I412995

公告本

102 年 6 月 27 日修正替換本

發明專利說明書

（本說明書格式、順序及粗體字，請勿任意更動，※記號部分請勿填寫）

※ 申請案號：97131379

※ 申請日期：97 年 8 月 18 日　　※IPC 分類：G06F 17/30 (2006.01)

一、發明名稱：（中文/英文）

驗證一已開發應用程式是否與其他至少一個應用程式適當運作之隨選資料庫服務系統、方法及電腦程式產品

ON-DEMAND DATABASE SERVICE SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR VERIFYING THAT A DEVELOPED APPLICATION WILL OPERATE PROPERLY WITH AT LEAST ONE OTHER APPLICATION

二、申請人：（共 1 人）

姓名或名稱：（中文/英文）

美商易享資訊技術有限公司

SALESFORCE.COM INC.

代表人：（中文/英文）

保羅 德迪克

PAUL A. DURDIK

住居所或營業所地址：（中文/英文）

美國加州舊金山市一號市場地標 300 室

THE LANDMARK @ ONE MARKET, SUITE 300, SAN FRANCISCO, CA 94105, U.S.A.

國　籍：（中文/英文）

美國　U.S.A.

三、發明人：（共 1 人）

姓　名：（中文/英文）克雷格 魏斯曼/ CRAIG WEISSMAN

國　籍：（中文/英文）美國/U.S.A.

102 年 6 月 27 日修正替換本

四、聲明事項：

☐ 主張專利法第二十二條第二項☐第一款或☐第二款規定之事實，其事實發生日期為：　年　月　日。

☑ 申請前已向下列國家（地區）申請專利：

【格式請依：受理國家（地區）、申請日、申請案號　順序註記】

☑ 有主張專利法第二十七條第一項國際優先權：

1. 美國　2007 年 8 月 17 日　60/956,629
2. 美國　2008 年 8 月 15 日　12/192,943
3. 美國　2008 年 8 月 15 日　12/192,948
4. 美國　2008 年 8 月 15 日　12/192,951
5. 美國　2008 年 8 月 15 日　12/192,956

☐ 無主張專利法第二十七條第一項國際優先權：

☐ 主張專利法第二十九條第一項國內優先權：

【格式請依：申請日、申請案號　順序註記】

☐ 主張專利法第三十條生物材料：

☐ 須寄存生物材料者：

國內生物材料 【格式請依：寄存機構、日期、號碼　順序註記】

國外生物材料 【格式請依：寄存國家、機構、日期、號碼　順序註記】

☐ 不須寄存生物材料者：

所屬技術領域中具有通常知識者易於獲得時，不須寄存。

五、中文發明摘要：

根據具體實施例，提供驗證與一隨選資料庫服務相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作的機構及方法。提供這種驗證的這些機構與方法可啟用具體實施例來確定已開發應用程式的新版本可將在一先前版本的相同應用程式環境內運作。具體實施例提供這種驗證的能力可造成一改善的應用程式遷移發展/執行時間架構等等。

六、英文發明摘要：

In accordance with embodiments, there are provided mechanisms and methods for verifying that a developed application associated with an on-demand database service will operate properly with at least one other application. These mechanisms and methods for providing such verification can enable embodiments to ensure that new versions of developed applications will operate in the same application environment of a previous version. The ability of embodiments to provide such verification may lead to an improved application migration development/runtime framework, etc.

102 年 6 月 27 日修正替換本

七、指定代表圖：

(一)本案指定代表圖為：第（一）圖。

(二)本代表圖之元件符號簡單說明:

100	系統	104	開發者
102	隨選資料庫服務	106	用戶

八、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

無

九、發明說明：

【發明所屬之技術領域】

本發明一般係關於開發者架構，尤其係關於一以改良方式開發應用程式。

【先前技術】

先前技術段落內討論的標的不應假設為先前技術，僅為先前技術段落內所提到的結果。類似地，先前技術段落內提到或先前技術段落中標的所相關聯的問題不應假設在先前技術當中已經證明。先前技術段落內的主題僅表示不同的方式，而其也可能為一項發明。

在傳統資料庫系統內，使用者在一個邏輯資料庫內存取其資料資源。這種傳統系統的使用者通常運用該使用者自己的系統從系統擷取資料或儲存資料在系統上。一使用者系統可遠端存取複數個伺服器系統之一而輪流存取資料庫系統。從該系統擷取的資料可包含從該使用者系統發出至該資料庫系統的查詢。該資料庫系統可處理對於查詢中所接收資訊的要求，並將與要求相關聯的資訊傳送至該使用者系統。

通常想要開發許多擴展前述資料庫系統能力的應用程式。不過到今日為止，這種應用程式通常以獨立於已開發應用程式將在其中操作的一預期應用程式環境之外之方式來開發。結果，該已開發應用程式將無法在這種環境內正常操作的機會大增。如此導致應用程式遷移等等當中許多缺陷。

【發明內容】

根據具體實施例，提供確認與一隨選資料庫服務相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作的機構及方法。提供這種確認的這些機構與方法可啟用具體實施例來確定已開發應用程式的新版本將在一先前版本的相

同應用程式環境內運作。具體實施例提供這種確認的能力可造成一改善的應用程式遷移發展/執行時間架構等等。

在根據具體實施例內並且藉由範例，提供一確認與一隨選資料庫服務相關聯的一已開發應用程式是否將可與至少一其他應用程式搭配來正常運作之方法。使用這種方法，從一隨選資料庫服務上的開發者中接收一已開發的應用程式。進一步，提供描述該已開發應用程式範圍的資料結構。仍舊是，在此有與至少一個訂戶的至少一個其他應用程式相關聯之資訊識別，該已開發應用程式用此來操作。利用此架構，運用該資料結構與該資訊，判斷該已開發應用程式與該至少一個其他應用程式是否將可正常一起操作。

在其他具體實施例並且藉由範例，提供一種用於實行一已開發應用程式包含測試之方法。使用這種方法，在一隨選資料庫服務上接收一已開發的應用程式。此外，判斷是否提供至少一種測試。仍舊是，根據該決定有條件驗證該已開發的應用程式。利用此架構，判斷該已開發應用程式與該至少一個其他應用程式是否將可正常一起操作。

在其他具體實施例並且藉由範例，提供一種用於驗證一已開發應用程式之方法。使用這種方法，在一隨選資料庫服務上接收一已開發的應用程式。此外，該已開發的應用程式已經驗證。仍舊是，根據該驗證有條件散佈該已開發的應用程式。利用此架構，驗證該已開發應用程式與該至少一個其他應用程式將可正常一起操作，如此可根據該決定散佈該已開發的應用程式。

在具體實施例並且藉由範例，提供一種用於一隨選資料庫服務內程式指令一適當版本的執行時間喚起之方法。使用這種方法，在一隨選資料庫服務上接收一應用程式的程式指令。此外，已決定要喚起的程式指令版本。進一步，根據該已決定的

版本喚起該等程式指令的適當部分。利用此架構，可喚起用於一預期應用程式版本的程式指令。

雖然以參照至與一多用戶資料庫隨選服務相關聯的已開發應用程式確認之具體實施例來說明本發明，本發明並不受限於多用戶資料庫或應用程式伺服器上的佈署。具體實施例可使用其他資料架構來實施，即是ORACLE®、DB2®等等，而不悖離所主張具體實施例的範疇。

任何上述具體實施例都可單獨使用，或與其他一起組合使用。本說明書內含的發明也包含在簡單發明總結或發明摘要內只有部分提及或暗示或根本未提及或暗示之具體實施例。雖然本發明許多具體實施例由該先前技術許多缺陷所促使，這些缺陷可在該說明書的一或多處內討論或暗示，本發明的具體實施例並不需要解決這些缺陷。換言之，本發明不同的具體實施例可解決該說明書內討論過的不同缺陷。某些具體實施例只能部分解決某些缺陷，並且某些具體實施例並未解決這些缺陷。

【實施方式】

一般概觀

在此提供確認與一隨選資料庫服務相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統及方法。

在使用資料庫系統的應用程式開發當中，許多挑戰來自於應用程式通常以獨立於應用程式開發後將在其中操作的一預期應用程式環境之外之方式來開發的事實。結果，該已開發應用程式無法在這種環境內正常操作的機會大增。如此，此處提供用於確認操作充足的機構與方法可啟用具體實施例來確定已開發應用程式的新版本將在一先前版本的相同應用程式環境內運作。進行這種決定的具體實施例能力可造成一改善的應用程式遷移發展/執行時間架構等等。

接下來，根據許多示例性具體實施例，將說明用於確認與一隨選資料庫服務相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統。

第一圖說明根據具體實施例，一用於確認與隨選資料庫服務102相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統100。在本說明的範圍內，一隨選資料庫服務可包含任何服務，這些服務依賴可在一網路上存取的一資料庫系統。

在一個具體實施例內，隨選資料庫服務102可包含一多用戶隨選資料庫服務。在本說明內，這種多用戶隨選資料庫服務可包含任何服務，這些服務依賴可在一網路上存取的一資料庫系統，其中一或多個用戶可共享該資料庫系統硬體與軟體的許多元件。例如：一已知的應用程式伺服器可同時處理大量用戶的要求，並且一已知資料庫表可儲存用於潛在大量用戶的資料列。

如所示，隨選資料庫服務102與複數個開發者104通訊。在使用當中，隨選資料庫服務102調適成從開發者104接收已開發的應用程式。在本說明的範圍內，開發者104可包含開發電腦程式碼的任一或多個人或團體(例如公司、組織等等)。進一步，該等應用程式可包含任何電腦程式碼(例如一完整程式、一部分程式、一程式碼片段等等)。

此外，隨選資料庫服務102可與隨選資料庫服務102的一或多個用戶106通訊。在前述其中包含一多用戶隨選資料庫服務的隨選資料庫服務102之具體實施例內，可存在複數個用戶106。在本說明的任何案例中，一用戶係指可存取隨選資料庫服務102的任一或多個人或團體。例如：用戶106可訂閱隨選資料庫服務102。

藉此設計，隨選資料庫服務102用來提供該等應用程式的存取權限給隨選資料庫服務102的用戶106。使用這種方法，該

等前述應用程式可在隨選資料庫服務102的控制之下。利用管理這種控制，藉此提供一改善的開發/執行時間架構等等。

在許多具體實施例內，這種控制可用任何所要的方式來管理。例如：在應用程式未依附至政策的情況下，防止用戶106存取該等應用程式，讓隨選資料庫服務102實行任何所要的政策。在其他具體實施例內，隨選資料庫服務102可在這種非相容案例當中，利用防止或限制開發者104可存取的功能性，來實行這種政策。例如：在未符合特定需求之下並不允許將一應用程式公佈至隨選資料庫服務102。在一個特定具體實施例內，隨選資料庫服務102可根據一動態範圍限制器(dynamic contextual limiter)監視並限制該等應用程式的許多態樣並且終止相關程式碼。當然，該前述控制可用任何所要的方式實施。

在一個具體實施例內，該前述控制可利用隨選資料庫服務102，採取限制至少一個應用程式態樣的形式。例如：這種態樣可關於讓開發者104的該等應用程式可用之處理、儲存、頻寬等資源。藉此設計，隨選資料庫服務102用讓隨選資料庫服務102透過該等應用程式服務用戶106的能力最佳化之方式來限制開發者。

在許多具體實施例內，這種資源相關態樣可牽涉到與隨選資料庫服務102相關聯的資料庫、其中可利用該等應用程式存取這種資料的方式等等。在這種具體實施例內，該等前述態樣可包含但不受限於一資料庫欄位數量、一預定時段內對資料庫的查詢數量、查詢所回傳的列數、資料庫陳述式的數量(例如修改陳述式等等)、資料庫陳述式之間腳本陳述式的數量、一預定時段內處理(例如修改等等)的列數、交易陳述式的數量、從一最後一個交易控制陳述式算起的未承諾列總數、從一最後一個資料庫呼叫算起的腳本陳述式總數、一處理週期等等。

當然，這種示例性清單不可當成限制。例如：隨選資料庫服務102的態樣(例如電子郵件管理等等)也可受限。在一個特

定實例中，每次要求中可傳送的電子郵件數量以及/或每次要求中所發出的外送網路服務呼叫數量都受到限制。在許多具體實施例內，依照要求或依照每個時間週期(例如每天)的限制可施加於一應用程式。在後者具體實施例內，這種限制只能以依照一使用者或依照用戶的方式施加。

在其他具體實施例內，該等應用程式的開發可受控制。例如：利用施加要求該等應用程式進行測試的隨選資料庫服務102需求(例如驗證等等)，來控制該等應用程式。在具體實施例內，藉由對與隨選資料庫服務102相關聯的一應用程式介面進行呼叫，這種測試自然以一自動方式執行。

在牽涉到開發控制的本具體實施例其他態樣中，隨選資料庫服務102會要求寫下應用程式的功能測試，並且進一步要求程式碼涵蓋的一預定百分比。 在此具體實施例內，這種技術允許不管何時只要修改隨選資料庫服務102時就可執行這種測試，以減少意外中斷工作中應用程式的風險。藉此設計，可避免退步以及/或任何其他負面特色。

仍舊在其他具體實施例內，可控制隨選資料庫服務102的用戶106存取該等應用程式。例如：每一應用程式的一單一實例都可在隨選資料庫服務102的複數個用戶106之間樣例化。如此，隨選資料庫服務102只需要儲存一單一個應用程式副本，並且同時以上述方式在用戶106之間共享。

仍舊在其他具體實施例內，隨選資料庫服務102可用來決定與隨選資料庫服務102相關聯的一已開發應用程式是否將與其他應用程式搭配來正確運作。在此案例中，可從隨選資料庫服務102上的開發者104中接收該已開發的應用程式。然後提供描述該已開發應用程式態樣的資料結構。在此識別與一或多個訂戶或用戶106的至少一個其他應用程式(與該已開發應用程式一起操作)相關聯之資訊。利用此架構，運用該資料結構與

該資訊，判斷該已開發應用程式與該至少一個其他應用程式是否將可正常一起操作。

在許多具體實施例內，可具體實施隨選資料庫服務102來確定與隨選資料庫服務102相關聯的一第一應用程式和一第二應用程式或第一應用程式的不同版本充分運作。例如：在一個具體實施例內，在隨選資料庫服務102上可接收一已開發的應用程式，並且可決定是否至少提供一項測試。在此案例中，該測試可包含測試一第一應用程式是否與一第二應用程式搭配正常運作的測試。然後根據該決定，有條件驗證該已開發的應用程式。

