在 Windows 系統上
實作 Oracle Forms 程式自動化部署

陳宏益
朝陽科技大學
副教授
hychen39@gm.cyut.edu.tw

李慧玲
朝陽科技大學
研究生
wrnd1246@gmail.com

摘要

一般 Oracle Forms 程式的部署流程，需先使用 FTP 或其它方式上傳 Forms 程式至遠端 Forms Server，再於遠端 Server 進行手動編譯。此流程不但操作耗時且未整合版本控管機制。本篇論文提出一個在 Windows 作業系統下的 Oracle Forms 程式自動部署系統以解決上述問題，利用批次檔整合編譯指令及 Git 版本控管指令，可在特定的時間週期自動針對版本異動的專案進行編譯。如此一來，不但解決資訊人員在部署 Forms 程式時所面臨到繁雜且瑣碎的部署時間達到快速部署，並且，當發生錯誤或異常時可以快速的回復至前一個穩定版本，降低版本管理上的失誤。

關鍵詞：Oracle Forms、自動化部署、Git 版本控管

Abstract

The process to deploy Oracle Forms programs is time consuming and does not integrated with version control systems. Users have to upload forms programs to remote Forms Server by FTP or other ways and compile the programs at the remote. The paper has proposed an auto-deploy system in Windows OS to improve the process. The system integrates the compile command of Oracle Form Builder and the Git version control system by a Windows batch scripting file. The batch scripting file running at a regular time on the Forms Server can pull the new version program files from a Git server and compile them. The proposed auto-deploy system can save the time for deploying Oracle Forms programs. Moreover, the updated Forms programs can be reverted to the previous stable version easily with the help of Git version control system when exceptions occur to these updated programs, which mitigates the impacts of the exceptions to the application.

Keywords: Oracle Forms、Auto-Deploy、Version Control

1. 緒論

Oracle Forms Builder 是 Oracle 公司的一套快速應用程式開發工具，能夠快速開發網際應用程式[7]，該系統之架構為三層式架構: 客戶端(Client Tier)使用 Java Applet 呈現使用者界面，商業層(Business Tier)為 Forms Server，資料，資源層(Resource Tier)則為資料庫。

Oracle Forms 程式開發至使用的流程，由開發者於本地端電腦開發並測試Forms程式，接著使用 FTP 或其它方式上傳 Forms 的原始檔(.fmb)至遠端 Forms Server 的指定目錄，再

於遠端 Server 開啟 Forms Builder 編譯上傳的.fm 成為.fmx 執行檔，最後一般使用者將可使用瀏覽器連線至遠端 Forms Server，要求 Server 執行特定的.fmx 檔案。

前述使用流程部署相當耗時，需先上傳 Forms 程式後再至 Server 端進行手動編譯。此外，此流程未整合本版控管機制，無法讓專案回復到先前的狀態、比對修改不同版本差異、或者對版本異動進行稽核。

將前述手動式部署改成自動化並整合版本管控系統，開發者將無需至 Server 端進行手動編譯程式，解決資訊人員在部署 Forms 程式時所面臨到繁雜且瑣碎的部署問題，減少作業所需耗費的時間成本，並大幅降低部署過程中人為介入操作的機會。並且，當發生錯誤或異常時，可以快速的回復至前一個穩定版本。本論文提出一個在 Windows 作業系統下的 Oracle Forms 程式自動部署系統，利用批次檔指令整合 Git 版本控管系統及 Oracle Forms Builder，可於特定時間週期下自遠端版本控管伺服器下載最新版本，並針對有異動的版本自動進行編譯，並將完成編譯之 Forms 執行檔移至指定之目錄，以達成前述自動部署及整合版本控管系統之目標。

本文後續章節安排如下，第二節回顧 Oracle Forms Builder 開發工具及 Git 版本控管系統。第三節說明本論文之 Forms 程式自動部署系統，並展示系統實作成果，最後則為結論。

