**TÀI LIỆU THIẾT KẾ KIẾN TRÚC**

|  |  |
| --- | --- |
| **Dự án** | Dự án ORMS |
| **Phiên bản** | 1.0 |
| **Ngày** | 02/03/2021 |
| **Người viết** |  |

| **Thay đổi bản ghi** | | | |
| --- | --- | --- | --- |
| **Ngày** | **Người viết** | **Phiên bản** | **Nội dung sửa đổi** |
| 02/03/2021 | FSS | 1.0 | Tạo mới |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**ORMS**

**Tài liệu kiến trúc tổng thể hệ thống**

Contents

[**I.** **GIỚI THIỆU CHUNG** 2](#_Toc65681457)

[1. Mục tiêu của tài liệu 2](#_Toc65681458)

[2. Phạm vi sử dụng tài liệu 2](#_Toc65681459)

[3. Đối tượng sử dụng tài liệu 2](#_Toc65681460)

[4. Thuật ngữ viết tắt 2](#_Toc65681461)

[**I.** **SƠ ĐỒ KIẾN TRÚC HỆ THỐNG** 3](#_Toc65681462)

[**II.** **SƠ ĐỒ LUỒNG HOẠT ĐỘNG NGHIỆP VỤ CỦA HỆ THỐNG** 4](#_Toc65681463)

[1. Đồng bộ đầu ngày 4](#_Toc65681464)

[2. Giao dịch từ Flex Client 4](#_Toc65681465)

[3. Đặt lệnh từ Front 6](#_Toc65681466)

[4. Luồng tra cứu từ Front 7](#_Toc65681467)

[III. **CƠ CHẾ DỰ PHÒNG CỦA HỆ THỐNG** 7](#_Toc65681468)

