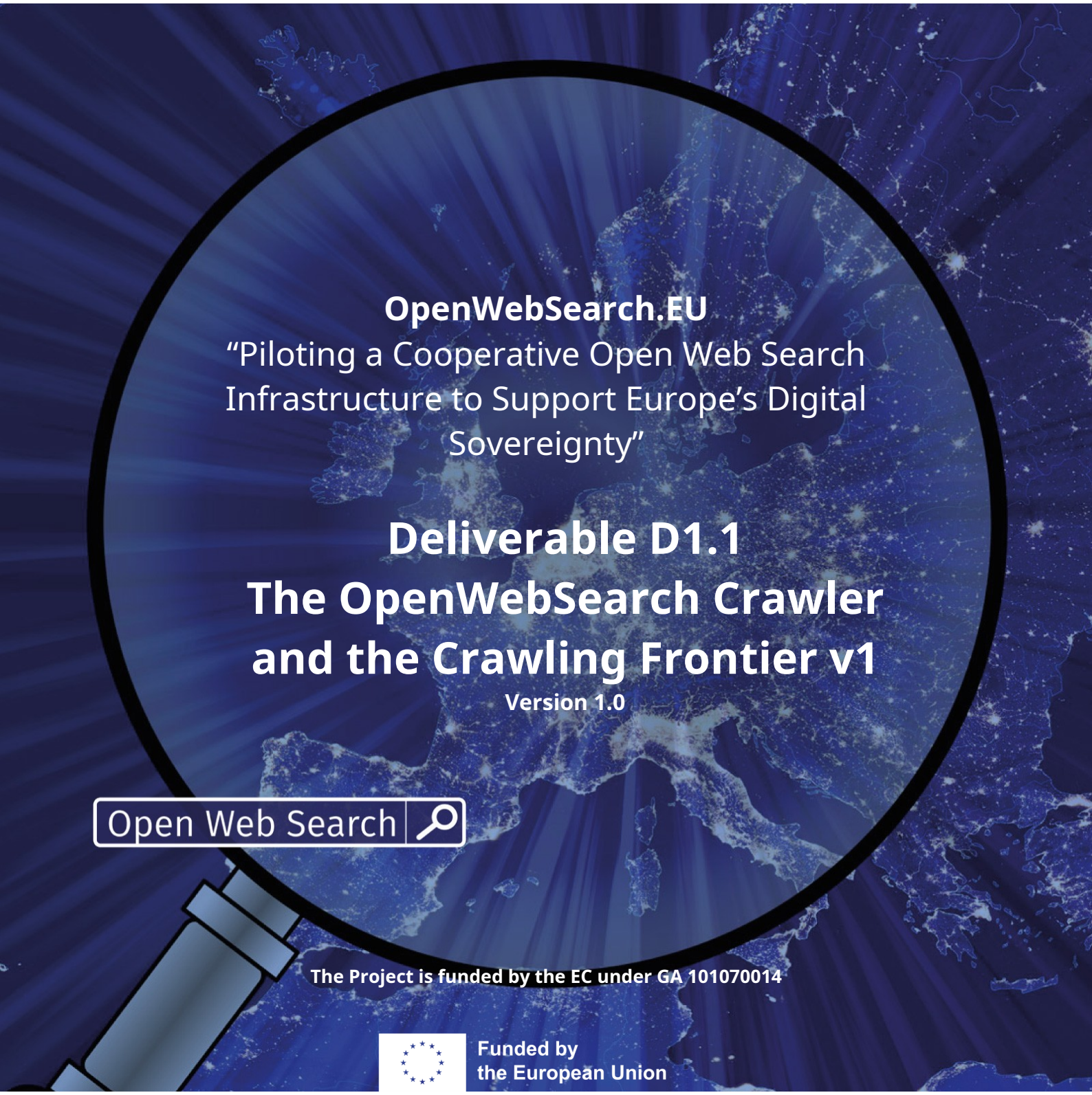


OpenWebSearch.EU

"Piloting a Cooperative Open Web Search Infrastructure to Support Europe's Digital Sovereignty"

Deliverable D1.1
The OpenWebSearch Crawler and the Crawling Frontier v1

Version 1.0

The Project is funded by the EC under GA 101070014

Funded by the European Union

NGI OPENWEBSEARCH.EU

Preliminaries

i. Project Info

Project number	101070014
Project acronym	OWS.eu
Project name	OpenWebSearch.eu – Piloting a Cooperative Open Web Search Infrastructure to Support Europe's Digital Sovereignty
Call	HORIZON-CL4-2021-HUMAN-01
Topic	HORIZON-CL4-2021-HUMAN-01-05
Type of action	HORIZON-RIA
Responsible unit	DG CNECT
Project starting date / Duration	01/09/2022 – 08/2025 (36 months)
Project reporting period	1
Project Coordinator	Prof. Dr. Michael Granitzer, University of Passau

ii. Project Partners

Acronym	Partner
UNI PASSAU	University of Passau
BADW-LRZ	Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities
RU	Radboud University
WEBIS	Leipzig University
TUGraz	Graz University of Technology
DLR	German Aerospace Center
CERN	CERN
IT4I@VSB	VSB - TECHNICAL UNIVERSITY OF OSTRAVA
BUW	Bauhaus-Universität Weimar (Associated Partner)
OSF	Open Search Foundation e.V.
A1	A1 Slovenia
SUMA-EV	Suma e.V. (Associated Partner)
CSC	CSC – Tieteen tietotekniikan keskus Oy
NLNET	Stichting NLnet

iii. Deliverable Info

Due Date / Delivery Date	M12
Deliverable Lead	UNI PASSAU
Deliverable type	DEM: Demonstrator, pilot, prototype, plan designs
Dissemination level	PU
Document Status / Version	V1.0
Work-package / Lead Partner	WP1/UP
Main authors	Saber Zerhoudi, Michael Dinzinger, Sebastian Schmidt, Gijs Hendriksen, Martin Potthast, Noor Afshan Fathima, Michael Granitzer
Approval	The deliverable expresses the opinion of the authors and has not yet been approved by the EC.
Related Documents	D 3.3 The OpenWebSearch Hub and the Open Web Index Y1 D4.2 Report of privacy, transparency, and trust models for search applications V1 Granitzer et al. Impact and development of an Open Web Index for open web search [1]

iv. Deliverable Summary

This document provides a detailed overview of the first deliverable for D1.1 "The OpenWebSearch Crawler and the Crawling Frontier", a deliverable of the OpenWebSearch.eu initiative, which is supported by the European Commission (EC) under the Horizon Europe Framework Programme grant agreement number GA 101070014. It outlines the current progress in the development and launch of two components: the Open Web Search Crawler (OWLer) and the OpenWebSearch Crawling Coordinator (OWLer-Frontier). These are reviewed in the context of their initial proposal stage, and the document also outlines the project's anticipated plans for the upcoming year.

[1] Granitzer, M., Voigt, S., Fathima, N. A., Golasowski, M., Guetl, C., Hecking, T., ... & Zerhoudi, S. (2023). Impact and development of an Open Web Index for open web search. Journal of the Association for Information Science and Technology. https://asistdl.onlinelibrary.wiley.com/doi/10.1002/asi.24818

v. Document Management

History of Changes

Name	Version	Publication date	Changes
Initial Draft	0.1	26.07.2023	Draft document structure
Revised Draft	0.2	02.08.2023	Added a few sections
Revised Draft	0.3	10.08.2023	Implementation of Sebastian's comments
Update Numbers	0.4	24.08.2023	
Finalisation	1.0	28.8.2023	

vi. Document Approver(s) and Reviewer(s)

NOTE: All Approvers are required. Records of each approver must be maintained. All reviewers in the list are considered required unless explicitly listed as "Optional".

