行政院國家科學委員會專題研究計畫　期中進度報告

下一世代超大型網路虛擬環境平台之研究(2/3)

計畫類別： 個別型計畫
計畫編號： NSC94-2213-E-009-026-
執行期間： 94 年 08 月 01 日至 95 年 07 月 31 日
執行單位： 國立交通大學資訊科學學系(所)

計畫主持人： 袁賢銘

計畫參與人員： 蕭存喻、彭品勻、宋牧奇

報告類型： 精簡報告
報告附件： 出席國際會議研究心得報告及發表論文
處理方式： 本計畫可公開查詢

中　華　民　國 95 年 5 月 27 日

摘要

隨著網際網路的日益發達，從九〇年代末期，虛擬世界的平台開發與發展也日益澎湃，尤其在寬頻普及的亞太地區，虛擬世界的其中一個應用：「線上遊戲」也成為亞太地區網際網路使用者最樂於付費的應用，而一個虛擬環境的平台及應用，不僅僅可運用在娛樂之上，更可運用在ｅ化教學，甚至是數位典藏方面的加值。而在行動通訊日益普及的年代，手持無線裝置的能力也逐漸增強，甚至已能部份取代傳統個人電腦才能執行的虛擬環境用戶端的能力。

本研究計畫的目地在於設計一個具高延展性、高可用性、高效能並整合行動裝置的虛擬環境平台，主要的訴求將有：易於開發、易於佈署、易於管理。我們將利用本實驗室在以往計畫中所累積的高效能通訊中介軟體技術、行動通訊及代理人技術、可調適顯示技術，來完成此一虛擬環境的開發平台，並在上面選擇一個應用來開發。最後我們也會將此一平台跟應用對外開放，量測並統計相關的數據，來証明此一平台的可行性。

關鍵字：線上遊戲、中介軟體、虛擬環境平台

Abstract

With the growth of Internet, the research and development of virtual environment and platform becomes popular. Massively Multiplayer Online Games (MMOGs), one of popular the application of virtual environments, is one of the most successful business model of game industry and .com business now. A virtual environment platform not only can use in entertainment but also can be used as add-value of e-learning and digital achieve. Moreover, mobile devices are getting popular nowadays, and it can be used as a virtual world client execution environment.

As the result, we plan to build a low-cost and efficient platform solution with mobile support for virtual environment developers. Our overall architecture is based on middleware technologies. Our platform solution will focus on three elements: ease of development, ease of deployment, ease of maintain. We will make use of our previous research, such as middleware, mobile computing, mobile agent, and adaptive display technologies, to build the virtual environment platform.

Our research on this project will consist of four elements. We had already survived many of the virtual environment and server design. Our first aim will be the development of architecture and framework for scalable and efficient virtual environment platform. The framework can help our platform programmer programs a distributed virtual world easily. Second, we will develop a virtual environment logic object process platform with mobile/thin-client support. This agent platform can help end user design their game logic objects, and then deploy and management these virtual player easily. Next, we will develop an efficient persistent storage manager system for fault-torelent. Last, we will develop a simple virtual world, such as virtual shopping mall or virtual museum, to demonstrate our platform. We will also like to join the workshop to discuss our ideas with other parties having similar interests.

Keywords: Massively Multiplayer Online-Game, Middleware, Virtual World Platform.

第一章、前言

MMOG(Massively Multiplayer Online Game)是一種可以支援上千人同時在一個虛擬遊戲世界互動的遊戲類型。然而，開發一個 MMOG 的難度遠超過我們所想像，很多議題都是遊戲開發者可能會遭遇到的，例如高效能的網路、分散式的技術、負載平衡等。同時，要開發一個具有競爭力的遊戲，Time-to-Market 也是十分重要的考量。因此，MMOG 中介軟體解決方案的需求大量增加，而 MMOG 中介軟體的研究也逐漸受到重視。

因此在本計畫中，我們將開發一個 MMOG Middleware，並將其中的經驗與其他人分享。一方面，我們試圖解決開發 MMOG 所會遇到的共同問題。另一方面，我們嘗試的簡化設計遊戲的 API 以減低開發 MMOG 的難度。除此之外，我們也保留彈性給開發者以部署各種不同類型的遊戲在平台之上。本報告在第二章闡述此研究的目標，第三章說明系統的分析與設計，第四章將進行結果討論，第五章會作個總結與說明未來展望及藍圖。

