逢 甲 大 學

資 訊 工 程 學 系

碩 士 論 文

Git Education Game - 程式碼版本控制學習遊戲之研發

Git Education Game - A Game-based Approach to Learning Version Control of Programming

指導教授：陳錫民

研 究 生：張佑瑋

中 華 民 國 一 百 一 十 一 年 六 月

誌謝

首先感謝我的指導教授，陳錫民教授，在研讀碩士期間我學會許多大學時無法未接觸的領域與技術知識，在論文研究過程中，教授很早地指導我去進行各項研究，使我在較早的時間下就大致上完成了研究實作與論文的雛型，並經常題點我研究方向，使我的研究得以更加順利並擁有更多豐富的內容。

感謝曾經的實驗室學長，讓我在剛進入實驗室期間能有人問問題，快速進入狀況，學會許多以前未接觸的新知，還有感謝一起在實驗室努力的同學，讓我在修課及努力研究時有人能一起幫忙，以及感謝學弟妹們幫忙處理許多實驗室的雜務活，此外還要感謝往昔的同學們，努力找出時間一起出遊使我在面對沉悶的疫情與研究壓力轟炸時仍能放鬆身心。

碩士的生活一晃而過，雖然在這段期間也有過些不順利的時候，但這些困難或麻煩終究在各種努力下得以解決，由衷感謝在這段期間所有幫助過我的人，讓我在這段期間可以度過。

摘要

版本控制系統對於軟體行業是不可或缺的工具，身為工程師必然需要具備使用版控工具的能力，然而在大多數大學教育中對於此項技術的教育著墨不多，傳統的教學方式也使得學生對於 Git 容易有概念上的混淆以及使用上的障礙。本研究提出了一個名為 GEG 的嚴肅遊戲用於教授 Git 的概念與使用方法，目的是改善學生的學習動機，並實現相對傳統授課更深入的學習。

本研究設計了一個實驗，在實驗中同一門課程的兩個班級被分為實驗組與控制組，為了測量遊戲對學生的影響，設計了一個測驗驗證遊戲是否能夠幫助學生獲得更好的學習成果，並設計了一個問卷調查遊戲是否對學生的動機有積極的影響。結果顯示，遊戲在學生的動機方面有積極的影響，在學習成果方面，儘管在最困難的課題中通過率並不高，但實驗組在每一項目中仍舊比控制組擁有更高的通過率。

關鍵詞: 遊戲式學習、版本控制、軟體工程、教育遊戲、嚴肅遊戲

Abstract

Version control systems are indispensable tools for the software industry, and engineers must have the ability to use version control tools. However, in most university education, not much attention has been paid to this technology, and the traditional teaching methods make students easily confused about the concept of Git and the obstacles in using it.

This study proposes a serious game called GEG to teach the concept and use of Git, with the aim of improving students' motivation and achieving deeper learning than traditional lectures. In this study, an experiment was designed in which two classes of the same course were divided into an experimental group and a control group. In order to measure the effect of the game on students, a test was designed to verify whether the game could help students achieve better learning outcomes, and a questionnaire was designed to investigate whether the game had a positive effect on students' motivation. The results showed that the game had a positive effect on students' motivation and that the experimental group had a higher pass rate than the control group for each item, although the pass rate for the most difficult items was not as high.

Keywords: *Game-based Learning, Version Control, Git, Software Engineering, Educational Game, Serious Game*

目錄

圖目錄

表目錄

第一章 緒論

1.1 研究背景

版本控制工具的能力是軟體工程師必不可缺的技能[1]，然而這項必備的技能並不一定會被當作計算機科學課程的一部分，有教學的課程中通常也使用很短的時間來教學這項技術，這使得學生對於版本控制工具的概念及使用方式等等混淆不清，傳統的教學方式也進一步導致這個現象更為嚴重，學生難以具有學習的動力、也難以理解教學內容。

在非傳統教育方式當中，基於遊戲的學習被認為是一種有潛力的教學方式，提高使用者的積極性被認為是遊戲式學習的一個關鍵特徵[2]，基於遊戲的學習作為一種教學方式可以強化學生的內在動機[3]，增加主動學習的意願，虛擬的環境中可以模擬各種課堂上難以立即重現的情境，並透過互動的方式使教學抽象概念更加容易理解。遊戲可以提供學生不停嘗試的機會，不必擔憂操作失敗可能帶來的風險，即時回饋也增進了學習的效率。由於遊戲的內在特性，例如競爭、挑戰、互動，他們能將學習過程轉變為有趣的體驗，並在可接受的教學時間和教師負擔範圍內實現深度學習[3]，因此遊戲式學習能夠有效改善學生的學習動機、學習效率。

1.2 研究目的

在本篇論文中我們將介紹我們的遊戲式學習方法，通過名為 Git Education Game（以下簡稱 GEG）的遊戲來教學 Git。GEG 是一款基於 Web 的嚴肅遊戲，它是由 Unity 結合 Java Server Pages (JSP)開發而成的，用於教學 Git 的概念與指令，目的是為了有效改善學生對於 Git 的學習動機、學習效率，並補足學校課程中所不足的部份。此遊戲將 Git 的概念與指令分為數道關卡，並引入遊戲元素如：點數、獎章、排行榜等等用以激勵學生參與學習，從學習中獲得成就感，鼓勵學

生在模擬的環境中不斷嘗試，而學生的學習行為則經由 API 發送至後台的資料庫當中，教師可以即時監控學生的學習狀況。

1.3 研究問題

為了檢驗所提出的遊戲的有效性，本研究設計了一個教育研究實驗，在實驗中將同一門課程的兩個班級分為實驗組及控制組，實驗組通過 GEG 學習 Git，對照組則以傳統的方式學習 Git，我們通過實驗評估學習效果，以回答以下研究問題:

- 研究問題 1（RQ1）： 加入 GEG 作為教學輔助工具是否比傳統的授課方式具有更高的學習成果？
- 研究問題 2（RQ2）: GEG 作為教學輔助工具是否為學生對 Git 的態度及行為帶來正面影響？
- 研究問題 3（RQ3）： 會主動以 GEG 進行學習的學生比例是多少？主動進行學習的學生是否具有較高的學習成果？
- 研究問題 4（RQ4）: 學生認為 GEG 有何優點與缺失？

為了回答這些研究問題，本研究擴展並應用了 UTAUT2[4]和 PLS-SEM[5]，考察使用遊戲式學習的方式是否正面影響學生對 Git 的態度，並考察遊戲式學習對自我效能、表現預期及享樂主意動機等等因素的影響，本研究引入了績效預期、努力期望、享樂主義動機[6]、自我效能感[7]、態度[7]、行為意向、實際行為，並加入遊戲有用性、遊戲動機於分析模型。

1.4 研究原創性

專門用於教學程式碼控制技術的遊戲並不多，更多以文章的形式存在，而既有的系統則不見得具有足夠的娛樂性能激起學生的學習動機，它們大多為互動式教學軟體，缺乏遊戲元素。除此之外，也有系統模擬的指令過於簡化，在實際使用時無法作為參考，僅能學習工作流程或概念，表 1.1 列出了相似系統的比較。

表 1.1 相似系統比較

	Learning Git Branching	Oh My Git!	Git Education Game
採用模擬 Git 指令方式	O	X	O
Git 指令關卡指導	O	O	O
完整指令用法學習過程	X	O	O
額外的指令講解機制（指令卡片說明）	X	O	O
可互動的提示系統	O	O	X
遊戲元素或機制	X	X	O
紀錄使用者活動供評估學習狀況	X	X	O
網頁上可直接使用	O	X	O

我們所提出的系統 Git Education Game 作為輔助學習工具相較其他模擬 Git 指令的教具在實際使用上具更高的可參考性，而相對讓玩家直接使用 Git 指令的系統執行更加快速，並對某些具有較複雜參數的指令可以限縮在學習曲線較可令學生吸收的範圍內。

因為引入了遊戲元素，我們提出的系統遊戲機制更為豐富，能更佳促進學生的學習動機，這些機制也作為教具的一部分給予學習時的幫助，改善學生的學習效率。老師方面則可以藉由後台的數據觀察學生的學習狀況，即時加強教學不足的部份，促使學習的成果更高。

1.5 重要性

如研究背景所述，無論是資訊工程的學生還是業界人士，版本控制工具的使用能力是必備的，在軟體業界工作必然需要使用版本控制軟體，除此之外，參與

開源專案、開發開源專案、與開源社群互動，最基本的門檻也都是具備使用 Git 版本控制工具的能力，甚至現在熱門的 DevOps 技術所需要的持續整合、持續部署技術也同樣必須透過 Git 來觸發。

而透過設計遊戲來教學 Git 的優勢有以下兩點：

1. 電腦可以扮演成員與學生互動，給予問題環境，模擬現實中無法短時間立刻重現的情境
2. 真實的工作環境會害怕犯錯，遊戲可以避免這種情況盡可能鼓勵學生嘗試

這些優勢是採用其他教學方式難以實現的優勢，因此本研究論文所提出之系統具有其必要性。並且此系統可以幫助老師即時了解學生的學習狀況，針對學生無法即時理解的單元做加強，而引入遊戲元素的方式也能大大增加學生學習的參與度，刺激學生學習，以實現相對傳統教育更深入的學習。

1.6 論文結構

本文的第二章對相關的論文做回顧；第三章詳細介紹本論文提出的系統設計與方法；第四章為系統展示；第五章為系統實驗；第六章為結論與未來研究。

第二章 文獻回顧

2.1 嚴肅遊戲

嚴肅遊戲是應用於嚴肅目的的電腦遊戲，近幾十年來嚴肅遊戲與遊戲化都被用於開發用於嚴肅目的，兩者的定義不同，嚴肅遊戲以完整的遊戲為基礎，將娛樂作為次要，以教育為中心[8][9]，遊戲化則是將遊戲元素加入到非遊戲的環境之中[10]，但都試圖使用遊戲或遊戲元素來教育和改變行為模式[11]，在教育、健康等環境中教育、鼓勵、說服使用者[12][13]。本研究的主要目的，即是通過嚴肅遊戲，來教育、改變學生的版本控制軟體的認知，藉由引進遊戲化元素，建立遊戲機制等等來使嚴肅遊戲能夠獲得不同於普通遊戲的有益成果[14]。

2.2 遊戲元素

遊戲元素為吸引使用者的重要動力，Katie Seaborn [15]等人統整了遊戲元素，如表 2.1 所示，並提遊戲元素往往是相互關聯的，例如「級別」和「等級」可以指通過經驗獲得的等級，本研究引入其中的點數、獎章、排行榜等元素作為主要的遊戲機制。

表 2.1 傳統遊戲元素的術語

Term	Definition	Alternatives
Points	Numerical units indicating progress.	Experience points; score.
Badges	Visual icons signifying achievements.	Trophies.
Leaderboards	Display of ranks for comparison.	Rankings, scoreboard.
Progression	Milestones indicating progress.	Levelling, level up.
Status	Textual monikers indicating progress.	Title, ranks
Levels	Icreasingly difficult environments.	Stage, area, world.
Rewards	Tangible, desirable items.	Incentives, prizes, gifts.
Roles	Role-playing elements of character.	Class, character.