# **GIỚI THIỆU CHUNG**

## Mục tiêu của tài liệu

Tài liệu này nhằm mô tả tổng quan kiến trúc của hệ thống ORMS

## Phạm vi sử dụng tài liệu

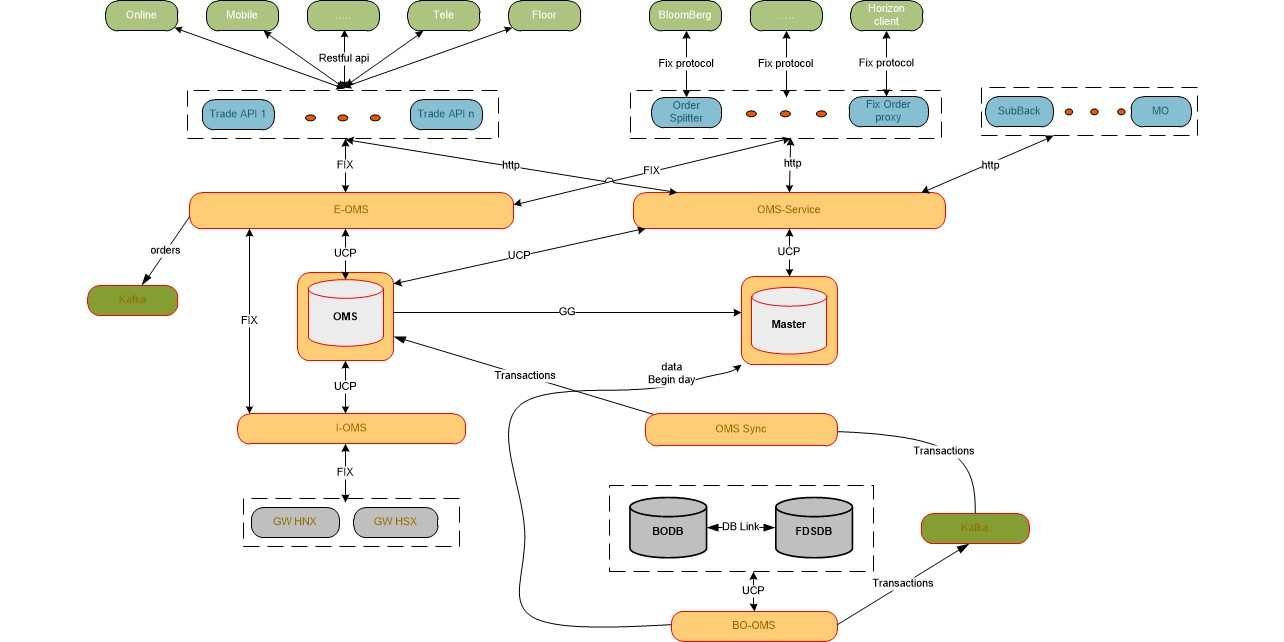
## Đối tượng sử dụng tài liệu

## Thuật ngữ viết tắt

|  |  |
| --- | --- |
| Tên thuật ngữ | Ý nghĩa |
| VND | Công ty chứng khoản VNDirect |
| HFT | Hệ thống đặt lệnh |
| CSDL | Cơ sở dữ liệu |
| Flex | Hệ thống back |
|  |  |

# **SƠ ĐỒ KIẾN TRÚC HỆ THỐNG**

## **Sơ đồ thiết kế hệ thống ORMS**



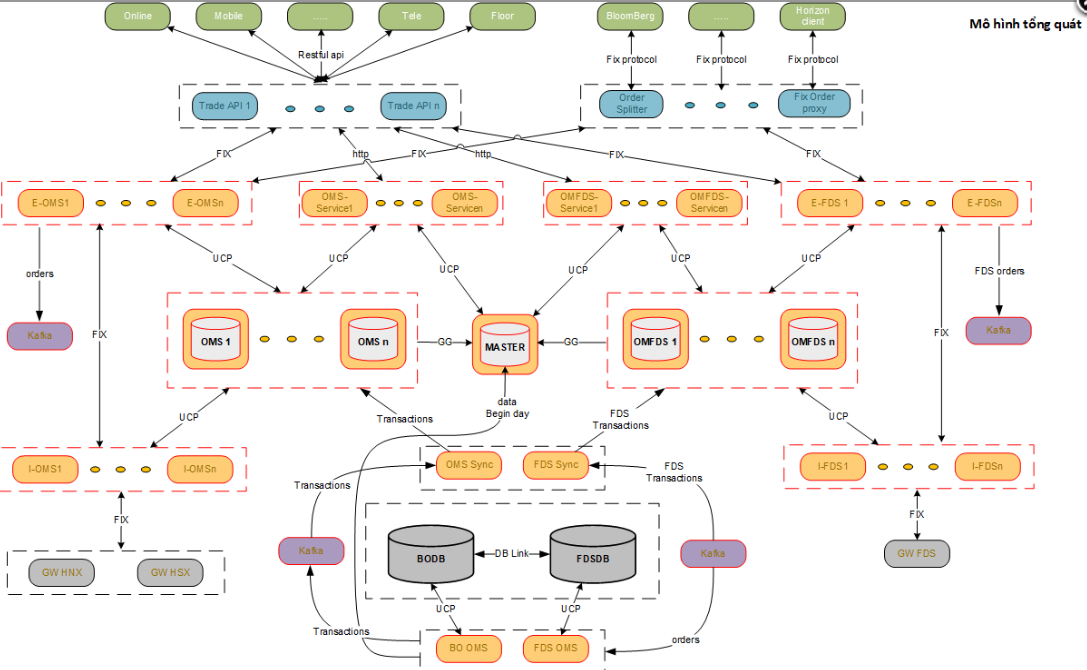
Gồm:

