國 立 交 通 大 學

電機學院與資訊學院 資訊學程

碩 士 論 文

CPSMS-跨平台主機管理系統

CPSMS-Cross Platform Server Management System

研 究 生：郭明傑

指導教授：蔡文能　教授

中 華 民 國 九 十 五 年　三 月

CPSMS-跨平台主機管理系統

CPSMS-Cross Platform Server Management System

研 究 生：郭明傑　　Student：Ming-Chieh Kuo

指導教授：蔡文能　　Advisor：Wen-Nung Tsai

國 立 交 通 大 學

電機學院與資訊學院 資訊學程

碩 士 論 文

A Thesis

Submitted to Degree Program of Electrical Engineering Computer Science
College of Electrical Engineering and Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
In
Computer Science
March 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年三月

CPSMS-跨平台主機管理系統

學生:郭明傑　　指導教授:蔡文能

國立交通大學電機學院與資訊學院 資訊學程碩士班

摘要

穩定有效率與簡潔主機管理是一項艱鉅又繁雜的工作，隨著現今企業資訊化的普及程度與日聚增，主機管理人員必須要有大量完整的資訊，才能順利達成即時的任務。然而管理或收集不同平台主機上的資料，並利用其特性來協助增進主機的利用率成為主機管理人員的另一項挑戰。

企業內常常因為不同的軟硬體解決方案需求，會使用不同的硬體與作業系統的主機，監控這些異質的平台的主機成了相當煩雜的日常管理工作。如何有效的整合這些不同平台主機的資訊、透過單一的整合介面進行主機的管理、提供清晰易懂的報表給主機管理人員、並在主機或服務發生問題時、及時通知相關的管理者進行處理，也可利用資料收集協助管理者作為事前的預防與事後異常分析的參考，即成為本論文之主要目的。

最後我們藉由資料庫來建置一套跨平台主機管理系統，整合企業內不同的平台主機所提供的資料，讓管理者可以非常容易的取得所需的資訊以做出最有效率的判斷，另外也可透過外掛模組的方式與微軟的 SMS 作結合，並透過本系統達到即時的錯誤回報。

關鍵字: 資料庫、主機管理

CPSMS-Cross Platform Server Management System

Student: Ming-Chieh Kuo Advisor: Wen-Nung Tsai

College of Electrical Engineering and Computer Science
National Chiao Tung University

Abstract

A steadily efficient and succinct server management is a tough challenge. Server administration needs a large amount of intact information to fix an issue immediately and smoothly since the amount of enterprise's information has been increasing dramatically, It has, therefore, become another challenge for server administration to manage or collect different platform data of servers and to improve server utilization.

Monitoring these heterogeneous platforms is yet another complicated task since enterprises would tend to use different hardwares or operation systems for solutions.

The main purpose of this thesis is to integrate these heterogeneous platforms by intergrating interface to provide administrators with accurate and clear report as well as notify them in time when a system goes down; and to help administrators prevent from systems going down as well as provide them with analysis by collecting the data through the systems.

Finally, we build an Cross Platform Management System to intergraded different platform Server by database. Through the information which they easy to get, administrators can make effectively policies, configure server simply and report abnormal server status.

Keywords: Database, Server Management

誌謝

感謝指導教授蔡文能老師與系上老師來的諄諄教誨，讓我在研究的過程中學到了嚴謹的研究方法和正確的學習態度，使我獲益匪淺。尤其感謝蔡老師在我的論文研究上給予的啟發與諸多指導，並且感謝口試委員周勝鄰組長與葉義雄教授在百忙中抽空幫我口試，並對我的論文方向與寫作技巧提供寶貴的意見與協助，才使得這篇論文能夠順利完成，您們的教導讓我獲得許多寶貴的專業知識與人生經驗，對我在工作與未來的規劃上皆有莫大的幫助。

也要特別感謝我工作上的長官：鄧經理，呂經理，以及夥伴長輝、宗錨、育平、勝雄、國葆，您們給予我學習期間工作及精神上的最大協助。同時也要感謝威德、俊霖、正忠、貞云同學的關懷照顧，以及國強、嘉宏、志村的指教，謝謝您們給予我的支持與鼓勵，讓我在工作學業兩頭忙的日子裏仍然得以感受到友情的溫馨。

最後，要感謝我的老婆惠升，謝謝您幫我搞定三個可愛的兒子(茂宏、育璋、育安)，不斷的給予我支持與鼓勵。也感謝爸媽給予我受教育的機會，沒有您們就不會有今日的我。

謹以此論文獻給所有關心我、幫助過我的人，謝謝你們！

郭明傑

謹誌於交通大學電機學院與資訊學院在職專班資訊組

2006 年 3 月

目錄

CPSMS-跨平台主機管理系統 I

CPSMS-CROSS PLATFORM SERVER MANAGEMENT SYSTEM II

誌謝 III

目錄 IV

表目錄 VII

圖目錄 VIII

第一章 緒論 1

1.1 研究動機 3

1.2 研究目的 4

1.3 論文架構 5

第二章 背景知識 7

2.1 管理平台之發展 7

2.2 常用主機效能監控工具 10

2.2.1 top、vmstat、free、df 10

2.2.2 Sysstat 11

2.2.3 CIM 11

2.2.4 WMI 12

2.3 簡易網管通訊協定(SNMP) 13

2.3.1 SNMP 的版本 13

2.3.2 SNMP 架構 14

2.3.3 MIB 簡介與架構 15

2.4 其他相關工具 15

2.4.1 Perl 16

2.4.2 JSP 16

2.4.3 LDAP (Lightweight Directory Access Protocol) 16

2.5 資料庫 18

2.5.1 Oracle 簡介 18

*2.5.2 SQL * Loader 簡介 19*

第三章 相關研究 20

3.1 OPENNMS 20

3.1.1 系統設計理念及架構 *20*
3.1.2 系統評估 *21*
3.2 HP OVO (HP OPENVIEW OPERATION) 22
3.2.1 系統設計理念及架構 *22*
3.2.2 系統評估 *23*
3.3 以知識庫為基礎之智慧型伺服器自動監控系統(KISMS) 24
3.3.1 系統設計理念及架構 *24*
3.3.2 系統評估 *25*
第四章 跨平台主機管理系統(CPSMS) **26**
4.1 CPSMS 系統簡介 26
4.2 CPSMS 系統架構 27
4.2.1 管理介面程式(Management Scripts) *29*
4.2.2 資料收集模組(Data Collection Module) *30*
4.2.3 資料處理模組(Data Processing Module) *32*
4.2.4 報表產生模組(Report Generation Module) *32*
4.2.5 組態管理模組(Configuration Module) *33*
4.2.6 警訊模組(Alerting Module) *34*
4.2.7 外掛模組(Plug-ins Module) *36*
4.2.8 認證模組(Authentication Module) *36*
第五章 系統實作、評估與比較 **37**
5.1 環境及開發平台 37
5.2 程式開發 38
5.2.1 資料收集模組之程式 *40*
5.2.2 資料處理模組之程式 *42*
5.2.3 報表產生模組之程式 *42*
5.2.4 組態管理模組之程式 *44*
5.2.5 警訊模組之程式 *45*
5.2.6 外掛模組程式 *47*
5.2.7 認證模組程式 *47*
5.3 系統評估 48
5.4 系統比較 49
第六章 結論與未來研究方向 **51**
6.1 結論 51
6.2 未來研究方向 52
參考文獻 **53**
附錄一 **55**

附錄二 **57**

附錄三 **59**

附錄四 **60**

附錄五 **61**

附錄六 **62**

附錄七 **63**

表目錄

表 一、 一般常用資料庫系統比較表 18
表 二、 CPSMS 組態預設值表 34
表 三、 主機管理系統比較表 49
表 四、 支援作業系統平台表 51
表 五、 支援硬體平台表 51

圖目錄

圖一、 主機管理成本趨勢圖 3
圖二、 管理架構示意圖－集中式 8
圖三、 管理架構示意圖－階層式 8
圖四、 管理架構示意圖－分散式 9
圖五、 TOP 10
圖六、 VMSTAT、 FREE 、DF 11
圖七、 WMI 架構圖 12
圖八、 WMI 命令列範例 13
圖九、 SNMP 系統架構圖 14
圖十、 MIB NAMING TREE 15
圖十一、 LDAP TREE 架構圖 17
圖十二、 ORACLE 系統架構圖 19
圖十三、 SQL *LOADER 系統架構圖 19
圖十四、 OPENNMS 設計架構圖 21
圖十五、 HP OVO 架構圖 23
圖十六、 KISMS 系統架構圖 24
圖十七、 CPSMS 示意圖 27
圖十八、 CPSMS 架構圖 29
圖十九、 資料收集模組架構圖 31
圖二十、 資料處理模組架構圖 32
圖二十一、 報表產生模組架構圖 33
圖二十二、 組態管理模組架構圖 34
圖二十三、 警訊模組架構圖 35
圖二十四、 SMS 警訊發送示意圖 35
圖二十五、 認證模組架構圖 36
圖二十六、 CPSMS 網路環境架構圖 37
圖二十七、 CPSMS 主畫面 39
圖二十八、 CPSMS 主機設備偵測結果畫面 40
圖二十九、 SNMP 效能資料(CPU) 41
圖三十、 CPSMS HEARTBEAT STATUS 42
圖三十一、 CPSMS 報表查詢管理畫面 43
圖三十二、 CPSMS CPU 使用率趨勢報表畫面 43
圖三十三、 CPSMS 監控項目設定畫面 44
圖三十四、 CPSMS 警訊發送方式設定 45

圖三十五、 CPSMS 警訊通報人員資料設定 46
圖三十六、 CPSMS 警訊發送條件設定 46
圖三十七、 CPSMS 警訊發送(EMAIL) 46
圖三十八、 CPSMS 警訊發送(SMS) 46
圖三十九、 JSP 透過 LDAP 取得 WINDOWS AD 認證資料 47

第一章　緒論

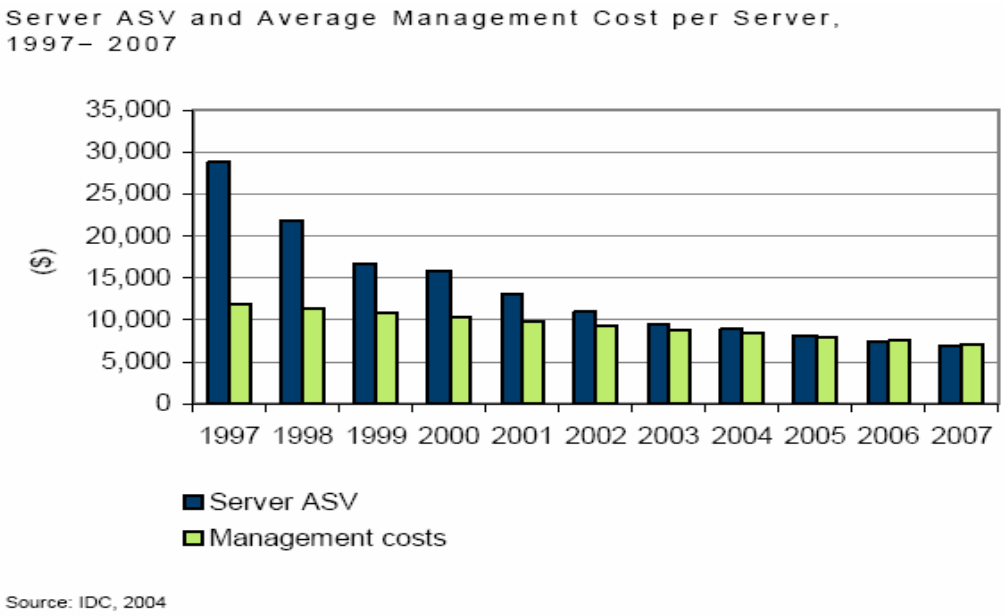
隨著企業營運擴大，業務內容愈趨複雜，企業內部資訊系統為了支援業務發展也變的更為複雜龐大。資訊系統管理人員需要面對各種不同的解決方案，例如 B2C 的 Web server、供應商的 SCM、企業內部的 ERP、資料倉儲系統、證券業的交易系統、銀行業的 core banking 系統等，且隨著企業不斷擴張，營業據點可能分散台灣各地或國外，例如：IT 系統分散在台北、台中、台南等地，考量企業永續經營，又在各個廠區建置備援系統。面對這麼多解決方案，資訊系統管理人員除了要忙著建置這些系統，最重要的、也是佔了資訊人員最多時間的工作就是維護這些系統。每一種解決方案對企業營運而言都是非常重要，若任一系統發生問題，皆會產生非常嚴重的問題。幾年前，倘若系統停機 1 小時，或許不會對企業造成嚴重影響，在今日高度競爭的環境下，可能停機 5 分鐘，客戶就跑了，所以如何確保系統的穩定性和可用性，是一個非常重要的議題。目前所有資訊管理人員所面臨的是為了建置各種不同解決方案所採用不同的平台造就了企業整個 IT 架構的複雜性;譬如說 Windows Server 作為 Web Application server、UNIX Server 作為 ERP 系統、Linux Server 作為 CRM 系統，Database Server 並執行最重要的應用系統；再加上網路設備、儲存設備以及公司內部使用的系統，如 E-mail server 等。

在這種異質平台組成的 IT 環境中，對企業資訊系統管理工作是一個很大的挑戰，面對這些主機管理的難題，廠商通常會提供不同的軟硬體解決方案，包含監控、效能分析，資產管理系統等。但由於其價格實在不是一般企業負擔的起，也有些開放原始碼(Open Source)的軟體，但是其功能不見得合乎一般企業的需求，若要修改其實有一定的難度,傳統的方式是由管理者自行利用內建的工具與 Shell 撰寫一些監控的程式，但這種方式移植性太差並不一定能在不同平台的主機上使用。這些解決方案在表面上或許可以達到其應有的功效。但是相對的也帶來另一項主機管理的課題。如購買進來的軟硬體設備，其種類及品牌通常不同，要學會了解其中一種設定及與操作就要花上許多的時間，還必須想辦法整合各種不同主機上的資訊，才能真正的達到管理的目的。另外企業內有很多主機其使用率不是過高就是太低，無法有效的運用其效能，使用率過高的機器常常會因為效能達到瓶頸，常常招致使用者的抱怨，使用率低的機器由於其效能並未被充分利用，造成資源的浪費。

本研究主要是以我在主機管理方面所累積的經驗，並與資料庫結合，利用單一的介面提供主機管理人員查詢其所負責的主機設備狀況，並可藉由已經設計好的報表顯示出現有的主機狀態，方便管理者即時掌控其負責系統的相關資訊，更可藉由效能資料的收集提供管理者對於所負責的主機做更有效的利用與調度，並可藉由自行設定所需的監控項目與發送警訊的方式與條件，及時的發送相關的警訊，可達到預防與及時警告的效果。除了能達到主機管理資料統一的目的外，也可以透過與資產系統的結合讓使用者能獲得更多的主機資訊(取得價值，殘值…)，且由於採用 Web Based 作為介面，使用者不需另外安裝軟體，也不需直接連接至所負責的主機，可以幫助不熟悉主機操作的同仁輕易設定監控與發送警訊項目。

1.1 研究動機

依據 IDC 2004 年的統計資料，(如附圖一) 由於半導體與電子技術的進步，使得主機硬體成本的降低也連帶造成企業內主機的數量不斷的增加，但相對的管理的成本並未隨之增加。



圖一、主機管理成本趨勢圖

系統管理者每天為了如何有效充分利用資源而奔波忙碌。如果您能在所掌控的多項資源、以及使用資源的人與程式間找到平衡，那麼不但可以省下大筆經費，還可以讓使用者滿意，更能充分發揮機器的效能，目前市面上雖然已經有一些現成的商業軟體與 Open Source 的軟體可以做到上述的功能，但是有底下幾個問題：

1. 整合性的控管軟體價錢昂貴且學習困難

目前市面上有許多整合性的主機管理系統，這些系統為了能達到全面性監視及控管的功能，一直以來皆被設計的相當複雜，主機管理人員就算是經過相當一段長時間的學習與訓練，還不能夠完全的掌握系統中功能的使用。而且每套系統為了其商業考量及目的，皆將很多管理功能拆開來銷售，造成購買成本無限的增加。

2. 商業化的軟體很難做到客製化

目前市面上的主機管理系統，為了取得廣大的客戶群，便將一般通用之主機管理功能包裝成一套系統，以符合大多數人的一般需求，相對來說所提供之功能常常無法達到單一客戶之特殊需求，若需客製化則需另外負擔費用，且不一定可以達成。

3.Open Source 軟體並不一定合乎需求

目前市面上的雖然也有一些 Open Source 的軟體，雖然為免費使用但其功能不見得完全合乎企業的需求，且因為它並不是一般的商業軟體，並無專屬廠商提供支援與服務，所以只能靠自己客製化。

4.軟體並非跨平台

有些系統管理軟體只能在特定作業系統或硬體平台使用(如微軟的 SMS 、MOM 只能用於 Windows Platform，IBM Director 只有支援 IBM 所生產與設計的主機)，無法滿足企業不同平台作業系統管理所需。

1.2 研究目的

我們希望能設計一套能跨越不同作業系統平台，操作介面簡單且方便擴充之主機監控與管理系統，讓管理者能以更簡便與有效率的監控與管理其所屬的系統，充份的發揮機器應有的效能，為達到上述的要求，我們所設計的系統應包含下列幾項功能:

1.整合式操作介面

將所收集的資訊利用單一而簡潔的畫面來提供管理者與使用者管理與查詢其所要的資訊。以管理者角度為出發點而設計出的報表系統，少去繁雜的選項功能，將可有助於管理人員很清楚主機的狀況，且能以簡單的方式設定所需監控的項目，並可藉由登入之權限控管設定一般使用者所能使用的功能。

2.主機效能管理

我們可藉由收集主機的各種效能，如 CPU，Memory，Disk I/O，File System 使用率等等..，將資料彙整並存入資料庫中，我們可以將這些收集到的資訊作成各種不同的報表，有助於更清楚的知道主機的運作狀況，也可藉由一些預先設定好的參數有助於及早發現系統的運作是否有異狀，與更靈活的調配主機的資源。

3.主機資源管理

我們將主機的相關軟硬體資訊收集分類，並可與其他系統管理軟體收集到的資訊作整合，當系統有問題時我們能很快速的找到相同性質的主機，方便我們找到替代的主機，也可以搭配效能管理對一些類似規格與功能的主機做一些效能的比較，與方便系統管理者作一些資源的調整，有利與整體主機利用率與提升整體的效能。

4.異常管理

當有系統異常狀況發生時管理者可在最短的時間內接到通知，且可依據異常的等級做不同的通知(例如 SMS，Email..) ，並可依其設定通知相關人員，以減少影響之時間與範圍，一般的使用者也可以自行設定接收異常通知。

5.報表功能

系統需將所收集到的資料經過整理後，產生管理者與使用者所需的資訊，方便管理者判斷其所管理主機的狀況，也可利用定期產生的報表提供管理者與使用者參考與使用。

1.3 論文架構

第二章為主機管理相關技術與本篇論文所使用的一些工具介紹。其中包含探討現今在主機管理上所使用到的方法及技術、常用效能監控工具、本文所用到的其他工具。最後會介紹本論文所使用的資料庫的特性與種類。

第三章為文獻探討，第一篇為目前一套熱門的開放原始碼之網

路管理與系統管理系統（OpenNMS）[1]介紹，第二篇為目前市面上功能最強大的商業軟體(HP OpenView Operation) [2]做介紹，第三篇為一份學術論文以知識庫為基礎之智慧型伺服器自動監控系統(KISMS) [19]介紹並提出不同的論點。

第四章為系統架構之解說，本系統共分為七個模組：收集各種不同主機資訊的資料收集模組(Data Collection Module)、負責將所收集的資料作處理並存入資料庫中的資料處理模組(Data Processing Module)、產生報表的報表產生模組(Report Generation Module)、控制 Agent 與相關設定的組態設定模組(Configuration Module) 、可自動發送警訊的警訊模組(Alerting Module) ，與其他軟體的作資料交換的外掛模組(Plug-in Module) ，負責認證與權限控管的認證模組(Authenticate Module)。並使用資料庫儲存透過 Agent 所收集並經處理過的資料。

第五章為系統之實作解說，內容包含實作環境、平台等等，並以實作出來的畫面展示本系統之成果。最後與第三章所提到的三套主機管理系統做比較。

第六章為本論文之結論與未來研究方向，其中勾勒出本系統對於主機管理系統之設計開發所做的貢獻並針對未來研究方向之建議。

第二章　　背景知識

市面上的常見的主機管理系統有些是由特定硬體廠商開發只支援自己製造的硬體(如 IBM 公司的 director 和 HP 公司 insight management)或特定的作業系統廠商開發只支援自家產品(Microsoft Operation Management),當然也有較支援較完整的主機管理系統如 HP Openview，與 open source 的 OpenNMS 等等..

