報告題名：物件導向設計學習心得與專案分析

作者：田杰
系級：資訊二丙
學號：D9223468

開課老師：薛念林老師

課程名稱：物件導向設計

開課系所：3C 科技學程資訊二

開課學年：93 學年度 第 2 學期

摘要

物件導向設計已成為軟體開發的趨勢，而這門課也順理成章的成為資訊系軟體工程學程的必修。本學期老師採用 Java 作為這門課的程式語言，並派了八個專案要我們演練，雖然專案不算大，但也著實讓我為這些專案廢寢忘食，日夜顛倒。除了透過專案瞭解物件設計與 Java 的精義外，老師也要我們確實的紀錄每個專案的程式大小與所花費的時間，藉以統計自己的程式產量。本報告的目的主要是對八個專案做一致性的討論，讓自己更清楚設計實做上的缺失，並透過資料的分析瞭解自己流程上、能力上的問題。

一、前言

程式語言與發展方法不斷的推陳出新，目前物件導向設計方法廣受歡迎，被認為是可以解決許多軟體工程問題的方法。物件導向具有不少潛在的優勢：可以提高程式碼的可重用性、可以增加資料的封裝能力，並且提供繼承的機制以減少程式的重複撰寫 -- 這些對於大型程式的發展都是很有幫助的。對於主修軟體工程學程的我而言，這門課是必然要修的。

老師一開始即將這門課的目標定義為兩方面：技術與專案管理。在技術方面，本門課授課目的主要是在教導學生物件導向的目的與精神，並學會使用 Java 實作、以物件導向的概念設計程式，並培養學生視窗程式設計、遊戲設計、網路程式設計、UML 輔助設計的能力。為了能讓我們深刻的體驗理論，本門課一共有八個專案，由淺至深。從基本的 Java 程式流程、物件的使用、API 資源的查詢使用，進而到到物件設計、視窗程式設計、UML 輔助設計、網路程式設計以及 MVC 架構，都可以有一個完整的了解。在技術方面老師還要我們練習一些工具，例如開發環境 Eclipse、產生 API 的 Javadoc、UML 設計的 Omendo。這八個專案之間是有銜接性的，我們最後的產出是一個專案預估管理系統。

在專案管理方面，老師採用輕量級 PSP（Personal Software Programming）[6] 方法論來要求我們收集專案的資訊。該方法論提供一個收集及分析專案資料的方法，例如收集每一個專案的開發時間、每一個專案的大小、及分析個人的程式產量等。透過此分析方法，我可以清楚的說明在專案上一共花了 56 小時又 8 分鐘、共寫了 2587 行的程式碼、專案實做的時間佔了所有時間的 56%等量化的資訊（詳見第三節與第四節）。一開始收集這些資料時真的很不舒服 – 寫程式就寫程式，為什麼還要收集這些資料？不過最後真的很高興堅持了下來，因為從這些資訊我可以更清楚的瞭解自己的物件設計能力。

這個報告共分為五節。第二節是我在修完這門課後對物件設計與 Java 的重點摘要與觀感；第三節是本門課八個專案的描述，以及我個人對這八個專案的相關資訊 – 包含我的物件設計圖與專案資訊；第四節是專案的綜合分析 – 包含產

量分析與階段比重分析。第五節是我對這門課的學習重點所做的自評與結論。

二、物件導向設計之我見

在我學習 Java 以前，C 是我唯一學過的程式語言。C 是一個功能導向的語言，也就是說程式的單位是以功能 (function) 為主。當你需要時，就自己設計或是呼叫已經寫好的功能，並且在這之間作參數傳遞。因此 C 程式是由一大堆功能組成的。物件導向是一種新的思維，將以物件為主的概念，帶入程式設計中。其特性是可以提高程式碼可重用性 (reusability)，以及讓日後程式的維護更加容易。我記得老師用一個象棋的例子說明功能性與物件導向的差異，可以充分的說明兩者的差異。在功能導向的實做中，一個棋子要吃另一個棋子會用以下的功能：*eat(chess1, chess2)*。其中 *eat()* 就是一個功能。但如果用物件導向來說，就會變成 *chess1.eat(chess2)* 可以看到主體已經不是功能，而是物件了。

物件設計與 UML 的相輔相成，亦有助於大型程式的發展。相較於傳統以功能導向的程式語言，物件導向的程式語言有幾個特性，這是我以往在學 C 語言時所完全沒有接觸過並且覺得很神奇的：封裝 (encapsulation)、繼承 (inheritance)、多型 (polymorphism)

- 封裝。以前我在寫 C 語言時，從老師那兒拿到題目後，就開始將問題切成一個一個功能，並且將這些工作分配給數個功能一起完成。而依照物件導向程式語言的規範，我們可以將一群完成特定功能的靜態成員與動態成員設計成一個物件，這便是封裝。將程式碼封裝成一個物件，目的是為了未來這個物件可以被重複使用。封裝也意味著資料會有更好的安全性，不會任意被外界存取。封裝還有另外一層重要的意義—對於日後使用該物件的程式設計師而言，只需知道如何使該物件便可，並不需要了解物件內部是如何運作的，如此一來有層次的分工，對於大型程式的發展與維護，也有許多幫助。這些特性在以往功能導向的 C 語言是不存在的。
- 繼承。以往寫 C 程式時，如果有要用到相同的功能時，可能都必須重新寫一遍，或是複製之前寫過的程式碼，如此一來不但不方便而且也會使程式的 size 變大，形成不必要的浪費。在物件導向的程式語言中，繼承可以解決這個問題。繼承可以保留原來程式的功能並擴充改寫之，原先的程式碼可以被重複使用，不須重新撰寫，如此一來可以省去大量的開發時間與空間。
- 多型。由父類別規範子類別共同的行為，再由父類別操作子類別，藉此可以達到簡化程式，使整體程式更為簡潔，勻稱。以上這些特性都是在 C 與言沒有辦法看到的。

三、專案敘述與討論

表一：專案表單使用說明

專案資訊	專案名稱、開發時間以及專案簡述。	
學習成果表	呈現的是本專案的學習目的以及成果評量。	
功能需求表	呈現專案要求要完成的功能以及完成度。	
時間紀錄 (Time Log)	Study	開發此專案時所閱讀的相關資訊。
	Plan	開發專案所規劃的方法。
	Coding	撰寫程式遇到的問題。
	Compile	語法錯誤的除錯。
	Test	語意錯誤 (邏輯錯誤) 的除錯。
	Refactoring	重新整理程式碼。
	Report	整理報告。
詳細資訊	呈現的是複雜度預估，專案開發的預估以及真實資訊，產量等。	
程式架構及執行結果	以圖片呈現程式的架構圖(class diagram)以及執行結果的重要截圖。	

專案作業是這門課比重最重的部分，因為程式設計首重親自練習實作理論，印象才會深刻。除了技術的演練外，老師還希望我們做事情必須有計畫，有步驟，而不是亂無章法、四處碰壁。老師將專案的步驟分為七個步驟（如表一），並要求我們記錄每個階段所花費的時間。將時間確實記錄下來的好處可以統計每個專案所花費的時間，進而瞭解自己的產能。

為了方便每個學生繳交作業，老師建立了一個 Java 品質保證系統（Java Quality Assurance System; JQAS）。該系統主要的功能是公布每個專案的需求，提供系統及報告上傳的介面，提供專案資料（每個時段花的時間）記錄的功能，最大的好處是當程式與報告上傳後，JQAS 會自動的計算程式的大小，並將 jar 的檔案解壓縮成 Java 檔案，再呈現上 web 上，所以我們可以上網看但自己所有的專案資訊。

本節將會對八個作業（1A-8A）做簡略的介紹，主要強調在專案資訊、專案的設計及實做後的心得。由於這門課是大二的物件導向設計，重點並不在 UML 的介紹，所以我們所用到的 UML 是相當簡單的。表一是我在此報告中所採用的專案表單，用以說明每一個專案。

其中 compile 階段是指程式寫完後第一次 compile 到第一次通過 compile 通過所花費的時間，其所花費的時間主要在解決語法上的錯誤。Testing 階段主要確定所產生出的結果是我們所期待的，其所花費的時間主要在解決邏輯上的錯誤。Refactoring 是指將程式重整，使得程式的可讀性更高。產量 = (程式碼行數 X 複雜度) / 時間。程式行數與複雜度相乘又稱為功能點[1]（function point; FP）[1]，

[1] 單以行數來推估所必須花費的時間是不夠精準的，因為不同的技術複雜度可能行數相同，但所花費的時間卻相差很大。本門課所採用的功能點計算是簡化過的計算方式，詳細的功能點計算到大三的軟體工程才會學到。

所以產量的單位為 FP/m（每分鐘幾個功能點）。而誤差 = (真實值 - 預估值) / 預估值 X 100%。

1A：“Hello World”

專案資訊	
專案名稱	Hello World
建立日期	2005/3/7
專案描述	第一個 Java 程式

學習成果表 (成效：1-5)	
成效	學習目的
5	● 練習 JDK。
5	● 練習簡易的 Java 指令。

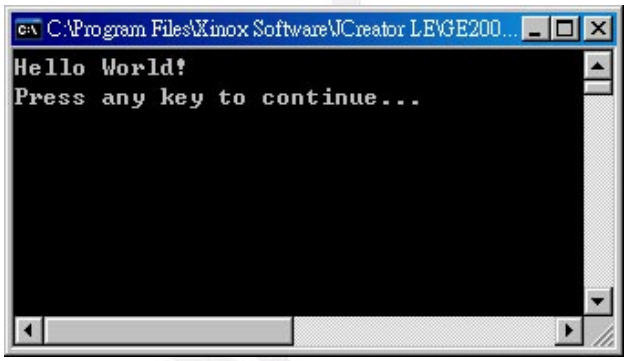
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 執行後秀出 Hello World。
5	● 若加入學生姓名，則出現 Hello World，學生姓名。

Time Log (單位：分鐘)		
時間	階段	詳細說明
45	Study	● 參考洪維恩博士所著，Java2 教學手冊[3]。 ● 閱讀 Java 的歷史背景以及由來。 ● 研讀第一個程式：Hello World。 ● 在命令提示字元下練習 Java。 ● 安裝第一個 IDE 工具：JCreator，並練習操作。
15	Plan	● 熟悉 JCreator 環境，並撰寫第一個程式。
7	Coding	● 將書上的程式碼抄下來。 ● 遇到保留字：class、static、public，感到很疑惑。 ● 對於格式化輸出，以前 C 語言用 printf()；Java 改用 System.out.println()，覺得不了解。
0	Compile	● 第一個程式很短，過程中沒有什麼錯誤。
0	Test	● 第一個程式很短，過程中沒有什麼錯誤。
0	Refactoring	● 第一個程式很短，沒有整理程式碼。
8	Report	● 對於不懂的保留字加以思考。

