國立交通大學

資訊科學系

碩 士 論 文

XMMI－可延伸式人機介面系統

XMMI – eXtensible Man Machine Interface System

研 究 生：莊盟錫

指導教授：李嘉晃　教授

中 華 民 國　九 十 三　年 六 月

可延伸式人機介面系統

研究生：莊盟錫　　指導教授：李嘉晃 教授

國立交通大學電機資訊學院 資訊科學系碩士班

中文摘要

現今行動電話這類嵌入式系統上的軟體開發，忽略了嵌入式系統的一些特定功能，仍然是把它當成一般的電腦平台。XMMI 的架構，是把嵌入式系統當成是單一的應用程式，例如：應用程式是“行動電話”，而行動電話的功能模組有電話簿、簡訊、…等。

本論文所提出的 XMMI 架構，利用 XML 描述狀態機的行為，將嵌入式系統操作流程描述在 XML 的檔案裡，不僅增加了開發的速度，也讓介面能更容易的做客製化及個人化的設定。藉由調整 XML 檔案不僅能改變圖形化介面的外觀，進一步也可以對機器的運作方式做調整。

Extensible Man Machine Interface System

Student : Meng-Xi Zhuang Advisor : Prof. Chia-Hoang Lee

Department of Computer and Information Science

National Chiao Tung University

ABSTRACT

Because the current application development on embedded system usually ignores its specific functionalities, the methodology of developing applications remains the same as that of a general desktop platform. The proposed XMMI architecture treats the embedded system itself as a single application from software perspective. For example: Cellular phone can be considered as an application with functional modules such as phone book, short message service, etc.

XMMI architecture describes the behavior of state machine using XML descriptions that recorded embedded system's operational process. This approach could achieve customization/personalization easily and speed up the manufacturing procedure. It could change not only the GUI style, but also the mode of operations of embedded system by modifying XML descriptions.

誌謝

本論文得以順利完成，首先要感謝的是指導教授李嘉晃老師，他在我研究所求學的過程中，不斷的給我機會發揮創造，給予我充分的空間發展，讓我在研究過程中獲益良多。也感謝口試委員王勝德教授、李肇林教授以及陳登吉教授的指正建議，使得本論文能更加的完善。

還要感謝聯發科技交大研究中心的研究夥伴，晉華、昭隆及楊裕全副理所提供的協助。另外還要感謝實驗室的同學們，彥琨、明德、兆良、存厚、哲寬、耀輝，及學弟們對我的關心與幫助，充實了我的實驗室生活。

最後，我要感謝我的家人，因為他們的支持與鼓勵是我努力的最大動力，僅以此篇論文，獻給他們。

目錄

圖目錄

第 1 章

簡介

1．1 動機

隨著計算機軟硬體技術的進步，嵌入式系統的應用，也漸漸的在我們的生活中漫延開來。嵌入式系統的特性是少有外接的零配件、具有特定的功能、容積小、穩定性強的特點，而系統軟體的設計與規劃須兼顧上述的特性。嵌入式系統的應用甚廣，包括在生活中常見的行動電話、個人數位助理（PDA）、數位相機、掌上遊樂器、汽車上的儀表板系統、…等等。

嵌入式系統和一般用途的個人電腦（Desktop PC）具有類似的軟硬體架構，硬體方面，有中央處理器（CPU）、記憶體、…等，而軟體方面，嵌入式系統也有作業系統、應用程式。常見的嵌入式系統的硬體架構有：

- x86 架構：美國國家半導體（NS）公司的 Geode 系列 CPU，這類的 CPU 有用於上網機（Set Top Box）。
- ARM 架構：ARM 系列 CPU 中最常見的是 Intel 公司的 StrongArm CPU，現階段被廣泛的應用在功能較強大的 PDA 或是行動電話上。
- MIPS 架構：MIPS 系列 CPU 中以 NEC 公司出產的 NEC VR 系列最出名。
- PowerPC 架構：此系列當推 Motorola 的 DragonBall 系列 CPU，大家耳熟能詳的 Palm 相容 PDA 即採用此架構。

嵌入系統的作業系統除了一般的作業系統功能外，它還需具備即時運算的能力（Real Time）常見作業系統有：

- DOS：系統簡單，適合一些功能簡單裝置使用，如 LED 字幕機。
- Windows CE：微軟所推出精簡版的 Windows CE 作為進攻嵌入式系統的主力，目前主要應用除了 PDA，也出現在微軟的行動電話 Smart Phone 上。
- Symbian EPOC：由英國手持裝置大廠 Psion 所開發，常用於 PDA 與手機結合的場合。

其他還有 Embedded Linux、VxWorks、RTXC、Nucleus、…等。[1]

隨著嵌入式系統及顯示原件技術的進步，許多嵌入式系統的人機介面（Man Machine Interface, MMI），為了提供更友善（Friendly）的使用者介面，皆採用高解析度的彩色螢幕來呈現圖型化的使用者介面（Graphical User Interface, GUI）。但現今嵌入式系統上的圖型化使用者介面系統的設計，大多沒有脫離以傳統的應用軟體設計方式。傳統的設計方式所開發出來的介面，與功能性的程式碼通常是很難分開來的，即使有整合性的圖型介面開發工具，也只是便於設計者在開發階段使用，使用者無法依照個人使用習慣對介面做調整及修改。即使整個系統的功能沒有改變，系統的開發者，都需要修改大量的原始碼，才能修改使用者介面或修改系統行為。目前廠商提供使用者調整介面功能的技術，都只侷限於佈景主題（theme）的個人化而已。

1．2 目標

在現今的嵌入式系統的架構上，要修改使用者圖型化介面或者是系統執行流

程，不論對使用者或者是系統開發人員來說，都是一件繁鎖而且困難的工作。現今在嵌入式系統或行動運算的使用介面問題研究上，是以研究如何將一個一般工作站的應用程式的介面呈現在嵌入式系統中[2][3]。嵌入式系統都有特定的功能且顯示器的空間有限，我們在這篇論文中提出一個新的嵌入式系統人機介面系統設計架構，能讓嵌入式系統開發人員及使用者能夠快速的開發或調整使用者介面及調整系統在運作上的各種行為。做調整時，不需要更動相關的系統功能程式碼。

我們要在嵌入式系統開發一套新的 MMI 運作機制，此機制包含使用者介面解譯器（Embedded User Interface Interpreter, EUII）及嵌入式系統使用者介面的描述語言（Embedded User Interface Description, EUID）。此描述語言採用可延伸式標示語言（eXtensible Markup Language, XML）[4]為其關鍵技術。EUII 將主導嵌入式系統的 MMI 運作，解讀 EUID 並處理相對應的事件。我們將這個架構稱為可延伸式人機介面系統（eXtensible Man Machine Interface system, XMMI）由於嵌入式系統的應用廣泛且種類繁多，我們將以個人化需求度最高的行動電話做說明及開發。未來採用 XMMI 架構設計的行動電話，將可透過紅外線傳輸（Infrared Data Association, IrDA）、藍芽（Blue tooth）、無線上網傳輸協定（Wireless Application Protocol, WAP）、…等方式，將符合使用者喜好的介面樣式下載至行動電話中。運用 XMMI 機制，不只能改變整個 GUI 樣式、版面配置，也能改變該行動電話的操作行為。

1．3 論文架構

本篇論文的架構如下：

第二章探討研究背景及說明 XML 基礎的相關技術。介紹可延伸式標記語言（eXtensible Markup Language, XML）[4]、文件型態定義（Document Type

Definition, DTD）[5]、可變式向量圖形(Scalable Vector Graphics, SVG)[6]、Winamp3.0 外皮系統（Skin System）[7]及目前行動電話對介面個人化的處理。

第三章主要說明本篇論文所提出的可延伸式人機介面（eXtemsible Man Machine Interface system, XMMI）架構。介紹 XMMI 的架構及想法，以及模擬行動電話使用 XMMI 架構的運作情形。

第四章則是以實際的行動電話為開發平台，在現行的人機介面系統（Man Machine Interface, MMI）架構下設計並且實作了被動式的 EUII 系統。

第五章為結論，以及討論為來的工作。

第 2 章

研究背景

2．1 背景

讓我們先假想一個狀況：一個發生在現實生活行動電話的開發者、販售者與使用者發生的故事。某行動電話設計製造公司 M，設計了一部新型的行動電話 no-X，行動電話的販賣商 V 與 M 洽談後，決定購入 no-X 型行動電話，但是有關介面的部份需要做某些簡單的調整，於是設計該型行動電話的 M 公司工程師，依據 V 的需求規格，開啟了行動電話的所有程式碼，重新調整及修改介面以符合 V 想要的樣式。經過幾次和 V 的交涉後，V 便將 no-X 型行動電話上市販售。但 no-X 型行動電話的銷售不如預期。V 做了市場調查後，發現原來使用者對介面的操作不習慣，約五成使用者希望待機畫面下的右軟鍵（Right Soft-key）能直接進入電話簿，約四成使用者希望右軟鍵能是一個快速鍵功能（Short-cut），還有一部份年輕的使用者認為圖型化介面樣式過於呆板。V 得到了市場調查的結果，便要求製造公司 M，以大部份人的需求將待機畫面的右軟鍵改成進入電話簿的功能，並且調整待機畫面的圖型畫介面樣式。M 公司的工程師，又打開了 no-X 行動電話的所有程式碼，按照 V 的新需求，重新修改整個程式。

以上的故事，雖然行動電話的功能完全沒有改變過，但在修改使用者介面的時候，卻一再需要重覆的修改。每一次修改過後，都需要經過許多的測試，來確定功能及介面都能正常的運作。另一個問題是，使用者沒有能力對介面做個人化的調整。由於行動電話的軟體設計，還是把它當成一般的桌上型電腦的應用程式開發，例如，在手機上“通訊錄”的功能，是被視為一個獨立應用程式，因此要

調整 GUI 時，就必須修改“通訊錄”這個程式的程式碼。而有關在故事中右軟鍵功能的問題，在現在的 MMI 設計上，也是在開發時就固定了，所以很難再進行調整。

在行動電話這類具備顯示器的嵌入式系統中，修改圖形化介面的問題相當困難及繁鎖，造成這些問題的原因，就是現在的行動電話設計上，未掌握嵌入式系統的主要特性—嵌入式系統都具備有特定功能。若將行動電話整個視為一個應用程式來開發，便能解決這個問題，而像“通訊錄”、“簡訊”、…等，這些功能，可看成是應用程式中的模組，而這個應用程式的 GUI 的工作，則全部交給一個 GUI 的模組處理。整個應用程式的核心部份為 MMI 模組，負責接收處理這個應用程式的事件（Events）及維繫這個應用程式的模組間的運作，參考圖２-１、圖２-２。這樣的架構能使整個系統高度模組化，區隔了介面相關的程式和功能性的程式。提高了功能性程式的重用性（Reusability），使得整個系統的維護上會更加容易。以下我們將說明 XMMI 所採取的相關技術。