一般而言由特定硬體廠商與軟體廠商所開發出來的系統，對於本身所設計與製造的硬體或作業系統的支援能力還算不錯，但是若遇到不同的硬體或作業系統平台就無能為力了，這樣的系統對於一個系統管理員而言幫助實在有限，如何能有效管理各種不同平台的主機，簡化管理人員的負擔並能於有異常發生時提出警訊是本文的重點。所以本章節以管理者的角度做出發，探討身為一個主機管理人員及發展一套主機管理系統所必須了解的背景知識。

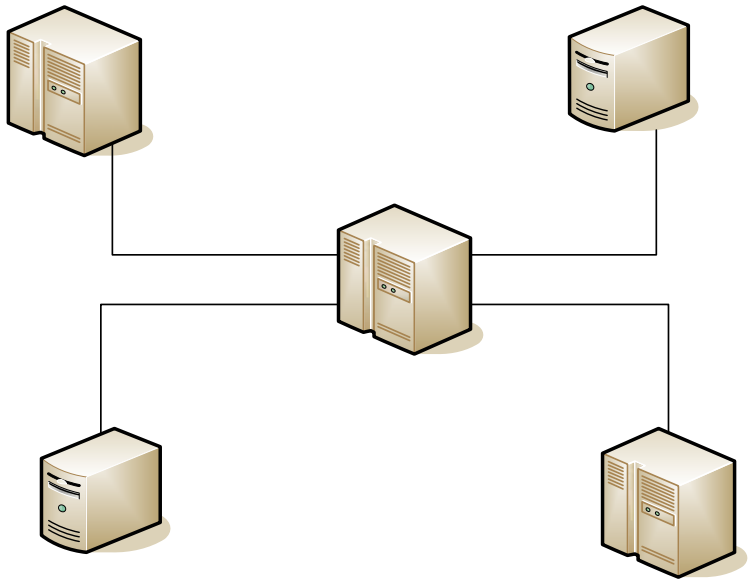
2.1　管理平台之發展

早在八零年代以前，主機與網路管理相關的議題早已被提出。一般說來，一個好的管理策略，必須要有明確的管理對象，再從此對象設定明顯且可行的管理方向及目標。而管理人員就可依循此制定的方向，來制定相關的管理流程，並以資訊技術作為管理的橋樑，來達成管理的目標。

以資訊技術作為工具,所發展出來的各種主機管理平台依其提供的服務方式及管理架構的不同而有所差異。筆者將目前所知的主機管理模型整理後，將其分為三類：

集中式架構 (Centralized Architecture)

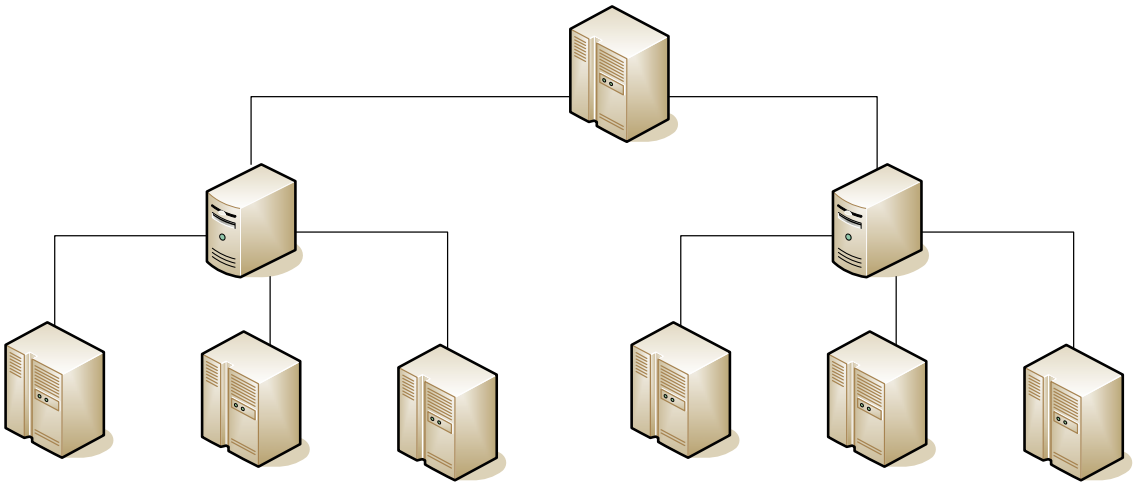
集中式架構是最早出現的管理架構，如圖二所示，在此種架構中，所有管理操作所需的功能及運算皆集中在一台大型主機上。其運作的管理通訊協定最具代表性的便是簡易網管協定之第一版[11]，然而隨著節點的增加及管理資料的暴增，這種管理模型已不敷使用者的需求。不過此種架構的好處就是單純且容易管理。



圖二、 管理架構示意圖－集中式

階層式架構 (Hierarchical Architecture)

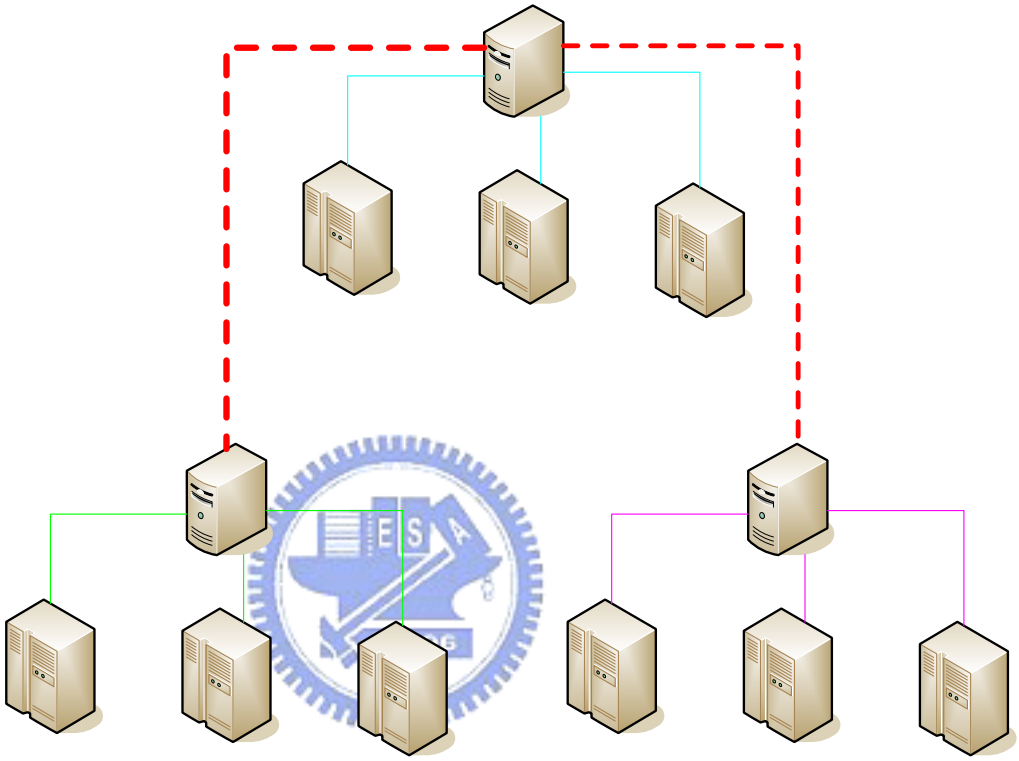
階層式架構 如圖三 所示在這種架構中使用階層的模式來分擔管理工作，並可藉著調整階層的大小及深度而適用於管理不同大小或區域的主機群組，例如我們可在 DMZ 區設立一個主機負責收集 DMZ 區內所有主機的資訊與負責轉送管理的訊息。此種架構解決了集中式架構的主要問題，但卻仍然有其缺陷存在，例如：管理操作仍集中在管理者端，且所定義之管理服務無法動態的進行修改等問題。



圖三、 管理架構示意圖－階層式

分散式架構 (Distributed Architecture)

為了能解決階層式架構所衍生的問題，於是提出了分散式架構，如圖四所示也就是將管理主機分散成很多台，每一台管理主機只負責一部分的管理工作，資料可藉由同步與複製達到一致性，如此可大幅減少管理主機的負載，也可避免因單一主機故障造成服務中斷。



圖四、 管理架構示意圖－分散式

2.2 常用主機效能監控工具

一般而言影響主機的效能主要有底下幾項(1)CPU Utilization、(2)Memory Usage、(3)Disk I/O，Unix Like 的作業系統中有一些作業系統內建的程式可以用來收集這些資訊，本節將針對這些常見的工具做一些簡單的說明。

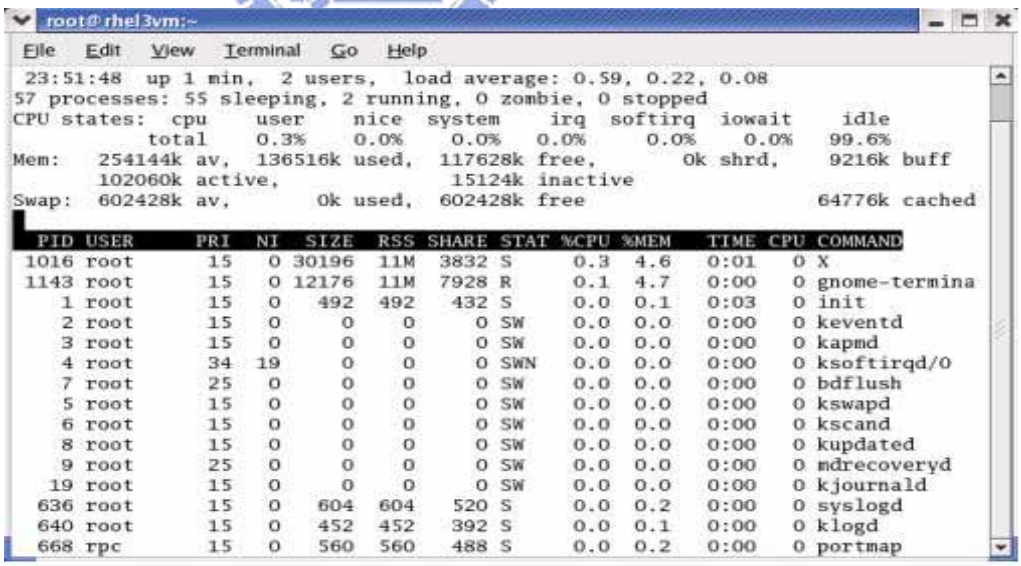
2.2.1 top、vmstat、free、df

1. top:利用這個指令可以用來監視目前的 process state，可以藉此了解系統資源的使用狀況，而且可以根據 **CPU 和 memory** 的使用狀況**排序**。

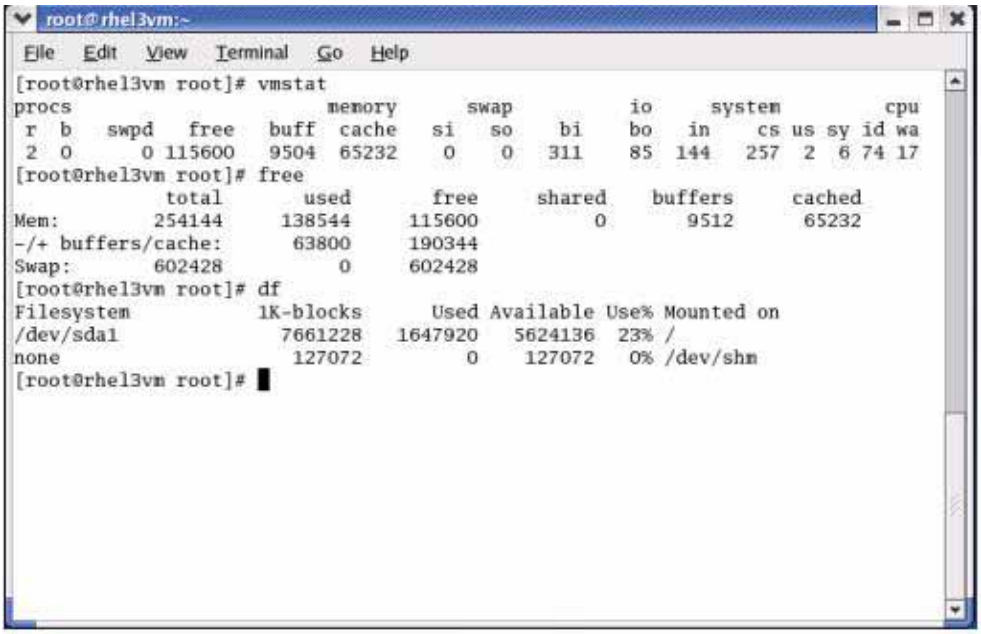
2. vmstat:用於顯示 processes、 memory、,paging、 block IO、traps 和 cpu activity 第一次執行時出現的 information 為最後一次 boot 到目前的平均數值。

3. free:顯示出系統中已使用和未使用的實體、虛擬記憶體統計也會顯示出 kernel 使用的 share memory 和 buffer 的大小。

4. df:顯示檔案系統所使用的空間大小，若沒有給定的檔名，會顯示目前所有 mount 進來的檔案系統情況。



圖五、 top



圖六、 vmstat、 free 、df

2.2.2 Sysstat

Sysstat [13]是一種用於收集與監控效能的工具，在一般常見的 Unix 作業系統中都內建這項工具，主要由 sar，sadf， mpstat，iostat，sa 所組成

1. sar: 用於產生系統效能報表
2. sadf: 產生系統效能的資料
3. mpstat : CPU 的相關統計
4. iostat : 對於 CPU 而言，輸入輸出裝置的 I/O 統計
5. sa: 收集與儲存每天所產生之效能資訊檔案

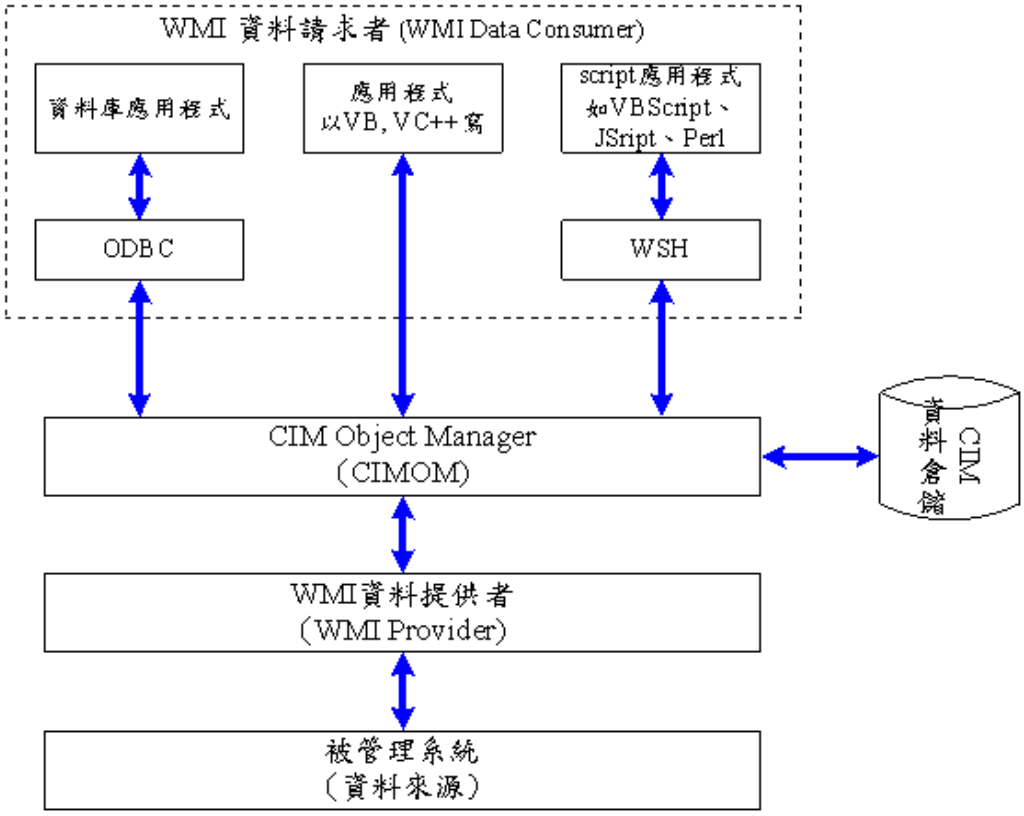
2.2.3 CIM

CIM [14]是一個在企業網路環境中描述所有電腦系統與網路設備整體管理資訊的模型，包括了一組規格書(Specification)及一組 Schema。Schema 定義了 CIM 模型的詳細描述，包括所有被管理物件及其描述方式。而規格書則定義了 CIM 模型與其他管理模型的整合。CIM 自從 1997 年提出 1.0 版以來，目前最新版本為 2.7 版，

您可以透過 http://www.dmtf.org 查閱詳細資料。

2.2.4 WMI

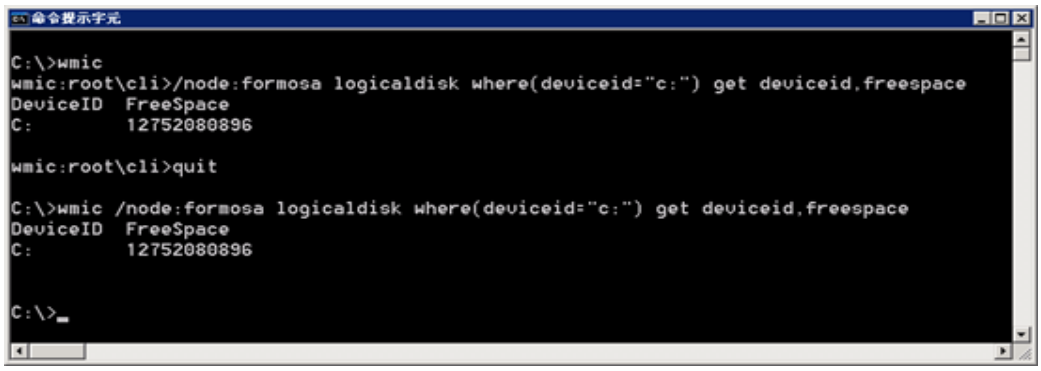
由於 CIM 只是一個物件的模型，各廠商均可以此模型提出各自（但彼此相容）的實做，而 Microsoft 所提出的實做，就稱之為 WMI (Windows Management Instrumentation)，並在 Windows 2000 中首次加入此一管理功能，目前 Windows NT 及 Windows 95/98 等均已先後支援 WMI 的管理



圖七、 WMI 架構圖

除了透過程式來控制 WMI 而達到系統管理的目的，也可以利用 WMIC 這個命令列工具。WMIC 提供了兩種操作模式，一種是互動模式，只要在『命令提示字元』輸入 WMIC 再按下 Enter 按鍵，就會進入互動模式。互動模式會等待管理者進一步的輸入 WMI 命令，此時輸入 /? 則會顯示說明訊息。如圖 八 是個取得特定主機（formosa）C 磁碟機可用空間的簡