Name	Role	Action	Date
Sebastian Schmidt / Martin Potthast	Reviewer	Added small corrections and comments	09.08.2023
Gijs Hendriksen	Reviewer	Added small corrections and comments	11.08.2023
Michael Granitzer	Reviewer	Additional comments and additions.	21.08.2023
Noor Afshan Fathima	Reviewer	Review	28.08.2023

vii. Executive Summary

This deliverable reports on the development of the crawling infrastructure for WP 1 and its current status.

The following key results have been achieved:

1. OWLer: A comprehensive web crawler system built on the robust and scalable foundation of StormCrawler running on Apache Storm. It not only emphasizes optimizing performance but also respects digital rights and guidelines set by the web community.
2. OWLer-Frontier: This acts as the nerve center of the crawling operations, a repository dictating the crawling activities' direction and pace. It's designed with scalability and efficiency at its core, allowing tailored management of URLs while being respectful to crawled websites.
3. Architecture & Infrastructure: The entire setup operates in cluster tiers, notably the Crawling Cluster Tier (CCT) for crawling and creating WARC files and the Crawler Frontier Tier (CFT) for managing and coordinating the multi-tier crawling process.
4. Statistics: As of the end of July, the project has processed approximately 475M documents, spanning over 10.6M unique domains, in 179 languages, resulting in a storage of 19.7 TiB.
5. Crawling Pipelines: Three core pipelines have been established – WARC2WARC for assimilating externally provided WARC files, Regular Crawling for explorative crawling, and Sitemap Crawling to utilize the Sitemap mechanism.
6. URL Frontier: Built on the OpenSearch-based URLFrontier framework, it facilitates a centralized, collaborative crawling mechanism. It's been upgraded to support dynamic partitioning for diverse crawling needs.
7. Monitoring and Reporting: Tools like Grafana provide operational metrics, ensuring a transparent and efficient review of the project's status and performance.

The realised software and its deployment provide all the necessary prerequisites for achieving the project results as planned.

Resources:

- Zenodo Repo: https://doi.org/10.5281/zenodo.8278932
- Source Code: https://opencode.it4i.eu/openwebsearcheu-public/owler
- Documentation: https://openwebsearcheu.pages.it4i.eu/wp1/owseu-crawler/owler/

1. Introduction

As we approach the end of the first year of the OpenWebSearch.eu project, we present an overview of our vision and the progress made thus far with the Open Web Search Crawler (OWLer) and the Crawling Frontier (OWLer-Frontier). This report highlights our strategic approach, the technical architecture, and the operational nuances of our tools, along with a recount of their evolution and ongoing enhancements for building an Open Web Index.

OWLer, a wide-ranging web crawler, functions with the objective to facilitate improved accessibility, searchability, and navigability of the Web. It operates as an ongoing development system, focused on optimizing its functionality while ensuring compliance with the digital rights and guidelines established by the web community.

Complementing OWLer in its mission, we have developed the OWLer-Frontier, a strategic element that plays a central role in managing and coordinating our crawling activities and to integrate external crawling efforts. This crucial component serves as a repository for URLs to be crawled, dictating the direction and pace of our crawling operations.

In recounting our progress, we engage in a detailed exploration of numerous aspects of our project, thereby offering an all-encompassing perspective on the strides made, lessons learned, and objectives for the future.

The document is accompanied with the following resources:

Zenodo Repo (for the source code described in this deliverable): https://doi.org/10.5281/zenodo.8278932

Source Code (including further development changes from the Zenodo Repo)**:** https://opencode.it4i.eu/openwebsearcheu-public/owler

Documentation: https://openwebsearcheu.pages.it4i.eu/wp1/owseu-crawler/owler/

1.1 Overview

We define the architecture for the full pipeline in terms of *cluster tiers*. A cluster tier is defined as *a well-defined set of services* (and corresponding software-stack) that *must* run in the same cluster / set of virtual machines. The figure below outlines a first version for cluster tiers and their interplay.

Crawling takes place on the following two cluster tiers:

- The Crawling Cluster Tier (CCT) contains a crawler responsible for crawling the Web, thereby creating WARC (Web ARChieve format)[2] files (i.e., collections of HTTP streams from the crawling process). These WARC Files are stored in an S3 Bucket that can be

[2] https://en.wikipedia.org/wiki/WARC_(file_format)

accessed by the other tiers. There can be multiple CCT, but there should be only one CCT per data center, as the CCT can scale with the number of resources provided.

- The Crawler Frontier Tier (CFT) is a singleton tier (i.e., only one instance exists across all participating data centers). It coordinates the multi-tier crawling process, keeping track of crawled URLs, access statistics and cache digest. The crawler frontier is the primary data source for the website registry, which provides the interface for interaction between webmasters and the frontier (e.g., an interface for take-down requests).

The CCT consists of the OWLer (OWS.eu crawler), a distributed web crawler built with StormCrawler[3] running on Apache Storm, at each individual data center. Aside from regular crawling functionality, the OWLer also includes a classifier for distinguishing between benign, malicious and adult content. Results are stored in the central, iRODS-based shared storage system via S3-compliant endpoints.

The CFT consists of a single OpenSearch[4] instance running at one of the data centers. It also contains a metrics server which aggregates logs from different stages and allows for (limited) statistical analysis.

