

WFCMS – 晶圓流程管理系統

WFCMS – A Wafer Flow Control Management System

研 究 生：許朝坤

Student：Chao-Kun Hsu

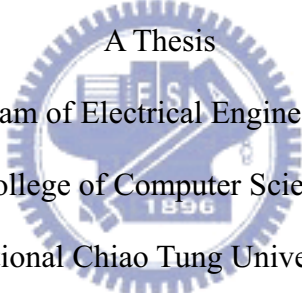
指 導 教 授：李素瑛 教授

Advisor：Dr. Suh-Yin Lee

國 立 交 通 大 學

電機學院與資訊學院專班 資訊學程

碩 士 論 文



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

Hsinchu, Taiwan, Republic of China

中華民國九十五年十月

晶圓流程管理系統

研究生：許朝坤

指導教授：李素瑛 教授

電機學院與資訊學院專班 資訊學程

摘要

近年來資料庫的技術雖有很大的進展，但資料庫的架構已從傳統的集中式轉換成分散式的架構，從實務面而言，仍缺少有效的系統開發方法。企業內部因部門分工需求，各部門自建所需的資料庫，提供該部門使用，未考慮到資料共享的需求，產生許多異質性的資料庫。如今，在全球化的競爭市場中，企業為求能夠掌握其所屬產業之前景與相關企業的脈動，對資訊整合與共用的需求日漸加重，企業內部異質性資料庫的整合，是當前企業內部資訊系統急需突破的課題。

本篇論文裡，配合企業內部資料庫規畫現狀的架構下，提出一套晶圓流程管理系統(Wafer Flow control Management System，簡稱WFCMS)，整合異質性資料系統。晶圓流程管理系統架構採三層式發展架構(Three-Tier Architecture)－客戶端(Client)、Web 伺服器(Web Server)，與資料庫伺服器(DB Server)。程式開發使用 JAVA 語言。客戶端動態網頁使用 JSP 開發。應用程式伺服器與資料庫伺服器連結使用 JDBC。客戶端(Client)與 Web 伺服器連結運用 JNDI 機制。為了減少伺服器與資料庫建立連線的次數，運用 Connection Pool 的技

術，避免程式與資料庫連結溝通頻繁，造成效能問題。

在系統效能評估上，分兩部分。第一部份根據各部門所要求更新及讀取資料的邏輯，撰寫測試事件。並請各部門根據此測試事件，測試晶圓流程管理系統(WFCMS)，評估更新及讀取資料的正確性。資料邏輯的正確性已達到 100%。第二部份以現行各部門使用未整合的應用程式介面，更改資料的時間與透過晶圓流程管理系統(WFCMS)，更改相同資料的時間做比較，評估工作效率。使用測試晶圓流程管理系統總共可節省 $41.3(\text{分鐘}) \times 120(\text{人數}) = 4956 \text{ 分鐘}(82.6 \text{ 小時})$ 。由以上測試反應時間比較結果，晶圓流程管理系統(WFCMS)確實替公司節省很多人力操作的時間。在更新各部門的資料庫方面，使用程式自動更新，確實避免許多人工輸入造成的錯誤。未來晶圓流程管理系統(WFCMS)能提供更自動化的功能，期望達到無人工廠的理想。

WFCMS – A Wafer Flow Control Management System

Student : Chao-Kun Hsu Advisor : Dr. Suh-Yin Lee

Degree Program of Electrical Engineering and Computer Science

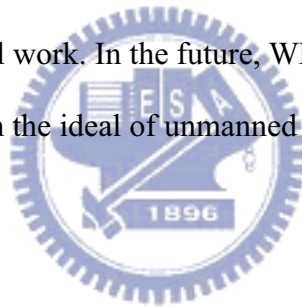
National Chiao Tung University

ABSTRACT

Although the technology of the database has very great progress in recent years , in practice, it still lacks an effective systematic development approach. The structure of the database system has already changed from traditional centralized type into the distributed structure of the dispersing heterogenous type. Because the specific system requirement of each individual department, the tasks inside the enterprises is divided and every department develops its own specific database for departmental use. There is no consideration of information sharing and exchange. Nowadays, in the globalized competitive business environment, in order to grasp the prospects of their affiliated industries and pulsation of relevant enterprises, the request for integrating and sharing information is aggravated day by day. The heterogeneous databases inside the enterprises needs to be integrated.

In the thesis, we design an integrated wafer flow control management system, abbreviated as WFCMS, which integrates the system of heterogeneity. The WFCMS adopts the Three-Tier Architecture Client, Web Server and DB Server. We use Java in the programming. We use Java Server Pages to develop the dynamic webpage in client side. Application server and database server are linked using JDBC. Client and Web server are linked using JNDI(Java Name and Directory Interface) mechanism. In order to reduce the number of times in the connection of server and database, the technology of Connection Pool is used in order to prevent the system from connecting with database frequently, for the efficiency.

In the assessment of the system, we consider two aspects : the effectiveness and efficiency. For the first part, we write testing cases according to the logic requirements of all departments in the enterprise. Then, ask every department to test WFCMS according to the testing cases, and we estimate the logical correctness of data updates and data access. The logical correctness has already been up to 100%. The second part, we calculate and compare the time which the engineers use the original application interfaces without integration and using WFCMS with integrated and updated data. We find that using WFCMS saves $41.3(\text{minute}) \times 120(\text{the number of people}) = 4956 \text{ minutes}(82.6 \text{ hours})$. We compare the result above, WFCMS really saves the time of the manpower who operates for the company. We use WFCMS updates data of every department automatically, which really prevent a lot of inputting mistakes caused due to manual work. In the future, WFCMS can offer more automatic functions, and expects to reach the ideal of unmanned factory.



誌 謝

能順利完成此篇論文，最要感謝指導老師李素瑛教授費心的指導，於受業間，李教授的耐心指正和諄諄教誨，亦讓我受益良多。

於專班求學過程中，特別感謝教授們的教導與鞭策。感謝在職班同學們的互相鼓勵與支持，特別感謝晉碩與貴榮的關心與協助。

此外，更有賴許多朋友的精神支持，甚或實質協助，我才能順利完成學業。例如介峰、紀承、宏杰等同學與學長，尚有一些未提及的朋友們，在此均一併致謝。

感謝，我可愛的女兒育淳與活潑的兒子士盛的陪伴，使壓力得以舒解。特別要感謝的是我母親，對我的養育之恩，在我人生的重要關鍵時刻對我的幫助與關懷。

最後要感謝我的摯愛，雅秋，從大學認識到現在全力地支持我，由於她的任勞任怨，平日除忙於上班工作外，更要辛勤操持家務與照顧家庭，讓我無後顧之憂，是我心靈上最大的支柱。

目 錄

摘要	i
ABSTRACT	iii
誌謝	v
目錄	vi
表目錄	viii
圖目錄	ix
第一章、緒論	1
1.1 前言	1
1.2 動機	2
1.3 研究目標	3
1.4 論文大綱	4
第二章、相關文獻探討	6
2.1 緣由.....	6
2.2 資料處理的系統架構	7
2.2.1 主從式系統(Client-Server System)	7
2.2.2 三層式應用系統發展架構	9
2.2.3 主從式系統與三層式系統的比較	10
2.3 客戶端與應用程式伺服器連結開發	11
2.3.1 可延伸性標示語言(XML)	11
2.3.2 應用程式開發-JAVA 語言	12
2.3.3 客戶端網頁開發-JSP	14
2.3.4 客戶端(Client)與應用程式伺服器連結的機制-JNDI	15
2.4 應用程式伺服器與資料庫伺服器連結	17
2.4.1 開放式資料庫連接界面(Open DataBase Connectivity)	17
2.4.2 JAVA 資料庫連結器(JDBC)	18
2.4.3 JDBC 與 ODBC 互相搭配	20
2.4.4 應用程式與資料庫連結的機制-connection pool	22
第三章 作業研究	24
3.1 晶圓(Wafer)	24
3.2 晶圓製造主要相關部門職掌	24
3.3 機台生產流程	29
3.3.1 機台擺設位置與工程師作業區	29
3.3.2 機台生產流程	30
第四章 整合各部門的資料庫系統	33
4.1 各部門的需求與各部門資料庫關係	33
4.2 晶圓流程控制系統(WFCMS)與各部門關係.....	35
4.2.1 整合各部門的資料庫系統	35
4.2.2 整合各部門功能系統	37
4.3 晶圓流程控制系統(WFCMS)與各部門人員作業模式	37
4.3.1 使用者登入	37

4.3.2 製造部門作業模式	39
4.3.3 製程部門作業模式	43
4.3.4 設備部門作業模式	44
第五章 系統實作	45
5.1 系統登入	46
5.2 共同查詢介面.....	47
5.2.1 站點查詢	48
5.2.3 站點流程查詢	49
5.2.4 機台資料查詢	50
5.2.5 晶圓資料查詢	52
5.2.6 離開	53
5.3 製造部門功能	53
5.3.1 晶圓編號新增	53
5.3.2 晶圓編號更新	54
5.3.3 晶圓編號刪除	56
5.3.4 製造準備(JOB PREP)	58
5.3.5 製造開始(JOB IN)	59
5.3.6 製造結束(JOB OUT)	61
第六章 結論	63
6.1 總結	63
6.2 系統效能評估	64
6.2.1 測試程式邏輯資料正確性	64
6.2.2 測試使用系統與介面切換的時間比較	65
6.3 未來發展方向	67
參考文獻	68

表目錄

表 1 lotinfo 資料欄位說明	26
表 2 lotinfo 資料資料範例表	26
表 3 operinfo 資料欄位說明	27
表 4 operinfo 資料範例表	27
表 5 operseq 的資料欄位說明	28
表 6 operseq 資料範例表	28
表 7 eqpinfo 的資料欄位說明	29
表 8 eqpinfo 資料範例表	29
表 9 userdef 的資料欄位說明	36
表 10 userdef 資料範例表	37
表 11 登入機制測試事件	64
表 12 站點查詢測試事件	65



圖目錄

圖 1 主從式系統架構	8
圖 2 CORBA 2.0 ORB 的系統架構[5]	8
圖 3 三層式發展架構(Three-Tier Architecture)[6]	10
圖 4 Java Naming and Directory Interface (JNDI)[15]	16
圖 5 ODBC 的架構圖[16]	19
圖 6 JDBC 共有四種類型[19]	21
圖 7 利用 JDBC/ODBC 橋接驅動程式連接資料庫	22
圖 8 Connection Pool[24]	23
圖 9 晶圓製造與主要相關部門關係	25
圖 10 無塵室機台擺設位置	30
圖 11 製造工程師與各資料庫的關係	34
圖 12 製程工程師與各資料庫的關係	35
圖 13 設備工程師與各資料庫的關係	35
圖 14 晶圓流程控制(WFCMS)系統與各資料庫架構	36
圖 15 晶圓流程控制系統功能圖	38
圖 16 驗證使用者身份順序圖	38
圖 17 JOBPREP 作業流程順序圖	39
圖 18 JOB IN 作業流程順序圖	41
圖 19 JOB OUT 作業流程順序圖	42
圖 20 製造部門與各部門資料庫的關係	43
圖 21 製程部門與資料庫的關係	44
圖 22 設備部門與資料庫的關係	44
圖 23 系統實作架構(Three-Tier Architecture)	45
圖 24 登入運作流程圖	46
圖 25 使用者登入表單畫面	47
圖 26 共同功能介面表單畫面	47
圖 27 站點查詢表單畫面	48
圖 28 站點查詢運作流程圖	48
圖 29 站點查詢結果	49
圖 30 站點流程表單畫面	49
圖 31 站點流程查詢運作流程圖	50
圖 32 站點流程查詢結果	50
圖 33 機台資料查詢表單畫面	51
圖 34 機台資料查詢運作流程圖	51
圖 35 機台資料查詢查詢結果	51
圖 36 晶圓資料查詢表單畫面	52
圖 37 晶圓資料查詢運作流程圖	52
圖 38 晶圓資料查詢結果	53
圖 39 晶圓編號新增表單畫面	53
圖 40 晶圓編號新增運作流程圖	54