2. 文獻回顧

2.1 Oracle Forms Builder 開發工具介紹

Oracle Forms Builder 是一個開發網際應用程式的快速應用程式開發(Rapid Application Development, RAD)工具，包含許多 Wizard 工具，如：Data Block Wizard、Layout Wizard、Property Palette、Integrated PL/SQL Editor 等，能夠幫助開發者迅速的開發應用程式，開發語言主要使用 PL/SQL。Oracle Forms Builder 目前也是 Oracle E-Business Suite (EBS) ERP 的開發工具之一，用來客製 EBS 功能[11]。

Oracle Forms Builder 雖然能夠快速開發網際應用程式，但因其前端主要使用 Java Applet 技術呈現操作介面，需整合其它架構才能在手持式裝式置上使用。例如，將 Oracle Forms 程式封裝後以 Web Service 的方式提供服務，在前端使用 Oracle Mobile Application Framework 應用此 Web Service 將資料傳到 Oracle Forms 程式[12]。

Oracle Forms 程式在 Oracle Forms Server (以下簡稱 Forms Server)中執行。Forms Server 為典型的三層式架構：Client、Application 與 Database 等層級。Client 層級的主要元件為 Forms Client，當使用者開始運行一個 session 於網頁上，Forms Client 會自動地下載 Java Applet 到用戶端儲存，並且瀏覽器中執行此 Applet。

Oracle Forms 伺服器位於 Application 層級，主要由二個元件組成，分別為：Forms Runtime Engine、Forms Listener Servlet[7]。以下分別說明：

Forms Listener Servlet 同樣也在伺服器端執行，當在客戶端下載完成 Applet 並執行時，該 Applet 會與 Forms Listener 交談，Form Listener 會配置一個 Form Runtime Engine 給該 Applet，做為與資料庫交談的程序。之後，客戶端的 Applet 便開始與其配置

到的 Forms Runtime Engine 連線，由 Forms Runtime Engine 負責客戶端與資料庫間的互動。

Forms Runtime Engine 是在伺服器端代表 Forms Client 與資料庫的連接的一個程序。所有在客戶端使用者界面產生的事件皆會被送到其對應的 Form Runtime Engine 進行處理，處理完後的結果會再送回客戶端的 Applet 呈現結果。當使用者關閉 Forms 應用程式，Form Runtime Engine 會被收回到執行引擎池(Running Engine Pool)中或者結束程序。

2.2 Git 版本控管工具介紹

Git 是一款開源分散式版本控管系統，每位開發者在自己的本地端電腦工作目錄都有完整的儲存庫，所有版本與分支管理工作都能在本地端的儲存庫上達成，不需要網路連線也能在本機單獨工作。還有一個與其它版本控管系統的重要差別，每次讀者提交更新時，Git 記錄的不是差異的部份，而是快照當時的整個儲存庫，並且將 HEAD 參考指標移至此次快照，HEAD 指標指向目前所在分支的某次快照。

Git 系統中的分支是一個提交物件(Commit Object)的鏈串結構。當使用者提交更新時，Git 會產生數個物件並放到儲存庫中。目錄中的每一個檔案會進行快照，之後產生相對應的 BLOB Objects。另外，產生 Tree Object 內含指向 BOLB Objects 的指標。最後，產生 Commit Object，內含指向 Tree Object 的指標以及最近一次提交的 Commit Object 的指標。經過多次的提交後，這些提交物件便形成一個鏈串結構。每個分支會有各自的分支指標指向某個提交物件，藉由向後回溯便可以得到該分支上的歷史快照或者不同時間下的版本。在 Git 下只要一次快照就可以瞬間完成建立分支，不需要花費太多時間，這樣的結構提高了 Git 分支的使用效率[8]。

分支主要目的是用來解決開發時不同版本的需求。假設此時你與團隊正在開發某個應用程式，臨時接到要添加某個新需求或是修復某個問題，那麼你可以使用分支進行處理。假設目前在主線上。首先使用 `git checkout -b newBranchName` 產生一個新的分支並切換至分支。接著處理要求並提交更新。測試成功後要將分支合併到主線，使用 `git merge nerBranchName` 指令進行合併。

