以可再用元件提升行動裝置 App 開發效率與品質

賴森堂
實踐大學資訊科技與管理學系 stlai@mail.usc.edu.tw

摘要

資訊網路熱潮下的 E 世代，行動裝置已成為人們隨身攜帶的一項物品，各種行動裝置中又以智慧型手機居多，攜帶方便、高度互動性、多樣化通訊方式與處理能力，是行動裝置的優勢。行動裝置應用軟體(Mobile Apps)正是此熱潮下的重要產品，為了滿足行動裝置用戶的需求、加強互動性與快速反應時間，行動裝置的硬體技術持續不斷的推陳出新，App 開發業者須隨裝置環境與技術持續的成長，快速開發高品質 App 以取得市場先機，如何有效縮短 App 的製程且保有關鍵的品質特性，已成為值得深入探究的課題。高品質元件是組構 App 的基層要件，為了有效提昇行動裝置 App 的開發效率與品質，就必須規劃一套優質可再用元件的確認機制，本文以文件化與品質特性為基礎，提出一套 App 可再用元件品質量測(*ARCQM*)模式，透過量化的評量機制確認可再用元件的關鍵品質，再搭配元件的分類與擷取功能，用以具體提昇行動裝置 App 的開發效率與品質。

關鍵詞：可再用元件、行動裝置、品質特性、App、*ARCQM*

1. 緒論

在網路應用與網路互動的熱潮下，促使行動裝置的普及率與用戶數持續不斷的增加，成長速度已超越個人電腦(產業新聞，2011)，旅遊、電腦網路遊戲、娛樂、社交網路及生產力等應用程式，都可以在智慧型手機或平板電腦等行動裝置上，充分的展現其特性與優勢(Bulter, 2011 ;Gavalas and Economou, 2011)，使得行動裝置應用程式(Applications; Apps)開發業者必須不斷且快速推出新的版本與應用，以迎合使用者的需求。行動裝置 App 存在許多的商機，App 採取低價行銷策略，用戶在幾乎沒有負擔的情況，直接透過網路下載且付款，促使一套熱賣的 App，短期內就可以獲得高額的收入，不過，更有許多 App 乏人問津，連開發成本都難以回收，因此如何從充滿商機的行動裝置 App 市場取得一席之地，是一項值得探究的議題。綜觀受歡迎的 App 除了必須具有高度的創意與吸引人的內容外，產品品質與開發效率更是成功重要關鍵。因此，為了有效提昇行動裝置 App 的開發效率與品質，本文探討且規劃一套優質可再用元件的確認與改善機制，用以協助 App 的開發作業。

隨著硬體技術與運作環境的不斷變化，行動裝置 App 為維持其市場商機，必須配合裝置技術的成長與外界環境的變動，不斷調整、修訂、擴充與強化其產品的適應力與效率，才能持續擁有廣大的用戶。行動裝置 App 依開發方式的差異，可以劃分成三種類型：(1) 全新(All New)開發方式—從無到有，透過創意規劃、分析、設計、製作與測試等步驟，完成一套新的 App。(2) 翻新(Transformation)開發方式—挑選已上市且受歡迎的

App，進行調整、修訂與測試，以適應新的行動裝置。(3) 跨平台(Cross Platform) 開發方式—從其他行動裝置平台上的 App，移植至目的作業平台(Gavalas and Economou, 2011)。無論採取那一類型的開發方式，最重要的目標就是快速完成受歡迎的 App，其中可再用元件是達成此目標的主要關鍵，因為，優質元件是組構行動裝置 App 的基層要件。任何類型的開發方式都與 App 元件關係密切，將優質的可再用特性融入 App 元件，再結合分類與擷取功能，建立一套高效率的行動裝置 App 開發製程，可以具體提昇行動裝置 App 的開發效率與品質，且有效縮短行動裝置 App 的開發時程。

行動裝置的硬體技術日新月異，持續不斷的推陳出新，隨時都有新機型或新版本被推出，促使行動裝置 App 的開發效率或適應能力，面臨強大的壓力與挑戰(Huntley, 2011)，本文針對行動裝置 App 開發型態與關鍵的特質進行探討，且深入探究 App 可再用元件與開發作業間的相互關係，剖析出模組單元、程式單元與單元測試等項目對於 App 建置作業的影響最為顯著，因為這些單元項目就是建構行動裝置 App 的基層元件，將可再用的品質特性融入這些基層元件中，使元件成為優質可再用元件，勢必可以有效改善行動裝置 App 的開發效率與品質，縮短行動裝置 App 的開發時程，且延續其生命期。為此，本文以文件化與品質特性為基礎，提出一套 App 可再用元件品質量測 (App Reusable Component Quality Measurement; *ARCQM*) 模式，透過量化的評量機制標示出 App 元件的品質缺失，再依缺失原因提出改善作業，用以強化元件的可再用品質，進而使行動裝置 App 的開發品質與效率可獲得提昇。本文第二節將探討行動裝置 App 的特性與開發環境，且說明不同開發方式在時程、品質、花費與成果展現的優勢。可再用元件直接衝擊行動裝置 App 的開發效率與品質，第三節將討論可再用元件對 App 的開發優勢，且說明如何透過以文件化與品質特性強化 App 元件的可再用性。量化機制是確認 App 元件是否具備可再用性的關鍵，第四節將提出一套 App 可再用元件品質量測 (*ARCQM*) 模式，將元件可再用品質加以量化，以確認元件的可再用性。第五節將以法則式結合元件品質量化的結果，進行可再用元件品質的強化與改善作業。第六節說明優質可再用元件具體提昇行動裝置 App 的開發效率與品質，且針對本文主題作個結論。

