國立交通大學

資訊科學與工程研究所

碩 士 論 文

XMMI v3.0 - 行動裝置介面客製化系統與在個人電腦平台上之模擬器

研 究 生：賴宗彥

指導教授：李嘉晃 教授

中 華 民 國 九 十 六 年 六 月

XMMI v3.0－行動裝置介面客製化系統與在個人電腦平台上之模擬器

XMMI v3.0 – Customize Interface System for mobile device and the Simulator on PC platform

研 究 生：賴宗彥　　Student：Zong-Yan Lai

指導教授：李嘉晃　　Advisor：Chia-Hoang Lee

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

XMMI v3.0 – 行動裝置介面客製化系統與在個人電腦平台上之模擬器

研 究 生 ：賴 宗 彥　　指 導 教 授 ：李 嘉 晃　教 授

國立交通大學電機資訊學院　資訊科學與工程所碩士班

摘要

隨著行動裝置的普遍性與應用廣泛地發展，如何讓使用者能夠定義介面，即軟體與硬體的溝通橋梁，將會是一門課題。利用 XMMI v3.0，使用者可以利用編輯器自行編改行動裝置介面的視覺性與操作，並且即時模擬。

第三代可延伸人機介面系統(XMMI v3.0)包含了 5 個元素：行動裝置介面架構(MIF, Mobile Interface Framework)，行動裝置介面描述語言(MIDL, Mobile Interface Description Language)，行動裝置介面編輯系統(MIES, Mobile Interface Edit System)，行動裝置介面模擬器(MIS, Mobile Interface Simulator)，行動模擬創造器(MSC, Mobile Simulator Creator)。本篇論文將著重於 MIF，MIDL，MIS，MSC 上的探討。

XMMI v3.0 – Customize Interface System for mobile device and the Simulator on PC platform

Student：Zong-Yan Lai Advisor：Chia-Hoang Lee

Department of Computer and Information Science
National Chiao Tung University

Abstract

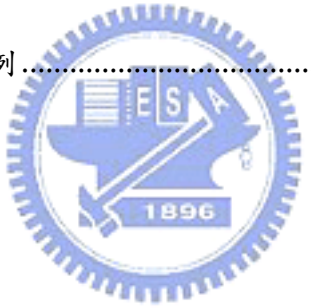
With the generality and extensive development of mobile device, how to let users define their own interface which communicate with hardware and software will be an important topic. By using XMMI v3.0, users can utilize editor to modify the vision and operation of interface, then simulate immediately.

XMMI v3.0 contains five components: Mobile Interface Framework(MIF), Mobile Interface Description Language(MIDL), Mobile Interface Simulator(MIS), Mobile Interface Edit System (MIES), Mobile Simulator Creator (MSC). This essay will notice the treat of MIF, MIDL, MIS and MSC.

目錄

圖目錄

表目錄



第一章　緒論

1．1　概述

近年來，隨著行動通訊裝置的普及化與方便性，幾乎是人手一機甚至是多機，並由於行動裝置具備了體積小、穩定性強，方便攜帶的特性，行動通訊系統的應用發展愈來愈廣泛，從最基本的通話、MMS 多媒體訊息，到延伸性的內建變焦相機、GPRS 上網、收發 Mail 等等不勝枚舉，不但證明行動裝置在現今日常生活中已成了一個不可或缺的工具，也意味著行動裝置在未來發展的潛力。

內建功能	PDA	有
	文件閱讀	Word, PowerPoint, Excel, PDF
	輸入法	注音, 智慧型中英文輸入, T9智慧型輸入, 手寫輸入, 英文輸入
	JAVA	有
	實用工具	計算機, 單位換算, 世界時鐘, 鬧鈴, 日曆, 行事曆, 待辦事項
	錄　音	有
	聲控指令	有

圖 1：Sony Ericsson W950i　內建功能應用

而在這個求新求變的時代，消費性電子產品不僅講究效能快速、功能多樣化，更隨著時尚潮流講求造型新穎、設計獨到的視覺外觀。這個事實在在說明了過去一樣產品研發的重心著重於程式開發者的角度，重視的是技術與效能層面，而現今已轉換到使用者的角度，講求以人為主的設計觀念；若一款科技產品能符合大多數使用者的需求，它就是款成功的產品！

對於現今的行動裝置，個人化意識也正在蔓延中，自由置換面板、來電答鈴、自創鈴聲、捷徑設定等等，攜帶著一支具有強烈個人風格的手機，也同時展現出使用者自身的品味時尚。

Nokia 7610　　LG KG 90　　MOTO Q q9　　Samsung SGH-X838

圖 2：市面上各家廠商造型華麗的手機

1．2　研究動機

正由於目前行動裝置隨著時尚流行而往個人化發展，舉例來說：不管是硬體形式上，華麗的外觀造型與配備，或是軟體形式上，實用的應用程式，都可看出廠商的開發導向不只講求產品本身的價值，更講求產品對使用者的價值。

在硬體與軟體的快速發展下，相對地，作為硬體與軟體的溝通橋樑：介面，其開發就顯得並不是那麼突出；然而在以人為主的概念下，一款手機是否便利與易操縱更影響使用者對此款裝置的喜好，如 Apple 公司推出的產品 iPhone 運用了運動加速器技術來偵測手機是直立還是橫放，藉此判斷螢幕的顯示位置，讓使用者可依照網頁、視訊與照片的需求隨時做轉換，充分展現了 user friendly；而

像目前智慧型手機上的更改捷徑、佈景主題，自製首頁等功能，介面的設計愈來愈多樣化，卻因為使用者的操作習慣不同，不同的使用者對同樣的介面有著不同的評價，因此，如何將介面符合大眾，達到客製化的功能，將是一門重要的課題。此外在以往的開發行動裝置應用程式上，介面的畫面內容與操縱方式均由應用程式開發人員自行設計開發，這使得每當一款新產品推出時，而須轉移模組時，修改資料都需重新編譯連結，花費多餘的時間精力；使用者亦只能接受廠商制式化的介面，無法更動，缺乏彈性。

1．3　研究目標

為了使介面能因應各式不同的使用者，並且能讓程式開發人員更專注於應用程式的功能性上，亦減少重覆開發的成本，也讓使用者自行定義介面的視覺化與操作的便利性，在以達成此目標的驅使之下，設計了第三代可延伸人機介面系統 XMMI v3.0。

XMMI v3.0 包含了 5 個元素：行動裝置介面架構(MIF, Mobile Interface Framework)，行動裝置介面描述語言(MIDL, Mobile Interface Description Language)，行動裝置介面編輯系統(MIES, Mobile Interface Edit System)，行動裝置介面模擬器(MIS, Mobile Interface Simulator)，行動模擬創造器(MSC, Mobile Simulator Creator)。本篇論文將著重於 MIF，MIDL，MIS，MSC 上的探討。

在 XMMI v3.0 的系統架構下，使用者可在個人電腦上方便的藉由介面描述語言的設定，編輯屬於自己的介面，也可立即作出模擬測試，更進一步地，在網路上與其他使用者分享自行創造的介面；同時廠商可以提供許多不同風格的介面描述檔，讓使用者自行下載，變更操作介面，使得介面能作到客製化，如同每個

人的手機都是獨特為自己設計的。

1・4 論文架構

本論文其餘部份，分為：第二章－文獻探討將介紹一些背景知識與相關研究，第三章－行動裝置介面客製化系統 XMMI v3.0 架構與實作，第四章－MIS 系統設計與實作，第五章－結論及未來工作。

第二章　文獻探討

２・１　背景知識

以下介紹一些 XMMI v3.0 系統相關的背景知識：

２・１・１　可延伸標記語言ＸＭＬ

XML 是「可延伸標記語言」（eXtensible Markup Language）的縮寫，規格是由「全球資訊網標準制定組織」（W3C）制定，並於 1998 年 2 月成為推薦規格。現今已有許多家廠商採用，且視為關鍵性技術。例如：Adobe，IBM，微軟，Netscape，Oracle，Sun 等。

通常對一份 XML 資料，會有以下幾點的要求：一、唯一的根節點（將整份 XML 文件的結構慢慢的收起來看，會發現 XML 文件本身，是以樹狀結構做展開，所以不妨把他想像成一顆樹，一棵樹可以有相當多的枝節往外生長，但是他只會有一個根，所以說，依循樹狀結構所建立的文件，一定都會擁有唯一的根節點）。二、所有的起始標籤一定相對有著結束標籤（因為 XML 是將展現層完全撥離的純粹資料層，所以對每一個資料實體而言，我們都應該去定義他的範圍，在 XML 中，所有的實體都是以標籤來定義的，當我們的個體中擁有資料的時候，我們就必須以起始標籤跟結束標籤將資料包含起來）。三、一致性（XML 是一種結構極為嚴謹的資料描述語言，整個文件結構都只為了一個目的：描述資料，所以對 XML 來說，資料個體的唯一性是相當重要的，XML 在尋找資料的時候，他必須以我們所給定的實體標籤作為判斷的依據，才能夠對應到正確的欄位）。四、正確的巢狀標籤（在 XML 的巢狀結構中，外圍父巢層與子巢層不能有重疊的現象，子巢層必須由父巢層完全包覆，如此一來，我們才能夠準確的使用樹狀結構來表示、傳遞資料。）五、所有的屬性值必須以引號包覆（在 XML 中，我

