Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Đại Học Khoa Học Tự Nhiên Thành Phố Hồ Chí Minh

Khoa Công Nghệ Thông Tin

Bộ Môn Công Nghệ Phần Mềm

Phạm Thế Hùng – Nguyễn Khuyến

**Phần Mềm Quản Lý Khách Hàng Thân Thiết**

Khóa Luận Cử Nhân Tin Học

Thành Phố Hồ Chí Minh năm 2010

Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Đại Học Khoa Học Tự Nhiên Thành Phố Hồ Chí Minh

Khoa Công Nghệ Thông Tin

Bộ Môn Công Nghệ Phần Mềm

Phạm Thế Hùng - 0612177

Nguyễn Khuyến - 0612193

**Phần Mềm Quản Lý Khách Hàng Thân Thiết**

Khóa Luận Cử Nhân Tin Học

Giảng Viên Hướng Dẫn :

Thầy Trần Hiển Đạt

Niên Khóa 2006 – 2010

**Nhận xét của giảng viên hướng dẫn**

**Nhận xét của giảng viên phản biện**

**Lời Cảm Ơn**

✍🕮🖎

Lời đầu tiên chúng em xin cảm ơn Khoa Công Nghệ Thông Tin Trường Đại Học Khoa Học Tự Nhiên đã tạo cơ hội cho chúng em thực hiện khóa luận.

Chúng em xin gởi lời cảm ơn chân thành nhất tới Thầy Trần Hiển Đạt, Thầy đã hướng dẫn rất tận tình cho chúng em trong suốt khóa luận vừa qua.

Bên cạnh đó chúng em xin gởi lời cảm ơn đến tất cả quý Thầy Cô của Khoa đã tận tình giảng dạy và trang bị cho chúng em kiến thức vững chắc góp phần rất lớn vào việc nghiên cứu thành công khóa luận này.

Cuối cùng,chúng em gởi lời cảm ơn sâu sắc đến cha mẹ và gia đình đã nuôi nấng dạy dỗ chúng em đến ngày hôm nay cũng như động viên giúp đỡ chúng em trong suốt quá trình thực hiện khóa luận.

Dù đã cố gắng hết sức để thực hiện khóa luận tuy nhiên những thiếu sót là không thể tránh khỏi vì vậy mong nhận được những lời chỉ bảo và góp ý của Thầy Cô cũng như những chia sẽ của tất cả các bạn.

Thành Phố Hồ Chí Minh, tháng 7 năm 2010

Nhóm sinh viên thực hiện

Phạm Thế Hùng – Nguyễn Khuyến

Mục Lục

[Chương 1 : Tổng Quan 12](#_Toc262767517)

[1.1 Yêu cầu thực tế và lý do thực hiện đề tài : 12](#_Toc262767518)

[1.2 Hiện trạng của đơn vị xây dựng phần mềm : 12](#_Toc262767519)

[1.3 Mục tiêu khóa luận 13](#_Toc262767520)

[1.4 Hướng tiếp cận của khóa luận : 13](#_Toc262767521)

[1.5 Các ưu điểm và khuyết điểm của hệ thống : 13](#_Toc262767522)

[1.6 Nội dung thực hiện của đề tài : 14](#_Toc262767523)

[Chương 2 : Cơ Sở Lý Thuyết 15](#_Toc262767524)

[2.1 ISO 8583 15](#_Toc262767525)

[2.1.1 ISO 8583 là gì? 15](#_Toc262767526)

[2.1.2 Cấu trúc thông điệp 15](#_Toc262767527)

[2.1.3 Chỉ thị thông điệp (Message Type Indicator – MTI) 15](#_Toc262767528)

[2.1.3.1 Phiên bản ISO8583 15](#_Toc262767529)

[2.1.3.2 Lớp thông điệp : 16](#_Toc262767530)

[2.1.3.3 Chức năng của thông điệp 16](#_Toc262767531)

[2.1.3.4 Gốc thông điệp (Message Origin) 17](#_Toc262767532)

[2.1.4 Bitmap 18](#_Toc262767533)

[2.1.5 Thành phần dữ liệu (Data Element) 21](#_Toc262767534)

[2.2 JPOS 22](#_Toc262767535)

[2.2.1 JPOS cách thức xử lý ISO8583 22](#_Toc262767536)

[2.2.2 Transaction trong JPOS 25](#_Toc262767537)

[2.2.2.1 Transaction là gì? 25](#_Toc262767538)

[2.2.2.2 Các tính chất của Transaction 26](#_Toc262767539)

[2.2.2.3 Transaction trong JPos 1.6.4 27](#_Toc262767540)

[2.3 Gateway 30](#_Toc262767541)

[2.3.1 Wire Protocol 30](#_Toc262767542)

[2.3.2 Packing và Unpacking 31](#_Toc262767543)

[2.3.3 Quản lý Wire Protocol với ISOChannel 33](#_Toc262767544)

[Chương 3 : Thiết Kế Chương Trình 36](#_Toc262767545)

[3.1 Server 36](#_Toc262767546)

[3.1.1 Thiết kế kiến trúc của chương trình : 36](#_Toc262767547)

[3.1.1.1 Thiết kế tổng quát 36](#_Toc262767548)

[3.1.1.2 Chi tiết của kiến trúc 37](#_Toc262767549)

[3.1.2 Mô hình hóa nghiệp vụ 41](#_Toc262767550)

[3.1.2.1 Actors 42](#_Toc262767551)

[3.1.2.2 User Cases 42](#_Toc262767552)

[3.1.3 Đặc tả Usercase : 42](#_Toc262767553)

[3.1.3.1 Đặc tả usercase Redeem 42](#_Toc262767554)

[3.1.3.2 Đặc tả usercase Reload 44](#_Toc262767555)

[3.1.3.3 Đặc tả usercase Void 46](#_Toc262767556)

[3.1.3.4 Đặc tả usercase Balance Inquiry 48](#_Toc262767557)

[3.1.4 Thiết kế thông điệp cho các giao dịch 51](#_Toc262767558)

[3.1.4.1 Redeem transaction 51](#_Toc262767559)

[3.1.4.2 Reload Transaction 56](#_Toc262767560)

[3.1.4.3 Balance Inquiry Transaction 59](#_Toc262767561)

[3.1.4.4 Void Transaction 62](#_Toc262767562)

[3.1.4.5 Đặc tả cho các trường thông điệp chung : 66](#_Toc262767563)

[3.1.5 Thực thi chương trình 67](#_Toc262767564)

[3.2 GateWay (Multi Channel) 69](#_Toc262767565)

[3.3 Database 73](#_Toc262767566)

[3.3.1 Sơ đồ logic 73](#_Toc262767567)

[3.3.2 Mô tả chi tiết các bảng dữ liệu trong sơ đồ logic 74](#_Toc262767568)

[3.3.2.1 JPOS\_Admin 74](#_Toc262767569)

[3.3.2.2 JPOS\_Card 74](#_Toc262767570)

[3.3.2.3 JPOS\_Customer 74](#_Toc262767571)

[3.3.2.4 JPOS\_Event 75](#_Toc262767572)

[3.3.2.5 JPOS\_Gift 76](#_Toc262767573)

[3.3.2.6 JPOS\_Issuer 76](#_Toc262767574)

[3.3.2.7 JPOS\_LOG 76](#_Toc262767575)

[3.3.2.8 JPOS\_Log\_Event 77](#_Toc262767576)

[3.3.2.9 JPOS\_Log\_Exchange 77](#_Toc262767577)

[3.3.2.10 JPOS\_Merchant 78](#_Toc262767578)

[3.3.2.11 JPOS\_PoSCC 78](#_Toc262767579)

[3.3.2.12 JPOS\_Rule 78](#_Toc262767580)

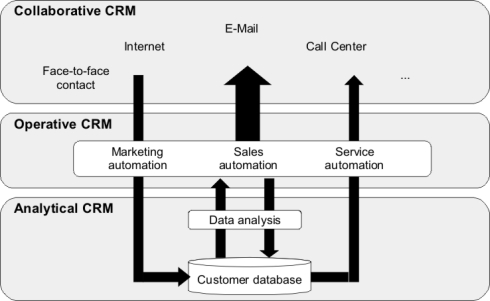
[3.3.2.13 JPOS\_Status 78](#_Toc262767581)

[3.3.2.14 JPOS\_Task 79](#_Toc262767582)

[3.3.2.15 JPOS\_Terminal 79](#_Toc262767583)

Lời Mở Đầu

Ý thức được tầm quan trọng của các hoạt động hướng khách hàng (KH), ngày càng nhiều doanh nghiệp quan tâm hơn đến các ứng dụng quản trị quan hệ khách hàng (CRM). Để đạt được giá trị lâu dài của giải pháp CRM, yêu cầu kiến trúc chiến lược phải gồm toàn bộ các hoạt động kinh doanh, dịch vụ liên quan đến khách hàng của doanh nghiệp và trên mức độ toàn doanh nghiệp.



Hình 1‑1

Hình trên biểu diễn một kiến trúc CRM được cho là thành công hiện nay.

Tại các doanh nghiệp Việt Nam, chiến lược CRM đầy đủ gặp rất nhiều khó khăn. Việc đánh giá vị trí hiện tại của doanh nghiệp liên quan đến giá trị, lòng trung thành và độ hài lòng của khách chưa đầy đủ vì thiếu thông tin khách hàng. Giá trị khách hàng phần lớn được đánh giá phiến diện theo doanh số trong khi lòng trung thành và độ hài lòng của khách hàng thì gần như không đánh giá được. Việc thiết lập các mục tiêu về khách hàng phần lớn mang tính mơ hồ, không rõ ràng. Đa phần doanh nghiệp đều không quan tâm đến việc chỉ ra các yêu cầu về con người: kỹ năng, văn hóa, tổ chức, trách nhiệm, quyền hạn…Các yêu cầu về dữ liệu khách hàng thì hoàn toàn thiếu và không thống nhất, dựa trên dữ liệu rời rạc hiện có của các phòng ban riêng rẽ. Ví dụ, một doanh nghiệp phân phối hàng tiêu dùng thì dữ liệu khách hàng đại lý gần như chỉ có dữ liệu cơ bản (tên khách hàng, địa chỉ, doanh số, dữ liệu kế toán). Trong khi CRM cần nhiều hơn rất nhiều các thông tin này để xây dựng lại mối quan hệ kinh doanh với các khách hàng đã mất hoặc xây dựng các kế hoạch marketing hay chính sách bán hàng, dịch vụ...

Vì những nhược điểm được chỉ ra ở trên vấn đề xây dựng một kiến trúc CRM là thiết thực và lâu dài cho doanh nghiệp. Trong khuôn khổ của khóa luận chúng ta không quan tâm tất cả các giải pháp xây dựng kiến trúc CRM mà thay vào đó chúng ta quan tâm đến một nhánh nhỏ trong CRM là khách hàng thân thiết. Và việc xây dựng hệ thống quản lý khách hàng thân thiết(LMS) trở thành vấn đề đang được doanh nghiệp quan tâm hiện nay.

Sau cái nhìn tổng quát về CRM và thấy được vị trí của LMS trong mô hình CRM. Chúng ta đề ra một số giải pháp cho hệ thống LMS. Các giải pháp của LMS cũng không lạc khỏi mục tiêu đã đề cập ở trên là làm cách nào để giữ các khách hàng tốt nhất của bạn.Một số câu hỏi được đặt ra và bằng việc trả lời các câu hỏi này chúng ta có cái nhìn sâu hơn về LMS.

* Tại sao khách hàng rời bỏ chúng ta:

-Có phải do chất lượng sản phẩm của bạn chưa tốt?

-Thông tin truyền thông phát triển khiến cho khách hàng có nhiều thông tin hơn về sản phẩm và sự cạnh tranh của thị trường ngày càng khóc liệt?

-Chế độ quan tâm của bạn đối với khách hàng thân thiết chưa tốt?

* Đánh giá lợi ích khi giữ được khách hàng thân thiết :

-Khách hàng thân thiết đồng nghĩa với cơ hội bán hàng.

-Lợi ích được thấy rõ dựa trên số sản phẩm khách hàng thân thiết tiêu thụ.

-Lợi nhuận cho mỗi lần mua bán đối với một khách hàng thân thiết.

* Khi bạn xây dựng một hệ thống khách hàng thân thiết bạn sẽ được những lợi ích gì ?

-Làm cho khách hàng ngày càng thân thiết hơn.

-Nắm được giá trị của của mỗi khách hàng thân thiết.

-Sự giới thiệu của khách hàng thân thiết cho các khách hàng mới.

-Làm cho họ quay trở lại với sự tiêu tụ cao hơn lần trước.

-Hiểu rõ khách hàng hơn nữa và kích thích nhu cầu của họ.

-Kich thích khách hàng mà không cần giảm giá.

-Theo dõi và đo lường việc mua sắm của khách hàng.

-Tăng khả năng chiến đấu với các sản phẩm khác.

Từ những điều đã được đề cập phía trên khóa luận tập trung vào xây dựng một hệ thống quản lý khách hàng thân thiết thông qua thẻ khách hàng thân thiết do doanh nghiệp phát hành.

Khóa luận này gồm 4 chương có nội dung như sau :

**Chương 1** : Giới thiệu bối cảnh, lý do thực hiện đề tài cũng như các giải pháp hiện thời có liên quan từ đó rút ra hướng tiếp cận của khóa luận để thực hiện đề tài.

**Chương 2** : Trình bày sơ lược các lý thuyết về chuẩn ISO8583, lý thuyết về JPOS Framework các chi tiết kỹ thuật liên quan đến đề án.

**Chương 3** : Phân tích và thực hiện hệ thống quản lý khách hàng thân thiết dựa trên lý thuyết đã đề ra ở trên.

**Chương 4** : Thực nghiệm,đánh giá và tổng kết : nêu đánh giá về toàn bộ đề tài , trình bày các kết quả đã đạt được cũng như hạn chế của đề tài từ đó đề xuất ra những hướng nghiên cứu trong tương lai.

# : Tổng Quan

## Yêu cầu thực tế và lý do thực hiện đề tài :

Trong vài năm tới Việt Nam của chúng ta sẽ có chính phủ điện tử, chính phủ điện tử ra đời sẽ giúp hoàn chỉnh hệ thống thương mại điện tử của Việt Nam vì vậy thương mại điện tử trở thành mảnh đất màu mở và giàu tiềm năng hiện nay. Cùng với xu hướng đó quản lý mối quan hệ khách hàng cụ thể là quản lý khách hàng thân thiết được các doanh nghiệp đặc biệt quan tâm . Hệ thống quản lý này có thể áp dụng từ những nhà sách nhỏ muốn giữ chân bạn đọc của mình cho đến các hệ thống siêu thị lớn kích thích khả năng mua sắm của người tiêu dủng.

