Design Pattern Final Project – Social Network Search Engine

R00922009 楊詠翔, D98921028 王祐邦

June 25, 2012

1 簡介

在本次期末專題中，我們這組以 C++ 和 Python 實做了社群網站搜尋引擎，現階段支援 Plurk 以及 Twitter 的搜尋。主要的功能包括搜尋關鍵字和/或人名，瀏覽搜尋結果，還有推文以及回文的功能。

2 系統架構與模式使用

圖 2是本組程式的系統架構 (framework)。主要包括四個設計模式，將在接下來的段落說明。

2.1 User – Singleton

首先，在 program startup 時，會 instantiate 一個 User 的 instance，往後就只由這個 User 進行登入、登出、瀏覽等等。之所以在 program startup 就 instantiate，是為了避免 multi-thread 問題。

2.2 SocialNet – Singleton + Prototype Manager

我們想將 social network 也設計成 Singleton，讓每次只有一個 User 跟一個 SocialNet 互動。然而由於 SocialNet 本身是 Abstract Base Class，無法 instance，同時也為了在不同的社群網站之間切換，我們將 Prototype Manager 跟 Singleton 結合在一起。

首先，在 program startup 時，每個 SocialNet 的 derived class (Plurk, Twitter etc.) 各自 instantiate 各自的 Singleton (在圖 3的 private mambers "dummy")，並向 SocialNet 的 Prototype Manager 註冊。爾後當 client code (main.cpp) 想取得 SocialNet 的 instance 時，呼叫 SocialNet::Instance()，SocialNet 先將使用者要求的 singleton of derived class 複製 (Clone) 成自己的 singleton，再回傳給 User。如此一來也達到了避免 multi-thread problem 的功能，因為 SocialNet::Instance() 沒有 New，只有 Clone。