3. Oracle Forms 程式自動化部署實作

以下內容將會介紹 Oracle Forms(以下簡稱為 Forms)程式自動化部署的使用情境與系統架構，接著說明實作步驟及成果。

3.1 自動化部署情境

開發者會使用 Git 將 Forms 程式的原始碼 push 至 Git Server。Server 端系統自動於隔日六點進行差異比較，若有差異，將差異的.fmb 原始檔進行編譯，並且將編譯後的.fmx 檔案放至部署目錄下。

3.2 系統架構

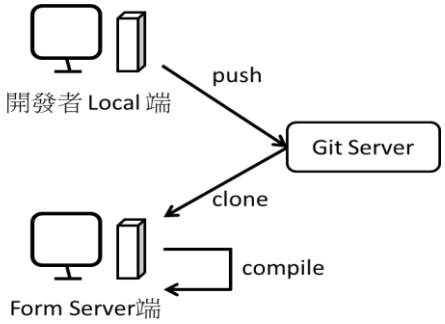

圖 1 自動化部署系統架構

使用 Git 進行自動化部署程序時，如圖 1 所示，開發者會將 Forms 程式的原始碼 push 至 Git Server。當我們在 Server 端部署 Forms 時，需在 Server 端使用 Git 版本控管系統 clone 在 Git Server 上最新檔案版本。完成更新後，便可以在 Server 端針對差異檔案進行編譯 Forms 原始檔(.fmb)產生執行檔(.fmx)。

3.3 系統實作

使用 Git 進行自動化部署的設定可分為三大步驟:

1. Git Server 的專案複製與身份鑑別(Authentication)設定: 在 Server 端初始化時需進行專案複製，並且向 Git Server 身份鑑別設定。為了免去手動輸入密碼，可以設定 Sever 端電腦使用金鑰進行自動鑑別。
2. 批次編譯更新後的 .fmb 檔: 把找出異動檔與 Compile 的指令寫在批次檔，自動針對更新後的.fmb 檔執行編譯。
3. 設定 Server 端排程進行自動更新及編譯: 設定自動排程，在特定的時間自動完成部署。

以下分別說明。

(a) Git Server 的專案複製與身份鑑別設定

1. 新增一個檔案 git_command(名稱沒有一定)，不需要副檔名。為了方便，將檔案建在打開 Git Console 之後最初的目錄底下 C:\Users\Administrator\。
2. 在 git_command 檔案中，輸入以下指令並儲存。

```
cd c:/Forms_Proj

git clone
https://user@bitbucket.org/user/
project.git
```

上述指令碼將會至 Bitbucket 下載最新的專案，儲存至 C:/Forms_Proj/key_case 目錄。

3. 設定 Git Server 進行通訊時所使用的 SSH key，以達到去除輸入密碼的步驟。

Git Server 可使用金鑰方式進行身份驗證，是一種非對稱式加密[1]。甲方在將資訊送出去之前，先以乙方的公鑰加密，而乙方在收到經過加密的資訊之後，就以乙方的私鑰解密，由於只有乙方才知道乙方的私鑰，所以也只有乙方能夠解密。公鑰將放在 Git Server 上，私鑰則放在 Form 伺服器。

到 Form 伺服器端開啟 Git 版本控管系統，輸入指令 ssh-keygen -t rsa -b 2048，如圖 2 所示，這會生成一對公鑰及私鑰[3]。參數 -t 是指定要創建的密鑰類型(加密演算法)，可用的選項為 rsal、rsa 與 dsa;若想要金鑰長度更長，可在參數 -b 後指定，最小長度為 768。

```
$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users
/Administrator/.ssh/id_rsa_bitbucket
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

圖 2 SSH key 創建過程畫面

4. 接著系統會詢問金鑰檔名，建議在預設檔名(id_rsa)後加上 Git Server 識別字。