們會使用屬性來定義每個資料實體，當然，這不是必然的，給定屬性的目的主要是為了區隔資料的內容，如同我們在選一本書的價格時，可能有美金、台幣跟日幣計價單位，這時候我們就得給定一個計價單位的屬性，在 XML 中使用屬性的方法是於資料實體的起始標籤中加上屬性敘述，因為屬性的目的大多為資料定義，所以我們用字串型態來表示。

在我們的系統中，對於 MIDL，我們採用了 XML 的技術，由於 XML 眾多的優點與編譯器，使得以 MIDL 為基礎的 MIS 與 MIES 能夠便利且快速的解讀檔案。

２・１・２　人機介面ＭＶＣ架構

MVC 是用於表示一種軟體架構模式。它把軟體系統分為三個基本部分：模型（Model），視圖（View）和程式控制（Controller）。MVC 主要的目的是實現一種動態的程式設計，使後續對程式的修改和擴展簡化，並且使程式某一部分的重覆利用成為可能。除此之外，此模式通過對複雜度的簡化使程式結構更加直觀。MVC 應用程式是由三個部分組成。事件(Event)導致 Controller 改變 Model 或 View，或者同時改變兩者。只要 Controller 改變了 Models 的資料或者屬性，所有依賴的 View 都會自動更新。同樣地，只要 Controller 改變了 View，View 會從潛在的 Model 中獲取資料來刷新自己。

MVC 設計模式是一個很好創建軟件的途徑，在我們的系統中亦使用了 MVC 模式，如我們將內容和顯示互相分離，並且利用元件本身的動態繫結性質來達到所見即所得，同時，運用 MVC 帶來的好處即是讓邏輯層與介面層分開實作，藉由 Controller 與 Model 和 View 的互通，分層的效果能讓各階段的開發者致力於自身的專業方向。在第三章中，我們將會詳細的介紹 MVC 在系統的架構。

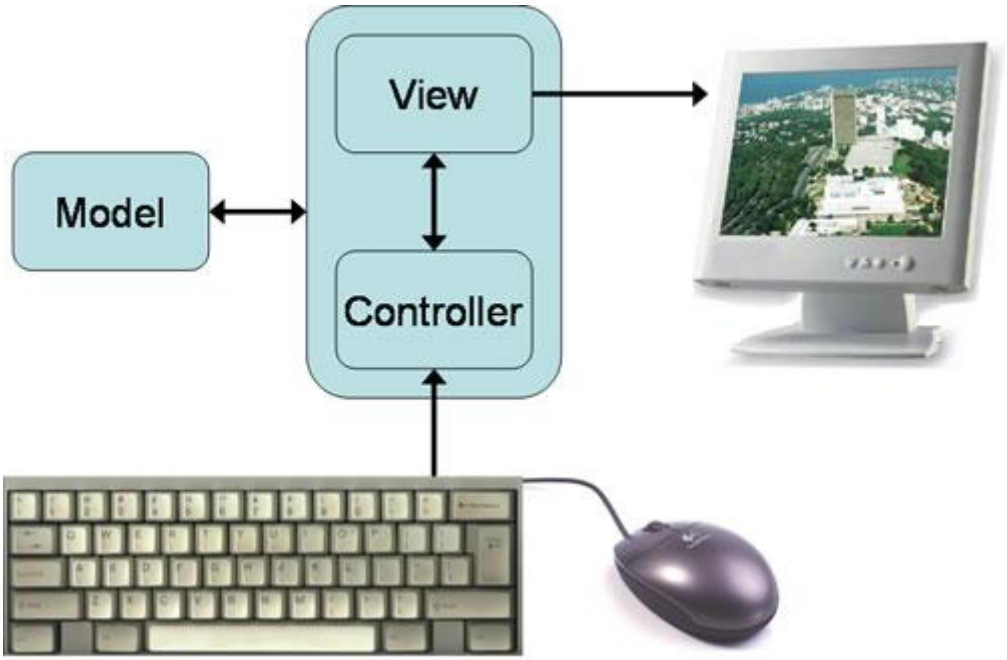


圖 3：MVC 架構示意圖

2．1．3 Microsoft Windows CE 5.0

Microsoft Windows CE（簡稱 WinCE）是 Microsoft 公司專為嵌入式系統（Embedded System）所打造的 32 位元硬式即時作業系統（Hard Real-Time OS），其核心的運作方式則是繼承 WinNT 的技術，可以適用在智慧型、具連接性、與精巧的裝置上。在作業系統核心部分，Microsoft 從 5.0 開始，擴大核心原始碼授權，將 Windows CE 分享原始碼（Windows CE Shared Source）擴充至兩百五十萬行以上的程式碼。讓授權對象可以自行更改 Windows CE 5.0 分享原始碼（shared source code），使用於商業用途與產品開發，同時保有其衍生程式碼的所有權，無須與微軟、合作夥伴或競爭者共享。

至於一般使用 SDK 的應用程式開發部分，在 v4.2 時代，Microsoft 公司已經把.NET Compact Framework 移植到行動裝置上面，而在新的 5.0 裡面，開發應用程式則是由新版的 Visual Studio 2005 全部包辦，在新增專案的時候，就包含

Windows CE 的應用程式框架，並且可以在模擬器上面看到程式執行的結果。

Today 畫面　　Email 郵件管理畫面　　Pocket Excel 畫面

圖 4：Microsoft Windows CE 畫面

2・2 相關研究

以下介紹一些 XMMI v3.0 系統相關的研究探討：

2・2・1 自訂佈景主題

在市面上的智慧型手機（SmartPhone），大部份都有自訂佈景主題的功能，由於設定選項十分豐富，可以在打造自己獨特的手機介面的同時，增加很多方便的功能選項。而其桌面主題和普通手機的主題有很大的區別，普通手機的主題主要是利用解壓特有的壓縮包來更換手機操作介面的外觀和來電鈴聲，壓縮包內包含的主要是：圖片（背景，九宮，選擇欄等）和音頻文件（開關機鈴聲，來電鈴聲）。而 SmartPhone 的桌面主題則是把重點放在了桌面的感觀美化和功能改進上。他的主題文件裡除了一些必要的圖片和 XML 文件外，還擁有許多具有特殊作用的插件，這些插件改善了 SmartPhone 的桌面便捷性，透過它們，可以使 SmartPhone 更加獨一無二。但也由於，SmartPhone 手機的桌面主題構造十分複

雜，每一個 SmartPhone 主題都擁有數量不等且功能不一的主題插件，在插件不全的情況下，多數主題都不能正常使用，只徒增麻煩與浪費資源。

LCD 插件

模擬時鐘插件

媒體播放器插件

圖 5：自訂佈景主題插件

2．2．2　視覺化元件

一般市面上軟體工具普遍來說，對於使用者介面開發，均作到了所見即所得 (What You See Is What You Get)，如軟體開發以 Microsoft 公司的 Visual Studio 及 Borland 公司的 Borland C++ Builder 為主要的工具；由於此兩種工具並提供了多種 UI 元件，如：狀態列、標籤、下拉式選單、捲軸等，讓使用者有多樣的選擇。又如 Adobe 公司旗下的 Flash 軟體，在使用者介面編輯區中，設計狀態所編排得到的版面，執行時，也將呈現當初設計的版面。這樣的優點，讓使用者可以在設計時，大大降低開發時間，不必總是在錯誤中修改(Try Error)。

因為程式人員必須負責實作使用者介面，而介面規格又是出於美工人員之手，導致二者之間相依度高、互動性頻繁。如此將大幅增加軟體開發時間成本。在本系統中，我們即著重在這樣的問題，如何在快速的時間內排版使用者介面，

並方便的將軟體功能嵌入在介面之外，如此開發中才能令使用者介面設計與軟體功能分離，而大幅增加開發效率。

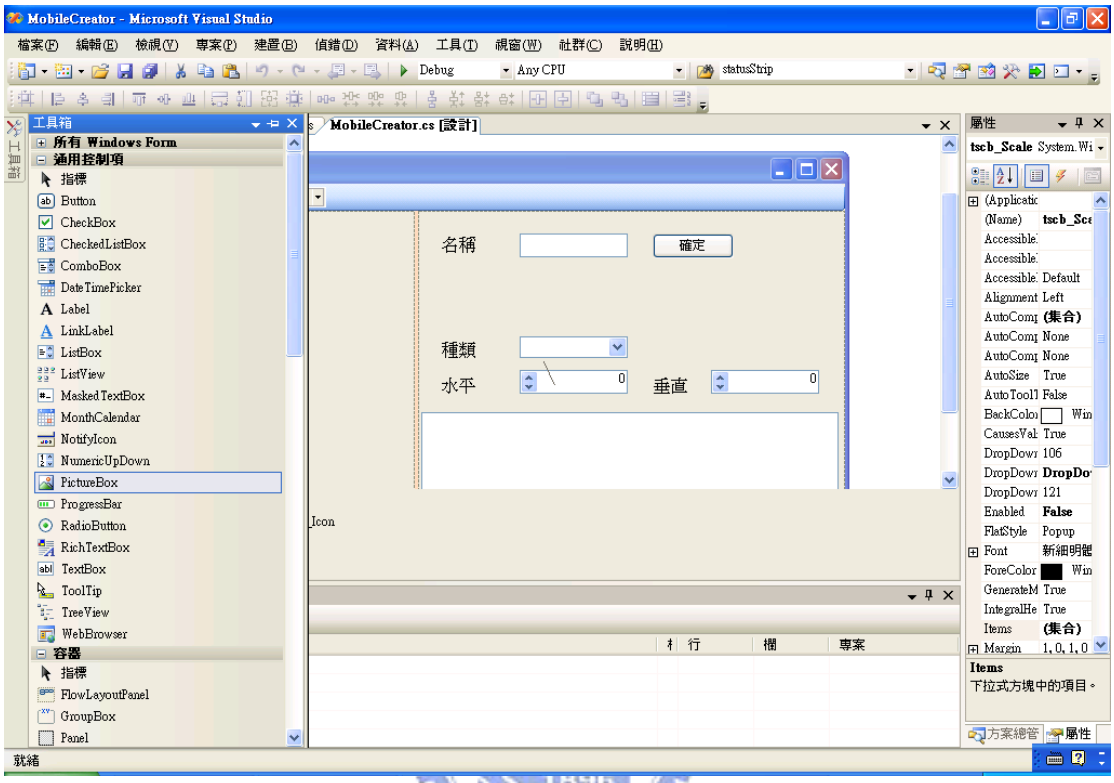


圖 6：VS2005 視覺化元件開發介面

2．2．3 第二代可延伸人機介面系統(XMMI v2.0)