單範例，上半部是互動模式的執行結果；輸入 QUIT 可以結束 WMIC 的互動模式。



圖八、 WMI 命令列範例

2.3 簡易網管通訊協定(SNMP)

SNMP 是簡易網路管理通訊協定(Simple Network Management Protocol)的縮寫，主要的目的是用於管理位於 TCP/IP 網路上的各節點，包含有主機(Host)、網路路由器(Router)、網路交換器(Switch)或其他有支援 SNMP 裝置。IETF(Internet Engineering Task Force)建議在 TCP/IP 網路上的所體節點都應該有支援 SNMP 通訊協定，以便於透過網路進行遠端管理。

SNMP 之所以可以在業界如此成功，主要是因為 SNMP 具有以下的一些優點:

1.合乎標準規格 2.廣受工業界支援 3.具備強大擴充能力 4.具備高度可攜性 5.提供了分散式的管理

2.3.1 SNMP 的版本

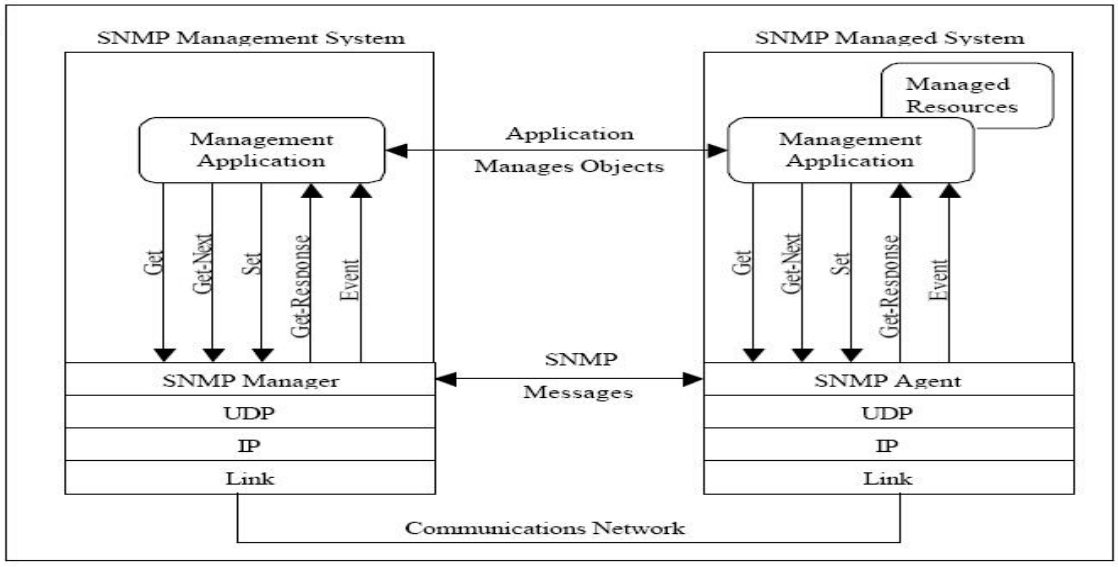
目前主要的 SNMP 版本有 SNMPv1、 SNMPv2 [12]和 SNMPv3。SNMPv1 是一個建議使用的標準通訊協定，主要是定義在 RFC1155 與 RFC1157 中。而 SNMPv2 則是解決了 SNMPv1 的一些程式設計上的問題與定義了一些新的資料型別與新的訊息格式並針對安全的問題作處理，主要定義在 RFC1901~1907。而 SNMPv3 則是針對 SNMPv2 在安全與管理上不足的部份做修改，主要定義在 RFC2570~2575 中。

SNMPv1 是目前被廣泛採用版本，幾乎所有管理相關產品(網路管理、系統管理、應用程式管理)都有實作的支援，但是安全性是 SNMPv1 最大的問題，相信在不久的將來，SNMPv3 將會取代 SNMPv1。

2.3.2 SNMP 架構

SNMP 的主要架構圖請參考圖九 所示。當 SNMP 管理系統(SNMP Management System)對 SNMP 受管理系統(SNMP Managed System)送出 Get 、Get-Next 和 Set 的相關命令準備去取得或設定一個或多個相關資訊時，SNMP 受管理端會傳送出個相對的訊息(Response Message)以完成 Get 、Get-Next 和 Set 的命令。SNMP 受管系統亦會自動送出事件通知(Event Notification)稱為 trap，用來通知 SNMP 管理系統有不正常的事件發生或已經超過 SNMP 管理系統預設值的門檻。

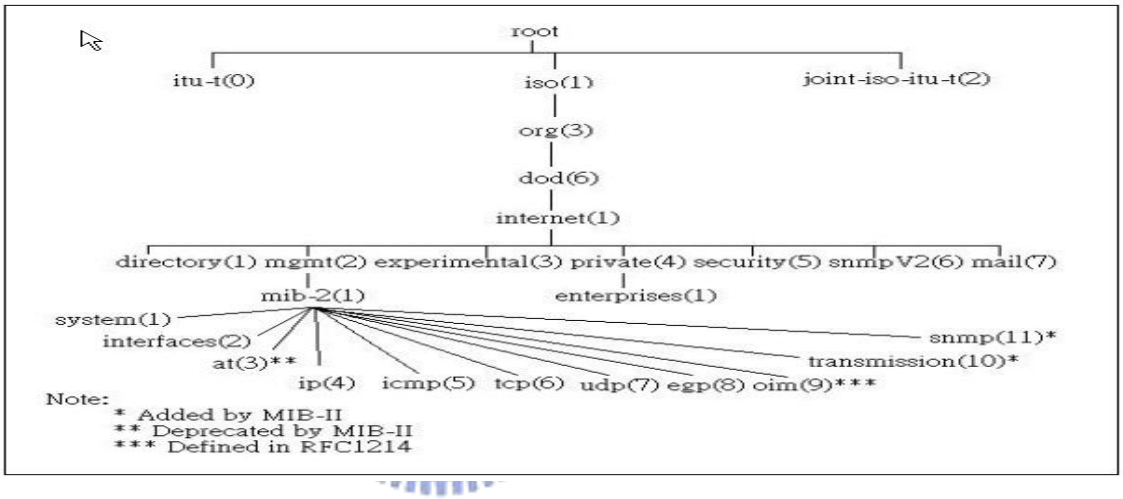
SNMP 使用 UDP 協定中 Port 161 和 Port 162。Port 161 主要的用途是用於接收所有的 SNMP Request 的訊息，而管理系統與代理程式間的溝通也是透過 Port 161。Port 162 主要的目的為傳送 Trap 的資訊。



圖九、 SNMP 系統架構圖

2.3.3 MIB 簡介與架構

管理資訊基礎(Management Information Base，MIB) ，是由代理程式所維護之資訊的資料庫，此資料庫可經由管理系統查詢或設定。因此 MIB 的主要目的便是定義受管物件並且使用。一般而言所謂的受管物件指的就是定義 MIB 的內容，MIB 本身只是一群受管物件的定義也稱為 MIB 物件、資料項、或變數。而對程式設計師來說，MIB 就像是一個參考的文件，有了 MIB 的定義，程式設計師便可以依據這個 MIB 定義撰寫 SNMP Agent 讓系統管理者得以存取受管物件中的資料。



圖十、 MIB Naming Tree

2.4 其他相關工具

本系統由於需使用在不同的作業系統平台上，所以在工具程式的選擇上需考慮可跨平台使用，Perl 是一項不錯的選擇,他不但是跨平台且一般的作業系統已經內建不需另外安裝(Windows 除外)。

另外由於本系統主要是以 Web GUI 作為操作的介面，且須使用到動態方式產生網頁，所以我們也須選要一套 Web 的開發程式，目前市面上有許多的 Web Program 如 ASP 、.Net 、PHP 、JSP.. ，其中 ASP 、 .Net 僅能用於微軟的作業系統，無法跨平台所以我們無法使用，JSP 由於使用 Java 技術當核心對於我們開發有幫助所以我們利用他來當作 Web GUI 開發工具。

2.4.1 Perl

Perl (Practical Extraction and Report Language) ，他原始的目的是用來取代 UNIX 原有的 SED 過隨著版本的改進，功能越來越強，現在的功能已經超乎原先設計時的想像，幾乎任何事都可以做到，也變成每一部 UNIX 工作站必備的標準工具，

Perl 是直譯式的語言，寫好之後，馬上就可以執行，不必像 C 語言必須經過編譯、組譯、連結等冗長的過程，因此開發周期較短，也較為輕鬆。雖然 Perl 是直譯式的，但它的效能不錯，主要是因為 Perl 並非逐列直譯，Perl 執行前會先編譯為一種中間 bytecode，然後再來執行。因此它有直譯式語言開發快速的優點，卻有編譯語言效率的優點。我們可以說 Perl 既是直譯又是編譯式的。

除此之外，Perl 自第 5 版之後，支援物件導向設計，具模組功能。Perl 的模組簡明易用，而且 Perl 的社群非常活躍，產量豐富，幾年下來，累聚非常龐大的模組程式庫，我們稱之為 CPAN (Comprehensive Perl Archive Network)[15]。

2.4.2 JSP

JSP(Java Server Pages)[16]是由 Sun Microsystem 公司於 1999/6 推出的新技術，是基於 Java Servlet 以及整個 java 體系的 Web 開發技術，JSP 的結構與 ASP 非常相似。不過 ASP 一般只應用於 Windows NT/2000 平台，而 JSP 則可以不加修改地在 85% 以上的 Web Server 上執行，其中包括了 NT 的系統，符合"write once ,run anywhere"的 java 標準。

JSP(Java Server Pages) 是一種結合了跨平台與跨網站伺服器支援，以 Java 為主的技術來產生動態網頁,我們用於管理介面與報表。

2.4.3 LDAP (Lightweight Directory Access Protocol)

目錄服務廣義來說是一種資料儲存的結構與擷取的協定。換句話說，目錄服務就像是一般電話簿或小型資料庫提供資訊，全球的使用者將可透過網際網路查詢放在網路上的資料內容，而企業內部相關人員也能透過企業內部網路查詢所需的資料，使用者經由標準介面與語法即可取得資訊，非常方便。

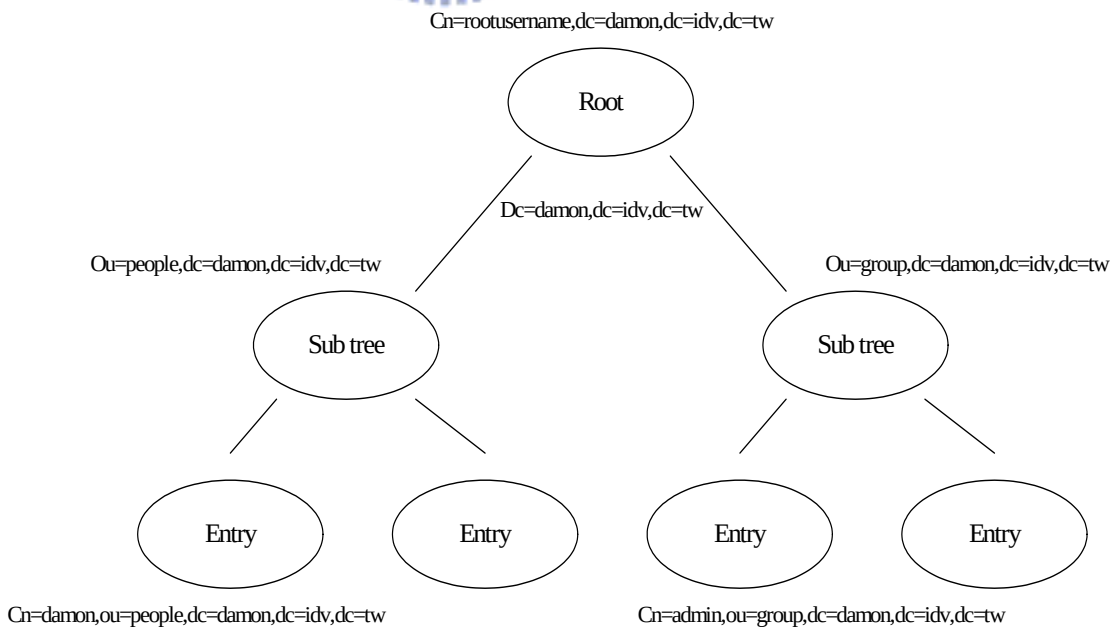
功能及特性

1.資料單一化：各應用系統不必自行維護員工的個人資料(尤其是帳號以及密碼)，利用 DS 即可進行身分認證以及資料查詢，降低資源重複投資。

2.快速與簡便的查詢：**LDAP** 提供比傳統資料庫更快速的查詢回應以及更簡便的使用介面。

3.分散式架構：**LDAP** 提供分散式系統的功能，可達到負載平衡(Load Balance)及高妥善率(High Availability)的服務品質。

4.多數應用軟體支援 **LDAP** 存取：**LDAP** 是一 Internet 上的標準協定，許多軟體皆已內建對 **LDAP** 伺服器的存取。因此，使用內建支援 **LDAP** 的軟體要介接 DS 既有的員工資料是相當容易且方便。



圖十一、　LDAP Tree 架構圖

2.5 資料庫

資料庫為一龐大且具有關聯的資料組成，且這些資料的特色為具有某種特殊結構以供多個系統讀取，資料庫的優點:減少資料重複(Reduced redundancy) 、整合資料(Integrated Data) 、完整性(Integrity) 。資料庫的缺點:昂貴、存取控制複雜。

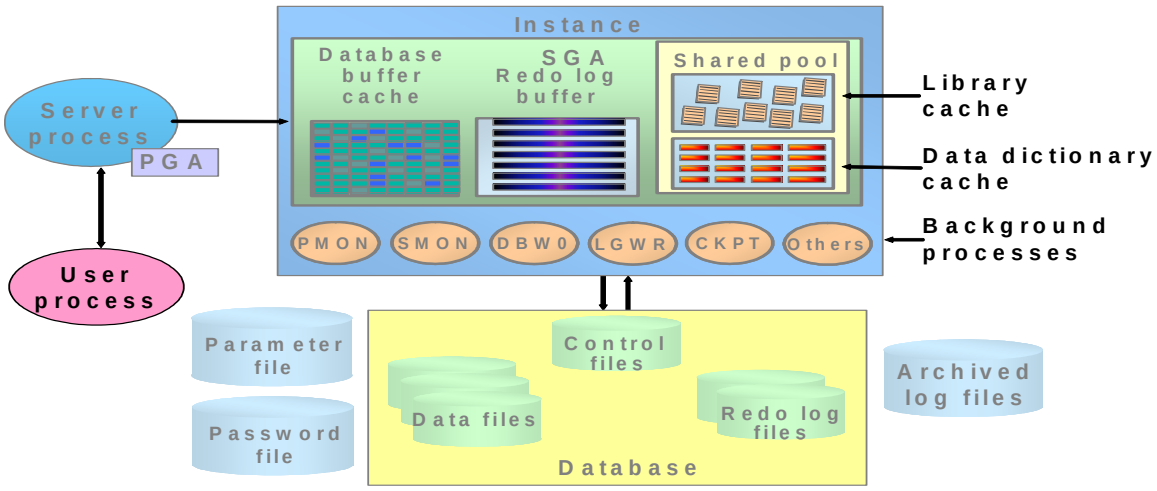
表一、 一般常用資料庫系統比較表

廠商	資料庫名稱	適用作業系統	功能	Cost
Microsoft	MS SQL Access	Windows	完整 少	中
Oracle	Oracle	Windows Unix Linux VMS	完整	昂貴
IBM	DB2	Unix	完整	昂貴
Open source	MySQL Postgres	Windows Linux Unix Linux	中 少	免費

2.5.1 Oracle 簡介

Oracle [18]是由美商甲骨文公司所設計的一個大型的關聯式資料庫系統，目前最新的版本為 Oracle 10g，在一般的大型企業中都可以看到它的蹤跡，幾乎所有的作業系統平台都可以使用 Unix/Linux/Windows/OpenVMS，Oracle 是針對 e 化的企業一個高度相容性、擴充性與高效能的資料庫，不管是應用於網際網路、系統開發、建置、成長、管理都具有不錯的好評，它

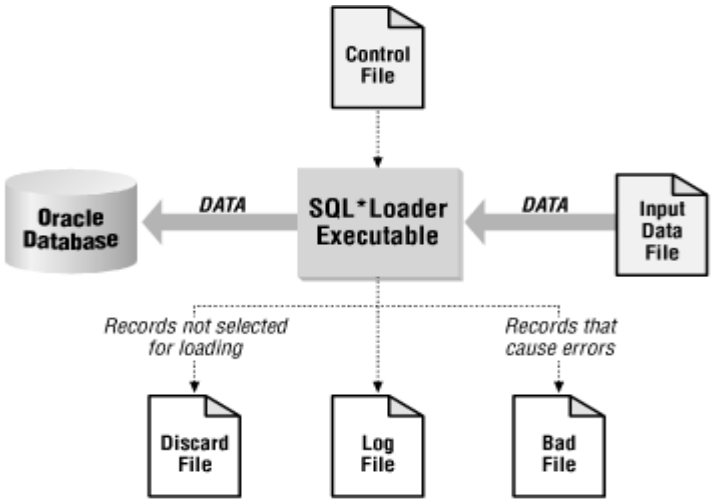
是一套可信度、可用性、安全性、易於建置的資料庫，有完整的管理應用程式，多項符合公定標準的技術容易學習：



圖十二、　Oracle 系統架構圖

2.5.2　SQL * Loader 簡介

SQL *Loader 是由 Oracle 所提供的一個公用程式，主要是用於處理大量的資料由文字檔轉入資料庫中，主要是由 Input Data File(輸入資料檔)、Control File(控制檔)與 Log File(記錄檔)所組成，如圖十三 所示，它是利用控制檔預先設定好的輸入資料與資料庫相對應之欄位，並可做適當的處理，例如:異常的資料不處理，可大幅減少資料輸入資料庫時錯誤的發生，並可藉由記錄檔中記錄，很輕易的得知有哪些資料再輸入時有異常(詳細資訊可參考附錄一)。



圖十三、　SQL *Loader 系統架構圖

第三章　　相關研究

隨著資訊科技與半導體技術的發達伺服器功能與日俱增，若不依靠相關的管理系統來維護各種不同的平台伺服器，不管是對企業維護成本或是對於管理人員來說，是一件勞心且費時的工作。所以許多的大型伺服器系統(Server Management System， SMS)被發展出來，在這些系統中，我們選擇了其中三種與本文研究較為相近的系統，本章節就針對三套管理系統做分析探討

第一篇為 OpenNMS，它是一套開放原始碼(Open Source)的軟體，其功能幾乎都能達到伺服器管理人員的需求，但是當要擴充額外的功能時，這個系統就不敷使用。

第二篇為 HP OpenView Operation(OVO) ，它是一套大型商業整合性平台的伺服器管理系統，表面上有解決上述的問題，但是一般說來在價碼上幾乎是天價，而在軟體系統上的操作也相當複雜及繁瑣，而且其報表功能上亦提供太多且複雜的資訊，使用者必須經過長時間的訓練才能上手。

