CS 411W Lab II

Product Specification

Noah Jennings, Team Crystal

07/05/2020

Version 2

Table of Contents

List of Figures

List of Tables

1. Introduction

As the means of spreading knowledge increases, so does the need to store, attain, and combine academic resources (Kim, 2011). In today's world, institution-owned scholarly resources such as course content, research findings, and open source projects are predominantly electronic (Daghfous et al., 2013, pg. 639). This demonstrates a need for a technical framework where scholars can offer their resources and intellect. Proper knowledge management and group contribution procedures have allowed for more efficient, widespread intellectual communication within the academic community (Davenport, 1997). However, the aggregation or assembling of reference material, also referred to as artifacts, remains rather decentralized, discipline-specific, negligent of normalized or consistent formatting, and lacking in automation tactics (Brunelle, 2020, Kennedy, 2020). Popular resource archival solutions such as GitHub or Sharepoint give all users access to an external storage system that will hold and track most artifacts they want to archive (GitHub, 2020; Fox, 2010). Although these systems are commonly used for collaboration and management, they do not normalize the files they accept nor do they focus on all disciplines. Rather, these systems restrict memory usage and focus on a single area of expertise such as programming or coursework, forcing scholars to utilize multiple knowledge management systems (KMS) at a time (Brunelle, 2020, Kennedy, 2020).

Ideally, KMSes help collect and recollect information by archiving artifacts in a centralistic, normalized, and easily-compressed format, tracking changes applied to those artifacts, and providing cross-discipline utilization. KMSes by nature should also allow the efficient distribution and exportation of reference material in a specified format, allow artifact owners and contributors to manipulate their artifacts at will, report any changes to the owners and contributors of an artifact, and offer service automation. A^3, or "a-cubed", is an institutionally centralized, artifact archival and retrieval approach utilizing normalization, web scraping, and service automation techniques to provide the users with the ability to manipulate academic resources within a domain-specific, discipline-negligent pool of knowledge.

1.1. Purpose

A^3, or "a-cubed", is an application allowing it's users to connect with, contribute to, and manipulate a centralized well of academic materials, or artifacts. This will allow hobbyists and scholars alike to expand their respective skill sets with materials from distinguished institutions and their peers. The application is intended to be used as a means of storing, sharing, and tracking academic materials whether by those in academia or those seeking to find a way in.

A^3 will enforce access levels ranging from guests to members to administrators to testers, increasing in level of abilities. Depending on each user's access level, they will have the ability to search through the entire A^3 database for all public and authorized artifacts; however, users must enter their desired search parameters. Depending on their access level, users can export public or authorized artifacts to their local machine. A^3 will enable authenticated users to upload artifacts from their local machine. Artifacts of a specific file type such as Office Open XML document (.docx) and portable document format (.pdf), will be converted to Markdown (.md), or normalized, for more efficient storage. Each artifact's raw file attributes will be examined to further describe the artifact as well as optional, user input, descriptive tags. The system will permit users to update owned or authorized artifacts. Upon each successful update, A^3 will generate and return a change record, tracking and describing the difference between artifact states. A^3 will capture a line-by-line difference of normalized artifacts and return an enriched change record compared to artifacts that could not be normalized. A^3 will give users the ability to scrape backlinks, or URLs to a web resource, in attempts to update existing artifacts or upload new ones.

A^3 will not allow users to share or comment on artifacts directly through the interface. Users also will not be able to delete existing materials; instead, they must issue requests to remove material. A^3 will not enable users to specify a list of contributors, which would grant free reign over artifact permissions.

1.2. Scope

A^3 is envisioned as a means to contribute to a communal pool of knowledge. This would provide efficient access to and acquisition of knowledge, as well as a version tracking system. This would allow scholars to quickly expand their understanding of various tools and practices. A^3 aims to create a platform for scholars to swiftly explore different scholarly fields and contribute their knowledge to their community.

The A^3 prototype will utilize a centralized, domain-specific database and access control to create a community-driven repository of academic materials. Raw file input and limited web scraping methods will allow users to upload and update artifacts at will.

1.3. Definitions, Acronyms, and Abbreviations

Aggregate: To combine or group data from smaller pieces.