Các phần mềm quản lý khách hàng thân thiết trên thị trường bắt đầu đã xuất hiện trong vài năm trở lại đây điển hình là hệ thống quản lý khách hàng thông qua thẻ khách hàng thân thiết ở các siêu thị như BigC và Nguyễn Kim tuy nhiên các phần mềm này phần lớn là mã nguồn đóng và được thực hiện cụ thể nhầm vào một đối tượng doanh nghiệp nhất định vì thế chi phí thực hiện là rất lớn và khả năng tái sử dụng thấp.Vì thế việc xây dựng một hệ thống quản lý khách hàng thân thiết mã nguồn mở trở thành nhu cầu thực tế được quan tâm và khả năng phát triển dựa trên sự đóng góp của cộng đồng cho các phần mềm mã nguồn mở rất hứa hẹn.

## Hiện trạng của đơn vị xây dựng phần mềm :

Phần mềm được xây dựng trong phạm vi của một khóa luận ở trường đại học.Các gói chương trình được nhóm sinh viên thực hiện dưới sự hướng dẫn của giảng viên.Các trang thiết bị dùng để thực hiện và kiểm thử dưới sự giúp đỡ của một số công ty chuyên về thương mại điện tử ,nhà trường tạo mọi điều kiện tốt nhất cho sinh viên khảo sát thực tế cũng như phát triển dự án.

## Mục tiêu khóa luận

Khóa luận gồm 3 mục tiêu chính :

* Thứ nhất tìm hiểu về chuẩn ISO8583 cách thức gởi và nhận thông điệp ISO8583.
* Thứ hai : tìm hiểu về JPOS framwork.
* Thứ ba : Sử dụng JPOS framework gởi và nhận các gói tin ISO8583 từ đó ứng dụng xây dựng nên hệ thống quản lý khách hàng thân thiết dựa vào thẻ khách hàng thân thiết do doanh nghiệp phát hành.

## Hướng tiếp cận của khóa luận :

Khóa luận tập trung vào xây dựng giải pháp server cho hệ thống LMS. Hệ thống server sẽ nhận các gói dữ liệu gởi theo chuẩn ISO8583 từ các thiết bị đọc mã vạch hoặc các máy POS terminal hay thậm chí là từ thiết bị di động có hỗ trợ phần mềm đọc mã vạch. Sau đó hệ thống sẽ tiến hành thực hiện các tác vụ có liên quan như cộng điểm khi mua hàng trừ điểm khi trả hàng hoặc các tác vụ khác như kích hoạt thẻ sau đó lưu trữ vào database và trả kết quả về cho client.

Hệ thống hỗ trợ khách hàng quản lý truy vấn thông tin tài khoản của mình thông qua giao diện web. Quản lý thống kê các tài khoản khách hàng, thống kê việc mua bán của khách hàng nhầm giúp các doanh nghiệp đưa ra chuyến lược kinh doanh hợp lý.

## Các ưu điểm và khuyết điểm của hệ thống :

**Ưu điểm** : Hỗ trợ các thiết bị có sẵn như điện thoại di động có khả năng đọc barcode.Các kênh truyền được thiết kế độc lập nhờ một module trung gian giúp cho việc khả chuyển các kênh truyền phục vụ cho việc mở rộng bài toàn sau này.

**Khuyết điểm** : Khả năng bảo mật thấp do mã nguồn mở.Việc ứng dụng vào thực tế cần có sự đầu tư của các doanh nghiệp.

## Nội dung thực hiện của đề tài :

Đề tài được thực hiện theo từng bước sau :

* Tìm hiểu về hệ thống và thực trạng của các giải pháp quản lý khách hàng thân thiết hiện nay.
* Phân tích và định hình bài toán quản lý khách hàng thân thiết đề ra mục tiêu và nhu cầu của người dùng cũng như các chức năng mà hệ thống hỗ trợ.
* Tìm hiểu về chuẩn ISO8583 và các phần mềm hỗ trợ gởi nhận thông điệp ISO8583 trên nền tảng cả Java và .Net.
* Tìm hiểu về JPOS framework và cách sử dụng JPOS framework để gởi nhận thông điệp theo chuẩn ISO8583
* Tìm hiểu về việc quản lý transaction sử dụng JPOS framework.
* Tổ chức xây dựng cơ sở dữ liệu cho bài toán.
* Phân tích và thiết kế tài liệu cũng như mô hình cho bài toán thông qua ngôn ngữ UML cũng như đề ra kiến trúc cho hệ thống và quy trình phát triển hệ thống.
* Tiến hành thực thi dự án.
* Kiểm thử và sửa lỗi
* Đóng gói
* Viết tài liệu về dự án.

Vừa rồi chúng ta lướt qua tổng quan của dự án. Chương tiếp theo chúng ta sẽ bàn sâu hơn về các chi tiết kỹ thuật cũng như cơ sở lý thuyết của dự án.

# : Cơ Sở Lý Thuyết

## 

## ISO 8583

### ISO 8583 là gì?

ISO8583 là một tiêu chuẩn cho các thẻ giao dịch tài chính bằng cách trao đồi các thông điệp được chỉ định.

### Cấu trúc thông điệp

Một thông điệp ISO8583 được chia làm các phần sau:

-Chỉ thị thông điệp (Message Type Indicator)

-Một hay nhiều bitmap chỉ tới các thành phần dữ liệu trong thông điệp.

-Thành phần dữ liệu hay trường các dữ liệu.

### Chỉ thị thông điệp (Message Type Indicator – MTI)

Một MTI chứa các thành phần sau :

-Phiên bản ISO8583

-Lớp thông điệp

-Chức năng thông điệp

-Gốc của thông điệp

#### Phiên bản ISO8583

Vị trí đầu tiên của MTI chỉ định phiên bản cho thông điệp của chuẩn ISO8583.

Cụ thề sau:

|  |  |
| --- | --- |
| Vị trí | Ý nghĩa |
| 0xxx | Phiên bản ISO 8583-1:1987 |
| 1xxx | Phiên bản ISO 8583-2:1993 |
| 2xxx | Phiên bản ISO8583-1:2003 |
| 9xxx | Sử dụng cá nhân. |

#### Lớp thông điệp :

Vị trí thứ 2 của MTI chỉ ra mục đích của của thông điệp.Sau đây là một số vị trí của thông điệp thường sử dụng.

|  |  |
| --- | --- |
| Ví trí | Ý nghĩa |
| x1xx | Thông điệp xác nhận |
| x2xx | Thông điệp tài chính |
| x4xx | Thông điệp đảo ngược hành động trước đó. |
| x7xx | Thông điệp chuyển phí |
| x8xx | Thông điệp quản lý đường truyền |
| x9xx | Dự trữ bởi ISO |

Chúng ta xem xét ví dụ sau :

x**4**xx chỉ ra mục đích sử dụng cho thông điệp này.Tình huống khi một client gởi thông điệp (có MTI x2xx) để xác nhận mã thẻ đồng thời rút một số tiền từ tài khoản của mình.Tuy nhiên khi giao dịch vẫn chưa kết thúc client mong muốn hủy quá trình giao dịch rút tiền đang diễn ra client tiếp tục gởi thông điệp có MTI x4xx. Thì vị trí thứ 2 có giá trị 4 trong x**4**xx chỉ ra rằng đây là một thông điệp xin đảo ngược lại giao dịch rút tiền trước đó.

#### Chức năng của thông điệp

Vị trí thứ 3 của MTI chỉ ra chức năng của thông điệp định nghĩa cách mà thông điệp sẽ chạy trong hệ thống.

|  |  |
| --- | --- |
| Vị trí | Ý nghĩa |
| xx0x | Request |
| xx1x | Request Response |
| xx2x | Advice |
| xx3x | Advice Response |
| xx4x | Notification |

-Request : Xem xét xx**0**x . Request là một chức năng từ cuối tới cuối (end to end) .Ví dụ từ nhà chấp nhận thẻ(các ngân hàng) cho tới các nhà phát hành thẻ(các nhà phát hành thẻ Vista,MasterCard).

-Advice : Trong khi đó xx2x Advice là một chức năng điểm tới điểm(point to point). Ví dụ từ terminal(POS devide hay máy ATM chẳng hạn) tới nhà chấp nhận thẻ, từ nhà chấp nhận thẻ ra đường mạng từ đường mạng tới nhà phát hành thẻ việc chuyển đổi được bảo đảm trên mỗi liên kết.

-Request response và advice response : response chỉ ra rằng đây là thông điệp gởi trả cho một thông điệp request (hoặc advice) trước đó.

#### Gốc thông điệp (Message Origin)

Vị trí thứ tư trong MTI chỉ ra nguồn của thông điệp trong chuỗi chi trả.

|  |  |
| --- | --- |
| Vị trí | Ý nghĩa |
| xxx0 | Acquirer |
| xxx2 | Issuer |
| xxx4 | Other |

-Acquirer : Là các tổ chức chấp nhận thẻ. Ví dụ ngân hàng VietComBank có thể chấp nhận các thẻ do chính ngân hàng VietCombank phát hành và các thẻ khác như Vista hay MasterCard. VietComBank trở thành một Acquirer.

-Issuer : Là tổ chức phát hành ra các thẻ giao dịch trên thị trưởng.

* Ví dụ về MTI

Sự kết hợp các vị trí của một MTI chỉ ra ý nghĩa đầy đủ của MTI.

Xét MTI 0200 : Trường thứ nhất trong 0200 mang giá trị 0 chỉ ra thông điệp này sử dụng phiên bản ISO 8583-1: 1987. Trường thứ 2 trong 0200 mang giá trị 2 chỉ ra đây là thông điệp tài chính . Trường thứ 3 trong 0200 mang giá trị 0 chỉ ra rằng đây là một thông điệp Request và trường thứ 4 có giá trị 0 chỉ ra rằng thông điệp được gởi từ Acquirer.

### Bitmap

Bitmap là một trường hay là một trường con của thông điệp chỉ ra các thành phần dữ liệu được trình bày ở những nơi khác trong thông điệp.

Một thông điệp ít nhất có một bitmap được gọi là Primary Bitmap (Bitmap chính) chỉ định các thành phần dữ liệu từ số 1 tới số 64. Một seconary bitmap(Bitmap thứ hai) chỉ định các trường dữ liệu từ 65 cho đến 128 và cũng có thể có bitmap thứ 3 chỉ định các trường dữ liệu từ 129 cho tới 192(Bitmap thứ 3 này rất ít sử dụng).

Bitmap có thể được truyền bằng 8 byte nhị phân hoặc 16 kí tự thập lục phân trong bảng mã ASCII hoặc tập kí tự [[1]](#footnote-2)EBCDIC.

Một trường được chỉ định khi mà bit đại diện cho trường đó bật là true trong bitmap.Ví dụ byte x82 chuyển qua giá trị nhị phân “1000 0010” chỉ ra rằng thông điệp chứa trường số 1 và số 7.

Xem xét Primary Bitmap sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Byte | Hex Value | Bit Value | Field # |
| 0 | 20 | 0010 0000 | 3 |
| 1 | 20 | 0010 0000 | 11 |
| 2 | 00 | 0000 0000 |  |
| 3 | 00 | 0000 0000 |  |
| 4 | 00 | 0000 0000 |  |
| 5 | 80 | 1000 0000 | 41 |
| 6 | 00 | 0000 0000 |  |
| 7 | 00 | 0000 0000 |  |

Primary bitmap (2020000000800000) chứa các bit 1 bật ở các vị trí 3,11,41 cho chúng ta biết rằng thông điệp chứa các trường 3,11,41.

Xem xét thông điệp có bitmap và các trường như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| # | Name | Value | Hex Value |
| 0 | MTI | 0800 | 08 00 |
| 1.a | PRIMARY BITMAP | Chỉ ra sự hiện của trường 3,11,41 | A0 20 00 00 00 80 00 10 |
| 1.b | Secondary Bitmap | Chỉ ra hiện diện của trường 70 | 04 00 00 00 00 00 00 00 |
| 3 | PROCESSING CODE | 000000 | 00 00 00 |
| 11 | SYSTEM TRACE AUDIT NUMBER | 000001 | 00 00 01 |
| 41 | CARD ACCEPTOR TERMINAL IDENTIFICATION | 29110001 | 32 39 31 31 30 30 30 31 |
| 60 | RESERVE FOR PRIVATE USE | jPos 1.4.1 | 00 10 6A 50 4F 53 20 31 2E 34 2E 31 |
| 70 | NETWORK MANAGEMENT INFORMATION | 301 | 03 01 |

Chúng ta xem xét Primary Bitmap và Seconary Bitmap.

* Primary Bitmap

|  |  |  |  |
| --- | --- | --- | --- |
| Byte | Hex Value | Bit Value | Field # |
| 0 | A0 | **1**010 0000 | Sendcondary bitmap và field 3 |
| 1 | 20 | 0010 0000 | 11 |
| 2 | 00 | 0000 0000 |  |
| 3 | 00 | 0000 0000 |  |
| 4 | 00 | 0000 0000 |  |
| 5 | 80 | 1000 0000 | 41 |
| 6 | 00 | 0000 0000 |  |
| 7 | 00 | 0000 0000 |  |

***Bit đầu tiên trong chuỗi bit value của Primary Bitmap bật lên thành 1 cho biết thông điệp có chứa Seconary Bitmap.***

* Seconary Bitmap :

|  |  |  |  |
| --- | --- | --- | --- |
| Byte | Hex value | Bit value | Field # |
| 0 | 04 | 0000 0100 | 70 |
| 1 | 00 | 0000 0000 |  |
| 2 | 00 | 0000 0000 |  |
| 3 | 00 | 0000 0000 |  |
| 4 | 00 | 0000 0000 |  |
| 5 | 00 | 0000 0000 |  |
| 6 | 00 | 0000 0000 |  |
| 7 | 00 | 0000 0000 |  |

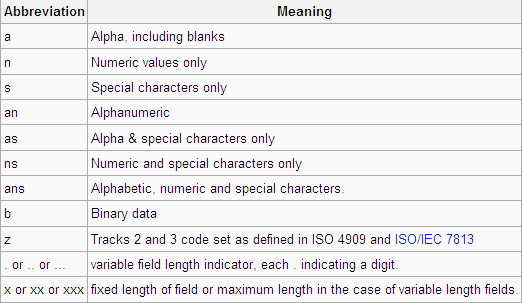
Chuyển giá trị thập lục phân về giá trị nhị phân rồi tiến hành xác định vị trí của các bit bật lên thành 1 để xác định chỉ số của các trường tương tự như Primary Bitmap.

### Thành phần dữ liệu (Data Element)

Thành phần dữ liệu (Data Element) hay còn được gọi là trường dữ liệu (Field) chứa các trường dữ liệu cá nhân và mang các thông tin giao dịch.