第二代可延伸人機介面系統(Extensible Man Machine Interface System version 2.0, XMMI v2.0)，可讓使用者自行定義行動裝置介面狀態的畫面內容及操作方式。系統包含了行動電話人機介面系統(Mobile Phone Man-Machine Interface System, MMIS)、行動電話人機介面描述語言(Mobile Phone Man-Machine Interface Description Language, MMDL)與編輯系統。XMMI v2.0 即是 XMMI v3.0 的前身版本。

```
 1 <mmi>
 2    <resource type="color" x="0" y="0" width="101" height="80">ffffff</resource>
 3    <event name="keyup" parameter="65">
 4       <link>
 5          <variable name="next"/>
 6       </link>
 7    </event>
 8    <event name="keyup" parameter="68">
 9       <link>10000_menu9.xml</link>
10    </event>
11    <control type="list">
12       <variable name="lrange">5</variable>
13       <variable name="llink">11100_callcontact.xml,11200_addcontact.xml, ...略</variable>
14       <variable name="ltext">1Call contact,2Add contact, ...略</variable>
15       <resource type="text" x="0" y="0" width="101" height="16">
16          <variable name="ltext"/>
17       </resource>
18       <resource type="text" x="0" y="16" width="101" height="16">
19          <variable name="ltext"/>
20       </resource>
21       <resource type="text" x="0" y="32" width="101" height="16">
22          <variable name="ltext"/>
23       </resource>
24       <resource type="text" x="0" y="48" width="101" height="16">
25          <variable name="ltext"/>
26       </resource>
27       <resource type="text" x="0" y="64" width="101" height="16">
28          <variable name="ltext"/>
29       </resource>
30       <event name="focus" parameter="0">
31          <resource type="color" x="0" y="0" width="101" height="16">ff00</resource>
32       </event>
33    </control>
34 </mmi>
```

表 1：XMMI v2.0 中 MMDL 範例

Line 1, 34：MMDL 根元素，mmi 之開頭與結尾，將內容標示為一人機介面狀態。
Line 2：將純白色塊放置在(0,0)的位置，顯示視窗之大小為 101x80。
Line 3~7：當按鍵事件發生且鍵值為 65 時，轉移至系統變數 next 值之狀態。
Line 8~10：當按鍵事件發生且鍵值為 68 時，轉移至 10000_menu9.xml 之狀態。
Line 11：此狀態包含 list(項目清單)元件。
Line 12：將 list 元件之 lrange 變數設為 5。
Line 13：將 list 元件之 llink 變數設為 11100_callcontact.xml, ...略。
Line 14：將 list 元件之 ltext 變數設為 1Call contact, ...略。
Line 15 ~ 29：分別指定五個資源之位置及寬高資訊，並顯示 ltext 之資訊。
Line 30 ~ 32：當元件之焦點事件發生時，顯示綠色色塊。

XMMI v2.0 利用變數作為與介面狀態的溝通，對於一個介面狀態，如上述的清單畫面(List)，則存在有其對應的變數(llink, ltext, lrange)等，設定不同型別的數值；待機畫面(Idle)與選單畫面(Menu)亦有自身對應變數，如此一來，在使用者的角度，在編輯器上操作時，無從得知介面中對應的變數代表什麼樣的意義，更進一步地，傳入的型態、傳入的長度等也無從而知。

第三章　行動裝置介面客製化系統 XMMI v3.0 架構與實作

3・1　第三代可延伸人機介面系統概述

3・1・1　前言

人機介面(Man Machine Interface, MMI)不管在任何設備上，都佔有極重的地位，可以說是每位使用者接觸科技產品的第一道門。MMI 設計的目標是希望讓每位使用的人能夠利用系統作為完成某一項工作的輔助工具，我們應該用使用者的邏輯來思考操作方法，而非以設計者的邏輯來思考。舉例來說，當一個檔案拷貝到另一個相同檔名的檔案時，由系統的觀點來看我們知道原來的資料就不見了，但是由使用者的觀點來看，他一定覺得很不可思議，怎麼會把資料蓋掉了呢? 不是只是重疊在一起而已嗎？

因此，對一個操作行動裝置的使用者來說，介面大大地影響著使用者運用工具的效率，例如一個選單上的圖示，若沒有文字說明，圖示的表達力有限，使用者將花費些許時間去熟悉這個介面，而非自己所能認知的介面，在這樣的情況下，對於多種不同產品的介面，使用者須記取不同介面下的操作方式，對於使用者來說亦是種無形的負擔。

在上述的前提之下，我們發展了 XMMI v3.0 系統，有別於以前行動裝置死板且制式化的介面，系統能讓使用者編輯屬於自己的介面，使用者不需要知道手機應用程式底層的處理運作，卻可以因為元件的彈性開發，而受惠於更改元件視覺性與元件操作性。

3・1・2　XMMI v3.0 系統

XMMI v3.0 系統包含了行動裝置介面架構(MIF, Mobile Interface Framework)，行動裝置介面描述語言(MIDL, Mobile Interface Description Language)，行動裝置介面編輯系統(MIES, Mobile Interface Edit System)，行動裝置介面模擬器(MIS, Mobile Interface Simulator)，行動模擬創造器(MSC, Mobile Simulator Creator)，架構如圖 7 所示。

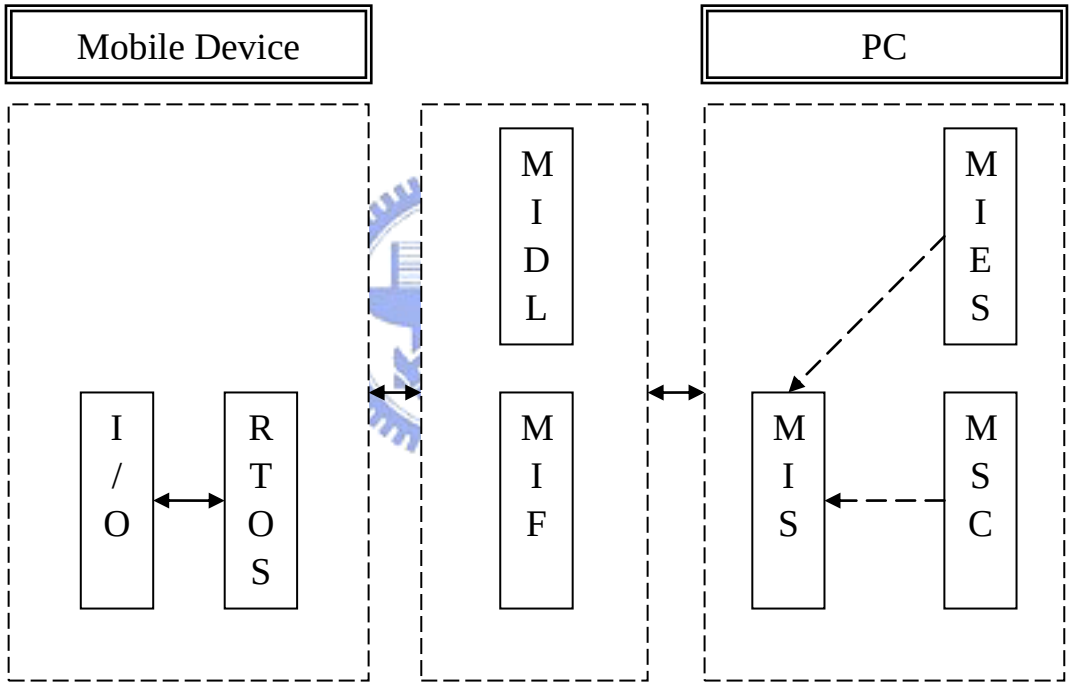


圖 7：XMMI v3.0 系統架構圖

行動裝置介面元件(MIF)負責整個介面的重組及各種元件，包括將一個介面所有元件呈現出來，同時，接受 TouchPad 或鍵盤的輸入，產生元件對應事件。行動裝置介面描述語言(MIDL)描述著一個介面狀態，包含元件的視覺化與操作性。行動裝置介面編輯器(MIES)是用來編寫介面描述語言，讓使用者方便編輯行