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Thành phần** | **Công nghệ** | **Mô tả** |
| 1 | OMS | Oracle 19c | Hệ thống quản lý lệnh tách biệt. OMS sử dụng công nghệ Oralce 19c in memory để tăng tốc độ xử lý và tính toán.  Để tăng hiệu suất hệ thống có thể dựng nhiều OMS và phân chia tập khách hàng giao dịch trên các OMS này |
| 2 | E-ORS | Java | E-ORS là service trung gian, dùng để tiếp nhận các yêu cầu về lệnh từ các kênh theo chuẩn FIX và Restful..  E-ORS có thể dựng 1 hay nhiều để giảm tải cho hệ thống, trong đó 1 E-ORS sẽ cần kết nối đến tất cả các subOMS |
| 3 | I-ORS | Java | I-ORS là ứng dụng trung gian, dùng để giao tiếp giữa gateway lệnh và hệ thống OMS. Có nhiệm vụ:  - Bốc lệnh lên Gateway  - Chuyển tiếp message từ gateway (sở) xuống OMS  I-ORS có thể dựng 1 hay nhiều để giảm tải cho hệ thống |
| 4 | Gateway HNX |  | Gateway. Hệ thống hiện có. |
| 5 | Gateway HSX |  | Gateway. Hệ thống hiện có. |
| 6 | Host Service | .Net | Host Service. Hệ thống hiện có. |
| 7 | BO@ | Oracle | Database BO@. Hệ thống hiện có. |
| 8 | BO@ client | .Net | BO@ client. Hệ thống hiện có. |
| 9 | subBack |  | Các hệ thống back quản trị tài sản |
| 10 | MO |  | Hệ thống quản trị rủi ro chung |
| 11 | AM |  | Hệ thống quản trị tài sản chung |
| 12 | FIX gateway | Java | Cổng giao tiếp với FIX Gateway để đặt lệnh. Hệ thống hiện có. |
| 13 | TradeAPI | Java | TradeAPI để đặt lệnh qua chuẩn restfull. Hệ thống hiện có |
| 14 | BO OMS | Java | Hệ thống hiện có, nhận nhiệm vụ đẩy dữ liệu đầu ngày từ BO@ vào database đồng bộ tập trung |
| 16 | Kafka |  | Là nơi lưu trữ các message dùng để đồng bộ từ hệ thống khác đến OMS và ngược lại |
| 17 | OMSSyn | Java | Ứng dụng dùng để đọc các message từ Kafka để đồng bộ dữ liệu vào hệ thống OMS |
| 18 | Data | Oracle | DB dùng để lưu trữ dữ liệu cần đồng bộ từ các subback, MO/AM… lên OMS |
| 19 | OMSService | Java | Service trung gian kết nối giữa OMS và các hệ thống khác, sử dụng để các hệ thống khác gọi đến giao dịch **khác lệnh** trên OMS (tiền, chứng khoán, thay đổi giá trị limit cho pool/room….)  Có thể scale thành n-ORSService để tăng performance hệ thống |
| 20 | OracleMaster | Oralce | Database goldengate dùng để lưu trữ sổ lệnh tập trung của các OMS |
| 21 | ORSadmin |  | Hệ thống dùng để quản lý log tập trung trên hệ thống ORSM. Có các chức năng cơ bản:  - Quản lý các message đồng bộ giữa hệ thống OMS với các cấu phần khác  - Cho phép đẩy lại các message manual trong 1 số trường hợp khi message đồng bộ bị xử lý lỗi |

Note:

* **Về việc sử dụng DB DATA để đồng bộ dữ liệu đầu ngày:**
* Đầu ngày, lượng dữ liệu cần đồng bộ từ các hệ thống subBack/AM/MO cần đồng bộ vào hệ thống OMS tương đối lớn nên không thể sử dụng chung luồng qua Kafka như trong ngày để đảm bảo hiệu năng hệ thống.
* Ngoài ra, Trong 1 số trường hợp đặc biệt, cần replay lại quá trình đồng bộ dữ liệu đầu ngày vào OMS, việc sử dụng db DATA sẽ giúp quá trình replay được dễ dàng hơn
* Ngoài việc cần đồng bộ dữ liệu từ các hệ thống khác đến OMS, sau này hệ thống sẽ phát sinh nhiều nhu cầu tổng hợp dữ liệu từ các subBack để gửi lên MO/AM, việc sử dụng DB cơ sở dữ liệu quan hệ sẽ giúp thuận tiện hơn trong quá trình tổng hợp dữ liệu từ nhiều nguồn
* VềCơ chế đẩy lệnh trực tiếp từ E-ORS lên I-ORS ( chi tiết luồng tham khảo mục ***3. Đặt lệnh từ Front):***
* Thay vì sử dụng cơ chế quét lệnh tại OMS để bốc lên I-ORS, E-ORS sau khi nhận được phản hồi đặt lệnh thành công từ OMS sẽ trực tiếp đẩy lệnh vào I-ORS nếu thỏa mãn điều kiện gửi lệnh
* Việc sử dụng giải pháp này nhằm tránh độ trễ khi sử dụng tiến trình job quét theo các giải pháp hiện tại đang sử dụng.
* Nếu trong quá trình đẩy lệnh từ I-ORS lên E-ORS gặp lỗi, lệnh sẽ được đẩy lên theo tiến trình quét lệnh tại OMS từ I-ORS ( dùng cho lệnh đặt trước giờ)
* Về cơ chế Reply khi có lỗi về đồng bộ:
* Hệ thống đồng bộ có 1 DB để log lại các message đồng bộ
* Ứng dụng quản trị ORS-Admin sẽ có chức năng quản lý các message đồng bộ này, hệ thống có chức năng manual cho phép đẩy lại các message lỗi để tổng hợp lại
* Chức năng đồng bộ trong hệ thống OMS sẽ có key chống trùng để đảm bảo 1 message không được xử lý 2 lần