2.3 軟體工程中的教育遊戲

傳統授課方式只允許被動學習，教育軟體工程的過程難以提供足夠實用的知識，因此多位教育工作者軟體工程課程中融入了遊戲式學習的方法，如[16]中Alex Baker 等人開發了一款模擬軟體工程過程的教育紙牌遊戲，該研究描述了他們如何設計遊戲機制以令學生充分參與、了解軟體工程的流程與可能遭遇的問題，結果顯示該遊戲在引領學生入門方面取得了一定的成功，學生一致認為該遊戲使軟體工程概念更容易理解。[3]則描述了他們如何運用遊戲的內在特性，將教授Scrum 開發方法，將學生的學習過程轉變為有趣的體驗，並提到遊戲可以在可接受的教學時間和教師負擔範圍內實現深度學習，結果顯示由於遊戲的競爭性，激發了學生在學習的參與度，對學生的學習體驗及學習動機造成了積極的影響。因此本研究也選擇加入競爭元素，如排行榜機制來刺激學生產生學習動機。

2.4 業界期望與計算機教育

在畢業生進入勞動力市場的能力研究中，團隊技能、協作與這些技能的工具被提到需要改進，[17]中提到了行業期望與畢業生能力的差異，文獻中提到的能力與團隊如何管理軟體有相當的關係。[18]提到了 Git 是一個成熟的、廣受好評的程式碼版本控制系統，並提到對業界標準工具經驗缺乏的學生來說，被充分限制了參與實習的能力，並講述了他如何引入 Git 作為一種機制，用於發布課程練習，[19]中介紹了他們在計算機科學中使用 Git 作為他們課程平台的經驗，並認為從課程的角度來看。因此本研究提出以 Git 為主題的遊戲化教育實驗，用以增進學生對 Git 的掌握與學習動機、態度。

2.5 延伸整合科技接受模型

在教育技術背景中，技術接受模型（TAM）及其後繼者被廣泛採用，延伸整合科技接受模型（The Unified Theory of Acceptance and Use of Technology，簡稱UTAUT）便是其一，UTAUT 模型整合了八種主要的技術接受理論[20]，[21]研究了五所大學的學生對數位學習的接受度。UTAUT 模型後來被擴展到消費者環境中，強調科技使用者的享樂價值（內在動機），包括三個新的概念，如：享樂主義動機、價格價值和習慣[22]，UTAUT2 模型被用來確定計算機自我效能感和計算機遊戲性對電子學習系統的感知易用性是否有顯著影響， Baptista [23]等人研究了遊戲化對對行動銀行服務的影響，研究結果顯示遊戲化與使用手機銀行服務的意向之間存在直接和強烈的關係。因此對本研究而言，它是最適合我們發展與驗證理論的模型。

第三章 系統設計與方法

本章節中將會介紹 3.1 遊戲設計原則、3.2 遊戲總體設計、3.3 遊戲機制以及 3.4 關卡主題與學生數據監控。

3.1 遊戲設計原則

本研究採用了[24]所提出的設計原則，將 flow experience 與教育設計結合，試圖創造出具有高度沈浸感的遊戲體驗，進而使學習過程的效益更高，因此本研究提出之系統以以下步驟進行設計:

1. 分析學習者: 我們的學習者以 computer science 科系的大二學生為主，具有基本的編程能力，但大多還不具備有版本控制工具的使用能力，也大多沒有長期維護專案或大型專案的開發經驗，因此需要讓他們能夠具備基本的版本控制知識與版本控制工具使用能力。

2. 設立明確的教育目標，選擇適合的遊戲內容: 本研究的教育目標設立為令學生具備 Git 指令的使用能力，這些指令包含建立 git repository, commit, push 等等，並且能夠理解使用版本控制工具的原因與 Git 的工作流程。因此本研究選擇採用模擬開發情境的方式，讓學生在遊戲中操作模擬的 Git 工具，完成我們所設立的任務目標。

3. 根據教學目標和遊戲內容設計教學方法: 本研究將 Git 的指令設計成數道關卡，要求學生在模擬的情境中完成我們所設立的任務目標，學生在學習過程可以獲得遊戲內的機制作為學習輔助，並且由於我們所設計的競爭性的遊戲機制與遊戲獎勵，可以激勵學生在學習過程中的參與程度，增加學生的學習動機。

4. 將教學視為主要目標，遊戲作為輔助工具: 本研究將 GEG 作為課程的一部分，在後續的課堂中仍以傳統的講座方式教授 Git 的概念與操作方式，對學習遇到困難的學生可以以遊戲進行自主學習，選擇適合他們的學習方式。

5. 良好運用電腦遊戲的特徵: 我們的遊戲屬於模擬遊戲，同時加入了一些競爭遊戲的特徵，藉由電腦遊戲的特性，帶來重複嘗試的機會與動力，遊戲的提示說明、指令卡牌作為輔助學生理解的工具。

6. 將學生置於教學過程的中心，幫助他們享受學習: 學生在學習過程中基於遊戲的積分機制、成就機制等等，主動嘗試各種可能，增加主動參與學習的動機，使學生能夠學習到更多關於 Git 的知識與使用方法。

7. 定期評估學生的學習情況，不斷改進教學: 學生的 activity 被紀錄並發送至伺服器後台，從關卡通過數量、學生通關花費的指令數、學生通關花費的時間等等資訊可以判斷學生的學習狀況，並即時調整可能過於困難的關卡，或增加更多關卡來幫助學生釐清混淆不清的概念。除此之外，也進行測驗與問卷調查來了解學生的學習狀況。

3.1.1 布魯姆分類原則

在設計原則的第二步設計教育目標當中，本研究根據Bloom's taxonomy[25]，將學生在知識、理解和應用上等等獲得的 Git 能力分為以下幾點:

1. Remember: 學生可以記得跟 Git 關聯的概念名詞與部份指令，比如: repository, commit, branch, merge 等等。

2. Understand: 學生可以在操作、提示與視覺化的 Git 工作流程概念中理解 local 與 remote repository 的差別、Git 與 Git server 的差別、分散式版本控制系統的作用、branch 的作用與功能、merge 的作用及 conflict 是什麼。

3. Apply: 學生可以在反覆練習中獲得操作各種指令或解決跟 Git 有關問題的能力，比如:clone, commit, push, pull, create branch, merge, conflict solve。

4. Analyze: 藉由循序漸進的關卡，學生必須在了解前面關卡的基礎上才得以通過後續的關卡，這能使學生了解每一個指令或 Git 概念的功能與區別，最終學會整體的版本控制工具及 Git 的工作流程與概念。

3.2 遊戲總體設計

GEG 模擬了開發情境以及 Git Command Line（Git CLI），玩家在遊戲中必須操作遊戲模擬的 Git CLI，遊戲共有十三道關卡，除了第一關與第二關屬於遊戲的介紹章節以外，其餘每道關卡各代表了一個或數個 Git 的指令或概念，學生必須在每一個關卡中藉由遊戲內的提示來學習並嘗試操作，最終達成關卡內的任務目標並通過關卡。

遊戲內的關卡難度是循序漸進的，最初的關卡只需要完全跟隨提示輸入指令即可完成，後續的關卡則需要學會應用前面關卡的概念才能得到正確答案，當玩家通過關卡後便能獲得積分，達成某些特殊條件後也能獲得徽章與積分，這些結果將呈現在排行榜上，用以激勵學生。

此外遊戲被分為基礎關卡與進階關卡，基礎關卡以教學指令為主，目標為讓使用者認識、理解、學會使用某項指令，進階關卡則著重於設計實際的開發情境，讓玩家參與真實的開發流程，學習版本控制技術在實際開發中的應用。

3.2.1 系統架構

遊戲的系統分為前景與後台，前景除了負責使用者與遊戲介面的互動，還會發送通知給遊戲的後台，後台會去與遊戲的伺服器進行請求，包含使用者註冊、登入服務，使用者的事件紀錄 以及相關的資料請求（學生的積分、成就狀態），系統的架構如圖 3.1 所示。

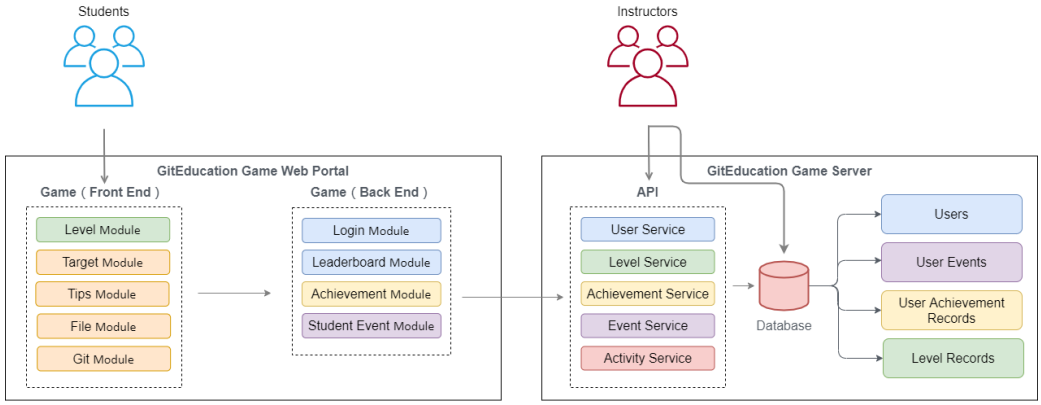


圖 3. 1 系統架構

老師方面可以透過 API 獲取資料了解學生的學習狀況，包含活動紀錄、關卡通過紀錄等等，也可以直接從資料庫查詢來獲得更詳細的資訊。

後端伺服器採用了 Spring Boot 框架，使用 RESTful API 作為與遊戲系統溝通的橋樑，並負責處理系統的資料邏輯，比如：檢查帳號是否存在以拒絕重複註冊、對已存在關卡紀錄的資料予以更新而非建立一筆新資料，資料庫則採用 MongoDB 以紀錄各種學生的事件。

3.2.2 遊戲流程

本研究的遊戲是基於 Web 的，學生必須進入遊戲的網址才可以開始進行遊戲。圖 3.2 顯示了遊戲的開始選單，我們要求學生必須以學號註冊才能開始進行遊戲，圖 3.3 則顯示了遊戲的章節選單，我們所要教授的 Git 概念與指令被包含在這些關卡當中，學生必須通過相應的關卡才能解鎖後續的關卡。

圖 3. 2 開始選單

圖 3. 3 關卡選單

在學生正式開始第一關之前，學生首先會進入第零關，這個關卡會向學生進行解釋遊戲如何進行。

圖 3.4 中顯示了的遊戲介面被分為六大區塊，下列的號碼與圖片中的識別號碼相對應：

1. 關卡的簡述以及關卡的指示開啟按鈕
2. 關卡的任務目標，完成的目標會以綠色顯示，否則顯示紅色
3. 檔案區塊，學生有時必須操作檔案進行修改及儲存
4. Git console 介面，學生必須在此輸入相應的 Git 指令
5. Git 的視覺化顯示區塊，包含遠端（上半部份）與本地（下半部份），右上角有關卡重啟按鈕，當學生使用不可逆操作時可以使用
6. 與關卡相關的 Git 指令提示卡片，將鼠標移動至卡片上方時會放大

圖 3. 4 遊戲的六大區塊

學生進入關卡後，關卡的指示會自動被開啟，如圖 3.5，閱覽至最後一頁時再按下一頁即會關閉，也可以手動立即關閉，而學生必須照著指示完成目標。

圖 3. 5 遊戲主畫面

圖 3. 6 關卡通過畫面

圖 3.6 顯示了當關卡完成後，除了基本的恭賀訊息外，也向學生顯示在關卡中耗費的時間與輸入指令的次數，這些紀錄都會被傳送至伺服器資料庫，除了可用於後續分析外，還作為遊戲內排行榜依據的一部分，同時章節選單會解鎖相應

的關卡，使其成為可自由進出的關卡，以利學生複習。

3.2.3 教學 CLI

J. Lawrance 等人提到在圖形界面（GUI）中學習 Git 被觀察到雖然可以避免學生不適應命令行界面，但容易使學生感到混淆，因此使用 CLI 仍是學生的首選[19]。本研究採用教學 CLI，相關原因有以下幾點：

1. 初學者直接使用 GUI 容易混淆對 Git 運作原理的理解，並可能誤將 GUI 設計的功能當作 Git 的直接功能

2. Git CLI 在所有環境與機器上都是相同的，使用 GUI 則可能因作業系統不同而無法使用

3. CLI 相對 GUI 更為完整，Git 的所有功能都被包含，GUI 則不一定

4. 當使用遇到困難時要尋求幫助更為容易，GUI 不一定存在完整且良好的文檔，而 CLI 則更容易在網路上獲得幫助

3.3 遊戲機制

遊戲元素作為本研究吸引學生主動學習 Git 的方式，在 GEG 中引入了以下幾項遊戲設計元素：點數、排行榜、徽章。

3.3.1 點數

作為最常見的遊戲元素之一[15]，Star[26]發現僅僅採用積分就能增加對任務表現的量化指標，當學生每通過一個關卡時，便會獲得分數，為了配合循序漸進的關卡難度，通過越後期的關卡，學生所獲得的分數也會更多，使學生可以檢視自己的學習進度，而分數作為排行榜的排序依據之一，玩家可以在起始選單中開啟總排行榜查看自己的分數。

3.3.2 排行榜

排行榜也是最常見的遊戲元素之一，同時也是 GEG 主要的遊戲性所在，GEG

中的排行榜分為兩類，第一種是關卡排行榜，它根據學生通關時的數據進行排名，花費時間較少的學生會在排行榜的前方，花費時間相同時則比較花費的指令行數，並且會列出學生的完成時間，如圖 3.7，顯示了關卡 4 的排行榜紀錄。

圖 3. 7 關卡四的排行榜

第二種則是總排行榜，如圖 3.8 所示，根據學生在整個遊戲關卡獲得的點數進行排名，同時列出學生獲得的成就數量。

圖 3. 8 總排行榜

考慮到排行榜可能對於較低名次學生造成反效果[27]，GEG 設置了一個開關用以控制是否顯示全部排名，放在排行榜介面的右下角，預設狀態下只會顯示部份排名（前十位），若學生自願開啟則可以查看全部的名次。

3.3.3 成就與獎章

獎章通常用於象徵玩家的功績，視覺化的成就本身就代表一種獎勵，為成就提供一個獨特的標誌[28]。GEG 中設置了十項成就，當學生完成特定的任務時便可以獲得，獲得成就的學生除了可以蒐集到獎章以外也能獲得一定量的點數，鼓勵學生藉由完成特定的操作以在排行榜的競爭中獲得更前面的排名。成就也被分為較基礎且易於獲得的成就與較難獲得的成就，比如：當學生通過了遊戲中的第一關時，可以獲得一個名為入門的成就，圖 3.9 顯示了學生獲得該成就時跳出提示的畫面。

