Design Document

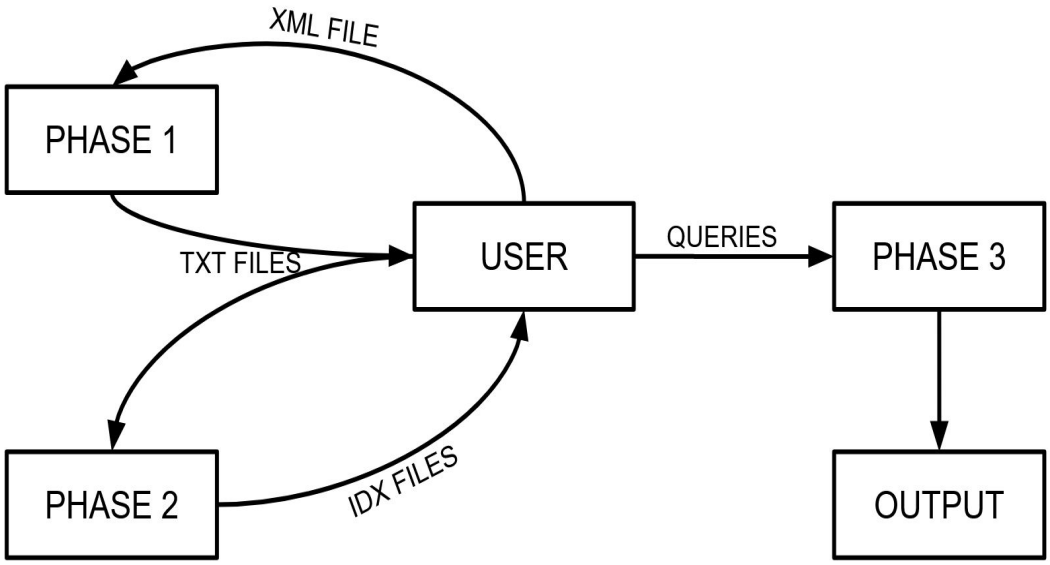
By

Jonathan Martins

Kevin Li

GENERAL OVERVIEW

Mini-project 2 is a 3 phase command line interface program that parses and then queries an xml file based on the user's wants. Phase 1 takes in an xml file as an argument and parses it into txt files. Phase 2 indexes the outputs of phase 1 into corresponding idx files. Phase 3 controls the user's queries and searches through the idx files to find matching outputs.



USER GUIDE

1. Run phase1.py with your xml file and an argument. ie. "python3 phase1.py x.xml"
2. Run the bash script phase2.sh to create idx files. Ie. "./phase2.sh"
3. Run phase3.py. ie. "python3 phase3.py"
    a. Program will prompt the user to enter a query. Here you have 3 options:
        i. Enter a query
        ii. Change output mode
            ➢ "output=full/brief"
        iii. Quit

b. If user input is a query, then the program will output all matches in the idx file. Output will either be just the row ID and the subject or the row ID and the full record depending on whether output mode is brief or full, respectively. Then go back to 3a and repeat.
c. If user input changes output mode, then nothing will be output, but the next query input will use the new output mode. Then go back to 3a and repeat.
d. If user input is quit, then the program terminates.

Algorithm

The Algorithm used for evaluating queries was very simple. First we decided on what format the Regex was going to be in, and depending on the situation (dates, multiple conditions, range searches). Then we grouped each section into its own category. For example, all queries involving dates are in one section, all queries involving multiple conditions are in one section. We started by coding the simple things, and the range searches and conditions were added on top of the simple sections, this is the method of decomposition, where we take a big problem and break it down into small parts and solve it one part at a time, and then we put it together at the end. Our program is very efficient, because in all the small parts we attempted to achieve the maximum efficiency, and as a result, if every small part is at maximum efficiency, when you combine them, it remains maximum efficiency. (Learnt in CMPUT204)

DETAILED DESIGN

Mini-project 2 consists of 3 phases; phase 1 and 3 are written in python, and phase 2 is a series of linux terminal commands written in a bash script. In phase 1, an xml file is passed as an argument when running the phase1.py. This python file parses the

xml file into 4 different txt files: terms.txt,emails.txt, dates.txt, and recs.txt.

These 4 files are then sorted and indexed in phase 2 with the use of piping in phase2.sh. Stdout of the sort command ran in the script is piped to break.pl as stdin and then the stdout of the perl program is piped to db_load. Db_load is responsible for indexing the file by hash or B+-tree and producing the corresponding idx files to be used in phase 3.

Phase3.py utilizes regular expressions to match and split the user's inputted queries in order to know how to most efficiently access the idx files to be used for searching.

TESTING STRATEGY

This time around we used a different testing strategy. One partner was responsible for Regex, and the other partner was responsible for queries. When we each completed our part, we passed our code to one another, and had each other try to crash the others program only with valid inputs. During the testing phase, we ran into a lot of bugs, such as putty and windows compatibility, python malfunctions, and there were a lot of scenarios and error checks are not considered in both the regex and query section. But the testing strategy worked wonderfully, and we were able to fix all the bugs.

Group Work Break-down

This time we split the work up.

Phase1	Kevin Li	(2-3 hours)
Phase2	Jonathan Martins	(1-2 hours)
Phase3(Regex)	Kevin Li	(19-22 hours)
Phase3(Queries)	Jonathan Martins	(21-23 hours)

After finishing all the above, we got together and finished the rest together.

Phase3(Output)	Together	(3-4 hours)