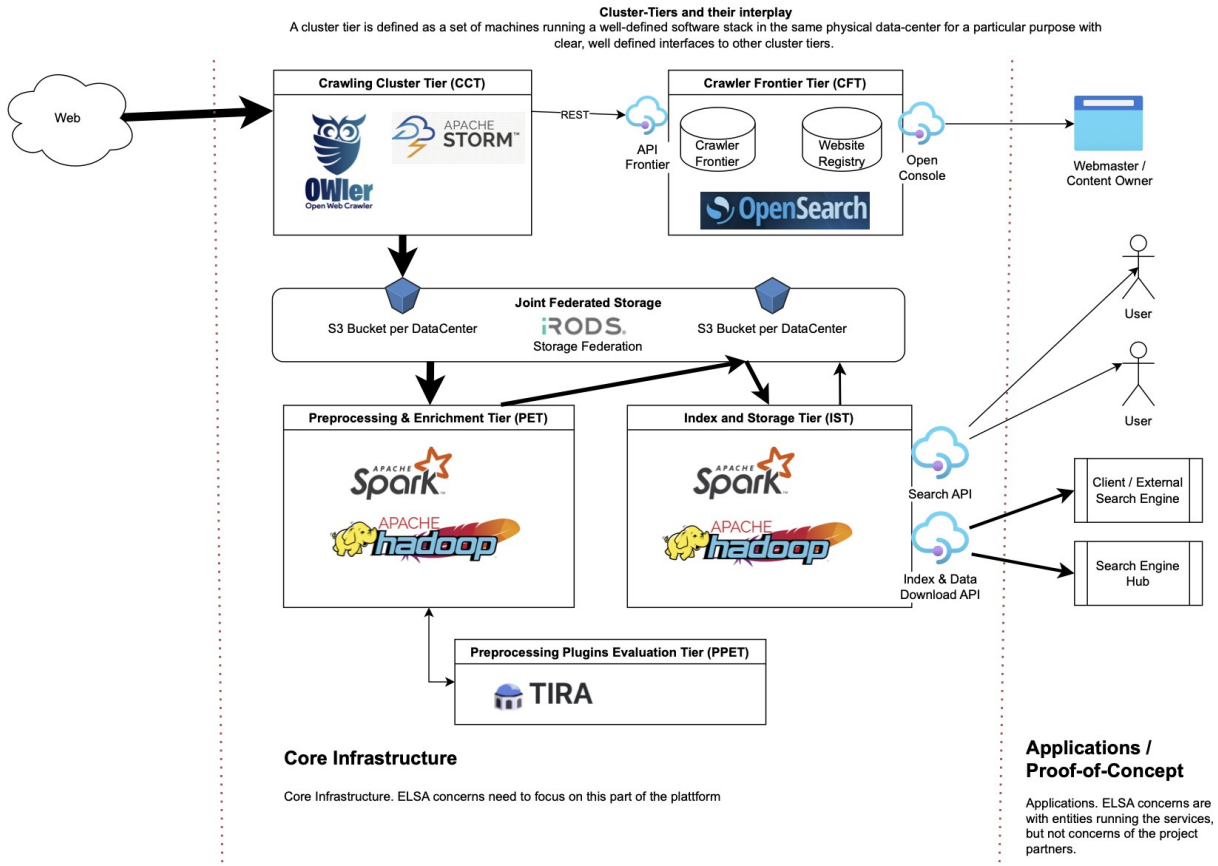


Figure 1: Overview Cluster Tiers and Interfaces for the Y1 Infrastructure.

[3] https://stormcrawler.net/

[4] https://opensearch.org/

1.2 Vision for the Open Web Search Crawler (OWLer)

1.2.1 StormCrawler

StormCrawler is an open-source software project used for web crawling that is built on Apache Storm[5]. Apache Storm, designed for real-time processing of large volumes of data, is scalable, robust, and fault-tolerant, making it ideal for web crawling tasks. StormCrawler extends this functionality, providing a collection of resources and tools to implement and deploy a scalable web crawler. It allows the extraction of a vast array of data and metadata, supports various protocols, and can be customized to handle unique use-cases.

StormCrawler's architecture involves the concept of "spouts" and "bolts." Spouts are the data sources or streams, while bolts process the input data. Together, they form a "topology," or a network of spouts and bolts working in concert to complete a task. A critical feature is its ability to scale linearly, permitting a seamless increase in crawling speed based on resources available.

1.2.2 OWLer

Deriving from the power and flexibility of StormCrawler, we developed the Open Web Crawler (OWLer). OWLer is an enhancement and a tailored adaptation, designed to accommodate the specific needs of the OpenWebSearch.eu project.

OWLer inherits the robust and scalable nature of its progenitor, and further extends the topologies with additional spouts and bolts to fit the demands of the project. To manage and optimize the crawling process, we have fine-tuned these topologies based on our requirements, creating a system that balances efficiency and respect for the websites we crawl.

1.2.3 Extending the Topologies

In OWLer, we utilize the inherent flexibility of StormCrawler's topologies to create bespoke spouts and bolts. For instance, we have developed spouts to handle various data sources like Sitemaps and externally provided WARC files. Our bolts have also been tailored for processes like parsing the HTML content, extracting hyperlinks, filtering URLs, and managing the cache system to avoid duplications.

The concept of the URL Frontier, the central repository for URLs to be crawled, has also been integrated within the OWLer's topologies. The dedicated API communicates with the URL Frontier, feeding the crawling pipeline, reporting newly discovered URLs, and updating the status of known URLs.

[5] https://storm.apache.org

All these elements are interconnected, enabling an effective, scalable, and adaptable web crawling process within OWLer. This allows us to build and operate various crawling pipelines, each addressing a different aspect of web crawling, ensuring the process is both comprehensive and respectful of the websites we crawl.

1.3. Vision for the Crawling Frontier (OWLer-Frontier)

Drawing inspiration from the URL Frontier[6] of the Crawler Commons project, we envision the Crawling Frontier (OWLer-Frontier) as a specialized extension, tailored to cater to the unique needs of the OpenWebSearch.eu project.

Our vision for OWLer-Frontier is to create a system that not only inherits the efficient and scalable characteristics of its predecessor but also expands its capabilities. We aim to design it in a way that it can manage and optimize the URL management process and plan upcoming crawls, striking a perfect balance between the speed of crawling and respect for the websites we crawl.

We foresee OWLer-Frontier as a platform that leverages the inherent modularity of the URL Frontier to develop custom components. We plan to design modules that can handle a variety of URL sources, such as external crawlers. Our vision also includes custom components for tasks like URL filtering, priority assignment, and managing the recrawl intervals to avoid overloading websites.

We envision the Crawling Frontier as the central hub for managing URLs to be crawled, seamlessly integrated within the OWLer's architecture. The dedicated API will communicate with the Crawling Frontier, managing the flow of URLs into the crawling pipeline, reporting newly discovered URLs, and updating the status of known URLs.

Our vision for OWLer-Frontier is to have all these elements intricately interconnected, facilitating an efficient, scalable, and adaptable URL management process. This will allow us to construct and operate various URL management pipelines, each addressing a different aspect of URL management, ensuring the process is both thorough and respectful of the websites we crawl.

1.4 Statistics of the Open Web Search Crawler (OWLer)

Documents crawled	20M-40M / day
Documents preprocessed	~ 475M until end of July
Number of unique crawled domains	10’649’696 until end of July
Number of unique languages	179
Terabyte crawled and stored	19.7 TiB

[6] http://urlfrontier.net

1.4.1 Top domains crawled

The following table shows the top domains crawled in the time between 12th July 2023 and 24th July 2023 to give an overview on the domain distribution in our crawl.

Domain	Fraction	Absolute Count [for 12 days]
blogspot.com	6.06%	761425
wordpress.com	4.22%	530621
wikipedia.org	2.64%	331622
europa.eu	1.11%	139832
yahoo.com	0.93%	116319
hatenablog.com	0.89%	111458
mit.edu	0.62%	78211
nih.gov	0.60%	75568
web.app	0.58%	73432
free.fr	0.57%	71916
google.com	0.55%	69052
fc2.com	0.51%	64462
harvard.edu	0.51%	64019
microsoft.com	0.45%	56230
altervista.org	0.45%	56224
uk.com	0.44%	55700
nasa.gov	0.43%	54270
libsyn.com	0.43%	53579
stanford.edu	0.42%	52404
home.blog	0.42%	52362
pinterest.com	0.41%	52016
amazonaws.com	0.41%	51201
appspot.com	0.39%	48906
berkeley.edu	0.37%	47045
mpg.de	0.37%	46842
airbnb.com	0.36%	45529
ox.ac.uk	0.33%	41425
sakura.ne.jp	0.32%	40730
alibaba.com	0.32%	39906
typepad.com	0.32%	39587
wordpress.org	0.31%	39448
wiktionary.org	0.31%	39058
nsw.gov.au	0.31%	38565
aif.ru	0.31%	38481
medium.com	0.30%	38251
cam.ac.uk	0.30%	37736
cornell.edu	0.30%	37284
googlesource.com	0.30%	37282
umich.edu	0.30%	37257
archive.org	0.29%	37020
debian.org	0.29%	36675
workplace.com	0.28%	34949
apple.com	0.28%	34754
us.com	0.27%	34526
wikidot.com	0.26%	33165

1.4.2 Data Size Crawled as of August 24th

The following tables shows the amount of data crawled in the different months stored in our central storage infrastructure, where we report in detail per storage node. The number of objects is the number of WARC files created. Note that the crawlers are limited to secure crawling of WARC files using the *warc2warc* pipeline until the network security structure is setup correspondingly.