第二章、研究目的

MMOG (Massively Multiplayer Online Game)是一種提供多人同時在一個虛擬遊戲世界互動的遊戲類型。但是開發一個 MMOG 的難度是非常高的，開發者會遭遇到問題，例如高效能的網路、分散式的技術、延展性、容錯機制、負載平衡…等，而這些問題大多是一般遊戲所不會遭遇到的問題，因此開發一個 MMOG 是非常耗費時間與人力的。本計畫中第一年已經成功的研發的 DOIT (Distributed Organized Information Terra)的主要核心中介軟體(包括有高效能網路引擎、永續資料存取元件)，以及一個簡易的示範程式，証明了 DOIT 是一個可用的平台。我們在第二年的研究重點則著重在：

1. 提昇網路引擎的效能。
2. 為了增加平台架構擴充性(extensibility)，我們設計了一個 Plugins 的軟體框架
3. MMOG 是一種服務，因此管理以及監控的支援也相形重要，因此我們必需要設計一個容易開發管理、監控的平台環境。
4. 在平台上開發一個簡易的示範程式，証明 DOIT 平台是一個可用的平台。

第三章、分析與設計

3.1. 平台的擴充框架設計

理想上，中介軟體層必需要提供內容開發者不同層次的 API，比方說，平台必需要提供伺服器層次(server scope)的 API，讓開發者可以使用系統計時器、執行緒池，而有另外的地域性層次(region scope)的 API，可以直接存取虛擬世界中的物件。而 DOIT 中則實現了這個想法，我們提供了一個伺服器層次的元件並與整個伺服器平台共生，而地域性層次則與虛擬世界共生。而地域性層次的元件可以與伺服器層次的元件相依，反之則不行。圖 1 則是這個概念的一個系統元件層次關係圖。

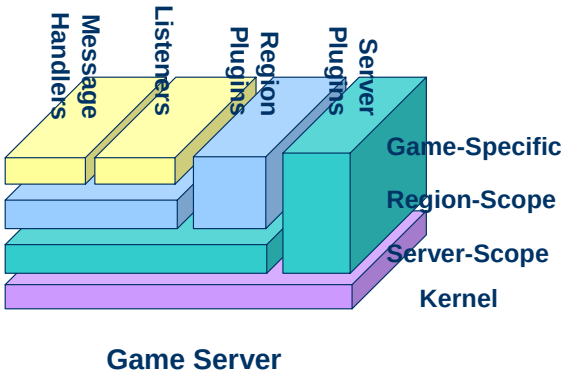


圖 1. DOIT 伺服端端主要元件之間的關聯示意圖

DOIT Platform 允許透過 plugin 的方式來延伸平台的功能，寫成 plugin 的好處是可以取得較低階的資訊。在 DOIT 平台是採取 component-based 的 architecture，元件可能是 server scope component(繼承 mmog.server.ServerScopeComponent)或是 region scope component(繼承 mmog.server.RegionScopeComponent)。前者跟 Server 生命週期一樣，一個 server 就只會有一個 instance；後者跟 Region 的生命週期一樣，所以如果 server 上有三個 region，就會產生三個 instance，並它會隨著 region migration 資料跟著 migrate 過去。而 plugin 事實上就跟 component 一樣，只是透過特殊的部署方式去安裝 plugin。

開發 Plugin 的開發步驟如下：

1. 撰寫 Plugin
2. 編譯及包裝成 jar 檔
3. 部署

如前面所說有兩種 Plugin：Server Scope Plugin 及 Region Scope Plugin。這兩個 Plugin 分別要繼承 ServerScopeComponent 及 RegionScopeComponent。以下是一個範例：

```
//MyPlugin.java
package hello;
import mmog.server.*;
import java.util.Hashtable;

public class MyPlugin extends ServerScopeComponent{
    public void init(ServerContext context,
                     Hashtable prop){
        System.out.println("hello");
        System.out.println("foo=" + prop.get("foo"));
        System.out.println("foo2=" + prop.get("foo2"));
    }
}
```