2. 行動裝置 App 的特性與開發環境

結合網路平台與互動介面的行動裝置 App 有別於一般應用軟體的特性，本節將探討行動裝置 App 的特性以及開發環境。

2.1 行動裝置 App 的特性

行動裝置已成為人們隨身攜帶的一項產品，眾多的行動裝置中又以智慧型手機(如 Apple iphone, HTC desire 等手機)居多，行動裝置的硬體配置有別於一般的個人電腦，行動裝置強調攜帶方面、觸控式操作介面、快速的資料存取、多項的通訊能力等，至於強大處理能力、大量儲存空間、多樣化輸出入裝置及實體網路通訊等，則不是行動裝置所訴求的項目，表一即為行動裝置與個人電腦之硬體配置特質比較表。受到硬體配置不同與開發時效的影響，行動裝置 App 與一般個人電腦 App 也有很許多差異，以下從五個層面說明行動裝置 App 的差異：

(1) 觸控操作介面：行動裝置的操作介面較為單純，一般都是透過螢幕輸出入，因此具

備高度親和力的觸控式操作介面是吸引行動裝置使用者的重要因素，也是行動裝置 App 受歡迎的關鍵。

(2) 快速開發效率：行動裝置的硬體技術日新月異，持續不斷的推陳出新，隨時都有新機型或新版本被推出，行動裝置 App 開發業者必須具備快速開發能力才持續滿足使用者的需求。

(3) 高擴充與調整能力：行動裝置 App 必須隨著通訊環境與硬體設備的演進，進行相關功能的擴充與調整， App 須具備高可維護性，才能即時調整與擴充各項功能，以滿足用戶的需求。

(4) 易安置新環境：行動裝置的市場競爭非常激烈，設備廠商為了取得較高的市占率，經常配合新版作業系統不定期推出功能更先進的設備，因此行動裝置 App 必須具備高度移植與跨平台能力，以便快速將 App 安置於新的設備環境上。

(5) 物件導向開發平台：以目前較受歡迎的行動裝置中，開發平台與開發語言都以物件導向為基礎(如 iOS—Objective-C, Android—java, Window phone—java 等)，善用物件導向的優勢與特性，可以有效提昇行動裝置 App 的開發與維護品質。

表一：行動裝置與 PC 之硬體配置特質比較表

比較項目 / 硬體配置特質	**行動裝置(手機/平板電腦)**	**個人電腦/筆記型電腦**
CPU 功能	弱	強
記憶體存取速度	快	慢
螢幕大小	小	大
螢幕觸控能力	強	弱或無
輸出入裝置	單純化	多樣化
網路通訊能力	無線通訊	實體/無線通訊

2.2 行動裝置 App 的開發環境

目前市面上眾多行動裝置搭配許多種類的作業系統，其中以下面四種作業系統安裝於行動裝置的比率最高：Apple/iOS, Google/Android, *Nokia/Symbian*, MicroSoft/Window Phone 等(Hense etc., 2009; Gavalas and Economou, 2011)，不同作業系統提供的 App 開發平台也不一樣，採取的開發程式語言也不同(參閱表二所示)。工欲善其事，必先利其器，進行行動裝置 App 開發作業之前，必須對於行動裝置平台採用的作業系統深入瞭解，且熟悉 App 的開發工具與實作的程式語言。剖析 App 的開發工具與程式語言，可以發現行動裝置 App 較傾向於物件導向的開發模式，此開發模式具備封裝、繼承、模組化、資料抽象化、易於擴充與維修等優勢，若再配合優質可再用元件，且採取以元件為基礎軟體開發方法(Component-based software development; CBSD)(Dogru, 2003; Szyperski, 1998; Vitharana, 2003)，勢必大幅提昇行動裝置 App 的開發效率與關鍵品質，更可以有效延續 App 的生命期。

表二：關鍵行動裝置 App 開發環境的比較表

設備廠商 **App 開發環境**	***Apple***	***Google 等***	***MicroSoft***	***Nokia 等***
作業系統	***iOS***	***Android***	***Window Phone***	***Symbian***
開發語言	**Objective-C**	**Java**	***Visual Studio and Java***	***C++***
開發工具	***iOS SDK***	***Eclipse***	***Windows Phone SDK***	***Eclipse***

目前已發表且常被下載的行動裝置 App，大致可分為旅遊、遊戲、娛樂、社交網路及生產力等五大類型，若考量行動裝置硬體設備特性有別一般的個人電腦，許多操作介面必須全新規劃與設計，此類型的行動裝置 App 較適合採取全新開發方式；若考量一般電腦上有許多 App 不僅發行多年且非常受歡迎，很適合移植至行動裝置上，此類型的行動裝置 App 較適合採取翻新開發方式；若已發行的行動裝置 App 下載率非常高且獲得極大好評，卻無法安置於其他平台的行動裝置上，此類型的行動裝置 App 較適合採取跨平台開發方式。因此，本文將行動裝置 App 的開發方式劃分成三種類型，說明如下：

