出國報告（ 出國類別：實習 ）

興達電廠複循環四號機機組之
DCDAS 升級國外實習

服務機關：台灣電力公司

姓名職稱：紀伯翰 儀電工程師

派赴國家：德國

出國期間：108 年 08 月 24 日～108 年 09 月 06 日

報告日期：108 年 10 月 4 日

QP－08－00　F04

行政院及所屬各機關出國報告提要

出國報告名稱：**興達電廠複循環四號機機組之 DCDAS 升級國外實習**

頁數 26 含附件：□是■否

出國計畫主辦機關/聯絡人/電話

台灣電力公司/陳德隆/02-23667685

出國人員姓名/服務機關/單位/職稱/電話

紀伯翰/台灣電力公司/興達發電廠/儀電工程師/07-6912811#3703

出國類別：□1 考察□2 進修□3 研究■4 實習□5 其他

出國期間：108.08.24～108.09.06 出國地區：德國

報告日期：108.10.4

分類號/目

關鍵詞：西門子電廠自動化系統(SIEMENS T3000)、應用伺服器(Application Server)、Web 容器(Tomcats)、爪哇程式語言(Java Program Language)、運行容器 (Runtime container)、自動化伺服器(Automation Server)、嵌入式整合服務技術(Embedded components Services)

內容摘要：（二百至三百字）

興達複循環機組因 TME 控制系統已超過使用年限且已無備品,為確保機組維護品質及運轉安全，於 101 年開始逐漸將全部五部複循環機組 DCDAS TME 控制系統更新為 Siemens T3000。

新一代 SIEMENS T3000 Application Server 是由 Apache 網頁伺服器與 Tomcats Web 容器為基本組成，這部分由第三章第一節來介紹說明。

此外，為了實現 T3000 嵌入式整合服務技術(Embedded components Services)。即一台監控電腦可操作(operation)、工程編輯(engineering)、警報 (alarms)、檔案(archive)與診斷(diagnostics)的功能。西門子用 Java 程式開發了一系列的不同的運行容器(runtime container)在應用伺服器與自動化伺服器，每一個容器各有其工作與相關對應操作機制，最後還有應用伺服器容器重新啟動程序。這些部分將由第三章第二節討論。

本文電子檔已傳至出國報告資訊（http://open.nat.gov.tw/reportwork）

目　　錄

圖　目　錄

表 目 錄

第一章 研習目的

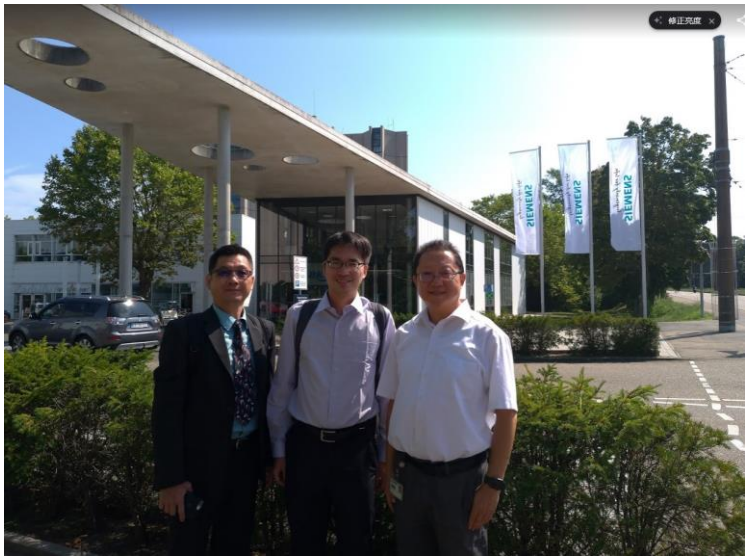
興達複循環機組因 Teleperm ME 控制系統已超過使用年限且原廠家已無生產備品，為確保機組維護品質及運轉安全，陸續將五部複循環機組 DCDAS 控制系統更新為 Siemens T3000。目前本廠複一至複五機已全部更新成 T3000 完成。藉由複五機還尚未大修 core-engine 的空檔，至西門子公司實習，期許自己能了解目前新型 T3000 application server 系統架構與程式設計原理。

此次實習主要赴位於德國柏林西門子訓練中心，接受安排訓練課程，除熟悉 SPPA-T3000 操作維護技巧外，由講師特別安排 Siemens T3000 application server 架構的課程，使能充分了解西門子針對伺服器這方面在程式設計原理。此外再介紹 application server 上每一個 runtime container 之功能與啟動程序。讓學員在 application server 的維護及檢修上能具備應有的觀念。

第二章 研習行程

日期	實習內容
08/24 ~ 08/25	桃園--法蘭克福
08/26~08/30	參觀西門子模組製造工廠 Siemens T3000 系統架構、硬體介紹 參觀西門子遠端專家中心
08/31~09/04	Siemens T3000 Application Server 架構介紹 Siemens T3000 Application/Automation Server container 之功能 Siemens T3000 Application Server 啟動程序
09/05~09/06	柏林---法蘭克福---高雄

現場操作實習



西門子柏林公司合影

第三章 研習內容

第一節 Java網頁伺服器介紹

興達複循環機組 Teleperm ME 控制系統陸續更新成 T3000 系統，而這 T3000 伺服器系統在同仁維護與討論時，僅能表達此網頁伺服器由 Java 程式撰寫成，但鮮少能真正探討到其內部組成。因此利用這次實習機會來進一步了解 T3000 伺服器的製作架構，首先 T3000 伺服器以網頁伺服器為出發點，用無互動的靜態網頁與有互動的動態網頁作為區別。接著探討 J2SE、J2ME、J2EE 開發的差異。最後西門子開發人員以 J2EE Jakarta JDK 來實現 T3000 伺服器。

壹、 靜態與動態網頁伺服器

目前現今比較常聽到的網頁伺服器為微軟的 IIS 與 Apache 軟體基金會的 Apache，當完成安裝網頁伺服器軟體後，即可實現一個靜態網頁，此時我們可以透過 Web browser(如 IE、Chrome、Firefox....etc)由 client 端連線至網頁伺服器得到一個純文字與圖片的網頁。但此網頁沒有任何的互動，僅需要 HTML、CSS、JavaScript 語言即可實現。若要達成動態網頁，則需要網頁伺服器透過程式與資料庫作連結產生的頁面，才可做資料的操作及更改，達到動態互動的效果。

而市面上常提及前端是指靜態網頁部分，程式語法用 HTML、CSS、JavaScript。後端是指動態網頁部分，在這動態網頁我們常提及使用的程式語言如 PHP、ASP.net、Python、Ruby 與 Java 等等程式。上述的程式語言除了 Java 外幾乎使用 CGI(common gateway interface)的機制讓 client 端可以呼叫位於伺服器的程式。當將網站應用程式撰寫在 CGI Program 後，client 端用 Web browser 送出 HTML 表單請求給網站，網站接收該請求後將資料透過 CGI gateway 轉傳給 CGI Program 處理，之後至資料庫存取資料，最終結果以 HTML 的格式回應 Web browser(參考圖 1 流程)。而 CGI 的特色為執行期間，會以獨立的程序來執行，所以每一個 client 端請求，皆會產生一個新的程序(process)，因此會導致回應慢、用較多資源與擴展性較差。

而 Java 程式原本不支援網站應用程式，但開始有支援 HTTP 請求後

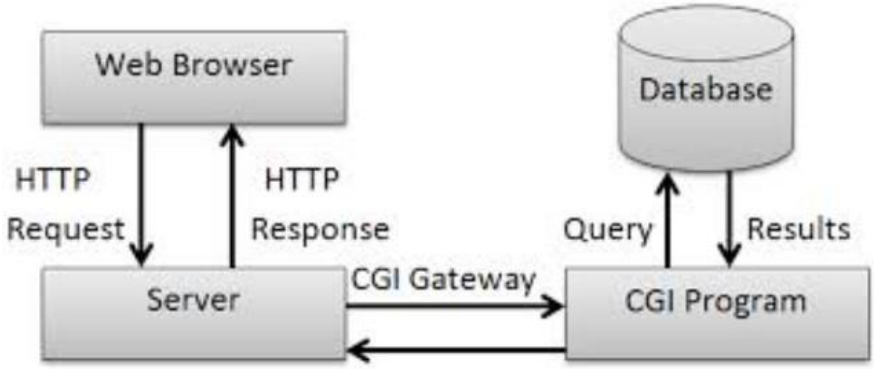


圖 1 CGI 程式處理 request 和 response 的流程

出現了 Servlet(英文意思為 server 端的小程式)的技術，該技術提供了一個框架使用 Java 處理 HTTP 請求產生 HTML 格式回應 Web browser，其執行類似 CGI，但架構有些差異。像 Servlet 只存在 Web container(為一個獨立程序與服務存在)，當有 client 端請求出現，Web container 會建立一個執行緒(thread)去執行該 servlet。因為是單一程序(process)裡可以跑多個執行緒(thread)，因為效能會比 CGI 程式來的快。不過 servlet 處理 HTML 格式回應 Web browser 的部分比較無法提供複雜且優雅的使用者介面設計，後來才又出現了 Java Server Page(JSP)以提高其能力，簡單來說 Servlet 檔名為.java，使用 java 程式內嵌 HTML 語法。Java Server Page 檔名為.jsp，在 html 文件內嵌 Java 語法。如果以圖二流程來看，比較像似 client 端用 Web browser 送出 HTML 表單請求給網站，網站接收該請求後將透過 Servlet 處理，之後至資料庫存取資料，最終結果用 Java Server Page(JSP)以 HTML 的格式回應 Web browser。

