

Android 手持裝置於適地性應用服務之電源管理機制

| | | | |
|--|---|--|--|
| 李建緯 | 施再繁 | 蔡謝輝 | 張怡君* |
| 朝陽科技大學 資訊與通訊系 助理教授 lijw@cyut.edu.tw | 朝陽科技大學 資訊工程系 助理教授 tfshih@cyut.edu.tw | 朝陽科技大學 資訊工程系 s10027607@cyut.edu.tw | 弘光科技大學 資訊工程系 助理教授 changyc@sunrise.hk.edu.tw |

摘要

Android 智慧型手持裝置與無線通訊技術的興起，行動應用服務的需求量大增。行動用戶能夠藉由定位技術的輔助來獲得更符合用戶地域性的需求，這類應用服務稱作適地性服務 (Location-Based Service; LBS)。然而手持裝置上相關網路通訊元件長時間運作容易產生電源快速消耗的問題，因此，如何提供元件有效的工作週期與休眠週期，進而延長手持裝置的使用時間，是值得探討的研究方向。本論文採用動態電源管理技術，設計一套高電源與低電源的策略演算法，針對不同電量階段執行合適元件開關時間，有效延長手持裝置的使用時間。

關鍵詞：動態電源管理、Android、智慧型手持裝置、適地性服務、無線通訊技術。

1. 前言

近年來，在資訊科技不斷的進步之下，人類取得資訊的途徑不再侷限於桌上型電腦或是筆記型電腦，智慧型手持裝置不論是平板電腦 (例如：iPad、Android Pad) 或是智慧型手機 (例如：iPhone、Android Phone) 搭配 3/3.5G、Wi-Fi、WCDMA... 等無線通訊技術，讓行動用戶能夠隨時隨地運用手上的裝置連上網際網路進行資料的下載與搜尋。為了滿足用戶在行動應用服務上的各種需求，手機裝置上的感測與通訊元件頻繁地與應用服務搭配運作，像是 GPS 感測器能夠執行手持裝置在電子地圖上的定位功能，以及 Wi-Fi 與 3G 通訊元件提供無線網路的功能... 等需要位置資訊來輔助提供更適合使用者需求的應用服務，稱為適地性服務 (Location-Based Service; LBS)。不僅目前熱門的 Facebook 地標打卡功能屬於 LBS，還有許多廣泛的應用方式，而依據服務屬性大致可分為定位追蹤、地圖導航、查詢定位資料、社群交友推播式廣告與遊戲。上述這些應用服

務固然能夠帶來許多便利性，相對的開啟這些感測元件所消耗的電源也非常可觀，這對於需要長時間使用手持裝置的行動用戶是不可忽略的問題。

因此，本研究為了解決 LBS 所造成電源快速消耗的問題提出了一套節能方案，將透過監控應用程式介面 (Application Program Interface; API) 來獲取感測與通訊元件的工作狀態與參數，並根據不同電源消耗階段設計出高電源管理策略與低電源管理策略。在高電源管理策略當中，行動用戶能夠根據目前的電量消耗階段計算出相對的電源查詢週期，並且偵測相關感測與通訊元件是否能夠正常運作，若無法正常運作則將感測元件關閉進入休眠週期，降低元件閒置時所造成的功率消耗；而在低電源管理策略當中，除了高電源管理策略能夠做到的節能方案，我們另外考慮加入元件執行週期，限制感測元件能夠持續運作的時間。在這樣高/低電源策略的搭配之下，除了有效的降低感測元件閒置時的功率消耗，更重要的是能夠延長手持裝置的使用時間，滿足行動用戶在 LBS 上廣泛的應用。

本篇論文的結構如下，第二章介紹國內外相關電源管理設計的研究，並舉出相關實例討論；第三章則對於 Android 電源管理架構以及低功率消耗機制進行詳細的說明；第四章則是針對手持裝置的電量消耗階段設計高電源與低電源策略演算法；第五章為電源策略演算法執行後的效能評估，並做出分析與討論；第六章為結論與未來研究方向。

2. 相關研究

對於近期智慧型手持裝置的電源管理方案，許多國內外學者已提出相關的研究論文與實驗討論。Brock 等學者[1]提出透過監控手持式裝置元件的狀態，將所收集到的各種不同的狀態資訊進行計算，並選定合適的電源管理策略執行電源管理工作。Shih 等學者[2]提出一種