- 全新開發方式: 從創意規劃、分析、設計、製作與測試等步驟，全部都由原點開始，進行 App 全新開發作業。
- 翻新開發方式: 挑選出一般電腦極受歡迎的 App 為依據，再配合行動裝置相關特性，進行調整、修訂與測試，達成 App 翻新作業。
- 跨平台開發方式: 從其他行動裝置平台上的 App，移植至目標的作業平台，達成 App 跨平台作業。

針對三種不同 App 開發方式的優劣，可以從開發時程、產品品質、資源花費與效果展現等四個評估項目進行探討，發現跨平台開發方式在開發時程、產品品質、資源花費等項目較具優勢，全新開發方式的效果展現項目最佳，評估彙整表如表三所示。不過，無論那一種 App 開發方式，優質可再用元件都是提昇行動裝置 App 開發效率與品質的重要關鍵。

表三：以四個評估項目探討三種 App 開發方式的優劣

評估項目 **App 開發方式**	**開發時程**	**產品品質**	**資源花費**	**效果展現**
全新開發	弱	弱	弱	強
翻新開發	尚可	尚可	尚可	弱
跨平台開發	強	強	強	尚可

3. 優質 App 元件應具備的條件

可再用元件是提昇行動裝置 App 開發效率與品質的關鍵,優質的可再用元件則是必要條件，本節探討優質可再用元件應具備的文件與關鍵特性。

3.1 文件化的可再用元件

以元件為基礎的 App 開發方法是提昇行動裝置 App 開發效率與品質的關鍵，而成為 App 元件的先決條件就是具有強化安置與調整能力的優質文件,App 元件的文件品質必須融入高度的正確性、完整性、一致性與可讀性等特性，至於文件應涵蓋的項目則是提昇行動裝置 App 可再用性的另一項關鍵，本研究探討且蒐集軟體品質與可再用性相關文獻(Lai, 2000; Schach, 2008)，歸納出文件化的 App 元件必須涵蓋元件細部設計規格書、原始程式碼、元件引用說明及完整的元件測試個案等重要項目，這些文件在開發作業上扮演的角色說明如下：

(1) 元件細部設計規格書：此份文件應該針對元件所定義的每個方法詳細描述其功能、介面、運作邏輯及資料結構等項目，且能夠配合元件間相互引用參考表快速協助完成下列的安置作業：

- 功能新增作業—依新增功能的內容加入新的方法，且補足新方法的功能、介面、運作邏輯及資料結構等項目，同時修改元件間相互引用參考表文件。
- 功能修改作業—依功能修改的內容調整對映的方法，且適度修改方法的功能、介面、運作邏輯及資料結構等項目，同時修改元件間相互引用參考表文件。
- 功能刪除作業—依刪除作業的內容刪除或調整對映的方法，且適度修改方法的功能、介面、運作邏輯及資料結構等項目，同時修改元件間相互引用參考表文件。
- 更正作業—依更正作業的內容調整對映的方法，且適度修改方法的功能、介面、運作邏輯及資料結構等項目，同時修改元件間相互引用參考表文件。

(2) 原始程式碼：此份文件應該依據元件細部設計規格書完成程式碼的撰寫，原始程式碼必須有統一的撰寫格式(Coding style)、輸出入參數的詳細定義與說明、變數的詳細定義與說明、關鍵邏輯與路徑的定義與說明等項目。當元件的細部設計規格書隨著開發作業進行修訂時，原始程式碼必須配合訂後的細部設計規格書及元件間相互引用參考表，快速完成原始程式碼的修改作業。

(3) 元件引用說明：此份文件應該依據元件細部設計規格書完成元件的引用說明，引用說明的內容必須有輸入參數與輸出參數的詳細規格、運作環境的限制、引用的方式與步驟等項目。當元件的細部設計規格書隨著維護作業進行修訂後，引用說明也必須配合修訂後的細部設計規格書進行修改。

(4) 完整的元件測試個案：此份文件應該依據元件細部設計規格書完成完整的元件測試個案規劃，完整的測試個案可以確保元件的一定品質，更可以配合開發作業進行快速的迴歸測試(Regression testing)，具體提昇行動裝置 App 的品質。完整的測試個案內容應包括每項個案的測試目的、測試資料、測試結果(或預期結果)、以及白箱測試 (White-box testing)策略的敘述、分支、條件與關鍵路徑等項目的涵蓋度記錄。當元件細部設計規格書隨著開發作業進行修訂後，完整的

測試個案也必須配合訂後的細部設計規格書進行新增、修改、刪除與更正等作業。

3.2 App 可再用元件的品質特性

App 元件要具備高度的可再用性，在規劃與設計的過程中，就必須融入文件基本品質、低複雜度、模組性、測試完整性及環境相依性等重要品質特性，以下即針對如何達成這些品質特性的條件進行說明：