動裝置介面。行動裝置介面模擬器(MIS)負責在個人電腦平台即時模擬出在行動裝置上的介面。行動模擬裝置創造器(MSC)可以創造出不同的模擬器，以符合各式不同款式手機的規格。

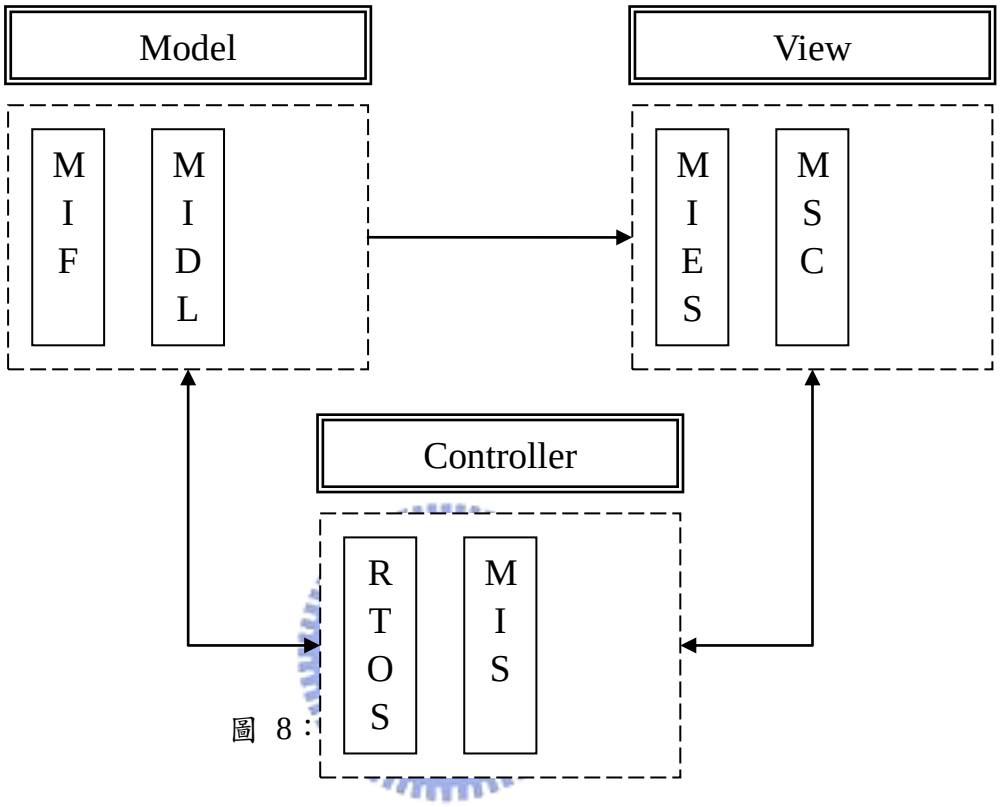


圖 8：

本系統的編輯系統與模擬器、創造器是建構於 Microsoft Windows 作業系統，開發環境為 Microsoft Visual Studio 2005。將本系統轉換到 MVC 模式中，如圖 8 所示，Model 部份代表著開發者的角度，在本系統即程式設計者與使用者，程式設計者可以自行利用 MIF 中的元件開發創建一個應用程式，包括元件的一些相關事件；而使用者可以在編輯器上編輯一個程式設計者開發的應用程式，並且對其內部的元件作修改，由於考慮到使用者是沒有程式基礎的背景，修改的權限大致上都只屬於視覺性與操縱介面的更改，不讓使用者動到邏輯上這部份，一方面，能保證系統的穩定性；另一方面，使用者無須理會應用程式內部如何運行，而只須專注在介面的設計上。View 部份則架構在 Model 上，當開發族群（程式設計者與使用者）修改了內部元件，系統在視覺上，會立即回饋給使用者，例如使用者在利用 MIES 新增了一個內建元件，使用者可以立即看到一個元件的型

態，並可修改其屬性。Controller 部份主要是控制系統的流程與輸出輸入，負責 Model 與 View 的溝通介面，MIS 即是，其負責動態呼叫介面，並將其顯示出來，也接受使用者輸入而產生對應的動作。

3・1・3　系統演進

在第二代可延伸人機介面系統(Extensible Man Machine Interface System version 2.0, XMMI v2.0)中，考慮到資源內容分為靜態與動態兩種，而加入了變數元素，雖然對程式設計者增加了功能上的彈性，但對於使用者而言，對於變數所代表的涵意卻一無所知，加上變數型別繁多，在使用者不知道變數意義的情況下，根本無從知道在編輯器上對於變數的輸入是如何的型態，而導致不直觀，無法達到「所見即所得」。加上資源種類少（色塊、影像、文字）與多餘的 link 標籤，對不熟悉系統的使用者而言，卻是種負擔。

為了改進 XMMI v2.0 的缺點，我們建構了元件(Component)這種元素。在以物件導向的觀念底下，我們試想手機上的一個應用程式即是一種物件，不同的應用程式可能會引用相同的元件，將元件如何視覺化呈現並讓使用者修改即是我們著重的方向。以下是另外第三代與第二代系統的比較，詳細的差異與比較會在後面的章節中作說明。

	XMMI v3.0	XMMI v2.0
系統結構	以元件為基礎，可使用元件來開發應用程式，對應用程式開發者來說，不用顧慮其外觀問題，而能專心於功能性的發展上；對使用者來說，可以不須知道其功能性，而對應用程式的外觀加以調整。	以應用程式與元件為基礎（同層），利用變數去存取資源，使得架構上不好理解，容易混淆。
編輯器	建立在以建構元件與功能性的應用程式的觀念下，使用者開發的編輯器將有視覺化以利方便使用（以拖曳、選單等方式）。	無法相當視覺化的呈現在使用者介面上，使用者需要知道元件定義的背景後才能操縱上手。
應用程式	對於一個應用程式的開發者來說，只須將所要的元件定義其初始值，而其功能性可自由發展，雖然功能性與外觀均在應用程式端實作，但特別的是，其外觀可以有新設定，即應用程式中的元件編排、大小、樣式均操縱在使用者手中，達到功能性與外觀分離的效果。	由於應用程式與元件相衝突，無法共用，會使應用程式須定義自身的介面，而造成資源的浪費。
模擬器多樣化	可設計不同款式手機（新創手機）的模擬手機外觀，讓使用者使用自己喜愛的手機來模擬系統。	舊系統無此功能。
元件與事件	由於一個動作的產生通常配合著一個元件視覺的變化，所以將事件內嵌於元件中有兩個優點：對程式開發者處理事件，可以簡便的處理而不混淆。對於使用者編輯介面，可以不須將元件與事件分開編輯。	須將元件與事件分開處理，無形中增加結構的複雜而造成應用程式開發者與使用者操作上的困難。

表 2：XMMI v2.0 與 v3.0 的比較

３・２　行動裝置介面架構

３・２・１　架構設計與系統流程

考慮使用者並沒有程式設計的相關背景，卻也能夠方便且準確地操作編輯器，並讓應用程式開發者專注致心於功能性上，而無須顧及外觀；一個資源通常有其外觀與功能，若我們將這個資源的外觀開放給使用者修改，而功能性則是交給程式開發者設計，如此一來元件的設計流程也由此而生，如圖 9 所示。