IT4I storage infrastructure

Mai / June			July			August		
Storage	**# objects**	**Date**	**Storage**	**# objects**	**Date**	**Storage**	**# objects**	**Date**
32GiB	327	2023-05/26	129GiB	1316	2023-07/04	26GiB	262	2023-08/04
77GiB	783	2023-05/27	167GiB	1708	2023-07/05	83GiB	845	2023-08/05
65GiB	668	2023-05/28	93GiB	988	2023-07/06	82GiB	843	2023-08/06
67GiB	683	2023-05/29	355GiB	3627	2023-07/07	81GiB	831	2023-08/07
24GiB	241	2023-05/30	398GiB	4062	2023-07/08	77GiB	792	2023-08/08
163GiB	1664	2023-05/31	326GiB	3336	2023-07/09	80GiB	814	2023-08/09
427GiB	4366	2023-05	367GiB	3748	2023-07/10	82GiB	844	2023-08/10
130GiB	1332	2023-06/01	512GiB	5230	2023-07/11	83GiB	845	2023-08/11
143GiB	1462	2023-06/02	488GiB	4991	2023-07/12	82GiB	843	2023-08/12
239GiB	2439	2023-06/03	502GiB	5128	2023-07/13	83GiB	845	2023-08/13
221GiB	2262	2023-06/04	264GiB	2700	2023-07/14	113GiB	1155	2023-08/14
208GiB	2130	2023-06/05	165GiB	1686	2023-07/19	165GiB	1688	2023-08/15
200GiB	2046	2023-06/06	439GiB	4513	2023-07/20	165GiB	1685	2023-08/16
196GiB	2006	2023-06/07	636GiB	6530	2023-07/21	165GiB	1688	2023-08/17
193GiB	1972	2023-06/08	285GiB	3023	2023-07/22	165GiB	1685	2023-08/18
189GiB	1929	2023-06/09	85GiB	977	2023-07/23	165GiB	1687	2023-08/19
191GiB	1956	2023-06/10	84GiB	971	2023-07/24	165GiB	1688	2023-08/20
114GiB	1172	2023-06/11	85GiB	975	2023-07/25	165GiB	1689	2023-08/21
30GiB	312	2023-06/12	46GiB	532	2023-07/26	610GiB	6240	2023-08/22
8.9GiB	91	2023-06/15	101GiB	1031	2023-07/27	698GiB	7142	2023-08/23
43GiB	437	2023-06/16	160GiB	1634	2023-07/28	221GiB	2260	2023-08/24
701MiB	7	2023-06/21	165GiB	1688	2023-07/29			
500MiB	5	2023-06/22	164GiB	1679	2023-07/30			
600MiB	6	2023-06/26						
2.1TiB	*21564*	*2023-06*	*5.9TiB*	*62073*	*2023-07*	*3.5TiB*	*36371*	*2023-08*
Total						**12TiB**	**124374**	**Total**

LRZ storage infrastructure

Mai / June			July			August		
Storage	**# objects**	**Date**	**Storage**	**# objects**	**Date**	**Storage**	**# objects**	**Date**
35.3GiB	361	2023-05/25	1.9GiB	19	2023-07/01	3.6GiB	37	2023-08/02
59.5GiB	610	2023-05/26	174.8GiB	1788	2023-07/02	59.0GiB	604	2023-08/03
94.8GiB	971	2023-05	82.3GiB	866	2023-07/03	8.7GiB	89	2023-08/04
109.5GiB	1120	2023-06/01	140.0GiB	1449	2023-07/04	8.7GiB	89	2023-08/05
205.7GiB	2104	2023-06/02	135.1GiB	1395	2023-07/05	8.6GiB	88	2023-08/06
210.2GiB	2149	2023-06/03	164.8GiB	1693	2023-07/06	8.6GiB	88	2023-08/07
195.0GiB	1996	2023-06/04	97.1GiB	1021	2023-07/07	8.7GiB	89	2023-08/08
185.4GiB	1897	2023-06/05	119.3GiB	1238	2023-07/08	8.7GiB	89	2023-08/09
179.2GiB	1834	2023-06/06	101.3GiB	1063	2023-07/09	8.7GiB	89	2023-08/10
172.4GiB	1764	2023-06/07	86.0GiB	915	2023-07/10	8.7GiB	89	2023-08/11
171.3GiB	1753	2023-06/08	67.2GiB	694	2023-07/11	8.6GiB	88	2023-08/12
181.6GiB	1858	2023-06/09	51.8GiB	529	2023-07/12	8.7GiB	89	2023-08/13
22.6GiB	231	2023-06/10	336.9GiB	3444	2023-07/13	8.7GiB	89	2023-08/14
59.0GiB	604	2023-06/12	195.5GiB	1998	2023-07/14	8.7GiB	89	2023-08/15
91.9GiB	940	2023-06/14	4.1GiB	42	2023-07/15			
34.9GiB	357	2023-06/15	331.7GiB	3394	2023-07/16			
136.4GiB	1395	2023-06/23	170.6GiB	1746	2023-07/17			
219.9GiB	2248	2023-06/24	320.8GiB	3283	2023-07/18			
189.0GiB	1932	2023-06/25	28.4GiB	291	2023-07/19			
120.1GiB	1228	2023-06/26	183.0GiB	1935	2023-07/20			
164.2GiB	1680	2023-06/27	613.8GiB	6280	2023-07/21			
140.6GiB	1451	2023-06/28	490.6GiB	5020	2023-07/22			
117.0GiB	1227	2023-06/29	165.0GiB	1688	2023-07/23			
48.6GiB	554	2023-06/30	165.0GiB	1688	2023-07/24			
			162.2GiB	1660	2023-07/25			
			92.0GiB	941	2023-07/26			
2.9TiB	*30322*	*2023-06*	*4.5TiB*	*46080*	*2023-07*	*167GiB*	*1706*	*2023-08*
Total						**7.7TiB**	**79079**	**Total**

2. The Open Web Crawler (OWLer)

This segment offers a comprehensive outline of the OpenWebSearch.eu project, with a specific focus on Crawling and Crawling Storage. The project is centered around refining the Open Web Crawler (OWLer) – a derivative and tailored version of the StormCrawler. Integral components consist of an array of crawling pipelines and the frontier – the crawl manager for URLs awaiting crawling.

2.1 Crawling Pipelines

The efficiency of the Open Web Crawler hinges on core pipeline elements that enable proficient web crawling. Each pipeline conducts HTML parsing, hyperlink extraction, possible URL filtering, and caching checks via HTTP to ensure that the crawled page has not been duplicated within our system. It also performs content-based hashing to assign hash-based identifiers for URLs and generates status metrics and WARC files. Communication with the URL Frontier is enabled through a designated API which fills the crawling pipeline, reports newly unearthed URLs, and refreshes the status of known URLs.