編譯所要注意的跟開發 game logic 一樣，主要是要把 mmog2.jar 放在 classpath 裡，以下是一個範例。

```
javac -cp mmog2.jar -d . MyPlugin.java
```

要部署到 DOIT 平台，我們必須把所有的 class 檔包成 jar 檔。我們可以用 JDK 裡面的 jar 這個執行檔來包裝，以下是一個範例

jar -cvf MyPlugin.jar hello/*

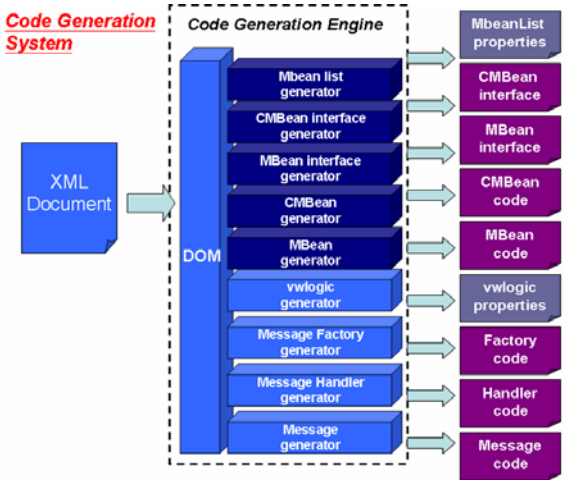
要部署 plugin，我們要把剛剛產生的 jar 檔放在<DOIT_HOME>/plugin 的目錄之下，同時我們還要提供一個同名的 properties file 來提供此 plugin 的一些資訊(plugin 的 class 檔以及初始化參數)，以下為一個範例：

```
#MyPlugin.properties
#所有的 server plugin 都要以 mmog.plugin.server 爲前置字元
#所有的 region plugin 都要以 mmog.plugin.region 爲前置字元
#再此定一個 name 爲 test 的 plugin 並指定 class name
mmog.plugin.server.test = hello.MyPlugin
#給 test 這個 plugin 一些參數
mmog.plugin.server.test.foo = bar
mmog.plugin.server.test.foo2 = bar2
```

3.2. DOIT management framework：

為了要讓整個 MMOG 環境裡所有的伺服器都能夠被監控，我們使用 JMX (J2EE 延伸管理技術) 所提出的架構來實作 MMOG 管理系統，整個系統是透過一個中央管理伺服器來進行管理工作，在整個管理系統上的管理元件可以被動態的加入或是移除，並且可以使用 MMOG 開發系統來開發管理元件，管理人員可以自行開發使用者管理介面程式或是透過瀏覽器來連結到中央的管理伺服器。關於 JMX 架構請參考
http://java.sun.com/products/JavaManagement/index.jsp

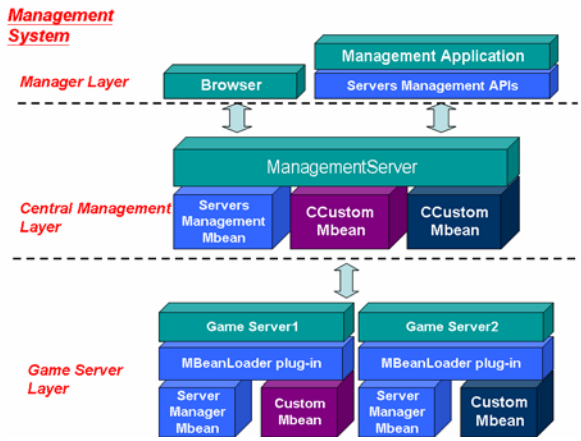
下圖是整個 Development System 元件組成關係圖，整個開發系統包含在 mmogsdk package 底下。



- CodeGenerationEngine

這是整個開發系統核心的部份，負責做 xml 文件解析以及將所得到的內容丟給適當的 code generator，由不同的 generator 負責不同 code 的產生工作。下圖是整個 Management System 元件組成關係圖，分成三個部分，分別是 Game Server Layer、Central Management Layer 及 Manager Layer，整個管理系統包含在 mmog_management package 底下。Game Server layer 是由不同的遊戲伺服器所組成，再每壹台遊戲伺服器上面會包含一個 mmog management plug in 的元件 (MbeanLoader) 及一個伺服器管理者元件 (ServerManagerMbean)，及其他由使用者自訂的管理元件(CustomMbean)。Central Management Layer 包含了一個中央管理伺服器(ManagementServer)及一個 mmog 伺服器管