詳細資訊					
類別數目	1				
邏輯複雜度	1		預估	實際	誤差(%)
需求複雜度	1	行數 (行)	10	7	-30%
技術複雜度	1	時間 (分)	120	75	-37.5%
總複雜度	3	產量 (FP/m)	0.25	0.28	12%

程式架構 (class diagram)

執行結果



心得：J2SDK(Java2 Software development kit)是昇陽(SUN)所發展的一套 Java 程式開發軟體，本專案由安裝 J2SDK 開始，熟悉 Java 的基本命令、開發環境、以及一些基本的程式流程。之後我安裝了第一個 IDE 工具 JCreator，並且熟悉 JCreator 的開發環境。不過因為第一次接觸 Java，雖然寫出第一個程式，卻知其然不知其所以然。比如說 C 語言慣用的 function：printf()，現在則改用 System.out.println()、保留字 class、public、static、參數(String[] arg)等等都讓我覺得很疑惑，幸好在後來課程都瞭解到了這些關鍵字的含意。

2A：“Counting Line of Code”

專案資訊	
專案名稱	Counting Line of Code
建立日期	2005/3/10
專案描述	計算程式碼以及註解的行數

學習成果表 (成效：1-5)	
成效	學習目的
4	● 練習讀取檔案。
5	● 練習數字的運算。

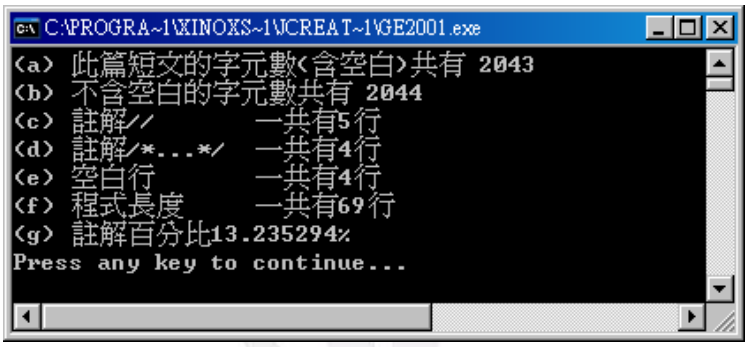
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 可以輸入一個 java 檔案。
5	● 算出程式總行數。
5	● 算出程式碼(code)的行數。
4	● 算出註解(comment)的行數。
5	● 算出註解率(註解/程式總行數)。
2	● 加入其他的功能。

Time Log (單位：分鐘)		
時間	階段	詳細說明
35	Study	● 閱讀 Java 的基礎：變數宣告、輸入輸出、多重分支、程式流程。
		● 比較與 C 語言基本程式流程的不同。
		● 閱讀 Java 的讀檔物件。
		● 了解 Java 註解的撰寫方式，如此才能計算註解行數。
22	Plan	● 構想要用哪些變數紀錄行數，以及計算行數的方式。
		● 為註解下明確的定義。
112	Coding	● 用自己的想法分別計算程式行數。
		● 本來想用堆疊的方式解決雙引號字串及星號註解問題，因剛接觸 Java 不熟而改用其他方式。
		● 克服邏輯上的問題花了較久時間，因為必須解決引號內的雙斜線以及跨行註解的問題。
38	Compile	● 依照 JCreator 提示，解決編譯錯誤，因為剛接觸 Java，所以忘記把表示式(expression)寫在 method 裡，算是比較誇張的錯誤。
28	Test	● 自己先數出正確程式行數，再比對輸出結果，解決邏輯錯誤。
5	Refactoring	● 整理程式碼的排版，將變數重新命名以利辨認。
24	Report	● 整理出幾個.java 檔的執行結果。

詳細資訊					
類別數目	1				
邏輯複雜度	2		預估	實際	誤差(%)

需求複雜度	2	行數 (行)	150	187	-24.6%
技術複雜度	2	時間 (分)	240	248	-3.3%
總複雜度	6	產量 (FP/m)	3.75	4.5	20%

程式架構

執行結果


心得：本專案透過存取檔案，了解 Java 的 I/O 類別庫並且學習使用 I/O 類別。學會使用 FileInputStream、InputStreamReader、BufferedReader、FileOutputStream OutputStreamWriter、BufferedWriter 等等物件。Java 讀取文字檔必需建立 InputStreamReader 物件並在建構元指定讀取的檔案，再由 BufferedReadedr(將字元轉成字串的物件)搬進記憶體, 再由字串處理的物件 String 加以處理. 讀檔中間可以學到 Composition(包含)或者是 Cascading(聯結)的程式撰寫技巧，使得程式更精簡易讀。計算程式碼行數的學習目的是為了幫助我們了解 Java 的程式流程，相當類似 C 語言。迴圈、多重分支結構、陣列等等程式語言基本結構，都與 C 一樣。由於這個專案必須計算註解行，因此必須先了解 Java 的注解的撰寫格式，以免作虛功。

3A：“File Process”

專案資訊	
專案名稱	File Process
建立日期	2005/3/12
專案描述	處理網頁原始碼字串

學習成果表 (成效：1-5)

成效	學習目的
5	● 練習檔案處理，字串解析。
3	● 練習網路資料的讀取。

功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 由 System.in 讀進一個 URL。
5	● 可以支援"忘了打"http"。
5	● 該網址的前五行印出，並輸出到一個指定的檔案。
5	● 若輸出檔案已存在，必須提供 rename 的功能。
5	● 從該檔案讀入資料，將每個字依照排序印出，並計算其出現次數。
5	● 用 2A 的程式計算此次專案的行數，將行數寫在 3A 的報告中。
5	● 字的解析可用空白鍵。
5	● 字的解析可用除了英文以外的字。

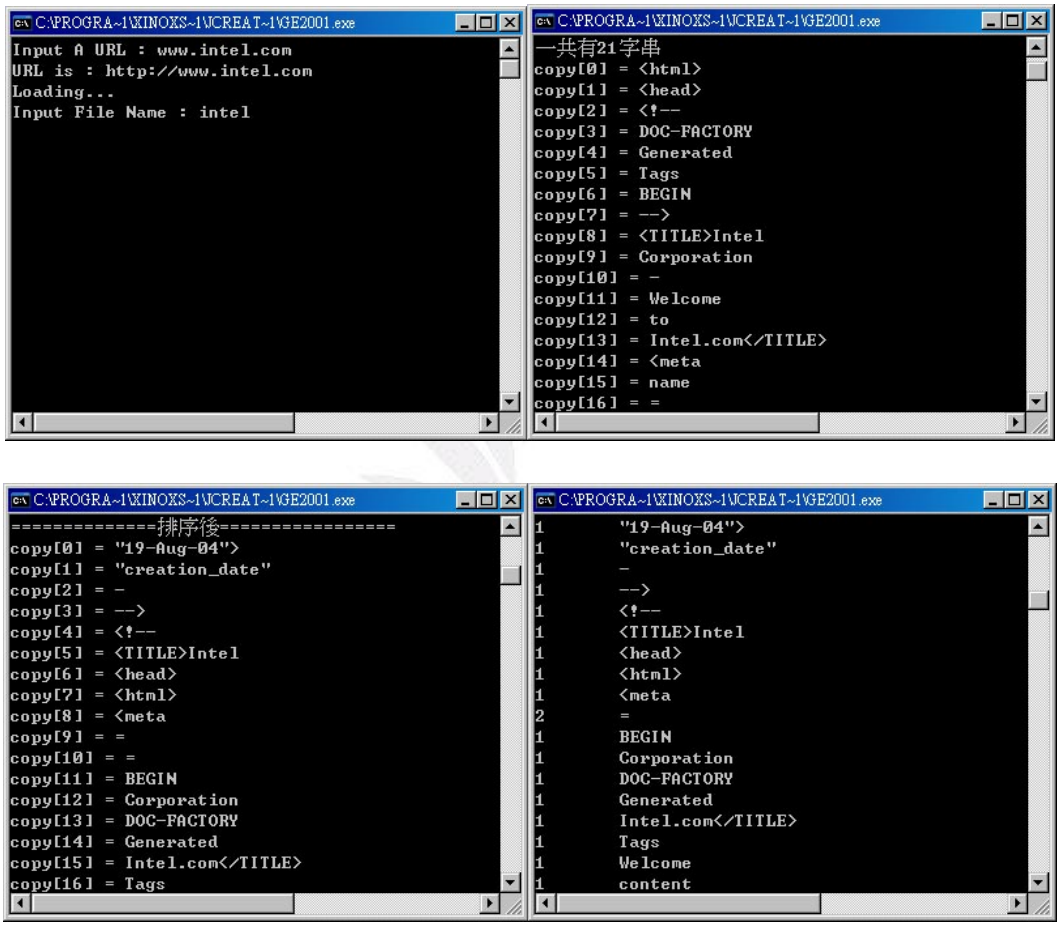
Time Log (單位：分鐘)		
時間	階段	詳細說明
52	Study	● 閱讀 Java：URL 這個物件，以讀取網路資料。
		● 閱讀 String 裡的 split method，以切割字串。
		● 閱讀陣列的使用。
		● 閱讀 Java 的排序 Arrays.sort()、以將每個字依照排序印出。
		● 練習查詢 API，以使用背後支援 Java 的豐富資源。
28	Plan	● 因為需求功能多，所以須先規劃程式流程，以免混亂。
		● 排序以及計算字串有許多方式，因此事先規劃好方法。
122	Coding	● 依照計畫的流程實作程式。
		● 使用 URL 讀取網路資料，以 File 的 exists()判斷檔案是否存在再寫入讀出，之後使用 split("\\s")依照空白切割字串，接下來加以排序計算。
28	Compile	● 變數常常忘了初始化就使用，導致編譯錯誤，Java 編譯器會先要求變數初始化，避免無意義的變數導致程式錯誤。
26	Test	● 自己先檢視網頁原始碼，再比對輸出結果，解決邏輯錯誤。
		● 另外要檢查使否有做檔案存在的 check 以及 rename 功能。
12	Refactoring	● 整理程式碼的排版，將變數重新命名以利辨認。
25	Report	● 整理出執行結果。

詳細資訊	
類別數目	1

邏輯複雜度	3		預估	實際	誤差(%)
需求複雜度	1	行數 (行)	200	145	-27.5%
技術複雜度	1	時間 (分)	240	293	22%
總複雜度	5	產量 (FP/m)	4.1	2.4	40.6%

程式架構

執行結果



心得：練習使用 URL 物件存取 Internet 上的檔案，學習 System.in、System.out 格式化的輸入及輸出，以及熟悉多重分支結構在 Java 上的實作，以處理執行時期(run time)可能發生的各種狀況，並且學會排序、計數。練習利用 String 物件的 split method 切割字串。這個程式中我體會到了 API 強大的地方，可以用 String 類別裡的 compareTo(String anotherString)跟其他字串作比較以判斷先後順序，或者是用 Arrays.sort()直接進行字串的排序。字串處理是程式語言非常重要的一門