In an effort to manage potential security or licensing breaches more effectively, we have instituted a risk-based categorization of crawling pipelines. The following three conceptual pipelines have been developed:

2.1.1 WARC2WARC Pipeline

This pipeline assimilates WARC files provided externally into the index and populates the frontier with identified links. This allows to continuously integrate crawling results from external crawlers, most prominently Common Crawl, into our pipelines. Functionally, it retrieves WARC files from S3-compliant storage, parses and extracts the embedded content and relevant hyperlinks, submits the extracted URLs to the frontier, and archives the WARC entries without modification. This non-fetching crawling pipeline hinges solely on already obtained web content, thus eliminating direct risks associated with unauthorized or impolite crawling. For data and high performance computing centers, this crawling mode also eliminates potential security impacts and alarms (e.g. due to accessing bot sites while crawling) as well as complaints from webmaster.

2.1.2 Regular Crawling Pipeline

Designed for explorative crawling, the regular crawling pipeline consumes URLs from the frontier, follows the general pipeline elements for URL crawling, appends the extracted URLs to the frontier, and archives new pages. Given its unrestricted scope, it carries the highest risk for security and licensing infractions. Some risks, such as inadvertent interaction with a botnet, are inevitable in the context of explorative crawling. Also, the crawling mode can face requests and/or complaints from webmasters. While we developed a corresponding web-page to inform

webmasters about our crawling activity and how to control our crawler (see https:/ows.eu/owler/ or also see the Appendix), exploratory crawling can still trigger requests from webmaster (currently handled via e-mail). Thus, this crawling mode needs a stricter governance.

2.1.3 Sitemap Crawling Pipeline

Unlike the regular pipeline that discovers new webpages traditionally via hyperlink extraction, this pipeline adopts an alternative link discovery method - Sitemaps. These are crawler-friendly tools utilized by webmasters to expedite robot access to website content. The crawler retrieves and parses Sitemaps as well as webpages discovered through Sitemaps. Sitemaps also specify changes on a website and allow for refreshing content in an efficient and timely manner. This pipeline has a lower risk profile as the retrieved webpages are anticipated to be crawled based on the Sitemap mechanism.

2.2 Implementation of the Pipelines

In our research and development of the Open Web Crawler (OWLer), we have focused on operational scalability. Our objective is to achieve a daily output of 1TB of WARC files for the first reporting period and scaling it up to 10 TB in the second funding period. Consequently, it would take around 100 days to collect 1 PB of data. However, more importantly we aim to keep the system scalable with the number of machines, such that we can move beyond the above limits if needed.

To visualize this operational context, it's pertinent to understand that one Virtual Machine (VM) corresponds to a single crawling node, capable of generating around 150-200 GB daily. Consequently, with eight VMs functioning in tandem, we can generate roughly 1.2 TB each day. Ensuring we maintain optimal productivity levels, it's indispensable that we have four productive VMs, supplemented by one experimental VM, operational at each of our infrastructure providers.

Furthermore, we've incorporated distinct 'Crawling Modes', each of which corresponds to one of the aforementioned pipelines. These modes not only allow us to organize the crawling process but also serve as vital tools to assess and manage the varying levels of risk inherent to each pipeline:

- **WARC2WARC Mode:** This mode carries the lowest level of risk, primarily because it involves recrawling previously produced WARC files. However, it presents a challenge regarding bandwidth limitation. Since all Common Crawl files are stored on AWS, it imposes a cap on our operational bandwidth. Despite this limitation, it serves as a dependable mode with a minimal risk profile. For the future we plan to integrate other

crawlers, particularly the NGI-funded SERCI Crawler[7] in order to have another external, high performance crawler running.

- **Sitemap Crawling Mode:** This mode represents a medium level of risk. It includes crawling only those webpages that webmasters have willingly added to their sitemaps, indicating their intent to have these pages crawled. As an added layer of security, we run all URLs against our database of blacklisted and opt-out URLs. Currently, this mode operates only within a closely monitored, controlled setup at our infrastructure.
- **Explorative Crawling Mode:** Representing the highest risk profile, this mode encompasses an explorative approach to crawling. However, due to the challenges we encountered during explorative crawling (see also 2.3), this mode has been put on hold until we can devise suitable solutions and strategies to manage its risk. These strategies involve specialised network IP ranges to eliminate the problem of false positive security alerts and potential IP-limited services available for academic data centers.

Each of these modes embodies a balance between the degree of exploration and the associated security implications, ensuring that our crawling operations remain efficient, effective, and respectful of digital rights.

2.3 Current operational status

As for the current operational status, the LRZ location houses four non-fetching crawlers. These are tasked with the generation of WARC files from CommonCrawl, which are subsequently stored in our S3 endpoints at both LRZ and IT4I. We are also in the process of establishing a Sitemap crawling pipeline at the University of Passau (UP).

This low-risk crawling operation will further enhance our data acquisition capabilities, whilst maintaining a low security and licensing violation footprint.

While explorative pipelines have been running during in the first weeks of July 23 and have been shown to be functionally working, we encountered security issues at the partners LRZ and IT4I. Particularly the access to botnets and IP-based access controls to licensed content (see 2.3.2) posed problems at the infrastructure partners. As of now, we develop a network concept to avoid raising many false alarms in the local and national network security department. We plan to resume exploratory crawling in September 2023, after implementing additional black and whitelisting capabilities as well as on using specialized subnets devoted for crawling. It is important to note that based on these experiences we have been able to device the risk-levels associated with the different pipelines, which will help in onboarding additional data centers.

[7] https://opencode.it4i.eu/hstein/serci-searchengine
The derivative and the publication of the Serci WebCrawler was funded through the NGI0 Discovery Fund, a fund established by NLnet with financial support from the European Commission's Next Generation Internet programme, under the aegis of DG Communications Networks, Content and Technology under grant agreement No 825322.

2.3.1 Logging and Tracking

In our aim to provide transparency and maintain accountability, we have moved our log files to a central ows-metric service hosted by CERN[8]. The services tracking the progress and monitoring the activities of our crawling operations, so logs can be accessed. For security reasons, the access is IP controlled. This transparency and accessibility ensure that we can maintain a high standard of accountability and continuous improvement in our processes.

We have developed dashboards to get insights in the crawling behaviour and the crawled websites. The following screenshot shows the dashboards on page fetches and crawled URLs. The second dashboard allows for (a limited) content-based analysis of the current crawling status, by analysing http-status codes, statistics of domains-crawled etc.