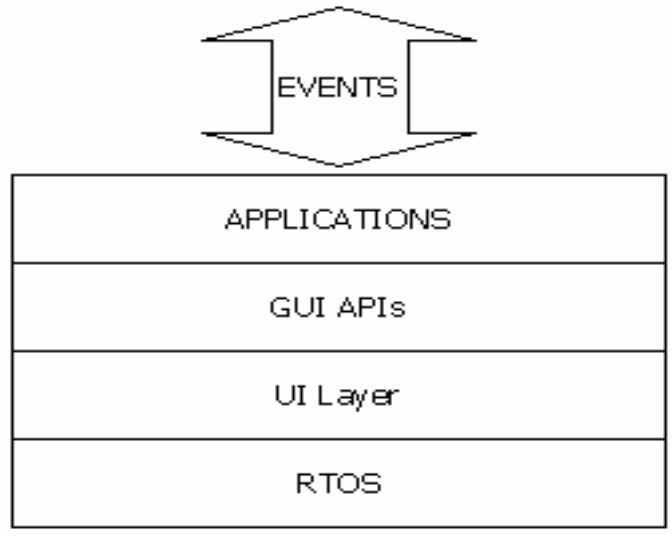


圖２-１ 傳統的 MMI 架構

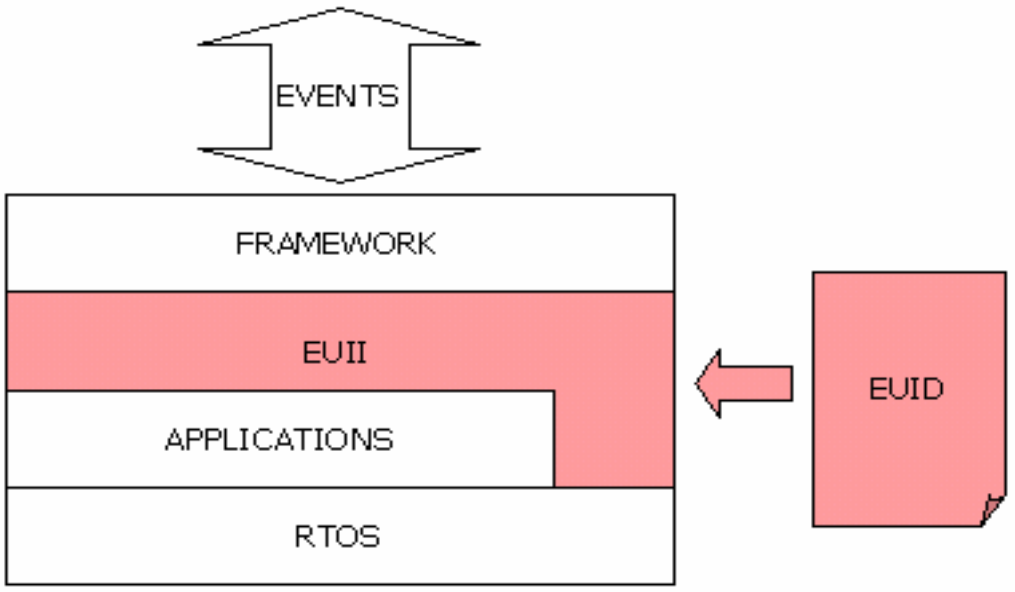


圖２-２ XMMI 架構

２・２ 相關基礎技術

為了在第三章我們會說明 XMMI 的實作。底下先介紹在實作 XMMI 會用到相關的基礎技術及理論，包含有可延伸式標示語言（eXtensible Markup Language, XML）、文件型態定義（Document Type Definition, DTD）[5]。

２・２・１ 可延伸式標示語言（eXtensible Markup Language, XML）

隨著網際網路的發展，在網路上的應用越來越多樣化，但這些應用常常都有各自描述資料的格式。例如：圖２-３-１為 WordPad 資料原始檔案 RTF（Rich Text Format）；圖２-３-2 為該原始檔案在 WordPad 應用程式的呈現。WordPad 的原始檔案除了可讀性低以外，描述資料的格式是 WordPad 的特定規格，使得這份文件不容易在其他應用軟體上使用。即使資料的內容沒有改變，這些資料的維護人員，必須為不同的應用軟體，編寫符合各種特定規範的文件，但是這些編寫文件的人員，常不是專業的軟體設計工程師，這樣也使得文件的重用性降低。

{\rtf1\ansi\ansicpg950\deff0\deflang1033\deflangfe1028
{\fonttbl{\f0\fmodern\fprq6\fcharset136 \'b7\'73\'b2\'d3\'a9\'fa\'c5\'e9;}}
{\colortbl ;\red255\green0\blue0;\red0\green0\blue255;}
{*\generator Msftedit 5.41.15.1503;}\viewkind4\uc1\pard\cf1\lang1028\f0
\fs24\'a4\'e5\'a5\'f3\'a6\'57\'ba\'d9\'a1\'47\'aa\'4f\'a4\'e2\'a7\'de
\'b3\'4e\'a4\'e2\'a5\'55\cf0\par\'a7\'c7\'b8\'b9\'a1\'478000101\par
\'a7\'40\'aa\'cc\'a1\'47\'a4\'6a\'c4\'5f\par\'c3\'fe\'a7\'4f\'a1\'47\'be\'f7
\'b1\'4b\par\cf2\'a4\'e5\'a5\'f3\'a6\'57\'ba\'d9\'a1\'47\'aa\'4f\'a4\'e2
\'a8\'cf\'a5\'ce\'a4\'e2\'a5\'55\cf0\par\'a7\'c7\'b8\'b9\'a1\'477000201\par
\'a7\'40\'aa\'cc\'a1\'47\'a4\'70\'b1\'6a\par
\'c3\'fe\'a7\'4f\'a1\'47\'a4\'bd\'b6\'7d\fs20\par}

圖 2 - 3 - 1　WordPad 資料原始檔案

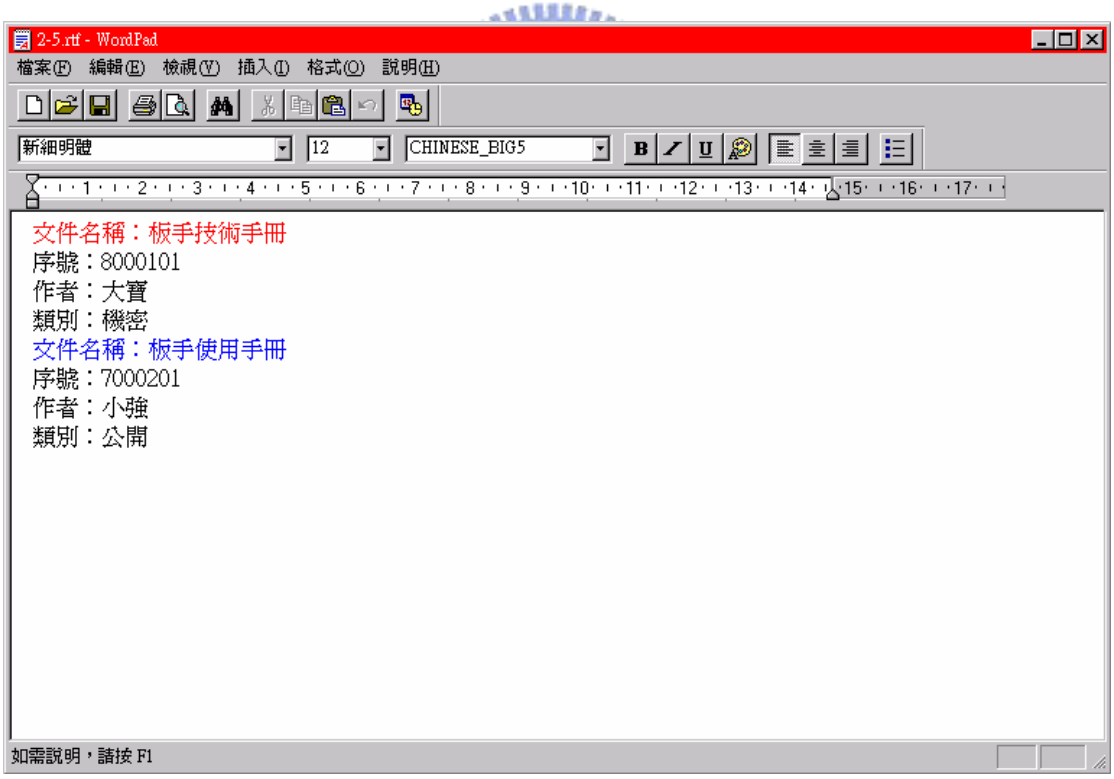


圖 2 - 3 - 2　WordPad 原始檔案的呈現

```
<html>
<title>Chapter 2 HTML Example</title>
<body>
<font size="3" color="red">文件名稱：板手技術手冊</font><br>
序號：8000101<br>
作者：大寶<br>
類別：機密<br>
<font size="3" color="blue">文件名稱：板手使用手冊</font><br>
序號：7000201<br>
作者：小強<br>
類別：公開<br>
</body>
</html>
```

圖 2-4-1 HTML 文件原始檔

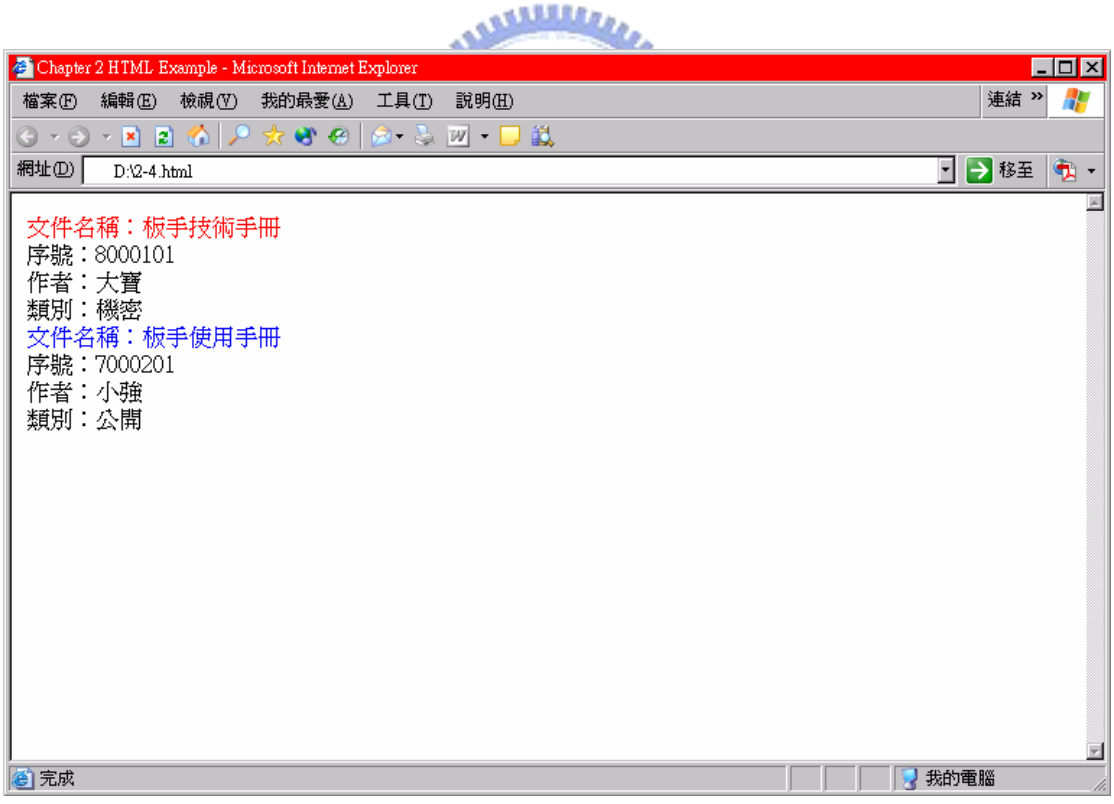


圖 2-4-2 HTML 文件在瀏覽器上的呈現

超文字標示語言（Hyper Text Markup Language, HTML）和 XML 一樣屬於標示語言的一種，“標示”就是在文件中加入標籤註記有關文件內容的資訊。

