

以 Pebble 為基礎的跨平台點對點部落格系統

許文峰, 高永威, 洪偉翔, 袁賢銘,

台灣省新竹市大學路 1001 號交通大學資訊工程學系,

fxsh70133@gmail.com, ywkao@cs.nctu.edu.tw, weihsiang.hung@gmail.com, smyuan@cis.nctu.edu.tw

摘要

部落格服務在最近幾年越來越受歡迎。網路使用者將他們的文章放到他們的部落格網站。另一方面, 如果服務是建立在主從架構模型上, 使用者無法在他們將文章放到網站後完全控制他們的文章。所以有些人建議使用點對點技術來建立部落格服務。然而, 和傳統的部落格系統比較, 點對點系統的使用者經驗無法像傳統部落格的一樣。點對點系統無法分類使用者的文章, 而且無法提供或只能部分提供多媒體內容。這篇論文提出一個新的平台(Pebble2P), 試著去結合傳統的部落格系統和點對點部落格系統。因此, 使用者能夠藉由同樣的方式在傳統的部落格中取得使用者經驗, 且保留點對點部落格系統的優點。

關鍵詞

點對點; 部落格; Pebble;

1. 引言

現今, 網路對於人類活動來說變得不可或缺。全球網路基礎建設集中建立且在我們每天的生活中越來越受歡迎。越來越多人、公司或其他組織維護他們自己的網站。他們也利用這些網站提供資訊和服務給使用者。在早期, 建立網站對沒有相關背景的人來說是非常困難的。隨著網路科技的進步, 一些網路服務提供者提供使用者一個簡單的方式去建立屬於他們自己的網站。部落格就是這些服務當中的一個, 它是網路日誌的縮寫。

部落格(也就是網路日誌)是一些包含定期更新文章的網站, 最近更新的文章會放在頁面

的最上方, 之前的文章會照時間順序更新到網頁上[10]。部落格很容易去建立網頁。部落格條目, 也就是部落格文章的小單元, 藉由利用簡單的格式來產生。部落格條目包括文章的內容(正文、圖片、超連結或影片)、永久鏈結、提交文章時間...等等。其他使用者可以使用評論或是引言和部落格作家互動。

人們可以用兩種方式來建立他們的網站。第一種方式是自己建立部落格系統。這包括撰寫部落格程式或部落格套裝軟體像是 WordPress 或是 Pebble, 及在網路上租虛擬空間或用個人機器來儲存部落格。第二種方式是仰賴部落格服務提供者(BSPs)。

第一種方式的優點在使用者可以控制所有的資料。但使用者必須維護他們自己的軟體、硬體和資料, 所以他們必須要對電腦技術有足夠的知識。因此, 只有擁有足夠電腦技術的人才可以採取這種方式。相反地, 第二種方式讓人們更容易去建立自己的部落格。任何人都可以是部落格作家, 而不用去處理一些關於系統安全性、資料備份、機器更新的問題。所以第二種方式是主要建立部落格的方式。舉例來說, 有很多部落格提供者, 像是 Blogger.com 和 WordPress.com。使用這些服務更容易建立這些部落格, 但還是有些潛在的問題。每個部落客在部落格上撰寫文章和資料, 但控制者是部落格提供者(BSP)而不是作者本身。所以越來越多的部落客不想要他們的努力掌控在 BSP 上。一些部落客開始自己建立部落格。雖然現今建立部落格服務很容易, 但是維護部落格服務對大部的人來說很困難。如果部落格服務端

的資料損壞或是沒有備份，部落客會失去他們所有的文章。

最近幾年，個人電腦的計算能力和網路頻寬快速成長。人們願意去提供計算能力及和其他人分享資訊，而點對點的技術提供了這個可能性。點對點系統的主要特色是能提供匯集和利用大量資源、自我組織、負載平衡、適應性、容錯的能力。所以有些人試著用點對點的技術去解決上述的問題。

