Lab 2 – CLASH Prototype Product Specification

Erich O’Saben

CS 411W

Professor Price and Professor Brunelle

March 24, 2015

Version 1

Table of Contents

[**1. Introduction 2**](#_Toc415043211)

[**1.1 Purpose 4**](#_Toc415043212)

[**1.2 Scope 5**](#_Toc415043213)

[**1.3 Definitions, Acronyms and Abbreviations 7**](#_Toc415043214)

[**1.4 References 9**](#_Toc415043215)

[**1.5 Overview 9**](#_Toc415043216)

[**2. General Description 10**](#_Toc415043217)

[**2.1 Major Components (Hardware/Software) 10**](#_Toc415043218)

[**2.2 Prototype Functional Description 11**](#_Toc415043219)

[**2.3 External Interfaces 13**](#_Toc415043220)

List of Figures

[Figure 1. Prototype Major Functional Component Diagram 10](#_Toc415002301)

[Figure 2. CLASH Process Flow 12](#_Toc415002302)

List of Tables

[Table 1. Real World Product vs Prototype Continued 8](#_Toc415042107)

[Table 2. User Interface Views 15](#_Toc415042108)

1. **Introduction**

During the 2013-2014 academic year, the United States welcomed nearly 900,000 international students to its colleges and universities around the country. This is the largest quantity of international students ever enrolled in U.S. secondary education in a single school year (Haynie, 2014). While these numbers are record breaking, they are not surprising. Over the past 15 years, enrollment of international students into U.S. institutions has grown 72% and there is no sign of this number decreasing in the near future (Open Doors, 2014). International students are not only important to the diversity of a university, but they are also important to national and local economies. Between 2013 and 2014, international students contributed $27 billion to the U.S. economy (Open Doors, 2014), of which Virginia’s economy received $487.5 million (Institute of International Education, 2014).

Most traditional majors and curriculums have found innovative ways to leverage technology in the classroom. Some curriculums, such as ESL, still struggle to find ways to improve the student experience and help students be successful in their courses. Current software solutions designed to help students learn parts of speech (POS) or increase reading speed continually fail English Language Learners (ELLs) because they are not the primary intended audience of the tools currently on the market. For example, many current software tools that assist with learning POS are geared towards children, making the games childish and inappropriate for adults. Other products for learning POS are on the other end of the spectrum, geared towards users that are more technically savvy. The output of such tools are difficult to consume, even for a native English speaker. In addition, tools that are made to help with speed reading such as Spreeder (7-Speed-ReadingTM) break sentences into individual words so ELLs continue to learn to read word-by-word, which does not help in learning to read faster or with improving reading comprehension. Thus, the ELLs are at a disadvantage when they required to read college level material because more time and effort must be spent on required course readings.

* 1. **Purpose**

CLASH is a web-based, Software as a Service (SaaS), Single Page Application (SPA) that is being developed to accomplish three main goals. The first two goals are to assist ELLs improve their reading speed and reading comprehension, and the final goal assist ELLs in identifying the correct part of speech for all words in a sentence. CLASH will accomplish these goals by providing ELLs with two different ways to consume a document that has been assigned by their ESL instructor. The first way that ELLs will be able to view a document is through the SLASH Handler. The SLASH Handler will increase students’ reading speed and reading comprehension by teaching them to read text in Lexical Bundles. The second way that ELLs will be able view a document is through the COLRS module. The COLRS module will assist students in identifying the parts of speech by colorizing words within the document. Each part of speech will have a designated color so that the part of speech of a word will be easily identified without the use of a separate lookup table. A simple example to highlight this point is that all nouns could be colored blue and therefore whenever the user sees a blue word, they will know it is a noun.

