Sem Gebresilassie

Harvesting Statistical Metadata from an Online Repository for Data Analysis and Visualization

Concept application on Theseus

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Media Engineering

Thesis

21 April 2015

Author Title Number of Pages Date	Sem Gebresilassie Harvesting statistical metadata from an online repository for data analysis and visualization 35 pages + 3 appendices 21 April 2015
Degree	Bachelor of Engineering
Degree Programme	Media Engineering
Specialisation option	Web Development
Instructor	Olli Alm, Senior Lecturer

Theses and publications from Finnish universities of applied sciences are accessible from an open online repository called Theseus. This repository has an application programming interface (API) that provides tools for harvesting its contents. By properly utilizing this API, it is possible to gather and reuse metadata of thesis documents for any other objective.

This thesis mainly intends to explain how to gather the author name, title, submission year, keywords, subjects, department, university, language, and the number of pages of every thesis document in Theseus and then reuse the gathered data for building a Web portal. This Web portal provides tools to examine thesis documents and visualize statistical facts about the contribution of each university of applied sciences in Finland. To achieve this goal, robotic agents that fetch and store the metadata of thesis documents into a separate MYSQL database were created using the PHP programming language. Moreover, Google Charts API was extensively used to visualize the gathered statistical data.

The thesis first discusses the anatomy of Theseus and its communication protocol followed by a summary of concepts and technologies in data extraction process. Afterwards, it gives an illustration on the application of these concepts to parse and store metadata of every thesis document in Theseus. Finally, a brief description and benefits of the built Web portal are discussed.

Keywords	OAI-PMH, API, harvesting, parsing, Web portal, visualization, PHP, MYSQL, Web robots

Contents

Appendices

1 Introduction

Elba is an ancient kingdom that was located around what is currently known as northern Syria. It was a place where the oldest known library was created. Since then, libraries have been expanding and prospering as they continue to preserve the collective memory of the human race. In return, this expansion has caused the growth in volume of information in libraries which has led to a necessity to build specialized data structures for fast searching. Since the beginning of the 1990s, with the introduction of the World Wide Web (the Web), libraries have been extensively transferred into the Web which has become a universal repository of human knowledge and culture. [1,624.]

Theses and publications from Finnish universities of applied sciences are accessible from an open digital library named Theseus. These papers can be read and utilized by any research and development work [2]. All publicly available data on this digital repository can also be further manipulated through the use of the Theseus API. This API provides the metadata of each thesis document in different structured formats and offers options for separately querying a list of thesis documents from a particular university of applied sciences or even a particular department.

This final year project was divided into two parts. First, in the technical part, data harvesting techniques were applied to automatically fetch metadata such as author name, title, keywords, submission year, department, subjects, university, language, and number of pages from every thesis document in Theseus and store the information in a separate MYSQL database. The stored data was then later used to develop a prototype Web portal that can be used as a tool for analysing thesis documents and visualizing overall statistical facts about each university of applied sciences in Finland. This was achieved by using PHP and visualization tools from Google Charts API. Secondly, in this written part of the project, a special attention was given to elaborate the processes and techniques involved in harvesting Theseus to selectively extract statistical metadata necessary for the Web portal project. The thesis first starts by describing the anatomy of Theseus and the communication protocol that governs it, followed by theoretical concepts behind technologies used in data extraction process and finally it gives an illustration on the usage of these technologies to build the Web portal. The thesis also documents the problems that arose in the development process.

2 Theseus

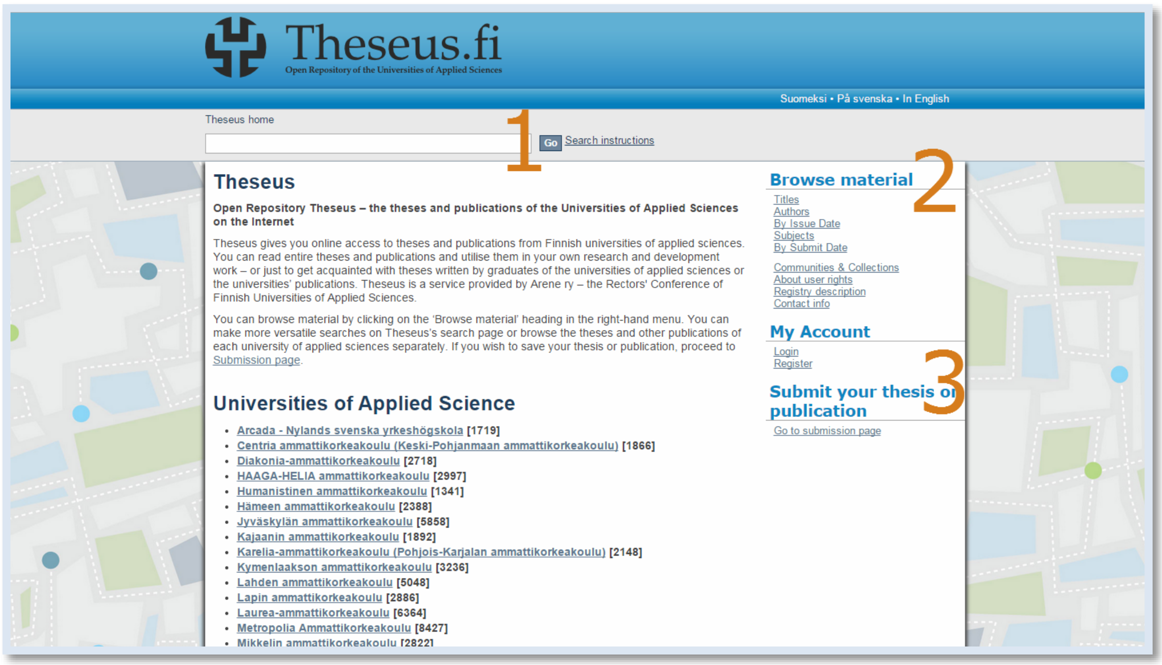
As the processing power of computers and the Web improve, information is being given a well-defined structure. This progression not only has made finding useful information over the World Wide Web easier for quick searching, but also has facilitated data analysis and data mining feasibilities.

Online repositories promote the preservation of structured intellectual outputs as they benefit more and more from the convergence of technology developments and digital assets. In recent years, the significant drop in storage and networking cost has made digital libraries and repositories more affordable than ever. For this reason alone, Web based databases (repositories) for managing scholarly materials are commonly offered by universities and institutions across the globe [4].

Finnish universities of applied sciences collectively use such online digital library, Theseus, to provide an open access to their theses and publications. Theseus.fi provides ways for visitors to search thesis documents for reading or to browse the whole content by thesis title, author, subject and more.

2.1 Features and Functionalities of Theseus.fi

Theseus.fi provides three key features for its end users. These are the search, browse and upload features. The search feature in Theseus, represented by number 1 in figure 1 below, lets users search the entire library or just specific collections. If users are only interested in search results from a specific collection, they can limit their search scope after making their search on the front page and using the options that will appear under the search box. In addition, the Theseus search box accepts different search formulates such as Field, Boolean, Wild card, Fuzzy, Proximity and Relevance that are well documented in its search instructions [2]. At the time of writing this thesis (April 2015), there were 84,391 theses and publications that are already available for an open access. Using the Field-search option, specific metadata about each document such as title, author, abstract, date (when the work was accepted), identifier, university (name of the school), programme (degree programme), language, and keywords (descriptive words given by the author) can be used to search and find any of the documents.


Theseus.fi
Open Repository of the Universities of Applied Sciences
Suomeksi • På svenska • In English
Theseus home
Go
Search instructions
1
2
3

Theseus

Open Repository Theseus – the theses and publications of the Universities of Applied Sciences on the Internet

Theseus gives you online access to theses and publications from Finnish universities of applied sciences. You can read entire theses and publications and utilise them in your own research and development work – or just to get acquainted with theses written by graduates of the universities of applied sciences or the universities' publications. Theseus is a service provided by Arene ry – the Rectors' Conference of Finnish Universities of Applied Sciences.

You can browse material by clicking on the 'Browse material' heading in the right-hand menu. You can make more versatile searches on Theseus's search page or browse the theses and other publications of each university of applied sciences separately. If you wish to save your thesis or publication, proceed to Submission page.

Universities of Applied Science

- Arcada - Nylands svenska yrkeshögskola [1719]
- Centria ammattikorkeakoulu (Keski-Pohjanmaan ammattikorkeakoulu) [1866]
- Diakonia-ammattikorkeakoulu [2718]
- HAAGA-HELIA ammattikorkeakoulu [2997]
- Humanistinen ammattikorkeakoulu [1341]
- Hämeen ammattikorkeakoulu [2388]
- Jyväskylän ammattikorkeakoulu [5858]
- Kajaanin ammattikorkeakoulu [1892]
- Karelia-ammattikorkeakoulu (Pohjois-Karjalan ammattikorkeakoulu) [2148]
- Kymenlaakson ammattikorkeakoulu [3236]
- Lahden ammattikorkeakoulu [5048]
- Lapin ammattikorkeakoulu [2886]
- Laurea-ammattikorkeakoulu [6364]
- Metropolia Ammattikorkeakoulu [8427]
- Mikkelin ammattikorkeakoulu [2822]

Browse material

Titles
Authors
By Issue Date
Subjects
By Submit Date

Communities & Collections
About user rights
Registry description
Contact info

My Account

Login
Register

Submit your thesis or publication

Go to submission page

Figure 1. Landing page of Theseus portal [2.]

On the other hand, the browsing feature, represented by number 2 in figure 1, lets users filter out theses and publications ordered by for example collection (universities or department name), title or author name. Typically, browsing starts with a collection. For instance, by selecting a particular university from a list of universities on the front page a list of departments in that particular university can be retrieved and when a single department is selected from this retrieved list another list of individual thesis documents will be displayed. In this way, end users can utilize the browsing feature to access thesis documents of their desire.

The upload feature, represented by number 3 in figure 1, is for students and authors who want to publish their thesis or publication to the repository. By registering and creating a profile, it is possible to upload documents in electronic format to make them available as open access publications on Theseus.fi. When uploading a thesis or publication, uploaders are required to submit author name, title of the thesis and other detailed information about their document. This information is later used as the metadata of the uploaded document in retrieval processes.

Theseus is powered by a pioneer open source digital asset management system named Dspace, provided by a non-profit organization called Duraspace [4].

2.2 Dspace

Dspace is an open source software platform that provides stable, long-term storages commonly for digital intellectual materials. Originally launched in November 2002 by MIT in collaboration with HP, this flexible and customizable software platform captures and describes digital materials that are submitted over its forms. Dspace grants data providers the ability to offer their users an easy access to contents with minimal to no customization of the application. It also allows individual documents to be well organized and described in its built in structure. As an open source software, Dspace can be freely downloaded and used or even modified to store digital materials for any unspecific need. Some features of Dspace include searching and unified browsing. [4.] These features ease the process of accessing relevant contents in Dspace repositories. Theseus's features and functionalities discussed above are directly linked to this nature of Dspace.

2.2.1 Getting Data out from Dspace driven Repositories

OAI, short for Open Archives Initiative, is an enterprise that develops and promotes standards for transferring digital objects or metadata from one system to another aiming to facilitate the efficient dissemination of contents in online repositories. Such standards are called interoperability standards. Open Archives Initiative Protocol for Metadata Harvesting (abbreviated as OAI-PMH) is an interoperability standard developed by this enterprise that defines clear methods and protocols for accessing contents from Dspace repositories. Dspace uses OAI-PMH to define methods for sharing, publishing and archiving metadata, to enable access to Web materials within repositories such as Theseus. [4; 5]

Overall, Open Archives Initiative (OAI) driven repositories provide an API that can be used by third party organizations to utilize their data. In this manner, OAI provides application independent framework that helps establish metadata harvesting processes between the following two participants.

- Data providers that allow OAI-PMH for exposing their metadata
- Data consumers that harvest and use metadata via the OAI-PMH for different operations

It is estimated that about 75% of academic repositories worldwide use the standard OAI-PMH protocol to provide access to their digital intellectual materials. In authenticity to this adherence, some or all of the metadata about each intellectual material in such repositories is exposed for harvesting by external data consumers. When data consumers (harvesters) request for data access, the returned metadata from OAI-PMH data providers is XML formatted metadata and usually includes a URL for the full text file which can potentially be further processed if required. [8.]

2.2.2 Dspace OAI-PMH, the Data provider for Theseus

Whenever a need to access data from a third party website arises, there is a good chance a developer starts his/her work by checking to see if there is an official application programing interface (API). APIs provide tools essential to build software and services that use data from external sources. They specify how these software and services interact with the data source by describing a set of methods and protocols for accessing the data. Many companies make use of Web APIs to uncover data and functionality in their existing system.

Theseus API provides harvesters a way for accessing metadata of theses and publications from Finnish universities of applied sciences. This API uses open archives initiative protocol for metadata harvesting (OAI-PMH) to deliver thesis documents in different metadata formats from Theseus to harvesters. Theseus OAI-PMH exposes thesis documents in twelve unique metadata formats. Each metadata format has the following common properties [20].

