A^3 framework

Aggregation and Archiving of Artifacts

Joshua S. Murphy
Old Dominion University

CS411W
Thomas J. Kennedy
Janet Brunelle
June 15, 2020

Version 2

Table of Contents

List of Figures

List of Tables

1. Introduction

Students and teachers need a framework that allows teachers to share specific knowledge and reference materials with students, without having to rely on outside sources. Currently, students and teachers lack a framework to aggregate, and archive, knowledge artifacts. *Knowledge Management Support for Teachers* (Carroll, 2003) states that Knowledge Repositories are one of the five sets of methods and tools found to help in successful implantation of knowledge management. The user lacks the ability to track changes over the lifetime of an artifact, not knowing if a certain piece of knowledge is out of date. The current aggregation used cannot be considered centralized, and accessible in one location. Artifacts are often isolated by specialization (J. Brunelle, personal communication, March 2, 2020), which makes accessing that knowledge difficult if outside of a student's specialization. A teacher may have artifacts that relate to a specialization but could also be useful to those outside of it, but not know how to share artifacts, or have a good platform with which to share said artifacts. (T. Kennedy, personal communication, February 12, 2020). Artifacts may vary in format, not allowing for the comparison of said artifacts. If a teacher leaves their position for any reason, then it is also possible that the artifacts they own are lost or abandoned entirely.

Aggregation and Archiving of Artifacts framework, from here on called A^3, is a centralized repository that can be utilized by both teachers and students to reference knowledge artifacts. A^3 will track changes, allowing the user to see when and how often an artifact has been updated. When an artifact is updated, a report will be supplied to the user, detailing what has changed in the artifact. A^3 will be deployed on a single instanced server, allowing for repositories and artifacts to be in a central location. This approach allows for abstract data, which would otherwise be locked away by specialization, to be useful to users that would otherwise

have more difficulty finding said information. A^3 will normalize artifacts, compare them, and then send a report back to the user detailing the differences between said artifacts. If an artifact is unable to be normalized, then a simple comparison of the two (e.g., file size, date created) will be done instead. Artifacts, once uploaded to the central A^3 repository, will be archived. Artifacts and repositories will be taggable, allowing for users who upload them to associate relative terms with artifacts, and allowing other users to search based on said tags.

2. Product Description

A^3 framework is a repository designed to allow teachers to upload artifacts for the use of students and other faculty for reference. It will allow for the normalization of artifacts, in accepted formats, to be compared using a diff command, which will output to the user the differences between the two artifacts. A^3 will also allow for the tagging of artifacts and repositories, to help filter when searching through the system.

2.1. Key Product Features and Capabilities

A^3 will connect the user to a single, centralized repository. The user will be able to create their own repositories, from which they can then upload and store their own artifacts. Both repositories and artifacts may be tagged with a string of characters that may then be used to filter when searching the entire repository for other artifacts or repositories. The user may also filter artifacts by the most recent modification date. The user will set up an account, consisting of a username and password, to log in to the repository. The user will be able to set their account to be public or private by default. The user can set any artifacts or repositories they own to be public or private, as well as which level a user must have to access them. Through their account, users may bookmark artifacts or repositories, to later revisit. Only the user of said bookmarks will be able to see and access these bookmarks. The owner of an artifact, if in a valid format,

may compare differences between an artifact, and an updated version of said artifact, through normalization. The two artifacts will be normalized into mark-down text, which will then be compared line by line, using the diff command. A^3 will then save a copy of the differences and display them to the user. The user may request to be notified when an artifact or repository, that they have bookmarked, is updated. The owner of an artifact or repository may set up through their account if they want to be notified after a set amount of time has passed since they have last touched or updated an artifact or repository.

2.2 Major Components

A^3 consist of major components: the front-end, which will consist of the Command Line Interface and User Interface/User Experience, from which the user will interact with A^3; the server on which A^3 will be housed; and the Database portion of A^3.

Figure 1 illustrates the major functional components of A^3. The user must first enter their username and password, and then assigned an account level which defines the permissions they have in A3. From there, the user may upload or download artifacts, manage their account settings, or tag specific artifacts and repositories (if they are the owner). Through the UI/UX and CLI, the artifacts and notifications, if set up, will be displayed to the user.