而市面上常用與目前 T3000 使用的 Web container 則是 Tomcat。它是由 Apache 軟體基金開發出的一個 Web container，實現了 Servlet 與 Java Server Page(JSP)的支援。在 T3000 online help guide 寫著 The Apache Tomcat provides servlet engine for the SPPA-T3000 system，其中的 servlet engine 即為 Web container。在第三章第二節討論 T3000 各項 Runtime container 功能介紹與操作機制說明就會看到很多 servlet 或 container 的名稱，到時候會把每一個 container 詳細介紹。所以明瞭到 T3000 Application server(由於複三機與複四機 T3000

系統為最新版本，架構亦相同，因此底下各 server 架構說明以複三機為例來敘述)的基本組成就是架了一個 Apache Http Server 2.4，之後又安裝了 Apache Tomcat 8.0，出現了 Web container，之後在安裝額外的 Runtime container。使我們能透過 thin-client 用 client container 方法連至 T3000 Application server 來取得伺服器資料與 Automation server 資料。

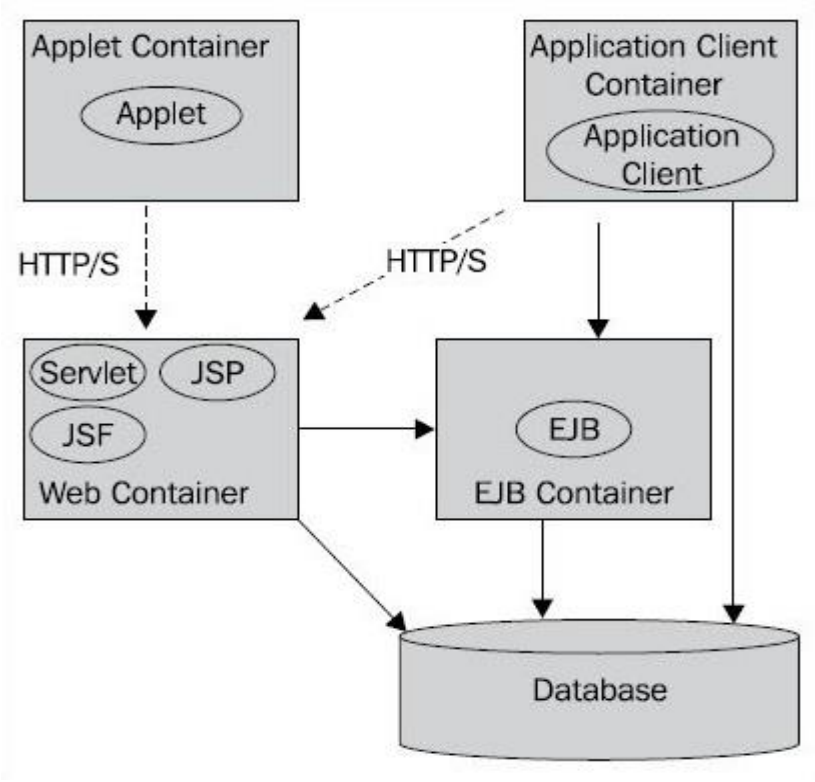


圖 2　Servlet 程式處理 request 和 response 的流程

貳、　Java EE 實現工業控制用 T3000 伺服器

提到 Java 時，大多會聯想到 Android APP 的開發，所以這邊介紹了 Sun Microsystem 提供了完整的 Java 技術產品，依據市場需求區分為三個總類，包含了 Java SE、Java ME 及 Java EE。

(1). Java SE:SE 表示為 Standard Edition，也就是 Java 標準版本，這個主要用來開發桌上型應用程式或者 Web browser 中內嵌 Applet 程式。

(2). Java ME:ME 表示為 Micro Edition，就是 Java 微型版本，用來開發手機、PDA、機上盒及嵌入式消費型電子裝置。當要開發 Android APP 就需要到 Java 網站安裝 Java ME Development Kit(JDK)，此外還有 Android 與 Software Development Kit(SDK)等，才能完成開發環境

(3). Java EE:EE 表示為 Enterprise Edition，也就是 Java 企業版本，以 Jave SE 為基礎，在架構上都與開發規模都比 Jave SE 龐大很多，應用技術上像 Servlet、EJB 比較常聽過，應用於大型、企業級網站開發用，像我們 T3000 就是使用 Jave EE Enterprise Edition 的 Jakarta 來開發完成。

既然剛剛提及 JDK 手機 APP 開發，那接著也把 JRE、JVM 這些常見的名詞也說明一下。

(1).JDK:DK 表示為 Development Kit，為一個程式開發的工具箱，提供 compiler 將 Java 語言編譯為低階語言的 byte code 外，還有作為開發、執行、測試。

(2).JRE: RE 表示為 Runtime Environment，為一個程式執行的環境，比如興達電廠網頁發電量為 Java Applet 開發，我們透過 Web browser 僅看到灰白畫面，無法觀看，就表示沒有安裝此 JRE，可去 Java 網站下載。而我們的 T3000 每台 Thin-client 也是需要安裝 JRE，但因為 T3000 為封閉式網路，無法對外連結，所以我們都會去它的 Application Server 網站點選 Open Work bench 來下載 workbench client container 與 JRE 檔(參考圖 3)。當下載成功後在桌面產生一個 workbench 的捷

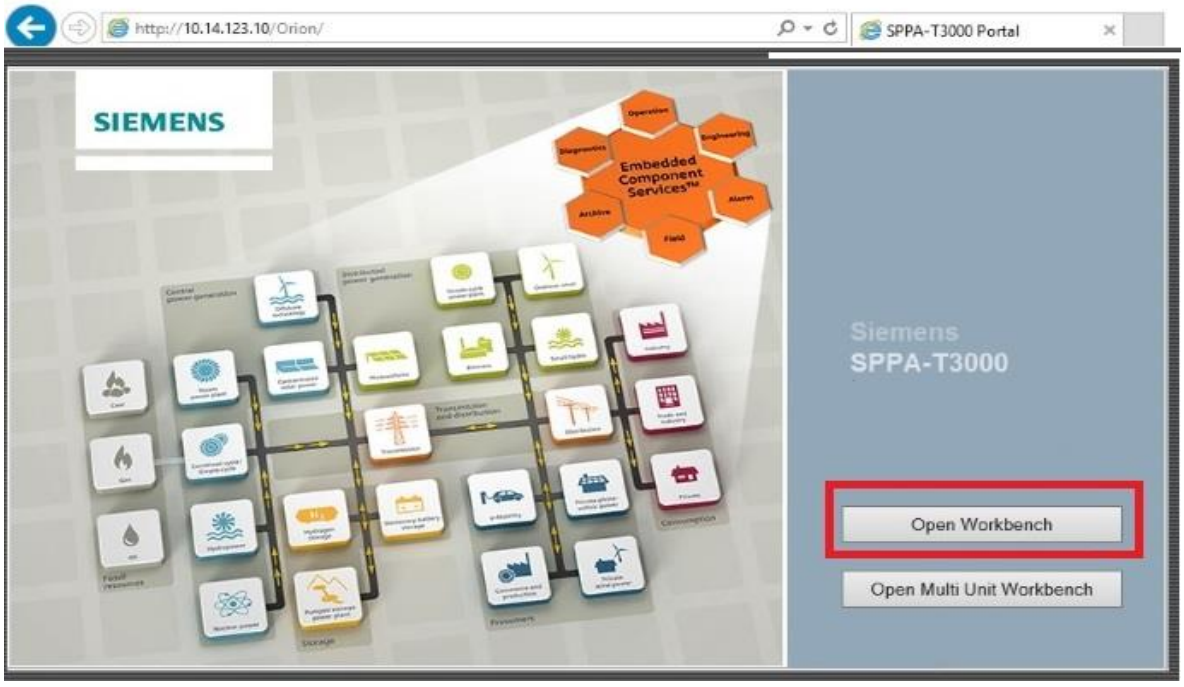


圖 3　連至 Application Server 網頁伺服器的首頁

徑，往後點選捷徑所走的連結方式參考圖二即由 Client Container (thin-client)連至 Web Container(T3000 Tomcat)。而我們從 thin-client 的 Java control Panel 找到 Java Runtime Environment 為 JRE v1.8 版(參考圖 4)。

