具負載平衡機制之雲端服務中介平台設計與實作

許育倫
國立屏東科技大學資訊管理系
M9956001@mail.npust.edu.tw

吳哲一
國立屏東科技大學電子計算機中心
joey@mail.npust.edu.tw

陳志華*
國立交通大學資訊管理研究所
chihua0826@gmail.com

蔡騰緯
國立屏東科技大學資訊管理系
M10056002@mail.npust.edu.tw

龔旭陽
國立屏東科技大學資訊管理學系
kung@mail.npust.edu.tw

摘要

資訊科技演進又有新的技術發展，雲端運算(Cloud Computing)概念在 2006 年由 Google 執行長於搜尋引擎大會正式提出，主要形容網路上服務的無所不在。雲端運算主要可提供使用者高效能(High Performance)運算、分散式儲存與通訊等服務，在同一時間內更可負載大量使用者之服務請求。除此之外，虛擬化(Virtualization)技術更是雲端運算能夠快速發展關鍵之一，可輕易達到快速部署與動態資源調度，在近幾年雲端運算已逐漸成為各家資訊科技業者發展之重點目標。

有鑑於此，本論文設計一「具負載平衡機制之雲端服務中介平台(A Cloud Service Middleware with Load Balance Mechanism)」，結合具彈性(Flexible)與跨異質平台(Crossing Heterogeneous Platform)之網路服務(Web Service)，提供開發者能夠將開發之網路服務註冊與部署至雲端環境，透過排隊理論實現雲端運算環境中虛擬機器(Virtual Machine)之系統動態負載平衡(Load Balance)機制，減少虛擬機器因負載不平衡之狀態影響使用者服務情況。此外，本系統針對使用者互動方面，採用獨立豐富型網際網路應用程式與使用者介面標記語言(User Interface Markup Language)進行介面之描述，以提升系統彈性。

關鍵詞：雲端運算、負載平衡、網路服務、排隊理論、中介平台

壹、緒論

近年來，雲端運算已成為資訊科技產業發展主要目標，國際著名科技大廠也紛紛將資源投入雲端技術與服務之研發，如 Google、IBM、Microsoft、Yahoo、Amazon、Oracle 等，雲端服務類型常會分成以基礎建設為服務(Infrastructure as a Service, IaaS)、以平台為服務(Platform as a Service, PaaS)，以及以軟體為服務(Software as a Service, SaaS)，其本質皆需透過網路連結以獲取軟體或服務[2, 4]，雲端運算包含了許多技術元件，其中最重要則是虛擬化(Virtualization)，可將伺服器主機資源切割成多台虛擬機器(Virtual Machine, VM)，並且能夠快速部署至伺服器主機，當虛擬機器無法運作時，也不至於影響伺服器主機[6]。

在中介平台軟體技術中以服務導向架構(Service Oriented Architecture, SOA)之 Web Service 最為熱門，Web Service 實現了跨異質平台之能力，採用開放式標準之延伸標記語言(eXtensible Markup Language, XML)建構以簡單物件存取協定(Simple Object Access Protocol, SOAP)為其通訊協定，但目前傳統 Web Service 註冊與部署方式無法承受大量服務請求(Request)，並且服務品質(Quality of Service, QoS)取決於服務提供者之伺服器主機，因此結合雲端運算環境將開發之 Web Service 部署至雲端虛擬機器，為了有效利用雲端運算環境之優勢，透過動態負載平衡機制，隨著雲端虛擬機器不同之狀態能有效將其服務請求分散，可維持系統與服務之穩定。

