM Digital Library provides a document that requires removal of HTML tags. *2*) The infrastructure and computational resources of Google and Microsoft should be larger, compared to the other services. Plus, it is known that these organizations have powerful search engines that may contribute to a broader reach when indexing metadata records.

About consolidation and access to the records in the system (section V-B5), it is possible to see that consolidating takes a long time. With more records in the system, the task takes

longer, as the consolidation algorithm has a complexity of $\mathcal{O}(N \times M)$ (section IV-E). With this complexity, a cautious configuration of the system is necessary regarding frequency of harvests. In relation to the access time to the records, we can see that it is almost negligible when compared to the consolidation time. Therefore we conclude that the effort in consolidation generates a high reward when accessing the data, due to the existence of cache. The results of these tests show the consequences of the flexibility in the configurable duplicate detection criteria: it is a trade-off between flexibility and execution time. If the duplicate criteria was not configurable, the consolidation process would be faster, but we would lose the possibility of choosing which fields to consider when de-duplicating.

VI. Conclusions

This thesis approached the problem of bibliographic metadata harvesting as a support for institutional repositories management, following a semi-automatic process. We introduced a system that aimed at automatically harvest metadata records of authors' publications from different sources, identify and consolidate duplicate records, and make available the existent information to outside parties.

The implemented system was evaluated in a real context environment, using real metadata records of a sample of authors at Instituto Superior Técnico, harvested from the supported information sources. Based on the results presented in section V, we can conclude that the implemented system is capable of:

1) Harvest metadata records from external sources and update the contents of the system, without introducing duplicate records.
2) Share stored information taking advantage of the available technology for interoperability, namely RESTful web services.
3) Identify and consolidate duplicate records into a smaller collection in which each record is an information-enriched version of all the records identified as duplicates.

However, as section V states, the consolidation of harvested records takes a long time. We accept this drawback because we assume that the system will focus on serving information instead of retrieving it. With the implemented solution, even with long times for consolidation, the access to the information by any external stakeholder is almost immediate.

A. Future Work

Even though the implemented system provides an answer to the research questions, there is still room for several improvements. Some possible extensions to the system include:

- Introduce support to other descriptive metadata schemas, such as Dublin Core (section II-C) or RIS[19], that allow a larger set of publication types and more descriptive fields.
- Implement support for more information sources. In this work, only four external services were used to harvest metadata records but there are more services from where to harvest records, such as ResearcherID and Mendeley. This involves introducing new functionalities such as authentication in external services and new ways of retrieving information, such as XPath or JSON parsing.
- Develop a new algorithm to detect duplicate records since that, at the moment, the system checks for equality between field values. Another possibility is to use one of the similarity measures presented in section II-E.
- Add an indexing engine such as Hibernate Search[20] or Solr[21] that add search capabilities to the system, as well as other ways of improving the detection and consolidation of duplicate records.

[19] http://www.refman.com/support/risformat_intro.asp

[20] http://www.hibernate.org/subprojects/search.html

[21] http://lucene.apache.org/solr/

References

[1] R. K. Johnson, "Institutional Repositories: Partnering With Faculty To Enhance Scholarly Communication," *DLib Magazine*, vol. 8, no. 11, 2002. [Online]. Available: http://www.dlib.org/dlib/november02/johnson/11johnson.html

[2] J. W. L. Cals and D. Kotz, "Researcher identification: the right needle in the haystack," *The Lancet*, vol. 371, no. 9631, pp. 2152–2153, Jul. 2008. [Online]. Available: http://dx.doi.org/10.1016/S0140-6736(08)60931-9

[3] P. Christen, "A Comparison Of Personal Name Matching: Techniques And Practical Issues," in *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*. IEEE, 2006, pp. 290–294.

[4] C. Lynch, "Institutional Repositories: Essential Infrastructure For Scholarship In The Digital Age," *portal: Libraries and the Academy*, vol. 3, no. 2, pp. 327–336, 2003.

[5] M. Smith, M. Barton, M. Branschofsky, G. McClellan, J. H. Walker, M. Bass, D. Stuve, and R. Tansley, "DSpace: An Open Source Dynamic Digital Repository," *DLib Magazine*, vol. 9, no. 1, 2003. [Online]. Available: http://www.dlib.org/dlib/january03/smith/01smith.html

[6] K. Markey, S. Y. Rieh, B. St. Jean, J. Kim, and E. Yakel, "Census of Institutional Repositories in the United States: MIRACLE Project Research Findings," Tech. Rep., Feb. 2007. [Online]. Available: http://www.clir.org/pubs/execsum/sum140.html

[7] C. Lagoze, S. Payette, E. Shin, and C. Wilper, "Fedora: An Architecture For Complex Objects And Their Relationships," *International Journal on Digital Libraries*, vol. 6, no. 2, pp. 124–138, 2006.

[8] N. Paskin, "Digital Object Identifier (DOI) System," *Encyclopedia of library and information sciences*, 2008.

[9] P. Bradley, "Book Numbering: The Importance Of The ISBN," *The Indexer*, vol. 18, no. 1, 1992.

[10] T. Berners-Lee, L. Masinter, and M. McCahill, "Uniform Resource Locators (URL)," Internet Engineering Task Force, RFC 1738, 1994.

[11] M. Enserink, "Are you ready to become a number?" *Science*, vol. 323, no. 5922, pp. 1662–1664, 2009. [Online]. Available: http://dx.doi.org/10.1126/science.323.5922.1662

[12] National Information Standards Organization, "Understanding metadata," 2004. [Online]. Available: http://www.niso.org/publications/press/UnderstandingMetadata.pdf

[13] A. Sen, "Metadata Management: Past, Present And Future," *Decis. Support Syst.*, vol. 37, pp. 151–173, April 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=992877.992887

[14] L. M. Chan and M. L. Zeng, "Metadata Interoperability And Standardization - A Study Of Methodology Part I," *DLib Magazine*, vol. 12, no. 6, 2006. [Online]. Available: http://www.dlib.org/dlib/june06/chan/06chan.html

[15] NISO, "Information Retrieval (Z39.50): Application Service Definition And Protocol Specification: An American National Standard," 2003.

[16] C. Lagoze, H. Van De Sompel, M. Nelson, and S. Warner, "The Open Archives Initiative Protocol For Metadata Harvesting," *Open Archives Initiative*, vol. 2004, pp. 1–6, 2008. [Online]. Available: http://www.openarchives.org/OAI/openarchivesprotocol.html

[17] S. Currier, "SWORD: Cutting Through The Red Tape To Populate Learning Materials Repositories," *JISC CETIS. Retrieved February*, vol. 11, p. 2010, 2009.

[18] J. Allinson, S. Francois, and S. Lewis, “SWORD: Simple Web Service Offering Repository Deposit,” *Ariadne*, vol. 54, no. 54, pp. 1–10, 2008. [Online]. Available: http://www.ariadne.ac.uk/issue54/allinson-et-al/

[19] R. T. Fielding, “Architectural styles and the design of network-based software architectures,” Ph.D. dissertation, University of California, Irvine, Irvine - Irvine, CA 92697, USA, 2000. [Online]. Available: http://portal.acm.org/citation.cfm?id=932295

[20] R. L. Costello, “Building web services the REST way.” [Online]. Available: http://www.xfront.com/REST-Web-Services.html