HTML 的標示，指定了文件的呈現方法，因為簡單易讀，已經在網際網路上流行，可說是標示語言應用最成功的一種。由圖２-４-１的 HTML 原始檔案，我們能了解文件中的“文件名稱：板手技術手冊”，會以 3 單位大小的紅色字型呈現，大大的改善了原始檔案不易閱讀的問題。但由於 HTML 的標示只描述了有關文件呈現方式的資訊，其他的應用程式，若是要對該文件做其他處理是很困難的。例如：應用程式能容易的找到需要以紅色呈現的部份，但是卻很難找到文件中有關文件編號的部份做處理。

```
<?xml version="1.0" encoding="big5"?>
<?xml:stylesheet type="text/css" href="2-6.css"?>
<!DOCTYPE documents SYSTEM "2-6.dtd">
<documents>
    <document serial="8000101">
        <confidential>文件名稱：<dn>板手技術手冊</dn></confidential>
        <description>序號：<sn>8000101</sn></description>
        <description>作者：<an>大寶</an></description>
        <description>類別：<dt>機密</dt></description>
    </document>

    <document serial="7000201">
        <public>文件名稱：<dn>板手使用手冊</dn></public>
        <description>序號：<sn>7000201</sn></description>
        <description>作者：<an>小強</an></description>
        <description>類別：<dt>公開</dt></description>
    </document>
</documents>
```

圖２-５-１ XML 原始檔案

```
confidential{color: red;}
public{color: blue;}
description{display: block;}
```

圖２-５-２ CSS 樣式檔案

XML 是由全球資訊網聯合會（World Wide Web Consortium, W3C）所推出正式的標準定義規範。XML 是一種把文件內容與呈現方式分開的標示語言，它不像 HTML 一樣有固定的標籤，使用者必須定義自己的標籤。由使用者定義的這些標籤通常代表著這份文件內的資料描述，以便其他的應用程式再對文件做處理。對於 XML 文件的呈現方式，可透過串接樣式（Cascading Stylesheet, CSS）定義。圖２-５-１是將圖２-４-１的例子寫成 XML，檔案裡的標籤描述了文件的內容，這些標籤能幫助應用程式在這份文件中，很快的找到有關作者或其他資訊；圖２-５-２的 CSS 檔案，描述了有關 XML 文件的呈現方式，透過這樣的方式，我們一樣能得到圖２-４-２的呈現效果。XML 文件格式有幾項限制：

- 只有一個根原素
- 必需要有結束標籤
- 文件標示結構必需為巢狀結構

符合上述限制才算是一份良好格式（Well formed）的 XML 文件。

２・２・２ 文件型態定義（Document Type Definition, DTD）

DTD 是更進一步對 XML 文件結構檢查的方法，它能對一份 XML 文件中的元素或屬性做更明確的定義。以圖２-５-１的 XML 例子來說，若是我們想限制每份文件中只能有一個文件序號，但是作者可能有兩人以上，這些限制條件，我們就能透過 DTD 定義。符合 DTD 規則的 XML 則稱為具合法性（Validate）的，也就是實際可用的 XML 文件，圖２-６為限制圖２-５-１的 DTD 宣告。

```
<!ELEMENT an (#PCDATA)>
<!ELEMENT confidential (#PCDATA | dn)*>
<!ELEMENT description (#PCDATA | sn | an | dt)*>
<!ELEMENT dn (#PCDATA)>
<!ELEMENT document (confidential?, public?, description+)>
<!ATTLIST document
      serial (7000201 | 8000101) #REQUIRED
>
<!ELEMENT documents (document+)>
<!ELEMENT dt (#PCDATA)>
<!ELEMENT public (#PCDATA | dn)*>
<!ELEMENT sn (#PCDATA)>
```

圖2-6 DTD

我們採用 XML 及 DTD 的技術為 XMMI 的基礎技術參考，在 XMMI 架構的 EUID 將會採用 XML 的描述方式，而在嵌入式系統（行動電話）中，採用 DTD 驗證的方式來確定讀入的 XML 檔案為合法的定義。我們採用 XML 為 XMMI 的主要原因有：

- 不必重新設計 EUID 的格式，資料維護容易
- 程式語言大多支援剖析 XML 文件的功能，方便 EUII 的設計
- XML 應用廣泛，容易在不同平台的應用工具上開發

2．3 相關應用技術

在我們採用 XML 設計 XMMI 架構前，已經有許多的 XML 應用，這些應用都是值得我們參考的。在以下的小節我們將介紹可變式向量圖形（Scalable Vector Graphics, SVG）[6]、Winamp 多媒體應用程式的外皮系統（Skin System）[7]及

目前市面上行動電話對於使用者介面能做的調整。

2・3・1 可變式向量圖形（Scalable Vector Graphics, SVG）

SVG 是 W3C 的推薦規範之一，屬於 XML 的子集合，其語法定義皆符合 XML 的規範，SVG 的功能就類似大家所熟悉的 Macromedia Flash 的功能。SVG 以 XML 標籤的方式定義許多強大的向量繪圖功能。向量圖與一般 BMP 或 JPEG 圖形不一樣的地方在向量圖形的原始檔案是一個文字描述檔，描述基本圖形與動畫，透過剖析程式解讀後，再將圖形畫出。SVG 的原始檔案是符合 XML 規範的檔案，所以具有可讀的性質。在人機介面的問題上，GUI 是主要考慮的因素之一，GUI 的部份，XMMI 架構中的 EUID 關於向量繪圖的描述將參考 SVG。

```
<?xml version="1.0"?>
<svg width="128px" height="128px">
<image x="0" y="16" width="128" height="95" xlink:href="./IMGS/wall.gif"/>
<rect x="0" y="111" rx="5" ry="5" width="50" height="18" fill="blue"/>
<text x="7" y="125" font-size="18" fill="yellow" stroke="yellow">MENU</text>
<rect x="77" y="111" rx="5" ry="5" width="50" height="18" fill="blue"/>
<text x="80" y="125" font-size="18" fill="yellow" stroke="yellow">NAMES</text>
<image x="110" y="4" width="16" height="8" xlink:href="./IMGS/battery.gif"/>
<image x="1" y="3" width="16" height="9" xlink:href="./IMGS/signal.gif"/>
<image x="18" y="2" width="13" height="13" xlink:href="./IMGS/msg.gif"/>
<image x="35" y="1" width="13" height="13" xlink:href="./IMGS/ring.gif"/>
<text x="35" y="30">NO-SIG GSM</text>
</svg>
```

圖 2-7 SVG 原始碼

圖 2-7 為 SVG 原始碼的例子，就像 HTML 一樣，SVG 描述了有關畫面呈現的所有資訊，與 HTML 不同的是，SVG 是以一個圖形為主的觀點來看呈現的

畫面，而 HTML 則是以文字為主。以下我們對這個 SVG 所用到的四個標籤功能作簡單的說明：

- <svg>：根原素，定義了 SVG 畫面的大小為 128*128
- <image>：由檔案載入一般圖形
- <rect>：畫方型向量圖
- <text>：關於文字呈現的描述

圖 2-8 為圖 2-7 透過網頁瀏覽器配合 Adobe SVG Viewer 3.0 的展現結果，是模擬行動電話的待機畫面。但是，由於 SVG 的定義過於龐大，使得 SVG 的剖析程式不容易安裝在行動電話這類嵌入式系統中。雖然已經有專為嵌入式系統設計的 Mobile SVG[8]格式，但仍不完全符合在我們架構中的應用，在 EUID 中除了 GUI 需要考慮外，我們還要考慮到有關事件的描述。所以在圖形描述方面，EUID 參考 SVG 的描述方式，但有關事件部份，我們必需自行設計。

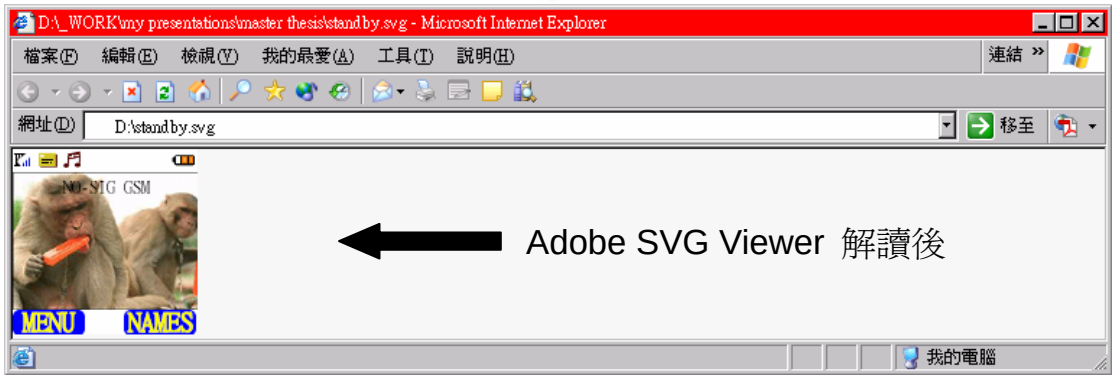


圖 2-8 SVG 的呈現

2・3・2 Winamp 3.0 外皮系統（Skin System）

Winamp 是一個多媒體的播放程式，早期主要是播放 MP3 的應用程式，此應

用程式的特色之一，就是它的 Skin System。Skin System 能夠透過介面檔案的置換而改變應用程式 GUI 的外觀，而不用更改程式碼的機制。Winamp 在 2.0 版時，Skin System 對 GUI 的改變僅止於配色及視窗底圖的改變，在 Winamp 3.0 時，Skin System 已經能對使用者外觀做更多的改變，除了原來的配色，它進一步能改變程式視窗的形狀及控制項的位置。Winamp 3.0 的 Skin System 正是採用 XML 的描述技術。圖 2-9 左為 Winamp 原始的外觀；右為採用不同的 Skin 樣式。

圖 2-9 Winamp Skin System

有關 Winamp Skin System 與 XML 更詳細的說明，請參考：

http://www.winamp.com/nsdn/winamp/skinning/modern/tutorials/2.3-simpleskin.php

2・3・3 行動電話上的使用者介面

行動電話在現代人的生活中，已經是不可或缺的一項通訊設備。這類設備的介面，應該要有能力達到高度的個人化，以符合不同使用者的需求。目前市面上已知的行動電話，對使用者介面的調整功能卻僅止於佈景主題的改變（即介面配色的改變），例如，Microsoft Smartphone Theme Generator 1.0[9]。而我們欲提出

的 XMMI 的架構，除了改變使用者介面的樣式、配色外，更進一步能使用 XML 文件定義的方式，導入行動電話上的事件，進而達到改變行動電話操作的行為模式。

第 3 章

可延伸式人機介面系統

3．1 前言

人機介面（Man Machine Interface, MMI），不管在任何設備上，都佔有極重要的地位。MMI 所包含的範疇很廣，不只是使用者圖形化介面(Graphical User Interface, GUI)而已，更包含了硬體的按鍵、語音輸入、提示鈴聲、…等。MMI 負責與使用者的溝通，包括將機器的狀態回報給使用者，也處理使用者欲執行的命令。

行動電話的功能越來越多，使得 MMI 的設計越趨複雜。目前 MMI 的 GUI 部份是採用類似設計視窗程式的方式，將有可能出現的視窗準備好，讓應用程式的設計者使用。現今的 MMI 架構，行動電話的設計廠商需要不斷的開發新的使用者介面，以應付不同的使用者需求，而這些工作通常是需要大量的時間及人力。在使用者方面，也難以對介面做個人化的修改。

在本篇論文中，我們提出了一個新的嵌入式系統 MMI 架構，我們稱之為可延伸式人機介面系統（eXtensible Man Machine Interface system, XMMI）。採用 XMMI 架構的介面設計，能縮短介面在設計開發階段所耗費的時間及人力，也能讓使用者依照個人對介面及操作方式，做個人化的修改。由於開發 XMMI 的架構，許多原始行動電話中的應用程式需要重新設計，因此，我們在個人電腦上開發一個模擬 XMMI 運作的行動電話架構，實作平台為 Microsoft .NET；程式語言使用 Visual C#。