有鑑於目前尚未有雲端運算相關之標準架構與協定，本論文設計一「具負載平衡機制之雲端服務中介平台(A Cloud Service Middleware with Load Balance Mechanism)」，目標是透過雲端虛擬機器提供服務開發者將自行開發之 Web Service 快速部署至伺服器中，針對目前虛擬機器負載情況或運作狀態，藉由動態負載平衡機制進行服務請求重新導向(Redirect)。為了達到 Web Service 部署與動態負載平衡機制，本平台提出具有重新導向伺服器(Redirect Server)之架構，當服務提供者將 Web Service 部署至雲端伺服器，會自行產生該 Web Service 對應之重新導向服務(Redirect Service)，重新導向服務會依據虛擬機器負載情況或運作狀態回應(Response)透過排隊理論(Queueing Theory)運算該虛擬機器網路服務拒絕率(Blocking Probability)，提供使用者較適當之服務位置。使用者介面部份，利用豐富型網際網路應用程式(Rich Internet Applications, RIA)概念與使用者介面標記語言(User Interface Markup Language)，透過網路伺服器提供使用者所需之應用程式，使用者透過網頁瀏覽器(Web Browser)即可進行使用，並利用使用者介面標記語言獨立描述使用介面，以增加其系統維護彈性。

本論文共分為四個章節，第二節針對技術背景與相關學者提出之議題進行討論，第三節則為本論文預計設計之雲端服務中介平台(Cloud Service Middleware, CSM)，並對於該平台架構下之設計與運作流程進行說明，最四節則為結論與未來研究。

貳、文獻探討

一、 雲端運算

服務導向架構是軟體中介平台最早被提出具彈性、可重複使用之架構模型，主要由標準化網頁服務技術元件組成。因此，學者 Wei 與 Blake 探討服務導向運算(Service Oriented Computing, SOC)與雲端運算(Cloud Computing)之關聯性，認為企業為提升競爭力，在短時間內經常需要新增新功能或刪除舊功能，SOC 則是利用基本服務來快速建構低成本、安全與可靠的應用程式，藉此可減少每次開發新的軟體元件所必需的處理流程[10]。在分散式環境透過網路服務描述語言(Web Service Definition Language, WSDL)即能夠執行複雜運算，藉由 SOAP 或表徵狀態轉移(Representational State Transfer, REST)協定則可提供與外部連結之介面。

綜合上述，文中認為雲端運算是建構在傳統分散式運算(Distributed Computing)或是網格運算(Grid Computing)上，可提供具有動態縮放之商業模式，在文中也探討幾項未來雲端運算需面臨之挑戰與機會，例如：保持服務高可靠性(High Service Availability)、安全性問題以及透過聯合雲(Federated Clouds)發現可用服務等，最後提出雲端運算將需面對最新挑戰將是「如何在雲端運算環境中執行服務導向運算」[10]。Al-Shammary、Khalil 等學者提出了雲端網路服務(Cloud Web Service)[1]之概念，透過 SOAP 通訊能夠使雲端網路服務不受平台限制。文中認為採用 XML 編碼技術之 SOAP 訊息交換方式所造成擁塞問題會影響雲端網路服務性能因素，因此提出可藉由減少 SOAP 訊息大小方式，以提升訊息交換速度減少擁塞問題。但是造成擁塞問題並非只是訊息大小，服務處理速度與複雜度也可能造成擁塞問題，藉由良好服務排程機制將使用者服務請求分散至不同地方即可解決部份問題。

另外針對雲端服務平台 Siddhisena、Warusawithana 等學者則提出具三層次之雲端服務平台 Uranus 架構，此架構包含網域名稱系統層(DNS Layer)、反向代理層(Reverse Proxy Layer)與虛擬層(Virtualization Layer)[8]。在網域名稱系統層主要負責使用者重新導向至適當的反向代理層，反向代理層的網域名稱註冊於網域名稱系統層，此架構將網域名稱系統層置於 Uranus 外部使它部署更具彈性，反向代理層則是一個或許多 Apache 伺服器組成，運行標準 HTTP Port (80)負責與虛擬層進行連結並將 Apache 伺服器的 HTTP 服務請求移交到虛擬層端口，透過調整反向代理層快取模組(Cache Module)，降低虛擬層因反向代理層成為熱點(Hotspot)所造成之服務請求數，虛擬層組成包含數個 Apache 與 MySQL，虛擬層實現多租戶(Multi-tenant)概念。學者在虛擬層提出 Apache 與 MySQL 虛擬化之方式，每個租戶配有個人處理之專用主目錄，不同於硬體虛擬化成作業系統映像檔(Operating System Image)方式需包含所有應用程式，此架構可有效減少硬體虛擬化之映像檔產生，但透過反向代理層之服務請求方式可能會因反向代理層負載過大而影響虛擬層性能。