Algorithm: A set of instructions designed to perform a specific task.

Application Programming Interface (API): A set of functions and procedures allowing the creation of applications that access features of an operating system, applications, etc.

Archive: A collection of multiple files and/or folders related to a specific topic. It may be created by several different utilities and may be saved in different formats.

Artifact: A file or document. Any electronic academic resource.

Attribute: An umbrella term identifying all front and back end.

Backlink: A web resource that references or holds another resource. An artifact can have a URL backlink, signifying its original source, that holds it's daily updates outside of the A3 framework.

Centralistic Architecture: A type of knowledge management system, which is implemented in organizations and offered on the market (client-server solutions).

Cascading Style Sheet (CSS): Used to format the layout of web pages. Defines text styles, table sizes, among other things that previously could only be defined in HTML.

Database: Collection of data structures, that are organized for rapid search and retrieval.

Data Loss: An instance in which information is destroyed by corruption or neglect.

Diff: A comparison of artifact file attributes if not normalized. If normalized, artifact contents will be compared line-by-line.

Docker: A tool to create, deploy, and run applications by using “containers”. Allows developers to package up an application, with all parts needed, to be deployed in one package.

Export: Transferring data from one program or computer to another.

GitLab: Popular remote repository system to provide internal management of git repositories. It is a self-hosted Git-repository management system that keeps the user code private.

Graphical User Interface (GUI): A user interface that contains graphical elements. Examples include windows, icons, and buttons.

Hypertext Markup Language (HTML): A language used to create web pages. “Hypertext” refers to hyperlinks in a page, and “Markup language” refers to the way tags are used to define page layout.

Hyperlink: An element that links to another file or object.

Institutional Repository (IR): Scholarly institution-owned repository structure used to establish scholarly communication through various forms of digital media and spread research over the internet.

JavaScript (JS): A programming language used in web development whose syntax is influenced by Java but more similar to C.

Knowledge Management: The management process of creating, capturing, sharing, retrieving, and storing data, information, knowledge experiences, and skills by using appropriate information and network technology.

Knowledge Management Systems (KMS): Information and technology systems in support of effective and efficient knowledge management.

Markdown: A markup language that can be used to format plain text. It can be converted into another language.

Markup: A language that uses tags to define elements within a document.

MySQL: Open source SQL database management system. Developed and distributed by Oracle Corporation.

Normalization: Converting ingested objects into a small number of pre-selected formats.

Python: An interpreted, object-oriented language.

Personal Learning Environment (PLE): An interface used in flexible online courses. Designed by ODU's Center for Learning and Teaching.

Pydoc: Automatically generates documentation from Python modules. It can be presented as pages of text on the console, served to a web browser, or saved to HTML files.

Pylint: A Python static code analysis tool. Looks for programming errors and warnings from within the code, as well as from an extensive configuration file.

React: A JavaScript library that is used to create User Interfaces for web applications.

Secure File Transfer Protocol (SFTP): Secure version of File Transfer Protocol. Facilitates data access and data transfer over a Secure Shell data stream

Sphinx: A Python documentation generator. Converts reStructuredText files into HTML websites and other formats.

Tags: This is a keyword or term assigned to a piece of information.

tox: Aims to automate and standardize testing in Python. It is a generic virtual environment management and testing command-line tool.

Visual Studio Code: A source code editor that runs on Mac, Linux, and Windows.

1.4. References

Blackboard Archive Extractor. (2016, December 15) cs.odu.edu. Retrieved March 10, 2020, from https://www.cs.odu.edu/~cpi/old/411/crystals17/.

Brunelle J., personal communication, March 2, 2020.

Carroll, J., Choo, C. W., Dunlap, D., Isenhour, P., Kerr, S., MacLean, A., & Rosson, M. (2003). Knowledge Management Support for Teachers. Educational Technology Research and Development, 51(4), 42-64. www.jstor.org/stable/30221184

Daghfous, A., Belkhodja, O., & Angell, L. C. (2013). Understanding and managing knowledge loss. Journal of Knowledge Management.

Davenport, T., Long, M. & Beers, M.. (1997). Building Successful Knowledge Management Projects [Working Paper]. Retrieved March 8, 2020, from https://www.researchgate.net/publication/200045855_Building_Successful_Knowledge_Management_Projects.

