arns
SPIKESEED

PILOT PROJECT:

PUBLICACCESS.EU

PROOF OF CONCEPT OF A SEARCH ENGINE FOR EU DOCUMENTS

FINAL REPORT

Customer	DIGIT/OP
Contract reference	STIS IV-000635
Public version	14/9/2017

CONTENTS

EXECUTIVE SUMMARY

PublicAccess.eu is a pilot project launched in 2014 by the European Parliament to facilitate online access to a wider range of unclassified documents held by EU institutions, agencies and bodies (hereinafter the 'EU institutions') and to increase the transparency of the EU decision-making process.

Publicly available documents from EU institutions are scattered across a large number of different databases and repositories (hereinafter 'document sources'). In order to find required information, citizens usually have to perform searches in each of them separately, which can be time-consuming and cumbersome. An insight into this fragmentation of EU information has been provided by one of the outputs of the PublicAccess.eu project: the study entitled 'Ensuring integrated access to all publicly available EU documents' (hereinafter the 'Study'). The Study has analysed in detail the data sources of 14 selected EU institutions and proposed four integrated access solutions, how the documents coming from these data sources could be accessed from a single place. The feasibility of a central search engine has been tested by creating a proof of concept prototype: the Search@PublicAccess.eu (hereinafter the 'PoC'). It has been developed in cooperation between the Publications Office, DG Informatics (hereinafter 'DIGIT') of the Commission and DIGIT's contractors.

The PoC has aimed for accessing — from a single interface — a sample of documents present in a preselected set of data sources analysed in the study:

- European Parliament's Register of Documents;
- Public register of Council documents;
- European Commission's Register of Commission documents;
- European Central Bank's website;
- Court of Justice of the European Union website;
- European Economic and Social Committee's website;
- European Research Council's website;
- The Body of Regulator for Electronic Communications website.

Of these data sources, the Publications Office had privileged access to the European Parliament's Register of Documents. The Register of Commission documents was also easily accessible thanks to a dedicated webpage made available by DIGIT ([1]).

The main principle on which the project has been built is the reuse of already existing data and thus avoiding transmissions of the documents from their original sources. The user interface that would be the outcome of the PoC, would offer the information about the available documents and the links to these documents, pointing to their original data source. Consequently, for each data source, a two-step procedure has been followed: firstly, the available information (metadata) and links to the documents have been identified and collected (harvested), and, secondly, all this data has been indexed in order to make it searchable.

([1]) This page is currently available to DIGIT because of the NextEuropa project.

Due to significant differences in technical concepts of the different data sources, harvesting of available documents and metadata could not be done as a one-off exercise but different technical approaches had to be applied, respecting the specific features of each data source. Therefore, the first added value of this exercise consists in identifying the best-fit solution for each of the analysed data sources.

For the indexation of the harvested documents and metadata, an already existing user interface of DIGIT has been reused, which was one of the reasons why the PoC was developed in cooperation with this directorate. In addition, this interface — DIGIT's Corporate Search API ([2]) — has supported the use of an open source search engine Elasticsearch, selected for the PoC. Thus, by joining the cooperation on the PoC, DIGIT could test in practice the cloud-based hosting, as well as full test scenario for the Elasticsearch integration. Thanks to this win–win situation, the PoC could take advantage from a complete, Google-like user interface, including advanced functionalities such as faceted search and filtering ([3]).

In addition to the actual work of building the PoC, it was decided during the implementation phase that in order to get a better picture about the feasibility of the integrated access in the full scope of the Study, the remaining data sources, not covered by the PoC, should be at least analysed. By adopting this approach, this more complete perspective has enabled to make an overall estimate of the necessary resources, if the PoC was going to be changed into developing of the fully fledged production application. This estimate has been created for three scenarios, following two different technical approaches: one scenario has been presented under the same technical solution that was used for building the PoC, while two scenarios have been proposed under the infrastructure of the OP Portal search of the Publications Office.

However, it has to be kept in mind, that a mere availability of necessary resources would not be sufficient. The experience on this PoC has led to many lessons learnt showing that in order to build a reliable application, there is a strong need for active cooperation between the developer of the application and the EU institutions whose data sources would be covered. The main constraints have been identified as follows:

- There are no common cross-institution 'vocabularies' used for the document metadata. Thus, the metadata to be used and its mapping towards the common search needs must almost always be defined specifically per institution;
- Harvesting from the data sources which do not offer any harvesting facilities (e.g. web services) is laborious and can be unreliable.

Ideally, the concerned EU institutions, as the data source owners, should be involved already during the inception phase of the project, in order to help prepare their data sources for a more efficient connection to the search application.

([2]) API: Application Programming Interface — a set of routines available for programmatic calls making it possible to build software made of components.
([3]) The filters available for the PoC have been: data source, year, language of the document, date of document. Concerning the date of documents, it should be noted that since the sets of metadata used by the institutions are not uniform, some compromises had to be made. Namely, when more dates were available, the date metadata has been chosen for the one that seemed to be the most relevant in each case. Also, for some data sources the date metadata field could take only a year as value; as a result all documents from the same year share the same date.

1 INTRODUCTION

1.1 HISTORY AND CONTEXT

PublicAccess.eu ([4]) is a pilot project launched in 2014 by the European Parliament to facilitate and improve online access to a wider range of unclassified documents held by the EU institutions, agencies and bodies (hereinafter the 'EU institutions') and to increase the transparency of the EU decision-making process. The main objectives of the pilot project are to identify and bring together the relevant unclassified documents which can be made available, as well as to structure them in a way that ensures interoperability and linking. The project has been managed by the Publications Office of the European Union (hereinafter 'OP') together with the Secretariat-General of the Commission (hereinafter 'SG') and DG Informatics of the Commission (hereinafter 'DIGIT'), under the lead of OP.

Currently, public documents from EU institutions are scattered across many repositories, and they are made available to the citizens via multiple public registers and/or websites. Each EU institution maintains their own repositories and outlets, making it impossible for the citizens to access a comprehensive range of EU documents on a given topic from a single website. Instead, the citizens need to search in each of the EU institutions websites separately. Even the major search engines cannot access all the registers, and thus the overall scope of the documents can remain hidden.

Among the main achievements of the PublicAccess.eu project is the elaboration of a study entitled 'Ensuring integrated access to all publicly available EU documents' (hereinafter the 'Study') ([5]) which has analysed the available data sources (databases and websites) of 14 different EU institutions:

- European Parliament (EP):
  - Register of Documents;
  - Legislative Observatory;
  - IPEX;
- Council of EU and European Council (Council):
  - Public register of Council documents;
  - Central Archives Search Engine of the Council of the European Union;
  - Council database of agreements and conventions;
- European Commission (EC):
  - Register of Commission documents;
  - Comitology register;
  - Register of Commission expert groups.
- European Central Bank's (ECB) website;
- Court of Justice of the European Union (CJEU) website;

([4]) http://publications.europa.eu/en/web/public-access/home
([5]) http://publications.europa.eu/en/web/public-access/study

- European Court of Auditors (ECA) website;
- European Economic and Social Committee (EESC) website;
- European Committee of the Regions (CoR) website;
- European Ombudsman (EO) website;
- Publication Office of the European Union (OP) database, feeding the following websites:
  - EUR-Lex;
  - TED;
- Education, Audiovisual and Culture Executive Agency's (EACEA) website;
- European Research Council's (ERC) website;
- The Body of European Regulators for Electronic Communications' (BEREC) website;
- The European Union Intellectual Property Office (EUIPO);

The Study describes these data sources, analyses them in terms of the metadata compatibility of their documents and defines the extent of metadata harmonisation needed for ensuring their integration. At the end, the Study proposes and compares four alternatives for potential integrated access solutions. OP has decided to test the proposal for development of a central search engine by means of a proof of concept (hereinafter the 'PoC').

The goal of the PoC was to test the feasibility of developing a unified search interface, which would enable searching for unclassified documents in multiple data sources of different EU institutions but display the results in a single outlet. The guiding principle consisted in reusing the already public data and not creating a single repository for all EU documents. Therefore, the information from the indexed data was to be served only in the presentation layer, while the whole documents' content would remain at the source owner's side. This implied that all requests to access a document would be redirected to the data source owner which meant that availability of the content would depend on stability of the document sources infrastructure. That is why it has been suggested in the Study that the indexes of the search interface would be supplied by the Application Programming Interfaces ([6]) (hereinafter the 'API') implemented on top of each document source. However, as the implementation of the APIs required an active participation of the concerned EU institutions, which was not always possible, alternative ways had to be explored.

1.2 PURPOSE AND THE INTENDED AUDIENCE

The purpose of this document is to describe the outcomes of the respective analyses and developments and summarise the experience and lessons learnt during the implementation of the PoC. In addition, suggestions are made for future upcoming tasks, including an estimate of necessary resources, should the PoC turn into a fully-fledged production service in the future.

([6]) https://en.wikipedia.org/wiki/Application_programming_interface

This document should serve all colleagues in the EU institutions interested in implementing a unified endpoint for dissemination of unclassified EU documents. The document will also be publicly available on the PublicAccess.eu website.

1.3 STRUCTURE OF THE DOCUMENT

The document is organised as follows:

- Chapter 1, entitled ‘Introduction’, presents the context of the PublicAccess.eu project within which the PoC has been developed.
- Chapter 2, entitled ‘Project stages’, summarises the main phases of the project.
- Chapter 3, entitled ‘Approaches and strategies’, explains respective methodologies and approaches used for harvesting, ingestion and presentation of the documents’ metadata.
- Chapter 4, entitled ‘Outcome of the Proof of concept’ presents the main modules of the PoC: harvesting, ingestion and user interface modules.
- Chapter 5, entitled ‘Going into production – roadmap and estimated resources’ describes the tasks that should be performed to transform the current PoC into a fully-fledged production application. It covers also a rough estimation of necessary resources.
- Chapter 6, entitled ‘Lessons learnt’, summarises the general experience acquired on the project, enumerating the identified risks and ways of improvement to mitigate these risks.

2 PROJECT STAGES

2.1 COMMUNICATION WITH THE EU INSTITUTIONS

In order to facilitate the collection of the wished information (documents and their associated metadata), OP and DIGIT had to first establish working relationships with the EU institutions' data source owners. In the cases where the data source owners cooperated actively with OP and DIGIT, a greater coverage and, in general, a higher quality of the information indexed has been reached.

The data sources owners of all 14 EU institutions listed under section 1.1 were informed about the development of the PoC and received the necessary information and technical details, with an invitation to participate in the project. All EU institutions expressed their interest in this initiative.

However, despite the initial interest, only some EU institutions chose to be actively involved in the project, such as the European Commission and the European Parliament who already had or were in the process of implementing services which could provide a more uniform, reliable and permanent channel to pull information from their data sources.

Concerning the scope of documents to be accessible by the search interface, some EU institutions chose to participate with a representative set of documents. For example, the European Parliament participated with four different types of documents dating from 2004 till 2017, while the European Commission provided access to all documents available on their public Register of Documents that had a PDF manifestation.

For the non-active participants, DIGIT with help of external entities (on the basis of the DIGIT's contracts ([7])) has endeavoured to collect as much of the publicly available information as possible.

Also, due to the limited time foreseen for the PoC exercise, not all 14 EU institutions covered by the Study (listed under section 1.1) were going to be indexed. Therefore, OP and DIGIT had to prioritise among these 14 EU institutions. Consequently, based on the above described level of participation, available services, as well as the diversity of their services and the volume of documents, the scope of the PoC has been reduced, compared to the scope of the Study, to indexation of the following data sources:

- European Parliament (EP):
  - Register of Documents;
- Council of EU and European Council (Council):
  - Public register of Council documents;
- European Commission (EC):
  - Register of Commission documents;
- European Central Bank's (ECB) website;

([7]) Mainly Arηs Spikeseed (hereinafter as 'Arηs'), who has also been in charge of writing this report.

- Court of Justice of the European Union (CJEU) website;
- European Economic and Social Committee (EESC) website;
- European Research Council's (ERC) website;
- The Body of European Regulators for Electronic Communications' (BEREC) website.

2.2 POC ELABORATION

Once the initial communication and the priorities were established, the focus of the project was to study the implementation of dedicated connectors for a subset of the selected data sources.

The goal of this phase was to design early solutions and strategies to harvest data from heterogeneous data sources. This allowed to understand the challenges at hand and to estimate the necessary effort and approaches foreseen for including other EU institutions.

From a high-level point of view, two categories of data source have been identified:

- The ones offering an interface to ease the harvesting of the documents (structured data);
- The ones solely offering their website as a dissemination medium (unstructured data), these websites having, in most cases, disparate structures.

Another aspect covered by this PoC was to identify the main constraints that have been encountered and to identify the possible ways to make improvements. It became clear that the level of participation by the data source owners will have an impact on the quantity of items indexed, as well as the quality of the metadata.

From a technical point of view, three modules were developed during this phase:

- Harvesting modules: to execute the actual harvesting from each data source and to call the ingestion module at the end;
- Ingestion module: to receive the harvested data and index it;
- User interface module: by using the indexed data, to offer the possibility for end-users to search for the documents of several institutions from the same search portal. This module has been hosted on the DIGIT's premises, accessible from a predefined list of URLs. The list of these URLs could be extended, if there is a demand for a broader testing of the search engine developed under the PoC.

2.3 REPORTING AND PRESENTATION OF THE RESULTS

The development of the PoC has been accompanied by numerous reports created in order to measure the progress of the PoC. At the end of the project, this final report has been drafted with the intention to present it to all interested stakeholders, as well as to publish it on the PublicAccess.eu website.

3 APPROACHES AND STRATEGIES

3.1 OVERVIEW

Due to the diversity of the available services, several different approaches have been considered and applied during the implementation of this project.

In principle, when an application interface was available at the data source's layer, this interface was exploited to establish a more reliable channel for harvesting the information. However, in most of the cases, no machine readability features existed.

This chapter will describe the main approaches and solutions which have been identified for the three main modules of the project: the harvesting, the ingestion and the user interface.

3.2 HARVESTING

3.2.1 APPROACHES CONSIDERED FOR THE INITIAL HARVESTING

3.2.1.1 MAIN SOLUTION — CALL TO AN ENDPOINT

Among the analysed data sources, some are exposing endpoints. These endpoints are of different formats (e.g. web service, SPARQL querying endpoint, search form).

This harvesting method is easier than the 'web scraping' method, described under section 3.2.1.3. Indeed, the documents metadata are already formatted and directly harvestable. It is also easier to filter the harvested documents (e.g. by modification date).

The steps are:

- Understand how the endpoint works (e.g. by reading the documentation provided by the data source owners);
- Call the endpoint ([8]) and get in response the documents metadata.

A drawback is that this approach fully relies on the endpoint implementation and on the data source owners in case of any issues.

In the scope of this PoC, the data sources of the following EU institutions have been harvested using a call to an endpoint:

  - Council of EU and European Council (Council);

([8]) API: Application Programming Interface — a set of routines available for programmatic calls making it possible to build software made of components.

