

## Tiburon 遊記 2 DataSnap 和 JSON

什麼是 JSON，我想我不必多說，因為 Internet 上一堆有關 JSON 的說明各位可以自行搜尋，簡單的說 JSON 是一種資料傳遞的格式，流行於 JavaScript 和 Ajax 的世界。OK，那麼 JSON 和 DataSnap 又有什麼關係？當 DataSnap 應用於分散式架構時不是使用 COM/COM+嗎？Tiburon 的 DataSnap 為什麼要使用 JSON 呢？

其實這些問題在我第一次知道 DataSnap 使用 JSON 時也立刻出現在我腦中，不過一經思索很多問題的答案就立刻浮現了出來而且也立刻延伸出了許多其他的想法，知道為什麼嗎？因為在我觀察任何技術時我還是喜歡馬上看看站在這個技術後面的人的背景以及這個技術代表的趨勢，現在讓我還原一下當初我腦中思考的過程。

還記得 Steve 嗎？它現在是 Delphi/C++Builder 在資料存取技術方面的架構師，我以前也寫過有關 Steve 的文章，其中敘述了 Steve 是來自 Java 的背景，因此在 VCL 架框中屬於 Steve 小團隊開發的程式碼中我都可以感覺到些許 Java 的味道，特別是在一些架構方面更是清楚的透露了 Steve 的背景。既然我們瞭解了 Steve 的背景，那麼當 Steve 在面臨負責 DataSnap 的研發方向時你覺得 Steve 會怎麼做？

因此讓我們試著分解前面的數個問題，：

問題	種類	答案&方向
當 DataSnap 應用於分散式架構時不是使用 COM/COM+嗎	這是屬於分散式架構訊息傳遞層的問題。OK，想想看，當 COM/COM+不再是主流時，DataSnap 如果仍然使用 COM/COM+ 那麼將會愈來愈封閉。	找出一種跨平台和主流的分散式通訊協定和訊息傳遞格式，TCP + JSON 肯定是最好的選擇之一。
DataSnap 為什麼要使用 JSON 呢	這是屬於策略的問題，DataSnap 使用了 JSON 之後可以開啓什麼樣的機會之門？簡單，Delphi/BCB 可以立刻和所有使用 JSON 技術的軟體整合在一起。	DataSnap 使用 JSON 做為分散式架構訊息的格式後，不但可以脫離 COM/COM+ 的限制，又可以結合跨平台 (Win64 也可以使用)，也可以和 Java，Ajax，.NET 整合，讓 Delphi 可以立刻融入 Web 上主流訊息格式的世界。
JSON 和 DataSnap 又有什麼	這是屬於實作的問題，	當然可以，JSON 雖然來自

關係	Delphi/BCB 在提供分散式架構時可以使用 JSON 做為訊息封裝的格式嗎?	Java/JavaScript 的世界，但也只是一種訊息封裝的格式而已，JSON 簡單又快速，實作也不難。
----	---	---

所以現在很清楚了，Delphi/BCB 的 DataSnap 採用 TCP + JSON 之後不但提供了更具彈性的分散式架構，而且由於 JSON 的簡潔，因此在傳遞速度上將會非常的理想，

使用 JSON 另外一個好處是 JSON 也定義了傳遞物件的格式，這非常的有趣，因為 DataSnap 中的 DataSet，Delta 等都是物件，那麼不是都可以使用 JSON 來傳遞嗎？再者遠端物件的方法也可以當成物件來看待，那麼再進一步也許再藉由 Delphi/BCB 原有的 RTTI 那麼是不是可以做到像 Java 的 RMI(Remote Method Invocation)呢，我想當時 Steve 心中一定是這樣想的。如果能夠這樣做到的話，那麼代表 Delphi/BCB 的用戶端就可以像 Java 一樣在用戶端呼叫遠端的伺服器服務，再藉由 DataSnap 原本豐富的 DataSet/Delta 等處理資料的功能，那麼這將比 Java 的 RMI 強太多了。

這樣的思路的確發人省思，因為如此一來不但發展出了 DataSnap 新的分散式架構，允許 Delphi/BCB 用戶端不必使用 COM/COM+ 就可以呼叫遠端服務，而且使用 JSON 之後 Delphi/BCB 可以立刻整合於 Java，Ajax 和 .NET。

不過要這樣做仍然需要克服 2 個最大的技術問題，那就是：

1 原本的 DataSnap 分散式架構使用 COM/COM+，用戶端的 TClientDataSet 等元件是藉由 IAppServer 介面和遠端的 Remote 資料模組溝通，因此如果新的 DataSnap 使用 TCP + JSON，那麼舊的 DataSnap 應用程式怎麼辦？

2. Delphi/BCB 原本的 RTTI 提供的資訊不夠豐富，而 Delphi/BCB 又沒有像 Java/.NET 那樣的 Reflection API，因此用戶端無法擷取足夠的遠端方法的 MetaData 進而組合正確方法原型以呼叫遠端的服務方法。

第一個問題不難解決，舊的 DataSnap 仍然會存在於 Tiburon 之中(也就是說仍然可以使用 COM/COM+)，問題只是在如何通透的讓舊的 DataSnap 架構能夠使用 JSON，其實這個關鍵只是 IAppServer 介面，因此我們只需要使用 Adapter 設計樣例就可以簡單的解決，那就是讓 DataSnap 用戶端連結到新的元件，這個元件同樣實作了 IAppServer 介面，這可以讓 DataSnap 用戶端以為仍然是連結到以前的 Remote 資料模組，但事實上已經是連結到使用 TCP +

JSON新架構的元件了(就是稍後我們會討論的TDSServer和TDSServerClass元件)。

第二個問題則需要新的編譯器技術，在原本的 RTTI 中產生足夠的 Metadata 可以讓 DataSnap 用戶端呼叫遠端方法，這就是 Tiburon 新的編譯器指令{\$MethodInfo ON}和{\$MethodInfo OFF}的目的。

OK，瞭解這些基本的思路和想法之後，下次就讓我們看看如何在 Tiburon 中使用 TCP + JSON 建立一個新的分散式應用程式，同時我們也會觸及到 JSON 是出現在這個新的分散式架構的那個地方，Have Fun!