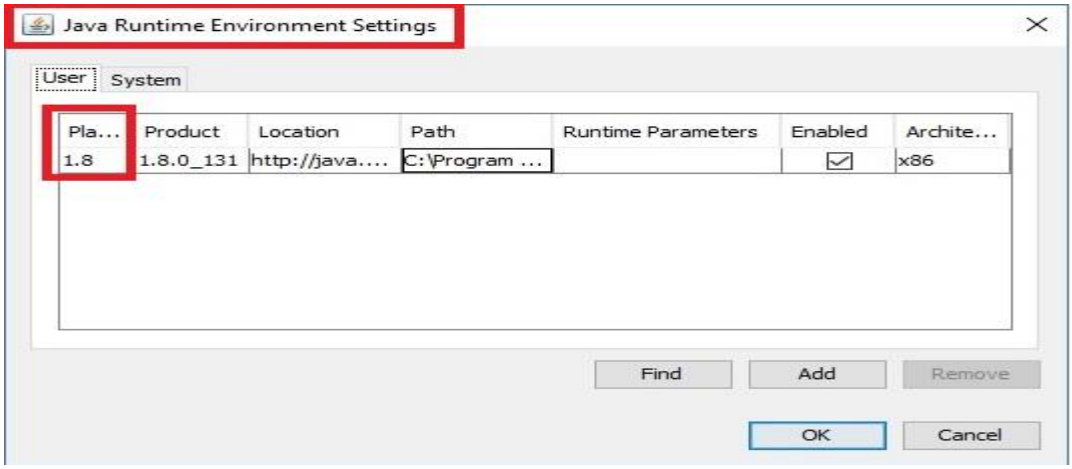


圖 4　複三機 thin-client 目前 JRE 的版本

(3).JVM:VM 表示為 Virtual Machine，已被包在 JRE 中，其目的讓 Java 程式在不同設備上執行，如 Linux、windows 等在有 VM 下均可執行。

圖五表示 Java SE 基本架構，可以看出 JDK 涵蓋了 JRE 與 JVM。JRE 涵蓋了 JVM，所以一般僅要有安裝 JRE 且版本正確，基本上都可以執行用 Java 開發出的程式，相當方便。

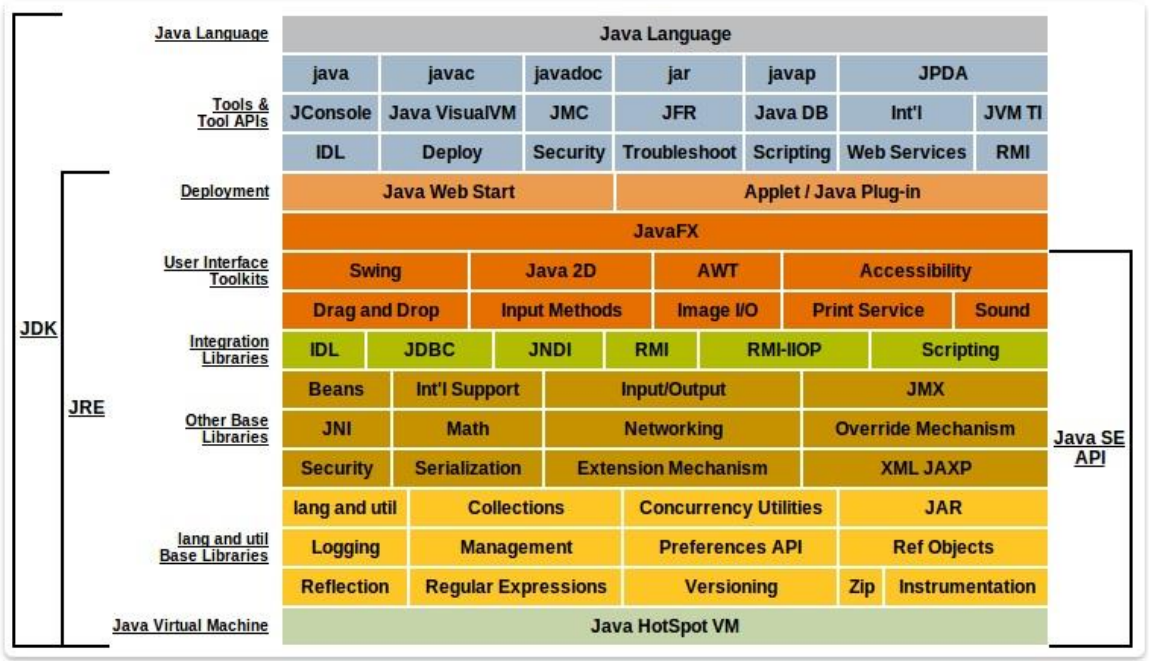


圖 5　JDK、JRE 及 JVM 涵蓋的範圍說明

第二節 T3000應用伺服器介紹

從第一節中，我們了解到透過架設 Apache 網頁伺服器與安裝 Tomcats Web 容器後，再安裝資料庫，即可達成一般企業級網站基本功能。但 T3000 是給工業控制用，需要 application server 與其他很多 automation sever 做連結，這時候就需要在 application server 與 automation sever 開發很多 container，所以這 container 扮演的很重要要角色，在這節中將介紹各 runtime container 所在位置與負責工作。

壹、 T3000 各項 container 功能介紹

我們先看 application server container 部分，第一個要提到是 project container，顧名思義，他就是專案容器，一個資料管理中心，掌管所有的容器資料，所以在 project view 所看到的程式與圖控畫面資料夾都與 project container 有所關聯(參考下圖 6)，專案架構、程式編輯、資料管理與權限管理都跟它有關。以圖六 Session 為例，它會因我們帳號 view、operator 與 engineer 我們能監控或操作畫面就有所差異

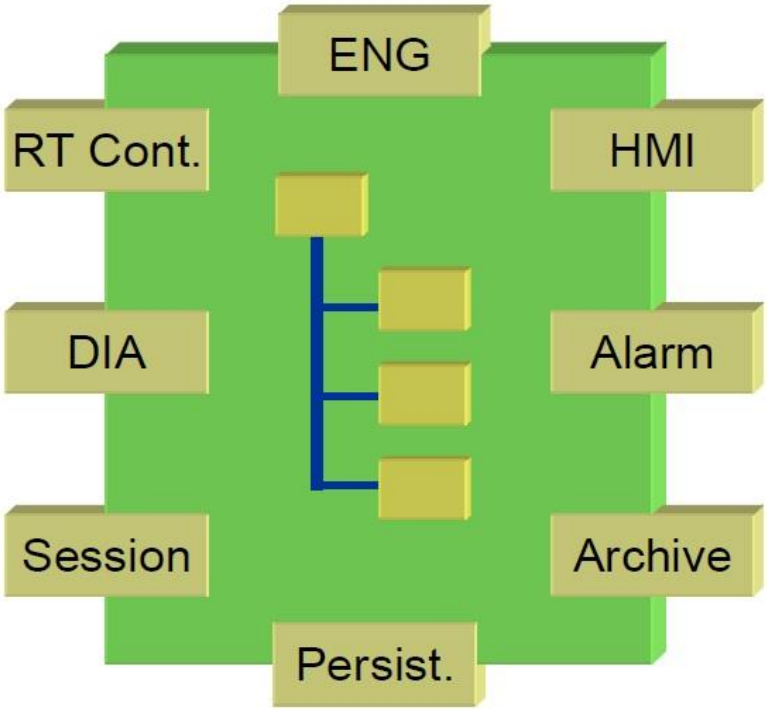


圖 6 Project Container 的功能方塊圖

像 view 僅能觀看，無法更改程式(及 function diagram 會反白)與無法看診斷畫面(Diagnostic view 會反白)，這些都由 Session 判斷權限作切換。此外，此容器啟動程序第一步要啟動這一項，若這項無法正常運轉，其他操作都無法正常操作。

