MyAdvantechAPI 專案

1. 現況

MyAdvantech這幾年已累積開發了許多重要商業流程、邏輯與功能。但是現行的系統架構無法容易的將這些已開發好的功能分享給其它同質性的系統使用，例如當eStore想要使用MyAdvantech的價格查詢功能時，我們還要花額外的時間將價格查詢功能另外包裝成eStore可以呼叫的服務。

另外，隨著越來越多的需求被整合與開發，長久下來除了產生出不易閱讀與維護的程式碼，而且過長未經重整的程式碼也拖慢了執行時間,導致海外的使用者抱怨系統太慢，因此效能問題也是急需改善的部份。

因此我們需要重構核心程式來解決上述所面臨的問題，提昇系統品質。

1. 專案目地

我們已使用現今主流的軟體設計方法「UML」，將MyAdvantechAPI主體架構設計出來，請參考第5點架構設計。

所以此行的目地是到A+TC與MyAdvantech、eStore開發團隊說明此專案的架構，然後帶領團隊成員按設計將專案的主體建置完成，並且讓團隊成員都能了解未來如何按照系統設計藍圖來開發程式。

另外，此專案預期的成效如下：

* 建置出可共用於不同系統的電子交易與網路行銷核心程式庫
* 改善程式效能
* 讓程式碼清楚易懂易於維護
* 減少儲存重覆的資料

1. 人力配置

專案架構設計：Frank.Chung

程式開發：Ming.Zhao、IC.Chen、Frank.Chung

程式碼品質驗證：Frank.Chung

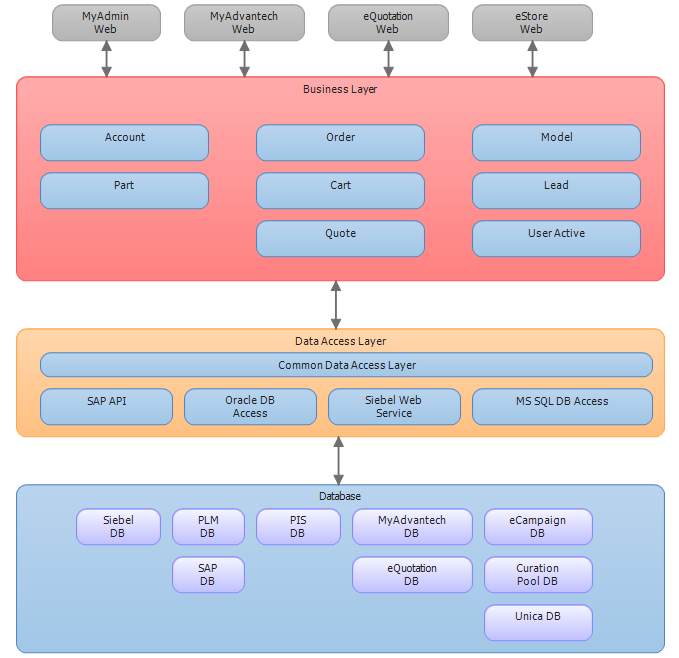
測試：James.Wu

1. 專案架構設計

MyAdvantechAPI區分為三大層，說明如下：

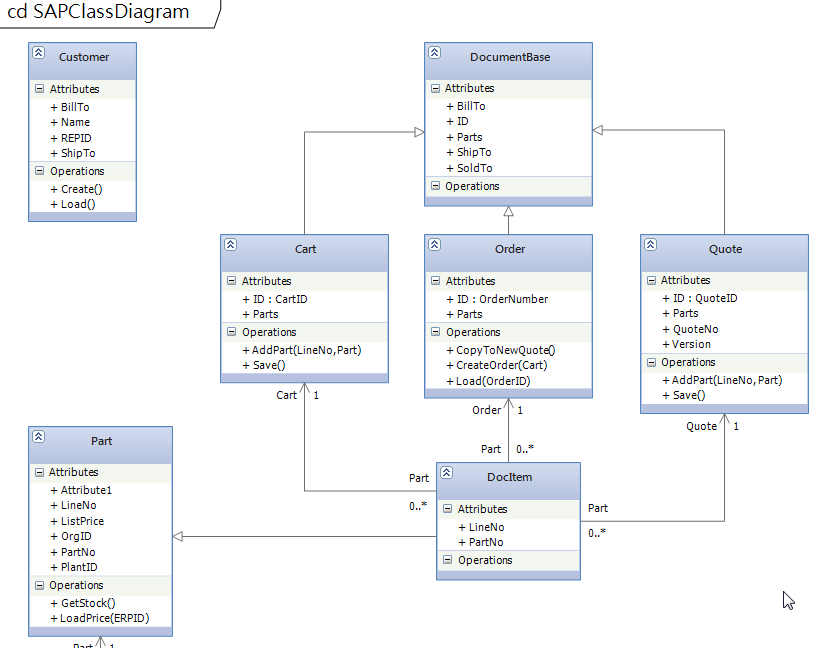
* Business Layer：從UI層接收使用者所想要執行的商業功能並且呼叫Data Access Layer取得相關資料後進行商業邏輯運算，然後把運算結果回傳給UI層，讓使用者知道執行結果。舉例來說，執行報價單轉換成訂單功能、或是建立SAP的新客戶主檔等功能。
* Data Access Layer：單一的資料存取匝道，用來提供Business Layer存取資料庫中的資料。Business Layer內的物件不用考慮該到哪一個資料庫取得資料，統一由Data Access Layer來決定資料該到哪裡取得，或是應該更新到哪一個資料庫。
* Data Layer：實體資料層

Layer Diagram



Class Diagram

以真實世界的角度將電子交易行為中的角色實體化成為一個一個的物件，所以將來系統內部是以物件與物件之間合作，來完成使用者所需要的各項服務。



Sequence Diagram

以時間序列的方式來描述某個功能從開始到結束這段時間內，使用者、介面與商業邏輯物件之間的互動行為，讓開發人員了解需要用到哪些商業邏輯物件來開發此功能。