3．2 XMMI 設計概念

藉由觀察行動電話的硬體裝置，我們發現，“便於攜帶”是行動電話的主要設計因素之一，所以行動電話的畫面通常不會很大，目前市面上常見的行動電話螢幕尺寸為 128x128 及 128x160 像素的規格。在這樣的畫面，能顯示的文字及圖形有限，一次都只能顯示一個應用程式的其中一個視窗畫面（一個應用程式能包含數個視窗畫面），而行動電話畫面的變換，則是由系統或使用者所引起的事件驅動。我們能將行動電話的行為想像成是一個狀態機（state machine），每一個狀態下，繪製相關資訊至螢幕之後，等待事件，準備將狀態改變至另一個狀態點。

行動電話這類的嵌入式系統，不像是一般用途功能的電腦，它們都有著特定的功能。例如，行動電話：我們就會想到它應該有的應用程式有簡訊發送、電話簿、…等功能；個人數位助理（PDA）：應該有行事曆、記事本、…等功能。這些特定的功能，在設計時就會規劃出固定的記憶體儲存該應用程式的資料。簡單的說，這些應用程式只是將記憶體中的資料呈現至螢幕上，而這些呈現的方式，我們能採用類似 XML+CSS 的方法，將呈現方式與資料分開。

3．3 XMMI 架構

圖 3-1 為本篇論文所提出的 XMMI 架構運作，底下我們主要說明的部份是嵌入式系統使用者介面解譯器（Embedded User Interface Interpreter, EUII）、嵌入式系統使用者介面的描述語言（Embedded User Interface Description, EUID）、特定功能應用程式（Specific applications）、一般應用程式（General applications）及圖形化介面應用程式介面（GUI APIs）這五層。至於使用者介面層（UI Layer）、框架（Framework）及作業系統及驅動程式（Hardware drivers and RTOS）這三層

與傳統 MMI 架構的功能相仿，將在第四章做說明。

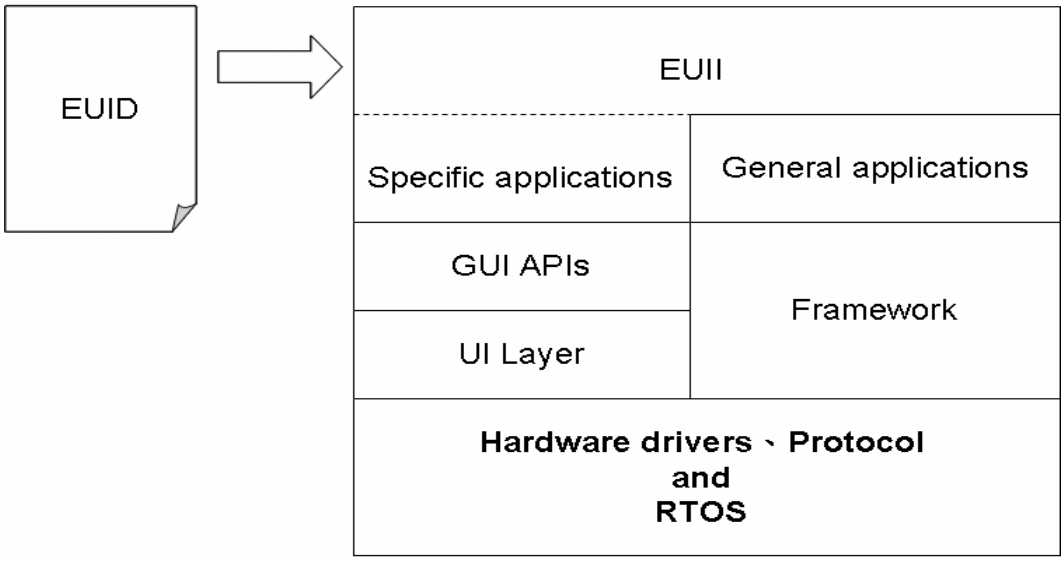


圖 3-1 XMMI 架構

3．3．1 嵌入式系統使用者介面的描述語言（EUID）

在 XMMI 的架構下，我們將行動電話的螢幕當成是一個繪圖區，將需要呈現的資訊，繪製到螢幕上。EUID 的其中一部份是描述畫面所需要呈現的圖形、文字、…等資訊；另一部份則是描述在該畫面下，受不同事件觸發後，行動電話進入不同的視窗畫面。

以狀態機的角度來看，圖 3-2 中的 EUID 1、EUID 2 及 EUID 3 分別代表了行動電話中的三個不同的狀態。假設行動的待機畫面是 EUID 1，則 EUID 1 中的畫面 1 描述了行動電話在待機狀態螢幕所呈現的資訊，我們採用類似 SVG 的描述方式達成；事件 1 的部份則是描述在待機狀態下對事件發生時，所需要做的相對反應。例如，在待機畫面會出現的訊號強度圖示、電池強度圖示、…等，描述在畫面 1 的部份，當行動電話待機狀態下，我們按下電話鍵盤上的左功能鍵時，要進入的是主選單畫面，按下左功能鍵的事件反應，是描述在事件 1 的部

份。我們設計的 EUID 不僅描述了畫面的呈現，也描述整個系統的運作流程。

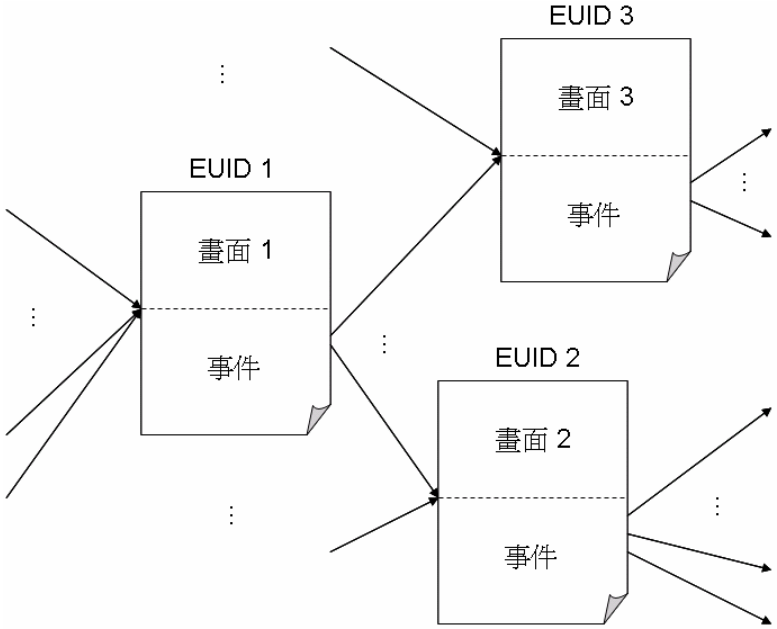


圖 3-2 行動電話狀態

3・3・2 嵌入式系統使用者介面解譯器（EUII）

EUII 的核心部份是 XML 的剖析器，其設計是針對 EUID 的描述作對應的解讀，將畫面實際繪製到行動電話的螢幕上。EUII 能掌握所有行動電話上的事件及狀態（status）。以圖 3-2 的例子做說明，EUII 解讀待機畫面的 EUID 1 會依據行動電話的狀態畫出指定的畫面。當畫面繪製完成後，EUII 便等待事件發生，假設這個時候訊號強度減弱一格，則 EUII 會重畫螢幕上有關訊號強度的圖示，當使用者按下左功能鍵時，EUII 便尋找 EUID 中有關左功能鍵的描述，而對行動電話畫面做相對的改變，如果這時候按下左功能鍵是進入主選單畫面的話，則 EUII 會載入描述主選單畫面的 EUID，再依 EUID 的描述，繪出主選單畫面。

從 EUII 及 EUID 開發介面的功能來看，EUII 實際執行行動電話的螢幕繪製，及功能執行；而所有 EUII 的運作，是來自 EUID 的描述。因此，我們只要改變

EUID 的描述，行動電話的畫面及操作方式，便會變的截然不同。

3．3．3 特定功能應用程式（Specific applications）

嵌入式系統通常都會有一些特定的功能，行動電話的特定功能有電話簿、簡訊發送、撥打電話、…等。在傳統的 MMI 架構，這些特定的應用也是以類似視窗程式的方式撰寫，如此一來功能性的動作便與呈現方式難以分開。在 XMMI 的架構中，這些特定的功能，我們將它的資料或執行後的結果，存放在一段固定的記憶體結構中，在需要呈現至螢幕上時，由 EUII 從記憶體中取出資料，配合 EUID 所描述的呈現方式，再將資料顯示在行動電話螢幕上。

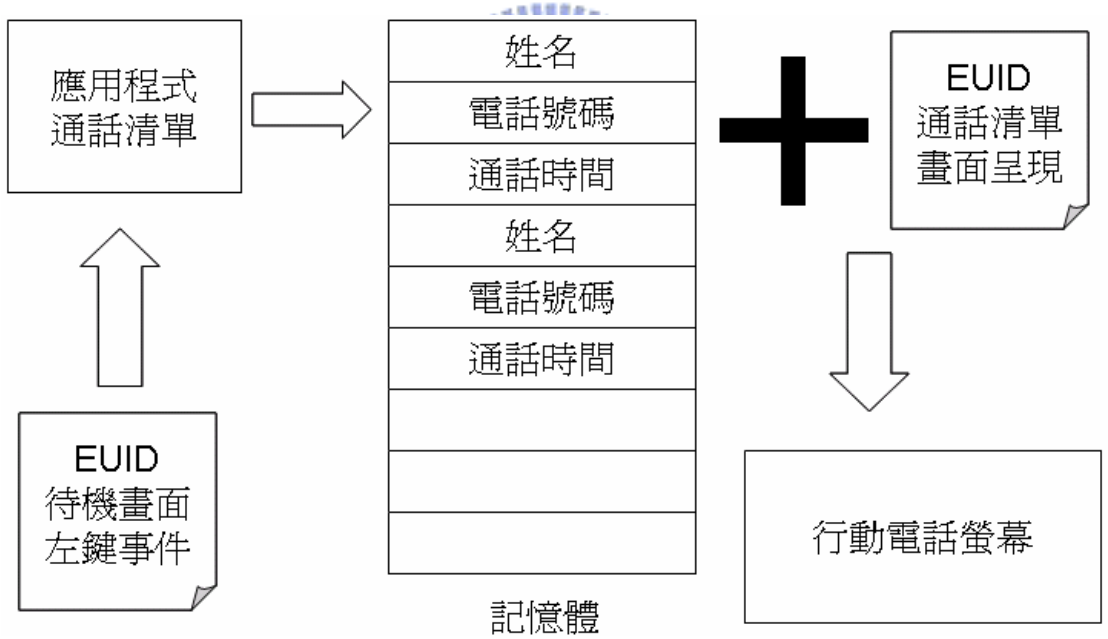


圖 3-3 特定應用程式運作流程

圖 3-3 說明了特定應用程式的運作流程。圖左下角待機畫面 EUID 描述了行動電話按下左鍵的行為是進入通話清單的展示畫面，EUII 首先會執行通話清單的應用程式，通話清單應用程式將手機的通話清單填入一段固定的記憶體裡之後，EUII 才將描述通話清單的 EUID 載入並展示於行動電話螢幕。這個作法，

