軟體產業與軟體工程

軟體開發市場環境逐漸成熟，一個專案的上架速度壓縮快進，使得軟體產業前延快速變化，加速新興應用的發展，也衝擊著所有消費市場。

媒合平台的崛起，如:Airbnb、Uber，整合了現今市場的龐大資訊量，以中間商的身分面向使用者，提供買方與賣方全新的商業模式，更改變了大家的生活習慣。至此，軟體以單機、網頁、手機 APP 的形式，與大眾生活緊密結合。

建構理想軟體的方法，需要思考三大面向:需求、流程、技術(PPT)。發掘市場的潛在需求，提供一個穩定的、品質保證的軟體，並且有彈性能因應興新技術的迭代更新。也因此，如何能達成理想的軟體建構方法，不單單是技術工程師獨有的課題，團隊合作是必不可少的，各方面人才的技能與思維碰撞，最後生成專案供市場檢核。

在不斷迭代更新的現在，新世代的思考邏輯，或許該從專精單一的 I 型人才，轉變為擁有軟實力的 T 型斜槓人才。

軟體工程流程與團隊

把真實世界的應用需求依流程轉化為軟體，將系統化、規範化、可度量的方式應用於軟體開發、執行與維護，即是軟體工程。

軟體開發流程可視情況調整，但大致分為以下步驟，並同步進行專案管理:

一、需求分析:捕捉使用者需求，分析系統可行性、需求正確性等。常文件化撰寫需求分析，可情境思考市場需求。

二、設計:大框架設計，包含架構設計、演算法設計與資料結構設計，強調彈性。

三、實作:撰寫程式碼，實現軟體。

四、測試:檢測軟體功能、效能等是否契合需求設計。

專案管理是以既定目標來規劃一系列子流程，用以確保項目進度。軟體專案的參與角色可包含以下九種:

客戶、主管:提供產品需求擷取與分析，參與所有開發流程。

系統分析師 SA:研究應用領域、需求分析。聯繫使用者與開發人員。

程式設計師 PG/研發人員 RD:撰寫程式碼。

測試工程師 ST/QA:執行軟體的功能和非功能(效能等)測試。

使用者體驗設計師 UX:負責使用者感受，規劃流暢的操作邏輯。

使用者介面設計師 UI:負責使用者介面布局，視覺化體現操作流程。

前端工程師 F2E/前端 Web 開發人員:負責選擇、開發與測試網站的使用者介面，設計網站外觀和功能。

客戶支援工程師 FAE:負責售前、售後支援工作。了解功能特性與客戶溝通協調。

敏捷開發方法-Scrum

Scrum 是一種敏捷式的軟體開發專案管理，迭代、遞增式的頻繁產出成果以反覆評估目標與願景。

有別於原先的定義式流程強調計畫、指令與控制、嚴謹的變更控制機制，實務流程擁抱改變、根據歷程來學習、觀察並調適，適合較為複雜或無法預期的軟體開發工程。Stacey Matrix 以 How 和 What 兩個面向，把專案劃分為 Simple、Complicated、Complex、Chaotic 四種複雜程度，Complicated 和 Complex 即適合更靈活的敏捷開發方法。

Scrum 遵循著固定的流程框架。將專案歷程切割為固定短周期的 Sprint，每次必產出具體成果，逐步疊加成既定願景。Scrum Team 有三種主要角色:

Product Owner (產品擁有者):願景的產品特性，包括功能與需求。

Development Team (開發團隊):根據需求進行專案開發。

Scrum Master:協助團隊自主發揮功能、為團隊排除障礙。

Scrum 的產出可分為:

一、產品待辦清單(Product Backlog):

含多項 Product Backlog Item(PBI)，PBI 常以使用者故事呈現。

二、工作待辦清單(Sprint Backlog):

將 PBI 拆解成多項 SBI，優先挑選高價值高獲利(ROI)的 SBI。

三、潛在可發布增量(Potentially Releasable Increment, PRI):

可隨時交付給客戶的軟體版本，Sprint 逐步疊加軟體功能。

因為 Sprint 的特性，Scrum 需要開四種會議:Sprint Planning、Daily Scrum、Sprint Review、Sprint Retrospective。

程式碼版本控制簡介

Git 是一個版本控制系統。建立、同步、歷史回復等版本控制功能使得產品開發有了良好的共同創作平台。

Github 的服務出現吸引了大量人流，其魅力在於個人化、社群化和互動協作，如提供使用者營運自身版面、觀摩他人作品、留言互動等功能體驗。不僅如此，Github 提供了速度、離線化，以及更容易排解版本衝突。

以下為 Git 的幾項操作指令:

一、常用初始化指令:

git init(目前的資料夾)

git init demo(新增 demo 資料夾)

二、查看狀態:

git status(顯示分支名稱、提交歷史紀錄和檔案)

三、設定使用者姓名跟信箱，依據使用層級範圍:

--local(專案範圍設定配置)

--global(使用者範圍設定配置)

--system(對系統內所有使用者配置)。

四、config 常用指令:

git config --global user.name "demo_user"(設定使用者名稱)

git config --global user.email "demo_user@demo.com"(設定使用者信箱)

git config --global alias.st "status"(設定 status 暱稱)

git config --list(查看設定)

五、add 常用指令，新增檔案並列入追蹤(編列索引):

git add README.md(將指定的檔案建立 index)

git add(將目前目錄現在更變建立 index)

git add --all(將所有變動的檔案建立 index)

六、commit 常用指令，提交當前目錄的變動檔案並提交備註訊息:

git commit-m "備註訊息" (新增一筆提交紀錄)

git commit -a -m "備註訊息" (git add .+ git commit -m)

軟體測試簡介

程式錯誤(Bug)是指軟體執行時因程式本身有錯誤(程式碼編寫錯誤，多為邏輯錯誤)而產生當機、資料遺失、功能不正常、非正常中斷等錯誤現象。理想中的 Bug-free software 在現實執行中幾乎不存在，因此，我們需要良好的測試方式來檢測並排除程式錯誤。

軟體測試類型可依據技術/商務導向、支援程式設計/評估專案，分為四個象限:

一、系統、整合、單元測試:偏向為工程師導向，含黑箱、白箱測試。

二、功能驗收測試:為使用者面向的狀態測試，主流是端對端測試，確保使用者操作流程順利無誤。有自動化工具跑腳本測試。

三、非功能性驗收測試:含壓力測試、跨平台測試、安全測試(滲透測試)等。可為自動化、手動測試。

四、探索性測試:仰賴人(如經驗、遍歷)手動操作找出漏洞。