第貳章 文獻探討

本研究以服務導向架構與事件導向架構為基礎，提出一套商業事件(business event)管理與事件結合服務提供的系統機制，因此以下將針對此架構的相關文獻進一步探討：第一節將探討服務導向架構與其應用，第二節將探討事件導向架構與其應用，第三節將綜合探討兩種架構的相異點與結合應用上的不足之處，並提出本研究的解決方案。

第一節　服務導向架構(Service-Oriented Architecture)

所謂服務導向架構(SOA)是指將各別的程式或系統視為服務，利用共通的服務介面進行溝通，以達到整合與重覆使用服務的一種應用程式架構(TIBCO,2004)。此種架構主要是為了解決傳統分散式應用程式缺乏統一溝通標準的問題，而由於以 Web Service 為介面的服務導向架構利用 HTTP、XML、SOAP 和 WSDL 這些 Internet 上的標準技術，所以可以實現跨平台、跨語言的整合目的(李清培，2003)，所以也是目前實作服務導向架構中最被廣泛接受的實作技術。

服務導向架構的基本架構圖主要由服務需求者(Service Consumer)、服務提供者 (Service Producer)及註冊服務者(Service Registry) 彼此互動而形成(如圖2-1-1)，而運作過程為服務提供者可至註冊服務者處公開註冊(Publish)服務；服務需求者可至註冊服務者處尋找(Find)服務，當找到適當的服務時，可以呼叫(Invoke)服務提供者所提供的服務(K. Mani Chandy and Jonathan Lurié Carmona and Robert Alexander,2005)。

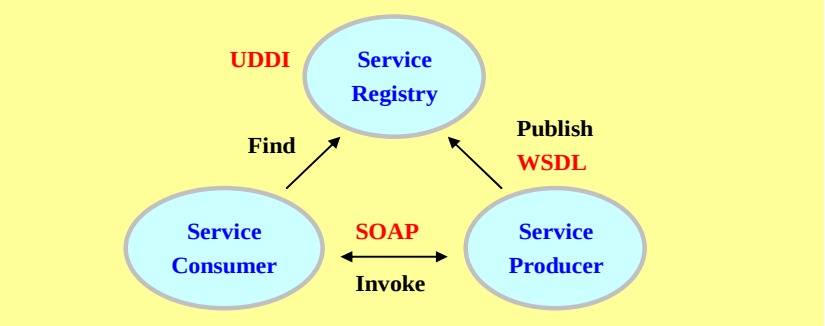

圖 2-1-1　SOA 基本架構圖與 Web Service 實作技術

服務導向架構設計具有以下特性(Michael Stevens, 2002；Michael Stevens, 2005)：

- Discoverable and Dynamically Bound：服務具有可搜尋性(discoverable)，可利用服務註冊者搜尋服務，也能動態連結(dynamically bound)服務。
- Modular：服務可被模組化(modular)，主要由介面(interface)與實作(implementation)兩部分組成。
- Interoperability：服務可跨平台與跨語言互相溝通(interoperability)。
- Loose Coupling：利用註冊服務與服務合約(contract)的機制，使服務需求者與服務提供者之間的關係延遲發生，達到服務 loosely coupled 的目的。
- Network-Addressable Interface：服務有可網路存取的介面(network-addressable interface)。
- Coarse-Grained Interfaces：服務有 coarse-grained 的介面。
- Location Transparency：服務有位置透明性(location-transparent)。
- Composability：服務具有組合性(composability)。

由於服務導向架構具有以上的特性，所以此架構有以下優點(Kishore Channabasavaiah and Kerrie Holley and Edward M. Tuggle, Jr., 2004)：

1. 提高服務的重複利用性。
2. 服務具有組合性，可提高系統的彈性與靈活度。

3. 統一系統架構，藉此降低整合與維護系統的成本與時間。

若以 Web Service 實作服務導向架構角度來看，其實作方式如下(如圖 2-1-1)：服務提供者可將服務用 WSDL 描述，並將服務描述註冊於採用 UDDI 規範的註冊服務者處，而當服務需求者需要服務時可利用註冊服務者處的服務描述尋找合適的服務，並在找到服務後以 SOAP 的溝通方式要求服務提供者提供服務，而目前 Web Service 的應用範疇主要分為三大類(李清培，2003)：