有關通話清單的展示方式，就能從功能性的程式碼中分離出來，轉而寫在 EUID 檔案中。

3・3・4 一般應用程式及圖形化介面應用程式介面

行動電話的功能越來越多樣化，有些功能需要能直接在螢幕上呈現，例如：照相機、攝影機、Java 遊戲、…功能。這些應用程式，我們將它歸類為一般應用程式。這些程式的開發可以直接使用圖形化介面應用程式介面（GUI APIs）所提供的 GUI 元件，不過關於一般應用程式的執行時機，仍是定義在 EUID 中。

由於行動電話的所有畫面已交由 EUID 描述達成，在 XMMI 架構與傳統 MMI 架構下最大的不同在於 XMMI 架構下的 GUI APIs，所提供的 GUI 元件並不需要包含視窗畫面的 APIs，只需要提供如按鈕（button）、捲動軸（scroll bar）、浮動視窗、…等元件。

3・4 XMMI 模擬器

要實現 XMMI 的架構，我們在 3・3・3 節所提出的特定應用程式（Specific applications）必需要重寫，然而 XMMI 的設計重點是在於 EUII 及 EUID。為了快速對 EUID 配合 EUII 的方式做實驗，我們在 PC 上實作了一個行動電話採用 XMMI 架構的模擬程式，藉此也定義出 EUID 的雛形。

本篇論文主要探討的部份在於 XMMI 架構使用程式語言實現的可行性，尚未考慮有關嵌入式系統的記憶體限制、及行其運算能力。XMMI 的模擬器架構如圖 3-4，與圖 3-1 所不一樣的地方只有底層作業系統、應用程式介面（APIs）及框架層（Framework），這部份我們使用 Microsoft .NET 搭配 GDI+ 繪圖函式

庫，程式語言使用 Visual C#。上層的 EUII 及特定應用程式（Specific applications）在設計完成後，若要在實際的行動電話上實現，只需要配合硬體平台的需求以適當程式語言改寫，而 EUID 可以直接使用。

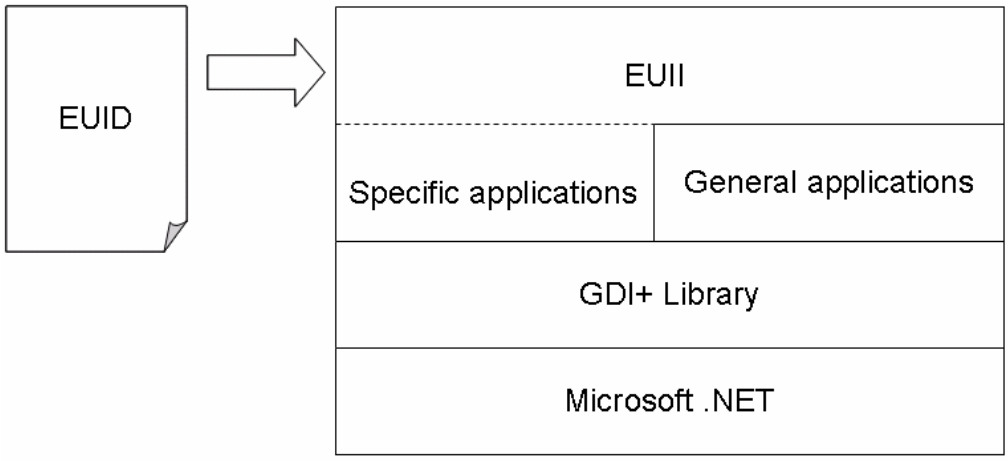


圖 3-4 XMMI 模擬器

3・4・1 設計 EUID

EUID 在 XMMI 的架構中扮演了描述行動電話狀態機的重要角色，EUID 採用符合 XML 規範的語法設計。如圖 3-5 為 EUID 的樹狀圖，根結點為<euid>，根元素的兩個子結點為<this>及<next>。<this>的子結點是描述行動電話現態的螢幕上所需要呈現的畫面；而<next>的子結點主要是定義行動電話在現態受事件觸發後，所需要做的相對反應。在圖中我們發現不論在<this>或<next>的子結點中都會出現<animation>、<image>、…等葉子（leaves）結點，這些結點的功能主要是描述繪圖的動作，我們主要是參考 SVG 向量繪圖的描述方法實作。

以行動電話待機畫面的例子說明，在待機畫面下的背景圖及一般圖示，就定義 leaves 1 裡，而訊號強度資訊，則是由<status>的描述配合行動電話實際收訊的狀況，繪製 leaves 2 所定義的圖形。

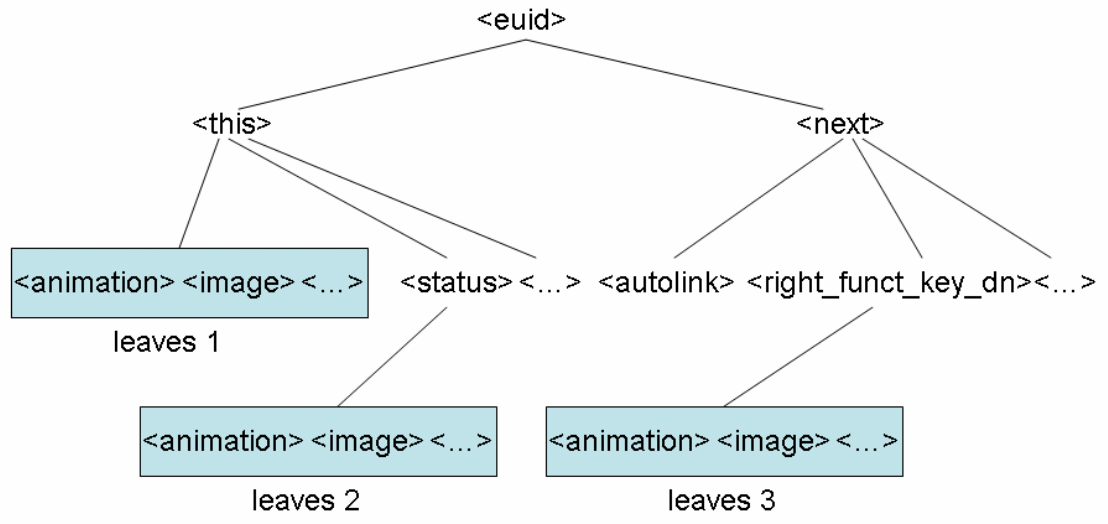


圖 3-5 EUID 樹狀圖

3・4・2 設計 EUII 及 XMMI 的運作流程

EUII 中的 XML 剖析器，我們採用 Microsoft .NET 所提供的 XmlDocument 類別實作，XmlDocument 類別是一個 XML DOM 剖析器。EUII 的設計，就是為了對 EUID 做相對應的解讀，當我們增加新的 EUID 標籤，我們也要在 EUII 加入解讀該標籤程式碼。

圖 3-6 為 XMMI 運作的流程圖，在載入 EUID 後，EUII 首先依據 EUID 裡的<this>區塊定義，繪製相關的訊息及圖示到行動電話的螢幕上，完成螢幕的繪製後，EUII 進入傾聽事件迴圈，當觸發事件發生，EUII 到 EUID 的<next>區塊中，尋找對應事件的處理方式。事件的發生有兩種可能，一種是不改變行動電話的現態，我們不需要載入一個新的 EUID，對畫面的改變仍是描述在現態的<this>區塊中，例如收訊強度的改變；另一種是讓行動電話進入另一個狀態的事件，這種事件發生時，EUII 會載入代表行動電話另一個狀態的 EUID，再重新對被載入的 EUID 做螢幕繪製，例如按下行動電話的右功能鍵進入主選單畫面。

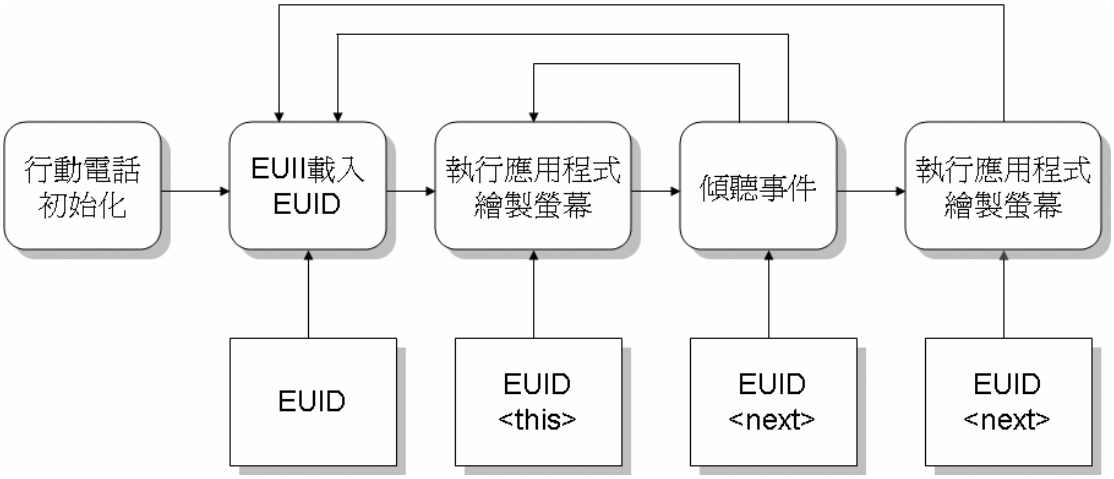


圖 3-6 XMMI 運作流程圖

3．5 行動電話模擬器

圖 3-7 為行動電話模擬器以 XMMI 架構實作，展示待機畫面的情形，這個待機畫面的 EUID 為圖 3-8 的 XML 檔案片段。EUID 中描述的<image>代表目前的桌布圖案，<m_text>其中的 source 屬性值為 network_name，表示這個字串的來源是行動電話記憶體中的系統商名稱，EUII 解讀後，會呼叫特定應用程式（specific applications），依照行動電話設定的語言模式，將系統商的名稱填入記憶體，而<m_text>的其他屬性就是描述有關系統商名稱字型的呈現，這樣的做法就能把呈現方式分開。<this>區塊中還包一有<status>標籤，其中的 type 屬性值為 message，代表當行動電話有未讀訊息的時候，需要展示什麼樣的資訊在螢幕上，在這個例子中，當有未讀訊息的時候，會秀出一張 message.png 的圖片在螢幕上。在<next>區塊中的<right_function_key_up>標籤，代表當行動電話在收到右功能按鍵事件的時候，所需要做的處理。當電話按下右功能鍵後會進入主選單畫面，其動作為載入 main_menu.xml。