在其他具體實施例內，在隨選資料庫服務102上可接收一已開發的應用程式，並且驗證該已開發的應用程式。然後根據該驗證，有條件散佈該已開發的應用程式。

仍舊在其他具體實施例內，可在隨選資料庫服務102上接收一應用程式的程式指令。然後決定要喚起的程式指令版本。進一步，根據已決定的版本喚起該等程式指令的適當部分。

而該前述控制可為靜態或動態、可或不可一致施加等等。例如：該等前述態樣與相關控制條件可或不可用於不同的應用程式、用戶106等等。僅作為例示，隨選資料庫服務102在執行一升級腳本時，相對於執行一每列資料庫觸發器等等時，允許更多資源。進一步，隨選資料庫服務102允許更多資源用於大型用戶106等等。

第二圖說明根據具體實施例，用於決定與隨選資料庫服務相關聯的一已開發應用程式是否將可與用至少一其他應用程式搭配來正常運作之方法200。選擇性地，可在第一圖的隨選資料庫服務102態樣內實施本方法200。不過，當然方法200可在任何所要的環境內執行。前述定義可應用在本說明當中。

如所示，於一隨選資料庫服務中，一已開發的應用程式從開發者處接收。請參閱操作202。然後建立描述該已開發應用程式態樣的一資料結構。請參閱操作204。

該資料結構所述的態樣可包含任何數量的態樣。例如：在許多具體實施例內，該等態樣可包含一版本識別碼、一法則（schema）、一訊息外型（message shape）、一語意行為（semantic behavior）、一致資源定位器等等。

此外，識別與至少一個訂戶的至少一個其他應用程式相關聯之資訊，該已開發應用程式欲與該至少一個其他應用程式一起操作。請參閱操作206。進一步，運用該資料結構與該資訊，判斷該已開發應用程式與該至少一個其他應用程式是否將可正常一起操作。請參閱操作208。

選擇性地，允許至少一個訂戶與該開發者編輯至少一個該已開發應用程式的組件。例如：若判斷該已開發應用程式與至少一個其他應用程式將無法一起正常運作，則允許至少該開發者與訂戶之一編輯該已開發應用程式。在此情況中，該至少一個組件可包含一欄位。例如：該欄位可為與該已開發應用程式或將不允許與該應用程式一起正常運作的至少一個其他應用程式相關聯之欄位。

在一個具體實施例內，可追蹤複數個已開發應用程式的版本，來決定複數個訂戶運用哪個版本。在此情況中，可執行該追蹤來確定訂戶朝向使用該已開發應用程式。

在其他具體實施例內，一隨選應用程式服務平台可接收資訊。在此情況中，這種資訊可包含內含該平台可運行應用程式的程式碼。進一步，該應用程式可關聯於版本的至少一個識別。此外，該資訊可儲存為與該版本相關聯的中繼資料(metadata)，讓該應用程式變更可使用該版本來追蹤。

如此，該平台可為一使用者組織將應用程式儲存成為中繼資料。選擇性地，一組織的所有開發都可在個別發展組織內達成，這可由呈現該組織的中繼資料副本開始。

此外，一組織所做的所有組態變更(例如組織許可、黑名單設定以及所有中繼資料等等)可擷取在一可攜式中繼資料檔案格式中，並可儲存在一原始碼控制系統內，如此只使用中繼資料應用程式程式編輯介面(API, "application programming interface")就可從基本原理（first principles）建立該客戶的開發組織。

選擇性地，多位開發者每一都可具有自己的開發組織。在一個具體實施例內，多開發者可根據這些變更的XML表示以及一專案資訊清單(例如指出專案內所有頂級(top-level)記錄/檔案的檔案)，使用中繼資料API在彼此之間共享處理中的變更。在該開發組織內，該專案可呈現成一未管理的套件，此套件可用於維持追蹤該專案需要哪些物件。該檔案系統資訊清單可為此資料庫專案維持最新，如此所有開發者可看見相同的專案工作項目。

選擇性地，中繼資料蜘蛛程式可在需要添加許多物件至該資訊清單來進行變更當中維護該資料庫專案。例如：該中繼資料蜘蛛程式可呈現的為「這些是我認為你還需要現在加入的，你想要加入哪一項?」一旦已經開發並測試該等變更，則會使用一佈署的API將這些變更套用至該生產組織。

在一個具體實施例內，生產組織可與該隨選服務供應商提供的標準功能性和管理安裝的交換應用程式整合。這些自訂可繫結至這些應用程式的確切主要版本號碼(設定可呈現該等組織控制的中繼資料)，並且開發者可具有能力選擇何時升級這些應用程式的新主要版本。

選擇性地，一組織可對一管理套件實施自訂(例如對訂戶控制的欄位)。在此情況中，可從明確從該資料結構中或從該

檔案格式中的該訂戶列中分割出一開發者列。例如：具有一明確的「覆寫」檔案格式用於訂戶自訂，並且該等變更可包含成為該資訊清單檔案的一部分。

若要產生一現有應用程式的一新開發組織(例如一新工作空間)，一開發者可簽署一新組織，並且可指定一應用程式命名空間、密碼以及該開發者所分支的版本。這可安裝所有必要應用程式(例如依附管理應用程式)，並且可將該應用程式命名空間放入一開發模式中，則可讓其組件可進行編輯。在相同應用程式上或該應用程式的版本上工作之多個開發者可使用原始碼控制以及中繼資料API來通訊與共享進行中的變更。

在一個具體實施例內，該應用程式定義的一部分可包含對於其他被管理應用程式以及/或特色（features）、許可（permissions）和編輯（editions）的依賴。該平台所提供的測試環境可允許在所有容許的組態中測試該應用程式，此組態包含目標組織編輯設定。這可牽涉到使用這些編輯、安裝所有程式碼以及/或運行該程式碼，來建立拋開測試組織。進一步，允許該測試碼模擬暫時取代該開發者組織的不同編輯或設定，而不用確定其中任何變更。

選擇性地，針對一應用程式的一新主要版本，可參照的所有組件都用具有此新版本的最低版本戳記標示。在一個具體實施例內，可參照的組件包含法則(例如物件與欄位等等)以及/或Apex Code識別碼。另外，若應用程式開發者想要結束特定元件(例如反對該等組件等等)，該版本中繼資料的一部分可包含此資訊。例如：最新版本可變成該組件的最高版本。

一旦版本的開發與測試已經完成，則可使用這些開發組織之一來上載該新版本。然後建立一新版本號碼。例如：當分支時可建立一分支點釋放版本號碼。在一應用程式版本全域表中也可建立一新版本列。此列出現會讓開發組織的之前版本無效，因為之前版本已經成為該應用程式的一非現用版本。

在一個具體實施例內，可使用主要與次要版本的一標準附註以及分支來實施版本編號。選擇性地，可實施焦點版本化，如此版本為數值並且沿著每一分支線單一增加。在其他具體實施例內，該版本化使用三階段號碼法則，像是1.0.0和2.0.0這類主要版本用於導入主要功能性、新法則以及/或用於改變Apex Code和法則物件(例如新增和丟棄等等)的公開簽署。

在此案例中，次要版本可分支。這可允許在開發其他主要版本時新增新功能性至主要版本。此外，允許次要版本融合至其他主要版本內。例如：次要版本號碼可包含2.1、2.2、3.1等等。

以此方式，完全合格版本的頭兩個號碼可決定所使用應用程式版本的公開簽署。如此，「版本」一詞就是主要與次要版本號碼(例如該版本號碼的x.y部分)。

在一個具體實施例內，一應用程式中每一外部可定址組件都可接收到一最小版本號碼，這對應於其之前加入的版本。該應用程式開發者也可藉由指出該組件不再出現於目前開發版本之中，來結束(例如反對等等)一組件。這表示該組件的最高版本等於從目前開發工作之基本版本。為此，較高的應用程式版本可能不再是較低版本的一高級版本。如此，所有附屬版本都繫結於一現有版本號碼。

在本說明的範圍內，外部可定址組件可包含法則組件(例如客戶物件、客戶欄位等等)、Apex識別碼(例如類型、變數、方法等等)以及可參照的所有其他頂級中繼資料組件(例如特定檔參照至佈局、工作流程規則參照至工作流程動作等等)。因此，自訂物件與欄位可賦予一最高版本，這允許忽略。這同樣適用於Apex識別碼。

在一個具體實施例內，與其他受管理的應用程式互動之所有受管理的應用程式可指定其所依賴的現有版本。選擇性地，這可匯出成為一應用程式版本的中繼資料部分。類似地，一組

織內的程式碼可明確指定使用以及/或依賴哪個應用程式版本。

在此案例中，可運用一組織版本束（organization version bundle）。一組織版本束包含一組應用程式版本號碼。選擇性地，每一組織都具有至少一個版本束，在此一預設版本束對應至所有客製化，其中該已安裝應用程式的組織和/或版本會在Apex Code或其他客製化已經授權時假設。

在一個具體實施例內，應用程式開發者撰寫向下相容於舊版本的最新程式碼。這可利用將新執行時間存取曝露(例如在Apex Code內)至目前要求的版本號碼來達成。應用程式開發者可開啟此資訊來模擬舊行為。

在某些案例中，應用程式開發者可負責實行規則，像是不會針對舊版本改變語意。若要促進此效果，測試碼可當成該應用程式的舊版本來運行，如此一開發者可主張和/或測試該舊版本仍舊可用，即使該應用程式的最新版本行為可能有所不同。

在一個具體實施例內，該等中繼資料API依照名稱參照至頂級中繼資料物件。這可增加一使用者編輯這些檔案的能力。名稱可屬於該組織或應用程式內部，或至其他管理應用程式內的名稱參照。一組織/應用程式內的參照可立刻全部改變。

在應用程式之間，該基本應用程式開發者可重新命名識別碼或反對該等識別碼。如此，一應用程式的中繼資料可包含識別碼的舊名稱，因為仍舊支援的版本需要版本資訊。因為開發者總是可繫結至其他應用程式的正確版本，該等名稱不可重複，即使該應用程式的一新版本已經推送入該訂閱組織也一樣。

當已經安裝一應用程式版本，可在該訂戶組織內將一符號表具體化，可代表將一版本束、團體類型以及/或團體名稱映

射至該組織內該列的ID。所有編譯和/或中繼資料API操作都可使用此符號表進行名稱解析。

藉由使用鎖定的版本束(例如可主張一應用程式的影像由該組織所使用)，安裝一新版應用程式至組織內並不會改變該訂閱組織的應用程式觀點。進一步，若一組織準備使用該新版應用程式內的功能性，則該訂閱組織可改變其目前的版本束來在觀察到任何變更之前，簽署或功能性方面，指向該新版本束。選擇性地，在一測試環境中首先完成此動作。

此外，可追蹤一組織內一應用程式的最低安裝版本，因為該組織內許多不同版本束指定不同的版本號碼。如此，其最高版本低於該訂閱組織內所使用最低版本的任何元件都可扣留不安裝。

在一個具體實施例內，上述該低版本號碼可用於變更(例如固定)項目，像是Apex碼、Apex頁面、文件等等。選擇性地，這些變更可推送至所有訂閱組織。其他選擇性地，低版本並不會改變任何處理或法則物件的公開簽署。如此，任何版本取代對於該訂閱組織來說是顯而易見的。

在一個具體實施例內，只會安裝一應用程式目前最低的版本。新低版本可短時間內推送至所有訂戶組織。例如：新低版本可透過一可靠、可重新啟動的背景處理來推送。

吾人應該注意，在不用改變該訂閱組織的應用程式行為之下，也可將一新主要版本的中繼資料推至所有組織，只要正確撰寫應用程式往下相容即可。如此，安裝在一組織內的版本可與該組織所使用的版本不同。在不改變該等語意之下可安裝最新程式碼。

如此，一夥伴客戶可變更其應用程式的內部運作，只要該等公開語意維持一樣即可。為此，可將升級推送至一訂戶組織所使用的任何分支之頭版即可。不過，一般來說，該夥伴不會強迫訂戶改變該舊版本使用的語法。

在其他方面，鼓勵一夥伴及其訂戶之間的通訊。一夥伴應用程式的組織允許該夥伴觀察誰已經安裝該應用程式，及/或者目前正在使用哪個版本(例如透過版本束或最低版本等等)。

在大多數情況下，該夥伴不會移除某些訂戶組織仍舊使用的版本所用之任何元件。不過，一旦任何訂閱組織看不見某個自訂欄位，則其可能已經移除。這會發生在反對之後一長段時間。選擇性地，該夥伴可指示新組織可能安裝的應用程式之最低版本。

第三圖顯示根據其他具體實施例，用於確認與隨選資料庫服務302相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統300。選擇性地，可在第一圖至第二圖的架構以及功能性範圍內實施本系統300。不過，當然系統300可在任何所要的環境內執行。同樣地，在本說明當中可應用前述定義。

如所示，隨選資料庫服務302仍舊透過一網路308與一開發者304和至少一末端使用者用戶306通訊。進一步，隨選資料庫服務302包含分別與開發者304和使用者用戶306介接的一應用程式伺服器310。尤其是，應用程式伺服器310在一編譯時間時相期間（compile-time phase）與開發者304介接，在一執行時間相位期間則與使用者用戶306介接。

例如：在一個具體實施例內，應用程式伺服器310調適成從開發者304接收程式語言指令(例如腳本等等)，開發者304就是要擴充隨選資料庫服務302 API的人。在回應接收到這種腳本當中，應用程式伺服器310處理(例如編譯等等)和儲存腳本在一資料庫312內。選擇性地，這種處理可進一步包含任何所要的控制以及/或上述更新，以確定開發者304具體實施最佳實施方式，或腳本開發內任何其他預定實施。在一個具體實施例內，這種編譯腳本可儲存成中繼資料形式，用於回應來自末端使用者用戶306的要求。利用此功能，該腳本可調適成被觸發

來回應來自末端使用者用戶306的一特定相關要求(例如回應選擇、存取、修改等等物件)。

尤其是，應用程式伺服器310進一步調適成接收來自末端使用者用戶306的要求。在回應這種要求當中，藉由使用這種要求來識別並從資料庫312取得修正編譯的腳本，運用應用程式伺服器310的執行時間解譯器（runtime interpreter）314來處理。執行時間解譯器314進一步具備處理該已編譯腳本的能力。如此該已編譯的腳本可致力於滿足要求的方式中。

第四圖顯示根據具體實施例，用於實行在一已開發應用程式內包含測試之方法400。選擇性地，可在第一圖至第三圖的架構以及功能性範圍內實施本方法400。不過，當然方法400可在任何所要的環境內執行。在本說明當中可應用前述定義。

如所示，在一隨選資料庫服務上接收一已開發的應用程式。請參閱操作402。進一步，判斷是否提供至少一種測試。請參閱操作404。

在一個具體實施例內，該測試由該已開發應用程式的一開發者所提供。選擇性地，該測試可提供給該已開發應用程式。一旦決定是否提供一測試，則根據該決定有條件驗證該已開發的應用程式。請參閱操作406。