- European Parliament (EP).

To conclude, this harvesting strategy is the recommended one. Indeed, the endpoints are built to be read by machines and thus propose standardised and well formatted metadata. Also, it is often possible to filter the data and to check for the data consistency.

3.2.1.2 SECOND SOLUTION — RSS SCRAPING

The RSS scraping aims at harvesting information from what is displayed on an RSS feed by parsing the HTML code of the page.

All data from an RSS feed are structured and sorted by date. This allows quick harvesting of the content. Most of the time, an RSS feed is also paginated. Therefore, multithreading the fetching is easily possible, thus increasing the speed of fetching.

The main disadvantage of this method is the lack of metadata. The RSS standard allows for a very basic overview of the content and all advanced metadata tend to be within the document itself. Therefore, a full analysis of the document is required before any ingestion. Also, the RSS feed usually lists new and modified documents but does not list documents that have been deleted.

In the scope of this PoC, the RSS scraping method has been used for:

- European Economic and Social Committee (EESC).

To conclude, RSS scraping provides for a good and automatically updated source of information to harvest, although without sufficient metadata coverage.

3.2.1.3 FALLBACK SOLUTION — WEB SCRAPING

'Web scraping' aims at harvesting information based on what is displayed on a website page by parsing the HTML code of the page.

Before starting the harvesting, the first required step is a deep analysis of the website structure in order to find out where the documents are and how their metadata are harvestable. While the 'web crawling' ([9]) mechanism would fetch the whole website content without business knowledge, the 'web scraping' targets specific pages, identified (by web crawling or otherwise) as containing the relevant information.

Once the URLs of all relevant pages have been identified (for this PoC, the pages containing the documents metadata), the next step is to fetch (download) the HTML content of these pages. For this, there are many methods available (every programming language has a way to develop such functionality). In the case of this project, Groovy scripts and pure JavaScript have proven to work equally well.

([9]) Web crawling: an internet bot that automatically browses the website's content. It is commonly used by search engines for indexation purpose.

Afterwards, several methods can be used for parsing the document's metadata from the HTML content.

In the scope of this PoC, it has been decided to use harvesting scripts to harvest the document metadata from the HTML code of the pages.

The first main disadvantage of this solution is that the parsing scripts are designed to harvest the content of a specific page, thus these scripts are strongly impacted by any update of the concerned website pages. Another main identified constraint is that the HTML pages are usually not designed to be read by automatic systems.

In the scope of this PoC, the data sources of the following EU institutions have been harvested using a 'web scraping' method:

- European Central Bank (ECB);
- Court of Justice of the European Union (CJEU);
- The Body of European Regulators for Electronic Communications (BEREC).

To conclude, with no active participation of the institution, this harvesting strategy should be considered as a fallback. However, it brings a lot of inconsistency and poor metadata quality due to the unstructured nature of the data, the entire mechanism being sensitive to layout changes of the target website and the possibility of being blacklisted.

3.2.1.4 MIXED APPROACH — WEB SCRAPING WITH ACTIVE PARTICIPATION OF THE TARGETED INSTITUTION

The 'web scraping', as described above, aims at harvesting information based on what is displayed on a website page, by parsing the HTML code of the page. For this case, a special page can be prepared, often easily, listing all documents to be harvested. So, even if it is a conventional HTML page, it is prepared with the target of automatic consumption by another system.

This saves the effort of analysis of the target website, making it possible to prepare pages listing documents in a format suitable for automatic processing and, when designed properly, also being able to detect updates automatically.

The advantages of this solution are that it can often be quickly implemented at the data source and the agreed structure will normally not change, not affecting the scraping functionality.

In the scope of this PoC the following data sources have been harvested using this mixed method:

- European Commission (EC).

To conclude, with limited participation and involvement of the target institution, the harvesting strategy can be improved versus the pure web scraping by creating custom web pages listing the documents to be ingested, in a machine-friendly format.

3.2.2 APPROACHES CONSIDERED FOR THE INCREMENTAL HARVESTING

Once the initial full ingestion of a data source has been performed, the following objective is to ensure to take in account the information about the updates of the data source. Therefore, it makes sense to propose a solution to launch a harvesting which will only harvest the newly created, modified and deleted documents since the last harvesting (instead of re-harvesting the whole data source and resending the whole content for ingestion).

Namely, the purpose is to identify the creation of new documents, the modification of existing documents (content and/or metadata of the document could be modified) and the deletion of documents.

In this scope, an analysis has been performed to determine the best solutions for this incremental harvesting, depending on the constraints of the specific data sources. The two selected solutions are described in the following sections, together with a split of all in-scope data sources to the related category.

3.2.2.1 **MAIN SOLUTION — LAST MODIFICATION DATE**

A comparison between the ‘last modification date’ of a document and the ‘last harvesting date’ is performed to detect new and modified documents.

The ‘last harvesting date’ is saved after each harvesting in a local database.

This solution relies on the assumption that there is a date metadata (such as ‘publication date’ or another dedicated metadata) that is correctly updated each time a document is created or modified.

Below, the two algorithms are described, used depending on the availability (or not) of an endpoint:

3.2.2.1.1 DATA SOURCE WITHOUT ENDPOINT

Creation and modification of documents:
For the website data sources (like the one of ECB), all documents are reviewed at every harvesting, in order to retrieve the ‘last modification date’ from the documents.

When the incremental ingestion is launched, the ‘last harvesting date’ is checked from the local database. Two scenarios are then possible:

- If the document’s ‘publication date’ is >= ‘last harvesting date’, then the document is ingested;
- If the document’s ‘publication date’ is < ‘last harvesting date’, then the document is skipped.

Deletion of documents:
In order to detect deleted documents, firstly, the ids of the documents need to be retrieved from the local database before the harvesting itself, only then, in the second stage, it is checked during the harvesting, if certain ids have been removed from the initial list of ids.

At the end of harvesting, a deletion request is sent to the ingestion API for all remaining document ids.

3.2.2.1.2 DATA SOURCE WITH ENDPOINT

Creation and modification of documents:

For those data sources which expose an endpoint (web service, query engine …) to retrieve the documents, it is almost always possible to filter the documents before the harvesting by using the 'last ingestion date' parameter.

- The SPARQL endpoint of the Council contains encoded information about the date of publication ('publication_date' metadata). The value of this field can be compared with the 'last ingestion date' parameter to identify the new or updated documents. When the incremental ingestion is launched, the 'last ingestion date' is added to the query inside the 'WHERE' clause, in order to retrieve only the documents for which the 'publication date' is greater than or equal to the 'last ingestion date'.
- For the European Parliament API a start and end date is required for the request. As a start date the 'last ingestion date' could be used. We specify them during our initial search, so to only return the documents changed since the last ingestion.

Deletion of documents:

- In order to detect deleted documents of a data source, a comparison is performed between all document ids that have been fetched with the ones that have been stored in the local database during the previous harvesting.

  Afterwards, for all documents that have been identified in the database but have not been re-fetched, it means that they are no longer available in the data source. Thus, a deletion request is then sent to the DIGIT ingestion API.
- An alternative to process the deletion of documents would be, for each harvested document, to query the data source endpoint after the ingestion. For each call that returns an empty result, it means that the document is no longer available and thus a deletion request is sent to the DIGIT ingestion API.

Data sources:

This solution can be used as an incremental harvesting mechanism for the data sources of the Council and the European Parliament. It is because this EU institution proposes a SPARQL endpoint containing a 'publication date' metadata that can be used as the 'last publication date'.

3.2.2.2 **SECONDARY SOLUTION — RSS FEED**

Creation and modification:

An RSS ([10]) feed is a list sorted by date. This allows to loop over all the results and to perform a check against the index in order to find out if the document is already ingested. If the document is not found in the index, the next

([10]) RSS (Rich Site Summary) is a type of web feed which allows users to access updates to online content in a standardised, computer-readable format.

entry is processed. Once a document is already in the index, the fetching mechanism is stopped.

The downside of this approach is that it is not possible to know if a document has changed.

Deletion:
A list of the whole indexed content is kept in a local database. Each time a fetch is performed, a verification is made in the entire RSS feed page to check that no document is missing. If the document is present in the database but not in the RSS feed, a delete command is sent to the API to remove the entry from the database.

Data sources:
This solution can be used as an incremental harvesting mechanism for the data sources of the following EU institution:

- The European Economic and Social Committee (because it uses the RSS feed update mechanism) ([11]).

3.2.2.3 FALLBACK SOLUTION — HASH COMPARISON

The hash of the document content and the document metadata is calculated and compared with the one stored in a local database.

For HTML documents, a cleaning step might be needed before computing the hash value, so that all un-relevant parts (such as the header, menus and footer) are removed from the HTML page and only the HTML part that contains the document content is kept.

Creation of documents:
Once the document metadata is harvested, if the document's id is not found in the local database, it means that it is a new document.

Afterwards, a new entry is created in the local database containing the document id, the hash value of the document metadata and the hash value of the document content.

Modification of documents:
Once the document metadata is harvested, if the document id exists in the local database, the hash value of the harvested metadata is calculated and compared to the value stored in the local database.

If a difference is detected, a request to update the document metadata is sent to the ingestion API.

Afterwards, by using the harvested 'document URL' metadata, the content of the document is downloaded and the hash value is calculated and compared to the value stored in the local database.

If a difference is detected, a request to update the document content is sent to the ingestion API.

[11] The incremental ingestion is not turned on because of the machine being turned off.

Deletion:
In order to detect deleted documents, firstly, the document ids are retrieved from the local database before the harvesting itself and only then, at the second stage during the harvesting, are the harvested ids removed from the list of ids.

At the end of the harvesting, a deletion request is sent to the ingestion API for all remaining document ids.

Data sources:
This incremental harvesting mechanism has been used for the data source of:

- The European Central Bank's (ECB).

3.3 INGESTION

Ingestion is a process of inserting the data into the search database.

The ingestion has been done with a Java Client, which communicates with DIGIT's Enterprise Search Solution, an already exiting interface that has been reused for purposes of the PoC (see section 3.4).

The ingestion API is used to check if a document already exists in the database so the ingestion only happens once, allowing for an incremental ingestion.

3.3.1 METADATA

The metadata are supplied to the API by the end-user using the Java Client. The metadata are then transformed to allow ingestion into the index of the Elasticsearch engine, which is an open-source search engine developed based on another open-source library called Lucene ([12]).

3.3.2 CONTENT

The content can come from two various sources, either a web page or a document. In the case of a web page, the end-user sends a URL to the API. The API will then download the content of the webpage, process its content and prepare it for the indexation into Elasticsearch. When a document is sent, it is processed locally by harvesting the content into a readable value (text content).

Once the content is harvested and sent to the Elasticsearch engine, the content of the files is also sent to a preview API, the task of which is to create picture snapshots of the document's content, for the users to be able to see a glimpse of the results without actually opening the documents themselves. This preview API will add the web

([12]) More information about Elasticsearch can be found at https://www.elastic.co

page or document to a queue, which will eventually be processed in order to generate the first five pages of that document, or the entire page in case of HTML. This process is asynchronous and it can take up to several days to complete, depending on the ingestion speed.

3.4 USER INTERFACE

The user interface, entitled DIGIT's Enterprise Search Solution, is a reused user interface of DIGIT's 'Corporate Search' project. It exposes the capabilities of the search engine and underlying API. The following functionalities are covered:

- Search box;
- Results in an 'infinite scroll' form;
- Choosing the language of the results;
- Filtering by date;
- Faceting by data source (institution) and year;
- Sorting by relevance or date;
- Displaying basic information about the results (title, source, date, type of document);
- Displaying of a text snippet of the result with the term being searched highlighted;
- Related documents;
- Previews of the resulting documents;
- Autocomplete
- Estimated number of matching results.

The user interface is multilingual and compatible with all major browsers in the market.

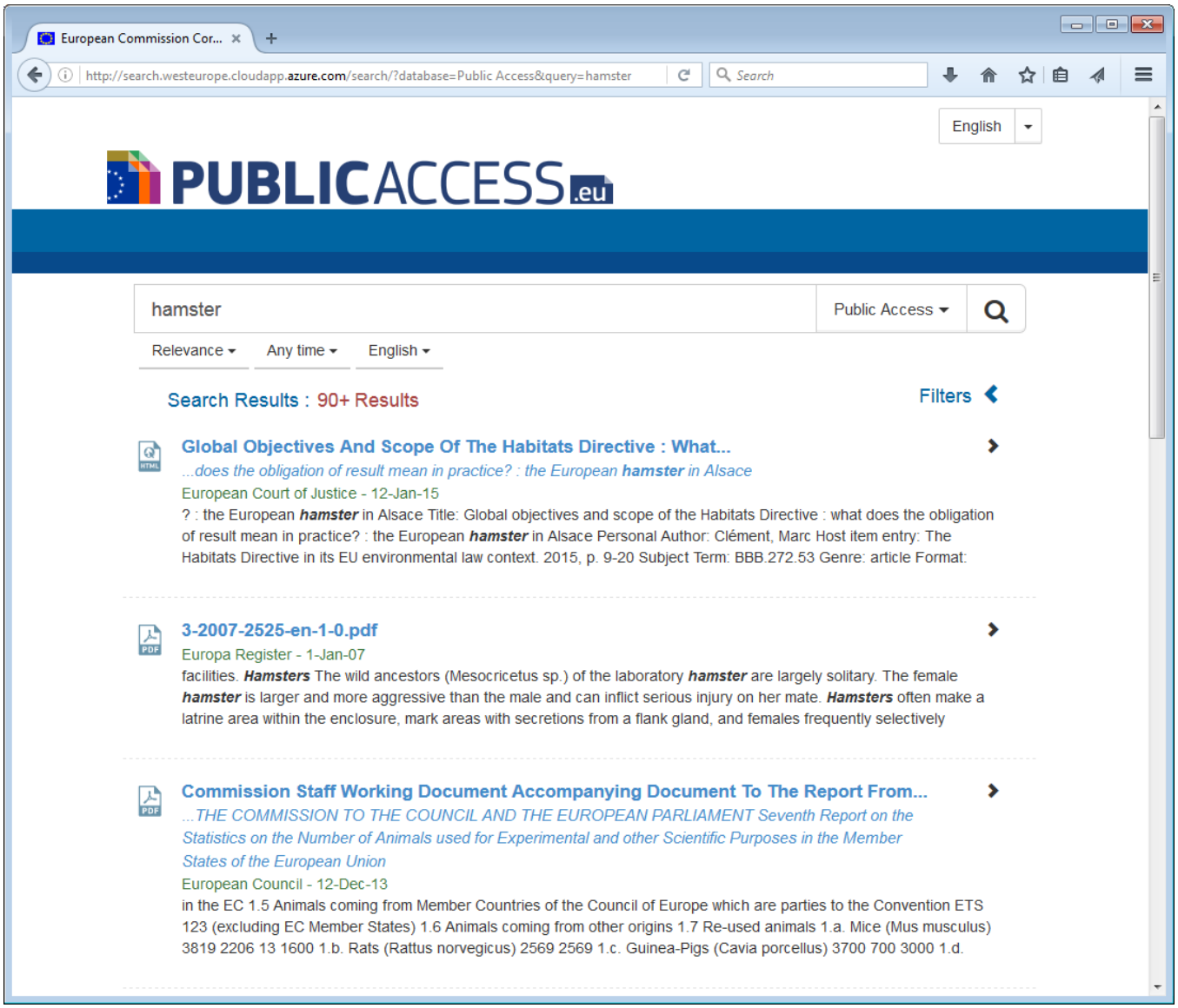

Figure 1 — PAS Search user interface

The technical solution supporting the search engine is Elasticsearch, accessed over an API developed at DIGIT.