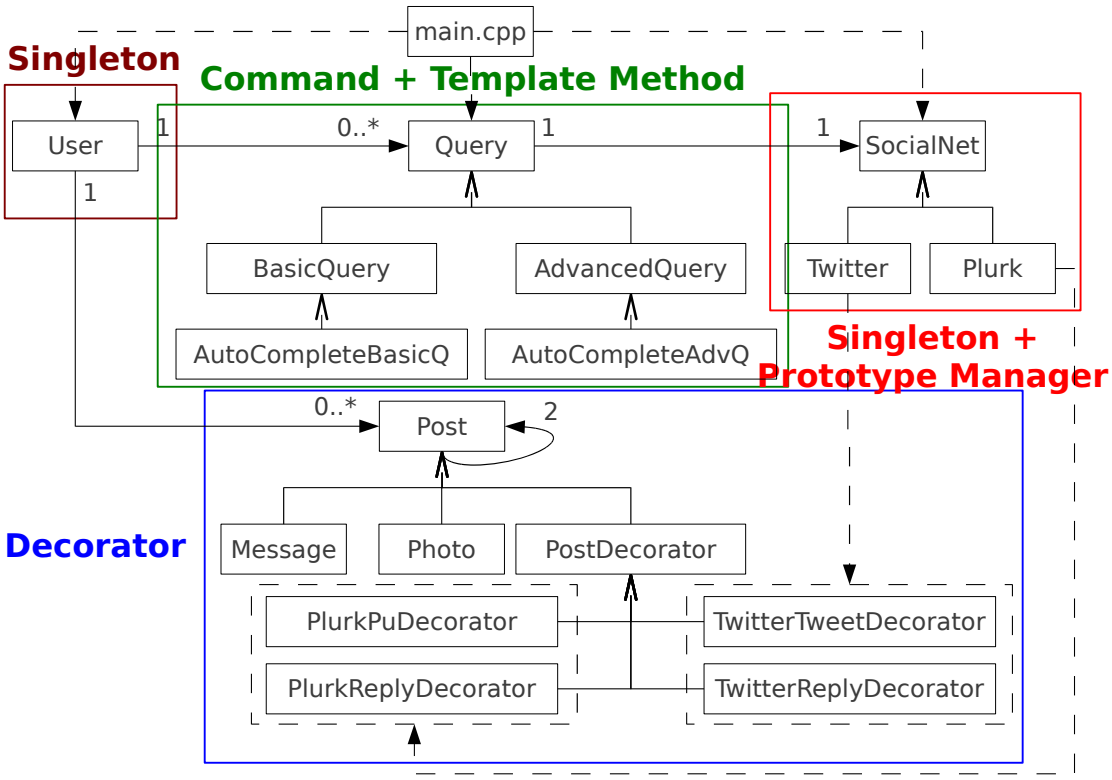

Figure 1: *The framework of our program*

2.3 Query – Command + Template Method

圖 4說明 Query 身為 Command pattern 與其他 class 的對應關係，同時也可用來說明我們系統的執行流程。首先由 User 透過 Query::InitQuery() 指定想要搜尋之關鍵字以及人名，之後 Query 再透過 SocialNet 的 BasicSearch() 或者 AdvancedSearch()，經由 Python 腳本呼叫社群網站的 API，進行相對應的搜索。最後再把搜尋結果 return 給 User。

我們認為將 Query 實做成 Command pattern 是相當直覺的。第一是因為 Query 是由 User 指定，但是是由 SocialNet 執行。好比點菜的要求由服務生記錄，但是是由廚師執行一樣。第二是為了可以將 Query 歷史儲存起來，以便之後進行的自動完成功能。

除了 Command pattern 以外，Query 同時含有 Template Method pattern。在圖 5中，class AutoCompleteBasicQ 跟 AutoCompleteAdvQ 分別是為了 override class BasicQuery 跟 AdvancedQuery 所增加的 derived classes。原本在 BasicQuery 跟 AdvancedQuery，系統直接用 cin 要求使用者輸入關鍵字以及人名；而在 AutoCompleteBasicQ 跟 AutoCompleteAdvQ，則是實現了自動完成的功能，稍候將在 4節說明。

2.4 Post – Decorator

在尚未加上 PostDecorator 之前，class Post 的主要 methods 包括印出自己的內容 (Post::PrintContent())，還有提供前往上一篇、下一篇、或者結束瀏覽的選項 (Post::PrintOption(),

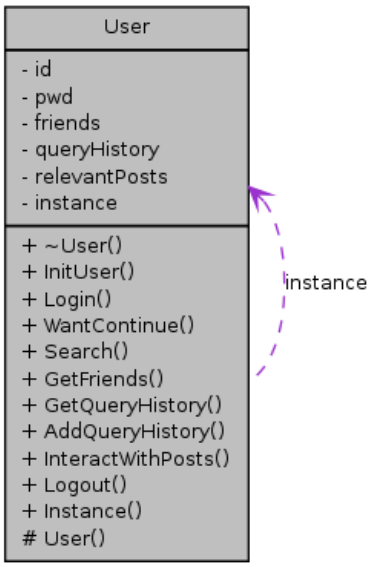

Figure 2: *The collaboration graph of class User*

Post::ExecuteOption())。注意我們目前實做的 concrete Post 只有 Message，而 Photo 當初雖然制定了介面，但沒有被使用過，只是為了示範其延伸性所以保留著。

而 PostDecorator 的目的，是為 Post 加上推文以及回文的功能。Concrete SocialNet 在完成搜尋之後，負責 create Post 物件，並加上相對應的 PostDecorator。每個 concrete SocialNet 對應到固定一組 PostDecorator。因此換個角度講，SocialNet 也可以視為 Abstract Factory pattern，負責創造 Post，並回傳給 Query。