1. 整合企業內部系統：利用 Web Service 整合企業內部系統，強化內部商業流程的效率與降低成本，最典型的是用在企業應用整合(EAI)上或利用 Web Service 延伸舊有應用程式功能或提供 Internet 的存取介面，以減少開發網際網路應用程式所需的時間與成本。
2. 整合企業間的系統：利用 Web Service 處理 B2B 的系統溝通或交易，Web Service 比目前的 EDI 應用程式簡單，大幅降低了程式的複雜性，而且架構更為標準與彈性。
3. 提供軟體租用新模式：服務提供者可利用 Web Service 提供個人或公共服務，比如 Google 所提供的搜尋 Web Service 可與應用程式結合，而服務需求者可直接利用 Internet 上的 Web Service，以軟體積木的方式選取所需的服務結合現有的系統應用，藉此加快系統的開發速度與降低成本，此方式尤其適用於輔助性的功能或共通的商業智慧的服務。

第二節　事件導向架構(Event-Driven Architecture)

所謂事件導向架構(EDA)是指利用事件通知的方式，告知事件訂閱者所關心的訊息，讓事件接收者可以更快採取行動(TIBCO,2004)。一般實作事件導向的系統均使用 publish/subscribe 的模式(蔡澤銘、張均合、吳欽誠、廖俊傑，2005)，其中最基礎的 publish/subscribe 系統互動模式如下(如圖 2-2-1)： (Patrick Th. Eugster and Pascal A. Felber and Rachid Guerraoui and Anne-Marie Kermarrec , 2003)

1. 事件發出者(event publisher)先向事件服務中介者(event service)刊登(publish)所提供的事件。
2. 事件訂購者(event subscriber)可向事件服務中介者(event service)訂閱事件發出者(event publisher)所提供的事件，而事件服務中介者(event service)負責管理訂閱事件的資訊。
3. 當某事件發生時，事件發出者(event publisher)將事件傳送至事件服務中介者(event service)，而事件服務中介者(event service)會將事件儲存起來，並盡快通知(notify)有訂閱此事件的事件訂購者(event subscriber)。

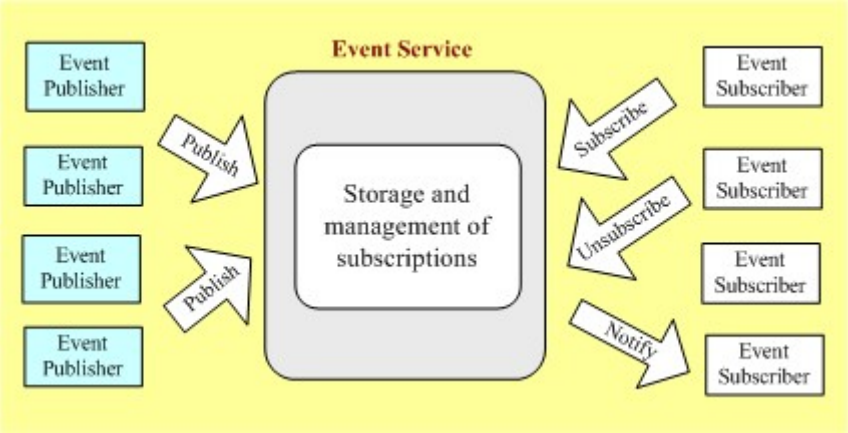

圖 2-2-1　簡單的 object-based publish/subscribe 系統

透過事件服務中介者(event service)可使 publish/subscribe 系統具有以下特性：(Patrick Th. Eugster and Pascal A. Felber and Rachid Guerraoui and Anne-Marie Kermarrec , 2003)

1. Space decoupling：事件發出者(event publisher)只需將事件資訊刊登在事件服務中介者(event service)，不需知道有哪些事件訂購者(event subscriber)，而事件訂購者(event subscriber)向事件服務中介者(event service)訂閱事件，也不需知道知道事件發出者(event publisher)的資訊。
2. Time decoupling：事件發出者(event publisher)和事件訂購者(event subscriber)不需同時在線上才能傳遞事件，因為事件服務中介者(event service)會負責儲存事件並將事件傳遞給事件訂購者(event subscriber)。