圖 3.9 獲得成就

此外當學生在某個關卡排行榜中成為第一時也可以獲得一個成就，又或是當學生在三十秒內就通關卡時也能夠獲得一個成就，並且 GEG 還設置了一些趣味性的成就，比如學生將遊戲指示一頁都不閱覽直接關閉時可以獲得一個名為我不需要提示的成就；又或是當學生閱覽一張卡片超過二十五次時可以獲得一個名為卡排之王的成就，這些成就可以增進學生的參與度，同時鼓勵學生在遊戲過程中盡可能的嘗試，以及增加可重玩性。

圖 3.10 顯示了成就閱覽器，學生可以在成就閱覽器中查看自己目前解鎖的成就，也可以查看那些未被解鎖的成就，在成就閱覽畫面中那些成就的圖示被隱藏起來，但是仍可透過對成就的敘述來了解需要達成什麼目標才能獲得，藉以鼓勵學生嘗試完成這些目標。

圖 3. 10 成就閱覽畫面

3.4 關卡主題與數據監控

在 GEG 中，遊戲的第一關（遊戲介紹在第零關）介紹為什麼要使用版本控制軟體，其餘總共有十二道關卡直接教授 Git 指令與概念，分別介紹以下主題：

1. 建立 Git 倉儲庫（git init）
2. Commit 操作（git commit）
3. Push 操作（git push）
4. 創立分支（git branch）
5. 合併與刪除分支（git merge and git branch -D）
6. 同步與衝突（git pull and conflict solve）
7. 分支合併與衝突（git merge and conflict solve）
8. 暫存與釋放（git stash and git pop）
9. 另一種整合方式（git rebase）
10. 版本發布（git tag）
11. 維護專案並開發新功能（進階關卡）

學生在 GEG 中的活動會被紀錄下來發送至資料庫，教師可以根據狀況即時了解學生的學習狀況，這些活動包含：

1. 學生開始進行關卡
2. 學生完成關卡中的某項目標
3. 學生完成關卡（包含花費時間與行數）
4. 學生開啟遊戲中的提示
5. 學生關閉遊戲中的提示
6. 學生在 CLI 中輸入的指令
7. 學生目前的獎章數量
8. 每道關卡通過的學生人數

教師可以查看每道關卡通過的人數來判斷學生是否在某些環節遇到困難，比如在圖 3.11 當中，查閱了關卡八在實驗日時的通過紀錄，我們觀察到只有大約 5 成的學生有通過關卡，關卡九也同樣只有約一半的學生有通過，教師可以由此得知學生對於分支合併、衝突解決等等並不能即時充分理解與掌握。

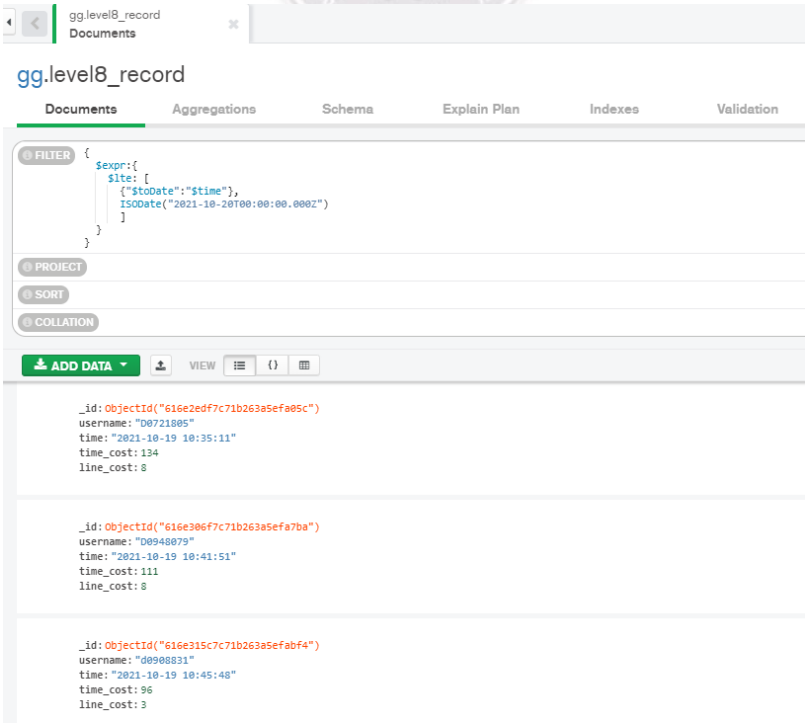


圖 3. 11 關卡八學生通關資料

第四章 系統展示

此章節展示本研究所提出之系統的各個關卡，並解說我們在各個關卡中如何教學 Git 的概念，學生應該獲取什麼樣的知識。

4.1 遊戲基礎關卡

如第三章所述，遊戲關卡被區分為基礎關卡與進階關卡，在基礎關卡中我們提出情境、遭遇問題、相應的指令解方、指令使用方法，循序漸進的引導使用者學會指令。

當學生成功進入遊戲後會先進入第零關，本關卡屬於遊戲的教學關卡，系統會以打字機方式呈現對話，並講解遊戲的機制、進行方式、不同區塊的功能，如圖 4.1。

圖 4. 1 Git Education Game 教學關卡

在正式進入第一關後，我們會先說明備份檔案的重要性，提出所謂「版本」的概念，並請玩家使用較為土法煉鋼的方式複製檔案以實現備份，如圖 4.2。

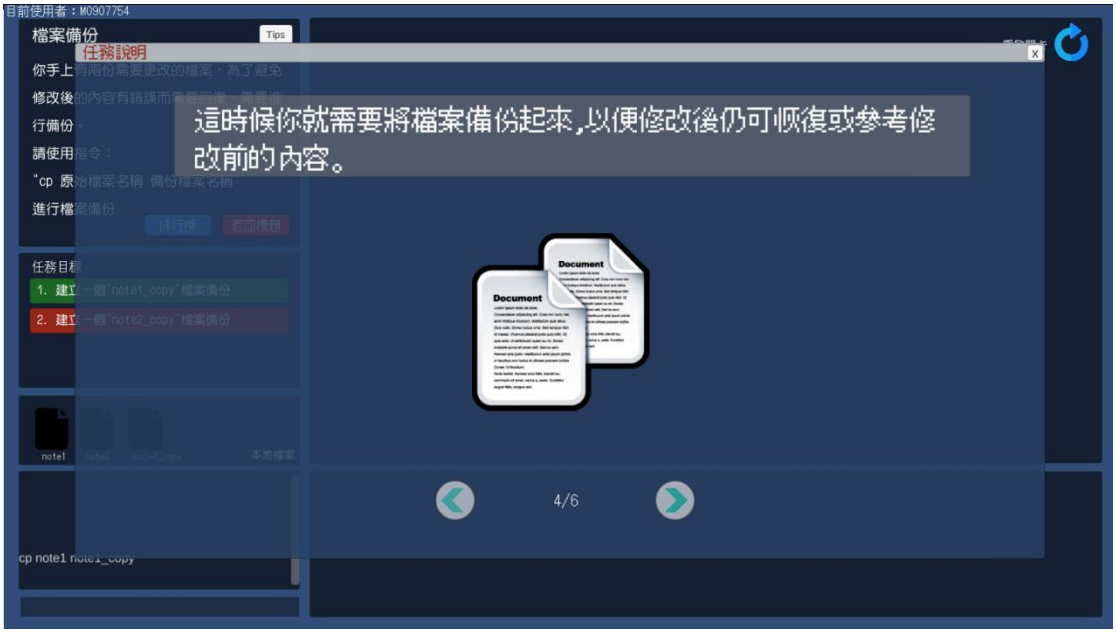


圖 4. 2 Git Education Game 關卡一

當進入關卡二之後，我們開始說明這種備份方式帶來的管理成本，並引入 Git 這套版本控制軟體，說明此套軟體可以使程式碼的版本管理變得容易，並有助於多人協作，並請玩家操作第一個指令，建立 Git 儲存庫，同時加入卡片提示系統使指令能被更完整的理解，如圖 4.3。

圖 4. 3 Git Education Game 關卡二

當進入下一關卡後，學生可以正式開始學習 Git 的重要指令，為求避免需要使用 Vim 編輯器帶來的額外學習成本，這裡讓學生學習直接將訊息放在參數的 commit 指令形式，如圖 4.4。

圖 4. 4 Git Education Game 關卡三

完成了在本地的版本儲存操作後，我們開始引入遠端的概念，首先我們講解了在伺服器端儲存的好處與需求，如圖 4.5，為了導入這項概念，我們將 Git 的視覺化區塊一分為二，分為本地端與遠端。

圖 4. 5 Git Education Game 關卡四的解說

同時我們簡化了 git remote add、git push 的指令，玩家僅需輸入「myserver」即可完成設定遠端，push 也不需要額外輸入參數，以避免過多的參數讓玩家混亂指令的使用方法與意義，如圖 4.6。

圖 4. 6 Git Education Game 關卡四的目標達成方式

前面關卡教學的為從頭建立專案的情況，然而許多情況是需要維護既有專案，因此我們接著教學這方面的指令，同時也如前一關對指令進行了參數簡化，如圖 4.7。

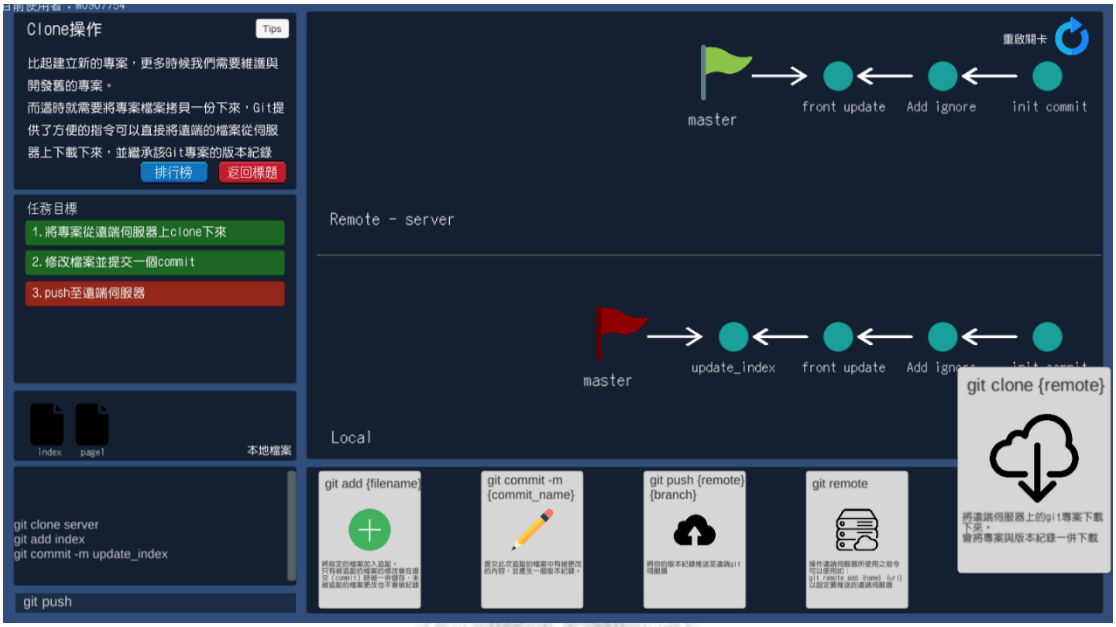


圖 4. 7 Git Education Game 關卡五

通過基本操作的章節後，為了讓玩家了解協同開發如何進行，我們開始引入分支的概念，講述多人開發面臨的情況與沒有分支功能時的作法，及其帶來的時間成本、管理困難，接著說明分支功能帶來的好處與其作用，並要求玩家使用相關的指令建立分支、切換分支提交版本與切回原分支，以令其對指令使用有基本的了解，詳細如圖 4.8。

圖 4. 8 Git Education Game 關卡六

多人協作中必然會需要整合所開發的功能，在此關卡中我們接續前一關的狀況，並提出 git merge 指令可以進行合併，在此關中我們選擇讓玩家進行未有衝突的合併，以求循序漸進的吸收、理解知識，避免對玩家造成過大的負荷，如圖 4.9。

