IIPC Discretionary Funding Program 2019-2020

FINAL REPORT

LinkGate: Core Functionality and Future Use Cases

2021-04-18

LEAD IIPC INSTITUTION: Bibliotheca Alexandrina (BA)

2ND IIPC INSTITUTION: National Library of New Zealand (NLNZ)

OTHER INSTITUTIONS OR CONSULTANTS: n/a

PROJECT TEAM MEMBERS: Amr Morad (BA), Amr Rizq (BA), Andrea Goethals (NLNZ), Ben O'Brien (NLNZ), Lana Alsabbagh (NLNZ), Mohammed Elfarargy (BA), Mohamed Elsayed (BA), Youssef Eldakar (BA)

AWARDED FUNDING (IN USD)**:** $19,845.00 (travel dropped due to Covid-19, was $24,439.00)

PROJECT WEBSITE: https://netpreserve.org/projects/linkgate/

BRIEF ABSTRACT OF THE PROJECT:

In all domains of science, visualization is essential for deriving meaning from data. In web archiving, data is linked data that may be visualized as graph with web resources as nodes and outlinks as edges. This project's objective was to develop the core functionality of a scalable link visualization environment and to document potential research use cases within the domain of web archiving for future development.

While tools such as Gephi exist for visualizing linked data, they lack the ability to operate on data that goes beyond the typical capacity of a standalone computing device. The link visualization environment being developed works with data kept in a remote data store, enabling it to scale up to the magnitude of a web archive with tens of billions of web resources. With the data store as a separate component accessed through the network, data may be largely expanded on a server or cloud infrastructure and may also be updated live as the web archive grows.

The project was subdivided into three components: the *link service* where linked data is stored, the *link indexer* for extracting outlink information from the web archive and inserting it into the link service, and the *link visualizer* for rendering and navigating linked data retrieved through the link service.

Bibliotheca Alexandrina worked on design and development. The National Library of New Zealand worked with the researchers in IIPC institutions and the wider web archiving community to gather feedback on the idea and compile the use case inventory, and also facilitated webinars for the project through the IIPC Research Working Group.

PROJECT OUTPUTS / OUTCOMES:

The following summarizes project outputs:

(1) *link-serv* - versioned graph data service
https://github.com/arcalex/link-serv

(2) *link-indexer* - linked data collection tool
https://github.com/arcalex/link-indexer

(3) *link-viz* - web frontend for temporal graph rendering and exploration
https://github.com/arcalex/link-viz

(4) Research use cases for web archive graph visualization
https://github.com/arcalex/linkgate/wiki/Use-cases

(5) LinkGate.bibalex - LinkGate deployment with sample data
https://linkgate.bibalex.org

Source code for *link-serv*, *link-indexer*, and *link-viz* is licensed under the GNU General Public License, version 3, and available on GitHub, each in its own repository. The documented research use cases are available on GitHub on the wiki of the project's main repository. Deployment automation code is also available in the project's main repository:

https://github.com/arcalex/linkgate

link-serv

From an architectural point of view, *link-serv* (link service) is the central component that accepts data from a linked data collection tool to store and serve on demand to a temporal graph visualization frontend. Internally, *link-serv* functions as a layer between an underlying data store and clients that read and write versioned graph data. *link-serv* is written in Java and uses the Spring Boot framework. Highlights include:

- Work with two alternative graph data stores(Neo4j and ArangoDB)
- Data model to accommodate versioned graph data
- RESTful API with Gephi compatibility
- Custom cluster manager for Neo4j to scale out the community edition
- Generic design to accommodate versioned linked data in general, not just web archive data
- Data insertion concurrency support

link-indexer

Data flow begins with the *link-indexer* command-line tool processing source data files and producing partial graph data to pass to *link-serv* using the API. *link-indexer* is written in Python and uses *urlcanon*, *warcio*, and *webarchive-commons* from the web archiving tool ecosystem. Highlights include:

- Extract an archived web resource's outlinks from WAT (Web Archive Transformation) metadata files
- Directly process WARC and generate WAT on the fly
- Script-friendly logging output
- Switches to control processing behavior
- Accommodate varying network conditions using configurable parameters for request batch size and retries
- Format handling code is isolated in separate code modules to make it convenient to add support for more source data formats

link-viz

To the end user, *link-viz* (link visualizer) is the viewport to the data. *link-viz* is the user interface for rendering and exploring temporal graph data streamed via *link-serv*. *Link-viz* is a web application written in JavaScript and uses Vue.js as the web application framework and Cytoscape.js as the graph drawing library. Highlights include:

- Node locator based on address and timestamp
- Graph navigation controls
- Capture screenshots
- Node infobox
- Node highlighting based on substring matching
- Options for customizing graph appearance to user's preferences
- Incremental graph loading
- Retrieve a node's corresponding archived page through a wayback service
- Sample finders and vizors, namely, PathFinder, FlagVizor, and FaviconVizor
- Animated timeline visualization

Research use cases for web archive graph visualization

The National Library of New Zealand reached out to researchers from a wide array of backgrounds, ranging from data scientists to historians, to gather feedback on potential use cases and the types of features researchers would like to see in a web archive visualization tool. Several issues have been brought up, including frustration with existing tools' lack of scalability, being tied to a physical workstation, time wasted on preprocessing datasets, and inability to customize an existing tool to a researcher's individual needs. Gathering first-hand input from researchers has led to many interesting insights.

