**Loyalty Manage System**

Transaction

|  |  |
| --- | --- |
| Date | 03/06/2010 |
| Version | 1.0 |
| Status | Inital |
| Author | Hung Pham |
| Review By | Khuyen Nguyen |
| Approved By | Dat Tran |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 03/06/2010 | 1.0 | Initial | Hung Pham |

Contents

[1 Transaction là gì? 3](#_Toc255727425)

[1.1 Atomic 3](#_Toc255727426)

[1.2 Consistency 3](#_Toc255727427)

[1.3 Isolation 3](#_Toc255727428)

[1.4 Durable 4](#_Toc255727429)

[2 Transaction trong JPos 1.6.4 4](#_Toc255727430)

[2.1 Participant 4](#_Toc255727431)

[2.2 Pha 5](#_Toc255727432)

[2.3 Transaction Context: 6](#_Toc255727433)

# Transaction là gì?

Chúng ta từng nghe nhắc đến rất nhiều lần về transaction mà đặc biệt là transaction trong database.Vì vậy bài viết chỉ nêu lên những nét tổng quát và chủ yếu nhất của một transaction đồng thời đi sâu vào việc quản lý transaction trong Jpos 1.6.4 .

Transaction là một chuỗi các hành động được thực thi trong đó mỗi hành động phải **nguyên tử**.Đưa ra quyết định về sự thành công cũng như thất bại của tất cả các hành động.

Việc đưa ra khái niệm dựa trên hiểu biết của tác giả về transaction.

Để một chuỗi các hành động được gọi là một transaction thì chuỗi hành động đó phải đảm bảo 4 tính chất sau :

+ Atomic

+ Consistency

+ Isolation

+Durable

## Atomic

Một hành động được gọi là nguyên tử nghĩa là nó không thể phân chia ra những hành động con bên trong nó nữa.

Ví dụ : Xét ví dụ về hai hành động sau : Tính điểm trung bình của một sinh viên và tính điểm trung bình của một khóa học trong trường đại học.Chúng ta xem xét hành động thứ hai : tính điểm trung bình của một khóa học mà điểm trung bình của một khóa học thì nó bao gồm điểm trung bình của các học sinh. Do đó chúng ta nói rằng hành động thứ hai không đạt tiêu chí atomic. Còn hành động thứ nhất thì đáp ứng được tiêu chí này. Việc xét các tiêu chí dựa trên ngữ cảnh của bài toán.

## Consistency

Giữa các chuỗi hành động đôi lúc cần trao đổi dữ liệu qua lại cho nhau. Tính duy trì của thể hiện ở chỗ các dữ liệu trao đổi qua lại giữ các hành động phải được di trì một cách chính xác về giá trị.

## Isolation

Sự độc lập giữa các transaction thể hiện ở chỗ các tài nguyên (cả tĩnh và động) **chỉ được** truy xuất đúng trong transaction của nó.

## Durable

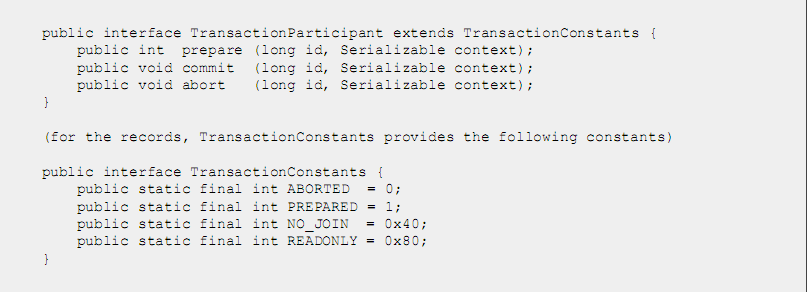
Thuộc tính này chỉ ra rằng khi một sự kiện xảy ra trên một hành động của transaction thì transaction manager sẽ đưa ra quyết định cho hành động đó hoặc thậm chí **đưa ra quyết định** cho toàn bộ các hành động của transaction hoặc các hành động có liên quan.