3. Synchronization decoupling：事件發出者(event publisher)在發出事件後可以繼續執行工作，不需等待事件訂購者(event subscriber)的回應，而事件服務中介者(event service)會主動通知事件訂購者(event subscriber)有事件發生，所以事件訂購者(event subscriber)也不需等待回應。

事件訂購者(event subscriber)在訂閱事件時，通常會對特定的事件或事件模式(event pattern)感興趣。目前可將 publish/subscribe 的訂閱事件的描述分三種類型：(Patrick Th. Eugster and Pascal A. Felber and Rachid Guerraoui and Anne-Marie Kermarrec , 2003)

1. Topic-based publish/subscribe：事件依主題(topic)訂閱，而主題同時也代表是一個群組。若事件訂購者(event subscriber)訂閱某主題，就代表願意加入群組。當事件發出者(event publisher)對此主題發出事件時，則有訂閱此主題的事件訂購者(event subscriber)就會接收到事件通知，但此種訂閱描述過於簡略，因此無法提供適合不同事件訂購者(event subscriber)的事件篩選能力。
2. Content-based publish/subscribe：事件依內容訂閱，事件訂購者(event subscriber)可用語法描述事件內容的條件，唯有當事件內容符合條件時，才會通知事件訂購者(event subscriber)，雖然此種訂閱描述可提供事件訂購者(event subscriber)較符合需求的事件，但常導致處理事件的演算法(algorithm)過於複雜，影響事件服務的效能。
3. Type-based publish/subscribe：事件依類型分類，甚至可以用階層的方式表達事件類型，此種方式是為了改善 Topic-based 訂閱描述過於簡單的問題，也是三種類型中最新的一種事件訂閱描述類型。

基於以上 publish/subscribe 系統的基本概念，企業也結合 publish/subscribe 的概念逐漸發展出不同類型的事件導向系統的應用。Roy Schulte(2004)將商業事件應用系統由簡單到複雜分為四類(如圖 2-2-2)：

1. Simple Events Application：指傳送與接收方直接以事件或訊息互相溝

通，溝通的方式可透過 message-oriented middleware(MOM)[1] (Cory Vondrak and Redondo Beach, 2004)、SOAP/HTTP、e-mail 或其他方式 (Patrick Th. Eugster and Pascal A. Felber and Rachid Guerraoui and Anne-Marie Kermarrec , 2003)，為四大類型中最簡單的應用類型。

2. Mediated Events Application：指傳送方透過整合中介者(integration broker)或 Enterprise Service Bus(ESB)[2](IBM, 2004)傳送訊息給接收方，而整合中介者或 ESB 具有轉換訊息的資料形式(transformation)與 content-based routing 的能力，也可以支援事件導向架構(SOA)。
3. BPM Events Application：指傳送與接收方透過 business process management(BPM)[3](David Kelly,2005)的流程管理機制傳送與接收事件。BPM 可管理程序(process)的流程，並記憶每個程序(process)的處理狀態，而在接收到事件後，可利用規則(rule)判斷目前流程的狀態，並決定應該執行的下一個步驟，比如可能是觸發(trigger)某個活動(activity)、副程序(subprocess)或服務執行還未完成的任務。BPM 除了具有中介者觸發(trigger)應用程式的功能外，它也可以利用 BPM 導向類型的事件在 BPM adapter 與 central BPM 之間溝通，另外，也支援事件導向架構(SOA)。
4. Complex Events Application(CEP)(David Luckham, 2005)：指利用一個集中式的事件管理者(event manager)或多個分散式的事件處理者(event processing agent, EPA)管理事件，藉此辨識事件的關係與使用者所重視的事件模式(event pattern)，進而形成不同層級的事件(event

[1] Message-oriented Middleware(MOM)是一種提升互動性、可移動性和彈性的 client/server 中介軟體(middleware)概念，主要的功能為提供 API 整合異質平台的系統，藉此達到系統間互相溝通的目的。