第二重要的則是 monitor container，為系統管理核心之一，最主要功能為監控所有的容器，當容器有所故障時，連帶著會將訊息連結至 diagnostic container。

以上是 application server container 中最重要的兩項，所以 server 重新啟動時，這兩項順序為第一優先順序，一定要先啟動。之後的如 Simatic communication container(cc)、Application_Function (Appl_Funct)、Diagnostic Container(dc)、Alarm Container(ac)、Plant Display Server(pds)、Archive Container(arc)、Report Container(rc)、Trend Display Server(tds)、CFC Chart Container(cfc)及 Simatic hardware config container(hw)這十項相關負責工作介紹如下表格 1 所示，每個 container 都有其功用，可以由容器名稱大概看得出其功能。

	容器名稱	啟動順序	負責工作
1	Project Container(pc)	**0**	專案容器:為一個資料管理中心，包含了專案資料、資料儲存 、安排與管理應用，此容器若無運轉，整個 T3000 應用伺服器系統無法正常運作。
2	Monitor Container(mc1)	**0**	監控容器:為一個系統管理最主要的核心之一，主要功能在監控其他的容器，當有一個監控故障點發生時，連帶著診斷容器也可以偵測到。
3	Simatic Com Container(cc)	**1**	simatic 通訊容器:為一個取得所有 AS S-7 動態製成數值(dynamic process values)容器，此容器沒運轉，則無法做程式畫面監控與更改
4	Application_Function (Appl_Funct)	**1**	應用函式容器:可在上面寫控制程式與測試程式的 container。目前很多 EOH 計算程式放這裡。
5	Diagnostic Container(dc)	**2**	診斷容器:為一個強大的診斷功能的容器，可以診斷各容器的運轉狀態、自動伺服器容器一些儀資設備哪裡有錯誤或通訊中斷與路由器、thin-client 連線狀態等等…藉由此容器可以監控到所有設備狀態正常與否。

6	Alarm Container(ac)	**3**	警報容器:為一個專門負責管理警報的容器，當沒運轉下，接連會影響到 ASD 的顯示，及圖控不會有警報出示。
7	Plant Display Server(pds)	**4**	圖控顯示伺服器:為一個專門負責管理操作圖控畫面的 servlet(後改名 server)，此沒運轉，則 thin-client 無法操作圖控畫面。
8	Archive Container(arc)	**5**	檔案容器:為一個專門負責管理檔案的容器並與資料庫連結儲存，當沒運轉下，接連會影響報表顯示與歷史趨勢圖。
9	Report Container(rc)	**6**	報表容器:為一個專門負責管理報表的容器，此沒運轉，則 thin-client 無法抓報表資料。
10	Trend Display Server(tds)	**7**	趨勢顯示伺服器:為一個專門管理趨勢顯示畫面的 servlet，此沒運轉，則 thin-client 無法看歷史趨勢圖。
11	CFC Chart Container(cfc)	**14**	圖表容器:為一個專門負責 Fail-safe 函式的圖表容器，此沒運轉，則 thin-client 無法看 Fail-safe 函式 function diagram。
12	Simatic hardware config container(hw)	**15**	硬體規劃容器: 為一個專門負責 s7 cpu 硬體規劃容器，此沒運轉，則無法做 s7 cpu 或 io 硬體規劃

表 1 Application Server 上 Container 的啟動順序其與負責工作

接著要談的是 automation server runtime container，這是要放在 S7 或 AS3000 CPU 內程式中，下圖七為 automation server runtime container 的功能方塊圖，此 runtime container 用來執行程式與管理程式。在這容器中新增函式(圖七的藍色圖形)與 IO Proxy 點的連結(圖七的白色圖形)、執行順序的管理與程式的反應時間(圖七的 EXEC)，此外還有作外部連結，如連至其他的 runtime container(peer to peer)、連至圖控的 faceplates(圖七的 HMI)與連至警報(圖七的 ASD)。

目前以複三機為例，使用的 automation server runtime container 共有 19 個，屬於 S7 CPU 有 8 個，用於 ST 汽機控制與保護有兩個 GT31/32/33/HRB31/ HRB32/ HRB33 有 6 個，裡面的 runtime container 又可以分一般的 container 與 FS_container，FS_container 主要用於 Fail-safe 的黃色 IO 卡片，程式也屬於 Fail-safe。另外用於 AS3000 CPU 的共有 11 個，如 GT 輔助設備、HRB 的輔助電動閥、WaterSteam(WS)、廠內 4160/480V 斷路器、泵室相關控制及電機方

面控制。此 AS3000 CPU 相較 S7-CPU 較為陽春，其改硬體與程式方便，但相對可靠度會比 S7 差一點。

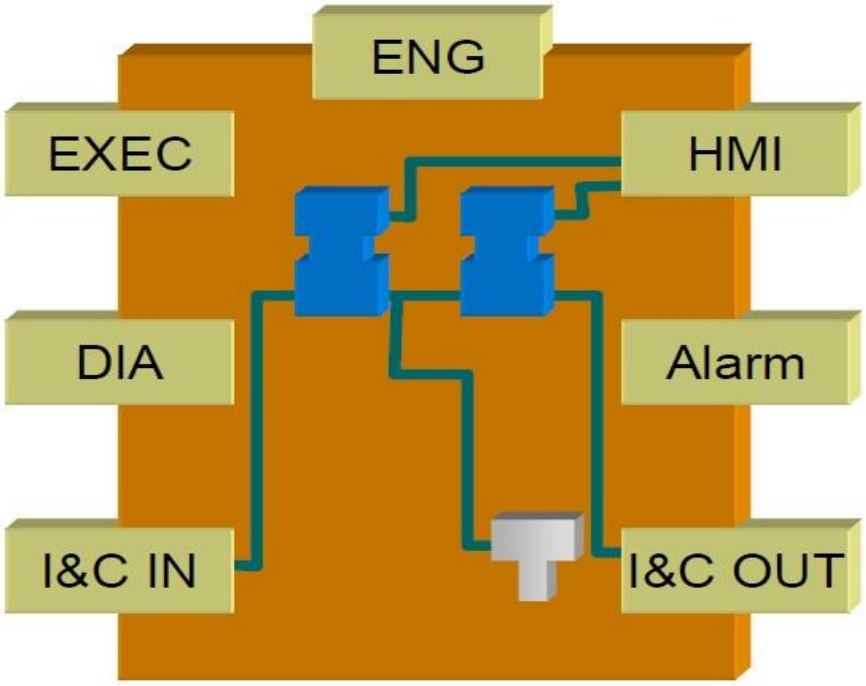


圖 7　Automation Server Runtime Container 的功能方塊圖

	容器名稱	啟動順序	負責系統	負責工作
1	ST30_Ctrl_S7 ST30_Ctrl_F_S7	**0**	ST	1. S7-410 的 CPU 2. 負責 ST 機組控制用
2	ST30_Prot_S7 ST30_Prot_F_S7	**0**	ST	1. S7-410 的 CPU 2. 負責 ST 機組保護跳脫用
3	GT31/32/33_S7 GT31_F/32_F/33_F_S7	**0**	GT	1. S7-410 的 CPU 2. 負責 GT 機組控制與保護跳脫用
4	GT_AUX30_AS3000	**0**	GT	1. AS3000 的 CPU 2. 負責如 GT 主變、減壓站、緊急柴油機相關訊號用
5	HRB31/32/33_S7 HRB31_F/32_F/33_F_S7	**0**	HRB	1. S7-410 的 CPU 2. 負責 HRB 控制與保護跳脫用
6	HRB31/32/33_AS3000	**0**	HRB	1. AS3000 的 CPU 2. 負責 HRB 相關輔助電動閥用

7	BoP_IO1_AS3000	**0**	Water Steam	1. AS3000 的 CPU 2.負責除氧器等相關訊號
8	BoP_IO2_AS3000	**0**	Water Steam	1. AS3000 的 CPU 2.負責冷凝水泵 CP 及相關訊號
9	BoP_UnitCtrl_AS3000	**0**	Water Steam	1. AS3000 的 CPU 2.負責高壓飼水泵及 UnitControl 訊號
10	Common_AS3000	**0**	Common	1. AS3000 的 CPU 2.負責 CWP 及泵室相關訊號
11	Electrical_AS3000	**0**	Common Electric	1. AS3000 的 CPU 2.負責廠內 4160/480V 斷路器用
12	30cs01/02_CS3000	**0**	Common Electric	1. CS3000 的 CPU 2.僅監控高壓馬達一些設備狀態用