理元件(ServersManagementMbean)及其他使用者自訂的中央管理元件(CCustomMbean)。Manager Layer 定義了一組中央伺服器管理 APIs，開發人員可以使用它來開發管理元介面程式，或是她們可以透過 ManagementServer 預設的 http 連接阜，透過 web 瀏覽器來進行管理工作。下



我們並把這個機制，加入到我們第一年中所提出來的快速開發工具之中，接下來我們將介紹關於在 MMOG development system 上的開發方式，包含描述文件的格式及 CodeGenerationEngine 的使用方法。

首先，我們必需先要撰寫 MMOG.xml，MMOG.xml 是一份 MMOG 描述文件，他描述了所有 Game 相關的訊息以及使用者自訂的管理元件內容，下面是一個範例的部分內容:

```
<?xml version="1.0" encoding="utf-8" ?>
- <MMOG_Message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="mmog_message.xsd">
    <Version>1.0</Version>
    <PackageName>mmog.message</PackageName>
    <Classpath>../MMOG_development_system/</Classpath>
  - <Messages>
    - <Message>
        <MessageName>LoginMessage</MessageName>
        <MessageType>0x01</MessageType>
      - <Params>
        - <Param>
            <ParamName>id</ParamName>
            <ParamType>long</ParamType>
          </Param>
        - <Param>
            <ParamName>pass</ParamName>
            <ParamType>String</ParamType>
          </Param>
        </Params>
      </Message>
    </Messages>
  + <MBeans>
  </MMOG_Message>
```

在這份文件裡面包含兩種資料一種是關於 codegeneration 必要的資料，例如 version 、PackageName、Classpath 等，另一種資料則是開發人員所要產生的訊息及管理元件的描述資料，另外我們也定義了一份 mmog_message.xsd 來進行文件驗證的工作。在範例裡面，我們首先描述了這份文件的根元素<MMOG_Message>並定義了 xml schema 文件，接著是<version>、<PackageName>、<Classpath>這些標籤，分別記錄著這份文件的版本資訊、及所產生出來的程式碼所屬封裝以及程式碼路徑位置，接著進行訊息及管理元件的描述，所有的訊息都是包含在<Messages>這個標籤裏面，每一個訊息再以一個<Message>標籤當作是根元素，接著分別描述訊息名稱、型態以及所屬的變數。

例如在範例中我們描述了一份名稱為 LoginMessage 的訊息，他的型態是 0x01，他擁有兩

個變數分別為長整數型態的 id 變數以及字串型態的 pass 變數。因此所有的訊息都是以相同的方式來進行描述，在 DOIT MMOG PLATFORM 裡所支援的訊息變數型態有 short、boolean、long、byte、float、int 及 String，開發人員可以根據所定義的訊息內容來決定合適的變數型態。另一部份關於 mbean 的描述方式如下圖所示

```
<?xml version="1.0" encoding="utf-8" ?>
<MMOG_Message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="mmog_message.xsd">
  <Version>1.0</Version>
  <PackageName>mmog.message</PackageName>
  <Classpath>../MMOG_development_system/</Classpath>
+ <Messages>
- <MBeans>
  - <MBean>
      <MBeanName>PlayerManager</MBeanName>
      <ServerName>server1</ServerName>
      <MBeanIp>localhost</MBeanIp>
      <MBeanPort>9998</MBeanPort>
    - <MBeanParams>
      - <MBeanParam>
          <MBeanParamName>regionnumber</MBeanParamName>
          <MBeanParamType>Integer</MBeanParamType>
          <Authority>w</Authority>
        </MBeanParam>
      - <MBeanParam>
          <MBeanParamName>regioncount</MBeanParamName>
          <MBeanParamType>Integer</MBeanParamType>
          <Authority>r</Authority>
        </MBeanParam>
      </MBeanParams>
    - <MBeanMethods>
      - <MBeanMethod>
          <MBeanMethodName>Debug</MBeanMethodName>
          <ReturnParamType>void</ReturnParamType>
        - <Attributes>
          - <Attribute>
              <AttributeName>debug</AttributeName>
              <AttributeType>boolean</AttributeType>
            </Attribute>
          </Attributes>
        </MBeanMethod>
      </MBeanMethods>
    </MBean>
  </MBeans>
</MMOG_Message>
```