(1) 文件基本品質：前一小節(3.1)中，除了說明可再用元件文件化包裝方式，且針對可再用元件的四項基本文件特質進行描述，構成可再用元件的文件基本品質就是正確性、完整性、一致性及可讀性，這些特質應在設計過程融入產品中，且透過審查機制進行改善。文件基本品質可以由正確性、完整性、一致性及可讀性等項目進行評量。

(2) 低複雜度：元件內部的邏輯設計與資料結構設計複雜度，也是影響可再用元件修改、維護作業另一項重要因素，複雜度過高的邏輯設計或資料結構設計不僅影響元件文件與程式的可讀性，更將大幅增加可再用元件修改、維護需投入的人力與時間。元件複雜度可以由邏輯複雜度(McCabe, 1976)、資料結構複雜度(Halstead, 1997)及條件敘述的深度進行評量。

(3) 模組性：軟體元件的規劃作業受到設計階段所融入的模組介面複雜度深入的牽制，過高的模組耦合度(Coupling) (如，控制耦合度及共用耦合度) (Fenton, 1991; Schach, 2008)、數量過多的參數以及複雜的參數類型都將增加模組單元隔離作業的困難。內聚力(Cohesion)是量測模組內部功能的相關性(Fenton, 1991; Schach, 2008)，高內聚力的軟體元件只處理單一功能或是處理多項高度相關且資料共用的功能，具備高內聚力的軟體元件於維護修改進行過程時，可以提供高度明確且具體維護能力。元件的模組性可以由耦合度及內聚力進行評量。

(4) 測試完整性：一般基層單元的測試方法都採取白箱測試策略，為確保可再用元件的測試完整性，應針對敘述、分支及重要路徑進行涵蓋度蒐集，越高的涵蓋度代表測試的越完整，也越能提高其功能品質的可信度。元件測試完整性可以由敘述、分支及重要路徑的涵蓋度進行評量。

(5) 環境相依性：行動裝置 App 受到硬體設備及運作環境等方面的衝擊，因此 App 元件與運作環境的相依關係對於元件可再用性有絕對的影響力，環境相依的項目包括作業系統、螢幕介面、記憶體及輸出入裝置等，元件環境相依性可以由作業系統、螢幕介面、記憶體及輸出入裝置的相容度進行評量。

2.2 節所描繪的行動裝置 App 三種類型開發方式，每種開發方式的 App 元件都受到五項品質特性的高度影響，其中後面四項與可再用元件品質特性關係密切，圖 1 即以圖示描述 App 四項關鍵特質與可再用元件品質特性之間的關聯圖。