```
Enter file in which to save the key
(/c/Users/Administrator/.ssh/id_
rsa):
/c/Users/Administrator/.ssh/id_r
sa_bitbucket
```

5. 出現以下訊息即成功生成。

```
Your identification has been saved
in
/c/Users/Administrator/.ssh/id_r
sa_bitbucket.
Your public key has been saved in
/c/Users/Administrator/.ssh/id_r
sa_bitbucket.pub.
```

6. 在~/.ssh/ 底下新增 config 檔案，設定 ssh 連線到指定的服務器時所需的私鑰，如圖 3 紅框部分，就是剛剛生成的私鑰。

```
Host bitbucket.org
 IdentityFile ~/.ssh/id_rsa_bitbucket
```

圖 3 Config 檔設置內容畫面

所以當 Git 連線至 bitbucket.org 時，因 config 檔案的設定，會使用 ~/.ssh/id_rsa_bitbucket 私鑰進行解密。

7. 將先前生成的公鑰上傳至 Bitbucket。登入 Bitbucket 網站，如圖 4 所示，點選右上角頭像，選擇 Manage account，進入 Manage account 頁面，點選頁面中左欄裡的 SSH Keys。

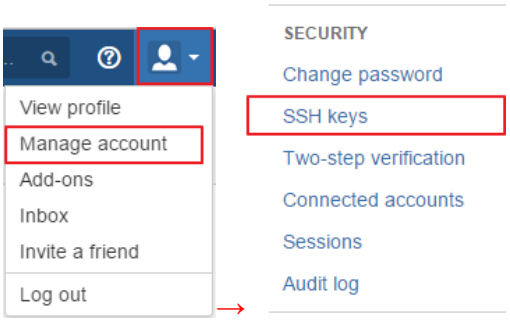

圖 4 Bitbucket SSH keys 設置路徑

8. 新增 SSH key，如圖 5 所示，點選 Add key，輸入 Label，貼上剛才生成的公鑰到 Key 欄位，之後按下 Add key。

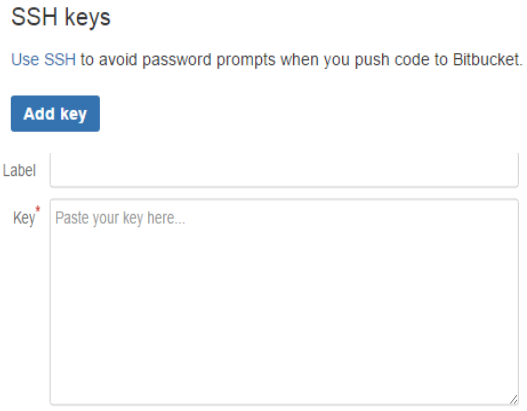

圖 5 Bitbucket SSH keys 設置畫面

9. 完成後，再次執行 git_command，便不需要手動輸入密碼，如圖 6 所示。

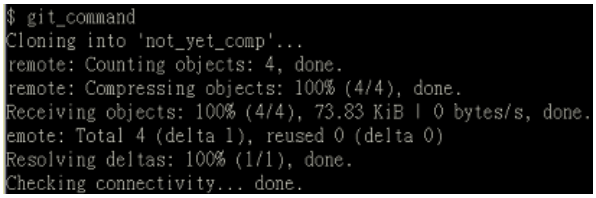

圖 6 輸入 git_command 執行畫面

(b) 批次編譯更新後的 .fmb 檔

設定於特定時間都會針對差異檔執行編譯的批次檔。

1. 建立一批次檔 comp.bat，輸入以下指令:

```
1:  cd c:/Forms_Proj/key_case
2:  git fetch origin
3:  git checkout origin/master
4:  git checkout master
5:  git diff --name-only
    master..origin/master >
    changed_fmb.txt
6:  for /f "tokens=1* delims=.fmb" %%i
    in (changed_fmb.txt) do (echo %%i
    >> changed.txt)
7:  git merge origin/master
8:  if exist C:\Forms_Proj
    \key_case\changed.txt (
9:    for /f "tokens=1*" %%f IN
    (changed.txt) do (
10:    if exist
    C:\Forms_Proj\key_case \%%f.fmb (
11:    frmcmp Module=key_case/%%f.fmb
    Userid=user/password@sid
    compile_all=yes
    window_state=minimize
12:   copy /y C:\Forms_Proj\
    key_case\%%f.fmb
    C:\Forms_Proj\cyba\
13:   copy /y C:\Forms_Proj\
    key_case\%%f.fmx C:\Forms_Proj
14:   copy /y C:\Forms_Proj
    \key_case\%%f.err C:\Forms_Proj
15:     )
16:   )
17: )
18: del /q C:\Forms_Proj
    \key_case\*.txt
```

上述程式碼中：

#1: 切換目錄至 c:/Forms_Proj /key_case。

#2~6: 使用 fetch 指令取得 Bitbucket 上專案的最新版本，找出異動檔案，並放到 changed_fmb.txt 中。

#7: 合併本地端及遠端的分支到最新版本。

