第四章、中介軟體之技術討論

在國內 RFID 應用尚不普遍的情況下，為解決物流業初期導入 RFID 技術，且硬體設備不提供軟體服務之因難，本研究以設計一應用程式介面為發展概念，發展處理 Reader 資料的介面程式，使前端 Reader 所讀取到之資料，透過介面軟體，得以轉換輸入後端資訊系統，如 ERP，WMS 等後端系統的應用程式或資料庫，藉此達成與後端系統或服務串接之目的。程式撰寫完成後，並將選取適當之第三方物流中心之現有資訊系統，進行整合性的測試與評估。以下分別就中介軟體之特色、系統架構、及發展階段加以探討，做為開發應用程式介面之基礎。

4.1 中介軟體之特色

一般來說，RFID 中介軟體 具有下列的特色[5]：

1. Insulation Infrastructure

   RFID 中介軟體獨立並介於Reader 與後端應用程式之間，並且能夠與多個 Reader 以及多個後端應用程式連接，以減輕架構與維護的複雜性。

2. Data Flow

   RFID 主要目的在於將實體物件轉換為資訊環境下的虛擬物件，因此資料處理是RFID 最重要的特徵，而中介軟體具有資料的蒐集、過濾、整合與傳遞等特性，以便將正確的物件資訊傳到企業後端的應用系統。

3. Process Flow

   RFID 中介軟體 採用程序邏輯及儲存再轉送（store-and-forward）的功能來提供循序的訊息流，具有資料流程設計與管理之能力。

4. Standard：

   RFID 為自動資料擷取技術與辨識實體物件的應用，EPCglobal 目前提出全球唯一識別號碼通用標準EPC。EPC 存放在Tag 中，被Reader讀出後，即可提供追蹤EPC碼所代表的物品名稱及相關資訊，並立即識別及分享供應鏈中的物品資料，有效率地提供資訊透明度。

因應EPC的制定，未來在RFID 中介軟體的發展也需要納入標準資料之處理，Auto-ID Center[8]於2003年9 月發表「Auto-ID Savant Specification 1.0」作為中介軟

體技術規範架構，該技術規範針對RFID 應用EPC 的議題，訂出EPC Network技術架構，如圖20所示。

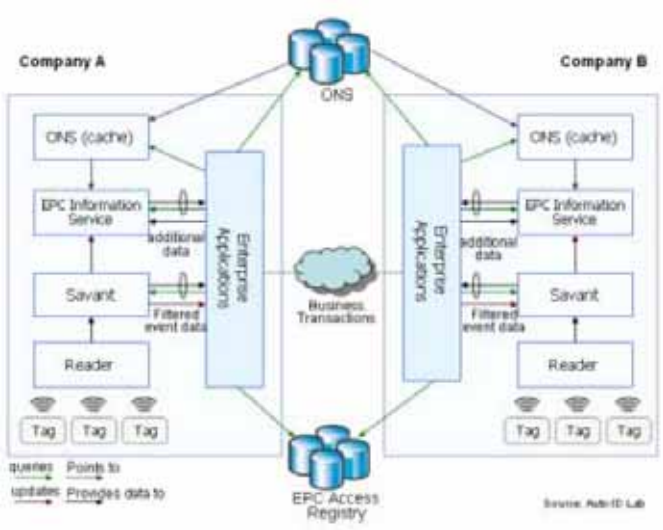

圖 20、EPC Network 技術架構

資料來源：[8]

在技術面，如讀取率，未克服之條件下，本研究認為業者應先建構其資訊系統，培養建置 RFID 環境。而在通訊協定未明朗化下，及各國標準尚未統一的情況下，EAN-UCC 轄下的 Auto-ID Center 推出了 RFID 之 EPC 標準，供商務鏈等產業使用，將更進一步促使 RFID 被大量使用。因此本研究嘗試在制定通訊標準的大環境下，EPC 應用於整體供應鏈，將實體商品從入庫、儲存、出貨至銷售或退貨等的動態資訊，透過中介軟體，與第三方物流公司現有之 ERP 的資訊整合，提升 WMS 整體功能。

4.2 中介軟體系統架構

根據 Auto-ID Center 提出的中介軟體系統架構，將RFID 中介軟體區分為 Reader Interface、Processing Module、及Application Interface。各模組(如圖21)的功能說明如下[11]：

1、Reader Interface

- 提供RFID Reader硬體與中介軟體之連接介面。
- 負責Reader Adapter與後端軟體之通訊介面，並要能支援多種Reader Adapter。

- 能夠接受遠端命令，控制Reader Adapter。

2、Process Module

- 在系統管轄下，能夠觀察所有Reader的狀態。
- 提供Processing Modules向系統註冊的機制。
- 提供EPC code和 non-EPC code轉換的功能。
- 提供Reader管理的功能，例如新增、刪除、停用、群組…等功能。
- 提供過濾不同Reader所接收內容的功能。