表 2 Automation Server 上 Container 的啟動順序與其負責工作

貳、 T3000 操作機制說明

上節介紹了各項容器，這節介紹 thin-client 是如何做這個交易(transaction)，如我們怎麼透過 thin-client 監控整個機組圖控、怎麼透過 thin-client 更改或觀看機組控制程式、怎麼透過 thin-client 診斷機組運轉狀態。

(1) 監控機組圖控(Plant Display)操作機制:圖 8 為監控機制的流程圖，我們可以由 thin-client 點擊圖控畫面的同時，其實它是透過 web browser 產生一個請求(Request)至網頁伺服器(web server)，接著網頁伺服器資料傳遞至 Plant display servlet(即 plant display server)直接到我們要看的圖控畫面的 automation server runtime container 抓取資料，再回傳至 Plant display servlet，再到網頁伺服器做回應(Response)給 thin-client 做圖控畫面的顯示。這其實跟我們一般互動式網站由 client 請求至 server，server 抓取資料做回應其實是大同小異，最大的差別在於 application server 上的 Plant display servlet 需至 automation server runtime container 抓取資料。這部分較複雜些。

(2) 搜尋程式(Function diagram)流程圖: 圖 9 為搜尋程式的流程圖，它是透過 web browser 產生一個請求(Request)至網頁伺服器(web server)，接著網頁伺服器資料傳遞至 Engineering Servlet 再至 Project

Container 到 simatic manager 連至 automation server runtime container 抓取資料，再回傳至 Engineering Servlet，再到網頁伺服器做回應

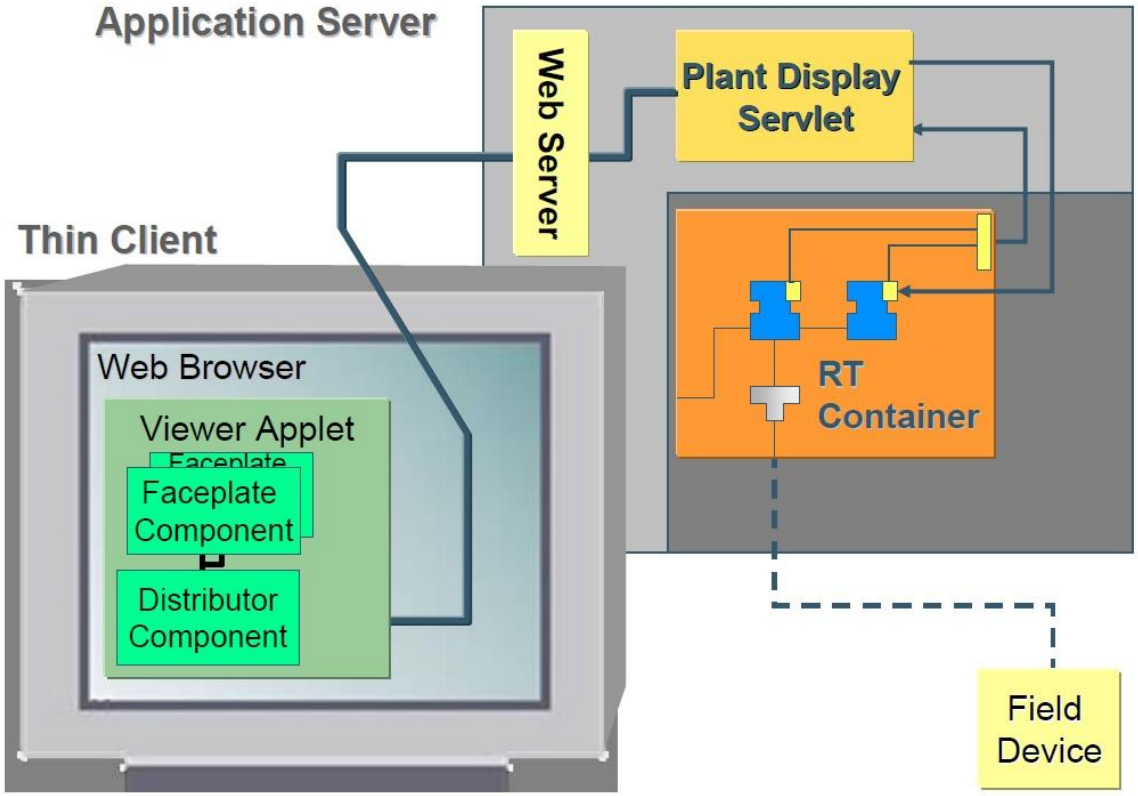


圖 8　Plant Display 監控機制流程圖

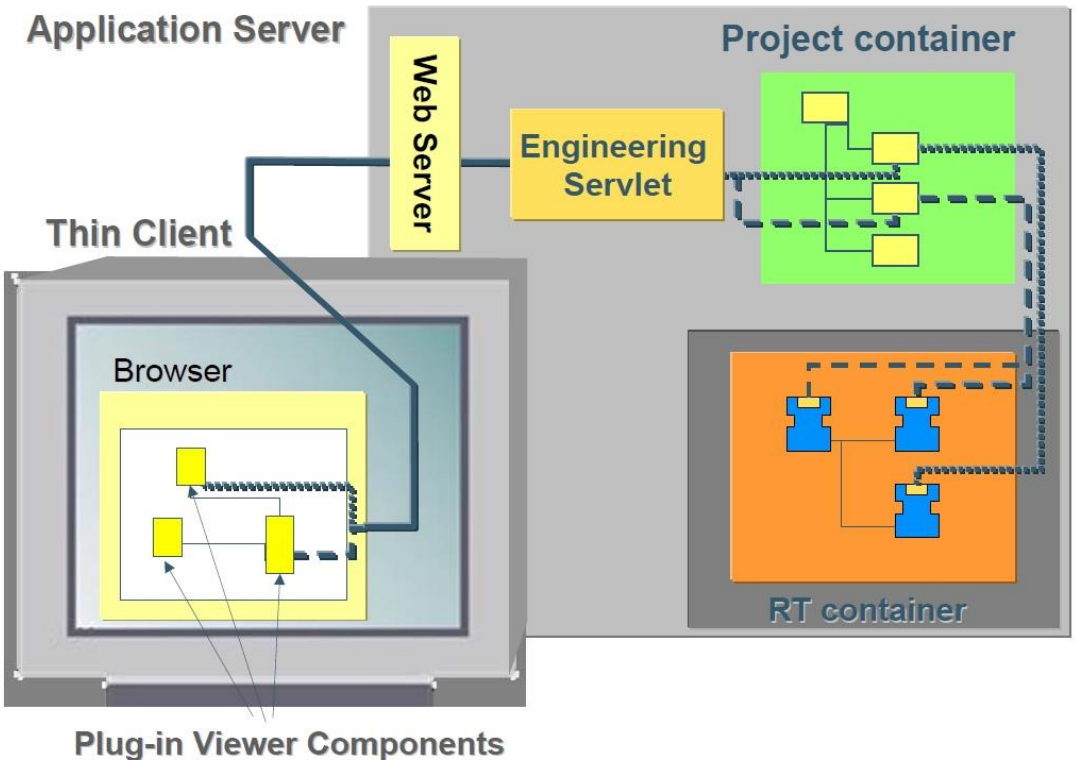


圖 9　Function diagram 搜尋程式流程圖

(Response)給 thin-client 做圖控畫面的顯示。

上述部分僅查程式部分，若要到更改程式則需要以點選 function diagram in operation 切換至 configuration mode，接著開始修改參數或者新增刪除程式，再更改完成後，可以按 commit(如圖 10 所示)，此時程式已在 diagram edit 做修改動作成功，但在 Project container 與 automation runtime container 卻還沒更改。