混合式電源動態管理策略，針對手持裝置感測元件的活動或非活動狀態進行電源管理。

Bellavista 等學者[3]提出一個獨創的解決方式讓行動用戶端的手持裝置在大都市環境當中扮演無線感測網路資料的接收者並傳送給後端進行資料處理。李學者[4]針對 Android 手持裝置各元件的功率消耗進行調整，並持續的更新電源管理策略，控制工作中或閒置中的元件開關。方學者[5]針對手持裝置嵌入式系統實作一個使用適應性學習樹預測之動態功率管理器，藉由預測元件閒置時間與功率狀態來調整裝置開關，實驗結果顯示經由功率管理的系統有較佳的省電效果。Chiasserini 學者[6]開發一種電池供電的設備透過不同元件的工作與睡眠週期來權衡能源消耗以及延遲時間，並計算出合適的元件喚醒時間進行電源管理的節能動作。Holland 等學者[7]提出了動態頻譜接入技術，將電源節能轉移給其他介面做使用，實驗結果也表明能夠有效降低非必要的元件電源使用功率。Chen[8]學者提出優化動態電源管理的微調因子，準確的計算元件的休眠與工作時間，降低處理器的功率消耗。Ravishankar 等學者[9] 提出了多單晶片處理器平台的電源管理技術，針對嵌入式系統的動態電壓與頻率縮放進行監控。Choi 等學者[10] 提出降低元件功率消耗的檢測和關閉處理器的方案，並結合動態電壓縮放進行計算，最後效能顯示能夠有效降低電源消耗約 37%。Ann 等學者[11] 提出了一個考慮 CPU 和多媒體加速器之間頻率的關係，同時保證高品質服務 (Quality of Service; QoS) 和減少能源消耗的電源管理機制。

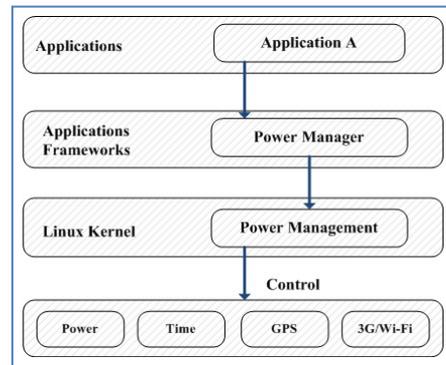
3. Android 電源管理架構

Android 是基於 Linux kernel 發展而來的行動平台版本，它是使用推疊 (Stack) 作為軟體平台的建構方式。Android 主要系統架構層分別為作業系統核心層、系統函式庫、應用架構層、應用程式層。為了延長手持裝置的使用時間，Android 的 Linux 核心特別加入了電源管理的功能。其中應用架構層主要有幾個重要文件：

- (i) frameworks/base/core/java/android/os/PowerManager.java
- (ii) frameworks/base/services/java/com/android/server/PowerManagerService.java
- (iii) frameworks/base/core/java/android/os/Power.java
- (iv) frameworks/base/core/jni/android_os_power.cpp

(v) hardware/libhardware/power/power.c

文件(i)是給應用程式層使用的，關鍵的核心文件在(ii)，它確實提供了系統電源管理的功能以及元件的狀態監控。



圖一、Android 電源管理架構

Android 電源管理架構如圖一[12]，目前 Android 作業系統大約有三種電源管理模式，分別為常開 (On)、早期暫停 (Early Suspend) 以及喚醒鎖定 (Wake Lock)。Android 的電源管理方式與一般不同的地方在於，當應用程式需要用到某樣裝置元件的資源時，需要向系統提出申請，接著再由系統向該元件執行指令。上述的這些電源管理模式雖然可以解決傳統不支援 Linux 驅動程式的情況，卻也造成無法有效管理裝置元件耗電的情形。因此，如何有效的在電源管理架構上設計一套電源管理策略達到節能目標，是本論文研究的重點。

4. 動態電源管理架構

在手持式裝置操作的過程當中，作業系統核心根據目前的使用狀況來取得周邊裝置的工作頻率以及電壓值，並且搭配電源管理機制，選擇合適的電源策略以符合當前的操作狀態。在我們所設計的電源管理機制當中，主要能夠分為以下幾個部分：

- (1) **硬體裝置 (Hardware)**：手持裝置中的感測與通訊元件，這些硬體項目接受作業系統的監控與調節，並且透過動態電源管理應用程式介面 (Dynamic Power Management Application Programming Interface; DPM API) 傳送給策略管理器 (Policy Manager; PM)。
- (2) **作業系統 (Operating System)**：使用 Android 2.0 系統以上的版本，其 Linux Kernel 為 2.6 版本，且目前已整合 Android 專案所使用的程式碼，並改進及提供多種功能，像是記憶體管理、程序管理、驅動