The first module, called SLASH Handler, is aimed to improve the reading speed and reading comprehension of students by displaying text on the screen as individual Lexical Bundles. A Lexical Bundle is a group of words that repeatedly occur together within the same register (Hanel, 2011). Some common Lexical Bundles include “I don’t want to”, “in the United States”, and “at the end of.” Teaching students to read in Lexical Bundles reduces the number of fixations and regressions by the reader, allowing them to understand sentences by stringing together thought groups rather than by stringing together each individual word of a sentence.

The SLASH Handler provides a playback environment that displays a selected document in Lexical Bundles one at a time. The display is similar to the scrolling text on a teleprompter used by television hosts. The user will have the ability to start, stop, fast-forward, and rewind playback. The user will also be able to speed up and slow down playback to allow the student to practice at a speed that they are comfortable with.

The second module, called COLRS, will colorize the eight essential POS within the document to assist the student in learning the uses of those words. These eight POS (nouns, verbs, adjectives, adverbs, pronouns, conjunctions, and prepositions) will be displayed with a separate color for each POS and the user will be able to select which POS will be highlighted at any given time. A primary goal of COLRS is to ensure that the user is able view the output text in the original sentence and paragraph structure, making it easy for the user to consume.

* 1. **Scope**

The CLASH prototype was designed to function as closely as possible to the real world product. Although the prototype will not have all the features of the real world product, the prototype will be a highly functional proof of concept that can have additional functionality added in subsequent CS411W sessions. Table 1 on the following page provides an overview of the primary features of the prototype and real world product.

The biggest difference between the real word product and the prototype is that the prototype will not have the Homework Mode. The Homework Mode would allow instructors to assign readings to students to test their ability to identify POS within the document. Once the student completes the assignment, the instructor would receive a status report indicating how the student performed on the assignment. Given the time and complexity of building the Homework Mode user interface and backend document management, the Homework Mode will not be included in the prototype.

Another difference between the prototype and real word product is that the prototype will only be able to process plain text documents. The real world product would have the functionality to process a wide selection of document types such as plain text, MS Word documents, and PDFs. However, MS Word and PDFs have their own proprietary encodings which make it difficult for the NLP library and SLASH Algorithm to process, therefore making the complexity and time to implement too much for a single semester.

The final major difference between the prototype and the real world product is that the prototype will have limited reporting capabilities. The real world product would have a robust reporting suite of pre-built reports that the Instructor could use to track progress of their ELLs. However, this functionality is not essential to show a proof of concept, and therefore only limited reporting will be included.

(This space intentionally left blank.)

|  |  |  |
| --- | --- | --- |
| **Features** | **Real World Product** | **Prototype** |
| Parsing Capabilities | Ability to Parse different kinds of documents | Ability to parse text copy and pasted in text block |
| Text Modification | Ability to modify and store previously parsed documents | Ability to modify and store previously parsed documents |
| Color Capabilities | Ability to color chosen parts of speech using a JSON format and JavaScript functions. | Ability to color chosen parts of speech using a JSON format and JavaScript functions. |
| Slashing Capabilities | Ability to identify Lexical Bundles through the inserting of slashes. | Ability to identify Lexical Bundles through the inserting of slashes. |
| Displaying Lexical Bundles in a single bundle form | Ability to speed up, slow down and pause Lexical Bundles being displayed. | Ability to speed up, slow down and pause Lexical Bundles being displayed. |
| Exception list | Lists of commonly used expressions that would otherwise be incorrectly handled by the SLASH Algorithm. | Lists of commonly used expressions that would otherwise be incorrectly handled by the SLASH Algorithm. |
| Login interface | User Authentication in a stand-alone environment | User Authentication in a stand-alone environment |
| Student Data Reporting | Tracks individual and collective student progress. To include words per minute, total time and total Lexical Bundles. Data to be stored in database. Displayed in graphs and statistics. | Limited basic student metrics will be available such as Lexical Bundles per Minute. |
| Homework Mode | Instructors have the ability to remove coloring of words and have students correctly identify the part of speech. | Not Included. |
| Administrative Privileges | Administrators are able to edit, add, or remove users and saved documents in the system. | Administrators are able to edit, add, or remove users and saved documents in the system. |
| SLASH Document Viewing Mode | Ability to view documents with slashes inserted and SLASH Handler. | Ability to view documents with slashes inserted and SLASH Handler. |

Table 1. Real World Product vs Prototype Continued

* 1. **Definitions, Acronyms and Abbreviations**

CLASH: Color Lexical Analysis algorithm and Slash Handler.