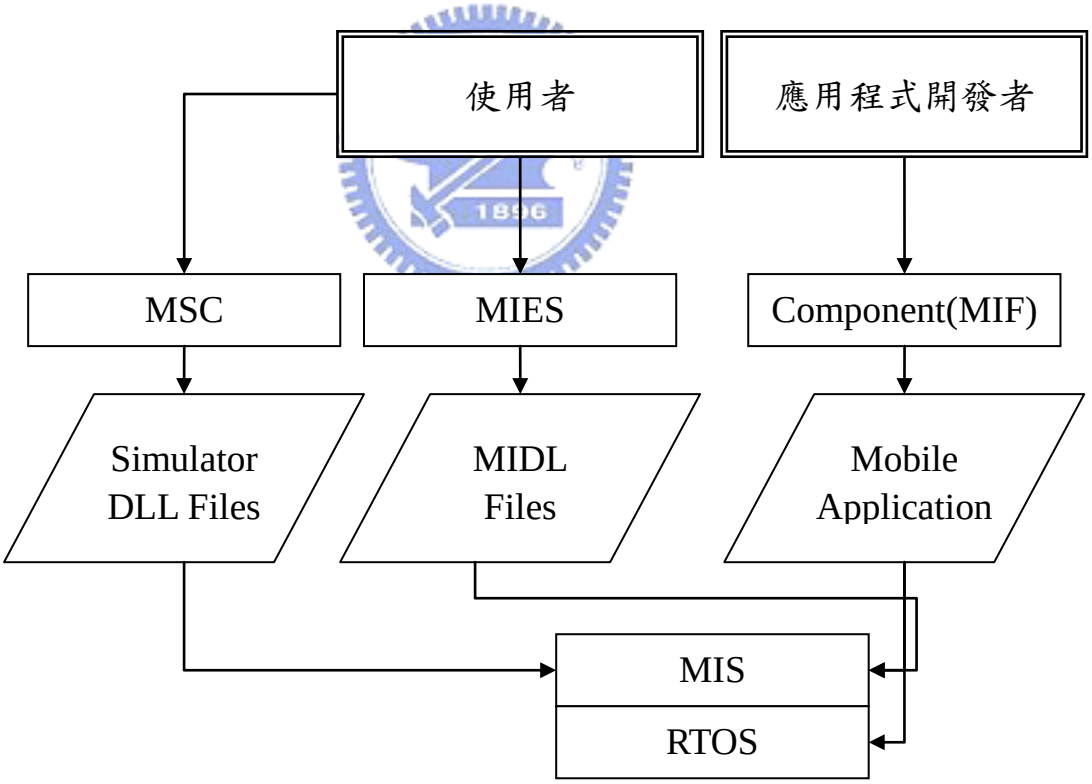


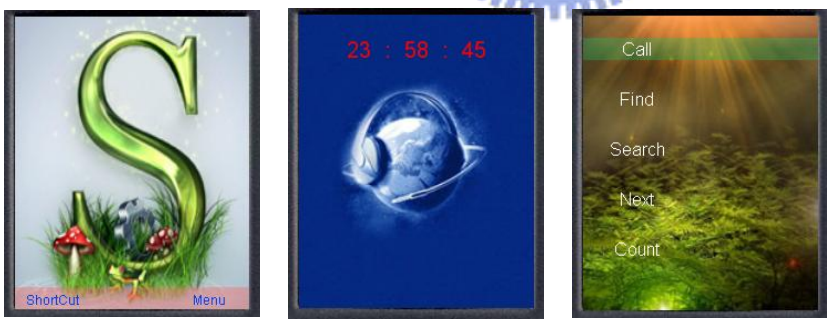
圖 9：XMMI v3.0 設計流程圖

介面架構包含了開發的數種 Component 與內建事件、介面的整合與動態呼叫；這些 Component 可以在 Application 中引用，讓應用程式開發者設計其功能

性，開發者只須宣告所須元件，建構並且初始化後，即可專注發展應用程式。而使用者可以利用 MIES 編輯出以 MIDL 為架構，能作到修改介面的 XML 檔案。在有了 Application 與 XML 檔案後，MIS 與 RTOS 會利用 Engine 動態引用 Application 來對介面作初始化，及讀取 XML 檔案來對介面修改外觀及操作方式。

3・2・2　Component 設計

因為行動裝置是一個特定目的應用的環境，加上硬體設備的限制，運作範圍也有所設限，所以我們可以先把一個應用程式可能的介面略分為幾種：第一：圖片與色塊，主要是顯示各種影像與影像。第二：文字，用來顯示出相關訊息或指示。第三：清單，讓使用者選取不同選項。第四：輸入，當使用者由行動裝置輸入（鍵盤、觸控板）時，其觸動的操作方式及引動的事件回饋，如使用者在待機畫面時，按下右軟鍵時（操作方式），會將畫面狀態轉移到選單上（事件回饋）。



1.圖片與色塊　　2.文字　　3.清單　　4.輸入

圖 10：常見的數種介面

我們參考以物件導向為主的 C#語言中的視覺化元件，並且考慮行動裝置上可能的介面規格，目前定義了下列幾種介面元件：MobileScreen，MobileButton，MobileLabel，MobilePicture，MobileInformation，以下對每種元件作介紹：

MobileScreen：每一個介面都需要一個 MobileScreen，如同字意，用來設定一個螢幕的相關屬性，而其他種介面元件均存放在 MobileScreen 這種容器中。MobileButton：觸發一個 MobileButton 的事件可以透過按鍵或觸控板，使用者可以自行設定是否要將按鈕顯示於介面上。MobileLabel：主要是用來顯示一些文字訊息，配合背景色塊，提示使用者資訊。MobilePicture：用來顯示圖片與影像，使用者可以自行隨意放置喜愛的圖示。MobileInformation：提供行動裝置系統資訊，讓使用者可自行引用來告知目前系統相關訊息。

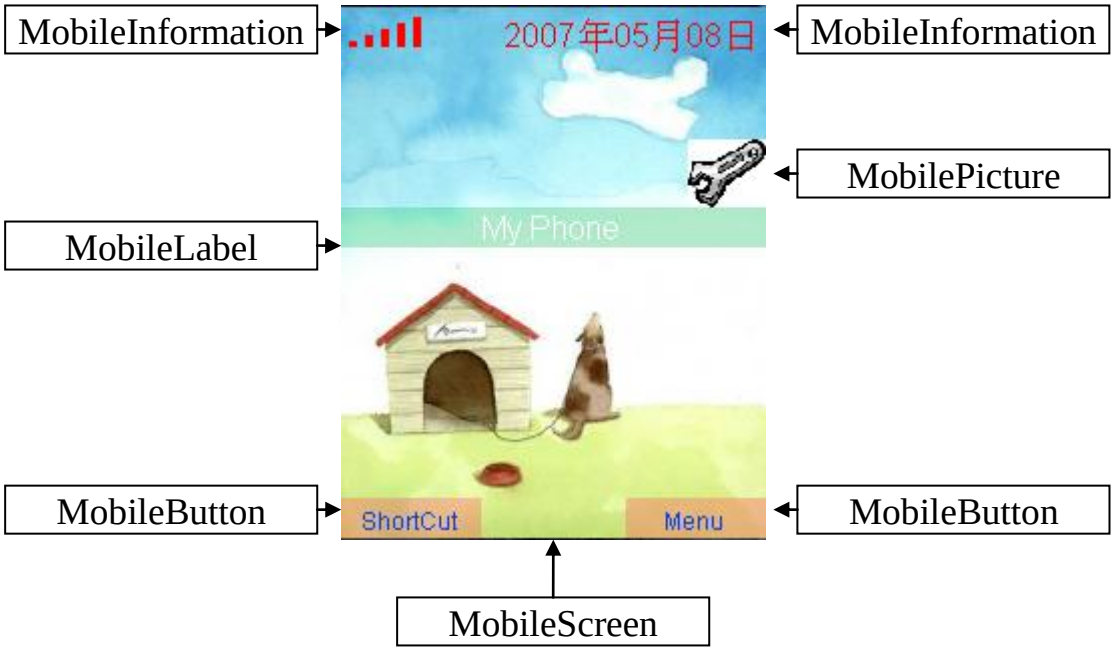


圖 11：Component 元件介紹

3・2・3 Application 設計

如同設計行動裝置上的應用程式，應用程式工程師只須負責功能性，無須理會介面的微調。舉例子來說，應用程式開發者正在設計一個聯絡人清單介面，需要一個方塊文字讓使用者在聯絡人清單中移動時，能夠顯示使用者目前選到的聯

絡人的註解說明，此時開發者只須在撰寫程式時，引用 Component，用此宣告一個 MobileLabel，建構並給定初始值後，開發者就可致力於元件功能性的發展上，將元件外觀開放給使用者修改。

在我們的想法中，一個介面即等於一個應用程式，觀察行動裝置的主要人機介面與參閱相關文件後，我們發現既有的介面狀態有三種：第一種是待機模式，即行動裝置最基本的介面狀態，位於狀態樹狀圖中的根節點。第二種是選單模式，在市面上手機的選單模式，大致都是九宮格為主，進入選單模式後，使用者可以依照所需功能選擇下一層的清單。第三種即是清單模式，不同於選單，清單大部份都是直式的選項。我們以此三種介面狀態為參考，設計出以 MIF 為架構的新介面，以下是較詳盡的介紹。

待機模式(Idle)：待機模式會提供一些系統資訊讓使用者知曉目前的狀態，比如說電池電量、信號強弱、日期、鍵盤鎖等等…所以在 Idle 的設計中，我們宣告了數個 MobileInformation 元件，能讓使用者自行決定是否顯示這些資訊，我們也考慮到個人化風格，放入了一個 MoibleLable 陣列，給使用者相當彈性的空間來增減欲呈現的訊息，如同 MSN 的暱稱。

選單模式(Menu)：當待機模式轉換至選單模式時，也意味著使用者需要去執行某個工作，利用圖示與文字，提示使用者可能的操作方式，希望使用者可以不去用記憶操作指令，減輕使用者負擔，所以我們也在設計中加入了一個 MobilePicture 陣列與一個與 MobilePicture 連接的 MobileLabel，當使用者按鍵移動，改變選取的焦點時，會有一個色塊顯示出目前選取的選項，並且連接的 MobileLabel 會自動更改為目前對應的選項文字。而按鍵的設計，我們宣告了四個隱藏按鍵（使用者也可以改為可見），這四個鍵一開始是對應著上下左右四個方向鍵，使用者可以在編輯系統操作，將對應的鍵值改為自己所習慣的按鍵，如此一來，有著不同操作方式的使用者，即使使用同一款手機，都能得心應用。

清單模式(List)：大致上與選單模式類似，差別在於選單模式中只有一個與 MobilePicture 連接的 MobileLabel，這個 MobileLabel 隨著焦點改變而變動，來達到提示使用者的目的；而清單則是一個 MobilePicture 陣列與一個 MobileLabel 陣列，每一個 MobilePicture 分別對應到一個 MobileLabel，每個選項的功能藉由文字與圖示，一目瞭然。

在 MIF 的架構底下，開發應用程式設計者可以不用調整介面元件外觀，開放內部元件資源讓使用者更改，而使用者所能更改的亦只有元件的外觀與操作方式，關於元件的事件撰寫與其他處理，均由應用程式負責。