學問，Java 本身只接受字串的輸入，可以避免 C 語言接受數字型態的輸入發生無法辨認它是以 ASCII 碼儲存的字元或者是數字。Java 本身強大的字串處理能力, 以及發展完熟的字串處理物件, 都為程式設計師在開發程式時省下不少力氣。

4A：“Define Class”

專案資訊	
專案名稱	Define class
建立日期	2005/3/17
專案描述	時間紀錄

學習成果表 (成效：1-5)	
成效	學習目的
4	● 學習如何定義與使用類別。
5	● 學習 Date 類別的使用。

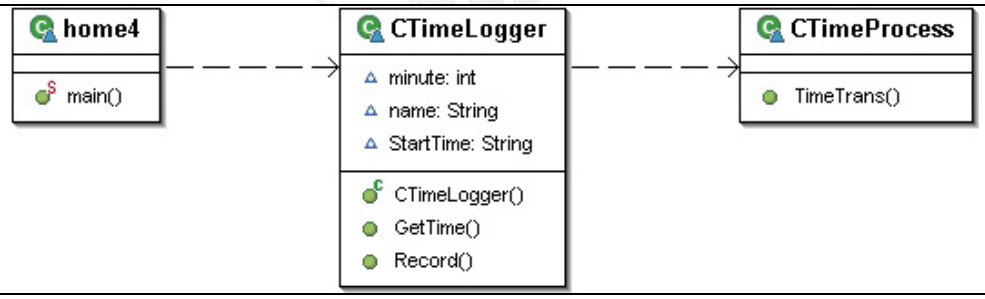
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 定義一 HomeworkTimeLogge 的類別可以紀錄學生設計程式的時間。
5	● 可以定義記錄時間的階段，例如 coding 階段，計畫階段，讀書階段等。
4	● 可以中斷，例如休息時段按個鍵就會暫停。
5	● 當專案結束時，會分階段統計所有的時間，並寫入一個檔案中。
5	● 安裝 Eclipse，並使用 Eclipse 撰寫此程式。
5	● 使用 source\Format 的功能將你的程式做排版。
5	● 每個公開的副程式(public method)，對該方法作詳細的說明。
5	● 一定要寫兩個 class 以上。

Time Log (單位：分鐘)		
時間	階段	詳細說明
43	Study	● 閱讀 Java：Date 這個物件，以存取系統時間。
		● 安裝並練習 Eclipse。
		● 閱讀 Java 註解的撰寫格式，以產生易讀的 API 說明文件。
		● 閱讀 Java 封裝物件的方式，以設計出一個可用的類別。
		● 查詢 API 內 Date 相關的 method，以存取及轉換時間。
21	Plan	● 於由要設計一個以上的類別，所以便須切割工作，再把工作交給各類別去執行。我的計畫是將檔案存取跟時間處理分開，接下來考量這些類別的行為為他們設計 method。

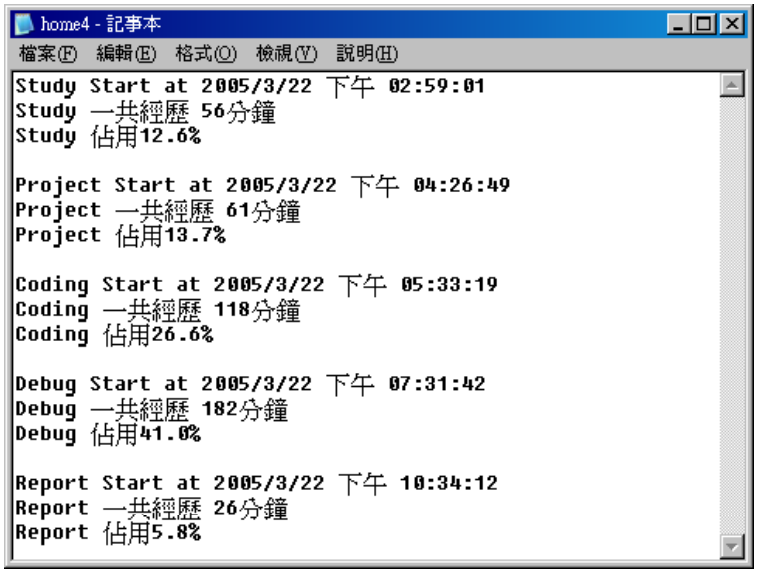
104	Coding	● 依照計畫的流程實作程式。 ● 先由控制使用者輸入開始，再由使用者輸入決定計時的行為，開始、暫停或是停止，接下來將參數傳給處理時間的類別處理。處理完以後再將資料輸出至檔案。
32	Compile	● 因為對於時間類別的 method 不是很熟悉，導致常常誤用，以產生編譯甚至執行時期的錯誤。
21	Test	● 自行調整系統時間，一邊用程式紀錄，測試有無邏輯錯誤。
12	Refactoring	● 用 format 功能整理程式碼的排版、重新整理類別內封裝的 method 以及 field、依照特定格式撰寫註解。
30	Report	● 整理輸出檔，由 Javadoc 產生的 API 文件。

詳細資訊					
類別數目 ˇ	3				
邏輯複雜度	2		預估	實際	誤差(%)
需求複雜度	2	行數 (行)	150	101	-32.6%
技術複雜度	2	時間 (分)	180	263	46.1%
總複雜度	6	產量 (FP/m)	5	2.3	53.9%

程式架構



執行結果



心得：本專案讓我學會設計類別，依照需求找出主要的類別。設計包跨類別的靜態成員 field(屬性)、動態成員 method(方法)。這個計時程式我將使用者輸入的指令傳遞給時間處理類別，資料處理好以後再輸出至檔案。學習使用 Date 物件，抓取即時的系統時間。getTime()是 Java 以 1970 年為基準，回傳經歷的毫秒數。toLocaleString()可以回傳當前的時間，這兩個算是比較常用到的 method。學習使用 Eclipse。Eclipse 是由 IBM 所開發的 Java IDE 工具，有著友善的介面以及即時監控程式碼的能力，令人著迷。另外，Eclipse 也可以很容易製作出 API 參考文件、製作可執行的 jar 檔、讓從事開發的人員自行研發 plug-in，可以說是集眾多優點於一身的 Java 開發工具。關於關於 Eclipse 身世背景可以參考以下連結：http://www.taiwan.cnet.com/enterprise/topic/0,2000062938,20096842,00.htm。

5A：“Project Manager”

專案資訊	
專案名稱	Project Manager
建立日期	2005/4/4
專案描述	GUI 程式

學習成果表 (成效：1-5)	
成效	學習目的
5	● 學習設計 GUI(Graphics User Interface)程式。

功能需求表 (完成度：1-5)	
完成度	功能需求說明

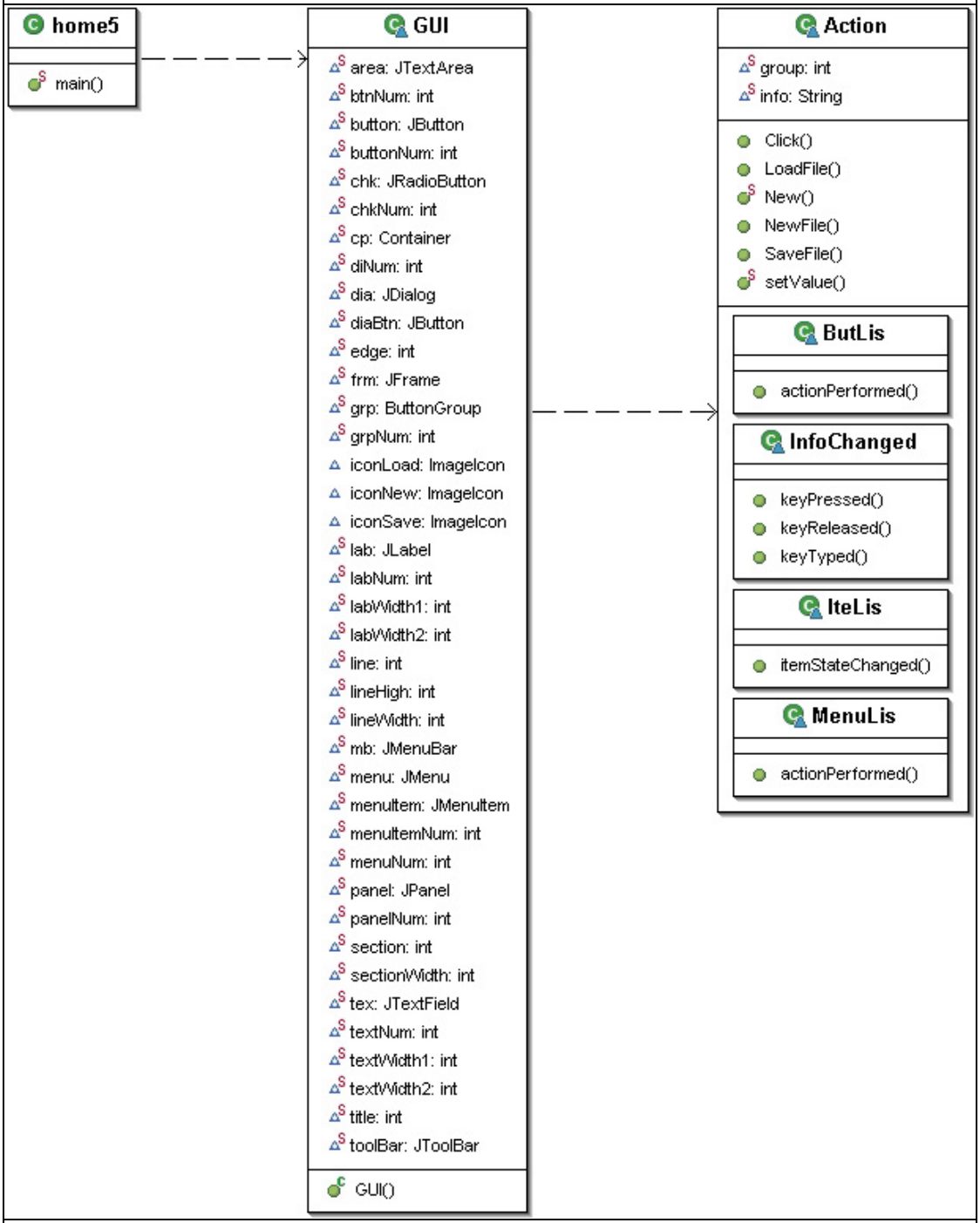
5	● 利用 JFrame 設計一個 GUI 畫面，可以讓使用者輸入以下資料：計畫 ID、計畫名稱、計畫敘述、計畫預估行數、計畫真實行數。計畫預估複雜度(包含技術複雜度、需求複雜度、邏輯複雜度)，每一個值介於 1 - 5 之間。
5	● 計畫複雜度=技術複雜度+需求複雜度+邏輯複雜度，並在畫面上呈現出此資訊。
5	● 當使用者按確定後，該資料會以檔案的形式存入。
4	● 可以 load 檔案，以作資料的修改。
5	● 至少兩個 class，一個作畫面的處理，另一個作檔案寫入與讀出的動作。
5	● 使用 Eclipse 的 project--> Generate Javadoc 產生此專案程式的 API.，觀察你所產生的 API 是否清楚的說明你的程式？擷取部分 重要的畫面放在報告中。
5	● 需交紙張報告，報告包含專案資訊(id, name, description)、student information、time log、需求查核表、執行畫面與說明、API、code