在一個具體實施例內，決定至少一個測試是否符合預定條件。選擇性地，該預定條件可包含一測試結果。例如：可利用已知的參數來執行該測試，並且判斷該執行結果是否在一可接受的預定結果內。在某些案例中，該預定結果可包含一標準差，如此該執行結果可在數值範圍內。

在其他具體實施例內，該預定條件可包含一臨界。在此情況中，該測試可執行並且判斷該執行結果是否在一臨界內。仍舊在其他具體實施例內，該測試可執行並且判斷該執行結果是否導致錯誤。

還是在其他具體實施例內，判斷至少一個測試是否足夠測試該已開發應用程式是否可執行一預定版本。例如：該已開發應用程式可包含新或更新的電腦程式碼、變數、變數名稱、方法、清單以及/或許多其他物件。在此情況中，該測試可測試該已開發應用程式是否可執行至少部分舊版應用程式，這可包含在該已開發應用程式當中。

選擇性地，根據至少一個決定可有條件驗證該已開發應用程式，像是該測試是否符合預定條件以及/或該至少一個測試是否足以測試該已開發應用程式是否可執行一預定版本。

在一個具體實施例內，該至少一個測試可測試該已開發應用程式是否可根據一已決定版本喚起程式指令不同部分。例如：該已決定版本可包含一較舊或較新版已開發應用程式。在此情況中，該等程式指令可包含與該已開發應用程式相關聯的任何指令或程式碼。

在某些案例中，該至少一個測試可測試該應用程式是否可運作，如此若在執行時間期間判斷要喚起該等程式指令的一第一版本，則會喚起該等程式指令的一第一部分，並且若在執行時間期間判斷要喚起該等程式指令的一第二版本，則會喚起該等程式指令的一第二部分。

在此案例中，該等程式指令的第一版本與該等程式指令的第二版本可同時包含在該應用程式內。第一和第二版本可為一個較舊並且另一個較新的版本，其中版本的至少一個態樣有所不同。例如：該等程式指令的第一版本與該等程式指令的第二版本在至少一方法、至少一等級或至少一變數方面有所不同。

選擇性地，決定該至少一個測試是否包含於該已開發應用程式內。在一個具體實施例內，該已開發應用程式的驗證係根據該決定。例如：可判斷一測試是包含於該已開發應用程式，該測試對應至一第一和第二版本間之一已知變更。若包含該測

試，則可執行該測試並根據該執行結果決定有效性。若否，則該已開發應用程式無效。

在一個具體實施例內，可強迫一開發者撰寫模擬一至少部分應用程式一第一版本的測試，如此可判斷該至少部分應用程式的第一版本之作用足以匹配該應用程式的一第二版本。在此方式中，可在模擬模式中撰寫一測試，如此判斷一較舊版應用程式之作用足夠匹配一較新版應用程式。更進一步，若該較舊版的功能與該較舊版應用程式併入一較新版應用程式之前的功能一樣或類似，則可判斷該較舊版應用程式的功能足夠。

作為範例，一開發者可開發包含表1內所示偽碼的應用程式一第一版本。

表1

```
global integer m (string s) % % where m is any method {
        if (s == “a”)
            return 1;
        else
            return 2; }
```

因此，該開發者可判斷回傳的整數陣列，並且該字串「s」不應該與「a」比較，而是與「x」比較。因此，該開發者可建立包含表2內偽碼的應用程式第二版本。

表2

```
c deprecated global integer m (string s) {
        return m(s)[0];
}

global integer [ ] m(string s) {
        if (system.client_version < 2) {
            if (s == “a”)
```

```
                return new_integer[ ] {1}; }
        return new_integer[ ] {5,6}; }
```

在此方式中，在執行時間上可判斷運用該應用程式的哪個版本。根據此判斷，若判斷已經使用一較舊版本，則可利用與一舊版相關聯的程式碼或變數，若判斷已經使用一較新版本(例如在此案例中為{5, 6}等等)，則可利用與一新版相關聯的程式碼或變數。

此外，該應用程式內可包含一測試來判斷與一較舊版應用程式功能相關聯的行為在一已升級的應用程式版本內是否充足或正確。例如：表3顯示根據上述範例，可包含在該應用程式內來測試當該應用程式的第一版本在第二升級版本內實施時是否可充分操作之測試之偽碼。

表3

```
clstest (version = 1) test
system assert vals (1, m ('a'));
system assert vals (2, m ('x'));
```

如此，可測試至少部分應用程式，若是第一版本，則判斷該測試功能內使用的值是否與該應用程式第二版本內的值相同。在此方式中，一測試可模擬一特定用戶端以及/或一至少部分應用程式。進一步，執行時間程式碼可測試要利用哪個版本，以及可利用的正確程式碼。在一個具體實施例內，可實施第四圖內說明的測試方法，如此可在該隨選資料庫服務所提供一平台上建立一隨選資料庫服務的使用者。

第五圖顯示根據具體實施例，用於驗證一已開發應用程式之方法500。選擇性地，可在第一圖至第四圖的架構以及功能性範圍內實施本方法500。不過，當然方法500可在任何所要的環境內執行。同樣地，在本說明當中可應用前述定義。

如所示，在一隨選資料庫服務上接收一已開發的應用程式。請參閱操作502。此外，該已開發的應用程式被驗證。請參閱操作504。仍舊是，根據該驗證有條件散佈該已開發的應用程式。請參閱操作506。

在一個具體實施例內，若該已開發應用程式未驗證的話，則不會散佈該已開發應用程式。在另一方面，若該已開發應用程式已驗證的話，則散佈該已開發應用程式。在這些案例中，該驗證包含實施許多技術來驗證該已開發應用程式。

在一個具體實施例內，該驗證可驗證該已開發應用程式是否可執行其預定版本。例如：一第一較舊版應用程式可併入一第二較新版應用程式內。在此案例中，該驗證可包含驗證該應用程式第二版本是否可關聯於第一版本充分正常運作。

在其他具體實施例內，該驗證可驗證該已開發應用程式是否可根據一已決定版本喚起程式指令不同部分。例如，該驗證可驗證該應用程式是否可運作，如此若在執行時間期間判斷要喚起該等程式指令的一第一版本，則會喚起該等程式指令的一第一部分，並且若在執行時間期間判斷要喚起該等程式指令的一第二版本，則會喚起該等程式指令的一第二部分。在此案例中，該等程式指令的第一版本與該等程式指令的第二版本可同時包含在該已開發應用程式內。

進一步，該等程式指令的第一版本與該等程式指令的第二版本在至少一方法、至少一等級或至少一變數方面有所不同。選擇性地，可避免一開發者以會導致該前版已開發應用程式功能性喪失的方式來變更該已開發應用程式。例如：一開發者系統可明確避免該開發者做出會破壞該應用程式現有版本的移除或變更動作。此外，若將新版本推送給該訂戶時，可避免一開發者修改或刪除部分一已開發應用程式，而導致一訂戶組織內任何問題(例如使用舊版應用程式等等的問題)。

仍舊是其他具體實施例，該驗證可包含在該已開發應用程式上運行至少一個測試。如上述，該至少一個測試可包含在該已開發應用程式內。此外，判斷至少一個測試是否運行成為前版已開發應用程式。

例如：標註一測試應該運行「成為」一舊版的語法可包含在該測試與/或該已開發應用程式內。更進一步，一執行時間機制可包含測試一訂戶認為該實際程式碼內運行哪個版本。在此案例中，該已開發應用程式可包含至少一個測試，從該訂戶的觀點來說，在執行時間內判斷該隨選資料庫服務的訂戶所使用之已開發應用程式版本。

在一個具體實施例內，驗證用的測試程式碼需要涵蓋定址不同版本的所有真實程式碼行（all lines of real code）。如此，即使一應用程式具有相當高的測試涵蓋率，也必須100%涵蓋這些版本特定行。在此方式中，該驗證可包含判斷該已開發應用程式內是否呈現電腦程式碼涵蓋的一預定等級。然後若電腦程式碼涵蓋的預定等級呈現在該已開發應用程式內，則驗證該已開發應用程式。若電腦程式碼的預定等級不存在，則避免上傳該應用程式。

該已開發應用程式可運用許多種技術來散佈，例如：該已開發應用程式可利用推送技術來散佈。在一個具體實施例內，該已開發應用程式可利用推送技術進行非同步散佈。仍舊在其他具體實施例內，可下載該已開發應用程式。選擇性地，只散佈已更新的已開發應用程式部分。

第六圖顯示根據具體實施例，用於一隨選資料庫服務內程式指令一適當版本的執行時間喚起之方法600。選擇性地，可在第一圖至第五圖的架構以及功能性範圍內實施本方法600。不過，當然方法600可在任何所要的環境內執行。在本說明當中可應用前述定義。

如所示，在一隨選資料庫服務上接收一應用程式的程式指令。請參閱操作602。此外，已決定要喚起的程式指令版本。請參閱操作604。進一步，根據已決定的版本喚起該等程式指令的適當部分。請參閱操作606。

在一個具體實施例內，利用與該隨選資料庫服務的一訂戶相關聯之版本資訊來判斷版本。例如：一訂戶可運用對應至一舊版程式指令的一第一版程式指令。在另一方面，可使用與該已開發應用程式相關聯的程式指令一第二更新版本。

在此案例中，該應用程式可推送給複數個訂戶(例如以非同步方式等等)。至少該等訂戶之一可依賴與第一版本相關聯的功能，並判斷應該繼續使用第一版本。如此，根據此決定喚起該等程式指令的適當部分。

在一個具體實施例內，包含該等程式指令的測試可套用至該版本資訊，來找出要喚起的該等程式指令適當部分。在此案例中，該測試可包含根據版本資訊判斷要運用哪個程式指令版本。該版本資訊可包含一版本號碼、一版本日期、一最新修改日期以及/或許多其他版本資訊。

在一個具體實施例內，若判斷要喚起該等程式指令的一第一版本，則會喚起該等程式指令的一第一部分，並且若判斷要喚起該等程式指令的一第二版本，則會喚起該等程式指令的一第二部分。在此案例中，該等程式指令的第一版本與程式指令的第二版本可同時包含在該應用程式內。

進一步，該等程式指令的第一版本與該等程式指令的第二版本在至少一方法、至少一等級和/或至少一變數方面有所不同。例如：該等程式指令的第一版可為一先前版本，並且該等程式指令的第二部分為一新版。在此案例中，可根據一預定格式來標示該等程式指令的第一版本。例如：該等程式指令可用指示一版本(例如利用版本號碼等等)、一版本的子類別(例如利用一版本號碼的延伸等等)或其他標示技術等方式來標示。

運用這些技術，可允許一已開發應用程式的推送升級，如此該等軟體組件可為用一開發者可展開該升級套件以不中斷現有安裝的方式所釋放之正式版本。更進一步，允許一隨選資料庫服務的訂戶以可預測、可測試的方式切換至最新升級的應用程式版本。

版本化可實施讓軟體應用程式之間的擴充與參考概念正常化。傳統上，大部分應用程式組件以該訂戶所定義的方式來參照。不過，藉由專屬組件，一隨選資料庫服務可避免這種參照。版本化與反對允許對全域組件產生參考限制（restriction of references）。

這可使用許多技術來達成，如上面所述。例如：一應用程式內含的組件(例如類別、變數、方法等等)可匯出該應用程式之中所有的最低與最高版本。此外，所有對於一組件的參照都可在屬於一應用程式版本延伸的一特定版本束範圍內製作。進一步，一版本束可指定該隨選資料庫服務供應商的一應用程式版本，以及可從內容參照的每一應用程式之一單一版本(即是命名空間等等)。

仍舊是，所有對於組件的參照都可在一特定束內容內實施。在此方式中，擴充組件可儲存一特定版本束ID，並且可在執行時間上用該版本束來運作，該版本束與該要求的該初始版本束不同。在一個具體實施例內，所有輸入點(例如API呼叫等等)都包含一版本束內容。

使用與該等組件相關聯的版本資訊(例如版本ID等等)，開發者可利用刪除該組件來終止一組件。在此案例中，該組件可出現在具有相關一未刪除鏈結的一專案清單中。

在某些類別識別碼(例如Apex類別識別碼等等)和某些VF屬性等等的案例中，該開發者可使用像是一「@deprecated」注釋這類指示碼指出一識別碼已被移除，指示碼例如一標籤、一令符、一旗標等等。在此案例中，該@deprecated令符只允

許用於已經管理釋放的識別碼。若該等識別碼尚未管理釋放，則會導致錯誤。在這些案例中，一旦一識別碼已經在該@deprecated狀態下上傳，則該識別碼必須維持該@deprecated注釋。喪失該@deprecated注釋會導致一新的錯誤。

當反對一個類型時，也暗示反對該類型內的所有識別碼(例如內或外類型)。在一個具體實施例內，不可用此方式反對該等子代識別碼。例如：若已組態該等子代識別碼，如此當無法檢視該周圍親代識別碼時就無法檢視這些子代識別碼，不過該等子代識別碼並無法被反對。

在一個具體實施例內，Apex類別的一功能原型(例如在一訂戶組織內)可顯示版本資訊。在此案例中，該版本資訊可包含用於一全域識別碼的最低與/或最高版本。另外，該隨選資料庫服務的訂戶，像是開發組織，可允許透過一鏈結檢視原型。

選擇性地，可利用一表格來追蹤版本資訊。在此案例中，利用該隨選資料庫服務和/或該訂戶儲存或維護該表格。在這些案例中，當已上傳一更新的應用程式或應用程式套件(例如一部分應用程式等等)，之前的版本資訊(例如一版本ID等等)可儲存在指出最高版本的列或行內(例如「max_all_package_version_id」行等等)。此全域狀態可用於指示不存在於套件版本的一組件從最新上傳的版本開始。

選擇性地，一應用程式或一應用程式套件可包含一使用者介面(UI)，其顯示版本資訊給訂戶(例如開發者等等)，其中包含給子代識別碼(例如Apex類別識別碼等等)和佈局之資訊。其他選擇性地，一訂戶組織內的類別原型也顯示此資訊。

在一個具體實施例內，已上傳並且導入一已更新應用程式版本的新組件可用於該版本的訂閱組織，用於版本化。在此案例中，一第一上傳可決定該最低版本。一般而言，一組件並不會在一更新版本中被反對/移除。如此，該指定在訂戶上並無影響。

一應用程式可匯出或上傳該應用程式需要運作的版本束。一版本束可包含一組織/應用程式內的任何名稱/命名空間組件。在一個具體實施例內，該版本束只包含專屬組件。在此案例中，專屬組件就是其他命名空間內其他應用程式不可參照的組件。