程式、安全性與電源管理。

- (3) **動態電源管理應用程式介面(DPM API)**：透過程式人員設計的應用程式 API 取得元件與感測器的各種參數狀態。
- (4) **策略管理器 (PM)**：這部分是本架構的核心,根據DPM API所取得的感測器參數,制定相關的策略演算法傳送給動態電源執行程序 (Dynamic Power Execution Process; DPEP) 來執行合適的策略變更。
- (5) **動態電源執行程序 (DPEP)**：最後這個部分負責接受 PM 所決定的電源管理策略,執行我們所設計的高電源/低電源策略演算法。

我們強調作業系統在加入電源管理機制後能夠大幅降低手持裝置在「閒置」或「工作」狀態下的系統功率耗損(例如:螢幕亮度調整、GPS 定位以及 Wi-Fi 與 3G 連線等所耗費的電源)。

5. 電源管理演算法設計

此章節將說明如何收集目標裝置的使用狀況資訊,並且將該資訊傳送至策略演算法中計算,產生出適合目前環境的電源管理策略,來決定控制裝置開關的時間。

5.1 電源管理介面

如前面章節所述,Android 作業系統提供了多種電源管理模式。我們根據 Android Battery Dog[13]這套軟體來設計我們的電源監控介面,它能夠測量並取得手持裝置的電源資訊。Android Battery Dog 是 Android 系統上用來監控電池電量的服務軟體,能夠產生電量記錄文件:/sdcard/BatteryDog/battery.csv,該文件包含了時間、電量訊息、溫度和電壓以及波形圖。

5.2 高/低電源策略演算法

由於手持裝置周邊元件的種類繁多,像是螢幕元件、3G 元件、Wi-Fi 元件、藍芽元件、GPS 元件...等等。本研究基於 LBS 廣泛的位置應用服務作為演算法設計考量,因此在電源管理的策略設計上針對 3G、Wi-Fi 以及 GPS 元件分為高電源管理策略與低電源管理策略。在高電源管理策略當中,行動用戶能夠根據目前的電量消耗階段計算出相對的電源查詢週期,並且偵測相關感測與通訊元件是否能夠正常運作,若無法正常運作則將感測元件關閉進入休

眠週期,降低元件閒置時所造成的功率消耗;而在低電源管理策略當中,除了高電源管理策略能夠做到的節能方案,我們另外考慮加入元件執行週期,限制感測元件能夠持續運作的時間。在這樣高/低電源策略的搭配之下,除了有效的降低感測元件閒置時的功率消耗,更重要的是能夠延長手持裝置的使用時間。

為方便描述演算法,相關參數定義說明如下所示:

- 電源監控介面:Android 監控電源應用程式介面,定義為 I_{pm} 。
- 電源資訊:目前手持裝置的電源資訊,定義為 $P_{current}$ 。
- 電源消耗資訊:假設電源資訊的期望值是 100%,期望值與目前電源資訊相減後能夠計算出電源消耗的數值,定義為 $P_{consumed}$ 。
- 電源臨界值:為了有效計算不同電源消耗階段的查詢週期,在我們的高/低電源策略當中考慮加入電源臨界值,定義為 $P_{threshold}$ 。
- 電源查詢週期:監控電源應用程式介面首次獲取手持裝置的電源資訊之後,下一次查詢電源資訊所需要經過的時間,定義為 T^Q ;而 T^Q 根據目前電源消耗階段又可分為高電源查詢週期 T^{Qh} 以及低電源查詢週期 T^{Ql} 。
- 元件執行週期:各種感測元件所能持續執行的時間,定義為 T^E 。

在完成參數定義之後,接下來提出高電源與低電源演算法設計的概念。

- (1) **高電源管理策略**:在手持裝置電源充足的情況下,分成 L_1 與 L_2 階段,為中高電量。 L_1 與 L_2 階段指的是行動用戶能夠透過 I_{pm} 向 Power Sensor 取得 $P_{current}$ (假設 Power Sensor 能夠處理電壓/頻率已轉換為百分比單位),透過所設計之公式進行計算以決定 T^Q 。
- (2) **低電源管理策略**:在手持裝置電量已經漸漸降低的情況底下又可分為 L_3 與 L_4 以及 L_5 階段,為了能夠延長手機電量的使用時間,用戶如同高電源管理策略所述能夠計算 T^Q 之外,另外還需加入 T^E 進行計算。