The use of Elasticsearch is in line with the desire of DIGIT to use open source solutions in parallel with proprietary software, in order to reduce the risk of captivity, while being able to provide the best tool for the task at hand.

The search is performed on both metadata and data (content). The solution is equipped with facets that allow users to filter the result based on their needs. These filters are based on the metadata harvested from the data sources. In the figure below, only results from the Europa Register are shown with a possibility to further narrow down the search results by year.

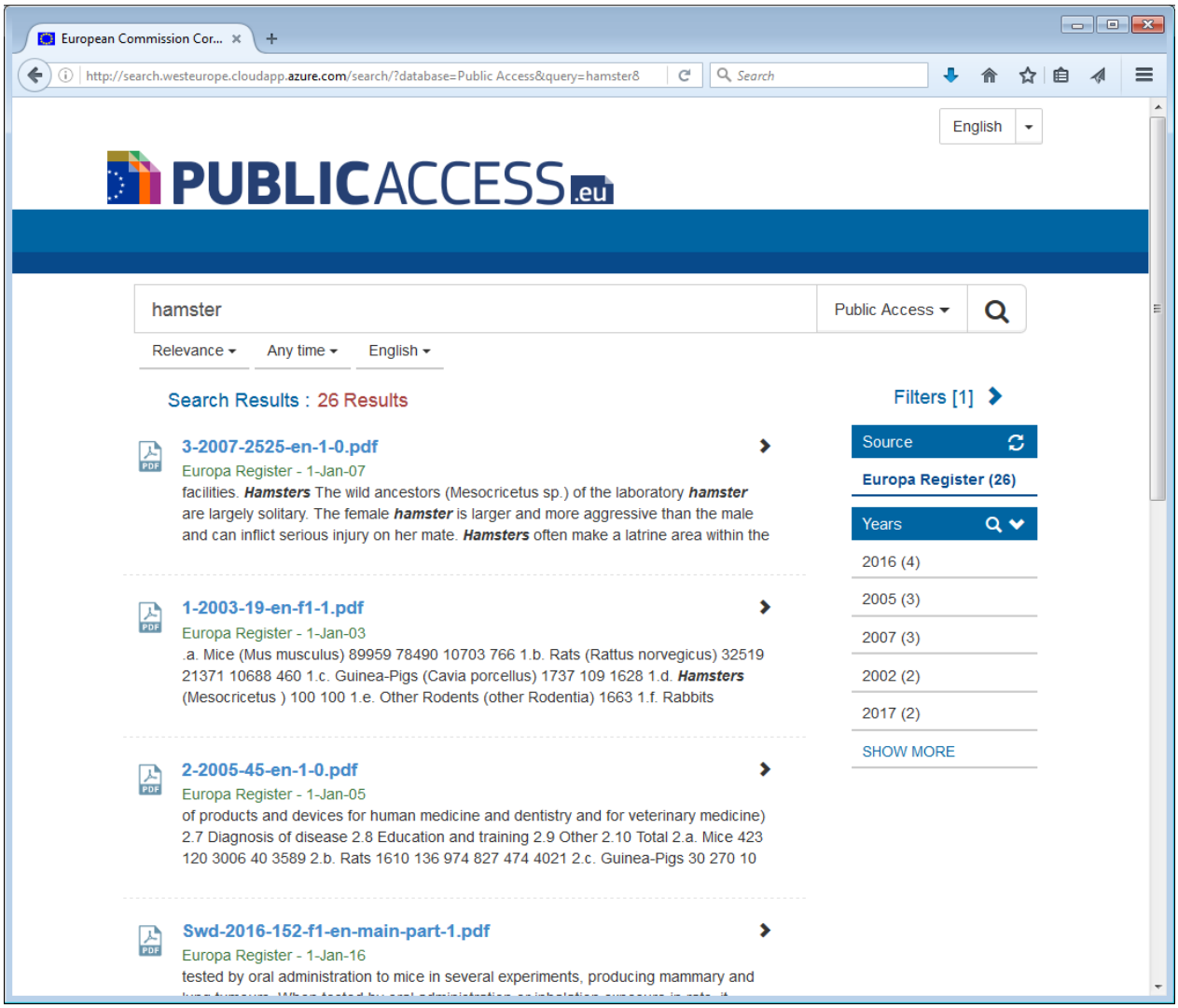

Figure 2 — PAS Search facets

The data indexed is multilingual and the user interface allows displaying the results in given languages only. Some more advanced features like Boolean logic and wildcards are also available. Precise instructions on how to use them can be provided as part of the user interface (however, this was not the case in the version used for the PoC).

3.4.1 KEYWORD SEARCH

The search conceptually matches queries that consist of a single keyword. It stems the keyword, and then it finds documents that contain words that have the same stem as the keyword. For example, in case of a query with the word ‘lovely’, it stems the word to ‘love’ and finds documents that contain words that are also stemming to ‘love’, for example, ‘lovely’, ‘lover’, ‘loved’, ‘loving’, and so on. It needs to be noted that this stemming is language specific. Therefore, one should always specify the language of the search terms in the language selector.

3.4.2 EXACT PHRASE SEARCH

By putting the searched term into quotation marks, the term is treated as a phrase. In such case, the search result returns only the documents in which a matching phrase occurs (a phrase in a document qualifies as a match, if it stems the same way as the query phrase). For example, searching for a phrase ‘fresh and lovely’ will match not only the documents containing that exact phrase, but also documents containing the phrase ‘fresh love’ (for example). This is because of stemming, and the application of Stop Words (see below under section 3.4.4).

3.4.3 BOOLEAN SEARCH

By default, the search will apply the OR operator for each term present in the search query. For example, a query for ‘European politics’ will match documents that contain ‘European’, ‘politics’, or both.

+ A query for ‘European +politics’ will only return documents that contain both terms.

– A query for "European –politics’ will return only documents that contain the term European and exclude the word politics.

3.4.4 STOP WORDS

Stop Words refer to very common words that the search does not index. In English, words such as ‘the’ or ‘a’ occur too frequently to carry any significance. Therefore, the search does not require them to understand the concept of the text. A set of Stop Words exists for every language that a user can select.

4 OUTCOME OF THE PROOF OF CONCEPT

4.1 OVERVIEW

The goal of this phase was to design early solutions and strategies to harvest data from heterogeneous data sources. As a result, a single search engine has been developed, browsing documents from different EU institutions, thus proving the feasibility of the initially foreseen harvesting and ingestion approaches. The proof of concept has covered around 2 450 500 indexed documents.

This PoC has been divided into three modules:

- The harvesting module, in charge of harvesting the documents and their metadata from the data sources and sending them to the ingestion module;
- The ingestion module, in charge of receiving the harvested documents and their metadata through the ingestion API and in charge of indexing these documents and their metadata in order to make them available through a search API;
- The user interface module, which is a web application enabling end-users to search and preview the documents from different EU institutions and access the original documents on the EU institutions data sources.

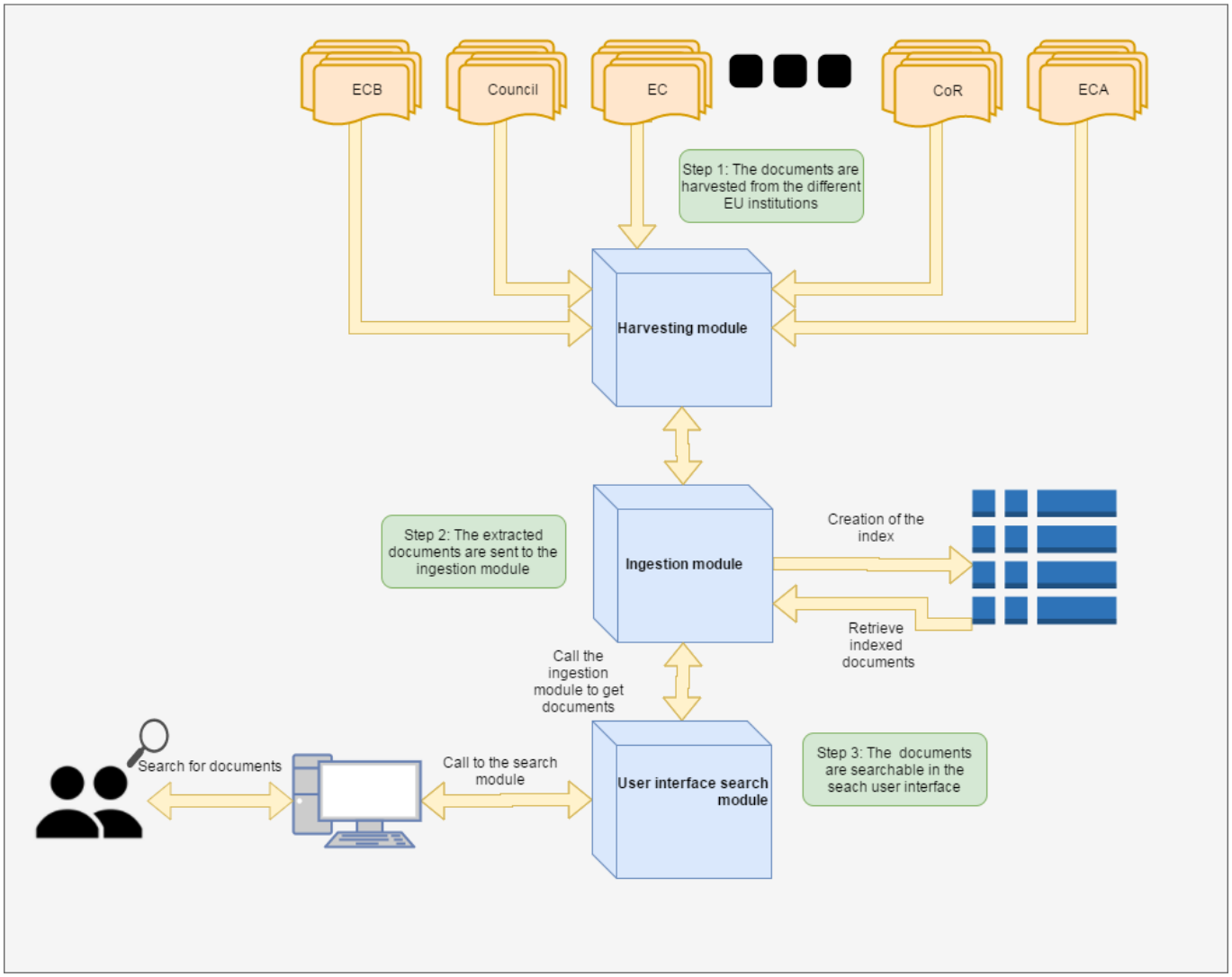

Figure 3 — Public Access PoC workflow

he following sections describe each module of the above figure, with specificities of each EU institution already harvested in the scope of the PoC.

4.2 HARVESTING MODULE

4.2.1 EUROPEAN CENTRAL BANK (ECB)

4.2.1.1 IMPLEMENTATION

The Study has indicated that there are around 198 000 documents available on the ECB website, in multiple formats (PDF, HTML or EPUB) and languages. By the ECB website is meant the official website https://www.ecb.europa.eu/home/html/index.en.html

No endpoint is available. Therefore, for harvesting, the 'web scraping' method (described in the section 3.2.1.3) has been used, based on Groovy harvesting scripts. The complete workflow implemented for the ECB full harvesting has been as follows:

1. Before launching the harvesting application, the URL of the pages that will be harvested must be configured.
2. Once the harvesting application is launched, the HTML content of the pages is downloaded and cached on the file system.
3. For each harvested page, harvesting scripts are called by a Java application in order to harvest a list of documents from that page.
4. Once the list of documents harvested, a CSV ([13]) report is created containing all metadata of the harvested documents.
5. All harvested documents are sent to the ingestion API to be indexed.

From the above process, the fetching of each page is done by using the Apache HTTP client. In order to respect the robots.txt crawling delay ([14]) of each data source, a throttling mechanism has been implemented. Finally, in order to avoid downloading the same resource several times, a simple file system-based caching mechanism has been implemented: if the resource is not in the cache, the resource is downloaded.

Once a page has been fetched, it is loaded in the memory as a Java object. The loading is performed by the Tagsoup ([15]) parser that allows parsing malformed HTML. Then, the data harvesting is delegated to several Groovy ([16]) scripts, taking as an input the previously loaded object.

In order to stay flexible and be able to parse any website, the choice has been made to use Groovy scripts to harvest data. One Groovy script can be used for one or more web pages, depending on the structure of the website's HTML code.

([13]) Comma-Separated Values, a simple text format.
([14]) https://en.wikipedia.org/wiki/Robots_exclusion_standard#Crawl-delay_directive
([15]) https://hackage.haskell.org/package/tagsoup
([16]) http://groovy-lang.org

The Groovy technology offers a simple way to navigate through the HTML ([17]) DOM ([18]) tree by using the CSS ([19]) selector, the HTML node name and the HTML attributes.

After the harvesting, the application cleans the data and builds a DTO ([20]) corresponding to the fields needed to call the ingestion API.

As described in section 3.2.2, the PoC application can run in an incremental harvesting mode.

Concerning the ECB data source, the 'Hash comparison' method (described in section 3.2.2.3) must be used for the incremental harvesting. This choice has been made because the ECB data source does not have a reliable 'last modification date' metadata on its documents for performing the 'last modification date' method (described in section 3.2.2.1).

In the scope of the PoC, the following set of document types has been harvested, the information in the parenthesis providing the link to the page from which the harvesting has been performed (this sample shows only the English language URLs from 2017):

- Article (http://www.ecb.europa.eu/pub/economic-bulletin/articles/html/index.en.html)*;*
- Legal documents: Opinion, Decision, Guideline, Recommendation and Regulation (https://www.ecb.europa.eu/ecb/legal/date/previous/html/index.en.html);
- Economic bulletin (https://www.ecb.europa.eu/pub/economic-bulletin/html/index.en.html);
- Economic Research — Research Bulletin (https://www.ecb.europa.eu/pub/economic-research/resbull/html/index.en.html);
- Press release (https://www.ecb.europa.eu/press/pr/date/2017/html/index.en.html);
- Interview (https://www.ecb.europa.eu/press/inter/date/2017/html/index.en.html);
- Speech (https://www.ecb.europa.eu/press/key/date/2017/html/index.en.html);
- Annual report (https://www.ecb.europa.eu/pub/annual/html/index.en.html);
- Convergence report (https://www.ecb.europa.eu/pub/convergence/html/index.en.html);
- Financial stability review (https://www.ecb.europa.eu/pub/fsr/html/index.en.html);
- Macroeconomic Projections (https://www.ecb.europa.eu/pub/projections/html/index.en.html).