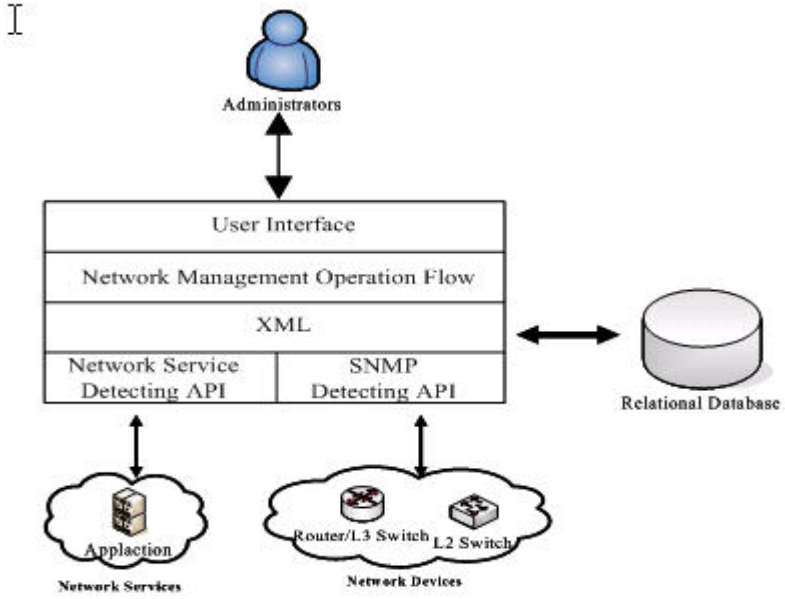
第三篇為以知識庫為基礎之智慧型伺服器自動監控系統 KISMS，此為學術研究上有人提出的解決方案，期望能夠利用本身的伺服器管理經驗，來發展伺服器管理系統。

3.1　OpenNMS

3.1.1　系統設計理念及架構

一群有經驗的系統工程師及網路顧問，為了改善投資於商業性的網路管理軟體所耗費的成本，與實現企業及整合性的網路管理系統的理想。在 1999 年便展開了這套 OpenNMS 的開發計畫。直到 2002 年的春天 OpenNMS 1.0 版終於孕育而生。

一般的網路管理系統多是以網路設備的管理角度作開發，而所利用的偵測技術多是以我們在所提到的 SNMP 來實現。可是當網路相關的服務大量出現時，我們就不得不導入額外的技術來做全面性的監控，而通常這類的監控軟體皆有其相當程度的複雜性。而 OpenNMS 是以管理者的角度出發，主要是提供一個更具親和力的管理架構，所以在使用上也就顯得容易許多。



圖十四、 OpenNMS 設計架構圖

其設計架構，如圖十四所示。包含了利用最基礎的網路通訊協定來偵測網路設備，也加強了針對網路各項服務的偵測模組。並利用 XML 文件在資料交換及程式開發上的便利性，來描述企業內部的管理區段。也透過資料庫的儲存及查詢的強大功能，建置一套大型的網路監測系統。

3.1.2 系統評估

OpenNMS 雖然是一套主要用於網路監測與管理的系統，但與一般的網管系統不同的地方是它突破了基本的監測功能而將網路與主機服務也納入監測項目之中，以單一的整合介面，提供不同種類的監測機制。另外，它也利用了 XML 的資料格式來描述企業內部的管理邏輯。

但是以主機管理系統而言，除了要能夠監測外，還必須提供適當的基本設備管理介面，讓管理人員也能夠利用此管理系統達到設備管理的目的，但由於其主要是透過 SNMP 作為監控與管理的工具，由於 SNMP 對不同作業系統平台的支援程度並不相同，所以使用起來一般專業用的系統管軟體還是有一段的差距。

在回報機制方面，這套系統目前也只提供網頁與電子郵件的回報機制。而若要達到全面性的錯誤回報還必須要有不同介面的回報系統。

3.2 HP OVO (HP OpenView Operation)

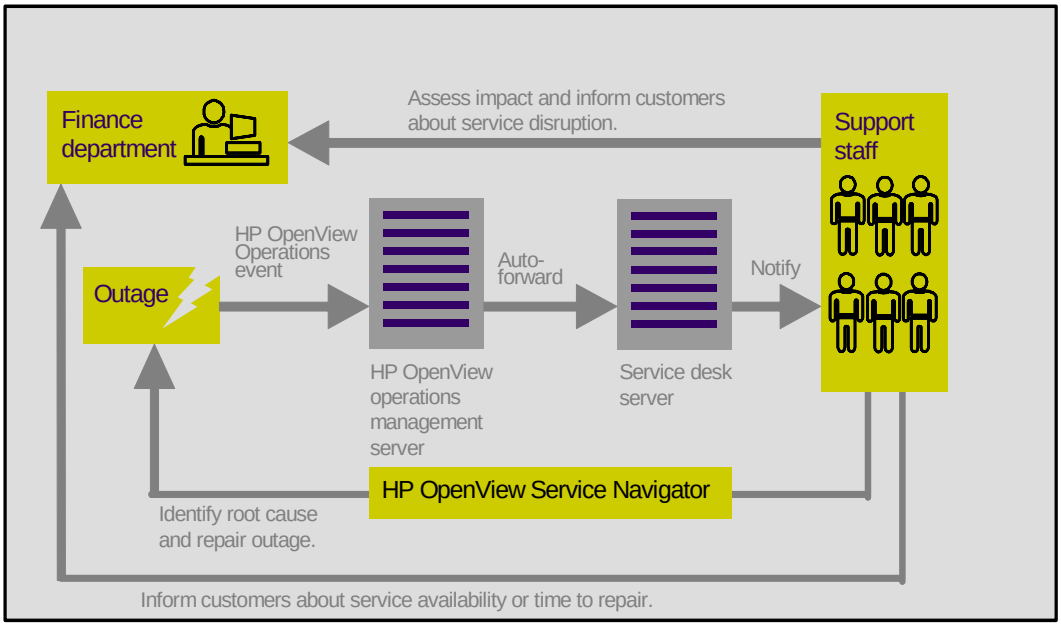
3.2.1 系統設計理念及架構

系統管理者需 24 小時全天候檢視登錄眾多的應用系統及管理監控介面，如檢查主要系統有無事件發生，包括系統運作是否有異常？備份是否完整、完全？回應使用者的需求與問題。系統維運管理人員對於各應用系統之管理監控，不僅複雜費時，也造成學習上的困難。而過去需要多人分別看管不同系統的運作狀況，運用 HP OpenView Operation 的維運管理方案後，系統管理者只需利用單一的管理介面，就可以直接進入每個不同的應用系統，監看運作狀況，約 10 分鐘就能夠完成所有第一步的檢視作業，「透過標準化、自動化的主動管理機制，可以避免人為錯誤。除降低成本外亦可大幅提昇系統效益。」

在硬體系統部份，同樣的由單一管理介面就可以即時了解到資源利用情形；系統的障礙管理及排除也可由單一窗口進行。在監控功能方面，「以不中斷服務目標為例，我們可以監控網路流量、 CPU 速度、記憶體、儲存容量，進行必要的擴充，更重要的是能夠在問題發生前即有所行動。」

除了便利的管理功能外， HP OpenView Operation 更為系統管理者所喜好的，還在於它可以依據監控管理狀況自動產生報表。這些圖形化的分析資料，是提供主管報告非常重要的資訊。也可依使用者要求的分析內容包括 CPU 、記憶體使用狀況，找出造成系統負載的原因、保存歷史資料，尤其後者更是作為未來規劃系統需求時非常重要的依據。

HP OVO 是由 HP OpenView 延伸而來的一種商業軟體，它可即時為您提供全面的事件管理，主動效能監控，以及針對 Windows 和 UNIX 系統、中間件和應用的自動提出警示、報告與繪圖。



圖十五、 HP OVO 架構圖

3.2.2 系統評估

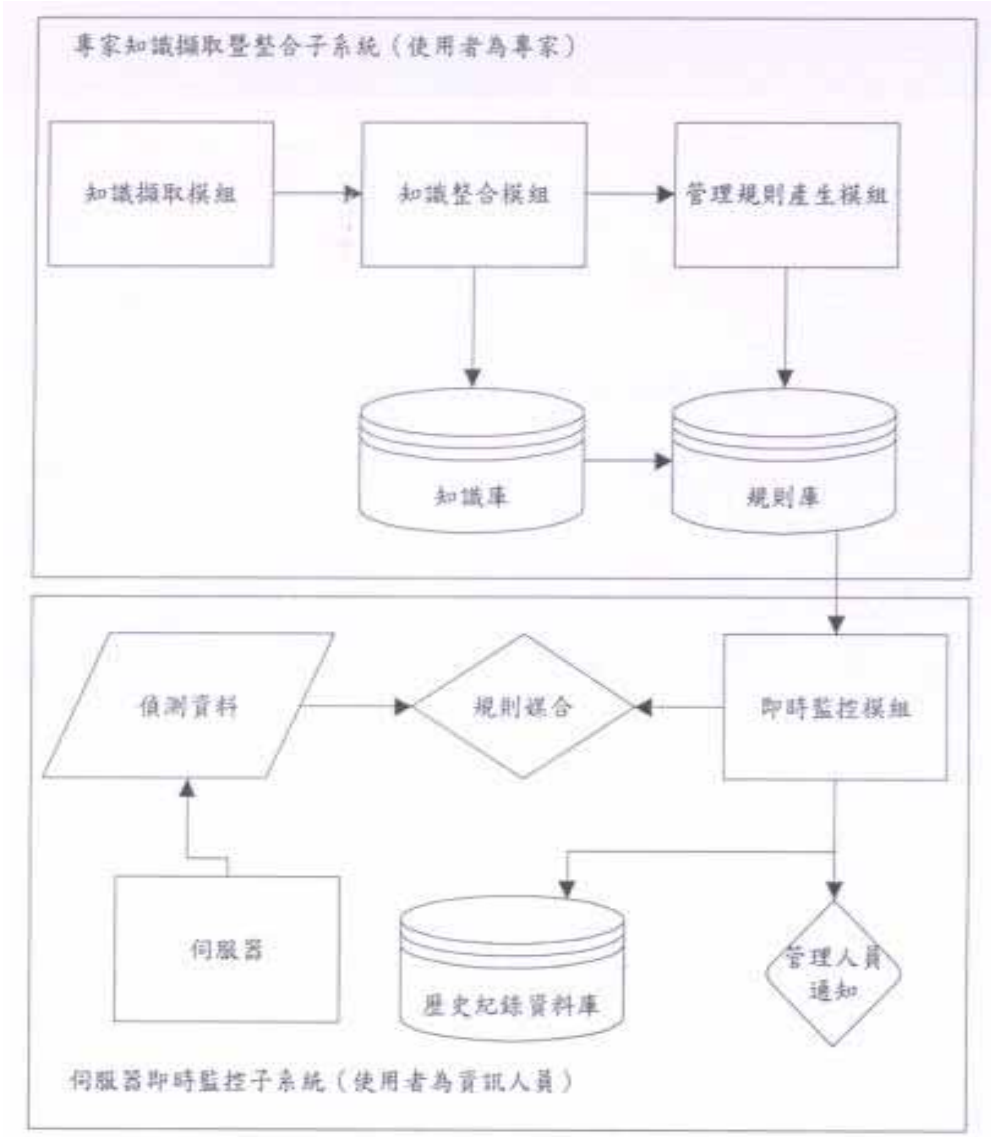
OpenView Operation 是一套很完整的系統管理軟體，其所提供的資訊也非常充分，並可與其另外一套重量級軟體 HP OpenView 互相搭配並作緊密的結合，可達到企業全方面管理(主機、網路、服務)所需。

其缺點為建構成本昂貴，操作相當複雜須有專門人員負責維護，一般的使用者需經過專門的訓練才能操作，對於一般中小型企業而言是一項非常沉重的負擔。

3.3 以知識庫為基礎之智慧型伺服器自動監控系統 (KISMS)

3.3.1 系統設計理念及架構

此篇研究論文是由中華大學資工所李中銘所提出，他以如何快速的擷取及整合伺服器管理員的管理經驗及知識，研發一套”以知識庫為基礎之智慧型伺服器自動監控系統” [19]，此系統可以擷取及整合多專家的管理知識產生管理的規則並且能夠依照管理規則及時監控伺服器狀態、自動處理事件和及時通知網站管理者的系統。



圖十六、 KISMS 系統架構圖

3.3.2 系統評估

KISMS 是一套採用人工智慧與知識庫之系統管理軟體，它提供的一般系統管理軟體比較少使用的知識庫的概念，可藉由資料的收集與累積，當有異常現象發生時可利用人工智慧的方式進行資料的比對，提出警訊給管理者，使得管理者能及早處理，達到事前預防的效果。

KISMS 目前只能適用於微軟視窗作業系統平台(Microsoft Windows)，對於一般大型企業常用的 Unix 或 Linux 作業系統並不支援，對於作業系統的支援程度並不是很理想，且它所使用的管理介面是作者利用 VB 設計好的管理介面，並不是採用目前最普遍的 WEB Based，若需使用需另外安裝管理介面，使用起來並不是那麼的方便，另外發送警訊部份目前只支援電子郵件傳送，對於警訊傳送的即時性方面較為薄弱。

第四章　跨平台主機管理系統(CPSMS)

企業內有各種不同的主機與作業系統，不同的作業系統操作的方式並不相同，甚至有些相同的作業系統但是因為版本的差異也會有所不同，這對一個系統管理員而言，實在是一個很大的挑戰，常會因為對於系統的熟悉度不同影響到問題解決的速度，若處理速度過慢會使得原本只是一個小問題卻演變成大災難，這些不確定的因素不斷的挑戰著管理人員的應變與處理能力。

企業內的主機數量隨著硬體的價格不斷的下降與 Windows/Linux 的崛起造成主機數量不斷的上升，每個管理員所要負責與管理的機器越來越多，如果沒有一個自動監控與管理的機制協助實在很難應付。另外，確保服務主機的運作正常也是管理人員日常的重點工作之一，但是單靠人力的監控還是會有疏忽的時候，若缺乏自動通知的機制會使得管理人員錯過最適當的處理時間。

機器多了以後又衍生出另外一個問題就是機器要如何調配，才能將其效能充分發揮，且如果無法對機器設備的狀況有充分的了解，與其效能表現，那麼要使其發揮最大的功效，簡直就是天方夜譚，所以我們也需要效能統計的功能

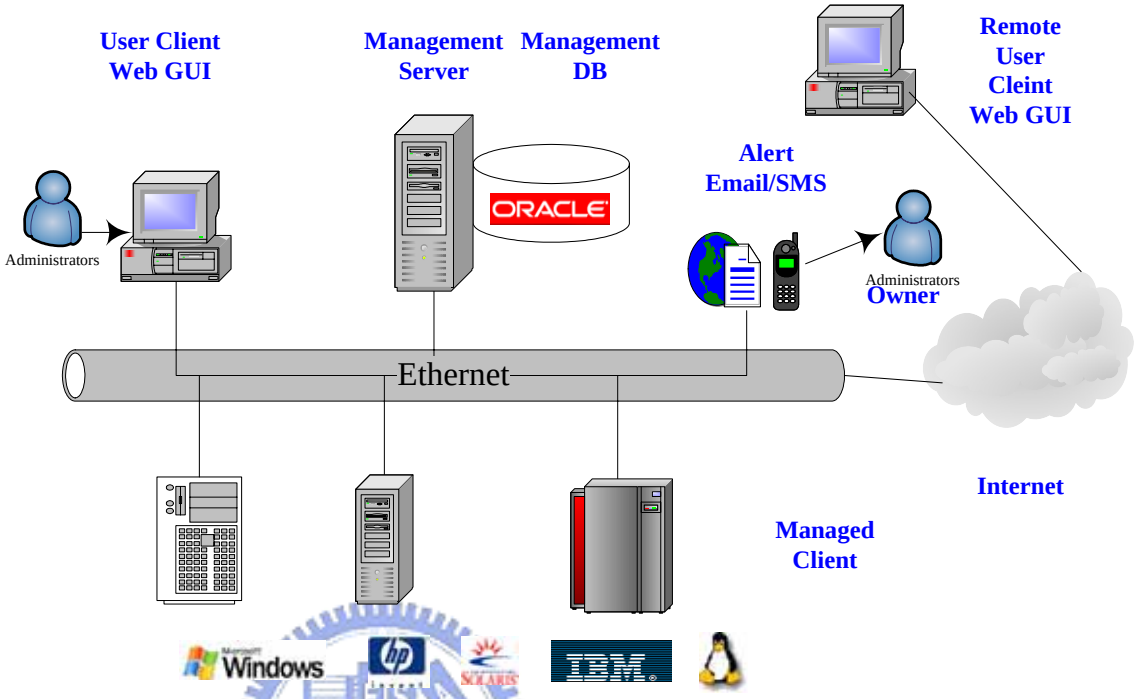
如果能做到事前的預防，那麼就能減少主機異常帶來的衝擊，當然這需要一些資料的統計與專家系統的協助。

4.1　CPSMS 系統簡介

本系統最重要的環節就是如何將異質平台的主機設備加以整合，並將所收集來的資料儲存於資料庫中，以利未來資料的交換及系統之方便擴充。在即時回報機制方面同樣的必需採用不同的系統來做到錯誤回報的完整性。

基於以上的問題我們建置了一套跨平台的主機管理系統(CPSMS)，如圖十七所示。希望藉由有效率的資料收集方法來擷取所有的主機設備所提供的相關資料，以供未來主機服務品質改善的依據，並提供簡單的報表查詢機制供管理人員使用、利用簡單的控管介面來管理所有的主機設備方便管理人員、以即時快速的錯誤通報機制讓管理人員於第一時間解決問題。最後藉由上述的各項特點

能提高管理人員的工作效率並讓使用者達到最高的滿意度，並能充分發揮主機的資源。



圖十七、 CPSMS 示意圖

4.2 CPSMS 系統架構

將日常管理的工作自動化，除了能夠節省大量的管理人力，也能夠將管理資訊長期的保存作為日後的管理依據，而這樣的管理系統才能夠具有完善的管理能力，以及具有高度的可移植性。

我們從功能面來設計一套整合性的主機管理系統。這套系統必須要能整合不同作業系統與硬體平台所提供的主機管理資訊，並經由網路的傳送與預先設計好的程式將所收集的資訊存放於資料庫中，並且利用系統內部預先設計的自動化的管理程式，協助檢查系統是否有異常發生，提供管理人員與一般的使用者利用簡單且單一的管理介面，取得所需的資訊與報表，並且能在有異常發生時收到警訊，達成智慧型管理系統的目標。

本系統將主機管理系統的功能分為下列五點。

1. 整合式管理介面

發展主機管理系統的首要就是要有整合式管理介面與管理人員作互動，管理人員可以從管理介面去增加所要管理的主機設備或相關監控項目與警訊，並能夠從介面觀看管理系統所收集的相關資訊，所以系統介面必須能夠以最簡單且完整的畫面呈現給管理人員使用。

2. 資料收集

如何能讓主機管理人員知道目前主機的狀態，就必須要靠主機管理系統能夠即時與定期的自動將主機的相關資訊收集回來。一個整合性的主機管理系統必須要有能力去收集不同作業系統與硬體平台的主機所產生的資訊，並可由管理者設定收集項目，這樣才能夠提供管理人員取得所需要的相關資訊。

