**TwitterSearch Version 1**

**SFWRENG 2XB3**

**Software Engineering Practice and Experience: Binding Theory to Practice**

**Department of Computing and Software**

**McMaster University**

**April 13, 2019**

Yumo Bai

Xiangxin Kong

Shuo Zhang

Dios Zhou

Chenwei Song

Table of Contents

[Report revision History 3](#_Toc6141800)

[Team member and roles 3](#_Toc6141801)

[Contributions 4](#_Toc6141802)

[Exclusive Summary 5](#_Toc6141803)

[Description of Modules 5](#_Toc6141804)

[The UML specification: 6](#_Toc6141805)

[Search Module 8](#_Toc6141806)

[Module 8](#_Toc6141807)

[Uses 8](#_Toc6141808)

[Syntax 8](#_Toc6141809)

[Exported Access Programs 8](#_Toc6141810)

[Semantics 8](#_Toc6141811)

[State invariant 8](#_Toc6141812)

[Access Routine Semantics 8](#_Toc6141813)

[Local Types 9](#_Toc6141814)

[Local functions 9](#_Toc6141815)

[Sort Module 10](#_Toc6141816)

[Module 10](#_Toc6141817)

[Syntax 10](#_Toc6141818)

[Exported Access Programs 10](#_Toc6141819)

[Semantics 10](#_Toc6141820)

[Access Routine Semantics 10](#_Toc6141821)

[Local functions 11](#_Toc6141822)

[Generic twitterADT Module 11](#_Toc6141823)

[Generic Template Module 11](#_Toc6141824)

[Module 11](#_Toc6141825)

[Uses 11](#_Toc6141826)

[Syntax 11](#_Toc6141827)

[Exported Types 11](#_Toc6141828)

[Exported Access Programs 11](#_Toc6141829)

[Semantics 12](#_Toc6141830)

[State invariant 12](#_Toc6141831)

[Access Routine Semantics 12](#_Toc6141832)

[Local Types 14](#_Toc6141833)

[Local functions 14](#_Toc6141834)

[ReadData Module 15](#_Toc6141835)

[Module 15](#_Toc6141836)

[Uses 15](#_Toc6141837)

[Syntax 15](#_Toc6141838)

[Exported Access Programs 15](#_Toc6141839)

[Semantics 15](#_Toc6141840)

[State invariant 15](#_Toc6141841)

[Access Routine Semantics 16](#_Toc6141842)

# Report revision History

Version 0: March, 2019

Version 1: April, 2019

# Team member and roles

| **Name** | **Student Number** | **Role in the Project** |
| --- | --- | --- |
| Yumo Bai | 400137535 | programmer |
| Xiangxin Kong | 400141397 | Leader&programmer |
| Chenwei Song | 400124879 | programmer |
| Shuo Zhang | 400065241 | programmer |
| Dios Zhou | 400082351 | progarmmer |

*By virtue of submitting this document we electronically sign and date that the work being submitted by all the individuals in the group is their exclusive work as a group and we consent to make available the application developed through SE-2XB3 project, the reports, presentations, and assignments (not including my name and student number) for future teaching purposes.*

# Contributions

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role(s) | Contributions | Comments |
| Yumo Bai | Programmer | Sort method  Search method  Meeting log  Presentation PowerPoint  Project presentation  Part of requirement specification  Part of design specification  Testing |  |
| Xiangxin Kong | Leader  Programmer | Sort method  Search method  Made the gui  ReadData  Presentation PowerPoint  Project presentation  Part of requirement specification  Part of design specification  Testing |  |
| Shuo Zhang | Programmer | Sort method  Search method  ReadData  Meeting log  Presentation PowerPoint  Project presentation  Part of requirement specification  Part of design specification  Testing |  |
| Dios Zhou | Programmer | Created the ADTs  Sort method  ReadData  Meeting log  Presentation PowerPoint  Project presentation  Part of requirement specification  Part of design specification  Testing |  |
| Chenwei Song | Programmer | Created the ADTs  Presentation PowerPoint  Project presentation  Part of requirement specification  Part of design specification  Testing |  |

# Exclusive Summary

As social medias has become a huge part of the human society, the value of information they carry is increasing day by day. Developing a method of efficiently finding information from a certain user group can be helpful in a lot of ways such as personal recommendations, society studies, etc.

In this project, we developed a dataset query that can show the most trending topic over a certain time period from a certain user group. The dataset we are using is from Twitter and the frequency of hashtag keywords being used are used as the metric.

# Description of Modules

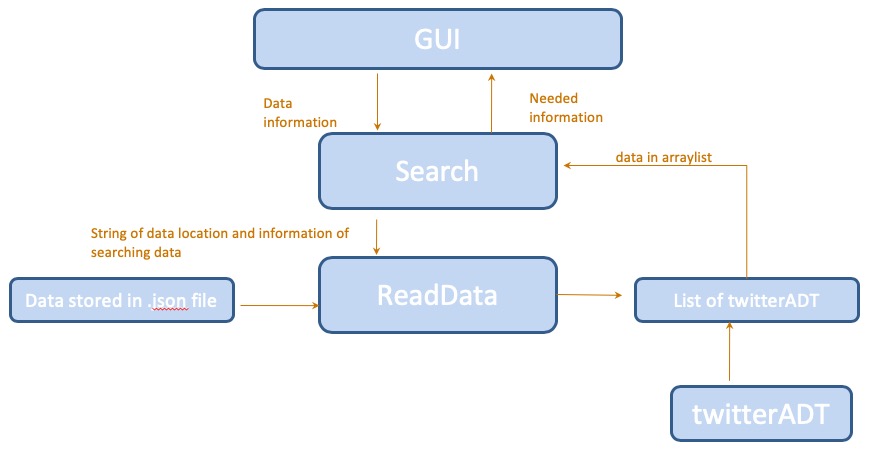
TwitterSearch is made up of three ADTs: ReadData, twitterADT and search. The program firstly read the information from the dataset, which is categorized as date, time, hashtag words, language, region, and like numbers. The user will be asked to choose the filtering conditions, and then the program will filter the stored data with those conditions, and count the numbers of how many times every hashtag word appears, then return the most frequently used one.