For each of the above document types, the following metadata have been harvested when available:

- Title;
- Source URL;

([17]) HTML: HyperText Markup Language is the standard mark-up language for creating web pages and web applications.
([18]) DOM: Document Object Mode is a cross-platform and language-independent application programming interface that treats an HTML, XHTML, or XML document as a tree structure wherein each node is an object representing a part of the document.
([19]) CSS: Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a mark-up language.
([20]) DTO: Data Transfer Object is an object that carries data between processes.

- Sub-title;
- Publication date;
- Document ID;
- Document URL;
- Language;
- Document type;
- Author;
- Website Section.

Some ECB document types are multilingual.

The available languages on the ECB website are: BG, ES, CS, DA, DE, ET, EL, EN, FR, HR, IT, LV, LT, HU, MT, NL, PL, PT, RO, SK, SL, FI and SV. However, the translated documents are not always translated in all languages.

The following list gives details on the ECB multilingual document types, if the content, the metadata or both are multilingual:

- [Legal documents:] Opinion, Decision, Guideline, Recommendation and Regulation: both content and metadata are translated for some documents of this type;
- Press release: only the content is translated for some documents of this type;
- Interview: only the content is translated for some documents of this type;
- Speech: only the content is translated for some documents of this type;
- Annual report: both content and metadata are translated for all documents of this type;
- Convergence report: both content and metadata are translated for all documents of this type;
- Projection: both content and metadata are translated for all documents of this type.

4.2.1.2 **ISSUES AND CONSTRAINTS**

During the implementation of the harvesting of ECB documents, some issues and constraints have been encountered. Here are the main ones:

- The lack of knowledge about the website structure. It was not easy to find where each document type was located on the website;
- The un-standardised web pages on the website, i.e. two pages both containing a list of documents are structured in a different way;
- The documents metadata are not in a uniform format. For instance, for two different documents types the publication date was displayed differently;
- With no business knowledge about the ECB documents and metadata, it has been more complicated to identify the relevant parts which need to be harvested;
- There was no contact point at ECB with whom it would have been possible to interact and get help on finding the best way for retrieving the documents metadata;

- It is difficult to check, if the harvesting is complete, as there is neither a single access point to get all documents, nor a way to know the exact number of available documents.
- For some documents, the metadata are harvested from different places (from the HTML content and from the PDF of the document when the HMTL content does not contain the metadata);
- The Groovy scripts used to parse the content of the page are strongly bound to the content of the web pages; therefore, if the ECB website changes, meaning that the path to harvest the node in the HTML source changes, then the scripts must be updated. Otherwise, the harvesting would not work.

As a general remark, related to what is behind all these issues, it needs to be mentioned that the 'web scraping' method should be used only as a fallback solution because the HTML content is not made to be parsed by another application but is made to be displayed in a web browser.

4.2.1.3 **RESULTS**

During the implementation of the PoC, around 36 000 documents (about 20 % of the content identified by the Study) of the ECB website have been harvested and sent for ingestion (see http://search.westeurope.cloudapp.azure.com/search/?lang=any&database=Public%20Access&query=***&facets=DATABASE%3DOP_ECB). In this sample, not all the document types (as listed above) were in the scope of this initial ingestion.

The validation of the harvested documents has been performed by checking random documents on the ECB website and verifying that the selected document is available in the search user interface. During this validation phase, the following points have been raised:

- For the 'Economic Research — Research Bulletin', only the documents from year 2016 have been harvested. (Documents from 2017 can be harvested by simply providing the related URL to the harvesting application).
- For the 'Financial stability review' document type, only the 'Full PDF' documents have been harvested.
- For the 'Macroeconomic Projections' document type, only the documents under the filter 'Breakdown by country' have been harvested. Also for this page, it needs to be highlighted that the page structure has recently completely changed and thus the harvesting script for this page needs to be refactored to keep the harvesting working.
- The 'Decision' documents have not been fully harvested, as some documents were available from different web pages of the ECB website and only one of these pages has been harvested.

4.2.1.4 **RECOMMENDATIONS**

First of all, ECB could be encouraged to propose an endpoint (Web Service API, FTP access, query engine …) which would help retrieving the documents, thus preventing the constraints listed in section 4.2.1.2.

A good asset to complete the ingestion of ECB would be to work together with the ECB website owner, who could explain further how the website is structured, in order to have a better global overview and to know where each document type is and which metadata are available.

An earlier in-depth analysis of the website content in collaboration with OP before the implementation process (rather than the mere analysis of the ingestion result) would ease the implementation of the connector for ECB.

4.2.2 COUNCIL OF EU AND EUROPEAN COUNCIL

4.2.2.1 IMPLEMENTATION

For purposes of the PoC, only the Public register of Council documents has been harvested, thus the following description is only about this register ([21]).

This register contains about 1 195 000 documents. The documents of this data source are exposed on a SPARQL endpoint, located on the open data portal http://data.consilium.europa.eu/sparql

Using the endpoint documentation ([22]), the harvesting has been performed by querying the endpoint.

SPARQL queries aim to provide a query language to fetch data from RDF stores. RDF stores keep only metadata of documents, not the content itself.

The used SPARQL query for the harvesting is:

```
PREFIX codi: <http://data.consilium.europa.eu/def/codi/>
PREFIX dcterms: <http://purl.org/dc/terms/> PREFIX schema: <https://schema.org/>
SELECT * where {
  SELECT DISTINCT ?docExp as ?reference str(?docTitle) as ?title ?docLang as ?language ?docDate as ?pub_date ?docPDF as ?pdf_uri ?docCat as ?doc_type str(?docSourceURL) as ?source_url
     WHERE {
        ?docExp a codi:DocumentExpression.
        ?docExp dcterms:title ?docTitle.
        ?docExp dcterms:language ?docLang.
        ?docExp dcterms:audience ?adressee.
        ?docExp dcterms:issued ?docDate.
        ?docExp foaf:page ?docPDF.
```

([21]) Description of the data published by the Council is available on: http://www.consilium.europa.eu/en/general-secretariat/corporate-policies/transparency/Understanding-Open-Data-Datasets_pdf.
([22]) Documentation of the EU Council SPARQL endpoint could be found on: http://www.consilium.europa.eu/en/general-secretariat/corporate-policies/transparency/open-data

```
        ?doc codi:expressed ?docExp.
        ?doc codi:documentCategory ?docCat.
        ?doc foaf:page ?docSourceURL. }
        ORDER BY DESC (?pub_date)
    }
OFFSET ${from} LIMIT ${to}
```

The process of fetching the data from the SPARQL REST endpoint is as follows:

1. Execution is split per chunk; the chunk size is determined by using the '${from}' and '${to}' parameter in the above SPARQL query.
2. For each chunk, execute the SPARQL query:
    - For each return row, build a DTO ([23]);
    - For each DTO built, call the DIGIT ingestion API using the DIGIT API client.

Here are some technical libraries used to perform the harvesting:

- The global application is a Spring Boot ([24]) project.
- The RestTemplate ([25]) class of the Spring Framework ([26]) has been used to make the call to the SPARQL endpoint.
- Then, to build the DTO from the returned JSON answer of the SPARQL endpoint, the FasterXML Jackson Databind ([27]) library has been used.

The following document types have been harvested:

- Standard — multilingual metadata and content;
- Communication (provisional agendas of working parties) — multilingual metadata and content;
- Intergovernmental conference — multilingual metadata and content;
- European Convention — multilingual metadata and content;
- European Parliament document — monolingual.

([23]) https://en.wikipedia.org/wiki/Data_transfer_object
([24]) https://projects.spring.io/spring-boot
([25]) https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/web/client/RestTemplate.html
([26]) https://projects.spring.io/spring-framework
([27]) https://github.com/FasterXML/jackson-databind

For all harvested documents, the following metadata have been harvested:

Field	Type	Description
Title	Text	The title of document
Source URL	Text	The URL that point to the document webpage in the data source describing the document metadata.
Publication date	Date	The date when the document is published.
Document ID	Text	The identifier of the document in the data source.
Document URL	Text	This is the URL that point to the document content.
Language	Text	This is the language (ISO code) of the document.
Document type	Text	This is the type (code) of the document.

4.2.2.2 ISSUES AND CONSTRAINTS

During the harvesting of this data source, some memory issues have been faced in the harvesting application, thus the harvesting workflow had to be modified.

Another encountered issue has been that the communication between the harvesting and the ingestion modules was too long, thus the call to the DIGIT ingestion API had to be parallelised.

Each time a technical error at DIGIT ingestion API side (see section 4.3.2) was raised, the harvesting application stopped (as it was required) but without a resume mechanism. The restart of several complete harvesting of the data source and the associated calls to the ingestion API was very time consuming.

4.2.2.3 RESULTS

To validate the harvesting of the Public register of Council documents, a check was performed to count how many documents were received by the DIGIT ingestion API.

Here are the results of the harvesting performed in May 2017:

1 192 454 documents out of 1 194 327 sent documents have been received by the ingestion API:

- 1 194 462 (from the harvesting module log files) harvested documents – 1 194 327 (from the harvesting module log files) sent to ingestion API documents = 135 documents have not been sent to the ingestion API;
- 1 194 327 sent to the ingestion API document – 1 192 454 (from the ingestion module log files) indexed documents = 1 873 documents have not been indexed correctly.

In the log files, the harvesting application received about 140 error responses from the ingestion API. Thus, what we can conclude is that 135 documents have not been sent because the harvesting application execution stopped. For the 1 733 remaining documents (1 873 – 140), as there is no further error in the log file, we can guess that

errors occurred internally in the ingestion application. To conclude, about 99.8 % of the harvested documents have been correctly indexed.

4.2.2.4 RECOMMENDATIONS

All the faced issues for this data source were not related directly to the data source itself, thus there is no particular recommendation to be made for the data source as such.

In the future, in order to save the time needed for several restarts of the complete ingestion, an automatic restart mechanism or failover mechanism with associated logging could be foreseen.

Finally, it would be good to have a contact point for the Council's endpoint, to get reliable information about the harvested metadata and the frequency of updates of the data source.

4.2.3 EUROPEAN PARLIAMENT (EP)

4.2.3.1 IMPLEMENTATION

The Parliament's API has been supplied to index the whole Parliament's content:
http://www.refregrs.europarl.europa.eu/RegisterRS-1.0/services

For purposes of the PoC, the following subset of documents has been indexed:

— Verbatim report of the proceedings — revised version (item 1.2.3.2 of the 'tree');
— Finalised minutes (item 1.2.4 of the tree);
— Texts adopted in plenary — finalised edition (item 1.2.6.2 of the 'tree');
— Consolidated texts first and second reading (items 1.2.6.3.1 and 1.2.6.3.2 of the 'tree').

The provided Parliament's API is a secured JSON rest API. The advantage of using this API is that it is possible to request all documents changed since the last indexation. This offers the advantage of not having to store the information about the harvesting status in a local database to keep track of the changes.

Quick overview of the metadata available:

Field	Type
Type	Description of the document
Author	Author of the document
Authority	An authority of the document is the organism that created the document. This could be DIGIT for example.
Reference	Provides extra details on the document
Theme	The general theme of the document
Type	The type of the document

The metadata supplied is:

Field	Type	Description
authorCode	Keyword	Authority Code links to Authority Description and Authority Rights
authorLogin	Keyword	The Username of the Authority, used for author filtering
authorType	Keyword	Author Type links to authority Description
authorityClassification	Keyword	The Classification of the Authority. All of our documents had authority Classification 'I'. This is related to the document we had permission to access.
authorityCode	Keyword	Authority Code. The values found are 600345, 600365, 60027. The codes allow us to find the authority Descriptions
authorityDescription	Keyword	Description of the authority
authorityGedaCode	Keyword	Geda Code of the authority. No values found during the harvesting
authorityRegisterCode	Keyword	Register Code of the authority. Values harvested: P5, P6 and P7
authorityServiceCode	Keyword	Service Code of the authority. No values found during the harvesting
authorityStartDate	Date	Date on when the authority started. Three different ranges were found.
authorityStatus	Keyword	Status of the Authority, values found A and H
authorityType	Keyword	Type of the Authority, value found PE
comment	Keyword	Comment made on the document. This text is searchable and can be used for highlighting. However at the moment it will not be highlighted and only the content can be highlighted. This is configurable in the API.
docId	Keyword	Unique identifier of the document
documentDate	Date	The document date is used for filtering and sorting on.
extension	Keyword	Extension of the document. Many values were found, the most popular ones are PDF (50 %), xml (15 %), html (15 %), doc (10 %).
fragment	Keyword	Multiple version of a document can exist. These can be split up in fragments. Values can be FULL, LP, VOT. Many more were found but have less than a total of 100.
language	Keyword	All European Languages are represented
name	Keyword	Name of the document
original	Keyword	If the document is the original document of not. Values can be true or false
path	Keyword	The path of the document. This can be a path on the local drives or and http path. The path does not contain the filename. The path seems to be present in about 7 % of the documents
receiptDate	Date	Document receipt Date

refName	Keyword	Name of the reference
refType	Keyword	Type of the reference, most popular reference types, PTAD, PPVD, PPVP, TCPL.
relationName	Keyword	Multi value field of the relations this document has. Most popular values are NUPE, NUSE, NUTA
relationValue	Keyword	The relation value is a numeric (with comma). This seems to be the importance of the relation.
status	Keyword	Status of the document. Values pound P and A
term	Keyword	Parliamentary term during which the document was finalised. 5, 6, 7 are available.
themeCode	Keyword	The internal theme identifier
themeDescription	Keyword	A short description of the theme
themeType	Keyword	The type of the theme, values found: Eurovoc-REP, Eurovoc and Eurovoc-Matiere
typeAuthorityRegisterCode	Keyword	Link from type to authority
typeCategorie	Keyword	The category of the type. No categories were found.
typeClassification	Keyword	The document type classification, value found 'I'.
typeComment	Keyword	A short comment about the document type
typeConfidentiality	Keyword	Confidentiality of the document. All our documents are PUB since they are public.
typeDefaultPath	Keyword	The type default paths are all network drive paths.
typeDescription	Keyword	A short description of the type
typeDiffusion	Keyword	The diffusion of the type, value found PUBLIC
typeDiffusionAuthorityCode	Keyword	Link from Type Diffusion to Authority (code and type generate a UUID)
typeDiffusionAuthorityType	Keyword	Link from Type Diffusion to Authority (code and type generate a UUID)
typeDiffusionPath	Keyword	Diffusion path, all values seem to be NAS paths.
typeFamily	Keyword	Family of the Document type, values found, TA,PV, PV-PROV, TC1-COD, CRE-PROV
typeGroupId	Keyword	Group of the type, value found DPLEN_GRP
typeId	Keyword	Unique identifier of the type, values found PTAD, PPVD, PPVP, TCPL, PCREP
typeInstitutionalAuthorityCode	Keyword	Type Institution link to Authority (code and type generate a UUID)
typeInstitutionalAuthorityType	Keyword	Type Institution link to Authority (code and type generate a UUID)
typeManagementSystem	Keyword	Management Type, values found 6,18, WS APIE, REDMAP
typeParentId	Keyword	Identifier of the parent type id
typePublicationDelay	Keyword	Publication delay. All of the documents found had a publication delay of 0
typeStatus	Keyword	Type status, value found A

typeThemeType	Keyword	A Type can also have a theme.
typeTreatment	Keyword	Treatment Type, values found 5, 11, 10
typeTreeCoordinates	Keyword	Type Coordinates, many numeric values were found.
version	Keyword	Version of the document
year	Keyword	The year value is harvested from the document date. This is used for facet searches on years.