3 社群網站 API

我們在實作社群網站搜尋系統的過程中使用了大量的 API。由於各大社群網站的資料的存取方法以及傳遞規範不盡相同，即使大同小異，在實作上仍然有許多繁瑣的規定以及要求。所幸每個社群網站都有提供他本身的 API 原始碼操作說明，此外還有許多第三方的開放原始碼供所有欲開發者使用。我們這次總共實作了 Plurk 以及 Twitter 這兩個社群網站的搜尋，使用的 API Library 為 plurk-oauth by clsung 以及 tweepy by joshthecoder : 這兩個都是 Python Library for API，我們利用 Library 提供的函式實作了各種我們自定的功能，並且利用系統主程式 (c++) 呼叫額外寫的這部份的程式 (python code)。

plurk api : http://www.plurk.com/API

twitter api : https://dev.twitter.com/docs/api

3.1 Login

由於目前大部分的 social network api 都是基於 OAuth2 的認證方式，所以為了讓不同使用者可以自由登入我們系統，我們找了 OAuth2 的 python library 來幫助我們完成登入的機制。我們讓初次使用的用戶在登入的時候指定到認證網頁輸入出正確的數字，就可以

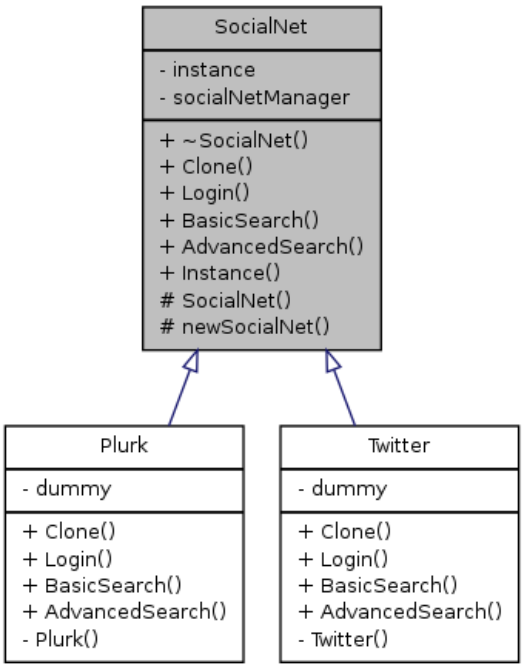

Figure 3: *The inheritance graph of class SocialNet*

讓程式獲得所需要的 Access Token，之後程式就可以利用該 Access Token 進行各種 API 的操作，而獲得的 Access Token 也會自動存下來，讓該使用者下一次登入的時候不需要再去索取 Access Token。

3.2 Reply and Like

這部份都是利用不同社群網站所提供的 API 來實作，我們只需要給定正確格式的參數，就可以實作搜尋，回覆，或是按讚等不同的功能。由於多數社群網站的回傳格式都是 JSON 格式，我們將其整理過後把我們程式需要的資料留下來存成檔案供我們的主程式去讀檔案。在系統中我們實作了兩種搜尋的方法，第一種是給定一個搜尋的關鍵字，對所有該社群網站的用戶所發的文章進行搜尋。這種方法是使用社群網站提供的 API 直接進行搜尋。缺點是他不能過濾發文作者是否為用戶認識的人。所以我們額外實作了另外一種搜尋方法，其一是指定發文者以及關鍵字，我們先去取得所有該發文者所發的文章，再一一比對是否其文章中有出現關鍵字。另外一種是搜尋使用者的時間軸（在 Plurk 以及 Twitter 之中皆有 Timeline 的觀念）所有的文章，並且一一比對是否有關鍵字的出現。利用這種方法可以讓使用者找到身邊的朋友或是指定使用者的文章。

4 其他特殊功能

除了搜尋、瀏覽、推文以及回文之外，我們還另外加了兩個功能。一個是 Password Masking，也就是在輸入密碼的時候輸出"*" 記號。另一個是自動完成。在搜尋時，輸入