4. App 可再用元件品質量測模式

量化機制是確認 App 元件是否具備可再用性的關鍵，本節將影響元件可再用性的品

質特性加以量化，且提出一套量度結合模式，用以確認 App 元件的可再用性。

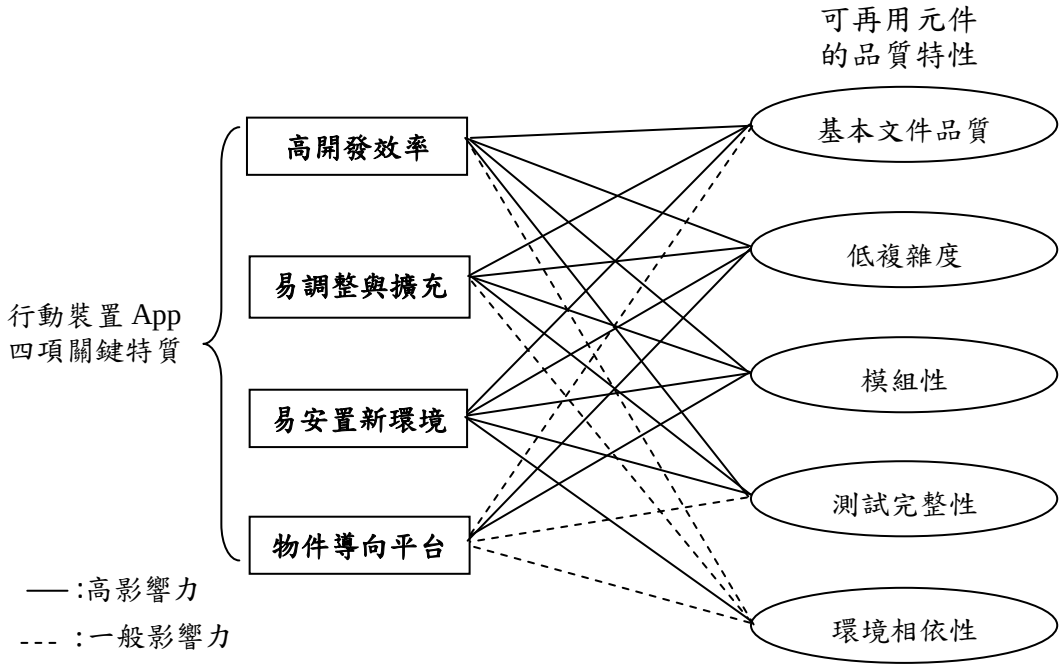

圖 1. App 四項關鍵特質與可再用元件品質特性之間的關聯

4.1 品質因子的蒐集與正規化

App 可再用元件可以從兩方面取得：(1)從已發行的行動裝置 App 中萃取出具有高度再用價值與高品質的元件，稱為偶發再用(Accidental reuse)；(2) 依 App 的應用領域，刻意規劃、設計與製作具有高度再用價值的優質元件，稱為系統再用(Systematic reuse) (Schach, 2008)。無論偶發方式萃取而得的 App 元件，或是系統方式開發而得的 App 元件，都必須融合文件基本品質、低複雜度、高模組性、測試完整性與環境相依性等品質特性，且通過一套嚴格的確認機制。結合文件基本品質、複雜度、模組性、測試完整性與環境相依性等查核表的正式檢視活動，可蒐集到 App 元件各項文件與可再用特性的關鍵品質因子。個別品質因子只能評量某項品質特性的單項特質，為了量測 App 元件關鍵特性品質，必須將個別品質因子做適當的結合(Deutsch and Willis, 1988)。在進行品質因子結合之前，首先必須將量化的品質因子進行正規化(Normalization)，因為影響不同項目品質特性的品質因子，代表不同層面的意義，展現的數據範圍也有很大的落差，正規化後的品質因子量化值將介於 0 到 1 之間，其中優質的因子正規化將趨近 1，劣質的品因子正規化將趨近 0。

4.2 App 元件品質量測模式

大部份的品質特性量測方式都是由一些基層的量度資料值結合而成(賴森堂，2002; Boehm, 1981；Conte et al., 1986)，正確性、完整性、一致性及可讀性等文件的基本品質特性可以透過審查作業的查核項目(checklist)蒐集相關量度資料值，其中每個查核項目的

評分就相當於一個基層的量度值，這些基層量度值依其重要等級又可設定不同的參數或權位值，接著再配合特定的結合公式就可計算出某項品質特性的量測值。利用分析工具所蒐集到的基層量度資料值，也可以依同樣的方式計算出某項品質特性的量測值，因為個別的量度資料值只能評量某項軟體品質特性的單項特質，為了量測整體的品質特性，必須將個別的量度值做適當的結合。量度結合的方式可以分為線性結合(linear combination)與非線性結合(nonlinear combination) (Boehm, 1981；Conte *et al.*, 1986)，考量實用性、修改彈性與公式簡單性，本文以線性結合方式來建立量度資料結合模式(Fairely, 1985；Lai, 2000)。在進行線性結合之前，首先必須將相關的量度值正規化(Normalization)，再配合事先設定的權位值及結合公式，可以將具高度相關性的基層量度值結合成元件品質特性量測值，最後，再將高階品質特性量測值結合成元件可再用品質量測指標，以下即為五個元件品質特性量測值的結合公式：

(1) 文件化的 App 元件包括細部設計規格、原始程式碼、測試個案及引用說明書等項文件，透過查核表的元件包裝檢視作業可以蒐集到各項文件的關鍵品質因子，結合關鍵品質因子可以產生文件完整性、一致性、正確性及可讀性等基層量度值，包裝元件的文件化品質量測值(CDQM)可由這些基層量度值所結合而成：

***CDQM*: Component Documentation Quality Measurement**

C_1MSC: Metric of Completeness of SC	*W_1: Weight of C_1MSC*
C_2MSC: Metric of Consistency of SC	*W_2: Weight of C_2MSC*
C_3MSC: Metric of Correctness of SC	*W_3: Weight of C_3MSC*
RMSC: Metric of Readability of SC	*W_4: Weight of RMSC*

$$CDQM = W_1 * C_1MSC + W_2 * C_2MSC + W_3 * C_3MSC + W_4 * RMSC \quad (1)$$

$$W_1 + W_2 + W_3 + W_4 = 1$$

(2) 元件複雜度量測值(*CCM*)可以由細部設計過程中，融入各個元件之邏輯結構量度(McCabe, 1976)、資料結構量度(Halstead, 1977)及程式巢狀深度等三個基層量度的量度值結合而成：

CCM*: Component Complexity Measurement*

LSM: Logic Structure Metric of SC	*W_{ls}: Weight of LSM*
DSM: Data Structure Metric of SC	*W_{ds}: Weight of DS*
NDM: Nesting Depth Metric of SC	*W_{nd}: Weight of NDM*

$$CCM = W_{ls} * LSM + W_{ds} * DSM + W_{nd} * NDM \quad (2)$$

$$W_{ls} + W_{ds} + W_{nd} = 1$$

(3) 元件模組性量測值*(CMM)*可以由軟體元件的耦合度與內聚力兩項影響模組性的關鍵量度值所結合而成：

***CMM*: Component Modularity Measurement**

CP: Coupling of SC	*W_{cp}: Weight of CP*
CH: Cohesion of SC	*W_{ch}: Weight of CH*

$$CMM = W_{cp} * CP + W_{ch} * CH \quad (3)$$

$$W_{cp} + W_{ch} = 1$$

(4) 元件測試完整性量測值*(CTCM)*可以由白箱測試方法所蒐集到的軟體元件程式碼之敘述、分支及關鍵路徑等涵蓋度結合而成：

***CTCM*: Component Test Completeness Measurement**

SCM: Statement Coverage Metrics of SC	W_{sc}*: Weight of SCM*
BCM: Branch Coverage Metrics of SC	W_{bc}*: Weight of BCM*
CPCM: Critical Path Coverage Metrics of SC	W_{cpc}: Weight of *CPCM*

$$CTCM = W_{sc} * SCM + W_{bc} * BCM + W_{cpc} * CPCM \qquad W_{sc} + W_{bc} + W_{cpc} = 1 \quad (4)$$

