The Dynamic Business Object Pattern - Paper Reading Present

Dynamic Business Object Pattern

本文提出了一種為**業務應用程序指定動態業務對象**的新模式。該模式的目標是提供一種通用的方法來為業務應用程序設計可擴展的業務對象及其框架。

每個組織或部門都需要特定於其職能領域的應用程序。這些應用程序使用標准的web技术在基於web的基礎設施上實現，并依賴於“業務對象”，如采購訂單、發票和帳戶。這些業務對象在設計級別上有一組公共的屬性和不同的實現解決方案

動態業務對象指的是這些不同的應用程序特定對象的抽象概括，它提供了一個統一的、可重用的規范。動態業務對象模式描述了一種組織和描述業務對象的通用和系統的方法。它在靜態文檔、工作流和動態數據中搆建業務對象。

e.g. 通用會計模塊聚集了支持會計應用程序所涉及的業務對象。典型的會計模塊可以包含動態業務對象，如總帳、項目帳戶、費用帳戶等。該模塊的業務對象規范包括創建、更新和維護數據所需的基本元素，以及執行業務會計任務所需的功能。該會計模塊中的業務對象可以擴展，以適應為單個公司或行業定制的特定屬性和行為。當新模塊從現有模塊擴展而來時，它會繼承父模塊的所有對象和屬性，包括動態數據對象、靜態文檔和工作流。

Intent

動態業務對象模式描述了用於業務應用程序的業務對象的可擴展設計及其平台中立規范的結搆。

Example

抽象的客戶請求流程現在可以擴展為一個反映特定業務流程的更復雜的例子，考慮一個購物車。 從根本上說，它是客戶使用基於web的應用程序對商品或服務的請求。 選擇商品并將其添加到購物車后，客戶提供賬單和運輸信息，然后提交請求進行處理。然后，供應商接收購物車請求，驗證賬單信息，填寫訂單并將其發送給客戶。最后，供應商通過電子郵件通知客戶購物車請求已經完成。

Problem

業務應用程序涉及在客戶端和服務器端編排的復雜分布式操作。應用程序依賴各種具有靜態和動態內容的文檔進行表示和數據存儲。文檔結搆和應用程序行為需要一個平台中立且可重用的規范，該規范可以針對特定的業務案例進一步專門化。

Solution

動態業務對象模式提供了一種規范和設計業務對象的方法。該模式的頂級規范元素是公共模塊(圖1)，由三個主要元素組成，這是支持動態業務對象的描述和行為所需的最少元素:

1. 動態數據對象 (DDO)
2. 靜態文檔

3. 工作流程 幾乎每個業務對象都可以使用公共模塊中引入的三個元素(動態數據對象、靜態文檔和工作流)在模式中呈現。

Dynamic Data Object(DDO)

該對象表示與動態業務對象相關聯的數據，該動態業務對象可能會因某些業務流程或用戶輸入而隨時間發生變化。

動態數據對象的重要區別在於，與靜態文檔不同，它的內容可以(并且經常)被更改，并且與工作流不同，它不是一個過程，而是一個表示特定數據集的對象(或實體)。

e.g. purchase order對象包含一組元素，這些元素表示購買的項目、購買者、購買者以及購買條件。另一方面，采購訂單工作流是一個從提交采購請求到將采購項目交付給買方的過程

公共模塊中的DDO是一個抽象對象類型，假定它在公共模塊中包含一組有限的屬性。這些字段是對象的內部id、創建和更新的時間戳，以及標識對象創建者和更新對象的最后一個用戶ID的字段(如果適用)。 DDO可以包含其他動態數據對象以及靜態文檔。

e.g.購物車(DDO對象)包含一個或多個購物車物品(購物車物品也是DDO對象)。購物車物品可以包括物品的圖像，其中圖像是靜態文檔。

DDO使用一組約束來控制對象上允許的訪問規則和特定行為，以及應該如何完成它們。這些約束必須在可能的情況下使用UML的對象約束語言來指定，或者使用順序邏輯或其他行為UML圖來明確指定。