3. 報表產生

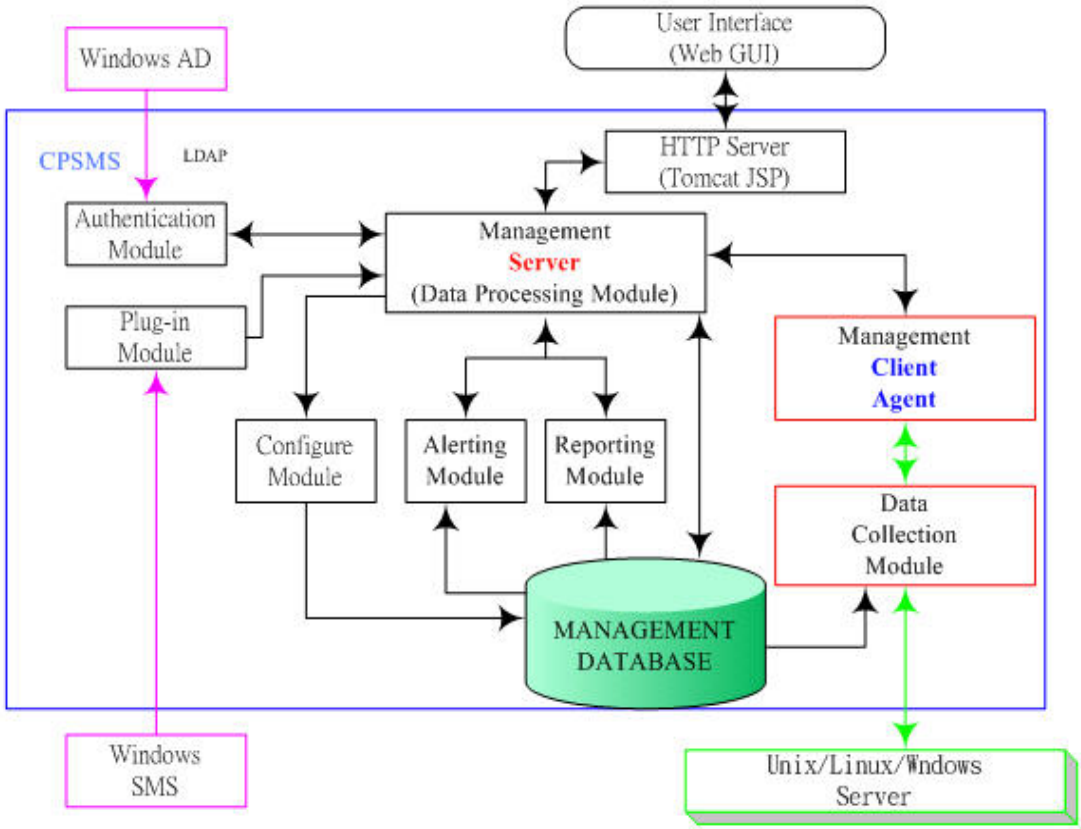
當資料收集回來後，主機管理系統必須要能夠依照管理人員透過系統介面所傳達的要求，將資料做最簡單且完整的呈現，以利於管理人員作最快速的判斷。而現今的報表除了要達成上述的目標之外，還要能將這些報表資料，透過標準資料格式方式與不同性質的系統做資料的分享，例如：專家系統等，除了提高資訊再利用的能力，也可增加主機管理系統的可用度。

4. 組態設定

完整的主機管理系統必須要有機制讓相關的管理人員能透過系統介面去直接設定監控項目組態，這樣除了可以減少許多組態設定的時間，也可依據不同主機與應用程式(AP)的特性設定符合需求的監控組態。

5. 異常回報

當主機發生異常狀況時，管理系統必須能夠以最即時與適當的機制通知相關的管理人員前來處理，才能及時的將異常狀況排除，減少異常所造成的影響。



圖十八、 CPSMS 架構圖

從上述的功能簡述，本系統之整體性架構如圖十八所示。本系統將以各功能所要達成的目標，來區分不同的程式模組。並在下列各節中介紹。

4.2.1 管理介面程式(Management Scripts)

主機管理系統之操作介面必須要能夠讓管理人員在任何時間與地點都能夠不需要另外安裝軟體就可以直接使用，這樣才可以發揮管理系統的好用性。所以本系統將採用在節所提到的 Web-Based 的管理介面，我們將利用一組 JSP 伺服器端程式與管理者做互動，來達到下列的管理功能：

(1)新增與修改資料

Client 端會自動將主機的軟硬體與效能相關資訊自動回傳給 management server，經過處理後寫入資料庫，有些資訊由於目前的系統上無法自動收集，需利用人工以手動的方式輸入。我們也提供輸入資料的界面，管理者可藉由手動的方式新增或修改被管理端的資料。

(2)產生報表

主機所產生之資料對管理者而言是非常寶貴的，但散落在各地的資料使得管理人員無法快速的取得，經年累月下來的資料無法發揮它的功用，若有一機制將這些資料做適當的分類並以最簡單的畫面呈現給管理者，即能將雜亂的資料轉換成有用的資訊，所以本系統將透過 Web-Based 介面來達到上述的功能，讓管理者能方便的查詢由報表產生模組(Report Generation Module)所產生的 HTML 文件。

(3)組態設定

我們可以透過網頁將所需設定的監控項目與 client 端程式的相關參數值，利用 4.2.5 所提到的組態管理模組將相關的設定傳送給 client，省去人工手動一台一台的設定，如此一來不但能方便的設定亦可降低管理人員訓練之成本，減少出錯的機率。

4.2.2 資料收集模組(Data Collection Module)

不同異質性作業系統與硬體平台，如何有效的管理這些系統進而提昇主機服務之品質與充份發揮主機的效能是管理人員首要的目標。為了實現這樣的願景我們建構了一個模組，如圖十九所示，以自動化的方式定期的收集各主機所提供之管理相關資料，以及定時監測主機之狀態，並將之儲存至資料庫中，管理人員可透過報表產生模組(Report Generation Module)查詢已收集到的資料，利用這些資訊協助判斷主機的服務品質，與藉由資料的收集預測主機的異常的發生。

需收集資料的主機藉由 client agent 程式，取回相關之主機設備與管理所需的資訊，將資料儲存與資料庫中供後續管理使用。

我們將利用不同的程式來收集這些異質型態之主機設備資訊及主機效能所提供的資料，分述如下：

1.主機設備資料收集

不同的主機由不同的軟硬體架構所組成，對管理者而言不同的軟硬體架構並無法直接替換使用，當系統有問題時我們需要找到相同軟硬體架構的主機方可使用，有時候

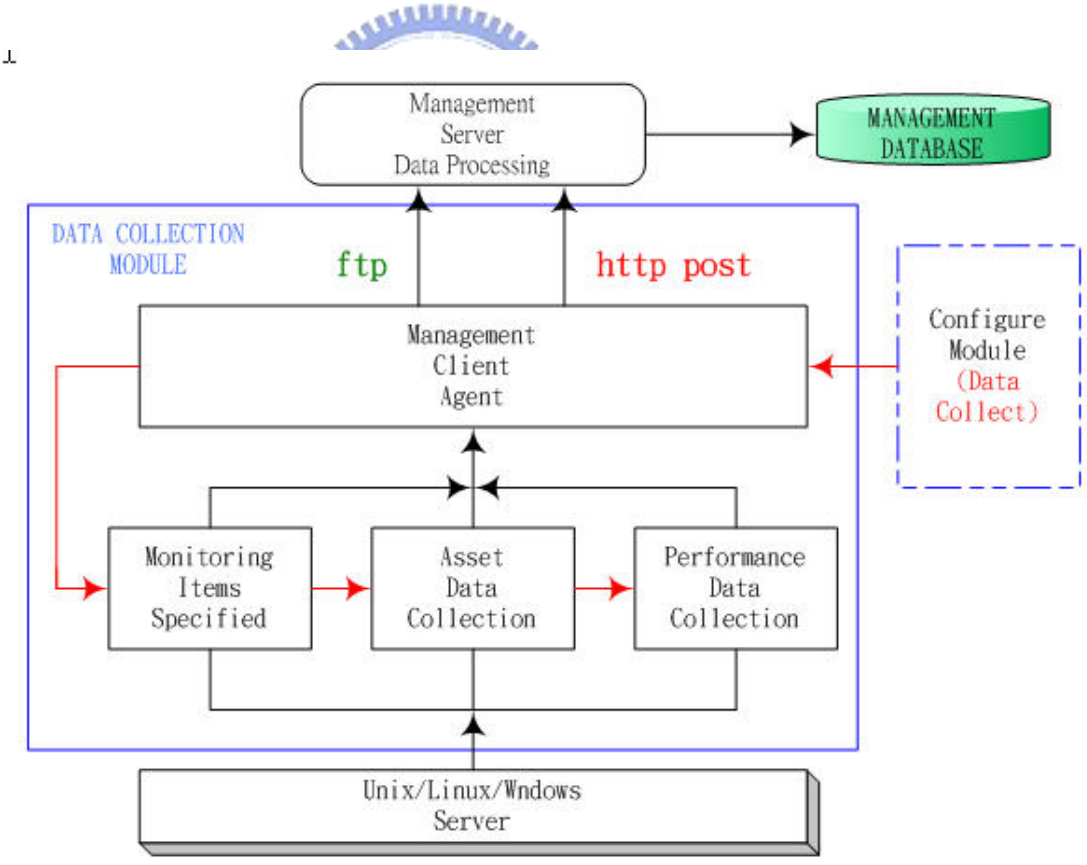
我們也需要對所負責的系統做分類，所以這些資料必需列入收集項目。

2.主機效能資料收集

主機效能資料可讓管理者判斷主機之使用是否有不尋常之情形，或是主機是否有效能上的瓶頸發生，所以也需列入我們要收集的項目之一。因此我們利用在 2.2 節提到的工具收集相關的效能資料。

3.主機監控資料收集

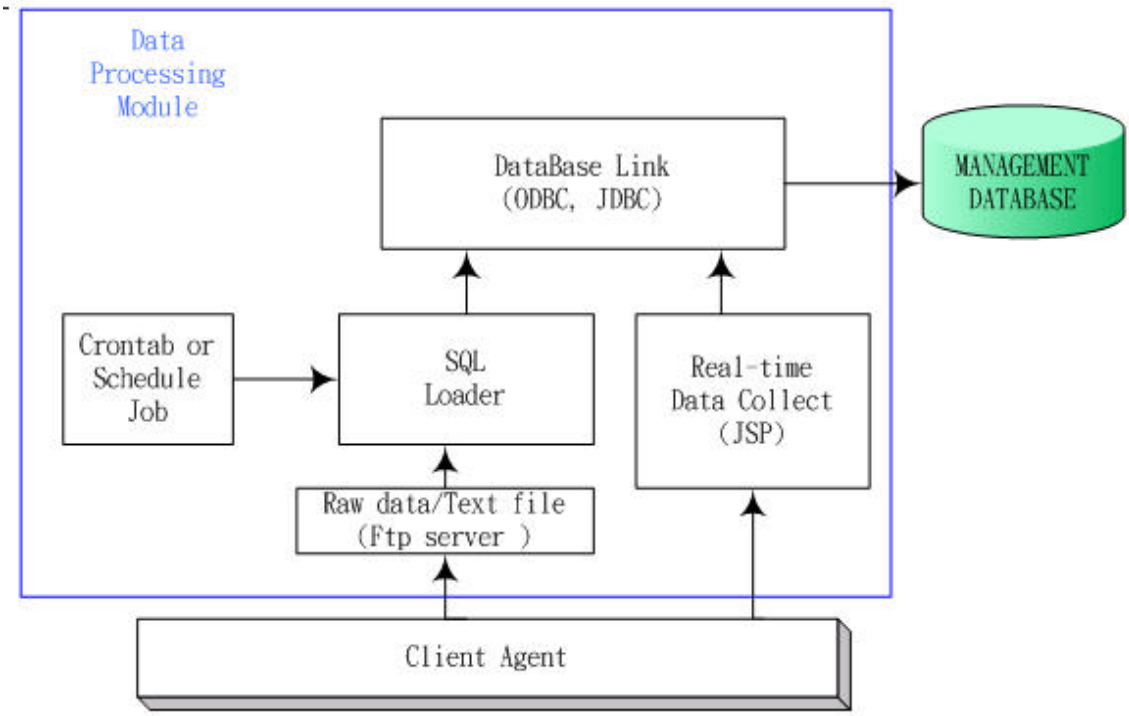
主機目前的運作狀態，必須要定時監測，這樣才能確保其服務正常運作。像是企業內部的生產資訊系統主機，必須要有 24 小時皆不當機的能力，這樣才能確保生產不會受到影響。所以說長期監測伺服器主機的運作狀態也是本模組應收集的項目之一。



圖十九、　資料收集模組架構圖

4.2.3 資料處理模組(Data Processing Module)

因為不同主機系統所收集到的資訊與格式並不相同，我們設定將所收集到的資料採用純文字格式，以利於資料交換與處理，我們須將文字格式整理成符合資料庫格式，由於每天所收集到的資料量還算不少，我們用到 2.5.2 所提到的 SQL Loader 利用排程將經過處理後的資料寫入資料庫，另外有一部分即時更新的資訊我們採用 JSP 直接將資料透過網頁的方式直接寫入資料庫中，已達到資料即時更新的效果。

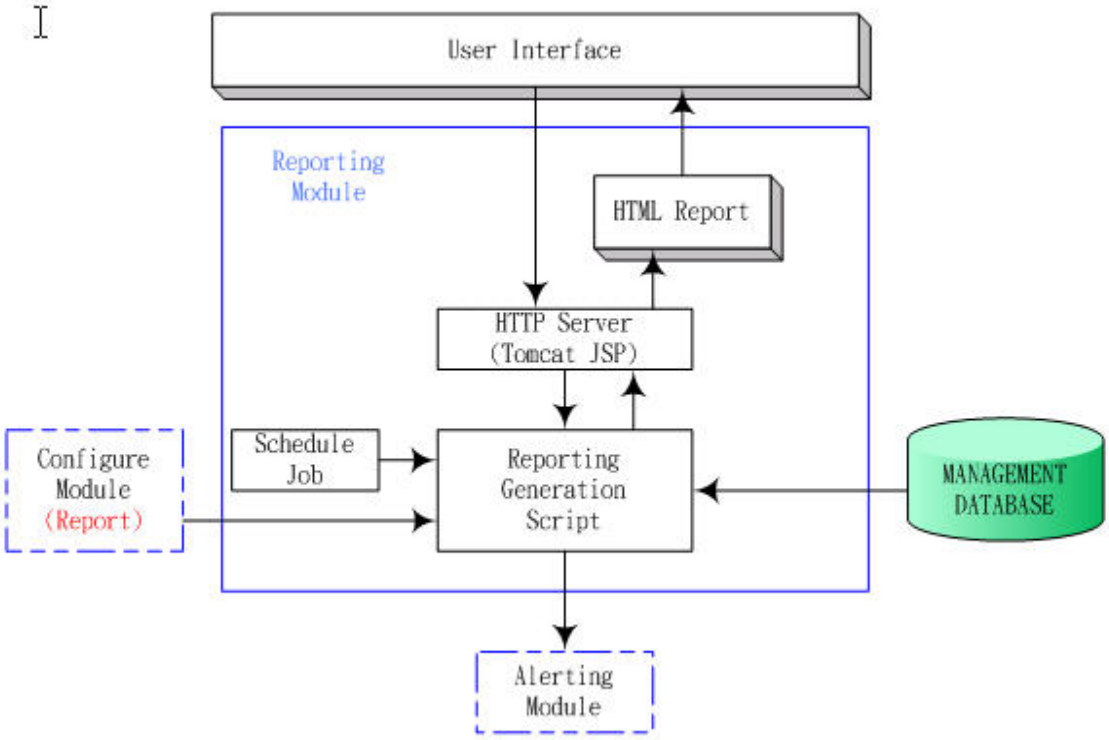


圖二十、　資料處理模組架構圖

4.2.4 報表產生模組(Report Generation Module)

各種不同的主機管理系統所產生之報表皆有其特殊格式，有些極其複雜的造成管理人員閱讀上的不易，也有些簡單無參考價值，以致無法滿足管理人員之需求，因此我們將一般管理者最需要且最有用的資訊定義成一固定之報表格式，透過本模組的報表程式(Reporting Script)將資料收集模組(Data Collection Module)所收集的異質型態資料，依照管理者所設定的時間點產

生相對應的 HTML 報表文件，管理者即可透過單一整合介面方便且快速的取得其所要的資訊，且可以最用簡單易懂之方式呈現給管理者，如圖二十一所示。也可將之與不同性質的系統做資料的交換，例如：專家系統，以增加資訊的再利用價值。

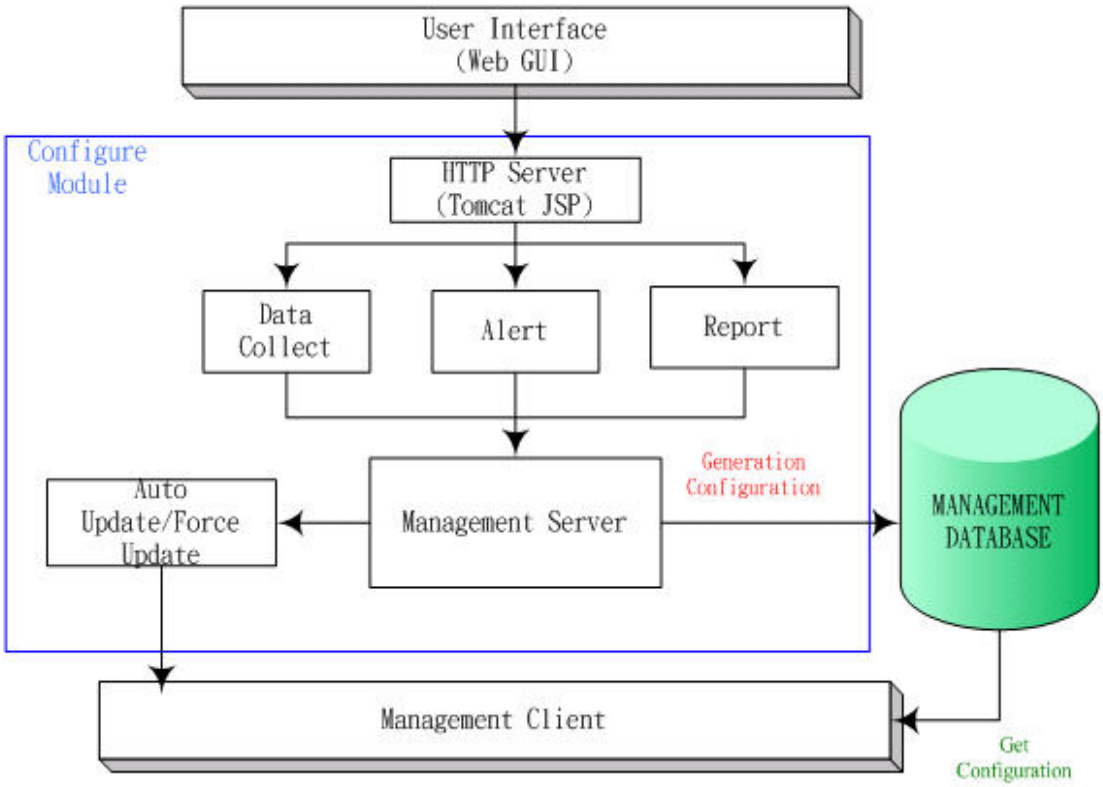


圖二十一、報表產生模組架構圖

在定時產生報表時本模組亦會做異常資料之判斷，一旦發現資料有異狀時，例如：所收集到的資料反應出某一台主機異常停機，即會觸發警訊模組(Alerting Module)將此錯誤訊息依據警訊發送方式設定，即時的透過手機簡訊及電子郵件的方式發送給相關之負責人員。

4.2.5 組態管理模組(Configuration Module)

當管理者要針對主機執行組態設定與變更時將透過本模組將主機變更後之組態值儲存於資料庫中，被管理端將定期如表二、所示之間隔時間，比對資料庫中該主機的組態設定值 如圖二十二、所示，當資料庫中設定值有變動時即可更新其本身組態設定值。此自動化之機制可省卻管理人員手動更新被管理端組態動作，且可隨時檢查被管理端的組態設定。