我們發現兩個點對點部落格系統問題的解決方法，分別是 P2PBlog 和 P2PmBlog，這些在下一節會介紹。我們發現以下問題。第一個是他們都有自己的部落格內容瀏覽器。瀏覽器無法在部落格文章的內文之間顯示多媒體的內容。第二個是他們沒有任何方法分類部落格文章。舉例來說，傳統部落格可以根據標籤或類別來分類

為了解決以上問題，我們決定修改現存以服務為基礎的點對點系統部落格軟體。用這種方法，使用者可以使用一般的網路瀏覽器來瀏覽部落格文章。

在論文接下來的章節中，我們首先介紹兩個上面提到的兩個相關工程。在第三節，我們描述系統架構。在第四節，我們用上面提出的兩個相關工程來比較我們提出的系統 (Pebble2P)。在第五節，作此篇論文的結論。在第六節，列出未來展望。

2. 相關工程

在此節，我們介紹兩種和 Pebble2P 類似的應用。第一個是 P2PBlog，用之前的函式庫成員實作。另一個是 P2PmBlog。它們都是點對點部落格系統的解決方式。

2.1 P2PBlog

P2PBlog[1] 是點對點部落格的應用。圖 1 顯示 P2PBlog 的圖形用戶介面。P2PBlog 的主要特色是使用 OpenID[2]來解決點對點網路的匿名功能。它的圖型用戶介面分成五個區域。使用者可以在區域 1 中找到作者或部落格條

目，在區域 2 可以執行部落格功能，像是登入、編輯用戶資料、發布部落格條目或回到首頁。區域 3 顯示使用者的書籤和訂閱資訊。區域 4 顯示部落格條目的內容區域。最後，區域 5 紀錄應用訊息。

P2Pblog 以下的元件:GUI(圖形用戶介面)。使用者透過圖型用戶介面使用此系統。除了點對點的發布/訂閱模組(FreePastry)，GUI 可以和其他的元件溝通。OpenID 負責 OpenID 的整合，每一點都需要執行網路服務來處理 OpenID 身分驗證。靜態資料控制重要的靜態資源。(Action Function)動作函數在此應用中擁有特別的能力。點對點發布/訂閱模組能管理點對點和發布/訂閱操作。

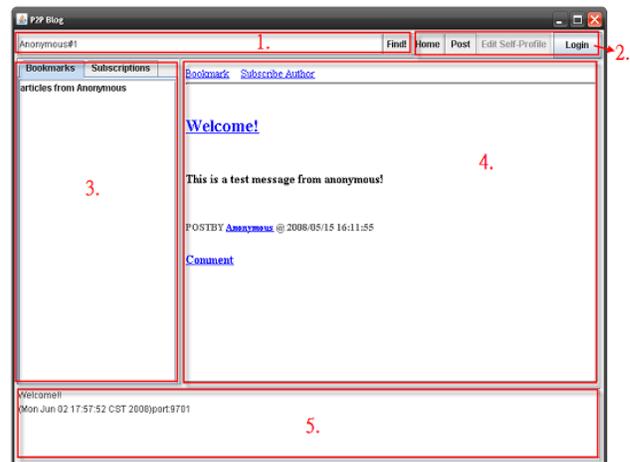


圖 1. 點對點部落格截圖

2.2 P2PmBlog

P2PmBlog 是 Pepito 專案中其中一個成就。該系統由“部落格節點”組成。這些節點形成一個點對點網路，並維護部落格資料和很多的“部落格客戶端”。部落格客戶端可以透過 xmlrpc 連接 到這些節點獲取部落格資料。該專案提供三種部落格客戶端：