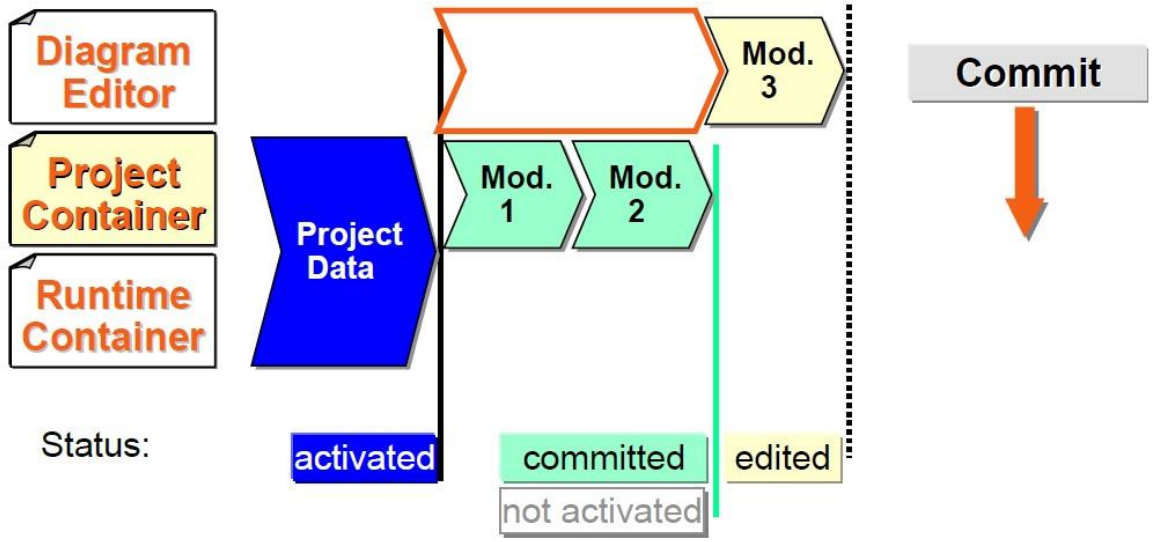


圖 10　Diagram Editor 更改程式按下 Commit 時的狀態

呈上，如果發現我們所改程式好像改錯了，那時候可按 Rollback(如圖 10 所示)，它主要功能為恢復至原本最初程式狀態，所以原本在 diagram edit 更改程式的部分，全部都會不見，此時又可以重新開始。

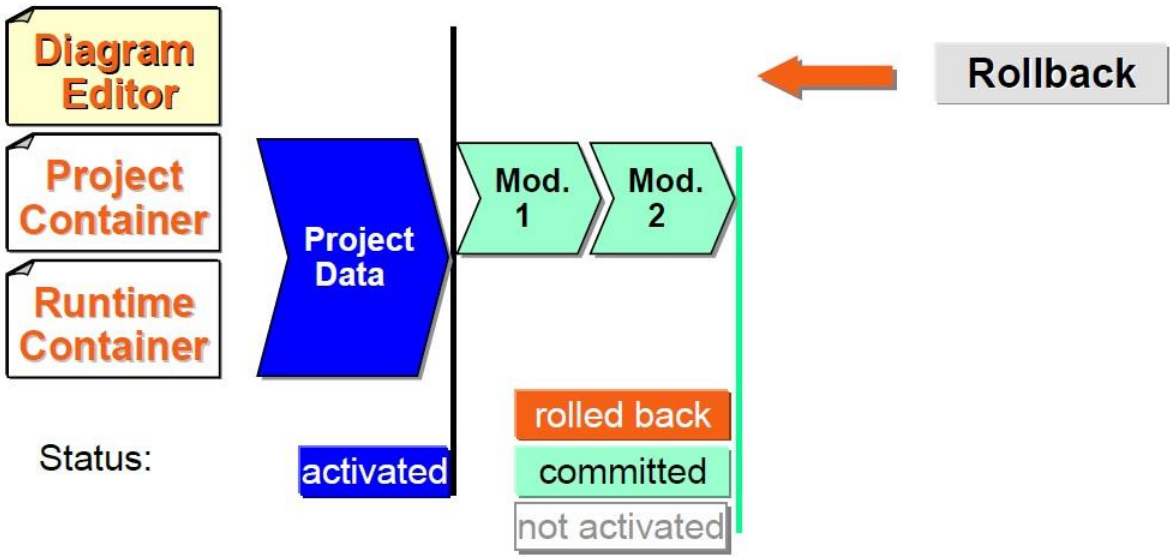


圖 11　Diagram Editor 更改程式按下 Rollback 時的狀態

當我們又再次改完程式，確認都沒問題，就可以按 commit，此時 diagram edit 部分程式已更改，再按下 Activate，即 Project container 與 Automation runtime container 整個程式皆已修改完成。

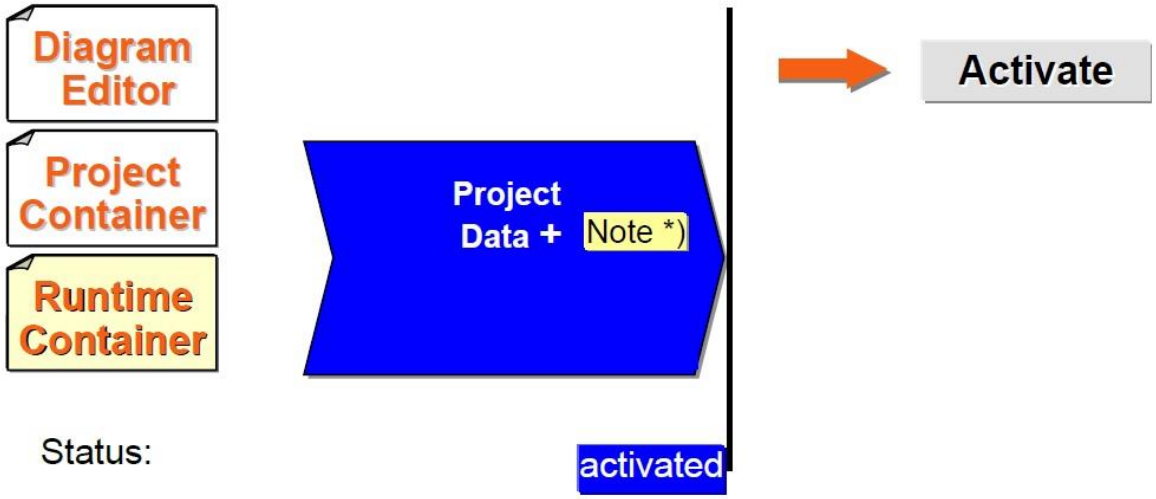


圖 12　Diagram Editor 更改程式按下 Activate 時的狀態

(3) 程式(Function diagram)繼承功能:物件導向程式三個概念為封裝(Encapsulation)、繼承(Inheritance)與多型(Polymorphism)。而 Java 就是個物件導向程式，所謂的封裝，簡單來講就是把程式包起來(涵蓋了屬性與方法)並加上公有與私有參數，我們以建立一個人作為類別(class)為例，人本身基本的屬性像身高、體重、儲蓄的金額這些就是屬性。而人的一些能力，像會講話、會跳、會開車等這些就是方法。而公私有屬性或方法我們可以加以區別。像我們儲蓄的金額可能就是私有，只有自己可以取得，他人不可取得。而其他則是公有，大家都可以取得，封裝的概念大致如此。繼承則是比較像父傳子，所以子類別繼承父類別，此外還可以重新定義某些屬性或方法。圖 13 則是在敘述 Java 程式開發語言上使用物件導向的繼承功能,從我們由 function diagram 命名 kks 開始,之後由 function diagram 加上 automation function，發現 kks 的名稱是跟連動的，至最後的 signal connect，訊號名稱(即 kks)都繼承至 function diagram。所以再新增程式與其他 io 或 peer to peer 連結時候，使用的名稱不用一直更改，因為物件導向程式幫我們繼承，讓我們新增或修改程式變得很方便，不用花大量時間寫 kks 描述(description)。

接著則是多型，多型此中文名稱很像有多個型態，以 overloading 來說我們人可以吃東西，但可以吃一個或多個東西，像 Eat(int rice)或 Eat(int rice, int noodle)，就是一個只有吃米的函式方法跟一個吃米跟麵的函式方法，而這兩個函式都叫做 Eat，但可以產生多個不同的函式方法這稱為 overloading。此外可能還會看到類似 print(int a)、print(double b)、print(char c)(在 console 列印出來)，這就是屬於多型的動態繫結，它可參考你的輸入型態(如 int,double,char)去決定要執行的函式方法。這兩部分簡單來講就是多型的概念。