Time Log (單位：分鐘)		
時間	階段	詳細說明
131	Study	● 閱讀 AWT、Swing 以作畫面。
		● 閱讀委派事件模式(delegation event model)，由於委派事件模式牽涉到實作(implements)介面(interface)以及繼承(inheritance)，直接切入物件導向的核心，因此這部分相當重要。
		● 閱讀 Java 註解的撰寫格式，以產生易讀的 API 說明文件。
		● 閱讀類別的繼承，介面的實作。不論是物件導向，或是設計視窗程式，這些都是相當重要的。
45	Plan	● 先由設計視窗介面開始，考慮如何設計出一個有親合力的介面，接下來設計事件處理的類別。另外要考慮檔案寫入的格式以利日後讀取。
189	Coding	● 處理圖形介面的工作我交給 GUI 類別處理，並將圖形的產生寫在建構元(Constructor)裡，由生成這個物件直接產生圖形，接下來撰寫事件處理的類別，這邊也是 Java 比較困難的部分。
42	Compile	● 第一次撰寫圖形介面，不是很熟悉，會有放上去的元件沒有正常顯示的狀況，也因為對於委派事件模式不熟悉，常常無法正確觸發事件。
45	Test	● 這個程式主要是測試檔案讀寫。
52	Refactoring	● 因為程式碼蠻大的，所以依照處理畫面或是處理事件作分類來整理。
21	Report	● 整理輸出檔，由 Javadoc 產生的 API 文件。

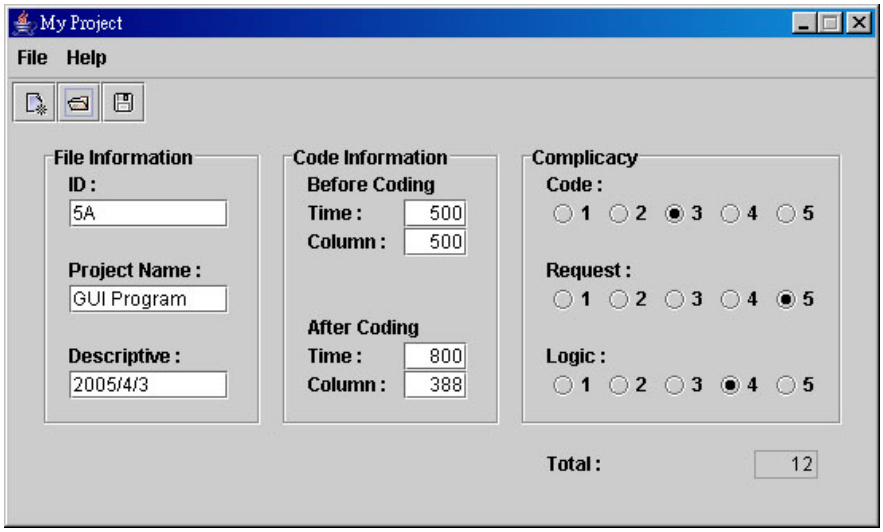
詳細資訊					
類別數目	3				
邏輯複雜度	2		預估	實際	誤差(%)

需求複雜度	4	行數 (行)	500	425	-15%
技術複雜度	4	時間 (分)	600	525	-12.5%
總複雜度	10	產量 (FP/m)	8.3	8	-2%

程式架構



執行結果



心得：這是我學習程式兩年以來第一次寫視窗程式，因此蠻期待的。AWT(Abstract Windowing Toolkit)是用來處理圖形(包含視窗與繪圖的最基本處理方式) ，AWT 也可以用來建立 Java 的 Applet 程式。另一種繼承 AWT 而來的類別庫 Swing，則提供了更佳的執行效能、更多的元件以及更好看的外觀。Swing 可以說是 Java 的未來架構，Swing 所提供的物件遠超過 AWT。目前 SUN 也已經不再擴充 AWT，而致力於發展 Swing 了。因為學習 Swing 引發我的興趣，因此特別去買了一本由 Matthew Robinson & Pavel Vorobiev 共同編著 Swing 實作手冊[4]。裡面有著許多豐富而且精采的範例，對於各種 Swing 元件也有詳細的解說，你可以在 http://www.manning.com/books/robinson2 下載到本書的所有範例。另外，在學習視窗程式的設計時，也必須考量使用者介面是否具有親和力，這是和之前的程式設計一個相當大的不同，亦是視窗程式設計有趣的一面。

許多科技產品的成功，其中一項重要的原因就是有良好的人機互動，比如說在繪圖界屹立不搖的 Adobe Photoshop、MP3 當紅炸子機 Apple iPod，都有著極具親和力的人機介面，因而在市場上受到廣大的歡迎。而當我實際寫過這個程式後，也發現控制使用者介面的程式碼佔去一半以上，因此學習設計人機介面，提供使用者一個簡單易用的操作環境，也是相當重要的。相關資料參考如下：http://www.adobe.com/ 及 http://www.apple.com.tw/。此外，Javadoc 是一個很方便的小程式，可以協助程式開發者很容易的產生 API 說明文件。這對於大型程式的發展相當有用，這表示你所開發出來的元件，未來可以更容易被其他程式設計師所使用。也因為如此老師要求我們一定要用 Javadoc 來產生系統的 API，而最棒的是他可以與程式的註解互動，例如在在註解中 @param 對 Javadoc 而言是有意義的，可以做出很美觀的 API 文件。

6A “Object Design- Using UML”

專案資訊	
專案名稱	Object Design- Using UML
建立日期	2005/4/29
專案描述	物件設計

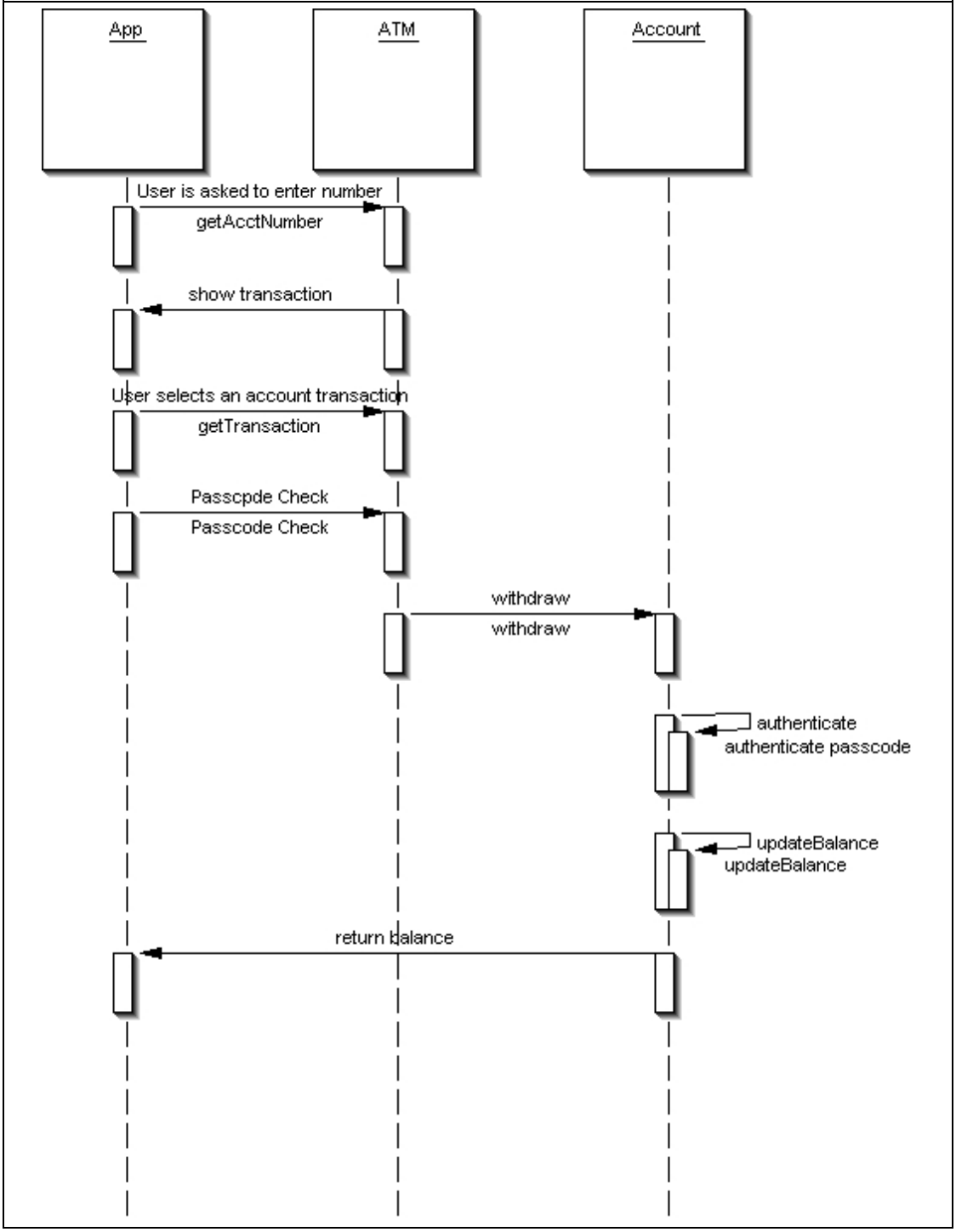
學習成果表 (成效：1-5)	
成效	學習目的
4	● 本次專案的主要目的在學習如何透過系統化的方式完成物件設計與實做，並瞭解 UML 對設計上的協助。

功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 請到 http://www.omondo.com/下載 Omondo(free UML)，並安裝至 Eclipse。
5	● 依步驟一步步的完成 ATM （課本 7.4）實例。
5	● 重複 2 所描述的方法，完成 Dispatching Repairpeople。
4	● 將以上兩個演練寫在 word 中，UML 的圖必須穿插其中。
5	● 印出及上傳繳交。