如此一來能夠有效的管理手持裝置的電源,在非必要的時期將元件關閉,以節省手持裝置的耗電量。接下來說明公式的計算,假設 $P_{consumed}$ 為電量消耗的百分比, $P_{current}$ 為目前的手持裝置電量,可以得到公式 (1)。

$$P_{consumed} = 100\% - P_{current} \quad (1)$$

根據消耗電量的不同，又可以將 $P_{consumed}$ 分為5個層級， $\{c_1, c_2, c_3, c_4, c_5\} \in P_{consumed}$ 如表一所示。

| | |
|-------|---------|
| c_1 | 0%~19% |
| c_2 | 20%~39% |
| c_3 | 40%~59% |
| c_4 | 60%~79% |
| c_5 | 80%~99% |

表一、電源消耗的百分比階級

這邊預設電量臨界值 $P_{threshold}$ 為20%以及當前手持裝置的耗電量 $P_{consumed}$ 去計算電源查詢週期 T^Q 與元件執行週期 T^E 如公式(2)與公式(3)。

$$T^Q = \left(\left[\frac{P_{consumed}}{P_{threshold}} \right] + 1 \right) * 6 \quad (2)$$

$$T^E = \left(\frac{1}{2} \right) * T^Q \quad (3)$$

在公式(2)當中，為了不讓電源查詢週期時間過長或過短，我們透過公式計算出行動用戶合理操作手持裝置的時間（假設行動用戶操作手持裝置的時間限制為30分鐘，可應行動用戶電量消耗狀況調整公式內的參數）。詳細的消耗電源控管階段如表二所示。

| 電源階段 | 消耗電源 $P_{consumed}$ (單位：百分比) | 查詢週期 T^{Qi} (單位：分鐘) |
|-------|---------------------------------|--------------------------|
| L_1 | $c_1 = 100\% - P_{current}$ | $T^{Q1} = 6$ |
| L_2 | $c_2 = 100\% - P_{current}$ | $T^{Q2} = 12$ |
| L_3 | $c_3 = 100\% - P_{current}$ | $T^{Q3} = 18$ |
| L_4 | $c_4 = 100\% - P_{current}$ | $T^{Q4} = 24$ |
| L_5 | $c_5 = 100\% - P_{current}$ | $T^{Q5} = 30$ |

表二、電源消耗的百分比與查詢週期關係

5.3 執行電源策略演算法

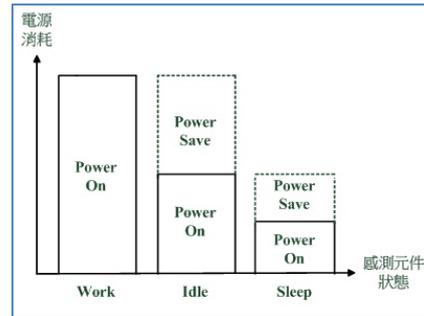
在前面所討論的電源管理方案當中，我們提到監控介面在取得當前的電源資訊後必須先將電源消耗情況分為高/低總共5個電源階段； $\{L_1, L_2\} \in L_H$ 是屬於高電源階段， $\{L_3, L_4, L_5\} \in L_D$ 則屬於低電源階段。而每個電源階段根據公式(2)會有不同的高/低電源查詢週期 $\{T^{Qh}, T^{Qi}\} \in T^{Qi}$ 。

Step1: 當 T^{Qi} 計算出來是屬於高電源查詢週期 T^{Qh} （如果 $T^{Qi} \leq 12$ ，則 $T^{Qh} = T^{Qi}$ ），即將 T^{Qh} 帶入高電源策略演算法執行電源管理。在高電源管理策略中，首先會確認目前手持裝置電源是屬於 L_1 或是 L_2 階段，接著偵測Wi-Fi元件是否連接的上固定AP（Access Point），若連接不上則關閉Wi-Fi元件，則確認是否偵測的到3G網路流量，若無網路流量即可關閉3G元件（同時關閉GPS元件），使各元件進入休眠狀態以節省電源。待查詢週期到達以後，則繼續進行元件功能的偵測。在高電源管理策略中不考慮元件執行週期。