#8~11: 若有異動檔案，取出檔名並執行 frmcmp 指令進行編譯(#11)。參數 Module：要編譯的 fmb 檔；Userid 為使用者帳密；Compile_all：編譯所有 PL/SQL 程式碼；Window_state：視窗狀態，可用的選項為 Normal、Maximize、Minimize。

#12~14: 編譯結束之後會出現 fmx(編譯檔)，我們將編譯好的 fmb 檔、fmx 檔與 err 檔複製到部署 Forms 的目錄底下。

#18: 最後刪除異動檔名檔。

(c) 設定 Server 端排程進行自動更新及編譯

在作業系統中設定自動排程，在特定的時間自動完成部署。

1. 先將 Git 的執行檔的路徑加入環境變數[6]。點選「我的電腦」右鍵，依序選取: 內容->進階系統設定->環境變數，設定系統中的 Path 環境變數，如圖 7 所示。

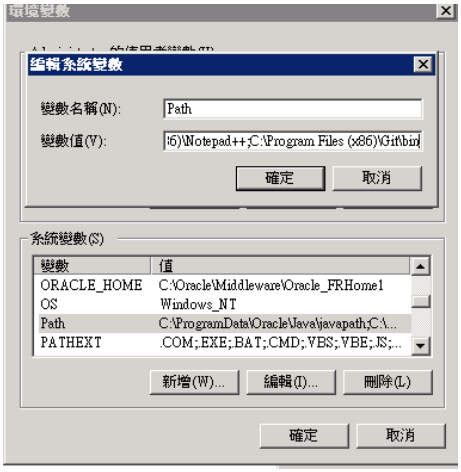

圖 7 設置 Git 環境變數

2. 設定自動排程[6]。先開啟控制台，點選「系統及安全性->系統管理工具->工作排程器」，開啟後點選右方「建立工作」。

3. 在「一般」頁籤中輸入「名稱」與「描述」，接著設定安全性選項為「不論使用者登入與否均執行」，如圖 8 所示。如此一來，當觸發工作時，即使執行排定工作的帳戶未登入，都會執行。

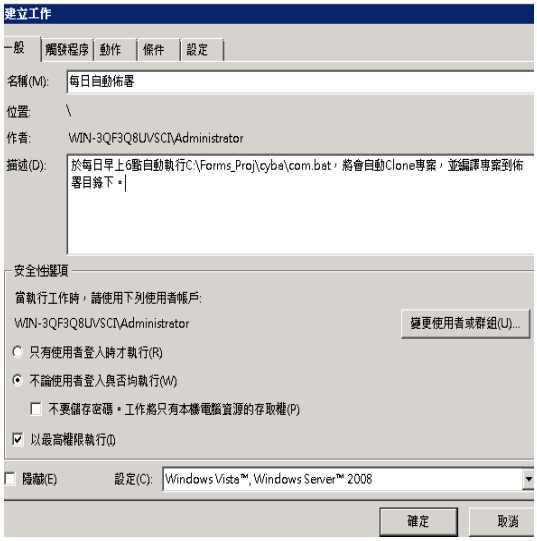

圖 8 建立工作排程

4. 切換頁籤至「觸發程序」，如圖 9 所示，點選「新增」，選擇開始工作為「依排程執行」，設定為「每天」，按下確定。

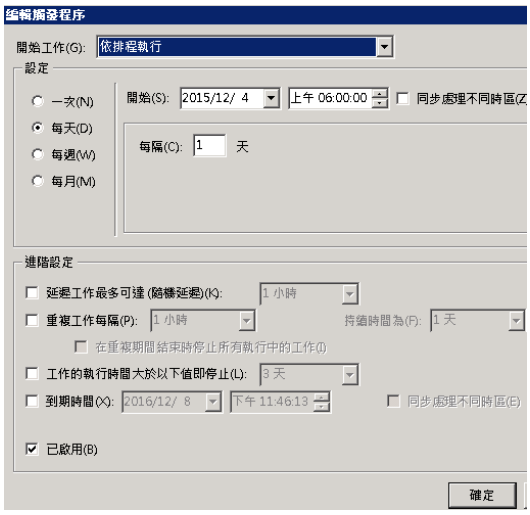

圖 9 設定觸發程序

5. 切換頁籤至「動作」，點選「新增」，設定批次檔的路徑及名稱，如圖 10 所示，按下確定。

圖 10 設定執行程式

6. 回到「建立工作」視窗後，再按下確定，如圖 11 所示，系統要求輸入用來執行此工作的使用者帳戶資訊，輸入完後按下確定。

圖 11 設定帳戶資訊

設定完成之後，將來只要 Server 是開機著的狀態，就會每天自動執行部署。

在由 Git Server 更新檔案時，如果出現以下訊息：