## Mô hình tích hợp và triển khai với hệ sinh thái VNDS:



Hệ thống OMS sẽ kết nối với nhiều hệ thống subBack khác, cũng như hệ thống MO và AM để tổng hợp các dữ liệu cần thiết cho việc quản trị rủi ro và giao dịch chứng khoán

## **Sơ đồ tổng quan luồng đồng bộ như sau (chi tiết luồng đồng bộ xem mục *II.Đồng bộ dữ liệu*)**



## **Luồng cập nhật giá và phiên:**



## **Mô hình kết nối khi phân tải hệ thống vào nhiều E-ORS, I-ORS, ORSService, OMS:**



* Note:
  + 1 E-ORS cần kết nối đên tất cả cả các database Sub-OMS
  + 1 ORS-Service cần kết nối đến tất cả các database Sub-OMS
  + IORS và OMS kết nối với nhau theo cơ chế 1-1
  + NORS và OMS kết nối với nhau theo cơ chế 1-1

# **SƠ ĐỒ LUỒNG HOẠT ĐỘNG NGHIỆP VỤ CỦA HỆ THỐNG**

Một số luồng nghiệp vụ đặc trưng trong hệ thống.

## **Đồng bộ dữ liệu từ các hệ thống tới OMS**



**Dữ liệu đầu ngày tổng hợp vào OMS**

* Sau khi chạy Batch, MO/AM và các hệ thống subBack khác tổng hợp dữ liệu đầu ngày cho OMS vào hệ thống DATA tập trung thông qua service BOSync ( tiến trình 1,2)
* Đồng thời, BOsyn sẽ gửi 1 tín hiệu vào Kafka để OMS biết có event cần tổng hợp dữ liệu ( tiến trình 3)
* Hệ thống ORSService sau khi biết có event cần tổng hợp sẽ lấy dữ liệu từ DATA để chuyển tới OMS ( 7,5,6)

**Dữ liệu đầu ngày tổng hợp vào MO/AM và các hệ thống subBack**

* Sau khi chạy Batch, MO/AM và các hệ thống subBack khác tổng hợp dữ liệu đầu ngày cho OMS vào hệ thống DATA tập trung thông qua service BOSync ( tiến trình 1,2)
* Đồng thời, BOsyn sẽ gửi 1 tín hiệu vào Kafka để thông báo việc bắt đầu tổng hợp dữ liệu lên kafka cũng như **startOffset** của việc đồng bộ
* BOSyn đẩy dữ liệu vào kafka
* Các subBack và MO/AM lấy dữ liệu từ kafka để đồng bộ dữ liệu vào hệ thống
* **Dữ liệu trong ngày**
  + Trong ngày , các hệ thống subBack/MO/AM khi có nhu cầu đồng bộ thông tin lên OMS sẽ thông qua hệ thống BOSync để tổng hợp dữ liệu vào Kafka ( tiến trình 1, 3)
  + Hệ thống ORSService đọc dữ liệu từ Kafka và đẩy lên hệ thống OMS ( tiến trình 4,5,6)
  + Các hệ thống khác khi cần lấy thông tin đồng bộ trong ngày cũng đọc dữ liệu từ kafka giống như hệ thống OMS
* **Các vấn đề cần thảo luận:**

- Dữ liệu đẩy vào kafka theo cơ chế ghi tiếp vào topic cũ hay mở topic mới ?