Có tới 128 trường dữ liệu trong chuẩn ISO8583 phiên bản 1987. Và lên đến 192 trường dữ liệu cho các phiên bản về sau.

Mỗi trường dữ liệu thì tuân theo một chuẩn định dạng để chỉ định nội dung của các trường và độ dài của trường đó. Bảng sau mô tả việc kí hiệu và ý nghĩa về các chuẩn định dạng cho một trường dữ liệu.



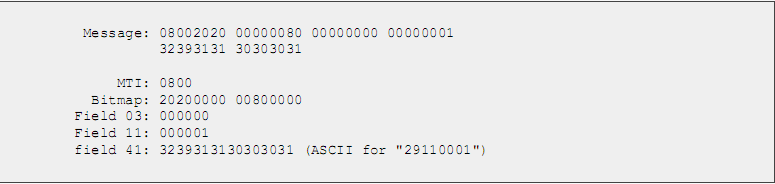
Chiều dài của các trường được chia làm 2 dạng :

* Chiều dài cố định (fixed length) kí hiệu x hay xx hay xxx đó là chiều dài tối đa mà trường đó có thể chứa.(xxxx chỉ dùng cho chuẩn ISO8583 phiên bản 2003).Các trường có chiều dài cố định thường không biểu diễn chiều dài trong nội dung thông điệp.[[2]](#footnote-3)
* Chiều dài có thể biến đổi (variable length) kí hiệu ‘**.’** hay ‘**..’** hay **‘…’** Mỗi ‘**.’** đại diện cho một chữ số. Các chiều dài có thể biến đổi trong quá trình biểu diễn phải thể hiện chiều dài trước nội dung trường đó.

Nội dung của các trường có thể là số,chữ hoặc là các kí hiệu đặc biệt hoặc là sự kết hợp của số, chữ và các kí hiệu đặc biệt.

Một điểm chúng ta cần lưu ý, chúng ta có một trường là số (numeric field) nhưng có thể biểu diễn nó bằng một chuỗi các kí tự ASCII hay EBCDIC.

Xem xét ví dụ sau :



Trong ví dụ của chúng ta thấy truờng số #03 biểu diễn theo kiểu BCD sử dụng 3 byte thập lục phân với giá trị của nó là “00 00 00”.Tương tự đối với trường #11 giá trị là “00 00 01” được biểu diễn bởi 3 byte thập lục phân giá trị “00 00 01”.Trường số #41 được biểu diễn gồm 8 byte (32 39 31 31 30 30 30 31) thập lục phân thể hiện 8 kí “29110001” trong bảng mã ASCII.

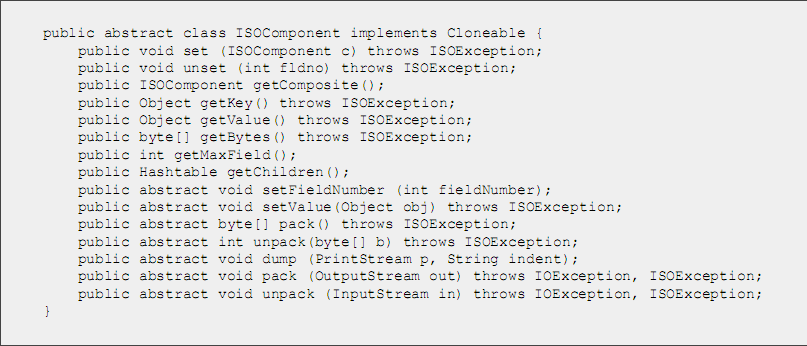
## JPOS

### JPOS cách thức xử lý ISO8583

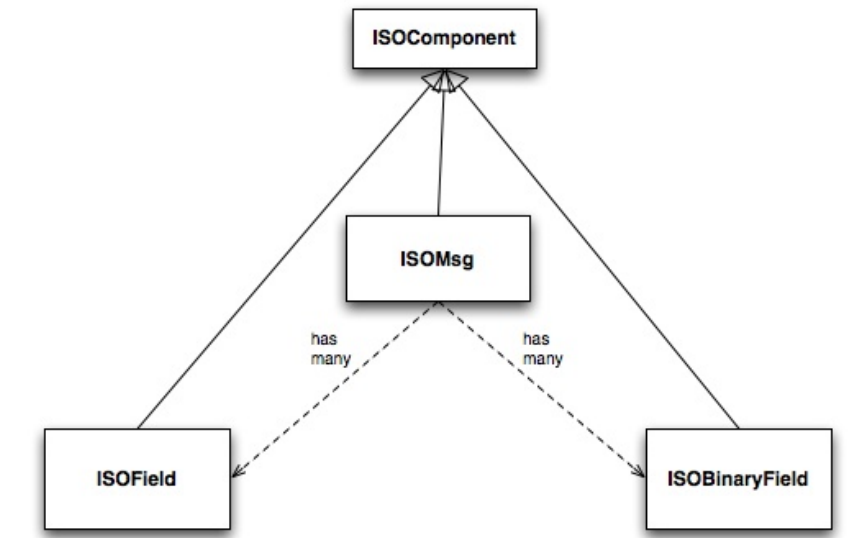
Phần này tập trung mô tả cách thức JPOS xử lý thông điệp ISO8583.

Đại diện cho một thông điệp ISO8583 JPOS sử dụng đối tượng ISOMsg.

ISOMsg,ISOField,ISOBitmapField,ISOBinaryField và bất cứ Custom Field nào của ISO8583 đều hiện thực từ lớp ISOComponent. Chúng ta xem xét các phương thức của lớp ISOComponent sau :



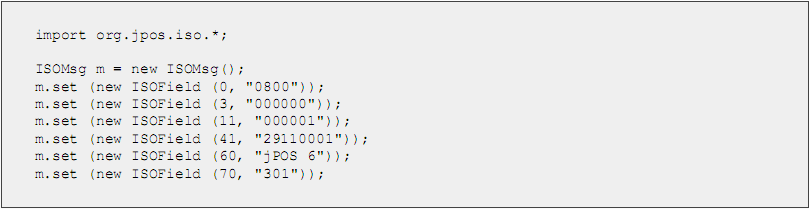
Chúng ta theo dõi hình 2-1 [[3]](#footnote-4)sau để xem cách mà ISOComponent tương tác với các lớp đối tượng khác :



Hình 2‑1

Các lớp đối tượng quản lý thông điệp ISO8583 được thiết kế theo mẫu Composite[[4]](#footnote-5) như hình trên.

Chúng ta có thể sử dụng phương thức ISOComponent.set(ISOComponent) để tạo một message ISO8583 như sau :



Đoạn mã nguồn trên tạo mới một thông điệp ISO8583. Sau đó khởi tạo MTI với giá trị 0800(MTI được xem là trường số 0 trong thông điệp 8583) và thiết lập giá trị cho các trường 3,11,41,60,70.

Tuy nhiên để rút gọn và code dễ đọc hơn JPOS cung cấp các phương thức tiện dụng như sau :

ISOMsg.setMTI(String)

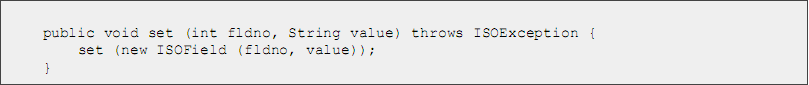
Phương thức trên thiết lập MTI cho message ISO8583.

Và

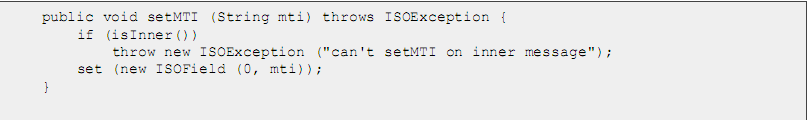
ISOMsg.set(int fieldNumber,String fieldValue)

Phương thức ISOMsg.set thiết lập trường giá trị (tham số fieldValue) cho trường ở vị trí được chỉ định (tham số fieldNumber).

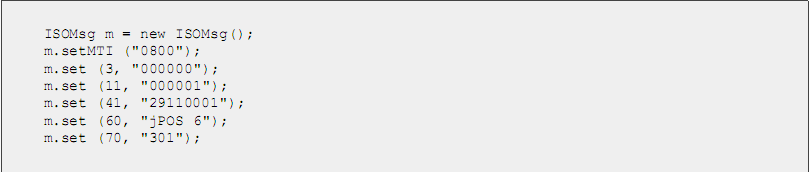
Thực sự bên trong các phương thức này bao bọc lại các phương thức còng kềnh phía bên trên :



Tương tự



Vì thế ISO8583 message ban đầu có thể viết lại như sau :



### Transaction trong JPOS

#### 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[[5]](#footnote-6).

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

* Atomic (Nguyên tử)
* Consistency (Nhất quán)
* Isolation (Độc lập)
* Durable(Duy trì)

#### Các tính chất của Transaction

##### 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 sinh viên. 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 nhất quán thể hiện ở chỗ các dữ liệu trao đổi qua lại giữ các hành động phải được nhất quán 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

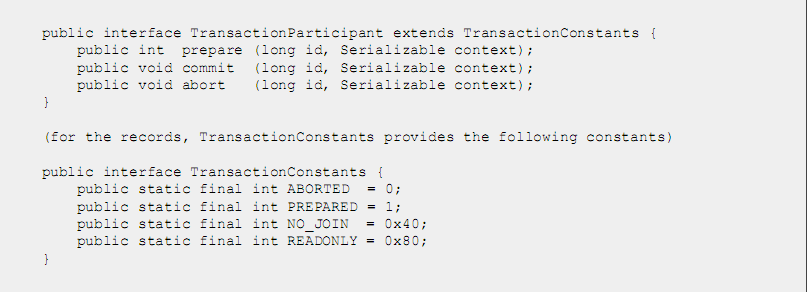
Thuộc tính này chỉ ra rằng khi một sự kiện xảy ra trên một hành động của transaction thì transaction manager sẽ đưa ra quyết định cho hành động đó hoặc thậm chí **đưa ra quyết định** cho toàn bộ các hành động của transaction hoặc các hành động có liên quan.

#### Transaction trong JPos 1.6.4

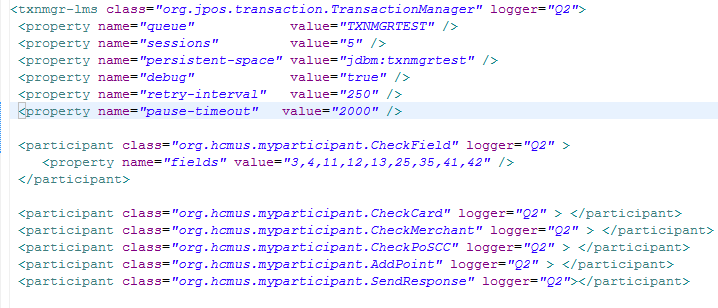
##### Participant

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 sau đó lưu lại trong thư mục deploy. Một điều đáng chú ý trong việc cấu hình transaction là thuộc tính class của thẻ gốc phải chỉ tới lớp ***org.jpos.transaction.TransactionManager***.



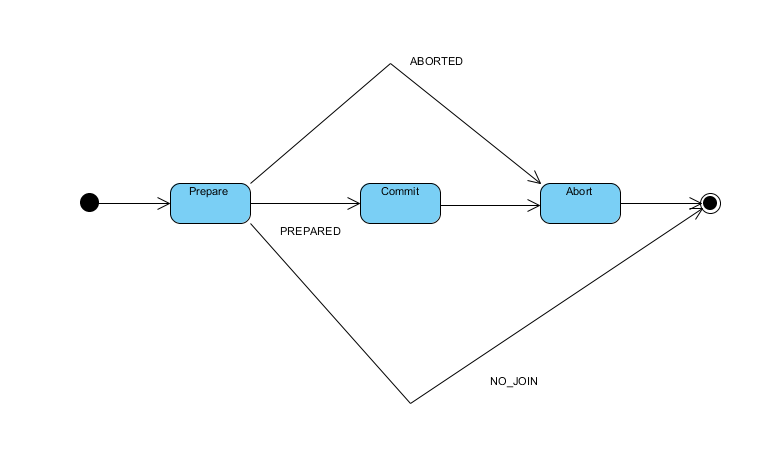
Các participant được cấu hình sẽ thực thi interface ***TransactionParticipant***.

##### 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 trả về PREPARED thì transaction sẽ chuyển đến pha commit và tại đây phương thức commit của tất cả các participant sẽ được gọi.

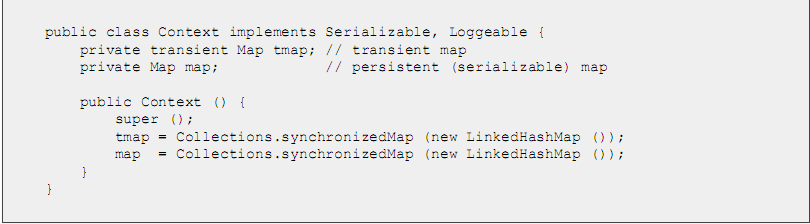
Nếu một trong các participant trả về ABORTED thì transaction sẽ chuyển đến pha abort. Và tại đây phương thức abort của tất cả các participant sẽ được gọi.

##### 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 nhất quán 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.



## Gateway

Phần này tập trung vào mô tả cơ sở lý thuyết liên quan đến các protocol và việc vận chuyển thông điệp trên đường truyền với các protocol chỉ định.

### Wire Protocol

Sau khi chúng ta đã có thông điệp ISO8583 chúng ta phải vận chuyển nó trên đường truyền sử dụng một vài giao thức giao tiếp như (TCP/IP, UDP/IP, X.25, SDLC, SNA, ASYNC, QTP, SSL,…..)

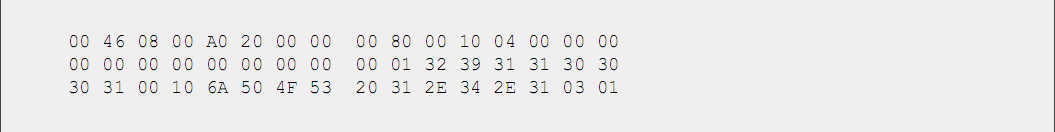
Nhưng các phương thức này không phải là một phần của thông điệp ISO8583 vì thế các nhà sản xuất và các doanh nghiệp có thể tùy chọn phương thức giao tiếp phù hợp với họ.

Một số giao thức(Đặc biệt là là các giao thức cũ) thường cần thêm header để cung cấp thông tin cho router. Trong khi một số khác thì cần có trailer.