(5)元件環境相依性量測值*(CEDM)*可以由元件與行動裝置之作業系統、螢幕介面、記憶體及輸出入裝置的相依性或相容性所結合而成：

***CEDM*: Component Environment Dependence Measurement**

Operation System Dependence Degree	W_{os}*: Weight of OSDD*
Screen Interface Dependence Degree	W_{si}*: Weight of SIDD*
Memory Access Dependence Degree	W_{ma}*: Weight of MADD*
I/O Devices Dependence Degree	W_{id}*: Weight of IDDD*

$$CEDM = W_{os} * OSDD + W_{sidd} * SIDD + W_{ma} * MADD + W_{id} * IDDD$$

$$W_{os} + W_{sidd} + W_{ma} + W_{id} = 1 \quad (5)$$

最後，將元件文件品質量測值(CDQM)、元件複雜度量測值(*CCM*)、元件模組性量測值*(CMM*)、元件測試完整性量測值(*CTCM*)及元件環境相依性量測值*(CEDM)*加以結合，便可產生 App 元件可再用性品質量測指標，其公式如下：

RCQM: Reusable Component Quality Measurement

CDQM: Component Document Quality Measurement	W_{cdqm}*:Weight of CDQM*
CCM: Component Complexity Measurement	W_{ccm}*:Weight of CCM*
CMM: Component Modularity Measurement	W_{cmm}*:Weight of CMM*
CTCM: Component Test Completeness Measurement	W_{ctcm}*:Weight of CTCM*
CEDM: Component Env. Dependence Measurement	W_{cedm}*:Weight of CEDM*

$$RCQM = W_{cdqm} * CDQM + W_{ccm} * CCM + W_{cmm} * CMM + W_{ctcm} * CTCM + W_{cedm} * CEDM$$

$$W_{cdqm} + W_{ccm} + W_{cmm} + W_{ctcm} + W_{cedm} = 1 \quad (6)$$

透過五項結合公式一共產生五項 App 元件品質特性，再由這些品質特性結合成 App 元件可再用品質量測值，成為 App 元件可再用品質量測指標，引用 16 個基層量度及 6 個結合公式所產生的元件可再用品質量測指標即為 App 可再元件品質量測(App Reusable Component Quality Measurement; ***ARCQM***) 模式，其架構如圖 2 所示。本品質量測模式中，權位值是產生各項品質量測值的重要關鍵，累積多年行動裝置 App 發展經驗與知識的軟體工程專家是協助制定權位值的主力，透過軟體品質專家量測權位值設定問卷表，可以萃取出專家在權位值上的設定值。此外，配合各種狀況資料的蒐集則是調整權