- metadataPrefix
- schema URL and
- XML namespace URI

The metadataPrefix is a string consisting of any URI-unreserved characters to uniquely specify the format during OAI-PMH communication. The schema URL is the URL of the XML schema that associates defined sets of rules to test validity of the metadata. The XML namespace URI is a global identifier of the metadata format.

Table 1. Metadata formats in Theseus

Metadata prefix	Full name
UKETD_DC	United Kingdom Electronic Thesis and Dissertation – Dublin Core
OAI_DC	Open Achieve Initiative – Dublin Core
MARC	Machine Readable Cataloging
ETDMS	Electronic Thesis and Dissertation Metadata Standard
QDC	Qualified Dublin Core
RDF	Resource Description Framework
QDC_finna	Qualified Dublin Core - Finna search service
ORE	Object Reuse and Exchange
KK	Kansallis Kirjasto, a metadata format provided by the national library of Finland.
MODS	Metadata Object Description Schema
METS	Metadata Encoding and Transmission Standard
DIDL	Digital Item Declaration Language

For instance, the following samples show the author name of a single thesis document in three different metadata formats.

MARC format : `<subfield code="a"> Denut, Nicolae </subfield>`

OAI DC format : `<dc:creator> Denut, Nicolae </dc:creator>`

KK format:

```
<kk:field schema="dc" element="contributor" qualifier="author"
language="none" value=" Denut, Nicolae "/>
```

For purposes of basic communication between data providers and harvesters, OAI-PMH requires data providers to at least offer the “oai_dc” metadata format shown in table 1. The “oai_dc” metadataPrefix refers to OAI DC metadata format provided by an initiative named The Dublin Core Metadata Initiative (DCMI). However, within some groups and institutions other metadata specifications may be provided as is the case of Theseus. This is because it is sometimes necessary to adequately describe resources with complex structures in a specialized way for special needs. Whichever metadata format is chosen by data consumers, an agreement on its use with the data providers must be reached. [9; 19.]

Depending on the interest of the developer, thesis documents can be delivered in any chosen format from table 1 above. Each metadata format can be queried to harvest any document from the repository for different objectives. OAI-PMH also supports incremental harvesting allowing harvesters to retrieve only the records which have changed since the last successful harvest. [5.]

2.3 OAI-PMH Principles that apply when Harvesting Theseus

The OAI-PMH protocol works based on HTTP to allow communication between applications issuing OAI-PMH requests (data consumers) and repositories (data providers) to harvest metadata and return XML formatted metadata respectively.

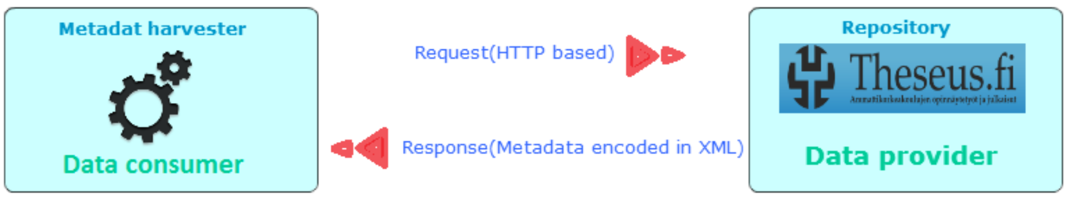


Figure 2. Basic OAI-PMH communication. Modified from the Open Archives Forum (OAF) (2015) [7.]

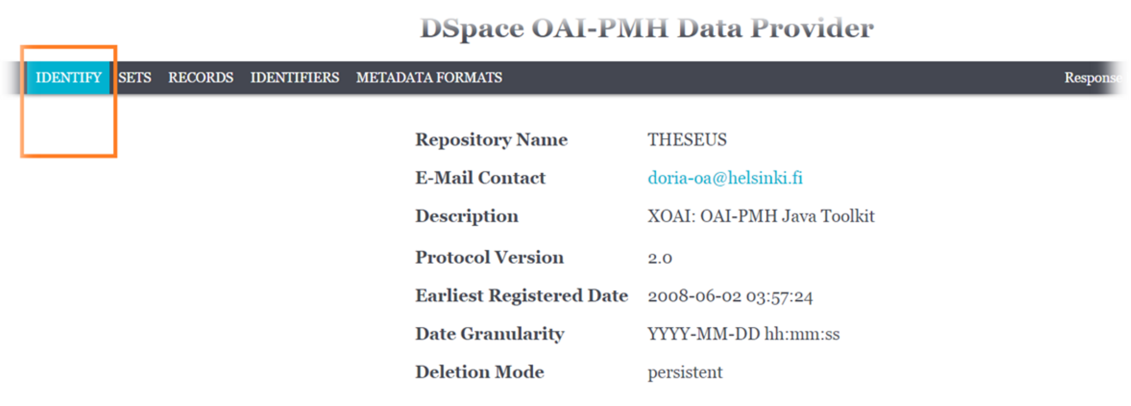
As an OAI based Dspace repository, Theseus has a separate data provider that is organised and structured in a model convenient for OAI-PMH communications. Consequently, Theseus has an OAI based URL at http://publications.theseus.fi/oai/request? in addition to its URL for human users at www.theseus.fi. On its own, the OAI based URL of Theseus simply returns XML error message. Its real power unveils when a proper request type is appended to it. [6; 18.]

In OAI-PMH, there are six request types (known as "verbs") that can be appended to OAI based URLs together with other arguments to access repository contents. [18.] Harvesting data as a client does not require the use of all request types. Nevertheless, data providers must implement all request types. Depending on the request type, it might also be necessary to use additional required or optional arguments for an effective response during data harvesting. [5.] In the following section, each request type in OAI-PMH communications is described and explained in relation to Theseus.

2.3.1 Request Types

Although OAI-PMH is intended for machine-to-machine communication, it returns results as XML that can be displayed by all major Web browsers. [9.] For this reason, the examples that follow are given as direct links.

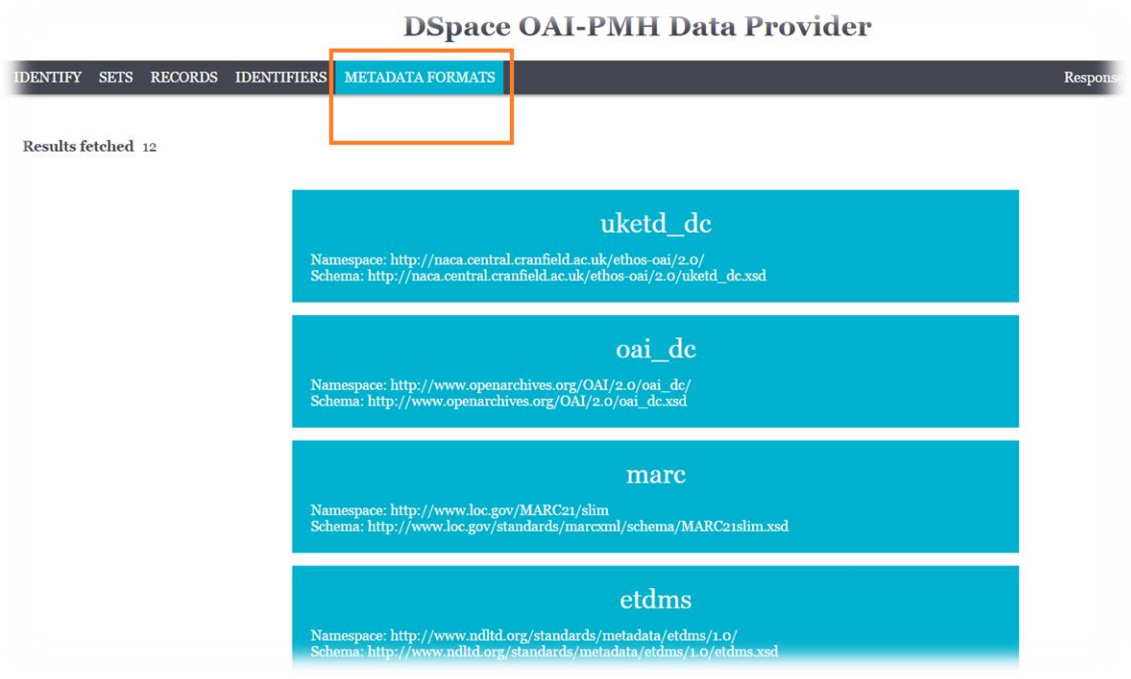
Identify: This is a function or request type that returns information about the data provider itself mainly for describing it. Appending this function to the end of Theseus's OAI based URL will result http://publications.theseus.fi/oai/request?verb=Identify. As it can be seen in figure 3 below, Theseus returns XML formatted description of its data provider for harvesters when this request URL is used to make harvesting request.



Repository Name	THESEUS
E-Mail Contact	doria-oa@helsinki.fi
Description	XOAI: OAI-PMH Java Toolkit
Protocol Version	2.0
Earliest Registered Date	2008-06-02 03:57:24
Date Granularity	YYYY-MM-DD hh:mm:ss
Deletion Mode	persistent

Figure 3: Response of Theseus to “Identify” request

ListMetadataFormats: This is another function that lists all the metadata formats supported by data providers. Appending this function to the end of Theseus’s OAI based URL will give http://publications.theseus.fi/oai/request?verb=ListMetadataFormats. This request URL can be used when harvesting a list of metadata formats supported by Theseus’s data provider.



Results fetched 12

uketd_dc
Namespace: http://naca.central.cranfield.ac.uk/ethos-oai/2.0/
Schema: http://naca.central.cranfield.ac.uk/ethos-oai/2.0/uketd_dc.xsd

oai_dc
Namespace: http://www.openarchives.org/OAI/2.0/oai_dc/
Schema: http://www.openarchives.org/OAI/2.0/oai_dc.xsd

marc
Namespace: http://www.loc.gov/MARC21/slim
Schema: http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd

etdms
Namespace: http://www.ndltd.org/standards/metadata/etdms/1.0/
Schema: http://www.ndltd.org/standards/metadata/etdms/1.0/etdms.xsd

Figure 4: Partial list of Theseus’s metadata formats

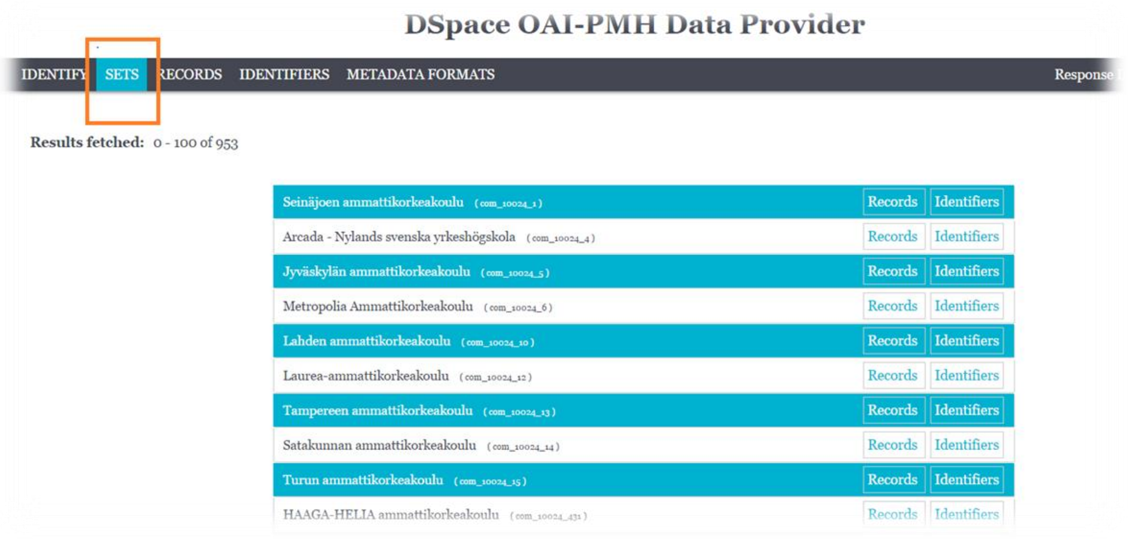
ListIdentifiers: In Theseus, ListIdentifiers are used to list theses identifiers (that uniquely identify each thesis document), date stamps (that shows the last modification date) and two set specs (set identifiers), one for university identifier and another for department identifier of each thesis document.