二、 負載平衡

學者 Wee 與 Liu 則認為傳統對於大規模 Web 應用程序所使用的負載平衡標準方法皆是採用硬體設備的負載平衡方式。對於硬體設備來說一個 IP 位置等同於一個 Web 應用程序，並且與所有 Web 應用程序通訊之內容皆需透過該設備進行。設備後端則是連結一個或多個相同的 Web 伺服器，依據每個 Web 伺服器不同負載情況進行封包的轉送。但負載平衡硬體設備往往是使用特定應用程序的設備元件並且昂貴，因此在雲端環境中常見方式是在通用伺服器上執行負載平衡軟體技術。針對負載平衡軟體技術，則利用“使用者端(Client-side)負載平衡”[9]之概念，其負載平衡架構主要建置於 Amazon S3，主要包含了兩個關鍵元件分別為(1) Web 伺服器負載檔案與(2) 負載平衡的腳本(JavaScript)，運作方式則是依據 Web 伺服器負載檔案去確認目前有那些 Web 伺服器可以發送服務請求，再藉由加權隨機分佈(Weighted Random Distribution)演算法避免在同一時間產生伺服器熱點，訊息傳送過程主要是透過 JavaScript 發送至目標代理服務器(Proxy)。但利用 JavaScript 傳送訊息方式會對使用者使用上有安全性疑慮，另外透過加權隨機分佈演算法僅能減少熱點的產生，並無法完全有效的達到負載平衡。

Ren、Lin 等學者認為目前仍沒有統一的標準去計算伺服器負載情況，因此針用在運作過程中主要會影響伺服器的各種參數(CPU、記憶體、使用者連結數、硬碟的使用大小)，透過係數向量方式來呈現各參數在伺服器中總負載的比例[7]，但伺服器影響參數如果產生劇烈的浮動，則可能無法有效透過參數來評估伺服器當前負載狀況。另外，Hui、Zhao 等學者則針對傳輸距離，提出以距離最近與負載最小之基於雲端多媒體負載平衡(Cloud-based Multimedia Load Balancing, CMLB)演算法[3]，其演算法主要是以使用者為基準點定義範圍內延遲的三個層次，分別為 L0 (0, 20)、L1 (20, 50)與 L2 大於 50。並且透過各節點(Node)的延遲時間取得在定義範圍內可用的設置節點，進行節點伺服器使用率測量與網路流量偵測。透過測量結果數值計算其成本(Cost)，依據成本優劣判斷是否為最佳節點，並將該節點加入最佳節點列表，當列表中有多個最佳節點則使用者則可以任意選擇一個節點使用。該演算法目標是減少節點到使用者端多媒體傳輸之成本，但透過延遲時間取得在範圍內的節點方法可能因網路狀況不穩而導致節點延遲時間過長情況發生，無法確實算出節點位置。

除此之外， Lu、Xie 與 Kliot 等學者提出了加入空閒佇列(Join-Idle-Queue, JIQ)之負載平衡演算法[5]。在此演算法架構具有 n 個透過網路連結的同質性處理器，且有 m 個調度器負責處理請求訊息調度，其中 $m < n$，系統之請求到達率為 λ 之泊松(Poisson)事件與服務時間為常態服務時間分佈，每一請求事件到達會隨機分派至任意調度器中再經由它分派給處理器。調度器與處理器之間定義多個 I-Queue，當有服務請求時並且 I-Queue 有空閒(Idle)時將其存入至佇列中，藉由處理器目前佇列情況以最短佇列或隨機方式分佈進行分析，其目標是為能夠讓處理器快速回應，避免過多通訊消耗的產生。學者認為特別在最關鍵的通訊路徑，因為過多的通訊消耗反而會造成整體回應時間增加。