- Kafka là nơi queue dữ liệu tập trung cho rất nhiều subBack, MO/AM, để tránh việc dữ liệu bị đồng bộ thừa đến các hệ thống không cần nguồn dữ liệu đó, FSS đề xuất 1 trong 2 giải pháp sau:

* Trên Kafka sẽ tổ chức thành những topic chung và topic riêng cho từng hệ thống, các hệ thống sẽ chỉ lấy dữ liệu từ topic chung và topic của riêng nó
* Trên Kafka sử dụng chung 1 topic, dữ liệu chia thành các objectType, các hệ thống dựa vào ObjectType để lọc dữ liệu cần đồng bộ

## **Giao dịch từ các hệ thống khác đến OMS**

Trong ngày, các hệ thống subBack/MO/AM (tạm gọi chung là các subBack) khi có phát sinh giao dịch làm thay đổi sức mua hoặc quản trị rủi ro, cần gọi lên hệ thống OMS để đồng bộ dữ liệu cũng như verify giao dịch.

Các giao dịch chia làm hai loại:

* Giao dịch trực tiếp làm giảm sức mua (Luồng Sync)
* Các giao dịch không trực tiếp làm giảm sức mua ( Luồng Async)
* **Giao dịch làm giảm sức mua từ BO@( Luồng Sync)**



|  |  |  |
| --- | --- | --- |
| STT | Thành phần luồng | Mô tả |
| 1 | Bước 1,2 | BO@ client gửi request giao dịch lên HostService. BO@ check các điều kiện làm giao dịch, sau đó HostService sẽ kiểm tra giao dịch này có cần gọi sang OMS hay không. Nếu cần thì sang bước 2, không cần thì sang bước 3. |
| 2 | Bước 3 | HostService gọi vào ORSService để tạo giao dịch. Nếu hạch toán không thành công thì trả về mã lỗi. Nếu thành công thì sang bước 4 |
| 3 | Bước 4 | HostService gọi vào database Flex để sinh giao dịch. |
| 4 | Bước 5 | DB BO@ thực hiện sinh giao dịch, trả kết quả về cho HostService. |
| 5 | Bước 6 | Nếu bước 5 có lỗi, HostService sẽ thực hiện rollback giao dịch trên DB OMS |
| 6 | Bước 7 | Trả kết quả về cho Client. |

* **Các giao dịch khác từ BO@ ( luồng Asyns)**

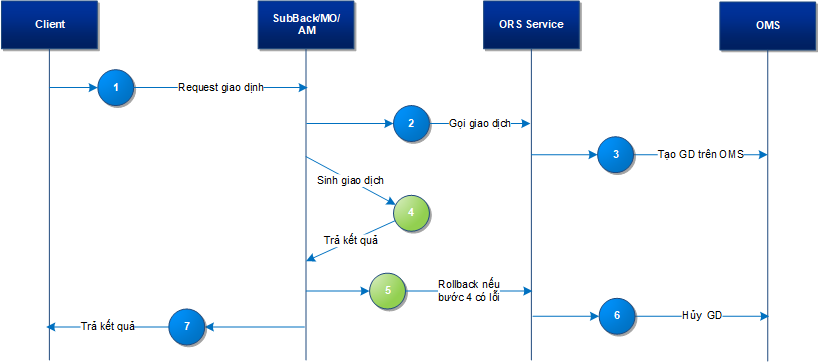
Các giao dịch gọi từ hệ thống khác mà không làm giảm sức mua của tiểu khoản sẽ được đi theo luồng Asyns để đẩy request lên OMS