Document Management Software | eFileCabinet. (2020). eFileCabinet. Retrieved February 20, 2020, from https://www.efilecabinet.com.

Doorn, P., Tjalsma, H. Introduction: archiving research data. Arch Sci 7, 1–20 (2007). https://doi.org/10.1007/s10502-007-9054-6\

Domes, S. (2017). Progressive Web Apps with React: Create lightning fast web apps with native power using React and Firebase. Packt Publishing Ltd.

File Sharing and Sync For Education, Schools and Universities - FileCloud. (2020). FileCloud. Retrieved February 20, 2020, from https://www.getfilecloud.com/file-sharing-and-sync-for-education/.

Fox, S. (2010). Beginning SharePoint 2010 Development. John Wiley & Sons. GitHub Features: The right tools for the job. (2020). GitHub. Retrieved March 10,il 2020, from https://github.com/features#team-management.

Kennedy, T. (2020, January 21). Home · Wiki · Thomas J. Kennedy / cs-roars-proposal. GitLab. Retrieved 26 April 2020, from https://git-community.cs.odu.edu/tkennedy/cs-roars-proposal/-/wikis/home.

Kennedy T. J., personal communication, February 12, 2020.

Kim, J. (2011). Motivations of faculty self-archiving in institutional repositories. The Journal of Academic Librarianship, 37(3), 246-254.

MacFarlane, J. (2006). Pandoc - About pandoc. Pandoc.org. From https://pandoc.org/index.html.

Maier, R., & Hädrich, T. (2006). Centralized versus peer-to-peer knowledge management systems. Knowledge and Process Management, 13(1), 47-61.

Nvlpubs.nist.gov. (n.d.). Glossary of Key Information Security Terms. From https://nvlpubs.nist.gov/nistpubs/ir/2013/NIST.IR.7298r2.pdf.

Tsapps.nist.gov. (2020). Data Loss Prevention. From https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=904672.

Van de Sompel, H., Nelson, M. L., Sanderson, R., Balakireva, L. L., Ainsworth, S., & Shankar, H. (2009). Memento: Time travel for the web. arXiv preprint arXiv:0911.1112.

Xie, I., & Matusiak, K. K. (2016, July 29). Digital preservation. Science Direct (255-279). Retrieved March 10, 2020, from https://www.sciencedirect.com/science/article/pii/B9780124171121000090

Zeil, S. (2019, December 26). Building the Website. cs.odu.edu. Retrieved 26 April 2020, from https://www.cs.odu.edu/~zeil/cowem/Public/buildingTheWebsite/index.html.

Zeil, S. (2020, January 21). zeil / CoWeM - Course Websites from Markdown. GitLab. From https://git-community.cs.odu.edu/zeil/Course_Website_Management.

1.5. Overview

This product specification provides the hardware and software configuration, external interfaces, capabilities, and features of the A^3 prototype. The information provided in the remaining sections of this document includes a detailed description of the hardware, software, and external interface architecture of the A^3 prototype; the key features of the prototype; the parameters that will be used to control, manage, or establish that feature; and the performance characteristics of that feature in terms of outputs, displays, and user interaction.

2. General Description

A^3 framework will encompass a command-line interface (CLI), a graphical user interface (GUI), a web scraper, and a database with repositories deployed via Docker containers. While the algorithms traverse both front and back end, the most resource-intensive processes will be handled by the back end, which includes the normalization and filtering of artifacts. This will enable the A^3 framework to provide a similar user experience regardless of customer location. Figure 1 illustrates the major functional components of the A^3 framework.

2.1. Prototype Architecture Description

Product A^3 is comprised of the following major components:

Database storage: provides a location for users to store their material.

File Transfer Protocol: provides a mechanism for users to upload and update material within the system.