位值的依據。

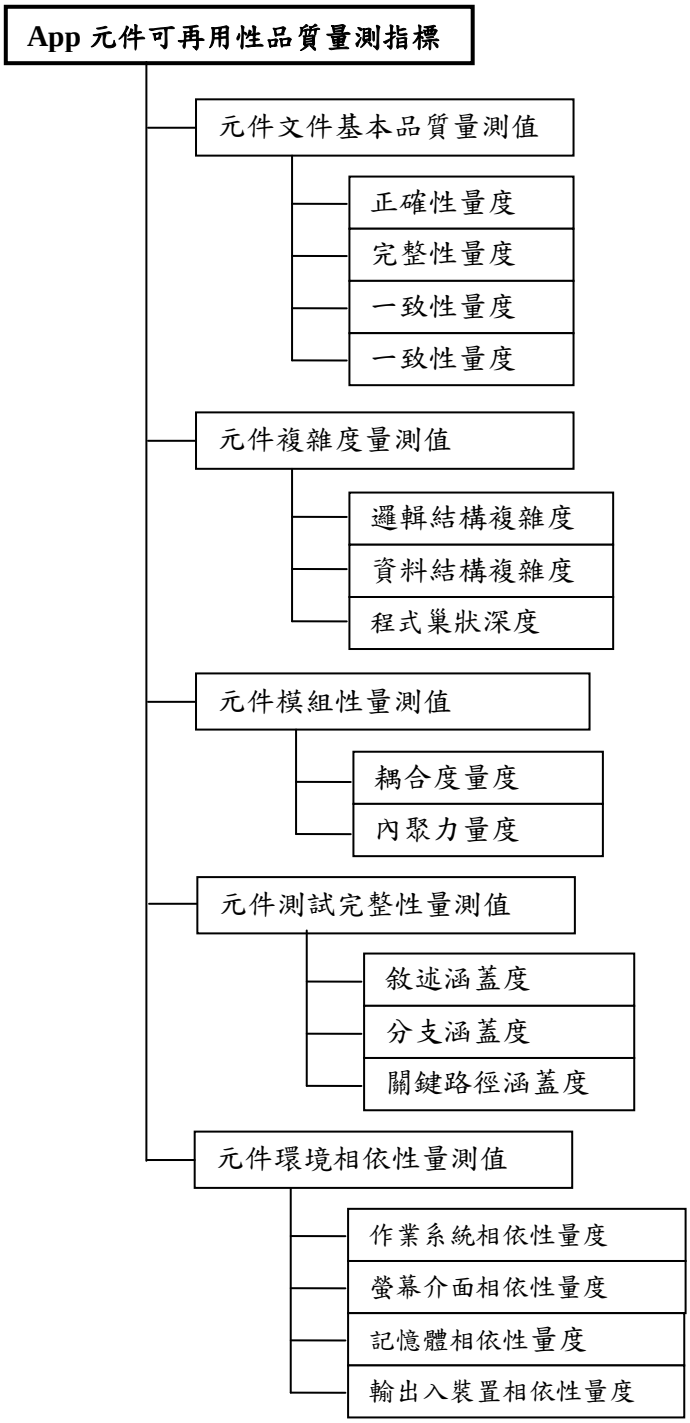

圖 2. App 元件可再用品質量測模式架構圖

5. 可再用元件品質補強與改善作業

品質量測值是一種相對的判斷指標(Indicator)，透過相關的軟體基層量度結合而成的品質量測值是判斷特定元件是否具備可再用性的依據。因此，當可再用品質量測值低於「品質基準值 (Quality Criteria)」時，就必須找出 App 元件的可再用品質缺失，從可再

用品質量測模式中的量度結合公式，可以判斷出劣質的基層量度，再從基層量度對映剖析出元件的設計與製作程序，進而找出元件設計與製程細項作業的缺失與不足，依據缺失與不足可以提出具體的補強與改善措施。以下即針對 App 元件可再用品質量測指標無法滿足「品質基準值」所提出的法則式改善方法：

<*Rule 1*> IF「App 元件可再用品質量測指標」不能滿足品質基準值
THEN 可以進一步分析出那幾項元件品質特性量測值太低造成「元件可再用品質量測指標」無法符合預期的可再用元件品質基準值，再針對過低的元件品質特性量測值所對映的基層量度進行深入剖析。

<*Rule 2*> IF「文件基本特性量測值」不能滿足品質基準值
THEN 可以進一步分析出那幾項基層量度太低造成「文件基本特性量測值」無法符合預期的品質基準值，再針對基層量度所對映的文件撰寫工作項目進行修訂與改善。

<*Rule 3*> IF「複雜度量測值」不能滿足品質基準值
THEN 可以進一步分析出那幾項基層量度太低造成「複雜度量測值」過高，無法符合預期的品質基準值，再針對基層量度所對映的細部設計工作項目進行修訂與改善。

<*Rule 4*> IF「模組性量測值」不能滿足品質基準值
THEN 可以進一步分析出那幾項基層量度太低造成「模組性量測值」過低，無法符合預期的品質基準值，再針對基層量度所對映的模組設計工作項目進行修訂與改善。

<*Rule 5*> IF「測試完整性量測值」不能滿足品質基準值
THEN 可以進一步分析出那幾項基層測試不夠完整造成「測試完整性量測值」過低，無法符合預期的品質基準值，再針對基層測試所對映的元件測試工作項目進行修訂與改善。

<*Rule 6*> IF「環境相依性量測值」不能滿足品質基準值
THEN 可以進一步分析出那幾項基層量度太低造成「環境相依性量測值」過低，無法符合預期的品質基準值，再針對基層量度所對映的元件與行動裝置之作業系統、螢幕介面、記憶體及輸出入裝置的相依度或相容度等項目進行修訂與改善。

可再用元件缺乏關鍵品質特性，對於行動裝置 App 的開發效率與品質將形成負面的影響，不過，透過法則改善方式，可以從量測模式及時找出品質的不足與缺失，讓元件可再用品質可以獲得適時補強與改善，進而提昇行動裝置 App 的開發效率與品質，且有效延續行動裝置 App 的生命期。

6. 結論

為了滿足行動裝置用戶的需求、加強互動性與快速反應時間，行動裝置的硬體技術日新月異，持續不斷的推陳出新，隨時都有新機型或新版本被推出，促使行動裝置 App 面臨強大的壓力與挑戰，必須配合裝置技術的成長，快速調整、修訂、擴充與強化其產