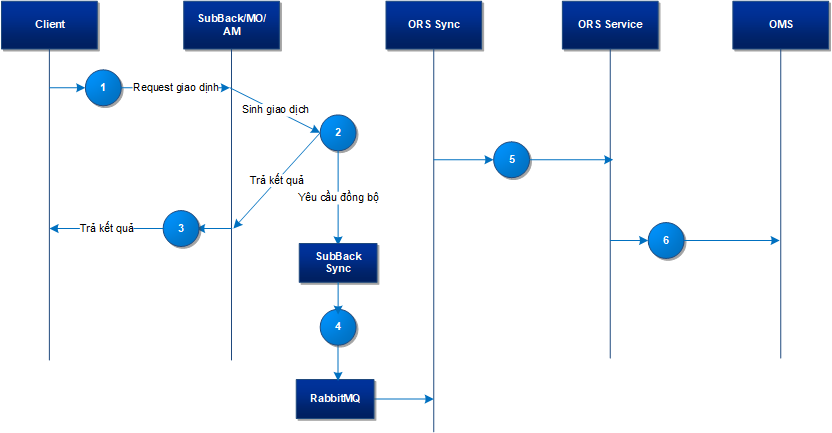
|  |  |  |
| --- | --- | --- |
| STT | Thành phần luồng | Mô tả |
|  | Bước 1 | Bo@ gửi request giao dịch lên HostService. |
| 2 | Bước 2 | HostService gọi vào database Flex để sinh giao dịch. |
| 3 | Bước 3 | DB Flex thực hiện sinh giao dịch, trả kết quả về cho HostService đồng thời ghi nhận lại yêu cầu đồng bộ sang OMS. |
| 4 | Bước 4 | Trả kết quả về cho Client. |
| 5 | Bước 5 | BO Sync sẽ ghi nhận request để đẩy message giao dịch vào Kafka |
| 6 | Bước 6 | ORSSyn đọc dữ liệu từ Kafka để đẩy vào ORSService |
| 7 | Bước 7 | ORSService gọi giao dịch cập nhật ở OMS, gửi phản hồi thành công hay thất bại về phía DB BO@ thông qua N-ORS để đẩy vào Kafka |

**Note**: Các giao dịch từ các kênh SubBack/MO/AM cũng gọi theo cơ chế tương tự theo 2 luồng Sync và Async, thông qua ORS Service theo sơ đồ sau :

* Luồng Sync: các giao dịch phong tỏa sức mua, phong tỏa tài sản từ subBack chuyển sang OMS….:



* Luồng Async: Các giao dịch cập nhật hạn mức Pool/Room trong ngày…..:



## **Đặt lệnh từ Front**



|  |  |  |
| --- | --- | --- |
| STT | Thành phần luồng | Mô tả |
| 1 | Bước 1 | Front send request đặt lệnh vào TradeAPI. |
| 2 | Bước 2 | TradeAPI call E-ORS |
| 3 | Bước 3 | **E-ORS call Thủ tục đặt lệnh tại OMS.** |
| 4 | Bước 4 | OMS tiến hành đặt lệnh |
| 5 | Bước 6,8 | Lệnh đặt nếu thành công, sẽ được đồng bộ xuống FLEX |
| 6 | Bước 7 | Sau khi đặt lệnh thành công vào OMS, nếu thỏa mãn điều kiện sẽ được đẩy thẳng từ E-ORS vào I-ORS |
| 7 | Bước 8 | I-ORS sẽ đẩy lệnh lên Gateway HSX/HNX tương ứng. |
| 8 | Bước 8 | Gateway trả message confirm/ khớp lệnh về cho I-ORS. |
| 9 | Bước 9 | I-ORS trả message confirm/ khớp lệnh về cho OMS. |
| 10 | Bước 10 | I-ORS thực hiện khớp lệnh. |
| 11 | Bước 11 | I-ORS đồng bộ lệnh khớp sang DB BO@ thông qua N-ORS để đẩy vào Kafka |
| 12 | Bước 12 | DB BO@ dựa vào tín hiệu đồng bộ lệnh khớp để sinh giao dịch khớp lệnh. |

Tại bước 7: Nếu như lệnh không thỏa mãn điều kiện về phiên E-ORS sẽ không đẩy lệnh lên

I-ORS service. I-ORS sẽ có tiến trình để quét các lệnh trạng thái chờ gửi trên DB OMS, các phiên cần queue lệnh trước giờ sẽ có tham số thời gian bốc lệnh trước giờ để queue lệnh sớm lên Gateway (VNDS confirm lại nội dung này????)