```
Permission denied <publickey>. fatal:
Could not read from remote repository.
Please make sure you have the correct
access rights and the repository
exists.
```

表示設定的私鑰無法正確讀取。

有兩種解決方式。第一種，在 Git 程式安裝目錄下的 .ssh 目錄放置私鑰並使用其

預設檔名 id_rsa[9]。將預設檔名 id_rsa 私鑰複製到 C:\Program Files (x86)\Git\.ssh。因為使用批次檔執行 Git 指令時，會使用的 Git 安裝目錄下的.ssh 中的預設私鑰。第二種是在系統中新增使用者變數 Home=%USERPROFILE%，如此 Git 程式會使用使用者個人目錄下的 .ssh 目錄中的私鑰[10]。

3.4 實作展示

1. 如圖 12 所示，索引 aff0df2 是於 Bitbucket 上的最後一筆 Commit，也就是最新的版本。

圖 12 Bitbucket Commit 之前畫面

2. 於 Local 端電腦進行 push，新增一個.fmb 檔案至 Bitbucket。

最新的 Commit 索引已從 aff0df2 更變為 ac44804，如圖 13 所示。

圖 13 Bitbucket Commit 之後畫面

3. 當 Server 端系統於隔日 6 點進行自動部署時，會找出差異檔案：ORDG08，如圖 14 所示。

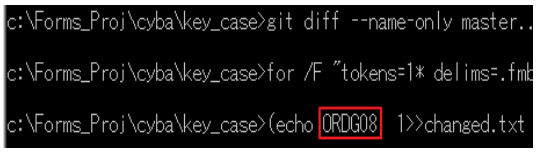

圖 14 找出差異檔畫面

4. 將本地端的分支與遠端分支合併，如圖 15 所示。

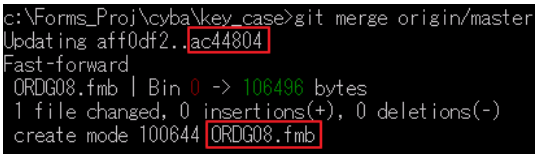

圖 15 合併分支

5. 如圖 16 所示，只針對此 ORDG08.fmb 檔案進行編譯，產生 ORDG08.fmx。

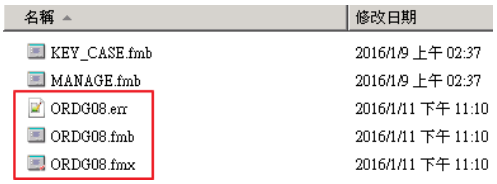

圖 16 針對差異檔編譯

4. 結論

一般 Forms 程式的部署流程使用 FTP 上傳原始碼檔至遠端 Server 的指定目錄，再於遠端 Server 執行編譯。使用 FTP 上傳檔案進行部署時，不但操作耗時且可能發生版本管理上的失誤。此機械式及耗時的操作可以改為自動部署，由 Server 端系統於固定時間更新最新異動的檔案並執行編譯，不但減省了原本需至 Server 端進行繁雜程序的部署時間，達到快速部署，也降低人為介入的機會，資訊人員將有更多時間能夠處理其它事情。

為了實現 Oracle Forms 程式的自動化部署，需設定三大部分：一、需進行專案複製，以及設定 Git Server 的身份鑑別功能，達到省去手動輸入密碼的步驟。二、批次編譯更新後的原始檔，利用批次檔，針對新版檔案執行編譯。三、設定排程進行自動至 Bitbucket 下載最新版本，並在 Server 端執行自動編譯完成自動部署。

後續將發展 Linux 系統上的自動化部署命令稿並將編譯的 log 檔以電子郵件的方式通知應用程式管理員，減輕應用程式管理員的工作負擔。

參考文獻

[1] 陳惠貞，***新趨勢網路概論(第三版)***，碁峰資訊股份有限公司，2014。

[2] 張顯洋，***DOS 批次檔案設計技巧(第三版)***，第三波文化事業股份有限公司，1993。

[3] 產生 SSH 給 Bitbucket/Github 使用（2012）。2015 年 10 月 28 日，取自 https://nyllep.wordpress.com/2012/01/21/sshkey-for-bitbucket/。

[4] 如何利用批次檔(Batch)讀取指令執行的結果或文字檔案內容(2010)。2015 年 12 月 23 日，取自 http://blog.miniasp.com/post/2010/09/24/How-to-parse-text-from-file-or-command-using-Batch.aspx。

[5] 如何讓 Git 僅匯出在特定版本中新增或修改過的檔案(2014)。2015 年 12 月 23 日，取自 http://blog.miniasp.com/post/2014/04/02/Git-Export-Only-Added-Modified-Files.aspx。

[6] 使用 Git & Windows 工作排程來定期差異備份(2013)。2015 年 11 月 12 日，取自 http://coding.bang.tw/Article/42。

[7] Oracle Fusion Middleware Forms Services Deployment Guide (2015). Retrieved January 11, 2016, from https://docs.oracle.com/middleware/11119/classic/deploy-forms/intro.htm#FSDEP728

[8] Pro Git[2nd Edition] (2014). Retrieved January 10, 2016, from https://progit2.s3.amazonaws.com/zh-tw/2015-07-27-05234/progit-zh-tw.604.pdf

[9] Permission denied (publickey) Error using git on windows 7 [Msg 8](2012). Retrieved December 11, 2015, from http://stackoverflow.com/questions/2127104/permission-denied-publickey-error-using-git-on-windows-7

[10] Permission denied (publickey) issues with Git on Windows(2013). Retrieved December 11, 2015, from http://catch404.net/2013/01/permission-denied-publickey-issues-with-git-on-windows/

[11] Oracle E-Business Suite Developer's Guide[Release 12.1](2010). Retrieved January 25, 2016, from https://docs.oracle.com/cd/E18727_01/doc.121/e12897/T302934T391729.htm

[12] Using Oracle Mobile Framework to Modernize Oracle Forms (2005). Retrieved January 25, 2016, from http://www.slideshare.net/AuraPlayer/using-oracle-mobile-framework-to-modernize-oracle-forms