圖二十二、 組態管理模組架構圖

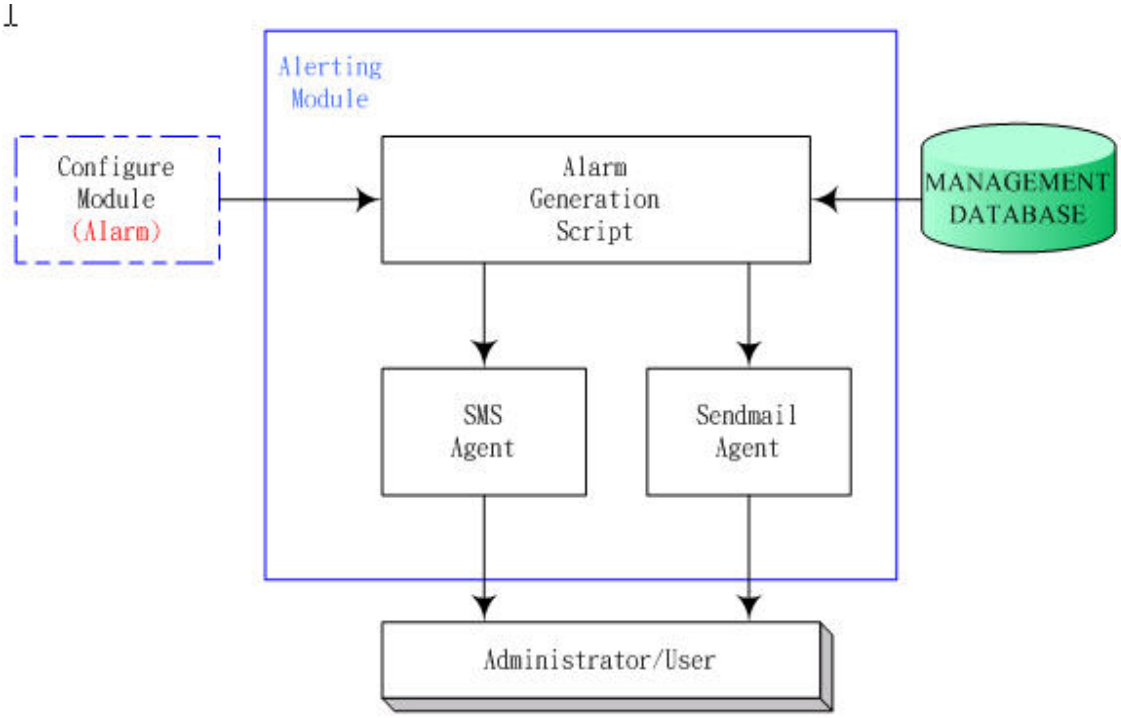
表二、CPSMS 組態預設值表

Item	interval	status
Heartbeat	180	active
Task update	1800	active
Config update	3600	active
Agent update	3600	active
Self check	1800	active
Agent Info	3600	active
System Time Check	3600	active
System Information	86400	active

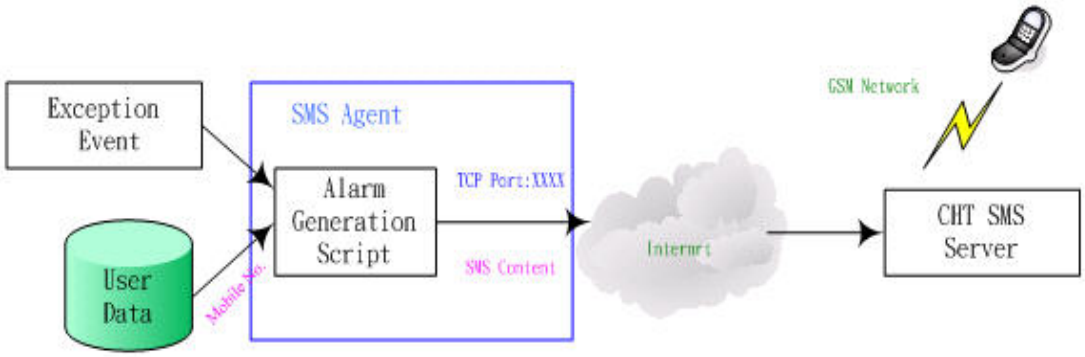
4.2.6 警訊模組(Alerting Module)

在沒有行動電話及呼叫器的年代裡，當主機或服務發生異常時，無法自動的發出警訊通知相關人員，而必需藉由人力一週七天一天 24 小時不斷的盯著螢幕監看，但人畢竟不是機器，無法長時間的進行如此單調枯燥的工作，以致於常常會錯過最關鍵的時間甚

至於會釀成嚴重的錯誤。隨著科技的發達及行動電話的普及，行動式的溝通已成了最主要的連絡方式，因此我們將這樣的便利性應用至主機及服務之監控上，當發生異常時將重要的資訊透過 GSM 網路系統即時的傳送簡訊(SMS)給管理人員進行處理，如圖二十三、圖二十四、所示為了確保管理系統的完整性及時效性，我們也利用電子郵件系統來發送訊息，雙管齊下以確保訊息能正確無誤的送達管理人員。另外本模組也提供儲存歷史資料之功能，將已發送之訊息記錄下來供未來之分析。



圖二十三、　警訊模組架構圖



圖二十四、　SMS 警訊發送示意圖

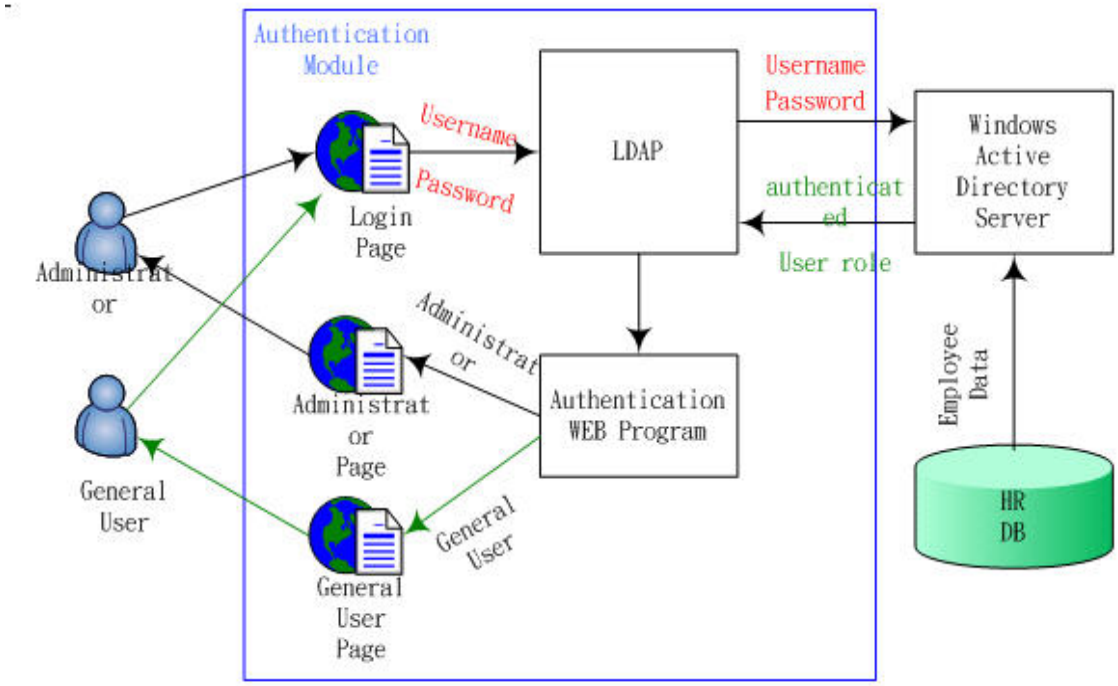
4.2.7　外掛模組(Plug-ins Module)

由於本系統可以利用資料庫與其它的系管軟體連結例如微軟的 SMS，也可以與企業內的資產資料庫做連結提供管理者更詳盡的主機資訊，由於不同的系統資料庫的 schema 並不相同所以互相連結或交換資料時需要經過轉換，本模組可將不同 schema 的資料庫轉換。

4.2.8　認證模組(Authentication Module)

由於本系統可以提供很多主機資訊給不同的管理者，例如系統管理員(SA)、資料庫管理員(DBA) 、應用程式負責人(AP owner) ，所以我們並須有認證的機制用於確認使用者的身分，一般企業使用 WINDOWS AD 已經非常的普遍，大部分的 AD 皆與企業內部的組織與架構皆有互相結合，而且會即時更新。

我們利用 LDAP 與企業內的 WINDOWS AD 做認證不但可以控制使用者登入本系統權限，且可利用 LDAP 取回相關使用者的資訊，可用於限制使用者能使用的功能，當使用者角色有所變動時(例如離職調職或有新進同仁時) ，並不需要另行設定或刪除帳號，可大幅減輕管理者的負擔。



圖二十五、　認證模組架構圖

第五章　　系統實作、評估與比較

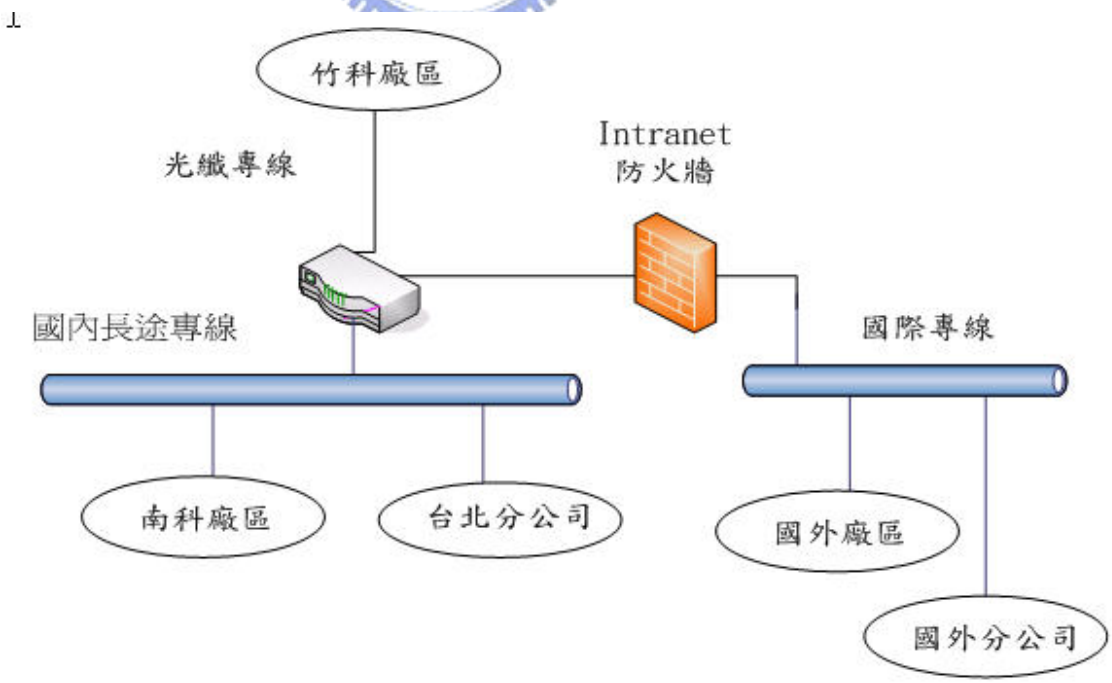
系統實作即是將所有的想法付諸行動，但在行動前必須有完善的規劃，因此本章節將先針對整體的環境及開發平台作介紹，接著對所選用的程式開發工具及模組(Module)作說明，並以實作出來的畫面展示本系統之成果，之後再進一步評估其實用價值，最後與第三章所提到的三套管理系統做比較。

5.1　環境及開發平台

1.網路環境

我目前任職於新竹科學園區某電子公司，負責主機維護與管理等工作，所以本系統所使用的環境是以公司內的主機作為管理對象。

如圖二十六、所示，全公司網路由一台大型的網路交換機作為核心骨幹，竹科各廠區利用光纖，南科廠區、台北分公司利用國內長途專線、國外廠區與分公司利用國際長途專線經過防火牆連接至核心骨幹設備。



圖二十六、　CPSMS 網路環境架構圖

2. 作業系統

由於本系統強調的是跨平台，所以可以將系統建構於不同平台的作業系統上，但考量到系統開發的便利性。筆者是使用 Windows 2000 Server 作為開發平台，另外也成功的在 IBM AIX 主機上面運作。

3. Web 伺服器

本系統主要是使用 JSP 程式做為動態網頁程式開發，為了考量可跨平台使用，我們選擇市面上使用率最高並且也是最穩定的免費動態網頁伺服器：Tomcat JSP 伺服器，並且透過 LDAP 與企業內部的 Windows AD 作認證，使得 WEB 伺服器具備更高的安全性。

4. 資料庫

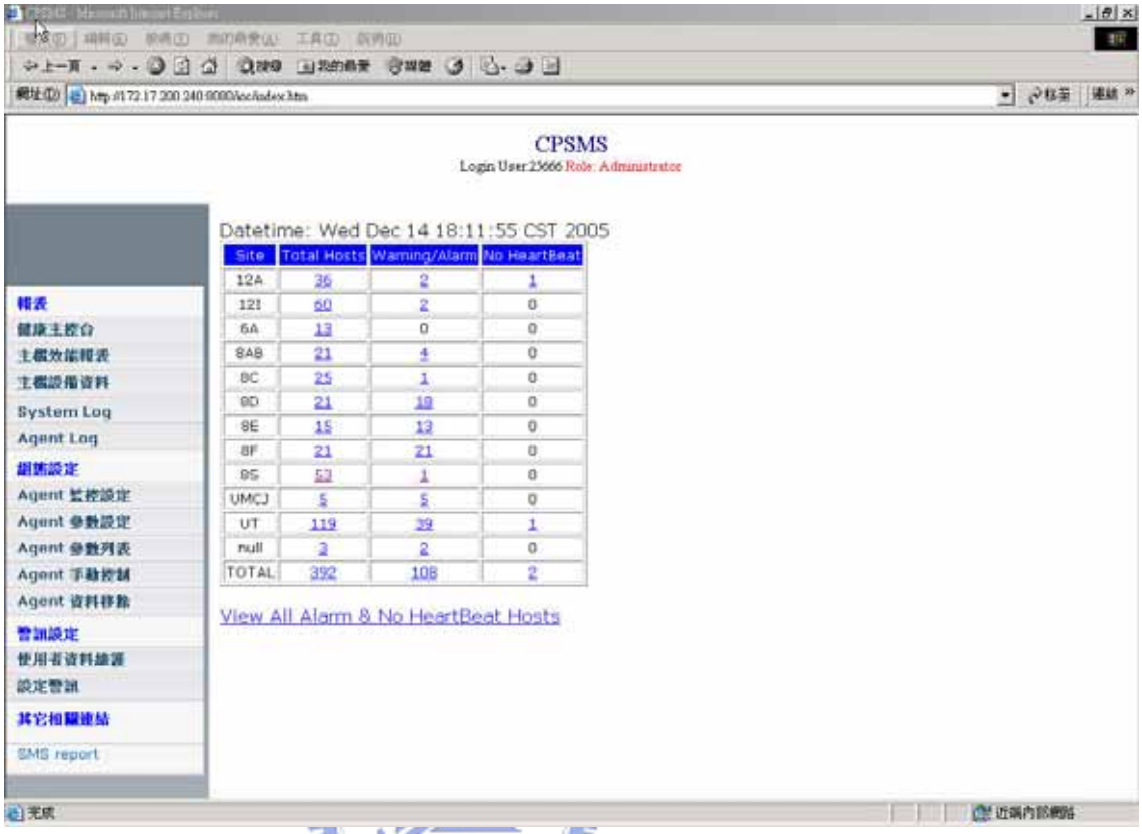
本系統會依照管理者的設定，與系統預設的收集項目，將所收集到資料儲存於資料庫中，所以資料量會非常龐大。由於企業內部主要使用 Oracle Database ，不論是效能或穩定性都是符合需求的，再加上大部分的程式語言皆有連結 Oracle 資料庫的程式介面供開發人員使用，集合以上的特點所以我們選用 Oracle 作為資料儲存之平台。

5.2 程式開發

本系統是一套整合性的跨平台主機管理系統，所以在程式開發上面會需要用到相當多的輔助工具及程式庫來協助以加速我們的系統開發的速度。為了整體開發成本之考量，筆者決定採用開放原始碼(open source)所提供之程式庫或工具，並將其適當的整合運用至本系統中。利用開放原始碼來開發最大的問題就是在於如何尋找適當且穩定的軟體或程式庫，有時需稍加修改，有時只需利用其所提供之程式介面。

由於主機管理者需要收集不同的資訊來提昇主機運行的順暢度，所以主機管理系統首要的目標就是儘可能的利用標準及有效率的方法來收集主機資料，並利用目前大家最熟悉的介面來當作整個管理平台，這樣才能增進管理人員使用上的便利性。因此我們採用 Web-Based 作為系統的介面方便管理者使用，圖二十七為管理者登入本系統後之主畫面，上面部分為目前所登入的使用者與其角色，

畫面左邊的部份為本系統所提供功能選項，右邊部分則為資料顯示區(圖二十七 顯示的範例為健康主控台)。



圖二十七、 CPSMS 主畫面

目前本系統已完成設計及初步的系統開發，我們採用 JSP 以及 Perl 作為主要的程式開發語言，主要是因為這兩種語言皆可跨平台使用，且都是免費的。

JSP 語言是以 Java 技術為主，不但可以產生動態網頁，更可以利用一些現有的 Java 模組增強其功能，在加上 JSP 可以適用於大部分常用的作業系統平台，有很高的移植性，與各種資料庫也有提供與 JSP 連結的模組。

Perl 程式語言本身就是主機管理人員常用之開發工具，因為它同樣提供了許多的模組供我們使用，且其擅長處理文字型態的資料格式，這些都使的開發系統顯得容易許多，當然的成了首選的程式語言。

下面我們就針對各模組所需之程式逐一做介紹。

5.2.1 資料收集模組之程式

因為我們所要收集的主機資料來自於不同作業系統的主機，所以必須利用跨平台可使用的程式語言，方可適用於所有的主機， 2.3.1 所提到的 Perl 程式語言就符合這項要求。整個模組的程式皆是以 Perl 程式語言來開發，並透過其本身提供的模組與函式來達到我們的需求，雖然在程式開發上筆者花了很多時間做程式的寫作並除錯，但是筆者曾經有過開發類似程式的經驗，所以整體來說還算順利。

擷取主機設備硬體資料之方法:

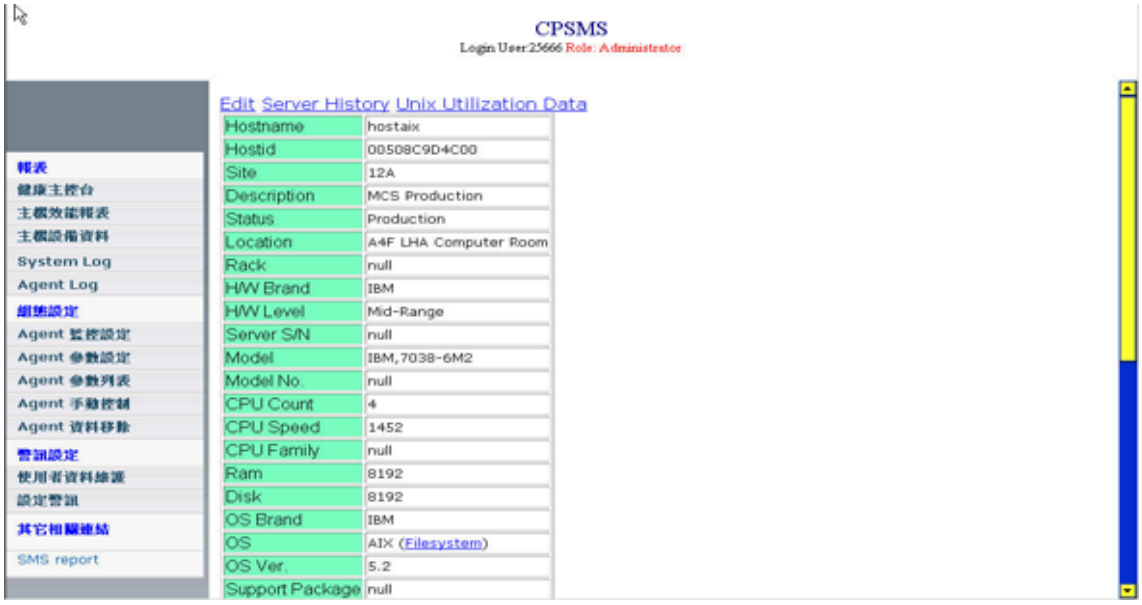
1.Unix/Linux 作業系統平台:

利用 Unix /Linux 內建指令(例如 OSF 5.1B 之 hwmgr 指令)取得主機硬體相關資訊，可參考附錄六。

2.Windows 作業系統平台:

利用 perl 程式取得 2.3.4 節所提到的 WMI 取得 Windows 主機硬體相關資訊，可參考附錄七。

筆者是利用透過 Perl 所提供的 WWW 模組，即時將所收集的資料利用 HTTP Post 方式如附錄二，上傳至 Management server 主機的 web server 上，我們利用事先設計好 JSP 程式，將資料即時寫入資料庫中，這些資料可讓管理人員馬上獲得即時性的資訊並可提供警訊模組使用。

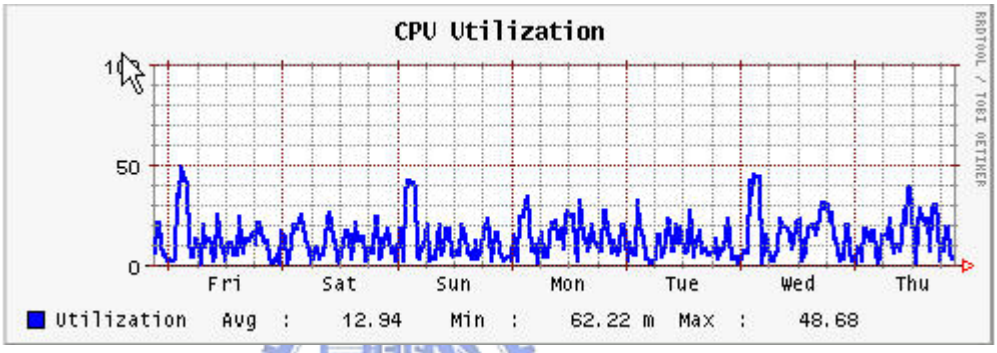


圖二十八、 CPSMS 主機設備偵測結果畫面

收集主機效能的資料之方法:

1. UNIX/Linux 作業系統:我們使用 2.2 節所提到的監控工具收集 Unix/Linux 主機效能，由於資料較無即時性更新要求，所以我們採用每天透過 Perl 所提供的 FTP 模組如附錄三，將所收集到的資料上傳到管理主機上，在透過資料處理模組將上傳的資料處理後寫入資料庫中。

2. Windows 作業系統:我們採用與 3.1 節所提到 OpenNMS 使用相同收集效能資訊的方法，利用 2.4 節所提到的 SNMP 收集 Windows 主機效能資訊。



圖二十九、 SNMP 效能資料(CPU)

收集主機記錄檔的方法:我們可利用預先設定好可接受的警告或固定會寫入紀錄檔的資訊的項目，我們在對每日產生的記錄檔資料做比對，若有不在名單內的紀錄產生及判定為有異常產生，也就是一般所謂的白名單(如附錄四)方法，我們也是利用 Perl 所提供的 FTP 模組將所收集有異常的記錄檔內容上傳至管理主機上，並可在健康主控台上顯示出有異常紀錄產生便於管理者及早發現主機異常。

收集 agent 程式記錄檔的方法:我們可利 Perl 所提供的 FTP 模組將所程式執行時所產生的記錄檔上傳至管理主機上，我們可以透過網頁的方式觀看每台被管理端主機的記錄檔，便於我們對於 agent 程式運作狀況檢查與程式除錯使用。

上述的資料收集程式除了會依據我們的設定定期的收集資料外，當網路管理人員要新增收集目標時也可透過修改 agent 程式，並透過組態設定模組自動將修改後的程式佈署到被管理端的主機上。

5.2.2 資料處理模組之程式

資料處理模組主要分成兩大部分:

1.利用 JSP 動態網頁程式將 Client 端利用 http post 方式上傳的資料藉由預先設計好的 JSP 程式,將相對應的資料藉由 database Link 直接寫入或更新資料庫中,利用這種方式可達到資料即時更新的效果,但是 http post 的方式並不適合大量的資料傳送,例如:Server heartbeat status (如圖三十所示)需即時將 heartbeat 資訊回傳至 management server 就是使用這個方法。

2.利用 SQL *Loader 如 2.5.2 定時將 client 上傳的資料藉由已經設定好的格式(Control file)直接寫入資料庫中,此種方式適用於非即時性與數量較大的資料,附錄一、 為本系統將 Client 上傳 CPU Utilization 的資料利用 load_cpu.ctl(control) 將資料利用 SQL *Loader 寫入資料庫的資訊。

Refresh: Thu Mar 02 00:59:12 CST 2006

Hostname	OS	OS Ver.	Agent Start	Agent Ver.	Message	Last Update
d2catr6a	AIX	5.2	2006-02-27 14:34:59.0	1.94.31		2006-03-02 00:59:12.0
d2catr6b	AIX	5.2	2006-02-28 01:50:29.0	1.94.31		2006-03-02 00:57:21.0
d2catr6c	AIX	5.2	2006-02-28 23:38:51.0	1.94.31		2006-03-02 00:56:53.0
F2ACL01	AIX	5.2	2006-03-01 16:32:21.0	1.94.31		2006-03-02 00:56:22.0
F2DBS01	AIX	5.2	2006-03-01 16:31:54.0	1.94.31		2006-03-02 00:58:55.0

圖三十、 CPSMS HeartBeat Status

5.2.3 報表產生模組之程式

報表產生程式主要是利用 JSP 程式語言來實作,我們所使用到的 JSP 函式包含有:連結資料庫(Thin client Functions)以及處理圖形的 JfreeChart。產生報表的方法主要可分四大類。

1.由管理者預先設定好的格式:利用 JSP 程式透過與資料庫連結將資料由資料庫取出並依預先設定好的格式將資料轉換成網頁的方式,一般使用者只需透過瀏覽器即可以看到所需的資料,由於資料是由使用者點選的同時才由資料庫中取出資料,可確保資

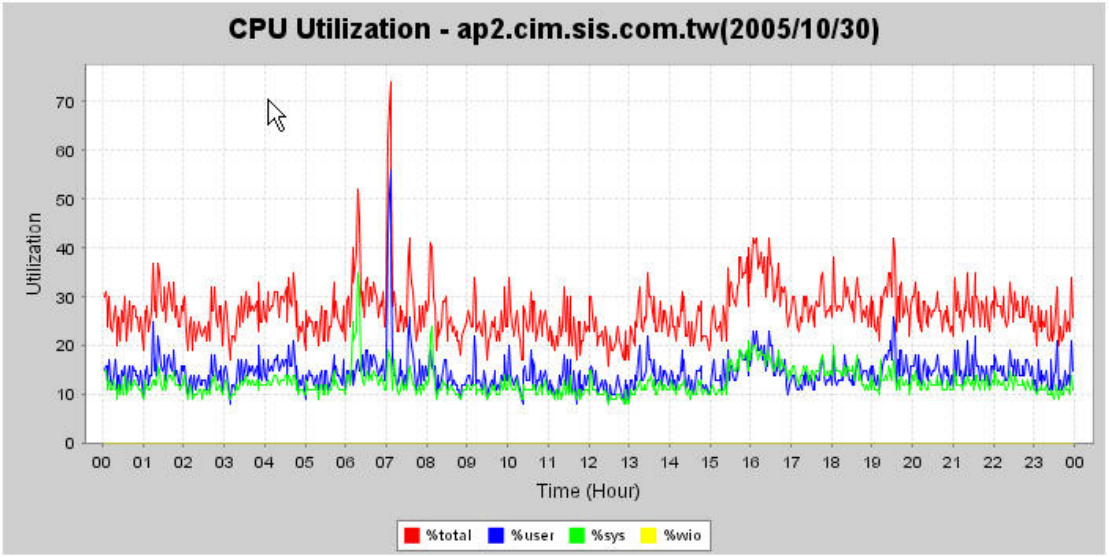
料的即時性與正確性，也就是一般常見的固定報表例如: 各種平台的主機數量、主機設備的詳細資訊。

2.可由使用者依所輸入的條件產生之報表:系統可依據管理者或一般的使用者所選之條件產生其所需之報表。

3.使用者利用其他的報表產生器:如使用 crystal report 自行設計所需的報表依所輸入的條件產生之報表。

4.利用 JfreeChart 程式產生圖形，如圖三十二、所示 CPSMS CPU 使用率趨勢圖

圖三十一、 CPSMS 報表查詢管理畫面



圖三十二、 CPSMS CPU 使用率趨勢報表畫面

5.2.4　組態管理模組之程式

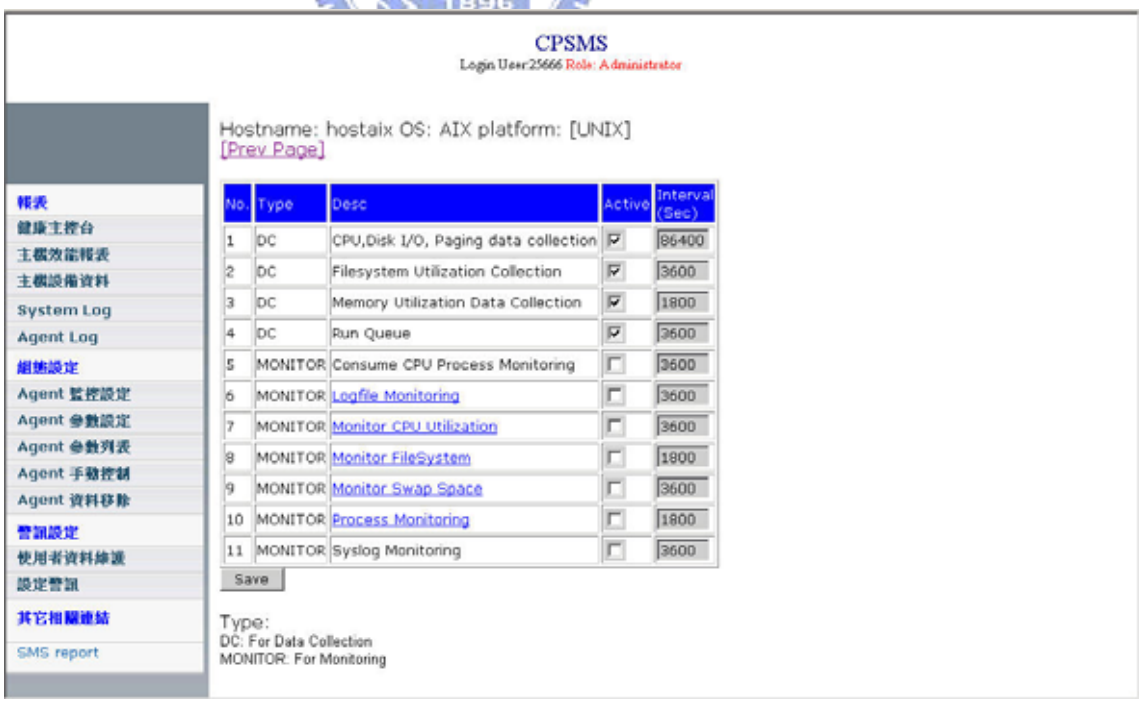
組態設定模組程式主要是利用 JSP 程式語言來實作，主要的功能為設定想要監控或資料收集的項目，設定監控或收集資料的區間與是否啟動該監控項目，透過網頁輸入後將設定值存放至資料庫中如圖三十三所示，被監控端經由 Agent 程式從資料庫讀取其相關的設定值，可分為底下幾個項目：

1.參數修改:

可用於設定監控或資料收集項目取得資料的間隔，例如: CPU/DISK IO 使用率資料收集的區間預設為一天(86400 秒) 、File system 資料收集與監控資料收集區間為一小時(3600 秒) 。

2.啟動控制

可由管理者依各主機特性，選擇是否啟動該監控或資料收集項目，以方便管理與減少收集資料量。



圖三十三、　CPSMS 監控項目設定畫面

5.2.5　警訊模組之程式

警訊模組主要是利用 JSP 程式語言來實作，主要可分為底下幾個項目：

1.自動發送之警訊:

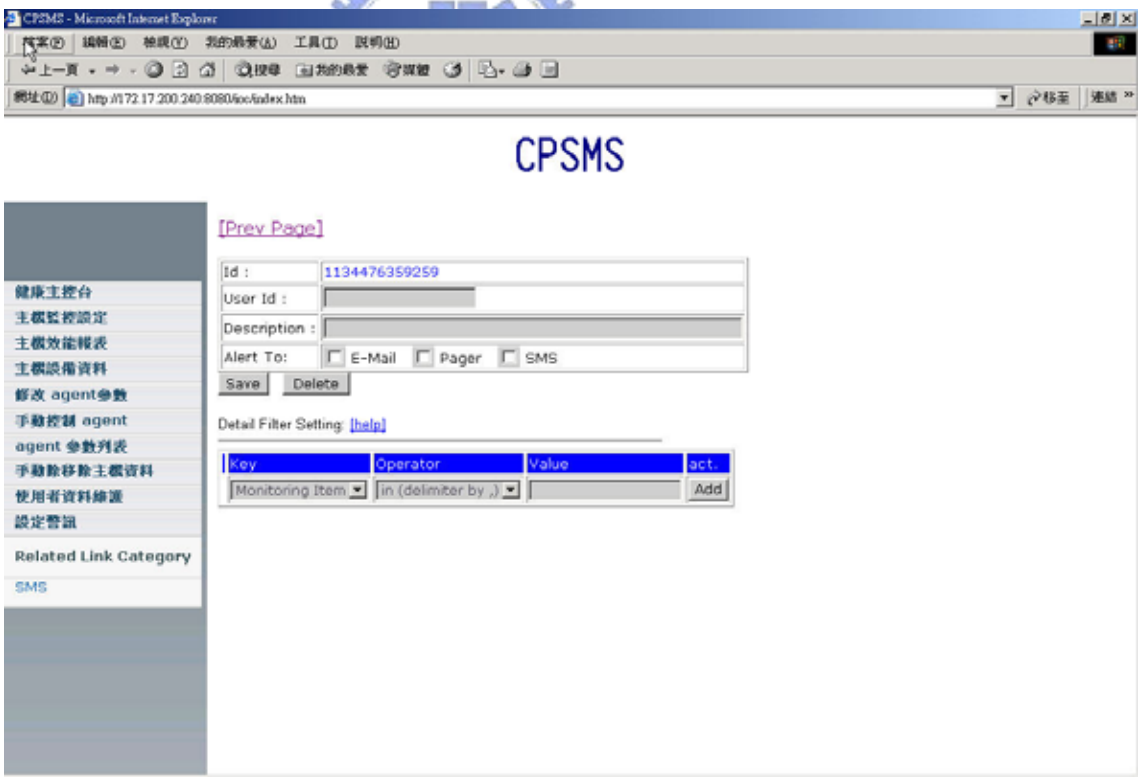
系統可將一些特定的事件設定為當事件被觸發時即會自動發送警訊。

2.警訊項目設定

使用者可藉由相關監控與資料收集，自行設定警訊觸發的門檻。

3.警訊發送設定

使用者可依事件的緊急程度，設定發送警訊的方式，以利於即時且適當的通知相關人員，目前可用的警訊發送方式有底下兩種 Email(圖三十七) ，SMS(圖三十八)



圖三十四、　CPSMS 警訊發送方式設定

如圖三十六所示使用者預先設定主機名稱為(openft,ftp1..)且作業系統為 UNIX、被監控的項目為檔案系統是否已滿、當警訊的等級為 alarm 則發送 Email 和 SMS 通知 User ID 為 66666 的使用者

User Id	66666
User Name	郭大俠
Mail Addr	jeffrey.eic90g@nctu.edu.tw
Pager No	
Cellphone	0930747888

圖三十五、 CPSMS 警訊通報人員資料設定

Id :	1122460776905		
User Id :	66666		
Description :	Server File System Full		
Alert To:	E-Mail Pager SMS		
	Key	Operator	Value
	Monitoring Item	in (delimiter by ,)	
	Alarm_Level	in	alarm
and	Hostname	in	openft,ftp1,dsmtftp.u
and	Monitoring_Item	in	FSFULL
and	Platform	in	UNIX

圖三十六、 CPSMS 警訊發送條件設定

ra2 No HeartBeat

圖三十七、 CPSMS 警訊發送(Email)

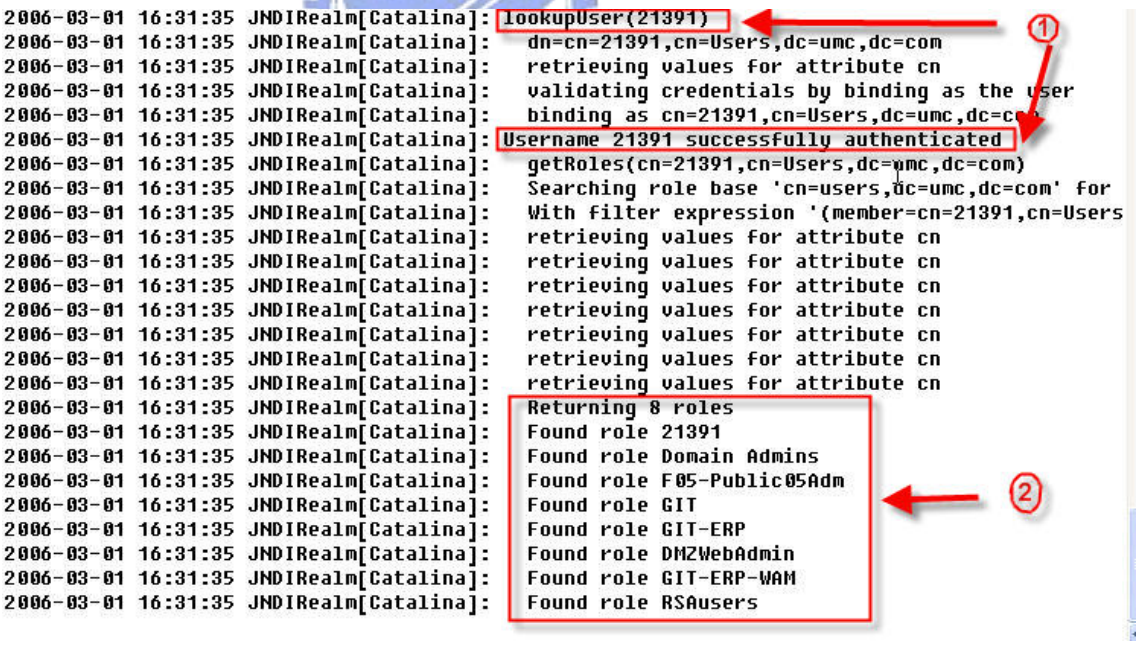
圖三十八、 CPSMS 警訊發送(SMS)

5.2.6 外掛模組程式

外掛模組要是用於處裡其他的系統管理軟體所收集的資料，我們利用預先設計好的 SQL 程式將其所收集的資料與本系統彙總與整理，例如我們將 Microsoft SMS 系統管理資料利用 Microsoft DTS (Data Transfermation Service)將資料直接匯出至 Management 資料庫中 SMS 資料表中，在利用已經設計好的 SQL 程式將我們所需要的資料從 SMS 資料表匯入 CPSMS 相對應的資料表與相關的欄位中，可用於補強本系統所收集資料的完整性，對於其他的系統管理軟體我們也可以用此模組將其所收集的資料匯入本系統。