This request type requires an additional argument, “metadataPrefix”, for specifying the chosen metadata format of the required data. For example, appending this request type to Theseus’s OAI based URL and setting the metadataPrefix to kk (kansalliskirja-sto, a metadata format provided by national library of Finland) will give the request URL http://publications.theseus.fi/oai/request?verb=ListIdentifiers&metadataPrefix=kk.This request URL can be used when harvesting a list of thesis record identifiers, dates and set specs of each thesis document. Listing 1 below shows the partial XML response from Theseus to ListIdentifiers request.

```
<request verb="ListIdentifiers" metadataPrefix="kk">
        http://publications.theseus.fi/oai/request
</request>
  <ListIdentifiers>
      <header>
         <identifier>oai:www.theseus.fi:10024/474</identifier>
          <datestamp>2013-08-19T10:18:05Z</datestamp>
          <setSpec>com_10024_14</setSpec>
          <setSpec>col_10024_174</setSpec>
      </header>
      <header>
         <identifier>oai:www.theseus.fi:10024/592</identifier>
          <datestamp>2013-05-06T15:30:26Z</datestamp>
          <setSpec>com_10024_12</setSpec>
          <setSpec>col_10024_270</setSpec>
      </header>
```

Listing 1. XML response to ListIdentifiers request (partially)

ListSets: This function lists all sets prearranged by Theseus (universities of applied sciences and departments). Sets in Theseus are groups representing each university of applied sciences and department. Each set contains its own list of thesis documents. For example, appending ListSets to Theseus’s OAI based URL will give http://publications.theseus.fi/oai/request?verb=ListSets. When harvesting request is made using this request URL, Theseus’s data provider returns a list of universities and departments shown in figure 5 below.

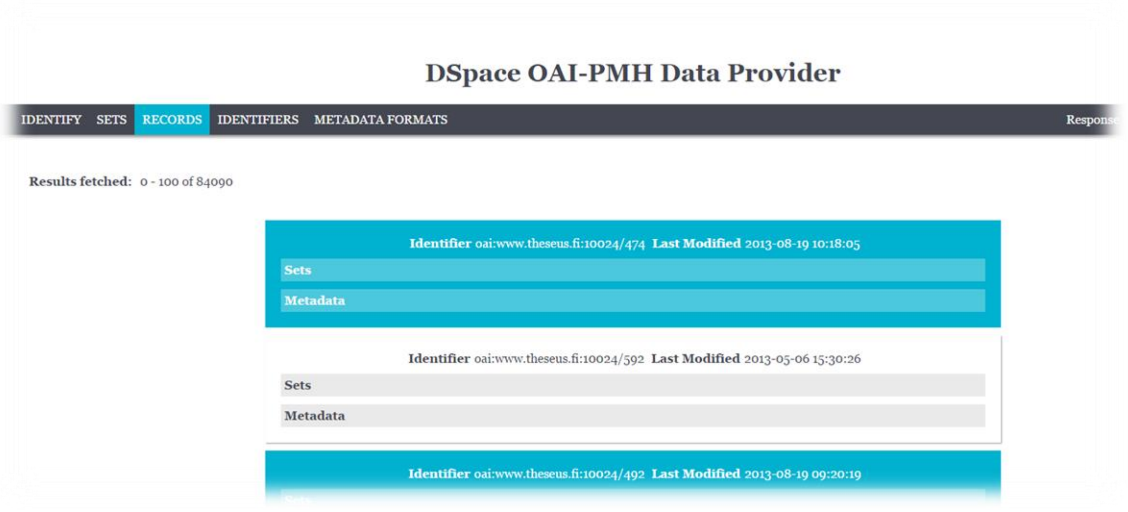


Results fetched: 0 - 100 of 953

Seinäjoen ammattikorkeakoulu (com_10024_1)	Records	Identifiers
Arcada - Nylands svenska yrkeshögskola (com_10024_4)	Records	Identifiers
Jyväskylän ammattikorkeakoulu (com_10024_5)	Records	Identifiers
Metropolia Ammattikorkeakoulu (com_10024_6)	Records	Identifiers
Lahden ammattikorkeakoulu (com_10024_10)	Records	Identifiers
Laurea-ammattikorkeakoulu (com_10024_12)	Records	Identifiers
Tampereen ammattikorkeakoulu (com_10024_13)	Records	Identifiers
Satakunnan ammattikorkeakoulu (com_10024_14)	Records	Identifiers
Turun ammattikorkeakoulu (com_10024_15)	Records	Identifiers
HAAGA-HELIA ammattikorkeakoulu (com_10024_431)	Records	Identifiers

Figure 5. ListSets response in partial

ListRecords: This function also requires additional argument “metadataPrefix” and it is used to fetch a list of thesis metadata from Theseus. Other optional arguments can also be appended to limit result lists to a specific subset. For example, it is possible to add set specs (university and department identifiers) as arguments in order to retrieve a list of thesis metadata from a single department or university of applied sciences only.



Results fetched: 0 - 100 of 84090

Identifier oai:www.theseus.fi:10024/474 **Last Modified** 2013-08-19 10:18:05
Sets
Metadata

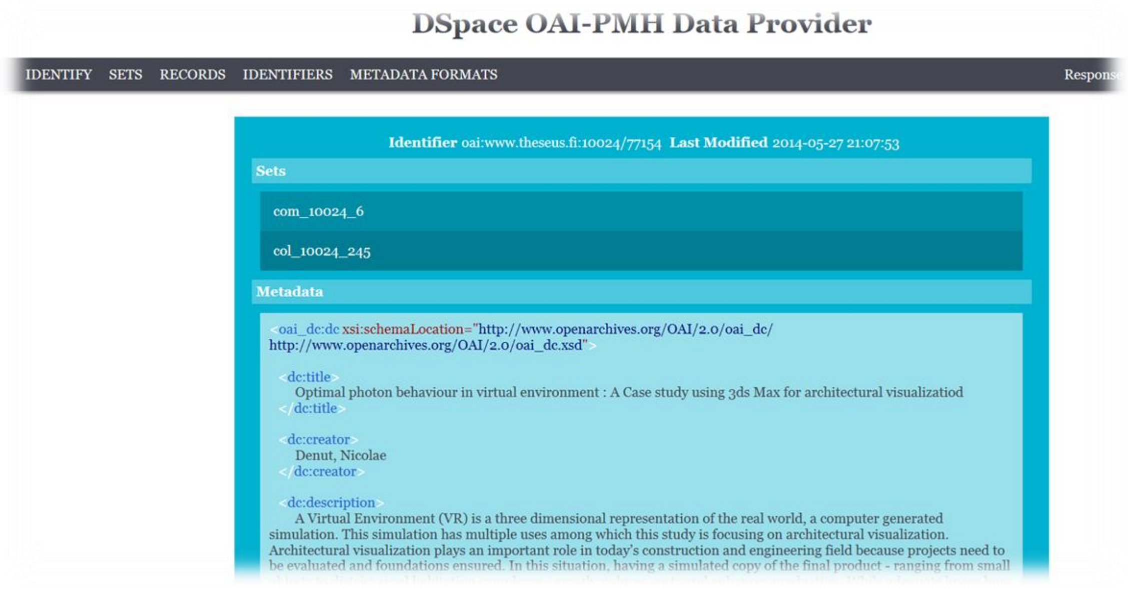
Identifier oai:www.theseus.fi:10024/592 **Last Modified** 2013-05-06 15:30:26
Sets
Metadata

Identifier oai:www.theseus.fi:10024/492 **Last Modified** 2013-08-19 09:20:19

Figure 6. Partial ListRecords response showing fetched result

Appending ListRecords to Theseus's OAI based URL and setting the metadataPrefix to kk again, will give the request URL http://publications.theseus.fi/oai/request?verb=ListRecords&metadataPrefix=kk, which can be used by harvesters to access the response shown in figure 6 above.

GetRecords: This function can be used to access an individual thesis document from the repository. It requires the combination of thesis record 'identifier' and 'metadataPrefix' arguments. For example the request URL http://publications.theseus.fi/oai/request?verb=GetRecord&metadataPrefix=kk&identifier=oai:www.theseus.fi:10024/77154 is used to retrieve the metadata of the a single thesis document shown in figure 7 below,



Identifier oai:www.theseus.fi:10024/77154 **Last Modified** 2014-05-27 21:07:53

Sets

com_10024_6

col_10024_245

Metadata

```
<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
http://www.openarchives.org/OAI/2.0/oai_dc.xsd">

  <dc:title>
    Optimal photon behaviour in virtual environment : A Case study using 3ds Max for architectural visualizatiod
  </dc:title>

  <dc:creator>
    Denut, Nicolae
  </dc:creator>

  <dc:description>
    A Virtual Environment (VR) is a three dimensional representation of the real world, a computer generated
simulation. This simulation has multiple uses among which this study is focusing on architectural visualization.
Architectural visualization plays an important role in today's construction and engineering field because projects need to
be evaluated and foundations ensured. In this situation, having a simulated copy of the final product - ranging from small
```

Figure 7. GetRecords request returning metadata of a single thesis document

2.3.2 Flow control

Theseus has a fairly large set of data. Results from the above listed functions or request types, can sometimes get messy and too long to display on web browsers. In such cases, the repository maintains measure of flow control by using articulates known as "resumption tokens". [8, 18.] Resumption tokens are options from OAI protocol that allow data providers to chunk long list responses in parts. Implementation of resumption tokens is beneficial for both the data providers and data consumers when handling a large set of data.

The four request types - ListMetadataformat, ListIdentifiers, ListSets and ListRecords - return a list of items [8]. Three of them - ListIdentifiers, ListSets and ListRecords - return large lists from Theseus. In such cases, Theseus OAI-PMH supports partitioning of a list items that make use of resumption tokens. By default, Theseus's data provider returns 100 list items and a resumption token when a response list contains more than 100 items. In order to get the next items in the response list, a second request has to be made using the resumption token as an argument. The second request also returns another list of items and new resumption token continuing from the first response. This process is repeated until the complete list of items is gathered. This work flow is better explained in figure 8 below.

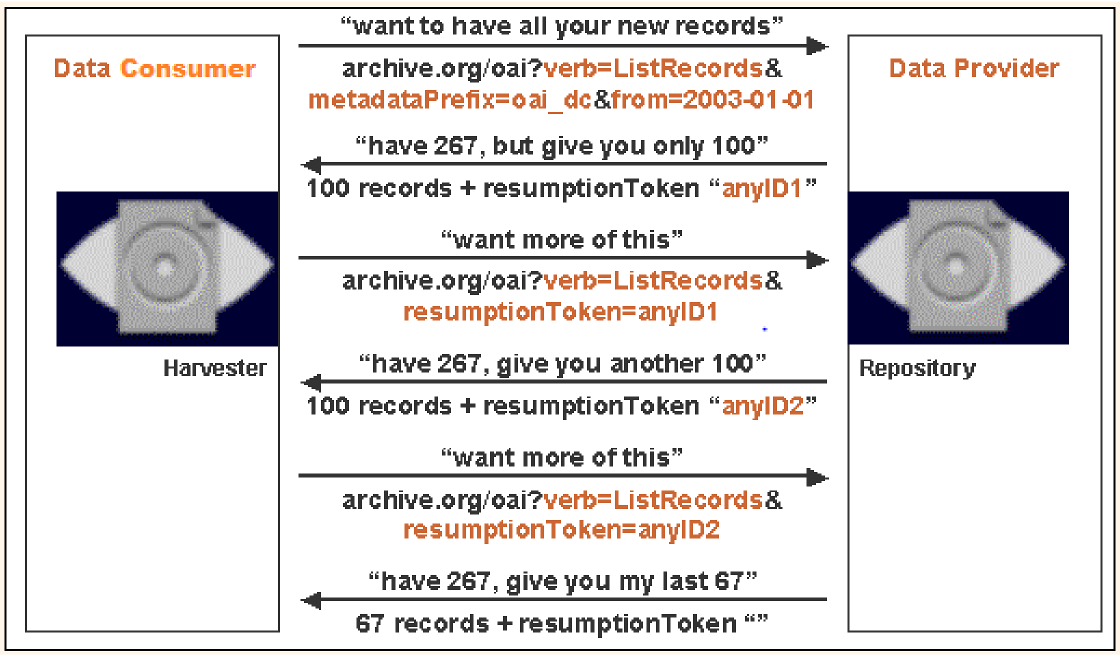

Data Consumer
Harvester
Data Provider
Repository
"want to have all your new records"
archive.org/oai?verb=ListRecords&metadataPrefix=oai_dc&from=2003-01-01
"have 267, but give you only 100"
100 records + resumptionToken "anyID1"
"want more of this"
archive.org/oai?verb=ListRecords&resumptionToken=anyID1
"have 267, give you another 100"
100 records + resumptionToken "anyID2"
"want more of this"
archive.org/oai?verb=ListRecords&resumptionToken=anyID2
"have 267, give you my last 67"
67 records + resumptionToken ""

Figure 8. Resumption token work flow [5.]

The anatomy of an example resumption token in Theseus is shown below

```
<resumptionToken completeListSize="84090" cursor="2">
MToxMDB8Mjp8Mzp8NDp8NTpraw==
</resumptionToken>
```

For every new list requests, harvesters must append resumption tokens as a parameter to request URLs. The resumption token is empty if the list returned is the last section. [18.]

In summary, Theseus’s data provider uses the OAI-PMH protocol for exposing metadata of thesis documents to data consumers who wish to use the data for different purposes. Data consumers issue OAI-PMH requests to Theseus’s data provider to get the XML formatted metadata of thesis documents. There are six request types that must be submitted by harvesters using HTTP methods to get the metadata. These request types are listed below.