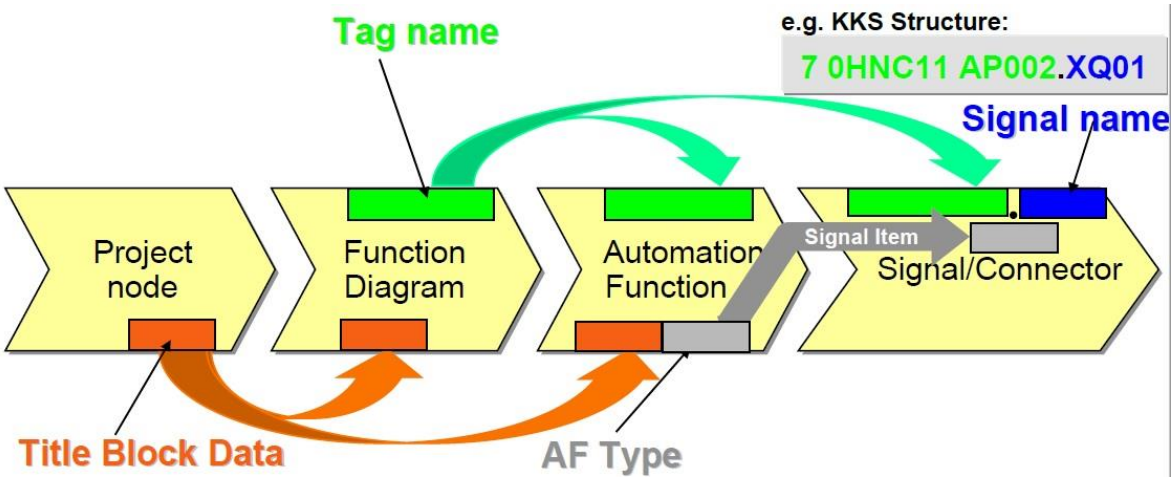


圖 13　由 Function diagram 繼承至 Signal/Connector

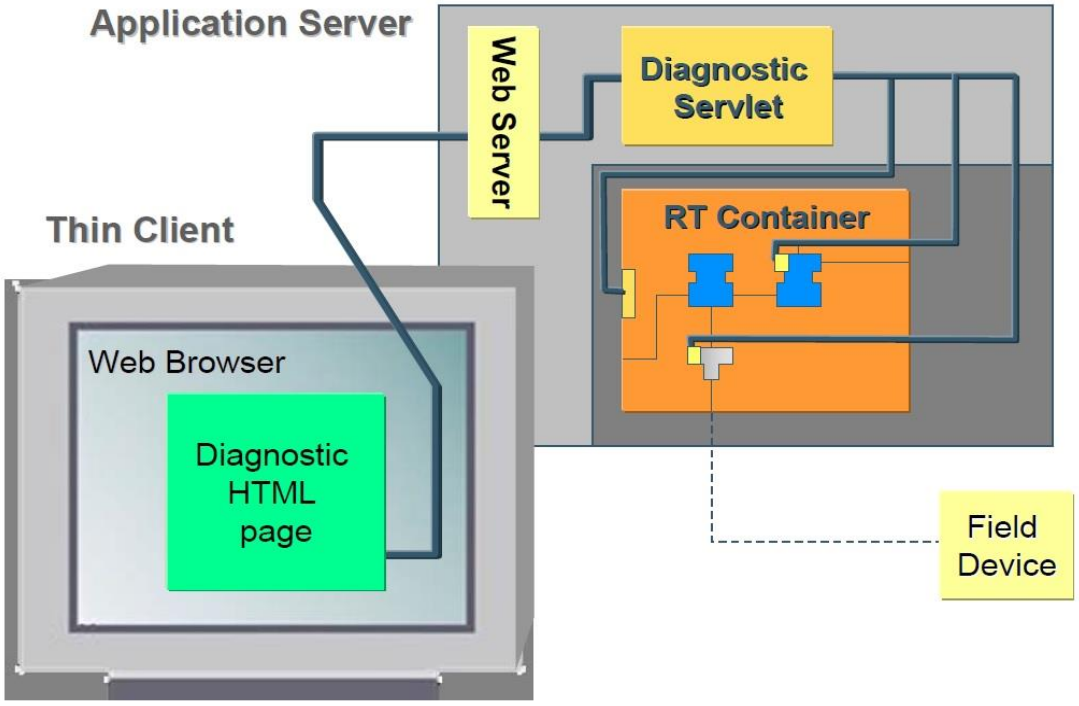


圖 14　Diagnostic view 診斷機制流程圖

(4) 診斷(Diagnostic view)機制流程圖:診斷其目的為診斷各容器的運轉狀態、自動伺服器容器與 scalance、thin-client 連線狀態等等…藉由此容器可以監控到所有設備狀態正常與否。其操作機制為透過 web browser 產生一個請求(Request)至網頁伺服器(web server)，接著網頁伺服器資料傳遞至 Diagnostic servlet(即 Diagnostic container)直接到 application server 的 container 與 automation server runtime container 抓取資料，再回傳至 Diagnostic container，再到網頁伺服器做回應(Response)給 thin-client 做圖控畫面的顯示。

參、 T3000 應用伺服器容器啟動程序

應用伺服器容器啟動程序可以由應用伺服器開始的 AdminConsole 點

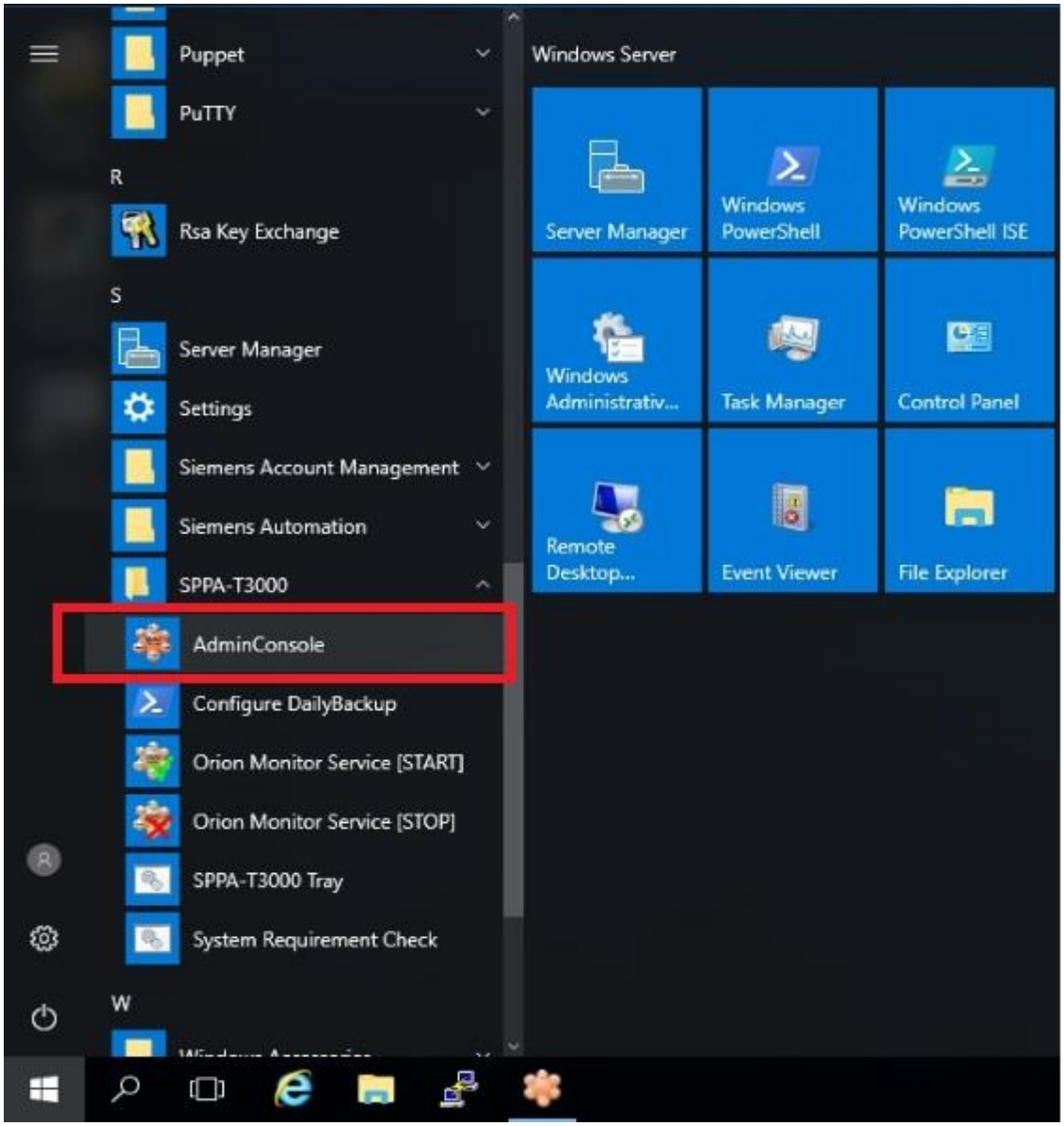


圖 15 AdminConsole 可查到 Runtime container 啟動程序

選執行(參考圖 15)，它的畫面類似 Diagnostic view，我們點選其 container，再看它的 start number 就可以看到啟動順序，基本上 T3000 應用伺服器重開後網頁伺服器 Apache Http Server 與 Web container Tomcat 會先執行完畢，再執行容器啟動，容器第一個啟動為 Project