The documented use cases will inform future development of LinkGate beyond the term of this phase of the project. What is now implemented, through the data service (*link-serv)*, addresses the need to generally manipulate arbitrarily large linked web data extracted from a web archive. The web-based user interface (*link-viz*) is designed to be flexible with the requirements of web archive researchers in mind. Graphs are customizable, and future implementation of *vizors* will allow users to have alternative views of the visualized data (use case 2). Graphs can be animated over a given period of time and integrated with a wayback service for retrieval of corresponding archived pages. Future implementation of *finders* will further add to the ability to interact with the visualization. Continued development of the data collection tool (link-indexer) will address data preprocessing needs (use case 5). There is the potential to support Crawl logs as a source for data collection to address the interest in visualizing the web crawl process (use case 6).

linkgate.bibalex.org

An instance of *link-viz* in addition to a *link-serv* backend with sample data is deployed on BA's infrastructure and publicly accessible at linkgate.bibalex.org. A description of the deployment and Ansible configuration automation are available in the project's main repository. The process of loading data into this deployment is ongoing.

OTHER RESULTS (IF APPLICABLE):

There has been a number of dissemination activities during this phase of the project:

- LinkGate: Let's build a scalable visualization tool for web archive research (IIPC blog post)
- LinkGate project introduction and use cases (videos for 2020 IIPC General Assembly)
- LinkGate: Visualizing web archives (IIPC RSS webinar)
- LinkGate: Initial web archive graph visualization demo (IIPC blog post)
- LinkGate: Visualizing a Museum of the Web (Museum Big Data 2020 conference presentation)
- LinkGate: Visualizing linked data at scale (video on BA Information Networks YouTube channel)

The following activities are planned in the short term:

- LinkGate: Available Now (IIPC RSS webinar, to be scheduled in April 2021)
- Scalable services for web archive graph visualization (IIPC blog post, in drafting, to be published in May 2021)
- LinkGate for Web Archive Visualization (IIPC WAC 2021 conference presentation, abstract accepted, to take place in June 2021)

ANECDOTAL INFORMATION:

In the link service, adapting the underlying graph data store for versioned data by designing a suitable data model to accommodate temporal linked data from a web archive, where each version relates to an instance in time for a web resource, has been an interesting modeling exercise. A key challenge was pushing the bounds of the different parts of the solution to scale up and maintain reasonable read and write performance. Work on this component was split between two tracks, each basing the implementation on an alternative graph data store (namely, Neo4j and ArangoDB, which were initially selected as the two most suitable contenders in an early survey of the openly licensed alternatives). The scalability question remains open for further investigation and improvement beyond this phase of core functionality development.

From the perspective of the system operator of a web archive, running the process of linked data collection on the web archive storage system and processing WARC files to populate the link service with data has been a classic case of iterative software development, with additional option switches gradually added to the tool as new needs are encountered. Examples of such include excluding possibly anomalous data and on-the-fly WAT generation.

The process of developing the frontend saw several twists and turns in terms of the choice of technology and design of user interface. Alternative graph drawing libraries were experimented with (namely, D3.js, Vis.js, and finally settling for Cytoscape.js). How nodes in the graph are to be laid out was a key concern, especially as the graph becomes larger. Different algorithms were experimented with to address this. UI design of the frontend went through a sequence of ideas in an

effort to deliver a favorable experience to the user, and this aspect will certainly continue to be worked on as the frontend is put to actual use and in response to feedback from users. Developing the animated timeline visualization has been an interesting programming exercise involving graph comparison logic to determine what changes from one point in time to another.

The team hopes work done in this project will be of value to other temporal linked data applications in addition to web archiving.

BEST PRACTICES:

LinkGate is founded on modular, interoperable components, namely, the link service, the link indexer, and the link visualizer. The components leverage and extend existing APIs and data formats. This modular and interoperable design practice is reflected in the following respects:

- *link-serv* (link service) interfaces with *link-viz* (link visualizer) and *link-indexer* using an API that extends the streaming API defined by the Gephi graph visualization tool, which makes it possible, as demonstrated, to visualize data served by *link-serv* using Gephi as an alternative to the project's *link-viz* web frontend
- *link-indexer* expects source data in the WAT (Web Archive Transformation) file format that's already in adoption in quite a few places in the web archiving tool ecosystem and that's produced by preprocessing the standard WARC file format using existing tools, such as *webarchive-commons*
- The source data format handling logic in *link-indexer* is separated from the main processing logic to make it convenient for developers to extend format support (currently, besides the primary handler for the WAT format, one is also implemented for CSV as a demonstration)
- Although not yet realized in this phase of the project, implementing support for *finders* and *vizors* as add-ons to extend the visualization environment's functionality is planned
- The link service (*link-serv*) is intended as a layer between possibly alternative underlying data stores and clients that would consume the data, and support for both the ArangoDB and Neo4j graph data stores is already implemented.

PROGRAM CONTINUITY:

As initially set out, this development phase of Project LinkGate has been for the core functionality of a scalable, modular graph visualization environment for web archive data. The developers share a common passion for this work and remain committed to continuing to build up the components, specifically towards the following objectives:

- Improved scalability
- Design and development of the plugin API to support the implementation of add-on *finders* and *vizors* (graph exploration tools)
- Enriched metadata
- Integration of alternative data stores (e.g., the Solr index in SolrWayback, so that data may be served by *link-serv* to visualize in *link-viz* or Gephi)
- Improved implementation of the software in general

BA intends to maintain and expand the deploymentat linkgate.bibalex.org on a long-term basis.