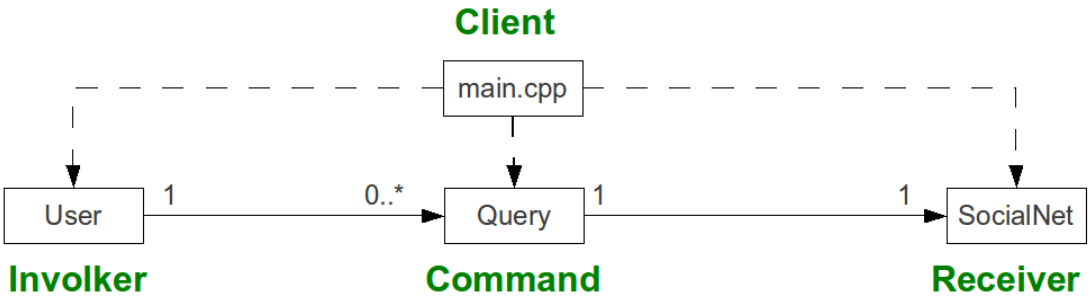

Figure 4: *The collaboration graph of class Query*

關鍵字以及人名的時候，按 Tab 即可自動完成。關鍵字是藉由搜尋使用者的過往的紀錄，系統會保留最近 100 次以內的關鍵字搜尋，在程式終止時寫成 log 檔，並在下次程式啓動時讀入。而人名則是搜尋好友名單。

5 系統延伸性與未來展望

本系統的設計足以提供包括新增社群網站 (class socialNet)、文章類別 (class Post)、與文章的互動方式 (class PostDecorator) 以及搜尋類別 (class Query)。要新增社群網站，除增加 socialNet 的 derived class，也要同時增加 PostDecorator，以支援其推文、回文功能等等。同樣地，要增加文章類別或者與文章的互動方式，除增加 Post、PostDecorator 的 derived class 之外，也要修改或者新增 socialNet。而如果要增加搜尋類別，除增加 Query 之外，也要修改或者新增 socialNet，支援與該 Query 相對應的搜尋演算法。

而如果將來想要重構 (refactor) 程式碼，一個可能的方向是將 Query::InitQuery() 以及 SocialNet::BasicSearch()、SocialNet::AdvancedSearch() 以 Bridge Pattern 抽離出來。因為這些 methods 關聯緊密且可能時常需要擴充。

6 編譯與執行

本程式在 Linux 跟 MacOS 上測試過，介面為 command line。直接在資料夾下執行 make 即可以 g++ 編譯，並產生一名為 bin 的執行檔。