[2] Enterprise Service Bus(ESB)是一種能夠整合異質平台系統的應用軟體概念，同時可提供 routing、transformation、mediation、security 等功能，也支援服務導向架構(SOA)。

[3] Business Process Management(BPM)是一種利用流程導向的方式整合與管理系統與資源的應用軟體概念，可達到流程整合、流程自動化、流程管理、流程最佳化與增加流程彈性等目的。

pattern abstraction)，以便企業內的人員作決策之用，其主要功能包含篩選(filter)事件、聚集(aggregate)事件和偵測事件是否違反規則，也可以結合 Business Activity Monitoring(BAM)[4](David Luckham,2004; David Luckham,2005)設計。

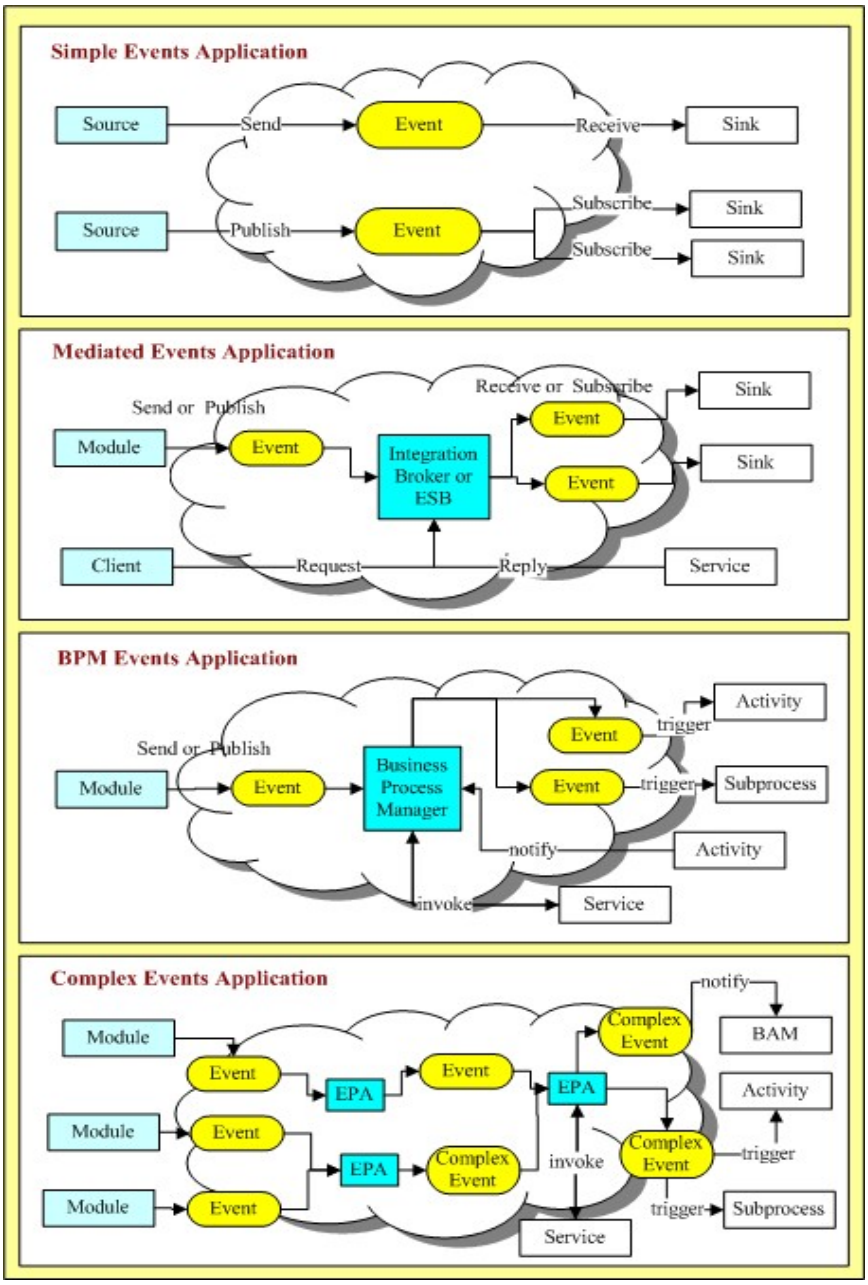

圖 2-2-2　商業事件應用系統類別圖

[4] Business Activity Monitoring(BAM)是一種可以即時監控系統情況並處理系統問題的應用軟體概念，主要是透過有效處理複雜事件的機制，讓企業可針對問題快速採取反應。