所有的 Mbean 都是包含在 MBeans 標籤底下，每一個 Mbean 則是以<MBean>為根元素，描述了下面這些資訊包含 MBean name、所屬的 Game Server Name、所屬的 Mbean Server IP 和 port 以及所包含的變數名稱、型態及權限(read or write) 以及其他 method 內容。由於在 JMX 定義之下的 Mbean 是由許多的 getAttribute、setAttribute 及其他 method 所組成，因此我們可以藉由定義一些變數及其權限來產生這些 get 及 set methods 另外再透過一些 method name、type 及傳入變數的描述，我們可以產生其他的 methods，例如這份文件定義了一個名稱為 PlayerManager 的 Mbean，他所屬的 Game Server 為 server1，IP 位址為 localhost,port 為 9998,這個 Mbean 包含兩個變數分別是型態為 Integer 的 regionnumber 以及形態為 Integer 的 regioncount，他們的權限分別為 write 和 read，因此我們會在 Mbean 裡產生 setregionnumber 及 getregioncount 這兩個 method,另外這份文件還描述了一個額外的 method，名稱為 Debug 回傳型態 void 而傳入的變數為一個名稱為 debug 的 boolean 變數，因此我們會在 Mbean 裡產生一個 void Debug(Boolean debug)的 method。透過這樣描述方式所產生出來的 CustomMbean 及 CCustomMbean 還必須透過開發人員進行 CustomMbean 內部 method 內容的實作之後才可以把它們 deploy 到 GameServer 及中央管理伺服器之上。(細節以及範例可以參照本期中報告中的論文集)

第四章、結果與討論

截至目前為止，本計畫的第二年目標可以說大部份都已經完成，DOIT 已經具備了一個 MMOG Service platform 應該有的絕大部份重要元件。同時我們在今年仍舊針對底層的 network engine 做出調校，並使用較新的機器來進行測試，得到了不錯的成果，估計大約只要 10 台 P4 等級的一般機器(6 台 Gateway,4 台 Server)就可以撐起一萬個用戶端的連線。不過在測試中也發現,主要的校能瓶頸點落在 Gateway 上,以訊息量來計算的話,大約一台 gateway 每秒可以處理 6000 個訊息為上限，超過的話校能就會開始大幅下降。

依照我們目前的平台開發環境，遊戲開發者只需依照遊戲內容的不同，設計不同的 Message Protocol 與處理遊戲邏輯的 Handler，就可讓一群玩家在一個虛擬遊戲世界中進行遊戲。進而透過各種 Channel 與 NPC 行為的實做，設計出更豐富互動與內容的遊戲。而在本年度中所增加的 Plugins、Management Framework 也使得 DOIT 平台更具實用性及擴充性。

第五章、計畫成果自評

截至目前為止，本計畫的第二年目標可以說大部份都已經完成，DOIT 已經具備了一個 MMOG Service platform 應該有的絕大部份重要元件。而目前我們也跟雷爵工作取得合作關係，正準備將一個實際的產品在我們平台上重新實作出來，相信在第三年的部份可以對平台的效能以及架構，進行實戰測試並蒐集相關的資訊，藉以來評估 DOIT 平台的實用度。

目前本計畫在論文的產出方面，第二年的成果如下:

國際期刊論文:

1. Tsun-Yu Hsiao, Shyan-Ming Yuan, "**A Scalable and Flexible Management Framework for Distributed Computing System**", WSEAS Transactions on Computers **(EI)** Volume 5, Issue 6, June 2006, pp. 1162-1168.
2. Tsun-Yu Hsiao, Shyan-Ming Yuan, "**Practical Middleware for Massively Multiplayer Online Games**", IEEE Internet Computing **(SCI)**, Volume 9, Issue 5, Sep/Oct 2005, pp.47-54

國際會議論文:

3. Tsun-Yu Hsiao, Ko-Hsu Su, Shyan-Ming Yuan , "**A Scalability Model for Managing Distributed-organized Internet Services**", The 5th WSEAS International Conference on Applied Computer Science (ACOS '06), Hangzhou, China, April 16-18. 2006