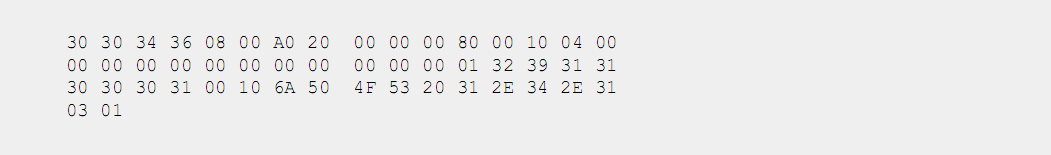
Vì vậy một chuỗi thông điệp truyền trên đường mạng (stream-based) thì thường gồm các phần sau :

* Header(Có thể không có)
* ISO8583 Message(Bắt buộc)
* Trailer(Có thể không có)

Với phương thức TCP/IP sử dụng 2 byte để chỉ ra chiều dài của chuỗi truyền trên đường truyền.Vì thế một chuỗi truyền 46 byte có thể truyền như sau :



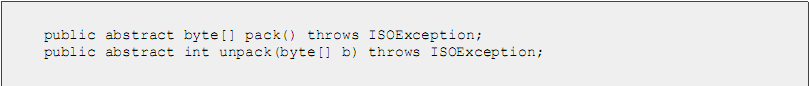
Tuy nhiên đây chỉ là một cách trong việc chỉ ra chiều dài chuỗi truyền.Chúng ta có thể chọn cách truyền chiều dài của chuỗi truyền dưới dạng ASCII như sau :



Trong đó 0046 🡪 30 30 34 36 (Giá trị ASCII của 0046)

### Packing và Unpacking

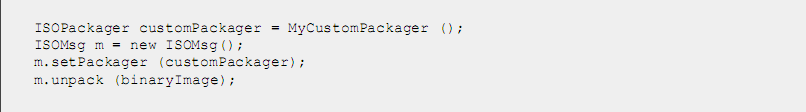
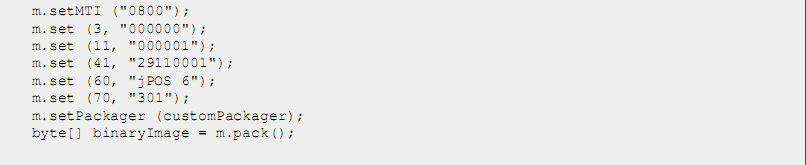
JPOS có hai phương thức khá hữu dụng :



Phương thức pack() trả về một mảng byte[] chứa giá trị đại diện của một thành phần cho trước( Thành phần này có thể là một field hoặc là toàn bộ message).

Phương thức unpack thì thực thi ngược lại tức là chuyển đổi một mảng byte[] trở thành giá trị của một field hoặc toàn bộ message.

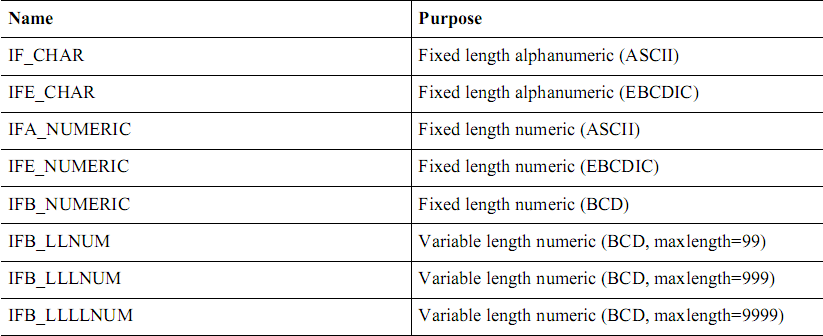
Do mỗi thông điệp chứa rất nhiều trường trong đó. Các trường được cấu hình theo những cách rất khác nhau (fix-length ASCII ,fix-length HEX, var length ASCII, var length HEX hoặc thậm chí ASCII kết hợp với HEX) do đó mỗi message đều được chỉ ra rằng các trường của mình được mô tả như thế nào thông qua packager. Packager đơn giản là một bản mô tả về message và nhờ bản mô tả này JPOS có thể packing/unpacking theo bản mô tả đó.



Ví dụ trên chỉ cho chúng ta cách thức JPOS packing và unpacking message thông qua packager.

Như đã nói ở trên mỗi package mô tả cho các trường của thông điệp ISO8583 mỗi trường trong packager đại diện bằng đối tượng ISOFieldPackager.

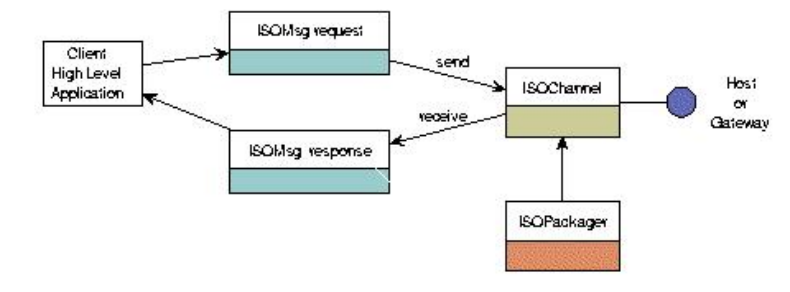
JPOS thiết kế đầy đủ các ISOFieldPackager và chúng ta hầu như không cần phải viết lại.



### Quản lý Wire Protocol với ISOChannel

JPOS sử dụng lớp đối tượng ISOChannel để quản lý Wire Protocol.

ISOChannel được sử dụng để gởi và nhận thông điệp ISOMsg.

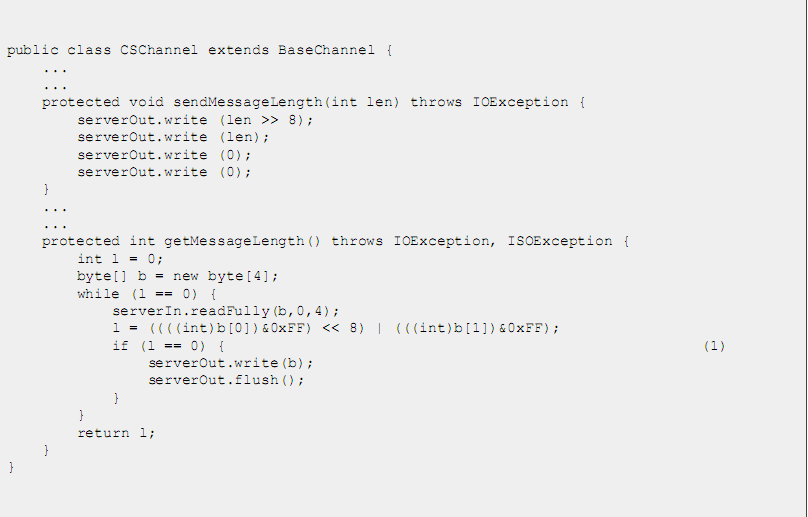


Hình 2‑2

Hình[[6]](#footnote-7) 2-2 thể hiện các thông điệp được truyền và nhận trên cùng một channel.

JPOS hổ trợ các loại channel thông dụng. Hầu hết các channel trên nền tảng TCP/IP được mở rộng từ lớp org.jpos.iso.BaseChannel. Các channel này đơn giản là override 2 phương thức sendMessageLength và getMessageLength.

Xem xét CSChannel (org.jpos.iso.channel.CSChannel) sử dụng 2 byte để chỉ ra chiều dài của thông điệp ISO 8583 cộng với header của nó và 2 byte dự trữ cho những mục đích mở rộng sau này.



Bây giờ chúng ta lướt qua các channel mà JPOS hỗ trợ :

|  |  |
| --- | --- |
| Channel Name | Wire Protocol |
| CSChannel | * LL LL 00 00 [header] ISO-DATA * LL LL 2 byte chỉ ra chiều dài của [Header] cộng với ISO8583 image * 00 00 để dành cho tương lai. * Header (lựa chọn). * ISO-DATA-8583 image (byte thô) |
| NACChannel | * LL LL [TPDU] ISO-DATA * LL LL 2 byte chỉ ra chiều dài của [TPDU] cộng với ISO8583 image * TPDU (Transport protocol data unit)[Lựa chọn] * ISO-DATA-8583 image (byte thô) |
| NCCChannel | * LL LL [TPDU] ISO-DATA * LL LL 2 byte chỉ ra chiều dài của [TPDU] cộng với ISO8583 image(Khác với NACChannel ở chỗ 2 byte này được gởi theo kiểu BCD) * TPDU (Transport protocol data unit)[Lựa chọn] * ISO-DATA-8583 image (byte thô) |
| ASCIIChannel | * LLLL[Header]ISO-Data * LLLL 4 byte ASCII chỉ ra chiều dài [Header] và ISO-DATA * Header (Lựa chọn) * ISO-DATA – 8583 image(byte thô) |
| RawChannel | * LL LL LL LL [header] ISO-DATA * LL LL LL LL chỉ ra chiều dài của [Header] và ISO-DATA * ISO-DATA -8583 Image(Byte thô) |

# : Thiết Kế Chương Trình

Chương tnày tập trung vào việc thiết kế và thực thi chương trình.

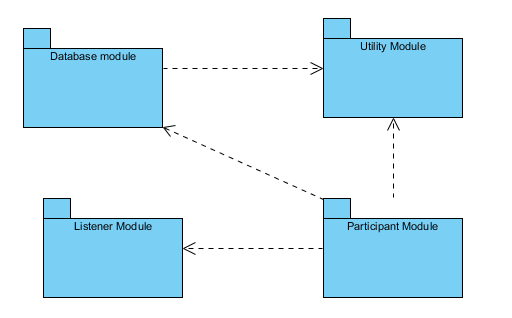
## Server

### Thiết kế kiến trúc của chương trình :

#### Thiết kế tổng quát

Hệ thống được thiết kế gồm 4 module chính : database module, participant module, utility module and listener module.

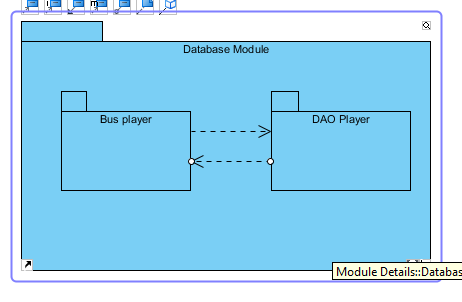
* Database module: Bao gồm tất cả các lớp và interface thực hiện liên kết và truy xuất cũng như tạo các lớp ánh xạ các bảng của database và thực thể của lớp. Module được thiết kế với kiến trúc 1 tier 2 player. Trong đó lớp DAO player được thiết kế theo mẫu Abstract Factory.Chi tiết cho module này được trình bày bên dưới.
* Participant module: Bao gồm tất cả các lớp liên quan đến nghiệp vụ và luồng của transaction.
* Utility module: Định nghĩa các lớp dùng chung hỗ trợ cho việc lấy giá trị của thông điệp ISO8583 và các lớp dùng chung khác cho tất cả các module đã đề cập ở trên.
* Listen module : Đây là module đơn giản nhất lắng nghe kết nối của client đồng thời đưa kết nối đó vào hàng đợi để xử lý.



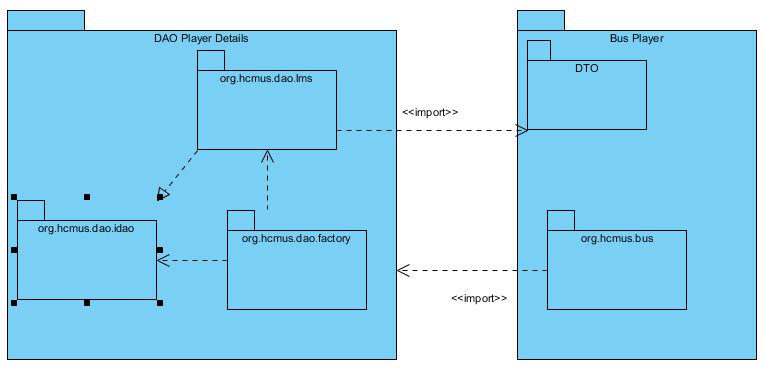
Hình ‑

#### Chi tiết của kiến trúc

##### Database module:



Hình ‑



Hình ‑

###### Org.hcmus.dao.package:

SQLDAOFactory

LMSDAOFactory

|  |  |
| --- | --- |
| Class | Description |
| LMSDAOFactory | Đây là lớp trừu tượng. Khi tạo một cơ sở dữ liệu mới bạn phải kế thừa từ lớp này. |
| SQLDAOFactory | Đây là một thể hiện của SQL database. |

###### Org.hcmus.dao.idao

|  |  |
| --- | --- |
| Interface | Description |
| IJPOS\_Card | Đây là interface cho lớp card |
| IJPOS\_Customer | Đây là interface cho lớp customer |
| IJPOS\_Gift | Đây là interface cho lớp gift |
| IJPOS\_Log\_Exchange | Đây là interface cho lớp log exchange |
| IJPOS\_Log | Đây là interface cho lớp log |
| IJPOS\_Merchant | Đây là interface cho lớp merchant |
| IJPOS\_PoSCC | Đây là interface cho lớp PoSCC |
| IJPOS\_Task | Đây là interface cho lớp task |

###### Org.hcmus.dao.lms

Tất cả các lớp trong package này thực thi từ interface trong ***org.hcmus.dao.idao***.

|  |  |
| --- | --- |
| Class | Description |
| JPOS\_CardDAO | Đây là lớp thiết kế cho card |
| JPOS\_CustomerDAO | Đây là lớp thiết kế cho customer |
| JPOS\_GiftDAO | Đây là lớp thiết kế cho for gift |
| JPOS\_Log\_ExchangeDAO | Đây là lớp thiết kế cho log exchange |
| JPOS\_LogDAO | Đây là lớp thiết kế cho log |
| JPOS\_MerchantDAO | Đây là lớp thiết kế cho merchant |
| JPOS\_PoSCCDAO | Đây là lớp thiết kế cho PoSCC |
| JPOS\_TaskDAO | Đây là lớp thiết kế cho task |

###### Org.hcmus.bus

Tất cả các lớp trong package này thực hiện nghiệp vụ cho các lớp tương ứng trong ***org.hcmus.dao.lms*** package.

|  |  |
| --- | --- |
| Class | Description |
| JPOS\_CardBUS | Đây là lớp nghiệp vụ của lớp JPOS\_CardDAO |
| JPOS\_CustomerBUS | Đây là lớp nghiệp vụ của lớp JPOS\_CustomerDAO |
| JPOS\_GiftBUS | Đây là lớp nghiệp vụ của lớp JPOS\_GiftBUS |
| JPOS\_Log\_ExchangeBUS | Đây là lớp nghiệp vụ của lớp JPOS\_Log\_ExchangeDAO |
| JPOS\_LogBUS | Đây là lớp nghiệp vụ của lớp JPOS\_LogDAO |
| JPOS\_MerchantBUS | Đây là lớp nghiệp vụ của lớp JPOS\_MerchantDAO |
| JPOS\_PoSCCBUS | Đây là lớp nghiệp vụ của lớp JPOS\_PoSCCDAO |
| JPOS\_TaskBUS | Đây là lớp nghiệp vụ của lớp JPOS\_TaskDAO |

