Maintenance plan guidelines

目錄

1. 簡介

我們的目的(Purpose)在於跨平台與不同作業環境下，達到同步、版本相容以及更新。目標(Goal)是建立一個系統，可以更加方便管理學生成績，自動匯入大量數據資料，由系統幫忙作分析，以省去教師評分和管理的麻煩。所含括的範圍(Scope)包含國中小、高中職、大專院校、公職考試.....等機關都可以做更多元的應用。

2. 參考資料

- IEEE Standard for Software Maintenance.pdf
- Software Maintenance Support Plan-v1.0.pdf
- 經濟部工業局軟體維護作業規範.pdf

3. 軟體組織架構維護

組織架構如下：在有學生的成績資料庫的前提之下（含 Lab1、Lab2、Lab3、Mid-term、Final exam）設定權重，總平均成績將依照權重計算之。輸入學生的學號後，隨即顯示該生姓名與主選單，依照指令輸入做操作，要維護的功能包含顯示成績、排名、平均，以及計算加權平均……等。下方列出所有我們須維護的區塊。

- **class Main**
  - main ( )
- **class UI**

  處理主選單、處理使用者的成績（各項以及加權平均）、排名查詢以及配分更新要求
  - start() //印出輸入學號的初始畫面
  - checkIfExist(String ID) //檢查使用者輸入的學號
  - showList(int id) //印出主選單

- chooseList(int id, String choose) //處理主選單的使用者輸入
- showGrades(int id) //印出各項成績
- showAverage(int id)
- showRank(int id) //印出排名
- reCalculate(int id) //更新權重
- showNowWeights() //顯示目前權重
- inputNewWeights() //引導使用者輸入各項配分
- checkNewWeights() //確認使用者輸入的新配分
- Exit(int index)
- UI() 建構子

■ **class Grades**

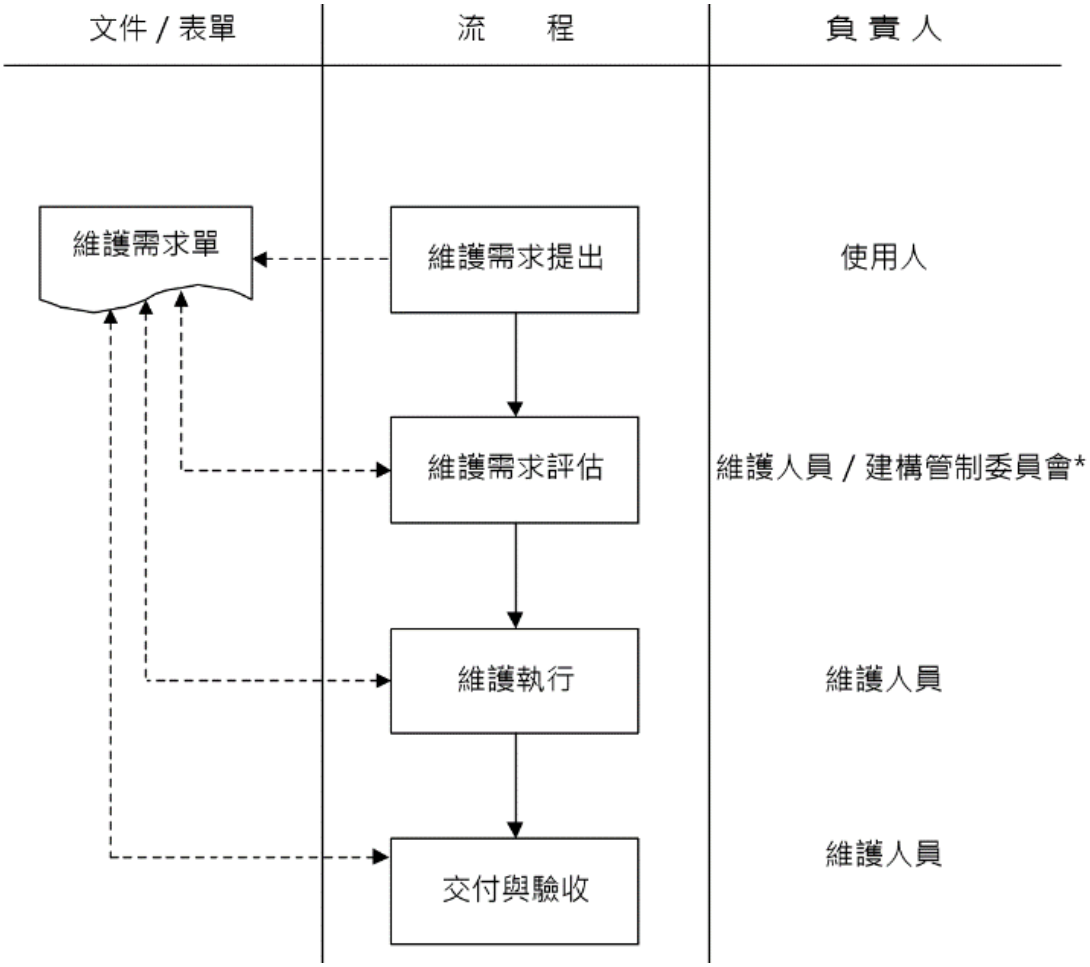
儲存 ID, name, lab1, lab2, lab3, midterm, final Exam 和加權平均後總成績