1. **Identify:** fetches information about Theseus data-provider itself
2. **ListMetadataFormats:** returns a list of available metadata formats supported by a Theseus data-provider
3. **ListIdentifiers:** lists thesis record identifiers, dates & other headers of each thesis document
4. **ListSets:** retrieves the set structure (list of universities and departments) .
5. **ListRecords:** gets list of complete metadata of thesis documents from a Theseus and
6. **GetRecord:** retrieves individual metadata of a thesis document

Theseus’s response to ListIdentifiers, ListSets and ListRecords request types is large. For this reason, the repository replies to these requests with an incomplete list and a resumption token. In order to make the response a complete list, harvesters are required to issue multiple requests with resumption tokens as arguments. This multiple request response cycle is referred to as Flow control. [20.]

3 Fundamental Concepts in Data Harvesting

The task of obtaining relevant and useful information from a large data set requires assistance from automated information extraction systems. Such systems are in existence today with the help of software programs and programming concepts that provide tools for building high-performance, natural language processing applications. Fundamentally, automated web harvesting from the Web requires the use of web robots that apply web crawling, web harvesting and or parsing techniques to meet the goal. [10.] "OAI-PMH harvesters are robotic agents and care should be taken to avoid creating an accidental denial-of-service attack against repositories" [20].

3.1 Web Robots (Internet Bots)

Repetitive tasks are not only tedious and time consuming for computer users; they also create a gap for errors to occur. Enter Web robots, in their productive nature bots are software applications that are capable of performing computer tasks automatically at a much higher rate than their human counterpart. They can be used for extracting information from the internet, gathering/harvesting comparison data, examining a website for errors or invalid links, or even handling more advanced matters such as crawling websites on the internet. The most important objective of internet bots is to transfer required webpage content from an online data sources to a separate storage. [10; 11.]

Web crawling: In broader sense, the process of finding the most relevant and desired content involves deploying internet bots to follow links and iterate through URLs of webpages. This practice is called web crawling. Understanding the techniques used in web crawling can be a good start in writing data extraction software for a specific webpage.

Web scraping: Contrary to Web APIs, the ability to grab any online data while having the complete freedom to choose how to store and retrieve it requires the knowledge and understanding of another computer programming concept known as Web scraping. Web scraping (also known as Web harvesting or Web data extraction) is a technique in computer science to automatically extract data by parsing structured or semi-structured web contents such as the HTML of websites.

Web browsers locate, retrieve and display content from the World Wide Web. Software written for Web scraping purposes interacts with website hosts in the same manner as a Web browser. The only difference is, it holds on the data for further manipulation instead of displaying it. For this particular nature, Web scraping software is used in different areas around the Web. Whether it is for comparing prices from different sites or detecting changes on webpages or even creating a Web mashup, the possibilities are wide open and endless. [13; 14.]

3.2 XML Parsing

In general, parsing may be defined as the act of taking a set of data and breaking it apart into its components to help extract the meaningful information out of it. For example, XML parsing can be understood as a process of identifying the tags and attributes inside it with their relation to each other.

XML processors are more commonly referred to as parsers. There are typically two types of XML parsers, XML DOM parsers and XML SAX parsers

DOM parsers: These type of XML parsers work by creating a Dom Object Model interface of the entire XML document to navigate, add, modify or delete parts of it while still in memory. This approach is typically used for small XML structures. [15.]

SAX parsers: These parsers (also called “Simple API for XML parsing”) are event based parsers. It is mainly useful to extract specific tags and attributes from a large XML document. Unlike DOM parsers, a SAX API never has to hold the whole document in memory, just the parts it is interested in. [15.]

3.3 Simple HTML DOM Parser

Parsing can be done in many ways. Simple HTML DOM parser, is an open source parser library written in PHP to read, modify, and return structured content from external data sources. This parser library can create an object either by loading structured data from a string, or from a file on a computer. Loading a file can be done either via URL, or from a local file system. For example, loading file via URL can be achieved in the following manner.

```
$content=
file_get_html('http://publications.theseus.fi/oai/request?verb=L
istSets');
```

Once a DOM object is created this way and stored in the "$content" variable, getting the contents of the DOM object can be attained by using a method called "find ()". For example, to get each "record" tag from the above URL of Theseus's ListSets response, it is possible to simply use a *foreach* construct in PHP as follow.

```
foreach ($content->find('record') as $element) {
    Do something…
}
```

Other methods to get tags and their attributes are also provided by the parser in a syntax that is quite similar to jQuery. This parser can be freely downloaded for any use and it was used to harvest thesis metadata from Theseus for building the web portal project.

3.4 PHP built-in Functions

PHP.net provides a helpful documentation for different pre-defined functions at its website. Some of these functions are extensively used while working on the data harvesting part of the practical project described in this thesis.

4 Parsing Data from Theseus's Data provider

Part of the practical project was parsing thesis documents from Theseus's data provider using a parser library called Simple HTML DOM parser. The data gathered was later stored in a MYSQL database.

4.1 Preparation

Statistics is vital for producing variety of interpretational information based on a data set and thus it is useful and meaningful scientific knowledge [12]. However, understanding data and its patterns is far easier for most people when visualization methods are used to put the numbers as pictures. On this ground, a challenge was taken to build a Web portal that can present the statistics of thesis data, from all universities of applied sciences in Finland, in a more appealing manner with visualization. When producing this Web project, different target groups were foreseen to show an interest including universities and students around the country.

At its current stage, the Web portal is a good tool to see how thesis submissions to theseus.fi have increased throughout the years since the first submission. Users can now judge and compare universities of applied sciences based on their thesis submission rate every year or based on the total number of thesis papers in Theseus from each of them. Moreover, if users are interested to know which department in a given school is producing the highest number of thesis documents so far, the web portal makes it easy to see this information in a pie chart. In this pie chart, each department will have its own section showing its share of submitted thesis documents in percent. On this Web portal, comparing thesis documents with each other according to the number of pages or seeing keywords of thesis documents from any department in any given school and in any given year is just two clicks away. A list of keywords from a chosen year is intended to help users make their own analysis on how trending topics in each field of study are being more practiced and explored by students.

Theseus contains 84,391 theses documents from students in twenty five universities of applied sciences. In order to build the web portal, it was necessary to parse all these documents and harvest interesting data for visualization. Before proceeding with the parsing process, a decision was first made on what kind of data about these documents is interesting enough to be extracted for inclusion. The web portal built provides users a statistical breakdown of data about these thesis documents. For this reason, collecting metadata with variable nature that can answer the questions listed below was necessary.

- How many thesis documents are in Theseus?
- Which school has what amount of papers in Theseus?
- How many papers is each school publishing every year?
- What departments are there in each school?
- How many theses belong to which department?
- How many pages does each thesis have?
- In what language are the theses written?
- How many times has each paper been downloaded by Theseus visitors?
- What are the keywords of each thesis document?

Additionally, information on individual thesis documents such as author name, school it belongs to, department, title of thesis, or number of pages and year of publication were gathered and stored as essential components.

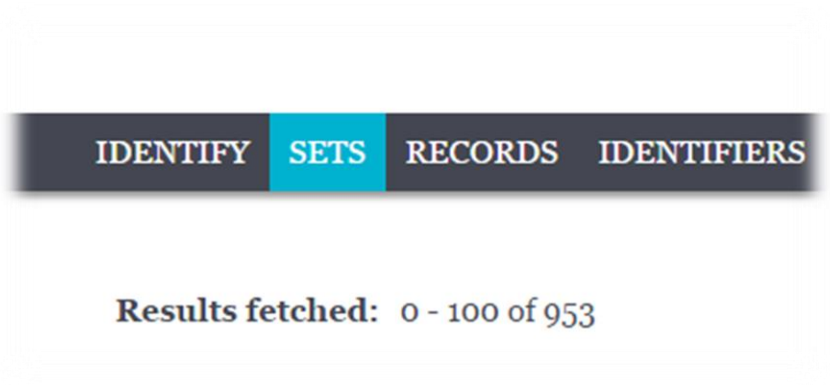
4.2 Choosing Metadata Format

The structure and organization of the same information is different in every metadata format. Even though it was expected that the information carried by each metadata format is the same, it was not the case for Theseus. For this reason, it was very important to decide what metadata format to use before starting to harvest. This decision was crucial because the web portal was built based on metadata harvested from Theseus. As Theseus OAI-PMH implements multiple metadata formats, identifying the metadata format that has all the information needed for the project was a must. Unfortunately, the development process of this project was in a halt multiple times because this decision was not made on time. Some harvesting was done with “OAI DC” and then “METS” metadata formats that had to be abandoned because of error responses and data inconsistency. After further technical instructions from the National Library, the “KK” metadata format was proved to be reliable for use and it was utilized during the harvesting process.

4.3 Parsing Process

It is apparent that querying the metadata from the Theseus data provider can be done in a variety of programming languages such as Java, Visual Basics, PHP or Python. PHP was chosen for this particular project because the language is famous for its simple syntax, convenient string-parsing capabilities and portability [11].

After setting up a PHP development environment, the parsing process began by first making a “ListSets” request with the help of Simple HTML DOM parser. ListSets request to Theseus returns lists of universities and departments that are used as sets. Since the list of universities and departments is large (953 items to be specific), Theseus implements the use of “resumptionToken” to chunk this list of sets and returns 100 list items per a single request.



Results fetched: 0 - 100 of 953

Figure 9. List request returning an incomplete list.

In order to get the complete list, it is required to issue multiple requests using resumption tokens as arguments. For example, to retrieve the second page of the set list, resumption token from the first response can be appended to the new request URL before making a new request. The third page of the set list can also be accessed by appending the corresponding resumption token and making another new request and so on. Using the Simple HTML DOM parser and PHP, this can be done by first making a request to get the first page of the list as shown below.

```
<?php
    require 'simple_html_dom.php';
    $firstPage=file_get_html('http://publications.theseus.
    fi/oai/request?verb=ListSets');
?>
```

To make the next request, it is required to parse through the response from the first request and get the resumption token and store it in a variable.

```
$resumptionToken=$firstPage->find('resumptionToken',0)->plaintext;
```

The next request must use the value of the `$resumptionToken` as the value of the resumption token argument.

```
$secondPage
=file_get_html('http://publications.theseus.fi/oai/request?verb=ListSe
ts&resumptionToken=$resumptionToken);
```

By creating a PHP loop, this process can be repeated to crawl through each section of the set list response. The concatenation of these sequential chunks of crawled lists from sequential requests will form a complete list of all departments and university of applied sciences in Theseus. Such a sequential list request is known as list request sequence. [19.]

Theseus returns 100 set items (schools and departments) when a “ListSets” request is made. Therefore, creating a “for loop” that iterates nine times, making new list request, will suffice to get the complete list. However, since this crawling process will be repeated the same way to get the list of thesis documents (84,391 list items), it is a good idea to device a generic code that can be reused when necessary. To accomplish this, a special *function* was created.

This function takes the first resumption token as an argument and use it to make the next request. When a response is returned, it gets the new resumption again and use it to make another request and iterates the same way until resumption token returns an empty result.

Using the Simple HTML DOM parser and PHP again, this iteration can be achieved by first creating two constant variables that can later be assigned with whichever request type.

For example in the case of ListSets request the two variables will be:

```
$url='http://publications.theseus.fi/oai/request?verb=ListSets';
$nextset='http://publications.theseus.fi/oai/request?verb=ListSets&res
umptionToken=';
```

The first variable represents the first request URL. The first ListSet request should be made with this variable as follow.

```
    <?php
         require 'simple_html_dom.php';
         $firstPage=file_get_html($url);
     ?>
```

The response from this request will return the first 100 list of universities and departments and a resumption token for the next page. By scraping this resumption token, a new variable can be created in preparation to make the next request.

```
$first_RToken=$firstPage->find('resumptionToken',0)->plaintext;
```

By concatenating the new resumption token and the second constant variable (`$nextset`), the next request URL can be constructed as follow.

```
$nextPage=$nextset.$first_RToken ;
```

At this point, setting up a generic *function* to continue the same process until there is no more resumption token value is possible. This *function* will take `$first_RToken` (defined outside of the *function*) as an argument and iterates through each section by following the steps listed below

- First use a harvesting techniques to get the resumption token from a section
- Append this resumption token to the variable `$nextset` to make a new request URL `$nextPage`
- Make a new request with the formed new URL (`$nextPage`)
- Get the new resumption token from the new response and re-assign the first resumption token with the new value

- Finally, check to see if the reassigned resumption token is empty or not. If it is empty the iterations stops and that will be the end of the set list. Otherwise, the function calls itself again to get the complete list of request URLs.

Parsing each element in the response XML can be done by including scripts inside this generic *function* while it is iterating through each set list section.