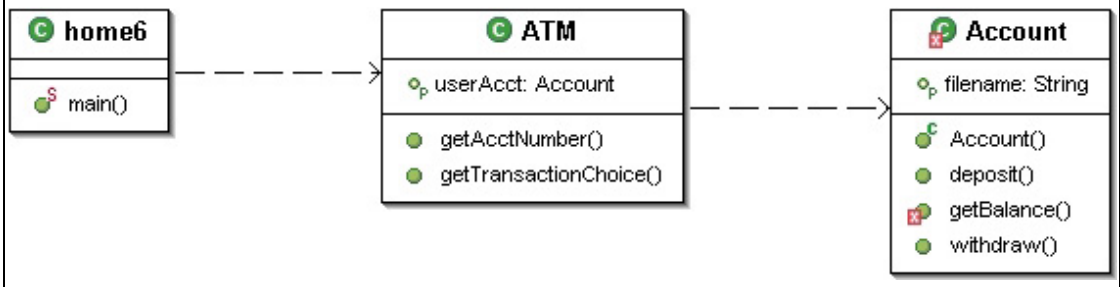
Time Log (單位：分鐘)		
時間	階段	詳細說明
15	Study	● 閱讀課本第六章，以了解設計類別正確步驟。 ● 安裝並且熟悉 Omondo 的使用。
12	Plan	● 依據課本第六章設計物件的正確步驟，搭配 Omondo 完成類別的設計。 ● 敘述問題 ● 描述流程 ● 找出主要物件 ● 找出物件的行為 ● 設計類別的介面 ● 找出 Instance Variables ● 實作 method
0	Coding	● 本專案沒有 Coding 部分。
0	Compile	● 本專案沒有 Compile 部分。
0	Test	● 本專案沒有 Test 部分。
0	Refactoring	● 本專案沒有 Refactoring 部分。
20	Report	● 整理 class diagram、sequence diagram，並詳列實作步驟。

詳細資訊					
類別數目	3				
邏輯複雜度	1		預估	實際	誤差(%)
需求複雜度	1	行數 (行)	0	0	0
技術複雜度	1	時間 (分)	60	47	-21.6
總複雜度	3	產量 (FP/m)	0	0	0

程式架構



System Dynamic Model with Sequence Diagram



Class Diagram

執行結果

本專案無執行結果

心得：這是我第一次使用 UML 軟體輔助程式設計。Omondo 是一個 UML 軟體，可以輔助程式設計，與 Eclipse 相輔相成，產生基本的程式碼架構。我們可以依據程式的需求建立 UML 圖形，再由 UML 圖形生成基本的程式架構，接下來實作這個程式架構，以完成整個專案開發。透過 UML，我可以從不同角度在 UML 跟程式碼之間切換，見樹又見林，如此一來，我們更容易掌握程式的開發了。此專案要求我們以下程序設計系統：

1. 敘述問題：描述要解決的問題，開出功能需求表。
2. 描述流程：我使用 Omondo 所提供的 Sequence Diagram 功能，描述物件的行為以及 message(信息)的傳遞。
3. 找出主要物件：依據需求訂出主要的物件。
4. 找出物件的行為：依據需求訂出物件的行為。
5. 設計類別的介面：利用 Omondo 我們可以很容易的設定物件的行為，也就是動態成員(method)。利用 Class Diagram 模式定義 method name、return type、argument 以及其他的修飾子（private、public、static 等等）。Omondo 會自動幫我們產生這些程式碼，我們只要實做這些產生的程式碼便可。
6. 找出 Instance Variables：依照需要，設計物件的屬性(靜態成員) Omondo 可以輔助我們設定屬性的型態、初始值，甚至可以幫我們完成 getX()、setX() 等等 method。
7. 實作 method：將定義好的 method 依照功能需求實作，如此便可完成整個程式。

以上設計物件的步驟，由 UML 輔助設計，我們可以從不同的角度來設計程式，兼顧程式的整體性，同時也顧及到了細節的部分。使用 UML 大幅提升我們設計程式的品質，同時也提升了開發的效率，在今日大型專案開發裡，扮演著重要的

腳色。除了 Omondo 之外，Power Design 等 UML 軟體也可以提供相似的功能。

7A “System Integration”

專案資訊	
專案名稱	System Integration
建立日期	2005/5/6
專案描述	GUI 整合程式

學習成果表 (成效：1-5)	
成效	學習目的
5	● 整合之前所開發的物件，製作一個進階版的專案管理系統。

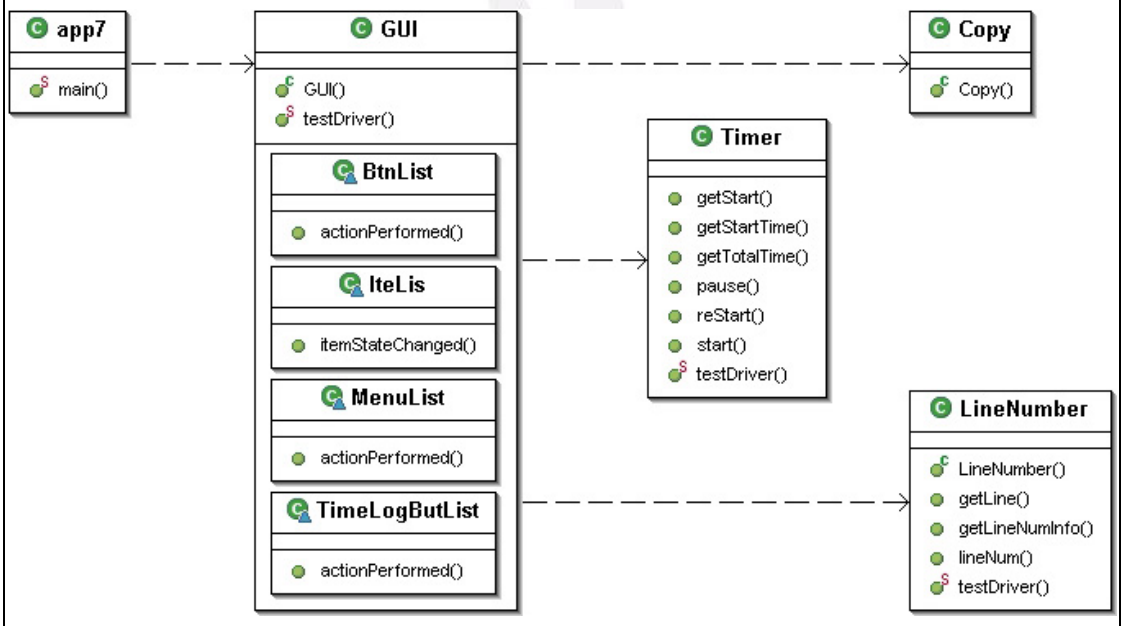
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 當專案開始時，可開啟此系統輸入專案的資料與預估值。
5	● 隨著不同的階段進行，紀錄每個階段的時間。
5	● 當專案完成後，可利用此系統將專案的檔案備份。
5	● 依照 3 所產生的備份檔，系統算出此專案的 LOC 等衡量值。
5	● 系統比較預估值與真實質的誤差。
5	● 加入新的實際值以算出新的產能。
5	● 設計你的測試案例，測試 test driver。
5	● 以 GUI 的介面整合此系統。

Time Log (單位：分鐘)		
時間	階段	詳細說明
0	Study	● 由於這個程式是整合之前的專案，所以沒有閱讀新的資訊。
75	Plan	● 要整合之前的專案並不是件容易的事情，尤其是剛開始學物件導向的人，通常物件設計的不好，程式碼 reuse 程度低，所以針對之前的功能，我全部重寫，設計好之後就可以被引用了。
585	Coding	● 由專案 5A 的想法延伸，設計出符合專案要求的介面，並且一一實作功能。我將計算程式碼、計時、事件處理、備份、圖形處理以物件為單位分別寫在不同的.java 檔裡，讓程式整體更有質感，也更好讀，其他與專案 5A 並無太大不同。
		● 備份的部分因為找不到 copy method, 所以只好使用檔案物件組合出一個複製檔案的程式碼。
		● testDriver 需用 static 修飾子。
55	Compile	● Compile 沒有預到太大的問題，大部分的是小細節的錯誤。

65	Test	● 由於這時沒有學到多執行緒，又得紀錄不同階段的時間，這裡發生很多問題，後來用了陣列解決，算是這個專案比較大的瓶頸。
71	Refactoring	● 程式碼蠻大的，所以依照功能設計物件，並存放在不同的.java檔裡。
26	Report	● 擷取並整理結果。

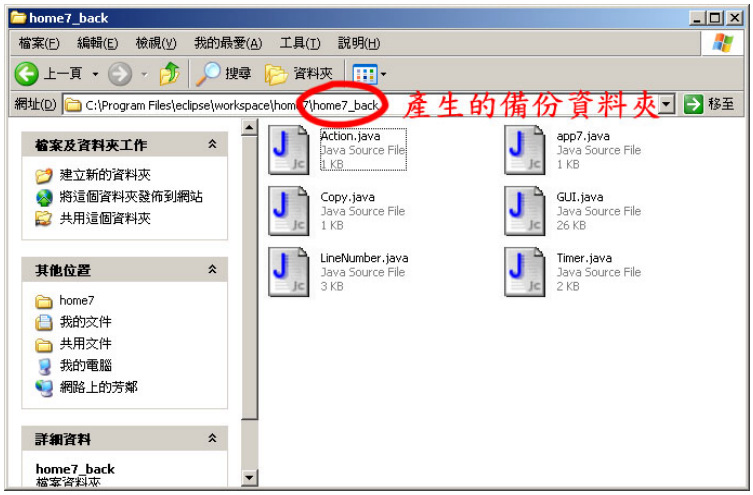
詳細資訊					
類別數目	5				
邏輯複雜度	3		預估	實際	誤差(%)
需求複雜度	4	行數 (行)	1000	989	-1.1%
技術複雜度	3	時間 (分)	600	877	46.1%
總複雜度	10	產量 (FP/m)	16.6	11.2	-32%

程式架構



執行結果


專案管理系統
檔案 說明
預估 執行 結果
專案資訊
專案名稱 7A
建立日期 2005/5/15
專案描述 軟體評估工具
預估行數 1000
預估時間 600
專案複雜度
邏輯複雜 1 2 3 4 5
需求複雜 1 2 3 4 5
程式複雜 1 2 3 4 5
總合複雜 12
下一頁
專案管理系統
檔案 說明
預估 執行 結果
執行專案的時間紀錄
開始 停止 閱讀書籍 2005/5/17 上午 02:03:09 經歷(分) 59
開始 停止 草擬計畫 2005/5/17 上午 03:03:09 經歷(分) 60
開始 停止 寫程式碼 2005/5/17 上午 04:03:12 經歷(分) 120
開始 停止 進入編譯 2005/5/17 上午 06:03:14 經歷(分) 60
開始 停止 測試程式 2005/5/17 上午 07:03:16 經歷(分) 120
開始 停止 重新整理 2005/5/17 上午 09:03:18 經歷(分) 120
開始 停止 整理報告 2005/5/17 上午 11:03:19 經歷(分) 60
專案開始 2005/5/17 上午 02:03:09
專案結束 2005/5/17 上午 11:03:19 經歷(分) 599
上一頁
下一頁
專案管理系統
檔案 說明
預估 執行 結果
預估資訊
專案名稱 7A
建立日期 2005/5/15
專案描述 軟體評估工具
檔案數目 5
邏輯複雜 4
需求複雜 5
程式複雜 3
總合複雜 12
專案報告
預估行數 1000
實際行數 913
誤差行數 -8.7%
預估時間 600
實際時間 599
誤差時間 -0.1%
預估產量 20
實際產量 18
誤差產量 -10.0%
上一頁
備份 / 報告