- 部落格提取：一個程序，可以將外部部落格資料推到 P2P 網路。
- 可移動的部落格閱讀器：讓用戶藉由使用手機在 P2P 網路中瀏覽部落格資料。

•桌面部落格閱讀器：一個 java 應用程序，讓用戶可以使用桌面計算機連接到一個部落格節點，並進入 P2PmBlog 系統。

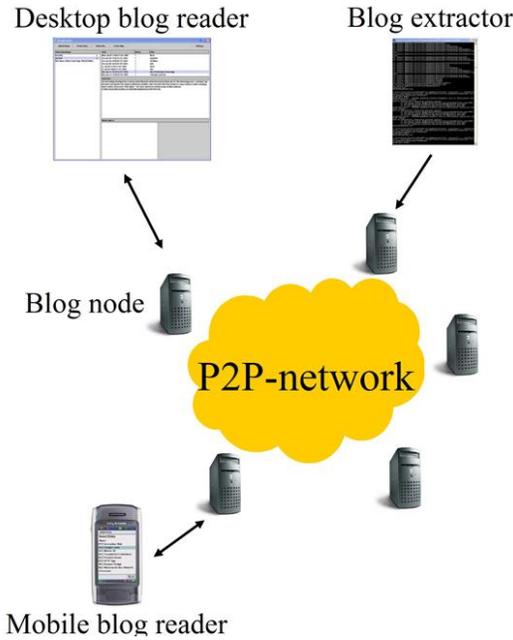


圖 2. P2PmBlog 概觀

3. 系統架構

在本節中，我們目前概述 Pebble2P 部落格系統。它包括 Pebble2P 網路應用程序、引導服務器以及在前面兩者裡的組件。

3.1 概述

Pebble2P 包含兩個應用程序，Pebble2P 引導服務器和 Pebble2P 網路應用程序。Pebble2P 引導服務器是一個 Java 應用程序，而 Pebble2P 網路應用程序是一個以 Pebble 為基礎的網路應用程序。此 Pebble 是以部落格軟體為基礎，在 2.3.2 版的 JAVA 上實作的服務器。圖 3 模擬 Pebble2P 系統的實際狀況，並註明這兩個組件和用戶之間的關係。

在該方面的用戶，用戶（部落格擁有者/觀看者）可以使用一個共同的網路瀏覽器如 Internet Explorer（IE）和火狐，連接到 Pebble2P 網路應用程序，且像一般的部落格執行作業。例如，用戶可以查看部落格，註冊使用者帳戶創建一個新的部落格，發布部落格

條目或發表部落格評論等。詳細功能演示將在第 4 章顯示。

我們想強調，無論 Pebble2P 引導服務器和 Web 應用程序在 P2P 網路上都是同等實體，他們都可以持久存留部落格領域資料。這些資料像是部落格的數據對象，如部落格、使用者詳細資料和部落格日誌。在下面的小節中，我們將解釋這兩個應用程序。