##### Participant Module: org.hcmus.participant

|  |  |
| --- | --- |
| Class | Description |
| ActivatedCard | Kích hoạt thẻ với mã thẻ chỉ định. |
| AddPoint | Cộng số điểm chỉ định vào database ứng với thẻ giao dịch. |
| Balance Inquiry | Truy vấn số điểm dựa theo mã thẻ |
| CheckActivatedCard | Kiểm tra xem thẻ đã kích hoạt chưa. |
| CheckField | Kiểm tra các các field gởi đến. |
| CheckCard | Kiểm tra Card gởi đến.(Ngày hết hạn, sự tồn tại của Card) |
| CheckForward | Gởi mã lỗi về PROCESSING CODE cho client. |
| CheckGiftPoint | Kiểm tra điểm trước khi đổi quà. |
| CheckPoSCC | Kiểm tra mã các thiết bị được hệ thống hỗ trợ. |
| CheckMerchant | Kiểm tra Mã thiết bị và mã cửa hàng. |
| Close | Đóng Connection tới Database. |
| MakeAdvertisementMessage | Tạo thông điệp quảng cáo. |
| Open | Mở Connection tới Database. |
| Redemtion | Trừ điểm vào vào database ứng với món quà cần đổi của thẻ giao dịch. |
| SendActivationResponse | Trả kết quả kích hoạt về cho client. |
| SendResponse | Trả kết quả về cho client. |
| SubtractPoint | Trừ số điểm chỉ định vào database ứng với thẻ giao dịch. |

##### Utilily Module: org.hcmus.util

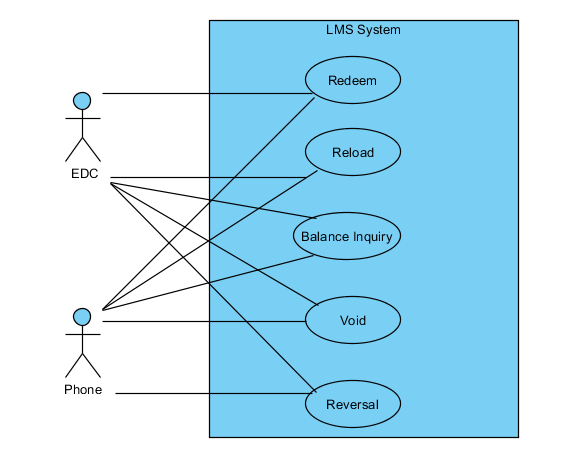
|  |  |
| --- | --- |
| Class | Description |
| Constants | Định nghĩa các hằng số cho hệ thống. |
| LMSConfig | Lấy database configuration từ xml file |
| MessageHelper | Hỗ trợ việc lấy thông tin đón nhận từ client |
| ConvertHelper | Hỗ trợ chuyển đổi giá trị từ các hệ đơn vị 10 sang 16 hoặc từ ASCII sang HEX |
| ILMSConfigMBean | Interface đăng kí LMSConfig với Mbean Server |
| LMSLogListener | LogListener của LMS |
| LMSLogSource | LogSource của LMS |
| NormalRule | Quy định số tiền đổi thành điểm |
| InirmalRule | Đăng kí NormalRule với Mbean server. |

##### Listener Module : org.hcmus.lms

|  |  |
| --- | --- |
| Class | Description |
| LMSISOListener | Nhận thông điệp từ client. |
| Switch | Phân phối các thông điệp về đúng các nhóm xử lý tác vụ. |

Sau khi đã thiết kế kiến trúc hệ thống chúng ta tiến hành mô hình hóa nghiệp vụ hệ thống.

### Mô hình hóa nghiệp vụ



Hình ‑

#### Actors

|  |  |
| --- | --- |
| Actor | Actor Description |
| EDC System | Các thiết đọc mã thẻ và chuyển mã thẻ lên server dưới dạng thông điệp ISO8583. |
| Phone | Thiết bị điện thoại có hỗ trợ độc barcode và có phần mềm chuyển thông điệp dưới dạng ISO 8583 lên server. |

#### User Cases

|  |  |  |
| --- | --- | --- |
| User Case ID | UserCase Name | UserCase Description |
| UC\_LMS\_01 | Redeem Point | Cộng điểm khi khách mua hàng. |
| UC\_LMS\_02 | Reload Point | Trừ điểm khi khách trả hàng. |
| UC\_LMS\_03 | Void | Đổi điểm lấy quà tặng |
| UC\_LMS\_04 | Banlance Inquiry | Truy vấn số điểm hiện có. |
| UC\_LMS\_05 | Activate card | Kích hoạt thẻ để sử dụng. |

### Đặc tả Usercase :

#### Đặc tả usercase Redeem

##### Tóm tắt

Usercase này nhầm cộng điểm cho khách hàng và trừ tiền vào thẻ ứng với số tiền mà họ đã mua.

##### Dòng dự kiện

###### Dòng sự kiện chính :

Sự kiện này bắt đầu khi người bán hàng dùng máy EDC quét mã thẻ của khách hàng đồng thời cung cấp số tiền khách hàng đã.

Hệ thống yêu cầu máy EDC cung cấp thông tin về : Mã thẻ, mã tác vụ,số tiền của khách hàng đã mua.

Hệ thống tiến hành kiểm chứng các giá trị được chuyển lên.

Hệ thống cộng điểm và tiến hành lưu lại vào log trên database.

Thông báo giao dịch thành công và trả thông tin quảng cáo cho người dùng.

###### Các dòng sự kiện khác

* Không gởi đủ các trường của thông điệp. Khi EDC chuyển lên không đủ các trường mà hệ thống yêu cầu thì hệ thống sẽ trả về người dùng thông báo lỗi các trường của thông điệp bị lỗi.
* Mã thẻ đã hết hạn sử dụng : khi EDC chuyển lên mã thẻ đã hết hạn sử dụng. Hệ thống sẽ trả về cho người dùng lỗi mã thẻ đã hết hạn sử dụng.
* Mã thẻ không tồn tại trong hệ thống : EDC chuyển lên mã thẻ chưa được đăng kí. Hệ thống sẽ trả về mã lỗi thẻ không tồn tại trong hệ thống.
* Mã thẻ chưa được kích hoạt : EDC chuyển lên mã thẻ chưa được kích hoạt. Hệ thống sẽ trả về mã lỗi thẻ chưa được kích hoạt trong hệ thống.
* TID và MID chưa được đăng kí trên hệ thống : Nếu EDC chuyển lên mã TID và MID chưa được đăng kí trên hệ thống thì hệ thống sẽ báo TID và MID không tồn tại.

##### Các yêu cầu đặc biệt

Không có

##### Trạng thái hệ thống khi bắt đầu thực hiện usercase.

Hệ thống server đã được kích hoạt và lắng nghe trên port đã cấu hình.

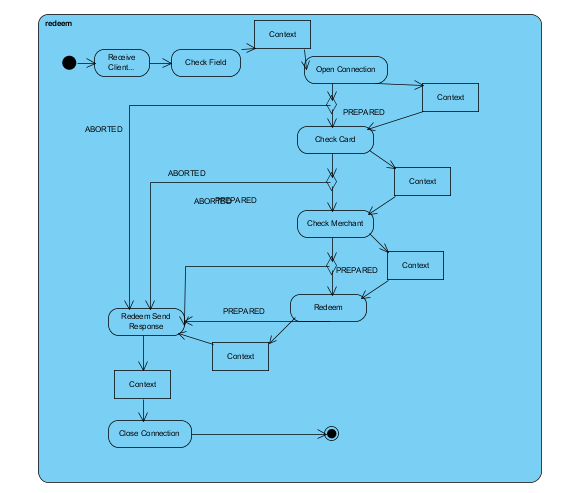
##### Trạng thái của hệ thống sau khi thực hiện usercase

Khách hàng nhận được số tiền còn lại và số điểm cộng của mình. Ngược lại giá trị không thay đổi.

##### Điểm mở rộng :

Không có.

##### Luồng sự kiện cho giao dịch redeem



#### Đặc tả usercase Reload

##### Tóm tắt:

Usercase này nhầm nạp thêm tiền vào thẻ của khách hàng.

##### Dòng dự kiện

###### Dòng sự kiện chính :

Sự kiện này bắt đầu khi người bán hàng dùng máy EDC quét mã thẻ của khách hàng đồng thời gởi số tiền cần nạp lên hệ thống.

* Hệ thống yêu cầu máy EDC cung cấp thông tin về : Mã thẻ, mã tác vụ,số tiền của khách hàng cần nạp.
* Hệ thống tiến hành kiểm chứng các giá trị được chuyển lên.
* Hệ thống trừ điểm và tiến hành lưu lại vào log trên database.
* Thông báo giao dịch thành công và trả thông tin và số tiền đã nạp cũng như tổng số tiền khách hàng hiện có.

###### Các dòng sự kiện khác

* Không gởi đủ các trường của thông điệp. Khi EDC chuyển lên không đủ các trường mà hệ thống yêu cầu hệ thống sẽ trả về người dùng thông báo lỗi các trường của thông điệp bị lỗi.
* Mã thẻ đã hết hạn sử dụng : khi EDC chuyển lên mã thẻ đã hết hạn sử dụng. Hệ thống sẽ trả về cho người dùng lỗi mã thẻ đã hết hạn sử dụng.
* Mã thẻ không tồn tại trong hệ thống : EDC chuyển lên mã thẻ chưa được đăng kí. Hệ thống sẽ trả về mã lỗi thẻ không tồn tại trong hệ thống.
* Mã thẻ chưa được kích hoạt : EDC chuyển lên mã thẻ chưa được kích hoạt. Hệ thống sẽ trả về mã lỗi thẻ chưa được kích hoạt trong hệ thống.
* TID và MID chưa được đăng kí trên hệ thống : Nếu EDC chuyển lên mã TID và MID chưa được đăng kí trên hệ thống thì hệ thống sẽ báo TID và MID không tồn tại.

##### Các yếu cầu đặc biệt

Không có

##### Trạng thái hệ thống khi bắt đầu thực hiện usercase.

Hệ thống server đã được kích hoạt và lắng nghe trên port đã cấu hình.

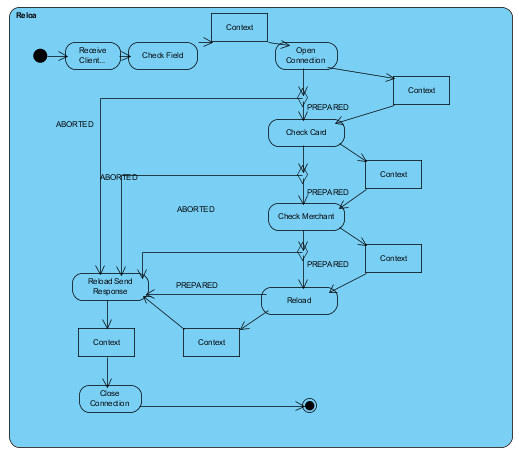
##### Trạng thái của hệ thống sau khi thực hiện usercase

Nếu usercase thành công khách hàng nhận được số tiền đã nạp và tổng số tiền trong hệ thống. Ngược lại giá trị không thay đổi.

##### Điểm mở rộng :

Không có.

##### Luồng Sự Kiện Cho usercase Reload



#### Đặc tả usercase Void

##### Tóm tắt:

Usercase này nhầm phục hồi lại các giao dịch trước đó như giao dịch redeem và giao dịch reload.

##### Dòng dự kiện

###### Dòng sự kiện chính :

Sự kiện này bắt đầu khi người bán hàng dùng máy EDC quét mã thẻ của khách hàng đồng thời mã tác vụ thực hiện là void.

* Hệ thống yêu cầu máy EDC cung cấp thông tin về : Mã thẻ, mã tác vụ,số tiền cần void,Mã phiều giao dịch của các giao dịch trước đó.
* Hệ thống tiến hành kiểm chứng các giá trị được chuyển lên.
* Hệ thống trừ điểm của khách hàng và tiến hành lưu lại vào log trên database.
* Thông báo giao dịch thành công và trả thông tin về số tiền đã void và tổng số tiền của khách hàng trong tài khoản.

###### Các dòng sự kiện khác

* Không gởi đủ các trường của thông điệp. Khi EDC chuyển lên không đủ các trường mà hệ thống yêu cầu hệ thống sẽ trả về người dùng thông báo lỗi các trường của thông điệp bị lỗi.
* Mã thẻ đã hết hạn sử dụng : khi EDC chuyển lên mã thẻ đã hết hạn sử dụng. Hệ thống sẽ trả về cho người dùng lỗi mã thẻ đã hết hạn sử dụng.
* Mã thẻ không tồn tại trong hệ thống : EDC chuyển lên mã thẻ chưa được đăng kí. Hệ thống sẽ trả về mã lỗi thẻ không tồn tại trong hệ thống.
* Mã thẻ chưa được kích hoạt : EDC chuyển lên mã thẻ chưa được kích hoạt. Hệ thống sẽ trả về mã lỗi thẻ chưa được kích hoạt trong hệ thống.
* TID và MID chưa được đăng kí trên hệ thống : Nếu EDC chuển lên mã TID và MID chưa được đăng kí trên hệ thống thì hệ thống sẽ báo TID và MID không tồn tại.
* Mã phiều giao dịch của các giao dịch trước (redeem hoặc reload) không tồn tại trong hệ thống.Hệ thống sẽ trả về mã lỗi mã phiều giao dịch không tồn tại trong hệ thống.

##### Các yêu cầu đặc biệt

Không có

##### Trạng thái hệ thống khi bắt đầu thực hiện usercase.

Hệ thống server đã được kích hoạt và lắng nghe trên port đã cấu hình.

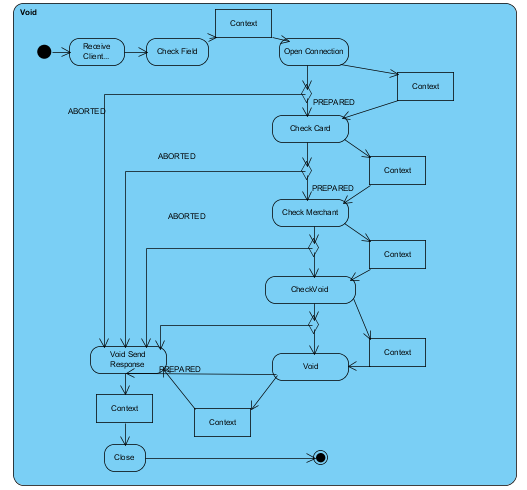
##### Trạng thái của hệ thống sau khi thực hiện usercase

Nếu Usercase thành công khách hàng sẽ nhận được số tiền đã void và tổng số tiền của họ torng hệ thống. Ngược lại giá trị không thay đổi.