1.待機狀態　　2.選單狀態　　3.清單狀態

圖 12：Application 介面狀態

3・2・4　動態鏈結函式庫

在整個系統架構中，不論是 MIF 中的 Component，或是 Application 與下一章將介紹的 Simulator Dll Files，均是以動態鏈結函式庫(Dynamic Linked Library, dll)檔案形式存在，設計動態鏈結函式庫檔案的好處在於應用程式可以隨時動態

地去連結程式庫，平時是以檔案形式存放於硬碟中，當應用程式需要呼叫它時，才將程序載入記憶體中執行，能夠節省記憶體資源。

當我們需要呼叫動態鏈結函式庫時，方法可分為兩種：隱式連結(Implicit Linking)與顯式連結(Explicit Linking)，在呼叫 Application 時，我們使用的是顯式連結技術，使用顯式連結的優點是在編譯時不須額外 include DLL 函式的宣告檔，並且在應用程式載入時並不需要一併將 DLL 載入到行程中，因此應用程式載入的速度較隱式連結快。

動態鏈結函式庫在我們系統中有其一定的重要性，不只是能夠節省記憶體資源，也是為了將應用程式的資源開放出來，藉由 MIDL 檔案修改介面，而 MIS 中的 Engine 即是作為兩者間的溝通者，如此一來，才能達到動態修改介面的功能。

3．3　行動裝置描述語言

3．3．1　Component 於 MIDL 的形式

系統中使用可延伸標記語言(eXtensible Markup Language, XML)作為元件介面的描述語言，在經過使用者利用 MIES 產生 XML 檔案後，可以確定 XML 文件為格式良好的，如此在 Parse 時才能維持資料的正確性。以下介紹各個元件的元素、意義與型別。

元素	意義	型別
Name	螢幕在應用程式中的名稱	（無法更改）
X	指定螢幕在水平的座標	正整數
Y	指定螢幕在垂直的座標	正整數
Width	指定螢幕的寬度	正整數
Height	指定螢幕的長度	正整數
BackColor	指定螢幕的背景顏色	正整數（Argb 值）
BackImage	指定螢幕的背景圖片	字串值（若值為空，則顯示背景顏色）
Visible	指定螢幕是否顯示	布林值

表 3：MobileScreen 屬性表

元素	意義	型別
Name	按鈕在應用程式中的名稱	（無法更改）
X	指定按鈕在水平的座標	正整數
Y	指定按鈕在垂直的座標	正整數
Width	指定按鈕的寬度	正整數
Height	指定按鈕的長度	正整數
BackColor	指定按鈕的背景顏色	正整數（Argb 值）
FontColor	指定按鈕的文字顏色	正整數（Argb 值）
FontSize	指定按鈕的文字大小	正整數
Text	指定按鈕的文字	字串值
Visible	指定按鈕是否顯示	布林值
Key	指定按鈕對應鍵值	字串值（列舉值）
Args	指定按鈕的事件參數	字串值

表 4：MobileButton 屬性表

元素	意義	型別
Name	標籤在應用程式中的名稱	（無法更改）
X	指定標籤在水平的座標	正整數
Y	指定標籤在垂直的座標	正整數
Width	指定標籤的寬度	正整數
Height	指定標籤的長度	正整數
BackColor	指定標籤的背景顏色	正整數（Argb 值）
FontColor	指定標籤的文字顏色	正整數（Argb 值）
FontSize	指定標籤的文字大小	正整數
Text	指定標籤的文字	字串值
Visible	指定標籤是否顯示	布林值

表 5：MobileLabel 屬性表

元素	意義	型別
Name	圖像在應用程式中的名稱	（無法更改）
X	指定圖像在水平的座標	正整數
Y	指定圖像在垂直的座標	正整數
Width	指定圖像的寬度	正整數
Height	指定圖像的長度	正整數
BackColor	指定圖像的背景顏色	正整數（Argb 值）
BackImage	指定圖像的背景圖片	字串值（若值為空，則顯示背景顏色）
Text	指定圖像的文字	字串值
Visible	指定圖像是否顯示	布林值
Link	指定圖像的連結檔案	字串值

表 6：MobilePicture 屬性表

元素	意義	型別
Name	資訊在應用程式中的名稱	（無法更改）
X	指定資訊在水平的座標	正整數
Y	指定資訊在垂直的座標	正整數
Width	指定資訊的寬度	正整數
Height	指定資訊的長度	正整數
BackColor	指定資訊的背景顏色	正整數（Argb 值）
FontColor	指定資訊的文字顏色	正整數（Argb 值）
FontSize	指定資訊的文字大小	正整數
Visible	指定資訊是否顯示	布林值
Type	指定資訊的類別	字串值（列舉值）

表 7：MobileInformation 屬性表

3・3・2　MIDL 範例

以下我們用一個 XML 檔案中的 MIDL 格式來說明元件介面的修改。

```
1  <MIS>
     <Application Name="Menu">
2      <MobileScreen Name="MenuScreen" X="0" Y="0" Width="212" Height="265" BackColor="0"
       BackImage="Menu.jpg" Visible="True"></MobileScreen>
3      <MobileButton Name="MenuLinkButton" X="0" Y="244" Width="70" Height="21"
       BackColor="-2004458980" FontColor="16635" FontSize="14" Text="Select" Visible="True"
       Key="SoftLeft" Args=""></MobileButton>
      <MobileButton Name="MenuBackButton" X="142" Y="244" Width="70" Height="21"
      BackColor="-2004458980" FontColor="16635" FontSize="14" Text="Exit" Visible="True"
      Key="SoftRight" Args="Idle.xml"></MobileButton>
```

```
4   <MobileButton Name="MenuNextButton" X="0" Y="245" Width="70" Height="20"
    BackColor="1694458980" FontColor="16635" FontSize="12" Text="ShortCut" Visible="false"
    Key="Left" Args=""></MobileButton>
    <MobileButton Name="MenuLastRightButton" X="0" Y="245" Width="70" Height="20"
        BackColor="1694458980" FontColor="16635" FontSize="12" Text="ShortCut" Visible="false"
        Key="Right" Args=""></MobileButton>
5   <MobileLabel Name="MenuTitleLabel" X="0" Y="0" Width="212" Height="20"
    BackColor="-2004458980" FontColor="16777215" FontSize="14" Text="" Visible="True">
    </MobileLabel>
6   <MobileInformation Name="MenuInformation" X="132" Y="20" Width="80" Height="12"
    BackColor="-200445898" FontColor="-65536" FontSize="10" Visible="true" Type="Date">
    </MobileInformation>
7   <MobilePicture Name="MenuItemPicture" Number="9">
        <Array X="6" Y="40" Width="60" Height="60" BackColor="1294458980" BackImage="A.png"
        Text="Messages" Visible="True" Link="1.xml"></Array>
        <Array X="76" Y="40" Width="60" Height="60" BackColor="1294458980" BackImage="B.png"
        Text="Call Register" Visible="True" Link="2.xml"></Array>
        <Array X="146" Y="40" Width="60" Height="60" BackColor="1294458980" BackImage="C.png"
        Text="Contacts" Visible="True" Link="3.xml"></Array>
        <Array X="6" Y="110" Width="60" Height="60" BackColor="1294458980" BackImage="D.png"
        Text="Settings" Visible="True" Link="4.xml"></Array>
        <Array X="76" Y="110" Width="60" Height="60" BackColor="1294458980" BackImage="E.png"
        Text="Gallery" Visible="True" Link="5.xml"></Array>
        <Array X="146" Y="110" Width="60" Height="60" BackColor="1294458980" BackImage="F.png"
        Text="Media" Visible="True" Link="6.xml"></Array>
        <Array X="6" Y="180" Width="60" Height="60" BackColor="1294458980" BackImage="G.png"
        Text="Organiser" Visible="True" Link="7.xml"></Array>
        <Array X="76" Y="180" Width="60" Height="60" BackColor="1294458980" BackImage="H.png"
        Text="Applications" Visible="True" Link="8.xml"></Array>
        <Array X="146" Y="180" Width="60" Height="60" BackColor="1294458980" BackImage="I.png"
        Text="Services" Visible="True" Link="9.xml"></Array>
    </MobilePicture>
8   </Application>
</MIS>
```

表 8：XMMI v3.0 MIDL 範例

1：MIS 為 MIDL 的根節點，描述行動裝置介面狀態的開頭，而 Application 中的 Name 則說明這個介面的應用程式名稱，以讓 Engine 動態呼叫與名稱相符

的 dll 檔案。

2：設定螢幕的介面，位置、大小、背景顏色、背景圖片、可視性。背景圖片可接受 jpg, gif, bmp, png, ico 等一般圖片格式。

3：設定作為連結狀態按鍵的介面，位置、大小、背景顏色、文字顏色、文字大小、文字、可視性、鍵值、事件參數。當連結為下一頁時，需傳入一個連結檔案參數，而當連結為上一頁時，由於系統放有堆疊記憶狀態路徑，所以在回到上一頁時，可直接由堆疊中取出來，而不需給定連結檔案參數。鍵值為固定的列舉值，目前有左右軟鍵、方向鍵、數字鍵、星井字鍵、通話鍵、取消鍵２０個鍵值。

4：設定作為移動焦點按鍵的介面，由於這類型的按鍵通常不會顯示於螢幕上，所以可以不用理會外觀上的調整，最主要的是調整對應的鍵值，好讓使用者選擇自己所習慣的操作方式。