Static Document

動態業務對象模式中的靜態文檔是其數據不能被應用程序的最終用戶(即客戶)更改的任何文檔。

e.g. 可以是報告打印輸出或產品圖像文件，用戶(客戶端)可以查看這些文件，但客戶端不能修改其內容。

與動態數據對象不同，靜態文檔通常與廣泛接受的客戶端查看器相關聯，并且不需要針對客戶端表示的定制解決方案。

公共模塊中，靜態文檔可分為兩組:印刷媒體和電子媒體。由於實體印刷仍然被企業廣泛使用，它不應該被忽視。

Constraints

約束提供了一種方法來限制對動態數據對象的訪問，并在某種程度上擴展到靜態數據對象。

1. Visibility Constraint(ViC) 可見性約束(ViC)包含規定該對象的組件或屬性的可見性的邏輯。不是每個對象或實體都需要ViC它應該只在應用程序邏輯需要時使用。

e.g. 購物車業務對象的ship-to組件應該總是顯示在購物車UI上，因為如果沒有ship-to，就沒有辦法知道購物車中的商品應該送到哪里

3. Validity Constraint(VaC) 有效性約束(VaC)確定DDO的元素(子對象或屬性)的有效性。默認情況下，并非每個對象或實體都需要VaC。

e.g. 購物車的注釋字段是可選的，無論它是空的還是包含注釋文本都是有效的。

5. Editability Constraint(EC) 可編輯性約束(EC)包含確定DDO的元素(組件或屬性)是否可編輯的邏輯，即是否可通過某些用戶界面進行更改。

e.g. 購物車評論字段應該總是可編輯的，因為它旨在從用戶那里收集附加信息。但是，稅字段或運費字段不可編輯，因為它們是根據項目成本和運輸目的地得出的。

素的可編輯性可以用上下文信息來限定(參數化)，比如用戶角色。

e.g. 倉庫的職員可以核實需要裝運的物品的數量，但是如果需要，只有倉庫經理可以編輯該數量。

Workflow

工作流組件描述了與動態業務對象相關聯的邏輯，即動態業務對象在其生命周期中經曆的操作序列。

工作流的運行時是平台無關的、通用的，并由參與其執行的各種參與者來解釋。

工作流包括動態業務對象的動態和靜態文檔，可能還包括其他業務對象，以及基礎設施提供的服務。

e.g. 當客戶提交想要購買的商品的購物車時，購物車動態業務對象的工作流就開始了。然后，工作流執行驗證客戶付款信息的邏輯，然后將購物車請求轉發到倉庫進行包裝和運輸。一旦項目發貨，工作流將通知客戶完成。