##### Điểm mở rộng :

Không có.

##### Luồng Sự Kiện Cho Usercase Void



#### Đặc tả usercase Balance Inquiry

##### Tóm tắt:

Usercase này nhầm truy vấn xem trong thẻ có bao nhiêu tiền và bao nhiêu điểm.

##### Dòng dự kiện

###### Dòng sự kiện chính :

Sự kiện này bắt đầu khi người bán hàng dùng máy EDC quét mã thẻ của khách hàng đồng thời mã tác vụ thực hiện là truy vấn điểm.

* Hệ thống yêu cầu máy EDC cung cấp thông tin về : Mã thẻ, mã tác vụ
* Hệ thống tiến hành kiểm chứng các giá trị được chuyển lên.
* Hệ thống truy vấn điểm của khách hàng
* Thông báo giao dịch thành công và trả thông tin điểm cho người dùng.

###### Các dòng sự kiện khác

* Không gởi đủ các trường của thông điệp. Khi EDC chuyển lên không đủ các trường mà hệ thống yêu cầu hệ thống sẽ trả về người dùng thông báo lỗi các trường của thông điệp bị lỗi.
* Mã thẻ đã hết hạn sử dụng : khi EDC chuyển lên mã thẻ đã hết hạn sử dụng. Hệ thống sẽ trả về cho người dùng lỗi mã thẻ đã hết hạn sử dụng.
* Mã thẻ không tồn tại trong hệ thống : EDC chuyển lên mã thẻ chưa được đăng kí. Hệ thống sẽ trả về mã lỗi thẻ không tồn tại trong hệ thống.
* Mã thẻ chưa được kích hoạt : EDC chuyển lên mã thẻ chưa được kích hoạt. Hệ thống sẽ trả về mã lỗi thẻ chưa được kích hoạt trong hệ thống.
* TID và MID chưa được đăng kí trên hệ thống : Nếu EDC chyuển lên mã TID và MID chưa được đăng kí trên hệ thống thì hệ thống sẽ báo TID và MID không tồn tại.

##### Các yêu cầu đặc biệt

Không có

##### Trạng thái hệ thống khi bắt đầu thực hiện usercase.

Hệ thống server đã được kích hoạt và lắng nghe trên port đã cấu hình.

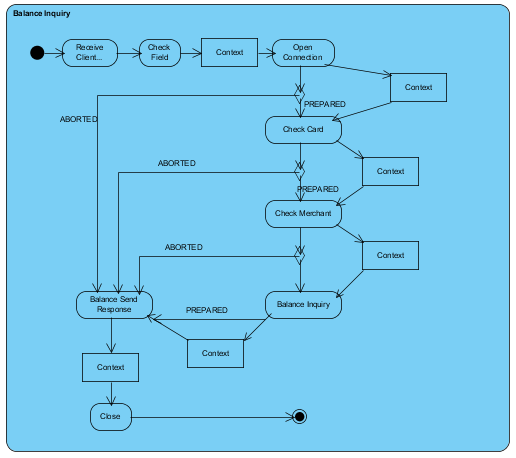
##### Trạng thái của hệ thống sau khi thực hiện usercase

Nếu usercase thành công thì khách hàng nhận được điểm và số tiền ứng với mã thẻ của mình.

##### Điểm mở rộng :

Không có.

##### Luồng sự kiện Balance Inquiry



### Thiết kế thông điệp cho các giao dịch

#### Redeem transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field No | Data Element Name | Attribute | Request | Respone | Note |
|  | MTI | n 4 | M | M | HEX |
|  | Bitmap | b 8 |  |  | HEX |
| 3 | Processing code | n 6 | M | M | HEX |
| 11 | System Trace Number | n 6 | M | M | HEX |
| 35 | Track 2 Data | z..37 | M |  | HEX |
| 39 | Response Code | an 2 |  | M | ASCII |
| 41 | Card Acceptor Terminal ID | ans 8 | M | M | ASCII |
| 42 | Card Acceptor Identification Code | ans 15 | M | M | ASCII |
| 48 | Reserse Private | ans…999 | M | M | Length of field (Hex) and Content of field (ASCII). |
| 61 | Reverse Private | ans …999 | M | M | Length of field (Hex) and Content of field (ASCII). Use Tag Length value for each message. |

Bảng đặc tả cho cách trường :

##### MTI (Message Type Indicator) :

|  |  |  |
| --- | --- | --- |
| Request | Respone | Description |
| 0200 | 0210 | Redeem |

##### Field No : 3 Processing code

Processing code chia làm 3 trường nhỏ :

* Transaction Type code,AN2
* Account Type Code 1,AN2
* Account Type Code 2,AN2
* Request

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0200 | 888888 | Redeem |

* Response

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0210 | 888888 | Redeem |

##### Field 48

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Description |
| FF21 | 02 | Báo cho server biết đây là giao dịch redeem với giá trị 7207 |
| FF2F | 06 | Mã phiếu thanh toán cho redeem |

##### Field 61

###### Request

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Description |
| FF3C | 14 | Chứa số tiền cần redeem |

###### Response

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Description |
| FF3E | 16 | Tổng số tiền hiện có |
| FF41 | 16 | Số tiền đã gởi lên redeem |
| FF58 | 05 | Số điểm hiện có của khách hàng trong hệ thống. |

##### Message sample :

###### Message request:

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0200 🡺 2 byte (hex)

**Bitmap** : 2020000020C10008 🡺 8 byte (Hex)

**F3** : 888888 🡺 3 byte (Hex)

**F11** : 000006 🡺 3 byte (hex)

**F35** : 379704215000000046D140610114680300000000 🡺 20 byte ( hex)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

F48: 0032**FF21027207**FF2203000000FF25091655D7C5A50A1B2C7B**FF2F06051909580106**

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF21 | 02 | 7207 |
| FF2F | 06 | 051909580106 |
| FF22 | 03 | 000000 |
| FF25 | 09 | 1655D7C5A50A1B2C7B |

Field 48 length : 2 + 1 + 2 + 2 + 1 + 6 + 2 + 1 + 3 + 2 + 1 + 9 = 32 + 2 = 34 byte

**F61**:

0023FF3C140002000000000000000000000000000000000000

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF3C | 14 | **00020000**00000000000000000000000000000000 |

Tổng chiều dài : 5 + 2 + 8 + 3 + 3 + 20 + 8 + 15 + 34 + 25 = 123 🡺 2 byte (hex) 007B

Message hoàn chỉnh :

007B600353000002002020000020C10008888888000006379704215000000046D14061011468030000000038313130313030313831313030393939393930303030310032FF21027207FF2203000000FF25091655D7C5A50A1B2C7BFF2F060519095801060023FF3C140002000000000000000000000000000000000000

###### Message Response :

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0210 🡺 2 byte (hex)

**Bitmap** : 2020000020C1000 🡺 8 byte (Hex)

**F3** : 888888 🡺 3 byte (Hex)

**F11** : 000006 🡺 3 byte

**F39** : 3030 🡺 2byte (ASCII)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

**F48**: 0032**FF21027207**FF2203000000FF25091655D7C5A50A1B2C7B**FF2F06051909580**

**F61** :

0058FF3E1611110000000000000000000000000000000014000000FF411611110000000000000000000000000000000002000000FF58050000001400

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Value |
| FF3E | 16 | 111100000000000000000000000000000000**140000**00 |
| FF41 | 16 | 1111000000000000000000000000000000000**20000**00 |
| FF58 | 05 | **00000014**00 |

**Total F61 length** : 25 + 25 + 08 = 58 + 2 = 60

**Tổng chiều dài** : 5 + 2 + 8 + 3 + 3 + 8 + 15 + 34 + 60 = 138 🡺 2 byte (hex) 008A

008A600353000002102020000020C10008888888000006379704215000000046D140610114680300000000303038313130313030313831313030393939393930303030310032FF21027207FF2203000000FF25091655D7C5A50A1B2C7BFF2F060519095801060058FF3E1611110000000000000000000000000000000014000000FF411611110000000000000000000000000000000002000000FF58050000001400

#### Reload Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field No | Data Element Name | Attribute | Request | Respone | Note |
|  | MTI | n 4 | M | M | HEX |
|  | Bitmap | b 8 |  |  | HEX |
| 3 | Processing code | n 6 | M | M | HEX |
| 11 | System trace number | n 6 | M | M | HEX |
| 35 | Track 2 Data | z..37 | M |  | HEX |
| 39 | Response Code | an 2 |  | M | ASCII |
| 41 | Card Acceptor Terminal ID | ans 8 | M | M | ASCII |
| 42 | Card Acceptor Identification Code | ans 15 | M | M | ASCII |
| 48 | Reserse Private | ans…999 | M | M | Length of field (Hex) and Content of field (ASCII) |
| 61 | Reverse Private | ans …999 |  | M | Length of field (Hex) and Content of field (ASCII). Use Tag Length value for each message |

##### MTI (Message Type Indicator) :

|  |  |  |
| --- | --- | --- |
| Request | Respone | Description |
| 0200 | 0210 | Reload |

##### Field No : 3 Processing code

Processing code chia làm ba thành phần:

* Transaction Type code,AN2
* Account Type Code 1,AN2
* Account Type Code 2,AN2 (skip : default : 00)

**Request :**

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0200 | 888888 | Reload |

**Response :**

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0210 | 888888 | Reload |

##### Field 48

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Description |
| FF21 | 02 | Báo cho server biết đây là giao dịch reload với giá trị 7707 |
| FF2F | 06 | Mã phiếu thanh toán |
| FF2B | 05 | Chứa số tiền cần reload. |

**Field61** :

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Value |
| FF3E | 16 | Tổng số tiền hiện có |
| FF41 | 16 | Số tiền đã nạp |

##### Message sample :

###### Message request:

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0200 🡺 2 byte (hex)

**Bitmap** : 2020000020C10000 🡺 8 byte (Hex)

**F3** : 888888 🡺 3 byte (Hex)

**F11** : 000013 🡺 3 byte (hex)

**F35** : 328704353300000015D999912310000000 🡺 17 byte ( hex)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

**FF48**:

0028FF21027707FF2203000000FF2B050000040000FF2F06052615240213

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF21 | 02 | 7707 |
| FF22 | 03 | 000000 |
| FF2B | 05 | 0000040000 |
| FF2F | 06 | 052615240213 |

Tổng chiều dài : 5 + 2 + 8 + 3 + 3 + 17 + 8 + 15 + 30 = 91 🡺 2 byte (hex) 005B

005B600353000002002020000020C10000888888000013328704353300000015D99991231000000038313130313030313831313030393939393930303030310028FF21027707FF2203000000FF2B050000040000FF2F06052615240213

###### Message Response :

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0210 🡺 2 byte (hex)

**Bitmap** : 2000000000C10008 🡺 8 byte (Hex)

**F3** : 8888888 🡺 3 byte (Hex)

**F11** : 000013 🡺 3 byte (hex)

**F39** : 3030 => 2 byte (ASCII)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

**F48**: 0028FF21027707FF2203000000FF2B050000040000FF2F06052615240213 🡺 30 byte

**F61** :

0050FF3E1611110000000000000000000000000000000043000000FF411687070000000000000000000000000000000004000000

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF3E | 16 | 1111000000000000000000000000000000**00430000**00 |
| FF41 | 16 | 8707000000000000000000000000000000**00040000**00 |

Total F61 : 2 + 1 + 22 + 2 + 1 + 22 = 50 + 2 = 55

Tổng chiều dài : 5 + 2 + 8 + 3 + 3 + 2 + 8 + 15 + 30 + 50 = 126 🡺 2 byte (hex) 007E

007E600353000002102020000002C100088888880000133030383131303130303138313130303939393939303030303130300028FF21027707FF2203000000FF2B050000040000FF2F060526152402130050FF3E1611110000000000000000000000000000000043000000FF411687070000000000000000000000000000000004000000

#### Balance Inquiry Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field No | Data Element Name | Attribute | Request | Respone | Note |
|  | MTI | n 4 | M | M | HEX |
|  | Bitmap | b 8 |  |  | HEX |
| 3 | Processing code | n 6 | M | M | HEX |
| 11 | System trace number | n 6 | M | M | HEX |
| 35 | Track 2 Data | z..37 | M |  | HEX |
| 39 | Response Code | an 2 |  | M | ASCII |
| 41 | Card Acceptor Terminal ID | ans 8 | M | M | ASCII |
| 42 | Card Acceptor Identification Code | ans 15 | M | M | ASCII |
| 48 | Reserse Private | ans…999 | M | M | Length of field (Hex) and Content of field (ASCII) |
| 61 | Reverse Private | Ans …999 |  | M | Length of field (Hex) and Content of field (ASCII). Use Tag Length value for each message |

##### MTI (Message Type Indicator) :

|  |  |  |
| --- | --- | --- |
| Request | Respone | Description |
| 0200 | 0210 | Balance Inquiry |

##### Field No : 3 Processing code

Processing code chia làm ba thành phần:

* Transaction Type code,AN2
* Account Type Code 1,AN2
* Account Type Code 2,AN2 (skip : default : 00)

**Request**

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0200 | 888888 | Balance Inquiry |

**Response**

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0210 | 888888 | Balance Inquiry |

##### Field 48

|  |  |  |
| --- | --- | --- |
| Tag | Length | Description |
| FF21 | 02 | Báo cho server biết đây là giao dịch truy vấn điểm với giá trị là 7567 |

Field 61:

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Value |
| FF3E | 16 | Tổng số tiền của thẻ trong hệ thống |

##### Message sample :

###### Message request:

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0200 🡺 2 byte (hex)

**Bitmap** : 2020000020C10000 🡺 8 byte (Hex)

**F3** : 427000 🡺 3 byte (Hex)

**F11** : 000008 🡺 3 byte

**F35** : 379704215000000046D140610114680300000000 🡺 20 byte ( hex)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

F48 :

0023**FF21027567**FF2203000000FF25091655D7C5A50A1B2C7B

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF21 | 02 | 7567 |

Tổng số tiền : 5 + 2 + 8 + 3 + 3 + 20 + 8 + 15 + 25 = 89 🡺 2 byte (hex) 0059

0059600353000002002020000020C10000888888000008379704215000000046D14061011468030000000038313130313030313831313030393939393930303030310023FF21027567FF2203000000FF25091655D7C5A50A1B2C7B

###### Message Response :

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0210 🡺 2 byte (hex)