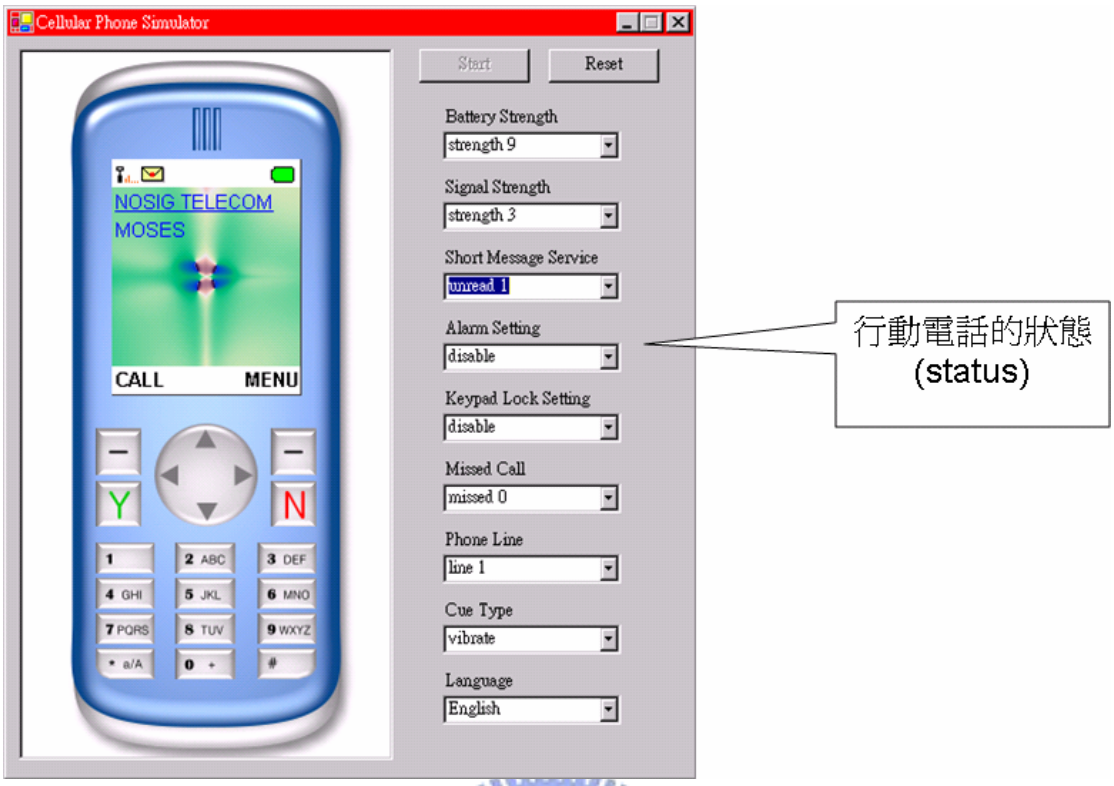


圖 3 - 7　行動電話模擬器

```
<?xml version="1.0" encoding="utf-8"?>
<euid alias="idle_screen">
<this>
        <image x="0" y="0" w="128" h="160" source="./wall.png"/>
            ⋮
        <m_text x="0" y="21" w="128" h="16" font="Arial" style="Underline"
                size="10" color="0,0,255" source="network_name"/>
            ⋮
        <status tpye="message" min="1" max="9">
                <image x="20" y="5" w="16" h="11" source="./message.png"/>
                    ⋮
        </status>
            ⋮
</this>
<next>
        <right_function_key_up linkto="./main_menu.xml"/>
            ⋮
</next>
</euid>
```

圖 3 - 8　待機畫面的 EUID

圖 3-9 為當行動電話的電池強度，與語言模式改變的時候，反應在螢幕上的效果。

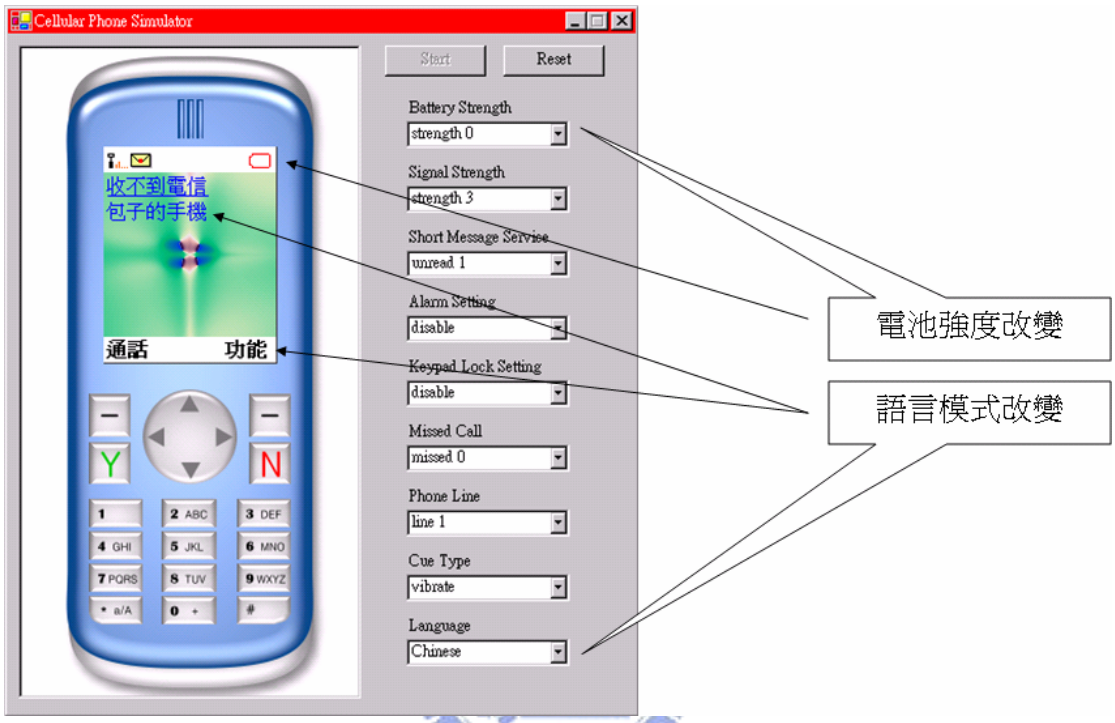


圖 3-9 行動電話狀態（status）改變

第 4 章

被動式 EUII 設計與實作

4．1 前言

我們在現今的行動電話開發平台上設計了被動式的使用者介面解譯器，簡稱 EUII，被動式 EUII 需要傳統的 MMI 驅動。我們把使用傳統 MMI 架構設計出來的行動電話的 GUI 樣式以 EUID 描述，藉此方式改變使用者介面。建構被動式 EUII 的平台處理器採用 Arm-7，嵌入式作業系統以 Nucleus 為主，實作程式語言為 C。

4．2 MMI 架構簡介

圖 4-1 為現今 MMI 的運作情形，底下我們以分層的方式介紹 MMI 架構及說明這樣架構下的應用程式的運作。

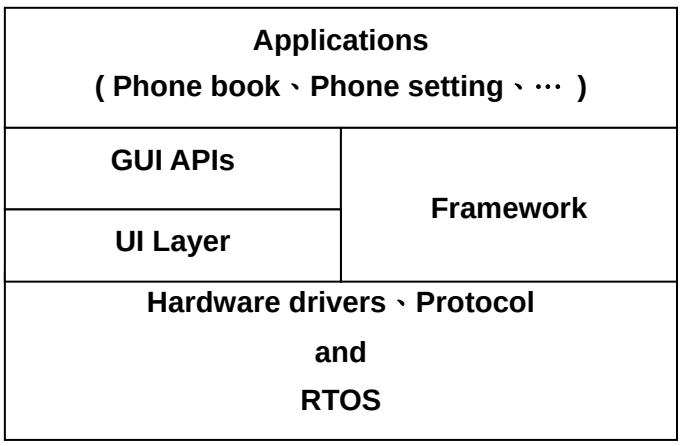


圖 4-1 MMI 架構

● 作業系統及驅動程式（Hardware drivers and RTOS）：

這一層包含了硬體結構及其驅動程式、通訊協定實作及作業系統。在硬體結構有液晶顯示器（LCD）、記憶體（RAM、ROM）、硬體按鍵、…等。行動電話的通訊協定上有 GSM（Global System for Mobile communication）、GPRS（General Packet Radio Services）、…等。作業系統則負責工作排程、裝置管理及記憶體管理的工作。

● 框架（Framework）：

Framework 層提供了系統呼叫（System call）函式，並且負責包裝事件，處理事件發生後執行的回呼函式（Call-back function）。這些事件的種類包括：通訊協定事件、硬體事件、時間事件、按鍵事件、…等等。

● 使用者介面層（UI Layer）：

UI Layer 包含了繪製圖形的功能，最基本的繪圖元素就是螢幕上的一個像素（pixel）及其顏色。其他還包含基本圖形，如：直線、矩形、圓、文字和圖片的載入及顯示函式。

● 圖形化介面應用程式介面（GUI APIs）：

GUI APIs 層利用 UI Layer 所提供的基礎繪圖功能和 Framework 的事件處理模型，組合成圖形介面的元件，例如：按鈕（button）、捲動軸（scroll bar）、浮動視窗及其他特效介面元件，例如：佈景主題（theme）的特效介面。除了這些元件外，也提供已規畫完成的應用程式視窗（相當於 Java 程式語言中的 JFrame

類別)，這些“視窗”包含了所有行動電話中有可能出現的樣式，通常這些視窗常會多達三、四百種。一個視窗的完整定義應該包含三種：

- ○ 入口函式：將應用程式欲展現在視窗內的文字、圖片填入視窗結構中；註冊事件的回呼函式。
- ○ 重繪函式：實際執行視窗的重繪。
- ○ 離開函式：這個函式是由下一個要顯示的視窗呼叫。

- ● 應用程式層（Applications）：

程式設計師透過 GUI APIs 及 Framework 層所提供的函式開發各種應用程式，應用程式之中，可再分成三種：

- ○ 一般應用程式：電話簿、行事曆、遊戲、…等。
- ○ 通訊應用程式：通話、簡訊、電子郵件、…等。
- ○ 特殊應用程式：WAP、J2ME、智慧型輸入法、…等。

下一節我們舉一個實際的例子說明有關 GUI APIs 及應用程式間的操作關係。

4．3 MMI 運作實例

考慮底下的情形：行動電話在待機狀態時使用者按下左功能鍵使得行動電話進入主選單畫面，在這個時候接使用者收到來電訊息，使用者結束通話後，行動電話自動再度進入主選單畫面，在這樣簡單的狀況下，MMI 實際的運作將會牽涉圖 4-2，4-3 及 4-4。底下我們逐項說明流程。

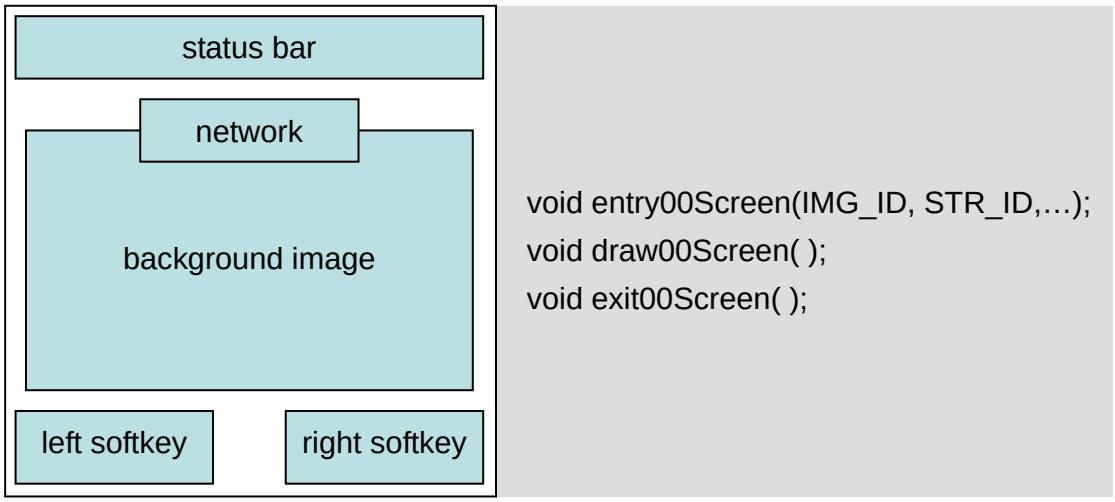


圖４-２ 待機畫面

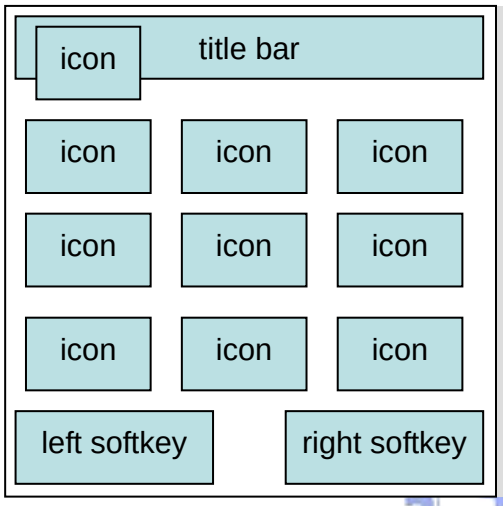
行動電話在完成開機及網路連結以後，MMI 會自動導入待機畫面，圖４-２為待機畫面的示意圖，待機畫面包含了狀態列（status bar）、系統服務商名稱（network）、背景圖（background image）、左右軟鍵（left softkey, right softkey）。在 MMI 進入待機畫面時，會先呼叫函式 entry00Screen(IMG_ID, STR_ID,…)，此函式裡的主要功能包括回復前一次執行資訊（稍後解釋）、註冊事件及定義畫面的顯示。