5.2.7 認證模組程式

認證模組主要是利用 JSP 動態網頁程式語言透過 LDAP 與企業內之 Windows AD Server 認證並取得相關資訊如圖三十九所示，我們可利用 LDAP 所取得使用者的認證資訊作為網頁權限控管的依據，設計範例可參考附錄五



圖三十九、 JSP 透過 LDAP 取得 Windows AD 認證資料

5.3 系統評估

一個完整的主機管理系統必須要能夠整合不同平台的主機，並透過系統達到監看及控制主機的狀況。我們就針對下面五點對本系統做評估。

1.整合性

本系統採用資料庫為核心，可以透過資料庫的連結或經由異質資料庫的轉換，很輕易的就可以和別的系統管理軟體做資料的交換與結合(例如微軟的 SMS) 。

2.完整性

本系統包含一般企業內常見的系統管理需求，並預留可擴充之空間，且對於不同作業系統版本與不同硬體之主機皆可適用。

3.移植性

本系統由於設計時即考慮跨平台使用，所以並不侷限與單一作業系統平台使用，目前已成功移植至 IBM AIX 平台使用，當 management sever 有問題時可很輕易的移轉至另外一台主機。

4.使用性

本系統採用 Web base 當作輸入與輸出的介面，一般使用者與管理者不要另外安裝軟體，只需利用瀏覽器即可使用，方便使用者利用任何可上網的設備(如 PDA, SmartPhone..) 。

5.安全性

系統本身有設計認證機制，可透過 LDAP 與企業內的 AD 授權的方式達到資料的控管，且當企業內人員組織有異動時不需另外維護，可減輕管理者的負擔。

5.4 系統比較

我們將本系統與第三章所提的系統管理系統做比較，如表三，並在之後針對各比較項目做說明。

表三、 主機管理系統比較表

Functions Name	CPSMS	KISMS	OpenNMS	OVO
Agent Platform Support	Windows /Unix/Linux	Windows	Windows /Unix/Linux	Windows /Unix/Linux
Cost	Low	Low	Free	Expensive
Server Requirement	Windows /Unix/Linux	Windows	Linux	Windows /Unix
Web GUI Support	Yes	No	Yes	Yes
Agent Auto Update	Yes	No	No	Yes
Customerlize Report	Yes	No	Yes	Yes
Knowledge-based	No	Yes	No	Yes
Alert by SMS	Yes	No	No	No
Authenticated by AD	Yes	No	No	Yes

1.Agent 支援硬體平台(Agent Platform Support)

除了 KISMS 只支援 Windows 作業系統平台外，本文之 CPSMS 與 OpenNMS HP OVO 皆可適用於目前大部分企業內所使用的作業系統。

2.價格(Cost)

除了 OpenNMS 為 Open source 不需要費用，本文之 CPSMS 與 KISMS 皆為自行開發還是需要有些人力資源的成本，OVO 由於是大型的商業軟體其價格並不是一般中小型企業負擔的起。

3.管理用主機平台

本文之 CPSMS 因為使用 跨平台之程式語言設計所以不論是 Windows/Unix 甚至是 Linux 都可以使用，OVO 則能在 Windows 與特定的 Unix 上運作，至於 KISMS 與 OpenNMS 都只能在單一作業

系統平台式使用。

4.網頁界面管理系統(Web GUI Support)

KISMS 有專屬的顯示畫面(使用 VB 設計)其餘三套系統皆有支援，利用 Web-Based 當作管理界面已成為未來的趨勢。

5.Agent 程式更新

本文之 CPSMS 與 OVO 都有設計 Agent 程式自動更新之功能，OpenNMS 由於沒有用到 Agent 程式所以不需要此功能，KISMS 並無設計 Agent 程式更新功能。

6.客製化報表

除了 KISMS 未提供客製化報表的設計外其餘皆有提供。

7.知識庫

KISMS 與 OVO 皆有提供類似人工智慧知識庫的功能 CPSMS 與 OpenNMS 並無提供相對應的功能。

8.SMS 支援

除了本文 CPSMS 外其餘三套系統目前尚不支援利用手機發送簡訊的警訊發送方式。

9.利用 Windows AD 作認證(**Authenticated by AD**)

CPSMS 與 HP OVO 可藉由透過 Windows AD 作認證，OpenNMS 與 KISMS 無此功能。

第六章　結論與未來研究方向

6.1 結論

本論文已達成下列目標:

1.收集異質平台主機資訊

目前已經確定可以正常運作的作業系統平台，與硬體如下表(表四、表五)所示:

表四、 支援作業系統平台表

作業系統	版本
HP UX	10.2，11.0，11.11，11.23
IBM AIX	4.3.3，5.2
Tru64	4.0F，5.1a，5.1b
Linux (Redhat)	6.2，7.1，9.0，AS2.1，ES3.0，AS3.0，AS4.0
Linux (SuSE)	8.0，9.0
Microsoft	Windows NT4.0，2000，2003

表五、 支援硬體平台表

HP	Model
IBM	H50，H80，S80，F50，B50，6M2，6C4，570
Compaq Alpha	4100，DS10，DS20，ES40，ES45，ES80，GS60，GS80，GS1280
HP	J6000，K380，K570，L2000，N4000，rp3440，rp7620
HP IA64	rx2620，rx4640，rx7620
SUN	220R，880，280R，480R，Ultra 80，E3500，E4500，E10K

2.單一的主機管理介面

本系統採用網頁的方式可與其餘的管理系統的資料作連結，可免除使用者在不同的管理軟體間做切換，未來如果有新的功能也不需要修改 User 端的軟體。

3.發送警訊自動化

本系統可依據使用者或管理者依不同緊急等級條件設定，當系統收到或經由資料處理後達到貨超過使用者或管理者設定之數值，系統即會透過預先定義發送警訊的方法發出警訊。

6.2 未來研究方向

1.利用所收集到的資料作為基礎，加入自我學習的機制。

本系統提供收集主機所產生的系統紀錄檔資料，包含主機軟硬體認證 錯誤等等資訊..。未來希望能將這些資料經過有效率的學習機制，使得系統能在事件發生前就能先提出警訊，讓本系統之管理功能更趨完善。

2.增加更多平台的主機支援。

由於現有的環境中並無 FreeBSD 的作業系統平台的主機，所以目前暫時無法確定是否可在該系統上執行，希望未來能補上對該系統的支援。

3.增加自動直接處理的功能。

另外由於時間上的關係本系統尚無法直接對 Agent 端異常的 process 作處理，也希望將來能加上此功能，使整個管理系統更趨完善。

參考文獻

[1] The OpenNMS Project, http://www.opennms.org

[2] Jill Huntington-Lee, Kornel Terplan, Jeffrey A. Gibson. “ HP OpenView /a manager's guide ”,McGraw-Hill,c1997.

[3] Srinivasan, Sriram “Advanced perl programming”, ,O'reilly & Associates,c1997

[4] Micah Brown, Chris Bellew, Dan Livingston. “Essential Perl 5 for Web professionals”, Upper Saddle River, NJ :Prentice Hall PTR,c1999.

[5] Tom Christiansen and Nathan Torkington. “ Perl cookbook”, O'Reilly & Associates,c1998.

[6] Martin C. Brown. “ Perl programmer's reference”, McGraw-Hill,c1999.

[7] Phil Hanna. “JSP :the complete reference”, McGraw-Hill,c2001

[8] Bob DuCharme, “The operating systems handbook /UNIX, OpenVMS, OS”,McGraw-Hill/c1994.

[9] Cameron Newham and bill Rosenblatt, “Learning the bash shell /UNIX shell programming”, O'Reilly & Associates, Inc.,c1995

[10] David Tansley. , “Linux and Unix shell programming”, Addison-Wesley,2000.

[11] The NET-SNMP Project, http://net-snmp.sourceforge.net/

[12] William Stallings,” SNMP and SNMPv2:The Infrastructure for Network Management,” IEEE Communications,Vol.36,No.3,March 1998,p37-45

[13] The SYSSTAT Utilities, http://perso.wanadoo.fr/sebastien.godard/

[14] The CIM Project, http://www.dmtf.org

[15] The CPAN Project, http://www.cpan.org

[16] Java Server Pages Technology ,http://java.sun.com/products/jsp/

[17] Yutaka Nakamura, Shinji Shimojo, Suguru Yamaguchi, Eiji Kawai, and Hideki Sunahara, "Development of a WWW Server Management Support System," Applications and the Internet (SAINT) Workshops, 2002. Proceedings. 2002 Symposium on, Feb. 2002, pp. 99-106.

[18] Oracle,http://www.oracle.com/index.html

[19] 李中銘 ，“以知識庫為基礎之智慧型伺服器自動監控系統 ”，中華大學碩士學位論文，民 93。

[20] 陳志良, "伺服器健康診斷及簡訊報知系統," 2003 年台灣網際網路研討會, pp.185-188

[21] 林上傑,林康司著, “ JSP 2.0 技術手冊”, 碁峰,民 93

[22] 陳會安著, “JSP 2.0 網頁設計範例教本”, 學貫行銷,民 94

[23] 位元文化著, “JSP 動態網頁入門實務(Linux 版) ”, 文魁,民 91

[24] Carasik, Anne H.著/謝崑中譯, “LINUX 系統管理”, 博碩,民 88

[25] 李蔚澤著, “Red Hat Linux 系統管理”, 麥格羅.希爾,民 91

[26] Nemeth, Evi 等著/鄭士豪,林英超,蕭景鴻譯, “UNIX 系統管理手冊”, 台灣培聲教育,民 91

[27] Winsor, Janice 著/林志強譯, “Solaris 系統管理”, 台灣培聲教育,民 91

[28] DeRoest, James W.著/劉祖亮,陳李雍譯, “AIX RS.6000 完全系統管理手冊”, 麥格羅.希爾,民 86

[29] Spalding, George 著/吳東賢譯, “Windows 2000 系統管理徹底研究”, 麥格羅.希爾,民 90

[30] Blank-Edelman, David N.著/蔡憶懷,蔣大偉,林長毅譯, “多平台環境系統管理”, 歐萊禮,民 90

[31] 毛元君・梁竹柳著, “新洞悉 UNIX－－系統管理篇”, 和碩科技,民 86

[32] Carter, Gerald 著/蔣大偉譯, “LDAP 系統管理”, 歐萊禮,民 92

[33] Veeraraghavan, Sriranga 著/陳清豪,廖家鋒譯, “即學活用 Shell Programming”, 博碩,民 90

[34] 林存德著, “Oracle 9i 資料庫管理指南”, 旗標,民 91

[35] Adkoli, Anand & Velpuri, Rama 著/何致億,潘得龍譯/(一)/系統管理手冊, “Oracle9i Windows 系列”, 麥格羅.希爾,民 91

[36] Mishra, Sanjay & Beaulieu, Alan 著/艾瑞克譯, “精通 Oracle SQL”, 歐萊禮,民 92

附錄一

SQL*Loader Example

perf_cpu.data

utrvrd03,2005/06/26,00:02:00,0,0,0,1

utrvrd03,2005/06/26,00:04:00,0,1,0,1

utrvrd03,2005/06/26,00:06:00,0,1,0,1

utrvrd03,2005/06/26,00:08:00,0,0,0,1

utrvrd03,2005/06/26,00:10:00,0,0,0,1

utrvrd03,2005/06/26,00:12:00,0,0,0,1

utrvrd03,2005/06/26,00:14:00,0,0,0,1

load_cpu.ctl

```
options (rows=1500, bindsize=10000000)
load data
infile 'x:\cpsms\perf\work\perf_cpu.dat'
truncate
into table w_perf_unix_cpu
fields terminated by ","
(
hostname     ,
logdate    ,
logtime        ,
ut_user     ,
ut_sys      ,
ut_wio      ,
ut     )
```

load_cpu.log

```
SQL*Loader: Release 9.2.0.1.0 - Production on 星期三 7月 23 13:11:18 2003

Copyright (c) 1982, 2002, Oracle Corporation.  All rights reserved.

Control File:   x:\cpsms\perf\load_cpu.ctl
Data File:      x:\cpsms\perf\work\perf_cpu.dat
  Bad File:     x:\cpsms\perf\perf_cpu.bad
  Discard File:  none specified

 (Allow all discards)

Number to load: ALL
Number to skip: 0
Errors allowed: 50
Bind array:     1500 rows, maximum of 10000000 bytes
Continuation:    none specified
Path used:      Conventional

Table W_PERF_UNIX_CPU, loaded from every logical record.
Insert option in effect for this table: REPLACE

   Column Name                  Position   Len  Term Encl Datatype
------------------------------ ---------- ----- ---- ---- ---------------------
HOSTNAME                           FIRST     *   ,       CHARACTER
LOGDATE                             NEXT     *   ,       CHARACTER
LOGTIME                             NEXT     *   ,       CHARACTER
UT_USER                             NEXT     *   ,       CHARACTER
UT_SYS                              NEXT     *   ,       CHARACTER
UT_WIO                              NEXT     *   ,       CHARACTER
UT                                  NEXT     *   ,       CHARACTER


Table W_PERF_UNIX_CPU:
  703 Rows successfully loaded.
  0 Rows not loaded due to data errors.
```

附錄二

```
####      perl http post example    ####
use Net::HTTP::NB;
sub request_server
{
      my($method,$url,$content) = @_;
      logger("beg> request server","DEBUG");
      my($server,$port)=get_activeserver();
      if ("$server" eq "0") {
   logger("request_server: Sorry !! No available server!!","ERROR");
          logger("<end request server","DEBUG");
          return;
      }
      $method=uc($method);
      if ($method eq "GET") {
          my($rt,$buff,$n);
          $rt="";    $n=0;
          while ($rt ne "200" && $n<3) {
              ($rt,$buff) = http_get($server,$port,$url);
              ($server,$port)=get_activeserver();
              $n++;
              last if ($rt eq "200" || $n>=3);
              sleep(3);
          }
          logger("<end request server","DEBUG");
          return ($rt,$buff);
```

```
        }
        if ($method eq "POST") {
            my($rt,$buff,$n);
            $rt="";    $n=0;
            while ($rt ne "200" && $n<3) {
                ($rt,$buff) = http_post($server,$port,$url,$content);
                $n++;
                last if ($rt eq "200" || $n>=3);
                sleep(3);
            }
            logger("<end request server","DEBUG");
            return ($rt,$buff);
        }
    }
    sub st_check_sys_time
    {
        my $btm = time;
        my($rt,$buff,$subname,$ltm);
        $subname = "st_check_sys_time";
        ($rt, $buff) = request_server("POST",
"$g_urlroot/st_check_sys_time.jsp", "X=0");
        my $atm = time;
        $ltm = ($btm + $atm) /2;
        $g_taskmesg{$subname} = "";
        if ($buff =~ /Success/ && abs($ltm - (split(/,/,$buff))[1]) > 10) {
       $g_taskmesg{$subname} = sprintf(" Time skew : %d seconds",
                                        $ltm - (split(/,/,$buff))[1]);
        }}
```

附錄三

```
### perl ftp example    #####
my($g_ftpserver,$g_ftpuser,$g_ftppwd,$g_os,$g_hostname);
$g_ftpserver = "192.168.1.2";
        $g_ftpuser = "ftp";
        $g_ftppwd = "ftp@passwd";

sub send_data()
{
        my($dd) = @_;
        my($ftpcmd,@flist,$f,$outmesg);
        $ftpcmd="ftp -v -n $g_ftpserver << EOF\n";
        $ftpcmd="${ftpcmd}user $g_ftpuser $g_ftppwd\n";
        $ftpcmd="${ftpcmd}cd incoming\n";
        $ftpcmd="${ftpcmd}cd perf\n";
        $ftpcmd="${ftpcmd}lcd /tmp\n";
        @flist=`ls /tmp/sar_*_*.$dd`;
        for $f (@flist) {
                chop($f);
                my($null,$dir,$fn) = split(/\//,$f);
                print "$fn\n";
                $ftpcmd="${ftpcmd}put $fn\n";
        }
        $ftpcmd="${ftpcmd}bye\n";
        $ftpcmd="${ftpcmd}EOF\n";
        $outmesg=`$ftpcmd`;
```

附錄四

System log Check 白名單

ftpd[1234]:

ftpd[[0-9]+]:

inetd[1234]: ftp/tcp: Connection from

inetd[[0-9]+]: ftp/tcp: Connection

su: ?? root-xxxx

su: .+ root-

su : + ?? root-xxxx

su : .+ root-

in.mpthd[1234]:

in.mpathd[[0-9]+]:

root: Solstice Backup media:

root: Solstice Backup media:

last message reeated 1 time

last message repeated [0-9]+ time

above message repeats 2 times

 above message repeats [0-9] time

 above message repeats [0-9][0-9] time

telnetd[...]: getpid: peer died: Error 0

telnetd[[0-9]+]: getpid:

xntpd[...]:

xntpd[[0-9]+]:

附錄五

JSP 認證程式範例

```
<%

out.println("Login User:" + request.getUserPrincipal().getName());

if (request.isUserInRole("GIT-EDC")) {
out.println("<Font color=red>  Role: Administrator</Font>");
.....administrator web page ............
}
else {
out.println("<Font color=Green>  Role: General User</Font>");
....general user web page.................
}
%>
```

附錄六

利用 UNIX 內建 SHELL 與指令取得 UNIX 硬體資訊範例

```
sub xt_unix_dc_system_info_osf
{
      my($model,$ram,$cpu_count,$cpu_speed);

      $cpu_count=`sizer -p`; $cpu_count =~ s/\n//g;

      $cpu_speed=`psrinfo -v|grep Hz|awk '{print \$8}'| tail -1`;
      $cpu_speed =~ s/\n//g;

      if ( -X "/sbin/hwmgr" || -X "/usr/sbin/hwmgr") {
          $model=`hwmgr show component | grep AlphaServer | tail -1 | cut -c 34-`;
          $model=~ s/\n//g;
      }
      else {
          $model=`grep AlphaServer /var/adm/messages* |tail -1|awk '{print \$6 ,\$7,
\$8}'`;
          $model =~ s/\n//g;
      }

      $ram=`vmstat -P | grep "Total Physical Memory =" | cut -d"=" -f 2`;
      $ram =~ s/\n//g; $ram =~ s/M//g;

      return($model,$ram,$cpu_count,$cpu_speed);
}
```

附錄七

利用 WMI 取得 Windows 硬體資訊範例

```
if ($info{'Windows Version'}{'CurrentVersion'} =~ /5\.\d/) {
        my $strQuery = "Select * From Win32_LogicalMemoryConfiguration";

        my $WMI = Win32::OLE->GetObject("winmgmts://");
        my $Results = $WMI->ExecQuery($strQuery);

        foreach my $Interface ( in $Results ){
          if ($Interface eq undef){
              # No Information
          } else {
              $info{'Memory'}{'TotalPhysicalMemory'} =
                      $Interface->TotalPhysicalMemory;
               logger(sprintf("%sMemory%sTotalPhysicalMemory%s===%s", $del,
$del, $del, $info{'Memory'}{'TotalPhysicalMemory'}));
              $info{'Memory'}{'TotalPageFileSpace'} =
                      $Interface->TotalPageFileSpace;
logger(sprintf("%sMemory%sTotalPageFileSpace%s===%s", $del, $del, $del,
$info{'Memory'}{'TotalPageFileSpace'}));
              $info{'Memory'}{'TotalVirtualMemory'} =
                      $Interface->TotalVirtualMemory;
              logger(sprintf("%sMemory%sTotalVirtualMemory%s===%s", $del, $del,
$del, $info{'Memory'}{'TotalVirtualMemory'}));
              $info{'Memory'}{'AvailableVirtualMemory'} =
                      $Interface->AvailableVirtualMemory;
              logger(sprintf("%sMemory%sAvailableVirtualMemory%s===%s", $del,
$del, $del, $info{'Memory'}{'AvailableVirtualMemory'}));
          }
        }
```