附件二

可供推廣之研發成果資料表

■ 可申請專利　■ 可技術移轉　　日期：95 年 5 月 27 日

國科會補助計畫	計畫名稱：下一世代超大型網路虛擬環境平台之研究(2/3) 計畫主持人：袁賢銘 計畫編號： 94－2213－E－009－026　學門領域：計算機系統結構
技術/創作名稱	DOIT 下一世代虛擬世界建造平台
發明人/創作人	袁賢銘、蕭存喻
技術說明	中文： DOIT Platform，也是源自於目前國際上線上遊戲的蓬勃發展，並在台灣掀起熱潮，但國內一直沒有足夠的系統軟體廠商可以協助傳統單機遊戲設計公司進入到線上遊戲的市場，造成國內有能力製作線上遊戲公司的數量一直無法提昇。以市場為例，不論中外，許多知名的線上遊戲開發時程往往需要三到五年的時間，其中一個主因就在於該公司必需要自行從網路底層到應用層的溝通一手包辦，花費公司太多成本及時間。DOIT Platform 的設計，除了讓遊戲開發廠商可以不用接觸到複雜的網路程式以及分散式軟體技術，更將多年來所累積的中介軟體技術(Middleware)運用在此一平台之上，將使得遊戲廠商可以容易開發(Development)、容易維護(Maintain)、容易佈屬(Deployment)，將其心力專注於該公司所擅長的數位內容的開發，將網路底層以及伺服器溝通的部份完全交由我們的 DOIT Platform 來解決，將有助於傳統遊戲設計公司快速地開發出所想要的線上遊戲。 DOIT 平台技術同時兼具了高延展性(Scalability)、高可用性(Availability)、高彈性(Flexibility)、簡易性(Simplicity)，並可供上萬人同時間在同一個世界中進行互動。並且可以達到動作型連線遊戲的低反應時間需求，只消耗些許的頻寬。使用 DOIT 平台將使得遊戲公司可設計下一世代的線上遊戲，而不再只是侷限在傳統的角色扮演型的線上遊戲。 （100~500 字）
	英文：
可利用之產業及可開發之產品	DOIT 平台將有助於國內自製的線上遊戲水準以及服務的提昇。線上遊戲的經營其實與經營一個網路服務無異，因此擁有一個穩固、可信賴、容易維護、低成本的後端平台，將有助於自製線上遊戲的經營者提供玩家更好的服務，才能讓遊戲廠商與玩家們同時受惠。因此 DOIT 平台將有助於提昇國內遊戲廠商的地位，也才能朝進軍

	國際市場邁進。 除了 PC、Console 上的線上遊戲都可以從 DOIT 平台受惠，在目前 Java 手機逐漸普及的年代，DOIT 平台更可提供各大行動電話業者建置與手持裝置相關的內容。因此對於有想要切入高互動性內容供應商的行動電話業者，也可以透過 DOIT 平台建置可讓用戶群進行即時互動的數位內容。
技術特點	整體特色： ● 提供一個可無限延伸、具容錯系統的遊戲世界。 ● 強大的伺服器管理、動態設定、自我調適功能。 ● 可自訂封包格式（可防駭）且高效能的網路通訊引擎。 ● 遊戲邏輯熱插拔(Hotswap)的功能，可讓線上遊戲不因小臭蟲而需關閉服務做修正。 ● 高效能的訊息導向中介軟體技術，可使廠商開發下一代高互動性產品。 ● 簡易並具彈性的伺服器端程式開發流程及 API 介面。 ● 可彈性地設計管理程式 ● 平台只需 PC-based 伺服器機種即可執行。 ● 減少開發網路遊戲所需的時間及成本。 ● 未來可與手持無線裝置緊密結合，開發行動裝置上的線上遊戲。
推廣及運用的價值	DOIT 平台將有助於國內自製的線上遊戲水準以及服務的提昇。

※ 1.每項研發成果請填寫一式二份，一份隨成果報告送繳本會，一份送　貴單位研發成果推廣單位（如技術移轉中心）。

※ **2.本項研發成果若尚未申請專利，請勿揭露可申請專利之主要內容。**

※ 3.本表若不敷使用，請自行影印使用。