Requirements Document – Team 02

1 User Requirements

1.1 Software Interfaces

List all the external systems with which the software product interacts. These are external systems/libraries that you have to interact with.

- Java Runtime Environment 1.6
- Local host file system

1.2 User Interfaces

Specify the logical characteristics of each interface between the software product and its users. This is a description of how the system will interact with its users.

- Main Software Interface
    - The product will be executed from the command line.
    - It is assumed the user knows how to get to a command line prompt.
    - The software will accept, but not require, flags which specify user preferences for the program run
    - The software should require a single argument which specifies the input file
    - The software should accept the required parameter and optional flags in any order

- Help Interface
    - The help documentation will be displayed when the user specifies the -h flag
    - The help interface will explain the required argument(s)
    - The help interface will explain the optional arguments, and their default values

- Program Output
    - The software should display, to the command line, the average sentence length of the input file
    - The software will terminate after a single execution

1.3 User Characteristics

Describe those general characteristics of the intended users of the product, including educational level, experience, and technical expertise.

- Users are expected to be of high school or college age
- Approximately 270 users will be using the application throughout the semester
- Users are not expected to have any technical proficiency
- Users will be running the software from a range of operating systems

2 System Requirements

These subsections contain all the software requirements at a level of detail sufficient to enable designers/developers to design/develop a system that satisfies those requirements, and testers to test that the system satisfies those requirements. This part of the document should provide a numbered (possibly hierarchical) list of simple, complete, and consistent functional and non-functional requirements.

2.1 Functional Requirements

1. Input
    a. a single command line argument is required
    b. 2 optional flags are accepted but not required
        i. -l *number* This flag is used to specify the minimum length of a word to consider in the calculation. If any word length is less than this number the word is excluded from the calculation.
            1. exit with a message if *number* is not a number.
            2. default is 3 if not specified
        ii. -d *delimiters* This flag is used to specify the delimiters the program should use when determining the end of a sentence. The delimiters should not be space separated (ie, for example a list of delimiters would look like this .;?)
            1. default is .;?!
2. Output
    a. print the average number of words per sentence in the file
    b. display the argument(s) and flags the user input
3. Help
    a. display a help screen if the -h option is specified
    b. if the -h option is specified, ignore all other options
    c. exit after the help information is displayed
4. Run
    a. Open the file
    b. read next group of words up to the delimiter specified by the -d flag or if -d is not specified up to the defaulted delimiter. If there is no more delimiters found in the file, read to EOF
    c. parse line into tokens, eliminating tokens less than minimum word size as specified by the -l flag
    d. increment the sentence count
    e. increment the total number of tokens
    f. calculate the average

g. when file is processed, display the average sentence length rounded down to the nearest integer

5. Error or Exit Conditions
    a. Exit with message if user does not specify an input file
    b. Exit with message if file is not a raw text file
    c. Exit with message if the file specified is not found
    d. Exit with message if the file specified cannot be opened
    e. Exit with message if the user specifies any flag other than -l, -d, or -h
    f. Exit with message if the file is empty

2.2 Non-Functional Requirements

- The software should be available to anyone with access to the computer it resides on
- The software should be a single, standalone utility or exectuable that is easy to execute
- The software should be platform independent
- The software should execute in 3 seconds or less to calculate and display the results
- The software should perform exactly the same when run multiple times
- The software should be inexpensive to create and cost less than 80 man hours to complete