圖 41	晶圓編號新增結果	54
圖 42	晶圓編號更新表單畫面(一)	55
圖 43	晶圓編號更新運作流程圖	55
圖 44	晶圓編號新增表單畫面(二)	55
圖 45	晶圓編號更新結果	56
圖 46	刪除晶圓編號表單畫面(一)	56
圖 47	刪除晶圓編號運作流程圖	57
圖 48	刪除晶圓編號表單畫面(二)	57
圖 49	刪除晶圓編號結果	57
圖 50	製造準備運作流程圖	58
圖 51	製造準備表單畫面(一)	59
圖 52	製造準備表單畫面(二)	59
圖 53	製造準備結果	60
圖 54	製造開始運作流程圖	60
圖 55	製造開始表單畫面(一)	60
圖 56	製造開始表單畫面(二)	60
圖 57	製造開始結果	61
圖 58	製造結束運作流程圖	61
圖 59	製造結束表單畫面(一)	62
圖 60	製造結束表單畫面(二)	62
圖 61	製造結束結果	62
圖 62	製造準備階段花費時間比較圖	67



第一章、緒論

目前企業內部的資訊系統及生產製造系統等，因當初規劃需求，各有其特屬資料庫。異質性資料庫的產生，乃基於企業策略上的考量，企業內部各資料庫依各部門需求開發的結果，造成企業資料分散而且各自有其特殊的介面及溝通方式，而資料庫管理工具更是多樣性、異質性。

在生產製造系統上，各家生產設備的廠商不同，相對使用的系統也不同，增加在生產管理上的困難度。在控制生產流程，往往需要至性質相異的資料庫取得參考的資料來源。各部門必須提供相對應資料庫的應用程式，在共同使用的電腦平台上，讓使用者讀取及更新各部門資料庫的資料。各部門所開發的應用程式在同一部電腦上執行，容易因使用的物件不相容，導致電腦當機及資料讀取與更新錯誤的風險，因此異質資料庫間資料的整合成為企業內部重要的研究議題。

1.1 節前言介紹企業內部的資訊整合方式。1.2 節為動機，考量企業內部各異質性資料庫系統發展歷史及策略聯盟的合作發展計畫等因素。1.3 節為研究目標，改善目前作業流程風險、因應自動化的需求、避免資料維護的人為錯誤，研究一套結合網際網路應用系統。1.4 節為論文大綱說明論文章節安排。

1.1 前言

企業內部的資訊整合，若從頭重新開發全新的整合性資料庫，這意味需要把各家廠商生產設備及系統一起整合起來，是一件很龐大而且不可能的工程，花費在資訊系統重新建置上的成本將是很可觀。如果建立一個分散式資料整合系統，來整合企業內部現有的資訊系統及各生產製

造系統所屬獨立異質資料庫，則各家生產設備的廠商及系統不需要重新設計，而減少許多不必要的成本。

異質資料庫整合方式有以下幾種：1. 以閘道(Gateway) [1]的方式，需要各異質系統配合，研擬出一個可行的統一規格，並依規格去開發整合。2. 以中介軟體(Middle Ware)及三層架構(Three-Tiers) [2]的方式，使用共同的介面(Interface)去整合各異質資料庫系統。3. 以 XML [3]技術的方式，利用自定標準的資料格式，來達到各異質資料庫資訊交流的媒介。

許多企業內部的資料庫或應用系統，希望能有效且充分地利用這些資訊，但這些資料庫因介面及系統平台不同，而導致在資料流通上，非常困難。為了存取異質性資料的方便，此時程式設計師可能要面臨將這些資料庫介面、新的應用程式介面改寫、或是使用應用程序將各自的資料做匯出至文字檔，再匯入另一個資料庫、或是使用手動的方式將資料直接輸入等。上述的方式，複雜而費時，但是又無法根本解決異質性資料庫資料流通的問題，因此需要一個有效的方式整合異質性資料系統。

1.2 動機

目前企業內部資訊系統及生產製造系統通常是異質性的資料庫系統，而且有各自特有的存取介面及系統平台，此乃因當初技術發展的歷史及策略聯盟的合作發展計畫等因素的考量。

目前企業內部各異質性資料庫系統的規格不同，產生資料庫系統、電腦系統硬體、作業系統之間的差異如下：

1. 資料庫系統：如 Oracle 的資料庫系統、MS SQL 等，當中的差異在

於資料模式及資料結構的不一致，資料表綱要(Schema)的表達方式，查詢介面及語法不同，很容易渾淆。

2. 電腦系統硬體：如同伺服器及工作站就有 HP、DEC、IBM、SUN、Toshiba 等廠牌，系統規格不同，大部份的硬體都不相容，如 HP 廠牌的硬碟，在 IBM 的機器可能無法使用等。
3. 作業系統：有使用 DEC-UNIX、HP-UNIX、SUN Solaris、Linux 等作業系統，甚至為了維護方便，使用 Windows NT、Windows 2000 Server 等作業系統，這些系統不論在介面及管理方式都有很大的差異。

上述的差異增加異質性資料系統整合的困難度及複雜度，如果以閘道(Gateway)的方式，需要與提供資料庫的各廠商協調，研擬出整合資料的方式，請各資料庫廠商更改資料庫的資料交換程式。因公司企業的资料整合需求，更改資料庫程式，並配合不同平台的資料庫做程式整合，以達到資料交換的目的。如以上述方式做資料庫整合，各資料庫廠商因商業機密、時間成本等考量，配合願意不高，很難達成資料整合目標。

以 XML 技術的方式，主要是應用在企業對企業的電子商務上，雖然 XML 文件的功能可以做資料交換而且能用來描述網路資源及相關語意，但是企業內部有些資料庫系統，如 Process DB，屬於早期的資料庫系統，並不支援 XML 的文件物件模式及資源描述架構，因此以 XML 技術的方式，不適合整合目前公司內部的異質性資料庫系統。

1.3 研究目標

由於半導體的製程非常繁雜，製造工程師在製造流程控管上，需要各部門提供相關的製程資料。各部門為提供給製造工程師在製造流程上的相關資料，各自開發資料庫的應用程式。把開發完成的應用程式放置在無塵室的公用電腦上，讓製造工程師透過這些應用程式連結至各部門資料庫，或是提供給自己部門使用。

製造工程師在每一個製程站點，因為流程製造上的控管，需要不斷在公用電腦上切換各部門的應用程式。查詢及更新資料，會因為應用程式切換頻繁，導致製造工程師的混亂，產生操作錯誤(如資料輸入錯誤、查詢與更新錯誤的資料等)，影響產能。

各部門的應用程式，在更新版本上，會因為使用到的物件發生衝突，導致有些部門的應用程式可以使用，有些部門的應用程式無法使用，嚴重一點的衝突，讓公用電腦整個當機。

為改善上述的作業流程風險、因應自動化的需求、避免資料維護的人為錯誤，研究一套結合網際網路應用系統，建立以瀏覽器為標準的架構。透過統一的使用者介面，在各異質性資料庫系統間交換訊息，來達成資料維護的目標，使得企業內部的異質性資料資訊系統，能有效率的進行資料交換與整合。

1.4 論文大綱

在本篇論文中，在現行的異質資料庫系統，發展出晶圓流程管理系統(Wafer Flow control Management System)，簡稱WFCMS，減少資料維護的人力資源及避免資料維護的人為錯誤，同時又能在各異質資料庫系統上做資料整合。

論文章節安排如下：第二章是資料整合及相關文獻研究。第三章為研究方法，研究現行作業流程。第四章則說明結合網際網路應用系統整合異質型資料庫。第五章是系統實作，針對晶圓流程管理系統(WFCMS)運作流程做描述。第六章則是本論文的結論。



第二章、相關文獻探討

將各個獨立資料庫整合成為一個整體資料庫，並不是一件容易的事，因為所有的獨立系統從資料庫系統、電腦系統硬體、作業系統都具有不同程度的差異。本章說明整合公司內部的資料處理的系統架構，比較應用程式與資料庫系統的連結方式，並依公司內部資料管理系統的現況做討論與評估。

2.1 節簡述研究緣由。2.2 節介紹資料處理的系統架構，比較主從式系統與三層式系統。2.3 節說明客戶端與應用程式伺服器連結開發。說明可延伸性標示語言(XML)、應用程式開發、客戶端網頁開發、應用程式伺服器連結的機制。2.4 節描述應用程式伺服器與資料庫伺服器之連結。說明 JDBC、ODBC 架構與資料庫連結的機制。

2.1 緣由

近年來，電腦系統與網路環境漸漸朝向開放式的走向，加上網際網路的擴充速度驚人，企業內部資料的處理早期由人工手寫的紙本做資料的管理，進入電腦化資料管理。公司各部門將部門所控管的資料作分析與應用，選擇合適的資訊系統，將資料資訊化。各部門因技術支援的考量各自購買所需的資料庫系統。

現今企業面臨競爭優勢的抉擇，對企業內部資料要能準確的掌握、分析與應用，將相關的獨立資料庫做整合，讓管理者可以透過整合後的介面查詢到所需要的資訊，依據即時的資訊做出對公司最有利的決策。管理者與使用者不知道資料是由那一個資料庫所提供的，對於管理者與使用者來說，只要透過相同的介面，就能掌握所需的資訊。

2.2 資料處理的系統架構

資訊技術的應用已深入各行各業，加上電腦網路的擴散速度驚人、全球資訊網的建立，電腦已是企業不可或缺的工具。企業目前的資料已經電腦化，資料處理的系統架構大致分為主從式系統(Client-Server Systems)[2]、三層式(Three-Tier)[2]應用系統發展架構等。

2.2.1 主從式系統(Client-Server System)

主從式系統的作業方式，是以客戶端(client)向伺服器(server)提出要求服務，是屬於主動要求；伺服器(server)回應客戶端(client)的各種服務，即是被動式服務。如圖 1，客戶端(client)只要透過網際網路向伺服器(server)，如電子郵件伺服器、檔案伺服器、資料庫伺服器等，提出要求服務，這些伺服器(server)會回應客戶端(client)所提出的服務，當然會有權限控管機制。

在軟體元件設計考量上，使用分散式元件中介軟體，如CORBA[4][5]所提供分散在各個機器上的物件透過中介軟體的連結，讓這些可重複使用的現有軟體能夠相互合作來完成網際網路的服務。

CORBA的機制與策略，如圖2所示。先實作出客戶端(client)與伺服器(server)兩個元件，以IDL(Interface Definition Language)描述個別的CORBA介面，再編譯出stub和skeleton，透過這兩個介面可以讓client元件以類似RPC(remote procedure call)的方式使用server元件提供的服務。CORBA的客戶端(client)與伺服器(server)的運作方式，當客戶端(client)提出要求服務時，客戶端(client)會利用透過CORBA

的ORB(Object Request Broker)的介面向伺服器(server)提出需求，伺服器(server)也會把客戶端(client)的需求，透過ORB的介面回傳給客戶端(client)。