傳入的參數 IMG_ID 代表各個 GUI 元件所需要顯示圖的來源；參數 STR_ID 是代表待機畫面下顯示文字的來源。上圖待機畫面中，需要傳入的 IMG_ID 及 STR_ID 的只有左軟鍵及右軟鍵。因此在待機畫面左軟鍵的提示文字可以彈性的更改為“功能”；右軟鍵的提示文字則可以彈性的更改為“電話簿”。其他待機畫面上的背景圖、系統服務商名稱、狀態列都無法提供開發者在此處修改，例如：當開發者想要將狀態列換成垂直並在右邊擺放，就需要到有關狀態列元件設定的地方去更動相關的程式碼。

另外，入口函式的註冊事件部份，是以函式指標（function pointer）的方式

完成。在這個例子中：按下左功能鍵（硬體鍵盤）的事件發生後，在入口函式中需定義相對處理函式的指標為指向進入主選單功能前需相對應的函式。

在入口函式執行完之後，MMI 會執行 draw00Screen( )繪製函式，這個函式便將定義好的待機畫面，顯示在行動電話的顯示器上。




```
void entry01Screen(IMG_ID, STR_ID,…);
void draw01Screen( );
void exit01Screen( );
```

圖 4-3 選單畫面

當使用者在待機畫面按下左功能鍵以後，行動電話的狀態便進入主功能選單畫面，主選單畫面的入口函式功能跟待機畫面的入口函式功能相似：設定文字、圖片及註冊事件，當然，依畫面的不同，傳入設定參數也不盡相同。當使用者看到主選單畫面顯示後，將選項移至正中央的圖示上。這時，使用者突然收到一個來電訊息，行動電畫的顯示畫面便自動切換至圖 4-4 的來電畫面。

在任何畫面的入口函式之前，會先呼叫上一個畫面的離開函式。離開函式的主要功能為紀錄畫面的執行後的資訊，以這個例子來說，使用者在主選單畫面將選項移至正中央的圖示時，突然收到一個來電訊息，在呼叫來電畫面的入口函式之前，MMI 會先呼叫主選單畫面的離開函式，紀錄目前選項的位置，待使用者

通話完畢要回到主選單畫面時，再將這個資訊取回。這樣的情況在行動電話常常發生，例如：當我們在待機畫面輸入電話號碼時，有來電訊息進入，通話完畢後，回到的待機畫面，應該保有通話前輸入的電話號碼。

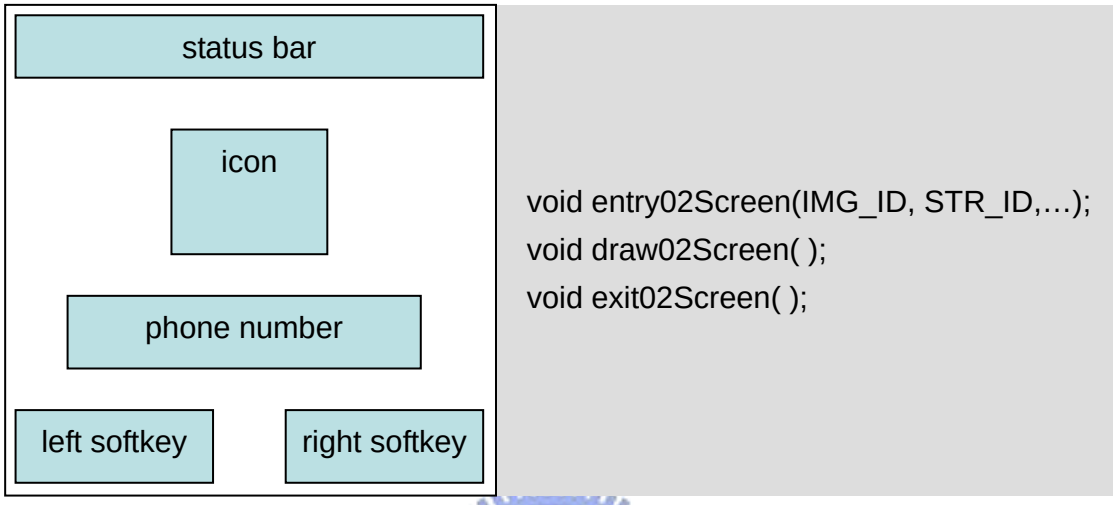


圖 4-4 來電畫面

4．4 被動式嵌入式系統使用者介面解譯器（EUII）設計

在介紹過現今行動電話的 MMI 系統架構及其運作流程後，在這一個小節將說明論文中所提出的被動式 EUII 在此 MMI 架構下的定位所在，請參考圖 4-5。

有限嵌入式系統的資源（如，記憶體）一般是依據特定的應用，分配給適當大小的記憶體空間。在目前的一般功能的行動電話上，記憶體容量分配最多的是與來電鈴聲相關的應用程式（大部份行動電話都有提供使用者下載鈴聲功能）。EUII 是一個新的應用程式，需要記憶體存放 EUID 的描述檔案。在我們不特別為它準備記憶體空間的情況下，只有 2k bytes 的空間能夠給予 EUID 使用。本論文以不改變系統原有的環境下為基礎，設計及開發 EUII。重點在將行動電話的待機畫面的 GUI 導向改由 EUID 描述。這樣的選擇是因為待機畫面是最適合個

人化介面的調整。

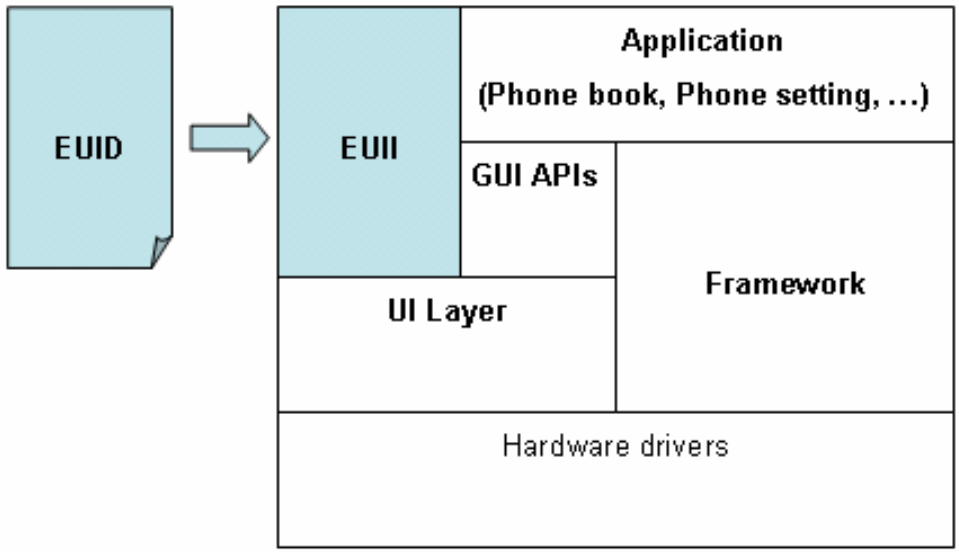


圖４-５ 被動式 EUII

本實驗中的 EUII 能控制所有待機畫面中出現的 GUI 元件在畫面上的配置。EUII 的核心部份為 XML 剖析器，和所有嵌入式系統的應用程式一樣，EUII 也面臨著程式體積（code size）、執行效能與執行時所需記憶體的問題。

４・４・１ EUII 資料結構

EUII 中我們定義了兩個描述待機畫面的結構（structure），圖４-６是紀錄有關畫面上所有 GUI 元件的座標；圖４-７是紀錄使用者能佈景主題（theme）配色的定義。採用結構的方式，能讓我們對記憶體的使用更容易掌握（由作業系統靜態分配），此外也提高了系統的可靠度。

```
typedef struct _X_icon_alarm { int x, y; } X_icon_alarm;
typedef struct _X_icon_battery { int x, y; } X_icon_battery;
typedef struct _X_icon_date { int x, y; } X_icon_date;
                    ⋮
typedef struct _entry00Widgets
{
        X_icon_alarm XML_icon_alarm;
        X_icon_battery XML_icon_battery;
        X_icon_date XML_icon_date;
        X_icon_gprs XML_icon_gprs;
                    ⋮
} entry00Widgets;
```

電池圖示的顯示座標

有關待機畫面的GUI圖示

圖 4-6 EUII 有關 GUI 圖示及元件的結構定義

```
Filled_region LSK_down_PS =
{
FILLED_AREA_TYPE_COLOR,
IMAGE_ID_NULL,
NULL,
{26,72,181,100},
{0,0,0,100},
{0,0,0,100},
{0,0,0,100},
0};
```

UI Layer的著色結構

UI Layer的顏色定義

圖 4-7 有關佈景主題配色定義（左軟鍵按下時的顏色）

4・4・2 XML 剖析器及 EUID 設計

XML 剖析器的選擇在實際嵌入式系統中是一個重要的議題：

- XML DOM 剖析器：在讀入 EUID 檔案時，只啟動開檔動作一次，所以速度較快，但缺點在 DOM 建出來的資料結構，耗費大量記憶體。
- XML SAX 剖析器：雖然不會建立耗費記憶體的資料結構，但在每次進入需要讀取 EUID 的畫面時，都要重新剖析 EUID，速度慢。

由於傳統的 XML 剖析器都有不適用於嵌入式系統的地方，我們實作了一個專為 EUII 設計的 XML 剖析器。我們使用的 XML 剖析器配合 EUID 的設計（固定的描述格式），在執行時只需要兩個“char”便能完成剖析的工作，之後將結果設定到 EUII 的資料結構中，如此一來，XML 剖析器只要在 EUII 功能啟動時執行一次，改善了效能的問題。