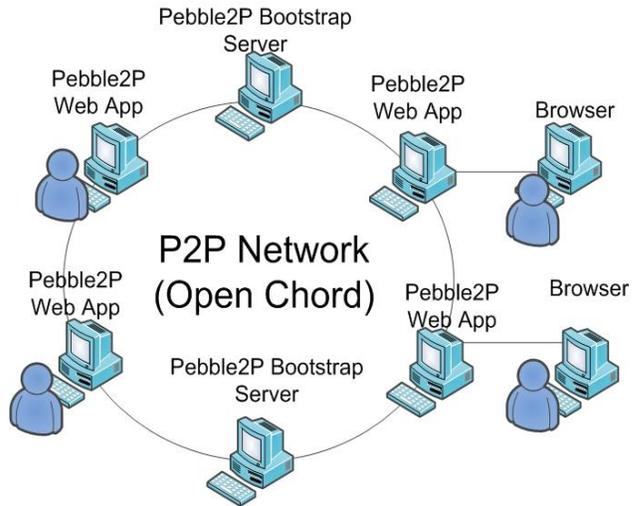


圖 3. Pebble2P 系統概觀

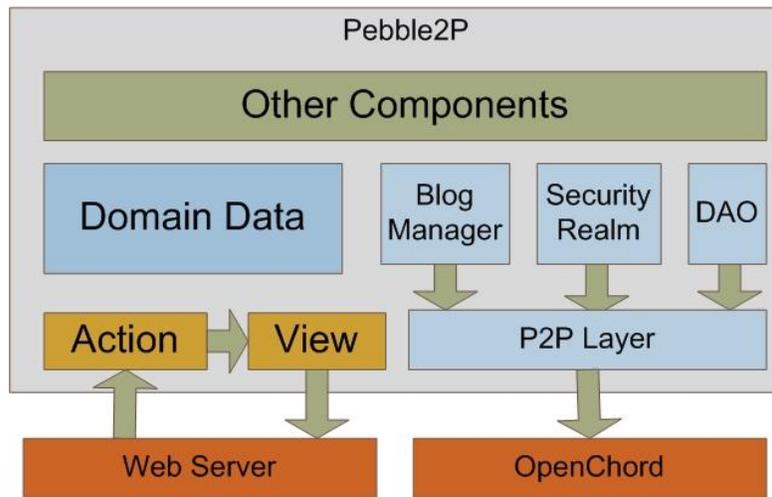


Figure 4. Pebble2P 網路應用程序概觀

3.2 Pebble2P 引導服務器

Pebble2P 引導服務器是一個 Java 圖形用戶界面 (GUI) 應用程序，而且它是 P2P 網路中的對等實體。大多數用戶不需要運行此應用程序，除非他們想建立自己的 Pebble2P 網路。換句話說，如果他們想建另一組部落格，他們可以運行 Pebble2P 引導服務器實例，而且讓其他 Pebble2P 網路應用程序加入到這個網路。

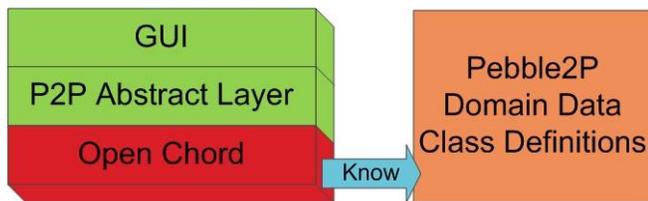


圖 5. Pebble2P 引導服務器三層設計

在設計 Pebble2P 引導服務器可以分為三個層次，圖形用戶界面、點對點抽象層和 Open Chord，如圖 5 所顯示的。Open Chord [3] 層是 Chord[4] 網路的實作。點對點抽象層提供 API，讓 GUI 可以控制網路行為，如創建一個網路，連接到一個已存在的網路或離開網路。圖形用戶界面層讓使用者可以控制應用程序，而且使用者也可以從服務端(同等實體)得到更多資訊以及查看訊息視圖的 P2P 抽象層。點對點抽象層提供介面讓 GUI 能控制服務器。

這種設計的好處是如果我們想改變網路層 (Open Chord)，我們並不需要重寫整個應用程序。換言之，我們僅僅只要修改 P2P 抽象層的實作。

此外，由於 Open Chord 的限制，Pebble2P 引導服務器必須知道 Pebble2P 網域類別的實作，如部落格、BlogEntry、UserDetail 和所有相關的類別，這些我們將在下一節中展示。

3.3 Pebble2P 網路應用程序

Pebble2P 網路應用程序是一個以 Pebble 為基礎的部落格網路應用程序，而且這是我們系統的主要應用程序。Pebble 是輕量級且開放原始碼的客戶端/服務端部落格軟體。它由 Java

EE 實作，而所有的維護和管理透過網路瀏覽器執行。這就是為什麼我們決定將 Pebble 的架構，從傳統的客戶端/服務端架構修改成 P2P 架構。

普通使用者可以在 JSP 2.0/Servlet 2.4 兼容的網路服務器 (Apache Tomcat 或 GlassFish) 上運行此網路應用程序，然後使用者可以使用任何網路瀏覽器如 Internet Explorer 或 Firefox 連接到該應用程序，就像他們連接到一個網站一樣。