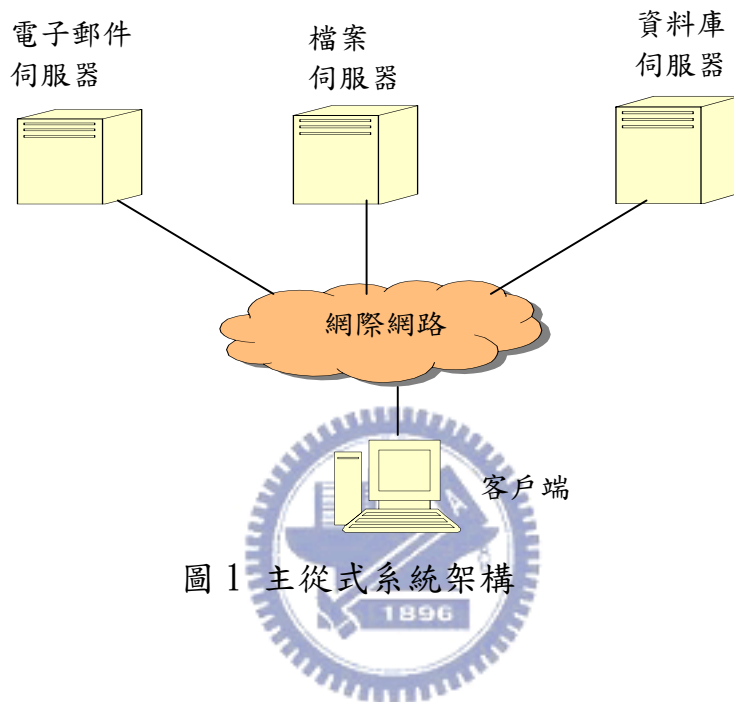


圖 1 主從式系統架構

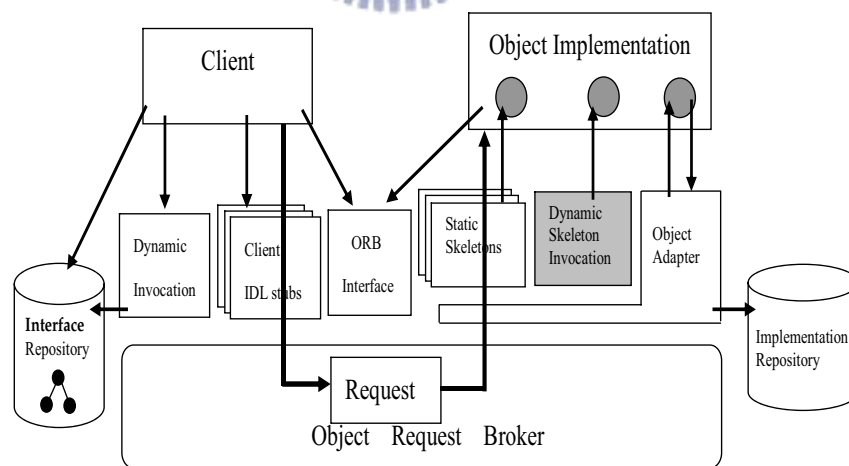


圖 2 CORBA 2.0 ORB 的系統架構 [5]

2.2.2 三層式應用系統發展架構

隨著國際間網際網路 (Internet) 與全球資訊網 (WWW, World Wide Web) 的興起，使得國內企業的資訊運用興起一股網路熱。瀏覽器 (Web Browser) 的使用非常的普遍及便利，使用者或管理者透過瀏覽器提供資料庫存取的介面，存取資料變得很方便。加上與資料庫存取的軟體工具 (如 Java script、Java、VB script、Active X 等) 的發展非常迅速而且不斷地提供功能強大的物件，讓程式設計師可以設計出更有彈性的程式，運用於公司內部異質性資料庫的整合。使得應用程式的發展架構漸漸朝向「三層式發展架構」(Three-Tier Architecture)——分成客戶端 (Client)、Web 伺服器 (Web Server)，與資料庫伺服器 (DB Server) 三個層次來發展。

三層式發展架構 (Three-Tier Architecture) [6]，如圖 3 所示。資料庫伺服器 (DB Server)，如 Oracle、Sybase、MSSQL 等資料庫管理系統，負責儲存或處理應用程式 (AP Server) 所需的資料。Web 伺服器透過中介軟體 (middleware) 提供資料庫連結的功能，至資料庫存取所需要的資料。這類中介軟體 (如 JDBC、ODBC 等)，提供給前端開發程式統一的資料庫驅動程式，程式設計師透過此資料庫驅動程式，便能夠存取不同廠商所提供的資料庫 (異質資料庫)。

Web 伺服器則是負責接收來自客戶端的需求、管理客戶端的連結、建立瀏覽器的網頁。客戶端利用瀏覽器與 Web 伺服器連接，透過瀏覽器網頁讓使用者與伺服端和資料庫做資料的溝通。客戶端與 Web 伺服器之間透過中介軟體 (如 Java、Java Script、VB Script 等) 達到資料交換的目的。

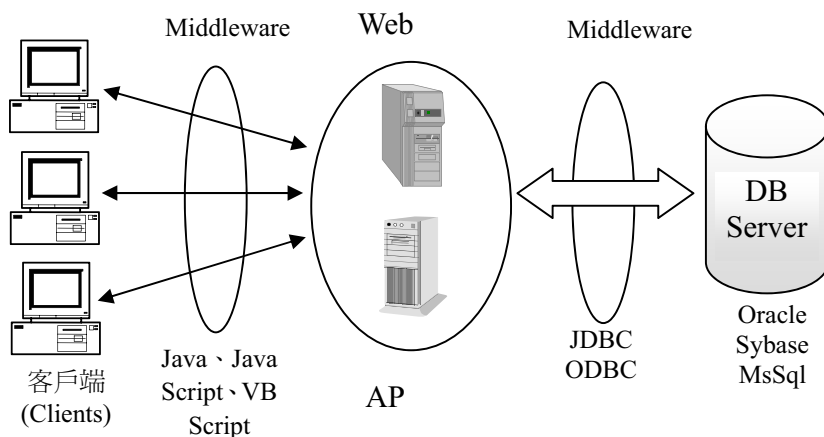


圖 3 三層式發展架構(Three-Tier Architecture)[6]

2.2.3 主從式系統與三層式系統的比較

公司的網路應用程式如採主從式系統架構，會遭遇到分散式元件中介軟體(如 CORBA)並沒有提供預設元件管理、資源分配以及服務搜尋的功能。程式設計師則要額外撰寫元件管理、資源分配的程式，容易造成錯誤。有些元件中介軟體(如EJB server)，需要跟提供軟體公司購買專業版本，才能有完整功能的元件可以使用，購買專業版本的售價對公司部門的成本是一種負擔。公司的網路應用程式如採三層式系統架構，會有如下優點：

1. 所有客戶端運用瀏覽器，便可以使用介面統一應用程式，應用程式則可以與其他資料庫連結，並回傳客戶端所需要的資料。
2. 由於所有的資料均由伺服器端集中掌控，所以管理上也比較不會有主從式架構中所引發的缺點，如因沒有共用資料的模型，資料的交換可能就會沒有效率、每一部伺服器都必須做重複的管理、伺

服务器的名稱和服務沒有集中登記，所以不容易找到哪一部伺服器提供哪些服務等。

3. 在資料庫方面，應用程式伺服器端透過如 ODBC，JDBC 等中介軟體共通介面，可以存取不同類型的資料庫。
4. 在程式維護方面，主從式架構上的應用程式有改版，則所有客戶端均需重新安裝。但是採用三層式系統架構後，只要將應用伺服器端的網頁更換，所有客戶端便可以使用到應用程式的最新版本，完全不需要重新安裝。

2.3 客戶端與應用程式伺服器連結開發

2.3.1 可延伸性標示語言(XML)

XML(eXtensible Markup Language)[7]是新一代的可延伸性標示語言。由國際標準組織(International Standards Organization, ISO)所公佈的一個名為「標準通用標示語言」(Standard Generalized Markup Language, SGML)的精簡版。可以改善傳統網頁設計語言 HTML 的描述方式，未來將可取代 HTML 成為新一代的網頁標示語言。

利用 XML 所具有的可延伸性以及自我描述(self-descriptive)[8]特性，從電子文件內容之收集、資料的處理解析、儲存、自動傳輸，乃至後端的資料搜尋比較、運算、再處理和呈現，使用 XML 可以輕易達成。不同廠商的電子商品型錄可以在一個使用者介面同時呈現，資訊的搜尋變得更為精確快速，不同系統間可以流暢的互通。

一般應用 XML 處理異質資料時，大致可以分為四個步驟[9]：

1. 準備原始文件：將欲轉換成 XML 的原始資料準備妥當，然後撰

寫 XML 基本文件。

2. 製作 XML 文件：將準備好的異質資料原始文件做結構化的處理，根據結構化處理後的架構利用 XML Schema 定義所需的元素和屬性。然後藉由定義好的 XML Schema 來撰寫 XML 文件。
3. 處理 XML 文件：在使用 XML 文件之前，必須使用 SAX 或者是 DOM 擷取 XML 文件裡的資料內容。這兩者之間的差別在於 SAX 是使用循序處理的方式，而 DOM 則是利用文件的樹狀結構擷取。然後把擷取出來的資料片段，傳送給下個步驟的程式。
4. 使用 XML 文件：ASP，VBScript，JavaScript 等程式可以將 SAX 或者是 DOM 擷取出來的 XML 文件資料做進一步的處理。

許多資訊業界的龍頭廠商如 Microsoft、Netscape、IBM、Oracle、Adobe、Sun、HP 和 SAP...等，皆各自提出了支援 XML 的解決方案和相關應用。針對目前企業內部各自獨立資料庫系統而言，如 Process DB，在準備原始文件的階段，因其資料庫系統沒有提供 Web 的型式及介面，是獨立封閉的，存取須要透過自己特有的介面，無法與 Web 應用伺服器作溝通。因此想以 XML 的方式來定義文件格式及標示語法，並以標準化的方式呈現以及交換文件，似乎有很大的困難。

2.3.2 應用程式開發-JAVA 語言

Java 語言[10]是昇陽公司 (Sun Microsystems, Inc.) 在 1990 年初所設計的程式語言。從 1995 年的暑假開始即受到電腦業界的高度注意，特別是在 Internet 和多媒體 (Multimedia) 相關產品類方面。而「Java」之所以命名為「Java」是美國昇陽電腦公司 Java 發展小組歷

經無數次的腦力激盪會議之後才被選擇出。

Sun 的 Java 語言白皮書(White paper)[11]中說明 Java 語言的特點：

1. 簡單(Simple)：容易撰寫程式，不需要長時間的訓練，而能滿足現代的需求。
2. 物件導向的(Object-Oriented)：物件導的設計是一種重心在資料和介面的技巧；隨插即用(Plug and Play)的物件是物件導向設計的重點。
3. 分散式的(Distributed)：Java 有一個很週全的程式庫，且很容易地與 HTTP 和 FTP 等 TCP/IP 通訊協定相配合。Java 應用程式(Applications)能在網路上開啟及連結使用物件，就如同透過 URL 連結使用一個區域檔案系統(Local File System)。
4. 強韌性的(Robust)：由 Java 所撰寫出的程式能在多種情況下執行而具有其穩定性 Java 與 C/C++ 最大不同點是 Java 有一個指標器模型(Pointer Model)來排除記憶體被蓋寫(Overwriting Memory)和資料毀損(Corrupting Data)的可能性。
5. 安全性的(Secure)：Java 擁有數個階層的互鎖(Interlocking)保護措施，能有效地防止病毒的侵入和破壞行為的發生。
6. 架構中立性的(Architecture Neutral)：Java 的編譯器產生一種結構中立物件的檔案格式(Object File Format)；這使得編譯碼得以在很多種處理器中執行。
7. 可攜帶性的(Portable)：程式庫屬於系統的一部份，它定義了一些可攜帶的介面，Java 本身具備有很好的可攜帶性。
8. 解譯的(Interpreted)：Java 解譯器能直接地在任何機器上執行

Java 位元碼(Bytecodes)，此在進程式連結時，時間的節省，這對於縮短程式的開發過程，有莫大的助益。

9. 高效能的(High Performance)：Java 位元碼迅速地能被轉換成機械碼(Machine Code)。
10. 多緒的(Multithreaded)：Java 語言具有多重線串的功能，這對於互動回應能力及即時執行行為是有助益的
11. 動態的(Dynamic)：Java 比 C 或 C++語言更具有動態性，更能因應時時刻刻在變的環境，Java 不會因程式庫的更新，而必須重新編譯程式。