Client Side: The user-interface of CLASH.

COLRS: Colored Organized Lexical Recognition Software.

Document Processor: A Server Side component responsible for processing the text entered by an Instructor user type.

ELC: English Learning Center at Old Dominion University.

ESL: English as second language.

ELL: English Language Learner.

JSON: JavaScript Object Notation. A nested data structure commonly used to pass data between a server and a client.

Intensive English Program: A short and intensive English language training program offered by US colleges and universities to improve the English language skills of international students who did meet the minimum TOEFL scores for typical enrollment.

Lexical Bundle: A group of words that occur repeatedly together within the same register

MFCD: Major Functional Component Diagram.

NLTK: A suite of libraries and programs for symbolic and statistical natural language

processing (NLP).

Node.js: Open source, cross-platform run-time environment for server-side and networking

applications.

POS: Part-of-Speech such as noun, adjective, verb, etc….

Server Side: The back-end of the CLASH system responsible text processing, the database, user-authentication, and web-hosting.

SPA: Single page application. A highly responsive web application that fits on a single page

and does not reload as the web page changes states.

TOEFL: English language proficiency test required by universities for enrollment for internationally based students.

Software as a Service (SaaS): Software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet.

Token: Text that has been processed into individual words by the Document Processor

VM: Virtual Machine.

* 1. **References**

7-Speed-ReadingTM. (n.d.). *Spreeder*. Retrieved March 3, 2015, from Spreeder: www.spreeder.com

Hanel, S. D. (2011, March 5). *Lexical Bundles*. Retrieved February 7, 2015, from Communicating in English - An Internet-based, English Language Resource : http://sdhanel.com/corpuslinguitics/lexicalbundles.html

Haynie, D. (2014, Novemeber 14). *Number of International College Students Continues to Climb*. Retrieved February 6, 2015, from US News: http://www.usnews.com/education/best-colleges/articles/2014/11/17/number-of-international-college-students-continues-to-climb

Institute of International Education. (2014). *Open Doors Fact Sheet: Virginia.* IIE with support from the U.S. Department of State's Bureau of Educational and Cultural Affairs.

Open Doors. (2014). *Open Doors 2014: A 15-Year Snapshot.* Washington DC: Institute of International Education.

Powell, M. &. (2014). *Single Page Web Applications.* Shelter Island: Manning Publications Co.

* 1. **Overview**

The following section provides a detailed overview of the technical aspects of the CLASH prototype. This will included additional information on the major functional components, the software interfaces and architecture, and the user interfaces. Since the CLASH prototype is a web application it will not have any hardware interfaces or communication protocol interfaces.

1. **General Description**

The primary features of the CLASH prototype will be the SLASH Handler, the COLRS module, and the ability to allow an instructor to edit the text that was returned by the Document Processor. In order to successfully demonstrate these features, the CLASH prototype will also include a User module that will have three different user types Administrator, Student, and Instructor. Since the CLASH prototype will not integrate with any of the ODU systems, student user accounts will be manually entered by an Administrator user type or will be simulated in the database. These four features are the core functionality of the CLASH real world product and by successfully implementing these features it will demonstrate that CLASH is a viable product for the ESL department at ODU.

* 1. **Major Components (Hardware/Software)**

Figure 1 is the Major Functional Component Diagram (MFCD) of CLASH, which highlights the major software and hardware components. CLASH is structured in two major areas, the Client Side and the Server Side. The client side is the user interface which is written completely in JavaScript and the server side which is composed of the document processor, the web and application server and the database server.