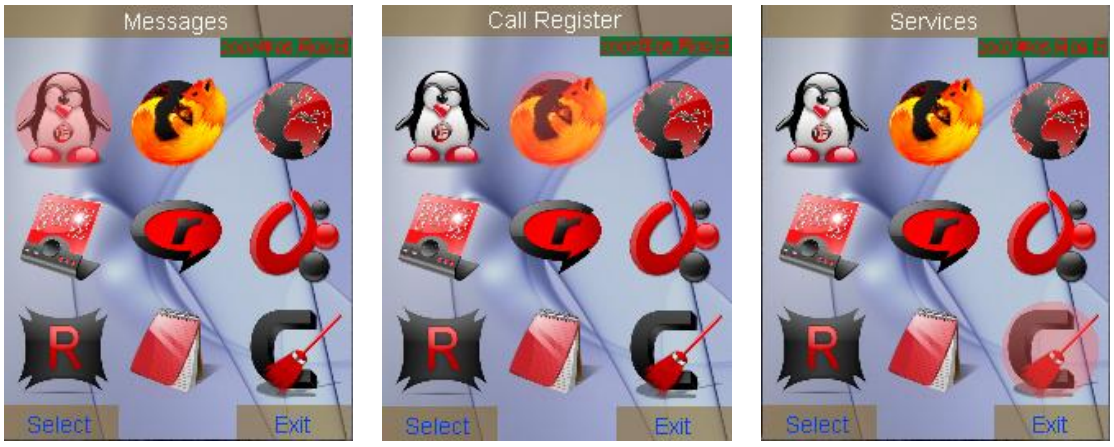
5：設定作為標題列標籤的介面，位置、大小、背景顏色、文字顏色、文字大小、文字、可視性，因為這個標籤的文字會隨著焦點不同而更改，文字屬性的輸入可以省略。

6：設定顯示系統資訊的介面，位置、大小、背景顏色、文字顏色、文字大小、可視性、類型，使用者可以調整類型決定顯示不同的系統訊息，目前有提供五種類型，電池電量、訊號強弱、按鍵鎖、日期、時間。

7：設定選單圖像陣列的介面，由使用者自行決定陣列大小，動態產生圖像，可設定位置、大小、背景顏色、背景圖片、文字（顯示於標題列）、可視性、連結檔案，每一個圖像代表著一個連結，當使用者焦點改變時，對應的連結檔案也會改變。

8：表示一個 Application 內元件的描述已完畢，MIS 結束。

以下是上述 MIDL 在個人電腦平台上模擬器執行出來的結果：



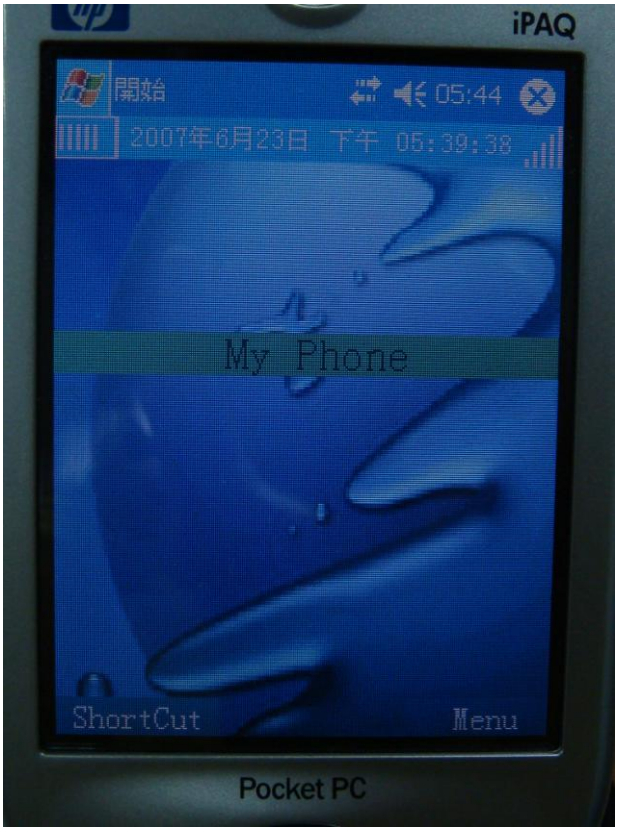
開始執行　　鍵入 MenuNextButton　　鍵入 MenuLastButton

圖 13：選單狀態於 MIS 模擬出的畫面

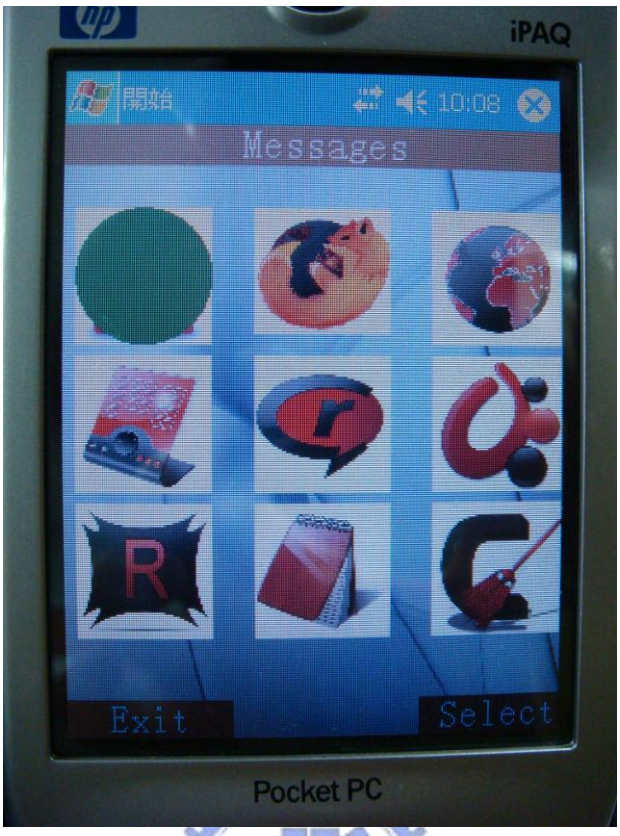
3．4　實作 XMMI v3.0 於行動裝置

本節以 HP 公司出產的 PDA 型號 iPAQ-4150 作為實作的行動裝置，iPAQ 作業系統為 Windows Mobile 5.0，我們先將 MIF 架構建立於 iPAQ 的 RTOS 上，並將 MIS 安裝至 iPAQ 中，有了 MIS 模擬與 MIF 元件後，僅需要的是 MIDL 檔案與 Application 檔案。使用者可經由 MIES 編輯出 MIDL 檔案後，可利用 Microsoft 公司為 Smartphone 所提供的 ActiveSync 軟件，將已建構的 Application 檔案與 XML 檔案同步移至行動通訊的儲存裝置中，而在 PDA 上，事件可支援 TouchPad 觸控，利用觸控筆可在所需要的選項上點選，而不需再如同一般手機利用鍵盤控制選項。在 PDA 操作時，一開始的起始畫面即為 Idle 的待機畫面，當使用者利

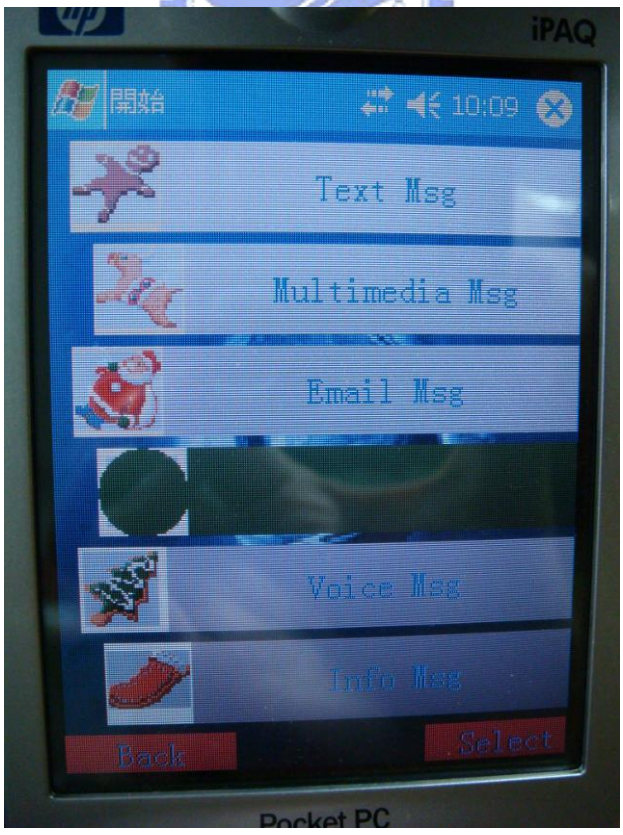
用觸控筆碰觸 Idle 介面的右鍵時(顯示 Menu 處)，可引發 Link 事件，將目前的模式狀態改變為 Menu 介面，同樣地使用者可利用觸控筆輕觸各個自製的選單物件，則會有焦點顯示使用者目前所按的物件，在選取好選單後，可輕觸 Menu 介面的左鍵(顯示 Exit 處)回到上一頁，即 Idle 狀態，亦可觸碰右鍵(顯示 Select 處)，進入 List 清單狀態，而 List 的操作如同 Menu，可自由選取物件、回上層(Menu)、或繼續進入下個狀態，以下為實作時的擷取畫面：