- Grades () //建構子
- loadGrades() //讀入成績資料檔案
- calculateGrades(double lab1, double lab2, double lab3, double mid, double fe) //計算總成績
- weightedAverage(double lab1, double lab2, double lab3, double mid, double fe) //計算加權平均成績
- getNewData() //取得所有人資料的所在 list
- getRank(int id) //計算此 id 學生之排名

有了組織架構以後，維護須排成優先順序，舉例來說，我們的優先順序為：程式無法執行>執行結果錯誤>新功能實作，依照此順序一一解決維護需求。所使用的開發工具為 eclipse，並安裝分析擴充工具。在技術方面則是用 Junit Test 的方式去做單元測試以及整合測試，找到問題以後回報維護需求。

4. 流程

維護需求提出→評估→確定→優先順序確定→執行→交付與驗收



照理說，需求者要填寫維護需求單，並提供錯誤發生處、輸入資料、硬體環境.....等輔助資料說明，有助於維護需求評估。但我們小組為了加速流程，省略了填寫維護需求單的步驟，僅提供錯誤發生處以及資料給負責寫程式的人修正。

再來評估需求是否為必須，若非必須，則可先暫緩，接著就可以確定評估後的維護項目。按照功能重要性以及錯誤嚴重性來排序維護的優先順序，緊急的錯誤(例如危及使用者個資)會優先做維護處理。

經過以上步驟後，交給程式開發人員執行，所有工作皆必須經過測試及審查，以確保工作之完整。在開發的工作中，我們不斷的測試以及更新版本，修正各種錯誤。

最後修改過後的程式碼與文件報告均直接交付給原維護需求者，也就是我們的組員，省略簽字驗收的步驟。

5. 異常解決與報告

- 輸入了學號，但資料庫無此學號
  →登入畫面
  →輸入不合法學號
  →螢幕顯示「ID 錯了」

→提示使用者重新輸入

- 主選單畫面，輸入不合法指令
  →登入畫面
  →輸入合法學號
  →進入主選單
  →使用者輸入不合法指令
  →螢幕顯示「指令錯了」
  →提示使用者重新輸入指令

6. 維護計畫

我們的小組有八人，成員的工作分別為開發程式（四名）、測試人員（兩名）、維護人員（兩名）。城市的基本架構由四名開發者所撰寫，交付測試人員測試，最後再給維護人員進行後續維護動作。一旦發先新錯誤，測試人員將立即通報維護人員，執行維護流程。

7. 維護紀錄與 log

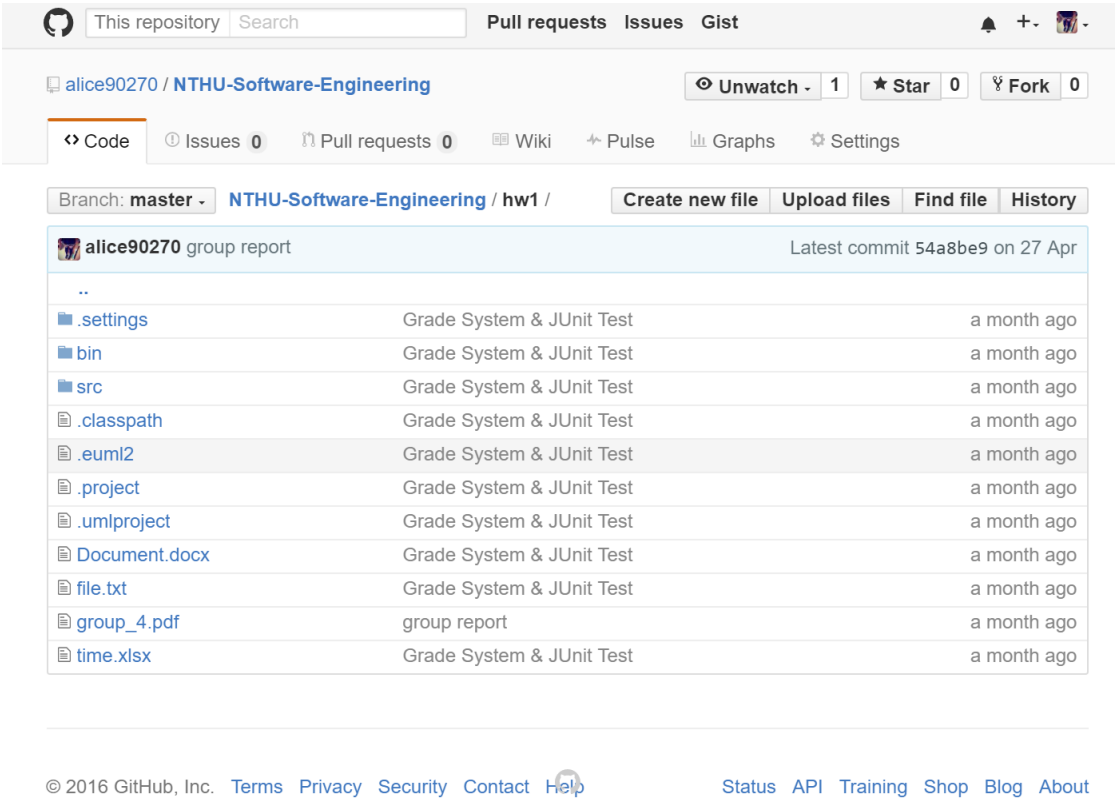
log	說明	日期
功能實作	匯入並讀取檔案	2016/4/3
修正 bug	輸入不合法學號的辨識，顯示「ID 錯了」提示使用者重新輸入	2016/4/3
功能實作	計算加權成績	2016/4/7
功能實作	計算排名	2016/4/7
功能實作	顯示成績、平均、排名	2016/4/7
功能實作	更新加權配分	2016/4/7
修正 bug	配分加總可以超過 100	2016/4/9
新增	不及格成績加上*一起顯示	2016/4/10

