Lab – II Prototype Product Specification for CESR

Blue/Purple Teams

Cory Morewitz

CS 411W

Janet Brunelle, Hill Price

Old Dominion University

April 13, 2015

Version 2

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

Based on state-reported data, in 2004, 4,999,481 English as a second language (ESL) students were enrolled in public schools. Despite those numbers, 15% of these ESL students had no special resources or programs to help them learn the English language. In 2001, in the states that tested ESL students in reading comprehension, only 18.7% of ESL students were assessed as being at or above the norm. That same year, 10% of ESL students in grades 7-12 were retained. In February 2001, it was reported that ESL students had dropout rates up to four times that of their native English-speaking peers. (McKeon) Here at ODU, there is an entire department focused on addressing this problem, but while they work diligently to help ESL students, the processes of teaching reading and grammar are outdated. (McKeon) ESL students go through a year and a half long Intensive English Program and must pass the TOEFL (an English language proficiency test) at the end of that time. (Raver Lampman) If better tools that increased comprehension and retention could be brought to market, those numbers will shrink.

CESR, or Color Enhanced Slash Reader is a web application that includes three modules, COLRS Slash and Slash Playback. The COLRS module colorizes each part of speech (POS) in a text document with a chosen color to help increase comprehension of sentence structure and grammar. The Slash module parses text into lexical bundles (a group of words that occur repeatedly together within the same paragraph, or present one single thought), or thought groups to help increase reading speed and comprehension. The Slash Playback module will take the parsed text and play it back in a speed reader fashion. Though the program could be useful in a number of settings, CESR is primarily aimed at with ESL students.

Currently, the process is simple, for grammar, the professor writes a sentence on the board, then circles, or marks in some way, each part of speech. This is time consuming, and limits the amount, and size of the examples that the professor can explain. Psychologically, it has been proven that color impacts learning, by relieving eye fatigue, increasing information retention, increasing productivity and accuracy, and supporting developmental processes. (Engelbrecht) Extrapolating from that information, the COLRS module should help students identify POS and increase their potential for learning grammar. For increasing reading speed and comprehension, reading assignments are given, or students are directed to sites like Spreeder. These sites only increase the reader's ability to read faster. Do to this focus on speed; they also teach students to read word for word. ESL students are hampered by reading word for word. Not only is it tedious, but even if they learn to read quickly, their comprehension doesn't necessarily improve. Native born English speakers read in lexical bundles, this allows us to read more quickly, and with less re-reading. So, native born English speakers take advantage of this and read faster, as well as comprehend more completely and easily.

1.1 Purpose

CESR is a web application that will help professors teach ESL students POS and parse text into lexical bundles. The graphical user interface will be simple, easy to use, and contain three modules. The website will give the user the ability to highlight different POS to help delineate syntax as well as break text into lexical bundles. Additionally, the site will provide a reader application (Slash Playback) that will act as a speed reader using the lexical bundles provided as opposed to simply reading word for word.

1.2 Scope

The CESR prototype will be built as a single page application (SPA). This form of web application first on a single page and does not reload due to state changes. It will rely heavily on JavaScript as the primary programming language, as will most of the modules interacting with it. It will be using Node.js, .json file formats, and a JavaScript interface. There will also be a python shell that interacts with the natural language processor and the JavaScript. Where CESR will differ from traditional SPAs is it uses a relational database, as opposed to a NoSQL database.

There are many benefits to using a SPA. It is easy for the end user to access and use, no matter how limited their technical background may be. It is also beneficial because there is no software for the user to install. As there is no software to install, it also means the user can access CESR anywhere he has an Internet connection. Finally, being a single page means that the user will not get 'lost' in a path of web pages. All functionality is accessible through the menus on the main page. There will however be a steep learning curve to implement SPA functionality, which should be assessed in the risk matrix.

1.3 Definitions, Acronyms, and Abbreviations

CESR: Color Enhanced Slash Reader

Client Side: The user-interface of CESR

COLRS: Colored Organized Lexical Recognition Software. Module of CESR that colorizes part of speech based on what NLTK tokenizes and returns.

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.

ELL: English Language Learner.

ESL: English as second language.