4.2.3.2 ISSUES AND CONSTRAINTS

There were some initial problems while retrieving the data from the production environment. These issues have been resolved by the Parliament.

4.2.3.3 RESULTS

Number of documents found: 318 615

Number of documents ingested: 315 321

Number of documents failed: 3 294

- 1983 due to invalid URL's being supplied;
- 1142 due to an invalid language being found;
- 166 content of document was not extracted, but document was ingested;
- 2 due to load balancer issues.

4.2.3.4 RECOMMENDATIONS

A thorough reading of the official documentation is recommended because the schema of the API is vast and needed for a fully successful ingestion.
Data quality needs to be analysed, since a lot of URLs do not exist. For example the following document is supplied via the API but does not exist:
http://www.europarl.europa.eu/RegData/seance_pleniere/textes_adoptes/definitif/2002/05-16/0249/P5_TA(2002)0249_ES.pdf

4.2.4 EUROPEAN COMMISSION (EC)

4.2.4.1 IMPLEMENTATION

For purposes of the PoC, only the Register of Commission documents have been harvested, thus the following description is only about this register.

The register was already exposed via the following webpage, listing all available files:

http://ec.europa.eu/transparency/regdoc/?fuseaction=idolindexation

The connector to the register had already been implemented within the DIGIT's 'Corporate Search' project. Therefore, it was merely added to the PoC.

The harvesting workflow is as follows:

- List all files;
- Verify against index, if a file is already present;
- If a file is not present, download the file;
- Send the file to the ingestion API.

This workflow allows also for performing the incremental harvesting of the content.

All HTML pages have been ignored, since their contents are the same as the one of the PDF files. Therefore, only PDF files are indexed.

The metadata harvested are:

Field	Type	Description
Title	Keyword	Harvested from the document, this is the title of the document
Date	Date	Only the year of the document could be harvested from the URL. However, as the date is a mandatory element, the sorting is possible. That is why a date field, with the date on 1 January of the year found inside the URL, has been created.
Year	Keyword	The year is harvested from the URL. This metadata is used across all data sources (for a generic filter).
DocId	Keyword	A unique identifier of the document
filename	Keyword	The actual name of the file. If no document title is present, the filename is displayed.

All official EU languages are present and the year is present in the URL, starting at 2001.

In relation to the abovementioned incremental harvesting of the content, the update mechanism works because the list of files is sorted. The scraping is stopped at the moment when encountering a file that has already been ingested.

4.2.4.2 ISSUES AND CONSTRAINTS

231 files failed to get ingested:

- 46 due to an invalid language being passed to the API;
- 150 due to the Apache maximum number of connections being reached;
- 35 due to too many database connections being made.

4.2.4.3 RESULTS

All retrieved content has been indexed. This resulted in 349 638 files being indexed as of 19/06/2017. The incremental ingestion is activated; therefore, this number keeps growing.

4.2.4.4 RECOMMENDATIONS

More metadata could be harvested from the PDF files using standard Java libraries. This was not done since it requires a significant amount of processing power, not available for this PoC project.

4.2.5 COURT OF JUSTICE OF EUROPEAN UNION (CJEU)

4.2.5.1 IMPLEMENTATION

For purposes of the PoC, the data source to harvest was the website of the Court of Justice.

However, due to a misunderstanding and no cooperation on the Court of Justice side, the implementation has wrongly covered the Library content: http://www.bib-curia.eu/client/en_US/default/search/results?te=ILS

This mistake would have been quickly revealed, had the Court of Justice been consulted. Without it, however, the mistake was discovered very late and therefore, within the allocated timeframe, it wasn't possible to redo this part of harvesting.

No API was supplied by the Court of Justice, therefore a custom crawler for the fetching of the content had to be written.

The HTML page of each search result has been indexed as the main content. The maximum of available metadata has been harvested from the description page and added to the indexation process.

The harvested metadata are:

Title	Type	Description
formats	Keyword	Formats of the book. Most values were ARTICLE, BOOK and CR
isbn	Keyword	Reference of the result
language	Keyword	Language of the book. Multiple values possible.
year	Keyword	Year as harvested from the search results
shelf	Keyword	Shelf were the book is located
series	Keyword	Series of the book
subject_term	Keyword	Reference to a subject. This is a hyperlink to another page with descriptions. We did not index those since we were already running into blacklisting issues.
curia	Keyword	Description of the book
edition	Keyword	Edition of the book

genre	Keyword	Genre of the book, for example article, these, suggestion commande, etc …
issn	Keyword	Reference to a secondary book
general_note	Keyword	Notes about the book
docId	Keyword	Reference of the doc. We used the URL of the page.
Date	Date	The date is based on the year. We had to use 1 January since we did not have exact date. This is then used for filtering and sorting.

4.2.5.2 ISSUES AND CONSTRAINTS

The RSS feed page is broken and does not allow the incremental indexation.

There is a security feature which blocks the fetcher from getting too much information at once. This means that the fetcher gets very easily blacklisted. To overcome this, the fetching was manually re-run several times. Additionally, there is a high number of pages that were refused by the fetcher (precise numbers not stored).

4.2.5.3 RESULTS

Number of documents ingested: 462 288

Number of documents not ingested: 1 061

All the failures of the documents ingestion occurred because the document language was not supported by the ingestion module. For example, the following linguistic versions appear among the harvested documents: RU, UK, LA, NO ([28]).

There are also around 20 000 documents not fetched from the site, due to security constraints.

4.2.5.4 RECOMMENDATIONS

- Collaboration of the Court of Justice is necessary for a successful ingestion.
- The RSS feed needs to be fixed or an API needs to be provided.
- The fetcher IP address ([29]) must be whitelisted, so that a heavier load fetch can be performed.
- Data quality can be improved by validating the ingested data.
- More languages need to be added to the user interface (currently supporting only EU official languages).

([28]) Russian, Ukrainian, Latgalian, Norwegian.
([29]) An Internet Protocol address is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication.

4.2.6 EUROPEAN ECONOMIC AND SOCIAL COMMITTEE (EESC)

4.2.6.1 IMPLEMENTATION

The data source does not provide an API. Therefore, the content needed to be harvested from the EESC webpage. The easiest way to do this was to use the RSS feed.

https://dm.eesc.europa.eu/EESCDocumentSearch/_layouts/srchrss.aspx

The harvesting workflow is as follows:

1. Loop through all the pages.
2. Each result found is checked for availability in the index.
3. If it's present in the index, the fetcher is stopped. Since an RSS feed is sorted by date, and each document in the RSS is fetched sequentially, if one document is found to be already in the index, it means that all the following documents (older ones) have already been ingested.
4. Until an ingested document is found, the content continues to be sent to the ingestion API.

In this way, the full harvesting can be run for the first time around and the same code can be reused over and over again for the incremental harvesting.

The script is currently run on demand, not on a daily basis. This is because the infrastructure does not stay on 24/7, in order to lower the costs.

The RSS feed allows for going page by page and harvest the whole content.

The following metadata has been harvested from the RSS page:

Title	Type	Description
title	Keyword	Available in the RSS feed; this is the title of the document.
year	Keyword	Year of the document is available in the RSS and is used for facet filtering. The value is derived from the date field.
docId	Keyword	Unique identifier of the document. This is the URL.
filename	Keyword	File name of the document is harvested from the URL.
author	Keyword	Author of the document, available inside the RSS feed.
summary	Keyword	A summary of the document is also available in the RSS feed. This is indexed but is not highlighted, since its content is the same as of the file itself.
date	Date	The date is available inside the RSS feed.

4.2.6.2 ISSUES AND CONSTRAINTS

There was a need to limit the number of requests made to the RSS feed, otherwise fetcher would get blacklisted. The fetch got limited to one request per second. This means that the full indexation took 4 days.

There have been a few documents with invalid language codes (such as 'UK' as a language).

4.2.6.3 RESULTS

Number of documents ingested: 171 632

Number of documents not ingested: 10

All errors occurred due to the timeout issues with the Apache machines. This can easily be solved by configuring the apache machines to allow more simultaneous connections.

4.2.6.4 RECOMMENDATIONS

The implementation could be improved, if an access to better metadata is provided, namely through an API or a database that could be used for indexation. Since the application used to host the documents on the data source is SharePoint, this could be done easily.

4.2.7 EUROPEAN RESEARCH COUNCIL (ERC)

4.2.7.1 IMPLEMENTATION

No API was supplied for this source, so the content was harvested from the ERC website:

https://erc.europa.eu/document-library

The website does not allow for getting the search results sorted by date. Therefore, a script had to be written which would click on all categories and paginate all documents.

The metadata harvested are:

Title	Type	Description
Title	Keyword	Title of the document. Available on the page.
Date	Date	Date of the document. Available on the page
Year	Keyword	The year field is harvested from the date field.
DocId	Keyword	Unique identifier of the document. Based on the URL.
Category	Keyword	Category of the document. Available on the page.

4.2.7.2 ISSUES AND CONSTRAINTS

There was no way to perform an incremental ingestion for this source. Therefore, each document is downloaded every time to the fetcher machine and content gets re-ingested.

In the case of this data source, this is not a major issue, since the number of documents is relatively small.

4.2.7.3 RESULTS

Number of documents ingested: 350

4.2.7.4 RECOMMENDATIONS

- If possible, use of an internal API is recommended.
- Harvest more metadata from the document downloaded.

4.2.8 BODY OF REGULATOR FOR ELECTRONIC COMMUNICATIONS (BEREC)

4.2.8.1 IMPLEMENTATION

There was no API available to index the data. Therefore, a custom crawler was used to fetch all the data on the BEREC website: http://www.berec.europa.eu/eng/document_register/subject_matter

The website does not allow for performing a search sorted by date. Therefore, a script had to be written which would click on all categories and paginate all documents.

The metadata harvested are:

Title	Type	Description
Title	Keyword	Title of the document. Harvested from the page.
Date	Date	Date of the document. Harvested from the page.
Year	Keyword	Year of the document. Harvested from the date field.
DocId	Keyword	Document reference. This is based on the URL.
Category	Keyword	Category of the document. Harvested from the page.

4.2.8.2 ISSUES AND CONSTRAINTS

Firstly, a small number (10) of URLs could not be indexed. This happened due to a file system limitation concerning a permitted length of the path. The URLs extracted from the file were encoded in base64[30] to the file system. If the path is too long, then the file path is too long, too, and, consequently, a file cannot be saved. In order to fix this, the fetcher´s logic would need to be improved.

Secondly, there was no way to perform an incremental ingestion for this data source. Therefore, all documents get downloaded every time to the fetcher machine and the content gets re-ingested.
In the case of this data source, this is not a big issue, since the number of documents is relatively small.

4.2.8.3 RESULTS

Number of documents ingested: 3 008

[30] https://en.wikipedia.org/wiki/Base64

4.2.8.4 RECOMMENDATIONS

- If possible, use of an internal API is recommended.
- Harvest more metadata from the downloaded documents.

4.3 INGESTION MODULE

Fetching is a process of collecting information from the data sources, with some pre-processing if necessary, including harvesting of metadata, and passing it to the ingestion API.

4.3.1 IMPLEMENTATION

The fetchers are running on two different machines: one for DIGIT and the other one for Arŋs. Both of them have been connected to the ingestion API.

The ingestion API has been installed on two medium-sized machines, composed of 4 cores and 16 GB RAM. Each machine has been connected to a small cluster of three Elastic machines, each composed of 2 cores and 7 GB RAM.

The ingestion API has been installed on Tomcat 8 with JDK8. Each ingestion machine has an apache2 in front of it, so the load balancers can use the port 80.

The load balancers are configured to be non-sticky.

Each ingestion API machine connects to one preview API machine which saves the document for a preview generation. This preview API machine is backed by a preview generating machine used to generate the previews. The preview generation machine is turned off until the data is validated. This allows to save some processing cycles.

PostgreSQL has been used for all API instances.

