A³ framework
Aggregation and Archiving of Artifacts

Joshua S. Murphy
Old Dominion University

CS411W
Thomas J. Kennedy
Janet Brunelle
July 7, 2020

Lab 2 - Version 2

Table of Contents

List of Figures

List of Tables

1.   Introduction

A^3 framework will allow for a centralized location for students and teachers to share artifacts to reference in an academic environment. Detailed below will be the purpose and the scope of A^3 framework.

1.1.   Purpose

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, a teacher may have artifacts isolated by specialization but are unavailable to students, or a teacher may leave a position which results in loss or abandonment of artifacts. 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 difference between two artifacts, or the difference between the two artifacts' attributes if they are not applicable for use of the diff command. A^3 will create a knowledge base that will be invaluable to faculty and students.

1.2.   Scope

The A^3 framework prototype application will exhibit how a user uploads an artifact to the repository, compares two artifacts using the diff command, outputting a report that details the differences to the user, view other artifacts not owned by them, and perform basic web scraping of an artifact to upload to the repository. The application's database will be run on MySQL using a virtual machine deployed by ODU's CS department. A^3 framework will be deployable on any system that is able to run docker, as the containers will be necessary for use of the application.

1.3.   Definitions, Acronyms, and Abbreviations

- **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.

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/.

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.

Team Crystal. (2020, June 15). Joshua Murphy, Lab 1 – A³ framework Product Description. Retrieved June 20, 2020 from https://www.cs.odu.edu/~411crystal

1.5.   Overview

The product specification details the major functional components, hardware, software, capabilities and features for the A^3 framework prototype. The remaining sections of this document contains detail descriptions of the hardware, software, and external interface architecture of the A^3 framework prototype.

2.   General Description

The A^3 framework prototype will not contain all the functions of the real-world product, due to time constrains. It will, however, be a proof of concept showing what makes A^3 framework novel. Detailed blow will be what the A^3 protype will consist of, and how it will differ.

2.1.   Prototype Architecture 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. Figure 1 illustrates the major functional components and architecture that is planned for the A^3 framework prototype.
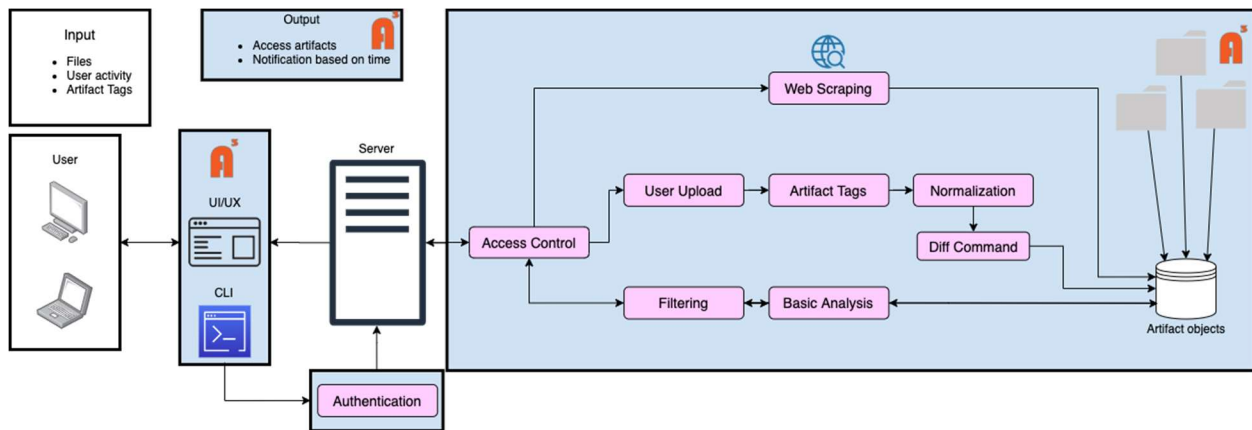

Figure 1: Major functional components of A³

2.2.   Prototype Functional Description

The A^3 prototype will be like the real-world application, however some features will be condensed or omitted 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**	**A³ Prototype Actual**
Database Storage	X	X	X
Graphical User Interface	X	Limited	Limited
Command Line Interface	X	X	X
User Authentication	X	Limited	Limited
Access Control	X	X	X
Artifact Upload	X	X	X
Repository Creation	X	X	X
Artifact Normalization	X	X	X
Artifact Comparison	X	X	X
Artifact Update	X	X	X
Artifact/Repo Deletion	X		
Web Scraping	X	Limited	Limited
Artifact Charge Record	X	X	X
Artifact Exporting	X	X	X
Artifact/Repo Searching	X	Limited	Limited
Artifact Contributor List	X		
Artifact/Repo Sharing	X		
Artifact/Repo Tags	X	Limited	Limited

Table 1: Feature/Capabilities Comparison Chart of A³

Table 1 illustrates what major functional components and architecture is planned for the A^3 framework prototype. The GUI will output a requested artifact to the user, as well as allow the user to upload an artifact, however, due to time, uploads may only be available through 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.

2.3.   External Interfaces

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. 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.

2.3.1. Hardware Interfaces

The A^3 framework prototype will be accessed and run using either a command line interface or a GUI. It will run on any operating system or computer that has docker installed to run the necessary containers used by the prototype.

2.3.2. Software Interfaces

The A^3 framework prototype will be utilizing docker and docker compose to run containerized versions of the database which will be using MySQL. The GUI languages will be HTML, CSS, and JavaScript, the latter of which will use Angular and React libraries.

2.3.3. User Interfaces

The A^3 framework prototype will contain both a GUI and CLI for users to interface with. The CLI and GUI will allow all the same functions, however the GUI will be accessible for those uncomfortable or unfamiliar with CLI commands. Both of these interfaces will allow users to access their account by supplying a username and password, upload artifacts to the repository, view other artifacts in the repository, web scrape a page to save to the repository, and make use of a diff command for comparison of artifacts.

2.3.4. Communications Protocols and Interfaces

- The A^3 framework prototype will utilize a web scraper that will, upon command, go to a specified location, scrape the page, and save it into file on the repository.

- The A^3 framework prototype will utilize a custom build REST API that is detailed in section 3.

- The A^3 framework prototype will utilize a MySQL database connector in python for the use of creating and using the database.