GUI: Graphic User Interface

HTML: Hyper Text Markup Language

IBT: International Benchmark Test

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.

JS: JavaScript

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

Lexical Bundle: A group of words that occur repeatedly together, or represent a single thought group.

MFCD: Major Functional Component Diagram.

NLP: Natural Language Processing

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.

NoSQL: (often interpreted as Not only SQL) database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.

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

Python: a widely used general-purpose, high-level programming language.

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

Slash: Module of CESR that Slashes text into lexical bundles and displays them

Slash Playback: Module of CESR that displays a text stream showing one lexical bundle, of three to five words, at a time with the feature of speed control for display time. Speed reader that plays pre-slashed lexical bundles in the fashion of Spreeder.

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.

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.

Spreeder: Speed reading tool www.spreeder.com

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

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

Ubuntu: a Debian-based Linux operating system

VM: Virtual Machine.

1.4 References

Anderson, N. (1999, April 1). Improving Reading Speed - Activities for the Classroom. Retrieved February 1, 2015, from http://dosfan.lib.uic.edu/usia/E-USIA/forum/vols/vol37/no2/p2.htm

Engelbrecht, K. (2003, June 18). The Impact of Color on Learning. Retrieved February 25, 2015, from http://sdpl.coe.uga.edu/HTML/W305.pdf

English Proficiency. (2015, February 2). Retrieved February 6, 2015, from https://www.odu.edu/content/odu/admission/proficiency.html

Hoffman, D. (n.d.). Academictips.org - Reading and Highlighting Tips. Retrieved February 25, 2015, from http://www.academictips.org/acad/literature/readingandhighlighting.html

Improve Reading Speed and Comprehension. (2006, January 1). Retrieved February 25, 2015, from http://spreeder.com/

Lab 1 CS 411 Morewitz, Cory

McKeon, D. (n.d.). Research Talking Points on English Language Learners. Retrieved December 11, 2014.

Mikowski, M., & Powell, J. Single Page Applications. Manning Publications 2014.

Monarch Diversity. (n.d.). Retrieved February 6, 2015, from https://www.odu.edu/admission/international/global

Nishida, H. (2013). The Influence of Chunking on Reading Comprehension: Investigating the Acquisition of Chunking Skill. *THE JOURNAL OF ASIA TEFL*, *Vol.10*(No. 4), Pp. 163-183.

Raver-Lampman, Greg. (2014. August). Personal Interview.

The Condition of Education 2014. (2014, January 1). Retrieved February 6, 2015, from http://nces.ed.gov/fastfacts/display.asp?id=96

Tremblay, A., Derwing, B., Libben, G., & Westbury, C. (2011, January 15). Processing Advantages of Lexical Bundles: Evidence From Self-Paced Reading and Sentence Recall Tasks. Retrieved December 10, 2014.

1.5 Overview

This product specification provides the hardware and software configuration, capabilities and features of the CESR prototype. The information provided in the remaining sections of this document includes a detailed description of the hardware, and software architecture of the CESR Prototype; the key features of the prototype; and the parameters that will be used to control, manage or establish that feature.

2 GENERAL DESCRIPTION

The prototype is merely a proof of concept. As such, some functionality may be simulated, faked, or non-existent. For example, the COLRS module will not have homework modes for the student. Also the POS tagging will rely on a pre-built library as opposed to learning from the instructor editing the mistaken tagging. The prototype will also only be built for Old Dominion University. It will not provide functionality to expand the user base beyond campus. Table 1 on the next page briefly illustrates some of the prototype functionality.

(This space intentionally left blank.)

Features	Prototype
Parsing Capabilities	Ability to parse text copy and pasted in text block
Text Modification	Ability to modify and store previously parsed documents
Color Capabilities	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.
Displaying Lexical Bundles in a single bundle form	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.
Login interface	User Authentication in a stand-alone environment
Student Data Reporting	Limited basic student metrics will be available such as Lexical Bundles per Minute.
Administrative Privileges	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 Playback.
Text input	Will accept copy and pasted text and .docx

Table 1. Prototype Features

2.1 Prototype Architecture Description

From a hardware perspective, as illustrated in Figure 1, there will only be a machine which will run a virtual machine to hold the server which will contain the database, and website files. The software used will be a collection of open source, free to use, and programs the CESR team has written. The virtual machine will run Ubuntu 14.04 LTS, and the web and application servers will be Node.js. The Node.js server will interact with the MySQL database and the Natural Language Toolkit.