When making a ListSets request, the goal is to get the list of

- Department identifiers and their correspond name
- University identifiers and their correspond name

The response from ListSets request will return these values in a structured form, shown in listing 2 below, for an easy harvesting.

```
<request verb="ListSets">
http://publications.theseus.fi/oai/request
</request>
      <ListSets>
        <set>
            <setSpec>com_10024_1</setSpec>
            <setName>Seinäjoen ammattikorkeakoulu</setName>
        </set>
        <set>
            <setSpec>com_10024_4</setSpec>
            <setName>Arcada - Nylands svenska yrkeshögskola</setName>
        </set>
```

Listing 2. Partial ListSets XML response showing fetched results

A set in OAI-PMH is used for grouping items for the purpose of selective harvesting. setSpec inside each set holds a unique identifier for the particular set it is in, and must be unique for each set in the repository. setName is a short descriptive string that is used for naming the set.

Theseus organizes universities of applied sciences and departments into sets. A unique identifier inside setSpec is used to identify each school and department in Theseus. The names of universities and their departments are kept within the setName. Since setName in OAI-PMH does not have to be unique the same department name from multiple universities of applied sciences is used together with setSpec that uniquely identifies them.

The first noticeable concern when parsing a ListSets request in Theseus is the fact that the XML response has a list containing mix of universities and departments. To separately parse universities of applied sciences and departments a PHP *string function*, *strops()*, that checks whether the setName tag has a string “ammattikorkeakoulu” or in two other cases “yrkeshögskola” can be used.

To parse setSpecs and setNames representing each university and department from the complete list, the generic *function* that was created earlier can be used. To do this, first make the *function* to return every ListSet request URL and store them in an *array*. By packing this *array* in a variable, `$requestURL,` then use a *foreach* construct to get setSpec and setName as follow.

```
foreach($requestURL->find('set') as $element) {

            foreach($element -> find('setSpec') as $id) {
                    $item_id=$id ->plaintext;
                    }
            foreach($element -> find('setName') as $name){
                    $item_name=$name ->plaintext;
                    }
}
```

The same process can be used to acquire all required metadata for our web portal. Continuing this manner, responses from “ListRecords” request to get metadata about each thesis document in Theseus can be harvested. As discussed earlier, it is good to remember that Theseus OAI-PMH requires the use of “metadataPrefix” when making a ListRecords request.

A proper ListRecord request with only the “metadataPrefix” argument will return all thesis documents on the repository. To limit the returned responses based on department or university, an additional argument, setSpecs, is required. In such cases, appending set identifiers (setSpecs) of a department or university to the request URL will result a response containing thesis documents from that department or university only.

For example, to get the metadata of thesis documents from the department of media engineering in Metropolia (setSpec = col_10024_245), the following request URL should be used.

http://publications.theseus.fi/oai/request?verb=ListRecords&metadataPrefix=kk&set=col_10024_245

This is an important concept in order to understand the processes used in parsing metadata of thesis documents. The response from the ListRecords request is organized in a different XML structure, shown in listing 3 below, than what was shown for ListSets request in listing 2.

```
<request verb="ListRecords" metadataPrefix="kk" set="col_10024_245">
http://publications.theseus.fi/oai/request
</request>
    <ListRecords>
      <record>
            <header>
                <identifier>oai:www.theseus.fi:10024/1374</identifier>
                <datestamp>2013-07-10T07:07:22Z</datestamp>
                <setSpec>com_10024_6</setSpec>
                <setSpec>col_10024_245</setSpec>
            </header>
            <metadata><kk:metadata        xmlns:kk="http://example.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:doc="http://www.lyncode.com/xoai">
<kk:identifier type="handle" value="10024/1374"/>
<kk:link href="http://www.theseus.fi/handle/10024/1374"/>
<kk:field  schema="dc"  element="contributor"  qualifier="author"  lan-
guage="none" value="Guo, Jun"/>
<kk:field   schema="dc"   element="identifier"   qualifier="uri"   lan-
guage="none" value="URN:NBN:fi:amk-200812124410"/>
```

Listing 3. ListRecords XML response of a single thesis document

A record inside the ListRecords tag in the above XML represents a single thesis document. It contains different fields to expose different types of information about a thesis document for harvesters including title, author, university, department, keywords, abstract, language and more. It is also uniquely identified in the repository by its identifier in the header section. Making a ListRecords request to the repository will return a list of records of each thesis document in Theseus.

As mentioned earlier, Theseus had 84,391 thesis documents at the time of this writing. When a list request is made, the data source returns 100 records per each single request. The rest of the records are sectioned using a resumption token in a manner shown previously for the ListSets request which makes it convenient to use the same parsing techniques that was used then. In fact, this is the reason why implementing the use of a generic code was important. By slightly modifying the generic *function* elements, specifically the values of the constant variables $url and $nextset, gathering request URLs for ListRecords requests can be done the same way.

The problem here is that there is a larger number of list sections to accommodate the amount of thesis documents. For this reason, the *function* needs to iterate a lot more time to get the complete list of request URLs causing max_execution_time error. Also, storing each one of them in an array for parsing purpose will causes memory_limit issues in PHP. One way to address these issues is increasing the memory limit and extending the maximum execution time in the php.ini file and restarting the server. It is also possible to add the following two lines of codes in the PHP script to avoid this problem.

```
ini_set('max_execution_time', 600);
ini_set('memory_limit', '-1');
```

Setting this issue aside, parsing thesis metadata from each section can be done by making ListRecords requests and repeating the same process that was done for ListSets requests above. From the header of each record, the setSpec of departments and setSpec of universities which the records belongs to can be parsed using PHP's *foreach* construct as shown below.

```
foreach ($requestUrl->find('record') as $element) {

$department_id= $element -> find('setSpec')[0]->plaintext;
        $uas_id= $element -> find('setSpec')[1]->plaintext;
}
```

To get other metadata of each a thesis document in the record, a careful analysis on how the metadata format of the XML response is structured is important. Now, as can be seen from the XML structure in Figure 12 above, the KK metadata format consists each information about the record as an attribute in its tag <*kk:field*>. The information of interest is contained inside the attribute element “`value`” of each <*kk:field*>. But since each <*kk:field*> has the attribute “`value`”, we have to find another way to identify what the “`value`”s stand for in each <kk:field>. This can be done by using the other attributes inside each <*kk:field*>.

```
<kk:field schema="dc" element="contributor" qualifier="author"
language="none" value="Guo, Jun"/>
```

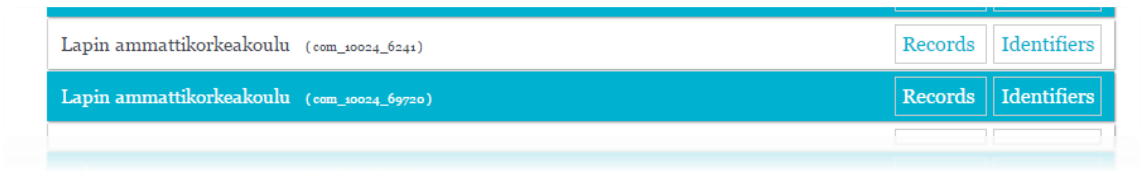
For example, the name of the author in the above <kk:field> tag (‘Guo, Jun’), can be parsed using the combination of “`element`”, “`qualifier`” and “`value`” attributes in a nested *if statement* inside a PHP code as follow.

```
        foreach($record -> find('kk:field') as $kk_tag) {
          $kk_attribute = $kk_tag -> element;

          if($kk_attribute =='contributor') {
            $qualifier_attr = $kk_tag -> qualifier;
            if($qualifier_attr == 'author') {
              $author = $kk_tag -> value;
            }
          }
```

The same method was repeated and used to parse and gather other metadata information of each thesis document.

There were numerous drawbacks when the parsing process was applied on Theseus. Error responses due to a requests to non-existent data or duplicate entries were causing the parsing process to pause multiple times which usually required the restart of the whole iteration process.



Lapin ammattikorkeakoulu (com_10024_6241) Records Identifiers

Lapin ammattikorkeakoulu (com_10024_69720) Records Identifiers

Figure 10. Duplicate entry with two different set specs

5 Storing Parsed Data in a MYSQL Database

After successfully completing the parsing process, the next step was to store relevant parsed information to a separate database system on a MYSQL server. Storing parsed metadata to a separate database was necessary so that the web portal can be fast and independent of Theseus's data provider. Deciding which metadata to store depends on the user requirement of the web portal and how the stored data is utilized. This Web portal was built to help users see graphical representation of of the following things.

- How the number of thesis documents published by universities of applied sciences have increased/decreased over the years
- A comparison between departments in a university based on the number of papers they have in the repository
- Trending keywords used by students in different departments and universities of applied sciences in any academic year

Additionally, other information about thesis documents such as author, title and number of pages of a thesis document in any given year, university and department should be graphically represented by the web portal. In order for these functionalities to be possible, metadata from each university, department and thesis document were stored separately in their own table. Table 2 below summarizes the metadata that was harvested and stored

Table 2. Summary of required metadata

Metadata from University of applied sciences	**Metadata from departments**	**Metadata from thesis docs**
identifier (setSpec)	identifier (setSpec)	Thesis Identifier
name	name	Author
ListSets Request URLs	ListSets Request URLs	Title
Total Number of papers	Total Number of papers	GetRecord URL
-	University identifier	Department identifier
-	-	University identifier
-	-	Keywords
-	-	Subjects (official keywords)
-	-	Number of pages
-	-	year
-	-	Language

With this information at hand, it is now possible to start working on the relational model of the database.