Step2: 當 T^{Qi} 計算出來是屬於低電源查詢週期 T^{Qi} （如果 $T^{Qi} \geq 13$ ，則 $T^{Qi} = T^{Qi}$ ），在手持裝置電源漸漸降低的情況底下，我們另外考慮計算元件執行週期 T^E （ $T^E = (1/2) * T^{Qi}$ ），並將 T^{Qi} 與 T^E 帶入低電源策略演算法執行電源管理。在低電源管理策略中，一樣會確認目前手持裝置電源是屬於 L_3 、 L_4 或是 L_5 哪一種電源階段，接著不論是否連接的到Wi-Fi或3G元件，當元件執行時間到達以後自動將Wi-Fi與3G以及GPS元件關閉，節省手持裝置電源的消耗。

6. 效能評估與分析

實驗結果表明，手持裝置的周邊元件在加入高/低電源管理策略以後，皆可降低系統功率消耗。



圖二、電源管理策略效能評估

總結來說，當Wi-Fi、3G以及GPS元件處於工作狀態時，電源是持續不斷在消耗的情形；而元件處於閒置狀態時，電源策略演算法能夠關閉元件的開關，使系統進入休眠狀態來節省部分的電源消耗。

7. 結論與未來研究方向

隨著智慧型手持裝置與無線通訊技術的發展，用戶透過Wi-Fi或3G連上網際網路進行LBS服務已然成為未來的趨勢。因此，如何

在不增加裝置設計花費成本的前提之下進一步提升電池的續航力，是可以研究的方向。本篇論文針對 Android 智慧型手持裝置設計了高電源與低電源的動態管理策略，經實驗結果效能分析顯示，有效的電源管理策略能夠讓感測元件獲得更多的休眠時間，進而達到手持裝置系統節能的效果。未來將持續設計更具有節能效率的策略原則，以及不同的嵌入式系統上的電源管理功能，來應對未來多樣化的網路行動服務。

致謝

本研究感謝行政院國家科學委員會 (NSC 101-2221-E-324 -043 與 NSC101-2221-E-241-018) 的經費支持。

參考文獻

- [1] Brock, B. and Rajamani, K., "Dynamic Power Management for Embedded Systems," **IEEE International SOC Conference**, Sep 2003.
- [2] Shih, H. C., Wang, K., "An adaptive hybrid dynamic power management algorithm for mobile devices," **IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing**, 2006.
- [3] Bellavista, P., Cardone, G., Corradi, A. and Foschini, L., "The Future Internet convergence of IMS and ubiquitous smart environments," **Journal of Network and Computer Applications**, Volume 35, Issue 4, Pages 1203–1209, July 2012.
- [4] 李宗庭，手持式裝置電源管理-以 Android 為例，2012 年大同大學資訊工程研究所。
- [5] 方志鵬，使用適應性學習樹預測之動態功率管理系統研製，國立台北科技大學電機工程系。
- [6] Chiasserini, C. F., "Improving Energy Saving in Wireless Systems by Using Dynamic Power Management," **IEEE Transactions on Wireless Communications**, vol.2, Issue:5, pp. 1090-1100, Sep 2003.
- [7] Holland, O. and Aghvami, A. H., "Intra-operator dynamic spectrum management for energy efficiency," **IEEE Communications Magazine**, vol.50, Issue:9, pp. 178-184, Sep 2012.
- [8] Chen, J., "A research on an optimized adaptive dynamic power management," **IEEE International Conference on Computer Science and Information Technology**, pp. 52-55, Aug 2009.
- [9] Ravishankar, C., Ananthanarayanan, S., Garg, S. and Kennings, A., "Analysis and evaluation of greedy thread swapping based dynamic power management for MPSoC platforms," **2012 13th International Symposium on Quality Electronic Design**, pp. 617-624, March 2012.
- [10] Choi, J. and Cha, H., "A Processor Power Management Scheme for Handheld Systems Considering Off-Chip Contributions," **IEEE Transactions on Industrial Informatics**, vol.6, Issue:3, pp. 255-264, Aug 2010.
- [11] Ann, J., Min, J. H., Cha, H. and Ha, R., "A Power Management mechanism for Handheld Systems having a Multimedia Accelerator," **IEEE International Conference on Sixth Annual Pervasive Computing and Communications**, pp. 663-668, March 2008.
- [12] **Android Open Source Project**, <http://developer.android.com/reference/android/os/PowerManager.html>.
- [13] **Android Battery Dog**, <http://andbatdog.sourceforge.net/>.