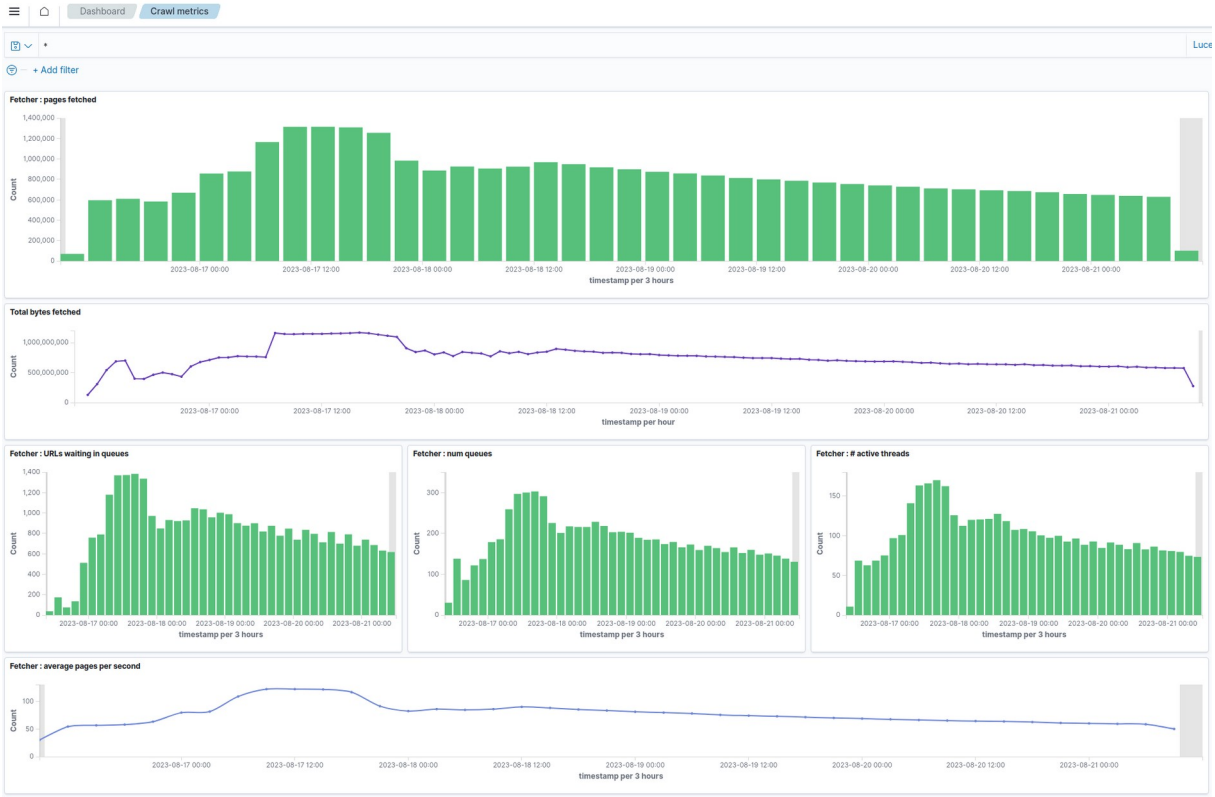


Figure 2: Screenshot of our metrics services for seeing page fetches, number of treads and queue status.

[8] http://ows-metrics.cern.ch:5601/

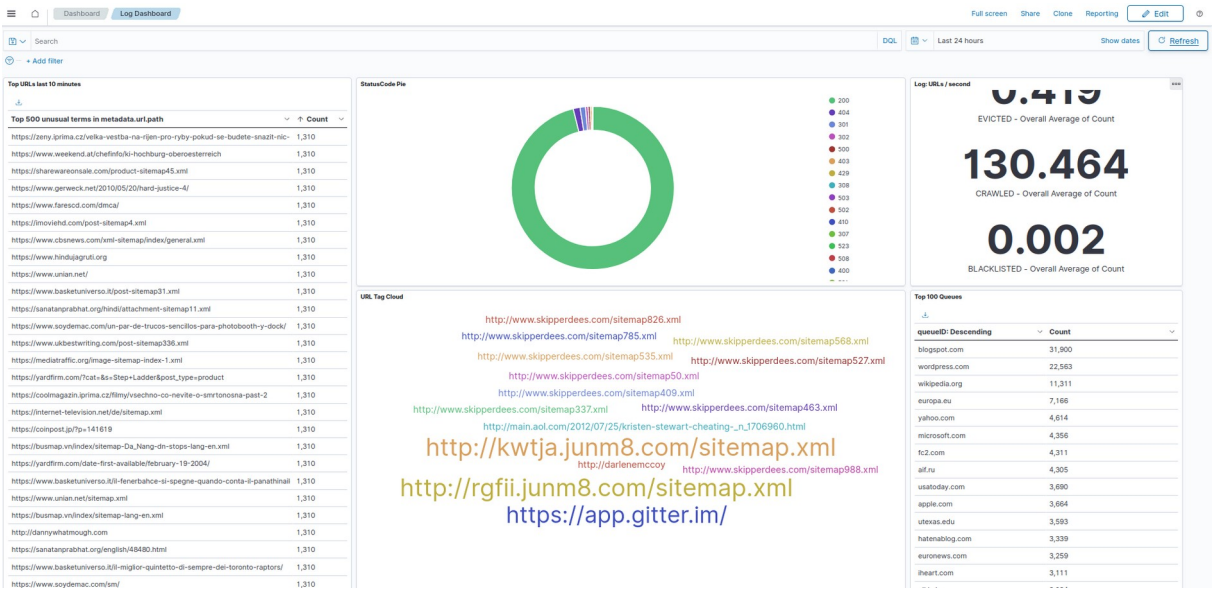


Figure 3: content-based summary of log in ows-metrics.

2.3.2 Institutional Licenses and Botnets

In the world of web crawling, institutional licenses present a potential stumbling block. Unauthorized PDF downloads, for instance, could lead to license infringements. We have committed ourselves to reducing our system's susceptibility to such license issues, as well as crawler traps and interactions with botnets, which can raise security alarms. Thus, as mentioned above, we are in the process of creating specialised crawling sub networks that allows to disable botnet monitoring from national network providers and are excluded from institutional access controlled via IP ranges.

2.3.3 The Politeness of Crawling

It is crucial for our operations to maintain a level of politeness while crawling. We ensure this by implementing a delay between fetches from the same domain/host. Despite our stringent policies, we have encountered situations where one of our crawlers was IP-blacklisted by a domain. One problem here is that acceptable delays are not standardized across websites, which requires a more fine-grained level of control, as it is planned in the open web console.

The standard framework for our crawler operates within the parameter of one request per second per domain. However, for certain sites, this rate seems impolite. To counteract this, we have increased the time window per domain to 30 seconds.

3. The Crawling Frontier

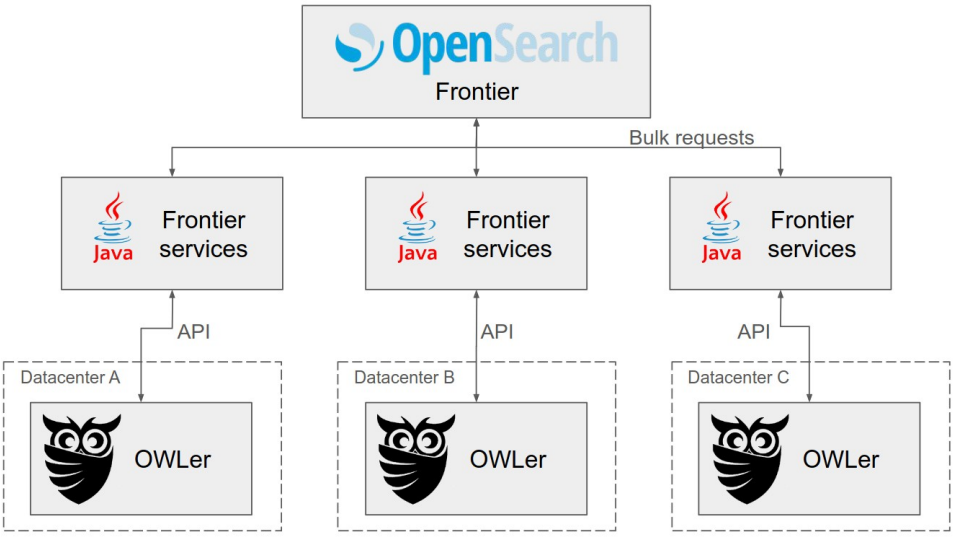


Figure 4: Role of frontier to coordinate different crawlers

The URL Frontier is a crawler component which monitors the status of both crawled and to-be-crawled URLs. Within the structure of the OWLer crawling system, it takes the central position in a star-shaped architecture. The term Frontier originally describes the data structure for storing the URLs during a crawl; in our case they are stored in an OpenSearch index. Therefore, we have built upon the existing URLFrontier framework[9] – which has been funded by NGI Zero - in the OpenSearch-based implementation[10]. This open-source software project defines an API for facilitating communication between the frontier services and the crawler nodes. The StormCrawler framework natively supports the URLFrontier programming interface, and is able to retrieve and upload URLs from the Frontier with the API's GetURLs and PutURLs command, respectively.