品的適應力與效率，才能持續擁有廣大的用戶。行動裝置 App 一般的開發方式可以劃分成三種類型：(1)全新開發方式; (2)翻新開發方式; (3)跨平台開發方式。無論採取那一類型開發方式，最主要的目的就是快速完成高品質的 App，軟體元件是組構行動裝置 App 的基層要件，優質的可再用元件不僅可以提昇行動裝置 App 的開發效率與關鍵品質，更可以有效延續 App 的生命期。本文針對行動裝置 App 有別於一般應用軟體的特殊性質進行探討，且深入探究與剖析行動裝置 App 開發作業與軟體元件的相互關係，以及元件可再用品質對於開發作業的影響力。結合文件化與品質特性的可再用元件是改善行動裝置 App 開發效率與品質的關鍵，為此，本文以文件化與品質特性為基礎，提出一套 App 可再用元件品質量測 (App Reusable Component Quality Measurement; *ARCQM*) 模式，*ARCQM* 模式採取多層次的線性結合模式，具有高度的調整彈性、簡單的結合公式、機動的擴充能力等優勢。透過量化的評量機制找出可再用元件品質不足與缺失，進而針對品質不足與缺失進行補強與改善，從確認與改善元件的可再用品質具體提昇行動裝置 App 的開發效率與關鍵品質，有效降低行動裝置 App 的開發成本且延長其生命期。本文提出的 *ARCQM* 模式具體達成下面三項優勢：

- 量度結合公式具簡單化與高度的調整彈性。
- 以量化的量測機制確認與改善 App 元件的可再用品質。
- 具體提昇行動裝置 App 的開發效率與關鍵品質，且以高可維護性延長 App 生命期。

誌謝

本論文接受 100 學年度國科會研究計畫(計畫編號：NSC100-2221-E-158-007)之補助。

參考文獻

1. 賴森堂，2002，「電子商務軟體品質量測模式」，企業管理學報，第 53 期，國立臺北大學企業管理學系，第 53-72 頁。
2. 產業新聞，「行動裝置夯，IMS：2011 年平板電腦和電子閱讀器成長率分別為 242% 和 146%」，http://www.smartmobix.com.tw/node/448。
3. Boehm, B. W., *Software Engineering Economics*, Prentice-Hall, New Jersey, 1981.
4. Butler, Margaret, “Android: Changing the Mobile Landscape,” *IEEE Pervasive Computing*, vol. 10, no. 1, pp. 4-7, 2011.
5. Conte, S. D., H. E. Dunsmore, and V. Y. Shen: Software Engineering Metrics and Models, Benjamin/Cummings, Menlo Park, 1986.
6. Deutsch, M. S. and R. R. Willis, Software Quality Engineering: A Total Technical and Management Approach, New Jersey: Prentice-Hall Inc. Pub. 1988.
7. Dogru, Ali H; Tanik, Murat M. “A process model for component-oriented software engineering,” IEEE Software, vol.20, no.2, :34-41, 2003.
8. Fairley, R., Software Engineering Concepts, McGraw-Hill, Inc., 1985.
9. Fenton, N.E., Software Metrics - A Rigorous Approach, Chapman & Hall, 1991.
10. Gavalas, D. and D. Economou, "Development Platforms for Mobile Applications: Status

and Trends," *IEEE Software*, vol. 28, no. 1, pp. 77-86, 2011.

11. Halstead, M. H., Elements of Software Science, North-Holland, New York, 1977.
12. Hense, Andreas, Florian Quadt, Matthias Römer, "Towards a Mobile Workbench for Researchers," 2009 Fifth IEEE International Conference on e-Science, pp.126-131, 2009.
13. Huntley, C. L., "Onshore Mobile App Development: Successes and Challenges," *Computer*, vol. 44, no. 9, pp. 102-104, Sept. 2011.
14. Lai, S.T., "A Quality Measurement Model for Software Maintenance," Proceeding of the Second World Congress on Software Quality (2WCSQ), Sept. Japan, 2000.
15. McCabe, T. J., "A Complexity Measure," *IEEE Trans. Software Eng.*, Vol 2, No 4, 1976, pp.308-320.
16. Pressman, R. S., *Software Engineering: A Practitioner's Approach*, McGraw-Hill, New York, 2008.
17. Schach, S. R., *Object-Oriented Software Engineering*, McGraw-Hill Companies, 2008.
18. Szyperski, C., *Component Software*, Addison-Wesley, 1998.
19. Vitharana, P., "Risks and challenges of component-based software development," CACM, Vol. 46 No. 8, August 2003.

Applying Reusable Components to Improve Development Productivity and Quality of Mobile Apps

Sen-Tarng Lai
Department of Information Technology and Management, Shih Chien University
stlai@mail.usc.edu.tw

Abstract

In the E Generation of internet fever, mobile devices become a personal portable object. Portable, highly interactive, multiple communication styles and processing mechanism are major advantages of mobile devices. Mobile Applications (Mobile Apps) is an important product of internet fever. For satisfying the mobile user requirements, enhancing interaction and reducing response time, the new develop technology of Mobile Apps are proposed continuous. App developer should enhance process capability for growing environment and technology, and rapid development new App to get market opportunities. How to effectively shorten the App development time and keep high quality has become a subject worthy of further exploration. App component is a primitive element of Mobile App. In order to effective increase App development efficiency and quality, planning a high quality reusable component assurance mechanism is necessary. In this paper, based on documentation and quality characteristics, a model of *App Reusable Component Quality Measurement* (*ARCQM*) is proposed. Applying the quantity mechanism, the critical quality of App components can be assured, and with component classification and retrieval functions. Mobile App development efficiency and quality can be increased.

Keywords: reusable components, quality characteristics, Mobile Apps, *ARCQM*