## **Luồng tra cứu từ Front**



|  |  |  |
| --- | --- | --- |
| STT | Thành phần luồng | Mô tả |
| 1 | Bước 1 | Front gửi request đến TradeAPI. TradeAPI phải router xem request này được gọi sang ORSservice/ FLEX/ ORS cũ, |
| 2 | Bước 2 | Nếu là truy vấn lệnh và sức mua, TradeAPI chuyển tiếp request đến ORSService để request đến OMS. |
| 3 | Bước 3 | OMS thực hiện call thủ tục query data. |
| 4 | Bước 7,8 | ORSServiec gửi phản hồi lại cho TradeAPI |
| 5 | Bước 4 | TradeAPI call BOAPI |
| 6 | Bước 5,6 | BOAPI call thủ tục dưới BO@ |
| 7 | Bước 6 | BOAPI và TradeAPI gửi phản hồi lại cho front |

## **5. Luồng lệnh thỏa thuận**

### 5.1. Lệnh thỏa thuận 1 firm

* **Gửi lệnh lên Gateway**



|  |  |  |
| --- | --- | --- |
| **STT** | **Thành phần luồng** | **Mô tả** |
| 1 | Bước 1 | Client sinh lệnh thỏa thuận 1 firm rồi gửi lên EORS |
| 2 | Bước 2 | EORS gửi lệnh lên subOMS tương ứng với 2 tài khoản mua và bán |
| 3 | Bước 3,4 | Sau khi nhận phản hồi thành công từ phía 2 subOMS, EORS gửi lệnh thỏa thuận lên IORS ( tương ứng gắn với OMS của khách hàng) để gửi lên Gateway |
| 4 | Bước 5 | IORS gửi phản hồi lệnh đã gửi lên Gateway cho EORS |
| 5 | Bước 6 | EORS sau khi nhận tín hiệu lệnh đã bốc lên Gateway thì cập nhật trạng thái lệnh đang gửi vào các subOMS tương ứng |

* **Lệnh thỏa thuận đặt trước giờ**

Bốc lệnh thỏa thuận trước giờ tương tự cơ chế của lệnh thướng đặt trước giờ:

* Tiến trình trên IORS quét các lệnh đặt trước giờ xem đã thỏa mãn điều kiện bốc lệnh lên sở chưa
* Nếu thỏa mãn điều kiện thì bốc lệnh từ OMS lên gateway
* **Confirm lệnh và confirm khớp từ Gateway**

Theo SPEC mới của dự án HSX04, qui trình lệnh thỏa thuận 1 firm và 2 firm là giống nhau, khi bên yêu cầu gửi yêu cầu lệnh thỏa thuận lên sở, sở sẽ gửi lại message confirm cho bên gửi, đồng thời gửi thêm message yêu cầu xác nhận lệnh cho bên đối ứng (\*)

Hệ thống OMS chia thành các subOMS theo tập khách hàng, nên khi nhận message yêu cầu xác nhận (\*) từ sở, sẽ không có cơ sở để định tuyến message vào subOMS nào.

🡪 **Giải pháp đề xuất:** Trên các IORS, qui định 1 IORS mặc định (có thể tách riêng cho lệnh thỏa thuận để không ảnh hưởng hiêu năng lệnh thường) để nhận tất cả các message phản hồi lệnh thỏa thuận từ sở.