由於 JAVA 語言有以上的優點，公司內部的應用程式都採 JAVA 語言開發，因此晶圓流程管理系統(WFCMS)在程式開發上選擇使用 JAVA 語言。

2.3.3 客戶端網頁開發-JSP

隨著網際網路的快速發展，附加在網際網路上的功能越來越多樣化，網頁的設計，也已從最初單純靜態的呈現資料、簡單的資料處理，進展到可以與使用者互動，也可以加入各種物件使之更富有變化。Java Server Pages(JSP)[12]，是由 Sun Microsystem 公司所發展出來一種新的規格標準，能夠動態的呈現網頁、與使用者產生互動、可以完成資料處理的網頁。

JSP 是使用將 Java 程式包裹在標籤(tag)中，執行 JSP 程式時，程式部分由伺服器端處理，而客戶端只需作 HTML 的處理，提供了網頁設計者一個更簡捷、更快速的方法，以動態產生的方式來設計、維護一個網頁。JSP 既然是 Java 家族的一員，自然也承襲了 Java 一直以來所強

調的優點，那就是跨平台的設計，JSP 並不限定在特定的作業平台或網路伺服器上才能執行，因此，給予網頁設計者更大的發揮空間。

JavaServer Pages 包括下列的優點[13]：

1. 可在 Apache、Netscape、IIS 等伺服器上執行。
2. 支援的作業平台有 Solaris、Windows、Mac、Linux 等。
3. JSP 允許程式設計師使用 Java 語言來撰寫程式，給予程式設計人員較多的選擇。
4. 第一次執行 Jsp 程式時即執行 Compile，不需要每次執行 Jsp 程式時都 Compile 一次。
5. 可以與 ODBC、JDBC 相容的資料庫整合。

由於 JSP 有以上的優點，晶圓流程管理系統(WFCMS)在客戶端動態網頁部分選擇使用 JSP 開發。



2.3.4 客戶端(Client)與應用程式伺服器連結的機制-JNDI

JNDI(Java Name and Directory Interface)[14]是一種樹狀的結構，每個節點都可以儲存任何物件，客戶端程式可以透過JNDI路徑(JNDI path)來查詢所儲存的物件或是更新物件的值，如同檔案系統是樹狀的結構，必須透過路徑來存取檔案，延伸到client-server的觀念。一個實作JNDI介面的軟體，將有能力提供企業應用程式標準的目錄運作，JNDI的路徑的定義不限是檔案，可以是廣義的物件，跟檔案系統的定義不太相同。企業的應用程式將可以存取任何型態Java所命名的物件，同時JNDI也可以用來整合各種知名的目錄服務（如LDAP、NDS、DNS、NIS等），達到與傳統應用程式或系統共存的目的。

當欲更換連結資料庫的驅動程式及伺服器端使用的資料庫時，須更改所有連結資料庫的敘述，對程式設計師來講，需在針對不同程式的連結資料庫部份更改相同的程式碼，是非常沒有效率。運用 JNDI 的技術，在 Web 程式受到驅動程式、資料庫位置等資料改變，不需做重複修改程式的動作。

針對 JNDI 的技術做說明，如圖 4 分兩部分[15]：應用程式程式介面(API)和服務提供者介面(SPI)。JNDI API 容許 Java 應用程式存取各種命名和目錄服務。JNDI SPI 提供給服務資料庫供應商（也包括目錄服務提供者）使用，透過 JNDI SPI 可以取用 DNS、RMI、CORBA 等所提供的服務，這使得各種各樣的目錄服務和命名服務能夠透明地插入到使用 JNDI API 的 Java 應用程式中。

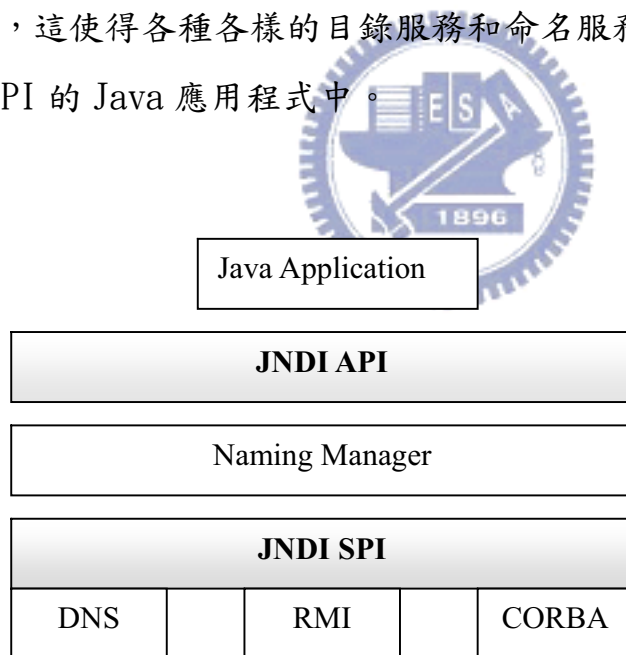


圖 4 Java Naming and Directory Interface (JNDI)[15]

客戶端與應用程式做連結定義為 JNDI 資源，在客戶端的程式以 context 物件去 JNDI 取得 Home 物件。在應用程式伺服器端的程式，取

用 JDNI 資源的介面，先在 DataSource 介面的 javax.sql 套件檔案定義，並宣告實作此介面之類別實體，便可取得實作連結的物件。

2.4 應用程式伺服器與資料庫伺服器連結

應用程式伺服器透過標準資料界面（如 JDBC、ODBC 等）與資料庫連接，因此開發過程中不需指定特定的資料庫系統，資料庫系統的開放性被建立，打破彼此間的異質性。應用系統開發者只需依循既有的標準，全心全意投入系統開發，將來系統完成後，很容易可以移植在不同的平台上。

目前企業內部的資料庫系統，應用程式伺服器與資料庫連接可透過標準資料界面有 JDBC、ODBC 兩種，敘述如下：

2.4.1 開放式資料庫連接界面(Open DataBase Connectivity)

開放式資料庫連接界面 ODBC (Open DataBase Connectivity) [16]，是一組程式界面的定義，由 Microsoft 公司根據 SQL Access Group 制定的 Call Interface 規格設計出來的資料庫系統定義的標準界面，因此應用程式可以同時與一個或多個資料庫系統連接，為資料庫的開放架構建立了一個發展方向。ODBC 建立在主從架構上，並提供一組 API (Application Programming Interface)，使應用程式可以透過這一組 API 把 SQL (Structured Query Language) [17]指令送到資料庫系統中存取資料。如圖 5，ODBC 的架構包含了 4 個部份：

1. 應用程式 (Application)

應用程式對外提供使用者交談介面，同時對內執行資料之準

備工作和呼叫 ODBC 程式函數，傳送 SQL 指令以及接收資料庫系統所傳回來的結果，顯示給使用者。

2. ODBC 驅動管理員 (ODBC Driver Manager)

驅動管理員本身在 MS Windows 中一個動態連接程式庫檔 (ODBC.DLL)。應用程式透過驅動管理員去載入並連結資料來源的驅動程式 (driver) 並連接資料來源。

3. 驅動程式 (Driver)

驅動程式也是一個動態連接程式庫檔，當應用程式呼叫 ODBC 函式，SQLConnect 或 SQLDriverConnect 時，驅動管理員就會載入相對的驅動程式與應用程式呼應。驅動程式主要是執行 ODBC 之相對函式，並與對應之資料來源 (Data Source) 做溝通。

如對應之資料來源是 Oracle，驅動管理員就會載入 Driver to Oracle、對應之資料來源是 SQL Server，驅動管理員就會載入 Driver to SQL Server、對應之資料來源是 Sybase，驅動管理員就會載入 Driver to Sybase。

4. 資料來源 (Data Source)

資料來源是指資料庫系統 (DMBS) 或是資料庫操作系統，也就是說，應用程式系統可以其中一個資料庫來源連接(如 Oracle、SQL Server、Sybase)。

2.4.2 JAVA 資料庫連結器(JDBC)

JDBC(Java Database Connectivity)[18]是由昇陽(Sun)公司開發的一組存取資料庫的應用程式介面 API(Application Programming

Interface)，定義使用 JAVA 實作資料庫連結。JDBC API 分為兩個部份 [19]，第一部分為低階 API 部分是 JDBC 驅動程式真正的連結並與資料庫作溝通，第二部份為高階 API 部分通常是傳遞資料到應用程式。

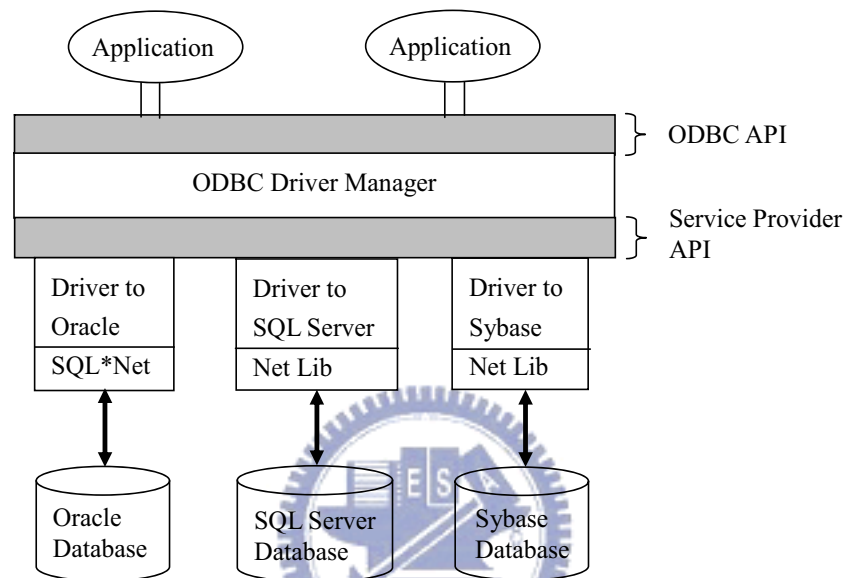


圖 5 ODBC 的架構圖 [16]

JDBC 是一種存取虛擬資料庫的應用程式介面，可在各種不同的資料庫系統運作，支援多重資料庫和多重平台之間的資料溝通。JDBC 有三個工作 [20]，(1) 建立與資料庫的關聯用，(2) 傳送資料查詢語言，(3) 處理結果。如圖 6，JDBC 共有四種類型 [18]，稱為 TYPE 1~4，各類型說明如下：

Type1：JDBC-ODBC 橋接(bridge)驅動程式

這個驅動程式是內建的 package，透過這個驅動程式跟 ODBC 做連結，因為大部分資料庫都已經支援 ODBC，所以使用 JDBC-ODBC 可

以與現今大部分的資料庫進行存取的动作，缺點是透過中間轉換的過程，所以速度慢。

Typy2：Native-API Bridge(原生 API 結合 Java 驅動程式)

將每個 JDBC 的呼叫指令轉換成個別的資料庫系統的原生程式呼叫(native function call)。驅動程式上層使用 Java 程式與 Java 應用程式作溝通，將 JDBC 呼叫轉為原生程式碼的呼叫，下層為原生語言（像是 C、C++）來與資料庫作溝通，下層的函式庫是針對特定資料庫設計的，此類型的驅動程式，由資料庫廠商所提供。

Type 3：網路協定搭配完整的 Java 驅動程式(JDBC-middleware)

將每個 JDBC 的呼叫指令(function call)轉換成個別的資料庫系統的網路協定，接著再轉換成資料庫的原生呼叫(native call)。透過資料庫中間物件(DB Middleware)來存取資料庫，使用者不必安裝特定的驅動程式，而是由驅動程式呼叫中間物件，由中間物件來完成所有的資料庫存取動作，然後將結果傳回給驅動程式。

Type 4：Java 驅動程式(Pure Java Driver)

使用純 Java 撰寫驅動程式與資料庫作溝通，而不透過橋接或中間件來存取資料庫，例如 MySQL 的 JDBC 驅動程式屬於 Type 4。

2.4.3 JDBC 與 ODBC 互相搭配

使用 JDBC 與 ODBC 撰寫的應用系統可以存取不同廠牌的資料庫管理系統甚至非關聯性資料庫，在跨平台或跨資料庫的應用上更具可攜性。所以當一個應用系統必須考慮到未來使用的彈性時，選擇標準界面