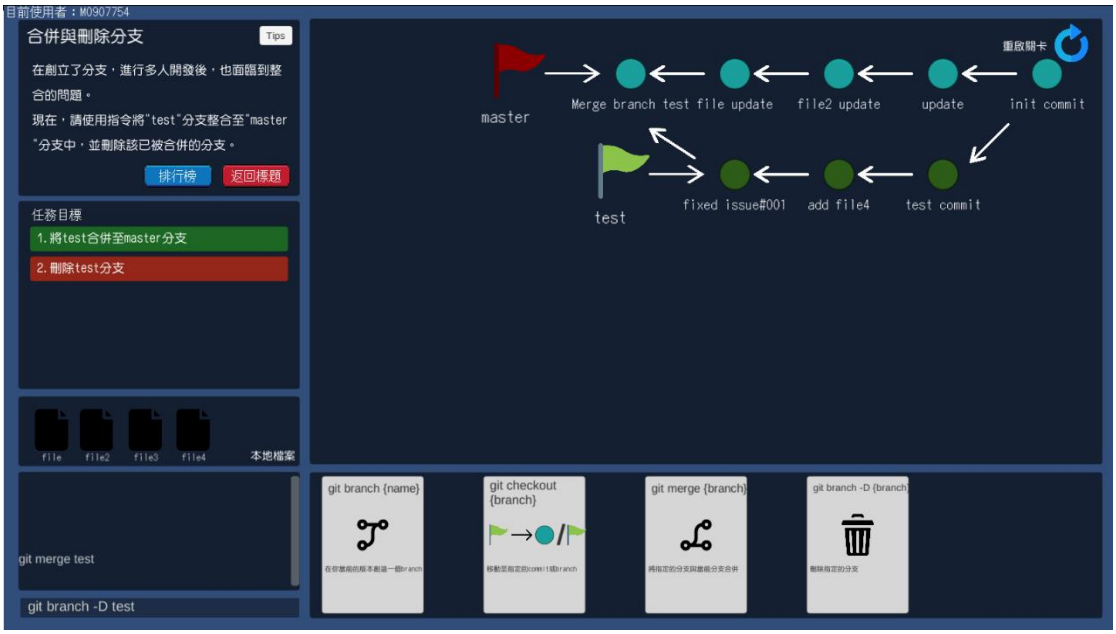


圖 4. 9 Git Education Game 關卡七

分支的章節結束後，我們開始介紹關於「衝突」的知識，不同版本中的檔案衝突是協同開發中必然發生的事件，也是大多數學習者較為難學會的地方，為了降低難度，我們的關卡首先設計為同一分支上的合併衝突，玩家在本關卡中需要從遠端以 pull 指令將伺服器上的 commit 下載下來，並處理所發生的衝突，如圖 4.10。

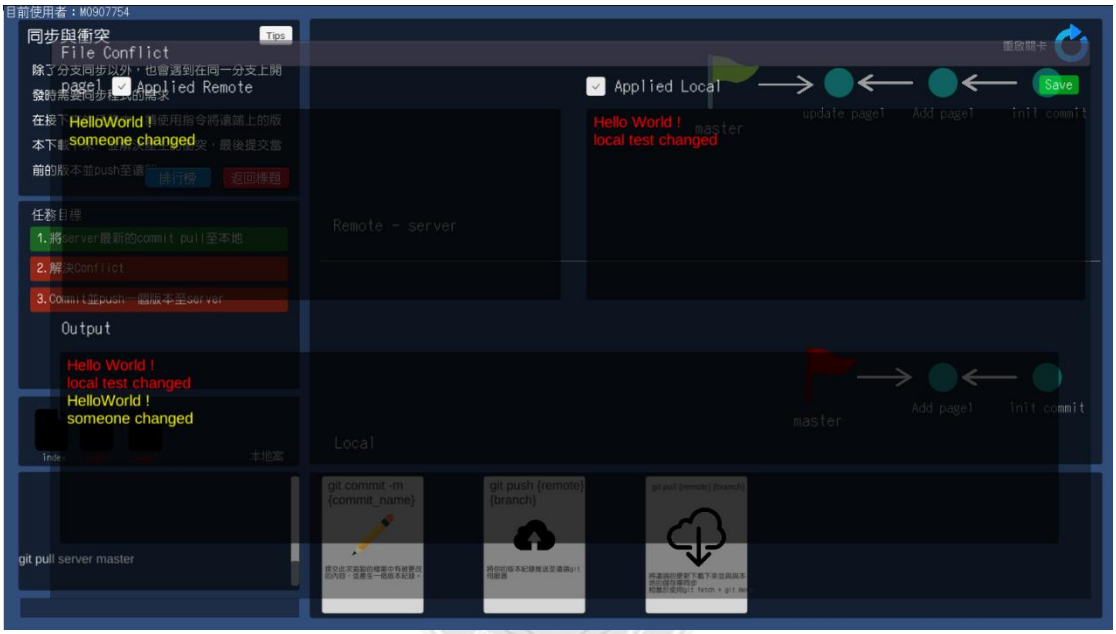


圖 4. 10 Git Education Game 關卡八

為了避免關卡難度的曲線過於陡峭，我們在本關卡選擇讓玩家以 GUI 的形式學習解決衝突，玩家可以在界面上直接看兩份檔案有何不同，並選擇要採用哪個版本。

當通過同一分支的衝突處理關卡後，我們安排了不同分支合併的衝突解決，主要的流程與前一關大同小異，解說以分支開發功能整合時的檔案衝突問題，要求玩家合併關卡中的分支、解決衝突，並 push 至伺服器，如圖 4.11。

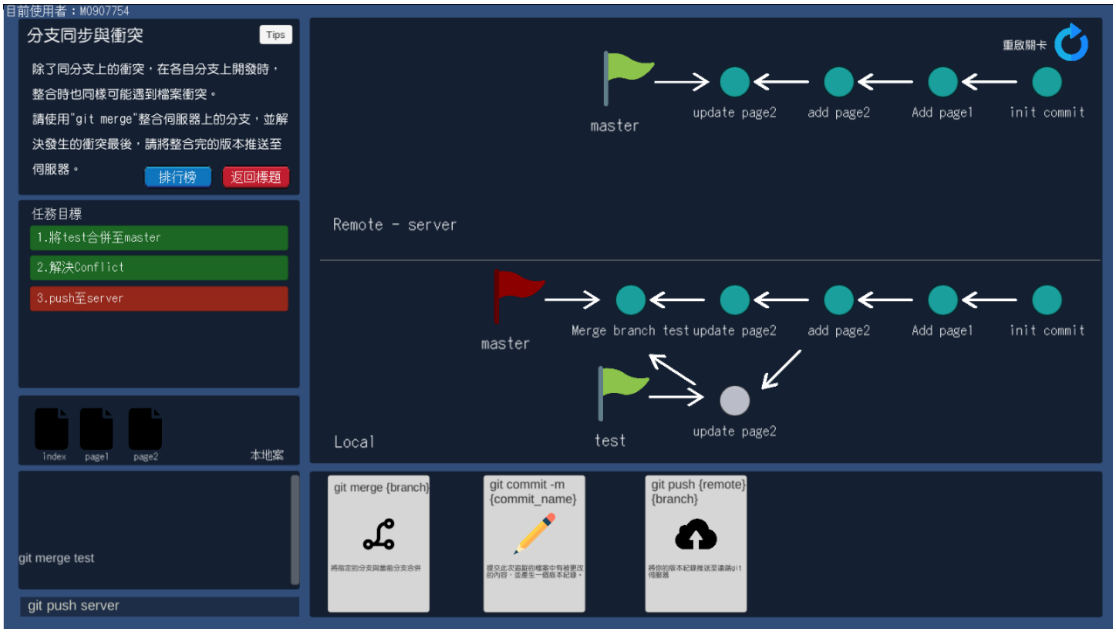


圖 4. 11 Git Education Game 關卡九

學習 Git 大多數會用到的指令後，我們開始教學比較非核心功能的指令，這些指令雖然不見得是開發上必要的，但能夠使開發時更靈活的進行版本控制，為了體現這點，我們設計的開發情境會試圖令使用者感到採用這些方法能帶來更佳的工作效率。

圖 4.12 當中我們教學 stash 與 pop 的指令，在實際開發中需要停下手邊工作的情況並不稀少，此時若學會運用指令則能免去手動複製的麻煩，並保留工作狀態，因此本關的情境中需要切換至其他分支進行 commit 再切回原始工作狀態。

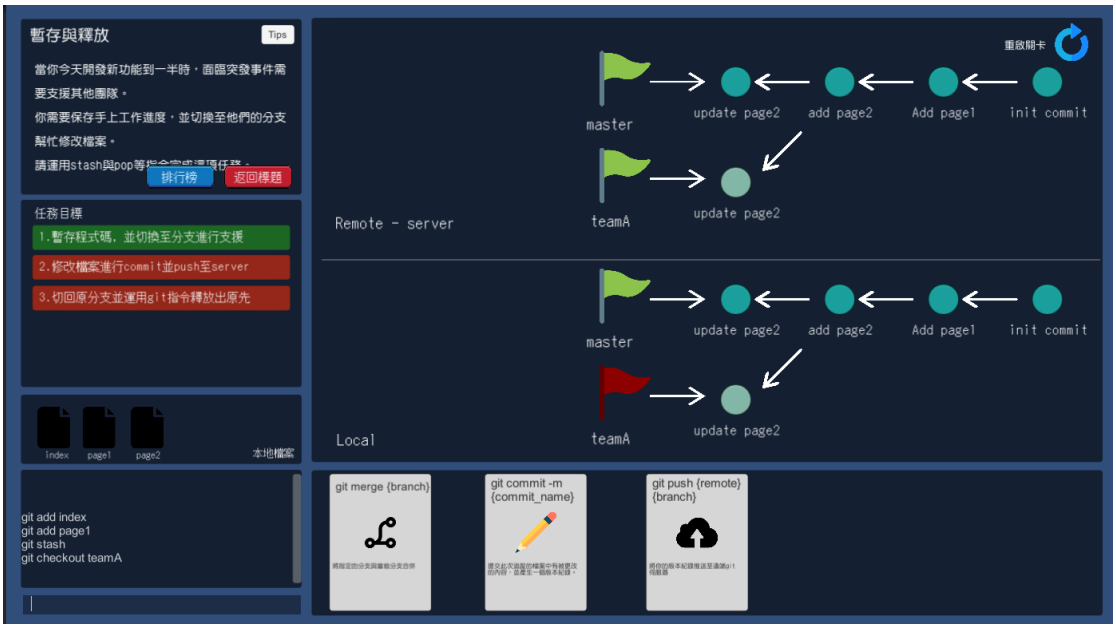


圖 4. 12 Git Education Game 關卡十

整合分支中，除了 merge 指令，我們還有 rebase 指令可以進行合併，我們講解兩種整合方式的差別並以視覺化效果突顯造成的結果差異，教學玩家另一種程式碼整合方式。