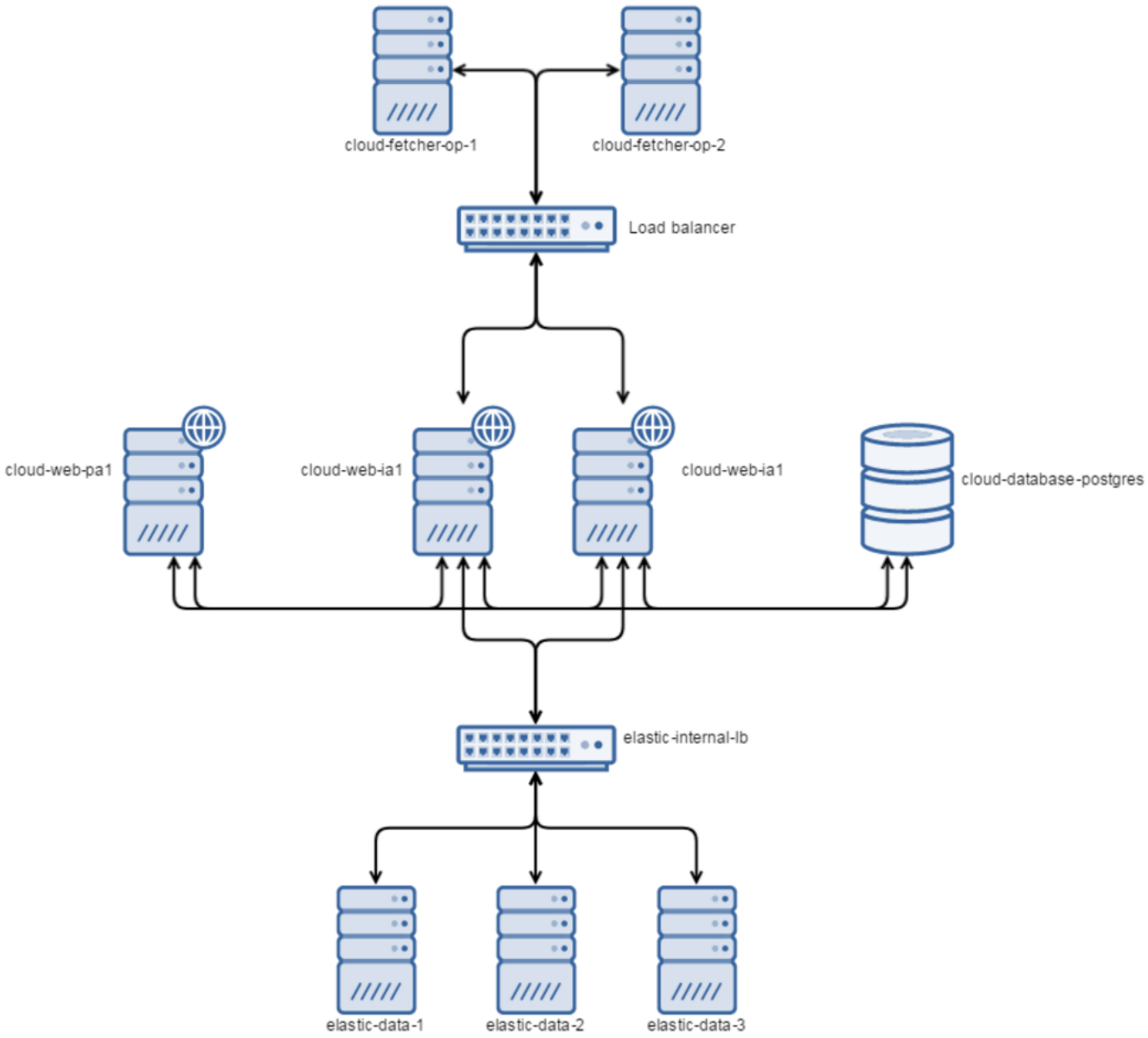

Figure 4 — Ingestion module technical architecture

4.3.2 ISSUES AND CONSTRAINTS

Non-supported languages:

A vast number of documents were not ingested because the language was either invalid or not supported by the ingestion API. The language is of vital importance for a search engine due to stemming. Adding an official language to the index is possible, however trying to ingest a document where the document is non-existent is impossible. This problem could be resolved by allowing the indexation to happen by defaulting to a default language, or even trying to guess the language by reading the content. But, by default, the used interface does not allow this behaviour.

Apache ‘Too many requests’ error:
During the ingestion, there have been several performance issues due to a relatively low speed of the machines used. This meant either having a Tomcat running out of memory, or too many sessions opened and queued on apache. The Tomcat issues were resolved by scaling the machine correctly but the Apache configuration only appeared at a later stage during the project which meant a lack of time to fix it and then re-ingest everything. It should require only a simple configuration on Apache level, to resolve this issue.

4.3.3 RECOMMENDATIONS

The following recommendations have been made for the ingestion module:

- The list of languages in the data sources needs to be known up front.
- Date field needs to be specified for sorting and filtering.
- Year field needs to be specified across all data sources for cross site filtering.
- Content Type field can be added during the ingestion in order to improve the preview generation.
- A limit of 500 MB per document needs to be introduced.
- The ingestion could use multiple threads for faster ingestion.
- Tracking ID can be used after an ingestion command in order to check, if a document has been ingested. This can happen after the full ingestion.
- The database and field configuration need to be defined before the ingestion.
- Machines used for ingesting need to have a fixed IP, so that they can be whitelisted. An IP range is also acceptable in the format 127.0.0.*.

Additionally, the necessary time needs to be foreseen for fixing the issues that appear only at the production scale level.

4.4 USER INTERFACE MODULE

4.4.1 IMPLEMENTATION

The user interface and search API were installed on one small 2 Core, 4 GB RAM machines. There is no redundancy and no fallback. There was no need for a bigger instance, since only a few people would be using it concurrently, but it can be easily scaled up should the need arise.

A Custom Public Access Logo was installed on the search user interface, which has been its only modification.
The user interface is shared by multiple clients and therefore the default landing page is not for the Public Access project but a generic search that goes across all sources.

4.4.2 ISSUES AND CONSTRAINTS

There were no major issues encountered with the user interface and the search API.

5 GOING INTO PRODUCTION — ROADMAP AND ESTIMATED RESOURCES

5.1 OVERVIEW

In order to turn the PoC into a fully-fledged application in production, first of all the scope of the application would need to be extended to all data sources covered by the Study. This means that, on one hand, new data sources (those non-covered by the PoC, see Chapter 4) would need to be added, and, on the other hand, the connectors for the already included data sources would need to be improved, in order to go beyond the selected samples and cover the full scope of the documents of the EU institutions listed at the beginning of this report.

Secondly, technical adaptations would be necessary, in terms of the infrastructure and developments needed for a real life application, in comparison to the PoC, that has been realised within the DIGIT environment. This would concern all three modules: harvesting, ingestion and the user interface.

Therefore, after the initial analysis of the remaining data sources (in section 5.2), this chapter summarises the necessary adaptations of different modules (in sections 5.3, 0 and 5.5) and, where possible, indicates the resources necessary for these adaptations (in section 5.6).

5.2 REMAINING DATA SOURCES (INITIAL ANALYSIS)

As the following data sources would also need to be covered by the harvesting module, these data sources are described in the following sections as result of only a quick initial analysis ([31]). This description mainly focuses on how the documents are exposed, what is their expected volume, what are the main available metadata and which approach (from the ones described in Chapter 3 could be used for the full and incremental harvesting of these data sources. If identified, a list of the main issues and recommendations is added for these data sources.

5.2.1 EUROPEAN COURT OF AUDITORS (ECA)

5.2.1.1 IMPLEMENTATION

After the first investigations, it has been found that documents from the ECA are available through a search form of the ECA website. The result list of the search engine is paginated:

([31]) The following data sources have not been excluded from this analysis: the European Parliament's Legislative Observatory and IPEX, the Central Archives Search Engine of the Council of the European Union, the Council database of agreements and conventions, the European Commission's Comitology register and the Register of Commission expert groups.

http://www.eca.europa.eu/en/Pages/BrowsePublications.aspx?k=&ty=&y=&top=

The foreseen solution for harvesting these documents is to navigate through all the search result pages and then parse the resulting HTML list using 'web scraping' solution (see 3.2.1.3).

For the incremental harvesting of this data source, as the ECA data source proposes an RSS feed with the last publication, this solution could be used (see 3.2.2.1). However, this strategy is mostly used for detection of new documents, and thus the fallback hash comparison method (see 3.2.2.3) could be used at the end in order to detect modifications and deletions of documents.

Estimated volume: After the first investigation, the total number of available documents could not be retrieved.
Available languages: The documents are available in all official EU languages.
Available metadata: The available metadata are limited to the title and the date of the document.

5.2.1.2 ISSUES AND CONSTRAINTS

There is no machine usable API.
The search result is not easily usable for automatic harvesting.
The number of available metadata is very limited.

5.2.1.3 RECOMMENDATIONS

A contact point at ECA is a must, as currently there is no easy solution identified for harvesting the documents of this data source. The data source owner should be encouraged to expose the documents through a web service API.

It would be good to have access to a machine-readable endpoint or an access to the documents storage folder (http://www.eca.europa.eu/Lists/ECADocuments).

5.2.2 EUROPEAN COMMITTEE OF REGION (COR)

5.2.2.1 IMPLEMENTATION

After the first investigations, it has been found that the CoR documents are available through a public 'Document Manager' search engine.

So far, three registers of documents have been identified on this data source:

- The CoR 'Opinion' documents register: https://dm.cor.europa.eu/CoRDocumentSearch/Pages/opinionssearch.aspx
- The CoR 'Public' documents register: https://dm.cor.europa.eu/CoRDocumentSearch/Pages/redsearch.aspx

- The CoR ‘Commission’ documents register:
  https://dm.cor.europa.eu/CoRDocumentSearch/Pages/comsearch.aspx

The foreseen solution for harvesting this data source is to call the search engine of each register and then parse the HTML search result list by using a ‘web scraping’ approach (see section 3.2.1.3). However, the user manual of the Document Manager is only available to registered users and it is possible that other ways of harvesting are described there.

For the incremental harvesting of this data source, it is foreseen to mix both last modification date (see section 3.2.2.1) and ‘web scraping’ (see section 3.2.1.3). Indeed, it is possible to call the search by filtering the documents by modification date; in that case the HTML search result list would only contain the newly modified documents and could be parsed by using the ‘web scraping’ method (see 3.2.1.3) like for the initial ingestion.

The CoR ‘Opinion’ documents register:
This register contains documents from 1995 till 2017; the production date of the document is available through a dedicated metadata.

Some documents are partially or fully translated; the following languages are available: ES, DA, DE, EL, EN, FR, IT, NL, PT, FI and SV.

The following document types are available in this register:

- AC;
- RES.

The CoR ‘Public’ documents register:
This register contains documents from 1995 till 2017; the production date of the document is available through a dedicated metadata.

Some documents are partially or fully translated; the following languages are available: AR, BG, BS, CS, DA, DE, EL, EN, ES, ET, FI, FR, HR, HU, IT, IW, LV, LT, ME, MT, NL, PL, PT, RO, RU, SK, SQ, SV, TR.

The following document types are available in this register:

- TCD;
- NB;
- AM;
- PV;
- AC;
- PA;
- PAC;
- CR;
- CP;
- CONV-POJ;
- CONV;

- DT;
- AMP;
- AMC;
- AMRC;
- AMRPANN;
- PSP;
- POJ.

The documents are available in several formats: Microsoft Word document, WordPerfect document and Rich Text document.

The CoR 'Commission' documents register:
This register contains documents from 1998 till 2017; the production date of the document is available through a dedicated metadata.

Some documents are partially or fully translated; the following languages are available: ES, CS, DA, DE, ET, EL, EN, FR, IT, LV, LT, HU, MT, NL, PL, PT, SK, SL, FI, SV.

The following document types are available in this register:

- COM;
- SEC;
- SWD;
- C;
- JOIN.

For the three registers of CoR, in addition to the already mentioned metadata, the following metadata are available:

- 'Title';
- 'File name';
- 'Meeting date';
- 'Rapporteur';
- 'Modified date';
- 'Rank'.

As the data source proposes an endpoint (the search engine) and as the document has a 'modified date' metadata, the 'last ingestion date' method described in section 3.2.2.1 could be used for the incremental harvesting.

Estimated volume:
The outcome of the first investigation shows that around 455 800 documents are available in the CoR data source, distributed as follows:

- Around 26 000 documents in the 'Opinions' documents register;
- Around 294 900 documents in the 'Public' documents register;
- Around 134 910 documents in the 'Register of Commission documents'.

5.2.2.2 ISSUES AND CONSTRAINTS

The documentation of the search engine being available only for the registered users ([32]), the endpoint features could not be fully analysed.

From the first analysis, the search engine is returning the result on HTML format (to be displayed in a web browser) or in partial (paginated) XML format (RSS feed).

The last modification date metadata should be confirmed as suitable to be used for the corresponding incremental method.

5.2.2.3 RECOMMENDATIONS

A contact point at the CoR is needed in order to get the credential to retrieve the Document Manager service user manual and to get further help and information on this manual.

It would be desired, if the search engine could return the documents on a 'machine readable format' (such as XML or JSON).

It would be good to confirm that the 'Modified date' metadata is correctly maintained, so that the 'last modification date' method can be used for the incremental harvesting.

5.2.3 EUROPEAN OMBUDSMAN (EO)

5.2.3.1 IMPLEMENTATION

After the first investigations, is has been found that the documents of the European Ombudsman are available on the EO website through a search engine returning an HTML paginated result list of document. The documents are retrievable as exposed on a configurable RSS feed.

So far, two registers of documents have been identified for this data source:

- The public register of documents: https://www.ombudsman.europa.eu/resources/pr/publicregistersearchpage.faces
- The cases register of documents: https://www.ombudsman.europa.eu/en/cases/home.faces

In addition to these two registers, HTML documents (i.e. pages of the website itself) are disseminated all over the website without any single endpoint to access them. A non-exhaustive list of these documents is the following:

- Press release: https://www.ombudsman.europa.eu/en/press/pressreleases.faces
- Speeches: https://www.ombudsman.europa.eu/en/activities/speeches.faces
- Annual report: https://www.ombudsman.europa.eu/en/activities/annualreports.faces

([32]) https://jsnet.eesc.europa.eu/CookieAuth.dll?GetLogon?curl=Z2FENZ2FdlZ2FitZ2FguidesZ2FDocumentsZ2FDM_user_manual_en_2_5.pdf&reason=0&formdir=29

Depending on the scope of documents that would be relevant for harvesting, three harvesting solutions are foreseen:

- If the RSS feed is usable for retrieving 'old' documents, it could be used to retrieve the data source content;
- If only the documents from the two registers are relevant, then a call to the search form and the parsing of the resulting list could be performed;
- If the documents disseminated through the website need to be harvested, then in addition to the first point solution, the 'web scraping' strategy (see 3.2.1.3) should be used.

For performing the incremental harvesting, the foreseen solution could be to use the exposed RSS feed (see section 3.2.2.2), despite the fact that it might not detect the deleted documents, only the new or modified ones. Therefore, if the RSS feed could not be used for the incremental harvesting, the 'hash incremental method' (see section 3.2.2.3) could be used as a fall-back solution.

Concerning the available metadata, the following part only focuses on the documents of the two registers, as for other documents, the metadata might not be consistent or standardised.

Public register of documents:
This register contains only the administrative or policy documents of the European Ombudsman and the information related to these two groups of documents.

It contains the documents from 2012 till 2017; the production date of the document is available through a dedicated metadata.

Some documents are partially or fully translated. However, the complete list of available languages has not been found.

The documents of this register come from three medium types: audio files, electronic files and paper (scanned documents).

The available documents types of this register are:

- Acknowledgment of receipt;
- Certificate;
- Decision;
- Document transfer;
- Email;
- Fax;
- For information only;
- Invoice;
- Letter;
- Note;
- Note telephone call;
- Order;

- Press release;
- Public procurement;
- Publication;
- Publication EO;
- Reply to request for information;
- Report;
- Report EO;
- Request for Information;
- Request for access to document;
- Request publication;
- Returned document;
- Service complaint;
- Thank you letter.

The documents of the public register are also classified regarding their initiator business unit:

- Cabinet of the European Ombudsman;
- Communication Unit;
- Directorate;
- Personnel, Administration and Budget Business Unit;
- Registry;
- Secretariat-General.

In the search result list, the following metadata are also available for the documents of this public register but these metadata are not available for all documents:

- ‘Description’;
- ‘Reference’;
- ‘Classification’;
- ‘Language of the document’.

Cases register of documents:
This register contains all the cases-related documents and information.
Some documents are partially or fully translated; the following languages are available: BG, ES, CS, DA, DE, ET, EL, EN, FR, GA, HR, IT, LV, LT, HU, MT, NL, PL, PT, RO, SK, SL, FI, SV.

The available documents types of this register are:

- Cases opened;
- Closing summaries;
- Decisions;
- Recommendations;
- Special reports;

- Case descriptions;
- Correspondence;
- Solutions.

