敏捷式開發

敏捷式開發(agile  Development)並非一種制式的開發方法，而是一種軟體開發的精神(spirit)，任何開發方法都可以加入敏捷式開發的一些原則進而改善軟體開發的成效。

在敏捷式開發中有個很重要的觀點是筆者很感興趣的，它認為塑模（Modeling）的目的在於增加開發者了解軟體的程度，進而使得軟體更接近於使用者的需求，而非使用塑模之後產生的文件。

The purpose of modeling (Sketching UML,…) is primarily to understand,not to document.[Apply UML and Patterns,3/e]

換句話說，它希望開發者使用塑模的時機，是當使用這個技術有助於開發者更了解被開發的軟體時才使用，例如某些具關鍵性的議題或者高風險性的項目，而非不管三七二十一的將軟體所有範圍的設計都加以塑模留下文件。

草稿與藍圖

有趣的是，如果我們延伸這個觀點，塑模在敏捷式開發的精神下是一種類似草圖或者草稿的作用。也就是說，用以在團隊開發時討論以及研究議題的一種工具，在過程中利用塑模的技術來讓問題得到解決，一開始的動機並非為了留下設計圖讓程式設計師去實作。

同樣的觀點在Martin Fowler的UML Distilled Third Edition中也有所著墨，Martin認為，UML作為一項工具，可以有三種被使用的方式，一種是草圖，一種是所謂的藍圖，最後一種則是程式語言。

Martin本身則是將作為一種草圖的工具來使用，當然也有人將其作為藍圖來使用。而將UML作為藍圖來使用到一種極至時將出現UML本身就可視為一種程式語言，例如所謂的模型驅動（MDA）開發。當然這並不在本文的討論範圍內。

Agile Alliance

2001年時支持敏捷式開發的社群組成了Agile Alliance，並且發表了Agile宣言與原則。

The Agile Manifesto (敏捷宣言)

獨立的工作成員與人員互動 勝於 流程與工具的管理

工作產生的軟體 勝於 廣泛而全面的文件

客戶的合作 勝於 契約的談判

回應變動 勝於 遵循計畫

The Agile Principles (敏捷原則)

1. 最為優先的事情是透過早期與持續交付有價值的軟體來使客戶滿意。
2. 歡迎需求的變動，即使是在開發的晚期。敏捷式流程駕馭變動來作為客戶的競爭優勢。
3. 頻繁的交付工作產生的軟體，自數週至數月，週期越短越好。
4. 領域專家與開發成員必須一同作業，並貫穿整個專案開發時期。
5. 使用積極的工作成員來建構專案，給予他們環境以及支援所需的一切，然後信任他們能夠完成工作。
6. 在開發團隊中最快也最有效的傳遞資訊方法就是面對面的溝通。
7. 工作產生的軟體是衡量進度最主要的依據。
8. 敏捷式流程倡導水平一致的軟體開發
9. 專案發起者，開發人員以及使用者都必須持續的維持專案進度。
10. 持續重視技術的優勢以及設計品質
11. 最好的架構、需求以及設計會出現在能夠自我管理的團隊裡
12. 在規律的反覆之間，團隊會反省與思考如何更有效率，然後相對的來調整與修正團隊的開發方式。

透過以上的條文相信讀者都能比較了解敏捷式開發的一些精神。當然，既然稱作一種原則，就代表團隊引用敏捷式開發時，並非照本宣科的一股腦引用，而是應該審視團隊與公司的文化及產業特性，來評估能夠參考的部份。

延伸思考

其實延伸敏捷式開發的議題，筆者聯想到一個一直在台灣軟體資訊界爭論不休的問題，就是CMMI究竟對臺灣軟體有沒有幫助。其實敏捷式開發的精神與CMMI這類流程與制度導向的觀點在某些層面是有衝突的，如果制度化一切是個良方，那麼敏捷式開發出現就顯得沒有道理。