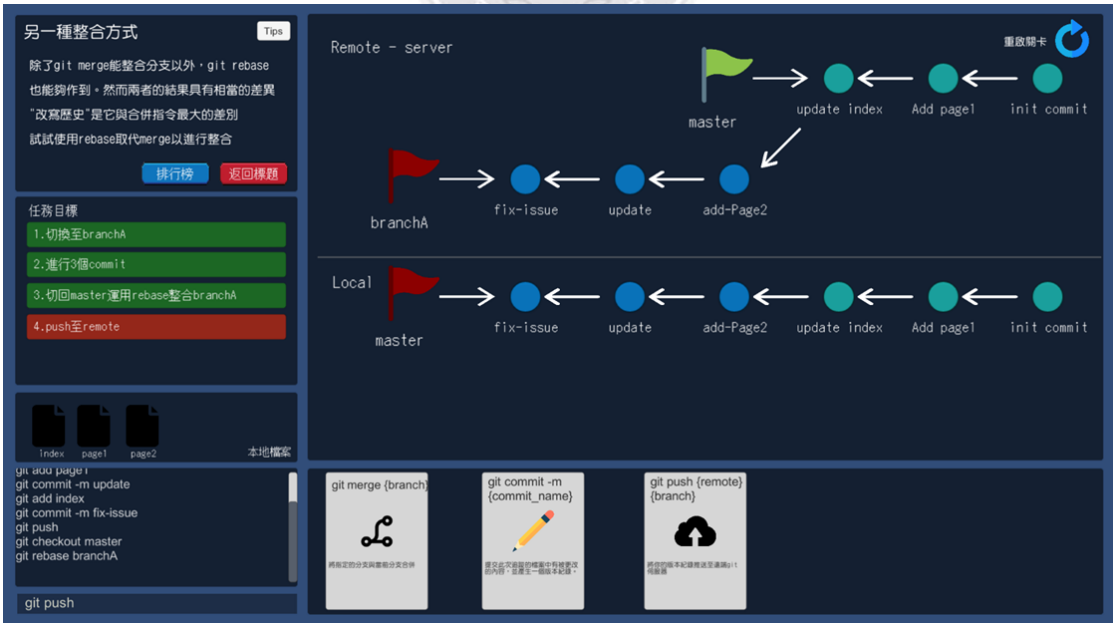


圖 4. 13 Git Education Game 關卡十一

版本發布也是 Git 中一項重要的功能，雖然並不困難，但我們仍為使用者引進關卡中，以令其了解更完整的 Git 功能。

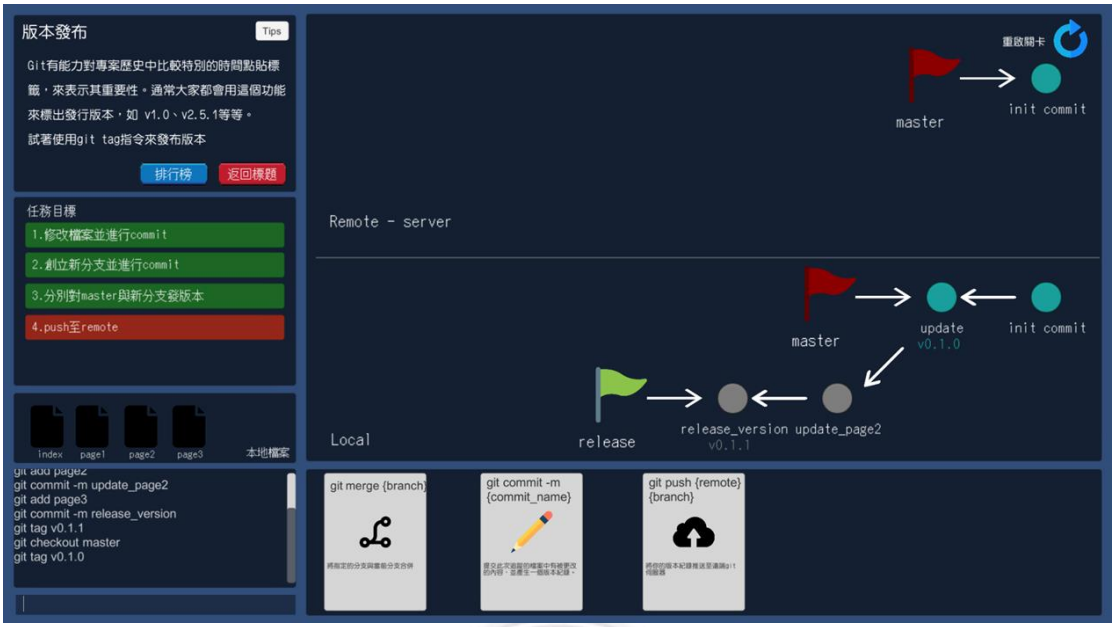


圖 4. 14 Git Education Game 關卡十二

4.2 遊戲進階關卡

進階關卡是為強化學生的學習成果所設計，單次的指令教學關卡學習的成效有所極限，必須透過學會靈活應用才能真正學會。在進階關卡中我們同樣提出情境、遭遇問題，使用者需要活用基礎關卡中學會的指令以解決當前開發過程遇到的困難。

我們為此進階關卡設計了一個腳本，玩家在團隊中扮演開發中專案的開發者，要在修復 bug 的同時與他人和合作開發新功能，如圖 4.15。

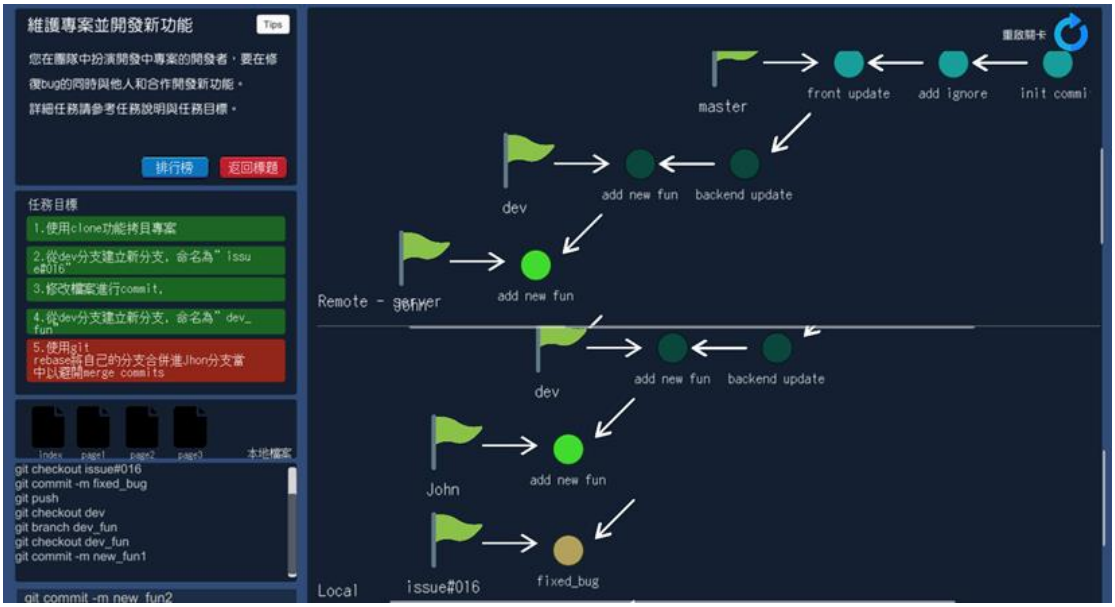


圖 4. 15 Git Education Game 進階關卡

玩家在本關卡中必須完成多項目標，包含：

1. 使用 clone 功能拷貝專案
2. 從 dev 分支建立新分支，命名為 issue#xxx
3. 修改檔案進行 commit, push，並合併至 dev
4. 從 dev 分支建立新分支，命名為 dev_xxx
5. 扮演互動成員的電腦也進行了 bug 修正與功能開發，玩家需使用 git rebase 將自己的分支合併進對方的分支當中以避開 merge commits

當玩家能夠順利完成目標通過關卡時，便已經具備應用這些指令的能力，在實務操作上或許仍無法保證不會遭遇困難，但對於指令的應用方法、時機、概念已經具備相當的能力。

第五章 系統實驗

5.1 實驗環境

本研究在台灣逢甲大學 110 學年度上學期的物件導向設計課程中實施，這堂課程分為兩個班級，為了設計與驗證這項系統，隨機選擇一組作為實驗組，另一組作為控制組，分別有 54 位學生與 59 位學生，參與課程的學生多為大二的計算機科學學生，此外這堂物件導向設計課程使用的學習平台需要使用 Git 上傳作業程式碼（包含但不限於 CLI）。

5.2 實驗流程

實驗的流程如圖 5.1，在實驗開始之前，實驗組與控制組分別有一個不涉及實作的 Git 教學課程及選擇題前置測驗，作為判斷學生是否具有相同的基準，在前置課程的下一週，分別讓實驗組及控制組進行 Git 的課程，範圍包含從建立 Git Repository 到解決衝突，課程時間為兩小時，實驗組學生則為一小時的遊戲時間與一小時的課程教學（教學範圍與控制組相同），為了使學生有時間得以吸收知識，後置測驗安排在數周之後。

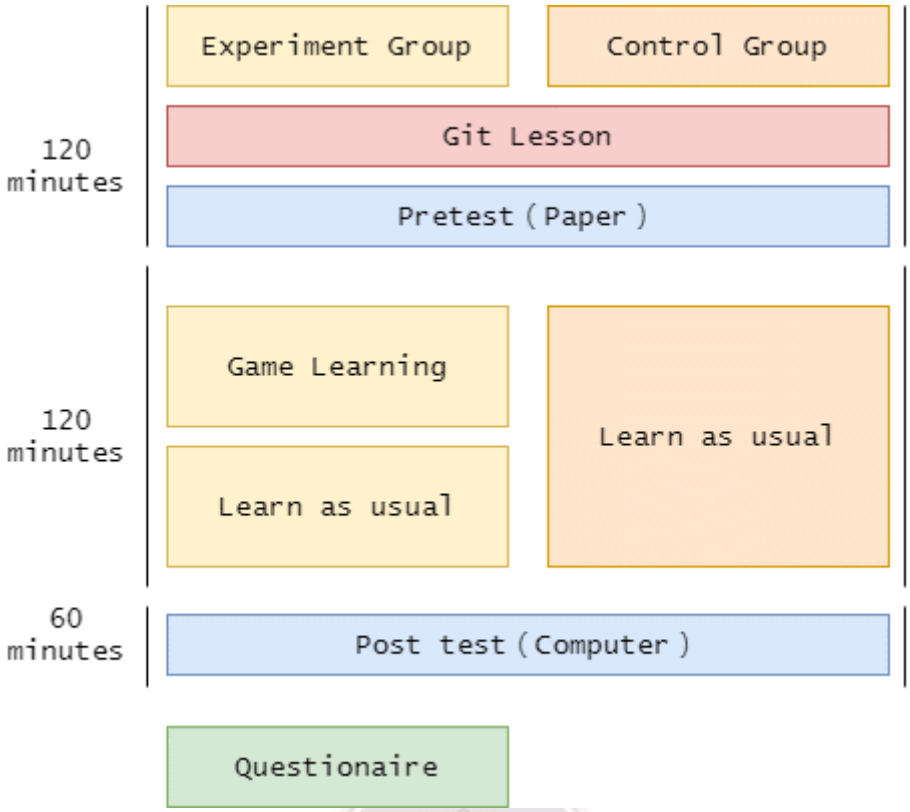


圖 5.1 實驗流程圖

在實驗組的課程中，首先教導了學生如何操作 GEG 這款遊戲，並當場示範如何通過第一個關卡，接著讓學生自主學習，遊戲的主要教學範圍為第零至第九關，學生操作的遊戲版本並未包含十以後的關卡，第二小時則教導與控制組相同的課程內容，教導的範圍與遊戲關卡所教授的範圍一致，如圖 5.2，學生在課堂上進行遊戲並學習 Git 的概念與使用方式。

圖 5. 2 學生在課堂上以遊戲進行學習

後置測驗為上機考的形式，學生使用預先架設好了私人 Gitlab 進行考試，在進行考試前說明了如何使用 Gitlab，考試時限為一小時，考題的範圍包含了先前所教授的內容如克隆、提交、推、新增分支、解決衝突等等，總共三題，如圖 5.3 與圖 5.4。

README.md

Git期末測驗

- 在這次考試中，你必須從助教的專案中使用fork功能複製一份專案並進行操作
- 題目要求如下

Fork與clone

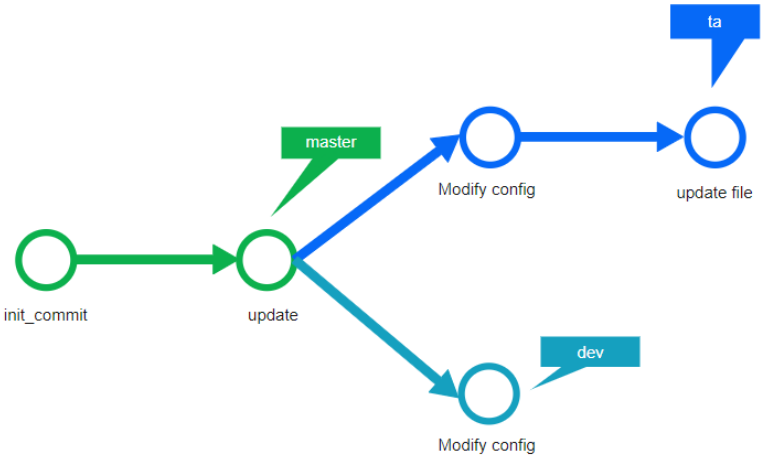
- 首先你必須使用fork功能複製此專案至自己的帳號中
- 進入fork完成的專案，並從gitlab上clone至本地電腦

建立個人分支

- 當專案被拷貝至本地後，在本地創立一個分支並命名為"dev"
- 切換至該分支，並修改專案中的"config.txt"檔案
- 將檔案內容中的"ans="後的部份隨意填上（填上回覆完問卷後的一串字串有加分）
- 修改完成後請進行commit並推送至gitlab

目前分支狀況

- 當你做完以上操作後，目前的專案版本狀態應該是這個樣子的（視覺化後的樣子）：



- 而我們希望在下一步時，變成這個樣子：

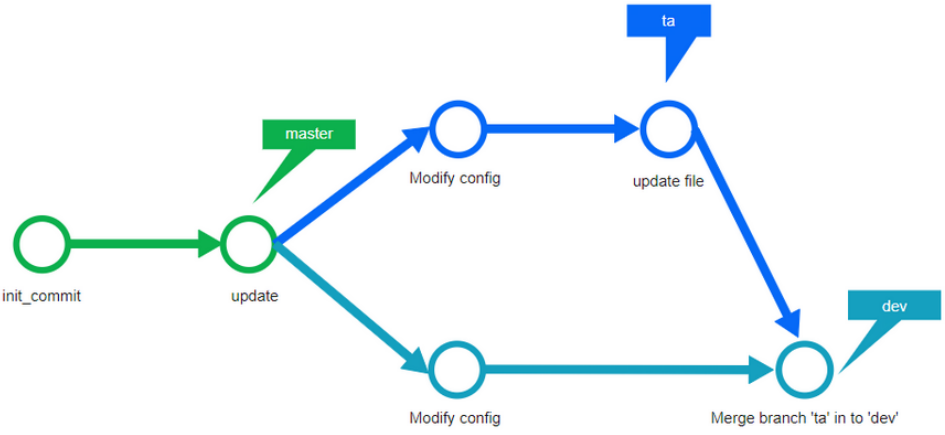


圖 5. 3 後置測驗上機考題上半部份

圖 5. 4 後置測驗上機考題下半部份

測驗完成後，請學生填寫了關於遊戲體驗與對 Git 態度方面的問卷，共有 25 個項目，分為九個類別：

1. 績效預期 Performance Expectancy（PE）

2. 期望確認 Effort Expectancy（EE）
3. 自我效能 Self-Efficacy（SE）
4. 享樂動機 Hedonic Motivation（HM）
5. 遊戲動機 Game Motivation（GM）
6. 遊戲有用性 Game Usefulness（GU）
7. 態度 Attitude（AT）
8. 行為意向 Behavioral Intention（BI）
9. 使用者行為 User Behavior（UB）