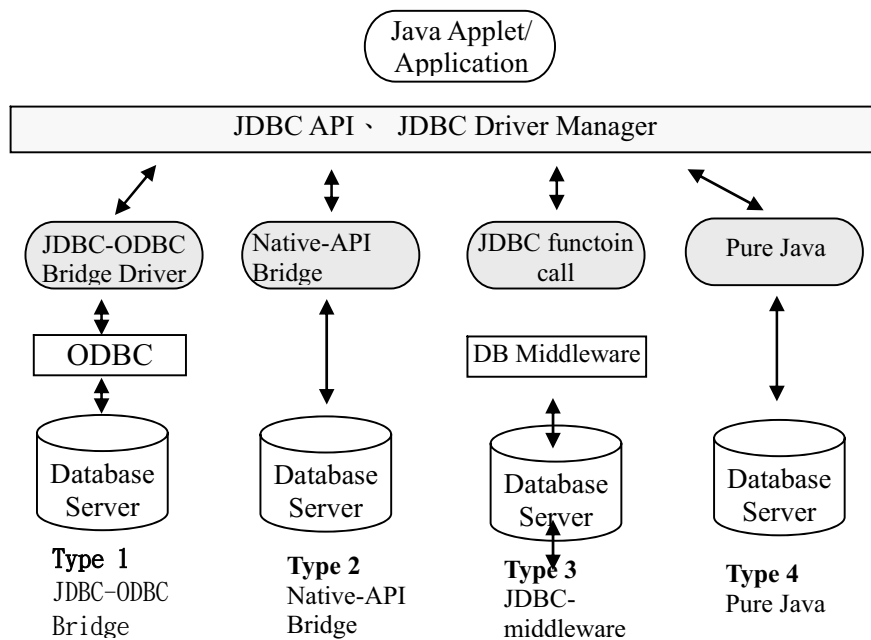


圖 6 JDBC 共有四種類型[19]

可以提高環境適應能力。從技術面來看，開放式系統最基本好處在於可移植性 (Portability)，以 JDBC 與 ODBC 完成的應用程式都可以同時連接一個到多個相同或不同的資料庫。

JDBC 是一種用來提供執行動態 SQL 語法敘述的 Java API，也是在 Java 和資料庫互動中，使用的一項機制。ODBC 使用 C 語言撰寫，若使用 Java 直接呼叫 C 的原生碼，會有安全性與移植性的問題，而且 JAVA 所使用的字串與 ODBC 的字串有相容性的問題。依企業本身的考量，由於 JAVA 語言的可攜式，使得公司內部的應用系統已趨向使用 JAVA 語言來設計，因此在撰寫與資料庫溝通的應用程式，晶圓流程管理系統(WFCMS)選擇與 Java 相容性最高的 JDBC 方式連結。

對於比較早期的製程資料庫系統(如 Process DB)只提供 ODBC API，

JDBC 則無法提供相對的 API，如圖 7 所示，JDBC Type1 的 JDBC-ODBC 橋接器，JAVA 程式透過 JDBC，再呼叫 JDBC-ODBC 橋接器與支援 ODBC 驅動程式的資料庫作溝通。



圖 7 利用 JDBC/ODBC 橋接驅動程式連接資料庫

2.4.4 應用程式與資料庫連結的機制-connection pool

伺服器與資料庫建立連線會佔伺服器記憶體空間，如果同一時間有很多個客戶端需要與資料庫建立連線，容易造成系統記憶體不足，產生系統不穩定。通常為了確保伺服器有足夠的記憶體空間去處理客戶端需要，資料庫連線數是有限制的。

一般資料庫建立連線的方式是由客戶端提出請求後，伺服器判斷客戶端需要擷取的資料，與資料庫建立連線並取得資料，回傳給客戶端的網頁後，便釋放資料庫的連線。在資料庫連線數未達上限，允許其他客戶端繼續與資料庫建立連線。

當很多客戶端或是應用程式需要很頻繁與資料庫進行連線擷取資料，Web 應用程式需要不斷重覆與資料庫建立連結。完成資料傳輸動作，則需要與資料庫建立連結關閉，對伺服器來說，建立與關閉資料庫連結是非常耗費系統資源。

為了減少伺服器與資料庫建立連線的次數，運用 Connection

Pool[21]的技術取代傳統每次與資料庫交易都要建立與釋放連線。伺服器開始啟動時，如圖 8，在記憶體劃分出一塊區域(Connection Pool)來存放連線的相關資料。編號為 A 實心圓點，表示當客戶端需要與資料庫連線時，直接向記憶體取得連線就可以與資料庫溝通。每當需要與資料庫連線時，都會向 Connection Pool 要求取得一個連線，節省了不斷建立連線的時間並提高 CPU 使用率，提昇網站效能。如果連線已被其它請求所使用，伺服器在 Connection Pool 的連線數量未達上限時，可再建立一個新的連線。編號為 B 實心圓點，即是伺服器再建立一個新的資料庫的連線。當完成資料傳輸動作，與資料庫建立連結還給 Connection Pool 管理，給下一個請求使用，如編號 C 空心的圓點。如果 Connection Pool 的連線已達上限且目前沒有可使用的連線，則客戶端送來的這個請求會停留在等待的佇列，直到 Connection Pool 有可用的連線。

本論文實作的系統(WFCMS)在與資料庫做連結時，採用 Connection pooling 的機制，增加程式的執行效率，避免耗損資料庫伺服器的系統資源。

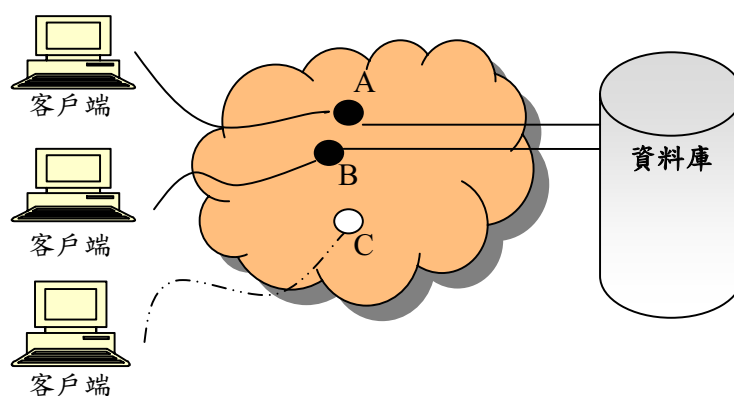


圖 8 Connection Pool[24]

第三章 作業研究

本章針對晶圓製造流程、企業各部門的工作職掌、機台生產程序與各部門配合事項做說明。

3.1 節說明半導體積體的矽晶圓編號等。3.2 節介紹晶圓製造主要相關部門職掌說明，晶圓製造需要製造部門、製程部門、設備部門、自動化部門等四個部門配合。3.3 節描述機台生產流程，說明生產晶圓完整製程流程經歷的三個步驟。

3.1 晶圓(Wafer)

晶圓(Wafer)乃是指矽半導體積體電路製作所用之矽晶片，由於其形狀為圓形，稱為晶圓；在矽晶片上可加工製作成各種電路元件結構，而成為有特定電性功能之 IC 產品。一片晶圓的厚度大約是 600um，在生產製造過程中，通常會以 24 片晶圓為一個單位，裝在一個晶盒裡，此晶盒稱為一個 LOT，並生產製造開始會給晶盒一個編號，稱為批號(LOT ID)，以後的製程研發、生產製造管理、機台產能管理等，會與批號相關。

生產製程的工程師將晶盒送入各相關的機台(EQP)，進行生產製造。每一個晶圓的製程步驟，給於一個站點編號(Operation ID)，方便對每一個製程步驟做管理控制。對於不同功能晶圓的產品，其製程站點步驟的順序會不同，因此在製程站點步驟上，會定義其生產順序，依據站點順序編號(Sequence No)，執行生產製程。

3.2 晶圓製造主要相關部門職掌

在晶圓的每個製程站點步驟，都相當的重要，需要各相關部門配合及協調並主動積極支援，才能完成多達上百個製程站點的晶圓製造。主要的相關部門職掌與各部門的各自獨立的資料庫說明，如圖 9 所示。晶圓的製造，大致需要製造部門、製程部門、設備部門、自動化部門等四個部門配合。

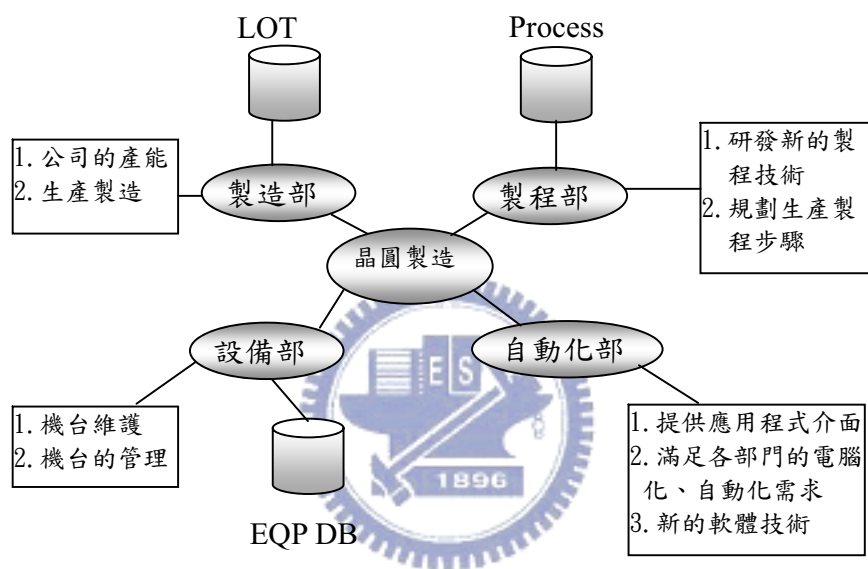


圖 9 晶圓製造與主要相關部門關係

1. 製造部門

在整個晶圓的製造過程中，製造部門是晶圓的生產製造的第一線，也是整個晶圓製造的主軸，背負著公司的產能，任何影響公司晶圓產能的因素，製造部門都要尋求製程部門、設備部門、自動化部門主動積極解決。製造部門在產能與製造的管理，使用晶圓資料庫(Lot DB，屬於 SQL server)系統。

製造部門會使用到 Lot DB 中的 Lotinfo 資料表從事資料查詢與更新，資料欄位、格式、大小、資料範例如表 1、表 2 說明：

表 1 lotinfo 資料欄位說明

欄位名稱	格式	大小	null	說明
Lotid	char	8	不允許	晶圓編號
Product	Char	4	不允許	晶圓產品編號
operation	char	4	不允許	目前晶圓所在的站點
Eqpid	char	10	允許	如果有進入機臺執行，機臺編號，否則是空的
Qty	char	4	不允許	晶圓的數量，通常是 24 片
Status	char	10	不允許	機臺的狀態(Q:queue, 未進入機臺執行;R:run 已進入機臺執行;P:已完成準備動作)
nextoper	char	4	不允許	下一個流程的站點，通常在製造結束後，至 Process DB 查詢，決定下一個流程。

表 2 lotinfo 資料資料範例表

Lotid	operation	Prod	eqpid	qty	status	nextoper
4810000	1004	P003		24	Q	1006
4820000	1001	P003	MARK.002	24	R	1002
4830000	1001	P003		24	Q	1002
4840000	1001	P003		24	Q	1002
4850000	1001	P003		24	Q	1002

2. 製程部門

負責研發新的製程技術，規劃生產製程步驟，晶圓的製程流程開始由製程部門統籌規劃，並將生產站點流程相關資訊(Operation Information)與站點順序(Operation Sequence)寫入至製程資料庫