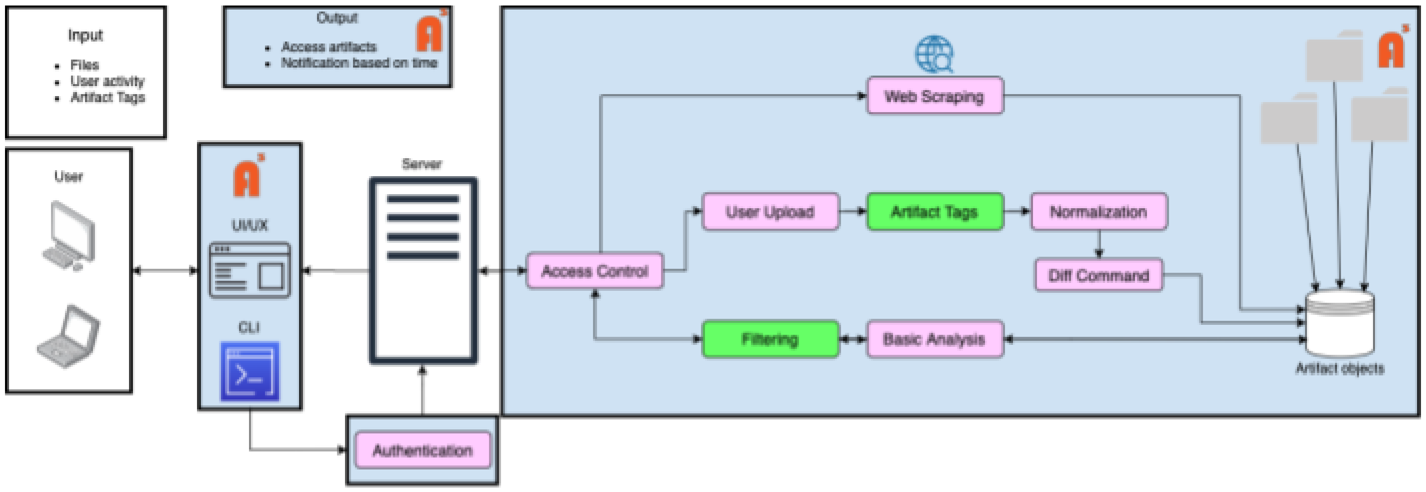


Figure 1: Major Functional Component Diagram for A³ prototype

Web Scraping System: connects users to an external interface in order to upload and update material within the system.

Normalization: allows for efficient storage and enables a more detailed file comparison.

File Comparison System: provides version tracking and in-depth material analysis.

2.2. Prototype Functional Description

Following Table 1, the A^3 prototype will encompass the following functionalities:

User Authentication: Used to identify individual users and to set and assess user access levels.

Access levels: Dictates user abilities.

Artifact Searching: Allow users to traverse the entire A^3 database for artifacts that are public or they have access to.

Resource Upload: Provides users the option to upload academic resources of their own for others to explore.

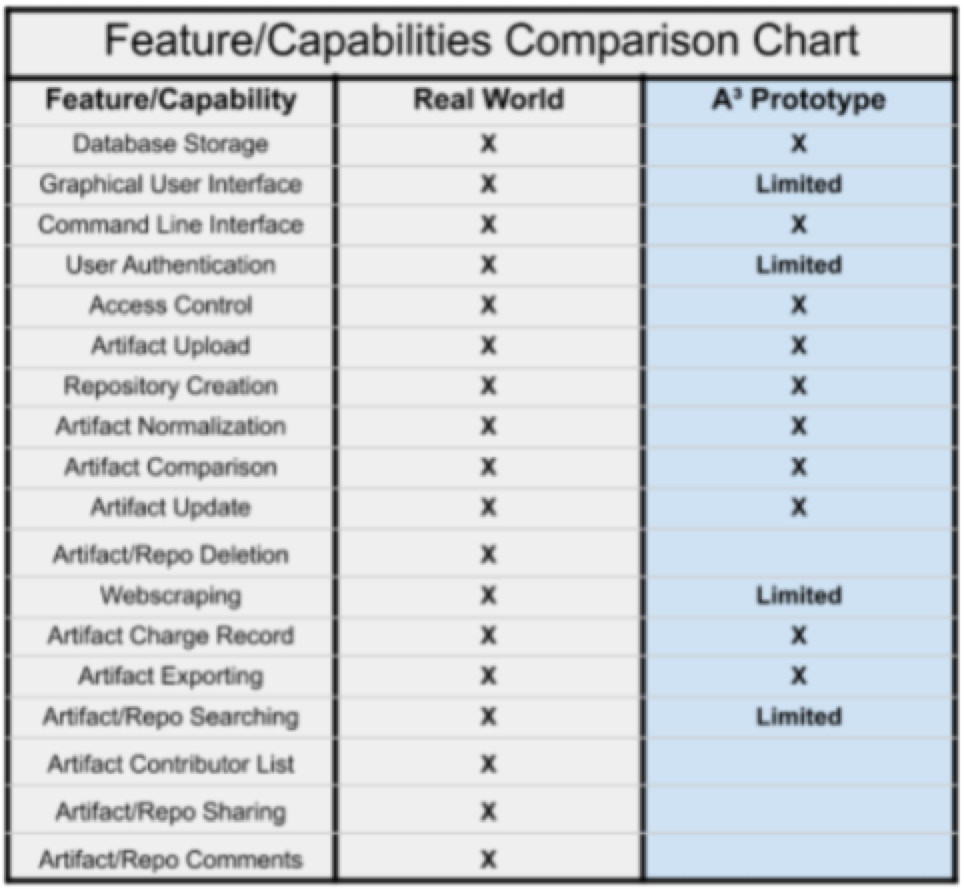
Normalization: Used to convert resources into a consistent format for storage and examination purposes.