7 分工

R00922009 楊詠翔：負責社群網站 API，並參與架構設計。
D98921028 王祐邦：負責系統架構設計、實做，以及增加特殊功能。

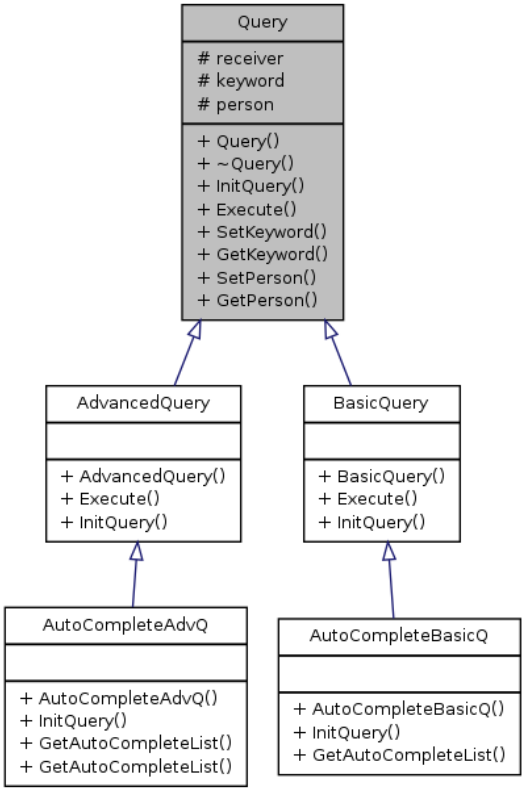

Figure 5: *The inheritance graph of class Query*