此外有兩個開放式問答。這個量表調查了學生對於系統的遊戲體驗，包含遊戲對於學習 Git 的有用性以及是否帶來學習動機，本研究使用了五點李克特量表（five-point Likert scale）來測量我們設計的問卷，範圍從 1（非常不同意）到 5（非常同意）。

5.3 研究分析方法

本研究採用並擴展延伸整合科技接受模型，並以結構方程模型 PLS-SEM 進行分析，PLS-SEM 路徑建模運用恰當的情況，被認為在許多經驗數據為特徵的背景下估計因果模型[29]，並且它對樣本數量的要求較低[5]，對小樣本數的容忍是本研究採取其的主因，PLS-SEM 以測量模型與結構模型檢測因變量與自變量[29]，圖 5.5 為我們所提出的研究模型。

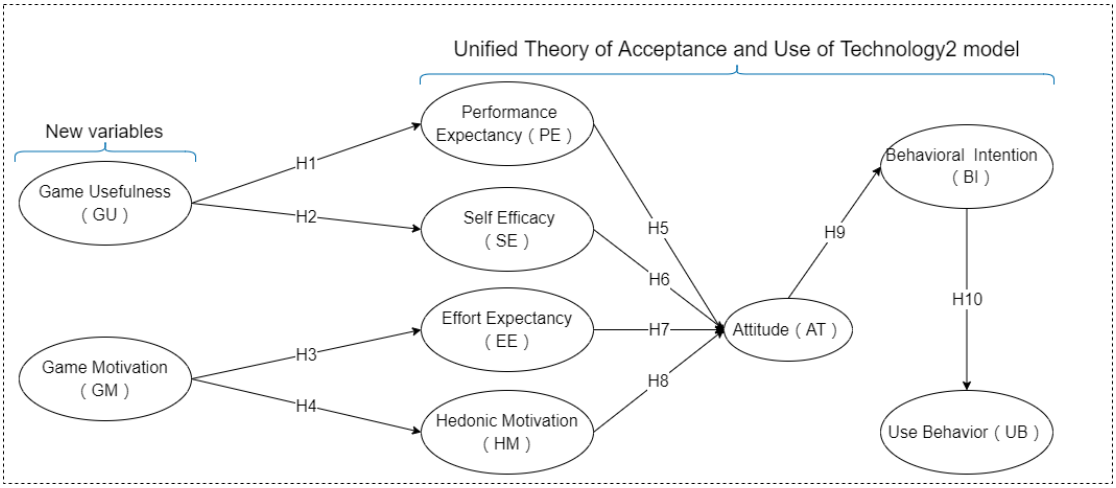


圖 5. 5 研究模型

根據研究模型各構面之間的關係，提出以下假設：

- **Hypothesis 1 (H1):** 遊戲化系統內的機制會影響學生使用遊戲學習 Git 的績效預期
- **Hypothesis 2 (H2):** 遊戲化系統內的機制會影響學生使用遊戲學習 Git 的自我效能感
- **Hypothesis 3 (H3):** 遊戲化系統內的機制會影響學生對使用遊戲學習 Git 的努力期望值
- **Hypothesis 4 (H4):** 遊戲化系統內的機制會影響學生對使用遊戲學習 Git 的享樂主義動機
- **Hypothesis 5 (H5):** 使用遊戲學習 Git 的績效預期會影響學生對 Git 的態度
- **Hypothesis 6 (H6):** 使用遊戲學習 Git 的自我效能感會影響學生對 Git 的態度
- **Hypothesis 7 (H7):** 使用遊戲學習 Git 的努力期望值會影響學生對 Git 的態度
- **Hypothesis 8 (H8):** 使用遊戲學習 Git 的享樂主義動機會影響學生對 Git 的態度
- **Hypothesis 9 (H9):** 學生對 Git 的態度會影響其對 Git 的行為意向
- **Hypothesis 10 (H10):** 學生對 Git 的行為意向會影響其實際使用 Git 的行為

我們根據模型中的各構面來設計問卷，如表 5.1。

表 5. 1 問卷內容

一、績效預期 Performance Expectancy（PE）
PE-1.使用 Git Education Game 使我學會的速度增快
PE-2.使用 Git Education Game 增進了我學習的成果
二、期望確認 Expectation-confirmation（EE）
EE-1.Git Education Game 使用起來不會太耗費心力
EE-2.Git Education Game 的功能及介面是很清晰易懂的
三、自我效能 Self-Efficacy（SE）
SE-1.當使用 Git 出現錯誤時，如果周圍沒有人可以教我，我能藉由遊玩以理解與解決發生的問題
SE-2.當學習 Git 出現困難時，如果周圍沒有人可以教我，我能藉由遊玩來釐清 Git 的概念或操作方法
四、享樂主義動機 Hedonic Motivation（HM）
HM-1.我認為透過 Git Education Game 來學習 Git 是有趣的
HM-2.我認為透過 Git Education Game 來學習 Git 能帶給我更多動力
五、遊戲有用性 Game Usefulness（GU）
GU-1.使用 Git Education Game 學習 Git 時，遊戲內的提示或指導能幫助我理解關卡內所要教學的 Git 概念
GU-2.使用 Git Education Game 學習 Git 時，我能透過完成關卡理解各項 Git 指令的使用方法
GU-3.使用 Git Education Game 學習 Git 時，遊戲內的視覺化效果及機制能幫助我理解 Git 的工作概念或流程
六、遊戲動機 Game Motivation（GM）
GM-1.使用 Git Education Game 學習時，遊戲內的積分與排行榜系統使我更有動力參與學習
GM-2.使用 Git Education Game 學習時，遊戲內的成就系統使我更有動力參與學習
GM-3.遊戲內的提示與指導比起純授課更使我有動力理解 Git 概念
七、態度 Attitude（AT）
AT-1.我認為使用版本控制工具來管理程式碼是個好主意
AT-2.我認為學習使用版本控制工具是個好主意
AT-3.我認同開發程式時應使用版本控制工具
AT-4.我對使用版本控制工具持積極態度
八、行為意向 Behavior Intention（BI）
BI-1.我打算在以後的專案採用版本控制工具
BI-2.我打算經常使用版本控制工具

BI-3.我打算成為版本控制工具的忠實使用者
九、實際行為 User Behavior（UB）
UB-1.我在所有的軟體專案中都經常使用版本控制工具
UB-2.我推薦我的同學像我一樣使用版本控制工具
十、開放式問題 Open ended Question（OEQ）
OEQ-1.你認為 Git Education Game 在教學或遊戲方面有什麼吸引你的優點？或是為你帶來了什麼學習上的好處？
OEQ-2.你認為 Git Education Game 在教學或遊戲方面有什麼缺點？或是在教學上有什麼缺失的地方？

5.4 實驗前置測驗結果與分析

關於前置測驗的結果如表 5.2，本研究使用獨立樣本 t 檢定來比較控制組與實驗組的平均數是否有所差異，控制組抽樣 54 個，平均數為 72.593；實驗組抽樣 59 個，平均數為 76.271，在變異數同質性檢定中，檢定統計量 f 值為 1.1761，機率值 p 值為 1.4487，未達α=0.05 的顯著水準，表示兩組樣本的變異數並無顯著差異，因此獨立樣本 t 檢定採用變異數相同的檢定統計量 t 值計算方式，在獨立樣本 t 檢定中，檢定統計量 t 值為-1.1918，機率值 p 值為 0.2359，未達α=0.05 的顯著水準，因此無法拒絕虛無假設，表示控制組與實驗組兩組的平均數並沒有顯著差異。

表 5. 2 實驗組與控制組的前置測驗結果

Group	Count	Mean	Median	Minimum	Maximum	Std. dev.
Control group	59	72.5926	70	40	100	15.6838
Experiment group	54	76.2712	80	30	100	17.0090

5.5 實驗後置測驗結果與分析

在後置測驗部份，呼應研究所假設的問題，本研究將考試題目的要求劃分為三部份：

1. 能夠 fork 專案並進行 clone、commit 及 push
2. 能夠新增 branch，並切換分支進行提交
3. 能夠進行合併，並解決衝突

後置測驗的結果中，實驗組中有 73%的學生能夠通過要求一，而控制組中有 64%的學生能夠通過要求一；實驗組中有 64%的學生能夠通過要求二，而控制組中有 50%的學生能夠通過要求二；實驗組中有 43%的學生能夠通過要求三，而控制組中有 33%的學生能夠通過要求三，圖 5.6 顯示了兩個組別在三項要求的通過率。

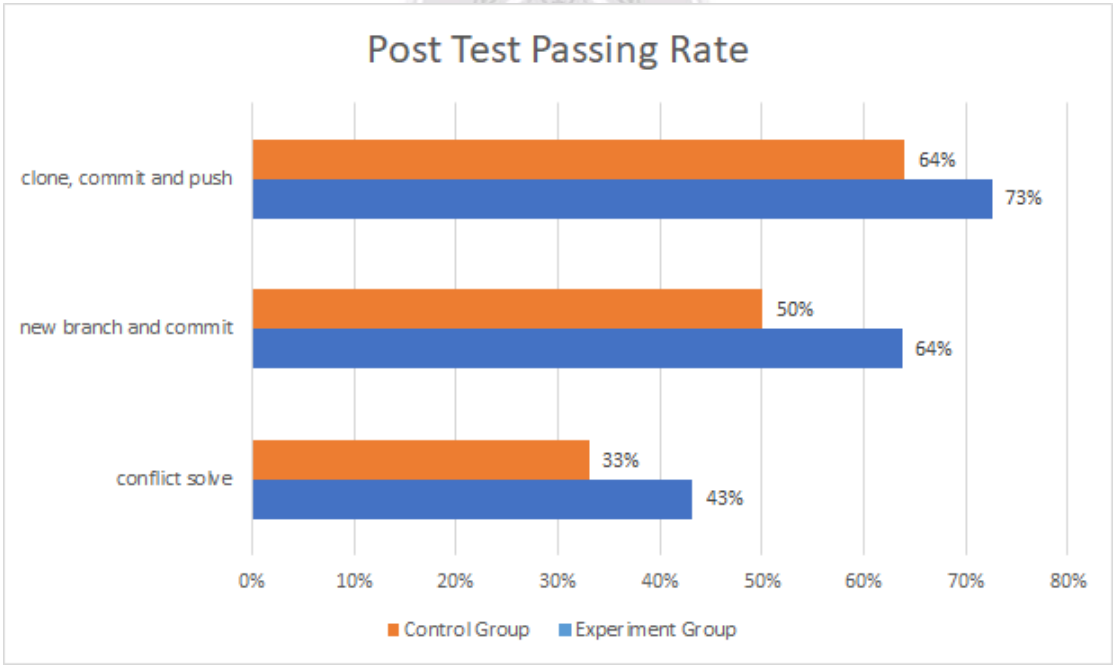


圖 5. 6 後置測驗通過率

5.6 問卷調查結果

關於問卷調查的結果如表 5.3，我們共回收了 39 份有效問卷，這些問卷來自實驗組中願意填寫的學生，其中大約有 22 位學生在開放性問題中填寫了對於 GEG 的意見，學生對大多數題目的同意程度偏高，然而我們有觀察到在 UB1 中的數值明顯較低。

表 5.3 問卷內容與調查結果

#	Questions	Mean	Std
PE	**一、績效預期 Performance Expectancy（PE）**	**X**	**X**
PE1	PE-1.使用 Git Education Game 使我學會的速度增快	4.1026	0.8824
PE2	PE-2.使用 Git Education Game 增進了我學習的成果	4.0769	0.8701
EE	**二、期望確認 Expectation-confirmation（EE）**	**X**	**X**
EE1	EE-1.Git Education Game 使用起來不會太耗費心力	3.8974	1.0462
EE2	EE-2.Git Education Game 的功能及介面是很清晰易懂的	3.8974	0.9678
SE	**三、自我效能 Self-Efficacy（SE）**	**X**	**X**
SE1	SE-1.當使用 Git 出現錯誤時，如果周圍沒有人可以教我，我能藉由遊玩以理解與解決發生的問題	3.8205	0.9966
SE2	SE-2.當學習 Git 出現困難時，如果周圍沒有人可以教我，我能藉由遊玩來釐清 Git 的概念或操作方法	3.7949	1.0047
HM	**四、享樂主義動機 Hedonic Motivation（HM）**	**X**	**X**
HM1	HM-1.我認為透過 Git Education Game 來學習 Git 是有趣的	4.1282	0.7320
HM2	HM-2.我認為透過 Git Education Game 來學習 Git 能帶給我更多動力	4.0513	0.7930
GM	**五、遊戲有用性 Game Usefulness（GU）**	**X**	**X**
GM1	GU-1.使用 Git Education Game 學習 Git 時，遊戲內的提示或指導能幫助我理解關卡內所要教學的 Git 概念	3.8974	0.9946
GM2	GU-2.使用 Git Education Game 學習 Git 時，我能透過完成關卡理解各項 Git 指令的使用方法	3.9487	0.9986
GM3	GU-3.使用 Git Education Game 學習 Git 時，遊戲內的視覺化效果及機制能幫助我理解 Git 的工作概念或流程	4.1538	0.8441