參、研究架構與方法

本論文設計一雲端服務中介平台，系統架構如圖 1 所示，為能增加其系統彈性、可擴充性及可利用性，獨立將網路服務、工作流程及使用者介面分層獨立運作，以能夠迎合企業及使用者網路服務之使用需求。因此 CSM 架構共可分成四層(4 Layers)，分別為(1) 網路服務層(Web Service Layer)：主要為由服務提供者(Service Providers)負責提供或由服務需求者(Service Requesters)所自行建置網路服務，經由服務負載平衡中介層所提供中介平台提供者(Middleware Provider)將服務註冊至重新導向伺服器以及部署至虛擬機器層；(2) 虛擬機器層(Virtual Machine Layer)：為能提供服務開發者高效能與可靠性雲端運算環境，利用虛擬化技術有效將實體伺服器進行調度；(3) 服務負載平衡中介層(Service Load Balance Middleware Layer)：為能有效將雲端服務利用，面之關聯對應，並提供一整合服務之使用介面，讓程式設計師能快速上傳及新增新的服務，使用者亦可單純化操作所有整合服務；以及(4) 使用者應用層(Client Application Layer)：經由服務負載平衡中介層所設計之雲端服務中介平台後，使用者透過網頁瀏覽器(Web Browser)即可直接使用所有雲端服務，以下針對各層之主要設計進行細部說明。