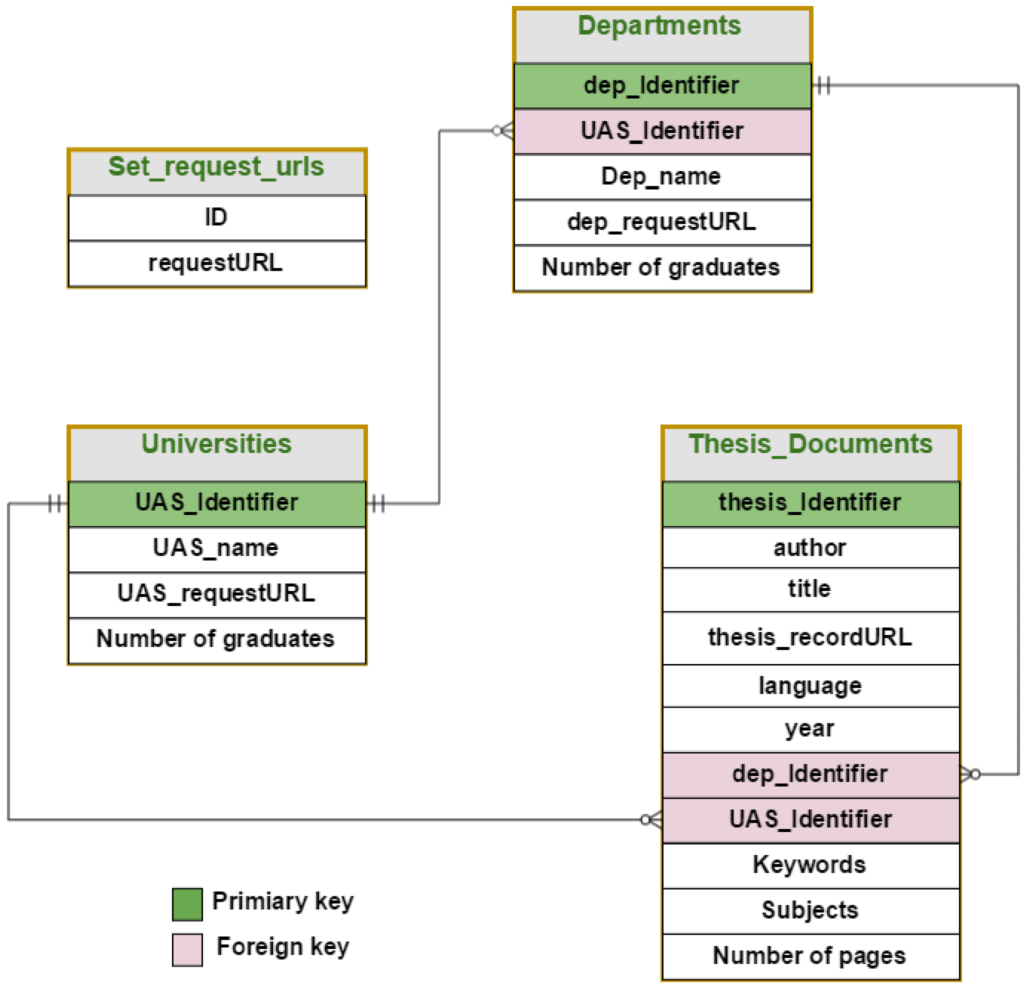


Figure 11. ER model of tables that was mapped in the database

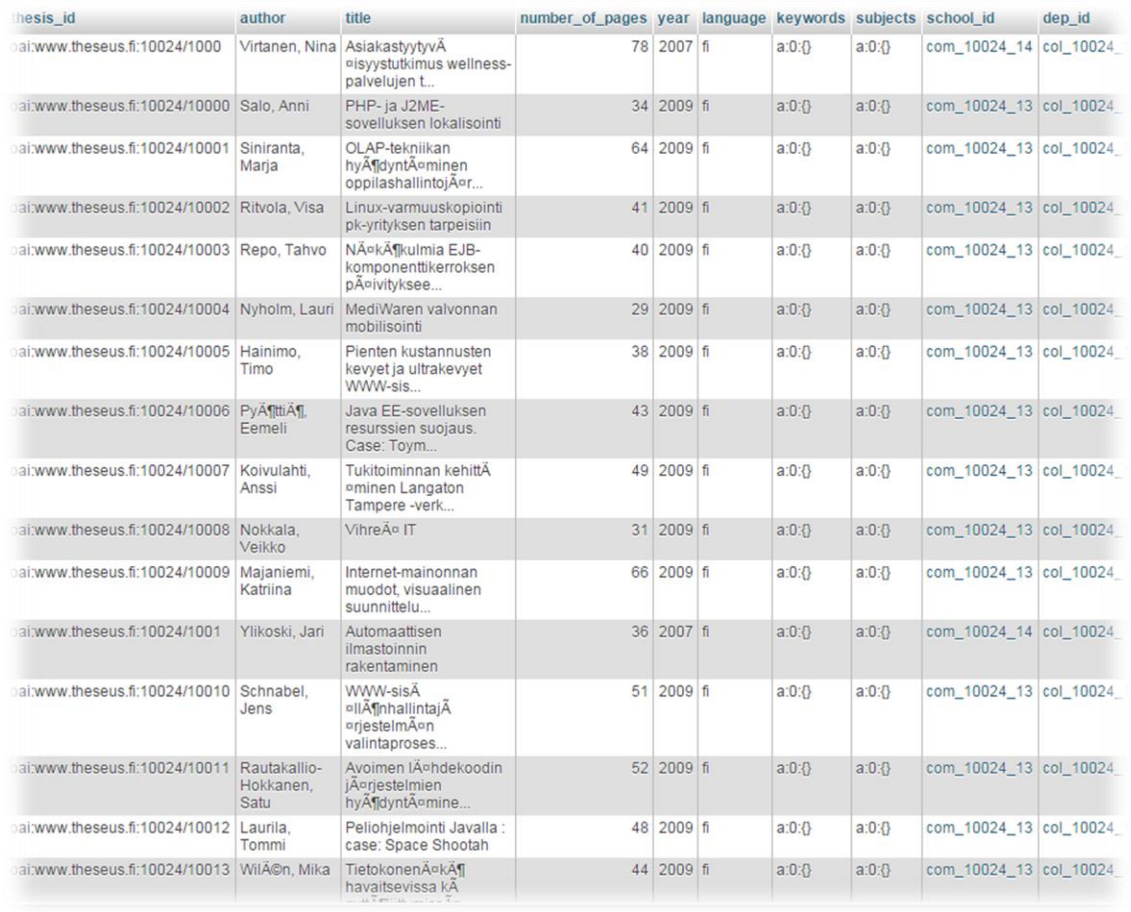
Based on the relational model design (see figure 11), a MYSQL database that has four tables was first created using the phpMyAdmin interface and then the tables were populated with the parsed data. The four tables have their own particular benefits in relation to each other.

Set_request_URL: This is a table to store only the request URLs of each set list section (list of schools and departments). The generic function devised earlier, is used to populate this table.

Universities: This table stores metadata about each university. Populating this table was done by using the first request URL in the set_request_URL table to make ListSets request. This is because all of the university are returned in the first ListSets request.

Departments: this table stores metadata about each department. By using all entities of the set_request_URL table, multiple ListSets requests were made to gather each department in each list section. Sets of schools gathered from this request were manually deleted from this table.

Thesis_documents: This table has metadata of every thesis documents as can be seen from figure 12. It can be regarded as the most important table for the Web portal project.



thesis_id	author	title	number_of_pages	year	language	keywords	subjects	school_id	dep_id
oai:www.theseus.fi:10024/1000	Virtanen, Nina	AsiakastyytyvÄ ¤isyystutkimus wellness-palvelujen t...	78	2007	fi	a:0:{}	a:0:{}	com_10024_14	col_10024_
oai:www.theseus.fi:10024/10000	Salo, Anni	PHP- ja J2ME-sovelluksen lokalisointi	34	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10001	Siniranta, Marja	OLAP-tekniikan hyÃ¶dyntÃ¤minen oppilashallintojÃ¤r...	64	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10002	Ritvola, Visa	Linux-varmuuskopiointi pk-yrityksen tarpeisiin	41	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10003	Repo, Tahvo	NÃ¤kÃ¶kulmia EJB-komponenttikerroksen pÃ¤ivityksee...	40	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10004	Nyholm, Lauri	MediWaren valvonnan mobilisointi	29	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10005	Hainimo, Timo	Pienten kustannusten kevyet ja ultrakevyet WWW-sis...	38	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10006	PyÃ¶ttiÃ¶, Eemeli	Java EE-sovelluksen resurssien suojaus. Case: Toym...	43	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10007	Koivulahti, Anssi	Tukitoiminnan kehittÃ ¤minen Langaton Tampere -verk...	49	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10008	Nokkala, Veikko	VihreÃ¤ IT	31	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10009	Majaniemi, Katriina	Internet-mainonnan muodot, visuaalinen suunnittelu...	66	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/1001	Ylikoski, Jari	Automaattisen ilmastoinnin rakentaminen	36	2007	fi	a:0:{}	a:0:{}	com_10024_14	col_10024_
oai:www.theseus.fi:10024/10010	Schnabel, Jens	WWW-sisÃ ¤llÃ¶nhallintajÃ ¤rjestelmÃ¤n valintaproses...	51	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10011	Rautakallio-Hokkanen, Satu	Avoimen lÃ¤hdekoodin jÃ¤rjestelmien hyÃ¶dyntÃ¤mine...	52	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10012	Laurila, Tommi	Peliohjelmointi Javalla : case: Space Shootah	48	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_
oai:www.theseus.fi:10024/10013	WilÃ©n, Mika	TietokonenÃ¤kÃ¶ havaitsevissa kÃ	44	2009	fi	a:0:{}	a:0:{}	com_10024_13	col_10024_

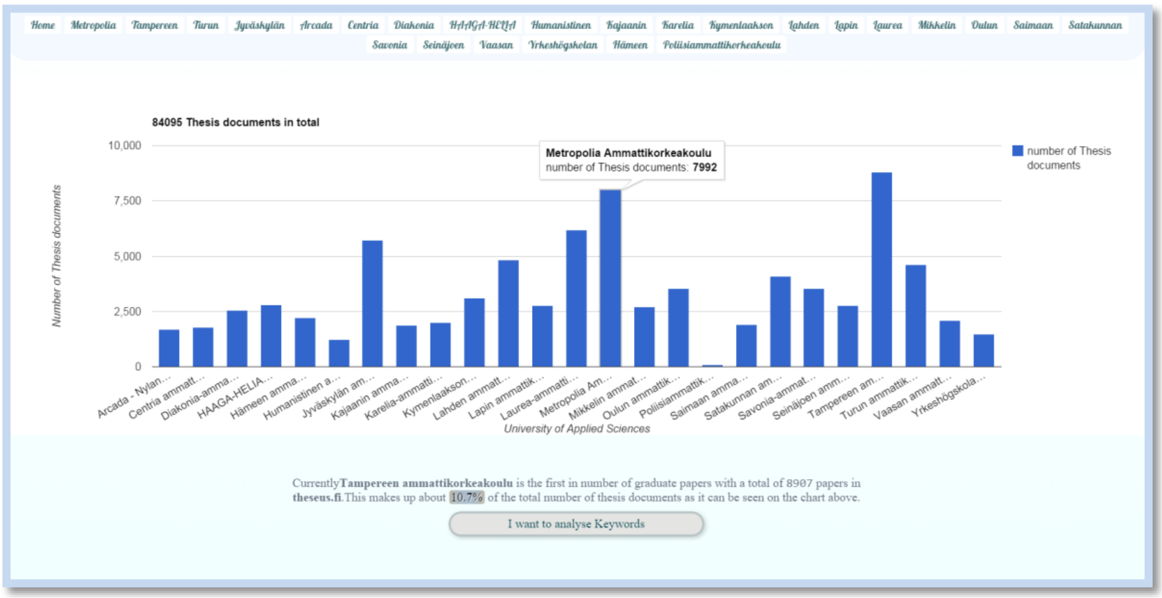
Figure 12.Partial list of thesis metadata in *Thesis_documents* table

At this stage, all the required information to build the web portal is stored and ready for fetching and display. This paper does not discuss how the web portal was built. However, main functionalities, benefits and results achieved are discussed in the next chapter.

6 Project Results in a Nutshell

So far this thesis has presented the two critical stages in the development of the technical project. First, how harvesting techniques were applied on Theseus was shown and then the thesis has discussed how this harvested data was stored into a separate MYSQL database. The third stage in the development process was bringing the stored data and using it to build a web portal that adds visualization to it so users can easily digest and observe interesting patterns. Although the thesis does not discuss how processing the stored data for visualization was implemented, this chapter intends to describe the main functionalities of the end product.

The built Web portal aims to give better insights on the contribution of each university to Theseus on its front page.



Currently**Tampereen ammattikorkeakoulu** is the first in number of graduate papers with a total of 8907 papers in **theseus.fi**.This makes up about 10.7% of the total number of thesis documents as it can be seen on the chart above.

Figure 13. Landing page of the Web portal

On this landing page (see figure 13 above), the portal shows overall statistics that compares each university of applied sciences based on the amount of thesis documents they have published on Theseus. The graph is friendly and easy to understand and it attempts to create the impression to a visitor that the portal is mostly all about statistical visualization of data within Theseus.

There is also a fixed navigation bar for users to navigate and see statistical data about a university of their choice. The portal is divided into sections using a jQuery plugin named fullPage.js. Each section contains statistical data for one university of applied science and is reachable by links on the navigation bar. Furthermore, the sections enclose landscape sliders to divide more statistical information about the university represented by them.

At the web portals current stage of development, information displayed in each section include graphical depiction for the following details.

- The number of thesis documents versus publication year
- Departments and their respective number of thesis documents (see figure 14)
- Comparing thesis documents based on number of pages

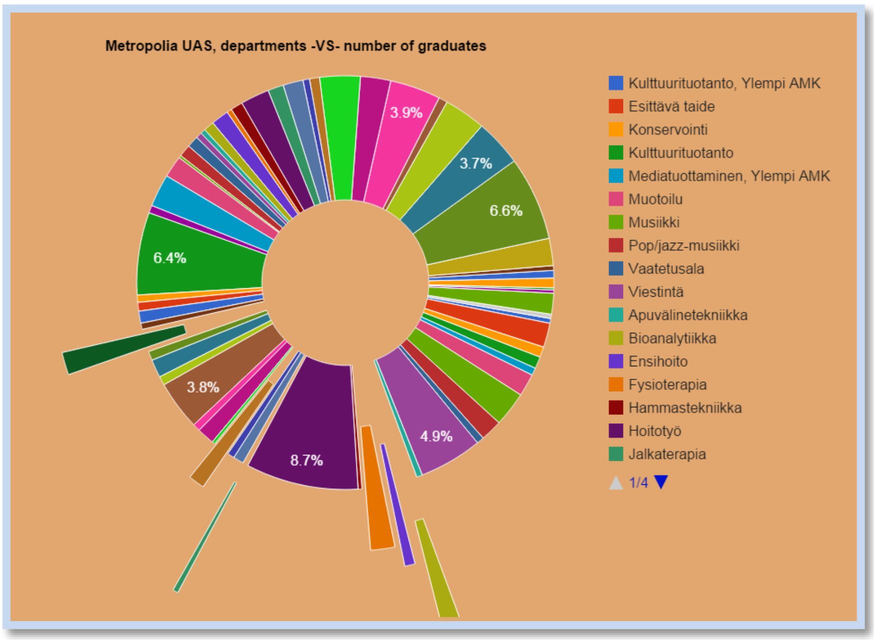


Figure 14. Departments and their respective number of thesis documents

The other main feature of the web portal is the keyword analyser that enables users to explore keywords used by students. By selecting year, university and department, users are able to filter and see keywords based on their selection. Moreover, users can also compare universities using another feature in the web portal that generates the combination graphs from two chosen universities. (More screen caption images from the web portal are included in Appendix 1 and 2.)

7 Final Thoughts and Discussions

Working on this project has required several revisions of programming courses to be made together with learning new concepts and programing techniques. I was out for a challenge from the start and I have gotten exactly what I needed. I think I can safely say that this was the toughest challenge and journey I took in my academic journey. Numerous excitement periods about discovering new concepts and sleepless nights trying to implement previously gained knowledge have made the process quite overwhelming. In the end, it has all paid off and the result achieve was self-satisfying.

There were three phases of work involved in achieving the final Web portal. First, a data from the concerned digital library had to be selectively harvested to match the desired content of the Web portal, and then a database system was designed and configured to store the harvested data. Finally, different approaches and techniques were used to bring the data and its components together and deliver the Web portal for consumption.