**Bitmap** : 2020000002C1000 🡺 8 byte (Hex)

**F3** : 8888888 🡺 3 byte (Hex)

**F11** : 000008 🡺 3 byte

**FF39** : 3030 🡺 2byte (ASCII)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

**F48**: 0023FF21027567FF2203000000FF25091655D7C5A50A1B2C7B 🡺 25 byte

**F61** :

0025FF3E1611110000000000000000000000000000000016000000

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF3E | 16 | 1111000000000000000000000000000000**00160000**00 |

Total F61 : 2 + 1 + 22 = 25 + 2 = 27 ( 2byte to send 0044)

To63g chiều dài : 5 + 2 + 8 + 3 + 3 + 2 + 8 + 15 + 25 + 27 = 98 🡺 2 byte (hex) 0062

0062600353000002102020000002C10008888888000008303038313130313030313831313030393939393930303030310023FF21027567FF2203000000FF25091655D7C5A50A1B2C7B0025FF3E1611110000000000000000000000000000000016000000

#### Void Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field No | Data Element Name | Attribute | Request | Respone | Note |
|  | MTI | n 4 | M | M | HEX |
|  | Bitmap | b 8 |  |  | HEX |
| 3 | Processing code | n 6 | M | M | HEX |
| 4 | Amount Transaction | n 12 | M |  | HEX(Point is contained here) |
| 11 | System Trace Audit Number | n 6 | M | M | HEX |
| 35 | Track 2 Data | z..37 | M |  | HEX |
| 39 | Response Code | an 2 |  | M | ASCII |
| 41 | Card Acceptor Terminal ID | ans 8 | M | M | ASCII |
| 42 | Card Acceptor Identification Code | ans 15 | M | M | ASCII |
| 48 | Reserse Private | ans…999 | M | M | Length of field (Hex) and Content of field (ASCII) |
| 61 | Reverse Private | Ans …999 |  | M | Length of field (Hex) and Content of field (ASCII). Use Tag Length value for each message |

##### MTI (Message Type Indicator) :

|  |  |  |
| --- | --- | --- |
| Request | Respone | Description |
| 0200 | 0210 | Redemption |

##### Field No : 3 Processing code

Processing code chia làm ba thành phần:

* Transaction Type code,AN2
* Account Type Code 1,AN2
* Account Type Code 2,AN2 (skip : default : 00)

Request

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0200 | 888888 | Redemption |

Response

|  |  |  |
| --- | --- | --- |
| MTI | TTC | Description |
| 0210 | 888888 | Redemption |

##### Field 48

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Description |
| FF21 | 02 | Báo cho server biết đây là giao dịch void với giá trị 7208 với void cho redeem và 7708 đối với giao void cho reload |
| FF2F | 06 | Mã phiếu thanh giao dịch void |
| FF34 | 0F | Chứa mã số phiếu của giao dịch cần void |

##### Field 61:

###### Request

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF3C | 14 | **Gởi số tiền cần void** |

Response

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Value |
| FF3E | 16 | Tổng số tiền của thẻ |
| FF41 | 16 | Số tiền đã void |

##### Message sample :

###### Message request:

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0200 🡺 2 byte (hex)

**Bitmap** : 3020000020C1000 🡺 8 byte (Hex)

**F3** : 888888 🡺 3 byte (Hex)

**F4** : 000000000000 🡺 6 byte (hex)

**F11** : 000000 🡺 3 byte (hex)

**F35** : 288704353300000015D99991231000 🡺 15 byte ( hex)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

**F48**:

0032FF21027708FF2F06052515412008FF340F020072070000000000052615240213

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF21 | 02 | 7708 |
| FF2F | 06 | 052515412008 |
| FF34 | 0F | 020072070000000000052615240213 |

Field 48 length : 2 + 1 + 2 + 2 + 1+ 6 + 2 + 1 + 15 = 32 + 2 = 34 byte

**F61**:

0023FF3C140001000000000000000000000000000000000000

|  |  |  |
| --- | --- | --- |
| Tag | Length | Value |
| FF3C | 14 | **00010000**00000000000000000000000000000000 |

Field 61 length : 2 + 1 + 20 = 23 + 2 = 25 byte

Tổng chiều dài : 5 + 2 + 8 + 3 + 6 + 3 + 15 + 8 + 15 + 34 + 25 = 124

🡺 2 byte (hex) 007C

007C600353000002003020000020C10008888888000000000000000008288704353300000015D9999123100038313130313030313831313030393939393930303030310032FF21027708FF2F06052515412008FF340F0200720700000000000526152402130023FF3C140001000000000000000000000000000000000000

###### Message Response :

**Header** : 6003530000 🡺 5 byte (hex)

**MTI** : 0210 🡺 2 byte (hex)

**Bitmap** : 3020000020C1000 🡺 8 byte (Hex)

**F3** : 888888 🡺 3 byte (Hex)

**F11** : 000000 🡺 3 byte (hex)

**F35** : 288704353300000015D99991231000 🡺 15 byte ( hex)

**F39** : 3030 🡺 2 byte (ASCII)

**F41** : 3831313031303031 🡺 8 byte(ASCII) 🡺 (222)

**F42**: 383131303039393939393030303031 🡺 15 byte (ASCII)

F48 :

0032FF21027708FF2F06052515412008FF340F020072070000000000052615240213 🡺 48 byte

**F61** :

0050FF3E1611110000000000000000000000000000000039000000FF411682080000000000000000000000000000000001000000

|  |  |  |
| --- | --- | --- |
| Tag | Length(Hex) | Value |
| FF3E | 16 | 1111000000000000000000000000000000**00390000**00 |
| FF41 | 16 | 8208000000000000000000000000000000**00010000**00 |

Field 61 length : 2 + 1 + 22 + 2 + 1 + 22 = 50 + 2 = 52 byte.

Total Message length : 5 + 2 + 8 + 3 + 3 + 15 + 2 + 8 + 15 + 48 + 52 = 161 🡺 2 byte (hex) 00A1

00A1600353000002103020000020C1000888888288704353300000015D99991231000303038313130313030313831313030393939393930303030310032FF21027708FF2F06052515412008FF340F0200720700000000000526152402130050FF3E1611110000000000000000000000000000000039000000FF411682080000000000000000000000000000000001000000

#### Đặc tả cho các trường thông điệp chung :

##### Field No : 35 Track 2 Data

Trường này chứa giá trị trong track thứ 2 của thẻ từ.Thường sử dụng để chứa mã thẻ.

|  |  |
| --- | --- |
| Name | Format |
| Primary Account Number | up to 16 digit |
| Field Separator | can be either “=” or “D” |
| Expiration Day | in YYMM format |
| Service Restriction Code | Three Digit |
| Discretionary Data | balance of avaiable digits |

##### Field No : 39 Response code

Trường này trả về mã thông điệp từ client. Giá trị của mã thông điệp được lưu trong trường 61.

|  |  |
| --- | --- |
| **Value** | **Description** |
| 00 | TRANSACTION SUCESSFULLY |
| 03 | MID OR TID NOT FOUND |
| 14 | INVALID CARD NUMBER |
| 54 | EXPIRE CARD |
| 12 | OTHER ERROR |
| 15 | INVALID FIELD |

##### Filed No : 41 Card Acceptor Terminal ID “TID”

Trường này chứa mã di nhất xác định nguồn gốc xuất xứ của thông điệp.Thường chứa mã của thiết bị đã gởi thông điệp.

##### Field No 42 Card Acceptor Identify Code “MID”

Trường này thường chứa mã của nơi chấp nhận sử dụng thẻ.

##### Field 61

Gởi thông báo về cho client.

##### Conditional Code

|  |  |
| --- | --- |
| Type | Comment |
| O | Option |
| M | Mandantory |

### Thực thi chương trình

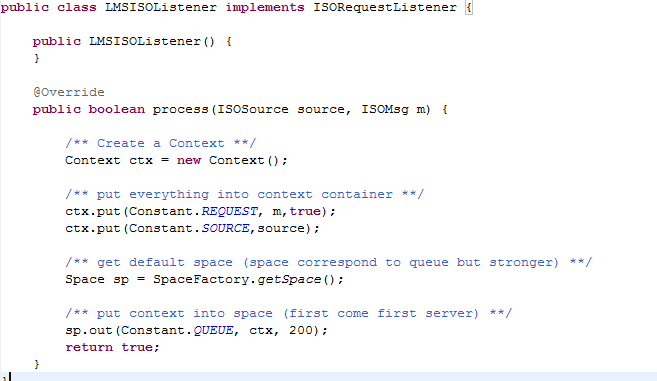
Khi một thông điệp gởi đến chương trình theo đúng kênh truyền(channel) và cổng(port) mà server đang lắng nghe (Việc cấu hình và đăng kí server với QSP và Q2 xem phần phụ lục QSP và Q2). Server sử dụng đối tượng thực thi từ Interface ISORequestListener trong đó có 2 tham số :

* Một là ISOSource : đây thực chất là một socket nhận nhiệm vụ trả message về cho client sau khi đã xử lý.
* Hai là ISOMsg : đây là thông điệp nhận từ client.

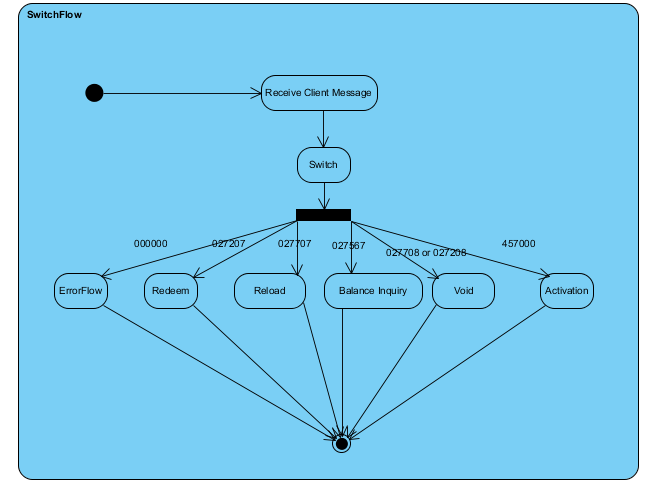
Tuy nhiên vì một bussiness thực hiện qua rất nhiều các tác vụ con (participant) nên thông điệp nhận được này cần được xử lý như một transaction để có thể đối phó tốt khi có lỗi xảy ra hay các luồng sự kiện phụ xảy ra.

Như đã đề cập trong phần cơ sở lý thuyết về transaction phía trên. Chúng ta cần sử dụng một đối tượng Context. Đối tượng này nhầm mục đích duy trì và độc lập giá trị giữa các transaction với nhau.

Sau đó chúng ta sẽ đặt Context này vào trong Space (Xem phần phụ lục về Space) để đảm bảo tính tuần tự khi xử lý cũng như chuyền tay Context này qua TransactionManager quản lý.



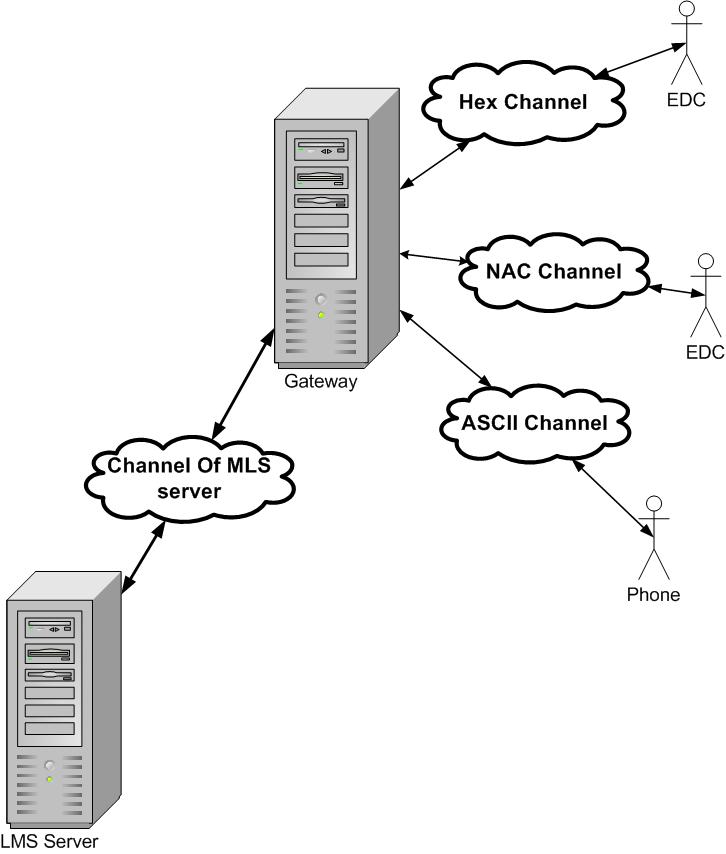
Khi một Context chuyển qua TransactionManager thì Context đó sẽ được TransactionManage lấy ra từ trong Space sau đó khởi tạo một transaction. Và cuộc đời của Transaction bắt đầu từ đây. Context này sẽ  ‘lăn’ qua tất cả các (Atomic Bussiness của Transaction mà trong JPOS gọi là participant ). Và Participant đầu tiên mà Context phải trãi qua đó là Switch làm nhiệm vụ chụp bắt và kiểm tra MIT và trường số 3 (Processing Code) của message chứa trong Context để biết tác vụ cần thực hiện. Khi đã xác định được tác vụ cần thực hiện thì switch sẽ forward thông điệp trên qua đúng tác vụ xử lý cho thông điệp đó.



Sau khi về đúng với các tác vụ cần thực hiện Context sẽ lăn tiếp qua các participant trong tác vụ đó.

## GateWay (Multi Channel)

Như đã đề cập ở trên thông điệp truyền trên đường truyền theo các kênh truyền khác nhau. Có kênh truyền sử dụng header có kênh truyền thì không sử dụng, có kênh truyền gởi thông điệp theo dạng Hex lại có kênh truyền gởi thông điệp theo dạng ASCII.Cụ thể đối với thiết bị di động ở xa, đường truyền tương đối yếu hơn so với các thiết bị EDC đặt tại chỗ kết nối trực tiếp với hệ thống như một mạng LAN vì thế điện thoại thường sử dụng channel với mã HEX để truyền thông điệp nnhưng EDC lại chọn mã ASCII để truyền.



Vì vậy việc đề xuất ra một gateway đóng vai trò trung gian và chuyển đổi các kênh truyền độc lập với server là một điều hết sức khả thi.Đó là lý do mà gateway ra đời.

Gateway sẽ nhận các thông điệp trên các kênh truyền do client chuyển đến chuyển kênh truyền phù hợp với kênh truyền của Server và sau đó nhận thông điệp theo kênh truyền của Server và trả thông điệp về theo kênh truyền của client.