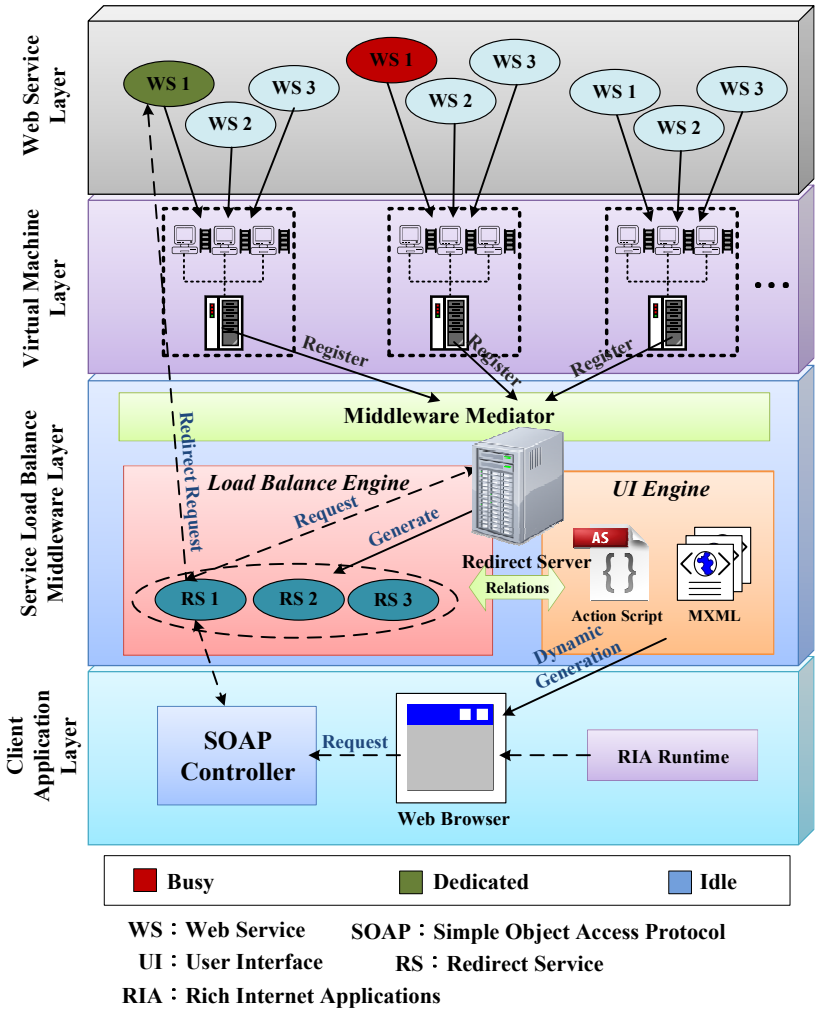

圖 1、系統架構圖

一、 網路服務層

網路服務層即為雲端服務中介平台運算單元。網路服務主要由服務提供者所建置，為提升其網路服務負載程度，網路服務建置完成後，需透過服務負載平衡中介層(Service Load Balance Middleware Layer)上傳並部署至虛擬機器層(Virtual Machine Layer)中。再依照使用者服務之需求，透過重新導向伺服器(Redirect Server)提供使用者適當之網路服務位置，因網路服務具有良好使用之彈性，因此服務提供者透過中介平台即可進行網路服務刪除或重新部署之功能。

二、 虛擬機器層

虛擬化技術是雲端運算關鍵技術之一。虛擬機器層主要利用其虛擬化軟體(例如：Virtualbox、VMware)將實體伺服器資源有效劃分成數個虛擬機器(Virtual Machine)。而虛擬機器(Virtual Machine)本身即是一個具有作業系統的完整系統平台，如圖 2 所示。其虛擬機器之資源具有可重覆利用與可調性，依據使用者需求不同，可進行虛擬機器資源的重新配置。傳統將網路服務部署至實體伺服器之方式，容易受到許多外部之影響導致網路服務停擺，將網路服務部署至虛擬機器中，即便有一個或多個虛擬機器無法運作，也不會影響到其他虛擬機器運作，當有需求產生時也可快速將虛擬機器之網路服務環境快速部署並開始運作。

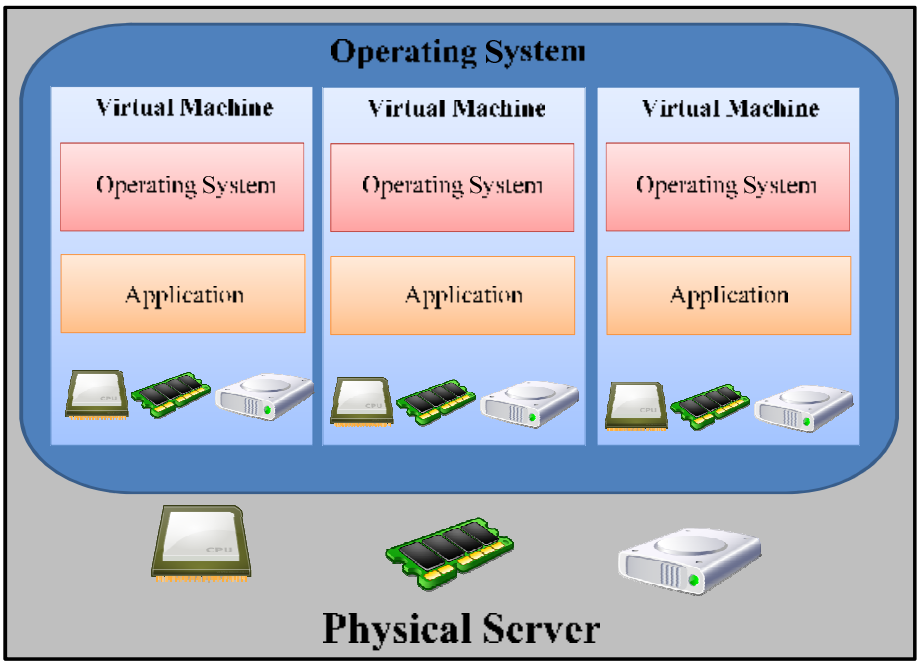

圖 2、虛擬化技術(Virtualization)

三、 服務負載平衡中介層

服務負載平衡中介層主要提出一負載平衡機制有效分散使用者服務請求，服務負載平衡中介層本身具備重新導向伺服器。當有服務請求產生時，需透過重新導向伺服器運算取得網路服務位置，本機制主要採用排隊理論(Queueing Theory)分析拒絕率進行服務