3、Application Interface

- 透過一致的XML-RPC/SOAP-RPC溝通方式。連接企業內部既有資料庫（如存貨系統)或EPC相關資料庫，使外部應用系統可透過此RFID 中介軟體取得相關EPC/nonEPC資訊。

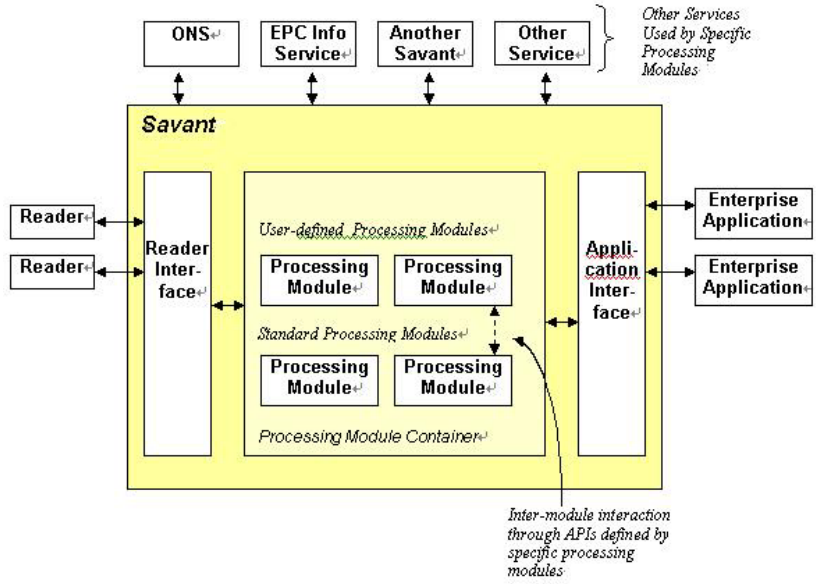

圖 21、RFID 中介軟體架構圖

資料來源：[11]

4.2.1 處理模塊的介紹

Savant 定義為一系列特定屬性的「處理模塊」（Processing Modules）及「服務」（Services）來定義，再配合用戶的需要來滿足對程式的應用。Savant 本身就是一個包含很多處理模塊的箱子，如圖 21 所示，Savant 包括有與 Reader 連接的 Reader 介面、企業應用程式介面，中間包含一些應用的處理模塊，包括滿足用戶需要的用戶定義處理模塊、基本處理應用模塊等。模塊除了連接著兩個介面間的

應用外，也可以被其他的服務所連接，如兩個 Savant 間的連接，以達到資料處理後再傳至另一 Savant 的作業。

處理模塊是由 Auto-ID Center 或用戶及第三方所定義的，其中由 Auto-ID Center 所定義的模塊為標準處理模塊（Standard Processing Module），是每一個 Savant 均需要提供的模塊；另外，一些標準模塊需要應用在 Savant 的所有應用模式中，這種模塊叫做必需標準處理模塊（Required Standard Processing Module）；其他一些可以根據用戶定義包含或者排除于一些具體實例中，這些就叫做可選標準處理模塊（Optional Standard Processing Module）。在 Auto-ID Savant Specification 1.0 所提及的標準處理模塊包括有「autoid.core」 及 「autoid.readerproxy」；而在 EPCglobal.cn 的 Savant 技術說明書中，所提及的必需標準處理模塊包括有：事件管理系統（Event Management System, EMS）、即時內存數據架構（Real-time In-memory Event Database, RIED）、任務管理系統（Task Management System, TMS），以下一一作介紹。

1. 標準處理模塊（Standard Processing Module）

   a. autoid.core

      autoid.core 模塊在企業介面上提供一些指令，使企業程式簡化對 Savant 的溝通，主要包括有：

      - autoid.core.GetSavantID

        企業程式發出 GetSavantID 訊息時，是在向所屬的 Savant 取得它的唯一碼（EPC 碼）。

      - autoid.core.GetCapabilities

        Savant 需向企業程式回應 Savant 內所包含的模塊，作列示。

      - autoid.core.Shutdown

        要求 Savant 作關閉動作。

      - autoid.core.ResetAll

        把模塊的資料作重放，相等於電腦的「warm boot」功能。

   b. autoid.readerproxy

      autoid.readerproxy 主要提供企業程式對指定的 Reader 連接時的簡化，可以連接指定的 Reader 作命令傳送。autoid.readerproxy 有包括以下兩個訊息：