The search result is split by document type.

Estimated volume:
The outcome of the first investigation showed that around 7 370 documents are available in the EO data source, distributed as follows:

- Around 980 documents in the ‘Public’ documents register;
- Around 6 390 documents in the ‘Cases’ documents register.

5.2.3.2 ISSUES AND CONSTRAINTS

There is no endpoint that is oriented for a ‘machine harvesting’ usage.
As described, there are two registers of documents, but a lot of other documents are disseminated all over the website without a single access point.

5.2.3.3 RECOMMENDATIONS

A contact point at the European Ombudsman would be needed in order to get explanations on the data source content and if the RSS feed could be configured for retrieving all documents available in this data source.

It would be good if the search engine could return the documents in a ‘machine readable format’ (such as XML or JSON). However, as it has been pointed out, the data source exposes its documents on a configurable RSS feed, and the usage of this feed could be an appropriate alternative to the call of the search engine.

It would be good to have an exhaustive list of the documents available on the EO website.

5.2.4 EUR-LEX

5.2.4.1 IMPLEMENTATION

Documents of the EUR-Lex data source are exposed through a search engine endpoint and on a web service API. The search endpoint is available at: http://eur-lex.europa.eu/advanced-search-form.html

The result list of the search form can be exported in several formats: TSV, CSV, Excel, PDF, Excel and XML (metadata of the document).

The web service API proposed by EUR-Lex requires a registration. The full documentation concerning the API can be found at:

- http://eur-lex.europa.eu/content/tools/webservices/SearchWebServiceUserManual_v2.00.pdf and
- http://eur-lex.europa.eu/content/tools/webservices/DataExtractionUsingWebServices-v1.00.pdf

The web service API allows registered users to get the documents metadata in XML format.
This data source contains all EU laws and numerous related documents since the early 1900s.The entire document content and metadata are stored in the CELLAR ([33]) data store.

Most of the documents are translated in all 24 EU languages (only some old documents are not translated).
In addition to the search, EUR-Lex offer a web service API to retrieve the documents and the metadata.

Available metadata:
A lot of metadata are available about the documents of EUR-Lex data source, the main ones being:

- Title;
- The CELEX id of the document (identifier in the CELLAR data store);
- The document form;
- The author;
- The date of the document;
- The domain of the document;
- The sub-domain of the document;
- The type of procedure of the document;
- The type of act of the document.

Estimated volume:
There are around 24 063 360 documents (~ 1 002 640 documents per language) available in this data source.

There are currently two foreseen solutions to harvest the content of this data source:

- Call the search engine and then the export feature in order to export the documents and to retrieve the document (the PDF export for the content and the XML export for the metadata);
- Use directly the web service API. This solution is the preferred one but it would needs to be confirmed on the basis on an in-depth analysis, if not only the metadata but also the document content could be retrieved by using the web service API.

For performing the incremental harvesting of the EUR-Lex data source, the 'last modification date' method (see section 3.2.2.1) could be used. Indeed, both ways in which the documents are exposed (the search engine or the web service API) have the feature to filter the documents by the last modification date.

5.2.4.2 ISSUES AND CONSTRAINTS

It is not certain, if the web service API could retrieve the document content or a link to it.

This data source contains a huge volume of documents and metadata (several terabytes regarding the content of the CELLAR data store), thus the expected heavy load of documents could be an issue, if not properly handled.

([33]) https://joinup.ec.europa.eu/software/cellar/description

5.2.4.3 RECOMMENDATIONS

As this data source is managed by OP, a cooperation with respective teams in charge of EUR-Lex could identify the best way of harvesting the documents relevant for the PublicAccess.eu project.

With regard to the solutions described in the section 5.3, the team of the OP Portal search could be involved to discuss their approaches and the issued encountered when harvesting the data from EUR-Lex.

5.2.5 TENDERS ELECTRONIC DAILY (TED)

5.2.5.1 IMPLEMENTATION

First investigations have revealed that the documents of the TED website are available through a search engine and through a web service API.

This data sources contains all notices that are published on a daily basis in the Supplement to the *Official Journal of the European Union* since 2012.

The web service API is the preferred solution to retrieve TED website documents but the current analysis has revealed that only the documents as such could be retrieved, not the documents' URIs. Consequently, the foreseen solution for harvesting the TED website documents would be to navigate the paginated search result pages and parse the documents metadata by using the 'web scraping' method (see section 3.2.1.3).

Available metadata:
The available metadata using the search engine solution are:

- The document number;
- The description of the document;
- The original country of the document;
- The publication date of the document;
- The deadline of the document.

The document URI could be retrieved from the document page. It is then possible, on this page, to get the document in PDF, signed PDF and XML format.

In the scope of the current pre-analysis, not all the additional metadata which could be harvested from the XML version of the document have been listed.

By using the web service, more metadata have been available in the response (in JSON format).

For the incremental harvesting, an RSS feed could be used (also proposed by the TED data source.

Estimated volume:
There are around 2 377 375 documents available on the TED website.

5.2.5.2 ISSUES AND CONSTRAINTS

The web service API is currently in a pilot phase and does not offer the full feature, as would be needed (such as for harvesting of the documents URIs).

5.2.5.3 RECOMMENDATIONS

As this data source is maintained by OP, a cooperation with respective teams in charge of TED could identify the best ways of harvesting the documents relevant for the PublicAccess.eu project.

5.2.6 EDUCATION, AUDIOVISUAL AND CULTURE EXECUTIVE AGENCY (EACEA)

5.2.6.1 IMPLEMENTATION

The first investigations have revealed that the EACEA documents are available on the EACEA website on web pages. So far, two registers of documents have been identified:

- The 'Document register': https://eacea.ec.europa.eu/about-eacea/document-register_en
- The 'Call for Tenders' register: http://eacea.ec.europa.eu/about-eacea/calls-for-tenders_en

As there is no endpoint or search engine that could be used for the harvesting, the 'web scraping' method (see section 3.2.1.3) could be used for both registers.

Concerning the incremental harvesting, the EACEA website proposes several RSS feeds (listed at http://eacea.ec.europa.eu/about-eacea/news-feeds-rss_en), however, it is not certain that these RSS feeds could be used to retrieve the last updated documents. Therefore, the currently proposed solution could be the use of the hash comparison method (see section 3.2.2.3).

Document register:
This register consists in one single web page listing some documents, with no clearly identified metadata. A possible way to retrieve some documents' metadata would be to harvest them from the PDF metadata (when available), as all documents exist in PDF format.

Some documents are partially or fully translated. However, the complete list of available languages has not been found.

The documents are classified in the following categories:

- EACEA legal background;
- EACEA activities;
- EACEA financial information;
- Calls for proposals;
- Calls for tenders;

- Audit procedure;
- Others.

Call for Tenders register:

There are two document types in this register:

- The list of contractor's documents (available since 2006);
- The call for tender's documents (available since 2007).

Available metadata:
For the list of contractor's documents, there are no further metadata available than those that could be harvested from the metadata of the PDF of the documents.

For the call for tenders' documents, the documents being presented as a result list, the following metadata have been identified:

- The title;
- The 'deadline for submission';
- The 'status';
- The 'call reference'.

Estimated volume:
There are around 210 documents available in the EACEA data source:

- Around 140 documents in the document register.
- Around 70 documents in the call for tenders register.

5.2.6.2 ISSUES AND CONSTRAINTS

There is no search engine or endpoint to access all documents.
The only way of harvesting the metadata of some documents is to harvest them from the PDF of the documents.

5.2.6.3 RECOMMENDATIONS

A contact point at the EACEA would be needed to verify that no other document than the one identified exist.

It would good to see, if the RSS feeds could be used for the incremental harvesting.

It could be recommended to EACEA to propose the documents through an endpoint which returns document metadata directly readable by a machine (to prevent the inconsistencies and empty metadata).

5.2.7 EUROPEAN UNION INTELLECTUAL PROPERTY OFFICE (EUIPO)

5.2.7.1 IMPLEMENTATION

First investigations have revealed that the documents of the EUIPO are available through search endpoints and an open data platform is used for some types of documents.

So far, three search endpoints to retrieve the documents have been identified:

- The 'eSearch plus' search endpoint: https://euipo.europa.eu/eSearch or https://euipo.europa.eu/ohimportal/en/open-data
- The 'eSearch Case Law' search endpoint: https://euipo.europa.eu/eSearchCLW
- The 'EuroLocarno' search endpoint: https://euipo.europa.eu/eurolocarno

eSearch plus:
This register is exposed in two different manners: a search endpoint and an open dataset to be downloaded. It contains information about trademarks, designs, owners, representatives and bulletins since 1996.

Both in the dataset and in the search result list of the search engine, the documents are split by the following types:

- Trademarks;
- Representatives;
- International registrations;
- Designs;
- Applicants.

The documents of this register being directly accessible in XML format, the foreseen harvesting approach would be to use the open dataset which is more machine readable than the search result list from the search endpoint. The dataset is also maintained on a daily basis, as stated in the dataset web page.

The dataset is available through downloadable ZIP archives. These archives contain either XML documents representing the documents, or pictures linked to the documents.

The XSD schema of the documents is available here: http://euipo.europa.eu/schemas/OpenData
For each type of document, available metadata are described in an Excel file downloadable at: https://euipo.europa.eu/tunnel-web/secure/webdav/guest/document_library/Documents/OpenData/OpenData-Information-Model.xlsx

Additionally, in the dataset, for each type of document, there is a 'Differential' folder. This folder seems to contain a monthly update of the dataset, and thus could be used for the incremental ingestions. However, it must be confirmed that using this 'Differentia' folder would indicate deleted documents.

eSearch Case Law:

The documents of this register are available through a search engine and are displayed in a search result list. The register contains documents about the EUIPO decisions, judgments of the General Court, Court of Justice and of national courts since 1996.

The search engine results are split by the following document types:

- Trademark decisions;
- Design decisions;
- National court judgments;
- Preliminary rulings.

The documents are not translated but documents in the following languages are available: BG, ES, CS, DA, DE, ET, EL, EN, FR, HR, IT, LV, LT, HU, MT, NL, PL, PT, RO, SK, SL, FI and SV.

EuroLocarno:

This dataset is smaller than the other two, described above. It only contains the classifications and terms for indications of products.

The documents are available in the EU official languages.

All documents are available through a search engine but all the terms are also summarised in a single PDF file per language at:
https://euipo.europa.eu/tunnel-web/secure/webdav/guest/document_library/eurolocarno/eurolocarno_en.pdf ([34])

Volume estimation:

There are around 3 662 300 documents available in the EUIPO data source:

- Around 3 482 980 documents in the ‘eSearch plus’ register.
- Around 179 300 documents in the ‘eSearch Case Law’ register.
- 24 documents (1 per EU language) in the ‘EuroLocarno’ register.

5.2.7.2 ISSUES AND CONSTRAINTS

Not all available documents of EUIPO are easily ‘machine readable’, only the ones present in the open dataset. All documents metadata from the ‘eSearch Case Law’ register need to be parsed from the HTML result list.

5.2.7.3 RECOMMENDATIONS

A contact point at the EUIPO would be needed in order to get a better understanding of the data source content. The ‘eSearch plus’ dataset is convenient for harvesting because the documents are represented as XML files.

([34]) This is the English version, but by changing the language code in URL all official EU languages are available.

The other two registers are only available in the HTML search result list on the website which needs to be parsed. Therefore, it would be good to have all documents represented as XML files.
It could be suggested to EUIPO to provide a web service API that would enable a direct retrieving of XML documents (this endpoint would be called directly, instead of having to download all ZIP files representing the dataset).

5.3 ADAPTATIONS TO THE HARVESTING MODULE

5.3.1 INTRODUCTION

Apart from extending the current scope of the harvesting module consisting in the number of the data sources covered by the application developed as the PoC, it needs to be taken into consideration that this application is currently not ready to face production constraints like for instance the scalability to support a heavier load or failover mechanism to ensure less downtime.

For this reason, Arηs has proposed three possible scenarios: one consisting in a minimal improvement of the current PoC application, while the other two would require more adaptations but bringing, potentially, many benefits in return.

5.3.2 TECHNICAL SOLUTIONS PROPOSED BY ARHS — THREE SCENARIOS

The three scenarios have been proposed within two main technical solutions:

- On one hand, a solution that continues in the direction of the applied philosophy of the PoC which requires to launch a dedicated instance of the application per data source to be indexed (see Scenario 1);
- On the other hand, by reusing with the current application mechanisms and concepts implemented in the scope of another project of the Publications Office, namely the Portal Search System. (Scenarios 2 and 3).

A general advantage of the first solution is that only minimal adaptations should be required. However, it offers no reliable scalability mechanism which could prevent it from harvesting huge data sources.
The second technical solution is considered by Arηs as optimal, because it fulfills the minimal needs (scalability and failover mechanisms) for going into production, and provides many features for futures needs.

5.3.2.1 SCENARIO 1: UPGRADING THE POC APPLICATION — INSTANCE PER DATA SOURCE

This main idea of this solution is to start a dedicated instance ([35]) of the harvesting application per data source to be indexed. Custom or monitoring could also be added for raising alerts in case of application failures (currently missing in the PoC).

([35]) For each data source, one instance of the application is done.

The following table describes the 'Pros' and 'Cons' of this scenario:

Pros	Cons
Lower costs: The adaptations to be performed in this solution are limited.	Limited scalability: As the processing of a given data source is spawn on a single instance (i.e. a single server), the only way to increase the processing is to increase the server hardware specifications (vertical scalability).
	No failover mechanism: As only a single instance of the application is running for each data source, there is no 'backup' instance that will take over in case of errors. Thus the given data source harvesting will stop until the next execution. (A kind of 'resume' mechanism could possibly be added for certain data sources, in order to prevent the full re-harvesting of a data source.)

5.3.2.2 SCENARIOS 2 AND 3: INTEGRATE THE POC WITH A STREAMING FRAMEWORK

This technical solution has been inspired by another OP project: Portal Search System (PSS). This project has implemented the fetching of the CELLAR data store (which feeds EUR-Lex website) and the indexation of the content in a search engine (based on Elasticsearch). Therefore, on a business level, it is very close to the approach of the PoC.

The main idea of this solution is to use Spring XD ([36]) streaming framework and therefore benefit of the properties of a distributed application ([37]).