我們嘗試使用 github 來作維護，目前 repository 中的紀錄並不多也不夠完整，乃因我們是開發到一定程度之後，才開始決定使用 github 作版本控管。裡面的 log 仍可作為維護紀錄參考依據。
網址：https://github.com/alice90270/NTHU-Software-Engineering/tree/master/hw1

8. 維護工具

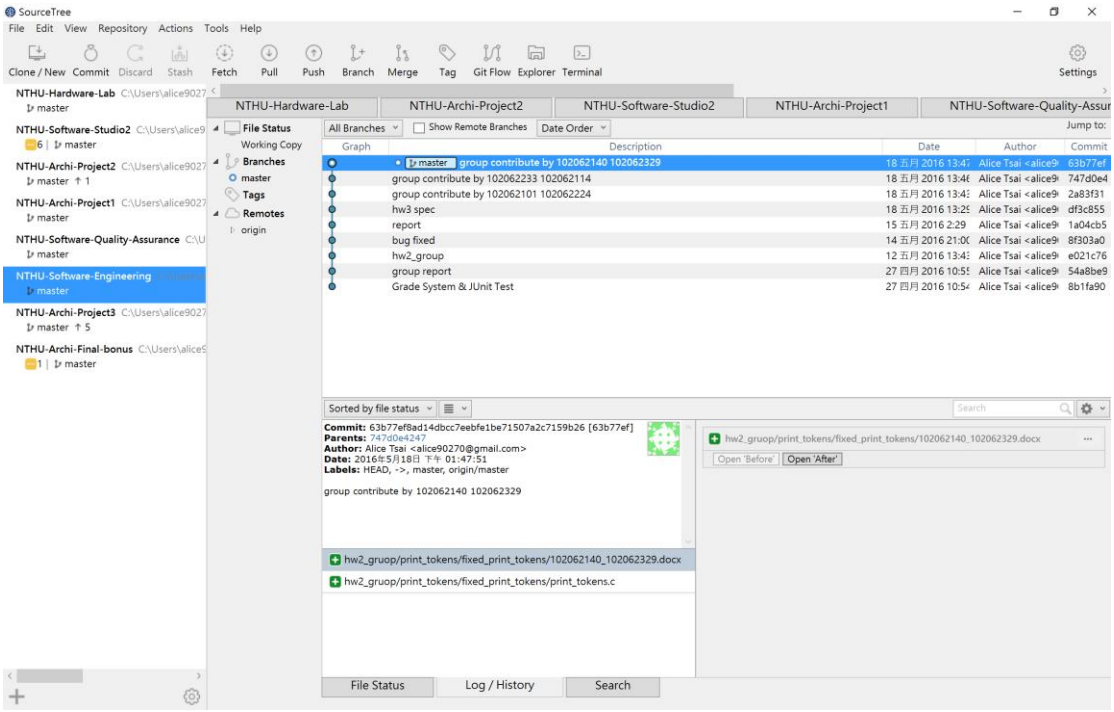
A. Github ：記錄版本與程式碼的分支

在伺服器端和本地端做同步，網站上主要是連線到伺服器端，只要有網路皆可以檢視目前的最新進度，並 PULL 至自己的本地端作修改、PUSH 自己修正過後的程式碼到主分支。



B. Source Tree：和 github 配合，在本地端做版本控制。

負責開發的組員們皆可以透過這個軟體來同步自己本地端的程式碼到我們的 github 上，方便檢視最新版本的進度以及修改的分支。主要是本地端的控管，只要不 PUSH 到 github 上，版本分支就不會被其他人所看見。



C. Slack：小組討論所用，除了 Facebook 以外的溝通工具

有鑑於談正事時使用社群軟體實在不洽當，我們統一使用 SLACK 來做溝通，便於記錄所有開會的項目、討論事項……等。他的搜尋功能使我們可以快速找到過去討論過的問題。