Resource Export: Gives users the ability to export academic resources they own or have access to.

Artifact Update: Allows users to update existing resources they own or have access to through raw file transfer or web scraping.

Change Tracking: A report generated upon successful artifact update used to depict human-readable differences between artifact states.

Web Scraping: Web protocol used to locate artifacts embedded in a web page and update resources existing in the A^3 database.



Feature/Capabilities Comparison Chart

Feature/Capability	Real World	A^3 Prototype
Database Storage	X	X
Graphical User Interface	X	Limited
Command Line Interface	X	X
User Authentication	X	Limited
Access Control	X	X
Artifact Upload	X	X
Repository Creation	X	X
Artifact Normalization	X	X
Artifact Comparison	X	X
Artifact Update	X	X
Artifact/Repo Deletion	X	
Webscraping	X	Limited
Artifact Charge Record	X	X
Artifact Exporting	X	X
Artifact/Repo Searching	X	Limited
Artifact Contributor List	X	
Artifact/Repo Sharing	X	
Artifact/Repo Comments	X	

Table 1: A^3 real-world vs. prototype feature/Capability comparison chart

2.3. External Interfaces

A3 framework prototype will not interact with outside services; however, through the use of Docker-compose, docker containers, and Python frameworks, such as Flask and MySQL-connector, A3 framework will bridge a connection between client-side GUI/CLI to the A3 framework server and from the A3 framework server to the A3 framework database. Additional Python frameworks, such as BeautifulSoup4, will allow the interface and retrieval of web-hosted artifacts.

2.3.1. Hardware Interfaces

Laptop/Desktop: Laptops and Desktops are the only types of computers capable of running the A^3 application.

Internet connectivity: This is essential to the A^3 framework as it draws the bridge between the database and the user, and enables web scraping capabilities.

2.3.2. Software Interfaces

Python: A high-level, all-purpose programming language equipped with virtual environments, an array of importable packages, and the ability to interact with many other systems.

NPM: A JavaScript package manager connecting users to an online registry to share and utilize various packages. Used by the A^3 team to install and uninstall packages, as well as install and run the framework's GUI.

MySQL: A structured query language used for relational database management. Used to design the A^3 database and the tables within.

Docker: A containerization platform used to efficiently build and ship software applications. Used by the A^3 development team to build and deploy the framework's database and server functionality.

2.3.3. User Interfaces

A^3 will come equipped with a command-line interface (GUI) as well as a graphical user interface (GUI). Both requiring internet connectivity, a keyboard, a mouse, and a monitor, the CLI, and GUI will expose their users to the functionalities of the A^3 system; however, the CLI will be geared towards those more technically inclined and will have limited use of graphics.

2.3.4. Communications Protocols and Interfaces

Https: Otherwise known as Hypertext Transfer Protocol Secure, HTTPS is used for communication over a computer network in a secure, confidential manner.