The works in the first stage started out by answering some questions beforehand and making analysis on Theseus itself.

- What kind of data can be extracted from Theseus?
- Which of this data is best fit in the Web portal?
- Why is the chosen data interesting enough to include it?
- Who would be interested in it?
- Does it really make a point to include it?

To answer these questions, extensive discussions and arguments were made with experts and peers. Afterwards, newly discovered techniques including web harvesting/scraping, developing a web bot to do things automatically and also other previous knowledge such as how to make a uniform request-response cycle between servers, utilizing PHP built-in functions and XML parsing libraries was applied to gathered contents of Theseus that best match the purpose of the Web portal.

The second phase was rather easier and was not in the core goals of the project. However, hours were spent figuring out the best practice in MYSQL to create the tables and their entities so the gathered data can be stored for easy and faster access.

At the end, by carefully addressing usability and interface concerns to present the stored data, the Web portal was built. Despite some limitations, the product interprets the stored data and provides users a visualization platform full of statistical charts and figures. The benefits from using JQuery plugins and the Google Charts API have also eased the work in coding some of the portal’s functionalities. During the process of development knowledge of HTML 5, CSS and procedural PHP were exceedingly practiced.

Limitations and Future Developments

Even though the developed Web portal is fully functioning and ready for use, not all the data gathered from Theseus is put to use in the foreseen manner. For this reason, some limitations apply to the Web portal. These limitations include inability to compare two thesis documents based on their respective number of downloads, inability to show most popular or trending topics, and inability to show average page length per university.

I strongly believe that the concept of this product is interesting and important enough to continue its production and add more features to it. Whether it is for comparing thesis documents based on the number of downloads and using the information to motivate upcoming students or showing how many thesis documents from each university was produced every year compared to how many students were accepted as a first year student, the possibilities of the project are very vast and can be extended further. By being creative on using the statistical data, the web portal can be further developed and even influence decision making processes.

8 Conclusion

The purpose of this project was to harvest thesis metadata from Theseus's data-provider, store it separately and then later develop a Web portal that visualizes statistical facts about theses and publications from Finnish universities of applied sciences. Harvesting the metadata was carried out by making "harvesting requests" to Theseus's data provider that uses Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) to provide and deliver metadata of thesis documents in different metadata formats. The OAI-PMH has made this harvesting process possible by providing a simple, yet powerful framework. This thesis paper has introduced some technical ways to use the OAI-PMH for harvesting metadata from a digital repository using Theseus as an example repository.

The toughest challenge that was faced in achieving this goal was dealing with a large set of data. There were 84,391 thesis documents at the time when the project was carried out. This made harvesting and storing metadata of thesis documents a demanding task. Also, frequent need for modification of written codes to accommodate bad responses from the data provider's server was a time consuming task that seek patience. However, with continuous guidance and support from the instructor, a successful result was achieved.

It was quite a ride into the world of programing and web development world. As interesting as it is to visualize statistical thesis data, it was a surprise to find out that it was not implemented by anyone before. Thus, with a mission to implement it and show its relevance, this project was conducted. The result achieved was satisfying and the product can now be used as a tool that visualises contents of Theseus.fi to give an overall insight into each university's contribution.

References

1. Baeza-Yates R., Ribeiro-Nato B. Modern Information Retrieval: the Concepts and Technology behind Search. Second edition. Harlow, England: Addision Wesley; 2011.

2. Theseus. Theseus.fi: Open Repository of Universities of Applied Sciences [online]. Jyväskylä, Finland: AMKIT Consortium; 2015.
   URL: http://theseus.fi.
   Accessed 27 April 2015.

3. Malik S. Information Extraction Using Web Usage Mining, Web Scrapping and Semantic Annotation. New Delhi, India: Computational Intelligence and Communication Networks (CICN); 2011.

4. DuraSpace. DuraSpace wiki [online]. Sydney, Australia: Atlassian Corporation Pty Ltd; 2015.
   URL: https://wiki.duraspace.org/display/DSPACE/Home.
   Accessed 7 March 2015.

5. Carpenter Leona. Open Archives Forum. Basic OAI Concepts and Features [online]. Bath, United Kingdom: University of Bath; 2003.
   URL: https://www.oaforum.org/tutorial/english/page1.htm#section2.
   Accessed 7 April 2015.

6. Downes Stephen. Open Archives Initiative [online]. North Carolina, United States: University of North Carolina; 2003.
   URL: http://technologysource.org/article/226/.
   Accessed 17 April 2015.

7. Heery R. Review of Metadata Formats [online]. Bath, United Kingdom: University of Bath; 1998.
   URL: http://www.ukoln.ac.uk/metadata/review.html.
   Accessed 7 March 2015.

8. The Repositories Support Project. Harvesting Repository Data and OAI-PMH [online]
   URL: http://www.rsp.ac.uk/grow/registration/harvesting/.
   Accessed 14 March 2015.

9. Leona Carpenter. Open Archives Forum. Main Technical Ideas of OAI-PMH [online]. Bath, United Kingdom: University of Bath; 2003.
   URL: https://www.oaforum.org/tutorial/english/page3.htm.
   Accessed 22 March 2015.

10. Schrenk M. Webbots, Spiders, and Screen Scrapers: A Guide to Developing Internet Agents with PHP/CURL. 2nd ed. San Francisco, U.S.A: No Starch Press; 2012.

11. Michel J. P. Web Service APIs and libraries. USA: American Library Association; 2012.

12. Gray j., Chambers L., Bounegru L. The Data Journalism Handbook [online]. Newton, Massachusetts: O'Reilly Media: 2012.
URL: http://datajournalismhandbook.org/1.0/en/getting_data_3.html.
Accessed 18 April 2015.

13. Mitchell R. Instant Web Scraping with Java. Birmingham, United Kingdom: Packt Publishing Ltd; 2013.

14. Anderson S. M. Instant Simple Botting with PHP. Birmingham, United Kingdom: Packt Publishing Ltd; 2013.

15. Hunter D., Rafter J., Fawcett J. Beginning XML. River Street, USA: John Wiley & Sons; 2007.

16. Google Developers. Using Google Charts [online]. UAS: Google, Inc.; 2015.
URL: https://developers.google.com/chart/interactive/docs/index.
Accessed 4 February 2015.

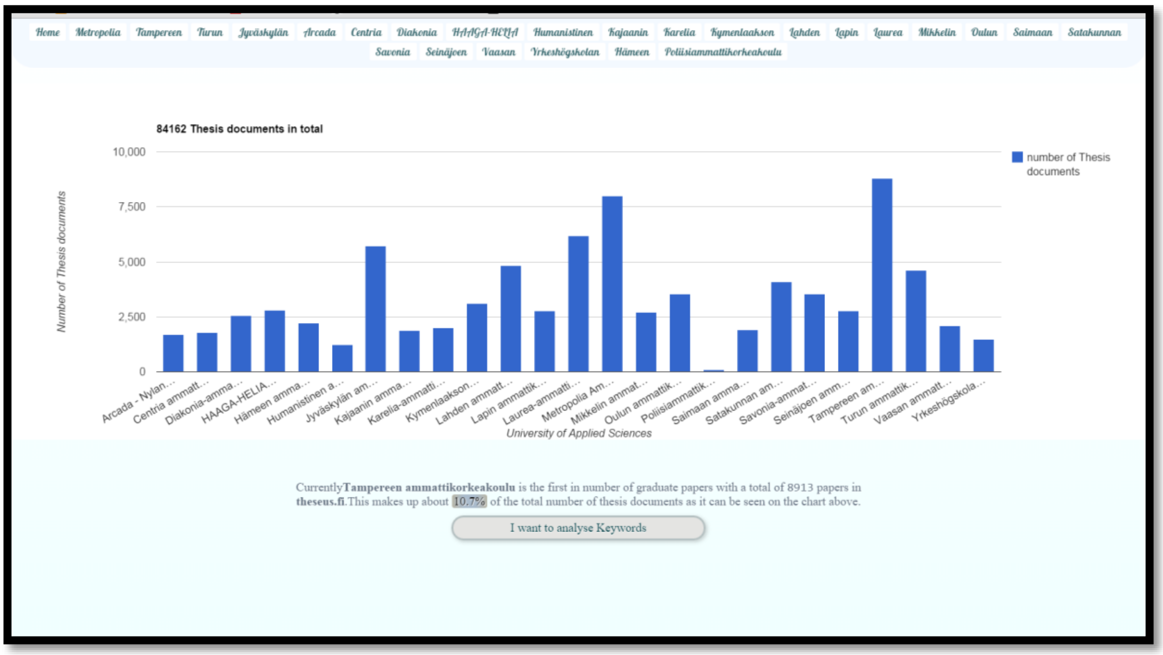
17. Marc J. Web application description language (WADL). Mountain View, CA, USA: Sun Microsystems, Inc.; 2006.

18. Carl Lagoze, Herbert Van de Sompel. Open Archives Initiative Protocol for Metadata Harvesting - v.2.0 [online]. New York, NY: Andrew W. Mellon Foundation;2008.
URL: http://www.openarchives.org/OAI/openarchivesprotocol.html#FlowControl.
Accessed 8 April 2015.

19. Carl Lagoze, Herbert Van de Sompel. Open Archives Initiative Protocol for Metadata Harvesting - v.2.0 [online]. New York, NY: Andrew W. Mellon Foundation;2008.
URL:
http://www.openarchives.org/OAI/openarchivesprotocol.html#MetadataNamespaces.
Accessed 8 April 2015.

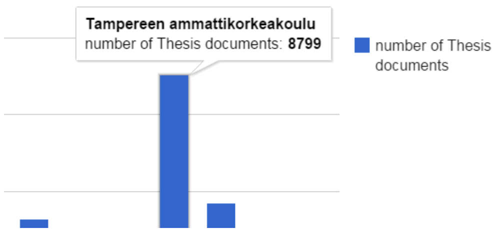
20. Carl Lagoze, Herbert Van de Sompel. Open Archives Initiative Protocol for Metadata Harvesting - v.2.0 [online]. New York, NY: Andrew W. Mellon Foundation;2008.
URL: http://www.openarchives.org/OAI/2.0/guidelines-harvester.htm#RunningAHarvester.
Accessed 18 April 2015.

Appendix 1. Overall statistics on university of applied sciences



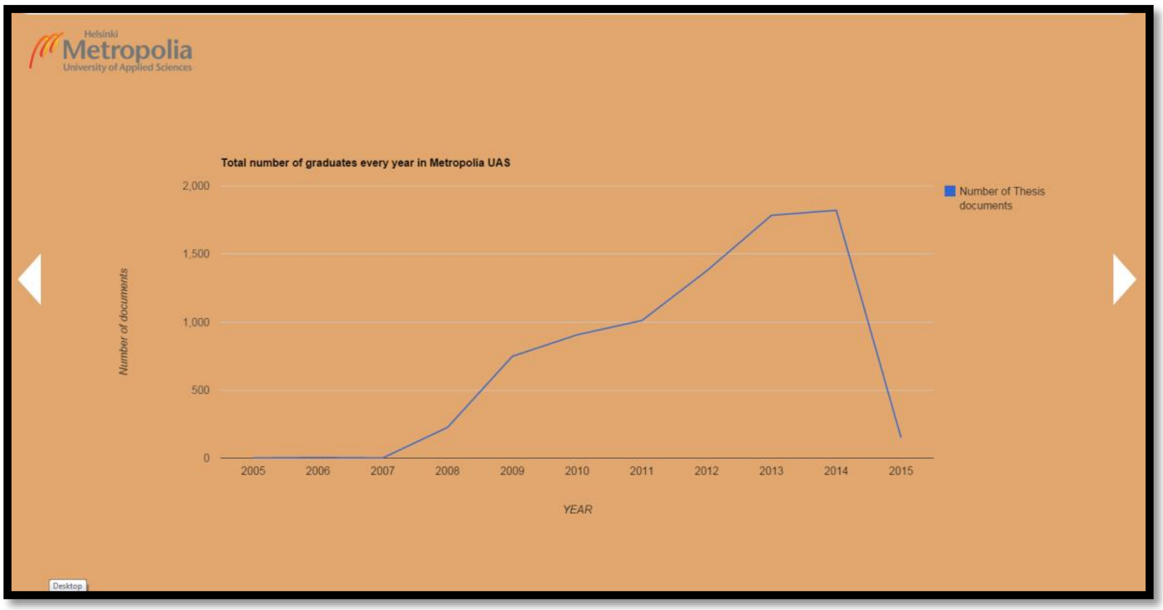
Currently**Tampereen ammattikorkeakoulu** is the first in number of graduate papers with a total of 8913 papers in **theseus.fi**.This makes up about 10.7% of the total number of thesis documents as it can be seen on the chart above.

The column chart, on the front page of the Web portal (shown in the above figure), is a vertical bar that shows the total number of thesis documents from each university of applied sciences in Finland. Each bar in the chart represents one university of applied sciences. The column chart also displays a tooltip when users hover over any of the bars in the chart (shown in the figure below). This tooltip shows the total number of thesis documents together with the name of the school represented by any of the bar selected by the user.