在一個具體實施例內，一版本束可指定其他應用程式的版本。例如：一標題列（header row）可指定該隨選資料庫服務應用程式的版本。選擇性地，指定每一套件ID（例如一「AllPackageId」標籤等等）所使用版本ID（例如一「AllPackageVersionId」標籤等等）的一子代表（例如一「VersionBundleDefinition」表等等）。

吾人應該注意，可快取版本束及其定義。在一個具體實施例內，一組織資訊快取鍵或其他快取鍵可用來幫助每一快取。

在一個具體實施例內，當已經安裝一新應用程式時，一開發者並不必新增該應用程式至現有版本束。在此案例中，該已安裝的版本數可自動插入。在此方式中，該等版本束可包含複數個已安裝的應用程式，指定一版本給每一個應用程式。選擇性地，當解除安裝一應用程式，可自動從現有的版本束中移除該應用程式。

在其他具體實施例內，所有組織都可具有最新一個版本束。選擇性地，一組織可具有一個預定版本束。當該組織使用該隨選資料庫服務中目前的應用程式簽署時，可建立此預設版本束。在此方式中，在不改變該預設或其他版本束之下，組織將看不見該隨選資料庫服務應用程式內的新法則。

在某些案例中，一組織可複製或編輯一現有版本束。在一個具體實施例內，並不允許在生產當中即時發生編輯現有的版本束。例如：編輯受限為只對目前組件(例如Apex程式碼等等)有不利衝擊的編輯。

在一個具體實施例內，一開發者和/或一管理員可用一工具或介面呈現，指出哪個組件使用每一版本束。選擇性地，此工具可包含一版本束細節畫面。在其他具體實施例內，當嘗試以打破使用版本束的現有組件這種方式編輯現有版本束時可避免開發者和管理員繼續編輯並且/或被警告。

進一步，版本束可允許一應用程式內的每一組件指定該組件的特定需求。這些需求(例如功能資訊等等)與隨應用程式推展的不同組件而不同。在一個具體實施例內，利用在每一組件表的外來金鑰欄(例如一「VersionBundleId」欄等等)內儲存功能資訊等等，就可達成此目的。在此方式中，所有管理組件都具有「VersionBundleId」外來金鑰。

在升級上，一指示器指出可將該版本號碼(例如一「InstalledPackageVersion」標籤等等)解譯為已安裝應用程式的實際版本號碼。因為版本束確定一已安裝應用程式的所有使用都「鎖定」至一特定版本，則該推送升級處理可修改此號碼而不影響任何組件的語意。

此外，該實際安裝版本可指定在一版本束內可選擇的最高版本。最低版本標籤或指示器可指出該應用程式的最低非反對版本。其他選擇性地，在使用客戶已安裝的版本之前，禁止一訂戶使用一邏輯版本。

仍舊是，可實施延遲使用的時間，如此一版本束可能無法用於一應用程式的新安裝實際版本或用於一預定時間的應用程式部分。在此方式中，若發現問題，在使用該問題程式之前可「回頭」使用一舊版。在此案例中，一中繼資料安裝可支援該實體退回。

如上述，API呼叫可指定其完整版本資訊，而不是只有該API版本資訊。在許多具體實施例內，可利用即時指定要求內的版本束資訊(例如在標題內)或利用版本束特定端點達成。

在某些案例中，與該隨選資料庫服務的訂戶相關聯之開發者要模擬一舊版應用程式的應用程式碼行為(例如Apex行為等等)。如此，提供執行時間存取給至少該應用程式的目前執行之版本資訊(例如與「AllPackageVersion」相關聯的資訊等等)。這可提供在應用程式內，如以一種方法提供。此外，若要測試此程式碼，測試可實施(例如Apex測試)並且可具有能力指定一個測試區段執行為一特定版本。

在一個具體實施例內，一版本供應者介面可用來提供版本資訊。在此案例中，可實施該介面來取得並設定版本資訊。更進一步，此介面可用來決定已安裝組件以及相關版本資訊的能見度。該介面也利用最低/最高版本號碼、命名空間以及與組件相關聯的其他資訊，來決定組件的背景能力(backward capability)。

為了維護應用程式被訂戶使用的功能性，有關一已上傳類別階層的資訊可用避免一開發者移除或改變組件關係(這會讓訂閱組織的能力受損)之方式來記錄。例如：一全域表(例如一「core.all_managed_class_rel」[AMCR]表等等)可用來記錄類別與介面之間的管理關係。在此案例中，該表可包含複數個欄位，其中包含與其他全域類型、一親代列舉或ID (例如一使用者定義類型或一內件系統類型常數等等)、一套件ID和許多其他欄位有一關係(例如擴充或實施等等)的一全域Apex類別類型識別碼(例如「AMPM_id」)、。

在一管理釋放套件的上傳時間上，可記錄所有新類別關係。在此案例中，只有立即關係可記錄在該全域表內。換言之，並非所有暗示及物關係都記錄在該表內。當稍後儲存類別，可驗證在該表內現有的關係都沒有被移除。在介面實施的案例中，在該類別簽署當中留下備援的祖代關係。

在一個具體實施例內，擴充類別可保持常數。例如：一旦一全域類別延伸至其他全域類別，可保留確切關係。類似地，

在一類別已釋放而無全域超級類別的案例中，稍後不可加入一超級類別。更進一步，一類型的「性別」在上傳後可能不具有改變的能力。例如：一類別不可改變成一列舉或介面。

有關推送更新，將呼叫新增一新全域方法用於版本化。若擴充發生在使用相同簽署上，則基本類別內一新全域方法的存在會導致一擴充類別問題。不過，在Apex版本化的範圍內，當該訂戶升級至該版本束內最新版本時，該訂戶可新增一個覆寫關鍵字至方法。

在一個具體實施例內，在一較新應用程式版本內並未加入新全域最終方法（global final methods）。在此案例中，所有最終方法都可為虛擬（virtual）。例如：若該訂戶已經使用一種方法簽署，則該訂戶可能無法將一類別融合到基本類別的最新版本上，這是因為最終指示（final destination）並不允許。

選擇性地，因為該子類型可擁有具有相同名稱的類型，所以可新增新全域類型。在此案例中，至該基本類型的參照需要完全合格以便使用。此外，因為該訂戶可擁有具有相同名稱的變數，所以允許新增一新全域變數。有關靜態變數，使用完全合格的類別名稱可參照確切變數。有關成員變數，利用簽署該超級類型物件至子類型的一變數宣告，即使其被遮蔽，總是可達到該子類別變數。

在一個具體實施例內，並無新抽象或介面方法可加入一全域類別更新。例如：該基本類別可嘗試呼叫新抽象或介面方法，假設所有子類別或實施者都提供這些方法的實施。不過，使用一早先版本應用程式的訂戶尚無法實施這種方法。若從該基本類別的觀點上來看無法避免喚起這些方法，則可避免新增新抽象方法(例如在此類別等級上或階層內在其之下的任何類別)，以及避免新增具有相同效果的介面。

更進一步，全域抽象類別可避免包含任何公開抽象方法。這可因為並無法擴充應用程式來存取該應用程式組態執行的方法而實施。

此外，並未從一管理方法中取走一虛擬關鍵字。例如：一旦版本為虛擬，則該關鍵字應該保持虛擬。選擇性地，允許最終方法變成虛擬。仍舊是，若之前無全域建構子，則只允許一堅固類別變成抽象。

運用這些技術，一開發生態系統可具備管理組件壽命與實施簽署不變，這可將一隨選資料庫服務訂戶所使用的實體安裝版本與一邏輯版本分開。這確定一推送升級的相容性，這在生態系統內為自動管理升級處理，可確定直到該訂戶準備好之前都無可見的變更。更進一步，可運用一具有內建用戶端版本的簽署版本化以及執行時間檢查之編碼語言，允許自動測試後向的相容性。

系統概觀

第七圖顯示其中使用一隨選資料庫服務的環境 710 之方塊圖 700。根據選項，任何之前說明的前述圖式具體實施例都可或不可在環境 710 內實施。環境 710 可包含使用者系統 712、網路 714、系統 716、處理器系統 717、應用程式平台 718、網路介面 720、用戶資料儲存庫 722、系統資料儲存庫 724、程式碼 726 以及處理空間 728。在其他環境內，環境 710 可能不會擁有列出的所有組件且/或具有其他元件來取代或加入上列組件。

環境710為其中存在一隨選資料庫服務的環境。使用者系統712可為一使用者用來存取一資料庫使用者系統的任何機器或系統。例如：任何使用者系統712可為一手持式計算裝置、一行動電話、一膝上型電腦、一工作站以及/或一計算裝置的網路。如第七圖內所示(以及第八圖內更多細節)，使用者系統

712可透過一網路與一隨選資料庫服務，就是系統716，來進行互動。

一隨選資料庫服務，像是系統716，為一建立及/或維護該資料庫系統不需要考慮到的外部使用者可用之資料庫系統，但是當該等使用者需要該資料庫系統時，則可供其使用(例如使用者的隨選)。某些隨選資料庫服務可將來自一或多用戶的資訊儲存到一共用資料庫影像表內，來形成一多用戶資料庫系統(MTS, "multi-tenant database system")。「隨選資料庫服務716」和「系統716」兩詞將交替使用。一資料庫影像可包含一或多個資料庫物件。一關聯性資料庫管理系統(RDMS, "relational database management system")或均等系統可針對該(等)資料庫物件執行儲存以及資訊擷取。應用程式平台718可為一允許系統716的應用程式執行的架構，像是該硬體以及/或軟體，例如該作業系統。在一個具體實施例內，隨選資料庫服務716可包含應用程式平台718，其可建立、管理以及執行由該隨選資料庫服務提供者所發展的一或多應用程式、使用者可透過使用者系統712存取該隨選資料庫服務，或第三方應用程式開發者可透過使用者系統712存取該隨選資料庫服務。

使用者系統712的使用者在其個別能力上有所不同，並且一特定使用者系統712的能力完全由該目前使用者的權限(權限等級)來決定。例如：在一業務員正在使用一特定使用者系統712與系統716互動時，則使用者系統具有分配給該業務員的能力。不過，當一管理員正在使用該使用者系統與系統716互動時，則使用者系統具有分配給該管理員的能力。在具有一階層角色模型的系統內，一個權限等級上的使用者可以存取較低權限等級使用者可存取的應用程式、資料以及資料庫資訊，但是無法存取一較高權限等級使用者可存取的特定應用程式、資料庫資訊以及資料。如此，根據一使用者的安全或權限等級，

不同的使用者將具有不同的能力來存取與修改應用程式和資料庫資訊。

網路714可為任何網路或與其他裝置通訊的裝置網路之組合。例如：網路714可為LAN (區域網路)、WAN (廣域網路)、電話網路、無線網路、點對點網路、星形網路、令符環網路、集線器網路或其他合適組態的任一或任意組合。目前電腦網路中最常用的種類為一TCP/IP (傳輸控制通訊協定與網際網路通訊協定)網路，像是全球互聯網路通常稱之為「網際網路」，以大寫字母「I」表示，此處的許多範例將使用此網路。不過，吾人應該瞭解，雖然TCI/IP為一常用的通訊協定，不過本發明可使用的網路並不受限。

使用者系統712可使用TCP/IP與系統716通訊，並且在一較高網路等級上，使用其他常用網際網路通訊協定來通訊，像是HTTP、FTP、AFS、WAP等等。在使用HTTP的範例中，使用者系統712可包含一通稱為「瀏覽器」的HTTP用戶端，用於傳送與接收HTTP訊息至與來自系統716上的HTTP伺服器。這種HTTP伺服器可實施當成系統716與網路714之間的該單一網路介面，但是也可使用其他技術來取代。在某些實施當中，系統716與網路714之間的介面包含負載共享功能性，像是輪替HTTP要求分配器來平衡負載並且在複數個伺服器上均衡分配傳入的HTTP要求。至少對於存取該伺服器的使用者而言，每一複數個伺服器都具有對於MTS資料的權限；不過，可使用其他替代組態來取代。

在一個具體實施例內，第七圖內所示的系統716實施一網路式客戶關係管理(CRM, “customer relationship management”)系統。例如在一個具體實施例內，系統716包含應用程式伺服器，其設置成實施與執行CRM軟體應用程式，並且提供相關資料、程式碼、表單、網頁與其他資訊至與來自使用者系統712，並且將相關資料、物件以及網路內容儲存至一資料庫系

統或從中擷取。運用一多用戶系統，多用戶的資料可儲存在相同實體資料庫物件內，不過，用戶資料通常排列成一個用戶的資料在邏輯上保持與其他用戶的資料分開，如此一個用戶就不會存取到其他用戶的資料，除非該資料表明被共享。在特定具體實施例內，系統716實施CRM應用程式以外的應用程式。例如：系統716可提供用戶存取多個主要(標準與自訂)應用程式，包含一CRM應用程式。應用程式平台718可支援使用者(或第三方開發者)應用程式，包含或不包含CRM，而應用程式平台718管理應用程式的建立、儲存至一或多個資料庫物件，並在系統716的處理空間內於一虛擬機器內執行該等應用程式。

用於系統716元件的一個排列顯示在第八圖內，包含一網路介面720、應用程式平台718、用戶資料723的用戶資料儲存庫722、可存取至系統716以及多個用戶的系統資料儲存庫724、實施系統716許多功能的程式碼726以及用於執行MTS系統處理與用戶專屬處理，像是執行應用程式當成一應用程式主要服務一部分的一處理空間728。可在系統716上執行的額外處理包含資料庫索引處理。

第七圖內所示系統內的許多元件包含此處簡要解釋的傳統、熟知元件。例如：每一使用者系統712應包含一桌上型個人電腦、工作站、膝上型電腦、PDA、行動電話或任何無線接取通訊協定(WAP, “wireless access protocol”)啟用的裝置，或可直接或間接與網際網路或其他網路連線介接的任何其他計算裝置。使用者系統712通常運行一HTTP用戶端，例如一瀏覽程式，像是微軟的Internet Explorer瀏覽器、網景的Navigator瀏覽器、Opera的瀏覽器或在一行動電話、PDA或其他無線裝置案例中的WAP啟用瀏覽器等等，允許使用者系統712的使用者(例如多用戶資料庫系統的訂戶)透過網路716從系統714存取、處理以及檢視其可用的資訊、網頁與應用程式。每一使用者系統712通常也包含一或多個使用者介面裝置，像是一鍵