Xét một một chuỗi thông điệp sau :

0043600003000002003000000020C00000407000000000200000379604215000000046D140610114680300000000**3232322020202020322020202020202020202020202020**

0038600003000002003000000020C00000407000000000200000379604215000000046D140610114680300000000**000000010000000000000010**

Chuỗi thông điệp đầu được cấu hình dưới channel trong đó trường số 41 và trường số 42 của thông điệp được gởi dưới dang ASCII vì thế chiều dài tổng cộng của thông điệp là 67 byte (0x43).

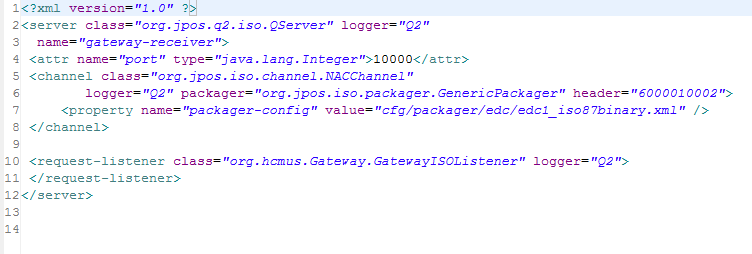
Xét chuổi thông điệp thứ hai trong đó trường số 41 và trường số 42 được gởi theo kiểu HEX vì thế độ dài của thông điệp ngắn hơn chỉ có 56 byte(0x38).

Nhiêm vụ của Gateway là sẽ nhận chuỗi thông điệp thứ 2 sau đó chuyển chuỗi thông điệp thứ 2 về đúng định dạng của chuổi thông điệp thứ nhất và gởi lên cho server.

**Để làm được việc này thì Gateway tiến hành như sau :**

Giá trị đầu vào của Gateway đó là các packager của Server và của mỗi client với các kênh truyền khác nhau.

Gateway thiết kế các server(Server của Gateway) với các port khác nhau để đón các thông điệp trên các kênh truyền khác nhau. Vì thế bài toán đa kênh truyền trở thành bài toàn đa port.



Hình 3‑5

Hình trên biểu diễn việc cấu hình cho một server lắng nghe trên port 10000 với kênh truyền edc1\_iso87binary.xml. Và thông điệp đến trên server này sẽ được lắng nghe như sau :

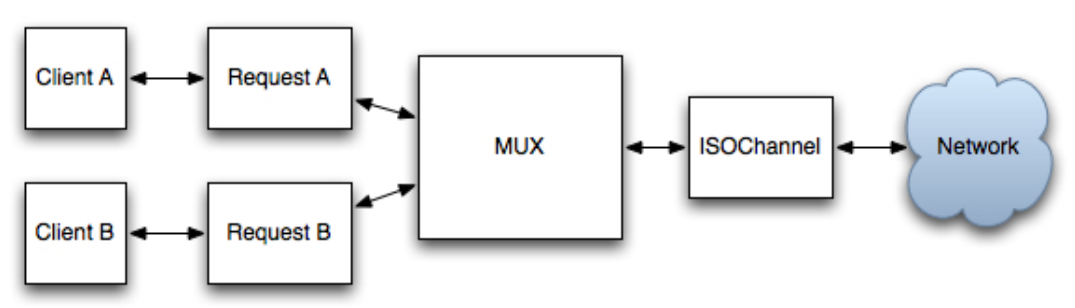
ISOMsg response = sender.Request(msg);

source.send(response);

Sử dụng đối tượng Sender để gởi thông điếp đến server đồng thời chờ thông điệp trả về sau đó gởi thông điệp trở về cho client.

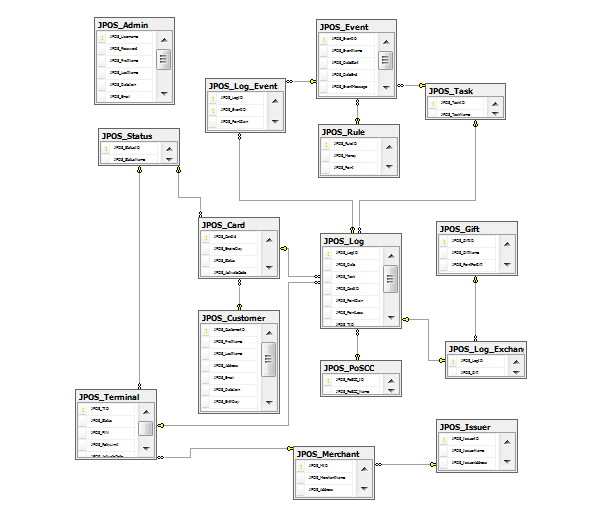
**Xem xét đối tượng Sender**.

Sender đơn giản là một đối tượng MUX được cấu hình trong file xml và đăng kí với Q2 server. MUX bảo đảm rằng một thông điệp dưới client sẽ chuyển thông điệp lên server một cách tuần tự và nhận kết quả trả về một cách tuần tự. (lưu ý MUX được dùng ở client không được dùng như một server)



## Database

### Sơ đồ logic



### Mô tả chi tiết các bảng dữ liệu trong sơ đồ logic

#### JPOS\_Admin

Lưu trữ thông tin Admin của hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_Username | nvarchar(50) | PK | Tên dùng để đăng nhập |
| 2 | JPOS\_Password | nvarchar(50) | not null | Mật khẩu |
| 3 | JPOS\_FirstName | nvarchar(50) | not null | Họ của Admin |
| 4 | JPOS\_LastName | nvarchar(50) | not null | Tên của Admin |
| 5 | JPOS\_DateJoin | datetime |  | Ngày tham gia vào hệ thống |
| 6 | JPOS\_Email | varchar(100) |  | Email của Admin |
| 7 | JPOS\_LoginCount | int |  | Chứa số lần Login thất bại nếu quá 3 lần sẽ khóa tài khoản. |
| 8 | JPOS\_LastLogin | datetime |  | Lần đăng nhập cuối cùng. |

#### JPOS\_Card

Thông tin thẻ khách hàng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_CardID | varchar(16) | PK | Mã thẻ tài khoản trên jpos |
| 2 | JPOS\_ExpireDay | datetime | not null | Ngày hết hạn |
| 3 | JPOS\_Status | Int | FK | Trạng thái của thẻ hiện tại. |
| 4 | JPOS\_ActivatCode | varchar(16) |  | Mã kích hoạt của thẻ |
| 5 | JPOS\_CustomerID | int | FK | Mã khách hàng sở hữu thẻ |

#### JPOS\_Customer

Thông tin khách hàng trên hệ thống của JPOS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_CustomerID | int | PK | Liên kết khóa ngoại đến hệ thống khách hàng của database nopCommerce |
| 2 | JPOS\_FirstName | nvarchar(50) | not null | Họ của khách hàng |
|  | JPOS\_Lastname | nvarchar(5) |  | Tên của khách hàng |
| 3 | JPOS\_Address | nvarchar(200) |  | Địa chỉ của khách hàng |
| 4 | JPOS\_Email | varchar(100) |  | Email của khách hàng. |
| 5 | JPOS\_DateJoin | datetime |  | Ngày khách hàng tham gia vào hệ thống. |
| 6 | JPOS\_Birthday | datetime |  | Ngày sinh của khách hàng |
| 7 | JPOS\_Gender | bit |  | Giới tính của khách hàng |
| 8 | JPOS\_Favourite | nvarchar(100) |  | Sở thích của khách hàng. |
| 9 | JPOS\_CurrentPoint | int | Not null | Số điểm hiện tại của khách hàng trên hệ thống |

#### JPOS\_Event

Tạo sự kiện cho hệ thống. Các giao dịch diễn ra trong thời gian sự kiện sẽ được c6ng điểm dựa trên tác vụ của giao dịch và số điểm được công dựa trên luật của giao dịch.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_EventID | int | PK | Mã của sự kiện |
| 2 | JPOS\_EventName | nvarchar(100) |  | Tên của sự kiện |
| 3 | JPOS\_DateStart | datetime | not null | Ngày bắt đầu sự kiện |
| 4 | JPOS\_DateEnd | datetime | not null | Ngày kết thúc sự kiện |
| 5 | JPOS\_EventMessage | nvarchar(200) |  | Thông điệp quảng cáo cho sự kiện |
| 6 | JPOS\_TaskID | int | FK | Mã của tác vụ mà sự kiện áp dụng |
| 7 | JPOS\_RuleID | int | FK | Mã của luật mà sự kiện áp dụng. |

#### JPOS\_Gift

Lưu trữ thông tin về các món quà cho phép khách hàng đổi điểm lấy quà.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_GiftID | int | PK | Mã món quà tặng |
| 2 | JPOS\_GiftName | nvarchar(200) | Not null | Tên món quà tặng |
| 3 | JPOS\_PointForGift | int | Not null | Số điểm để đổi lấy món quà |
| 4 | JPOS\_Status | int | FK | Đánh dấu xóa. |

#### 

#### JPOS\_Issuer

Bảng này chứa thông tin của nhà phát hành thẻ.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_IssuerID | int | PK | Mã của nhà phát hành thẻ. |
| 2 | JPOS\_IssuerName | nvarchar(100) |  | Tên nhà phát hành thẻ |
| 3 | JPOS\_IssuerAddress | nvarchar(200) |  | Địa chỉ nhà phát hành thẻ |
| 4 | JPOS\_DateFound | datetime |  | Ngày sáng lập. |

#### JPOS\_LOG

Lưu trữ thông tin các tác vụ đã thực hiện trên hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_IDLog | Int | Not null | Mã thao tác với hệ thống đã thực hiện được log lại |
| 2 | JPOS\_Date | Datetime | Not null | Ngày thực hiện thao tác |
| 3 | JPOS\_Task | Int | Not null | Mã thao tác thực hiện (liên kết khóa ngoại với bảng Task liệt kê các thao tác cho phép thực hiện trên hệ thống) |
| 4 | JPOS\_CardID | varchar(16) | FK | Mã thẻ thự hiện giao dịch. |
| 5 | JPOS\_PointGain | Int | Not null | Số điểm point đạt được khi thực hiện thao tác của khách hàng |
| 6 | JPOS\_PointLoss | Int | Not null | Số điểm point mất đi khi thực hiện thao tác của khách hàng. |
| 7 | JPOS\_TID | Varchar(8) | Not null | Terminal ID (xem thêm trong Spec draft) |
| 8 | JPOS\_MID | Varchar(15) | Not null | Identify code (Xem thêm trong Spec draft) |
| 9 | JPOS\_PoSCC\_ID | Varchar(2) | Not null | Các thiết bị mà hệ thống hỗ trợ. |

#### JPOS\_Log\_Event

Bảng này lưu lại thông tin của giao dịch liên quan đến các sự kiện.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_LogID | int | FK | Tham chiếu đến kháo chính trong bảng JPOS\_Log |
| 2 | JPOS\_EventID | int | FK | Mã sự kiện.Tham chiếu đến khóa chính torng bảng JPOS\_Event |
| 3 | JPOS\_PointGain | int |  | Số điểm đạt được |
| 4 | JPOS\_PointLoss | int |  | Số điểm mất đi |

**Khóa chính trong bảng này (JPOS\_LogID,JPOS\_EventID)**

#### JPOS\_Log\_Exchange

Thông tin Log các tác vụ thực hiện đổi điểm lấy quà

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_LogID | int | PK | Liên kết khóa ngoại tới bảng Log, lưu trự thông tin đặc thù với việc đổi quà |
| 2 | JPOS\_Gift | int | Not null | Mã món quà được đổi |

#### JPOS\_Merchant

Bảng này chứa thông tin về các nhà sách có trong hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_MID | varchar(15) | Not null | Mã của nhà sách. |
| 2 | JPOS\_MerchantName | nvarchar(200) |  | Tên của nhà sách |
| 3 | JPOS\_Address | nvarchar(200) |  | Địa chỉ của nhà sách |
| 4 | JPOS\_IssuerID | int | FK | Mã của nhà phát hành thẻ. |

## 

#### JPOS\_PoSCC

Bảng này chứa thông tin về các loại thiết bị gởi thông điệp lên hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_PoSCC\_ID | Varchar(2) | Not null | Point Of Service Condition Code |
| 2 | JPOS\_PoSCC\_Name | Varchar(100) | Allow null | Mô tả thiết bị. |

## 

#### JPOS\_Rule

Bảng này chứa các luật để đổi từ tiền ra điểm.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_RuleID | int | PK | Mã của luật. |
| 2 | JPOS\_Money | float |  | Số tiền |
| 3 | JPOS\_Point | int |  | Số điểm ứng với số tiền |
| 4 | JPOS\_Description | nvarchar(200) |  | Mô tả về luật. |

#### JPOS\_Status

Bảng này chứa các trạng thái của thẻ,quà hoặc các thiết bị của hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_StatusID | int | PK | Mã của trạng thái |
| 2 | JPOS\_StatusName | varchar(50) |  | Mô tả về trạng thái. |

#### JPOS\_Task

Thông tin các tác vụ cho phép thực hiện trên hệ thống

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_IDTask | Int | Not null | Mã tác vụ cho phép thực hiện các nghiệp vụ trên hệ thống |
| 2 | JPOS\_TaskName | Nvarchar(200) | Not null | Tên tác vụ thực hiện |

#### JPOS\_Terminal

Bảng này chứa thông tin về các thiết bị EDC dùng cho hệ thống.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Kiểu | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | JPOS\_TID | int | PK | Mã của thiết bị |
| 2 | JPOS\_Status | int | FK | Trạng thái của thiết bị |
| 3 | JPOS\_PIN | varchar(16) |  | Mã PIN của thiết bị |
| 4 | JPOS\_RetryLimit | int |  | số lần tối đa để kích hoạt. |
| 5 | JPOS\_ActivateCode | varchar(16) |  | Mã kích hoạt |
| 6 | JPOS\_MID |  | FK | Mã của nhà sách nơi thiết bị hoạt động |

**Khóa chính của hệ thống (JPOS\_TID , JPOS\_MID)**

1. http://en.wikipedia.org/wiki/Extended\_Binary\_Coded\_Decimal\_Interchange\_Code [↑](#footnote-ref-2)
2. Trong biểu diễn người ta thường lắp đầy(padding) các vị trí dư ra bằng số 0 đối với số và khỏang trắng đối với kí tự [↑](#footnote-ref-3)
3. Hình trên được tham khảo từ trang 22 sách jPos-Programmer’s Guide 1.6.1 [↑](#footnote-ref-4)
4. Chi tiết về mẫu Composite tham khảo tại trang 183 Elements of Reusable Object-Oriented Software. [↑](#footnote-ref-5)
5. Việc đưa ra khái niệm dựa trên hiểu biết của tác giả về transaction [↑](#footnote-ref-6)
6. Hình 2-2 được tham khảo tại trang 29 sách jPos Programmer’s Guide. [↑](#footnote-ref-7)