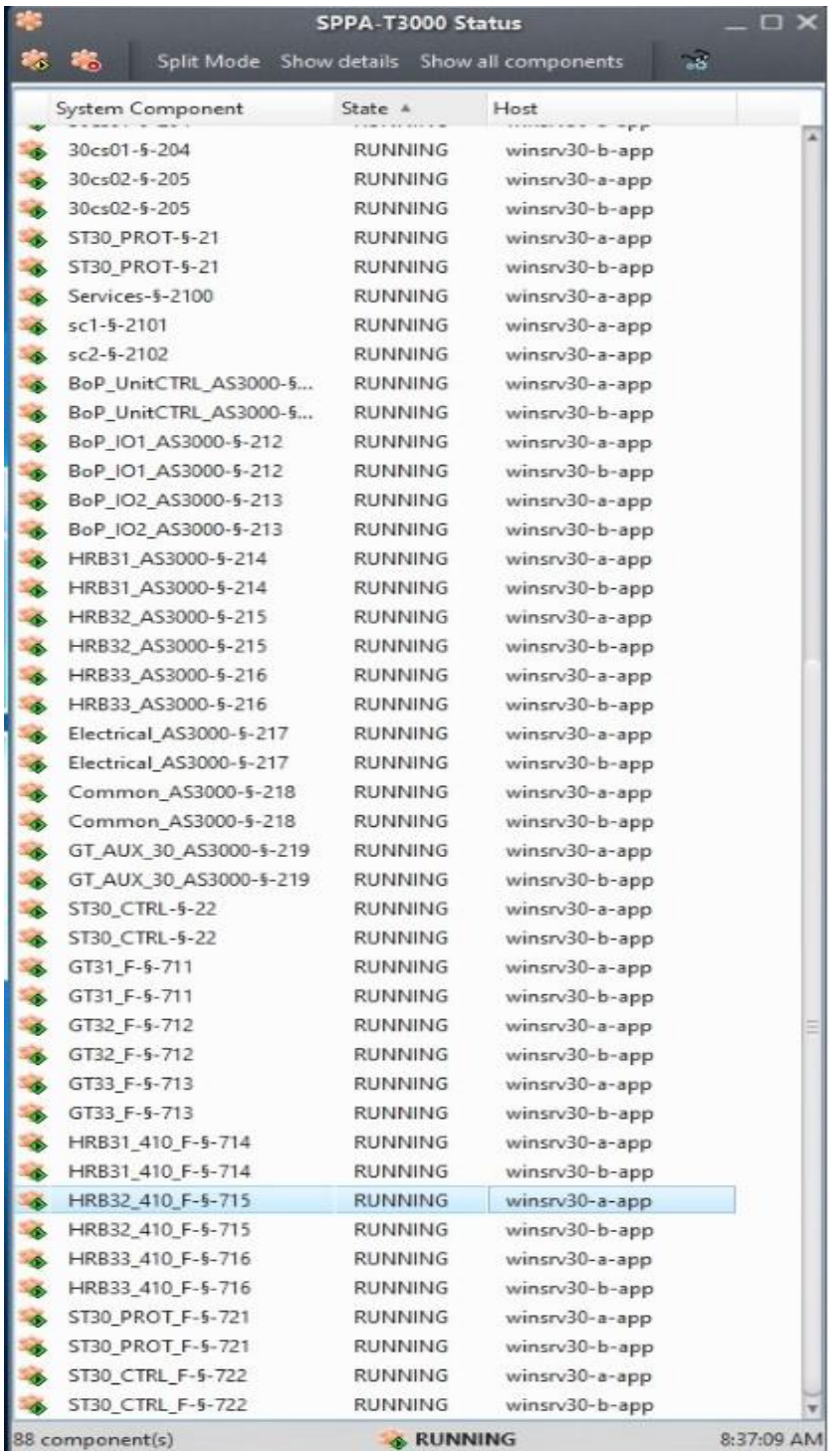


圖 16　SPPA-T3000 Status 可查到各 Runtime container 的狀態

container、Monitor container 與眾多的 automation server 的 runtime container。它 start number 設為 0(相關的啟動值可以參考表 1(page 14)與表 2(page 16))，表 1 呈現了每個應用伺服器容器的啟動數值，值最小由 0 開始，跑到最後的 15。表 2 呈現了每個自動伺服器容器的啟動數值，值由 0 開始一起並列執行。容器的狀態可有兩個地方可以查到，第一個地方為應用伺服器的 SPPA-T3000 Status，可參閱圖 16，但這部分需在應用伺服器才可以監控的到。若要在 thin-client 看到，則點選 Diagnostic view，那邊會有所有的 container 的狀態，正常為 running，若為 starting 表示啟動中，terminating 表示正常 shutdown 中，stopped 表示 shutdown 完成，出示 failed 表示有錯誤出示，此時可以再看相關的 report 與 event 來判別錯誤的原因。

第四章 心得與建議

此次實習最主要是了解應用伺服器的容器功能與操作原理，了解的過程中也發現其實我們複一至複五機因更新時間有所不同，造成一些硬體架構與容器會有些差異，以下是我這次出國實習的心得與建議。

- core-engine 更新伺服器:趁大修期間將複一至複五機將所有應用伺服器更新成同版本與同架構，方便後續維護。也趁西門子原廠工程師更新時候，好好從旁學習。
- 自動伺服器的容器: 這部分坊間書籍與網路幾乎找不到使用 Java 與 DCDAS 的底層程式溝通連結(僅圖控存取 DB 顯示與操作時有)，這部分也是需要研讀相關資料或再請教其他西門子原廠工程師。這樣才比較有能力處理一些 critical issue。
- 應用伺服器的模擬機架設: 這部分自己覺得蠻重要的，於西門子公司實習期間，西門子工程師也是架一台 2 對 1 機組並且測試與了解動作原理，這將是我接下來想繼續深入探索的目標。

誌謝

本人於 98 年通過經濟部特考，分發至澎湖尖山發電廠，負責維護電廠柴油機儀資設備相關工作。於 102 年因家庭因素，在時任台電總經理李漢申、尖山廠長陳清江、副座歐致誠、儀電經理王長進、儀資課長陳啟賢及工會常理蕭鉉鐘的大力幫忙與協助下，於 102 年 5 月調至興達發電廠儀二組並服務至今。

此次機會能赴德國原廠西門子公司實習，首先感謝各長官們的推薦及興達電廠楊廠長仁和、孫副廠長禹華、南火蔡副廠長松融、儀資二組鄭經理建業、儀資一組林經理坤泉(再三交代要去考語文證照)與賴俊雄課長、張志榮課長、蔡琮傑課長、陳宗賢課長的幫忙。再來感謝我出國的這兩週期間，複儀三課林佶立主辦、蘇金獅領班、李清相副領班、林譽謙師傅、蔡元竹師傅、郭殷宏師傅、峰名及心淳的幫忙處理本人的業務及機組的維護。

此外，在德國實習兩個禮拜中，感謝西門子吳明裕經理提供搭車、訂飯店與飲食等日常生活的幫忙。還特地安排假日帶著我參觀古蹟、古堡、教堂等，讓我更了解德國歷史文化。還有感謝同事許家銘剛好在德國參訪過程有相遇一週的時間，讓我生活不至於那麼枯燥乏味。

最後，感謝我的妻子、岳父及岳母，這段期間幫忙照顧我女兒，讓我能毫無掛念地在德國實習訓練，感謝。

參考資料

1. “Java 相關網頁開發與維基百科
2. “T3000 online help guide”
3. “SPPA-T3000 Basic Engineering & Operations PG L332 Training Center”
4. “SPPA-T3000 User Guide Diagnostics System”