在此應用程序中大多數的類別是用 Pebble 原始的版本來實作。我們替代和補充一些類別以滿足我們的需求。如同圖 4 所顯示的，我們將這些類別分成幾個不同的組件。

此外，在原本 Pebble 的實作中，只有具有管理員權限的使用者可以添加一個用戶或創建一個新的部落格。在 Pebble2P，我們沒有管理員的角色。因此，我們必須在 Pebble2P 增加註冊功能。

以下各段是 Pebble2P 網路應用程序的實作細節：

3.3.1 領域數據 (Domain Data)

這些我們需要放到 P2P 網路上的類別稱之為領域數據。領域數據類別可以是 Blog，代表部落格的類別；BlogEntry，代表部落格條目的類別；PebbleUserDetails，代表 Pebble2P 用戶資料的類別。為了將這些物件放到網路上，我們需要讓這些物件可以被序列化。在以下的表格中，我們會列出一些需要被放到 P2P 網路上的物件。

TABLE 1. BLOG 物件組織

Name	Description
id	The blog owner's user name as blog id.
years, months, days	Record ids of all blog entries in specific date.
properties	Settings of this blog.
root category	The root of a category tree.

TABLE 2. BLOGENTRY 物件組織

Name	Description
permalink	The permalink of this blog entry.
categories	The category that the blog entry falls into.

Tags	The list of tags for this blog entry.
title, body, excerpt	Blog entry contents.
comments	the collection of comments for the blog entry

TABLE 3. PEEBLEUSERDETAILS 物件組織

Name	Description
username	The user name of this user
password	Encrypted password of this user.
name, email address, profile, etc.	User data.

3.3.2 點對點層(P2P Layer)

點對點層提供一些方法，像是在 Chord 網路上管理部落格、部落格條目和用戶詳細資料。如果想要用其他類別管理網路上的物件，他們只能調用這些方法。

如果其他開發人員要修改網路層的實作。例如，將 Open Chord 改寫成另一種 P2P 協議實作，他們只需要改寫這個類別而不用做其他改變。一般來說，如果我們想要實作新的 P2P 類別，我們可以將此類別加入到 P2P 網路的構造函數中，然後再實做其它介面。被叫用的該方法會在 P2P 網路應用程序停止前離開。

在我們的實作中，每個領域資料物件都有一個字串鍵，讓我們可以在 Chord 網路使用鍵插入或是取回領域資料物件。該部落格的字串鍵格式是“部落格/[用戶名]”，部落格條目的格式為“BlogEntry/[部落格條目號]”，用戶的格式是“用戶/[用戶名]”

3.3.3 數據存取物件(Data Access Object 簡稱 DAO)

Pebble 實做一個抽象類別 DAOFactory，用來存取部落格條目。Pebble 默認的 DAO 策略是 FileDAOFactory，主要是在檔案系統以 XML 的格式存儲每個部落格條目。在 Pebble2P，我們寫一個新的 DAOFactory，並把它稱為 P2PDAOFactory。我們用它把部落格條目放置到 P2P 網路上或是從 P2P 網路取回部落格條目。

3.3.4 點對點層(P2P Layer)

安全領域的是由原始的 Pebble 所定義的。它是用來提供詳細的用戶資訊和管理用戶。換

言之，其他組件可以透過這些類別管理用戶資訊細節。為了將使用者設定檔放到 P2P 網路上，我們實作了 Pebble2PSecurityRealm。

3.3.5 點對點層(P2P Layer)

在原始的 Pebble 實作中，Pebble 有管理權限來管理用戶。但 Pebble2P 沒有這樣的用戶。因此，我們需要添加的功能，讓用戶可以註冊一個用戶名在 Pebble2P。

當用戶連接到 Pebble2P 網路應用程序，用戶將看到 Pebble2P 中默認部落格的首頁。然後，用戶可以藉由註冊按鈕註冊一個帳戶。

用戶按“註冊”按鈕之後，Pebble2P 將註冊表格送到使用者瀏覽器。然後用戶填寫表格，並提交給 Pebble2P。然後，Pebble2P 檢查用戶是否存在。如果沒有，Pebble2P 將創建新的用戶細節和部落格及上傳它們到 P2P 網路。