# Transaction trong JPos 1.6.4

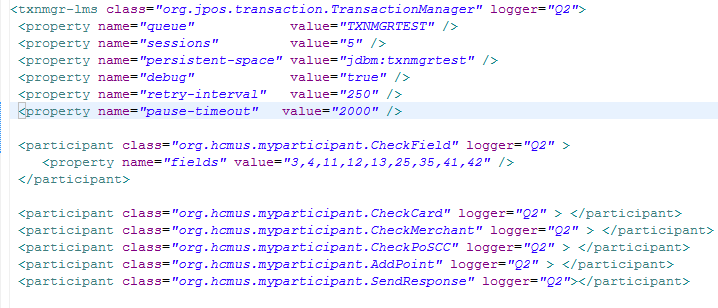
## Participant

Transaction trong Jpos 1.6.4 mang đầy đủ các tính chất của một transaction.

Các hành động atomic sẽ được được gọi là participant. Jpos cung cấp các interface để chúng ta thực thi các hành động này .



Để khai báo một transaction trong Q2 đơn giản là tạo một file xml để cấu hình cho transaction đó trong sau đó lưu lại trong thư mục deploy.



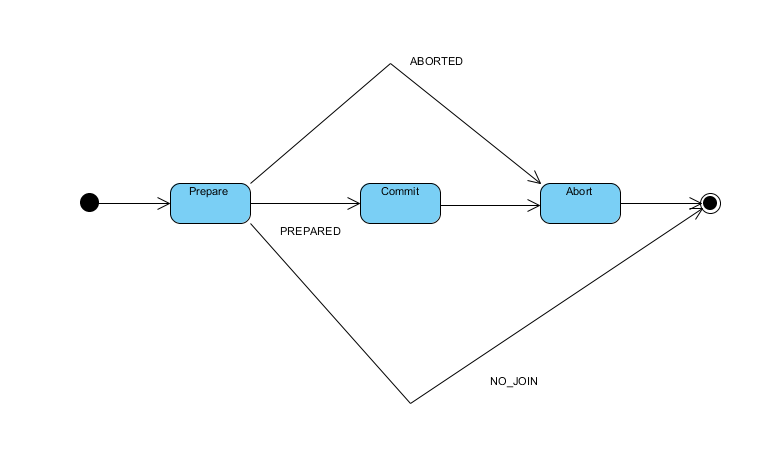
Các participant được cấu hình sẽ chỉ đến các lớp thực thi interface TransactionParticipant.

## Pha

Trước khi bàn về việc quản lý và đưa ra quyết định cho các participant của jpos . Chúng ta xem xét các pha mà một participant tham gia vào. Jpos chia ra làm 3 pha chính Prepare, Commit và Abort.

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| ABORTED | 0 | Participant trong pha Prepare. Participant sẽ chuyển qua pha Abort. |
| PREPARED | 1 | Participant đã được prepare và và kiểm tra tính prepared cho các paricipant khác. |
| NO\_JOIN | 0x40 | Báo cho bộ quản lý transaction biết rằng participant này không tham gia vào pha commit cũng như abort. |
| READ\_ONLY | 0x80 | Báo cho bộ quản lý transaction biết rằng participant có không sửa đổi các dữ liệu được duy trì vì thế việc lưu lại một bản sao của nó là không cần thiết. |

Xem xét state chart sau :



TransactionManager sẽ thực hiện nhiệm vụ điều khiển các Transaction bằng cách gọi phương thức prepare của tất cả các Participant.Nếu tất cả chúng đều return PREPARED thì transaction sẽ chuyển đến pha commit và tại đây phương thức commit của tất cả các participant sẽ được gọi.

Nếu một trong các participant return ABORTED thì transaction sẽ chuyển đến pha abort. Và tại đây phương thức abort của tất cả các participant sẽ được gọi.

Mở rộng vấn đề :

Khi một participant return NO\_JOIN để báo cho bộ quản lý transaction biết rằng participant đó không muốn tham gia vào quá trình transaction nữa. Tuy nhiên trong các trường hợp recovery thì transaction vẫn phải gọi phương thức commit và abort của nó. Vì vậy developer phải chuẩn bị để nhận phương thức commit hay abort của một (unknow transaction).

Source code của bạn sẽ kiểm tra long id hoặc Seriablizable context sau đó làm cái tác vụ mà bạn muốn.

## Transaction Context:

Để đảm bảo cho tính chất Isolation và Consitency. Transaction dùng đối tượng Context để quản lý dữ liệu dùng chung cho các participant cũng như duy trì dữ liệu đó xuyên qua các participant.

Trong context sử dụng 2 đối tượng **map : persitent map và trancient map** để quản lý và duy trì các đối tượng dùng chung .

Trancient map đưa cho bạn khả năng để đặt các đối tượng động vào trong context như socket connection, jdbc connection, thread, manage object.