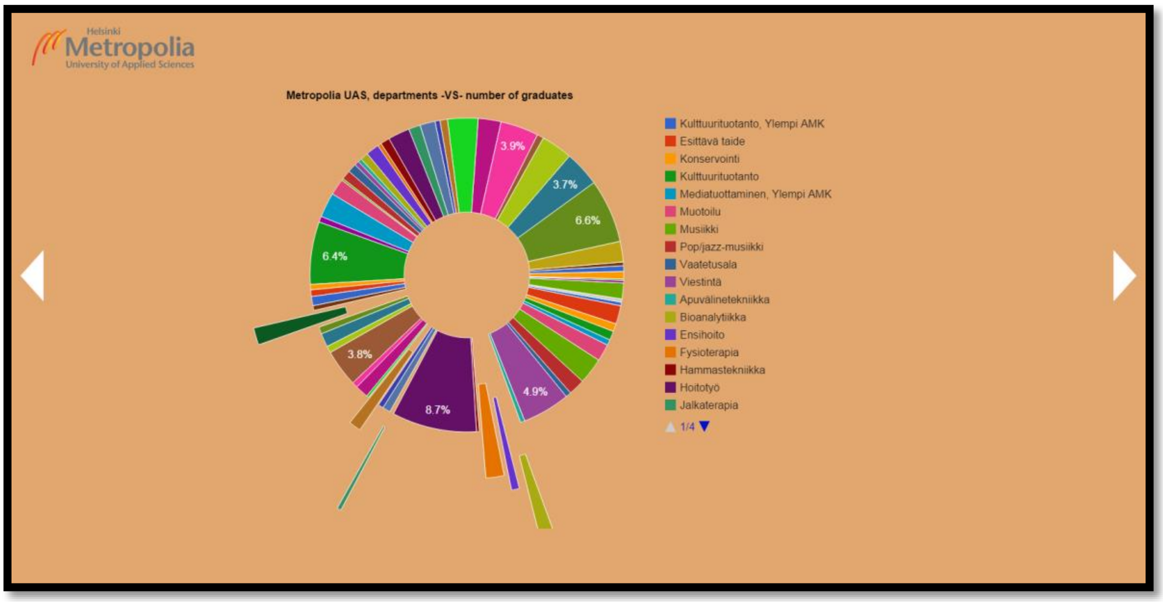

Tampereen ammattikorkeakoulu
number of Thesis documents: 8799
number of Thesis documents

From this chart, users can see the number of thesis documents in Theseus and how many are published by each school in comparison to other schools.

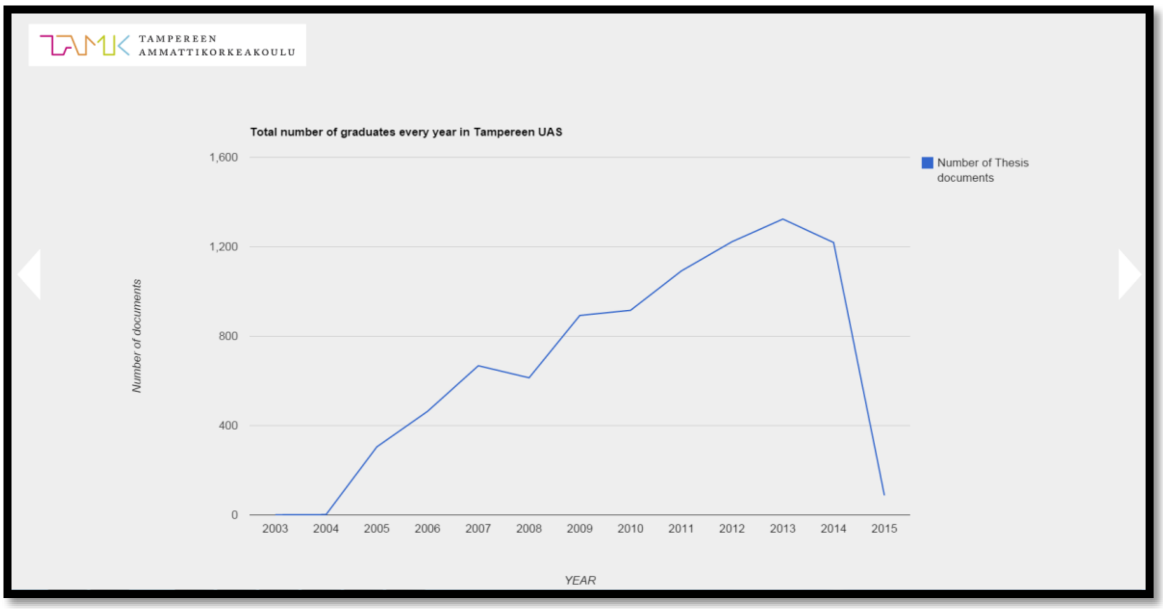
Appendix 2. Sample statistics on thesis documents, Metropolia UAS



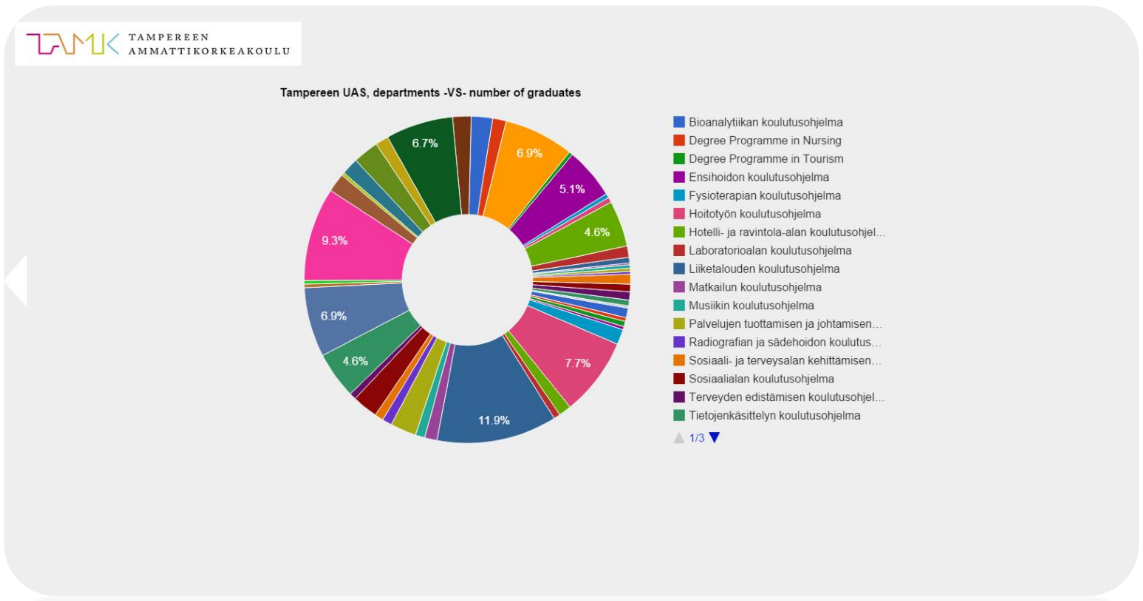
The above chart is generated by the Web portal. It shows the number of thesis documents being published by Metropolia university of applied sciences in every academic year. This chart also has a tooltip, which appears when hovering over the graph to show the year selected and the number of thesis documents published in that selected year. Departments in Metropolia UAS and the number of thesis documents published by them are also represented by a pie chart in comparison with each other. This is shown in the figure below.



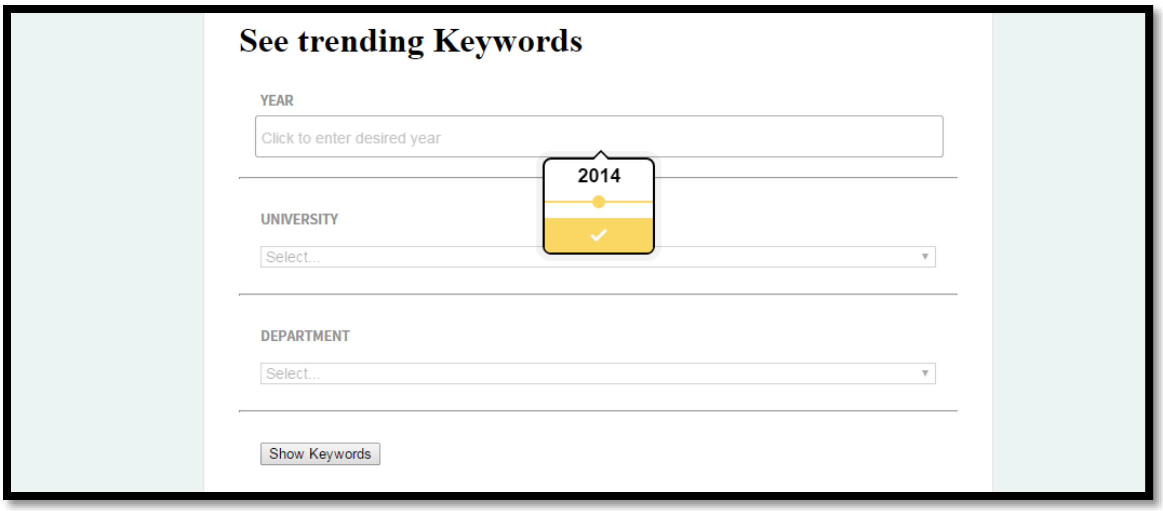
Appendix 3. Sample statistics on thesis documents, Tampere UAS



The same chart is generated for each university of applied sciences in Finland. This particular chart, in this appendix, represents a sample from Tampere university of applied sciences.



Appendix 4. Keywords filtering form and filtered results in partial



See trending Keywords

YEAR

Click to enter desired year

2014

UNIVERSITY

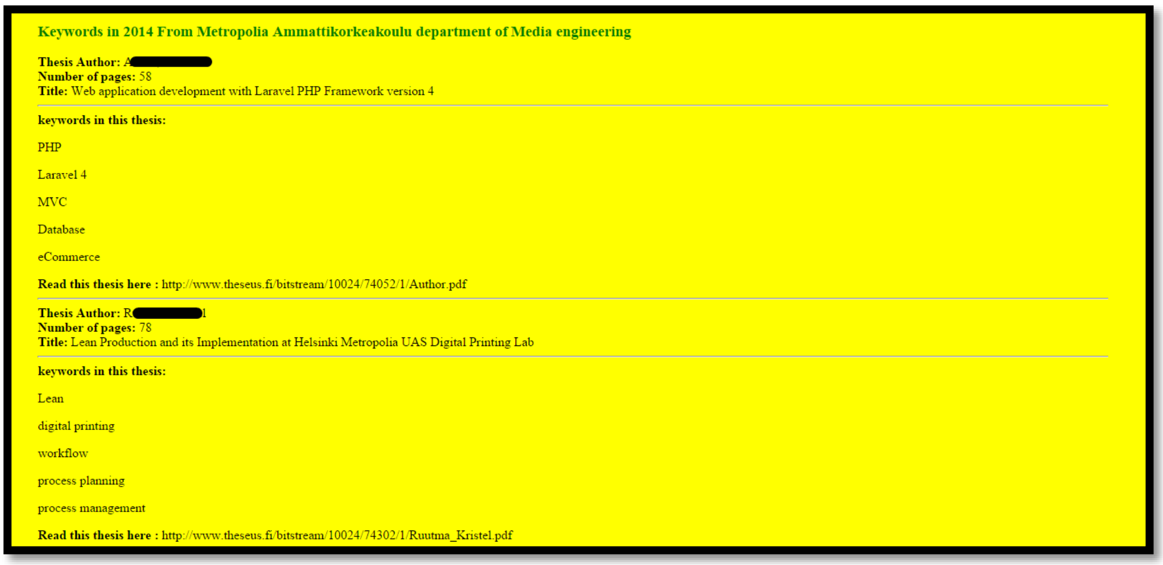
Select...

DEPARTMENT

Select...

Show Keywords

In order to see trending keywords used by students from universities of applied sciences in Finland in every academic year, users can fill a form, shown in the above figure, to filter keywords by year, university and department. Keywords used by students in the selected year from the selected university and department will be returned (see figure below) by the Web portal after the form is submitted.



Keywords in 2014 From Metropolia Ammattikorkeakoulu department of Media engineering

Thesis Author: A[redacted]
Number of pages: 58
Title: Web application development with Laravel PHP Framework version 4

keywords in this thesis:

PHP

Laravel 4

MVC

Database

eCommerce

Read this thesis here : http://www.theseus.fi/bitstream/10024/74052/1/Author.pdf

Thesis Author: R[redacted]l
Number of pages: 78
Title: Lean Production and its Implementation at Helsinki Metropolia UAS Digital Printing Lab

keywords in this thesis:

Lean

digital printing

workflow

process planning

process management

Read this thesis here : http://www.theseus.fi/bitstream/10024/74302/1/Ruutma_Kristel.pdf