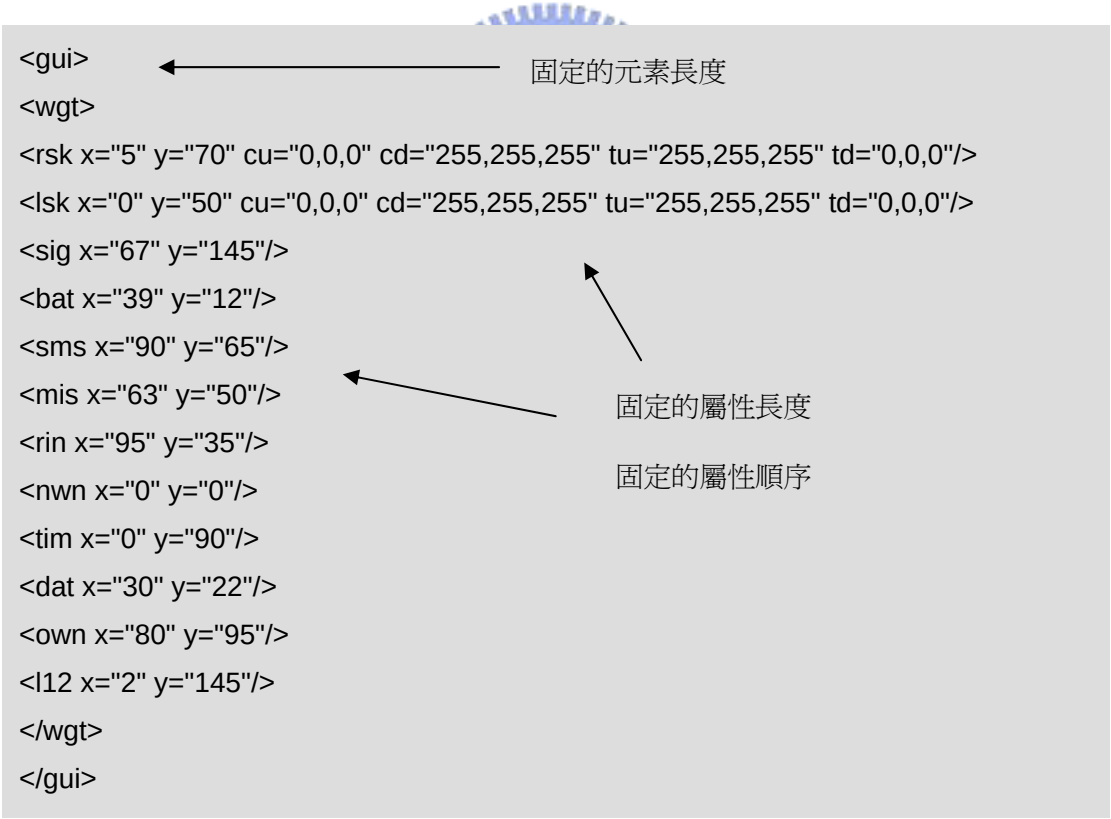


圖 4-8 固定格式的 EUID 檔案

以“固定格式”的 EUID 檔案，雖然犧牲了 XML 的自由度和可讀性，不過

在將 GUI 導入由符合 XML 定義的 EUID 描述之後，便能在桌上型電腦開發整合性圖形化介面開發工具，我們並不需要實際撰寫 EUID 的 XML 檔案。

4．5 EUII 在 MMI 中的運作流程

透過 EUID 的描述，行動電話上的使用介面樣式便可透過 WAP、藍芽、紅外線傳輸、…等方式，載入行動電話。我們以圖 4-9 說明 EUII 在行動電話 MMI 中的運作流程。當使用者啟動 EUII 的時候，圖 4-9 的左邊的流程只會執行一次。右邊的部份說明我們在原來行動電話顯示畫面中三個主要的函式所做的修改。

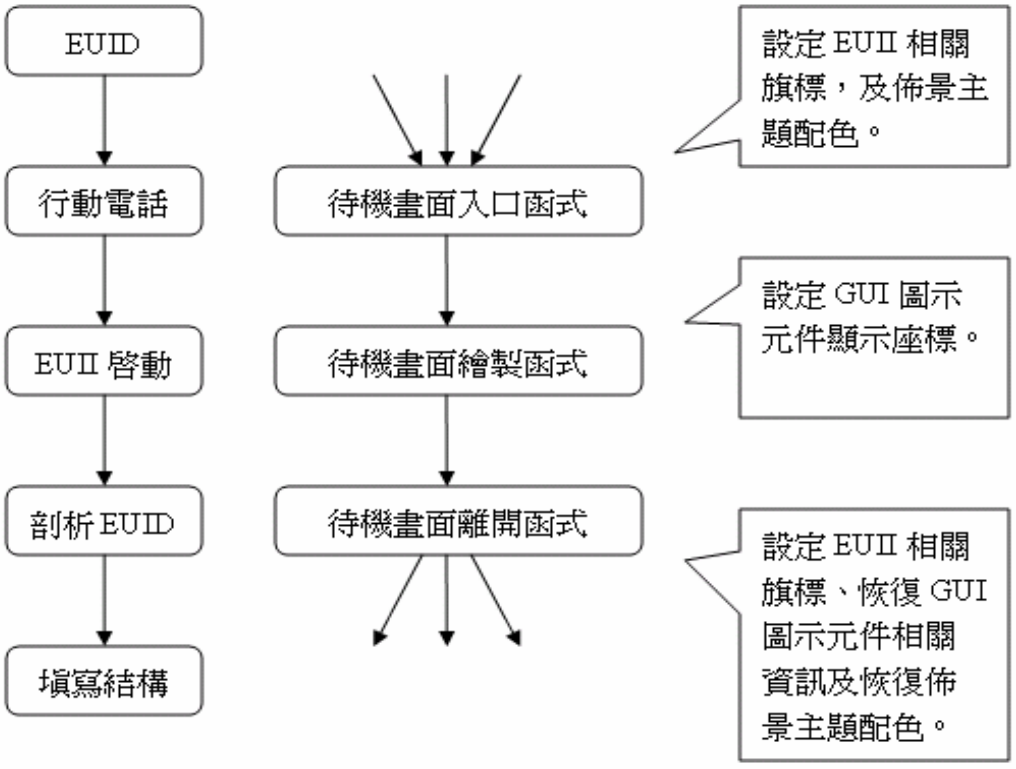


圖 4-9 EUII 運作流程

由於我們實作的平台本身的佈景主題配色是由獨立的應用程式運作，在繪圖函式中並不會有關於佈景主題配色的資訊，因此我們在待機畫面的入口函式中，

先將佈景主題配色設定的指標，指向由 EUII 設定的結構。

在待機畫面的繪製函式中，將畫面中的 GUI 圖示元件座標資訊由 EUII 的資料結構取得並且畫出。當繪製函式執行結束後，由 EUID 所定義的 GUI 樣式便顯示在螢幕上。

因為有關左右軟鍵的座標及佈景主題配色的資訊，在 MMI 中是屬於全域（global）的資料結構，所以當行動電話進入別的狀態畫面的時候，我們必需將其所有資訊回復。這些回復的動作，我們利用 MMI 在切換狀態畫面時一定會先呼叫上一個畫面的離開函式的特性，把所有復原 GUI 樣式的動作，寫在待機畫面的離開函式裡。

4．6 實驗結果

圖 4-10 是我們將 EUII 在行動電話上實現的結果，目前所使用特定的 XML Parser 佔用 Flash ROM 1460 Bytes，執行期使用 264 Bytes RAM。在行動電話上操作，並不會有效能不佳或記憶體不足的問題。

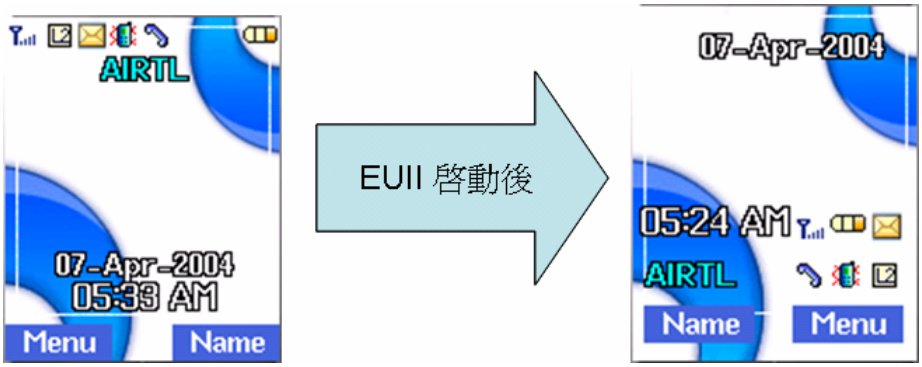


圖 4-10 平台實現

第５章

結論及未來工作

５．１ 結論

一個良好的人機介面系統需要具備使用者友善度（friendly），嵌入式系統的應用越來越廣，各家廠商所生產設備的操作模式不盡相同。我們認為一個好的使用者介面，也應該具備有使用者定義的機制，讓使用者能夠定義最符合自己習慣的操作模式。

本論文所提出的 XMMI 架構，利用 XML 描述狀態機（state machine）的概念，將嵌入式系統的介面及操作行為，描述在我們所設計的 EUID 裡。這樣的做法不僅能達成使用者介面的個人化定義，也能使嵌入式系統的設計廠商為不同的行動電話，快速的設計不同的介面。例如，在２．１節所描述的故事裡，若是行動電話的開發廠商採用 XMMI 的架構，所有的問題，只需要透過修改 EUID 的描述就能解決。

５．２ 未來工作

未來對於 XMMI 架構的發展，以下四點說明：

一、設計更完整的 XMMI 流程：

完整的 XMMI 除了 EUID 要對嵌入系統的所有功能及事件有完善的定義之外。為提高 XMMI 架構的系統穩定性，驗證合格的 EUID 檔案也是 EUII 的重要工作之一。

二、嵌入式 XML 剖析器：

嵌入式系統裡的運算能力及記憶體空間遠不及一般個人電腦，我們應開發一個適合 XMMI 架構的嵌入式系統 XML 剖析器。

三、其他運作方式的思考：

為了提昇 XMMI 的效能，我們也提出了另一個實作 XMMI 的方式。在本篇論文中的 XMMI 屬於即時的運作方式，每次需要繪製畫面或是事件觸發時，EUII 都要在 EUID 中搜尋對應的處理方式。我們也提出一個非即時的運作方法：在一般電腦平台實作一個 EUII，在讀取 EUID 之後，輸出對應的程式碼，這些程式碼就能透過編譯、連結成可執行程式，再載入行動電話中，以提昇效能，但此種方式，只適合在行動電話設計者使用。

四、整合設計開發環境：

另一個工作是在一般電腦平台開發一個整合設計環境，透過圖形化 的方式讓系統開發者或使用者設計 EUID 描述檔，以減少人工撰寫 EUID 檔案可能發生的錯誤。

參考文獻

[1] 陳漢儀,“嵌入式系統導覽”, http://www.study-area.org/linux/embedded/articles/Embedded_Linux_Introduction/Embedded_System_Introduction.html, 17 July 2001

[2] James A. Landay and Todd R. Kaufmann, “User Interface Issues in Mobile Computing”, Appeared in the Proceedings of the Fourth Workshop on Workstation Operation Systems, Napa, CA, October 1993

[3] Jacob Eisenstein, Jean Vanderdonckt and Angel Puerta, “Applying Model-Based Techniques to the Development of UIs for Mobile Computers”, Proceeding of the 6th international conference on Intelligent user interfaces 2001

[4] XML Core Working Group, “Extensible Markup Language (XML) 1.0 (Third Edition)”, http://www.w3.org/TR/2004/REC-xml-20040204/, 04 February 2004

[5] Eve Maler, “Guide to the W3C XML Specification DTD Version 2.1”, http://www.w3.org/XML/1998/06/xmlspec-report.htm, June 1998

[6] SVG Working Group, “Scalable Vector Graphics (SVG) 1.1 Specification”, http://www.w3.org/TR/2003/REC-SVG11-20030114/, 14 January 2003

[7] Ken Chen, “WINAMP3: SKINING TUTORIAL WEBPAGE”, http://www.winamp.com/nsdn/winamp/skinning/modern/, 2002. 09. 19

[8] SVG Working Group, “Mobile SVG Profiles: SVG Tiny and SVG Basic”, http://www.w3.org/TR/2003/REC-SVGMobile-20030114/, 14 January 2003

[9] Microsoft, “Smartphone 2002 Theme Generator”, http://www.microsoft.com/windowsmobile/resources/downloads/smartphone/themegenerator.mspx