The figure below shows an overview over the collaborative crawling architecture, consisting of the OpenSearch backend, several frontier services and peer-to-peer crawling nodes. The OpenSearch instance, which hosts the Frontier index, is shared among several frontier services. This system design has proven valuable in maintaining a single, collaborative crawl. Each frontier service is connected to one crawler, respectively. Due to the light-weight nature of the frontier services, it is easy to start new frontier services which serve as endpoints for future crawlers. Hence, the crawl can be scaled up vertically without being interrupted and new resources can be added in a plug-&-play style manner.

The adoption of URLFrontier was motivated by the nature of our crawling system, which is both heterogeneous, highly distributed and should be open to independent crawling efforts. Within

[9] https://www.ngi.eu/funded_solution/urlfrontier/

[10] https://github.com/PresearchOfficial/opensearch-frontier

the OWS project, crawlers are located in computing sites across Europe and can join or be removed arbitrarily. The performance of the frontier services is therefore particularly crucial and should not be a bottleneck for the crawling activities. Hence, the Frontier functions as the central storage of URLs and enables peer-to-peer crawling despite the large geographic distances between the crawlers. In this context, peer-to-peer means that several workers collaborate on the same crawl. This is technically possible as the crawl space is partitioned among the frontier services. Our future contribution to the open-source community will be the enhanced implementation of the utilized partitioning method. In order to equip our crawling system for various use case scenarios, we have improved the even, hash-based partitioning of the crawl space in favor of dynamic partitioning. This allows crawlers to define their specific scope of interest and only retrieve URLs within this focus. Overall, this modification facilitates the employment of different crawling pipelines, such as explorative crawling and Sitemap crawling, in a collaborative manner.

The OWS Frontier also serves as point for monitoring the operational status of our services. We collect API request statistics via a Grafana instance, as shown in the screenshot below. Thus, we can observe the operational status of all participating crawling sites in a single dashboard. Note that the Grafana instance monitors the operational metrics for our services, while the ows-metrics service monitors the content-based metrics.

Figure 5: Grafana Dashboard for crawling sites accessing the OWS Frontier

4. Conclusion and Outlook

In this document, we have summarized our work on setting up the Crawler and the Crawling Frontier in the first year of project OpenWebSearch.EU. The technical documentation as well as the source code can be found online[11]. In the first year, we have developed a scalable solutions for crawling which collects data for the Open Web Index and ingests data to the whole indexing pipeline.

As is the case with any extensive and intricate project, the crawling operations of our Open Web Crawler (OWLer) and the URL Frontier can encounter several challenges and potential issues, including those relating to institutional licenses, politeness towards websites, and botnet interaction. While these are challenging on the technical level and associated with a considerable effort, they need to be reliably solved in order to provide a base crawling service. Consequently, we will further consider these issues in upcoming development cycles by integrating blacklists, whitelists and filters and put our development focus there before realising additional features. Furthermore, future work we will work on fine tuning the pipelines and scaling out such that the project targets can be met. This also includes the tighter integration of external crawling services. The first prototype indicates, that we will be able to meet the project targets, particularly on providing sufficient amounts of data for the whole indexing pipeline.

On the research side, we plan to tackle copyright and privacy issues for legally compliant crawling. Particularly, we plan to develop metadata standards for webmasters to express their legal and ethical constraints as well as providing a webmaster console, which is not only relevant for search but also AI. Webmaster tools and services will be important for engaging with the Web as ecosystem and to attract relevant stakeholders. To bootstrap the open webmaster console, we will also plan to analyse privacy and license statement via large language models in order to fill up the envisioned metadata standards for copyright and privacy statements.

[11] https://openwebsearcheu.pages.it4i.eu/wp1/owseu-crawler/owler/

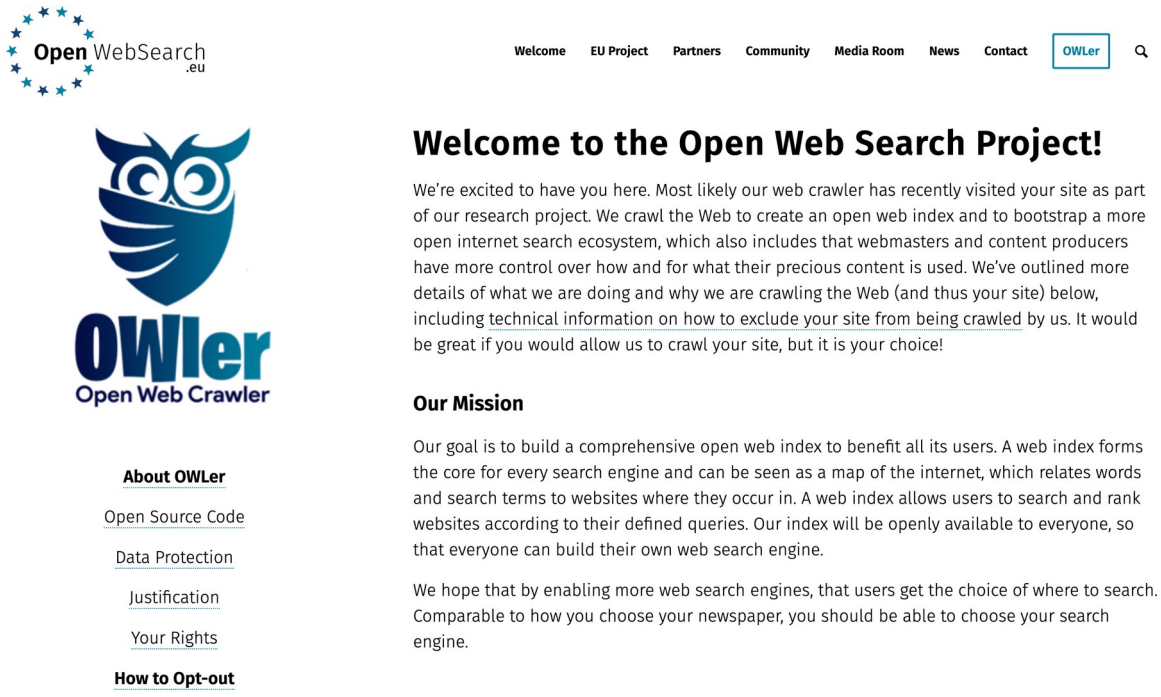
5. Appendix

5.1 Open Web Search Crawler Webpage

We have incorporated a dedicated page for website owners to understand the functionality and the purpose of our crawler. It also offers the provision to connect with us if they have any inquiries or wish to exclude their website from our crawling process.