The general 'Pros' and 'Cons' of this solution are presented in the following table:

Pros	Cons
High fault tolerance: The system being based on several redundant instances distributed on multiple locations, it enables to continue to process in degraded mode in case of partial failure (loss of one or several instances).	Extensive adaptations necessary: More adaptions would be needed, as the business logic of the current PoC consists in a split into several modules. However, the following facts lower the scope of these adaptations: — The current PoC implantation and the

([36]) http://projects.spring.io/spring-xd
([37]) https://en.wikipedia.org/wiki/Distributed_computing:

	proposed solution are based on the same underlying core framework (Spring framework). — The PoC has been split in several clean layers. This could ease the adaptation into several modules — The proposed streaming framework (Spring XD) has, out of the box (i.e. already available without the need to add extra modules), several embedded connectors and several modules which could be reused.
Scalability: The scalability is inherent to distributed systems. Processing of each data source is distributed on several instances. Moreover, the processing is split in several steps, and each step can be scaled independently. If an additional processing is required, additional instances could be added.	
Flexibility: The streaming framework offers a high flexibility. Each process is independent in terms of scalability and configuration.	
Reusability: Each process is composed of several modules. These modules allow to compose a process with common modules and to reuse out of the box.	
Cloud ready: All tools proposed in this solution are by nature cloud ready and highly take advantage of the cloud. This solution is also fully compliant with any traditional hosting (as the PSS project is currently running in production in OP's data centers).	

However, for now, this is only a general solution, and further consideration is required in order to develop this technical approach into a working scenario. Therefore, under this solution, Arŋs has thought of two different alternatives, representing Scenario 2 and Scenario 3:

— **SCENARIO 2: setting up from scratch a dedicated streaming runtime (based on Spring XD) and migrating the current PoC's modules into proper Spring XD modules**.

The following table describes the general 'Pros' and 'Cons' of this solution:

Pros	Cons
No dependency to the PSS project.	Setting up a complete Spring XD cluster.
Taking advantage of the experience acquired on the PSS project.	Maintenance tasks of the new set up.
	No reusability of the already harvested data sources (like EUR-Lex)

— **SCENARIO 3: integrating the application developed within the PoC as module of the PSS, so that the application can take advantage of already implemented streaming runtime platform of the PSS project**.

The following table describes the general 'Pros' and 'Cons' of this solution:

Pros	Cons
Reuse of the existing Spring XD runtime.	Dependency on the PSS project.
Taking advantage of the experience acquired on the PSS project.	
Sharing the maintenance costs.	
Re-use of the already harvested metadata from EUR-Lex.	
The application developed within the PoC will take advantage of the ongoing technological innovation foreseen for the PSS project.	

5.4 ADAPTATION OF THE INGESTION MODULE

The ingestion module of the PoC is an existing service that has been created and used in production in other DIGIT's projects. Thus, it is already ready for the production also for this project.

However, as more data sources would be added in case of industrialising the PoC into a fully-fledged production project, new databases, fields and applications would need to be added to the ingestion API. Some of these databases and applications might require specific needs that are not yet available in the ingestion API (such as implementing new field type definitions, new type of security implementation, etc.

Other adaptations could come up from completely new needs of the new search application, for example, if it is decided to add new filters (according to document types, authors, etc.) or other features.

A general issue to be pointed out and that would need to be further analysed, is the stability. To make the ingestion API production ready in full context of the new search application in production, some performance (stress) tests

would need to be performed. These tests could be performed with the knowledge of the incremental ingestions happening on a daily basis times the number of sources to be indexed.

5.5 ADAPTATION OF THE USER INTERFACE

The user search interface module of the PoC is an existing service that has been created and used in production in other DIGIT projects. Thus, it is already ready for the production also for this project.

However, in order to industrialise the PoC, further customisations of the search user interface might be needed. Namely, a branch project would have to be created, which would add extra maintenance cost. This would allow a complete customisation of the search user interface. In case of new requirements, these would need to be reflected in changes of the specifications and customisations agreed and implemented beforehand. The user interface design enables an easy customisation according to the client needs.

5.6 ESTIMATION OF RESOURCES NEEDED FOR GOING INTO PRODUCTION[38]

An estimation of resources necessary for transferring the current PoC into a fully functional search application in production has been elaborated by Arηs (harvesting module) and DIGIT (ingestion and user interface modules).

The estimations produced by Arηs have been done under the following assumptions:

- The presented tables are only informative and do not represent any commitment from Arηs.
- The estimates have been calculated on the basis of Arηs' experience gained during the elaboration of the harvesting module of the PoC.
- The estimates cover only the implementation and associated testing tasks for extending the harvesting model for the remaining data sources. These estimates have been based on duration of similar tasks already performed in the scope of current PoC.
- The estimates do not cover necessary improvements of the harvesting module for already existing data sources (extending the scope from the selected samples to the full scope of data sources) and the costs related to the technical solutions and developments necessary on the PSS side.
- If needed, the estimates expressed in working days (wdays) could be decreased depending on availability of human resources and the efficiency of an exact planning.

5.6.1 RESOURCES RELATED TO THE HARVESTING MODULE

[38] Both financial estimates and estimates expressed in working days have been removed from this public version of the report due to confidentiality reasons.

The estimate has been split into six tables:

— First two tables covering the implementation of the harvesting of the remaining data sources,

— Second two tables relating to the implementation of the technical adaptations of the harvesting module, and 3.

— Third two tables summing the global estimation.

Of each set of the tables, one expresses the estimated resources in working days, and the other one in euros.

Table 1a: Harvesting module — remaining data sources — in working days

Table 1: Remaining data sources (harvesting)			
Web scraping	**Rough duration (in wdays)**	**Endpoint**	**Rough duration (in wdays)**
ECA	…	EUR-Lex	…
COR	…	TED	…
EACEA	…	EUIPO	…
EO	…		
TOTAL	…		…

Table 1b: Harvesting module — remaining data sources — in euros

Table1: Remaining data sources (harvesting)			
Web scraping	**Estimate *(k EUR)***	**Endpoint**	**Estimate *(k EUR)***
ECA	…	EUR-Lex	…
COR	…	TED	…
EACEA	…	EUIPO	…
EO	…		
TOTAL	…		…

Table 2a: Harvesting module — implementation of the technical solution — in working days

Table 2: Technical solution (harvesting)		
Scenario	**Description**	**Rough duration (in wdays)**
1	Several JVM, as for current PoC	…
2	Using an independent copy of PSS application	…
3	Module within PSS project/application	…

Table 2b: Harvesting module — implementation of the technical solution — in euros

Table 2: Technical solution (harvesting)			
Scenario	**Description**	**Minimum estimate *(k EUR)***	**Maximum estimate *(k EUR)***
1	Several JVM, as for the current PoC	…	…
2	Using an independent copy of PSS application	…	…
3	Within PSS project/application	…	…

However, Scenario 1 is not recommended by Arns, as it could cause a high risk of bad performances for at least EUR-Lex and TED. In Scenarios 2 and 3, TED and EUR-Lex would already be covered by the Portal Search System (PSS) project. It means that there would be a dependency on the execution of this project. Concerning ECB and Council ODP, adaptations would need to be handled by the PSS.

Table 3a: Adaptations of the harvesting module — total estimated costs — in working days

Table 3: Global module (harvesting)		
Scenario	**Description**	**Rough duration (in wdays)**
1	Several JVM, as for current PoC	…
2	Using an independent copy of PSS application	…
3	Within PSS project/application	…

Table 3b: Adaptations of the harvesting module — total estimated costs — in euros

Table3: Global harvesting module			
Scenario	**Description**	**Min rough budget *(k EUR)***	**Max rough budget *(k EUR)***
1	Several JVM, as for current PoC	…	…
2	Using an independent copy of PSS application	…	…
3	Within PSS project/application	…	…

5.6.2 RESOURCES RELATED TO THE INGESTION MODULE

As explained in section 0, the ingestion API has been developed and is production ready. However, although no major costs are expected in relation to deploying it into production, certain costs would be needed for minor evolutive developments and hosting.

The scope of the minor developments has already been stated in section 0.

The costs of the hosting would depend on several factors:

- Selection of the infrastructure on the basis of number of documents and the daily number of updates of those documents in case of incremental ingestion;
- If the hosting is done in the Cloud or in the Commission Data Centre;
- Results of the complete stress tests;
- If the Elasticsearch is continued to be used (providing a high availability on the API level); in case of switching to IDOL, additional considerations would need to be made;
- All APIs used in this project would need to be added to daily integration testing.

Specifically concerning the hosting, the ingestion estimates can be based on the current state of the API and on the current states of the elastic indexes. During the initial ingestion this figure might go up to handle the initial ingestion load. This will then go down as the extra capacity is not needed anymore.

5.6.2.1 CONCERNING THE API

Of course, the more people that are using the APIs the more the cost of the search APIs will go up.

API Machines	Description	Minimum estimate in working days	Maximum estimate in working days
Search API	Search API costs will vary depending on how many queries are performed. The minimum is for two machines, the maximum is for four machines.	...	...
Ingestion API	The ingestion API will vary depending on how much ingestion takes place at the same time. Low budget estimate is closer to the daily running while max budget is closer to the initial ingestion.	...	...
Preview API	The preview API only store commands to the database. So estimates are based on ingestion queries.	...	...
Preview Generation API	The preview generation machines performance is based on how many preview you need to generate per minute. The slower the lower the cost.	...	...
Administration API/UI	The machines are used to manage the instance.	...	...
Database Machine	All the API machines need to connect to a database schema. High IO is needed for these machines. Low cost is low availability. High cost if high availability.	...	...

A big cost of the infrastructure would be the disk solution chose and how much space is required. I am unable to provide any figures for this since I do not have access to these numbers.

5.6.2.2 ELASTIC INSTANCES

The estimates are based on high availability, at least one replication across a minimum of three content nodes and two client nodes.

Each content node should be able to handle around 5 million documents. Therefore, for every additional 5 million documents above the initial 5 we need to add 2 content nodes. The current infrastructure has around 2 million documents. Therefore the estimates are based on 2 million documents being accessible and ingested.

The 'Min rough budget' is based on slower machines and the 'Max rough budget' on faster machines. The need of one or the other can only be fully determined once a performance test has been performed.

Number documents	Description	Minimum estimate	Maximum estimate
Around 5 million		EUR ...	EUR ...
Around 10 million		EUR ...	EUR ...

Otherwise, it is difficult to provide a viable estimation of the total costs.

5.6.3 USER INTERFACE

As explained in section 5.5, the user interface has been developed and is production ready. There are no major costs foreseen to deploy it into production, apart from possible customisations.

The costs of the hosting would depend on several factors:

- Selection of the infrastructure on the basis of the number of end-users expected (relatively small during this PoC phase);
- If the hosting is done in the Cloud or in the Commission Data Centre;
- Results of the complete stress tests;
- If the Elasticsearch is continued to be used (providing a high availability on the query level); in case of switching to IDOL, additional considerations would need to be made;
- In case of using a custom UI, it would need to be added to daily integration testing.

Specifically concerning the hosting, it can be noted that the user interface is a very light application, therefore most of the time, the user interface server can run on the same machine as the API server. The current recommendation is to have the same amount of Search UI servers as there are Search API servers. Therefore, there would be no extra cost involved. The only case where changes might be needed is when an extremely high end-user load is put in place. Consequently, an estimation of costs necessary to cover this aspect of hosting would be:

	Description	Minimum estimate	Maximum estimate
Search user interface	The minimum is based on running on the same machine as the search API. The maximum is based on running	EUR ...	EUR ...

	on its own machines with high availability.		

Otherwise, it is difficult to provide a viable estimation of the total costs.

6 LESSONS LEARNT

Developing the PoC has been a valuable opportunity to gain knowledge about the constraints that need to be faced on this type of projects. This chapter summarises the identified risks and suggest ways of their prevention.

6.1 RISKS

The identified risks are presented in four sub-categories:

1. Risks related to the document metadata:

- **Unstructured metadata:** the metadata may not be uniform throughout the different EU institutions (different names, different meaning and different set of metadata). As a consequence, the aggregation facet in the search user interface will not work correctly and documents with common metadata will not be aggregated together.
- **Lack of business knowledge** about the metadata of the documents of EU institutions can have for a result that not only meaningful interinstitutional metadata are harvested.
- The **inability to assess the consistency of the harvested metadata** could lead to issues in the ingested content (for example: if it is impossible to check that a publication date is correctly maintained, it cannot be guaranteed that the interface will display the correct values).
- **Multiple presences of the same documents:** EU institutions may keep in their data sources not only their own documents but also documents adopted by other EU institutions. This results in multiple occurrences of the same document and a false perception as for the origin (author) of the document.

2. Risks identified for the website data sources without endpoint:

- **Incompleteness:** the harvestable set of metadata is limited when using the web scraping strategy.
- **Inefficiency:** more effort (analysis and implementation) is necessary for identifying the metadata, where they are located and how they can be collected.
- **Scalability:** the implemented solution cannot be generic, as the harvesting scripts need to be customised for each page of a website. Moreover, if the web pages change, modification of the associated scripts might be necessary.
- **Low reliability:** for the incremental ingestion, as there is no way to know if a document has been added/modified/deleted on the website, a full harvesting of the website is needed.

3. Technical risks:

- If two different teams are working on the harvesting and ingestion modules (as it has been the case of the PoC), integration issues between these modules can be faced.

4. Risks related to the lack of cooperation from the side of the data source owners:

- Even if an EU institution, of which the data source is integrated in the search engine, gives a general approval for the harvesting, there is no guarantee that it also nominates its official contact point and

provides cooperation necessary for a good understanding of the institution's processes, documents and metadata.

- With no collaboration from the websites data source owners, it can happen that the harvesting module is banned from the website after too many requests, or that a content of a wrong library is harvested.

6.2 SUGGESTED IMPROVEMENTS

In order to prevent the identified risks, the following suggestions are made, should the project continue with the aim of turning the PoC into the search application in production:

(a) Improvements related to data source owners

- An **active involvement of the EU institutions as the data source owners** would bring:
    - A better understanding of the document metadata of each data source;
    - Information about the data sources completeness and updates;
    - A better overview of the best solutions to harvest the content from the data sources;
    - The mitigation of failure due to the workload of documents (if an expected volume of documents to harvest is known, the processing power of the application can be adapted).
- A **dedicated access to the data sources content** would result in a better quality and metadata consistency of harvesting.
- EU institutions could be proposed to expose their data sources on **'machine readable' endpoints**.
- Use of a common set of **cross institutions metadata**, for instance by mapping available metadata to common ones, could be better promoted.

(b) Technical improvements

- **Failover solutions** together with faster and more reliable ways to monitor processes of harvesting and ingestion could be introduced, so that it will not be any more necessary to restart over the harvesting or the ingestion processes when an error occurs.
- The quality of statistics on the indexed items could be improved by introducing a **system that would track down and resolve indexation issues** directly with the source owners;
- **A mechanism for an easy testing of all incremental scenarios** (new documents — updated documents — documents to be deleted) from end-to-end could be added.

7 ABBREVIATIONS AND ACRONYMS

Abbreviation	Meaning
OP	Publications Office of the European Union
EP	European Parliament
Council	Council of EU and European Council
EC	European Commission
ECB	European Central Bank
CJEU	European Court of Justice
ECA	European Court of Auditor
EESC	European Economic and Social Committee
CoR	European Committee of the Regions
EO	European Ombudsman website
EACEA	Education Audiovisual and Culture Executive Agency
ERC	European Research Council
EUIPO	European Union Intellectual Property Office
PoC	proof of concept
API	application programming interface

8 LIST OF FIGURES