心得：這個程式整合了之前所開發的物件，可以測試我們之前所開發的物件 reuse 程度高不高。很不幸的是，之前所開發的物件都無法派上用場。可見開發物件，日後能不能很容易的被其他人重複使用，也是一門不容易的學問。物件設計的不好，也會造成日後大型程式發展整合的阻礙。物件導向的另一個好處是你可以將測試的程式碼 (test Driver) 一併封裝進物件裡。將測試的程式碼封裝進物件裡。我們可以節省大量人為輸入的時間，由直接呼叫 test Driver 觀察執行結果，便可以得知物件設計是否有錯誤，並且即時更正。方法是直接將 test Driver 宣告成 static，test Driver 測試這個物件的 method，並給一個正確的值與 method 的 return 作比對，直接輸出結果。如此一來只要呼叫 test Driver，便可以得知所設計的物件是否有錯誤，取代傳統的測試(必須由人為不斷輸入數據)。

8A “MVC and Thread”

專案資訊	
專案名稱	MVC and Thread
建立日期	2005/6/3
專案描述	MVC 架構與多執行緒

學習成果表 (成效：1-5)	
成效	學習目的
5	● 了解 MVC 架構以及 Java 的多執行緒功能。

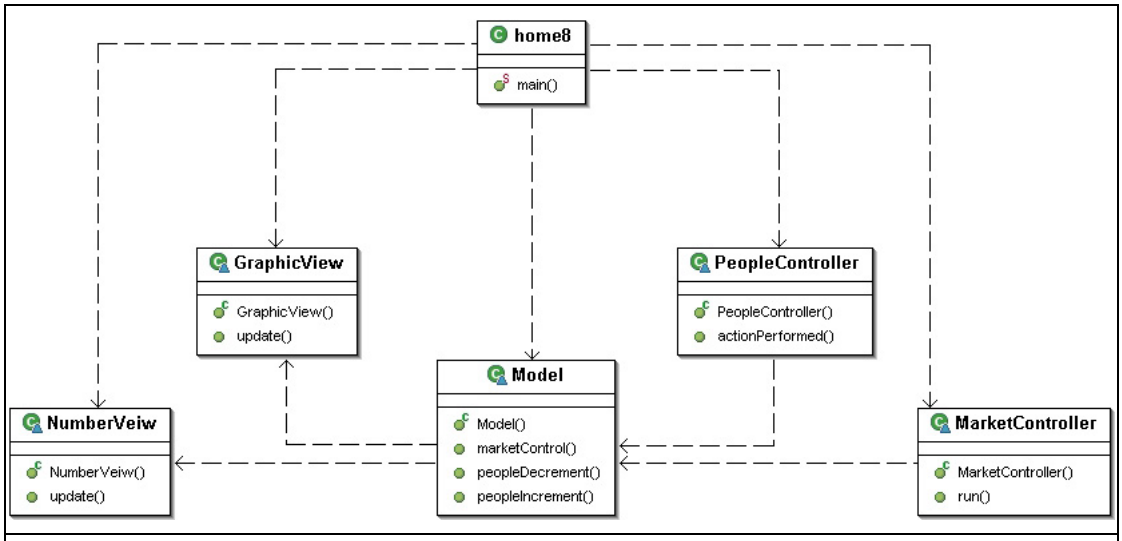
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 此用 MVC 架構設計此系統。
5	● 至少設計兩個 controller，一為市場，一為使用者。

5	● 至少設計兩個 view，一為數字，一為波浪圖。
5	● 當股價改變時，所有的 view 應即時反應股價。
5	● 參考課本 page481，multiple thread 的實例將兩個 controller 分別以 thread 設計，以便兩個 controller 都可以改變股價。
5	● 程式碼及執行結果印出，並上網繳交。
4	● 先上網預估程式碼大小及要花的時間，再開始寫程式。
5	● 抄襲一律零分。請使用 checklist 檢查 R1-R8 是否都有做到。

Time Log (單位：分鐘)		
時間	階段	詳細說明
34	Study	● 閱讀課本的 MVC 架構。 ● 閱讀課本以及輔助教材的多執行緒。 ● 參考老師放在 JQA 系統上的 MVC 以及 multi thread 範例。 ● 閱讀 AWT 的繪圖模式，以產生股價波動圖。
18	Plan	● 一定要先了解 MVC 架構的精神, 接下來要了解多執行緒在 Java 的實作，然後分別將股價改變模式放入企業邏輯模式、控制股價放入 controller、由圖表及數字呈現的放入 view。
89	Coding	● 將 JQA 系統的 MVC 範例抓下來，依照上面的架構增加另一個 controller-market，並用 multi thread 實作、另外增加一個 view 以顯示股價波動的圖形，圖形以 AWT 的繪圖完成。
12	Compile	● Compile 沒有預到太大的問題，大部分的是小細節的錯誤。
23	Test	● 有一個隨機產生股價的邏輯錯誤，由於我取的亂數介於 1-00，我想除以 40 取餘數，表示正負 20 的範圍，沒想到造成亂數分部不均，股下一直下跌，後來花了一些時間才修正這個邏輯錯誤。
8	Refactoring	● 依照 MVC 架構整理程式碼，由於使用 MVC，程式的架構更為漂亮。
10	Report	● 擷取並整理結果。

詳細資訊					
類別數目	6				
邏輯複雜度	2		預估	實際	誤差(%)
需求複雜度	2	行數 (行)	200	158	-21%
技術複雜度	2	時間 (分)	180	186	3.3%
總複雜度	6	產量 (FP/m)	6.6	5.0	-23%

程式架構



執行結果

心得：MVC(Model、View、Controller)模式是現在當紅的架構。MVC 架構將企業邏輯模式、呈現方式、以及控制方式完全分離。在互相不甘擾的情況下，我們仍然可以藉由 Controller 改變 Model，再由 Model 通知 Observer。MVC 主要的精神就是將三者分離，使得日後程式設計師能夠獨立工作，互不影響，增加工作效率。MVC 的另一層意義是，當 Model 有改變時，才通知 View。如此便可以免除 View 必須持續監控 Model，造成系統資源的浪費。很幸運的，Java 已經有現成的 MVC 架構，我們只要直接引用便可以實踐 MVC 架構了。在這個程式中，我設計了五個物件並遵循 MVC 架構，分別是 Model (企業邏輯模式)、NumberView (數字的呈現)、GraphicView (圖形的呈現)、MarketContrllor (市場控制模式)、PeopleContrller (人為控制模式)，如此有層次的程式分工，是之前沒有學過的。多執行緒也是 Java 另一個值得一提的特色。傳統程式語言一次只能執行一個程式區塊，Java 改變了這個機制，一次可以同時執行多個程式區塊，提高程式的執行效率。這個程式裡，我們將“市場”交給一個執行緒，因此我們同時間還可以作

其他事情。有了多執行緒的程式設計，可以讓我們更妥善的使用資源，讓電腦執行時更有效率。

延伸學習 “2 Pass Assembler”

這個專案並不是老師派的作業，而是我自己有興趣寫的。

專案資訊	
專案名稱	2Pass Assembler
建立日期	2005/5/10
專案描述	兩次執行的組譯器

學習成果表 (成效：1-5)	
成效	學習目的
5	● 了解 SIC Assembler 並實作之。

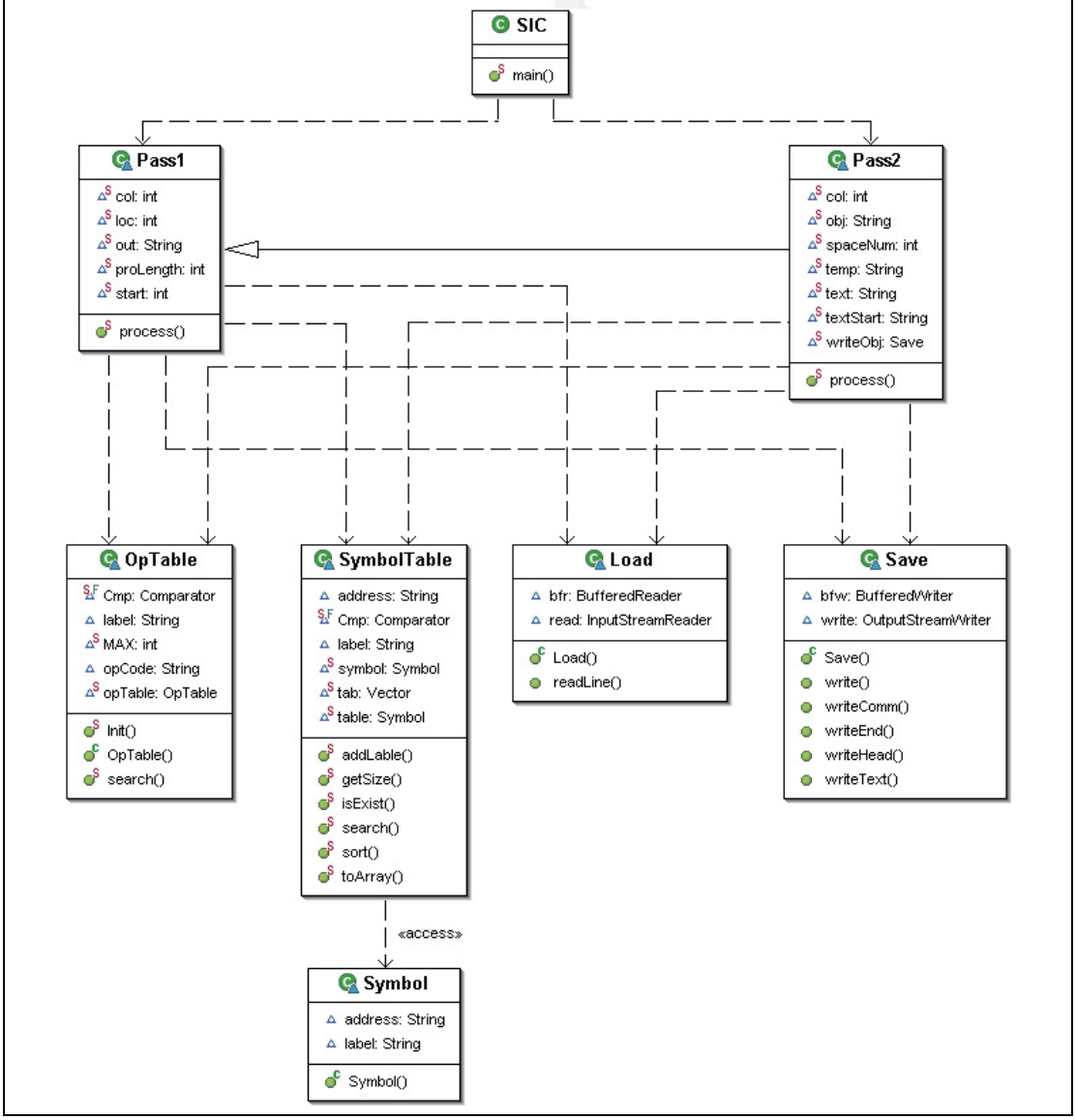
功能需求表 (完成度：1-5)	
完成度	功能需求說明
5	● 必須輸出中介檔(Intermediate)。
5	● 必須輸出清單(List)。
5	● 必須輸出目的碼(Object)。
5	● 以自己熟悉的語言實作之。