Idle 待機模式



Menu 選單模式



List 清單模式

第四章　個人電腦平台上之 MIS 模擬設計與實作

4．1　MIS 模擬器設計

模擬器的設計目的在於個人電腦缺乏一支真實的手機，在操作介面編輯器時能夠輔助使用者，順利模擬出真實畫面。在我們的模擬器中主要包含了三個部份：Module 及 Engine 與 EventHandler；如同一般市面上的模擬器，一個模擬器的設計可以區分成三個階段：裝置的模組，畫面的呈現，事件的輸入，以下說明 MIS 中各部份與這三個階段的相關性。

Module 簡單的來說即代表不同款示的模擬器，MIS 中最重要的即是一個虛擬的行動裝置，Module 中可以發展出各式的模擬器裝置，這可利用下一節介紹的 MobileCreator 製造。

Engine 負責的是將 MIES 產生出的以 MIDL 為基礎的 XML Files 解析，修改一個應用程式的元件介面。由於我們的系統將元件開放給使用者與應用程式者，而當應用程式設計者開發出新的應用程式發佈後，使用者可以在編輯器引用應用程式，並開始編輯應用程式開放的元件介面，當編輯出此應用程式的介面描述檔案後，透過 Engine 會讀取檔案，並動態建構這個介面的主體，即應用程式，而後剖析檔案中定義的結構，動態修改其中的介面元件，最後顯示畫面在裝置模組的螢幕上；進一步地，我們在動態鏈結上作到了陣列的動態宣告大小，不同於以往程式所規定的，陣列大小必須在一開始宣告時就給定大小，而是在 Engine 進行 Parse 時動態決定大小。

比如說，在待機狀態時，我們不知道使用者需要幾個 MobileLabel，而在 Idle 應用程式設計時我們只須宣告一個 MobileLabel 陣列，但不須宣告其大小，當使

用者在 MIES 編輯 Idle 這個應用程式介面時，可以自行決定需要的大小，並寫入 XML 檔案中。在經過 Engine 編譯後，Idle 應用程式的 MobileLabel 陣列此時才真正有其大小；如此作到動態的效果，不僅能讓使用者編輯多了相當大的彈性，應用程式開發者亦不需要因為宣告過大或過小的空間而造成資料浪費或陣列不足。

EventHandler 處理的是裝置模組上的輸入，即 Module，由於 Module 的輸入事件處理是定義在自身模組的程序中，MIS 是作為 Engine 與 EventHandler 的中間溝通者，當裝置模組由外接受輸入時，會引動內部事件處理，但執行什麼事件卻是 Application 所設計決定的；所以 EventHandler 所需要處理的即是將 Engine 編譯過後中的產生的 Application 中元件對應的事件指定給 Module 中的事件模組，如此一來才來正確的輸入，並執行對應事件。

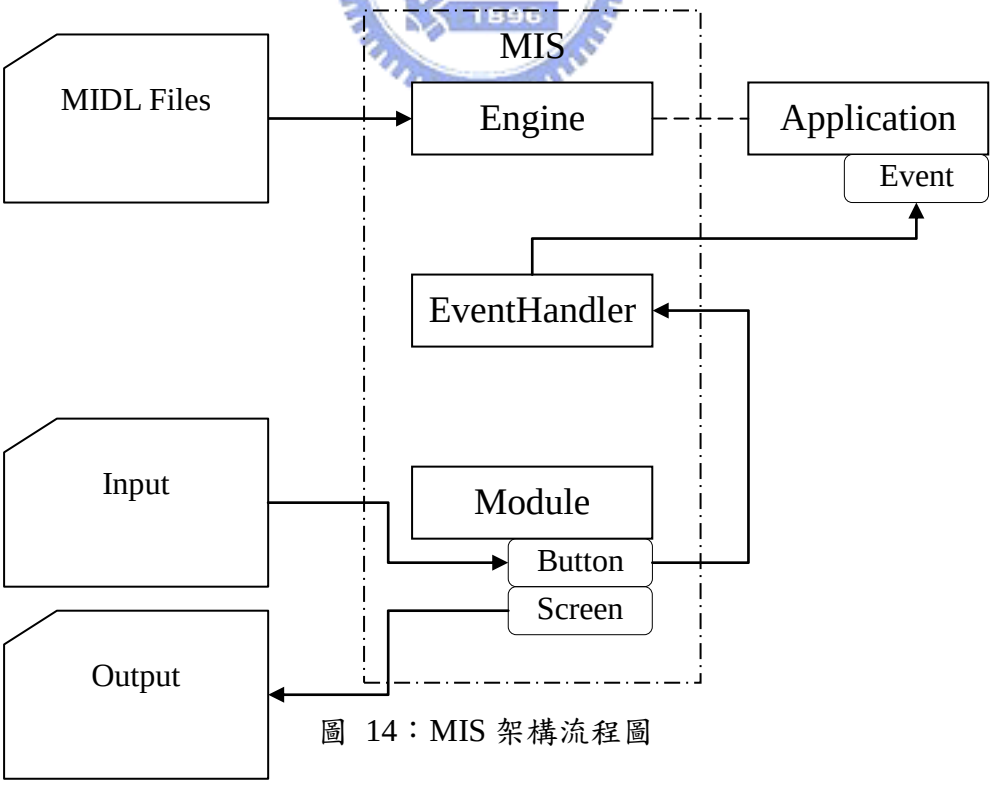


圖 14：MIS 架構流程圖

4．2　MSC 行動模擬創造器

考量到使用者所使用的行動裝置款示與內建模擬器不同而影響到外觀與按鍵的操作不便，我們建立了一個可以讓使用者創造出屬於自己的模擬器的簡易編輯器(MSC)。MSC 可以讓使用者製作出個人化的行動裝置模擬器，即 MIS 中的 Module 延伸類別，包括此裝置的資訊，外觀，螢幕與按鍵位置等，這些資訊都是一小段程式碼，但當使用者在創造時卻並不會覺得自己在寫程式，加上使用動態編譯，可將圖片與程式碼嵌入轉換成一個 dll 檔，作到了防止資源外洩的目的。在 MSC 中，我們運用了 csc.exe 這個.Net Framework 內建 Compiler；在使用者創造後一款行動裝置模擬器後，會產生一個對應的 cs 程式碼檔案，並且呼叫 csc.exe 將此程式碼檔案與圖片編譯為 dll 檔案，以讓 MIS 引用。

圖 15：MSC 設計 Simulator 時截圖

第五章　結論

5．1　總結

在現今的社會中，行動裝置已經融入生活，成為每個人的一部份，也由於行動裝置的廣泛及應用，如何將其客製化，符合每位使用者的需求，將會是一種趨勢；在電信界居領導地位的日本，甚至發展出另一套「訂製化手機」模式。訂製化手機讓系統業者握有更大的主導權，以服務內容來決定手機規格。

一個好的使用者介面的客製化，應該具備讓使用者定義的機制，讓使用者能夠定義最符合自己習慣的操作模式。於是我們設計了 XMMI v3.0，將使用者介面設計與軟體功能分離，行動裝置使用者可利用 MIES 平順地(Smoothly)處理 MIDL Files，無須理會底層的構造，而程式開發者也不必將介面外觀列入設計考量，只須專注於功能性發展上。

5．2　未來工作

在未來，我們期望 XMMI v3.0 系統能夠真正的移植到行動裝置晶片上，實作出真正的產品，另外，由於透過 VS 內建元件的輔助，我們開發 Component 的時間雖然縮短了許多，但也相對地，內建元件的資源消耗也較大，在未來，希望考量使用資源，開發完全真正屬性行動裝置的 Component。

另一部份，由於行動裝置多媒體資源類型愈來愈多，愈來愈廣泛，如聲音、

影片、Flash，我們的 Component 目前還只限於文字與圖片，在將來，我們會把 Component 繼續擴充，支援更多樣的多媒體資源型態。而在事件輸入部份，我們會配合裝置硬體再增加 KeyHover(壓住鍵不放)、KeyConnect(連續按固定一系列的鍵值，才產生對應事件)。

由於在手機中有不同模式的選擇，如一般模式轉至無聲模式，每一種模式的 XML 檔案只有一些許的差異設定值，如何用最少的 XML 檔案來表示多種模式的選擇也值得我們將來探討的問題。

參考文獻

[1] Theme Generator Smartphone http://www.downpda.com/downinfo/5987.html

[2] 手機客製化 http://www.phonedaily.com/news/?news_id=4218

[3] 人機介面之設計 http://squall.cs.ntou.edu.tw/UserInterface/DesignOfHCI.html

[4] Xml 教學 http://www.study-area.org/coobila/category_XML_u6559_u5B78.html

[5] WinCE http://playstation2.idv.tw/iacolumns/jl00022.html

[6] Nokia N73 http://www.phonedaily.com/mobile/phone/?prod_id=2216

[7] Wiki Adobe Flash http://en.wikipedia.org/wiki/Macromedia_Flash

[8] 個性化手機 http://3c.msn.com.tw/page.asp?m=&s=&id=7095

[9] 智慧型手機比較 http://www.phonedaily.com/news/?news_id=2706

[10] 莊盟錫，「可延伸式人機介面系統」，國立交通大學，碩士論文，民國 93 年

[11] 黃明超，「行動裝置可延伸人機介面系統及其快速建模平台」，國立交通大學，碩士論文，民國 94 年

[12] MVC 架構 http://blog.csdn.net/qjyong/archive/2006/12/04/1429907.aspx