盤、一滑鼠、軌跡球、觸碰板、觸碰螢幕、筆等等，與系統716或其他系統或伺服器所提供的網頁、表單、應用程式與其他資訊結合，用於與一顯示器(例如監視器螢幕、LCD顯示器等等)上該瀏覽器所提供的一圖形使用者介面(GUI)互動。例如：該使用者介面裝置可用來存取系統716所主控的資料與應用程式，並且在儲存的資料上執行搜尋，否則允許一使用者與呈現給一使用者的許多GUI頁面互動。如上面所討論，具體實施例適合用網際網路來使用，網際網路就是一特殊目標網路互聯。不過，吾人將瞭解，可使用其他網路來取代網際網路，像是一企業內部網路、一外部網路、一虛擬私人網路(VPN)、一非TCP/IP網路、任何LAN或WAN等等。

根據一個具體實施例，每一使用者系統712和其所有組件都是操作者可用應用程式來組態的，像是一瀏覽器，包含使用像是Intel Pentium®處理器等等一中央處理單元來執行的電腦程式碼。類似地，系統716 (以及MTS的額外實例，其存在一個以上)以及其所有組件都是操作者可用應用程式來組態的，包含使用像是第七圖的處理系統717，包含Intel Pentium®處理器等等，這類中央處理單元來執行的電腦程式碼。一電腦程式產品具體實施例包含一機器可讀取儲存媒體，其上/其內儲存指令用於程式編輯一電腦來執行此處所說明具體實施例的任何處理。如此處所說明用於操作與設置系統716來相互通訊並且處理網頁、應用程式和其他資料與媒體內容之電腦程式碼較佳下載並儲存在一硬碟上，但是該整個程式碼或一部分程式碼也可儲存在任何其他已知的揮發性或非揮發性記憶體媒體或裝置內，像是一ROM或RAM，或提供在可儲存程式碼的任何媒體上，像是任何一種旋轉媒體，包含磁碟片、光碟片、數位多用途光碟(DVD)、小型光碟(CD)、微型磁碟以及光學磁性碟片和磁性或光學卡、奈米系統(包含分子記憶體IC)或任何一種適合儲存指令以及/或資料的媒體或裝置。此外，該整個程式碼

或一部分程式碼可從一軟體來源透過一傳輸媒體，例如透過網際網路，來傳輸與下載，或已知從其他伺服器透過使用任何已知通訊媒體與通訊協定(例如TCP/IP、HTTP、HTTPS、Ethernet等等)的任何其他已知傳統網路連線(例如外部網路、VPN、LAN等等)來傳輸。吾人也將了解，實施本發明具體實施例的電腦程式碼可在任何可於一用戶端系統和/或伺服器或伺服器系統上執行的程式編輯語言內實施，例如C、C++、HTML、任何其他標記語言、Java™、JavaScript、ActiveX、任何其他腳本語言，像是VBScript，以及許多其他已知的程式編輯語言都可使用(Java™是SUN Microsystems, Inc. 的商標)。

根據一個具體實施例，每一系統716都可組態成提供網頁、表單、應用程式、資料以及媒體內容給使用者(用戶端)系統712，來支援由使用者系統712存取當成系統716的用戶。如此，除非資料共享，否則系統716提供安全機制來保持每一用戶的資料分開。若使用超過一個MTS，則其彼此之間的位置相當靠近(例如位於一單一建築物或校園內的一伺服器群)，或其彼此之間的位置可分散(例如一或多個伺服器位於A城市，並且一或多個伺服器位於B城市)。如此處所使用，每一MTS都應包含一或多個區域分佈或橫跨一或多個地理位置的邏輯以及/或實體相連伺服器。此外，「伺服器」該詞表示包含一電腦系統，該系統包含處理硬體以及處理空間，以及業界內已知的一相關儲存系統與資料庫應用程式(例如OODBMS或RDBMS)。吾人應該瞭解，此處通常交換使用該「伺服器系統」與「伺服器」。類似地，此處所說明的資料庫物件可實施當成單一資料庫、一分散式資料庫、一分散式資料庫的集合、一具有冗餘上線或離線備份或其他冗餘等等的資料庫，並且可包含一分散式資料庫或儲存網路以及相關處理智慧。

第八圖也說明環境 710。不過在第八圖內，進一步說明系統 716 的元件以及具體實施例內許多互聯關係。第八圖顯示使

用者系統 712 可包含處理器系統 712A、記憶體系統 712B、輸入系統 712C 以及輸出系統 712D。第八圖顯示網路 714 和系統 716。第八圖也顯示系統 716 包含用戶資料儲存庫 722、用戶資料 723、系統資料儲存庫 724、系統資料 725、使用者介面(UI) 830、應用程式介面(API) 832、程式語言/SalesForce 物件查詢語言(PL/SOQL) 834、儲存常式 836、應用程式設定機制 838、應用程式伺服器 800_1-800_N、系統處理空間 802、用戶處理空間 804、用戶管理處理空間 810、用戶儲存區 812、使用者儲存庫 814 以及應用程式中繼資料 816。在其他具體實施例內，環境 710 可能不會具有與上列相同的元件並且/或具有其他元件來取代或加入上列元件。

使用者系統712、網路714、系統716、用戶資料儲存庫722以及系統資料儲存庫724已於上面第七圖內討論過。有關使用者系統712，處理器系統712A可為一或多個處理器的任意組合。記憶體系統712B可為一或多個記憶體裝置、短期以及/或長期記憶體的任意組合。輸入系統712C可為輸入裝置的任意組合，像是一或多個鍵盤、滑鼠、軌跡球、掃瞄器、數位相機以及/或網路介面。輸出系統712D可為輸出裝置的任意組合，像是一或多個監視器、印表機以及/或網路介面。如第八圖所示，系統716可包含(第七圖的)一網路介面720實施當成一組HTTP應用程式伺服器800、一個應用程式平台718、用戶資料儲存庫722以及系統資料儲存庫724。同時顯示的有系統處理空間802，其中包含個別用戶處理空間804以及一用戶管理處理空間810。每一應用程式伺服器800都可組態成用戶資料儲存庫722和其中的用戶資料723，以及系統資料儲存庫724和其中的系統資料725，來服務使用者系統712的要求。用戶資料723可分成個別用戶儲存區812，其可為資料的一實體排列以及/或一邏輯排列。在每一用戶儲存區812內，使用者儲存庫814和應用程式中繼資料816可類似分配給每一使用者。例如：一使用者

最近使用("MRU", most recently used)項目的一複本可儲存至使用者儲存庫814內。類似地，屬於用戶的一整個組織之MRU項目一複本也儲存至用戶儲存區812。一UI 830提供一使用者介面並且一API 832提供一應用程式設計師介面，讓系統716保留處理給使用者系統712上的使用者以及/或程式設計師。該用戶資料以及該系統資料可儲存在許多資料庫內，像是一或多個Oracle™資料庫。

應用程式平台718包含支援應用程式設計師建立與管理應用程式的應用程式設定機制838，儲存常式836可將其當成中繼資料儲存至用戶資料儲存庫722內，由訂戶執行當成例如受用戶管理處理810管理的一或多個用戶處理空間804。運用PL/SOQL 834提供一程式編輯語言風格介面延伸給API 832，來對這種應用程式的召喚進行編碼。某些PL/SOQL語言具體實施例的詳細說明都在由Craig Weissman於2006年10月4日提出，標題為「PROGRAMMING LANGUAGE METHOD AND SYSTEM FOR EXTENDING APIS TO EXECUTE IN CONJUNCTION WITH DATABASE APIS」的共同擁有美國臨時專利申請案第60/828,192號內討論，其在此完整併入當成參考。一或多個系統處理可偵測到對於應用程式的召喚，其管理擷取應用程式中繼資料816讓該訂戶進行召喚並執行該中繼資料當成一虛擬機器內的一應用程式。

每一應用程式伺服器800可透過一不同網路連線通訊耦合至資料庫系統，例如具有存取至系統資料725和用戶資料723的能力。例如：一個應用程式伺服器800_1可透過網路714 (例如網際網路)耦合，其他應用程式伺服器800_{N-1}可透過直接網路鏈結來耦合，並且其他應用程式伺服器800_N可透過一不同網路連線來耦合。傳輸控制通訊協定以及網際網路通訊協定(TCP/IP)為應用程式伺服器800與該資料庫系統之間一常見的通訊協

定，不過精通此技術的人士將瞭解，根據使用的網路互連，可使用其他傳輸通訊協定來將系統最佳化。

在特定具體實施例內，每一應用程式伺服器800都組態成處理與屬於一戶的任何組織相關聯之任何使用者要求。因為要可隨時從該伺服器池新增與移除伺服器，所以較佳不要有伺服器與一使用者有關聯並且/或組織與一特定應用程式伺服器800有關聯。因此在一個具體實施例內，一實施負載平衡功能的一介面系統(例如一F5 Big-IP負載平衡器)通訊耦合在應用程式伺服器800與使用者系統712之間，來將要求分配至應用程式伺服器800。在一個具體實施例內，該負載平衡器使用最新連線演算法來將使用者要求繞送至應用程式伺服器800。也可使用負載平衡演算法的其他範例，像是輪替與觀察回應時間。例如：在特定具體實施例內，來自相同使用者的三個連續要求命中三個不同的應用程式伺服器800，並且來自不同使用者的三個要求命中相同的應用程式伺服器800。在此方式中，系統716為多用戶，其中系統716處理橫跨分散使用者與組織的不同物件、資料以及應用程式之儲存和存取。

針對一儲存的範例，一個用戶可為一銷售公司，其中每一業務員都使用系統716來管理其銷售處理。如此，依使用者可維護聯絡資料、領先資料、客戶追蹤資料、績效資料、目標與進度資料等等，這全都適用於使用者的個人業務處理(例如在用戶資料儲存庫722內)。在一MTS配置的範例中，因為要存取、檢視、修正、回報、傳輸、計算等等的所有資料與應用程式都可由只具備網路存取的使用者系統所維護與存取，則該使用者可管理其業務成果，並且循環用於任何許多不同的使用者系統。例如：若一業務員拜訪一客戶並且該客戶接待大廳那邊可以連接網際網路，則該業務員可在大廳內等待時獲得有關該客戶的重要更新。

雖然每一使用者的資料可從其他使用者的資料分離，而不管每一使用者的僱主，某些資料可為全組織共享的資料或可由當成用戶的一已知組織內許多或全部使用者存取。如此，可能會有某些資料結構由分配在該用戶等級上的系統716來管理，而其他資料結構由該使用者等級上的系統來管理。因為一MTS可支援包含可能競爭者的多個用戶，該MTS應該具有安全通訊協定來維持資料、應用程式以及應用程式使用分離。另外，因為許多用戶可能選擇存取至一MTS而非維護自己的系統，則冗餘、執行以及備份為可能在MTS內實施的額外功能。除了一使用者特定資料與用戶特定資料以外，系統716也可維護多個用戶可使用的系統等級資料或其他資料。這種系統等級資料可包含在用戶之間共享的產業報表、新聞、佈告等等。

在特定具體實施例內，使用者系統712 (可為用戶端系統)與應用程式伺服器800通訊，來要求並更新來自系統716的系統等級與用戶等級資料，該系統可要求傳送一或多個要求至用戶資料儲存庫722以及/或系統資料儲存庫724。系統716 (例如系統716內的應用程式伺服器800)自動產生一或多個設計來存取所要資訊的SQL陳述式(例如一或多SQL查詢)。系統資料儲存庫724可產生查詢計畫來存取來自該資料庫的要求資料。

每一資料庫都可檢視為物件的集合，像是一組邏輯表，其包含裝入預定類別的資料。一「表格」為一資料物件的一個代表，並且可用根據本發明來簡化物件與自訂物件的概念說明。吾人應該瞭解，此處通常交換使用「表格」與「物件」。每一表格一般包含一或多個邏輯上以可一檢視法則配置成列或欄位之資料類別。一表格的每一行或記錄都包含資料的實例，用於該等欄位定義的每一類別。例如：一CRM資料庫可包含一個用基本聯絡人資訊欄位，像是姓名、地址、電話號碼、傳真號碼等等，說明客戶的表格。其他表格可說明一採購訂單，其中包含資訊欄位，像是客戶、產品、銷售價格、日期等等。在

某些多用戶資料庫系統中，可提供此標準記錄表格給所有用戶的使用者。對於CRM資料庫應用而言，這種標準記錄可包含帳號、聯絡人、領先以及機會資料的表格，每一都包含預定的欄位。吾人應該瞭解，此處通常交換使用「記錄」、「物件」和「表格」這些詞。

在某些多用戶資料庫系統內，可允許用戶建立並儲存客戶物件，或允許其自訂標準記錄或物件，例如利用建立自訂欄位或標準物件，其中包含自訂索引欄位。於2004年4月2日提出在此併入當成參考，標題為「CUSTOM ENTITIES AND FIELDS IN A MULTI-TENANT DATABASE SYSTEM」，序號第10/817,161號的美國專利申請案揭示用於建立自訂物件以及在一多用戶資料庫系統內自訂標準物件之系統及方法。在特定具體實施例內，例如所有客戶記錄資料行都儲存在一單一多用戶實體表格內，該多用戶實體表格中每一組織可包含多個邏輯表格。對於客戶來說清楚其多個「表格」事實上儲存在一個大表格內，或其資料可儲存在與其他客戶資料相同的表格內。

吾人應該注意，此處所說明的任何不同具體實施例都可或不可配備下列一或多項公佈申請案中所公佈的一或多種特性集合：2002年11月4日提出編號為US2003/0233404，標題為「OFFLINE SIMULATION OF ONLINE SESSION BETWEEN CLIENT AND SERVER」之申請案、2003年4月17日題出編號為US2004/0210909，標題為「JAVA OBJECT CACHE SERVER FOR DATABASES」之申請案、此時被授與美國專利第7,209,929號、2003年9月23日提出編號為US2005/0065925，標題為「QUERY OPTIMIZATION IN A MULTI-TENANT DATABASE SYSTEM」之申請案、2004年4月2日提出編號為US2005/0223022，標題為「CUSTOM ENTITIES AND FIELDS IN A MULTI-TENANT DATABASE SYSTEM」之申請案、2004年6月16日提出編號為US2005/0283478，標題為

「SOAP-BASED WEB SERVICES IN A MULTI-TENANT DATABASE SYSTEM」之申請案以及/或2005年3月8日提出編號為US2006/0206834，標題為「SYSTEMS AND METHODS FOR IMPLEMENTING MULTI-APPLICATION TABS AND TAB SETS」之申請案；這些每一都在此完整併入當成參考。

雖然已經根據範例以及特定具體實施例來說明本發明，吾人可瞭解本發明並不受限於公佈的具體實施例。相反地，本發明用於涵蓋精通此技術人士應瞭解的許多修改和類似組態。因此，下列申請專利範圍的範疇將以廣義方式來解釋，如此涵蓋所有這種修改以及類似配置。

【圖式簡單說明】

第一圖說明根據具體實施例，一用於確認與一隨選資料庫服務相關聯的一已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統。