(This space intentionally left blank.)

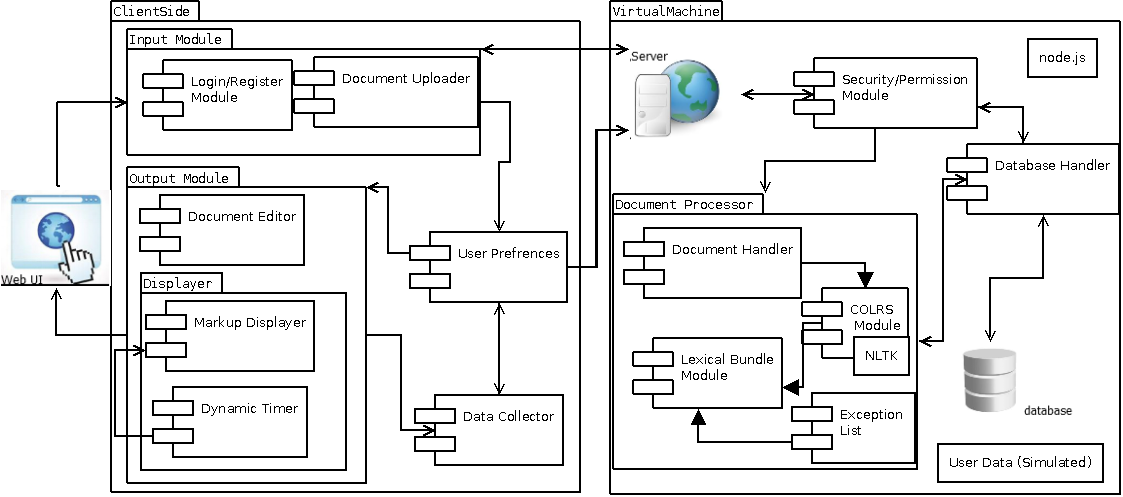
****

Figure 1. Prototype Major Functional Component Diagram

* 1. **Prototype Functional Description**

The main goal of the Client Side is to provide the user with a seamless experience to SLASH Handler and COLRS by providing the interface used to interact with the software. It will be written completely in JavaScript, while using some HTML and CSS as needed for additional text formatting. The Server Side is composed of three major elements. The first component is Node.js, which is the web and application server. Node.js is a prime choice for the application because it is also written completely in JavaScript and allows for synchronous communication with the Client Side. Node.js provides the functionality to allow the Client Side to change states without refreshing the webpage.

The next component on the Server Side is the database server. The database server that CLASH will utilize is MySQL. MySQL is an open source, production ready database server that will allow CLASH to scale both horizontally and vertically as more users and universities adopt the system. The database’s main function is to store user accounts, previously processed text documents, and user SLASH Handler performance metrics that can be reported to ESL instructors.

The final component of the Server Side is the Document Processor (DP). The DP is responsible for processing the text that will be viewed in the COLRS module and SLASH module. The DP takes the text entered by the instructor in the Client Side and parses the text into individual words, called tokens. Each token is then tagged with a POS using the open source Natural Language Processing (NLP) library, Natural Language Tool Kit (NLTK). Once all the words in the text are tagged with a POS, the text is sent through the SLASH module to build Lexical Bundles. Once the DP is complete, the document is saved to the database server and the final result is sent back to the client to be rendered in the SLASH Handler or the COLRS module. Figure 2 shows the process flow and lifecycle of how a document is entered, processed, and returned back to the user.