(Process DB)，讓製造部門依站點順序從事生產製造，也讓設備部門參考站點流程相關資訊(包含溫度、溼度、相關氣體濃度等)調整機台參數。

製程部門負責維護 Process DB 中的 Operinfo(Operation Information，表 3、表 4)資料表與定義晶圓生產執行的步驟順序 Operseq(Operation Sequence，表 5、表 6)資料表，資料欄位、格式、大小、資料範例說明如下：

表 3 operinfo 資料欄位說明

欄位名稱	格式	大小	null	說明
Operation	char	4	不允許	站點名稱
Opertype	char	4	不允許	站點類型
operdesc	char	50	不允許	站點基本資料描述
ppid	char	50	不允許	定義站點的參數值

表 4 operinfo 資料範例表

operation	operdesc	ppid
1000	lasermark	laser001
4000	lasermark	laser004
4001	view	v004

3. 設備部門

設備部門是晶圓製造的主體，如果沒有設備部門提供生產製造機台，則晶圓將無法被生產製造。設備部門為了公司的產能，透過各種維護方式，讓機台維持在最好的狀態，並且要尋求製造品質更優良，製造速度更快速的生產機台供製造部門使用，共同為公司的產能貢獻心力。

表 5 operseq 的資料欄位說明

欄位名稱	格式	大小	null	說明
Operation	char	4	不允許	站點名稱
Prod	char	4	不允許	產品編號
seq	int	double	不允許	定義站點的先後執行步驟順序，並不是由站點的大小去決定先後執行順序

表 6 operseq 資料範例表

operation	prod	seq
1000	P003	1
1001	P003	2
1002	P003	3
1003	P003	4
1004	P003	5
1006	P003	6
1005	P003	7

機台的管理，是設備部門的重點，透過機台資料庫(Eqp DB，屬於 MS Access)的管理機制，管理工廠上百部的機台狀態，讓製造部門運用機台資料庫的資訊，分配晶圓的產能於各個機台。

設備部門負責維護機台資料庫(Eqp DB)中的 Eqpinfo 資料表，定義機臺所生產的型態、最大可執行晶圓數、機臺狀態與目前在機臺生產的晶圓數量，資料欄位、格式、大小說明、資料範例如表 7、表 8 說明：

4. 自動化部門

自動化部門屬於二線支援的角色，提供良好的應用程式介面，尋求
表 7 eqpinfo 的資料欄位說明

欄位名稱	格式	大小	Null	說明
eqpid	char	10	不允許	機臺編號
eqptype	char	4	不允許	機臺生產型態，此欄位與整合的資料庫中站點的所生產的型態有關
maxbatchsize	int	double	不允許	機臺最大可生產的晶圓數
eqpstatus	char	4	不允許	機臺目前的狀態(I：idle，機臺可以進行生產；R：run，機臺有晶圓正在執行生產；D：down，機臺有問題不能執行生產)
currnsize	int	double	不允許	目前在此機臺執行生產的晶圓數

表 8 eqpinfo 資料範例表

eqpid	eqptype	maxbatchsize	eqpstatus	currnsize
85DD..01	85DD	2	I	0
85DD..02	85DD	2	I	0
85DD..03	85DD	2	I	0
I12...01	DUVI	1	I	0

新的軟體技術，滿足各部門的電腦化、自動化需求。讓各部門能有良好的軟體工具能更有效率的管理各部門的資源運用，幫助公司藉著自動化部門的努力，排除人為的錯誤，節省成本，提升公司的競爭力。

3.3 機台生產流程

3.3.1 機台擺設位置與工程師作業區

在生產製造上，會有很多相同製程的機台可以使用，並將相同製程的機台依序做機台編號(Eqp ID)，如 DRS...01、DRS...02 等。在管理

上，將相同的製程的機台，歸為同型號(Type)的機台，同機台型號(epqtype)的機台遇到保養或當機導致該機台無法從事生產時，會互相支援使用。

在半導體無塵室裡，相同機台通常位置擺放會在同一區塊，機台主體設備擺設與生產製造現場會區隔開來。留機台操作面板與晶圓入口與出口，供製造工程師使用。這樣擺設會形成一個個的通道(tunnel)，如圖 10，並依序編號(如 A Tunnel、B Tunnel 等)，製造工程師可利用中央的走道，至不同的通道從事生產製造。

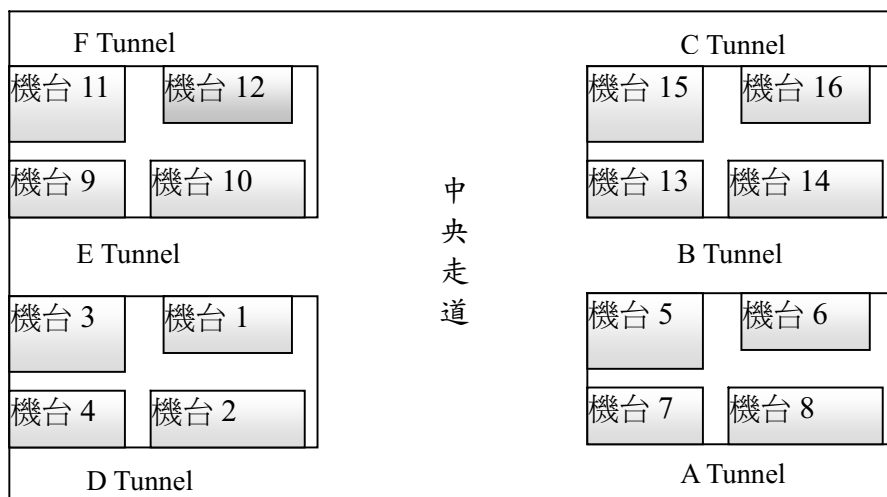


圖 10 無塵室機台擺設位置

3.3.2 機台生產流程

製造工程師負責晶圓的生產製造，會先檢查晶圓的批號、機台狀態是否確定要進入機台生產，之後操作機台讓晶圓進入進行生產製造，晶圓完成製程時，製造工程師將晶圓從機台退出，並送往下一個製程。晶圓一個站點(operation)完整製程會經歷如下三個步驟：

1. 製造準備(Job Prepare)

晶圓送入機台生產製造之前製造工程師要做的檢查工作，稱為製造準備(Job Prepare)。工作項目說明如下，當晶圓被送到要生產製程的機台前面，製造工程師在機台前的一部可以連上公司網路的電腦，執行由製造部門所提供的介面。此介面連結至製造部晶圓管理的資料庫(資料庫類型是 SQL Server)，輸入要生產的晶圓批號，依所得的資料欄位，如站點編號的製程類型(operatoin type)判斷是否是這一類型的生產機台。此批號的狀態(lot status)是否在準備好的狀態(Queue)，如果不是，就要與上一站點的製造工程師做確認等。

接著製造工程師，執行由設備部門所提供的介面。此介面連結至設備的機臺管理的資料庫(資料庫類型是 MS Access)，輸入執行生產的機台編號。依所得的資料的機台狀態(Eqp Status)的欄位去判斷，此機台是否可以進行生產。如機台狀態是當機(Down)，則無法在這台機台生產，必須要選擇其他同型機台的機台狀態是準備好(Idle)的狀態，進行生產。再檢查機台的可生產批號數量(Max batchsize)是否超過機臺最大的負荷等。確認無誤後，製造工程師將此批晶圓(lotid)的批號的狀態(lot status)透過製造部門所提供的介面更改為準備好(Prepare)的狀態。

2. 製造開始(Job In)

生產製造工程師將晶圓送入機台從事生產製造，稱為製造開始(JOB IN)。工作項目說明如下，執行製造準備(JOB PREP)與執行晶

圓送入機台製造開始(JOB IN)的生產製造工程師，可能不是同一位生產製造工程師。因此生產製造工程師需要再次透過機台設備部門所提供的介面確認機台狀態(Eqp status)是在準備好(Idle)的狀態、晶圓的製程站點(operation)、晶圓狀態是準備好(Prepare)的狀態等。確認無誤後，將晶圓送入機台，並在機台上按開始生產的按鈕，執行生產製造。

接著，製造工程師透過製造部門所提供的介面，將此批晶圓(lotid)的批號的狀態(lot status)更改為執行中(Run)的狀態，再透過設備部門所提供的介面把機台狀態(Eqp status)改為執行中(Run)的狀態。

3. 製造結束(Job Out)

晶圓從機台退出，完成製程程序，稱為製造結束(JOB OUT)。工作項目說明如下，製造工程師從機台將晶圓退出，再次確認晶圓有進入機台並完成製程程序。防止製造工程師只將晶圓送入機台，忘記按機台開始生產的按鈕，造成晶圓未執行該製程的生產，然後將晶圓送往下一個製程站點的錯誤發生。

製造工程師確認無誤後，透過製造部門所提供的介面，將此批晶圓(lotid)的批號的狀態(lot status)更改準備好的狀態(Queue)。製造工程師確認機台裏沒有任何晶圓從事生產製造，透過機台設備部門所提供的介面，確認機台狀態(Eqp status)是在準備好(Idle)的狀態，然後至製程資料庫查詢下一個站點製程，將晶圓送往下一個製程機台。

第四章 整合各部門的資料庫系統

製造部門、製程部門、設備部門使用相異的資料庫系統，各部門透過各自相異的應用程式介面，維護各自所屬資料庫的資料與提供資料給各部門參考使用。各部門所提供介面的應用程式的版本更新時，則需要對上百台電腦做重新安裝應用程式的動作。如果有些電腦沒有更換新的介面應用程式，則會出現舊的應用程式更改到舊的資料庫或是舊的應用程式更改新資料庫的資料不完整等，造成資料不一致的嚴重問題，或是舊的介面應用程式對應新的資料庫欄位不一致，導致電腦當機。

4.1 節描述各部門的需求與各部門資料庫關係，說明製造部門、製程部門、設備部門的需求與資料庫關係。4.2 節介紹晶圓流程控制系統(WFCMS)與各部門關係，說明整合各部門的資料庫系統與功能。4.3 節描述晶圓流程控制系統(WFCMS)與各部門人員作業模式，說明製造部門、製程部門、設備部門在 WFMCS 的作業模式。

4.1 各部門的需求與各部門資料庫關係

1. 製造部門需求

製造工程師主要工作從事晶圓的生產製造，為防止人為的疏忽，需要執行製程三步驟，如製造準備、製造開始、製造結束等，需要至各資料庫做資料檢查與更新。與各部門資料庫的關係，如圖 11。製造工程師需要透過製程部門提供的應用程式介面至 Process DB 查詢製程站點的先後順序等。需要透過設備部門提供的應用程式介面到 Eqp DB 更改或查詢機台狀態，並檢查機台的類型與製程站點類型是否一致等。需要透

過製造部門的應用程式介面至 Lot DB 查詢及更改晶圓的相關資訊。如上所述種種資料檢查動作，非常沒有效率，因此製造工程師希望自動化部門能提供一個應用程式系統，能查詢到需要的資料並能自動做更新，減輕製造工程師資料檢查的工作。

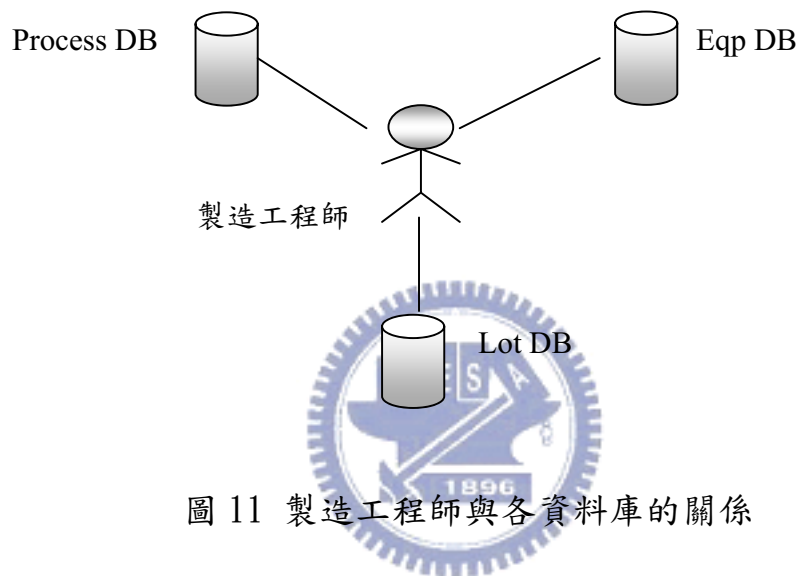


圖 11 製造工程師與各資料庫的關係

2 製程部門需求

製程工程師需要維護 Process DB 中的站點資訊 (Operation information) 資料表與站點順序 (operation Sequence) 資料表，如圖 12，提供給製造工程師與設備工程師參考。希望自動化部門能提供一個標準的介面，維護 Process DB 的資料。