第二圖說明根據具體實施例，一用於決定與一隨選資料庫服務相關聯的一已開發應用程式是否將與用至少一其他應用程式搭配來正常運作之方法。

第三圖顯示根據其他具體實施例，一用於確認與一隨選資料庫服務相關聯的已開發應用程式將可與至少一其他應用程式搭配來正常運作之系統。

第四圖顯示根據具體實施例，一用於實行在一已開發應用程式內包含測試之方法。

第五圖顯示根據具體實施例，一用於驗證一已開發應用程式之方法。

第六圖顯示根據具體實施例，一用於一隨選資料庫服務內程式指令一適當版本的執行時間喚起之方法。

第七圖顯示一其中使用一隨選資料庫服務的環境範例之方塊圖。

第八圖說明第七圖元件以及這些元件之間許多可能互連的具體實施例方塊圖。

【主要元件符號說明】

100 系統
102 隨選資料庫服務
104 開發者
106 用戶
300 系統
302 隨選資料庫服務
304 開發者
306 末端使用者用戶
308 網路
310 應用程式伺服器
312 資料庫
314 執行時間解譯器
700 方塊圖
710 環境
712 使用者系統
712A 處理器系統
712B 記憶體系統
712C 輸入系統
712D 輸出系統
714 網路
716 系統
717 處理器系統
718 應用程式平台
720 網路介面
722 用戶資料儲存庫
723 用戶資料
724 系統資料儲存庫
725 系統資料
726 程式碼
728 處理空間
800_1-800_N 應用程式伺服器
802 系統處理空間
804 用戶處理空間
810 用戶管理處理空間
812 用戶儲存區
814 使用者儲存庫
816 應用程式中繼資料
830 使用者介面
832 應用程式介面
834 PL/SOQL
836 儲存常式
838 應用程式設定機構

十、申請專利範圍：

1. 一種方法，包含：

在一隨選資料庫服務上，從一開發者接收一已開發應用程式，該已開發應用程式包括一應用程式的一更新版本；

建立描述該已開發應用程式複數個態樣的一資料結構；

識別與至少一訂戶的至少一其他應用程式相關聯之資訊，該已開發應用程式欲與其一起操作，該至少一其他應用程式包括該應用程式的一較舊版本，該更新版本是由該較舊版本發展而來；以及

運用該資料結構與該資訊，藉

識別使用於該至少一訂戶(包括該至少一其他應用程式)的一應用程式環境的設定，使用與該至少一其他應用程式相關聯的資訊；

創造一具有被識別的該設定的暫時測試環境；

由該設定決定該應用程式的一版本作為測試；

使用該資料結構，識別該已開發應用程式的一部分，其包括作為測試的該應用程式的該版本；

於該暫時測試環境中，測試該已開發應用程式的已識別的該部分；

以判斷該已開發應用程式與該至少一其他應用程式是否將可正常一起操作。

2. 如申請專利範圍第 1 項之方法，其中該等態樣包含一版本識別碼。
3. 如申請專利範圍第 1 項之方法，其中該等態樣包含一法則。
4. 如申請專利範圍第 1 項之方法，其中該等態樣包含一訊息外型。
5. 如申請專利範圍第 1 項之方法，其中該等態樣包含一語意行為。

6. 如申請專利範圍第 1 項之方法，其中該等態樣包含一致資源定位器。

7. 如申請專利範圍第 1 項之方法，其中允許該至少一訂戶與該開發者編輯該已開發應用程式的至少一組件。

8. 如申請專利範圍第 7 項之方法，其中該至少一組件包含一欄位。

9. 如申請專利範圍第 1 項之方法，並且進一步包含追蹤複數個該訂戶運用該已開發應用程式複數個版本中哪一個。

10. 如申請專利範圍第 1 項之方法，若判斷該已開發應用程式與該至少一其他應用程式將無法一起正常運作，則允許至少該開發者與該訂戶之一編輯該已開發應用程式。

11. 一種攜帶一或多指令序列的機器可讀取媒體，這些指令由一或多處理器執行時，導致該一或多處理器執行下列步驟：

在一隨選資料庫服務上，從一開發者接收一已開發應用程式，該已開發應用程式包括一應用程式的一更新版本；

建立描述該已開發應用程式複數個態樣的一資料結構；

識別與至少一訂戶的至少一其他應用程式相關聯之資訊，該已開發應用程式欲與其一起操作，該至少一其他應用程式包括該應用程式的一較舊版本，該更新版本是由該較舊版本發展而來；以及

運用該資料結構與該資訊，藉

識別使用於該至少一訂戶(包括該至少一其他應用程式)的一應用程式環境的設定，使用與該至少一其他應用程式相關聯的資訊；

創造一具有被識別的該設定的暫時測試環境；

由該設定決定該應用程式的一版本作為測試；

使用該資料結構，識別該已開發應用程式的一部分，其包括作為測試的該應用程式的該版本；

於該暫時測試環境中，測試該已開發應用程式的已識別的該部分；

以判斷該已開發應用程式與該至少一其他應用程式是否將可正常一起操作。

12. 一種設備，其包括：

一處理器；以及

一或多儲存的指令序列，當這些指令由該處理器執行時，導致該處理器執行下列步驟：

在一隨選資料庫服務上，從一開發者接收一已開發應用程式，該已開發應用程式包括一應用程式的一更新版本；

建立描述該已開發應用程式複數個態樣的一資料結構；

識別與至少一訂戶的至少一其他應用程式相關聯之資訊，該已開發應用程式欲與其一起操作，該至少一其他應用程式包括該應用程式的一較舊版本，該更新版本是由該較舊版本發展而來；以及

運用該資料結構與該資訊，藉

識別使用於該至少一訂戶(包括該至少一其他應用程式)的一應用程式環境的設定，使用與該至少一其他應用程式相關聯的資訊；

創造一具有被識別的該設定的暫時測試環境；

由該設定決定該應用程式的一版本作為測試；

使用該資料結構，識別該已開發應用程式的一部分，其包括作為測試的該應用程式的該版本；

於該暫時測試環境中，測試該已開發應用程式的已識別的該部分；

以判斷該已開發應用程式與該至少一其他應用程式是否將可正常一起操作。

13. 一種用於在一傳輸媒體上一多用戶資料庫系統內傳輸程式

碼之方法，該方法包含：

傳輸用於識別一隨選資料庫服務內一開發者的一已開發應用程式之程式碼，該已開發應用程式包括一應用程式的一更新版本；

傳輸用於建立描述該已開發應用程式複數個態樣的一資料結構之程式碼；

傳輸用於識別與至少一訂戶的至少一其他應用程式相關聯的資訊之程式碼，該已開發應用程式欲與其一起操作，該至少一其他應用程式包括該應用程式的一較舊版本，該更新版本是由該較舊版本發展而來；以及

傳輸用於運用該資料結構與該資訊，藉

識別使用於該至少一訂戶(包括該至少一其他應用程式)的一應用程式環境的設定，使用與該至少一其他應用程式相關聯的資訊；

創造一具有被識別的該設定的暫時測試環境；

由該設定決定該應用程式的一版本作為測試；

使用該資料結構，識別該已開發應用程式的一部分，其包括作為測試的該應用程式的該版本；

於該暫時測試環境中，測試該已開發應用程式的已識別的該部分；

以判斷該已開發應用程式與該至少一其他應用程式是否將可正常一起操作之程式碼。

14. 如申請專利範圍第 13 項之方法，其中該等態樣包含一版本識別碼。

15. 如申請專利範圍第 13 項之方法，其中該等態樣包含一法則。

16. 如申請專利範圍第 13 項之方法，其中該等態樣包含一訊息外型。

17. 如申請專利範圍第 13 項之方法，其中該等態樣包含一語意行為。

18. 如申請專利範圍第 13 項之方法，其中該等態樣包含一致資源定位器。

19. 如申請專利範圍第 13 項之方法，其中允許該至少一訂戶與該開發者編輯該已開發應用程式的至少一組件。

20. 如申請專利範圍第 19 項之方法，其中該至少一組件包含一欄位。

21. 一種方法，包含：

在一隨選資料庫服務上，接收一已開發應用程式的至少一方面，該已開發應用程式的該至少一方面由一第一使用者透過一網路瀏覽器於一第一電腦上接收；

透過該第一使用者的網路瀏覽器，提供一指示給該第一使用者，說明該已開發應用程式的該至少一方面已被驗證；

代表一第二使用者，接收一要求以使用該已開發應用程式的該至少一方面，該要求是回應於該第二使用者使用該已開發應用程式的一第二方面；

其中，若該要求在提供該第一使用者該已開發應用程式的該至少一方面已被驗證的該指示前被接收，該第二使用者就不能使用該已開發應用程式的該至少一方面，且該第二使用者的要求會被拒絕。

22. 如申請專利範圍第 21 項之方法，其中若該已開發應用程式的該至少一方面未驗證，則不會散佈該已開發應用程式。

23. 如申請專利範圍第 21 項之方法，其中若該已開發應用程式的該至少一方面已驗證，則會散佈該已開發應用程式。

24. 如申請專利範圍第 21 項之方法，其中利用推送技術散佈該已開發應用程式。

25. 如申請專利範圍第 21 項之方法，其中利用推送技術非同步散佈該已開發應用程式。

26. 如申請專利範圍第 21 項之方法，其中該驗證包含在該已開發應用程式的該至少一方面上運行至少一測試。

27. 如申請專利範圍第 31 項之方法，其中該至少一測試包含於該已開發應用程式。

28. 如申請專利範圍第 31 項之方法，進一步包含判斷該至少一測試是否運行成為前一版本的該已開發應用程式。

29. 如申請專利範圍第 21 項之方法，其中該隨選資料庫服務包含一多用戶隨選服務。

30. 如申請專利範圍第 21 項之方法，其中該驗證包含判斷該已開發應用程式的該至少一方面內是否呈現電腦程式碼涵蓋的一預定等級。

31. 如申請專利範圍第 37 項之方法，其中若該已開發應用程式內的該至少一方面呈現電腦程式碼涵蓋的該預定等級，則驗證該已開發應用程式。

32. 如申請專利範圍第 21 項之方法，其中該已開發應用程式包含至少一測試，從該訂戶的觀點來說，以在執行時間內判斷該隨選資料庫服務的一訂戶所使用之該已開發應用程式的版本。

33. 一種攜帶一或多指令序列的機器可讀取媒體，這些指令由一或多處理器執行時，導致該一或多處理器執行下列步驟：

在一隨選資料庫服務上，接收一已開發應用程式的至少一方面，該已開發應用程式的該至少一方面由一第一使用者透過一網路瀏覽器於一第一電腦上接收；

透過該第一使用者的網路瀏覽器，提供一指示給該第一使用者，說明該已開發應用程式的該至少一方面已被驗證；

代表一第二使用者，接收一要求以使用該已開發應用程式的該至少一方面，該要求是回應於該第二使用者使用

該已開發應用程式的一第二方面；

其中，若該要求在提供該第一使用者該已開發應用程式的該至少一方面已被驗證的該指示前被接收，該第二使用者就不能使用該已開發應用程式的該至少一方面，且該第二使用者的要求會被拒絕。

34. 一種設備，其包括：

一處理器；以及

一或多儲存的指令序列，當這些指令由該處理器執行時，導致該處理器執行下列步驟：

在一隨選資料庫服務上，接收一已開發應用程式的至少一方面，該已開發應用程式的該至少一方面由一第一使用者透過一網路瀏覽器於一第一電腦上接收；

透過該第一使用者的網路瀏覽器，提供一指示給該第一使用者，說明該已開發應用程式的該至少一方面已被驗證；

代表一第二使用者，接收一要求以使用該已開發應用程式的該至少一方面，該要求是回應於該第二使用者使用該已開發應用程式的一第二方面；

其中，若該要求在提供該第一使用者該已開發應用程式的該至少一方面已被驗證的該指示前被接收，該第二使用者就不能使用該已開發應用程式的該至少一方面，且該第二使用者的要求會被拒絕。

35. 一種用於在一傳輸媒體上一多用戶資料庫系統內傳輸程式碼之方法，該方法包含：

傳輸用於在一隨選資料庫服務上，接收一已開發應用程式的至少一方面，該已開發應用程式的該至少一方面由一第一使用者透過一網路瀏覽器於一第一電腦上接收之程式碼；

傳輸透過該第一使用者的網路瀏覽器，提供一指示給

該第一使用者，說明該已開發應用程式的該至少一方面已被驗證；

代表一第二使用者，接收一要求以使用該已開發應用程式的該至少一方面，該要求是回應於該第二使用者使用該已開發應用程式的一第二方面；

其中，若該要求在提供該第一使用者該已開發應用程式的該至少一方面已被驗證的該指示前被接收，該第二使用者就不能使用該已開發應用程式的該至少一方面，且該第二使用者的要求會被拒絕。

36. 如申請專利範圍第 35 項之方法，其中若該已開發應用程式的該至少一方面未驗證，則不會散佈該已開發應用程式。

37. 如申請專利範圍第 35 項之方法，其中若該已開發應用程式的該至少一方面被驗證，則會散佈該已開發應用程式。

38. 如申請專利範圍第 35 項之方法，其中利用推送技術散佈該已開發應用程式的該至少一方面。

39. 如申請專利範圍第 35 項之方法，其中利用推送技術非同步散佈該已開發應用程式的該至少一方面。

40. 一種方法，包含：

於一應用程式中，在一隨選資料庫服務上接收一應用程式的程式指令，該程式指令包括多個不同部分，其中該程式指令的各部分各包括該應用程式的不同版本，其中該些版本中至少一版本是該些版本中另一版本的更新；

判斷要喚起的該應用程式的該些不同版本的其中之一版本；以及

於該應用程式中，喚起該程式指令的該些部分的其中之一部分，該應用程式包括該應用程式的該判斷版本，以喚起該應用程式的該判斷版本。

41. 如申請專利範圍第 40 項之方法，其中利用與該隨選資

料庫服務的一訂戶相關聯之版本資訊來判斷該版本。

42. 如申請專利範圍第 41 項之方法，其中包含於該程式指令的一測試被套用至該版本資訊，以辨識要喚起的該等程式指令之該版本。

43. 如申請專利範圍第 40 項之方法，其中若判斷要喚起該程式指令的一第一版本，則會喚起接收的該應用程式中的該程式指令的一第一部分，並且若判斷要喚起該程式指令的一第二版本，則會喚起接收的該應用程式中的該程式指令的一第二部分。