請求之策略排程規劃。因網路服務必需執行在 Apache、IIS 等，故需考量 Apache 能夠處理網路服務請求之最大數量。因此，負載平衡機制主要採用 M/M/c/c 模型，如圖 3 所示，其模型假設如下：

- 系統服務事件到達為呈現泊松(Poisson)分佈，平均到達率為 λ (request/minute)
- 服務時間呈現指數分佈，平均服務時間為 $1/\mu$ (minute/request)

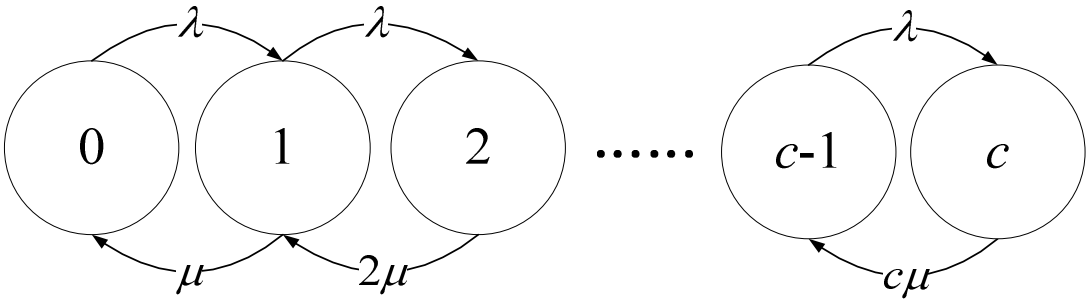

圖 3、M/M/c/c 模型[4]

在此部份，我們以公式(1)分析 k 個服務請求在系統之機率 P_k，令 $\rho = \lambda / \mu$。運用 M/M/c/c 模型以公式(2)推導服務拒絕率 P_C，並推導從以 k 個服務請求機率 P_k 進入服務拒絕率 P_C 之進入拒絕機率 $B(k,c)$，如公式(3)所示。選擇進入拒絕機率 $B(k,c)$ 最低的網路服務伺服器提供服務。

$$P_k = \frac{1}{k!}(\frac{\lambda}{\mu})^k P_0 \tag{1}$$

$$P_C = \frac{\rho^c}{c!} P_0 = \frac{\frac{\rho^c}{c!}}{\sum_{k=0}^{c} \frac{\rho^k}{k!}} \tag{2}$$

$$B(k,c) = \Pr\left(P_C \mid P_k\right) = \frac{\frac{\rho^c}{c!} P_0}{\frac{\rho^k}{k!} P_0} = \frac{\rho^{c-k}}{\prod_{i=k+1}^{c} i} \tag{3}$$

四、 使用者應用層

標準網路服務使用者應用程式開發方式，由程式設計師針對使用者需求進行開發，最後將使用者應用程式編譯成執行檔，直接於使用者端(Client Side)進行執行使用。因此當未來服務需求改變或其他因素必須進行系統更新修改，包含網路服務、組合服務或使用者介面。必須依其修改部份進行修改，並將所有部份(包含使用者操作邏輯、介面)程

式碼進行重新編譯成執行檔。因此採用此開發方式對於程式設計師對於進行系統維護將造成負擔，並且也容易造成使用者所操作之版本不一致情況。有鑑於此問題，本論文針對使用者應用端部份，結合豐富型網際網路應用程式概念，讓使用者直接透過瀏覽器即可使用本論文設計之雲端服務中介平台。

肆、結論與未來研究

本論文設計一具負載平衡機制之雲端服務中介平台設計與實作，採用雲端運算與服務導向運算之概念設計，利用 RIA 技術將使用者介面獨立描述，因此，此雲端服務平台兼具效能與彈性之服務平台，服務開發者藉由中介平台所提供之發佈上傳介面，能快速將建置完成之網路服務上傳，系統即會自動將其部署至雲端運算環境，並將該網路服務相關資訊註冊至重新導向伺服器。本論文提出之負載平衡機制具有動態調度，則會利用重新導向伺服器依據異質狀態之虛擬機器目前服務狀況藉由排隊理論推導出其服務拒絕率進行負載平衡動作，減少與虛擬機器之間訊息交換所造成之消耗，預期未來針對虛擬機器不同資源虛擬化設定條件下，負載平衡狀況與壓力測試數據分析與探討，並驗證使用者網路服務到達率與伺服器服務率是否符合其假說。