(This space intentionally left blank.)

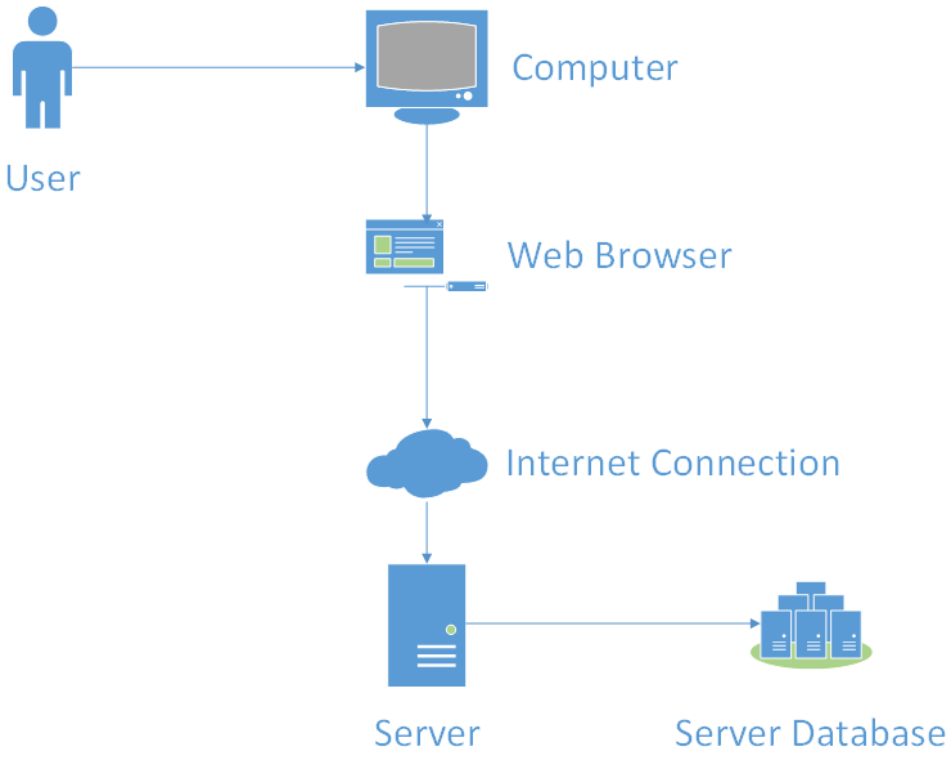

Figure 1. Prototype Hardware/Software Architecture

CESR will be a web-based application that will provide an easy to use interface. The CESR site will be accessible through any standard browser with access to the Internet. Using the intuitive user interface, users will be able to display parsed text that is colorized, Slashed, or both. The Slash module will feed into a Slash Playback module that will display the lexical bundles. The COLR module will allow users to single out particular part(s) of speech to display so they can easily study one, or multiple concepts. For example, if they would like to see only the nouns, all other colors and POS will be hidden.

CESR will have three different user roles, delineated by login: administrator, instructor, and student. The student will be able to view parsed documents, and interact with the Slash Playback module. The student will not be able to enter text, because CESR is not infallible. There will be times when the part of speech is marked incorrectly in a particular scenario. So as

not to lead to confusion, the students will only be able to view documents that have been reviewed, edited and approved by instructors. The instructor will be able to do everything the student can as well as: input text, edit the parsed output in case of error, and save and delete documents from the server. The administrator will have all the previously mentioned functionality plus the ability to add or remove instructor permissions to particular users, and add or remove student access.

(This space intentionally left blank.)