44. 如申請專利範圍第 43 項之方法，其中該應用程式的第一版本與該應用程式的第二版本在至少一方法上不同。

45. 如申請專利範圍第 43 項之方法，其中該應用程式的第一版本與該應用程式的第二版本在至少一類別上不同。

46. 如申請專利範圍第 43 項之方法，其中該應用程式的第一版本與該應用程式的第二版本在至少一變數上不同。

47. 如申請專利範圍第 43 項之方法，其中該應用程式的第一版本為一先前版本，並且該應用程式的該第二版本為一較新版本。

48. 如申請專利範圍第 48 項之方法，其中根據一預定格式來標示該應用程式的第一版本。

49. 如申請專利範圍第 40 項之方法，其中該隨選資料庫服務包含一多用戶隨選服務。

50. 如申請專利範圍第 40 項之方法，其中該應用程式被推送至複數個訂戶。

51. 如申請專利範圍第 50 項之方法，其中該應用程式被非同步推送至複數個訂戶。

52. 一種攜帶一或多指令序列的機器可讀取媒體，這些指令由一或多處理器執行時，導致該一或多處理器執行下列步驟：

於一應用程式中，在一隨選資料庫服務上接收一應用程式的程式指令，該程式指令包括多個不同部分，其中該程式指令的各部分各包括該應用程式的不同版本，其中該些版本中至少一版本是該些版本中另一版本的更新；

判斷要喚起的該應用程式的該些不同版本的其中之一版本；以及

於該應用程式中，喚起該程式指令的該些部分的其中之一部分，該應用程式包括該應用程式的該判斷版本，以喚起該應用程式的該判斷版本。

53. 一種設備，其包括：

一處理器；以及

一或多儲存的指令序列，當這些指令由該處理器執行時，導致該處理器執行下列步驟：

於一應用程式中，在一隨選資料庫服務上接收一應用程式的程式指令，該程式指令包括多個不同部分，其中該程式指令的各部分各包括該應用程式的不同版本，其中該些版本中至少一版本是該些版本中另一版本的更新；

判斷要喚起的該應用程式的該些不同版本的其中之一版本；以及

於該應用程式中，喚起該程式指令的該些部分的其中之一部分，該應用程式包括該應用程式的該判斷版本，以喚起該應用程式的該判斷版本。

54. 一種用於在一傳輸媒體上一多用戶資料庫系統內一傳輸程式碼之方法，該方法包含：

傳輸用於在一應用程式中，在一隨選資料庫服務上接收一應用程式的程式指令，該程式指令包括多個不同部分，其中該程式指令的各部分各包括該應用程式的不同版本，其中該些版本中至少一版本是該些版本中另一版本的更新之程式碼；

傳輸用於判斷要喚起的該應用程式的該些不同版本的其中之一版本之程式碼；以及

傳輸用於於該應用程式中，喚起該程式指令的該些部分的其中之一部分，該應用程式包括該應用程式的該判斷版本，以喚起該應用程式的該判斷版本之程式碼。

55. 如申請專利範圍第 54 項之方法，其中利用與該隨選資料庫服務的一訂戶相關聯之版本資訊來判斷該版本。

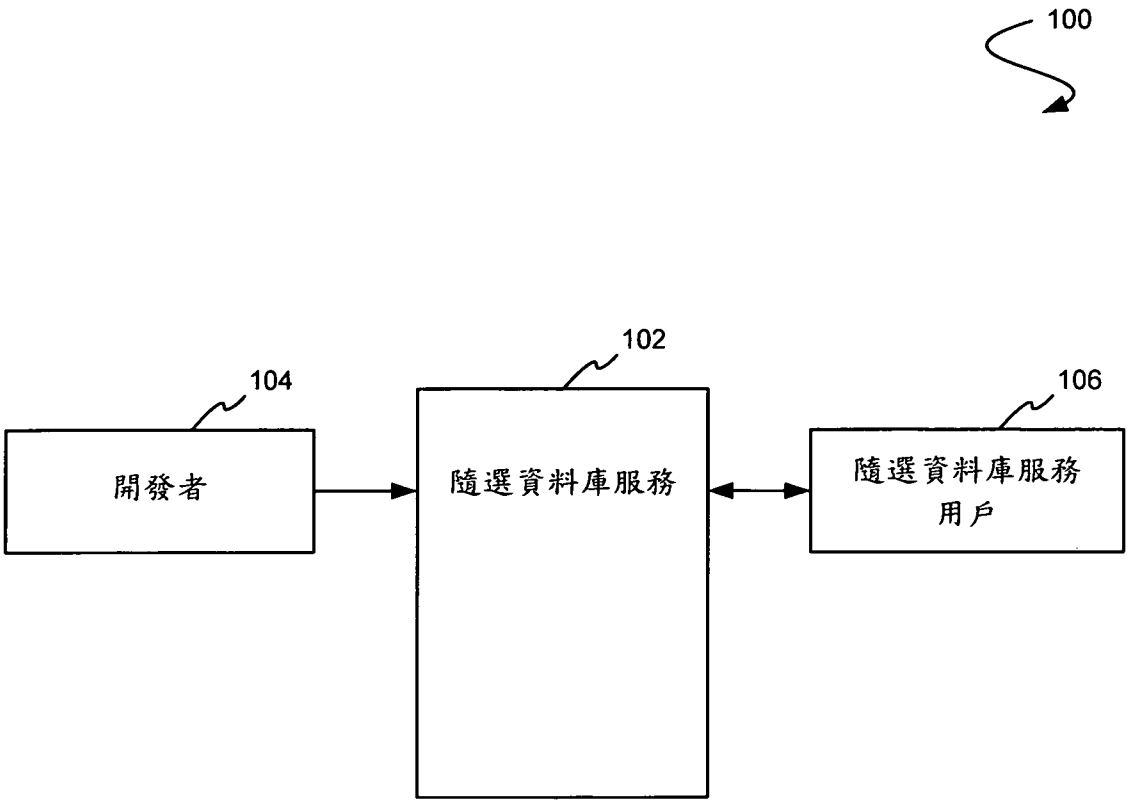
56. 如申請專利範圍第 55 項之方法，其中包含於該等程式指令的一測試被套用至該版本資訊，以辨識要喚起的該應用程式之該版本。

57. 如申請專利範圍第 54 項之方法，其中若判斷要喚起該應用程式的一第一版本，則會喚起該程式指令的一第一部分，並且若判斷要喚起該應用程式的一第二版本，則會喚起該程式指令的一第二部分。