誌謝

本研究承蒙國科會計畫之補助支持，核定計畫編號為 NSC100-2218-E-020-001 及 NSC100-2218-E-020-002，特此誌謝。

參考文獻

1. Al-Shammary, D., Khalil, I., and George, L.E., “Clustering SOAP web services on internet computing using fast fractals,” *Proceedings of IEEE International Symposium on Network Computing and Applications*, Cambridge, MA, 2011.
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., and Zaharia, M., “A view of cloud computing,” *Communications of the ACM* (53:4), April 2010, pp. 50-58.
3. Hui, W., Zhao, H. Y, Lin, C., Yang, Y., “Effective load balancing for cloud-based multimedia system,” *Proceedings of IEEE International Conference on Electronic and Mechanical Engineering and Information Technology*, Harbin, China, 2011.
4. Lawton, G., “Developing Software Online With Platform-as-a-Service Technology,” *Computer* (41:6), June 2008, pp. 13-15.
5. Lu, Y., Xie, Q., Kliot, G., Geller A., Larus, J., and Greenberg, A., “Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services,” *Performance Evaluation* (68:11), November 2011, pp. 1056-1071.
6. Mell, P. and Grance, T., “The NIST definition of cloud computing,” *National Institute of Standards and Technology Special Publication* 800-145, September 2011, pp. 1-3.

7. Ren, X., Lin, R., and Zou H., "A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast," *Proceedings of IEEE International Conference on Cloud Computing and Intelligence Systems*, Beijing, China, 2011.
8. Siddhisena, B., Warusawithana, L., and Mendis, M., "Next generation multi-tenant virtualization cloud computing platform," *Proceedings of the 13th IEEE International Conference on Advanced Communication Technology*, Seoul, Korea, 2011.
9. Wee, S. and Liu, H., "Client-side load balancer using cloud," *Proceedings of the 25th ACM Symposium on Applied Computing*, New York, USA, 2010.
10. Wei, Y. and Blake, M. B., "Service-oriented computing and cloud Computing: challenges and opportunities," *Internet Computing* (14:6), November 2010, pp. 72-75.

A Cloud Service Middleware with Load Balance Mechanism: Design and Implementation

Yu-Lun Hsu
Department of Management Information System, National Pingtung University of Science and Technology
M9956001@mail.npust.edu.tw

Che-I Wu
Computer Center, National Pingtung University of Science and Technology
joey@mail.npust.edu.tw

Chi-Hua Chen*
Institute of Information Management, National Chiao Tung University
chihua0826@gmail.com

Teng-Wei Tsai
Department of Management Information System, National Pingtung University of Science and Technology
M10056002@mail.npust.edu.tw

Hsu-Yang Kung
Department of Management Information System, National Pingtung University of Science and Technology
kung@mail.npust.edu.tw

Abstract

The evolution of information technology has been developed in recent years. The Google CEO proposed a topic of cloud computing which described the ubiquitous services over the internet at Search Engine Strategies conference in 2006. Cloud computing can support high performance computing, distributed storage and scalable services to provide services to users at the same time. In addition, virtualization technology is the key of cloud computing development, easy to deployment and dynamic resource allocation. Cloud computing has gradually become the mainstream of information technology industry.

In this paper, we present a middleware of cloud services with load balance mechanism. The middleware supports flexibility of web service via heterogeneous platforms. Developers can register and deploy web services to the cloud environment. We use the queuing theory to calculate the blocking rate of service request via each *virtual machine* (VM) in cloud

environment and design the dynamic load balance mechanism to reduce load imbalance. Moreover, we use *rich internet application* (RIA) and *User Interface Markup Language* (UIML) for user interaction to promote system flexibility.
Keywords: Cloud Computing, Load Balance, Web Service, Middleware, Queueing Theory