2.2 Prototype Functional Description

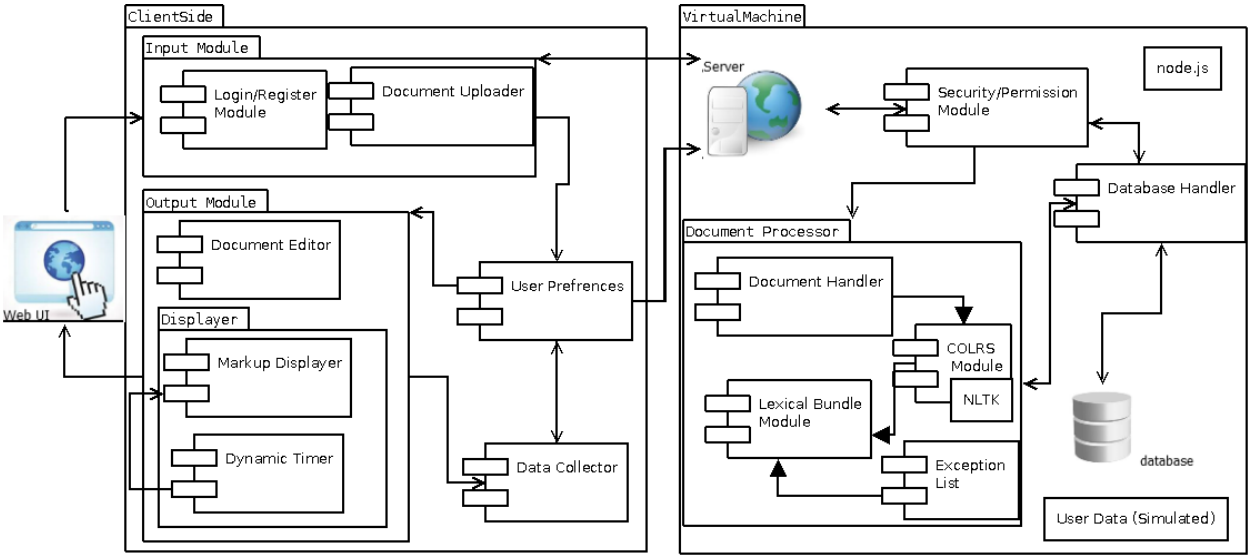

Figure 2. Major Functional Component Diagram Proof of Concept

Figure 2 illustrates the major functional components of the CESR proof of concept, which is a web application accessed through a browser. There are no special hardware requirements, as CESR is mostly a software product. The only hardware necessary is the server hosting the CESR site and backend.

The COLRS module will make use of a natural language processor (NLP) via python to parse text. This text will then be inserted into a .json file. The .json file will then be parsed by attribute, one of which is the part of speech. A JavaScript will run against the .json file and color each word with the same part of speech attribute tag and color it a matching color.

The SLASH module will take the parsed COLR .json file, and using the same attributes, will parse the text into lexical bundles. It will do this by following a few basic rules. First, the module will compare the document looking for "exceptions" set by the professor. This will ensure those lexical bundles on the "exception" list will not be split up. Then, it starts at the

beginning of the document and adds a slash after each period, coma, semicolon, colon, or question mark. Next, the module will add a slash before each proposition and each conjunction.

On the client side, there will be a website which receives text from the user, and then parses it according to what the user requests. In reality, the entire thing is parsed completely, but what is shown to the user varies depending on what the user selects. The user will select from the options: COLR, SLASH, and SLASH PLAYBACK. This will show them colored text delineating parts of speech, the text parsed into lexical bundles, or a playback module. The playback module will be similar to a speed reader.

2.3 External Interfaces

2.3.1 Hardware Interfaces

The only external hardware will be the client's machine which will connect to the CESR web service via the internet. While in the future, depending on capacity, a standalone server may be necessary, for now the virtual machine on the host machine is more than capable.

2.3.2 Software Interfaces

CESR will access a few software interfaces to facilitate certain functionality. MySQL will be used to facilitate storage of data and access control. The NLTK will be accessed using Python to facilitate the tokenizing of words into the proper POS. Squirt.io will be used as a wrapper for the Slash Handler to facilitate the playback functionality, and the Slash algorithm itself will be accessed to facilitate both the slashing of text for viewing and playing on the Slash Handler.

2.3.3 User Interfaces

N/A