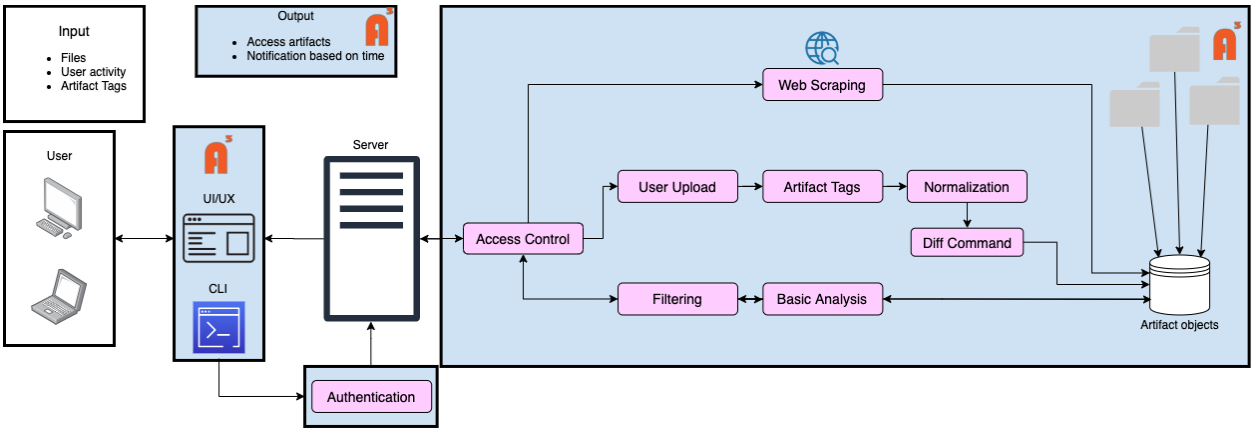

Figure 1: Major functional components of A^3

Access Control within the Database determines what the user has permission to see and do in A^3. There are five different levels:

- Guest being the default, which may only view artifacts that are made public.
- Student, which may bookmark artifacts and repositories.
- Faculty, which may upload artifacts and create repositories, and set whether they wish for them to be viewable by students or not, as well as invite students to have access to a specific artifact or repository.
- Administrator, access all artifacts on the repository, manipulate said artifacts, and manipulate user accounts on the database.
- Tester, which is only used during development of A^3 to ensure reliability.

Every role has the same capabilities as the one below it, for example every student has the same capabilities as a guest, as well as capabilities allowed to that of a student account.

The User, if permitted, may upload an artifact. They will then be required to at least add one tag to that artifact. If the artifact is in a valid format, then the artifact may be normalized. If this is an update to an existing artifact, and is able to be normalized, then a diff command may be executed and the results will be saved to a file and displayed to the user, which will contain a line by line comparison of the two different versions.

Web Scraping will be set up either to happen at a specific time and date or demand (e.g., a button). When called, the web scraper will be told a specific location to search for the artifact desired and procced to pull the resource. It will then hand the process over to update, and the diff command will be executed.

3. Identification of Case Study

A^3 is targeted towards Faculty and Students of higher academia. Faculty, specifically teachers, would be able to upload and share artifacts with other students and faculty of a university. This would allow for the sharing of knowledge that a student might not have previously but is expected for a class. Students would be able to access artifacts allowed to them, to be able to find credible information.

The specific case will involve faculty uploading artifacts to the repository, being able to create their own repositories, and being able to normalize and compare artifacts, “as well as" look at other user’s artifacts and then export them out of the system.

4. A^3 Product Prototype Description

The A^3 prototype application will consist of the Faculty and Tester roles, allowing users to upload artifacts to the database, view other artifacts, and perform basic web scraping of artifacts. The application will be running via Command Line Interface, and a basic UI/UX.