58. 如申請專利範圍第 57 項之方法，其中該應用程式的第一版本與該應用程式的第二版本均包含在該應用程式內。

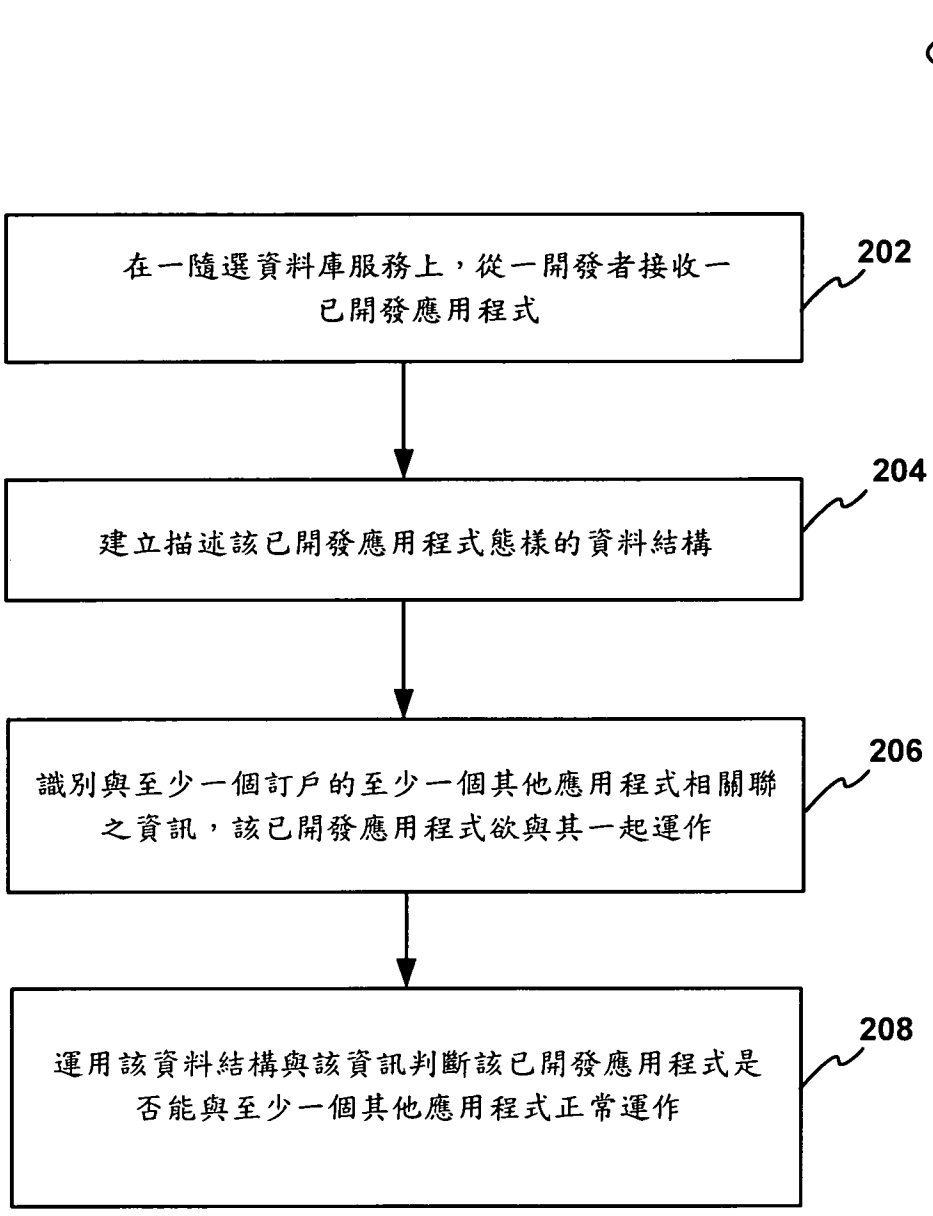
102 年 6 月 27 日修正替換本

十一、圖式：



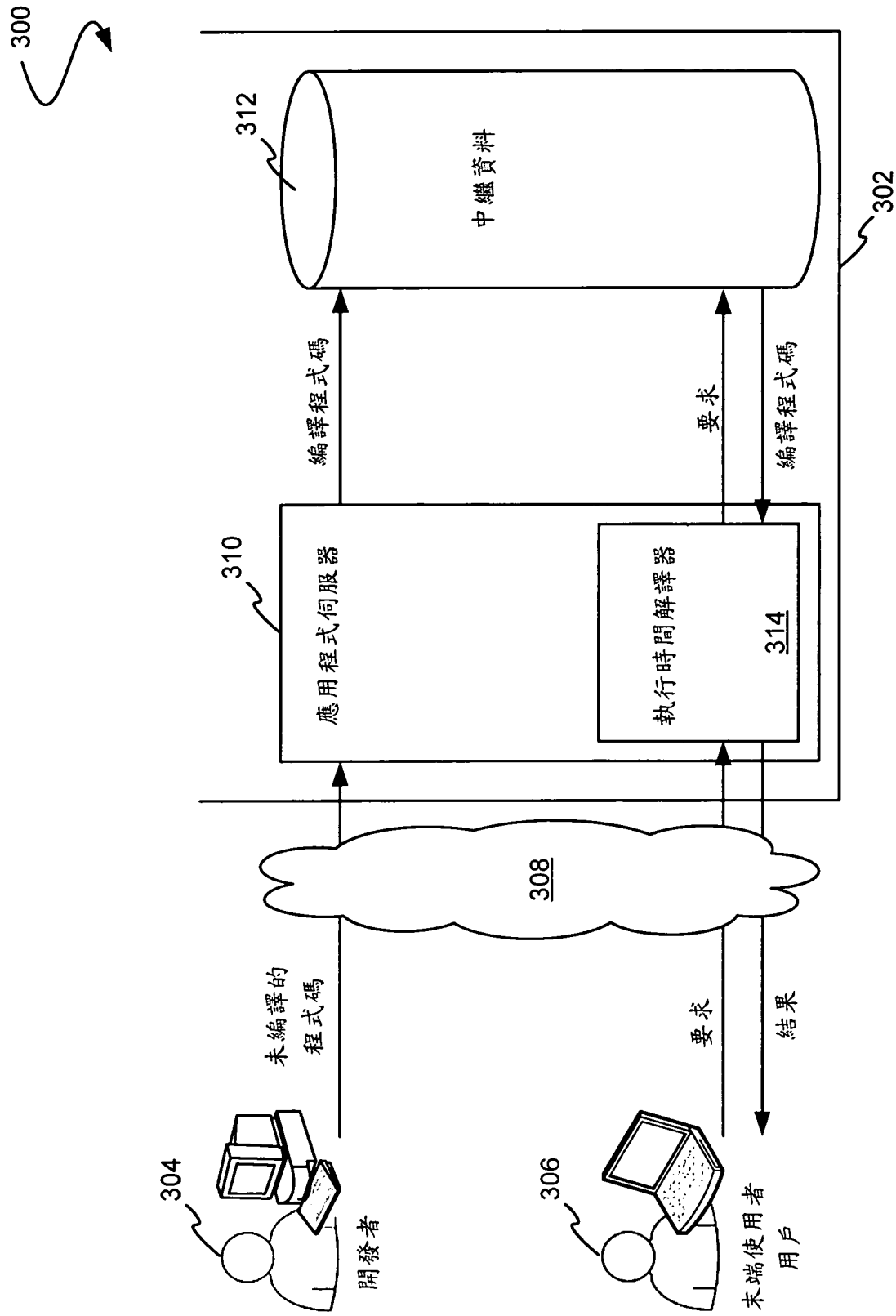
第一圖

102 年 6 月 27 日修正替換本

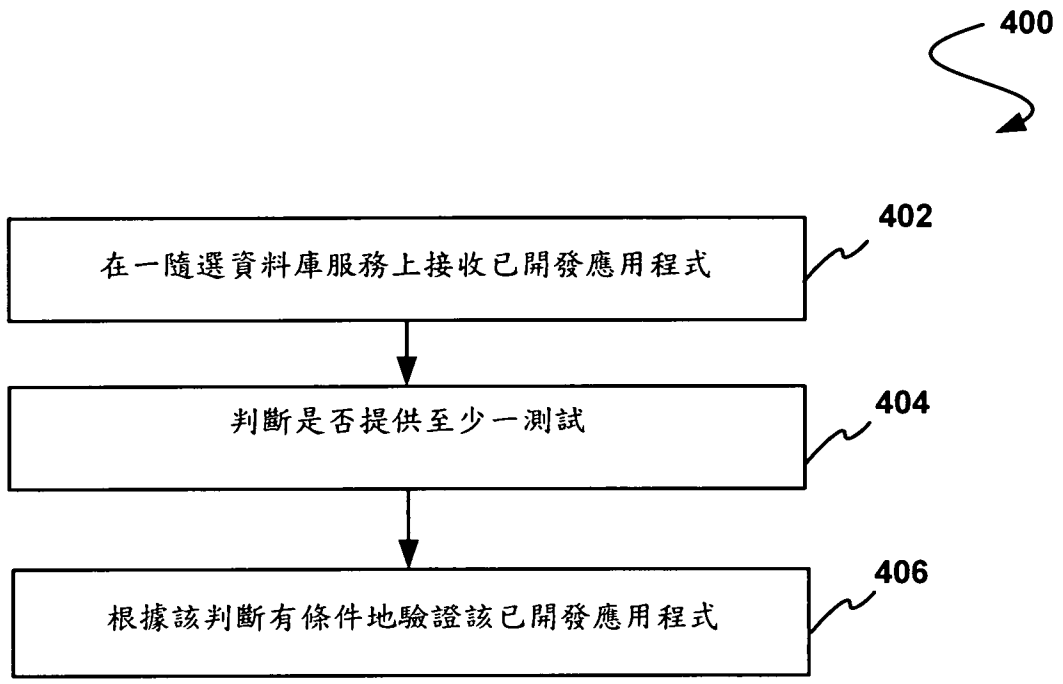


第二圖

102 年 6 月 27 日修正替換本

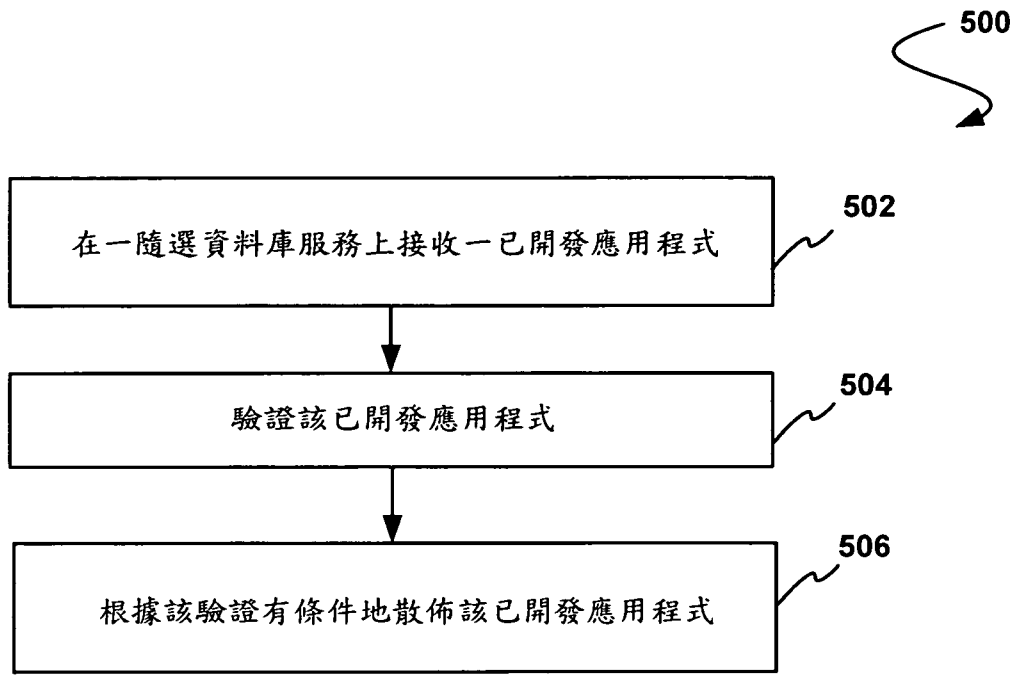


第三圖

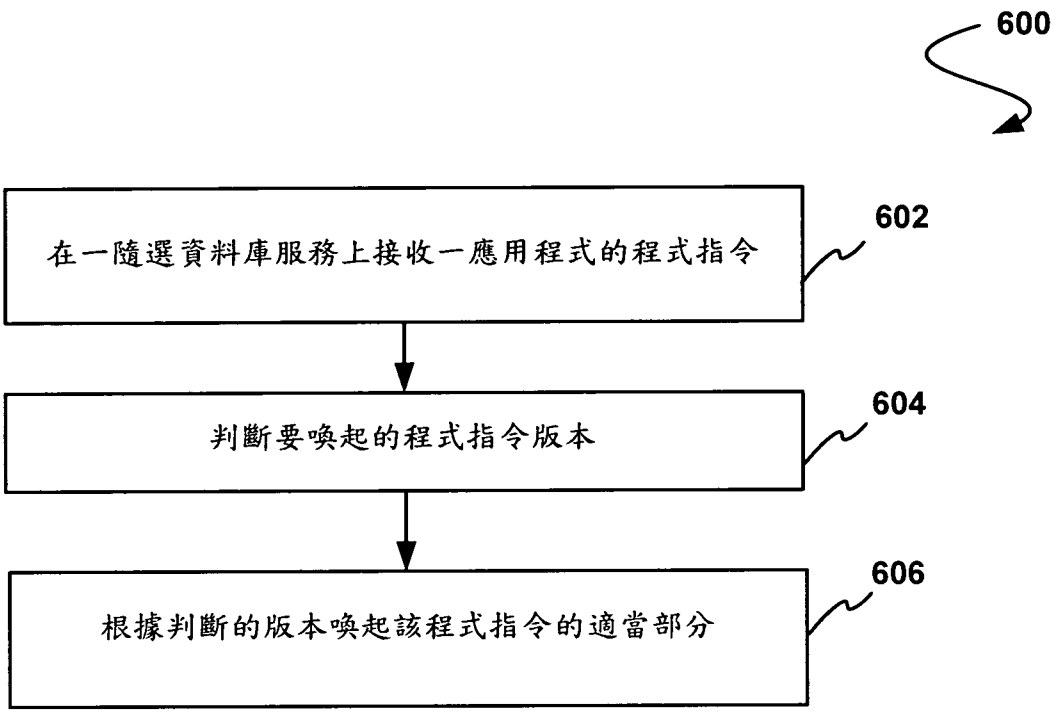

400
402
在一隨選資料庫服務上接收已開發應用程式
404
判斷是否提供至少一測試
406
根據該判斷有條件地驗證該已開發應用程式

第四圖

102 年 6 月 27 日修正替換本

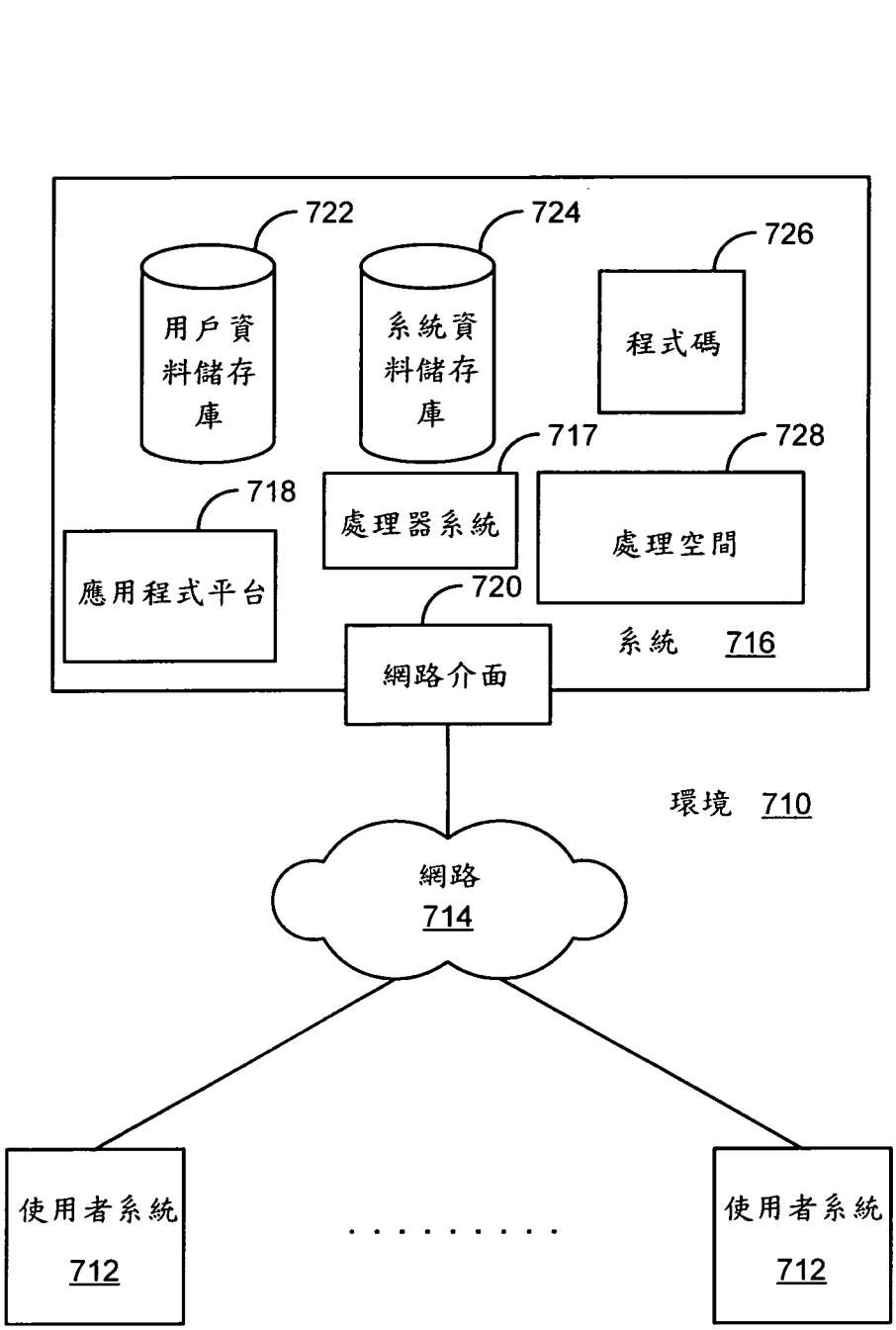


第五圖

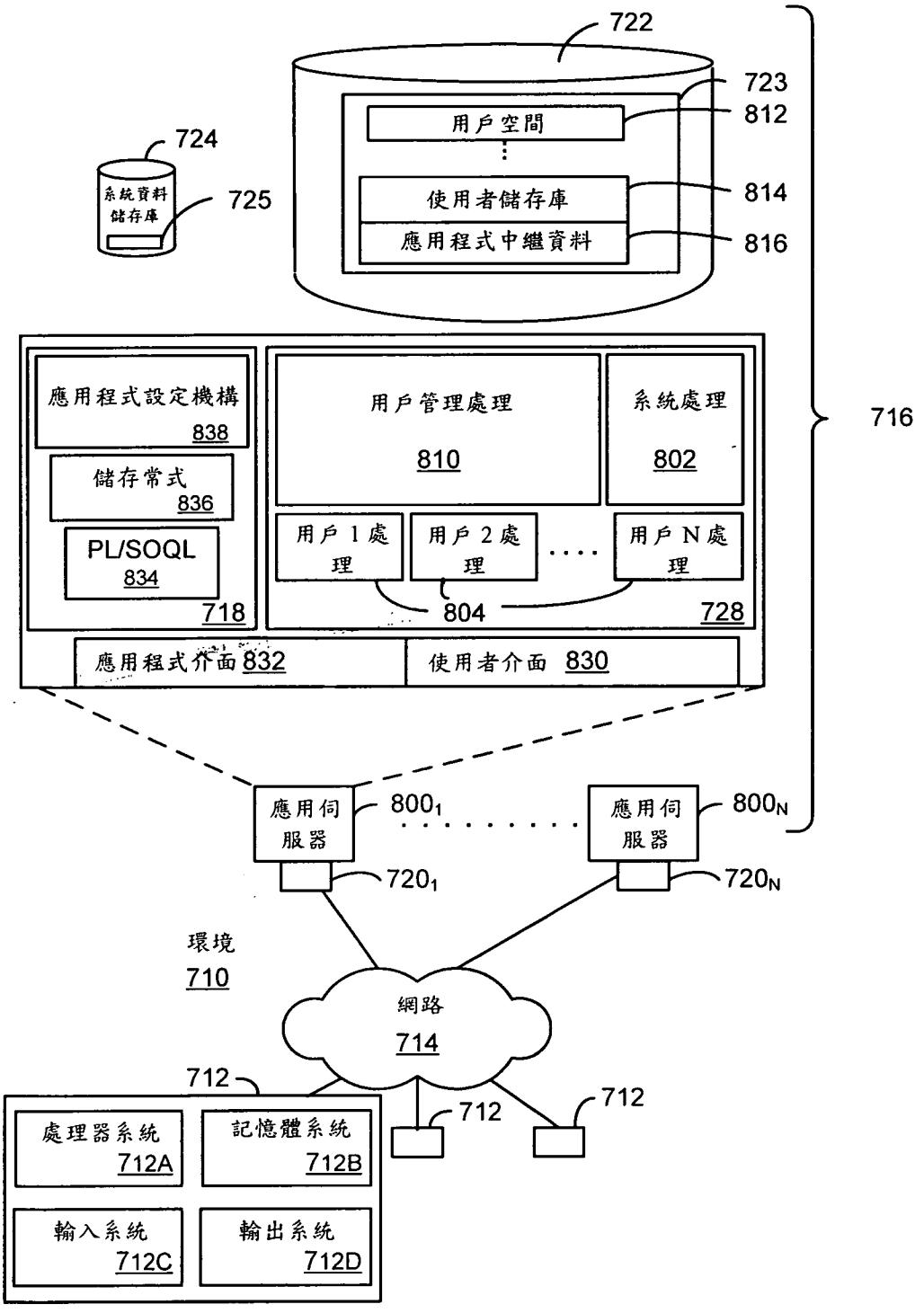

600
在一隨選資料庫服務上接收一應用程式的程式指令
602
判斷要喚起的程式指令版本
604
根據判斷的版本喚起該程式指令的適當部分
606

第六圖

102 年 6 月 27 日修正替換本



第七圖


722
723
用戶空間
812
724
系統資料儲存庫
725
使用者儲存庫
814
應用程式中繼資料
816
應用程式設定機構
838
儲存常式
836
PL/SOQL
834
718
用戶管理處理
810
系統處理
802
用戶 1 處理
用戶 2 處理
用戶 N 處理
804
728
716
應用程式介面 832
使用者介面 830
應用伺服器
800₁
800N
720₁
720N
環境
710
網路
714
712
處理器系統
712A
記憶體系統
712B
輸入系統
712C
輸出系統
712D

第八圖