I-ORS mặc định này sẽ kết nối đến tất cả các subOMS và cache được tiểu khoản theo tag OMS ( sử dụng cho việc router)

Trong các subOMS qui định 1 sub OMS default dành cho thỏa thuận ( có thể dùng chung với subOMS của lệnh thường)



|  |  |  |
| --- | --- | --- |
| STT | Thành phần luồng | Mô tả |
| 1 | Bước 1 | Tất cả các Message liên quan đến lệnh thỏa thuận từ sở được gửi từ Gateway đến 1 **IORS default** dùng cho lệnh thỏa thuận |
| 2 | Bước 2 | IORS chuyển tiếp message đến subOMS tương ứng nếu là message confirm cho yêu cầu xuất phát từ 1 subOMS nào đó  IORS chuyển tiếp message đến subOMS default nếu như message không phải là confirm cho 1 yêu cầu từ subOMS |
| 3 | Bước 3 | Dữ liệu liên quan đến lệnh thỏa thuận sau khi được cập nhật trên subOMS sẽ được tự động đồng bộ lên sổ lệnh tập trung (OracleMaster)  Chức năng xác nhận lệnh thỏa thuận từ client sẽ query dữ liệu từ sổ lệnh tâp trung này |

### 5.2. Lệnh thỏa thuận 2 firm

Luồng lệnh thỏa thuận 2 firm xuất phát từ công ty tương tự như luồng thỏa thuận 1 firm ở trên

Luồng lệnh thỏa thuận 2 firm xuất phát từ đối tác như sau:



|  |  |  |
| --- | --- | --- |
| **STT** | **Thành phần luồng** | **Mô tả** |
| 1 | Bước 1 | Message từ sở được gửi từ Gateway đến **1 IORS default dùng cho lệnh thỏa thuận** |
| 2 | Bước 2 | IORS chuyển tiếp message đến subOMS default |
| 3 | Bước 3 | Dữ liệu được đồng bộ đến db Oracle Master |
| 4 | Bước 4 | Client lệnh thỏa thuận lấy dữ liệu từ OralceMaster cho chức năng xác nhận lệnh thỏa thuận  Client lệnh thỏa thuận gửi yêu cầu để confirm lệnh thỏa thuận 2 firm lên hệ thống OMS thông qua luồng lệnh thông thường |

# **CƠ CHẾ DỰ PHÒNG CỦA HỆ THỐNG**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Cấu phần** | **Cơ chế HA** | **Giải pháp HA** | **Ghi chú** |
| 1 | HFT | Active – Active | Cluster Database |  |
| 2 | BPS Service | Active – Active | F5 |  |
| 3 | Exchange Bridge | Active – Standby |  |  |
| 4 | Gateway HNX | Active – Standby |  |  |
| 5 | Gateway HSX | Active – Standby |  |  |
| 6 | Front | Active – Active | F5 |  |
| 7 | Host Service | Active – Active | F5 |  |
| 8 | Flex | Active – Active | Cluster Database |  |

# **MÔ HÌNH TRIỂN KHAI**

## **Cơ sở hạ tầng hệ thống UAT**



Đề xuất cơ sở hạ tầng máy chủ cho hệ thống UAT như sơ đồ trên. Cấu hình chi tiết các máy chủ.

|  |  |
| --- | --- |
| **Máy chủ APP** | |
| Số lượng | 05 Server. Bao gồm M1, M2, M3, M4, M8 (Có thể sử dụng máy ảo) |
| CPUs | 08 |
| Hệ điều hành | Unix/Linux |
| RAM | 16 GB |
| Dung lượng ổ cứng để cài đặt phần mềm và lưu log | 100 GB Available |

|  |  |
| --- | --- |
| **Máy chủ Database** | |
| Số lượng | 03 Server. Bao gồm M5, M6, M7 (Sử dụng máy vật lý) |
| CPUs | 24 |
| Hệ điều hành | Unix/Linux |
| RAM | 32 GB |
| Dung lượng ổ cứng lưu trữ database | 500 GB Available |

*Chú ý:*

* + *Khi thực hiện Load Test thì sẽ tách riêng các ứng dụng của máy APP (M1,2,3,4) ra nhiều mày đơn để đảm bảo hiệu năng.*
  + *Giai đoạn UAT1 (test tích hợp EORS và IORS) chưa cần các server M7, M8*

## **Cơ sở hạ tầng hệ thống Production**