3 設備部門需求

設備工程師需要維護 EQP DB 中的 Eqpinfo 資料表，如圖 13，提供給製造工程師參考。希望自動化部門能提供一個標準的介面，維護

Process DB 的資料。

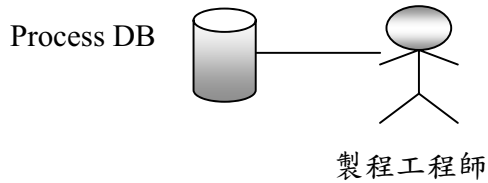


圖 12 製程工程師與各資料庫的關係

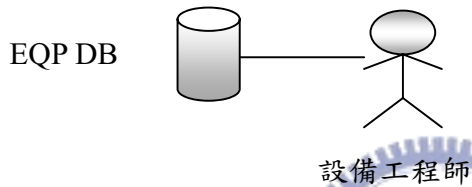


圖 13 設備工程師與各資料庫的關係

4.2 晶圓流程控制系統(WFCMS)與各部門關係

4.2.1 整合各部門的資料庫系統

晶圓流程控制系統(WFCMS, Wafer Flow control Management System)目的是要整合製造部門、製程部門、設備部門的異質資料庫，各部門使用統一的介面，去獲得所需的資訊，節省透過各自獨立且相異的操作介面的時間。晶圓流程控制(WFCMS)系統與各資料庫架構，如圖 14。製程部門資料庫系統(Process DB)屬於早期的資料庫系統，其資料庫廠商有提供 ODBC 介面，可以與資料庫溝通。製造部門資料庫系統(LOT DB)是

屬於 SQL Sever。設備部門料庫系統(EQP DB)是屬於 MS Access。UserDef DB 是屬於 SQL Sever，主要目的是管理登入 WFCMS 系統的使用者帳號、密碼。

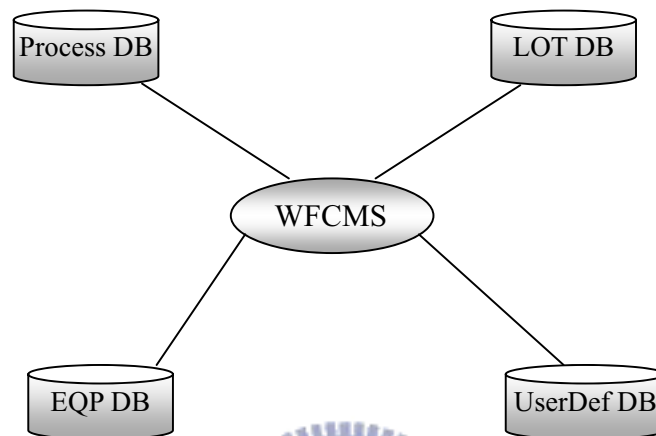


圖 14 晶圓流程控制(WFCMS)系統與各資料庫架構

當使用者登入 WFCMS 系統，依使用者的帳號所屬部門，給予相關的程式使用，UserDef DB 資料欄位、格式、大小、資料範例如表 9、表 10 說明：

表 9 userdef 的資料欄位說明

欄位名稱	格式	大小	Null	說明
Userid	char	15	不允許	定義使用者帳號
Passwd	char	15	不允許	定義使用者密碼
Role	char	15	不允許	定義使用者所屬的單位(PRO：製造部工程師、EQP：設備工程師、MOU：製程工程師)

表 10 userdef 資料範例表

userid	passwd	role
cchaun	123	PRO
jeffer	fgh	EQP
ken	1234	MOU

4.2.2 整合各部門功能系統

晶圓流程控制系統(WFCMS)依各部門需求，提供各部門使用的功能，分為共有的功能(Public function)及私有的功能(Private function)兩大類。如圖 15，共有的功能，每個部門人員都可以使用，包括站點查詢、站點流程查詢、機台狀態查詢、晶圓狀態查詢、離開等五種。私有的功能，提供給屬於該部門人員使用，如製造工程師有製造準備(JOB PREP)、製造開始(JOB IN)、製造完成(JOB OUT)、晶圓批號新增、晶圓批號更新、晶圓批號刪除等五種。設備工程師有機台新增、機台更新、機台刪除等三種。製程工程師有站點新增、站點更新、站點刪除、站點流程調整等四種。

4.3 晶圓流程控制系統(WFCMS)與各部門人員作業模式

4.3.1 使用者登入

各部門使用者登入晶圓流程控制系統(WFCMS)，輸入帳號及密碼，系統會透過 UserDef DB 驗證使用者，根據使用者帳號所屬的部門，給于該部門使用的程式介面，驗證使用者步驟，如圖 16。

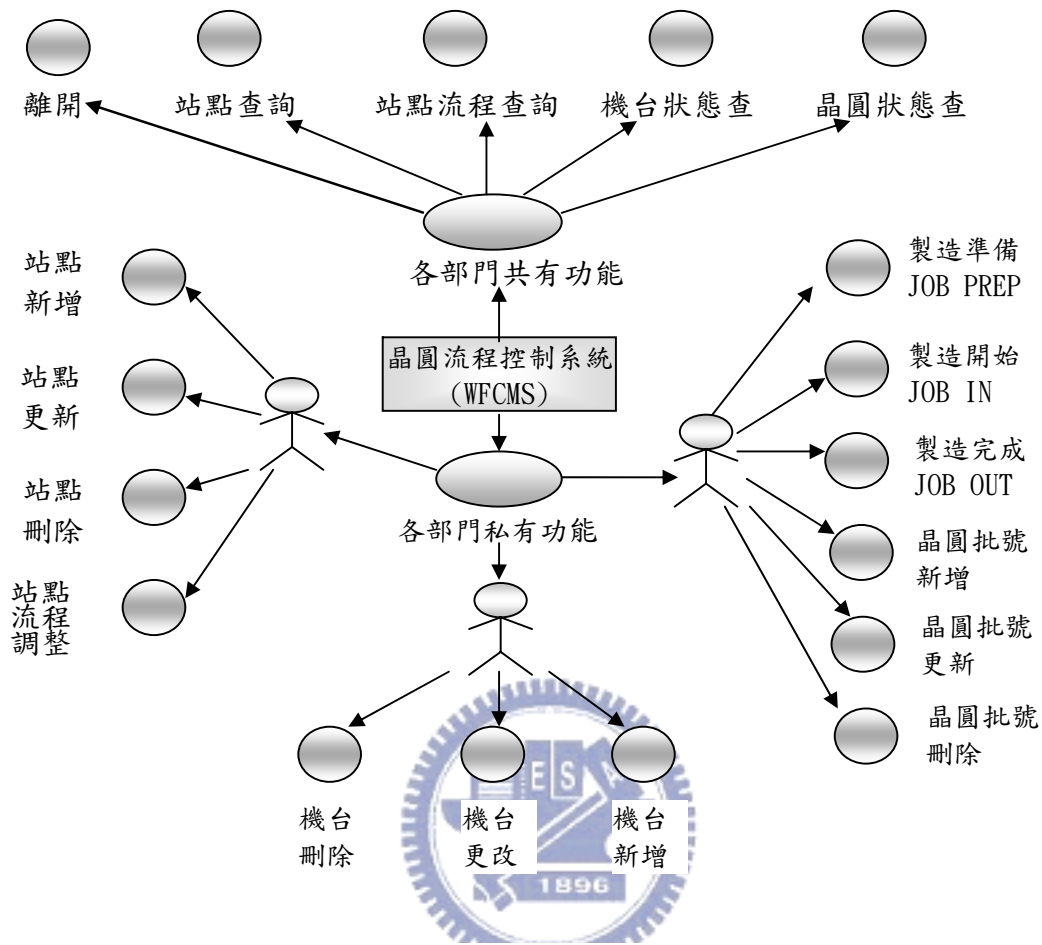


圖 15 晶圓流程控制系統功能圖

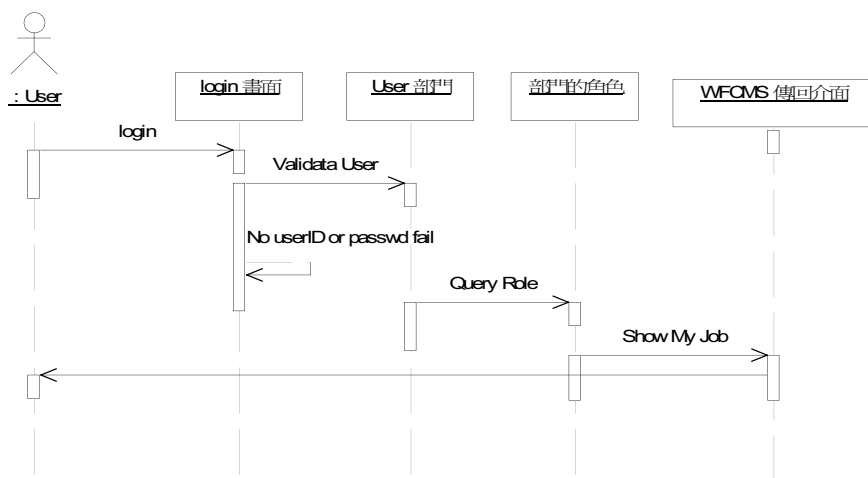


圖 16 驗證使用者身份順序圖

1. 登入(Log in)：使用者登入畫面，輸入帳號/密碼。
2. 驗證使用者(Validate User)：驗證使用者身份。
3. 錯誤訊息回傳(Error Return)：輸入 userID 或密碼錯誤則會回傳錯誤訊息。
4. 查詢部門(Query Role)：查詢使用者所屬部門。
5. 顯示功能介面(Show My Job)：將使用者可使用的功能回傳給使用者。

4.3.2 製造部門作業模式

- 製造準備(JOBPREP)作業流程

製造工程師使用晶圓流程控制系統(WFCMS)的製造準備(JOBPREP)每個步驟作業流程，如圖 17 說明。

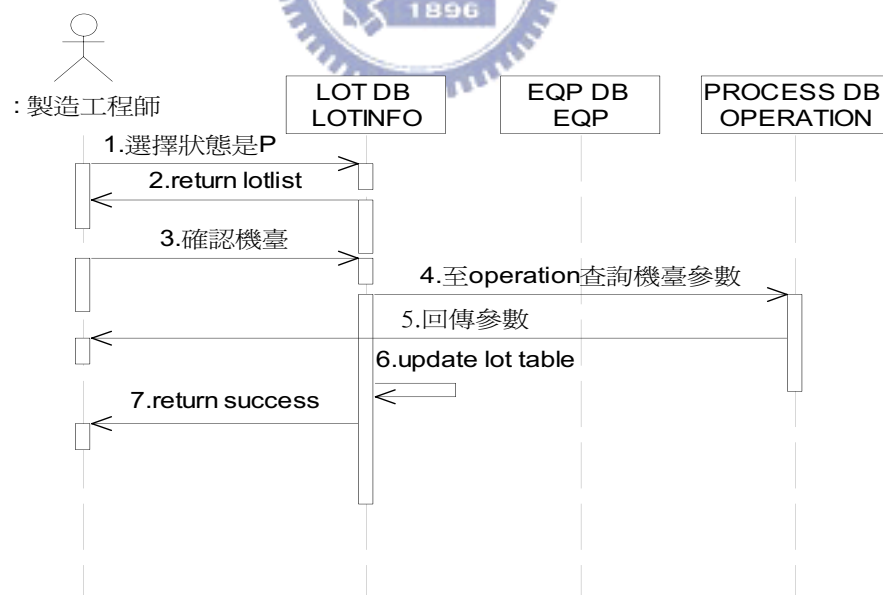


圖 17 JOBPREP 作業流程順序圖

1. WFCMS 至製造資料庫的 lotinfo 資料表，選擇晶圓批號(LOTID)的狀態是準備好(Queue)。
 2. 回傳可以選擇生產的晶圓批號列表(lotlist)，供製造工程師選擇。
 3. 依據製造工程師選擇晶圓批號的站點，至製程資料庫的站點(operatoin)資料表，查詢其站點類型(operation type)。
 4. 依據站點類型(operation type)，至設備資料庫的機台(eq)資料表，選擇有相同機臺類型(eq type)的機臺。
 5. 回傳相同機臺類型(eq type)的機臺列表，顯示在介面供製造工程師選擇。
 6. 製造工程師選擇適合的機臺，並確認。
 7. 更新設備資料庫的機台(eq)資料表的相關資料，例如將 currnsiz 欄位的值加一。
 8. 更新製造資料庫的 lotinfo 資料表的狀態(status)欄位為準備好(JobPrep)。
 9. 顯示已製造準備(JOBPREP)成功的相關訊息。
- 製造開始(JOB IN)作業流程