      - autoid.readerproxy.GetReaders

要求 Savant 傳回 Savant 與它所在通訊的所有 Reader 的資料。

- autoid.readerproxy.RunCommand

  要求列出 Reader 所包含的指令。

2. 必需標準處理模塊（Required Standard Processing Module）

Savant 的必需標準模塊有三個：

a. Event Management System 事件管理系統

事件管理系統應用在 Edge Savant（ES）上採集 Tag 解讀事件，它與解碼器應用程式通訊，管理解碼器發送的事件流。在 Savant 系統中，EMS 是最重要的組件，它為用戶提供了集成其他應用系統的平台，達到過濾 EPC 數據的效果，然後將資料傳送至 RIED 儲存。

b. 即時內存事件數據庫（Real-time In-memory Event Database）

RIED 是一個用來保存 Edge Savant 訊息的內存數據庫。從 EMS 提供過濾和記錄事件的框架，記錄器可以將事件保存在數據庫中。但是，數據庫不能在一秒內處理幾百次以上的交易。RIED 提供了與數據庫一樣的界面，但性能要好的多。企業應用程式可以透過 JDBC 或本地 Java 界面訪問 RIED。RIED 支持像 select,update,insert 和 delete 這樣的 SQL 操作，還支持一部分 SQL92 中定義的數據操作方法。RIED 也可以保存不同時間點上數據庫的「快照」，以記錄 Reader 每秒讀取的數以百萬計的資料，再供企業程式查詢已記錄的事件。

c. 任務管理系統（Task Management System）

Savant 用戶定製的任務進行數據管理、數據監控，TMS 負責管理由上級 Savant（IS）或企業應用程式發送到本級 Savant 的任務。Savant TMS 使分佈式 Savant 的維護變得簡單。企業程式可以僅僅透過在一組類伺服器保存最新任務和在 Savant 中恰當的安排任務進度來維護 Savant。寫入 TMS 的任務可以獲得 Savant 的所有便利條件。TMS 可以完成企業的多種操作，例如：

- 數據交互，即向其他 Savant 中發送產品信息或從其他 Savant 獲取產品信息。
- PML 查詢：查詢 ONS/PML 服務器獲得產品實例的穩態或動態信息。

- 刪除任務進度：確定和刪除其他 Savant 上的任務。
- 值班報警：當某些事件發生時，警告值班人員，例如需要向貨架補貨、懷疑小偷和產品到期。
- 遠程上載：向遠處的供應鏈管理服務器發送產品信息。

4.2.2 Reader 通訊協定

簡單的來說，RFID 的 Reader 和主機(Host)間的溝通，必須經由一套有系統的架構，使得資料傳輸更具快速、效率及高保密性。有鑑於此，EPC 制定了一套 Reader 與主機間的通訊協定，作為各家 Reader 製造商和中間軟體開發商遵循的規範。而此規範在 Auto-ID Reader Protocol 1.0[10]中已有初步的雛形。

大致上 Reader 通訊協定可分為三大層級，如圖 22：

1. Reader 層

   定義 Reader 和 Host 間訊息交換傳輸的格式和內容，為 Reader Protocol 最基本亦是最核心的部份。此層級詳細定義 Reader 和 Host 間訊息，交換傳輸的格式和內容，為 Reader Protocol 最基本亦是最核心的部份，當中定義 Reader 操作執行時各編碼的意義和組態。

2. 訊息層

   規範封包在 Reader 層的架構、演變流程以及在傳輸網路中的特定傳輸模式，另外也定義安全加密的基本架構。此層級規範封包在 Reader 層的架構、演變流程以及在傳輸網路中的特定傳輸模式，另外也定義安全加密的基本架構(涵蓋認證、授權、可性度和資訊的完整性等部分)。此層級的要點特別重於，如何下層網路之間的聯繫如何被建立?回覆訊息是否需要同步?安全加密的預設模式及流程，初始化因為任何起始的封包都必須建立相同的或出使的安全服務，例如在封包傳遞過程中都必須採取像資料加密的動作。

3. 傳輸層

   規範後端網路資訊傳輸驗證的操作方式。

   而訊息層與傳輸層兩者之間的關係往往又密不可分，故習慣上又統稱為「Messaging /Transport Binding」即『MTB』。指明符合後端網路資訊傳輸驗證的操作系統等等