4.1. Prototype Architecture (Hardware/Software)

The A^3 prototype will be run using either Windows 10 OS or Linux OS. It will be connected, via an internet connection, to an ODU CS virtual server, where the database will be housed. The database will be running on MySQL or MongoDB. Visual Studio Code will be used as the primary IDE. Python 3.8 or newer will be used for A^3 and will be documented with pydoc and Sphinx. The GUI languages will be HTML, CSS, and JavaScript, the latter of which will use Angular and React libraries. The structure of the database and the prototype will be the same as shown in Figure 1, however, some algorithms will be condensed due to time.

4.2. Prototype Features and Capabilities

The A³ prototype will be like the real-world application, however some features will be condensed or missing due to time constraints. A list of what features will be included will be denoted by 'X', and features that will be limited, or features that will be excluded are listed in Table 1.

Feature/Capabilities Comparison Chart		
Feature/Capability	**Real World**	**A³ 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: Feature/Capabilities Comparison Chart of A³

The GUI will output a requested artifact to the user, as well as allow the user to upload an artifact, however, due to time could be faked using the CLI. User Authentication will be limited, as there will only be two roles, Faculty and Tester, in the prototype. Permission levels will allow for all users to see each other's artifacts. Authentication will use an email address and a password created for use with the prototype, as opposed to using a school or MIDAS account.

The web scraper will scrape from a pre-determined website or source. Artifact and repository searching will be limited, with pre-determined tags chosen to show how a search in A^3 would work.

4.3. Prototype Development Challenges

There are myriad challenges that will have to be overcome in the development of A^3. Some of the challenges I see are:

1. This is the first time that CS 411W is being offered in the summer, and thus will have a condensed time period of 12 weeks.
2. Successful implementation and creation of the database, as nobody on the team has ever created one from start to finish and having to learn how databases work in a short period of time.
3. A lot of data will be uploaded, downloaded, and accessed, which could result in corruption of data or loss of data. The volume of data contained in the database could result in misinterpretation of the data contained therein.
4. The workload will have to be distributed to ensure that everyone on the team makes meaningful contributions to the project in order to prevent a couple of members having to carry the project.
5. Making sure that the application is accessible to users and allowing for users to set preferences.

Despite the challenges listed above, development of A^3 will be successful.

5. Glossary

- **Aggregate**: Data that is composed of smaller pieces that form a larger whole.
- **Algorithm**: Set of instructions designed to perform a specific task.
- **Angular**: A framework for dynamic web apps. Allows for the use of HTML as a template language.
- **Application Programming Interface (API)**: Set of functions and procedures allowing the creation of applications that access features of an operating system, applications, etc.
- **Archive**: Contains multiple files and/or folders. May be created by several different utilities and may be saved in different formats.
- **Artifact**: Combination of arte, "by skill", and factum, "to make". A file or document.
- **Backlink**: A hyperlink that links from a web page, back to your own web page or website.
- **Blackboard**: A tool that allows faculty to add resources for students to access online.
- **Centralized**: Type of network where all users connect to a central server.
- **Course Websites from Markdown (CoWeM)**: A system for building course websites, including notes, slides, and organizational pages, from Markdown documents.
- **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 information, that is organized for rapid search and retrieval.
- **Data Loss**: An instance in which information is destroyed by failures or neglect.
- **Diff**: A line by line comparison of normalized artifacts.
- **Docker**: Tool to create, deploy, and run applications by using containers. Allow developers to package up an application, with all parts needed, to be deployed in one package.
- **Export**: Taking data from one program or computer to another.
- **GitLab**: Used to provide internal management of git repositories. Is a self hosted Git-repository management system that keeps the user code private.
- **Graphical User Interface (GUI)**: 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.
- **JavaScript (JS)**: A language used in web development. While influenced by Java, It's syntax is 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.
- **Markdown**: A markup language that can be used to format plain text. 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. 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.
- **reStructuredText**: A plaintext markup syntax and parser system. Useful for in-line program documentation.
- **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**: Is a keyword or term assigned to a piece of information.
- **tox**: Aims to automate and standardize testing in Python. Is a generic virtualenv management and test command line tool.
- **Visual Studio Code**: A source code editor that runs on Mac, Linux, and Windows.

6. References

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

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.

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.

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/.

GitHub Features: The right tools for the job. (2020). GitHub. Retrieved March 10, 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.

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

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

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

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.