有一個事實是當用戶完成註冊，Pebble2P 為了用戶創建一個部落格及使用用戶名當作部落格的 ID。換句話說，Pebble2P 為了每個使用者只能創建一個部落格。

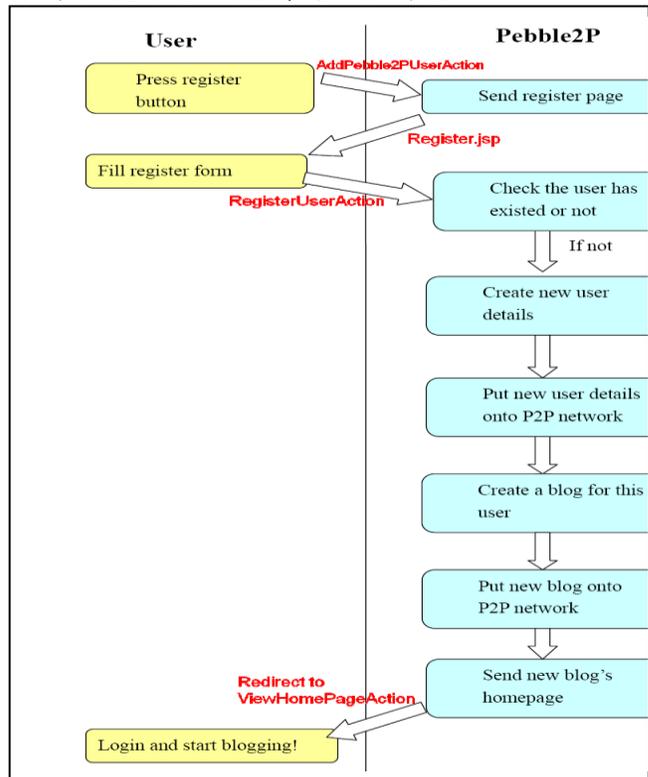


Figure 6. Flow chart of flow of user registration

TABLE 2. COMPARE WITH P2P BLOG AND P2PMBLOG

	Pebble2P	P2P Blog	P2PmBlog
P2P Library	Open Chord	FreePastry	P2PS [8]
P2P protocol	Chord	Pastry	Tango [9]
Registration	To register on system	OpenID	To register on system
Browser	Web browser	Custom windows application	PC or Mobile client
Multimedia content	Yes	No	independent column
Keyword search	Yes	No	No
Article classify	archives, tags, categories	No	No
Subscription	RSS	Scribe	RSS, Atom
Platform	Cross platform	Windows only	Cross platform

4. 比較

在這一章中，我們用兩個相關的工程來比較 Pebble2P，P2PBlog 和 P2PmBlog，以表格 2 陳述。

雖然點對點部落格使用 OpenID 註冊，OpenID 帳戶通常太長而難以記住。與其他兩個系統不同的是，必須使用他們自己的瀏覽器；Pebble2P 可以使用一個共同的網路瀏覽器查看部落格。因此，用戶可以張貼與多媒體內容（圖片或視頻）相關的部落格條目。Pebble2P 可以藉由自動更新封存部落格條目，用戶可以使用標籤和類別進行分類文章這種在 Pebble2P 裡的傳統客戶端/服務端部落格系統。這三種實作方式都有訂閱機制。雖然這三個系統都是用 Java 實作，因為 P2Blog 使用以 IE 瀏覽器為基礎的嵌入式瀏覽器，所以 P2PBlog 只能在 Windows 上執行。另一方面，Pebble2P 和 P2PmBlog 都是跨平台的。

當用戶連接到 Pebble2P 網路應用程序，用戶將看到 Pebble2P 的部落格默認首頁。然後，用戶可以按註冊鈕來註冊一個帳戶。

用戶按註冊鈕後，Pebble2P 傳送註冊格式到用戶的瀏覽器。然後用戶填寫表格並提交給 Pebble2P。之後，Pebble2P 會檢查用戶是否存在。如果沒有，Pebble2P 將創建新的用戶資料