The ReadData class is used to read the dataset and load the information into a list a twitterADTs by using the function LoadUsers.

The twitterADT class defines the type twitterADT. Each twitterADT object collects the date, time language, region, hashtag and like number of each tweet.

The core function is defined in the class search. The function filter takes date, time language, region, hashtag and like number requirements and returns all the tweets that agree with requirements given. The search functions takes the list of twitterADTs returned and find the popular hashtags involved.

# The UML specification:



# Search Module

### Module

Search

### Uses

ReadData

twitterADT

### Syntax

### Exported Access Programs

|  |  |  |  |
| --- | --- | --- | --- |
| **Routine name** | **In** | **Out** | **Exceptions** |
| search | String, N, N, String, N, N | String | None |
| filter | String, N, N, String, N, N | Seq of String | FileNotFound |

### Semantics

### State invariant

FilePath = “src/javaapplication2/test.json”

### Access Routine Semantics

search(lang, begin\_date, end\_date, region, begin\_time, end\_time):

* Output: s such that (∀ t : String | t ∈ filter(lang, begin\_date, end\_date, reigion, begin\_time, end\_time) : count(t) ≤ count (s))
* Exception: None

filter(lang, begin\_date, end\_date, region, begin\_time, end\_time):

* Output: StringSeq seq such that (∀ s : twitterADT | s ∈ LoadUsers(FilePath) & getDate(s) ≥ begin\_date & getDate(s) ≤ end\_date & getTime(s) ≥ begin\_time & getTime(s) ≤ end\_time & getLanguage(s) = lang & getRegion(s) = region : (∃ t : String | t ∈ seq : t = getTag(s) ) )
* Exception: FileNotFound

### Local Types

StringSeq : seq of String

### Local functions

Count : String → N

Count(s) : (+s : String | (∃ str : String | str ∈ filter(lang, begin\_date, end\_date, reigion, begin\_time, end\_time) & s = str : ) : 1)

# Sort Module

### Module

Sort

### Syntax

### Exported Access Programs

|  |  |  |  |
| --- | --- | --- | --- |
| **Routine name** | **In** | **Out** | **Exception** |
| Sort | Seq of t, N |  | None |

### Semantics

### Access Routine Semantics

Sort(a, n):

* Transition: a := a such that (∀ num : N | num ≥ 0 & num < n - 1 : a[num] < a[num + 1] )
* Exception: None

**Local Types**

SeqT: seq of t

### Local functions

Merge: SeqT × SeqT × SeqT × N × N → SeqT

* Merge(a, l, r, lo, hi): a := SeqT such that (∀ t : Comparable | t ∈ l ∪ r : (∃ term : Comparable | term ∈ a : a = t)) & (∀ num : N | num ≥ 0 & num < n - 1 : a[num] < a[num + 1] )

# Generic twitterADT Module

## **Generic Template Module**

### Module

twitterADT

### Uses

N/A

### Syntax

## **Exported Types**

twitterADT = ?

### Exported Access Programs

|  |  |  |  |
| --- | --- | --- | --- |
| **Routine name** | **In** | **Out** | **Exceptions** |
| twitterADT | String, String, String, String, String, N | twitterADT | none |
| getDate |  | String |  |
| getTag |  | String |  |
| getLanguage |  | String |  |
| getTime |  | String |  |
| getRegion |  | String |  |
| getLikeNum |  | N |  |
| compareTo | twitterADT | N |  |

### Semantics

### State invariant

date: String

tag: String

language: String

time: String

region: String

likeNum: Z

### Access Routine Semantics

twitterADT(date, tag, language, time, region, likeNum):

* transition: date, tag, language, time, region, likeNum := date, tag, language, time, region, likeNum
* output: out := self
* Exception: None

getDate():

* Output: out := date
* Exception: None

getTag():

* Output: out := tag
* Exception: None

getRegion():

* Output: out := region
* Exception: None

getLikeNum():

* Output: out := likeNum
* Exception: None

getLanguage():

* Output: out := language
* Exception: None

getTime():

* Output: out := time
* Exception: None

compareTo(j):

* Output: out := (this.getLikeNum() < j.getLikeNum(): -1 | this.getLikeNum() > j.getLikeNum(): 1 | this.getTime() < j.getTime(): -1 | this.getLikeNum() > j.getLikeNum(): 1)
* Exception: None

### Local Types

N/A

### Local functions

N/A

# ReadData Module

### Module

ReadData

### Uses

twitterADT

### Syntax

### Exported Access Programs

|  |  |  |  |
| --- | --- | --- | --- |
| **Routine name** | **In** | **Out** | **Exceptions** |
| LoadUsers | String | Seq of twitterADT | FileNotFound |

### Semantics

### State invariant

date: Null

tag: Null

language: Null

time: Null

region: Null

likeNum: 0

### Access Routine Semantics

LoadUsers(path):

* Output: Read data from file associated with path and load a list of twitterADT with information including date, tag, language, time, region and likeNum from each tweet. Return the list of twitterADT.
* Exception: FileNotFound