GU	六、遊戲動機 Game Motivation（GM）	X	X
GU1	GM-1.使用 Git Education Game 學習時，遊戲內的積分與排行榜系統使我更有動力參與學習	4.1795	0.7564
GU2	GM-2.使用 Git Education Game 學習時，遊戲內的成就系統使我更有動力參與學習	4.1538	0.7793
GU3	GM-3.遊戲內的提示與指導比起純授課更使我有動力理解 Git 概念	4.0256	0.9315
AT	七、態度 Attitude（AT）	X	X
AT1	AT-1.我認為使用版本控制工具來管理程式碼是個好主意	4.3077	0.7662
AT2	AT-2.我認為學習使用版本控制工具是個好主意	4.4103	0.7511
AT3	AT-3.我認同開發程式時應使用版本控制工具	4.3590	0.7776
AT4	AT-4.我對使用版本控制工具持積極態度	4.2051	0.8639
BI	八、行為意向 Behavior Intention（BI）	X	X
BI1	BI-1.我打算在以後的專案採用版本控制工具	4.1538	0.9330
BI2	BI-2.我打算經常使用版本控制工具	4.0513	0.9445
BI3	BI-3.我打算成為版本控制工具的忠實使用者	4.1282	0.9509
UB	九、實際行為 User Behavior（UB）	X	X
UB1	UB-1.我在所有的軟體專案中都經常使用版本控制工具	3.4359	1.3533
UB2	UB-2.我推薦我的同學像我一樣使用版本控制工具	4.0000	1.0260

5.7 模型可靠性和有效性測試

關於研究模型的結果，為了計算項目的可靠性、內部一致性與收斂有效性，使用 PLS 演算法對測量模型進行了評估，結果如表 5.4 與圖 5.7。如果指標具有高度相關性和互換性，則被認為是反射性的，應該檢查其可靠性和有效性[30]。「CL」、「α」、「CR」應該大於或等於 0.7，「AVE」值應該大於 0.5[31]，如表 5.2 所示，所有 CL 值都在 0.764 以上、α 值都在 0.801 以上，這表明測量模型的信度良好，並且所有的 CR 值都超過了 0.884，表示內部一致性良好，而 AVE 值皆在 0.708 以上，表示模型的收斂校度很高。

表 5. 4 內部一致性、項目可靠性、收斂有效性

Constructs	Items	CL(>0.7)	α(>0.7)	CR(>0.7)	AVE(>0.5)
Game Usefulness	GU1	0.894	0.920	0.950	0.864
	GU2	0.988			
	GU3	0.904			
Game Motivation	GM1	0.875	0.801	0.879	0.708
	GM2	0.898			
	GM3	0.764			
Performance Expectancy	PE1	0.996	0.992	0.996	0.992
	PE2	0.996			
Effort Expectancy	EE1	0.927	0.886	0.944	0.895
	EE2	0.964			
Self-Efficacy	SE1	0.983	0.966	0.983	0.967
	SE2	0.984			
Hedonic Motivation	HM1	0.954	0.901	0.953	0.910
	HM2	0.954			
Attitude	AT1	0.893	0.941	0.957	0.849
	AT2	0.920			
	AT3	0.943			
	AT4	0.929			
Behavioral Intention	BI1	0.973	0.973	0.983	0.950
	BI2	0.973			
	BI3	0.977			
Use Behavior	UB1	0.938	0.862	0.936	0.879
	UB2	0.937			

* **Notes**: CL - cross loadings; α - Cronbach's alpha; CR - composite reliability; AVE - average variance extracted.

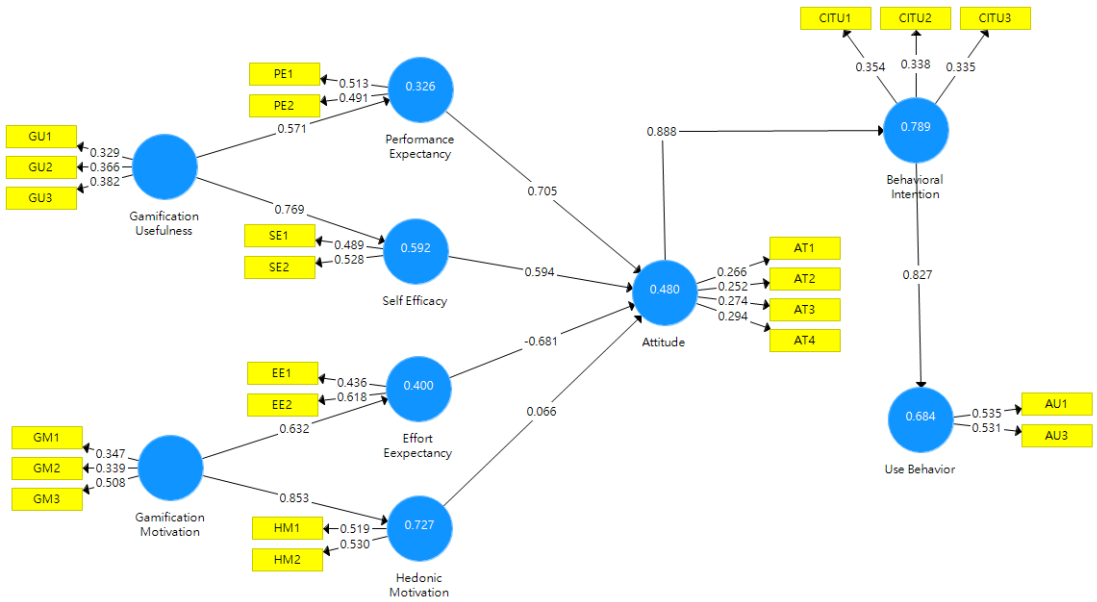


圖 5. 7 PLS 演算法的測量結果

模型驗正的重要步驟還有確認潛在變項之間是否具有良好的收斂效度（Convergent Validity）和區別效度(Discriminant Validity)，根據 Fornell-Larcker 評估收斂效度的標準，當對角線上的數值高於其他構面的 CL 值，結果被認為是可接受的[31]，分析結果如表 5.5。

表 5. 5 Fornell-Larcker 標準分析的結果

	UB	AT	BI	EE	GM	GU	HM	PE	SE
UB	**0.938**								
AT	0.793	**0.921**							
BI	0.827	0.888	**0.974**						
EE	0.371	0.345	0.411	**0.946**					
GM	0.525	0.483	0.500	0.632	**0.841**				
GU	0.442	0.393	0.413	0.605	0.765	**0.930**			
HM	0.421	0.461	0.334	0.584	0.853	0.767	**0.954**		
PE	0.425	0.492	0.447	0.848	0.617	0.571	0.640	**0.996**	
SE	0.677	0.568	0.667	0.656	0.797	0.769	0.574	0.541	**0.984**

* **Notes**: UB - use behavior; AT - attitude; BI - behavior intention ; EE - effort expectancy; GM - game motivation ; GU - game usefulness; HM - hedonic motivation; PE - performance expectancy; SE - self-efficacy.

5.8 結構模型之結果分析

我們評估了結構模型中的潛變量之間的關係，我們使用 Blindfolding 技術來評估研究模型的預測相關性，在此技術下，如果交叉驗證冗餘（cross-validated redundancy，即 Q^2）大於零，模型即被認為具有預測相關性，當大於 0.02 時被認為具有小預測相關性；大於 0.15 時有中等預測相關性；大於 0.35 時有大預測相關性。當解釋方差（explained variance ，即 R^2）大於 0.67 時，被認為是「 substantial」[29][31]，結果如表 5.6。

表 5. 6 解釋方差(R^2) 和預測的相關性 (Q^2)

Dependent variables	R^2 (>0.5)	Q^2 (>0)	Result
Performance Expectancy	**0.327**	0.306	weak
Self Efficacy	0.592	0.670	moderate
Effort Expectancy	**0.400**	0.314	weak
Hedonic Motivation	0.727	0.583	moderate
Attitude	**0.480**	0.378	weak
Behavioral Intention	0.789	0.741	substantial
Use Behavior	0.684	0.593	moderate

P 值通過 Bootstrapping 與雙尾檢定得到，Bootstrapping 對原始數據集進行隨機，以估計 PLS 路徑模型的統計意義[30][32]，結果如表 5.7 所示，內部的一些路徑係數具有統計學意義，結果發現「Game Usefulness」對 「Performance Expectancy」(p = 0.000, t = 3.946), 「Self-Efficacy」 (p = 0.000, t = 13.693)有明顯影響；因此 H1, H2 得到支持。「Game Motivation」對「Effort Expectancy」(p = 0.000, t = 6.648),「Hedonic Motivation」(p = 0.000, t = 28.222)有明顯影響；因此 H3, H4 得到支持。「Performance Expectancy」(p = 0.012, t = 2.516),「Self-Efficacy」(p = 0.008, t = 2.663), 「Effort Expectancy」(p = 0.007, t = 2.701)對「Attitude」有明顯影響，因此 H5, H6, H7 得到支持。「Attitude」對「Behavior Intention」(p = 0.000, t = 20.191)與「Behavior Intention」對「Use Behavior」(p = 0.000, t = 16.097) 有明顯影響，因此 H9, H10 得到支持。

令人意外的是 H8 沒有得到支持，「Hedonic Motivation」(p = 0.740, t = 0.332) 無法對「Attitude」進行預測。

表 5. 7 與直接效應有關的研究假設的檢驗結果 (p** <= 0.01, p* <= 0.05)

H	Relation	Original Sample(O)	Sample Mean (M)	Standard Deviation (SD)	T Statistics (>1.96)	P Values (<0.05)
H1	GU->PE	0.571	0.567	0.145	3.946	0.000
H2	GU->SE	0.769	0.776	0.056	13.693	0.000
H3	GM->EE	0.632	0.638	0.095	6.648	0.000
H4	GM->HM	0.853	0.869	0.030	28.222	0.000
H5	PE->AT	0.705	0.676	0.280	2.516	0.012
H6	SE->AT	0.594	0.617	0.223	2.663	0.008
H7	EE->AT	-0.681	-0.670	0.252	2.701	0.007
H8	HM->AT	0.066	0.070	0.200	**0.332**	**0.740(NS)**
H9	AT->BI	0.888	0.890	0.044	20.191	0.000
H10	BI->UB	0.827	0.890	0.044	16.097	0.000

* **Notes**: UB - use behavior; AT - attitude; BI - behavior intention ; EE - effort expectancy; GM - game motivation ; GU - game usefulness; HM - hedonic motivation; PE - performance expectancy; SE - self-efficacy.