Time Log (單位：分鐘)		
時間	階段	詳細說明
42	Study	● 閱讀 SIC 2Pass Assembler 演算法。
33	Plan	● 一定要先了解 SIC 2Pass Assembler 演算法，然後再以 Java 的方是實作。 ● 需要用到一個符號表、一個運算碼表，資料結構通常維雜湊表格，因為這是最快的。除此之外我還設計了其他物件，分別是 Pass1、Pass2、以及存檔、讀檔物件。
681	Coding	● Pass1：讀進輸入檔，先作字串的切割，依照空白或是 tab。然後去掉備註欄，判斷結構是否為標籤、運算碼、運算元。接下來為每個指令加入記憶體位置，建立符號表格，寫入中介檔(Intermediate)。 ● Pass2：讀入中介檔，由查表組譯運算碼以及運算元，假指令的部分則依照指令保留記憶體位置，然後分別輸入 List 及 Object。
33	Compile	● Compile 沒有預到太大的問題，大部分的是小細節的錯誤。

28	Test	● 比較輸出檔與課本的異同，不同則是代表程式邏輯有誤。 ● 由於組譯器需要用到索引法而課本的演算法沒有，因此這部分除錯花了一些時間。
21	Refactoring	● 依照 Pasa1，Pass2 的架構重新整程式。
16	Report	● 擷取並整理結果。

詳細資訊					
類別數目	8				
邏輯複雜度	4		預估	實際	誤差(%)
需求複雜度	4	行數 (行)	500	575	15
技術複雜度	4	時間 (分)	900	854	-5.1
總複雜度	12	產量 (FP/m)	6.6	8.0	-21

程式架構

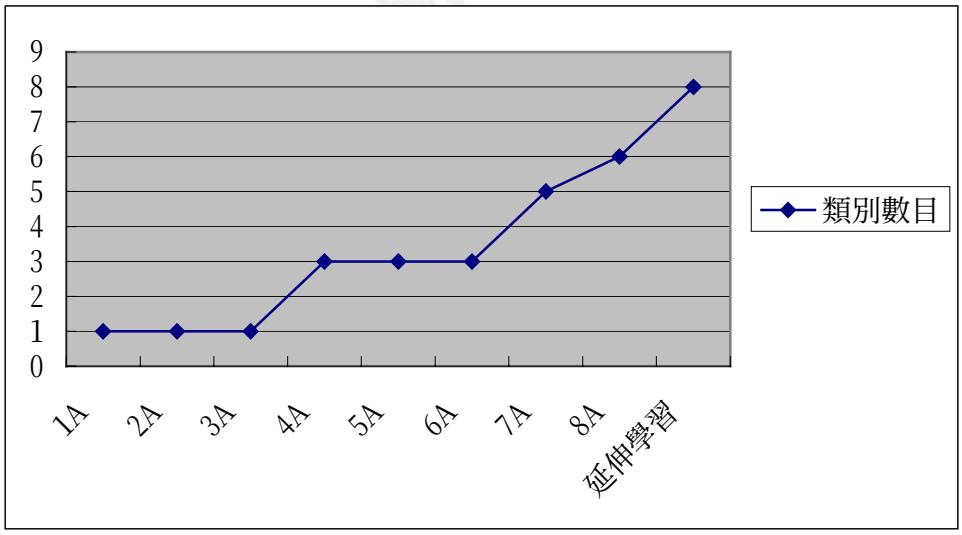


執行結果
HTEST 00100000107A
T00107A1E14103348203900103628103030101548206 13C100300102A0C103900102D
T00101E150C10364820610810334C0000454F46000003000000
T0020391E041030001030E0205D30203FD8205D2810303020575490392C205E38203F
T0020571C1010364C0000F1004096041030E02079302064509039DC20792C1036
T00207373820644C000005
E001000

心得：因為系統程式這門課需要開發組譯器 (Assembler) ，所以我嚐試用物件導向的方式並用 Java 實作 Assembler。我設計了七個物件：符號表儲存 (靜態)、符號表處理 (動態)、助記憶碼表、存檔、讀檔、Pass1、Pass2 物件。表格的資料結構是雜湊表格，我以 binary-search 取代，由於課本演算法沒有提到索引法，因此索引法的指令翻錯，除錯花了一些時間。這個程式必須要有資料結構的基礎，也要對組譯器夠了解，還需要夠熟悉 Java 的語法，因此是對自己能力的一種考驗，我寫出來實在算蠻高興，蠻有成就感的。

四、專案量化分析

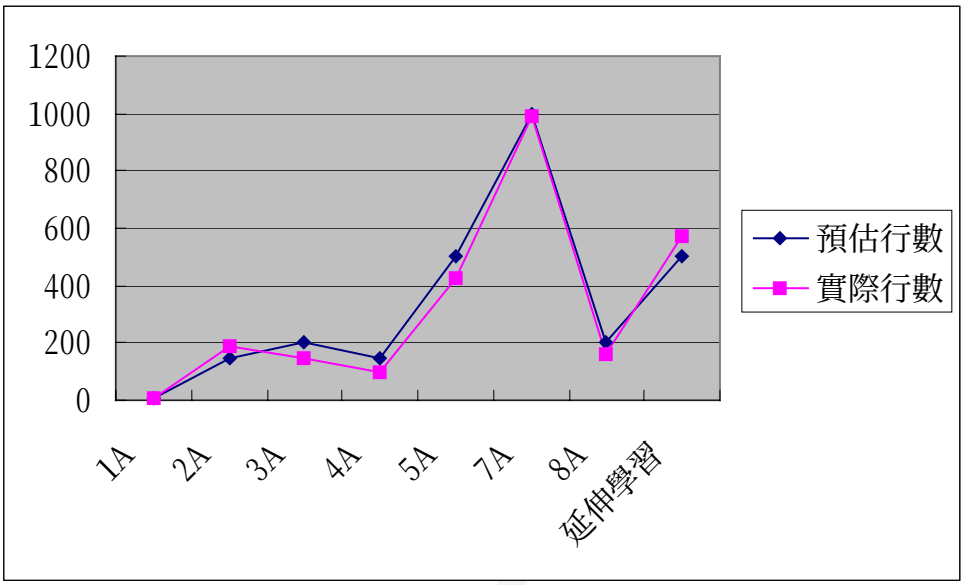
我們課程有一個 JQAS（Java Quality Assurance System）學習輔助系統，可以輔助學生作專案複雜度分析、程式大小預估、專案開發時間預估、產能計算等等，可以隨時紀錄自己開發專案的情況。以下是參考 JQAS 系所作的專案量化分析：



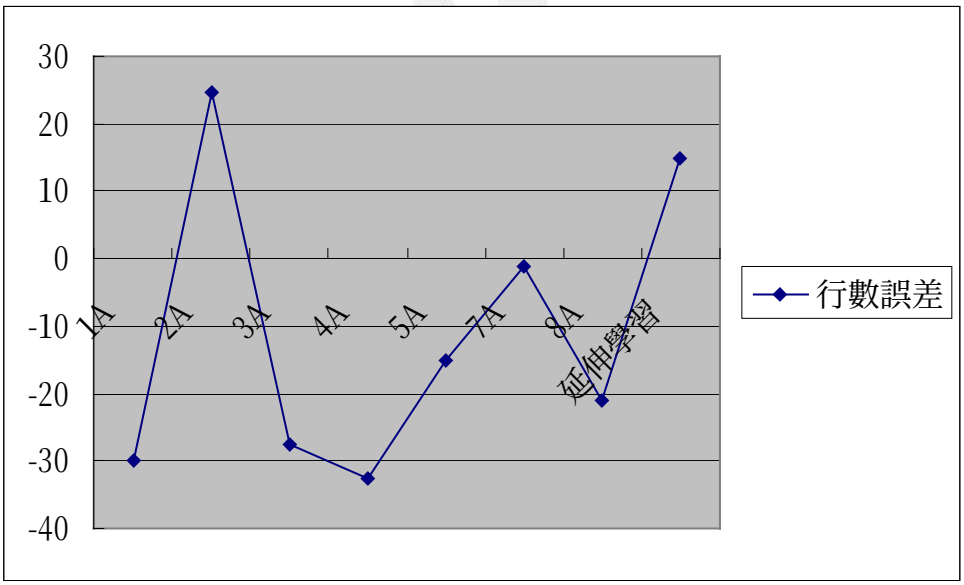
圖一：專案類別數目曲線圖

這是專案中使用類別數目的曲線圖。一開始的專案的重點在熟悉工具、環境與語法，所以都僅用的一個類別，隨著專案越來越大，所用的類別數目也越來越多，

也表示我對物件設計的模組觀念越來越清楚了。



圖二：程式碼行數曲線圖 (Y 軸單位:LOC)

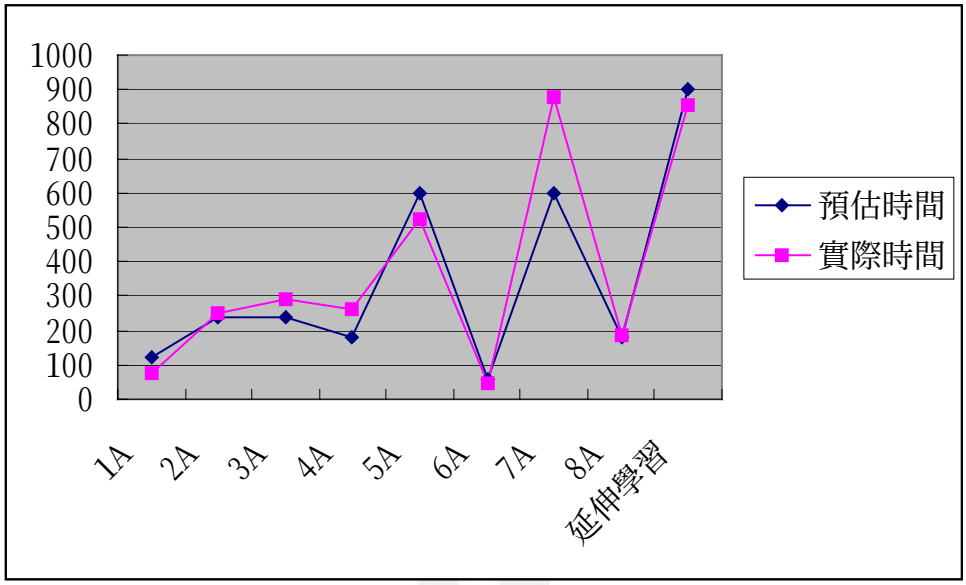


圖三：程式碼預估誤差圖 (Y 軸單位:百分比)