下圖是2004年新的大型應用系統中理論上與實際上應用事件導向架構(EDA)的情況，其中以簡單事件應用的比例最多，越複雜則應用愈少(如圖2-2-3)，而學者Peter Fingar(2005)認為未來企業流程管理(business process management)發展的五大方向，包含了發展企業流程管理系統與Complex Events Application，由此可見，系統處理複雜事件的能力將隨時間逐步提升。

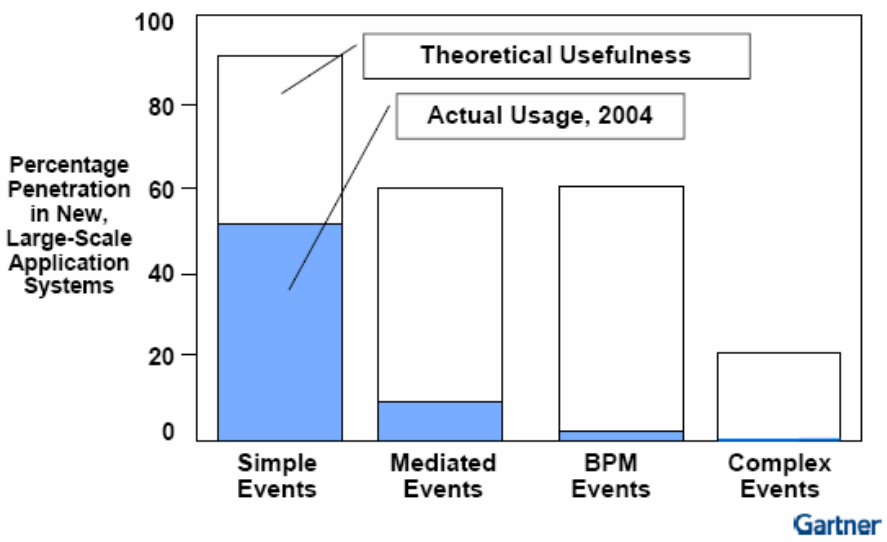

圖 2-2-3　2004 年新的大型應用系統中理論上與實際上應用 EDA 的情況(source: Gartner)

以下是幾個事件導向系統應用的實例(Diana Walker,2004)：

- 訂單生產(build to build)的製造系統：將訂單化為事件的形式呈現，並利用事件觸發取得所要求的零件。
- 網路系統管理：當有網路問題發生時，網路失敗(failure)的訊息會以事件的方式觸發系統發出警告訊息。
- 商業活動監督系統(BAM)：多個相關的商業事件結合(combine)後可以產出即時的分析報告，例如：多筆訂單可以結合分析出即時的分析報告，提供給決策者分析。
- 自動的股票交易系統：當股票價格高於或低於某個數值的時候，可以自動觸發事件購買或賣出股票。

第三節　服務導向架構與事件導向架構的結合

Roy Schulte(2004)與 TIBCO 一致認為服務導向架構與事件導向架構是兩種相容且互補的架構，利用兩架構的相異特質可以補強單一架構的不足之處(如表 2-3-1) (Fiona Chau, 2004)，也將是未來 Real-Time Enterprise 應用軟體架構的主流趨勢。

表 2-3-1　SOA 與 EDA 的相異點(Source: Tibco Software and Gartner Research)

SOA	EDA
Demand-services based (e.g. request for information) i.e. mostly synchronous	Based on real-time events, mostly asynchronous and unpredictable, typically models are long-running business processes
Clients and servers loosely coupled	Clients and servers decoupled
One-to-one request/reply (Pull)	Many-to-many publish/subscribe (Push)
Bi-directional, but no delivery guarantee	One-way guaranteed event delivery
Ideal for sequential or straight through processing and composite applications where all associated systems and services are available	Ideal for parallel processes, events (triggers and joins) and exception handling across disconnected systems
Uses interface metadata	Uses event descriptor metadata
Client directs flow	Sink (recipient) determines flow
Closed to unforeseen input once a flow is started	Can react to new external input while process is in flight