製造工程師使用晶圓流程控制系統(WFCMS)的製造開始(JOB IN)每個步驟作業流程，如圖 18 說明。

1. WFCMS 至製造資料庫的 lotinfo 資料表，選擇晶圓批號(LOTID)的狀態是已準備好(P)。
2. 回傳可以選擇晶圓批號的列表(lotlist)，提供製造工程師選擇。
3. 製造工程師確認機台狀況。

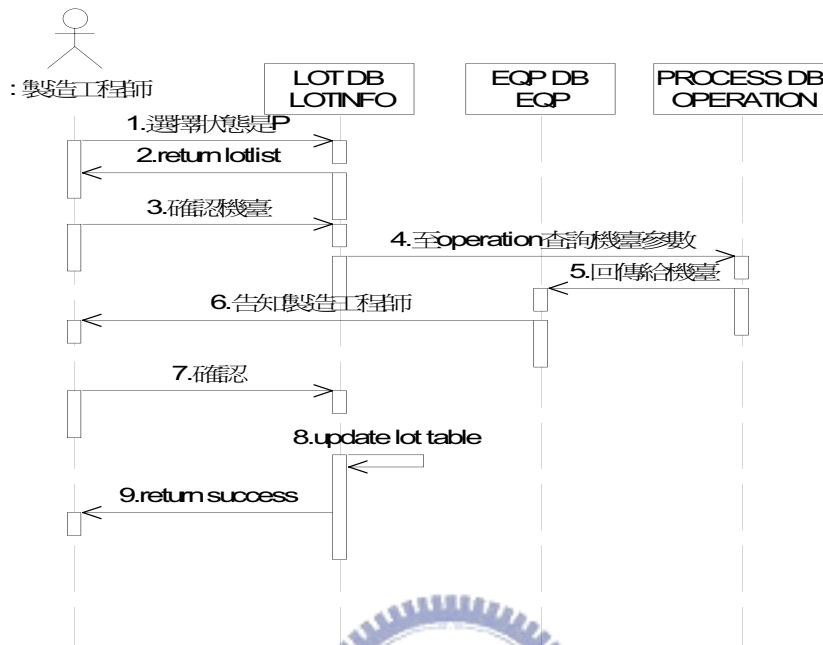


圖 18 JOB IN 作業流程順序圖

4. 至製程資料庫的站點(operation)資料表，查詢機臺參數。
5. 回傳給機臺使用。
6. 機台告知製造工程師，可以將晶圓放入機台生產。
7. 製造工程師確認，將晶圓放入機台生產。
8. 更新製造資料庫的 lotinfo 資料表的晶圓狀態(status)為執行 (Run)狀態。
9. 顯示已製造開始(JOB IN)成功的相關訊息。

- 製造結束(JOB OUT)作業流程

製造工程師使用晶圓流程控制系統(WFCMS)的製造結束(JOB OUT)每個步驟作業流程，如圖 19 說明。

1. WFCMS 至製造資料庫的 lotinfo 資料表的狀態是 R(Run)的晶圓批號。
2. 回傳可以選擇晶圓批號的列表(lotlist)，供製造工程師選擇。
3. 製造工程師確認生產過後機臺狀況及晶圓狀況。
4. 至製程資料庫中的站點資料表，取得下一個站點流程相關資料。
5. 更新製造資料庫的 lotinfo 資料表的站點欄位資料為下一站點的站點。
6. 更新設備資料庫中的 EQP 資料表資料，如將 crrunsize 減一、機臺如果沒有貨生產，將該機臺的狀態更新為等待(Idle)。
7. 更新製造資料庫的 lotinfo 資料表的 eqp 欄位為空白。
8. 顯示已製造成功(JOB OUT)的相關訊息，將晶圓往下一個流程傳送。

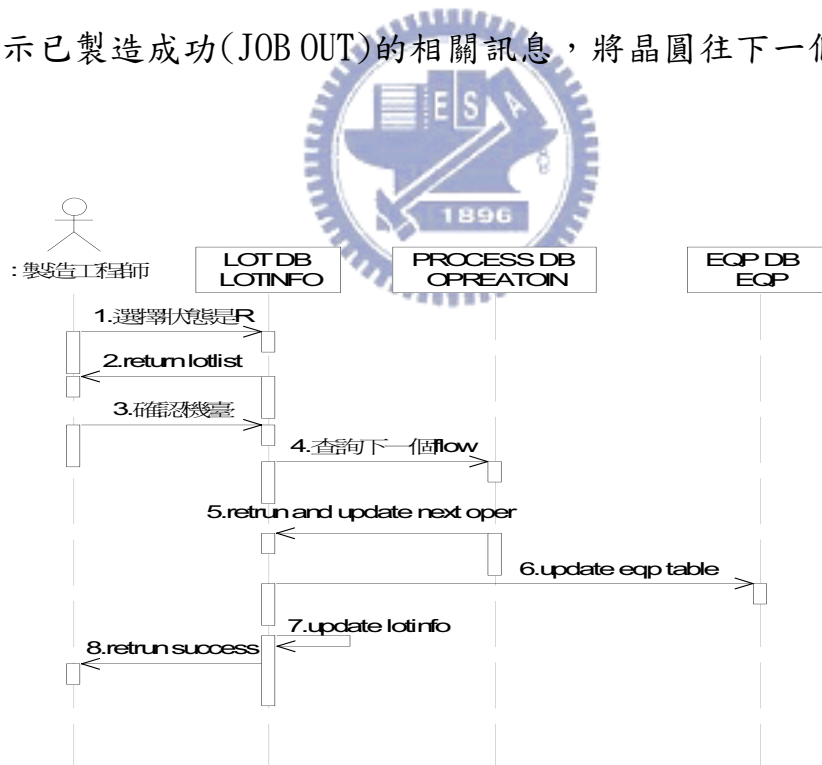


圖 19 JOB OUT 作業流程順序圖

- 製造部門與各部門資料庫的欄位關聯性

製造部門與各部門資料庫的關係，如圖 20，主要以 LOT DB 的 lotinfo 資料表為主要的資料來源。lotinfo 資料表以 lotid、operation 欄位為主要鍵(primary key)。lotinfo 資料表的 operation 欄位會參考到製程資料庫的站點(operation)資料表的站點(operation)及站點流程(operation seq)資料表的 operation 欄位。lotinfo 資料表的 eqpid 欄位會參考到設備資料庫的機台(eq)資料表的機台(eqpid)欄位。設備資料庫的機台資料表中的機台類型(eqtype)與製程資料庫的站點流程(operation seq)資料表的 opertype 欄位類型要一致，才有機台可以選擇。

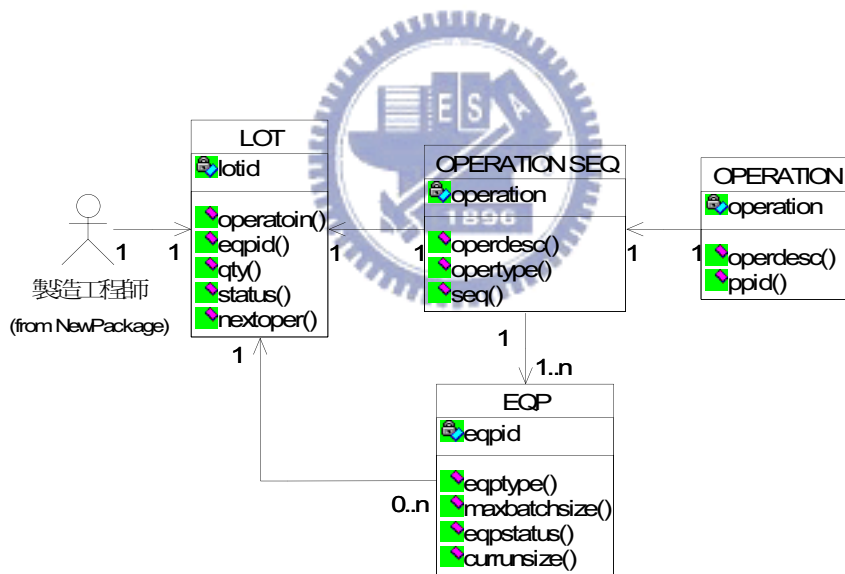


圖 20 製造部門與各部門資料庫的

4.3.3 製程部門作業模式

製程部門透過 WFCMS 維護 process DB 的 operation 及 operation

seq 資料表，如圖 21。製程工程師會先建立 operation 資料表的站點資料，依據站點資料，建立站點流程資料表的站點順序(seq)欄位，最後定義站點流程資料表中的站點製程種類。

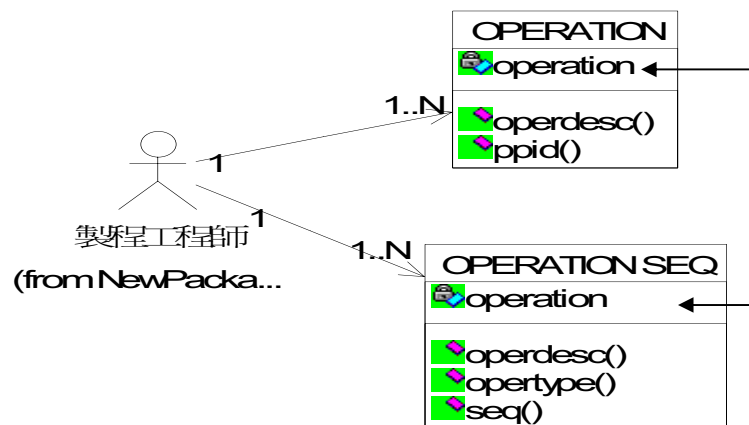


圖 21 製程部門與資料庫的關係

4.3.4 設備部門作業模式

設備部門透過 WFCMS 維護 EQP DB 的機台(EQP)資料表，如圖 22。設備部門要以製程部門所定義的站點類型，維護機台(EQP)資料表的機台類型(eqptype)，並定義機台(EQP)資料表的機台最多同時生產的晶圓批號數量(maxbatchsize)。

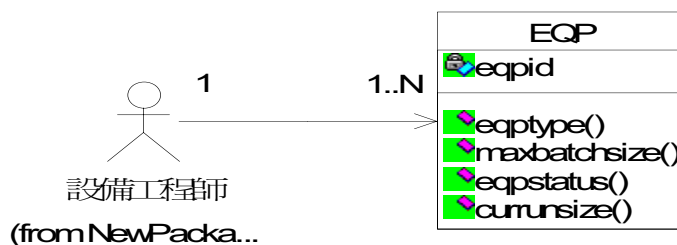


圖 22 設備部門與資料庫的關係