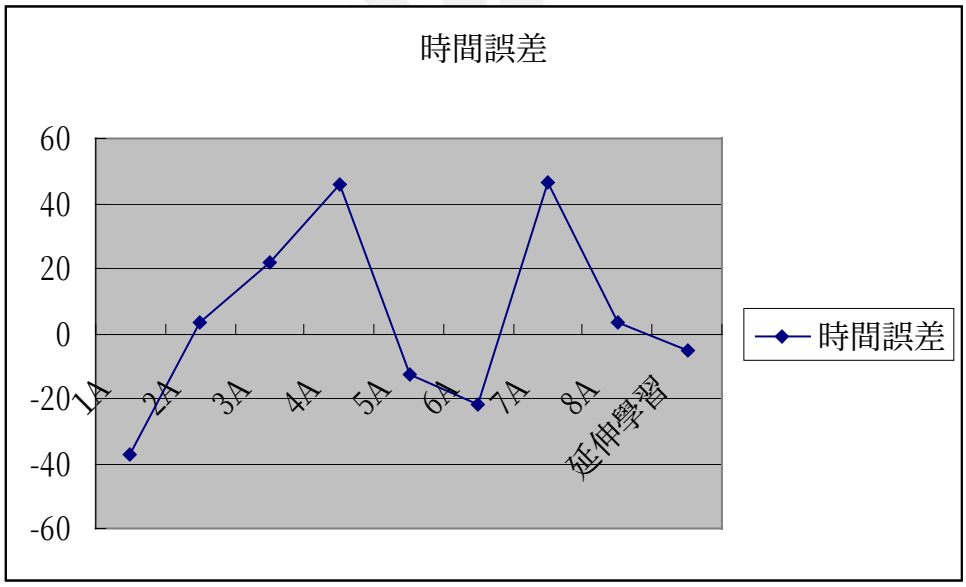
程式碼行數（Line of Code; LOC）代表程式的大小，我寫程式時盡量以最精簡的方式完成需求。由此圖可以發現到後面的程式越來越大，功能也越來越複雜。不過掌控程式碼的大小還是比掌控開發時間容易多了。

由圖三可以發現曲線有越來越接近 0 的趨勢，特別是在專案 5A-7A，這表示我越來越清楚 Java 需要多少的程式碼便可以完成需求了，所以預估的能力越來越精準了。我想這多半是歸功於在前面的專案裡有收集我的產量吧。請注意圖二與

圖三並沒有 6A 的資料，這是因為 6A 這個作業老師並沒有要我設計程式，僅是將課本中的題目轉為 UML 的設計圖而已，故其程式碼行數不計。圖六亦是相同的情況。



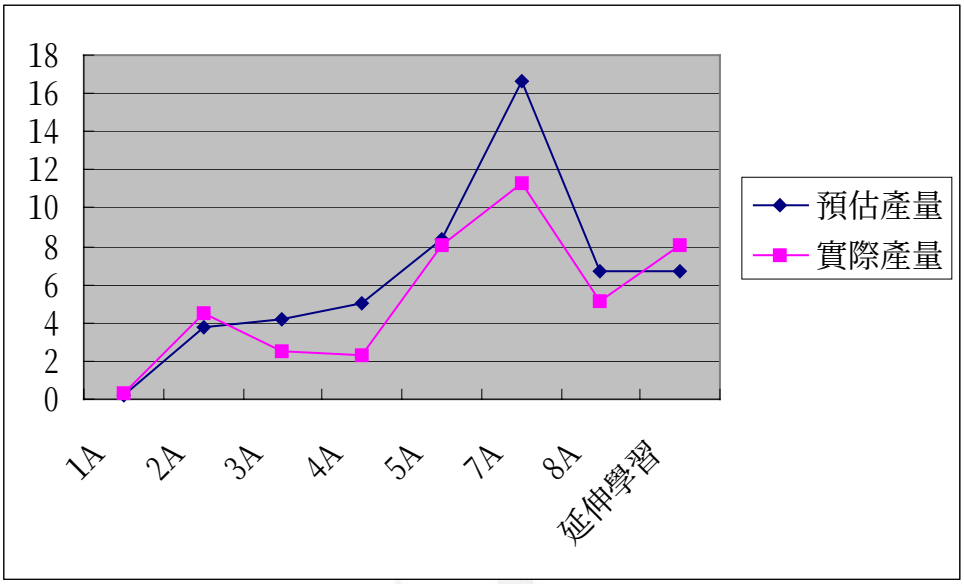
圖四：專案開發時間曲線圖 (Y 軸單位：分鐘)



圖五：專案開發時間預估誤差圖 (Y 軸單位：百分比)

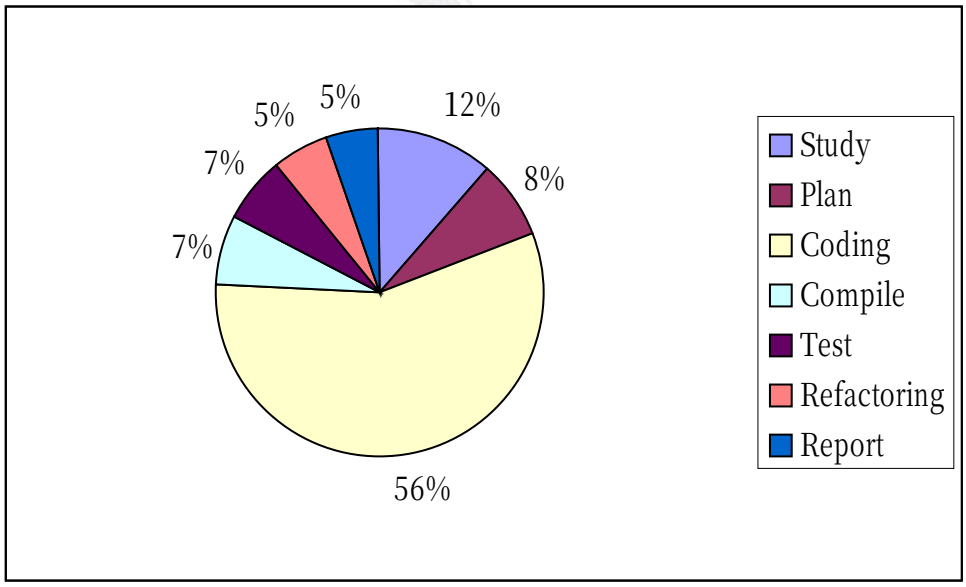
專案開發時間包含閱讀資料、計畫、實做、編譯、測試、整理及報告，可能是因為我第一次接觸 java，所以需要花多少時間閱讀，需要花多少時間除錯還拿捏的不是很準，從圖中可以看到越大型的專案時間誤差的越大。另外一個原因是開發中會有很多因素影響進度，比如說意外的語意錯誤、邏輯錯誤、或是自己突然想利用不同的方法完成需求等。儘管開發時間比較難掌握，但如果扣除 7A，其實

我的預估誤差還是有越來越小的趨勢。



圖六：產量曲線圖 (Y 軸單位：行/分鐘)

圖六可以發現我對 Java 可以說是漸入佳境，因此產量漸漸的提高。雖然四個月的課程中沒有太巨量的變化，不過我相信未來多加練習一定可以讓程式的產量更高的。5A、7A、8A 程式產量偏高，是因為程式碼較大，因此佔專案開發時間比重較重的原因，因此產量較高。



圖七：專案開發的時間分布圖

這是開發專案的時間分布圖，Coding 還是佔了絕大部分，其次因為第一次學習

物件導向設計，因此每開發一個新專案就必須研讀新的資訊，因此也佔用不少時間。老師說一般的人在 compiler 與 testing 也會花不少的時間，這是因為沒有仔細的思考程式的邏輯，因此在 compiler 與 testing 階段花很多的時間除錯。依老師的說法我寫程式的習慣還不錯。Report 雖然僅佔了所有專案時間的 5%，可是對我學習專案有很大的幫助。

(五)：自評與學習心得

經過一個學期的學習，1A-8A 的專案磨練之後，對於物件導向以及 Java 都與不少的認識。不但學會了用 Java 開發程式，也學會了物件導向的精神與意義。另外就是業界流行的 Eclipse、UML 輔助設計與 MVC 架構，可以說是獲益匪淺。我對這門課的期許是希望老師能加入更多 UML 的訓練以及培養學生整合的能力因為台灣軟體環境欠缺整合，台灣的學生也缺乏這方面的能力，因此希望能在這門課上獲得更多這方面的資訊。

表 學期學習自評(1-10)

成效	學習項目
10	練習 J2SDK。
7	練習簡易的 Java 指令。
10	練習檔案的存取。
9	練習基本的程式流程：變數宣告、輸入輸出、分支結構、迴圈結構、陣列等。
10	Composition(包含)或者是 Cascading(連結) 。
10	了解 Java 註解撰寫的格式。
8	練習網路資料的讀取。
9	練習字串的處理。
10	練習檔案物件的使用。
9	練習排序。
8	練習設計物件。
10	時間類別的使用。
8	練習使用 Eclipse。
9	設計 GUI 程式。
10	了解委派事件模式(delegation event mode) 。
10	如何使用 Eclipse 產生 API 說明文件。
10	了解物件的繼承、介面的實作，以寫出視窗程式。
8	安裝並練習 Omendo。
7	了解 UML 與物件導向程式設計的關係。
10	以 UML 輔助設計物件。
8	整合所開發出來的物件。
10	封裝 test Driver。

10	了解 MVC 架構。
9	認識多執行緒。
9	學習 AWT 繪圖模式。

參考資料

[1] R.S. Pressman. Software Engineering – A Practitioner's Approach. Mc graw Hill.2001.

[2] D.M. Arnow, S. Dexter and G. Weiss. Introduction Programming Using Java. Pearson, Addison Wesley, 2004.

[3] 洪維恩. Java2 教學手冊，博碩出版出版社，2003.

[4] M. Robinson and P. Vorobiev，Swing 實作手冊，鄒修銘、吳俊儀、胡凱智譯，博碩出版出版社，2003。

[5] Java 技術論壇，http://www.javaworld.com.tw/jute/。

[6] W.S. Humphrey. A Discipline for Software Engineering. Addison Wesley, 1995.