工作流是動態業務對象模式的重要組成部分，因為它的主要職責是跟蹤業務對象在其生命周期中的狀態，并在每個階段采取適當的措施。

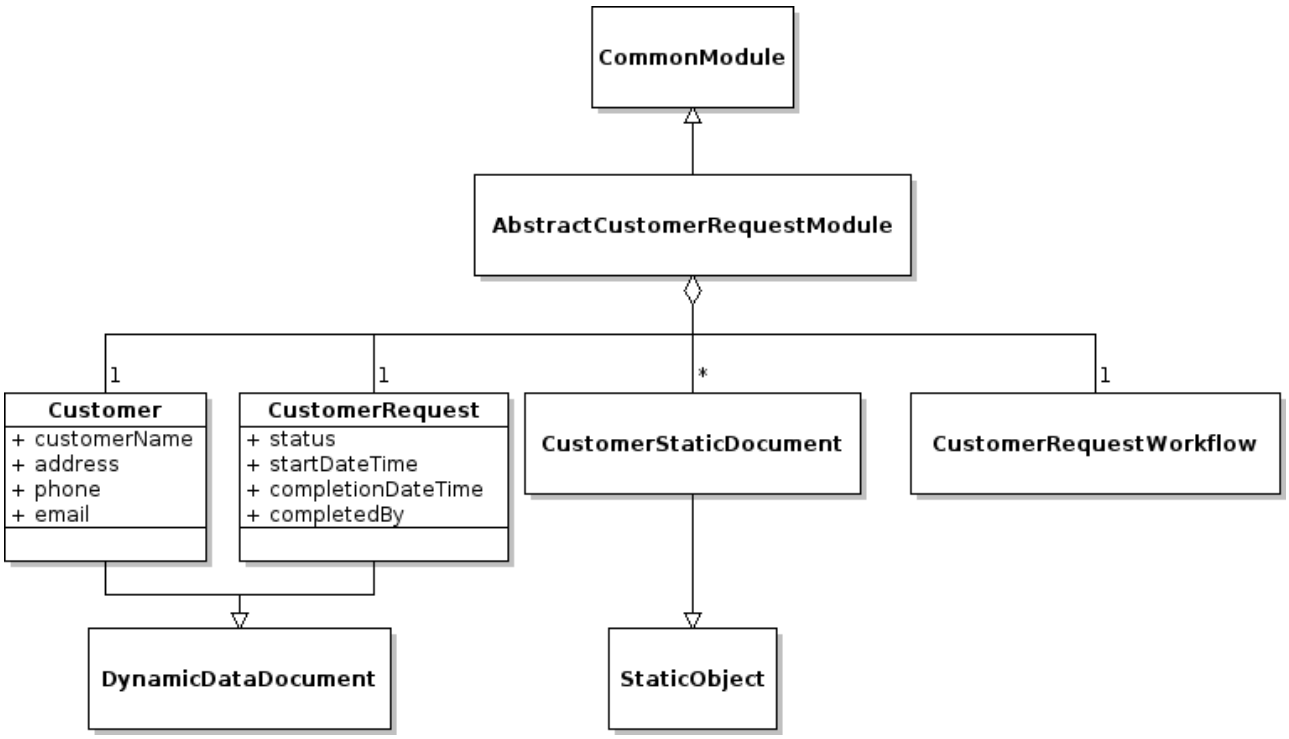
e.g. 當用戶提交待處理的購物車，購物車的工作流流程決定了接下來需要發生什么。工作流可能首先檢查購物者的賬單數據，以確保其有效。然后，If可以檢查倉庫庫存，以確保訂購的商品可用，如果沒有，工作流將向購物者發送一封電子郵件，提供進一步的說明。工作流將業務對象從一個階段“移動”到另一個階段，直到流程完成。

Implementation: a shopping cart

這個例子將展示動態業務對象模式如何利用現有模塊來創建(或擴展)一個新模塊，以滿足特定的業務需求。在本例中，我們將創建模式并將其應用於購物車請求模塊。我們的購物車模塊將基於現有的模塊(抽象客戶請求模塊)。 最基本的客戶要求可以分為以下几個步驟:

1. 客戶發起請求
2. 業務部門接收并處理請求

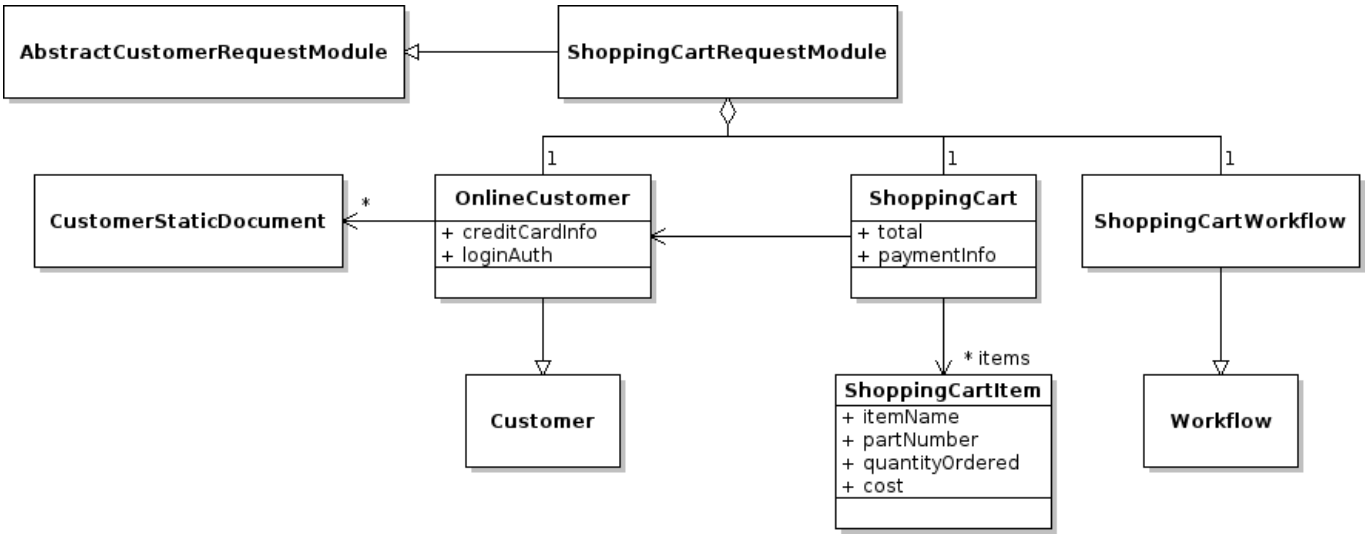
3. 通知客戶請求已完成



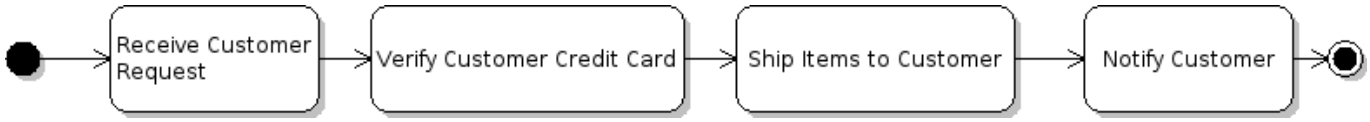
客戶和客戶請求對象是公共模塊中動態數據對象的擴展。客戶靜態文檔和客戶請求工作流分別是靜態文檔和工作流的擴展，也可以在通用模塊中找到。

客戶靜態文檔對象沒有任何特定的變化，因為它屬於抽象客戶請求模塊——該對象只是從通用模塊繼承而來。ACR模塊中的工作流經過修改，以反映處理一般客戶請求所需的最低要求

下一步是根據動態業務對象模式對其進行擴展，以分析和構建購物車請求模塊。 在其最基本的形式中，購物車是客戶對商品或服務的請求。客戶瀏覽目錄，從中選擇商品，最后提交進行處理。因此，依賴ACR模塊來構建購物車請求模塊(SCR)是有意義的。ACR模塊針對新的購物車請求模塊進行了擴展



擴展以反映購物車請求的性質:它是由一個在線客戶(因此是新的登錄和信用卡信息)和購物車及其商品的新類發起的。 擴展了客戶請求工作流，以滿足處理購物車請求的新需求



通過擴展抽象客戶請求模塊，只有處理客戶請求操作被兩個新的操作取代:驗證客戶信用卡和向客戶發送商品。

Consequence

Benefit

動態業務對象模式可以極大地簡化業務對象的開發，特別是當動態業務對象庫隨着時間的推移而不斷增長和使用時。該模式為描述給定的業務對象及其內容提供了一種清晰的方法，即使存在高度的復雜性。該模式還考慮了支持業務對象生命周 期所需的流程(通過工作流)。如果需要的話，該模式還足夠靈活，允許將來進行更改和添加。

Liabilities

動態業務對象模式不考慮用戶界面和數據持久性級別的動態數據對象支持的細節。此外，在積累動態業務對象庫之前，許多工作都需要從頭開始。