5.9 研究問題之結果分析

上一節當中已經介紹了實驗的結果並完成結構模型的分析與驗證，本節將依照實驗與結構模型的結果回應所述的研究問題。

5.9.1 研究問題 1

關於研究問題 1，從後置測驗的結果來看，實驗組在三項要求的通過率皆優於控制組，因此可以得知加入 GEG 作為教學輔助工具是比傳統的授課方式具有更高的學習成果。但是衝突解決的題目通過率並不高，這點我們認為這是由於在

這門課程中並沒有需要協作的作業或專題，分支與合併的功能不被大多學生所需要，也因為學習的難度較高，使這項要求的通過率相對較低。

5.9.2 研究問題 2

關於研究問題 2，必須從測量模型的結果來看，首先根據 H1, H2, H3, H4, 的假設得到支持來看，GEG 設計的遊戲元素在學生的認知裡為他們學習 Git 帶來了正面的影響；根據 H5, H6, H7 的假設得到支持來看，績效預期、努力期望、自我效能評估等使他們對學習及使用 Git 的態度有正面的影響；從 H9, H10 的成立來看，對於 Git 的態度影響學生後續使用 Git 的意願，並進一步影響學生後續的使用行為。

因此可以認為，GEG 作為教學輔助工具可以為學生對學習、使用 Git 的態度帶來正面影響，並影響其後續行為。此外 H8 不成立令人感到意外，我們認為是由於享樂主義動機無法直接對 Git 的態度造成影響，僅能提高學生在參與學習時的動機。

5.9.3 研究問題 3

關於研究問題 3，首先我們的紀錄資料顯示，大約有 55 位學生參與 Git 的教學活動，而帳號卻有 98 位，其中有一大部分帳號是在實驗結束後一段時間註冊的，並且來源自學校的 ip，可以推測是忘記密碼的學生重複註冊的，有一小部份（約 14 位）則是完全沒有活動的帳號，大多是註冊為非學號的帳號，是被學生棄用的帳號。本研究根據學生在實驗日後是否仍有活動事件判斷學生是否進行了主動學習，而在 55 位學生當中，約有 34 位學生在實驗過後一段時間仍有活動紀錄，也就是說，至少約有 60%的學生有以 GEG 進行主動學習。

由於不少在實驗後活動的帳號不是以學號命名，因此我們只選取約 27 位以學號命名並且有進行主動學習的學生與全體實驗組進行成績對比。我們發現有進行主動學習的學生在各方面都取得了更佳的成果，如圖 5.8。

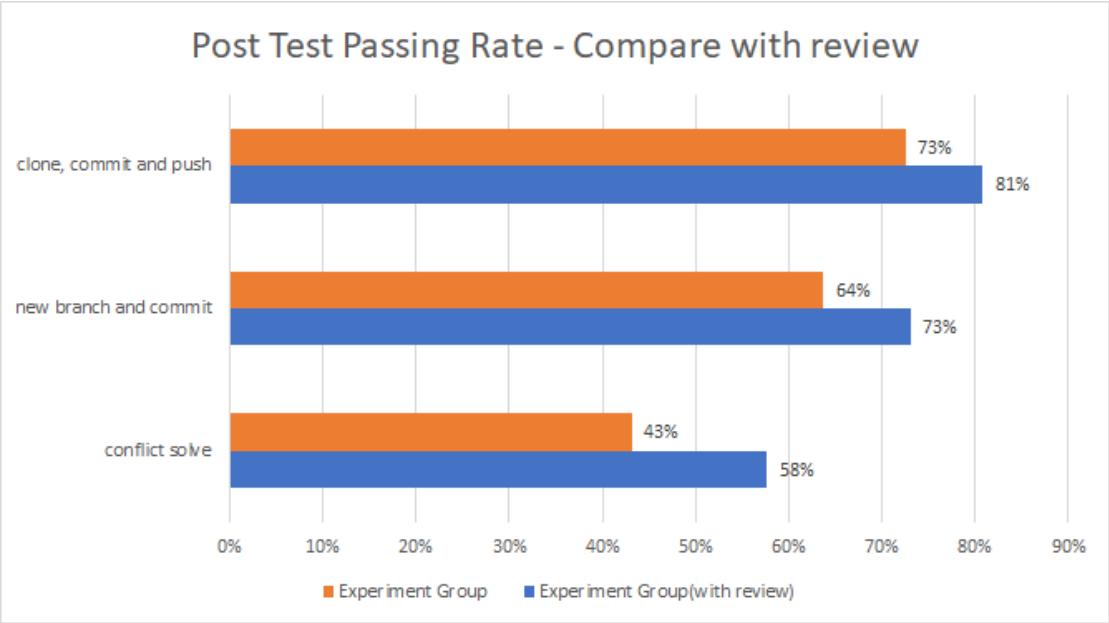


圖 5. 8 後置測驗通過率（與有 review 的學生比較）

5.9.4 研究問題 4

關於研究問題 4， 我們統整了問卷中 OEQ1 與 OEQ2 的問卷回覆，將類似的回饋整合，並列出回饋所提到的優點：

1. 有趣較不無聊，更有動力學習（42%）
2. 容易理解與知道錯誤，有反饋的學習很好（30%）
3. 介面設計更加容易上手、網站方便與好看（19%）
4. 學習難度循序漸進，比較容易學習（19%）

回饋中提出的缺失則有大約以下幾個：

1. 會當機或卡住(30%)
2. 指令因為簡化過，在真正使用時可能會搞錯(30%)
3. 有些提示或說明不清楚，會看不懂（30%）

此外有兩個回饋分別提到希望課程有更多時間，以及認為某些關卡過於困難（推測可能是因為關卡八與關卡九）。

關於這些負面回饋，我們打算首先根據後台的資料找出學生在什麼情況下發生當機，修正這些狀況，並將簡化過的指令還原為原始的 Git 指令，以避免學生在實際使用時的混淆，最後則要加強提示，藉由在說明中示範與增加更多圖示、描述以令學生更加容易讀懂遊戲的關卡提示。

第六章 結論與未來研究

版本控制軟體的使用能力對軟體工程師而言至關重要，然而學校教育對其著墨甚少，與之相關的研究則大多是結合版本控制的應用軟體，而軟體工程的非傳統教育方式當中，遊戲式學習具有相當潛力，因此本研究開發了一個用於教育 Git 的嚴肅遊戲，並設計與進行一個教育研究實驗，評估系統的教育效果，同時本研究基於 UTAUT2 模型設計了一個研究模型，並通過 PLS-SEM 對問卷調查結果的數據進行分析與測試。

根據研究結果我們發現加入 GEG 作為教育輔助工具的組別相比單純傳統授課的組別有更高的學習成果，並且在這些學生當中，會進行主動學習的學生超過了一半，而這些學生相比整體實驗組學生具有更高的學習成果。此外模型中的因素除了享樂動機以外皆對學生對於 Git 的態度產生了明顯的直接影響，而遊戲設計元素也使學生對於 Git 的態度產生了間接影響，進而影響學生的行為意圖與後續行為，因此可以認為本研究所提出之系統使學生產生對學習、使用 Git 的正面影響。

關於未來研究，我們打算更改遊戲的內容，使其在使用時更接近於真實的 Git 指令，並修正回饋中提到的當機、指示不清楚等等問題。此外我們打算增加更多關卡，這些關卡可以統稱為「挑戰」，如遊戲中的進階關卡，提供學生更多練習機會，藉由將學生較不容易理解的部份整合成各種情境挑戰，使遊戲從單純的入門工具成長為可以兼具入門與熟練的工具，遊戲性的部份則要為點數機制設計更多可應用的場景，並增加更多成就、獎章。

參考文獻

[1] Lassi Haaranen and Teemu Lehtinen. 2015. Teaching Git on the Side: Version Control System as a Course Platform. In Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '15).

[2] Zichermann, G., & Linder, J. 2010. Game-based marketing: inspire customer loyalty through rewards, challenges, and con-tests. John Wiley & Sons..

[3] Christiane Gresse von Wangenheim, Rafael Savi, Adriano Ferreti Borgatto, SCRUMIA—An educational game for teaching SCRUM in computing courses, Journal of Systems and Software, Volume 86, Issue 10, 2013, Pages 2675-2687, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2013.05.030.

[4] V. Venkatesh, J. Y. L. Thong, and X. Xu, ‘‘Consumer acceptance and use of information technology: Extending the unified theory of acceptance and use of technology,’’ MIS Quart., vol. 36, no. 1, pp. 157–178, 2012.

[5] H. M. Lin, M. H. Lee, J. C. Liang, H. Y. Chang, P. Huang, and C. C. Tsai, ‘‘A review of using partial least square structural equation modeling in E- learning research,’’ Brit. J. Educ. Technol., vol. 51, no. 4, pp. 1354–1372, 2019.

[6] F.-H. Huang, ‘‘Adapting UTAUT2 to assess user acceptance of an e- scooter virtual reality service,’’ Virtual Reality, vol. 24, no. 4, pp. 635–643, Dec. 2020.

[7] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis, ‘‘User acceptance of information technology toward a unified view,’’ MIS Quart., vol. 27, pp. 425–478, Jul. 2003.

[8] Uskov, A.; Sekar, B. Serious Games, Gamification and Game Engines to Support Framework Activities in Engineering: Case Studies, Analysis, Classifications and Outcomes. In Proceedings of the IEEE International Conference on Electro/Information Technology, Milwaukee, WI, USA, 5–7 June 2014; pp. 618–623.

[9] Fleming T, Cheek C, Merry S, Thabrew H, Bridgman H, Stasiak K, et al. Serious games for the treatment or prevention of depression: a systematic review. Revista de Psicopatología y Psicología Clínica (2014) 19(3):227-2.10.5944/rppc.vol.19.num.3.2014.13904

[10] Deterding S, Dixon D, Khaled R, Nacke L. From game design elements to gamefulness: defining gamification. Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments. Tampere: ACM; (2011). p. 9–15.

[11] Fleming, T.M.; Bavin, L.; Stasiak, K.; Hermansson-Webb, E.; Merry, S.N.; Cheek,

C.; Lucassen, M.; Lau, H.M.; Pollmuller, B.; Hetrick, S. Serious Games and Gamification for Mental Health: Current Status and Promising Directions. Front. Psychiatry 2017, 7, 215.

[12] Burke JW, McNeill MDJ, Charles DK, Morrow PJ, Crosbie JH, McDonough SM. Optimising engagement for stroke rehabilitation using serious games. Vis Comput (2009) 25(12):1085–99.10.1007/s00371-009-0387-4

[13] Chatham RE. Games for training. Commun ACM (2007) 50(7):36–43.10.1145/1272516.1272537

[14] Mayer, I. Playful Organisations & Learning Systems; Breda University of Applied Sciences: Breda, The Netherlands, 2016.

[15] Katie Seaborn, Deborah I. Fels, Gamification in theory and action: A survey, International Journal of Human-Computer Studies, Volume 74, 2015, Pages 14-31, ISSN 1071-5819, https://doi.org/10.1016/j.ijhcs.2014.09.006.

[16] A. Baker, E. O. Navarro and A. van der Hoek, "An experimental card game for teaching software engineering," Proceedings 16th Conference on Software Engineering Education and Training, 2003. (CSEE&T 2003)., 2003, pp. 216-223, doi: 10.1109/CSEE.2003.1191379.

[17] A. Radermacher and G. Walia. Gaps between industry expectations and the abilities of graduates. In Proceeding of the 44th ACM technical symposium on Computer science education, pages 525–530. ACM, 2013.

[18] J. Kelleher. Employing git in the classroom. In Computer Applications and Information Systems (WCCAIS), 2014 World Congress on, pages 1–4. IEEE, 2014.

[19] J. Lawrance and S. Jung. Git on the cloud. Journal of Computing Sciences in Colleges, 28(6):14–15, 2013.

[20] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis, ‘‘User acceptance of information technology toward a unified view,’’ MIS Quart., vol. 27, pp. 425–478, Jul. 2003.

[21] S. A. Salloum, M. Al-Emran, A. A. Monem, and K. Shaalan, ‘‘Exploring students’ acceptance of E-learning through the development of a comprehensive technology acceptance model,’’ IEEE Access, vol. 7, pp. 128445–128462, 2019.

[22] K. Tamilmani, N. P. Rana, and Y. K. Dwivedi, ‘‘Consumer acceptance and use of information technology: A meta-analytic evaluation of UTAUT2,’’ Inf. Syst. Frontiers, vol. 23, no. 4, pp. 987–1005, Aug. 2021.

[23] Baptista, Goncalo & Oliveira, Tiago. (2017). Why so serious? Gamification impact in the acceptance of mobile banking ser-vices. Internet Research. 27. 118-139. 10.1108/IntR-10-2015-0295.

[24] LIU, Eric Zhi Feng. Avoiding internet addiction when integrating digital games into teaching. Social Behavior and Personality: an international journal, 2011, 39.10: 1325-1335.

[25] Anderson, Lorin W., and David R. Krathwohl. A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Longman,, 2001.

[26] Star, K. 2016. Gamification, Interdependence, and the Moderating Effect of Personality on Performance. Doctoral Thesis. University of Coventry.

[27] Michael D. Hanus and Jesse Fox. Assessing the effects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance. Computers & Education, 80:152–161, 2015

[28] Juho Hamari, Jonna Koivisto, and Harri Sarsa. Does gamification work? – a literature review of empirical studies on gamification. In Proceedings of the 2014 47th Hawaii International Conference on System Sciences, HICSS '14, pages 3025–3034, Washington, DC, USA, 2014. IEEE Computer Society.

[29] J. F. Hair, C. M. Ringle, and M. Sarstedt, ''PLS-SEM: Indeed a silver bullet,'' J. Marketing Theory Pract., vol. 19, no. 2, pp. 139–152, Apr. 2011.

[30] J. F. Hair, C. M. Ringle, and M. Sarstedt, ''Partial least squares structural equation modeling: Rigorous applications, better results and higher acceptance,'' Long Range Planning, vol. 46, nos. 1–2, pp. 1–12, 2013.

[31] K. K. K. Wong, ''Partial least squares structural equation modeling (PLS- SEM) techniques using SmartPLS,'' Marketing Bull., vol. 24, no. 1, pp. 1–32, 2013.

[32] J. F. J. Hair, G. T. M. Hult, C. M. Ringle, and M. Sarstedt, ''A primer on partial least squares structural equation modeling (PLS-SEM),'' Eur.J. Tourism Res., vol. 6, pp. 211–213, Dec. 2014.