在事件導向架構(EDA)方面，本研究認為目前的事件導向架構(EDA)應用系統有以下幾點不足之處：

1. 事件的篩選能力不足：publish/subscribe 系統的事件訂閱描述中，Topic-based 的訂閱類型過於簡單且為 domain dependent，無法為不同事件訂閱者(event subscriber)提供有效的事件篩選能力；Content-based 的訂閱類型雖然可以提供事件訂閱者(event subscriber)較精確與符合需要的事件訂閱描述，但容易使得篩選事件的功能過於複雜，且此類的訂

閱描述多為 domain dependent，無法用來篩選不同 domain 的事件；Type-based 的訂閱類型雖然改善了 Topic-based 訂閱類型過於簡單的問題，但仍為 domain dependent 的訂閱描述，同樣也無法用來篩選不同 domain 的事件。雖然目前事件導向架構(EDA)應用系統中，BPM Events Application 和 Complex Events Application(CEP)提供事件篩選的功能，但篩選條件仍屬於 domain dependent 的描述，無法適用於大多數的事件，因此本研究認為在利用事件通知機制強化資訊收集能力之餘，還必須提昇分辨事件重要性的能力，因為若無法有效分辨真正重要的訊息，則會導致 false alarm 或忽視具有威脅性的訊息，造成額外的困擾與風險。

2. 未提供智慧型調適事件篩選規則的能力：事件訂閱者 (event subscriber)的事件篩選條件會隨時間而改變，但目前的事件導向應用系統尚未提供主動分析並更新事件訂閱者 (event subscriber)所偏好的事件篩選條件，所以事件訂閱者(event subscriber)必須主動更新篩選條件，才能持續確保接收到較符合需求的事件通知。
3. 未提供自動訂閱或取消事件的管理機制：現有的事件導向應用系統中，仍未提供有效管理事件訂閱與取消的功能。一旦事件訂閱的數量增多時，將加重事件訂閱者管理事件的負擔與困擾，使事件通知機制的效益大打折扣。

針對以上問題，本研究提出一 domain independent 的事件處理機制來管理事件訂閱者所訂閱的事件。在篩選事件的能力方面，本研究從經濟學、心理學與社會學的觀點提出事件可用其所提供的價值來衡量對事件訂閱者(event subscriber)的重要性，也就是利用事件價值(Value-driven)的觀點來一般性的(generic)衡量事件對於事件訂閱者(event subscriber)的價值與效益，讓事件可以依事件訂閱者(event subscriber)重視的程度篩選，藉此增強 domain independent 的事件篩選的能力，補足現有事件導向應用系統中事件訂閱描述與篩選事件能

力不足的問題；在更新事件篩選規則的能力方面，本研究設計智慧型的 Agent 學習事件訂閱者(event subscriber)採用事件的行為，並歸納分析出更適切的事件過濾規則，藉此主動更新事件訂閱者(event subscriber)的事件篩選條件；在自動訂閱或取消事件的管理機制方面，本研究設計能夠觀察使用者採用事件行為的 Agent，自動建議事件訂閱者(event subscriber)取消事件訂閱。綜合以上的事件管理機制，期望能設計一個更貼近使用者需求的應用系統。

在服務導向架構(SOA)方面，本研究認為此架構雖然具有整合系統的能力，但當使用者需要服務時，必須主動要求服務而非系統主動提供服務，所以此架構若能藉助事件機制主動促成服務的提供，則可提升系統的整體服務效能，因此，本研究設計一事件與服務對應的 Agent，使系統能適時並主動的提供服務。

綜合以上論點可看出，此兩種架構必須有良好的搭配設計，才能真正完全發揮服務導向架構與事件導向架構結合應用的互補優勢，因此本研究以事件管理機制為核心，輔以服務導向架構的設計，期望結合事件與服務提供更兼具彈性與效用的 Real-Time Enterprise 系統架構設計。