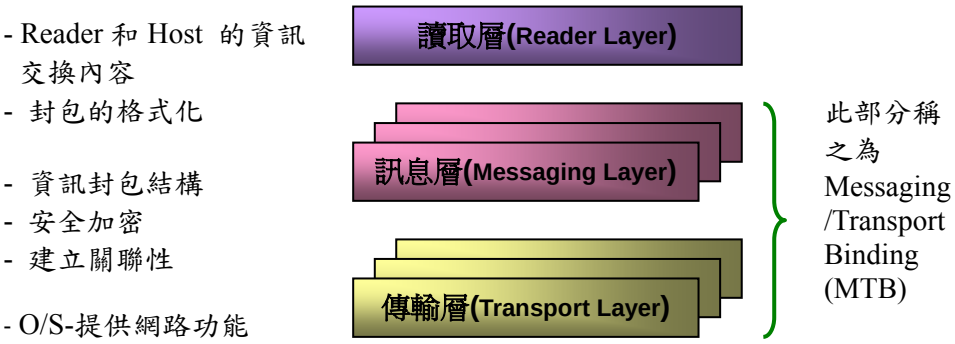

圖 22、Reader 至網路系統的運作機制

資料來源：[10]

4.3 中介軟體發展階段

蕭榮興等[30]提到 RFID 中介軟體發展趨勢來看，其可分為三大階段如圖 23。

- Application中介軟體發展階段

  RFID 初期的發展多以整合、串接RFID Reader為目的，本階段多為RFID Reader廠商主動提供簡單應用程式介面，以供企業將後端系統與RFID Reader串接，例如德州儀器、AWID、Matrics、EMS、Alien Technology Corp 等。以整體發展架構來看，此時企業的導入尚須自行花費許多成本去處理前後端系統介接的問題，通常企業在本階段會透過Pilot Project 方式來評估成本效益與導入的關鍵議題。

- Infrastructure中介軟體發展階段

  本階段是RFID中介軟體成長的關鍵階段，由於RFID 的強大應用，Wal-Mart與美國國防部等關鍵使用者相繼進行RFID 技術的規劃並進行導入的Pilot Project，促使各國際大廠持續關注RFID 相關市場的發展，在系統架構的中介軟體層的廠商，例如Oracle、Sun、IBM、TIBCO 等，原本專注於應用程式對應用程式的溝通，並開發應用軟體伺服器等相關產品，也開始在原本的中介軟體自行開發RFID Adapter（例如Sun、TIBCO、IBM 等）；也有些廠商（例如Oracle、Manhattan Associates 等）與RFID Reader廠商合作提供特定Solution，以加快進入市場的速度與競爭力，本階段RFID 中介軟體 的發展不但已經具備基本資料蒐集、過濾等功能，同時也滿足企業多對多

（Devices-to-Applications）的介接需求，並具備平台的管理與維運功能。

- Solution中介軟體階段

未來在RFID標籤、Reader與中介軟體發展成熟過程中，各廠商針對不同領域提出各項創新應用Solution，例如Manhattan Associates 提出「RFID in a Box」，企業不需再煩惱前端RFID 硬體與後端應用系統的連結，該公司與Alien Technology Corp 在RFID 硬體端的合作，發展Microsoft .Net 平台為基礎的中介軟體，針對該公司900 家的暨有供應鏈客戶群發展Supply Chain Execution（SCE） Solution，原本使用Manhattan Associates SCE Solution 的企業只需透過「RFID in a Box」，即可在原有應用系統上快速透過RFID來加強供應鏈管理的透明度。

由以上文獻之探討，可以瞭解中介軟體之重要性不言而語，而其發展也針對不同同途和階段有所異同。本研究計劃先針對「Application中介軟體發展階段」著墨，撰寫應用程式介面，完成第一階段資料系統之間資料轉換之介面。

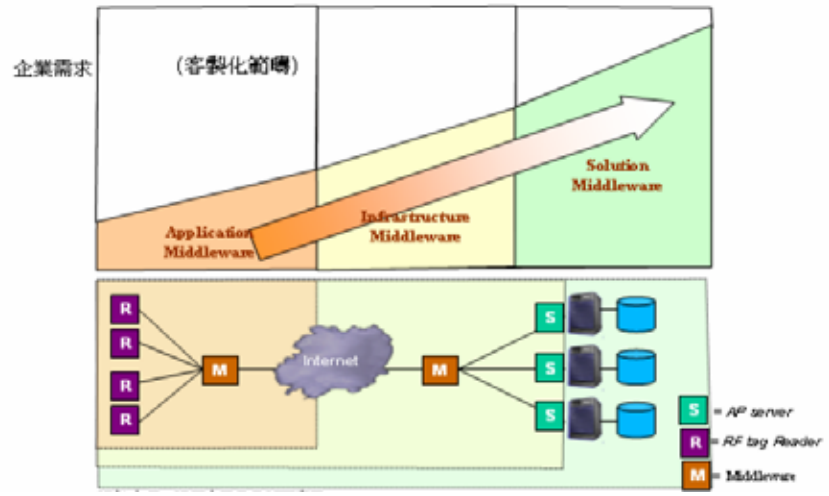

圖 23、中介軟體發展階段

資料來源：[30]