細節和部落格給此用戶，並將他們上傳到 P2P 網路上。

有一個事實是當用戶完成註冊後，Pebble2P 將創建一個部落格給用戶並將用戶名作為部落格的 ID。換句話說，Pebble2P 只能為一個使用者創建一個部落格。

5. 結論

本文提出了另一種名為 Pebble2P 的點對點部落格系統，以改善點對點部落格系統的使用者經驗。Pebble2P 的用戶很容易能夠使用 Web 瀏覽器，連接到 Pebble2P Web 應用程序來啟動部落格而且用戶可以呈現文章的內容不僅在文字的部分，而且也有多媒體內容，像是圖片、視頻。在 Pebble2P 上的使用者經驗也保留了其他點對點部落格系統的優勢，如分散式系統架構、自動複製、保護人權來對抗部落格服務提供者。此外，Pebble2P 是一個獨立於 P2P 部落格系統的平台。

然而，Pebble2P 仍然有缺點。雖然 Pebble2P 以 Pebble 為基礎建立，但有一些 Pebble 的功能無法整合到 Pebble2P 上，像是用戶自定義插件、文件上傳。P2P 網路需要足夠的使用者才能保持網路運行穩定。此外，該系統的安全性在每台 Pebble2P 上的主機也是一個問題。

6. 未來展望

在不久的將來我們應該解決以下問題：

- Peeble 的使用者可以編寫自己的插件來添加其他功能。但 Open Chord 至今不支援遠程加載類別。所以，使用者只能在內建的 Pebble 裡使用 Pebble 官方插件。我們將等待 Open Chord 開放新版本或自己實作遠程類別加載器。
- Spring 框架 2.0 或更新的版本提供一些 API 來支持 OpenID 的身份認證。我們可以升級 Pebble2P 的 Spring 框架來支援 OpenID 的身份認證。
- Pebble2P 不支援檔案上傳，所以多媒體內容視其他服務提供商而定，如 YouTube 或 Flickr。如果我們找到一個好的機制能將大檔案分成數個片段，我們就能實作這個部分。
- Pebble2P 關鍵字搜索功能繼承了 Pebble。它需要一個索引器將整個部落格編入索引。當部落格條目的數目增加時，效率會降低。我們將努力改善與點對點機制的關鍵字系統

7. 參考

- [1] Chen-Pu Lin, Yung-Wei Kao, Shyan-Ming Yuan. "A P2P Blog System with OpenID Integration", Third 2008 International Conference on Convergence and Hybrid Information Technology, 2008.
- [2] David Recordon, Drummond Reed. "OpenID 2.0: a platform for user-centric identity management", Proceedings of the second ACM workshop on Digital identity management, pp. 11-16, Alexandria, Virginia, USA, 2006.
- [3] Sven Kalle and Karsten Loesing. "Open Chord version 1.0.4 User's Manual", 2007
- [4] Ion Stoica , Robert Morris , David Karger , M. Frans Kaashoek , Hari Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, p.149-160, San Diego, California, United States, August 2001.
- [5] Pebble's official web site: <http://pebble.sourceforge.net/>
- [6] P2PmBlog's official web site: <http://www.sics.se/pepito/D5.2/www/p2pmblog.html>
- [7] Datta, A.K., Gradinariu, M., Raynal, M., Simon, G. "Anonymous publish/subscribe in p2p networks", Proceedings of the 17th International Symposium on Parallel and Distributed Processing, pp.74.1, Washington, DC, USA: IEEE Computer Society, 2003.
- [8] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, François Yergeau, John Cowan, ed. XML 1.1 (Second Edition), W3C Recommendation, 16 August 2006.
- [9] Carton, B., Mesaros, V. "Improving the scalability of logarithmic-degree DHT-based peer-to-peer networks", Euro-Par-International Conference on Parallel Processing 2004.