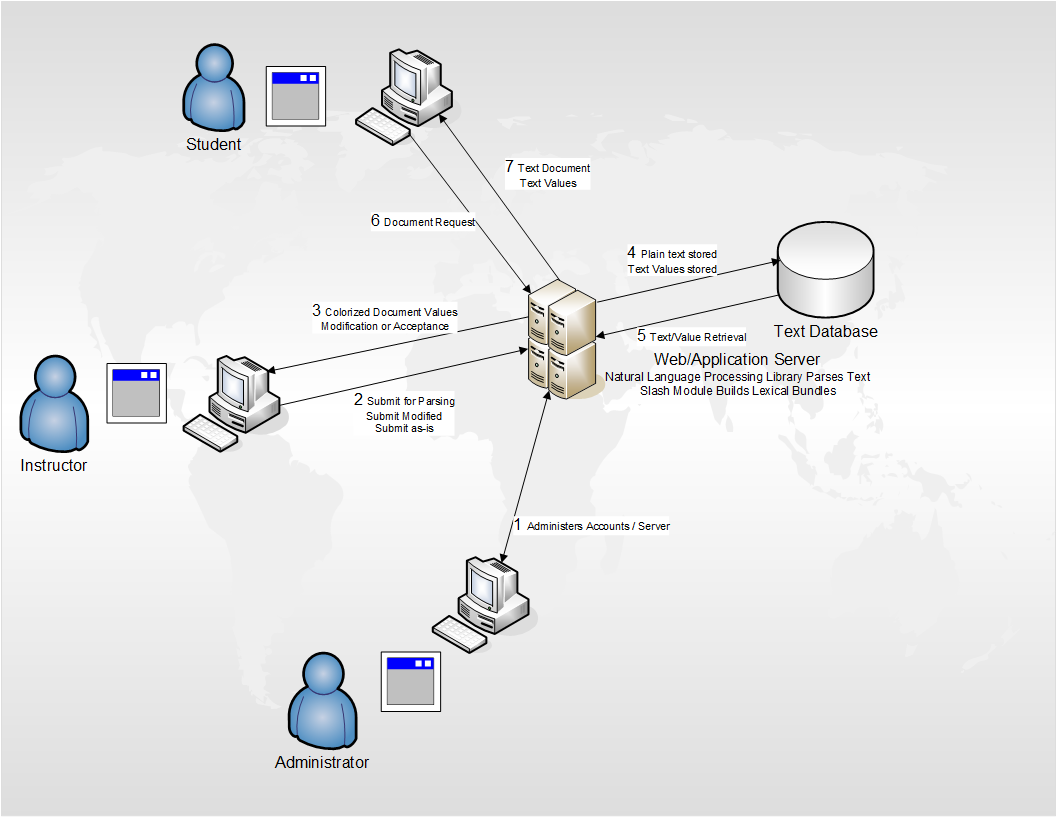


Figure 2. CLASH Process Flow

* 1. **External Interfaces**
     1. **Hardware Interfaces**

Not Applicable

* + 1. **Software Interfaces**

The CLASH prototype will have multiple software interfaces. In the front-end of the application, CLASH will integrate with Squirt.io. Squirt.io is an open-source text reader that will be the user interface of the Slash Handler. In the backend of the application CLASH will have multiple custom built entities including a MySQL database server, Node.js as the application server and webserver, and NLTK as the NLP engine.

* + 1. **User Interfaces**

Table 2 provides an overview of the available views in the user interface and the user type that has access to each view.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Views | Brief Description of View | Administrator | Instructor | Student |
| Logon Screen | Landing Page and initial logon screen. | X | X | X |
| COLRS Module | Colorizes the eight essential POS within the document and displays the output text in the original sentence and paragraph structure. | X | X | X |
| SLASH Handler | A playback environment that displays a selected document in Lexical Bundles one at a time. | X | X | X |
| Document Management | Add, Delete, Edit documents stored in the database | X | X |  |
| Document Editor | Set the POS or boundaries of the lexical bundles within a document. | X | X |  |
| User Management | Add, Delete, Edit user accounts stored in the database | X | X |  |

Table 2. User Interface Views

* + 1. **Communications Protocols and Interfaces**

Not Applicable