Webpage: https://openwebsearch.eu/owler/

Content:



OWLer

OWLer – our web crawler – is a friendly explorer that strictly follows the robots.txt protocol, ensuring legal and respectful online crawling. As we're in the pioneering stages, there may be a hiccup or two along the way, and we apologize in advance for any potential inconvenience. We appreciate your understanding and are always open to feedback.

OWLer uses two main versions of our web crawler: the *Experimental version* and *Version 1*. Both versions are built on the robust Apache Storm Framework and the StormCrawler technologies, but they have distinct objectives and functionalities. Here's a brief comparison:

EXPERIMENTAL VERSION

This version is our playground for innovation. We use it primarily for testing various topologies and configurations before implementing them in our main crawler versions.

- *Topologies:* In the context of Apache Storm, a topology defines a data stream or the transformation of data within a computation. It's essentially a map of the processing logic. Our Experimental crawler tests different arrangements of spouts (sources of data) and bolts (units of processing) to find the most efficient topology.
- *Configurations:* This version also allows us to experiment with different settings to maximize our crawler's efficiency and effectiveness. For example, we might test different politeness policies, crawling speeds, or ways to handle various data types.

VERSION 1

This is the current main version of our web crawler. It includes all the stable, tested features from the Experimental version that have proven to improve the crawler's performance.

- *Stable and Reliable:* After extensive testing in the Experimental version, the features and configurations that pass our strict quality and performance standards make their way to this version.
- *Focused on Performance:* Unlike the Experimental version, which is designed for testing, Version 1 is optimized for performance. It is geared toward effectively indexing the web and providing useful, up-to-date data for the Open Web Index project.

The second version of our web crawler, planned for release next year, will continue this cycle of innovation, testing, and implementation to ensure the ongoing improvement of our indexing efforts.

You can always stay updated about our progress and learn more about our crawler versions at https://opencode.it4i.eu/openwebsearcheu-public/owler. If you have any further questions, feel free to contact us anytime.

Technical info

Open Source Code

In our endeavor to keep the internet open and accessible, we believe in transparency and collaboration. We have therefore made our web crawler's source code available to the public. You can access, review, and even contribute to the code on our Gitlab repository (https://opencode.it4i.eu/openwebsearcheu-public/owler).

Data Protection

We prioritize your privacy. While we collect data primarily related to your organization, we may also process certain personal data. Be assured, this data is always treated with the highest level of confidentiality. We only process such data when it is publicly available on your website and necessary for our project. Moreover, we don't hold onto it forever – it will be deleted after a maximum of 90 days from removal on your site.

Justification

In harmony with GDPR, our data processing is supported by one of the six legal grounds provided in the regulation, specifically protecting the legitimate interests of our project and users. We have ensured these interests do not override your fundamental rights and freedoms.

Your Rights

We're dedicated to protecting your rights. We are currently developing a platform allowing users to request access, rectification, or deletion of personal data, limit the processing, object to the processing, and even request data portability.

Opting-Out

Your control over your online presence is paramount. If you prefer to keep our web crawler from accessing your site, you can do so by updating your website's robots.txt file. Just add our user-agent identifier. Our current user-agent identifier is Owler@ows.eu/1 which represent our main crawler and Owler@ows.eu/X for our experimental crawler, and we plan to launch version 2 next year, identified by Owler@ows.eu/2. To prevent any future versions from accessing your site, simply add Owler@ows.eu/1, Owler@ows.eu/2, and Owler@ows.eu/X to the file.

Please following the step by step guideline bellow:

Guidelines for Updating Your robots.txt File

Adding our user-agent identifiers to your robots.txt file is a simple and effective way to control the access of our web crawler to your site. Here's a step-by-step guide on how to do it:

1. ACCESS YOUR WEBSITE'S ROBOTS.TXT FILE

This file is usually located in the root directory of your site. For example, if your website is

```
www.example.com, you can find the robots.txt file at

www.example.com/robots.txt.
```

2. EDIT YOUR ROBOTS.TXT FILE

Open the file with a text editor. This could be any program that lets you view and edit text files – Notepad on Windows, TextEdit on macOS, or a dedicated code editor like Sublime Text or Visual Studio Code.

3. ADD OUR USER-AGENT IDENTIFIERS

To block our current web crawler (version 1), add the following lines to your robots.txt file:

```
User-agent: Owler@ows.eu/1

Disallow: /
```

To block our experimental web crawler, add the following lines to your robots.txt file:

```
User-agent: Owler@ows.eu/X

Disallow: /
```

To also block our planned web crawler version 2, add these lines:

```
User-agent: Owler@ows.eu/2

Disallow: /
```

To block all future versions of our web crawler, add these lines:

```
User-agent: Owler@ows.eu/X

Disallow: /

User-agent: Owler@ows.eu/1

Disallow: /

User-agent: Owler@ows.eu/2

Disallow: /
```

4. SAVE YOUR CHANGES

After you've added these lines, save your robots.txt file and upload it back to your website's root directory, if necessary.

Remember: the "Disallow: /" line tells the user-agent specified not to crawl any pages on your site. If you want to block only certain pages, you can specify those pages instead of using "/". For example, "Disallow: /private" would prevent the crawler from accessing any page on your website that includes www.example.com/private.

Feel free to refer to our **GitLab repository** for any further clarification. If you have additional questions or need assistance, don't hesitate to reach out.

5.2 List of Acronyms

Acronym or Abbreviation	Term
AI	Artificial Intelligence
BMBF	Bundesministerium für Bildung und Forschung
CFT	Crawler Frontier Tier
CIFF	Common Index File Format
CCT	Crawling Cluster Tier
DEM	Demonstrator
DG Connect	European Commission Directorate-General for Communications Networks, Content and Technology
DMA	Digital Markets Act
DSA	Digital Services Act
ECJ	European Court of Justice/ D6.1, page 29, European Court of Justice (EuGH), need to be double checked!

DSM	Digital Single Market
EDIC	European Digital Infrastructure Consortium
ELSA	Ethical, Legal, and Social Aspects
EULA	End User License Agreement
FSTP	Financial Support to Third-parties
GPU	Graphics Processor Unit
GDPR	General Data Protection Regulation
HORISON-RIA	Horison - Research and Inovation Action
HPC	High-Performance Computing
IPR Management	Intellectual Property Management
IST	Index and Storage Tier
LLMs	Large Language Models
LTD	Limited (Liability company)
MEP	Member of European Parliament
MP	Member of a Parliament
NGI	Next Generation Internet
NGO	Non-Governmental Organisation
OSSYM	Open Search Symposium
OWI	Open Web Index
OWLer	Open Web Search Crawler
OWS or OWS.eu	OpenWebSearch.eu project
OWSAI	Open Web Search and Analysis Infrastructure
PCT	Project Core Team
PET	Preprocessing and Enrichment Tier
PPET	Preprocessing Plugins Evaluation Tier

RIA	Research and Innovations Action
SERPs	Search Engine Results Pages
SMEs	Small and Medium Enterprises
URL	
VM	Virtual Machine
WARC / warc /Warc	Web Archive Fileformat