但是反過來思考，制度化的開發也使得印度內的軟體公司不斷壯大，他們很顯然的不會是使用敏捷式開發來運作。

若是有時間，或許大家可以來思考這樣的議題，希望對於台灣的軟體界能夠有所助益吧。

關於軟體開發到底該重視制度還是強調自由，網友Kate的回應值得參考：

我覺得軟體開發，就好比養一個小孩一樣 (養成遊戲的觀點), 在適當的年齡還是得進入必要的管理流程，如果讓小孩自己順著他的天分成長，少有的天才，才能在18歲大學畢業。

我從 Programmer 晉升到專案經理後，在回頭看看這個過程，才發現當眼界變大變廣的時候，思考角度又不同了。

CMMI全名Capability Maturity Model – Integrated，是SEI繼CMM成功發展後的新修訂版本，不再只限定於軟體開發成熟度，增加整合和服務等不同專業領域的能力成熟度模式，仍以系統工程和軟體工程為基礎架構，套用於任何規模大小的組織，促進其流程的改善。

舉例及回應：

1. 客戶的合作 勝於 契約的談判----------同意

當需求訪談時，為了要討好客戶，也為了整個Team 在客戶端有好的工作atmosphere，對於客戶的要求，多半會做很多溝通，有時就義務性的多給1,2個 Service. 但要求多了，就得把合約拿出來提醒一下、PM 要保持良好的交際手腕—打電話給合約部門或洽談者接洽。由合約部門的來扮演黑臉。

2. 領域專家與開發成員必須一同作業，並貫穿整個專案開發時期

我記得自己是程式設計師時，參加需求訪談時，客戶著重在講流程，在初始定義的時候，brainstorming(集思廣益、團體討論) 做一堆，當時覺得這些人真是囉唆，例如一個下拉值可以做很多不同的選擇 list, 把要選的東西定好就行了，但是Field Define, list item define, option define, 怎麼 user site 的不能區分清楚呢，覺得好浪費我的時間喔～

等到我變成專案經理的時候，老闆著重在成本節省，很多programmer 只要專心coding 即可，因此，在project initiate 時候，只有Project manager , process analyst 分析，程式設計師不必介入，因為他們還在忙別的project 上面的coding.

等到 project lifecycle from initiate transfer to design. PM 要花很多時間從頭講起，若直接切入重點，會有很多回應說應該如何做，其實都是之前與客戶討論過的，只不過是在沙盤推演的時候，因為執行性的不合適被否決，有些是起因於標準，或是已經的policy. 這時，你不得不告訴Engineer 照做就行的。別再扯了。

但心裡也犯低估，老闆要是准許他們一開始參加會議，PM我就不必解釋這麼多了，寫的文件，人家也懶得看，還不如用講的比較快～

但是，有可能你的需求訪談要經歷2,3個月，因為Project scope (專案範疇)and Business process 不同，在Cost view 的考量下，不可能這麼做喔～

外界定了一堆 CMM/CMMI, 北美還有 SDLC/ITIL 在有了實際經驗後，去參加PMP Project Management Profession,

真的要說死老外，定這麼多東西，要累死這些 IT 領域的工作者～

但是說真的，管理階層看的View 就好比父母親，希望自己的孩子能在10歲就大學畢業既能省時，又有good名聲，

這時就引進一堆管理手法，讓自己的小孩，照書養～這樣也比較能預期成品在什麼時候可以達到什麼樣的火候。

而天真快樂的小孩本身-(程式設計師本身)，重點看的是這個過程能累積什麼樣好的記憶，好氣氛，好情景在自己腦海裡，而後顯現出自己的行為給外在的人做評估，這個部分不用做文件的。隨便長，也是一個樣，照書養也是一個樣，基本上功能性健全就可以了，至於氣質及完善程度，或有沒有賣相就無所謂了。

這是我個人的比喻及看法，也許有些人看不太懂我的比喻，Product Lifecycle Management and Project Lifecycle Management 又是另一套大系統，我從這個導入經驗中得到很多人生成長的想法，所以就用這個例子來比喻。

作者：呂中力Terence Leu

作者簡介：曾任電視節目執行製作、網站企劃、程式設計師、軟體公司研發部副理。深信在全球化浪潮下，只有不斷的思考才是成長的唯一途徑。

From：<https://www.projectup.net/article/view/id/15726>