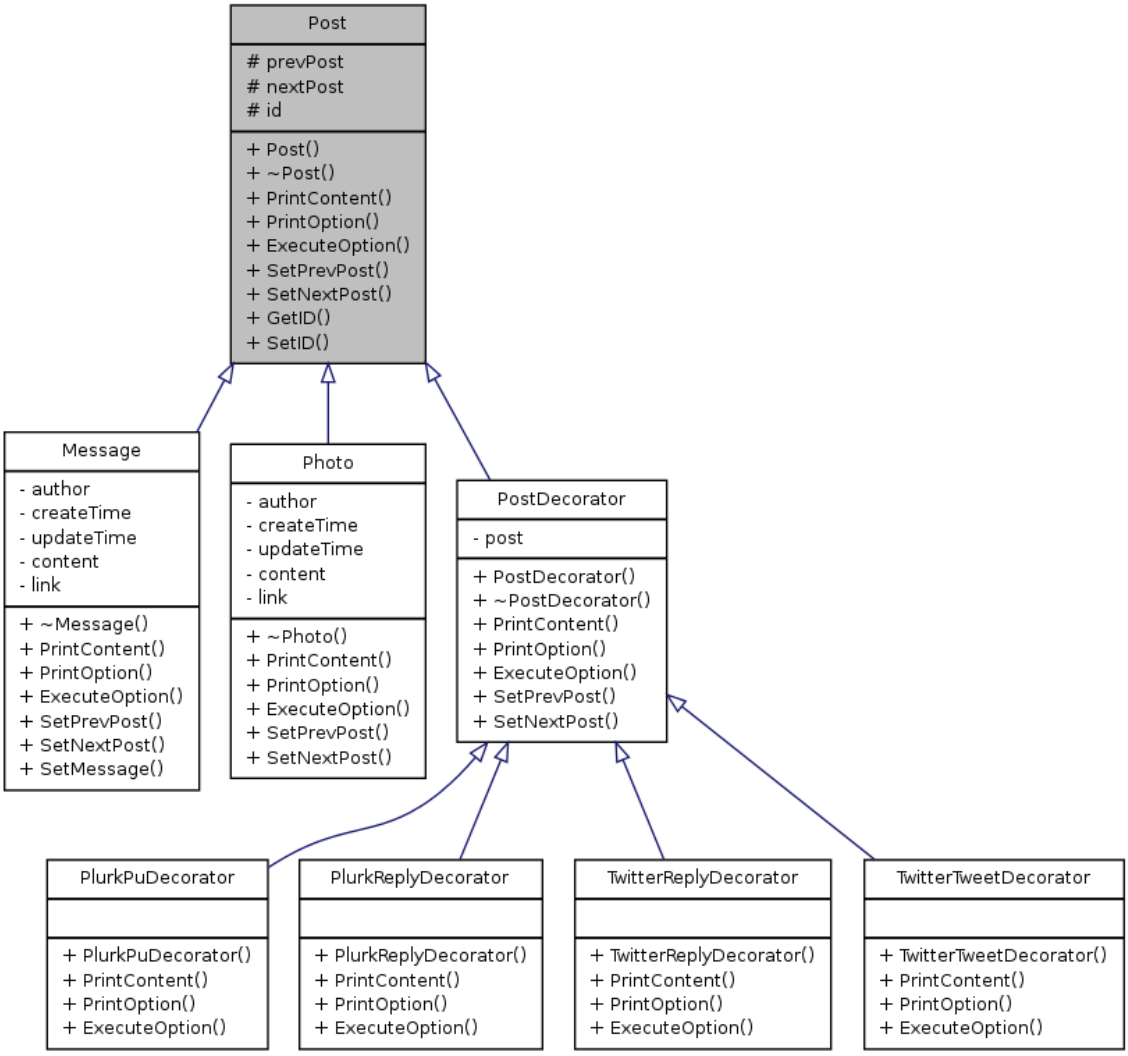

Figure 6: *The inheritance graph of class Post*

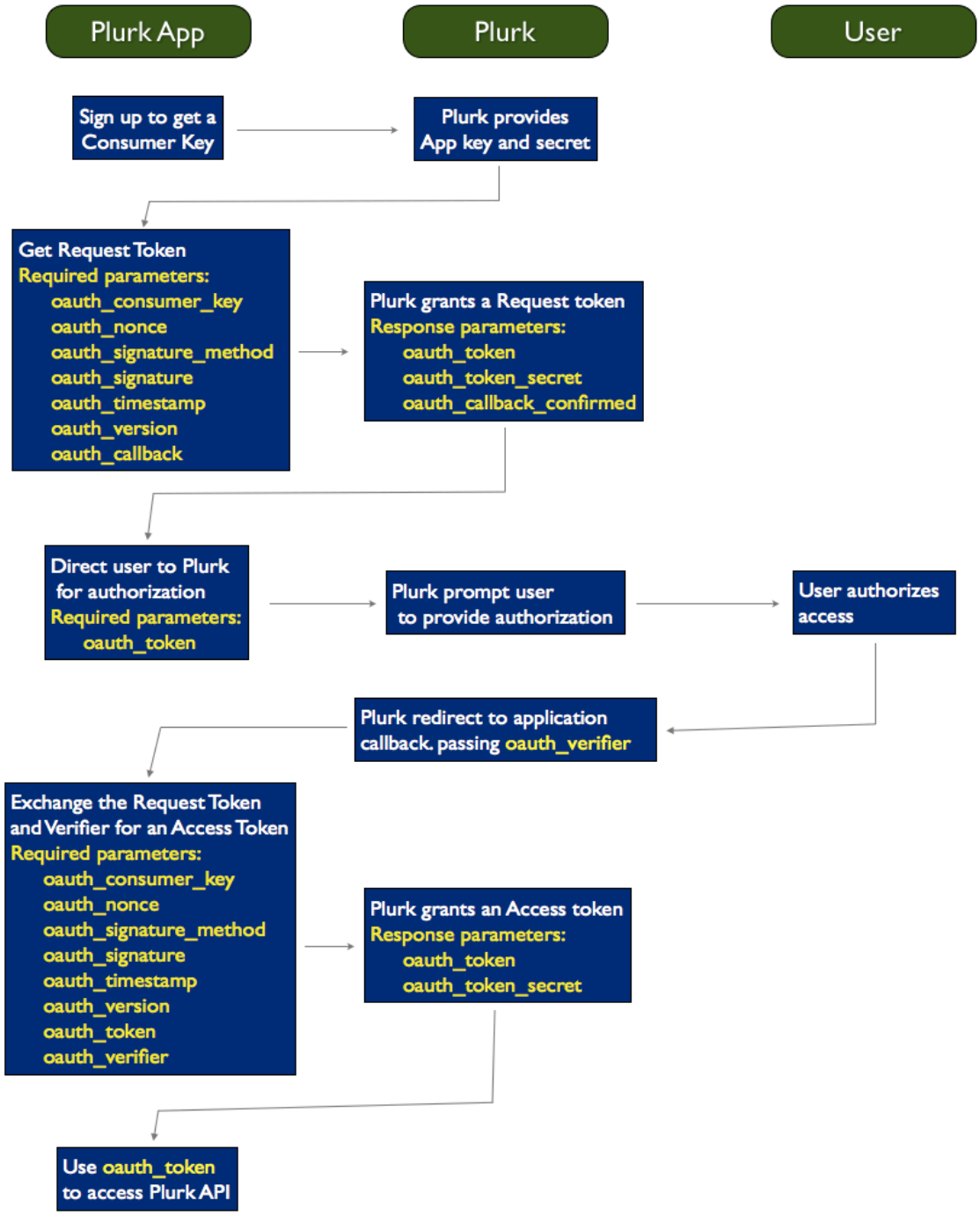

Figure 7: *Plurk* 利用 *OAuth* 認證的流程圖，其他各大社群網站的方法亦同。